ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ανάπτυξη διαδικτυακής εφαρμογής για ανταλλαγή βιβλίων

ΚΑΛΑΪΤΖΙΔΗΣ ΤΡΙΑΝΤΆΦΥΛΛΟΣ .

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

ΤΖΙΑΛΛΑΣ ΓΡΗΓΟΡΙΟΣ
ΚΑΘΗΓΗΤΗΣ

……..Ονοματεπώνυμο Υπεύθυνου……..
……Βαθμίδα……

Λαμία …… έτος ……

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ανάπτυξη διαδικτυακής εφαρμογής για ανταλλαγή βιβλίων

ΤΡΙΑΝΤΑΦΥΛΛΟΣ ΚΑΛΑΪΤΖΙΔΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

ΤΖΙΑΛΛΑΣ ΓΡΗΓΟΡΙΟΣ
ΚΑΘΗΓΗΤΗΣ

Λαμία ..... 2024.....

UNIVERSITY OF
THESSALY

SCHOOL OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE & TELECOMMUNICATIONS

Development of a Book Trading Web Application

TRIANTAFYLLOS KALAITZIDIS

FINAL THESIS

ADVISOR

TZIALLAS GRIGORIOS
PROFESSOR

Lamia ..... 2024.....

«Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

*1. Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

*2. Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: 26/10/2024

Ο – Η Δηλ.

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση
του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων
σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.»

ΠΕΡΙΛΗΨΗ

Σκοπός αυτής της εργασίας είναι η δημιουργία και η παρουσίαση μιας διαδικτυακής εφαρμογής ανταλλαγής βιβλίων. Θα γίνει παρουσίαση ανάλογων εφαρμογών καθώς και μελέτη πάνω στις διαθέσιμες τεχνολογίες ανάπτυξης διαδικτυακών εφαρμογών (Full-Stack Web Development). Στη συνέχεια θα υλοποιηθεί μια εφαρμογή ανταλλαγής βιβλίων βασισμένη στο Framework της React JS η οποία θα αποθηκεύει τα δεδομένα της σε μια βάση δεδομένων μέσω της PostgreSQL. Τέλος, θα παρουσιάσουμε την λειτουργικότητα της εφαρμογής παρέχοντας οδηγίες χρήσης όπου αυτές είναι απαραίτητες.

ABSTRACT

The purpose of this Thesis is the creation and presentation of a Book Trading web application. Some already existing applications that serve the same purpose will be presented and technologies used in Full-Stack Web development will be studied. Furthermore, a book trading web application will be implemented using the React JS framework which will save its data in a database through PostgreSQL. Finally, we will present the functionality of the application providing user instructions where these are deemed necessary.

Table of Contents

1. ΕΙΣΑΓΩΓΗ

1.1 ΣΤΟΧΟΙ ΤΗΣ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο βασικότερος στόχος της πτυχιακής εργασίας είναι η μελέτη και η παρουσίαση τεχνολογιών διαδικτύου καθώς και η υλοποίηση μιας εφαρμογής που αποσκοπεί στην ψηφιοποίηση της διαδικασίας ανταλλαγής βιβλίων. Θα παρουσιάσω ανάλογες εφαρμογές που υπάρχουν ήδη στην αγορά. Έπειτα, θα δώσω μια εκτενείς παρουσίαση της θεμελιώδους αρχιτεκτονικής μιας διαδικτυακής εφαρμογής καθώς και μια εισαγωγή στα Frameworks της βασικής γλώσσας που χρησιμοποιείται στην ανάπτυξη διαδικτυακών εφαρμογών (JavaScript). Αργότερα, θα παρουσιαστεί η εφαρμογή η οποία υλοποιήθηκε με ανάλογες εξηγήσεις γύρω από τον κώδικα και τα μέρη της διεπαφής χρήστη (User Iterface). Τέλος θα δοθούν οδηγίες για την διαχείριση χρηστών καθώς και την χρήση της εφαρμογής.

2. ΠΡΟΥΠΑΡΧΟΥΣΕΣ ΕΦΑΡΜΟΓΕΣ

Στο κόσμο της βιβλιοφιλίας, η ανταλλαγή βιβλίων είναι από τους πιο διαδεδομένους τρόπους ανακάλυψης καινούργιων βιβλίων και ανάπτυξης γνώσεων

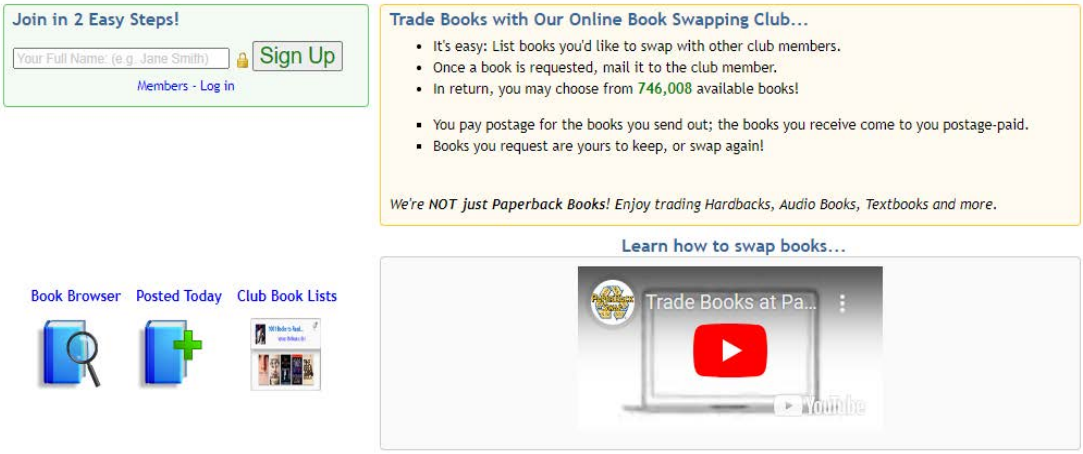
των αναγνωστών. Φυσικό επακόλουθο είναι λοιπόν, η ψηφιοποίηση αυτής της διαδικασίας να προσφέρει σημαντικά οφέλη στους αναγνώστες. Οι εφαρμογές οι οποίες θα παρουσιάσω προσφέρουν διαφορετικές προσεγγίσεις πάνω στην ίδια ιδέα προωθώντας την έννοια της ανταλλαγής πληροφορίας και της επαναχρησιμοποίησης υλικού.

2.1 PAPERBACK SWAP

Το PaperBack Swap είναι μία ιστοσελίδα η οποία δημιουργήθηκε το 2004 από τους Richard Pickering και Robert Swarthout. Η ιδέα πίσω από το PaperBack Swap είναι η χρήση του διαδικτύου για την εδραίωση ανταλλαγής βιβλίων μεταξύ μελών στο πλαίσιο των Ηνωμένων Πολιτειών της Αμερικής χρησιμοποιώντας ένα σύστημα πόντων.

Οι πόντοι στο PaperBack Swap διευκολύνουν την διαδικασία της ανταλλαγής καθώς εξαλείφουν την ανάγκη επικοινωνίας των μελών που συμμετέχουν στην ανταλλαγή. Ένας χρήστης μπορεί να λάβει πόντους στέλνοντας και ανταλλάσσοντας βιβλία η μπορεί ακόμη και να τους αγοράσει.

Τα βιβλία που μπορούν να ανταλλαχθούν από την ιστοσελίδα είναι είτε χαρτόδετα, σκληρόδετα η ηχητικά βιβλία.



2.2 BOOKMOOCH

Το BookMooch είναι μία κερδοσκοπική, διεθνής πλατφόρμα ανταλλαγής βιβλίων που δημιουργήθηκε το 2006 από τον John Buckham. Η εφαρμογή θεωρείται αρκετά επιτυχημένη καθώς μέχρι το 2008 η εφαρμογή έφτασε τα 74000 μέλη παγκοσμίως και περίπου 2000 ανταλλαγές βιβλίων την ημέρα.

Όπως και το PaperBackSwap έτσι και το BookMooch λειτουργεί με σύστημα πόντων. Στο BookMooch σχεδόν όλα τα βιβλία, κοστίζουν τον ίδιο αριθμό πόντων

και οι χρήστες κερδίζουν πόντους στέλνοντας βιβλία η παρέχοντας κριτικές όταν λαμβάνουν βιβλία.

Οι χρήστες έχουν τη δυνατότητα να επεξεργαστούν και να εξατομικεύσουν τις προσωπικές τους σελίδες με τη βοήθεια ενός εύχρηστου συστήματος widgets. Κάθε προσωπική σελίδα είναι σχεδιασμένη να περιλαμβάνει όχι μόνο βασικά βιογραφικά στοιχεία του χρήστη, αλλά και μια λεπτομερή περιγραφή της κατάστασης των βιβλίων που διαθέτει. Αυτή η δυνατότητα εξατομίκευσης επιτρέπει στους χρήστες να δημιουργούν έναν ψηφιακό χώρο που αντικατοπτρίζει τις προσωπικές τους προτιμήσεις και ανάγκες, καθιστώντας την εμπειρία χρήσης πιο ευχάριστη και ουσιαστική.

Η εφαρμογή διαθέτει πολλές καινοτόμες τεχνολογίες που συμβάλλουν στην ευχρηστία της, καθιστώντας τη διαδικασία διαχείρισης βιβλίων πιο αποδοτική και πρακτική. Ένα από τα χαρακτηριστικά της είναι το αυτοματοποιημένο σύστημα e-mail, το οποίο διευκολύνει την επικοινωνία και την ενημέρωση των χρηστών σχετικά με τις τελευταίες εξελίξεις στην πλατφόρμα. Επιπλέον, η δυνατότητα εισαγωγής συνδέσμων από την Amazon για τα βιβλία προσφέρει μια πρακτική λύση για την ευκολότερη και γρηγορότερη προσθήκη βιβλίων στην πλατφόρμα, ενισχύοντας έτσι την εμπειρία του χρήστη.

2.3 READITSWAPIT

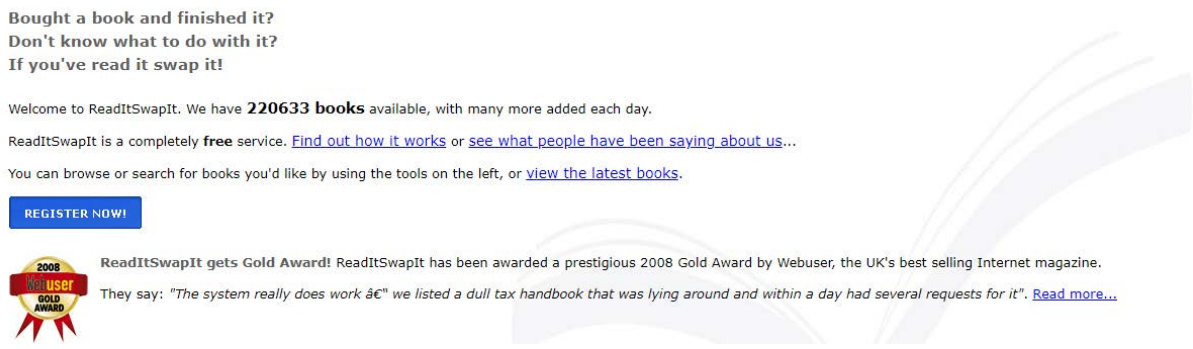
Το ReadItSwapIt είναι μια καινοτόμα πλατφόρμα ανταλλαγής βιβλίων που επιτρέπει σε χρήστες να ανταλλασουν βιβλία εύκολα και γρήγορα. Με μία φιλική προς τον χρήστη διεπαφή, η πλατφόρμα ReadItSwapIt επιτρέπει την εύκολη αναζήτηση βιβλίων μέσα από μια πλούσια γκάμα επιλογών χωρίς το κόστος αγοράς καινούργιων βιβλίων. Οι χρήστες μπορούν επίσης να προσφέρουν τα δικά τους

βιβλία ενθαρρύνοντας έτσι την ανταλλαγή και την επαναχρησιμοποίηση τον ίδιων βιβλίων.

Κάθε χρήστης μπορεί να δημιουργήσει ένα προσωπικό προφίλ στο οποίο να παραθέτει τα βιβλία που επιθυμεί να ανταλλάξει καθώς και τα βιβλία τα οποία αναζητά. Επιπρόσθετα, ένας χρήστης μπορεί να προσθέσει κριτικές για διάφορα βιβλία επιτρέποντας έτσι την καλύτερη κατανόηση της αισθητικής και των ενδιαφερόντων του κάθε χρήστη. Αυτή η κοινωνική διάσταση της πλατφόρμας ReadItSwapIt, είναι ένα από τα βασικά πλεονεκτήματα της καθώς προωθεί την δημιουργία κοινότητας και την αλληλεπίδραση ανάμεσα στους αναγνώστες.

Όπως και οι ανταγωνιστές του έτσι και το ReadItSwapIt προσφέρει διάφορα εργαλεία τα οποία το καθιστούν πιο εύχρηστο στη διαδικασία της ανταλλαγής. Η αυτοματοποιημένη υπηρεσία ειδοποιήσεων ενημερώνει τους χρήστες για προσφορές η για ευκαιρίες ανταλλαγής στα βιβλία τα οποία τους ενδιαφέρουν διασφαλίζοντας έτσι ότι οι χρήστες μένουν ενημερωμένοι και ενεργοί στην πλατφόρμα.

Η πλατφόρμα επίσης επιτρέπει την κριτική μεταξύ των χρηστών. Μια λειτουργικότητα η οποία είναι πολύ χρήσιμη καθώς αυξάνει την εμπιστοσύνη και την ασφάλεια μεταξύ των ατόμων που χρησιμοποιούν η σκοπεύουν να χρησιμοποιήσουν την πλατφόρμα.



2.4 SWAPNOW

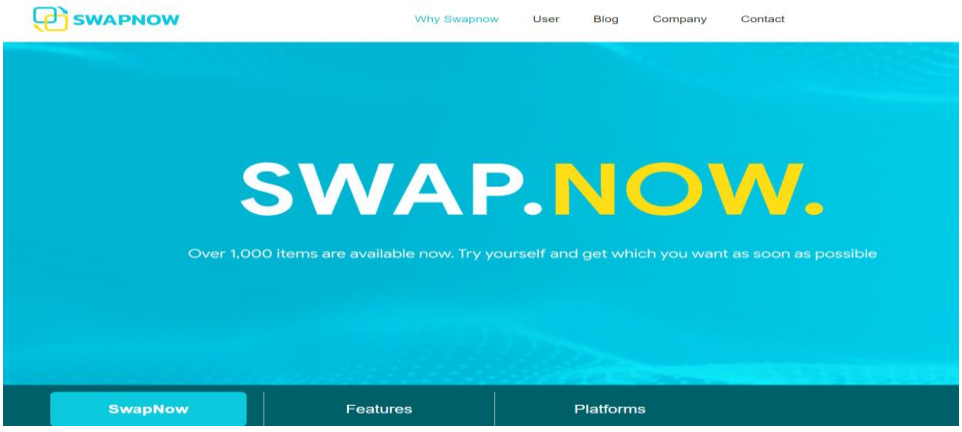
Η εφαρμογή SwapNow αν και δεν προσφέρεται μόνο για ανταλλαγή βιβλίων, είναι μια πολύ χρήσιμη πλατφόρμα για την διαδικασία της ανταλλαγής λόγω της

έμφασης που δίνει στο κοινωνικό χαρακτήρα και την δικτύωση μεταξύ χρηστών.

Η SwapNow είναι μια καινοτόμα εφαρμογή που επιτρέπει στους χρήστες να ανταλλάξουν διάφορα αντικείμενα και υπηρεσίες. Με τη βοήθεια μιας εύκολα κατανοήσιμης διεπαφής, τα μέλη της εφαρμογής μπορούν να δημιουργήσουν καταχωρήσεις για τα αντικείμενα τα οποία επιθυμούν να ανταλλάξουν καθώς και αιτήσεις για τα αντικείμενα τα οποία επιθυμούν να λάβουν. Κάτι το οποίο κάνει τη διαδικασία της ανταλλαγής όχι μόνο εύκολη αλλά και διασκεδαστική καθώς δίνεται η ευκαιρία στους χρήστες να λάβουν αντικείμενα τα οποία δεν είχαν σκεφτεί ότι χρειάζονται η θέλουν.

Κάτι το οποίο διαφοροποιεί την εφαρμογή SwapNow από τον ανταγωνισμό είναι η δυνατότητα δημιουργίας “πακέτων ανταλλαγής”. Τα “πακέτα ανταλλαγής” είναι συνδυασμοί αντικειμένων οι οποίοι είναι φτιαγμένοι για να ανταλλάσσονται όλοι μαζί Αυτό το χαρακτηριστικό επιτρέπει την δημιουργία ενδιαφέρον και ελκυστικών συνδυασμών και ομαδοποιήσεων αντικειμένων, δημιουργώντας έτσι ένα πιο οργανωμένο περιβάλλον ανταλλαγής. Η εφαρμογή επίσης διαθέτει φίλτρα αναζήτησης τα οποία κάνουν την εύρεση των κατάλληλων βιβλίων η αντικειμένων πολύ πιο εύκολη.

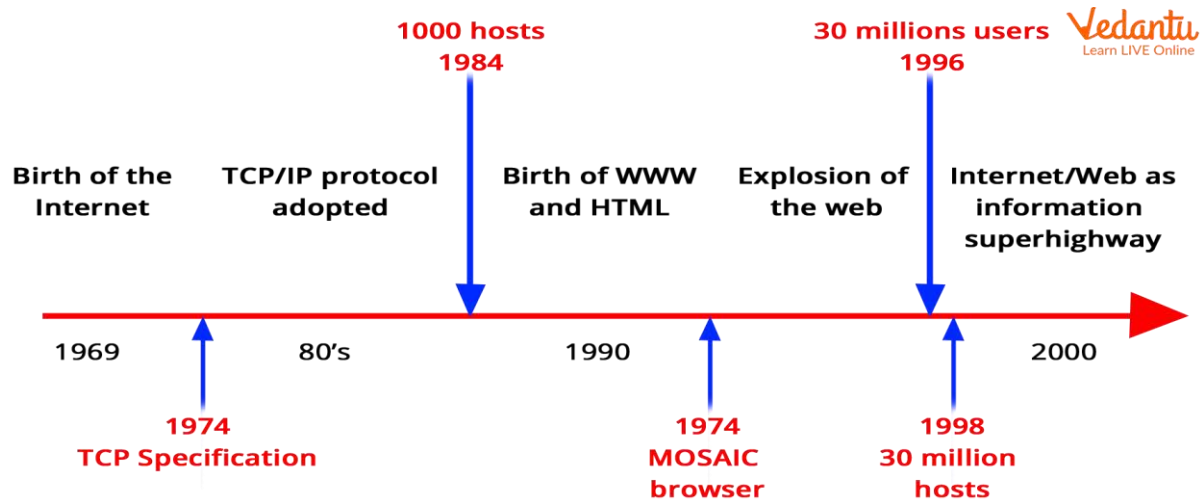
Η κοινότητα της SwapNow επεκτείνεται συνεχώς, αφού οι χρήστες έχουν την δυνατότητα να συμμετάσχουν σε εκδηλώσεις οι οποίες δημιουργούνται από την ίδια τη πλατφόρμα. Τέτοιες δράσεις δημιουργούν το αίσθημα της κοινότητας καθώς και προωθούν την ομαδικότητα και την κοινωνική δικτύωση, μετατρέπονται έτσι την πλατφόρμα σε ένα δυναμικό και ζωντανό μέρος για την ανταλλαγή ιδεών και αντικειμένων.



3. ΤΕΧΝΟΛΟΓΙΕΣ ΑΝΑΠΤΥΞΗΣ ΔΙΑΔΙΚΤΥΑΚΩΝ ΕΦΑΡΜΟΓΩΝ

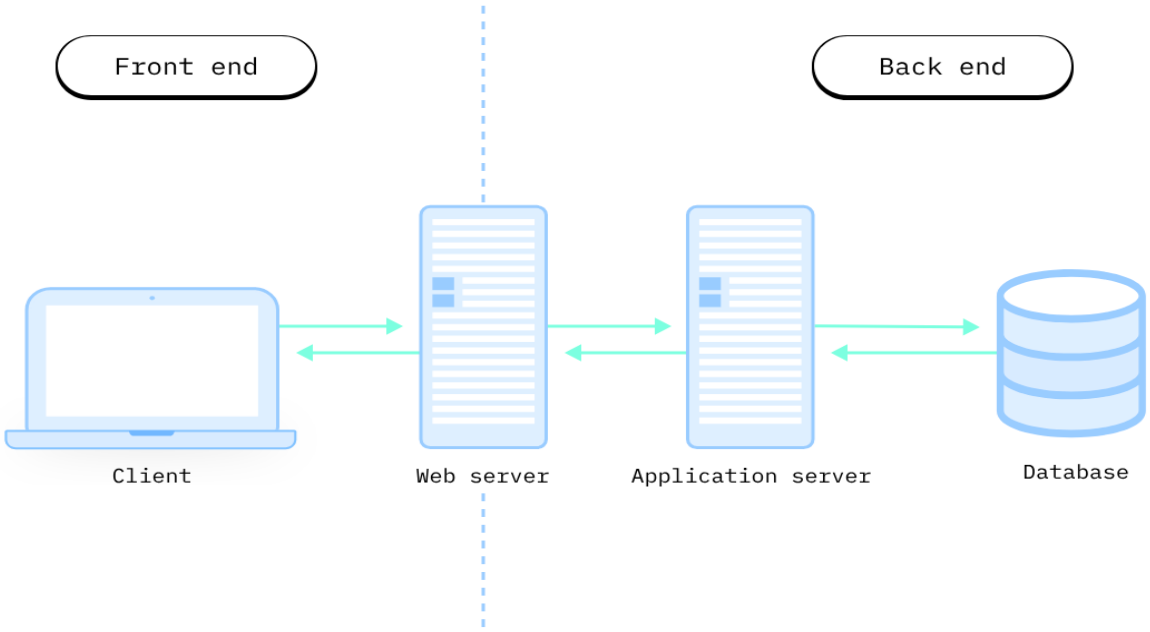
3.1 ΣΥΝΤΟΜΗ ΙΣΤΟΡΙΑ ΤΟΥ ΔΙΑΔΙΚΤΥΟΥ

- **1960:** Ο Αμερικάνικος Στρατός δημιουργεί το δίκτυο επικοινωνίας ARPANET, βασισμένο στην Ανταλλαγή Πακέτων (Packet Switching) και στη σουίτα πρωτοκόλλων TCP/IP.
- **1980:** Ο Tim Burners Lee γράφει το πρόγραμμα ENQUIRE που υλοποιεί την ιδέα των δεσμών μεταξύ κόμβων.
- **1989:** Ο Tim Burners Lee δημοσιεύει τα: "Information Management: A Proposal" και "HyperText" στο CERN.
- **1990:** Ο Tim Burners Lee έχει δημιουργήσει όσα χρειάζονται για την πρώτη έκδοση του Διαδικτύου: ένα πρόγραμμα περιήγησης, κάποιες ιστοσελίδες, τα πρωτόκολλα HTTP και HTML και έναν διακομιστή HTTP.
- **1994:** Ο Tim Burners Lee δημιουργεί το World Wide Web Consortium, έναν οργανισμό που έφερε κοντά αντιπροσώπους από διάφορες τεχνολογικές εταιρείες για να δουλέψουν μαζί για τη δημιουργία διαφόρων διαδικτυακών τεχνολογιών. [1]



3.2 ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΑΝΑΠΤΥΞΗ ΔΙΑΔΙΚΤΥΑΚΩΝ ΕΦΑΡΜΟΓΩΝ

Η ανάπτυξη διαδικτυακών εφαρμογών (Web Development) συμπεριλαμβάνει όλο το εύρος των εργασιών που είναι απαραίτητες για την δημιουργία εφαρμογών που προορίζονται για χρήση στο Internet. Μπορεί να πάρει διάφορες μορφές όπως μεταξύ άλλων, διαχείριση Βάσεων Δεδομένων, σχεδιασμό διεπαφών ιστοσελίδων και διαδικτυακό προγραμματισμό.[2] Η ανάπτυξη διαδικτυακών εφαρμογών μπορεί να ταξινομηθεί σε δύο κατηγορίες: Front-End Development και Back-End Development. Στο πλαίσιο του Back-End Development θα αναφέρω και μια τρίτη εξίσου σημαντική κατηγορία που είναι οι τεχνολογίες βάσεων δεδομένων.

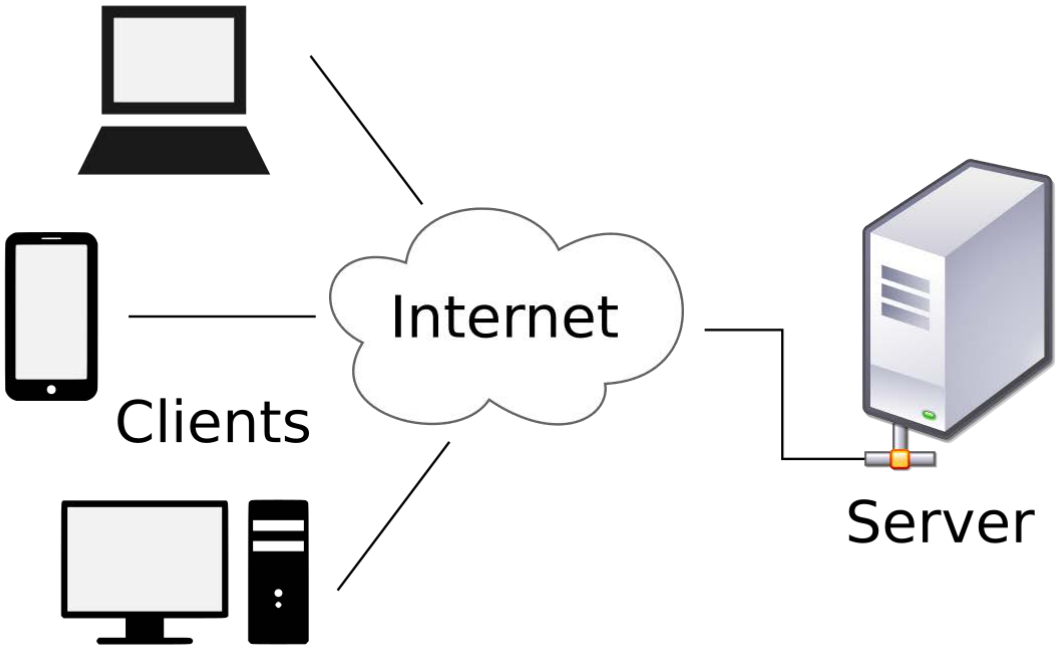


Ο διαχωρισμός στις δυο αυτές κατηγορίες βασίζεται στο μοντέλο αρχιτεκτονικής Πελάτη-Διακομιστή, γνωστό και ως Client-Server Model.

3.3 CLIENT-SERVER MODEL

Το μοντέλο Client-Server περιγράφει τη σχέση μεταξύ ενός προγράμματος του οποίου ζητάει μια υπηρεσία (Client) και ενός προγράμματος του οποίου τη παρέχει (Server). Τα αιτήματα του Client οργανώνονται και τίθενται σε προτεραιότητα μέσω ενός συστήματος δρομολόγησης, το οποίο βοηθάει τους Servers στην περίπτωση που λάβουν πολλά μηνύματα από διαφορετικούς Clients την ίδια

χρονική στιγμή. Ο Client δημιουργεί μια σύνδεση με τον Server, συνήθως μέσω Internet. [3]



3.4 ΠΡΟΤΟΚΟΛΛΟ TCP/IP

Για τις ανάγκες της επικοινωνίας των δυο προγραμμάτων, είναι απαραίτητο ένα πρωτόκολλο το οποίο θέτει τους κανόνες με τους οποίους επικοινωνούν. Το μοντέλο Client-Server χρησιμοποιεί το πρωτόκολλο TCP/IP. Το TCP/IP είναι μια σουίτα πρωτοκόλλων που βασίζονται στη σύνδεση για την ανταλλαγή και διατήρηση δεδομένων. Το οποίο σημαίνει πως το πρωτόκολλο δημιουργεί και κρατάει ενεργή μια σύνδεση μέχρι και τα δύο προγράμματα να ολοκληρώσουν την

ανταλλαγή μηνυμάτων τους. Στο μοντέλο Open Systems Interconnection (OSI) το TCP/IP καλύπτει μέρη του επιπέδου 4 και του επιπέδου 5.[4]

Το πρωτόκολλο TCP/IP παρέχει τα εξής πλεονεκτήματα:

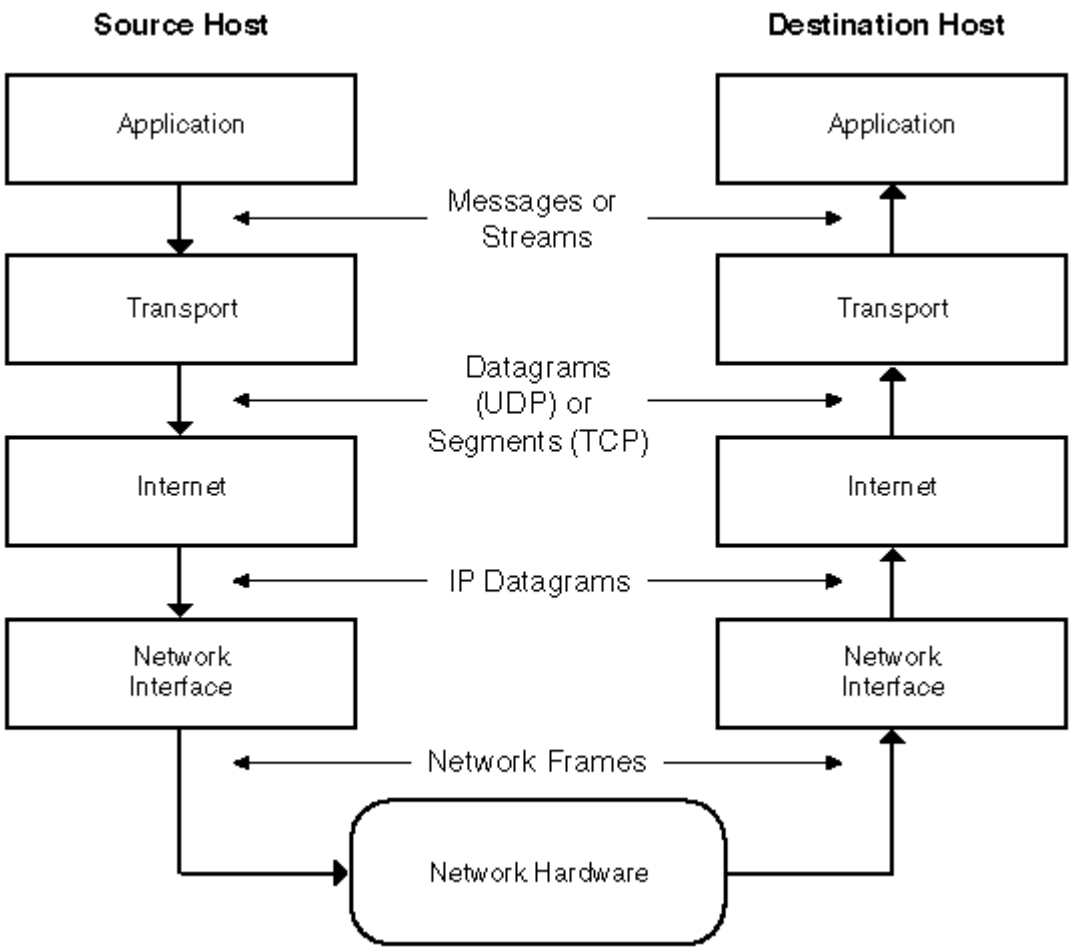
1. Διαλειτουργικότητα: Το πρωτόκολλο TCP/IP επιτρέπει διαφορετικούς τύπους υπολογιστών και δικτύων να επικοινωνούν μεταξύ τους.

2. Επεκτασιμότητα: Το πρωτόκολλο TCP/IP είναι αρκετά επεκτάσιμο, καθιστώντας το κατάλληλο για μικρά και μεγάλα δίκτυα.

3. Τυποποίηση: Το πρωτόκολλο TCP/IP είναι βασισμένο σε ανοιχτά πρωτόκολλα, βεβαιώνοντας έτσι ότι διαφορετικές εφαρμογές και προγράμματα μπορούν να επικοινωνούν μεταξύ τους χωρίς προβλήματα προσβασιμότητας.

4. Ευελιξία: Το πρωτόκολλο TCP/IP υποστηρίζει διάφορα πρωτόκολλα, τύπους δεδομένων και μεθόδους επικοινωνίας. Κάτι το οποίο το καθιστά ευέλικτο για διάφορες ανάγκες δικτύωσης.

5. Το πρωτόκολλο TCP/IP περιλαμβάνει χαρακτηριστικά ελέγχου σφαλμάτων και επαναμετάδοσης διασφαλίζοντας έτσι, ασφαλή μεταφορά δεδομένων ακόμη και σε μεγάλες αποστάσεις και διάφορες συνθήκες δικτύου. [5]



3.5 FRONT-END ΤΕΧΝΟΛΟΓΙΕΣ

Το Front-End Web Development περιγράφει τη διαδικασία ανάπτυξης Γραφικών Διεπαφών Χρηστών (Graphical User Interfaces – GUIs). Το σύνολο των στοιχείων δηλαδή, με τα οποία έρχονται σε επαφή και μπορούν να αλληλεπιδρούν οι χρήστες. [6] Οι λειτουργίες μιας Front-End εφαρμογής εκτελούνται στο πρόγραμμα περιήγησης του κάθε χρήστη. Στο μοντέλο Client-Server δηλαδή, αντιπροσωπεύουν το κομμάτι του Client. [7]

3.5.1 ΒΑΣΙΚΕΣ ΓΛΩΣΣΕΣ FRONT-END WEB DEVELOPMENT

Οι βασικές γλώσσες που χρησιμοποιούνται στις Front-End εφαρμογές ειναι οι HTML, CSS και JavaScript. Η HTML (HyperText Markup Language) ορίζει τη δομή μιας ιστοσελίδας. Μεσω της HTML, τα προγράμματα περιήγησης μπορούν να τυπώσουν κείμενο η να φορτώσουν στοιχεία που περιέχουν Hyperlinks (διασυνδέσεις) σε άλλες ιστοσελίδες.

Χρησιμοποιώντας την CSS (Cascading Style Sheets) μπορούμε να ορίσουμε το πως φαίνεται μια ιστοσελίδα. Ο σχεδιασμός και τα ο έλεγχος των στιλιστικών στοιχείων (Γραμματοσειρές, χρώματα κλπ.) για διάφορες συσκευές όπως κινητά, σταθερούς υπολογιστές και tablets γίνεται μέσω της CSS.[8]

Τέλος, η JavaScript είναι μια υψηλού επιπέδου, διερμηνευόμενη γλώσσα προγραμματισμού. Επιτρέπει στους προγραμματιστές να προσθέσουν διαδραστικότητα και δυναμικά στοιχεία στις ιστοσελίδες τους. Η JavaScript είναι πολύ σημαντικό κομμάτι του Front-End Web Development καθώς μπορεί να χρησιμοποιηθεί για την επεξεργασία στοιχείων μια ιστοσελίδας, για υπολογισμούς αλλά και για την αλληλεπίδραση με πόρους του Server. [9]

3.5.2 FRONT-END JAVASCRIPT FRAMEWORKS

Ένα Framework, στον προγραμματισμό, είναι ένα εργαλείο το οποίο παρέχει ήδη υλοποιημένα κομμάτια εφαρμογών που είναι σχεδιασμένα για την επιτάχυνση της ανάπτυξης μιας εφαρμογής. Τα Frameworks βασίζονται στην αρχή της αντιστροφής ελέγχου (Inversion of Control – IoC). Σε ένα συνηθισμένο σενάριο προγραμματισμού ο κώδικας καλεί την εκάστοτε βιβλιοθήκη για τη χρήση ήδη υλοποιημένων λύσεων. Με το IoC, το Framework καλεί την εκάστοτε βιβλιοθήκη όταν αυτό είναι απαραίτητο.[10]

Όπως αναφέρθηκε πριν, η JavaScript επιτρέπει τη προσθήκη διαδραστικότητας στις διαδικτυακές εφαρμογές. Κάθε φορά όμως που αλλάζουμε

την κατάσταση μιας εφαρμογής, πρέπει να αλλάξουμε και την διεπαφή του χρήστη. Αυτό ακριβώς το πρόβλημα λύνουν τα Front-End JavaScript Frameworks.

Υπάρχουν διάφορες επιλογές που θα μπορούσαν να αναφερθούν, παρολαυτα τα τέσσερα μεγαλύτερα τη συγκεκριμένη στιγμή είναι τα εξής:

1. **Ember:** Η Ember δημοσιεύτηκε τον Δεκέμβρη του 2011 ως συνέχεια του project Sproutcore. Είναι ένα παλιότερο Framework αλλά συνεχίζει να έχει δημοφιλία λόγω της σταθερότητας και της υποστήριξης από την κοινότητα που λαμβάνει.

2. **Angular:** Η Angular είναι ένα Web Application Framework ανοιχτού κώδικα που καθοδηγείται από την Angular Team στην Google και από τη κοινότητα προγραμματιστών και εταιρειών τεχνολογίας. Δημοσιεύτηκε πρώτη φορά στις 14 Σεπτεμβρίου του 2016. Η Angular είναι ένα Component-Based Framework και χρησιμοποιεί δηλωτική HTML. Κατά το build της εφαρμογής, χωρίς οι προγραμματιστές να μπορούν να το δουν, η Angular μετατρέπει την HTML σε βελτιστοποιημένες συναρτήσεις JavaScript. Η Angular χρησιμοποιεί την TypeScript, ένα υπερσύνολο της JavaScript.

3. **Vue:** Δημοσιεύτηκε το 2014 από τον Evan You. Είναι το νεότερο Framework των τεσσάρων. Η Vue, όπως και η Angular επεκτείνει την HTML με σύνταξη η οποία επιτρέπει την δηλωτική περιγραφή HTML στοιχείων με βάση την κατάσταση της JavaScript. [12]

4. **React**: Δημοσιεύτηκε από το Facebook το 2013. Η React χρησιμοποιούνταν ήδη εσωτερικά στην εταιρεία για την επίλυση πολλών προβλημάτων. Παρότι τεχνικά δεν είναι Framework αλλα βιβλιοθήκη που χρησιμοποιείται για την τύπωση στοιχείων UI, η προγραμματιστική κοινότητα κατηγοριοποιεί την React ως Framework καθώς μαζι με την ReactDOM χρησιμοποιούνται πολύ συχνά για την δημιουργία διαδικτυακών εφαρμογών. [13] Το Framework της React χρησιμοποιήθηκε και στην εφαρμογή που υλοποίησα.

3.6 BACK-END ΤΕΧΝΟΛΟΓΙΕΣ & ΤΕΧΝΟΛΟΓΙΕΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Με τον όρο Back-End εννοούμε το σύνολο των τεχνολογιών που χρησιμοποιούμε στο κομμάτι του Server μιας εφαρμογής. Αυτές συνήθως συμπεριλαμβάνουν κάποια βάση δεδομένων για την αποθήκευση των απαραίτητων δεδομένων για τη λειτουργία της εφαρμογής και κάποιον κώδικα που υλοποιεί ενα Application Programming Interface (API). Το Back-End κομμάτι της εφαρμογής συνήθως είναι εκεί που υλοποιείται η βασική λειτουργικότητα και λογική της εφαρμογής.

3.6.1 APPLICATION PROGRAMMING INTERFACES (APIs)

Με τον όρο Application Programming Interface εννοούμε ένα σύνολο από κανόνες η πρωτόκολλα που επιτρέπουν εφαρμογές να επικοινωνούν μεταξύ τους. Η χρήση ενός API επιτρέπει την ανταλλαγή μόνο των απαραίτητων πληροφοριών κρατώντας όλα τα υπόλοιπα εσωτερικά κρυμμένα. [14]

Όπως ανέφερα στην ανάλυση του μοντέλου Client-Server, όταν μιλάμε για μία διαδικτυακή εφαρμογή, ουσιαστικά αναφερόμαστε σε δύο διαφορετικά προγράμματα. Ένα το οποίο εκτελείται στον περιηγητή του χρήστη (Front-End) και περιέχει τη διεπαφή χρήστη και ένα που εκτελείται στον server (Back-End) και περιέχει την βάση δεδομένων και την λειτουργικότητα του προγράμματος.

Επειδή η Βάση Δεδομένων πιθανόν να περιέχει αρκετά ευαίσθητα δεδομένα (κωδικοί, προσωπικές διευθύνσεις κλπ.), μια εφαρμογή που παρέχει την κατάλληλη ασφάλεια αυτών των δεδομένων κρίνεται απαραίτητη. Αυτό ακριβώς μας προσφέρει ένα API.

Είναι χρήσιμο να σκεφτόμαστε την λειτουργικότητα των APIs ως αιτήματα και απαντήσεις. Η εφαρμογή από την πλευρά του περιηγητή (Client) κάνει ένα αίτημα ζητώντας κάποια δεδομένα. Συνήθως υπάρχει κάποια μορφή ταυτοποίησης του αιτούντα και έπειτα, εφόσον αυτός ταυτοποιηθεί, το API απαντάει με τα δεδομένα τα οποία ζητούνται.

Υπάρχουν διάφοροι τύποι από APIs όπως SOAP (Simple Object Access Protocol) APIs, RPC (Remote Procedure Calls) APIs, Websocket APIs αλλά εγώ θα επικεντρωθώ στα REST (Representational State Transfer) APIs καθώς αυτά είναι τα πιο ευέλικτα και δημοφιλή APIs καθώς και το API της εφαρμογής που υλοποίησα είναι αυτού του τύπου.

Τα REST APIs ορίζουν ένα σύνολο από λειτουργίες όπως GET, PUT και DELETE που οι clients μπορούν να χρησιμοποιήσουν για να αποκτήσουν πρόσβαση σε δεδομένα. Η ανταλλαγή των δεδομένων γίνεται μέσω του πρωτοκόλλου HTTP. Τα αιτήματα του Client έχουν την μορφή URL και οι απαντήσεις είναι απλά δεδομένα (τα οποία συνήθως μεταφέρονται σε μορφή αρχείων JSON). Το βασικό χαρακτηριστικό των REST APIs έχει να κάνει με το ότι δέν αποθηκεύουν δεδομένα του Client μεταξύ των αιτημάτων. [15]

3.6.2 BACK-END FRAMEWORKS/ΣΧΕΔΙΑΣΜΟΣ API

Σε αντίθεση με το Front-End όπου ο λειτουργικός προγραμματισμός, δηλαδή οτιδήποτε πέρα από τα δομικά (HTML) και στιλιστικά (CSS) στοιχεία, γίνεται εξ'ολοκλήρου σε JavaScript, το Back-End παρέχει μια ποικιλία γλωσσών και Frameworks για την δημιουργία και των σχεδιασμό των Application Programming Interfaces (APIs).

Θα αναφέρω τα 7 πιο δημοφιλή Frameworks για τον σχεδιασμό APIs:

1. **ASP.NET/ASP.NET Core:** Δωρεάν, ανοιχτού κώδικα Framework που δημοσιεύτηκε για πρώτη φορα το 2002. Είναι βασισμένο στη γλώσσα C# και στη πλατφόρμα .NET. Παρέχεται σε τρεις διαφορετικές εκδόσεις: Web Forms, ASP.NET MVC και ASP.NET Web Pages. Το ASP.NET παρέχει προηγμένη λειτουργικότητα για την ανάπτυξη real-time εφαρμογών. [16]

2. **Django:** Το Django είναι ένα απο τα πιο δημοφιλή Web Development Frameworks στην αγορά. Δημοσιεύτηκα τον Ιούλιο του 2005 και είναι βασισμένο στη γλώσσα Python.

   Το Django επιτρέπει στους προγραμματιστές να γράφουν ασφαλή και επεκτάσιμο κώδικα ο οποίος μπορεί να λειτουργήσει σε όλες τις πλατφόρμες που η γλώσσα Python υποστηρίζει, κάνοντάς το ένα από τα πιο εύχρηστα εργαλεία για την ανάπτυξη διαδικτυακών εφαρμογών.

   Το Django ελέγχει το σύστημα αιτημάτων-απαντήσεων του με το μοντέλο Model-View-Template. Το Model αντιπροσωπεύει την διεπαφή μεταξύ της βάσης δεδομένων και του κώδικα, είναι η λειτουργικότητα δηλαδή η οποία μετατρέπει τους πίνακες της βάσης δεδομένων σε κλάσεις και αντικείμενα της γλώσσας Python, μια διαδικασία γνωστή και ως Object Relational Mapping.

   Τα Views επεξεργάζονται τα αιτήματα που δέχονται από τα Models. Στα Views μπορούν να γραφτούν οι εκάστοτε συναρτήσεις που παράγουν τα απαραίτητα δεδομένα για την λειτουργικότητα της εφαρμογής και ανάλογα το αίτημα να επιστρέφουν τα κατάλληλα δεδομένα.
   Τέλος τα Templates χρησιμοποιούνται για τον ορισμό της δομής των δεδομένων των οποίων θα επιστραφούν.[17]

3. **Express.js:** Το Express.js είναι Framework βασισμένο πάνω στο περιβάλλον Node.js, το οποίο είναι γραμμένο στη γλώσσα JavaScript. Το Express.js δημοσιεύτηκε για πρώτη φορά το 2010 ενώ το περιβάλλον Node.js το 2009 στο λογισμικό Linux και το 2012 στο λογισμικό Windows.

   Το Express.js είναι αρκετά μινιμαλιστικό σαν Framework και επιτρέπει πλήρη ελευθερία στο τρόπο με τον οποίο δομείται ο κώδικας. Μια κλήση ενός κώδικα Express έχει τα εξής χαρακτηριστικά: ορίζεται από μια από τις βασικές δράσεις ενός REST API (GET, PUT, POST, DELETE κλπ.), ένα μοτίβο URL στο οποίο χρησιμοποιείται η κλήση (γνωστό και ως “Route”) και μεθόδους για τον ορισμό της δομής των δεδομένων των οποίων θα επιστραφούν.

   Μπορεί να χρησιμοποιηθεί επίσης και Middleware για την υποστήριξη τεχνολογιών όπως Cookies, sessions κλπ. Οι συνδέσεις με τις βάσεις

δεδομένων γίνονται μέσω της Node.js καθώς το Express δεν ορίζει κάποιον συγκεκριμένο τρόπο αλληλεπίδρασης με αυτές.[18]
Το Express.js είναι το Framework το οποίο επέλεξα για την ανάπτυξη του API μου.

4. **Java Spring/Spring Boot:** Το Java Spring είναι ένα περιβάλλον το οποίο χρησιμοποιείται για ανάπτυξη εφαρμογών με την γλώσσα Java. Το Java Spring προσφέρει Dependency Injections, τη δυνατότητα των αντικειμένων δηλαδή, να ορίζουν το προαπαιτούμενα προγράμματα που χρειάζονται για να τρέξουν, τα οποία το Java Spring αργότερα προσθέτει. Αυτό το χαρακτηριστικό κάνει το Java Spring ιδανικό για την ανάπτυξη αρχιτεκτονικών microservices.

   Ένα βασικό αρνητικό του Java Spring παρόλαυτα είναι η σχετική πολυπλοκότητα που προσδίδει στην ανάπτυξη μιας εφαρμογής. Αυτό ακριβώς το πρόβλημα λύνει το Java Spring Boot. Τρία βασικά χαρακτηριστικά που ορίζουν το Java Spring Boot είναι τα εξής: αυτόματη παραμετροποίηση, κατευθυνόμενη προσέγγιση και δημιουργία αυτόνομων εφαρμογών.

   Το Java Spring Boot αρχικοποιεί τις εφαρμογές με προ-ρυθμιζμένες βιβλιοθήκες. Αυτής της μορφής η παραμετροποίηση μειώνει την πιθανότητα λαθών και απλοποιεί την εφαρμογή ελατώνοντας τις αποφάσεις που πρέπει να πάρει ο προγραμματιστής.

   Με τον όρο κατευθυνόμενη προσέγγιση εννοούμε τις ενσωματωμένες απόψεις του Java Spring Boot για το πως πρέπει να δομηθεί και να αναπτυχθεί μια εφαρμογή. Πιο συγκεκριμένα στην διαδικασία της αρχικοποίησης ενός project μπορούν να οριστούν οι αρχικές ανάγκες του προγράμματος που λέγονται Spring Starters.
   Τέλος τον όρο αυτόνομες εφαρμογές εννοούμε τη δυνατότητα του Java Spring Boot να παράγει εφαρμογές οι οποίες τρέχουν αυτόνομα, χωρίς την ανάγκη κάποιου Web Server. Το Java Spring δημιουργήθηκε το 2003 ενώ το Java Spring Boot το 2014 απο την VMWare. [19]

5. **Flask:** Το Flask είναι ένα Framework βασισμένο στη γλώσσα Python που δημοσιεύτηκε τον Απρίλιο του 2010 από τον προγραμματιστή Armin Ronacher. Το Flask είναι σχεδιασμένο να είναι μινιμαλιστικό και να παρέχει μόνο τα απαραίτητα στοιχεία για την ανάπτυξη διαδικτυακών εφαρμογών και για αυτό το λόγο αποκαλείται Micro-Framework.

   Το Flask χρησιμοποιεί το Web Server Gateway Interface που είναι μια κοινή διεπαφή μεταξύ Servers και εφαρμογών που χρησιμοποιείται στην ανάπτυξη διαδικτυακών εφαρμογών μέσω της γλώσσας Python. Τα

βασικά θετικά του Flask είναι η απλότητα, η επεκτασιμότητα καθώς και η ευελιξία η οποία προσφέρει. [20]

6. **Ruby on Rails (Rails):** Το Rails είναι ένα Framework ανοιχτού κώδικα βασισμένο στη γλώσσα Ruby το οποίο δημοσιεύτηκε για πρώτη φορά το 2004.

   Ακολουθεί δύο βασικές αρχες: Don't Repeat Yourself και Convention over Configuration. Η πρώτη αρχή έχει να κάνει με τη μείωση της επανάληψης κώδικα. Αυτό σημαίνει ότι κάθε κομμάτι λογικής, λειτουργικότητας ή αναπαράστασης δεδομένων πρέπει να υπάρχει μόνο μία φορά στο σύστημα, κάτι το οποίο κάνει τον κώδικα πολύ πιο συντηρίσιμο, επιτρέπει πολύ μεγαλύτερη δυνατότητα επέκτασης καθώς και μειώνει την πιθανότητα σφαλμάτων.

   Η δεύτερη αρχή έχει να κάνει με την κατευθυνόμενη προσέγγιση. Στο Rails υπάρχει ένας βέλτιστος τρόπος να γραφτεί ένα πρόγραμμα ο οποίος συνήθως ενθαρρύνεται, πολλές φορές επίσης άλλοι τρόποι υλοποίησης της ίδιας λειτουργικότητας αποθαρρύνονται. [21]

7. **Laravel:** Το Laravel είναι ένα δημοφιλές Framework βασισμένο στη γλώσσα PHP το οποίο δημοσιεύτηκε το 2011. Παρέχει πολλές λειτουργικότητες όπως ενσωματωμένη ταυτοποίηση και ενα Command Line Interface (Artisan CLI) και χρησιμοποιείται κυρίως για δημιουργία blogs και ιστοσελίδων τύπου e-commerce.

   Το Artisan CLI παρέχει λειτουργικότητες όπως έλεγχο βάσεων δεδομένων, δημιουργία κλήσεων και μεταφορά δεδομένων καθως και για την δημιουργία συστημάτων ταυτοποίησης χρηστών. Δυο πολύ χρήσιμα εργαλεία στο Laravel Framework είναι επίσης το Redis, το οποίο είναι ενα key-value store που χρησιμοποιείται για προσωρινή αποθήκευση δεδομένων επιτρέπωντας έτσι τη βέλτιστη απόδοση της εφαρμογής καθώς και το Eloquent Object Relation Mapper το οποίο απλοποιεί την αλληλεπίδραση με την εκάστοτε βάση δεδομένων. [22]

3.6.3 ΤΕΧΝΟΛΟΓΙΕΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Με τον όρο Βάση Δεδομένων εννοούμε μια συλλογή δεδομένων η οποία είναι οργανωμένη για εύκολη πρόσβαση, διαχείριση και ενημέρωση. Σε μια διαδικτυακή εφαρμογή η Βάση Δεδομένων χρησιμοποιείται για την αποθήκευση όλων των δεδομένων που μπορεί να παράξει η εφαρμογή η μπορεί να είναι απαραίτητα για

την λειτουργικότητα της.

Ένα απλό παράδειγμα μιας τέτοιας λειτουργικότητας είναι ένα σύστημα χρηστών (κάτι το οποίο έχω υλοποιήσει και στην εφαρμογή μου). Όταν ένας χρήστης εγγράφεται σε μία υπηρεσία εισάγει κάποια στοιχεία σε αυτήν (όπως το όνομα χρήστη και ο κωδικός του). Τα στοιχεία αυτά μετά αποθηκεύονται σε μια βάση δεδομένων ώστε, όταν ο χρήστης ξανασυνδεθεί στην υπηρεσία, τα δεδομένα αυτά να χρησιμοποιηθούν για την ταυτοποίηση του.

Οι βάσεις δεδομένων συνήθως διαχειρίζονται από ένα Database Management System (DBMS) και πολλές απο αυτές χρησιμοποιούν την γλώσσα SQL (Structured Query Language) για την δόμηση, την εγγραφή, την διαγραφή και την αναζήτηση δεδομένων.

Ένα Database Management System είναι μια διεπαφή μεταξύ ενός χρήστη και μίας βάσης δεδομένων. Το DBMS διαχειρίζεται τα δεδομένα, την εφαρμογή και το σχήμα της βάσης δεδομένων και επιτρέπει ασφαλής και ακριβής αλληλεπιδράσεις με δεδομένα. Τα DBMSs μπορούν να διαχωριστούν σε τέσσερις κατηγορίες: Κατανεμημένα DBMSs που περιέχουν ένα σύνολο από λογικά συνδεδεμένες βάσεις δεδομένων κατανεμημένες σε ένα δίκτυο, Ιεραρχικά DBMSs όπου τα δεδομένα οργανώνονται σε μορφή δέντρου (top-down η bottom-up), Δικτυακα DBMSs οπού οργανώνονται σε μορφή γράφου και επιτρέπουν την πιο περίπλοκη οργάνωση δεδομένων, Σχεσιακά DBMSs όπου τα δεδομένα οργανώνονται σε πίνακες (η πιο δημοφιλή μορφή DBMS) και Αντικειμενοστρεφή DBMSs οπού τα δεδομένα αποθηκεύονται σε αντικείμενα με τη δυνατότητα υλοποίησης μεθόδων.

Υπάρχουν πολλές κατηγορίες βάσεων δεδομένων αλλά οι δύο βασικές είναι οι Σχεσιακές και οι Μη-Σχεσιακές βάσεις δεδομένων. [23]

3.6.3.1 ΣΧΕΣΙΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Με τον όρο Σχεσιακή Βάση Δεδομένων εννοούμε την συλλογή δεδομένων σε πίνακες με σειρές και στήλες οι οποίες μας επιτρέπουν να δούμε πως τα δεδομένα σχετίζονται αναμεταξύ τους. Ως σχέσεις ορίζουμε τις λογικές συνδέσεις μεταξύ δεδομένων ακόμη και από διαφορετικούς πίνακες οι οποίες γίνονται με βάση την αλληλεπίδραση των δεδομένων.

Μία τέτοια σχέση θα μπορούσε να αναπτυχθεί πάνω στο παράδειγμα που έδωσα πριν. Έστω πως τα στοιχεία τον χρηστών αποθηκεύονται σε ένα πίνακα ονόματι “Χρήστες” όπου έχει τις στήλες “Όνομα Χρήστη” και “Κωδικός Πρόσβασης”. Η εφαρμογή που υλοποιούμε είναι ένα σαιτ τύπου e-commerce (ηλεκτρονικής αγοραπολησίας). Θα χρειαζόμασταν ένα ακόμη πίνακα ονόματι “Παραγγελίες” με τις στήλες “Προιόν” και “Όνομα Χρήστη”, βλέπουμε εδώ πως δύο στήλες στους

πίνακες μας είναι ίδιες και υπάρχει μια σχέση μεταξύ τους. Μπορούμε έτσι να χρησιμοποιήσουμε το ίδιο Όνομα χρήστη ώστε να ελέγξουμε όλες τις παραγγελίες που έχει κάνει.

Οι σχεσιακές βάσεις δεδομένων χρησιμοποιούν την γλώσσα SQL (Structured Query Language). Εντολές της γλώσσας SQL μπορούν να χρησιμοποιηθούν για την εγγραφή, τη διαχείριση, την εμφάνιση και τη διαγραφή δεδομένων μέσα σε ένα πίνακα. Η γλώσσα SQL στις διαδικτυακές εφαρμογές χρησιμοποιείται συνήθως σε συνδυασμό με άλλες γλώσσες στο Back-End μέρος της εφαρμογής για την αλληλεπίδραση με την βάση δεδομένων.

Οι εντολές SQL μπορούν να κατηγοριοποιηθούν ως εξής: Γλώσσα Ορισμού δεδομένων (Data Definition Language) η οποία ορίζει τη δομή μιας βάσης δεδομένων (CREATE εντολή), Γλώσσα Ερωτήσεων Δεδομένων (Data Query Language) που περιλαμβάνει τις οδηγίες για την ανάκτηση δεδομένων από την βάση (SELECT εντολή), Γλώσσα Χειρισμού Δεδομένων (Data Manipulation Language) που χρησιμοποιείται για την επεξεργασία η την εισαγωγή καινούργιων δεδομένων σε μία βάση (INSERT εντολή), Γλώσσα Ελέγχου Δεδομένων (Data Control Language) που χρησιμοποιείται για την εξουσιοδότηση πρόσβασης στη βάση δεδομένων σε χρήστες (GRANT εντολή) και Γλώσσα Ελέγχου Συναλλαγών (Transactional Control Language) για αυτοματοποιημένες αλλαγές στη βάση δεδομένων (ROLLBACK εντολή). [24]

Οι σχεσιακές βάσεις δεδομένων σχεδιάστηκαν το 1970 από την E.F. Codd της IBM, ο όρος “σχεσιακή” εμφανίστηκε πρώτη φορά στη δημοσίευσή του με τίτλο: “A Relational Model of Data for Large Shared Data Banks”. Ο Codd όρισε δεκατρείς κανόνες τους οποίους πρέπει να τηρεί μια βάση δεδομένων για να θεωρηθεί σχεσιακή, οι οποίοι μπορούν να περιληφθούν σε δύο βασικά χαρακτηριστικά. Τα οποία είναι η παρουσίαση των δεδομένων σε σχεσιακή μορφή (σε πίνακες με γραμμές και στήλες) και η παροχή σχεσιακών πράξεων για την διαχείριση των δεδομένων σε μορφή πίνακα. [25]

3.6.3.2 ΜΗ-ΣΧΕΣΙΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Με τον όρο Μη-Σχεσιακές Βάσεις Δεδομένων εννοούμε τις βάσεις δεδομένων που δεν χρησιμοποιούν το μοντέλο πινάκων με γραμμές και στήλες το οποίο συναντάμε στα πιο πολλά συστήματα. Αυτό που διαχωρίζει τις Μη-Σχεσιακές Βάσεις Δεδομένων είναι το ότι η δομή τους συνήθως είναι σχεδιασμένη έτσι ώστε να εξυπηρετεί την ανάγκη μιας εφαρμογής. Για παράδειγμα τα δεδομένα μπορεί να είναι οργανωμένα σε ζεύγη κλειδιών-τιμών, σε JavaScript Object Notation (JSON) αρχεία η ακόμη και σε διαφορετικές δομές δεδομένων όπως γράφοι η δέντρα. [26]

Οι Μη-Σχεσιακές Βάσεις Δεδομένων προσφέρουν πολλά θετικά χαρακτηριστικά όπως οργάνωση δεδομένων μεγάλου μεγέθους καθώς αρκετές φορές τέτοιες βάσεις δεδομένων διευκολύνουν την αναζήτηση κάποιων συγκεκριμένων τιμών περισσότερο από ότι μια βάση δεδομένων που χρησιμοποιεί εντολές SQL. Σημαντικό πλεονέκτημα επίσης είναι το ότι οι Μη-Σχεσιακές Βάσεις Δεδομένων επιτρέπουν την αποθήκευση δεδομένων

ανεξαρτήτως του τύπου τους κάτι το οποίο τις κάνει ιδανικές για σύγχρονές διαδικτυακές εφαρμογές οι οποίες πολύ συχνά έχουν την ανάγκη αποθήκευσης πολλών διαφορετικών τύπων δεδομένων. [27]

Οι Μη-Σχεσιακές Βάσεις Δεδομένων υπάρχουν από το 1960, άλλα άρχισαν να λαμβάνουν προσοχή στις αρχές του 2000 με την άνοδο των τεχνολογιών Web 2.0. [28]

4. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

4.1 ΣΚΟΠΟΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Ο σκοπός της εφαρμογής είναι η δημιουργία μιας υπηρεσίας ανταλλαγής είτε ψηφιακών (σε μορφή PDF) είτε αναλογικών βιβλίων. Κάτι τέτοιο θα μπορούσε να αποβεί πολύ κερδοφόρο για τους χρήστες, καθώς θα έχουν τη δυνατότητα να ανταλλάσσουν τα βιβλία τους με άλλους αναγνώστες χωρίς το κόστος αγοράς νέων βιβλίων. Επιπλέον έμφαση έχει δοθεί στην οργάνωση της πλατφόρμας έτσι ώστε να είναι εύχρηστη και αποδοτική για όλους τους χρήστες, σε αυτό το κεφάλαιο θα αναλύσω τις τεχνολογίες που χρησιμοποίησα καθώς και τον τρόπο με τον οποίο υλοποιήθηκε η εφαρμογή. Αργότερα, στο επόμενο κεφάλαιο, θα δοθεί μια πλήρη παρουσίαση της διεπαφής καθώς και επεξηγηματικές οδηγίες χρήσης για την εγγραφή, την χρήση και τη διαχείριση της εφαρμογής και από χρήστες αλλά και από διαχειριστές.

4.2 ΔΙΕΠΑΦΗ ΧΡΗΣΤΗ (USER INTERFACE)

4.2.1 REACT COMPONENTS

Όπως αναφέρθηκε και πρότερα, το Front-End μέρος της εφαρμογής είναι βασισμένο στο Framework της React. Η React έχει παρέχει πολλά εργαλεία που επιταχύνουν και οργανώνουν την διαδικασία της ανάπτυξης διαδικτυακών εφαρμογών. Ένα πολύ σημαντικό είναι τα components.

Τα React Components είναι επαναχρησιμοποιήσιμα κομμάτια κώδικα υπεύθυνα για την τύπωση ενός στοιχείου μιας εφαρμογής. [29] Τα Components μας επιτρέπουν ουσιαστικά να εφαρμόσουμε ιεραρχικό προγραμματισμό στην σελίδα μας “σπάζοντας” ένα πρόβλημα σε μικρότερα, πολύ πιο διαχειρίσιμα, υποπροβλήματα και οργανώνοντας την δομή, τον σχεδιασμό και την λογική μιας σελίδας. Τα Components που υλοποίησα για την εφαρμογή είναι τα εξής: Item, ItemList, Navbar, Pagination, Sidebar και TaC.

Μια σημαντική λειτουργικότητα που μου ήταν απαραίτητη για την εφαρμογή είναι η απαρίθμηση στοιχείων. Είτε αυτά είναι χρήστες, είτε βιβλία η εφαρμογή χρειάζεται μια λογική μονάδα που να περιγράφει οντότητες και τα χαρακτηριστικά τους. Σε αυτήν την μονάδα πρέπει να μπορώ να δίνω συγκεκριμένες παραμέτρους ορίζοντας το κάθε αντικείμενο και κάνοντας το χρήσιμο για την εφαρμογή.

Αυτή η μονάδα είναι το Item Component, το οποίο δέχεται διάφορες παραμέτρους ανάλογα με την εκάστοτε περίπτωση και μπορεί να περιγράψει είτε χρήστες, είτε βιβλία.

J.R.R. Tolkien	**The Hobbit**	Fantasy	1	Physical

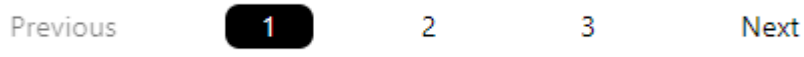
Παρότι αυτό το Component μας παρέχει αρκετές πληροφορίες για το εκάστοτε αντικείμενο (στο διάγραμμα βλέπουμε πληροφορίες όπως τον τίτλο, τον συγγραφέα και τον τύπο ενός βιβλίου), δεν αρκεί από μόνο του για τη λειτουργία της απαρίθμησης. Ένα εξίσου σημαντική μονάδα που χρειάζεται είναι η λίστα, αυτό τον ρόλο εξυπηρετεί το ItemList Component.

Το ItemList Component χρησιμοποιεί έντονα την δομή επανάληψης και την πολύ χρήσιμη λειτουργία της React να μπορεί να χρησιμοποιεί JavaScript κώδικα πάνω σε HTML στοιχεία, για να εμφανίζει επαναληπτικά τον απαραίτητο αριθμό απο Items για την εκάστοτε περίπτωση. Ένα χαρακτηριστικό παράδειγμα είναι ο πίνακας ελέγχου του διαχειριστή στον οποίο έχουμε την ανάγκη για απαρίθμηση χρηστών.

USER ID	USERNAME	EMAIL	AREA	POINTS	DAYS BANNED
2	**admin**	learn2earnproj@gmail.com	Athens	3 Ph. & 3 PDF	0
3	**test1**	test1@gmail.com	Athens	3 Ph. & 3 PDF	0
5	**test3**	test3@gmail.com	Chalcis	3 Ph. & 3 PDF	0
4	**test2**	test2@gmail.com	Athens	3 Ph. & 4 PDF	0
1	**trikone**	trikprd@gmail.com	Athens	3 Ph. & 2 PDF	0

Ένα σημαντικό πρόβλημα το οποίο προκύπτει στην διαδικασία της απαρίθμησης είναι το μέγεθος το οποίο μπορεί να φτάσει μια λίστα. Οι σελίδες της εφαρμογής πρέπει να είναι ευδιάκριτες και η ανάγκη τύπωσης πάρα πολλών στοιχείων θα απαιτούσε αρκετή υπολογιστική ισχύ για μια λειτουργία η οποία εν τέλει δεν είναι και ιδιαίτερη χρήσιμη η πρακτική.

Για αυτό τον λόγο δημιούργησα το Pagination component, το οποίο υλοποιεί τη λειτουργία της σελιδοποίησης. Το Pagination component δέχεται ως παραμέτρους το μέγεθος και το περιεχόμενο του ItemList και τυπώνει στη διεπαφή ένα μέρος του ItemList, μια σειρά από αριθμούς που αντιπροσωπεύουν σελίδες την οποία μπορούμε να πατήσουμε για να μεταφερθούμε στις σελίδες αυτές καθώς και δύο κουμπιά που αντιπροσωπεύουν το “εμπρός” και το “πίσω” τα οποία μπορούμε να χρησιμοποιήσουμε επίσης για μετακίνηση.



Μια λειτουργικότητα σχετική με την απαρίθμηση, η οποία πρέπει η εφαρμογή μας να επιτρέπει στον εκάστοτε χρήστη, είναι τα φίλτρα. Με ένα φίλτρο ο χρήστης μπορεί να ελέγξει την πληροφορία η οποία εμφανίζεται στην οθόνη του. Αυτό επιτρέπει στον χρήστη μεγαλύτερη ακρίβεια και κατηγοριοποίηση είτε μεταξύ βιβλίων, είτε μεταξύ χρηστών.

Τα φίλτρα έχουν βασικά χαρακτηριστικά είτε για τα βιβλία, όπως οι κατηγορίες η ο τύπος -φυσικός η ψηφιακός- του κάθε βιβλίου, είτε για τους χρήστες, όπως η περιοχή και το αν ο χρήστης έχει η όχι αποκλειστεί από την σελίδα. Τα φίλτρα τυπώνονται δυναμικά, ανάλογα με το αν χρησιμοποιούνται σε απαρίθμηση βιβλίου η χρήστη σε ένα Component που λέγεται Sidebar. Όπως υποδηλώνει και το όνομα το Sidebar βρίσκεται στην αριστερή άκρη της σελίδας, δίπλα στο ItemList.

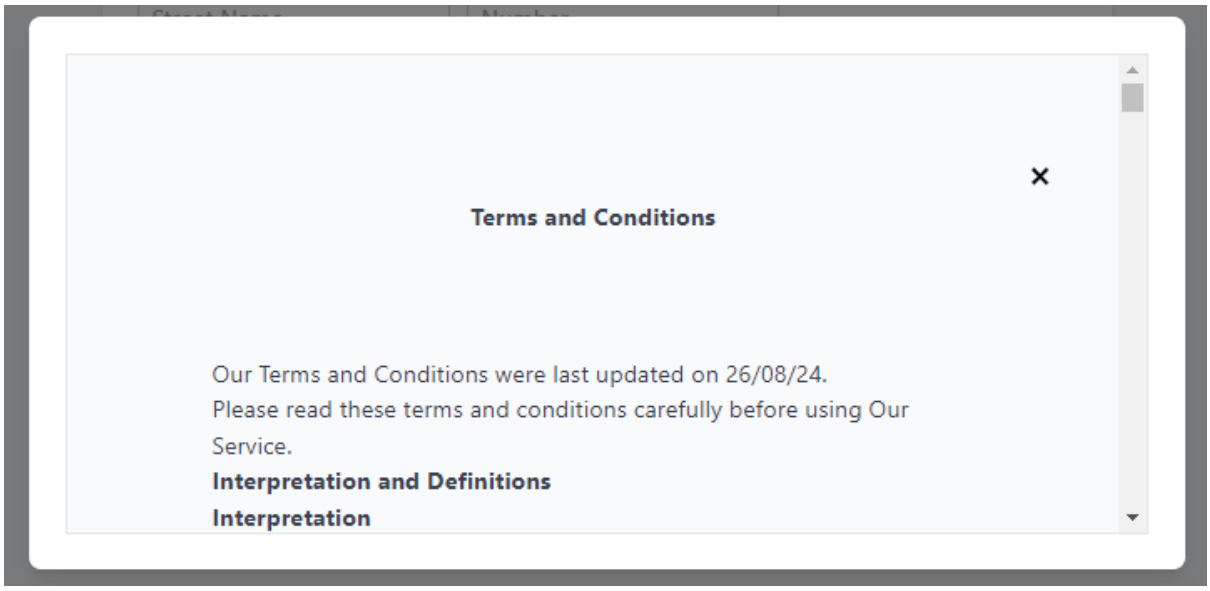
Το Sidebar όπως και το Pagination Component, δέχεται ως παραμέτρους την λίστα με τα αντικείμενα και επιστρέφει στη διεπαφή, μέσω των φίλτρων, τα αντικείμενα τα οποία θέλει ο χρήστης να δει. Ένα απλό παράδειγμα ενός φίλτρου είναι η κατηγορία των βιβλίων, ο χρήστης μπορεί να επιλέξει να δει μόνο βιβλία επιστημονικής φαντασίας και να επιλέξει από αυτά, αυτό που τον ενδιαφέρει.

Μια επίσης πολύ σημαντική λειτουργικότητα για την εφαρμογή είναι η πλοήγηση. Η εφαρμογή για να είναι λειτουργική χρειάζεται να προσφέρει έναν τρόπο στους χρήστες να μετακινούνται από το ένα κομμάτι της διεπαφής στο άλλο. Αυτό το πρόβλημα λύνει το Navbar Component.

Το Navbar Component είναι μια μπάρα η οποία υπάρχει στο πάνω μέρος σχεδόν κάθε σελίδας της εφαρμογής. Μέσω του Navbar ο χρήστης μπορεί να μετακινηθεί στις διάφορες σελίδες που του παρέχονται ως κουμπιά. Λειτουργεί ως περιηγητής και εκτυπώνει δυναμικά τις σελίδες στις οποίες έχει πρόσβαση ο κάθε χρήστης. Για παράδειγμα, όταν ένας χρήστης δεν είναι συνδεδεμένος, το Navbar τυπώνει το κουμπί Register/Log In που αντιπροσωπεύει την ανάλογη σελίδα σύνδεσης η

εγγραφής. Ενώ όταν συνδεθεί, το Navbar τυπώνει το όνομα χρήστη, το οποίο αν πατηθεί πηγαίνει τον χρήστη στη σελίδα Προφίλ.

Το τελευταίο και το πιο απλό component το οποίο υλοποίησα είναι το TaC το οποίο ορίζει τους Όρους και Προϋποθέσεις (Terms and Conditions) της εφαρμογής. Κάθε εφαρμογή η οποία δημοσιεύεται στο Internet υπόκειται σε ορισμένα νομικά πλαίσια οπότε ένα Component το οποίο να ορίζει τους όρους, τις προϋποθέσεις καθώς και την γενικότερη νομοθεσία γύρω από τη χρήση της σελίδας κρίνεται απαραίτητο.



Συνδιάζοντας τα παραπάνω Components και τον απαραίτητο κώδικα για τις ανάγκες τις κάθε λειτουργίας δημιουργούνται οι σελίδες της εφαρμογής οι οποίες θα παρουσιαστούν μαζί με τις οδηγίες χρήσης τους στο επόμενο κεφάλαιο.

4.2.1 TAILWIND CSS: ΣΧΕΔΙΑΣΜΟΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Πέραν των λειτουργικών κομματιών μιας εφαρμογής, ένα σημαντικό κομμάτι το οποίο την κάνει συνειδητά πιο εύχρηστη καθώς και πιο ενδιαφέρουσα είναι οι στιλιστικές επιλογές. Η αισθητική της εκάστοτε εφαρμογής είναι αυτό το οποίο ορίζει τον χαρακτήρα και την ταυτότητά της. Μια εφαρμογή πρέπει να διατηρεί ένα κοινό θέμα έτσι ώστε να δίνεται η αίσθηση της ομοιογένειας χωρίς αυτό να καταλήγει βαρετό και επαναλαμβανόμενο. Αυτό ακριβώς μας επιτρέπει η

CSS.

Στο κεφάλαιο 3.5 ανέφερα πως ο σχεδιασμός και ο στιλιστικός έλεγχος των στοιχείων γίνεται μέσω της γλώσσας CSS. Η γλώσσα CSS, συνήθως λειτουργεί με κλάσεις τις οποίες ορίζουμε πάνω στα στοιχεία HTML. Έπειτα μπορούμε να κάνουμε αναφορά σε αυτές τις κλάσεις σε ένα διαφορετικό αρχείο και να ορίσουμε διάφορα αισθητικά χαρακτηριστικά τους όπως το χρώμα η τη γραμματοσειρά αν αναφερόμαστε σε κείμενο.

Αυτή η προσέγγιση επιτρέπει μια ομοιογενή στιλιστική επεξεργασία καθώς μας επιτρέπει να ορίσουμε την ίδια κλάση σε πάνω από ένα στοιχεία, ομαδοποιώντας τα και δημιουργώντας μια πιο κατηγοριοποιημένη διαδικασία στιλιστικής επεξεργασίας η οποία τείνει να είναι πολύ αποδοτική για μικρές εφαρμογές και ιστοσελίδες.

Ένα πρόβλημα το οποίο συναντάμε πολύ συχνά στην CSS, είναι η δυσκολία που παρουσιάζει σε θέματα επεκτασιμότητας. Μια εφαρμογή η οποία έχει πολύ κώδικα μπορεί να απαιτεί πολλές διαφορετικές κλάσεις, κάτι το οποίο μπορεί γρήγορα να γίνει πολύ περίπλοκο. Αυτό ακριβώς το πρόβλημα λύνει η Tailwind CSS.

Με την Tailwind CSS, αντί για τη δημιουργία κλάσεων οι οποίες ορίζονται σε κάποιο ξεχωριστό αρχείο, η διαδικασία της στιλιστικής επεξεργασίας γίνεται κατευθείαν επάνω στο εκάστοτε στοιχείο HTML. Αυτό επιτρέπει πιο λεπτομερής στιλιστικές επιλογές και δίνει την επιλογή της άμεσης και ξεκάθαρης επεξεργασίας των αισθητικών στοιχείων μιας εφαρμογής χωρίς την αφαιρετικότητα μίας κλάσης.

Στο σημείο αυτό να αναφερθεί πως η Tailwind CSS λειτουργεί πολύ καλά με την React λόγο της ιδιότητας των Components να οργανώνουν των κώδικα σε μικρά κομμάτια. Κάτι το οποίο προσδίδει στα στιλιστικά στοιχεία την οργάνωση που προσδίδει και η CSS και χάνεται με την χρήση της Tailwind CSS.

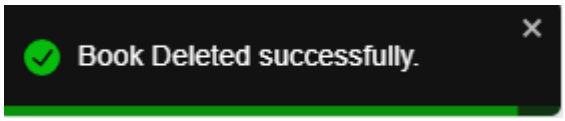
```
            <th className="px-5 py-3 border-b-2 border-gray-200 bg-gray-100
text-left text-xs font-semibold text-gray-600 uppercase tracking-wider">
                                        Username
                                    </th>
```

4.2.2 TOASTIFY: ΕΙΔΟΠΟΙΗΣΕΙΣ ΧΡΗΣΤΗ

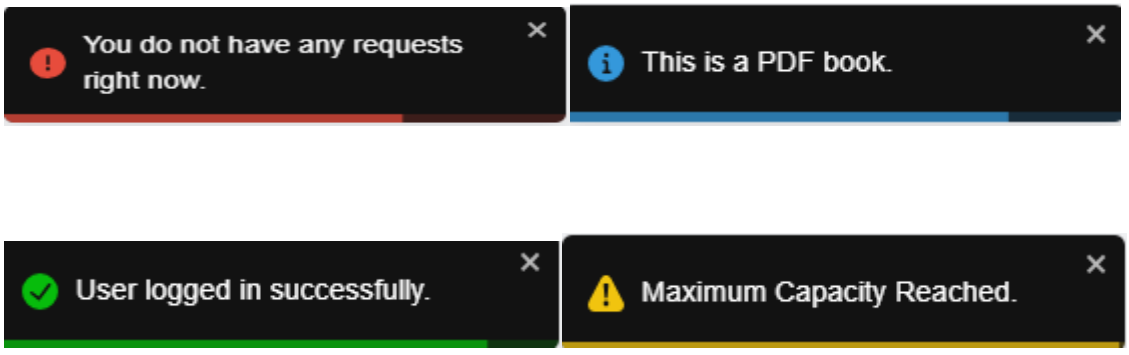
Πολλές φορές είναι σημαντικό να μπορούμε να επικοινωνήσουμε κάποια πληροφορία στον χρήστη χωρίς απαραίτητα να τυπώσουμε ένα ολόκληρο στοιχείο στην διεπαφή, το οποίο θα μείνει μόνιμα εκεί. Ένα τέτοιο σενάριο θα ήταν η διαγραφή κάποιου βιβλίου από ένα ItemList. Μετά από μία τέτοια δράση θα πρέπει να είμαστε σε θέση να τυπώσουμε μια ενημέρωση στη διεπαφή του χρήστη η οποία θα επικοινωνεί πως το βιβλίο διαγράφηκε και μετά θα εξαφανίζεται. Αυτή

η ενημέρωση λέγεται ειδοποίηση και την υλοποιούμε με την βοήθεια της βιβλιοθήκης Toastify.

Η βιβλιοθήκη Toastify παρέχει ειδοποιήσεις οι οποίες χωρίζονται σε τέσσερις κατηγορίες: Επιτυχία, Σφάλμα, Πληροφορία και Προειδοποίηση. Αυτές οι κατηγορίες μας επιτρέπουν να μετατρέπουμε το κατάλληλο μήνυμα στον χρήστη. Στο προηγούμενο παράδειγμα η ειδοποίηση θα έπρεπε να φαίνεται κάπως έτσι:



Η ειδοποιήσεις της Toastify είναι πλήρως προσαρμόσιμες σε θέμα χρωμάτων και κειμένου. Τα μόνα που μένουν σταθερά είναι τα χρώματα της εκάστοτε κατηγορίας, το κόκκινο για τα σφάλματα, το μπλε για τις πληροφορίες, το κίτρινο για τις προειδοποιήσεις και το πράσινο για τις επιτυχίες. Παρέχουν τέλος ένα κινούμενο χρονόμετρο που φαίνεται στο κάτω μέρος της ειδοποίησης το οποίο δείχνει την ώρα που θα μείνει ενεργή η ειδοποίηση.



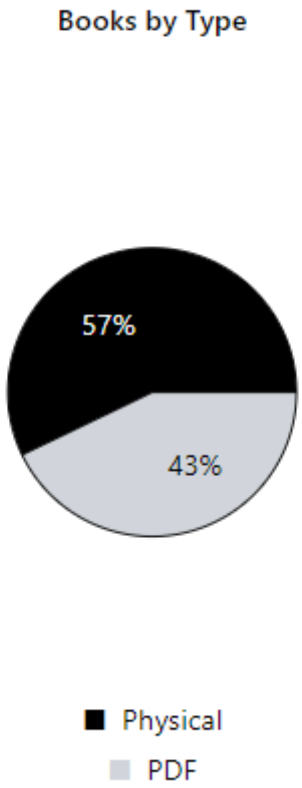
4.2.3 RECHARTS: ΓΡΑΦΗΜΑΤΑ ΔΕΔΟΜΕΝΩΝ

Ίσως το πιο χρήσιμο εργαλείο για έναν διαχειριστή είναι ένας τρόπος να οπτικοποιεί τα δεδομένα του. Κάτι τέτοιο του επιτρέπει να προβλέπει σφάλματα σε μία πλατφόρμα και να περιορίζει περιπτώσεις απάτης η προβλήματα ασφαλείας. Τα γραφήματα δεδομένων παίρνουν διάφορες μετρικές της εφαρμογής και τις παρουσιάζουν σε μορφή τέτοια, έτσι ώστε να αποκαλύπτουν κάποια γνώση η να περιγράφουν κάποια συνθήκη του συστήματος στον χρήστη.

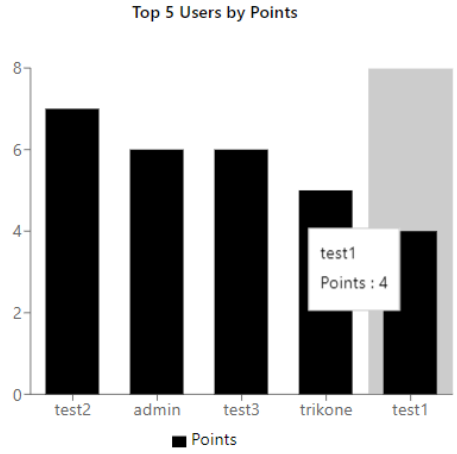
Τα γραφήματα της εφαρμογής υλοποιήθηκαν με την βοήθεια της βιβλιοθήκης ReCharts, η οποία παρέχει μια ολοκληρωμένη λύση για την παραγωγή πλήρως

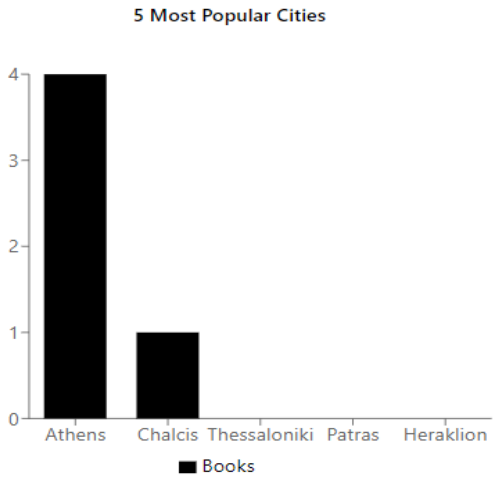
παραμετροποιήσημων γραφημάτων, όλων των ειδών. Οι δύο τύποι γραφημάτων που υλοποιήθηκαν είναι τα γραφήματα “Πίτας” και τα Ραβδογράμματα.

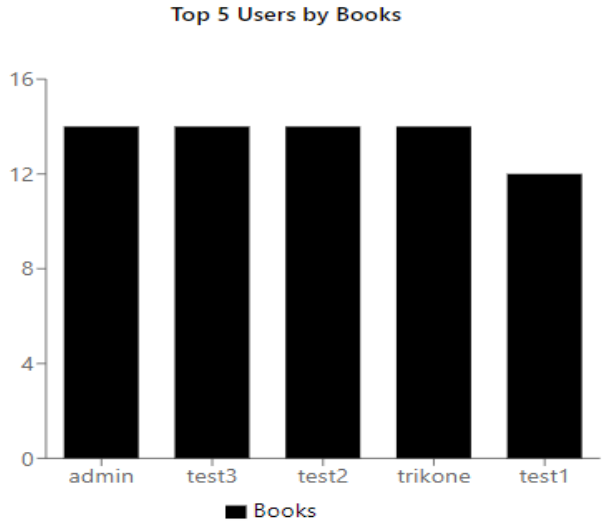
Τα γραφήματα “πίτας” απεικονίζουν τις τιμές ως τμήματα ενός κύκλου, όπου κάθε τμήμα αντιπροσωπεύει ένα μέρος του συνόλου. Συνήθως χρησιμοποιούνται για να δείξουν την συμβολή ενός χαρακτηριστικού σε ένα σύνολο. Στην εφαρμογή τα χρησιμοποιώ για να δείξω τα ποσοστά τον βιβλίων ανάλογα με τον τύπο τους (Φυσικό η PDF).



Τα Ραβδογράμματα χρησιμοποιούν ράβδους για την απεικόνιση δεδομένων. Το ύψος της εκάστοτε ράβδου απεικονίζει το μέγεθος μίας μονάδας. Τα ραβδογράμματα είναι πολύ χρήσιμα για την οπτικοποίηση των τιμών ξεχωριστά. Στην εφαρμογή υλοποίησα ραβδογράμματα για την οπτικοποίηση πολλών χαρακτηριστικών όπως τους κορυφαίους πέντε χρήστες με βάση τους πόντους τους, τους κορυφαίους πέντε χρήστες με βάση τα βιβλία που κατέχουν καθώς και τις πέντε πιο δημοφιλής πόλεις από τις οποίες εγγράφονται χρήστες στην εφαρμογή.







4.3 ΣΧΕΔΙΑΣΜΟΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Όπως ήδη αναφέρθηκε η Βάση δεδομένων που υλοποίησα στην εφαρμογή μου είναι η PostgreSQL. Η PostgreSQL είναι σχεσιακή βάση δεδομένων, κάτι το οποίο σημαίνει πως τα δεδομένα είναι αποθηκευμένα σε πίνακες με γραμμές και στήλες στις οποίες κάθε γραμμή υποστηρίζει μια οντότητα και κάθε στήλη ενα χαρακτηριστικό της.

4.3.1 ΠΙΝΑΚΑΣ “books”

Στη βάση δεδομένων μου υπάρχουν πέντε πίνακες και ο καθένας εξυπηρετεί έναν συγκεκριμένο σκοπό. Ο πρώτος και πιο βασικός πίνακας είναι ο πίνακας “books” ο οποίος χρησιμοποιείται για την αποθήκευση όλων των βιβλίων των οποίων υπάρχουν στην βάση δεδομένων. Έχει όλα τα βασικά χαρακτηριστικά που χρειάζονται για την ταυτοποίηση των βιβλίων όπως το όνομα του συγγραφέα, τον τίτλο, το τύπο, το είδος και τους πόντους του κάθε βιβλίου (βλέπε επόμενο κεφάλαιο). Στον πίνακα “books” τα βιβλία αποθηκεύονται ως μοναδικές τιμές. Δηλαδή, αν δύο χρήστες έχουν το ίδιο βιβλίο, το βιβλίο αυτό θα εμφανιστεί μόνο μία φορά στην βάση δεδομένων, υποδεικνύοντας έτσι πως το βιβλίο υπάρχει στο σύστημα.

book_id	book_name	book_author	book_genre	book_points	book_type
1	The Great Gatsby	F. Scott Fitzgerald	Fiction	1	PDF
2	To Kill a Mockingbird	Harper Lee	Fiction	1	Physical
3	1984	George Orwell	Dystopian	1	PDF
4	The Catcher in the Rye	J.D. Salinger	Fiction	1	PDF
5	The Hobbit	J.R.R. Tolkien	Fantasy	1	Physical
6	The Old Man and the Sea	Ernest Hemingway	Adventure	1	Physical

4.3.2 ΠΙΝΑΚΑΣ “userbase”

Ένας εξίσου σημαντικός πίνακας για την εφαρμογή είναι ο πίνακας “userbase”. Στον πίνακα “userbase” αποθηκεύονται όλοι οι χρήστες μαζί με τα χαρακτηριστικά τα οποία είναι απαραίτητα για την χρήση της πλατφόρμας καθώς και την ταυτοποίησή τους. Αυτα είναι το όνομα, το email, ο κωδικός, η περιοχή η οποία έχουν δηλώσει και οι φυσικοί πόντοι και οι διαδικτυακοί πόντοι οι οποίοι έχουν.

Δυο χαρακτηριστικά που επίσης έχουν τοποθετηθεί σε αυτόν τον πίνακα είναι το αν ο χρήστης είναι η όχι διαχειριστής, το οποίο ορίζει την διεπαφή την οποία θα δει και τις μέρες για τις οποίες για τις οποίες ο χρήστης έχει αποκλειστεί. Αν ο αριθμός είναι 0, ο χρήστης έχει κανονική πρόσβαση στην εφαρμογή, αν είναι παραπάνω η πρόσβαση του απαγορεύεται. Ένα επίσης ενδιαφέρον χαρακτηριστικό που γίνεται εμφανές είναι πως οι κωδικοί είναι σε μορφή συμβολοσειράς hash. Αυτό γίνεται γιατί οι κωδικοί κρυπτογραφούνται πριν εισέλθουν στην βάση δεδομένων, αναφέρομαι εκτενέστερα στο κεφάλαιο που εξηγώ το back-end της εφαρμογης.

user_id	user_name	user_email	user_pass	user_area	user_ph_points	user_pdf_points	isadmin	bandays
2	admin	learn2earnproj@gmail.com	$2b$10$kSIV7/DFcmA0aK4rrsu4DuGUg0da4yvvg099NvyEduCQK1LPm7DAC	Athens	3	3	t	0
5	test3	test3@gmail.com	$2b$10$UzZ9BZ/Ai0du2kmoIwVYxutw03fp/tAiL43.pf9Ef8zRXOzrVCKCm	Chalcis	3	3	f	0
4	test2	test2@gmail.com	$2b$10$LIJ1IMGYdhRO0sVWPw7dkOm57mVbOSr2g/5j0fYaBe01.ah5ZafXm	Athens	3	4	f	0
1	trikone	trikprd@gmail.com	$2b$10$2/w.aGliRrWf.apwn5sHI.cT6lSem5fd8frNsuoTb8T8RyJlVURfu	Athens	3	2	f	0
3	test1	test1@gmail.com	$2b$10$Mtya6CACwMS2oHgdz2uEdusqCR7088WYUgvcF8Qc3vCS47Hq5DA8a	Athens	2	2	f	0

(5 rows)

4.3.3 ΠΙΝΑΚΑΣ “booksentry”

Οι υπόλοιποι πίνακες έχουν κυρίως να κάνουν με συσχετίσεις στοιχείων απο αυτούς τους δύο. Ένα βασικό χαρακτηριστικό το οποίο χρειάζεται για την

υλοποίηση της εφαρμογής είναι τα βιβλία τα οποία έχει ο κάθε χρήστης. Αυτό αντιπροσωπεύει ο πίνακας “booksentry”. Ο πίνακας “booksentry” ουσιαστικά είναι μία συσχέτιση του αναγνωριστικού βιβλίων “b_id” που αντιστοιχει στο “book_id” του πίνακα “books” και του αναγνωριστικού χρήστη “u_id” που αντιστοιχει στο “user_id” του πίνακα “userbase”. Κάθε γραμμή ουσιαστικά δίνει την πληροφορία: Αυτός ο χρήστης εχει αυτό το βιβλίο.

entry_id	b_id	u_id
1	1	1
2	2	1
3	3	2
5	5	4
6	1	5
7	2	3

Η λειτουργικότητα αυτού καθώς και των άλλων δύο πινάκων γίνεται πιο εμφανής με την χρήση της λειτουργίας JOIN που μας παρέχει η γλώσσα SQL. Αν εφαρμόοουμε την εντολή:

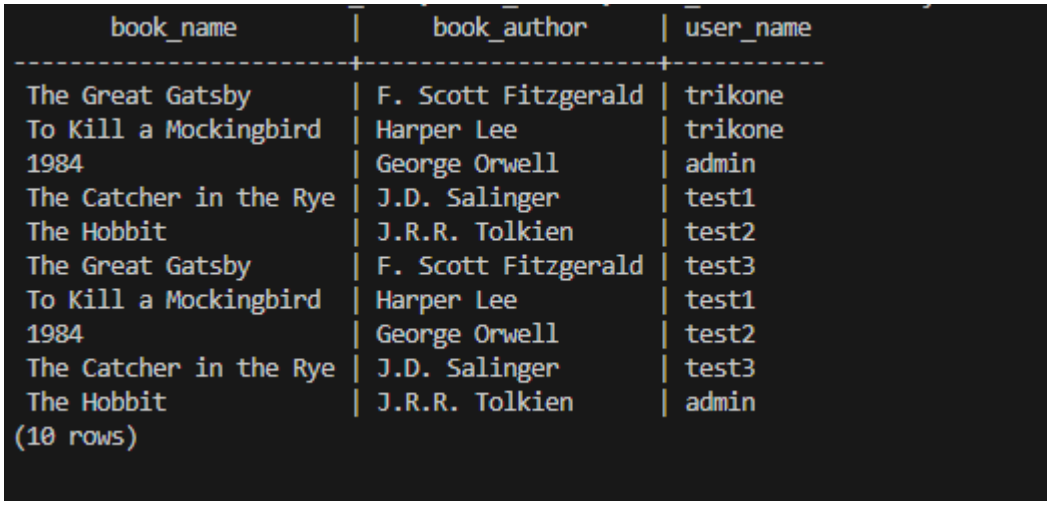
```
   SELECT book_name, book_author, user_name FROM booksentry JOIN userbase ON
booksentry.u_id = userbase.user_id JOIN books ON booksentry.b_id =
books.book_id;
```

Ο πίνακας που παράγεται είναι ο εξής:

book_name	book_author	user_name
The Great Gatsby	F. Scott Fitzgerald	trikone
To Kill a Mockingbird	Harper Lee	trikone
1984	George Orwell	admin
The Hobbit	J.R.R. Tolkien	test2
The Great Gatsby	F. Scott Fitzgerald	test3
To Kill a Mockingbird	Harper Lee	test1
1984	George Orwell	test2

4.3.4 ΠΙΝΑΚΑΣ “history”

Ο πίνακας “history” ακολουθεί ακριβώς την ίδια δομή με τον πίνακα “booksentry”. Η διαφορά τους βρίσκεται στην χρήση, μια γραμμή στον πίνακα “history” ουσιαστικά αντιπροσωπεύει την πληροφορία: “Αυτός ο χρήστης είχε αυτό το βιβλίο”. Είναι ένας τρόπος αρχειοποίησης του κάθε χρήστη και μας επιτρέπει να λάβουμε πληροφορίες όπως τα ενδιαφέροντα και την δραστηριότητά του. Μετά απο μία παρόμοια εντολή SQL με αυτήν που αναφέρθηκε ο πίνακας “history” φαίνεται έτσι:



book_name	book_author	user_name
The Great Gatsby	F. Scott Fitzgerald	trikone
To Kill a Mockingbird	Harper Lee	trikone
1984	George Orwell	admin
The Catcher in the Rye	J.D. Salinger	test1
The Hobbit	J.R.R. Tolkien	test2
The Great Gatsby	F. Scott Fitzgerald	test3
To Kill a Mockingbird	Harper Lee	test1
1984	George Orwell	test2
The Catcher in the Rye	J.D. Salinger	test3
The Hobbit	J.R.R. Tolkien	admin

(10 rows)

4.3.5 ΠΙΝΑΚΑΣ “requests”

Ο λόγος ο οποίος υλοποιήθηκε αυτή η εφαρμογή είναι για την διαδικασία της ανταλλαγής βιβλίων. Αυτή η διαδικασία απαιτεί μια αλληλεπίδραση μεταξύ δυο χρηστών. Ένας χρήστης πρέπει να ζητήσει κάποιο βιβλίο από έναν άλλον και ένας χρήστης πρέπει να αποδεχτεί αυτή την αίτηση. Η υλοποίηση αυτής της διαδικασίας γίνεται στο API, παρόλαυτα τα δεδομένα των αιτήσεων, πρέπει να αποθηκεύονται. Αυτή την ανάγκη εξυπηρετεί ο πίνακας “requests”.

Μια γραμμή του πίνακα “requests” ουσιαστικά αντιπροσωπεύει την εξής πληροφορία: Αυτο το βιβλίο του χρηστη Α ζητήθηκε από τον χρήστη Β. Τα χαρακτηριστικά που αναφέρει αυτός ο πίνακας είναι το αναγνωριστηκό αυτού που ζητάει το βιβλίο (requester_id-αντιστοιχεί στο user_id του “userbase”), το αναγνωριστικό αυτού που το προσφέρει (offerer_id -αντιστοιχεί στο user_id του “userbase”) και το αναγνωριστικο του βιβλίου (b_id-αντιστοιχεί στο book_id του “books”). Με την εξής πράξη JOIN της γλώσσας SQL:

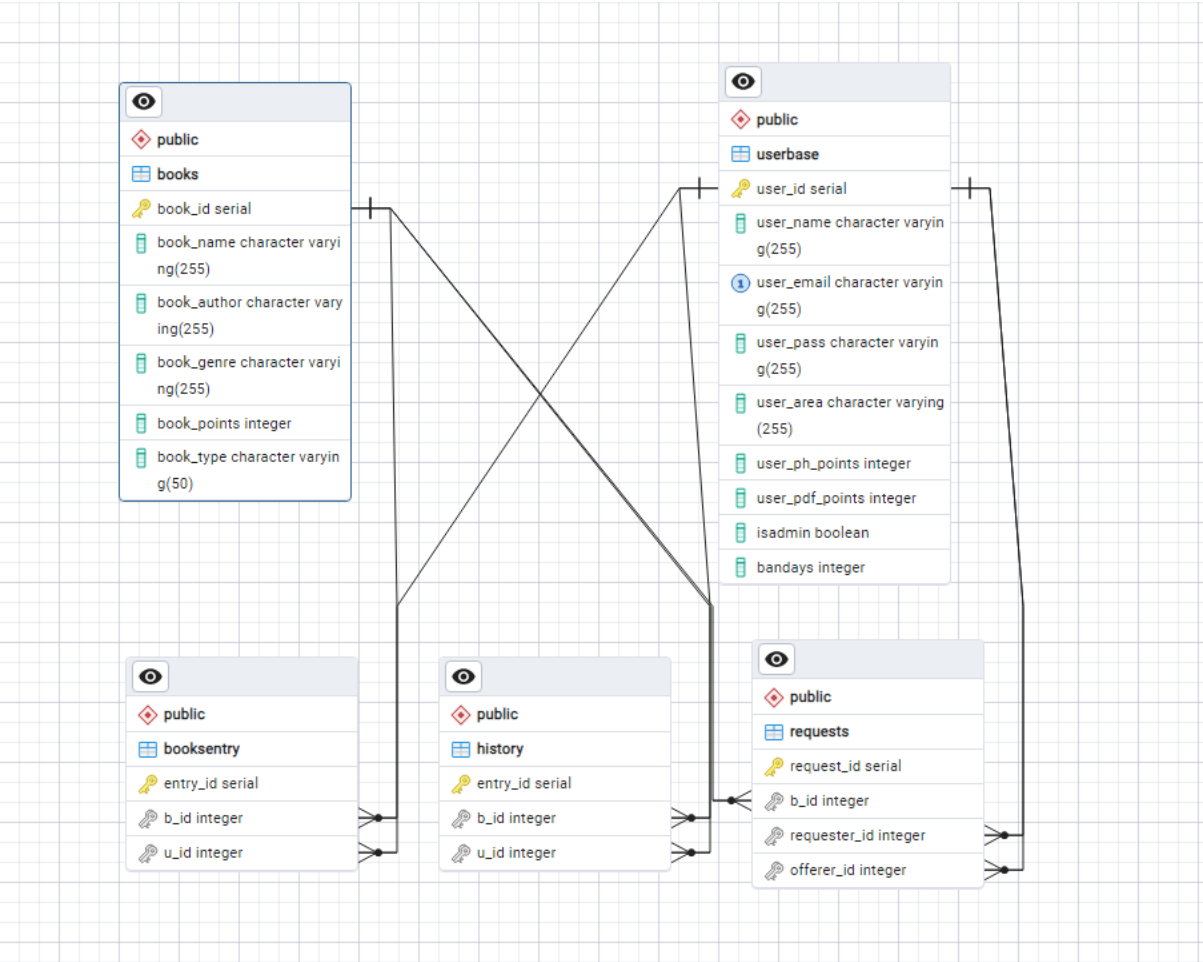
```
   SELECT requester.user_name AS requester_name, offerer.user_name AS
offerer_name , book_name FROM requests r JOIN userbase requester ON
r.requester_id = requester.user_id JOIN userbase offerer ON r.offerer_id =
offerer.user_id JOIN books ON b_id=books.book_id;
```

Ο πίνακας που παράγεται είναι ο εξής:

```
      book_name        |     book_author     | user_name
-----------------------+---------------------+-----------
 The Great Gatsby      | F. Scott Fitzgerald | trikone
 To Kill a Mockingbird | Harper Lee          | trikone
 1984                  | George Orwell       | admin
 The Catcher in the Rye | J.D. Salinger      | test1
 The Hobbit            | J.R.R. Tolkien      | test2
 The Great Gatsby      | F. Scott Fitzgerald | test3
 To Kill a Mockingbird | Harper Lee          | test1
 1984                  | George Orwell       | test2
 The Catcher in the Rye | J.D. Salinger      | test3
 The Hobbit            | J.R.R. Tolkien      | admin
(10 rows)
```

4.3.6 ΣΧΗΜΑ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Παρακάτω παρουσιάζω το ολοκλήρωμενο Σχήμα βάσης δεδομένων (Database Schema) της εφαρμογής μου.



4.4 ΣΧΕΔΙΑΣΜΟΣ APPLICATION PROGRAMMING INTERFACE (API)

4.4.1 ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ EXPRESS.JS

Για τον σχεδιασμό του API χρησιμοποιήθηκε το framework Express.js. Ένα πολύ θετικό χαρακτηριστικό του Express.js είναι ο μινιμαλισμός ο οποίος παρέχει, κάτι το οποίο κάνει τον σχεδιασμό του API αρκετά πιο εύκολο.

Στο Express.js ορίζουμε κάποια routes τα οποία μας επιτρέπουν να κατηγοριοποιούμε σωστά τις κλήσεις τις οποίες επιτρέπουμε. Για παράδειγμα οι κλήσεις που έχουν να κάνουν με βιβλία, πηγαίνουν στο route “books” το οποίο ορίζουμε έτσι:

```
//books route

app.use("/books", require("./routes/booklists"));
```

Πολύ χρήσιμα είναι επίσης και τα middleware, τα κομμάτια κώδικα δηλαδή που μεσολαβούν μεταξύ της επικοινωνίας του Client και του Server.

Στην εφαρμογή έχω χρησιμοποιήσει τα:

```
// middleware
app.use(express.json())
app.use(cors())
```

Το express.json επιτρέπει τη μοντελοποίηση των δεδομένων σε JavaScript Object Notation (JSON) και το cors επιτρέπει την επικοινωνία μεταξύ του Client και του Server.

Δυο middleware τα οποία υλοποίησα ο ίδιος και είναι πολύ χρήσιμα για την εφαρμογή είναι τα authorization.js και validInfo.js. Το authorization.js θα παρουσιαστεί αργότερα, το validInfo.js χρησιμοποιείται για την επιβεβαίωση της ύπαρξης και της μορφής των στοιχείων του χρήστη. Το validInfo.js καλύπτει πολλά σενάρια, όπως για παράδειγμα η επιβεβαίωση της σωστής μορφής e-mail με τη χρήση Κανονικών Εκφράσεων (RegEx):

```
function validEmail(userEmail) {
        return /^\w+([\.-]?\w+)*@\w+([\.-
]?\w+)*(\.\w{2,3})+$/.test(userEmail);
      }
```

Τέλος πριν την αρχή σχεδίασης των κλήσεων, πρέπει να εδραιωθεί μια σύνδεση με την βάση. Αυτό γίνεται στο αρχείο db.js:

```
const Pool = require("pg").Pool

const pool = new Pool({
    user: 'postgres',
    password: 'superuser',
    host: 'localhost',
    port: 5432,
    database: 'learn2earn'
```

```
});

module.exports = pool;
```

Το οποίο μου επιτρέπει να εκτελώ εντολές SQL (Sequential Query Language), χρησιμοποιώντας το Pool module που ορίζω εδώ.

4.4.2 ΔΟΜΗ ΚΛΗΣΕΩΝ

Τα δεδομένα που επιστρέφει το API είναι διαφόρων μορφών και καλύπτουν όλες τις ανάγκες της εφαρμογής. Η δομή μιας κλήσης API παρόλαυτα είναι η ίδια σε όλα. Θα την εξηγήσω χρησιμοποιώντας το παρακάτω παράδειγμα:

```
router.get("/getusers", async (req, res) => {
  try {
    users = await pool.query(
      "SELECT user_id,user_name,user_email,user_area, user_pdf_points,
user_ph_points,isadmin,bandays FROM userbase;"
    );
    res.send(users.rows);
  } catch (error) {
    res.send(`Error getting users: ${error}`);
  }
});
```

Όπως έχουμε αναφέρει ήδη στο Express.js ορίζουμε μια μέθοδο από τις βασικές μεθόδους κλήσης ενός REST API. Αυτές είναι οι: GET, PUT, POST και DELETE. Χρησιμοποιώντας μια από αυτές τις τέσσερις μεθόδους μπορώ να λάβω, να στείλω και να επεξεργαστώ δεδομένα.

Έπειτα πρέπει να οριστεί κάποιο συγκεκριμένο route, εδώ χρησιμοποιούμε το “/getusers”, και βρίσκεται μέσα στο αρχείο που έχουμε ορίσει ως “/dashboard”. Άρα το τελικό route είναι (διεύθυνση της ιστοσελίδας)/dashboard/getusers.

Το επόμενο βήμα είναι η επικοινωνία με την βάση δεδομένων. Αυτό γίνεται μέσω της γλώσσας SQL και του Pool module. Εδώ χρησιμοποιούμε την εντολή:

```
SELECT user_id,user_name,user_email,user_area, user_pdf_points,
user_ph_points,isadmin,bandays FROM userbase;
```

Η οποία επιλέγει όλα τα χαρακτηριστικα των χρηστών της εφαρμογής και τα επιστρέφει στον Client. Αν υπάρχει κάποιο σφάλμα το API επιστρέφει το ανάλογο μήνυμα στον Client.

4.4.3 JWT TOKENS: ΑΥΘΕΝΤΙΚΟΠΟΙΗΣΗ ΚΑΙ ΔΙΑΧΕΙΡΗΣΗ ΣΥΝΕΔΡΙΩΝ

Μια από τις πιο βασικές λειτουργίες που πρέπει να υποστηρίζει η εφαρμογή μας είναι σύστηματα χρηστών. Ένας χρήστης πρέπει να μπορεί να πληκτρολογήσει τα στοιχεία ταυτοποίησής (το όνομα και τον κωδικό) του στην εφαρμογή και αυτή να μπορεί να επιβεβαιώσει πως τα στοιχεία αυτά είναι υπαρκτά. Αυτή η διαδικασία λέγεται Αυθεντικοποίηση. Επίσης, πρέπει να μπορεί να κρατάει τον χρήστη Αυθεντικοποιημένο για όλο το διάστημα το οποίο κρατάει ανοιχτή την εφαρμογή. Αυτή η διαδικασία λέγεται Διαχείριση Συνεδρίας (Session Control).

Για την υλοποίηση αυτών των λειτουργιών χρησιμοποιούμε το JWT Token. Το JWT Token παράγει ένα Token, δηλαδή, μια κρυπτογραφημένη συμβολοσειρά τύπου hash κάθε φορά που ένας χρήστης αυθεντικοποιείται στην εφαρμογή. Αυτό το υλοποιούμε στο αρχείο jwtAuth.js στο route “/login”.

```
router.post("/login", validInfo, async (req, res) => {
  try {
    const { username, password } = req.body;

    const user = await pool.query(
      "SELECT * FROM userbase WHERE user_name = $1",
      [username]
    );

    if (user.rows.length == 0) {
      res.status(401).json("Password or User Name is incorrect");
      return;
    }

    (Διαφορες λογικες διαδικασίες)

    const token = jwtGenerator(user.rows[0].user_id);
    res.json({ token });
  } catch (err) {
    console.error(err.message);
    res.status(500).send("Server Error");
  }
});
```

Ο χρήστης έπειτα χρησιμοποιεί αυτό το Token για να αυθεντικοποιήσει τον εαυτό του κάθε φορά που ανακατευθύνεται σε διαφορετική σελίδα της εφαρμογής. Αυτό γίνεται με το middleware authorization.js, το οποίο καλείται σε διάφορες κλήσεις του API.

```
module.exports = function(req, res, next) {
  const token = req.header("token");

  if (!token) {
    return res.status(403).send("authorization denied");
```

```
    }

    try {
      const verify = jwt.verify(token, process.env.jwtSecret);

      req.user = verify.user;
      next();
    } catch (err) {
      res.status(401).json({verify:false});
    }
  };
```

Να σημειωθεί σε αυτό το σημείο, πως το χρονικό όριο του Token είναι μία ώρα, έπειτα ο χρήστης πρέπει να ξανασυνδεθεί. Αυτός ο αριθμός μπορεί να αλλάξει ανάλογα με τις ανάγκες της εκάστοτε εφαρμογής.

4.4.4 BCRYPT: ΚΡΥΠΤΟΓΡΑΦΗΣΗ ΚΩΔΙΚΩΝ

Το πιο ευαίσθητο προσωπικό στοιχείο ενός χρήστη είναι ο κωδικός του και αυτό γιατί ένας κωδικός πρόσβασης είναι το μόνο στοιχείο το οποίο δεν είναι δημόσιο στην εφαρμογή και αν ένας χρήστης το γνωρίζει μπορεί εύκολα να αποκτήσει πρόσβαση στον λογαριασμό του.

Αυτό το κάνει πολύ σημαντική ευπάθεια ασφαλείας (Security Vulnerability) και κρίνεται απαραίτητο να βρεθεί ένας τρόπος ασφαλής αποθήκευσης και θωράκισης των κωδικών. Αυτό ακριβώς το πρόβλημα λύνει η κρυπτογράφηση κωδικών με την βιβλιοθήκη BCrypt.

Η βιβλιοθήκη BCrypt επιτρέπει κρυπτογράφηση μιας κατεύθυνσης με αλγόριθμο τύπου hash. Αυτό σημαίνει ότι ένας κωδικός ο οποίος κρυπτογραφείται δεν μπορεί να αποκρυπτογραφηθεί. Κάτι το οποίο διασφαλίζει τους κωδικούς ακόμη και αν αυτοί διαρρεύσουν.

```
router.post("/register", validInfo, async (req, res) => {
  try {
    const { email, username, password, city } = req.body;
```

```
(Λογική κλήσης)
    const saltRound = 10;
    const Salt = await bcrypt.genSalt(saltRound);
    const bcryptPassword = await bcrypt.hash(password, Salt);
```

Ειδικές συναρτήσεις για τη σύγκριση κωδικών παρέχονται επίσης, έτσι ώστε να είναι δυνατή η διαδικασία της σύνδεσης.

```
    const validPassword = await bcrypt.compare(
      password,
      user.rows[0].user_pass
    );
```

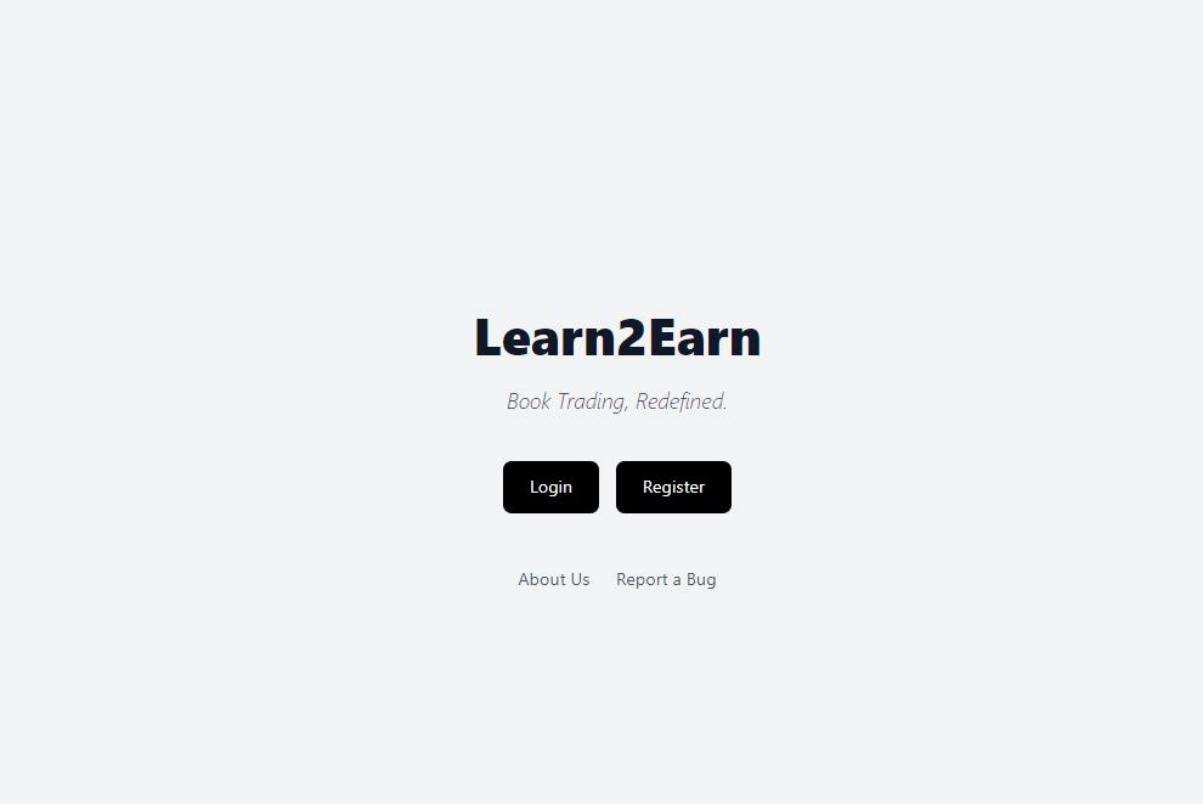
Η βιβλιοθήκη BCrypt προτιμάται πολύ συχνά για την ανάπτυξη διαδικτυακών εφαρμογών λόγω της ευκολίας και του υψηλού επιπέδου ασφάλειας του οποίου οποίου παρέχει.

```
|                          user_pass                          |
+-------------------------------------------------------------+
| $2b$10$kSIV7/DFcmA0aK4rrsu4DuGUg0da4yvvg099NvyEduCQK1LPm7DAC |
| $2b$10$UzZ9BZ/Ai0du2kmoIwVYxutw03fp/tAiL43.pf9Ef8zRXOzrVCKCm |
| $2b$10$2/w.aGliRrWf.apwn5sHI.cT6lSem5fd8frNsuoTb8T8RyJlVURfu |
| $2b$10$Mtya6CACwMS2oHgdz2uEdusqCR7088WYUgvcF8Qc3vCS47Hq5DA8a |
| $2b$10$LIJ1IMGYdhRO0sVWPw7dkOm57mVbOSr2g/5j0fYaBe01.ah5ZafXm |
```

5. ΕΠΙΔΕΙΞΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ (ΟΔΗΓΙΕΣ ΧΡΗΣΗΣ)

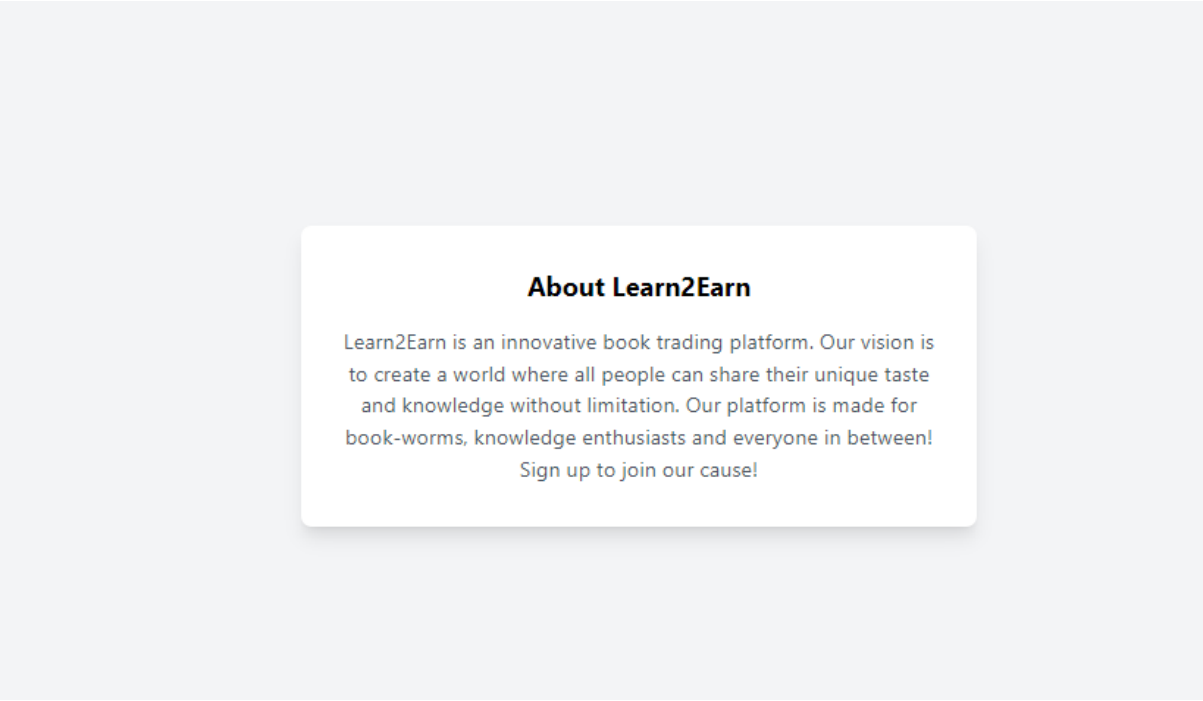
5.1 ΑΡΧΙΚΗ ΟΘΟΝΗ

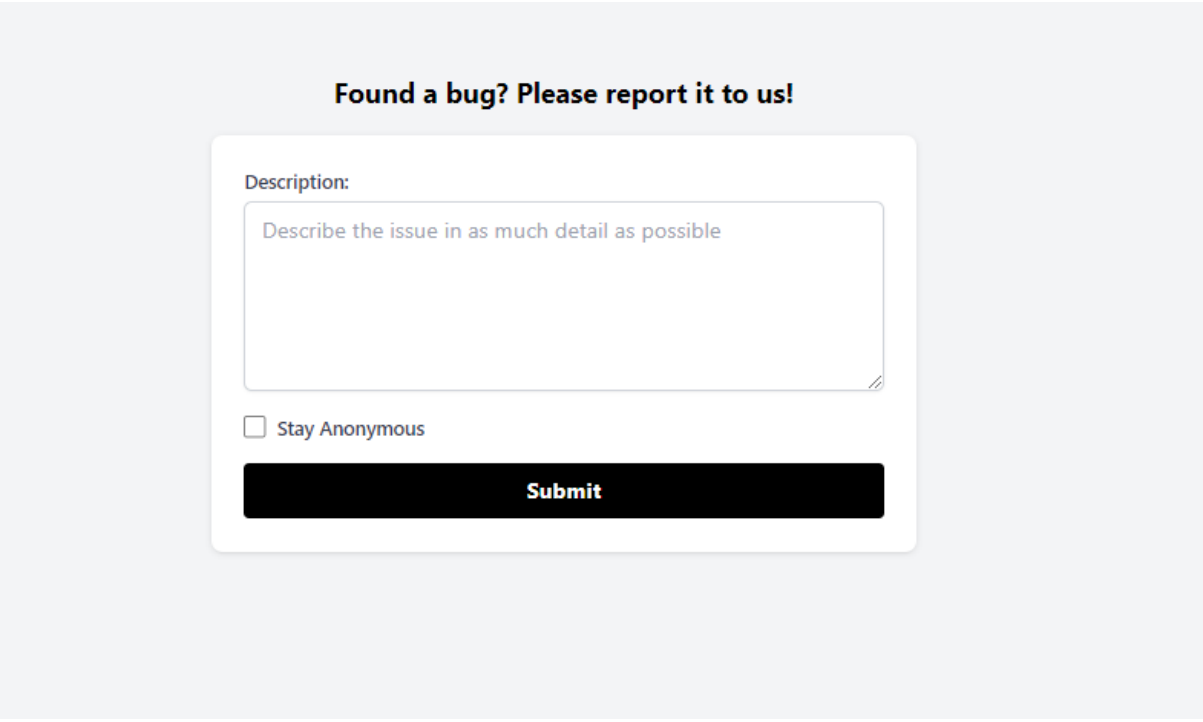
Η αρχική οθόνη που βλέπει ένας χρήστης όταν συνδέεται στην εφαρμογή είναι αυτή:



Οι τέσσερις επιλογές που μας δίνονται είναι η Σύνδεση η εγγραφή καινούργιου λογαριασμού, η αναφορά ενός σφάλματος της σελίδας καθώς και η προβολή μιας σελίδας με πληροφορίες αναφορικά με την εφαρμογή.

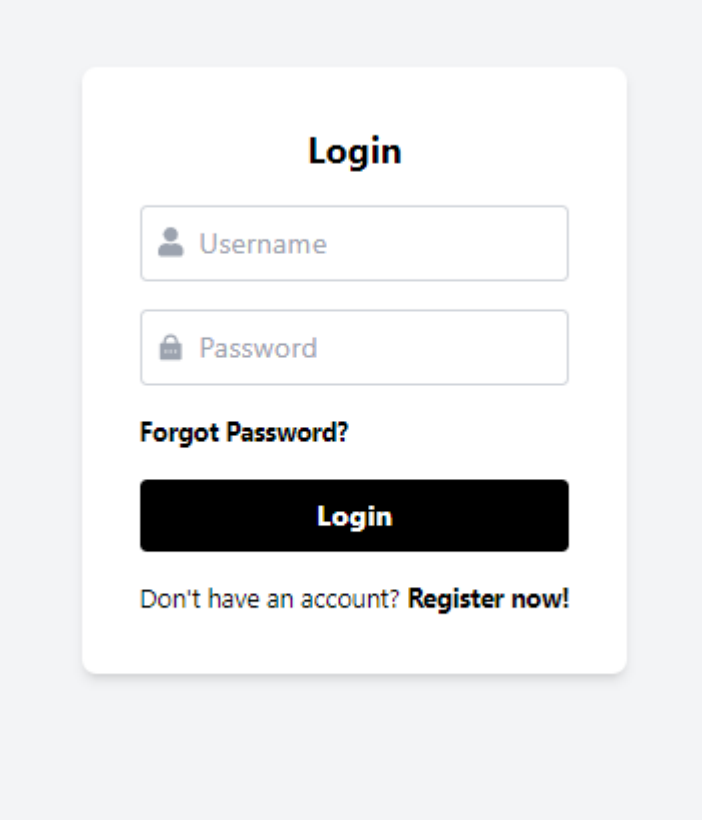
Οι δύο τελευταίες είναι οι εξής:



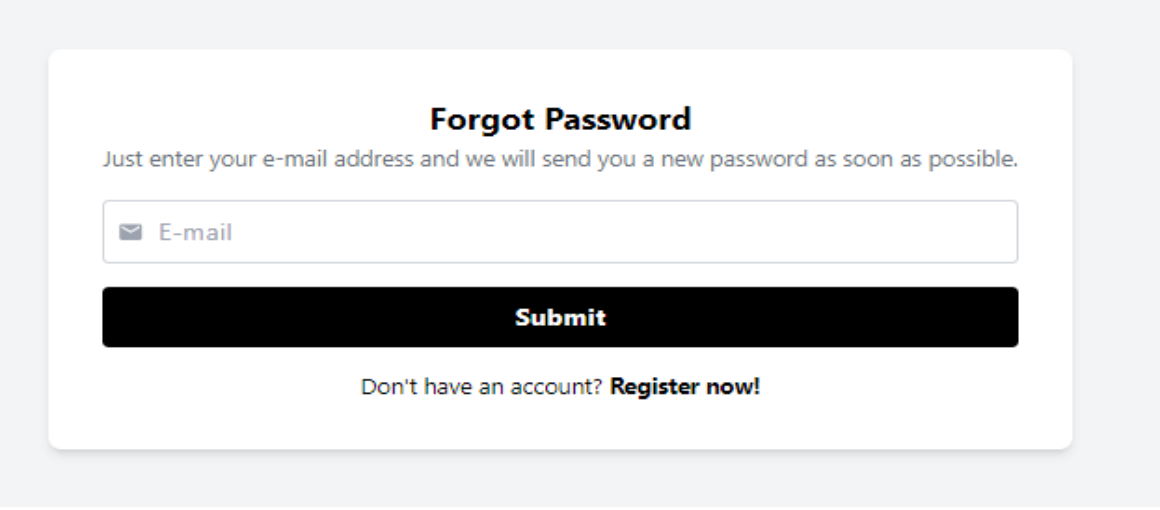


Οι αναφορές των “bugs” στέλνονται αυτόματα σε ένα προ-ρυθμιζμένο e-mail.

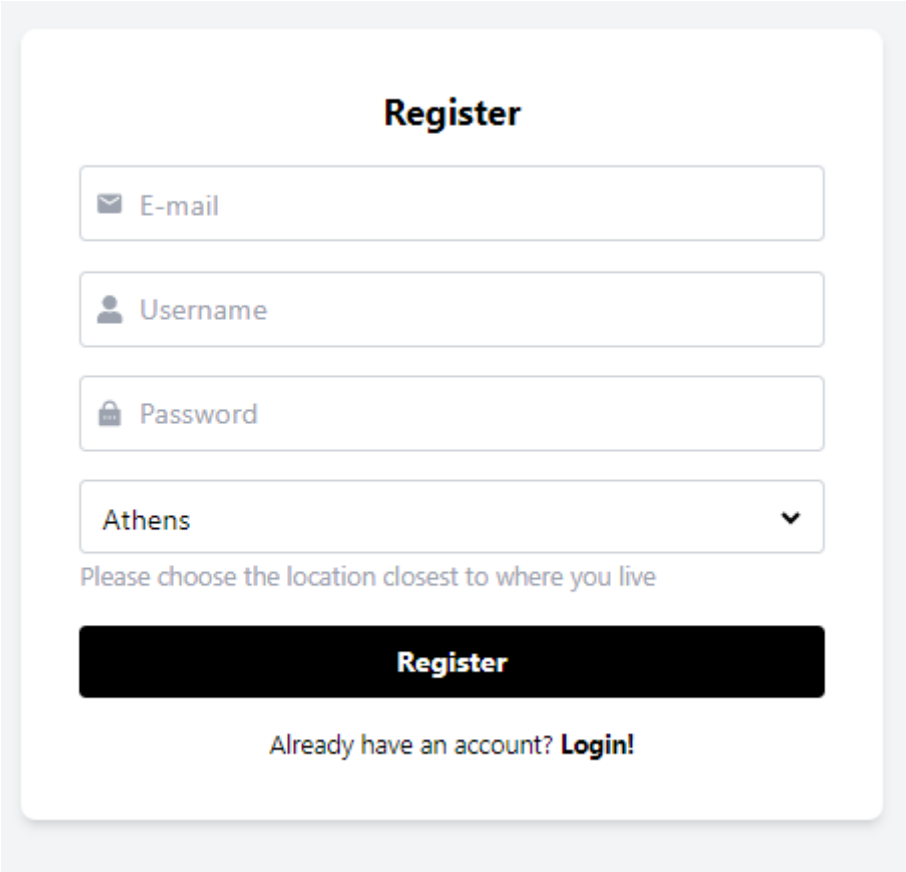
Με την επιλογή της σύνδεσης ο χρήστης ανακατευθύνεται σε αυτή την σελίδα:



Στην οποία καλείται να βάλει το όνομα και τον κωδικό πρόσβασης του, του δίνεται επίσης η επιλογή επαναφοράς του κωδικού του σε περίπτωση που τον έχει ξεχάσει καθώς και εγγραφής σε περίπτωση που δεν έχει λογαριασμό. Στην πρώτη περίπτωση ο χρήστης καλείται να δώσει το email του, και αν βρεθεί λογαριασμος με αυτό τότε του έρχεται ένας καινούργιος κωδικός στη ηλεκτρονική διεύθυνσή του. Σε οποιαδήποτε άλλη περίπτωση του έρχεται ειδοποίηση σφάλματος.



Με την επιλογή της σύνδεσης ο χρήστης ανακατευθύνεται σε αυτή την σελίδα:



Όπου τοποθετώντας τα στοιχεία που του ζητούνται ο χρήστης δημιουργεί έναν καινούργιο λογαριασμό. Αν για οποιοδήποτε λόγο η δημιουργία λογαριασμού αποτύχει, ο χρήστης θα λάβει ειδοποίηση σφάλματος. Αν η δημιουργία λογαριασμού επιτύχει, ο χρήστης θα λάβει ειδοποίηση επιτυχίας. Έπειτα, θα πρέπει να συνδεθεί στον λογαριασμό.

5.2 ΠΕΡΙΓΡΑΦΗ ΔΙΑΔΙΚΑΣΙΑΣ ΑΝΤΑΛΛΑΓΗΣ

Πρώτου προχωρήσουμε στη βασική λειτουργικότητα της εφαρμογής πρέπει να γίνει μια αναφορά στη διαδικασία με την οποία οι χρήστες ανταλλάζουν βιβλία,

καθώς είναι βασική για τη κατανόηση αλλά και τη σωστή χρήση της εφαρμογής.

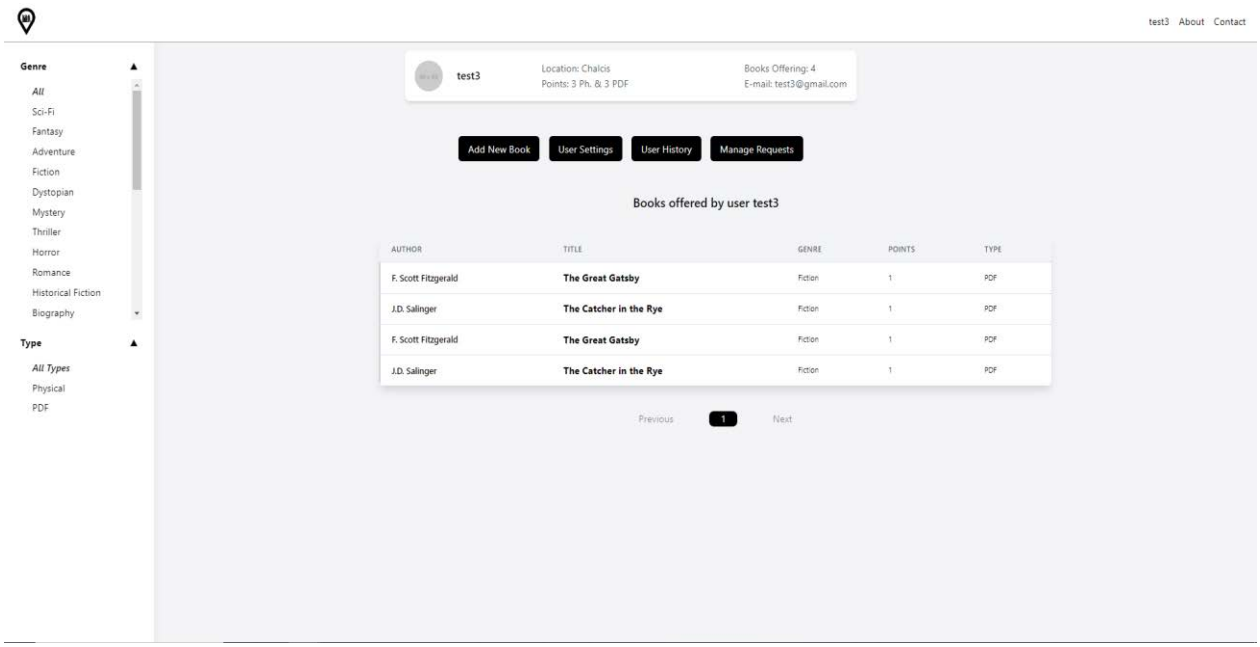
Ο κάθε χρήστης έχει κάποιους πόντους, που διαχωρίζονται σε φυσικούς (Physical) η ψηφιακούς (PDF). Όταν ο χρήστης εγγράφεται στην πλατφόρμα ξεκινάει με 3 φυσικούς και 3 PDF πόντους. Κάθε βιβλίο “κοστίζει” κάποιους πόντους ανάλογα με τον τύπο του. Τα βιβλία όταν εισάγονται, κατά προεπιλογή, κοστίζουν 1 πόντο (είτε φυσικό είτε PDF). Ένας διαχειριστής μπορεί να αλλάξει αυτή τη τιμή.

Αν ένας χρήστης έχει αρκετούς πόντους μπορεί να κάνει αίτηση για να λάβει ένα βιβλίο, συμπληρώνοντας τα στοιχεία του. Όταν ο χρήστης κάνει κάποια αίτηση, αυτή φαίνεται στη σελίδα προφίλ του χρήστη που προσφέρει το εν λόγω βιβλίο. Έπειτα ο χρήστης ο οποίος το προσφέρει μπορεί να δεχτεί η να αρνηθεί την αίτηση. Αν ο χρήστης δεχτεί την αίτηση του στέλνεται ένα e-mail στη ηλεκτρονική διεύθυνση που έχει δηλώσει, το οποίο τον ενημερώνει για τη διεύθυνση του χρήση που έκανε την αίτηση. Επίσης του προστίθενται στο λογαριασμό του οι πόντοι οι οποίοι “κοστίζει” το βιβλίο, ο ίδιος αριθμός πόντων αφαιρούνται από τον χρήστη που έκανε την αίτηση. Ο χρήστης που προσφέρει το βιβλίο είναι υποχρεωμένος εν τέλει να στείλει το βιβλίο στο χρήστη τον οποίο το αιτήθηκε. Τέλος, το βιβλίο είτε διαγράφεται από την πλατφόρμα αν είναι φυσικού τύπου, αν είναι ψηφιακού τύπου παραμένει.

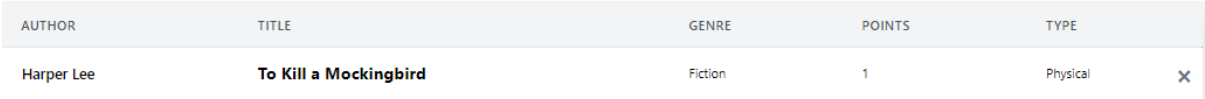
5.3 ΔΙΕΠΑΦΗ ΧΡΗΣΤΗ

5.3.1 ΣΕΛΙΔΑ ΠΡΟΦΙΛ

Αφότου ο χρήστης δημιουργήσει η συνδεθεί στον λογαριασμό του, ανακατευθύνεται στη σελίδα προφιλ.

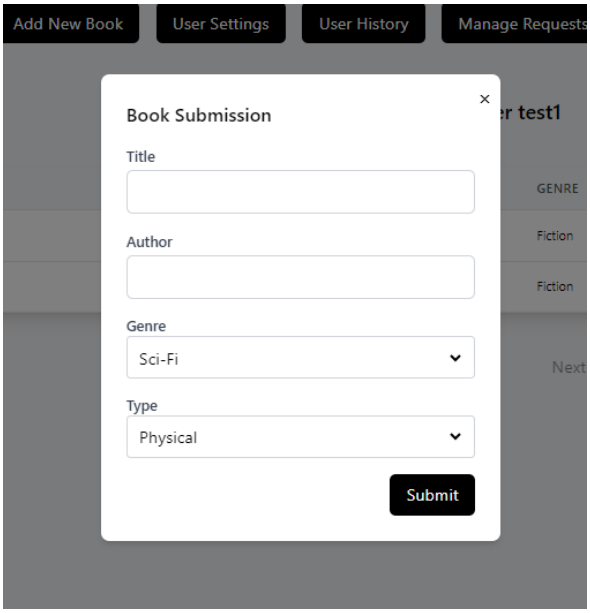


Απο την οποία διαχειρίζεται το προσωπικό του προφίλ. Στο κέντρο υπάρχει ένα ItemList Component που περιέχει όλα τα βιβλία που ο χρήστης προσφέρει για ανταλλαγή. Με την τοποθέτηση του κέρσορα πάνω από ένα βιβλίο εμφανίζεται ένα κουμπί διαγραφής βιβλίου από την πλατφόρμα.



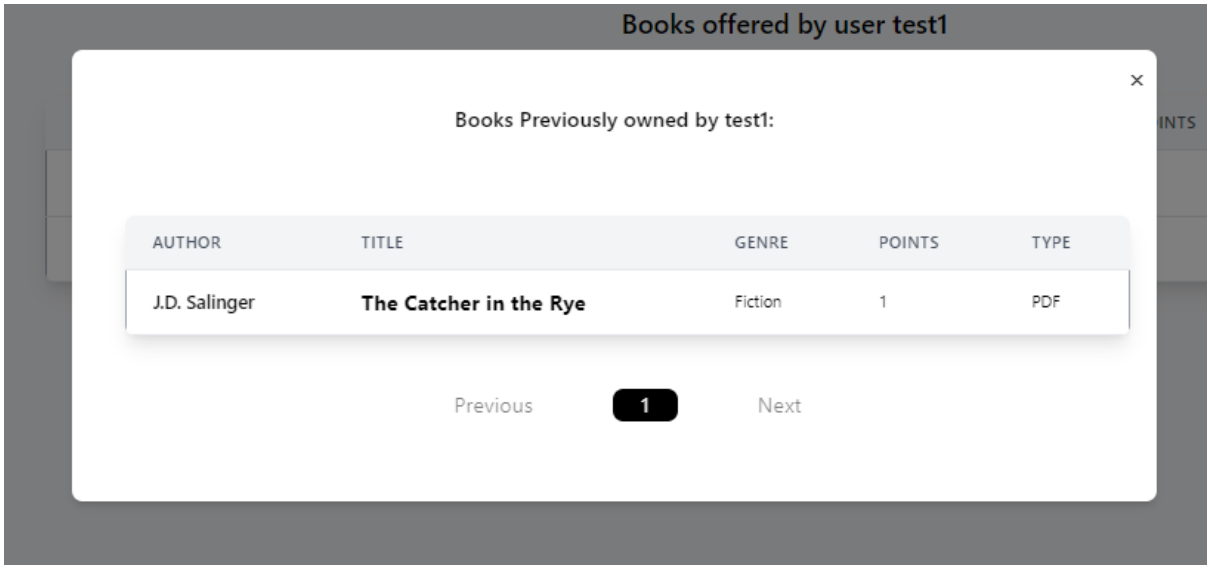
Στο πάνω μέρος της σελίδας υπάρχουν κάποιες βασικές πληροφορίες για τον χρήστη όπως ο αριθμός των βιβλίων που προσφέρει, την περιοχή, τους πόντους και το email του. Λίγο πιο πάνω υπάρχει το Navbar Component μέσα από το οποίο ο χρήστης μπορεί να περιηγηθεί στην εφαρμογή. Στα αριστερά υπάρχει το Sidebar Component που περιέχει φίλτρα για την διευκόλυνση αναζήτησης κάποιου βιβλίου.

Κάτω απο τις πληροφορίες υπάρχουν τέσσερα κουμπιά τα οποία προσφέρουν τις βασικές επιλογές που μπορεί ένας χρήστης να χρειαστεί. Κάνοντας κλικ στο κουμπί “Add New Book”, εμφανίζεται αυτή η επικάλυψη (overlay).



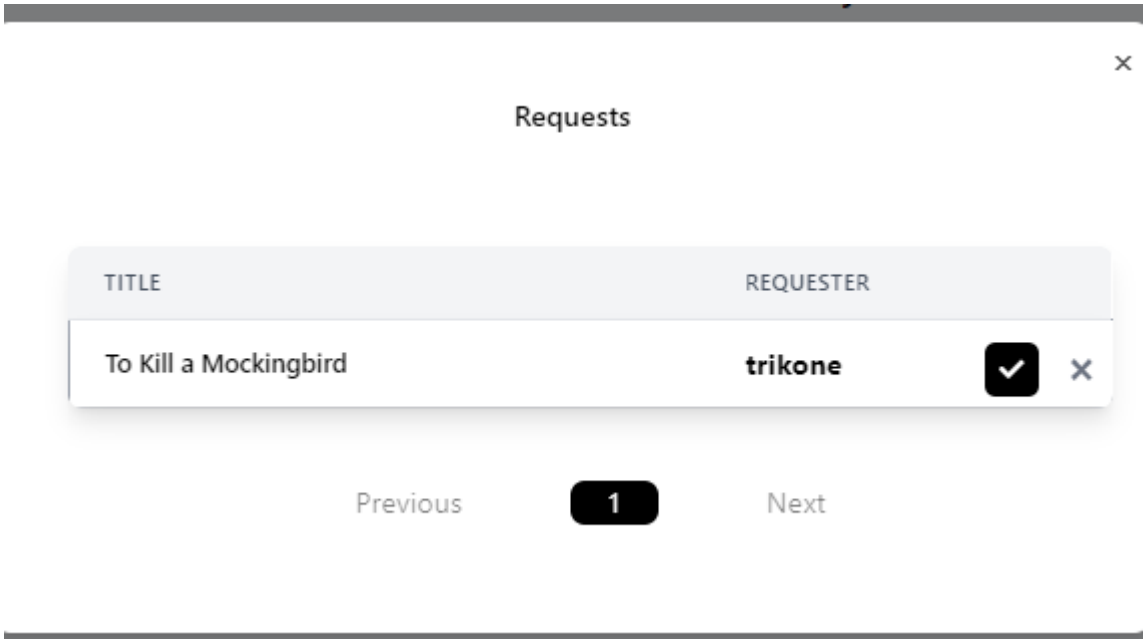
Στην οποία ένας χρήστης, δίνοντας τις απαραίτητες πληροφορίες, μπορεί να προσθέσει ένα καινούργιο βιβλίο το οποίο προσφέρει. Όπως προειπώθηκε, το βιβλίο κατά προεπιλογή εκχωρείται με έναν πόντο, ο οποίος προστίθεται και στον χρήστη.

Κάνοντας κλικ στο κουμπί “User History” εμφανίζεται η εξής επικάλυψη:



Η οποία παρουσιάζει όλα τα βιβλία τα οποία είχε στο παρελθόν ο χρήστης.

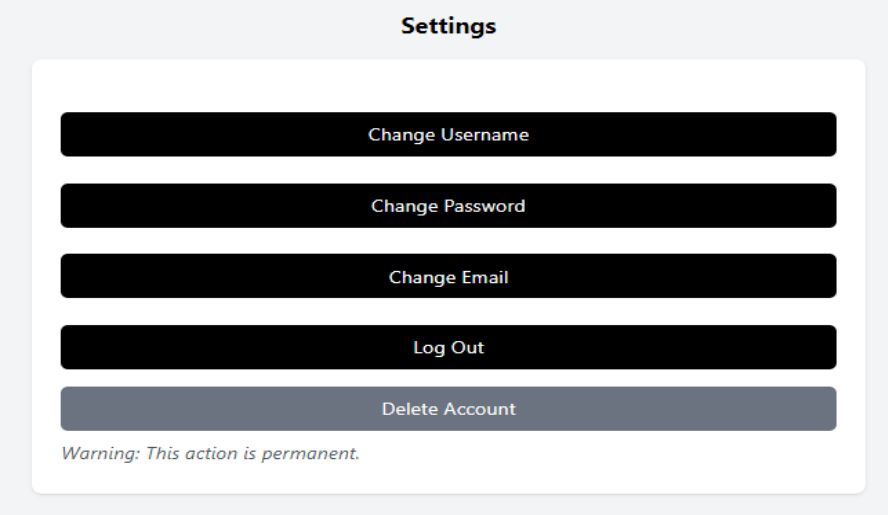
Κάνοντας κλικ στο κουμπί “Manage Requests” εμφανίζεται η εξής επικάλυψη:



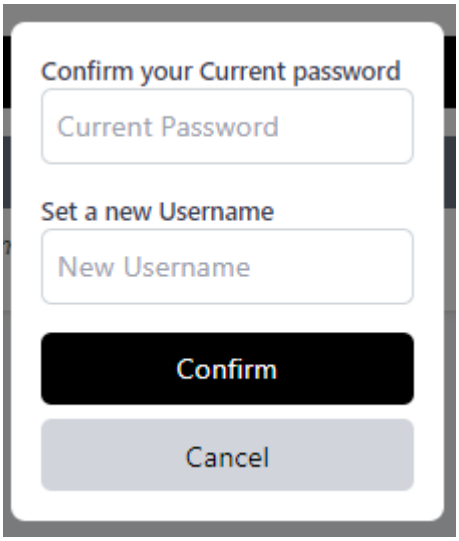
Στο οποίο φαίνονται οι αιτήσεις που έχουν γίνει για βιβλία στον χρήστη καθώς και οι χρήστες οι οποίοι τα αιτούνται. Τοποθετώντας τον κέρσορα επάνω από το αντικείμενο εμφανίζονται τα κουμπιά της αποδοχής και της ακύρωσης της αίτησης.

5.3.2 ΡΥΘΜΙΣΕΙΣ ΧΡΗΣΤΗ

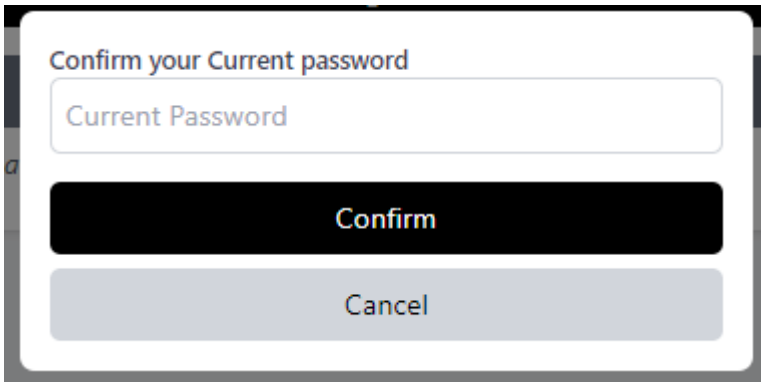
Κάνοντας κλικ στο κουμπί “User Settings” ο χρήστης ανακατευθύνεται σε αυτή τη σελίδα:



Κάνοντας κλικ σε οποιοδήποτε από τα κουμπιά “Change Username”, ”Change Password” η ”Change Email” θα εμφανιστεί στον χρήστη μία επικάλυψη η οποία του επιτρέπει να αλλάξει το ανάλογο προσωπικό στοιχείο του λογαριασμού του.

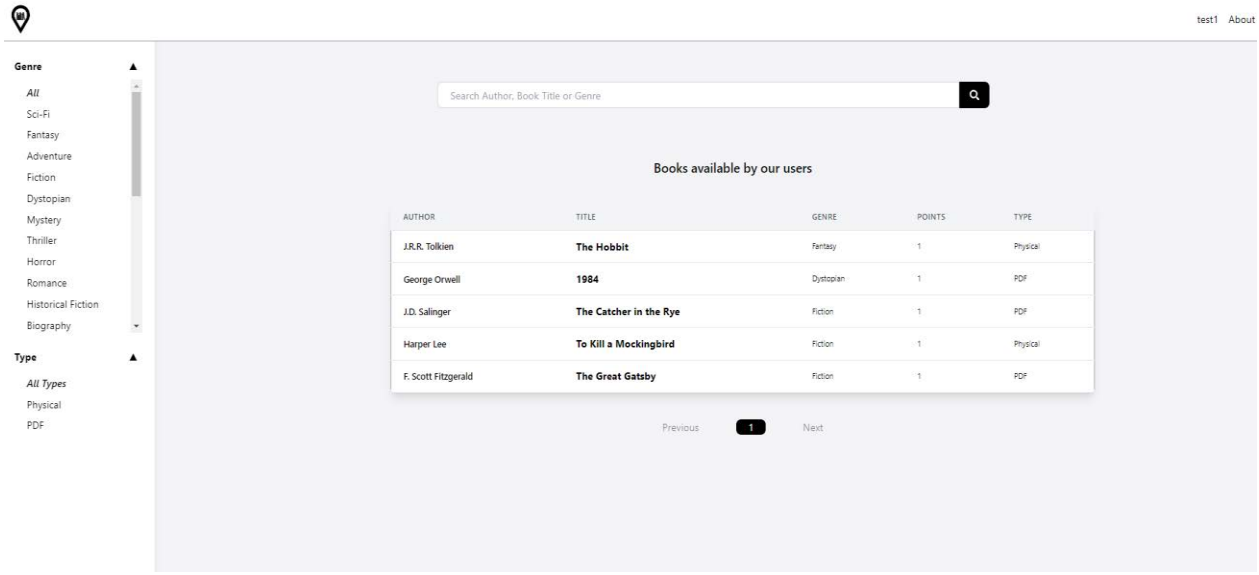


Για οποιαδήποτε απο αυτές τις πράξεις είναι απαραίτητη η επιβεβαίωση κωδικού από τον χρήστη. Τέλος, κάνοντας κλικ στο κουμπί “Delete account” ο χρήστης έχει τη δυνατότητα να διαγράψει τον λογαριασμό του. Αυτή η δράση είναι μόνιμη και μη αναστρέψιμη.

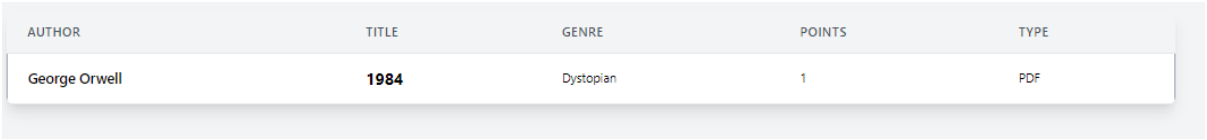


5.3.3 ΚΕΝΤΡΙΚΗ ΣΕΛΙΔΑ

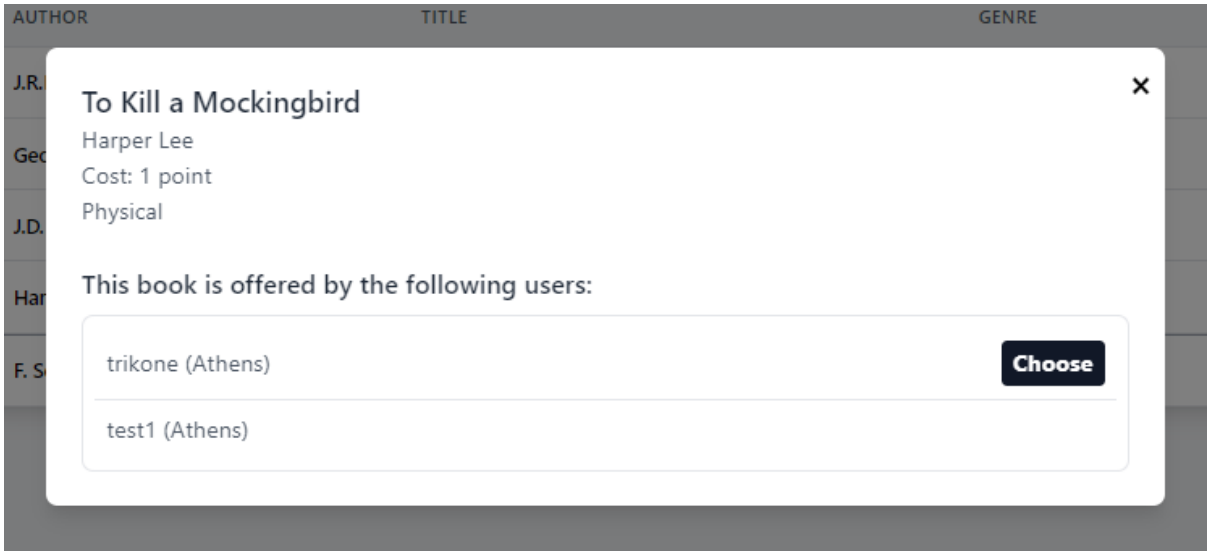
Κάνωντας κλικ στο λογότυπο πάνω αριστερά ο χρήστης ανακατευθύνεται στη Κεντρική Σελίδα.



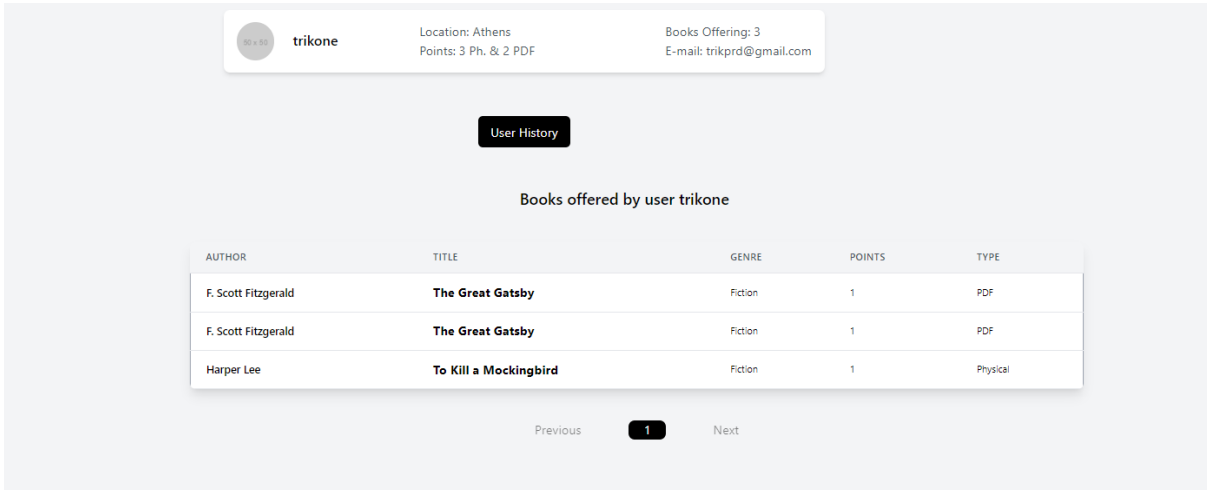
Εδώ βασίζεται η κύρια λειτουργία της σελίδας. Στα αριστερά βλέπουμε ένα Sidebar Component που έχει ακριβώς την ίδια λειτουργικότητα με αυτό της σελίδας Προφίλ. Στο πάνω μέρος επίσης, υπάρχει το Navbar Component για την περιήγηση.
Στο κέντρο υπάρχει μια μπάρα αναζήτησης η οποία χρησιμεύει για την αναζήτηση διαφόρων τίτλων, συγγραφέων ακόμη και ειδών βιβλίων, κάτι το οποίο βοηθάει τους χρήστες να βρίσκουν πιό εύκολα τα αποτελέσματα που αποζητούν. Γράφωντας έναν τίτλο και κάνοντας κλικ στο μεγεθυντικό φακό παρατηρούμε το εξής αποτέλεσμα.



Ακριβώς απο κάτω υπάρχει ένα ItemList Component το οποίο αντιπροσωπεύει όλα τα βιβλία που υπάρχουν στην εφαρμογή τη συγκεκριμένη στιγμή. Κάνοντας κλικ σε οποιοδήποτε από τα Items εμφανίζεται η εξής επικάλυψη:



Η οποία δείχνει δείχνει αναλυτικά τα χαρακτηριστικά (Όνομα, Συγγραφέας, Τύπος, Πόντοι) του βιβλίου καθώς και τους χρήστες τους οποίους το προσφέρουν μαζί με τη περιοχή την οποία βρίσκονται. Κάνοντας κλικ επάνω στο όνομα οποιουδήποτε χρήστη, ανακατευθυνόμαστε στη σελίδα προφίλ του. Αυτό μας επιτρέπει να δούμε όλα τα βιβλία τα οποία προσφέρει για ανταλλαγή καθώς και τα στοιχεία του.

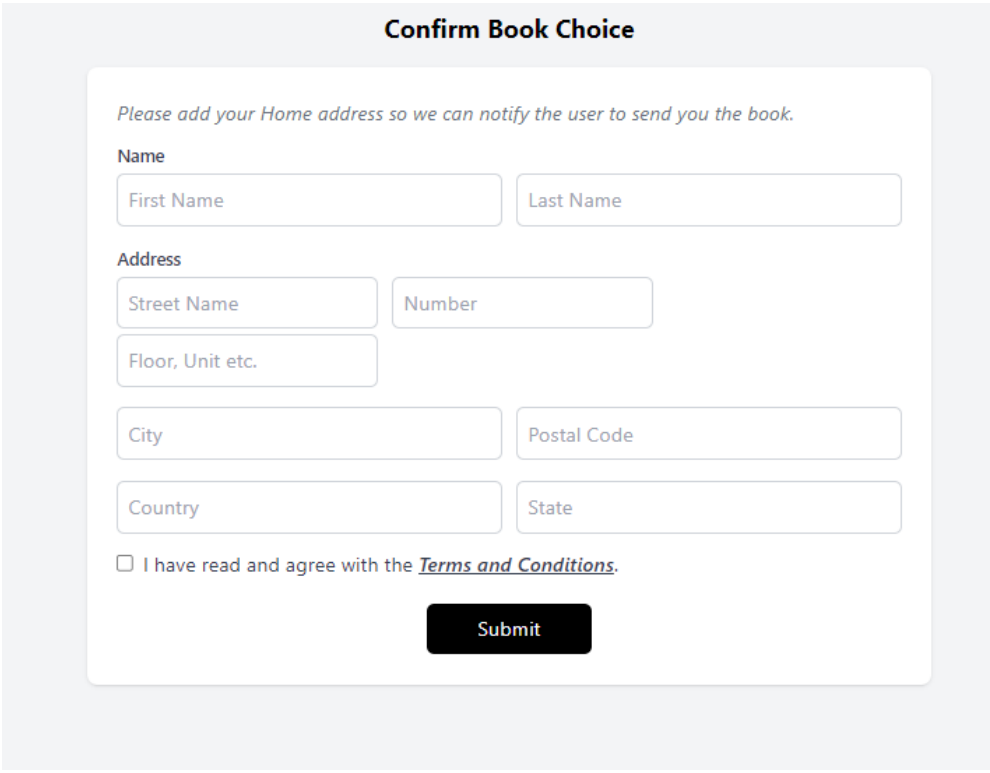


Παρατηρούμε πως στα προφίλ άλλων χρηστών δεν έχουμε επιλογές διαχείρισης λογαριασμού.

Στο πρώτο χρήστη (trikone) παρατηρούμε πως έχουμε την επιλογή “Choose”, στο δεύτερο χρήστη (test1) αυτή η επιλογή δεν υπάρχει καθώς αυτός είναι ο λογαριασμός στον οποίο είμαστε συνδεδεμένος.

5.3.4 ΕΠΙΛΟΓΗ ΒΙΒΛΙΟΥ

Κάνοντας κλικ στην επιλογή “Choose” ανακατευθυνόμαστε στην εξής σελίδα, μέσω της οποίας ξεκινάει η διαδικασία της ανταλλαγής:



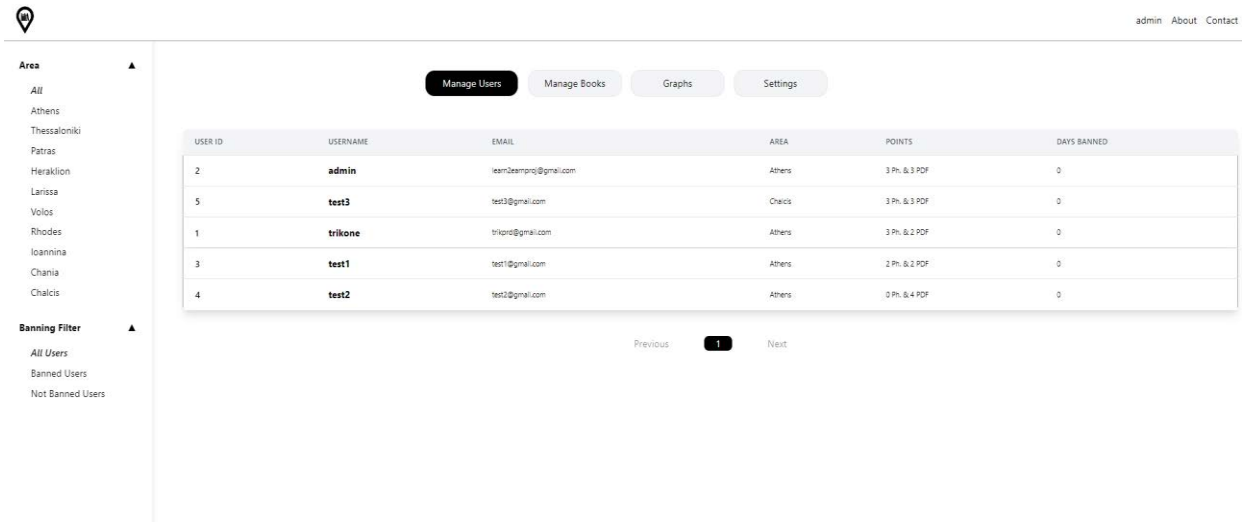
Συμπληρώνοντας τα στοιχεία του σε αυτή τη σελίδα, δεδομένου ότι έχει τους κατάλληλους πόντους, ο χρήστης μπορεί να κάνει αίτηση για το βιβλίο το οποίο τον ενδιαφέρει.

5.4 ΠΙΝΑΚΑΣ ΕΛΕΓΧΟΥ ΔΙΑΧΕΙΡΙΣΤΗ

Ο ρόλος του διαχειριστή σε μια διαδικτυακή εφαρμογή είναι η διαχείρηση όλων των στοιχείων της εφαρμογής, έτσι ώστε να μην υπάρχουν προβλήματα στην εμπειρία του κάθε χρήστη στην πλατφόρμα. Για αυτή την δουλειά, ένας πίνακας ελέγχου κρίνεται απαραίτητος.

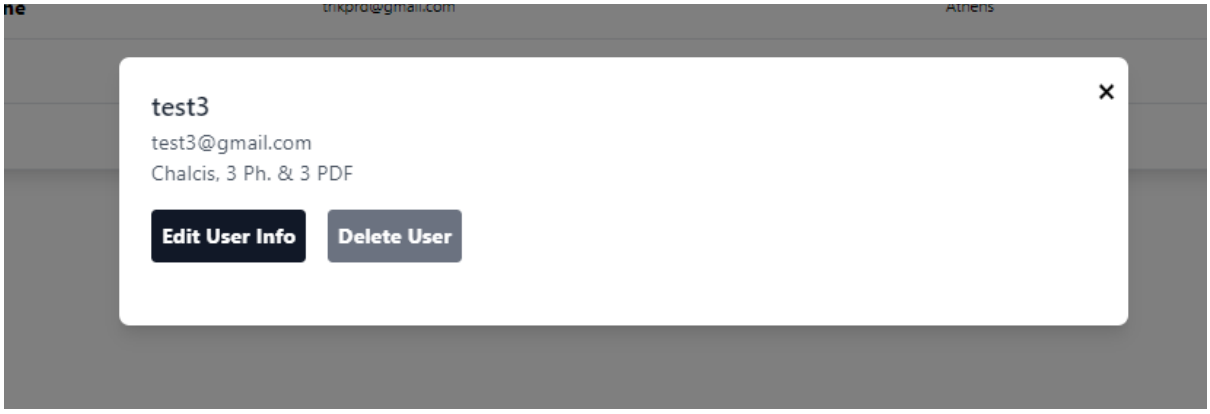
5.4.1 ΔΙΑΧΕΙΡΙΣΗ ΧΡΗΣΤΩΝ

Αν ένας χρήστης έχει τεθεί ως διαχειριστής από το σύστημα (βλέπε κεφ. 4.3.2), όταν συνδεθεί η πρώτη οθόνη η οποία βλέπει, είναι η εξής:

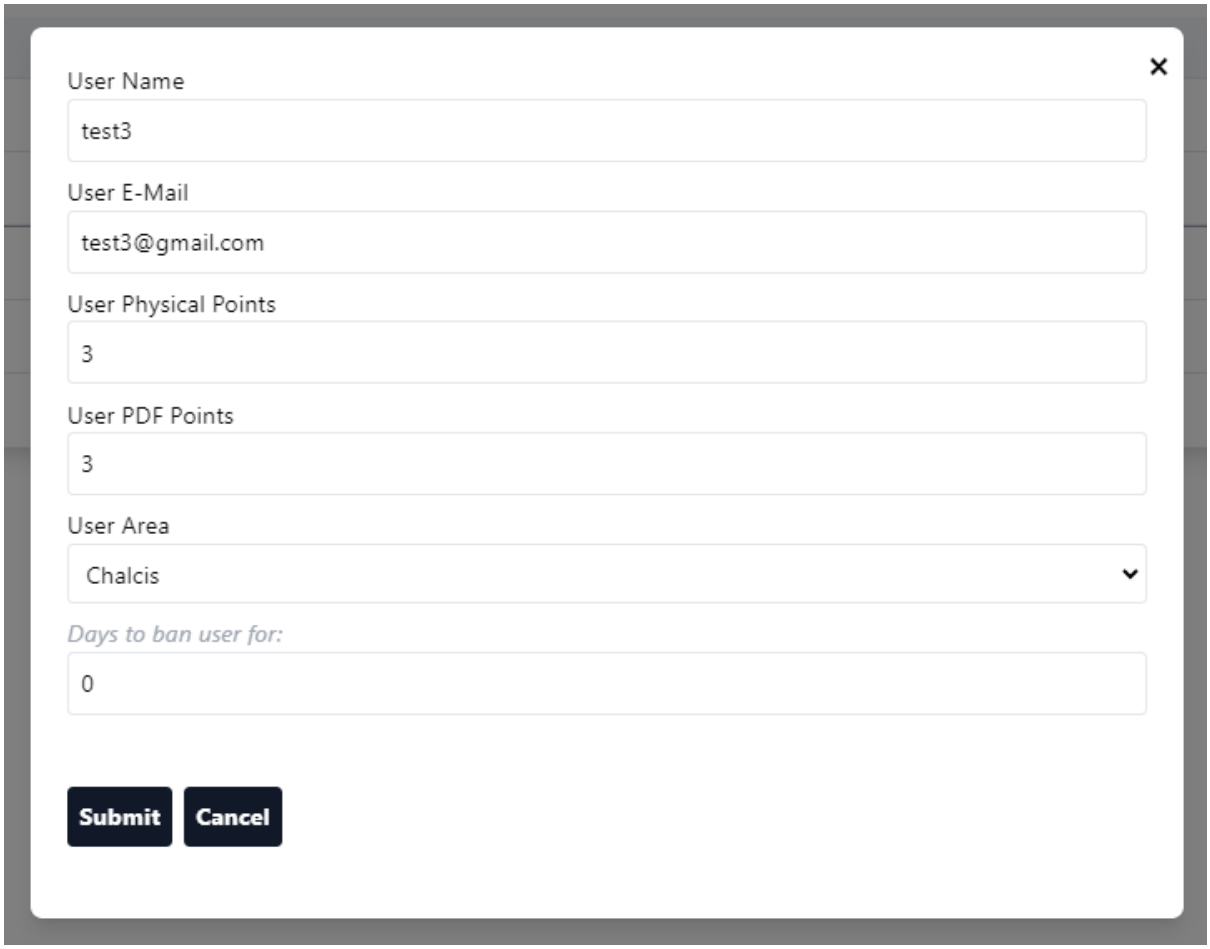


Κάνοντας κλικ στο όνομα χρήστη η στο λογότυπο ο διαχειριστής ανακατευθύνεται σε αυτή τη σελίδα. Στα αριστερά υπάρχει ένα Sidebar Component που αυτή τη φορά έχει ως φίλτρα τη περιοχή του χρήστη και το αν ο χρήστης έχει η όχι αποκλειστεί από την σελίδα.
Στο πάνω μέρος της σελίδας υπάρχουν τέσσερα κουμπιά τα οποία χρησιμοποιούνται για την πλοήγηση του διαχειριστή στον πίνακα ελέγχου. Αυτά είναι: η διαχείρηση χρηστών, η διαχείρηση βιβλίων, τα γραφήμματα και οι ρυθμίσεις λογαριασμού. Οι ρυθμίσεις λογαριασμού είναι πανομοιότυπες με αυτές ενός χρήστη, οπότε δεν θα αναφερθώ ξανά σε αυτές.

Κάνοντας κλίκ σε οποιονδήποτε χρήστη, εμφανίζεται η εξής επικάλυψη:



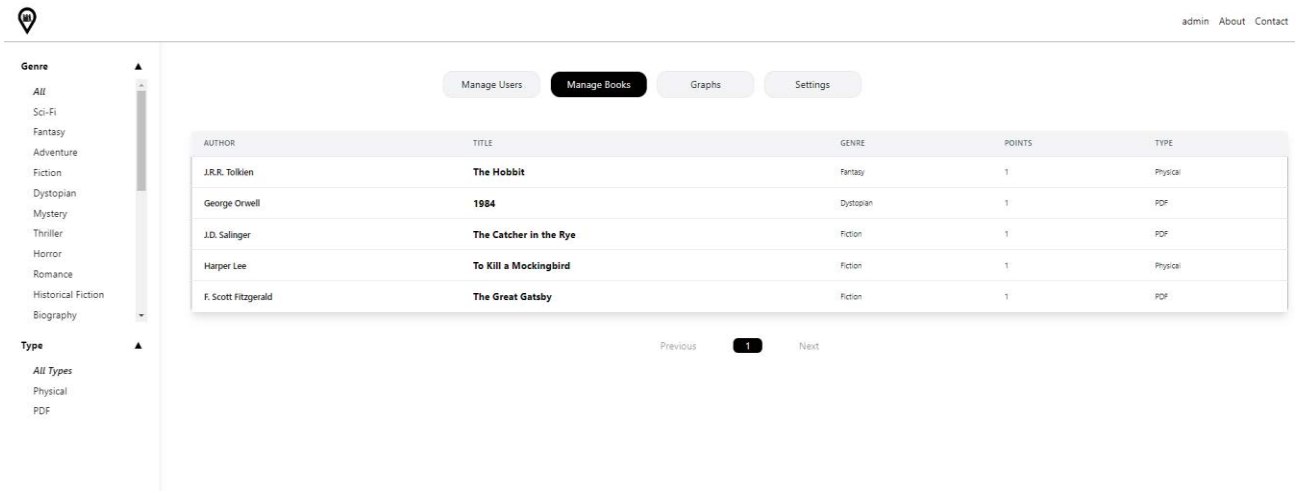
Κάνοντας κλικ στο κουμπί “Delete User” ο χρήστης διαγράφεται από την εφαρμογή. Η επιλογή αυτή είναι μη αναστρέψιμη. Κάνοντας κλικ στο κουμπί “Edit User Info” εμφανίζεται η εξής φόρμα:



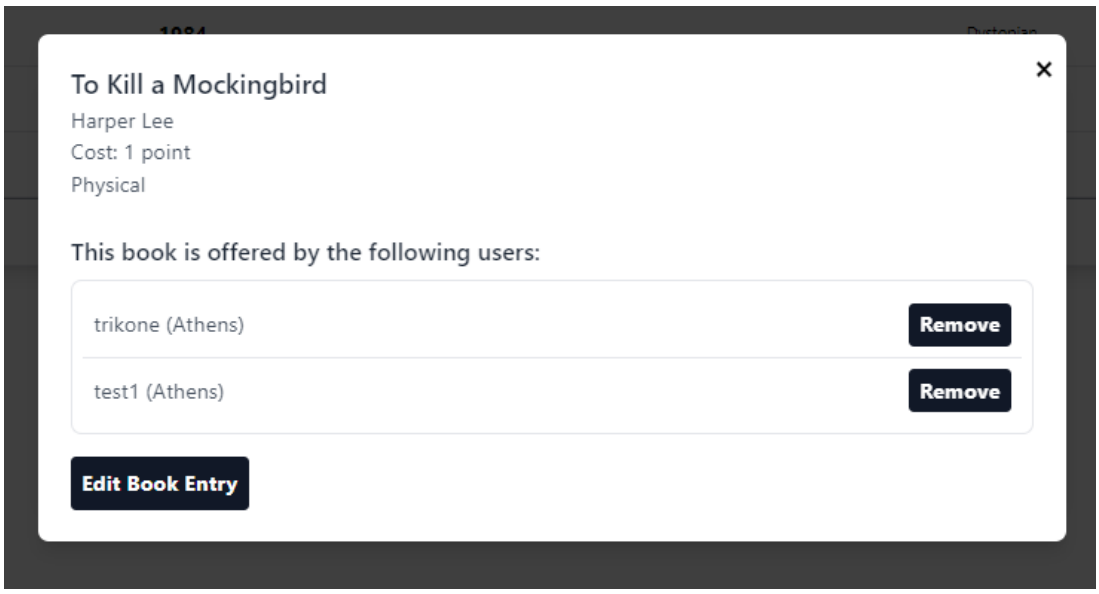
Η οποία έχει προσυμπληρωμένα τα πεδία με τα στοιχεία του χρήστη. Οποιαδήποτε αλλαγή σε αυτη θα αλλάξει και ένα στοιχείο του χρήστη. Αλλάζοντας τη τιμή στο πεδίο “Days to ban user for” ο διαχειριστής μπορεί να αποκλείσει τον χρήστη από την εφαρμογή για ένα συγκεκριμένο χρονικό διάστημα.

5.4.2 ΔΙΑΧΕΙΡΙΣΗ ΒΙΒΛΙΩΝ

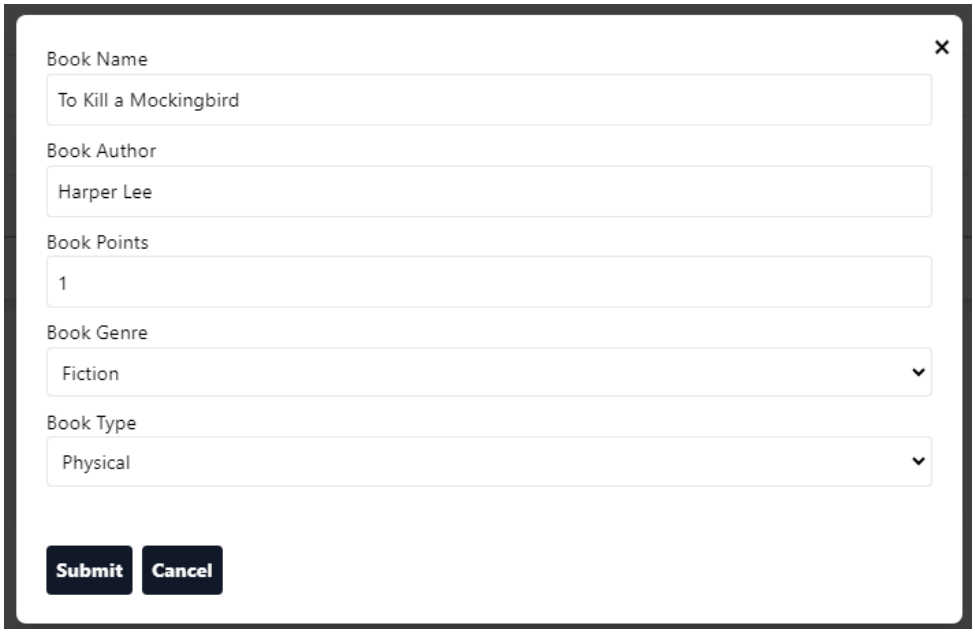
Κάνοντας κλικ στην επιλογή “Manage Books” ο διαχειριστής ανακατευθύνεται στην εξής οθόνη.



Το Sidebar Component έχει τα ανάλογα φίλτρα των βιβλίων. Στο κέντρο υπάρχει ένα ItemList Component το οποίο δείχνει όλα τα βιβλία που υπάρχουν στην βάση δεδομένων. Κάνοντας κλικ σε κάποιο από τα Items, εμφανίζεται η εξής επικάλυψη:



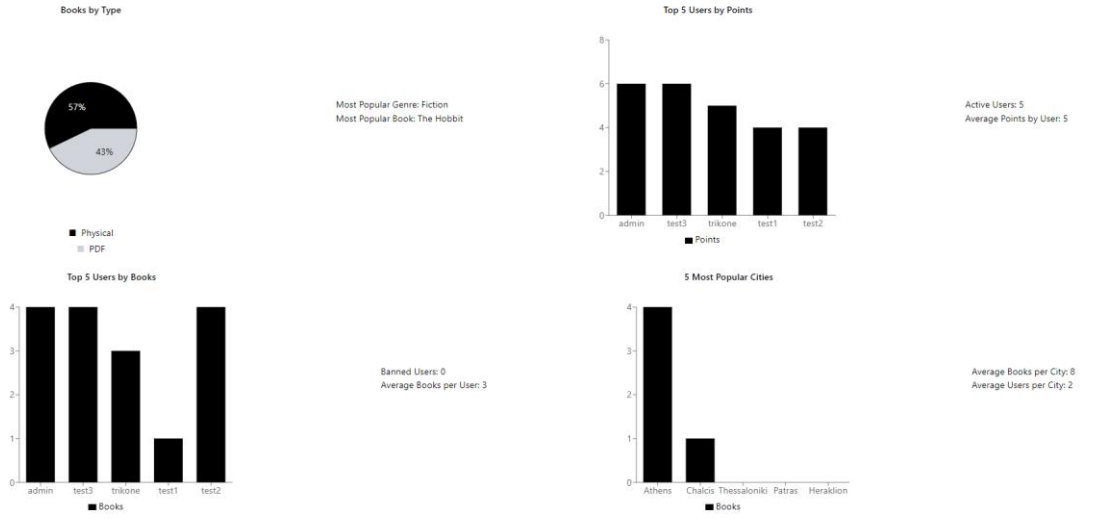
Στην οποία ο διαχειριστής μπορεί να επιλέξει να διαγράψει ένα βιβλίο απο κάποιον χρήστη η να επεξεργαστεί τα στοιχεία του βιβλίου. Κάνοντας κλικ πάνω στο “Edit Book Entry”, εμφανίζεται η εξής φόρμα:



Εδώ ο διαχειριστής μπορεί να επεξεργαστεί οποιοδήποτε από τα προσυμπληρωμένα στοιχεία του βιβλίου. Κάθε αλλαγή σε αυτή τη φόρμα είναι μη αναστρέψιμη.

5.4.3 ΠΙΝΑΚΑΣ ΓΡΑΦΗΜΑΤΩΝ

Κάνοντας κλικ στην επιλογή “Graphs” ο διαχειριστής ανακατευθύνεται στην εξής οθόνη:



Η οποία δείχνει κάποια πολύ σημαντικά γραφήματα τα οποία αντιπροσωπεύουν την κατάσταση της εφαρμογής. Δίπλα σε κάθε γράφημα υπάρχουν σημαντικές μετρικές, όπως ο μέσος αριθμό βιβλίων ανα χρήστη η ο αριθμός των αποκλεισμένων χρηστών.

6. ΣΥΜΠΕΡΑΣΜΑΤΑ

6.1 ΣΥΝΟΨΗ

Στη παρούσα εργασία έκανα μια βασική παρουσίαση του πεδίου της ανάπτυξης διαδικτυακών εφαρμογών και είδαμε το πόσο χρήσιμες μπορεί να φανούν στον κάθε άνθρωπο. Η εφαρμογή η οποία υλοποιήθηκε μπορεί να χρησιμοποιηθεί από ένα μεγάλο εύρος ανθρώπων και έχει τις πρωοπτικές να φανεί πολύ κερδοφόρα σε

αυτούς μειώνοντας το κόστος αγοράς βιβλίων και προωθώντας την έννοια της ανταλλαγής γνώσεων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1:https://developer.mozilla.org/enUS/docs/Learn/Getting_started_with_the_web/The_web_and_web_standards

2: https://www.techopedia.com/definition/23889/web-development

3: https://www.heavy.ai/technical-glossary/client-server

4: https://www.techtarget.com/searchnetworking/definition/client-server

5: https://www.geeksforgeeks.org/tcp-ip-model/

6: https://www.eef.edu.gr/el/programme/front-end-web-development/

7: https://www.cloudflare.com/learning/serverless/glossary/client-side-vs-server-side/

8: https://cloudinary.com/guides/front-end-development/front-end-development-the-complete-guide

9: https://www.romexsoft.com/blog/javascript-for-front-end-development-basics-and-concepts/

10: https://www.netsolutions.com/insights/what-is-a-framework-in-programming/
11: https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/Introduction
12: https://vuejs.org/guide/introduction.html
13: https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/Introduction
14: https://www.ibm.com/topics/api
15: https://aws.amazon.com/what-is/api/
16: https://learn.microsoft.com/en-us/aspnet/overview
17: https://aws.amazon.com/what-is/django/
18: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction
19: https://www.ibm.com/topics/java-spring-boot
20: https://flatirons.com/blog/what-is-flask-overview-of-the-flask-python-framework-2024/
21: https://guides.rubyonrails.org/getting_started.html
22: https://builtin.com/software-engineering-perspectives/laravel
23: https://www.appdynamics.com/topics/database-management-systems#~1-what-is-dbms
24: https://aws.amazon.com/what-is/sql/
25: https://reldb.org/c/index.php/twelve-rules/
26: https://learn.microsoft.com/en-us/azure/architecture/data-guide/big-data/non-relational-data
27: https://www.mongodb.com/resources/basics/databases/non-relational
28: https://www.dataversity.net/a-brief-history-of-non-relational-databases/
29: https://medium.com/@reactmasters.in/what-are-components-and-types-of-components-in-react-js-4e2642b136a2