ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη διαδικτυακής εφαρμογής κοινωνικού δικτύου με δυνατότητες υλοποίησης εμπορικών πράξεων

Φοιτήτρια: Μελίνα-Μαρία Ερμείδη
ΑΜ: 18390119

Επιβλέπων Καθηγητής: Νικόλαος Ζάχαρης, Καθηγητής

ΑΘΗΝΑ-ΑΙΓΑΛΕΩ, ΙΟΥΛΙΟΣ 2024

UNIVERSITY OF WEST ATTICA

FACULTY OF ENGINEERING

DEPARTMENT OF INFORMATICS AND COMPUTER ENGINEERING

Diploma Thesis

Development of a social networking web application with e-commerce capabilities

Student: Melina Maria Ermeidi
Registration Number: 18390119

Supervisor: Nikolaos Zaharis, Professor

ATHENS-EGALEO, JULY 2024

Η Διπλωματική Εργασία έγινε αποδεκτή και βαθμολογήθηκε από την εξής τριμελή επιτροπή:

Νικόλαος Ζάχαρης, Καθηγητής	Γεώργιος Πρεζεράκος, Καθηγητής	Παναγιώτης Γιαννακόπουλος, Καθηγητής
Digitally signed by Nikolaos Zacharis Date: 2024.07.15 15:02:44 +03'00' (Υπογραφή)	Digitally signed by Georgios Prezerakos Date: 2024.07.18 10:39:20 +03'00' (Υπογραφή)	Panagiotis Giannakopoulos Digitally signed by Panagiotis Giannakopoulos Date: 2024.07.18 10:54:38 +03'00' (Υπογραφή)

Copyright© Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Μελίνα-Μαρία Ερμείδη, Ιούλιος, 2024
Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τους συγγραφείς.
Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον/την συγγραφέα του και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις θέσεις του επιβλέποντος, της επιτροπής εξέτασης ή τις επίσημες θέσεις του Τμήματος και του Ιδρύματος.

ΔΗΛΩΣΗ ΠΕΡΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ ΚΑΙ ΛΟΓΟΚΛΟΠΗΣ

Ο/η κάτωθι υπογεγραμμένος/η Μελίνα-Μαρία Ερμείδη του Παναγιώτη, με αριθμό μητρώου 18390119 φοιτητής/τρια του Πανεπιστημίου Δυτικής Αττικής της Σχολής ΜΗΧΑΝΙΚΩΝ του Τμήματος ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ,

δηλώνω υπεύθυνα ότι:

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

Ο/Η Δηλών/ούσα
Μελίνα-Μαρία Ερμείδη

Υπογραφή:

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέπων καθηγητή κ. Νικόλαο Ζάχαρη για την καθοδήγηση και την υποστήριξη που μου πρόσφερε κατά την υλοποίηση της διπλωματικής εργασίας αυτής. Ακόμα θα ήθελα να ευχαριστήσω την οικογένεια μου για την συμπαράσταση και την υποστήριξη τους κατά την διάρκεια των σπουδών μου. Αφιερώνω την διπλωματική μου εργασία στους γονείς μου Αντωνία και Παναγιώτη και στην αδελφή μου Ελεάνα.

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία αφορά την δημιουργία μιας διαδικτυακής εφαρμογής κοινωνικού δικτύου με την δυνατότητα αγοροπωλησίας διάφορων προϊόντων που προσφέρονται από τα άτομα στο κοινωνικό δίκτυο. Οι χρήστες μπορούν να δημιουργούν προσωπικά προφίλ, να επεξεργάζονται το προφίλ τους, να αναζητούν και να προσθέτουν προϊόντα που επιθυμούν να πουλήσουν και να τα τροποποιήσουν, και τέλος να επικοινωνούν με άλλους χρήστες για τα προϊόντα που επιθυμούν να παραλάβουν ή να τους αποσταλούν.

Η δημιουργία και η δομή της πλατφόρμας βασίζεται σε ένα σύνολο τεχνολογιών. Συγκεκριμένα για το frontend, χρησιμοποιήθηκε το εργαλείο ανάπτυξης Vite, συνοδευόμενο από το CSS Bootstrap που επιτρέπει τη σχεδίαση και τη διάταξη του περιεχομένου της εφαρμογής. Επιπλέον, εφαρμόστηκε η επέκταση Sass της CSS, που επιτρέπει τη χρήση μεταβλητών και ενσωματωμένων λειτουργιών για πιο ευέλικτο σχεδιασμό.

Για το backend, η ανάπτυξη βασίστηκε στο Laravel Framework, ένα από τα κορυφαία PHP frameworks για τη δημιουργία web εφαρμογών, γραμμένο σε γλώσσα προγραμματισμού PHP. Για τη φιλοξενία του backend, χρησιμοποιήθηκε ο server Apache. Η διαχείριση των εξαρτήσεων και των scripts στο frontend και στο backend πραγματοποιήθηκε μέσω του npm (Node Package Manager) και του Composer αντίστοιχα, δυο πακέτων που παρέχουν εύκολη διαχείριση των απαιτούμενων εργαλείων και εξαρτήσεων.

Όσον αφορά τη βάση δεδομένων, επιλέχθηκε το MySQL, ένα σύστημα διαχείρισης βάσεων δεδομένων για την αποθήκευση και την ανάκτηση των δεδομένων.

Η βάση αποτελείται από 9 πίνακες, στους οποίους αποθηκεύονται όλα τα δεδομένα, όπως τα στοιχεία εγγραφής του κάθε χρήστη, πληροφορίες για τα προϊόντα με το οποία ο αγοραστής εμπλουτίζει τα προϊόντα του, και το περιεχόμενο των μηνυμάτων που ανταλλάσσονται μεταξύ των χρηστών.

Η ανάπτυξη και ο προγραμματισμός της εφαρμογής έγινε μέσω του Visual Studio Code και η στήριξη του server Apache μαζί με την βάση δεδομένων χρησιμοποιώντας το εργαλείο XAMPP.

Λέξεις-Κλειδιά: Frameworks, Front-end, Back-end, Server, Laravel, Apache, MySQL, Βάσεις δεδομένων, Client, Server, Διαδικτυακές εφαρμογές, Marketplace, Εμπορικές συναλλαγές, Προϊόντα, Κοινωνικό Δίκτυο, Προσωπικό Προφίλ.

ABSTRACT

This thesis concerns the development of a social networking web application with the capability of facilitating the buying and selling of various products offered by individuals within the social network. Users are able to create personal profiles, edit their profiles, search for and add products they wish to sell, modify them, and communicate with other users regarding the products they wish to receive or send.

The creation and structure of the platform are based on a set of technologies. Specifically, for the frontend, the Vite development tool was utilized, accompanied by the CSS framework Bootstrap, enabling the design and layout of the application's content. Additionally, the Sass extension of CSS was implemented, allowing for the use of variables and embedded functions for more flexible design.

For the backend, development relied on the Laravel Framework, one of the leading PHP frameworks for creating web applications, written in the PHP programming language. Regarding backend hosting, the Apache server was employed. Dependency and script management for both frontend and backend were handled through npm (Node Package Manager) and Composer respectively, two packages providing easy management of required tools and dependencies.

As for the database, MySQL was chosen, a relational database management system for storing and retrieving data. The database consists of 9 tables, storing all data, such as user registration details, information about the products through which buyers enrich their products, and the content of messages exchanged between users.

Development and programming of the application were carried out using Visual Studio Code, with support from the Apache server and the database utilizing the XAMPP tool.

Keywords: Frameworks, Front-end, Back-end, Server, Laravel, Apache, MySQL, Databases, Client, Server, Web Applications, Marketplace, Commercial Transactions, Products, Social Network, Personal Profile.

Περιεχόμενα

Κατάλογος εικόνων

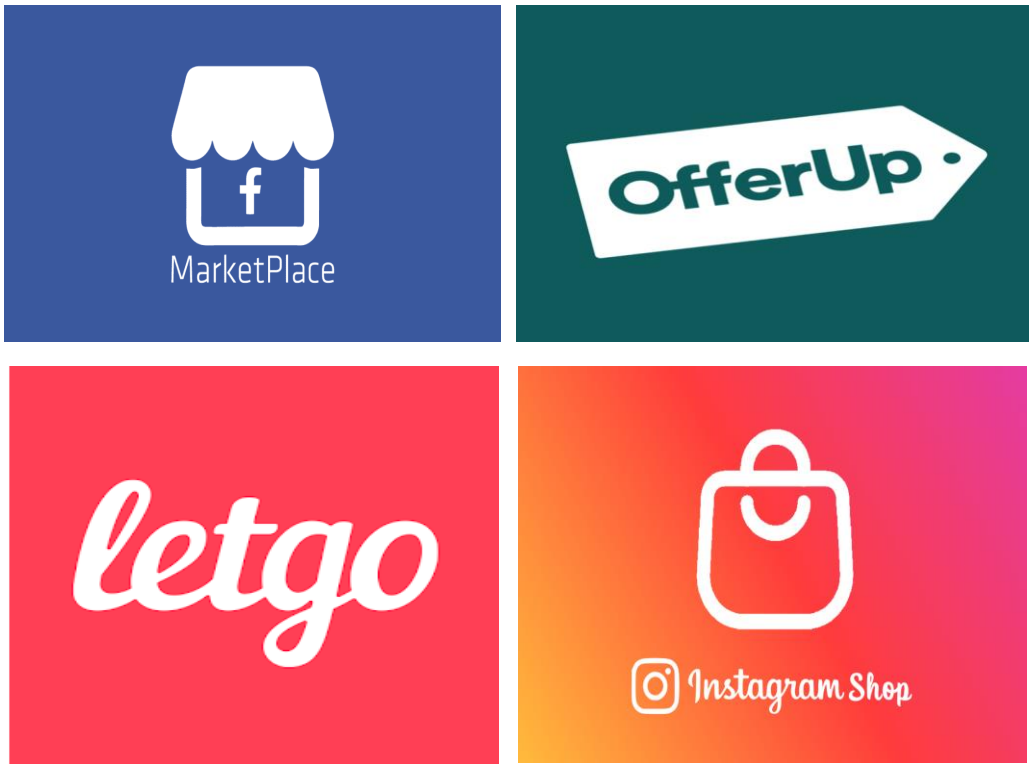
Εισαγωγή

Στην εποχή αυτή που διανύουμε όλο ένα και περισσότερες καινοτόμες εφαρμογές δημιουργούνται για να εξυπηρετήσουν τους χρήστες του διαδικτύου. Μία σημαντική κατηγορία διαδικτυακών εφαρμογών αυτών είναι των commerce. Είναι σημαντικό για έναν χρήστη-πωλητή να μπορεί να χειρίζεται μια διαδικτυακή εφαρμογή με ευκολία, και να είναι user-friendly προς αυτόν. Επιπλέον, είναι επίσης σημαντικό για τον χρήστη αυτόν να μπορεί να χειρίζεται συγκεκριμένες υπηρεσίες μέσα σε λίγα «κλικ». Μάλιστα, μια καινοτόμα υπηρεσία σε μια τέτοια εφαρμογή είναι η δυνατότητα επικοινωνίας με τους άλλους χρήστες.

Η εφαρμογή αυτή, η οποία είναι μια διαδικτυακή εφαρμογή κοινωνικού δικτύου με δυνατότητα αγοροπωλησίας προϊόντων, προσφέρει στους χρήστες της τα εξής:

- Δημιουργία προσωπικού προφίλ
- Αναζήτηση και προσθήκη προϊόντος προς πώληση
- Άμεση Επικοινωνία με τον χρήστη σε περίπτωση ενδιαφέροντος κάποιου προϊόντος (Επικοινωνία για τις συναλλαγές)

Παρόμοιες εφαρμογές τέτοιου τύπου περιλαμβάνουν το Facebook Marketplace [1], το OfferUp [2], το Letgo [3]και το Instagram Shop [4]. Αυτές οι πλατφόρμες είναι σημαντικές στον χώρο του e-commerce και δίνουν στους χρήστες τη δυνατότητα να αναζητούν και να κοινοποιούν προϊόντα μέσω ενός κοινωνικού δικτύου. Η παρούσα εφαρμογή εμπνέεται από αυτές τις λύσεις και αναπτύσσεται στο πλαίσιο της διπλωματικής εργασίας αυτής και αναλαμβάνει να προσφέρει παρόμοιες λειτουργίες.



Εικόνα 1 - Εφαρμογές e-commerce που συνδυάζουν το κοινωνικό δίκτυο. Με την σειρά: Facebook Marketplce [1], OfferUp [2], Letgo [3], Instagram Shop [4].

Αντικείμενο της διπλωματικής εργασίας

Το αντικείμενο της διπλωματικής εργασίας αυτής είναι η δημιουργία μιας διαδικτυακής εφαρμογής κοινωνικού δικτύου με δυνατότητα αγοροπωλησίας προϊόντων με σκοπό να επιτρέπει στους χρήστες να δημιουργούν προσωπικά προφίλ, να προσθέτουν προϊόντα προς πώληση και να επικοινωνούν με τους άλλους χρήστες και την διεκπεραίωση των συναλλαγών αυτών. Για την δημιουργία της χρησιμοποιήθηκαν τεχνολογίες όπως το framework Laravel και γλώσσες προγραμματισμού Php και Javascript. Η αποθήκευση, η ανάκτηση και η εμφάνιση των δεδομένων έγινε με την βάση δεδομένων MySQL.

Σκοπός και στόχοι

Ο σκοπός της εφαρμογής αυτής είναι η δημιουργία ενός χρήσιμου εργαλείου όχι μόνο για τους απλούς χρήστες που επιθυμούν να αγοράσουν κάθε λογής προϊόν, αλλά και για τους πωλητές που αναζητούν γρήγορους και εύκολους τρόπος πώλησης των προϊόντων τους. Οι κύριοι στόχοι είναι να είναι φιλική προς τους χρήστες της, με απλή πλοήγηση και ενσωματωμένες λειτουργίες για την διευκόλυνση της αγοροπωλησίας και η προσφορά ενός μέσου επικοινωνίας μεταξύ των χρηστών για την αποτελεσματική εξέλιξή των συναλλαγών.

Μεθοδολογία

Η μεθοδολογία της διπλωματικής μου εργασίας άρχισε από μια θεωρητική ερευνά στο διαδίκτυο για δημοφιλή frameworks που χρησιμοποιούνται συγκεκριμένα για εφαρμογές e-commerce. Σημείωσα τις απαιτήσεις και τις λειτουργίες , και αφού εξετάστηκαν οι τεχνολογίες και τα εργαλεία προχώρησα στην πρακτική υλοποίηση. Η εφαρμογή αναπτύχθηκε σε στάδια, τόσο για το frontend όσο και για το backend. Αρχικά δόθηκε έμφαση στην ανάπτυξη του frontend για τη δημιουργία ενός user-friendly περιβάλλοντος. Στη συνέχεια, επικεντρώθηκα στην ανάπτυξη του backend, διασφαλίζοντας τη σωστή διαχείριση των δεδομένων και την αποτελεσματική λειτουργία των βασικών υπηρεσιών της εφαρμογής. Καθ' όλη τη διάρκεια της ανάπτυξης, η εφαρμογή υποβλήθηκε σε συνεχή δοκιμαστική χρήση, προκειμένου να διασφαλιστεί η ομαλή λειτουργία της. Τέλος, αφού ολοκλήρωσα το πρακτικό μέρος συνέχισα στην συγγραφή του θεωρητικού μέρους.

Δομή

Η διπλωματική μου εργασία αποτελείται από δύο τμήματα, το θεωρητικό και το πρακτικό. Στο θεωρητικό μέρος περιλαμβάνεται ένα αρχείο Word 78 σελίδων, το οποίο αποτελείται από 6 κεφάλαια τα οποία αναλύουν λεπτομερώς τις τεχνολογίες που εφαρμόστηκαν και τη θεωρητική βάση πίσω από την ανάπτυξη της πλατφόρμας . Το πρακτικό μέρος περιλαμβάνει την ίδια πλατφόρμα, παρουσιάζοντας την υλοποίηση της διαδικτυακής εφαρμογής.

1. Κεφάλαιο 1ο: Front-end Τεχνολογίες (client side scripting)

Παρακάτω περιγράφονται αναλυτικά όλες οι τεχνολογίες που χρησιμοποιηθήκαν για την ανάπτυξη του front-end μέρους της εφαρμογής.

1.1 Bootstrap Framework

Το Bootstrap [5] είναι ένα από τα πιο δημοφιλή πλέον front-end framework για τη δημιουργία διαδικτυακών εφαρμογών και ιστοσελίδων. Προσφέρει ένα σύνολο εργαλείων και στυλ για την ανάπτυξη ευέλικτων και ανταποκρίσιμων ιστοσελίδων. Βασίζεται σε HTML, CSS και JavaScript και παρέχει προκαθορισμένα στυλ για κουμπιά, φόρμες, πλαίσια κ.ά., τα οποία μπορούν να προσαρμοστούν εύκολα ανάλογα με τις ανάγκες του έργου.

Τα βασικά χαρακτηριστικά του Bootstrap περιλαμβάνουν:

- Πλήρη ανταπόκριση: Οι ιστοσελίδες που δημιουργούνται με Bootstrap προσαρμόζονται αυτόματα σε διάφορες συσκευές και μεγέθη οθονών.
- Πλούσια συλλογή στοιχείων: Περιλαμβάνει πολλά έτοιμα στυλ και στοιχεία που μπορούν να ενσωματωθούν στο έργο χωρίς πολλές τεχνικές δεξιότητες.
- Υποστήριξη πολλαπλών προγραμματιστικών γλωσσών: Εκτός από HTML/CSS, το Bootstrap μπορεί να χρησιμοποιηθεί με JavaScript frameworks όπως το Angular, το React κ.λπ.
- Ευκολία χρήσης: Η σύνταξη του Bootstrap είναι ευκολοδιάβαστη και οργανωμένη, επιτρέποντας στους προγραμματιστές να δημιουργήσουν γρήγορα και εύκολα ένα σύγχρονο UI.

Εικόνα 2 – Bootstrap [5] logo

Στην εφαρμογή συγκεκριμένα, το Bootstrap Framework χρησιμοποιείται ως dependency και είναι εγκατεστημένο στο περιβάλλον ανάπτυξης. Έχει προστεθεί ως πακέτο στον φάκελο resources της εφαρμογής. Τα αρχεία (views) στα resources, τα οποία είναι τα αρχεία που δείχνουν τις σελίδες στον χρήστη, χρησιμοποιούν το Bootstrap framework. Μερικά από αυτά τα στοιχεία των σελίδων είναι τα κουμπιά (btn), φόρμες (form), διαχωριστικές γραμμές (row,col) τα οποία είναι καθαρά στοιχεία δόμησης σελίδας με το framework αυτό. Η χρήση των στοιχείων του Bootstrap σε αυτά τα αρχεία βοηθά στη δημιουργία μιας ενιαίας και ευχάριστης εμπειρίας του χρήστη.

1.2 Vite εργαλείο

Το Vite [6] είναι ένα σύγχρονο εργαλείο δημιουργίας περιβάλλοντος για την ανάπτυξη εφαρμογών και συνήθως χρησιμοποιείται μαζί με το Laravel Framework. Το Vite, το οποίο σημαίνει "γρήγορο" στα γαλλικά, δικαιολογεί το όνομά του προσφέροντας γρήγορους χρόνους ανάπτυξης και δημιουργίας για σύγχρονες εφαρμογές JavaScript. Η ολοκληρωμένη ενσωμάτωση του Vite με δημοφιλή frameworks για το frontend όπως το Vue.js και το React, επιτρέπει στους προγραμματιστές να δημιουργήσουν διαδραστικά και δυναμικά περιβάλλοντα με ευκολία. Είναι ένα ανεκτίμητο εργαλείο για τους προγραμματιστές που επιθυμούν να βελτιώσουν τη ροή εργασίας ανάπτυξης του frontend και να δημιουργήσουν σύγχρονες εφαρμογές.

Εικόνα 3 - Vite [6] logo

Τα πλεονεκτήματα του είναι τα εξής:

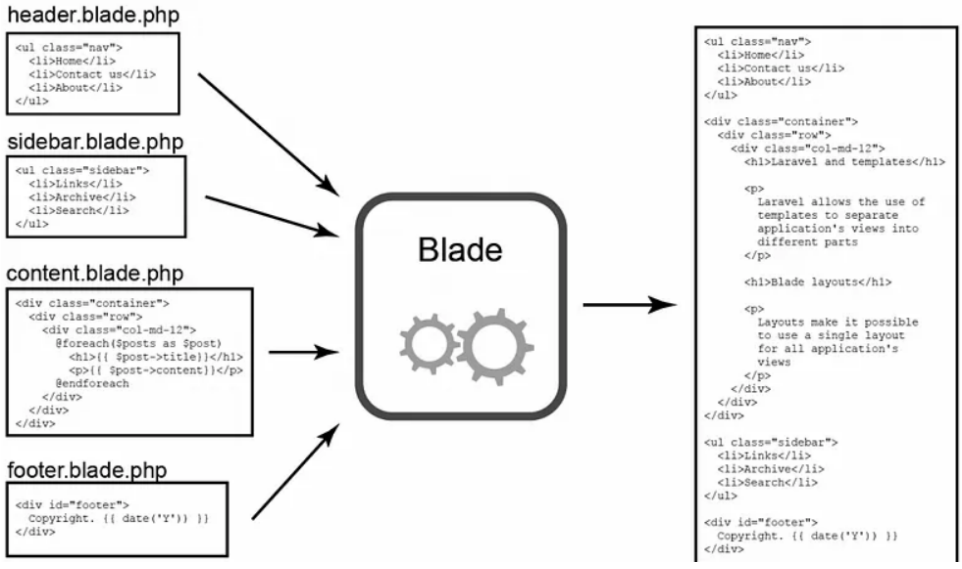
1. **Ταχύτητα ανάπτυξης**: Το Vite είναι γρήγορο ως προς την ανταπόκριση κατά την ανάπτυξη, και αυτό επιτρέπει να εμφανίζονται αμέσως οι αλλαγές στον κώδικα χωρίς την ανάγκη ανανέωσης της σελίδας.
2. **Βελτιστοποίηση απόδοσης**: Το Vite βασίζεται στη σύγχρονη τεχνολογία των ES Modules για να παρέχει γρήγορο χρόνο φόρτωσης και αποδοτική μνήμη στην ανάπτυξη του frontend.
3. **Εύκολη ενσωμάτωση:** Η ενσωμάτωση του Vite σε ένα περιβάλλον όπως του Laravel είναι αρκετά απλή, και συνήθως αυτά τα 2 frameworks συνδυάζονται μαζί για την υλοποίηση εφαρμογών.

Ουσιαστικά στην εφαρμογή χρησιμοποιείται για την ανάπτυξη του client-side κώδικα και λειτουργεί ως ένα εργαλείο που ανιχνεύει αυτόματα τις αλλαγές στα αρχεία καθώς αυτά τροποποιούνται με τον χρόνο, και κατά την αποθήκευσή τους, παρέχει αμέσως τις ανανεώσεις στον κώδικα που προβάλλονται στον web browser. Συγχρόνως, η εφαρμογή παρέχει ασφαλή περιεχόμενο μέσω του πρωτοκόλλου HTTPS και διαχειρίζεται αιτήματα μέσω του Apache HTTP Server (back-end web server).

1.3 Blade templates

Τα αρχεία Blade [7] αποτελούν μια ενσωματωμένη γλώσσα προτύπου του Laravel, η οποία επιτρέπει στους προγραμματιστές να ενσωματώνουν κώδικα PHP στα HTML αρχεία τους. Αυτό παρέχει τη δυνατότητα δημιουργίας δυναμικού περιεχομένου, επέκτασης των δυνατοτήτων της συντακτικής σήμανσης HTML και οργάνωσης του κώδικα σε πιο δομημένο τρόπο. Μέσα από τη χρήση των αρχείων Blade, δημιουργούνται δυναμικές προβολές δεδομένων και προβάλλουν διαφορετικά περιεχόμενα ανάλογα με την κατάσταση της εφαρμογής. Τα αρχεία προτύπου Blade χρησιμοποιούν την επέκταση αρχείου .blade.php και αποθηκεύονται στον κατάλογο resources/views της εφαρμογής.

Επιπλέον, με τη χρήση των partials, τα οποία αντιστοιχούν στις προβολές, διαχωρίζονται τα διάφορα τμήματα του HTML κώδικα σε επαναχρησιμοποιήσιμα μικρά κομμάτια, όπως η κεφαλίδα, το περιεχόμενο και το σώμα κτλ. Αυτές οι προβολές είναι μικρά αρχεία προτύπων που περιέχουν μέρη του HTML κώδικα που επαναχρησιμοποιούνται σε πολλές σελίδες της εφαρμογής. Η χρήση τους βοηθά στη διατήρηση ενός οργανωμένου και επαναχρησιμοποιήσιμου κώδικα.Αυτό επιτρέπει την ευκολότερη συντήρηση και επεξεργασία του κώδικα, καθώς όταν ενημερώνουμε ένα μέρος, όπως η κεφαλίδα, σε ένα από τα αρχεία partials, αυτή η αλλαγή αντανακλάται αυτόματα σε όλα τα μέρη του ιστότοπου που χρησιμοποιούν αυτό το συγκεκριμένο τμήμα.



Εικόνα 4 - Τα "partials" στο Laravel Blade. [8]

1.5 Sass

Η εφαρμογή χρησιμοποιεί το Sass [9] (Syntactically Awesome Stylesheets), έναν προεπεξεργαστή CSS που επιτρέπει επιπλέον δυνατότητες και ευκολίες στη συγγραφή κώδικα CSS. Αυτός ο προεπεξεργαστής δημιουργεί πιο οργανωμένα και ευανάγνωστα στυλ, με χρήση μεταβλητών και άλλων χαρακτηριστικών. Είναι ιδανικός για τη διαχείριση του CSS σε μεγάλα έργα και εφαρμογές.

Στον φάκελο "resources" της εφαρμογής, στον υποφάκελο "sass", υπάρχουν δύο αρχεία:

1. **variables**: Αυτό το αρχείο περιλαμβάνει μεταβλητές που χρησιμοποιούνται για τον ορισμό των χαρακτηριστικών του εμφανίσιμου στυλ. Εδώ ορίζονται μεταβλητές για το χρώμα του φόντου του body και για τα χαρακτηριστικά τυπογραφίας, όπως το όνομα της γραμματοσειράς και το μέγεθος της γραμματοσειράς.
2. **main**: Αυτό το αρχείο είναι το κύριο αρχείο Sass της εφαρμογής, όπου γίνονται οι εισαγωγές των άλλων αρχείων Sass και δηλώνονται οι κανόνες της μορφής CSS.

1.6 Javascript

Η Javascript [10] είναι η γλώσσα προγραμματισμού που χρησιμοποιείται για τη δημιουργία δυναμικών και αλληλεπιδραστικών ιστοσελίδων. Είναι αναγκαία για την προγραμματιστική πλευρά του front-end μέρους, δίνοντας τη δυνατότητα για την επεξεργασία και τη διαχείριση των δεδομένων που εμφανίζονται στην ιστοσελίδα. Προβάλλει τις ενημερώσεις και βοηθάει την εφαρμογή να την κάνει πιο «ζωντανή».

Στην εφαρμογή η Javascript έχει χρησιμοποιηθεί για την συγγραφή διάφορων συναρτήσεων στις επικεφαλίδες <scripts> των αρχείων με κατάληξη «.blade.php» (φάκελος resources/views) προκειμένου να υλοποιηθούν οι ζητούμενες λειτουργείες κάθε σελίδας.

Μερικά πλεονεκτήματα που προσφέρει η γλώσσα αυτή είναι τα εξής:

1. **Ταχύτητα**: Η JavaScript είναι μια γρήγορη γλώσσα, που σημαίνει ότι μπορεί να χρησιμοποιηθεί για τη δημιουργία διαδραστικών χαρακτηριστικών χωρίς να επιβραδύνει τη σελίδα.
2. **Απλότητα**: Είναι μια σχετικά εύκολη γλώσσα για να μάθει κανείς, γεγονός που την καθιστά προσιτή σε ένα ευρύ φάσμα προγραμματιστών.
3. **Δημοτικότητα**: Αποτελεί από τις πιο δημοφιλείς γλώσσες προγραμματισμού στον κόσμο, πράγμα που σημαίνει ότι υπάρχουν πολλοί διαθέσιμοι πόροι για να βοηθήσουν τους προγραμματιστές να τη μάθουν και να τη χρησιμοποιήσουν.
4. **Πολλαπλή χρησιμότητα**: Μπορεί να χρησιμοποιηθεί για τη δημιουργία μιας μεγάλης ποικιλίας διαδραστικών χαρακτηριστικών, από απλές κινούμενες εικόνες μέχρι σύνθετα παιχνίδια.
5. **Διαλειτουργικότητα**: Η JavaScript μπορεί να χρησιμοποιηθεί με άλλες γλώσσες προγραμματισμού, όπως η HTML και η CSS, για τη δημιουργία ισχυρών διαδικτυακών εφαρμογών.

1.7 Html

Η HTML [11] (HyperText Markup Language) αποτελεί τη βάση κάθε ιστοσελίδας, καθώς παρέχει τη δομή για το περιεχόμενο που εμφανίζεται σε μια ιστοσελίδα. Χρησιμοποιείται για την κωδικοποίηση των διαφόρων στοιχείων, όπως κείμενο, εικόνες, και συνδέσμους, που συνθέτουν την εμφάνιση κάθε σελίδας.

Παράδειγμα ενός τέτοιου αρχείου [12]:

```
<!DOCTYPE html>
<html>
<head>
   <title>Marketplace</title>
</head>
<body>
      <h1>Welcome to the Marketplace Web Application</h1>
</body>
</html>
```

Αυτό το κομμάτι κώδικα HTML περιγράφει μια βασική ιστοσελίδα αγοράς. Ο τίτλος είναι "Marketplace", και μέσα στο κορμό της σελίδας υπάρχει ένας τίτλος επικεφαλίδας (h1) που γράφει "Welcome to the Marketplace Web Application". Δημιουργεί μια σελίδα ιστοσελίδας με ένα καλωσόρισμα στον διαδικτυακό χώρο αγορών.

1.8 Css

Η CSS [13] (Cascading Style Sheets) είναι η γλώσσα στυλ που χρησιμοποιείται για τη σχεδίαση και τη μορφοποίηση του περιεχομένου που ορίζεται σε μια ιστοσελίδα HTML. Περιγράφει το πώς θα πρέπει να εμφανίζεται το περιεχόμενο της σελίδας, δηλαδή το μέγεθος, η γραμματοσειρά, το χρώμα κτλ. Η γλώσσα αυτή ορίζεται μέσα στην επικεφαλίδα <style> ενός html αρχείου και μπαίνει στο <head> τμήμα.

Παράδειγμα χρήσης της σε ένα html αρχείο μπορεί να είναι το εξής [14]:

```
<style>
  p {
   color: blue;
   font-size: 10px;
   background-color: red;
}
</style>
```

Εδώ η μεταβλητή p επιλέγει όλα τα <p> στοιχεία, η μεταβλητή color αλλάζει το χρώμα των στοιχείων σε μπλε συγκεκριμένα εδώ, το font-size ορίζει το μέγεθος των γραμμάτων και το background color ορίζει το χρώμα του background, Μπορούν να προστεθούν και άλλα τέτοιες μεταβλητές που να προσδιορίζουν διάφορα στιλιστικά στοιχεία κ.ο.κ. Ο κώδικας αυτός ορίζεται μέσα στο tag <style> ενός html αρχείου. Για παράδειγμα, σαν προσθήκη στο παράδειγμα [12] μπορεί να γίνει η εξής αλλαγή:

```
<!DOCTYPE html>
<html>
<head>
<style>
  p {
   color: red;
   font-size: 10px;
   background-color: red;
}
</style>
   <title>Marketplace</title>
</head>
<body>
   <h1>Welcome to the Marketplace Web Application</h1>
   <p>This is a paragraph with blue text and background.</p>
</body>
</html>
```

Προστέθηκε το στοιχείο <p> στην κεφαλίδα <body>. Αυτός ο κώδικας προσθέτει μία παράγραφο (<p>) με μπλε κείμενο και κόκκινο φόντο.

2. Κεφάλαιο 2ο: Back-end Τεχνολογίες (server side scripting)

Παρακάτω περιγράφονται αναλυτικά όλες οι τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη του back-end μέρους της εφαρμογής.

2.1 Laravel Framework

Το Laravel [15] είναι ένα δημοφιλές PHP framework για την ανάπτυξη web εφαρμογών. Προσφέρει ένα ευέλικτο και ευανάγνωστο περιβάλλον ανάπτυξης, με πληθώρα λειτουργιών που επιταχύνουν την ανάπτυξη και βελτιστοποιούν την απόδοση των εφαρμογών. Αναπτύχθηκε από τον Taylor Otwell και κυκλοφόρησε για πρώτη φορά το 2011. Από την πρώτη του εμφάνιση, το Laravel έχει κερδίσει ευρεία αποδοχή στην κοινότητα των προγραμματιστών λόγω της ευελιξίας, της ευανάγνωστης δομής και των πλούσιων δυνατοτήτων που προσφέρει. Το Laravel προσφέρει ένα πλήθος λειτουργιών και εργαλείων που διευκολύνουν και επιταχύνουν την ανάπτυξη web εφαρμογών. Αυτές οι λειτουργίες περιλαμβάνουν:

- **Ενσωματωμένο σύστημα διαδρομών (Routing System)**: Επιτρέπει τη δημιουργία ευέλικτων και προσαρμόσιμων διαδρομών για τις εφαρμογές.
- **Μηχανισμός προτύπων Blade (Blade Templating Engine)**: Διευκολύνει τη δημιουργία δυναμικών προβολών και τη διαχείριση των δεδομένων που παρουσιάζονται στον χρήστη.
- **Σύστημα μετανάστευσης βάσης δεδομένων (Database Migration System)**: Επιτρέπει τη διαχείριση και την παρακολούθηση των αλλαγών στη δομή της βάσης δεδομένων με οργανωμένο και ελεγχόμενο τρόπο.
- **Εργαλείο γραμμής εντολών Artisan (Artisan CLI)**: Παρέχει εργαλεία και εντολές που αυτοματοποιούν κοινές εργασίες ανάπτυξης, όπως η δημιουργία κώδικα και η εκτέλεση δοκιμών.

Εικόνα 5 - Laravel logo [15]

Το Laravel προσφέρει πολλά πλεονεκτήματα που το καθιστούν δημοφιλές:

1. **Ευανάγνωστη και καθαρή δομή κώδικα**: Υιοθετεί το πρότυπο Model-View-Controller (MVC), το οποίο προωθεί τη διαχωριστική αρχιτεκτονική και διευκολύνει την ανάπτυξη και τη συντήρηση των εφαρμογών. Η καθαρή δομή του κώδικα καθιστά ευκολότερη την κατανόηση και την επέκταση αυτών των εφαρμογών.

2. **RESTful routing**: Προσφέρει ενσωματωμένη υποστήριξη για τη δημιουργία RESTful APIs, διευκολύνοντας την ανάπτυξη εφαρμογών που επικοινωνούν με άλλες υπηρεσίες και συστήματα. Συμβάλλει στο να οριστούν routes εύκολα και τα διαφορετικά αιτήματα HTTP να αλληλοεπιδρούν ανάλογα με πολλά endpoints.
3. **Ασφάλεια**: Παρέχει ενσωματωμένες λειτουργίες ασφαλείας, όπως προστασία από SQL injections, cross-site scripting (XSS), και cross-site request forgery (CSRF). Αυτές οι λειτουργίες διασφαλίζουν ότι οι εφαρμογές που αναπτύσσονται με το Laravel είναι προστατευμένες από κοινές διαδικτυακές απειλές.
4. **Γρήγορη ανάπτυξη**: Αυτό το framework προσφέρει μια κομψή και καθαρή σύνταξη που απλοποιεί τη διαδικασία ανάπτυξης διαδικτυακών εφαρμογών. Παρέχει μια ευρεία γκάμα εργαλείων, βιβλιοθηκών και προκατασκευασμένων στοιχείων, τα οποία βοηθούν τους προγραμματιστές να δημιουργήσουν γρηγορότερες εφαρμογές.
5. **Αντικειμενοστραφής χαρτογράφηση σχέσεων**: Το ενσωματωμένο ORM (Object-Relational Mapping) στο Laravel , γνωστό ως Eloquent, παρέχει έναν απλό και κατανοητό τρόπο αλληλεπίδρασης με τη βάση δεδομένων. Οι προγραμματιστές μπορούν να ορίζουν μοντέλα που αντιπροσωπεύουν τις βάσεις δεδομένων τους και να χρησιμοποιούν αντικείμενα αντί για παραδοσιακά SQL queries, κάνοντας τη διαχείριση των δεδομένων πιο ευέλικτη και ευανάγνωστη. Το Eloquent υποστηρίζει σχέσεις, δημιουργία ερωτημάτων, eager loading, και πολλά άλλα, γεγονός που μειώνει σημαντικά τον χρόνο που απαιτείται για εργασίες σχετικές με τη βάση δεδομένων.
6. **Blade templating engine**: Η μηχανή διαμόρφωσης Blade στο Laravel για ανάπτυξη διαδικτυακών εφαρμογών βοηθά τους προγραμματιστές να δημιουργήσουν επαναχρησιμοποιήσιμες και δυναμικές προβολές. Είναι πλούσια σε χαρακτηριστικά και αισθητική, προσφέροντας ισχυρές δομές ελέγχου για βρόχους, συνθήκες και κληρονομικότητα προτύπων. Το Blade διευκολύνει τη δημιουργία δυναμικών σελίδων HTML και την εισαγωγή λογικής στα views, διασφαλίζοντας ότι ο κώδικας παραμένει καθαρός και ευανάγνωστος.
7. **Ενσωμάτωση με υπηρεσίες τρίτων**: Αυτό το framework ενσωματώνεται με πολλές βιβλιοθήκες και υπηρεσίες τρίτων, προσφέροντας APIs για υπηρεσίες όπως ουρές, αποθήκευση στην κρυφή μνήμη, αποθήκευση στο cloud και πολλά άλλα. Αυτή η δυνατότητα προσφέρει στους προγραμματιστές να επεκτείνουν τις δυνατότητες των εφαρμογών τους εύκολα και γρήγορα, χωρίς να χρειάζεται να αναπτύξουν εκ νέου καινούριες λειτουργίες.
8. **Δοκιμές και γρήγορος εντοπισμός σφαλμάτων**: Υποστηρίζει εκτεταμένες δυνατότητες testing στον εντοπισμό σφαλμάτων και στη δοκιμή, όπως unit tests, feauture tests. Η δυνατότητα εύκολου testing βοηθά στην ανίχνευση και στην επίλυση σφαλμάτων κατά την διαδικασία της ανάπτυξης. Οι προγραμματιστές μπορούν να βεβαιωθούν ότι ο κώδικάς τους λειτουργεί όπως αναμένεται πριν από την κυκλοφορία μιας εφαρμογής μειώνοντας έτσι τις πιθανότητες για σφάλματα.

Σύμφωνα με τα πλεονεκτήματα που αναφέρθηκαν το framework αυτό διευκολύνει και δικαιώνει όχι μόνο την πλευρά των χρηστών αλλά και των προγραμματιστών. Καθίσταται κατάλληλο για τον σκοπό της εργασίας αυτής.

2.2 PHP

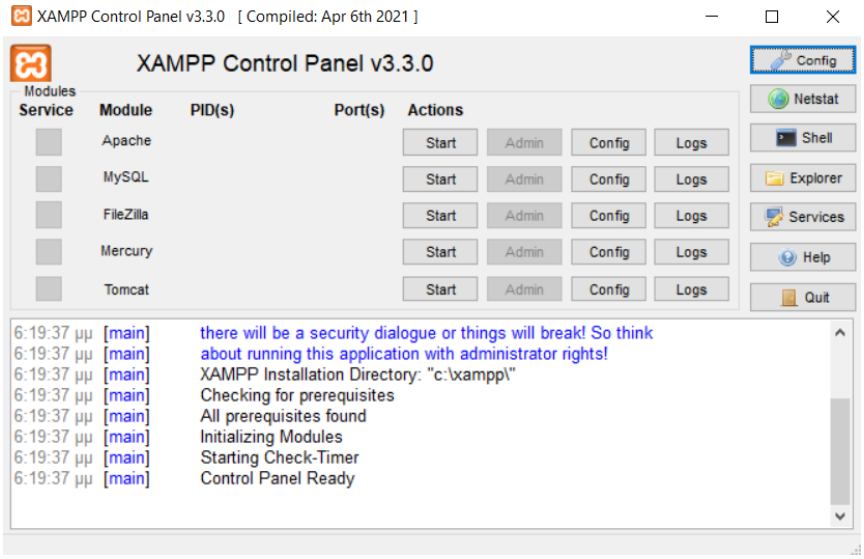
Η PHP [16] (Hypertext Preprocessor) είναι μια διαδεδομένη γλώσσα προγραμματισμού ειδικά σχεδιασμένη για την ανάπτυξη δυναμικών ιστοσελίδων και εφαρμογών ιστού. Είναι server-side γλώσσα, δηλαδή ο κώδικας εκτελείται στον διακομιστή και τα αποτελέσματα στέλνονται στον πελάτη. Η PHP έχει πολλές δυνατότητες, όπως ενσωμάτωση με βάσεις δεδομένων σαν την MySQL, PostgreSQL, και πολλές άλλες. Με τη χρήση της PHP, είναι δυνατή η επικοινωνία με αυτές τις βάσεις δεδομένων για την ανάκτηση, την ενημέρωση και την διαγραφή των δεδομένων, καθώς και την δημιουργία δυναμικών ιστοσελίδων που εξυπηρετούν τις ανάγκες των χρηστών με βάση τα δεδομένα. Επιπλέον η PHP υποστηρίζει διάφορες λειτουργίες όπως διαχείριση cookies και sessions, αρχείων, και αποστολή email. Το δυνατότερο χαρακτηριστικό της είναι η δυνατότητά της να ενσωματώνεται στον HTML κώδικα, επιτρέποντας έτσι την δημιουργία σύνθετων και δυναμικών ιστοσελίδων με ελάχιστο κώδικα.

2.3 Servers

Για την ανάπτυξη μιας εφαρμογής και της δοκιμής αυτηνής, είναι απαραίτητο να χρησιμοποιηθεί ένας διακομιστής. Ένας διακομιστής παρέχει το περιβάλλον όπου εκτελείται ο κώδικας και εκεί αποστέλλονται τα αποτελέσματα στον πελάτη.

2.3.1 Apache Server από πακέτο XAMPP

Ο Apache Server [17] , που περιλαμβάνεται στο πακέτο ανάπτυξης XAMPP, είναι ένας από τους πιο δημοφιλείς διακομιστές HTTP. Το XAMPP είναι μια διανομή Apache που περιλαμβάνει επίσης MySQL, PHP και Perl. Είναι εύκολο στην εγκατάσταση και χρήση, κάνοντάς το ιδανικό για προγραμματιστές που θέλουν να αναπτύξουν και να δοκιμάσουν τις εφαρμογές τους τοπικά. Ο Apache Server παρέχει ένα σταθερό και αξιόπιστο περιβάλλον εκτέλεσης, επιτρέποντας στους προγραμματιστές να εντοπίζουν σφάλματα και να βελτιστοποιούν τις εφαρμογές τους πριν από την ανάρτησή τους σε έναν ζωντανό διακομιστή. Προκειμένου να ξεκινούμε τις υπηρεσίες που θέλουμε (συγκεκριμένα για την εφαρμογή θέλουμε τον Apache και MySQL) πατάμε το κουμπί start (κάτω από την σήμανση Actions) δίπλα από κάθε υπηρεσία.



Εικόνα 6 - Το εργαλείο XAMPP [18]

2.3.2 Laravel's built in server

Ο ενσωματωμένος server του Laravel [19] είναι ένα εργαλείο που προσφέρει εύκολη και γρήγορη εκτέλεση των εφαρμογών Laravel σε τοπικό περιβάλλον. Με την εντολή 'php artisan serve' ξεκινάει ένας απλός server ανάπτυξης, διευκολύνοντας έτσι την δοκιμή και την ανάπτυξη των εφαρμογών. Αυτός ο server είναι ιδανικός για τοπική ανάπτυξη και debugging, παρέχοντας άμεση ανατροφοδότηση στις αλλαγές του κώδικα.

2.4 Εργαλείο ανάπτυξης Composer

Το Composer [20] είναι το εργαλείο διαχείρισης εξαρτήσεων για την PHP και χρησιμοποιείται για την διαχείριση των εξαρτήσεων του back-end. Επιτρέπει στους προγραμματιστές να δηλώνουν τις βιβλιοθήκες που χρειάζονται για το έργο τους και να τις εγκαθιστούν εύκολα. Το Composer διασφαλίζει ότι όλες οι εξαρτήσεις του έργου είναι συμβατές και ενημερωμένες, διατηρώντας τη συνοχή του έργου και μειώνοντας τα πιθανά προβλήματα συμβατότητας. Στο Laravel, το Composer χρησιμοποιείται ευρέως για τη διαχείριση των πακέτων και βιβλιοθηκών που απαιτούνται για την ανάπτυξη των εφαρμογών.

Εικόνα 7 - Composer logo [20]

2.5 Εργαλείο ανάπτυξης Artisan

Το Artisan [21] είναι η γραμμή εντολών του Laravel που προσφέρει μια σειρά από χρήσιμες εντολές για την ανάπτυξη και τη διαχείριση εφαρμογών. Μέσω του Artisan, μπορούν να δημιουργηθούν νέα μοντέλα, controllers, migrations και άλλες δομές μέσω διάφορων συγκεκριμένων εντολών που μπορούν να γραφτούν στο terminal του εκάστοτε IDE. Για παράδειγμα η εντολή 'php artisan serve' εκκινεί τον ενσωματωμένο server για την τοπική ανάπτυξη του framework Laravel, ενώ άλλες εντολές όπως 'php artisan migrate' εκτελούν τις εκκρεμείς μεταναστεύσεις (migrations) στη βάση δεδομένων κτλ.

2.6 Εργαλείο ανάπτυξης Node Package Manager (NPM)

Το Node Package Manager [22] (NPM) είναι ο διαχειριστής πακέτων για την JavaScript και χρησιμοποιείται για τη διαχείριση των εξαρτήσεων του front-end. Με το NPM, δίνεται η δυνατότητα να γίνει η εγκατάσταση, η αναβάθμιση και η διαχείριση των πακέτων και των βιβλιοθηκών που είναι απαραίτητα για την ανάπτυξη εφαρμογών. Στην συγκεκριμένη εργασία, το NPM χρησιμοποιείται για τη διαχείριση των εξαρτήσεων του εργαλείου Vite, το οποίο παρέχει ταχύτατο build και development server . Η εντολή 'npm run dev' εκκινεί το Vite, επιτρέποντας την γρήγορη και αποδοτική ανάπτυξη και testing του front-end.

3. Κεφάλαιο 3ο: Αρχιτεκτονική Εφαρμογής

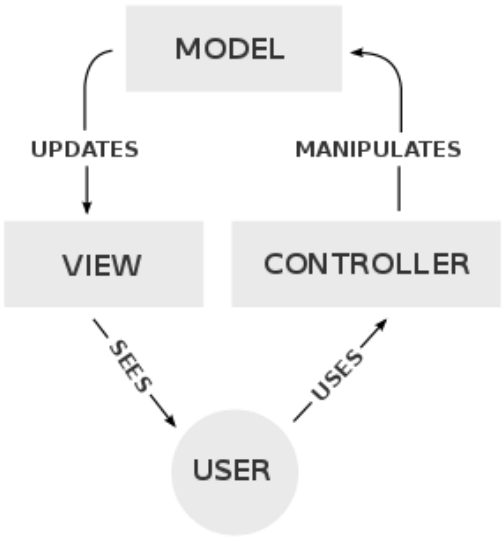
Στο κεφάλαιο αυτό περιγράφεται η αρχιτεκτονική της εφαρμογής για την κατανόηση της.

3.1 Μοντέλο αρχιτεκτονικής

Το μοντέλο της αρχιτεκτονικής που χρησιμοποιείται είναι το MVC [23] (Model-View-Controller). Είναι μία αρχιτεκτονική λογισμικού και η χρήση του προορίζεται για την αλληλεπίδραση του χρήστη με το πρόγραμμα. Αυτό το μοντέλο χωρίζεται σε τρία διασυνδεδεμένα μέρη:

- Το μοντέλο (Model) διαχειρίζεται τα δεδομένα, συμπεριλαμβανομένης της αποθήκευσης και της ανάκτησης των πληροφοριών από την βάση δεδομένων. Τα μοντέλα που χρησιμοποιούνται για την συγκεκριμένη εφαρμογή είναι: User, Product και Message. Το model ενημερώνει τις αντίστοιχες προβολές (Views) και τους ελεγκτές (Controllers) όταν υπάρχει αλλαγή στα δεδομένα, επιτρέποντας στις προβολές να ενημερώνουν την γραφική απεικόνιση.
- Οι προβολές (Views) παρουσιάζουν τα δεδομένα στον χρήστη με γραφικό τρόπο. Οι προβολές αντιπροσωπεύουν τις σελίδες ιστότοπου, τα στοιχεία των προϊόντων, τις διεπαφές του χρήστη κτλ. παρουσιάζοντας γραφικά τις πληροφορίες από το μοντέλο. Συγκεκριμένα στην εφαρμογή, ο φάκελος views περιέχει 7 υποφακέλους αντίστοιχα για τα messages, products, profile, user, admin, auth, layouts (menu) και 2 αρχεία Views για το welcome και το dashboard page.
- Οι ελεγκτές (Controllers) διαχειρίζονται τις αιτήσεις του χρήστη που αλληλοεπιδρούν με το μοντέλο και τις προβολές. Ο ελεγκτής επεξεργάζεται τις αιτήσεις για την προβολή των προϊόντων, την δημιουργία νέων καταχωρίσεων των προϊόντων, την ανταλλαγή μηνυμάτων μεταξύ των χρηστών κτλ. Επίσης ο ελεγκτής στέλνει εντολές στο μοντέλο (model) και ενημερώνει την κατάσταση του. Περισσότερες πληροφορίες για τους controllers που χρησιμοποιήθηκαν θα αναλυθούν παρακάτω.

Η συγκεκριμένη αρχιτεκτονική είναι κοινώς χρησιμοποιούμενη στο Laravel framework και αποτελεί βάση για ανάπτυξη διαδικτυακών εφαρμογών με το συγκεκριμένο framework.



Εικόνα 8 - Η λειτουργία του Model-View-Controller (MVC). Ο χρήστης στέλνει εντολή στον controller, ο οποίος διαχειρίζεται τα δεδομένα μέσω του μοντέλου (model) και με την σειρά του ενημερώνει την προβολή (view), προσφέροντας στον χρήστη το αποτέλεσμα.[24]

3.1.1 Ανάλυση MVC για την συγκεκριμένη εφαρμογή

Στην συγκεκριμένη εφαρμογή που υλοποιείται, οι φάκελοι Controllers, Models και Views παίζουν σημαντικό ρόλο τόσο για το backend όσο και για το frontend της εφαρμογής. Για το backend χρησιμοποιούνται οι controllers και τα models ενώ για το frontend τα views. Αναλυτικότερα:

Controllers

Συγκεκριμένα στην εφαρμογή υπάρχουν 8 βασικοί controllers:

- Ο controller «Admin Controller» είναι υπεύθυνος για τη διαχείριση της σελίδας ελέγχου (dashboard) του διαχειριστή. Συγκεκριμένα, η μέθοδος dashboard φορτώνει όλους τους χρήστες από τη βάση δεδομένων μαζί με τα προϊόντα τους και τα περνάει στην προβολή (view) 'admin.dashboard'.
- ο βασικός controller ονόματι «Controller», λειτουργεί ως ένας μητρικός controller από τον οποίο κληρονομούνται οι υπόλοιποι controllers της εφαρμογής και «εξοπλίζει» κάθε controller με τις βασικές δυνατότητες εξουσιοδότησης (authorization) και επικύρωσης (validation) για τα http αιτήματα.
- Ο controller «Favorite Controller» χειρίζεται την προσθήκη και την αφαίρεση προϊόντων από τα αγαπημένα ενός χρήστη. Η μέθοδος store προσθέτει ένα προϊόν στα αγαπημένα, ενώ η delete το αφαιρεί. Πριν προσθέσει ένα προϊόν, ελέγχει αν είναι ήδη στα αγαπημένα του χρήστη.
- ο «Home controller», ο οποίος είναι υπεύθυνος για την διαχείριση των αιτημάτων που σχετίζονται με την σελίδα dashboard της εφαρμογής. Οι χρήστες θα πρέπει να έχουν συνδεθεί προκειμένου να έχουν πρόσβαση σε αυτή την σελίδα της εφαρμογής και επιστρέφει την αντίστοιχη προβολή για την εμφάνισή της. Χρησιμοποιεί 2 μεθόδους:
  1. __construct(): Αυτή η μέθοδος είναι ο constructor του controller. Ορίζει ένα middleware με το όνομα 'auth'. Το middleware 'auth' διασφαλίζει ότι ο χρήστης πρέπει να έχει πραγματοποιήσει είσοδο (σύνδεση) στην εφαρμογή προτού μπορέσει να έχει πρόσβαση σε οποιονδήποτε δρομέα του Home Controller.
  2. index(): Αυτή η μέθοδος είναι υπεύθυνη για την εμφάνιση της σελίδας της εφαρμογής. Επιστρέφει την προβολή με το όνομα 'home'. Η προβολή 'home' έχει οριστεί στον φάκελο των προβολών της εφαρμογής και περιέχει το view που απαιτείται για την απεικόνιση της αρχικής σελίδας (home.blade.php).
- ο «Message controller» διαχειρίζεται τα μηνύματα μεταξύ των χρηστών. Οι 7 μέθοδοι του κάνουν τα εξής:
  1. index(): Ανακτά τα μηνύματα του συνδεδεμένου χρήστη, όλους τους χρήστες και ταξινομεί τους χρήστες βάσει των πιο πρόσφατων ανταλλαγών μηνυμάτων. Επίσης, υπολογίζει τον αριθμό των μη αναγνωσμένων μηνυμάτων για κάθε χρήστη.
  2. send(): Επικυρώνει τα δεδομένα εισόδου, δημιουργεί ένα νέο μήνυμα, το αποθηκεύει και μεταδίδει ένα γεγονός (event) νέου μηνύματος.
  3. triggerNewMessageEvent(): Επικυρώνει τα δεδομένα εισόδου, δημιουργεί ένα νέο μήνυμα (χωρίς αποθήκευση στη βάση δεδομένων) και μεταδίδει ένα γεγονός νέου μηνύματος, όπως και η μέθοδος send(), αλλά για να ειδοποιήσει τον παραλήπτη για το καινούριο μήνυμα και να εμφανιστεί το εικονίδιο ειδοποίησης.

4. getMessagesForReceiver(): Ανακτά τα μηνύματα για έναν συγκεκριμένο παραλήπτη, προσθέτει το όνομα του αποστολέα σε κάθε μήνυμα και επιστρέφει μια απάντηση σε JSON μορφή.
5. markAsRead(): Σημειώνει ένα μήνυμα ως αναγνωσμένο, αν ανήκει στον συνδεδεμένο χρήστη.
6. Count(): Μετρά τα μη αναγνωσμένα μηνύματα από έναν συγκεκριμένο αποστολέα προς τον συνεδεμένο χρήστη.
7. deleteMessage(): Διαγράφει ένα μήνυμα, αν ανήκει στον συνδεδεμένο χρήστη.

- ο «Product controller» είναι υπεύθυνος για τη διαχείριση των προϊόντων στην εφαρμογή. Συγκεκριμένα περιέχει 6 μεθόδους:
  1. store(): Αυτή η μέθοδος αποθηκεύει ένα νέο προϊόν στη βάση δεδομένων. Πρώτα επικυρώνει τα δεδομένα εισόδου, συμπεριλαμβανομένου του ελέγχου του αρχείου εικόνας. Στη συνέχεια, δημιουργεί το προϊόν και αποθηκεύει τη φωτογραφία του προϊόντος στον διακομιστή.
  2. show(): Αυτή η μέθοδος εμφανίζει τη λεπτομερή προβολή ενός συγκεκριμένου προϊόντος.
  3. index(): Αυτή η μέθοδος επιστρέφει την προβολή που περιέχει τα προϊόντα που έχει δημοσιεύσει ο τρέχων χρήστης, καθώς και προϊόντα από άλλους χρήστες, με δυνατότητα φιλτραρίσματος και ταξινόμησης.
  4. edit(): Αυτή η μέθοδος επιστρέφει την προβολή για την επεξεργασία ενός συγκεκριμένου προϊόντος. Μόνο αν ο τρέχων χρήστης είναι ο κάτοχος του προϊόντος προς πώληση.
  5. update(): Αυτή η μέθοδος χειρίζεται το αίτημα ενημέρωσης ενός υπάρχοντος προϊόντος. Επικυρώνει τα δεδομένα εισόδου και ενημερώνει το προϊόν στη βάση δεδομένων, συμπεριλαμβανομένου του ανέβασματος νέας εικόνας.
  6. destroy(): Αυτή η μέθοδος χειρίζεται το αίτημα διαγραφής ενός προϊόντος. Ελέγχει αν ο τρέχων χρήστης είναι ο κάτοχος του προϊόντος προς πώληση πριν προχωρήσει στη διαγραφή και διαγράφει το προϊόν από τον διακομιστή.
- ο «Profile controller» είναι υπεύθυνος για τη διαχείριση των προφίλ των τρέχοντα συνδεδεμένων χρηστών στην εφαρμογή. Αναλαμβάνει την προβολή, την επεξεργασία, την ενημέρωση και την διαγραφή του προφίλ. Αναλυτικά οι 4 μέθοδοι:
  1. show(): Αυτή η μέθοδος εμφανίζει το προφίλ του τρέχοντος συνδεδεμένου χρήστη. Ανακτά τον χρήστη και τα αγαπημένα του προϊόντα (αν έχει) από το αίτημα και επιστρέφει την προβολή profile.show, περνώντας τα δεδομένα του χρήστη στην προβολή.
  2. edit(): Αυτή η μέθοδος επιστρέφει την προβολή για την επεξεργασία του προφίλ του τρέχοντος συνδεδεμένου χρήστη. Ανακτά τα δεδομένα του χρήστη από το authentication και επιστρέφει την προβολή profile.edit.
  3. update(): Αυτή η μέθοδος χειρίζεται την ενημέρωση του προφίλ του χρήστη. Επικυρώνει τα δεδομένα εισόδου και ενημερώνει τα στοιχεία του χρήστη στη βάση δεδομένων. Αν υπάρχει νέο αρχείο φωτογραφίας, το αποθηκεύει και ενημερώνει το πεδίο photo.
  4. destroy(): Αυτή η μέθοδος διαγράφει τον λογαριασμό του τρέχοντος συνδεδεμένου χρήστη. Μετά τη διαγραφή του χρήστη, ανακατευθύνει τον χρήστη στο welcome page.

- ο «User Profile controller» είναι υπεύθυνος για την εμφάνιση του προφίλ ενός συγκεκριμένου χρήστη. Περιέχει μία μέθοδο show(). Αυτή η μέθοδος ανακτά το προφίλ ενός συγκεκριμένου χρήστη από τη βάση δεδομένων και επιστρέφει τα δεδομένα είτε ως JSON είτε ως Blade προβολή, ανάλογα με το αίτημα. Εάν το αίτημα αναμένει JSON απάντηση (δυναμικό αίτημα AJAX call), η μέθοδος ελέγχει με το expectsJson(). Εάν είναι true, επιστρέφει τα δεδομένα του χρήστη ως JSON μορφή. Εάν το αίτημα δεν αναμένει JSON, επιστρέφει την Blade προβολή (user.profile.php) με τα δεδομένα του χρήστη.

Οι υπόλοιποι controllers που βρίσκονται στον φάκελο auth, συνολικά 6, περιλαμβάνει τους controllers για την εγγραφή και την σύνδεση των χρηστών και την ανάκτηση του κωδικού (forget password) στην περίπτωση που κάποιος χρήστης ξεχάσει τον κωδικό, να μπορεί να τον επαναφέρει με έναν νέο.

Models

Στην εφαρμογή αυτή έχουμε τα εξής τρία βασικά μοντέλα:

- **Μοντέλο Χρήστη (User)**
  Το μοντέλο χρήστη είναι υπεύθυνο για τη διαχείριση των πληροφοριών των χρηστών της εφαρμογής και αναπαριστά έναν χρήστη στο σύστημα. Τα πεδία που περιλαμβάνονται είναι:
  1. name: Το όνομα του χρήστη.
  2. email: Η διεύθυνση email του χρήστη.
  3. password: Το κρυπτογραφημένο password του χρήστη.
  4. role: Ο ρόλος του χρήστη στην εφαρμογή (admin ή user).
  5. photo: Η φωτογραφία του χρήστη.
  6. interests: Τα ενδιαφέροντά του.

  Επιπλέον, το μοντέλο περιλαμβάνει σχέσεις με άλλα μοντέλα, όπως τα μηνύματα και τα προϊόντα.

- **Μοντέλο Προϊόντος (Product)**

  Το μοντέλο προϊόντος αναλαμβάνει τη διαχείριση των πληροφοριών που αφορούν τα προϊόντα που προσφέρονται στην εφαρμογή και αναπαριστά ένα προϊόν που προσφέρεται στο σύστημα. Αυτό περιλαμβάνει:

  1. name: Το όνομα του προϊόντος.
  2. description: Η περιγραφή του προϊόντος.
  3. price: Η τιμή του προϊόντος.
  4. user_id: Το αναγνωριστικό του χρήστη που δημιούργησε το προϊόν.
  5. photo_product: Η φωτογραφία του προϊόντος.
  6. category: Η κατηγορία του προϊόντος.
  7. shipping_method: Ο τρόπος αποστολής του προϊόντος.
  8. map: Η τοποθεσία του προϊόντος.
  9. availability: Η διαθεσιμότητα του προϊόντος.

  Το μοντέλο αυτό έχει σχέση με το μοντέλο του χρήστη, καθώς και μια σχέση πολλών προς πολλά με τα αγαπημένα προϊόντα των χρηστών.

- **Μοντέλο Μηνύματος (Message)**

Το μοντέλο μηνύματος διαχειρίζεται τις επικοινωνίες μεταξύ των χρηστών της εφαρμογής και αναπαριστά ένα μήνυμα που ανταλλάσσεται μεταξύ των χρηστών του συστήματος. Περιλαμβάνει τα πεδία:

1. sender_id: Το αναγνωριστικό του αποστολέα του μηνύματος.
2. receiver_id: Το αναγνωριστικό του παραλήπτη του μηνύματος.
3. message_content: Το περιεχόμενο του μηνύματος.
4. is_read: Ένδειξη εάν το μήνυμα έχει διαβαστεί ή όχι.

Αυτό το μοντέλο περιλαμβάνει σχέσεις με τα μοντέλα των χρηστών, όπου κάθε μήνυμα ανήκει σε έναν αποστολέα και σε έναν παραλήπτη.

Views

Οι προβολές (views) παρουσιάζουν τα δεδομένα στον τελικό χρήστη με γραφικό τρόπο. Όπως αναφέρθηκε πριν ο φάκελος αυτός διαιρείται σε 7 επιμέρους υποφακέλους που αντιστοιχούν σε διάφορα τμήματα της εφαρμογής. Κάθε υποφάκελος παρουσιάζει τα δεδομένα ανάλογα με το συγκεκριμένο τμήμα της εφαρμογής. Παρακάτω γράφεται μια σύντομη περιγραφή κάθε σελίδας-αρχείου σε κάθε συγκεκριμένο φάκελο:

- **Σελίδα Admin στον φάκελο "Admin":**

Η σελίδα αυτή παρουσιάζει έναν διαχειριστικό πίνακα (μόνο για τον χρήστη admin) διαχείρισης χρηστών και προϊόντων. Περιλαμβάνει έναν πίνακα που παρουσιάζει πληροφορίες για τους χρήστες, όπως το όνομα, το email, το ρόλο κ.λπ., και τα προϊόντα που έχουν δημιουργήσει. Επιπλέον, παρέχει κουμπιά ενεργειών για τη διαγραφή προϊόντων ή χρηστών.

- **Menu επίλογων στον φάκελο "layout":**

Το μενού αυτό ο χρήστης μπορεί να το βρει πάνω δεξιά στην εφαρμογή. Περιλαμβάνει την σύνδεση με το Bootstrap για το σχεδιασμό και την αρχικοποίηση της υπηρεσίας Pusher για τις real-time ειδοποιήσεις. Προσφέρει λειτουργίες μενού όπως την εγγραφή και την σύνδεση, την προβολή προφίλ του συνδεδεμένου χρήστη , την αποσύνδεση και τα μηνύματα.

- **Σελίδα Μηνυμάτων στον φάκελο "messages":**

Είναι η σελίδα ανταλλαγής μηνυμάτων. Έχει τρεις κύριες ενότητες: την πληροφορία του προφίλ του χρήστη (όταν επιλεχθεί) , το ιστορικό των μηνυμάτων , τη λίστα χρηστών και τη φόρμα αποστολής μηνύματος. Η σελίδα αυτή προσφέρει ανανέωση του ιστορικού μηνυμάτων σε πραγματικό χρόνο, το δυναμικό φιλτράρισμα των χρηστών και την αποστολή μηνυμάτων.

- **Σελίδα Προϊόντων στον φάκελο "products":**

Αρχείο show.blade.php: Αυτή η σελίδα προβάλλει λεπτομέρειες για ένα προϊόν, συμπεριλαμβανομένου του ονόματος, της περιγραφής, της τιμής, της κατηγορίας, του τρόπου αποστολής, της τοποθεσίας και της διαθεσιμότητας του προϊόντος. Επίσης περιέχει ένα κουμπί που επιτρέπει στους χρήστες να προσθέσουν το προϊόν στα

αγαπημένα τους ή να το αφαιρέσουν από τα αγαπημένα τους, ανάλογα με το εάν το προϊόν είναι ήδη στα αγαπημένα τους ή όχι. Υπάρχουν επίσης κουμπιά για να στείλουν μήνυμα στον χρήστη που πρόσθεσε το προϊόν ή να επεξεργαστούν την αγγελία του προϊόντος.

Αρχείο edit.blade.php: Επιτρέπει στους χρήστες να επεξεργαστούν τις πληροφορίες των προϊόντων που έχουν ήδη ανεβάσει. Παρουσιάζεται μία φόρμα με πεδία για την επεξεργασία του ονόματος, της περιγραφής, της τιμής, της φωτογραφίας, της κατηγορίας, του τρόπου αποστολής, της τοποθεσίας και της διαθεσιμότητας του προϊόντος. Οι χρήστες μπορούν να επιλέξουν να αφαιρέσουν την τρέχουσα φωτογραφία του προϊόντος, αν υπάρχει, και να ανεβάσουν μια νέα φωτογραφία. Επίσης, παρέχεται η δυνατότητα διαγραφής του προϊόντος ή της επιστροφής στον πίνακα ελέγχου.

- **Σελίδα Προφίλ συνδεδεμένου χρήστη στον φάκελο “profile”**

Αρχείο show.blade.php: Αυτή η σελίδα προφίλ παρουσιάζει τις πληροφορίες του χρήστη και τα αγαπημένα του προϊόντα. Στο προφίλ εμφανίζονται το όνομα, το email, τα ενδιαφέροντα και η ημερομηνία εγγραφής του χρήστη, καθώς και επιλογές για επεξεργασία των πληροφοριών του προφίλ και την διαγραφή του λογαριασμού του. Επίσης, εμφανίζονται τα αγαπημένα προϊόντα του χρήστη με επιλογές για προβολή λεπτομερειών και αφαίρεση από τα αγαπημένα.

Αρχείο edit.blade.php: Σε αυτή τη σελίδα ο χρήστης ενημερώνει τα προσωπικά του στοιχεία. Περιέχει μία φόρμα που περιλαμβάνει πεδία για το όνομα, το email, τη φωτογραφία του χρήστη, τα ενδιαφέροντα του και την αλλαγή του κωδικού πρόσβασης του. Ο χρήστης μπορεί να αλλάξει τα πεδία που επιθυμεί και να πατήσει το κουμπί "Update" για να αποθηκεύσει τις αλλαγές.

- **Σελίδα Προφίλ συγκεκριμένου χρήστη στον φάκελο “user”**

Αυτή η σελίδα προφίλ εμφανίζει τις πληροφορίες ενός χρήστη, συμπεριλαμβανομένου του ονόματος, του email, των ενδιαφερόντων και της ημερομηνίας εγγραφής. Επίσης εμφανίζει τυχόν προϊόντα που έχει προσθέσει ο χρήστης στην πλατφόρμα, αν υπάρχουν. Η φωτογραφία του προφίλ του χρήστη επίσης εμφανίζεται εάν είναι διαθέσιμη. Σε περίπτωση που ο χρήστης έχει προσθέσει προϊόντα, αυτά εμφανίζονται ως σύνδεσμοι με το όνομά τους, οδηγώντας τον χρήστη στις αντίστοιχες σελίδες λεπτομερειών προϊόντος.

Οι σελίδες welcome και dashboard είναι οι κυρίες σελίδες που βλέπει ο χρήστης όταν εισέρχεται στην εφαρμογή.

- **Σελίδα Dashboard (Home Page)**

Αυτή η σελίδα εμφανίζει ένα σύνολο πληροφοριών και λειτουργιών για τον χρήστη. Αρχικά, εμφανίζεται ένα μήνυμα καλωσορίσματος και, εάν υπάρχουν αναπάντητα μηνύματα, εμφανίζεται και μια ειδοποίηση για αυτά. Υπάρχουν επίσης λειτουργίες για την προσθήκη νέου προϊόντος, την εμφάνιση των προϊόντων του χρήστη και των προϊόντων άλλων χρηστών. Η "Προσθήκη Προϊόντος" περιλαμβάνει μία φόρμα που επιτρέπει στον χρήστη να καταχωρήσει ένα νέο προϊόν. Οι τίτλοι "Τα Προϊόντα μου" και "Προϊόντα από άλλους χρήστες" εμφανίζουν τα προϊόντα που έχουν προστεθεί στο

σύστημα. Ο χρήστης μπορεί να αναζητήσει προϊόντα με βάση το όνομα ή να τα ταξινομήσει ανάλογα με την ημερομηνία δημιουργίας αυτών.

- **Σελίδα καλωσορίσματος (Welcome Page)**

  Είναι η πρώτη προβολή της εφαρμογής marketplace. Εμφανίζει ένα μήνυμα καλωσορίσματος με τον τίτλο "Welcome to the Marketplace Web Application" και τους δύο συνδέσμους για σύνδεση (Log in) και εγγραφή (Register) στην εφαρμογή.

Υπάρχει και ο φάκελος “auth” ο οποίος περιέχει τα αρχεία σχετικά με την πιστοποίηση των χρηστών σε μια εφαρμογή. Αυτά τα αρχεία περιλαμβάνουν τις φόρμες σύνδεσης, εγγραφής και επαναφοράς κωδικού πρόσβασης. Περιλαμβάνει αρχεία όπως μια σελίδα φόρμας σύνδεσης ενός χρήστη, σελίδα φόρμας εγγραφής ενός νέου χρήστη κτλ.

3.2 Υπηρεσία Mailtrap.io για αποστολή email

Το Mailtrap.io [25] είναι μια cloud υπηρεσία που παρέχει ένα τοπικό περιβάλλον για τη δοκιμή και τον εντοπισμό σφαλμάτων σε email χωρίς να στέλνονται πραγματικά email στους τελικούς χρήστες. Με το Mailtrap, το οποίο το χρησιμοποιούν οι προγραμματιστές, μπορούν να ανακατευθύνουν όλα τα εξερχόμενα email από τις εφαρμογές τους σε μια εικονική θυρίδα αλληλογραφίας όπου μπορούν να τα δουν, να τα αναλύσουν και να τα δοκιμάσουν, προτού τα στείλουν στους πραγματικούς παραλήπτες. Δηλαδή γίνεται προσομοίωση αποστολής email, αναλύεται το περιεχόμενο και είναι το ιδανικό εργαλείο για εντοπισμό σφαλμάτων. Συγκεκριμένα στην εφαρμογή χρησιμοποιούμε αυτή την υπηρεσία για να κάνουμε testing την αποστολή των email όταν ένας χρήστης έχει ξεχάσει τον κωδικό και θέλει να τον ανακτήσει.

Εικόνα 9 - Mailtrap.io logo [26]

3.3 JSON (JavaScript Object Notation)

Το JSON [27] είναι ένα ανοικτό, ελαφρύ μορφότυπο δεδομένων που βασίζεται σε κείμενο και χρησιμοποιείται κυρίως για τη μεταφορά δεδομένων μεταξύ ενός διακομιστή και ενός πελάτη στο διαδίκτυο. Τα δεδομένα σε μορφή JSON αναπαρίστανται ως ζευγάρια "κλειδί-τιμή". Στην εφαρμογή το JSON χρησιμοποιείται στην:

- **Ανταλλαγή δεδομένων**: Χρησιμοποιείται για τη μεταφορά δεδομένων μεταξύ του frontend και του backend της εφαρμογής. Για παράδειγμα, όταν ο χρήστης κάνει αναζήτηση για προϊόντα, το backend επιστρέφει τα αποτελέσματα σε μορφή JSON κτλ.
- **Αποθήκευση δεδομένων**: Χρησιμοποιείται για την αποθήκευση δεδομένων στη βάση δεδομένων της MySQL. Οι πληροφορίες των προϊόντων, των χρηστών και των μηνυμάτων είναι σε μορφή JSON.
- **Αλληλεπίδραση με τα API**: Το JSON χρησιμοποιείται για την αλληλεπίδραση με API. Πολλά APIs επιστρέφουν δεδομένα σε μορφή JSON. Όταν ένας χρήστης θέλει να ανακτήσει τα δεδομένα ενός προϊόντος από το API της εφαρμογής, γίνεται ένα αίτημα HTTP στο αντίστοιχο endpoint του API και επιστρέφεται ένα αντικείμενο JSON που περιέχει τα στοιχεία του προϊόντος. Έχει την εξής μορφή:

```
{
   "id": 1,
   "name": "Mobile Phone",
   "price": 250,
   "description": "Excellent condition and affordable price."
         ... more fields etc.
}
```

3.4 Πρωτόκολλο HTTP (Hypertext Transfer Protocol)

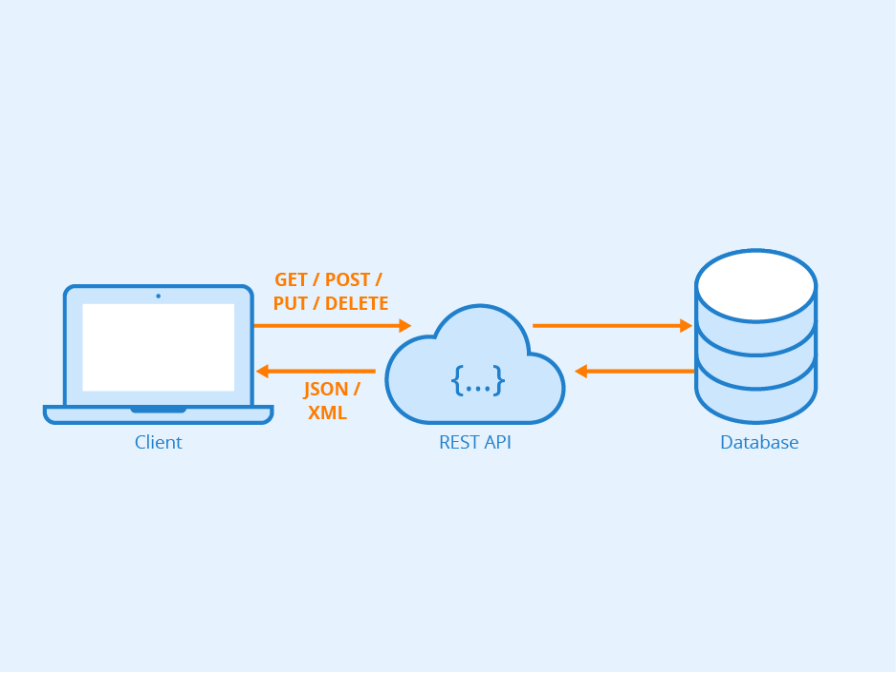
Το HTTP [28] είναι ένα πρωτόκολλο επικοινωνίας που χρησιμοποιείται για τη μεταφορά πληροφοριών. Καθορίζει τον τρόπο με τον οποίο μεταδίδονται τα μηνύματα, καθώς και τον τρόπο με τον οποίο οι εξυπηρετητές και οι web browsers πρέπει να ανταποκρίνονται στα αιτήματα που λαμβάνουν. Αποτελεί τον κύριο τρόπο επικοινωνίας μεταξύ πελάτη και διακομιστή, όπου οι αιτήσεις από τον πελάτη υποβάλλονται στο διακομιστή, ο οποίος παρέχει τις απαντήσεις των αιτημάτων αυτών.

3.5 API (Application Programming Interface)

Το API [29] (Application Programming Interface) είναι ένα σύνολο από κανόνες που επιτρέπουν σε δύο ή περισσότερα λογισμικά συστήματα να επικοινωνούν μεταξύ τους. Τα APIs μπορούν να χρησιμοποιηθούν για να διευκολύνουν τη σύνδεση και την αλληλεπίδραση μεταξύ διαφορετικών συστημάτων λογισμικού, προσφέροντας προκαθορισμένες λειτουργίες και υπηρεσίες που επιτρέπουν την ανταλλαγή δεδομένων και την εκτέλεση λειτουργιών. Τα APIs βρίσκονται παντού στην τεχνολογία σήμερα και αποτελούν την βάση πολλών σύγχρονων εφαρμογών και υπηρεσιών, όπως τα κοινωνικά δίκτυα, οι διαδικτυακές πληρωμές, οι υπηρεσίες cloud κτλ.

3.6 REST API (Representational State Transfer Application Programming Interface)

Το REST API [30] είναι ένα είδος API που ακολουθεί τις αρχές του REST (Representational State Transfer), μιας αρχιτεκτονικής σχεδίασης για διαδικτυακές εφαρμογές. Τα REST APIs χρησιμοποιούν το πρωτόκολλο HTTP για τη μετάδοση δεδομένων και επιτρέπουν τη δημιουργία, την ανάγνωση, την ενημέρωση και τη διαγραφή δεδομένων σε έναν διακομιστή. Οι πόροι που προσφέρονται από ένα REST API αναπαρίστανται ως URIs (Uniform Resource Identifiers) και μεταφέρονται συνήθως μέσω του πρωτοκόλλου HTTP, με τα δεδομένα να επιστρέφονται συνήθως σε μορφή JSON.



Εικόνα 10 - Ένας REST πελάτης μπορεί να αλληλοεπιδρά με κάθε πόρο αποστέλλοντας ένα αίτημα HTTP.[31]

Οι κύριες μέθοδοι που χρησιμοποιούνται σε ένα REST API περιλαμβάνουν:

- GET: Ανάκτηση δεδομένων από τον διακομιστή.
- POST: Δημιουργία νέων δεδομένων στον διακομιστή.
- PUT: Ενημέρωση υπαρχόντων δεδομένων στον διακομιστή.
- DELETE: Διαγραφή δεδομένων από τον διακομιστή.

Τα REST API, όταν σχεδιάζονται και υλοποιούνται με βάση τις αρχές του REST, αναφέρονται ως RESTful APIs. Τα RESTful APIs [32] παρέχουν μια πιο ευέλικτη και επεκτάσιμη προσέγγιση για την κατασκευή διαδικτυακών υπηρεσιών. Χρησιμοποιώντας τις μεθόδους που αναφέρθηκαν παραπάνω, τα RESTful APIs διευκολύνουν την πρόσβαση και τη διαχείριση των πόρων στον διακομιστή.

Τα RESTful API περιλαμβάνουν συνήθως "endpoints" (σημεία πρόσβασης) που αντιστοιχούν σε συγκεκριμένες λειτουργίες ή δεδομένα στον διακομιστή. Αυτά τα "endpoints" [33] είναι συγκεκριμένα URL που προσδιορίζουν τον τρόπο πρόσβασης σε διάφορες λειτουργίες του API, όπως την ανάκτηση, την δημιουργία, την ενημέρωση και την διαγραφή των δεδομένων. Αυτή η συγκεκριμένη αρχιτεκτονική χρησιμοποιείται ακριβώς στην εφαρμογή.

Στον φάκελο routes της εφαρμογής (αρχείο web.php) , έχουν οριστεί διάφορα endpoints για την εκτέλεση λειτουργιών όπως η προβολή και η επεξεργασία προφίλ χρηστών, η αποστολή και η διαχείριση μηνυμάτων, η διαχείριση προϊόντων και οι λειτουργίες για επαναφορά κωδικού πρόσβασης. Για παράδειγμα, υπάρχουν endpoints για την προβολή της αρχικής σελίδας, την εγγραφή και την είσοδο χρηστών, την προβολή του προφίλ χρήστη, την αποστολή μηνυμάτων, την προσθήκη και την προβολή προϊόντων, τη διαχείριση αγαπημένων προϊόντων κτλ. Επιπλέον περιλαμβάνονται ειδικά endpoints για τη διαχείριση των λογαριασμών και των δικαιωμάτων πρόσβασης των χρηστών, όπως η επαναφορά κωδικού πρόσβασης και η προβολή του πίνακα ελέγχου για τον διαχειριστή. Η αρχιτεκτονική αυτή επιτρέπει την οργανωμένη και αποδοτική διαχείριση των διαφόρων λειτουργιών της εφαρμογής, διασφαλίζοντας ότι κάθε αίτημα κατευθύνεται στο κατάλληλο controller method για την επεξεργασία και την ανταπόκριση.

Παρακάτω παρατίθενται παραδείγματα routes που δείχνουν πώς καθορίζονται τα διάφορα "endpoints" στην εφαρμογή:

Route::post('/favorites/{product}', [FavoriteController::class, 'store'])->name('favorites.store');

Route::delete('/favorites/{product}', [FavoriteController::class, 'delete'])->name('favorites.delete');

Το πρώτο route χρησιμοποιείται για την αποθήκευση ενός προϊόντος στα αγαπημένα. Χρησιμοποιεί τη μέθοδο POST, η οποία στέλνει τα δεδομένα στο διακομιστή. Η διαδρομή αυτή κατευθύνεται στη μέθοδο store του controller «FavoriteController».

Το δεύτερο route χρησιμοποιείται για την διαγραφή ενός προϊόντος από τα αγαπημένα. Χρησιμοποιεί τη μέθοδο DELETE, η οποία διαγράφει τα δεδομένα από τον διακομιστή. Η διαδρομή αυτή κατευθύνεται στη μέθοδο delete του controller «FavoriteController».

3.7 Fetch Requests

Η εφαρμογή χρησιμοποιεί fetch requests για την αποστολή και λήψη δεδομένων μέσω HTTP αιτημάτων. Αυτό επιτρέπει τη δυναμική αλληλεπίδραση του frontend με το backend, χωρίς την ανάγκη για πλήρη ανανέωση της σελίδας, προσφέροντας μια πιο ομαλή και ευχάριστη εμπειρία στον χρήστη. Οι μέθοδοι αυτοί είναι γραμμένοι σε javascript γλώσσα στα αρχεία views μέσα στις επικεφαλίδες <script>.

Fetch API

Το Fetch API [34] είναι μια σύγχρονη διεπαφή που παρέχει έναν ευκολότερο και πιο ευανάγνωστο τρόπο για να εκτελούνται αιτήματα HTTP σε σχέση με το παραδοσιακό XMLHttpRequest που χρησιμοποιείται στο AJAX. Στην εφαρμογή, το Fetch API χρησιμοποιείται για:

- GET αιτήματα: Λήψη δεδομένων όπως οι λίστες προϊόντων, οι πληροφορίες προφίλ χρηστών και τα μηνύματα.
- POST αιτήματα: Αποστολή δεδομένων για την εγγραφή νέων χρηστών ή την προσθήκη νέων προϊόντων.
- PUT αιτήματα: Ενημέρωση των υπαρχόντων εγγραφών στη βάση δεδομένων, όπως η ενημέρωση προφίλ ή προϊόντων.
- DELETE αιτήματα: Αποστολή αιτημάτων για τη διαγραφή συγκεκριμένων εγγραφών από τη βάση δεδομένων.

Χρησιμοποιώντας fetch requests, η εφαρμογή επιτυγχάνει την αποστολή και την λήψη των δεδομένων με ασύγχρονο τρόπο, βελτιώνοντας την απόδοση και την εμπειρία του χρήστη. Τα αιτήματα αυτά επιτρέπουν τη διαχείριση των δεδομένων και την αλληλεπίδραση με την εφαρμογή σε πραγματικό χρόνο, χωρίς την ανάγκη για ανανέωση της σελίδας.

Παρακάτω ακολουθεί ένα παράδειγμα χρήσης της μεθόδου αυτής στην συνάρτηση που «μαρκάρει» τα μηνύματα ως αναγνωσμένα στον φάκελο messages των views (προβολών):

```
async function markMessageAsRead(messageId) {
    try {
        const response = await fetch(`/messages/${messageId}/mark-as-read`, {
            method: 'PUT',
            headers: {
                'Content-Type': 'application/json',
                'X-CSRF-TOKEN': csrfToken
            },
            body: JSON.stringify({ is_read: 1 })
        });
        if (!response.ok) {
            console.error('Failed to mark message as read');
```

```
    }
  } catch (error) {
    console.error('Error marking message as read:', error);
  }
}
```

Σε αυτό το παράδειγμα, η συνάρτηση markMessageAsRead χρησιμοποιεί το Fetch API για να στείλει ένα αίτημα PUT στον διακομιστή, ενημερώνοντας την κατάσταση του μηνύματος ως αναγνωσμένο (is_read: 1). Οι επικεφαλίδες (headers) περιλαμβάνουν το Content-Type για να δηλώσουν ότι τα δεδομένα είναι σε μορφή JSON, και το X-CSRF-TOKEN για την προστασία από επιθέσεις CSRF. Τα δεδομένα μετατρέπονται σε JSON με τη χρήση της JSON.stringify.

3.8 Πρωτόκολλο Websocket

Το πρωτόκολλο WebSocket [35] είναι ένα πρωτόκολλο επικοινωνίας που επιτρέπει τη διαρκή, διπλή κατεύθυνση επικοινωνίας ανάμεσα σε έναν πελάτη και έναν διακομιστή μέσω μιας μόνο σύνδεσης. Αυτό επιτρέπει την ανταλλαγή δεδομένων σε πραγματικό χρόνο.

Πώς λειτουργεί:

1. Χειραγωγία σύνδεσης: Η σύνδεση αρχίζει με μια αίτηση HTTP, κατόπιν πραγματοποιείται μια διαδικασία χειραγώγησης ώστε να δημιουργηθεί μια σταθερή σύνδεση WebSocket.
2. Ανταλλαγή δεδομένων: Αφού η σύνδεση έχει καθιερωθεί, τόσο ο πελάτης όσο και ο διακομιστής μπορούν να στείλουν δεδομένα στον άλλον ανά πάσα στιγμή, χωρίς την ανάγκη να περιμένουν για αιτήματα.

3.9 Υπηρεσία Pusher για websockets

Το Pusher [36] είναι μία υπηρεσία API που φιλοξενείται και κάνει πιο εύκολη την προσθήκη πραγματικού χρόνου δεδομένων και της λειτουργικότητας σε εφαρμογές. Η κύρια διαφορά με τα WebSockets είναι ότι το Pusher είναι μια φιλοξενούμενη υπηρεσία API.

Πώς λειτουργεί το Pusher:

1. Βιβλιοθήκες πελάτη: Το Pusher παρέχει βιβλιοθήκες πελάτη για διάφορες πλατφόρμες και γλώσσες όπως JavaScript, iOS, Android κ.λπ. Αυτές οι βιβλιοθήκες χειρίζονται τη σύνδεση στους διακομιστές του Pusher και παρέχουν μεθόδους για να εγγραφούν σε κανάλια και να συνδεθούν σε συμβάντα.
2. Κανάλια: Το Pusher οργανώνει τα δεδομένα σε κανάλια. Οι clients μπορούν να εγγραφούν σε συγκεκριμένα κανάλια για να λαμβάνουν ενημερώσεις. Τα κανάλια χρησιμοποιούνται τυπικά για να αντιπροσωπεύσουν διαφορετικά θέματα ή δωμάτια όπου κοινοποιούνται δεδομένα.
3. Συμβάντα: Όταν ένας client δημοσιεύει ένα συμβάν σε ένα κανάλι, το Pusher μεταδίδει αυτό το συμβάν σε όλους τους πελάτες που εγγράφονται σε εκείνο το κανάλι. Τα συμβάντα μπορούν να περιέχουν οποιαδήποτε δεδομένα θέλει ο προγραμματιστής της εφαρμογής και να στείλει σε πραγματικό χρόνο, όπως μηνύματα συνομιλίας, ειδοποιήσεις, ενημερώσεις κ.λπ.

4. WebSockets: Το Pusher χρησιμοποιεί το πρωτόκολλο WebSocket ως το κύριο πρωτόκολλο επικοινωνίας για τη μεταφορά δεδομένων σε πραγματικό χρόνο. Τα WebSockets επιτρέπουν την πλήρη διπλή επικοινωνία μεταξύ πελάτη και διακομιστή μέσω μιας μόνο, μακροχρόνιας σύνδεσης. Αυτό επιτρέπει την άμεση αποστολή και λήψη ενημερώσεων σε πραγματικό χρόνο.

Το Pusher απλοποιεί τη διαδικασία της λειτουργικότητας πραγματικού χρόνου επικοινωνίας στις εφαρμογές παρέχοντας ένα εύκολο στη χρήση API .Συγκεκριμένα στην εφαρμογή χρησιμοποιείται τόσο για τη μετάδοση μηνυμάτων όσο και για τις ειδοποιήσεις που αφορούν τα μηνύματα. Ένα από τα σημαντικά στοιχεία του Pusher είναι η δημιουργία συγκεκριμένων συμβάντων (events) που επιτρέπουν στις εφαρμογές να ενημερώνονται αυτόματα όταν συμβαίνουν κάποιες συγκεκριμένες ενέργειες. Για παράδειγμα, στην εφαρμογή και συγκεκριμένα στη λειτουργία των μηνυμάτων, ένα συμβάν που έχει δημιουργηθεί είναι το "New Message", το οποίο εκτελείται κάθε φορά που δημιουργείται ένα νέο μήνυμα. Στη συνέχεια, μέσω του Pusher, τα συμβάντα αυτά μεταδίδονται στην εφαρμογή μέσω ενός καναλιού (messages), όπου οι χρήστες μπορούν να εγγραφούν για να λαμβάνουν ειδοποιήσεις όταν συμβούν αυτά τα συμβάντα.

Παρακάτω παρατίθεται ένα παράδειγμα κώδικα χρήσης του Pusher (αρχείο pusher.js):

```
document.addEventListener('DOMContentLoaded', function() {
    var pusher = new Pusher('4475f46df53f83682842', {
        cluster: 'eu',
        encrypted: true
    });
    var channel = pusher.subscribe('messages');
    channel.bind('NewMessage', function(data) {
        console.log('Received new message:', data);
        //έλεγχος εάν ο εξουσιοδοτημένος χρήστης είναι ο παραλήπτης
        if (data.message.receiver_id === authUserId) {
            var senderName = data.sender_name;
            var messageContent = data.message.message_content;
            //έλεγχος εάν ο χρήστης δεν βρίσκεται στη σελίδα μηνυμάτων
            if (window.location.pathname !== '/messages') {
                //αναπαραγωγή ήχου ειδοποίησης
                var alertSound = document.getElementById('alertSound');
                alertSound.play();
```

```
            //εμφάνιση ειδοποίησης
            alert('New message from' + senderName + ': ' + messageContent);
        }
    }
});
});
```

Στο παραπάνω κώδικα, βλέπουμε τη χρήση του Pusher στο σύστημα μηνυμάτων. Ένα συγκεκριμένο γεγονός (event) που έχει δημιουργηθεί είναι το "New Message" (αρχείο NewMessage.php στον υποφάκελο Events του φακέλου app). Αυτό το γεγονός ενεργοποιείται κάθε φορά που δημιουργείται ένα νέο μήνυμα στο σύστημα. Κατά την επεξεργασία του συμβάντος, ο έλεγχος γίνεται για τον παραλήπτη του μηνύματος, ελέγχοντας το αναγνωριστικό του πιστοποιημένου χρήστη. Αν ο χρήστης είναι ο παραλήπτης του μηνύματος και δεν βρίσκεται στη σελίδα /messages, τότε πραγματοποιείται ειδοποίηση με ήχο και μήνυμα ειδοποίησης στον χρήστη για το νέο μήνυμα που έχει λάβει.

3.10 Αρχιτεκτονική Εφαρμογής

Συνολικά, η εφαρμογή αποτελείται από:

- Μία βάση δεδομένων MySQL. Η βάση δεδομένων χρησιμοποιεί τα δεδομένα σε μορφή JSON για την αποθήκευση τους. Τα δεδομένα ανταλλάσσονται σε μορφή JSON μεταξύ του backend και του frontend.
- Αρχιτεκτονική REST APIs με το Laravel: Το Laravel χρησιμοποιείται ως το backend framework για την ανάπτυξη ενός RESTful API, το οποίο εξυπηρετεί αιτήματα HTTP από το frontend. Κάθε αίτημα χειρίζεται από έναν ελεγκτή (controller), ο οποίος επεξεργάζεται το αίτημα και επιστρέφει την κατάλληλη απάντηση.
- Το Pusher χρησιμοποιείται για την υλοποίηση της πραγματικού χρόνου ενημερώσεων (real-time updates) στην εφαρμογή.

Το API υλοποιεί HTTP requests για να επιτρέπει την αλληλεπίδραση των χρηστών με την εφαρμογή. Το front-end ανανεώνεται δυναμικά και παρέχει μια φιλική προς τον χρήστη εμπειρία.

Backend

Το API του Laravel εξυπηρετεί τα αιτήματα του frontend μέσω HTTP requests. Κάθε αίτημα διαχειρίζεται από έναν συγκεκριμένο ελεγκτή (controller), ο οποίος αναλαμβάνει την επεξεργασία του αιτήματος και την επιστροφή της κατάλληλης απάντησης.

Frontend

Για το frontend, χρησιμοποιείται το Vite, ένα μοντέρνο εργαλείο ανάπτυξης που προσφέρει ταχύτητα και αποδοτικότητα στην ανανέωση και τη μεταγλώττιση των πόρων. Το CSS Bootstrap χρησιμοποιείται για τη σχεδίαση και τη διάταξη του περιεχομένου, προσφέροντας ένα καλαίσθητο και ευέλικτο περιβάλλον εργασίας.

Δομή της Εφαρμογής

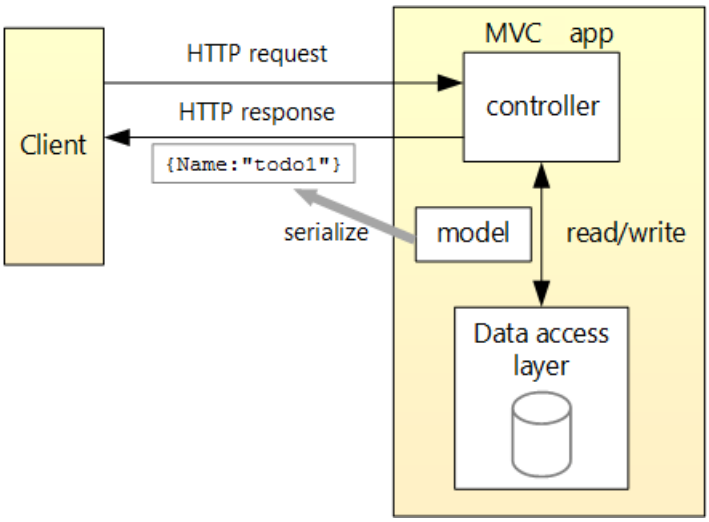
Η δομή της εφαρμογής ακολουθεί την τυπική δομή ενός Laravel project και είναι οργανωμένη σε διάφορους φακέλους και αρχεία.

Κύριοι Φάκελοι:

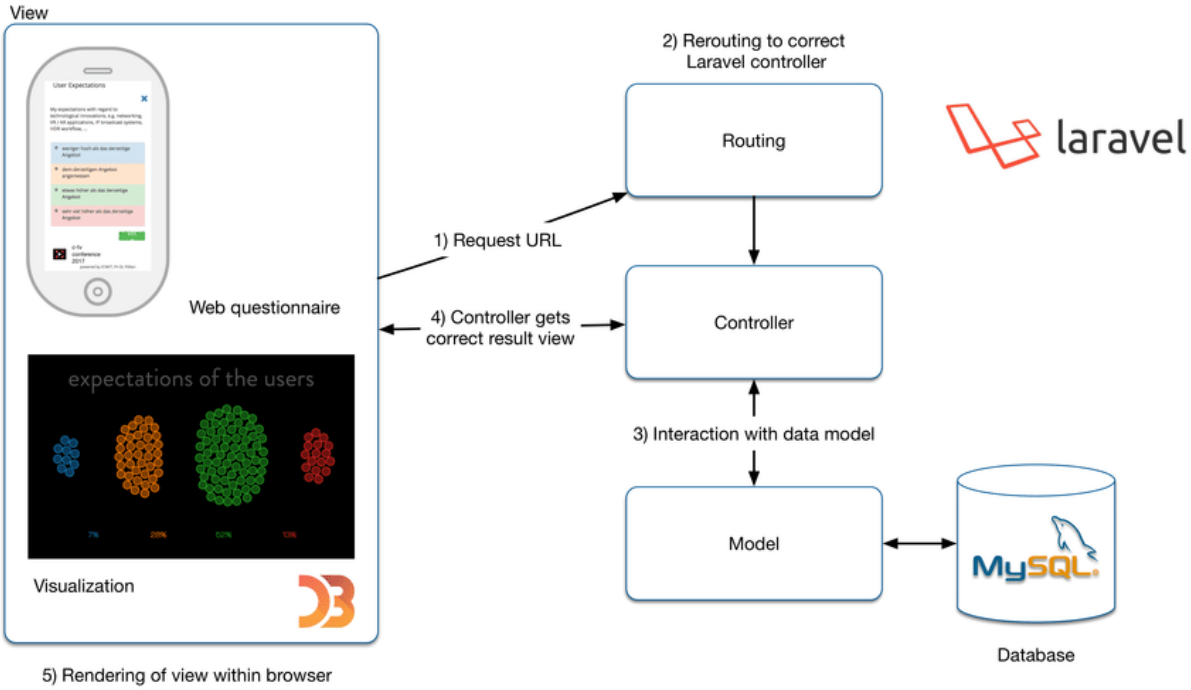
- **App**: Περιέχει τον κώδικα του backend, οργανωμένο σε υποφακέλους και υποσυστήματα.
  Models: Περιέχει τα αρχεία των μοντέλων που αντιστοιχούν στους πίνακες της βάσης δεδομένων. Για παράδειγμα, τα αρχεία User.php, Product.php, και Message.php ορίζουν τα μοντέλα για τους πίνακες users, products, και messages αντίστοιχα.
  Controllers: Περιέχει τους ελεγκτές που διαχειρίζονται τα αιτήματα HTTP. Για παράδειγμα, ο UserController.php χειρίζεται αιτήματα σχετικά με τους χρήστες, ενώ ο ProductController.php διαχειρίζεται αιτήματα που αφορούν τα προϊόντα.
  Events: Αυτός ο φάκελος περιλαμβάνει τα αρχεία που σχετίζονται με τα γεγονότα της εφαρμογής. Για παράδειγμα, περιέχει το αρχείο "NewMessageEvent.php", το οποίο ορίζει τη λειτουργικότητα που πρέπει να εκτελείται όταν δημιουργείται ένα νέο μήνυμα στην εφαρμογή.
- **Database**: Περιέχει τα migrations και τα seeders για τη βάση δεδομένων, τα οποία χρησιμοποιούνται για τη δημιουργία και την αρχικοποίηση των πινάκων της βάσης δεδομένων.
- **Public**: Περιέχει τα δημόσια αρχεία που είναι προσβάσιμα από το διαδίκτυο, όπως εικόνες, ήχοι και JavaScript αρχεία (pusher.js κτλ).
- **Resources**: Περιέχει τα raw assets για το frontend και τα blade templates (views).
  Views: Περιέχει τα blade templates που χρησιμοποιούνται για την εμφάνιση των δεδομένων στους χρήστες. Για παράδειγμα, τα αρχεία profile.blade.php και product.blade.php είναι templates για την προβολή του προφίλ των χρηστών και των προϊόντων αντίστοιχα.
- **Routes**: Περιέχει τους ορισμούς των routes της εφαρμογής. Το αρχείο web.php ορίζει τις διαδρομές που είναι προσβάσιμες από τον χρήστη μέσω του browser.

Ανάπτυξη και Φιλοξενία

Η ανάπτυξη και ο προγραμματισμός της εφαρμογής πραγματοποιήθηκε μέσω του Visual Studio Code [37] , ενός ισχυρού και ευέλικτου IDE. Για τη φιλοξενία του backend και της βάσης δεδομένων, χρησιμοποιήθηκε το XAMPP, το οποίο παρέχει έναν εύκολο στη χρήση Apache server και μια MySQL βάση δεδομένων.



Εικόνα 11 - Αναπαράσταση λειτουργίας εφαρμογής [38]



Εικόνα 12 - Μετά την αποστολή του αιτήματος URL (1), ο Laravel δρομολογεί στον κατάλληλο ελεγκτή (2). Το μοντέλο δεδομένων θα ενσωματωθεί από την βάση δεδομένων MySQL (3) και η προβολή αποτελέσματος καλείται (4). Τέλος, η προβολή αποδίδεται στον ιστότοπο του χρήστη και στην οπτικοποίηση (5). [39]

4. Κεφάλαιο 4ο: Βάση Δεδομένων

Στο παρόν κεφάλαιο, παρουσιάζονται τα είδη των βάσεων δεδομένων καθώς και η εστίαση στη βάση δεδομένων MySQL, συμπεριλαμβανομένων των λόγων που οδήγησαν στην επιλογή της, του σκοπού της, των πλεονεκτημάτων και των μειονεκτημάτων της.

4.1 Είδη βάσεων δεδομένων

Οι βάσεις δεδομένων κατηγοριοποιούνται σε τέσσερα βασικά είδη: οι σχεσιακές, οι μη σχεσιακές, οι ιεραρχικές και οι αντικειμενοστραφείς. Κάθε είδος έχει τα πλεονεκτήματά του και την κατάλληλη χρήση του σε συγκεκριμένες περιπτώσεις. [40]

1. **Σχεσιακές Βάσεις Δεδομένων (RDBMS)**
   Οι σχεσιακές βάσεις δεδομένων είναι οι πιο διαδεδομένες μορφές βάσεων δεδομένων. Χρησιμοποιούν ένα μοντέλο δεδομένων που βασίζεται σε πίνακες (tables) και σχέσεις (relations) ανάμεσα σε αυτούς. Κάθε πίνακας αποτελείται από γραμμές (rows) και στήλες (columns) και οι πίνακες μπορούν να συνδέονται μεταξύ τους μέσω ξένων κλειδιών (foreign keys). Κλασικά παραδείγματα RDBMS περιλαμβάνουν τις MySQL, PostgreSQL, Oracle και Microsoft SQL Server. Μερικά πλεονεκτήματα είναι η αυστηρή ακεραιότητα δεδομένων και η χρήση της γλώσσας SQL για τη διαχείριση δεδομένων.
2. **Μη Σχεσιακές Βάσεις Δεδομένων (NoSQL)**
   Οι NoSQL βάσεις δεδομένων είναι σχεδιασμένες για να αντιμετωπίζουν τις ανάγκες αποθήκευσης μεγάλων όγκων δεδομένων και υψηλής απόδοσης που δεν μπορούν να διαχειριστούν οι RDBMS. Υπάρχουν διάφορες κατηγορίες NoSQL βάσεων δεδομένων, όπως:
   - Βάσεις δεδομένων κειμένου (document databases): Αποθηκεύουν δεδομένα σε μορφή εγγράφων, συνήθως σε JSON μορφή. Χαρακτηριστικό παράδειγμα είναι η βάση MongoDB.
   - Βάσεις δεδομένων κλειδί-τιμή (key-value stores): Χρησιμοποιούν ένα απλό μοντέλο ζεύγους κλειδιού και τιμής για την αποθήκευση δεδομένων. (Redis)
   - Βάσεις δεδομένων στηλών (column-family stores): Αποθηκεύουν δεδομένα σε μορφή στηλών, που επιτρέπουν την αποθήκευση μεγάλων ποσοτήτων δεδομένων.
   - Γραφηματικές βάσεις δεδομένων (graph databases): Εστιάζουν στη διαχείριση δεδομένων που συνδέονται μεταξύ τους μέσω σχέσεων.
3. **Ιεραρχικές (Hierarchical Databases):**
   Σε αυτές τις βάσεις δεδομένων, τα δεδομένα οργανώνονται σε μια ιεραρχία όπου κάθε δεδομένο έχει ένα μόνο προηγούμενο. Οι XML βάσεις δεδομένων είναι ένα παράδειγμα ιεραρχικών βάσεων δεδομένων.
4. **Αντικειμενοστραφείς (Object-oriented Databases):**
   Οι βάσεις δεδομένων αντικειμένων αποθηκεύουν τα δεδομένα ως αντικείμενα, όπως αυτά ορίζονται στα αντικειμενοστραφή προγραμματιστικά πρότυπα.

4.2 Η βάση δεδομένων MySQL

Η επιλογή της βάσης δεδομένων MySQL [41] για την εφαρμογή αυτή έγινε μετά από λεπτομερή εξέταση των απαιτήσεων και των χαρακτηριστικών του συστήματος. Η MySQL επιλέχθηκε λόγω της σταθερότητας, της ευελιξίας και της σημαντικής υποστήριξης που προσφέρει. Χρησιμοποιείται σε πλήθος εφαρμογών και παρέχει ικανοποιητική απόδοση. Η χρήση της στην παρούσα εργασία στοχεύει στην σωστή διαχείριση, αποθήκευση και ανάκτηση των δεδομένων.

Η επιλογή της MySQL έναντι άλλων βάσεων όπως η MongoDB, έγινε κυρίως λόγω της σχεσιακής φύσης των δεδομένων που χρειάστηκαν για να αποθηκευτούν τα δεδομένα αυτά. Επίσης η MySQL παρέχει αυστηρότερους μηχανισμούς ακεραιότητας δεδομένων μέσω των σχέσεων και των περιορισμών, κάτι το οποίο ήταν σημαντικό για την εφαρμογή. Παρακάτω επισημαίνονται τα πλεονεκτήματα και τα μειονεκτήματα της.

Πλεονεκτήματα:

- **Απόδοση**: Η MySQL είναι γνωστή για την υψηλή απόδοσή της στις περισσότερες περιπτώσεις χρήσης. Μπορεί να διαχειριστεί μεγάλους όγκους δεδομένων με αποτελεσματικό τρόπο. Η δυνατότητα χρήσης ευρετηρίων (indexes) και βελτιστοποίησης των ερωτημάτων (query optimization) αποτελούν σημαντικοί παράγοντες για την απόδοση.
- **Ευκολία χρήσης**: Η εγκατάσταση και η διαμόρφωση της MySQL είναι απλή και γρήγορη. Το εργαλείο διαχείρισης, phpMyAdmin, προσφέρει γραφικό περιβάλλον που διευκολύνει την αλληλεπίδραση με τη βάση δεδομένων.
- **Ασφάλεια**: Η MySQL προσφέρει ισχυρά χαρακτηριστικά ασφαλείας που εξασφαλίζουν την προστασία των δεδομένων. Υποστηρίζει την κρυπτογράφηση δεδομένων, την πιστοποίηση χρηστών και τα δικαιώματα πρόσβασης.

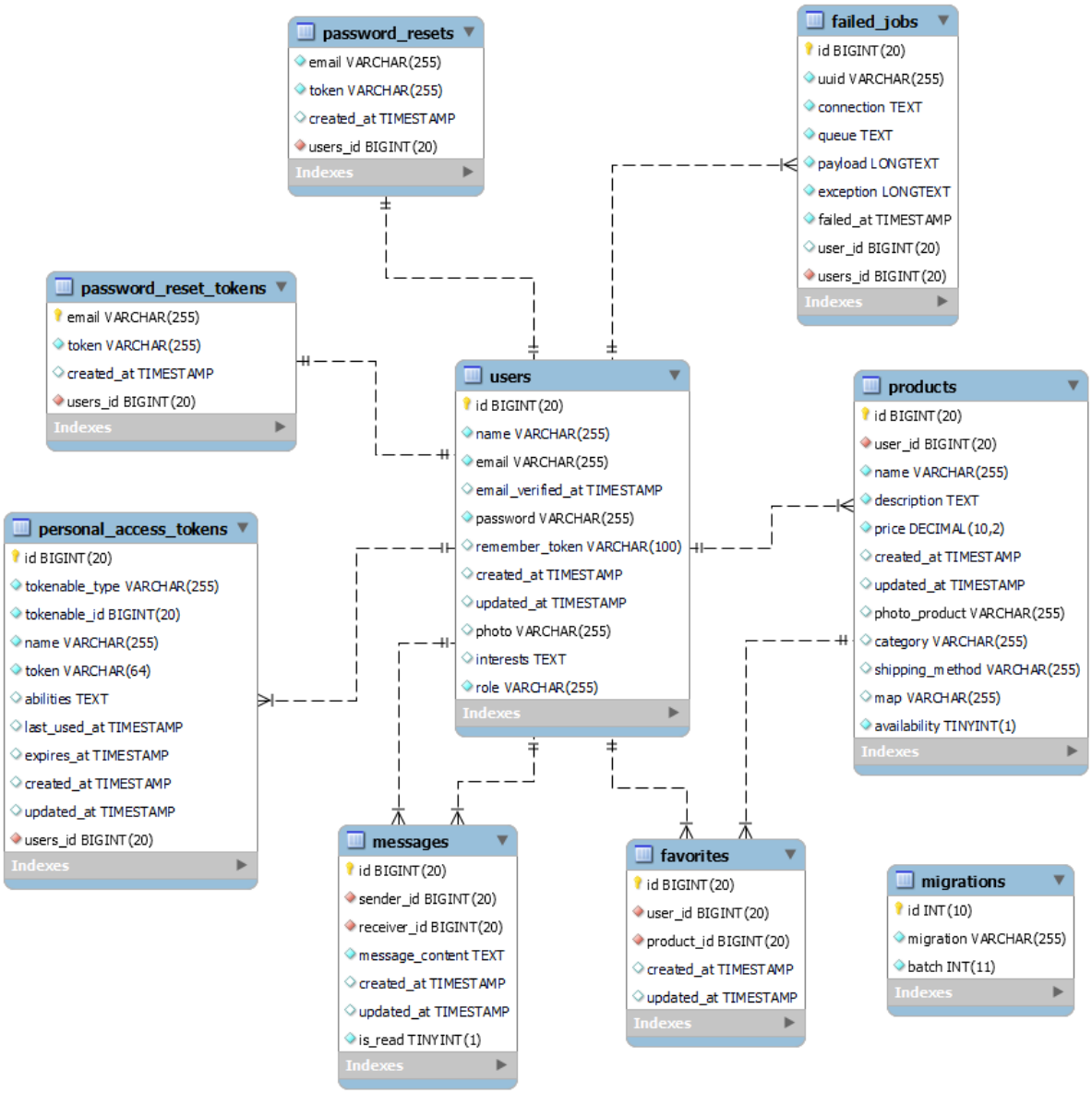
Μειονεκτήματα:

- **Κλιμάκωση**: Παρά την καλή απόδοση της MySQL, η κλιμάκωση της σε πολύ μεγάλες εφαρμογές μπορεί να απαιτήσει περισσότερους πόρους και εξειδικευμένη διαμόρφωση. Σε περιπτώσεις που απαιτούνται τεράστιοι όγκοι δεδομένων και πολύ υψηλή διαθεσιμότητα, η MySQL μπορεί να χρειαστεί να ενσωματώσει επιπλέον λύσεις όπως clustering ή sharding, που προσθέτουν πολυπλοκότητα στη διαχείριση.
- **Λιγότερη ευελιξία σε δομή δεδομένων**: Σε σύγκριση με τις NoSQL βάσεις δεδομένων, όπως η MongoDB, η MySQL απαιτεί προκαθορισμένη δομή δεδομένων. Αυτό σημαίνει ότι οι πίνακες και οι σχέσεις τους πρέπει να οριστούν εκ των προτέρων, κάτι που μπορεί να περιορίσει την ευελιξία όταν οι απαιτήσεις του έργου αλλάζουν δυναμικά. Σε περιπτώσεις όπου τα δεδομένα είναι λιγότερο δομημένα ή η δομή τους αλλάζει συχνά, μια NoSQL βάση δεδομένων μπορεί να είναι πιο κατάλληλη επιλογή.

Παρά τα μειονεκτήματα αυτά, η επιλογή της MySQL θεωρήθηκε η καταλληλότερη για τις ανάγκες του παρόντος έργου, λαμβάνοντας υπόψη την ισορροπία μεταξύ λειτουργικότητας, απόδοσης και ευκολίας χρήσης.

4.3 Μορφή βάσης δεδομένων (Schema)

Παρακάτω απεικονίζεται η μορφή βάσης δεδομένων που χρησιμοποιεί η εφαρμογή σε EER διάγραμμα από την εφαρμογή MySQL Workbench.



Εικόνα 13 - Βάση δεδομένων εφαρμογής

Η βάση ονομάζεται “marketplace” και αποτελείται από 9 πίνακες συνολικά εκ των οποίων οι 5 από αυτούς δημιουργήθηκαν αυτόματα κατά την δημιουργία του καινούριου project (migrations, failed_jobs ,passwords_reset_tokens , password_resets και personal_access_tokens). Οι υπόλοιποι 4 πίνακες (users, products,favorites, messages) δημιουργήθηκαν για τον σκοπό της εφαρμογής αυτής.

4.4 Πρωτεύοντα και ξένα κλειδιά (primary και foreign keys)

- **Πίνακας users**
  Primary Key: id
  Foreign Key: Δεν υπάρχουν foreign keys στον πίνακα αυτόν. (Δεν έχει ξένα κλειδιά καθώς αποτελεί τον κεντρικό πίνακα που συνδέει πολλούς άλλους πίνακες μέσω των δικών του πρωτευόντων κλειδιών.)
- **Πίνακας products**
  Primary Key: id
  Foreign Key: user_id (αναφέρεται στο id του πίνακα users)
- **Πίνακας favorites**
  Primary Key: id
  Foreign Keys: user_id (αναφέρεται στο id του πίνακα users) και
  product_id (αναφέρεται στο id του πίνακα products)
- **Πίνακας messages**
  Primary Key: id
  Foreign Keys: sender_id (αναφέρεται στο id του πίνακα users) και
  receiver_id (αναφέρεται στο id του πίνακα users)
- **Πίνακας failed_jobs**
  Primary Key: id
  Foreign Key: user_id (αναφέρεται στο id του πίνακα users)
- **Πίνακας personal_access_tokens**
  Primary Key: id
  Foreign Keys: users_id (αναφέρεται στο id του πίνακα users)
- **Πίνακας password_reset_tokens**
  Primary Key: email, token
  Foreign Keys: users_id (αναφέρεται στο id του πίνακα users)
- **Πίνακας password_resets**
  Primary Key: email
  Foreign Keys: users_id (αναφέρεται στο id του πίνακα users)
- **Πίνακας migrations**
  Primary Key: id
  Foreign Keys: Δεν υπάρχουν foreign keys στον πίνακα αυτόν.

4.5 Περιγραφή των μοντέλων και των πεδίων τους

Οι πίνακες περιλαμβάνουν τους χρήστες (users), τα προϊόντα (products),τα μηνύματα (messages), τα αγαπημένα προϊόντα για κάθε χρήστη (favorites), τις αποτυχημένες εργασίες (failed_jobs), τα προσωπικά διακριτικά πρόσβασης (personal_access_tokens), τις μάρκες επαναφοράς κωδικού (password_reset_tokens), τις επαναφορές κωδικού (password_resets), και τις μεταναστεύσεις (migrations). Κάθε πίνακας περιέχει διάφορα πεδία με συγκεκριμένες ιδιότητες, τα οποία περιγράφονται παρακάτω.

Πίνακας users

- id (BIGINT): Ο μοναδικός αναγνωριστικός αριθμός του χρήστη.
- name (VARCHAR): Το όνομα του χρήστη.
- email (VARCHAR): Το email του χρήστη.
- email_verified_at (TIMESTAMP): Η ημερομηνία και ώρα επιβεβαίωσης του email του χρήστη.
- password (VARCHAR): Ο κωδικός πρόσβασης του χρήστη.
- remember_token (VARCHAR): Το διακριτικό "θυμήσου με" για αυτόματη είσοδο.
- created_at (TIMESTAMP): Η ημερομηνία και ώρα δημιουργίας της εγγραφής του χρήστη.
- updated_at (TIMESTAMP): Η ημερομηνία και ώρα τελευταίας ενημέρωσης του χρήστη.
- photo (VARCHAR): Η φωτογραφία του χρήστη.
- interests (TEXT): Τα ενδιαφέροντα του χρήστη.
- role (VARCHAR): Ο ρόλος του κάθε χρήστη. Οι δύο ρόλοι είναι είτε απλός χρήστης (user) ή διαχειριστής (admin).

Πίνακας products

- id (BIGINT): Ο μοναδικός αναγνωριστικός αριθμός του προϊόντος.
- user_id (BIGINT): Ο αναγνωριστικός αριθμός του χρήστη που δημιούργησε το προϊόν.
- name (VARCHAR): Το όνομα του προϊόντος.
- description (TEXT): Η περιγραφή του προϊόντος.
- price (DECIMAL): Η τιμή του προϊόντος.
- created_at (TIMESTAMP): Η ημερομηνία και ώρα δημιουργίας της εγγραφής του προϊόντος.
- updated_at (TIMESTAMP): Η ημερομηνία και ώρα τελευταίας ενημέρωσης του προϊόντος.
- photo_product (VARCHAR): Η φωτογραφία του προϊόντος.
- category (VARCHAR): Η κατηγορία του προϊόντος.
- shipping_method (VARCHAR): Η μέθοδος αποστολής του προϊόντος.
- map (VARCHAR): Η τοποθεσία του προϊόντος.
- availability (TINYINT): Η διαθεσιμότητα του προϊόντος (0 για μη διαθέσιμο ή 1 για διαθέσιμο).

Πίνακας messages

- id (BIGINT): Ο μοναδικός αναγνωριστικός αριθμός του μηνύματος.
- sender_id (BIGINT): Ο αναγνωριστικός αριθμός του αποστολέα.
- receiver_id (BIGINT): Ο αναγνωριστικός αριθμός του παραλήπτη.
- message_content (TEXT): Το περιεχόμενο του μηνύματος.
- created_at (TIMESTAMP): Η ημερομηνία και ώρα δημιουργίας της εγγραφής του μηνύματος.
- updated_at (TIMESTAMP): Η ημερομηνία και ώρα τελευταίας ενημέρωσης του μηνύματος.
- is_read (TINYINT): Η ένδειξη αν το μήνυμα έχει διαβαστεί (0 για ότι δεν έχει διαβαστεί ή 1 για το ότι έχει διαβαστεί από τον παραλήπτη).

Πίνακας favorites

- id (BIGINT): Ο μοναδικός αναγνωριστικός αριθμός του αγαπημένου προϊόντος.
- user_id (BIGINT): Ο αναγνωριστικός αριθμός του χρήστη που πρόσθεσε το αγαπημένο.
- product_id (BIGINT): Ο αναγνωριστικός αριθμός του προϊόντος που προστέθηκε ως αγαπημένο.
- created_at (TIMESTAMP): Η ημερομηνία και ώρα δημιουργίας της εγγραφής του αγαπημένου προϊόντος.
- updated_at (TIMESTAMP): Η ημερομηνία και ώρα τελευταίας ενημέρωσης του αγαπημένου προϊόντος.

Πίνακας failed_jobs

- id (BIGINT): Ο μοναδικός αναγνωριστικός αριθμός της αποτυχημένης εργασίας.
- uuid (VARCHAR): Το μοναδικό αναγνωριστικό της εργασίας.
- connection (TEXT): Η σύνδεση που χρησιμοποιήθηκε για την εργασία.
- queue (TEXT): Η ουρά στην οποία προστέθηκε η εργασία.
- payload (LONGTEXT): Τα δεδομένα της εργασίας.
- exception (LONGTEXT): Η εξαίρεση που προκλήθηκε.
- failed_at (TIMESTAMP): Η ημερομηνία και ώρα που απέτυχε η εργασία.
- user_id (BIGINT): Ο αναγνωριστικός αριθμός του χρήστη που σχετίζεται με την εργασία.

Πίνακας personal_access_tokens

- id (BIGINT): Ο μοναδικός αναγνωριστικός αριθμός του διακριτικού πρόσβασης.
- tokenable_type (VARCHAR): Ο τύπος του διακριτικού.
- tokenable_id (BIGINT): Ο αναγνωριστικός αριθμός του διακριτικού.
- name (VARCHAR): Το όνομα του διακριτικού.
- token (VARCHAR): Το διακριτικό.
- abilities (TEXT): Οι δυνατότητες του διακριτικού.
- last_used_at (TIMESTAMP): Η τελευταία χρήση του διακριτικού.
- created_at (TIMESTAMP): Η ημερομηνία και ώρα δημιουργίας της εγγραφής.

- updated_at (TIMESTAMP): Η ημερομηνία και ώρα τελευταίας ενημέρωσης της εγγραφής.
- users_id (BIGINT): Ο αναγνωριστικός αριθμός του χρήστη που σχετίζεται με το διακριτικό.

Πίνακας password_reset_tokens

- email (VARCHAR): Το email του χρήστη.
- token (VARCHAR): Το διακριτικό επαναφοράς κωδικού.
- created_at (TIMESTAMP): Η ημερομηνία και ώρα δημιουργίας της εγγραφής.
- users_id (BIGINT): Ο αναγνωριστικός αριθμός του χρήστη που σχετίζεται με το διακριτικό.

Πίνακας password_resets

- email (VARCHAR): Το email του χρήστη.
- token (VARCHAR): Το διακριτικό επαναφοράς κωδικού.
- created_at (TIMESTAMP): Η ημερομηνία και ώρα δημιουργίας της εγγραφής.
- users_id (BIGINT): Ο αναγνωριστικός αριθμός του χρήστη που σχετίζεται με την επαναφορά κωδικού.

Πίνακας migrations

- id (INT): Ο μοναδικός αναγνωριστικός αριθμός της μετανάστευσης.
- migration (VARCHAR): Το όνομα της μετανάστευσης.
- batch (INT): Ο αριθμός της παρτίδας της μετανάστευσης.

4.6 Σχέσεις μεταξύ των πινάκων

Όπως φαίνεται από το διάγραμμα στην ενότητα 4.3, υπάρχουν σχέσεις μεταξύ των πινάκων που είναι 1:1, 1:N κτλ. Αναλυτικά, οι σχέσεις μεταξύ των πινάκων είναι οι εξής:

4.6.1 Πίνακας users

Ο πίνακας users είναι ο κεντρικός πίνακας της βάσης δεδομένων και περιέχει πληροφορίες για τους χρήστες. Έχει τις ακόλουθες σχέσεις:

- 1:N με τον πίνακα products: Ένας χρήστης μπορεί να έχει πολλά προϊόντα. Η σχέση αυτή υλοποιείται μέσω του πεδίου user_id στον πίνακα products.
- 1:N με τον πίνακα messages: Ένας χρήστης μπορεί να στείλει και να λάβει πολλά μηνύματα. Η σχέση αυτή υλοποιείται μέσω των πεδίων sender_id και receiver_id στον πίνακα messages.
- 1:N με τον πίνακα favorites: Ένας χρήστης μπορεί να έχει πολλά αγαπημένα προϊόντα. Η σχέση αυτή υλοποιείται μέσω του πεδίου user_id στον πίνακα favorites.
- 1:N με τον πίνακα failed_jobs: Ένας χρήστης μπορεί να έχει πολλές αποτυχημένες εργασίες. Η σχέση αυτή υλοποιείται μέσω του πεδίου user_id στον πίνακα failed_jobs.
- 1:N με τον πίνακα personal_access_tokens: Ένας χρήστης μπορεί να έχει πολλά προσωπικά διακριτικά πρόσβασης. Η σχέση αυτή υλοποιείται μέσω του πεδίου users_id στον πίνακα personal_access_tokens.

- 1:1 με τον πίνακα password_reset_tokens: Ένας χρήστης μπορεί να έχει ένα και μόνο ένα token για επαναφορά κωδικού πρόσβασης.
- 1:1 με τον πίνακα password_resets: Ένας χρήστης μπορεί να έχει ένα και μόνο ένα αίτημα επαναφοράς κωδικού πρόσβασης ταυτόχρονα.

4.6.2 Πίνακας products

Ο πίνακας products περιέχει πληροφορίες για τα προϊόντα που προσθέτουν οι χρήστες. Έχει τις ακόλουθες σχέσεις:

- N:1 με τον πίνακα users: Κάθε προϊόν ανήκει σε έναν χρήστη. Η σχέση αυτή υλοποιείται μέσω του πεδίου user_id στον πίνακα products.
- 1:N με τον πίνακα favorites: Ένα προϊόν μπορεί να προστεθεί ως αγαπημένο από πολλούς χρήστες. Η σχέση αυτή υλοποιείται μέσω του πεδίου product_id στον πίνακα favorites.

4.6.3 Πίνακας favorites

Ο πίνακας favorites περιέχει πληροφορίες για τα αγαπημένα προϊόντα των χρηστών. Έχει τις ακόλουθες σχέσεις:

- N:1 με τον πίνακα users: Κάθε αγαπημένο ανήκει σε έναν χρήστη. Η σχέση αυτή υλοποιείται μέσω του πεδίου user_id στον πίνακα favorites.
- N:1 με τον πίνακα products: Κάθε αγαπημένο αντιστοιχεί σε ένα προϊόν. Η σχέση αυτή υλοποιείται μέσω του πεδίου product_id στον πίνακα favorites.

4.6.4 Πίνακας messages

Ο πίνακας messages περιέχει πληροφορίες για τα μηνύματα που ανταλλάσσουν οι χρήστες. Έχει τις ακόλουθες σχέσεις:

- N:1 με τον πίνακα users: Κάθε μήνυμα έχει έναν αποστολέα και έναν παραλήπτη. Η σχέση αυτή υλοποιείται μέσω των πεδίων sender_id και receiver_id στον πίνακα messages.

4.6.5 Πίνακας failed_jobs

Ο πίνακας failed_jobs περιέχει πληροφορίες για αποτυχημένες εργασίες. Οι σχέσεις του πίνακα είναι οι εξής:

- N:1 με τον πίνακα users: Κάθε αποτυχημένη εργασία σχετίζεται με έναν χρήστη. Η σχέση αυτή υλοποιείται μέσω του πεδίου user_id στον πίνακα failed_jobs, που αντιστοιχεί στο πεδίο id του πίνακα users.

4.6.6 Πίνακας personal_access_tokens

Ο πίνακας personal_access_tokens περιέχει πληροφορίες για προσωπικά διακριτικά πρόσβασης. Οι σχέσεις του πίνακα είναι οι εξής:

- N:1 με τον πίνακα users: Κάθε διακριτικό πρόσβασης σχετίζεται με έναν χρήστη. Η σχέση αυτή υλοποιείται μέσω του πεδίου users_id στον πίνακα personal_access_tokens, που αντιστοιχεί στο πεδίο id του πίνακα users.

4.6.7 Πίνακας password_reset_tokens

Ο πίνακας password_reset_tokens περιέχει πληροφορίες για διακριτικά επαναφοράς κωδικού. Οι σχέσεις του πίνακα είναι οι εξής:

- 1:1 με τον πίνακα users: Κάθε διακριτικό επαναφοράς κωδικού σχετίζεται με έναν χρήστη. Η σχέση αυτή υλοποιείται μέσω του πεδίου users_id στον πίνακα password_reset_tokens, που αντιστοιχεί στο πεδίο id του πίνακα users.

4.6.8 Πίνακας password_resets

Ο πίνακας password_resets περιέχει πληροφορίες για επαναφορές κωδικού. Οι σχέσεις του πίνακα είναι οι εξής:

- 1:1 με τον πίνακα users: Κάθε επαναφορά κωδικού σχετίζεται με έναν χρήστη. Η σχέση αυτή υλοποιείται μέσω του πεδίου users_id στον πίνακα password_resets, που αντιστοιχεί στο πεδίο id του πίνακα users.

4.6.9 Πίνακας migrations

Ο πίνακας migrations περιέχει πληροφορίες για τις μεταναστεύσεις της βάσης δεδομένων. Δεν έχει σχέσεις με άλλους πίνακες.

5. Κεφάλαιο 5ο: Λειτουργίες Εφαρμογής και περιγραφή UI

Στο κεφάλαιο αυτό θα περιγράφουν αναλυτικά όλες οι λειτουργίες της εφαρμογής, δηλαδή οι βασικές υπηρεσίες, και στην συνέχεια θα αναλυθεί το UI (user interface) της εφαρμογής, δηλαδή η αλληλεπίδραση του χρήστη με την εφαρμογή. Ακολουθεί ο προτεινόμενος τρόπος χειρισμού της εφαρμογής, δηλαδή η δυνατότητα ενός χρήστη να αναζητά ή να ανεβάζει προϊόντα και να επικοινωνεί με άλλους χρήστες του κοινωνικού δικτύου για αυτά.

5.1 Βασικές Υπηρεσίες – Περιγραφή Λειτουργιών

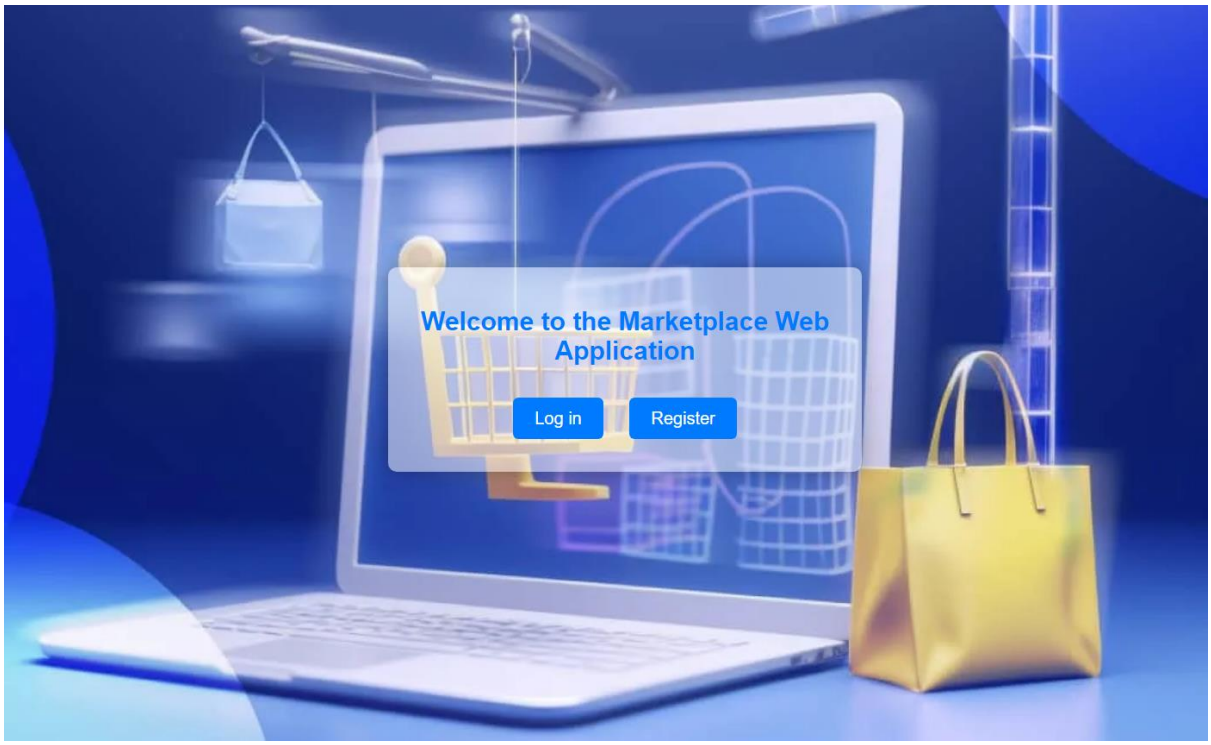
Οι βασικές υπηρεσίες που εξυπηρετούν τους χρήστες της εφαρμογής είναι οι εξής:

1. **Εγγραφή Χρηστών, σύνδεση και αποσύνδεση**: Οι χρήστες μπορούν να δημιουργήσουν λογαριασμό και να συνδεθούν για να έχουν πρόσβαση στις λειτουργίες της εφαρμογής αλλά και να αποσυνδεθούν από τον λογαριασμό τους.
2. **Καταχώρηση, Ενημέρωση και διαγραφή προϊόντων**: Οι χρήστες μπορούν να ανεβάζουν λεπτομερές πληροφορίες και φωτογραφίες για τα προϊόντα που θέλουν να πουλήσουν. Θα πρέπει να συμπληρωθούν τουλάχιστον τα πεδία Name, Description και Price. Αφού γίνει έλεγχος ότι έχουν συμπληρωθεί τα πεδία αυτά το προϊόν καταχωρείται στην βάση δεδομένων και εμφανίζεται στην σελίδα Dashboard έτσι ώστε να το δουν και οι άλλοι χρήστες της εφαρμογής. Επίσης ο χρήστης μπορεί να ενημερώσει οποιαδήποτε πληροφορία επιθυμεί από τα προϊόντα του, να αλλάξει την διαθεσιμότητα (αν βάλει ότι είναι μη διαθέσιμο οι χρήστες της εφαρμογής δεν έχουν την δυνατότητα να επικοινωνήσουν με τον χρήστη για το συγκεκριμένο προϊόν) και ακόμα να μπορούν να τα διαγράψουν.
3. **Αναζήτηση και Φιλτράρισμα προϊόντων**: Οι χρήστες μπορούν να αναζητήσουν προϊόντα κάτω από τον τίτλο “Products from other Users” από την μπάρα αναζήτησης (search bar) και να φιλτράρουν τα αποτελέσματα με βάση τα πιο καινούρια προϊόντα που έχουν προσθέτει στην πλατφόρμα ή τα πιο παλιότερα. Κάθε προϊόν αναγράφει τον τίτλο ,από κάτω την φωτογραφία του προϊόντος και την τιμή του, αριστερά κάτω την διαθεσιμότητα του (Available), στο κέντρο ένα κουμπί “View Details” το οποίο αν το πατήσει ο χρήστης θα ανακατευθυνθεί στην σελίδα του προϊόντος με τις λεπτομερές πληροφορίες του προϊόντος, και τέλος δεξιά την φωτογραφία προφίλ με το όνομα του χρήστη που καταχώρισε το προϊόν.
4. **Προβολή προφίλ των χρηστών**: Υπάρχουν δύο επιλογές για να δει ένας χρήστης το προφίλ ενός άλλου χρήστη: ο πρώτος τρόπος είναι να πατηθεί το όνομα του χρήστη δίπλα από την ένδειξη “Posted by:” από την απεικόνιση του προϊόντος στην σελίδα dashboard και ο άλλος τρόπος είναι στην σελίδα με τα μηνύματα να γραφτεί το όνομα του χρήστη στο search bar αναζήτησης χρηστών (κάτω από την ένδειξη User List). Στον δεύτερο τρόπο με το που εμφανιστεί το όνομα πρώτο πρώτο στην αναζήτηση και πατηθεί από τον χρήστη αριστερά εμφανίζεται μια καρτέλα “Profile Information” με τις αναλυτικές πληροφορίες του συγκεκριμένου χρήστη. Το προφίλ του χρήστη περιέχει πληροφορίες όπως όνομα, email, πότε έκανε εγγραφή στην πλατφόρμα, ενδιαφέροντα, φωτογραφία προφίλ, και links με τα προϊόντα που έχει ανεβάσει (ανεξάρτητα από το αν είναι διαθέσιμα ή όχι).

5. **Απεικόνιση λεπτομερειών προϊόντος**: Η απεικόνιση ενός προϊόντος (στην λεπτομερή περιγραφή "View Details") περιλαμβάνει το όνομα, την τιμή, την κατηγορία, την μέθοδο αποστολής, την τοποθεσία, την διαθεσιμότητα και την φωτογραφία του προϊόντος. Ο χρήστης μπορεί να εξετάσει τις λεπτομέρειες αυτές και να κρίνει αν τον ενδιαφέρει το προϊόν με βάση τις πληροφορίες αυτές.
6. **Προσθήκη προϊόντος στα αγαπημένα**: Στην απεικόνιση ενός προϊόντος ο χρήστης έχει την δυνατότητα να αποθηκεύσει το συγκεκριμένο προϊόν στα αγαπημένα του (favorites). Ο χρήστης απλά πατάει το κίτρινο κουμπί "Add to Favorites" και έτσι η αποθήκευση αυτή καταχωρείται στην βάση δεδομένών. Για να προβάλει τα αγαπημένα του προϊόντα πηγαίνει στην σελίδα Profile στην ένδειξη με τίτλο "My Favourite Products" . Εκεί φαίνονται ο τίτλος, η φωτογραφία και η τιμή κάθε αγαπημένου προϊόντος. Υπάρχουν και δύο κουμπιά για να δει τις λεπτομέρειες του προϊόντος "View Details" και να αφαιρέσει το προϊόν από τα αγαπημένα "Remove from Favorites".
7. **Διαχείριση Προφίλ:** Οι χρήστες έχουν τη δυνατότητα να διαχειρίζονται το προφίλ τους, να προσθέτουν πληροφορίες και να τις ενημερώνουν, να αλλάζουν φωτογραφία προφίλ, να αλλάζουν κωδικούς πρόσβασης κλπ. Υπάρχει και η δυνατότητα να διαγράψουν τον λογαριασμό τους, εφόσον το επιθυμούν.
8. **Σύστημα Μηνυμάτων:** Υπάρχει η δυνατότητα ανταλλαγής μηνυμάτων μεταξύ των χρηστών για την οργάνωση των συναλλαγών. Ο χρήστης επιλέγει τον χρήστη από την λίστα χρηστών (User List) ή κάνει αναζήτηση του ονόματος για να βρει κάποιον συγκεκριμένο. Εφόσον επιλεχθεί ο χρήστης, εμφανίζεται στο κέντρο της σελίδας η ιστορία μηνυμάτων (Message History) και αριστερά το Profile Information του χρήστη. Αν δεν υπάρχουν μηνύματα μεταξύ των χρηστών εμφανίζεται το μήνυμα "No messages found" αλλιώς αν έχει υπάρξει επικοινωνία με τους χρήστες εμφανίζεται η ιστορία των μηνυμάτων. Το περιεχόμενο τον μηνυμάτων εμφανίζει τα εξής στοιχεία: τα ονόματα του παραλήπτη ή του αποστολέα, το περιεχόμενο, και την ημερομηνία που στάλθηκε. Στην περίπτωση που έχει διαβαστεί από τον παραλήπτη ο αποστολέας λαμβάνει ένδειξη "read at:" μαζί την ημερομηνία που διαβάστηκε. Επίσης ο αποστολέας έχει την δυνατότητα να διαγράψει τα μηνύματα που έχει στείλει με αποτέλεσμα αυτά (εφόσον διαγραφτούν) να μην εμφανίζονται πια στον παραλήπτη και να διαγράφονται και από την βάση δεδομένών.
9. **Ειδοποιήσεις μηνυμάτων:** Υπάρχει σύστημα ειδοποιήσεων για να ενημερώνει τους χρήστες για τα νέα μηνύματα που έχουν λάβει από άλλους χρήστες. Μόλις ο παραλήπτης λάβει ένα νέο μήνυμα, σε οποιαδήποτε σελίδα βρίσκεται (εκτός από αυτή των messages γιατί εκεί βλέπει απευθείας τις ειδοποιήσεις) εμφανίζεται μια ειδοποίηση alert με ήχο ότι έλαβε ένα καινούριο μήνυμα από τον συγκεκριμένο αποστολέα μαζί με το κείμενο. Ο χρήστης πατάει "OK" πάνω στην ειδοποίηση και στην σελίδα dashboard όπως και στο menu εμφανίζεται ο αριθμός των συνολικών μηνυμάτων που έχει λάβει. Παρακάτω θα χρησιμοποιηθούν εικόνες για την καλύτερη κατανόηση του συστήματος αυτού.

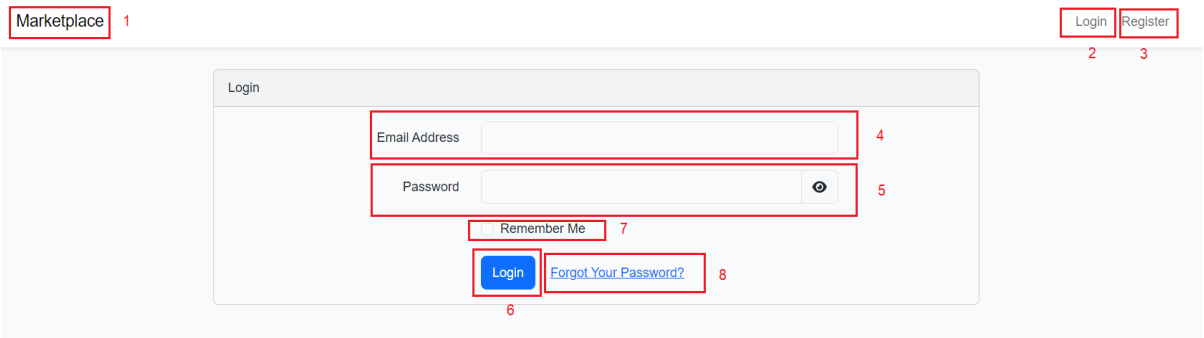
5.2 Διαδικασία σύνδεσης στην εφαρμογή (login)

5.2.1 Αρχική Σελίδα (Welcome page):



Εικόνα 14 - Επιλογή σύνδεση (login) ή εγγραφής (register)

5.2.2 Σελίδα Login:



Εικόνα 15 - Σελίδα Login

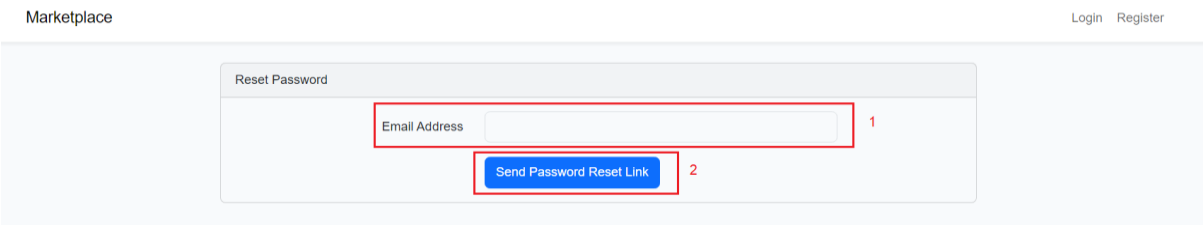
Η κεφαλίδα περιέχει το όνομα της εφαρμογής (1), το Login (2) και το Register (3). Ο πίνακας κάτω από την κεφαλίδα περιέχει την φόρμα Login. Ο χρήστης γράφει το email του στο πεδίο “Email Address” (4) και τον κωδικό του στο πεδίο “Password” (5) για να συνδεθεί στην εφαρμογή. Πάνω δεξιά μπορεί να ανακατευθυνθεί στην σελίδα Register (3) αν δεν έχει κάνει εγγραφή ήδη και δίπλα από το κουμπί Login (6) μπορεί να πατήσει στο Link «Forgot your Password» (8) για να ανακατευθυνθεί στην σελίδα προκειμένου να ανακτήσει τον κωδικό του σε περίπτωση που τον έχει ξεχάσει. Επίσης υπάρχει και η δυνατότητα να αποθηκευτούν τα στοιχεία σύνδεσης του χρήστη πατώντας το κουτάκι “Remember Me” (7).

Δίπλα από το πεδίο “Password” (5) υπάρχει ένα σύμβολο με ματάκι στο οποίο όταν ο χρήστης πατήσει πάνω σε αυτό μπορεί να δει τον κωδικό του χωρίς bullets.

5.3 Διαδικασία ανάκτησης κωδικού

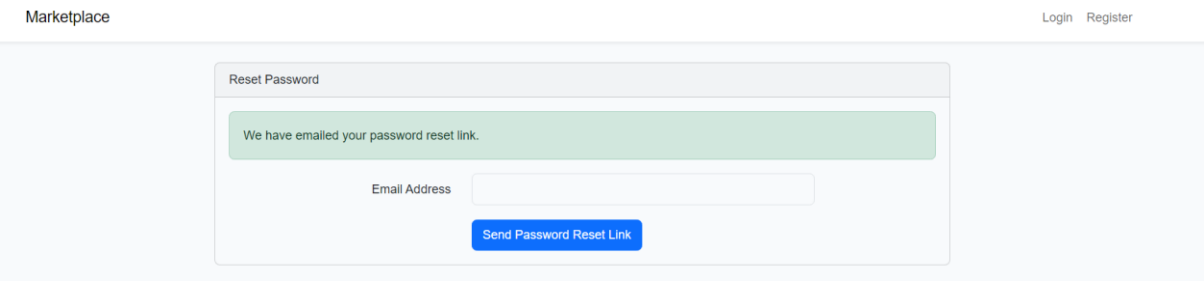
Παρακάτω ακολουθεί ένα παράδειγμα ανάκτησης κωδικού ως χρήστη “Laura” με email: “laura@aaa.aaa”

5.3.1 Σελίδα Forgot Password:



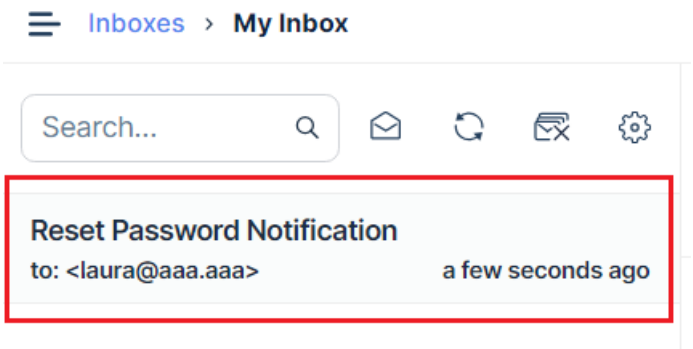
Εικόνα 16 - Σελίδα Forgot Password

Ο χρήστης προκειμένου να ανακτήσει τον κωδικό πρόσβασης του σε περίπτωση που τον έχει ξεχάσει μπορεί να πληκτρολογήσει το email του (1) και να πατήσει το κουμπί «Send Password Reset Link» (2).

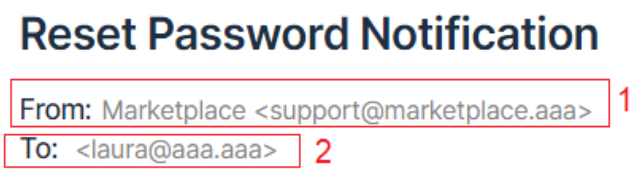


Εικόνα 17 - Αφού ο χρήστης έγραψε το email του στο πεδίο εμφανίζεται στη σελίδα ένα μήνυμα επιτυχίας ότι το password reset link στάλθηκε στο email του χρήστη

Στην συνέχεια ο χρήστης θα πρέπει να πάει στο email του και να ψάξει το Link που του έχει σταλθεί και να ακολουθήσει τις αντίστοιχες οδηγίες για την ανάκτηση του κωδικού του. Χρησιμοποιώντας την υπηρεσία **Mailtrap.io**, σε πραγματικό σενάριο το email στέλνεται με επιτυχία στον χρήστη όπως φαίνεται στην εικόνα παρακάτω.

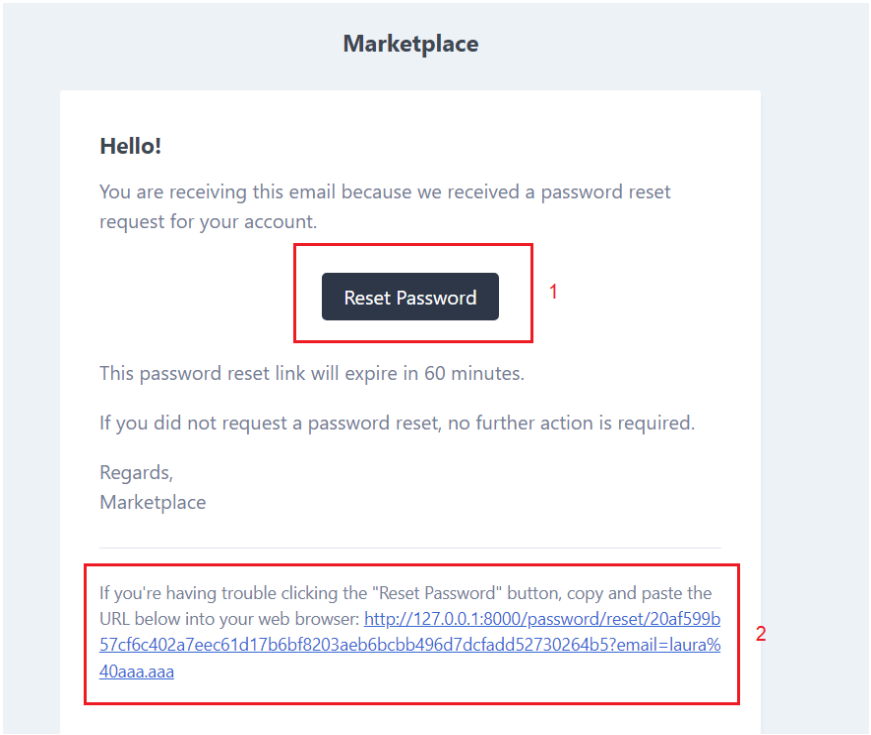


Εικόνα 18 - Εικονικό inbox υπηρεσίας mailtrap



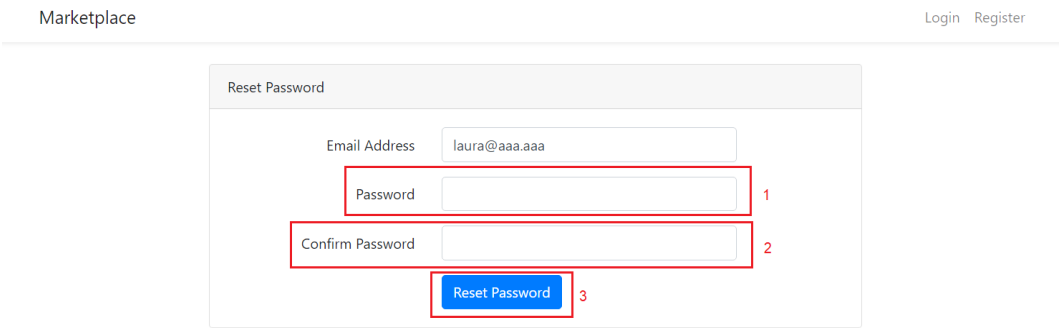
Εικόνα 19 - Ονόματα αποστολέα και παραλήπτη

Πάνω από το μήνυμα βλέπουμε το email του αποστολέα (1) και το email του παραλήπτη (2).



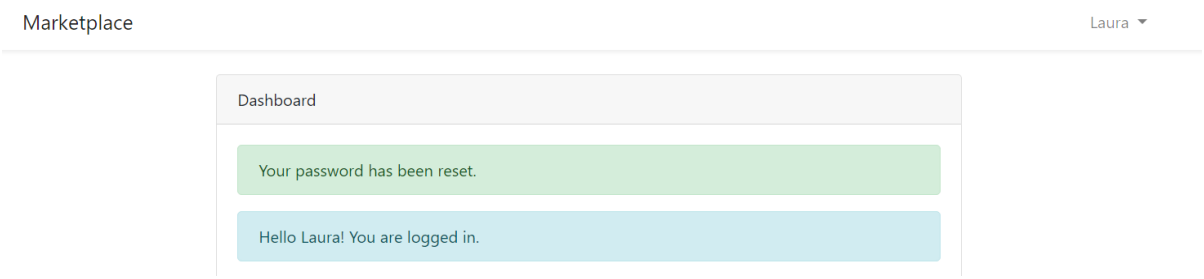
Εικόνα 20 - Περιεχόμενο μηνύματος email

Κάτω ακριβώς φαίνεται το περιεχόμενο του email. Ο χρήστης μπορεί να πατήσει πάνω στο 'Reset Password' κουμπί (1) για να ανακατευθυνθεί στην σελίδα για την ανάκτηση του κωδικού του, ή αν έχει πρόβλημα με το κουμπί να αντιγράψει τον URL σύνδεσμο και να τον κάνει επικόλληση σε μια καρτέλα περιήγησης.



Εικόνα 21 - Σελίδα ανάκτησης του κωδικού.

Ο χρήστης γράφει τον καινούριο κωδικό του στα πεδία Password (1) και Confirm Password (2) και τέλος πατάει το κουμπί “Reset Password” (3).



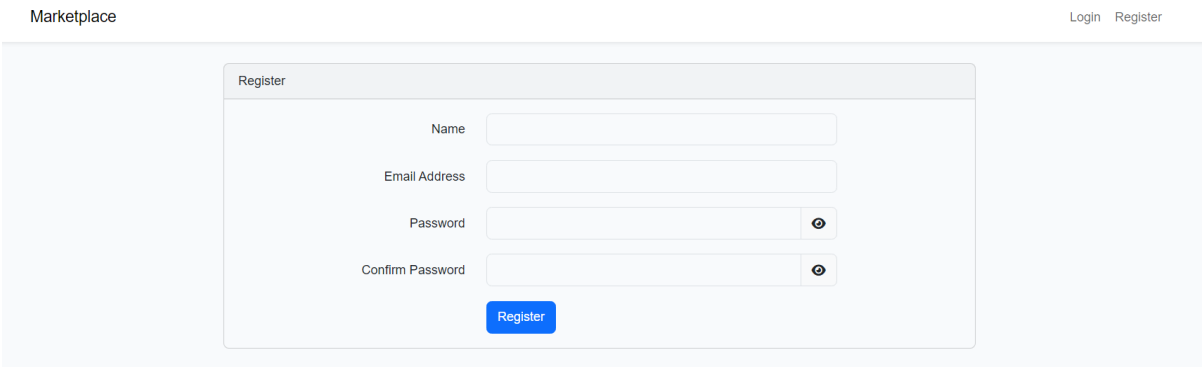
Εικόνα 22 - Μήνυμα επιτυχίας αλλαγής password στη σελίδα dashboard

Μόλις ο χρήστης πατήσει το κουμπί Reset Password, ο χρήστης ανακατευθύνεται στην αρχική σελίδα (Dashboard) και εμφανίζεται ένα μήνυμα επιτυχίας (με πράσινο) ότι ο κωδικός άλλαξε.

5.4 Εγγραφή στην εφαρμογή (register)

Σε περίπτωση που κάποιος δεν έχει γραφτεί στην πλατφόρμα ακόμη, μπορεί να κάνει την εγγραφή πάνω δεξιά στο κουμπί «Register».

5.4.1 Σελίδα Register:

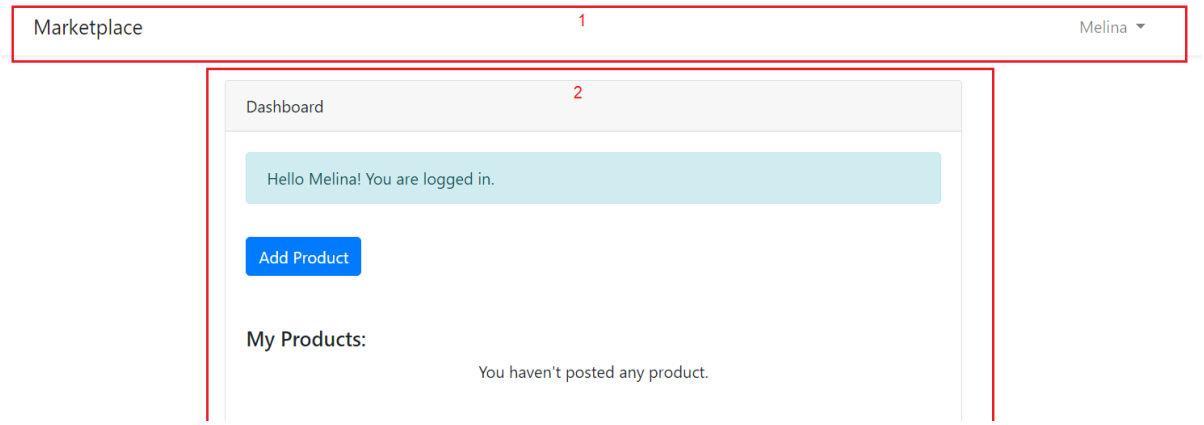


Εικόνα 23 - Σελίδα εγγραφής στην εφαρμογή (Register).

Αφού συμπληρώσει ο χρήστης τα πλαίσια της παραπάνω φόρμας θα καταχωρηθούν τα στοιχεία του στην βάση δεδομένων και θα δημιουργηθεί το προφίλ του ως χρήστης (user) στην εφαρμογή.

5.5 Dashboard σελίδα

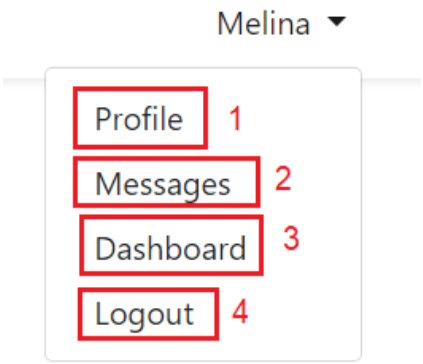
Εφόσον κάνουμε εγγραφή στην πλατφόρμα ως user “Melina”, δημιουργήθηκε επιτυχώς το προφίλ και έχει καταχωρηθεί ο user στην βάση. Η αρχική σελίδα χωρίζεται στην κεφαλίδα header (1) και στον πίνακα dashboard (2) με τα περιεχόμενα όπως φαίνεται παρακάτω στην εικόνα.



Εικόνα 24 - Σελίδα Dashboard

(1) Κεφαλίδα:

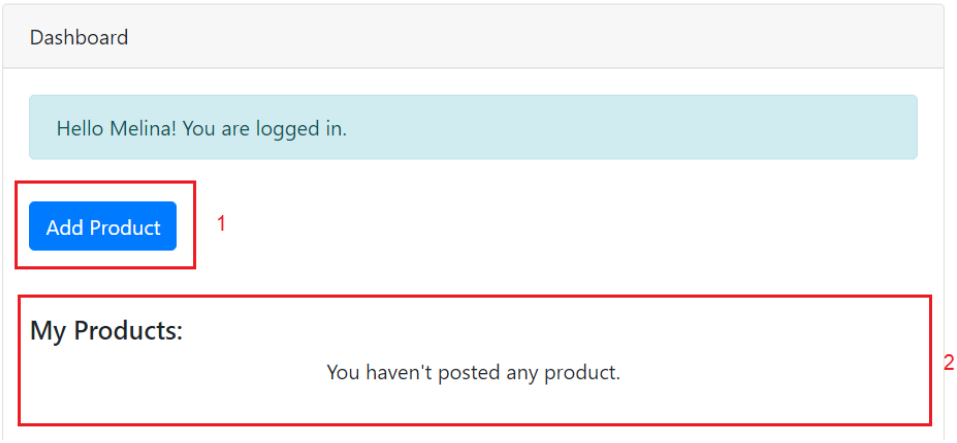
Η κεφαλίδα header αριστερά περιέχει το όνομα της εφαρμογής (Marketplace) και δεξιά το menu επιλογών για την ανακατεύθυνση στις άλλες σελίδες.



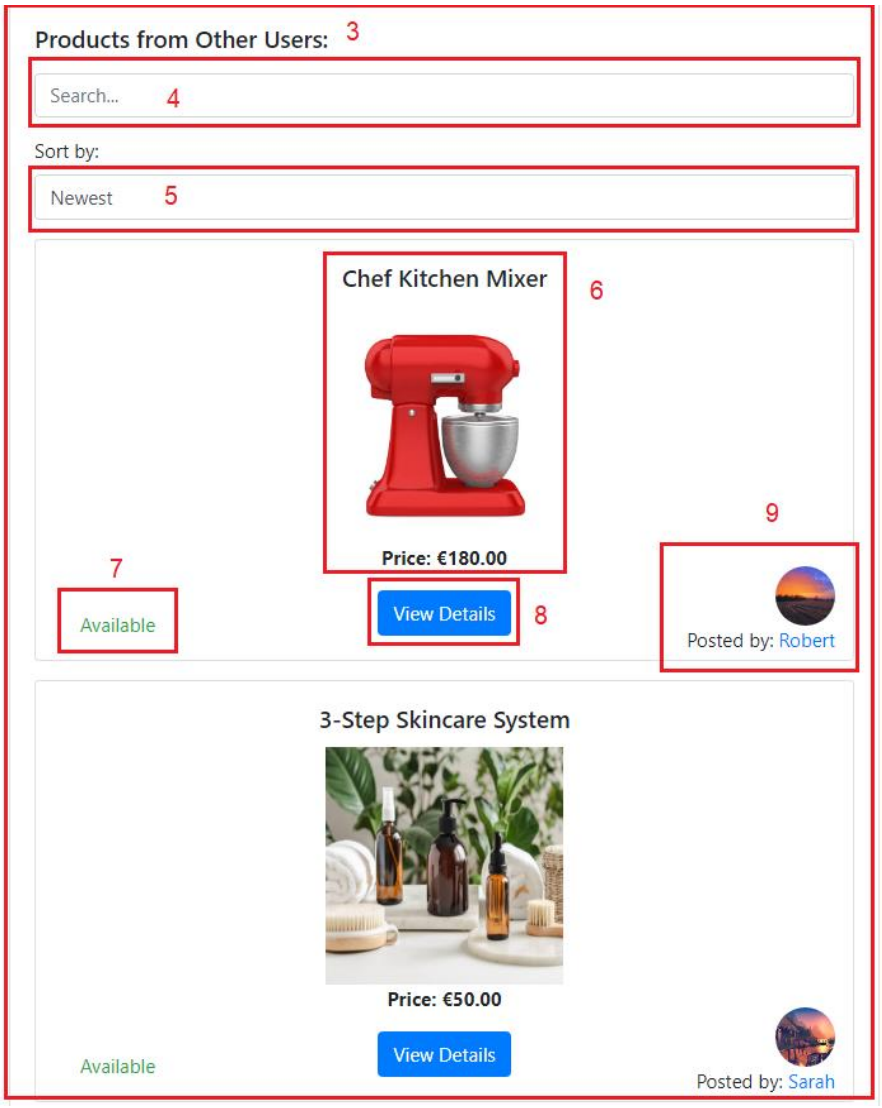
Εικόνα 25 - Μενού επιλογών

1. Ανακατεύθυνση στο προφίλ του χρήστη
2. Ανακατεύθυνση στη σελίδα με τα μηνύματα
3. Ανακατεύθυνση στην αρχική σελίδα Dashboard (αν ο χρήστης είναι ήδη στην σελίδα εκείνη απλά κάνει ανανέωση της σελίδας)
4. Αποσύνδεση από την πλατφόρμα

(2) Πίνακας Dashboard:



Εικόνα 26 - Πίνακας Dashboard



Εικόνα 27 - Υπόλοιπος πίνακας Dashboard

1. Προσθήκη προϊόντος
2. Καρτέλα με τα προϊόντα του συνδεδεμένου χρήστη που έχει ανεβάσει
3. Καρτέλα με τα προϊόντα όλων των υπόλοιπων χρηστών
4. Μπάρα αναζήτησης προϊόντων
5. Φίλτρο εύρεσης προϊόντων με βάση τα παλιότερα και τα καινούρια προϊόντα
6. Πληροφορίες Προϊόντος
7. Ένδειξη διαθεσιμότητας
8. Κουμπί προεπισκόπησης περισσοτέρων πληροφοριών για το προϊόν
9. Ο χρήστης που ανέβασε το προϊόν

5.6 Διαδικασία Edit και Delete profile

Κάθε χρήστης έχει την δυνατότητα να τροποποιήσει οποιαδήποτε πληροφορία επιθυμεί στο προφίλ του, ακόμα και να διαγράψει τον λογαριασμό του.

Εικόνα 28 - Σελίδα Profile

1. Καρτέλα My Profile
2. Καρτέλα My Favourite Products
3. Στοιχεία My Profile
4. Τροποποίηση Πληροφοριών
5. Διαγραφή λογαριασμού
6. Ανακατεύθυνση στο Dashboard

(4) Τροποποίηση πληροφοριών:

Εικόνα 29 - Σελίδα τροποποίησης πληροφοριών

1. Πεδίο ονόματος

2. Πεδίο email

3. Πεδίο προσθήκης φωτογραφίας προφίλ

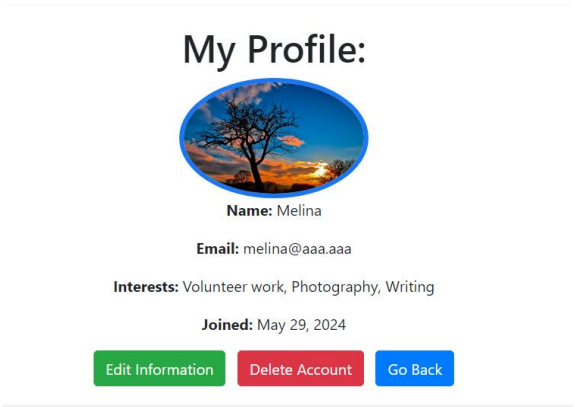
4. Πεδίο ενδιαφερόντων. Μπορεί να χρησιμοποιηθεί και ως περιγραφή του εαυτού ενός χρήστη

5. Πεδίο αλλαγής Password

6. Κουμπί αποθήκευσης πληροφοριών

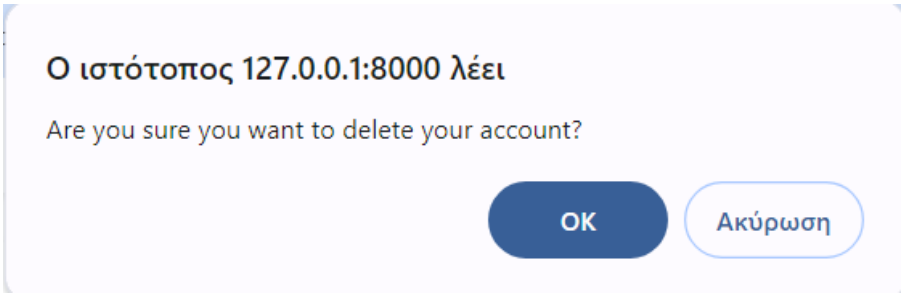
7. Ανακατεύθυνση στο Dashboard

Για παράδειγμα, θέλουμε να προσθέσουμε μια φωτογραφία προφίλ και να γράψουμε στο πεδίο ενδιαφέροντα. Το αποτέλεσμα της αλλαγής φαίνεται στην παρακάτω εικόνα:



Εικόνα 30 - Εμφάνιση αποτελέσματος αφού προστέθηκαν τα πεδία φωτογραφία και ενδιαφέροντα.

(5) Διαγραφή λογαριασμού:

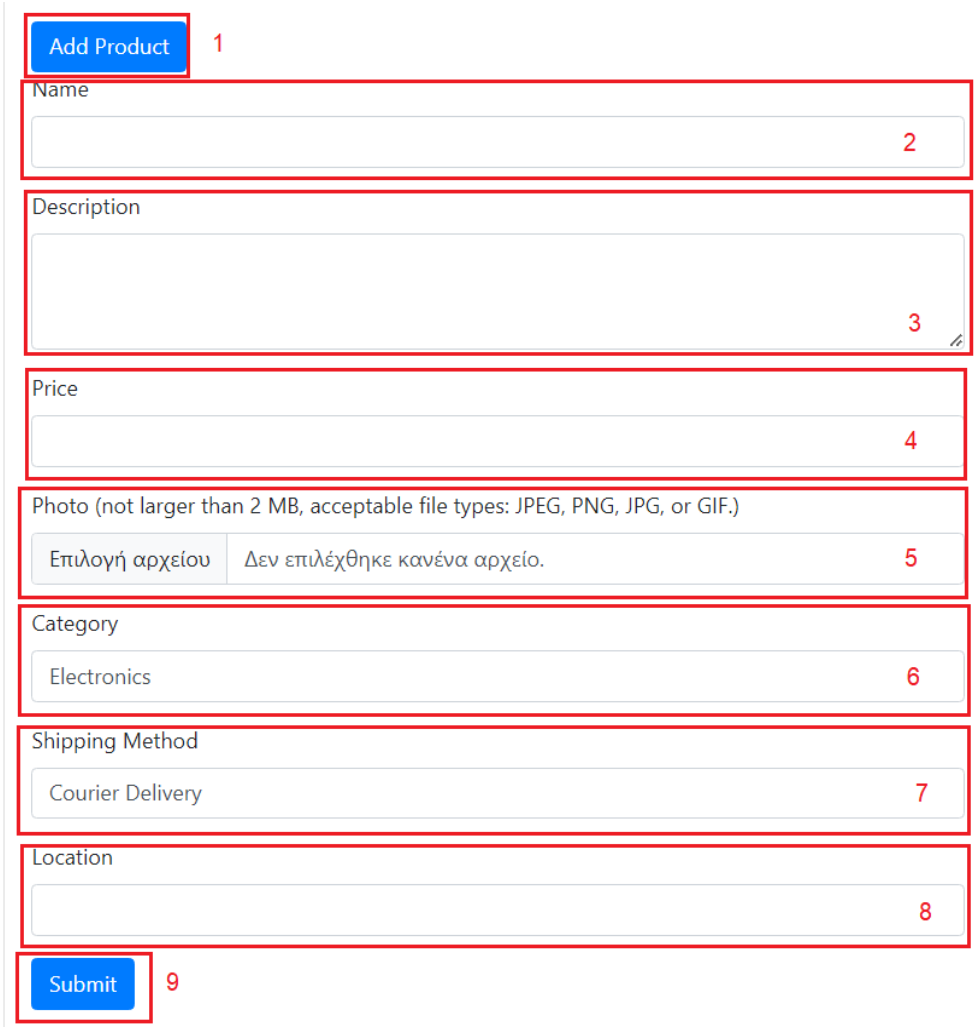


Εικόνα 31 - Μήνυμα ειδοποίησης alert για διαγραφή λογαριασμού

Πατώντας το κουμπί «OK» ο λογαριασμός διαγράφεται οριστικά από την βάση δεδομένων. Αν θέλει ο χρήστης να ακυρώσει την διαδικασία αυτή πατάει το κουμπί «Ακύρωση».

5.7 Διαδικασία Add, Edit και Delete product

Η φόρμα για την προσθήκη ενός προϊόντος βρίσκεται στην σελίδα dashboard πατώντας το κουμπί “Add Product”. Εκεί ο χρήστης συμπληρώνει τα πεδία για τις πληροφορίες του προϊόντος.



Εικόνα 32 - Φόρμα συμπλήρωσης πληροφοριών προϊόντος

1. Κουμπί προσθήκης προϊόντος για εμφάνιση των παρακάτω πεδίων

2. Προσθήκη ονόματος προϊόντος

3. Προσθήκη περιγραφή προϊόντος

4. Προσθήκη τιμής προϊόντος

5. Προσθήκη φωτογραφίας προϊόντος

6. Προσθήκη κατηγορίας προϊόντος. Οι κατηγορίες (6) είναι οι εξής: “Electronics”, “Clothing and Accessories”, “Home and Garden”, “Sports”, “Books and Media”, “Health and Beauty”.

7. Προσθήκη μεθόδου αποστολής. Οι μέθοδοι αποστολής (2) είναι οι εξής: “Courier Delivery”, “User Pickup”.

8. Προσθήκη τοποθεσίας

9. Αποθήκευση πληροφορίων και κατοχύρωση προϊόντος στη βάση δεδομένων

Για παράδειγμα, προσθέτουμε ένα προϊόν. Αφού πατήσουμε το κουμπί “submit” πλέον στο Dashboard θα φαίνεται το προϊόν που έχουμε ανεβάσει κάτω από τον τίτλο “My products”:

Εικόνα 33 - Εμφάνιση προϊόντος στη καρτέλα My products

1. Καρτέλα My Products

2. Πληροφορίες προϊόντος (τίτλος και φωτογραφία)

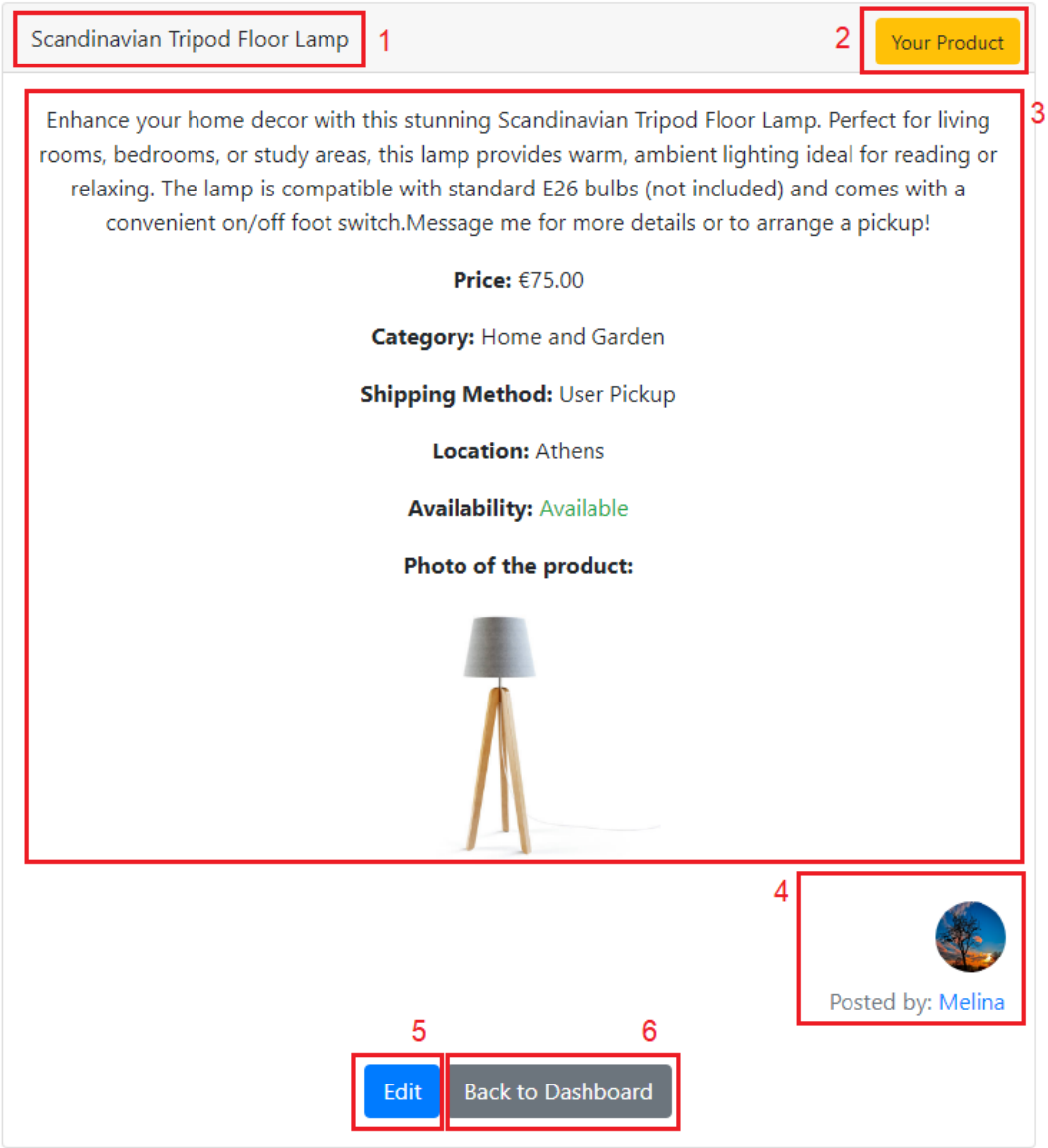
3. Κουμπί «View Details» για λεπτομερείς πληροφορίες προϊόντος (όπως φαίνεται στους άλλος χρήστες)

4. Κουμπί τροποποίησης πληροφοριών προϊόντος

5. Κουμπί διαγραφής προϊόντος

Φυσικά ο κάθε χρήστης μπορεί να προσθέσει περισσότερα από ένα προϊόντα για πώληση. Για να κάνει navigation στα υπόλοιπα προϊόντα απλά πατάει δεξιά ή αριστερά στα κενά του προϊόντος (ανάμεσα στο κενό 1 και 2).

(3) Εμφάνιση λεπτομερών πληροφοριών προϊόντος(για τον χρήστη στον οποίο ανήκει το προϊόν):

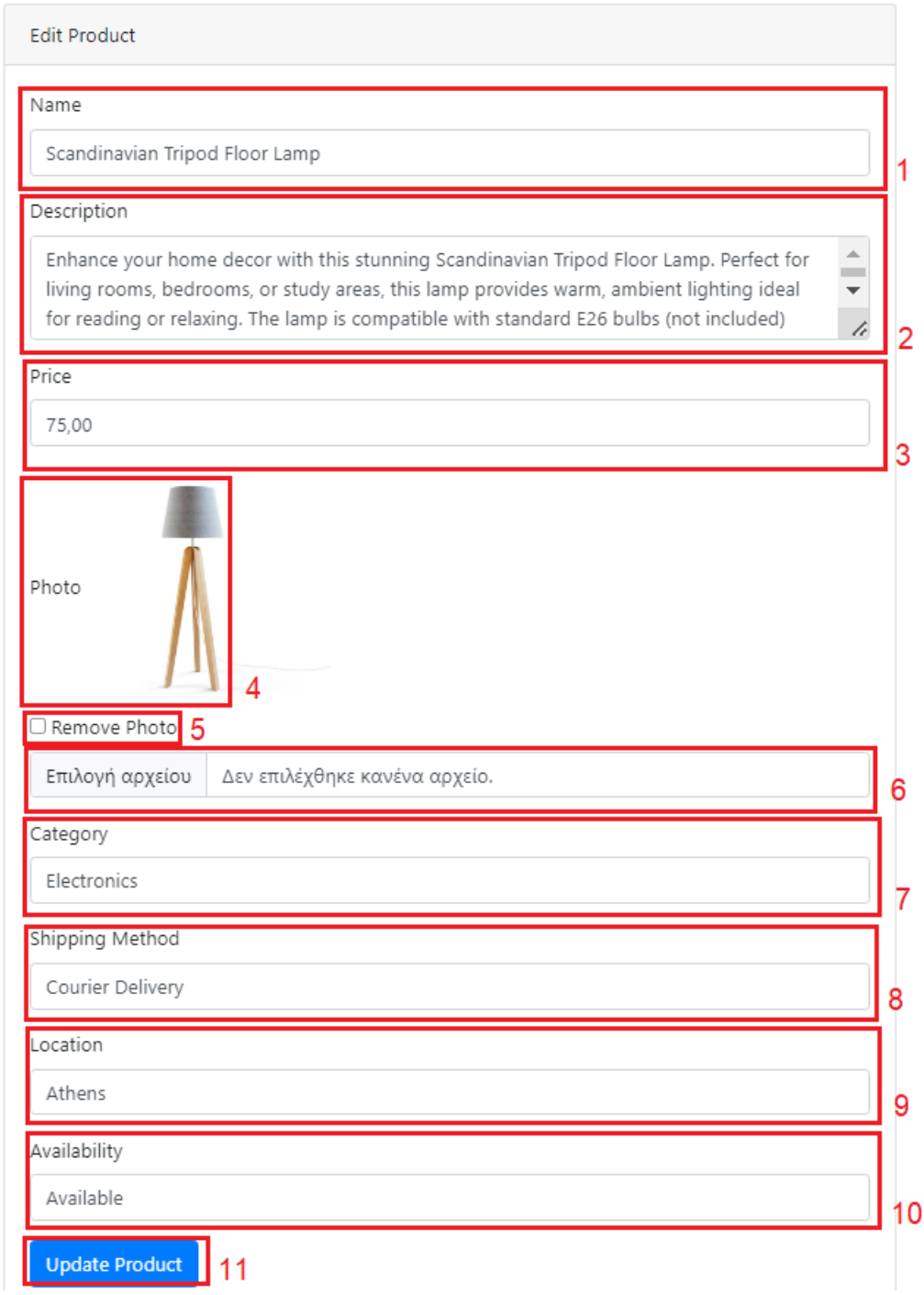


Εικόνα 34 - Σελίδα πληροφοριών προϊόντος

1. Όνομα προϊόντος
2. Ένδειξη ότι το προϊόν είναι του χρήστη
3. Πληροφορίες προϊόντος όπως θα φαίνονται στους άλλους χρήστες
4. Ένδειξη ποιος ανέβασε το προϊόν (ο χρήστης)
5. Τροποποίηση προϊόντος
6. Ανακατεύθυνση στο Dashboard

(4) Τροποποίηση πληροφοριών προϊόντος:

Πατώνοντας το κουμπί “Edit” ανακατευθυνόμαστε στην σελίδα τροποποίησης στοιχείων ενός προϊόντος.



Εικόνα 35 - Φόρμα τροποποίησης στοιχείων προϊόντος

1. Πεδίο αλλαγής ονόματος προϊόντος
2. Πεδίο αλλαγής περιγραφής προϊόντος
3. Πεδίο αλλαγής τιμής προϊόντος

4. Πεδίο εμφάνισης φωτογραφίας προϊόντος που έχει προστεθεί

5. Κάνοντας κλικ στο κουτάκι η φωτογραφία αφαιρείται από το προϊόν

6. Πεδίο αλλαγής φωτογραφίας προϊόντος

7. Προσθήκη κατηγορίας προϊόντος

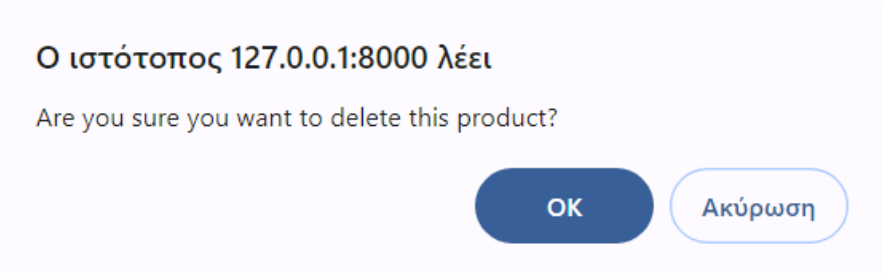
8. Μέθοδος αποστολής προϊόντος

9. Πεδίο αλλαγής τοποθεσίας προϊόντος

10. Πεδίο αλλαγής διαθεσιμότητας προϊόντος

11. Κουμπί αποθήκευσης αλλαγών

(5) Διαγραφή προϊόντος:



Εικόνα 36 - Μήνυμα ειδοποίησης alert για διαγραφή προϊόντος

Πατώντας το κουμπί «OK» το προϊόν διαγράφεται οριστικά από την βάση δεδομένων. Αν θέλει ο χρήστης να ακυρώσει την διαδικασία αυτή πατάει το κουμπί «Ακύρωση».

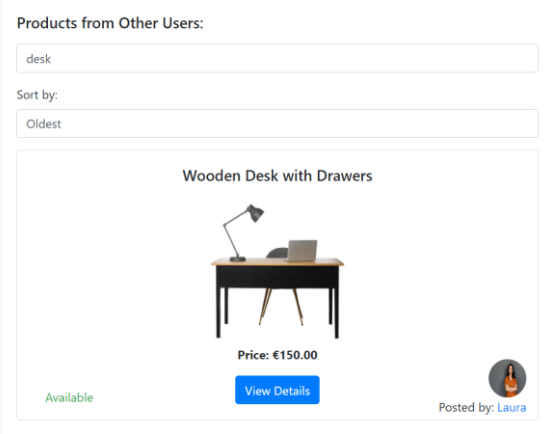
5.8 Προβολή προϊόντων από τους χρήστες

Η διαδικασία αναζήτησης και περιήγησης στη συλλογή των προϊόντων γίνεται πιο εύκολη μέσω των παρακάτω εργαλείων:

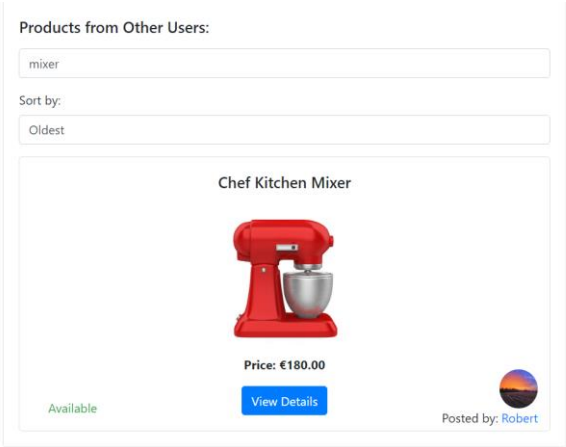
5.8.1 Γραμμή αναζήτησης (Search bar)

Για την ευκολία των χρηστών να βρουν συγκεκριμένα προϊόντα, έχει ενσωματωθεί ένα search bar. Οι χρήστες μπορούν να πληκτρολογήσουν λέξεις-κλειδιά και να φιλτράρουν τα αποτελέσματα των προϊόντων σε πραγματικό χρόνο, κάνοντας την αναζήτηση γρηγορότερη και πιο αποτελεσματική. (Η συγκεκριμένη λειτουργία επιτυγχάνεται μέσω της συνάρτησης searchProducts() στην κεφαλίδα script του αρχείου home.blade.php). Η λειτουργία της συγκεκριμένης συνάρτησης έχει ως εξής: υλοποιεί μια λειτουργία αναζήτησης προϊόντων σε μια λίστα καρτών προϊόντων. Πρώτα, λαμβάνει την τιμή που εισάγει ο χρήστης στο πεδίο αναζήτησης και κάνει την αναζήτηση ανεξάρτητη από το μέγεθος γραμμάτων (πεζά ή κεφαλαία). Έπειτα ανακτά τη λίστα με τις κάρτες προϊόντων και για κάθε κάρτα ελέγχει αν ο τίτλος ή η περιγραφή περιέχουν τη λέξη-κλειδί. Αν ο τίτλος ή η περιγραφή περιέχουν τη λέξη-κλειδί, η κάρτα παραμένει ορατή. Αν όχι, η κάρτα αποκρύπτεται. Με αυτόν τον τρόπο, μόνο οι κάρτες που ταιριάζουν με την αναζήτηση του χρήστη εμφανίζονται στη σελίδα.

Παρακάτω στις εικόνες φαίνονται παραδείγματα χρήσης της λειτουργίας αυτής:



Εικόνα 37 - Ο χρήστης αναζητά με την λέξη κλειδί "desk" και το αποτέλεσμα εμφανίζεται πρώτο στη λίστα

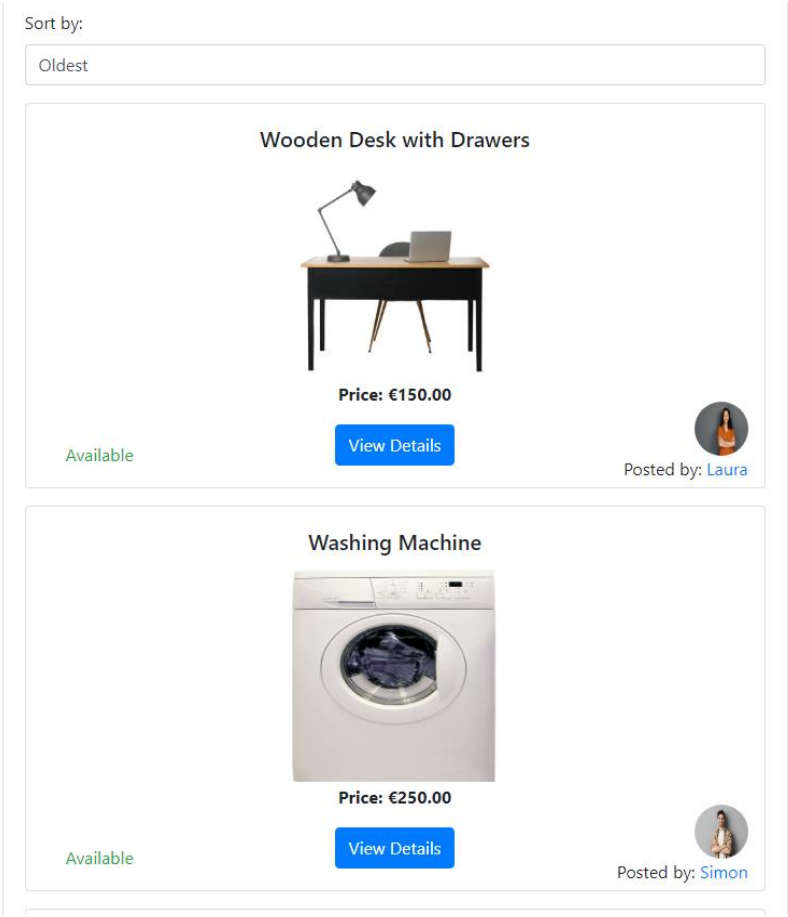


Εικόνα 38 - Ο χρήστης αναζητά με την λέξη κλειδί "mixer" και το αποτέλεσμα εμφανίζεται πρώτο στη λίστα

5.8.2 Φίλτρο ταξινόμησης για νέα ή παλιά προϊόντα (Sort by newest or oldest products)

Για να δώσουμε στους χρήστες τη δυνατότητα να ταξινομούν τα προϊόντα κατά χρονολογική σειρά, έχει προστεθεί ένα φίλτρο ταξινόμησης. Αυτό επιτρέπει στους χρήστες να βλέπουν πρώτα τα νεότερα ή τα παλαιότερα προϊόντα ανάλογα με την προτίμησή τους.(Η συγκεκριμένη λειτουργία επιτυγχάνεται μέσω της συνάρτησης sortProducts() στην κεφαλίδα script του αρχείου home.blade.php). Η συγκεκριμένη συνάρτηση αρχικά ανακτά την τιμή του φίλτρου ταξινόμησης και στη συνέχεια ανακτά τη λίστα με τις κάρτες προϊόντων. Χρησιμοποιεί έναν επαναληπτικό μηχανισμό (while loop) όπου ελέγχει αν χρειάζεται ανταλλαγή καρτών (shouldSwitch). Εάν χρειάζεται ανταλλαγή, η συνάρτηση τοποθετεί την κάρτα που ακολουθεί μετά την τρέχουσα κάρτα στη σωστή θέση. Η ταξινόμηση γίνεται ανάλογα με την επιλογή του χρήστη στο φίλτρο (newest ή oldest), όπου συγκρίνονται οι ημερομηνίες δημιουργίας των προϊόντων (πεδίο created_at για κάθε προϊόν).

Παρακάτω αναπαρίσταται η λειτουργία αυτή:



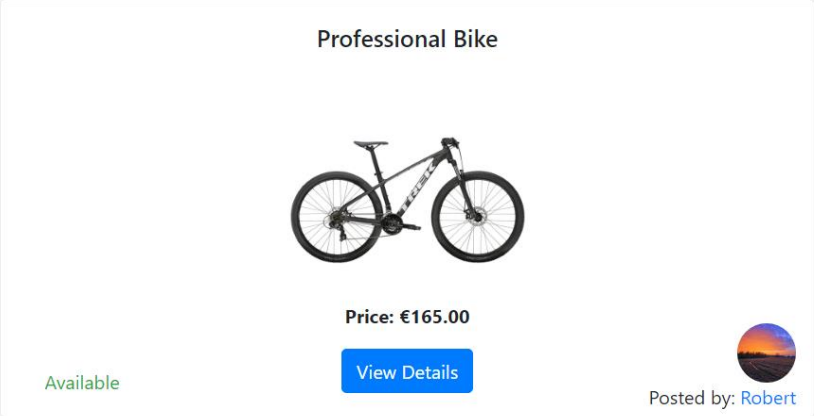
Εικόνα 39 - Ταξινόμηση παλιότερων προϊόντων

id	user_id	name	description	price	created_at	updated_at	photo_product	category	shipping_method	map	availability
48	13	Wooden Desk with Drawers	Wooden desk in good condition with three spacious ...	150.00	2024-05-29 15:08:30	2024-05-29 15:08:30	1716995310.jpg	Home and Garden	User Pickup	Athens	1
49	12	Washing Machine	Washing machine with 8kg capacity and 1400 rpm. In...	250.00	2024-05-29 15:11:05	2024-05-29 15:11:05	1716995465.jpg	Home and Garden	Courier	Athens	1

Εικόνα 40 - Πεδίο "created at" με κόκκινο χρώμα

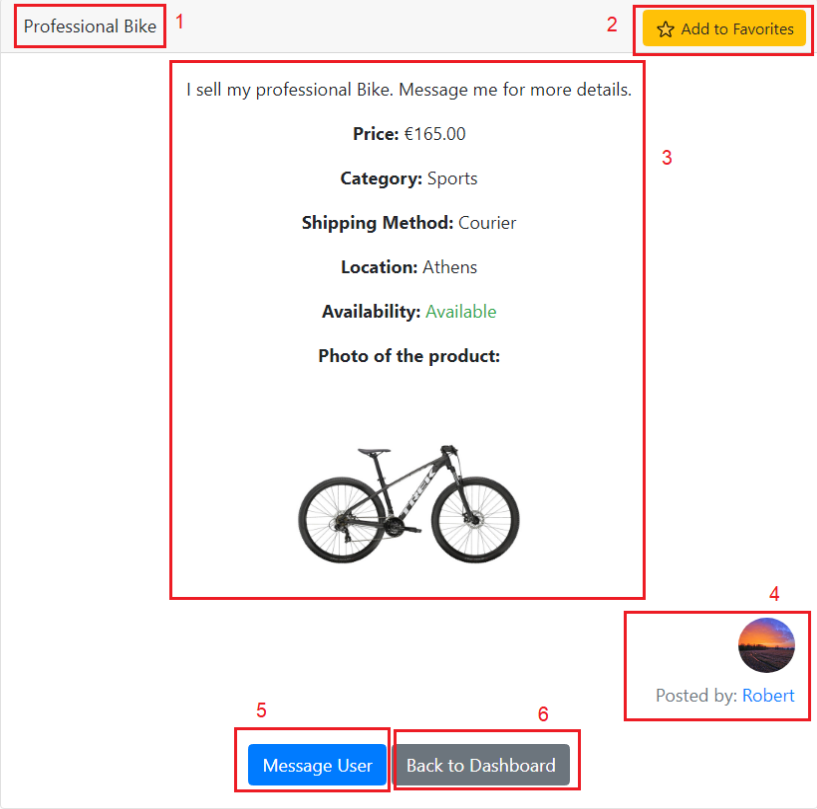
5.8.3 Προβολή Λεπτομερειών για ένα προϊόν (View Details)

Οι χρήστες μπορούν να βλέπουν λεπτομέρειες για κάθε προϊόν κάνοντας κλικ στο κουμπί "View Details". Αυτό το κουμπί τους κατευθύνει σε μια σελίδα που περιέχει όλες τις σχετικές πληροφορίες για το συγκεκριμένο προϊόν, όπως την περιγραφή, την τιμή, φωτογραφία κτλ.



Εικόνα 41 - Προβολή ενός προϊόντος

Πατώντας το κουμπί “View Details” εμφανίζεται η εξής σελίδα με τα στοιχεία:



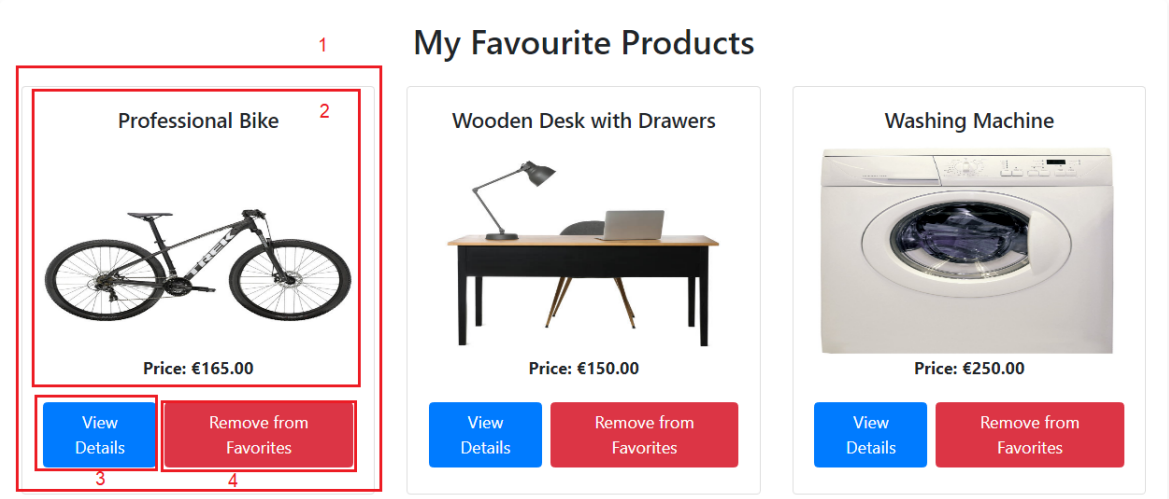
Εικόνα 42 - Προβολή λεπτομερειών προϊόντος

1. Όνομα προϊόντος
2. Κουμπί “Add to Favorites” για προσθήκη στα αγαπημένα.
3. Πληροφορίες προϊόντος
4. Ένδειξη ποιος ανέβασε το προϊόν
5. Κουμπί “Message User” για αποστολή αυτοματοποιημένου μηνύματος στον χρήστη
6. Ανακατεύθυνση στο Dashboard

(2) Προσθήκη στα αγαπημένα (Add to Favorites)

Έχει προστεθεί η δυνατότητα στους χρήστες να προσθέτουν προϊόντα στα αγαπημένα τους. Με αυτόν τον τρόπο, μπορούν να αποθηκεύουν προϊόντα που τους ενδιαφέρουν για μελλοντική αναφορά ή αγορά. Αυτή η λειτουργία έχει ενσωματωθεί μέσω ενός κουμπιού "Add to Favorites".

Οι χρήστες θα πρέπει να ανακατευθυνθούν στην σελίδα με τις πληροφορίες του προφίλ τους για να δουν τα αποθηκευμένα τους προϊόντα.



Εικόνα 43 - Προβολή αγαπημένων προϊόντων

1. Κάρτα προϊόντος
2. Σύντομες πληροφορίες
3. Κουμπί προεπισκόπησης λεπτομερειών
4. Αφαίρεση από τα αγαπημένα

Τα αγαπημένα προϊόντα προστίθενται στη λίστα των αποθηκευμένων βάση την χρονολογική σειρά που έχουν προστεθεί στα αγαπημένα για κάθε χρήστη. Το πιο παλιότερο που έχει προστεθεί στη συλλογή εμφανίζεται πρώτο στη λίστα κτλ.

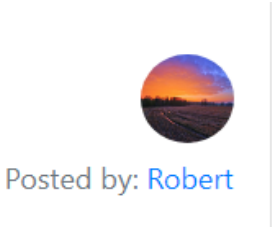
(5) Αποστολή Μηνύματος στον Χρήστη (Message User)

Οι χρήστες έχουν τη δυνατότητα να στείλουν μήνυμα στον πωλητή ενός προϊόντος. Αυτή η δυνατότητα αναλύεται παρακάτω στην ενότητα 5.10. Μέσω αυτής της λειτουργίας, μπορούν να επικοινωνούν άμεσα με τον πωλητή για ερωτήσεις ή διαπραγματεύσεις σχετικά με το προϊόν.

5.9 Προβολή προφίλ ενός χρήστη

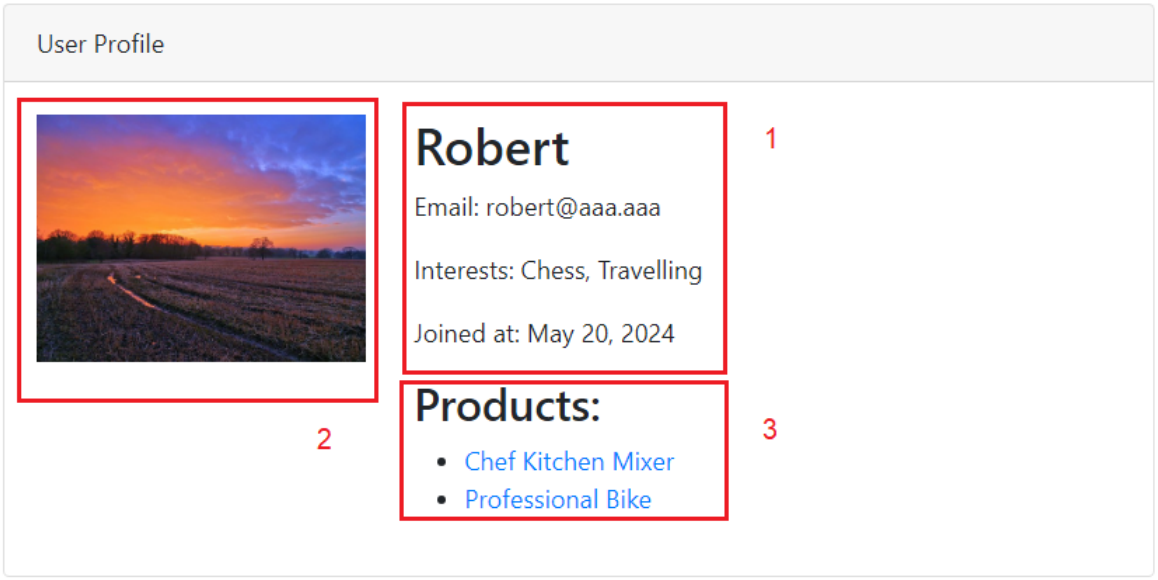
Όπως έχει ήδη αναφερθεί στην ενότητα βασικές υπηρεσίες, υπάρχουν δύο τρόποι για να δει ένας χρήστης το προφίλ ενός χρήστη: ο ένας είναι να πατήσει πάνω στο όνομα του δίπλα από το posted by μιας αγγελίας και ο άλλος είναι να ανακατευθυνθεί στη σελίδα των μηνυμάτων και να ψάξει το όνομα του user στο search bar του user list, και αφού βρεθεί, να επιλεχθεί για να εμφανιστεί το profile information του συγκεκριμένου χρήστη. Παρακάτω αναπαρίστανται παραδείγματα από τους δυο τρόπους αυτούς.

1. Πρώτος τρόπος: πάτημα στο όνομα του User (δίπλα από το Posted by:) είτε από την καρτέλα αγγελίας στο dashboard είτε στη λεπτομερή περιγραφή της αγγελίας.



Εικόνα 44 - Φωτογραφία προφίλ χρήστη με όνομα (link) για προβολή προφίλ

Πατώντας το όνομα ανακατευθυνόμαστε στο user profile του χρήστη Robert:



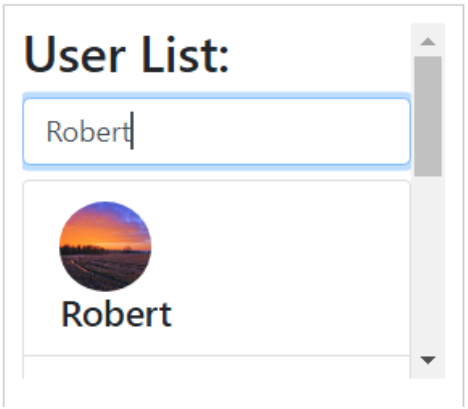
Εικόνα 45 - Σελίδα προβολής προφίλ χρήστη

1. Πληροφορίες του χρήστη: Name, Email, Interests, Joined at.
2. Φωτογραφία προφίλ του χρήστη
3. Links προϊόντων που έχει δημοσιεύσει για πώληση (διαθέσιμα και μη)

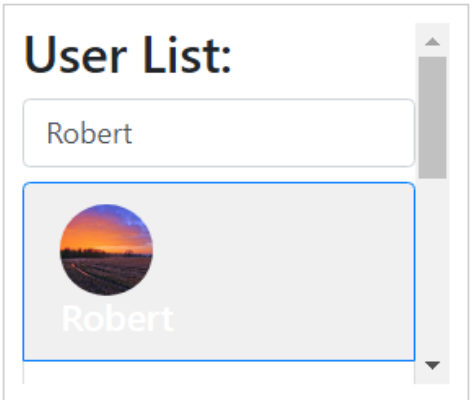
2. Δεύτερος τρόπος: Εύρεση του όνομα του χρήστη από το search bar (search user).

Βήμα 1: Ανακατεύθυνση στην σελίδα Messages από το menu

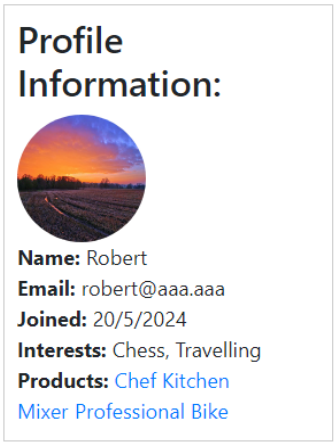
Βήμα 2: Γράφουμε το όνομα του User στο search bar κάτω από το User List



Βήμα 3: Επιλέγουμε τον χρήστη που εμφανίστηκε πρώτος (σωστό αποτέλεσμα)



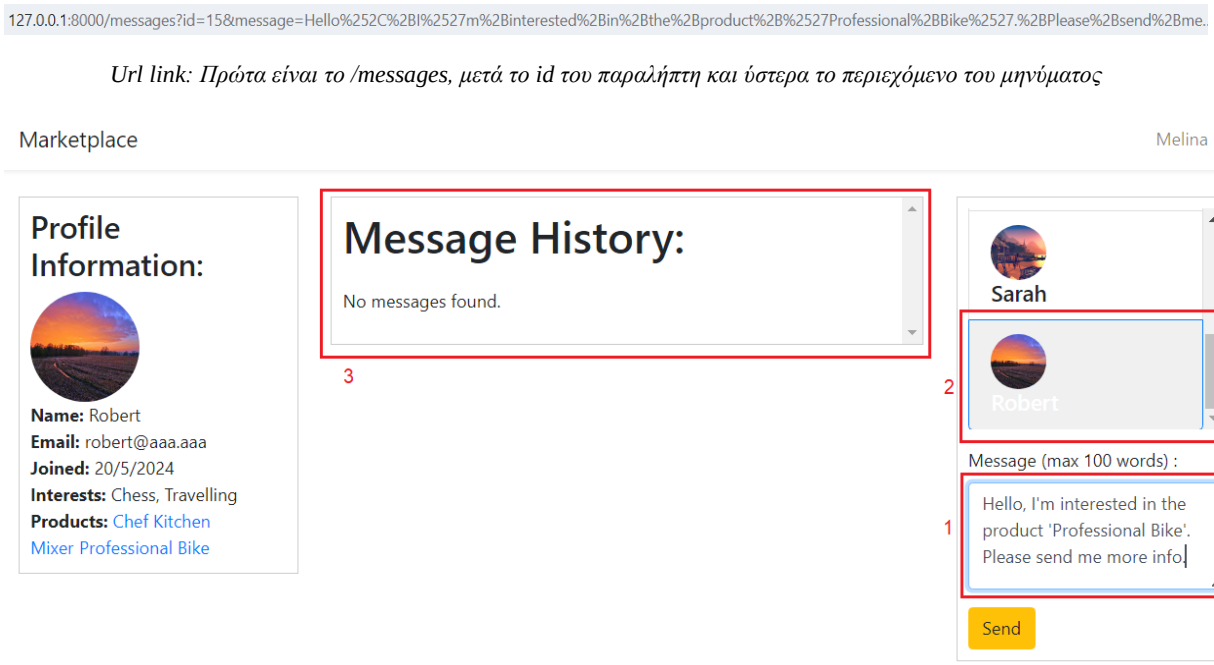
Βήμα 4: Αυτόματη εμφάνιση πληροφοριών (Αριστερά από το Message history)



5.10 Διαδικασία αποστολής μηνύματος στην λεπτομερή περιγραφή του προϊόντος

Η διαδικασία αποστολής μηνύματος στην λεπτομερή περιγραφή του προϊόντος υλοποιείται μέσω ενός δυναμικού συνδέσμου που δημιουργείται με την βοήθεια του Blade template engine. Συγκεκριμένα, αν το προϊόν είναι διαθέσιμο, εμφανίζεται ένα κουμπί με την ένδειξη "Message User". Αυτό το κουμπί οδηγεί τον χρήστη στη σελίδα αποστολής μηνύματος με προσυμπληρωμένο κείμενο που αναφέρει το ενδιαφέρον του για το συγκεκριμένο προϊόν και ζητά επιπλέον πληροφορίες. Η διεύθυνση του συνδέσμου περιλαμβάνει το ID του ιδιοκτήτη του προϊόντος και το όνομα του προϊόντος, το οποίο κωδικοποιείται για να είναι κατάλληλο για χρήση στη διεύθυνση URL. Με αυτόν τον τρόπο, διευκολύνεται η επικοινωνία μεταξύ των χρηστών και των ιδιοκτητών των προϊόντων, κάνοντας τη διαδικασία πιο άμεση και αποδοτική.

Παρακάτω φαίνεται το αποτέλεσμα πατήματος του κουμπιού “Message User” για το προϊόν “Professional Bike” του χρήστη Robert.



Url link: Πρώτα είναι το /messages, μετά το id του παραλήπτη και ύστερα το περιεχόμενο του μηνύματος

Εικόνα 46 - Αποτελέσματα πατήματος κουμπιού message user

1. Το περιεχόμενο μηνύματος που δημιουργείται αυτόματα από το link URL

2. Ο επιλεγμένος παραλήπτης (έχει ήδη επιλεχθεί αυτόματα)

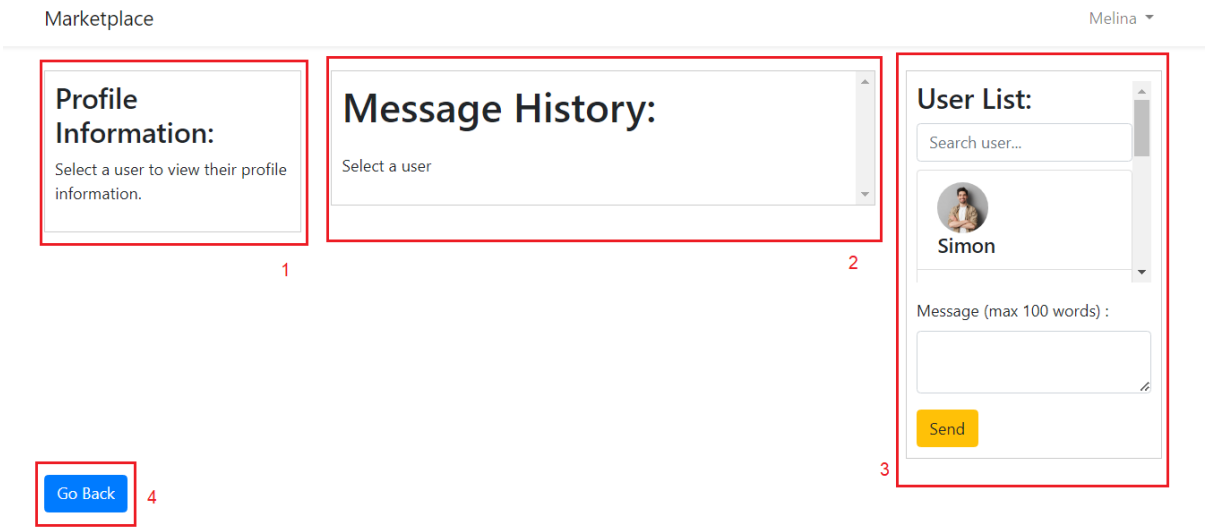
3. Η ιστορία των μηνυμάτων. Στην συγκεκριμένη περίπτωση δεν υπάρχουν παλιότερα μηνύματα για αυτό και εμφανίζει το μήνυμα “No messages found”.

Φυσικά δίνεται η δυνατότητα το μήνυμα να μπορεί να τροποποιηθεί από τον αποστολέα πριν σταλθεί στον παραλήπτη. Μόλις πατήσει το κουμπί “Send” τότε μόνο στέλνεται το μήνυμα στον παραλήπτη.

Ανάλογα το id του παραλήπτη υπάρχει συνάρτηση javascript μέσα στην κεφαλίδα script του αρχείου messages.blade.php που επιλέγει δυναμικά από την λίστα user list το id του επιλεγμένου παραλήπτη για να διευκολύνει την εύρεση του παραλήπτη και να επιλεχθεί αντί να γίνει χειροκίνητα αυτή από τον αποστολέα.

5.11 Διαδικασία επικοινωνίας μεταξύ των χρηστών

Η σελίδα Messages επιτρέπει στους χρήστες να επικοινωνούν μεταξύ τους μέσα από την πλατφόρμα. Η διάταξη της σελίδας περιλαμβάνει τρεις βασικές περιοχές: το προφίλ πληροφοριών (1) , το ιστορικό μηνυμάτων (2) και τη λίστα χρηστών (3).

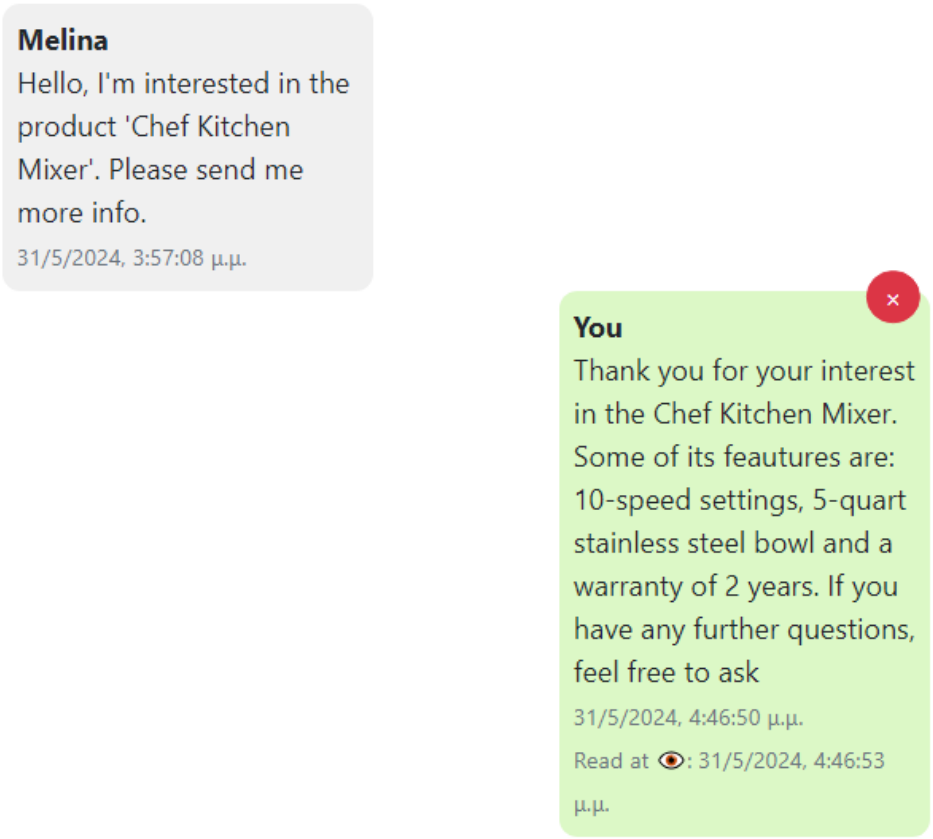


Εικόνα 47 - Σελίδα μηνυμάτων

1. **Προφίλ Πληροφοριών (Profile Information):**
   Στην περιοχή αυτή, οι χρήστες μπορούν να δουν πληροφορίες για το προφίλ του χρήστη με τον οποίο επιθυμούν να επικοινωνήσουν. Αυτή η περιοχή είναι κενή όταν δεν έχει επιλεχθεί κάποιο άτομο από την λίστα και εμφανίζεται το μήνυμα "Select a user to view their profile information". Για να εμφανιστούν οι πληροφορίες του προφίλ ενός χρήστη, ο χρήστης πρέπει να επιλέξει ένα όνομα από τη λίστα χρηστών.
2. **Ιστορικό Μηνυμάτων (Message History):**
   Εδώ εμφανίζεται το ιστορικό συνομιλιών μεταξύ του συνδεδεμένου χρήστη και του επιλεγμένου χρήστη. Όπως και με το προφίλ πληροφοριών, αρχικά αυτή η περιοχή είναι κενή και εμφανίζει το μήνυμα "Select a user". Μόλις επιλεγεί ένας χρήστης από τη λίστα χρηστών, το ιστορικό μηνυμάτων μεταξύ των δύο χρηστών θα εμφανιστεί σε αυτή την περιοχή.
3. **Λίστα Χρηστών (User List):**
   Η λίστα χρηστών περιλαμβάνει όλα τα μέλη της πλατφόρμας με τα οποία μπορεί να επικοινωνήσει ο χρήστης. Η λίστα έχει μια λειτουργία αναζήτησης (Search bar: search user) που επιτρέπει στον χρήστη να βρει γρήγορα κάποιον συγκεκριμένο χρήστη. Κάτω από τη λίστα υπάρχει ένα πεδίο text area για την αποστολή μηνύματος (με μέγιστο όριο λέξεων τις 100). Αφού ο χρήστης πληκτρολογήσει το μήνυμά του, μπορεί να πατήσει το κουμπί "Send" για να το στείλει.
4. **Επιστροφή (Go Back):**
   Υπάρχει επίσης ένα κουμπί "Go Back" που επιτρέπει στον χρήστη να επιστρέψει στο Dashboard της πλατφόρμας.

5.11.1 Δομή περιεχομένου των μηνυμάτων:

Παρακάτω βλέπουμε ένα στιγμιότυπο από την επικοινωνία μεταξύ των χρηστών για ένα διαθέσιμο προϊόν.



Εικόνα 48 - Μηνύματα μεταξύ των χρηστών

Η δομή των μηνύματος που ανταλλάσσεται μεταξύ των χρηστών της πλατφόρμας περιλαμβάνει τα εξής στοιχεία:

1. **Κουμπί διαγραφής:** Το κουμπί αυτό διαγράφει το μήνυμα που έχει στείλει ο αποστολέας.
2. **Όνομα χρήστη:** Το όνομα του χρήστη που στέλνει το μήνυμα επισημασμένο με bold γράμματα (π.χ., "Melina").
3. **Κείμενο μηνύματος:** Το περιεχόμενο του μηνύματος που αποστέλλεται (π.χ., "Hello, I'm interested in the product 'Chef Kitchen Mixer'. Please send me more info.").
4. **Ώρα και ημερομηνία αποστολής**: Η χρονική σήμανση της αποστολής του μηνύματος (π.χ., "31/5/2024, 3:57:08 μ.μ.").
5. **Ένδειξη ανάγνωσης (αν έχει διαβαστεί από τον παραλήπτη)**: Σήμανση ότι το μήνυμα έχει διαβαστεί και η χρονική στιγμή ανάγνωσης (π.χ., "Read at: 31/5/2024, 4:46:53 μ.μ.").

Όλα τα μηνύματα καταχωρούνται στον πίνακα messages όπως φαίνεται παρακάτω:

id	sender_id	receiver_id	message_content	created_at	updated_at	is_read
783	16	15	Hello, I'm interested in the product 'Chef Kitchen...	2024-05-31 12:57:08	2024-05-31 13:30:33	1
785	15	16	Thank you for your interest in the Chef Kitchen Mi...	2024-05-31 13:46:50	2024-05-31 13:46:53	1

Εικόνα 49 - Πίνακας messages

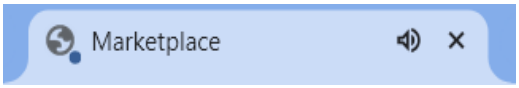
Η συγκεκριμένη δομή διασφαλίζει ότι τα μηνύματα είναι σαφή και περιέχουν όλες τις απαραίτητες πληροφορίες, όπως ποιος έστειλε το μήνυμα, το περιεχόμενο του μηνύματος, καθώς και πότε στάλθηκε. Επιπλέον, η δυνατότητα διαγραφής μηνυμάτων (για κάθε συνδεδεμένο χρήστη ο οποίος είναι ο αποστολέας των μηνυμάτων) παρέχει ευελιξία στους χρήστες να διαχειρίζονται το ιστορικό των συνομιλιών τους.

5.12 Ειδοποίηση για καινούρια μηνύματα

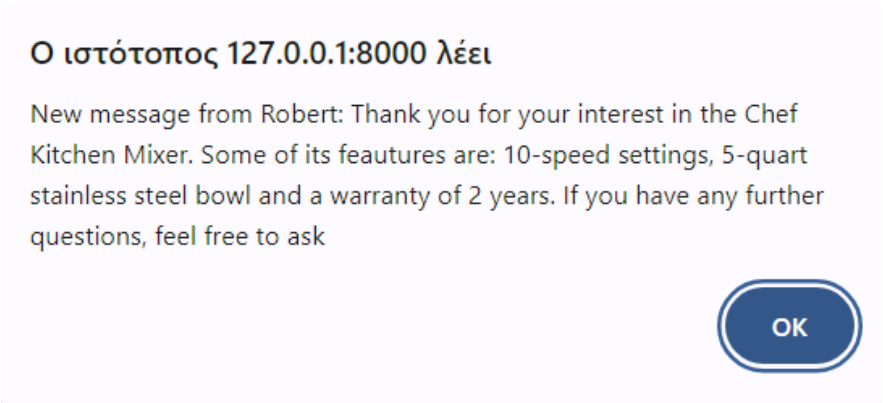
Οι ειδοποιήσεις των καινούριων μηνυμάτων είναι συγχρονισμένες, δηλαδή δουλεύουν σε πραγματικό χρόνο (με την λειτουργία της υπηρεσίας pusher). Κάθε καινούριο μήνυμα αντιστοιχεί και σε μια ξεχωριστή ειδοποίηση. Η τύπος της ειδοποίησης είναι μία ειδοποίηση alert συνοδευόμενη από ήχο, δηλαδή εικονική και ηχητική ειδοποίηση μαζί.

Σε οποιαδήποτε σελίδα βρίσκεται ο χρήστης (εκτός από αυτή των messages) ο χρήστης όταν είναι συνδεδεμένος στην πλατφόρμα και λαμβάνει ένα καινούριο μήνυμα, λαμβάνει την εξής ολική ειδοποίηση:

1. Ηχητική ειδοποίηση

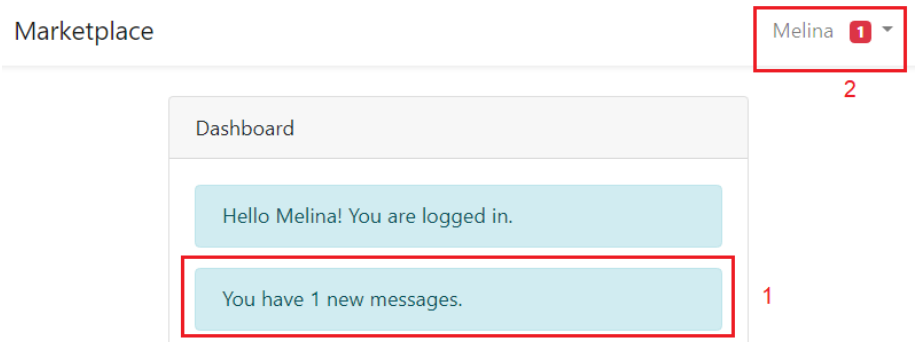


2. Εικονική (Alert) ειδοποίηση



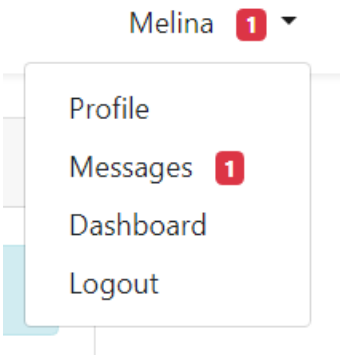
Ο τύπος της εικονικής ειδοποίησης είναι ο εξής: New message from «το όνομα του αποστολέα»: «περιεχόμενο κειμένου μηνύματος».

Μέχρι να ανοιχτεί το μήνυμα από τον παραλήπτη (τιμή is_read από 0 σε 1) εμφανίζονται οι ενδείξεις ειδοποιήσεων στο dashboard (1) και στο menu (2) όπου παραμένουν εκεί μέχρι ο χρήστης ανοίξει την συνομιλία και διαβάσει το μήνυμα με τον συγκεκριμένο χρήστη στην σελίδα messages.



Εικόνα 50 - Ειδοποιήσεις μηνυμάτων

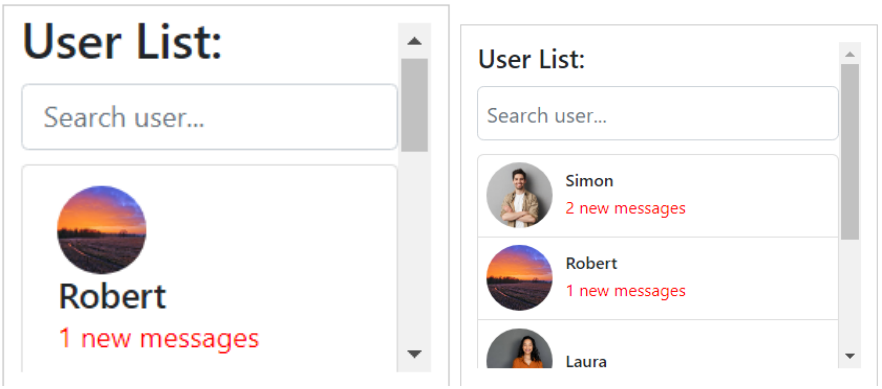
Οι ειδοποιήσεις στο menu (κόκκινα badges) μένουν σε όλες τις σελίδες της εφαρμογής μέχρι ο χρήστης ανοίξει το καινούριο μήνυμα και το διαβάσει. Η εμφάνιση του αριθμού των μηνυμάτων αφορά τα μηνύματα που έχει λάβει συνολικά (δηλαδή αν έχει λάβει μηνύματα από 2 ξεχωριστούς χρήστες, για παράδειγμα 2 μηνύματα από τον χρήστη Robert και 3 μηνύματα από τον χρήστη Laura τότε ο συνολικός αριθμός που θα εμφανιστεί στις ειδοποιήσεις θα είναι 5)



Εικόνα 51 - Ειδοποιήσεις μηνυμάτων στο menu

Όταν γίνεται ανακατεύθυνση στην σελίδα των μηνυμάτων ο χρήστης βλέπει πόσα μηνύματα έχει στείλει ο συγκεκριμένος αποστολέας (με κόκκινα γράμματα κάτω από το όνομα). Στην περίπτωση που είναι πολλοί οι αποστολείς, κάτω από κάθε αποστολέα αναγράφεται πόσα μηνύματα έχουν στείλει. Επίσης το sorting μέσα στην λίστα με τα μη ανοιγμένα μηνύματα γίνεται ανάλογα ποιος έχει στείλει τελευταίος μήνυμα στον συγκεκριμένο παραλήπτη.

Παρακάτω φαίνονται παραδείγματα μηνυμάτων που δεν έχουν διαβαστεί στη λίστα συνομιλιών:



6. Κεφάλαιο 6ο: Συμπεράσματα – Μελλοντικές επεκτάσεις

Μετά την ολοκλήρωση του έργου μου και μίας αξιολόγησης των αποτελεσμάτων, γίνεται μια αναφορά σε συμπεράσματα και μελλοντικές επεκτάσεις που μπορούν να γίνουν για την βελτίωση και την επέκταση αυτής της εφαρμογής.

6.1 Συμπεράσματα και εμπειρίες από χρησιμοποιηθέντα εργαλεία

Κατά τη διάρκεια της ανάπτυξης της εφαρμογής αποκόμισα πολλές και σημαντικές γνώσεις και εξοικοιώθηκα περισσότερο με την ανάπτυξη και την σχεδίαση διαδικτυακών εφαρμογών. Κάθε εργαλείο που χρησιμοποίησα συνέφερε σημαντικά στην υλοποίηση της εφαρμογής. Ιδιαίτερα χρήσιμο ήταν το διαδικτυακό documentation της Laravel το οποίο περιείχε αναλυτικές πληροφορίες και οδηγό για όποια λειτουργία ήθελα να υλοποιήσω. Το Laravel Framework ουσιαστικά ήταν η καρδιά του backend της εφαρμογής. Η ευκολία με την οποία μπορούσε να διαχειριστεί τις κοινές λειτουργίες ενός web application, όπως η διαχείριση των χρηστών, η επεξεργασία των δεδομένων και η επικοινωνία με τη βάση δεδομένων, έκανε την ανάπτυξη πιο αποτελεσματική. Μετά την εγκατάσταση του framework του Laravel εγκατέστησα και το εργαλείο Vite το οποίο αποτέλεσε σημαντικό εργαλείο για το front-end της εφαρμογής, μαζί με το Bootstrap. Το Vite μαζί με το Bootstrap παρέχουν ένα απλό και αισθητικό user interface, και σημαντικά πλεονεκτήματα τους είναι ότι είναι σταθερά και γρήγορα. Σημαντικό να αναφερθεί είναι πως κομμάτια όπως το authentication system και ο χειρισμός των κωδικών ήταν ήδη έτοιμα προς χρήση όταν εγκαταστάθηκε το framework και έγινε η σύνδεση με την βάση δεδομένων.

Η επιλογή του server Apache για τη φιλοξενία της εφαρμογής αποτέλεσε μια αξιόπιστη και ευέλικτη λύση για την πλατφόρμα. Το εργαλείο XAMPP διευκόλυνε την εγκατάσταση και τη διαχείριση του Apache server και της MySQL βάσης δεδομένων.

Η MySQL αποτέλεσε το σύστημα διαχείρισης βάσης δεδομένων για την αποθήκευση και την ανάκτηση των δεδομένων της εφαρμογής. Η απόδοσή της, η σταθερότητα και η ευκολία στη διαχείριση των δεδομένων ήταν καθοριστικής σημασίας. Κατά την εγκατάσταση του framework Laravel δημιουργήθηκαν αυτόματα κάποιοι πίνακες όπως «migrations», «password_resets», «failed_jobs» κτλ. στην βάση δεδομένων “Marketplace” για την διαχείριση των μεταναστεύσεων και άλλων λειτουργιών που βρίσκονται στο παρασκήνιο ενώ η δημιουργία των υπόλοιπων 4 πινάκων για την αποθήκευση διαφόρων δεδομένων, όπως προφίλ χρηστών, προϊόντα και μηνύματα, έγινε εύκολα μέσω της MySQL.

Ωστόσο, ένα μικρό μειονέκτημα που αξίζει να αναφερθεί είναι πως όταν χρειάστηκε να γίνει προσθήκη πεδίων σε κάποιο πίνακα, έπρεπε να γίνουν τα λεγόμενα migrations (μεταναστεύσεις) τα οποία είναι κάποιες εντολές που πρέπει να γράφονται και να εκτελούνται στο cmd terminal του αντίστοιχου IDE για να ανανεωθούν οι δομές των πινάκων. Αυτή η διαδικασία μπορεί να γίνει κουραστική και χρονοβόρα κάποιες φορές για έναν προγραμματιστή. Φυσικά υπάρχουν τρόποι που μπορεί να αποφευχθεί αυτό όπως να γίνει αλλαγή της βάσης δεδομένων κτλ.

Το Visual Studio Code ήταν το κύριο εργαλείο ανάπτυξης και προγραμματισμού. Η ευκολία χρήσης του, τα αμέτρητα εργαλεία που προσφέρει και το debugging του το κατέστησαν απαραίτητο εργαλείο για την ανάπτυξη της εφαρμογής.

Η χρήση αυτών των εργαλείων και των τεχνολογιών συνέβαλαν στη δημιουργία μιας πλήρως λειτουργικής και αποδοτικής εφαρμογής και οι εμπειρίες από την χρήση τους ανέδειξαν την

σημασία της σωστής επιλογής των εργαλείων ανάπτυξης, καθώς και την σπουδαιότητα της συνεχούς μάθησης και προσαρμογής στις τεχνολογικές εξελίξεις.

6.2 Μελλοντικές επεκτάσεις

Πάντα υπάρχει η δυνατότητα μια εφαρμογή να γίνει ακόμα καλύτερη και αξίζουν να αναφερθούν οι επιπλέον λειτουργίες και βελτιστοποιήσεις που θα μπορούσαν να γίνουν στην συγκεκριμένη. Αναφέρω δύο κατηγορίες μελλοντικών επεκτάσεων: τις λειτουργικές, δηλαδή αυτές που θα μπορούσαν να προστεθούν ως επιπλέον λειτουργίες και οι καλυτερεύσεις, δηλαδή μια λειτουργία που υπάρχει ήδη και θα μπορούσε να γίνει καλύτερη.

Σίγουρα μία σημαντική λειτουργική υπηρεσία που δεν έχει υλοποιηθεί και θα ήταν χρήσιμη να υπάρχει, θα ήταν η υλοποίηση ενός πλήρους admin panel. Δηλαδή ένας διαχειριστής να έχει τον πλήρη έλεγχο των χρηστών και τον προϊόντων, να προσθέτει χρήστες, να τους αναζητεί, να ελέγχει τα δεδομένα των προϊόντων που εισάγονται και να μπορεί να τα τροποποιεί ή να τα διαγράφει. Θα μπορούσε επίσης να τροποποιεί και τα πεδία (δομή) των πινάκων, για παράδειγμα να προσθέτει μία καινούρια κατηγορία προϊόντων ή να αλλάζει το όριο των mb στην εισαγωγή μίας φωτογραφίας κτλ. Ακόμη μία από τις λειτουργικές μελλοντικές επεκτάσεις θα μπορούσε να είναι η προσθήκη και η αφαίρεση φίλων, καθώς και το μπλοκάρισμα των χρηστών. Η δημιουργία ενός ασφαλούς και προσωπικού περιβάλλοντος χρήσης επιτρέπει στους χρήστες να διαχειρίζονται τις επαφές τους και να αποφεύγουν ανεπιθύμητες αλληλεπιδράσεις. Επιπλέον η προσθήκη μιας φόρμας παραπόνων θα δώσει στους χρήστες τη δυνατότητα να αναφέρουν προβλήματα που αντιμετωπίζουν στην πλατφόρμα. Αυτό θα βελτιώσει την επικοινωνία μεταξύ των χρηστών και της διαχειριστικής ομάδας. Ακόμη η δυνατότητα αναγνώρισης online-offline συνδεδεμένων χρηστών θα βοηθήσει στη βελτίωση της αλληλεπίδρασης μεταξύ των μελών του κοινωνικού δικτύου, επιτρέποντας στους χρήστες να βλέπουν ποιοι είναι διαθέσιμοι για συνομιλία σε πραγματικό χρόνο. Επιπρόσθετα η ανταλλαγή αρχείων, φωτογραφιών και ηχητικών στα μηνύματα θα προσφέρει μια πιο πλούσια εμπειρία επικοινωνίας, διευκολύνοντας τη μεταφορά των πληροφοριών και την προσωπική έκφραση. Τέλος η προσθήκη περισσοτέρων φωτογραφιών για ένα προϊόν θα επιτρέψει στους πωλητές να παρουσιάζουν τα προϊόντα τους με περισσότερες λεπτομέρειες, αυξάνοντας την πιθανότητα πώλησης.

Μερικές από τις βελτιστοποιήσεις που μπορούν να γίνουν είναι ο καλύτερος έλεγχος εισαγωγής δεδομένων. Μέχρι τώρα γίνεται ένας έλεγχος δεδομένων για την εισαγωγή του πεδίου email για έναν user (θα πρέπει να περιέχει το σύμβολο «@»), για το μέγεθος ενός μηνύματος να μην υπερβαίνει πολλούς χαρακτήρες και να έχουν συμπληρωθεί τουλάχιστον 3 πεδία για το ανέβασμα ενός προϊόντος. Επιπλέον έλεγχοι που μπορούν να συμπεριληφθούν είναι να γίνεται η επιβεβαίωση του email ενός χρήστη δηλαδή να επαληθεύεται η ταυτότητα του. Η επιβεβαίωση ταυτότητας ενός χρήστη ενισχύει την ασφάλεια στην πλατφόρμα, μειώνοντας την πιθανότητα απάτης και κακής χρήσης. Επίσης τα μηνύματα να μην περιέχουν άσχημους χαρακτηρισμούς ή περίεργες συμβολοσειρές και φυσικά να γίνεται ένας έλεγχος των προϊόντων πριν ανεβάσει κάποιος χρήστης ένα προϊόν για να αποφεύγονται περιεχόμενα που παραβιάζουν τους ορούς χρήσης ή είναι επικίνδυνα. Επιπλέον περισσότερα φίλτρα στην αναζήτηση προϊόντων θα βοηθήσουν τους χρήστες να βρίσκουν ακριβώς αυτό που ψάχνουν, ελαχιστοποιώντας τον χρόνο αναζήτησης. Τέλος ο καλύτερος σχεδιασμός και το UI στα μηνύματα θα βελτιώσει την εμπειρία χρήσης κάνοντας την επικοινωνία πιο ευχάριστη, κομψή και εύκολη στη χρήση.

Βιβλιογραφία-Αναφορές-Διαδικτυακές Πηγές

1. https://juphy.com/blog/how-to-use-facebook-marketplace-for-business
2. https://moneytips.debt.com/product-reviews/what-is-offerup-how-does-work/
3. https://en.wikipedia.org/wiki/Letgo
4. https://www.sorted.digital/blog/instagram-shop-easy-explanation/
5. https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)
6. https://cleancommit.io/blog/what-is-vite/
7. https://laravel.com/docs/11.x/blade
8. https://www.metacubic.com/unleashing-the-power-of-laravels-blade-templates-2/
9. https://sass-lang.com/documentation/
10. https://www.linkedin.com/pulse/benefits-using-javascript-basit-ch
11. https://developer.mozilla.org/en-US/docs/Web/HTML
12. https://www.w3schools.com/html/html_intro.asp
13. https://developer.mozilla.org/en-US/docs/Web/CSS
14. https://www.programiz.com/css/syntax
15. https://en.wikipedia.org/wiki/Laravel
16. https://www.php.net/
17. https://www.apachefriends.org/
18. https://www.geeksforgeeks.org/how-to-install-xampp-on-windows/
19. https://laravel.com/docs/4.2/quick#installation
20. https://getcomposer.org/doc/00-intro.md
21. https://laravel.com/docs/11.x/artisan
22. https://www.npmjs.com/
23. https://www.geeksforgeeks.org/introduction-to-laravel-and-mvc-framework/
24. https://el.wikipedia.org/wiki/Model-view-controller
25. https://help.mailtrap.io/article/12-getting-started-guide
26. https://www.crazyegg.com/blog/mailtrap-review/
27. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON
28. https://developer.mozilla.org/en-US/docs/Web/HTTP
29. https://simple.wikipedia.org/wiki/Application_programming_interface
30. https://www.altexsoft.com/blog/rest-api-design/
31. https://tech-norm.com/software-development/

32. https://konghq.com/blog/learning-center/what-is-restful-api
33. https://apexcode.dev/blog/building-api-endpoints-with-laravel
34. https://www.geeksforgeeks.org/javascript-fetch-method/
35. https://www.wallarm.com/what/a-simple-explanation-of-what-a-websocket-is
36. https://blog.back4app.com/what-is-pusher/
37. https://code.visualstudio.com/
38. https://learn.microsoft.com/en-us/aspnet/core/tutorials/first-web-api?view=aspnetcore-8.0&tabs=visual-studio
39. https://www.researchgate.net/figure/The-LiveVis-architecture-consists-of-a-Laravel-application-After-sending-the-URL-request_fig1_320100676
40. https://www.dataversity.net/types-of-databases-and-their-uses/
41. https://hzhou.scholar.harvard.edu/blog/mysql-vs-mongodb