



ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΤΜΗΜΑ

Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

**Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης
για κινητές συσκευές σε React Native
με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL**

ΙΩΑΝΝΗΣ ΣΤΑΥΓΙΑΝΟΥΔΑΚΗΣ

A.M. 171079

Εισηγητής: Δρ. Γεώργιος Πρεζεράκος

ΑΘΗΝΑ

ΦΕΒΡΟΥΑΡΙΟΣ 2023

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL



ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ και ΥΠΟΛΟΓΙΣΤΩΝ

Η παρούσα διπλωματική εργασία παρουσιάστηκε

από τον

ΙΩΑΝΝΗ ΣΤΑΥΓΙΑΝΟΥΔΑΚΗ

A.M. 171079

στις 9/3/2023

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

Εισηγητής:

Δρ Γεώργιος Πρεζεράκος, Καθηγητής

Εξεταστική Επιτροπή:

Δρ. Γεώργιος Πρεζεράκος, Καθηγητής

Δρ. Παναγιώτης Γιαννακόπουλος, Καθηγητής

Δρ. Νικόλαος Ζάχαρης, Καθηγητής

Ημερομηνία εξέτασης 9/03/2023

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Ιωάννης Σταυγιανουδάκης του Μιχαήλ, με αριθμό μητρώου 171079 φοιτητής του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών της Σχολής Μηχανικών του Πανεπιστημίου Δυτικής Αττικής, δηλώνω ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής 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

Ιωάννης Σταυγιανουδάκης

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία ασχολείται με την ανάπτυξη μιας εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές με χρήση με χρήση της React Native, ενός cross-platform framework όπου θα αναλυθεί η διαφορά ενός cross-platform framework και ενός native platform framework. Ακόμα για την παροχή υπηρεσιών θα γίνεται μέσω API και θα είναι βασισμένο σε GraphQL όπου και εδώ θα αναλυθούν οι διάφορες υπηρεσίες API και τι μπορεί να προσφέρει η κάθε μια.

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

ABSTRACT

The present thesis concerns the development of a social networking application for mobile devices using React Native, a cross-platform framework where the difference between a cross-platform framework and a native platform framework will be analyzed. For the provision of services, it will be done via API and will be based on GraphQL, where the various API services and what each one can offer will be analyzed.

ΠΕΡΙΕΧΟΜΕΝΑ

Κεφάλαιο 1: Εισαγωγή	11
1.1 Περιγραφή του αντικειμένου της διπλωματικής εργασίας	11
1.2 Τι είναι Framework	11
1.2.1 Γιατί χρησιμοποιούνται τα frameworks στην ανάπτυξη λογισμικού.....	13
1.2.2 Τύποι frameworks προγραμματισμού.....	13
1.3 Framework ανάπτυξης εφαρμογών για κινητά.....	15
1.3.1 Native frameworks.....	16
1.3.2 Cross-Platform frameworks.....	17
1.4 Τι είναι τα APIs.....	23
Κεφάλαιο 2: Τεχνολογίες που θα χρησιμοποιηθούν στην ανάπτυξη του λογισμικού...	25
2.1 Εισαγωγή κεφαλαίου	25
2.2 Τα διάφορα είδη των APIs	26
2.3 Τύποι Αρχιτεκτονικών API.....	30
2.4 React Native.....	33
2.5 GraphQL.....	36
Κεφάλαιο 3: Εφαρμογή κοινωνικής δικτύωσης.....	40
3.1 Περιγραφή λειτουργίας εφαρμογής κοινωνικής δικτύωσης.....	40
3.2 Αυθεντικοποίηση και στοιχεία χρήστη.....	43
3.3 Οι κλήσεις τις graphql που χρειάστηκαν.....	43
Κεφάλαιο 4: Προβλήματα, Βελτιώσεις και Συμπεράσματα.....	49
4.1 Προβλήματα που συνάντησα και λύσεις τους.....	49
4.2 Μελλοντικές Βελτιώσεις.....	52
4.3 Συμπεράσματα.....	43

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

REST Representational state transfer

API Application Programmable Interface

SDK Software Development Kit

JSON JavaScript Object Notation

IDE Integrated Development Environment

OS Operation System

ΚΕΦΑΛΑΙΟ 1: ΕΙΣΑΓΩΓΗ

1.1 ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΑΝΤΙΚΕΙΜΕΝΟΥ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Αντικείμενο της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη μιας εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές, με την βοήθεια ενός cross-platform framework θα αναπτυχθεί και για τα δύο επικρατέστερα OS (android, ios) για κινητά χρησιμοποιώντας την παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL με στοχο την βελτιστη αναπτυξη της εφαρμογη κοινωνικης δικτύωσης αυτης.

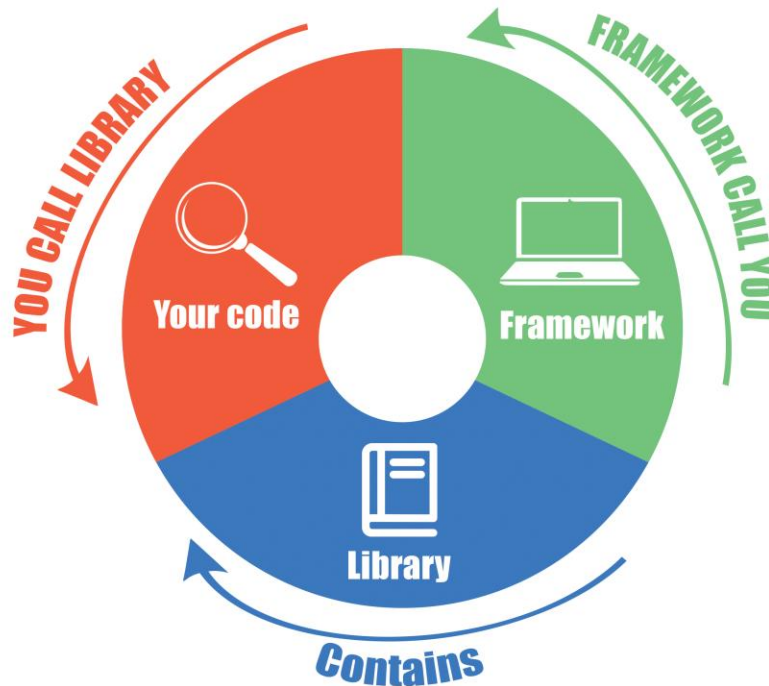
1.2 ΤΙ ΕΙΝΑΙ FRAMEWORK

Ο ορισμος του framework ως λεκτικο ειναι «βασική δομή που βασίζεται σε ένα σύστημα» δεν απέχει πολύ από την κατανόηση του τι είναι framework στον προγραμματισμό. Είτε το σύστημα είναι ένα σπίτι, ένα αυτοκίνητο, μια θεωρία, ή ,μια εφαρμογή για κινητα, η ιδέα του πλαισίου είναι η ίδια: παρέχει την υποστήριξη και τον βασικό «οδηγό» της δομής που κατασκευάζεται.

Ένα framework στον προγραμματισμό είναι ένα εργαλείο που παρέχει έτοιμα στοιχεία ή λύσεις που προσαρμόζονται προκειμένου να επιταχυνθεί η ανάπτυξη. Ένα framework μπορεί να περιλαμβάνει μια βιβλιοθήκη, αλλά ορίζεται από την αρχή της αντιστροφής του ελέγχου (IoC). Με τον παραδοσιακό προγραμματισμό, ο προσαρμοσμένος κώδικας καλεί τη βιβλιοθήκη για πρόσβαση σε επαναχρησιμοποιήσιμο κώδικα. Με το IoC, το framework καλεί προσαρμοσμένα

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

κομμάτια κώδικα όταν είναι απαραίτητο. Ένα framework μπορεί να περιλαμβάνει προγράμματα υποστήριξης, μεταγλωττιστές, βιβλιοθήκες κώδικα, σύνολα εργαλείων και API για την ανάπτυξη λογισμικού και τη δημιουργία συστημάτων. Τα framework ανοιχτού κώδικα ενημερώνονται και βελτιώνονται πάντα



1.2.1 Γιατί χρησιμοποιούνται τα frameworks στην ανάπτυξη λογισμικού;

Ο σκοπός ενός framework είναι να βοηθήσει στην ανάπτυξη, παρέχοντας τυπική, χαμηλού επιπέδου λειτουργικότητα, έτσι ώστε οι προγραμματιστές να μπορούν να επικεντρώσουν τις προσπάθειες στα στοιχεία που κάνουν το έργο μοναδικό.

Η χρήση υψηλής ποιότητας, προελεγμένης λειτουργικότητας αυξάνει την αξιοπιστία του λογισμικού, επιταχύνει τον χρόνο προγραμματισμού και απλοποιεί τις δοκιμές. Με μια ενεργή βάση χρηστών και συνεχείς βελτιώσεις κώδικα, τα frameworks συμβάλλουν στη βελτίωση της ασφάλειας και προσφέρουν μια βάση υποστήριξης.

Τελικά, τα πλαίσια χρησιμοποιούνται για εξοικονόμηση χρόνου και χρημάτων.

1.2.2 Τύποι frameworks προγραμματισμού

Υπάρχει μια ποικιλία διαφορετικών framework προγραμματισμού, καθένα από τα οποία βασίζεται σε μια γλώσσα προγραμματισμού και ειδικεύεται στη λειτουργία της, είτε πρόκειται για εφαρμογή Ιστού, βάση δεδομένων ή εφαρμογή για κινητά. Σε αυτήν την ενότητα, θα αναλύσουμε διάφορους τύπους framework προγραμματισμού, δημοφιλή frameworks για κάθε τύπο και κοινά παραδείγματα.

1. Web Frameworks

2. Front-End Frameworks

Angular

React JS

Vue JS

3. Back-End Frameworks

Ruby on Rails

PHP Frameworks (Laravel, CodeIgniter, CakePHP)

Django

ASP.NET and ASP.NET Core

4. Mobile App Development Frameworks

React Native

Flutter

Xamarin

5. Content Management Frameworks

WordPress

Drupal

Joomla

Kentico

6. Data Science Frameworks

Tensorflow

1.3 FRAMEWORK ΑΝΑΠΤΥΞΗΣ ΕΦΑΡΜΟΓΩΝ ΓΙΑ ΚΙΝΗΤΑ

Τα σύγχρονα framework ανάπτυξης εφαρμογών για κινητά παρέχουν πολλά ενσωματωμένα πλεονεκτήματα, όπως ταχύτητα, αποτελεσματικότητα και ένα περιβάλλον χωρίς σφάλματα. Το framework χρησιμοποιεί προκατασκευασμένα στοιχεία, συμπεριλαμβανομένων μεταγλωττιστών, εργαλείων εντοπισμού σφαλμάτων και εργαλειοθήκης, για να εμφανίσει την εφαρμογή στη συσκευή προορισμού χρησιμοποιώντας τον κώδικα της εταιρείας.

Η χρήση της τεχνογνωσίας μιας αξιόπιστης εταιρείας ανάπτυξης εφαρμογών για κινητά μπορεί να βοηθήσει την εταιρεία να δημιουργήσει εφαρμογές για κινητές συσκευές που είναι γρήγορες, αντιδραστικές και παρέχουν άριστες αλληλεπιδράσεις με τους χρήστες. Η χρήση πλατφορμών ανάπτυξης εφαρμογών για κινητά επιταχύνει και απλοποιεί τη διαδικασία δημιουργίας εφαρμογών.

1.3.1 native frameworks

Ο όρος native app development αναφέρεται στη δημιουργία μιας εφαρμογής για κινητά αποκλειστικά για μια ενιαία πλατφόρμα. Η εφαρμογή είναι κατασκευασμένη με γλώσσες προγραμματισμού και εργαλεία που είναι ειδικά για μια ενιαία πλατφόρμα. Για παράδειγμα, μπορείτε να αναπτύξετε μια native εφαρμογή Android με Java ή Kotlin και να επιλέξετε Swift και Objective-C για εφαρμογές iOS.

Οι native εφαρμογές είναι γνωστό ότι προσφέρουν εξαιρετική εμπειρία χρήστη, καθώς έχουν γενικά υψηλή απόδοση. Η εμπειρία χρήστη βελτιώνεται επίσης καθώς τα γραφικά είναι προσαρμοσμένα στην πλατφόρμα UX. Ωστόσο, οι νεοφυείς επιχειρήσεις ανησυχούν για το υψηλό κόστος της ανάπτυξης native εφαρμογών, καθώς πρέπει να εκτελούν ταυτόχρονη ανάπτυξη και για τις δύο πλατφόρμες.

Πλεονεκτήματα της ανάπτυξης native εφαρμογών:

- Θα έχετε πρόσβαση σε κάθε API και εργαλείο που παρέχεται από την πλατφόρμα στην οποία εργάζεστε. Τεχνικά, δεν υπάρχουν όρια στον τρόπο με τον οποίο οι προγραμματιστές θα μπορούσαν να εργαστούν με τη νέα εφαρμογή.
- Μια native εφαρμογή δημοσιεύεται ευκολότερα και συνήθως κατατάσσεται υψηλότερα στο κατάστημα εφαρμογών της πλατφόρμας, επειδή προσφέρει καλύτερη απόδοση και ταχύτητα.
- Οι εφαρμογές που έχουν δημιουργηθεί για το native περιβάλλον τείνουν επίσης να είναι πιο επεκτάσιμες, χάρη στην ευελιξία στη διαχείριση πόρων και στη σειρά των διαθέσιμων εργαλείων.
- Η άμεση αλληλεπίδραση μεταξύ του κώδικα και των υποκείμενων πόρων έχει ως αποτέλεσμα υψηλή απόδοση. Επίσης, οι εγγενείς εφαρμογές έχουν γενικά καλύτερο UX που είναι συνώνυμο της πλατφόρμας.

Μειονεκτήματα της ανάπτυξης native εφαρμογών:

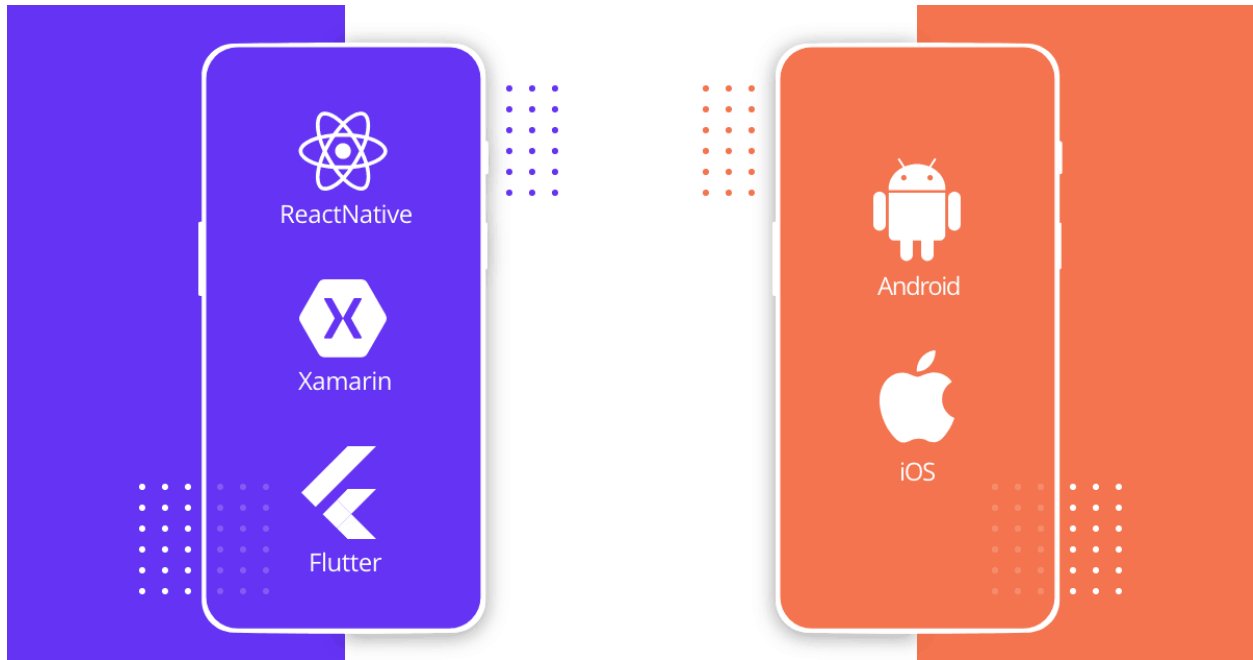
- Η δημιουργία native εφαρμογών μπορεί να είναι δαπανηρή όταν πρέπει να εκκινήσετε τόσο για iOS όσο και για Android. Σημαίνει ότι θα χρειαστεί να αναπτύξετε δύο ομάδες που εργάζονται σε διαφορετικές πλατφόρμες.
- Η ανάπτυξη native εφαρμογών είναι χρονοβόρα, καθώς η εργασία που γίνεται για μια πλατφόρμα δεν μπορεί να αντιγραφεί για άλλη. Αντίθετα, χρειάζεται μια ξεχωριστή ομάδα για να εργαστεί στην άλλη έκδοση.

1.3.2 Cross-Platform Frameworks

Η ανάπτυξη Cross-platform είναι η διαδικασία δημιουργίας μιας εφαρμογής, συνήθως μιας εφαρμογής για κινητά, που μπορεί να αναπτυχθεί σε πολλές πλατφόρμες με βάση έναν ενιαίο πηγαίο κώδικα.

Οργανισμοί, επιχειρήσεις και νεοφυείς επιχειρήσεις θέλουν συνήθως να κάνουν τις εφαρμογές τους για κινητά διαθέσιμες τόσο σε Android όσο και σε iOS, τα δύο λειτουργικά συστήματα για κινητά με σχεδόν 100% μερίδιο παγκόσμιας αγοράς. Η εργασία σε cross-platform εφαρμογές συνεπάγεται τη δημιουργία της ίδιας βάσης κώδικα όπου η εφαρμογή που προκύπτει προορίζεται να λειτουργεί τόσο σε Android όσο και σε iOS. Αυτό διαφέρει από την ανάπτυξη native εφαρμογών όπου διαφορετικές ομάδες εργάζονται σε μια έκδοση μιας εφαρμογής για κάθε διαφορετική πλατφόρμα σε παράλληλες διαδικασίες ανάπτυξης.

Σκεφτείτε την ανάπτυξη cross-platform ως την προσέγγιση με την αρχή "Γράψτε μια φορά, εκτελέστε παντού", επειδή η διαδικασία απαιτεί μόνο μια επαναχρησιμοποιήσιμη βάση κωδικών, αλλά έχει κατασκευαστεί για να εκτελείται σε πολλαπλές πλατφόρμες και λειτουργικά συστήματα για κινητές συσκευές. Ο βασικός ενεργοποιητής εδώ είναι η χρήση πλαισίων ανάπτυξης εφαρμογών για κινητά μεταξύ πλατφορμών, όπως το React Native και το Flutter.



Η επιλογή της ανάπτυξης cross-platform σε αντίθεση με την ανάπτυξη native εφαρμογών έχει τα πλεονεκτήματα και τα μειονεκτήματά της. Το θέμα δεν είναι ποια αναπτυξιακή προσέγγιση είναι καλύτερη, αλλά ποια ταιριάζει στις προτεραιότητες και τις περιστάσεις σας. Εδώ είναι τα πλεονεκτήματα και τα μειονεκτήματα της ανάπτυξης πολλαπλών πλατφορμών.

Πλεονεκτήματα της ανάπτυξης εφαρμογών cross-platform:

- Η δυνατότητα δημιουργίας μιας βάσης κώδικα μεταξύ εφαρμογών iOS και Android είναι ένα από τα κύρια πλεονεκτήματα της ανάπτυξης cross-platform για τους περισσότερους προγραμματιστές. Σε εγγενείς εφαρμογές, ο κώδικας για διαφορετικές πλατφόρμες πρέπει να γράφεται χωριστά, κάτι που συνήθως γίνεται

από δύο ξεχωριστούς προγραμματιστές ή ομάδες. Στην ανάπτυξη προσαρμοσμένων εφαρμογών για κινητά, ολόκληρη η βάση κώδικα διατηρείται σε ένα μέρος, γεγονός που εξοικονομεί χρόνο και προσπάθεια, επειδή η ομάδα ανάπτυξης μπορεί να εργάζεται και στις δύο πλατφόρμες ταυτόχρονα. Για παράδειγμα, κατά την κωδικοποίηση μιας εφαρμογής για συσκευές iOS, οι προγραμματιστές μπορούν να τη μεταγλωττίσουν για Android (και αντίστροφα) με σχετική ευκολία και σε σύντομο χρονικό διάστημα. Η ποσότητα του κώδικα που μπορεί να κοινοποιηθεί μεταξύ των πλατφορμών εξαρτάται από τις εγγενείς λειτουργικές μονάδες που είναι ενσωματωμένες στην εφαρμογή.

- Η ανάπτυξη εφαρμογών μεταξύ πλατφορμών συνεπάγεται μόνο μία διαδικασία ανάπτυξης για την παραγωγή μιας εφαρμογής που λειτουργεί σε διάφορες πλατφόρμες. Λόγω της δυνατότητας επαναχρησιμοποίησης κώδικα μεταξύ πλατφορμών, οι επιχειρήσεις εξαλείφουν την ανάγκη υλοποίησης ξεχωριστών έργων Android και iOS. Λειτουργικά, η ανάπτυξη είναι σημαντικά ταχύτερη επειδή οι ομάδες προϊόντων, σχεδιασμού και μηχανικής εργάζονται μόνο σε ένα έργο.
- Ως αποτέλεσμα της ανάγκης μόνο μιας ομάδας ανάπτυξης αντί για τουλάχιστον δύο, η δημιουργία εφαρμογών πολλαπλών πλατφορμών θα μπορούσε να είναι φθηνότερη έως και 30% από την ανεξάρτητη ανάπτυξη εφαρμογών iOS και Android. Επιπλέον, για όσους θέλουν να χρησιμοποιήσουν εργαλεία που μπορούν να δημιουργήσουν εφαρμογές ιστού και εφαρμογές για κινητά (όπως το React Native), οι προγραμματιστές μπορούν να χρησιμοποιήσουν ένα αρκετά μεγάλο

κομμάτι κώδικα και άλλα στοιχεία στον ιστό και στα κινητά. Αυτό μπορεί να συντομεύσει τη διαδικασία ανάπτυξης, επομένως να μειώσει το κόστος ανάπτυξης.

- Ο τρόπος με τον οποίο σχεδιάζονται η διεπαφή χρήστη και η πλοήγηση ποικίλλει μεταξύ διαφορετικών λειτουργικών συστημάτων και πλατφορμών, γι' αυτό χωριστές και εξειδικευμένες ομάδες κάνουν τη δουλειά στην ανάπτυξη εγγενών εφαρμογών. Τα πλαίσια ανάπτυξης πολλαπλών πλατφορμών χειρίζονται αυτές τις διαφορές από προεπιλογή. Οι cross-platform εφαρμογές παρουσιάζουν μια πιο συνεπή εμφάνιση και αίσθηση για τους χρήστες σε πολλές πλατφόρμες.
- Τα Cross-platform frameworks αυξάνονται ολοένα και περισσότερο σε δημοτικότητα, ιδιαίτερα το React Native και το Flutter. Όλο και περισσότεροι προγραμματιστές αναγνωρίζουν τις ευκαιρίες εκμάθησης αυτών των frameworks. Μαθαίνοντας μόνο ένα framework, μπορούν να εργαστούν σε εφαρμογές που αναπτύσσονται τόσο σε iOS όσο και σε Android. Επιπλέον, αναπτύσσονται επίσης περισσότερα cross-platform frameworks, δεδομένου του αυξανόμενου ενδιαφέροντος μεταξύ των νεοφυών επιχειρήσεων και των επιχειρήσεων. Οι προοπτικές για cross-platform οικοσυστήματα αναμένεται να είναι πιο ελπιδοφόρες και δυναμικές.

Επειδή η ανάπτυξη μιας εφαρμογής τόσο σε Android όσο και σε iOS με ανάπτυξη cross-platform είναι φθηνότερη και ταχύτερη, οι επιχειρήσεις μπορούν να προσεγγίσουν περισσότερους πελάτες ή χρήστες πιο αποτελεσματικά. Αυτό κάνει το cross-platform να ταιριάζει καλύτερα σε

όσους εξακολουθούν να δοκιμάζουν μια ιδέα, μια υπόθεση ή ένα πρωτότυπο που πρέπει να αναπτυχθεί σε πολλές πλατφόρμες.

Μειονεκτήματα:

- Χαμηλότερη ταχύτητα και απόδοση: Σε γενικές γραμμές, οι προγραμματιστές έχουν από καιρό αναγνωρίσει ότι οι εγγενείς εφαρμογές αποδίδουν ελαφρώς καλύτερα και πιο γρήγορα από τις εφαρμογές cross-platform όταν πρόκειται για απλούστερες εφαρμογές. Επειδή οι εφαρμογές μεταξύ πλατφορμών απαιτούν πρόσθετο επίπεδο αφαίρεσης και διαδικασία απόδοσης, είναι συνήθως πιο αργές από εκείνες που έχουν κατασκευαστεί ως εγγενείς εφαρμογές. Επιπλέον, όταν πρόκειται για εφαρμογές με ένταση CPU και GPU, φαίνεται να υπάρχουν κάποιες ενδείξεις για μεγαλύτερο χάσμα απόδοσης.
- Η προσαρμογή του κώδικα σε κάθε πλατφόρμα μπορεί να περιορίσει τη λειτουργικότητα: Οι εφαρμογές πολλαπλών πλατφορμών πρέπει να προσαρμόσουν το σχεδιασμό και τη λειτουργικότητά τους σε συγκεκριμένες πλατφόρμες, λειτουργικά συστήματα και συσκευές. Ως αποτέλεσμα, απαιτεί αρκετή προσπάθεια από τους προγραμματιστές που πρέπει να χειριστούν μεμονωμένες διαφορές για κάθε τύπο συσκευής και πλατφόρμας. Αυτό μπορεί να είναι ιδιαίτερα απαιτητικό για πολύπλοκα χαρακτηριστικά. Ως αποτέλεσμα, αυτή η δυσκολία κατά την υλοποίηση μπορεί να επηρεάσει τη λειτουργικότητα.

- Χρόνος αναμονής για ενημερώσεις και νέες δυνατότητες: Όταν η Google ή η Apple κυκλοφορούν μια ενημέρωση ή εισάγουν νέες δυνατότητες για τα αντίστοιχα λειτουργικά τους συστήματα, οι εφαρμογές πολλαπλών πλατφορμών χρειάζονται κάποιο χρόνο ή επιπλέον προσπάθεια για να υποστηρίξουν τις πιο πρόσφατες εκδόσεις. Για παράδειγμα, τα νέα SDK για εγγενείς εφαρμογές ενημερώνονται πολύ πιο γρήγορα από τα πλαίσια μεταξύ πλατφορμών.

1.4 ΤΙ ΕΙΝΑΙ ΤΑ APIS

Το API είναι το ακρωνύμιο του Application Programming Interface, το οποίο είναι ένας ενδιάμεσος λογισμικός που επιτρέπει σε δύο εφαρμογές να συνομιλούν μεταξύ τους. Κάθε φορά που χρησιμοποιείτε μια εφαρμογή όπως το Facebook, στέλνετε ένα άμεσο μήνυμα ή ελέγχετε τον καιρό στο τηλέφωνό σας, χρησιμοποιείτε ένα API. Ένα API είναι ένα σύνολο κώδικα προγραμματισμού που επιτρέπει τη μετάδοση δεδομένων μεταξύ ενός προϊόντος λογισμικού και ενός άλλου. Περιλαμβάνει επίσης τους όρους αυτής της ανταλλαγής δεδομένων.

How Do APIs Work



Οι διεπαφές προγραμματισμού εφαρμογών αποτελούνται από δύο στοιχεία:

- Τεχνική προδιαγραφή που περιγράφει τις επιλογές ανταλλαγής δεδομένων μεταξύ λύσεων με την προδιαγραφή να γίνεται με τη μορφή αίτησης για πρωτόκολλα επεξεργασίας και παράδοσης δεδομένων
- Διεπαφή λογισμικού γραμμένη σύμφωνα με τις προδιαγραφές που το αντιπροσωπεύουν.

Το λογισμικό που χρειάζεται πρόσβαση σε πληροφορίες ή λειτουργίες από άλλο λογισμικό, καλεί το API του καθορίζοντας παράλληλα τις απαιτήσεις για τον τρόπο παροχής δεδομένων/λειτουργικότητας. Το άλλο λογισμικό επιστρέφει δεδομένα/λειτουργικότητα που ζητήθηκε από την προηγούμενη εφαρμογή. Και η διεπαφή μέσω της οποίας επικοινωνούν αυτές οι δύο εφαρμογές είναι αυτό που καθορίζει το API. Σχεδόν κάθε εφαρμογή για κινητά χρησιμοποιεί API. Αυτό περιλαμβάνει κάθε τύπο εφαρμογών για κινητά, από εφαρμογές κοινωνικών μέσων έως τραπεζικές εφαρμογές και όχι μόνο. Και τα API για κινητά αποτελούν σημαντικό μέρος της στρατηγικής εφαρμογών για κινητά.

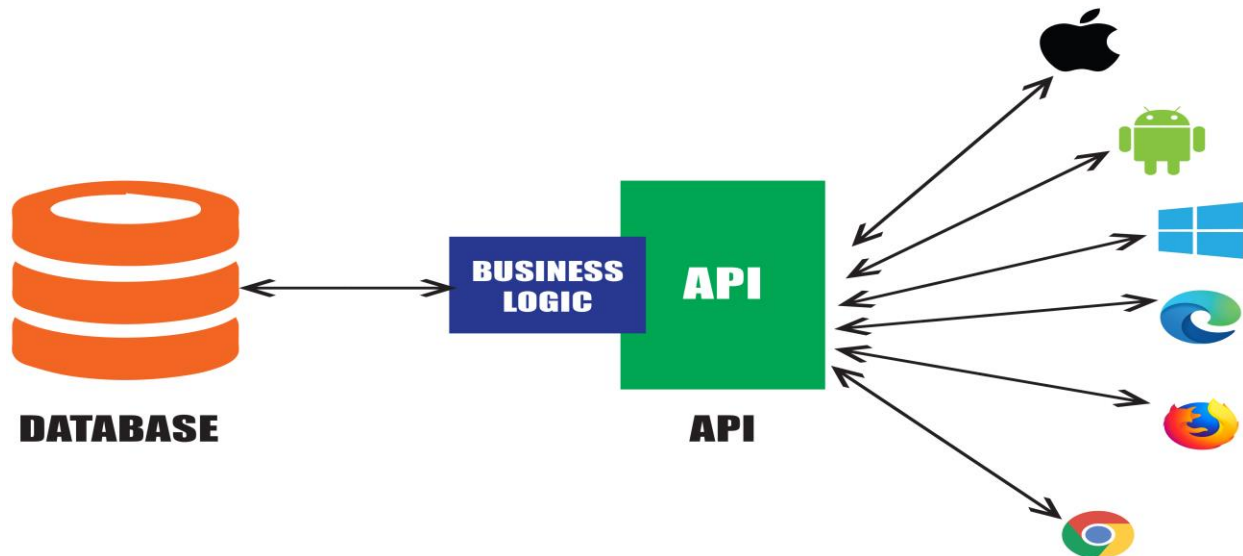
Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

ΚΕΦΑΛΑΙΟ 2 : ΤΕΧΝΟΛΟΓΙΕΣ ΠΟΥ ΘΑ ΧΡΗΣΙΜΟΠΟΙΗΘΟΥΝ ΣΤΗΝ ΑΝΑΠΤΥΞΗ ΤΟΥ ΛΟΓΙΣΜΙΚΟΥ

2.1 ΕΙΣΑΓΩΓΗ ΚΕΦΑΛΑΙΟΥ

Στο κεφάλαιο αυτό θα δούμε καλύτερα το cross-platform framework που θα χρησιμοποιηθεί για την κατεσκευή του λογισμικού, όπου θα γίνει με την βοήθεια της React Native, και η τεχνολογία που θα χρησιμοποιήσω για την μεταφορά δεδομένων απο το backend στο front end, το API δηλαδή όπου θα γίνει με την βοήθεια της GraphQL.

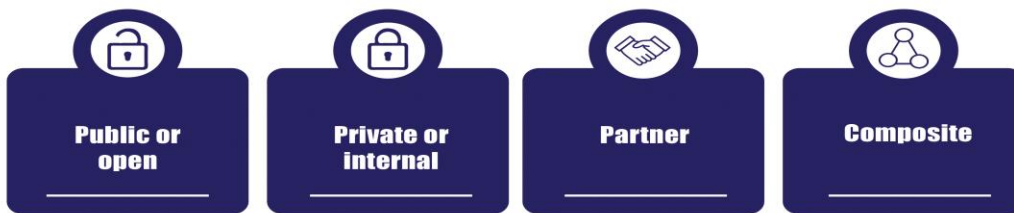
Τα API είναι κεντρικά στο σημερινό τεράστιο οικοσύστημα λογισμικού. Υπάρχουν ουσιαστικά ατελείωτοι τρόποι σύνδεσης διαφορετικών εφαρμογών ιστού και τα API τροφοδοτούν αυτές τις ενσωματώσεις στο παρασκήνιο. Επομένως, εάν θέλουμε να συνδέσουμε την εφαρμογή ή την υπηρεσία μας με τον ψηφιακό κόσμο, αξίζει να κατανοήσουμε πώς λειτουργούν τα API. Ένα βασικό πράγμα που πρέπει να γνωρίζουμε για τα API είναι ότι δεν είναι κάθε ενσωμάτωση λογισμικού το ίδιο — επομένως, δεν είναι όλα τα API ίδια. Υπάρχουν διαφορετικές κατηγορίες API που βασίζονται σε διαφορετικά πρωτόκολλα, λειτουργίες και επίπεδα πρόσβασης.



2.2 ΤΑ ΔΙΑΦΟΡΑ ΕΙΔΗ ΤΩΝ APIS

Όταν αναφερόμαστε σε API, συνήθως μιλάμε για μια υποκατηγορία API που ονομάζεται web API. Τα Web API είναι API στα οποία η πρόσβαση γίνεται χρησιμοποιώντας το πρωτόκολλο μεταφοράς υπερκειμένου (HTTP), το ίδιο πρωτόκολλο που χρησιμοποιείται για την ανάκτηση και την εμφάνιση ιστοσελίδων σε προγράμματα περιήγησης. Μπορούμε να χωρίσουμε τα Web API σε ομάδες ανάλογα με το επιδιωκόμενο επίπεδο πρόσβασης και το εύρος χρήσης. Υπάρχουν τέσσερις ευρέως αποδεκτοί τύποι WEB API: OPEN API, Partner API, Internal API και Composite API.

Types of API (Application Programming Interface)



- **Open Apis:** Τα ανοιχτά API, γνωστά και ως δημόσια API ή εξωτερικά API, είναι διαθέσιμα για χρήση από οποιονδήποτε προγραμματιστή. Ως αποτέλεσμα, τα ανοιχτά API έχουν συνήθως σχετικά χαμηλά μέτρα ελέγχου ταυτότητας και εξουσιοδότησης και συχνά περιορίζονται στα στοιχεία που μοιράζονται. Ενώ

ορισμένα ανοιχτά API είναι δωρεάν, άλλα απαιτούν χρέωση συνδρομής για χρήση, η οποία συχνά κλιμακώνεται με βάση τον αριθμό των κλήσεων που γίνονται στο API. Υπάρχουν πολλά πλεονεκτήματα στη δημοσιοποίηση των API, με το μεγαλύτερο να είναι η δυνατότητα κοινής χρήσης δεδομένων ανοιχτά. Αυτό ενθαρρύνει κάθε εξωτερική επιχείρηση ή προγραμματιστή να ενσωματωθεί με την εφαρμογή που κατέχει το API, καθιστώντας τόσο το λογισμικό τρίτων κατασκευαστών όσο και το API πιο πολύτιμα. Λόγω της έλλειψης περιορισμών και της εύκολης εφαρμογής που επιτρέπεται από το ανοιχτό API, τρίτα μέρη μπορούν να αξιοποιήσουν γρήγορα τα δεδομένα που παρέχει. Για παράδειγμα, η εφαρμογή κυκλοφορίας Waze χρησιμοποιεί δημόσια API που παρέχονται από δήμους και άλλους συνεργάτες σχετικά με το κλείσιμο δρόμων, τα ατυχήματα, τις καθυστερήσεις κατασκευής και τα οχήματα εξυπηρέτησης. Με τη σειρά του, το Waze διευκολύνει την πλοήγηση στις πόλεις, γεγονός που ευχαριστεί τους κατοίκους και προσελκύει περισσότερους επισκέπτες.

- **Partner Apis:** Τα API συνεργατών μοιράζονται εξωτερικά, αλλά μόνο μεταξύ εκείνων που έχουν επιχειρηματική σχέση με την εταιρεία που παρέχει το API. Η πρόσβαση περιορίζεται σε εξουσιοδοτημένους πελάτες με επίσημες άδειες και, επομένως, τα μέτρα ασφαλείας τείνουν να είναι ισχυρότερα με τα συνεργαζόμενα API παρά με τα δημόσια API. Ορισμένες επιχειρήσεις προτιμούν τα API συνεργατών, επειδή θέλουν (1) μεγαλύτερο έλεγχο σχετικά με το ποιος μπορεί να έχει πρόσβαση στους πόρους τους και (2) περισσότερο λόγο για τον τρόπο χρήσης

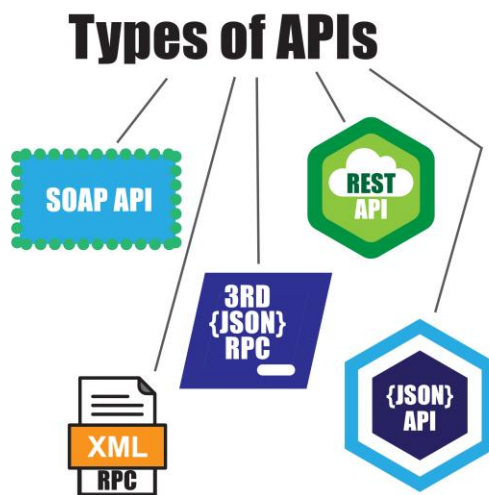
αυτών των πόρων. Για παράδειγμα, το Pinterest υιοθέτησε μια προσέγγιση βασισμένη στην υποβολή για την παροχή πρόσβασης σε νέες υπηρεσίες δεδομένων μέσω του API του, απαιτώντας από τους συνεργάτες να υποβάλουν ένα αίτημα που να περιγράφει λεπτομερώς πώς θα ήθελαν να χρησιμοποιήσουν το API πριν τους δοθεί πρόσβαση.

- **Internal Apis:** Σε αντίθεση με τα ανοιχτά API και τα API συνεργατών, τα εσωτερικά API (ονομάζονται επίσης ιδιωτικά API) δεν προορίζονται για χρήση από τρίτους. Τα εσωτερικά API διατίθενται μόνο για χρήση εντός μιας εταιρείας και προορίζονται για τον εξορθολογισμό της μεταφοράς δεδομένων μεταξύ ομάδων και συστημάτων. Οι προγραμματιστές που εργάζονται για την εταιρεία μπορούν να χρησιμοποιήσουν αυτά τα API, αλλά οι εξωτερικοί προγραμματιστές δεν μπορούν. Επειδή τα εσωτερικά API δεν τεκμηριώνονται σε ένα κιτ ανάπτυξης λογισμικού που κυκλοφόρησε δημόσια (ή καθόλου σε ορισμένες περιπτώσεις), συχνά είναι εντελώς κρυμμένα από το κοινό. Ωστόσο, πολλές εταιρείες τελικά βγαίνουν στο κοινό με τα εσωτερικά τους API. Η χρήση API για εσωτερικές μεταφορές δεδομένων θεωρείται πιο αποτελεσματική, ασφαλής και ανιχνεύσιμη. Είναι επίσης μια επεκτάσιμη λύση - όταν μια επιχείρηση εισάγει ένα νέο εσωτερικό σύστημα, αυτό το σύστημα μπορεί να επικοινωνήσει με τα υπάρχοντα συστήματα μέσω των API τους.
- **Composite Apis:** Τα Composite API συνδυάζουν πολλαπλά API που επιτρέπουν στους προγραμματιστές να ομαδοποιούν κλήσεις ή αιτήματα και να λαμβάνουν μία

ενοποιημένη απάντηση από διαφορετικούς διακομιστές. Εάν χρειάζεστε δεδομένα από διαφορετικές εφαρμογές ή πηγές δεδομένων, θα χρησιμοποιούσατε ένα σύνθετο API. Εναλλακτικά, μπορείτε να χρησιμοποιήσετε ένα σύνθετο API για να ενεργοποιήσετε μια αυτόματη αλυσίδα κλήσεων και απαντήσεων χωρίς να απαιτείται η παρέμβασή σας. Επειδή μειώνουν τον αριθμό των συνολικών κλήσεων API, τα σύνθετα API μπορούν να έχουν ως αποτέλεσμα μικρότερο φόρτο διακομιστή και συνολικά ταχύτερα συστήματα, καθώς και μειωμένη πολυπλοκότητα στο σύστημα. Συνήθως αναπτύσσονται σε μικροϋπηρεσίες στις οποίες μια εργασία μπορεί να απαιτεί δεδομένα από πολλά εσωτερικά API για να ολοκληρωθεί. Πάρτε αυτό το παράδειγμα από το Stoplight: Ας υποθέσουμε ότι θέλετε να δημιουργήσετε μια παραγγελία μέσα σε ένα API καλαθιού αγορών. Ίσως πιστεύετε ότι χρειάζεται μόνο ένα αίτημα. Αλλά, στην πραγματικότητα, πρέπει να υποβληθούν πολλά αιτήματα. Αρχικά, πρέπει να δημιουργήσετε ένα προφίλ πελάτη. Στη συνέχεια, πρέπει να δημιουργήσετε την παραγγελία, να προσθέσετε ένα στοιχείο, να προσθέσετε ένα άλλο και να αλλάξετε την κατάσταση της παραγγελίας. Αντί να κάνετε πέντε ξεχωριστές κλήσεις API διαδοχικά, μπορείτε να πραγματοποιήσετε μόνο μία με ένα σύνθετο API.

2.3 ΤΥΠΟΙ ΑΡΧΙΤΕΚΤΟΝΙΚΩΝ API

Μπορούμε επίσης να κατανοήσουμε τα API ως προς την αρχιτεκτονική τους. Η αρχιτεκτονική ενός API αποτελείται από κανόνες που καθοδηγούν ποιες πληροφορίες μπορεί να μοιραστεί ένα API με τους πελάτες και πώς μοιράζεται τα δεδομένα. Το REST, το SOAP και το RPC είναι οι πιο δημοφιλείς αρχιτεκτονικές API που χρησιμοποιούνται σήμερα — ας αποσυσκευάσουμε το καθένα με περισσότερες λεπτομέρειες.



REST

Σήμερα, η πλειονότητα των API ιστού είναι χτισμένα στο REST. Το REST, το οποίο σημαίνει μεταφορά κατάστασης αναπαράστασης, είναι ένα σύνολο οδηγιών για κλιμακούμενα,

ελαφριά και εύχρηστα API. Ένα REST API (ή "RESTful" API) είναι ένα API που ακολουθεί τις οδηγίες REST και χρησιμοποιείται για τη μεταφορά δεδομένων από έναν διακομιστή σε έναν πελάτη που ζητά.

Συνοπτικά, αυτές οι οδηγίες είναι:

- Διαχωρισμός πελάτη-διακομιστή: Όλες οι αλληλεπιδράσεις πελάτη-διακομιστή πρέπει να έχουν τη μορφή αιτήματος από τον πελάτη, ακολουθούμενο από απάντηση από τον διακομιστή. Οι διακομιστές δεν μπορούν να ζητήσουν και οι πελάτες δεν μπορούν να απαντήσουν.
- Ενιαία διεπαφή: Όλα τα αιτήματα και οι απαντήσεις πρέπει να χρησιμοποιούν το HTTP ως πρωτόκολλο επικοινωνίας και να διαμορφώνονται με συγκεκριμένο τρόπο ώστε να διασφαλίζεται η συμβατότητα μεταξύ οποιουδήποτε πελάτη και οποιουδήποτε διακομιστή. Οι απαντήσεις του διακομιστή μορφοποιούνται σε σημειογραφία αντικειμένου JavaScript (JSON).
- Χωρίς ιθαγένεια: Κάθε αλληλεπίδραση πελάτη-διακομιστή είναι ανεξάρτητη από κάθε άλλη αλληλεπίδραση. Ο διακομιστής δεν αποθηκεύει δεδομένα από αιτήματα πελατών και δεν θυμάται τίποτα από προηγούμενες αλληλεπιδράσεις.
- Επίπεδο σύστημα: Τα αιτήματα και οι απαντήσεις πρέπει πάντα να μορφοποιούνται με τον ίδιο τρόπο, ακόμη και όταν περνούν από ενδιάμεσους διακομιστές μεταξύ του πελάτη και του API.
- Προσωρινή αποθήκευση: Οι απαντήσεις διακομιστή πρέπει να υποδεικνύουν εάν ένας παρεχόμενος πόρος μπορεί να αποθηκευτεί προσωρινά από τον πελάτη και για πόσο χρονικό διάστημα.

Ακολουθώντας αυτές τις οδηγίες, τα REST API μπορούν να χρησιμοποιηθούν για γρήγορες, εύκολες, ασφαλείς μεταφορές δεδομένων, καθιστώντας τα μια δημοφιλή επιλογή μεταξύ των προγραμματιστών.

SOAP

Το SOAP (Simple Object Access Protocol) είναι ένα πρωτόκολλο για τη μετάδοση δεδομένων σε δίκτυα και μπορεί να χρησιμοποιηθεί για τη δημιουργία API. Το SOAP είναι τυποποιημένο από το World Wide Web Consortium (W3C) και χρησιμοποιεί XML για την κωδικοποίηση πληροφοριών. Το SOAP ορίζει αυστηρά πώς πρέπει να αποστέλλονται τα μηνύματα και τι πρέπει να περιλαμβάνεται σε αυτά. Αυτό καθιστά τα API SOAP πιο ασφαλή από τα REST API, αν και οι άκαμπτες οδηγίες τα καθιστούν επίσης πιο βαριά και πιο δύσκολα στην εφαρμογή τους γενικά. Για το λόγο αυτό, το SOAP εφαρμόζεται συχνά για εσωτερικές μεταφορές δεδομένων που απαιτούν υψηλή ασφάλεια και η πιο ευέλικτη αρχιτεκτονική REST αναπτύσσεται πιο συχνά οπουδήποτε αλλού. Όμως, ένα ακόμη πλεονέκτημα του SOAP είναι ότι λειτουργεί πάνω από οποιοδήποτε πρωτόκολλο επικοινωνίας (όχι μόνο το HTTP, όπως συμβαίνει με το REST).

RPC

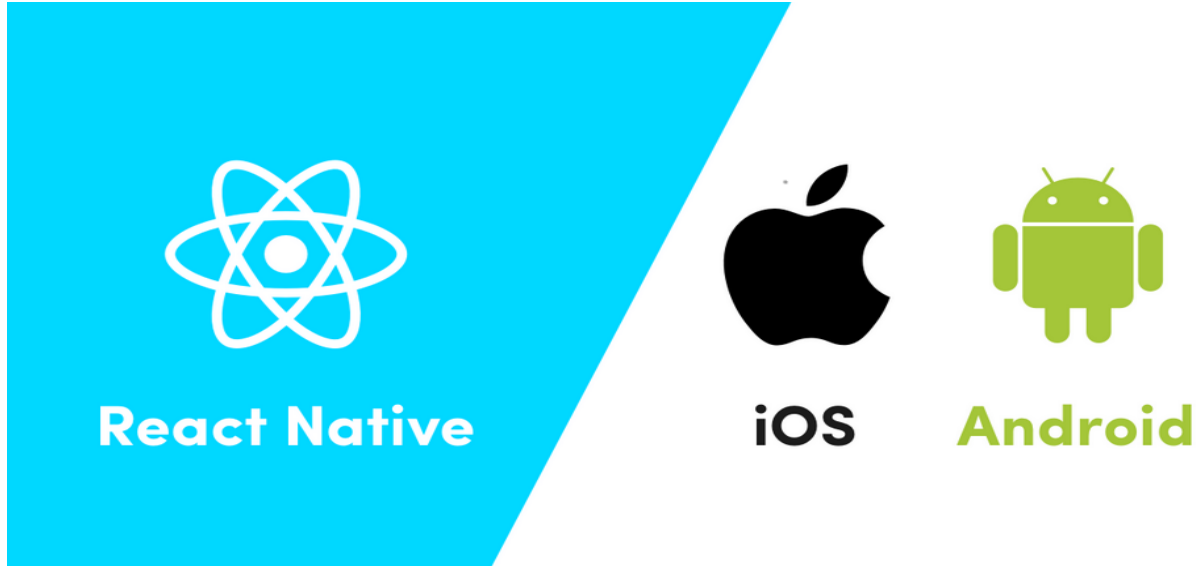
Το πρωτόκολλο RPC (Remote Procedural Call) είναι η πιο απλή από τις τρεις αρχιτεκτονικές. Σε αντίθεση με το REST και το SOAP που διευκολύνουν τη μεταφορά δεδομένων, τα RPC API επικαλούνται διεργασίες. Με άλλα λόγια, εκτελούν σενάρια σε έναν διακομιστή. Τα API RPC μπορούν να χρησιμοποιούν είτε JSON (πρωτόκολλο JSON-RPC) είτε XML (πρωτόκολλο XML-RPC) στις κλήσεις τους. Το XML είναι πιο ασφαλές και πιο εξυπηρετικό από το JSON, αλλά αυτά τα δύο πρωτόκολλα είναι κατά τα άλλα παρόμοια. Αν και το πρωτόκολλο RPC είναι αυστηρό, είναι ένας σχετικά απλός και εύκολος τρόπος για την εκτέλεση κώδικα σε απομακρυσμένα δίκτυα. Τα API RPC είναι περιορισμένα ως προς την ασφάλεια και τις δυνατότητές τους, επομένως πιθανότατα δεν θα τα βλέπετε τόσο συχνά όσο τα API REST ή SOAP στον ιστό. Ωστόσο, μπορεί να χρησιμοποιηθεί για εσωτερικά συστήματα για την υποβολή βασικών αιτημάτων διεργασιών, ειδικά πολλά ταυτόχρονα.

2.4 REACT NATIVE

Η React Native είναι ένα JavaScript framework για τη σύνταξη πραγματικών, εγγενών εφαρμογών για κινητά για iOS και Android. Βασίζεται στη React, τη βιβλιοθήκη JavaScript του Facebook για τη δημιουργία διεπαφών χρήστη, αλλά αντί να στοχεύει το πρόγραμμα περιήγησης, στοχεύει πλατφόρμες για κινητές συσκευές. Με άλλα λόγια: οι προγραμματιστές ιστού μπορούν τώρα να γράφουν εφαρμογές για κινητά που φαίνονται και αισθάνονται πραγματικά «εγγενείς», όλα από την άνεση μιας βιβλιοθήκης JavaScript που ήδη γνωρίζουμε και αγαπάμε. Επιπλέον, επειδή το μεγαλύτερο μέρος του κώδικα που γράφουμε μπορεί να μοιραστεί μεταξύ πλατφορμών, η React Native διευκολύνει την ταυτόχρονη ανάπτυξη τόσο για Android όσο και για iOS. Παρόμοια με τη React για το Web, οι εφαρμογές React Native γράφονται χρησιμοποιώντας ένα μείγμα JavaScript και σήμανσης τύπου XML, γνωστό ως JSX. Στη συνέχεια, κάτω από την κουκούλα, η "γέφυρα" της React Native καλεί τα εγγενή API απόδοσης σε Objective-C (για iOS) ή Java (για Android). Έτσι, η εφαρμογή μας θα αποδίδεται χρησιμοποιώντας πραγματικά στοιχεία διεπαφής χρήστη για κινητά, όχι προβολές ιστού και θα μοιάζει με οποιαδήποτε άλλη εφαρμογή για κινητά. Η React Native εκθέτει επίσης διεπαφές JavaScript για API πλατφόρμες, ώστε οι εφαρμογές React Native να έχουν πρόσβαση σε λειτουργίες πλατφόρμας, όπως η κάμερα του τηλεφώνου ή η τοποθεσία του χρήστη. Μπορούμε πραγματικά να χρησιμοποιήσουμε την React Native για να δημιουργήσουμε εφαρμογές για κινητές συσκευές έτοιμες για παραγωγή! Μερικά παραδείγματα:

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

το Facebook, το Palantir και το TaskRabbit το χρησιμοποιούν ήδη στην παραγωγή για εφαρμογές που αντιμετωπίζουν χρήστες.



Πλεονεκτήματα της React Native

Το γεγονός ότι η React Native αποδίδει χρησιμοποιώντας τα τυπικά API απόδοσης της πλατφόρμας κεντρικού υπολογιστή του, επιτρέπει να ξεχωρίζει από τις περισσότερες υπάρχουσες μεθόδους ανάπτυξης εφαρμογών πολλαπλών πλατφορμών, όπως η Cordova ή η Ionic. Οι υπάρχουσες μέθοδοι σύνταξης εφαρμογών για κινητά χρησιμοποιώντας συνδυασμούς JavaScript, HTML και CSS συνήθως αποδίδονται χρησιμοποιώντας προβολές ιστού. Αν και αυτή η προσέγγιση μπορεί να λειτουργήσει, έχει επίσης μειονεκτήματα, ειδικά όσον αφορά την απόδοση. Επιπλέον, συνήθως δεν έχουν πρόσβαση στο σύνολο εγγενών στοιχείων διεπαφής χρήστη της πλατφόρμας κεντρικού υπολογιστή. Όταν αυτά τα πλαίσια όντως προσπαθούν να μιμηθούν εγγενή στοιχεία διεπαφής χρήστη, τα αποτελέσματα συνήθως είναι λίγο εκτός. Η αντίστροφη μηχανική όλων των λεπτομερειών πραγμάτων όπως τα κινούμενα σχέδια απαιτεί τεράστια προσπάθεια και

μπορεί γρήγορα να ξεπεραστούν. Αντίθετα, η React Native μεταφράζει στην πραγματικότητα τη σήμανση σε πραγματικά, εγγενή στοιχεία διεπαφής χρήστη, αξιοποιώντας τα υπάρχοντα μέσα απόδοσης προβολών σε οποιαδήποτε πλατφόρμα εργάζεστε. Επιπλέον, η React λειτουργεί ξεχωριστά από το κύριο νήμα διεπαφής χρήστη, ώστε η εφαρμογή σας να διατηρεί υψηλή απόδοση χωρίς να θυσιάζει τις δυνατότητες. Ο κύκλος ενημέρωσης στο React Native είναι ο ίδιος όπως στη React: όταν αλλάζουν τα props ή η κατάσταση, η React Native αποδίδει ξανά τις προβολές. Η κύρια διαφορά μεταξύ React Native και React στο πρόγραμμα περιήγησης είναι ότι η React Native το κάνει αυτό αξιοποιώντας τις βιβλιοθήκες διεπαφής χρήστη της πλατφόρμας κεντρικού υπολογιστή του, αντί να χρησιμοποιεί σήμανση HTML και CSS. Για προγραμματιστές που έχουν συνηθίσει να εργάζονται στον Ιστό με τη React, αυτό σημαίνει ότι μπορείτε να γράφετε εφαρμογές για κινητά με την απόδοση και την εμφάνιση και την αίσθηση μιας εγγενούς εφαρμογής, ενώ χρησιμοποιείτε γνωστά εργαλεία. Η React Native αντιπροσωπεύει επίσης μια βελτίωση σε σχέση με την κανονική ανάπτυξη για κινητά σε δύο άλλους τομείς: την εμπειρία προγραμματιστή και τη δυνατότητα ανάπτυξης μεταξύ πλατφορμών.

Κίνδυνοι και μειονεκτήματα

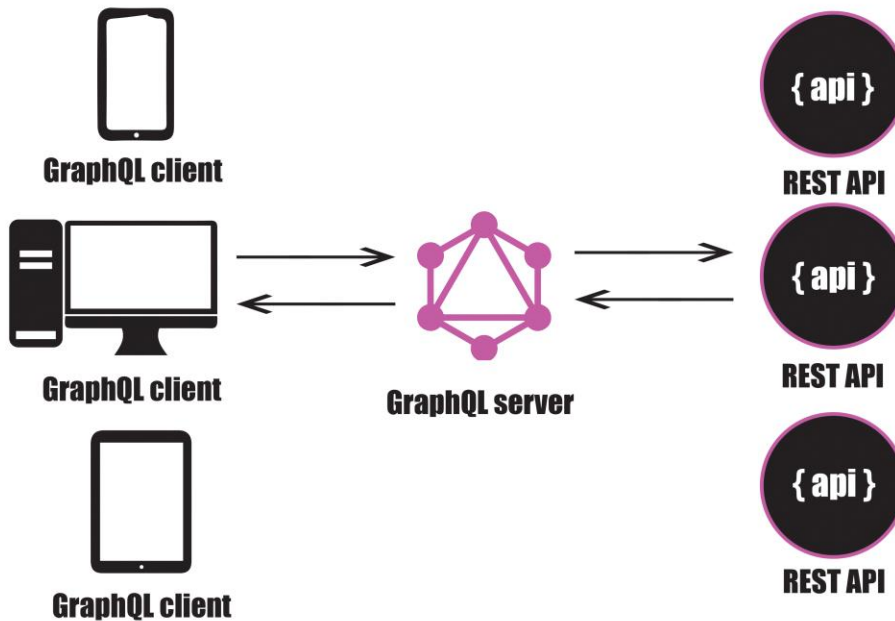
Όπως συμβαίνει με οτιδήποτε άλλο, η χρήση της React Native δεν γίνεται χωρίς τα μειονεκτήματά της. Ο μεγαλύτερος κίνδυνος είναι πιθανώς η ωριμότητα του React Native, καθώς το έργο είναι ακόμα σχετικά νέο. Η υποστήριξη iOS κυκλοφόρησε τον Μάρτιο του 2015 και η υποστήριξη Android κυκλοφόρησε τον Σεπτέμβριο του 2015. Το documentation σίγουρα έχει περιθώρια βελτίωσης και συνεχίζει να εξελίσσεται. Ορισμένες λειτουργίες σε iOS και Android εξακολουθούν να μην υποστηρίζονται και η κοινότητα εξακολουθεί να ανακαλύπτει τις βέλτιστες πρακτικές. Τα καλά νέα είναι ότι στη συντριπτική πλειονότητα των περιπτώσεων, μπορείτε να

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

εφαρμόσετε μόνοι σας υποστήριξη για API που λείπουν. Επειδή η React Native εισάγει ένα άλλο επίπεδο στο έργο σας, μπορεί επίσης να κάνει τον εντοπισμό σφαλμάτων πιο δύσκολο, ειδικά στη διασταύρωση της React και της πλατφόρμας υποδοχής. Η React Native είναι ακόμα νέα και εδώ ισχύουν οι συνήθειες προειδοποιήσεις που συνδυάζονται με την εργασία με νέες τεχνολογίες. Ωστόσο, γενικά, νομίζω ότι θα δείτε ότι τα οφέλη υπερτερούν των κινδύνων.

2.5 GRAPHQL

Η GraphQL είναι μια query γλώσσα για το API και ένας χρόνος εκτέλεσης από την πλευρά του διακομιστή για την εκτέλεση queries χρησιμοποιώντας ένα σύστημα τύπων που ορίζετε για τα δεδομένα σας. Η GraphQL δεν συνδέεται με κάποια συγκεκριμένη βάση δεδομένων ή μηχανή αποθήκευσης και αντ' αυτού υποστηρίζεται από τον υπάρχοντα κώδικα και τα δεδομένα σας.



Η GraphQL έχει σχεδιαστεί για να κάνει τα API γρήγορα, ευέλικτα και φιλικά προς τους προγραμματιστές. Μπορεί ακόμη και να αναπτυχθεί σε ένα ολοκληρωμένο περιβάλλον ανάπτυξης

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

(IDE) γνωστό ως GraphiQL. Ως εναλλακτική λύση στο REST, η GraphQL επιτρέπει στους προγραμματιστές να κατασκευάζουν αιτήματα που αντλούν δεδομένα από πολλαπλές πηγές δεδομένων σε μία κλήση API.

Επιπλέον, η GraphQL δίνει στους συντηρητές API την ευελιξία να προσθέτουν ή να καταργούν πεδία χωρίς να επηρεάζουν τα υπάρχοντα queries. Οι προγραμματιστές μπορούν να δημιουργήσουν API με όποιες μεθόδους προτιμούν και η προδιαγραφή GraphQL θα διασφαλίσει ότι λειτουργούν με προβλέψιμους τρόπους για τους πελάτες.

Οι προγραμματιστές API χρησιμοποιούν την GraphQL για να δημιουργήσουν ένα σχήμα για να περιγράψουν όλα τα πιθανά δεδομένα που μπορούν να αναζητήσουν οι πελάτες μέσω αυτής της υπηρεσίας.

Ένα σχήμα GraphQL αποτελείται από τύπους αντικειμένων, οι οποίοι καθορίζουν ποιο είδος αντικειμένου μπορείτε να ζητήσετε και ποια πεδία έχει.

Καθώς εισέρχονται τα ερωτήματα, το GraphQL επικυρώνει τα ερωτήματα έναντι του σχήματος. Στη συνέχεια, το GraphQL εκτελεί τα επικυρωμένα ερωτήματα.

Ο προγραμματιστής API επισυνάπτει κάθε πεδίο σε ένα σχήμα σε μια συνάρτηση που ονομάζεται αναλυτής. Κατά την εκτέλεση, ο αναλυτής καλείται να παράγει την τιμή.

Πλεονεκτήματα

- Ένα σχήμα GraphQL θέτει μια ενιαία πηγή αλήθειας σε μια εφαρμογή GraphQL. Προσφέρει σε έναν οργανισμό έναν τρόπο να συνενώνει ολόκληρο το API του.
- Οι κλήσεις GraphQL διεκπεραιώνονται σε ένα μόνο ταξίδι μετ' επιστροφής. Οι πελάτες λαμβάνουν αυτό που ζητούν χωρίς υπερβολική ανάκτηση.
- Οι αυστηρά καθορισμένοι τύποι δεδομένων μειώνουν την κακή επικοινωνία μεταξύ του πελάτη και του διακομιστή.
- Το GraphQL είναι ενδοσκοπικό. Ένας πελάτης μπορεί να ζητήσει μια λίστα με τους διαθέσιμους τύπους δεδομένων. Αυτό είναι ιδανικό για την αυτόματη δημιουργία τεκμηρίωσης.
- Το GraphQL επιτρέπει σε μια εφαρμογή API να εξελίσσεται χωρίς να διακόπτει τα υπάρχοντα ερωτήματα.

- Πολλές επεκτάσεις ανοιχτού κώδικα GraphQL είναι διαθέσιμες για να προσφέρουν λειτουργίες που δεν είναι διαθέσιμες με REST API.
- Η GraphQL δεν υπαγορεύει μια συγκεκριμένη αρχιτεκτονική εφαρμογής. Μπορεί να εισαχθεί πάνω από ένα υπάρχον API REST και μπορεί να λειτουργήσει με υπάρχοντα εργαλεία διαχείρισης API.

Μειονεκτήματα

- Η GraphQL παρουσιάζει μια καμπύλη εκμάθησης για προγραμματιστές που είναι εξοικειωμένοι με τα REST API.
- Η GraphQL μετατοπίζει μεγάλο μέρος της εργασίας ενός ερωτήματος δεδομένων στην πλευρά του διακομιστή, γεγονός που προσθέτει πολυπλοκότητα για τους προγραμματιστές διακομιστών.
- Ανάλογα με τον τρόπο εφαρμογής του, η GraphQL μπορεί να απαιτεί διαφορετικές στρατηγικές διαχείρισης API από τα REST API, ιδιαίτερα όταν εξετάζονται τα όρια τιμών και η τιμολόγηση.
- Η προσωρινή αποθήκευση είναι πιο περίπλοκη από ό,τι με το REST.
- Οι συντηρητές API έχουν το πρόσθετο καθήκον να γράφουν διατηρήσιμο σχήμα GraphQL.

Ένα παράδειγμα ερωτήματος GraphQL

Το παράδειγμα δείχνει πώς ένας πελάτης μπορεί να κατασκευάσει ένα ερώτημα GraphQL, ζητώντας από ένα API να επιστρέψει συγκεκριμένα πεδία σε ένα σχήμα που έχετε καθορίσει.

```
{
  me {
    name
  }
}
```

Ένα GraphQL API θα επέστρεφε ένα αποτέλεσμα όπως αυτό σε μορφή JSON:

```
{
  "me": {
    "name": "Dorothy"
  }
}
```

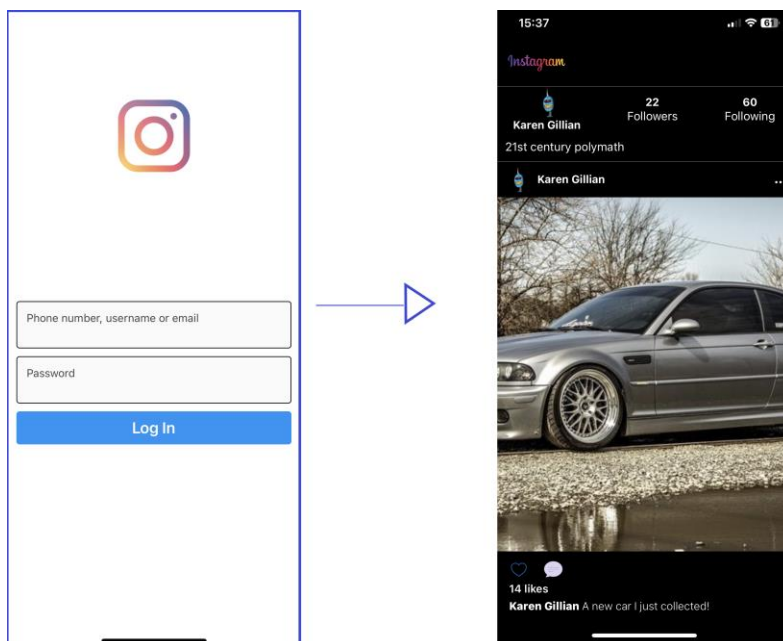

ΚΕΦΑΛΑΙΟ 3 : ΕΦΑΡΜΟΓΗ ΚΟΙΝΩΝΙΚΗΣ ΔΙΚΤΥΩΣΕΙΣ

3.1 ΠΕΡΙΓΡΑΦΗ ΛΕΙΟΥΡΓΙΑΣ ΕΦΑΡΜΟΓΗΣ ΚΟΙΝΩΝΙΚΗΣ ΔΙΚΤΥΩΣΕΙΣ

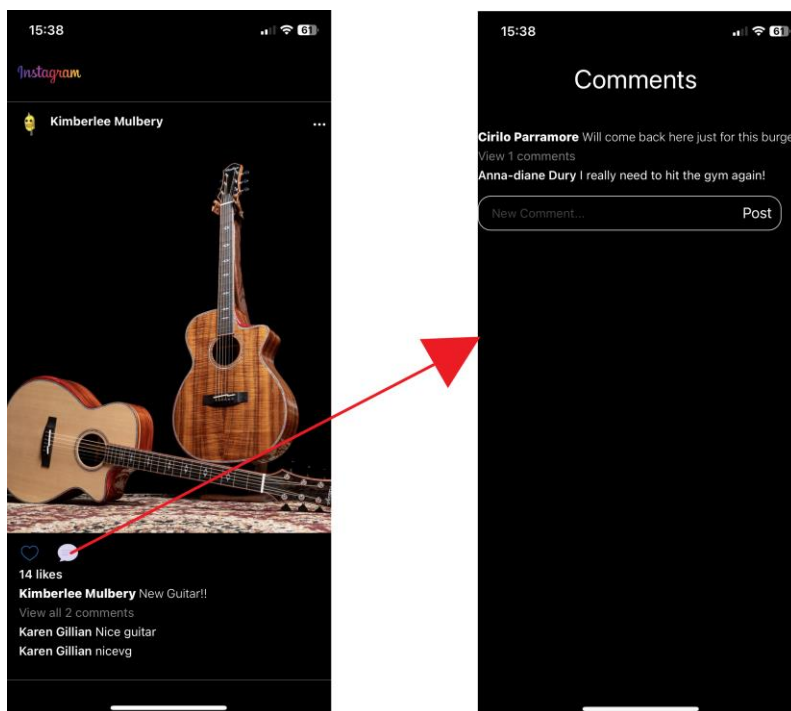
Η εφαρμογή κοινωνικής δικτύωσης που έφτιαξα είναι μια απομίμηση της εφαρμογής (Instagram) όπου μέσα από αυτήν μπορώ να σας δείξω τις τεχνολογίες GraphQL και την ανάπτυξη μιας εφαρμογής σε react native ώστε να μπορεί να παίζει και σε ios και σε android. Οι λειτουργίες που υπάρχουν σε αυτήν την εφαρμογή είναι να μπορεί ο χρήστης να κάνει Login με δεδομένα που χρησιμοποιεί η εφαρμογή μέσω του Dgraph Cloud, μια λύση που χρησιμοποιήθηκε για το στήσιμο ενός πρόχειρου backend για την εφαρμογή. Στην συνέχεια αφού κάνει Login μπορεί να δει το main feed του, τα λεγόμενα (posts) όπως υπάρχουν και στο κανονικό Instagram, ακόμα στο επάνω μέρος δείχνει τις πληροφορίες του λογαριασμού του. Σαν αλληλεπίδραση με τους άλλους χρήστες της εφαρμογής μπορεί να κάνει like και comment στα post που υπάρχουν στην αρχική του.

Η περιήγηση στην εφαρμογή γίνεται με χρήση της βιβλιοθήκης Navigation της react. Η σχεδίαση των διαδρομών περιλαμβάνει την πρόσθεση οθονών (Screens) και τη σύνδεση μεταξύ τους με τη μορφή ενεργειών.

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

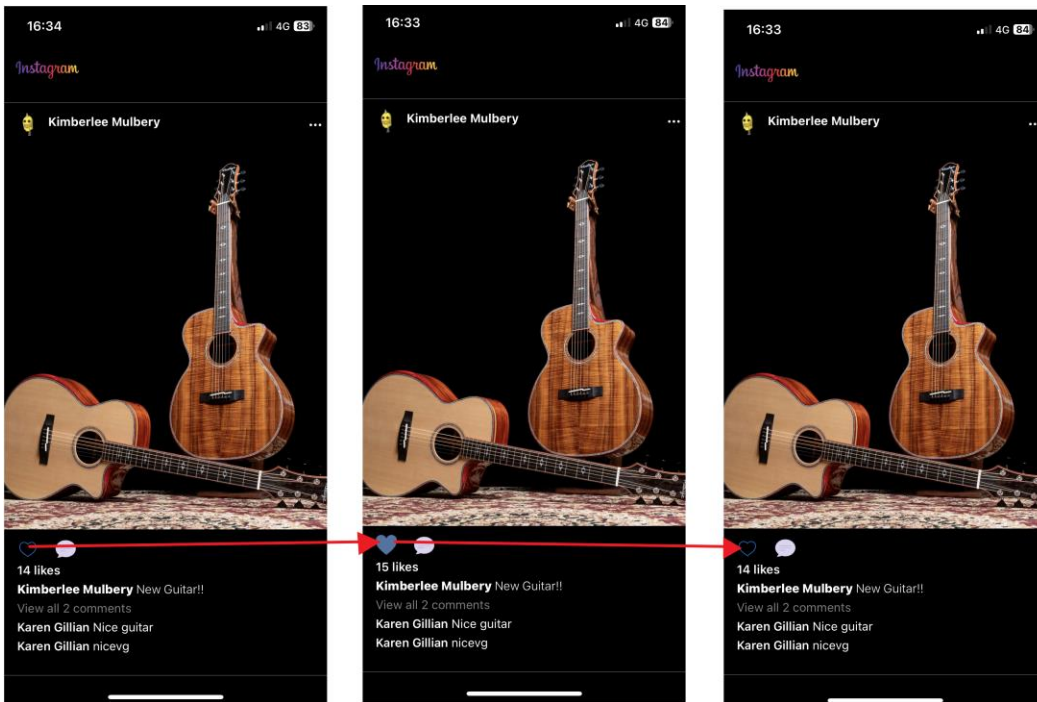


Από την οθόνη του Login στην οθόνη με τα posts του χρήστη που συνδέθηκε.

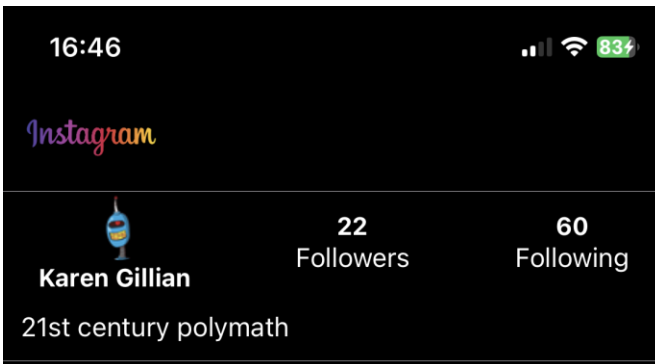


Από την οθόνη με τα posts στην οθόνη των comments.

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL



Από την οθόνη των posts εάν πατήσουμε το κουμπί των likes αυτόματα αλλάζει η εικόνα του κουμπιού και προστίθεται το like στο post. Εάν ξαναπατήσουμε το κουμπί ξανά αλλάζει η εικόνα του κουμπιού και αφαιρείτε το like από το post.



Πάνω από τα posts μπορούμε να δούμε τα στοιχεία του με τον χρήστη που έχουμε συνδεθεί όπως ονοματεπώνυμο, πόσοι ακολουθούν, πόσους ακολουθεί ο χρήστης και μερικές λεπτομέρειες που έχει επιλέξει ο χρήστης να εμφανίζονται στο προφίλ του.

3.2 ΑΥΘΕΝΤΙΚΟΠΟΙΗΣΗ ΚΑΙ ΣΤΟΙΧΕΙΑ ΧΡΗΣΤΗ

Με την εκκίνηση της εφαρμογής ζητείται η σύνδεση του χρήστη. Ο χρήστης πρέπει να συμπληρώσει το username και password του σε περίπτωση που δεν υπάρχει το username αυτό ή το password δεν είναι αυτό που έχει το username αυτό ο χρήστης δεν θα μπορεί να συνδεθεί. Με την επιτυχή εγγραφή, πραγματοποιείται σύνδεση και προώθηση στην οθόνη του main feed όπου μπορεί να δει να posts και πληροφορίες τους λογαριασμού του.

3.3 ΟΙ ΚΛΗΣΕΙΣ ΤΙΣ GRAPHQL ΠΟΥ ΧΡΕΙΑΣΤΗΚΑΝ

Ενδιαφέρον είναι να δούμε τις κλήσεις που χρειάστηκε η εφαρμογή αυτή για να έχουμε πρόσβαση στα δεδομένα.

Η πρώτη που θα δούμε είναι η κλήση που χρειάζεστε για το Login όπου μας φέρνει τον χρήστη με το username που έχουμε συμπληρώσει στο αντίστοιχο πεδίο και στην συνέχεια εγώ με βάση το password που έχω πάρει από το πεδία τσεκάρω εάν ταιριάζει με το password του χρήστη αυτού.

```
const GET_USERS = gql`
query MyQuery($username: String!) {
  getUser(username: $username) {
    password
    about
    avatarImageURL
    follower
    following
    name
    username
  }
}
`;
```

Όπως φαίνεται στην κλήση της graphql ζητάμε να μας φέρει το password, το όνομα, τον αριθμό των ακόλουθων, τον αριθμό που ακολουθάει, την εικόνα προφίλ, και κάποιες πληροφορίες του χρήστη (about)

Η επόμενη είναι τα posts που θα φαίνονται στον χρήστη στο main feed του. Καλούμε την κλήση όπου μας επιστρέφει όλα τα posts και στην συνέχεια μοιράζω τις πληροφορίες στο κάθε screen που τις χρειάζεται.

```
const GET_POSTS = gql`
{
  queryPost {
    id
    description
    imageURL
    likes
    postedBy {
      avatarImageURL
      name
    }
    comments {
      text
      commentBy {
        avatarImageURL
        name
      }
    }
    like {
      id
      likeBy {
        username
      }
      likeTo {
        id
      }
    }
  }
}
```

Οι πληροφορίες που μας επιστρέφει η κλήση της graphql είναι το id του post, την περιγραφή, την εικόνα του post, τα likes του post, ποιος το κοινοποίησε (όνομα και εικόνα προφίλ), τον πίνακα με τα σχόλια του post όπου περιέχει το σχόλιο και από ποιον έγινε το σχόλιο (όνομα και εικόνα προφίλ), επίσης από ποιον έχει γίνει το like.

Στην συνέχεια στη οθόνη με τα post έχουμε την τρεις κλήσεις για τα like, η μια προσθέτει ένα like στο post αυτό (ποιος το έκανε, κλπ.), η δεύτερη το αφαιρεί και η τρίτη ανανεώνει τον αριθμό των likes του post, που είναι ξεχωριστό πεδίο με τα προηγούμενα. Κάθε φορά τσεκάρει η εφαρμογή εάν το συγκεκριμένο post έχει like από αυτόν τον χρήστη ή όχι ώστε να ξέρει ποια κλείσει να καλέσει.

```
const ADD_LIKE = gql`
mutation AddLikes($username: String!, $id: ID!) {
  addLike(input: {likeBy: {username: $username}, likeTo: {id: $id}}) {
    like {
      likeBy {
        username
      }
      likeTo {
        id
      }
    }
  }
}
`;
```

Η κλήση αυτή είναι που προσθέτει ένα like στο post αυτό και χρειάζεστε τις εξής πληροφορίες το username του χρήστη και το id του post για να γίνει η κλείσει.

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

```
const REMOVE_LIKE = gql`
mutation MyMutation($eq: String = "", $id: [ID!] = "") {
  deleteLike(filter: {}) {
    like {
      likeBy(filter: {username: {eq: $eq}}) {
        username
      }
      likeTo(filter: {id: $id}) {
        id
      }
    }
  }
}
```

Η κλείσει αυτή είναι που αφαιρεί ένα like από ένα post και χειρίζετε τις ίδιες πληροφορίες με το προηγούμενο.

```
const UPDATE_NUMBER_OF_LIKES = gql`
mutation MyMutation($id: [ID!], $likes: Int!) {
  updatePost(input: {filter: {id: $id}, set: {likes: $likes}}) {
    numUids
  }
}
```

Και η τρίτη κλείσει που ανανεώνει τον αριθμό των likes στο post αυτό όπου χειρίζετε το id του post και τον αριθμό των likes.

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

Και τελευταία κλήση είναι η κλήση για τα comments για να προσθέσω ένα comment στο post που έχει επιλέξει ο χρήστης. Αφού έχει γράψει το σχόλιο του στην οθόνη των comments όταν πατήσει δημοσιεύσει χρησιμοποιείτε η κλήση αυτή.

```
const ADD_COMMENT = gql`
mutation MyMutation($username: String!, $id: ID!, $text: String!) {
  addComment(input: {text: $text, commentBy: {username: $username}, commentOn: {id: $id}}) {
    numUids
  }
}
`;
```

Η κλησει αυτη χρειαζετε το username του χρησητη, το id του post και φυσικα το comment που εγραψε ο χρησητης.

ΚΕΦΑΛΑΙΟ 4 : ΠΡΟΒΛΗΜΑΤΑ, ΒΕΛΤΙΩΣΕΙΣ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ

4.1 ΠΡΟΒΛΗΜΑΤΑ ΠΟΥ ΣΥΝΑΤΗΣΑ ΚΑΙ ΟΙ ΛΥΣΕΙΣ ΤΟΥΣ

Στην υλοποίηση της εφαρμογής αυτής υπήρξαν κάποια προβλήματα όπου μετά από ψάξιμο και μελέτη λυθήκαν μαθαίνοντας έτσι νέα πράγματα. Θα αναφέρω κάποια από αυτά που αξίζει να υποθούν.

1. Στην προσπάθεια να γίνει το query προς την βάση, αλλά να γίνεται το query όταν πατάω το login button. Η graphql έχει το useQuery αλλά το useQuery χρησιμοποιείτε αυτόματα με το που ανοίξει αυτό το screen. Γι' αυτό τον σκοπό υπάρχει το useLazyQuery όπου είναι σαν function που ενεργοποιείτε μόλις το καλέσεις. Αλλά όταν το καλέσεις μια φορά μετά εκτελείτε επανειλημμένα, αυτό λύθηκε χρησιμοποιώντας μια useEffect όπου κοιτάει τα data που γυρνάνε από το query και κάθε φορά που αλλάζει θα κάνει trigger

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

```
const LoginForm = ({navigation}) => {  
  
  const [ username, setUsername ] = useState("");  
  const [ password, setPassword ] = useState("");  
  const [ dataFlag, setDataFlag ] = useState(false);  
  const [ getUsers, { loading, error, data } ] = useLazyQuery(GET_USERS, {fetchPolicy: 'network-only'}, {  
    onCompleted: someData => {  
      console.log(someData)  
    }  
  });  
  
  React.useEffect(() => {  
    {data &&  
      data.getUser &&  
      data.getUser.password === password ? handleValidCredentials(data.getUser) : handleWrongCredentials()  
    }, [data]  
  });  
  
  const handleValidCredentials = (user) => {  
    navigation.navigate("HomeScreen", {  
      screen: "HomeScreen",  
      user : {user},  
      navigation: {navigation}  
    });  
  };  
  
  const handleWrongCredentials = () => {  
    if (dataFlag && password != null && password != "") {  
      return <Text style={styles.setColorRed}>Username or Password is wrong!</Text>;  
    }  
  }  
}
```

(Screenshot από τον κώδικα του προβλήματος 1)

2. Στην οθόνη των comments όταν γινόταν κοινοποίηση ενός comment και σε επέστρεφε στην αρχική οθόνη δεν είχε προστεθεί οπτικά το comment αλλά μόνο στην βάση, μόνο εάν γινόταν refresh εμφανιζόταν. Η λύση ήταν ότι μόλις γινόταν κοινοποιήσει μπήκε ένα delay ώστε να προλάβει να ανέβει στην βάση πριν φορτώσει την αρχική οθόνη με τα posts.

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

```
const delay = ms => new Promise(res => setTimeout(res, ms));

const addComment = async () => {
  console.log("asdas")
  addcomment({
    variables: {
      id: post.id,
      username: user.username,
      text: comment
    }
  })
  await delay(1000);
  navigation.replace("HomeScreen", {
    screen: "HomeScreen",
    user : {user},
    navigation: {navigation}})
};
```

(Screenshot από τον κώδικα του προβλήματος 2)

4.2 ΜΕΛΛΟΝΤΙΚΕΣ ΒΕΛΤΙΩΣΕΙΣ

Σαφώς στην παρούσα φάση που είναι εφαρμογή υπάρχουν πολλά περιθώρια βελτιώσεις ανάλογα το πόσο μεγάλη θέλουμε να γίνει. Υπάρχουν όμως κάποια βασικά πράγματα που θα μπορούσαν να γίνουν ώστε να βελτιωθεί η εφαρμογή.

1. Η πρώτη βελτίωση που θα μπορούσε να γίνει για να βελτιώσει το UX είναι στο Login να λαμβάνω κάποιο token του χρήστη και έτσι κάθε φορά που ανοίγει την εφαρμογή ο χρήστης να μην χρειάζεται να βάζει τα στοιχεία του για να συνδεθεί.

2. Στην συνέχεια θα χρειαστεί μια φόρμα για να μπορεί να κάνει register ο χρήστης όπου θα μπορεί να εξατομικεύσει το προφίλ του πριν ξεκινήσει την περιήγηση.

3. Επιπλέον λειτουργία είναι να υπάρχει η δυνατότητα ο χρήστης να μπορεί να επεξεργαστεί το προφίλ του και να αλλάξει κάτι που δεν του αρέσει όπως να βάλει καινούργια εικόνα προφίλ, κλπ.

4. Μια ενδιαφέρουσα αλλά δύσκολη υλοποίηση είναι η δημιουργία συστήματος αυτοματοποιημένων ειδοποιήσεων push notification, όπου οι χρήστες θα ενημερώνονται για κάποιο like που έγινε σε post τους ή εάν έγινε κάποιο σχόλιο στο posts τους με αποτέλεσμα την αμεσότερη αλληλεπίδραση. Τα push notifications μοιάζουν με sms, είναι ένα pop-up και χρησιμοποιούνται συνήθως σε εφαρμογές για κινητά.

Τέλος εξαρτάται από εμάς το πόσο με μεγάλη θέλουμε να γίνει η εφαρμογή, όπου θα μπορούσε στην συνέχεια να προστεθούν τα stories όπως υπάρχουν στο Instagram όπου θα έχει άμεση αλληλεπίδραση ο χρήστης και εάν θα θέλουμε να έχει κάποιου είδους επικοινωνίας ο ένας χρήστης με τον άλλον με προσωπικά μηνύματα ή απλά θα αρκεστούμε στα posts.

4.3 ΣΥΜΠΕΡΑΣΜΑΤΑ

Στην διπλωματική εργασία προσπάθησα μέσα από μια εφαρμογή κοινωνικής δικτύωσης που έφτιαξα να δείξω δυο τεχνολογίες, η μια είναι το cross platform framework όπου είναι η react native, η επιλογή έγινε διότι είχα ασχοληθεί ήδη σε προσωπικό επίπεδο με την react native και επειδή είχα μια εμπειρία με το native platform του android studio και την java μπορούσα να δείξω τις διαφορές μεταξύ τους και που μπορούμε να επωφεληθούμε εάν γράφουμε έναν κώδικα και να έχουμε εφαρμογή και για τα δυο επικρατέστερα λογισμικά (android, ios), επειδή όλο και περισσότερο βλέπουμε να επικρατούν τα cross platform frameworks για την ανάπτυξη εφαρμογών στα κινητά, κυρίως με έμπνευση ότι το facebook και το Instagram είναι γραμμένα κατά κύριο λόγο σε react native μου έδωσε την ώθηση να γράψω και εγώ μια παρόμοια εφαρμογή κοινωνικής δικτύωσης.

Η δεύτερη τεχνολογία που δείχνω στην πτυχιακή είναι μια διαφορετική προσεγγίσει που έχουν οι περισσότερες εφαρμογές για να παίρνουν δεδομένα από κάποιο api, οι περισσότερες εφαρμογές χρησιμοποιούν το rest api, όμως το κακό με αυτό είναι ότι μας έρχονται όλα τα δεδομένα από τον server. Αυτό που χρησιμοποίησα εγώ είναι το GraphQL, μετά από παρότρυνση του καθηγητή μου Κ. Πρεζερακου, χωρίς να έχω εγώ κάποια γνώση σε αυτό αλλά αποδείχτηκε χρήσιμη γνώση και πιο εύκολο εργαλείο για το fetching των δεδομένων από τον server αφού το κύριο πλεονέκτημα είναι ότι μπορούμε να ζητήσουμε συγκεκριμένα ποια δεδομένα θέλουμε να μας έρθουν από την βάση μας.

Ανάπτυξη εφαρμογής κοινωνικής δικτύωσης για κινητές συσκευές σε React Native με παροχή υπηρεσιών μέσω API βασισμένο σε GraphQL

Εν τέλη, όσον αφορά την εφαρμογή σίγουρα δεν είναι κάποια μεγάλη εφαρμογή, αφού χρειάζεται και ένα καλό backend για τα δεδομένα και λείπουν κάποιες λειτουργίες αλλά τα βασικά λειτουργούν κανονικά ώστε να επιδείξω τις τεχνολογίες που αναφέρθηκαν, με ένα σωστό backend η εφαρμογή θα μπορεί να χρησιμοποιηθεί από χρήστες.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. <https://www.netsolutions.com/insights/what-is-a-framework-in-programming/>
2. <https://www.altexsoft.com/blog/engineering/what-is-api-definition-types-specifications-documentation/>
3. <https://www.mulesoft.com/resources/api/what-is-an-api>
4. <https://technostacks.com/blog/mobile-app-development-frameworks/>
5. <https://www.netguru.com/blog/cross-platform-vs-native-app-development>
6. <https://blog.hubspot.com/website/types-of-apis>
7. <https://stoplight.io/api-types>
8. <https://www.oreilly.com/library/view/learning-react-native/9781491929049/ch01.html>
9. <https://www.redhat.com/en/topics/api/what-is-graphql>