Πολυτεχνείο Κρήτης
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Σχεδιασμός και ανάπτυξη εφαρμογής (mobile/web app) για την καταγραφή, διαμοιρασμό και αξιοποίηση γραφικής αναπαράστασης αλγορίθμων και μελέτη περίπτωσης για συνταγές μαγειρικής

Διπλωματική Εργασία

Παναγιώτης Πατσόγλου

Εξεταστική Επιτροπή:
Σαμολαδάς Βασίλης, Αναπληρωτής Καθηγητής (Επιβλέπων)
Δεληγιαννάκης Αντώνιος, Καθηγητής
Μανιά Αικατερίνη, Καθηγήτρια

Χανιά, Ιούνιος 2022

Περίληψη

Η παρούσα Διπλωματική Εργασία υλοποιήθηκε στα πλαίσια του σχεδιασμού και της ανάπτυξης της διαδικτυακής εκπαιδευτικής πλατφόρμας Cooking STEAM. Σκοπός της εφαρμογής είναι να προσφέρει ένα εργαλείο διδασκαλίας που χρησιμοποιεί την μεθοδολογία STEAM και επικεντρώνεται στη διδασκαλία βασικών αλγοριθμικών εννοιών μέσω της γραφικής αναπαράστασης αυτών με διαγράμματα ροής που απεικονίζουν τα βήματα εκτέλεσης συνταγών μαγειρικής. Εκτός από τα διαγράμματα ροής το εν λόγω εργαλείο δίνει τη δυνατότητα αναπαράστασης συνταγών μαγειρικής και σε μη ντετερμινιστικά διαγράμματα και έτσι επιτυγχάνεται μία υβριδική προσέγγιση της αναπαράστασης αυτών και έτσι τελικά γεννάται το ερώτημα για το ποιά από τις δύο εκδοχές είναι περισσότερο κατάλληλη για την εκπαίδευση των μαθητών.

Οι χρήστες της εφαρμογής έχουν τη δυνατότητα να δημιουργήσουν, να αποθηκεύσουν και να επεξεργαστούν συνταγές μαγειρικής οι οποίες περιγράφονται μέσω διαγραμμάτων ροής. Έτσι επιτυγχάνεται ο σχεδιασμός, ο διαμοιρασμός και η περιγραφή ενός αλγορίθμου, δηλαδή τα βήματα εκτέλεσης μιας συνταγής μαγειρικής, με τη χρήση ενός δημιουργικού, εύχρηστου και ευχάριστου περιβάλλοντος που προσπαθεί να προσομοιώσει το περιβάλλον μιας σχολικής τάξης. Επιπλέον η εφαρμογή δίνει τη δυνατότητα στους χρήστες να επεξεργάζονται και να συμμετέχουν ταυτόχρονα στο σχεδιασμό συνταγών σε πραγματικό χρόνο.

Η εφαρμογή αναπτύχθηκε με την χρήση της τεχνολογίας MERN (MongoDB, Express, React, Node), μια από τις δημοφιλέστερες και καταλληλότερες τεχνολογίες που χρησιμοποιείται παγκοσμίως τα τελευταία χρόνια για την ανάπτυξη διαδικτυακών εφαρμογών. Πιο συγκεκριμένα, η διαδικτυακή υπηρεσία είναι υλοποιημένη με την Node και Express, ενώ για την διεπαφή χρήστη επιλέχθηκε η βιβλιοθήκη React. Για τις ανάγκες αποθήκευσης δεδομένων χρησιμοποιήθηκε μη σχεσιακή βάση δεδομένων (MongoDB). Τέλος, για την εγκατάστασή της επιλέχθηκε η τεχνολογία των containers με τη βοήθεια της πλατφόρμας Docker ενώ η επικοινωνία μεταξύ πελάτη και διακομιστή επιτυγχάνεται με την χρήση REST API.

Abstract

This Diploma Thesis was implemented in the context of the design and development of the online educational platform Cooking STEAM. The purpose of the application is to offer a teaching tool that uses the STEAM methodology and focuses on teaching basic algorithmic concepts through their graphical representation with flowcharts that illustrate the steps of cooking recipes. In addition to flowcharts, this tool also allows the representation of cooking recipes in non-deterministic diagrams and thus achieves a hybrid approach to their representation and thus the question arises as to which of the two versions is more suitable for educating students.

Users of the application can create, store and edit cooking recipes which are described through flowcharts. This achieves the design, sharing, and description of an algorithm, especially the steps of executing a recipe, using a creative, easy-to-use, and enjoyable environment that tries to simulate a classroom environment. In addition, the application allows users to edit and participate in the design of recipes in real-time.

The application was developed using MERN technology (MongoDB, Express, React, Node), one of the most popular and appropriate technologies used worldwide in recent years to develop web applications. The online service specifically is implemented with Node and Express, while the React library was chosen for the user interface. A non-relational database (MongoDB) was used for data storage purposes. Finally, the container technology was selected for its installation with the help of the Docker platform, while the communication between the client and the server is achieved using a REST API.

Ευχαριστίες (Acknowledgements)

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου, κύριο Σαμολαδά Βασίλη για την επίβλεψη της παρούσας διπλωματικής εργασίας αλλά και για τις συμβουλές που μου προσέφερε. Επίσης θα ήθελα να ευχαριστήσω τους καθηγητές Δεληγιαννάκη Αντώνιο και Μανιά Αικατερίνη που δέχτηκαν να είναι στην εξεταστική επιτροπή.

Ένα ιδιαίτερο ευχαριστώ ανήκει στον μέντορά μου και ΕΔΙΠ της σχολής ΗΜΜΥ, κύριο Μουμουτζή Νεκτάριο, για την ιδέα της διπλωματικής εργασίας, την πολύτιμη βοήθειά του, την στήριξη και την καθοδήγηση που μου προσέφερε με τις συμβουλές του.

Επιπλέον, θα ήθελα να ευχαριστήσω τους γονείς μου Νίκο και Μαρίνα για όλη την ψυχική και υλική υποστήριξη κατά τη διάρκεια όλης της διαδικασίας των σπουδών μου αλλά και έναν έναν ξεχωριστά τους φίλους μου.

Τέλος, το μεγαλύτερο ευχαριστώ ανήκει στην Μαρία η οποία με στήριξε όσο κανένας άλλος από την αρχή μέχρι το τέλος.

Περιεχόμενα

1. Εισαγωγή

1.1. Γενικά

Η παρούσα διπλωματική εργασία αποσκοπεί στην υλοποίηση μιας εφαρμογής η οποία θα επεκτείνει και θα εμπλουτίσει την εκπαιδευτική δραστηριότητα Cooking STEAM δίνοντας τόσο στους μαθητές όσο και στους εκπαιδευτικούς την δυνατότητα χρήσης ενός εκπαιδευτικού εργαλείου κατάλληλα σχεδιασμένο για την συγκεκριμένη δραστηριότητα. Η δραστηριότητα Cooking STEAM υποστηρίζει τη δημιουργική εξερεύνηση αλγορίθμων με τη μορφή διαγραμμάτων ροής για την αναπαράσταση συνταγών μαγειρικής σε συνδυασμό με παραστάσεις βασισμένες στο δράμα για την παρουσίαση των συνταγών από επαγγελματίες μάγειρες σε συνδυασμό με καλλιτεχνικές παρεμβάσεις που χρησιμοποιούν μουσική, χορό και αφήγηση. Η συνολική προσέγγιση βασίζεται στο PerFECt, ένα επιτελεστικό πλαίσιο υποστήριξης της συνεργατικής μάθησης και της δημιουργικότητας που αναπτύχθηκε αρχικά για να συλλάβει αρχές σχεδιασμού κατάλληλες για την ανάπτυξη ανοικτών περιβαλλόντων μάθησης [6] έτσι ώστε να υποστηρίζει αποτελεσματικά τη συνεργατική μάθηση [7] δίνοντας έμφαση στην ανάγκη σύνδεσης της μάθησης με αποτελεσματικές κοινωνικές δομές. Αυτό το πλαίσιο έχει ήδη χρησιμοποιηθεί για να καθοδηγήσει τις εξελίξεις σε πολλά έργα αντιμετώπισης της διατήρησης της ψηφιακής κληρονομιάς και της δημιουργικής μάθησης [2][3]. Το Cooking STEAM ξεφεύγει από τη συνηθισμένη αντίληψη της εκπαίδευσης της επιστήμης των υπολογιστών που προϋποθέτει την υποχρεωτική χρήση ηλεκτρονικών υπολογιστικών συσκευών. Παρόλα αυτά η χρήση μιας κατάλληλα διαμορφωμένης και σχεδιασμένης εφαρμογής μπορεί να βοηθήσει τους μαθητές να κατανοήσουν τη σημασία της κωδικοποίησης και τις δυνατότητές της για την καλλιέργεια της δημιουργικότητας. Έτσι η εφαρμογή Cooking STEAM θα δώσει τη δυνατότητα στους μαθητές να δημιουργούν και να επεξεργάζονται διαγράμματα ροής για την αναπαράσταση συνταγών μαγειρικής σε ένα ειδικά σχεδιασμένο, επαιδευτικό και διασκεδαστικό περιβάλλον.

1.2. Παρόμοιες εφαρμογές

Μια από τις πρώτες ενέργειες που έγιναν κατά τον σχεδιασμό της εφαρμογής ήταν η αναζήτηση και η έρευνα για την εύρεση παρόμοιων εφαρμογών στο διαδίκτυο, έτσι ώστε να διαπιστωθούν σε πρώτη φάση τα μήκη κύματος στα οποία πρέπει να γίνει ο σχεδιασμός και η ανάπτυξη της εφαρμογής. Επιπλέον, είναι σημαντική η διαπίστωση του κατά πόσο πρόκειται για το σχεδιασμό και την ανάπτυξη μιας καινοτόμας εφαρμογής ή όχι . Τα αποτελέσματα της έρευνας έδειξαν ότι δεν υπάρχει αντίστοιχη εφαρμογή, η οποία να συνδυάζει τις δυνατότητες και τα σενάρια λειτουργίας που πρόκειται να προσφέρει η εφαρμογή cooking STEAM.

Φυσικά υπάρχουν κάποιες εφαρμογές που προσφέρουν μεμονωμένα τα χαρακτηριστικά και τις λειτουργίες. Για παράδειγμα, υπάρχουν πολλές εφαρμογές που δίνουν τη δυνατότητα στο χρήστη να δημιουργήσει διαγράμματα ροής, να τα αποθηκεύει στο σκληρό του δίσκο και στη συνέχεια να τα επανεπεξεργαστεί. Αντίστοιχα, υπάρχουν πολλές εφαρμογές που προσπαθούν να εισαγάγουν έναν χρήστη είτε σε έννοιες πληροφορικής είτε στην αλγοριθμική σκέψη μέσω κατάλληλων διαδραστικών παιχνιδιών. Τέλος, υπάρχουν και πιο απλές ιστοσελίδες που εξηγούν τις παραπάνω έννοιες με τρόπο τέτοιο, ώστε να γίνονται εύκολα κατανοητές σε παιδιά μικρής ηλικίας.

Γίνεται αντιληπτό ότι δεν υπάρχει καμία εφαρμογή στην αγορά που να συνδυάζει όλα τα παραπάνω με τέτοιο τρόπο ώστε να είναι διαθέσιμο ένα ολοκληρωμένο εργαλείο με το οποίο ο χρήστης θα μπορεί να δημιουργήσει, να επεξεργαστεί, να ανταλλάξει και να μοιραστεί διαγράμματα ροής σε ένα ψηφιακό περιβάλλον που προσομοιώνει μία σχολική τάξη, όπου οι έννοιες της διδασκαλίας, της μάθησης και της ομαδικότητας θα είναι τα κύρια χαρακτηριστικά.

1.3. Κίνητρα και Στόχοι Εργασίας

Στα πλαίσια της παρούσας διπλωματικής εργασίας αναπτύχθηκε ένα εκπαιδευτικό εργαλείο που ανήκει στις γλώσσες οπτικού προγραμματισμού και πιο συγκεκριμένα στην κατηγορία των γλωσσών διαγραμμάτων. Έτσι δίνεται η δυνατότητα στους χρήστες να δημιουργήσουν συνταγές μαγειρικής μέσω του γραφικού χειρισμού ενός συνόλου προγραμματιστικών στοιχείων με αποτέλεσμα ο προγραμματισμός γίνεται εφικτός μέσω οπτικών εκφράσεων, δηλαδή τα στοιχεία με τα οποία απεικονίζεται ένα διάγραμμα ροής. Έτσι επιτυγχάνεται η βαθιά κατανόηση των βασικών εννοιών της υπολογιστικής σκέψης και των αλγορίθμων και χτίζεται το απαραίτητο υπόβαθρο για τους μαθητές σε σχέση με τις παραδοσιακές διδακτικές προσεγγίσεις. Το εργαλείο αποσκοπεί στο να δωθεί η δυνατότητα στους μαθητές να μάθουν τις αλγοριθμικές έννοιες, τον προγραμματισμό και την υπολογιστική σκέψη καθοδηγώντας τους να επικεντρωθούν στη λογική και την ουσία αυτών παρά στο συντακτικό μιας γλώσσας προγραμματισμού και την κατασκευή ενός προγράμματος με αυτή κάτι το οποίο είναι σε γενικές γραμμές δυσνόητο. Αυτό έχει ως αποτέλεσμα οι διδακτικές διαδικασίες να γίνονται πιο παραγωγικές με τη χρήση ενός τέτοιου εκπαιδευτικού εργαλείου παρά με τον κλασσικό προγραμματισμό και τη χρήση του συντακτικού μιας γλώσσας. Επιπροσθέτως, οι μαθητές μπορούν να εσωτερικεύσουν και να κατανοήσουν τις προγραμματιστικές και αλγοριθμικές έννοιες πιο εύκολα προσομοιώνοντας τα βήματα εκτέλεσης μιας συνταγής μαγειρικής βάσει των οπτικών μοντέλων (διαγράμματα ροής) που δημιουργούν χρησιμοποιώντας το εργαλείο και έτσι γίνεται μια προσπάθεια αξιοποίησης όλων των δυνατοτήτων των παραπάνω εννοιών. Το εργαλείο αυτό στοχεύει στο να βοηθήσει τους μαθητές να αναπτύξουν υπολογιστική και αλγοριθμική σκέψη και να ενισχύσουν δεξιότητες που αφορούν την επίλυση προβλημάτων. Πιο συγκεκριμένα στοχεύει στην εκμάθηση της προγραμματιστικής λογικής, των αλγοριθμικών δεξιοτήτων και εννοιών μέσω της αναπαράσταση συνταγών μαγειρικής ως διαγράμματα ροής. Αυτό δίνει παράλληλα την ευκαιρία στους εκπαιδευτικούς να παροτρύνουν τους μαθητές σε νέους και πιο δημιουργικούς τρόπους εκμάθησης προγραμματισμού και αλγοριθμικών δεξιοτήτων. Τέλος, οι εκπαιδευτικοί έχουν την δυνατότητα να εντοπίσουν κενά στην κατανόηση των μαθητών, να προσαρμόσουν τη διδασκαλία και να ικανοποιήσουν τις ανάγκες του κάθε μαθητή.

1.4. Μελέτη Περίπτωσης: Συνταγές Μαγειρικής

Γίνεται αντιληπτό από τον τίτλο της εργασίας ότι η μελέτη περίπτωσης για τα διαγράμματα ροής και τις αλγοριθμικές έννοιες ανάγεται στα βήματα εκτέλεσης συνταγών μαγειρικής. Η μαγειρική αποτελεί μία προσβάσιμη, αξιόλογη και αξιομνημόνευτη διαδικασία και το πιο σημαντικό είναι ότι πέρα από τη χρήση της για διδακτικούς σκοπούς μπορεί να συνεχίζει και στο οικιακό περιβάλλον σε τακτική βάση. Αυτό κάνει τη μαγειρική ένα κατάλληλο και υπέροχο όχημα για την εισαγωγή και την απεικόνιση των αρχών της επιστήμης και των αλγορίθμων. Τα βήματα με τα οποία μπορεί να κατανεμηθεί και να διασπαστεί μία συνταγή μαγειρικής

είναι το κλειδί και ο σημαντικότερος λόγος που η μαγειρική διαδικασία καθίσταται ιδανική περίπτωση για την εκμάθηση των αλγοριθμικών εννοιών και της υπολογιστικής σκέψης. Μέχρι σήμερα, η μαγειρική χρησιμοποιείται στο πλαίσιο της εκπαίδευσης STEAM κυρίως για:

- Ανάπτυξη μαθηματικής σκέψης - εμπειρίες που σχετίζονται με την έννοια του περισσότερου και λιγότερου, τη διάρκεια μιας διαδικασίας, τον αριθμό των κουταλιών και πολλές άλλες εμπειρίες που συνδέουν τους μαθητές με τις έννοιες του όγκου, της μάζας και των κλασμάτων.
- Εκμάθηση εννοιών χημείας (παρατηρήσεις σχετικά με τις ιδιότητες των υλικών ή των συστατικών και πώς αυτές οι ιδιότητες μπορούν να αλλάξουν όταν ένα μείγμα υλικών θερμαίνεται ή ψύχεται.
- Έννοιες πολιτισμού.

Εκτός από τα παραπάνω η διαδικασία Cooking STEAM έρχεται να “εκμεταλλευτεί” και να αναδείξει και την αλγοριθμική - προγραμματιστική φύση της μαγειρικής που μπορεί να εξαχθεί από αυτήν.

1.5. Σύνοψη Εργασίας

Στο **πρώτο κεφάλαιο** γίνεται αναφορά σε παρόμοιες εφαρμογές που υπάρχουν ήδη και αναλύονται τα κίνητρα και οι στόχοι που γεννήθηκαν για την εκπόνηση της παρούσας διπλωματική εργασίας.

Στο **δεύτερο κεφάλαιο** γίνεται αναφορά στο θεωρητικό υπόβαθρο το οποίο χρειάζεται να είναι γνωστό.

Στο **τρίτο κεφάλαιο** αναλύονται τα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής αλλά και ερευνητικά ερωτήματα όσον αφορά τη χρήση των συγκεκριμένων τεχνολογιών και εργαλείων.

Στο **τέταρτο κεφάλαιο** παρουσιάζονται τα στάδια υλοποίησης, όπως για παράδειγμα η διαδικασία paper prototyping που χρειάστηκε να γίνει, η αξιολόγηση των αποτελεσμάτων και η τελική υλοποίηση που προέκυψε από αυτά.

Στο **πέμπτο κεφάλαιο** περιγράφεται ο τρόπος με τον οποίο χρησιμοποιήθηκαν τα συγκεκριμένα εργαλεία για τον σχεδιασμό και την ανάπτυξη της εφαρμογής αλλά και οι αρχιτεκτονικές που χρειάστηκαν να αναπτυχθούν.

Στο **έκτο κεφάλαιο** παρουσιάζονται τα σενάρια χρήσης της εφαρμογής και η τελική υλοποίηση αυτής.

Στο **έβδομο κεφάλαιο** αναφέρονται τα συμπεράσματα και τα αποτελέσματα που εξήχθησαν από την εκπόνηση της παρούσας διπλωματικής εργασίας αλλά και οι μελλοντικές εργασίες και επεκτάσεις του εργαλείου που αναπτύχθηκε.

2. Θεωρητικό Πλαίσιο

Σε αυτό το κεφάλαιο, θα γίνει μια προσπάθεια να αναλυθούν οι θεωρητικοί πυλώνες πάνω στους οποίους χτίστηκε και σχεδιάστηκε η εφαρμογή.

2.1. Εκπαίδευση STEAM

Αρχικά, σκόπιμο είναι να γίνει αναφορά στον όρο STEM (Science, Technology, Engineering and Mathematics) και στη συνέχεια να αναλυθεί ο όρος STEAM (Science, Technology, Engineering, Arts and Mathematics). Το STEM είναι μια προσέγγιση της εκπαίδευσης η οποία είναι σχεδιασμένη έτσι ώστε να εισαχθούν στη διδασκαλία των Μαθηματικών και των Φυσικών Επιστημών τόσο οι Τεχνολογίες όσο και η Επιστήμη των Μηχανικών. Το STEM και το STEAM αποτελούν ακρωνύμια τα οποία χρησιμοποιούνται συνήθως από άτομα σχετικά με την εκπαίδευση.

Το STEAM είναι ένας τρόπος να επωφεληθεί κανείς από αυτά που του προσφέρει το STEM και επιπλέον να ενσωματώσει την τέχνη μέσα στις 4 θεμελιώδεις αρχές του STEM. Με αυτόν το τρόπο, το STEAM πηγαίνει το STEM ένα επίπεδο παρακάτω επιτρέποντας στους μαθητές να συνδέσουν τη μάθησή τους σε αυτούς τους κρίσιμους τομείς μαζί με πρακτικές και στοιχεία σχετικά με την τέχνη. Έτσι, με αυτές τις αρχές σχεδιασμού και τα πρότυπα ολοκληρώνεται η παλέτα μάθησης που μπορούν να έχουν οι μαθητές στη διάθεσή τους. Κατ' ουσίαν, το STEAM έρχεται να αφαιρέσει όλους τους περιορισμούς και να τους αντικαταστήσει με την κατάπληξη, την κριτική και την καινοτομία. Το STEAM μπορεί να φαίνεται ενδιαφέρον και συναρπαστικό σαν ένας τρόπος εκπαίδευσης, αλλά μπορεί επίσης να είναι επικίνδυνο. Αυτό συνήθως είναι αποτέλεσμα της ελλιπούς κατανόησης του τι σημαίνει πραγματικά το STEAM τόσο στον σκοπό του όσο και στην υλοποίησή του.

Στην παρούσα διπλωματική εργασία θα αναπτυχθεί μια εφαρμογή με την οποία οι μαθητές θα μπορούν να σχεδιάζουν και να δίνουν τη δική τους λύση στα προβλήματα που θα καλούνται να λύσουν, δηλαδή τα βήματα εκτέλεσης μιας συνταγής μαγειρικής μέσω διαγραμμάτων ροής. Τα προβλήματα αυτά θα διατυπώνονται συνήθως από έναν άλλο χρήστη της εφαρμογής, όπως για παράδειγμα ένα δάσκαλο ή συντονιστή, ο οποίος θα εφαρμόζει την εκπαίδευση STEAM με τη χρήση της εφαρμογής.

2.2. Ανάπτυξη Εκπαιδευτικής Εφαρμογής

Επειδή το μεγαλύτερο μέρος των χρηστών της εφαρμογής θα είναι συνήθως μαθητές της πρωτοβάθμιας ή της δευτεροβάθμιας εκπαίδευσης, είναι απαραίτητο να υλοποιηθεί ένα εκπαιδευτικό εργαλείο ειδικά σχεδιασμένο με τέτοιο τρόπο, ώστε να είναι εύχρηστο, κατανοητό και διασκεδαστικό για το συγκεκριμένο κοινό.

Η εφαρμογή έχει δύο βασικά χαρακτηριστικά τα οποία πρέπει να παίξουν μεγάλο ρόλο κατά τον σχεδιασμό της. Το πρώτο είναι ότι απευθύνεται σε παιδιά μικρής ηλικίας (9-12) και το δεύτερο είναι ότι η γενικότερη θεματολογία είναι η μαγειρική. Αυτό σημαίνει ότι τα γραφικά, η εμπειρία του χρήστη (user experience ή UX) και η αλληλεπίδραση του χρήστη (user interface ή UI) με την εφαρμογή εξαρτάται άμεσα από αυτά τα χαρακτηριστικά, θα τα συνδυάζει άρτια και τελικά θα γίνει η προσπάθεια να αποδοθεί και να επιτευχθεί το επιθυμητό αποτέλεσμα.

2.3. Αναπαράσταση Αλγορίθμων

Το διάγραμμα ροής (flowchart) είναι ένα κοινού τύπου διάγραμμα που αναπαριστά έναν αλγόριθμο ή μια διαδικασία. Τα βήματα συνήθως συμβολίζονται με κουτιά διαφόρων ειδών και σχημάτων που συνδέονται μεταξύ τους με βέλη. Αυτή η διαγραμματική παρουσίαση μπορεί να δώσει λύση βήμα προς βήμα σε ένα γνωστό πρόβλημα, όπως για παράδειγμα μια συνταγή μαγειρικής. Τα δεδομένα αναπαριστώνται σε κουτιά και τα βέλη δείχνουν τη ροή των δεδομένων. Τα διαγράμματα ροής χρησιμοποιούνται στην ανάλυση, το σχεδιασμό, την τεκμηρίωση ή τον έλεγχο μιας διαδικασίας ή ενός προγράμματος σε διάφορα πεδία. Στην περίπτωση της εκτέλεσης των βημάτων μιας συνταγής μαγειρικής μέσω διαγραμμάτων ροής θα επιτευχθεί η προσπάθεια εισαγωγής του χρήστη-μαθητή στη λογική με την οποία μπορεί να αναπαραστήσει και να σχεδιάσει μια τέτοια διαδικασία. Αυτό θα έχει ως κύριο σκοπό την εξοικείωσή του με τα διαγράμματα ροής και την αναπαράσταση των αλγορίθμων.

Μέχρι σήμερα στη διαδικασία cooking STEAM χρησιμοποιούνται τα κλασσικά διαγράμματα ροής. Μέσα από την εκπόνηση της παρούσας διπλωματικής εργασίας τα διαγράμματα flow-based φαίνεται να παρουσιάζουν μεγαλύτερο ενδιαφέρον για την αναπαράσταση των συνταγών μαγειρικής. Ουσιαστικά τα flow-based διαγράμματα αποτελούν μια μη ντετερμινιστική προσέγγιση για την αναπαράσταση ενός αλγορίθμου, δηλαδή κάθε κόμβος του διαγράμματος μπορεί να έχει παραπάνω από μία εναλλακτικές εξόδους, εφόσον σχεδιαστεί κατάλληλα, σε αντίθεση με τα διαγράμματα ροής που αποτελούν κατά κύριο λόγο μια ντετερμινιστική προσέγγιση, δεδομένου ότι κάθε έξοδος οδηγεί σε συγκεκριμένη και αυστηροποιημένη είσοδο σχήματος.

2.4. Υπολογιστική Σκέψη (Computational Thinking)

Η υπολογιστική σκέψη είναι το βήμα που προηγείται του προγραμματισμού. Είναι η διαδικασία ανάλυσης ενός προβλήματος σε απλούστερα βήματα τα οποία ακόμη και ένας υπολογιστής θα μπορούσε να κατανοήσει. Οι υπολογιστές παίρνουν τις οδηγίες λέξη προς λέξη, το οποίο μερικές φορές οδηγεί ακόμη και σε δυσνόητα αποτελέσματα. Εάν δεν παρέχουμε στους υπολογιστές οδηγίες που είναι ακριβείς και λεπτομερείς, τότε ο αλγόριθμος που προσπαθούμε να σχεδιάσουμε πιθανόν να μην συμπεριλάβει ενέργειες που οι περισσότεροι άνθρωποι θεωρούν δεδομένες. Έτσι η υπολογιστική σκέψη βοηθά τους μαθητές να αναπτύξουν δεξιότητες που είναι ελκυστικές για μελλοντικές ευκαιρίες απασχόλησης. Η επιστήμη των υπολογιστών είναι η ταχύτερα αναπτυσσόμενη αγορά εργασίας και οι δεξιότητες των μαθητών στον προγραμματισμό και την κωδικοποίηση τους κάνει ιδιαίτερα περιζήτητους για εργασία. Αυτές οι δεξιότητες επιτυχίας είναι το κλειδί για να κατανοήσουμε γιατί η υπολογιστική σκέψη είναι τόσο πολύτιμη.

Επιπλέον, η ανάπτυξη της κριτικής σκέψης και των συναισθηματικών ικανοτήτων προετοιμάζουν τους μαθητές για μακροπρόθεσμη επιτυχία. Όταν τα παιδιά μαθαίνουν δεξιότητες υπολογιστικής σκέψης, τα βοηθά να αναπτύξουν δεξιότητες σημαντικές όχι μόνο για τα θέματα STEAM, αλλά και για τις κοινωνικές επιστήμες και τις γλωσσικές τέχνες. Όταν τα παιδιά αναπτύσσουν υπολογιστικές δεξιότητες είναι σε θέση να διατυπώσουν ένα πρόβλημα και να σκεφτούν λογικά. Τους βοηθά να αναλύσουν τα ζητήματα που αντιμετωπίζουν και να προβλέψουν τι μπορεί να συμβεί στο μέλλον. Επιπλέον, τους βοηθά να εξερευνήσουν την αιτία και το αποτέλεσμα και να αναλύσουν πώς οι πράξεις τους ή οι

πράξεις άλλων επηρεάζουν τη δεδομένη κατάσταση. Αυτές οι δεξιότητες μπορούν να έχουν ισχυρό αντίκτυπο στα παιδιά και στο πώς διαχειρίζονται τις σχέσεις τους με τον περίγυρό τους.

2.5. Paper Prototyping

Το paper prototyping είναι μία ευρέως γνωστή μέθοδος για το σχεδιασμό, τον έλεγχο και την βελτίωση της αλληλεπίδρασης του χρήστη με μια διαδικτυακή εφαρμογή, μια διαδικτυακή ιστοσελίδα και γενικότερα οποιουδήποτε συστήματος εμπεριέχει το στοιχείο της διεπαφής ανθρώπου-υπολογιστή. Από τα μέσα της δεκαετίας του ΄90 μέχρι και σήμερα η τεχνική αυτή χρησιμοποιείται από τη μικρότερη μέχρι και τη μεγαλύτερη εταιρεία ανάπτυξης λογισμικού και αποτελεί αναπόσπαστο κομμάτι της διαδικασίας ανάπτυξης και σχεδιασμού των προϊόντων της.

Κατά την διαδικασία και την εφαρμογή του paper prototyping πιθανοί χρήστες της εφαρμογής καλούνται να εκτελέσουν ορισμένες ρεαλιστικές διαδικασίες-εργασίες αλληλεπιδρώντας όμως με στιγμιότυπα της διεπαφής (interface) σχεδιασμένα σε χαρτί. Τα συγκεκριμένα στιγμιότυπα συνήθως διαχειρίζεται ένας εκπρόσωπος της εφαρμογής, ο οποίος ουσιαστικά παίζει το ρόλο του υπολογιστή χωρίς όμως να εξηγεί στους χρήστες πώς προορίζεται να λειτουργήσει η διεπαφή με την οποία αλληλεπιδρούν.

Η διαδικασία της εκτέλεσης του paper prototyping προϋποθέτει ορισμένα βήματα και αποφάσεις που πρέπει να ληφθούν, βάσει των οποίων θα προκύψει η ομαλή και επιτυχημένη εφαρμογή της μεθόδου. Αρχικά, θα πρέπει να αποφασιστεί ο τύπος χρήστη που εκπροσωπεί καταλληλότερα το κοινό της εφαρμογής. Στη συνέχεια, θα πρέπει να καθοριστούν οι διαδικασίες-εργασίες που πρόκειται να εκτελέσουν οι χρήστες κατά την εκτέλεση της μεθόδου. Το επόμενο βήμα είναι η υλοποίηση των στιγμιοτύπων όλων των οθονών και παραθύρων με τα οποία πρόκειται να αλληλεπιδράσει ο χρήστης κατά την εκτέλεση των εργασιών.

Αφού σχεδιαστεί το paper prototyping, ξεκινάει η διαδικασία εκτέλεσής του. Αρχικά, καλείται ο κατάλληλος εκπρόσωπος του κοινού της εφαρμογής από τον οποίο ζητείται να αλληλεπιδρά στις αποφάσεις του, αγγίζοντας με το δάχτυλό του ή με κάποιο μολύβι πάνω στα υποτιθέμενα κουμπιά και συνδέσμους ή γράφοντας με το μολύβι πάνω στα υποτιθέμενα πλαίσια κειμένων. Όπως αναφέρθηκε παραπάνω ο εκπρόσωπος της εφαρμογής που παίζει το ρόλο του υπολογιστή είναι υπεύθυνος καθ΄ όλη τη διάρκεια της διαδικασίας να ανταποκρίνεται στις αποφάσεις του χρήστη, όπως για παράδειγμα στη μετάβαση από ένα στιγμιότυπο σε ένα άλλο μετά την επιλογή κάποιου υποτιθέμενου κουμπιού. Κατά τη διάρκεια της διαδικασίας εύκολα μπορεί να διαπιστωθεί ποια κομμάτια της διεπαφής λειτουργούν σωστά και ποια όχι βάσει των επιλογών του χρήστη. Μπορεί εύκολα να τροποποιηθεί η διεπαφή σχεδιάζοντας κάτι επιπλέον στο χαρτί ή αφαιρώντας κάτι άλλο έτσι ώστε να ελεγχθεί επί τόπου ο εναλλακτικός σχεδιασμός και αν αυτός ανταποκρίνεται καλύτερα. Κατά το πέρας της διαδικασίας έχουν ληφθεί χρήσιμα συμπεράσματα για το σχεδιασμό της εφαρμογής που έχουν βασιστεί σε πραγματικούς και πιθανούς χρήστες της εφαρμογής.

3. Τεχνολογίες

Σε αυτό το κεφάλαιο θα γίνει αναφορά στα βασικά εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής αλλά και στη σχετική έρευνα που έγινε για την επιλογή αυτών. Το σύνολο των παρακάτω εργαλείων και τεχνολογιών αποτελούν την τεχνολογική στοίβα που επιλέχθηκε.

3.1. Ερευνητικά Ερωτήματα

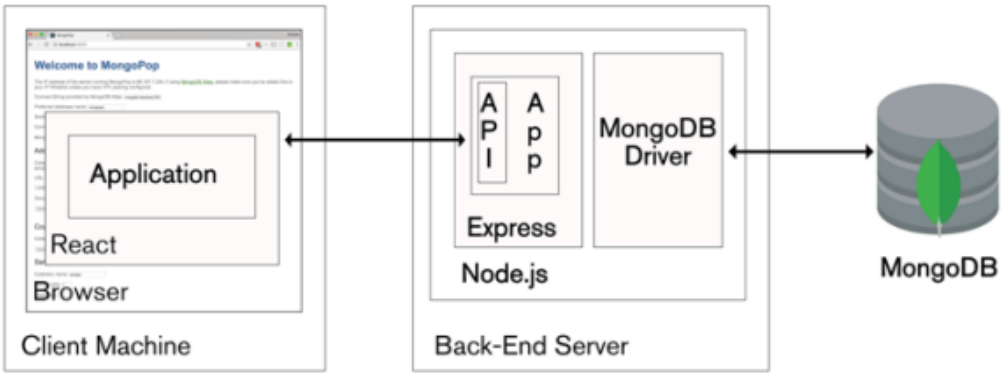
Ένα από τα σημαντικότερα ερωτήματα που προέκυψαν για την υλοποίηση της εφαρμογής ήταν η επιλογή και η αντίστοιχη έρευνα των εργαλείων με τα οποία θα γίνει η ανάπτυξη.

3.1.1. Τεχνολογική Στοίβα

Η πρώτη και σημαντικότερη επιλογή αφορούσε την τεχνολογική στοίβα (tech stack) που θα χρησιμοποιηθεί, δηλαδή το συνδυασμό των τεχνολογιών (γλώσσες προγραμματισμού, βιβλιοθήκες, frameworks, APIs) που θα συνθέσουν την εφαρμογή. Μια τεχνολογική στοίβα θα μπορούσε να χωριστεί σε 3 μέρη. Το πρώτο μέρος αφορά το front end το οποίο περιέχει όλες τις τεχνολογίες που χρειάζονται για να δημιουργηθεί η διεπαφή του χρήστη και τα γραφικά της εφαρμογής. Το δεύτερο μέρος είναι το back end, δηλαδή το σύνολο των τεχνολογιών που υποστηρίζουν την ιστοσελίδα στο μέρος του εξυπηρετητή (server), το λειτουργικό και τις βάσεις δεδομένων. Το τρίτο μέρος αφορά τα APIs που αναπτύσσονται από τους προγραμματιστές των εφαρμογών και χρησιμοποιούνται για την σύνδεση του front end και του back end, έτσι ώστε να επικοινωνήσουν μεταξύ τους τα δύο αυτά μέρη. Πολλές φορές για την ανάπτυξη μιας εφαρμογής χρησιμοποιούνται ορισμένα 3rd party APIs, τα οποία αφορούν έτοιμες τεχνολογίες που μπορεί να συμπεριλάβει κανείς στην ανάπτυξη της εφαρμογής του. Αφορούν συνήθως υπηρεσίες για ανταλλαγή μηνυμάτων εντός εφαρμογής, για αποστολή email ή ακόμα και για τραπεζικές συναλλαγές. Οι τεχνολογίες αυτές λειτουργούν ως εργαλεία που αλληλοσυμπληρώνονται για να δημιουργήσουν αποτελεσματικά ένα λειτουργικό σύστημα ή πιο συγκεκριμένα μια διαδικτυακή εφαρμογή. Βέβαια, η επιλογή της τεχνολογικής στοίβας ανέκαθεν αποτελούσε πονοκέφαλο για τους προγραμματιστές και τους σχεδιαστές ενός έργου, αφού οι επιλογές είναι πολλές και τα κριτήρια ακόμα περισσότερα.

3.2. MERN Stack

Η τεχνολογία MERN (MERN Stack) αποτελεί ένα ακρωνύμιο για τις λέξεις MongoDB, Express, React.js και Node.js οι οποίες απευθύνονται σε λογισμικά και τεχνολογίες ανοιχτού κώδικα. Η τεχνολογική στοίβα αποτελείται από μία βάση δεδομένων (MongoDB), ένα back-end framework (Express), μία βιβλιοθήκη front-end (React.js) και ένα runtime περιβάλλον (Node.js) στην πλευρά του εξυπηρετητή αντίστοιχα. Η Node.js επιτρέπει στη στοίβα να είναι πολύ αποτελεσματική ως υπηρεσία φιλοξενίας ιστού δεδομένου ότι μπορεί να εξυπηρετεί μεγάλο αριθμό πελατών ταυτόχρονα. Η συγκεκριμένη στοίβα τεχνολογίας προγραμματίζεται μόνο με τη χρήση της γλώσσας JavaScript, κάτι το οποίο προσδίδει σχετική ασφάλεια στον προγραμματιστή αφού χρειάζεται να χρησιμοποιήσει μόνο μια γλώσσα προγραμματισμού. Κάθε μια από αυτές τις τεχνολογίες θα αναλυθούν περισσότερο και πιο λεπτομερώς στη συνέχεια του κεφαλαίου. Στην εικόνα 1 απεικονίζεται η αρχιτεκτονική που χρησιμοποιείται στην τεχνολογική στοίβα MERN.



Εικόνα 1: Αρχιτεκτονική MERN Stack

3.2.1. MongoDB

Η MongoDB αποτελεί ένα δωρεάν και ανοικτού κώδικα σύστημα διαχείρισης βάσεων δεδομένων. Πρόκειται για μία NoSQL, μη σχεσιακή, βάση δεδομένων και χρησιμοποιείται στη τεχνολογική στοίβα MERN. Στην MongoDB τα έγγραφα είναι συλλογές κλειδιών που μπορούν να αποθηκεύσουν διάφορους τύπους δεδομένων, ενώ οι συλλογές βοηθούν στην οργάνωση των εγγράφων. Κάθε έγγραφο στη MongoDB έχει ένα μοναδικό αναγνωριστικό αντικειμένου (Object Identifier) που δημιουργείται αυτόματα και μέσω αυτού είναι δυνατή η πρόσβαση στο έγγραφο. Τα δεδομένα αποθηκεύονται στη βάση με τη μορφή BSON (Binary JavaScript Object Notation) ως συλλογές (collections) εγγράφων (documents). Αυτό οφείλεται στο γεγονός ότι η MongoDB υποστηρίζει την εισαγωγή διαφόρων τύπων δεδομένων σε διαφορετικές γλώσσες προγραμματισμού, δεδομένου του κατάλληλου driver για την γλώσσα που υποστηρίζεται. Στη συνέχεια ο driver μετατρέπει τους διαφορετικούς τύπους δεδομένων που χρησιμοποιούνται στις εφαρμογές ,σε τύπους BSON . Αυτός ο σχεδιασμός επιτρέπει ταχύτερη σάρωση κατά την υποβολή ερωτημάτων (querying) και αυξημένη αποτελεσματικότητα στην αποθήκευση δεδομένων. Υπάρχουν αρκετές βιβλιοθήκες που είναι ανεπτυγμένες σε περιβάλλον Node.js και προσφέρουν ποικίλες δυνατότητες για την χρήση MongoDB. Μία από τις πιο γνωστές είναι η βιβλιοθήκη Mongoose, η οποία μάλιστα χρησιμοποιήθηκε και σε αυτή την εργασία.

Querying

Η MongoDB βασίζεται στην γλώσσα προγραμματισμού JavaScript και η γλώσσα για την υποβολή ερωτημάτων (queries) βασίζεται σε JSON. Χρησιμοποιώντας JSON ως γλώσσα για υποβολή ερωτημάτων υπάρχει μια σχετική ευκολία για το querying. Αυτό οφείλεται στο γεγονός ότι ο προγραμματιστής μπορεί να δημιουργήσει περισσότερα ερωτήματα καθορίζοντας το είδος της λειτουργίας σε ένα αντικείμενο JSON, δηλαδή αν πρόκειται για δημιουργία, διαγραφή, παραλαβή, ανανέωση ή αναζήτηση σε έγγραφο. Από την άλλη πλευρά, οι βάσεις δεδομένων στις οποίες χρησιμοποιείται η γλώσσα SQL είναι πιο περιορισμένες και πιο αυστηρές στην υποβολή αιτημάτων. Η αποθήκευση δεδομένων σε ένα αντικείμενο αντί για την αποθήκευση αυτών σε μορφή σχέσεων ή πινάκων, όπως είναι στις παραδοσιακές SQL βάσεις δεδομένων, δίνει στον προγραμματιστή μία σημαντική ευελιξία στη δόμηση μοντέλων.

Μοντέλα Δεδομένων

Στη MongoDB ένα έγγραφο (document) ουσιαστικά είναι η αντίστοιχη γραμμή ενός πίνακα σε μία SQL βάση δεδομένων, ενώ η συλλογή (collection) είναι ένας πίνακας. Η MongoDB υποστηρίζει πίνακες και ένθετα αντικείμενα (nested objects), επιτρέποντας στον προγραμματιστή να έχει πολύπλοκα μοντέλα σε ιεραρχία μέσα σε ένα έγγραφο.

Σχήματα (Schemas)

Στην MongoDB υπάρχουν τα μη προκαθορισμένα σχήματα, το οποίο σημαίνει ότι δεν χρειάζεται να προσδιορίζεται το μέγεθος ή ο τύπος των δεδομένων πριν την αποθήκευση των δεδομένων. Αυτό επιτρέπει στον προγραμματιστή να κάνει διάφορες αλλαγές χωρίς την ανάγκη αναδιάρθρωσης του μοντέλου κάθε φορά. Έτσι η ανάπτυξη εφαρμογών με χρήση της MongoDB γίνεται γρηγορότερη.

Κλιμακωσιμότητα (Scalability)

Όσον αφορά τον όγκο δεδομένων και την κλιμακωσιμότητα (scalability) μιας εφαρμογής, συνήθως οι προγραμματιστές έχουν δυο επιλογές. Θα πρέπει να επενδύσουν σε ένα μεγαλύτερο μηχάνημα στο οποίο μπορούν να αποθηκεύονται μεγαλύτεροι όγκοι δεδομένων ή να κατανέμουν τον όγκο των δεδομένων σε πολλά και διαφορετικά μηχανήματα. Η πρώτη επιλογή μπορεί να αποβεί πολύ κοστοβόρα για να μπορέσει κάποιος να πετύχει το επιθυμητό scalability. Η MongoDB είναι σχεδιασμένη για τη δεύτερη επιλογή, κατανέμοντας τα δεδομένα σε πολλούς servers εξισορροπώντας αυτόματα το φορτίο δεδομένων. Το σύστημα δρομολόγησης της MongoDB χρησιμοποιείται για να διασφαλίσει ότι τα δεδομένα διαβάζονται και εγγράφονται στο σωστό μηχάνημα κάθε φορά.

3.2.2. Express.js

Το Express.js αποτελεί ένα framework για την Node.js και μάλιστα το δημοφιλέστερο αφού η ταχύτητα και η ευελιξία του κατέκτησε γρήγορα τον κόσμο της Node.js. Πρόκειται για ένα server side framework που παρέχει ένα ισχυρό σύνολο δυνατοτήτων για εφαρμογές ιστού και για κινητές συσκευές. Το εν λόγω framework μπορεί να υποστηρίξει τόσο το backend μιας μονολιθικής εφαρμογής όσο και τον σχεδιασμό μιας αρχιτεκτονικής με micro services, όπως για παράδειγμα ένα RESTful API, αφού παρέχει μία πληθώρα μεθόδων χρηστικότητας HTTP και διάφορων middlewares. Ένα από τα πλεονεκτήματά του είναι ότι δίνει την ελευθερία στον προγραμματιστή να προσαρμόσει και να αναπτύξει προγράμματα στις δικές του ανάγκες χωρίς να έχει περιορισμούς από το ίδιο το framework. Για παράδειγμα θα μπορούσε κανείς να δουλέψει τόσο με σχεσιακές βάσεις δεδομένων όσο και με μη σχεσιακές – document based βάσεις δεδομένων. Το Express.js είναι σχετικά ελαφρύ και για τη χρήση του απαιτείται η γλώσσα JavaScript. Αυτό αποτελεί ένα ακόμα πλεονέκτημα αφού διατηρείται η κοινή χρήση της γλώσσας προγραμματισμού σε όλη τη στοίβα που θα χρησιμοποιήσουμε.

3.2.3. React.js

Η React.js αποτελεί μια δωρεάν και ανοιχτού κώδικα front-end JavaScript βιβλιοθήκη η οποία χρησιμοποιείται για τη δημιουργία διεπαφών χρήστη και βασίζεται σε τέτοιου είδους στοιχεία . Δημιουργήθηκε από τον Jordan Walke, μηχανικό λογισμικού της Facebook.

Χρησιμοποιήθηκε για πρώτη φορά στη ροή ενημερώσεων του Facebook το 2011 και αργότερα στο Instagram το 2012. Σήμερα, διατηρείται από τη Meta (πρώην Facebook) και μια κοινότητα μεμονωμένων προγραμματιστών και εταιρειών.

Βασικό χαρακτηριστικό της React.js είναι ότι μπορεί να χρησιμοποιηθεί ως βάση για την ανάπτυξη εφαρμογών μιας σελίδας ή για κινητές συσκευές. Ωστόσο, ασχολείται μόνο με τη διαχείριση κατάστασης και την απόδοση αυτής της κατάστασης στο εικονικό Μοντέλο Αντικειμένου Εγγράφου (Virtual DOM). Αυτό γίνεται δημιουργώντας μια δομή δεδομένων εντός της μνήμης και στη συνέχεια υπολογίζονται οι διαφορές για την αποτελεσματική ενημέρωση του DOM που εμφανίζεται το πρόγραμμα περιήγησης. Αυτό δίνει τη δυνατότητα στον προγραμματιστή να γράφει κώδικα σαν να αποδίδεται ολόκληρη η σελίδα σε κάθε αλλαγή, ενώ οι ρουτίνες της React αποδίδουν μόνο τα υποστοιχεία του DOM που πραγματικά αλλάζουν.

Η δημιουργία εφαρμογών React συνήθως απαιτεί τη χρήση πρόσθετων βιβλιοθηκών για δρομολόγηση, καθώς και ορισμένες λειτουργίες από την πλευρά του πελάτη. Λόγω της δημοτικότητας και της ανάπτυξής της υπάρχουν διαθέσιμες πάρα πολλές πρόσθετες βιβλιοθήκες ανοιχτού κώδικα που διευκολύνουν τη δουλειά του προγραμματιστή. Στη συνέχεια θα γίνει μία αναφορά των βασικών χαρακτηριστικών της βιβλιοθήκης.

JavaScript XML (JSX)

Η JSX αποτελεί μια επέκταση του συντακτικού της JavaScript και συνήθως χρησιμοποιείται σε συνδυασμό με την React για την περιγραφή της απεικόνισης της διεπαφής χρήστη. Η JSX διακατέχει όλη την λειτουργικότητα της JavaScript και μέσω αυτής παράγονται τα στοιχεία της React. Η χρησιμότητα της ύπαρξης της JSX μπορεί να διαπιστωθεί καλύτερα περιγράφοντας μία βασική λειτουργικότητα που έχει η React. Στην React η λογική της απόδοσης (rendering logic) είναι άρρηκτα συνδεδεμένη με την λογική της διεπαφής χρήστη, όπως στην περίπτωση της αλλαγής της κατάστασης (state), του χειρισμού των γεγονότων (events) αλλά και της προετοιμασίας των δεδομένων για απεικόνιση. Η React δεν απαιτεί τη χρήση της JSX, αλλά αποτελεί χρήσιμο οπτικό βοήθημα κατά τη διάρκεια εργασίας με τη διεπαφή χρήστη εντός κώδικα JavaScript. Επιπλέον, με τη χρήση της JSX εμφανίζονται περισσότερα χρήσιμα μηνύματα λάθους και ειδοποιήσεις εφόσον η εφαρμογή βρίσκεται σε στάδιο ανάπτυξης.

Πρακτικά η χρήση της JSX επιτρέπει την μεταφορά HTML στοιχείων μέσα από μια μεταβλητή. Επιπλέον, επιτρέπει μια μεταβλητή ή μια συνάρτηση που επιστρέφει ένα ή περισσότερα στοιχεία να γραφτούν μέσα σε άγκιστρα. Μετά την μεταγλώττιση (compiling), οι εκφράσεις JSX γίνονται κανονικές κλήσεις συναρτήσεων της JavaScript και μετατρέπονται σε αντικείμενα της JavaScript. Αυτό έχει ως αποτέλεσμα η JSX να μπορεί να χρησιμοποιηθεί ακόμα και εντός δομών ελέγχου, να εκχωρηθεί σε μεταβλητές ή ακόμα να χρησιμοποιηθεί ως όρισμα και να επιστραφεί από συναρτήσεις.

```
let name = "Panagiotis";
let greeting = <p>Hi, nice to see you {name} </p>
```

Εικόνα 2: Παράδειγμα χρήσης JSX

Components & Props

Στην React τα components επιτρέπουν το διαχωρισμό της διεπαφής του χρήστη σε ανεξάρτητα, αυτοτελή και επαναχρησιμοποιήσιμα κομμάτια, επιτρέποντας στον προγραμματιστή να τα σχεδιάζει χωρίς να τα περιπλέκει μεταξύ τους. Τα components ουσιαστικά λειτουργούν ως συναρτήσεις της JavaScript, καθώς δέχονται ορίσματα, τα οποία ονομάζονται props (properties - ιδιότητες) και επιστρέφουν React elements, καθορίζοντας έτσι τι θα απεικονίζεται στη διεπαφή. Ο απλούστερος τρόπος για τον ορισμό ενός component είναι η σύνταξη μιας συνάρτησης Javascript:

```
export default function Hello(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

Εικόνα 3: Παράδειγμα function component

Όπως φαίνεται στην εικόνα 3 η συνάρτηση δέχεται ως όρισμα το “props” μέσα στο οποίο μπορούν να υπάρχουν διάφορα ορίσματα έτσι ώστε να περνούν μέσα στη συνάρτηση. Στη συνέχεια επιστρέφει το React Element που κατασκευάστηκε με ή χωρίς την επεξεργασία των props. Τέτοιου είδους components ονομάζονται function components δεδομένου ότι αποτελούν συναρτήσεις της JavaScript. Κάθε component αποτελεί ένα μικρό και αυτοτελές κομμάτι της διεπαφής χρήστη πραγματοποιώντας συγκεκριμένες λειτουργίες. Ένα component, λόγω της σύνθεσής του (component composing), έχει τη δυνατότητα να συμπεριλαμβάνεται σε άλλο component, να συνδυάζεται με πολλά components ή να αναφέρεται σε άλλο component, κάτι το οποίο έχει ως αποτέλεσμα να μπορεί να χρησιμοποιηθεί το ίδιο component abstraction για κάθε επίπεδο λεπτομέρειας. Έτσι, μεταξύ ορισμένων components μπορεί να αποδοθεί ακόμα και η σχέση parent-child. Μέσω της εξαγωγής τους (component extraction), αν κάποιο component είναι τόσο περίπλοκο που καθίσταται δύσκολη η επεξεργασία του μπορεί να χωριστεί σε μικρότερα components. Αυτό είναι ιδιαίτερα χρήσιμο σε μεγαλειώδεις εφαρμογές, αφού η πρόσβαση σε μία μεγάλη παλέτα επαναχρησιμοποιήσιμων components μπορεί να φανεί ιδιαίτερα κερδοφόρα. Ένας χρήσιμος κανόνας σχετικά με τη χρήση των components αφορά τα μέρη της εφαρμογής που χρησιμοποιούνται συχνά, όπως για παράδειγμα ορισμένα κουμπιά, να θεωρούνται επαναχρησιμοποιήσιμα components.

```jsx
function Hello(props) {
  return <h1>Hello, {props.name}</h1>;
}

function Age(props) {
  return <h1>You are {props.age} years old!</h1>;
}

export default function App() {
  return (
    <div className="App">
      <Hello name="panagiotis" />
      <Age age="26" />
    </div>
  );
}
```

Εικόνα 4: Παράδειγμα component composition και extraction

Όσον αφορά τα props, σκόπιμο είναι να αναφερθεί ότι προορίζονται μόνο για ανάγνωση. Αυτό σημαίνει ότι ένας component δεν μπορεί να αλλάξει τα δικά του props. Αυτές οι συναρτήσεις χαρακτηρίζονται ως pure (αμιγείς), δεδομένου ότι σε κάθε περίπτωση επιστρέφουν το ίδιο αποτέλεσμα όταν δέχονται το ίδιο όρισμα, σε αντίθεση με τις impure συναρτήσεις που επιχειρούν το αντίθετο, δηλαδή την αλλαγή της εισόδου τους. Υπάρχει ένας αυστηρός κανόνας της React σύμφωνα με τον οποίο όλα τα components πρέπει να συμπεριφέρονται ως pure συναρτήσεις σε σχέση με τα props τους.

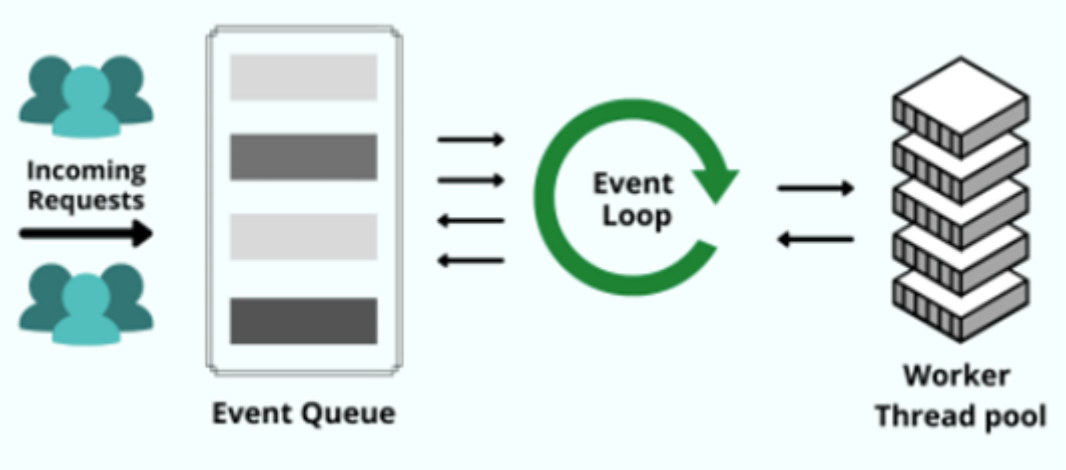
Hooks

Τα Hooks είναι συναρτήσεις που επιτρέπουν το «γάντζωμα» του state και των Lifecycle μεθόδων από τα ίδια components συναρτήσεων που χρησιμοποιούνται, επιτρέποντας έτσι τη σύνδεση επαναλαμβανόμενης συμπεριφοράς σε components. Τα hooks δεν λειτουργούν μέσα σε κλάσεις, έτσι επιτρέπουν τη χρήση της React χωρίς κλάσεις. Τα βασικότερα Hooks είναι οι συναρτήσεις useState και useEffect, αλλά μπορεί κανείς να δημιουργήσει και τα δικά του hooks για την επαναχρησιμοποίηση της κατάστασης της συμπεριφοράς μεταξύ διαφορετικών components. Κάποιοι βασικοί κανόνες σχετικά με την χρήση των hooks είναι ότι πρέπει να δηλώνονται στο πάνω μέρος των συναρτήσεων και να μην καλούνται σε συνθήκες και επαναληπτικούς βρόγχους. Επίσης, θα πρέπει να καλούνται αυστηρά από React συναρτήσεις και όχι γενικότερα από οποιαδήποτε javaScript συνάρτηση.

```jsx
import React, { useState } from 'react';

export function Example() {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```

Εικόνα 5: Παράδειγμα χρήσης του useState React hook

3.2.4. Node.js



Εικόνα 6: Αρχιτεκτονική Node.js

Η Node.js είναι ένα περιβάλλον (Runtime Environment) στο οποίο εκτελείται ο κώδικας της JavaScript χωρίς την βοήθεια του browser. Δημιουργήθηκε από τον Ryan Dahl, βασίζεται στη μηχανή JavaScript V8 του Chrome και χρησιμοποιείται αποκλειστικά για την ανάπτυξη back-end εφαρμογών.

Η Node.js φημίζεται για την ταχύτητά της και την εξοικονόμηση πόρων που επιτρέπει να επιτυγχάνονται με τη χρήση της. Αυτό οφείλεται κυρίως στην ασύγχρονη φύση της JavaScript όπου αυτό αποτελεί μεγάλο πλεονέκτημα για έναν server. Πρακτικά, η Node.js κάνει χρήση μόνο μιας διαδικασίας (single-process) χωρίς να δημιουργεί καινούρια νήματα σε κάθε αίτηση (single-threaded) και έτσι ακολουθεί ασύγχρονη επικοινωνία εισόδου/εξόδου (I/O). Λόγω αυτού δεν γίνεται αποκλεισμός όλου του κώδικα όταν αναμένεται απάντηση από ένα γεγονός. Αυτό επιτυγχάνεται με τη χρήση των callback functions. Όταν ο επεξεργαστής περιμένει ένα γεγονός (π.χ. διάβασμα αρχείου) ορίζεται μια τέτοια συνάρτηση που θα εκτελεστεί όταν ολοκληρωθεί αυτό το γεγονός. Έτσι ο επεξεργαστής παραμένει ελεύθερος χωρίς να παγώνει. Όταν ολοκληρωθεί αυτό το γεγονός, καλείται η callback συνάρτηση η

οποία θα κρατήσει τον επεξεργαστή κατειλημμένο μέχρι να ολοκληρωθεί η διαδικασία. Έτσι επιτυγχάνεται η ελαχιστοποίηση της αναμονής και η μικρή απαίτηση σε μνήμη, επιτρέποντας στην Node.js να διαχειρίζεται ταυτόχρονα χιλιάδες συνδέσεις με έναν μόνο επεξεργαστή. Παρόλα αυτά υπάρχει και τρόπος συγχρονισμού των γεγονότων που χρησιμοποιούνται σε λιγότερες περιπτώσεις.

Η μάχη των μεγάλων εταιρειών, όπως η Google και η Microsoft, για το ποια θα βγάλει τον πιο γρήγορο φυλλομετρητή ιστοσελίδων (Google Chrome, Microsoft Edge κλπ) οδήγησε στη δημιουργία πολύ δυνατών μηχανών για την JavaScript γεγονός που καθιστά και την Node.js πολύ γρήγορη. Η Node.js είναι υλοποιημένη σε γλώσσα JavaScript γεγονός που επιτρέπει τη χρήση μιας και μόνο γλώσσας προγραμματισμού, τόσο στο front-end όσο και στο back-end. Ένα ακόμα πλεονέκτημα της Node.js είναι και η ευκολία επικοινωνίας με object βάσεις δεδομένων , όπως η MongoDB. Κατά την ανάκτηση ή προσθήκη δεδομένων στη βάση, τα δεδομένα παρέχονται σε μορφή JSON και έτσι δεν χρειάζεται καμία άλλη μετατροπή τύπου δεδομένων αφού η Node.js μπορεί να επεξεργαστεί με ευκολία τέτοιου τύπου δεδομένα.
Παρά το γεγονός ότι η Node.js είναι σχετικά νέα, κατέχει ένα μεγαλειώδες οικοσύστημα το οποίο έχει σχηματιστεί από μία τεράστια κοινότητα ανοιχτού λογισμικού. Αποτέλεσμα αυτού είναι η ανάπτυξη και δημιουργία χιλιάδων εκατοντάδων πακέτων και βιβλιοθηκών ανοιχτού κώδικα για όλες τις ανάγκες που μπορούν να παρουσιαστούν σε έναν προγραμματιστή. Έτσι κάθε προγραμματιστής μπορεί ελεύθερα να έχει πρόσβαση σε αυτά τα πακέτα και τις βιβλιοθήκες με τη χρήση του διαχειριστή πακέτων npm (node package manager). Στο κεφάλαιο 4 θα γίνει αναφορά και ανάλυση των npm πακέτων που χρησιμοποιήθηκαν κατά την ανάπτυξη της εφαρμογής.

3.3. Εικονικοποίηση (Virtualization)

Στη συγκεκριμένη ενότητα του τρίτου κεφαλαίου θα παρουσιαστεί συνοπτικά η τεχνολογία της εικονικοποίησης ονόματι containerization που χρησιμοποιήθηκε για την εγκατάσταση της εφαρμογής. Εικονικοποίηση ορίζεται ως η τεχνολογία με την οποία είναι δυνατή η εξομοίωση της λειτουργίας ενός αντικειμένου ή ενός πόρου με την αντίστοιχη λειτουργία του φυσικού αντικειμένου. Δηλαδή, η εικονικοποίηση αναφέρεται στη δημιουργία μιας εικονικής πηγής, όπως για παράδειγμα ενός λειτουργικού συστήματος, ενός εξυπηρετητή ή ενός δικτύου. Παραμένει στο προσκήνιο της πληροφορικής για δεκαετίες και εφαρμόζεται σε διάφορα επίπεδα συστημάτων, όπως για παράδειγμα στην εικονικοποίηση του επιπέδου του λειτουργικού, του υλικού αλλά και της εικονικοποίησης εξυπηρετητή.

3.3.1. Containerization

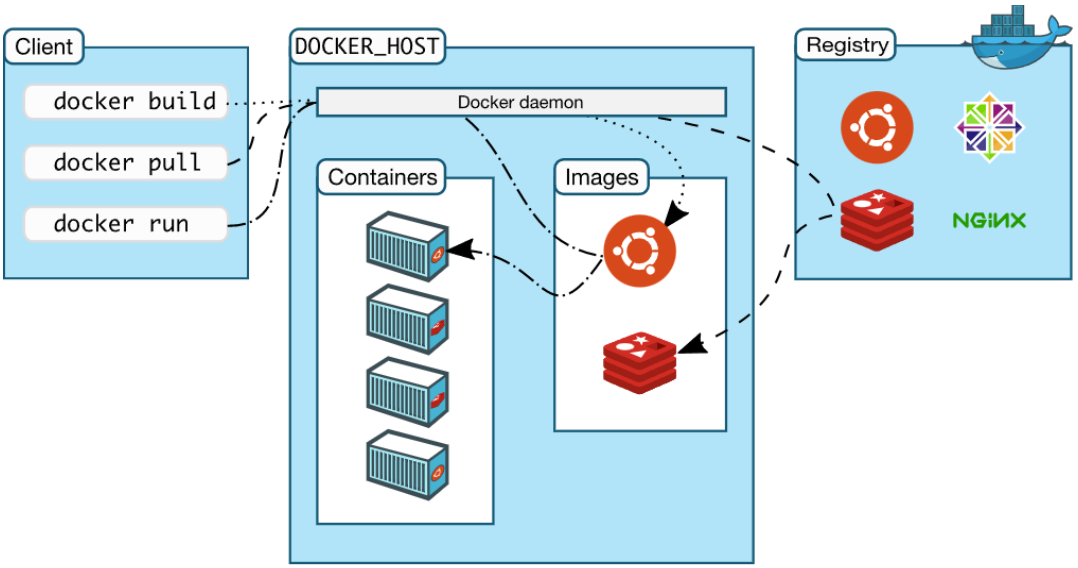
Η συγκεκριμένη τεχνολογία πρωτοεμφανίστηκε το 2008 όταν ο Linux kernel εισήγαγε τα C-groups (control groups). Αυτό έπαιξε σημαντικό ρόλο για όλες τις τεχνολογίες containerization που υπάρχουν σήμερα όπως το Docker, το Cloud Foundry, το Rocket και άλλα. Πρακτικά κάθε container αποτελεί μία μονάδα λογισμικού που πακετάρει όλο τον κώδικα και τις εξαρτήσεις του με τέτοιο τρόπο που το ανεξαρτητοποιεί από την υποδομή στην οποία λειτουργεί. Αυτό έχει ως αποτέλεσμα η εφαρμογή να μπορεί να εγκατασταθεί εύκολα και με συνέπεια σε ένα τοπικό μηχάνημα, ένα ιδιωτικό κέντρο δεδομένων (data center), ένα δημόσιο cloud ή οποιαδήποτε άλλη υποδομή υπολογιστών. Ακόμη μπορεί να

τρέξει σε οποιοδήποτε περιβάλλον λειτουργώντας αρμονικά. Αυτό αποδίδει στα Containers το πλεονέκτημα της φορητότητας.

Συχνά τα containers συγκρίνονται με τις εικονικές μηχανές (VM), καθώς παρέχουν παρόμοιες δυνατότητες. Τα containers όμως χρησιμοποιούν λιγότερους πόρους και είναι πιο ελαφριά σε σχέση με τις εικονικές μηχανές. Αυτό οφείλεται στο γεγονός ότι με τα containers εικονικοποιείται μόνο το λειτουργικό σύστημα που τρέχει πάνω από τον φυσικό εξυπηρετητή και κάθε container μοιράζεται αυτό το λειτουργικό σύστημα και τις βιβλιοθήκες του. Επιπλέον σε σχέση με τις εικονικές μηχανές, τον ρόλο του Hypervisor τον παίζει μια μηχανή για containers, όπως το Docker, η οποία είναι εγκατεστημένη πάνω από το λειτουργικό σύστημα. Επομένως τα κύρια πλεονεκτήματα των containers σε σχέση με τις εικονικές μηχανές είναι το κέρδος και η υψηλή αποδοτικότητα όσον αφορά την μνήμη, τον επεξεργαστή αλλά και τον αποθηκευτικό χώρο που καταναλώνουν.

3.3.2. Docker

Το Docker είναι ένα δημοφιλές, ανοιχτού κώδικα έργο το οποίο είναι βασισμένο πάνω στα Linux containers. Είναι ουσιαστικά μια μηχανή container που χρησιμοποιεί τα χαρακτηριστικά του Linux kernel και δημιουργεί containers πάνω από ένα λειτουργικό σύστημα με αποτέλεσμα να αυτοματοποιεί την ανάπτυξη εφαρμογών στο container. Διαθέτει μια αποτελεσματική ροή για την μετακίνηση της εφαρμογής από τους υπολογιστές των προγραμματιστών και τα περιβάλλοντα δοκιμής, στην παραγωγή. Η ροή αυτή ακολουθήθηκε για την εγκατάσταση της εφαρμογής στον εξυπηρετητή και για την επιτυχή ολοκλήρωσή της χρησιμοποιήθηκαν τα κομμάτια όπως απεικονίζονται στη παρακάτω εικόνα (βλ. Εικόνα 7) .



Εικόνα 7: Τα μέρη του Docker

Το Docker ακολουθεί μια αρχιτεκτονική πελάτη-εξυπηρετητή. Το docker daemon έχει τον ρόλο του εξυπηρετητή και είναι υπεύθυνο για όλες τις ενέργειες που έχουν σχέση με τα containers. Το docker daemon, αντίστοιχα με κάθε αρχιτεκτονική πελάτη-εξυπηρετητή, δέχεται αιτήματα/εντολές από τον Docker πελάτη μέσω του Client ή ενός API. Ο Docker πελάτης μπορεί να λειτουργεί στον ίδιο πάροχο με τον docker daemon ή σε διαφορετικό από

αυτό που τρέχει το Docker daemon. Έτσι, αρχικά έγινε εγκατάσταση της Docker μηχανής και κατά επέκταση του docker daemon στον εξυπηρετητή.

Οι εικόνες (images) αποτελούν θεμελιώδες συστατικό του Docker, δεδομένου ότι τα containers χτίζονται από αυτές. Οι εικόνες έχουν τη δυνατότητα να διαμορφώνονται ανάλογα με την εφαρμογή και να χρησιμοποιηθούν σαν πρότυπα για τη δημιουργία νέων containers. Για παράδειγμα, για τη δημιουργία ενός container μιας React εφαρμογής, χρειάζεται μόνο να γίνει η προσθήκη του φακέλου της εφαρμογής στην εικόνα ενός εξυπηρετητή ιστού και έπειτα να δημιουργηθεί το container. Πιο συγκεκριμένα, για την εφαρμογή χρησιμοποιήθηκαν οι εικόνες των mongo και node. Για τις ανάγκες της εφαρμογής τροποποιήθηκαν οι εν λόγω εικόνες και έπειτα πάνω σε αυτές δημιουργήθηκαν τα τρία συνολικά containers.

Οι εικόνες Docker αποθηκεύονται στο Docker Registry. Χρησιμοποιώντας αυτό, παρέχεται η δυνατότητα ανάπτυξης εικόνων, καθώς και η διαμοίρασή τους ενώ το αρχείο που παράγεται μπορεί να είναι είτε ιδιωτικό είτε δημόσιο. Το Docker διαθέτει μια υπηρεσία που ονομάζεται Docker hub και επιτρέπει στον εκάστοτε προγραμματιστή να ανεβάσει και να κατεβάσει προς και από μια κεντρική τοποθεσία. Αν η αποθήκη του χρήστη είναι δημόσια, όλες οι εικόνες που βρίσκονται μέσα σε αυτή είναι προσβάσιμες από άλλους χρήστες του Docker hub. Στο Docker hub ο κάθε προγραμματιστής έχει τη δυνατότητα να αναπτύξει εικόνες τοπικά στον υπολογιστή του και στη συνέχεια να τις ανεβάσει στο Docker hub.

Τέλος, το container αποτελεί το περιβάλλον εκτέλεσης για το Docker. Τα containers δημιουργούνται από εικόνες που είναι ένα εγγράψιμο επίπεδό τους. Υπάρχει η δυνατότητα ένα container το οποίο περιέχει μια εφαρμογή και έχει εισαχθεί στο Docker hub, να χρησιμοποιηθεί σαν εικόνα ώστε πάνω σε αυτό να δημιουργηθούν νέα containers.

4. Αρχική Υλοποίηση και Αξιολόγηση

Σε αυτό το κεφάλαιο θα παρουσιαστεί η διαδικασία paper prototyping που εφαρμόστηκε για τον σχεδιασμό της εφαρμογής στα αρχικά στάδια ανάπτυξης, καθώς και οι αξιολογήσεις των διαδικασιών σχεδίασης από πιθανούς χρήστες της εφαρμογής. Για κάθε διαδικασία και αξιολόγηση αναλύεται ο τρόπος εκτέλεσής της και τα αποτελέσματα που προέκυψαν. Ένα από τα σημαντικότερα κομμάτια της παρούσας διπλωματικής εργασίας αποτέλεσε η διαδικασία paper prototyping που εφαρμόστηκε σε διάφορα στάδια της υλοποίησης. Όπως αναλύθηκε και σε προηγούμενο κεφάλαιο το paper prototyping είναι μια μέθοδος και χρησιμοποιήθηκε για τον σχεδιασμό, τον έλεγχο και τη βελτίωση του User Interface (UI) της εφαρμογής σε διάφορα στάδια της ανάπτυξής της. Η διαδικασία που περιγράφεται παρακάτω αποτελεί τη σημαντικότερη και πιο καθοριστική διαδικασία paper prototyping που έγινε κατά το διάστημα ανάπτυξης και σχεδίασης της εφαρμογής. Εκτός από αυτή ακολούθησαν κι άλλες πειραματικές διαδικασίες με τις οποίες παράχθηκαν σημαντικά αποτελέσματα που βοήθησαν στην τελική υλοποίηση. Το κοινό που έλαβε μέρος σε αυτές τις διαδικασίες ήταν πιο ευρύ, καθώς σε αυτό συμμετείχαν τόσο παιδιά από μεγαλύτερο ηλικιακό φάσμα όσο και δάσκαλοι και εκπαιδευτικοί.

4.1. Εφαρμογή Paper Prototyping

Κατά το πρώτο στάδιο του paper prototyping η εφαρμογή παρουσιάστηκε στη σχολική ημερίδα «cSTEAM – cooking STEAM Μικροί Επιστήμονες στην Κουζίνα» που διεξήχθη στο Μουσείο Σχολικής Ζωής στα Χανιά τον Νοέμβριο του 2021. Στη συγκεκριμένη ημερίδα έλαβαν μέρος δεκάδες μαθητές της πρώτης τάξης του Γυμνασίου. Μέχρι σήμερα κάθε μία από τις ημερίδες cooking STEAM ακολουθεί τρία μέρη.

Αρχικά γίνεται μία καλλιτεχνική εισαγωγή με αφήγηση και τραγούδι αλλά και δραματοποιημένη αφήγηση με μαριονέτα. Σε αυτό το μέρος ενσωματώνονται πληροφορίες σχετικά με την τοπική διατροφή αλλά και ανάγνωση πληροφοριών που αφορούν τα διατροφικά προϊόντα. Στη συνέχεια, γίνεται παρουσίαση συνταγής με διάγραμμα ροής, η εκτέλεση της συνταγής και στο τέλος τα παιδιά δοκιμάζουν από το φαγητό που ετοιμάστηκε. Κατά την εκτέλεση και το σερβίρισμα, τα παιδιά συζητούν με τον μάγειρα για τα υλικά και την αξία τους, τις μεθόδους μαγειρέματος και ό,τι άλλες απορίες έχουν. Αυτό το διαδραστικό μέρος έχει μεγάλες δυνατότητες και δίνει την ευκαιρία για περαιτέρω συζήτηση ιδεών και εννοιών που παρουσιάζονται στο πρώτο μέρος. Στο τελευταίο μέρος λαμβάνουν χώρα δραστηριότητες μαθηματικών, επιστήμης και χρήση υπολογιστών με τη μορφή μικρών ασκήσεων.

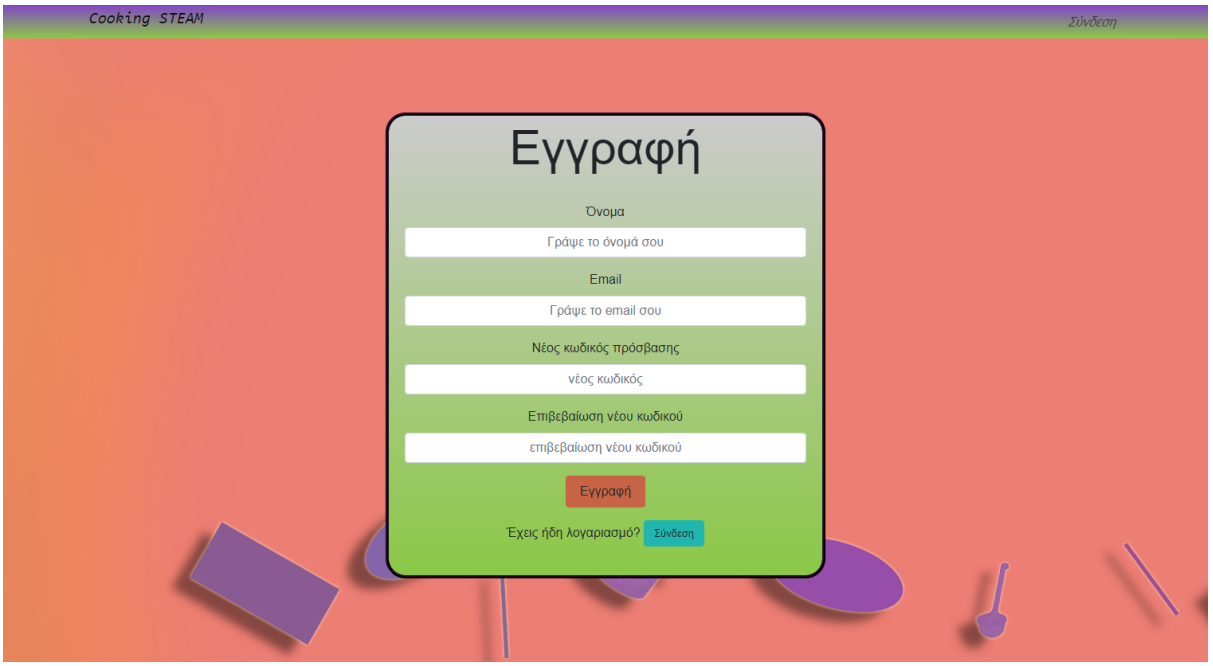
Έτσι στο τρίτο και τελευταίο μέρος δόθηκε η ευκαιρία να εφαρμοστεί το paper prototyping για την εφαρμογή cooking STEAM. Οι μαθητές μετά από αυτή τη διαδικασία ήταν πλέον σε θέση να αλληλεπιδράσουν με ένα ψηφιακό εργαλείο που θα μπορούσε να τους δώσει τη δυνατότητα να δημιουργήσουν και να επεξεργαστούν τέτοιου είδους διαγράμματα ροής με τη χρήση ηλεκτρονικού υπολογιστή ή tablet. Έτσι ακολούθησε η μέθοδος paper prototyping της εφαρμογής κατά την οποία μερικοί από τους μαθητές είχαν την ευκαιρία να γνωρίσουν την πρώτη έκδοση της εφαρμογής, να αλληλεπιδράσουν και να την αξιολογήσουν. Μέσω της διαδικασίας προέκυψαν διάφορες αλλαγές και βελτιστοποιήσεις για την εφαρμογή. Πιο συγκεκριμένα, παρουσιάστηκαν στους μαθητές διάφορα στιγμιότυπα της εφαρμογής σε κόλλες Α4 και ακολούθησαν τα βήματα της διαδικασίας του paper prototyping. Στο τέλος της διαδικασίας εξήχθησαν χρήσιμες πληροφορίες και σημειώθηκαν διάφορες αλλαγές που επισήμαναν οι ίδιοι.

Οι σημαντικότερες αλλαγές αφορούσαν τη διεπαφή χρήστη που είχε σχέση με τον editor των διαγραμμάτων ροής, αναλυτικότερα:

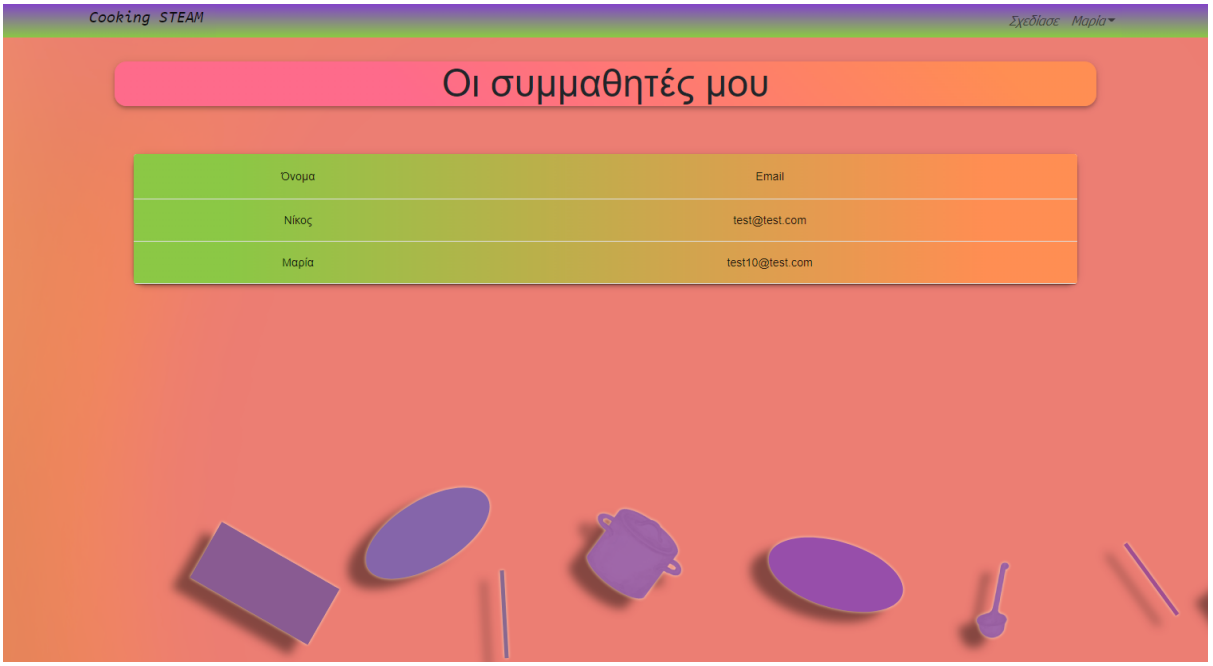
1. Στην έκδοση της εφαρμογής που παρουσιάστηκε στα παιδιά υπήρχαν δύο διαφορετικά σύμβολα για την αναπαράσταση του συμβόλου της Αρχής και του Τέλους. Αυτό όπως διαπιστώθηκε δεν ήταν πολύ ευανάγνωστο και κατανοητό στα παιδιά, επομένως θα έπρεπε να χρησιμοποιηθεί μόνο ένα κοινό σύμβολο.
2. Επιπροσθέτως, οι μαθητές έκριναν ότι κατά την εισαγωγή ενός συμβόλου στον Editor είναι προτιμότερο να σέρνουν το σύμβολο προς την επιφάνεια εργασίας παρά να επιλέγουν το σύμβολο με ένα κλικ και να τοποθετείται αυτόματα.
3. Κατά την εκτέλεση της διαδικασίας διαπιστώθηκε ότι ορισμένες επιλογές που υπήρχαν στη στήλη με τα εργαλεία αποπροσανατόλιζαν τα παιδιά, επομένως θα έπρεπε να αφαιρεθούν.

4. Η ενότητα «Οι συμμαθητές μου» αποτελούσε δυσνόητο στοιχείο για τους περισσότερους από τους μαθητές επομένως θα έπρεπε να αφαιρεθεί η συγκεκριμένη λογική.
5. Τέλος, υπήρχε πρόβλημα σχετικά με την επιλογή της γραμματοσειράς και των κουμπιών που είχαν επιλεχθεί. Επομένως, έπρεπε να γίνει εκ νέου σχεδιασμός και για αυτά.

Στις παρακάτω εικόνες απεικονίζεται ο αρχικός σχεδιασμός της εφαρμογής Cooking STEAM που παρουσιάστηκε στη διαδικασία paper prototyping.

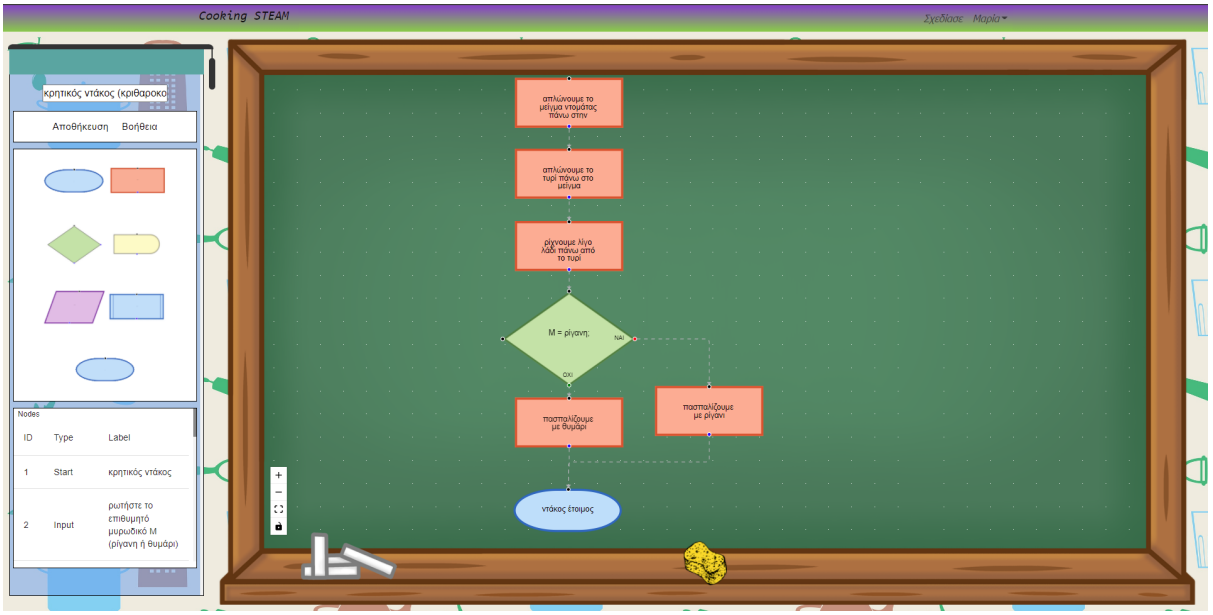


Εικόνα 8: Σελίδα εγγραφής χρήστη κατά την 1η έκδοση

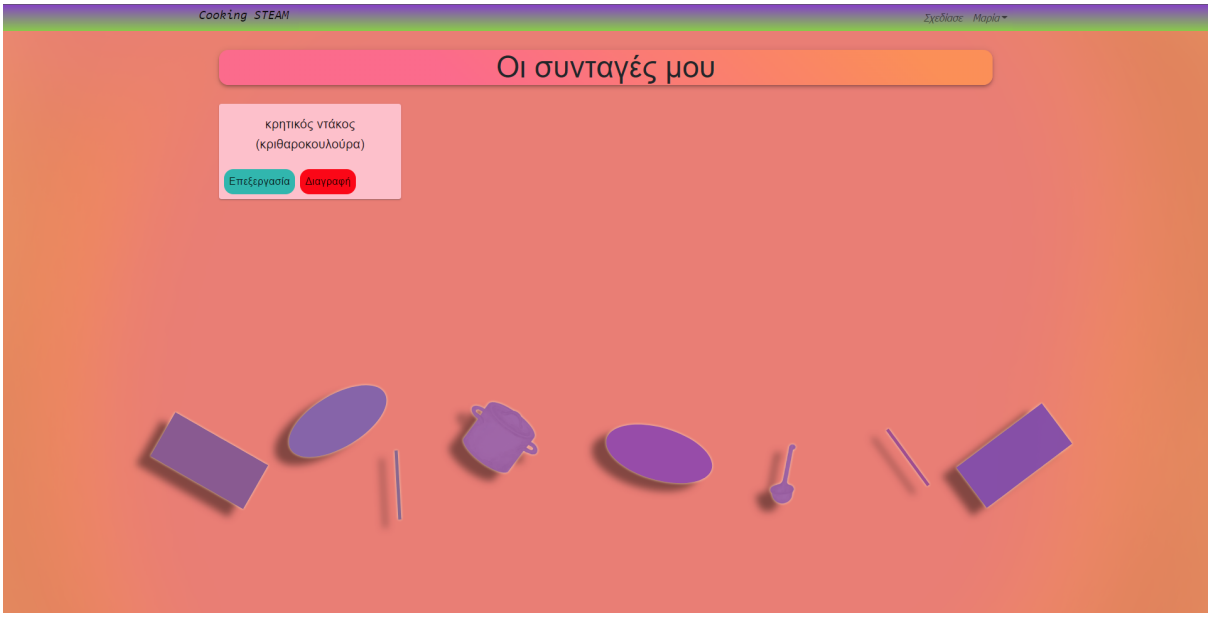


Εικόνα 9: Σελίδα απεικόνισης συμμαθητών χρήστη κατά την 1η έκδοση

Εικόνα 10: Διεπαφή editor κατά την 1η έκδοση



Εικόνα 11: Διεπαφή editor κατά την 1η έκδοση



Εικόνα 12: Διεπαφή profile κατά την 1η έκδοση

Ωστόσο πέρα από τους μαθητές, αντλήθηκαν χρήσιμες πληροφορίες σχετικά με την βελτιστοποίηση της διεπαφής του χρήστη της εφαρμογής και από την παρέμβαση της ψυχολόγου που βρισκόταν στην ημερίδα. Πιο συγκεκριμένα έκρινε ότι τα χρώματα που επιλέχθηκαν στην εφαρμογή ήταν άγρια και έντονα για ένα παιδί αυτής της ηλικίας και μας παρότρυνε να επανεξετάσουμε την επιλογή των χρωμάτων. Έτσι γεννήθηκε η ανάγκη μιας σχετικής έρευνας που αφορά τη χρήση πιο κατάλληλων χρωμάτων η οποία περιγράφεται σε επόμενο κεφάλαιο.

4.2. Αποτελέσματα Αξιολόγησης

Τα αποτελέσματα της αξιολόγησης από τη διαδικασία paper prototyping ήταν πολύ σημαντικά. Ουσιαστικά επισημάνθηκαν αδυναμίες της αρχικής έκδοσης της εφαρμογής που αφορούσαν προβλήματα με τα γραφικά και τα χρώματα που είχαν επιλεχθεί για την εφαρμογή. Επιπλέον, ξεκαθαρίστηκαν οι λειτουργικότητες που ήταν απαραίτητες και λειτουργούσαν θετικά με αποτέλεσμα να αφαιρεθούν εκείνες που δημιουργούσαν προβληματισμό και έλλειψη κατανόησης αυτών από τους μαθητές. Τέλος, διαπιστώθηκαν και πιο λεπτομερή προβήματα που αφορούσαν την κεντρική λειτουργικότητα της εφαρμογής, δηλαδή τον Editor, έτσι ώστε τελικά να υπάρχει ένα καταλληλότερο και καλύτερο για την εμπειρία του χρήστη εργαλείο. Έτσι συνοπτικά οι αλλαγές που χρειάστηκε να γίνουν είναι οι εξής:

- Επαναδιαμόρφωση της ενότητας των εργαλείων του Editor που αφορούν την χρήση μόνο ενός κόμβου που αφορά την απεικόνιση της Αρχής/Τέλους ενός διαγράμματος ροής.
- Διαγραφή της ενότητας “Nodes”.
- Ανάπτυξη λογικής “drag n drop” για την εισαγωγή των κόμβων στον Editor.
- Αφαίρεση της ενότητας «Οι συμμαθητές μου» από την κεντρική οθόνη του χρήστη.
- Προσθήκη ενότητας «Οι τάξεις μου» και «Χρήστες εφαρμογής» στην κεντρική οθόνη του χρήστη.

- Επαναδιαμόρφωση και σχεδιασμός των κουμπιών της εφαρμογής.
- Αλλαγή στη γραμματοσειρά και εισαγωγή μιας καταλληλότερης.
- Ανάπτυξη λογικής για επιλογή μενού όσον αφορά την εναλλαγή της θεματολογίας της συνολικής Εφαρμογής.
- Ανάπτυξη λογικής για απεικόνιση στατιστικών συνατγής (χρόνος εκτέλεσης).

4.3. Αναλυτική περιγραφή περιπτώσεων χρήσης της εφαρμογής Cooking STEAM

Λαμβάνοντας υπόψη και τις παρατηρήσεις που δόθηκαν κατά το paper prototyping αποφασίστηκε η τελική υπολοποίηση της εφαρμογής. Παρακάτω περιγράφονται αναλυτικά οι περιπτώσεις χρήσης της εφαρμογής Cooking STEAM.

Περίπτωση Χρήσης: Δημιουργία συνταγής
Ο χρήστης θέλει να δημιουργήσει ένα νέο διάγραμμα ροής μιας συνταγής. Επιλέγει το αντίστοιχο κουμπί από το πλέγμα που βρίσκεται στην οθόνη του profile. Στη συνέχεια μεταφέρεται στην οθόνη του Editor. Ο χρήστης μπορεί να επιλέξει τη συγκεκριμένη ενέργεια και από το μενού που βρίσκεται πάνω δεξιά στην οθόνη.

Περίπτωση Χρήσης: Προβολή συνταγών
Ο χρήστης θέλει να δει τις συνταγές που έχει υλοποιήσει. Επιλέγει την μετάβαση στην οθόνη του profile του και στο αντίστοιχο πλέγμα απεικονίζονται όλες οι συνταγές που έχει υλοποιήσει ο ίδιος ή ανήκουν στη τάξη του.

Περίπτωση Χρήσης: Επιλογή συνταγής για επεξεργασία
Ο χρήστης θέλει να επεξεργαστεί ένα υπάρχον διάγραμμα ροής συνταγής. Επιλέγει το αντίστοιχο κουμπί από το πλέγμα που βρίσκεται στην οθόνη του profile. Στη συνέχεια μεταφέρεται στην οθόνη του Editor και επεξεργάζεται την αντίστοιχη συνταγή.

Περίπτωση Χρήσης: Διαγραφή συνταγής
Ο χρήστης θέλει να επεξεργαστεί ένα υπάρχον διάγραμμα ροής συνταγής. Επιλέγει το αντίστοιχο κουμπί από το πλέγμα που βρίσκεται στην οθόνη του profile. Στη συνέχεια μεταφέρεται στην οθόνη του Editor και επεξεργάζεται την αντίστοιχη συνταγή.

Περίπτωση Χρήσης: Αποθήκευση/Ενημέρωση συνταγής
Ο χρήστης θέλει να αποθηκεύσει ένα νέο διάγραμμα ροής ή να ενημερώσει ένα υπάρχον. Επιλέγει το αντίστοιχο κουμπί από το την οθόνη του Editor. Στη συνέχεια μεταφέρεται στην οθόνη του profile όπου εμφανίζεται η νέα συνταγή.

Περίπτωση Χρήσης: Εισαγωγή/Διαγραφή στοιχείων στον editor
Ο χρήστης θέλει να εισάγει ή να διαγράψει στοιχεία στο διάγραμμα ροής. Επιλέγει από το πλέγμα των εργαλείων και σέρνει τον αντίστοιχο κόμβο μέσα στην επιφάνεια εργασίας. Εάν θέλει να διαγράψει κάποιο στοιχείο το επιλέγει και πατάει το κουμπί διαγραφής. Εάν θέλει να ενώσει δύο στοιχεία επιλέγει με τον κέρσορα την αρχή και το αντίστοιχο τέλος.

Περίπτωση Χρήσης: Εναλλαγή σε flow-based ή flowchart
Ο χρήστης θέλει να κάνει εναλλαγή του πλέγματος των εργαλείων από flowchart σε flow-based ή και αντίστροφα. Επιλέγει το αντίστοιχο κουμπί και εκτελείται η αυτόματη εναλλαγή όπου διαμορφώνεται κατάλληλα το πλέγμα εργαλείων.

Περίπτωση Χρήσης: Επιλογή βοήθειας
Ο χρήστης θέλει να δει την βοήθεια που του παρέχεται στα πλαίσια της καθοδήγησής του για την υλοποίηση ενός διαγράμματος. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των εργαλείων στη οθόνη του Editor.

Περίπτωση Χρήσης: Δημιουργία τάξης
Ο χρήστης θέλει να δημιουργήσει μία νέα τάξη μαθητών. Επιλέγει το αντίστοιχο κουμπί από το πλέγμα που βρίσκεται στην οθόνη του profile. Στη συνέχεια εμφανίζεται η αντίστοιχη φόρμα δημιουργίας νέας τάξης και την καταχωρεί.

Περίπτωση Χρήσης: Προβολή τάξεων
Ο χρήστης θέλει να δει τις τάξεις που έχει δημιουργήσει. Επιλέγει την μετάβαση στην οθόνη του profile του και στο αντίστοιχο πλέγμα απεικονίζονται όλες οι τάξεις που έχει δημιουργήσει.

Περίπτωση Χρήσης: Επιλογή τάξης για επεξεργασία
Ο χρήστης θέλει να επεξεργαστεί μία τάξη. Επιλέγει την μετάβαση στην οθόνη του profile του και στο αντίστοιχο πλέγμα απεικονίζονται όλες οι τάξεις που έχει δημιουργήσει. Στη συνέχεια επιλέγει το αντίστοιχο κουμπί επεξεργασίας και μεταφέρεται στην σελίδα της συγκεκριμένης τάξης.

Περίπτωση Χρήσης: Εισαγωγή/Εξαγωγή μαθητή από τάξη
Ο χρήστης θέλει να εισαγάγει ή να εξάγει έναν μαθητή από την τάξη. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των μαθητών της τάξης από την σελίδα της συγκεκριμένης τάξης.

Περίπτωση Χρήσης: Δημιουργία συνταγής που ανήκει στην τάξη (subscribe)
Ο χρήστης θέλει να δημιουργήσει μία συνταγή για όλους τους μαθητές της τάξης. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των συνταγών της τάξης από την σελίδα της συγκεκριμένης τάξης. Στη συνέχεια μεταφέρεται στον Editor. Όταν αποθηκεύσει τη συνταγή δημιουργείται ένα αντίγραφο για κάθε μαθητή της τάξης.

Περίπτωση Χρήσης: Δημιουργία real-time συνταγής
Ο χρήστης θέλει να δημιουργήσει μία real-time συνταγή για όλους τους μαθητές της τάξης. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των συνταγών της τάξης από την σελίδα της συγκεκριμένης τάξης. Στη συνέχεια μεταφέρεται στον Editor. Όταν αποθηκεύσει τη συνταγή δημιουργείται ένα κοινό αντίγραφο για όλους τους μαθητές της τάξης.

Περίπτωση Χρήσης: Διαγραφή συνταγής που ανήκει στην τάξη
Ο χρήστης θέλει να διαγράψει μία συνταγή που ανήκει στης τάξη. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των συνταγών της τάξης από την σελίδα της συγκεκριμένης τάξης.

Περίπτωση Χρήσης: Διαγραφή τάξης
Ο χρήστης θέλει να διαγράψει μία τάξη. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των τάξεων στη σελίδα του profile.

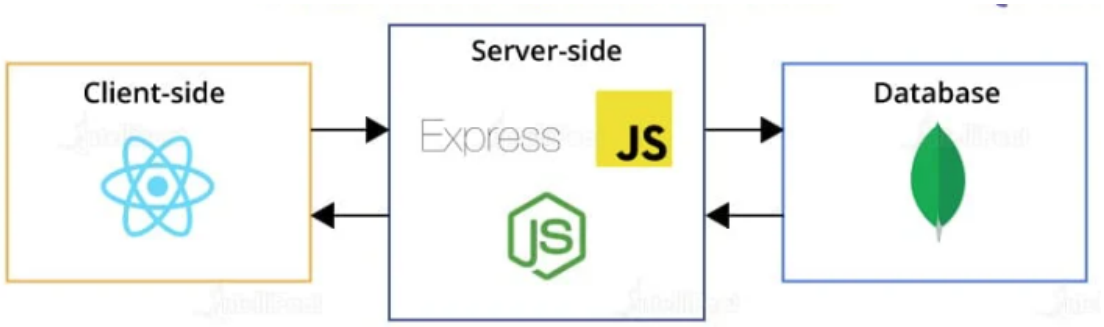
Περίπτωση Χρήσης: Δημιουργία χρήστη
Ο χρήστης θέλει να δημιουργήσει έναν νεό χρήστη. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των χρηστών στη σελίδα του profile. Στη συνέχεια εμφανίζεται η φόρμα δημιουργίας νέου χρήστη και κατά την καταχώρηση δημιουργείται ο νέος χρήστης.

Περίπτωση Χρήσης: Επεξεργασία χρήστη
Ο χρήστης θέλει να επεξεργαστεί έναν χρήστη. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των χρηστών στη σελίδα του profile. Στη συνέχεια επιλέγει το κουμπί της ενημέρωσης.

Περίπτωση Χρήσης: Διαγραφή χρήστη
Ο χρήστης θέλει να διαγράψει έναν χρήστη. Επιλέγει το αντίστοιχο κουμπί που βρίσκεται στο πλέγμα των χρηστών στη σελίδα του profile και στη συνέχεια ο χρήστης διαγράφεται.

Περίπτωση Χρήσης: Αλλαγή θέματος της εφαρμογής
Ο χρήστης θέλει να αλλάξει τη θεματολογία χρωμάτων στην εφαρμογή. Επιλέγει τη θεματολογία από το αντίστοιχο μενού που βρίσκεται πάνω και αριστερά στην οθόνη.

5. Ανάπτυξη Εφαρμογής



Εικόνα 13: Αρχιτεκτονική Εφαρμογής και απεικόνιση των βιβλιοθηκών/frameworks

Σε αυτό το κεφάλαιο θα γίνει ανάλυση και περιγραφή της αρχιτεκτονικής που εφαρμόστηκε, βάσει της οποίας σχεδιάστηκε, δομήθηκε και αναπτύχθηκε η εφαρμογή. Η εφαρμογή ουσιαστικά αποτελεί ένα κατανεμμημένο σύστημα που ακολουθεί την 3-επιπέδων αρχιτεκτονική με micro-services (micro-service-based 3-tier architecture). Όπως φαίνεται και στην εικόνα 13, η εφαρμογή μπορεί εύκολα να διασπαστεί σε τρία μέρη, δηλαδή τον πελάτη (client-side) που αποτελεί το presentation tier, τον εξυπηρετητή (server-side) που αποτελεί το middle tier και τη βάση δεδομένων (database) που αποτελεί το data tier. Το μέρος του πελάτη αποτελεί το front-end κομμάτι της εφαρμογής, δηλαδή την React εφαρμογή με την οποία αλληλεπιδρούν οι χρήστες με τη βοήθεια του browser. Το μέρος του εξυπηρετητή είναι υπεύθυνο για την αποδοχή διάφορων αιτημάτων (requests) από τον πελάτη, την εκτέλεση των διεργασιών που απαιτούνται για την επεξεργασία των αιτημάτων και την τελική απάντηση των αιτημάτων (responses) πίσω στον πελάτη.

Τέλος, στη βάση δεδομένων (MongoDB) αποθηκεύονται όλα τα δεδομένα της εφαρμογής. Τα δεδομένα μπορεί να δημιουργούνται, να ενημερώνονται ή να διαγράφονται κάθε φορά που γίνεται αποστολή κάποιου αιτήματος από το μέρος του πελάτη. Πρακτικά δημιουργήθηκαν δύο διαφορετικές εφαρμογές, μία εφαρμογή React που αφορά το μέρος του client και μια εφαρμογή Node/Express που αφορά το μέρος του εξυπηρετητή. Στη συνέχεια συνδέθηκαν οι δύο εφαρμογές καθώς και η βάση δεδομένων με τη χρήση του docker και έτσι δημιουργήθηκε μία ολοκληρωμένη εφαρμογή. Για την επικοινωνία πελάτη και εξυπηρετητή κατασκευάστηκε ένα RESTful API το οποίο θα αναλυθεί σε επόμενη ενότητα του κεφαλαίου.

5.1. Η αρχιτεκτονική της εφαρμογής στο μέρος του εξυπηρετητή

Σε αυτή την ενότητα του κεφαλαίου θα γίνει ανάλυση της αρχιτεκτονικής που αναπτύχθηκε στο μέρος του εξυπηρετητή της εφαρμογής. Όπως αναφέρθηκε στο τρίτο κεφάλαιο επιλέχθηκε η Node.js, η οποία σε συνδυασμό με το express.js framework αποτελούν τα κύρια εργαλεία για την ανάπτυξη της εφαρμογής στο μέρος του εξυπηρετητή. Σκόπιμο είναι να γίνει μία αναφορά στα βασικότερα πακέτα npm που εγκαταστάθηκαν και χρησιμοποιήθηκαν στην εφαρμογή και σχετίζονται με το περιβάλλον της Node.js.

5.1.1. Πακέτα npm

Express

Είναι το βασικότερο πακέτο με το οποίο αρχικά θα δημιουργηθεί η εφαρμογή express. Επιπλέον, δίνει τη δυνατότητα προσθήκης διάφορων middlewares χρησιμοποιώντας μεθόδους της βιβλιοθήκης. Για παράδειγμα μπορεί να οριστεί το μέγεθος πληροφορίας που λαμβάνεται μέσω του σώματος του request.

body-parse

Δίνει τη δυνατότητα να χρησιμοποιηθεί ως ένα middleware με το οποίο επεξεργάζονται τα requests προτού αναλυθούν από τους handlers. Έτσι γίνεται επεξεργασία και διαχείριση του σώματος του request πολύ πιο εύκολα σε μορφή JSON.

Mongoose

Η βιβλιοθήκη mongoose αποτελεί μια κομψή μοντελοποίηση MongoDB αντικειμένων για το περιβάλλον της Node.js. Παρέχει μια απλή λύση για την μοντελοποίηση των δεδομένων της εφαρμογής που βασίζεται σε σχήματα (schemas).

Cors

Η βιβλιοθήκη Cors αποτελεί μια βιβλιοθήκη για το περιβάλλον της Node.js και παρέχει ένα middleware για την επιλογή των CORS (Cross-Origin Resource Sharing). Είναι πολύ απλή η επιλογή αυτών αφού χρειάζεται η χρήση μόνο μιας συνάρτησης μέσα στην οποία ορίζονται πολύ συγκεκριμένα οι επιλογές που επιθυμεί ο προγραμματιστής.

Nodemon

Είναι ένα εργαλείο το οποίο βοηθάει στην ανάπτυξη εφαρμογών στο περιβάλλον της Node.js επανεκκινώντας την εφαρμογή αυτόματα, κάθε φορά που εντοπίζονται αλλαγές σε φακέλους ή αρχεία της εφαρμογής.

bcryptjs

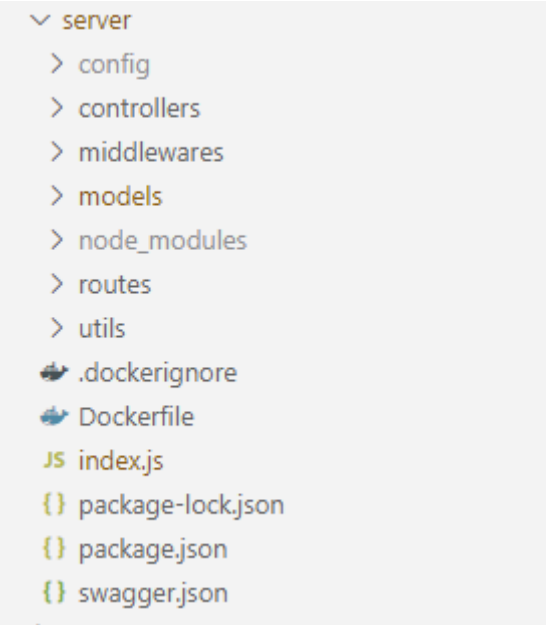
Είναι μία βιβλιοθήκη η οποία επιτρέπει την κρυπτογράφηση αλλά και την αποκρυπτογράφηση δεδομένων, όπως για παράδειγμα τον κωδικό του χρήστη. Κατά τη δημιουργία ενός νέου λογαριασμού ,ο κωδικός που καταχωρεί ο χρήστης αποθηκεύεται στη βάση δεδομένων αφού περάσει πρώτα από μία διαδικασία. Αρχικά, δημιουργείται ένα salt και στη συνέχεια κρυπτογραφείται ο κωδικός, δηλαδή γίνεται hashed. Κάθε φορά που ο χρήστης συνδέεται στην εφαρμογή συγκρίνεται ο κωδικός που δίνει και ο κωδικός που έχει αποθηκευτεί στη βάση δεδομένων. Φυσικά, ο κωδικός που είναι αποθηκευμένος πρέπει να αποκρυπτογραφηθεί. Αυτό επιτυγχάνεται με ασφάλεια χρησιμοποιώντας την κατάλληλη συνάρτηση της βιβλιοθήκης.

socket.io

Το Socket.io είναι μια βιβλιοθήκη που επιτρέπει την αμφίδρομη επικοινωνία χαμηλής καθυστέρησης, και βασισμένης σε συμβάντα (event-based) μεταξύ ενός πελάτη και ενός διακομιστή.

5.1.2. Δομή

Αρχικά, δημιουργήθηκε μια αυστηρή δομή φακέλων (folder structure) και αρχείων με την οποία μπορεί εύκολα να γίνει ο διαχωρισμός της επιχειρησιακής λογικής (business logic), των υπηρεσιών (services), των μοντέλων δεδομένων (data models) αλλά επίσης των routes και των controllers που αφορούν το API. Έτσι αποφεύχθηκε η μονολιθική και ογκώδης πρακτική που θα μπορούσε να οδηγήσει στη δημιουργία μεγάλων αρχείων τα οποία δεν θα ήταν εύκολα διαχειρίσιμα, θα ήταν δυσανάγνωστα και πολύ δύσκολα θα λύνονταν τυχόν προβλήματα. Ουσιαστικά, ακολουθήθηκε μια από τις πιο κλασικές και εύχρηστες δομές φακελοποίησης αρχείων που χρησιμοποιείται στις περισσότερες node/express εφαρμογές.



Εικόνα 14: Δομή φακέλων για server-side

Όπως φαίνεται στην εικόνα 14 το σύνολο του πηγαίου κώδικα και των υπόλοιπων εξαρτήσεων που αφορά το κομμάτι του εξυπηρετητή βρίσκεται σε έναν γονικό φάκελο με την ονομασία «server». Το αρχείο «index.js» αποτελεί το βασικότερο αρχείο και καλείται κάθε φορά που τρέχει η εφαρμογή. Θα μπορούσε να χαρακτηριστεί και ως το server αρχείο αφού ουσιαστικά αποτελεί το αρχείο δημιουργίας και εκκίνησης του server. Όσον αφορά τη δομή του αρχείου, αρχικά έγινε εισαγωγή όλων των παραπάνω πακέτων με την βοήθεια των οποίων θα κατασκευαστεί και θα τρέξει ο server. Στη συνέχεια δηλώθηκαν οι middleware συναρτήσεις και έπειτα ορίστηκαν τα routes. Τέλος, με την κατάλληλη μέθοδο της βιβλιοθήκης mongoose έγινε σύνδεση του server με την βάση δεδομένων χρησιμοποιώντας την κατάλληλη πόρτα (port).

Εκτός από το βασικό αρχείο που αναφέρθηκε ο γονικός φάκελος «server» περιέχει τους φακέλους «models», «routes», «controllers», «middlewares» και «utils». Στον πρώτο φάκελο δημιουργήθηκε ένα αρχείο για κάθε μοντέλο της βάσης δεδομένων, τα οποία θα αναλυθούν εκτενέστερα στην επόμενη ενότητα. Ο φάκελος των routes έχει ένα αρχείο για κάθε route ομαδοποίηση βάσει των requests που αφορούν μεταξύ άλλων τους χρήστες, τις τάξεις ή τα διαγράμματα-συνταγές. Ο φάκελος των controllers εμπεριέχει όλους τους controllers που χρειάζεται η εφαρμογή. Οι controllers αποτελούν μεθόδους οι οποίες παίρνουν ως παράμετρο το αίτημα του πελάτη από τα routes και το μετατρέπουν σε HTTP αποκρίσεις (responses) χρησιμοποιώντας το απαραίτητο middleware. Ο φάκελος των middlewares ουσιαστικά απομονώνει οποιοδήποτε middleware χρειάζεται για την εφαρμογή σε ένα μέρος. Στην περίπτωση της συγκεκριμένης υλοποίησης δημιουργήθηκε middleware για το authentication και το authorization του χρήστη. Τέλος, έχουμε και τον φάκελο «utils» μέσα στον οποίο βρίσκονται όλες οι βοηθητικές λειτουργικότητες που χρειάζεται η εφαρμογή, όπως για παράδειγμα η δημιουργία του JSON Web Token. Το αρχείο «swagger.json» αφορά την περιγραφή του REST API που θα αναλυθεί παρακάτω.

5.1.3. Σχεδιασμός Βάσης Δεδομένων

Για τη διαχείριση της βάσης δεδομένων χρησιμοποιήθηκε η MongoDB Atlas. Η MongoDB Atlas είναι μία cloud-based υπηρεσία βάσης δεδομένων που παρέχει ισχυρή ασφάλεια και αξιοπιστία δεδομένων. Επιπλέον παρέχει ένα δωρεάν πακέτο με έναν cluster το οποίο δεν λήγει ποτέ και σου επιτρέπει να χρησιμοποιήσεις όλες τις λειτουργικότητες της βάσης δεδομένων. Κατά τη δημιουργία του cluster παράγεται ένα string με τα οποίο δίνεται η δυνατότητα σύνδεσης στη MongoDB από τον εξυπηρετητή. Ουσιαστικά αυτό το string αποτελεί ένα uri μέσα στο οποίο υπάρχει μεταξύ άλλων το username και το password της βάσης δεδομένων, επομένως αποθηκεύτηκε στο αντίστοιχο environment αρχείο της εφαρμογής για λόγους ασφάλειας. Στην εικόνα 15 φαίνεται ο τρόπος με τον οποίο συνδέεται ο server με την βάση δεδομένων χρησιμοποιώντας τη βιβλιοθήκη mongoose. Η μεταβλητή “CONNECTION_URL” περιέχει το string που παράχθηκε από την MongoDB Atlas.

```
// DATABASE
const CONNECTION_URL = process.env.DATA_BASE_CONN;

mongoose.connect(CONNECTION_URL, {
    // avoid warnings in console
    useNewUrlParser: true,
    useUnifiedTopology: true
});

const PORT = process.env.PORT || 5000;
let server = app.listen(PORT, () => {});
```

Εικόνα 15: Σύνδεση στη βάση δεδομένων της MongoDB με την βοήθεια της express

Στη συνέχεια, δημιουργήθηκαν τα μοντέλα δεδομένων που θα χρησιμοποιηθούν για την εφαρμογή. Έτσι, στον φάκελο «models» δημιουργήθηκαν τρία διαφορετικά αρχεία, ένα για την συλλογή των χρηστών (User), ένα για την συλλογή των τάξεων (Class) και ένα για την συλλογή των συνταγών (Recipe). Μέσα σε κάθε αρχείο αρχικά δημιουργείται ένας ορισμός του εγγράφου (Schema αντικείμενο της mongoose) το οποίο αντιστοιχεί σε μια συλλογή της MongoDB. Κάθε έγγραφο αποτελείται από διάφορα κλειδιά (keys) που ορίζονται με μια ιδιότητα (property). Εάν μια ιδιότητα απαιτεί μόνο έναν τύπο, μπορεί να καθοριστεί χρησιμοποιώντας μόνο τον τύπο. Εκτός από αυτόν , μια ιδιότητα μπορεί να περιέχει και άλλες παραμέτρους όπως για παράδειγμα την αρχική τιμή ή την υποχρεωτικότητα του συγκεκριμένου κλειδιού. Στα κλειδιά μπορεί επίσης να εκχωρηθούν ένθετα αντικείμενα (nested objects) που περιέχουν περαιτέρω ορισμούς κλειδιού/τύπου ή και αναφορά σε άλλο έγγραφο. Εκτός από τον ορισμό του εγγράφου τα αρχεία μπορεί να περιέχουν και διάφορες μεθόδους που σχετίζονται με το αντίστοιχο έγγραφο και εκτελούνται πριν ή μετά τη δημιουργία ή την ενημέρωση αυτού. Παρακάτω απεικονίζονται τα έγγραφα που δημιουργήθηκαν.

User model:

```
name: {
    type: String,
    required: true
},
email: {
    type: String,
    required: true,
    unique: true
},
password: {
    type: String,
    required: true
},
isAdmin: {
    type: Boolean,
    required: true,
    default: false
},
isTeacher: {
    type: Boolean,
    required: true,
    default: false
},
isStudent: {
    type: Boolean,
    required: true,
    default: true
},
recipes: [{ type: ObjectId, ref: 'Recipe' }],
```

Εικόνα 16: Περιγραφή του User model

Class Model:

```
name: {
    type: String,
    required: true
},
creator: {
    type: ObjectId,
    ref: 'User',
    required: true
},
teacher: {
    type: ObjectId,
    ref: 'User',
    required: true
},
students: [{type: ObjectId, ref: 'User'}],
recipes: [{ type: ObjectId, ref: 'Recipe' }]
```

Εικόνα 17: Περιγραφή του Class model

Recipe Model:

```
elements: {
    type: String,
    required: true
},
name: {
    type: String,
    required: true
},
user: {
    type: ObjectId,
    ref: 'User'
},
class: {
    type: ObjectId,
    ref: 'Class'
},
realtime: {
    type: Boolean,
    default: false
},
```

Εικόνα 18: Περιγραφή Recipe model

5.1.4. Authentication και Authorization

Authentication

Ένα σημαντικό κομμάτι της υλοποίησης ήταν η ανάπτυξη ενός συστήματος πιστοποίησης των χρηστών βάσει της ηλεκτρονικής τους διεύθυνσης (email), τον προσωπικό κωδικό τους (password) αλλά και την χρήση JWT. Στη συνέχεια θα γίνει ανάλυση του εν λόγω μηχανισμού. Για την κατασκευή του μηχανισμού , αρχικά έγινε η εγκατάσταση της βιβλιοθήκης Jsonwebtoken. Η βιβλιοθήκη επιτρέπει την εισαγωγή κάποιων μεθόδων για να δημιουργούνται πολύ εύκολα JSON Web Tokens.

Όταν ο χρήστης συνδέεται στην εφαρμογή, δημιουργείται ένα JSON Web Token βάσει του id της εγγραφής του (_id), το οποίο έχει διάρκεια ζωής μια μέρα. Στη συνέχεια, αυτό το token αποστέλλεται στον πελάτη (client) κατά την απόκριση (response). Έτσι, το token αποθηκεύεται τοπικά στο client-side της εφαρμογής και χρησιμοποιείται ως header για κάθε νέο αίτημα που επιχειρεί ο πελάτης, εφόσον αυτό το αίτημα χρειάζεται πιστοποίηση. Στη συνέχεια, για κάθε νέο αίτημα καλείται ένα middleware, το οποίο είναι υπεύθυνο για τον έλεγχο της ύπαρξης token. Πιο συγκεκριμένα ελέγχει αν έχει δηλωθεί στο αίτημα το «Bearer Token». Τέλος, εφόσον δεν προκύψει πρόβλημα από τον πρώτο έλεγχο καλείται η μέθοδος verify της βιβλιοθήκης Jsonwebtoken με την οποία γίνεται η επαλήθευση του token. Η επαλήθευση γίνεται ασύγχρονα χρησιμοποιώντας ένα κρυφό κλειδί που έχει οριστεί στο environment αρχείο (“.env”).

Authorization

Επιπλέον δημιουργήθηκε και μία λογική middleware έτσι ώστε να ελέγχετε το δικαίωμα του χρήστη για το κάθε αίτημα που στέλνει. Όλα τα αιτήματα περιέχουν την πληροφορία για τον ρόλο του χρήστη, δηλαδή αν πρόκειται για χρήστη με ρόλο student, teacher ή admin. Έτσι η συγκεκριμένη συνάρτηση κάνει αυτό τον έλεγχο και αποφασίζει αν ο εκάστοτε χρήστης επιτρέπετε να πάρει την πληροφορία που ζητάει κατά την απόκριση (response).

5.2. Σχεδιασμός και ανάπτυξη REST API

Η αρχιτεκτονική REST (Representational State Transfer) χρησιμοποιείται για την δημιουργία και την επικοινωνία υπηρεσιών διαδικτύου (web services) βάσει του πρωτοκόλλου HTTP και παρουσιάστηκε πρώτη φορά από τον Roy Fielding το 2000. Συνήθως επιβάλλει στους πόρους του ιστού (web resources) να ορίζονται σε μορφή κειμένου (JSON, HTML ή XML) οι οποίοι μπορούν να προσπελαστούν ή να τροποποιηθούν από ένα προκαθορισμένο σύνολο λειτουργιών (operations) οι οποίες αντιστοιχούν σε HTTP μεθόδους όπως GET, POST, PUT ή DELETE. Για παράδειγμα, σε μία συλλογή δεδομένων όπως οι χρήστες μιας εφαρμογής, συχνά εκτελούνται ορισμένες ενέργειες για την δημιουργία (create), ανάγνωση (read), ενημέρωση (update) και διαγραφή (delete), γνωστή και ως λειτουργικότητα CRUD, δηλαδή ένα σύνολο βασικών λειτουργιών που είναι ενσωματωμένες στο πρωτόκολλο HTTP. Με την χρήση των HTTP λειτουργιών και μια ονομασία πόρου ως μια διεύθυνση, μπορεί να χτιστεί ένα REST API δημιουργώντας ένα endpoint για κάθε λειτουργία. Εφαρμόζοντας αυτό το μοτίβο, δημιουργείται μια σταθερή και εύκολη βάση που δίνει τη δυνατότητα της γρήγορης εξέλιξης του κώδικα καθώς και τη διατήρηση αυτού στη συνέχεια. Οι υπηρεσίες διαδικτύου που βασίζονται στην REST αρχιτεκτονική είναι γνωστές ως RESTful υπηρεσίες διαδικτύου. Μία RESTful υπηρεσία συνήθως ορίζει ένα URI (Uniform Resource Identifier), μια ακολουθία χαρακτήρων που παρέχει την αναπαράσταση των πόρων (για παράδειγμα ένα JSON) και ένα σύνολο HTTP μεθόδων.

Η διεπαφή προγραμματισμού εφαρμογών γνωστή και ως API (Application Programming Interface), όπως υποδηλώνει η ονομασία, αποτελεί μία διεπαφή που καθορίζει την αλληλεπίδραση μεταξύ διαφορετικών στοιχείων λογισμικού (software components). Τα web APIs καθορίζουν ποια αιτήματα (requests) μπορούν να υποβληθούν σε ένα στοιχείο (για παράδειγμα η λήψη της λίστας των χρηστών της εφαρμογής), τον τρόπο υποβολής τους (για παράδειγμα ένα αίτημα GET) και τις αναμενόμενες απαντήσεις τους (responses).

Swagger

Το swagger είναι μια γλώσσα περιγραφής διεπαφής που επιτρέπει την περιγραφή της δομής ενός API έτσι ώστε να μπορεί να διαβαστεί από μια μηχανή. Αποτέλεσμα αυτού είναι η αυτόματη δημιουργία ενός διαδραστικού και εύχρηστου εγχειριδίου που αφορά το API. Επιπλέον υπάρχει η δυνατότητα δημιουργίας διαφόρων βιβλιοθηκών - σε πολλές γλώσσες προγραμματισμού - που αποσκοπούν μεταξύ άλλων σε αυτοματοποιημένους ελέγχους του API. Το μόνο που χρειάζεται από την πλευρά του προγραμματιστή είναι η συγγραφή ενός YAML ή JSON αρχείου στο οποίο περιγράφεται αναλυτικά ολόκληρο το API. Πρόκειται ουσιαστικά για ένα αρχείο με μία λίστα πόρων για το API βάσει των προδιαγραφών του OpenAPI.

OpenAPI

Οι προδιαγραφές OpenAPI (OpenAPI specification) αποτελούν μια μορφή του τρόπου με τον οποίο περιγράφεται ένα REST API. Ένα αρχείο βασισμένο στις προδιαγραφές OpenAPI μπορεί να περιέχει τα διαθέσιμα endpoints (/recipes) και τις αντίστοιχες λειτουργίες (GET /recipes, POST /recipes) καθώς και τις παραμέτρους αυτών. Επιπλέον μπορεί να περιέχει μεθόδους για επιβεβαίωση της αυθεντικότητας (authentication) αλλά και πολλές άλλες πληροφορίες όπως για παράδειγμα την άδεια, τους όρους χρήσης ή την επικοινωνία.

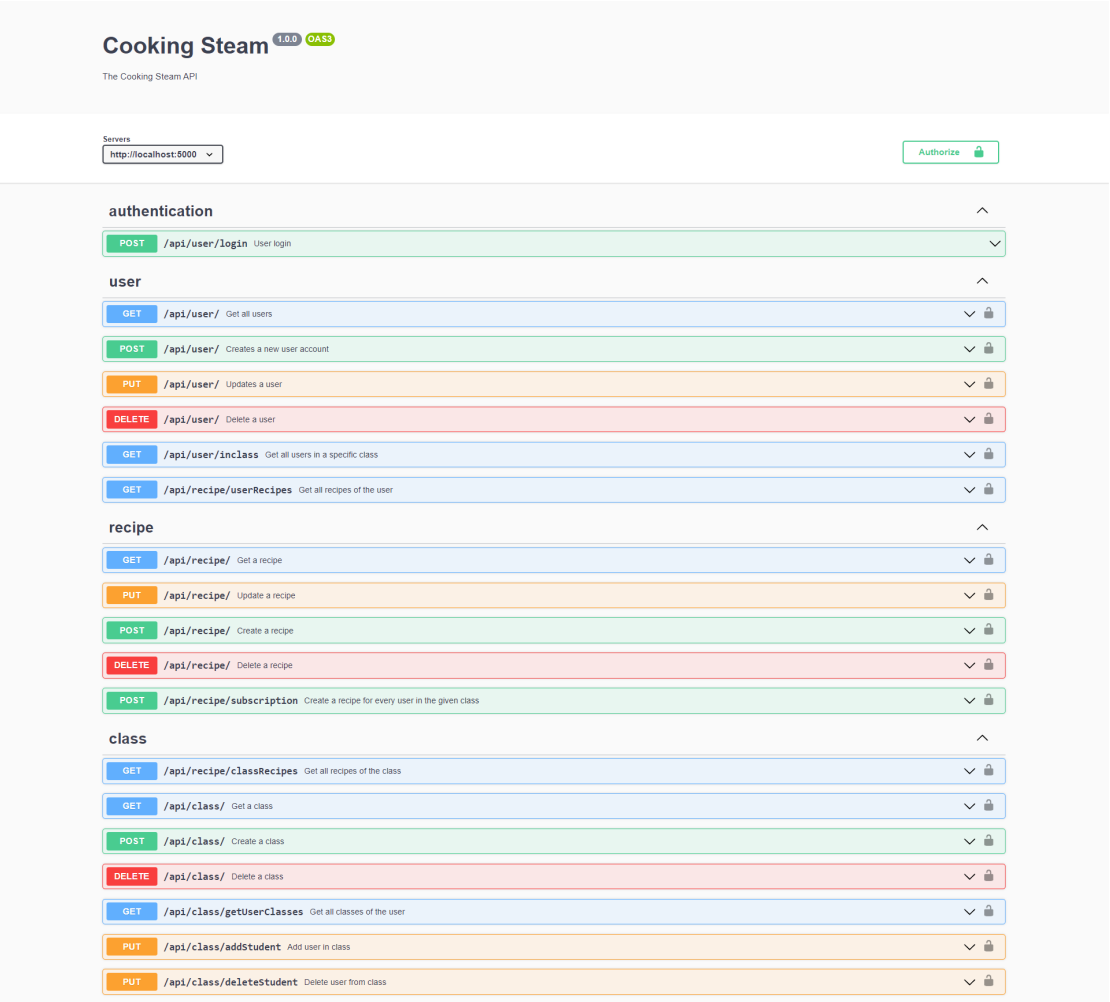
Για την υλοποίηση του εγχειριδίου του Cooking STEAM API χρησιμοποιήθηκαν τα πακέτα swagger-UI-express και swagger-jsdoc. Τα πακέτα αυτά επιτρέπουν την αυτοματοποιημένη δημιουργία API εγχειριδίων σε swagger-ui για το περιβάλλον της express και βασίζεται στο αρχείο “swagger.json” που βρίσκεται στο ριζικό φάκελο. Η έκδοση που χρησιμοποιήθηκε ήταν η OAS 3.0.

Authentication

Για το authentication χρησιμοποιήθηκε η τεχνική Bearer Authentication (token authentication) η οποία:

- Αποτελεί ένα σχήμα ελέγχου ταυτότητας HTTP που περιλαμβάνει token ασφαλείας που ονομάζονται bearer tokens.
- Το bearer token είναι μια κρυπτογραφημένη συμβολοσειρά, που συνήθως δημιουργείται από τον διακομιστή ως απόκριση σε ένα αίτημα σύνδεσης. Η συγκεκριμένη λογική υλοποιήθηκε με την βιβλιοθήκη jsonwebtoken.
- Ο πελάτης πρέπει να στείλει αυτό το token στην κεφαλίδα εξουσιοδότησης (header authentication) όταν υποβάλλει αιτήματα σε προστατευμένους πόρους.
- Το Bearer Authentication δημιουργήθηκε αρχικά ως μέρος του OAuth 2.0
- Ομοίως με τον Βασικό έλεγχο ταυτότητας, ο έλεγχος ταυτότητας φορέα θα πρέπει να χρησιμοποιείται μόνο μέσω HTTPS (SSL).

Παρακάτω φαίνονται όλα τα αιτήματα με τη γραφική αναπαράσταση που δημιουργείται με τη χρήση του swagger:



Εικόνα 19: Απεικόνιση του REST API με τη βοήθεια του swagger

5.3. Η αρχιτεκτονική της εφαρμογής στο μέρος του πελάτη

Όπως αναφέρθηκε στο προηγούμενο κεφάλαιο, για την υλοποίηση του front-end της εφαρμογής θα γίνει χρήση της React.js βιβλιοθήκης. Σε αυτήν την ενότητα του κεφαλαίου, αρχικά θα γίνει αναφορά στα βασικότερα πακέτα npm που εγκαταστάθηκαν και χρησιμοποιήθηκαν στο front-end κομμάτι της εφαρμογής και σχετίζονται με την React.js. Στη συνέχεια, κατά την ανάλυση του σχεδιασμού και της ανάπτυξης του front-end, τα εν λόγω πακέτα θα αναλυθούν σε μεγαλύτερο βαθμό και θα περιγραφεί η χρήση τους.

5.3.1. Πακέτα npm

React Flow

Η React Flow είναι μια δωρεάν βιβλιοθήκη για τη δημιουργία γραφημάτων που βασίζονται σε κόμβους (nodes). Δημιουργήθηκε από την εταιρεία Webkid για την εφαρμογή “datablocks”, μια εφαρμογή στην οποία ο χρήστης χρησιμοποιεί έναν editor για την αναζήτηση, ανάλυση και μετατροπή δεδομένων χωρίς τη χρήση κώδικα. Η βιβλιοθήκη παρέχει μια πληθώρα

επιλογών με τις οποίες μπορεί να κατασκευάσει κανείς από απλά στατικά διαγράμματα μέχρι πολύπλοκα, δυναμικά και διαδραστικά editors. Ωστόσο για την υλοποίηση του editor της εφαρμογής Cooking STEAM, χρειάστηκε να αναπτυχθούν και να τροποποιηθούν πολλές από τις αρχικοποιημένες επιλογές και λειτουργικότητες που παρέχει η βιβλιοθήκη, προκειμένου να υπάρξει το επιθυμητό αποτέλεσμα.

MUI / React-bootstrap

Τόσο η MUI όσο και η React–bootstrap αποτελούν δωρεάν, απλές και προσαρμόσιμες component βιβλιοθήκες για τη δημιουργία πιο γρήγορων και πιο προσιτών εφαρμογών React. Ουσιαστικά εγκαθιστώντας τις βιβλιοθήκες στην εφαρμογή, εισάγεται μία ποικιλόμορφη παλέτα επιλογών που σχετίζονται με την κλήση διάφορων έτοιμων components. Κατ' επέκταση μπορεί να γίνει χρήση των εκάστοτε components και να τροποποιηθούν ανάλογα με τις ανάγκες και τις απαιτήσεις της εφαρμογής. Ένα από τα μεγαλύτερα πλεονεκτήματα που έχουν τα δύο πακέτα είναι η εύκολη προσαρμογή των components της εφαρμογής, τόσο στην οθόνη του υπολογιστή όσο και σε οθόνες mobile. Αυτό επιτυγχάνεται με τη χρήση των κατάλληλων επιλογών μέσα στο tag του component που προέρχονται από ένα σύνολο μεθόδων και συναρτήσεων CSS.

Redux / React-redux / Redux-thunk

Η Redux αποτελεί μια βιβλιοθήκη διαχείρισης της κατάστασης (state management). Ουσιαστικά με τη χρήση της βιβλιοθήκης δίνεται η δυνατότητα αποθήκευσης κάποιων δεδομένων κεντρικά, σε μια οντότητα που ονομάζεται store. Στη συνέχεια μπορούν να χρησιμοποιηθούν τα δεδομένα του store από οποιοδήποτε component μέσα στην εφαρμογή. Στη συνέχεια του κεφαλαίου, στην ενότητα "Διαχείριση κατάστασης της εφαρμογής" θα γίνει εκτενής αναφορά στην αρχιτεκτονική Redux αλλά και στις βιβλιοθήκες react-redux και redux-thunk.

Axios

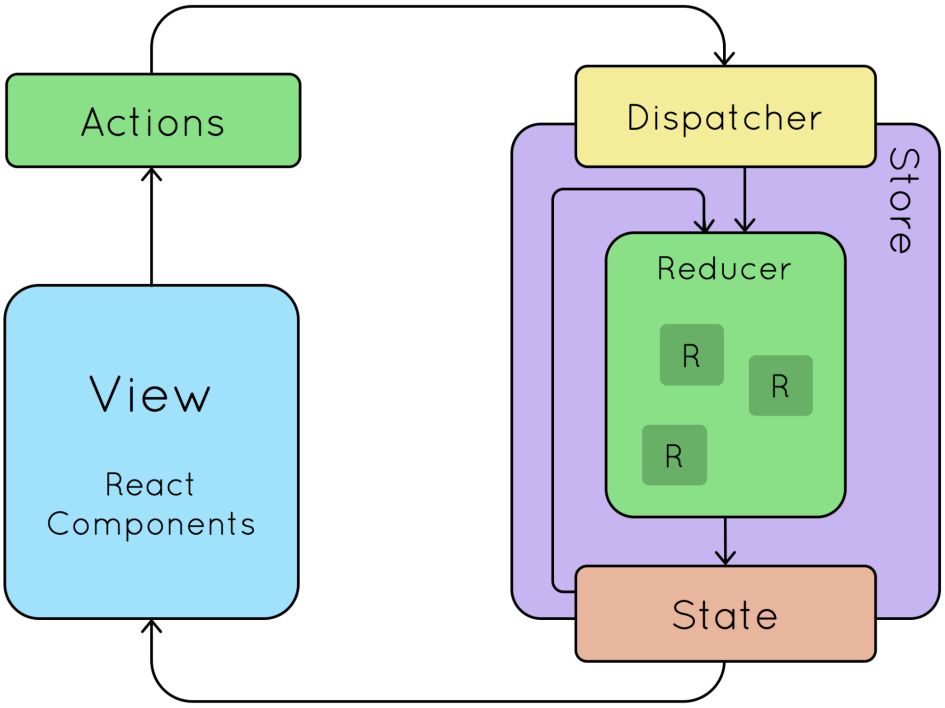
Το axios είναι ένα framework το οποίο χρησιμοποιείται για την κλήση μεθόδων HTTP αιτημάτων (GET, POST, PUT, DELETE). Το Axios βασίζεται στις "υποσχέσεις" (promises) τόσο για την node.js όσο και το πρόγραμμα περιήγησης. Μπορεί να τρέξει στο πρόγραμμα περιήγησης και στην node.js με τον ίδιο κώδικα. Από την πλευρά του εξυπηρετητή χρησιμοποιεί την εγγενή node.js http μονάδα, ενώ στον πελάτη χρησιμοποιεί XMLHttpRequests. Στη συγκεκριμένη υλοποίηση χρησιμοποιήθηκε από την πλευρά του πελάτη για την επικοινωνία με το Rest API. Παρακάτω φαίνεται ένα παράδειγμα χρήσης του framework κατά το οποίο επιτυγχάνεται επικοινωνία με το endpoint που αφορά τη διαγραφή μαθητή από μια τάξη.

```
/**
 * delete student from class
 */
Complexity is 3 Everything is cool!
const delete_student = useCallback((student_id) => {
 try {
   const config = {
     headers: {
       "Content-Type": "application/json",
       Authorization: `Bearer ${userInfo.token}`,
     },
   };

   let user_id = userInfo._id;

   axios.put(process.env.REACT_APP_BASE_URL + "/class/deleteStudent/", {user_id, class_id, student_id}, config)
   window.location.reload(false);
 } catch (error) {

 }
},[class_id, userInfo.token, userInfo._id]);
```

Εικόνα 20: Παράδειγμα χρήσης του axios framework για αίτημα πελάτη

5.3.2. Διαχείριση κατάστασης της εφαρμογής

Η Redux όπως αναφέρθηκε είναι μια JS βιβλιοθήκη με την οποία επιτυγχάνεται η διαχείριση και η ενημέρωση τη καθολικής κατάστασης (global state) της εφαρμογής χρησιμοποιώντας γεγονότα (events) τα οποία ονομάζονται actions. Η βασική ιδέα της Redux είναι η δυνατότητα ύπαρξης μιας κεντρικής οντότητας που ονομάζεται redux store η οποία περιέχει την κατάσταση στην οποία βρίσκεται η εφαρμογή. Μέσα από μια αλληλουχία διαφόρων ενεργειών υπάρχει προβλέψιμος κώδικας κατά την ενημέρωση της κατάστασης. Στη συνέχεια θα αναλυθεί η αλληλουχία των ενεργειών και από ποιες οντότητες απαρτίζεται.



Εικόνα 21: Η αρχιτεκτονική Redux

Actions

Ένα action ουσιαστικά είναι ένα αντικείμενο JS το οποίο συνήθως περιέχει έναν τύπο. Ουσιαστικά πρόκειται για ένα event το οποίο περιγράφει μια αλλαγή που έγινε στην εφαρμογή. Ένα αντικείμενο τύπου action εκτός από τον τύπο μπορεί να περιέχει και άλλα πεδία με επιπρόσθετη πληροφορία για τη συγκεκριμένη αλλαγή. Κατά κανόνα η πληροφορία αυτή αποθηκεύεται σε ένα πεδίο με την ονομασία “payload”.

Reducers

Ένας reducer πρόκειται για μια συνάρτηση η οποία παίρνει ως είσοδο την τωρινή κατάσταση (current state) και ένα αντικείμενο τύπου action. Η μέθοδος αποφασίζει πώς να ενημερωθεί η κατάσταση, εφόσον είναι απαραίτητο, και επιστρέφει τη νέα κατάσταση. Ένας reducer μπορεί να χαρακτηριστεί ως ένας event listener ο οποίος διαχειρίζεται events βάσει του action που δέχεται ως είσοδο. Τέλος, μπορεί να χρησιμοποιείται κάθε είδους λογική για να αποφασιστεί ποια θα πρέπει να είναι η νέα κατάσταση (if statement, switch statement).

Store

Η τωρινή κατάσταση της εφαρμογής αποθηκεύεται σε ένα αντικείμενο που ονομάζεται store. Το store δημιουργείται περνώντας ως όρισμα κατά την αρχικοποίησή του έναν reducer ένα πιο σύνθετο αντικείμενο που μπορεί να περιέχει περισσότερους από έναν reducers.

Dispatch

Το Redux store διαθέτει μια μέθοδο που ονομάζεται dispatch η οποία αποτελεί το μοναδικό τρόπο με τον οποίο μπορεί να ενημερωθεί η κατάσταση. Η μέθοδος παίρνει ως όρισμα ένα αντικείμενο τύπου action. Το store θα τρέξει την εκάστοτε συνάρτηση τύπου reducer και θα αποθηκεύσει τη νέα κατάσταση. Ουσιαστικά με τη μέθοδο dispatch πυροδοτείται ένα event στην εφαρμογή, δηλαδή το γεγονός ότι υπάρχει αλλαγή στην εφαρμογή και πρέπει να ενημερωθεί το store. Έτσι οι reducers δρουν ως event listeners και κάθε φορά που “ακούνε” ένα action τότε ενημερώνουν την κατάσταση.

Μία εφαρμογή που χρησιμοποιεί την αρχιτεκτονική Redux λειτουργεί με τη μέθοδο “one-way data flow” η οποία περιγράφει την αλληλουχία των βημάτων που γίνονται για να ενημερωθεί η εφαρμογή. Πιο συγκεκριμένα, το state περιγράφει την κατάσταση της εφαρμογής στη συγκεκριμένη χρονική στιγμή. Το UI διαμορφώνεται σε σχέση με αυτή την κατάσταση. Όταν συμβαίνει κάτι, όπως για παράδειγμα το πάτημα ενός κουμπιού, το state ενημερώνεται βάσει της εκάστοτε ενέργειας και το UI αναδιαμορφώνεται βάσει της νέας κατάστασης. Έτσι η αρχιτεκτονική Redux μπορεί να διασπαστεί στα επιμέρους βήματα:

Αρχικοποίηση

Ένα Redux store δημιουργείται χρησιμοποιώντας μια συνάρτηση τύπου Reducer. Το store καλεί μια φορά τη συγκεκριμένη κατάσταση και αποθηκεύει την τιμή που επιστρέφεται ως μια αρχική κατάσταση. Όταν το UI διαμορφώνεται για πρώτη φορά, τα UI components έχουν πρόσβαση στην υπάρχουσα κατάσταση του Redux store και χρησιμοποιούν τα δεδομένα της κατάστασης για να επιτευχθεί η αντίστοιχη αναδιαμόρφωση (rendering).

Ενημέρωση

Όταν πυροδοτείται μία αλλαγή αποστέλλεται ένα action στο Redux store μέσω της μεθόδου dispatch. Το store τρέχει την εκάστοτε συνάρτηση τύπου reducer έχοντας ως ορίσματα την προηγούμενη κατάσταση και το αντικείμενο τύπου action και τελικά αποθηκεύει την τιμή που επιστρέφεται ως τη νέα κατάσταση. Στη συνέχεια το store ενημερώνει όλα τα μέρη του UI με τη νέα κατάσταση και τα δεδομένα αυτής. Έτσι τα UI components ελέγχουν αν υπάρχουν αλλαγές σε σχέση με την πληροφορία που περιέχει η προηγούμενη κατάσταση και η τωρινή. Εάν υπάρχουν αλλαγές τότε γίνεται αναδιαμόρφωση με τη νέα πληροφορία και έτσι γίνονται αλλαγές οι οποίες τελικά φαίνονται και στην οθόνη.

Σκόπιμο είναι να γίνει αναφορά και στο middleware που χρησιμοποιείται κατά την αρχικοποίηση. Ως τώρα η αρχιτεκτονική που περιγράφηκε υλοποιείται σύγχρονα. Παρόλα αυτά χρησιμοποιώντας την JavaScript χρειάζεται να γίνει χρήση και ασύγχρονης λογικής, όπως για παράδειγμα όταν πρέπει να γίνει η επικοινωνία και η παραλαβή δεδομένων από ένα API. Σε αυτή την περίπτωση πρέπει να χρησιμοποιηθεί ασύγχρονη λογική και στην αρχιτεκτονική Redux. Έτσι, κατά την αρχικοποίηση του store χρησιμοποιείται μια συνάρτηση που ονομάζεται “thunk” η οποία περιέχει ασύγχρονη λογική. Για να επιτευχθεί αυτό ,είναι απαραίτητη η εισαγωγή του redux-thunk middleware και η αντίστοιχη βιβλιοθήκη.

Με τη βιβλιοθήκη React-Redux δίνεται η δυνατότητα χρήσης ορισμένων React hooks τα οποία επιτρέπουν στα React components να αλληλεπιδρούν με το Redux store. Στην περίπτωση της συγκεκριμένης υλοποίησης έγινε χρήση των hooks “UseSelector” και “UseDispatch”. Το πρώτο επιτρέπει στα components να εξάγουν την πληροφορία που χρειάζονται από την κατάσταση του Redux store, ενώ το δεύτερο ουσιαστικά δίνει τη δυνατότητα του dispatching στα actions. Τα παραπάνω Hooks αποτελούν τον δίαυλο επικοινωνίας μεταξύ του redux store και των React components. Αφού έγινε η ανάλυση της αρχιτεκτονικής Redux και πώς αυτή μπορεί να χρησιμοποιηθεί σε μια React εφαρμογή, σκόπιμο είναι να γίνει αναφορά του αρχικού σημείου της εφαρμογής στο οποίο όλα τα παραπαάνω κομμάτια και ενέργειες πυροδοτούνται και λειτουργούν αρμονικά. Στο αρχείο “index.js” ουσιασιαστικά αρχίζει το rendering της React εφαρμογής καλώντας το ReactDom.render. Για να γίνει η χρήση των hooks από την React-Redux πρέπει να εισαχθεί στο provider το store που δημιουργήθηκε. Έτσι κάθε React component που καλεί τα Hooks θα μπορεί να επικοινωνεί με το Redux store που δόθηκε στο provider.

Τέλος, είναι σημαντικό να μην γίνεται κατάχρηση της αρχιτεκτονικής Redux και να μην χρησιμοποιείται μόνο όταν η εφαρμογή την έχει πραγματικά ανάγκη. Κατά την υλοποίηση της εφαρμογής χρησιμοποιήθηκε κατά κύριο λόγο για την ενημέρωση της κατάστασης της εφαρμογής και πιο συγκεκριμένα όταν ο χρήστης εκτελεί τις ενέργειες Login, Register ή Logout. Λόγω αυτού χρειάστηκε η δημιουργία δύο συναρτήσεων reducer. Η πρώτη καλείται όταν ο χρήστης κάνει login και η δεύτερη όταν ο χρήστης κάνει νέο λογαριασμό (register).

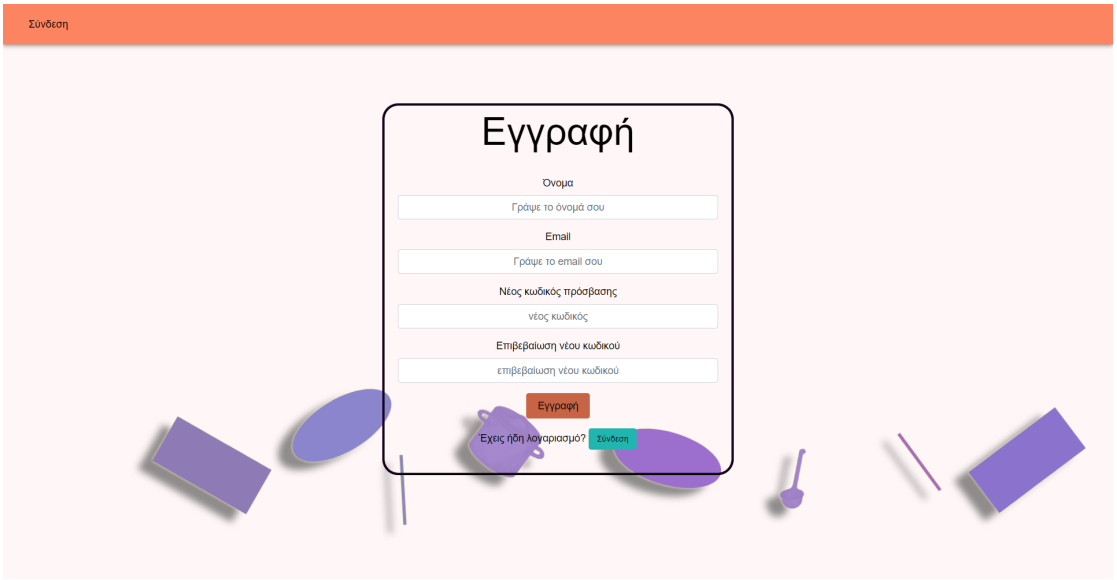
6. Τελική υλοποίηση και User Interfaces

Στα προηγούμενα κεφάλαια παρουσιάστηκε η αρχιτεκτονική που αναπτύχθηκε για κάθε κομμάτι της εφαρμογής και τα εργαλεία με τα οποία δημιουργήθηκαν. Στα επόμενα κεφάλαια θα παρουσιαστεί η τελική έκδοση της εφαρμογής, τα αποτελέσματα της εργασίας αλλά και το τι επιτεύχθηκε.

6.1. Λειτουργίες εφαρμογής

6.1.1. Λειτουργίες εφαρμογής για τον ρόλο “student”

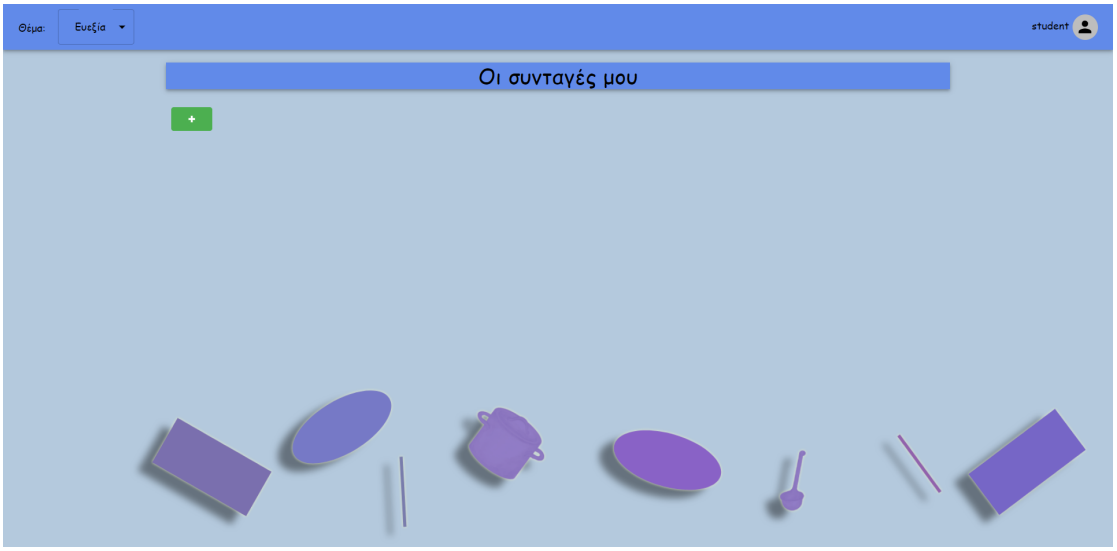
Με τη δημιουργία της διεπαφής του χρήστη τα οφέλη είναι πολλά. Σε έναν απλό χρήστη της εφαρμογής ουσιαστικά δίνονται δικαιώματα μαθητή, δηλαδή οι λειτουργίες που μπορεί να εκτελέσει είναι συγκεκριμένες και αφορούν δραστηριότητες ενός μαθητή. Η διεπαφή σχεδιάστηκε και αναπτύχθηκε με τέτοιο τρόπο ώστε να είναι προσιτή, κατανοητή και σχετικά απλή, εφόσον οι χρήστες είναι κυρίως μαθητές ηλικίας 10-12 ετών. Το μοτίβο αυτό χρησιμοποιήθηκε από τη σελίδα που υποδέχεται τον χρήστη και την φόρμα εισαγωγής στοιχείων κατά τη δημιουργία νέου λογαριασμού μέχρι και το βασικό εργαλείο της εφαρμογής, δηλαδή τον editor των διαγραμμάτων ροής.



Εικόνα 22: Σελίδα για εγγραφή νέου χρήστη

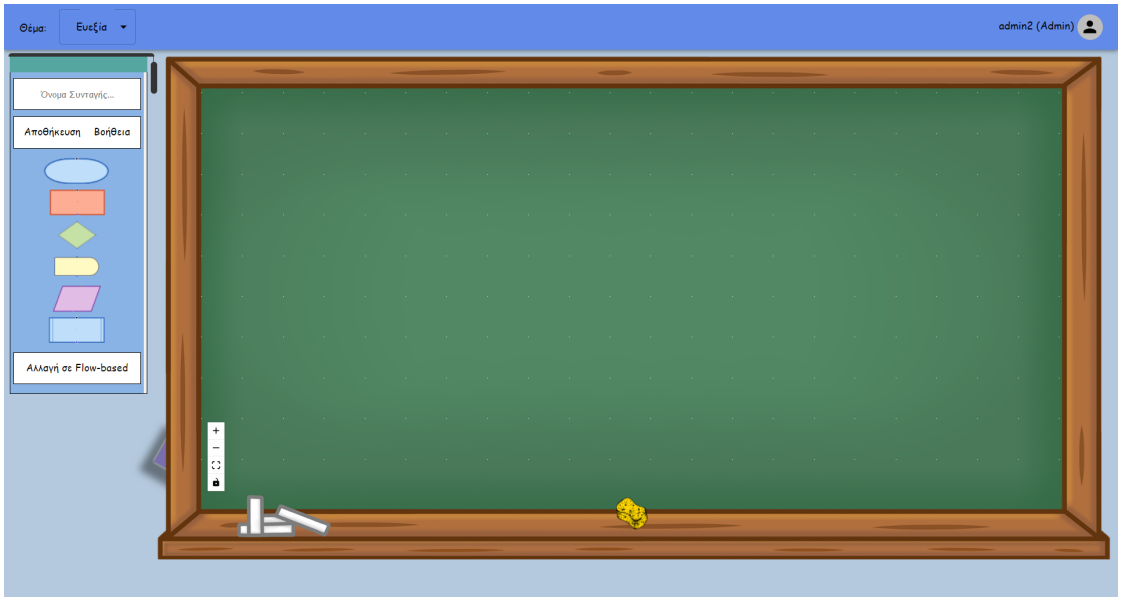
Απαραίτητη προϋπόθεση για να χρησιμοποιήσει ένας χρήστης τις λειτουργίες της εφαρμογής, είναι η δημιουργία προσωπικού λογαριασμού. Όπως φαίνεται και στην εικόνα 22, ο κάθε μαθητής αφού επισκεφτεί τον ιστότοπο έχει μόνο δύο επιλογές, είτε να δημιουργήσει νέο λογαριασμό, εφόσον είναι νέος χρήστης, είτε να συνδεθεί στον ήδη υπάρχοντα λογαριασμό του. Έτσι, επιλέγοντας το πορτοκαλί κουμπί μπορεί να δημιουργήσει νέο λογαριασμό ή επιλέγοντας το μπλε κουμπί έχει την δυνατότητα να συνδεθεί στον ήδη υπάρχοντα λογαριασμό του. Κατά το πάτημα του πορτοκαλί κουμπιού ο χρήστης θα μεταφερθεί στην αντίστοιχη σελίδα η οποία περιλαμβάνει τη φόρμα εγγραφής νέου χρήστη. Στη συνέχεια ο χρήστης θα πρέπει να συμπληρώσει τα στοιχεία της φόρμας και να πατήσει το κουμπί «εγγραφή». Εφόσον τα στοιχεία που δόθηκαν είναι σωστά και δεν

έχει προκύψει κάποιο πρόβλημα κατά τον έλεγχο και την αποθήκευση των στοιχείων στη βάση δεδομένων, τότε ο χρήστης θα μπορέσει να συνδεθεί στην εφαρμογή. Αντίστοιχα, πατώντας το μπλε κουμπί, ο χρήστης καταχωρεί τα στοιχεία του, επιλέγει το κουμπί «σύνδεση» και εφόσον τα στοιχεία που έδωσε επαληθευτούν, εισέρχεται στην εφαρμογή.



Εικόνα 23: Σελίδα profile ενός χρήστη τύπου student

Και στις δύο περιπτώσεις, αφού ο χρήστης εισέλθει στην εφαρμογή, θα μεταφερθεί στη βασική σελίδα στην οποία, όπως φαίνεται και στην εικόνα 23, εμφανίζεται η ενότητα «Οι συνταγές μου». Σε αυτή την ενότητα ο χρήστης έχει τη δυνατότητα να δημιουργήσει νέα συνταγή-διάγραμμα ροής επιλέγοντας το κουμπί «+». Εναλλακτικά, μπορεί να επιλέξει το κουμπί «Σχεδίασε» από την μπάρα πλοήγησης. Κατά τη δημιουργία μιας νέας συνταγής ο χρήστης μεταφέρεται στη βασικότερη σελίδα της εφαρμογής, τον editor των διαγραμμάτων ροής. Σε αυτή τη σελίδα ο χρήστης έχει διάφορες επιλογές μέσα από τις οποίες μπορεί να φτάσει στην επιτυχή δημιουργία ενός διαγράμματος ροής. Στο αριστερό μέρος της σελίδας, όπως φαίνεται και στην εικόνα 19, ο χρήστης έχει μία πληθώρα επιλογών και εργαλείων, ενώ στο δεξί μέρος βρίσκεται η επιφάνεια εργασίας, εκεί που σχεδιάζεται το διάγραμμα ροής του χρήστη.



Εικόνα 24: Σελίδα editor για κλασσικό διάγραμμα ροής

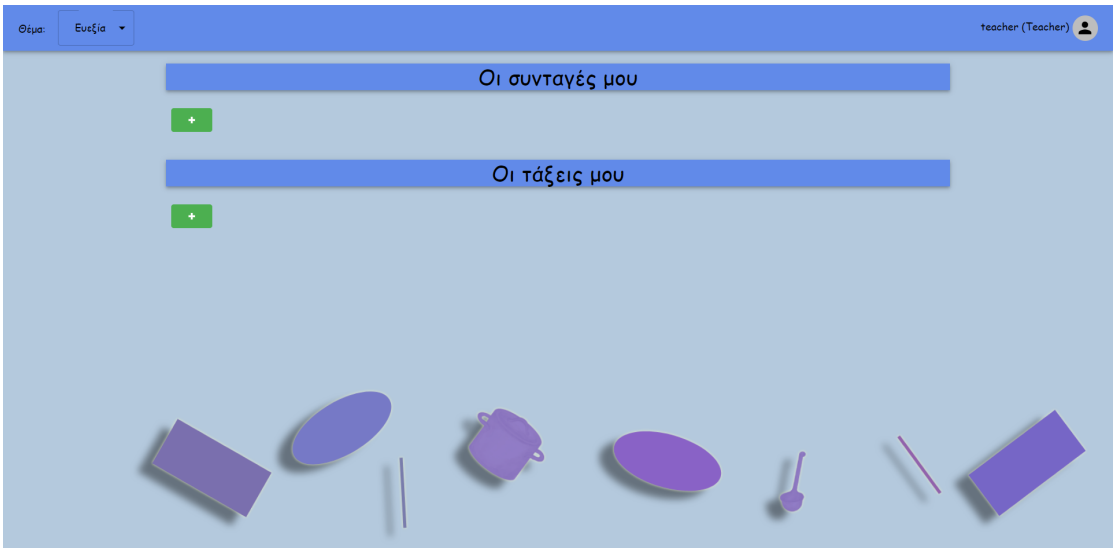
Στο πάνω μέρος της ενότητας των εργαλείων δίνεται η δυνατότητα στο χρήστη να επιλέξει την ονομασία του νέου διαγράμματος. Στη συνέχεια υπάρχει το κουμπί «αποθήκευση» με το οποίο ο χρήστης αποθηκεύει το νέο διάγραμμα ή ενημερώνει το ήδη υπάρχον. Ακόμη υπάρχει και το κουμπί «Βοήθεια» κατά την επιλογή του οποίου εμφανίζεται ένα modal στο οποίο διατυπώνονται και παρέχονται οι οδηγίες και τα βήματα που πρέπει να ακολουθήσει ο χρήστης έτσι ώστε να δημιουργήσει επιτυχώς ένα διάγραμμα ροής.

Το βασικότερο κομμάτι της ενότητας των εργαλείων αποτελεί η επιλογή των σχημάτων. Ο χρήστης έχει τη δυνατότητα να επιλέξει ανάμεσα σε 6 βασικά σύμβολα διαγραμμάτων ροής, τα οποία είναι τα σύμβολα Αρχής/Τέλους, Ενέργειας, Λήψης Απόφασης, Αναμονής, Εισόδου Πληροφορίας και Εκτέλεσης Διαδικασίας. Στη συνέχεια, αφού ο χρήστης εισάγει τα σχήματα στην επιφάνεια εργασίας σέρνοντάς τα, μπορεί να τα ενώνει μεταξύ τους επιλέγοντας τις χαρακτηριστικές κουκίδες χρώματος μπλε (είσοδος) και μαύρου (έξοδος). Επιπλέον, πατώντας στο εσωτερικό των σχημάτων μπορεί να δώσει την περιγραφή που επιθυμεί για το εκάστοτε σχήμα. Η επιφάνεια εργασίας απεικονίζεται ως ένας πίνακας σχολικής τάξης. Στο κάτω μέρος του πίνακα υπάρχει ένας σπόγγος, ο οποίος στην πραγματικότητα αποτελεί ένα κουμπί με το οποίο ο χρήστης μπορεί να διαγράψει-σβήσει το σύνολο των στοιχείων (σχήματα και ενώσεις) έτσι ώστε να καθαρίσει ο πίνακας, δηλαδή η επιφάνεια εργασίας.

Τέλος, στο κάτω και αριστερά μέρος της επιφάνειας εργασίας υπάρχουν τέσσερα μικρά κουμπιά σε κάθετη διάταξη. Επιλέγοντας το κουμπί «+» ο χρήστης μπορεί να κάνει zoom in στην επιφάνεια εργασίας, ενώ αντίστοιχα με το κουμπί «-» κάνει zoom out. Επιλέγοντας το κουμπί με τις τέσσερις γραμμές κεντράρεται το υπάρχον διάγραμμα ροής, δηλαδή επιλέγεται αυτόματα το zoom in έτσι ώστε να φαίνεται ολόκληρο το διάγραμμα στο κέντρο της επιφάνειας εργασίας. Πατώντας το κουμπί με το σύμβολο της κλειδαριάς υπάρχει η δυνατότητα να «κλειδώνει» το διάγραμμα ροής έτσι ώστε να μη μετακινούνται τα σχήματα αλλά να είναι σταθερά.

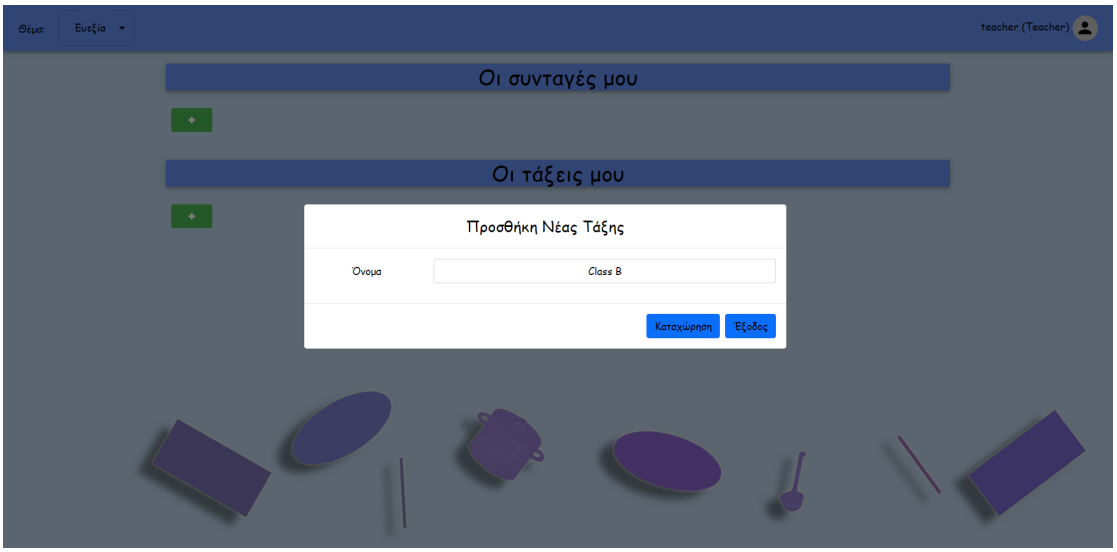
Τέλος, όταν ο χρήστης επιστρέφει στο προφίλ του μπορεί να επεξεργαστεί ή να διαγράψει τα διαγράμματα ροής που έχει αποθηκεύσει. Οι αποθηκευμένες συνταγές-διαγράμματα ροής απεικονίζονται η μια δίπλα στην άλλη εντός της ενότητας «Οι συνταγές μου».

6.1.2. Λειτουργίες εφαρμογής για τον ρόλο “teacher”



Εικόνα 25: Σελίδα profile ενός χρήστη τύπου teacher

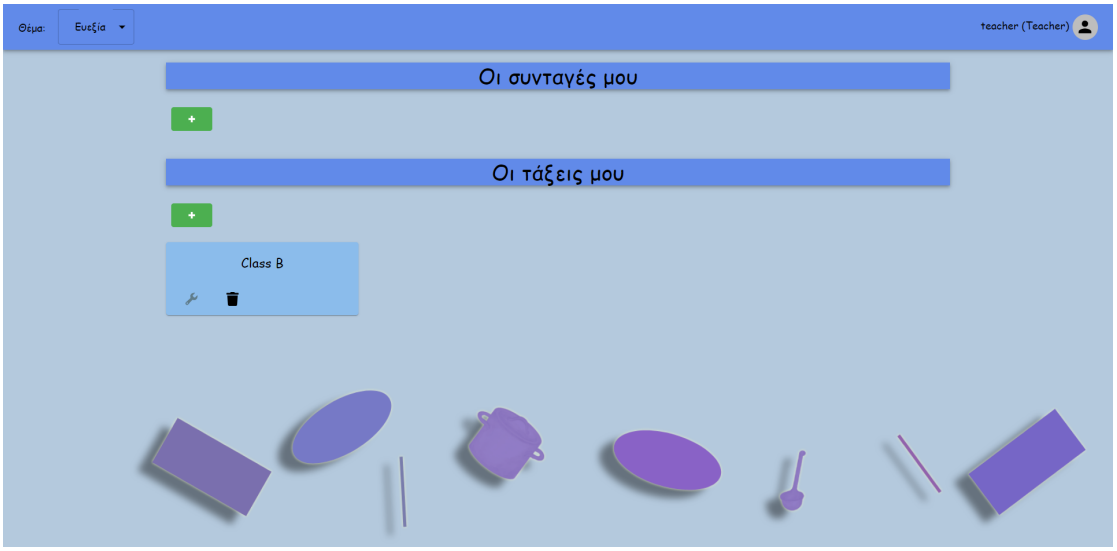
Ο ρόλος “teacher” έχει κάποιες επιπλέον δυνατότητες λειτουργιών σε σχέση με έναν απλό χρήστη. Πιο συγκεκριμένα, στη σελίδα του προφίλ του, εκτός από την ενότητα «Οι συνταγές μου», εμφανίζεται και η ενότητα «Οι τάξεις μου» (βλ. εικόνα 25). Στη συγκεκριμένη ενότητα, ο χρήστης της εφαρμογής έχει τη δυνατότητα να δημιουργήσει νέες τάξεις μαθητών επιλέγοντας το αντίστοιχο κουμπί «+».



Εικόνα 26: Modal προσθήκης νέας τάξης

Αφού επιλέξει το κουμπί, θα εμφανιστεί ένα Modal μέσα στο οποίο περιέχεται μια φόρμα για τη δημιουργία νέας τάξης. Ο χρήστης θα πρέπει να επιλέξει το όνομα της νέας τάξης και στη

συνέχεια να επιλέξει το κουμπί «Καταχώρηση» αν θέλει να αποθηκεύσει τη νέα τάξη ή το κουμπί «Έξοδος» αν θέλει να ακυρώσει τη συγκεκριμένη ενέργεια. Στη συνέχεια, αφού γίνει η καταχώρηση, η νέα τάξη θα εμφανιστεί στο πλέγμα κάτω από το κουμπί της προσθήκης.



Εικόνα 27: Απεικόνιση πλέγματος τάξεων

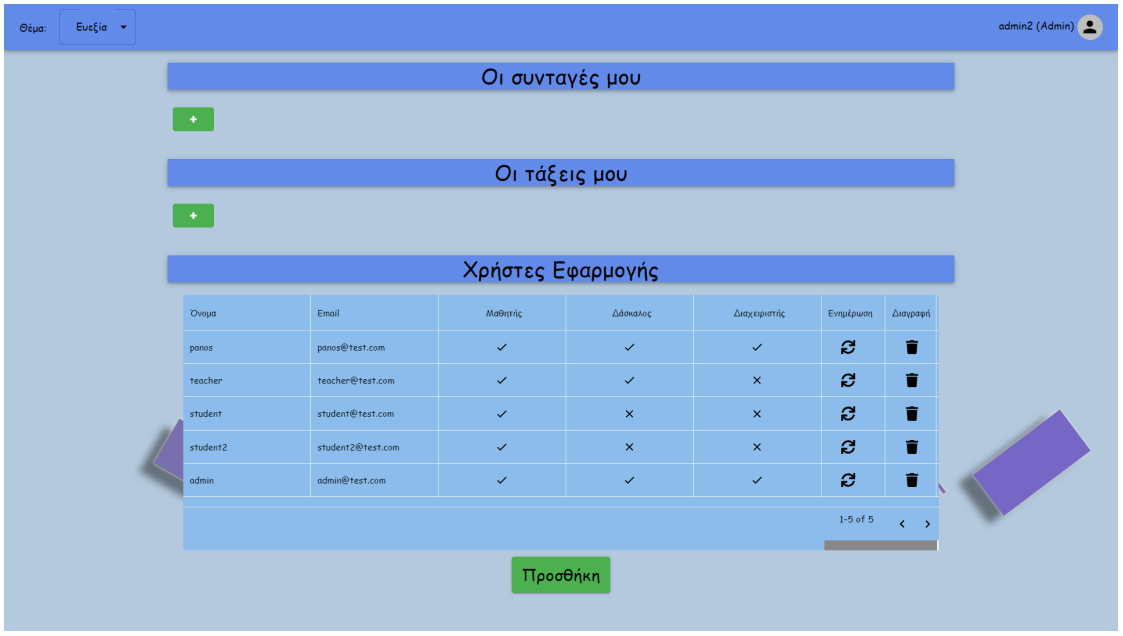
Σε αυτό το πλέγμα απεικονίζονται όλες οι τάξεις που ανήκουν στον διαχειριστή που είναι συνδεδεμένος. Έτσι ο χρήστης, εκτός από συνταγές-διαγράμματα ροής, έχει τη δυνατότητα να επεξεργαστεί ή να διαγράψει τάξεις μαθητών.

Εικόνα 28: Σελίδα επεξεργασίας τάξης μαθητών

Κατά την επιλογή του κουμπιού «Επεξεργασία» μιας τάξης ο χρήστης μεταφέρεται στη σελίδα utilities, η οποία είναι ξεχωριστή για κάθε τάξη δεδομένου ότι αναφέρεται σε στοιχεία της συγκεκριμένης τάξης. Σε αυτή τη σελίδα, υπάρχουν δύο ενότητες, η ενότητα «Προβολή Μαθητών Τάξης» και η ενότητα «Συνταγές Τάξης». Όπως φαίνεται και στην εικόνα 28, στην πρώτη ενότητα εμφανίζεται η λίστα όλων των χρηστών εφόσον έχουν την ιδιότητα του

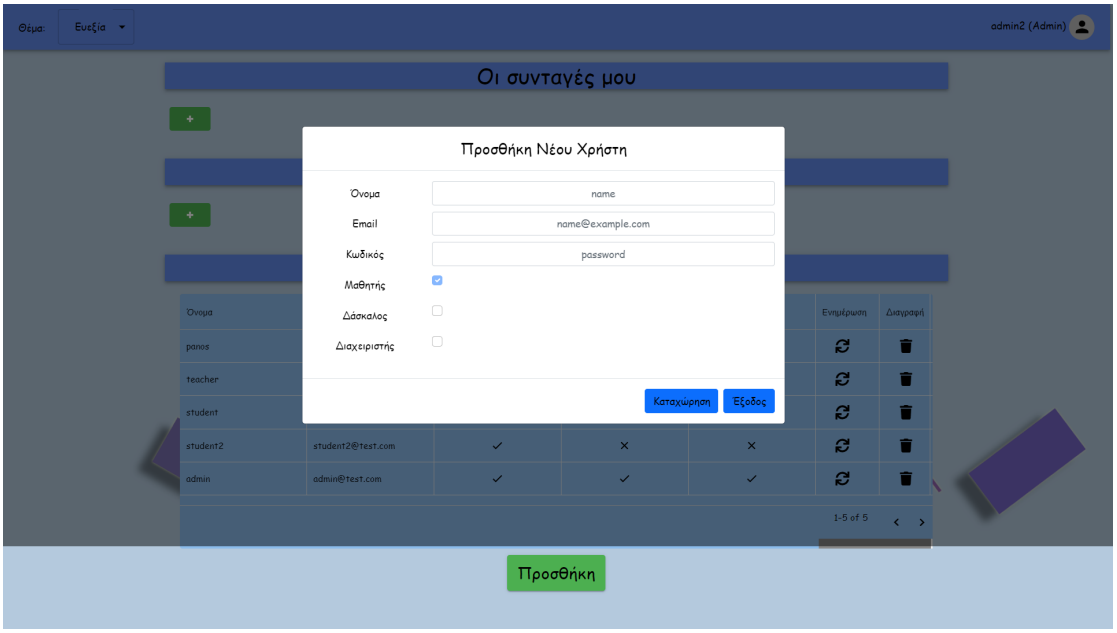
μαθητή “isStudent”. Ο χρήστης έχει τη δυνατότητα να προσθέσει ή να αφαιρέσει χρήστες στη συγκεκριμένη τάξη επιλέγοντας δίπλα από τα στοιχεία του κάθε χρήστη το κουμπί «Προσθήκη» ή «Αφαίρεση» αντίστοιχα. Στη δεύτερη ενότητα εμφανίζεται η επιλογή προσθήκης νέας συνταγής η οποία ανήκει στη συγκεκριμένη τάξη. Εφόσον ο χρήστης επιλέξει τη συγκεκριμένη ενέργεια μπορεί να δημιουργηθεί μια νέα συνταγή η οποία θα ανήκει στην τάξη και όσοι χρήστες-μαθητές ανήκουν σ΄αυτήν την τάξη θα εγγραφούν στη συγκεκριμένη συνταγή. Αυτό σημαίνει ότι θα δημιουργηθεί ένα αντίγραφο για κάθε μαθητή και ένα ακόμα αντίγραφο στο οποίο θα έχουν πρόσβαση όλοι και θα είναι κοινό. Αντίστοιχα με τη λογική των προηγούμενων ενοτήτων, ο χρήστης προσθέτει μια συνταγή πατώντας το κουμπί «+» και στη συνέχεια μπορεί να επεξεργάζεται ή να διαγράφει τις συνταγές από το αντίστοιχο πλέγμα συνταγών που δημιουργείται. Τέλος ο χρήστης μπορεί να δημιουργήσει και μία νέα συνταγή «Real Time». Ουσιαστικά με αυτή την ενέργεια θα δημιουργηθεί μια κοινή συνταγή για όλη την τάξη μέσα στην οποία μπορούν να εισέρχονται οι χρήστες και να επεξεργάζονται σε πραγματικό χρόνο τη συνταγή.

6.1.3. Λειτουργίες εφαρμογής για τον ρόλο “admin”



Εικόνα 29: Σελίδα profile ενός χρήστη τύπου admin

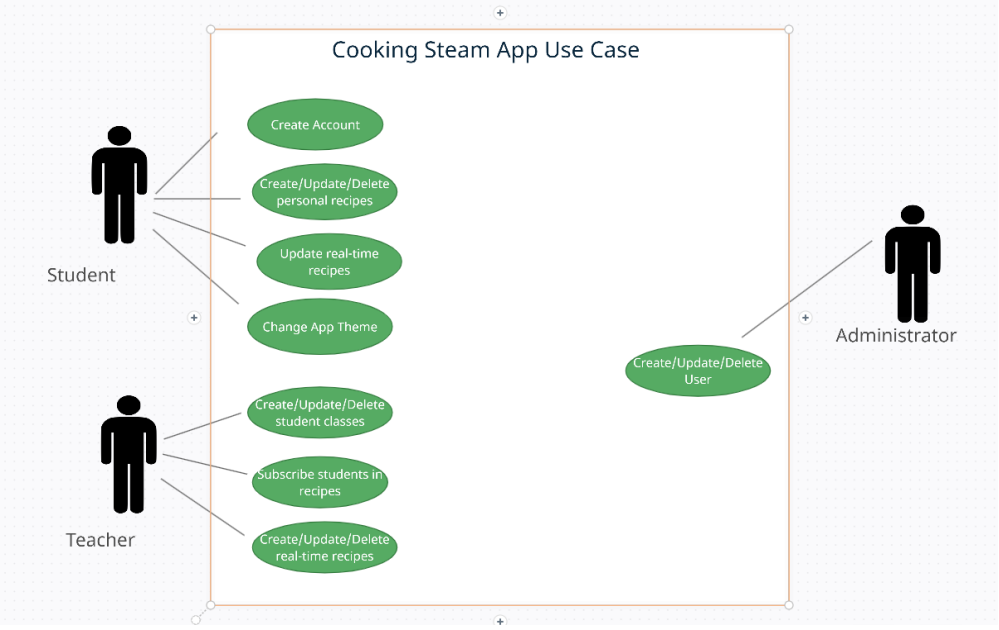
Ο ρόλος “admin” έχει κάποιες επιπλέον δυνατότητες λειτουργιών σε σχέση με έναν χρήστη teacher. Πιο συγκεκριμένα, στη σελίδα του προφίλ του, εκτός από την ενότητα «Οι συνταγές μου» και «Οι τάξεις μου», εμφανίζεται και η ενότητα «Χρήστες Εφαρμογής» (βλ. εικόνα 29). Στη συγκεκριμένη ενότητα, ο χρήστης της εφαρμογής έχει τη δυνατότητα να δημιουργήσει νέους χρήστες επιλέγοντας το αντίστοιχο κουμπί «Προσθήκη».



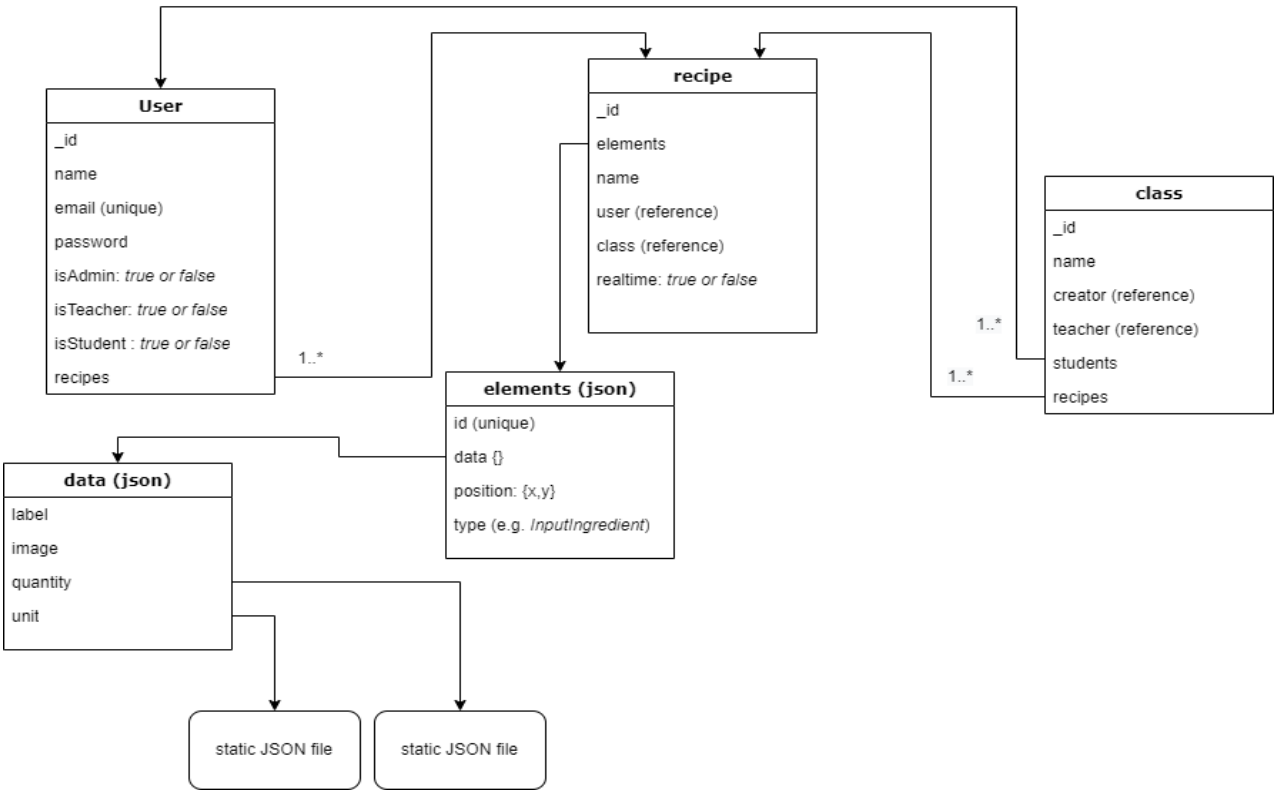
Εικόνα 30: Modal προσθήκης νέου χρήστη

Αφού επιλέξει το κουμπί, θα εμφανιστεί ένα Modal μέσα στο οποίο περιέχεται η φόρμα για τη δημιουργία νέου χρήστη (βλ. εικόνα 30). Ο χρήστης θα πρέπει να επιλέξει τα στοιχεία του νέου χρήστη και στη συνέχεια να επιλέξει το κουμπί «Καταχώρηση» αν θέλει να αποθηκεύσει το νέο χρήστη ή το κουμπί «Έξοδος» αν θέλει να ακυρώσει τη συγκεκριμένη ενέργεια. Στη συνέχεια, αφού γίνει η καταχώρηση, ο χρήστης θα εμφανιστεί στο αντίστοιχο πλέγμα. Σε αυτό το πλέγμα απεικονίζονται όλοι οι χρήστες της εφαρμογής. Έτσι ο χρήστης, εκτός από συνταγές-διαγράμματα ροής και τάξεις, έχει τη δυνατότητα να επεξεργαστεί και τους άλλους χρήστες της εφαρμογής.

Παρακάτω απεικονίζονται τα διαγράμματα περιπτώσεων χρήσης της εφαρμογής Cooking STEAM και το uml διάγραμμα που αφορά το μοντέλο δεδομένων της εφαρμογής:



Εικόνα 31: Διάγραμμα περιπτώσεων χρήσης της εφαρμογής



Εικόνα 32: Uml Διάγραμμα μοντέλων δεδομένων

6.2. Αρχές Σχεδίασης

Κατά την ανάπτυξη της εφαρμογής χρείαστηκε να παρθούν αποφάσεις για τον σχεδιασμό της. Τα χαρακτηριστικά στα οποία έπρεπε να δοθεί έμφαση αναλύονται στις παρακάτω παραγράφους.

6.2.1. Διάταξη Διεπαφής - Layout

Για την διάταξη των αντικειμένων χρησιμοποιήθηκαν Layout Managers που παρέχονται από την βιβλιοθήκη MUI της React. Αυτοί καθορίζουν πως θα εμφανίζεται η πληροφορία της οθόνης της εφαρμογής στην εκάστοτε οθόνη του υπολογιστή του χρήστη. Στην ουσία είναι η δομή των στοιχείων της κάθε οθόνης έτσι ώστε να μην έχει πρόβλημα στις διαφορετικές οθόνες, είτε αυτές είναι οθόνες επιτραπέζιου (desktop) είτε φορητού υπολογιστή (laptop) ή tablet.

6.2.2. Χρώματα και Γραφικά

Τα χρώματα αποτελούν μέρος της γλώσσας και του πολιτισμού μας και αποτελούν μεγάλο μέρος αυτού που μαθαίνουμε. Επιπλέον, βοηθούν στον προσδιορισμό του τρόπου με τον οποίο μαθαίνουμε. Τα χρώματα στέλνουν σήματα στον εγκέφαλο με αποτέλεσμα να αποσπούν την προσοχή ενός παιδιού ή να ενισχύουν τις δυνατότητες μάθησής του. Ο συνδυασμός του περιεχομένου μιας εκπαιδευτικής εφαρμογής με πολύχρωμα γραφικά μπορεί να βελτιώσει την απομνημόνευση ενός παιδιού. Έτσι τα χρώματα της εφαρμογής

επιλέχθηκαν με σκοπό να παίξουν σημαντικό ρόλο τόσο στη διάθεση των παιδιών όσο και στον τρόπο με τον οποίο θα μάθουν και θα απορροφήσουν την πληροφορία που τους παρέχει η συγκεκριμένη εφαρμογή. Πιο συγκεκριμένα, για τη βελτίωση της μάθησης και την επιρροή της διάθεσης των παιδιών επιλέχθηκαν 3 βασικοί μονοχρωματικοί συνδυασμοί χρωμάτων. Ο πρώτος συνδυασμός αφορά το κόκκινο χρώμα το οποίο αυξάνει την δημιουργικότητα και την όρεξη ενώ επίσης δημιουργεί αισθήματα εγρήγορσης και ενθουσιασμού. Ο δεύτερος συνδυασμός αφορά το μπλε χρώμα το οποίο συνήθως δημιουργεί μια αίσθηση ευεξίας. Τέλος, ο τρίτος συνδυασμός αφορά το κίτρινο χρώμα το οποίο δημιουργεί ένα θετικό συναίσθημα ενώ επίσης αποτελεί το βέλτιστο χρώμα για τη διατήρηση της προσοχής. Με την παραπάνω λειτουργικότητα οι μαθητές έχουν τη δυνατότητα να επιλέγουν τον τρόπο με τον οποίο επιτυγχάνεται η εφαρμογή της μάθησής τους μέσω της επιλογής του κατάλληλου θέματος.

6.2.3. Ευκολία πλοήγησης

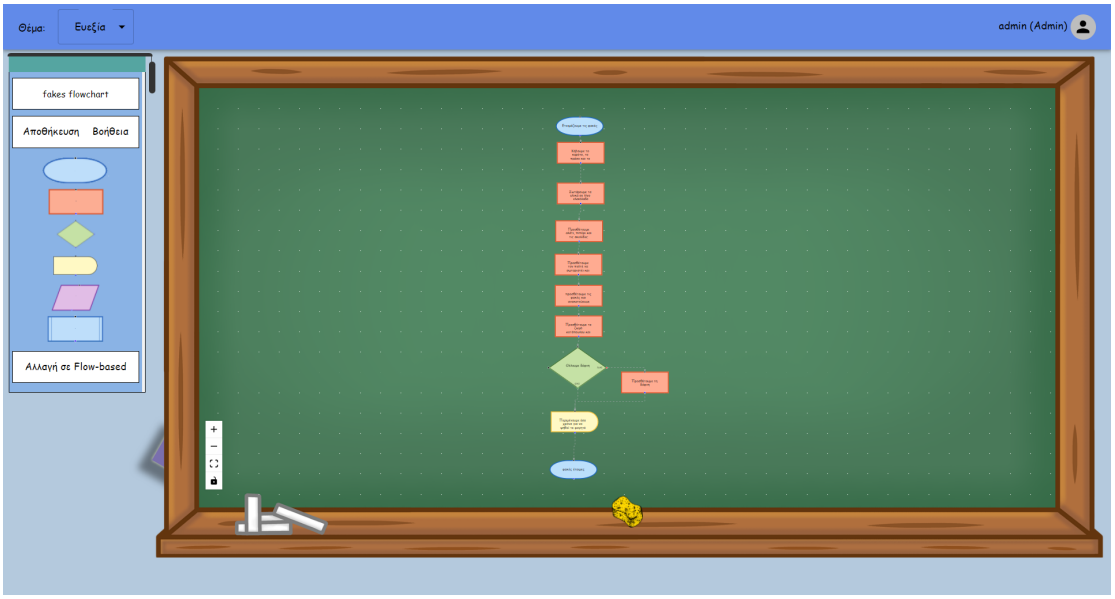
Στη διεπαφή που σχεδιάστηκε δώθηκε μεγάλη έμφαση στη κατανοητή αναπαράσταση των ενεργειών με εικονίδια καθώς και στην ομαδοποίηση των ενεργειών αυτών, έτσι ώστε ο χρήστης να μπορεί εύκολα να πλοηγείται μέσα στη διεπαφή. Χαρακτηριστικά παραδείγματα της εύκολης πλοήγησης που παρέχει η εφαρμογή είναι: το κουμπί νέας συνταγής που επιτρέπει στο χρήστη να μεταβεί στην οθόνη του Editor άμεσα.

6.2.4. Συνέπεια Διεπαφής

Η συνεπής εμφάνιση είναι πολύ σημαντική κατά την υλοποίηση μιας εφαρμογής. Οι λόγοι είναι ότι δημιουργεί μία αίσθηση σταθερότητας εμπνέοντας μεταξύ άλλων εμπιστοσύνη στο χρήστη. Επίσης βοηθάει τους χρήστες να απομνημονεύσουν καλύτερα τις πιθανές επιλογές που έχουν. Αυτό συμβαίνει γιατί όταν υπάρχει συνέπεια οι χρήστες συνδυάζουν στο μυαλό τους χρώματα με διαδικασίες, καθώς και θέσεις με πιθανές επιλογές. Για παράδειγμα όταν, ο χρήστης θέλει να μεταβεί στην αρχική σελίδα, το πρώτο πράγμα που θα κοιτάξει όταν θέλει να εκτελέσει αυτή την ενέργεια θα είναι το πάνω αριστερό μέρος της οθόνης.

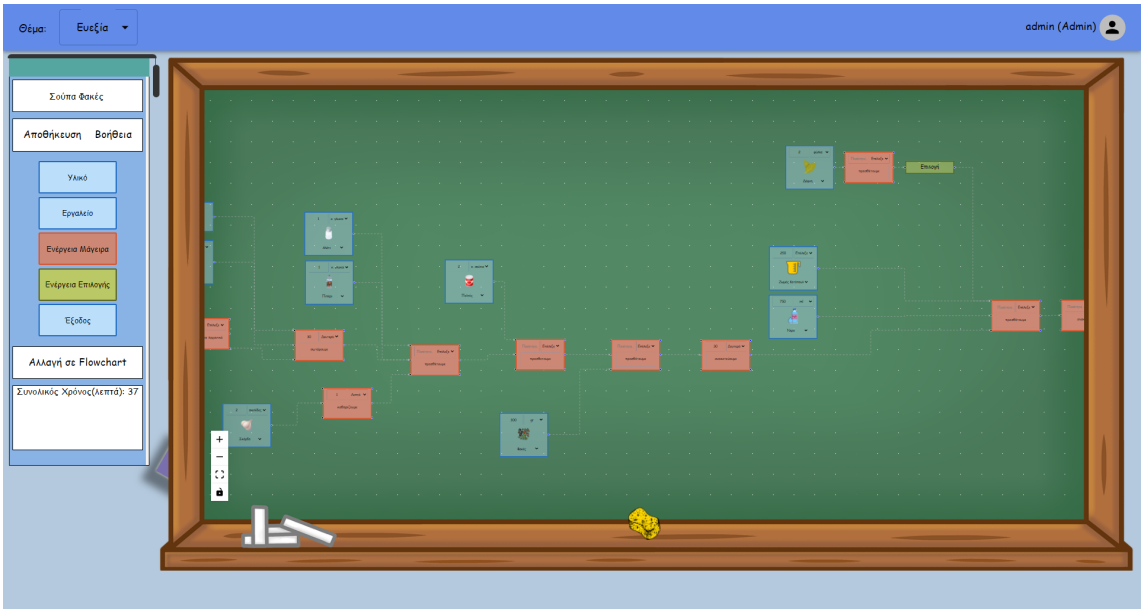
6.3. Editor

Είναι πολύ σημαντικό να γίνει μια εκτενέστερη αναφορά στην υλοποίηση του Editor εφόσον αποτελεί τη σημαντικότερη και πιο ουσιαστική λειτουργικότητα του συγκεκριμένου εργαλείου καθώς αποτελεί το χώρο παραγωγής και επεξεργασίας των διαγραμμάτων ροής για τον χρήστη. Αρχικά, κατά το πρώτο στάδιο της υλοποίησης ο Editor αφορούσε μόνο την κατασκευή και επεξεργασία κλασικών διαγραμμάτων ροής (flowchart diagram) το οποίο αποτελούσε και την αρχική προσέγγιση. Στην εικόνα 33 φαίνεται ο Editor καθώς και ένα παράδειγμα υλοποίησης διαγράμματος ροής με το εργαλείο cooking STEAM. Όπως αναφέρθηκε παραπάνω ο χρήστης μπορεί να επιλέξει μεταξύ έξι σχημάτων που αφορούν το κλασικό διάγραμμα ροής και να σχεδιάσει τη συνταγή του.



Εικόνα 33: Παράδειγμα χρήσης του Editor για κλασικό διάγραμμα ροής

Κατά τη διάρκεια της ανάπτυξης του Editor γεννήθηκε η ιδέα της υλοποίησης μιας άλλης εκδοχής και προσέγγισης για τον σχεδιασμό και την αναπαράσταση των συνταγών μαγειρικής. Ουσιαστικά η δεύτερη εκδοχή αφορά μια μη ντετερμινιστική προσέγγιση και αναπαράσταση των συνταγών μαγειρικής μέσω διαγραμμάτων. Ο χρήστης αφού επιλέξει το κουμπί «Αλλαγή σε flow-based» έχει τη δυνατότητα να επιλέξει ανάμεσα από ένα νέο σύνολο κόμβων τα οποία πλέον δεν αφορούν κλασικά σύμβολα διαγράμματος ροής.

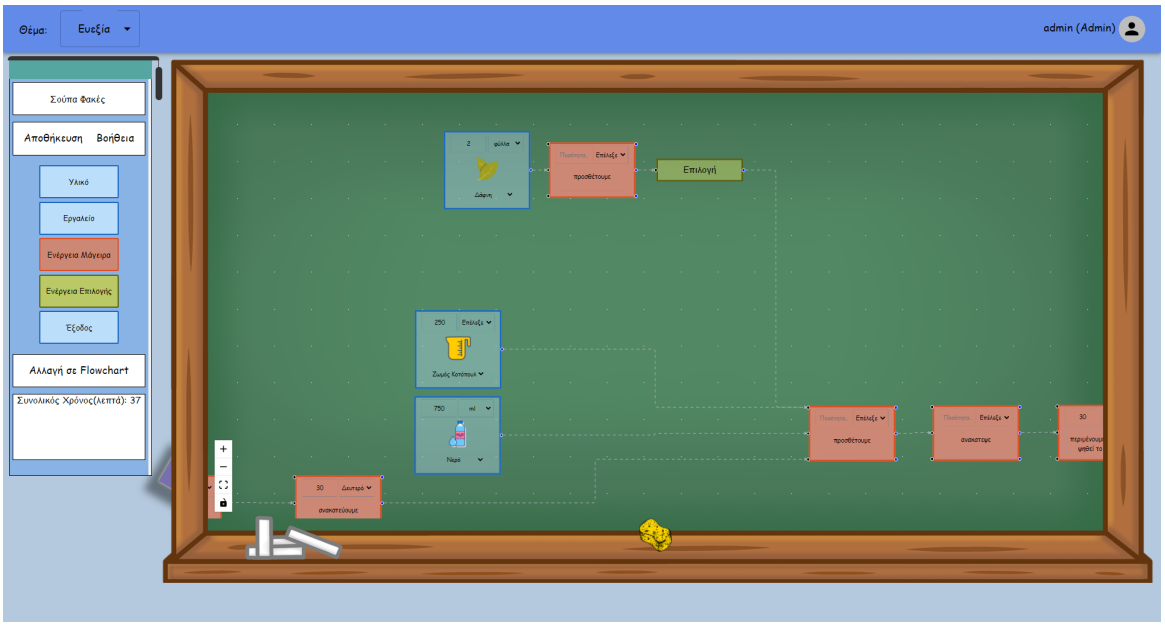


Εικόνα 34: Παράδειγμα χρήσης του Editor για μη ντετερμινιστικό διάγραμμα ροής

Πιο συγκεκριμένα, όπως φαίνεται και στην εικόνα 34 οι κόμβοι αυτοί αφορούν ένα υλικό που μπορεί να χρησιμοποιηθεί στη συνταγή. Ένας κόμβος τύπου «Υλικό» έχει τη δυνατότητα να δέχεται ως παραμέτρους την ποσότητα του υλικού αλλά και τη μονάδα μέτρησης για την ποσότητα που επιλέχθηκε. Έτσι μπορεί να απεικονιστεί η ποσότητα του συγκεκριμένου

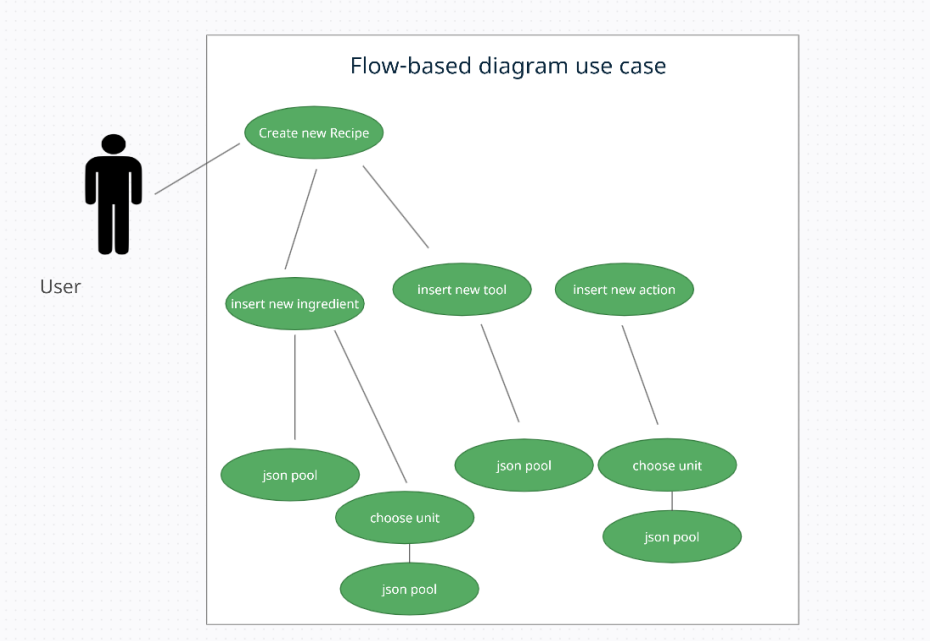
υλικού (π.χ. 100 gr). Οι μονάδες μέτρησης είναι συγκεκριμένες και μπορούν να επιλεχθούν μέσω ενός dropdown menu. Επιπλέον ο κόμβος του υλικού διαθέτει και ένα dropdown menu από το οποίο ο χρήστης μπορεί να επιλέξει μεταξύ συγκεκριμένων υλικών τα οποία περιγράφονται μέσω ενός στατικού json αρχείου, όπως συμβαίνει αντίστοιχα και για τις μονάδες μέτρησης. Έτσι επιτυγχάνεται μια πιο κατανεμημένη πληροφορία για το υλικό που χρησιμοποιείται και εισάγεται παράλληλα η έννοια των μεταβλητών.

Παρόμοια λογική αναπτύχθηκε και για τον κόμβο τύπου «Εργαλείο» κατά το οποίο ο χρήστης μπορεί να επιλέξει ένα από τα διαθέσιμα εργαλεία μέσω του dropdown menu. Τόσο τα υλικά όσο και τα εργαλεία μπορούν να χρησιμοποιηθούν ως είσοδοι στον κόμβο «Ενέργεια Μάγειρα». Ο συγκεκριμένος κόμβος έχει παρόμοια κατασκευή με τον κόμβο του υλικού. Πιο συγκεκριμένα ο χρήστης μπορεί να εισάγει μια ποσότητα χρόνου και να επιλέξει μεταξύ συγκεκριμένων μονάδων μέτρησης χρόνου που έχει στην διάθεσή του. Επιπλέον μπορεί να περιγράψει με μορφή κειμένου την ενέργεια (ανακατεύουμε, ψήνουμε κλπ). Ο συγκεκριμένος κόμβος μπορεί να δέχεται εισόδους αλλά μπορεί να χρησιμοποιηθεί και ως είσοδος σε έναν από τους δύο κόμβους «Ενέργεια Επιλογής» ή «Έξοδος». Η ενέργεια επιλογής μπορεί να χρησιμοποιηθεί ουσιαστικά ως διακόπτης, δηλαδή να δίνει την επιλογή στον χρήστη να συμπεριλαμβάνει στην υπόλοιπη συνταγή ή όχι τα βήματα που εκτελέστηκαν πριν από τον κόμβο της επιλογής. Ο κόμβος της εξόδου συμβολίζει το πέρας της συνταγής και δέχεται μόνο είσοδο. Στην εικόνα 35 φαίνεται ένα παράδειγμα χρήσης του κόμβου της επιλογής για την προσθήκη ενός υλικού στην συνταγή.



Εικόνα 35: Παράδειγμα χρήσης κόμβου επιλογής

Τέλος, όπως φαίνεται και από τις παραπάνω εικόνες στο κάτω και αριστερά μέρος της εφαρμογής υπάρχει η πληροφορία που αφορά τον συνολικό χρόνο της συνταγής. Η πληροφορία αυτή εμφανίζεται μόνο όταν πρόκειται για τη μη ντετερμινιστική εκδοχή του διαγράμματος και αλλάζει δυναμικά καθώς ο χρήστης εισάγει ή εξάγει κόμβους που περιέχουν την πληροφορία για τον χρόνο εκτέλεσης ενός βήματος.



Εικόνα 36: Απεικόνιση use case κατά τη δημιουργία flow-based διαγράμματος

7. Συμπεράσματα και Μελλοντικές Εργασίες

7.1. Συμπεράσματα και αποτελέσματα

Στα πλαίσια της παρούσας διπλωματικής εργασίας αναπτύχθηκε μια πρωτοποριακή εφαρμογή με τα κατάλληλα και πιο δημοφιλή εργαλεία. Η εφαρμογή μπορεί να χρησιμοποιηθεί στα πλαίσια διαφόρων εκπαιδευτικών προγραμμάτων και ημερίδων που αφορούν την εκπαίδευση παιδιών ηλικίας 10-12 ετών με την μεθοδολογία STEAM με αποκορύφωση τις διαδικασίες Cooking STEAM δεδομένου ότι ήταν η αφορμή να δημιουργθεί το Cooking STEAM App. Το εργαλείο δίνει τη δυνατότητα σε αυτές τις τάξεις να δουλέψουν και να χρησιμοποιήσουν μια ρεαλιστική εφαρμογή με αμφότερο σκοπό την καταγραφή και τον διαμοιρασμό των συνταγών μαγειρικής ως γραφική αναπαράσταση αλγορίθμων. Έτσι αξιοποιώντας τη γραφική απεικόνιση των αλγορίθμων επιτυγχάνεται η προσομοίωση και επεξεργασία ρεαλιστικών αλγοριθμικών διαδικασιών, όπου στη συγκεκριμένη περίπτωση είναι οι συνταγές μαγειρικής.

Η αρχική ιδέα και υλοποίηση του εργαλείου αφορούσε την απεικόνιση των συνταγών μαγειρικής μέσω των κλασικών διαγραμμάτων ροής. Άλλωστε μέχρι σήμερα σε αντίστοιχες ημερίδες και εκπαιδευτικά προγράμματα χρησιμοποιείται μόνο η συγκεκριμένη γραφική αναπαράσταση αφού είναι ευρέως γνωστή και κατανοητή. Μέσα από τη διαδικασία και την υλοποίηση της παρούσας διπλωματικής εργασίας, όπως προαναφέρθηκε και στο προηγούμενο κεφάλαιο, γεννήθηκε η ιδέα μιας διαφορετικής - μη ντετερμινιστικής γραφικής αναπαράστασης των συνταγών μαγειρικής η οποία όχι μόνο μπορεί να χρησιμοποιηθεί ως εναλλακτική προσέγγιση για την εκπαίδευση των μαθητών πάνω στο αντικείμενο, αλλά επιπλέον μπορεί να επεκταθεί ακόμα και σε ερευνητικό επίπεδο.

Πιο συγκεκριμένα, το δίπολο που δημιουργήθηκε μεταξύ των δύο γραφικών αναπαραστάσεων μας παρακινεί και μας οδηγεί σε μια πιο ερευνητική προσέγγιση καθώς γεννάται η ανάγκη να αποφασιστεί ποια από τις δύο μεθόδους είναι περισσότερο αποδοτική και κατάλληλη για την εκπαίδευση των μαθητών. Έτσι επιλέγοντας τα κατάλληλα εργαλεία και μεθοδολογίες θα γίνει η προσπάθεια της καταγραφής αυτής της πληροφορίας αλλά και η ανάδειξη του συγκεκριμένου εργαλείου.

7.2. Μελλοντικές εργασίες και επεκτάσεις εφαρμογής

Η συγκεκριμένη υλοποίηση και έκδοση της εφαρμογής Cooking STEAM αποτελεί μια βάση πάνω στην οποία μπορούν να αναπτυχθούν ακόμη περισσότερες λογικές και ενέργειες. Έτσι μπορεί να αναπτυχθεί ακόμη περισσότερο το περιεχόμενο και όλα όσα μπορεί να προσφέρει στην ευρύτερη κοινωνία και εκπαίδευση το συγκεκριμένο εργαλείο. Η εργασία διεκπεραιώθηκε μέσα σε ένα συγκεκριμένο χρονικό διάστημα, με αποτέλεσμα να τίθενται ζητήματα επέκτασης τα οποία περιγράφονται παρακάτω:

Οικονομία

Μία πιθανή επέκταση της εφαρμογής θα αφορούσε την ανάπτυξη λογικής για οικονομία εντός της εφαρμογής (νόμισμα, αγορά νέων προϊόντων και ενεργειών), έπαθλα κατά την εκπλήρωση διαφόρων στόχων και αποστολών αλλά και επιπλέον κίνητρα αξιοποίησης και χρήσης της εφαρμογής για τους μαθητές.

Μέσο Κοινωνικής Δικτύωσης

Η εφαρμογή Cooking STEAM θα μπορούσε να επεκταθεί με τέτοιο τρόπο ώστε να δίνει τη δυνατότητα στους χρήστες να επισκέπτονται προφίλ άλλων χρηστών έτσι ώστε να μπορούν να συνεργάζονται ακόμα περισσότερο και να ανταλλάσσονται ιδέες και υλοποιήσεις συνταγών. Επιπλέον θα μπορούσε να υπάρχει για τον κάθε χρήστη ένα ιστορικό ή και στατιστικά που αφορούν την απόδοσή του, πληροφορίες οι οποίες θα εμφανίζονταν στο κάθε προφίλ. Έτσι θα μπορούσαν να υπάρχουν και διάφορες κατηγορίες με στατιστικές για όλους τους χρήστες της εφαρμογής και να επιτυγχάνεται μ΄αυτόν τον τρόπο ένας υγιής ανταγωνισμός που θα ωθούσε και θα έδινε κίνητρο στους μαθητές για περισσότερη τριβή και μάθηση μέσω του συγκεκριμένου εργαλείου.

Εξαγωγή και Μετάφραση Συνταγής

Στην τρέχουσα έκδοση της εφαρμογής ουσιαστικά επιτυγχάνεται η απεικόνιση των συνταγών μαγειρικής μέσω διαγραμμάτων ροής. Μια πιθανή επέκταση θα μπορούσε να αφορά την υλοποίηση της συνταγής σε πραγματικό χρόνο. Θα μπορούσε λοιπόν να αναπτυχθεί μια λογική κατά την οποία θα δίνεται η δυνατότητα στον χρήστη να επιλέξει ένα κουμπί με το οποίο θα διαβάζεται η συνταγή με έναν έξυπνο αλγόριθμο και θα γίνεται μια αυτόματη παραγωγή και εξαγωγή των βημάτων συναρτήσει του χρόνου που απαιτείται για το κάθε βήμα.

Αντιγραφή Συνταγής

Αυτή είναι μια σημαντική λειτουργία για την ενεργοποίηση επαναχρησιμοποίηση συνταγών. Θα μπορεί να δίνει τη δυνατότητα στο χρήστη να δημιουργεί αντίγραφα από τα ήδη κατασκευασμένα διαγράμματα.

Παρόμοιες Εφαρμογές

Η εφαρμογή Cooking STEAM έχει αναπτυχθεί με τέτοιο τρόπο ώστε να απομονώνει το στοιχείο της μαγειρικής όσον αφορά το UI/UX σε επίπεδο προγραμματισμού. Αυτό σημαίνει ότι θα μπορούσε να αφαιρεθεί το στοιχείο της μαγειρικής από ολόκληρο το εικαστικό φάσμα της εφαρμογής και να εισαχθούν διαφορετικά εικαστικά στοιχεία που θα αφορούν την αντίστοιχη διαδικασία. Έτσι μπορεί να χρησιμοποιηθεί ο σκελετός της εφαρμογής και να γίνει ανάπτυξη σε επίπεδο εικαστικών έτσι ώστε η εφαρμογή να μπορεί να χρησιμοποιηθεί και σε άλλα πεδία. Για παράδειγμα θα μπορούσε να αναπτυχθεί η εφαρμογή Chemistry Steam με την οποία θα απεικονίζονται χημικά πειράματα και διαδικασίες μέσω των διαγραμμάτων ροής.

8. Βιβλιογραφία

[1] Nektarios Moumoutzis, Chara Xanthaki, Ioannis Maragkoudakis, Stavros Christodoulakis, Desislava Paneva-Marinova, Lilia Pavlova, Petros Lameras, Stavroula Misthou, George Kalmpourtzis Cooking STEAM: A Case Study on Establishing a STEAM Learning Community using a Performative Framework and Cooking.

[2] Kyfonidis, C., Moumoutzis, N., & Christodoulakis, S. Block-C: A block-based programming teaching tool to facilitate introductory C programming courses. 2017 IEEE Global Engineering Education Conference (EDUCON) (pp. 570-579). Athens: IEEE (2017).

[3] Moumoutzis, N., Christoulakis, M., Christodoulakis, S., & Paneva-Marinova, D. Renovating the Cultural Heritage of Traditional Shadow Theatre with eShadow. Design, Implementation, Evaluation and Use in Formal and Informal Learning. DiPP 2018 Conference on Digital Presentation and Preservation of Cultural and Scientific Heritage. Vol. 8, (pp. 51- 70). Sofia, Bulgaria.: Institute of Mathematics and Informatics – BAS, 2018, ISSN 1314- 4006 (Print), eISSN 2535-0366 (Online) (2018).

[4] Moumoutzis, N., Sifakis, Y., Christodoulakis, S., & Paneva-Marinova, D. A Reference Framework to Establish and Sustain Onlife Communities and Its Use. Rich Learning Experiences in History with ViSTPro. Digital Presentation and Preservation of Cultural and Scientific Heritage. Vol. 9, (pp. 27-42). Sofia, Bulgaria: Institute of Mathematics and Informatics – BAS, 2019, ISSN 1314-4006 (Print), eISSN 2535-0366 (Online) (2019).

[5] Moumoutzis, N., Boukeas, G., Vassilakis, V., Pappas, N., Xanthaki, C., Maragkoudakis, I, Deligiannakis, A., Christodoulakis, S. Design, Implementation and Evaluation of a Computer Science Teacher Training Programme for Learning and Teaching of Python Inside and Outside School. In Interactive Mobile Communication, Technologies and Learning (pp. 575-586). Cham: Springer (2017).

[6] Mylonakis, M., Arapi, P., Pappas, N., Moumoutzis, N., & Christodoulakis, S. Metadata management and sharing in multimedia open learning environment (MOLE). Research Conference on Metadata and Semantic Research (pp. 275-286). Berlin, Heidelberg: Springer (2011)

[7] Stylianakis, G., Moumoutzis, N., Arapi, P., Mylonakis, M., & Christodoulakis, S. COLearn and open discovery space portal alignment: A case of enriching open learning Infrastructures with collaborative learning capabilities. 2014 International Conference on Interactive Mobile Communication Technologies and Learning (IMCL2014) (pp. 252-256). IEEE (2014).

[8] Dr. Aparna Tembulkar, Prof. Upendra Lele. Potential for use of Educational App for young children . IOSR Journal of Research & Method in Education (IOSRJRME) · July 2021.

[9] Shinsuke Mori, Hirokuni Maeta, Yoko Yamakata, and Tetsuro Sasada. 2014. Flow Graph Corpus from Recipe Texts. In Proceedings of the Ninth International Conference on

Language Resources and Evaluation (LREC'14), pages 2370–2377, Reykjavik, Iceland. European Language Resources Association (ELRA).

[10] Taichi Nishimura, Suzushi Tomori, Hayato Hashimoto, Atsushi Hashimoto, Yoko Yamakata, Jun Harashima, Yoshitaka Ushiku, and Shinsuke Mori. 2020. Visual Grounding Annotation of Recipe Flow Graph. In Proceedings of the 12th Language Resources and Evaluation Conference, pages 4275–4284, Marseille, France. European Language Resources Association.

[11] Lyn English. 2016. Targeting all of STEM in the Primary School: Engineering Design as a Foundational Process.

[12] Janko Zufic. 2009. More efficient e-learning through design: color of text and background. World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (ELEARN) 2009:1 E-Learn 2009.

[13] Konrad Gos, Wojciech Zabierowski. 2020. The Comparison of Microservice and Monolithic Architecture. Conference: 2020 IEEE XVIth International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH).

[14] Mohadeseh Khazaee, Layla Sabourian. 2020. Improving Children's STEAM Education and Their Global Competence Through Collaborative Cooking.

[15]. React Flow https://reactflow.dev/

[16]. Express.js https://expressjs.com/

[17]. Docker https://docs.docker.com/

[18]. React.js https://reactjs.org/

[19]. npm packages https://www.npmjs.com/

[20]. Socket.io https://socket.io/docs/v4/

[21]. Redux https://redux.js.org/