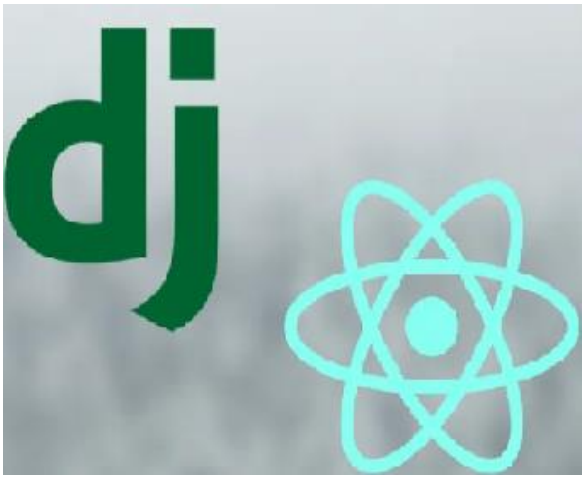
ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Διπλωματική Εργασία

Διαδικτυακή εφαρμογή χρησιμοποιώντας React και Django

Φοιτητής: Μάρκου Βασίλειος
ΑΜ: 71344116



Επιβλέπων Καθηγητής

Νικόλαος Ζάχαρης
Καθηγητής

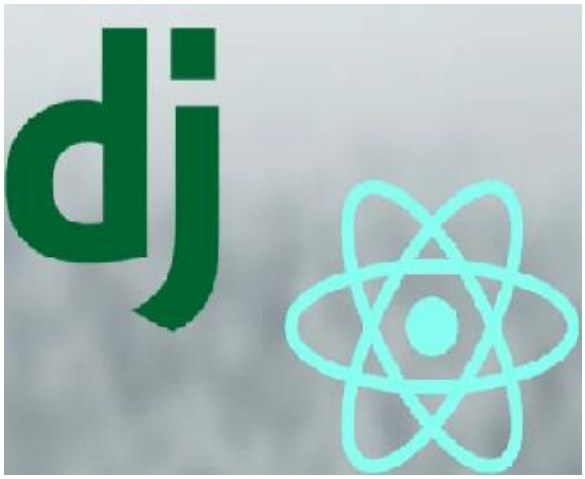
ΑΙΓΑΛΕΩ, ΜΑΡΤΙΟΣ 2021

UNIVERSITY OF WEST ATTICA
FACULTY OF ENGINEERING
DEPARTMENT OF COMPUTER ENGINEERING AND COMPUTERS

Diploma Thesis

Web application using React and Django

Student: Vasileios Markou
Registration Number: 71344116



Supervisor

Nikolaos Zaharis
Professor

AIGALEO, MARCH 2021

Νικόλαος Ζάχαρης : 

Digitally signed by Nikolaos Zacharis
Date: 2021.05.24 13:32:05 +03'00'

Παναγιώτης Γιαννακόπουλος : Panagiotis Giannakopoulos

Digitally signed by Panagiotis Giannakopoulos
Date: 2021.05.21 13:43:50 +03'00'

Γεώργιος Πρεζεράκος : Georgios Prezerakos

Digitally signed by Georgios Prezerakos
Date: 2021.05.24 11:56:05 +03'00'

ΔΗΛΩΣΗ ΠΕΡΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ ΚΑΙ ΛΟΓΟΚΛΟΠΗΣ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπόγραφα ότι η παρούσα εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα αποκλειστικά και ότι είμαι ο αποκλειστικός συγγραφέας του κειμένου της.

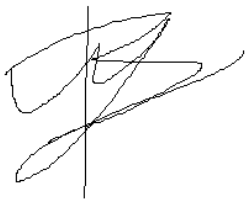
Η εργασία μου δεν προσβάλλει οποιασδήποτε μορφής δικαιώματα πνευματικής ιδιοκτησίας, προσωπικότητας ή προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής ή λογοκλοπής.

Κάθε βοήθεια που έλαβα για την ολοκλήρωση της εργασίας είναι αναγνωρισμένη και αναφέρεται λεπτομερώς στο κείμενό της. Ειδικότερα, έχω αναφέρει ευδιάκριτα μέσα στο κείμενο και με την κατάλληλη παραπομπή όλες τις πηγές δεδομένων, κώδικα προγραμματισμού Η/Υ, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών που χρησιμοποιήθηκαν, είτε κατά κυριολεξία είτε βάσει επιστημονικής παράφρασης, και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Επιπλέον, όλες οι πηγές που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης κατά τα διεθνή πρότυπα.

Τέλος δηλώνω ενυπόγραφα ότι αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της είναι προϊόν λογοκλοπής.

Ημερομηνία 18/5/2021
(Ονοματεπώνυμο φοιτητή)
Μάρκου Βασίλειος

(Υπογραφή)



Αφιερώνω την διπλωματική μου εργασία στους γονείς μου Γεώργιο και Ευτέρπη, στον αδελφό μου Αθανάσιο στη Νικολέττα και στη μνήμη των παππούδων μου Πανάγου και Αθανάσιου.

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Νικόλαο Ζάχαρη για την καθοδήγηση συμβουλές και υπομονή του. Τους δυο καθηγητές της επιτροπής κ. Πρεζεράκο και κ. Γιαννακόπουλο. Επίσης θα ήθελα να ευχαριστήσω τα κοντινά μου πρόσωπα, Σωτήρη Παπαϊωάννου για την βοήθεια και συμβουλές του που συνέβαλαν στην επίλυση δύσκολων τεχνικών προβλημάτων, την Νικολέττα Στάθη για τις διορθώσεις στο συντακτικό του κειμένου. Τέλος ευχαριστώ τους γονείς μου για την υποστήριξή τους σε όλα τα χρόνια της σχολής μου.

Περίληψη

Η διπλωματική αυτή εργασία αφορά τη δημιουργία μιας διαδικτυακής εφαρμογής για την διαχείριση μιας αποθήκης και του ηλεκτρονικού καταστήματός της χρησιμοποιώντας React framework και django. Οι αποθήκες αυτές είναι υπεύθυνες για την ταξινόμηση προϊόντων και την προετοιμασία παραγγελιών. Στόχος της εφαρμογής είναι η διευκόλυνση του εργατικού προσωπικού ώστε να υπάρχει ομαλή συνεργασία για την αποφυγή λαθών, η διευκόλυνση των πελατών στη περιήγηση και διαλογή των προϊόντων και στη δημιουργία της παραγγελίας τους.

Λέξεις – κλειδιά

Frameworks, Django, React, διαχείριση αποθήκης, Storage handler.

Abstract

This thesis concerns the creation of an online application for the management of a warehouse and its online store using React framework and django. These warehouses are responsible for product classification and order preparation. The aim of the application is to facilitate the working staff so that there is smooth cooperation to avoid mistakes, facilitating customers to browse and sort products and also to facilitate customers in creating their order.

Keywords

Frameworks, Django, React, warehouse managing, Storage handler.

Περιεχόμενα

ΕΙΣΑΓΩΓΗ

Οι εργασίες που πρέπει να διεκπεραιωθούν σε ένα περιβάλλον αποθήκης, απαιτούν προγραμματισμό και ακρίβεια. Είναι απαραίτητο να υπάρχουν καταγεγραμμένα τα προϊόντα και ταξινομημένα ανάλογα με τις κατηγορίες τους. Όταν ανατίθεται σε έναν εργαζόμενο να προετοιμάσει μια παραγγελία θα πρέπει να σπαταλήσει χρόνο στο να βρει την κατηγορία προϊόντος, να δει τη διαθεσιμότητά του και να ενημερώσει όλους τους συναδέλφους του ότι έχει διεκπεραιώσει το δικό του κομμάτι της εργασίας. Σε αυτά τα προβλήματα που κωλυσιεργούν την παραγωγικότητα βασίζεται η εφαρμογή αυτή. Με ιεραρχική κατανομή των ρόλων, μέσα σε μια πλατφόρμα όπου όλα τα προϊόντα είναι ταξινομημένα και αρχειοθετημένα μπορούν οι εργαζόμενοι να επεξεργάζονται ή να αναλαμβάνουν εργασίες ενημερώνοντας το σύστημα αυτόματα. Ο στόχος του ηλεκτρονικού καταστήματος είναι να παρουσιάζει με τρόπο κατανοητό τα προϊόντα της αποθήκης, να διευκολύνει τη διαδικασία εύρεσης συγκεκριμένου προϊόντος από τον πελάτη, να του παρέχει τις κατάλληλες πληροφορίες για οποιοδήποτε προϊόν, να τον καθοδηγεί στη διαδικασία παραγγελίας και να τον ενημερώνει για τα κόστη της παραγγελίας του. Στην εφαρμογή υπάρχουν καταγεγραμμένα τα προϊόντα, τα οποία είναι ταξινομημένα σε κατηγορίες, η ποσότητα και κατ' επέκταση η διαθεσιμότητά τους. Αυτά τα στοιχεία προβάλλονται στο ηλεκτρονικό κατάστημα και η ενημέρωση τους γίνεται αυτόματα. Υπεύθυνοι για την καταγραφή των απαραίτητων στοιχείων για την λειτουργία της εφαρμογής είναι το προσωπικό που διαχειρίζεται την εφαρμογή. Η εφαρμογή παρέχει και άλλες υπηρεσίες οι οποίες θα αναφερθούν και εξηγηθούν παρακάτω.

Αντικείμενο της διπλωματικής εργασίας

Η storage handler είναι μια διαδικτυακή εφαρμογή που έχει στόχο τη διαχείριση μιας αποθήκης και του ηλεκτρονικού καταστήματός της. Για τη δημιουργία της χρησιμοποιήθηκαν τα frameworks React και Django καθώς και άλλες τεχνολογίες. Οι αποθήκες είναι χώροι στους οποίους τα προϊόντα ταξινομούνται και προετοιμάζονται για τις παραγγελίες. Το ηλεκτρονικό κατάστημα είναι υπεύθυνο για τη παρουσίαση των προϊόντων και των πληροφοριών τους σε πελάτες καθώς και για τη παροχή δυνατότητας σε πελάτες να κάνουν παραγγελίες με σιγουριά και ευκολία. Πρόκειται για μια διαχείριση αποθήκης εσωτερικά που έχει στόχο να διευκολύνει τους εργαζόμενους και την παρουσίαση της εικόνα της σε πελάτες απομακρυσμένα μέσω διαδικτύου.

Σκοπός και στόχοι

Σκοπός αυτής της διπλωματικής εργασίας είναι να δημιουργηθεί μια εφαρμογή η οποία να διαχειρίζεται μια αποθήκη και το ηλεκτρονικό κατάστημά της χρησιμοποιώντας σύγχρονες τεχνολογίες προγραμματισμού. Ο στόχος είναι μέσα από αυτή την διαδικασία να μελετηθούν τα frameworks Django και React καθώς και άλλες απαραίτητες τεχνολογίες.

1 ΚΕΦΑΛΑΙΟ 1ο : Σύγχρονα frameworks για client – server side scripting

1.1 Τι είναι client – server

Client ονομάζουμε έναν υπολογιστή ή ένα λογισμικό που αιτείται υπηρεσίες. Οι υπηρεσίες αυτές μπορούν να υπάρχουν στους ίδιους σταθμούς εργασίας ή σε απομακρυσμένους σταθμούς εργασίας που συνδέονται μεταξύ τους μέσω ενός δικτύου. Την επικοινωνία την ξεκινάει πάντα ο client. Ένας client θα πρέπει να μπορεί να τρέχει το λογισμικό των γραφικών διεπαφών χρηστών (GUIs), να δημιουργεί τις αιτήσεις για πληροφορίες και να τις στέλνει στον server, καθώς και να αποθηκεύει επιστρεφόμενες πληροφορίες.

Με τη σειρά του ο server απαντάει στα αιτήματα που δέχεται από τους clients. Οι server δεν ξεκινούν επικοινωνίες, αναμένουν τις αιτήσεις των clients. Ένας server έχει τη δυνατότητα να αποθηκεύει, να ανακτά και να προστατεύει πληροφορίες, να επιθεωρεί τις αιτήσεις των clients, να δημιουργεί εφαρμογές διαχείρισης πληροφοριών (π.χ δημιουργία αντιγράφων, ασφάλεια κ.α.) αλλά και να διαχειρίζεται ο ίδιος πληροφορίες.

Όταν επομένως, ο client στέλνει μήνυμα για να καλέσει τον server και αποκτούν επικοινωνία μεταξύ τους, τότε ο client μπορεί να υποβάλλει την αίτησή του, μιλάμε για client –server μοντέλο.

1.2 Framework

1.2.1 Τι είναι framework

Όταν οι προγραμματιστές επιχειρούν να χτίσουν μια εφαρμογή έρχονται αντιμέτωποι με κάποια προβλήματα που καλούνται να λύσουν. Παρόλο που κάθε προγραμματιστής χτίζει διαφορετική εφαρμογή κάποια προβλήματα είναι παρόμοια. Για παράδειγμα, όλες οι διαδικτυακές εφαρμογές (web apps) χρειάζονται ένα σύστημα ταυτοποίησης (authentication system) που να ελέγχει τα στοιχεία των χρηστών, να συνδέονται σε μια βάση δεδομένων και να ενημερώνουν για τυχόν σφάλματα στον κώδικα με μηνύματα κτλ. Τα frameworks μας παρέχουν έτοιμες τις λύσεις αυτών των προβλημάτων καθώς και τη δυνατότητα παραμετροποίησης τους. Τα frameworks δημιουργούνται και ελέγχονται από έμπειρους προγραμματιστές. Οι λύσεις που προσφέρουν είναι οι βέλτιστες και οι πιο ασφαλής.

Συχνά παρουσιάζεται μια σύγχυση ανάμεσα στις έννοιες Framework και βιβλιοθήκη. Το framework καλεί τον κώδικα του προγραμματιστή ενώ η βιβλιοθήκη καλείται από τον κώδικα. Αντιλαμβανόμαστε καλύτερα την διαφοροποίηση τους με τον εξής παραλληλισμό: Σε ένα client – server σύστημα, ο client είναι κώδικας του προγραμματιστή και ο server η βιβλιοθήκη, ο client είναι αυτός που στέλνει τα αιτήματα και ο server επιστρέφει την απάντηση. Θέτοντας τον κώδικα του προγραμματιστή ως client και τον κώδικα του

framework ως server παρατηρείται μια αλλαγή, συγκεκριμένα η αντιστροφή της ροής του ελέγχου (inversion of control). Σε αυτήν την περίπτωση, ο κώδικας του framework (server) στέλνει αιτήματα στον κώδικα του προγραμματιστή (client) και ο κώδικας του προγραμματιστή στέλνει τις απαντήσεις. Καταλήγουμε λοιπόν στο συμπέρασμα πως η πραγματική διαφορά μεταξύ τους είναι το γεγονός πως το Framework διαχειρίζεται την ροή ελέγχου του προγράμματος ενώ η βιβλιοθήκη όχι.

1.2.2 Τα οφέλη των frameworks

Η δημιουργία λογισμικού είναι μια σύνθετη διεργασία που περιλαμβάνει διαφορετικά στάδια όπως είναι η σχεδίαση, η κωδικοποίηση και ο έλεγχος του προγράμματος. Το κάθε στάδιο είναι μια αρκετά σύνθετη εργασία. Για παράδειγμα, η κωδικοποίηση του προγράμματος, περιλαμβάνει την επιλογή των κατάλληλων τύπων δεδομένων, τις δηλώσεις των μεταβλητών, τη σύνταξη των εντολών, τη διαχείριση των εξαιρέσεων κ.λ.π..

Πλεονεκτήματα των frameworks :

- Παραγωγή ασφαλέστερου κώδικα (χωρίς κενά ασφαλείας).
- Έτοιμες λύσεις σε συνηθισμένα προβλήματα.
- Αποφυγή επανάληψης κώδικα.
- Λιγότερα λάθη στη σύνταξη.
- Εύκολος έλεγχος σφαλμάτων ακόμα κι από προγραμματιστές που δεν τον δημιούργησαν.
- Ένα framework εξελίσσεται και βελτιώνεται συνεχώς.
- Ο χρόνος για την δημιουργία ενός προγράμματος μειώνεται αισθητά.

1.3 Frameworks for server – client side scripting

1.3.1 Server – client scripting

Το server side scripting είναι μία τεχνική στην οποία με τη χρήση κώδικα στον server μπορούμε να επεξεργαστούμε και να παραμετροποιήσουμε τις απαντήσεις που στέλνει στα αιτήματα διάφορων clients. Επεμβαίνουμε δηλαδή στην διεπαφή του client με τον server, για παράδειγμα μπορούμε να χρησιμοποιήσουμε κώδικα ο οποίος ανάλογα με τα στοιχεία του κάθε πελάτη, στέλνει τις αντίστοιχες απαντήσεις. Κάποιες από τις γλώσσες που χρησιμοποιούνται για αυτόν το σκοπό είναι : Python, Java, Go, PHP, Perl, Ruby και άλλες.

Το client side scripting αλλάζει τις συμπεριφορές της ιστοσελίδας ανάλογα με τις ενέργειες του mouse (ποντικιού) ή πληκτρολογίου ή σε συμβάντα που πραγματοποιούνται σε συγκεκριμένη χρονική στιγμή. Σε αυτήν την περίπτωση, η δυναμική συμπεριφορά εμφανίζεται στο user interface. Το περιεχόμενο του client side scripting δημιουργείται στο τοπικό σύστημα του χρήστη πελάτη.

1.3.2 Frameworks for server side scripting

1.3.2.1 Django

Το Djando είναι ένα framework ιστού (web) σε γλώσσα Python που επιτρέπει την ταχεία ανάπτυξη ασφαλών και συντηρήσιμων ιστοσελίδων.

Το Django δημιουργήθηκε από έμπειρους προγραμματιστές με σκοπό να αναλάβει τα κουραστικά κομμάτια της δημιουργίας μιας διαδικτυακής εφαρμογής έτσι ώστε οι προγραμματιστές να επικεντρωθούν στη δημιουργία της εφαρμογής τους χωρίς να χρειάζεται να “επανεφεύρουν τον τροχό”. Είναι ένα δωρεάν framework ανοιχτού κώδικα με μία αναπτυσσόμενη, ενεργή κοινότητα και με ένα πολύ λεπτομερειακό και καλογραμμένο εγχειρίδιο χρήσεως. Το Django παρέχει σχεδόν όλα όσα μπορεί να χρειαστούν οι προγραμματιστές για την δημιουργία της εφαρμογής τους , όλα λειτουργούν σε συνεργασία και ακολουθούν με συνέπεια τις αρχές σχεδίασης. Το Django μπορεί να χρησιμοποιηθεί για τη δημιουργία οποιουδήποτε τύπου ιστοσελίδας από ιστοσελίδες περιεχομένου και πληροφοριών έως σελίδες κοινωνικής δικτύωσης και ειδήσεων. Το Django συνεργάζεται με όλα τα client-side frameworks και μπορεί να παραδώσει πληροφορίες με την εκάστοτε επιθυμητή μορφοποίηση (HTML, RSS feeds, JSON, XML και άλλα).

Το Django έχει σχεδιαστεί έτσι ώστε να αποφευχθούν τα κενά ασφαλείας προσφέροντάς έτοιμα εργαλεία, όπως διάφορες βιβλιοθήκες. Βοηθάει στην ασφάλεια του προγράμματος γενικότερα, αφού κρύβει τον πηγαίο κώδικα στον server και μετά δημιουργεί δυναμικές ιστοσελίδες. Αυτό σημαίνει ότι ο κώδικας στον server μεταφράζεται σε HTML και CSS έτσι ώστε όταν ο χρήστης προσπαθήσει να τον εντοπίσει να βρει τον κώδικα ως μια ερμηνεία που υπέθεσε ο browser του και όχι τον πραγματικό κώδικα που γράψαμε.

Το Django διασπά την αρχιτεκτονική του σε κομμάτια που δεν επικοινωνούν μεταξύ τους. Κάθε κομμάτι είναι ανεξάρτητο και για αυτό μπορεί να αντικατασταθεί ή να αλλάξει. Προσθαφαιρώντας κομμάτια μπορούμε να κλιμακώσουμε το μέγεθος της εφαρμογής ανάλογα τις ανάγκες μας. Ο κώδικας Django γράφεται χρησιμοποιώντας αρχές σχεδίασης και μοτίβα που ενθαρρύνουν τη δημιουργία διατηρήσιμου και επαναχρησιμοποιήσιμου κώδικα. Συγκεκριμένα, χρησιμοποιεί την αρχή «Μην επαναλαμβάνετε τον εαυτό σας» (DRY), ώστε να μην υπάρχει περιττή επανάληψη, μειώνοντας την ποσότητα του κώδικα. Το Django προωθεί επίσης την ομαδοποίηση σχετικών λειτουργιών σε επαναχρησιμοποιήσιμες "εφαρμογές".

Άλλο ένα θετικό στο Django είναι ότι είναι γραμμένο σε Python, μια γλώσσα που είναι εύκολη στην εκμάθηση και στην χρήση της. Είναι επίσης παραγωγική καθώς μπορείς να δημιουργήσεις περισσότερες λειτουργίες γράφοντας λιγότερες γραμμές κώδικα. Επιπλέον, λόγο της Python το Django μπορεί να χρησιμοποιηθεί σε όλα τα λειτουργικά συστήματα (windows , Linux , Mac). Σημαντικό να αναφερθεί το έτοιμο και εγκατεστημένο ORM που μας δίνει το Django ώστε να μας γλιτώσει χρόνο από backend προγραμματισμό για να επικεντρωθούμε στο user interface. Τέλος θα ήθελα να κάνω μια αναφορά στο έτοιμο πάνελ διαχειριστή (admin panel) που δίνει το Django. Μπορεί να χρησιμοποιηθεί σαν ένα εργαλείο διαχείρισης περιεχομένου το οποίο είναι τόσο απλό και κατανοητό στη χρήση που μπορούν να το χρησιμοποιήσουν άνθρωποι χωρίς της απαραίτητες γνώσεις στο αντικείμενο και να προσθαφαιρέσουν περιεχόμενο παραμετροποιόντας το όπως επιθυμούν.

Το Django αναπτύχθηκε αρχικά από το 2003 έως το 2005 από μια ομάδα που ήταν υπεύθυνη για τη δημιουργία και τη συντήρηση ιστοσελίδων για εφημερίδες. Μετά τη δημιουργία

πολλών ιστοσελίδων, η ομάδα άρχισε να αναλύει και να επαναχρησιμοποιεί πολλά κοινά μοτίβα κώδικα και σχεδίασης. Αυτός ο κοινός κώδικας εξελίχθηκε σε ένα γενικού πλαισίου διαδικτυακό framework ανοιχτού κωδικά τον Ιούλιο του 2005. Το Django συνέχισε να μεγαλώνει και να βελτιώνεται, από την πρώτη του έκδοση (1.0) τον Σεπτέμβριο του 2008 έως την πρόσφατα κυκλοφορούμενη έκδοση 3.1 (2020).

Κάθε έκδοση έχει προσθέσει νέες λειτουργίες και διορθώσεις σφαλμάτων, που κυμαίνονται από υποστήριξη για νέους τύπους βάσεων δεδομένων, μηχανές προτύπων (template engines) και προσωρινή αποθήκευση (cashing), έως την προσθήκη "γενικών" λειτουργιών προβολής (views) και κλάσεων (που μειώνουν τον αριθμό κώδικα που πρέπει να γράψουν οι προγραμματιστές μια σειρά από εργασίες προγραμματισμού).

Το Django είναι πλέον ένα ακμάζον, έργο ανοιχτού κώδικα, με πολλούς χιλιάδες χρήστες και προγραμματιστές που συνεισφέρουν στην εξέλιξή του. Αν και εξακολουθεί να έχει ορισμένα χαρακτηριστικά που προέρχονται από τον αρχικό σχεδιασμό του, το Django έχει εξελιχθεί σε ένα ευέλικτο πλαίσιο που είναι ικανό να αναπτύξει οποιοδήποτε τύπο ιστότοπου.

Το Django είναι περίπου opinionated (έχει άποψη σχετικά με τον «σωστό τρόπο» για τη διαχείριση κάθε συγκεκριμένης εργασίας), και ως εκ τούτου προσφέρει το "καλύτερο και των δύο περιπτώσεων". Παρέχει ένα σύνολο δομικών στοιχείων (components) για την διαχείριση των περισσότερων εργασιών ανάπτυξης ιστοσελίδων και έναν (ή δύο) προτιμώμενους τρόπους χρήσης τους. Ωστόσο, η αποσυνδεδεμένη αρχιτεκτονική του Django παρέχει μια σειρά από διαφορετικές επιλογές, οι οποίες παραμετροποιούνται σύμφωνα με τις ανάγκες της εφαρμογής.

1.3.2.1.1 Το μοντέλο MVC (model view controller):

Το Django καθώς και άλλα δημοφιλή frameworks κάνουν χρήση του μοντέλου MVC. Σε αυτό το σημείο είναι σημαντικό να αναφέρουμε λίγα λόγια για το μοντέλο αυτό.

Το μοντέλο MVC είναι μια τεχνική σχεδίασης (αρχιτεκτονική) λογισμικού, η χρήση του οποίου γίνεται συχνά για την δημιουργία της διεπαφής του προγράμματος με τον χρήστη (interface). Το μοντέλο αυτό χωρίζει τη λειτουργία του προγράμματος σε τρία μέρη τα οποία συνδέονται μεταξύ τους. Με αυτόν τον τρόπο διαχωρίζονται οι εσωτερικές παρουσιάσεις των πληροφοριών από τους τρόπους παρουσίασης τους και αποδοχής τους από τον χρήστη.

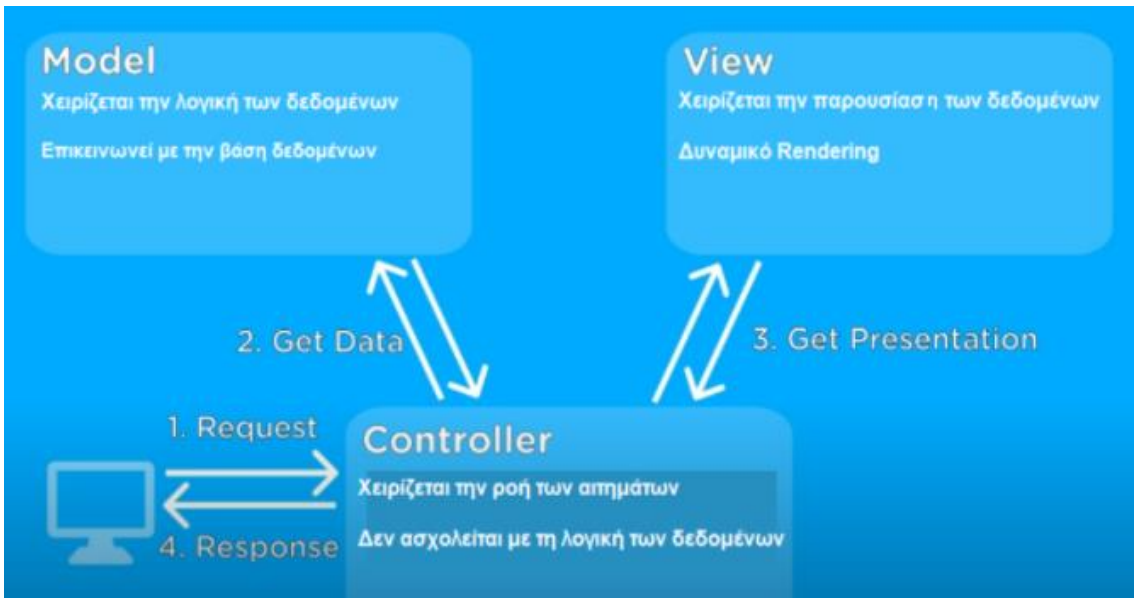
Τα τρία αυτά κομμάτια είναι το μοντέλο (model), η παρουσίαση (view) και ο ελεγκτής (controller):

- Model: Μοντελοποιεί τα δεδομένα με αξιόπιστο τρόπο ανάλογα τις εντολές του controller.
- View: Σύμφωνα με τις εντολές του χρήστη, παρουσιάζει τα δεδομένα με κατανοητό τρόπο.
- Controller: Δέχεται εντολές από τον χρήστη. Επικοινωνεί με το model στέλνοντας του εντολές με στόχο την παροχή δεδομένων. Πληροφορεί το view να αναβαθμίσει το interface.

Εξηγώντας την παρακάτω εικόνα : Ο χρήστης στέλνει μια εντολή η οποία λαμβάνεται από τον controller. Ο controller στέλνει εντολή στο model. Το model έχει πρόσβαση στα δεδομένα της βάσης, και επιστρέφει τα ζητούμενα δεδομένα πίσω στον controller. Ο

controller μεταφέρει τα δεδομένα στο view. Το view ταξινομεί και μορφοποιεί τα δεδομένα με τρόπο κατανοητό από τον χρήστη και τα επιστρέφει στον controller. Τέλος , ο controller ενημερώνει τον χρήστη. Σημαντικό να αναφερθεί ότι το model με το view δεν έχουν καμιά επικοινωνία μεταξύ τους παρά μόνο μέσω του controller.

Η κυκλοφορία των framework Django (Ιούλιος 2005, για την Python) και Rails (Δεκέμβριος 2005, για την Ruby), είχαν μεγάλη επίδραση στην ταχεία αύξηση της δημοτικότητας του MVC έξω από το παραδοσιακό επιχειρηματικό περιβάλλον στο οποίο υπήρξε από καιρό δημοφιλή. Τα MVC web frameworks διαθέτουν πλέον μεγαλύτερα μερίδια αγοράς σε σχέση με τα frameworks που δεν χρησιμοποιούν MVC.



1.3.2.2 Express

Ο Express είναι το πιο δημοφιλές framework για το Node.js. Το Node.js είναι ένα περιβάλλον που μας επιτρέπει τη χρήση javascript για server side scripting. Ο Express κυκλοφόρησε τον Νοέμβρη του 2010 και αυτήν την στιγμή βρίσκεται στην έκδοση 4.17.1. Ο Express είναι unopinionated framework, μπορούμε να χρησιμοποιήσουμε οποιοδήποτε middleware (αν βέβαια είναι συμβατό) στην αλυσίδα ελέγχου των αιτημάτων (requests), με όποια σειρά θέλουμε. Η εφαρμογή μπορεί να ταξινομηθεί σε πολλούς φακέλους (files) ή σε έναν, υπάρχουν πολλές επιλογές.

Ενώ ο Express είναι ένα μινιμαλιστικό framework από μόνο του διάφοροι προγραμματιστές δημιουργούν συμβατά middleware για την αντιμετώπιση των προβλημάτων που μπορεί να έρθει αντιμέτωπος ένας προγραμματιστής. Ο Express παρέχει μεθόδους για να καθοριστεί ποια λειτουργία καλείται για ένα συγκεκριμένο αίτημα HTTP (GET, POST, SET, κλπ) και διεύθυνση URL («Διαδρομή»), τις μεθόδους για να καθορίσετε τη μηχανή πρότυπων («view») που θα χρησιμοποιηθεί, την τοποθεσία των αρχείων προτύπων (template) και ποιο πρότυπο θα χρησιμοποιηθεί για την απόδοση μιας απάντησης. Μπορεί να χρησιμοποιηθεί οποιοσδήποτε μηχανισμό βάσης δεδομένων που υποστηρίζεται από το Node.js (το Express δεν καθορίζει καμία συμπεριφορά που σχετίζεται με τη βάση δεδομένων).

1.3.2.3 Laravel

Το Laravel είναι ένα framework ανοιχτού κώδικα γραμμένο στην PHP που χρησιμοποιείται για server side scripting. Με αυτό μπορούμε να δημιουργήσουμε διαδικτυακές εφαρμογές γρήγορα και εύκολα χάρη στα προεγκατεστημένα χαρακτηριστικά (λειτουργίες - εργαλεία) που έχει. Αυτά τα εργαλεία είναι ο λόγος που κάνει το Laravel τόσο ευρέος γνωστό στους διαδικτυακούς προγραμματιστές :

- Ένα σύστημα συσκευασίας σε κομμάτια, με διαχειριστή που είναι υπεύθυνος για τα προγράμματα που είναι απαραίτητα για την εκτέλεση του προγράμματος.
- Ένα πλήρες σύστημα ελέγχου ταυτότητας.
- Αντικειμενοστραφή αντιστοίχιση (mapping). Το Eloquent ORM που περιλαμβάνεται στο Laravel παρουσιάζει την βάση δεδομένων σαν έναν πίνακα κλάσεων για ευκολότερη πρόσβαση και τροποποίηση δεδομένων.
- Γραμμή εντολών που συνοδεύεται από δεκάδες προεγκατεστημένες εντολές (Artisan).
- Αυτόματος έλεγχος.
- Ένα φορητό, εικονικό περιβάλλον ανάπτυξης. Το Homestead παρέχει στους προγραμματιστές όλα τα απαραίτητα εργαλεία για την ανάπτυξη του Laravel.

1.3.2.4 Ruby on Rails

Το Ruby on Rails , συχνά Rails ή RoR, είναι ένα πλαίσιο ανάπτυξης λογισμικού Ιστού ανοιχτού κώδικα για τη γλώσσα προγραμματισμού Ruby. Προορίζεται για χρήση συνδυαστικά με ευέλικτες μεθοδολογίες ανάπτυξης (agile development methodologies), οι οποίες χρησιμοποιούνται από τους προγραμματιστές Ιστού για ταχεία ανάπτυξη λογισμικού (rapid application development).

Το Ruby on Rails είναι χωρισμένο σε διάφορα πακέτα:

- ActiveRecord (ένα σύστημα αντικειμενοστραφούς - σχεσιακής αντιστοίχισης (object-relational mapping) για την πρόσβαση σε βάσεις δεδομένων),
- ActiveResource (παρέχει web services)
- ActionPack
- ActiveSupport
- ActionMailer.

Πριν την έκδοση 2.0, το Rails περιλάμβανε και το πακέτο Action Web Service που τώρα αντικαθίσταται από το Active Resource. Εκτός από τα βασικά πακέτα, οι προγραμματιστές μπορούν να δημιουργήσουν plugins για να επεκτείνουν τα υπάρχοντα πακέτα. Το Ruby on Rails προήλθε από τη δουλειά του David Heinemeier Hansson στο Basecamp, ένα εργαλείο διαχείρισης project από την εταιρεία 37signals (η οποία τώρα είναι εταιρεία ανάπτυξης λογισμικού Ιστού). Ο David Hansson αρχικά κυκλοφόρησε το Rails σαν ανοιχτό κώδικα τον

Ιούλιο του 2004 αλλά δεν επέτρεπε όμως σε άλλους προγραμματιστές να συνεισφέρουν κώδικα στο εγχείρημα, μέχρι το Φεβρουάριο του 2005. Ο Αύγουστος του 2006, υπήρξε κομβικό σημείο για το Rails, όταν η Apple ανακοίνωσε ότι θα κυκλοφορούσε το Ruby on Rails μαζί με το Mac OS X v10.5 "Leopard", το οποίο κυκλοφόρησε τον Οκτώβριο του 2007. Η έκδοση 2.3 του Rails κυκλοφόρησε στις 15 Μαρτίου του 2009.

Βασικά νέα χαρακτηριστικά του Rails:

- Τα πρότυπα, τα οποία επιτρέπουν στον προγραμματιστή να δημιουργεί τον σκελετό μιας εφαρμογής με ειδικά gems και ρυθμίσεις (configurations).
- Οι μηχανές, οι οποίες επιτρέπουν τη χρήση τμημάτων εφαρμογών σε άλλες εφαρμογές, συμπεριλαμβανομένων των χαρακτηριστικών "routes", "view paths" και "models".
- Η Rack και το Metal interface (περιβάλλον) επιτρέπουν στον προγραμματιστή να γράφει βελτιστοποιημένα κομμάτια κώδικα που μπορούν να δρομολογούνται (route) σε σχέση με την ActionController.

1.3.3 Frameworks για client side scripting

1.3.3.1 React

Το React είναι ένα ελαφρύ (lightweight) framework γραμμένο στην Javascript και χρησιμοποιείται για client side scripting . Είναι framework τύπου ανοιχτού κώδικα (open source) που δημιουργήθηκε από την ομάδα του Facebook και κυκλοφόρησε 29 Μαΐου του 2013 .

Από τότε το React έχει γίνει πολύ διάσιμο και χρησιμοποιείται από πολλές μεγάλες εταιρίες , όπως Netflix, Airbnb, Facebook και New York Times όπου συνεισφέρουν στην ανάπτυξη του React χτίζοντας βιβλιοθήκες (libraries).

Με ιδιαίτερη ευκολία στην εκμάθηση και την χρήση του, το React μας επιτρέπει να αφιερώσουμε περισσότερο χρόνο πάνω στο πρόγραμμα που δημιουργούμε γλιτώνοντας μας χρόνο από την εκμάθηση του framework.

Οι εφαρμογές που είναι γραμμένες με React αποτελούνται από πολλά διακριτά μέρη που ονομάζονται components. Κάθε κομμάτι του user interface και όχι μόνο, είναι ένα ή περισσότερα ενωμένα ως ένα καινούργιο μεγαλύτερο component. Το κάθε component διαχειρίζεται το δικό του state, έχει τη δική του λειτουργία και μπορεί να επαναχρησιμοποιηθεί μειώνοντας τον χρόνο κατασκευής του προγράμματος.

Με αυτόν τον τρόπο τα προγράμματα γίνονται πιο εύκολα στην:

- Κατανόηση της λειτουργίας τους από οποιονδήποτε προγραμματιστή.
- Συντήρηση τους.

- Αποσφαλμάτωση του κώδικα τους.
- Κλιμάκωσή τους.

Το State ενός component είναι μια μεταβλητή που περιέχει διάφορες πληροφορίες για την κατάσταση του component. Για να μεταβάλουμε το state ενός component καλούμε μια μέθοδο την setState. Σε κάθε αλλαγή του state, το component ξαναφορτώνεται αυτόματα στην εφαρμογή, καθιστώντας το χρήσιμο για την δημιουργία διαδραστικών εφαρμογών

Το React χρησιμοποιεί μονής κατεύθυνσης ροή πληροφορίας (one way data flow) για περισσότερο έλεγχο ανάμεσα στο state και στο model της εφαρμογής. Με αυτόν τον τρόπο η αρχιτεκτονική της εφαρμογής γίνεται πιο κατανοητή.

Ένα επιπλέον βασικό χαρακτηριστικό του React είναι το JSX. Το JSX είναι μια γλώσσα με στατικά διατυπωμένο συντακτικό, προέκταση της JavaScript παρόμοια με αντικειμενοστραφή γλώσσα σχεδιασμένη να τρέχει σε μοντέρνα προγράμματα περιήγησης (browsers). Το JSX επιτρέπει να καθοριστούν τα στοιχεία του D.O.M (document object model) πριν την δημιουργία των components μέσα στα αρχεία JavaScript. Με αυτόν τον τρόπο η λογική και ο κώδικας για την εμφάνιση του component, θα βρίσκονται στο ίδιο μέρος.

Συνοψίζοντας παρακάτω βρίσκεται μια λίστα με θετικά και αρνητικά χαρακτηριστικά του React.

Θετικά:

- Αρχιτεκτονική προσέγγιση CBA (Component Based Architecture), της οποία τα θετικά έχουν αναγραφεί παραπάνω.
- Η χρήση του εικονικού DOM, βοηθά στην βελτίωση της εμπειρίας που έχει ο χρήστης με την εφαρμογή (user experience), ενώ παράλληλα επιταχύνει την εργασία των προγραμματιστών.
- Χρήση μονόδρομης ροής πληροφορίας που σημαίνει πως οι αλλαγές στα components δεν επηρεάζουν τα γονεϊκά (πιο ψηλά) components.
- Redux: ένα βολικό δοχείο για αποθήκευση των components state.
- Hooks: Ένα εργαλείο που προστέθηκε στην έκδοση 16.8. Η πιο σημαντική λειτουργία τους είναι ότι επιτρέπουν τον διαμοιρασμό της λογικής state στα components.
- Μια τεράστια βιβλιοθήκη ανοιχτού κώδικα που αναπτύσσεται συνεχώς και είναι ανοιχτή στην κοινότητα του React.
- Παροχή πολλών εργαλείων για React, Redux.

Αρνητικά:

- Το JSX είναι μια αμφιλεγόμενη περίπτωση. Υπάρχουν προγραμματιστές που είναι ευχαριστημένοι με το JSX και άλλοι που δυσαρεστούνται διότι το θεωρούν δύσκολο και πολύπλοκο.

- Παρουσιάζεται μια συνεχής αλλαγή των πραγμάτων και του περιβάλλοντος, σε βαθμό που οι προγραμματιστές καλούνται να μαθαίνουν νέους τρόπους λειτουργίας και να προσαρμόζονται αρκετά συχνά στα καινούργια δεδομένα.

1.3.3.2 Vue.js

Το Vue (προφέρεται όπως το view), είναι ένα προοδευτικό framework ανοιχτού κώδικα γραμμένο σε JavaScript. Χρησιμοποιείται για την δημιουργία εφαρμογών μονής σελίδας (single page applications) και για διαδικτυακά interfaces. Με το vue μπορούμε να δημιουργήσουμε εφαρμογές για υπολογιστές αλλά και για κινητές συσκευές.

Το vue ξεκίνησε να δημιουργείται το 2013 από τον Evan You, ενώ εργαζόταν στην Google και κυκλοφόρησε επίσημα το 2014. Στο vue συνεισέφεραν κυρίως άτομα από την κοινότητα του και χορηγήθηκε από το Patreon (site που ο καθένας έχει τη δυνατότητα να χορηγήσει σε διάφορα project ό,τι μπορεί). Το vue έχει υιοθετηθεί πια σαν ένα εργαλείο για front-end από μεγάλες εταιρείες, όπως Adobe, Alibaba, Gitlab, και Xiaomi.

Το vue έχει ένα αξιοσημείωτο πακέτο με εργαλεία που ο κάθε προγραμματιστής θα εκτιμούσε.

Μερικά από αυτά είναι :

• Εργαλεία ανάπτυξης του περιηγητή (browser) για αποσφαλμάτωση εφαρμογών φτιαγμένα με vue.

• Επίσημο CLI (command line interface) για εγκατάσταση των βιβλιοθηκών του vue αλλά και για εγκατάσταση επιπρόσθετων βοηθητικών εφαρμογών.

• Εργαλεία για δρομολόγηση (Routing) και χαρτογράφηση (mapping) .

• Επίσημος φορτωτής (loader) για διαδικτυακά πακέτα (webpacks).

Υπάρχουν θετικά και αρνητικά χαρακτηριστικά του Vue. Αναλυτικότερα:

Θετικά:

- Χρήση της αρχιτεκτονικής τεχνικής CBA (εξηγείται σελ.7 στο React).
- Διαδραστική δέσμευση δεδομένων διπλής κατεύθυνσης. Πρόκειται για μια σύνδεση μεταξύ των αναβαθμίσεων των δεδομένων του model και του view. Αυτή η σύνδεση καθιστά την αναβάθμιση πιο εύκολη και ιδανική για εφαρμογές που χρειάζονται αναβαθμίσεις σε πραγματικό χρόνο (real time). Να σημειωθεί, ότι η αναβάθμιση δημιουργείται εύκολα και γίνεται περισσότερο κατανοητή από τον προγραμματιστή.
- Το vue χρησιμοποιεί εικονικό DOM. Παρουσιάζεται σαν αντίγραφο του κανονικού DOM που καταλαβαίνει πιο κομμάτι πρέπει να ανανεώσει χωρίς να χρειάζεται να

ανανεωθεί όλη την σελίδα. Με αυτήν την τεχνική ο χρόνος φόρτωσης της ιστοσελίδας μικραίνει και βελτιώνεται η επίδοση της εφαρμογής.

- Ευελιξία και συμβατότητα. Αυτό αποδίδεται στο ότι το vue εξαρτάται μόνο από την JavaScript και δεν χρειάζεται κανένα άλλο εργαλείο για να δουλέψει. Έτσι μπορούμε να το ενσωματώσουμε σε υπάρχουσες εφαρμογές.
- Προσφέρει ένα πολύ δυνατό πακέτο εργαλείων.
- Το συμπιεσμένο αρχείο που περιέχει το framework, είναι μόλις 18 KB καθιστώντας το γρήγορο στο κατέβασμα αλλά και στην εγκατάστασή του.
- Εύκολο στην εκμάθηση. Χρειάζεται βασικές γνώσεις HTML, CSS και JavaScript.
- Ευανάγνωστο και καλογραμμένο εγχειρίδιο (documentation).
- Μια υποστηρικτική κοινότητα που συνέχεια βελτιώνεται και εξελίσσεται.

Αρνητικά:

- Η ευελιξία του veu είναι επικίνδυνη, καθώς η ομάδα μπορεί να διχαστεί λόγω των πολυποίκιλων προσεγγίσεων και τεχνικών. Δημιουργούνται έτσι εσωτερικές διαμάχες και διαφωνίες.
- Υπάρχει έλλειψη σε έμπειρους προγραμματιστές καθώς το vue είναι μια νέα τεχνολογία.
- Σε σύγκριση με React και Angular υπάρχει μια έλλειψη σε plug-ins.

1.3.3.3 Angular

Το Angular είναι ένα ανοιχτού κώδικα framework γραμμένο σε TypeScript, υποστηριζόμενο από την Google με στόχο την δημιουργία των user interface (front end ή client side scripting).

Το 2009 δυο μηχανικοί της Google ο Misko Hevery και ο Adam Abrons, δημιούργησαν ένα framework γνωστό με το όνομα AngularJs. Το AngularJs κυκλοφόρησε το 2010 με κύριο χαρακτηριστικό την μετατροπή ενός HTML έγγραφο, σε δυναμικό περιεχόμενο.

Τον Σεπτέμβρη του 2016 η Google κυκλοφόρησε το Angular 2. Ήταν μια έκδοση με πολλές αλλαγές οι οποίες είχαν στόχο το framework να ανταπεξέλθει στις απαιτήσεις των καιρών. Οι αλλαγές ήταν τόσο μεγάλες και τόσες πολλές που δεν μπορούσες απλά να κάνεις αναβάθμιση για να πας στην επόμενη έκδοση.

Αξιοσημείωτη είναι η καινούργια έκδοση της Angular 2+. Η Angular ανήκει στον σωρό των εφαρμογών ονόματι:

MEAN

- M – MongoDB, μη σχεσιακή βάση δεδομένων

- E – Express, (έχει περιγραφεί στο 1.3.2.2 σελ.5)
- A – Angular
- N – Node.js JavaScript περιβάλλον

Τα βασικά εργαλεία της Angular είναι :

- RxJS: Μια διαδραστική βιβλιοθήκη που στόχος της είναι ο χειρισμός ασύγχρονων δεδομένων με πολλαπλά event. Πιο απλά, επιτρέπει στους προγραμματιστές να στήσουν πολλά κανάλια μεταφοράς δεδομένων για να ελαττώσουν την κατανάλωση των πόρων.
- Angular CLI: Μια γραμμή εντολών για δημιουργία project, πρόσθεση αρχείων, updates (αναβαθμίσεις), αποσφαλμάτωση και ανάπτυξη.

Τα θετικά και αρνητικά στοιχεία της Angular:

Θετικά:

- Χρήση της αρχιτεκτονικής τεχνικής CBA (όπου εξηγείται παραπάνω στο React σελ.7).
- RxJS (έχει αναφερθεί παραπάνω, σελ. 10).
- Υψηλές επιδόσεις
- Angular Material: Βιβλιοθήκη με έτοιμα components από φόρμες ελέγχου, στοιχεία πλοήγησης, κουμπιά, pop ups και πολλά άλλα.
- Λόγω του ότι το Angular υπάρχει εδώ και αρκετά χρόνια υπάρχουν πολλά πακέτα, plug-ins και προγραμματιστικά εργαλεία από την κοινότητα.
- Από την έκδοση 6 και μετά το CLI της Angular με την εντολή NG update <package> ελέγχει όλα τα πακέτα της εφαρμογής και προτείνει αναβαθμίσεις για όποια υποπρογράμματα τις χρειάζονται.
- Angular elements: Δυνατότητα μεταφοράς Angular components σε άλλα περιβάλλοντα.

Αρνητικά:

- Η κοινότητα του Angular είναι χωρισμένη σε AngularJs και σε Angular2+.
- Χρειάζεται αρκετός χρόνος μέχρι να εκσυγχρονιστούν (προσαρμοστούν) οι εφαρμογές που φτιάχτηκαν σε AngularJS στην Angular2+.
- Το Angular είναι μακροσκελές και περίπλοκο framework.
- Απότομη καμπύλη εκμάθησης του framework.

2 ΚΕΦΑΛΑΙΟ 2ο : Διαδικτυακή εφαρμογή - Storage handler

2.1 Τι είναι διαδικτυακή εφαρμογή

Διαδικτυακή εφαρμογή (web application ή web app) ονομάζεται οποιαδήποτε εφαρμογή είναι διαθέσιμη στους χρήστες της μέσω του διαδικτύου (internet) ή του ενδοδικτύου (Intranet) μιας εταιρίας. Το μόνο απαραίτητο εργαλείο για τον χρήστη της εφαρμογής αυτής, είναι ο περιηγητής του. Για να εκτελεστούν όμως σωστά οι εφαρμογές αυτές, χρειάζονται ισχυρές υπολογιστικές μηχανές όπου ρόλος τους είναι η εξυπηρέτηση και η παροχή υπηρεσιών σε περισσότερους του ενός χρήστες.

Όντας, επομένως μια διαδικτυακή εφαρμογή προσφέρει τα εξής πλεονεκτήματα:

- Άμεση πρόσβαση όλων των χρηστών από οποιαδήποτε υπολογιστική ή άλλη συσκευή έχει ίντερνετ χωρίς την εγκατάσταση κάποιας εφαρμογής. Το μόνο που χρειάζεται είναι ο περιηγητής διαδικτύου ο οποίος είναι προεγκατεστημένος σε όλα τα λειτουργικά συστήματα. Με αυτόν τον τρόπο θα έχουν πρόσβαση όλοι οι υπολογιστές, οι φορητές συσκευές, ακόμα και κινητά τηλέφωνα. Αυτό διευκολύνει ακόμα περισσότερο μεγάλες αποθήκες με πολλούς χρήστες εφόσον δεν είναι απαραίτητη η εγκατάσταση σε κάθε ένα υπολογιστή ξεχωριστά.
- Χρήση της εφαρμογής ανεξαρτήτως τοποθεσίας. Το πλεονέκτημα αυτό, παρέχει τη δυνατότητα στους χρήστες να χρησιμοποιούν την εφαρμογή από οπουδήποτε, δημιουργώντας ευελιξία και επιτρέποντάς τους να εργάζονται από απομακρυσμένες περιοχές ή ακόμα κι από την οικεία τους.
- Δεν καταναλώνουν υπολογιστικούς πόρους. Εφόσον δεν εκτελούνται σε κάθε υπολογιστή του χρήστη, δεν καταναλώνουν και πόρους από το σύστημα γι' αυτό και πρόκειται για "ελαφριές" εφαρμογές που δεν επιβαρύνουν την υπολογιστική μονάδα.
- Δεν καταλαμβάνουν χώρο. Ως επακόλουθο των προηγούμενων, δεν χρειάζονται καθόλου ή σχεδόν καθόλου χώρο στον δίσκο του χρήστη αφού το σύνολο της εφαρμογής είναι αποθηκευμένο στον εξυπηρετητή και μόνο κατά την χρήση της εφαρμογής μπορεί να υπάρχει μεταφορά δεδομένων προς την υπολογιστική μονάδα του χρήστη και μόνο στην περίπτωση που ο χρήστης το επιθυμεί.

2.2 Storage – handler: Περιγραφή

Η διαδικτυακή εφαρμογή που προσπαθούμε να δημιουργήσουμε με όνομα ¨Storage – Handler", πρόκειται ουσιαστικά για μια εφαρμογή διαχείρισης αποθήκης με ένα ηλεκτρονικό κατάστημα e-shop. Ανταποκρίνονται σε όλους όσους διαχειρίζονται μεγάλες αποθηκευτικές μονάδες. Δεν θα ήταν πλεονασμός να αναφέρουμε πως τις περισσότερες αποθήκες τις διακατέχει μια ανοργάνωτη συσσώρευση προϊόντων, όπου οι διαχειριστές τους δυσκολεύονται να ανταπεξέλθουν στις εργασιακές τους ανάγκες, με αποτέλεσμα να υπάρχουν συχνά λάθη και καθυστερήσεις. Δημιουργείται, λοιπόν η ανάγκη η διαχείριση αυτή να γίνεται με την μέγιστη δυνατή οργάνωση, συνέπεια και συνεργασία μεταξύ των μελών της. Η

εφαρμογή εξοικονομεί χρόνο και από την αυτόματη ενημέρωση του ηλεκτρονικού καταστήματος με την αυτόματη ανανέωση σε αριθμό προϊόντων.

Με αφορμή αυτή την ανάγκη που έχει δημιουργηθεί, η εφαρμογή "Storage – Handler" θα κληθεί να επιλύσει τα προβλήματα που δημιουργούνται, μέσα από μια διαδικτυακή πλατφόρμα όπου μόνο οι χρήστες της θα έχουν πρόσβαση με έλεγχο εισόδου που θα απαιτεί συγκεκριμένο όνομα αλλά και μοναδικό κωδικό που θα γνωρίζει μονάχα ο εκάστοτε χρήστης και με την διαχείριση του e-shop.

Στόχος της είναι η διευκόλυνση των εργαζομένων να διαχειρίζονται τα προϊόντα τους, να αναλαμβάνουν παραγγελίες ανάλογα με τη διαθεσιμότητά τους και να υπάρχει διαρκής επικοινωνία για τον διαμερισμό των καθηκόντων τους με ασφαλή και αποτελεσματικό τρόπο.

2.2.1 Σύντομη περιγραφή

Η εφαρμογή αποτελείται από δυο μέρη, τη διαχείριση της αποθήκης και το ηλεκτρονικό κατάστημα (e-shop). Για να δουλέψει η εφαρμογή πρέπει να γίνουν τα βασικά data entries.

Για αρχή πρέπει να χωρίσουμε τους εργαζόμενους σε ομάδες. Αφού δημιουργήσουμε τις ομάδες των εργαζομένων θα πρέπει να τις περάσουμε στην εφαρμογή και να δώσουμε τα κατάλληλα δικαιώματα που χρειάζεται η κάθε ομάδα για να φέρει εις πέρας τον στόχο της. Στη συνέχεια θα πρέπει να καταγράψουμε τον κάθε χρήστη και να περάσουμε τα στοιχεία του. Τον κάθε χρήστη θα πρέπει να τον κατηγοριοποιήσουμε σε μια ομάδα από αυτές που ήδη έχουμε δημιουργήσει. Ολοκληρώνοντας αυτές τις ενέργειες θα έχουμε υλοποιήσει την κατάσταση των εργαζομένων και τους ρόλους που χρειάζεται η αποθήκη για να λειτουργήσει. Στη συνέχεια θα πρέπει να καταγραφούν όλα τα προϊόντα της αποθήκης. Στα απαραίτητα στοιχεία των προϊόντων υπάρχουν τα πεδία προμηθευτής και κατηγορία που θα ήταν χρήσιμο να δημιουργηθούν πριν το προϊόν. Αφού καταγραφούν όλα τα προϊόντα και κατανεμηθούν στις κατάλληλες κατηγορίες και τύπους προϊόντων, μπορούμε να επιλέξουμε κάποια από αυτά για να εμφανίζονται στην κατηγορία προτεινόμενα (featured) προϊόντα στο ηλεκτρονικό κατάστημα. Έπειτα θα πρέπει να συμπληρωθούν οι απαραίτητες πληροφορίες για την λειτουργία της εφαρμογής. Αρχικά στη σελίδα settings συμπληρώνουμε τον ταχυδρομικό κώδικα της αποθήκης, το ποσό για το οποίο οι παραγγελίες του καταστήματος της αποθήκης θα χρειάζονται έγκριση και ο αριθμός χαμηλού αποθέματος (low supply) που θα εντάσσει τα προϊόντα στην κατηγορία Low supply ώστε να γνωρίζουν οι εργαζόμενοι τι παραγγελία να κάνουν στο supply order. Τέλος θα πρέπει να εισαχθούν τα στοιχεία κοστολόγησης και να συμπληρωθεί ο κατάλογος με τους ταχυδρομικούς κώδικες. Τα στοιχεία κοστολόγησης είναι τα εξής: κοστολόγηση βάση χιλιομέτρων και ογκομετρικού βάρους.

Αφού συμπληρωθούν τα παραπάνω, η εφαρμογή του ηλεκτρονικού καταστήματος θα απεικονίζει την κατάσταση της αποθήκης. Ο κάθε πελάτης έχει την δυνατότητα να εισέλθει στην σελίδα του ηλεκτρονικού καταστήματος χωρίς έλεγχο. Σε αυτή θα μπορεί να χρησιμοποιήσει την μπάρα αναζήτησης στο πάνω μέρος της εφαρμογής ή το κουμπί <<κατηγοριών>> που παρουσιάζεται με τρεις παράλληλες οριζόντιες γραμμές για να αναζητήσει το προϊόν που τον ενδιαφέρει. Όταν το βρει μπορεί να το προσθέσει στο καλάθι του επιλέγοντας την ποσότητα και για να το παραγγείλει πατάει το κουμπί <<ολοκλήρωση

παραγγελίας>>. Στη συνέχεια θα παρουσιαστεί μια φόρμα για να συμπληρώσει τα στοιχεία του και τον τρόπο επιθυμητής παραλαβής.

2.2.1 Περιγραφή λειτουργιών.

Για τις λειτουργίες της εφαρμογής θα ληφθεί υπόψιν η κάλυψη των αναγκών μιας αποθήκης και η εξυπηρέτηση των χρηστών της.
Λειτουργίες σελίδας διαχείρισης αποθήκης:

Σύστημα ελέγχου εισόδου: Στην σελίδα διαχείρισης της αποθήκης είναι σημαντικό να υπάρχει ελεγχόμενη είσοδος. Για να μπορέσει ο χρήστης να εισέλθει στην εφαρμογή θα πρέπει πρώτα να περαστεί στο σύστημα από το προσωπικό της αποθήκης. Τότε θα λάβει τους κωδικούς για να χρησιμοποιεί την εφαρμογή. Οι κωδικοί αυτοί ελέγχονται από την εφαρμογή για να επιτραπεί είσοδος στους χρήστες.

Σύστημα απόδοσης ρόλων και δικαιωμάτων: Σε κάθε επιχείρηση υπάρχουν ομάδες με κοινούς στόχους και υποχρεώσεις. Η εφαρμογή μας επιτρέπει να δημιουργήσουμε δικές μας ομάδες και να αποδώσουμε σε αυτές τα δικαιώματα που χρειάζονται για να πετύχουν τον στόχο τους.

Σύστημα καταχώρησης και επεξεργασίας προσωπικού: Η εφαρμογή επιτρέπει την καταχώρηση προσωπικού και των απαραίτητων πληροφοριών για αυτό. Το προσωπικό υπεύθυνο για αυτό εισάγει τα δεδομένα του χρήστη και τον κατατάσσει σε μια ομάδα ανάλογα με τις ευθύνες που του έχουν ανατεθεί. Τέλος το προσωπικό προμηθεύει τον νέο υπάλληλο με τους κωδικούς για είσοδο στην εφαρμογή. Όλα τα στοιχεία όλων των χρηστών (εκτός κωδικού) είναι προσβάσιμα (από τον υπεύθυνο) και επεξεργάσιμα.

Σύστημα καταχώρησης και επεξεργασίας προϊόντων: Ο ρόλος των αποθηκών είναι η διαχείριση προϊόντων. Στην εφαρμογή μπορούμε να καταχωρήσουμε προϊόντα και να αποκτούμε πρόσβαση στις πληροφορίες αυτών ανατρέχοντας στο σύστημα. Τα προϊόντα ομαδοποιούνται σε κατηγορίες και οι κατηγορίες σε τύπους. Σε κάθε σελίδα προϊόντος υπάρχουν όλες οι πληροφορίες για την καλύτερη λειτουργία της αποθήκης και διευκόλυνση του προσωπικού να φέρει εις πέρας τους στόχους του.

Σύστημα καταχώρησης και επεξεργασίας παραγγελιών: Η εφαρμογή χωρίζει τις παραγγελίες σε δύο κατηγορίες, τις παραγγελίες του ηλεκτρονικού καταστήματος και τις παραγγελίες της αποθήκης. Οι παραγγελίες του ηλεκτρονικού καταστήματος είναι οι παραγγελίες που έρχονται στην αποθήκη από πελάτες μέσο του e-shop (ηλεκτρονικού καταστήματος). Οι παραγγελίες της αποθήκης συντάσσονται από το προσωπικό υπεύθυνο για το απόθεμα προϊόντων στην αποθήκη.

Σύστημα καταχώρησης και επεξεργασίας ταχυδρομικών κωδικών: Για να μπορέσει η εφαρμογή να υπολογίσει την απόσταση (σε χιλιόμετρα) που απέχει ο πελάτης από την αποθήκη γίνεται χρήση ταχυδρομικών κωδικών. Το προσωπικό χρειάζεται να συμπληρώσει

έναν κατάλογο με ταχυδρομικούς κωδικούς και να προσθέσει σε κάθε έναν το γεωγραφικό πλάτος και μήκος της περιοχής.

Σύστημα καταχώρησης και επεξεργασίας προμηθευτών: Η εφαρμογή διαχειρίζεται και τους προμηθευτές της αποθήκης. Αφού εισάγουμε τα στοιχεία τους μπορούμε να τους επιλέγουμε για τις παραγγελίες ανεφοδιασμού της αποθήκης.

Σύστημα υπολογισμού απόστασης: Οι παραγγελίες που έρχονται από το e-shop περιέχουν τον ταχυδρομικό κώδικα του πελάτη. Η εφαρμογή ανατρέχει στον κατάλογο με τους ταχυδρομικούς κώδικες για να βρει το γεωγραφικό μήκος και πλάτος. Κάνει την αφαίρεση και βρίσκει την απόσταση σε χιλιόμετρα που απέχει η αποθήκη από τον πελάτη.

Σύστημα υπολογισμού κόστους μεταφοράς: Η εφαρμογή δίνει τη δυνατότητα στον πελάτη να του ταχυδρομηθούν τα προϊόντα έναντι κάποιου επιπλέον κόστους που ονομάζεται shipping cost. Ο υπολογισμός του shipping cost γίνεται αυτόματα συνυπολογίζοντας την απόσταση και το ογκομετρικό βάρος. Η απόσταση υπολογίζεται από το σύστημα υπολογισμού απόστασης, το ογκομετρικό βάρος υπολογίζεται από το εμβαδόν του προϊόντος δια το βάρος του.

Σύστημα προβολής προϊόντων σε χαμηλό απόθεμα: Στην σελίδα των προϊόντων υπάρχει φίλτρο με όνομα Low supply. Ενεργοποιώντας το η εφαρμογή θα φιλτράρει τα προϊόντα με βάση το απόθεμα τους στην αποθήκη και θα μας παρουσιάσει τα προϊόντα στα οποία η αποθήκη χρειάζεται ανεφοδιασμό.

Παραμετροποίηση κοστολόγησης ογκομετρικού βάρους και απόστασης: Η κοστολόγηση της απόστασης και του ογκομετρικού βάρους μπορεί να παραμετροποιηθεί σύμφωνα με τις ανάγκες της αποθήκης. Το προσωπικό μπορεί να εισάγει πίνακα με την επιθυμητή κοστολόγηση όγκου και χιλιομέτρων.

Ιστορικό ενεργειών στην εφαρμογή: Στην αρχική σελίδα της εφαρμογής διαχείρισης αποθήκης υπάρχει στα δεξιά πίνακας όπου καταγράφονται οι τελευταίες ενέργειες που έχουν γίνει στην εφαρμογή.

Σύστημα ανάθεσης παραγγελιών σε εργαζόμενους της αποθήκης: Και στις δυο κατηγορίες παραγγελιών είναι υποχρεωτικό οι παραγγελίες να ανατεθούν σε κάποιον εργαζόμενο. Στις παραγγελίες της αποθήκης η ανάθεση γίνεται άμεσα την ώρα δημιουργίας της δηλαδή. Στις παραγγελίες του ηλεκτρονικού καταστήματος η ανάθεση πρέπει να γίνει από κάποιον υπεύθυνο. Για να ενημερωθεί ο εργαζόμενος σχετικά με ποιες παραγγελίες είναι υπό την ευθύνη του αρκεί να προηγηθεί στην σελίδα των παραγγελιών και στο φίλτρο assigned to να επιλέξει το username του. Όταν ο εργαζόμενος ολοκληρώσει την παραγγελία θα μπορεί να πατήσει το κουμπί <<complete order>> για να ολοκληρωθεί η παραγγελία.

Αποστολή μνήματος ηλεκτρονικού ταχυδρομείου στον πελάτη κατά την ολοκλήρωση της παραγγελίας: Όταν ο υπεύθυνος για μια παραγγελία του ηλεκτρονικού καταστήματος ολοκληρώσει μια παραγγελία τότε θα αποστέλλεται μήνυμα ηλεκτρονικού ταχυδρομείου στον πελάτη ενημερώνοντάς τον πως η παραγγελία του έχει ολοκληρωθεί και μπορεί να την παραλάβει από το κατάστημα αν έχει επιλέξει αυτόν τον τρόπο παραλαβής ή ότι η παραγγελία του είναι καθοδόν αν έχει επιλέξει με αντικαταβολή.

Λειτουργίες σελίδας ηλεκτρονικού καταστήματος:

Πλοήγηση και παρουσίαση των προϊόντων της αποθήκης: Στο επάνω μέρος του user interface της σελίδας του e-shop υπάρχει κουμπί που με το πάτημα του παρουσιάζει τους τύπους κατηγοριών που χωρίζονται τα προϊόντα. Όταν επιλέξουμε κατηγορία τότε θα μεταφερθούμε στην σελίδα παρουσίασης των προϊόντων της κατηγορίας που επιλέχθηκε. Επιπρόσθετα δεξιά του προαναφερθέντος κουμπιού υπάρχει μπάρα αναζήτησης για πιο στοχευμένη έρευνα. Στο πάνω αριστερό μέρος βρίσκεται το λογότυπο το οποίο χρησιμεύει και ως ανακατεύθυνση στην αρχική σελίδα αν πατήσουμε κλικ πάνω του. Τέλος, ο χρήστης μπορεί να ελέγξει το καλάθι αγορών του κάνοντας κλικ στο κουμπί με σήμα ένα καλάθι στο επάνω δεξί μέρος της σελίδας.

Χώρος παρουσίασης προτεινόμενων προϊόντων: Ένα κομμάτι της αρχικής σελίδας του e-shop είναι αφιερωμένο στη παρουσίαση προτεινόμενων προϊόντων. Τα προϊόντα επιλέγοντας μέσω της σελίδας διαχείρισης.

Προβολή στοιχείων προϊόντων: Υπάρχουν δυο τρόποι προβολής ενός προϊόντος. Η συνοπτική, όπου παρουσιάζεται σε μικρό μέγεθος η φωτογραφία, η κατηγορία και το όνομα. Η περιγραφική όπου παρουσιάζονται τα παραπάνω συν της περιγραφής και τιμής πώλησης.

Λειτουργεία παραγγελιών: Μέσα από την σελίδα του ηλεκτρονικού καταστήματος μπορούν οι χρήστες να παραγγείλουν τα προϊόντα που τους ενδιαφέρουν. Στη σελίδα περιγραφικής προβολής προϊόντος υπάρχει το κουμπί <<Προσθήκη στο καλάθι>>. Πατώντας το κουμπί το προϊόν μεταφέρεται στο καλάθι αγορών του χρήστη όπου μπορεί να επισκεφτεί για να ολοκληρώσει την διαδικασία παραγγελίας. Στο καλάθι παραγγελιών ο χρήστης έχει την δυνατότητα να επιλέξει πόσα κομμάτια από τα προϊόντα που έχει διαλέξει θέλει να παραγγείλει, να αφαιρέσει προϊόντα καθώς και να μάθει τον κωδικό κάθε προϊόντος. Πατώντας το κουμπί <<Ολοκλήρωση παραγγελίας>> ο χρήστης θα πρέπει να συμπληρώσει την φόρμα που θα του παρουσιαστεί με τα απαραίτητα στοιχεία του και να πατήσει το κουμπί <<Αποστολή παραγγελίας>>. Ο χρήστης διαβάζει ένα ευχαριστήριο μήνυμα και του δύναται η δυνατότητα μεταφοράς του στην αρχική σελίδα του e-shop.

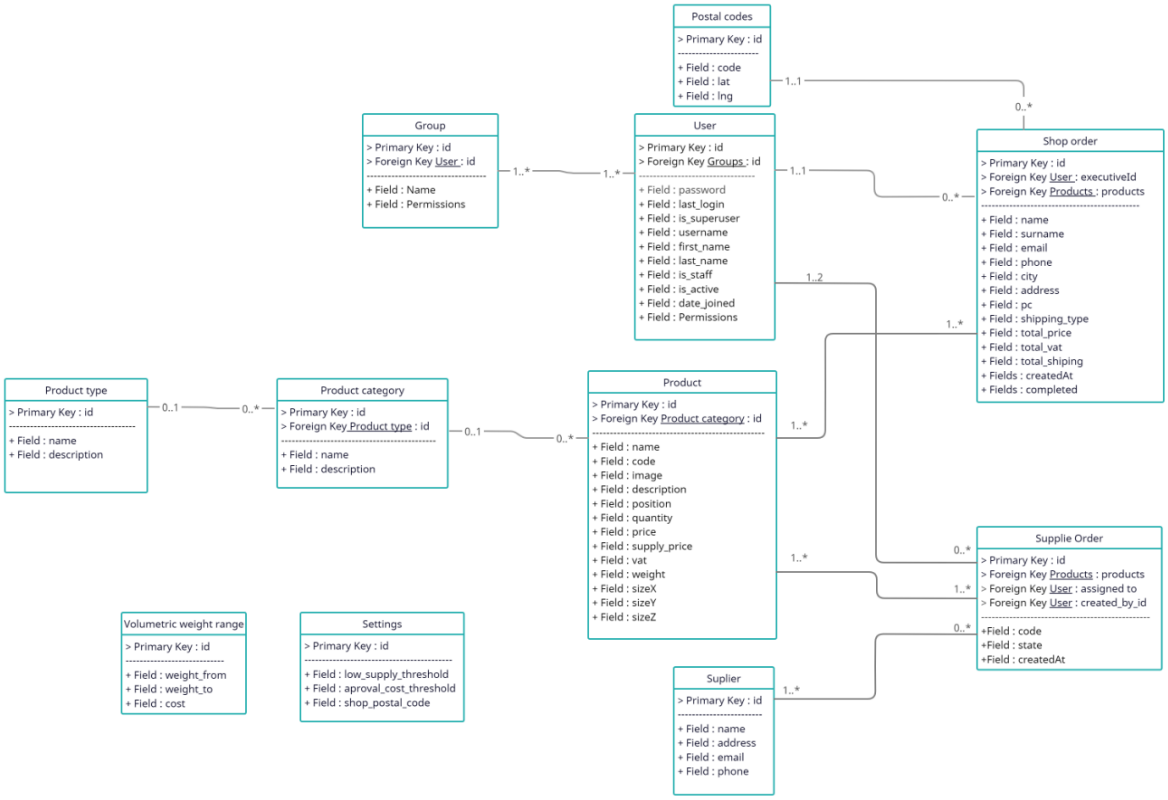
Ενημέρωση πελάτη για τα επιπλέων κόστη που μπορεί να κληθεί να πληρώσει: Στην σελίδα συμπλήρωσης των προσωπικών στοιχείων του χρήστη στο δεξί μέρος θα υπολογίζονται και θα παρουσιάζονται τα επιπλέων κόστη της παραγγελίας και θα υπάρχει και μια σύνοψη αυτής. Για αρχή ο χρήστης μπορεί να επιλέξει παραλαβή από το κατάστημα η αντικαταβολή. Αν επιλέξει με αντικαταβολή θα υπολογιστεί το χρέος απόστασης σύμφωνα με τον ταχυδρομικό κώδικα που συμπλήρωσε ο χρήστης και την τιμή χιλιομέτρου που έχει ορίσει η διοίκηση της αποθήκης στην σελίδα διαχείρισης. Στο χρέος των μεταφορικών θα συνυπολογίζεται και το βάρος των προϊόντων. Τέλος υπάρχει παρουσίαση του ποσού Φ.Π.Α και συνολικού κόστους όπου αθροίζει όλα τα παραπάνω.

Σύστημα ελέγχου της φόρμας προσωπικών στοιχείων του πελάτη. Το σύστημα ελέγχου θα έχει τη δυνατότητα να αποτρέπει τον χρήστη να καταχωρίσει ελλιπή φόρμα. Θα παρουσιάζεται ένα μήνυμα σφάλματος στην οθόνη του χρήστη που θα τον ενημερώνει για τυχόν λάθος ή κενό στη συμπλήρωση της φόρμας.

Σελιδοποίηση των προϊόντων (pagination). Ο αριθμός των προϊόντων που θα εμφανίζονται στο πλαίσιο της σελίδας είναι συγκεκριμένος. Σε περίπτωση που ξεπεραστεί αυτός ο αριθμός, τα περισσευούμενα προϊόντα δεν θα παρουσιάζονται και ο χρήστης θα έχει τη δυνατότητα πρόσβασης σε αυτά μονάχα αλλάζοντας σελίδα, πατώντας τους αριθμούς που δείχνουν πόσες σελίδες έχουν δημιουργηθεί.

2.2.2 Μορφή βάσης δεδομένων

Στην παρακάτω φωτογραφία απεικονίζεται η μορφή της βάσης δεδομένων που χρησιμοποιεί η εφαρμογή σε UM διάγραμμα.



2.2.3 Περιγραφή των μοντέλων και των πεδίων τους

Groups: Εργαζόμενοι με κοινούς στόχους στην αποθήκη βρίσκονται στο ίδιο Group (ομάδα). Ανάλογα με τον ρόλο τους στον εργασιακό κύκλο τους προσδίδονται και τα ανάλογα δικαιώματα.

Τα πεδία του Groups:

- Name: Εισαγωγή του ονόματος του Group.
- Permissions: Επιλογή των δικαιωμάτων του Group.

Users: Οι άνθρωποι που εργάζονται στην αποθήκη καθώς και όλες οι απαραίτητες πληροφορίες για αυτούς.

Τα πεδία του Users:

- Username: Το όνομα που πρέπει να χρησιμοποιήσει ο χρήστης για να εισέλθει στην εφαρμογή.
- Password: Ο κωδικός που πρέπει να χρησιμοποιήσει ο χρήστης για να εισέλθει στην εφαρμογή.
- First name: Το μικρό όνομα του χρήστη.
- Last name: Το επώνυμο του χρήστη.
- Email address: Η διεύθυνση του προσωπικού ηλεκτρονικού ταχυδρομείου του χρήστη.
- Active: Επιλέξτε αν ο χρήστης είναι ενεργός (συνεχίζει να εργάζεται στην αποθήκη).
- Staff status: Επιλέξτε αν επιθυμείτε ο χρήστης να μπορεί να εισέλθει στην σελίδα του προσωπικού της αποθήκης.
- Superuser status: Επιλέξτε για να δώσετε όλα τα δυνατά δικαιώματα της σελίδας σε αυτόν τον χρήστη (μετατροπή του χρήστη σε διαχειριστή).
- Groups: Επιλέξτε τον ρόλο και κατ' επέκταση σε ποια ομάδα κατηγοριοποιείται ο χρήστης.
- User permissions: Επιλέξτε κάποια παραπάνω δικαιώματα για τον χρήστη (σε περίπτωση που ένας χρήστης ταιριάζει σε μια ομάδα αλλά υπάρχει ανάγκη για παραχώρηση περεταίρω δικαιωμάτων).
- Last login: Τελευταία φορά που ο χρήστης συνδέθηκε στην σελίδα διαχείρισης (συμπληρώνεται αυτόματα από την εφαρμογή).
- Date joined: Η ημέρα που ο χρήστης καταχωρήθηκε στο σύστημα για πρώτη φορά.

Product type: Τύποι που χωρίζονται οι κατηγορίες προϊόντων για παράδειγμα ηλεκτρονικές συσκευές.

Τα πεδία του Product type:

- Name: Το όνομα που θα περιγράφει τον τύπο των κατηγοριών που θα περιέχονται.
- Description: Εισαγωγή κειμένου που θα περιγράφει επαρκώς ποιες κατηγορίες προϊόντων ταιριάζουν στον συγκεκριμένο τύπο.

Product category: Κατηγορίες που περιέχουν προϊόντα και ανήκουν μόνο σε ένα τύπο κατηγοριών.

Τα πεδία του Product category:

- Name: Το όνομα που θα περιγράφει τη κατηγορία των προϊόντων.

- Description: Εισαγωγή κειμένου που θα περιγράφει επαρκώς ποια προϊόντα ταιριάζουν σε αυτήν την κατηγορία.
- Type: Ο τύπος που ανήκει η συγκεκριμένη κατηγορία.

Supplier: Καταγραφή των προμηθευτών με τους οποίους συνεργάζεται η αποθήκη για να της παρέχουν τα προϊόντα.

Τα πεδία του Supplier:

- Name: Το όνομα του προμηθευτή.
- Address: Η φυσική διεύθυνση του καταστήματος του προμηθευτή.
- Email: Η διεύθυνση του προσωπικού ηλεκτρονικού ταχυδρομείου του προμηθευτή.
- Phone: Το τηλέφωνο επικοινωνίας του προμηθευτή.

Product: Καταγραφή των προϊόντων που βρίσκονται στην αποθήκη.

Τα πεδία του Product:

- Name: Το όνομα του προϊόντος.
- Code: Ο κωδικός αναφοράς του προϊόντος.
- Image: Εισαγωγή εικόνας στην οποία απεικονίζεται το προϊόν.
- Description: Κείμενο το οποίο περιγράφει το προϊόν.
- Position: Καταγραφή της θέσης που κατέχει το προϊόν στα ράφια της αποθήκης.
- Category: Επιλογή της κατηγορίας που κατατάσσεται το προϊόν.
- Quantity: Ο αριθμός των διαθέσιμων προϊόντων που βρίσκεται στην αποθήκη.
- Price: Η τιμή πώλησης του προϊόντος (σε τι τιμή θα πωλείται το προϊόν από το κατάστημα).
- Supply price: Η τιμή που αγοράζεται το προϊόν από την αποθήκη (η τιμή του προμηθευτή για την αποθήκη).
- Vat: Ποσοστό φορολόγησης (εισαγωγή μιας αριθμητικής τιμής η οποία θα μετατραπεί σε %).
- Weight: Εισαγωγή του βάρους του προϊόντος (η αριθμητική τιμή θα αναφέρεται σε κιλά).
- SizeX: Πλάτος του προϊόντος (μέτρηση σε εκατοστά στον άξονα X).
- SizeY: Ύψος του προϊόντος (μέτρηση σε εκατοστά στον άξονα Y).
- SizeZ: Βάθος του προϊόντος (μέτρηση σε εκατοστά στον άξονα Z).

Featured products: Σε αυτήν την κατηγορία εισάγουμε τα προϊόντα τα οποία θα εκθέτονται στην κατηγορία των προτεινόμενων προϊόντων στο ηλεκτρονικό κατάστημα.

Τα πεδία του Featured products:

- Product id: Επιλογή προϊόντος το οποίο θα φαίνεται στα Featured του ηλεκτρονικού καταστήματος.

Supply Order: Παραγγελίες οι οποίες συντάσσονται από το προσωπικό της αποθήκης με σκοπό τον ανεφοδιασμό προϊόντων σε χαμηλό απόθεμα.

Τα πεδία του Supply Order:

- Code: Ο κωδικός αναφοράς της παραγγελίας.
- Assigned to: Επιλογή εργαζομένου ως υπεύθυνο για την περάτωση αυτής της παραγγελίας.
- Products: Επιλέγουμε τα προϊόντα της παραγγελίας συμπληρώνοντας τα παρακάτω πεδία (για να προσθέσουμε πάνω από ένα προϊόν μετά την συμπλήρωση πατάμε το κουμπί <<Save and continue editing>>):
    - Product: Το όνομα του προϊόντος.
    - Supplier: Τον προμηθευτή από τον οποίο θα προμηθευτούμε το προϊόν.
    - Quantity: Τον αριθμό κομματιών (πόσα από αυτά τα προϊόντα χρειάζονται).
    - Delete: Το επιλέγουμε για να σβήσουμε το προϊόν από την λίστα.
- Created at: Η ημερομηνία στην οποία δημιουργήθηκε η παραγγελία (θα συμπληρωθεί αυτόματα από την εφαρμογή όταν ολοκληρωθεί η παραγγελία).
- State: Η κατάσταση της παραγγελίας. Οι παραγγελίες οι οποίες έχουν μεγάλο χρηματικό αντίτιμο χρειάζονται έγκριση. Οι καταστάσεις είναι Open, Waiting approval και Closed. Ο χρήστης δεν μπορεί να επεξεργαστεί διότι το πεδίο ενημερώνεται από την εφαρμογή.
- Created by: Το username του εργαζομένου που σύνταξε την παραγγελία.

Shop orders: Σε αυτή την κατηγορία βρίσκονται οι παραγγελίες που έρχονται από το ηλεκτρονικό κατάστημα. Για αυτόν τον λόγο το μόνο πεδίο που μπορεί να αλλάξει ο χρήστης είναι το Assigned to, όλα τα υπόλοιπα είναι ήδη συμπληρωμένα από τον πελάτη που έκανε την παραγγελία.

Τα πεδία του Shop orders:

- Assigned to: Επιλέγεται ο χρήστης που θα επωμισθεί την ευθύνη περάτωσης της παραγγελίας.
- Name: Το όνομα του πελάτη.

- Surname: Το επώνυμο του πελάτη.
- Email: Η διεύθυνση του προσωπικού ηλεκτρονικού ταχυδρομείου του πελάτη.
- Phone: Το τηλέφωνο του πελάτη.
- City: Η πόλη στην οποία κατοικεί ο πελάτης (ή στην οποία θα γίνει η παράδοση).
- Address: Η διεύθυνση στην οποία κατοικεί ο πελάτης (ή στην οποία θα γίνει η παράδοση).
- TK: Ο ταχυδρομικός κώδικας της περιοχής του πελάτη.
- Shipping type: Ο τρόπος αποστολής που επέλεξε ο πελάτης (παραλαβή από κατάστημα ή παράδοση ταχυδρομικώς).
- Total price: Το συνολικό πόσο της παραγγελίας σε ευρώ.
- Total vat: Το συνολικό πόσο που καταλαμβάνει το Φ.Π.Α της παραγγελίας σε ευρώ.
- Total shipping: Το ποσό που θα πρέπει να καταβάλει ο χρήστης για την παράδοση της παραγγελίας.
- Products: Τα προϊόντα που περιλαμβάνονται στην παραγγελία.
- Created at: Ημερομηνία που δημιουργήθηκε η παραγγελία.
- Completed: Κατάσταση της παραγγελίας (αν ολοκληρώθηκε η όχι).

Postal codes: Για να μπορέσει το πρόγραμμα να βρει την χιλιομετρική διαφορά ανάμεσα στην αποθήκη και την περιοχή του κάθε πελάτη θα πρέπει πρώτα να καταχωρήσουμε όλους τους ταχυδρομικούς κώδικες από τις περιοχές που έχουμε επιλέξει να μεταφέρουμε τα προϊόντα.

Τα πεδία του Postal code:

- Code: Ταχυδρομικός κώδικας (π.χ. 14122).
- Lat: Γεωγραφικό πλάτος περιοχής που χαρακτηρίζεται από τον ταχυδρομικό κώδικα (Latitude).
- Lng: Γεωγραφικό μήκος περιοχής που χαρακτηρίζεται από τον ταχυδρομικό κώδικα (longitude).

Settings: Στο πεδίο settings υπάρχουν οι ρυθμίσεις απαραίτητες για κάποιες από τις λειτουργείες της εφαρμογής.

Τα πεδία του Settings:

- Low supply threshold: Ορίζεται το κατώφλι στο οποίο όταν το απόθεμα κάποιου προϊόντος πέσει κάτω από αυτό θα εμφανίζεται στην κατηγορία των προϊόντων χαμηλού αποθέματος.

- Aproval cost threshold: Ορίζουμε το ποσό (σε ευρώ) συνολικής αξίας παραγγελίας που αν το ξεπεράσει χρειάζεται έγκριση από υπεύθυνο για να συνεχιστεί η παραγγελία.
- Shop postal code: Ορίζουμε τον ταχυδρομικό κώδικα της περιοχής που βρίσκεται η αποθήκη.

Volumetric weight range: Προσθήκη πίνακα τιμών για κοστολόγηση μεταφοράς ανάλογα με το ογκομετρικό βάρος. Το ογκομετρικό βάρος ενός αντικειμένου βρίσκεται διαιρώντας το εμβαδόν του με το βάρος του. Για παράδειγμα αντικείμενο το οποίο ζυγίζει (Weight from) 1kg έως (Weight to) 5kg θα κοστολογείται (Cost) 3 ευρώ το κιλό.

Τα πεδία του volumetric weight range:

- Weight from: Προσθήκη βάρους αρχής (από).
- Weight to: Προσθήκη βάρους τέλος (μέχρι).
- Cost: Κοστολόγηση πεδίου βάρους (το κιλό).

Distance range: Προσθήκη πίνακα τιμών για κοστολόγηση μεταφοράς ανάλογα με την χιλιομετρική απόσταση. Για παράδειγμα από (Distance from) 1χλμ. έως (Distance to) 10χλμ. θα κοστίζει (Cost) 5 ευρώ το χιλιόμετρο.

Τα πεδία Distance range:

- Distance from: Προσθήκη χιλιομέτρου αρχής (από).
- Distance to: Προσθήκη χιλιομέτρου τέλος (μέχρι).
- Cost: Κοστολόγηση χιλιομέτρου (το κιλό).

2.2.4 Αρχιτεκτονική Εφαρμογής

Σε αυτό το κεφάλαιο περιγράφεται η αρχιτεκτονική της εφαρμογής καθώς και βασικές πληροφορίες για την κατανόησή τους.

2.2.4.1 Τι είναι το Json

- JSON σημαίνει **J**ava**S**cript **O**bject **N**otation.
- JSON είναι μια ελαφριάς μορφής ανταλλαγής δεδομένων.
- Το JSON χρησιμοποιείται ανεξαρτήτως γλώσσας προγραμματισμού.
- Το JSON είναι εύκολα κατανοητό καθώς περιγράφει τον εαυτό του.

Τα δεδομένα πρέπει να μορφοποιηθούν σε απλό κείμενο για να πετύχει η αποστολή τους από εξυπηρετητή σε οποιοδήποτε πρόγραμμα περιήγησης και αντίστροφα. Το JSON είναι απλό κείμενο, μπορούμε να μετατρέψουμε οποιοδήποτε αντικείμενο της JavaScript σε JSON και να το στείλουμε σε έναν εξυπηρετητή. Μπορούμε επίσης να μετατρέψουμε με ευκολία οποιοδήποτε JSON αρχείο σε αντικείμενο JavaScript. Επομένως έχουμε τη δυνατότητα να κάνουμε χρήση των δεδομένων χωρίς περίπλοκες μεταφράσεις ή αλλαγή συντακτικού.

Το JSON παρουσιάζεται με τις εξής δυο αρχιτεκτονικές:

- Μια συλλογή ζευγαριών από ονόματα και τιμές. Σε πολλές γλώσσες αυτό μπορεί να μεταφραστεί ως ένα αντικείμενο, μια δομή δεδομένων, λεξικό, πίνακα η οποιαδήποτε συνεκτικό πίνακα.
- Μια λίστα από τιμές. Σε πολλές γλώσσες αυτό μπορεί να μεταφραστεί ως ένας πίνακας, διάνυσμα, λίστα από τιμές.

Αυτές είναι παγκόσμιες δομές δεδομένων. Μπορούν να χρησιμοποιηθούν από όλες τις γλώσσες προγραμματισμού.
Παρακάτω βρίσκεται ένα παράδειγμα ενός JSON εγγράφου που περιγράφει στοιχεία επικοινωνίας:

```
{
  "Όνομα" :  "Μάρκου Βασίλειος",
  "Τίτλος" : "Web Developer",
  "Τοποθεσία" : "Αθήνα",
  "twitter" : "@MongoDB",
  "facebook" : "@MongoDB"
}
```

Το JSON βασίζεται σε ένα υποσύνολο του JavaScript Programming Language Standard ECMA-262 3rd Edition - December 1999.

2.2.4.2 Τι είναι το HTTP

HTTP σημαίνει **H**yper**T**ext **T**ransfer **P**rotocol. Είναι το υποκείμενο πρωτόκολλο που χρησιμοποιείται από το διαδίκτυο και καθορίζει το πως μεταδίδονται και διαρθρώνονται τα μηνύματα καθώς και τις πράξεις που οι διαδικτυακοί εξυπηρετητές και τα προγράμματα περιήγησης πρέπει να εκτελέσουν ανάλογα με τα αιτήματα που θα λάβουν.

2.2.4.3 Τι είναι το REST

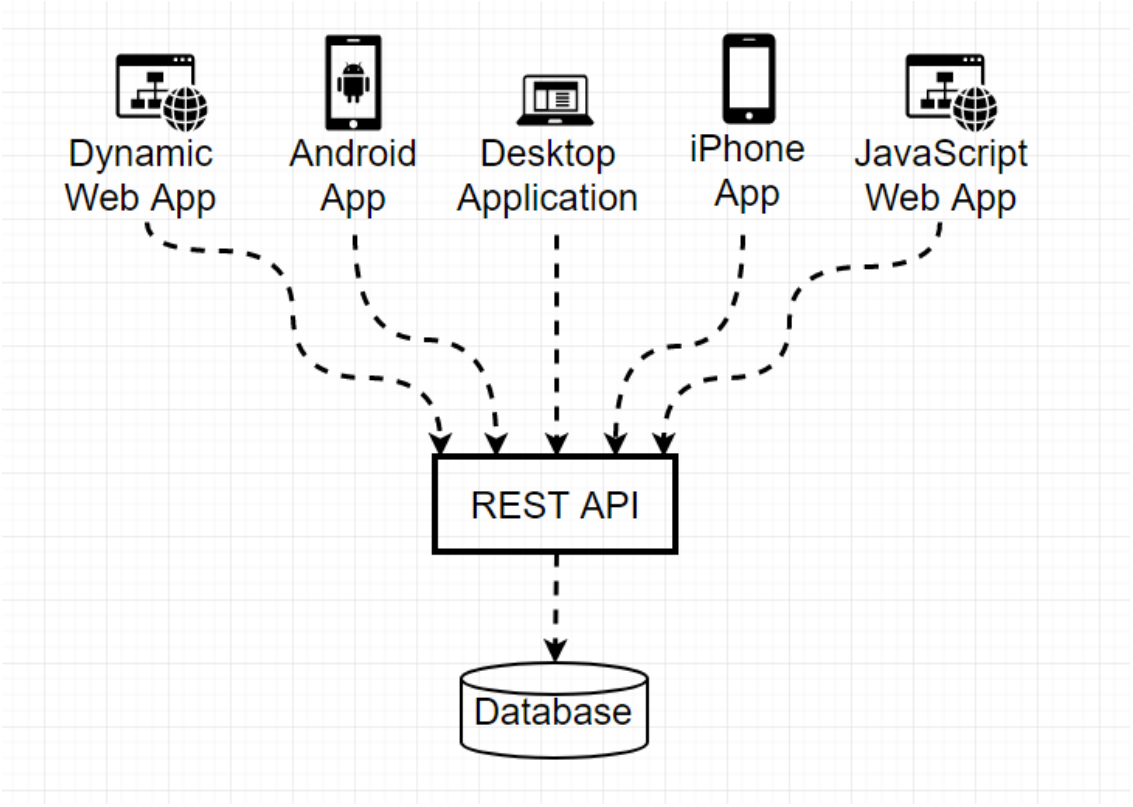
REST σημαίνει **Re**presentational **S**tate **T**ransfer. Βασίζεται σε ένα χωρίς κατάσταση (stateless), πελάτη - εξυπηρετητή, cacheable (προσωρινά αποθηκευμένο) πρωτόκολλο επικοινωνίας. Σχεδόν σε όλες τις περιπτώσεις γίνεται χρήση του HTTP πρωτόκολλου. Το REST είναι μια μορφή αρχιτεκτονικής για τον σχεδιασμό διαδικτυακών εφαρμογών. Το REST χρησιμοποιείται έναντι των πιο περίπλοκων μηχανισμών RPC, CORBA ή SOAP για να επιτευχθεί η σύνδεση μεταξύ μηχανημάτων.

Κάποια από τα προτερήματα του REST είναι:

- Ελαφρύ σε σχέση με τους ανταγωνιστές (RPC, CORBA, SOAP).
- Απλό στην υλοποίηση.
- Η αρχιτεκτονική REST έχει παραπλήσιο σχεδιασμό με αυτόν του διαδικτύου.
- Επεκτασιμότητα.
- Η χρήση των υπηρεσιών HTTP σημαίνει πως είναι εφικτή η επικοινωνία με σχεδόν οποιαδήποτε διαδικτυακή συσκευή χωρίς να καθιστά απαραίτητο την γνώση του κάθε API.

2.2.4.4 REST API

API σημαίνει **A**pplication **P**rogramming **I**nterface. Είναι ένα σύνολο κανόνων και διευκρινίσεων που μπορούν να ακολουθήσουν διάφορα προγράμματα για να επιτευχθεί η επικοινωνία μεταξύ τους. Εξυπηρετεί τον ρόλο της διεπαφής μεταξύ διαφόρων προγραμμάτων και διευκολύνει τις αντιδράσεις τους.



Ένα REST API καθορίζει ένα σύνολο συναρτήσεων όπου κατά την εκτέλεση τους εφαρμόζουν αιτήματα και λαμβάνουν απαντήσεις μέσω HTTP πρωτοκόλλου όπως GET και POST.

2.2.4.5 Τι είναι το Django Rest

Django Rest (DR) είναι μια βιβλιοθήκη που σε συνεργασία με τα μοντέλα του Django επιτρέπει την δημιουργία ενός ευέλικτου και δυνατού API.

Ένα DR API αποτελείται από τρία βασικά κομμάτια:

- Serializers: μετατρέπουν τις πληροφορίες που βρίσκονται αποθηκευμένες στην βάση δεδομένων και καθορίζονται από τα μοντέλα του Django με μορφοποίηση που είναι εύκολα διαβιβάσιμη μέσω ενός API.
- Viewset: καθορίζει τις συναρτήσεις (read, create, update, delete) που θα είναι διαθέσιμες μέσω του API.
- Routers: καθορίζει τα URLs που θα παρέχουν πρόσβαση στο κάθε viewset.

Serializers: Τα μοντέλα του Django αντιπροσωπεύουν τα δεδομένα που βρίσκονται αποθηκευμένα στη βάση δεδομένων. Για να καταφέρει ένα API την μετάδοση τους θα πρέπει

πρώτα να τα μετατρέψει σε μια πιο απλή δομή. Τα δεδομένα αναπαρίστανται ως ένα στιγμιότυπο της κλάσης του μοντέλου σε κώδικα Python, πρέπει να μορφοποιηθούν σε JSON για να μπορούν να σταλούν μέσω του API. Ο serializer που παρέχει το DR κάνει ακριβώς αυτό. Όταν ο χρήστης υποβάλει πληροφορίες μέσω του API ο serializer αφού τα ελέγξει για την εγκυρότητα τους τα μετατρέπει σύμφωνα με το μοντέλο του Django και τα αποθηκεύει σαν στιγμιότυπο αυτού του μοντέλου στη βάση. Με παρόμοιο τρόπο όταν ο χρήστης χρειάζεται πρόσβαση σε πληροφορίες το API τις συλλέγει από τη βάση, τις δίνει στο serializer ,τους αλλάζει μορφοποίηση και τις στέλνει ως JSON στον χρήστη. Με τη χρήση των πεδίων υπάρχει η δυνατότητα να ορίσουμε ποια πεδία είναι προσβάσιμα χρησιμοποιώντας τον συγκεκριμένο serializer.

ViewSets: Στα ViewSets υπάρχουν όλες οι διαδικασίες/συναρτήσεις δηλωμένες. Το πιο διαδεδομένο ViewSet είναι το ModelViewSet όπου έχει ήδη τις συγκεκριμένες λειτουργίες:

- Δημιουργία στιγμιότυπου: create()
- Διάβασμα/ανάκτηση πληροφορίας: retrieve()
- Ενημέρωση στιγμιότυπου (όλα τα πεδία ή επιλεγμένα): update()/partial update().
- Διαγραφή στιγμιότυπου: destroy()
- Δημιουργία στιγμιότυπου: list().

Οι συναρτήσεις μπορούν να αλλάξουν αν χρειαστεί περεταίρω παραμετροποίηση. Υπάρχει η δυνατότητα δημιουργίας view.

Routers: Τέλος, στο DR router υπάρχουν όλα τα URLs για ένα ViewSet σε μια σειρά.

2.2.4.5.1 Αρχιτεκτονική εφαρμογής

Η εφαρμογή αποτελείται από μια βάση δεδομένων Mongo DB, ένα API φτιαγμένο με Django Rest και στο front-end χρησιμοποιούμε React framework και Django.

Εμβαθύνοντας, η Mongo DB είναι ένα πρόγραμμα βάσης δεδομένων που χρησιμοποιεί έγγραφα τύπου JSON. Όλες οι πληροφορίες της εφαρμογής αποθηκεύονται εκεί. Το API το οποίο δημιουργήθηκε με Django REST παίζει τον ρόλο της διεπαφής ανάμεσα στον client και στη βάση δεδομένων. Με HTTP requests το API αποκτά τις απαραίτητες πληροφορίες που ζητήθηκαν από τον client σε μορφή JSON και τις στέλνει στο Front-end. Το Front-end παραλαμβάνει τις πληροφορίες και τις μορφοποιεί κατάλληλα με την χρήση React (JavaScript, HTML, CSS) ώστε να γίνουν ευπαρουσίαστες στον χρήστη με σκοπό την καλύτερη κατανόησή τους.

Στο αρχείο models.py γράφουμε τα μοντέλα που θα χρησιμοποιήσουμε για στην εφαρμογή. Για παράδειγμα, ένα βασικό μοντέλο είναι το προϊόν, το ονομάζουμε Product. Του προσδίδουμε τα πεδία που πρέπει να περιλαμβάνει ένα προϊόν όπως όνομα, κωδικό,

περιγραφή κ.τ.λ. καθώς και τον τύπο των δεδομένων. Υπάρχει δυνατότητα παραμετροποίησης του πεδίου που θα γραφτεί στη βάση δεδομένων. Η πιο συνήθης επιλογή είναι να χρησιμοποιείται το όνομα του μοντέλου. Η συνάρτηση def __str__(self):return self.name εκτελεί αυτό ακριβώς προσδίδοντας ένα όνομα πιο φιλικό προς τους ανθρώπους. Υπάρχουν πολλές δυνατότητες όπως η προσθήκη φυσικού κλειδιού (natural key). Ένα φυσικό κλειδί είναι μια ομάδα τιμών που μπορούν να χρησιμοποιηθούν για τον μοναδικό προσδιορισμό ενός μοντέλου χωρίς τη χρήση της αρχικής τιμής κλειδιού. Αφού δημιουργηθούν όλα τα μοντέλα σειρά έχουν τα URLs που βρίσκονται στο αρχείο urls.py. Κάνοντας χρήση της γλώσσας python σε αυτό το αρχείο μπορούμε να αντιστοιχήσουμε τις εκφράσεις διαδρομής (URL) σε συναρτήσεις ή σε views που έχουμε δημιουργήσει. Στον φάκελο views υπάρχουν όλα τα views που χρησιμοποιεί η εφαρμογή. Το view είναι μια συνάρτηση Python που λαμβάνει ένα αίτημα (request) και επιστρέφει μια απόκριση (responce). Αυτή η απόκριση μπορεί να είναι το περιεχόμενο HTML μιας ιστοσελίδας, μια ανακατεύθυνση, ένα σφάλμα 404, ένα έγγραφο XML, μια εικόνα ή οτιδήποτε. Η ίδια η προβολή περιέχει όποια αυθαίρετη λογική είναι απαραίτητη για την επιστροφή αυτής της απόκρισης. Τέλος χρησιμοποιήθηκε ένας serializer για την μετατροπή των δεδομένων σε JSON με σκοπό την μεταφορά τους.

Η εφαρμογή αποτελείται από δύο υπό-εφαρμογές, την εφαρμογή διαχείρισης της αποθήκης και το ηλεκτρονικό κατάστημα της (e-shop). Για το χτίσιμο των UI αυτών των εφαρμογών χρησιμοποιήθηκε για την πρώτη το Django και για το e-shop το framework React.

Το Django μας παρέχει το Django admin panel. Πρόκειται για μια διεπαφή η οποία διαβάζει τα μοντέλα που υπάρχουν και προσαρμόζεται αναλόγως δίνοντας μια έτοιμη σχεδόν εφαρμογή η οποία μπορεί να παραμετροποιηθεί σε έναν βαθμό. Τα μοντέλα που θα καταχωρηθούν στο αρχείο admin.py θα καθοριστούν επεξεργάσιμα από την διεπαφή. Στο αρχείο models_admin.py αλλάζουμε κάποιες ιδιότητες των συναρτήσεων που παρέχονται. Στον φάκελο templates/admin βρίσκονται οι αλλαγές στο UI της εφαρμογής διαχείρισης οι οποίες είναι ελάχιστες.

Ο σχεδιασμός του Frontend βρίσκεται στον φάκελο frontend. Όλα τα στοιχεία που χρησιμοποιήθηκαν για την δημιουργία του UI του ηλεκτρονικού καταστήματος βρίσκονται εκεί. Όπως προαναφέρθηκε γίνεται χρήση του React framework.

Η δομή είναι η εξής:

Φάκελος components: Περιέχει όλα τα components που χρησιμοποιούνται για την δημιουργία των σελίδων του e-shop.

Φάκελος core: Σε αυτόν τον φάκελο υπάρχουν υπάρχουν βασικά εργαλεία όπως μια συνάρτηση για redirect (ανακατεύθηνση), page context (οι βασικές πληροφορίες που χρειάζεται από την αρχή μια σελίδα), request.js (η συνάρτηση που χρησιμοποιείται για τα HTTP requests).

Φάκελος layouts: Περιέχει το βασικό κομμάτι των σελίδων. Αυτό που μένει ίδιο και περιβάλει όλες τις σελίδες.

Φάκελος pages: Ολες οι σελίδες που απαρτίζουν το ηλεκτρονικό κατάστημα βρίσκονται στον φάκελο pages σε φάκελο με το όνομα τους που μέσα υπάρχει και το scss αρχείο της κάθε σελίδας.

2.2.5 Deployment της εφαρμογής

Τα εργαλεία που χρησιμοποιήθηκαν για να ανεβεί η εφαρμογή στο διαδίκτυο είναι το Nginx, github και docker.

2.2.5.1 Τι είναι το Nginx

Το NGINX είναι λογισμικό ανοιχτού κώδικα για web serving, reverse proxying, προσωρινή αποθήκευση (cashing), εξισορρόπηση φορτίου (load balancing), ροή πολυμέσων και πολλά άλλα. Ξεκίνησε ως διακομιστής Ιστού (web server) σχεδιασμένος για μέγιστη απόδοση και σταθερότητα. Εκτός από τις δυνατότητες του διακομιστή HTTP, το NGINX μπορεί επίσης να λειτουργήσει ως διακομιστής μεσολάβησης (proxy server) για email (IMAP, POP3 και SMTP) και ως αντίστροφος διακομιστής μεσολάβησης και εξισορρόπησης φορτίου για διακομιστές HTTP, TCP και UDP.

Ένας διακομιστής μεσολάβησης (proxy server) λειτουργεί ως πύλη μεταξύ των χρηστών και του Διαδικτύου. Είναι ένας ενδιάμεσος διακομιστής που χωρίζει τους τελικούς χρήστες από τους ιστότοπους που περιηγούνται. Οι διακομιστές μεσολάβησης παρέχουν διαφορετικά επίπεδα λειτουργικότητας, ασφάλειας και απορρήτου ανάλογα με την περίπτωση χρήσης, τις ανάγκες ή την πολιτική της εκάστοτε εταιρείας. Με την χρήση διακομιστή μεσολάβησης, η κίνηση στο Διαδίκτυο ρέει μέσω του διακομιστή μεσολάβησης προς τη διεύθυνση που ζητείται. Στη συνέχεια, το αίτημα επιστρέφει μέσω του ίδιου διακομιστή μεσολάβησης (υπάρχουν εξαιρέσεις σε αυτόν τον κανόνα) και, στη συνέχεια, ο διακομιστής μεσολάβησης προωθεί τα δεδομένα που λαμβάνονται από τον ιστότοπο σε αυτόν που τα ζήτησε.

Ο Igor Sysoev έγραψε αρχικά το NGINX για την επίλυση του προβλήματος C10K, έναν όρο που δημιουργήθηκε το 1999 για να περιγράψει τη δυσκολία που αντιμετώπισαν οι υπάρχοντες διακομιστές ιστού στο χειρισμό μεγάλων αριθμών (10K) ταυτόχρονων συνδέσεων (C). Με την ασύγχρονη αρχιτεκτονική που βασίζεται σε events, το NGINX έφερε επανάσταση στον τρόπο λειτουργίας των διακομιστών σε περιβάλλοντα υψηλής απόδοσης και έγινε ο ταχύτερος διαθέσιμος διακομιστής ιστού. Μετά την αλλαγή του λογισμικού σε λογισμικό ανοιχτού κώδικα το 2004 και βλέποντας τη χρήση του να αναπτύσσεται εκθετικά, η Sysoev συν-ίδρυσε τη NGINX, Inc. για να υποστηρίξει τη συνεχή ανάπτυξη του NGINX και να εμπορεύεται το NGINX Plus ως εμπορικό προϊόν με πρόσθετα χαρακτηριστικά σχεδιασμένα για εταιρικούς πελάτες. Το NGINX, Inc. έγινε μέλος της F5, Inc. το 2019.

Ο στόχος πίσω από το NGINX ήταν η δημιουργία του ταχύτερου διακομιστή διαδικτύου και η διατήρηση αυτού του κατορθώματος εξακολουθεί να αποτελεί κεντρικό στόχο του έργου. Το NGINX έχει καλύτερες μετρήσεις απόδοσης (benchmarks) από τον Apache και άλλους διακομιστές.

2.2.5.2 Τι είναι το Docker

Το Docker είναι μια ανοιχτή πλατφόρμα για ανάπτυξη, αποστολή και εκτέλεση εφαρμογών. Το Docker επιτρέπει να διαχωρίζονται οι εφαρμογές από την υποδομή, ώστε να μπορεί να παραδίδεται λογισμικό γρήγορα. Με το Docker, μια υποδομή τη διαχειριζόμαστε με τον ίδιο τρόπο που διαχειριζόμαστε οποιαδήποτε άλλη εφαρμογή. Αξιοποιώντας τις μεθοδολογίες του Docker για αποστολή, δοκιμή και ανάπτυξη κώδικα γρήγορα, μειώνεται σημαντικά η καθυστέρηση μεταξύ της σύνταξης κώδικα και της εκτέλεσης του. Το Docker παρέχει τη δυνατότητα συσκευασίας και εκτέλεσης μιας εφαρμογής σε ένα όχι αυστηρό απομονωμένο περιβάλλον που ονομάζεται container. Η απομόνωση και η ασφάλεια επιτρέπουν να εκτελούνται πολλά container ταυτόχρονα σε έναν συγκεκριμένο κεντρικό υπολογιστή. Τα container είναι ελαφριά και περιέχουν όλα όσα χρειάζονται για την εκτέλεση της εφαρμογής, οπότε δεν χρειάζεται να είναι προ-εγκατεστημένο τίποτα στον κεντρικό υπολογιστή (εκτός του Docker). Υπάρχει δυνατότητα κοινής χρήσης container και όλα τα container λειτουργούν με τον ίδιο τρόπο.

Το Docker παρέχει εργαλεία και πλατφόρμα για τη διαχείριση του κύκλου ζωής των container :

- Ανάπτυξη εφαρμογής και υποστήριξη των εξαρτημάτων με τη χρήση των container.
- Το container γίνεται η μονάδα διανομής και δοκιμής της εφαρμογής.
- Ανάπτυξη της εφαρμογής στο περιβάλλον παραγωγής, ως container. Αυτό λειτουργεί το ίδιο εάν το περιβάλλον παραγωγής είναι ένα τοπικό κέντρο δεδομένων, ένας πάροχος cloud ή ένα υβρίδιο των δύο.

Image είναι ένα πρότυπο μόνο για ανάγνωση, με οδηγίες για τη δημιουργία ενός Docker container. Συχνά, ένα image βασίζεται σε άλλο image, με κάποια επιπλέον προσαρμογή. Για παράδειγμα, μπορεί να δημιουργηθεί ένα image που βασίζεται στο image του Ubuntu, αλλά εγκαθιστά τον διακομιστή ιστού Nginx και την εφαρμογή, καθώς και τις λεπτομέρειες διαμόρφωσης που απαιτούνται για την εκτέλεση της εφαρμογής. Για να δημιουργηθεί ένα image, πρέπει να δημιουργηθεί ένα Dockerfile με μια απλή σύνταξη για τον καθορισμό των βημάτων που απαιτούνται για τη δημιουργία και την εκτέλεση του image. Κάθε εντολή σε ένα Dockerfile δημιουργεί ένα επίπεδο στο image. Όταν αλλάζεται το Dockerfile και δημιουργείται ξανά το image, μόνο τα επίπεδα που έχουν αλλάξει ξαναχτίζονται. Αυτός είναι ο λόγος που τα images είναι τόσο ελαφριά, μικρά και γρήγορα, σε σύγκριση με άλλες τεχνολογίες.

Το container είναι μια τρέχουσα παρουσία (instance) ενός image. Ένα container μπορεί να δημιουργηθεί, να ξεκινήσει, να σταματήσει, να μετακινηθεί ή να διαγραφτεί χρησιμοποιώντας το Docker API ή CLI. Ένα container μπορεί να συνδεθεί σε ένα ή περισσότερα δίκτυα, να επισυναφτεί χώρος αποθήκευσης σε αυτό ή ακόμα και να

δημιουργηθεί μια νέα εικόνα με βάση την τρέχουσα κατάστασή του. Από προεπιλογή, ένα container είναι σχετικά καλά απομονωμένο από άλλα δοχεία και το μηχάνημα υποδοχής του. Υπάρχει έλεγχος στο πόσο απομονωμένο είναι το δίκτυο, ο αποθηκευτικός χώρος ή άλλα υποσυστήματα ενός container από άλλα container ή από τον κεντρικό υπολογιστή. Ένα container ορίζεται από το image του καθώς και από τυχόν επιλογές διαμόρφωσης που παρέχετε σε αυτό κατά τη δημιουργία ή την εκκίνηση του. Όταν αφαιρείται ένα container, τυχόν αλλαγές στην κατάστασή του που δεν αποθηκεύονται σε μόνιμο χώρο αποθήκευσης εξαφανίζονται.

2.2.5.3 Τι είναι το github

Το GitHub είναι ένας ιστότοπος και μια υπηρεσία που βασίζεται στο cloud που βοηθά τους προγραμματιστές να αποθηκεύουν και να διαχειρίζονται τον κώδικά τους, καθώς και να παρακολουθούν και να ελέγχουν τις αλλαγές στον κώδικά τους. Για τη καλύτερη κατανόηση του GitHub, θα εξηγηθούν δύο συνδεδεμένες αρχές.

Το version control (έλεγχος έκδοσης) βοηθά τους προγραμματιστές να παρακολουθούν και να διαχειρίζονται τις αλλαγές στον κώδικα ενός έργου λογισμικού. Καθώς μεγαλώνει ένα έργο λογισμικού, ο έλεγχος της έκδοσης καθίσταται απαραίτητος. Το version control επιτρέπει στους προγραμματιστές να εργάζονται με ασφάλεια με branching και merging. Με branching (διακλάδωση), ένας προγραμματιστής αντιγράφει μέρος του πηγαίου κώδικα (που ονομάζεται αποθετήριο). Ο προγραμματιστής μπορεί στη συνέχεια να κάνει με ασφάλεια αλλαγές σε αυτό το τμήμα του κώδικα χωρίς να επηρεάσει το υπόλοιπο έργο. Στη συνέχεια, μόλις ο προγραμματιστής πάρει το τμήμα του κώδικα που λειτουργεί σωστά, μπορεί να συγχωνεύσει (merging) αυτόν τον κωδικό στον κύριο πηγαίο κώδικα για να τον κάνει επίσημο. Το Git είναι ένα σύστημα version control ανοιχτού κώδικα που δημιουργήθηκε από τον Linus Torvalds το 2005. Συγκεκριμένα, το Git είναι ένα σύστημα ελέγχου κατανεμημένης έκδοσης, το οποίο σημαίνει ότι ολόκληρη η βάση κώδικα και το ιστορικό είναι διαθέσιμα σε κάθε υπολογιστή προγραμματιστή, το οποίο επιτρέπει την εύκολη διακλάδωση και συγχώνευση. Σύμφωνα με μια έρευνα για προγραμματιστές Stack Overflow, πάνω από το 87% των προγραμματιστών χρησιμοποιούν το Git.

2.2.6 Δημοσίευση της εφαρμογής στο διαδίκτυο

Αφού νοικιάστηκε ένα μηχάνημα server από το google cloud δημιουργήθηκε ένα εικονικό περιβάλλον (virtual machine) σε αυτό με Linux. Με το Docker δημιουργήθηκαν τα images που θα παρήγαγαν τα container της εφαρμογής τα οποία είναι container με το front-end, back-end, mongo και Nginx. Το Docker εγκαταστάθηκε στο virtual machine στο νοικιασμένο μηχάνημα του google cloud και από εκεί με χρήση του Git κατεβάσαμε την εφαρμογή. Με τις εντολές docker-compose build δημιουργήθηκε αυτόματα το περιβάλλον κατάλληλο για να λειτουργήσει η εφαρμογή και με την εντολή docker-compose up τέθηκε σε λειτουργία. Το Nginx χρησιμοποιήθηκε για proxy το οποίο τοποθετήθηκε ανάμεσα στην εφαρμογή και τους χρήστες με σκοπό την σωστή κατεύθυνση των request. Το Nginx βοήθησε να παρουσιαστεί σαν μια εφαρμογή το Django και το next.js (server side rendering για τις ανάγκες του React). Πιο λεπτομερειακά το Nextjs το χρησιμοποιούμε για να γίνεται η φόρτωση μόνο των

Javascript και CSS που απαιτούνται για οποιαδήποτε δεδομένη σελίδα. Αυτό καθιστά πολύ πιο γρήγορους τους χρόνους φόρτωσης σελίδων, καθώς το πρόγραμμα περιήγησης ενός χρήστη δεν χρειάζεται να κάνει λήψη Javascript και CSS που δεν χρειάζεται για τη συγκεκριμένη σελίδα που βλέπει ο χρήστης. Για να μην αναγκάζεται ο χρήστης να αναζητά την εφαρμογή χρησιμοποιώντας IP με το NO-IP αποκτήθηκε ένα domain και έγινε χρήση ενός dynamic DNS server για να χρησιμοποιηθεί το όνομα storage-handler.mydissent.net. Ο οποιοσδήποτε λοιπόν μπορεί γράφοντας στο διαδίκτυο http://storage-handler.mydissent.net/ να επισκεφτεί το ηλεκτρονικό κατάστημα και προσθέτοντας /admin δηλαδή γράφοντας http://storage-handler.mydissent.net/admin/ να εισέλθει στην σελίδα διαχείρισης της αποθήκης δεδομένου ότι κατέχει τα απαραίτητα στοιχεία (username, password).

2.3 Εργαλεία ανάπτυξης που χρησιμοποιήθηκαν

- Django
- Django Rest
- Expess.js
- Next.js
- React
- MongoDB
- Nginx
- Docker
- Για IDE vscode

2.4 Λόγοι χρήσης των εργαλείων

2.4.1 REST API

Μπορούμε να χρησιμοποιήσουμε το REST API ανεξαρτήτως γλώσσας προγραμματισμού η πλατφόρμας. Το REST API προσφέρει μια ελευθερία στην αλλαγή ή δοκιμή καινούργιου περιβάλλοντος στην φάση της ανάπτυξης διότι, υιοθετεί τον τύπο του συντακτικού της πλατφόρμας που χρησιμοποιούμε κάθε φορά. Ο μόνος περιορισμός είναι ότι τα request/responses πρέπει πάντα να χρησιμοποιούν την ίδια μορφοποίηση, συνήθως JSON ή XML.

Διαχωρισμός του server με τον client. Το πρωτόκολλο REST διαχωρίζει το user interface από τον server και την βάση δεδομένων. Με αυτόν τον τρόπο προκύπτουν τα εξής πλεονεκτήματα. Ξεχωριστή εξέλιξη των διαφόρων στοιχείων ανάπτυξης και βελτίωση της φορητότητας του interface σε πλατφόρμες άλλου τύπου.

Αξιοπιστία, επεκτασιμότητα και ορατότητα. Ένα από τα εμφανή προτερήματα του διαχωρισμού server/client είναι ότι οι ομάδες υπεύθυνες για την ανάπτυξη τους μπορούν να κλιμακώσουν το προϊόν με ευκολία. Μπορούν να εκτελέσουν οποιαδήποτε αλλαγή στη βάση δεδομένων με την προϋπόθεση τα δεδομένα από το κάθε request να στέλνονται σωστά.

2.4.2 Django

Το Django παρέχει έτοιμο ένα από τα καλύτερα συστήματα ασφάλειας που βοηθάει τους προγραμματιστές να αποφύγουν κοινά λάθη όπως: clickjacking, cross-site scripting και SQL injection. Επιπρόσθετα το Django κυκλοφορεί καινούργιες αναβαθμίσεις ασφαλείας και είναι από τα πρώτα Frameworks που αντιμετωπίζει νέα κενά ασφαλείας και προειδοποίει τα υπόλοιπα.

Πλούσιο οικοσύστημα. Το Django περιέχει πολλές «third-party» εφαρμογές οι οποίες μπορούν να ενσωματωθούν σε διάφορες εφαρμογές ανάλογα με τις ανάγκες. Το Django αποτελείται από πολλές τέτοιες εφαρμογές όπως για παράδειγμα εφαρμογή εξουσιοδότησης και αποστολής emails που μπορεί με ευκολία να ενσωματωθεί στο σύστημα. Επίσης το Django κυκλοφορεί έντεκα χρόνια. Είναι ένα ώριμο framework που έχει περάσει από πολλά στάδια και έχουν επιτευχθεί πληθώρα βελτιώσεων. Άλλο ένα προτέρημα αποτελεί και η κοινότητά του η οποία έχει δώσει απαντήσεις σε πολλά ερωτήματα σχετικά με τη λειτουργία του.

Η γλώσσα προγραμματισμού python είναι διάσημη για τον ευανάγνωστο κώδικά της. Αυτό συνεισφέρει στην ιστοσελίδα να τοποθετηθεί σε υψηλή θέση στα αποτελέσματα της μηχανής αναζήτησης. Με το Django έχουμε τη δυνατότητα να παράξουμε ευανάγνωστα URLs χρησιμοποιώντας σχετικές λέξεις -–κλειδιά και καλύτερες πρακτικές SEO (search engine optimization).

Ένα από τα πιο σημαντικά κομμάτια του Django είναι το ORM (Object Relational Mapper) του. Το ORM περιγράφεται ως μια στρώση ανάμεσα στη βάση δεδομένων και την εφαρμογή, με ένα δυνατό API για να καθορίζει , να διατηρεί και να παρέχει πρόσβαση στα δεδομένα.

Το documentation του django ήταν πάντα πλήρης και πολύ φιλικό προς τους χρήστες και στους νέους προγραμματιστές, χάρη στα πολλά παραδείγματα και σεμινάρια. Το documentation καλύπτει όλα τα API του django και περιγράφει λεπτομερειακά από την πιο απλή χρήση τους έως την πιο πολύπλοκη. Όπως εξηγείται στα έγγραφα, το documentation του django προσφέρει διαφορετικά επίπεδα ανάγνωσης:

- Τα σεμινάρια καθοδηγούν με σειρά βημάτων για τη δημιουργία μιας εφαρμογής Web.
- Οι οδηγοί θεμάτων πραγματεύονται βασικά θέματα και έννοιες σε αρκετά υψηλό επίπεδο και παρέχουν χρήσιμες βασικές πληροφορίες και εξηγήσεις.

- Οι οδηγοί αναφοράς περιέχουν τεχνική περιγραφή για API και άλλες πτυχές των στοιχείων του Django. Περιγράφει πώς λειτουργεί και πώς χρησιμοποιείται, με την προϋπόθεση μιας στοιχειώδους κατανόηση των βασικών εννοιών.

- Οι «How-to» οδηγοί βοηθούν στην αντιμετώπιση βασικών προβλημάτων. Είναι πιο προχωρημένοι από τα σεμινάρια και προϋποθέτουν γνώσεις για το πώς λειτουργεί το Django.

Προτέρημα αποτελεί και η γλώσσα Python που χρησιμοποιεί το Django. Διαθέτει μια πολύ δυνατή βιβλιοθήκη και είναι διαθέσιμη σε πολλές πλατφόρμες. Πολλές αναβλύζουσες τεχνολογίες όπως A.I και Big Data κάνουν χρήση της γλώσσας.

Το Django περιέχει το πακέτο django.contrib.admin, μια επαναχρησιμοποιήσιμη εφαρμογή που δημιουργεί μια διεπαφή για διαχειριστές που έχει βάση το διαδίκτυο για τα υπάρχοντα μοντέλα δεδομένων. Το django admin είναι πολύ ενδιαφέρων για πολλούς λόγους:

- Μπορεί να δοκιμαστεί με μηδενικό κόστος: είναι ενεργοποιημένο από προεπιλογή και χρειάζεστε 4 γραμμές κώδικα για να λάβετε μια πλήρη διεπαφή CRUD για ένα μοντέλο δεδομένων.

- Αξιοποιεί και δείχνει όλη τη δύναμη του django ORM

- Είναι ένα όμορφο παράδειγμα μιας παραμετροποιήσιμης εφαρμογής ιστού, με απλό API και καθαρό user interface.

- Είναι ένα ενδιαφέρον εργαλείο διαχείρισης, ιδίως εάν στην εφαρμογή ιστού χρησιμοποιείτε μόνο το ORM για πρόσβαση στα μοντέλα δεδομένων.

Το Django admin είναι κατάλληλο ως μια απλή διεπαφή διαχείρισης για ορισμένους χρήστες αν το μοντέλο δεδομένων δεν είναι περίπλοκο. Κατά τη διάρκεια της ανάπτυξης, το Django admin μπορεί επίσης να χρησιμοποιηθεί για εντοπισμό σφαλμάτων και για την τοποθέτηση ορισμένων δεδομένων στη βάση δεδομένων σας.

Το API του django admin παρέχει πολλές επιλογές για την προσαρμογή της σελίδας του διαχειριστή στις εκάστοτε ανάγκες και υπάρχουν πολλά πακέτα για τη βελτίωση της διεπαφής διαχειριστή και της εμπειρίας χρήστη.

Κάθε γλώσσα προγραμματισμού συνοδεύεται από τη δική της σειρά βιβλιοθηκών για την επίλυση κοινών προβλημάτων. Μια βιβλιοθήκη λογισμικού περιλαμβάνει έτοιμο κώδικα, κλάσεις, διαδικασίες, scripts, configuration data και άλλα. Κατά κανόνα, μια βιβλιοθήκη προστίθεται σε ένα πρόγραμμα για να παρέχει περισσότερη λειτουργικότητα ή να αυτοματοποιεί μια διαδικασία χωρίς να πρέπει να γραφτεί χειροκίνητα νέος κώδικας. Αυτό μειώνει το χρόνο παραγωγής.

Το Django επιτρέπει στους προγραμματιστές να χρησιμοποιούν βιβλιοθήκες κατά την κατασκευή οποιουδήποτε έργου. Ορισμένες δημοφιλείς βιβλιοθήκες περιλαμβάνουν το πλαίσιο Django REST, το οποίο είναι υπεύθυνο για τη δημιουργία application programming interfaces (API). Django CMS, το οποίο έχει σχεδιαστεί για τη διαχείριση περιεχομένου

ιστότοπου. και Django-allauth, το οποίο είναι ένα ολοκληρωμένο σύνολο εφαρμογών Django για έλεγχο ταυτότητας, εγγραφή, διαχείριση λογαριασμού και έλεγχο ταυτότητας λογαριασμού τρίτων (κοινωνικών).

Το Django είναι μέρος πολλών επιτυχημένων ιστότοπων και εφαρμογών που χρησιμοποιούνται από εκατομμύρια χρήστες, όπως:

- EventBrite
- Bitbucket
- Pinterest
- Instagram
- The mozilla support site
- Disqus
- Prezi
- The Washington Post

2.4.3 Django REST

Είναι ένα ισχυρό και ευέλικτο σετ εργαλείων που καθιστά εύκολη τη δημιουργία REST API.

- Browsable api
- Παραμετροποιήσιμο
- Χρήση γλώσσας Python
- Εύκολη εφαρμογή
- Εύχρηστο
- Γρήγορη ανάπτυξη
- Χρήση του Django ORM
- Πολύ καλό documentation
- Γρήγορη μετάφραση μορφοποίησης δεδομένων

2.4.4 MongoDB

Η MongoDB είναι μια βάση δεδομένων εγγράφων ανοιχτού κώδικα που βασίζεται σε μια αρχιτεκτονική scale-out που έχει γίνει δημοφιλής στους προγραμματιστές όλων των ειδών που δημιουργούν επεκτάσιμες εφαρμογές χρησιμοποιώντας ευέλικτες μεθοδολογίες. Η MongoDB είναι κατάλληλη για τη δημιουργία διαδικτυακών και επιχειρηματικών εφαρμογών που πρέπει να μπορούν να εξελιχθούν γρήγορα και κομψά. Η MongoDB είναι ο πρωτοπόρος αυτού που αποκαλείται βάσεις δεδομένων NoSQL, οι οποίες αναπτύχθηκαν επειδή τα συστήματα RDBMS (relational database management system) που βασίζονται σε SQL δεν υποστηρίζουν την κλίμακα ή την ανάγκη ταχείας ανάπτυξης που απαιτείται για τη δημιουργία σύγχρονων εφαρμογών.

Περιγραφικά κάποια από τα προτερήματα της Mongo:

- Το μοντέλο δεδομένων με έγγραφα είναι ένας ισχυρός τρόπος αποθήκευσης και ανάκτησης δεδομένων που επιτρέπει στους προγραμματιστές να κινούνται γρήγορα. Η οριζόντια scale-out αρχιτεκτονική του MongoDB μπορεί να υποστηρίξει τεράστιους όγκους δεδομένων και επισκεψιμότητας.

- Η Mongo έχει να δώσει μια έξοχη πρώτη εμπειρία χρήσης στους προγραμματιστές που με το που την εγκαταστήσουν μπορούν να αρχίζουν να γράφουν κώδικα άμεσα.

- Η Mongo μπορεί να χρησιμοποιηθεί παντού από όλους:
  Δωρεάν μέσο του Community edition.
  Στα μεγαλύτερα κέντρα δεδομένων με την enterprise edition.
  Στα δημόσια Clouds μέσω της MongoDB Atlas.

- Η MongoDB έχει αναπτύξει ένα μεγάλο και ώριμο οικοσύστημα, που σημαίνει:
  Η MongoDB έχει μια παγκόσμια κοινότητα προγραμματιστών και συμβούλων, επομένως είναι εύκολο να βρεθεί βοήθεια.
  Η MongoDB λειτουργεί σε όλους τους τύπους υπολογιστικών πλατφορμών, τόσο σε εγκαταστάσεις όσο και στο cloud (τόσο σε ιδιωτικά όσο και δημόσια clouds όπως AWS, Azure και Google Cloud).
  Η MongoDB μπορεί να χρησιμοποιηθεί από όλες τις μεγάλες γλώσσες προγραμματισμού.
  Μπορείτε να έχετε πρόσβαση στη MongoDB από όλα τα μεγάλα συστήματα διαχείρισης. δεδομένων και ETL (extract, transform, load).
  Το MongoDB έχει υποστήριξη επαγγελματικού επιπέδου.

Οι βάσεις δεδομένων εγγράφων είναι εξαιρετικά ευέλικτες, επιτρέποντας παραλλαγές στη δομή των εγγράφων και επιτρέποντας την αποθήκευση εγγράφων που είναι εν μέρει πλήρεις. Ένα έγγραφο μπορεί να έχει άλλα ενσωματωμένα σε αυτό. Τα πεδία σε ένα έγγραφο παίζουν το ρόλο των στηλών σε μια βάση δεδομένων SQL, και όπως οι στήλες, μπορούν να ευρετηριαστούν για να αυξήσουν την απόδοση αναζήτησης. Η δομή των πληροφοριών βρίσκεται υπό τον έλεγχο του προγραμματιστή. Οι προγραμματιστές προσαρμόζουν και αναδιαμορφώνουν τη βάση δεδομένων καθώς η εφαρμογή εξελίσσεται χωρίς τη βοήθεια διαχειριστή βάσης δεδομένων. Όταν απαιτείται, το MongoDB μπορεί να συντονίσει και να ελέγξει τις αλλαγές στη δομή των εγγράφων χρησιμοποιώντας την

επικύρωση σχήματος (schema validation). Μερικοί λένε ότι αυτό μετατρέπει τα δεδομένα σε κώδικα. Η MongoDB δημιούργησε τη μορφή Binary JSON (BSON) για να αυξήσει την αποδοτικότητα και να υποστηρίξει περισσότερους τύπους δεδομένων. Τα δεδομένα που αποθηκεύονται στο BSON μπορούν να αναζητηθούν και να ευρετηριαστούν, αυξάνοντας σημαντικά την απόδοση. Το MongoDB υποστηρίζει μια μεγάλη ποικιλία μεθόδων ευρετηρίασης, όπως κείμενο, δεκαδικό, γεωχωρικό και μερικό. Προστέθηκαν ετικέτες γεωγραφική χωροθέτηση έτσι ώστε τα έγγραφα να μπορούν να ταξινομηθούν ανά τοποθεσία. Η MongoDB βασίστηκε στην αρχιτεκτονική scale-out, μια δομή που επιτρέπει σε πολλά μικρά μηχανήματα να συνεργάζονται για τη δημιουργία συστημάτων που είναι γρήγορα και διαχειρίζονται τεράστιες ποσότητες δεδομένων. Η Mongo επικεντρώθηκε στο να παρέχει μια εξαιρετική εμπειρία χρήστη στους προγραμματιστές, η οποία, εκτός από όλες τις άλλες ιδιότητές της, έκανε τη MongoDB ένα από τα αγαπημένα εργαλεία των προγραμματιστών παγκοσμίως για μια τεράστια ποικιλία εφαρμογών. Το MongoDB Compass προσφέρει ένα GUI (γραφικό περιβάλλον) για όσους προτιμούν μια οπτική διεπαφή. Το MongoDB Compass προσφέρει έναν τρόπο οπτικοποίησης των δεδομένων σας, δημιουργίας ευρετηρίων και συναρμολόγησης σύνθετων αγωγών συγκέντρωσης που βελτιστοποιούν τον τρόπο με τον οποίο εργάζεστε με δεδομένα.

Μερικά από τα προβλήματα που επιλύει το MongoDB:

- Ενσωμάτωση μεγάλων ποσοτήτων διαφορετικών δεδομένων: Εάν συγκεντρώνετε δεκάδες ή εκατοντάδες πηγές δεδομένων, η ευελιξία και η ισχύς του μοντέλου εγγράφου μπορεί να δημιουργήσει μια ενοποιημένη ενιαία προβολή με τρόπους που άλλες βάσεις δεδομένων δεν μπορούν. Το MongoDB πέτυχε να αναβιώσει τέτοια έργα όταν οι προσεγγίσεις που χρησιμοποιούν άλλες βάσεις δεδομένων απέτυχαν.

- Περιγράφοντας σύνθετες δομές δεδομένων που εξελίσσονται: Οι βάσεις δεδομένων εγγράφων επιτρέπουν την ενσωμάτωση εγγράφων για την περιγραφή ένθετων δομών και ανέχονται εύκολα παραλλαγές δεδομένων σε γενιές εγγράφων. Υποστηρίζονται αποτελεσματικά εξειδικευμένες μορφές δεδομένων όπως το γεωχωρικό. Αυτό έχει ως αποτέλεσμα ένα αποθετήριο που είναι ανθεκτικό και δεν σπάει ή πρέπει να επανασχεδιαστεί κάθε φορά που κάτι αλλάζει.

- Παράδοση δεδομένων σε εφαρμογές υψηλής απόδοσης: Η αρχιτεκτονική scale-out της MongoDB μπορεί να υποστηρίξει τεράστιο αριθμό συναλλαγών σε τεράστιες βάσεις δεδομένων. Σε αντίθεση με άλλες βάσεις δεδομένων που είτε δεν μπορούν να υποστηρίξουν τέτοια κλίμακα ή μπορούν να το κάνουν μόνο με τεράστιες ποσότητες εφαρμοσμένης μηχανικής και πρόσθετων στοιχείων, το MongoDB έχει μια σαφή πορεία προς την επεκτασιμότητα λόγω του τρόπου με τον οποίο σχεδιάστηκε. Το MongoDB είναι κατασκευαστηκά επεκτάσιμο.

- Υποστήριξη υβριδικών εφαρμογών και εφαρμογών πολλαπλών σύννεφων (multi-cloud): Η MongoDB μπορεί να αναπτύξει και να εκτελέσει σε επιτραπέζιο υπολογιστή, σε ένα τεράστιο σύμπλεγμα υπολογιστών σε ένα κέντρο δεδομένων ή

σε ένα δημόσιο cloud, είτε ως εγκατεστημένο λογισμικό είτε μέσω του MongoDB Atlas, μια βάση δεδομένων ως υπηρεσία.

- Υποστήριξη ευέλικτης ανάπτυξης και συνεργασίας: Οι βάσεις δεδομένων εγγράφων θέτουν τους προγραμματιστές υπεύθυνους για τα δεδομένων. Τα δεδομένα ,μετατρέπονται σε κώδικα φιλικό προς τους προγραμματιστές. Αυτό διαφέρει πολύ από άλλες βάσεις δεδομένων που αναγκάζουν τους προγραμματιστές να χρησιμοποιούν ένα παράξενο σύστημα που απαιτείται ειδικός για την κατανόηση του. Οι βάσεις δεδομένων εγγράφων επιτρέπουν επίσης την εξέλιξη της δομής των δεδομένων καθώς οι ανάγκες είναι καλύτερα κατανοητές. Η συνεργασία και η διακυβέρνηση μπορούν να πραγματοποιηθούν επιτρέποντας σε μια ομάδα να ελέγχει ένα μέρος ενός εγγράφου και μια άλλη ομάδα να ελέγχει ένα άλλο μέρος.

Το όνομα Mongo είναι κομμάτι της λέξης humongous (τεράστιος). Η βάση δεδομένων έχει τις ρίζες της στις απογοητεύσεις των Dwight Merriman, Eliot Horowitz και Kevin Ryan, οι οποίοι συν-ίδρυσαν τη MongoDB στη Νέα Υόρκη το 2007, αφού προσπάθησαν να δημιουργήσουν κλιμακούμενες εφαρμογές Ιστού για το DoubleClick, έναν από τους πρωτοπόρους στη τεχνολογίας ψηφιακής διαφήμισης που τελικά έγινε μέρος της Google. Οι ιδρυτές ήθελαν να δημιουργήσουν μια βάση δεδομένων που θα απευθύνονταν στους προγραμματιστές, μια βάση δεδομένων που θα διέσχιζε τα εμπόδια στα Συστήματα Διαχείρισης Βάσεων Δεδομένων (RDBMS) που χρησιμοποιούν τη γλώσσα ερωτημάτων SQL.

Χιλιάδες εταιρείες όπως η Bosch, η Barclays και η Morgan Stanley διευθύνουν τις επιχειρήσεις τους στη MongoDB και τη χρησιμοποιούν για να χειριστούν τις πιο απαιτητικές εφαρμογές τους σε τομείς όπως IoT, Gaming, Logistics, Banking, e-Commerce και Content Management.

2.4.5 React.js

Από την κυκλοφορία του React, έχουμε δει μια εκρηκτική ανάπτυξη στη χρήση ελαφριών αλλά ισχυρών βιβλιοθηκών JavaScript. Οι χρήστες θέλουν όλο και συχνότερα να χρησιμοποιούν γρηγορότερες, πιο δυναμικές ιστοσελίδες, ενώ οι προγραμματιστές επιλέγουν μοντέρνα και ευέλικτα περιβάλλοντα χωρίς περιττό περιεχόμενο στο πακέτο. Γι 'αυτό το ReactJS είναι μια προφανής επιλογή για πολλούς. Για να εξηγήσουμε γιατί, ας δούμε τους κορυφαίους λόγους για τους οποίους χρησιμοποιούμε το React.

Απλότητα: Το πρώτο πράγμα που κάνει πολλούς ανθρώπους να χρησιμοποιούν το ReactJS στα έργα τους είναι η απλότητά του. Το React είναι μια βιβλιοθήκη JavaScript, οπότε αν ένας προγραμματιστής είναι εξοικειωμένος με τις λειτουργίες της, θα εξοικειωθεί γρηγορότερα και ευκολότερα με το ReactJS. Με αυτήν τη βιβλιοθήκη, οι προγραμματιστές ορίζουν διεπαφές με σύνταξη τύπου HTML που ονομάζεται JSX. Ως αποτέλεσμα, παράγεται κώδικας HTML και CSS. Το API του React είναι πολύ μικρό, αλλά ισχυρό και το μόνο που πρέπει να κάνετε

πριν ξεκινήσετε είναι να μάθετε μερικές βασικές λειτουργίες.

```
const AppHeaderBlock = () => (

  <div>

    <brandedheader>Inventory</brandedheader><ul classname="menu"><li classname="nav-item">

      <button onclick="{gotoHome}">Home</button>

    </li>

    <li classname="”nav-item”"">

      <button onclick="{gotoAccount}">Account</button>

    </li>

  </ul></div>

)
```

Το παράδειγμα ενός component React, που εφαρμόζεται στο JSX.
Εμφανίζεται μια μικρή καμπύλη εκμάθησης όταν υπάρχει ανάγκη χρήσης του React με άλλες βιβλιοθήκες JS, όπως Redux, Material UI ή Enzyme. Αν και δεν αποτελούν μέρος του React stack, τέτοιες βιβλιοθήκες προσθέτουν επιπλέον δυνατότητες και σας επιτρέπουν να διαχειρίζεστε τα components του React πιο εύκολα. Οι πιο συνηθισμένες βιβλιοθήκες είναι καλά τεκμηριωμένες και δεν πρέπει να προκαλούν προβλήματα σε κανέναν προγραμματιστή. Η απλότητα του React συνοδεύεται από ορισμένα οφέλη για τις επιχειρήσεις. Όσοι δυσκολεύονται ήδη σε άλλα JavaScript frameworks θα έχουν ευκολότερη μετάβαση στο React. Ακόμα κι αν ο στόχος σας είναι απλώς να δοκιμάσετε τη βιβλιοθήκη, χωρίς να δεσμευτείτε σε μια πλήρη μετάβαση, αυτό επίσης δεν θα προκαλέσει δυσκολία. Και αν πρέπει να κλιμακώσετε στο μέλλον ή να αλλάξετε την ομάδα ανάπτυξης, η αφθονία των προγραμματιστών ReactJS στην αγορά θα κάνει τα πράγματα πολύ πιο εύκολα.

Η έκδηλη συμπάθεια από τους προγραμματιστές: Η πρόσφατη έρευνα που πραγματοποιήθηκε από το StateofJS.com δείχνει ότι οι προγραμματιστές JavaScript είναι πολύ ευχαριστημένοι με τη χρήση του React στα έργα τους, συγκρίνοντάς το ευνοϊκά με τους πλησιέστερους ανταγωνιστές του, όπως το Angular ή το Vue.js. Το παραπάνω είναι ζωτικής σημασίας για όποιον σχεδιάζει να δημιουργήσει οποιοδήποτε προϊόν λογισμικού για τα επόμενα χρόνια. Εάν οι προγραμματιστές είναι τόσο ενθουσιασμένοι για το React, αυτό σημαίνει ότι θα υπάρχουν όλο και περισσότεροι από αυτούς εκεί έξω. Αυτό, ως αποτέλεσμα, σημαίνει ότι θα υπάρξει μεγάλο οικοσύστημα διαφόρων βιβλιοθηκών και εργαλείων React για την ενίσχυση της ανάπτυξης. Ως αποτέλεσμα, το ReactJS θα γίνει πιο ισχυρό με κάθε προσθήκη. Θα υπάρχουν επίσης πολλά ταλέντα στην αγορά, έτοιμα να χτίσουν νέα έργα και να υποστηρίξουν υπάρχοντα. Το ReactJS είναι εδώ για να μείνει.

Επαναχρησιμοποιήσιμα στοιχεία: Ένας από τους σημαντικότερους λόγους χρήσης του React. Εάν έχετε χρησιμοποιήσει ποτέ κάποια εργαλεία σχεδίασης, όπως το Sketch, πρέπει να

γνωρίζετε πόσο εύκολα και διαισθητικά είναι τα στοιχεία διεπαφής. Σχεδιάζετε ένα ορθογώνιο και αλλάζετε το μέγεθος σύμφωνα με τις προτιμήσεις σας. Στη συνέχεια εισάγετε κάποιο αντίγραφο μέσα και ομαδοποιήστε το ως «πεδίο κειμένου». Στη συνέχεια, αντιγράφετε την ομάδα αρκετές φορές, σχεδιάζετε ένα κουμπί που τραβάει την προσοχή με αντίγραφο και τυλίξτε τα όλα με ένα πλαίσιο και μια κεφαλίδα. Στη συνέχεια, τα ομαδοποιείτε όλα ως «φόρμα επικοινωνίας». Όποια επιχείρηση κι αν είναι αυτό το έργο, πιθανότατα θα επαναχρησιμοποιήσετε αυτήν τη φόρμα επικοινωνίας πολλές φορές καθ 'όλη τη διάρκεια του έργου. Δεν έχει νόημα να σχεδιάζετε τη φόρμα από το μηδέν κάθε φορά που χρειάζεστε ένα, έτσι προφανώς ένας αξιοπρεπής σχεδιαστής θα εισάγει ήδη ομαδοποιημένα στοιχεία με λίγα μόνο κλικ. Η ανάπτυξη ήταν πολύ πιο περίπλοκη από αυτό, αλλά το ReactJS δίνει τα εργαλεία για να κάνουμε ακριβώς το ίδιο πράγμα. Κάθε έργο React κατασκευάζεται χρησιμοποιώντας τα λεγόμενα επαναχρησιμοποιήσιμα στοιχεία (components). Αυτό σημαίνει ότι κάθε στοιχείο της διεπαφής που έχετε ήδη δημιουργήσει, μπορεί να χρησιμοποιηθεί οπουδήποτε στο έργο σας, κάνοντας απλώς κλήση από άλλα στοιχεία. Γράφετε μια τέτοια συνάρτηση μία φορά και μπορείτε να την χρησιμοποιήσετε ξανά οπουδήποτε. Επιπλέον, μπορείτε να ομαδοποιήσετε αυτά τα στοιχεία σε ενότητες ή σελίδες και να τα εισαγάγετε οπουδήποτε γύρω από το έργο με τις ίδιες λειτουργίες. Τα components στο ReactJS ακολουθούν επίσης ένα απλό πνευματικό μοντέλο. Εάν ένα στοιχείο έχει τα ίδια δεδομένα πολλές φορές, θα αποδίδει πάντα το ίδιο σύνολο HTML και CSS. Αυτή η ικανότητα, που ονομάζεται idempotence, καθιστά εύκολο να συλλογιστεί κανείς και να δοκιμάσει.

Εξετάστε αυτό το κοινό παράδειγμα: όλα τα κουμπιά στην εφαρμογή σας πρέπει να έχουν τον ίδιο τρόπο. Στο ReactJS, μπορείτε να ορίσετε ένα στοιχείο και να το χρησιμοποιήσετε ξανά οπουδήποτε αντί να επικολλάτε ξανά τον ίδιο κώδικα ξανά και ξανά.

```
const BrandedButton = ({icon, title, onClick}) => (

  <button classname="branded-button" onclick="{onClick}">

    <img src="%7Bicon%7D" classname=""icon”" data-pagespeed-url-hash="3997312273" onload="pagespeed.CriticalImages.checkImageForCriticality(this);">

    {title}

  </button>

)

// Από εδώ και πέρα το χρησιμοποιούμε ως:

<brandedbutton icon=""login”" title=""Log" in="" onclick="{gotoLogin}"></brandedbutton>
```

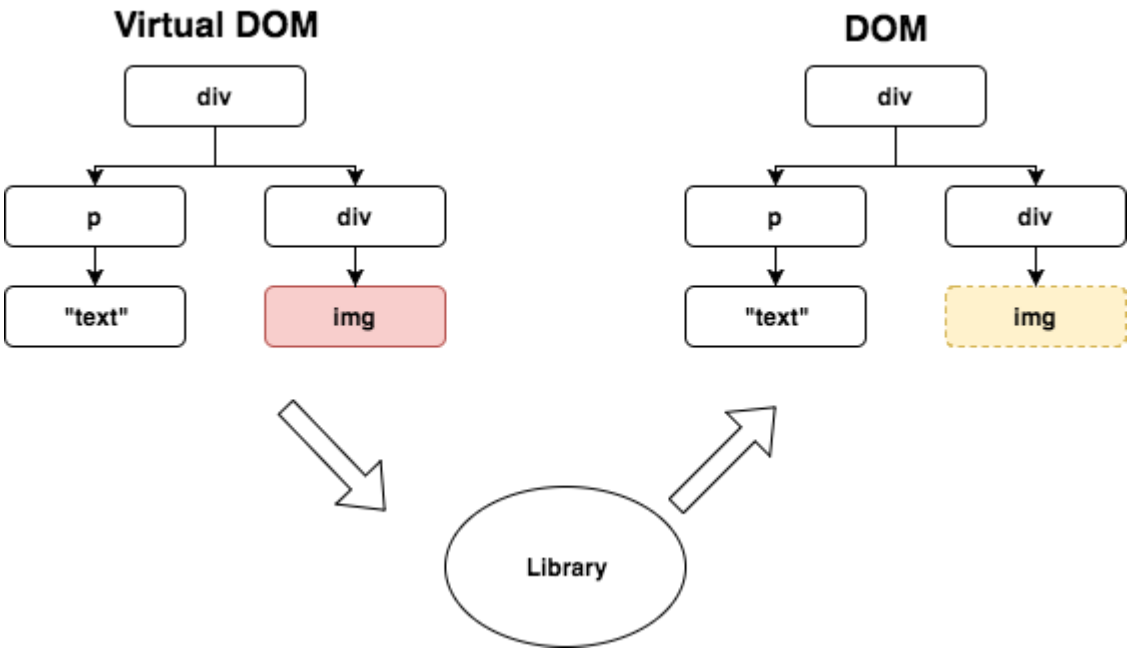
Εκτός από το ότι είναι ευκολότερο να γράψετε (και να διαβάσετε), αυτό το στοιχείο φέρνει ένα πολύ σημαντικό όφελος: η παρουσίασή του διαχωρίζεται από τη σημασιολογία του. Οι σχεδιαστές είναι ελεύθεροι να δημιουργήσουν μια βιβλιοθήκη στοιχείων χωρίς να ανησυχούν για τη λογική τους και οι μηχανικοί μπορούν να κατασκευάσουν διεπαφές από τα στοιχεία, γνωρίζοντας ότι θα φαίνονται καλά χωρίς καμία προσπάθεια. Όλα αυτά έρχονται με πολλά

πλεονεκτήματα για κάθε επιχείρηση. Χάρη στα επαναχρησιμοποιήσιμα στοιχεία, η πλατφόρμα θα είναι συνεπής και, ως εκ τούτου, εύκολη στην πλοήγηση για τους χρήστες. Υπάρχει λιγότερος κώδικας που πρέπει να γραφτεί χωρίς απώλεια λειτουργιών και αυτό αναπόφευκτα οδηγεί σε ταχύτερη ανάπτυξη. Είναι επίσης πιο απλό να προσθέσετε νέες λειτουργίες και να τις δοκιμάσετε καθώς έχετε ήδη γράψει αρκετές γραμμές κώδικα και είστε έτοιμοι να τις επαναχρησιμοποιήσετε.

Εύκολη ενημέρωση component, μεμονωμένα ή μαζικά: Το ReactJS επιτρέπει επίσης την γρήγορη αναπαραγωγή οποιουδήποτε component. Μέσα σε λίγα δευτερόλεπτα μπορείτε να τροποποιήσετε το χρώμα όλων των κουμπιών σε ολόκληρο τον ιστότοπο. Σε περίπτωση που επιθυμείτε να σταματήσετε την ανακατεύθυνση από κάθε μελέτη περίπτωσης σε μια παλιά σελίδα προορισμού, δίνεται η δυνατότητα να ανταλλάξετε όλους τους παλιούς συνδέσμους με τους νέους. Φυσικά, εάν θέλετε να τροποποιήσετε μόνο ένα ή μερικά στοιχεία, μπορείτε να το κάνετε χωρίς να επηρεάσετε άλλα στοιχεία του έργου σας. Η λογική πίσω από κάθε στοιχείο μπορεί να οριστεί από έναν προγραμματιστή μία φορά και το React θα το χρησιμοποιήσει για να εμφανίσει στοιχεία ακριβώς όπου θέλουμε να είναι, σε ολόκληρο το έργο. Για παράδειγμα, τα avatar μπορούν να έχουν τις ίδιες συμπεριφορές hover ή onClick σε ολόκληρο τον ιστότοπο. Μπορεί να μην φαίνεται μεγάλη διαφορά, αλλά για οποιονδήποτε είχε πέντε-δέκα διαφορετικούς τρόπους εμφάνισης των ειδώλων των χρηστών, είναι. Ένα άλλο παράδειγμα μπορεί να είναι η μορφοποίηση ημερομηνιών σε επιθυμητή μορφή. Όσο ένα μεμονωμένο στοιχείο χρησιμοποιείται για αυτό, η μορφή θα είναι συνεπής σε ολόκληρο το έργο. Η διατήρηση της βάσης κώδικα γίνεται πολύ πιο εύκολη.

Εικονικό DOM: Σε πολλά πλαίσια front-end, το DOM (Document Object Model) είναι ένα δέντρο σε σχήμα HTML που αντιπροσωπεύει όλα τα στοιχεία μιας εφαρμογής ιστού, ενσωματωμένη σε μια δομή γονέα-παιδιού. Κάθε φορά που υπάρχει αλλαγή στην κατάσταση οποιουδήποτε στοιχείου (για παράδειγμα, γίνεται κλικ σε ένα κουμπί ή υποβολή φόρμας), το DOM ενημερώνεται αμέσως. Με αυτόν τον τρόπο, αντιπροσωπεύει πάντα την τρέχουσα διεπαφή χρήστη ενός ιστότοπου. Ένας απλός χειρισμός ενός DOM είναι γρήγορος. Αλλά με κάθε αλλαγή, τόσο ο γονέας όσο και τα παιδιά του ενημερώνονται. Είναι εύκολο σε απλούς ιστότοπους, αλλά μόλις ένα έργο γίνει πολύπλοκο, προκαλεί πολύ συχνές ενημερώσεις σε πολλά στοιχεία, επιβραδύνοντας αισθητά τον ιστότοπο. Αυτό, ως αποτέλεσμα, δημιουργεί μια κακή εμπειρία χρήστη. Το ReactJS βελτιστοποιεί την προηγούμενη περίπτωση αξιοποιώντας το Virtual DOM για να επιταχύνει τους ιστότοπους. Το εικονικό DOM (vDOM), όπως υποδηλώνει το όνομα, δεν είναι πραγματικό DOM αλλά εικονική αναπαράστασή του. Κάθε φορά που τροποποιείται οποιοδήποτε στοιχείο ενός ιστότοπου, ενημερώνεται ένα vDOM. Στη συνέχεια, το ενημερωμένο VDOM συγκρίνεται με ένα πραγματικό DOM χρησιμοποιώντας έναν αλγόριθμο συμφιλίωσης από το React. Αυτή η διαδικασία χρησιμοποιείται για τον υπολογισμό του ελάχιστου συνόλου αλλαγών που θα εφαρμοστούν στο πραγματικό DOM. Αντί να ενημερώνει δεκάδες στοιχεία κάθε φορά που γίνεται μια αλλαγή, το ReactJS ενημερώνει μόνο το στοιχείο που μόλις τροποποιήθηκε, μειώνοντας σημαντικά τον χρόνο της λειτουργίας. Ως αποτέλεσμα, επιταχύνεται σημαντικά η σελίδα. Οι ταχύτερες σελίδες τείνουν να οδηγούν σε υψηλότερα ποσοστά μετατροπών και καλύτερη εμπειρία χρήστη. Αυτό, κατά συνέπεια, τείνει να φέρει υψηλότερες πωλήσεις,

περισσότερες εγγραφές και χαμηλότερα ποσοστά πτώσης (χρήστες να σταματάνε να χρησιμοποιούνε την εφαρμογή).



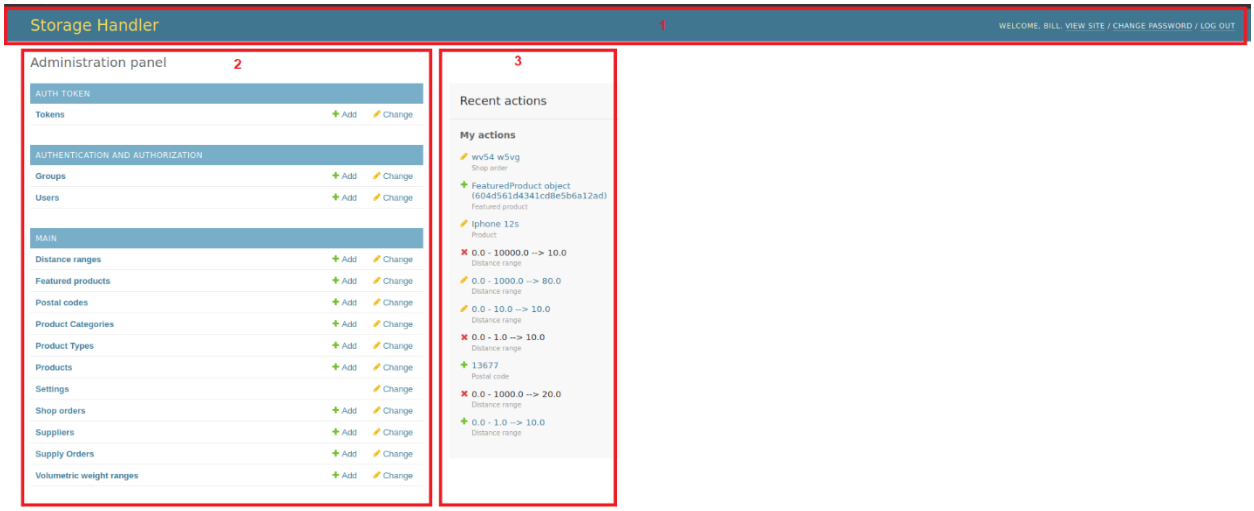
React εφαρμογές για υπολογιστή και κινητά: Μια επαναστατική δυνατότητα που συνοδεύει το React είναι η δυνατότητα δημιουργίας εφαρμογών για κινητά iOS και Android, χωρίς την ανάγκη προσλήψεων νέων προγραμματιστών. Οι εφαρμογές που έχουν χτιστεί με React μπορούν να χρησιμοποιηθούν και από κινητά, με την χρήση του React Native, υπάρχει επιλογή να γραφτούν πλήρως λειτουργικές εφαρμογές για κινητά γραμμένες σε Javascript. Αυτό επιτρέπει στους προγραμματιστές να επαναχρησιμοποιήσουν κώδικα από την εφαρμογή ιστού στο ReactJS για να επιταχύνουν την ανάπτυξη εφαρμογών σε κινητά. Τα πάντα εκτός από την παρουσίαση μπορούν να μεταφερθούν στην εφαρμογή για κινητά. Ένας προγραμματιστής του ReactJS μπορεί επίσης να χρησιμοποιήσει διάφορες βιβλιοθήκες JS με τις οποίες είναι ήδη εξοικειωμένοι. Μόλις μια κοινόχρηστη βάση κώδικα είναι έτοιμη, ένας προγραμματιστής αρχίζει να δημιουργεί στοιχεία iOS ή Android. Παρά τις διαφορές μεταξύ των δύο πλατφορμών (κινητό τηλέφωνο και υπολογιστής), ένας καλός προγραμματιστής του React μπορεί να χειριστεί όλες τις πλατφόρμες μετά από λίγη έρευνα.

3 ΚΕΦΑΛΑΙΟ 3ο : Περιγραφή UI

Σε αυτό το κεφάλαιο θα περιγραφεί το user interface (η διεπαφή της εφαρμογής με τον χρήστη) καθώς και ο προτεινόμενος τρόπος χειρισμού της εφαρμογής με χρήση φωτογραφιών.

3.1 Περιγραφή UI εφαρμογής διαχείρισης της αποθήκης

Περιγραφή αρχικής σελίδας:

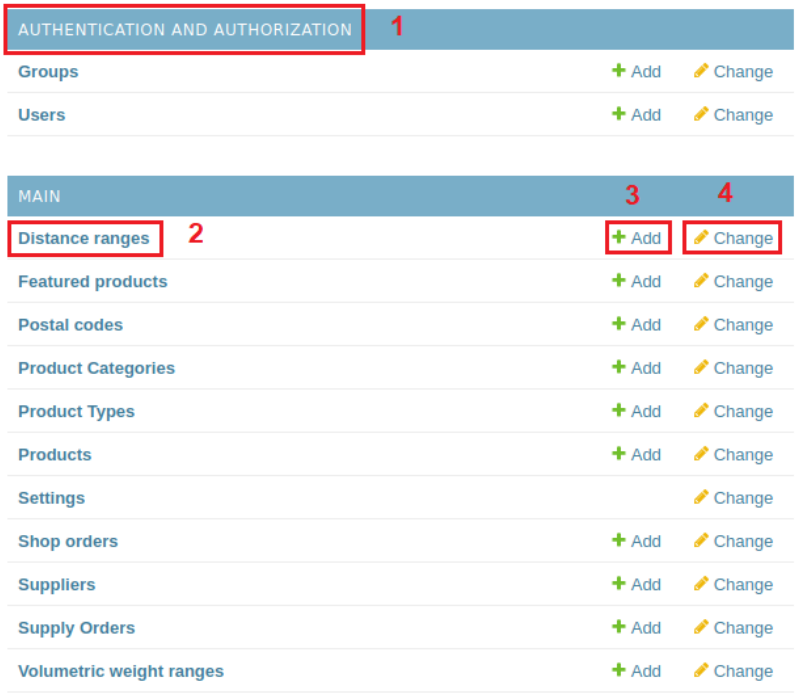


Η αρχική σελίδα χωρίζεται σε τρία μέρη:

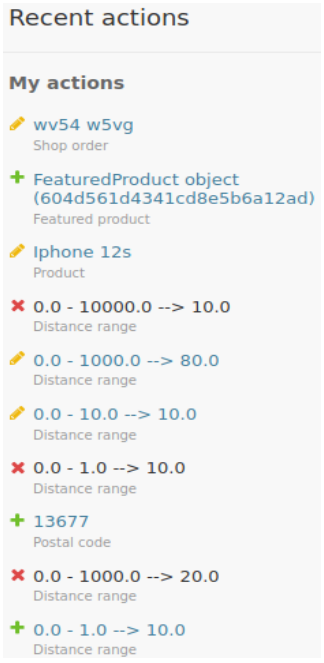
1. Την κεφαλίδα header.
2. Τον πίνακα με τα περιεχόμενα.
3. Τον πίνακα προσωπικού ιστορικού κινήσεων.

Η κεφαλίδα αποτελείται από τα εξής:

1. Το όνομα της εφαρμογής, λειτουργεί και ως κουμπί ανακατεύθυνσης στην αρχική σελίδα.
2. Μήνυμα χαιρετισμού και παρουσίαση του username του χρήστη που είναι συνδεδεμένος.
3. Κουμπί για αλλαγή κωδικού πρόσβασης. Ο χρήστης κάνοντας κλικ σε αυτό το κουμπί θα μεταφερθεί σε σελίδα που θα του ζητάει τον παλιό κωδικό και τον κωδικό που θέλει να εισάγει.
4. Κουμπί αποσύνδεσης.

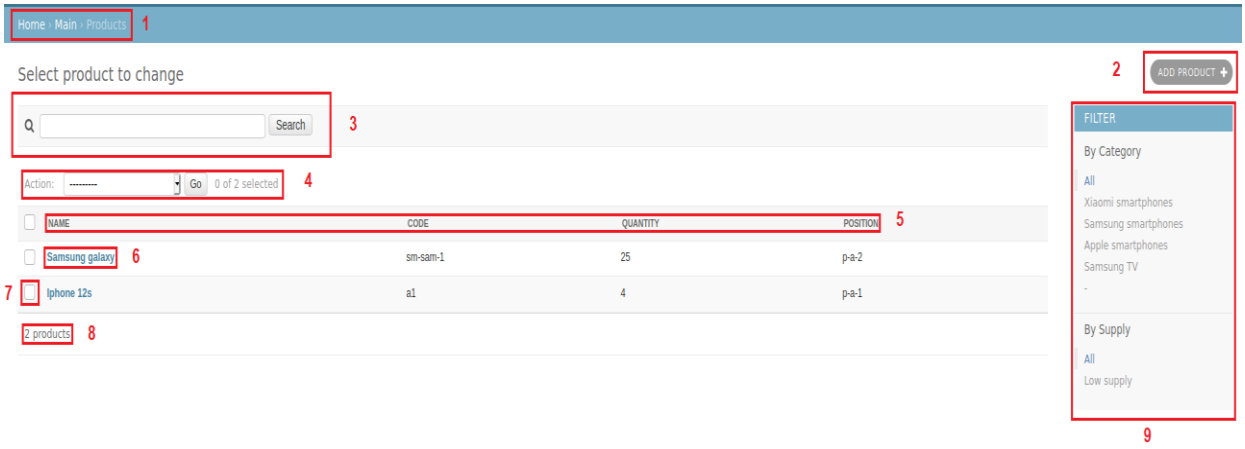


1. Κεφαλίδα περιγραφής περιεχομένου
2. Όνομα μοντέλου. Λειτουργεί και ως κουμπί μεταφοράς του χρήστη στην σελίδα του αναγραφόμενου μοντέλου.
3. Κουμπί πρόσθεσης αντικειμένου. Πατώντας το ο χρήστης θα κληθεί να συμπληρώσει τα απαραίτητα πεδία για να εισάγει το καινούργιο αντικείμενο.
4. Κουμπί επεξεργασίας των αντικειμένων που βρίσκονται σε αυτή την κατηγορία.



Πίνακας ιστορικού, όπου παρουσιάζει τις τελευταίες αλλαγές που έχει κάνει ο χρήστης στην εφαρμογή. Ο χρήστης μπορεί να πατήσει πάνω σε κάθε αλλαγή που έχει κάνει και θα μεταφερθεί στη σελίδα του αντικειμένου που άλλαξε (δεν λειτουργεί για διαγραμμένα αντικείμενα).

Πατώντας πάνω σε κάθε μοντέλο (κατηγορία) ο χρήστης θα μεταφερθεί στη σελίδα της κατηγορίας που πάτησε. Οι σελίδες των μοντέλων μοιάζουν αρκετά μεταξύ τους με ελάχιστες διαφορές.
Μια σελίδα μοντέλου για παράδειγμα της κατηγορίας προϊόντων είναι η εξής.

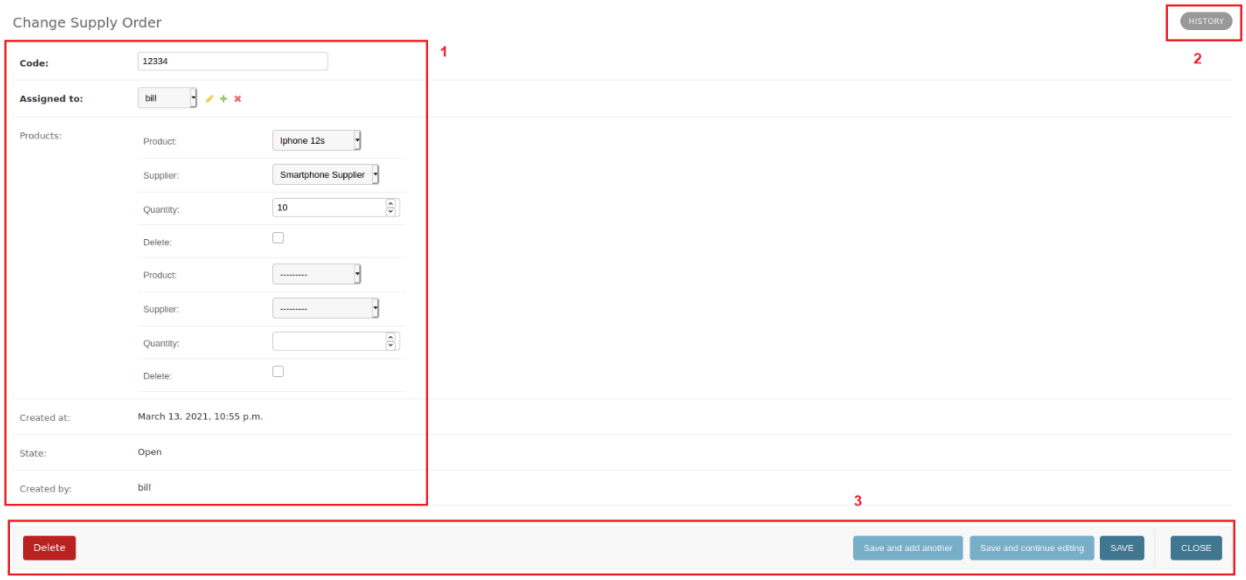


1. Μονοπάτι που ακολούθησε ο χρήστης για να βρεθεί σε αυτή τη σελίδα. Λειτουργεί και σαν κουμπί το οποίο μεταφέρει τον χρήστη στην αναγραφόμενη σελίδα.
2. Κουμπί πρόσθεσης αντικειμένου. Πατώντας το ο χρήστης θα κληθεί να συμπληρώσει τα απαραίτητα πεδία για να εισάγει ένα καινούργιο αντικείμενο σε αυτή την κατηγορία.
3. Κάποιες σελίδες θα περιέχουν μπάρα αναζήτησης. Ο χρήστης μπορεί να πληκτρολογήσει το όνομα ή τον κωδικό του αντικειμένου που αναζητά, να πατήσει το κουμπί search και να του παρουσιαστούν τα αποτελέσματα τα οποία είναι παρόμοια με την λέξη που έγραψε.
4. Επιλογέας επιθυμητής ενεργείας. Πατώντας το κουμπί <<drop down>> μας εμφανίζονται οι ενέργειες που μπορούν να εφαρμοστούν στα επιλεγμένα αντικείμενα. Μια από της ενέργειες είναι η διαγραφή. Αφού έχει γίνει η επιλογή ο χρήστης μπορεί να πατήσει το κουμπί <<Go>> για να ολοκληρώσει την ενέργεια.
5. Μπάρα εμφάνισης ονόματος πεδίου. Για την διευκόλυνση των χρηστών το κάθε αντικείμενο παρουσιάζεται μαζί με κάποια βασικά στοιχεία του. Αυτή η μπάρα περιγράφει αυτά τα στοιχεία.
6. Κύρια ταμπέλα αντικειμένου. Πατώντας πάνω στο όνομα του αντικειμένου ο χρήστης θα μεταφερθεί στη σελίδα που βρίσκονται όλες οι πληροφορίες για το συγκεκριμένο αντικείμενο.
7. Κουτί επιλογής. Αν ο χρήστης επιθυμεί να δράσει πάνω σε πολλαπλά αντικείμενα αρκεί να σημειώσει στο κουτί επιλογής που βρίσκεται δίπλα από το κάθε ένα και να

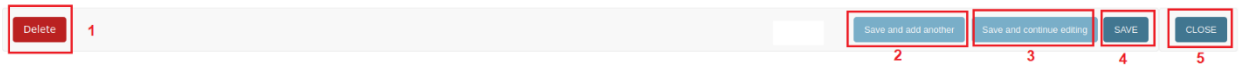
επιλέξει τον τρόπο με τον οποίο θέλει να ενεργήσει χρησιμοποιώντας τον επιλογέα επιθυμητής ενέργειας.

8. Στο τέλος των αντικειμένων υπάρχει ένας αριθμός που φανερώνει πόσα αντικείμενα παρουσιάζονται σε αυτήν την σελίδα.
9. Μενού φίλτρων. Σε ορισμένες σελίδες θα υπάρχουν φίλτρα αναζήτησης. Τα φίλτρα θα φέρουν ένδειξη του πεδίου στο οποίο δρουν και από κάτω τις επιλογές φιλτραρίσματος. Πατώντας πάνω τους επιλέγονται.

Το κάθε αντικείμενο έχει δικιά του σελίδα με όλα του τα στοιχεία και με τρόπους δράσεις πάνω σε αυτό. Η δομή της σελίδας είναι η ίδια για όλα τα αντικείμενα με ελάχιστες διαφορές.



1. Φόρμα παρουσίασης και επεξεργασίας των στοιχείων του αντικειμένου. Εδώ οι εξουσιοδοτημένοι χρήστες μπορούν να αναζητήσουν ή να αλλάξουν στοιχεία του αντικειμένου.
2. Κουμπί ιστορικού. Πατώντας αυτό το κουμπί ο χρήστης θα μεταφερθεί στη σελίδα ιστορικού του αντικειμένου, εκεί βρίσκονται πληροφορίες σχετικά με τις αλλαγές που έχει υποστεί το αντικείμενο και ποιος τις έχει εκτελέσει.
3. Μπάρα ενεργειών αντικειμένου. Σε αυτή τη περιοχή της σελίδας υπάρχουν κουμπιά ανάλογα με τις ενέργειες που επιτρέπονται στο αντικείμενο.



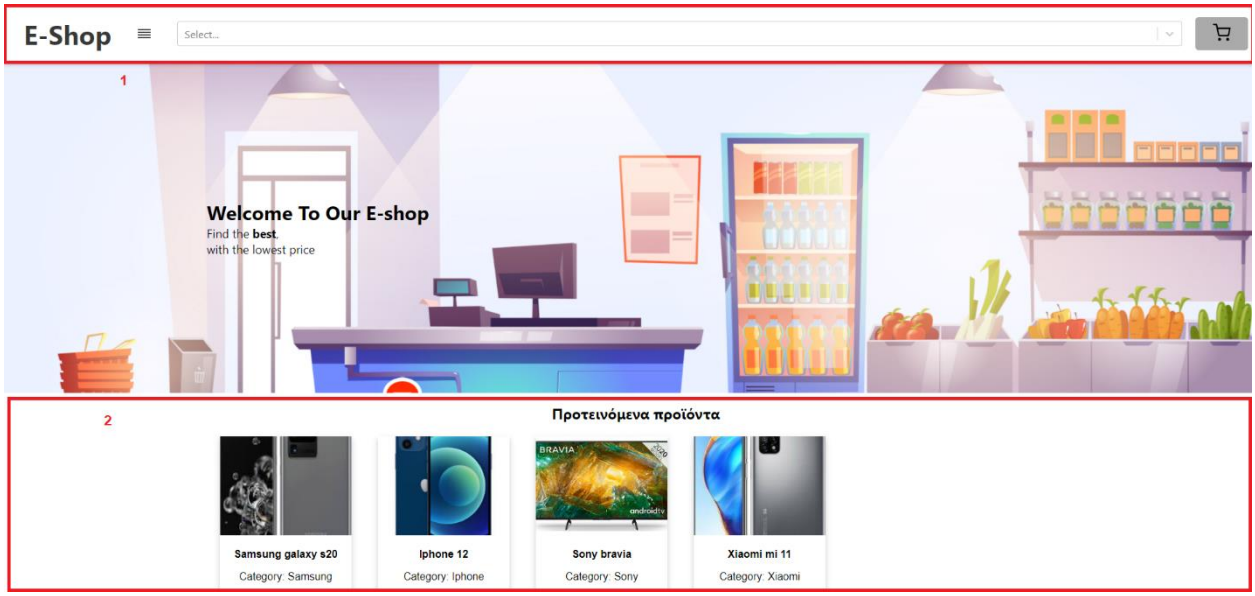
1. Κουμπί διαγραφής. Διαγράφει το αντικείμενο.
2. Κουμπί αποθήκευσης και πρόσθεσης και άλλου αντικειμένου. Πατώντας αυτό το κουμπί ο χρήστης αποθηκεύει το αντικείμενο που έχει δημιουργήσει και μεταφέρεται ξανά στη σελίδα δημιουργίας.
3. Κουμπί αποθήκευσης και επεξεργασίας. Πατώντας αυτό το κουμπί ο χρήστης αποθηκεύει το αντικείμενο που έχει δημιουργήσει και παραμένει στη σελίδα του.

Αυτό το κουμπί χρησιμοποιείται για προσθήκη υπό αντικειμένων στο αντικείμενο (για παράδειγμα προσθήκη προϊόντων σε μια παραγγελία).

4. Κουμπί αποθήκευσης. Πατώντας το αποθηκεύεται το αντικείμενο που δημιούργησε ο χρήστης και μεταφέρεται στη σελίδα κατηγορίας (μοντέλου) αντικειμένου.
5. Κουμπί ολοκλήρωσης παραγγελίας. Κάποια αντικείμενα έχουν παραπάνω λειτουργίες από άλλα. Στο αντικείμενο παραγγελίες αποθήκης ο εξουσιοδοτημένος χρήστης αφού φέρει εις πέρας την παραγγελία πατάει αυτό το κουμπί για να κλείσει την παραγγελία.

3.2 Περιγραφή UI εφαρμογής ηλεκτρονικού καταστήματος

Περιγραφή αρχικής σελίδας:

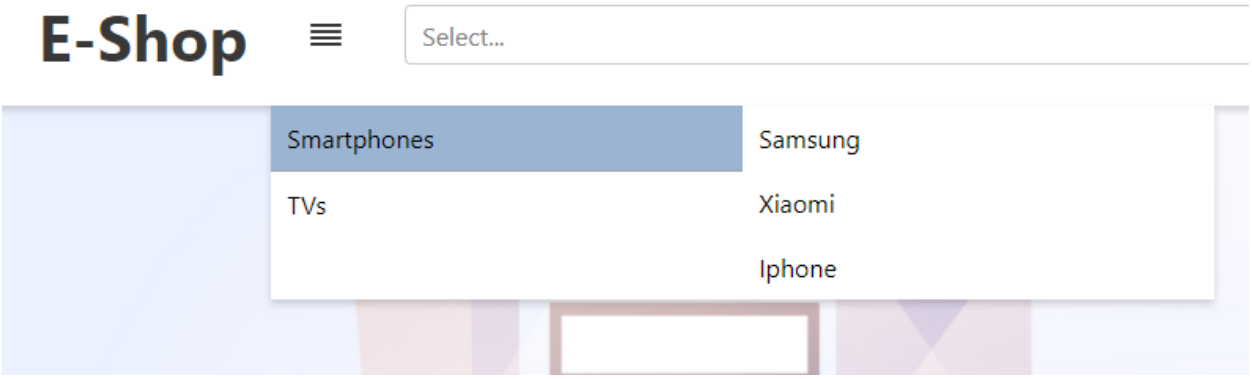


Η αρχική σελίδα χωρίζεται σε δύο μέρη:

1. Την κεφαλίδα header.
2. Την περιοχή προβολής των <<featured>> προϊόντων.

Η κεφαλίδα περιέχει τα εξής:

1. Λογότυπο της σελίδας το οποίο δρα και ως κουμπί μεταφοράς του χρήστη στην αρχική σελίδα.
2. Κουμπί περιήγησης. Πατώντας το θα παρουσιαστεί στον χρήστη ένα drop down menu με τους τύπους των προϊόντων της αποθήκης. Επιλέγοντας ένα τύπο με κλικ στα δεξιά του μενού θα παρουσιαστεί άλλο ένα παρόμοιο εμφανίζοντας τις κατηγορίες που περιέχει ο τύπος που διάλεξε ο χρήστης. Το drop down menu έχει την εξής μορφή:

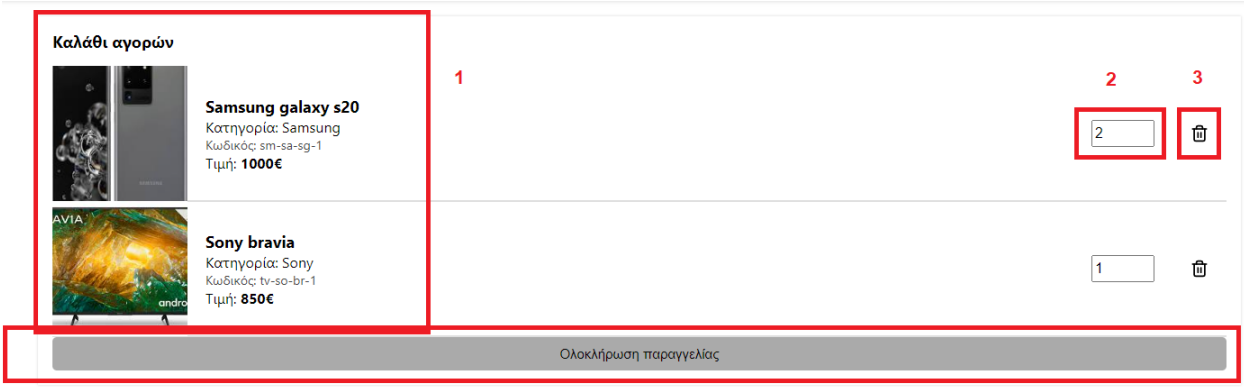


3. Μπάρα αναζήτησης. Ο χρήστης μπορεί να γράψει το όνομα ή τον κωδικό του προϊόντος που θέλει να βρει και η μπάρα θα του εμφανίσει το προϊόν που αναζητά η αυτά με παρόμοιο όνομα η κωδικό.
4. Καλάθι αγορών. Πατώντας αυτό το κουμπί ο χρήστης θα μεταφερθεί στο καλάθι με τα προϊόντα που έχει διαλέξει για αγορά.

Τα προϊόντα παρουσιάζονται με δύο μορφές:

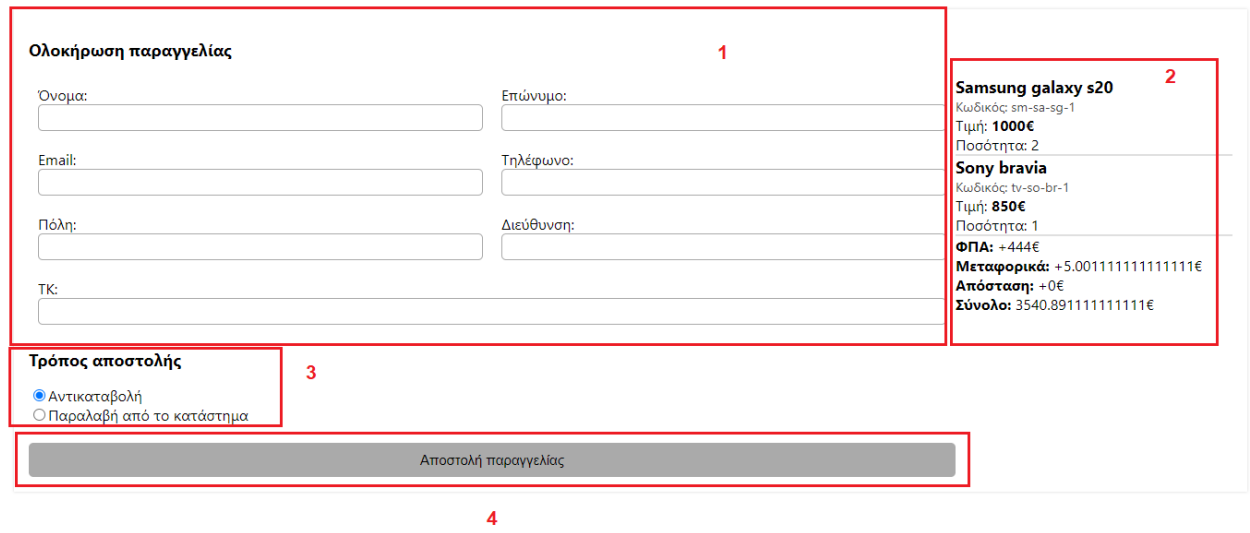
1. Η σύντομη παρουσίαση. Τα προϊόντα στις κατηγορίες παρουσιάζονται στην σύντομη μορφή τους όπου περιλαμβάνει κομμάτι της φωτογραφίας, το όνομα του προϊόντος και την κατηγορία στην οποία ανήκει.
2. Η περιγραφική παρουσίαση. Όταν επιλεγεί ένα προϊόν ο χρήστης μεταφέρεται στην σελίδα αυτού του προϊόντος όπου εκεί το προϊόν παρουσιάζεται με την περιγραφική του μορφή. Η μορφή αυτή περιλαμβάνει το όνομα του προϊόντος, την τιμή πώλησης, όλη την φωτογραφία, την κατηγορία όπου ανήκει και μια περιγραφή του. Τέλος δύναται η επιλογή στον χρήστη να το προσθέσει στο καλάθι αγορών του πατώντας στο κουμπί <<Προσθήκη στο καλάθι>> κουμπί που βρίσκεται μόνο στην ατομική σελίδα του κάθε προϊόντος.

Η σελίδα του καλαθιού αγορών περιλαμβάνει τα εξής:



1. Τα προϊόντα που έχει προσθέσει στο καλάθι του ο χρήστης με την παραπάνω μορφή. Περιέχοντας όνομα, κατηγορία, κωδικό, φωτογραφία και τιμή.
2. Επιλογέα που υποδεικνύει πόσα κομμάτια αυτού του προϊόντος έχει επιλέξει να αγοράσει ο χρήστης. Ο πελάτης ανάλογα με την επιθυμία του μπορεί να αλλάξει αυτόν τον αριθμό.
3. Κουμπί ακύρωσης. Ο χρήστης μπορεί να βγάλει από το καλάθι του ένα προϊόν πατώντας αυτό το κουμπί δίπλα από το προϊόν που θέλει να ακυρώσει.
4. Ολοκλήρωση παραγγελίας. Ο χρήστης αφού βεβαιωθεί για την παραγγελία του για να την επικυρώσει πατάει το κουμπί <<Ολοκλήρωση παραγγελίας>>. Πατώντας το θα μεταφερθεί στην σελίδα ολοκλήρωσής παραγγελίας.

Η σελίδα ολοκλήρωσής παραγγελίας είναι αυτής της μορφής:



1. Φόρμα συμπλήρωσης προσωπικών στοιχείων του πελάτη. Ζητείται από τον πελάτη να συμπληρώσει αυτή την φόρμα για να ολοκληρωθεί η παραγγελία του. Όλα τα πεδία είναι υποχρεωτικά.
2. Πίνακας ενημέρωσης οικονομικών. Από αυτόν τον πίνακα ο χρήστης μπορεί να ενημερωθεί για το συνολικό ποσό Φ.Π.Α, τα μεταφορικά και το συνολικό ποσό που ανέρχεται η παραγγελία του.

3. Επιλογή τρόπου παραλαβής. Ο χρήστης μπορεί να επιλέξει να παραλάβει τα προϊόντα που παρήγγειλε από το κατάστημα για μηδενίσει το κόστος αποστολής ή να του αποσταλούν στον τόπο που ο ίδιος θα επιλέξει.
4. Κουμπί ολοκλήρωσης παραγγελίας. Για να ολοκληρώσει την παραγγελία του ο χρήστης και να σταλεί στην αποθήκη θα πρέπει να πατήσει αυτό το κουμπί. Αν έχει συμπληρώσει σωστά τη φόρμα τότε θα μεταφερθεί σε μια σελίδα με ένα ευχαριστήριο μήνυμα και την επιλογή επιστροφής στην αρχική σελίδα του ηλεκτρονικού καταστήματος.

4 ΚΕΦΑΛΑΙΟ 4ο : Εμπειρίες, συμπεράσματα και βελτιώσεις

4.1 Εμπειρίες και συμπεράσματα από χρησιμοποιηθέντα εργαλεία.

Η εφαρμογή Storage Handler είχε σκοπό την καλύτερη διαχείριση μιας αποθήκης, καθώς και τη δημιουργία ενός εύχρηστου e-shop για την διευκόλυνση τόσο του προσωπικού όσο και των πελατών. Για την πραγματοποίηση της εργασίας χρησιμοποίησα το Django, το οποίο βοήθησε αρκετά σε αυτό το εγχείρημα. Το documentation μου παρείχε όλες τις πληροφορίες που χρειαζόμουν ώστε να το χρησιμοποιήσω σωστά. Οι τεχνικές σχεδίασης του είναι κατανοητές και εύκολες στην εφαρμογή τους. Πολλά κομμάτια της εφαρμογής όπως το authentication system και ο χειρισμός των κωδικών (password) ήταν έτοιμα προς χρήση. Με αρκετή ευκολία συνέδεσα το Django Rest plug-in στο Django για να χρησιμοποιήσω την λογική της επικοινωνίας του API με τη βάση δεδομένων με HTTP Requests. Τέλος, δεν γίνεται να μην αναφέρω το πακέτο Django admin, πάνω σε αυτό το έτοιμο πάνελ διαχείρισης χτίστηκε η εφαρμογή διαχείρισης της αποθήκης εμπλουτίζοντας την και τροποποιώντας την κατάλληλα.

Μειονεκτεί όμως σε κάποια θέματα που θεωρώ βασικά να αναφερθούν, συγκεκριμένα αντιμετώπισα δύο προβλήματα που παρουσιάστηκαν στη δημιουργία αυτής της εφαρμογής. Το Django παρουσίασε προβλήματα όταν παρεκκλίνοντας από τον προτεινόμενο τρόπο χρήσης του, δοκίμασα να φτιάξω ένα custom renderer. Χρησιμοποιώντας τον JSON renderer κάποια queries στα οποία τα δεδομένα δεν ήταν σε απλά data types δεν μπορούσαν να γίνουν serialized, συγκεκριμένα το μήνυμα σφάλματος που μου παρουσιαζόταν έγραφε unserialized queries. Κάνοντας inherit τον renderer πρόσθεσα κώδικα (overwrite) που θα επέτρεπε να γίνουν serialize data types όπως queries και models. Επίσης, χρειάστηκε αρκετή ώρα για να συνδέσω τη Mongodb η οποία είναι μία βάση δεδομένων χωρίς σχεσιακό μοντέλο. Το πρόβλημα ήταν πως κάποιες εκδόσεις του plug in για την σύνδεση του Django και της Mongo (Djongo) δεν ήταν συμβατές με κάποιες εκδόσεις του plug in Django Rest. Έπρεπε να βρεθούν οι εκδόσεις και των δύο plug in που να ταιριάζανε. Το Django μπορεί να μας διευκολύνει και να μας γλιτώνει χρόνο αλλά είναι δύστροπο όταν παρεκκλίνουμε από τον <<σωστό τρόπο>> που πρέπει να γίνονται τα πράγματα σύμφωνα με αυτό.

Το React, επίσης έχει άρτιο documentation το οποίο περιέχει και ένα μικρό tutorial. Η λογική σχεδίασης των components ήταν ιδιαίτερα χρήσιμη και συνέβαλε αρκετά σαν πρακτική για την ανάπτυξη απαραίτητων γνώσεων σε αυτόν τον τομέα. Ένα ακόμα πλεονέκτημα είναι πως

αποδομώντας όλο το interface σε μικρότερα κομμάτια, μπορούσα να τα επεξεργαστώ χωρίς να χρειαστεί να κάνω αλλαγές στην εφαρμογή, ενώ ταυτόχρονα μου δινόταν η δυνατότητα να τα ξαναχρησιμοποιήσω. Κατά τη διάρκεια κατασκευής της εφαρμογής το React δεν παρουσίασε κανένα πρόβλημα, αντιθέτως ενίσχυσε την δυνατότητα δημιουργίας καλύτερων user interface. Το React χρησιμοποιήθηκε για τη δημιουργία της διεπαφής του ηλεκτρονικού καταστήματος με τον πελάτη δηλαδή το UI του e-shop.

Για τη βάση δεδομένων χρησιμοποίησα τη mongodb. Είναι μία βάση δεδομένων η οποία δε χρησιμοποιεί σχεσιακό μοντέλο. Με αυτόν τον τρόπο εξασφαλίζουμε μεγαλύτερη ευελιξία στην αποθήκευση αλλά και στην ανάκτηση δεδομένων. Δεν εκτελούμε migrations και η μορφή των δεδομένων είναι σε JSON.

Θεωρώ επίσης μείζον να αναφέρω δύο εργαλεία που με βοήθησαν στον εντοπισμό σφαλμάτων στην εφαρμογή. Το πρώτο είναι το Compass της mongodb. Πρόκειται για ένα εργαλείο το οποίο μας επιτρέπει να επεξεργαστούμε με ευκολία τα στοιχεία της βάσης δεδομένων της εφαρμογής. Έχουμε την δυνατότητα να αλλάξουμε τις τιμές, να δούμε τον τύπο δεδομένων καθώς και να έχουμε πρόσβαση σε άλλες πληροφορίες για όποιο πεδίο επιθυμούμε. Το δεύτερο είναι το Postman. Στην εφαρμογή αυτή κάναμε χρήση της αρχιτεκτονικής REST (επικοινωνία client-server με “HTTP” request-response). Με το Postman μπορούμε να ελέγξουμε τα αιτήματα και τις απαντήσεις (request-response) μεταξύ του client και του server.

Τέλος είναι σημαντική η αναφορά στη χρήση του IDE (ολοκληρωμένο περιβάλλον ανάπτυξης) το οποίο είναι το Visual Studio Code της Microsoft. Χάρη στα αμέτρητα εργαλεία, τα plug-ins αλλά και το παραμετροποιήσιμο interface, η εφαρμογή αυτή αναπτύχθηκε ποιο εύκολα και πιο ευχάριστα με λιγότερα λάθη προς διόρθωση.

4.2 Βελτιώσεις, επιπλέον προσθήκες

Παρά τον αριθμό των δυνατοτήτων της εφαρμογής αυτής, θα μπορούσαν να υπάρξουν επιπλέον λειτουργίες.

Μια αρκετά σημαντική αλλά δύσκολα υλοποίησιμη λειτουργία είναι η δημιουργία συστήματος αυτοματοποιημένων ειδοποιήσεων push notification, όπου οι χρήστες θα ενημερώνονται για οποιαδήποτε αλλαγή στην εφαρμογή άμεσα, με αποτέλεσμα την αμεσότερη αλληλεπίδραση. Τα push notifications μοιάζουν με sms, είναι μνήματα pop-up και χρησιμοποιούνται κυρίως σε εφαρμογές smartphone. Η τεχνολογία Push, ή το push server,

είναι ένα ήδος επικοινωνίας που βασίζεται στο Διαδίκτυο. Το αίτημα για μια συγκεκριμένη συναλλαγή ξεκινά από τον server, σε αντίθεση με το pull / get, όπου το αίτημα για μετάδοση πληροφοριών ξεκινά από τον παραλήπτη ή τον πελάτη. Θα μπορούσε να γίνει χρήση και service worker. Το service worker είναι ένα script που εκτελείται στο παρασκήνιο του browser (πρόγραμμα περιήγησης), ξεχωριστά από μια ιστοσελίδα, επιτρέποντας την πρόσθεση λειτουργιών που δεν χρειάζονται web page η UI. Μια από αυτές τις λειτουργίες είναι και τα push notifications. Το service worker δίνει την δυνατότητα στους προγραμματιστές να υποστηρίξουν μια offline εμπειρία. Οι εργαζόμενοι θα ειδοποιούνται άμεσα για τα προϊόντα σε χαμηλό απόθεμα και για τις καινούργιες παραγγελίες μειώνοντας έτσι τον χρόνο ολοκλήρωσης αυτών των εργασιών.

Εν κατακλείδι, παρότι υπήρξαν δυσκολίες στην υλοποίηση της εφαρμογής αυτής και θα μπορούσαν να γίνουν κάποιες βελτιώσεις, το αποτέλεσμα είναι αρκετά ικανοποιητικό. Πρόκειται για μια εφαρμογή με αρκετές λειτουργίες όπου θα επιλύσουν ρεαλιστικά προβλήματα που αντιμετωπίζει καθημερινά το εργατικό δυναμικό μιας αποθήκης. Ταυτόχρονα μέσω του e-shop δίνεται η ευκαιρία στους πελάτες να έχουν μια ασφαλή και άμεση συναλλαγή με τις αποθήκες καθώς και έχουν πρόσβαση σε όλες τις πληροφορίες σχετικά με τα προιόντα και τις τιμές τους. Παρά τις ατέλειες της, έχει αρκετές προδιαγραφές για μια άκρως χρήσιμη εφαρμογή, ικανή να υποστηρίξει τις ανάγκες αυτών που την χρησιμοποιούν. Σε προσωπικό επίπεδο η δημιουργία αυτής της εφαρμογής συνέβαλε στην καλύτερη κατανόηση των διαδικτυακών εφαρμογών, εμπλούτισε τις γνώσεις μου πάνω σε χρήσιμα και σύγχρονα εργαλεία και τεχνικές σχεδίασης.

Βιβλιογραφία – Αναφορές - Διαδικτυακές Πηγές

Διαδικτυακές Πηγές:

1.1: http://www.it.uom.gr/project/client_server/theoria1.htm
https://en.wikipedia.org/wiki/Server_(computing)
1.2.1: https://hackr.io/blog/what-is-frameworks
1.2.2: https://hackr.io/blog/what-is-frameworks
1.3.1: https://en.wikipedia.org/wiki/Server-side_scripting,
https://en.wikipedia.org/wiki/Dynamic_web_page#Client-side_scripting
1.3.2.1: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction
1.3.2.1.1: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller,
https://www.freecodecamp.org/news/model-view-controller-mvc-explained-through-ordering-drinks-at-the-bar-efcba6255053/,
foto from: https://www.youtube.com/watch?v=DUg2SWWK18I
1.3.2.2: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction
1.3.2.3: https://dev.to/creativetim_official/what-is-laravel-explain-it-like-i-m-five-19eb
1.3.2.4: https://el.wikipedia.org/wiki/Ruby_on_Rails
1.3.3.1: https://www.theseus.fi/bitstream/handle/10024/130495/FInal_Year_Thesis.pdf
https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/
1.3.3.2: https://www.altexsoft.com/blog/engineering/pros-and-cons-of-vue-js/
1.3.3.3: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/
3.7.1:https://www.bbvaapimarket.com/en/api-world/rest-api-what-it-and-what-are-its-advantages-project-development/
3.7.2: https://djangostars.com/blog/why-we-use-django-framework/
https://steelkiwi.com/blog/why-django-best-web-framework-your-project/
https://inmagik.com/blog/10-reasons-to-use-django-in-2020
3.7.3: https://stackshare.io/django-rest-framework
3.7.4 https://www.mongodb.com/why-use-mongodb
3.7.5: https://railsware.com/blog/why-use-react/
https://javascript.plainenglish.io/5-reasons-why-you-should-be-using-react-238373cc245e