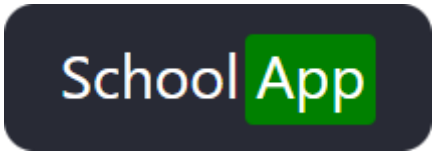
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Διαδικτυακή εφαρμογή διαχείρισης πληροφοριών στη δευτεροβάθμια εκπαίδευση – Μελέτη περίπτωσης»



Του φοιτητή
Θεόδωρου Θεοδωρόπουλου
Αρ. Μητρώου: 154452

Επιβλέπων
Βασίλειος Κώστογλου
Καθηγητής

Ημερομηνία 21-05-2023

Τίτλος Δ.Ε. Διαδικτυακή εφαρμογή διαχείρισης πληροφοριών στη δευτεροβάθμια εκπαίδευση – Μελέτη περίπτωσης

Κωδικός Δ.Ε. 23105

Ονοματεπώνυμο φοιτητή/των Θεοδωρόπουλος Θεόδωρος

Ονοματεπώνυμο εισηγητή Κώστογλου Βασίλειος

Ημερομηνία ανάληψης Δ.Ε. 09-02-2023

Ημερομηνία περάτωσης Δ.Ε. 21-05-2023

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 διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Πρόλογος

Στις μέρες μας η τεχνολογία αποτελεί αναπόσπαστο κομμάτι της καθημερινότητας μας. Καθημερινά χρησιμοποιούμε πολλές εφαρμογές σε διάφορες πλατφόρμες προκειμένου να καλύψουμε τις ανάγκες μας. Η έλευση της πανδημίας ωστόσο ανέδειξε κάποιες δυσκολίες σε ορισμένες υπηρεσίες στις οποίες μέχρι τότε δεν υπήρχε η δυνατότητα χρήσης της τεχνολογίας προκειμένου να διεκπεραιωθούν. Ένα παράδειγμα είναι η δευτεροβάθμια εκπαίδευση όπου πολλές από τις λειτουργίες σταμάτησαν καθώς απαιτούσαν την φυσική παρουσία προσώπων, κάτι το οποίο δεν ήταν εφικτό εκείνη την περίοδο. Η εφαρμογή SchoolApp στοχεύει στην επίλυση κάποιων από αυτών των λειτουργιών προκειμένου να μην χρειάζεται αποκλειστικά η φυσική παρουσία. Η εφαρμογή δίνει τη δυνατότητα πρόσβασης στους καθηγητές αλλά και τους μαθητές ώστε να μπορούν να δουν προσωπικές τους πληροφορίες. Ο ρόλος του καθηγητή παρέχει και την δυνατότητα τροποποίησης δεδομένων ενός μαθητή (π.χ. βαθμολογία μαθήματος) ενώ ο ρόλος του μαθητή έχει αποκλειστικά την δυνατότητα προβολής δεδομένων και όχι της μεταβολής τους. Υπάρχει ένας ακόμα ρόλος τύπου διαχειριστής ο οποίος διαχειρίζεται τις περισσότερες πληροφορίες, καθηγητών και μαθητών καθώς και πληροφορίες που αφορούν το σχολείο (π.χ. ωράριο σχολείου).

Περίληψη

Το αντικείμενο της πτυχιακής εργασίας πραγματεύεται την δημιουργία μιας διαδικτυακής εφαρμογής σχολείου δευτεροβάθμιας εκπαίδευσης. Η εφαρμογή υλοποιήθηκε με την χρήση και τον συνδυασμό αρκετών τεχνολογιών. Για το Back-End επιλέχθηκε το Laravel, ένα framework ανοιχτού κώδικα της γλώσσας προγραμματισμού php. Για το Front-End χρησιμοποιήθηκαν HTML, CSS και JavaScript. Για την δημιουργία της βάσης δεδομένων της εφαρμογής χρησιμοποιήθηκε το MySQL, είναι ένα από τα δημοφιλέστερα, ανοιχτού κώδικα, συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων. Στόχος της εφαρμογής είναι να δώσει τη δυνατότητα στους καθηγητές και τους μαθητές να βρίσκουν της πληροφορίες τους σχετικά με το σχολείο εύκολα, γρήγορα, από οπουδήποτε και οποιαδήποτε στιγμή της ημέρας, ανεξάρτητα από τη συσκευή που διαθέτουν και το λειτουργικό της σύστημα. Απαραίτητη προϋπόθεση είναι μόνο να υποστηρίζει πλοήγηση στο διαδίκτυο μέσω περιηγητών. Απαραίτητη είναι και η ύπαρξη ενός τρίτου χρήστη, τύπου διαχειριστή, ο οποίος θα διαχειρίζεται τις περισσότερες πληροφορίες του συστήματος.

Αρχικά, θα γίνει μια μικρή αναφορά στη μελέτη και την αξιολόγηση παρόμοιων υπαρχόντων εφαρμογών-υπηρεσιών έτσι ώστε στη συνέχεια να αναφερθεί η προστιθέμενη αξία της εφαρμογής. Στη συνέχεια, θα γίνει αναφορά στην προστασία των προσωπικών δεδομένων, την σημασία αυτής, στο GDPR καθώς και στην πολιτική απορρήτου της εφαρμογής MySchool. Έπειτα, θα αναλυθούν οι τεχνολογίες που χρησιμοποιήθηκαν καθώς και η δομή της εφαρμογής. Τέλος, θα γίνει αναφορά στα συμπεράσματα που προέκυψαν καθώς και στη μελλοντική επεκτασιμότητα της εφαρμογής με βελτιώσεις ή/και προσθήκες λειτουργιών που μπορούν να γίνουν.

Web information management application in secondary education – Case study

Theodoros Theodoropoulos

Abstract

The subject of the thesis deals with the creation of a Web application for a secondary school case. The application was implemented using and combining several technologies. Laravel, an open-source web framework of php programming language, was chosen for the Back-End part of the application. HTML, CSS and JavaScript were used for the Front-End part of the application. In order to create the database of the application, MySQL was used, it is one of the most popular open-source relational database management systems. The goal of the application is to allow teachers and students to find their personal school information easily, quickly, from anywhere at any time, regardless of the device they have and its operating system. A prerequisite is that the device supports web browser. It is also necessary to have a third type of user, an administrator type, who will be responsible and will manage most of the system information.

First of all, a quick reference will be made to the study and evaluation of similar existing application-services so that the added value of the application will be mentioned. Subsequently, reference will be made to the protection of personal data, its importance, the GDPR as well as the privacy policy of MySchool application. Afterwards, the technologies that were used as well as the structure of the application will be analyzed. Finally, a reference will be made to the conclusions reached as well as to the future application's scalability with improvements and/or addition functionality that can be added.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κύριο Κώστογλου Βασίλειο για την υπομονή του, την βοήθεια και την καθοδήγησή του. Επιπλέον ένα μεγάλο ευχαριστώ σε όλα τα μέλη της οικογένειάς μου, οι οποίοι στήριξαν την προσπάθεια μου όλα τα χρόνια της φοίτησης και δεν θα είχα φτάσει ως εδώ χωρίς αυτούς.

Περιεχόμενα

Κατάλογος Σχημάτων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
js	JavaScript
ΥΠ.Π.Ε.Θ	Υπουργείο Παιδείας, Έρευνας και Θρησκευμάτων
ΕΠΑ.Λ.	Επαγγελματικό Λύκειο
MVC	Model View Controller
HTTP	Hypertext Transfer Protocol
php	Hypertext Preprocessor
UI/UX	User Interface / User experience
API	Application Programming Interface
ORM	Object Relational Mapper
PDO	PHP Data Objects
id	Identifier
κ.α.	Και άλλα
π.χ.	Παραδείγματος χάριν
CLI	Command Line Interface
DOM	Document Object Model
SPAs	Single Page Applications
AJAX	Asynchronous JavaScript and XML
JSON	JavaScript Object Notation
SQL	Structured Query Language
DBMS	Database Management System
VS Code	Visual Studio Code
XAMPP	X (Cross Platform) A (Apache) M (MySQL) PP (PHP)
CSV	Comma Separated Values
PIPEDA	Personal Information Protection and Electronic Documents
COPPA	Children’s Online Privacy Protection Act

CCPA	California Consumer Privacy Act
CalOPPA	California Online Privacy Protection Act
URI	Uniform Resource Identifier
npm	No Promoter Score
CDN	Content Delivery Network
ms	Millisecond

Κεφάλαιο 1ο: Υπάρχουσες εφαρμογές και προστιθέμενη αξία

1.1 Εισαγωγή

Σκοπός της πτυχιακής εργασίας ήταν να δημιουργηθεί ένα πληροφοριακό σύστημα το οποίο θα δώσει την δυνατότητα στο σχολικό κοινό γενικών λυκείων να αλληλοεπιδρά και να παρέχει πληροφορίες που αφορούν τον εκάστοτε χρήστη, πολλές από τις οποίες ακόμα και σήμερα απαιτούν αποκλείστηκα την φυσική παρουσία προσώπων. Δεν στοχεύει στην αντικατάσταση της παραδοσιακής επικοινωνίας αλλά στην δημιουργία ενός έξτρα εργαλείου επικοινωνίας. Σε αυτό το κεφάλαιο θα αναφερθούν εφαρμογές που αξιοποιούνται από σχολικές μονάδες ή μονάδες εκπαίδευσης γενικότερα, καθώς και η προστιθέμενη αξία της εφαρμογής που αναπτύχθηκε στα πλαίσια της πτυχιακής εργασίας.

1.2 Μελέτη και αξιολόγηση υπαρχόντων εφαρμογών και υπηρεσιών

1.2.1 myschool

Η εφαρμογή myschool είναι ένα ενιαίο πληροφοριακό σύστημα που χρησιμοποιείται από το ΥΠ.Π.Ε.Θ. με στόχο τη μηχανογραφική υποστήριξη των σχολικών μονάδων και των διοικητικών δομών της εκπαίδευσης στην Ελληνική επικράτεια [1]. Ως βασικό αντικείμενο έχει την οργάνωση, διαχείριση, συντήρηση, λειτουργία και την αποθήκευση όλων εκείνων των πληροφοριών και δεδομένων που απαιτούνται για την σωστή και ομαλή λειτουργία μιας σχολικής μονάδας. Είναι μια διαδικτυακή εφαρμογή ωστόσο η είσοδος σε αυτή επιτρέπεται στους αρμόδιους φορείς, δηλαδή το εκάστοτε σχολείο, και τους καθηγητές [2]. Έτσι λοιπόν, υπάρχει μια εφαρμογή με πλήρη αποθήκευση και οργάνωση όλων των δεδομένων και πληροφορίων με αποκλειστικούς αποδέκτες τον φορέα και το προσωπικό. Δεν παρέχεται όμως η δυνατότητα πρόσβασης στη εφαρμογή από τους μαθητές, ώστε να μπορούν να προσπελάσουν προσωπικές πληροφορίες εύκολα και γρήγορα χωρίς να απαιτείται η φυσική τους παρουσία. Συνεπώς, οι μαθητές εξακολουθούν να απευθύνονται σε αρμόδιους καθηγητές προκειμένου να τους δείξουν τις αντίστοιχες πληροφορίες σε εκτυπωμένα έγγραφα ή να τους ενημερώσουν προφορικά.

1.2.2 mySchoolApp

Το mySchoolApp είναι μια εφαρμογή έξυπνων τηλεφώνων, διαθέσιμο για android και iOS, που στοχεύει στην οργάνωση και τη διαχείριση της λειτουργίας και των πληροφοριών σε εκπαιδευτικούς οργανισμούς [3]. Παρέχει πρόσβαση σε αρκετούς διαφορετικούς ρόλους χρηστών όπως καθηγητές, κηδεμόνες και μαθητές που εργάζονται ή φοιτούν αντίστοιχα σε συνεργαζόμενα φροντιστήρια που χρησιμοποιούν την εφαρμογή. Κάθε ένας χρήστης από αυτούς έχει ξεχωριστές δυνατότητες και λειτουργείς με τις οποίες μπορεί να αλληλοεπιδράσει. Επίσης, η νέα λειτουργεία Mobile Live Learning που παρέχει, δίνει την δυνατότητα στους εκπαιδευτικούς να μετατρέψουν την αίθουσα διδασκαλίας σε ψηφιακή αίθουσα, προκειμένου μαθητές που απουσιάζουν να μπορούν να παρακολουθήσουν την διεξαγωγή του μαθήματος εξ’ αποστάσεως. Παρ ’όλες τις δυνατότητες που δίνει στους χρήστες οι κρητικές στο Play Store (Android λειτουργικό σύστημα), δεν είναι οι καλύτερες δυνατές πράγμα που μπορεί να προβληματίσει τον φορέα ή τον χρήστη [4]. Επιπρόσθετα να επισημανθεί, πως δεν

χρησιμοποιείται από γενικά λύκεια της χώρας αλλά από ιδιωτικούς φορείς της δευτεροβάθμιας εκπαίδευσης.

1.2.3 Εκπαιδευτική Διφθέρα

Η Εκπαιδευτική Διφθέρα πρόκειται για μια εφαρμογή που αναπτύχθηκε έχοντας ως στόχο την οργάνωση όλων των δεδομένων των εκπαιδευτικών όπως το προσωπικό εβδομαδιαίο πρόγραμμα, τα μαθήματα, τα σχόλια, οι βαθμολογίες, οι μαθητές, οι απουσίες κ.α. [5]. Απευθύνεται σε φορητές συσκευές μεσαίου και μεγάλου μεγέθους οι οποίες βασίζονται στο λειτουργικό σύστημα Android και μόνο. Επίσης, δίνει την δυνατότητα να χρησιμοποιηθεί ως κεντρική συλλογή των απουσιών μιας σχολικής μονάδας της δευτεροβάθμιας εκπαίδευσης με σκοπό την διευκόλυνση της ενημέρωσης της εφαρμογής myschool, υιοθετώντας την τεχνική διαμοίρασης δεδομένων σε τρίτες εφαρμογές. Ωστόσο η εφαρμογή δεν είναι διαθέσιμη στο play store του Android.

1.3 Προστιθέμενη αξία

Το εύρος των περιπτώσεων χρήσης μιας εφαρμογής διαχείρισης πληροφοριών σε ΓΕΛ της δευτεροβάθμιας εκπαίδευσης μπορεί να χωριστεί σε τρείς άξονες:

1. Περίπτωση χρήσης από τον φορέα
2. Περίπτωση χρήσης από το εκπαιδευτικό προσωπικό
3. Περίπτωση χρήσης από τους μαθητές του σχολείου

Η ανάπτυξη της εφαρμογής που πραγματεύεται η πτυχιακή εργασία, έγινε με γνώμονα αυτούς τους τρείς άξονες και αποκλειστικό στόχο την εξυπηρέτηση των αναγκών των αντίστοιχων χρηστών. Αποτελεί μια διαδικτυακή εφαρμογή προσαρμόσιμου περιεχομένου δίνοντας στον εκάστοτε χρήστη την ευχέρεια να την χρησιμοποιήσει από οποιαδήποτε συσκευή διαθέτει, ανεξάρτητα από το μέγεθος και το λειτουργικό της σύστημα. Τέλος, η εφαρμογή έχει αναπτυχθεί με τέτοιον τρόπο ώστε να παρέχει εξυπηρέτηση σε γενικά λύκεια και δεν υποστηρίζει άλλους τύπους σχολείων της δευτεροβάθμιας εκπαίδευσης όπως είναι τα ΕΠΑ.Λ. Ο συνδυασμός των παραπάνω, κάνουν την εφαρμογή να διαφέρει από τις υπόλοιπες που αναφέρθηκαν στις προηγούμενες ενότητες.

1.4 Επίλογος

Στο κεφάλαιο αυτό έγινε παρουσίαση εφαρμογών που ήδη υπάρχουν και στοχεύουν στην οργάνωση και την διαχείριση πληροφοριών ενός φορέα δευτεροβάθμιας εκπαίδευσης, κάθε μια από τη δική της σκοπιά. Επιπλέον, έγινε αναφορά στα χαρακτηριστικά της εφαρμογής που υλοποιήθηκε στα πλαίσια αυτής της πτυχιακής εργασίας, συνδυαστικά τα οποία προσφέρουν και την προστιθέμενη αξία της εφαρμογής.

Κεφάλαιο 2ο: Προστασία δεδομένων

2.1 Εισαγωγή

Οι περισσότερες εφαρμογές χρησιμοποιούν διάφορα προσωπικά δεδομένα των χρηστών προκειμένου να επιτύχουν τους σκοπούς και τις λειτουργίες τους. Κάθε εφαρμογή όμως, θα πρέπει να σέβεται την ιδιωτικότητά των χρηστών και να προστατεύει τα προσωπικά τους δεδομένα από χρήσεις πέραν της λειτουργικότητας της εφαρμογής. Στο πλαίσιο αυτό έχουν θεσπιστεί νόμοι από κάθε χώρα, αλλά και από θεσμούς όπως η Ευρωπαϊκή Ένωση, οι οποίοι υποχρεώνουν τις εφαρμογές να πληρούν κάποιες προϋποθέσεις για το πώς πρέπει να διαχειρίζονται τα προσωπικά δεδομένα ώστε να αποτραπούν από κακόβουλες χρήσεις ή παράνομη χρήση τους από τρίτες εφαρμογές.

2.2 GDPR

Τα αρχικά GDPR προκύπτουν από τις λέξεις General Data Protection Regulation και στην ουσία αποτελεί ένα σύνολο κανόνων το οποίο έχει αναπτυχθεί από τις ευρωπαϊκές ρυθμιστικές αρχές [6]. Το GDPR δεν είναι το μοναδικό σύνολο κανόνων για την προστασία των δεδομένων, καθώς παγκοσμίως έχουν θεσπιστεί πολλοί παρόμοιοι κανονισμοί όπως PIPEDA (Καναδάς), COPPA, CCPA, CalOPPA (συνδυασμός των τριών στις ΗΠΑ) ωστόσο, κανένας εκ των αναφερθέντων δεν είχε καταφέρει να κάνει την διαφορά μέχρι και την έκδοση του GDPR το οποίο κατάφερε να ξεχωρίσει [7]. Η διάρκεια της δημιουργίας του ήταν περίπου τέσσερα χρόνια και τελικά η πλήρης έγκριση του έγινε στις 14 Απριλίου 2016 [8].

Το σύνολο των απαιτήσεων απορρήτου που απαιτούνται να τηρηθούν βάσει του GDPR είναι πολύ αυστηρό και οι συνέπειες σε περίπτωση μη τήρησης τους, είναι ανάλογες. Για το λόγο αυτό κάθε εταιρεία και οργανισμός θα πρέπει να είναι πολύ προσεκτικοί με αυτό το κομμάτι και θα ήταν ορθό να έχουν προσωπικό εξειδικευμένο για την δουλειά αυτή. Ένα παράδειγμα για να γίνει αντιληπτή η σημασία της τήρησης του συνόλου των κανονισμών του GDPR είναι πως στην περίπτωση μη τήρησης τους οι κυρώσεις οικονομικού τύπου ανέρχονται στα δέκα έως είκοσι εκατομμύρια ευρώ ή από 2% έως 4% του συνολικού τζίρου της εταιρείας [6][8]. Καταλαβαίνουμε λοιπόν και από τις κυρώσεις, πόσο ευαίσθητο και σημαντικό είναι το θέμα της προστασίας των προσωπικών δεδομένων. Είναι πολύ σημαντικό να διευκρινιστεί πως το σύνολο των κανόνων του GDPR δεν απαιτεί την έδρα ενός οργανισμού ή επιχείρησης στην Ευρώπη προκειμένου να υποχρεωθεί και να τους εφαρμόσει. Αντιθέτως, το κριτήριο είναι το κοινό στο οποίο παρέχεται ένα προϊόν ή μια υπηρεσία. Αυτό σημαίνει πως ανεξάρτητα από το που βρίσκεται η έδρα ενός φορέα εάν το προϊόν ή η υπηρεσία που προσφέρει διατίθεται σε ευρωπαίους πολίτες ή ευρωπαϊκές επιχειρήσεις πρέπει να συμμορφώνεται με τα κριτήρια του GDPR [7].

Το GDPR στοχεύει στον τρόπο με τον οποίο οι επιχειρήσεις και οι οργανισμοί θα ελέγχουν και θα επεξεργάζονται όλα τα δεδομένα, κάτω από απόλυτη νομιμότητα και διαφάνεια. Τα δεδομένα μιας εφαρμογής θα πρέπει να χρησιμοποιούνται αυστηρά και μόνο για συγκεκριμένο σκοπό, ο οποίος θα πρέπει να διατίθεται μέσω της εφαρμογής στους χρήστες ώστε να είναι διαρκώς ενήμεροι για το ποιος ακριβώς είναι αυτός ο σκοπός. Επιπλέον, το GDPR προσπαθεί να ερμηνεύσει τον ορισμό των προσωπικών δεδομένων. Ως προσωπικά δεδομένα μπορούν να χαρακτηριστούν πολλές πληροφορίες, κυρίως αυτές μέσω των οποίων γίνεται άμεση ή έμμεση ταυτοποίηση ενός χρήστη όπως

ονοματεπώνυμο, email, αριθμός τηλεφώνου κ.α. Μπορεί επίσης να είναι και πληροφορίες λιγότερο προφανείς όπως είναι η διεύθυνση IP και τα cookies [6].

2.3 Απαιτήσεις πολιτικής απορρήτου

Μια πολιτική απορρήτου μπορεί να διαφέρει από τις υπόλοιπες και αυτό γιατί εξαρτάται άμεσα από την φύση της κάθε εφαρμογής και των δεδομένων που αποθηκεύει και διαχειρίζεται. Ωστόσο, υπάρχουν κάποια βασικά πράγματα που κάθε πολιτική απορρήτου θα πρέπει να περιλαμβάνει όπως [7]:

- Τις πληροφορίες της επιχείρησης όπως, το όνομα, η διεύθυνση, το email και ο αριθμός τηλεφώνου.
- Τον τύπο των δεδομένων που συλλέγονται. Πρέπει ο τύπος των δεδομένων να αναφέρεται λεπτομερώς π.χ. αν πρόκειται για ονοματεπώνυμο, διεύθυνση, τοποθεσία κ.α. Επίσης πρέπει να αναφέρεται που και πως αποθηκεύονται αυτές οι συλλογές δεδομένων.
- Αναφορά στον νόμο που βασίζεται η συλλογή των δεδομένων. Πρέπει να εξηγηθεί στους χρήστες η νομοθεσία η οποία επιτρέπει την συλλογή των συγκεκριμένων δεδομένων.
- Αναφορά για το πως προστατεύονται αυτά τα δεδομένα και ποιες πρακτικές ακολουθούνται προκειμένου να υπάρχει η μέγιστη δυνατή ασφάλεια.
- Αναφορά στη διάρκεια αποθήκευσης και χρήσης των δεδομένων.
- Εκτενείς ανάλυση στο πως ακριβώς χρησιμοποιούνται τα δεδομένα π.χ. ανάλυση δεδομένων, ειδοποιήσεις, σκοπούς μάρκετινγκ κ.α.
- Παράθεση και εκτενείς ανάλυση στους οκτώ διαφορετικούς τύπους δεδομένων υποκείμενων δικαιωμάτων που αναφέρει το GDPR. Η λίστα με τους οκτώ τύπους δεδομένων υποκείμενων δικαιωμάτων περιλαμβάνει τα εξής:

    - Δικαίωμα πρόσβασης
    - Δικαίωμα ενημέρωσης
    - Δικαίωμα διαγραφής
    - Δικαίωμα αντίρρησης
    - Δικαίωμα διόρθωσης
    - Δικαίωμα φορητότητας
    - Δικαίωμα περιορισμού επεξεργασίας
    - Δικαιώματα όσον αφορά την αυτοματοποιημένη επεξεργασία και δημιουργία προφίλ

2.4 Πολιτική απορρήτου SchoolApp

Η εφαρμογή που πραγματεύεται η πτυχιακή εργασία είναι από την φύση της μια εφαρμογή που δημιουργήθηκε για την αποθήκευση και διαχείριση προσωπικών δεδομένων του ανθρώπινου δυναμικού που απαρτίζει μια σχολική μονάδα της δευτεροβάθμιας εκπαίδευσης της Ελλάδας και πιο συγκεκριμένα τα ΓΕΛ. Συνεπώς, η ανάγκη του να ακολουθεί η εφαρμογή τα πρότυπα ασφάλειας και προστασίας των δεδομένων είναι επιτακτική.

Η εφαρμογή υποστηρίζει τρείς τύπους χρηστών εκ των οποίων πολλά από τα δεδομένα είναι ίδια. Αυτοί οι τύποι χρηστών είναι ο διαχειριστής, ο καθηγητής και ο μαθητής. Για κάθε έναν χρήστη καταχωρείται ο τύπος του ώστε να μπορεί η εφαρμογή να παρέχει το αντίστοιχο περιεχόμενο και τις διαθέσιμες ενέργειες που επιτρέπονται. Επίσης, για κάθε χρήστη γίνεται αποθήκευση του ονοματεπωνύμου του, του email του και του κωδικού πρόσβασης. Τα δύο τελευταία είναι και τα δεδομένα που χρησιμοποιούνται προκειμένου ένας χρήστης να εξουσιοδοτηθεί από την εφαρμογή και να έχει πρόσβαση σε αυτή. Επιπλέον, στην οθόνη σύνδεσης δίνεται η δυνατότητα στον χρήστη να επιλέξει εάν

θέλει να τον «θυμάται» η εφαρμογή, στην περίπτωση που το επιλέξει, δημιουργείται ακόμα μια πληροφορία σαν remember_web cookie.

Στους χρήστες τύπου καθηγητές αποθηκεύονται και διαχειρίζονται αρκετές πληροφορίες όπως το ονοματεπώνυμο, email, πόλη και διεύθυνση κατοικίας, αριθμός τηλεφώνου, κωδικός εκπαιδευτικού και τύπος σύμβασης. Αντίστοιχα, για τους χρήστες μαθητές έχουμε τα δεδομένα ονοματεπώνυμο, email, πόλη και διεύθυνση κατοικίας, συνολικές, δικαιολογημένες και αδικαιολόγητες απουσίες, αριθμό τηλεφώνου, σχολικό τμήμα και ανάλογα με τη σχολική χρονιά και ομάδα προσανατολισμού. Όλα αυτά τα δεδομένα μπορούν να δημιουργηθούν, να τροποποιηθούν ή να διαγραφούν αποκλειστικά από τον διαχειριστή μη έχοντας άλλος τύπου χρήστη τέτοια δυνατότητα. Επιπρόσθετα, ο διαχειριστής είναι υπεύθυνος και για δεδομένα που αφορούν την σχολική μονάδα όπως είναι τα σχολικά τμήματα, το ωρολόγιο πρόγραμμά τους και το ωράριο του σχολείου. Οι υπόλοιποι δύο τύποι χρηστών μπορούν απλώς να δούνε τις προσωπικές τους πληροφορίες, με εξαίρεση την δυνατότητα των καθηγητών να εκχωρούν βαθμολογίες, σχόλια και σκαναρισμένες εικόνες των γραπτών των μαθητών για τα αντίστοιχα μαθήματα που διδάσκουν. Η συλλογή όλων αυτών των δεδομένων βασίζεται στο τρόπο οργάνωσης που έχει μια σχολική μονάδα καθώς και στη δυνατότητα να διατηρεί προσωπικά δεδομένα των φυσικών προσώπων. Η χρησιμοποίηση των δεδομένων γίνεται με αποκλειστικό τρόπο την παροχή πληροφοριών που σχετίζονται με την σχολική ζωή για την εξυπηρέτηση αναγκών της σχολικής ζωής.

Τα δεδομένα αυτά είναι δυναμικά και υφίστανται όσο καιρό υφίστανται και τα φυσικά πρόσωπα κυρίως. Αυτό δηλαδή σημαίνει, ότι η διάρκεια ύπαρξης των δεδομένων ενός μαθητή ή καθηγητή θα είναι όσο αυτός βρίσκεται εγγεγραμμένος στα πρακτικά της εκάστοτε σχολικής μονάδας και έχει ενεργό ρόλο. Στη περίπτωση που κάποιος δεν βρίσκεται πλέον στο ανθρώπινο δυναμικό του σχολείου είναι υποχρεωτική η διαγραφή των δεδομένων του. Περίπου το ίδιο ισχύει και με τα δεδομένα που αφορούν την ίδια τη σχολική μονάδα καθώς τα τμήματα δεν είναι κάθε χρονιά σταθερά, ούτε έχουν το ίδιο ωρολόγιο πρόγραμμα ούτε τους ίδιους μαθητές και καθηγητές. Αυτό σημαίνει πως κάθε τέλος ή αρχής μιας σχολικής χρονιάς τα δεδομένα θα πρέπει να ανανεώνονται.

Όλα αυτά τα δεδομένα που χρησιμοποιούνται στο πληροφοριακό σύστημα πρέπει να είναι ασφαλή και προστατευμένα τόσο από χρήστες της εφαρμογής όσο και από τρίτους ενδεχομένως κακόβουλους χρήστες. Ως προς αυτό το ζήτημα, η εφαρμογή παρέχει πολλαπλούς ελέγχους προτού συμβεί η οποιαδήποτε ενέργεια προς τα δεδομένα. Πρώτα από όλα, παρέχεται ένα authentication σύστημα ώστε να μην μπορεί ο οποιοσδήποτε χρήστης να αλληλεπιδρά με την εφαρμογή. Επίσης δεν υπάρχει η δυνατότητα εγγραφής νέου χρήστη αντίθετα η ενέργεια αυτή πραγματοποιείται από εξουσιοδοτημένο χρήστη τύπου διαχειριστή. Από τη στιγμή που κάποιος χρήστης ταυτοποιηθεί και εισέλθει στην εφαρμογή για οποιαδήποτε ενέργεια ζητάει να κάνει γίνεται έλεγχος εάν ο συγκεκριμένος τύπος χρήστης έχει τη δυνατότητα για αυτήν την ενέργεια. Τέλος, εάν όλοι οι παραπάνω έλεγχοι συμβούν με επιτυχία υπάρχουν δικλείδες ασφάλειας ώστε μια ενέργεια να μην αποβεί μοιραία. Για παράδειγμα εάν κάποιος χρήστης επιχειρήσει να τροποποιήσει τα δεδομένα ενός μαθητή και αντί για την πληροφορία που ζητάται, ας πούμε τον αριθμό τηλεφώνου, προσπαθήσει να βάλει κώδικα ο οποίος θα στοχεύει στην βάση δεδομένων για οπουδήποτε ενέργεια, από την περίπτωση να καταστρέψει την βάση μέχρι την περίπτωση να πάρει όλα της τα δεδομένα, αποτρέπεται από τους ελέγχους. Τέλος να αναφερθεί πως δεν χρησιμοποιείται τρίτη εφαρμογή στην οποία να μπορεί ο εκάστοτε χρήστης να δώσει την συγκατάθεσή του για διαμοιρασμό των προσωπικών του δεδομένων.

2.5 Επίλογος

Συνοψίζοντας, γίνεται κατανοητό πόσο σημαντική είναι η προστασία των δεδομένων στις διαδικτυακές και μη εφαρμογές και πόσο ευαίσθητα είναι τα προσωπικά δεδομένα κάθε χρήστη. Επιπλέον, είδαμε πως η Ευρωπαϊκή Ένωση έχει στοχεύσει στο να θωρακίσει αυτό το ευαίσθητο θέμα των προσωπικών δεδομένων, αναπτύσσοντας ένα σύνολο κανόνων ονόματι GDPR, υποχρεώνοντας έτσι κάθε οργανισμό και επιχείρηση να συμμορφώνεται σε αυτούς αλλά και να υπάρχει μια κοινή πλεύση ώστε τα πράγματα να είναι πιο ξεκάθαρα και πιο διαχειρίσμια. Τέλος, έγινε αναφορά στο πως το SchoolApp αποθηκεύει και διαχειρίζεται τα δεδομένα υπακούοντας στους βασικούς κανόνες που πρέπει να έχει μια εφαρμογή βάση της πολιτικής απορρήτου που διαθέτει.

Κεφάλαιο 3ο: Τεχνολογίες που χρησιμοποιήθηκαν

3.1 Εισαγωγή

Το πιο σημαντικό βήμα στο να ξεκινήσει η υλοποίηση της εφαρμογής ήταν η επιλογή των κατάλληλων τεχνολογιών έτσι ώστε η εφαρμογή να έχει την καλύτερη δυνατή απόδοση για τον τελικό χρήστη. Δεδομένου ότι η εφαρμογή στοχεύει στην αποθήκευση και διαχείριση πληροφοριών κρίθηκε απαραίτητη η ύπαρξη ενός server ο οποίος θα διαχειρίζεται όλες αυτές τις πληροφορίες. Η ανάπτυξη της εφαρμογής σε περιβάλλον ιστού ήταν η καλύτερη λύση καθώς η πρόσβαση μπορεί να γίνει από πληθώρα συσκευών ανεξαρτήτως χαρακτηριστικών και λογισμικού. Στο κεφάλαιο αυτό θα γίνει αναφορά και ανάλυση όλων των τεχνολογιών που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής καθώς και των λογισμικών-περιβαλλόντων ανάπτυξής.

3.2 Backend

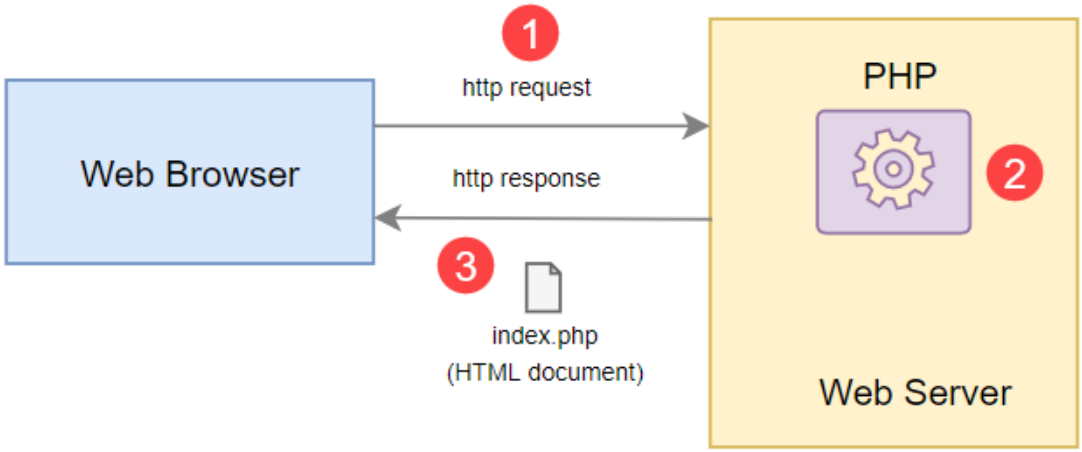
Με τον όρο Backend αναφερόμαστε στο κομμάτι-κώδικα μιας εφαρμογής ο οποίος εκτελείτε στο παρασκήνιο, στον server, και είναι υπεύθυνο για την διαχείριση των δεδομένων και την λογική που θα υπάρχει. Συχνά αναφερόμαστε σε αυτό και με τον όρο data access layer ενώ ο χρήστης δεν έχει πρόσβαση συνεπώς δεν μπορεί να δει ούτε να αλληλεπιδράσει με το συγκεκριμένο κομμάτι. Επιπροσθέτως, περιλαμβάνει την αλληλεπίδραση με την βάση δεδομένων της εφαρμογής στην οποία αποθηκεύονται όλα τα δεδομένα [9][10].

3.2.1 PHP

Η php είναι μια αντικειμενοστραφής server-side scripting γλώσσα προγραμματισμού και χρησιμοποιείται για την ανάπτυξη διαδικτυακών εφαρμογών με δυναμικό περιεχόμενο. Δημιουργήθηκε το 1994 από τον Δανό-Καναδό προγραμματιστή Rasmus Lerdorf και το όνομα προέκυψε αρχικά, από το αρχικό γράμμα των λέξεων Personal Home Page, ωστόσο με την πάροδο του χρόνου αυτό άλλαξε και σήμερα αντιπροσωπεύει το Hypertext Preprocessor [11][12][13].

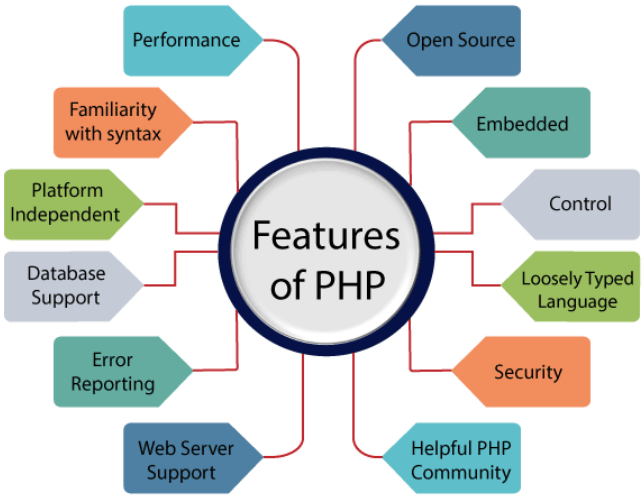
Όπως αναφέρθηκε η php είναι server-side γλώσσα προγραμματισμού το οποίο σημαίνει ότι μπορεί να χρησιμοποιηθεί αποκλειστικά και μόνο στο backend (server-side) μιας διαδικτυακής εφαρμογής με σκοπό να διαχειριστεί HTTP requests και να επιστρέψει δυναμικό περιεχόμενο στον χρήστη. Υπάρχουν δύο βασικοί τύποι γλωσσών προγραμματισμού domain specific και general-purpose. Η php είναι μια γλώσσα general-purpose καθώς μπορεί να συμβάλει στην δημιουργία πολλών και διαφορετικών εφαρμογών. Βασίζεται στην πιο γνωστή προγραμματιστική φιλοσοφία-μεθοδολογία, αυτή της αντικειμενοστρεφιας, η οποία εστιάζει στα δεδομένα-αντικείμενα και είναι αρεστή για εφαρμογές μεγάλες και περίπλοκες ενθαρρύνοντας τις συνεχείς αναβαθμίσεις και την συντήρηση τους. Τα αρχεία που περιέχουν php κώδικα πρέπει να έχουν την κατάληξη '.php' ενώ για να γραφτεί php κώδικας και να είναι έγκυρος πρέπει να περικλείεται μεταξύ '<?php' που σηματοδοτεί την έναρξη και '?>' που σηματοδοτεί το τέλος (εάν ένα αρχείο περιέχει μόνο κώδικα php τότε το δεύτερο 'άγκιστρο' μπορεί και να παραληφθεί). Είναι αξιοσημείωτο το γεγονός ότι μέσα από την php μπορεί να παραχθεί και περιεχόμενο HTML. Ένα ακόμη χαρακτηριστικό της είναι ότι αποτελεί μια cross-platform γλώσσα προγραμματισμού. Με τον όρο αυτό αναφερόμαστε στη δυνατότητα που έχει να υποστηρίζεται από όλα τα λειτουργικά συστήματα όπως Windows, Linux και macOS, γεγονός που αυξάνει το εύρος των

συσκευών που μπορούν να προσπελάσουν την εφαρμογή. Θεμελιώδους σημασίας είναι μια server-side γλώσσα να είναι συμβατή και να λειτουργεί σε servers. Η php μπορεί να χρησιμοποιηθεί από τους περισσότερους web servers, αν όχι από όλους, μερικοί εκ των οποίων είναι Nginx, Apache, OpenBSD ενώ μπορεί να χρησιμοποιηθεί και από cloud servers όπως Microsoft Azure και Amazon AWS. Κύριο γνώρισμα, server-side γλωσσών προγραμματισμού είναι η υποστήριξη που παρέχεται σε βάσεις δεδομένων. Και σε αυτό το κομμάτι η php υποστηρίζει μια μεγάλη λίστα βάσεων δεδομένων μεταξύ των οποίων και οι MySQL, PostgreSQL, MSSQL, db2, Oracle Database και MongoDB. Έχοντας αναφέρει τα χαρακτηριστικά και τις δυνατότητες που έχει η php θα ήταν καλό να αναφερθεί ένα παράδειγμα για να γίνει πιο ξεκάθαρο το πως δουλεύει. Αρχικά, ο περιηγητής (web browser) στέλνει ένα HTTP request στον web server. Στη συνέχεια η php επεξεργάζεται το request και μέσω του κώδικα διαμορφώνει το τελικό HTML document το οποίο και επιστρέφει σαν απάντηση στον περιηγητή [11][12][13][14]. Η εν λόγω διαδικασία φαίνεται και στο παρακάτω σχήμα.



Σχήμα 4.1: Πως λειτουργεί η php

Γιατί να επιλέξω την php ως γλώσσα προγραμματισμού για να διαχειρίζεται το backend της εφαρμογής; Αρχικά είναι απλή και εύκολη στην εκμάθησή της. Είναι σημαντικό για οποιονδήποτε θέλει να ξεκινήσει την ανάπτυξη server-side κώδικα και δεν έχει σχετική εμπειρία, να έχει καθαρή εικόνα για το τι γίνεται. Η php είναι open-source γλώσσα και παρέχει τη δυνατότητα στους προγραμματιστές να χρησιμοποιούν όσα παρέχει δωρεάν. Είναι ευέλικτη γλώσσα εφόσον το περιεχόμενο είναι δυναμικό δεν υπάρχουν πολύ αυστηροί κανόνες για να εφαρμοστούν ενώ ταυτόχρονα παρέχει αρκετά επίπεδα ασφάλειας. Τέλος, η ύπαρξή της τόσα χρόνια, έχει οδηγήσει στην δημιουργία μιας μεγάλης κοινότητας προγραμματιστών, οι οποίοι χρησιμοποιούν την php, μοιράζοντας μέσω αυτής γνώση και βοήθεια [12][13].



Σχήμα 4.2: Χαρακτηριστικά της γλώσσας php

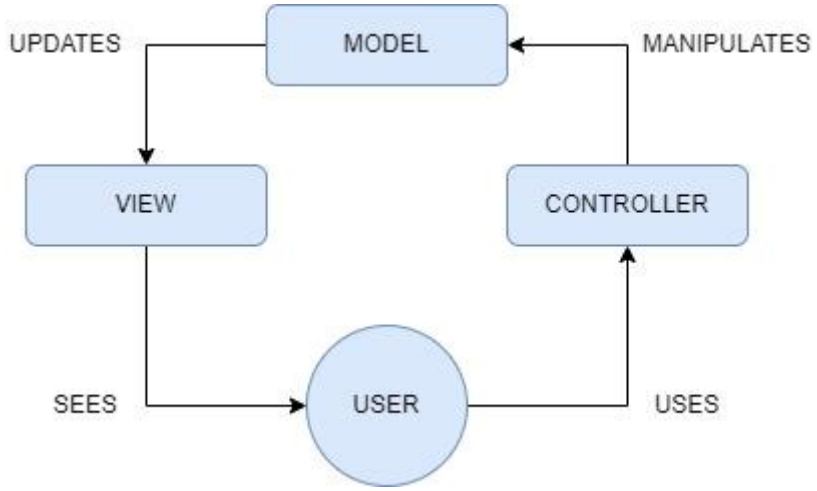
3.2.2 Laravel

Για την ανάπτυξη της διαδικτυακής εφαρμογής χρησιμοποιήθηκε το πλαίσιο προγραμματισμού (web framework) Laravel. Το Laravel είναι ένα δωρεάν web framework της php, ανοιχτού κώδικα και έχει γίνει ένα από τα πιο δημοφιλή php framework παγκοσμίως. Δημιουργήθηκε από τον Taylor Otwell το 2011 και παρέχει ένα ισχυρό και ευέλικτο σύνολο εργαλείων για τη δημιουργία σύγχρονων και επεκτάσιμων εφαρμογών διαδικτύου [15]. Έχει χτιστεί πάνω στην γλώσσα προγραμματισμού php και ένα από τα βασικά χαρακτηριστικά του είναι η ιδιαίτερη και κομψή σύνταξή που έχει κάτι το οποίο βοηθάει πολύ στην συγγραφή καθαρού και ευανάγνωστου κώδικα. Το Laravel βασίζεται σε ένα από τα πιο συχνά χρησιμοποιούμενα αρχιτεκτονικά μοτίβα ανάπτυξης διαδικτυακών εφαρμογών, προκειμένου οι εφαρμογές να είναι ευέλικτες και επεκτάσιμες, το MVC (Model – View - Controller). Το μοτίβο MVC διαχωρίζει την εφαρμογή σε τρία βασικά μέρη (components) στα μοντέλα (model), τις προβολές (view) και τους διαχειριστές (controller) [17]. Κάθε ένα από αυτά έχει σχεδιαστεί για να διαχειρίζεται συγκεκριμένες πτυχές σε κάθε εφαρμογή.

Σχήμα 4.3: Το λογότυπο του πλαισίου προγραμματισμού Laravel

- Model
  Το μοντέλο είναι στην πραγματικότητα μια κλάση η οποία χρησιμοποιείται για να αλληλοεπιδρά με την βάση δεδομένων. Κάθε κλάση-μοντέλο αντιστοιχεί σε έναν μοναδικό πίνακα της βάσης. Μέσω αυτού δίνεται η δυνατότητα αλληλεπίδρασης με τα δεδομένα της εφαρμογής.
- View
  Είναι στις περισσότερες περιπτώσεις ένα αρχείο με περιεχόμενο HTML, υπεύθυνο για την εμφάνιση περιεχομένου στην οθόνη του χρήστη (UI/UX). Στην περίπτωση των views το Laravel παρέχει ένα template engine, το Blade, ένα πολύ ισχυρό εργαλείο για την διαχείριση του δυναμικού περιεχομένου που επιστρέφει ο server στον περιηγητή, θα αναλυθεί στην συνέχεια. Επίσης, στα αρχεία των views, μπορούν να περιλαμβάνεται κώδικας CSS για την εμφάνιση των στοιχείων αλλά και JavaScript για την λειτουργικότητά τους.
- Controller
  Είναι μια κλάση η οποία αποτελείται από ένα σύνολο μεθόδων υπεύθυνων για την λογική και την λειτουργικότητα της εφαρμογής. Πρακτικά, κάθε μέθοδος μιας κλάσης controller αντιστοιχεί σε μια συγκεκριμένη διαδρομή (route) της εφαρμογής και ανάλογα με την διαδρομή πυροδοτείτε η κατάλληλη δράση από τον διαχειριστή (controller). Τα routes θα αναλυθούν στην συνέχεια.

Αυτά τα τρία βασικά μέρη του αρχιτεκτονικού μοτίβου MVC, το Laravel τα οργανώνει και τα τοποθετεί με ιδιαίτερα προσεκτικό και οργανωμένο τρόπο ώστε να είναι κατανοητά, απλά και έτοιμα για χρήση από τον προγραμματιστή.



Σχήμα 4.4: Η έννοια του μοντέλου MVC

Ένα ακόμη σημαντικό χαρακτηριστικό του Laravel Framework, είναι η εύκολη και ευέλικτη διαχείριση που παρέχει για τα Routes ή URLs ή διαδρομές καθώς, δίνει δύο διαφορετικούς τρόπους για την δημιουργία τους [16]. Ο πρώτος τρόπος είναι με την χρήση των web routes [18]. Στην ουσία το Laravel παρέχει ένα αρχείο στο οποίο μπορούν να ορισθούν όλες οι διαδρομές για όλα τα web interfaces της εφαρμογής διασαφηνίζοντας τον τύπο του κάθε request (π.χ. POST, GET), τον controller και την μέθοδο

από τον συγκεκριμένο controller που θα το διαχειριστεί κατάλληλα. Μας δίνει επίσης την δυνατότητα να ονομάζουμε τα web routes ώστε να είναι πιο εύκολη και καθαρή η διαχείριση και η χρησιμοποίηση τους σε όλη την εφαρμογή. Ένα παράδειγμα web route είναι το εξής:

Route::get('/users', [UsersController::class, 'index'])->name('users.list);

Το παράδειγμα μας λέει το request θα είναι τύπου GET ακολουθώντας την διαδρομή '/users' και θα το διαχειριστή η μέθοδος 'index' η οποία ανήκει στον controller 'UsersController'. Στο τέλος χρησιμοποιείται η μέθοδος name() με την οποία παίρνει ένα μοναδικό όνομα η συγκεκριμένη διαδρομή. Η ονομασία της διαδρομής δεν είναι υποχρεωτική και μπορεί να παραλειφθεί αλλά έχοντας όνομα μπορούμε να τη διαχειριστούμε καλύτερα από κάθε σημείο μέσα στην εφαρμογή. Να επισημανθεί πως το όνομα πρέπει να είναι μοναδικό. Ο δεύτερος τρόπος με τον οποίο μπορούμε να διαχειριστούμε τις διαδρομές είναι μέσο των API routes [18]. Τα API routes βασίζονται στην ίδια λογική με τα web routes διαφέρουν όμως στον πυρήνα τους. Ειδοποιός διαφορά είναι ότι ο κάθε τύπος διαδρομών έχει δικιά του ομάδα middleware την οποία και χρησιμοποιεί. Στην περίπτωση των API routes υπάρχει και ένα προκαθορισμένο prefix στην διαδρομή με τιμή api. Τελευταία διαφορά είναι η ύπαρξη ενός rate limit στα api routes στοχεύοντάς στον περιορισμό του χρήστη. Φυσικά η τιμή αυτή είναι μεταβλητή και όχι σταθερή. Δεν υπάρχει κανόνας ή περιορισμός ως προς την επιλογή μιας εκ των δύο μεθόδων που θα χρησιμοποιηθεί, ωστόσο συνηθίζεται να χρησιμοποιούνται τα web routes για την επιστροφή των views και τα api routes για την επιστροφή json (api resource/collection). Ορίζοντας μια διαδρομή είναι απαραίτητο μεταξύ άλλων να ορίσουμε τον τύπο της μεθόδου του HTTP αιτήματος. Παρακάτω, παρουσιάζεται μια λίστα με τις μεθόδους που χρησιμοποιούνται πιο συχνά [19]:

- GET
  Η μέθοδος GET ζητάει μια αναπαράσταση ενός συγκεκριμένου πόρου. Τα requests αυτής της μεθόδου πρέπει μόνο να ανακτούν δεδομένα. Οι λεπτομέρειες του συγκεκριμένου πόρου βρίσκονται στο σώμα του μηνύματος.

- POST
  Η μέθοδος POST υποβάλει-δημιουργεί μια οντότητα σε έναν συγκεκριμένο πόρο προκαλώντας αλλαγές στον server. Και σε αυτή τη μέθοδο οι λεπτομέρειες του συγκεκριμένου πόρου βρίσκονται στο σώμα του μηνύματος.

- HEAD
  Η μέθοδος HEAD είναι παρόμοια με την μέθοδο GET με την διαφορά ότι μεταφέρει μόνο την γραμμή κατάστασης και τις κεφαλίδες (headers) του HTTP response. Δεν περιλαμβάνεται σώμα μηνύματος.

- PUT
  Η μέθοδος PUT χρησιμοποιείται για να αντικαταστήσει έναν συγκεκριμένο πόρο και όλες αναφορές του. Στο σώμα του HTTP request βρίσκεται η πληροφορία για τον πόρο που θα ενημερωθεί.

- DELETE
  Η μέθοδος DELETE καλείται από τον πελάτη για να διαγράψει έναν συγκεκριμένο πόρο και τις αναπαραστάσεις του.

- CONNECT
  Η μέθοδος CONNECT καλείται από τον πελάτη για να δημιουργήσει μια σήραγγα προς τον server αναγνωρισθείς από το αίτημα-στόχο.

- TRACE
  Η μέθοδος TRACE καλείται από τον πελάτη για να εκτελέσει μια loopback δοκιμή στη διαδρομή ενός συγκεκριμένου πόρου.
- OPTIONS
  Η μέθοδος OPTIONS καλείται από τον πελάτη για να μάθει ποιες λειτουργίες είναι διαθέσιμες για έναν συγκεκριμένο πόρο ή τον server γενικά.

Στην προηγούμενη παράγραφο αναφέρθηκε η έννοια του middleware χωρίς να εξηγηθεί. Το Laravel προσφέρει έναν μηχανισμό φιλτραρίσματος με στόχο να αποκλείει τους χρήστες που δεν έχουν δικαιώματα για τις ενέργειες που ζητούν, αυτός ο μηχανισμός ονομάζεται middleware [20]. Αυτή η λειτουργία είναι πολύ σημαντική καθώς προστατεύει την εφαρμογή από κινδύνους και πιθανόν κακόβουλους χρήστες. Συμπεριφέρεται σαν γέφυρα μεταξύ των responses και των requests αποκλείοντας τους μη εξουσιοδοτημένους χρήστες [21][22].

Κοινό χαρακτηριστικό πολλών διαδικτυακών εφαρμογών, και όχι μόνο, είναι το σύστημα της ταυτοποίησης των χρηστών προκειμένου να αλληλοεπιδράσουν με το περιεχόμενο και τα δεδομένα της εφαρμογής. Η δημιουργία ενός τέτοιου συστήματος ταυτοποίησης εξ ’ολοκλήρου από την αρχή μπορεί να είναι αρκετά επώδυνη και περίπλοκη διαδικασία. Το πιο σοβαρό ζήτημα όμως έγκειται στα θέματα ασφαλείας καθώς μια τέτοια υλοποίηση ελλοχεύει υψηλό ρίσκο κινδύνου. Με στόχο να αποφευχθεί αυτή η επίπονη διαδικασία που ενδεχομένως να οδηγήσει και σε σοβαρά ζητήματα ασφάλειας το Laravel παρέχει ένα εργαλείο το οποίο απλοποιεί αυτήν την διαδικασία και παράγει ένα σύστημα ταυτοποίησης γρήγορα, εύκολα και με ασφάλεια. Για να αξιοποιηθεί αυτό το εργαλείο πρέπει να εγκατασταθεί μέσα από την γραμμή εντολών με όνομα artisan, που παρέχει το Laravel. Στον πυρήνα του το Laravel αποτελείται από φρουρούς (guards) και παρόχους (providers) [23]. Οι φρουροί είναι αυτοί που καθορίζουν τον έλεγχο των χρηστών σε κάθε request που πραγματοποιούν προς τον server. Από την άλλη, οι πάροχοι ορίζουν το πως οι χρήστες θα ανακτήσουν τα δεδομένα από την βάση δεδομένων. Το Laravel υποστηρίζει την διαδικασία με δύο τρόπους, χρησιμοποιώντας είτε το Eloquent, είτε το εργαλείο δημιουργίας ερωτημάτων βάσης δεδομένων (database query builder) [24], δύο πάροχοι αρκετά σημαντικοί οι οποίοι θα αναφερθούν και αναλυθούν στη συνέχεια. Παρότι το Laravel παρέχει τους δικούς του παρόχους δεν περιορίζει την χρήση κάποιου άλλου επιπρόσθετου και δίνει την δυνατότητα χρήσης οποιουδήποτε ταιριάζει στις ανάγκες της εφαρμογής.

Θεμελιώδους σημασίας λειτουργικότητα για ένα διαδικτυακό πλαίσιο προγραμματισμού είναι η δυνατότητα να επικοινωνεί με την βάση δεδομένων έτσι ώστε να διαχειρίζεται και να τροποποιεί-επεξεργάζεται τα δεδομένα. Το Laravel περιλαμβάνει από τον πυρήνα της, όπως αναφέρθηκε και πιο πάνω, δύο τρόπους επικοινωνίας και αλληλεπίδρασης με την βάση δεδομένων, το Eloquent και το εργαλείο δημιουργίας ερωτημάτων βάσης δεδομένων. Ας δούμε λίγο πιο αναλυτικά αυτές τις δύο τεχνικές. Αρχίζοντας από το Eloquent, είναι ένας αντικειμενοσχεσιακός χάρτης (ORM) ο οποίος απλοποιεί την διαδικασία και την κάνει εύκολη και ευχάριστη. Για κάθε πίνακα στη βάση δεδομένων αντιστοιχεί ένα συγκεκριμένο μοντέλο μέσω του οποίου επιτυγχάνεται η αλληλεπίδραση με τα δεδομένα [25]. Το μοντέλο αυτό είναι υποχρεωτικό ώστε να επιτευχθεί η αλληλεπίδραση. Επιτρέπει την εισαγωγή, την ενημέρωση και την διαγραφή δεδομένων. Το γεγονός ότι ακολουθεί ένα αντικειμενοσχεσιακό τρόπο διαχείρισης των δεδομένων και τα αναπαριστά ως αντικείμενα διευκολύνει τον προγραμματιστή ο οποίος θα πρέπει να είναι εξοικειωμένος εφόσον η php στην οποία έχει χτιστεί το Laravel, είναι αντικειμενοστραφής γλώσσα προγραμματισμού. Ακολουθεί παράδειγμα κώδικα.

```
use App\Models\Users;
// …
$user = Users::where('id', userId)->firstOrFail();
// …
```

Σύνταξη ερωτήματος βάσης δεδομένων με Eloquent

Όπως φαίνεται στο παράδειγμα στόχος είναι να βρεθεί ένας χρήστης με συγκεκριμένο id. Το Users είναι η αναφορά στο μοντέλο που αντιπροσωπεύει τον αντίστοιχο πίνακα στην βάση δεδομένων και για να χρησιμοποιηθεί θα πρέπει να το εισάγουμε στο αντίστοιχο controller αρχείο με την μέθοδο use. Προκειμένου να βρούμε τον χρήστη υπάρχει η δυνατότητα χρήσης της μεθόδου where() η οποία δέχεται σαν πρώτη παράμετρο το όνομα της στήλης του πίνακα στην οποία θα ψάξει για την εγγραφή και σαν δεύτερη παράμετρο την τιμή που θα ψάξει στην συγκεκριμένη στήλη. Πάντα μετά την μέθοδο που αποσκοπεί στην διαχείριση των δεδομένων ακολουθείται μια μέθοδος, στο παράδειγμά είναι η firstOrFail(), ώστε να πάρουμε τα δεδομένα που έχουν βρεθεί στη βάση, εάν υπάρχουν, να τα αποθηκεύσουμε στην εκάστοτε μεταβλητή και να τα διαχειριστούμε μέσα στην εφαρμογή.

Από την άλλη υπάρχει και η τεχνική δημιουργίας ερωτημάτων βάσης δεδομένων, γνωστή ως Database Query Builder. Η τεχνική αυτή παρέχει μια βολική, εύχρηστη και εύκολη διεπαφή για την δημιουργία και την εκτέλεση σχεδόν όλων των ερωτημάτων που δέχεται μια βάση δεδομένων και υποστηρίζεται από όλες τις βάσεις που το Laravel προσφέρει τη δυνατότητα σύνδεσης [26]. Πολύ σημαντικό προτέρημα της αυτής της τεχνικής είναι ότι κάνει δέσμευση παραμέτρων PDO (PDO parameter binding) στοχεύοντας στην προστασία της εφαρμογής από κακόβουλες επιθέσεις SQL ή όπως είναι η ορολογία στην βιομηχανία SQL injection attacks. Γίνεται σαφές το γεγονός, ότι εκτός από ευκολία και εξοικονόμηση χρόνου για αλληλεπίδραση με την βάση δεδομένων, το Laravel παρέχει και την απαιτούμενη ασφάλεια. Ακολουθεί παράδειγμα κώδικα.

```
use Illuminate\Support\Facades\DB;
// …
$user = DB::table('users')->where('id', $id)->get();
// …
```

Σύνταξη ερωτήματος βάσης δεδομένων με Database Query Builder

Όπως βλέπουμε, το ερώτημα του παραδείγματος μοιάζει με το ερώτημα Eloquent. Η βασική διαφορά είναι πως δεν χρειάζεται κάποιο μοντέλο προκειμένου να γίνει η αναφορά στον αντίστοιχο πίνακα της βάσης. Αντ' αυτού αξιοποιείται η κλάση DB την οποία πρέπει να εισάγουμε στο controller αρχείο με την use μέθοδο και πάλι. Η κλάση αυτή προέρχεται από τον πυρήνα του Laravel, και χρησιμοποιώντας την μέθοδο table() γίνεται η αναφορά στον κατάλληλο πίνακα της βάσης δεδομένων. Η υπόλοιπη λογική είναι παρόμοια με την τεχνική Eloquent καθώς χρησιμοποιείται μια μέθοδος για να γίνει το ερώτημα,

στο παράδειγμα η where(), με παραμέτρους το όνομα της στήλης του πίνακα και την τιμή που θα ψάξει να βρει στη στήλη. Και πάλι θα χρειαστεί στο τέλος μια μέθοδος για να πάρουμε το αποτέλεσμα και να το χρησιμοποιήσουμε, στο παράδειγμα η get(). Να σημειωθεί πως δεν υπάρχει κανόνας για το πότε πρέπει να επιλέγεται η κάθε τεχνική και είναι στην ευχέρεια του εκάστοτε προγραμματιστεί ποια από τις δύο θα επιλέξει ή και τις δυο. Η Database Query Builder έχει καλύτερη και ταχύτερη απόδοση σε επίπεδο χιλιοστών του δευτερολέπτου και αυτός είναι ο λόγος που συνήθως επιλέγεται όταν ο όγκος των δεδομένων είναι πολύ μεγάλος.

Μια ακόμα πολύ σημαντική λειτουργικότητα που δίνει το Laravel είναι τα Migrations της βάσης δεδομένων. Πρόκειται για μια λειτουργία που δίνει τη δυνατότητα ελέγχου ως προς το σχήμα-έκδοση της βάσης δεδομένων αλλά και τροποποίησης της μέσα από προγραμματιστικό περιβάλλον [27]. Το σκεπτικό πίσω από αυτήν την λειτουργικότητα είναι να απλοποιήσει την διαδικασία εποπτείας και διαχείρισης της βάσης δεδομένων όχι απλώς μεμονωμένα για ένα άτομο αλλά και μεταξύ μιας ομάδας προγραμματιστών. Δημιουργία ενός καινούργιου πίνακα, προσθήκη-αφαίρεση-τροποποίηση μιας στήλης ενός πίνακα είναι μερικές από τις δυνατές ενέργειες χρησιμοποιώντας τα migrations. Πώς όμως κάτι τέτοιο μπορεί να βοηθήσει μια ομάδα προγραμματιστών; Η απάντηση είναι αρκετά εύκολη, τα migrations είναι κλάσεις οι οποίες δημιουργούνται και αποθηκεύονται στον αντίστοιχο φάκελο μέσα στην εφαρμογή, τοποθετούνται με χρονολογική σειρά, από το πιο παλαιό προς το πιο καινούργιο και το κάθε αρχείο περιέχει πληροφορίες για μια συγκεκριμένη ενέργεια. Με αυτό το τρόπο, όλο το σχήμα της βάσεις βρίσκεται με οργανωμένο και καθαρό τρόπο εντός της εφαρμογής δίνοντας στον προγραμματιστή μια καθαρή εικόνα για οτιδήποτε σχετικό χρειαστεί από τη βάση δεδομένων χωρίς να χρειάζεται να μεταφερθεί στην βάση αυτή καθαυτή. Και σε αυτή την περίπτωση είναι ξεκάθαρο λοιπόν, το πόσο πολύτιμη είναι η συμβολή του Laravel στο να εξοικονομηθεί πολύτιμος χρόνος στην ανάπτυξη μιας εφαρμογής ενώ ταυτόχρονα δεν κάνει καμία έκπτωση στην ποιότητα.

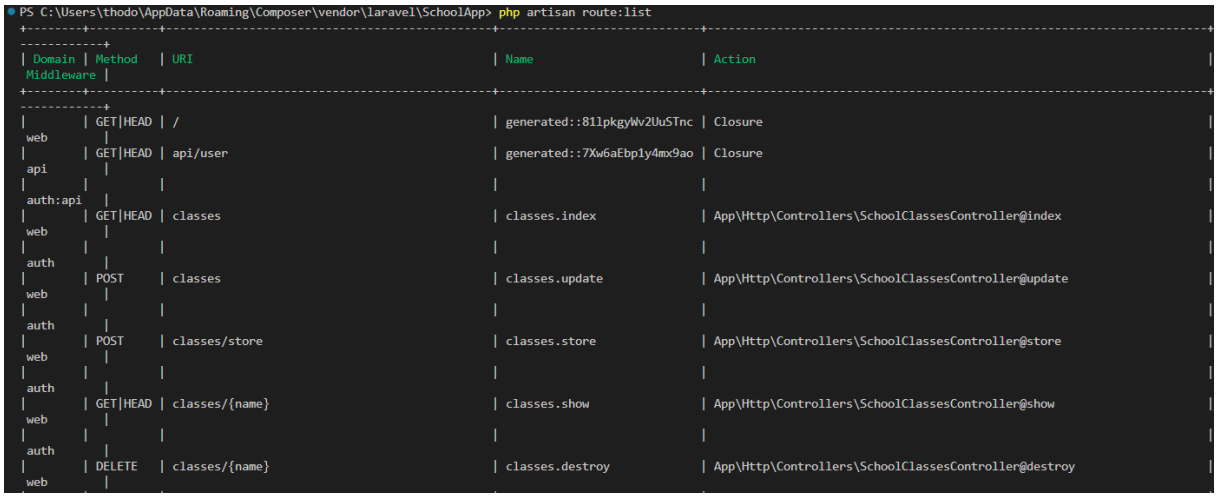
Καθώς έγινε η ανάλυση του αρχιτεκτονικού μοτίβου MVC πιο πάνω, στην περιγραφή της οντότητας View αναφέρθηκε ότι το Laravel παρέχει ένα template engine στον πυρήνα του που ονομάζεται Blade. Ένα template engine είναι ένας μηχανισμός ο οποίος μετατρέπει το δυναμικό περιεχόμενο που επιστρέφει ο server σαν απάντηση στο αίτημα του περιηγητή, σε αρχείο με περιεχόμενο HTML έτσι ώστε ο περιηγητής να εμφανίσει το αντίστοιχο αποτέλεσμα. Μερικά από τα πιο γνωστά php template engines είναι τα εξής: Twig, Blade, Smarty, Dwoo, Volt, Plates και Mustache. Τα αρχεία blade υποστηρίζουν την συγγραφή κώδικα HTML και μετατρέπονται σε php κώδικα ώστε να αποθηκευτούν προσωρινά μέχρι να τροποποιηθούν στο τελικό αποτέλεσμα χωρίς να προσθέτουν επιβάρυνση στην εφαρμογή. Από αυτό προκύπτει πως το blade παρέχει την δυνατότητα χρήσης της γλώσσας php σε αντίθεση με άλλα template engine που δεν το υποστηρίζουν. Επιπροσθέτως, το blade παρέχει δικές του δομές για κώδικα όπως συνθήκες όρου και λούπες. Ακόμα ένα πλεονέκτημα που παρέχει, είναι η δυνατότητα να ‘κληρονομηθεί’ από άλλα αρχεία στηρίζοντας έτσι την λογική των components (θα αναφερθεί η τεχνική στην ενότητα HTML). Τα αρχεία blade πρέπει υποχρεωτικά να έχουν την κατάληξη ‘name.blade.php’ [28].

Το Laravel πέρα από τις λειτουργείες που αναφέρθηκαν παρέχει και μια δικά του διεπαφή γραμμής εντολών ή αλλιώς command line interface, ονόματι Artisan. Είναι ένα περιβάλλον επεξεργασίας και χρησιμοποιεί εντολές για να αλληλοεπιδράσει με τον υπολογιστή και την εφαρμογή πιο συγκεκριμένα [29]. Το Artisan CLI μοιάζει πολύ με το command line του λειτουργικού συστήματος Windows καθώς και με το Linux command το αντίστοιχο στο λειτουργικό σύστημα Linux, με την διαφορά όμως να είναι αισθητή καθώς πέρα από τις όποιες ομοιότητες, είναι προσαρμοσμένο με τέτοιο τρόπο ώστε να συνεισφέρει τα μέγιστα στην υλοποίηση της εφαρμογής χωρίς να μπορεί να χρησιμοποιηθεί εκτός αυτής. Όλες οι εντολές στο Artisan πρέπει να ακολουθούν συγκεκριμένη μορφή «php aertisan ‘όνομα

εντολής'» [30]. Μέσω του Artisan μπορούμε να δημιουργήσουμε σχεδόν τα πάντα για την εφαρμογή. Τα models, οι controllers και τα migrations δεν είναι κάτι άλλο παρά αρχεία με κλάσεις το κάθε ένα υπεύθυνο για κάτι διαφορετικό μέσα στην εφαρμογή. Αντί λοιπόν να δημιουργούμε τα αρχεία ένα-ένα χειροκίνητα χρησιμοποιούμε το Artisan το οποίο τα δημιουργεί για εμάς τα τοποθετεί σε συγκεκριμένους φακέλους, ανάλογα με τον τύπο του κάθε αρχείου αλλά προσθέτει και προκαθορισμένες πληροφορίες όπως το όνομα της κλάσης, ονόματα μεθόδων και σχόλια για καθοδήγηση. Φυσικά όλες αυτές οι πληροφορίες μπορούν να τροποποιηθούν όποτε αυτό κρίνεται απαραίτητο χωρίς να υπάρχει κάποιος περιορισμός. Εξάλλου, υπάρχει για να εξυπηρετήσει και όχι να εμποδίσει. Ενδεικτικές εντολές για την δημιουργία τέτοιων οντοτήτων είναι οι εξής:

- php artisan make:model 'Όνομα μοντέλου'
- php artisan make:controller 'Όνομα του controller'
- php artisan make:migration "Όνομα του migration"
- php artisan make:auth -> Δημιουργία συστήματος ταυτοποίησης χρηστών

Επιπλέον, από το artisan μπορούμε να εποπτεύουμε τις διαδρομές της εφαρμογής οι οποίες είναι υπεύθυνες για το που θα κατευθύνουν τον χρήστη αλλά και να τις ανανεώνουμε όταν υπάρξει κάποια τροποποίηση, προσθήκη ή διαγραφή. Ιδιαίτερα σημαντική η ανανέωση των διαδρομών σε περίπτωση μεταβολής του αντίστοιχου αρχείου και γίνεται μέσα από την ανανέωση της μνήμης cache. Η cache μνήμη μπορεί να προκαλέσει το πρόβλημα και παρότι οι διαδρομές έχουν τροποποιηθεί με επιτυχία να μην μπορεί η εφαρμογή να εντοπίσει τις αντίστοιχες αλλαγές.



Σχήμα 4.5: Artisan, ενδεικτική εντολή προβολής διαδρομών εφαρμογής

3.3 Βάση δεδομένων

Ως βάση δεδομένων ορίζουμε μια οργανωμένη συλλογή δεδομένων ή πληροφοριών. Μια βάση δεδομένων ελέγχεται μέσω ενός συστήματος διαχείρισης βάσεων δεδομένων, Database Management System. Συνήθως αναφερόμαστε στον συνδυασμό των δεδομένων και του DBMS ως βάση δεδομένων [31]. Τα δεδομένα μπορούν να είναι προσβάσιμα και διαχειρίσιμα χρησιμοποιώντας την γλώσσα SQL (Structured Query Language). Η SQL πρόκειται για μια γλώσσα προγραμματισμού η οποία

χρησιμοποιείται από όλες σχεδόν τις σχεσιακές βάσεις δεδομένων προκειμένου να αλληλεπιδράσει με τα δεδομένα.

3.3.1 DBMS

Μια βάση δεδομένων απαιτεί ένα σύστημα διαχείρισης βάσεων δεδομένων (DBMS), ώστε να παρέχει την κατάλληλη διεπαφή μεταξύ της βάσης και του χρήστη, επιτρέποντας την διαχείριση των πληροφοριών. Επίσης, κάνει πιο εύκολη την επίβλεψη και τον έλεγχο των βάσεων δεδομένων επιτρέποντας πληθώρα λειτουργιών όπως παρακολούθηση απόδοσης, συντονισμός, δημιουργία αντιγράφων ασφαλείας και ανάκτηση δεδομένων [31]. Μερικά από τα δημοφιλή DBMS είναι MySQL, Oracle Database, PostgreSQL, Microsoft SQL Server και dBase. Για την υλοποίηση της πτυχιακής επιλέχθηκε το MySQL.

3.3.2 MySQL

Το MySQL αποτελεί το πιο διάσημο σύστημα διαχείρισης βάσεων δεδομένων. Είναι λογισμικό διαχείρισης σχεσιακών βάσεων δεδομένων (Relational DBMS) και είναι ανοιχτού κώδικα [32]. Σχεδιάστηκε για εφαρμογές διαδικτύου και με την πάροδο του χρόνου εξελίχθηκε σε μια από τις πιο ταιριαστές τεχνολογίες για τους προγραμματιστές που αναπτύσσουν εφαρμογές διαδικτύου. Με τον όρο σχεσιακή εννοούμε ότι τα δεδομένα σε μια βάση, τέτοιου τύπου, σχετίζονται μεταξύ τους και η μορφή με την οποία αναπαρίστανται είναι η μορφή πίνακα. Κάθε στοιχείο του πίνακα έχει ένα μοναδικό id που ονομάζεται κύριο κλειδί. Το MySQL υποστηρίζει όλα τα λειτουργικά συστήματα σε πολλές γλώσσες όμως, με την php γλώσσα προγραμματισμού έχει μια «ιδιαίτερη» σχέση. Είναι πιο γρήγορη βάση δεδομένων σε σύγκριση με άλλες κάτι που είναι πολύ σημαντικό όταν ο όγκος των δεδομένων είναι πολύ μεγάλος. Και επειδή θίξαμε το θέμα του μεγέθους της βάσης δεδομένων είναι σημαντικό να αναφερθεί πως το MySQL επιτρέπει την δημιουργία τεράστιων πινάκων, της τάξης των πενήντα εκατομμυρίων γραμμών ή και περισσότερων. Θεωρητικά το μέγεθος του αρχείου ενός πίνακα μπορεί να φτάσει έως 8 εκατομμύρια TB, όμως χρειάζεται ιδιαίτερη προσοχή και έλεγχο εάν το εκάστοτε λειτουργικό σύστημα μπορεί να διαχειριστεί ένα τόσο μεγάλο αρχείο. Μερικές από τις κορυφαίες εφαρμογές που χρησιμοποιούν το MySQL είναι: LinkedIn, YouTube, Facebook Uber και Airbnb [31].

3.4 Frontend

Με τον όρο Frontend αναφερόμαστε στο κομμάτι-κώδικα μιας εφαρμογής ο οποίος εκτελείτε στο προσκήνιο, στην οθόνη της συσκευής, και είναι υπεύθυνος για την παρουσίαση της εφαρμογής και την λειτουργικότητα που θα παρέχει στον χρήστη προκειμένου να αλληλοεπιδρά μαζί της [33]. Το frontend σχετίζεται με όλα όσα ο χρήστης βλέπει και αλληλοεπιδρά και για το λόγο αυτό είναι πολύ σημαντικό να είναι καλά σχεδιασμένο, οργανωμένο και ελκυστικό εμφανισιακά. Μια εφαρμογή η οποία είναι δυσνόητη στο πως να χρησιμοποιηθεί, είναι δύσκολη η πλοήγησή της και εμφανισιακά δεν είναι προσεγμένη είναι βέβαιο πως θα προκαλέσει δασαρχεία στους χρήστες και πιθανόν την απομάκρυνσή τους από αυτή. Το frontend αποτελείται από τρία κομμάτια την δομή (HTML), την εμφάνιση (CSS) και την λειτουργικότητα (JavaScript).

3.4.1 HTML

Η λέξη HTML προκύπτει από τα αρχικά HyperText Markup Language και αποτελεί το κομμάτι σε μια διαδικτυακή εφαρμογή που είναι υπεύθυνο για την δομή της ιστοσελίδας που προβάλλεται κάθε φορά στην οθόνη της συσκευής [34]. Δημιουργήθηκε από τον Tim Berners-Lee περίπου το 1991 και παρότι αναφέρεται συχνά ως γλώσσα προγραμματισμού, κυρίως από νέο εισαχθέντες στον κλάδο, δεν είναι. Αποτελεί μια γλώσσα σήμανσης και μάλιστα είναι πολύ εύκολη στην εκμάθηση της. Η HTML είναι στην πραγματικότητα ένα αρχείο όπου μέσα περιέχει την δομή. Όλα τα αρχεία αυτού του τύπου πρέπει να τελειώνουν με την κατάληξη '.html'. Τα αρχεία αυτά αποτελούνται από μια σειρά οντοτήτων που ονομάζονται στοιχεία (elements). Τα στοιχεία με τη σειρά τους, αναπαριστώντα με ετικέτες (tags) και κάθε στοιχείο θα πρέπει να έχει στοιχείο έναρξης και στοιχείο τερματισμού. Παράδειγμα ενός τέτοιου στοιχείου που αναπαριστά μια παράγραφο είναι το εξής ' … '. Ο αριθμός των στοιχείων είναι μεταβλητός και δεν υπάρχει κάποιο όριο ως προς τη χρήση τους. Ωστόσο υπάρχει όριο στον ελάχιστο αριθμό που μπορούν να χρησιμοποιηθούν καθώς υπάρχουν τρία συν ένα που πρέπει να περιέχει κάθε HTML έγγραφο [35].

1. <DOCTYPE! html>: Το συγκεκριμένο δεν αποτελεί στοιχείο αλλά είναι απαραίτητο για να καθορίσει τον τύπο του αρχείου.
2. <html></html >: Το στοιχείο χρησιμοποιείται για να ορίσει το ριζικό στοιχείο ενός αρχείου HTML και περιέχει όλα τα υπόλοιπα στοιχεία της σελίδας.
3. <head></head>: Σε αυτό το στοιχείο συμπεριλαμβάνονται όλα εκείνα που είναι απαραίτητα για την σελίδα αλλά τα οποία δεν είναι ορατά.
4. <body></body>: Σε αυτό το στοιχεία περιλαμβάνεται όλο το φανερό περιεχόμενο της σελίδας.

Με την πάροδο του χρόνου η HTML χρειάστηκε αλλαγές ώστε να μείνει κοντά στις απαιτήσεις του κοινού οι οποίες μεγάλωναν ολοένα και περισσότερο. Στην προσπάθεια αυτή χάθηκε η έννοια των σημασιολογικών στοιχείων και τη θέση τους πήραν νέα όπως το , τα οποία δεν προσφέρουν σημασιολογία. Είναι σημαντικό να χρησιμοποιούνται σημασιολογικά στοιχεία καταρχάς γιατί είναι εύκολο στην ανάγνωση, έχει καλύτερη προσβασιμότητα καθώς είναι πιο κατανοητά και δημιουργούν πιο συνεπή και οργανωμένο κώδικα. Παρά τα πλεονεκτήματα, παρατηρείται εκτενώς το φαινόμενο χρησιμοποίησης μη σημασιολογικών στοιχείων. Μερικά σημασιολογικά στοιχεία είναι τα παρακάτω [36]:

- <article></article>
- <nav></nav>
- <details></details>
- <aside></aside>
- <figure></figure>
- <header></header>
- <section></section>
- <footer></footer>
- <main></main>

Είναι καλή πρακτική το περιεχόμενο να διασπάται και να ακολουθείται η λογική των συστατικών (components). Πρακτικά αυτό σημαίνει ότι κάθε κομμάτι της εφαρμογής θα είναι ανεξάρτητο από τα υπόλοιπα, εμφανισιακά, και θα μπορεί να χρησιμοποιηθεί οπουδήποτε. Βεβαίως θα πρέπει να υπάρχει το γενικό αρχείο που θα 'εξηγεί' στον περιηγητή ότι πρόκειται για αρχείο HTML και μέσα σε αυτό θα

φορτώνονται τα επιμέρους τμήματα-components. Πιο συγκεκριμένα, στο blade template engine του Laravel που χρησιμοποιήθηκε για την παραγωγή HTML περιεχομένου στην εφαρμογή, είναι μια εύκολη και γρήγορη διαδικασία η λογική των components αφού παρέχει μεθόδους ειδικά γι' αυτόν τον σκοπό. Τέλος, τα αρχεία HTML έχουν την δυνατότητα να περιέχουν κώδικα για την μορφοποίηση του περιεχομένου καθώς και για την λειτουργεία του [34]. Αυτό επιτυγχάνεται με δύο τρόπους είτε χρησιμοποιώντας αναφορά προς άλλα αρχεία τα οποία περιέχον τον κώδικα, είτε χρησιμοποιώντας δύο ειδικές ετικέτες μία για κάθε περίπτωση, <stlyle></style> και <script></script>. Μέσα σε αυτές τις ετικέτες μπορεί να γραφεί κώδικας CSS και JavaScript αντίστοιχα. Στην περίπτωση των CSS υπάρχει και τρίτος τρόπος, αξιοποιώντας την ιδιότητα (attribute) style την οποία υποστηρίζουν όλα τα HTML στοιχεία. Να επισημανθούν δύο πράγματα, η χρήση της ιδιότητας style θα αντικαταστήσει οποιοδήποτε άλλο στυλ έχει το στοιχείο ενώ, η μέθοδος που προτιμάτε είναι η πρώτη, χρησιμοποιώντας δηλαδή αναφορές σε άλλα αρχεία οπού εκεί θα βρίσκεται ο ζητούμενος κώδικας. Για να μπορέσει ο κώδικας που τροποποιεί το HTML αρχείο να «δει» και να αλλάξει τα στοιχεία που θέλουμε πρέπει να χρησιμοποιηθούν ειδικοί HTML επιλογείς (selectors) όπως, όνομα id και όνομα κλάσης. Αυτοί οι δύο είναι οι πιο συχνά χρησιμοποιούμενοι επιλογείς, με το id να πρέπει να είναι μοναδικό ενώ το όνομα κλάσης μπορεί να χρησιμοποιηθεί σε πιο πολλά από ένα στοιχεία.

3.4.2 CSS

Τα αρχικά CSS προκύπτουν από τις λέξεις Cascading Style Sheets και πρόκειται για το κομμάτι που διαχειρίζεται την εμφάνιση κάθε διαδικτυακής εφαρμογής [37]. Όπως αναλύθηκε στην προηγούμενη παράγραφο, η HTML είναι υπεύθυνη για το περιεχόμενο που θα εμφανιστεί, το περιεχόμενο όμως από μόνο του είναι στατικό και 'βαρετό'. Εδώ έρχονται τα CSS το οποία παίρνουν το περιεχόμενο και το σχεδιάζουν όπως θέλει ο προγραμματιστής ώστε να είναι πιο ελκυστικό, όμορφο και διαδραστικό για τον χρήστη. Από μόνα τους τα CSS δεν μπορούν να χρησιμοποιηθούν. Είναι μια γλώσσα που βασίζεται σε κανόνες οι οποίοι κανόνες εφαρμόζονται στα HTML στοιχεία. Εκτός από το συντακτικό δεν βάζει περιορισμούς στο τι κανόνας θα εφαρμοστεί σε κάθε στοιχείο ή αν θα αλλάξουμε τους κανόνες του μία η περισσότερες φορές. Ωστόσο, αυτό μπορεί να γίνει παγίδα καθώς στη περίπτωση που αλλάξουμε τον ίδιο κανόνα σε ένα στοιχείο θα ισχύσει ο τελευταίος με φορά από την κορυφή του αρχείου προς το τέλος. Τα CSS παρέχουν μια τεχνική, ονόματι Media Queries η οποία δίνει την δυνατότητα να διαχειριστούμε το περιεχόμενο βάση της ανάλυσης της οθόνης, του πλάτους της οθόνης ή τον γενικό τύπο της συσκευής. Τεχνική η οποία απλοποιεί την διαδικασία ανάπτυξης και βοηθάει στο να μένει η εφαρμογή συμβατή με όλους τους τύπους συσκευών και οθονών. Η χρήση των CSS προσφέρει αρκετά πλεονεκτήματα όπως [38]:

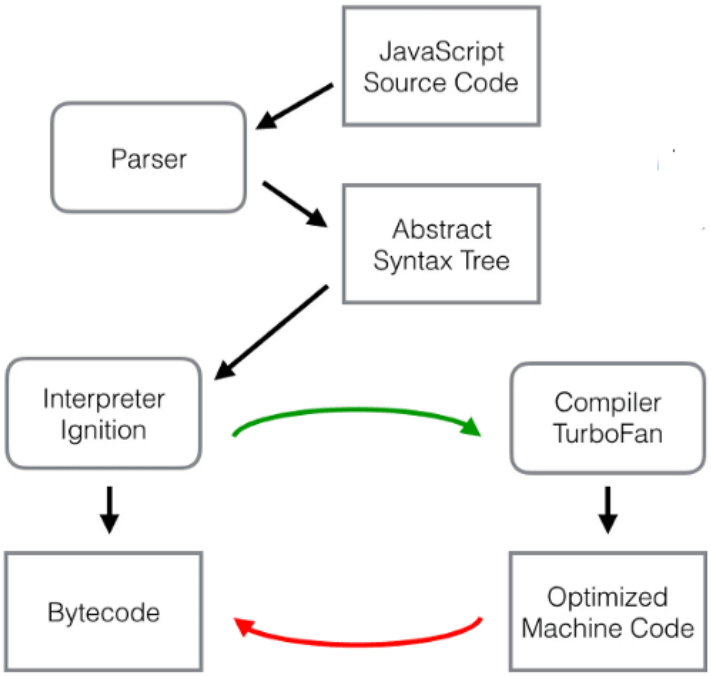
1. Εξοικονομεί χρόνο
2. Ευκολία στην συντήρηση
3. Γρηγορότερη φόρτωση των σελίδων
4. Συμβατότητα μεταξύ συσκευών
5. Περισσότερους κανόνες από την HTML
6. Ακολουθεί τα παγκόσμια πρότυπα ιστού

3.4.3 JavaScript

Η JavaScript είναι μια scripting γλώσσα προγραμματισμού που επιτρέπει την υλοποίηση περίπλοκων λειτουργιών στις ιστοσελίδες [39]. Είναι η τεχνολογία που έρχεται μετά την HTML και τα CSS να συμπληρώσει την τριπλέτα των εφαρμογών που χρησιμοποιούνται στις διαδικτυακές εφαρμογές στην μεριά του πελάτη-περιηγητή ιστού. Επιτρέπει την δημιουργία δυναμικού και διαδραστικού περιεχομένου. Εμφανίστηκε για πρώτη φορά το 1995 και ήταν γνωστή με το όνομα LiveScript το οποίο άλλαξε λίγο αργότερα σε αυτό που χρησιμοποιεί μέχρι και σήμερα [40].

Η JavaScript βασίζεται στο πρότυπο ECMAScript το οποίο ορίζει τον πυρήνα της γλώσσας [41]. Είναι μια γλώσσα προγραμματισμού general purpose ώστε να εξασφαλίσει την συμβατότητα με όλους τους περιηγητές ιστού και τις συσκευές. Αν και αρχικά σχεδιάστηκε για να τρέχει στη μεριά του πελάτη, με την πάροδο του χρόνου αναπτύχθηκε πολύ, σε σημείο που δημιουργήθηκε runtime περιβάλλον, ονόματι Node.js, για την μεριά του server. Ωστόσο δεν θα εμβαθύνουμε στο κομμάτι της backend χρήσης της JavaScript αφού δεν έχει χρησιμοποιηθεί στην υλοποίηση της εφαρμογής.

Επιπλέον χαρακτηριστικό της JavaScript αποτελεί το γεγονός ότι είναι μια γλώσσα που ερμηνεύεται (interpreted programming language). Ο περιηγητής ιστού μόλις λάβει την κώδικα JavaScript στην αρχική του μορφή αρχίζει να τον εκτελεί παρότι οι πιο καινούργιοι διερμηνείς (interpreters) χρησιμοποιούν την τεχνική just-in-time η οποία στοχεύει στην βελτίωση της απόδοσης. Αυτό που κάνει αυτή η τεχνική στην ουσία είναι να μετατρέπει τον κώδικα σε δυαδική μορφή καθώς το script χρησιμοποιείται για γρηγορότερη απόδοση. Το γεγονός ότι η μετατροπή κώδικα γίνεται κατά την διάρκεια της εκτέλεσης και όχι νωρίτερα είναι ο λόγος που η JavaScript θεωρείται γλώσσα που ερμηνεύεται [39].



Σχήμα 4.6: Διαδικασία εκτέλεσης JavaScript

Επιπρόσθετα, η JavaScript υποστηρίζει αρχές αντικειμενοστραφούς προγραμματισμού ωστόσο δεν βασίζεται σε κλάσεις, το πιο γνωστό μοντέλο αντικειμενοστρέφιας. Αντιθέτως, είναι μια γλώσσα που βασίζεται σε πρωτότυπα (prototype-based) [42]. Η έννοια ενός πρωτότυπου αντικειμένου, βασίζεται σε ένα πρωτότυπο, ενός αντικειμένου που χρησιμοποιείται ως πρωτότυπο και από το οποίο λαμβάνονται οι αρχικές ιδιότητες για ένα νέο αντικείμενο. Πρακτικά αυτό το μοντέλο προσθέτει σε κάθε αντικείμενο μια ιδιότητα (property) ονόματι prototype. Το prototype είναι αντικείμενο και μέσα έχει ένα άλλο prototype και αυτό συνεχίζει μέχρι να φτάσει στο τελικό αντικείμενο που ονομάζεται Object.prototype.

Μέσα από την JavaScript παρέχετε και η δυνατότητα άμεσης αλληλεπίδρασης με τα στοιχεία που απαρτίζουν την ιστοσελίδα μέσω του Document Object Model (DOM). Το DOM είναι μια προγραμματιστική διεπαφή των διαδικτυακών εγγράφων [43]. Ουσιαστικά, παίρνει το HTML περιεχόμενο και το μετατρέπει σε κόμβους και αντικείμενα προκειμένου ο προγραμματιστείς να μπορεί να αλληλοεπιδράσει μαζί του και να διαχειριστεί δυναμικά το περιεχόμενο. Το DOM δεν αποτελεί μέρος της JavaScript, είναι απλός ένα Web API το οποίο χρησιμοποιεί για να έχει προγραμματιστική πρόσβαση στη δομή της σελίδας.

Τα πλεονεκτήματα που προκύπτουν από τη χρήση της JavaScript είναι αρκετά. Όπως έχουμε ήδη αναφέρει μπορεί να αυξήσει την διαδραστικότητα της σελίδας με τον χρήστη καθώς και να παρέχει αντικείμενα τύπου slider, drag and drop κ.α. Επίσης σημαντικό είναι ότι μπορεί να μειώσει την κίνηση στον server καθώς μπορεί και παρέχει έως ένα βαθμό ασφάλεια και έτσι μπορεί να κρίνει αν θα στείλει τελικά το αίτημα ή όχι. Επιπλέον, υποστηρίζεται σχεδόν σε όλα, αν όχι σε όλα, τα λειτουργικά συστήματα, καθώς η χρήση της γίνεται μέσω των περιηγητών ιστού, ένας από τους λόγους που οδήγησε στην τελική απόφαση το frontend της εφαρμογής να γίνει σε περιβάλλον ιστού.

Παρά τις τεράστιες δυνατότητες και τα πλεονεκτήματα που έχει η JavaScript, έχει και κάποιες αδυναμίες τις οποίες είναι καλό να αναφέρουμε. Το πλεονέκτημα το ότι υποστηρίζεται από όλους τους περιηγητές ιστού μπορεί να είναι ταυτόχρονα και πρόβλημα. Ας μην ξεχνάμε ότι όλοι οι περιηγητές είναι ξεχωριστά λογισμικά, το κάθε ένα με τις ιδιαιτερότητές του, που εκτελούν τον κώδικα με διαφορετικό τρόπο ενδεχομένως. Δεν είναι κανόνας αλλά θέλει προσοχή καθώς και συνεχή έλεγχο προκειμένου να αποφευχθεί κάποιο δυσάρεστο αποτέλεσμα για τον χρήστη. Η ασφάλεια είναι ένα μεγάλο μειονέκτημα. Επειδή ο κώδικας της JavaScript, εκτελείται στον περιηγητή ιστού, είναι προσβάσιμος και αυτό μπορεί να είναι επικίνδυνο για την εφαρμογή από κακόβουλους χρήστες. Γι' αυτό και οι σημαντικές διαδικασίες που αφορούν την διαχείριση των δεδομένων καθώς και οι αλληλεπιδράσεις με τις βάσεις δεδομένων γίνονται αποκλειστικά στην μεριά του server όπου ο χρήστης δεν έχει καμία αλληλεπίδραση. Ακόμα ένα μειονέκτημα είναι το γεγονός ότι δεν υποστηρίζει πολλαπλά νήματα και πολλαπλές επεξεργασίες. Τελευταίο αλλά εξίσου σημαντικό, ο εντοπισμός σφαλμάτων (debugging) στην JavaScript μπορεί να γίνει επίπονος και χρονοβόρος.

Η JavaScript παρέχει την δυνατότητα στον προγραμματιστή να χρησιμοποιήσει πλαίσιο προγραμματισμού για την ανάπτυξη εφαρμογών στην μεριά του πελάτη-περιηγητή ιστού. Ένα πλαίσιο προγραμματισμού στην JavaScript αποτελείται από μια συλλογή βιβλιοθηκών κώδικα, οι οποίες παρέχουν συγκεκριμένες και επαναχρησιμοποιήσιμες λειτουργίες. Υπάρχει πληθώρα πλαισίων προγραμματισμού στην JavaScript, ενδεικτικά θα αναφέρουμε μερικά από τα πιο δημοφιλή [44]:

- Angular
  Είναι πλαίσιο προγραμματισμού ανοιχτού κώδικα, χρησιμοποιεί την αρχιτεκτονική MVC και έχει αναπτυχθεί από την Google. Είναι πολύ διαδεδομένο και μεγάλο ποσοστό των προγραμματιστών έχει ασχοληθεί με αυτό. Προτιμάται, για την ανάπτυξη μεγάλων και πολύπλοκων εφαρμογών.

- React
  Δημιουργήθηκε από την Facebook το 2013 και αποτελεί κορυφαία επιλογή πλαισίου προγραμματισμού ανοιχτού κώδικα. Είναι ελαφρύ, εύκολο στην εκμάθηση και έχει τεράστια κοινότητα προγραμματιστών. Η React είναι ιδανική για ανάπτυξη εφαρμογών μονών σελίδων (SPAs). Ενίοτε, χρησιμοποιείται και για ανάπτυξη μεγαλύτερων εφαρμογών.

- Vue.js
  Κυκλοφόρησε για πρώτη φορά το 2014 και αποτελεί ένα πλαίσιο προγραμματισμού ανοιχτού κώδικα το οποίο προτιμάται για μικρές εφαρμογές. Ξεχωρίζει για την ευελιξία του, το συμβατό μέγεθος του και την ταχύτητα του. Δεν περιπλέκει τον κώδικα και παράλληλα διατηρεί την διάσπαση της λογικής από την εμφάνιση.

- jQuery
  Δημοσιεύτηκε το 2006 και είναι μια από τις πιο παλιές βιβλιοθήκες ανοιχτού κώδικα της JavaScript. Είναι γνωστή για την ελαχιστοποίηση κώδικα JavaScript και ταυτόχρονα είναι ελαφριά και πλούσια σε δυνατότητες. Το μεγάλο της πλεονέκτημα είναι η υποστήριξη που υφίσταται από όλους τους περιηγητές ιστού καθιστώντας την ιδανική επιλογή για ανάπτυξη διαδικτυακών εφαρμογών.

- Ember
  Δημοσιεύτηκε το 2011 και πρόκειται για ένα πλαίσιο προγραμματισμού που υποστηρίζει δύο αρχιτεκτονικά μοντέλα, το MVC και το MVVM. Έχει χτιστεί πάνω στον κινητήρα απόδοσης Glimmer που θεωρείται ένας από τους πιο γρήγορους κινητήρες απόδοσης. Με την πάροδο του χρόνου εξελίχθηκε και προσφέρει τη δυνατότητα χρήσης σε ανάπτυξη εφαρμογών έξυπνων τηλεφώνων.

- Svelte
  Πρόκειται για έναν μεταγλωττιστή ανοιχτού κώδικα στο frontend. Παρουσιάστηκε πρώτη φορά το 2016 και από τότε έχει χρησιμοποιηθεί σε αρκετές ιστοσελίδες. Το γεγονός ότι είναι μεταγλωττιστής το κάνει να είναι ένα από τα πιο ελαφριά πλαίσια προγραμματισμού. Δεν έχει μεγάλη απήχηση και χρησιμοποιείται για μικρού όγκου εφαρμογές.

Στην υλοποίηση της πτυχιακής εργασίας χρησιμοποιήθηκε η jQuery βιβλιοθήκη, η οποία θα αναλυθεί στην επόμενη ενότητα.

3.4.4 jQuery

Η jQuery είναι μια μικρή, γρήγορη και πλούσια σε χαρακτηριστικά βιβλιοθήκη της JavaScript. Είναι ικανή για τη διαχείριση HTML αρχείων, διαχείριση γεγονότων και animations καθώς και χρήση ασύγχρονων αιτημάτων προς τον server, μέσω της μεθόδου AJAX. Χάρη στην ευελιξία της και της επεκτασιμότητας της άλλαξε ο τρόπος με τον οποίο εκατομμύρια άνθρωποι γράφουν κώδικα JavaScript.

[45]. Ενδεικτικά εταιρείες που χρησιμοποιούν αυτήν την τεχνολογία είναι η Microsoft, Google, Netflix, IBM. Υποστηρίζεται από όλους τους περιηγητές ιστού και συχνά αναφέρεται για αυτήν το λογοπαίγνιο «γράψε λιγότερα κάνε περισσότερα» επειδή έχει τη δυνατότητα μέσω μεθόδων να απλοποιεί τον κώδικα και να χρειάζεται λιγότερες γραμμές προκειμένου να πράξει το αποτέλεσμα σε σχέση με την JavaScript [46]. Λειτουργίες που προσφέρει η jQuery είναι:

- Διαχείριση εγγράφων HTML
- Διαχείριση του DOM
- Επιλογή στοιχείων του DOM
- Διαχείριση των CSS
- Προσθήκη εφέ και κίνησης στο περιεχόμενο
- AJAX επικοινωνία με τον server
- Διαχείριση γεγονότων
- Διαχείριση αρχείων τύπου JSON
- Επεκτασιμότητα

3.4.5 Bootstrap

Το bootstrap είναι μια συλλογή εργαλείων ανοιχτού κώδικα με σκοπό την δημιουργία διαδικτυακών εφαρμογών οι οποίες θα είναι προσαρμόσιμες [47]. Με την έννοια προσαρμόσιμες εννοούμε την ικανότητα που περιεχομένου της σελίδας να αναπροσαρμόζεται κατάλληλα ώστε η εφαρμογή να είναι λειτουργική σε όλα τα μεγέθη συσκευών. Θεωρείται πλαίσιο προγραμματισμού για πιο εύκολη και γρήγορη ανάπτυξη εφαρμογών από τη μεριά του frontend. Επίσης δίνει την δυνατότητα χρησιμοποίησης έτοιμων components, για παράδειγμα μπορεί να γίνει χρήση ενός έτοιμου navigation bar στην εφαρμογή έχοντας παράλληλα ο προγραμματιστής την δυνατότητα να το τροποποιήσει. Αυτό επιταχύνει σαφώς την ταχύτερη ανάπτυξη του λογισμικού. Επιπρόσθετα, είναι τεχνολογία που υποστηρίζεται σχεδόν σε όλους του περιηγητές ιστού.

3.5 Visual Studio Code

Για την ανάπτυξη της εφαρμογής, τόσο στο backend κομμάτι όσο και στο frontend, χρησιμοποιήθηκε το λογισμικό Visual Studio Code. Το Visual Studio Code είναι ένα ελαφρύ αλλά πολύ ισχυρό λογισμικό επεξεργασίας πηγαίου κώδικα το οποίο αναπτύχθηκε από την εταιρεία Microsoft και μπορεί να χρησιμοποιηθεί δωρεάν από τους προγραμματιστές [48]. Αποτελεί μια desktop εφαρμογή η οποία υποστηρίζεται από τα λειτουργικά συστήματα Windows, MacOS και Linux δίνοντας τη δυνατότητα στον εκάστοτε προγραμματιστή να εργασθεί ανεξάρτητα από τη συσκευή και την πλατφόρμα που διαθέτει.

Το Visual Studio Code βασίζεται στην λογική επεξεργασία - κατασκευή - εντοπισμός σφαλμάτων με στόχο ο προγραμματιστής να επικεντρώνεται πιο πολύ στις ιδέες του και την λογική που θέλει να αναπτύξει και λιγότερο στην αλληλεπίδραση με το λογισμικό περιβάλλον [49]. Όπως αναφέρθηκε είναι ένας απλός επεξεργαστής πηγαίου κώδικα όμως προσφέρει πολύ ισχυρά εργαλεία όπως είναι ο εντοπισμός σφαλμάτων και η αυτόματη συμπλήρωση κώδικα με χρήση της λειτουργίας IntelliSense. Η λειτουργία IntelliSense είναι ένας οδηγός συμπλήρωσης κώδικα που προτείνει στον προγραμματιστεί τι μπορεί να χρησιμοποιήσει όπως, συμπλήρωση της λέξης που πληκτρολογεί, ονόματα μεθόδων που μπορούν να χρησιμοποιηθούν καθώς και πληροφορίες για τις παραμέτρους τους, διαθέσιμες ιδιότητες αντικειμένων κ.α.

Το Visual Studio Code έχει ενσωματωμένη υποστήριξη για τις τεχνολογίες JavaScript, TypeScript και Node.js. Δεν σταματάει όμως εκεί. Δίνει την δυνατότητα προσθήκης επεκτάσεων για οποιαδήποτε άλλη γλώσσα και runtime χρειάζεται ο προγραμματιστής. Η προσθήκη επεκτάσεων δεν αφορά μόνο γλώσσες και runtime περιβάλλοντα. Υπάρχουν επεκτάσεις που στοχεύουν στις ανάγκες του χρήστη. Για παράδειγμα υπάρχει δυνατότητα εγκατάστασης επέκτασης η οποία διαχειρίζεται την μορφή HTML αρχείων με σκοπό να κρατάει την δομή καθαρή και οργανωμένη. Τέλος, το VS Code έχει ενσωματωμένες εντολές του git και παρέχει τη δυνατότητα στον χρήστη να τις χρησιμοποιεί μέσα από το γραφικό περιβάλλον του λογισμικού και όχι από γραμμή εντολών καθώς και να βλέπει τροποποιημένα αρχεία που εκκρεμούν. Με αυτό το τρόπο κρατάει τον χρήστη ενήμερο χωρίς να χρειάζεται να φύγει από τον επεξεργαστή πηγαίου κώδικα.

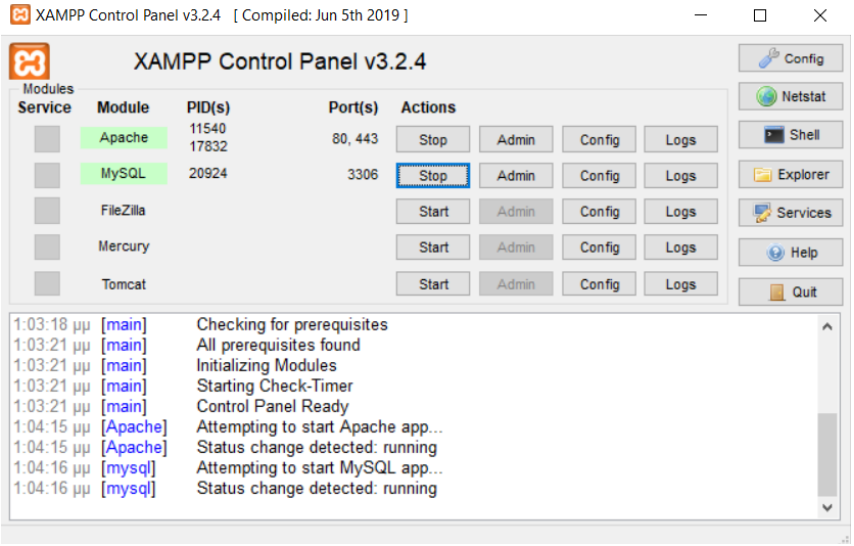
3.6 XAMPP

Απαραίτητη προϋπόθεση για κάθε διαδικτυακή εφαρμογή είναι η ύπαρξη ενός server ο οποίος θα εξυπηρετήσει το backend τμήμα της εφαρμογής. Το λογισμικό που επιλέχθηκε για να πραγματοποιήσει αυτή την λειτουργία είναι το XAMPP. Τα αρχικά προκύπτουν από τις λέξεις Cross Platform (X), Apache (A), MySQL (M), php και Perl (PP) και είναι ένα από τα πιο δημοφιλή και συχνά χρησιμοποιούμενα περιβάλλοντα υποστήριξης ανάπτυξης php κώδικα. Υποστηρίζεται από όλα τα λειτουργικά συστήματα Windows, MacOS και Linux δίνοντας την ελευθερία στον εκάστοτε προγραμματιστή να χρησιμοποιήσει την πλατφόρμα που επιθυμεί σε όλη την πορεία ανάπτυξης της εφαρμογής [50].

Κατά την διάρκεια ανάπτυξης μιας διαδικτυακής εφαρμογής συνηθίζεται το backend κομμάτι να εξυπηρετείτε τοπικά στον υπολογιστή του προγραμματιστή προκειμένου να ελέγχεται ώσπου να φτάσει στην τελική έκδοση και να «ανέβει» στον βασικό server. Το XAMPP περιέχει την τεχνολογία Apache η οποία εξυπηρετεί ακριβώς αυτόν τον σκοπό, του τοπικού διακομιστή ιστού (local web server).

Η βάση δεδομένων είναι εξίσου σημαντική στην υλοποίηση εφαρμογών καθώς εκεί θα αποθηκεύονται τα δεδομένα. Το XAMPP παρέχει ένα σύστημα διαχείρισης βάσης δεδομένων (DBMS) που λέγεται MariaDB. Αποτελεί μια επέκταση του MySQL και πρακτικά υποστηρίζει το 100% των λειτουργιών του. Αρχικά το XAMPP παρείχε την MySQL ως σύστημα βάσης δεδομένων ωστόσο με τη πάροδο των χρόνων αντικαταστάθηκε. Πρακτικά υποστηρίζονται και τα δύο αυτά συστήματα και δεν έχουν διαφορά στην χρήση. Η εφαρμογή υλοποιήθηκε με χρήση της MySQL.

Επιπρόσθετα, το XAMPP παρέχει ένα γραφικό περιβάλλον διαχείρισης της βάσης δεδομένων ονόματι, phpMyAdmin το οποίο χρησιμοποιήθηκε κατά την ανάπτυξη της εφαρμογής και θα αναλυθεί στην επόμενη ενότητα. Οι scripting γλώσσες προγραμματισμού php και Perl είναι το τελευταίο κομμάτι που περιέχει το πακέτο XAMPP. Η php ήταν η γλώσσα προγραμματισμού που χρησιμοποιήθηκε στην ανάπτυξη της εφαρμογής και έχει αναλυθεί σε προηγούμενο κεφάλαιο. Τέλος, το XAMPP παρέχει έναν γραφικό πίνακα ελέγχου προκειμένου να υπάρχει καλύτερη οργάνωση στις λειτουργίες του.

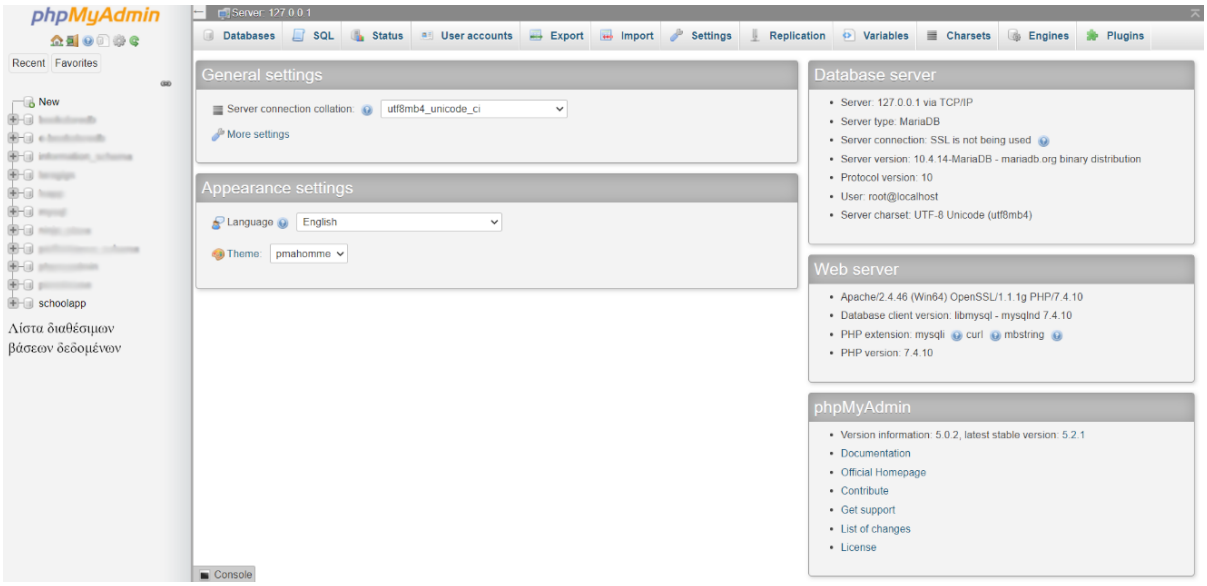


Σχήμα 4.7: Ο πίνακας ελέγχου του λογισμικού XAMPP

3.7 phpMyAdmin

Το phpMyAdmin είναι ένα δωρεάν λογισμικό, γραμμένο στη γλώσσα προγραμματισμού php με σκοπό την διαχείριση της MySQL και των δεδομένων της μέσω του ιστού [51]. Υποστηρίζει μεγάλο φάσμα λειτουργιών (π.χ. διαχείριση ολόκληρης της βάσης, πινάκων, στηλών) για την MySQL καθώς και την MariaDB. Για τις λειτουργίες αυτές προσφέρει ένα γραφικό περιβάλλον μέσω του οποίου ο χρήστης μπορεί να εκτελέσει αυτές τις λειτουργίες ενώ δίνεται και η δυνατότητα χρήσης κώδικα SQL και εφόσον είναι χτισμένο για το περιβάλλον του ιστού υπάρχει δυνατότητα χρήσης από οποιαδήποτε συσκευή και λειτουργικό σύστημα. Κάνοντας χρήση του λογισμικού phpMyAdmin μας προσφέρονται σημαντικά πλεονεκτήματα όπως [52]:

1. Μπορούμε με ευκολία να εκτελέσουμε ερωτήματα στη βάση καθώς να την διαχειριστούμε και συνολικά μέσα από ένα εύχρηστο και λειτουργικό γραφικό περιβάλλον αντί γραμμής εντολών.
2. Βοηθάει στον έλεγχο των δικαιωμάτων του χρήστη και μπορεί να διαχειρίζεται πολλούς servers ταυτόχρονα.
3. Μπορεί να υποστηριχθεί από οποιονδήποτε server ή λειτουργικό σύστημα αρκεί να υπάρχει περιηγητής ιστού.
4. Δίνει την δυνατότητα για δημιουργία αντιγράφων ασφαλείας καθώς και εξαγωγή των δεδομένων σε διάφορες μορφές όπως: Excel, CSV, Spreadsheet, XML, PDF κ.α.
5. Μπορούμε να εκτελέσουμε δύσκολα ερωτήματα SQL και να επεξεργαστούμε μεθόδους και γεγονότα μέσω του γραφικού περιβάλλοντος.

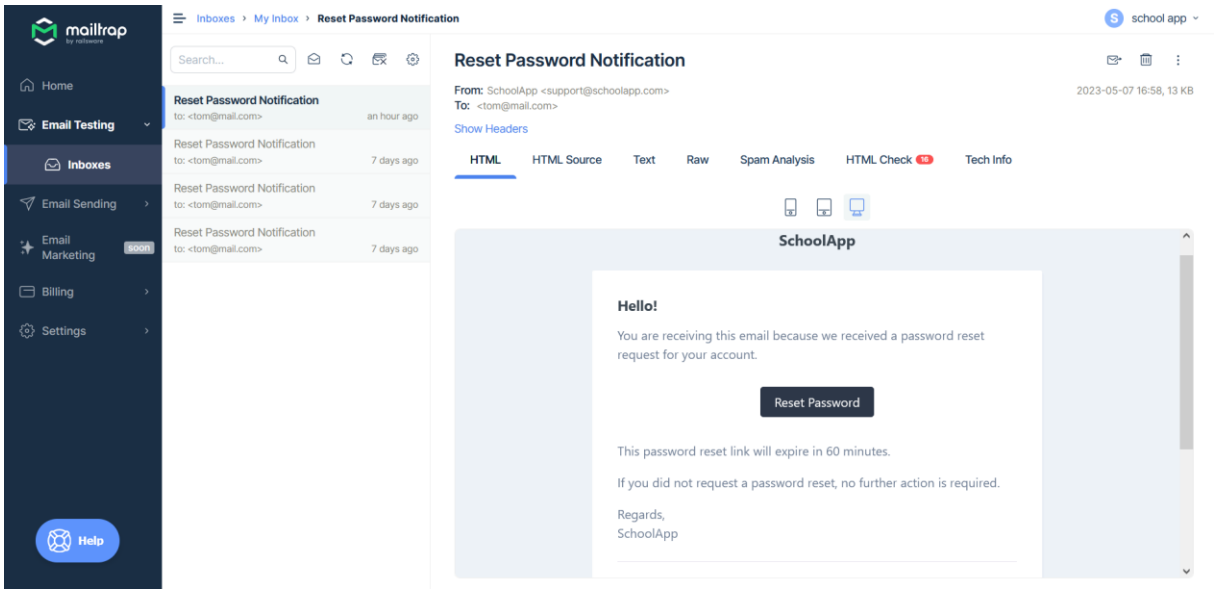


Σχήμα 4.8: Το γραφικό περιβάλλον του λογισμικού phpMyAdmin

3.8 Mailtrap

Μια από της βασικές λειτουργίες της εφαρμογής είναι η δυνατότητα επαναφοράς του κωδικού σύνδεσης σε περίπτωση που χρειαστεί από τον χρήστη. Αυτή η λειτουργία είναι αρκετά απλή για να επιτευχθεί, χρησιμοποιώντας απλώς έναν λογαριασμό Gmail και ορισμένες αλλαγές στον αρχείο περιβάλλοντος του πλαισίου προγραμματισμού Laravel. Για να μπορέσει να «τρέξει» αυτή η διαδικασία θα πρέπει στον λογαριασμό του Gmail να γίνει ενεργοποίηση της ρύθμισης, πρόσβαση σε λιγότερο ασφαλείς εφαρμογές. Ωστόσο, για λόγους ασφαλείας η Google από τις 30 Μαΐου 2022, δεν υποστηρίζει πλέον την χρήση εφαρμογών ή συσκευών τρίτου μέρους, και αυτή η ρύθμιση δεν είναι πλέον διαθέσιμη. Προκειμένου να ξεπεραστεί αυτή η δυσκολία και να είναι λειτουργική η εφαρμογή έγινε χρήση του λογισμικού Mailtrap.

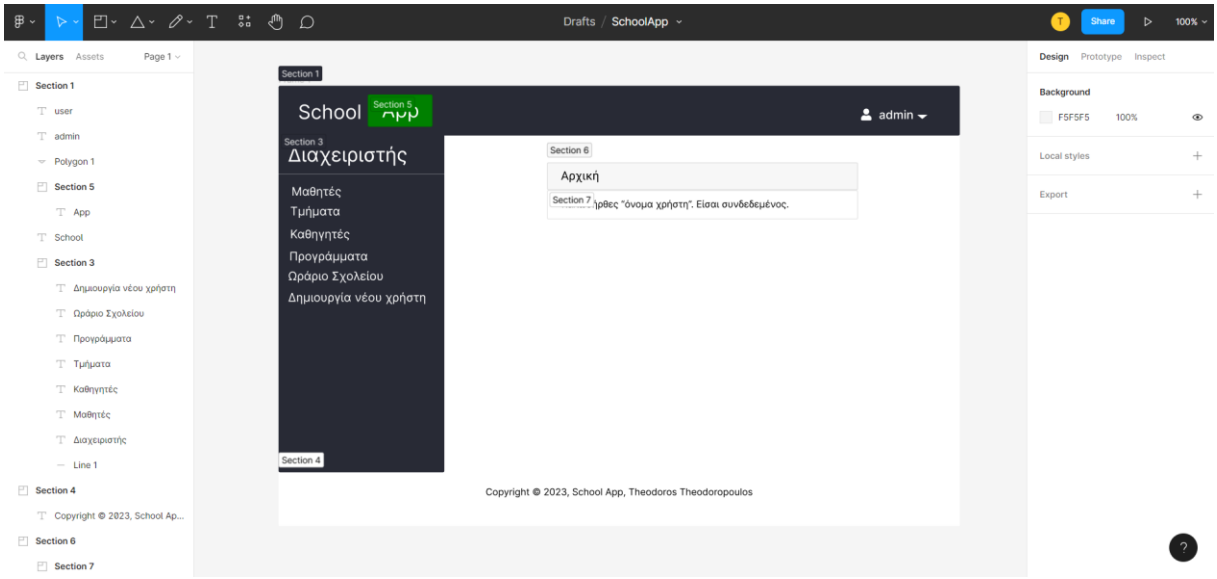
Το Mailtrap είναι μια πλατφόρμα παράδοσης email, που επιτρέπει στους χρήστες να διαχειρίζονται την δομή των emails σε ένα μέρος [53]. Διαθέτει τρείς διαθέσιμες εκδόσεις, μια εκ των οποίων είναι δωρεάν στη χρήση και καλύπτει όλες τις σχετικές ανάγκες με τα email όπως έλεγχο και αποστολή. Το Mailtrap αποθηκεύει όλα τα email στον φάκελο inboxes και αφού γίνει ο απαραίτητος έλεγχος της εμφανισιακής δομής του και της βαθμολογίας ανεπιθύμητων μηνυμάτων, μπορούμε να χρησιμοποιήσουμε το email API ή αναμεταδότη SMTP για να στείλουμε τα emails στους χρήστες. Το τελευταίο κομμάτι δεν χρειάστηκε εφόσον τα email που είναι κατοχυρωμένα στην βάση της εφαρμογής είναι testing emails.



Σχήμα 4.9: Το γραφικό περιβάλλον του λογισμικού Mailtrap

3.9 Figma

Η διεπαφή και η εμπειρία του χρήστη (UI/UX) με την εφαρμογή είναι πολύ σημαντική υπόθεση. Γι' αυτόν τον λόγο ενδείκνυται πριν ξεκινήσει η υλοποίηση του frontend τμήματος της εφαρμογής, που είναι υπεύθυνο για το περιεχόμενο που θα δει και θα αλληλοεπιδράσει ο χρήστης, να σχεδιαστεί η βασική διάταξη των περιεχομένων της κάθε σελίδας. Ένα λογισμικό που εξυπηρετεί αυτόν τον σκοπό είναι το Figma. Το Figma είναι ένα ισχυρό λογισμικό για την σχεδίαση διεπαφών χρήστη, το οποίο διατίθεται δωρεάν για όσους θέλουν να δημιουργήσουν [54]. Κυκλοφόρησε για πρώτη φορά το 2016 και μέχρι σήμερα έχει καταφέρει να έχει εκατομμύρια χρήστες. Το Figma επιλέχθηκε ώστε να υπάρξει μια πιο καθαρή και οργανωμένη εικόνα της διεπαφής κατά την διάρκεια ανάπτυξης της εφαρμογής.



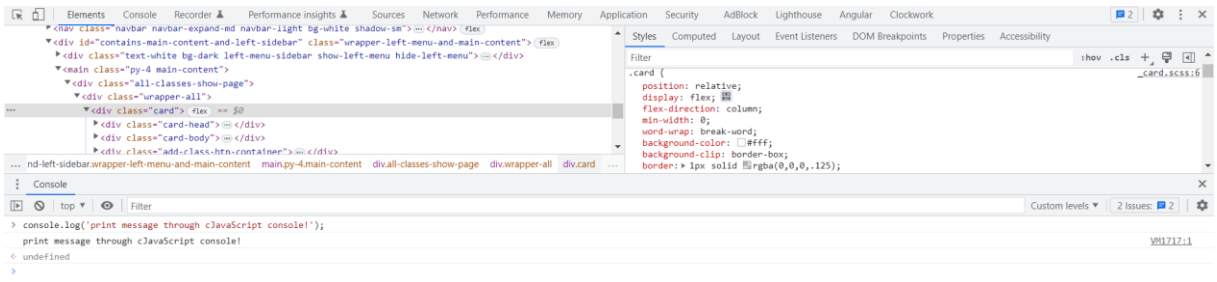
Σχήμα 4.10: Ενδεικτική εικόνα από το περιβάλλον εργασίας του λογισμικού Figma

3.10 Περιηγητής Ιστού

Φυσικά και όταν αναφερόμαστε σε μια διαδικτυακή εφαρμογή ο περιηγητής ιστού είναι το βασικό εργαλείο αφού μέσω αυτού θα μπορεί ο εκάστοτε χρήστης να αλληλεπιδράσει μαζί της. Όμως ένας περιηγητής ιστού μπορεί να βοηθήσει πάρα πολύ τον προγραμματιστή κατά την διάρκεια ανάπτυξης του frontend τμήματος της εφαρμογής. Κάθε περιηγητής ιστού περιλαμβάνει μια ισχυρή διεπαφή εργαλείων για προγραμματιστή [55]. Αυτά τα εργαλεία δίνουν τη δυνατότητα για πολλές ενέργειες όπως είναι επιθεώρηση των στοιχείων HTML που απαρτίζουν την ιστοσελίδα και των CSS τους, προβολή κώδικα JavaScript, τα αιτήματα που στέλνονται στον server και ο χρόνος απόκρισης τους και πολλά άλλα.

Η λειτουργία Inspector (επιθεωρητής) όπως υποδηλώνει και το όνομα της δίνει την δυνατότητα στον προγραμματιστή να επιθεωρήσει το τρέχον HTML περιεχόμενο της ιστοσελίδας καθώς και τι CSS κανόνες έχει κάθε στοιχείο [56]. Του δίνει την δυνατότητα να κάνει αλλαγές και στα στοιχεία που απαρτίζουν το περιεχόμενο καθώς και στα CSS τους. Οι αλλαγές αυτές είναι πρόσκαιρες καθώς στην πρώτη επαναφόρτωση της ιστοσελίδας ή αλλαγή σελίδας επανέρχονται οι προκαθορισμένες. Έστω και στιγμιαίες, αυτές οι αλλαγές είναι πολύ χρήσιμες για τον προγραμματιστή όταν θέλει να δοκιμάσει κάτι πάνω στο περιεχόμενο ή το στυλ καθώς είναι σε θέση να βλέπει ταυτόχρονα τις επιδράσεις των αλλαγών που κάνει πάνω στην ιστοσελίδα. Ο επιθεωρητής που χρησιμοποιείται για την επιλογή HTML στοιχείων ονομάζεται DOM Inspector. Τα CSS εμφανίζονται σε ένα παράθυρο που ονομάζεται επεξεργαστής των CSS και από εκεί ο εκάστοτε προγραμματιστής μπορεί να δει τι κανόνες εφαρμόζονται αλλά και να τους αλλάξει αν επιθυμεί, πάντα οι αλλαγές αυτές είναι στιγμιαίες.

Ακόμα ένα σημαντικό εργαλείο που παρέχεται είναι ένα πρόγραμμα εντοπισμού σφαλμάτων για κώδικα JavaScript (JavaScript debugger) [56]. Επιτρέπει την προβολή κώδικα καθώς και την χρήση breakpoints με σκοπό να σταματάει ο κώδικας και να εξακριβώνονται τα προβλήματα τα οποία προκαλούν σφάλματα στην εφαρμογή και δεν την αφήνουν να εκτελεστεί σωστά. Για την JavaScript παρέχεται ακόμη ένα εργαλείο, η κονσόλα. Η κονσόλα είναι πολύ ισχυρό και χρήσιμο εργαλείο καθώς επιτρέπει τον εντοπισμό σφαλμάτων αλλά δίνει και τη δυνατότητα να εκτελεστούν εντολές JavaScript πάνω στην τρέχον σελίδα. Τέλος, έχει την δυνατότητα να εμφανίζει μηνύματα από τέσσερις κατηγορίες. Η πιο σημαντική είναι η κατηγορία με τα σφάλματα. Είναι η πιο σημαντική διότι δίνει πληροφορίες στοχευμένα σε σημεία του κώδικα που δεν λειτουργούν όπως θα περιμέναμε. Οι υπόλοιπες κατηγορίες μηνυμάτων αφορούν πληροφορίες και προειδοποιήσεις.



Σχήμα 4.11: DOM Inspector, CSS επεξεργαστής, JavaScript κονσόλα

3.11 GitHub

Το GitHub είναι μια υπηρεσία repository hosting η οποία αναλαμβάνει την φιλοξενία του πηγαίου κώδικα της εφαρμογής. Πρόκειται για ένα λογισμικό ανοιχτού κώδικα που παρέχει version control αλλά και δυνατότητα διαχείρισης του πηγαίου κώδικα [57]. Στόχος του GitHub είναι να προφέρει σε ομάδες προγραμματιστών την δυνατότητα να «σπάνε» τον κώδικα σε κομμάτια και να δουλεύουν παράλληλα αλλά ανεξάρτητα μεταξύ τους. Το γεγονός αυτό προάγει την ταχύτητα ανάπτυξης, την συνεργασία μεταξύ των μελών της ομάδας καθώς και την καλύτερη διαχείριση της εφαρμογής.

Ωστόσο κατά την εκπόνηση της πτυχιακής εργασίας, το GitHub χρησιμοποιήθηκε με σκοπό την φιλοξενία του κώδικα σε μια cloud υπηρεσία προκειμένου να υπάρχει ένα αντίγραφο ασφαλείας το οποίο θα μπορεί να ανακτηθεί με ευκολία και ταχύτητα σε περίπτωση βλάβης ή καταστροφής του υπολογιστή που έγινε η ανάπτυξη της εφαρμογής.

3.12 Επίλογος

Στο κεφάλαιο αυτό έγινε λεπτομερής ανάλυση όλων των τεχνολογιών που επιλέχθηκαν για την ανάπτυξη της εφαρμογής τόσο στο backend όσο και στο frontend τμήμα της εφαρμογής. Μέσα από την λεπτομερή περιγραφή εξηγήθηκε το πλαίσιο προγραμματισμού Laravel, το οποίο είναι η βασική τεχνολογία της εφαρμογής, καθώς και ο λόγος που καθιστά μια τέτοια τεχνολογία ένα ισχυρό εργαλείο στα χέρια ενός προγραμματιστή μέσα από τις δυνατότητες που προσφέρει όπως ταχύτητα ανάπτυξης, ασφάλεια, ευκολία αλληλεπίδρασης με τα δεδομένα και πολλά άλλα. Επίσης, έγινε η αναφορά στην MySQL, μια σχεσιακή βάση δεδομένων με τεράστιες δυνατότητες καθώς και στο λογισμικό phpMyAdmin το οποίο δίνει την δυνατότητα αλληλεπίδρασης με την βάση μέσα από ένα γραφικό περιβάλλον. Εκτενής αναφορά έγινε και στο κομμάτι του frontend της εφαρμογής, και εξηγήθηκε πως αυτό λειτουργεί και ποιες τεχνολογίες το απαρτίζουν. Καθώς αναφερόμαστε σε διαδικτυακή εφαρμογή το frontend αναπτύχθηκε κυρίως με HTML, CSS και JavaScript. Τέλος, παρουσιάστηκαν και σχολιάστηκαν συντόμως, λογισμικά εργαλεία που βοήθησαν στην ανάπτυξη της εφαρμογής όπως ο επεξεργαστής κειμένου για την συγγραφή του κώδικα, ο τοπικός server που θα εξυπηρετεί την εφαρμογή, μια cloud υπηρεσία φιλοξενίας κώδικα με σκοπό την δημιουργία αντιγράφου ασφαλείας, εργαλεία ελέγχου και εντοπισμού σφαλμάτων που περιέχει ένας περιηγητής ιστού.

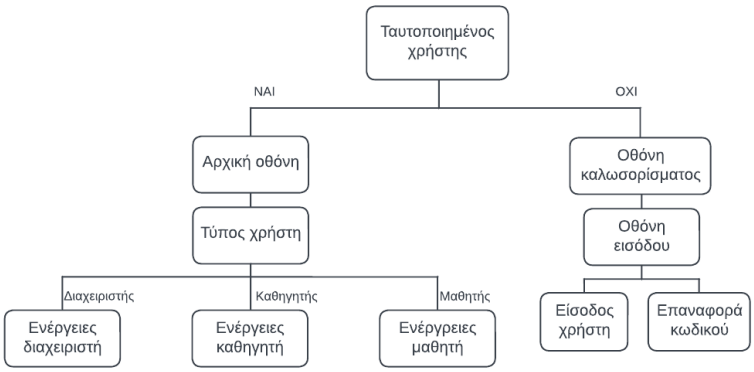
Κεφάλαιο 4ο: Δομή εφαρμογής

4.1 Εισαγωγή

Η δομή και ο σχεδιασμός μιας εφαρμογής είναι κάτι που ο εκάστοτε προγραμματιστής πρέπει να σκεφτεί και να οργανώσει πολύ καλά. Η τελική μορφή της προορίζεται για έναν απλό χρήστη ο οποίος δεν γνωρίζει τις τεχνολογίες ούτε πως έχει αναπτυχθεί και γι' αυτόν τον λόγο πρέπει η εφαρμογή που θα του παραδώσουμε να του είναι εύκολή και κατανοητή στην χρήση. Δύσκολα και πολύπλοκα γραφικά περιβάλλοντα και λειτουργίες πετυχαίνουν ακριβώς το αντίθετο με αποτέλεσμα ο χρήστης να μην επιθυμεί την αλληλεπίδραση με την εφαρμογή και σιγά σιγά να μειώσει την χρήση της. Από την άλλη μεριά, ένα όμορφο γραφικό περιβάλλον και απλό στη χρήση του, με συνεχή καθοδήγηση από την ίδια την εφαρμογή για τις λειτουργίες που του παρέχει, προσφέρει στον χρήστη μια όμορφη εμπειρία και την επιθυμία να χρησιμοποιεί αυτήν την εφαρμογή προκειμένου να καλύψει τις ανάγκες του. Σε αυτό το κεφάλαιο θα αναλυθεί η δομή και οι λειτουργίες που προσφέρει η εφαρμογή που αναπτύχθηκε στους χρήστες.

4.2 Περιγραφή εφαρμογής

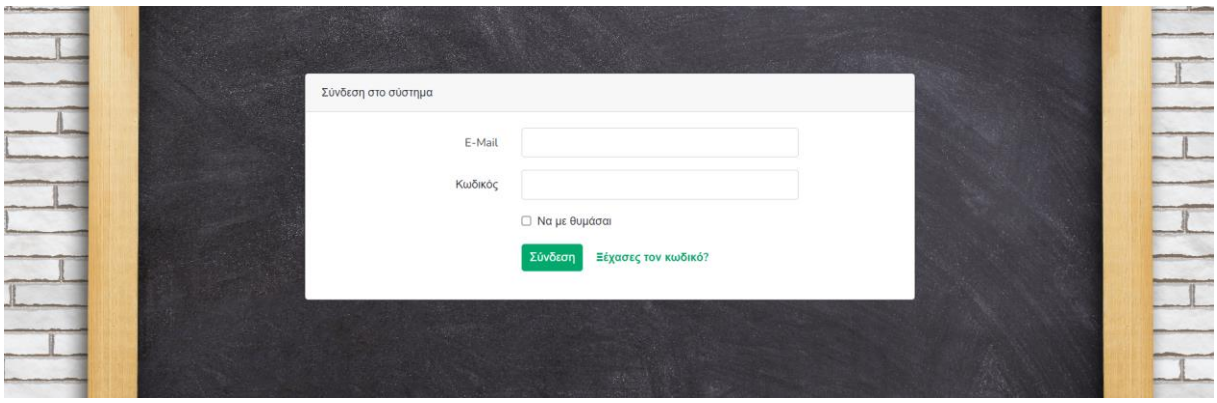
Η εφαρμογή απαρτίζεται από όλα τα βασικά χαρακτηριστικά που διέπουν μια διαδικτυακή εφαρμογή. Αποτελεί «εργαλείο» για ΓΕΛ της δευτεροβάθμιας εκπαίδευσης με στόχο τον οργάνωση και την διαχείριση των δεδομένων τους. Αυτό σημαίνει πως δεν μπορεί ο οποιοσδήποτε στο διαδίκτυο να αλληλεπιδράσει με το σύστημα αλλά μόνο οι εγγεγραμμένοι χρήστες. Για να θεωρηθεί κάποιος εγγεγραμμένος χρήστης θα πρέπει ο φορέας να του έχει δώσει πρόσβαση στο σύστημα. Η πρώτη οθόνη που αλληλεπιδρά ο χρήστης είναι η οθόνη εισόδου η οποία δίνει μόνο την δυνατότητα σύνδεσης στην εφαρμογή και επαναφοράς του κωδικού πρόσβασης ενώ δεν παρέχει την δυνατότητα εγγραφής. Αφού γίνει η επαλήθευση του χρήστη συμπληρώνοντας τα στοιχεία σύνδεσης του στην αντίστοιχη φόρμα εισόδου ο χρήστης κατευθύνεται στην κεντρική σελίδα της εφαρμογής εμφανίζοντάς του ένα μήνυμα καλωσορίσματος. Κατά την διάρκεια ταυτοποίησης ο χρήστης αντιστοιχίζεται σε έναν εκ των τριών τύπων χρηστών: διαχειριστής, καθηγητής ή μαθητής. Ανάλογα με τον τύπο του χρήστη του δίνονται αντίστοιχα δικαιώματα και λειτουργίες που μπορεί να χρησιμοποιήσει στην εφαρμογή.



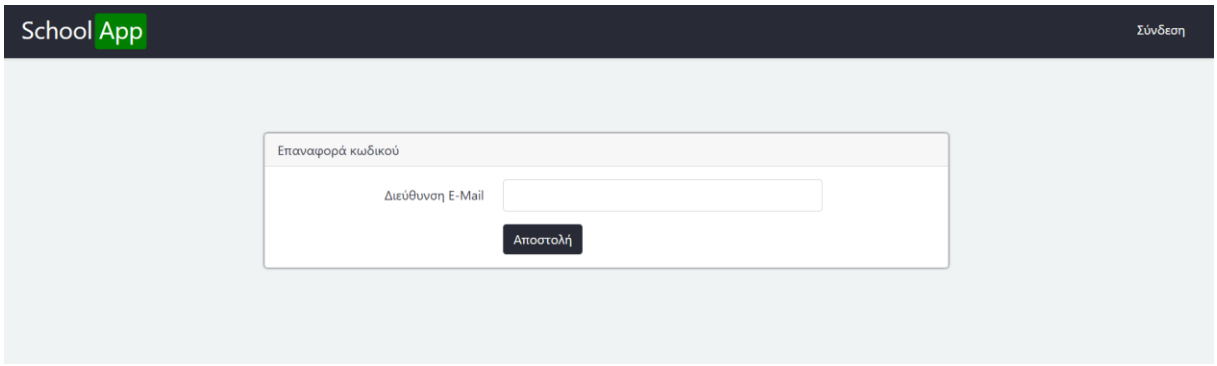
Σχήμα 5.1: Διάγραμμα δομής της εφαρμογής

4.3 Εξουσιοδότηση χρήστη – Σύνδεση στην εφαρμογή

Μπαίνοντας πρώτη φορά στην εφαρμογή πραγματοποιείται ένας έλεγχος για να διευκρινιστεί εάν ο χρήστης είναι ταυτοποιημένος ή όχι. Αυτό επιτυγχάνεται με την χρήση της μεθόδου check() που παρέχει το Auth facade του πλαισίου προγραμματισμού Laravel. Τα facades στο Laravel παρέχουν στατικά interfaces στις κλάσεις που είναι διαθέσιμες στο service container της εφαρμογής [58]. Με βάση λοιπόν το αποτέλεσμα της μεθόδου check() ο χρήστης οδηγείται στην κατάλληλη οθόνη. Στην περίπτωση που ο χρήστης δεν είναι έχει ταυτοποιηθεί οδηγείται στην οθόνη καλωσορίσματος. Η οθόνη αυτή περιλαμβάνει ένα κουμπί εισόδου με ενέργεια να μεταφέρει τον χρήστη στην σελίδα login η οποία περιέχει την φόρμα εισόδου του χρήστη. Όλες οι σελίδες είναι στην ουσία Laravel views και η κάθε τους ενέργεια διαχειρίζεται από κατάλληλες μεθόδους των αντίστοιχων controllers. Η σελίδα Login αποτελείται από την φόρμα συμπλήρωσης στοιχείων του χρήστη (email και password) και ακριβώς από κάτω έχει τρία actions, ένα για να «θυμάται» η εφαρμογή τον χρήστη ώστε να μην χρειάζεται να εισάγει τα δεδομένα στην φόρμα κάθε φορά που μπαίνει στην εφαρμογή, ένα για τον έλεγχο των στοιχείων που εισήγαγε ο χρήστης προκειμένου να γίνει η διαδικασία ταυτοποίησης του και ένα για επαναφορά κωδικού πρόσβασης. Εάν τα στοιχεία ταυτοποιηθούν τότε ο χρήστης μεταφέρεται στο περιβάλλον χρήσης της εφαρμογής στην αντίθετη περίπτωση λαμβάνει μήνυμα σφάλματος. Ο υπεύθηνος controller για αυτή την ενέργεια είναι ο LoginController. Στην περίπτωση της ενέργειας επαναφορά κωδικού πρόσβασης, μεταφέρεται στην οθόνη επαναφοράς κωδικού όπου υπάρχει μια φόρμα συμπλήρωσης του email του χρήστη προκειμένου να του σταλθεί ο σύνδεσμος για την επαναφορά του κωδικού ώστε να οδηγηθεί στην αντίστοιχη οθόνη και να επαναφέρει τον κωδικό του. Ο σύνδεσμος αυτός στέλνεται στο email του χρήστη και έχει ισχύ για μια ώρα. Μετά το πέρας της ώρας θα πρέπει να ζητήσει εκ νέου αίτημα αλλαγής κωδικού. Την επαναφορά κωδικού την διαχειρίζεται ο ResetPasswordController. Τέλος, στην περίπτωση που ο χρήστης επιλέξει να τον θυμάται η εφαρμογή, δημιουργείται ένα remember_web cookie το οποίο αποτελείται από το id του χρήστη, ένα τυχαίο token το οποίο αποθηκεύεται στην στήλη remember_token του πίνακα users της βάσης δεδομένων και των κωδικό του χρήστη. Φυσικά τα δεδομένα του χρήστη είναι hashed για λόγους ασφαλείας και το token ανανεώνεται κάθε φορά που αποσυνδέεται από την εφαρμογή για τον ίδιο λόγο.



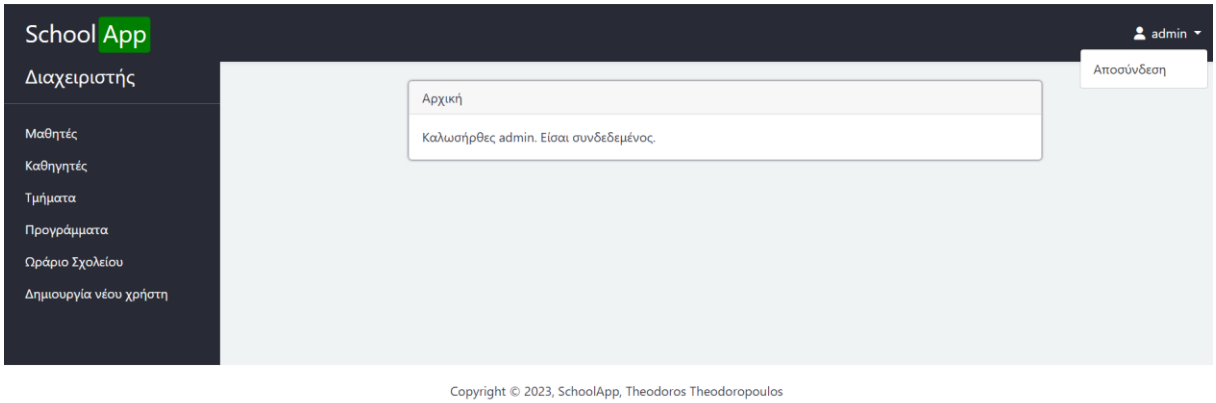
Σχήμα 5.2: Οθόνη σύνδεσης χρήστη



Σχήμα 5.3: Οθόνη επαναφοράς κωδικού

4.4 Κυρίως περιβάλλον

Αφού ο χρήστης εξουσιοδοτηθεί, μεταφέρεται στο βασικό περιβάλλον της εφαρμογής. Το περιβάλλον αυτό αποτελείται από τέσσερα βασικά τμήματα (components) τον header στην κορυφή της σελίδας, το footer στο κάτω μέρος της σελίδας και το ενδιάμεσο τμήμα χωρίζεται σε δύο υπο-τμήματα, αριστερά ένα navigation menu και δεξιά είναι το κυρίως περιεχόμενο. Το στοιχεία του navigation menu προσαρμόζονται ανάλογα με τον τύπο του χρήστη καθώς ο κάθε τύπος έχει διαφορετικά δικαιώματα και δυνατότητες, ενώ το κυρίως περιεχόμενο είναι δυναμικό και φορτώνεται μέσα από τα views της εφαρμογής ανάλογα με την ενέργεια που επιλέγει ο εκάστοτε χρήστης. Επιπροσθέτως, στην δεξιά πλευρά του header εμφανίζεται το όνομα του εκάστοτε χρήστη έχοντας ένα hover event ώστε όταν ο χρήστης μετακινήσει το ποντίκι πάνω του εμφανίζεται ένα αναπτυσσόμενο μενού με την ενέργεια αποσύνδεσης από την εφαρμογή. Κάθε αντικείμενο που βρίσκεται στη λίστα του πλαϊνού menu, περιέχει ένα redirect event το οποίο το διαχειρίζεται μια μέθοδος του πλαισίου προγραμματισμού Laravel, ονόματι route(). Η μέθοδος αυτή δέχεται σαν παράμετρο το όνομα μιας διαδρομής και όταν ο χρήστης κάνει κλικ ψάχνει στην διαθέσιμη λίστα με τις διαδρομές προκειμένου να βρει αυτή που ζήτησε ο χρήστης. Μόλις αυτή η διαδρομή βρεθεί στέλνεται σε συγκεκριμένη μέθοδο ενός controller ώστε να γίνουν οι απαραίτητες ενέργειες και να επιστρέψει στον χρήστη το αντίστοιχο view. Η ενεργή επιλογή από το menu ξεχωρίζει με ένα πράσινο χρώμα στο background ώστε να γνωρίζει ο χρήστης ανά πάσα στιγμή σε ποια ενότητα βρίσκεται.



Σχήμα 5.4: Γραφικό περιβάλλον εφαρμογής μετά από είσοδο χρήστη

4.5 Ρόλοι χρηστών

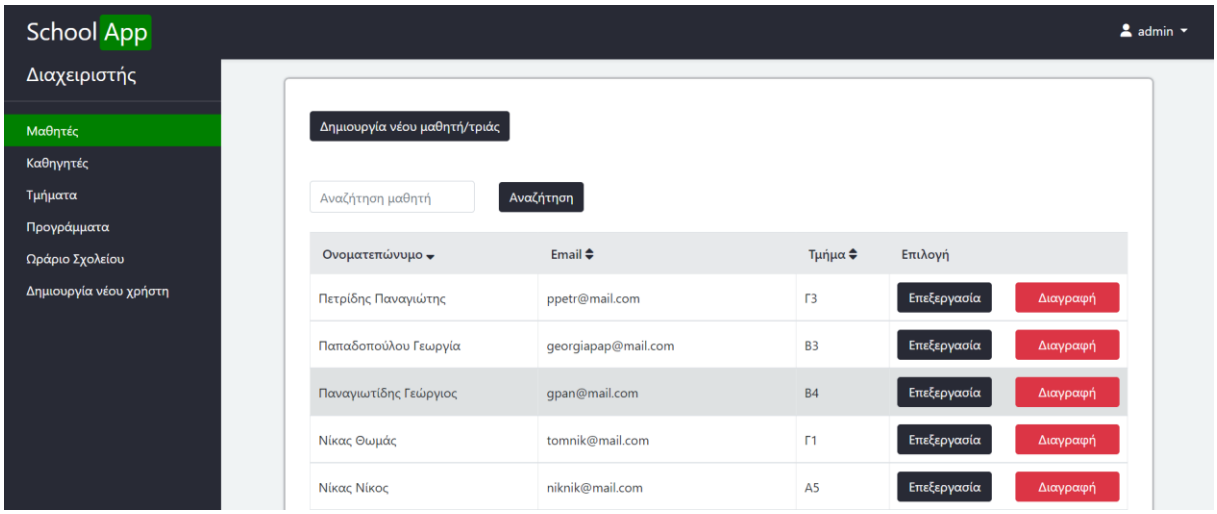
Όπως αναφέρθηκε και παραπάνω, υπάρχουν τρείς τύποι χρηστών που υποστηρίζει η εφαρμογή, ο χρήστης τύπου διαχειριστής, ο χρήστης τύπου καθηγητής και ο χρήστης τύπου μαθητής. Ας αναλύσουμε σε αυτό το σημείο αυτούς τους τρείς τύπους χρηστών.

4.5.1 Χρήστης τύπου διαχειριστής

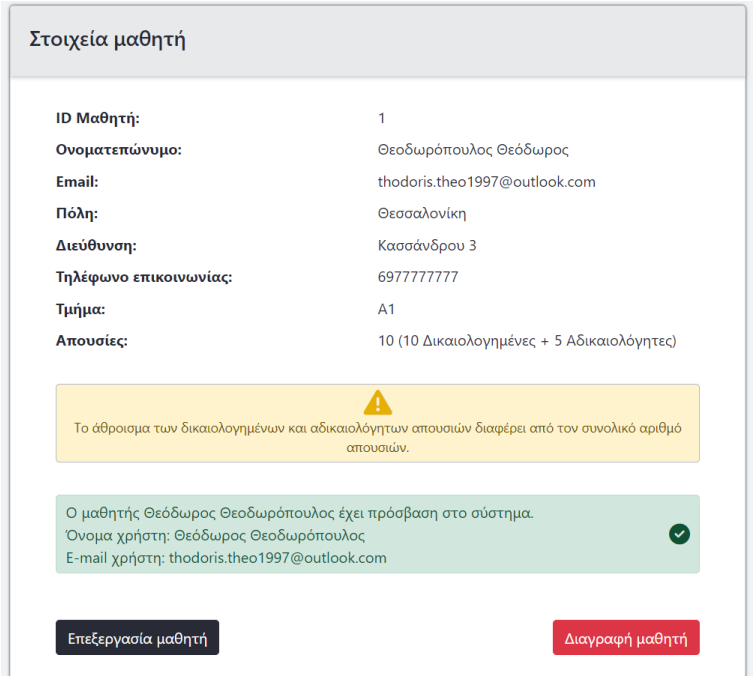
Ο χρήστης διαχειριστής είναι ο χρήστης με τις πιο πολλές δυνατότητες στην εφαρμογή και έχει πρόσβαση σχεδόν σε όλα τα δεδομένα. Πρακτικά αυτός διαχειρίζεται τις πληροφορίες των καθηγητών, των μαθητών αλλά και του σχολείου όπως προγράμματα σχολικών τμημάτων και ωράριο σχολείου. Η λίστα επιλογών που του παρέχονται μέσω του menu είναι μαθητές, καθηγητές, τμήματα, προγράμματα, ωράριο σχολείου και δημιουργία νέου χρήστη.

Επιλέγοντας την προβολή μαθητές, φορτώνεται ένα καινούργιο view το οποίο περιέχει την λίστα όλων των διαθέσιμων μαθητών τοποθετημένη σε έναν πίνακα περιλαμβάνοντας ονοματεπώνυμο, email, σχολικό τμήμα και τις ενέργειες επεξεργασία/διαγραφή. Ο πίνακας εξ ορισμού είναι ταξινομημένος με αύξουσα χρονολογική σειρά όμως, ο χρήστης μπορεί να κάνει εκ νέου ταξινόμηση με βάση τις στήλες ονοματεπώνυμο, email και σχολικό τμήμα, σε αύξουσα ή σε φθίνουσα σειρά. Στην στήλη ονοματεπώνυμο η ταξινόμηση γίνεται με βάση το επίθετο του μαθητή. Ο πίνακας υποστηρίζει την εμφάνιση συγκεκριμένου αριθμού γραμμών και όταν αυτός ο αριθμός ξεπεραστεί αξιοποιείται η σελιδοποίηση, μια λειτουργεία που επιτρέπει στον χρήστη να πάει σε προηγούμενη/επόμενη σελίδα του πίνακα ώστε να δει επιπλέον εγγραφές. Επιπλέον, πάνω από τον πίνακα υπάρχει ένα text input πεδίο μέσω του οποίου ο χρήστης μπορεί να αναζητήσει έναν μαθητή εισάγοντας το ονοματεπώνυμο του. Η αναζήτηση επιστρέφει τους μαθητές που ταιριάζουν στο λεκτικό που πληκτρολόγησε ο χρήστης και αν δεν υπάρχει καμία αντιστοίχιση επιστρέφει αντίστοιχο μήνυμα. Επίσης, μέσα στην οθόνη αυτή παρέχονται στον χρήστη οι δυνατότητες επεξεργασίας ή διαγραφής ενός υπάρχοντος μαθητή καθώς και η δημιουργία ενός νέου. Πατώντας την διαγραφή εμφανίζεται ένα modal επιβεβαίωσης ώστε ο χρήστης να είναι σίγουρος πριν κάνει την τελική ενέργεια. Το modal είναι πρακτικά ένα popup περιεχόμενο το οποίο εμφανίζεται πάνω από το ήδη υπάρχον περιεχόμενο της σελίδας και απενεργοποιεί το περιεχόμενο της βασικής σελίδας. Για να βγει ο χρήστης από ένα modal της εφαρμογής πρέπει να πατήσει το κουμπί έξοδος ή το κουμπί ‘Χ’ ή να πατήσει κλικ εκτός του περιεχομένου του modal. Την ενέργεια της διαγραφής διαχειρίζεται η μέθοδος destroy του StudentsController. Πατώντας επεξεργασία ή δημιουργία νέου μεταφέρεται στις αντίστοιχες οθόνες όπου υπάρχει η φόρμα συμπλήρωσης στοιχείων του μαθητή. Να αναφερθεί ότι πατώντας ο χρήστης στο όνομα του μαθητή οδηγείται στην οθόνη προβολής των στοιχείων του μαθητή και αντίστοιχα εάν πατήσει πάνω στο τμήμα μεταφέρεται στην οθόνη προβολής των πληροφοριών του συγκεκριμένου τμήματος. Στις οθόνες της δημιουργίας/επεξεργασίας ενός μαθητή όπως αναφέρθηκε υπάρχει μια φόρμα η οποία είναι ίδια εμφανισιακά, καθώς η εφαρμογή πρέπει να έχει συνοχή, όμως διαφέρουν σε δύο σημεία. Η φόρμα επεξεργασίας εμφανίζει τις πληροφορίες του μαθητή ενώ στη φόρμα δημιουργίας τα πεδία είναι κενά καθώς και ότι διαχειρίζονται η κάθε μια σε ξεχωριστή μέθοδο του StudentsController. Οι μέθοδοι update και store διαχειρίζονται αντίστοιχα τις δύο αυτές φόρμες. Στις φόρμες αυτές ενδεικτικά πεδία είναι το ονοματεπώνυμο, η διεύθυνση, οι απουσίες και το σχολικό τμήμα. Σημαντική λεπτομέρεια είναι πως αν το τμήμα ανήκει την δευτέρα ή τρίτη λυκείου εμφανίζεται ακόμη ένα πεδίο για την επιλογή της ομάδας προσανατολισμού του μαθητή. Η τελευταία οθόνη που υπάρχει για τους μαθητές είναι η οθόνη προβολής των στοιχείων τους. Έχει σαν περιεχόμενο μια καρτέλα με τις πληροφορίες του κάθε μαθητή

ενώ υπάρχει μήνυμα προς τον χρήστη για το αν ο συγκεκριμένος μαθητής έχει ή όχι πρόσβαση στην εφαρμογή. Αυτό παρέχεται σαν ένας έξτρα έλεγχος σε περίπτωση παραπόνων του μαθητή ώστε να αποφευχθεί η σπατάλη χρόνου εύρεσης αυτής της πληροφορίας. Να σημειωθεί ότι το παραπάνω μήνυμα δεν αναφέρεται στην περίπτωση που ο μαθητής χάσει τα στοιχεία πρόσβασης αλλά για το αν υπάρχει σαν χρήστης εγγεγραμμένος στον πίνακα users της βάσης δεδομένων. Ένα δεύτερο μήνυμα-προειδοποίηση εμφανίζεται όταν τα άθροισμα των δικαιολογημένων και αδικαιολόγητων απουσιών του μαθητή δεν ισούται με τον συνολικό αριθμό απουσιών. Επίσης αυτό παρέχεται για τον καλύτερο έλεγχο ώστε να διασφαλιστεί η ορθότητα των δεδομένων. Τέλος, στο κάτω μέρος της κάρτας υπάρχουν τρία κουμπιά για την επεξεργασία του μαθητή, την διαγραφή και για την μετάβαση στην λίστα των μαθητών. Στην περίπτωση δημιουργίας νέου μαθητή μετά την υποβολή ο χρήστης μεταφέρεται στην οθόνη εγγραφής χρήστη. Αυτό γίνεται γιατί την στιγμή δημιουργίας του μαθητή ο StudentsController δημιουργεί και αποθηκεύει έναν νέο μαθητή στον πίνακα students της βάσης δεδομένων, όμως αυτός ο μαθητής δεν θεωρείται έγκυρος χρήστης εφόσον δεν έχει αποθηκευτεί ακόμα στο πίνακα users. Στην οθόνη εγγραφής υπάρχει μια φόρμα με πεδία ονοματεπώνυμο, email, κωδικός πρόσβασης, επαλήθευση κωδικού πρόσβασης και τύπος χρήστη (μαθητής ή καθηγητής). Τα δύο πρώτα πεδία, ονοματεπώνυμο και email, τα έχει ήδη συμπληρωμένα και αναφέρονται στα δεδομένα που εισήγαγε ο χρήστης στο προηγούμενο βήμα. Συμπληρώνοντας και τα υπόλοιπα και κάνοντας υποβολή του νέου χρήστη δημιουργείται η αντίστοιχη εγγραφή στον πίνακα users και πλέον ο νέος μαθητής έχει πρόσβαση στην εφαρμογή. Να σημειωθεί πως για τον καινούργιο χρήστη θα πρέπει να του δοθούν τα στοιχεία πρόσβασης από τον φορέα προκειμένου να μπορεί να χρησιμοποιήσει την εφαρμογή.

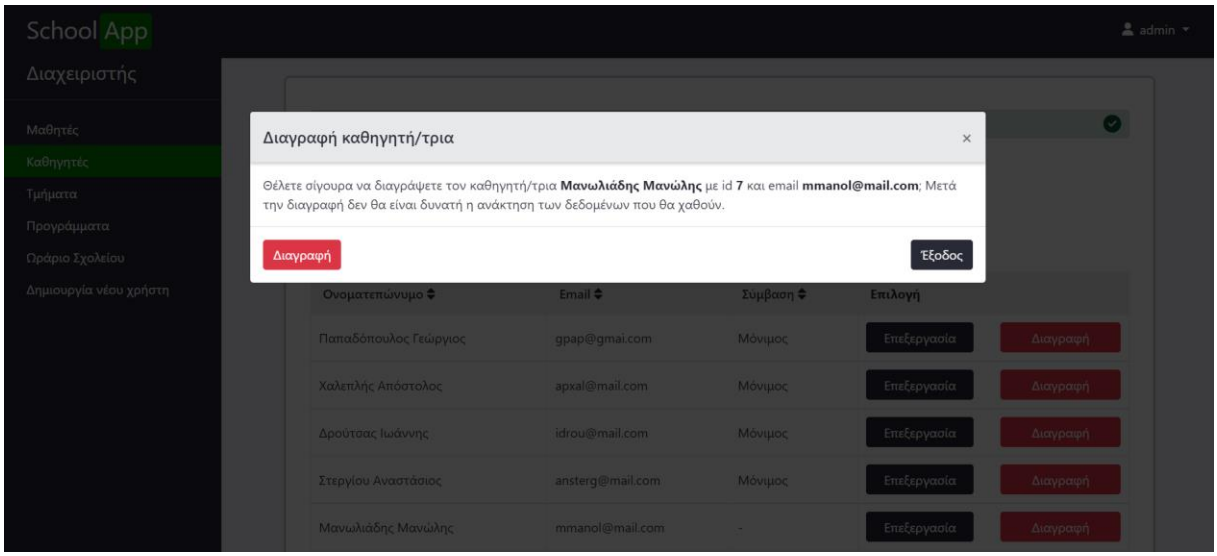


Σχήμα 5.5: Οθόνη λίστας μαθητών



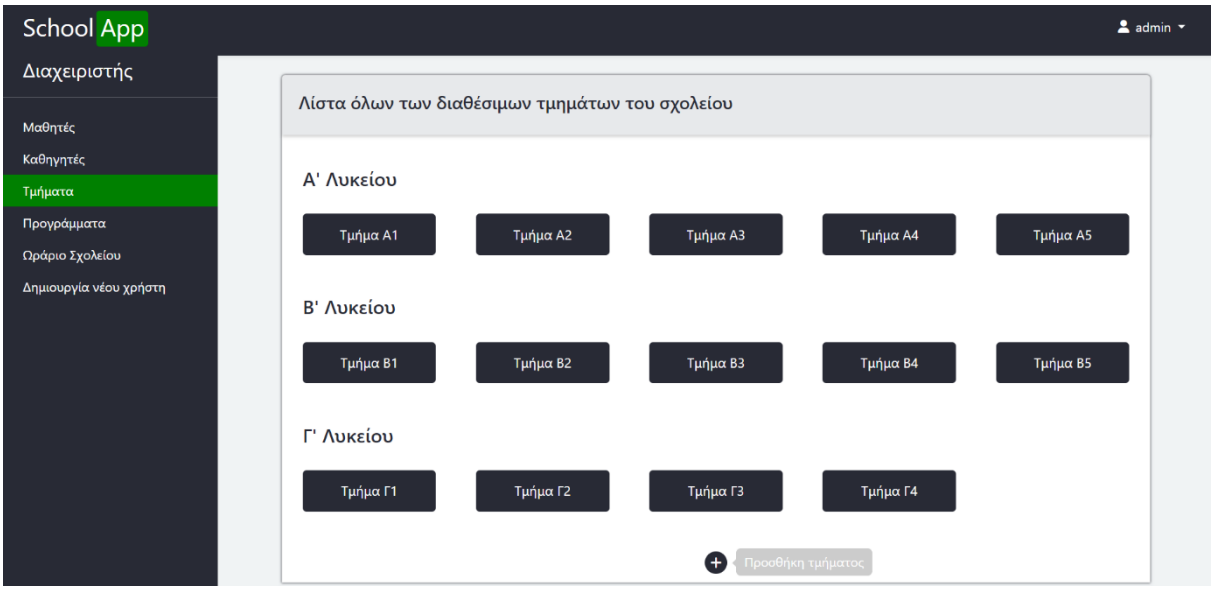
Σχήμα 5.6: Οθόνη προβολής στοιχείων μαθητή

Η δεύτερη επιλογή που μπορεί να επιλέξει ο χρήστης διαχειριστής από το menu είναι οι καθηγητές. Οι οθόνες που μπορεί να οδηγηθεί ο χρήστης προκειμένου να διαχειριστεί τους καθηγητές είναι ίδιες με αυτές των μαθητών. Βεβαίως με διαφορετικά δεδομένα. Η πιο βασική διαφορά βρίσκεται στο τρόπο διαχείρισης των ενεργειών οι οποίες καταστρατηγούνται μέσα από τον TeachersController. Παρότι τα views είναι ίδια με αυτά των μαθητών εμφανισιακά, δεν προέρχονται από τα ιδιά αρχεία views. Στην περίπτωση δημιουργίας νέου καθηγητή η διαδικασία είναι παρόμοια με αυτή που ακολουθείται στους μαθητές με την διαφορά ότι όλοι οι έλεγχοι και η αποθήκευση των δεδομένων θα γίνουν από τον αντίστοιχο controller για τους καθηγητές.



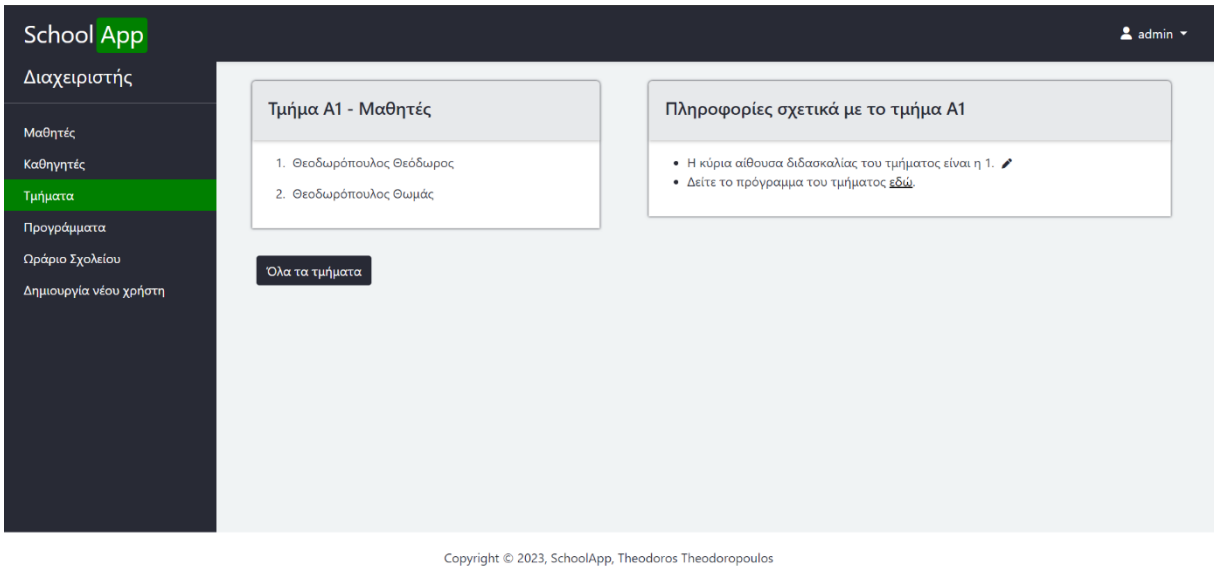
Σχήμα 5.7: Modal διαγραφής καθηγητή

Η τέταρτη επιλογή του χρήστη στο menu είναι τα σχολικά τμήματα. Πατώντας αυτήν την επιλογή ο χρήστης μεταφέρεται σε ένα άλλο view το οποίο περιέχει μια λίστα με όλα τα διαθέσιμα τμήματα του σχολείου. Τα τμήματα επιστρέφονται ανά χρονιά και ταξινομημένα. Κάθε τμήμα αναπαρίσταται από ένα κουτί και εφόσον ο χρήστης κάνει κλικ σε κάποιο τότε μεταφέρεται στο view που δείχνει τις λεπτομέρειες του τμήματος. Υπεύθυνος για αυτή την ενέργεια είναι ο SchoolClassesController. Κάτω από τα σχολικά τμήματα βρίσκεται ένα επιπλέον κουμπί που δίνει την δυνατότητα στον χρήστη να δημιουργήσει ένα σχολικό τμήμα. Πατώντας το εμφανίζεται modal με ένα select dropdown menu προκειμένου να επιλέξει σχολική χρονιά και ένα text input για την αίθουσα διδασκαλίας του τμήματος η οποία μπορεί να είναι μόνο αριθμός. Η δημιουργία νέου τμήματος γίνεται αυτόματα, ο χρήστης επιλέγει σχολική χρονιά ο controller βλέπει αυτό το αίτημα ψάχνει στην βάση να βρει το τελευταίο τμήμα της συγκεκριμένης σχολικής χρονιάς και προσθέτει το καινούργιο. Να επισημανθεί πως δυνατότητα διαγραφής έχουν μόνο τα τμήματα τα οποία δεν έχουν κανέναν εγγεγραμμένο μαθητή. Εφόσον η υποβολή για νέο τμήμα γίνει επιτυχώς ο χρήστης μεταφέρεται εκ νέου σε καινούργιο view προκειμένου να δημιουργήσει το σχολικό πρόγραμμα του καινούργιου τμήματος. Αυτό το βήμα είναι υποχρεωτικό και δεν μπορεί ο χρήστης να φύγει αν δεν υποβάλει το καινούργιο πρόγραμμα. Στην περίπτωση που δεν επιθυμεί να συμπληρώσει θα πρέπει να συμπληρώσει όλα τα πεδία με τον χαρακτήρα '-'. Τα κενά πεδία επιστρέφουν σφάλμα.



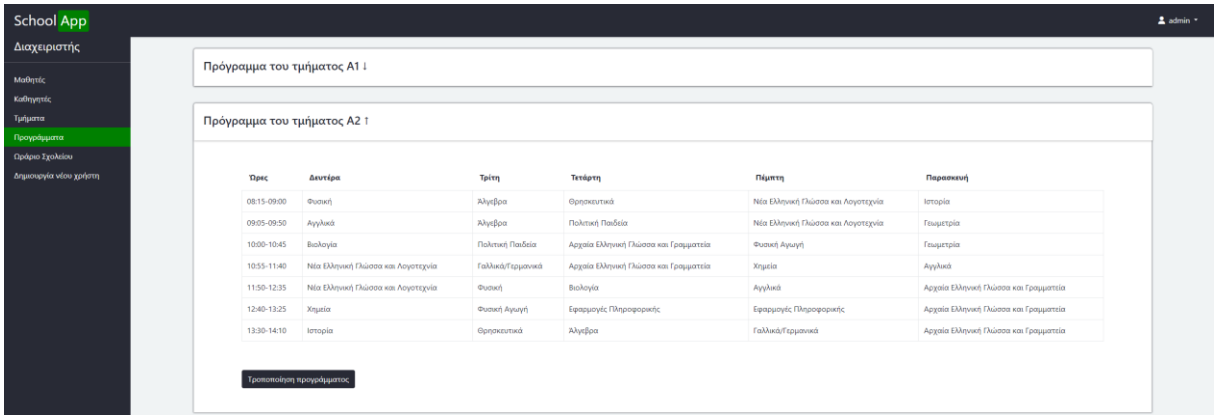
Σχήμα 5.8: Λίστα διαθέσιμων τμημάτων με hover tooltip στο add button

Πηγαίνοντας ο χρήστης στην οθόνη λεπτομερειών του τμήματος είναι σε θέση να δει δύο κάρτες με πληροφορίες. Στην πρώτη υπάρχει μια λίστα με όλους τους διαθέσιμους μαθητές του τμήματος ενώ στη δεύτερη μπορεί να δει πιο γενικές πληροφορίες για το τμήμα όπως το πρόγραμμα και η αίθουσα. Πατώντας στο όνομα κάποιου μαθητή είναι σε θέση να μεταφερθεί στην καρτέλα πληροφοριών του συγκεκριμένου μαθητή. Επίσης μπορεί να τροποποιήσει την αίθουσα διδασκαλίας του τμήματος καθώς και να ανοίξει ένα modal με το πρόγραμμα του τμήματος. Κάτω από τις δύο κάρτες πληροφοριών υπάρχει ένα κουμπί που οδηγεί τον χρήστη εκ νέου στην λίστα των διαθέσιμων σχολικών τμημάτων και στη περίπτωση που το τμήμα είναι κενό από μαθητές εμφανίζεται και το κουμπί της διαγράφης.



Σχήμα 5.9: Οθόνη πληροφορίες τμήματος

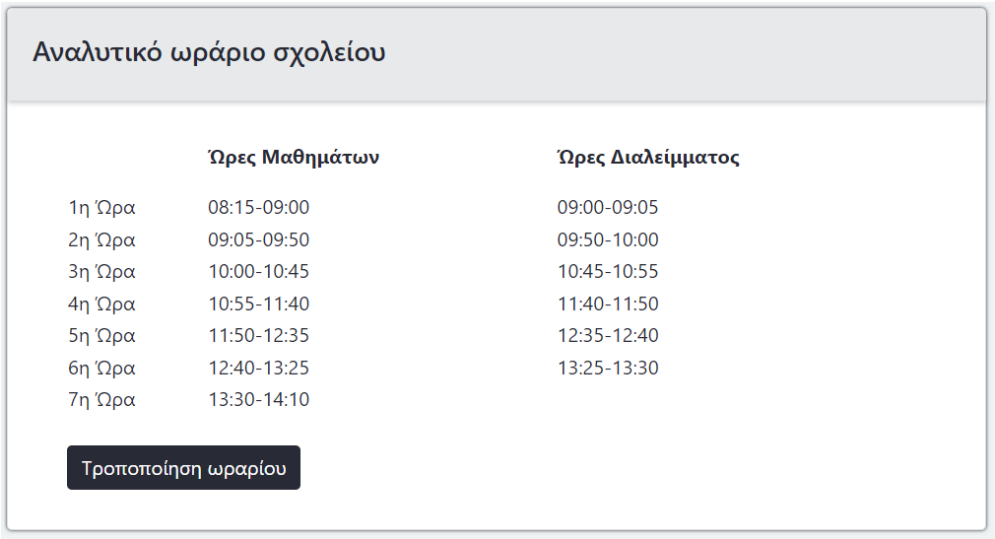
Η τέταρτη επιλογή που έχει διαθέσιμη ο χρήστης στο menu αφορά τα προγράμματα μαθημάτων όλων των τμημάτων. Επιλέγοντάς την, φορτώνεται ένα view το οποίο περιέχει μια λίστα με όλα τα διαθέσιμα προγράμματα με μορφή accordion. Το accordion είναι ένα custom component που αναπτύχθηκε χρησιμοποιώντας HTML, CSS και JavaScript. Υποστηρίζει την προβολή πολλών προγραμμάτων παράλληλα. Κάθε πρόγραμμα δίνει την δυνατότητα τροποποίησης του στον χρήστη. Επιλέγοντας την τροποποίηση επιστρέφεται το view επεξεργασίας προγράμματος ενός συγκεκριμένου τμήματος το οποίο αποτελείται από έναν πίνακα με text inputs και το κουμπί υποβολής που θα διαχειριστεί τις τιμές που ο χρήστης θα συμπληρώσει και ανάλογα με τους ελέγχους αυτό θα γίνει επιτυχώς ή ανεπιτυχώς.



Σχήμα 5.10: Accordion οθόνης σχολικών προγραμμάτων

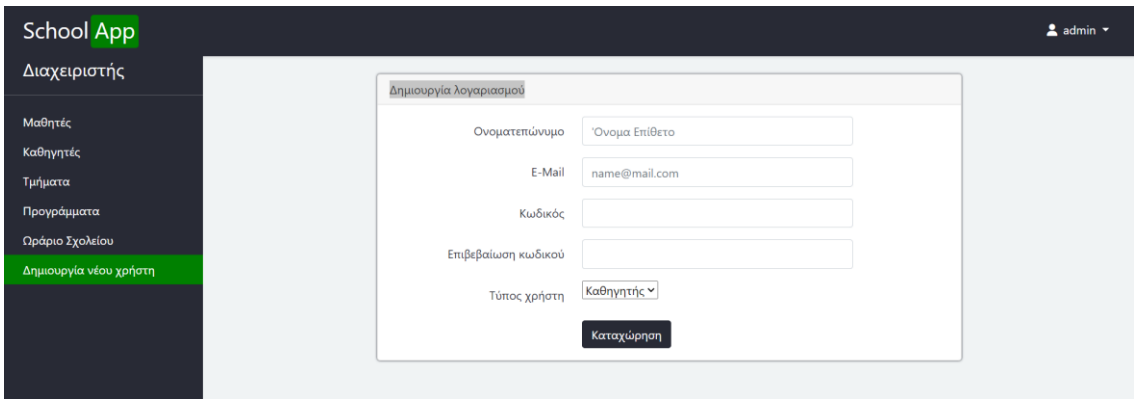
Η πέμπτη επιλογή από το menu αφορά το ωράριο του σχολείου. Κάνοντας κλικ σε αυτήν την επιλογή επιστρέφεται ένα view που περιέχει έναν πίνακα με τις ακριβείς ώρες μαθημάτων και διαλειμμάτων. Επίσης προσφέρεται στον χρήστη η ενέργεια τροποποίησης του ωραρίου από το αντίστοιχο κουμπί τροποποίησης που βρίσκεται κάτω από τον πίνακα. Πατώντας το κουμπί αυτό ο SchoolHoursController που διαχειρίζεται το ωράριο επιστρέφει ένα καινούργιο view το οποίο αποτελείται από μια φόρμα με text inputs, για τις ώρες μαθημάτων και διαλείμματος αντίστοιχα, με προκαθορισμένη τιμή την ήδη

υπάρχουσα. Από εκεί έχει τη δυνατότητα αποθήκευσης ή εξόδου από την επεξεργασία. Για την ενημέρωση των δεδομένων είναι υπεύθυνη η update μέθοδος του SchoolHoursController.



Σχήμα 5.11: Πίνακας αναλυτικού ωραρίου του σχολείου

Η τελευταία επιλογή που δίνεται στον χρήστη διαχειριστή είναι η δημιουργία νέου χρήστη. Πρακτικά αυτή η δημιουργία και εγγραφή χρήστη διαφέρει από την αντίστοιχη που γίνεται μετά την δημιουργία ενός μαθητή ή καθηγητή. Πρώτα απ' όλα η φόρμα αποτελείται από τα πεδία που είναι απαραίτητα για την δημιουργία ενός χρήστη της εφαρμογής και όχι ενός χρήστη συγκεκριμένου τύπου. Για παράδειγμα αποτελείται από ονοματεπώνυμο, email, κωδικό και τον τύπο του χρήστη αλλά δεν περιλαμβάνονται πληροφορίες όπως διεύθυνση, απουσίες, αριθμός τηλεφώνου και άλλα. Με την υποβολή της φόρμας, ο νέος χρήστης που καταχωρείται στο πίνακα users της βάσης δεδομένων καταχωρείται και στον αντίστοιχο πίνακα με βάση τον τύπο που έχει επιλεχθεί (μαθητής ή καθηγητής) με την διαφορά ότι στα υπόλοιπα πεδία δεν θα αποθηκευτεί κάποια τιμή. Αυτή η ενέργεια δίνεται στον διαχειριστή για την περίπτωση που επιθυμεί να δημιουργήσει έναν χρήστη χωρίς δεδομένα και να τα προσθέσει σε δεύτερο χρόνο μέσω της ενέργειας της επεξεργασίας. Η συγκεκριμένη φόρμα διαχειρίζεται από τον RegisterController της εφαρμογής.



Σχήμα 5.12: Οθόνη δημιουργίας νέου χρήστη

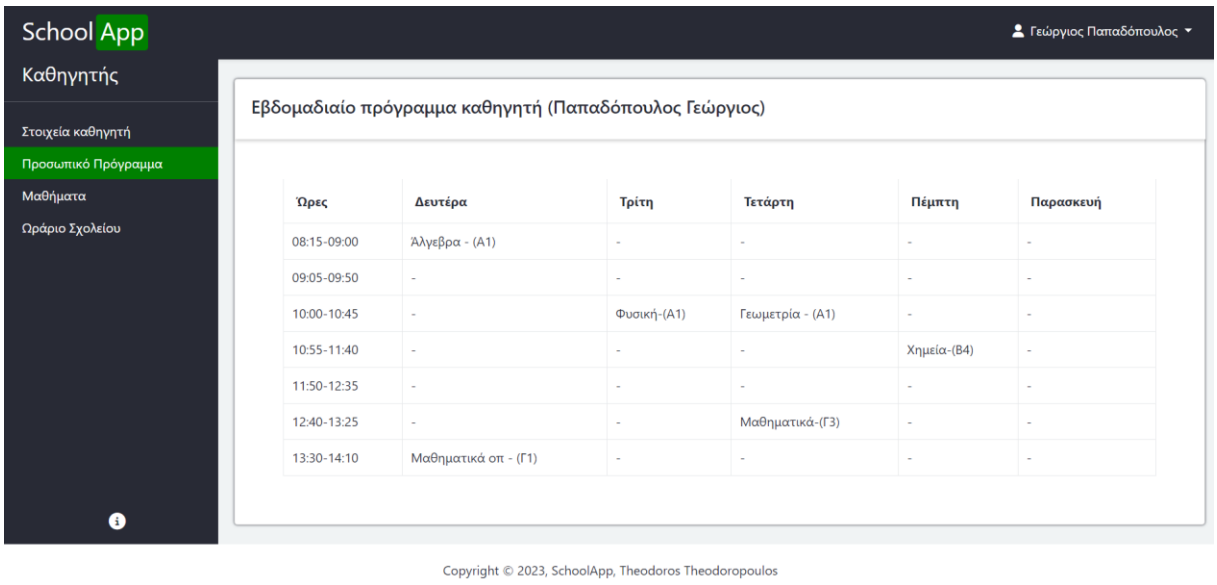
Συνοψίζοντας, ο χρήστης τύπου διαχειριστής είναι ο χρήστης με τα περισσότερα δικαιώματα και τις περισσότερες διαθέσιμες ενέργειες στην εφαρμογή. Μπορεί να δημιουργήσει, διαγράψει και να επεξεργαστεί μαθητές και καθηγητές, να διαχειριστεί τα σχολικά τμήματα και προγράμματά τους καθώς έχει τη δυνατότητα να επεξεργαστεί και το ωράριο του σχολείου.

4.5.2 Χρήστης τύπου καθηγητής

Ο επόμενος τύπος χρήστη μετά τον διαχειριστή, είναι ο τύπος του καθηγητή. Εφόσον ένας χρήστης τέτοιου τύπου εξουσιοδοτηθεί από την εφαρμογή μπορεί να έχει πρόσβαση σε προσωπικά του δεδομένα όπως τα στοιχεία του και το προσωπικό του εβδομαδιαίο πρόγραμμα αλλά δεν έχει δικαίωμα μεταβολής των αντίστοιχων δεδομένων. Πιο αναλυτικά, οι επιλογές που έχει από το menu είναι τα προσωπικά του στοιχεία, το προσωπικό του πρόγραμμα, τα μαθήματα που διδάσκει και το αναλυτικό ωράριο του σχολείου.

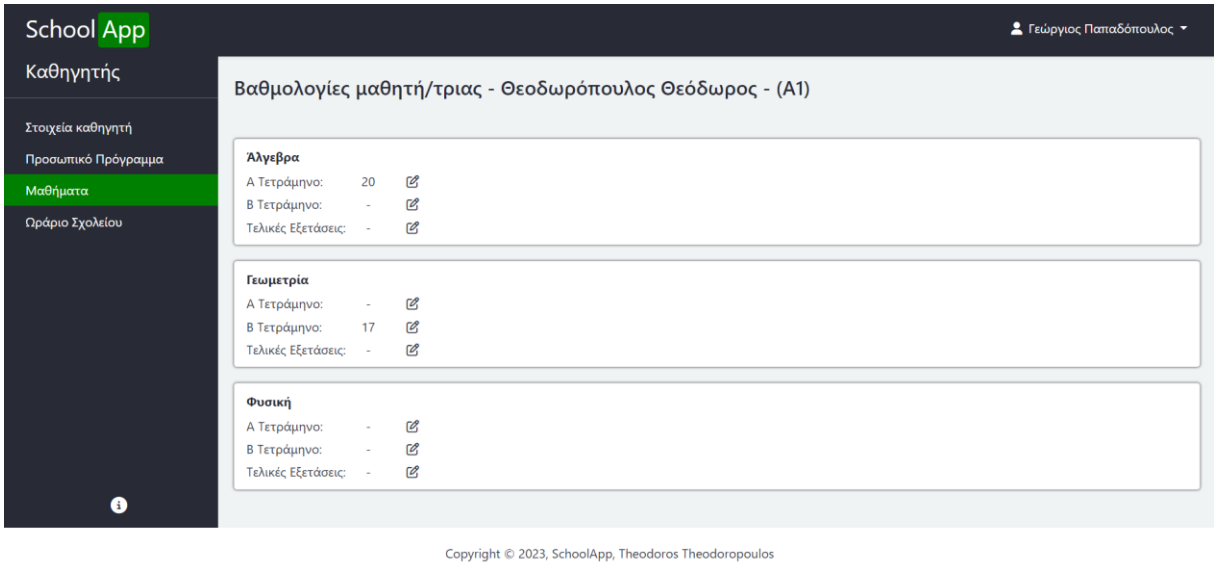
Επιλέγοντας τα στοιχεία του καθηγητή από το menu επιστρέφεται ένα view το οποίο περιέχει μια καρτέλα με όλες τις πληροφορίες του τρέχοντος χρήστη-καθηγητή. Δεν του παρέχεται καμία διαθέσιμη ενέργεια μεταβολής των δεδομένων στην οθόνη αυτή, αποτελώντας την κύρια διαφορά με το αντίστοιχο view που επιστρέφεται στον διαχειριστή. Σε αυτήν την οθόνη υπάρχει και συντόμευση για το προσωπικό εβδομαδιαίο πρόγραμμα του χρήστη που βρίσκεται στο αντίστοιχο πεδίο της κάρτας. Επιλέγοντας το κουμπί «Άνοιγμα» εμφανίζεται το πρόγραμμα με μορφή modal στην οθόνη.

Το επόμενο αντικείμενο στο menu είναι το προσωπικό πρόγραμμα. Σε αντίθεση με το προηγούμενο εβδομαδιαίο πρόγραμμα που ανοίγει σε modal component, εδώ ο TeacherController επιστρέφει ένα καινούργιο view το οποίο περιέχει έναν πίνακα με το πρόγραμμα. Και σε αυτήν την περίπτωση ο χρήστης δεν έχει δικαίωμα μεταβολής/τροποποίησης των δεδομένων αλλά μόνο δικαίωμα προβολής.



Σχήμα 5.13: Οθόνη εβδομαδιαίου προγράμματος καθηγητή

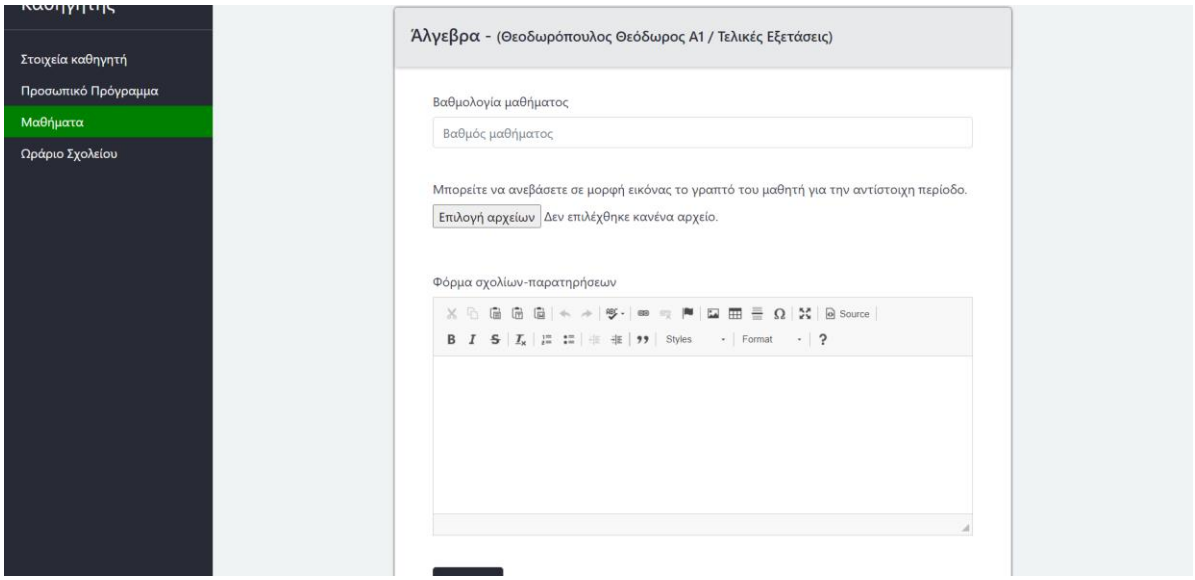
Το επόμενο στοιχείο πλοήγησης στο menu αφορά τα μαθήματα. Αυτή η επιλογή επιστρέφει στον χρήστη ένα view με μια ταξινομημένη λίστα των μαθημάτων που διδάσκει καθώς και το τμήμα στο οποίο διδάσκει το κάθε μάθημα. Η ταξινόμηση είναι αύξουσα με βάση το σχολικό τμήμα. Κάθε σχολικό τμήμα εκτός της σημασιολογικής του σημασίας, δίνει την δυνατότητα στον χρήστη να μεταφερθεί στην οθόνη πληροφοριών του τμήματος κάνοντας κλικ επάνω του. Φορτώνοντας το view με τις πληροφορίες του σχολικού τμήματος ο καθηγητής είναι σε θέσει να δει δύο καρτέλες, η πρώτη περιέχει την λίστα όλων των μαθητών του τμήματος ενώ η δεύτερη περιέχει γενικές πληροφορίες όπως η αίθουσα διδασκαλίας και το εβδομαδιαίο πρόγραμμα. Το εβδομαδιαίο πρόγραμμα ανοίγει και εδώ με την μορφή modal. Πολύ σημαντική λειτουργία για έναν καθηγητή είναι να μπορεί να επεξεργαστεί τις πληροφορίες των μαθητών στα τμήματα που διδάσκει, όσον αφορά τα μαθήματα που διδάσκει. Πρακτικά αυτό σημαίνει ότι ο καθηγητής μπορεί να τροποποιήσει πληροφορίες του μαθητή όπως η βαθμολογία ενός μαθήματος ανά τετράμηνο και τις τελικές εξετάσεις. Σε καμία περίπτωση όμως δεν μπορεί να δει ούτε να επεξεργαστεί προσωπικά δεδομένα των μαθητών όπως διεύθυνση, απουσίες κ.α. Δίπλα σε κάθε μαθητή της λίστας υπάρχει ένα εικονίδιο βιβλίο μέσω του οποίου ο καθηγητής μπορεί να μεταφερθεί στην οθόνη με όλα τα διαθέσιμα μαθήματα που αυτός διδάσκει στο συγκεκριμένο τμήμα για τον συγκεκριμένο μαθητή που επέλεξε. Το view που επιστρέφει το περιεχόμενο αυτής της οθόνης διαχειρίζεται εξ' ολοκλήρου από τον GradesController. Τα διαθέσιμα μαθήματα εμφανίζονται μέσα σε πλαίσια, όπου ένα μάθημα αποτελεί ένα πλαίσιο, και μέσα στο πλαίσιο αναφέρονται οι βαθμοί των τετράμηνων και της τελικής εξέτασης. Εάν ο βαθμός δεν υπάρχει εμφανίζεται ο χαρακτήρας '-'. Επίσης, δίπλα από τους βαθμούς υπάρχει το εικονίδιο επεξεργασίας από όπου ο καθηγητής μπορεί να οδηγηθεί σε νέα οθόνη για να επεξεργαστεί τους βαθμούς της αντίστοιχης περιόδου. Θα γίνει αναφορά λεπτομερείς σε αυτήν την οθόνη στη συνέχεια.



Σχήμα 5.14: Οθόνη διαθέσιμων μαθημάτων ενός καθηγητή προς συγκεκριμένο μαθητή

Πατώντας την επεξεργασία μιας συγκεκριμένης βαθμολογίας φορτώνεται ένα view το οποίο περιέχει μια φόρμα με τρία πεδία. Το πρώτο είναι ένα text input για την εισαγωγή βαθμολογίας αποκλειστικά ακέραιου αριθμού, το δεύτερο είναι ένα file input και τρίτο είναι ένα textarea input. Το file input δέχεται μόνο αρχεία εικόνες και εξυπηρετεί τον σκοπό ο καθηγητής να μπορεί να ανεβάσει ένα ή περισσότερα σκαναρισμένα γραπτά δίνοντας τη δυνατότητα στον μαθητή να μπορεί να δει διορθώσεις και λάθη στα οποία έχει υποπέσει. Επίσης, έχει την δυνατότητα να διαγράψει κάποια από τις ανεβασμένες αυτές φωτογραφίες καθώς και να τις δει μέσα από ένα open dialog box πατώντας πάνω τους. Τέλος, το textarea

input δίνει την δυνατότητα στον καθηγητή να προσθέσει σχόλια-παρατηρήσεις προς τον μαθητή. Στο textarea input έχει προστεθεί και ένα component ckEditor προκειμένου να παρέχει πιο πολλές δυνατότητες για τα σχόλια του χρήστη όπως να βάζει επικεφαλίδες, να βάζει bold ένα σημαντικό τμήμα των σχολίων, να προσθέσει στο κείμενο ένα σύνδεσμο προς κάποια πηγή αναφοράς και πολλά άλλα. Την υποβολή αυτής της φόρμας διαχειρίζεται η μέθοδος store GradesController.



Σχήμα 5.15: Οθόνη επεξεργασίας βαθμολογίας

Το τελευταίο στοιχείο πλοήγησης του χρήση καθηγητή από το menu είναι το ωρολόγιο πρόγραμμα του σχολείου. Στην οθόνη που επιστρέφεται ο υπάρχει αναλυτικός πίνακας των ωρών διδασκαλίας και διαλειμμάτων από όπου ο χρήστης μπορεί να ενημερωθεί. Δεν μπορεί να μεταβάλει τα δεδομένα.

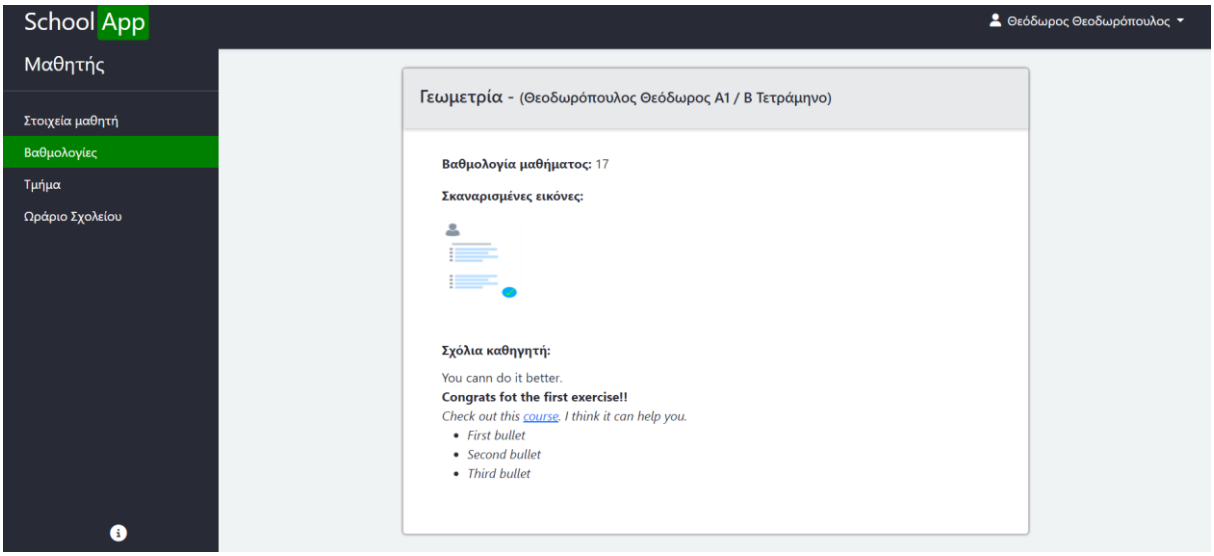
4.5.3 Χρήστης τύπου μαθητής

Ο τελευταίος τύπος χρήστη που έχει πρόσβαση στο σύστημα είναι ο μαθητής. Είναι ο χρήστης με τις λιγότερες δυνατότητες καθώς δεν έχει κανένα δικαίωμα μεταβολής δεδομένων. Από το menu πλοήγησης μπορείς να επιλέξει τις προσωπικές του πληροφορίες, τις βαθμολογίες των μαθημάτων του, πληροφορίες σχετικά με το σχολικό τμήμα στο οποίο εντάσσεται καθώς και το σχολικό ωράριο.

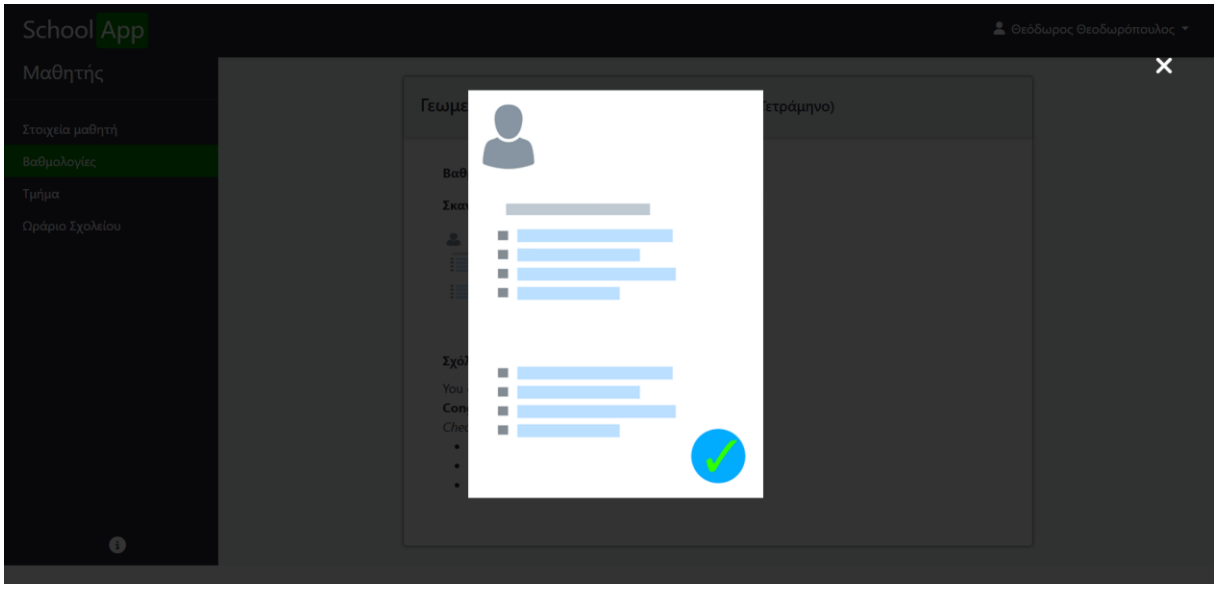
Επιλέγοντας να δει τα προσωπικά του στοιχεία επιστρέφεται το view που περιέχει την καρτέλα πηλοφοριών ενός μαθητή, Σχήμα 5.5, η βασική διαφορά όμως είναι πως ο μαθητής δεν έχει το δικαίωμα να τα τροποποιήσει κανένα από τα δεδομένα. Η ίδια λογική ισχύει και για το τμήμα του μαθητή και για το σχολικό ωράριο, τα οποία δεν θα αναλύσουμε περισσότερο καθώς έχει ήδη γίνει στις προηγούμενες ενότητες.

Πέρα από τις επιλογές που αναφέρθηκαν ο μαθητής μπορεί να δει και την αναλυτική του βαθμολογία σε όλα τα μαθήματα που παρακολουθεί από το στοιχείο πλοήγησης βαθμολογίες. Κάνοντας κλικ σε αυτήν την επιλογή επιστρέφεται ένα view που περιέχει όλη τη λίστα των μαθημάτων που διδάσκεται ο μαθητής με βάση το τμήμα και την ομάδα προσανατολισμού (εφόσον πρόκειται για μαθητή της β ή γ λυκείου) που ανήκει. Οι καρτέλες των μαθημάτων είναι αντίστοιχες με αυτές που φαίνονται στην οθόνη βαθμολογιών του καθηγητή, σχήμα 5.13, όμως στη θέση του εικονιδίου επεξεργασίας βρίσκεται

εικονίδιο για την προβολή των λεπτομερειών ενός μαθήματος. Εφόσον λοιπόν ο χρήστης επιλέξει το εικονίδιο ο GradesController επεξεργάζεται το αίτημα για να τον μεταφέρει ή όχι στην οθόνη προβολής τον λεπτομερειών. Εκεί μπορεί να δει τον βαθμό, σκαναρισμένες εικόνες από τα γραπτά του και πιο κάτω ένα πλαίσιο με σχόλια παρατηρήσεις από τον καθηγητή. Οι εικόνες έχουν εξ' ορισμού συγκεκριμένο ύψος και πλάτος ώστε η οθόνη να έχει συνοχή και να μην χαλάει η ομοιομορφία της. Ωστόσο, δίνεται η επιλογή στον μαθητή να επιλέξει μια από τις εικόνες και να ανοίξει η εικόνα σε μεγαλύτερο μέγεθος μέσα σε ένα custom open dialog box με σκοπό να μπορέσει να έχει καλύτερη ανάγνωση του γραπτού. Πατώντας το κουμπί 'X' ή κάνοντας κλικ κάπου εκτός εικόνας ο χρήστης μπορεί να κλείσει το open dialog box και να επιστρέψει στην καρτέλα λεπτομερειών του μαθήματος.



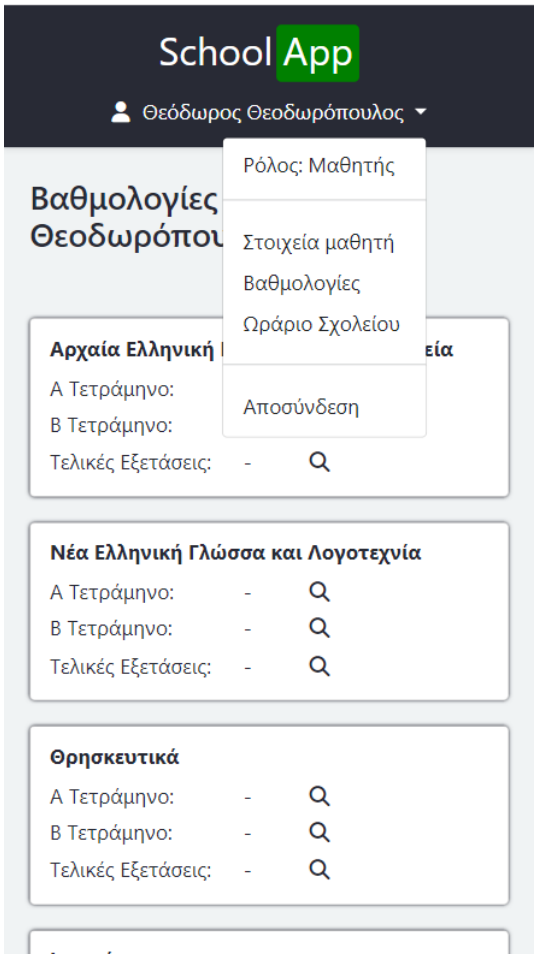
Σχήμα 5.16: Οθόνη αναλυτικής βαθμολογίας μαθήματος μαθητή



Σχήμα 5.17: Open dialog box της εικόνας του σκαναρισμένου γραπτού

4.6 Προσαρμόσιμο περιεχόμενο

Οι περισσότερες διαδικτυακές εφαρμογές δίνουν την δυνατότητα στο περιεχόμενο να αλλάζει σχηματισμό βάση των διαστάσεων της οθόνης ή και της ανάλυσής της. Αυτό συμβαίνει με αποκλειστικό σκοπό να μπορεί ο χρήστης να αλληλοεπιδρά με την εφαρμογή από οποιαδήποτε συσκευή έχει διαθέσιμη και η οποία υποστηρίζει πλοήγηση μέσω ενός περιηγητή ιστού. Ακολουθώντας αυτήν την πρακτική, η συγκεκριμένη εφαρμογή υποστηρίζεται σε όλες τις συσκευές, επιτραπέζιος υπολογιστής, laptop, tablet και έξυπνο τηλέφωνο, δίνοντας στον χρήστη την ευκαιρία και την ελευθερία να μπορεί να την προσπελάσει από οποιαδήποτε συσκευή έχει στην κατοχή του. Η λειτουργικότητα αυτή επιτυγχάνεται μέσω των media queries τα οποία παρέχονται από την τεχνολογία CSS που είναι υπεύθυνη για την εμφάνιση του περιεχομένου. Τα media queries μπορούν να στοχεύσουν στην οθόνη και να ελέγχουν το ελάχιστο ή μέγιστο πλάτος και ύψος και ανάλογα να προσαρμόζουν το περιεχόμενο. Δεν υπάρχει περιορισμός στην χρήση τους και μπορούν να στοχεύσουν σε πολλές τιμές του μεγέθους της οθόνης και για κάθε περίπτωση να εφαρμόζονται διαφορετικοί κανόνες CSS.



Σχήμα 5.18: Οθόνη εφαρμογής σε μικρότερες διαστάσεις (responsive web design)

4.7 Επίλογος

Συνοψίζοντας, η δομή της εφαρμογής παίζει πολύ σημαντικό ρόλο στην σωστή λειτουργία της, την συντήρηση και την επεκτασιμότητα της. Στο κεφάλαιο αυτό παρουσιάστηκαν όλα τα μέρη της εφαρμογής, σε επίπεδο χρήσης κυρίως. Επίσης έγινε αναφορά στους πιθανούς τύπους των χρηστών και στις δυνατότητες που παρέχεται στον κάθε έναν από αυτούς. Αρχικά, έγινε μια γενική περιγραφή της εφαρμογής και των λειτουργιών. Στη συνέχεια αναλύθηκε η διαδικασία εξουσιοδότησης του χρήστη από την εφαρμογή. Έπειτα έγινε αναφορά στο βασικό περιβάλλον της εφαρμογής το οποίο αποτελείται από components και εξηγήθηκε γιατί επιλέχθηκε η λογική των components καθώς και πως λειτουργεί η συγκεκριμένη λογική. Στη συνέχεια, αναφέρθηκαν και αναλύθηκαν όλοι οι πιθανοί τύποι χρηστών που εξουσιοδοτεί η εφαρμογή, διαχειριστής, καθηγητής, μαθητής, και έγινε αναφορά στις δυνατότητες τους καθώς και στο πως προσαρμόζεται το γραφικό περιβάλλον της εφαρμογής ανάλογα με τον τύπο χρήστη. Τέλος, έγινε αναφορά στην δυνατότητα της εφαρμογής να προσαρμόζει το περιεχόμενο ανάλογα με το μέγεθος της οθόνης ώστε να προσφέρει πρόσβαση στον χρήστη μέσω οποιασδήποτε πλατφόρμας.

Κεφάλαιο 5ο: Συμπεράσματα και προτάσεις βελτίωσης

5.1 Εισαγωγή

Παρά το γεγονός ότι η εφαρμογή προσφέρει μια λύση στην άμεση αλληλεπίδραση του ανθρώπινου δυναμικού ενός γενικού λυκείου με δεδομένα που το αφορούν υπάρχουν αρκετές βελτιώσεις και επεκτάσεις που θα μπορούσαν να γίνουν με στόχο την καλύτερη απόδοση της εφαρμογής και την δυνατότητα να παρέχουν στους χρήστες ακόμα περισσότερες λειτουργίες προς όφελός τους. Στο τελευταίο αυτό κεφάλαιο, θα γίνει αναφορά σε βελτιώσεις και μελλοντικές επεκτάσεις που θα μπορούσαν να γίνουν στην εφαρμογή με σκοπό να βελτιωθεί ακόμα περισσότερο.

5.2 Βελτιώσεις

Μια εφαρμογή όσο καλά και να είναι δομημένη πάντα χρίζει βελτιώσεων. Τέτοιες βελτιώσεις γίνονται πρωτίστως σε επίπεδο κώδικα. Ο κώδικας της εφαρμογής είναι πολύ προσεγμένος, με στοιχήσεις, σχόλια όπου κρίνεται απαραίτητο για την περιγραφή της λειτουργικότητας και με βασική πρακτική το να χρησιμοποιηθεί όσο το δυνατόν λιγότερος κώδικας για την υλοποίηση συγκεκριμένης ενέργειας. Ωστόσο κάνοντας συνεχείς ελέγχους και ανασκοπήσεις μπορεί να προκύψει κάποιο κομμάτι κώδικα το οποίο να μπορεί να γραφτεί και με πιο σύντομο τρόπο. Επιπλέον, διπλός κώδικας ή κώδικας ο οποίος υπάρχει αλλά δεν χρησιμεύει πρακτικά κάπου καθώς και η χρήση μεταβλητών που δεν υπάρχει λόγος να υπάρχει και να δεσμεύει μνήμη, είναι ανεπιθύμητα. Τέτοιου τύπου βελτιώσεις στοχεύουν στην καλύτερη και ταχύτερη απόδοση της εφαρμογής. Επίσης, ένα δεύτερο κομμάτι της εφαρμογής που θα μπορούσε να βελτιωθεί είναι το κομμάτι των component. Όπως έχει αναφερθεί, η υλοποίηση του frontend έγινε χρησιμοποιώντας την τεχνική των component, θα μπορούσαν όμως να ήταν περισσότερα πράγμα που θα έκανε ακόμα πιο εύκολη την συντήρηση και την επεκτασιμότητα της εφαρμογής.

5.3 Μελλοντικές επεκτάσεις

Το SchoolApp πετυχαίνει τον σκοπό του που είναι η δυνατότητα προβολής και τροποποίησης δυναμικών δεδομένων των μελών ενός γενικού λυκείου παρέχοντας αρκετές δυνατότητες και λειτουργείες στους χρήστες. Ωστόσο υπάρχουν κάποιες μελλοντικές επεκτάσεις που θα μπορούσαν να υλοποιηθούν αυξάνοντας έτσι τις λειτουργίες της εφαρμογής. Αρχικά θα μπορούσε να συνδεθεί με την εφαρμογή ένα σύστημα αποστολής email προς τους χρήστες ώστε να ειδοποιούνται όταν γίνεται μια αλλαγή που τους ενδιαφέρει. Για παράδειγμα όταν ένας καθηγητής προσθέσει μια βαθμολογία σε έναν μαθητή, να αποστέλνεται στον δεύτερο ένα email που να αναφέρει αυτή την αλλαγή. Με τον τρόπο αυτό οι χρήστες θα είναι ενήμεροι σε κάθε αλλαγή που γίνεται. Μια άλλη σημαντική επέκταση θα ήταν η υλοποίηση ενός πίνακα «ζωντανών» ανακοινώσεων. Υποστηρίζοντας η εφαρμογή μια τέτοια λειτουργία θα δίνει την δυνατότητα στους χρήστες να ενημερώνονται για γεγονότα, δράσεις και εκδηλώσεις που αφορούν το σχολείο καθώς και να ανατρέχουν σε αυτά ανά πάσα στιγμή. Επίσης, θα μπορούσε το σύστημα αυτό να υποστηρίζει και ανακοινώσεις καθηγητών, για παράδειγμα, ανακοίνωση ενημέρωσης πως μια συγκεκριμένη ημερομηνία θα λείπει και ως εκ τούτου δεν θα γίνουν τα μαθήματα που διδάσκει. Επιπροσθέτως, θα μπορούσε να υλοποιηθεί μια λειτουργία συλλογής στατιστικών στοιχείων ανά τμήμα, ανά χρονιά και για το σχολείο συνολικά. Αυτά τα στατιστικά θα αφορούν κυρίως τους μαθητές και θα γίνεται ανώνυμα. Με αυτό το τρόπο θα μπορεί να γίνει μια συλλογή δεδομένων

στοχεύοντας στην μελέτη τους και την εξαγωγή συμπερασμάτων για την λειτουργία του σχολείου και της αποδοτικότητας του ανθρώπινου δυναμικού. Θα είναι κατά κάποιο τρόπο ένα εργαλείο για αξιολόγηση βλέποντας και συγκρίνοντας αποτελέσματα. Η αξιολόγηση αυτή τελικά θα αφορά τη συνολική λειτουργεία και απόδοση ενός συγκεκριμένου σχολείου. Τέλος, θα μπορούσε να υλοποιηθεί ένας αλγόριθμος με σκοπό την αυτόματη δημιουργία σχολικού προγράμματος, απαλλάσσοντας έτσι τον αρμόδιο καθηγητή από αυτή την διαδικασία και να του εξοικονομήσει χρόνο για άλλες σχολικές υποχρεώσεις και δραστηριότητες.

5.4 Συμπεράσματα

Για κάποιον που θέλει να ξεκινήσει την ανάπτυξη δυναμικών διαδικτυακών εφαρμογών το πλαίσιο προγραμματισμού Laravel είναι σίγουρα η ιδανική τεχνολογία για να την αξιοποιήσει στο τμήμα του backend της εφαρμογής. Φαινομενικά μια δύσκολη και όχι ελκυστική τεχνολογία όμως ο χρήστης μπορεί να εξοικειωθεί γρήγορα και τότε οι δυνατότητες που έχει είναι πάρα πολλές. Πιο αναλυτικά:

1. Παρέχει μια πολύ καλή οργάνωση των αρχείων κάτι που βοηθάει στην κατανόηση ενός project και κρατάει «καθαρή» την λογική που χρησιμοποιείται στην εφαρμογή.
2. Δίνει την δυνατότητα χρήσης πολύ σημαντικών λειτουργειών με ελάχιστες γραμμές κώδικα οι οποίες για να δημιουργηθούν εξ' ολοκλήρου από τον προγραμματιστή είναι επίπονο και χρονοβόρο. Για παράδειγμα δημιουργία συστήματος εξουσιοδότησης, έλεγχος εξουσιοδοτημένου χρήστη κ.α.
3. Ταχύτερη διαδικασία ανάπτυξης, γεγονός που είναι πολύ σημαντικό για έναν προγραμματιστή καθώς τον βοηθάει να εξοικονομεί χρόνο για άλλες πτυχές της ανάπτυξης της εφαρμογής.
4. Καθιστά εύκολή διαδικασία την σύνδεση με την βάση δεδομένων καθώς και την αλληλεπίδραση μαζί της. Πρακτικά, δεν χρειάζεται καν η γνώση της γλώσσας SQL καθώς όλες οι ενέργειες μπορούν να γίνουν μέσα από το προγραμματιστικό περιβάλλον αξιοποιώντας μοντέλα και μεθόδους του ιδίου του Laravel.
5. Παρέχει συμβατότητα με το frontend τμήμα της εφαρμογής και των τεχνολογιών που το απαρτίζουν. Παρέχει ένα template engine ονόματι blade, με σκοπό να κάνει πιο εύκολο τον συνδυασμό του περιεχομένου HTML με τα δυναμικά δεδομένα που θα διαχειρίζονται από την εφαρμογή.
6. Μέσα στα αρχεία blade είναι δυνατή και η συγγραφή JavaScript και CSS κώδικα ή εισαγωγής τους από τρίτα αρχεία.
7. Παρέχει αρκετά επίπεδα ασφάλειας κάτι που είναι πολύ σημαντικό όταν αναφερόμαστε σε εφαρμογές με διαχείριση δυναμικών δεδομένων.
8. Παρέχει ένα command line ονόματι Artisan, το οποίο περιέχει πληθώρα επιλογών τις οποίες μπορεί ο προγραμματιστής να αξιοποιήσει προς διευκόλυνση του.

Αξιοποιώντας όλα τα παραπάνω η ανάπτυξη της εφαρμογής ήταν επιτυχημένη χωρίς να γίνουν εκπτώσεις στις λειτουργικότητες της.

5.5 Επίλογος

Σε μια εποχή που η ψηφιακή μορφή έχει κατακλύσει την καθημερινότητά μας και οι εφαρμογές που χρησιμοποιούμε για τις ανάγκες μας είναι πάρα πολλές, στην σχολική κοινότητα των γενικών λυκείων πολλά πράγματα γίνονται ακόμα με τον παραδοσιακό τρόπο. Το SchoolApp στοχεύει στην παροχή ενός πληροφοριακού συστήματος, χωρίς βεβαίως να καταργήσει τις παραδοσιακές μεθόδους αλληλεπίδρασης μεταξύ της σχολικής κοινότητας.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Εφαρμογή myschool. [Online]. Available: https://myschool.sch.gr/

[2] Ενημέρωση για την επεξεργασία δεδομένων προσωπικού χαρακτήρα μέσω του πληροφοριακού συστήματος myschool. [Online]. Available: https://myschool.sch.gr/help/myschoolGDPR.pdf

[3] Εφαρμογή mySchoolApp. [Online]. Available: https://www.myschoolapp.gr/

[4] mySchoolApp στο Google Store του Android συστήματος. [Online]. Available: https://play.google.com/store/apps/details?id=com.rdc.myschoolapp

[5] Εφαρμογή Εκπαιδευτική Διφθέρα [Online]. Available: https://www.sch.gr/android/

[6] TutorialsPoint – Kanal S Sajan, (2022, August 11). *What is the GDPR, or General Data Protection Regulation.* [Online]. Available: https://www.tutorialspoint.com/what-is-the-gdpr-or-general-data-protection-regulation

[7] PandaDoc Blog – Yauhen Zaremba (2023, January 17). *How to write an effective privacy policy to protect your business.* [Online]. Available: https://www.pandadoc.com/blog/how-to-write-a-privacy-policy/

[8] JavaPoint. *What is the full form of GDPR* [Online]. Available: https://www.javatpoint.com/gdpr-full-form

[9] TechTerms,com - Per Christensson (2020, April 11). *Backend* [Online]. Available: https://techterms.com/definition/backend

[10] WhatIs.com – Laura Fitzgibbons (2019, May). *Frontend and Backend* [Online]. Available: https://www.techtarget.com/whatis/definition/front-end

[11] PHP Documentation - php Group. *What is PHP?* [Online]. Available: https://www.php.net/manual/en/intro-whatis.php

[12] TutorialsPoint. *PHP Tutorial* [Online]. Available: https://www.tutorialspoint.com/php/index.htm

[13] JavaPoint. *PHP Tutorial* [Online]. Available: https://www.javatpoint.com/php-tutorial

[14] PHP Documentation – php Group. *What can PHP do?* [Online]. Available: https://www.php.net/manual/en/intro-whatcando.php

[15] Laravel Documentation. *The PHP Framework for Web Artisans* [Online]. Available: https://laravel.com/

[16] TutorialsPoint. *Laravel Overview* [Online]. Available: https://www.tutorialspoint.com/laravel/laravel_overview.htm

[17] JavaPoint. *Laravel Tutorial* [Online]. Available: https://www.javatpoint.com/laravel

[18] Laravel Documentation. *Routing* [Online]. Available: https://laravel.com/docs/10.x/routing

[19] mdn webdocs_ - Mozilla.org contributors (2023, April 10). *HTTP request methods* [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

[20] Laravel Documentation. *Middleware* [Online]. Available: https://laravel.com/docs/10.x/middleware

[21] TutorialsPoint. *Laravel – Middleware* [Online]. Available: https://www.tutorialspoint.com/laravel/laravel_middleware.htm

[22] JavaPoint. *Laravel – Middleware* [Online]. Available: https://www.javatpoint.com/laravel-middleware

[23] Laravel Documentation. *Authentication* [Online]. Available: https://laravel.com/docs/10.x/authentication

[24] Laravel Documentation. *Authentication* [Online]. Available: https://laravel.com/docs/5.8/authentication

[25] Laravel Documentation. *Eloquent: Getting Started* [Online]. Available: https://laravel.com/docs/10.x/eloquent

[26] Laravel Documentation. *Database: Query Builder* [Online]. Available: https://laravel.com/docs/10.x/queries

[27] Laravel Documentation. *Database: Migrations* [Online]. Available: https://laravel.com/docs/10.x/migrations

[28] Laravel Documentation. *Blade Templates* [Online]. Available: https://laravel.com/docs/10.x/blade

[29] Laravel Documentation. *Artisan Console* [Online]. Available: https://laravel.com/docs/10.x/artisan

[30] Laravel Documentation. *Artisan CLI* [Online]. Available: https://laravel.com/docs/5.0/artisan

[31] Oracle. *What is a Database?* [Online]. Available: https://www.oracle.com/database/what-is-database/

[32] Oracle. *What is MySQL?* [Online]. Available: https://www.oracle.com/mysql/what-is-mysql/

[33] TechTerms,com - Per Christensson (2020, April 18). *Frontend* [Online]. Available: https://techterms.com/definition/frontend

[34] mdn webdocs_ - Mozilla.org contributors (2023, February 24). *Structuring the web with HTML* [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/HTML

[35] GeeksForGeeks (2023, April 19). *What is HTML?* [Online]. Available: https://www.geeksforgeeks.org/html/

[36] freeCodeCamp (2019, December 12). *Semantic HTML5 Elements Explained* [Online]. Available: https://www.freecodecamp.org/news/semantic-html5-elements/

[37] mdn webdocs_ - Mozilla.org contributors (2023, April 16). *CSS: Cascading Style Sheets* [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/CSS

[38] TutorialsPoint. *CSS Tutorial* [Online]. Available: https://www.tutorialspoint.com/css/index.htm

[39] mdn webdocs_ - Mozilla.org contributors (2023, May 2). *JavaScript* [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript

[40] TutorialsPoint. *JavaScript – Overview* [Online]. Available: https://www.tutorialspoint.com/javascript/javascript_overview.htm

[41] ECMA INTERNATIONAL. *ECMAScript* [Online]. Available: https://262.ecma-international.org/

[42] freeCodeCamp – Dillion Megida (2020, February 13). *Object Oriented Programming in JavaScript – Explained with Examples* [Online]. Available: https://www.freecodecamp.org/news/how-javascript-implements-oop/

[43] mdn webdocs_ - Mozilla.org contributors (2023, April 15). *Introduction to the DOM* [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

[44] SIMFORM – Hardik Shah (2022, February 17). 7 Frontend JavaScript Frameworks Loved by Developers in 2023 [Online]. Available: https://www.simform.com/blog/javascript-frontend-frameworks/

[45] jQuery – OpenJS Foundation and jQuery. *What is jQuery?* [Online]. Available: https://jquery.com/

[46] w3schools – W3.CSS. *jQuery Introduction* [Online]. Available: https://www.w3schools.com/jquery/jquery_intro.asp

[47] GeeksForGeeks (2023, April 19). *What is Bootstrap?* [Online]. Available: https://www.geeksforgeeks.org/bootstrap/

[48] Microsoft (2023). *Visual Studio Code* [Online]. Available: https://code.visualstudio.com/docs

[49] Microsoft (2023). *Why Visual Studio Code?* [Online]. Available: https://code.visualstudio.com/docs/editor/whyvscode

[50] JavaPoint. *XAMPP TUTORIAL* [Online]. Available: https://www.javatpoint.com/xampp

[51] phpMyAdmin contributors (2003-2023). *phpMyAdmin About* [Online]. Available: https://www.phpmyadmin.net/

[52] JavaPoint. *phpMyAdmin* [Online]. Available: https://www.javatpoint.com/phpmyadmin

[53] Mailtrap – Railsware products Studio LLC. *Mailtrap* [Online]. Available: https://help.mailtrap.io/article/12-getting-started-guide?gclid=CjwKCAjw9pGjBhB-EiwAa5jl3NJLRnWr2AcmVIoqhySKQFZCAjUdi95KxXZdLyaC4QYtJH8JwD9hKhoCuW0QAvD_BwE

[54] freeCodeCamp – Adrian Twarog (2021, June 21). *What is Figma? A Design Crash Course [2021 Tutorial]* [Online]. Available: https://www.freecodecamp.org/news/figma-crash-course/

[55] Microsoft (2023). *Navigating the F12 Developers Tools Interface* [Online]. Available: https://learn.microsoft.com/en-us/previous-versions/windows/desktop/legacy/hh968260(v=vs.85)

[56] mdn webdocs_ - Mozilla.org contributors (2023, February 24). *What are browser developer tools?* [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Tools_and_setup/What_are_browser_developer_tools

[57] freeCodeCamp – Ihechikara Vincent Abba (2021, November 5). *Git and GitHub Tutorial – Version Control for Beginners* [Online]. Available: https://www.freecodecamp.org/news/git-and-github-for-beginners/

[58] Laravel Documentation. *Facades* [Online]. Available: https://laravel.com/docs/10.x/facades

ΠΑΡΑΡΤΗΜΑ Α : Ενδεικτικός κώδικας εφαρμογής

Σε αυτό το παράρτημα θα γίνει παρουσίαση και σύντομη ανάλυση επιλεγμένων σημείων του κώδικα της εφαρμογής. Τα κομμάτια κώδικα που θα παρουσιαστούν αντιπροσωπεύουν την λογική ολόκληρου του κώδικα, προφανώς κάθε κομμάτι του κώδικα έχει και κάτι διαφορετικό, όμως στόχος είναι η προβολή μιας γενικής λογικής που ακολουθήθηκε και γενικών πρακτικών που χρησιμοποιήθηκαν.

```
<div id="contains-main-content-and-left-sidebar" class="wrapper-left-menu-and-main-content">
    @if (Auth::check())
        <div class="text-white bg-dark left-menu-sidebar show-left-menu hide-left-menu">
            <div class="d-flex align-items-center mb-3 mb-md-0 me-md-auto text-white text-decoration-none">
                <span class="fs-4">
                    @if (auth()->user()->type == 'admin')
                        Διαχειριστής
                    @elseif (auth()->user()->type == 'student')
                        Μαθητής
                    @else
                        Καθηγητής
                    @endif
                </span>
            </div>
            <hr>
            <ul class="nav nav-pills flex-column mb-auto" id="ul-left-side-menu">
                @include('users.leftMenu')
            </ul>
            @php
                $infoMsg;
                if (auth()->user()->type != 'admin') {
                    $infoMsg = 'Για οποιοδήποτε πρόβλημα ή απορία επικοινωνήστε στο admin@mail.com';
                } else {
                    $infoMsg = 'Για οποιοδήποτε πρόβλημα ή απορία επικοινωνήστε στο support@schoolapp.com';
                }
            @endphp
            <span class="info-icon-container">
                <i class="fa-solid fa-circle-info"></i>
                <span class="info-tooltip">{{ $infoMsg }}</span>
            </span>
        </div>
    @endif
    <main class="py-4 main-content">
        @yield('content')
    </main>
</div>
@include('layouts.footer')
```

Σχήμα Α.1: Αρχείο blade 1

Στο παραπάνω σχήμα, Σχήμα Α.1, φαίνεται ο κώδικας HTML ενός αρχείου blade. Στόχος είναι να διαμορφώσει το περιεχόμενο της σελίδας. Φαίνεται στην πράξη πως ένα αρχείο blade υποστηρίζει εκτός από τα στοιχεία HTML όπως είναι τα div, span, ul και δομές κώδικα. Κάθε δομή ξεκινάει και σταματάει με τον ειδικό χαρακτήρα '@', π.χ. @if … @endif. Μέσω της μεθόδου check() του Facade Auth ελέγχει αν ο χρήστης έχει ταυτοποιηθεί από το σύστημα προκειμένου να του δείξει το περιεχόμενο της εφαρμογής. Επίσης, βλέπουμε την δυνατότητα να γραφτεί απευθείας php κώδικας μέσα στο αρχείο χρησιμοποιώντας την δομή @php … @endphp. Επιπλέον, μέσω ελέγχων του τύπου των χρηστών διαχειριζόμαστε το περιεχόμενο κατάλληλα ενώ για να χρησιμοποιηθεί μια μεταβλητή μέσα στον HTML κώδικα ώστε η πληροφορία να είναι δυναμική, πρέπει να περικλείεται με '{{ όνομα μεταβλητής

}}’. Τέλος, φαίνεται η λογική των components που έχει ακολουθηθεί στην ανάπτυξη της εφαρμογής αφού μέσα στον βασικό κορμό της εμφάνισης το πλαϊνό menu, το κυρίως περιεχόμενο και το footer εισάγονται από τρίτα αρχεία μέσω των μεθόδων @include() και @yield(). Αν και αυτές οι δύο μέθοδοι φαινομενικά πετυχαίνουν το ίδιο αποτέλεσμα διαφέρουν πολύ στον τρόπο λειτουργίας τους. Η μέθοδος @yield(‘όνομα τμήματος’) χρησιμοποιείται για να δηλώσει ένα τμήμα εμφάνισης. Για να μπει περιεχόμενο, πρέπει στο τρίτο αρχείο να χρησιμοποιηθεί η μέθοδος @extends(‘όνομα αρχείου που περιέχει τη μέθοδο yield που θέλουμε να χρησιμοποιήσουμε’) για να επιλέξουμε το συγκεκριμένο τμήμα και στη συνέχεια πρέπει να ορισθεί η μέθοδος @section(‘όνομα τμήματος’). Το όνομα τμήματος της @section() πρέπει να είναι το ίδιο με της @yield ώστε το περιεχόμενο να φορτωθεί. Από την άλλη μεριά, η include(‘όνομα αρχείου που θα εισαχθεί’) χρησιμοποιείται για να γίνει εισαγωγή ενός επαναχρησιμοποιούμενου HTML περιεχομένου. Η @include() δεν υιοθετεί την σχέση γονέα-παιδί όπως οι @yield() - @section().

```blade
@extends('layouts.app')

@section('content')
    <div class="wrapper create-student">
        <div class="card">
            <div class="card-head">
                <h4>Φόρμα δημιουργίας νέου μαθητή</h4>
            </div>
            <div class="card-body">
                {!! Form::open(["action" => "App\Http\Controllers\StudentsController@store", "method" => "POST"]) !!}
                    @csrf
                    <div class="form-group">
                        {{ Form::label('firstname', 'Όνομα*') }}
                        {{ Form::text('firstname', '', ['class' => ($errors->has('firstname')) ? 'form-control form-error' : 'form-control', 'placeholder' => 'Όνομα']) }}
                        {!! $errors->first('firstname', '<p class="form-error-msg-under-field">:message</p>') !!}
                    </div>
                    <div class="form-group">
                        {{ Form::label('lastname', 'Επόνυμο*') }}
                        {{ Form::text('lastname', '', ['class' => ($errors->has('lastname')) ? 'form-control form-error' : 'form-control', 'placeholder' => 'Επόνυμο']) }}
                        {!! $errors->first('lastname', '<p class="form-error-msg-under-field">:message</p>') !!}
                    </div>
                    <div class="form-group">
                        {{ Form::label('email', 'Email*') }}
                        {{ Form::email('email', '', ['class' => ($errors->has('email')) ? 'form-control form-error' : 'form-control', 'placeholder' => 'Email']) }}
                        {!! $errors->first('email', '<p class="form-error-msg-under-field">:message</p>') !!}
                    </div>
                    <div class="form-group">
                        {{ Form::label('city', 'Πόλη*') }}
                        {{ Form::text('city', '', ['class' => ($errors->has('city')) ? 'form-control form-error'  : 'form-control', 'placeholder' => 'Πόλη']) }}
                        {!! $errors->first('city', '<p class="form-error-msg-under-field">:message</p>') !!}
                    </div>
                    <div class="form-group">
                        {{ Form::label('address', 'Διεύθυνση*') }}
                        {{ Form::text('address', '', ['class' => ($errors->has('address')) ? 'form-control form-error'  : 'form-control', 'placeholder' => 'Διεύθυνση']) }}
                        {!! $errors->first('address', '<p class="form-error-msg-under-field">:message</p>') !!}
                    </div>
```

Σχήμα Α.2: Αρχείο blade 2

Στο σχήμα Α.2, φαίνονται αρχικά η χρησιμοποίηση των μεθόδων @extends(‘όνομα αρχείου γονέα’) @section(‘όνομα τμήματος’). Στη συνέχεια φαίνεται ο τρόπος με τον οποίο δίνει την δυνατότητα το blade template engine, να δημιουργηθούν HTML στοιχεία πέραν των κλασικών στοιχείων HTML. Για παράδειγμα με κλασικό HTML κώδικα ένα text input δημιουργείται ως εξής:

- <input type=’text’ name=’name’ value=’value’ class=’class name’, placeholder=’placeholder’/>

Η παραπάνω γραμμή κώδικα μπορεί να γραφτεί από το blade και με δεύτερο τρόπο ως εξής:

- {{ Form::text(‘name’, ‘value’, [‘class’ => ‘class name’, ‘placeholder’ => ‘placeholder’]) }}

Κατά την διάρκεια ανάπτυξης της εφαρμογής χρησιμοποιήθηκαν και οι δύο τρόποι.

```php
// student routes
Route::get('/students', [StudentsController::class, 'index'])->name('students.index');
Route::get('/students/create', [StudentsController::class, 'create'])->name('students.create');
Route::get('/students/{id}', [StudentsController::class, 'show'])->name('students.show');
Route::post('/students', [StudentsController::class, 'store'])->name('students.store');
Route::delete('/students/{id}', [StudentsController::class, 'destroy'])->name('students.destroy');
Route::get('/students/{id}/edit', [StudentsController::class, 'edit'])->name('students.edit');
Route::post('/students/{id}', [StudentsController::class, 'update'])->name('students.update');
Route::get('/students/{id}/grades', [StudentsController::class, 'grades'])->name('students.grades');
```

Σχήμα Α.3: Web Routes

Στο σχήμα Α.3 φαίνονται ενδεικτικά ένα σύνολο διαδρομών που χρησιμοποιήθηκαν στην εφαρμογή και αφορούν την οντότητα του μαθητή. Είναι καλή πρακτική στο αντίστοιχο αρχείο διαδρομών, οι διαδρομές ξεχωριστών οντοτήτων να ομαδοποιούνται προσφέροντας καλύτερη οργάνωση και ανάγνωση στον εκάστοτε προγραμματιστή. Παρατηρούμε πως όλες οι διαδρομές δημιουργούνται χρησιμοποιώντας τη κλάση Route η οποία παρέχεται μέσα από τα Facades που παρέχει το Laravel. Επίσης, για κάθε διαδρομή πρέπει να ορισθεί η HTTP μέθοδος που υποστηρίζεται. Οι μέθοδοι αυτοί περιμένουν δυο παραμέτρους προκειμένου να λειτουργήσουν. Πρώτη παράμετρος είναι το URI και η δεύτερη είναι ένας πίνακας που περιλαμβάνει τον controller και την μέθοδο του controller που θα διαχειριστούν την συγκεκριμένη ενέργεια. Προαιρετικό αλλά πολύ σημαντικό είναι η ονομασία μιας διαδρομής, μέσω του ονόματος μπορούμε να αναφερθούμε σε αυτήν από οπουδήποτε μέσα στην εφαρμογή. Στην περίπτωση που χρειάζεται να περάσουμε κάποιο δυναμικό δεδομένο στο URI γίνεται περικλείοντας ένα υποτμήμα του URI στα σύμβολα ‘{id}’. Να σημειωθεί πως οι διαδρομές με δυναμικό περιεχόμενο θα πρέπει να ορισθούν προς το τέλος κατά σειρά. Με το παρακάτω παράδειγμα θα εξηγηθεί ο λόγος. Ας δανειστούμε για το παράδειγμα δύο από τις διαδρομές του σχήματος Α.3:

Route::get(‘/students/{id}’, [StudentsController::class, show])->name(‘students.show’);

Route::get(‘/students/create’, [StudentsController::class, create])->name(‘students.create’);

Εάν οι δύο αυτές διαδρομές είναι με αυτή τη σειρά δηλωμένες και η ενέργεια που ζητηθεί είναι για το URI ‘/students/create’ ψάχνοντας θα αντιστοιχήσει το create με το δυναμικό {id} και επομένως θα γίνει κλήση της λάθος διαδρομής.

Στα σχήματα Α.4, Α.5, Α.6.1 και Α.6.2 που ακολουθούν παρατίθεται ενδεικτικός κώδικας από τον StudentsController που διαχειρίζεται όλες τις ενέργειες που σχετίζονται με την οντότητα των μαθητών.

```php
class StudentsController extends Controller
{

    // this function prevent all non-authenticated users from interact with the application
    public function __construct() {
        $this->middleware('auth');
    }
```

Σχήμα Α.4: StudentsController – constructor

Στο σχήμα Α.4 βλέπουμε τον constructor της κλάσης ο οποίος, περιέχει και έλεγχο για την ΄ταυτοποίηση του χρήστη. Είναι πολύ βασική λειτουργεία καθώς αποκλείει όλους τους μη ταυτοποιημένους χρήστες από το να αλληλεπιδράσουν με τις λειτουργικότητες που αφορούν τους μαθητές και τις διαχειρίζεται ο StudentsController.

```php
public function index()
{
    // check if the user is the correct type to do this action
    if (auth()->user()->type !== 'admin') {
        return redirect()->route('home')->with('error', 'Μη εξουσιοδοτημένος χρήστης.');
    }

    // check the filter
    $filter = request('filter');

    // if filter has value search for the name else display all students to screen
    if (!empty($filter)) {
        $students = Student::sortable()->where('lastname', 'like', '%'.$filter.'%')
                                       ->orWhere('firstname', 'like', '%'.$filter.'%')->paginate(10);
    } else {
        $students = Student::all();
        $students = Student::sortable()->paginate(10);
    }

    return view('students.index', ['students' => $students, 'filter' => $filter]);
}
```

Σχήμα Α.5: StudentsController – μέθοδος Index

Στο σχήμα Α.5 παρατίθεται η μέθοδος index(), η οποία χρησιμοποιείται για να πάρει από την βάση δεδομένων όλους τους διαθέσιμους μαθητές οι οποίοι βρίσκονται στο πίνακα students και να τους επιστρέψει στο αντίστοιχο view. Αρχικά πρέπει να γίνει και ένας δεύτερος έλεγχος εάν ο ταυτοποιημένος χρήστης είναι και εξουσιοδοτημένος για μια τέτοια ενέργεια. Αυτό γίνεται ελέγχοντας τον τύπου του τρέχον ενεργού χρήστη. Σε περίπτωση που δεν έχει το δικαίωμα της συγκεκριμένες ενέργειας μεταφέρεται στην αρχική σελίδα με αντίστοιχο μήνυμα σφάλματος. Στην μέθοδο υλοποιείται και η διαχείριση της αναζήτησης ενός μαθητή, ενέργειας που αναφέρθηκε σε προηγούμενο κεφάλαιο, η οποία γίνεται μέσω της μεταβλητής $filter. Έτσι λοιπόν, ανάλογα με το εάν ο χρήστης έχει εισάγει κάποια τιμή για αναζήτηση ή όχι, εκτελείται ένα αντίστοιχο Eloquent ερώτημα στη βάση δεδομένων με σκοπό να ανακτηθούν οι αντίστοιχες πληροφορίες. Στη περίπτωση που έχει τιμή, γίνεται αναζήτηση εάν η τιμή αυτή υπάρχει μέσα σε κάποιο όνομα ή επώνυμο κάθε εγγραφής του πίνακα students και επιστρέφει την αντίστοιχη διαθέσιμη λίστα. Σε αντίθετη περίπτωση επιλέγονται όλοι οι μαθητές. Και στις δύο περιπτώσεις επιστρέφεται η λίστα των μαθητών με όριο τις δέκα εγγραφές, αυτό εξυπηρετεί στην λειτουργεία της σελιδοποίησης του πίνακα. Επίσης, τα ερωτήματα Eloquent στη βάση δεδομένων γίνονται μέσω του μοντέλου Student. Τέλος, αφού όλα γίνουν επιτυχώς επιστρέφει στην οθόνη το αντίστοιχο view με το κατάλληλο περιεχόμενο. Αυτό γίνεται μέσω της μεθόδου view που παρέχει το Laravel η οποία δέχεται σαν πρώτη παράμετρο το όνομα του blade αρχείου view και σαν δεύτερη παράμετρο έναν πίνακα με key-value pairs τα οποία περνάνε μεταβλητές στο view ώστε αυτές να χρησιμοποιηθούν στο view αρχείο.

```php
/**
 * Update the specified resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    // check if the user is the correct type to do this action
    if (auth()->user()->type !== 'admin')  {
        return redirect()->route('home')->with('error', 'Μη εξουσιοδοτημένος χρήστης.');
    }
          You, now • Uncommitted changes
    // validation
    $this->validate($request, [
        'firstname' => 'required|alpha|max:255',
        'lastname' => 'required|alpha|max:255',
        'email' => 'required|email:filter|max:255',
        'city' => 'required|alpha|max:255',
        'address' => 'required|regex:/^[\x{0386}-\x{03CE},\.]+.*[\s]+.[0-9]{0,}[\x{0386}-\x{03CE}]?$/u|max:255',
        'absences' => 'required|integer',
        'excusedabsences' => 'required|integer',
        'unexcusedabsences' => 'required|integer',
        'phonenumber' => 'required|digits_between:10, 10',
        'class' => 'required|alpha_num',
        'secondYearOrientationGroup' => 'nullable',
        'thirdYearOrientationGroup' => 'nullable'
    ], $GLOBALS['studentFormValidationErrorMessages']);

    $student = Student::findOrFail($id);

    if ($student->email != request('email')) {
        DB::table('users')->where('email', $student->email)->update(['email' => request('email')]);
    }
```

Σχήμα Α.6.1: StudentsController – μέθοδος Update

```php
$student->firstname = request('firstname');
$student->lastname = request('lastname');
$student->email = request('email');
$student->city = request('city');
$student->address = request('address');
$student->absences = request('absences');
$student->excusedabsences = request('excusedabsences');
$student->unexcusedabsences = request('unexcusedabsences');
$student->phonenumber = request('phonenumber');

// get all school classes from db
$classes = SchoolClass::orderBy('name', 'ASC')->get();
$availableClassesNames = array();

foreach ($classes as $class) {
    array_push($availableClassesNames, $class->name);
}
$student->class = $availableClassesNames[request('class')];

// orientation group handling
// get all available class orientation groups
$orientationGroups = OrientationGroup::all();
$secondYearOrientationGroup = $orientationGroups[0];
$thirdYearOrientationGroup = $orientationGroups[1];

if (str_contains($availableClassesNames[request('class')], 'Β')) {
    foreach ($secondYearOrientationGroup->group_list as $key => $value) {
        if ($key == request('secondYearOrientationGroup')) {
            $student->orientation_group = $value;
        }
    }
} else if (str_contains($availableClassesNames[request('class')], 'Γ')) {
    foreach ($thirdYearOrientationGroup->group_list as $key => $value) {
        if ($key == request('thirdYearOrientationGroup')) {
            $student->orientation_group = $value;
        }
    }
} else {
    $student->orientation_group = null;
}
```

Σχήμα Α.6.2: StudentsController – μέθοδος Update

```
    $gradesList = [];
    foreach ($availableCourses->available_courses as $course) {
        $gradeObject = (object) ["course_name" => $course, "grade" => "", "img" => [], "comments" => ""];
        array_push($gradesList, $gradeObject);
    }
    $grades->first_quarter = $gradesList;
    $grades->second_quarter = $gradesList;
    $grades->final_exams = $gradesList;
    $grades->update();
    // update user record
    $student->update();
    // get an object of that student in order to inform the admin if the current student has an account or not
    $studentAccount = DB::table('users')->where('email', request('email'))->first();
    return view('students.show', ['student' => $student, 'studentAccount' => $studentAccount, 'createMsgSuccess' => 'Οι αλλαγές αποθηκεύτηκαν επιτυχώς!']);
}
```

Σχήμα Α.6.3: StudentsController – μέθοδος Update

Στα παραπάνω σχήματα Α.6.1 και Α.6.2 παρατίθεται μια ενδεικτική μέθοδος για την ενημέρωση δεδομένων στη βάση. Η γενική πρακτική που ακολουθείται σε αυτές τις μεθόδους είναι να ελέγχεται πρώτα απ' όλα ο τύπος του χρήστη προκειμένου να διασαφηνιστεί αν έχει το δικαίωμα για την ενέργεια ή όχι. Στη συνέχεια ακολουθεί ο έλεγχος της φόρμας που υποβλήθηκε ώστε να διασφαλισθεί η ασφάλεια των δεδομένων της βάσης. Ο έλεγχος αυτός γίνεται μέσω της μεθόδου this->validate() που παρέχει το Laravel στον πυρήνα του. Σε αυτή τη μέθοδο ο προγραμματιστής μπορεί να επιλέξει όποιο πεδίο της φόρμας θέλει και να εφαρμόσει κανόνες για την μορφή που πρέπει να έχε το κάθε ένα πεδίο. Έπειτα, πρέπει να βρεθεί η αντίστοιχη εγγραφή στη βάση και να ανανεωθούν οι τιμές. Ανάλογα και τα δεδομένα που θέλουμε να ανανεώσουμε ίσως χρειαστεί να γίνει ανανέωση και σε δεύτερο πίνακα ο οποίος έχει αναφορά στην ίδια εγγραφή. Στο παράδειγμα που φαίνεται στα σχήματα, θα πρέπει εκτός του μαθητή να ενημερωθεί και η αντίστοιχη εγγραφή του στον πίνακα grades ο οποίος κρατάει πληροφορίες σχετικά με τα μαθήματα και τις βαθμολογίες του. Στο σχήμα Α6.3 βλέπουμε και ερώτημα προς την βάση μέσω του Database Query Builder.

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Kyslik\ColumnSortable\Sortable;

class Student extends Model
{
    use HasFactory;

    use Sortable;

    public $sortable = ['lastname', 'email', 'class'];
}
```

Σχήμα Α.7: Student Model

Στο σχήμα Α.7 βλέπουμε το Model Student, το οποίο μπορούμε να χρησιμοποιήσουμε ώστε να κάνουμε ερωτήματα Eloquent στον αντίστοιχο πίνακα της βάσης. Τέτοιο ερώτημα παρουσιάστηκε στο σχήμα Α.5. Χρησιμοποιώντας το $sortable πίνακα ορίζουμε πως σε αυτές τις τρείς στήλες που αναφέρονται μέσα στο πίνακα υποστηρίζεται η ταξινόμηση.

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Program extends Model
{
    use HasFactory;

    protected $casts = [
        'monday' => 'array',
        'tuesday' => 'array',
        'wednesday' => 'array',
        'thursday' => 'array',
        'friday' => 'array'
    ];
}
```

Σχήμα Α.8: Program Model

Στο σχήμα Α.8 είναι ένα δέυτερο παράδειγμα ενός Model του Program. Αυτό αντιστοιχεί στο σχολικό πρόγραμμα ενός σχολικού τμήματος και χρησιμοποιείται για να αλληλεπιδράσει με τον αντίστοιχο πίνακα στη βάση. Βλέπουμε πως εδώ χρησιμοποιείται μια άλλη μεταβλητή η $casts η οποία αναπαριστά και αυτή έναν πίνακα με key-value pairs. Στην ουσία αυτό γίνεται για να ορίσουμε ότι οι στήλες αυτού του μοντέλου αναπαριστούν η κάθε μια συλλογή δεδομένων, δηλαδή πίνακα. Χωρίς αυτή τη δήλωση δεν θα είμαστε σε θέσει να διαχειριστούμε μέσα στον κώδικα αυτές τις στήλες ως πίνακες αλλά ως ένα μεγάλο string την κάθε μια.

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateStudentsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('students', function (Blueprint $table) {
            $table->id();
            $table->string('firstname');
            $table->string('lastname');
            $table->string('email');
            $table->string('city');
            $table->string('address');
            $table->integer('absences');
            $table->string('phonenumber');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('students');
    }
}
```

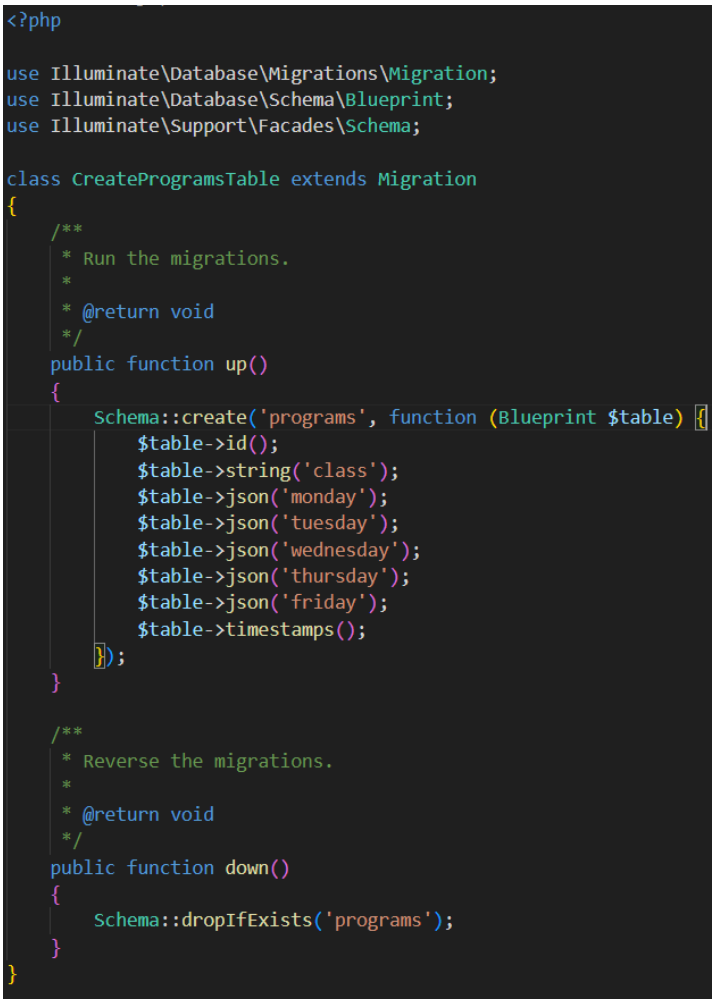
Σχήμα Α.9: Migration αρχείο – Δημιουργία πίνακα students

Στο σχήμα Α.9, φαίνεται πως είναι ένα αρχείο τύπου Migration. Στο συγκεκριμένο σχήμα έχει δημιουργηθεί ένα migration με στόχο την δημιουργία ενός πίνακα ονόματι students στην βάση δεδομένων. Κάθε migration αρχείο έρχεται με δύο μεθόδους, την up() και την down(). Αυτές οι δύο μέθοδοι είναι υπεύθυνες για την δημιουργία και την καταστροφή του συγκεκριμένου πίνακα. Επίσης, ευδιάκριτο το πόσο εύκολο είναι να δημιουργηθούν οι στήλες του πίνακα και να ορισθεί και ο τύπος τους.

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class AddClassToStudentsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::table('students', function (Blueprint $table) {
            $table->string('class');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::table('students', function (Blueprint $table) {
            //
        });
    }
}
```

Σχήμα Α.10: Migration αρχείο – Τροποποίηση πίνακα students

Στο σχήμα Α.10 βλέπουμε ένα migration αρχείο το οποίο τροποποιεί τον πίνακα students ο οποίος δημιουργήθηκε από το migration αρχείο του σχήματος Α.11. Πρακτικά προσθέτει μια επιπλέον στήλη στον πίνακα students με όνομα class και είναι τύπου string.



```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateProgramsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('programs', function (Blueprint $table) {
            $table->id();
            $table->string('class');
            $table->json('monday');
            $table->json('tuesday');
            $table->json('wednesday');
            $table->json('thursday');
            $table->json('friday');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('programs');
    }
}
```

Σχήμα Α.11: Migration αρχείο – Δημιουργία πίνακα programs

Το σχήμα Α.11 είναι το τελευταίο που αναφέρεται σε migration αρχείο. Το συγκεκριμένο migration πραγματεύεται τη δημιουργία ενός πίνακα ονόματι programs στη βάση δεδομένων. Αυτό που πρέπει να επισημανθεί είναι το πως μπορεί να δημιουργηθεί μια στήλη η οποία θα φιλοξενεί έναν πίνακα από τιμές για κάθε εγγραφή. Η βάση από τη φύση της δεν υποστηρίζει τον τύπο πίνακα και για να προσπεράσουμε αυτό το εμπόδιο χρησιμοποιούμε την τεχνολογία json. Ουσιαστικά στην βάση θα αποθηκευτεί η πληροφορία ως json. Σε συνδυασμό με το Model Program, του σχήματος Α.8, και την μεταβλητή $casts, γίνεται μετατροπή της της στήλης σε πίνακα όταν τα δεδομένα παίρνονται από τη βάση και στέλνονται σαν json όταν είναι να αποθηκευτούν στη βάση. Αυτό είναι μια πολύ σημαντική δυνατότητα που παρέχει το Laravel προκειμένου να μπορεί να εργαστεί ένας προγραμματιστής με πίνακες που θέλει να αποθηκεύσει στη βάση.

```
// img dialog section - start
// open img dialog
function openImgDialog(path) {
    let imgDialogDiv = document.getElementsByClassName('test-img-dialog')[0];
    let img = document.getElementById('test-img-for-dialog');
    img.src = path;
    imgDialogDiv.style.display = 'unset';
}

// close img dialog
function closeImgDialog(event) {
    if (event.target.tagName !== 'IMG') {
        let imgDialogDiv = document.getElementsByClassName('test-img-dialog')[0];
        imgDialogDiv.style.display = 'none';
    }
}
// img dialog section - end

// the next 3 methods are for remove icon next of images of courses start
// display remove icon
function displayRemoveIcon(e) {
    if (e.children[1]) e.children[1].style.display = "unset";
}

// hide remove icon
function hideRemoveIcon(e) {
    if (e.children[1]) e.children[1].style.display = "none";
}

// do the delete of image
function removeSpecificImg(e) {
    if (e.previousSibling.previousSibling) e.previousSibling.previousSibling.remove();
    if (e.parentElement) e.parentElement.remove();
    if (e) e.remove();
}
// the next 3 methods are for remove icon next of images of courses end
```

Σχήμα Α.12: Ενδεικτικός κώδικας JavaScript που χρησιμοποιήθηκε

Το σχήμα Α.12 παρουσιάζει ένα ενδεικτικό κομμάτι του κώδικα JavaScript που χρησιμοποιήθηκε στην εφαρμογή. Οι πέντε μέθοδοι αφορούν το image dialog box που υπάρχει στην οθόνη βαθμολογίας ενός μαθήματος. Οι δύο πρώτες μέθοδοι αφορούν και την οθόνη που βλέπει ο καθηγητής προκειμένου να επεξεργαστεί το περιεχόμενο καθώς και την οθόνη που βλέπει ο μαθητής που είναι μόνο για προβολή των δεδομένων. Η πρώτη διαχειρίζεται το κλικ στην εκάστοτε εικόνα προκειμένου να ανοίξει το dialog box και η δεύτερη διαχειρίζεται το κλείσιμο του dialog box. Οι υπόλοιπες τρείς μέθοδοι λειτουργούν μόνο στην οθόνη του χρήστη τύπου καθηγητή και αφορά την διαγραφή της εκάστοτε φωτογραφίας.

```css
.grades-card .grades-per-course span:last-child i {
    cursor: pointer;
}

.grades-card .grades-per-course span:last-child a {
    color: unset;
    text-decoration: none;
}

.form-control:focus {
    border-color: #282a35 !important;
}

.status-message {
    text-align: center;
    margin-bottom: 40px !important;
}

.test-img-container {
    text-align: center;
    display: flex;
    gap: 30px;
    flex-wrap: wrap;
}

.test-img-container img {
    height: 100px;
    width: 100px;
    align-self: center;
}
```

Σχήμα Α.13: Ενδεικτικός κώδικας CSS που χρησιμοποιήθηκε

```css
@media screen and (max-width: 768px) {
    .dropdown-menu-right {
      position: absolute !important;
    }
  }

@media screen and (max-width: 450px) {
    .navbar .container {
        justify-content: center !important;
    }

    .navbar .container #navbarSupportedContent {
        justify-content: center;
    }

    .navbar .container .navbar-nav {
        margin: auto;
        gap: 10px;
        flex-direction: row;
    }
}
```

Σχήμα Α.14: Ενδεικτικός κώδικας CSS - Media Queries (responsive web design)

```
<div class="form-group comments-observations-container">
    <label for="comments">Φόρμα σχολίων-παρατηρήσεων</label>
    <textarea name="comments" cols="30" rows="10" id="ckeditor"
        placeholder="Εδώ μπορείτε να γράψετε τα σχόλια και τις παρατηρήσεις σας.">
        {{ $courseComments }}
    </textarea>      You, 1 second ago • Uncommitted changes
</div>
```

Σχήμα Α.15.:ckEditor HTML

```
{{-- this script is used for load ckEditor --}}
<script src="{{asset('ckeditor/ckeditor.js')}}"></script>
<script>
    // set 1000ms delay otherwise script is loading before the content and can not find the ids
    setTimeout(() => {
        // display ckEditor
        if (document.getElementById('ckeditor')) CKEDITOR.replace('ckeditor');

        // create new img element for every new img user uploads
        let imgUploadedPreviewContainer = document.getElementsByClassName('test-img-container')[0];
        let fileUploader = document.getElementById('test-img-upload');
        fileUploader.onchange = function () {
            if ($("input:file") && $("input:file")[0]) {
                var numFiles = $("input:file")[0].files.length;
                const files = fileUploader.files;
                for (let i = 0; i < numFiles; i++) {
                    const div = document.createElement('div');
                    div.className = "img-and-remove-icon-container";
                    imgUploadedPreviewContainer.appendChild(div);
                    const img = document.createElement('img');
                    img.className = 'test-img-preview-section';
                    if (files[i]) {
                        img.src = URL.createObjectURL(files[i]);
                        img.alt = files[i].name;
                        div.appendChild(img);
                        img.addEventListener('click', () => openImgDialog(img.src));
                    }
                    const removeIcon = document.createElement('i');
                    removeIcon.className = "fa-solid fa-xmark remove-uploaded-img-from-test-images";
                    div.appendChild(removeIcon);
                }
            }
        };
    }, 1000);
</script>
```

Σχήμα Α.16:ckEditor - JavaScript

Τέλος, στα σχήματα Α.15 και Α.16 δείχνουν την σύνδεση του ckEditor καθώς και τον js κώδικα που διαχειρίζεται τον μεταβλητό αριθμό των σκαναρισμένων εικόνων που μπορεί να ανεβάσει ο χρήστης τύπου καθηγητής. Στο σχήμα Α.15 φαίνεται το text area στο οποίο θα εφαρμοστεί ο ckEditror, δεν μπορεί να εφαρμοστεί χωρίς HTML στοιχείο text area. Το επόμενο βήμα είναι να εισάγουμε τον ckEditor στο αρχείο. Υπάρχουν τρείς τρόποι για να χρησιμοποιηθεί ένας ckEditor:

1. Να γίνει εγκατάσταση του αντίστοιχου πακέτου μέσω npm.
2. Να γίνει λήψη των αρχείων στον υπολογιστή και να τοποθετηθούν μέσα στο project.
3. Να γίνει αναφορά μέσω CDN.

Στην ανάπτυξη της εφαρμογής επιλέχθηκε ο δεύτερος τρόπος. Έτσι χρησιμοποιήθηκε το πρώτο από τα δύο script του σχήματος Α.16 προκειμένου να εισαχθεί στο ζητούμενο blade αρχείο όπου βρίσκεται το text area. Στο δεύτερο script έχει τοποθετηθεί μια μέθοδος για να καθυστερήσει την εκτέλεση του κώδικα κατά ένα δευτερόλεπτο (1000ms = 1sec), προκειμένου να προλάβει το περιεχόμενο HTML να φορτώσει όλα τα στοιχεία και να αποφευχθεί κάποιο κρίσιμο error από την μεριά της js. Ψάχνοντας δηλαδή να βρει ένα στοιχείο το οποίο δεν θα έχει προλάβει να φορτώσει θα επιστρέψει error. Με την πρώτη εντολή CKEDITOR.replace('ckeditor') εμφανίζουμε εντός του text area τον ckeditor. Στη συνέχεια ο κώδικας αφορά και πάλι τις σκαναρισμένες εικόνες που θέλει να ανεβάσει ο καθηγητής. Εφόσον ο αριθμός των εικόνων είναι μεταβλητός θα πρέπει με δυναμικό τρόπο να δημιουργηθούν και τα αντίστοιχα HTML στοιχεία. Αυτό επιτυγχάνεται αξιοποιώντας το DOM μέσω της js. Μόλις ο χρήστης επιλέξει τις εικόνες που θα ανεβάσει γίνεται μια λούπα στο σύνολο των εικόνων δημιουργώντας τα αντίστοιχα HTML στοιχεία και περνώντας τους όλες τις παραμέτρους που χρειάζονται π.χ. το src (source) των εικόνων προκειμένου να εμφανιστούν στην οθόνη καθώς και το click event το οποίο πυροδοτεί την μέθοδο openImgDialog(), σχέδιο Α.12. Εκτός από τις εικόνες αυτές καθαυτές χρειάζεται να δημιουργηθεί και ένα 'x' κουμπί για κάθε μια, το οποίο θα διαχειρίζεται την διαγραφή της εκάστοτε εικόνας. Αφού υλοποιηθεί και αυτό το βήμα πρέπει να μπούνε πάνω στο DOM αξιοποιώντας την μέθοδο appendChild() που παρέχει η JavaScript.