



ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

**ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ
ΥΠΟΛΟΓΙΣΤΩΝ**

**Πρόγραμμα Μεταπτυχιακών Σπουδών
Επιστήμη και Τεχνολογία της Πληροφορικής και
των Υπολογιστών**

Ειδίκευση Λογισμικού και Πληροφοριακών Συστημάτων,

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

**Internet technologies for the development of
compatible applications on different computing
platforms**

**Αντώνιος-Διονύσιος Ν. Παύλοζας
Α.Μ. MCSE19019**

Εισηγητής: Δρ Νικόλαος Ζάχαρης, Καθηγητής

Αιγάλεω, 2021

Internet technologies for the development of compatible applications on different
computing platforms

(Κενό φύλλο)

Internet technologies for the development of compatible applications on different
computing platforms

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

**Internet technologies for the development of
compatible applications on different computing
platforms**

**Αντώνιος-Διονύσιος Ν. Παύλοζας
Α.Μ. MCSE19019**

Εισηγητής:

Δρ Νικόλαος Ζάχαρης, Καθηγητής

Εξεταστική Επιτροπή:

Δρ Νικόλαος Ζάχαρης, Καθηγητής

Δρ Γεώργιος Πρεζεράκος, Καθηγητής

Δρ Αντώνης Μπόγρης, Καθηγητής

Ημερομηνία εξέτασης: 04/11/2021

Internet technologies for the development of compatible applications on different
computing platforms

(Κενό φύλλο)

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Αντώνιος-Διονύσιος του Νικολάου, με αριθμό μητρώου mcse19019 φοιτητής του Προγράμματος Μεταπτυχιακών Σπουδών Επιστήμης της Τεχνολογίας και Πληροφορικής Υπολογιστών του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών της Σχολής Μηχανικών του Πανεπιστημίου Δυτικής Αττικής, δηλώνω ότι:

«Είμαι συγγραφέας αυτής της μεταπτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών



Αντώνιος-Διονύσιος Παύλοζας

Internet technologies for the development of compatible applications on different computing platforms

(Κενό φύλλο)

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου κύριο Νικόλαο Ζάχαρη για την βοήθεια και την άμεση ανταπόκριση που μου παρείχε κατά την διάρκεια εκπόνησης της διπλωματικής εργασίας. Ο κ. Ζάχαρης, κατ' εμέ, δεν είναι απλά ένας αξιότιμος καθηγητής, αλλά ένας διδάσκαλος, καθώς με συμβούλευσε καθ' όλη την διάρκεια συγγραφής της εργασίας και μέσω της καθοδήγησης, των σχολίων και των παρατηρήσεών του κατόρθωσα να βελτιώσω, όσο το δυνατόν περισσότερο, την εργασία μου.

Ακόμα θα ήθελα να ευχαριστήσω του καθηγητές μου κ. Γεώργιο Πρεζεράκο και κ. Αντώνη Μπόγρη που δέχθηκαν να είναι στην τριμελή επιτροπή και για το ενδιαφέρον και τον χρόνο που μου αφιέρωσαν.

Internet technologies for the development of compatible applications on different
computing platforms

(Κενό φύλλο)

Περίληψη

Στις μέρες μας όλο και περισσότεροι άνθρωποι χρησιμοποιούν τα κινητά τους τηλέφωνα πολύ περισσότερο από οποιαδήποτε άλλη συσκευή. Έτσι, κάθε εταιρεία λογισμικού επενδύει στην ανάπτυξη εφαρμογών για κινητά. Αναμφίβολα, ορισμένοι οργανισμοί μπορούν να επικεντρωθούν σε ένα μόνο λειτουργικό σύστημα για κινητά και να αποφύγουν όλα τα άλλα, ωστόσο είναι σημαντικό για πολλές επιχειρήσεις να επικεντρωθούν σε μια μυριάδα φορητών συσκευών με διάφορα λειτουργικά συστήματα. Μία από τις πιο δύσκολες καταστάσεις για τους προγραμματιστές εφαρμογών είναι εάν θα αναπτύξουν μια εγγενή ή μια εφαρμογή για κινητά σε πολλές πλατφόρμες. Html, JavaScript, CSS είναι μερικές από τις γλώσσες προγραμματισμού, που χρησιμοποιούνται για τη δημιουργία εφαρμογών στο Διαδίκτυο, μέσω των οποίων δημιουργούνται οι κατάλληλες διεπαφές για την αλληλεπίδραση του χρήστη με τις υπηρεσίες ενός διακομιστή. Τα τελευταία χρόνια, ο αριθμός των πακέτων λογισμικού (πλαισίων) που βασίζονται στις παραπάνω γλώσσες αυξάνεται συνεχώς και προσφέρουν τη δυνατότητα ανάπτυξης συμβατών εφαρμογών σε πολλά λειτουργικά συστήματα και φορητές συσκευές.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η παρουσίαση επτά πολυεπίπεδων πλαισίων προκειμένου να περιγραφεί το είδος της διαθεσιμότητάς τους και οι υπηρεσίες που προσφέρουν, τα πλεονεκτήματα και τα μειονεκτήματα της χρήσης τους και οι παράμετροι εγκατάστασης σε διαφορετικά λειτουργικά συστήματα. Συγκεκριμένα, θα γίνει ανάλυση στα Xamarin, Apache Cordova (πρώην Phonegap), Ionic, React Native, Flutter, Framework7 και NativeScript. Επιπλέον, παρουσιάζεται ένα παράδειγμα κώδικα για κάθε πολυεπίπεδο πλαίσιο και στο τέλος γίνεται σύγκριση για το ποιο είναι το καλύτερο σύμφωνα με δύο μεθόδους πολυκριτηριακής ανάλυσης αποφάσεων.

Λέξεις-Κλειδιά: Cross-platform application, Mobile frameworks, MCDA, AHP, TOPSIS

Abstract

Nowadays more and more people are using their mobile phones much more than any other device. Thus, every software company invests in the development of mobile applications. Undoubtedly, some organizations can concentrate on only one mobile operating system and avoid all the others, yet it is important for many businesses to focus on a myriad of mobile devices with various operating systems. One of the most challenging situations for app developers is whether to develop a native or a cross-platform mobile app. Html, JavaScript, CSS are some of the languages, which are used to create applications on the Internet, through which the appropriate interfaces for the user's interaction with the services of a server are created. In the recent years, the number of software packages (frameworks) based on the above languages is constantly increasing and offer the possibility of developing compatible applications on many operating systems and mobile devices.

The purpose of this thesis is the presentation of seven frameworks in order to describe the type of their availability and the services they offer, the advantages and disadvantages of their use and the environment setup on different operating systems. Particularly, an analysis will be made for the Xamarin, Apache Cordova (formerly Phonegap), Ionic, React Native, Flutter, Framework7 and NativeScript cross-platform frameworks. In addition, an example code for each framework is presented and at the end a comparison is made as to which is the best framework according to two methods of multi-criteria decision analysis.

Keywords: Cross-platform application, Mobile frameworks, MCDA, AHP, TOPSIS

Abbreviations and Acronyms

AHP	Analytic Hierarchy Process
API	Application Programming Interface
APK	Android Package
C.I.	Consistency Index
CLI	Command-line Interface
CMD	Command Prompt
C.R.	Consistency Ratio
CSS	Cascading Style Sheet
DHTML	Dynamic HyperText Markup Language
DM	Decision Maker
DOM	Document Object Model
Etc.	et cetera, a Latin expression meaning "and the other things"
GCC	GNU Compiler Collection
GUI	Graphical User Interface
HTML	Hypertext Markup Language
IBM	International Business Machines
ICC	Intel C++ Compiler
IDE	Integrated Development Environment
I/O	Input / Output
JIT	Just-In-Time
JS	JavaScript
JSON	JavaScript Object Notation
J2EE	Java 2 Platform, Enterprise Edition
MOC	Meta-Object Compiler
MCDA	Multi-criteria Decision Analysis
MSVC	Microsoft Visual C++
MVC	Model-View-Controller
MVVM	Model-View-View Model
NIS	Negative Ideal Solution
ORM	Object-Relational Mapping
OS	Operating System
PIS	Positive Ideal Solution
QML	Qt Modeling Language
RI	Random Index
SDK	Software Development Kit
TS	TypeScript
UI	User Interface
UWP	Universal Windows Platform
UX	User Experience
VM	Virtual Machine
VS	Visual Studio Code
VSIX	Visual Studio extension for Tizen packaging
WoSCC	Web of Science Core Collection
XAML	Extensible Application Markup Language
XML	Extensible Markup Language
2D	Two-Dimensional

List of Figures

Figure 1: Sales of smartphones (2007-2021), Source: https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/	6
Figure 2: Sales of computers in Great Britain (2015-2021), Source: https://www.statista.com/statistics/523294/computer-retail-sales-volume-quarterly-index-in-great-britain/	7
Figure 3: Internet usage: smartphones-desktops (2019-2020), Source: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=File:People_who_used_mobile_devices_to_access_the_internet_away_from_home_or_work_EU-27_2019_(%25)_BYIE20.png	7
Figure 4: Usage of operating systems, Source: https://www.statista.com/statistics/1078678/software-development-operating-system-mobile/	8
Figure 5: Google Play and App Store comparison, Source: https://buildfire.com/app-statistics/	9
Figure 6: Native application architecture, Source: [11]	11
Figure 7: Web application architecture, Source: [11]	12
Figure 8: Hybrid application architecture, Source: [11]	14
Figure 9: Comparison of Native, Hybrid and Web application development, Source: https://betterprogramming.pub/native-hybrid-or-web-apps-what-is-the-best-approach-for-lasting-success-91afbb872d89	16
Figure 10: Advantages of cross – platform development, Source: https://www.netsolutions.com/insights/cross-platform-app-frameworks-in-2019/	18
Figure 11: Comparison of native and cross-platform development, Source: https://www.simform.com/native-vs-cross-platform-development/	19
Figure 12: Xamarin logo, Source: https://commons.wikimedia.org/wiki/File:Xamarin-logo.svg	20
Figure 13: Xamarin native architecture, Source: [23]	21
Figure 14: Xamarin.Forms architecture , Source: [23]	21
Figure 15: App.xaml.cs	26
Figure 16: App.xaml	26
Figure 17: Xamarin app	27
Figure 18: Apache Cordova logo, Source: https://cordova.apache.org/artwork/	28
Figure 19: Apache Cordova architecture, Source [28]	28
Figure 20: Apache Cordova directory	31
Figure 21: Apache Cordova index.html code	32
Figure 22: Apache Cordova index.js script	33
Figure 23: Apache Cordova results	33
Figure 24: Ionic logo, Source: https://commons.wikimedia.org/wiki/File:Ionic_Logo.svg	34
Figure 25: Ionic architecture, Source: [37]	35
Figure 26: Ionic CLI message installation	37

Internet technologies for the development of compatible applications on different
computing platforms

Figure 27: Ionic project's name	38
Figure 28: Ionic app slide.....	39
Figure 29: Ionic styling.....	40
Figure 30: Ionic app.....	40
Figure 31: React Native logo, Source: https://developers.pendo.io/react-native-plugin-2-6-1-ios-android-beta/	41
Figure 32: React Native architecture, Source: [44]	42
Figure 33: React Native first page code.....	46
Figure 34: React Native barcode	47
Figure 35: React Native app	48
Figure 36: Flutter logo, Source: https://commons.wikimedia.org/wiki/File:Google-flutter-logo.png	49
Figure 37: Flutter architecture, Source: [50].....	49
Figure 38: FlutLab	54
Figure 39: Flutter app code	56
Figure 40: Apk message.....	56
Figure 41: Apk installation.....	57
Figure 42: Flutter app	57
Figure 43: Framework7 logo, Source: https://framework7.io/	58
Figure 44: Framework7 subnavbar example, Source: [59]	60
Figure 45: Subnavbar Framework7, a) iOS, b) Android, Source: [59].....	61
Figure 46: NativeScript logo, Source: https://dwglogo.com/nativescript/	62
Figure 47: NativeScript architecture, Source: [60]	62
Figure 48: NativeScript runtime, Source: [61].....	63
Figure 49: NativeScript first page code	65
Figure 50: NativeScript first page functionality.....	66
Figure 51: NativeScript second page	66
Figure 52: NativeScript second page functionality.....	67
Figure 53: NativeScript barcode	67
Figure 54: NativeScript app	68
Figure 55: Most used programming languages, Source: https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/	73
Figure 56: AHP, hierarchical structure tree	77
Figure 57: AHP, scale of relative importance, Source: [65].....	78
Figure 58: AHP, Random Index, Source: [67]	82
Figure 59: Diagram of weights.....	91

List of Tables

Table 1: Cross-platform search analysis	2
Table 2: Cross-platform frameworks search analysis.....	3
Table 3: Xamarin system requirements, Source: [22]	22
Table 4: Xamarin prerequisites, Source: [22]	24
Table 5: Famous Xamarin apps, Source: [27]	27
Table 6: Apache Cordova system requirements, Source: [28]	29
Table 7: Apache Cordova prerequisites, Source: [34]	30
Table 8: Famous Apache Cordova apps, Source: [33]	33
Table 9: Ionic system requirements, Source: [38].....	35
Table 10: Ionic prerequisites , Source: [38].....	37
Table 11: Famous Ionic apps, Source: [41].....	41
Table 12: React Native system requirements, Source: [46]	43
Table 13: React Native prerequisites, Source: [42]	45
Table 14: Famous React Native apps, Source: [48]	48
Table 15: Flutter system requirements, Source: [53].....	51
Table 16: Flutter prerequisites , Source: [50].....	53
Table 17: Famous Flutter apps, Source: [55].....	58
Table 18: Framework7 system requirements, Source: [58]	59
Table 19: Framework7 installation commands, Source: [57].....	60
Table 20: Famous Framework7 apps, Source: [59]	61
Table 21: NativeScript system requirements, Source: [62].....	63
Table 22: Famous NativeScript apps, Source: [64].....	68
Table 23: Cross-platform's attributes matrix	70
Table 24: DM's attributes ranking	71
Table 25: DM's optimal value of attributes.....	72
Table 26: Programming Language conversion	74
Table 27: Mobile OS conversion.....	75
Table 28: Price conversion.....	75
Table 29: Playground conversion	75
Table 30: Code Reuse conversion.....	76
Table 31: Transformed numeric matrix.....	76
Table 32: AHP, attribute's pair-wise matrix	78
Table 33: AHP, sum of attribute's pair-wise matrix	79
Table 34: AHP, normalized each column.....	79
Table 35: AHP, normalized attribute's pair-wise matrix	80
Table 36: AHP, attribute's weights matrix.....	80
Table 37: AHP, multiplication of weights of each column.....	81
Table 38: AHP, weighted sum matrix	81
Table 39: AHP, ratio matrix	81
Table 40: AHP, <i>Code Reuse</i> pair-wise comparison matrix.....	83
Table 41: AHP, weights of alternatives in respect to <i>Code Reuse</i> attribute.....	83
Table 42: AHP, <i>Code Reuse</i> consistency matrix.....	83

Internet technologies for the development of compatible applications on different
computing platforms

Table 43: AHP, <i>Code Reuse</i> formula matrix.....	84
Table 44: AHP, normalized alternatives in respect to attributes	84
Table 45: AHP, weighted sum matrix	85
Table 46: AHP, ranking matrix	85
Table 47: TOPSIS, normalized division of each column.....	86
Table 48: TOPSIS, normalized matrix	87
Table 49: TOPSIS, multiply columns with the weights of each attribute	87
Table 50: TOPSIS, weighted normalized matrix	88
Table 51: TOPSIS, PIS & NIS matrix.....	88
Table 52: TOPSIS, separation distance matrix.....	90
Table 53: TOPSIS, ranking matrix	90
Table 54: Total weights of attributes	91
Table 55: Total ranking	91

Table of Contents

Abstract.....	X
Abbreviations and Acronyms.....	XI
List of Figures.....	XII
List of Tables.....	XIV
Chapter 1: Introduction.....	1
1.1 Problem Description.....	1
1.2 Purpose.....	1
1.3 Motivation.....	2
1.4 Methodology.....	3
1.5 Thesis Outline.....	3
Chapter 2: Mobile Application Development Approaches.....	5
2.1 History of Smartphones.....	5
2.2 Smartphones Computation.....	6
2.3 Application Development of Smartphones.....	9
2.3.1 Native Applications.....	9
2.3.2 Web Applications.....	11
2.3.3 Hybrid Applications.....	13
2.4 Choosing Between the Three Types of Mobile Apps.....	14
2.5 Cross – platform Development.....	16
2.5.1 Cross – platform Advantages.....	17
2.5.2 Cross – platform Disadvantages.....	18
2.6 Native vs. Cross – platform: When we choose each one.....	18
Chapter 3: Cross-platform Frameworks Analysis.....	20
3.1 Xamarin.....	20
3.1.1 Advantages of Xamarin.....	22
3.1.2 Disadvantages of Xamarin.....	23
3.1.3 Xamarin App.....	24
3.2 Apache Cordova.....	27
3.2.1 Advantages of Apache Cordova.....	29
3.2.2 Disadvantages of Apache Cordova.....	29
3.2.3 Apache Cordova App.....	30
3.3 Ionic.....	34

Internet technologies for the development of compatible applications on different
computing platforms

3.3.1 Advantages of Ionic.....	35
3.3.2 Disadvantages of Ionic.....	36
3.3.3 Ionic App.....	36
3.4 React Native.....	41
3.4.1 Advantages of React Native	43
3.4.2 Disadvantages of React Native	44
3.4.3 React Native App	45
3.5 Flutter	49
3.5.1 Advantages of Flutter	51
3.5.2 Disadvantages of Flutter	52
3.5.3 Flutter App	52
3.6 Framework7.....	58
3.6.1 Advantages of Framework7	59
3.6.2 Disadvantages of Framework7	59
3.6.3 Framework7 App	60
3.7 NativeScript	61
3.7.1 Advantages of NativeScript	64
3.7.2 Disadvantages of NativeScript	64
3.7.3 NativeScript App	65
Chapter 4: Multi-criteria Decision Analysis of Best Cross-platform Framework.....	69
4.1 Similarities	69
4.2 Differences.....	69
4.3 Multi-criteria Decision Analysis	70
4.3.1 Decision Maker	71
4.3.2 Scale of Language Features.....	72
4.3.3 Analytic Hierarchy Process (AHP)	76
4.3.4 Technique for Order Preference by Similarity to Ideal Solution (TOPSIS).....	85
4.4 Results	91
Chapter 5: Conclusion	93
References.....	95

Chapter 1: Introduction

In recent decades, smartphones are an integral part of most people's daily lives and can be characterized as probably the most important invention of the twenty-first century. At the same time there are many different smartphones on the market with different operating systems. Among the most popular operating systems are Android, iOS and Windows. However, this diversity in operating systems is a challenge for businesses to develop applications targeting all or most operating systems from a single code base.

Thus, the applications for smartphones (mobile apps) are in a stage where they have to exist on multiple platforms in order to reach out as many users as possible. Developing in native languages for multiple platforms requires more resources for software development companies, which will also result in higher expenses. Through cross-platforming, developers can create applications for multiple platforms using the same code base. Although there are several cross-platform application development kits, it is still a challenge for business or developers to choose which development kit will suffice all their application functionality and user experience requirements.

1.1 Problem Description

With so many different functionalities in mobile devices and mobile platforms, it is difficult for developers to create applications that work on all devices without adaption and customization of the application code. Developers need to know what are the possibilities and limitations that exist in the different technologies that are available in order to apply correct methods when developing mobile apps. This is important for both application users and the companies that develop the applications. Restricting the use of an application to a particular device is no longer desirable. So, choosing the wrong method can become a costly and time-consuming effort.

Since the main hindrance is having to develop a complete mobile application separately for each platform, new technologies have emerged that allow developers to share parts of the codebase between platforms, and which refer to as cross-platform frameworks. These so-called cross-platform technologies promise to solve the issues of developing native mobile applications in different operating system, but currently there is no clear consensus on whether they fulfill that promise.

1.2 Purpose

The main goal of this thesis is to provide a comparison study on several widely used cross-platform application development kits. More specifically, the cross-platform frameworks that will be analyzed are the Xamarin, Apache Cordova, Ionic, React Native, Flutter, Framework7 and NativeScript.

The above frameworks present important similarities, but also differences that must be considered before starting the implementation of applications in them. Hence, this thesis aims at answering the following two main questions.

RQ1: What are the main similarities and differences of these frameworks?

RQ2: Can multi-criteria decision analysis methods be implemented in order to discover the best cross-platform framework?

Answering the above two research questions will have a significant impact as it aims to help developers and companies to make the right choice of development framework of an app.

1.3 Motivation

Doing a search in two web pages that contained scientific articles revealed a lack of research articles in the specific field of cross-platform development. Particularly, based on Web of Science Core Collection (WoSCC)¹ and Scopus², a document search was carried out in order to find the degree of research that has been done in the field of cross-platform frameworks in the last five years. The survey was conducted in two stages.

In the first stage the term “cross-platform” was searched as the title of articles as well as the terms “framework” and “app” in the form of topic (title, abstract, author keywords). Then, the year of publication is defined, which is from 2016 to 2021 and the search results are displayed in the table below.

Cross-Platform (2016-2021)		
Keywords	Web of Science	Scopus
cross-platform	19	37
framework		
app		

Table 1: Cross-platform search analysis

In the second stage a search was made on each framework separately. Each keyword was used as a title and for the keywords “Cordova”, “Ionic” and “Flutter” the term “cross-platform” was also used in the form of topic to extract a better and

¹ Web of Science is the world's leading scientific citation search and analytical information platform. It is used as both a research tool supporting a broad array of scientific tasks across diverse knowledge domains as well as a dataset for large-scale data-intensive studies. More details can be found in <https://clarivate.com/webofsciencegroup/solutions/web-of-science/>

² Scopus is the largest abstract and citation database of peer-reviewed literature- scientific journals, books and conference proceedings. More details can be found in <https://www.scopus.com/search/form.uri?display=basic#basic>

more specific results, which are shown in the table below.

Cross-Platform (2016-2021)		
Keywords	Web of Science	Scopus
Xamarin	3	6
Cordova	4	4
Ionic	1	4
React Native	5	7
Flutter	2	1
Framework7	0	0
NativeScript	1	1

Table 2: Cross-platform frameworks search analysis

The conclusion drawn from the above tables confirms the fact that very few research articles have been written and therefore further research is required, which marked the motivation for this thesis.

1.4 Methodology

Scientific articles, books as well as websites were used for the purposes and information of the objectives of the thesis. In general, an effort to use bibliography of the last five years was made. It should also be noted that the statistical surveys that have been conducted and are held on the web pages of Statista³ and Statistics Explained⁴ were used. Lastly, through these statistical studies a multi-criteria decision analysis (MCDA) is performed based on two methods, Analytic Hierarchy Process (AHP)⁵ and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)⁶, through which the final ranking of each framework is presented.

1.5 Thesis Outline

The arrangement of the chapters in this thesis is set up to guide the reader through the various steps that are necessary for the proper understanding of the

³ Statista is an online portal providing data on the global digital economy, industrial sectors, consumer markets, public opinion, media, demography and macroeconomic trends. More details can be found in <https://www.statista.com>

⁴ Statistics Explained is an official website presenting statistical topics in an easily understandable way. More details can be found in https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Main_Page.

⁵ AHP is a structured technique for organizing and analyzing complex decisions, using a pairwise comparison approach to allow a more accurate ordering of priorities for decision making.

⁶ TOPSIS is based upon the concept that the chosen alternative should have the shortest distance from the Positive Ideal Solution (PIS) and the furthest from the Negative Ideal Solution (NIS). The final ranking is obtained by means of the closeness index.

outcome of this research. After the introduction, the rest of the thesis is organized as follows.

- *Chapter 2: Mobile Application Development Approaches* presents the four mobile application approaches: native, web, hybrid and cross-platform.
- *Chapter 3: Cross-platform Frameworks Analysis* provides a detailed review of seven selected cross-platform frameworks: Xamarin, Apache Cordova, Ionic, React Native, Flutter, Framework7 and NativeScript.
- *Chapter 4: Multi-criteria Decision Analysis of Best Cross-platform Framework* is divided in two parts. The first part refers to the similarities and differences of the seven frameworks, and the second part and the second part offers a multi-criteria decision analysis for selecting the best framework based on six attributes.
- *Chapter 5: Conclusion* provides a summary of the thesis highlighting the achieved goals.

Chapter 2: Mobile Application Development Approaches

Mobile application development is the process of creating software applications that run on mobile devices, most commonly for Android and iOS operating systems. The software can be preinstalled on the device, downloaded from a mobile app store or accessed through a mobile web browser and until now, there are millions of apps available in different mobile application stores. This ever-increasing mobile application usage is driving businesses and solo developers to invest their time and money targeting these users. Such strong demand has led businesses and developers to find new ways of mobile application development to target massive amounts of users with less development cost for applications.

There are four major development approaches when building mobile applications native, web, hybrid and cross-platform mobile applications. Each of these approaches has its own set of advantages and disadvantages. When choosing the right development approach for a project, developers need to consider the desired user experience, the computing resources and native features required by the app, the development budget, time targets, and resources available to maintain the app.

This chapter provides an overview of all four approaches, starting with an introduction to the history of smartphones.

2.1 History of Smartphones

The first smartphone developed in 1994 by IBM. It was called Simon Personal Computer and it has a touch screen and several apps. The most important feature was that it could send and receive emails. However, IBM decided not to proceed to further development as it was really innovative for its time [1].

Blackberry was another company that played an important role in the smartphone industry and particularly in 1996 when it developed the interactive pagers. The great advantage over the Simon Personal Computer was the display of the emails in the pager as soon as they arrived. When they embedded the pager on a cell phone in 2002, conquered the business world and automatically became one of the best mobile companies.

Nevertheless, in 2007 another company, the Apple, entered the world of mobile phones with the iPhone. iPhone combined many services such as music, mobile phone and Internet. Also, it had larger, better quality screen display and a well-designed user interface (UI) than other mobiles in that period. Its use was simpler and aimed at the everyday consumer in relation to Blackberry. Apple developed a special operating system for the iPhone called iOS and a native Software Development Kit (SDK).

At the end of 2007, Google released its first smartphone with an operating system, which is called Android, with the aim to approach the general consumer

Internet technologies for the development of compatible applications on different computing platforms

market [2]. Google tried to improve the capabilities of the smartphones by adding more features such as email, Internet access, music, games and camera and at the same time to keep the cost at minimum. Since 2008, Android has seen many upgrades which have gradually improved the operating system, adding new features and fixing bugs from previous versions. These improvements mainly concerned the display quality, the durability of a battery without charging and the user interface (UI).

From 2008 until now, Android dominates smartphone market since the operating system is an open source and anyone can built an application and take benefits of any features [3].

In today's technological era, big enterprises like Google and Apple can shape the future and they have already begun to lay the foundation by implementing foldable smartphones that they have the benefits of a smartphone and a tablet in one device [4].

2.2 Smartphones Computation

Smartphones are now an integral part of the society and everyday life. Millions of people from every corner of the world enjoying the increased capabilities they offer, capabilities that they did not even think about a few years ago. Every year more and more people get smartphones, either for their first time, either to replace their old one. The image below shows a statistical survey on smartphone sales with a forecast of the sales of 2021.

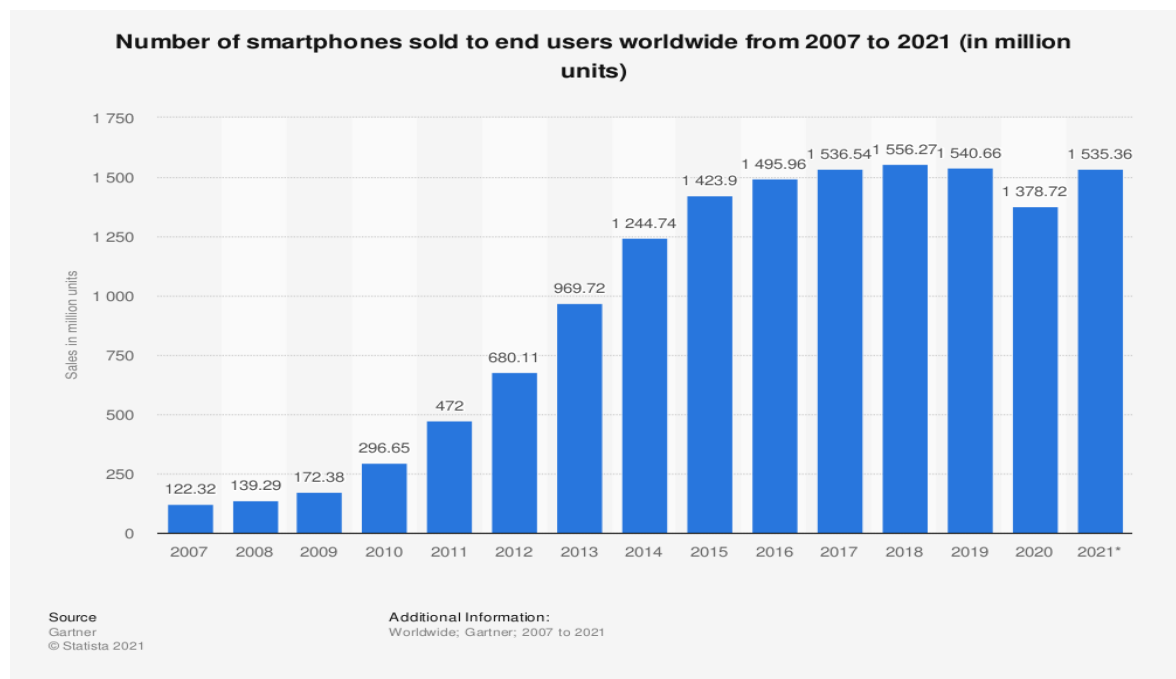


Figure 1: Sales of smartphones (2007-2021),

Source: <https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/>

Internet technologies for the development of compatible applications on different computing platforms

As shown above, the smartphone industry is characterized by new product proliferation where it can even surpass the computer industry. The image below clearly shows the decrease in computer sales in Great Britain until the second quarter of 2020, and an upward increase from the third quarter of 2020, which is normal as e-learning increased due to the global pandemic of Covid-19.

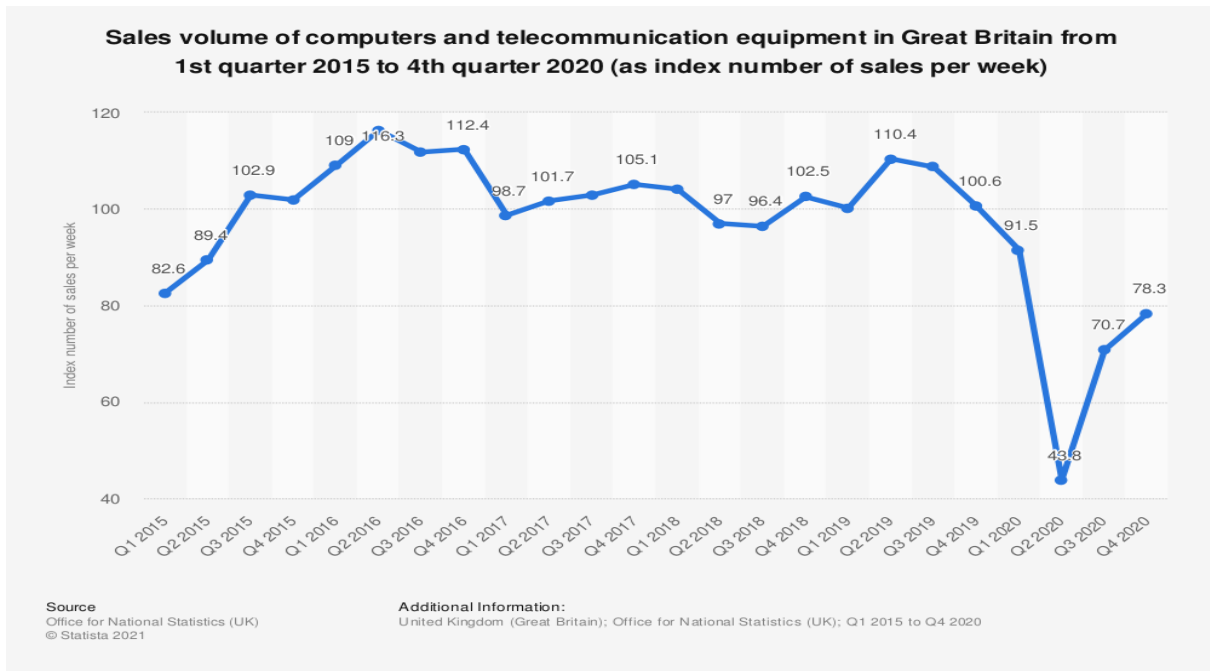
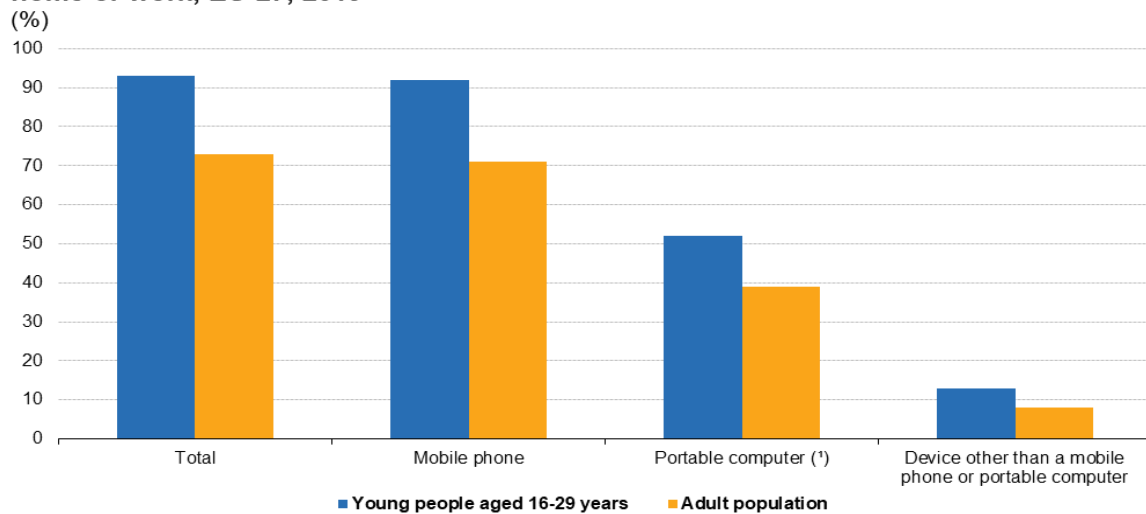


Figure 2: Sales of computers in Great Britain (2015-2021),

Source: <https://www.statista.com/statistics/523294/computer-retail-sales-volume-quarterly-index-in-great-britain/>

People who used mobile devices to access the internet away from home or work, EU-27, 2019



(*) Laptop, notebook, netbook or tablet computer.
Source: Eurostat (online data codes: isoc_ci_im_i)

eurostat

Figure 3: Internet usage: smartphones-desktops (2019-2020),

Source: <https://ec.europa.eu/eurostat/statistics->

Internet technologies for the development of compatible applications on different computing platforms

[explained/index.php?title=File:People who used mobile devices to access the internet away from home or work, EU-27, 2019 \(%25\) BYIE20.png](#)

Figure 3 show that the use of smartphone has exceeded the use of computer. There are several reasons why this happens. Firstly, smartphones offer benefits such as mobility and security to owners [5]. They contain personal information of the owner, as desktops, but they are accessible only by the owner and not from anyone who has access on the desktop [6]. Secondly, they use the radio spectrum rather than the physical infrastructures, such as phone wires. Lastly, a smartphone gives the ability to connect “on-the-go” and with a small consumption in power, the individuals enjoy their connection for a lot of hours daily [2].

Based on the above, all the attention has fallen to smartphones and the framework that they use. It is a fact that the enterprises are trying to find the best app development framework [7]. According to the Statista, 45% of the businesses invest in developing an application with a cross-platform technology, while 41% develop a native application for the Android store, 11% for the Apple Store and only 2% of them develop a native application in another app stores.

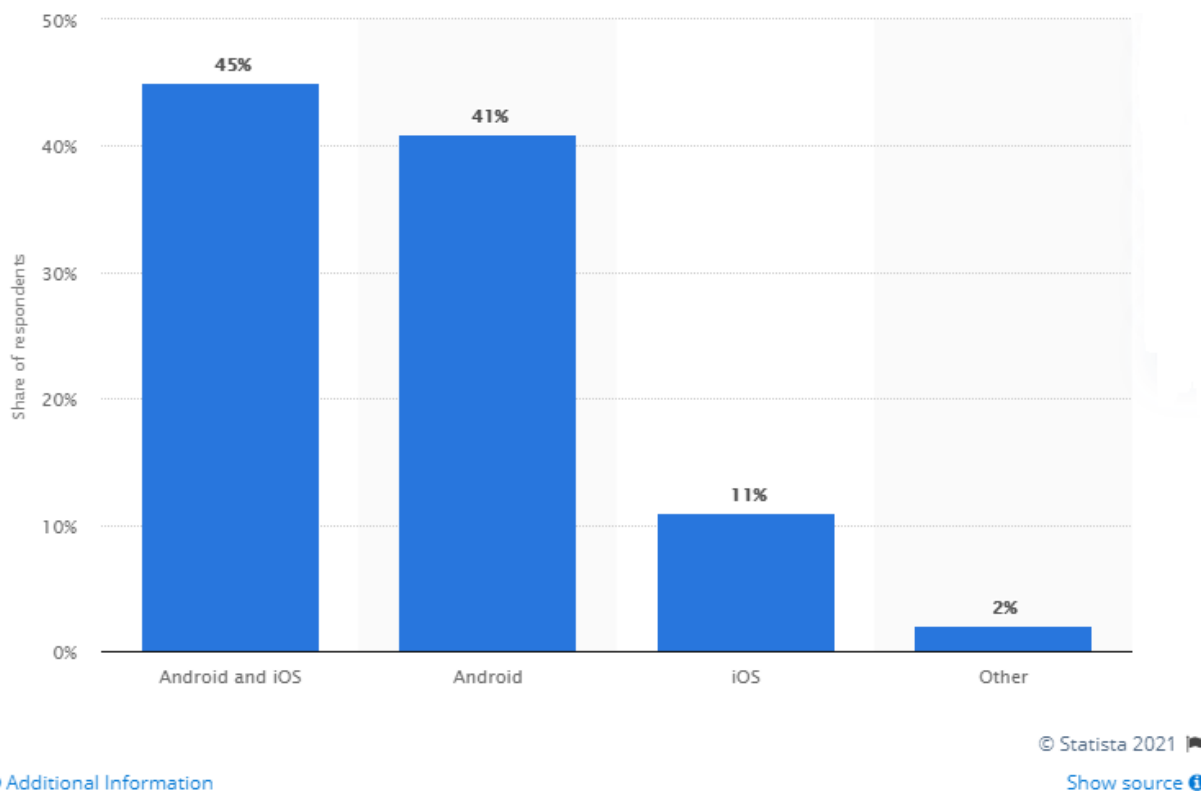


Figure 4: Usage of operating systems,
Source: <https://www.statista.com/statistics/1078678/software-development-operating-system-mobile/>

In the third quarter of 2020, Google Play Store holds the rein with the most applications downloads.

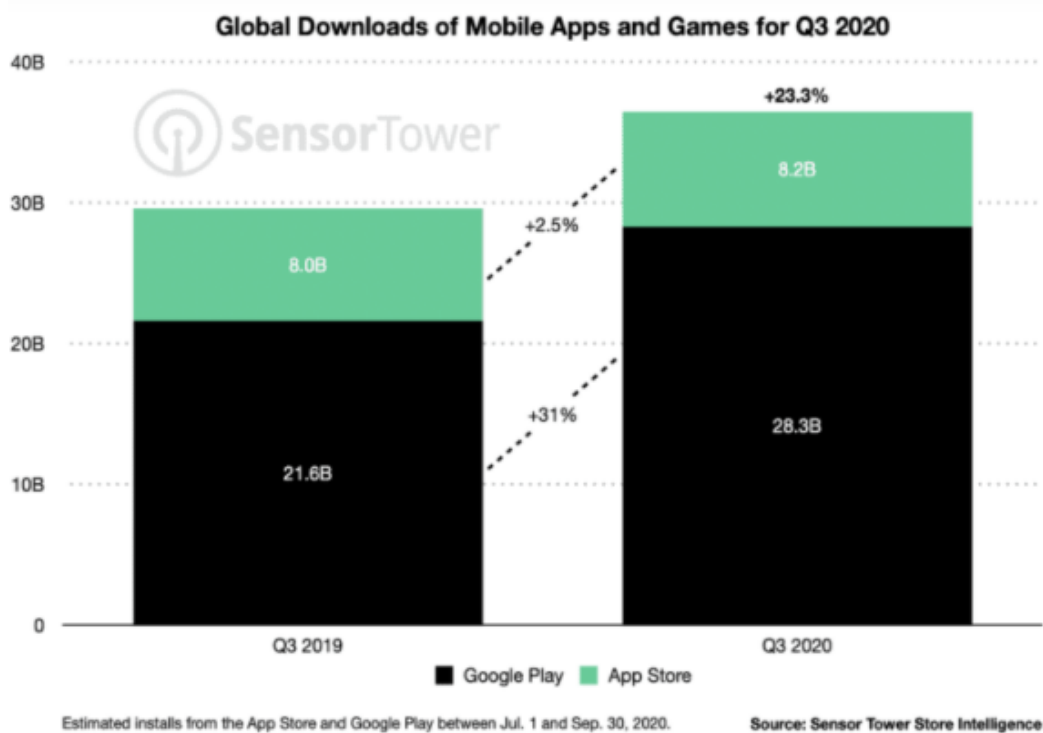


Figure 5: Google Play and App Store comparison, Source: <https://buildfire.com/app-statistics/>

The above statistics are something that well-established enterprises or start-up companies take into great consideration before start building an application [7].

2.3 Application Development of Smartphones

In mobile devices development there are three types of application that can be developed:

- ◆ Native Applications
- ◆ Web-Based Applications
- ◆ Hybrid Applications

Each development has its own disadvantages and advantages and every developer chooses according to their needs what kind of application they will develop. However, when people say “mobile app”, they refer to a native app.

2.3.1 Native Applications

An application is defined as a native when it is created for a specific operating system and it is available to users through the online store, which has the respective company. The native application can be installed on the device through the store, via a computer or via the Internet depending the operating system.

The nature of native applications allows the full exploitation of the hardware,

Internet technologies for the development of compatible applications on different computing platforms

sensors and software of mobile devices [8], and hence, these applications are faster than other types of applications. Native applications depending on their subject matter can run offline, but in many cases, they need an internet connection to download data and enable the user to process this data in a second time without an internet connection [9] [10].

To create a native application, the developer must write the source code in the programming language supported by the device's operating system and add information about the operation of the application, if necessary, such as audio files. Using tools provided by the company that created the operating system, the above files are compiled and a file is created which is the application that will be stored on the device [9] [10]. These tools combined with the additional facilities, are the application development environment (SDK) intended for each operating system. The process of developing a native application has similarities between the different operating systems, but the development environments (SDK's) differ as they are designed to meet the needs of a particular operating system and provide different tools [8].

Another set of tools available in native applications is the graphical user interface. The operating system provides a set of the basic graphical components such as buttons, menu, tap bars, notifications and much more. Applications that make use of these graphical components inherit the appearance of the specific operating system, in which it is installed resulting in the experience that the user receives to be within the framework to which he is accustomed [9] [10].

It is important to note that each operating system consists of its own particular graphical tools. Even between different versions of the same operating system, there are differences. The graphical interface is a very important part for the success of the operating system, resulting in continuous improvements, and therefore changes.

The programming interfaces required for the full utilization of the mobile device and consequently of the operating system, are specific and closely connected with the operating system for which they are created [8]. This adds complexity and cost to the development of a native application with a presence in different operating systems, since each operating system requires the development of the same application using different programming interfaces. Nevertheless, their role is particularly important as they are the ones that make possible the development of quite complex applications [10].

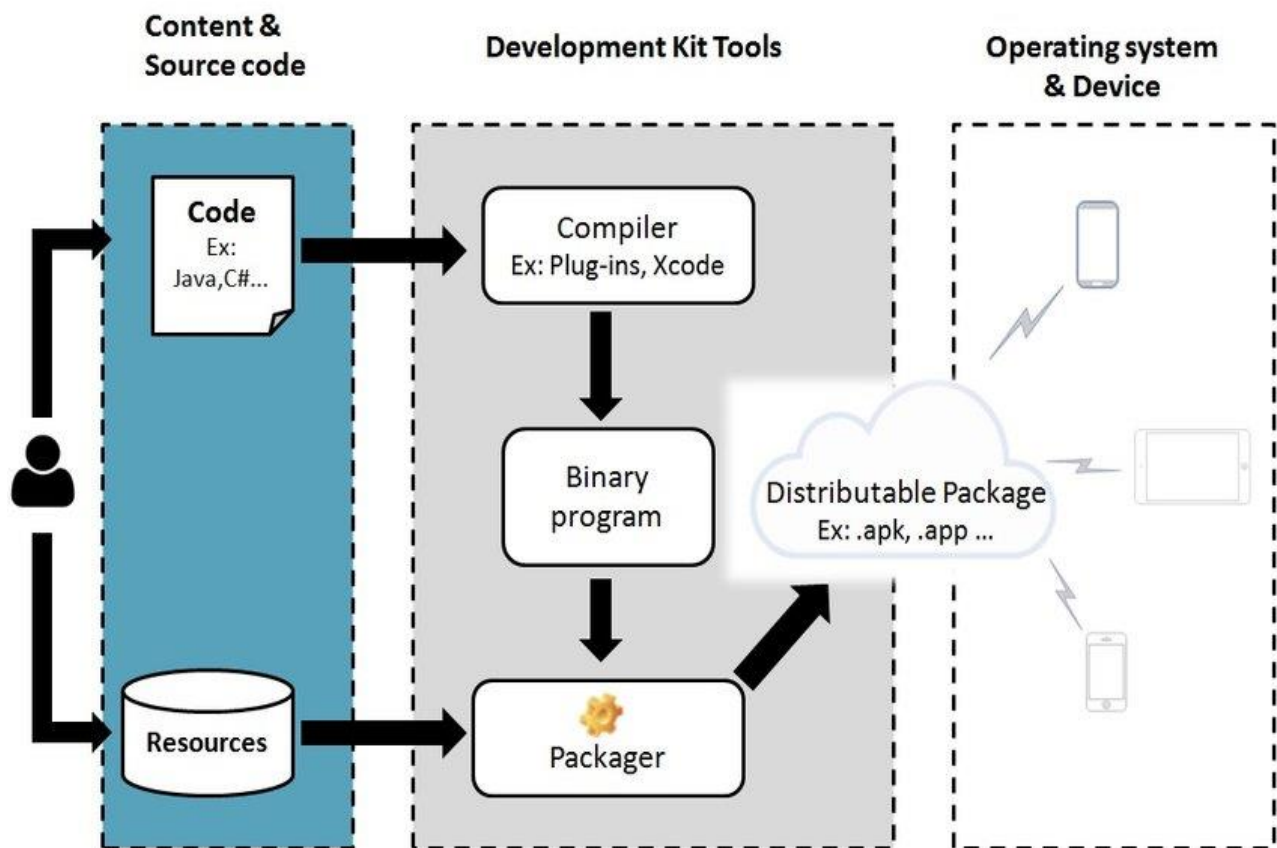


Figure 6: Native application architecture, Source: [11]

2.3.2 Web Applications

In the field of the development of the native applications for mobiles devices, the magnitude of the fragmentation was noticeable and so the specific category comes to give a solution to this problem. By choosing the web applications, it means that a user will use the browser in his device to see his site. One of the most important examples is the email applications (email), such as Google Gmail or Microsoft Outlook, which they use the browser for their visualization and they have been implemented with web technologies. Web applications can run on different operating systems and devices using web technologies (HTML, CSS and JavaScript). In this way developers avoid the obstacle of learning new programming languages for each different operating system [8].

The fact that the field of web applications already has several development tools does not meant that they can be used in the field of mobile devices without further changes and without further development. Mobile devices have special features that make existing technologies inadequate [9] [10].

The real starting point of the web applications was the advent of HTML5 and the constant evolution of browsers who can now take advantage of these new features. This technology alone is not capable of changing the application development

landscape on mobile devices, but several articles and references use the HTML5 name as a whole, not individually, to include the new version of CSS3 and especially the JavaScript programming interfaces (JavaScript APIs) that accompany it [9]. The term HTML5 has ceased to define another version of HTML and it now includes a suite of tools that enable the development of valuable applications and it is these tools that enable web applications to play a very important role in the development of applications on mobile appliances [9] [10].

At this point, it should be noted that unlike native applications, which have the operating system as their execution environment and are therefore connected to it, web applications run in the browser of the device. The web browser is also a native application, so it has direct access to the operating system APIs, but only a few of them are available in the applications that run on it, after examining its functions and checking if it has features that could be harmful to the device or to the sensitive personal data of the user [8].

In web applications this control is absent since they are available to any server. The second reason is the lack of implementation of various programming interfaces resulting in the lack of access to various functions of the device. As mentioned earlier, web applications depend on the course of HTML5 and especially the JavaScript APIs it supports [10]. Many of them have not yet been implemented or are not fully supported by browsers and not to mention that nowadays, as more and more sites use HTML5, the distinction between web apps and regular web pages has become blurry [9].

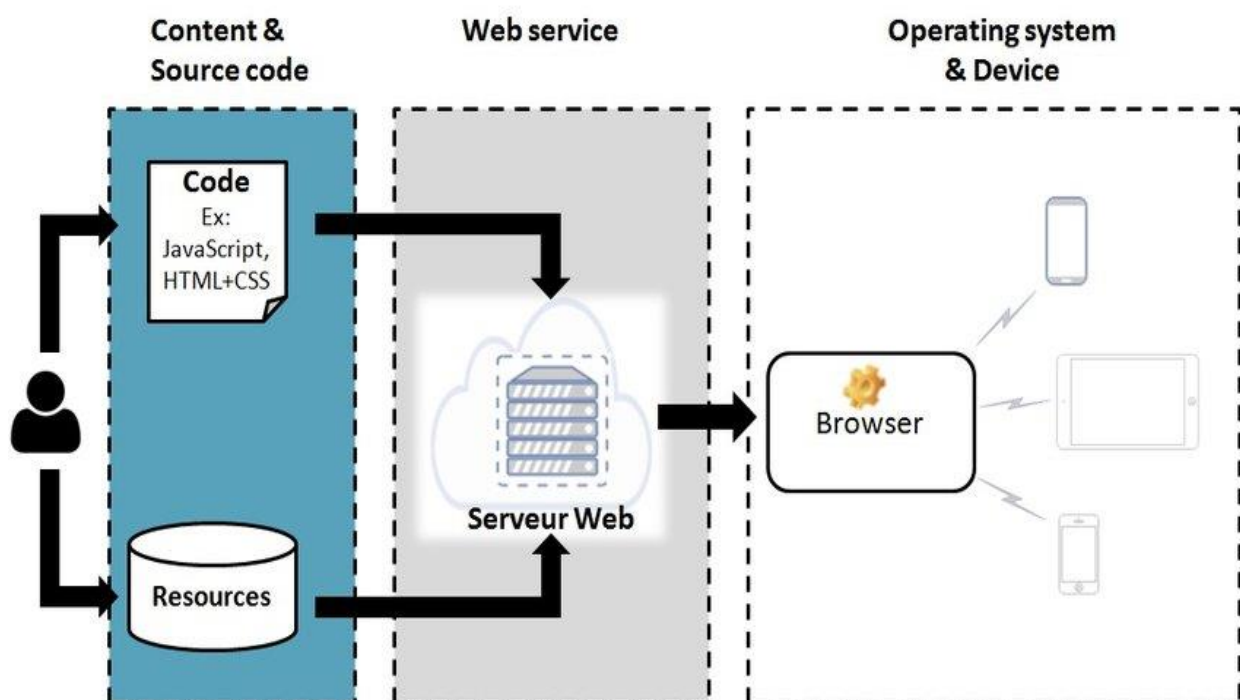


Figure 7: Web application architecture, Source: [11]

2.3.3 Hybrid Applications

Hybrid applications combine features from both of the above categories. More precisely, they are part of the native applications and part of the web applications. Following this approach, developers have the ability to implement the main functions of the application using web technologies and at the same time maintain access to device's features available only to native applications through the APIs that act as a bridge between the main functions of the application and the features of the device [9] [10] .

Hybrid applications are quite popular because they allow cross-platform development and thus, they reduce significantly the development costs. In a few words, the reuse of the part of the application that was implemented in web technologies is achieved, while all that needs to be modified for the application to work in other operating system is the change of the native APIs, which are used for the functions of the device [12]. The developers of the application have the ability to use ready-made tools such as PhoneGap and Ionic that allow them to write the same code on all platforms by using the possibilities of HTML. The web-implemented part of the application can be either a page located on the server to which the application communicates, or a set of files containing web languages which are collected in the application and stored locally on the device.

Each approach has its pros and cons that need to be considered. The first approach allows the provision of application updates by avoiding the evaluation and acceptance process from any store where the application is available [9]. The disadvantage in this approach is the absence of offline operation of the application, since the content is on the server. The second approach provides the possibility of offline functionality, since the required files are included in the application itself that it is installed on the device [12]. However, this option removes the possibility of immediate updates of the application. The solution is the combination of both approaches. The files that can be hosted on the server are installed on it and when it is first run the application will save them locally on the device (app-cache). Finally, if there is a file change, the device will be notified of the change and it will attempt to retrieve it from the server [9] [10].

Internet technologies for the development of compatible applications on different computing platforms

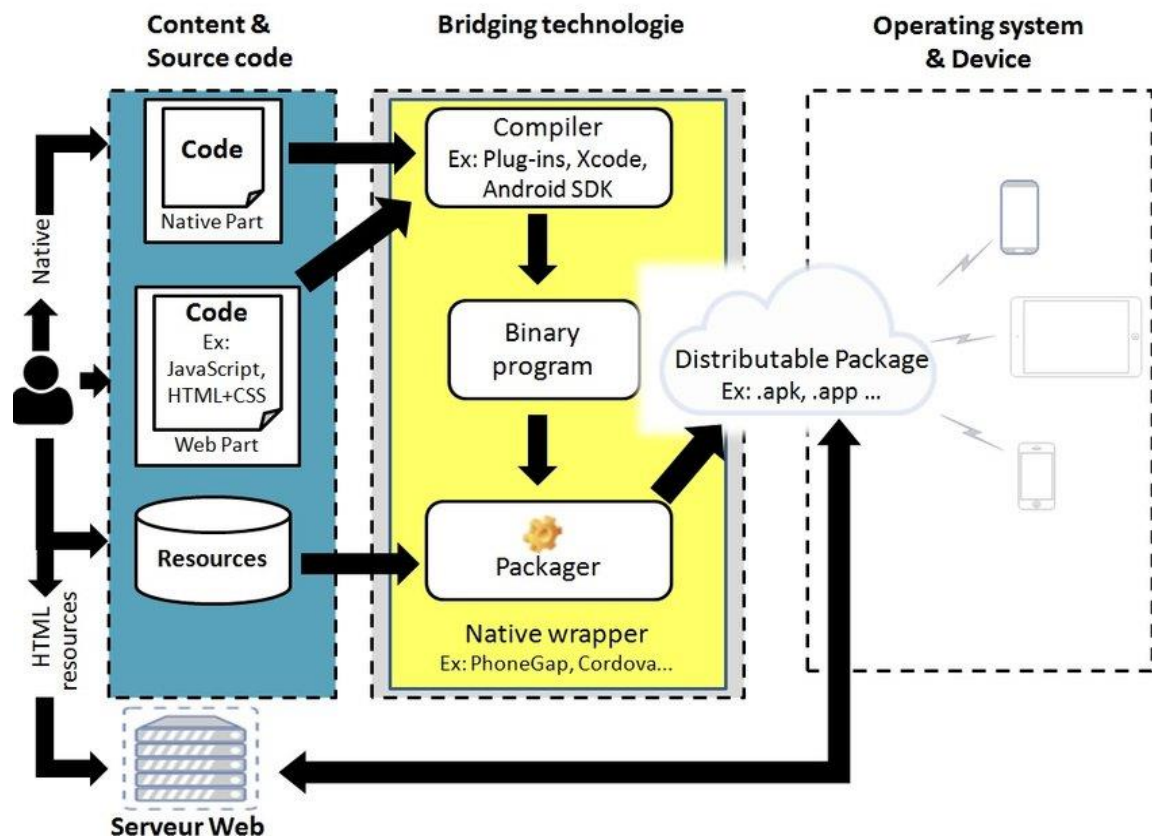


Figure 8: Hybrid application architecture, Source: [11]

2.4 Choosing Between the Three Types of Mobile Apps

The choice of which development is better depends on many factors. Some of them are:

- ❖ ***Time to Market:*** Web apps use less time to develop than the native apps while hybrid stands somewhere in the middle. Native apps development is slower because it has to be implemented on two different platforms, one for iOS and one for Android [9] [13].
- ❖ ***Target Audience and User Experience:*** User experience is one of the most important things in maintaining an app. Also, user experience is related directly to whom we are trying to target. The companies give significant attention on audience preferences and choices. By collecting them, they are trying to develop the best app with the right technology [13].
- ❖ ***Installation:*** Android native apps are free to download but they required frequent updates in order to ensure an excellent user experience, whereas iOS native apps are paid. Web apps are free to access on any device and browser and they required fewer updates than native apps. Then, hybrid apps are the best choice if someone is trying to target the largest number of

Internet technologies for the development of compatible applications on different computing platforms

users through multiple platforms with a minimum requirement of updates [10] [13].

- ❖ Cost of Development: Native apps cost more as different people with various skills are needed. In addition, the cost depends on the complexity of the app. Hybrid apps cost less to build but requires higher skills of the developer team as they write a single code base for use on multiple platforms. Web apps cost less than the other two [9].
- ❖ Offline Connectivity: Native apps can work without internet connection. Users are able to load the app and access previously loaded data. Web apps do not work without internet connection. Hybrid apps will work offline if they have some native code [10].
- ❖ Device Features: Native apps have access to the entire device APIs in contrast to web apps [13].
- ❖ Distribution Channel: Native and hybrid apps are hosted in an app store whereas web apps are directly available on the web [9] [13].
- ❖ Performance: Native apps provide high performance as they can easily access device functionality and elements to ensure a faster response rate. Hybrid apps are working on a platform to download data from the server and have limited access to device features. That is why the natives are better. On the other hand, web apps are based on internet connection and browser's performance [9] [10].
- ❖ Maintenance: Maintaining a native app can be complicated for developers as they have to deal with multiple versions of the app in different platforms. Changes have to be gathered in a new version and placed in the app store. Web and hybrid apps are as simple as maintaining a web page [13].
- ❖ Speed: Native apps are the fastest apps in the market because they access directly the hardware [14]. Many elements come preloaded. The user data is fetched from the web rather than the entire application, and since they work with the device's built-in features, they are speedy.

Internet technologies for the development of compatible applications on different computing platforms

	Native	Hybrid	Web
Cost	Native cost more to develop. The cost increases more if the app has to be developed for multiple platforms	Cost less than native apps to build but requires higher skills	Cost less than Native or Hybrid apps.
Performance	Native app can access device functionality, elements etc. for instant usage hence have faster response rate	Application mainly act as a medium to download data from the server. So, the performance is little downgraded in comparison to native apps	The performance is based on the internet connectivity and browser performance.
Distribution Channel	App are hosted in the App store of the operating system. This allows the app to leverage showcasing features and ranking	App are hosted in the App store of the operating system. This allows the app to leverage showcasing features and ranking	Are not stored in the App store but are directly available on web.
Device Feature utilization	Has access to all the device APIs and can leverage all device functions	Hybrid apps are supported by some device APIs but there are still some features that cannot be access by hybrid apps	Only few device functionalities can be accessed such as location.
User Interface	Apps are developed for the device so are more user friendly and close to the operating system	There are some restriction with the app due to cross platform operatibility but some apps can be configured to have a pretty close to native appeal	There are some restriction with the app due to cross platform operatibility but some apps can be configured to have a pretty close to native appeal
Code maintainence	Common code cannot be used for all devices. The maintenance of code is also on the higher side of the monetary scale.	Codebase can be ported to all major platforms. The maintenance is also less due to use of single code structure.	Code can be used in any browser that supports it. The maintainence cost is also low as the same code base is used across platforms.
Recommended Uses	For the application developed for single platform App that requires high optimization level and native UI feel	Apps that needs to be operated through wide variety of devices The apps that needs to be available on app store	Applications that have limited resources and funds Apps that doesn't requires to be featured in app stores.

Figure 9: Comparison of Native, Hybrid and Web application development,
Source: <https://betterprogramming.pub/native-hybrid-or-web-apps-what-is-the-best-approach-for-lasting-success-91afbb872d89>

The decision to build either a native, web or hybrid app should be based on every business objectives. Before someone starts building an application, the following factors should be considered:

1. How long the application should be ready.
2. How complex the features will be.
3. It should provide a high quality of user experience no matter the approach.

2.5 Cross – platform Development

Cross-platform or multi-platform is computer software that is implemented on multiple computing platforms. In practice, it allows developers with a specific

Internet technologies for the development of compatible applications on different computing platforms

programming language to develop software by writing the program only once and running on all systems with little or without any modification [15] [16].

One of the advantages we have by using cross-platforms applications is that the implementation with HTML and JavaScript languages is easier than the implementation with Java. This is due to the ease of the use of development tools and language familiarity. Thus, they reduce technical barriers that may hinder the approval of natural development, but they require great expertise in different technologies in order to use them [15] [17].

2.5.1 Cross – platform Advantages

There are several advantages of using cross - platform development:

- ⊕ Maximum Exposure to Consumers: With the cross-platform strategy can create an application that runs on other platforms such as iOS and Android. In this way the usefulness of an application is extended and at the same time more consumer audience is gained [15] [16].
- ⊕ Reduced Development Cost: Code reuse and flexible development can offer reduced cost to a business [15] [17].
- ⊕ Reusable Code: The code can be used many times resulting in a significant reduction of time [16] [17].
- ⊕ Easier Maintenance: Updates can be synced instantly across all platforms and devices. Moreover, if an error is detected in the common code base it will be corrected at once [15] [16].
- ⊕ Quicker Development Process: Developing a single code that can run on multiple platforms can reduce development efforts by 50 -80% [15] [17].
- ⊕ Cloud Integration: The single source code is enriched with plug-ins and extensions in order to improve the functionality of the application [15] [16].
- ⊕ Faster Time-to-Market and Customization: Time-to-market is reduced since developing one program takes a lot less time than creating two or three apps for each device's platform. If a customer needs to customize differently the app, it is easier for the developers to make small corrections in a single code [16].
- ⊕ Recognizable User Interface: Uniformity increases between different operating systems creating a smoother user experience [15] [17].

Internet technologies for the development of compatible applications on different computing platforms



Figure 10: Advantages of cross – platform development,
Source: <https://www.netsolutions.com/insights/cross-platform-app-frameworks-in-2019/>

2.5.2 Cross – platform Disadvantages

Despite the many positive elements, they also present some negative ones:

- ⊖ Integration Problems: It is not easy to integrate an app with local settings, preferences and notifications [15] [16].
- ⊖ Vendor Lock-In: Most development frameworks use a subset of JavaScript. If they are committed to one framework and need to move to a different framework, it may be difficult to make that transition [15] [17].
- ⊖ Lack of Updates and Features: Every time Google or Apple introduces a new feature for Android or iOS, it takes some time to update applications to support this new feature. In native apps, new SDKs are provided with the updates much faster than on cross-platform frameworks [16] [17].

2.6 Native vs. Cross-platform: When we choose each one

Each of these options has its pros and cons. The choice between cross-platform and native development should be made depending on the required functionality and the app's implementation area. It is impossible to give a universal answer on which one is better, since each case is unique, but there are some tips in what situations to use each development.

Native app development will be an excellent choice if:

- ◆ The application requires full access to all of the phone resources and services [18].
- ◆ Someone wants to build the most responsive application [19] [20].
- ◆ The product requires full access to hardware features [19] [20].

Internet technologies for the development of compatible applications on different computing platforms

- ◆ Someone wants an app that can be easily updated and enhanced with new features [18].
- ◆ The product must provide the highest possible quality of user experience [19] [20].

Cross-platform technology will be an excellent choice if:

- ◆ A less responsive app is suitable [18].
- ◆ There is a short window to test an idea in the app market under real life conditions [19] [20].
- ◆ There is a need to create an app for several platforms on a limited budget.
- ◆ The application does not involve complex functionality and logic [18] [20].

Comparison Factors	Native App Development	Cross-platform App Development
Tools/Technologies	Swift/Objective C (iOS) and Java/Kotlin(Android)	Javascript(React Native), C#(Xamarin) and Dart(Flutter)
UI/UX	Complete platform specific UI	Common UI for all kind of apps (Although Limited Customization)
Performance	Excellent performance	Performance issues are common or they can be solved with fewer workarounds
Testing	Testing is seamless with tools available in the framework itself or browser	Testing cross-platform frameworks vary with the frameworks available
Learning Curve	Easy Learning curve	Moderate Learning curve
Developer Experience	Good	Fine yet it's improving
Time to Market	Time to Market in Native apps is long as you have to build different apps for both platforms	Time to Market in Cross platform apps is good as development speed increases due to single codebase
Team Size	Big Team size(Different Teams for different platforms)	Small to mid-sized Team size (1 Team for all platforms)
3rd Party SDKs availability	Excellent	Average

Figure 11: Comparison of native and cross-platform development,
Source: <https://www.simform.com/native-vs-cross-platform-development/>

As mentioned above, both technologies provide some positive and negative aspects that need to be considered from the developers. Ultimately, the choice of going on with either of them depends on project requirements and the skills that the developer team has.

In summary, native apps are still the best choice when it comes to user experience performance. They are more expensive but the defect rates are lower. On the other hand, cross-platforms apps are easy and quick to build, but it will take more effort to deliver an equivalent user experience on the respective platform.

Chapter 3: Cross-platform Frameworks Analysis

Currently, there are handfuls of cross-platform options to choose from when a developer creates an application. However, each option targets to ease different aspects of development. Picking the right framework depends on various factors, but mainly on the type of application and the aim of the final product.

This chapter gives an overview of seven cross-platform frameworks starting with Xamarin and continuing with Apache Cordova, Ionic, React Native, Flutter, Framework7 and NativeScript. Their mode of operation, the advantages and disadvantages as well as an example of code of each framework are presented in detail.

3.1 Xamarin

Xamarin is an open source framework of Microsoft for creating cross-platforms mobile applications, which are written entirely in C#. Its commercial use is free but the business and enterprise use cost 999\$/year and 1899\$/year respectively [21].



Figure 12: Xamarin logo,
Source: <https://commons.wikimedia.org/wiki/File:Xamarin-logo.svg>

Xamarin offers two ways to create mobile applications. Xamarin Native is the first approach that uses Android and iOS SDKs, which have been mapped to C# classes. What can be done in a native application, it can also be done in Xamarin since the creators provide almost one hundred percent coverage of the Android and iOS API [22]. This is a useful way to build an app because the core library can be tested with a single platform and the UI can be worked out entirely on one platform before adding a second or a third platform. In this approach, the UI is created separately for each of the below platforms:

1. Xamarin.Mac
2. Xamarin.iOS
3. Xamarin.Android
4. Xamarin.Windows

Internet technologies for the development of compatible applications on different computing platforms

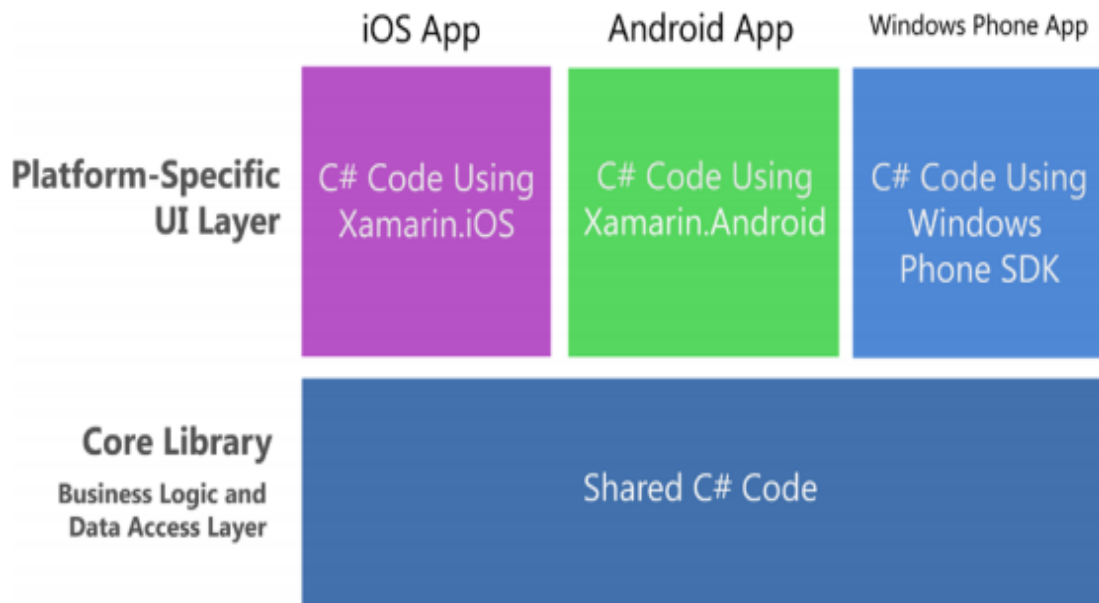


Figure 13: Xamarin native architecture, Source: [23]

The second approach to create applications is by using Xamarin Forms. This is a library that provides a platform-independent programming interface. Furthermore, it is possible to achieve a maximum amount of code shared between platforms. The application interface is created once for all platforms in the XAML language using shared controls library [21]. When the application is running, they are mapped to native components of the platform. This allows achieving native performance and quality of the user interface.

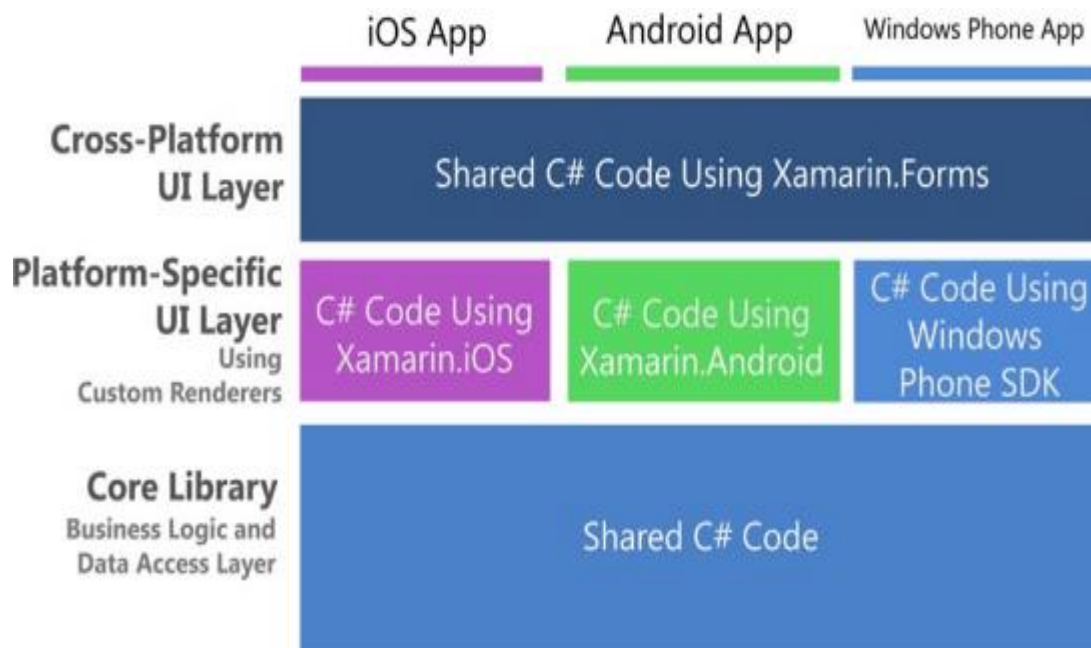


Figure 14: Xamarin.Forms architecture , Source: [23]

Internet technologies for the development of compatible applications on different computing platforms

The system compatibility for the Xamarin framework and the recommended development environment and SDK versions are shown in the following table:

Xamarin	Development Environment	Operating System	Xamarin.iOS	Xamarin.Android	Xamarin.Mac	Xamarin.Forms
Windows	Visual Studio 2017 or 2019	Windows 10 version 10.0.18362.0 or newer	iOS 10 SDK (installed on a Mac)	Android 6.0 / API level 23	Open project and compile only	Android, Windows/UWP ⁷ (iOS with Mac computer)
MacOs	Xcode 10 or Visual Studio for Mac	MacOS Mojave (10.14) or newer	iOS 12 SDK	Android 6.0 / API level 23	MacOS Mojave (10.14) SDK	iOS, Android

Table 3: Xamarin system requirements, Source: [22]

To develop an iOS application on Windows operating system, there must be a Mac computer accessible on the network, for remote compilation and debugging [22]. This also works if the Visual Studio running inside a Windows VM on a Mac computer.

3.1.1 Advantages of Xamarin

The advantages of using Xamarin are:

- ⊕ Single Technological Stack: It uses a single language C# complemented with .Net framework to create apps for any mobile platform. Moreover, there is no need to switch between environments as Xamarin supplies add-ins to Microsoft Visual studio [24].
- ⊕ Shareable Code: A single codebase can run on various platforms providing native performance. Xamarin Forms create a 60-95% reusable code [25] [26].
- ⊕ Native UE: Xamarin has full access to native APIs and toolkits used on Android, iOS and Windows platforms. As a result, it can provide performance quite close to native ones [26].
- ⊕ Time and Cost Saving: Since everything is written in C# and within the .NET framework, there is no need for separate teams working on the app. The development, testing and deployment are significantly streamlined when performed by a single team eliminating additional expenses and ensuring quick time to market [26].

⁷ UWP stands for Universal Windows Platform, which is a computing platform created by Microsoft and first introduced in Windows 10. The purpose of this platform is to help develop universal apps that run on Windows 10, Windows 10 Mobile, Xbox One and HoloLens without the need to be rewritten for each.

Internet technologies for the development of compatible applications on different computing platforms

- ⊕ *Simplified Maintenance*: It is easier to maintenance and updates a Xamarin app using Xamarin Forms [24]. If the source code is updated, the changes will be automatically applied to all platforms.
- ⊕ *Open Source*: That means that a programmer can find a solution to a problem by seeing similar code or even contribute by uploading a script [24] [25].
- ⊕ *Testing*: The platform offers comprehensive solutions to test and monitor application performance and UI-Xamarin Test Cloud and Xamarin Test Recorder. These tools allow running automated tests on multiple real devices in the cloud and finding performance issues before the release [25].
- ⊕ *Technical Support and Community*: Xamarin guarantees stability, continuous technical backing and fast issue resolution as Microsoft supports this platform. Also, Xamarin has an extensive community [24] [26].

3.1.2 Disadvantages of Xamarin

Despite the positive aspects, there are some disadvantages:

- ⊖ *Expensive for Enterprises*: Xamarin is free for individuals and small companies, however, enterprises need to purchase a license for Microsoft's Visual Studio [24].
- ⊖ *Not Suitable for Apps with Heavy Graphics*: If the application has rich UX/UI, it should be implemented natively [25] [26].
- ⊖ *Larger App Size*: Xamarin adds 3–5 megabytes for the release and around 20 megabytes for debug builds [24].
- ⊖ *Limited access to open source libraries*: While Xamarin does support most of .Net libraries; it does not support all of the available 3rd-party libraries for Android and iOS without specific wrappers [25].
- ⊖ *Limited Talent Pool and Community*: The range of those who know how to program in Xamarin is very limited as well as the community in relation to native Android and iOS community [26]. GitHub Stars⁸ reaches 5.400 stargazers.
- ⊖ *Knowledge of Native Languages*: If an app requires heavy UI, it is necessary to manually rewrite some parts of platforms-specific code using the required language (Swift or Objective-C for iOS and Kotlin or Java for android) [24].
- ⊖ *Lack of Online Coding Playground*: It is a browser-based environment for developing apps. It is a great place to get started learning a framework, as a developer can develop apps without needing to install the various SDKs and tools needed for native iOS and Android development.

⁸ The GitHub Stars program offers impactful developers an opportunity to showcase their work, reach more people, and shape the future of GitHub.

3.1.3 Xamarin App

In order to build a Xamarin app, it is necessary to set up the proper environment with the right workloads. The table below shows all the prerequisites depending on the operating system development.

Prerequisites for Xamarin			
Development OS	Target OS	Editor	Workloads
Windows	Desktop	Visual Studio 2019 Community or Visual Studio Professional or Visual Studio Enterprise	<ul style="list-style-type: none"> • Mobile development with .NET
	Android	Visual Studio 2019 Community or Visual Studio Professional or Visual Studio Enterprise	<ul style="list-style-type: none"> • Mobile development with .NET • Android SDK • Android Emulator
	iOS	Visual Studio 2019 Community or Visual Studio Professional or Visual Studio Enterprise	<ul style="list-style-type: none"> • Xcode • Visual Studio for Mac • iOS Emulator
macOS	Desktop	Visual Studio for Mac	<ul style="list-style-type: none"> • Xcode • Xamarin.Forms • .NET Core applications • ASP.NET Core Web Applications • Azure Functions
	Android	Visual Studio for Mac	<ul style="list-style-type: none"> • Android • Android SDK Manager • Android Emulator • .NET Core applications • ASP.NET Core Web Applications • Azure Functions
	iOS	Visual Studio for Mac	<ul style="list-style-type: none"> • Xcode • iOS Emulator • .NET Core applications • ASP.NET Core Web Applications • Azure Functions

Table 4: Xamarin prerequisites, Source: [22]

Internet technologies for the development of compatible applications on different computing platforms

To better understand the operation of the Xamarin platform in a code level, a simple application was implemented. Xamarin.Forms was used with the same code working equally well on Android and iOS. The app consists of 2 pages with a navigation bar. The first page has the Xamarin logo and a button that redirects to the second page. The second page consists of 2 paragraphs with a title. The features of the computer and the tools used are:

- ❖ *Operating System:* Windows 10 (64bit)
- ❖ *Ram:* 8GB
- ❖ *CPU:* Intel core i3-3110m 2.40 GHz
- ❖ *Disk Space:* 8,58 GB free disk space without emulator
- ❖ *Development Environment:* Visual Studio Community 2019 version 16.9 (with **Mobile development with .NET** workload)
- ❖ *Emulator:* 1) Android - Xiaomi Redmi Note 8 Pro
2) iOS – iPhone 8 iOS 11.4

The following code contains all the necessary files of the application. The first project contains several folders, code files, and user interface files. This project is a .NET Standard project that enables to have a single project that can be shared across different operating systems. This project is where most of the code was written. Also, it is observed that each folder consists of two files, one Xaml and one Xaml.cs. In a Xamarin.Forms application, XAML is mostly used to define the visual contents of a page and works together with a C# code-behind file.

iOS and Android projects are the "head" or "parent" project that is used to house platform specific code, settings, assets, and more. This is where to configure different settings such as display name, app icon, version numbers, and any code that is needed for iOS or Android specific things that are not available in cross-platform mode.

Both App.xaml and App.xaml.cs contribute to a class named App that derives from Application. Most other classes with XAML files contribute to a class that derives from ContentPage; those files use XAML to define the visual contents of an entire page.

Internet technologies for the development of compatible applications on different computing platforms

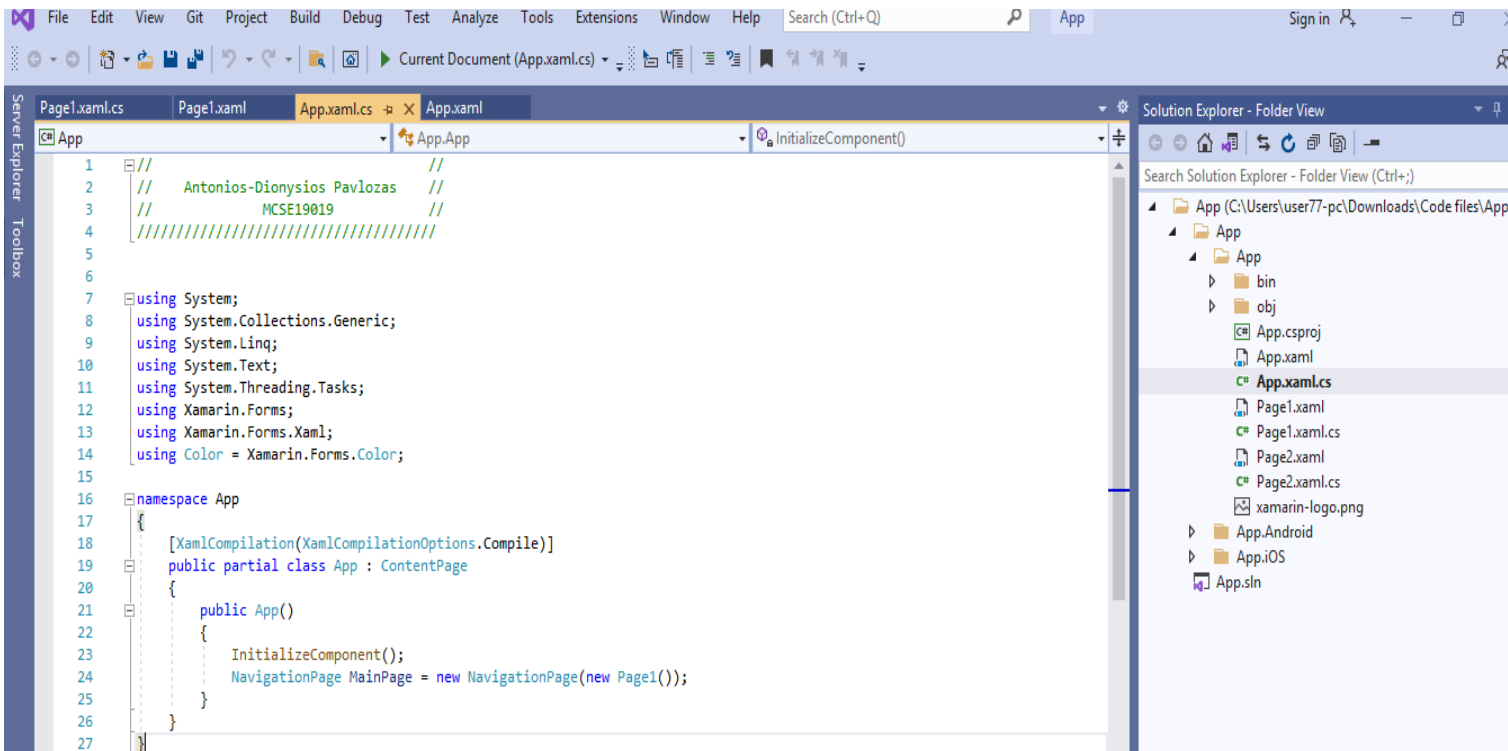


Figure 15: App.xaml.cs

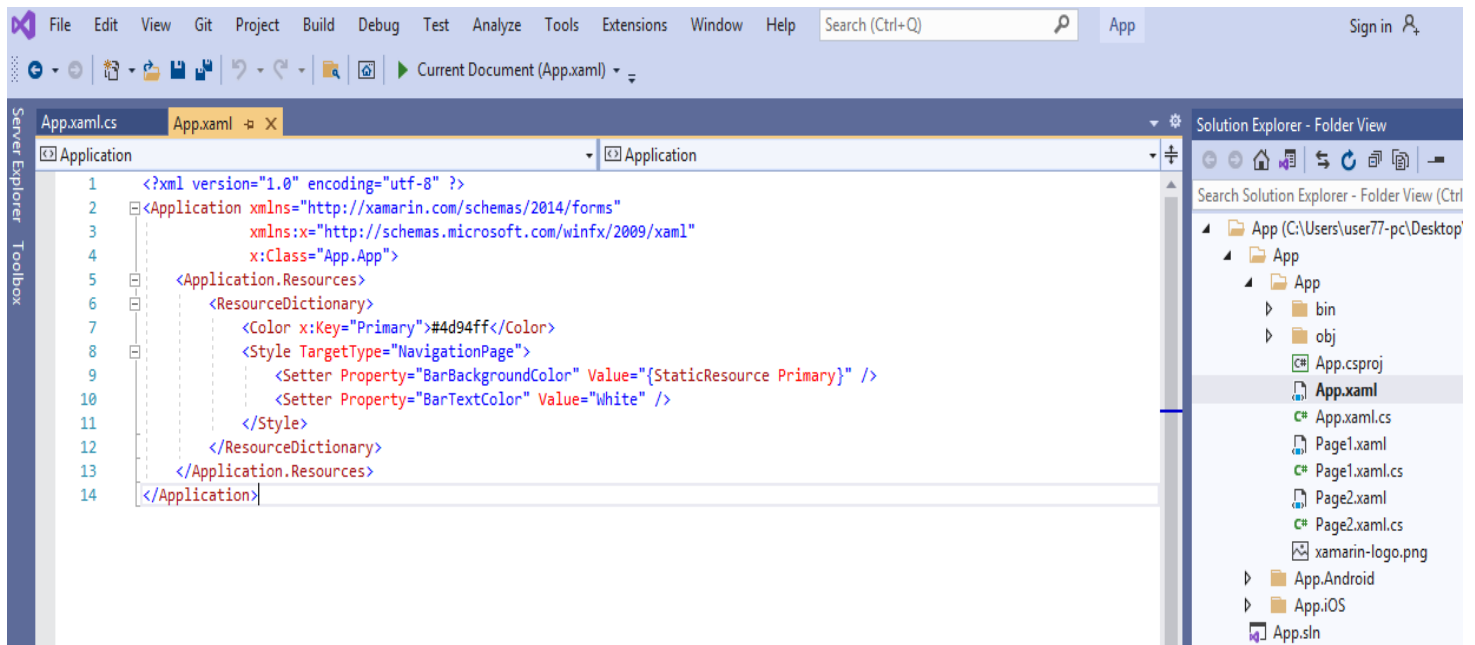


Figure 16: App.xaml

The first image is the main page of the app which is linked with the Page1 and the second image shows some styling of the navigation bar. In order to run the app, an emulator is needed. The Visual Studio provides the ability to download an emulator directly through the platform but it also allows us to connect user's smartphone and

Internet technologies for the development of compatible applications on different computing platforms

run the application in real time. The same images are display when running the application on an iOS smartphone.

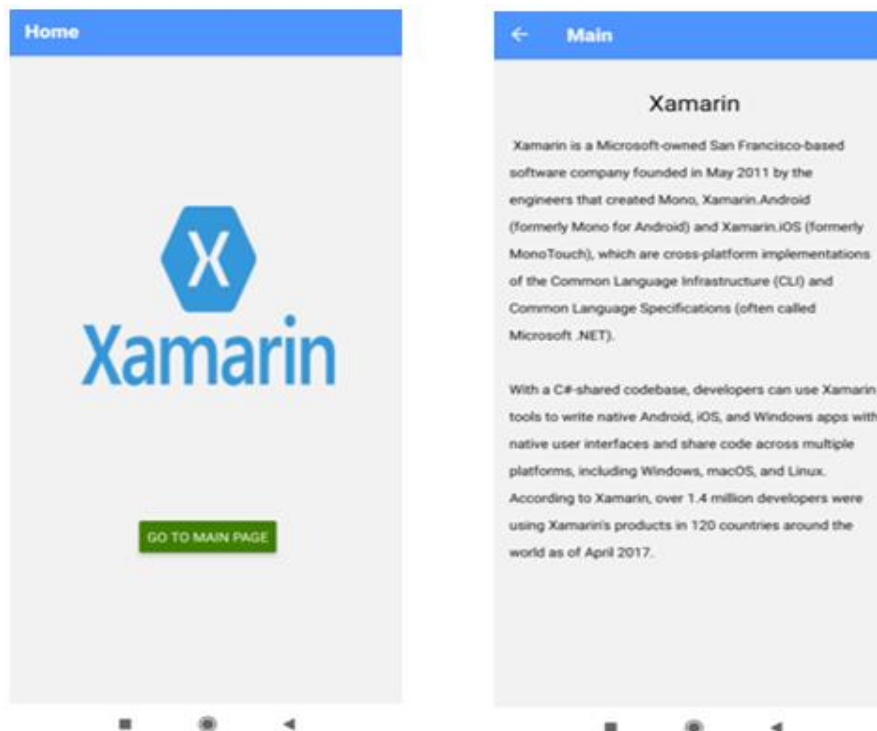


Figure 17: Xamarin app

Some of the most famous apps, which are redesigned with the Xamarin Forms with reusing of 75% of the old code, are:

Xamarin Apps			
1	Olo	7	Captio
2	MRW	8	Fareboom
3	APX	9	Picturex
4	Novarum DX	10	Vanderlande
5	World Bank	11	Insightly
6	Storyo	12	Alaska Airlines

Table 5: Famous Xamarin apps, Source: [27]

3.2 Apache Cordova

Apache Cordova (formerly PhoneGap) is a free open-source mobile development framework. It allows the user to use standard web technologies such as HTML5, CSS3, and JavaScript for cross-platform development avoiding each mobile platform's native development language [28].

Internet technologies for the development of compatible applications on different computing platforms



Figure 18: Apache Cordova logo, Source: <https://cordova.apache.org/artwork/>

Apache Cordova provides a set of APIs that can be used to access and stimulate the native mobile OS features like Camera, Storage, Accelerometer, Geolocation, etc. To access the device's native feature, JavaScript application objects are used via calling the Apache Cordova APIs [28]. Basically, Apache Cordova creates a single screen in the native application and this screen consists of a single WebView that holds available space on the device screen. Cordova uses the native application WebView for loading the application and its related JavaScript and CSS files [29] [30].

When the application launches, firstly, Apache Cordova loads the application's default startup page (usually index.html) in the application's WebView and passes its control to the WebView [28]. The WebView allows the user to interact with the application by entering data in input fields, clicking on the action buttons, and viewing results in the application's WebView [30]. To access the mobile's native functionalities such as audio or camera, Cordova provides a package of JavaScript APIs that can be used by the developers from their JavaScript code. The calls to the Cordova JavaScript APIs will translate into native device API calls by using a special bridge layer [29] [30]. The native APIs can be accessed from the Apache Cordova plugins. Hence, with Apache cordova 50-80% of reusable code is possible.

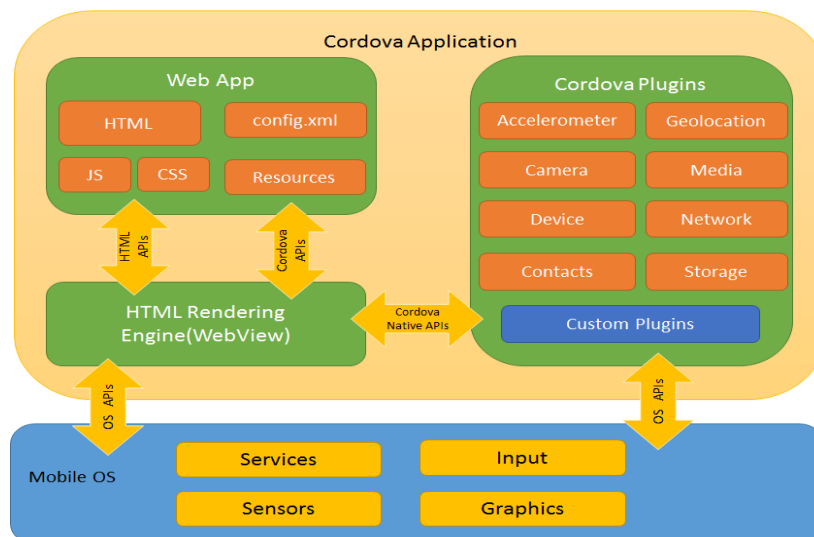


Figure 19: Apache Cordova architecture, Source [28]

The system compatibility for the Apache Cordova framework and the recommended development environment are shown in the following table:

Apache Cordova	Development Environment	Operating System
Windows	Visual Studio 2017 or 2015	Windows 10
	Visual Studio 2015 or 2013	Windows 8.1
MacOs	Xcode 11.0	MacOS Mojave (10.14.4) or later version

Table 6: Apache Cordova system requirements, Source: [28]

3.2.1 Advantages of Apache Cordova

The advantages of using Apache Cordova are:

- ⊕ Cordova offers a single platform that can be used to create cross-platform mobile applications. Hence, an application is created that will be used on different mobile platforms: iOS, Android, Windows Phone and UWP [31] [32].
- ⊕ Cordova is not just an HTML application that runs in a browser, it allows the developer to write native plugins that work with any of the supported platforms, and a JavaScript container that integrate the HTML application with the native code [32].
- ⊕ Easier to maintain as it bypasses version making it simple to update. With native apps, developers need to deliver a new version for each update and users need to update the app for each new version release [31] [32].
- ⊕ Saves time as hybrid apps take less time to develop compared to developing an app for each native platform separately [31].
- ⊕ Free and open source [31].
- ⊕ A single codebase can run on various platforms providing native performance with 50-80% of reusable code [31] [32].

3.2.2 Disadvantages of Apache Cordova

The disadvantages of using Apache Cordova are:

- ⊖ Cordova app code runs inside a WebView component. So, the performance is a little slower compared to the native apps [32] [33].
- ⊖ There may be compatibility issues for some plugins across different platforms and devices. That means that plugins may not function properly so it may need to be forked and modified [32].
- ⊖ Crashes can be hard to debug since the crash logs will not point the developer to the culprit JavaScript code [33].
- ⊖ Small community [31] [32]. GitHub Stars reaches just 1000 stargazers.
- ⊖ Lack of online coding playground [33].

3.2.3 Apache Cordova App

To create a cross-platform app with Apache Cordova, the tool Cordova Command-Line Interface (CLI) must be used, which serves as a higher level abstraction for configuring and building an application for different platforms [34]. Before installing and running anything related to Cordova, some necessary SDKs need to be installed.

Apache Cordova Prerequisites		
Development OS	Target OS	Dependencies
<i>Windows</i>	Android	<ul style="list-style-type: none"> • Git • JAVA SDK • Node.js • NPM • Android Studio • Android SDK • Cordova CLI • Android Emulator
	iOS	<ul style="list-style-type: none"> • Git • JAVA SDK • Node.js • NPM • Xcode • Android SDK • Cordova CLI • Android Emulator
	Windows Phone, UWP	<ul style="list-style-type: none"> • UWP Development Kit • Windows 10 SDK • Windows Phone Emulator
<i>macOS</i>	Android	<ul style="list-style-type: none"> • Android SDK Manager • Android Emulator • .NET Core applications • ASP.NET Core Web Applications • Azure Functions
	iOS	<ul style="list-style-type: none"> • Xcode • iOS • .NET Core applications • ASP.NET Core Web Applications • Azure Functions

Table 7: Apache Cordova prerequisites, Source: [34]

Internet technologies for the development of compatible applications on different computing platforms

Once the above are installed, the next step is to install the Cordova module globally by opening the command prompt⁹ (cmd) and run the following script:
npm install-g cordova

After installing the Cordova module via the command prompt, the following script need to be typed:

create CordovaProject com.cord.app CordovaApp

CordovaProject is the directory name where the app is created, *com.cord.app* is the default reverse domain value and *CordovaApp* is the title of the app.

To add platforms the following scripts needs to be written:

cordova platform add android

cordova platform add windows

cordova platform add iOS

cordova platform add blackberry

cordova platform add amazon-fireos

cordova platform add firefoxos

cordova platform add wp8

After adding the necessary platforms, the following folders are created in the directory which is defined earlier.

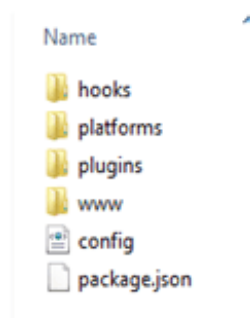


Figure 20: Apache Cordova directory

Hooks are pieces of code that can process, copy, or remove files before or after a specific Cordova command is executed. *Platforms* folder is where files needed by all the target platforms are stored. *Plugins* folder contains all the plugins that are installed in the app. The folder *www* contains the source code for the user interface of the app. *Config.xml* is the configuration file that is used for controlling the basic aspects and behavior of the app. And *package.json* is a file that contains details about the app's platform and plugin versions.

⁹ Command prompt is used to create a folder where the code files will be located.

Internet technologies for the development of compatible applications on different computing platforms

By default, *index.html* in *www* folder contains the following code:

```
01 <meta http-equiv="Content-Security-Policy" content="default-src 'self' data: gap: https://ssl.gstatic.co
02 <meta name="format-detection" content="telephone=no">
03 <meta name="msapplication-tap-highlight" content="no">
04 <meta name="viewport" content="user-scalable=no, initial-scale=1, maximum-scale=1, minimum-scale=1, widt
05
06 <link rel="stylesheet" type="text/css" href="css/index.css">
07 <title>Hello World</title>
08
09
10 <div class="app">
11   <h1>Apache Cordova</h1>
12   <div id="deviceready" class="blink">
13     <p class="event listening">Connecting to Device</p>
14     <p class="event received">Device is Ready</p>
15   </div>
16 </div>
17
18 <script type="text/javascript" src="cordova.js"></script>
19 <script type="text/javascript" src="js/index.js"></script>
20
```

Figure 21: Apache Cordova index.html code

As shown above, it is a simple HTML file. The only thing that is different is the meta tags. The first meta tag specifies the content security policy. This makes the Cordova application secure from cross-site scripting (XSS) attacks. Any script that an attacker manages to inject into the page would simply refuse to be loaded with this meta tag in place.

The *format-detection* meta tag automatically converts phone numbers into links. The user can then click the link to make a call to that phone number.

The *msapplication-tap-highlight* meta tag disables the grey tap highlight on Windows Phone 8 and later.

The next tag is the *viewport* meta tag. The tag first specifies that the user should not be able to zoom the page in or out. Next, the *initial-scale* is set to 1. This means that the content is loaded at 100%. The *maximum-scale* and *minimum-scale* are both set to 1. This sets the minimum and maximum values that the user can set for the zoom level. The *width* attribute specifies that the maximum device width is used for the content.

After the meta tags the *stylesheet* that contains basic styling for the app is presented. The *p* element with a class of *listening* is shown when the devices APIs are not fully loaded yet. It is hidden when the devices APIs are ready [30].

And lastly, there are two *script* tags. The first one includes *cordova.js*. This file provides a unified API for accessing native device features. The second *script* tag includes *index.js*. This file is specific to the default application that is created when a new Cordova project is started. All it does is to execute specific code when a specific event happens.

If the file *index.js* is opened, it will be observed that the *receivedEvent* function is responsible for hiding and showing the two *p* elements in the above code. It is being triggered by the *onDeviceReady* event. So the event identifier that is passed to the *receivedEvent* function is *deviceready*.

Internet technologies for the development of compatible applications on different computing platforms

```
01  onDeviceReady: function() {
02      app.receivedEvent('deviceready');
03  },
04
05  receivedEvent: function(id) {
06
07      var parentElement = document.getElementById(id);
08      var listeningElement = parentElement.querySelector('.listening');
09      var receivedElement = parentElement.querySelector('.received');
10
11      listeningElement.setAttribute('style', 'display:none;');
12      receivedElement.setAttribute('style', 'display:block;');
13
14      console.log('Received Event: ' + id);
15  }
16 }
```

Figure 22: Apache Cordova index.js script

Running the default application will show the following image:

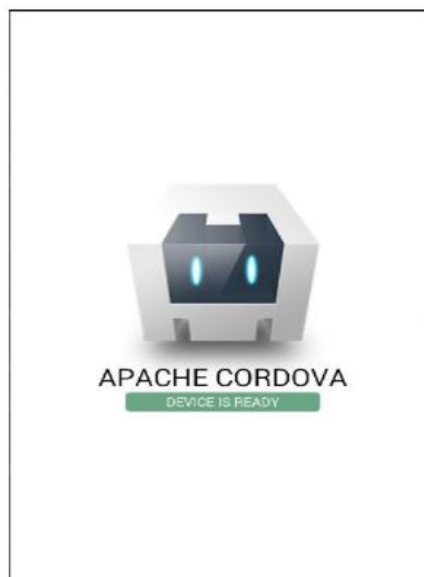


Figure 23: Apache Cordova results

Some of the most famous apps, which are designed with Apache Cordova, are:

Apache Cordova Apps			
1	Pacifica	7	Keep
2	Sworkit	8	Clever Baby
3	FanReact	9	SparkChess
4	JustWatch	10	Graded
5	Gudog	11	Buildr
6	Series Seven	12	One Verse BIBLE

Table 8: Famous Apache Cordova apps, Source: [33]

3.3 Ionic

Ionic is an open source HTML5 framework for developing hybrid applications for mobile devices. Basically, Ionic provides the front-end UI framework that handles all the look and feel, and enables UI interactions of a hybrid app. Its commercial use is free but the pro and enterprise use cost 539\$/year and 2999\$/year respectively [35].



Figure 24: Ionic logo,

Source: https://commons.wikimedia.org/wiki/File:Ionic_Logo.svg

In a typical HTML5 development framework, there are inherent limitations. For instance, a push notification for a website cannot be implemented on any iPhone or Android device. With Ionic, capabilities of a typical mobile app can be embedded to overcome the limitations in the existing HTML5 development framework. Therefore, Ionic framework can deliver an HTML5 with native-styled mobile UI elements and layouts as well as built-in mobile application capabilities with access to the native device functionality. Besides HTML5, CSS3 and JavaScript, it also uses AngularJS for a lot of the core functionality of the framework. However, in order for its code to run like a native app, Ionic needs a native wrapper such as Cordova [36].

The main function of Cordova is to allow software programmers to build applications for mobile devices using CSS3, HTML5, and JavaScript instead of relying on platform-specific APIs like those in Android, iOS, or Windows Phone. In other words, Cordova acts as a container for running a web application written in HTML, CSS, and JavaScript [35] [36]. Typically, Web applications cannot use the native device functionality like Camera, GPS, Accelerometer, Contacts, etc. With Cordova, developers can access the native device functionality and can package the web application in the device installer format.

Internet technologies for the development of compatible applications on different computing platforms

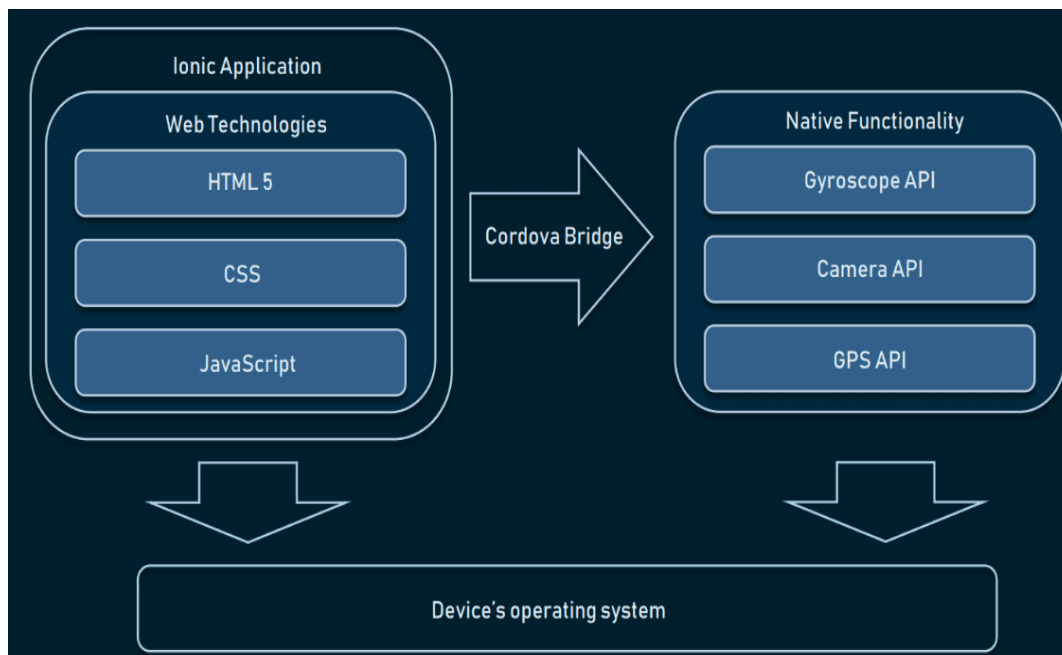


Figure 25: Ionic architecture, Source: [37]

The system compatibility for the Ionic framework is shown in the following table:

Ionic	Operating System
Windows	Windows XP or newer version
MacOs	MacOS X 10.5.8 or newer version with Intel chip
Linux	Operating system that includes GNU C Library 2.7 or newer

Table 9: Ionic system requirements, Source: [38]

3.3.1 Advantages of Ionic

The advantages of using Ionic are:

- ⊕ Ionic Framework offers a wide range of plugins and UI components to make an app look professional [37].
- ⊕ Ionic provides the developer with full, detailed documentation and support.
- ⊕ The apps are built in a modular way, so it is very maintainable and easy to update [37] [39].
- ⊕ Open source code [39].
- ⊕ Growing community [37]. GitHub Stars reaches 45.300 stargazers.
- ⊕ Single code for different operating systems providing native performance with 98% of reusable code [37] [39].
- ⊕ Easy learning curve as knowledge of HTML, CSS and JavaScript are sufficient [37].

Internet technologies for the development of compatible applications on different computing platforms

- ⊕ Quick development and time to market compared to native iOS/Android apps [39].
- ⊕ Testing is more convenient as it is used in a testing device and ensures everything runs smoothly [39].
- ⊕ Starting an app is very easy since Ionic provides useful pre-generated app setup with simple layouts [39].

3.3.2 Disadvantages of Ionic

The disadvantages are:

- ⊖ Hybrid apps tend to be slower than the native ones [37].
- ⊖ Expensive for Enterprises. Ionic is free for commercial uses, however, enterprises need to purchase an annual contract [37] [39].
- ⊖ Testing can be problematic since the browser does not always give the right information about the phone environment. There are so many different devices as well as platforms and a developer usually need to cover most of them [39].
- ⊖ Ionic development relies on Cordova which is not so developer-friendly [39].
- ⊖ In-app performance may not be as swift as if the application were developed natively for each device [37].
- ⊖ Despite the availability of various native plugins, one would not find a plugin for all native features. In such cases, the developers have to create plugins which is time-consuming [37] [39].
- ⊖ Lack of a hot reload feature. This staple feature of mobile application development allows programmers to test out their ideas without fully rebooting the project every time. Because of that Ionic development times are often longer when compared to other SDKs [37] [39].
- ⊖ Lack of online coding playground [37] [39].

3.3.3 Ionic App

In order to build an Ionic app, it is necessary to set up the proper environment. The table below shows all the prerequisites.

Prerequisites for Ionic			
Development OS	Target OS	Editor	Dependencies
	Android	Visual Studio Code	<ul style="list-style-type: none">• Node.js• NPM• Git• Android SDK• Android Emulator• Ionic CLI

Internet technologies for the development of compatible applications on different computing platforms

Windows			<ul style="list-style-type: none"> • Cordova SDK
	iOS	Visual Studio code	<ul style="list-style-type: none"> • Node.js • NPM • Xcode • Ionic CLI • Cordova SDK
MacOS	Android	Visual Studio code	<ul style="list-style-type: none"> • Node.js • NPM • Android SDK • Android Emulator • Ionic CLI • Cordova SDK
	iOS	Visual Studio code	<ul style="list-style-type: none"> • Node.js • NPM • Xcode • Ionic CLI • Cordova SDK
Linux	Android	Visual Studio Code	<ul style="list-style-type: none"> • Node.js • NPM • Android SDK • Android Emulator • Ionic CLI • Cordova SDK

Table 10: Ionic prerequisites , Source: [38]

The Ionic CLI is a preferred method for installing Ionic. It is the main tool for running the app and connects it to other services, such as Ionic AppFlow. To install the Ionic CLI globally with the NPM, the following command in the CMD needs to be written:

```
npm install -g ionic
```

When the Ionic is successfully installed on the system; the following screen should be displayed.

```
npm WARN deprecated superagent@4.1.0: Please note that v5.0.1+ of superagent removes User-Agent header by default, therefore you may need to add it yourself (e.g. GitHub blocks requests without a User-Agent header). This notice will go away with v5.0.2+ once it is released.
C:\Users\javatpoint\AppData\Roaming\npm\ionic -> C:\Users\javatpoint\AppData\Roaming\npm\node_modules\ionic\bin\ionic
+ ionic@5.2.7
added 226 packages from 169 contributors in 61.458s
```

Figure 26: Ionic CLI message installation

Internet technologies for the development of compatible applications on different computing platforms

After the successful installation, the Ionic app can be created. To create an Ionic app, it is necessary to navigate to the location where the application will be stored. In order to do this, the following command in cmd needs to be typed:

ionic start

Whenever the above command is running, it is necessary to write the Project's name and to choose the template which will install the appropriate dependencies. The following screen shows these options.

```
Every great app needs a name!
Please enter the full name of your app. You can change this at any time. To bypass this prompt next
time, supply name, the first argument to ionic start.
> Project name: myIonicApp
Let's pick the perfect starter template!
Starter templates are ready-to-go Ionic apps that come packed with everything you need to build your
app. To bypass this prompt next time, supply template, the second argument to ionic start.
> Starter template: (Use arrow keys)
> tabs | A starting project with a simple tabbed interface
  sidemenu | A starting project with a side menu with navigation in the content area
  blank | A blank starter project
  my-first-app | An example application that builds a camera with gallery
  conference | A kitchen-sink application that shows off all Ionic has to offer
```

Figure 27: Ionic project's name

After that, the file created on the code editor¹⁰ can be opened and the creation of the Ionic application can start. So, the example below depicts an app with two slider pages by using the *slides* components, which contain pages that can be changed by swiping or dragging the content screen. These slider pages are placed inside the sub child `<ion-slide>` of `<ion-slides>` components [40].

¹⁰ Visual Studio Code (VS) is recommended as a code editor. Its two big advantages are that it is free and works on macOS, Windows and Linux.

Internet technologies for the development of compatible applications on different computing platforms

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
6     <title></title>
7     <link href="lib/ionic/css/Ionic styling.css" rel="stylesheet">
8     <script src="lib/ionic/js/ionic.bundle.js"></script>
9     <script src="cordova.js"></script>
10    <script src="lib/ionic/js/angular/angular-resource.js"></script>
11    <script src="js/app.js"></script>
12  </head>
13  <body>
14    <ion-header translucent>
15      <ion-toolbar color="danger">
16        <ion-title>Slides</ion-title>
17      </ion-toolbar>
18    </ion-header>
19    <ion-content fullscreen class="ion-padding" scroll-y="false" color="success">
20      <ion-slides>
21        <ion-slide>
22          
23          <h2>Welcome to the <b>JavaTpoint</b></h2>
24          <p>JavaTpoint offers Corporate Training, Summer Training, Online Training and
25            Winter Training on Java, Android, Python, Oracle, PHP, and many more technologies.</p>
26        </ion-slide>
27        <ion-slide>
28          
29          <h2>What is Ionic?</h2>
30          <p><b>Ionic Framework</b> is an open source SDK that enables developers to build high quality mobile apps with web
31            technologies like HTML, CSS, and JavaScript.</p>
32        </ion-slide>
33      </ion-slides>
34    </ion-content>
35  </body>
36 </html>
```

Figure 28: Ionic app slide

The styling for this app is in external file. In this file, the margin of the ion-slide and the properties of the images have been set.

Internet technologies for the development of compatible applications on different computing platforms

```
ionic styling.scss
1  :root {
2    --ion-safe-area-top: 20px;
3    --ion-safe-area-bottom: 22px;
4  }
5  .swiper-slide {
6    display: block;
7  }
8  ion-slide > h2 {
9    margin-top: 2.8rem;
10 }
11 ion-slide > img {
12   max-height: 50%;
13   max-width: 60%;
14   margin: 36px 0;
15 }
```

Figure 29: Ionic styling

The screenshot of the ionic app is shown below.

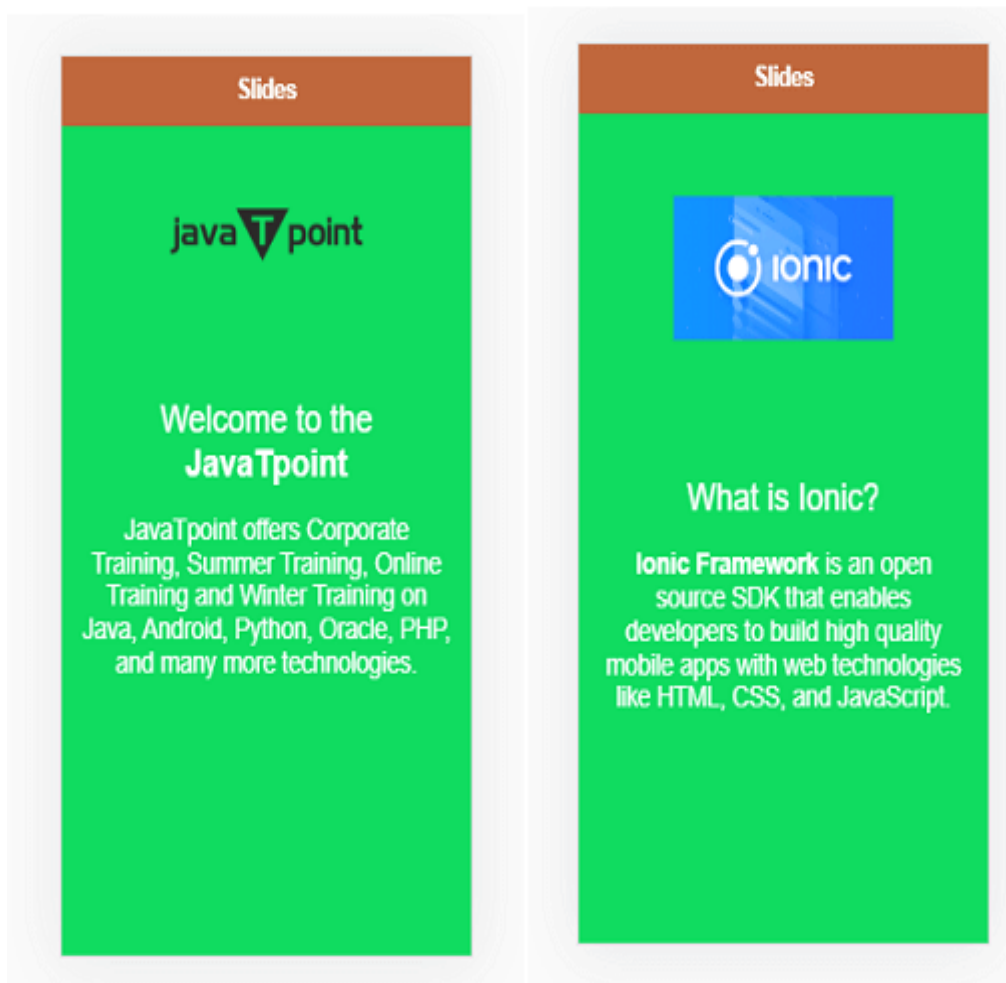


Figure 30: Ionic app

Internet technologies for the development of compatible applications on different computing platforms

Some of the most famous apps, which are designed with Ionic, are:

Ionic Apps			
1	Pacifica	9	Honeyfi
2	Sworakit	10	McLaren Automotive
3	MarketWatch	11	TD Trading
4	National Museum of African American History and Culture	12	JustWatch
5	Diesel	13	ChefSteps
6	StockPlan Connect	14	McDonald's Turkiye
7	Untappd	15	Nationwide
8	Cryptochange	16	Shipt

Table 11: Famous Ionic apps, Source: [41]

3.4 React Native

React Native is a free open source framework for developing cross-platform natively compiled mobile applications in JavaScript by using React JavaScript libraries [42].



Figure 31: React Native logo,

Source: <https://developers.pendo.io/react-native-plugin-2-6-1-ios-android-beta/>

The current React Native architecture is based on three major components:

- ***The JavaScript Thread:*** This is the place where the entire JavaScript code is placed and compiled. When the app is bundled for production, the JavaScriptCore runs the bundle when the user starts the app [42] [43].
- ***The Native Thread:*** This is the place where the native code is executed. This component handles the user's interface and ensures seamless communication with the JS thread whenever the app needs to update the UI, run native functions, etc [43]. All the native modules lie in the startup, which means they will always be bundled if the user wants to access them [42].
- ***The Shadow Thread:*** It is the place where the layout of an application is calculated [43]. It transforms flexbox layouts, calculates them and sends them to the app's interface [42].

By default, iOS and Android operating systems cannot run applications written in JavaScript. A common idea for developers to work out the differences in code would

be to translate JavaScript into Objective-C and Java [42] [44]. This however is troublesome due to the fact that both native programming languages are strongly typed while JavaScript is not. Instead of translating the code, React Native uses JavaScriptCore and React Native Bridge under the hood. JavaScriptCore is an open source JavaScript engine that is natively used on iOS for running JavaScript in Safari browser [45]. When React Native app is compiled for Android, the engine is bundled into the application together with the source code. React Native Bridge communicates between the JavaScript code and native code during the execution of the application.

When the application gets executed, the native code starts the JavaScript engine and thus the JavaScript thread. The business logic written in JavaScript is then executed on the JavaScript thread by the engine, which will make asynchronous calls through the Bridge to the main native thread that will carry out the tasks given. Once the native thread has done a given task, a response is sent back using the Bridge [44] [45].

Because the communication between the two executions are asynchronous, there is no blockage, in other words no waiting period in the execution between tasks. This allows better performance when rendering the view. By contrast, if the calls would be synchronous instead of asynchronous, the execution would have to be paused on one thread while waiting for the other thread to execute the task and to respond [45]. This would cause unresponsiveness when using the application [44].

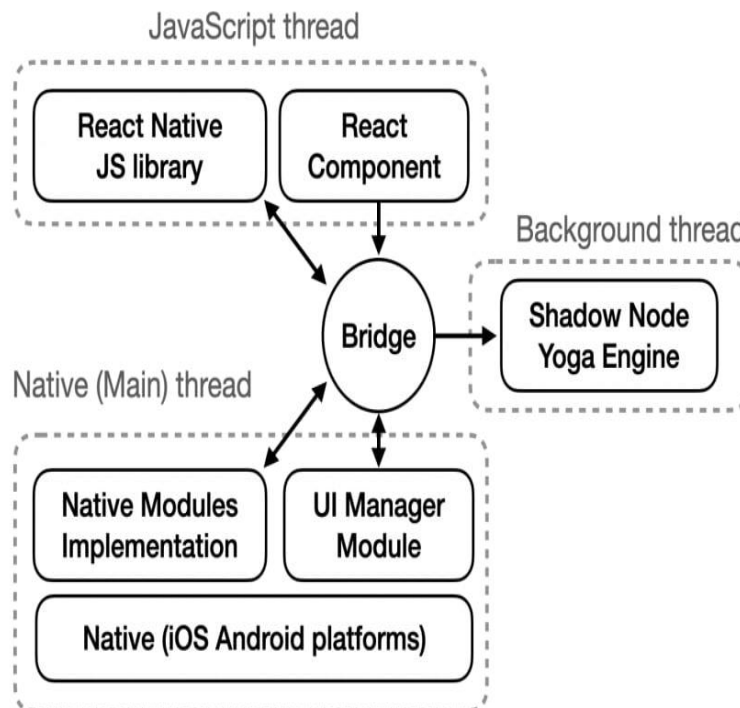


Figure 32: React Native architecture, Source: [44]

Internet technologies for the development of compatible applications on different computing platforms

The system compatibility for the React Native framework and the recommended development environment are shown in the following table:

React Native	Development Environment	Operating System
Windows	Visual Studio 2019 (version 16.5 or higher)	Windows 10 version 10.0.16299.0 (aka 1709, aka Redstone 3, aka Fall Creators Update) or higher
MacOs	Xcode 11.3.1 or newer	MacOS High Sierra (10.13) or newer version

Table 12: React Native system requirements, Source: [46]

3.4.1 Advantages of React Native

The advantages of using React Native are:

- ⊕ Time and Cost Saving: Saves time and cost to build mobile apps on multiple platforms [47].
- ⊕ Free and Open Source: This is important because anyone can use it [44].
- ⊕ Community Support: Being an open source framework, React Native enables the entire community of developers to examine all documentation concerning this technology free of any charge and also allows them to contribute to it whenever they want [43] [47]. It should be noted that it has one of the largest communities, as GitHub Stars reaches 98,500 stargazers.
- ⊕ Optimal Performance: The effectiveness of the platform lies in enhancing performances with the use of native modules and controls. It operates by interacting with Android and iOS native components, and proceeds to generate codes to native APIs, independent of any interference [44].
- ⊕ Code Reusability and Pre-Developed Components: Pre-developed components in the open source library enable developers to freely access codes [43]. These codes are already written and the developers will be getting ready to deploy them, which are resulting in faster development and reuse of up to 90% of the code.
- ⊕ Simplified UI: The driving force behind the application of React Native Technology is that assures a simplified and uncomplicated mobile user interface. The JavaScript library resembles an open source framework rather than a conventional framework. Usually, apps built with React Native have a more responsive UI, seamless UX and take less time to load [44] [43].
- ⊕ Third-Party Plugin Support: The main framework of React Native lacks certain components. To make up for that shortfall, it ensures that developers have access to third-party plugins like JavaScript modules and native modules [44] [47].

Internet technologies for the development of compatible applications on different computing platforms

- ⊕ Modular Architecture: a popular software design technique, modular programming ensures the segregation of program functions into free and interchangeable blocks called modules. It makes the application building process more adjustable by helping developers use each other's projects whenever required. It also improves team collaboration required to generate and receive updates [47].
- ⊕ Online Coding Playground: It is a browser-based environment for developing apps. It is a great place to get started learning a framework, as a developer can develop apps without needing to install the various SDKs and tools needed for native iOS and Android development.
- ⊕ Commonly-used Programming Language: According to Stack Overflow's 2020 Developer Survey¹¹, JavaScript currently stands as the most commonly-used language in the world (69.7%).

3.4.2 Disadvantages of React Native

The disadvantages of using React Native are:

- ⊖ Low Security: React Native is a JavaScript library and also an open source framework, due to which developers often face the challenge of keeping the app secure. Otherwise, malicious code snippets can pose a critical threat to the app's safety features [44] [47].
- ⊖ Longer Initialization Time: Another problem coders have with React Native is that it takes longer to initialize the runtime before it can render properly the first time. This problem exists even with hi-tech devices and could be caused as JavaScript threads usually take more time to initialize [44].
- ⊖ Memory Management: React Native is also not suitable for use in computation-intensive apps because it is based on JavaScript. React Native reduces performance and speed in these applications, and float computations are also handled in an inefficient fashion, making memory management and usage quite difficult [44].
- ⊖ Maintenance: Another issue that React Native has is that the tool is always adapting and changing to newer versions meaning that it is a high maintenance framework. A developer must be ensuring that they are able to know how to use current version of React Native that is required [44] [47].

¹¹ Stack Overflow is a question and answer website for programmers.

3.4.3 React Native App

In order to build a React Native app, it is necessary to set up the proper environment. The table below shows all the prerequisites depending on the operating system development.

Prerequisites for React Native		
Development OS	Target OS	Dependencies
<i>macOS</i>	Android	<ul style="list-style-type: none">• Node 12 or newer• Watchman¹²• JDK 8 or newer• Android Studio• React Native CLI
	iOS	<ul style="list-style-type: none">• Node 12 or newer• Watchman• JDK 8 or newer• Xcode 10 or newer• CocoaPods¹³
<i>Windows</i>	Android	<ul style="list-style-type: none">• Node 12 or newer• JDK 8 or newer• Android Studio• React Native CLI
<i>Linux</i>	Android	<ul style="list-style-type: none">• Node 12 or newer• JDK 8 or newer• Android Studio• React Native CLI

Table 13: React Native prerequisites, Source: [42]

However, through the Expo online editor, a developer can very easily create an application and run it immediately through the emulators available on the page. Expo is a set of tools and services built around React Native and native platforms that help to develop, build, deploy, and quickly iterate on iOS, Android, and web apps from the same JavaScript codebase. It has four options. A developer can run the application on either Android or iOS phone. The application can also run on the web and on developer's smartphone after the developer have first downloaded the application and having scanned the barcode.

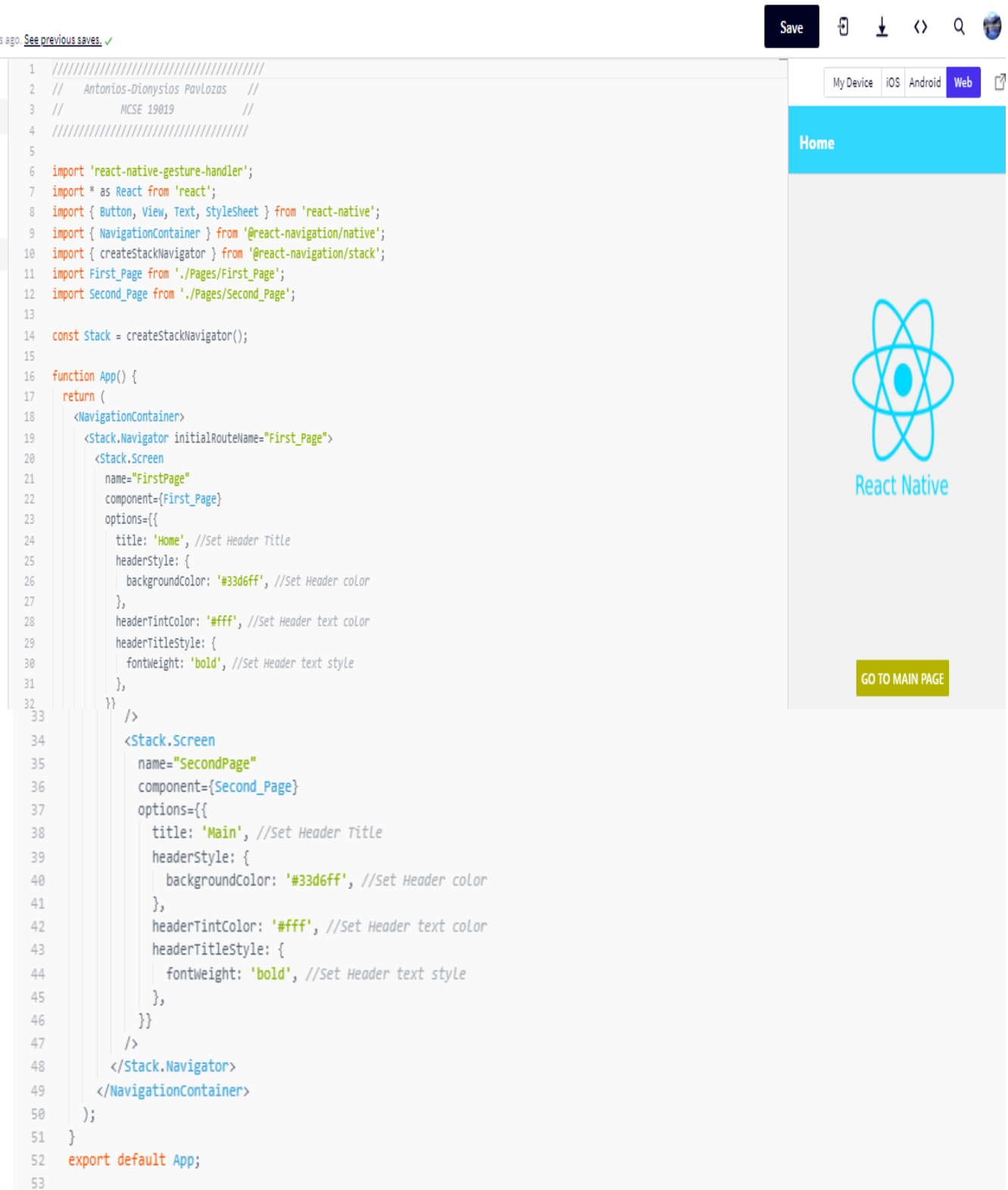
A big advantage of using an online editor is that the developer can quickly build an application and run it in minutes without having to install a program. All that is required is an internet connection. It is also convenient for inexperienced users to test each component and function with a better understanding of their usefulness.

¹² Watchman is a tool by Facebook for watching changes in the file system. It is highly recommended to install for better performance.

¹³ If the app is using Flutter plugins, CocoaPods dependency manager is needed to be installed.

Internet technologies for the development of compatible applications on different computing platforms

The application consists of two pages. The first page contains a logo of React Native and a button that redirects to the second page. The second page contains two informative paragraphs about React Native.



```
1 ///////////////////////////////////////////////////////////////////
2 //  Antonios-Dionysios PavLozas  //
3 //  MCSE 19019  //
4 ///////////////////////////////////////////////////////////////////
5
6 import 'react-native-gesture-handler';
7 import * as React from 'react';
8 import { Button, View, Text, StyleSheet } from 'react-native';
9 import { NavigationContainer } from '@react-navigation/native';
10 import { createStackNavigator } from '@react-navigation/stack';
11 import First_Page from './Pages/First_Page';
12 import Second_Page from './Pages/Second_Page';
13
14 const Stack = createStackNavigator();
15
16 function App() {
17   return (
18     <NavigationContainer>
19       <Stack.Navigator initialRouteName="First_Page">
20         <Stack.Screen
21           name="FirstPage"
22           component={First_Page}
23           options={{
24             title: 'Home', //Set Header Title
25             headerStyle: {
26               backgroundColor: '#33d6ff', //Set Header color
27             },
28             headerTintColor: '#fff', //Set Header text color
29             headerTitleStyle: {
30               fontWeight: 'bold', //Set Header text style
31             },
32           }}
33         />
34         <Stack.Screen
35           name="SecondPage"
36           component={Second_Page}
37           options={{
38             title: 'Main', //Set Header Title
39             headerStyle: {
40               backgroundColor: '#33d6ff', //Set Header color
41             },
42             headerTintColor: '#fff', //Set Header text color
43             headerTitleStyle: {
44               fontWeight: 'bold', //Set Header text style
45             },
46           }}
47         />
48       </Stack.Navigator>
49     </NavigationContainer>
50   );
51 }
52 export default App;
53
```

Figure 33: React Native first page code

Internet technologies for the development of compatible applications on different computing platforms

The above page is the main page of the application. In the first lines, the dependencies and the JS files are imported, which are referred to on the main page, then the function “CreateStackNavigator” is called, which is a function that returns an object containing two properties: Screen and Navigator. Both of them are React components used for configuring the navigator. The Navigator should contain Screen elements as its children to define the configuration for routes. Each screen in the navigator can specify some options for the navigator, such as the title to render in the header, the background color of the navigation bar, the color of the header title and the header text style. These options can be passed in the options prop for each screen component. A NavigationContainer is a component which manages the navigation tree and contains the navigation state. This component must wrap all navigators’ structure and it is placed at the root of our app.

In order to run the application in a smartphone, the following actions need to be taken:

1. Download the Expo Go
2. Open the Expo Go
3. Scan the barcode

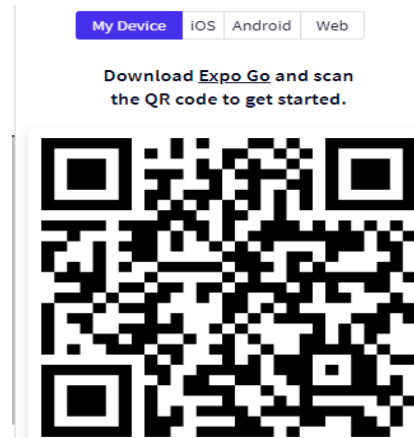


Figure 34: React Native barcode

The below images show the results of the App.

Internet technologies for the development of compatible applications on different computing platforms

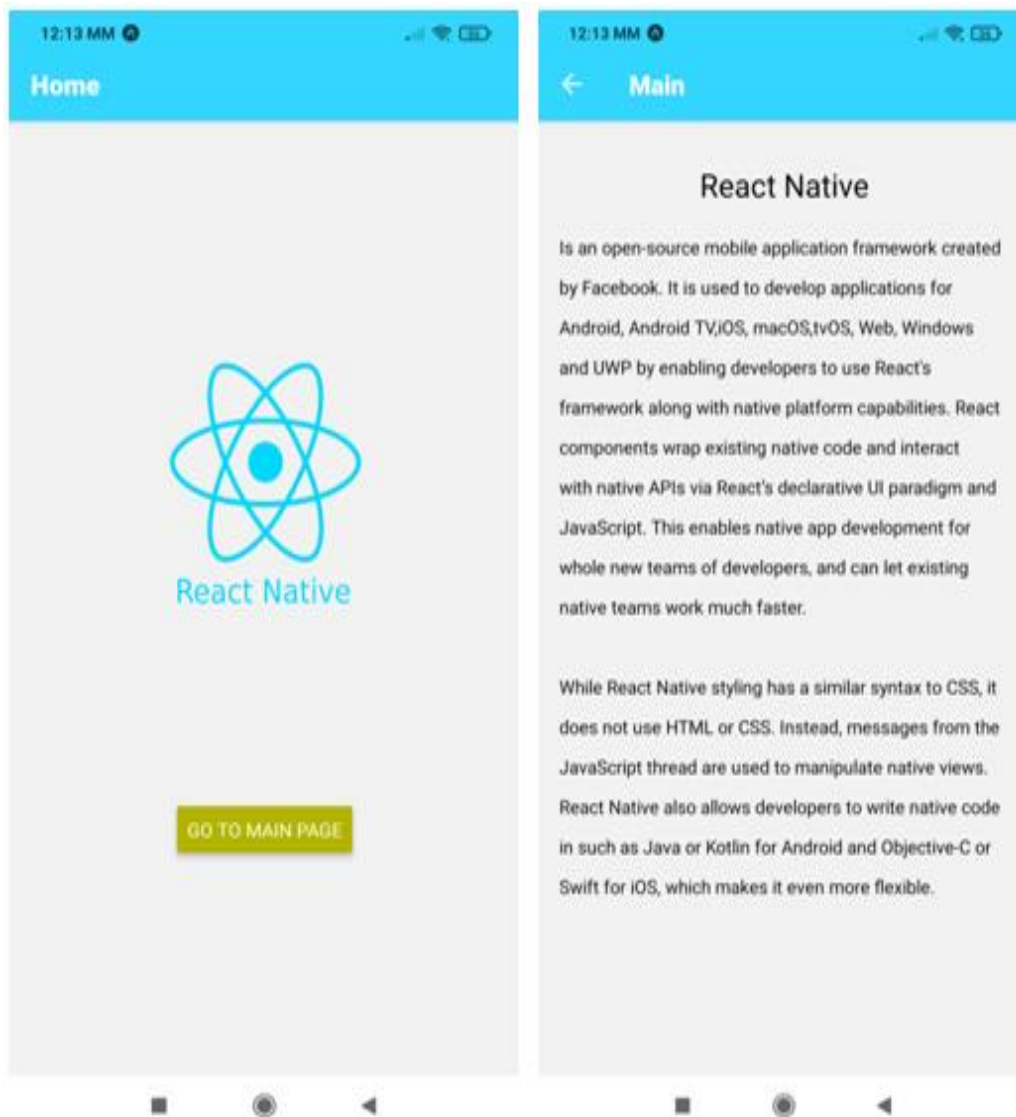


Figure 35: React Native app

Some of the most famous apps, which are designed with React Native are:

React Native Apps			
1	Facebook	9	Pinterest
2	Facebook Ads Manager	10	Tesla
3	Facebook Analytics	11	Vogue
4	Instagram	12	Uber Eats
5	Coinbase	13	Oculus
6	Shopify	14	FlipKart
7	Tableau	15	Mercari
8	Skype	16	Adidas GLITCH

Table 14: Famous React Native apps, Source: [48]

Internet technologies for the development of compatible applications on different computing platforms

3.5 Flutter

Flutter is a free open-source mobile SDK developer that can use to build native-looking Android and iOS applications from the same code base [49].



Figure 36: Flutter logo,

Source: <https://commons.wikimedia.org/wiki/File:Google-flutter-logo.png>

Flutter's core technologies are Dart, a programming language developed by Google [49]. The language has been optimized for building user interfaces. This makes it a good fit for the Flutter framework. The language is fairly easy to pick up, especially if a developer has a background in JavaScript and object-oriented programming generally [50]. Flutter architecture is composed by three major components as shown in the image below:

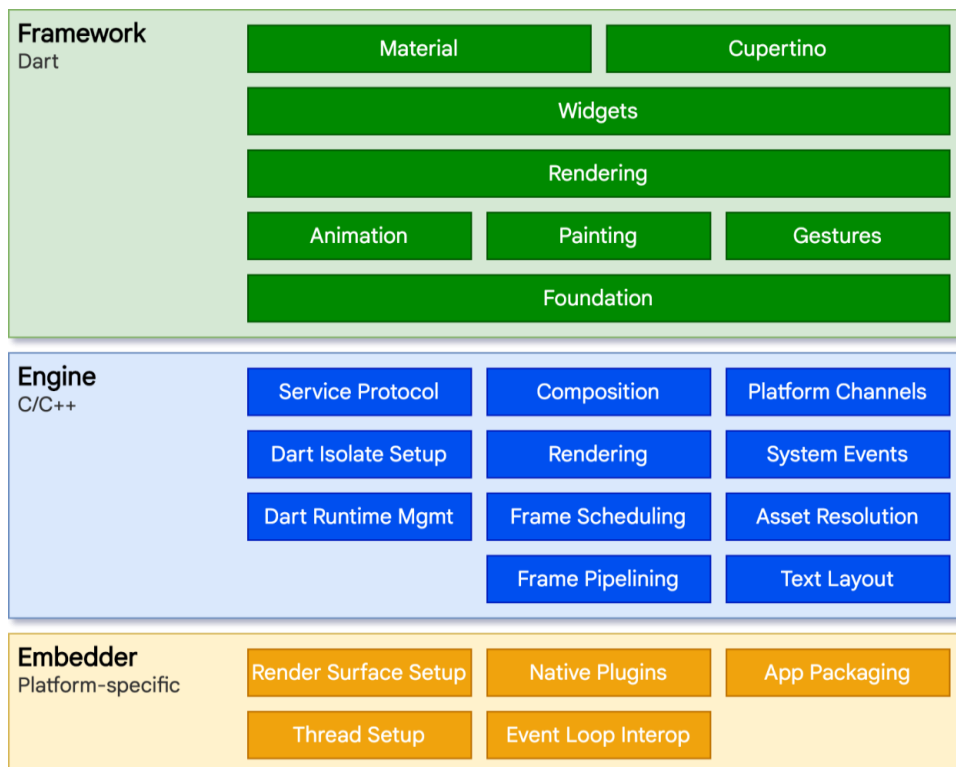


Figure 37: Flutter architecture, Source: [50]

- **Dart Platform**: Flutter runs in the Dart virtual machine which features a just-in-time execution engine. While writing and debugging an app, Flutter uses

Internet technologies for the development of compatible applications on different computing platforms

just-in-time compilation, allowing for “hot reload”, with which modifications to source files can be injected into a running application [49] [50].

- **Flutter Engine**: It is written primarily in C++. It implements Flutter’s core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and tool chain for developing, compiling, and running Flutter applications [51]. Flutter’s engine takes core technologies, Skia, a 2D graphics rendering library, and Dart, a VM for a garbage-collected object-oriented language, and hosts them in a shell. Different platforms have different shells, for example we have shells for Android and iOS [49] [50].
- **Framework is composed by Foundation library and Widgets**: The Foundation library is written in Dart, provides basic classes and functions which are used to construct applications using Flutter, such as APIs to communicate with the engine. Flutter development is all around the Widgets. The central idea is that a developer builds his UI out of widgets [49] [51]. Widgets describe what their view should look like given their current configuration and state [50]. Working from the bottom to the top, the architecture is described:
 - *Basic foundational classes and building block services* such as animation, painting, and gestures that offer commonly used abstractions over the underlying foundation [51] [52].
 - *The rendering layer* provides an abstraction for dealing with layout. With this layer, a developer can build a tree of renderable objects, and manipulate these objects dynamically, with the tree automatically updating the layout to reflect the changes [51] [52].
 - *The widgets layer* is a composition abstraction. Each render object in the rendering layer has a corresponding class in the widgets layer. In addition, the widgets layer allows defining combinations of classes that a developer can reuse. This is the layer at which the reactive programming model is introduced [51] [52].
 - *The Material and Cupertino libraries* offer comprehensive sets of controls that use the widget layer’s composition primitives to implement the Material or iOS design languages [51] [52].

The system compatibility for the Flutter framework and the recommended minimum development environment are shown in the following table:

Flutter	Tools	Operating System	Disk Space
Windows	Windows PowerShell 5.0 or newer, Git for Windows 2.x	Windows 7 SP1 or newer (64-bit)	1.64 GB (does not include disk space for IDE/tools)
MacOs	Xcode	macOS (64-bit)	2.8 GB (does not include disk space for IDE/tools)
Linux	bash ,curl, git 2.x, mkdir, rm, unzip, which, xz-utils	Linux (64-bit) with libGLU.so.1 library	600 MB (does not include disk space for IDE/tools)
Chrome OS	bash ,curl, git 2.x, mkdir, rm, unzip, which, xz-utils	Chrome OS (64-bit) with Linux (Beta) turned on and libGLU.so.1 library	600 MB (does not include disk space for IDE/tools)

Table 15: Flutter system requirements, Source: [53]

3.5.1 Advantages of Flutter

The advantages of using Flutter are:

- ⊕ Integration of Flutter with existing Android and iOS apps. Flutter can be integrated into the existing application piecemeal, as a library or module. That module can then be imported into Android or iOS (currently supported platforms) app to render a part of the app's UI in Flutter. Or, just to run shared Dart logic [49] [51].
- ⊕ Flutter can be developed on multiple IDEs [54].
- ⊕ Flutter has whole suites of web-based tooling of debugging and inspecting Flutter applications [49].
- ⊕ Dart is the programming language for Flutter. Dart has a clean and incredibly powerful syntax that sets ideal conditions for creating a clear architecture and design of the application, not to mention ensuring coherence of programmers working together, durability, ease of maintenance and much more than most other tools for cross-platform development [51] [54].
- ⊕ Dart compiles to native machine code for mobile and desktop, as well as to JavaScript for the web [54].
- ⊕ Flutter is free and open source [51].
- ⊕ Shared UI and Business logic for all platforms. The Flutter cross-platform app usually follows the principles of Material Design, with minor changes for different platforms (which Flutter usually processes on its own) or makes the entire user interface look individual, with elements based on the best solutions taken from iOS and Android [49] [51].
- ⊕ Native App Performance [54]. The single code provides native performance with 50-90% of reusable code.

Internet technologies for the development of compatible applications on different computing platforms

- ⊕ Hot Reload. Any changes made by developers will be instantly seen in the app. This will help fix bugs within seconds as the developer will be able to see the change on the go. This feature will also help the developer experiment with new features and improvise [52].
- ⊕ Large Community Support [51]. It should be noted that it has the largest communities, as GitHub Stars reaches 130,000 stargazers.
- ⊕ Online Coding Playground which called FlutLab. Flutlab is a modern Flutter online IDE and the best place to create, debug, and build cross-platform projects[49] [52].

3.5.2 Disadvantages of Flutter

The disadvantages of using Flutter are:

- ⊖ Large File Sizes. They occupy a lot of space and take longer time to download or update [54].
- ⊖ Issues with iOS. Flutter is developed by Google and since Google is directly interested in fixing bugs in the shortest amount of time, building Android apps on Flutter is fast and enjoyable. Thus, the updates on iOS significantly delayed [51] [54].
- ⊖ Lack of Third-party Libraries. Third-party libraries and packages have a significant impact on software development as it enables some features for developers. These third-party libraries are normally free, open-source, pre-tested, and easily available. However, since Flutter is new for mobile app development, it is not easy to find such free packages and libraries [52] [54].
- ⊖ Dart is also pretty immature. Not many beginners will be able to develop an app using this language [52] [54].
- ⊖ Performance. The look and feel is not 100% the same as with native solutions. Basically, Flutter does not create native components. It replicates Android's Material Design and iOS-specific components with its Cupertino library, but it is not exactly the same. It is visible especially with different system versions where text fields or buttons vary from one another, yet stay the same in Flutter [54].

3.5.3 Flutter App

In order to build a Flutter app, it is necessary to set up the proper environment. The table below shows all the prerequisites depending on the operating system development.

Internet technologies for the development of compatible applications on different computing platforms

Prerequisites for React Native		
Development OS	Target OS	Dependencies
<i>macOS</i>	Android	<ul style="list-style-type: none"> Flutter SDK Android Studio 2.2 or higher
	iOS	<ul style="list-style-type: none"> Flutter SDK Xcode CocoaPods if plugins are used
	Desktop	<ul style="list-style-type: none"> Xcode CocoaPods if plugins are used
	Web	<ul style="list-style-type: none"> Flutter SDK Chrome Android Studio 3.0 or higher / Visual Studio code / IntelliJ IDEA 2017.1 or higher Flutter and Dart plugins
<i>Windows</i>	Android	<ul style="list-style-type: none"> Flutter SDK Android Studio 2.2 or higher
	Desktop	<ul style="list-style-type: none"> Flutter SDK Visual Studio 2019
	Web	<ul style="list-style-type: none"> Flutter SDK Chrome Android Studio 3.0 or higher / Visual Studio code / IntelliJ IDEA 2017.1 or higher Flutter and Dart plugins
<i>Linux</i>	Android	<ul style="list-style-type: none"> Flutter SDK Android Studio
	Desktop	<ul style="list-style-type: none"> Flutter SDK Clang CMake GTK development headers Ninja build pkg-config
	Web	<ul style="list-style-type: none"> Flutter SDK Chrome Android Studio 3.0 or higher / Visual Studio code / IntelliJ IDEA 2017.1 or higher Flutter and Dart plugins
<i>Chrome OS</i>	Android	<ul style="list-style-type: none"> Flutter SDK Android Studio

Table 16: Flutter prerequisites , Source: [50]

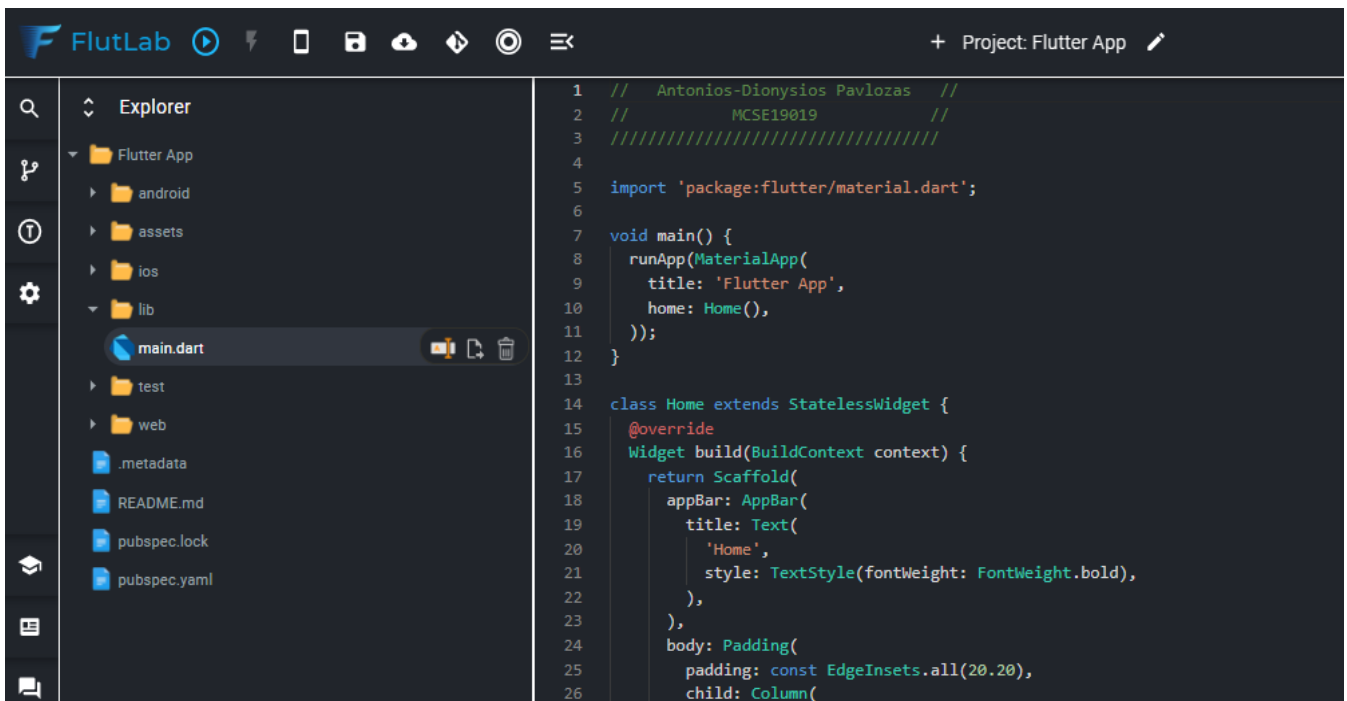
Nevertheless, a user can use directly the online editor FlutLab.io at <https://flutlab.io/>. Its main idea is to provide an alternative to local installations of Android/iOS/Flutter SDK's as well as Code Editors. Through this tool the results are visible immediately. The *main.dart* is the main file of the program and there is all the

Internet technologies for the development of compatible applications on different computing platforms

code. Everything in Flutter can be created using widgets. Flutter widgets implementing Material Design, and, in order to use, the *package: flutter/material.dart* should be imported.

The entry point of the application is the main function in which the call to the *runApp* is made. This is the first line which is executed; its task is to set up the Flutter framework and run the selected application. When the application is initially set up, it is a normal stateless widget. Next, the Build method is described. The Build method is the one that returns the *Scaffold* class, which returns a navigation bar, sets the title, and sets a general theme. In addition to this, the Build method also sets the routing of the application and the home screen.

The body contains the necessary components (logo, button, text), aligned in the center within the Scaffold widget. The whole code is depicted in the following images.



```
1 // Antonios-Dionysios Pavlozas //
2 // MCSE19019 //
3 ////////////////////////////////////////////////////
4
5 import 'package:flutter/material.dart';
6
7 void main() {
8   runApp(MaterialApp(
9     title: 'Flutter App',
10    home: Home(),
11  ));
12 }
13
14 class Home extends StatelessWidget {
15   @override
16   Widget build(BuildContext context) {
17     return Scaffold(
18       appBar: AppBar(
19         title: Text(
20           'Home',
21           style: TextStyle(fontWeight: FontWeight.bold),
22         ),
23       ),
24       body: Padding(
25         padding: const EdgeInsets.all(20.20),
26         child: Column(
```

Figure 38: FlutterLab

Internet technologies for the development of compatible applications on different computing platforms

```
1 // Antonios-Dionysios Pavlozas //
2 //           MCSE19019           //
3 ////////////////////////////////////////////////////////////////////
4
5 import 'package:flutter/material.dart';
6
7 void main() {
8   runApp(MaterialApp(
9     title: 'Flutter App',
10    home: Home(),
11  ));
12 }
13
14 class Home extends StatelessWidget {
15   @override
16   Widget build(BuildContext context) {
17     return Scaffold(
18       appBar: AppBar(
19         title: Text(
20           'Home',
21           style: TextStyle(fontWeight: FontWeight.bold),
22         ),
23       ),
24       body: Padding(
25         padding: const EdgeInsets.all(20.20),
26         child: Column(
27           mainAxisAlignment: MainAxisAlignment.center,
28           children: <Widget>[
29             SizedBox(
30               child: Image(
31                 image: AssetImage('assets/logo.png'),
32                 height: 450,
33                 width: 450,
34               ),
35             ),
36             ElevatedButton(
37               style: ButtonStyle(backgroundColor: MaterialStateProperty.all<Color>(Colors.blue)),
38               child: Text('GO TO MAIN PAGE', style: TextStyle(fontSize: 17)),
39               onPressed: () {
40                 Navigator.push(
41                   context,
42                   MaterialPageRoute(builder: (context) => Main()),
43                 );
44               },
45             ),
46           ],
47         ),
48       ),
49     );
50   }
51 }
52
53 class Main extends StatelessWidget {
54   @override
55   Widget build(BuildContext context) {
56     return Scaffold(
57       appBar: AppBar(
58         title: Text(
59           "Main",
60           style: TextStyle(fontWeight: FontWeight.bold),
```


Internet technologies for the development of compatible applications on different computing platforms

```
61     ),
62   ),
63   body: Padding(
64     padding: const EdgeInsets.all(20.20),
65     child: Column(
66       mainAxisAlignment: MainAxisAlignment.center,
67       children: <Widget>[
68         Text(
69           "Flutter\n",
70           style: TextStyle(fontSize: 24),
71         ),
72         Text(
73           "Flutter is an open-source UI software development kit created by Google. It is used",
74           style: TextStyle(height: 2, fontSize: 13),
75         ),
76       ],
77     ),
78   ),
79 );
80 }
81 }
82 }
```

Figure 39: Flutter app code

FlutterLab gives the ability to run the application on either android mobile, or iOS mobile or as a web application. It is also possible to run the application directly on a mobile phone and thus, watching the application in real time. To do this it is necessary to make the Apk file from FlutterLab through the available options (Android, iOS). If there are no errors, the following message will be displayed:

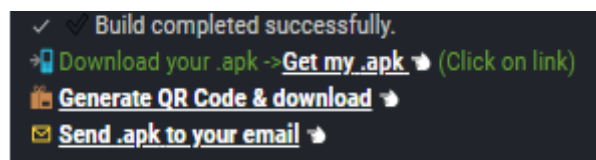


Figure 40: Apk message

Then by clicking on the option *Generate QR code & download* and scanning the barcode on the mobile, the following images appear:

Internet technologies for the development of compatible applications on different computing platforms

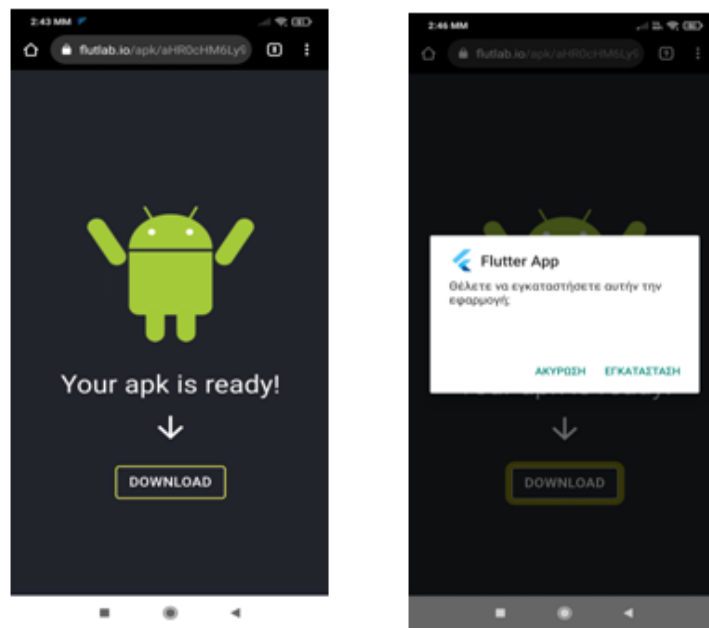


Figure 41: Apk installation

The application is shown in the images below:

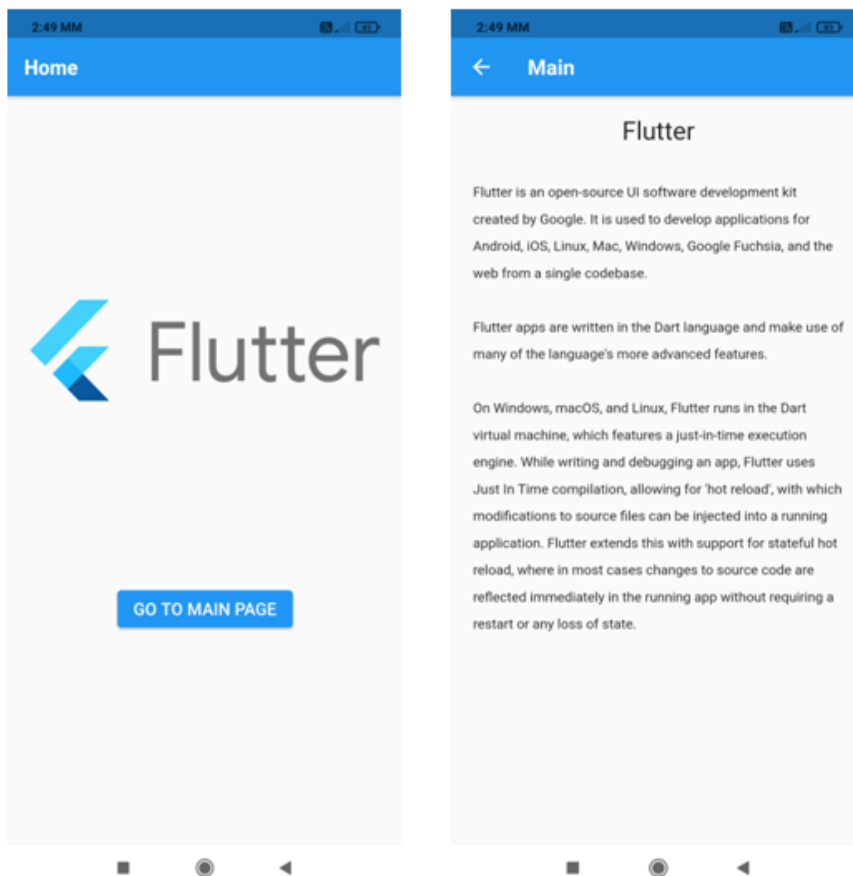


Figure 42: Flutter app

Internet technologies for the development of compatible applications on different computing platforms

Some of the most famous apps, which are designed with Flutter, are:

Flutter Apps			
1	Google	7	Dream 11
2	Ebay	8	SONOS
3	BMW	9	Ubank
4	Square	10	EMAAR
5	Alibaba Group	11	The New York Times
6	Capital One	12	Tencent

Table 17: Famous Flutter apps, Source: [55]

3.6 Framework7

Framework7 is a free and open source mobile HTML framework for developing hybrid mobile apps or web apps with iOS and Android native look and feel. Framework7 aims to give developers with knowledge of JavaScript, HTML and CSS the ability to create amazing mobile apps [56]. Framework7 apps can be compiled to Cordova so as to utilize device features like Geolocation, Camera and Contacts. Once the Framework7 apps have been compiled to Cordova apps, they can be published to iOS App Store or Google Play.



Figure 43: Framework7 logo, Source: <https://framework7.io/>

Its interface is effortless to use as it has syntax similar to jQuery. Furthermore, it has a built-in FastClick library, which gives it commendable speed. Sensitive, the tool integrates a grid system to organize the elements reactively [57]. Moreover, it loads the template pages through a flexible router API, which makes it fully dynamic. Framework7, therefore, has specific characteristics:

- ***Easy to Use***: Designing a mobile application using Framework7 is as easy as creating a website. It gives the user a freedom to use HTML code with the linkage of CSS and JavaScript files. It does not force the user to create custom tags or JSON Objects [56] [57].
- ***UI Components***: Framework7 provides ready to use UI elements and widgets like modals, pop-ups and others. All a developer needs to do is just drag and drop the element if desired [56] [57].

Internet technologies for the development of compatible applications on different computing platforms

- ***iOS specific***: It is an iOS specific framework as it provides iOS like UI interaction (it also provides swipe back feature). It was originally built for iOS users and later adopted for android [56] [57].
- ***Material Theme***: Its material theme is quite classic as it is built upon Google material design specification in order to bring pixel perfect design [56] [57].

The system compatibility for the Framework7 and the recommended development environment and tools are shown in the following table:

Framework7	Development Environment	Operating System	Tools		
Windows	Visual Studio 2017 or 2015	Windows 10	Framework7 CLI	Npm v1.4.28 or newer	Node.js v0.10.35 or newer
	Visual Studio 2015 or 2013	Windows 8.1	Framework7 CLI	Npm v1.1.69 or newer	Node.js v0.8.16 or newer
MacOs	Xcode 11.0	MacOS Mojave (10.14.4) or newer	Framework7 CLI	Npm v3.10.10 or newer	Node.js v6.11.5 or newer

Table 18: Framework7 system requirements, Source: [58]

3.6.1 Advantages of Framework7

The advantages of using Framework7 are:

- ⊕ Framework7 can also be used with Angular and React frameworks [56].
- ⊕ Framework7 does not depend on any third-party library, even for DOM manipulation. Instead, it has its own DOM7. It also supports faster development through Bower¹⁴ [56] [57].
- ⊕ By using Framework7, a developer can easily develop apps for iOS and Android without learning it [56] [57].
- ⊕ No need to readjust the code when changes of platforms occurred [56].
- ⊕ Framework7 contains many pre-styled widgets/components. It also has built-in help libraries [56] [57].
- ⊕ The Framework allows one-time maintenance [57].
- ⊕ It is free and open source [57].
- ⊕ The single code provides native performance by reusing up to 90% of the code.

3.6.2 Disadvantages of Framework7

The disadvantages of using Framework7 are:

¹⁴ Bower is a package manager for the web.

Internet technologies for the development of compatible applications on different computing platforms

- ⊖ It does not support platforms other than iOS and android [56] [57].
- ⊖ The online community support for Framework7 framework is less compared to iOS and Android [57].
- ⊖ Framework 7 is mostly built around the Apple environment. For this reason, themes can have a perfectible rendering on Android devices [56].
- ⊖ It has not an online coding playground [56] [57].
- ⊖ Small community [57]. GitHub Stars reaches just 16,400 stargazers.

3.6.3 Framework7 App

The most recommended way to start creating a new Framework7 app is to use Framework7 CLI. The commands to be typed in the command prompt are the following:

No	Commands
1	\$ pkg install node js
2	\$ npm install
3	\$ npm install -g cordova
4	\$ npm install -g framework7-cli

Table 19: Framework7 installation commands, Source: [57]

An example of code is creating a subnavbar, which is useful if there is a need to put any additional elements into Navbar, like Tab Links or Searchbar. It also remains visible when Navbar hidden. Thus, the subnavbar increase the discoverability of the content in lower-level categories of site architecture with the fewest number of clicks.

```
<div class="page">
  <div class="navbar">
    <div class="navbar-bg"></div>
    <div class="navbar-inner">
      <div class="title">Sub Navbar</div>
      <div class="subnavbar">
        <div class="subnavbar-inner">
          <div class="segmented segmented-strong">
            <a class="button tab-link tab-link-active" href="#tab1">Tab 1</a>
            <a class="button tab-link" href="#tab2">Tab 2</a>
            <a class="button tab-link" href="#tab3">Tab 3</a>
            <span class="segmented-highlight"></span>
          </div>
        </div>
      </div>
    </div>
  </div>
  <div class="page-content hide-navbar-on-scroll">
    <div class="tabs">
      <div class="tab tab-active" id="tab1">
        <div class="block">
          <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed nisi nisi, tincidunt ut eleifend non, sagittis a elit. Sed consequat odio at leo varius viverra. Proin et purus eu justo lacinia efficitur in sed orci. Morbi dignissim accumsan felis, at malesuada lorem aliquet finibus. Duis vitae enim a mauris luctus lobortis ac vel dolor. Aenean id pretium quam. Duis sed sem ullamcorper erat venenatis feugiat sed pretium magna. Suspendisse massa lectus, dictum congue arcu quis, auctor vulputate lectus. Duis vitae finibus purus, nec condimentum dui. Sed ultricies mauris vitae libero mollis lobortis. Duis ut tortor scelerisque, volutpat metus eget, rutrum est. Integer diam ex, condimentum vel orci a, lobortis bibendum urna. Aenean porttitor, eros sit amet ultrices efficitur, libero odio rhoncus justo, quis volutpat lorem arcu eu eros. Mauris gravida euismod diam, eget finibus quam ornare non. Nam metus massa, porttitor id tempor et, pharetra id felis.</p>
        </div>
      </div>
    </div>
  </div>
</div>
```

Figure 44: Framework7 subnavbar example, Source: [59]

The results in iOS and android devices are presented below:

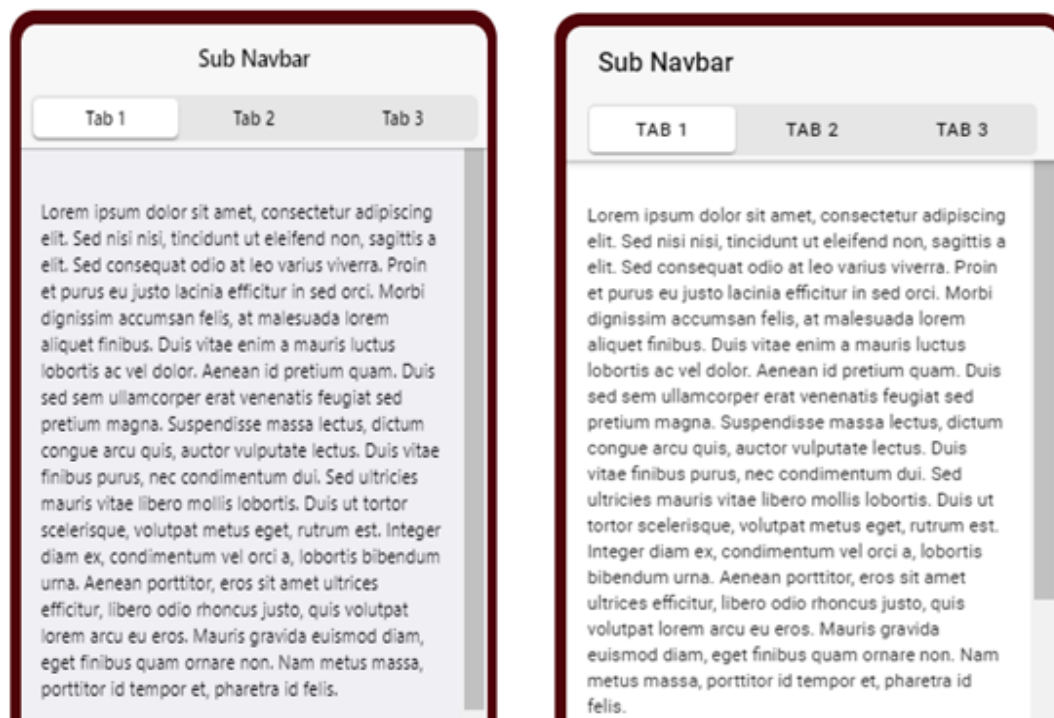


Figure 45: Subnavbar Framework7, a) iOS, b) Android, Source: [59]

Some of the most famous apps, which are designed with Framework7, are:

Framework7 Apps			
1	hafla	7	Workpose
2	EasyQuick Shop	8	Spilla
3	Link	9	Bandbiz
4	OnePile	10	Folver
5	Solver	11	TeamSpender
6	CoverID	12	Point Movie

Table 20: Famous Framework7 apps, Source: [59]

3.7 NativeScript

NativeScript is a free and open source framework for building cross-platform native mobile apps. It allows developers to use JavaScript, or any programming language that transpiles to JavaScript, such as TypeScript, in order to build apps for Android and iOS [60]. Moreover, NativeScript supports the Angular and Vue JavaScript frameworks. Unlike Cordova, which uses WebView for rendering the UI of the app, NativeScript uses the native platform's rendering engine, which means that it provides a truly native user experience.

Internet technologies for the development of compatible applications on different computing platforms



Figure 46: NativeScript logo, Source: <https://dwglogo.com/nativescript/>

In order to translate JavaScript code to the corresponding native APIs, some kind of proxy mechanism is needed. This is exactly what the "Runtime" parts of NativeScript are responsible for. The Android Runtime may be thought of as "The Bridge" between the JavaScript and Android worlds [60]. A NativeScript application for Android is standard native packages (apk) which besides the JavaScript files embed the runtime as well.

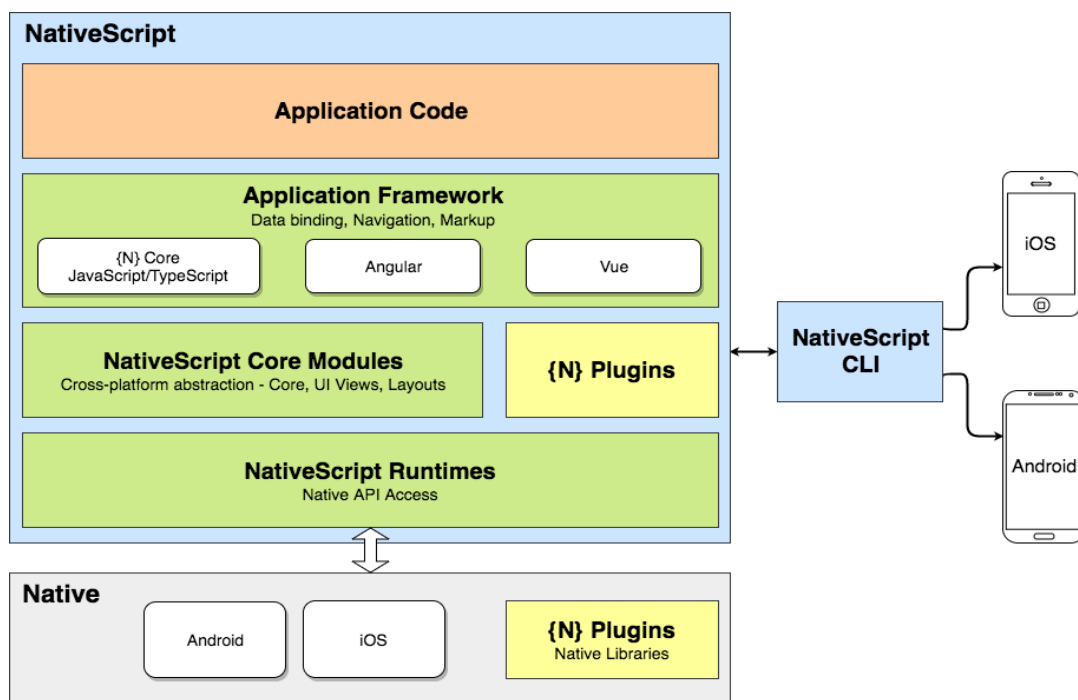


Figure 47: NativeScript architecture, Source: [60]

The Core Modules are there to provide the abstractions needed to access the underlying native platforms. Gestures module for example, it defines a common JS API that translates application TypeScript/JavaScript code into native gestures APIs calls (thanks to the Runtimes) [61].

The Runtimes enable to call APIs in the Android and iOS frameworks using

Internet technologies for the development of compatible applications on different computing platforms

JavaScript code. To do that they use JavaScript Virtual Machines - Google's V8 for Android and WebKit's JavaScriptCore implementation distributed with iOS 7.0+ [60] [61].

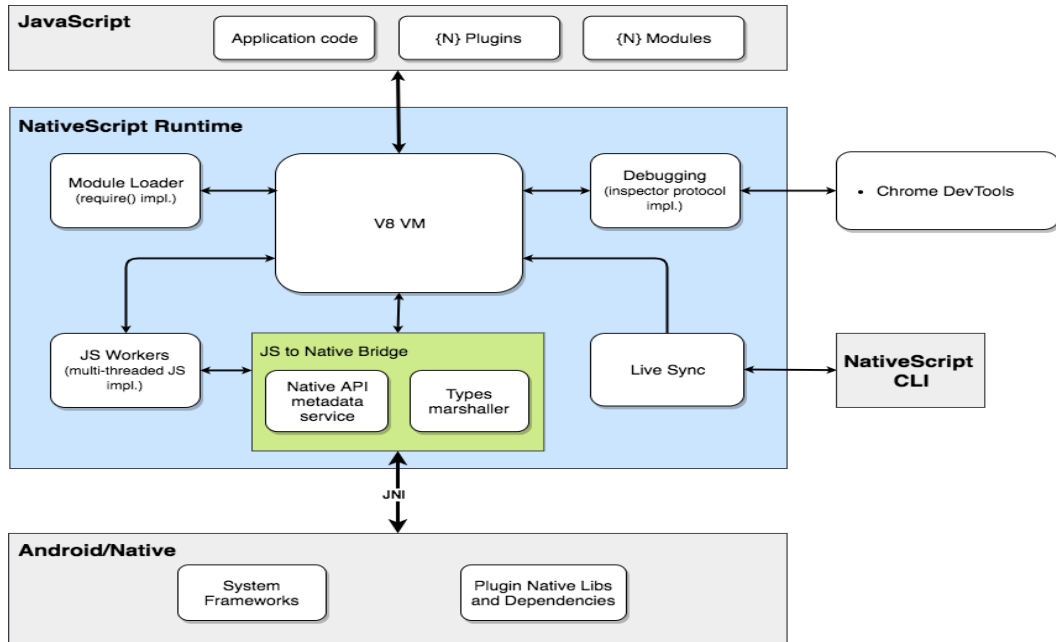


Figure 48: NativeScript runtime, Source: [61]

The NativeScript CLI is the command-line interface that lets the developer create, build, and run apps using NativeScript. The CLI runs on Windows, macOS or Linux [60] [61].

The NativeScript plugins are building blocks that encapsulate some functionality and help developers build apps faster (just like the NativeScript Core Modules, which is a plugin) [61]. Most are community-built written in TypeScript/JavaScript. Some developers can include native libraries, which are called from the TS/JS code thanks to the Runtimes.

The system compatibility for the NativeScript framework and the recommended development environment and SDK versions are shown in the following table:

NativeScript	Development Environment	Operating System	iOS Development	Android Development
Windows	Visual Studio 2019	Windows 10	-	Android 6.0 / API level 23
MacOs	Xcode 9	macOS High Sierra(10.13)	iOS 11 SDK	Android 6.0 / API level 23
Linux	Git 2.9	Ubuntu 16.04	-	Android 6.0 / API level 23

Table 21: NativeScript system requirements, Source: [62]

3.7.1 Advantages of NativeScript

The advantages of using NativeScript are:

- ⊕ Cross-technology compatibility: It uses Angular, TypeScript, or JavaScript that allows convenient data binding and more component reusability [63] [64].
- ⊕ Native functionality: It accesses the native device API via native components developed with native performance with up to 90% of reuse code [63].
- ⊕ Programming language: It uses an XML-based markup language like HTML to develop applications with customized features [64].
- ⊕ Extensibility: It gives complete and direct access to all kinds of iOS and Android APIs. This offers accessibility and allows for the reuse of free plugins, Android SDKs, and CocoaPods [63] [64].
- ⊕ Developer-friendly CLI: NativeScript CLI allows developers to do almost everything ranging from adding a platform to deploying apps to a specific platform or device. Installation of plugins and app debugging is quicker and more comfortable [64].
- ⊕ Growing Community: GitHub Stars reaches 20,500 stargazers [63].
- ⊕ Free and Open Source[63] [64]

3.7.2 Disadvantages of NativeScript

The disadvantages of using NativeScript are:

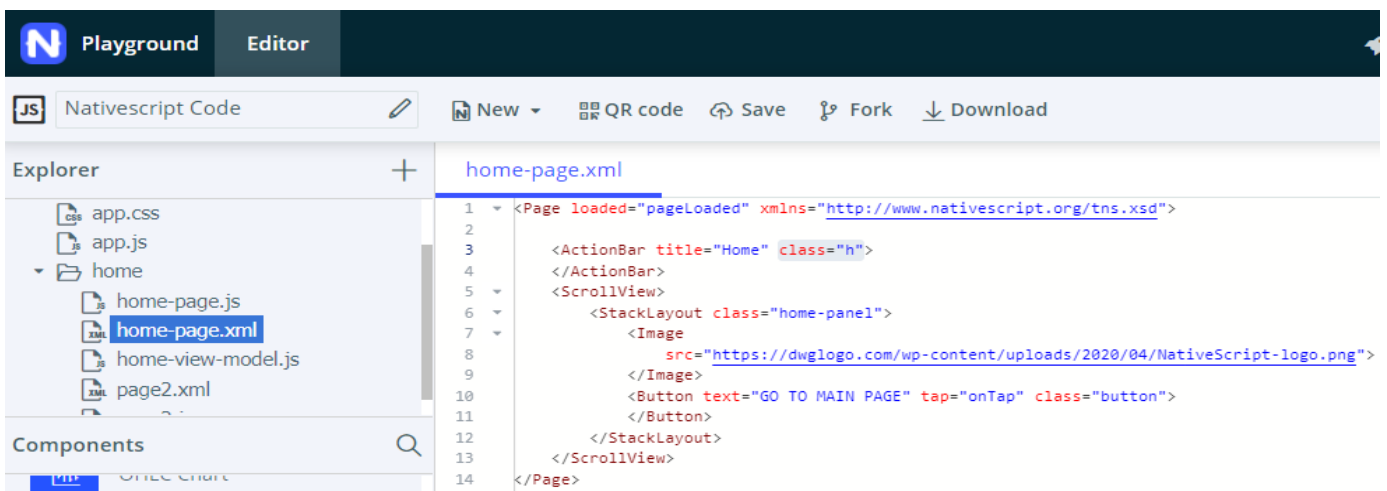
- ⊖ UI limitation: DOM and HTML are not widely supported, which leaves the necessity to learn the use of different UI components. This is time-consuming and bigger budget [63] [64].
- ⊖ Non-verified plugins: The total number of verified plugins is significantly less. Hence, there is no assurance of the quality of plugins used in this framework [63].
- ⊖ Native Knowledge: Developers must be aware of the native functionality and APIs of iOS and Android. Only then, they can access the hardware of a device and any other platform-specific elements [63] [64].
- ⊖ Slower testing: Due to its native nature, an application can only accurately be tested on an emulator or an actual device. Consequently, this slows down the initial testing rate [64].
- ⊖ React Native performs better on iOS than Android [63].

3.7.3 NativeScript App

NativeScript playground is a solution that allows a developer to write NativeScript code in a web client and deploy it to a device with the help of the NativeScript Playground app. No setup is required so a developer can start writing his code straightaway on Android and iOS, and also, using the NativeScript Preview app (Android, iOS).

When a developer navigates to <https://play.nativescript.org/>, a choice of four different templates (Angular, Vue.js, Plain TypeScript, and Plain JavaScript) will be presented to him. Once the developer chose his desired template, he will be asked to scan the QR code with the NativeScript Playground app, which will open the NativeScript preview app.

The application in Figure 49 is written in JavaScript and consists of two pages. The first page contains a logo of NativeScript and a button that redirects to the second page. The second page contains two informative paragraphs about NativeScript and a button that returns the user to the first page.



```
1 <Page loaded="pageLoaded" xmlns="http://www.nativescript.org/tns.xsd">
2
3 <ActionBar title="Home" class="h">
4 </ActionBar>
5 <ScrollView>
6 <StackLayout class="home-panel">
7 <Image
8   src="https://dwglogo.com/wp-content/uploads/2020/04/NativeScript-logo.png">
9 </Image>
10 <Button text="GO TO MAIN PAGE" tap="onTap" class="button">
11 </Button>
12 </StackLayout>
13 </ScrollView>
14 </Page>
```

Figure 49: NativeScript first page code

The home-page is the main page. The first line indicates the loading of the application screen. Then, *ActionBar* is a UI component that provides a toolbar at the top of the activity window. *ScrollView* is a UI component that shows a scrollable content area. It is important to note that *ScrollView* extends *ContentView*, so it can only have a single child element, the *StackLayout*. *StackLayout* is a layout container that contains the NativeScript logo and a button which, when pressed, takes you to the second page.

Internet technologies for the development of compatible applications on different computing platforms

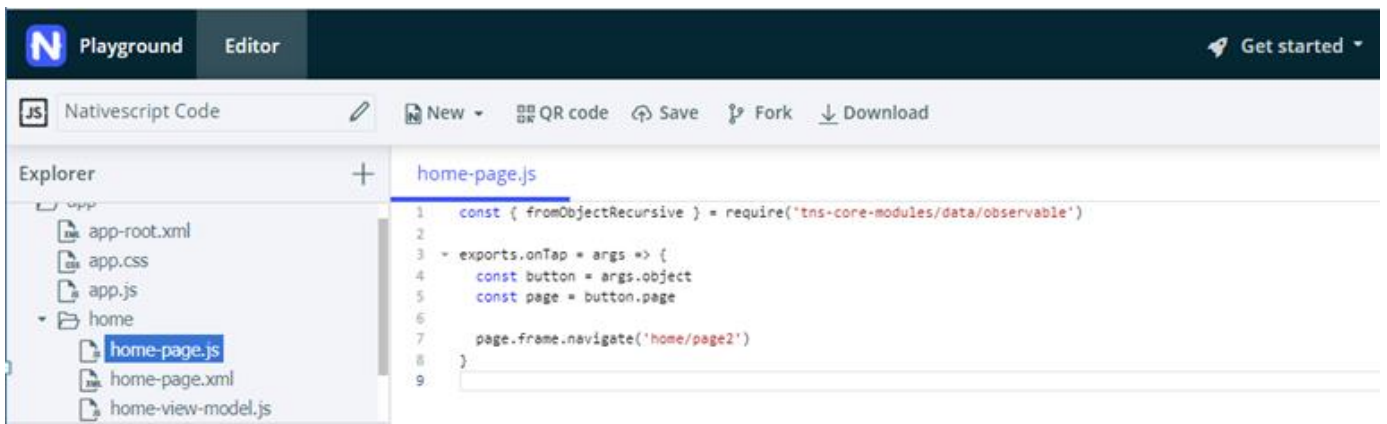


Figure 50: NativeScript first page functionality

In the above image the proper function is called to indicate which page the user should be taken to when the main page button is pressed.

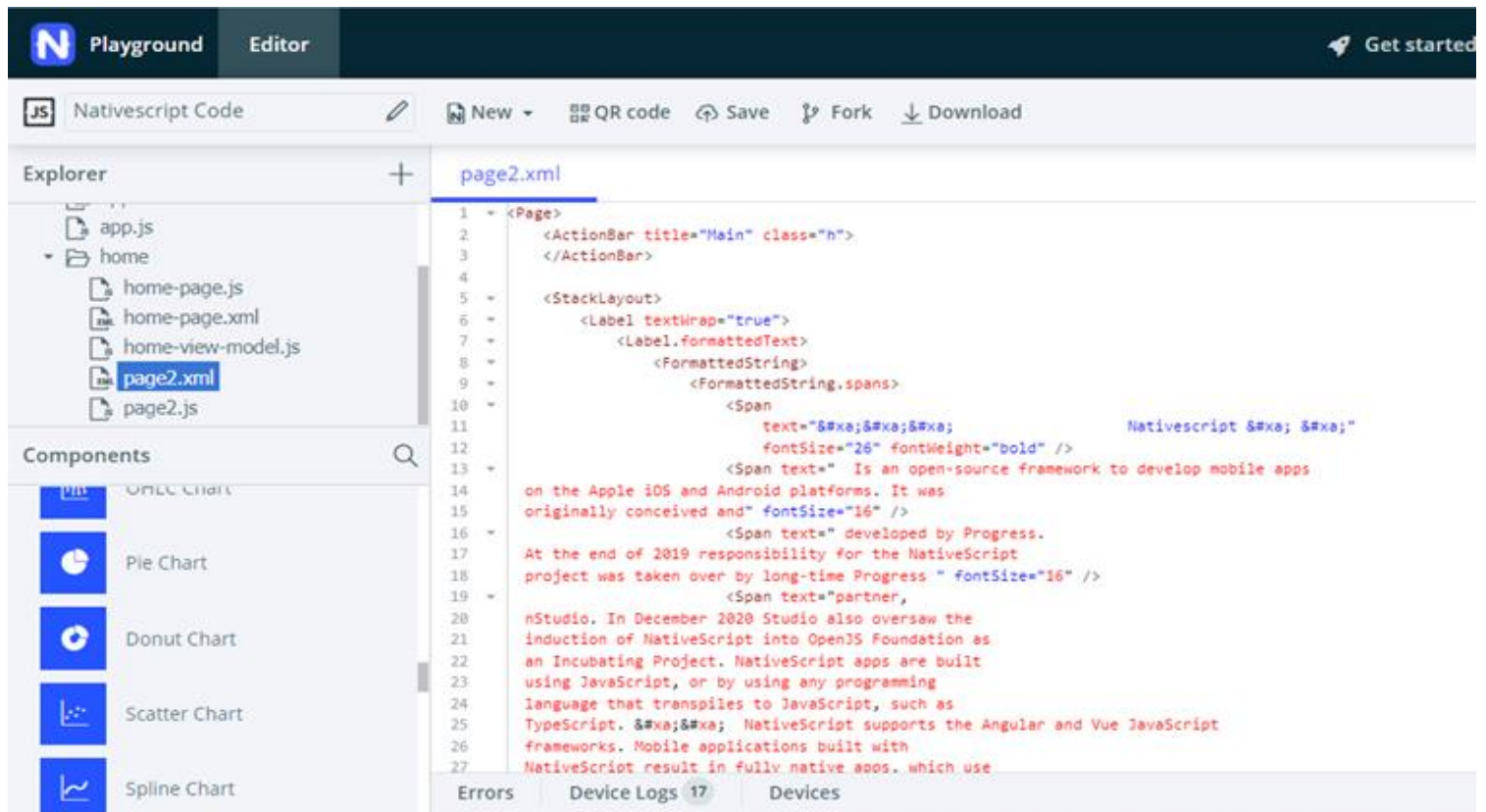


Figure 51: NativeScript second page

The new components in the second page are the *Label*, the *FormattedString* and the *Span*. The *Label* widget provides a text label that shows read-only text. *FormattedString* is a special class that is called to support various text transformations and decorations. Lastly, an observable collection of *Span* objects used to define common text properties.

Internet technologies for the development of compatible applications on different computing platforms

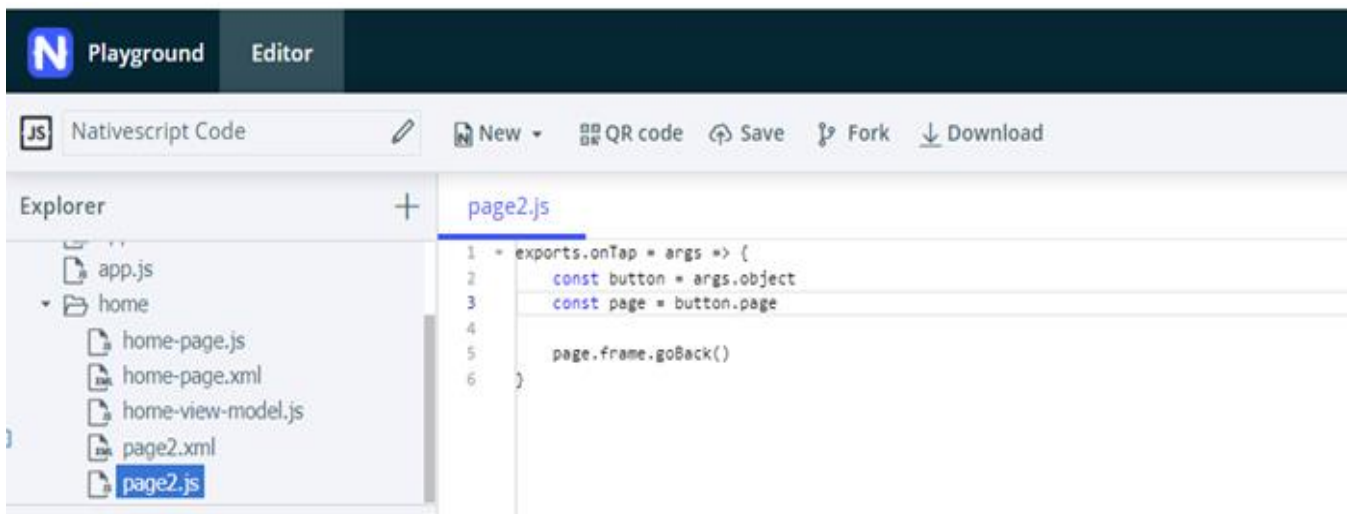


Figure 52: NativeScript second page functionality

Figure 52 depicts the function that is called to set the proper functionality of the return button.

In order to run the application in a smartphone, the following actions need to be taken:

1. Download the NativeScript Playground app
2. Open the NativeScript Playground app
3. Scan the barcode

The NativeScript Preview is automatically installed along with the NativeScript Playground app.

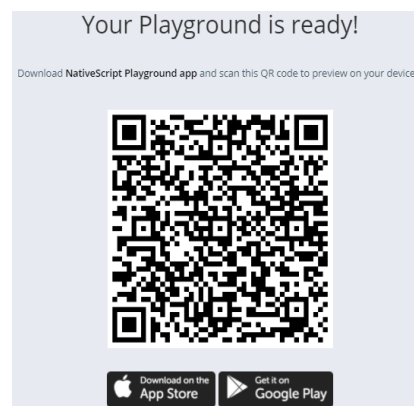


Figure 53: NativeScript barcode

The results are presented in the following image.

Internet technologies for the development of compatible applications on different computing platforms

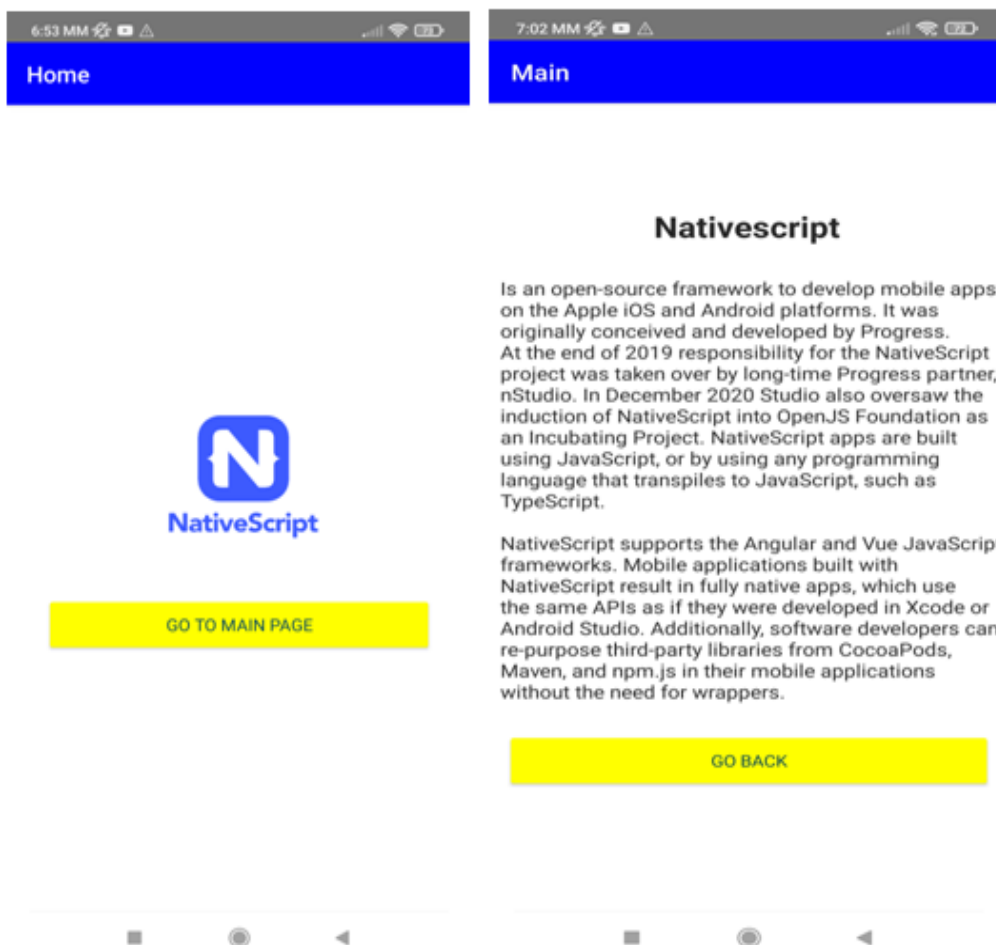


Figure 54: NativeScript app

Some of the most famous apps, which are designed with NativeScript, are:

NativeScript Apps			
1	PUMA	7	Raiffeisenbank
2	Sennheiser	8	California Court Access App
3	Airbnb Design	9	Dockbooking
4	MDBootstrap	10	Regelneef
5	Portable North Pole (PNP)	11	GeoAgro
6	SAP	12	Daily Nanny

Table 22: Famous NativeScript apps, Source: [64]

Chapter 4: Multi-criteria Decision Analysis of Best Cross-platform Framework

As discussed in the previous chapter, each framework has its own advantages and disadvantages, which some of them have the same benefits while others have a completely different structure. So, it is very important for a company and consequently for a developer to choose the right cross-platform framework to create an application, as many factors need to be considered, such as the code reuse percentage and the supported platforms.

Thus, in this chapter a comparison will be made of the similarities and differences of the seven frameworks that analyzed in chapter 3. In addition, through two methods, AHP and TOPSIS, a multi-critical decision analysis will be made where by defining certain characteristics, the ranking of each framework will be extracted.

4.1 Similarities

An important similarity is that all seven frameworks are open source, which is important as the code is publicly accessible and can be modified by anyone. Another similarity is that they are available for free with the remark that in Xamarin and Ionic only the commercial use is free. Moreover, all frameworks target, at least, the two main operating systems, Android and iOS, with a single codebase. Since everything is written in a single codebase, there is no need for separating teams working on the same app. As a result, all the cross-platform frameworks are time and cost saving.

4.2 Differences

Although these frameworks have some similarities, they have many differences too. Initially, they use a different development methodology with different programming languages with different rate of reuse code. Even though most cross-platform frameworks are free, two of them, Xamarin and Ionic, contain subscription depending on the use and the size of the company. Some of these frameworks are more famous than others and this can be seen from the GitHub community and repositories. Furthermore, as shown in the table below, Xamarin and Apache Cordova, target not only Android and iOS but also Windows Phone operating systems resulting in having a wider audience. Furthermore, some frameworks provide an online coding playground that allows for instant utility and display of code results. The following table summarizes the main differences analyzed in the previous chapter.

Attributes Frameworks	Programming Language	Code Reuse	Mobile OS	Price	Community	Playground
Xamarin	C#	60-95%	Android iOS Windows Phone	Commercial: free Business: 999\$/year Enterprise: 1899\$/year	5.4K	No
Apache Cordova	HTML,CSS, JavaScript	50-80%	Android iOS Windows Phone	Free	1K	No
Ionic	HTML,CSS, JavaScript, Angular	98%	Android iOS	Commercial: free Pro: 539\$/year Enterprise: 2999\$/year	45.3K	No
React Native	JavaScript	90%	Android iOS	Free	98.5K	Yes
Flutter	Dart	50-90%	Adroid iOS	Free	130K	Yes
Framework7	HTML,CSS, JavaScript	90%	Android iOS	Free	16.4K	No
NativeScript	JavaScript, TypeScript	90%	Android iOS	Free	20.5K	Yes

Table 23: Cross-platform's attributes matrix

4.3 Multi-criteria Decision Analysis

A multi-criteria decision analysis (MCDA) can be used to identify and compare different policy options by assessing their effects, performance, impacts, and trade-offs. Furthermore, it provides a systematic approach for supporting complex decisions according to pre-determined criteria and objectives [65].

So, MCDA is a useful tool that attempts to capture the informal, subjective and often the incalculable sense of preference of a user or decision maker. It is most applicable to solving problems that are characterized as a choice among alternatives. The alternatives in this thesis refer to the different cross-platform frameworks.

There are many multi-criteria decision analysis methods, but the two that will be used in this dissertation are the Analytical Hierarchy Process (AHP) and the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS). AHP is a method of measurement through pair-wise comparisons and relies on the judgments of experts to derive priority scales [65] [66] [67] [68]. On the other hand, the core concept of TOPSIS is that the chosen alternative should have the smallest geometrical distance from the PIS and the largest geometrical distance from NIS [69] [70] [71]. Studies have shown that these two methods are the most widely

used multiple criteria decision-making tools in the last 21 years [67]. These two methods are used by decision makers and researchers, because they are simple, easy to understand, efficient computation, and has the ability to measure the relative performance of the alternatives decision [69] [70]. However, before analyzing the two methods, it is necessary to define two basic steps which will be the same for both methods. This means that the DM options should be as consistent as possible between various methods. These steps are presented below:

- Assume a Decision Maker (DM) to set the proper conditions wherever it needs
- Change the scale of language features

4.3.1 Decision Maker

The first step is to assume a Decision Maker (DM) who will be the person responsible for taking important decisions (importance of attributes, optimal value of each attribute, assumptions, determination of thresholds, etc.). To begin with, the attributes should be classified based on the importance that the DM deems appropriate.

<i>Ranking</i>	<i>Attributes</i>
1°	Code Reuse
2°	Mobile OS
3°	Price
4°	Playground
5°	Community
6°	Programming Language

Table 24: DM's attributes ranking

Next, the optimal value for each attribute must be determined.

Attributes Alternatives	Programming Language (Best value: <u>most used</u>)	Code Reuse (Best value: <u>max %</u>)	Mobile OS (Best value: <u>most OS</u>)	Price (Best value: <u>free</u>)	Community (Best value: <u>max github stargazers</u>)	Playground (Best value: <u>yes</u>)
Xamarin	C#	60-95%	Android iOS Windows Phone	Commercial: free Business: 999\$/year Enterprise: 1899\$/year	5.4K	No
Apache Cordova	HTML,CSS, JavaScript	50-80%	Android iOS Windows Phone	Free	1K	No

Internet technologies for the development of compatible applications on different computing platforms

Ionic	HTML,CSS, JavaScript, Angular	98%	Android iOS	Commercial: free Pro: 539\$/year Enterprise: 2999\$/year	45.3K	No
React Native	JavaScript	90%	Android iOS	Free	98.5K	Yes
Flutter	Dart	50-90%	Android iOS	Free	130K	Yes
Framework7	HTML,CSS, JavaScript	90%	Android iOS	Free	16.4K	No
NativeScript	JavaScript, TypeScript	90%	Android iOS	Free	20.5K	Yes

Table 25: DM's optimal value of attributes

4.3.2 Scale of Language Features

It is observed that not all the attributes have a numerical unit, such as the values for whether the frameworks has playground are presented with *Yes* or *No*. So in the language features there will be a scale conversion keeping in mind which value is optimal. So the language features are three: Programming Language, Mobile OS, Price and Playground.

For the attribute *Programming Language*, a statistical study will be used that displays the most commonly used programming languages among software developers around the world. This study was conducted from 25 May 2021 to 15 June 2021 and involved 83,052 developers from around the world.

Internet technologies for the development of compatible applications on different computing platforms

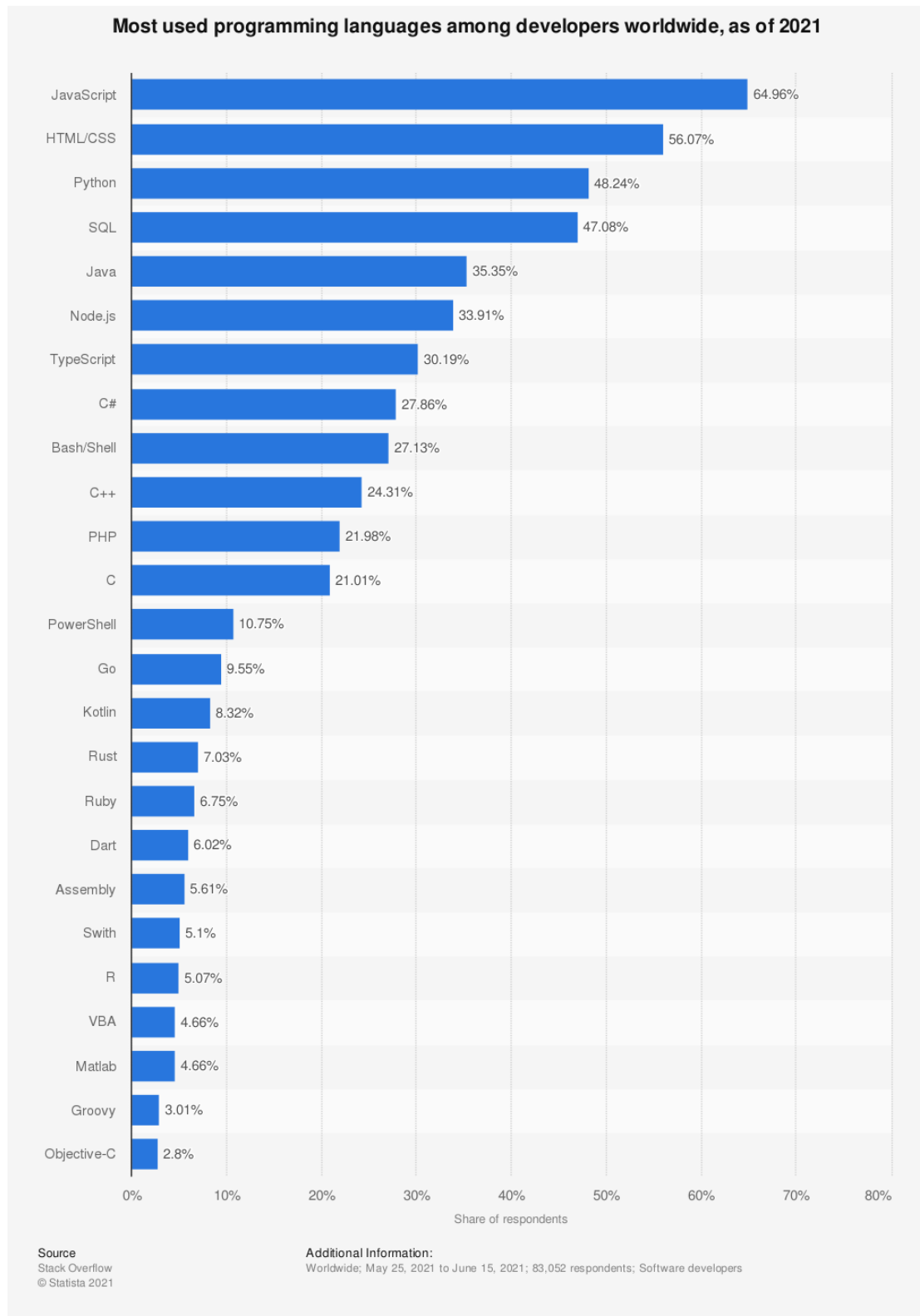


Figure 55: Most used programming languages, Source: <https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/>

Internet technologies for the development of compatible applications on different computing platforms

So, based on the above study, the following table is extracted that shows the percentage of each language. In case a framework includes more than one language, the one with the highest percentage is obtained. The higher the percentage, the more familiar the language of the framework is for the developers.

Attribute \ Alternatives	Programming Language (Best value: <u>most used</u>)	Programming Language (Best value: <u>max</u>)
Xamarin	C#	27,86
Apache Cordova	HTML,CSS, JavaScript	64,96
Ionic	HTML,CSS, JavaScript, Angular	64,96
React Native	JavaScript	64,96
Flutter	Dart	6,02
Framework7	HTML,CSS, JavaScript	64,96
NativeScript	JavaScript, TypeScript	64,96

Table 26: Programming Language conversion

The next attribute (Mobile OS) shows how many mobile operating systems can be used with the same code. The table below displays the number of mobile OS of each cross-platform framework.

Attribute \ Alternatives	Mobile OS (Best value: <u>most OS</u>)	Mobile OS (Best value: <u>max</u>)
Xamarin	Android iOS Windows Phone	3
Apache Cordova	Android iOS Windows Phone	3
Ionic	Android iOS	2
React Native	Android iOS	2
Flutter	Adroid iOS	2
Framework7	Android iOS	2
NativeScript	Android iOS	2

Internet technologies for the development of compatible applications on different computing platforms

Table 27: Mobile OS conversion

The usage price of each framework differs. It is advisable to use two values. So, with the number 0 will characterize the framework which is completely free and with the number 1 the framework that has a price in some of its professional uses.

Attribute \ Alternatives	Price (Best value: <u>free</u>)	Price (Best value: <u>min</u>)
Xamarin	Commercial: free Business: 999\$/year Enterprise: 1899\$/year	1
Apache Cordova	Free	0
Ionic	Commercial: free Pro: 539\$/year Enterprise: 2999\$/year	1
React Native	Free	0
Flutter	Free	0
Framework7	Free	0
NativeScript	Free	0

Table 28: Price conversion

The *Playground* attribute will be converted in two values too. With the number 1 will characterize the framework that has an online playground and with the number 0 the framework that has not.

Attribute \ Alternatives	Playground (Best value: <u>yes</u>)	Playground (Best value: <u>max</u>)
Xamarin	No	0
Apache Cordova	No	0
Ionic	No	0
React Native	Yes	1
Flutter	Yes	1
Framework7	No	0
NativeScript	Yes	1

Table 29: Playground conversion

An assumption will be made for the *Code Reuse* attribute. Some frameworks have a variation in the reuse code and therefore the assumption that will be made is to

take the upper limit as a value as it is interesting to drawing conclusions about the best possible result that each cross-platform framework offers.

Attribute Alternatives	Code Reuse (Best value: <u>max</u> %)	Code Reuse (Best value: <u>max</u>)
Xamarin	60-95%	95
Apache Cordova	50-80%	80
Ionic	98%	98
React Native	90%	90
Flutter	50-90%	90
Framework7	90%	90
NativeScript	90%	90

Table 30: Code Reuse conversion

The following table shows the six transformed attributes according to the scale change of the language features and the importance classification based on the DM.

Attributes Alternatives	Code Reuse (Best value: <u>max</u>)	Mobile OS (Best value: <u>max</u>)	Price (Best value: <u>min</u>)	Playground (Best value: <u>max</u>)	Community (Best value: <u>max</u>)	Programming Language (Best value: <u>max</u>)
Xamarin	95	3	1	0	5.400	27,86
Apache Cordova	80	3	0	0	1.000	64,96
Ionic	98	2	1	0	45.300	64,96
React Native	90	2	0	1	98.500	64,96
Flutter	90	2	0	1	130.000	6,02
Framework7	90	2	0	0	16.400	64,96
NativeScript	90	2	0	1	20.500	64,96

Table 31: Transformed numeric matrix

4.3.3 Analytic Hierarchy Process (AHP)

The Analytic Hierarchy Process, normally called AHP, is a powerful yet simple method for making decisions. It aims to quantify the relative priority of the given set according to the appropriate value scale. The decision is usually based on the perception of the individual (DM) who is supposed to make the final decision and to assess priorities, emphasizing the importance of consistency and correlation of the

alternatives which has been compared in the whole decision-making process [67].

AHP involves five main steps [66]: Step 1: Decompose the problem into a hierarchical structure tree; Step 2: Employ pair-wise comparisons. A pair-wise comparison is the process of comparing the relative importance of two elements (objectives) with respect to another element (the goal). Pair-wise comparisons are carried out to establish priorities. Decision elements at each hierarchy level are compared pair-wisely and then, the reciprocal matrix is completed; Step 3: Determine the logical consistency and if consistency is $>0,1$ revise the pair-wise classifications until the consistency index is below 0,1. When implementing AHP, the input data may be inconsistent and this may cause some adverse effects on the decision making process.; Step 4: Estimate the relative weights of alternatives by performing step 3 to all the alternatives in relation with each attribute; Step 5: Determine the priority of alternatives. In order to do that, it is necessary to multiply the weights of attributes in step 2 with the relative weights of alternatives in respect to each attribute in step 4 [68]. Finally, the weighted sum is calculated by summing each line of the normalized matrix. The framework with the highest weighted sum has the best score.

Step 1

Developing a hierarchical structure with the main goal at the first level, the attributes at the second level and the alternatives at the third level . Each attribute is associated with all seven alternatives.

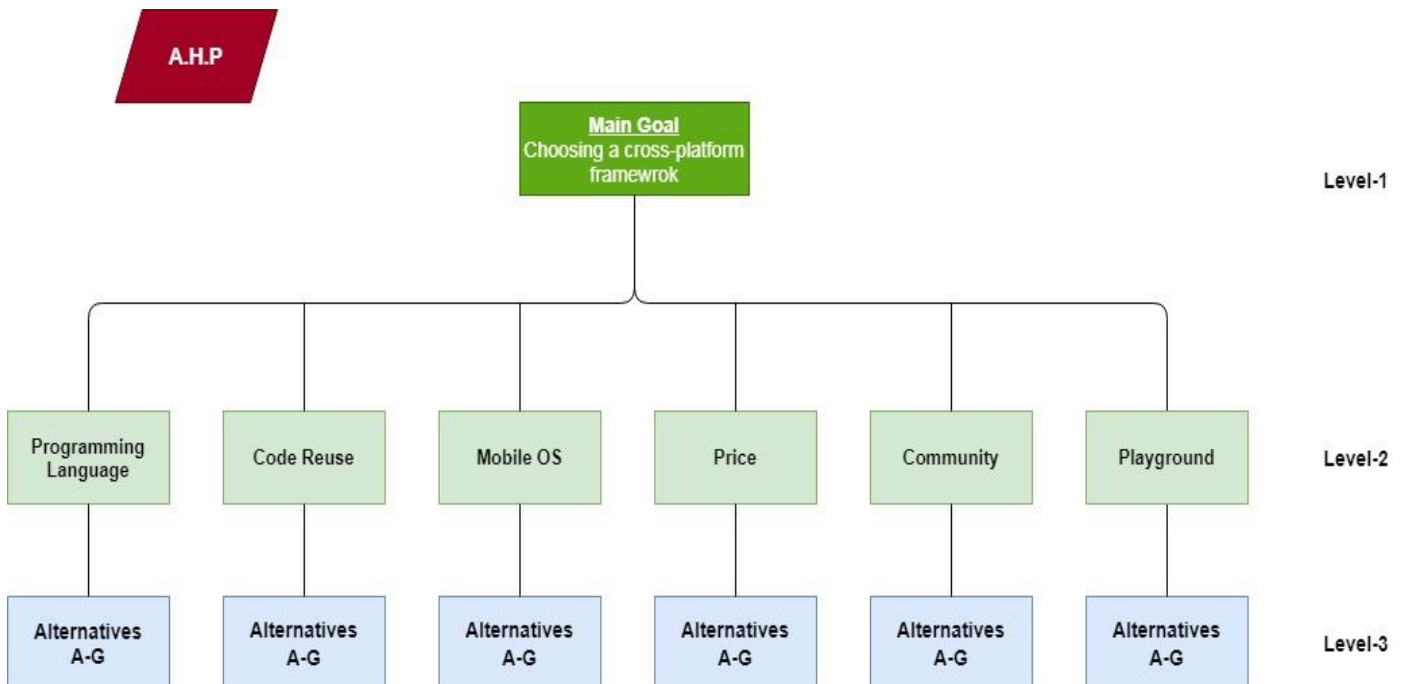


Figure 56: AHP, hierarchical structure tree

Internet technologies for the development of compatible applications on different computing platforms

Step 2

Creating a pair-wise comparison matrix. This pair-wise matrix gives the relative importance of different attributes with respect to the goal. The following scale will also be used to find the relative importance.

<i>Definition</i>	<i>Index</i>	<i>Definition</i>	<i>Index</i>
Equally important	1	Equally important	1
Equally to slightly more important	2	Equally to slightly less important	1/2
Slightly more important	3	Slightly less important	1/3
Slightly to much more important	4	Slightly to way less important	1/4
			1/5
Much more important	5	Way less important	1/5
Much to far more important	6	Way to far less important	1/6
			1/7
Far more important	7	Far less important	1/7
Far more important to extremely more important	8	Far less important to extremely less important	1/8
			1/9
Extremely more important	9	Extremely less important	1/9

Figure 57: AHP, scale of relative importance, Source: [65]

The pair comparison matrix is 6x6 as the attributes are. DM asks each time how important one attribute is in relation to another. For example, it asks the question "how important is the price in relation to the playground?" and DM decide that it is far more important.

	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
Code Reuse	1	2	4	6	7	7
Mobile OS	1/2	1	3	5	6	6
Price	1/4	1/3	1	4	6	6
Playground	1/6	1/5	1/4	1	3	4
Community	1/7	1/6	1/6	1/3	1	2
Programming Language	1/7	1/6	1/6	1/4	1/2	1

Table 32: AHP, attribute's pair-wise matrix

The fractional value is converted to decimal value and the sum of each value is calculated in the end of each column.

Internet technologies for the development of compatible applications on different computing platforms

	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
Code Reuse	1	2	4	6	7	7
Mobile OS	0,50	1	3	5	6	6
Price	0,25	0,33	1	4	6	6
Playground	0,17	0,20	0,25	1	3	4
Community	0,14	0,17	0,17	0,33	1	2
Programming Language	0,14	0,17	0,17	0,25	0,5	1
Sum	2,2	3,87	8,59	16,58	23,5	26

Table 33: AHP, sum of attribute's pair-wise matrix

Then, normalized pair-wise matrix is calculated. All the elements of each column is divided by the sum of the column.

	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
Code Reuse	$1/2,2$	$2/3,87$	$4/8,59$	$6/16,58$	$7/23,5$	$7/26$
Mobile OS	$0,50/2,2$	$1/3,87$	$3/8,59$	$5/16,58$	$6/23,5$	$6/26$
Price	$0,25/2,2$	$0,33/3,87$	$1/8,59$	$4/16,58$	$6/23,5$	$6/26$
Playground	$0,17/2,2$	$0,20/3,87$	$0,25/8,59$	$1/16,58$	$3/23,5$	$4/26$
Community	$0,14/2,2$	$0,17/3,87$	$0,17/8,59$	$0,33/16,58$	$1/23,5$	$2/26$
Programming Language	$0,14/2,2$	$0,17/3,87$	$0,17/8,59$	$0,25/16,58$	$0,5/23,5$	$1/26$

Table 34: AHP, normalized each column

	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
Code Reuse	0,455	0,517	0,466	0,362	0,298	0,269
Mobile OS	0,227	0,258	0,349	0,302	0,255	0,231
Price	0,114	0,085	0,116	0,241	0,255	0,231
Playground	0,077	0,052	0,029	0,060	0,128	0,154
Community	0,064	0,044	0,020	0,020	0,043	0,077
Programming Language	0,064	0,044	0,020	0,015	0,021	0,038

Internet technologies for the development of compatible applications on different computing platforms

Table 35: AHP, normalized attribute's pair-wise matrix

To find the weights of each attribute, it is necessary to find the average of all the attributes in each row.

$$W_{\text{Code Reuse}} = \frac{0,455+0,517+0,466+0,362+0,298+0,269}{6} \approx \mathbf{0,395}$$

$$W_{\text{Mobile OS}} = \frac{0,227+0,258+0,349+0,302+0,255+0,231}{6} \approx \mathbf{0,270}$$

$$W_{\text{Price}} = \frac{0,114+0,085+0,116+0,241+0,255+0,231}{6} \approx \mathbf{0,174}$$

$$W_{\text{Playground}} = \frac{0,077+0,052+0,029+0,060+0,128+0,154}{6} \approx \mathbf{0,083}$$

$$W_{\text{Community}} = \frac{0,064+0,044+0,020+0,020+0,043+0,077}{6} \approx \mathbf{0,045}$$

$$W_{\text{Programming Language}} = \frac{0,064+0,044+0,020+0,015+0,021+0,038}{6} \approx \mathbf{0,034}$$

So, the matrix with the attribute weights is shown below.

	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language	Weight
Code Reuse	0,455	0,517	0,466	0,362	0,298	0,269	0,395
Mobile OS	0,227	0,258	0,349	0,302	0,255	0,231	0,270
Price	0,114	0,085	0,116	0,241	0,255	0,231	0,174
Playground	0,077	0,052	0,029	0,060	0,128	0,154	0,083
Community	0,064	0,044	0,020	0,020	0,043	0,077	0,045
Programming Language	0,064	0,044	0,020	0,015	0,021	0,038	0,034

Table 36: AHP, attribute's weights matrix

Step 3

Calculate the consistency. That is to check whether the calculated value are correct or not. For this purpose, the matrix in Table 33, which is not normalized, is multiplied by the weight of each attribute.

Internet technologies for the development of compatible applications on different computing platforms

	Code Reuse $W_1=0,395$	Mobile OS $W_2=0,270$	Price $W_3=0,174$	Playground $W_4=0,083$	Community $W_5=0,045$	Programming Language $W_6=0,034$
Code Reuse	$1*0,395$	$2*0,270$	$4*0,174$	$6*0,083$	$7*0,045$	$7*0,034$
Mobile OS	$0,50*0,395$	$1*0,270$	$3*0,174$	$5*0,083$	$6*0,045$	$6*0,034$
Price	$0,25*0,395$	$0,33*0,270$	$1*0,174$	$4*0,083$	$6*0,045$	$6*0,034$
Playground	$0,17*0,395$	$0,20*0,270$	$0,25*0,174$	$1*0,083$	$3*0,045$	$4*0,034$
Community	$0,14*0,395$	$0,17*0,270$	$0,17*0,174$	$0,33*0,083$	$1*0,045$	$2*0,034$
Programming Language	$0,14*0,395$	$0,17*0,270$	$0,17*0,174$	$0,25*0,083$	$0,50*0,045$	$1*0,034$

Table 37: AHP, multiplication of weights of each column

The weighted sum value is calculated by taking the sum of each value in the row.

	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language	Weighted Sum
Code Reuse	0,395	0,540	0,696	0,498	0,315	0,238	2,685
Mobile OS	0,198	0,270	0,522	0,415	0,270	0,204	1,88
Price	0,099	0,089	0,174	0,332	0,270	0,204	1,168
Playground	0,067	0,054	0,044	0,083	0,135	0,136	0,519
Community	0,055	0,046	0,030	0,027	0,045	0,068	0,271
Programming Language	0,055	0,046	0,030	0,021	0,023	0,034	0,209

Table 38: AHP, weighted sum matrix

Once the total weight sum for each row is found, divided by the weight of each attribute to find the ratio. Below is the final matrix with all the weights and the ratio.

	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language	Weighted Sum	Weight	Ratio
Code Reuse	0,395	0,540	0,696	0,498	0,315	0,238	2,682	0,447	6
Mobile OS	0,198	0,270	0,522	0,415	0,270	0,204	1,879	0,313	6
Price	0,099	0,089	0,174	0,332	0,270	0,204	1,168	0,195	5,99
Playground	0,067	0,054	0,044	0,083	0,135	0,136	0,519	0,086	6,03
Community	0,055	0,046	0,030	0,027	0,045	0,068	0,271	0,045	6,02
Programming Language	0,055	0,046	0,030	0,021	0,023	0,034	0,209	0,034	6,15

Table 39: AHP, ratio matrix

Lambda max (λ_{max}) is calculated by taking the average of the ratio values of Table 39.

$$\lambda_{max} = \frac{6+6+5,99+6,03+6,02+6,15}{6} = \mathbf{6,03}$$

Next, the Consistency Index (C.I.) is calculated.

$$C.I. = \frac{\lambda_{max} - n}{n-1} = \frac{6,03-6}{6-1} = \mathbf{0,0063}$$
 where n is the number of compared attributes

After that, the Consistency Ratio (C.R.) is calculated. In order to find the C.R., it is necessary to use the Random Index (RI), which is the consistency index of randomly generated pair-wise matrix.

Order	1	2	3	4	5	6	7	8	9	10
Random Index	0	0	0,52	0,89	1,11	1,25	1,35	1,4	1,45	1,49

Figure 58: AHP, Random Index, Source: [67]

Because the attributes are six, the Random Index from the table is 1,25. So, the C.R. is given from the formula:

$$C.R. = \frac{\text{Consistency Index}}{\text{Random Index}} = \frac{0,0063}{1,25} = \mathbf{0,0051}$$

Since the value of C.R. **<0,1** then, the values that have been given, are considered sufficiently consistent and the analysis can be continued.

Step 4

Continuing the analysis with the AHP method, the comparison matrix of alternatives in relation to each attribute will be implemented. As the attributes are six, six 7x7 matrices will be made. Starting with the first feature (Code Reuse) followed exactly the same steps that were done to give the weights of the attributes.

DM sets the values of alternatives based on Figure 57.

Code Reuse	Xamarin	Apache Cordova	Ionic	React Native	Flutter	Framework7	NativeScript
Xamarin	1	4	1/2	3	3	3	3
Apache Cordova	1/4	1	1/5	1/3	1/3	1/3	1/3
Ionic	2	5	1	3	3	3	3
React Native	1/3	3	1/3	1	1	1	1
Flutter	1/3	3	1/3	1	1	1	1

Internet technologies for the development of compatible applications on different
computing platforms

Framework7	1/3	3	1/3	1	1	1	1
NativeScript	1/3	3	1/3	1	1	1	1
Sum	4,57	22	3,02	10,33	10,33	10,33	10,33

Table 40: AHP, Code Reuse pair-wise comparison matrix

Next, divide each value of each column by the sum of the column and then add each value of each row and divide it by the number of the alternatives, which is seven. This way, the weights of the alternatives in relation to the attribute “Code Reuse” were found. Calculating the weight sum and the weight, the ratio of consistency can be found.

Code Reuse	Xamarin	Apache Cordova	Ionic	React Native	Flutter	Framework7	NativeScript	Weight Sum	Weight
Xamarin	0,22	0,18	0,17	0,29	0,29	0,29	0,29	1,73	0,247
Apache Cordova	0,05	0,05	0,07	0,03	0,03	0,03	0,03	0,29	0,041
Ionic	0,44	0,23	0,33	0,29	0,29	0,29	0,29	2,16	0,31
React Native	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10
Flutter	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10
Framework7	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10
NativeScript	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10

Table 41: AHP, weights of alternatives in respect to Code Reuse attribute

Code Reuse	Xamarin	Apache Cordova	Ionic	React Native	Flutter	Framework7	NativeScript	Weight Sum	Weight	Ratio
Xamarin	0,22	0,18	0,17	0,29	0,29	0,29	0,29	1,73	0,247	7,004
Apache Cordova	0,05	0,05	0,07	0,03	0,03	0,03	0,03	0,29	0,041	7,07
Ionic	0,44	0,23	0,33	0,29	0,29	0,29	0,29	2,16	0,31	6,97
React Native	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10	7,1
Flutter	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10	7,1
Framework7	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10	7,1
NativeScript	0,07	0,14	0,11	0,097	0,097	0,097	0,097	0,71	0,10	7,1

Table 42: AHP, Code Reuse consistency matrix

Finally, the λ_{max} and the consistency index are calculated. The Random Index is 1,35 (Figure 58) as there are seven alternatives. In order to find λ_{max} , C.I. and C.R. , the same formulas used in step 3 will be used. The matrix below summarizes all the

Internet technologies for the development of compatible applications on different computing platforms

values of the three formulas. Still, because **C.R. <0,1** the values that have been given, are considered sufficiently consistent and the analysis can be continued.

Code Reuse	
λ_{\max}	7,06
C.I.	0,011
RI	1,35
C.R.	0,008
Consistent?	Yes

Table 43: AHP, Code Reuse formula matrix

The exact same steps are done for the other five attributes.

Step 5

After finding the weights of all the alternatives in relation to the respective attribute, multiplied by the weights found in step 2 in Table 36 of the AHP method.

Attributes Alternatives	Code Reuse $W_1=0,395$	Mobile OS $W_2=0,270$	Price $W_3=0,174$	Playground $W_4=0,083$	Community $W_5=0,045$	Programming Language $W_6=0,034$
Xamarin	0,247* 0,395	0,27*0,270	0,021* 0,174	0,034* 0,083	0,031*0,045	0,065*0,034
Apache Cordova	0,041* 0,395	0,27*0,270	0,196* 0,174	0,034* 0,083	0,023*0,045	0,183*0,034
Ionic	0,31* 0,395	0,09*0,270	0,021* 0,174	0,034* 0,083	0,111*0,045	0,183*0,034
React Native	0,10* 0,395	0,09*0,270	0,196* 0,174	0,307* 0,083	0,764*0,045	0,183*0,034
Flutter	0,10* 0,395	0,09*0,270	0,196* 0,174	0,307* 0,083	0,443*0,045	0,018*0,034
Framework7	0,10* 0,395	0,09*0,270	0,196* 0,174	0,034* 0,083	0,063*0,045	0,183*0,034
NativeScript	0,10* 0,395	0,09*0,270	0,196* 0,174	0,307* 0,083	0,084*0,045	0,183*0,034

Table 44: AHP, normalized alternatives in respect to attributes

The weighted sum value is calculated by taking the sum of each value in the row.

Attributes Alternatives	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language	Weighted Sum
Xamarin	0,098	0,073	0,0037	0,0028	0,0014	0,0022	0,1805
Apache Cordova	0,016	0,073	0,034	0,0028	0,001	0,0062	0,1333

Internet technologies for the development of compatible applications on different computing platforms

Ionic	0,123	0,024	0,0037	0,0028	0,005	0,0062	0,1650
React Native	0,040	0,024	0,034	0,026	0,034	0,0062	0,1645
Flutter	0,040	0,024	0,034	0,026	0,020	0,0006	0,1445
Framework7	0,040	0,024	0,034	0,0028	0,0028	0,0062	0,1098
NativeScript	0,040	0,024	0,034	0,026	0,0038	0,0062	0,1339

Table 45: AHP, weighted sum matrix

The weighted sum will yield the ranking of the alternatives.

Attributes Alternatives	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language	Weighted Sum	Ranking
Xamarin	0,098	0,073	0,0037	0,0028	0,0014	0,0022	0,1805	1st
Apache Cordova	0,016	0,073	0,034	0,0028	0,001	0,0062	0,1333	6th
Ionic	0,123	0,024	0,0037	0,0028	0,005	0,0062	0,1650	2nd
React Native	0,040	0,024	0,034	0,026	0,034	0,0062	0,1645	3rd
Flutter	0,040	0,024	0,034	0,026	0,020	0,0006	0,1445	4th
Framework7	0,040	0,024	0,034	0,0028	0,0028	0,0062	0,1098	7th
NativeScript	0,040	0,024	0,034	0,026	0,0038	0,0062	0,1339	5th

Table 46: AHP, ranking matrix

4.3.4 Technique for Order Preference by Similarity to Ideal Solution (TOPSIS)

TOPSIS using the principle that the alternatives chosen should have the shortest distance from the positive ideal solution and the farthest from the negative ideal solution from a geometric point using the Euclidean distance to determine the relative proximity of an alternative in the optimal solution [69]. Positive ideal solution (PIS) is defined as the sum of all the best value that can be achieved for each attribute, while the negative ideal solution (NIS) consists of all the worst value achieved for each attribute [70] [71].

TOPSIS involves five main steps: Step 1: Normalize the decision-matrix; Step 2: Calculate the weighted normalized decision matrix; Step 3: Determine the positive ideal and negative ideal solutions; Step 4: Calculate the separation measures; Step 5: Rank the preference order by calculating the relative closeness to the ideal solution [69] [70] [71].

Step 1

Calculation of the normalized decision matrix. To normalize the Table 31, the following formula will be used.

$$r_{ij} = \frac{X_{ij}}{\sqrt{\sum_{i=1}^m X_{ij}^2}}$$

Code Reuse: $\sqrt{95^2 + 80^2 + 98^2 + 90^2 + 90^2 + 90^2 + 90^2} \approx 240$

Mobile OS: $\sqrt{3^2 + 3^2 + 2^2 + 2^2 + 2^2 + 2^2 + 2^2} \approx 6,16$

Price: $\sqrt{1^2 + 0^2 + 1^2 + 0^2 + 0^2 + 0^2 + 0^2} \approx 1,41$

Playground: $\sqrt{0^2 + 0^2 + 0^2 + 1^2 + 1^2 + 0^2 + 1^2} \approx 1,73$

Community: $\sqrt{5400^2 + 1000^2 + 45300^2 + 98500^2 + 130000^2 + 16400^2 + 20500^2} \approx 4.795.178$

Programming Language: $\sqrt{27,86^2 + 64,96^2 + 64,96^2 + 64,96^2 + 6,02^2 + 64,96^2 + 64,96^2} \approx 148$

Thus, the matrix is configured as follows.

Attributes \ Alternatives	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
Xamarin	95/240	3/6,16	1/1,41	0/1,73	5.400/4.795.178	27,86/148
Apache Cordova	80/240	3/6,16	0/1,41	0/1,73	1.000/4.795.178	64,96/148
Ionic	98/240	2/6,16	1/1,41	0/1,73	45.300/4.795.178	64,96/148
React Native	90/240	2/6,16	0/1,41	1/1,73	98.500/4.795.178	64,96/148
Flutter	90/240	2/6,16	0/1,41	1/1,73	130.000/4.795.178	6,02/148
Framework7	90/240	2/6,16	0/1,41	0/1,73	16.400/4.795.178	64,96/148
NativeScript	90/240	2/6,16	0/1,41	1/1,73	20.500/4.795.178	64,96/148

Table 47: TOPSIS, normalized division of each column

Attributes \ Alternatives	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
Xamarin	0,40	0,49	0,71	0	0,0011	0,19
Apache Cordova	0,33	0,49	0	0	0,0002	0,44

Internet technologies for the development of compatible applications on different computing platforms

Ionic	0,41	0,32	0,71	0	0,0094	0,44
React Native	0,375	0,32	0	0,58	0,021	0,44
Flutter	0,375	0,32	0	0,58	0,027	0,041
Framework7	0,375	0,32	0	0	0,0034	0,44
NativeScript	0,375	0,32	0	0,58	0,0043	0,44

Table 48: TOPSIS, normalized matrix

Step 2

Determination of the weighted normalized decision matrix. In this step the DM determines the weights of each attribute. By multiplying every element of each column with the weight of each attribute, the weighted decision matrix is created.

Attributes Alternatives	Code Reuse $W_1=0,3$	Mobile OS $W_2=0,25$	Price $W_3=0,2$	Playground $W_4=0,15$	Community $W_5=0,075$	Programming Language $W_6=0,025$
Xamarin	$0,4*0,3$	$0,49*0,25$	$0,71*0,2$	$0*0,15$	$0,0011*0,075$	$0,19*0,025$
Apache Cordova	$0,33*0,3$	$0,49*0,25$	$0*0,2$	$0*0,15$	$0,0002*0,075$	$0,44*0,025$
Ionic	$0,41*0,3$	$0,32*0,25$	$0,71*0,2$	$0*0,15$	$0,0094*0,075$	$0,44*0,025$
React Native	$0,375*0,3$	$0,32*0,25$	$0*0,2$	$0,58*0,15$	$0,021*0,075$	$0,44*0,025$
Flutter	$0,375*0,3$	$0,32*0,25$	$0*0,2$	$0,58*0,15$	$0,027*0,075$	$0,041*0,025$
Framework7	$0,375*0,3$	$0,32*0,25$	$0*0,2$	$0*0,15$	$0,0034*0,075$	$0,44*0,025$
NativeScript	$0,375*0,3$	$0,32*0,25$	$0*0,2$	$0,58*0,15$	$0,0043*0,075$	$0,44*0,025$

Table 49: TOPSIS, multiply columns with the weights of each attribute

Attributes Alternatives	Code Reuse $W_1=0,3$	Mobile OS $W_2=0,25$	Price $W_3=0,2$	Playground $W_4=0,15$	Community $W_5=0,075$	Programming Language $W_6=0,025$
Xamarin	0,120	0,123	0,142	0	0.000083	0,004
Apache Cordova	0,099	0,123	0	0	0.000015	0,011
Ionic	0,123	0,080	0,142	0	0.00071	0,011
React Native	0,113	0,080	0	0,087	0,00158	0,011
Flutter	0,113	0,080	0	0,087	0,00203	0,001
Framework7	0,113	0,080	0	0	0.00026	0,011
NativeScript	0,113	0,080	0	0,087	0.00032	0,011

Table 50: TOPSIS, weighted normalized matrix

Step 3

Determining the ideal solution matrix of positive and negative ideal solution by using the below formulas.

$$A^+ = \{(\max v_{ij} | j \in J), (\min v_{ij} | j \in J'), i = 1,2,3, \dots, m\}$$

$$= \{V1^+, V2^+, V3^+, \dots, Vn^+\}$$

$$A^- = \{(\min v_{ij} | j \in J), (\max v_{ij} | j \in J'), i = 1,2,3, \dots, m\}$$

$$= \{V1^-, V2^-, V3^-, \dots, Vn^-\},$$

where J' is associated with the non-beneficial attributes and J is associated with beneficial attributes. Non-beneficial attribute is only the Price and so, A+ is the minimum value and A- is the maximum value.

Attributes Alternatives	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
Xamarin	0,12	0,123	0,142	0	0,000083	0,004
Apache Cordova	0,099	0,123	0	0	0,000015	0,011
Ionic	0,123	0,08	0,142	0	0,00071	0,011
React Native	0,113	0,08	0	0,087	0,00158	0,011
Flutter	0,113	0,08	0	0,087	0,00203	0,001
Framework7	0,113	0,08	0	0	0,00026	0,011
NativeScript	0,113	0,08	0	0,087	0,00032	0,011
A+	0,123	0,123	0,142	0	0,00203	0,011
A-	0,099	0,08	0	0,087	0,000015	0,001

Table 51: TOPSIS, PIS & NIS matrix

Step 4

Calculating separation distance from the positive ideal solution and negative ideal solution of each attribute. The two formulas that are going to use are shown below.

$$S_i^+ = \sqrt{\sum_{j=1}^m (V_{ij} - V_j^+)^2}, \text{ where } i=1,2,3,\dots,m$$

$$S_i^- = \sqrt{\sum_{j=1}^m (V_{ij} - V_j^-)^2}, \text{ where } i=1,2,3,\dots,m$$

$$S_i^+$$

Internet technologies for the development of compatible applications on different computing platforms

Xamarin: $\sqrt{(0,12 - 0,123)^2 + (0,123 - 0,123)^2 + (0,142 - 0,142)^2 + (0 - 0)^2 + (0,000083 - 0,00203)^2 + (0,004 - 0,011)^2} \approx \mathbf{0,0078}$

Apache Cordova: $(0,099 - 0,123)^2 + (0,123 - 0,123)^2 + (0 - 0,142)^2 + (0 - 0)^2 + (0,000015 - 0,00203)^2 + (0,011 - 0,011)^2 \approx \mathbf{0,15}$

Ionic: $(0,123 - 0,123)^2 + (0,08 - 0,123)^2 + (0,142 - 0,142)^2 + (0 - 0)^2 + (0,00071 - 0,00203)^2 + (0,011 - 0,011)^2 \approx \mathbf{0,042}$

React Native: $(0,113 - 0,123)^2 + (0,08 - 0,123)^2 + (0 - 0,142)^2 + (0,087 - 0)^2 + (0,00158 - 0,00203)^2 + (0,011 - 0,011)^2 \approx \mathbf{0,173}$

Flutter: $(0,113 - 0,123)^2 + (0,08 - 0,123)^2 + (0 - 0,142)^2 + (0,087 - 0)^2 + (0,00203 - 0,00203)^2 + (0,001 - 0,011)^2 \approx \mathbf{0,0298}$

Framework7: $(0,113 - 0,123)^2 + (0,08 - 0,123)^2 + (0 - 0,142)^2 + (0 - 0)^2 + (0,00026 - 0,00203)^2 + (0,011 - 0,011)^2 \approx \mathbf{0,023}$

NativeScript: $(0,113 - 0,123)^2 + (0,08 - 0,123)^2 + (0 - 0,142)^2 + (0,087 - 0)^2 + (0,00032 - 0,00203)^2 + (0,011 - 0,011)^2 \approx \mathbf{0,0297}$

S_i^-

Xamarin: $\sqrt{(0,12 - 0,099)^2 + (0,123 - 0,08)^2 + (0,142 - 0)^2 + (0 - 0,087)^2 + (0,000083 - 0,000015)^2 + (0,004 - 0,001)^2} \approx \mathbf{0,197}$

Apache Cordova: $(0,099 - 0,099)^2 + (0,123 - 0,08)^2 + (0 - 0)^2 + (0 - 0,087)^2 + (0,000015 - 0,000015)^2 + (0,011 - 0,001)^2 \approx \mathbf{0,097}$

Ionic: $(0,123 - 0,099)^2 + (0,08 - 0,08)^2 + (0,142 - 0)^2 + (0 - 0,087)^2 + (0,00071 - 0,000015)^2 + (0,011 - 0,001)^2 \approx \mathbf{0,17}$

React Native: $(0,113 - 0,099)^2 + (0,08 - 0,08)^2 + (0 - 0)^2 + (0,087 - 0,087)^2 + (0,00158 - 0,000015)^2 + (0,011 - 0,001)^2 \approx \mathbf{0,0173}$

Flutter: $(0,113 - 0,099)^2 + (0,08 - 0,08)^2 + (0 - 0)^2 + (0,087 - 0,087)^2 + (0,00203 - 0,000015)^2 + (0,001 - 0,001)^2 \approx \mathbf{0,014}$

Framework7: $(0,113 - 0,099)^2 + (0,08 - 0,08)^2 + (0 - 0)^2 + (0 - 0,087)^2 + (0,00026 - 0,000015)^2 + (0,011 - 0,001)^2 \approx \mathbf{0,0079}$

NativeScript: $(0,113 - 0,099)^2 + (0,08 - 0,08)^2 + (0 - 0)^2 + (0,087 - 0,087)^2 + (0,00032 - 0,000015)^2 + (0,011 - 0,001)^2 \approx \mathbf{0,0003}$

So, the Table 51 after the completion of S_i^+ and S_i^- is presented below.

Internet technologies for the development of compatible applications on different computing platforms

Attributes Alternatives	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language	S_i^+	S_i^-
Xamarin	0,12	0,123	0,142	0	0,000083	0,004	0,0078	0,197
Apache Cordova	0,099	0,123	0	0	0,000015	0,011	0,15	0,097
Ionic	0,123	0,08	0,142	0	0,00071	0,011	0,042	0,17
React Native	0,113	0,08	0	0,087	0,00158	0,011	0,173	0,0173
Flutter	0,113	0,08	0	0,087	0,00203	0,001	0,0298	0,014
Framework7	0,113	0,08	0	0	0,00026	0,011	0,023	0,0079
NativeScript	0,113	0,08	0	0,087	0,00032	0,011	0,0297	0,003
A+	0,123	0,123	0,142	0	0,00203	0,011		
A-	0,099	0,08	0	0,087	0,000015	0,001		

Table 52: TOPSIS, separation distance matrix

Step 5

Calculating the relative closeness of the ideal solution. The next formula that will give the ranking of alternatives is the following.

$$C_i = \frac{S_i^-}{(S_i^+ + S_i^-)}, 0 \leq C_i \leq 1$$

Attributes Alternatives	S_i^+	S_i^-	$S_i^+ + S_i^-$	C_i	Ranking
Xamarin	0,0078	0,197	0,204	0,97	1 st
Apache Cordova	0,15	0,097	0,247	0,39	3 rd
Ionic	0,042	0,17	0,212	0,8	2 nd
React Native	0,173	0,0173	0,190	0,091	6 th
Flutter	0,0298	0,014	0,044	0,32	4 th
Framework7	0,023	0,0079	0,031	0,25	5 th
NativeScript	0,0297	0,003	0,033	0,0909	7 th

Table 53: TOPSIS, ranking matrix

4.4 Results

At the MCDA that was conducted, the methodology for evaluating and selecting the most appropriate cross-platform framework was presented by implementing the AHP and TOPSIS. The weights used by DM are shown below.

Attributes \ Methods	Code Reuse	Mobile OS	Price	Playground	Community	Programming Language
AHP	0,395	0,27	0,174	0,083	0,045	0,034
TOPSIS	0,3	0,25	0,2	0,15	0,075	0,025

Table 54: Total weights of attributes

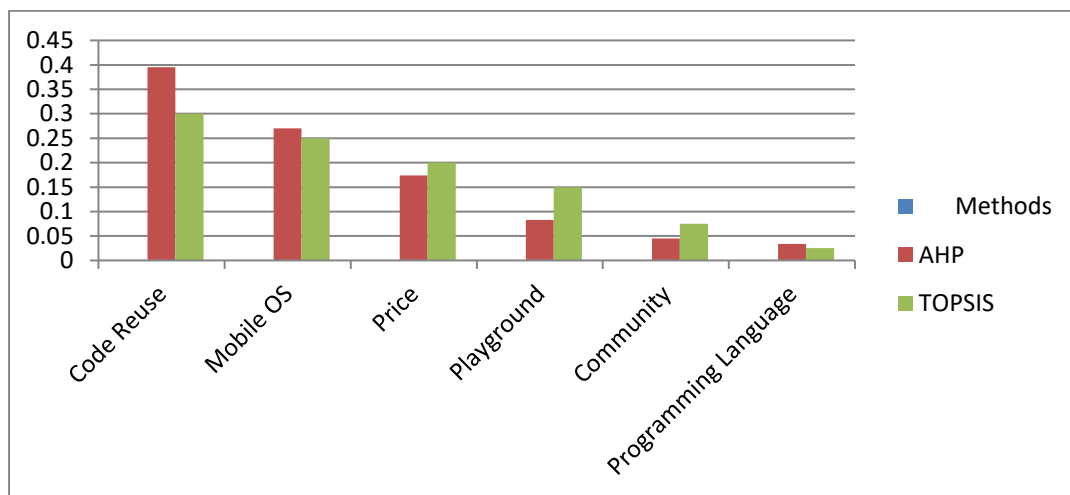


Figure 59: Diagram of weights

It is observed that the weights of each attribute have approximate values. This was to be expected as the DM weight choices in each attribute were consistent. The overall evaluation of the two methods is presented in the Table 55.

Methods \ Alternatives	AHP	TOPSIS
Xamarin	1st	1st
Apache Cordova	6th	3rd
Ionic	2nd	2nd
React Native	3rd	6th
Flutter	4th	4th
Framework7	7th	5th
NativeScript	5th	7th

Table 55: Total ranking

Internet technologies for the development of compatible applications on different computing platforms

The result from the analysis show that out of the seven competing frameworks, which satisfy six attributes, Xamarin is the best. It is also noted that Ionic is the second best choice of both methods.

Chapter 5: Conclusion

This thesis aimed to compare seven cross-platform mobile development frameworks, namely Xamarin, Apache Cordova, Ionic, React Native, Flutter, Framework7 and NativeScript. Initially, a literature review of the four mobile development approaches, native, web, hybrid, and cross-platform, was conducted. It turned out that cross-platform mobile development has many advantages over the other three development processes, as the use of the multi-platform development frameworks helps to bring application to many target platforms at once, make the process of maintenance simpler but also allows using any platform specific functionality if needed. So, cross-platform application development is certainly a worthy alternative to traditional native development.

Then, a thorough study was done for each framework separately. More specifically, their architecture, their advantages and disadvantages, which helped in their further analysis in the fourth chapter, the requirements for the setup environment as well as a code example, were studied. In four of the seven frameworks, the same application was made. The application is very simple and consists of two pages where with a button on the home page you go to the second. This was done to show which of them is implemented more easily and with the fewest lines of code. It has been observed that in Xamarin these two conditions are met. In addition, three of the seven frameworks, React Native, Flutter, and NativeScript have online coding playground where everyone can use quickly and easily. It should be noted that the playground of flutter, Flutlab is better than the other two because the code is stored in an apk file and does not need to install an intermediate app on the mobile to run the application.

The most important chapter is the fourth as through research original results are extracted via two methods of multi-criteria decision analysis (MCDA). At first, the similarities and differences of the seven cross-platform frameworks were analyzed and a table of differences was created which supported the whole multi-criteria decision analysis study. The comparison table contained six different attributes: code reuse, mobile OS, price, playground, community and programming language. After this table was created the analysis of MCDA was done.

MCDA is a general framework for supporting complex decision-making situations with multiple and often conflicting objectives that stakeholders groups and/or decision-makers value differently. In this study, a decision maker is supposed to judge the importance of attributes as well as others decisions, such as setting the weights of attributes. Furthermore, some attributes that contained language descriptions were converted to numeric values. Once the transformed table was created, AHP and TOPSIS multi-criteria decision methods were applied. It is a fact that there are many methods; however these two are the most commonly used in

Internet technologies for the development of compatible applications on different computing platforms

the last 21 years (Broniewicz and Ogrodnik [2021](#)). The results of the two methods showed a similarity in the first and the second alternative (framework) with the best choice being Xamarin followed by Ionic.

Summarizing, the objective of this thesis was, through the answering of the research questions posed in the introduction, to present the main similarities and differences of the seven cross-platform frameworks and to support the best candidate via MCDA. Consequently, the conclusion drawn is that MCDA process must be a continuous and a personalized process for any business with clear goals, which may change over time. What needs to be emphasized is that this MCDA is not a panacea, but a very good example of how a survey should be conducted to select the most suitable candidate.

References

- [1] Dainow, E., (2017), "A Concise History of Computers, Smartphones and the Internet", Ernie Dainow, pp. 44-45.
- [2] Sarwa, M. & Soomro, T.R., (2013), "Impact of Smartphone's on Society", European Journal of Scientific Research, 98 (2), pp. 216-226.
- [3] Sheikh, A.A., Ganai, P.T., Malik, N.A. & Dar, K.A., (2013), "Smartphone: Android Vs IOS", SIJ Transactions on Computer Science Engineering & its Applications (CSEA), 1 (4), pp. 31-38, doi: 10.9756/SIJCSEA/V1I4/0104600401
- [4] Islam, N. & Want, R., (2014), "Smartphones: Past, Present, and Future", IEEE Pervasive Computing, 13 (4), pp.89-92, doi: 10.1109/MPRV.2014.74
- [5] Donner, J., (2006), "The Social and Economic Implications of Mobile Telephony in Rwanda: An Ownership/Access Typology", Knowledge, Technology & Policy, 19 (2), pp. 17-28, doi: 10.1007/s12130-006-1021-7
- [6] Flora, H., Wang, X. & Chande, S., (2014), "An investigation on the Characteristics of Mobile Applications: A Survey Study", I.J. Information Technology & Computer Science, 11 (11), pp. 21-27, doi: 10.5815/ijitcs.2014.11.03
- [7] Martin, S, (2020, April 14), *7 Popular Cross-Platform App Development tools That Will Rule in 2020*, Medium, Retrieved: November 23, 2020, from <https://medium.com/datadriveninvestor/7-popular-cross-platform-app-development-tools-that-will-rule-in-2020-349c80fb51>
- [8] Charland, A. & Leroux, B., (2011), "Mobile Application Development: Web vs. Native ", Communications of the ACM, 54 (5), pp. 49-53, doi: 10.1145/1941487.1941504
- [9] Ahmad, A., Li, K., Feng, C., Asim, S. M., Yousif, A. & Ge, S., (2018), "An Empirical Study of Investigating Mobile Applications Development Challenges", IEEE Access, 6, pp. 17711-17728, doi: 10.1109/ACCESS.2018.2818724
- [10] Pouryousef, S., Rezaiee, M. & Chizari, A., (2018), "Let me Join Two Worlds! Analyzing the Integration of Web and Native Technologies in Hybrid Mobile Apps", 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), pp. 1814-1819, doi: 10.1109/TrustCom/BigDataSE.2018.00274
- [11] Zidoun, Y., Dehbi, R. & Talea, M., (2018), "Multi-Criteria Analysis and Advanced Comparative Study between M-learning Development Approaches", International Journal of Interactive Mobile Technologies (IJIM), 12 (3), pp. 38-51, doi: 10.3991/ijim.v12i3.8083
- [12] Rosler, F., Nitze, A. & Schmietendorf, A., (2014), "Towards a Mobile Application Performance Benchmark", 9th International Conference on Internet and Web Applications and Services (ICIW), pp. 55-59.

- [13] Mehrotra, O., (2019, May 03), *Native vs Cross-Platform: The Best App Development Approach for Startups*, Appventurez, Retrieved: November 16, 2020, from <https://www.appventurez.com/blog/cross-platform-vs-native-app-development/>
- [14] Monus, A.,(2019, March 19), *Understanding native app development – what you need to know in 2019*, RAYGUN, Retrieved: December 06, 2020, from <https://raygun.com/blog/native-app-development/>
- [15] Manchanda, A., (2020, September 08), *Where Do Cross-Platform App Frameworks Stand in 2020*, Net solutions, Retrieved: November 05, 2020, from <https://www.netsolutions.com/insights/cross-platform-app-frameworks-in-2019/>
- [16] Kaoudi, Z. & Quiane-Ruiz, J., (2018), “Cross-Platform Data Processing: Use Cases and Challenges”, IEEE 34th International Conference on Data Engineering (ICDE), pp. 1723-1726, doi: 10.1109/ICDE.2018.00223
- [17] Kesavan, M., (2017, February 06), *Pros and Cons of Cross-Platform App Development*, DZone, Retrieved: November 07, 2020, from <https://dzone.com/articles/pros-and-cons-of-cross-platform-app-development>
- [18] Dalmaso, I., Datta, S.K., Bonnet, C. & Nikaein, N., (2013), “Survey, Comparison and Evaluation of Cross Platform Mobile Application Development Tools”, 9th International Conference on Wireless Communications and Mobile Computing (IWCMC), pp. 323-328, doi: 10.1109/IWCMC.2013.6583580
- [19] Martin, S., (2020, September 30), *Native vs. Hybrid vs. Web App*, Medium, Retrieved: November 27, 2020, from <https://medium.com/better-programming/native-vs-hybrid-vs-web-app-f95c054a3c02>
- [20] Nawrocki, P., Wrona, K., Marczak, M. & Sniezynski, B., (2021), “A Comparison of Native and Cross-Platform Frameworks for Mobile Applications”, *Computer*, 54 (3), pp. 18-27, doi: 10.1109/MC.2020.2983893
- [21] Grzmil, P., Skublewska-Paszkowska, M., Łukasik, E. & Smołka, J., (2017), “Performance Analysis of Native and Cross-Platform Mobile Applications”, *Informatics Control Measurement in Economy and Environment Protection*, 7 (2), pp. 50-53, doi: 10.5604/01.3001.0010.4838
- [22] Microsoft, (2019, October 16), *System requirements*, Retrieved: March 18, 2021, from <https://docs.microsoft.com/en-us/xamarin/cross-platform/get-started/requirements>
- [23] Hermes, D., (2015), “Xamarin Mobile Application Development: Cross-Platform C# and Xamarin.Forms Fundamentals”, Apress, pp. 368-371.
- [24] Altexsoft, (2019, February 08), *The Good and the Bad of Xamarin Development*, Retrieved: January 09, 2021, from <https://www.altexsoft.com/blog/mobile/pros-and-cons-of-xamarin-vs-native/>

- [25] Vartanova Nika, (2019, December 16), *Pros and Cons of Mobile Development with Xamarin*, Iflexion, Retrieved: March 26, 2021, from <https://www.iflexion.com/blog/xamarin-pros-and-cons>
- [26] Vishal, K. & Kushwaha, A. S., (2018), "Mobile Application Development Research Based on Xamarin Platform", 4th International Conference on Computing Sciences (ICCS), pp. 115-118, doi: 10.1109/ICCS.2018.00027
- [27] Altexsoft, (2020, November 13), *13 Apps Made with Xamarin: Cross-Platform Development in Practice*, Retrieved: January 08, 2021, from <https://www.altexsoft.com/blog/mobile/13-apps-made-with-xamarin-cross-platform-development-in-practice/>
- [28] Cordova, *Overview*, Retrieved: January 12, 2021, from <https://cordova.apache.org/docs/en/3.0.0/guide/overview/>
- [29] Bosnic, S., Papp I. & Novak, S., (2016), "The Development of Hybrid Mobile Applications with Apache Cordova", 24th Conference on Telecommunications Forum (TELFOR), Belgrade, Serbia, pp. 1-4, doi: 10.1109/TELFOR.2016.7818919
- [30] Qing, Z., Ying, L., Yuan, P. G. & Sheng, L. Z., (2015), "Music Player Based on the Cordova Cross-Platform", 3rd International Conference on Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence, pp. 451-453, doi: 10.1109/ACIT-CI.2015.85
- [31] Tchrish, (2018, December 07), *Advantages of Apache Cordova Cross Platform*, Retrieved: January 13, 2021, from <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-ionic-mobile-development/>
- [32] GangBoard, (2019, June 28), *What is Apache Cordova?*, Retrieved: January 13, 2021, from <https://www.gangboard.com/blog/what-is-apache-cordova>
- [33] Rob Gravelle, *Pros and Cons of Cross-platform Mobile Development Frameworks*, HTMLGoodies, Retrieved: January 13, 2021, from <https://www.htmlgoodies.com/beyond/mobile/pros-and-cons-of-cross-platform-mobile-development-frameworks.html>
- [34] Apache Cordova, *Cordova App Showcase*, Retrieved: January 14, 2021, from <https://cordova.apache.org/>
- [35] Waranashiwar, J. & Ukey, M., (2018), "Ionic Framework with Angular for Hybrid App Development", International Journal of New Technology and Research (IJNTR), 4 (5), pp. 1-2.
- [36] Huynh, M., Ghimire, P., & Truong, D., (2017), "Hybrid App Approach: Could it Mark the End of Native App Domination?", Issues in Informing Science and Information Technology Education, 14, pp. 49-65.
- [37] Altexsoft, (2019, May 21), *The Good and the Bad of Ionic Mobile Development*, Retrieved: January 14, 2021, from <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-ionic-mobile-development/>

- [38] Tutorialspoint, *Ionic - Environment Setup*, Retrieved: March 19, 2021, from https://www.tutorialspoint.com/ionic/ionic_environment_setup.htm
- [39] Bakwa, D., Edim, E. & Yinka, O., (2017), "Hybrid Mobile Application Based on Ionic Framework Technologies", *International Journal of Recent Advances in Multidisciplinary Research*, 4 (12), pp. 3121-3130.
- [40] JavatPoint, *Ionic Slides*, Retrieved: April 24, 2021, from <https://www.javatpoint.com/ionic-slides>
- [41] Silicon IT Hub, *Top 15 Mobile Apps built on the Ionic Framework*, Retrieved: January 15, 2021, from <https://siliconithub.com/mobile-apps-built-with-ionic-framework/>
- [42] React Native, *React Native Internals*, Retrieved: January 15, 2021, from <https://www.reactnative.guide/3-react-native-internals/3.1-react-native-internals.html>
- [43] Akshat, P. & Abhishek, N., (2019), "React Native for Mobile Development: Harness the Power of React Native to Create Stunning iOS and Android Applications (2nd ed.)", Apress, pp.3-35,174-175, doi: <https://doi.org/10.1007/978-1-4842-4454-8>
- [44] Jun Kaneko, (2020, August 08), *Understanding React Native Architecture*, Retrieved: January 15, 2021, from <https://dev.to/goodpic/understanding-react-native-architecture-22hh>
- [45] Boduch, A. & Derks, R., (2020), "React and React Native: A complete hands-on guide to modern web and mobile development with React (3rd ed.)", Packt Publishing Limited, pp. 243-246.
- [46] React Native for Windows + macOS, *System Requirements*, Retrieved: March 23, 2021, from <https://microsoft.github.io/react-native-windows/docs/rnw-dependencies>
- [47] Mehul Rajput, (2020, October 24), *The Advantages and Disadvantages of Using React Native as Cross-Platform App Development*, Retrieved: January 15, 2021, from <https://www.mindinventory.com/blog/pros-cons-using-react-native/>
- [48] React Native, *React Native Overview*, Retrieved: January 16, 2021, from <https://reactnative.dev/>
- [49] Idan, G. & Al-Majdi, K., (2020), "A Freights Status Management System Based on Dart and Flutter Programming Language", *Journal of Physics: Conference Series*, 1530 (012020), pp1-6, doi: 10.1088/1742-6596/1530/1/012020
- [50] Flutter, *Flutter Architectural Overview*, Retrieved: January 17, 2021, from <https://flutter.dev/docs/resources/architectural-overview>
- [51] Shah, K., Sinha, H. & Mishra, P., (2019), "Analysis of Cross-Platform Mobile App Development Tools", *IEEE 5th International Conference for Convergence in Technology (I2CT)*, pp. 1-7, doi: 10.1109/I2CT45611.2019.9033872

- [52] Zammetti, F., (2019), "Practical Flutter: Improve your Mobile Development with Google's Latest Open-Source SDK", Apress, pp.3-97,
doi: <https://doi.org/10.1007/978-1-4842-4972-7>
- [53] Flutter, *System Requirements*, Retrieved: March 23, 2021, from
<https://flutter.dev/docs/get-started/install/windows>
- [54] Sannacode, Web & Mobile App Development, (2020, June 16), *Advantages and Disadvantages of Using Flutter*, Retrieved: January 18, 2021, from
<https://sannacode.medium.com/advantages-and-disadvantages-of-using-flutter-543bac7ced76>
- [55] Flutter, *Apps take flight with Flutter*, Retrieved: January 18, 2021, from
<https://flutter.dev/showcase>
- [56] Tutorialspoint, *Framework7 - Overview*, Retrieved: January 19, 2021, from
https://www.tutorialspoint.com/framework7/framework7_overview.htm
- [57] JavaTpoint, *Framework7 Tutorial*, Retrieved: January 19, 2021, from
<https://www.javatpoint.com/framework7-tutorial>
- [58] Framework7, *Device*, Retrieved: May 25, 2021, from
<https://framework7.io/docs/device.html>
- [59] Framework7, *Framework7 Apps Showcase*, Retrieved: January 20, 2021, from
<https://framework7.io/showcase/>
- [60] NativeScript, *How NativeScript Works*, Retrieved: January 22, 2021, from
<https://docs.nativescript.org/core-concepts/technical-overview>
- [61] NativeScript, *What is Android Runtime for NativeScript?*,
Retrieved: January 23, 2021, from <https://docs.nativescript.org/core-concepts/android-runtime/overview#what-is-android-runtime-for-nativescript>
- [62] NativeScript, *System Requirements*, Retrieved: March 23, 2021, from
<https://docs.nativescript.org/start/general-requirements>
- [63] Chue Yee Xiong, (2018, April 11), *Part 3: Cross-Platform Mobile App Frameworks-NativeScript*, Retrieved: January 24, 2021, from
<https://www.imagnet.com/2018/part-3-cross-platform-mobile-app-frameworks-nativescript/>
- [64] Alina Terebetska, (2021, January 13), *NativeScript vs React Native: Choosing a Cross-Platform Framework*, apiko, Retrieved: January 24, 2021, from
<https://apiko.com/blog/nativescript-vs-react-native-choosing-a-cross-platform-framework/>
- [65] F.S.M. Russo, R. & Camanho, R., (2015), "Criteria in AHP: A Systematic Review of Literature", *Procedia Computer Science*, 55, pp. 1123-1132,
doi: <https://doi.org/10.1016/j.procs.2015.07.081>

- [66] Atanasova-Pacemska, T., Lapevski, M. & Timovski, R., (2014), "Analytical Hierarchical Process (AHP) Method Application in the Process of Selection and Evaluation", International Scientific Conference "UNITECH 2014", Gabrovo, pp. II 373-380.
- [67] Broniewicz, E. & Ogrodnik, K., (2021), "A Comparative Evaluation of Multi-Criteria Analysis Methods for Sustainable Transport", *Energies*, 14, pp. 1-23, doi: <https://doi.org/10.3390/en14165100>
- [68] Velmurugan, R., Selvamuthukumar, S. & Manavalan, R., (2011), "Multi criteria decision making to select the suitable method for the preparation of nanoparticles using an analytical hierarchy process", *Die Pharmazie*, 66 (11), pp. 836-42, doi: 10.1691/ph.2011.1034
- [69] Rahim, R., Supiyandi, S., Siahaan, A.P.U., Listyorini, T., Utomo, A., Triyanto, W., Irawan, Y., Aisyah, S., Khairani, M., Sundari, S. & Khairunnisa, K., (2018), "TOPSIS Method Application for Decision Support System in Internal Control for Selecting Best Employees", *Journal of Physics: Conference Series*, 1028 (012052), pp. 1-8, doi:10.1088/1742-6596/1028/1/012052
- [70] Zulqarnain, R.M., Saeed, M., Ahmad, N., Dayan, F. & Ahmad, B., (2020), "Application of TOPSIS Method for Decision Making", *International Journal of Scientific Research in Mathematical and Statistical Sciences*, 7 (2), pp. 76-81.
- [71] Socorro García-Cascales, M. & Teresa Lamata, M., (2012), "On rank reversal and TOPSIS method", *Mathematical and Computer Modelling*, 56 (5–6), pp. 123-132, doi: <https://doi.org/10.1016/j.mcm.2011.12.022>