ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«IEEAlumni - Web based Εφαρμογή αποφοίτων πανεπιστημιακού επιπέδου»

Του φοιτητή
Χρήστου Παυλίδη
Αρ. Μητρώου: 164782

Επιβλέπων
Αντώνης Σιδηρόπουλος
Αναπληρωτής Καθηγητής

Ημερομηνία 26-05-2024

Τίτλος Π.Ε.: Web based Εφαρμογή αποφοίτων πανεπιστημιακού επιπέδου
Κωδικός Π.Ε.: 23188
Ονοματεπώνυμο φοιτητή: Χρήστος Παυλίδης
Ονοματεπώνυμο εισηγητή: Αντώνιος Σιδηρόπουλος
Ημερομηνία ανάληψης Π.Ε.: 29-03-2023
Ημερομηνία περάτωσης Π.Ε.: 26-05-2024

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως πτυχιακή εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

«Στην οικογένειά μου για τη στήριξή της,
στους φίλους μου για την ενθάρρυνσή τους,
και στους συναδέλφους μου,
που προσέφεραν με τις γνώσεις και την εμπειρία τους.»

Πρόλογος

Κατά τη διάρκεια φοίτησής μου στο τμήμα, είχα καθημερινή αλληλεπίδραση με τις υπηρεσίες της σχολής. Με το πέρασμα των χρόνων, ήταν εμφανής η ανανέωση αυτών των υπηρεσιών που συνήθως συνέβαινε μέσω εργασιών που αναλάμβαναν συμφοιτητές μου. Αντιλαμβανόμενος τη συνεχή εξέλιξη και βελτίωση του τμήματος σε αυτόν τον τομέα, η επιλογή αυτού του θέματος δεν ήταν δύσκολη, καθώς ήξερα ότι θα μπορούσα κι εγώ να συμβάλλω στην ανανέωση μιας ήδη υπάρχουσας πλατφόρμας, δημιουργώντας την διεπαφή χρήστη της νέας έκδοσης. Εκφράζοντας τον σεβασμό και την εκτίμηση για το ήδη υπάρχον έργο, που αποτελεί πηγή έμπνευσης, ο στόχος της νέας έκδοσης είναι η οπτική και λειτουργική βελτίωση, καθώς και η προσθήκη νέων χαρακτηριστικών που θα ανταποκρίνονται στις νέες απαιτήσεις.

Περίληψη

Η παρούσα Πτυχιακή Εργασία αφορά την υλοποίηση μιας Web-based Εφαρμογής, η οποία σχεδιάστηκε για την ενίσχυση της επαγγελματικής δικτύωσης μεταξύ των αποφοίτων, καθώς και τη διαχείριση αυτών από τη γραμματεία του τμήματος. Αποτελείται από δύο διαφορετικά περιβάλλοντα χρήστη, το καθένα από τα οποία εξυπηρετεί συγκεκριμένες λειτουργικότητες. Αρχικά, υπάρχει το περιβάλλον του διαχειριστή, το οποίο απευθύνεται στη γραμματεία και την ομάδα ΟΜΕΑ. Σε αυτό το περιβάλλον, οι χρήστες μπορούν να διαχειριστούν τα διάφορα καθήκοντα και εργασίες που αφορούν τη διαχείριση του εκάστοτε τμήματος. Επιπλέον, υπάρχει το περιβάλλον του απλού χρήστη, το οποίο σχεδιάστηκε ειδικά για τους αποφοίτους. Σε αυτό το περιβάλλον, οι απόφοιτοι έχουν πρόσβαση σε συγκεκριμένες λειτουργίες και πληροφορίες χωρίς την ανάγκη να αναλαμβάνουν τη διαχείριση ή τη διαμόρφωση του συστήματος. Μέσω αυτού του περιβάλλοντος επωφελούνται από τις πληροφορίες και τις δυνατότητες που παρέχονται μέσω της εφαρμογής, όπως τα στοιχεία επικοινωνίας κλπ. Στο κομμάτι του front-end, το οποίο αποτελεί και την πτυχιακή εργασία, χρησιμοποιήθηκε το framework Vue.js. Το back-end υλοποιήθηκε σε Laravel ως αντικείμενο της μεταπτυχιακής διπλωματικής εργασίας του Ταουκτσή Βασίλειου [1] ενώ η μεταξύ τους επικοινωνία επιτυγχάνεται μέσω αιτημάτων Axios.

«Web based Application of university level graduates»

Pavlidis Christos

Abstract

This Thesis concerns the implementation of a Web-based Application, which was designed to enhance professional networking among graduates, as well as their management by the secretariat of the department. It consists of two different user interfaces, each serving specific functionalities. Initially, there is the administrator interface, which is addressed to the secretariat and the OMEA team. In this environment, users can manage the various tasks and tasks related to the management of each department. In addition, there is the simple user interface, which was designed especially for graduates. In this environment, graduates have access to specific functions and information without the need to undertake the management or configuration of the system. Through this environment they benefit from the information and features provided through the application, such as contact details, etc. In the front-end part, which also constitutes the thesis, the Vue.js framework was used. The back-end was implemented in Laravel as the subject of Vasilios Taouktsis postgraduate thesis [1] while communication between them is achieved through Axios requests.

Ευχαριστίες

Η διεκπεραίωση αυτής της εργασίας οφείλεται σε πολλούς παράγοντες. Αρχικά, θα ήθελα να ευχαριστήσω τον κ. Σιδηρόπουλο για την ευκαιρία ανάληψης αυτής της πτυχιακής, αλλά και για την εμπιστοσύνη που επέδειξε προς την ολοκλήρωσή της. Η καθοδήγησή του, η εμπειρία του καθώς και η άμεση ανταπόκρισή του σε κάθε ζήτημα, αποδείχθηκαν καθοριστικές σε κάθε στάδιο της διαδικασίας. Επίσης, θα ήθελα να ευχαριστήσω τον κ. Ταουκτσή για την υλοποίηση του API στο οποίο βασίζεται η εργασία. Η συνεισφορά του στο τεχνικό κομμάτι της εργασίας ήταν κρίσιμη για την επιτυχή υλοποίηση του έργου. Τέλος, θέλω να εκφράσω τις ευχαριστίες μου προς τους φίλους, την οικογένειά μου και τους συναδέλφους μου, που βοήθησαν ο καθένας με τον τρόπο του κατά την πορεία της πτυχιακής εργασίας μου.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Αποσπασμάτων Κώδικα

Συντομογραφίες

ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
ΜΠΗΣ	Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων
DOM	Document Object Model
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
SASS	Syntactically Awesome Style Sheets
SCSS	Sassy Cascading Style Sheets
JS	JavaScript
TS	TypeScript
PHP	Hypertext Preprocessor
OMEA	Ομάδα Εσωτερικής Αξιολόγησης
XML	Extensible Markup Language
XHTML	Extensible Hypertext Markup Language
UI	User Interface
ER	Entity Relationship
VS Code	Visual Studio Code
Τ.Κ.	Ταχυδρομικός Κώδικας

Κεφάλαιο 1ο: Εισαγωγή

1.1 Εισαγωγή

Το IEEAlumni είναι μια διαδικτυακή πλατφόρμα που δημιουργήθηκε με σκοπό να συνδέσει τους πτυχιούχους του προπτυχιακού και των μεταπτυχιακών προγραμμάτων σπουδών του τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων (ΜΠΗΣ) του Διεθνούς Πανεπιστημίου της Ελλάδος (ΔΙΠΑΕ). Η πλατφόρμα αυτή επιτρέπει στη γραμματεία του τμήματος να καταχωρεί και να διαχειρίζεται δεδομένα σχετικά με τους απόφοιτους. Παράλληλα, προσφέρει στους απόφοιτους τη δυνατότητα να εισάγουν και να ενημερώνουν προσωπικές και επαγγελματικές πληροφορίες, οι οποίες είναι χρήσιμες για την αξιολόγηση και τη στατιστική ανάλυση από την ΟΜΕΑ (Ομάδα Εσωτερικής Αξιολόγησης).

Η εφαρμογή έχει τεθεί σε λειτουργία επιτυχώς, καλύπτοντας όλους τους βασικούς στόχους της. Παράλληλα, ξεκίνησε η ανάπτυξη της δεύτερης έκδοσης [1] της πλατφόρμας, η οποία ενσωματώνει σημαντικές βελτιώσεις και αλλαγές. Ένας από τους κύριους στόχους της νέας έκδοσης είναι η ανεξαρτητοποίηση του back-end από το front-end της εφαρμογής. Αυτό σημαίνει ότι η λογική και η επεξεργασία των δεδομένων θα είναι ξεχωριστές από το κομμάτι της διεπαφής χρήστη, προσφέροντας μεγαλύτερη ευελιξία και δυνατότητες αναβάθμισης.

Το αντικείμενο αυτής της πτυχιακής εργασίας είναι η ανάπτυξη του front-end της δεύτερης έκδοσης της πλατφόρμας και η υλοποίηση όλων των νέων χαρακτηριστικών που θα βελτιώσουν την εμπειρία των χρηστών. Η νέα έκδοση αναμένεται να προσφέρει βελτιωμένη λειτουργικότητα, καλύτερη απόδοση και μια πιο φιλική προς τον χρήστη διεπαφή.

1.2 Περιγραφή της αρχιτεκτονικής της εφαρμογής

Η εφαρμογή αποτελεί προϊόν ενός συνδυασμού τεχνολογιών, οι οποίες είναι άρτια συνδεδεμένες μεταξύ τους. Η ανάπτυξη βασίζεται στο Vue.js, ένα δημοφιλές JavaScript framework που χρησιμοποιείται για τη δημιουργία διεπαφών χρήστη. Το Vue.js επιλέχθηκε λόγω των χαρακτηριστικών του που προσφέρουν ευελιξία, απόδοση και ευκολία στη χρήση.

Για την περαιτέρω βελτίωση της ανάπτυξης και της εμφάνισης της εφαρμογής, χρησιμοποιήθηκε η βιβλιοθήκη Quasar, η οποία παρέχει έτοιμα components και κλάσεις styling. Επιπλέον, χρησιμοποιήθηκε η βιβλιοθήκη μορφοποίησης CSS κλάσεων Tailwind, που επιτρέπει τη γρήγορη και εύκολη δημιουργία προσαρμοσμένων σχεδιαστικών στοιχείων.

Η συνδυασμένη χρήση αυτών των τεχνολογιών εξασφαλίζει ότι η εφαρμογή είναι τόσο λειτουργική όσο και αισθητικά ευχάριστη, προσφέροντας μια ενιαία και συνεκτική εμπειρία στους χρήστες..

1.3 Οργάνωση της εργασίας

Η εργασία αποτελείται από τα παρακάτω κεφάλαια:

- **Εισαγωγή:** Παρουσιάζεται μια πρώτη γνωριμία με το αντικείμενο της εργασίας, οι στόχοι της και η συνοπτική περιγραφή της αρχιτεκτονικής της εφαρμογής.
- **Γλώσσες και Τεχνολογίες:** Περιλαμβάνεται λεπτομερής ανάλυση των τεχνολογιών που χρησιμοποιήθηκαν για την υλοποίηση της πλατφόρμας.

- **Περιγραφή Βάσης Δεδομένων:** Παρουσιάζεται η δομή της βάσης δεδομένων και οι σχέσεις μεταξύ των οντοτήτων κάνοντας την χρήση διαγραμμάτων ER.
- **Εργαλεία σχεδίασης και υλοποίησης του Alumni:** Περιγραφή των εργαλείων και του τρόπου με τον οποίο χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής.
- **Παρουσίαση του Alumni:** Παρουσιάζεται ολοκληρωμένα η πλατφόρμα, με αναλυτική περιγραφή των λειτουργιών της μέσω της χρήσης στιγμιότυπων οθόνης.
- **Συμπεράσματα και Μελλοντικές Επεκτάσεις:** Συμπεράσματα και προτάσεις για μελλοντικές επεκτάσεις που προέκυψαν στην πορεία της υλοποίησης της εφαρμογής.
- **Βιβλιογραφία**

Κεφάλαιο 2ο: Γλώσσες και Τεχνολογίες

2.1 Εισαγωγή

Στο κεφάλαιο αυτό, πραγματοποιείται μια εκτενής ανάλυση των τεχνολογιών που απαρτίζουν τον σκελετό της Web εφαρμογής που αναπτύχθηκε. Ξεκινώντας με το client side, γίνεται περιγραφή της HTML και της CSS που αποτελούν την βάση για την κατανόηση της δημιουργίας και της σχεδίασης ιστοσελίδων, ενώ στην προχωράμε στην ανάλυση πιο προηγμένων εργαλείων και τεχνικών, όπως η γλώσσα προεπεξεργασίας SASS και η δυναμική λειτουργικότητα που προσφέρει η JavaScript. Επιπλέον, εξετάζονται τα πλεονεκτήματα της χρήσης TypeScript για την ανάπτυξη πιο ασφαλών και εύκολα συντηρήσιμων εφαρμογών. Μετά την παρουσίαση των βασικών γλωσσών και εργαλείων, εστιάζουμε στις τεχνολογίες Vue.js και Quasar Framework, εργαλεία κλειδιά για την ανάπτυξη δυναμικών και ευέλικτων εφαρμογών. Τέλος, εξετάζουμε τη χρήση της Tailwind CSS, μιας σύγχρονης μεθόδου μορφοποίησης διεπαφής χρήστη. Το δεύτερο σκέλος του κεφαλαίου αφορά το server side εξετάζοντας τις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση του back-end. Η ανάπτυξη του βασίστηκε στην μεταπτυχιακή διπλωματική εργασία του Ταουκτσή Βασίλειου [1] και έγινε με την χρήση της Laravel ένα PHP framework. Με αυτήν την ενότητα, ο αναγνώστης θα αποκτήσει μια πλήρη εικόνα των τεχνολογιών που εφαρμόστηκαν στην ανάπτυξη της πλατφόρμας, καθώς και των πλεονεκτημάτων και των χαρακτηριστικών τους.

2.2 Client Side

2.2.1 HTML

Η HyperText Markup Language, ή αλλιώς γνωστή στο ευρύτερο κοινό ως HTML, είναι η βασική γλώσσα σήμανσης για έγγραφα που είναι σχεδιασμένα να εμφανίζουν περιεχόμενο σε ένα πρόγραμμα περιήγησης ιστού. Καθορίζει το περιεχόμενο και τη δομή του διαδικτυακού περιεχομένου καθώς σου δίνει την δυνατότητα να εισάγεις κείμενο, εικόνες, γραφικά και άλλα, τα οποία συνθέτουν μία ιστοσελίδα στον ιστότοπο. Σχεδόν πάντα υποβοηθείται από τεχνολογίες όπως τα Cascading Style Sheets (CSS) και scripting γλώσσες όπως η JavaScript, τις οποίες και θα αναλύσουμε στην συνέχεια της εργασίας. Η δομή των HTML αρχείων αποτελείται από μία σειρά στοιχείων, καθένα από τα οποία περιβάλλεται από ετικέτες ανοίγματος και κλεισίματος. Για παράδειγμα, ένα tag χρησιμοποιείται για τον ορισμό μιας παραγράφου και ακολουθείται από το οποίο δηλώνει το τέλος του tag [2].

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Hello, World!</h1>
    <p>This is a basic HTML document.</p>
</body>
</html>
```

Απόσπασμα κώδικα 1: Βασική δομή της HTML

2.2.1.1 Ιστορία

Η ιστορία της HTML αποτελεί ένα ταξίδι που διατρέχει δεκαετίες, από τη γέννησή της στις αρχές της δεκαετίας του 1990 έως τον τρέχοντα ρόλο της ως την αναπτυξιακή αρχή του Παγκόσμιου Ιστού.

Το 1989, ο Tim Berners-Lee, ένας οραματιστής Βρετανός επιστήμονας υπολογιστών, ξεκίνησε ένα επαναστατικό ταξίδι εφευρίσκοντας τον Παγκόσμιο Ιστό, με την HTML να χρησιμεύει ως θεμελιώδης γλώσσα του. [3] Το πρωτοποριακό έργο του οδήγησε στην εισαγωγή ενός πρωτότυπου προγράμματος περιήγησης στο Web στον υπολογιστή NeXT το 1990 [4], σηματοδοτώντας την αυγή της ψηφιακής εποχής όπως τη γνωρίζουμε σήμερα. Αυτή ήταν μια κρίσιμη στιγμή, θέτοντας τις βάσεις για τον διασυνδεδεμένο κόσμο που ζούμε τώρα.

Τον Σεπτέμβριο του 1991, ο Tim Berners-Lee ξεκίνησε έναν παγκόσμιο διάλογο σχετικά με την HTML μέσω της λίστας αλληλογραφίας WWW-talk. Αυτή η ανοιχτή πλατφόρμα συζήτησης έγινε το κατάλληλο μέρος όπου οι λάτρεις των υπολογιστών και γενικότερα της τεχνολογίας, να ανταλλάξουν ιδέες γνώσεις και προβληματισμούς. Τελικά, αυτή του η κίνηση αποδείχθηκε κομβική καθώς η παγκόσμια κοινότητα συνέβαλε στην ανάπτυξη και την εξέλιξη της HTML, προωθώντας τη συνεργασία και την καινοτομία. Ένα από τα πρώτα και πιο σημαντικά πρόσωπα που προσέγγισε αυτή η πλατφόρμα ήταν ο Dave Raggett ο οποίος και επισκέφτηκε τον Tim το 1992. Μαζί εξέτασαν το πως θα μπορούσε η HTML να διαμορφωθεί σε κάτι πιο κατάλληλο για μαζική κατανάλωση. Έτσι αργότερα ο Dave συνέθεσε την HTML+, μια απλούστερη έκδοση της αρχικής HTML.

Μέχρι τον Μάιο του 1994, το πρώτο συνέδριο World Wide Web πραγματοποιήθηκε στη Γενεύη, παρουσιάζοντας την HTML + και θέτοντας τις βάσεις για το μέλλον της ανάπτυξης του Web. Αυτή η συνάντηση συγκέντρωσε 380 ενθουσιώδεις από όλο τον κόσμο, συμπεριλαμβανομένων εκπροσώπων από τον ακαδημαϊκό χώρο και τη βιομηχανία, τονίζοντας τη σημασία της HTML στη διαμόρφωση του ψηφιακού κόσμου.

Με το πέρασμα του χρόνου, πολλά προγράμματα περιήγησης είχαν προσθέσει τα δικά τους bit στην HTML. Η γλώσσα γινόταν ασαφής. Κάπου εκεί έλαβαν δράση ο Dan Connolly και οι συνεργάτες του, συνέλεξαν όλες τις HTML ετικέτες σε ένα προσχέδιο έγγραφο και κάπως έτσι η HTML 2 άρχισε να παίρνει σάρκα και οστά. Δεν σταμάτησαν εκεί, καθώς το προσχέδιο κυκλοφόρησε στην κοινότητα του διαδικτύου για σχόλια και ο Dan έλαβε υπόψη του πολλές προτάσεις από ενθουσιώδεις της HTML από όλο τον κόσμο, διασφαλίζοντας ότι ο τελικός ορισμός της HTML 2.0 θα ικανοποιούσε τους πάντες.

Ο Φεβρουάριος του 1996 σηματοδότησε το σχηματισμό της HTML Editorial Review Board (ERB), η οποία είχε ως στόχο να βάλει τέλος στην εποχή των ειδικών υποσυνόλων HTML για κάθε περιηγητή, προωθώντας την συμβατότητα στο διαδίκτυο και ανοίγοντας το δρόμο για μια πιο ενιαία εμπειρία διαδικτύου.

Το Μάρτιο του 1995 είδε το φως της δημοσιότητας η HTML 3, εισάγοντας νέες λειτουργίες και ετικέτες όπως η FIG για εικόνες και η STYLE για την δημιουργία κανόνων μορφοποίησης. Παρά την καινοτόμα προσέγγισή της, το προσχέδιο αντιμετώπισε προκλήσεις στην επικύρωση λόγω του μεγέθους του και του αριθμού των νέων προτάσεων.

Κάτι παραπάνω από έναν χρόνο αργότερα, τον Δεκέμβριο του 1996, ξεκίνησε η ανάπτυξη της νέας έκδοσης της HTML με όνομα “Cougar”, η οποία εν τέλει οδήγησε στην ανάπτυξη της HTML 4 [5].

Για αρκετά χρόνια αργότερα η HTML έμεινε στην έκδοση 4.01 και υπήρχε η τάση μετάβασης από την HTML στην XML. Έτσι δημιουργήθηκε η XHTML, ένα μέλος της οικογένειας γλωσσών σήμανσης

XML που επεκτείνει την HTML. Ωστόσο οι οργανισμοί προγραμματιστών ιστού συνειδητοποίησαν ότι μια τέτοια μετάβαση είναι δύσκολο να δουλέψει στην πράξη οπότε και έκριναν απαραίτητη την περαιτέρω ανάπτυξη της HTML, έτσι τον Ιανουάριο του 2008 δημοσιεύθηκε το προσχέδιο της πέμπτης και τελευταίας έκδοσης. Η HTML5, συνεχίζει να εξελίσσεται, ενσωματώνοντας νέα χαρακτηριστικά και τεχνολογίες για να υποστηρίξει πλούσιες εφαρμογές διαδικτύου, πολυμέσα και διαδραστικές εμπειρίες [6].

2.2.2 CSS

Η CSS (Cascading Style Sheets) είναι μια τεχνολογία που δίνει την δυνατότητα στους προγραμματιστές να ελέγχουν την οπτική εμφάνιση των ιστότοπων, καθώς δίνει τον πλήρη έλεγχο στο πως θα αποδίδονται στην οθόνη η διάταξη, τα χρώματα, τις γραμματοσειρές και άλλα οπτικά χαρακτηριστικά [7].

2.2.2.1 Ιστορία

Η CSS σαν τεχνολογία προτάθηκε για πρώτη φορά το 1994. Ο στόχος της ήταν να προσφέρει στους δημιουργούς ιστοσελίδων μία γλώσσα για styles που θα ήταν ευέλικτη και θα παρείχε στιλιστική δύναμη τόσο σε αυτούς όσο και στους χρήστες των ιστοσελίδων.

Η πρώτη έκδοση, CSS1, κυκλοφόρησε το 1996 και πρόσφερε τις βασικές ιδιότητες στυλ, όπως την επιλογή γραμματοσειράς (font-family), το μέγεθος γραμματοσειράς (font-size), το χρώμα γραμματοσειράς (color), και τη διάταξη κειμένου (text-align).

Στις αρχές του 1998 κυκλοφόρησε η δεύτερη έκδοση, CSS2 όπου προστέθηκαν νέες δυνατότητες όπως το μοντέλο κουτιού, που επιτρέπει τον έλεγχο των περιοχών περιεχομένου με ιδιότητες όπως τα περιθώρια (margin), το περιθώριο εσωτερικά (padding), και οι περιοχές περιγράμματος (border).

Με την ολοκλήρωση της δεύτερης έκδοσης άρχισαν αμέσως οι εργασίες για την ανάπτυξη της CSS3, καθώς και μιας ενημερωμένης έκδοσης της CSS2 που ονομάστηκε CSS2.1. Η CSS3 εν τέλει δεν έγινε μια ολοκληρωμένη προδιαγραφή όπως η CSS2.1, αλλά αναπτύχθηκε ως ένα σύνολο από διάφορα modules, κάθε ένα από τα οποία εισήχθη και εξελίχθηκε ξεχωριστά. Θα έλεγε κανείς ότι αναπαριστά μια συλλογή από νέες λειτουργίες που ενσωματώθηκαν στο CSS με την πάροδο του χρόνου, αντί για μια μοναδική και ολοκληρωμένη εκδοχή. Οι δυνατότητες της CSS3 περιλαμβάνουν προηγμένες δυνατότητες ελέγχου διάταξης (όπως Flexbox και Grid), εντυπωσιακά γραφικά με μετασχηματισμούς 2D και 3D, καθώς και εκτεταμένη υποστήριξη για πολλές άλλες λειτουργίες και εφέ.

Από την άλλη κυκλοφορία ενημερωμένης έκδοσης CSS2.1, έλαβε μέρος το 2011 και περιελάβανε διορθώσεις και διευκρινίσεις στις προδιαγραφές, εξασφαλίζοντας τη συμβατότητα μεταξύ των διαφορετικών περιηγητών [8].

2.2.2.2 Τρόποι χρήσης

Οι πιο συχνοί τρόποι χρήσης της CSS σε ένα HTML αρχείο είναι οι εξείς:

Inline CSS: Εφαρμογή του CSS απευθείας στο HTML στοιχείο χρησιμοποιώντας το χαρακτηριστικό “style”.

```
<p style="color: red; font-size: 20px;">This is a paragraph.</p>
```

Απόσπασμα κώδικα 2: Inline CSS

Internal CSS: Χρήση του κώδικα σε μια ετικέτα <style> μέσα στην ενότητα <head> ενός εγγράφου HTML.

```
<head>
    <style>
        p {
            color: blue;
            font-size: 18px;
        }
    </style>
</head>
```

Απόσπασμα κώδικα 3: Internal CSS

External CSS: Μέσω της εισαγωγής ενός αρχείου CSS χρησιμοποιώντας την ετικέτα <link>. Αυτή η μέθοδος συνήθως προτιμάται σε μεγαλύτερα πρότζεκτ για την καλύτερη διαχείριση του κώδικα.

```
<head>
    <link rel="stylesheet" type="text/css" href="styles.css">
</head>
```

Απόσπασμα κώδικα 4: External CSS

2.2.3 SASS

Η SASS (Syntactically Awesome Style Sheets) είναι μια γλώσσα προεπεξεργασίας που ερμηνεύεται ή μεταγλωττίζεται σε CSS. Παρέχει σε αυτήν περισσότερη ισχύ και κομψότητα για τη μορφοποίηση ιστοσελίδων. Η SASS επεκτείνει την απλή CSS με χαρακτηριστικά όπως οι μεταβλητές, οι εμφωλευμένοι κανόνες, τα mixins, οι συναρτήσεις και άλλα, καθιστώντας το styling μιας ιστοσελίδας πιο εύκολο στη ανάπτυξη αλλά και στην συντήρηση [9].

2.2.3.1 Συντακτικό

Η SASS έχει δύο διαφορετικά συντακτικά:

Σύνταξη SASS: Αποτελεί την κλασική σύνταξη, είναι πιο συνοπτική και χρησιμοποιεί εσοχή αντί για αγκύλες και ερωτηματικά για να ορίζει τα τμήματα κώδικα. Τα αρχεία SASS χρησιμοποιούν την επέκταση ".sass".

```
$primary-color: #333

body
  font-family: Arial, sans-serif
  color: $primary-color

.container
  width: 100%
  margin: 0 auto
```

Απόσπασμα κώδικα 5: Σύνταξη SASS

Σύνταξη SCSS: Η SCSS (Sassy CSS) είναι ένα υπερσύνολο της CSS, δηλαδή είναι πιο κοντά στην τυπική σύνταξη της CSS. Χρησιμοποιεί αγκύλες "{}" και ερωτηματικά ";" για να διαχωρίσει τα τμήματα και τις δηλώσεις, κάνοντάς το πιο οικείο σε προγραμματιστές που είναι ήδη εξοικειωμένοι με την CSS. Τα αρχεία SCSS χρησιμοποιούν την επέκταση ".scss". Αυτή η σύνταξη προτιμήθηκε στην ανάπτυξη της εφαρμογής.

```
$primary-color: #333;

body {
  font-family: Arial, sans-serif;
  color: $primary-color;
}

.container {
  width: 100%;
  max-width: 1200px;
  margin: 0 auto;
}
```

Απόσπασμα κώδικα 6: Σύνταξη SCSS

2.2.4 JavaScript

Η JavaScript είναι μία γλώσσα προγραμματισμού που αντιπροσωπεύει μία από τις τρεις βασικές τεχνολογίες που χρησιμοποιούνται στην ανάπτυξη ιστοσελίδων, μαζί με την HTML και την CSS. Ενώ οι τελευταίες καθορίζουν τη δομή και το στυλ του ιστότοπου, η JavaScript είναι αυτή που με την χρήση της δίνει λειτουργικότητα σε αυτόν. Συχνά συνδέεται με την Java αλλά παρόλο που αντλεί ονόματα και κανόνες ονοματοδοσίας από αυτή, η JavaScript έχει διαφορετική σημασιολογία και δεν σχετίζεται στενά με αυτήν. Η σύνταξή της μοιάζει με αυτή της C και έχει προέλευση από τις γλώσσες προγραμματισμού Self και Scheme.

Βασικό γνώρισμα της είναι ότι επιτρέπει την δημιουργία δυναμικών ιστοσελίδων, όπου οι χρήστες μπορούν να αλληλεπιδρούν με το περιεχόμενο χωρίς την ανάγκη ανανέωσης της σελίδας. Σε αυτή την εμπειρία που προσφέρεται στον χρήστη, καθοριστικό ρόλο παίζει η διαχείριση συμβάντων (events) τα οποία μπορούν να χρησιμοποιηθούν σε HTML στοιχεία για την άμεση αλληλεπίδραση με την διεπαφή. Μερικά από τα πιο συχνά παραδείγματα είναι το κλικ, το πάτημα πλήκτρου, ή η υποβολή μιας φόρμας.

Στο πιο λειτουργικό κομμάτι της, η JavaScript είναι μία γλώσσα βασισμένη σε αντικείμενα, όπου η χρήση αυτών συμβάλει στην οργάνωση του κώδικα. Επιπρόσθετα, παρέχει πολλές ενσωματωμένες λειτουργίες για τη διαχείριση συμβολοσειρών και αριθμών, καθώς και δυνατότητες για τη διεξαγωγή μαθηματικών πράξεων.

Δεν θα μπορούσαμε να παραλείψουμε την δυνατότητα επικοινωνίας που μας δίνει η JavaScript με τον διακομιστή, για την ανάκτηση και την αποστολή δεδομένων χωρίς να καθίσταται απαραίτητη η ανανέωση της ιστοσελίδας. Η δυνατότητα αυτή σε συνδυασμό με τον ασύγχρονο προγραμματισμό που συνδράμει στην διαχείριση της σειράς εκτέλεσης εργασιών, αλλά και τους μηχανισμούς διαχείρισης σφαλμάτων και εξαιρέσεων, προσφέρουν ένα πλήρες πακέτο εργαλείων διαχείρισης ακόμα και των πιο πολύπλοκων api [10].

2.2.4.1 Ιστορία

Κατά τα πρώτα χρόνια του διαδικτύου, οι ιστοσελίδες ήταν μόνο στατικές, χωρίς δυνατότητα δυναμικής συμπεριφοράς και διαδραστικότητας. Αυτό δημιούργησε στην κοινότητα ανάπτυξης ιστού εκείνης της εποχής την επιθυμία να ξεπεράσει αυτόν τον περιορισμό. Έτσι τον Σεπτέμβριο του 1995, ένας προγραμματιστής της εταιρείας NetScape (σημερινή Mozilla) [11] ονόματι Brendan Eich ανέπτυξε μια νέα γλώσσα σεναρίων (scripting language) σε μόλις 10 ημέρες. Αρχικά ονομαζόταν Mocha, αλλά γρήγορα έγινε γνωστό ως LiveScript και, αργότερα, JavaScript. Η γλώσσα άντλησε τη σύνταξή της από την Java, τις πρώτης τάξεως συναρτήσεις της από την Scheme και την πρωτότυπη κληρονομικότητά της από την Self. Από τότε, η JavaScript έχει υιοθετηθεί από όλα τα μεγάλα γραφικά προγράμματα περιήγησης ιστού.

2.2.5 Typescript

Η TypeScript είναι ένα συντακτικό υπερσύνολο της JavaScript το οποίο προσθέτει στατικές δηλώσεις τύπου. Αυτό σημαίνει ότι μοιράζονται την ίδια βασική σύνταξη απλά σε αυτή δηλώνεις τύπους όπως το string, το number, τον void για δήλωση τύπου συνάρτησης, αλλά και custom types που έχουν σκοπό να περιγράψουν κάποιο object που δημιουργήσαμε. Στην JS, οι παράμετροι και οι μεταβλητές συνάρτησης δεν έχουν καμία πληροφορία το οποίο συχνά οδηγεί στους προγραμματιστές να υποπέσουν σε σφάλματα κώδικα, τα οποία ανακαλύπτουν κατά την εκτέλεση του κώδικα. Αντίθετα η TS, προσφέρει στο πρόγραμμα επεξεργασίας την δυνατότητα να εντοπίζει άμεσα τα σφάλματα όταν οι τύποι δεν ταιριάζουν και γίνεται επισήμανση αυτών στο κώδικα. Έτσι γίνεται εξοικονόμηση πολύτιμου χρόνου αποσφαλμάτωσης στα τυφλά [12].

Τα αποσπάσματα κώδικα που ακολουθούν αποτελούν ένα πολύ απλό παράδειγμα για το πόσο εύκολο είναι να πέσουμε σε κάποιο σφάλμα λόγω χρήσης λανθασμένου τύπου σε παραμέτρους μεθόδου, αλλά και το πόσο εύκολο είναι για την TS να εντοπίσει το λάθος αυτό:

Στο συγκεκριμένο παράδειγμα, λανθασμένα δόθηκε το string ‘2’ σαν παράμετρος στην μέθοδο πρόσθεσης, έτσι, ως αποτέλεσμα πήραμε λανθασμένα ‘23’.

```
function add(a, b) {
    return a + b;
}

console.log(add(2, 3)); // 5
console.log(add('2', 3)); // '23' - Potential bug
```

Απόσπασμα κώδικα 7: Type error on JS

Αντίθετα σε αυτή την περίπτωση που έχουμε την ίδια τιμή εισόδου, επειδή έχουν δηλωθεί οι τύποι των τιμών παραμέτρων, λόγω της TS παίρνουμε το error που ακολουθεί.

```
function add(a: number, b: number): number {
    return a + b;
}

console.log(add(2, 3)); // 5
console.log(add('2', 3)); // Error: Argument of type 'string' is
not assignable to parameter of type 'number'
```

Απόσπασμα κώδικα 8: Type error on TS

Τέλος, η TS μπορεί να εκτελεστεί οπουδήποτε εκτελείται η JS, καθώς και σε παλαιότερες εκδόσεις της, με αποτέλεσμα την συμβατότητα σε παλαιότερα προγράμματα περιήγησης.

2.2.6 Vue.js

Στις μέρες μας, η ανάπτυξη ιστού στο front-end κομμάτι έχει εξελιχθεί σημαντικά και η χρήση καθαρής HTML, CSS και JavaScript όλο και μειώνεται. Αντί αυτών, οι προγραμματιστές αξιοποιούν μια ποικιλία από frameworks, βιβλιοθήκες και εργαλεία για να δημιουργήσουν πιο δυναμικές, αποτελεσματικές και συντηρήσιμες εφαρμογές ιστού. Στην JavaScript συγκεκριμένα, βρίσκεις μια μεγάλη γκάμα από frameworks τα οποία παρέχουν στους προγραμματιστές μια δομή και ένα σύνολο εργαλείων που συμβάλλουν στην ορθολογική ανάπτυξη εφαρμογών, προσφέροντας επαναχρησιμοποιήσιμα στοιχεία, απλοποιώντας πολύπλοκες εργασίες και επιβάλλοντας βέλτιστες πρακτικές [13]. Μία από τις πολλές επιλογές που έχουν οι προγραμματιστές για την ανάπτυξη του UI εφαρμογών, είναι η Vue η οποία επιλέχθηκε για την υλοποίηση του Alumni και συγκεκριμένα η τρίτη έκδοση της.

Η Vue είναι ένα από τα πιο δημοφιλή και ευρέως χρησιμοποιούμενα frameworks για την ανάπτυξη front-end εφαρμογών. Δημιουργήθηκε από τον Evan You [14] και η πρώτη ολοκληρωμένη έκδοση της της κυκλοφόρησε το 2014. Από τότε, έχει αυξηθεί σημαντικά όσον αφορά την υποστήριξη της κοινότητας, την ανάπτυξη του οικοσυστήματος της και την υιοθέτηση από προγραμματιστές και εταιρείες. Η επίτευξη αυτού οφείλεται στην απλότητα, την ευελιξία και την ευκολία ενσωμάτωσής της με άλλα έργα και βιβλιοθήκες.

2.2.6.1 Template Syntax

Μια εφαρμογή γραμμένη σε Vue αποτελείται από πολλαπλά Vue components, δηλαδή αρχεία που έχουν την κατάληξη ".vue". Βλέποντας το περιεχόμενο ενός τέτοιου αρχείου παρατηρεί κάποιος ότι χωρίζεται από τρία μέρη. Αυτά είναι το `<template>`, το `<script>` και το `<style>`.

Ξεκινώντας με το template, αποτελεί το κομμάτι που προβάλλεται σε μια ιστοσελίδα και χρησιμοποιεί την DOM μορφή της HTML.

```
<template>
  <div>
    <h1>Hello, Vue!</h1>
    <p>This is a paragraph in a Vue component.</p>
  </div>
</template>
```

Απόσπασμα κώδικα 9: Παράδειγμα σύνταξης ενός Vue template

Συνεχίζοντας, το script αποτελεί το σημείο του κώδικα όπου λαμβάνουν μέρος όλες οι λειτουργικότητες του αρχείου. Ο προγραμματιστής έχει την δυνατότητα να επιλέξει για την συγγραφή αυτού του κομματιού την JavaScript, αλλά και την TypeScript, καθώς υποστηρίζεται από την Vue, απλά δηλώνοντας την με τον εξής τρόπο: `<script lang="ts">`.

Άξιο αναφοράς είναι, πως αναλόγως με την έκδοση της Vue που θα χρησιμοποιηθεί για την υλοποίηση της εκάστοτε εφαρμογής, διαφέρει σε ένα σημαντικό βαθμό η σύνταξη του κώδικα στο script κομμάτι. Στην έκδοση 2 γινόταν η χρήση του Options Api ενώ με την έλευση της έκδοσης 3 αντικαταστάθηκε με το Composition Api, παρόλο που υπάρχει δυνατότητα χρήσης και των δύο [15]. Αυτή η μετάβαση

σε Composition Api οφείλεται στην απλούστευση του κώδικα καθώς οι γραμμές του λιγοστεύουν σημαντικά με φυσικό επόμενο ο κώδικας να γίνεται πιο ευανάγνωστος. Στην πράξη, η βασική διαφορά τους είναι στο τμήμα setup, το οποίο είναι υπεύθυνο για τις λειτουργίες του component. Ακολουθούν αποσπάσματα κώδικα που εκτελούν την ίδια λειτουργία υλοποιημένες και με τους δύο τρόπους.

```
<script>
import { ref } from 'vue'

export default {
  setup() {
    const count = ref(0)

    function increment() {
      // .value is needed in JavaScript
      count.value++
    }

    // don't forget to expose the function as well.
    return {
      count,
      increment
    }
  }
}
</script>
```

Απόσπασμα κώδικα 10: Options Api syntax

Σε αντίθεση με το Options Api όπου το export του setup γίνεται χειροκίνητα και χρειάζεται η επιστροφή τον μεταβλητών αλλά και των μεθόδων, στο Composition Api μπορείς να τα αποφύγεις όλα αυτά απλά κάνοντας την χρήση του `<script setup>` [15].

```
<script setup>
import { ref } from 'vue'

const count = ref(0)

function increment() {
  count.value++
}
</script>
```

Απόσπασμα κώδικα 11: Composition Api syntax

Όσον αφορά το style κομμάτι, είναι κοινό με το `<style>` που χρησιμοποιείται και στην καθαρή HTML με την δυνατότητα χρήσης CSS αλλά και επεκτάσεις αυτής όπως η SASS και η SCSS που αναφέρθηκαν σε προηγούμενο κεφάλαιο (2.2.3 SASS).

2.2.6.2 Reactivity

Ένα από τα κύρια χαρακτηριστικά της Vue είναι η αντιδραστικότητα (reactivity). Η Vue παρακολουθεί αυτόματα τις αλλαγές κατάστασης της JavaScript και ενημερώνει αποτελεσματικά το DOM όταν συμβαίνουν αλλαγές.

Ο τρόπος με τον οποίο επιτυγχάνεται αυτό είναι με την χρήση της μεθόδου ref() η οποία μετατρέπει τις μεταβλητές σε δυναμικές και τις επιτρέπει να αλλάζουν περιεχόμενο. Πρακτικά παίρνει το όρισμα και το επιστρέφει μέσα σε ένα αντικείμενο (object) ref με την ιδιότητα ".value". Ωστόσο, για να έχεις πρόσβαση στην τιμή μιας ref μέσα στο template δεν χρειάζεται να ακολουθήσει το ".value" [15]. Παράδειγμα χρήσης της το Απόσπασμα κώδικα 10 και Απόσπασμα κώδικα 11 όπου η τιμή του counter έπρεπε να αυξάνεται δυναμικά με την χρήση της increment μεθόδου. Μια ref μπορεί να κρατήσει οποιονδήποτε τύπο τιμής συμπεριλαμβανομένων και των βαθιά ένθετων αντικειμένων (deeply nested objects). Χρησιμοποιώντας την ref το αντικείμενο γίνεται βαθιά δυναμικό, το οποίο σημαίνει ότι μπορούν να γίνουν αλλαγές μέχρι και στο τελευταίο στοιχείο του αντικειμένου αλλά και των πινάκων.

2.2.6.3 Computed Properties

Μια ακόμη πολύ ισχυρή λειτουργία που προσφέρει η Vue στο τομέα του reactivity είναι οι computed properties. Αποτελούν βασικό χαρακτηριστικό για την εξαγωγή τιμών με βάση άλλα δεδομένα. Παρόλο που μπορεί κάποιος να έχει πρόσβαση στην τιμή της, χρησιμοποιώντας ομοίως με τις ref το ".value" μετά το όνομα, δεν μπορεί να αλλάξει την τιμή της με τον ίδιο τρόπο. Αυτό μπορεί να επιτευχθεί μόνο αν αλλάξει η τιμή κάποιας μεταβλητής που αποτελεί μέρος του περιεχομένου της computed [15].

```
import { ref, computed } from 'vue'

const age = ref(15)

const adult = computed(() => {
  return age.value  >= 18 ? 'Yes' : 'No'
})

console.log(adult.value) // 'No'

age.value = 18
console.log(adult.value) // 'Yes'
```

Απόσπασμα κώδικα 12: Computed Method

2.2.6.4 Components

Βασικό γνώρισμα της Vue, αποτελεί η δυνατότητα που δίνει στον προγραμματιστή να δημιουργεί επαναχρησιμοποιούμενα κομμάτια διεπαφής χρήστη (components). Πρόκειται για κομμάτια κώδικα μορφής Vue, τα οποία μπορούν είτε να δηλωθούν ως global στη εφαρμογή είτε να καταχωρηθούν τοπικά σε άλλα components και να χρησιμοποιηθούν ως κλασικά HTML στοιχεία. Τα components μπορούν είτε να είναι ανεξάρτητα μεταξύ τους, είτε να επικοινωνούν μέσω των props. Τα props επιτρέπουν την μεταβίβαση δεδομένων από το parent component στο child component κάνοντας τα δυναμικά. Ένας ακόμη τρόπος επικοινωνίας μεταξύ των components είναι η χρήση συμβάντων (events) η οποία θα αναλυθεί σε επόμενο κεφάλαιο. Πέρα των πλεονεκτημάτων που αναφέρθηκαν μέχρι τώρα, με την χρήση των components ο προγραμματιστής δημιουργεί στοιχεία τα οποία μπορούν να

επαναχρησιμοποιηθούν σε διαφορετικά μέρη της εφαρμογής, γεγονός που μειώνει την αντιγραφή κώδικα και βελτιώνει τη συντηρησιμότητα.

2.2.6.5 Class and Style Bindings

Για να χαρακτηριστεί μια εφαρμογή εύχρηστη παίζουν ρόλο πολλοί παράγοντες. Ένας από αυτούς είναι το οπτικό κομμάτι της εφαρμογής το οποίο και καθορίζει άμεσα την εμπειρία του χρήστη καθώς αλληλοεπιδρά με αυτή. Για την επίτευξη ενός άρτιου οπτικά αποτελέσματος ο προγραμματιστής πρέπει να έχει τα κατάλληλα εργαλεία που θα του διευκολύνουν το έργο.

Για καλή του τύχη η Vue έχει αρκετούς τρόπους για την ανάθεση CSS μορφοποιήσεων στο UI, με τον πιο κλασικό εξ αυτών την χρήση inline σύνταξης χρησιμοποιώντας τα γνωρίσματα style και class. Παρόλο που αυτός ο τρόπος συναντάται και στην απλή HTML (Απόσπασμα κώδικα 2), στην περίπτωση της Vue γίνεται σε συνδυασμό με την χρήση του v-bind όπου εκχωρεί μια τιμή συμβολοσειράς δυναμικά [15]. Με τον τρόπο αυτό μετατρέπονται σε γνωρίσματα τύπου prop και έχουν την μορφή `:style` και `:class`. Πλέον υπάρχει η δυνατότητα δυναμικής εναλλαγής κλάσεων με την χρήση αλφαριθμητικών μεταβλητών, αντικειμένων αλλά και πινάκων.

Στο παράδειγμα που ακολουθεί γίνεται η χρήση αλφαριθμητικής τιμής δυναμικά στο γνώρισμα `:style`.

```
<template>
  <div :style="'color: ' + textColor">This is an error message</div>
</template>

<script setup>
import { ref } from 'vue'

const textColor = ref('red')
</script>
```

Απόσπασμα κώδικα 13: Dynamic inline style με την χρήση αλφαριθμιτικής τιμής

Αυτό που θα αποδοθεί εν τέλει είναι:

```
<div style=”color: red”></div>
```

Σε αντίθεση με το προηγούμενο παράδειγμα, εδώ γίνεται η χρήση αντικειμένου στο γνώρισμα `:class`.

```
<template>
  <div :class="{ 'text-danger': hasError }">This is an error
message</div>
</template>

<script setup>
import { ref } from 'vue'

const hasError = ref(true)
</script>
```

Απόσπασμα κώδικα 14: Dynamic inline class με την χρήση αντικειμένου

Η τιμή που θα λάβει η κλάση είναι:

```
<div class="text-danger"></div>
```

Ομοίως με τα παραπάνω η χρήση πινάκων έχει την ακόλουθη δομή:

```
<template>
  <div
    :class="[
      isHyperlink ? 'underline' : 'no-underline',
      isHyperlink && visitedBefore ? 'purple' : 'black'
    ]"
  >
    This is some random text
  </div>
</template>

<script setup>
import { ref } from 'vue'

const isHyperlink = ref(true)
const visitedBefore = ref(false)
</script>

<style>
.underline {
  text-decoration: underline;
}

.no-underline {
  text-decoration: none;
}

.purple {
  color: purple;
}

.black {
  color: black;
}
</style>
```

Απόσπασμα κώδικα 15: Dynamic inline class με την χρήση πίνακα

Εκτός της inline σύνταξης η Vue υιοθετεί και τη χρήση του style μέρους ενός Vue κώδικα, αλλά με την επιπλέον δυνατότητα χρήσης του ορίσματος scoped. Όταν το style τμήμα έχει ενεργοποιημένο το scoped όρισμα `<style scoped>`, όλες οι αλλαγές θεωρούνται τοπικές και επηρεάζουν μόνο το περιεχόμενο του συγκεκριμένου component. Με αυτόν τον τρόπο, αποφεύγονται προβλήματα από κλάσεις της εφαρμογής που μπορεί να έχουν το ίδιο όνομα αλλά να χρειάζονται διαφορετικές μορφοποιήσεις. Επιπλέον, ένα Vue αρχείο μπορεί να περιλαμβάνει περισσότερα από ένα style τμήματα, το καθένα από τα οποία μπορεί να είναι σε διαφορετική γλώσσα (π.χ. SCSS ή SASS) ή να έχει το όρισμα scoped.

2.2.6.6 Directives

Ένα σύγχρονο και συνεχώς αναπτυσσόμενο framework όπως η Vue δεν θα μπορούσε να παραλείψει την δυνατότητα διαχείρισης στοιχείων του DOM. Για τον λόγω αυτό έχει ενσωματωμένα directives.

Μεταξύ αυτών, το `v-model` ξεχωρίζει καθώς προσφέρει στα αμφίδρομη σύνδεση δεδομένων μεταξύ μιας εισαγωγής φόρμας ή ενός component και των δεδομένων του script μέρους ενός Vue αρχείου. Αυτό σημαίνει πως με οποιαδήποτε αλλαγή σε μια τιμή εισόδου από το χρήστη θα ενημερωθεί αντίστοιχα και η μεταβλητή που έχει δηλωθεί στο script αλλά και το αντίστροφο. Συνήθως συναντάται σε στοιχεία φόρμας όπως `<input>`, `<textarea>` και `<select>` [15].

```
<template>
  <input v-model="username" placeholder="Enter username" />
</template>

<script setup>
import { ref } from 'vue'

const username = ref('')
</script>
```

Απόσπασμα κώδικα 16: Παράδειγμα χρήσης v-model

Δεν θα μπορούσε να μην γίνει αναφορά στο `v-if`, το οποίο προσφέρει ένα απλό μέσο απόδοσης στοιχείων (rendering) υπό όρους με βάση συγκεκριμένες συνθήκες, επιτρέποντας δυναμική παρουσίαση περιεχομένου, προσαρμοσμένη σε διαφορετικά σενάρια. Όταν η συνθήκη αποτιμάται σε true, τότε το στοιχείο γίνεται render και ο χρήστης το βλέπει στην οθόνη του, αντίθετα το στοιχείο καταργείται από το DOM [15].

```
<template>
  <div v-if="error">
    An error has occured
  </div>
</template>

<script setup>
import { ref } from 'vue'

const error = ref(true)
</script>
```

Απόσπασμα κώδικα 17: Παράδειγμα χρήσης v-if

Το συγκεκριμένο directive μπορεί να επεκταθεί δημιουργώντας εμφωλευμένες συνθήκες για να προσφέρει την δυνατότητα πολλαπλών ελέγχων απευθείας στο DOM. Αυτό επιτυγχάνεται με την χρήση του `v-else-if` και του `v-else` με τον τρόπο σύνταξης στο απόσπασμα κώδικα που ακολουθεί.

```
<template>
  <div v-if="homeTeamGoals > awayTeamGoals">
    The winner is home team
  </div>
  <div v-else-if="homeTeamGoals < awayTeamGoals">
    The winner is away team
  </div>
  <div v-else>
    We have a draw
```

```
  </div>
</template>

<script setup>
import { ref } from 'vue'

const homeTeamGoals = ref(2)
const awayTeamGoals = ref(1)
</script>
```

Απόσπασμα κώδικα 18: Παράδειγμα χρήσης v-else-if & v-else

Παρόμοια λειτουργία με το `v-if` έχει το `v-show`, με την διαφορά ότι αντί να προσθέτει και να αφαιρεί στοιχεία από το DOM, εναλλάσσει την ιδιότητα εμφάνισης CSS για να αποκρύψει ή να εμφανίσει στοιχεία με βάση την αλήθεια της έκφρασης.

Όσον αφορά την επανάληψη δεδομένων και την απόδοση δυναμικού περιεχομένου, είναι η σειρά της `v-for` να λάμψει, καθώς δημιουργεί αβίαστα επαναλαμβανόμενα στοιχεία με βάση δεδομένων πινάκων ή αντικείμενα. Σε συνδυασμό με την `v-for` συνιστάται η χρήση του `:key` το οποίο προσφέρει μοναδική ταυτότητα σε κάθε στοιχείο που δημιουργείται κατά την επανάληψη.

```
<template>
  <li v-for="(item, index) in array" :key="index">
    {{ item }}
  </li>
</template>

<script setup>
import { ref } from 'vue'

const array = ref(['red', 'blue', 'green'])
</script>
```

```
<template>
  <li v-for="(value, key) in object" :key="key">
    {{ value }}
  </li>
</template>

<script setup>
import { ref } from 'vue'

const object = ref({ 1: 'red', 2: 'blue', 3: 'green' })
</script>
```

Απόσπασμα κώδικα 19: Παράδειγμα χρήσης v-for

Καθοριστικής σημασίας directive για την αλληλεπίδραση του χρήστη με το UI είναι v-on το οποίο δίνει την δυνατότητα ακρόασης συμβάντων στο HTML στοιχείο όπως κλικ, πατήματα πλήκτρων ή υποβολές φορμών. Λεπτομερής ανάλυση του τρόπου χρήσης του v-on και της διαχείρισης των συμβάντων θα γίνει στο κεφάλαιο που ακολουθεί (2.2.6.7 Event Handling).

2.2.6.7 Event Handling

Στην Vue, ο χειρισμός συμβάντων (events) αποτελεί πτυχή της δημιουργίας διαδραστικών εφαρμογών. Όπως έχει ήδη αναφερθεί, ο τρόπος δημιουργίας κάποιου event σε ένα HTML element ή σε ένα component, είναι η χρήση του v-on η οποία εκφράζεται και με το σύμβολο @ για συντομογραφία [15]. Οπότε στην πράξη, αν θέλουμε να ενεργοποιήσουμε ένα event χρησιμοποιώντας το κλικ, τότε θα προσθέσουμε το `v-on:click="handleEvent"` ή πιο σύντομα `@click="handleEvent"`.

Ο τρόπος χρήσης των event μπορεί να γίνει με δύο διαφορετικούς τρόπους:

- **Inline χειρισμός των event:** Ενσωματωμένο JavaScript στο HTML element (λόγω του `v-on`) που εκτελείται κατά την ενεργοποίηση του συμβάντος.

```
<template>
  <button @click="count++">Add 1</button>
</template>
```

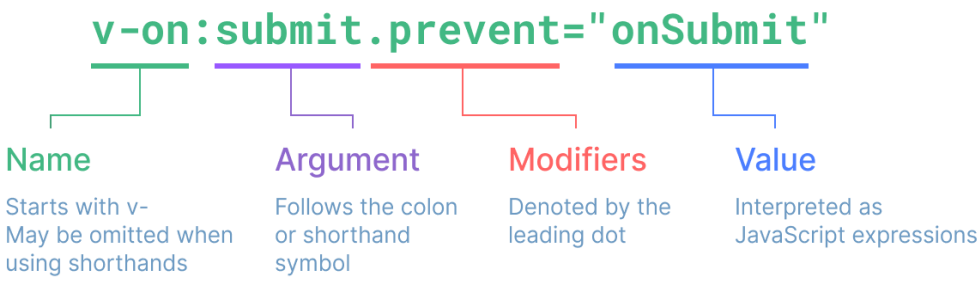
Απόσπασμα κώδικα 20: Παράδειγμα inline event handler

- **Χειρισμός με την χρήση μεθόδου:** Ο προγραμματιστής έχει την δυνατότητα να χρησιμοποιήσει σαν είσοδο στο event μια μέθοδο η οποία θα τρέξει όταν αυτό ενεργοποιηθεί.

```
<template>
  <button @click="countFn">Add 1</button>
</template>

<script setup>
import { ref } from 'vue'

const count = ref(0)

const countFn = () => {
  count.value++
}
</script>
```

Απόσπασμα κώδικα 21: Παράδειγμα method event handler

Συχνό φαινόμενο είναι η ανάγκη χρήσης της `event.preventDefault()` ή `event.stopPropagation()` μέσα σε προγράμματα χειρισμού συμβάντων για να αποφύγουμε κάποιες προκαθορισμένες συμπεριφορές. Παρόλο που δεν τίθεται θέμα δυσκολίας στην υλοποίηση μέσα στις μεθόδους, η Vue προσφέρει μια πολύ πιο εύκολη και γρήγορη λύση μέσω των τροποποιητών (modifiers). Οι τροποποιητές χρησιμοποιούνται ως επιθέματα οδηγιών που υποδηλώνονται με τελεία.



Εικόνα 1: Complete event syntax [15]

Με τον τρόπο αυτό οι μέθοδοι πλέον, αφορούν καθαρά την λογική των δεδομένων αντί να χρειάζεται να ασχοληθούν με λεπτομέρειες συμβάντων DOM. Μερικοί ακόμη modifiers που χρησιμοποιούνται την περίσταση είναι οι ".stop", ".self", ".capture", ".once" και ".passive". Ένα ενδιαφέρον χαρακτηριστικό είναι η δυνατότητα συνδυασμού δημιουργώντας μία αλυσίδα από modifiers.

```
<a @click.stop.prevent="doThat"></a>
```

Απόσπασμα κώδικα 22: Παράδειγμα χρήσης chained modifiers

Μια κατηγορία τροποποιητών είναι οι key modifiers. Πρόκειται για τους `@keyup` και `@keydown` οι οποίοι σε συνδυασμό με μια λίστα από key aliases χειρίζονται events με το πάτημα συγκεκριμένων κουμπιών. Και σε αυτή την περίπτωση υπάρχει δυνατότητα χρήσης αλυσίδας από aliases για την δημιουργία event με το πάτημα κάποιου συνδυασμού κουμπιών από τον χρήση.

```
<!-- Alt + Enter -->
<input @keyup.alt.enter="clear" />
```

Απόσπασμα κώδικα 23: Παράδειγμα χρήσης chained aliases

Αντίστοιχα, υπάρχουν και οι mouse button modifiers οι οποίοι ξεχωρίζουν τα κουμπιά του ποντικιού και ενεργοποιούν τα συμβάντα αναλόγως με την επιλογή του επιθέματος.

```
<!-- Handle left mouse button click -->
<button @click.left="handleLeftClick">Left Click</button>
```

Απόσπασμα κώδικα 24: Παράδειγμα χρήσης mouse button modifier

2.2.6.8 Lifecycle Hooks

Κάθε instance ενός Vue component περνά από διάφορα στάδια όταν δημιουργείται. Για παράδειγμα, πρέπει να ρυθμίσει την παρακολούθηση των δεδομένων, να προετοιμάσει το template, να συνδεθεί με το DOM και να ενημερώσει το DOM όταν τα δεδομένα αλλάζουν. Κατά τη διάρκεια αυτών των σταδίων, η Vue έχει την δυνατότητα να τρέξει ειδικές συναρτήσεις γνωστές ως lifecycle hooks, που επιτρέπουν στους χρήστες να επέμβουν σε διάφορα στάδια του κύκλου ζωής ενός Vue component. Έχοντας αυτό το προνόμιο μπορούν προσθέσουν δικό τους κώδικα σε συγκεκριμένα σημεία της

διαδικασίας αυτής, όπως όταν το component δημιουργείται, εισάγεται στο DOM, ενημερώνεται ή καταστρέφεται. Ταξινομώντας τα σύμφωνα με την σειρά εκτέλεσης τους έχουμε την εξής λίστα:

- **beforeCreate:** Καλείται πριν δημιουργηθεί το component. Σε αυτό το στάδιο, τα δεδομένα του component δεν είναι ακόμη αντιδραστικά και τα γεγονότα δεν έχουν ρυθμιστεί.

- **created**: Εκτελείται μετά τη δημιουργία του component αλλά πριν τοποθετηθεί στο DOM. Είναι χρήσιμο για εκτέλεση λειτουργιών που απαιτούν να έχει δημιουργηθεί πλήρως το instance του component.

- **beforeMount**: Εκτελείται λίγο πριν το component τοποθετηθεί στο DOM. Είναι καλή στιγμή για να προετοιμαστεί το component για την αρχική του απόδοση.

- **mounted**: Καλείται αφού το component τοποθετηθεί στο DOM. Εδώ υπάρχει δυνατότητα άμεσης αλληλεπίδρασης με το DOM, για παράδειγμα, μπορεί να λάβει χώρα η αρχικοποίηση κάποιον plugins ή η ανάκτηση δεδομένων.

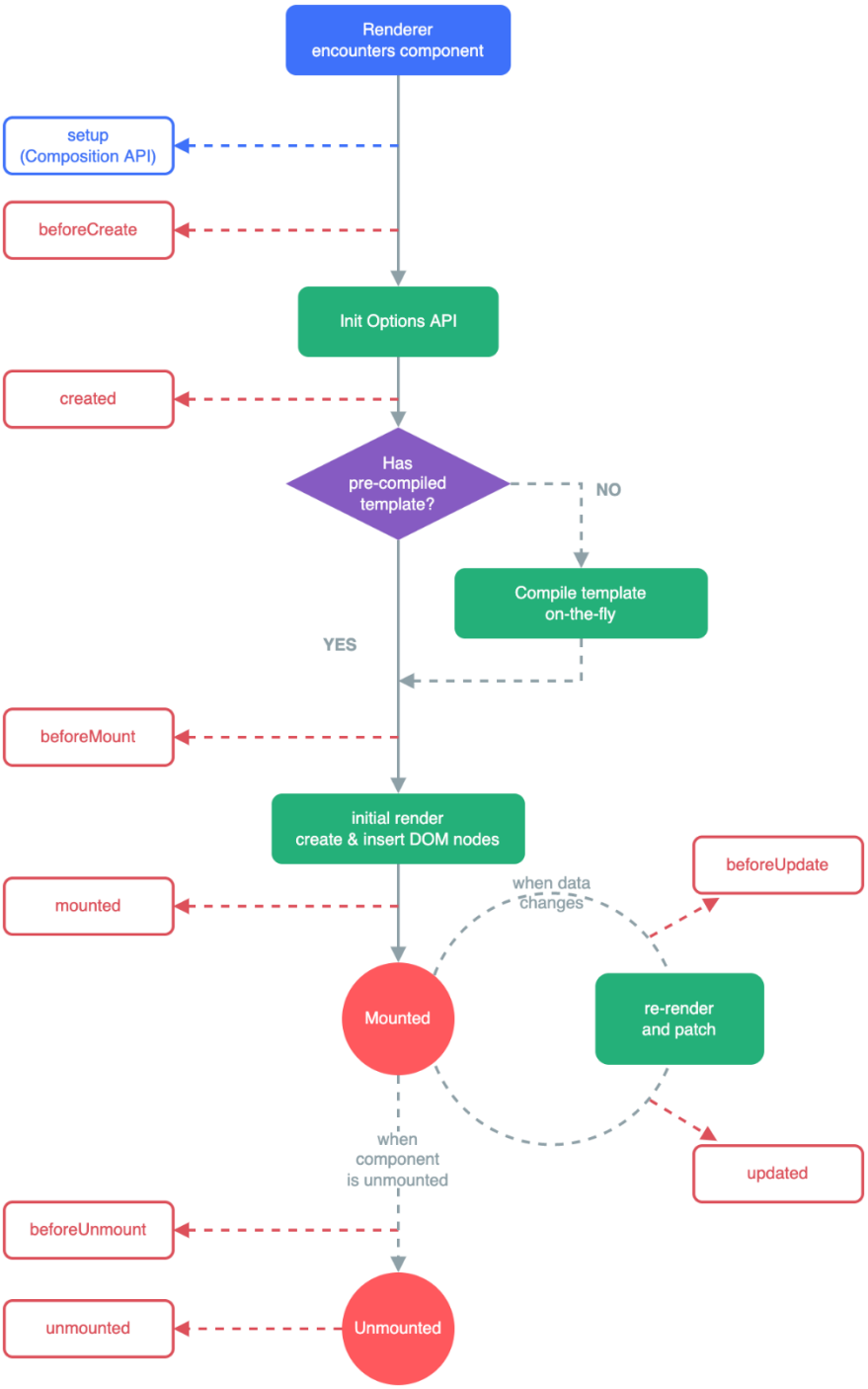
- **beforeUpdate**: Πυροδοτείται πριν αλλάξουν τα δεδομένα του component και ενημερωθεί το DOM. Παρέχει την ευκαιρία να γίνει αντίδραση σε αλλαγές δεδομένων πριν αυτές εφαρμοστούν στην προβολή.

- **updated**: Εκτελείται μετά την αλλαγή των δεδομένων του component και την ενημέρωση του DOM. Είναι χρήσιμο για εκτέλεση ενεργειών μετά την ενημέρωση της προβολής.

- **beforeDestroy**: Καλείται λίγο πριν καταστραφεί ένα component. Είναι μια ευκαιρία να εκτελεστούν τελικές ενέργειες πριν το component αφαιρεθεί από το DOM.

- **destroyed**: Εκτελείται αφού το component καταστραφεί. Εδώ μπορείτε να απελευθερώσετε πόρους που συνδέονται με το component.

Στην εικόνα που ακολουθεί (Εικόνα 2) γίνεται μία σχηματική αναπαράσταση ώστε να γίνει πιο κατανοητή η σειρά των λειτουργιών σε ένα Vue component.



Εικόνα 2: Vue Lifecycle Diagram [15]

2.2.6.9 Watchers

Οποιοσδήποτε έχει φτάσει σε αυτό το σημείο του κειμένου αβίαστα μπορεί να καταλάβει το πόσο καθοριστικό είναι το κομμάτι του reactivity στην Vue. Μία από τις σημαντικές λειτουργίες που μας προσφέρει για την παρακολούθηση των αλλαγών στα δεδομένα είναι η watch function. Αυτή η λειτουργία μας επιτρέπει να εκτελούμε συγκεκριμένες ενέργειες ως απάντηση σε αλλαγές συγκεκριμένων δεδομένων ή υπολογισμένων ιδιοτήτων. Σε αντίθεση με τα computed properties, που υπολογίζουν και επιστρέφουν τιμές με βάση τα εξαρτώμενα δεδομένα, οι watcher ενεργούν πιο άμεσα και μας επιτρέπουν να εκτελούμε side effects όταν παρατηρούνται αλλαγές. Όσον αφορά την δομή της,

δέχεται σαν πρώτη παράμετρο την οντότητα που θέλουμε να παρακολουθήσουμε, είτε αυτή είναι μία απλή ref, είτε αντικείμενο, είτε πίνακας πολλαπλών δεδομένων, ακόμη και κάποιας αριθμητικής πράξης ή συνθήκης αλήθειας.

```
<script setup>
import { ref, watch } from 'vue'

const item = ref() // the item that we want to track

watch(item, () => {
  // when the item value changes do some action
})

const x = ref(1)
const y = ref(1)

watch(
  () => x.value === y.value,
  (condition) => {
    console.log(`the condition of x === y is: ${condition}`)
  }
)

watch([x, y], () => {
  console.log('there is a change on the value of x or the value of y')
})

const obj = ref({ count: 0 });

watch(
  () => obj.value.count,
  (count) => {
    console.log(`Count is: ${count}`);
  }
);
</script>
```

Απόσπασμα κώδικα 25: Παραδείγματα χρήσης της watch function

Υπάρχουν και μερικές επιπλέον επιλογές για που μπορούν να φανούν χρήσιμες κάτω από συγκεκριμένες περιστάσεις όπως τα deep και immediate options. Το πρώτο χρησιμοποιείται σε κάποιες περιπτώσεις που ο χρήστης θέλει να παρακολουθήσει αλλαγές σε ένα αντικείμενο που περιέχει άλλα αντικείμενα ή πίνακες και θέλει να παρακολουθεί αλλαγές σε οποιοδήποτε από αυτά τα εσωτερικά στοιχεία ή όταν πίνακες που περιέχουν αντικείμενα ή άλλους πίνακες και θέλει να ανιχνεύει αλλαγές στα στοιχεία τους. Με την χρήση του immediate ο χρήστης αναγκάζει την watcher να εκτελεστεί αμέσως μόλις δηλωθεί, χωρίς να περιμένει κάποια αλλαγή στην παρακολουθούμενη τιμή. Στην ουσία δίνει την αίσθηση της λειτουργίας του “mounted” lifecycle hook [15].

2.2.6.10 Router

Ο Router είναι η επίσημη λύση για την πλοήγηση και τη δρομολόγηση (routing) στις εφαρμογές Vue. Παρέχει έναν τρόπο για να δημιουργείς και να διαχειρίζεσαι διαδρομές (routes) στην εφαρμογή σου, επιτρέποντας στους χρήστες να μετακινούνται μεταξύ διαφορετικών σελίδων ή τμημάτων μιας σελίδας χωρίς να γίνεται επαναφόρτωση του προγράμματος περιήγησης αλλά μόνο του αντίστοιχου component.

Η κάθε διαδρομή έχει ένα path και ένα component που θα πρέπει να φορτωθεί όταν η διαδρομή αντιστοιχεί σε αυτό. Οι διαδρομές μπορούν να αποτελούνται και από δυναμικά τμήματα, όπως “user/:id”, όπου το :id μπορεί να είναι οποιαδήποτε τιμή. Ένα ακόμη προνόμιο είναι οι παράμετροι (route params) που επιτρέπουν την μεταφορά δεδομένων μέσω του URL στις σελίδες της εφαρμογής. Υπερπολύτιμο χαρακτηριστικό του Router είναι οι φύλακες διαδρομής (navigation guards). Προσφέρουν hooks που μπορούν να εκτελεστούν πριν ή μετά την πλοήγηση σε μια διαδρομή, επιτρέποντας τον έλεγχο πρόσβασης και την εκτέλεση λογικής πριν την αλλαγή σελίδας. Η πιο συνηθισμένη χρήση τους είναι για την αποτροπή εισόδου σε σελίδα στην οποία δεν υπάρχει άδεια πρόσβασης στον χρήστη. Βελτιωμένη εμπειρία χρήσης μιας εφαρμογής με router, προσφέρει η λειτουργεία lazy loading όπου τα στοιχεία φορτώνονται μόνο όταν απαιτούνται, βοηθώντας στη μείωση του αρχικού μεγέθους του πακέτου και στη βελτίωση της απόδοσης. Δεν θα μπορούσαν να μην ληφθούν υπόψη οι nested routes ως χαρακτηριστικό κλειδί του router καθώς επιτρέπουν τη δημιουργία πολυεπίπεδων διαδρομών, επιτρέποντας την εμφάνιση δευτερευόντων στοιχείων μέσα σε κύρια components, βοηθώντας έτσι στην οργάνωση της εφαρμογής και τη δυναμική διαχείριση του περιεχομένου. Τέλος, διαθέσιμο είναι και το history mode, όπου επιτρέπει τη χρήση καθαρών URLs χωρίς το hash σύμβολο, καθιστώντας τις διευθύνσεις URL πιο φιλικές προς τον χρήστη [16].

2.2.6.11 Pinia

Η Pinia πρόκειται για μια βιβλιοθήκη της Vue, η οποία επικεντρώνεται στη διαχείριση της κατάστασης της εφαρμογής (state management), παρέχοντας έναν τρόπο για την οργάνωση και τη διαχείριση των δεδομένων σε διάφορα μέρη της εφαρμογής. Αυτό το πετυχαίνει δημιουργώντας stores στα οποία έχει την δυνατότητα να αποθηκεύει προσωρινά την πληροφορία, να την επεξεργάζεται, καθώς και να την μεταφέρει οπουδήποτε χρειαστεί μέσα στην εφαρμογή. Δίνει έμφαση στην απλότητα και την ευκολία χρήσης πράγμα που την κάνει να διαφέρει από τον προκάτοχο της, την VueX που χρησιμοποιούταν στις παλαιότερες εκδόσεις της Vue. Παρόλο που πρόκειται για μία καινούργια βιβλιοθήκη η οποία ξεκίνησε να σχεδιάζεται με γνώμονα την έκδοση 3 της Vue και το composition api δεν περιορίζεται από αυτό και είναι συμβατή και με την έκδοση 2 αλλά και το options api. Φυσικά, δεδομένο θα μπορούσε να θεωρηθεί ότι υποστηρίζει την TypeScript καθώς φτιάχτηκε με γνώμονα αυτή.

Όσον αφορά την δομή ενός Pinia store, όλα ξεκινάνε από την συνάρτηση defineStore με την βοήθεια της οποίας ορίζεται το store. Σαν πρώτη παράμετρο δέχεται το όνομα του store το οποίο είναι μοναδικό και η δεύτερη παράμετρος είναι το λειτουργικό κομμάτι. Σε αυτή θα βρούμε το state στο οποίο βρίσκονται τα δεδομένα, τις getters που αποτελούν συναρτήσεις επιστροφής οι οποίες λειτουργούν σαν computed properties που επιστρέφουν τις τιμές των δεδομένων του state και τέλος τα actions που πρόκειται για τις μεθόδους που εκτελούν οποιαδήποτε λογική και έχουν την δυνατότητα να τροποποιήσουν τα δεδομένα του state. Οι συχνότερη χρήση των actions είναι η λήψη δεδομένων από ένα api και η ενημέρωση του state με τα δεδομένα που λαμβάνονται. Σε αυτό συμβάλει καθοριστικά η υποστήριξη των ασύγχρονων μεθόδων στην pinia [17].

```
import { defineStore } from 'pinia'

export const useCounterStore = defineStore('counter', {
  state: () => ({
    count: 0 as number,
  }),
  getters: {
    getCount: (state) => state.count
  },
  actions: {
    increment() {
      this.count++
    }
  }
})
```

Απόσπασμα κώδικα 26: Παράδειγμα δομής Pinia Store

2.2.7 Quasar Framework

Η Quasar είναι ένα framework ανοιχτού κώδικα που βασίζεται στην Vue και έχει σχεδιαστεί για να βοηθά τους προγραμματιστές να δημιουργούν ανταποκρινόμενους ιστότοπους, εφαρμογές για κινητά και εφαρμογές Electron από μια ενιαία βάση κώδικα. Αυτή η προσέγγιση όχι μόνο μειώνει την πολυπλοκότητα αλλά και ενισχύει την αποτελεσματικότητα της ανάπτυξης. Λόγω της ενσωμάτωσης με την Vue έχει υιοθετήσει την αρχιτεκτονική της που βασίζεται σε components και έχει ένα οικοσύστημα που υποστηρίζει την εγκατάσταση πρόσθετων (plugins). Έτσι ένας χρήστης εξοικειωμένος με την Vue μπορεί να μεταβεί χωρίς δυσκολία στην χρήση της Quasar. Διαθέτει μια πλούσια βιβλιοθήκη στοιχείων περιβάλλοντος εργασίας χρήστη, συμπεριλαμβανομένων πινάκων δεδομένων, εισόδων φορμών και παραθύρων διαλόγου, όλα βελτιστοποιημένα για την μέγιστη απόδοση και χρηστικότητα. Αυτή η ολοκληρωμένη βιβλιοθήκη από components οδηγεί στην ταχεία ανάπτυξη και εξασφαλίζει συνέπεια μεταξύ των διαφόρων τμημάτων μιας εφαρμογής. Επιπλέον, προσφέρει ισχυρές δυνατότητες θεματοποίησης, επιτρέποντας την εύκολη προσαρμογή της εμφάνισης της εφαρμογής, συμπεριλαμβανομένης της υποστήριξης τόσο για φωτεινά όσο και για σκοτεινά θέματα. Ένα ακόμη χαρακτηριστικό είναι η ενσωματωμένη υποστήριξη για διεθνοποίηση (i18n), απλοποιώντας τη διαδικασία δημιουργίας εφαρμογών που απευθύνονται σε ένα παγκόσμιο κοινό [18].

2.2.8 Tailwind

Η Tailwind CSS είναι ένα framework CSS που έχει σχεδιαστεί για να επιτρέπει στους προγραμματιστές να δημιουργούν γρήγορα σύγχρονες διεπαφές ιστού χωρίς την ανάγκη χρήσης κώδικα CSS παραδοσιακής φύσης. Παρέχει ένα σύνολο προσχεδιασμένων κλάσεων που καλύπτουν ένα ευρύ φάσμα ιδιοτήτων CSS, επιτρέποντας στους προγραμματιστές να εφαρμόζουν στυλ απευθείας στα HTML στοιχεία.

```
<div style="font-weight: bold; display: flex; background: white;">
   This is a bold style using CSS
 </div>

 <div class="font-bold flex bg-white">
   This is a bold style using tailwind
 </div>
```

Απόσπασμα κώδικα 27: Παράδειγμα χρήσης κλάσης tailwind

Παρόλο που δεν παρέχει έτοιμα components από το κουτί όπως η Quasar, προτρέπει τον χρήστη στη δημιουργία επαναχρησιμοποιήσιμων custom components, κάτι που καθιστά εύκολο με τη χρήση συνδυασμών των κλάσεων της, προσφέροντας ένα άρτιο αποτέλεσμα. Ένα από τα δυνατότερα σημεία της είναι η ευελιξία που προσφέρει όσον αφορά τις ανταποκρινόμενες σελίδες. Παρέχει κλάσεις που ανταποκρίνονται στις διαστάσεις της οθόνης, επιτρέποντας στον χρήστη να προσαρμόζει τα στοιχεία ώστε να ανταποκρίνονται σε οποιαδήποτε συσκευή προβολής. Επίσης, διαθέτει ενσωματωμένη υποστήριξη για σκοτεινή λειτουργία, διευκολύνοντας τη δημιουργία διεπαφών που προσαρμόζονται στις προτιμήσεις των χρηστών. Δεν θα μπορούσαμε να παραλείψουμε το γεγονός ότι, μέσω του αρχείου ρυθμίσεων "tailwind.config.js", μας επιτρέπει να διαχειριστούμε, να επεκτείνουμε ή να παρακάμψουμε τις προεπιλεγμένες ρυθμίσεις της. Τέλος, με τη λειτουργία Just-In-Time, προσφέρει μια εξαιρετικά αποτελεσματική λύση για τη δημιουργία εφαρμογών. Αυτή η λειτουργία επιτρέπει στον προγραμματιστή να χρησιμοποιεί τις utility κλάσεις της Tailwind χωρίς το βάρος της παραγωγής ενός μεγάλου αρχείου CSS. Αντίθετα, αναλαμβάνει δυναμικά τη δημιουργία των απαραίτητων CSS κανόνων κατά τη διάρκεια της διαδικασίας ανάπτυξης, εξοικονομώντας χρόνο και βελτιστοποιώντας την απόδοση της εφαρμογής [19].

2.3 Server Side

2.3.1 PHP

Η PHP (Hypertext Preprocessor) είναι μια ευρέως χρησιμοποιούμενη γλώσσα προγραμματισμού για την ανάπτυξη δυναμικών ιστοσελίδων και εφαρμογών διαδικτύου. Σχεδιάστηκε αρχικά από τον Rasmus Lerdorf το 1993 και ξεκίνησε ως ένα σύνολο σεναρίων Common Gateway Interface (CGI) στη C για τη διαχείριση της προσωπικής του αρχικής σελίδας. Αρχικά ονομάστηκε "Personal Home Page / Forms Interpreter" (PHP / FI), κυκλοφόρησε στο κοινό το 1995, επιτρέποντας στους προγραμματιστές να το χρησιμοποιήσουν και να το βελτιώσουν. Αυτή η πρώιμη έκδοση παρείχε βασικές λειτουργίες όπως ο χειρισμός φορμών και η συνδεσιμότητα βάσεων δεδομένων, οι οποίες ήταν ζωτικής σημασίας για την ανάπτυξη ιστού εκείνη την εποχή.

Το 1997, η PHP υπέστη μια σημαντική μεταμόρφωση όταν ο Zeev Suraski και ο Andi Gutmans ξαναέγραψαν τον αναλυτή (parser), οδηγώντας στη δημιουργία της PHP 3. Αυτή η έκδοση, που κυκλοφόρησε επίσημα τον Ιούνιο του 1998, εισήγαγε μια πιο συνεπή σύνταξη και ένα εκτεταμένο σύνολο λειτουργιών και βιβλιοθηκών, καθιστώντας το ένα ισχυρό εργαλείο για προγραμματιστές ιστού. Η PHP 3 σηματοδότησε τη μετάβαση από ένα απλό εργαλείο δέσμης ενεργειών σε μια ισχυρή γλώσσα προγραμματισμού.

Η PHP 4, που κυκλοφόρησε τον Μάιο του 2000, τροφοδοτήθηκε από το Zend Engine 1.0, το οποίο βελτίωσε σημαντικά την απόδοση και την αξιοπιστία του. Αυτή η έκδοση εισήγαγε χαρακτηριστικά όπως ο χειρισμός συνεδριών (session handling) και η προσωρινή αποθήκευση εξόδου (output buffering), επιτρέποντας στους προγραμματιστές να δημιουργήσουν πιο σύνθετες εφαρμογές ιστού. Η PHP 4 βελτίωσε επίσης την υποστήριξη για αντικειμενοστραφή προγραμματισμό, θέτοντας τις βάσεις για μελλοντικές εξελίξεις.

Η κυκλοφορία της PHP 5 τον Ιούλιο του 2004 ήταν ένα άλλο σημαντικό ορόσημο. Με την εισαγωγή του Zend Engine II, η PHP 5 προσέφερε ολοκληρωμένη υποστήριξη για αντικειμενοστραφή προγραμματισμό, καθιστώντας την μια πιο ανταγωνιστική επιλογή έναντι γλωσσών όπως η Java και η C++. Αυτή η έκδοση εισήγαγε επίσης την επέκταση PHP Data Objects (PDO), η οποία παρείχε μια συνεπή διεπαφή για πρόσβαση στη βάση δεδομένων, βελτιώνοντας τόσο την απόδοση όσο και την

ασφάλεια. Η εκτεταμένη διάρκεια ζωής της PHP 5, λαμβάνοντας ενημερώσεις μέχρι το 2018, απέδειξε την ανθεκτικότητα και την προσαρμοστικότητά της [20].

Όσον αφορά την PHP 6, οι προσπάθειες για την ανάπτυξη της ξεκίνησαν το 2005, εστιάζοντας στην ενσωμάτωση εγγενούς υποστήριξης Unicode. Ωστόσο, το έργο αντιμετώπισε σημαντικές προκλήσεις και τελικά εγκαταλείφθηκε. Πολλά από τα χαρακτηριστικά που αναπτύχθηκαν κατά τη διάρκεια αυτής της περιόδου μεταφέρθηκαν στην PHP 5.3 και 5.4, διασφαλίζοντας ότι η εργασία δεν σπαταλήθηκε εντελώς [21].

Η PHP 7, που κυκλοφόρησε τον Δεκέμβριο του 2015, σηματοδότησε μια επανάσταση στην απόδοση της PHP. Η νέα μηχανή phpng (PHP Next Generation), που αναπτύχθηκε από τους Dmitry Stogov, Xinchen Hui και Nikita Popov, βελτίωσε σημαντικά την απόδοση και μείωσε την κατανάλωση μνήμης. Η PHP 7 εισήγαγε δηλώσεις κλιμακωτού τύπου, δηλώσεις τύπου επιστροφής και ένα νέο μοντέλο χειρισμού σφαλμάτων χρησιμοποιώντας αντικειμενοστρεφείς εξαιρέσεις, καθιστώντας την μια σύγχρονη και αποτελεσματική γλώσσα προγραμματισμού.

Η PHP 8, που κυκλοφόρησε τον Νοέμβριο του 2020, έφερε περαιτέρω εξελίξεις, συμπεριλαμβανομένης της λειτουργίας Just-In-Time (JIT), των τύπων ένωσης και των χαρακτηριστικών. Η μεταγλώττιση JIT στόχευε στη βελτίωση της απόδοσης με τη μεταγλώττιση κώδικα κατά το χρόνο εκτέλεσης, ενώ οι τύποι ένωσης επέτρεψαν στις μεταβλητές να διατηρούν πολλαπλούς τύπους δεδομένων. Τα χαρακτηριστικά παρείχαν έναν τρόπο προσθήκης μεταδεδομένων στον κώδικα PHP, παρόμοιο με τους σχολιασμούς σε άλλες γλώσσες. Η PHP 8 εισήγαγε επίσης επώνυμα ορίσματα και εκφράσεις αντιστοίχισης, ενισχύοντας περαιτέρω τις δυνατότητές της και τη φιλικότητα προς τους προγραμματιστές.

Καθ 'όλη τη διάρκεια της ιστορίας της, η PHP έχει εξελιχθεί από ένα απλό σύνολο εργαλείων για προσωπική διαχείριση ιστότοπων σε μία από τις πιο ευρέως χρησιμοποιούμενες server-side scripting γλώσσες. Η συνεχής ανάπτυξή της και η προσαρμογή της στις σύγχρονες ανάγκες ανάπτυξης ιστοσελίδων έχουν εξασφαλίσει τη συνάφεια και τη δημοτικότητά του. Το ταξίδι της PHP αντικατοπτρίζει τις συλλογικές προσπάθειες μιας παγκόσμιας κοινότητας προγραμματιστών, συμβάλλοντας σε ένα έργο ανοιχτού κώδικα που τροφοδοτεί ένα σημαντικό μέρος του ιστού σήμερα.

2.3.2 Laravel

Καθώς η PHP μεγάλωνε, οι προγραμματιστές άρχισαν να αναζητούν τρόπους για να απλοποιήσουν τις επαναλαμβανόμενες εργασίες, να βελτιώσουν τη συντηρησιμότητα του κώδικα και να εφαρμόσουν πιο αποτελεσματικά μοτίβα σχεδιασμού. Αυτή η ανάγκη οδήγησε στην ανάπτυξη διαφόρων frameworks, με την Laravel να αναδύεται ως ένα από τα πιο δημοφιλή με αξιοσημείωτη επιρροή. Η Laravel δημιουργήθηκε από τον Taylor Otwell το 2011, με την αρχική έκδοση να στοχεύει στην παροχή μιας εναλλακτικής λύσης στο framework CodeIgniter, το οποίο δεν διέθετε ορισμένα χαρακτηριστικά που χρειάζονταν οι σύγχρονοι προγραμματιστές ιστού. Σχεδιάστηκε για να διευκολύνει κοινές εργασίες όπως ο έλεγχος ταυτότητας, η δρομολόγηση, οι συνεδρίες και η προσωρινή αποθήκευση, παρέχοντας παράλληλα μια καθαρή και κομψή σύνταξη. Το όραμα του Otwell ήταν να δημιουργήσει ένα εργαλείο που θα έκανε την ανάπτυξη της PHP πιο ευχάριστη και αποτελεσματική [22].

Η Laravel επιλέχθηκε για την υλοποίηση του back-end του δεύτερης έκδοσης του Alumni, το οποίο όπως έχει ήδη αναφερθεί, αναπτύχθηκε ως μέρος της διπλωματικής εργασίας [1] λόγω των πολλών πλεονεκτημάτων που προσφέρει. Ένα από τα κύρια πλεονεκτήματα της Laravel είναι ότι παρέχει μια φιλική και προσιτή εμπειρία ανάπτυξης για αρχάριους προγραμματιστές, καθιστώντας εύκολη την κατανόηση και την εκμάθηση του πλαισίου χάρη στην απλή και καθαρή σύνταξη της. Ταυτόχρονα, η Laravel είναι αρκετά ισχυρή και ευέλικτη ώστε να ικανοποιήσει τις ανάγκες των έμπειρων

προγραμματιστών, προσφέροντας προηγμένα χαρακτηριστικά και εργαλεία που επιτρέπουν την ανάπτυξη σύνθετων και απαιτητικών εφαρμογών. Ακολουθώντας την αρχιτεκτονική Model-View-Controller (MVC), η Laravel οργανώνει τον κώδικα λογικά, βελτιώνοντας τη συντηρησιμότητα και την επεκτασιμότητα. Το Eloquent ORM απλοποιεί τις αλληλεπιδράσεις της βάσης δεδομένων με μια διαισθητική υλοποίηση ActiveRecord, ενώ το Blade, η μηχανή templating του Laravel, επιτρέπει την ενσωμάτωση απλού κώδικα PHP μέσα στα πρότυπα, εξασφαλίζοντας γρήγορη απόδοση και παρέχοντας χαρακτηριστικά όπως η κληρονομικότητα προτύπων. Το Artisan, η διεπαφή γραμμής εντολών (command-line) του Laravel, αυτοματοποιεί επαναλαμβανόμενες εργασίες, ενισχύοντας την παραγωγικότητα με εντολές για μετεγκαταστάσεις βάσεων δεδομένων και πολλά άλλα. Τέλος, συμπεριλαμβάνει ολοκληρωμένες δυνατότητες ελέγχου ταυτότητας και εξουσιοδότησης, όπου διευκολύνουν την ταυτοποίηση του χρήστη, όπως τα middlewares που λειτουργούν σαν φίλτρα για τα HTTP αιτήματα που εισέρχονται στην εφαρμογή.

2.4 Επικοινωνία Client-Server

Στη σύγχρονη ανάπτυξη Web εφαρμογών, η επικοινωνία μεταξύ του front-end και του back-end αποτελεί έναν κρίσιμο τομέα για τη σωστή λειτουργία και αλληλεπίδραση των συστημάτων. Στην παρούσα πτυχιακή η αλληλεπίδραση της εφαρμογής με το API γίνεται με την χρήση της βιβλιοθήκης Axios αφού αποτελεί μια από τις πλέον χρησιμοποιούμενες λύσεις ιδιαίτερα σε εφαρμογές που βασίζονται στη JavaScript και τα frameworks της.

Η Axios είναι μια εξαιρετικά ευέλικτη βιβλιοθήκη που υποστηρίζει όλα τα βασικά αιτήματα HTTP, όπως GET, POST, PUT, DELETE, και άλλα, καθιστώντας την ιδανική για διάφορες ανάγκες στην ανάπτυξη εφαρμογών. Η υποστήριξη για τα βασικά αυτά αιτήματα επιτρέπει στους προγραμματιστές να εκτελούν μια ευρεία γκάμα ενεργειών, από την ανάκτηση δεδομένων από έναν εξυπηρετητή, μέχρι την αποστολή νέων δεδομένων ή την ενημέρωση και διαγραφή υπαρχόντων πόρων [23].

Στο παράδειγμα που ακολουθεί (Απόσπασμα κώδικα 28) γίνεται η χρήση του GET αιτήματος στο URL “https://api.example.com/data” για να ζητήσει δεδομένα από αυτήν τη διεύθυνση. Αν η αίτηση είναι επιτυχής, η υπόσχεση (promise) που επιστρέφεται από την “axios.get” επιλύεται και εκτελείται η συνάρτηση then. Μέσα στη συνάρτηση then, η απάντηση της αίτησης περνάει ως όρισμα (response). Το response.data περιέχει τα δεδομένα που επιστράφηκαν από τον διακομιστή. Αυτά τα δεδομένα καταγράφονται στην κονσόλα με την εντολή console.log(response.data). Αν η αίτηση αποτύχει, η υπόσχεση απορρίπτεται και εκτελείται η συνάρτηση catch. Μέσα στη συνάρτηση catch, το σφάλμα που προέκυψε περνάει ως όρισμα (error) και στην συνέχεια καταγράφεται στην κονσόλα με την εντολή console.error('There was an error!', error).

```
axios.get('https://api.example.com/data')
  .then(response => {
    console.log(response.data);
  })
  .catch(error => {
    console.error('There was an error!', error);
  });
```

Απόσπασμα κώδικα 28: Παράδειγμα χρήσης axios request με get method

Ένα από τα κύρια πλεονεκτήματα της Axios είναι η αρχιτεκτονική της που βασίζεται σε Promises. Τα Promises διευκολύνουν τη διαχείριση των ασύγχρονων λειτουργιών, παρέχοντας μια πιο καθαρή και κατανοητή σύνταξη κώδικα σε σύγκριση με τις παραδοσιακές μεθόδους όπως τα callbacks. Αυτό σημαίνει ότι οι προγραμματιστές μπορούν να γράφουν κώδικα που είναι πιο ευανάγνωστος και εύκολος στη συντήρηση. Αυτός ο τρόπος προτιμήθηκε και στην υλοποίηση της πλατφόρμας του Alumni. Πιο συγκεκριμένα, με την χρήση της δομής "try…catch" και των λέξεων "async" και "await", δημιουργήθηκαν ασύγχρονες μέθοδοι για την επικοινωνία με το Api.

```
try {
  const response = await axios.get('https://api.example.com/data');
  console.log(response.data);
} catch (error) {
  console.error('There was an error!', error);
}
```

Απόσπασμα κώδικα 29: Παράδειγμα ασύγχρονης χρήσης axios με try...catch

Η Axios παρέχει απρόσκοπτη ενσωμάτωση με το JSON, την πλέον διαδεδομένη μορφή ανταλλαγής δεδομένων. Από προεπιλογή, το Axios μετατρέπει αυτόματα αντικείμενα JavaScript σε JSON κατά την αποστολή αιτημάτων και ορίζει την κεφαλίδα Content-Type σε application/json. Αυτό καθιστά απλή την αποστολή δεδομένων καθώς και την λήψη αυτών αφού η Axios αναλύει αυτόματα τις αποκρίσεις JSON, επιτρέποντας στους προγραμματιστές να εργάζονται απευθείας με αντικείμενα JavaScript.

Για παράδειγμα στο παρακάτω απόσπασμα κώδικα (Απόσπασμα κώδικα 30), την παράμετρος στο POST αίτημα αποτελεί ένα object το οποίο μετατρέπεται από την Axios εσωτερικά σε έγκυρη μορφή JSON πριν σταλεί.

```
try {
  const response = await axios.post('https://api.example.com/data', {
    firstName: 'John',
    lastName: 'Doe'
  });
  console.log(response.data);
} catch (error) {
  console.error('There was an error!', error);
}
```

```
// Το object θα αναλυθεί και θα σταλεί με την μορφή:
{
  "firstName": "John",
  "lastName": "Doe"
}
```

Απόσπασμα κώδικα 30: Παράδειγμα post request με αποστολή δεδομένων σε μορφή object

Τέλος, προσφέρει ένα εργαλείο κλειδί που δίνει τη δυνατότητα επεξεργασίας των αιτημάτων και των απαντήσεων. Δεν είναι άλλο από τους interceptors, που μέσω αυτών είναι εφικτή η προσθήκη κοινής λογικής σε όλα τα αιτήματα ή τις απαντήσεις, όπως η προσθήκη tokens για την αυθεντικοποίηση, η διαχείριση των σφαλμάτων των αιτημάτων ή η επεξεργασία των δεδομένων της απάντησης. Οι interceptors επιτρέπουν τη δυναμική προσαρμογή της συμπεριφοράς των αιτημάτων και των απαντήσεων, προσφέροντας μεγαλύτερο έλεγχο και ευελιξία στην ανάπτυξη εφαρμογών.

```
axios.interceptors.request.use(config => {
  // Do something before request is sent
  return config;
}, error => {
  // Do something with request error
  return Promise.reject(error);
});

axios.interceptors.response.use(response => {
  // Do something with response data
  return response;
}, error => {
  // Do something with response error
  return Promise.reject(error);
});
```

Απόσπασμα κώδικα 31: Παράδειγμα χρήσης axios interceptors

Κεφάλαιο 3ο: Περιγραφή Βάσης Δεδομένων

3.1 Εισαγωγή

Η βάση δεδομένων αποτελεί έναν θεμελιώδη παράγοντα για την εφαρμογή. Ο βασικός της σκοπός είναι να αποθηκεύει και να διαχειρίζεται όλα τα δεδομένα των χρηστών, εξασφαλίζοντας την ασφάλεια, την ακεραιότητα και τη διαθεσιμότητα των πληροφοριών. Ο σχεδιασμός της βάσης της δεύτερης έκδοσης του Alumni δεν αποτελεί αντικείμενο υλοποίησης της παρούσας πτυχιακής εργασίας. Αντίθετα, όπως έχει γίνει ήδη αναφορά, είναι μέρος της μεταπτυχιακής διπλωματικής εργασίας του Ταουκτσή Βασίλειου [1] στην οποία περιγράφεται λεπτομερώς η αρχιτεκτονική της. Με γνώμονα αυτή, θα γίνει μια σύντομη αναφορά στις οντότητες της βάσης αλλά και στους βασικούς πίνακες ώστε ο αναγνώστης να βρίσκεται σε θέση να καταλάβει τις λειτουργίες της εφαρμογής.

3.2 Σχεσιακό μοντέλο

Το μοντέλο της βάσης δεδομένων αποτελείται από οκτώ βασικές και πέντε βοηθητικές οντότητες. Βασικές οντότητες αποτελούν οι:

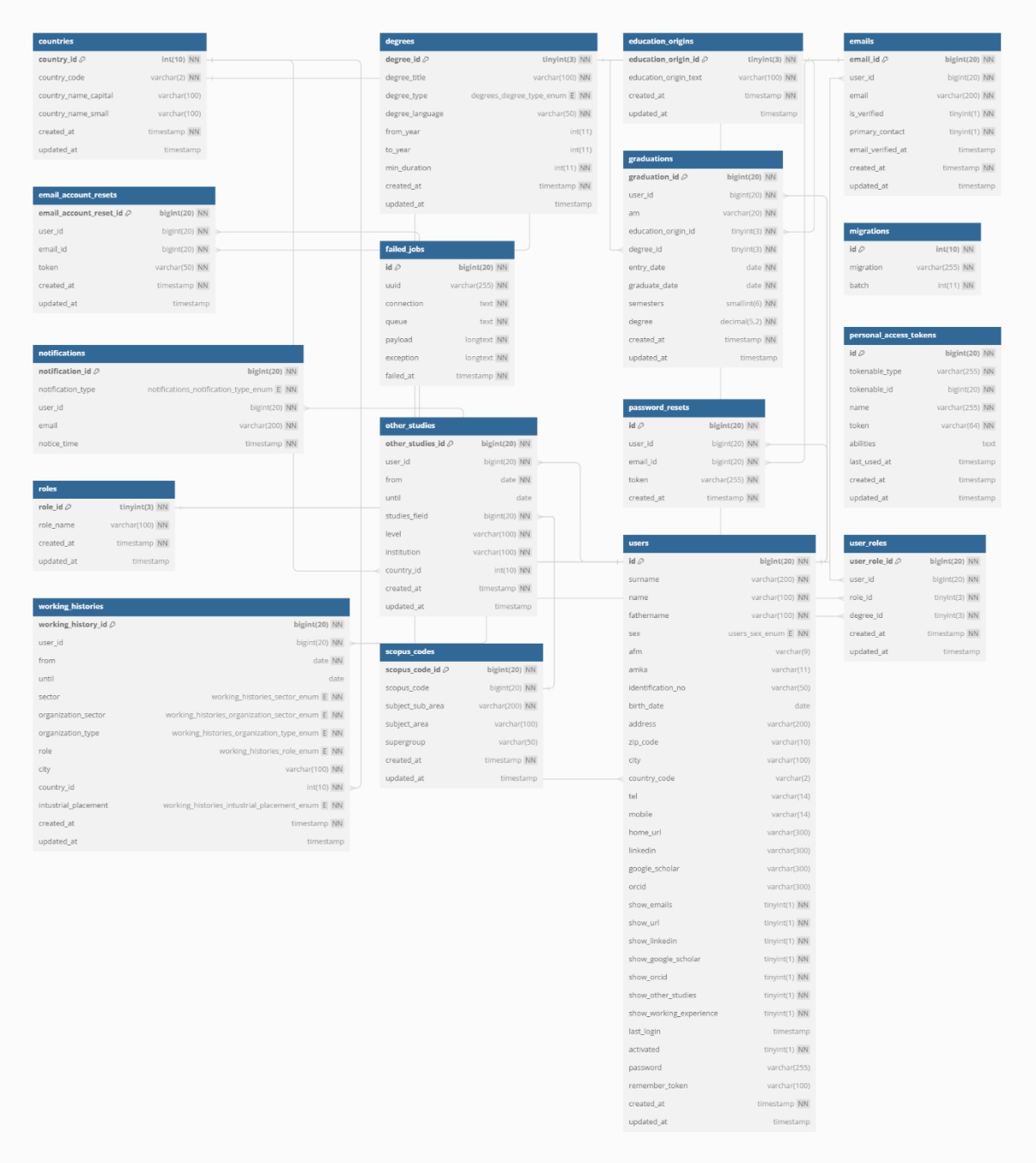
- **users**: Η οντότητα που αφορά τα προσωπικά δεδομένα ενός χρήστη.
- **degrees**: Αφορά τα προγράμματα σπουδών που προσφέρει το τμήμα ΜΠΗΣ.
- **roles**: Καθορίζει το ρόλο ενός χρήστη, με αποδεκτές τιμές Διαχειριστής, Γραμματεία, Μέλος ΟΜΕΑ και Απόφοιτος.
- **graduations**: Αναφέρεται στις αποφοιτήσεις από το τμήμα ΜΠΗΣ.
- **other_studies**: Αφορά τις σπουδές σε άλλα τμήματα.
- **working_history**: Αφορά την επαγγελματική εμπειρία.
- **emails**: Περιλαμβάνει τους λογαριασμούς αλληλογραφίας των χρηστών. Αυτοί μπορούν να είναι απεριόριστοι.
- **notifications**: Αφορά τις ειδοποιήσεις που λαμβάνουν οι χρήστες από την εφαρμογή.

Βοηθητικές θεωρούνται, οι οντότητες που χρησιμοποιούνται για την συμπλήρωση φορμών ή για την καταχώριση προσωρινών τιμών και είναι οι:

- **education_origins**: Αφορά τους τίτλους σπουδών που μπορούν να χρησιμοποιηθούν για την εγγραφή στο τμήμα ΜΠΗΣ.
- **password_resets**: Χρησιμοποιείται για την καταχώριση προσωρινών κωδικών για την επιβεβαίωση της πρόθεσης αλλαγής κωδικού από τον χρήστη.
- **email_account_reset**: Καταχωρεί προσωρινούς κωδικούς για την επιβεβαίωση της πρόθεσης αλλαγής λογαριασμού αλληλογραφίας από τον χρήστη.
- **countries**: Περιλαμβάνει τον κατάλογο των χωρών με τις επίσημες ονομασίες τους.
- **scopus_codes**: Αναφέρεται στις θεματικές επιστημονικές περιοχές.

3.3 Διάγραμμα ER

Σημαντικό για την κατανόηση των σχέσεων μεταξύ των οντοτήτων αποτελεί η εικονική αναπαράσταση αυτών η οποία επιτυγχάνεται μέσω του διαγράμματος ER που ακολουθεί το οποίο δημιουργήθηκε με την χρήση του online εργαλείου dbdiagram [24].



Εικόνα 3: Διάγραμμα ER σχεσιακού μοντέλου

3.4 Βασικοί πίνακες βάσης δεδομένων

Οι ανάλυση των παρακάτω πινάκων θεωρείται απαραίτητη ώστε ο αναγνώστης να έχει την δυνατότητα τα κατανοήσει ορισμένες λειτουργίες της πλατφόρμας. Μερικές από αυτές αποτελούν ο τρόπος με τον οποίο υλοποιήθηκε η σύνδεση του χρήστη, η μεταφορά του χρήστη στο αντίστοιχο UI, η επιλογή των δεδομένων που θα λάβει ο χρήστης κατά την είσοδο του στην εφαρμογή και άλλα.

3.4.1 Πίνακας users

Ο πίνακας "users" είναι ο κεντρικός πίνακας της βάσης δεδομένων, στον οποίο αναφέρονται όλα τα ξένα κλειδιά. Αποθηκεύει τα προσωπικά δεδομένα κάθε χρήστη, όπως το όνομα, το επώνυμο, το πατρώνυμο, το φύλο, τον ΑΦΜ (afm), τον ΑΜΚΑ (amka), τον αριθμό ταυτότητας (identification_no), την ημερομηνία γέννησης, τη διεύθυνση και στοιχεία επικοινωνίας. Περιλαμβάνει επίσης πεδία για προφίλ κοινωνικών δικτύων και επιστημονικών βάσεων δεδομένων, όπως το LinkedIn, το Google Scholar και το ORCID. Επιπλέον, έχει πεδία για την ενεργοποίηση λογαριασμού, την τελευταία σύνδεση και άλλες ρυθμίσεις ορατότητας των δεδομένων. Το κύριο κλειδί είναι το πεδίο "id".

3.4.2 Πίνακας degrees

Ο πίνακας "degrees" αποθηκεύει πληροφορίες σχετικά με τα πτυχία που μπορεί να έχει κάποιος χρήστης. Περιλαμβάνει τον τίτλο του πτυχίου, τον τύπο του (π.χ., προπτυχιακό, μεταπτυχιακό κλπ), τη γλώσσα του πτυχίου, τα έτη έναρξης και λήξης, και την ελάχιστη διάρκεια σπουδών. Το κύριο κλειδί είναι το πεδίο "degree_id". Αυτός ο πίνακας συνδέεται με τον πίνακα "graduations" και τον πίνακα "user_roles" μέσω του πεδίου "degree_id".

3.4.3 Πίνακας graduations

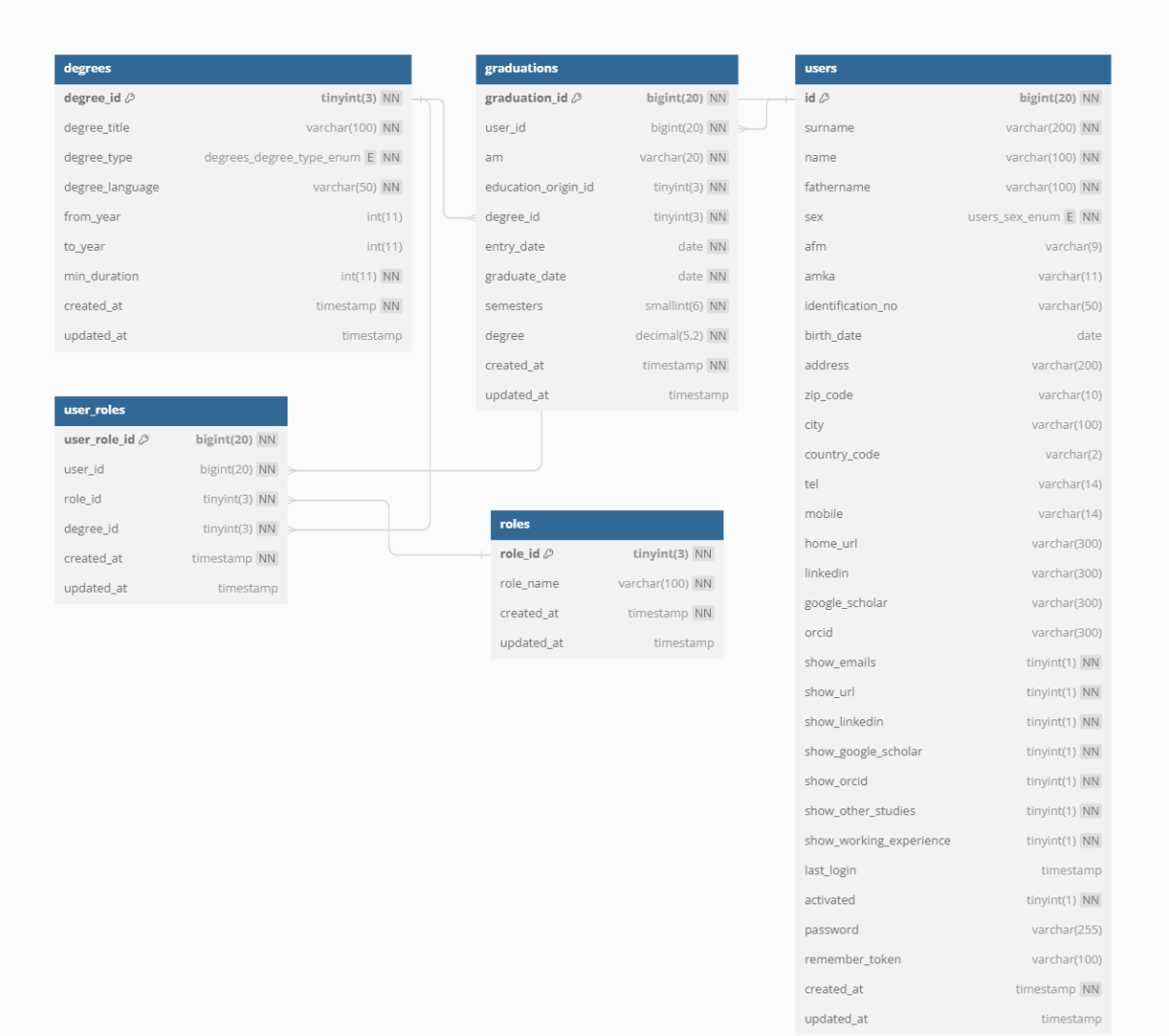
Ο πίνακας "graduations" καταγράφει τις αποφοιτήσεις των χρηστών. Περιλαμβάνει πληροφορίες όπως το αναγνωριστικό του χρήστη (user_id), τον αριθμό μητρώου (am), το εκπαιδευτικό ίδρυμα προέλευσης, το πτυχίο (degree_id), την ημερομηνία εισόδου και αποφοίτησης, τον αριθμό των εξαμήνων και τον βαθμό πτυχίου. Το κύριο κλειδί είναι το πεδίο "graduation_id". Συσχετίζεται με τον πίνακα "users" και τον πίνακα "degrees" μέσω των πεδίων "user_id" και "degree_id" αντίστοιχα.

3.4.4 Πίνακας roles

Ο πίνακας "roles" περιέχει πληροφορίες για τους διάφορους ρόλους που μπορεί να έχει ένας χρήστης (π.χ., Διαχειριστής, Γραμματεία κλπ). Κάθε ρόλος έχει ένα μοναδικό αναγνωριστικό και ένα όνομα. Το κύριο κλειδί είναι το πεδίο "role_id". Αυτός ο πίνακας συνδέεται με τον πίνακα "user_roles" μέσω του πεδίου "role_id".

3.4.5 Πίνακας user_roles

Ο πίνακας "user_roles" συνδέει τους χρήστες με τους ρόλους και τα πτυχία τους. Περιλαμβάνει το αναγνωριστικό του χρήστη (user_id), το αναγνωριστικό του ρόλου (role_id) και το αναγνωριστικό του πτυχίου (degree_id). Το κύριο κλειδί είναι το πεδίο "user_role_id". Συσχετίζεται με τους πίνακες "users", "roles" και "degrees" μέσω των πεδίων "user_id", "role_id" και "degree_id" αντίστοιχα.



Εικόνα 4: Διάγραμμα ER των βασικών πινάκων της βάσης

Η επιλογή ανάλυσης των παραπάνω έγινε βάσει κάποιων κριτηρίων. Για παράδειγμα ο πίνακας users επιλέχθηκε γιατί είναι ο κεντρικός πίνακας που συνδέεται σχεδόν με όλους τους υπόλοιπους και διαθέτει την περισσότερη πληροφορία. Ομοίως, ο πίνακας graduations είναι ο πίνακας που περιέχει την δεύτερη σε μέγεθος πληροφορία και συνδέεται με άλλους βασικούς πίνακες. Ο πίνακας degrees επιλέχθηκε για να γίνει η αναφορά πως οι πληροφορίες που λαμβάνει ο κάθε χρήστης αφορούν τον κύκλο σπουδών που ακολούθησε για την αποφοίτηση του. Και τέλος ο πίνακας user_roles ο οποίος μας δίνει την πληροφορία για το ποιον ρόλο έχει ο χρήστης στο εκάστοτε κύκλο σπουδών ώστε να του δίνει πρόσβαση στο ανάλογο UI (admin UI/user UI).

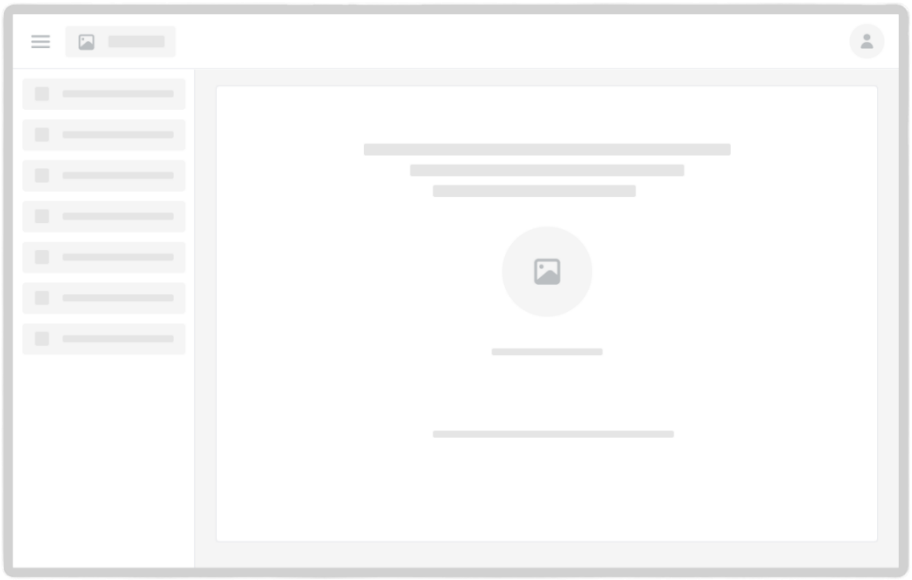
Κεφάλαιο 4ο: Εργαλεία σχεδίασης και υλοποίησης του Alumni

4.1 Figma

Το Figma είναι ένα διαδικτυακό εργαλείο σχεδιασμού που χρησιμοποιείται για τη δημιουργία και την προτυποποίηση διεπαφών χρήστη. Η διαδικτυακή του φύση διασφαλίζει επίσης ότι η εργασία μου είναι πάντα προσβάσιμη και εφεδρική. Χρησιμοποίησα το Figma για να δημιουργήσω το πρωτότυπο του έργου μου, επειδή προσφέρει ισχυρές δυνατότητες σχεδιασμού, ένα διαισθητικό περιβάλλον εργασίας και τη δυνατότητα εύκολης δημιουργίας διαδραστικών πρωτοτύπων.

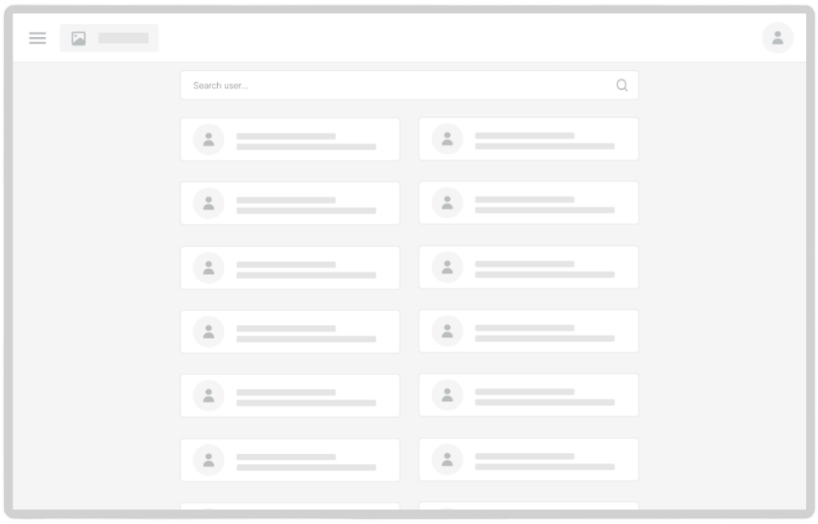
Η πλατφόρμα Alumni αποτελείται από δύο διαφορετικές διεπαφές χρήστη, αυτή του διαχειριστή και του απλού χρήστη/απόφοιτου. Έτσι μέσω του figma μπόρεσα και αποτύπωσα την αρχική ιδέα σχεδίασης, δημιουργώντας δύο πρωτότυπα, ένα για το καθένα. Πάνω σε αυτά τα προσχέδια βασίζεται η τελική μορφή των UIs, που θα παρουσιαστούν αργότερα στο κεφάλαιο Κεφάλαιο 5ο: Παρουσίαση του Alumni.

Ξεκινώντας με τη σελίδα του διαχειριστή, η ανάγκη της προβολής μεγάλης σε ποσότητα πληροφορίας κατά την χρήση διαφόρων διεργασιών, δεν καθιστούσε τις συνθήκες ιδανικές ώστε να υπάρξει mobile size UI. Έτσι, έλαβε μέρος η υλοποίηση της διεπαφής μόνο σε desktop size.

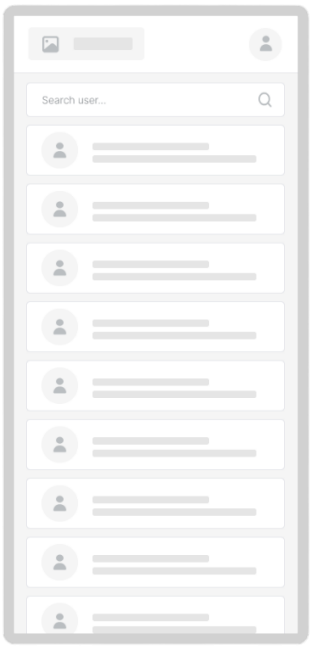


Εικόνα 5: Admin Layout Prototype

Όσον αφορά την σελίδα του απόφοιτου, διακρίνεται από μία απλότητα σε σχέση με αυτή του διαχειριστή καθώς απαλλάσσεται από τις διάφορες διοικητικές λειτουργίες. Αυτό έχει ως αποτέλεσμα στην δημιουργία ενός πιο φιλικού προς τον χρήστη UI το οποίο προσαρμόζεται σε κάθε μέγεθος οθόνης.



Εικόνα 6: User Layout Desktop



Εικόνα 7: User Layout Mobile

4.2 Github

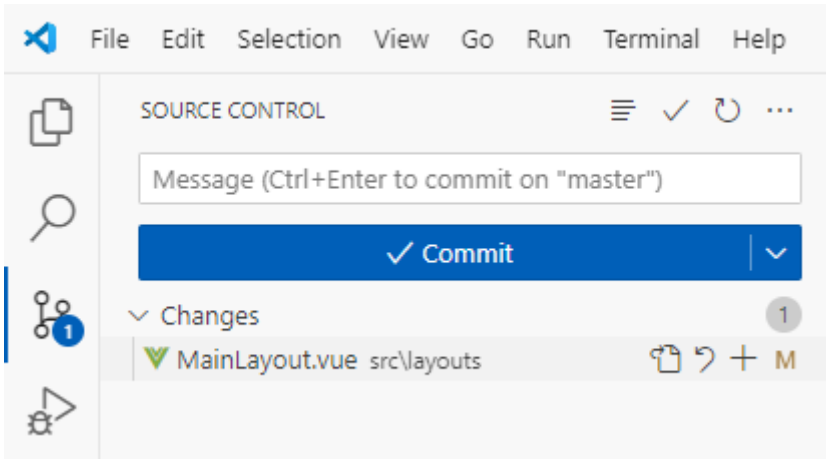
Το GitHub είναι μια πλατφόρμα ανάπτυξης λογισμικού που παρέχει ηλεκτρονικό χώρο αποθήκευσης και διαχείρισης πηγαίου κώδικα, επιτρέποντας τη συνεργασία μεταξύ προγραμματιστών και ομάδων ανάπτυξης. Προσφέρει μια ευρεία γκάμα εργαλείων και δυνατοτήτων που διευκολύνουν τη διαχείριση των έργων των χρηστών. Συγκεκριμένα, περιλαμβάνει τη δυνατότητα αναφοράς θεμάτων (issues) για

την καταγραφή προβλημάτων ή την υποβολή νέων ιδεών, τη δυνατότητα σχολιασμού στον κώδικα για την ενθάρρυνση της συζήτησης και της ανάδρασης, καθώς και την προσθήκη συνεργατών (collaborators) για να συμμετέχουν στην ανάπτυξη του έργου. Ένα από τα κύρια πλεονεκτήματα του GitHub είναι η δυνατότητα παρακολούθησης της εξέλιξης του έργου μέσω του συστήματος ελέγχου εκδόσεων (version control system). Αυτό επιτρέπει την παρακολούθηση των αλλαγών που έχουν γίνει στον κώδικα και την επαναφορά παλιών εκδόσεων σε περίπτωση προβλημάτων [25]. Όλα τα παραπάνω οδήγησαν στην επιλογή χρήσης του για την ανάπτυξη του πρακτικού μέρους της πτυχιακής, προσφέροντας πλήρη έλεγχο του κώδικα, καθώς και τη δυνατότητα επίβλεψης του έργου από τον επιβλέποντα καθηγητή.

4.3 Visual Studio Code

Το Visual Studio Code (VS Code) είναι ένας δωρεάν επεξεργαστής κώδικα που αναπτύχθηκε από τη Microsoft. Η επιλογή χρήσης του VS Code για να υλοποίηση της εφαρμογής έγινε επειδή προσφέρει ισχυρά χαρακτηριστικά όπως ενσωματωμένο τερματικό, ενσωματωμένο εντοπισμό σφαλμάτων και ενσωμάτωση με Git. Η εκτεταμένη βιβλιοθήκη επεκτάσεων και η δυνατότητα προσαρμογής του περιβάλλοντος εργασίας έκαναν τη διαδικασία ανάπτυξης αποδοτική και ευχάριστη [26].

Στο κομμάτι χρήσης του, τα προαπαιτούμενα βήματα πριν ξεκινήσει η χρήση του είναι η δημιουργία ενός repository στο GitHub αλλά και ενός φακέλου στο τοπικό σύστημα εργασίας. Επόμενο βήμα είναι το άνοιγμα του φακέλου με την χρήση του VS Code και με την βοήθεια του τερματικού (terminal) του επεξεργαστή, γίνεται η εκτέλεση του εντολής `git init`. Με την εκτέλεση της, δημιουργείται τοπικά ένα νέο κενό repository Git στο τοπικό σύστημα αρχείων. Αυτό σημαίνει ότι ξεκίνησε ένα νέο ιστορικό εκδόσεων στον τρέχοντα κατάλογο εργασίας. Στην συνέχεια με την χρήση της εντολής `git remote add origin <URL>` όπου URL είναι ο σύνδεσμος του repository στο GitHub επιτυγχάνεται η σύνδεση μεταξύ των δύο και πλέον η κάθε αλλαγή που γίνεται τοπικά μπορεί να σταλεί άμεσα στο GitHub και να αποθηκευτεί εκεί. Το VS Code περιέχει εγκατεστημένο στο σύστημα του ένα εργαλείο διαχείρισης των αλλαγών που επιτρέπει την άμεση αλληλεπίδραση με το Git χωρίς να χρειάζεται πλέον η χρήση εντολών στην κονσόλα.



Εικόνα 8: Source control tool

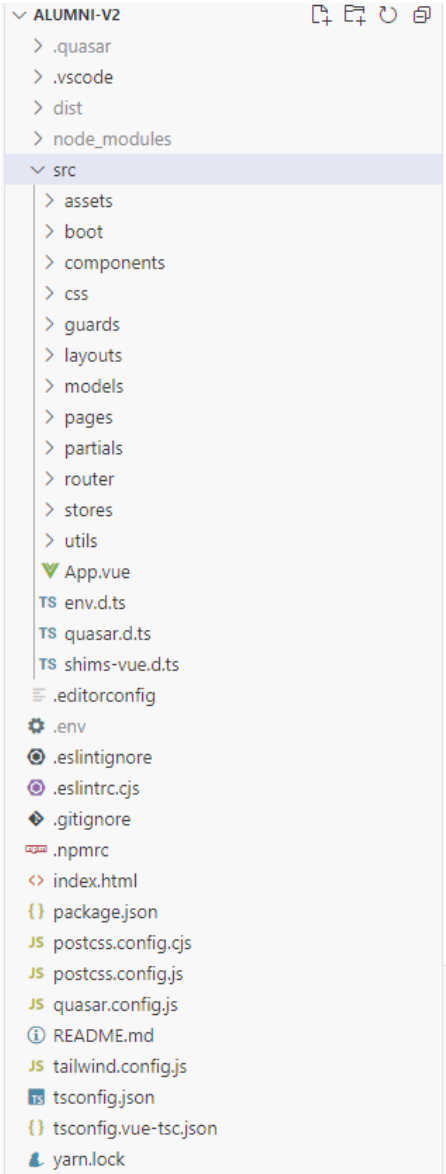
Όπως μπορεί κανείς να παρατηρήσει στην Εικόνα 8, μέσω αυτού του εργαλείο ο προγραμματιστής έχει την δυνατότητα να δει πόσα αρχεία έχουν δεχτεί επεξεργασία, ποια είναι αυτά καθώς και τη δυνατότητα να προσθέτει αρχεία στην περιοχή καταγραφής (staging area) πριν από τη δημιουργία του commit, κάνοντας κλικ στο εικονίδιο συν (+) δίπλα στο όνομα του αρχείου. Στην συνέχεια εισάγοντας ένα μήνυμα μπορεί να πραγματοποιήσει το commit με το οποίο οι αλλαγές είναι πλέον έτοιμες να συγχρονιστούν με το απομακρυσμένο repository κάνοντας push. Φυσικά, υπάρχει και η εντολή pull με την οποία ο χρήστης μπορεί να λάβει τυχών απομακρυσμένες αλλαγές στο τοπικό κατάλογο εργασίας του.

Το επόμενο βήμα είναι η δημιουργία του αρχικού project μέσω της χρήσης του Quasar Framework που έχει αναφερθεί σε προηγούμενο κεφάλαιο (2.2.7 Quasar Framework). Με την χρήση της εντολής `yarn quasar create` στον τερματικό του VS Code ξεκινά η δημιουργία ενός νέου έργου Quasar μέσω του Yarn, ενός διαχειριστή πακέτων για JavaScript. Η εντολή εκκινεί έναν διαδραστικό οδηγό που καθοδηγεί τους προγραμματιστές στη διαδικασία δημιουργίας. Θα ζητηθούν διάφορες επιλογές και προτιμήσεις για τη ρύθμιση του έργου, όπως το όνομα του, τα χαρακτηριστικά και τα πρόσθετα που θα συμπεριληφθούν (TypeScript, ESLint, Pinia κ.λπ.). Μετά την εισαγωγή των απαραίτητων πληροφοριών και επιλογών, η Quasar δημιουργεί τη δομή των περιεχομένων του project. Αυτό περιλαμβάνει όλα τα αρχεία και τους φακέλους που απαιτούνται για την ορθή λειτουργία της εφαρμογής. Μόλις ολοκληρωθεί η διαδικασία, το νέο έργο είναι έτοιμο για ανάπτυξη. Οι προγραμματιστές μπορούν να αρχίσουν να γράφουν κώδικα, να προσθέτουν συστατικά (components) και να προσαρμόζουν την εφαρμογή σύμφωνα με τις ανάγκες τους. Εκτός από τα βασικά πρόσθετα που εγκαταστάθηκαν μέσω της παραπάνω διαδικασίας, έγινε και η εγκατάσταση της βιβλιοθήκης Tailwind με την χρήση της εντολής `yarn add tailwindcss`.

Με την υλοποίηση όλων των παραπάνω το project έχει λάβει την βασική δομή του (Εικόνα 9), με όλους τους φακέλους και τα αρχεία που απαιτούνται για την λειτουργία του, αλλά και αυτών που δημιουργήθηκαν στην πορεία λόγω προσωπικής προτίμησης για αποδοτικότερη οργάνωση. Ακολουθεί μία σύντομη περιγραφή μερικών εξ αυτών:

- **src**: Ο κύριος φάκελος του κώδικα της εφαρμογής. Περιέχει όλα τα αρχεία και τους υποφακέλους που σχετίζονται με την ανάπτυξη της εφαρμογής:
- **assets**: Αρχεία πολυμέσων (π.χ. εικόνες, γραμματοσειρές).
- **boot**: Αρχεία εκκίνησης που εκτελούνται πριν την εκκίνηση της εφαρμογής.
- **components**: Επαναχρησιμοποιήσιμα Vue components.
- **css**: Αρχεία στυλ CSS που χρησιμοποιούνται στην εφαρμογή.
- **guards**: Αρχεία για guards των routes, τα οποία ελέγχουν την πρόσβαση σε διαφορετικά μέρη της εφαρμογής.
- **layouts**: Αρχεία διάταξης που ορίζουν τη βασική δομή της εφαρμογής.
- **models**: Αρχεία που ορίζουν τα μοντέλα δεδομένων που χρησιμοποιούνται στην εφαρμογή.
- **pages**: Κύριες σελίδες της εφαρμογής.
- **partials**: Μικρότερα κομμάτια σελίδων ή components που επαναχρησιμοποιούνται.
- **router**: Ορισμοί των routes της εφαρμογής.
- **stores**: Pinia stores για το state management της εφαρμογής.
- **utils**: Βοηθητικά αρχεία και λειτουργίες που χρησιμοποιούνται στην εφαρμογή.
- **App.vue**: Το κύριο αρχείο Vue της εφαρμογής.
- **index.html**: Το κύριο HTML αρχείο που φορτώνει την εφαρμογή Vue.

- **quasar.config.js**: Απαραίτητο αρχείο για τις ρυθμίσεις της Quasar. Δημιουργείται αυτόματα μετά την εκτέλεση εντολής `yarn quasar create`.
- **tailwind.config.js**: Απαραίτητο αρχείο για τις ρυθμίσεις για της Tailwind. Το αρχείο δεν δημιουργείται αυτόματα και ο χρήστης οφείλει να το δημιουργήσει χειροκίνητα για να λειτουργήσει η βιβλιοθήκη [19].



Εικόνα 9: Δομή Quasar Project

Τέλος, με την εντολή `yarn dev` γίνεται η εκκίνηση του τοπικού διακομιστή ανάπτυξης, επιτρέποντας στους προγραμματιστές να εκτελούν την εφαρμογή τους σε περιβάλλον ανάπτυξης. Αυτός ο διακομιστής παρέχει λειτουργίες όπως live reloading, που ανανεώνει αυτόματα την εφαρμογή στον περιηγητή κάθε φορά που γίνονται αλλαγές στον κώδικα, διευκολύνοντας την άμεση παρακολούθηση των αλλαγών και τη γρήγορη διόρθωση σφαλμάτων.

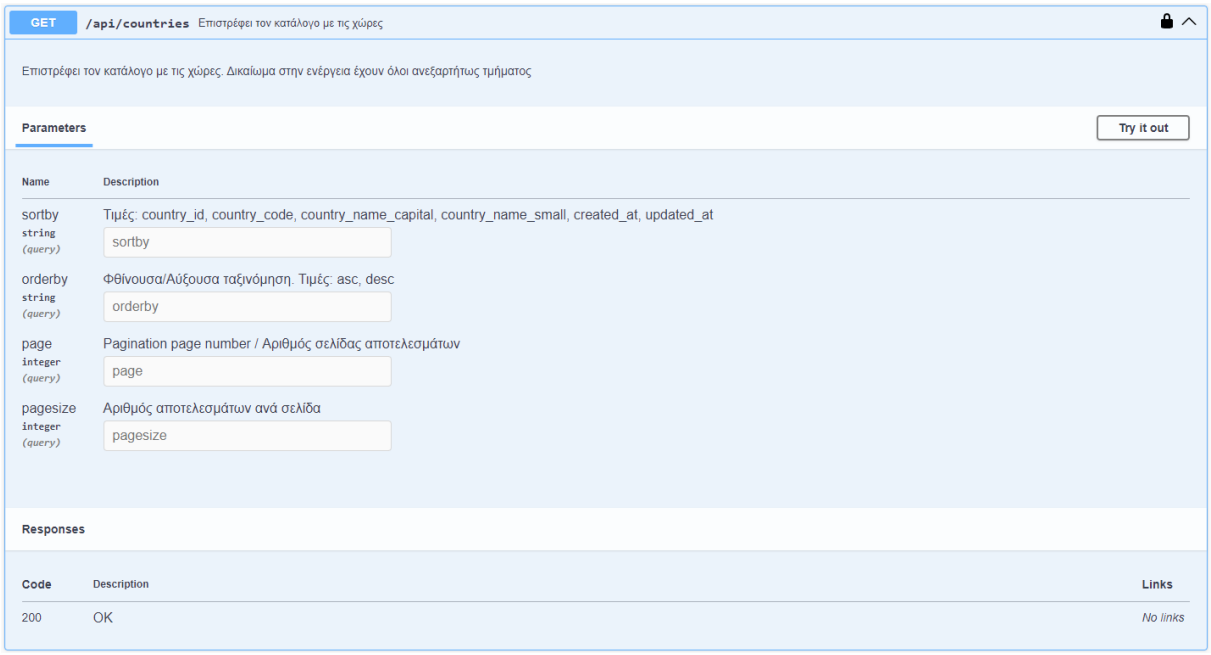
4.4 Swagger

Το Swagger είναι ένα εργαλείο κλειδί στην τεκμηρίωση και ανάπτυξη των API, ξεχωρίζοντας για τη δυνατότητά του να αυτοματοποιεί τη δημιουργία ολοκληρωμένης και διαδραστικής περιγραφής για RESTful API [27]. Ένα από τα σημαντικά πλεονεκτήματα του Swagger είναι η δυνατότητα δημιουργίας λεπτομερών και σαφών τεκμηριώσεων για κάθε endpoint του API. Αυτή η δυνατότητα επιτρέπει στους προγραμματιστές να παρουσιάζουν τις λειτουργίες του API με τρόπο που είναι τόσο κατανοητός όσο και οπτικά ελκυστικός. Η χρήση αυτού του εργαλείου στην δεύτερη έκδοση του Alumni, μου έλυσε τα χέρια καθώς μπορούσα να τεστάρω άμεσα οποιοδήποτε endpoint χωρίς κόπο. Σε αυτό συνέβαλε η λεπτομερής περιγραφή των παραμέτρων ή των δεδομένων των HTTP αιτημάτων καθώς και των ενδεικτικών απαντήσεων, που χρησιμοποιούνταν σαν οδηγός με αποτέλεσμα την άψογη εμπειρία χρήσης.

Η μορφή που παρουσιάζονται routes είναι αυτή που φαίνεται στην Εικόνα 10. Η κάθε εγγραφή αποτελείται από τα endpoints που μπορεί να χρησιμοποιήσει ο προγραμματιστής καθώς και το HTTP αίτημα που αντιστοιχεί στο καθένα.

Εικόνα 10: Παράδειγμα μορφής Api interface στο Swagger

Κάνοντας κλικ σε κάποιο endpoint τότε θα εμφανιστούν περεταίρω πληροφορίες όπως αναπαριστά η Εικόνα 11. Παρέχεται μια ολοκληρωμένη επισκόπηση της λειτουργίας του. Δίνει τη δυνατότητα στους χρήστες να προσαρμόσουν τα αποτελέσματα χρησιμοποιώντας διάφορες παραμέτρους, επιτρέποντας την ευέλικτη ταξινόμηση και οργάνωση των δεδομένων Επίσης, παρέχει ενδεικτικές τιμές για κάθε παράμετρο, διευκολύνοντας τον χρήστη στην κατανόηση της χρήσης τους. Στο περιεχόμενο περιλαμβάνεται και ένα διαδραστικό εργαλείο που επιτρέπει στους χρήστες να δοκιμάσουν την εντολή και να δουν ενδεικτική απάντηση, προσφέροντας έναν εύκολο τρόπο πειραματισμού και επαλήθευσης της λειτουργίας του API.



Εικόνα 11: Παράδειγμα τεκμηρίωσης API

4.5 phpMyAdmin

Το phpMyAdmin αποτελεί ένα ισχυρό εργαλείο γραμμένο σε PHP που χρησιμοποιείται για τη διαχείριση βάσεων δεδομένων MySQL και MariaDB μέσω ενός online γραφικού περιβάλλοντος χρήστη. Παρέχει ένα ευρύ φάσμα δυνατοτήτων, όπως η εισαγωγή και εξαγωγή δεδομένων σε διάφορες μορφές (CSV, SQL, XML, κλπ.), η εκτέλεση SQL εντολών για τη διαχείριση και τροποποίηση των δεδομένων, και η προβολή των δομών των πινάκων με αναλυτικές πληροφορίες για τις στήλες και τα ευρετήρια [28].

Επιπλέον, το phpMyAdmin επιτρέπει τη δημιουργία και διαγραφή βάσεων δεδομένων και πινάκων, την τροποποίηση δομών πινάκων, και την πρόσβαση στα δικαιώματα των χρηστών, καθιστώντας το ένα απαραίτητο εργαλείο για διαχειριστές βάσεων δεδομένων. Όσον αφορά την εφαρμογή, η χρήση του ήταν κυρίως για testing, καθώς επεξεργαζόμουν τιμές δεδομένων, όπως ο ρόλος του χρήστη, ώστε να ελέγχω τη λειτουργικότητα της πλατφόρμας.

Κεφάλαιο 5ο: Παρουσίαση του Alumni

5.1 Εισαγωγή

Στο κεφάλαιο αυτό, παρουσιάζεται μια ολοκληρωμένη περιγραφή της πλατφόρμας, συνοδευόμενη από επεξηγήσεις των βασικών λειτουργιών της. Χρησιμοποιούνται εικόνες του τελικού προϊόντος ώστε ο αναγνώστης να έχει μία οπτική αναπαράσταση των λειτουργιών δείχνοντας τη διασύνδεση των διαφόρων στοιχείων και τον τρόπο με τον οποίο ο χρήστης αλληλεπιδρά με την πλατφόρμα.

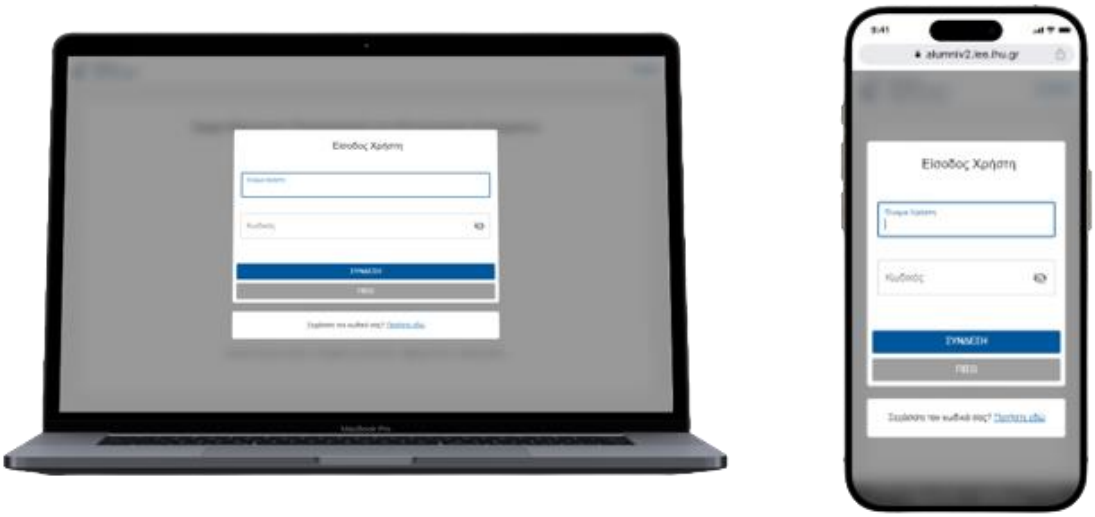
5.2 Αρχική σελίδα

Όπως είναι ήδη γνωστό, για την υλοποίηση της πλατφόρμας έγινε η ανάπτυξη δύο διαφορετικών διεπαφών, αυτή του διαχειριστή και αυτή του απλού χρήστη/απόφοιτου. Το κοινό τους στοιχείο είναι η αρχική σελίδα η οποία έχει προσαρμοστεί σε όλα τα μεγέθη οθόνης παρόλο που αν συνδεθείς ως διαχειριστής η χρήση του UI είναι εφικτή μόνο σε desktop μέγεθος οθόνης.

Εικόνα 12: Αρχική σελίδα

5.3 Σύνδεση χρήστη

Η σύνδεση του χρήστη δεν λαμβάνει μέρος σε νέα σελίδα. Αντί αυτού, αποτελεί μέρος της αρχικής σελίδας μέσω ενός αναδυόμενου παραθύρου που εμφανίζεται όταν ο χρήστης πατήσει το κουμπί “σύνδεση”.



Εικόνα 13: Σύνδεση χρήστη

Αφού ο χρήστης πληκτρολογήσει τα διαπιστευτήρια του και πατήσει το κουμπί “σύνδεση” στο παρασκήνιο εκτελούνται συνδυαστικά τέσσερα αιτήματα στο api. Επειδή το ένα εξαρτάται από τα δεδομένα της απάντησης του προηγούμενου λαμβάνουν χώρα σε μία ασύγχρονη μέθοδο ώστε να εκτελεστούν με την σωστή σειρά.

Το πρώτο που εκτελείται είναι το login, όπου εφόσον τα διαπιστευτήρια είναι σωστά, δημιουργείται ένα token και επιστρέφεται ως απάντηση μαζί με το user_id του χρήστη που συνδέθηκε. Και τα δύο αποθηκεύονται στο local storage του browser με το token να χρησιμοποιείται σε κάθε επόμενο api call για την αυθεντικοποίηση και το user_id όπου κρίνεται απαραίτητη η χρήση του.

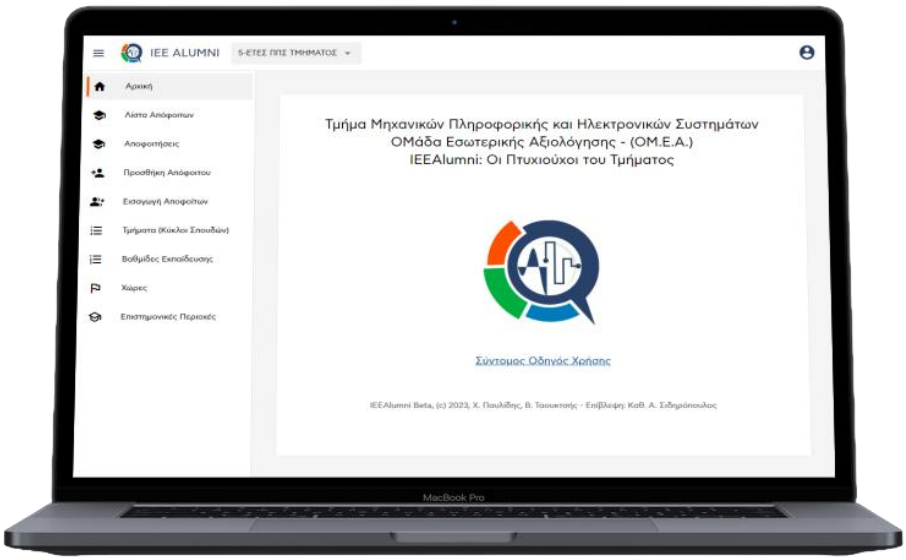
Ακολουθεί το αίτημα για τη λήψη των ρόλων του χρήστη. Η επιστρεφόμενη πληροφορία είναι ένας πίνακας, όπου κάθε στοιχείο περιέχει το πρόγραμμα σπουδών από το οποίο αποφοίτησε ο χρήστης και τον ρόλο που έχει σε αυτό, με τα αντίστοιχα ids τους.

Επειδή στην διεπαφή είναι απαραίτητη η προβολή του προγράμματος σπουδών αποφοίτησης του χρήστη, πράγμα αδύνατο μόνο με την χρήση του degree_id, ακολουθεί το αίτημα ανάκτησης της λίστας των προγραμμάτων σπουδών η οποία χρησιμοποιείται για να γίνει το mapping μεταξύ του κοινού τους στοιχείου degree_id, για την ανάκτηση του ονόματος.

Τέλος, λαμβάνει χώρα το αίτημα λήψης των στοιχείων του χρήστη που περιέχει απαραίτητη πληροφορία για προβολή.

5.4 UI διαχειριστή

Η πρώτη περίπτωση που θα εξεταστεί είναι η σύνδεση ενός χρήστη με ρόλο διαχειριστή. Με την ανάκτηση των ρόλων του χρήστη, γίνεται αμέσως ο έλεγχος αν ο χρήστης είναι διαχειριστής και ώστε να γίνει ι ανακατεύθυνση στο admin layout. Αυτό έχει την μορφή που βλέπετε στην Εικόνα 14.

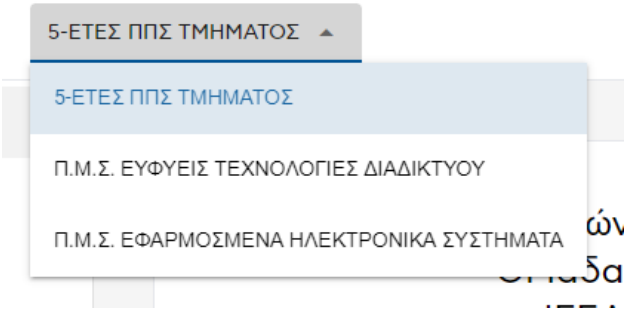


Εικόνα 14: Admin layout

Όπως εύκολα μπορεί να παρατηρήσει κανείς αποτελείται από τρία βασικά μέρη, την επικεφαλίδα στην κορυφή της σελίδας, το μενού πλοήγησης στα αριστερά και το κεντρικό μέρος προβολής που εναλλάσσεται αναλόγως την επιλογή στο μενού.

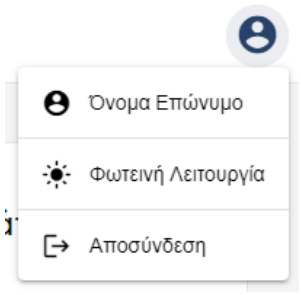
Ξεκινώντας με την επικεφαλίδα, σε αυτήν συναντάμε, αρχίζοντας από τα αριστερά προς τα δεξιά, το εξής περιεχόμενο:

- Το κουμπί εμφάνισης και απόκρυψης του μενού πλοήγησης.
- Το logo με τον τίτλο της υπηρεσίας.
- Την προβολή του προγράμματος σπουδών που αποφοίτησε ο χρήστης. Σε περίπτωση που έχει αποφοιτήσει σε περισσότερα από ένα αυτό μετατρέπεται σε ένα dropdown που επιτρέπει στον χρήστη να επιλέξει από ποιο από αυτά θα λαμβάνει πληροφορίες. Υπάρχει περίπτωση να επιλέξει κάποιο πρόγραμμα σπουδών στο οποίο δεν έχει ρόλο διαχειριστή. Σε εκείνη την περίπτωση να γίνει ανακατεύθυνση στο user layout.



Εικόνα 15: Degree selection

- Το εικονίδιο του συνδεδεμένου χρήστη όπου πατώντας το, εμφανίζεται μενού με διάφορες λειτουργίες όπως η προβολή του προφίλ του χρήστη, η αλλαγή του θέματος της εφαρμογής και η αποσύνδεση.



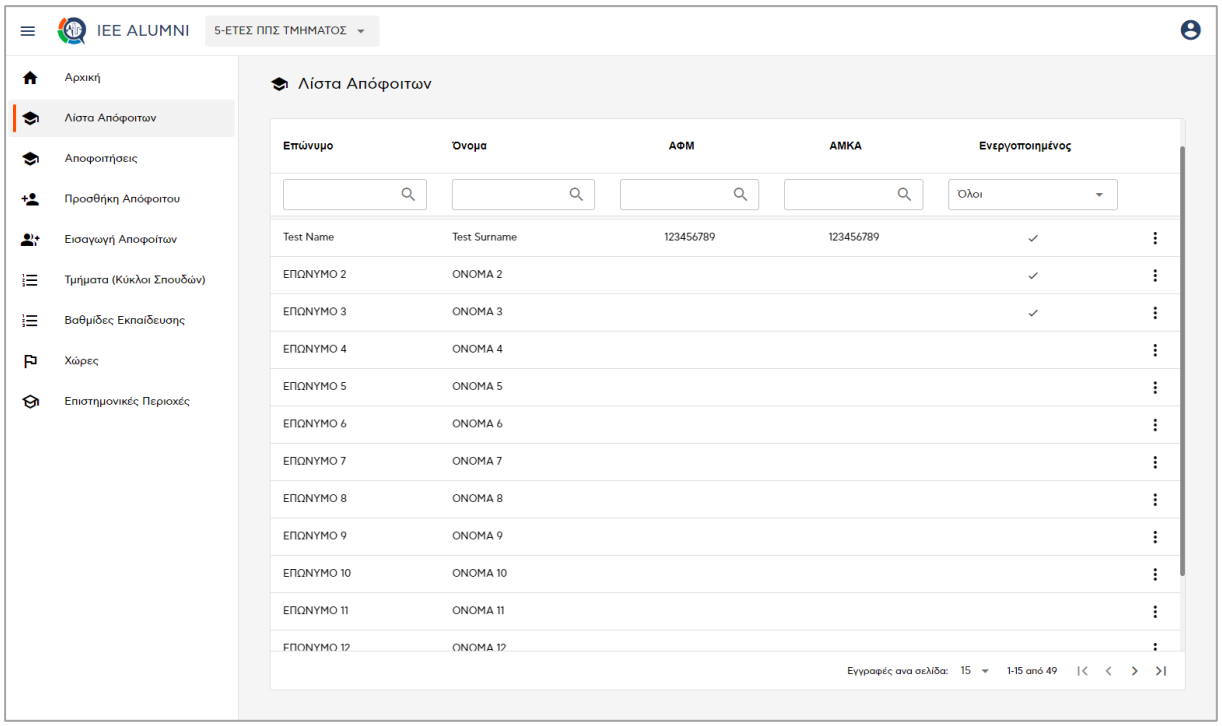
Εικόνα 16: Μενού χρήστη

Το μενού πλοήγησης είναι υπεύθυνο για την προβολή του περιεχομένου που λαμβάνει χώρα στο κεντρικό σημείο της ιστοσελίδας. Παρουσιάζει και αποκρύπτει δυναμικά το περιεχόμενο της λίστας αναλόγως με τον ρόλο που έχει ο χρήστης.

Όσον αφορά το μέρος προβολής του περιεχομένου, διακρίνεται από την δυναμική φύση του, καθώς όταν ο χρήστης επιλέγει από το μενού την προβολή κάποιας άλλης σελίδας, δεν χρειάζεται να γίνει ανανέωση όλου του περιεχομένου αλλά μόνο αυτού που περιέχεται στο συγκεκριμένο section.

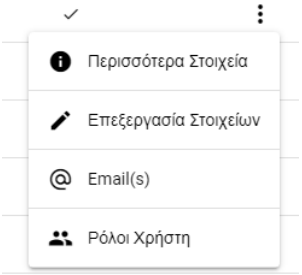
5.4.1 Σελίδα λίστας αποφοίτων

Η σελίδα λίστας αποφοίτων αποτελείται από έναν πίνακα ο οποίος προβάλει τους αποφοίτους του προγράμματος σπουδών που είναι επιλεγμένο στη γραμμή εργασιών (Εικόνα 15). Περιέχει στοιχεία όπως το επώνυμο, το όνομα, το ΑΦΜ, το ΑΜΚΑ και το αν είναι ενεργοποιημένος ο χρήστης. Σε κάθε στήλη υπάρχει η δυνατότητα αναζήτησης με βάσει το πεδίο της. Στο κάτω μέρος του πίνακα υπάρχουν επιλογές σελιδοποίησης για την διαχείριση των αποτελεσμάτων που προβάλλονται.



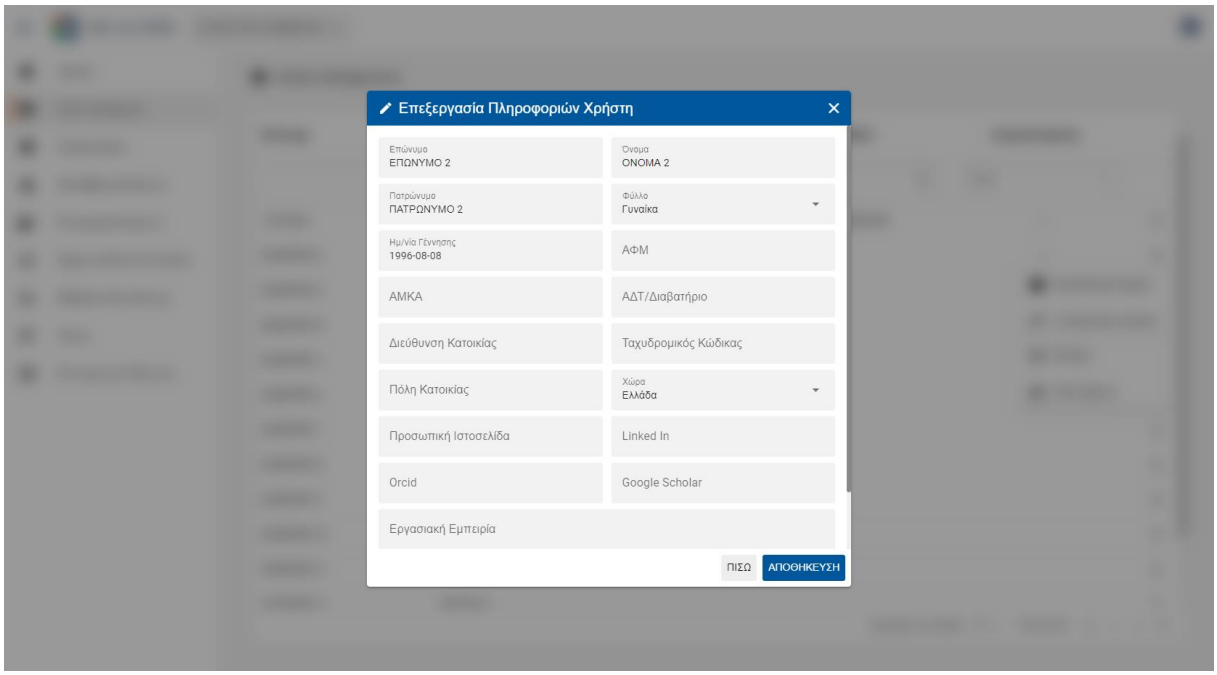
Εικόνα 17: Σελίδα λίστας αποφοίτων

Το κομμάτι της διαχείρισης των αποφοίτων γίνεται με την βοήθεια της τελευταίας στήλης όπου υπάρχει ένα κουμπί που εμφανίζει ένα μενού επιλογών διαχείρισης κάθε εγγραφής.



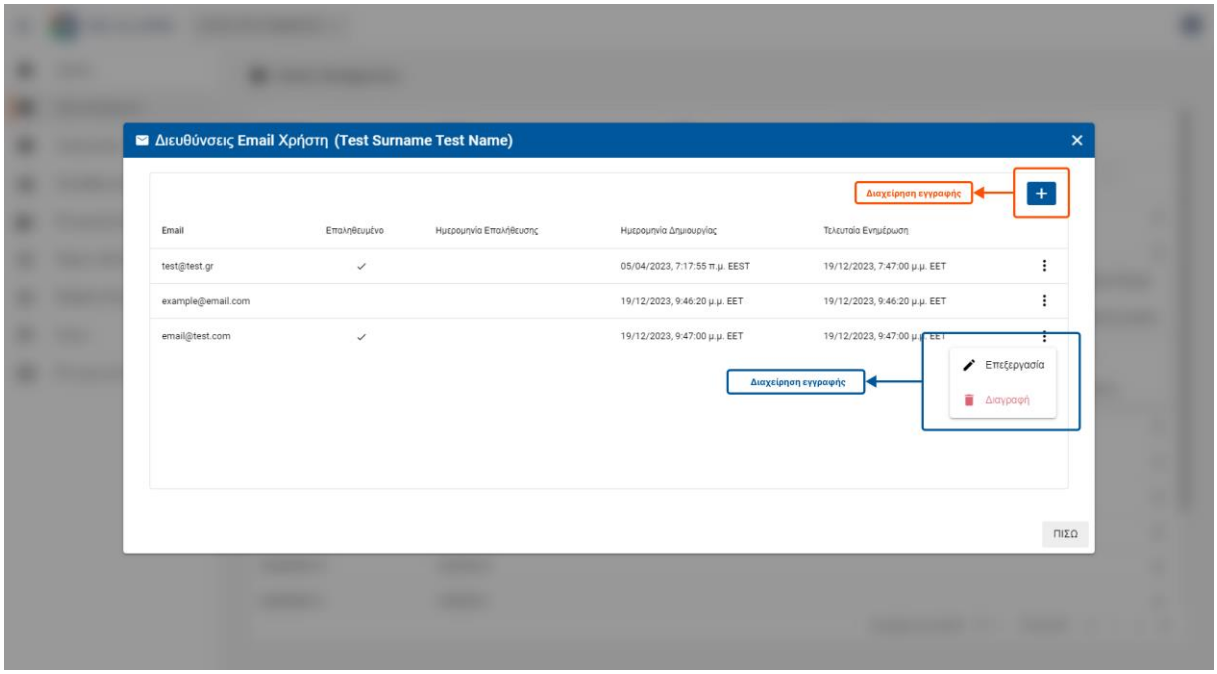
Εικόνα 18: Μενού διαχείρισης εγγραφών του πίνακα

Οι δύο πρώτες επιλογές εμφανίζουν ένα αναδυόμενο παράθυρο με περισσότερες πληροφορίες, με την διαφορά ότι στην μία υπάρχει δυνατότητα προβολής ενώ στην δεύτερη επιτρέπεται και η επεξεργασία.



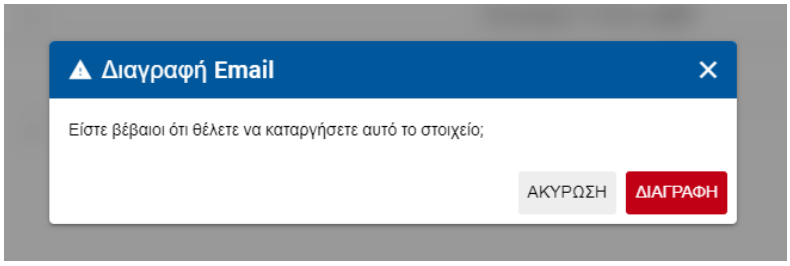
Εικόνα 19: Επεξεργασία αποφοίτου

Η επιλογή “email(s)” ακολουθεί το ίδιο μοτίβο λειτουργίας, με την εμφάνιση ενός αναδυόμενου παραθύρου που περιέχει έναν πίνακα με τα emails του χρήστη και τις πληροφορίες αυτών. Ο πίνακας αυτός ακολουθεί την ίδια λογική με τον πίνακα των αποφοίτων δίνοντας την δυνατότητα προσθήκης νέου email, επεξεργασίας και διαγραφής των ήδη υπαρχόντων.



Εικόνα 20: Emails αποφοίτου

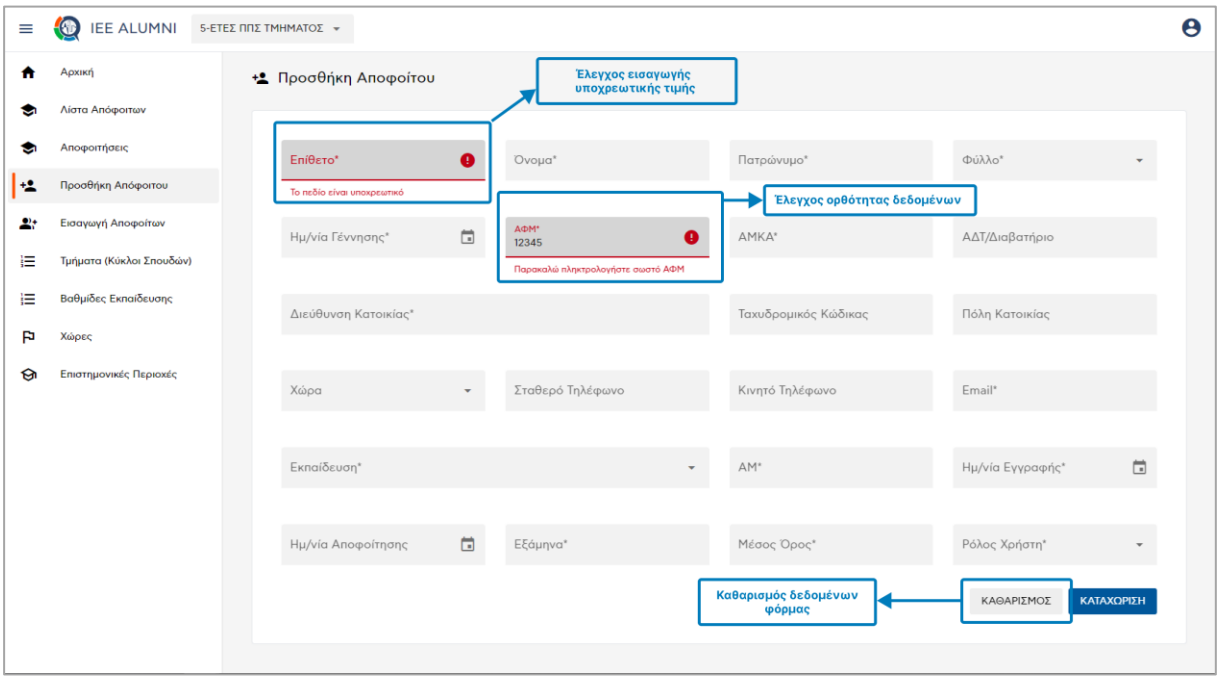
Επιλέγοντας την διαγραφή ενός email εμφανίζεται στην οθόνη ένα popup επιβεβαίωσης αποτρέποντας την διαγραφή της εγγραφής από λανθασμένη επιλογή στο μενού. Αυτό συμβαίνει σε κάθε οντότητα της εφαρμογής που έχει την επιλογή της διαγραφής.



Εικόνα 21: Delete confirmation

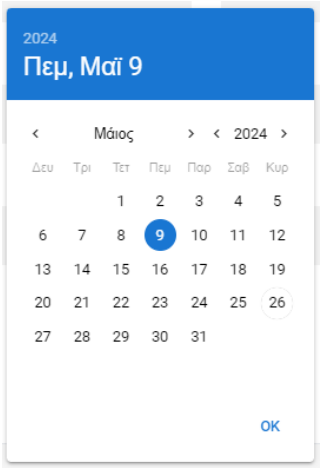
5.4.2 Προσθήκη απόφοιτου

Σε αυτή τη σελίδα ο χρήστης έχει την δυνατότητα εισαγωγής μεμονωμένων εγγραφών στην λίστα των αποφοίτων. Η φόρμα αποτελείται από όλα τα απαραίτητα στοιχεία του αποφοίτου. Διακρίνεται σε υποχρεωτικά και μη, όπου στα πρώτα υπάρχει ο κατάλληλος έλεγχος και η ειδοποίηση του χρήστη όπως βλέπουμε στην Εικόνα 22. Επίσης υπάρχει έλεγχος και για την ορθότητα των δεδομένων όπως για παράδειγμα το ΑΦΜ που πρέπει να αποτελείται από 10 ψηφία.



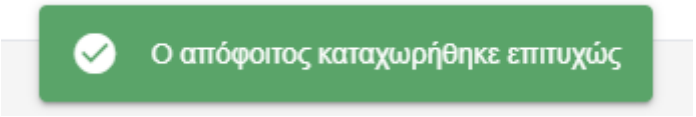
Εικόνα 22: Εισαγωγή αποφοίτου

Η εισαγωγή των ημερομηνιών γίνεται με την χρήση ειδικών popups που περιέχουν ένα ημερολόγιο για να την επιλογή της, δημιουργώντας ένα πιο φιλικό περιβάλλον για τον χρήστη. Αυτά τα popups εμφανίζονται κάνοντας κλικ στο εικονίδιο μέσα στο πεδίο εισαγωγής της ημερομηνίας.



Εικόνα 23: Εισαγωγή ημερομηνίας

Τέλος ο χρήστης έχει την δυνατότητα καταχώρισης του φοιτητή ή τον καθαρισμό τον δεδομένων με την χρήση των δύο κουμπιών που βρίσκονται στο τέλος της φόρμας εισαγωγής. Σε περίπτωση επιλογής καταχώρισης ο χρήστης θα ενημερωθεί για το αποτέλεσμα της ενέργειας αυτής με μια ειδοποίηση στο κάτω μέρος της σελίδας είτε σε περίπτωση επιτυχίας είτε σε περίπτωση αποτυχίας.



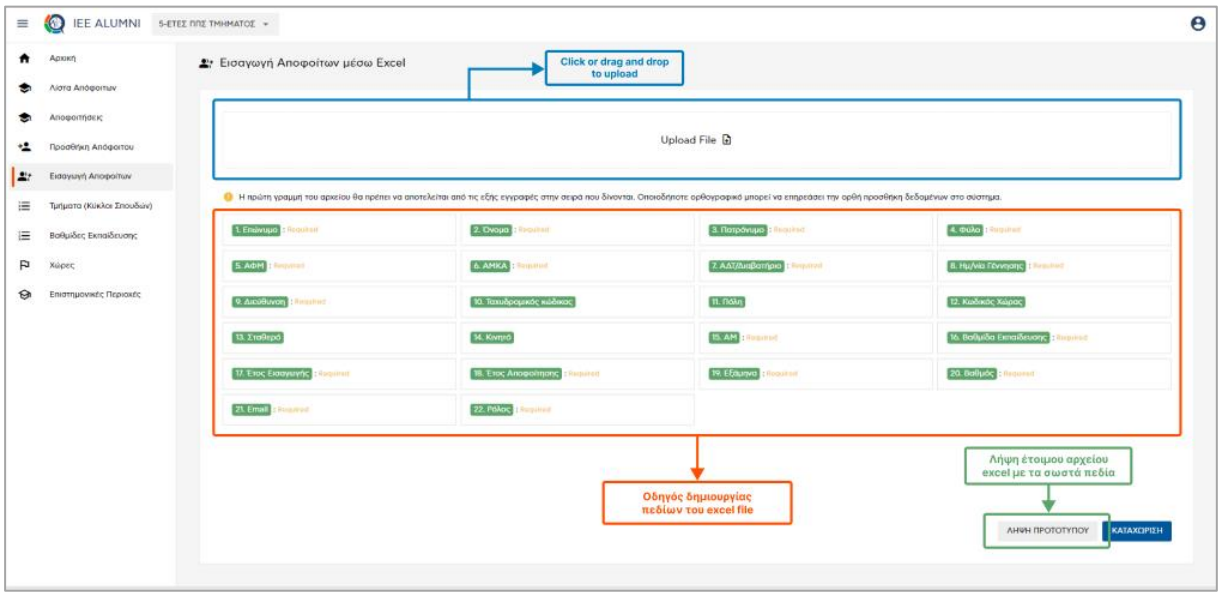
Εικόνα 24: Ειδοποίηση επιτυχίας

Εικόνα 25: Ειδοποίηση αποτυχίας

Οι παραπάνω ειδοποιήσεις λαμβάνουν μέρος σε οποιαδήποτε διεργασία γίνει στην εφαρμογή ενημερώνοντας τον χρήστη για την πορεία της, συμβάλλοντας στην εμπειρία χρήσης των λειτουργιών.

5.4.3 Εισαγωγή Αποφοίτων

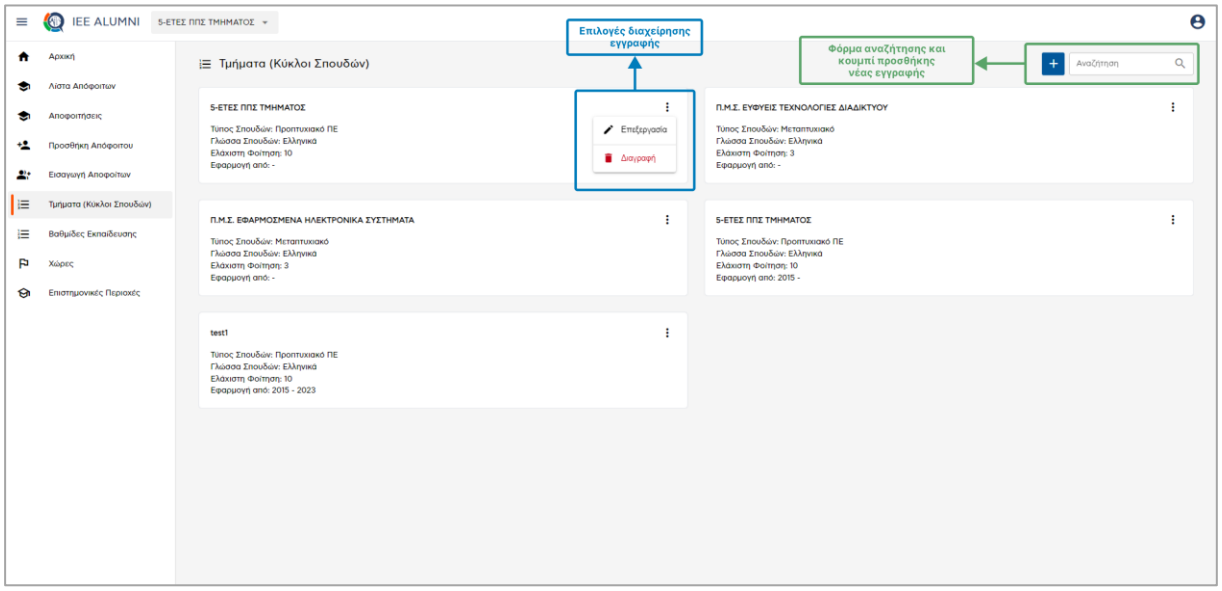
Σε αντίθεση με την προηγούμενη σελίδα που μπορούσε να γίνει εισαγωγή ενός αποφοίτου την φορά, σε αυτήν μπορείς να κάνεις εισαγωγή πολλαπλών εγγραφών με την χρήση ενός excel file. Κάνοντας click ή drag and drop στο upload file section της σελίδας και πιέζοντας το κουμπί καταχώρηση ξεκινάει η διαδικασία. Στην σελίδα ακόμη υπάρχουν πληροφορίες όπως τα ονόματα που πρέπει να έχουν οι στήλες του excel ώστε να γίνει η ορθή καταχώρηση τους. Τέλος υπάρχει το κουμπί “Λήψη πρωτότυπου” με το οποίο ο χρήστης έχει την δυνατότητα λήψης ενός αρχείου excel το οποίο συμπεριλαμβάνει ήδη τα σωστά πεδία και είναι έτοιμο για την εισαγωγή δεδομένων.



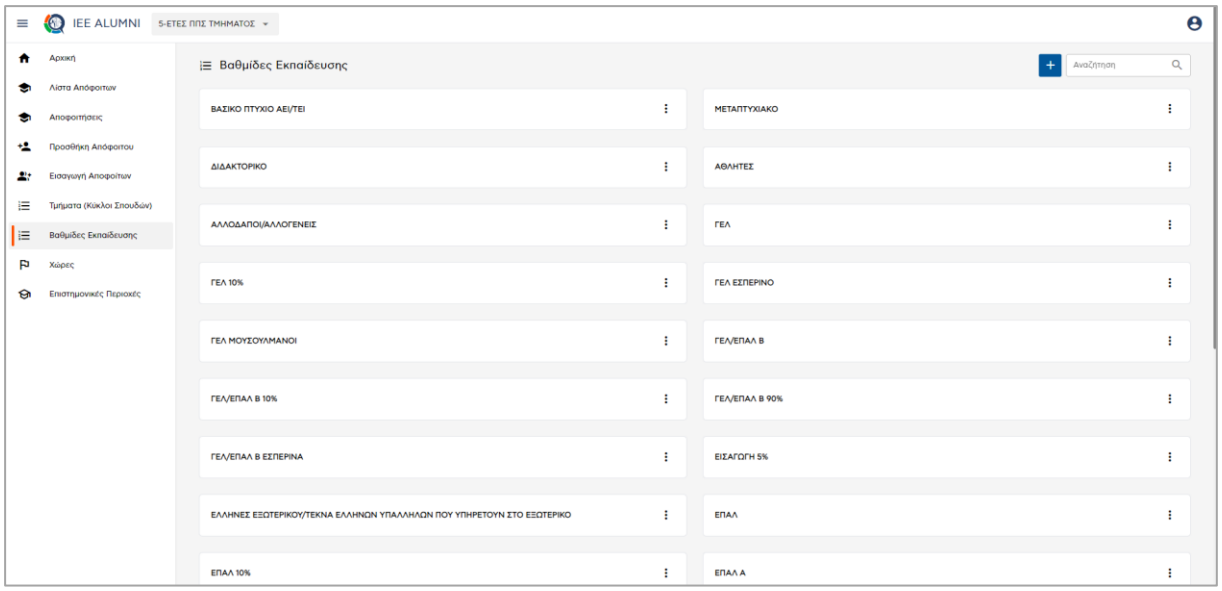
Εικόνα 26: Εισαγωγή αποφοίτων μέσω αρχείου excel

5.4.4 Τμήματα, Βαθμίδες Εκπαίδευσης, Χώρες, Επιστημονικές Περιοχές

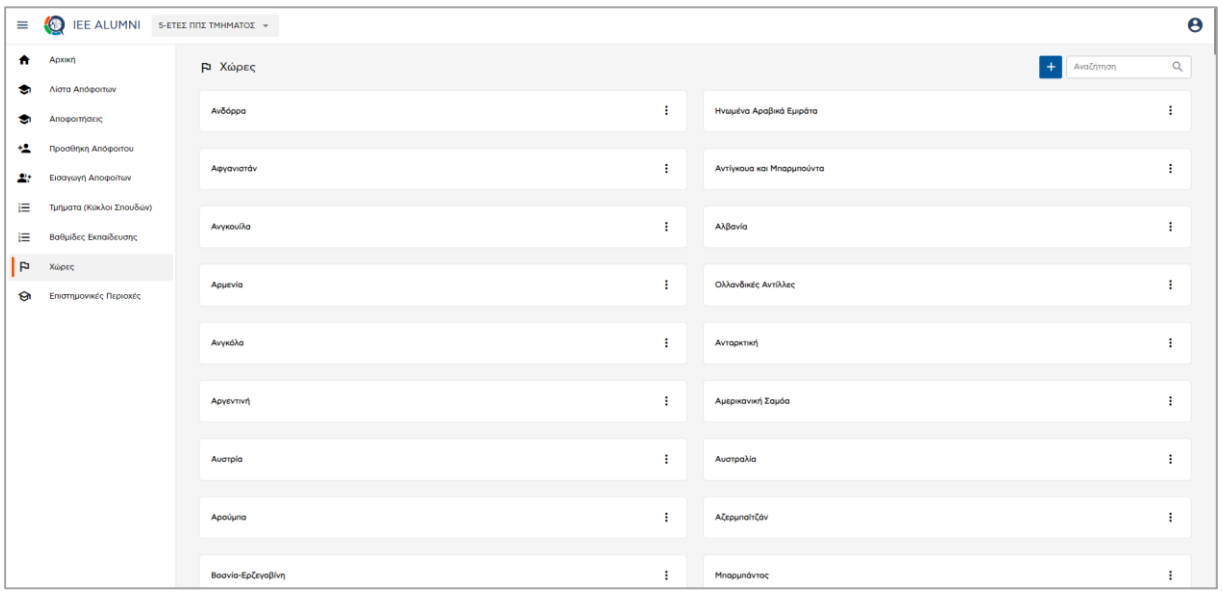
Και στις τέσσερις περιπτώσεις έχουμε σελίδες προβολής και διαχείρισης των εγγραφών δίνοντας την δυνατότητα αναζήτησης σε αυτές, δημιουργίας νέας εγγραφής, επεξεργασίας και διαγραφής των ήδη υπαρχουσών. Όλες οι λειτουργίες εκτελούνται με την ίδια λογική, δηλαδή με την χρήση αναδυόμενων παραθύρων.



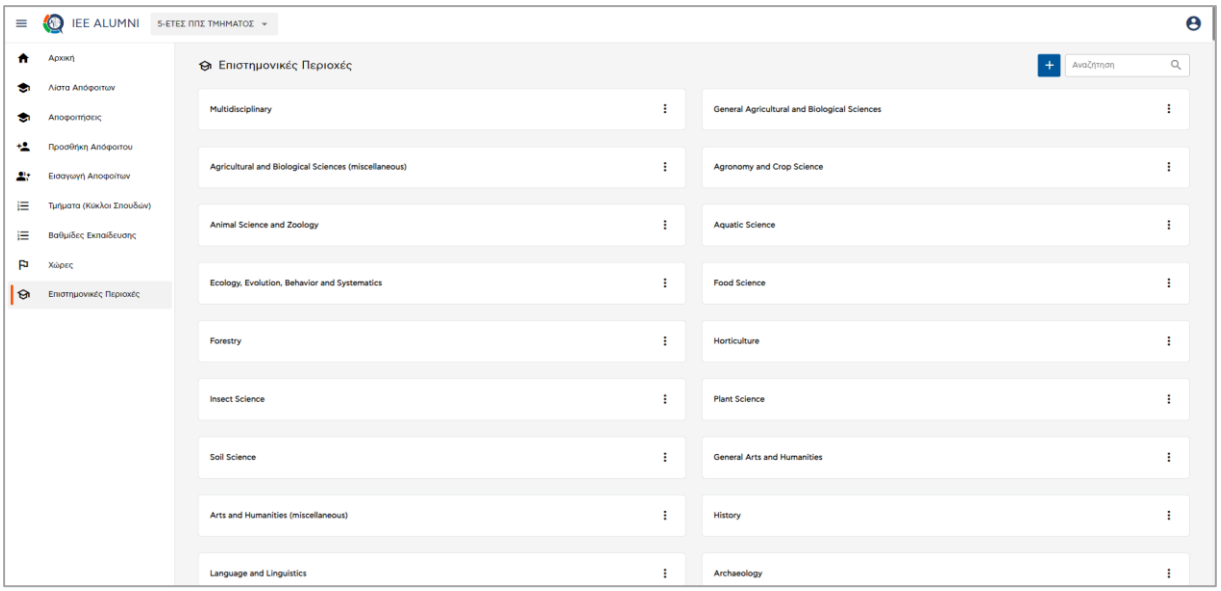
Εικόνα 27: Τμήματα (Κύκλοι σπουδών)



Εικόνα 28: Βαθμίδες Εκπαίδευσης



Εικόνα 29: Χώρες



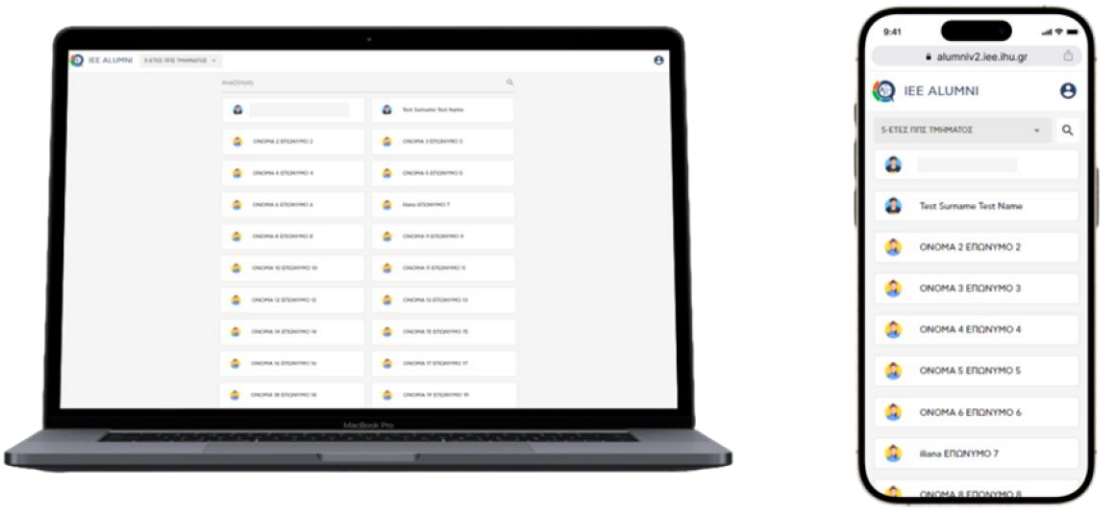
Εικόνα 30: Επιστημονικές Περιοχές

5.5 UI απλού Χρήστη/Απόφοιτου

Στην περίπτωση του UI που προορίζεται για τον απλό χρήστη/απόφοιτο, λόγω της απλότητας και της απαλλαγής του από τις διάφορες διοικητικές λειτουργίες, ήταν εφικτή η ανάπτυξη μιας ανταποκρινόμενης διεπαφής χρήστη που θα μπορεί να χρησιμοποιηθεί άνετα από οποιαδήποτε συσκευή. Πλέον δεν υπάρχει μεγάλος αριθμός σελίδων αλλά μόνο τρεις βασικές που δεν είναι άλλες από την λίστα των αποφοίτων, τις πληροφορίες του απόφοιτου που επιλέχθηκε προς προβολή και η ρυθμίσεις του προφίλ του συνδεδεμένου χρήστη.

5.5.1 Λίστα απόφοιτων

Με μια γρήγορη ματιά στην Εικόνα 31, εύκολα μπορεί να διακρίνει κανείς μερικές ομοιότητες με την διεπαφή του διαχειριστή.

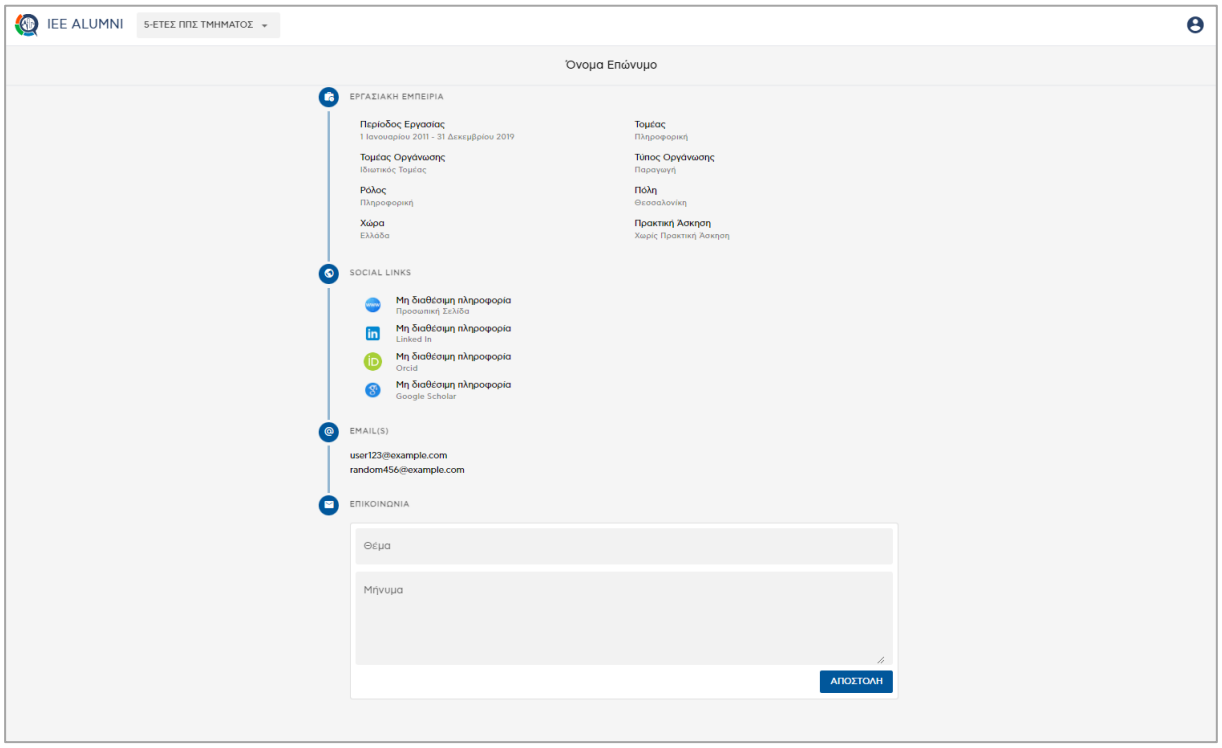


Εικόνα 31: Λίστα αποφοίτων

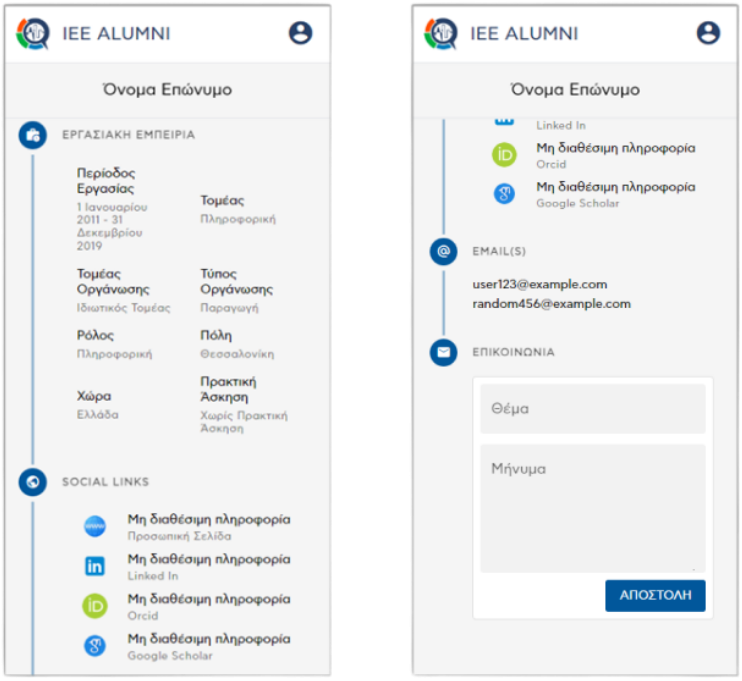
Όλες παρατηρούνται στην κορυφή της σελίδας όπου το navbar ακολουθεί την ίδια ακριβώς διάταξη αν εξαιρέσουμε το κουμπί εμφάνισης και απόκρυψης του μενού πλοήγησης, αφού το μενού αυτό δεν απουσιάζει. Όσον αφορά τα υπόλοιπα στοιχεία εκτός από το εμφανισιακό κομμάτι έχουν και ακριβώς την ίδια λειτουργικότητα καθώς ο χρήστης μπορεί να είναι απόφοιτος σε παραπάνω από ένα προγράμματα σπουδών. Έτσι απαιτείται η δυνατότητα εναλλαγής των αποτελεσμάτων βάσει της επιλογής του. Ακόμη, το εικονίδιο του συνδεδεμένου χρήστη περιέχει τις ίδιες επιλογές στο μενού του, αφού αποτελείται από βασικές λειτουργίες όπως η ρυθμίσεις του χρήστη και η αποσύνδεση που δεν μπορούν να απουσιάζουν.

5.5.2 Πληροφορίες Απόφοιτου

Επιλέγοντας κάποιον απόφοιτο, γίνεται ανακατεύθυνση σε νέα σελίδα στην οποία υπάρχουν περεταίρω πληροφορίες για αυτόν. Αυτές είναι η εργασιακή εμπειρία του, μια λίστα με τα social links του και τα emails του. Κάθε μία από αυτές τις πληροφορίες εξαρτάται από τον χρήστη αν θα είναι ορατή στους υπόλοιπους. Μέσω των ρυθμίσεων του, μπορεί να επεξεργαστεί την ορατότητα της κάθε πληροφορίας ξεχωριστά. Όπως καταλαβαίνει κανείς, υπάρχει πιθανότητα να απουσιάζουν όλες οι πληροφορίες του χρήστη. Σε αυτή την περίπτωση για να επικοινωνήσει κάποιος μαζί του, η πλατφόρμα δίνει την δυνατότητα αποστολής email μέσω της φόρμας εισαγωγής που φαίνεται στο τέλος της σελίδας. Με αυτόν τον τρόπο η επικοινωνία γίνεται χωρίς ο αποστολέας να γνωρίζει το email του παραλήπτη με αποτέλεσμα να διατηρείται η ιδιωτικότητα.



Εικόνα 32: User details desktop

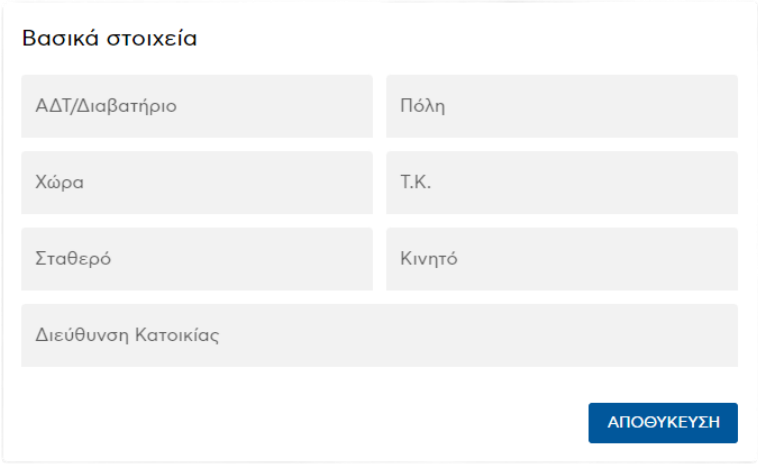


Εικόνα 33: User details mobile

5.6 Ρυθμίσεις Χρήστη

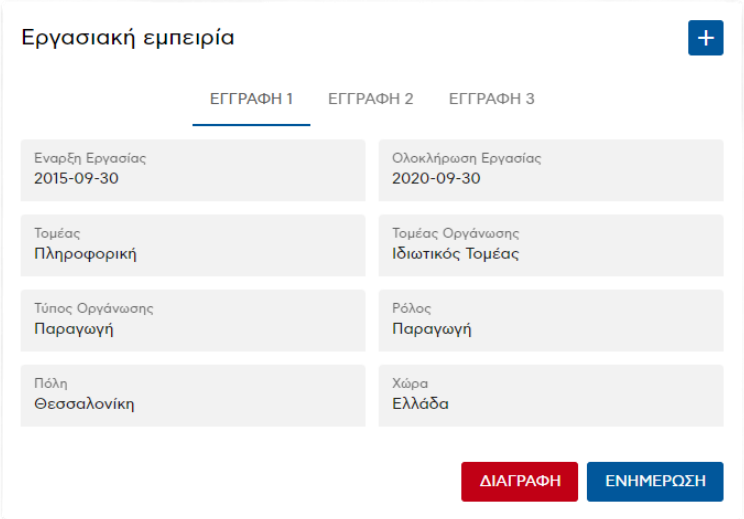
Η σελίδα αυτή είναι κοινή τόσο στο UI του διαχειριστή όσο και στου απλού χρήστη/απόφοιτου και αποτελεί την τυπική σελίδα ρυθμίσεων του χρήστη. Διακρίνεται από πέντε διαφορετικές κάρτες επεξεργασίας στοιχείων και όχι μόνο.

- Η πρώτη κάρτα αφορά τα βασικά στοιχεία του χρήστη. Αυτά είναι: ο ΑΔΤ/Διαβατήριο, η πόλη καταγωγής, η χώρα καταγωγής, ο Τ.Κ. (Ταχυδρομικός Κώδικας), το σταθερό, το κινητό και η διεύθυνση κατοικίας.



Εικόνα 34: Καρτέλα επεξεργασίας βασικών στοιχείων

- Η δεύτερη την εισαγωγή η ενημέρωση και η διαγραφή μιας εγγραφής εργασιακής εμπειρίας. Για την ενημέρωση ή διαγραφή κάποιας εγγραφής ο χρήστης μπορεί να επιλέξει την αυτή που επιθυμεί μέσω των tabs που υπάρχουν στην καρτέλα. Τα πεδία είναι: η περίοδος εργασίας, ο τομέας, ο τομέας οργάνωσης, ο τύπος οργάνωσης, ο ρόλος που είχε αναλάβει, η χώρα εργασίας, και η πόλη.



Εικόνα 35: Καρτέλα επεξεργασίας της εργασιακής εμπειρίας

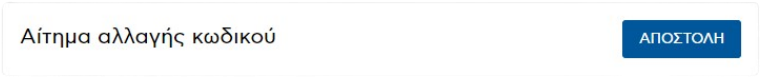
- Η τρίτη την επεξεργασία των social links του αλλά και επιλογή ορατότητας του καθενός ξεχωριστά. Η φόρμα αποτελείται από τα πεδία Προσωπική σελίδα, Linked in, Orcid και Google Scholar.

Εικόνα 36: Καρτέλα επεξεργασίας των social links

- Στην τέταρτη λαμβάνει χώρα η εισαγωγή, η ενημέρωση και η διαγραφή ενός email. Δεν υπάρχει δυνατότητα διαγραφής του βασικού email. Ο χρήστης έχει της δυνατότητα να προσθέσει όσα emails θέλει.

Εικόνα 37: Καρτέλα επεξεργασίας των emails

- Στην πέμπτη και τελευταία υπάρχει η δυνατότητα αποστολής αιτήματος αλλαγής κωδικού μέσω email που θα λάβεις.



Εικόνα 38: Καρτέλα αιτήματος αλλαγής κωδικού

Κεφάλαιο 6ο: Συμπεράσματα και μελλοντικές επεκτάσεις

Η ανάπτυξη της δεύτερης έκδοσης IEEAlumni κατάφερε να πετύχει τους βασικούς στόχους ενίσχυσης και επέκτασης της πρώτης έκδοσης. Τα κύρια οφέλη της πλατφόρμας περιλαμβάνουν την εύκολη διαχείριση δεδομένων από τη γραμματεία λόγω των προσθηκών και των βελτιώσεων, την ενίσχυση της επαγγελματικής δικτύωσης των αποφοίτων μέσω πρόσβασης σε χρήσιμες πληροφορίες και εργαλεία, καθώς και την ανεξαρτησία του back-end και του front-end που επιτρέπει μεγαλύτερη ευελιξία και συντήρηση της εφαρμογής.

Ωστόσο, η ανάπτυξη των δύο αυτών τμημάτων σε διαφορετικές χρονικές περιόδους επέφερε ορισμένες προκλήσεις στη μεταξύ τους συνεργασία. Η έλλειψη ταυτόχρονης ανάπτυξης του back-end και του front-end εμπόδισε την ομαλή ενσωμάτωση και συντονισμό, γεγονός που επηρέασε την απόδοση και την αποτελεσματικότητα του τελικού προϊόντος. Κατά την διάρκεια ενσωμάτωσης του api στο front-end, υπήρχαν προβλήματα που δεν μπορούσαν να προβλεφθούν την περίοδο ανάπτυξης του back-end, καθώς δεν μπορούσαν να γίνουν δοκιμές χωρίς την ύπαρξη ενός λειτουργικού front-end. Αυτά τα προβλήματα βγήκαν στο προσκήνιο εφόσον δημιουργήθηκε η αρχική μορφή της διεπαφής και ξεκίνησαν οι δοκιμές επικοινωνίας με το Api. Σε μερικά δεν βρέθηκε επίλυση ενώ άλλα οδήγησαν σε πολυπλοκότητα λειτουργιών και κώδικα ώστε να δοθεί ένα λειτουργικό τελικό προϊόν. Εν κατακλείδι για την επίλυση των ζητημάτων θα χρειαστεί η από κοινού εργασία σε front-end και back-end καθώς με οποιαδήποτε αλλαγή προκύψει στο ένα, το άλλο θα πρέπει να ενημερωθεί κατάλληλα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Ταουκτσής Βασίλειος, "IEEAlumni: Διαδικτυακή Υπηρεσία για τους Απόφοιτους του Τμήματος ΜΠΗΣ," Thessaloniki, 2023.

[2] Elizabeth Castro, *HTML for the World Wide Web*. Peachpit Press, 2003.

[3] Tim Berners-Lee, *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. Paperback, 2000.

[4] "Next Cube, Εκθέματα Ελληνικό Μουσείο Πληροφορικής." Accessed: May 14, 2024. [Online]. Available: https://elmp.gr/ekthemata/next-cube/

[5] Dave Raggett, Jenny Lam, Ian Alexander, and Michael Kmiec, *Raggett on Html 4*. Addison Wesley Longman, 1998.

[6] Bruce Lawson and Remy Sharp, *Introducing HTML 5*. New Riders, 2011.

[7] "CSS: Cascading Style Sheets." Accessed: May 15, 2024. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/CSS

[8] Eric Meyer and Estelle Weyl, *CSS: The Definitive Guide, 5th Edition*. O'Reilly Media, Inc , 2003.

[9] "Sass: Syntactically Awesome Style Sheets." Accessed: May 16, 2024. [Online]. Available: https://sass-lang.com/

[10] Paul Wilton, *Beginning JavaScript*. John Wiley & Sons, 2004.

[11] "NETSCAPE COMMUNICATIONS CORP", Accessed: May 16, 2024. [Online]. Available: https://www.encyclopedia.com/economics/encyclopedias-almanacs-transcripts-and-maps/netscape-communications-corp

[12] Josh Goldberg, *Learning TypeScript: Enhance Your Web Development Skills Using Type-Safe JavaScript*. O'Reilly Media, Inc, 2022.

[13] Rory Toal, "What is a JavaScript Framework?", ." Accessed: May 18, 2024. [Online]. Available: https://codeinstitute.net/global/blog/javascript-framework/

[14] "Evan You", Accessed: May 18, 2024. [Online]. Available: https://web.archive.org/web/20170603052649/https://betweenthewires.org/2016/11/03/evan-you/

[15] "Vue,js Documentation." Accessed: May 20, 2024. [Online]. Available: https://vuejs.org/

[16] "Vue Router Documentation." Accessed: May 20, 2024. [Online]. Available: https://router.vuejs.org/

[17] "Pinia Documentation." Accessed: May 20, 2024. [Online]. Available: https://pinia.vuejs.org/

[18] "Quasar Documentation." Accessed: May 22, 2024. [Online]. Available: https://quasar.dev/

[19] "Tailwind CSS Documentation." Accessed: May 22, 2024. [Online]. Available: https://tailwindcss.com/

[20] “History of PHP.” Accessed: May 23, 2024. [Online]. Available: https://www.php.net/manual/en/history.php.php

[21] “PHP Version History: Brief Timeline of the World’s Most Used Backend Language”, Accessed: May 22, 2024. [Online]. Available: https://www.cloudways.com/blog/php-version-history/

[22] “History of Laravel.” Accessed: May 24, 2023. [Online]. Available: https://www.javatpoint.com/history-of-laravel

[23] “Axios Documentation.” Accessed: May 24, 2023. [Online]. Available: https://axios-http.com/docs/intro

[24] “dbdiagram.io - Database Relationship Diagrams Design Tool.” Accessed: May 25, 2024. [Online]. Available: https://dbdiagram.io/home

[25] “GitHub Documentation”, Accessed: May 22, 2024. [Online]. Available: https://github.com/

[26] “VS Code Documentation”, Accessed: May 22, 2024. [Online]. Available: https://code.visualstudio.com/

[27] “Swagger Documentation.” Accessed: May 22, 2024. [Online]. Available: https://swagger.io/

[28] “phpMyAdmin Documentation.” Accessed: May 22, 2024. [Online]. Available: https://www.phpmyadmin.net/