Εργασία 3

Μάθημα: Επικοινωνία Αθρώπου Μηχανής

Ονόματα μελών:

Χαριτίνη Χαλακατέβα, Α.Μ.: 1115202100238

Ραλντιόν Κομίνι, Α.Μ.: 1115202000087

Θοδωρής Δημακόπουλος, Α.Μ.:1115201900048

//αρχείο για σημειωσεις σε σχεση με τον κωδικα κλπ

// να βαλω το συνδεσμο για το αποθευτηριο του GitHub

Η υλοποίηση της διεπαφής της εργασίας 3 έχει γίνει σε Vue.js

Bulk Grade Entry:

Υποστηρίζει μόνο αρχεία Excel (.xlsx) λόγω του χαρακτηριστικού *accept=".xlsx"* στο είσοδο αρχείου.
Εμποδίζει την προεπιλεγμένη υποβολή της φόρμας με το *@submit.prevent*.
Η μέθοδος *handleFileUpload* καλείται κατά την υποβολή της φόρμας. Παίρνει το εισαγόμενο αρχείο, ελέγχει αν έχει επιλεγεί το αρχείο, και κατόπιν εκτελεί ενέργειες επεξεργασίας.

Course Card:

Χρησιμοποιείται για την εμφάνιση πληροφοριών για ένα μάθημα (course).

Λαμβάνει ένα αντικείμενο *course* ως παράμετρο μέσω των props.

Χρησιμοποιεί τα δεδομένα αυτού του αντικειμένου για να εμφανίσει το όνομα του μαθήματος και τον διδάσκοντα.

Η χρήση μεταβλητών σαν *course.name* και *course.teacher* στο template δείχνει τη συντομία της μεταφοράς δεδομένων από το αντικείμενο *course*.

Έχει ενσωματωθεί έλεγχος μέσω των props για την ύπαρξη και τον τύπο του αντικειμένου *course*.

Courses for Students:

Γίνεται χρήση της δομής επανάληψης *v-for* για να πραγματοποιήσει επανάληψη μέσω των στοιχείων της λίστας **courses**.

Εντός κάθε επανάληψης, παρουσιάζονται τα δεδομένα του κάθε μαθήματος με τη χρήση της σύνταξης *{{ course.name }}* και *{{ course.teacher }}*.

Η χρήση ενός εξωτερικού * * για κάθε μάθημα επιτρέπει ευελιξία και εύκολη επέκταση του κώδικα.

Η λεπτομερής ορισμένη λίστα μαθημάτων και τα σχετικά δεδομένα διαχειρίζονται εντός του *data* block, διευκολύνοντας τη συντήρηση.

Dashboard (The home page)

Ο κώδικας είναι οργανωμένος με βάση τον ρόλο του χρήστη (student, professor, secretariat, visitor).

Χρησιμοποιεί τις δομές ελέγχου ροής *v-if, v-else-if, v-else* για δυναμική παρουσίαση δεδομένων ανάλογα με τον ρόλο του χρήστη.

Ορισμένα τμήματα κώδικα επαναχρησιμοποιούνται ανάμεσα σε διάφορους ρόλους, όπως για παράδειγμα, η παρουσίαση των μαθημάτων και των αποτελεσμάτων εξετάσεων.

Γίνεται χρήση του computed properties για να αποκτήσει πρόσβαση στα δεδομένα από τον store (*this.$store. State*), παρέχοντας καθαρότητα και αποφεύγοντας τον επαναληπτικό κώδικα.

Η παρουσίαση του προφίλ του χρήστη είναι κοινή για όλους τους ρόλους.

Για την περίπτωση που ο χρήστης δεν έχει συνδεθεί (*v-else*), παρουσιάζει ένα μήνυμα καλωσορίσματος.

db. json

Παρέχουμε ένα πίνακα με χρήστες σε μορφή JSON. Κάθε χρήστης έχει ένα μοναδικό αναγνωριστικό (id), ένα όνομα χρήστη (username), και έναν κωδικό πρόσβασης (password).

Edit Profile

Παρέχουμε μια φόρμα με πεδία επεξεργασίας για το όνομα (**name**) και το email (**email**) του χρήστη.

Χρησιμοποιεί το **v-model** για να συνδέσει δυναμικά τα πεδία εισαγωγής με τα δεδομένα του συστατικού.

Υπάρχει ένα κουμπί υποβολής (*<button type="submit">Save Changes</button>*).

Χρησιμοποιούνται τα δεδομένα του συστατικού για να αποθηκεύσουν το όνομα, το email και μια κατάσταση (*Updating profile...*) που δείχνει αν η ενημέρωση του προφίλ είναι σε εξέλιξη.

Η μέθοδος *updateProfile* καλείται όταν υποβάλλεται η φόρμα.

Εκτελείται μια HTTP POST αίτηση προς το */api/update-profile* με τα νέα δεδομένα.

Εμφανίζει μια ένδειξη φόρτωσης (*Updating profile...*) κατά τη διάρκεια της ανανέωσης.

Αν όλα πάνε καλά, γίνεται εμφάνιση ενός μηνύματος επιτυχίας στην κονσόλα, ενώ σε περίπτωση σφάλματος, γίνεται εμφάνιση ενός μηνύματος σφάλματος στην κονσόλα.

Γίνεται χρήση του Axios για την εκτέλεση HTTP αιτήσεων.

Grade Monitoring

Γίνεται χρήση του *v-for* για τη δυναμική παρουσίαση των βαθμολογιών ανά μαθητή.

Η μέθοδος *getCourseName* χρησιμοποιείται για να επιστρέψει το όνομα ενός μαθήματος δεδομένου του *course id*. Προσπαθεί να βρει το μάθημα στον πίνακα *teacherCourses* με βάση το *id*.

Επίσης υπάρχουν δύο σύνολα δεδομένων:

teacherGrades: Οι βαθμοί των μαθητών, με πληροφορίες όπως το όνομα του μαθητή, το ID του μαθήματος και ο βαθμός.

***teacherCourses**:* Ο πίνακας με τις πληροφορίες των μαθημάτων, που περιέχουν το ID, το όνομα του μαθήματος και το όνομα του δασκάλου

Η μέθοδος *getCourseName* βρίσκει το μάθημα με βάση το *course id* και επιστρέφει το όνομα του. Εάν το μάθημα δε βρεθεί, επιστρέφει το ίδιο το *course id***.**

Login

Παρέχεται μια φόρμα που περιλαμβάνει πεδία για το όνομα χρήστη (**username**) και τον κωδικό πρόσβασης (**password**).

Η Μέθοδος *handle Submit* καλείται όταν υποβάλλεται η φόρμα.

Κάνει μια αιτηση POST στο *http://localhost:3000/users* με τα δεδομένα σύνδεσης (username και password).

Σε περίπτωση επιτυχίας, εκτελείται η εντολή *this.$store.commit('setUser', response.data.user)* για να αποθηκευθεί ο χρήστης στο store της εφαρμογής, και στη συνέχεια γίνεται ανακατεύθυνση *(this.$router.push('/'))*. Ενώ σε περίπτωση αποτυχίας, η μεταβλητή **error** ορίζεται σε "Login failed!" για εμφάνιση μηνύματος σφάλματος.

Χρησιμοποιείται το Axios για την εκτέλεση HTTP αιτήσεων.

Μετά την επιτυχή σύνδεση, ο χρήστης ανακατευθύνεται στην αρχική σελίδα ('/'), ενώ αν υπάρξει σφάλμα κατά τη σύνδεση, το μήνυμα σφάλματος εμφανίζεται στον χρήστη.

main.js

Εισάγει το Vue και το αρχείο App.vue, καθώς και το αποθετήριο (store) που περιέχει την κατάσταση της εφαρμογής.

Γίνεται χρήση του *Vue.config.productionTip = false;* για να απενεργοποιήσει τις συμβουλές παραγωγής στην κονσόλα.

Γίνεται δημιουργία ενός νέου αντικειμένου Vue.

Γίνεται χρήση του App.vue ως το κεντρικό στοιχείο (component) που θα απεικονίζεται.

Χρησιμοποιείται το αποθετήριο (store) που εισήχθη προηγουμένως.

Το *$mount('#app')* συνδέει την Vue εφαρμογή με το DOM, εισάγοντας το περιεχόμενο της σελίδας με την ταυτότητα *app*.

package. json

"name": Το όνομα του έργου είναι "my-vue-app".
"version": Η έκδοση του έργου είναι "1.0.0".

"description": Μια περιγραφή του έργου ως "My Vue.js Application".
"serve": Χρησιμοποιεί το Vue CLI για να ξεκινήσει τον τοπικό διακομιστή ανάπτυξης.
"build": Χρησιμοποιεί το Vue CLI για να δημιουργήσει ένα παραγωγικό build της εφαρμογής.
"lint": Χρησιμοποιεί το Vue CLI για να ελέγξει τον κώδικα για πιθανά λάθη.
"axios": Η έκδοση "^0.21.1" της βιβλιοθήκης Axios για τη διευκόλυνση των HTTP αιτημάτων.
"vue": Η έκδοση "^2.6.14" του Vue.js, το κύριο πλαίσιο για την ανάπτυξη της εφαρμογής.
"vuex": Η έκδοση "^3.6.2" του Vuex, για τη διαχείριση της κατάστασης της εφαρμογής.
"@vue/cli-plugin-babel": Πρόσθετο του Vue CLI για το Babel.
"@vue/cli-service": Το βασικό πακέτο υπηρεσιών του Vue CLI.
"vue-template-compiler": Ο μεταγλωττιστής προτύπου του Vue.
Διακύμανση Περιηγητών (Browserslist): Το έργο υποστηρίζει τους περιηγητές που έχουν μερίδιο αγοράς μεγαλύτερο από 1%, τις τελευταίες 2 εκδόσεις κάθε περιηγητή, και δεν έχουν αποσυρθεί.

Register

Περιέχει φόρμα με πεδία για το όνομα χρήστη (**username**), τον κωδικό πρόσβασης (**password**) και την επιβεβαίωση κωδικού πρόσβασης (**password Confirm**).

Καλείται όταν υποβάλλεται η φόρμα.
Έλεγχος αν οι κωδικοί πρόσβασης ταιριάζουν. Εάν δεν ταιριάζουν, η μεταβλητή **error** ορίζεται σε "Passwords do not match."
Σε περίπτωση που ο έλεγχος περάσει, γίνεται αίτηση POST στο */api/register* με τα δεδομένα εγγραφής.
Σε περίπτωση επιτυχίας, η μεταβλητή *error* ορίζεται σε *null*, αποθηκεύεται ο χρήστης στο store της εφαρμογής, και ο χρήστης ανακατευθύνεται στην αρχική σελίδα ('**/**').
Σε περίπτωση αποτυχίας, η μεταβλητή **error** ορίζεται σε "Registration failed!"
Γίνεται χρήση του Axios για την εκτέλεση HTTP αιτήσεων.
Εάν υπάρξει σφάλμα κατά την εγγραφή, το μήνυμα σφάλματος εμφανίζεται στον χρήστη.

Request Certificate

Περιλαμβάνει ένα *<form>* με πεδία για την επιλογή μαθήματος (**course**) από ένα dropdown και τον λόγο υποβολής αιτήματος (**reason**) μέσω ενός πεδίου *textarea*.
Η μεταβλητή *courses* προσδιορίζει τα μαθήματα που είναι διαθέσιμα για επιλογή.
Οι μεταβλητές *course, reason* και *error* χρησιμοποιούνται για τη διατήρηση της κατάστασης της φόρμας και τυχόν σφαλμάτων.
Καλείται όταν υποβάλλεται η φόρμα.
Γίνεται χρήση του Axios για να κάνει μια ασύγχρονη αίτηση POST στον server (στο *http://localhost:3000/certificate-request*).
Σε περίπτωση επιτυχούς απάντησης, εμφανίζει ένα μήνυμα καταγραφής στην κονσόλα, επαναφέρει τη φόρμα (αδειάζοντας τα πεδία), και ορίζει το *error* σε *null*. Ενώ σε περίπτωση σφάλματος, καταγράφει το σφάλμα στην κονσόλα, ορίζει το *error* σε "Error submitting request."

Secretary Profile

Προβάλλει το όνομα χρήστη (**username**) και το email (**email**) της γραμματείας.
Προβάλλει μια λίστα από ανακοινώσεις (**announcements**).
Κάθε ανακοίνωση περιλαμβάνει τίτλο (**title**), περιεχόμενο (**content**), και ημερομηνία (**date**).
Παρέχει ένα φόρμα για τη δημιουργία νέας ανακοίνωσης.
Τα πεδία *newAnnouncementTitle* και *newAnnouncementContent* χρησιμοποιούνται για να δεχτούν τίτλο και περιεχόμενο της νέας ανακοίνωσης.
Η μέθοδος *createAnnouncement* προσθέτει τη νέα ανακοίνωση στον πίνακα *announcements* με ένα νέο ID, τίτλο, περιεχόμενο και τρέχουσα ημερομηνία.

STORE.js

Η *κατάσταση (state)* περιέχει τα δεδομένα που μοιράζονται και διαχειρίζονται από τον store.
user: Περιέχει πληροφορίες σχετικά με τον συνδεδεμένο χρήστη.
***courses**:* Λίστα μαθημάτων που αποθηκεύονται στο state.

exams: Λίστα εξετάσεων που αποθηκεύονται στο state.
error: Περιέχει πληροφορίες σχετικά με οποιοδήποτε σφάλμα που παρουσιάζεται.
Οι *μεταλλάξεις (mutations)* αλλάζουν την κατάσταση του store.
setUser: Ορίζει τα δεδομένα του χρήστη.
***addCourse**:* Προσθέτει ένα μάθημα στη λίστα μαθημάτων.
addExam: Προσθέτει μια εξέταση στη λίστα εξετάσεων.
setError: Ορίζει ένα σφάλμα.
Οι *Δράσεις (Actions)* προσφέρουν μια διεπαφή για τα στοιχεία εισόδου του χρήστη.
login: Πραγματοποιεί ασύγχρονη αίτηση στο */api/login* για σύνδεση. Εφόσον είναι επιτυχής, καλεί τη μετάλλαξη *setUser*. Σε περίπτωση αποτυχίας, καλεί τη μετάλλαξη *setError*.
register: Πραγματοποιεί ασύγχρονη αίτηση στο */api/register* για εγγραφή. Εφόσον είναι επιτυχής, καλεί τη μετάλλαξη *setUser*. Σε περίπτωση αποτυχίας, καλεί τη μετάλλαξη *setError*.

Student Profile

Γίνεται εμφάνιση των στοιχείων του χρήστη, όπως το όνομα χρήστη και το email.
Επίσης εμφανίζονται:

- Τα μαθήματα στα οποία είναι εγγεγραμμένος ο φοιτητής.
- Οι βαθμοί που έχει λάβει ο φοιτητής σε διάφορα μαθήματα.
- Το ιστορικό των μαθημάτων που έχει δηλώσει ο φοιτητής.
- Το ιστορικό των πιστοποιητικών που έχει αποκτήσει ο φοιτητής. Κάθε εγγραφή περιλαμβάνει τον τύπο του πιστοποιητικού και την κατάστασή του, καθώς και ένα κουμπί για να κατεβάσει το πιστοποιητικό.

Υπάρχει η φόρμα για:

- Να δηλώσει τον εαυτό του σε ένα νέο μάθημα. Μετά την υποβολή της φόρμας, καλείται η μέθοδος *submitDeclaration* που κάνει αίτηση POST στον server και ενημερώνει το ιστορικό δηλώσεων.
- Να υποβάλει αίτηση για πιστοποιητικό. Μετά την υποβολή της φόρμας, καλείται η μέθοδος *submitCertificateRequest* που κάνει αίτηση POST στον server και ενημερώνει το ιστορικό πιστοποιητικών.

Teacher Grading System

Πεδία Δεδομένων (Data Properties):

- *studentName:* Το όνομα του φοιτητή για τον οποίο θα υποβληθεί ο βαθμός.
- *selectedCourse:* Το επιλεγμένο μάθημα για τον οποίο θα υποβληθεί ο βαθμός.
- *grade:* Ο βαθμός που θα υποβληθεί.
- *teacherCourses:* Λίστα με τα μαθήματα που διδάσκει ο εκπαιδευτικός.

Μέθοδος submitGrade:

- Καλείται όταν υποβάλλεται η φόρμα.
- Έλεγχος αν όλα τα πεδία έχουν συμπληρωθεί. Αν όχι, εμφανίζεται ένα μήνυμα σφάλματος.
- Κάνει αίτηση POST στον server (υποθέτοντας το endpoint */api/submitGrade*) με τα δεδομένα της φόρμας.
- Εμφανίζει μηνύματα κατάλληλα για την επιτυχή υποβολή ή τυχόν σφάλμα.

Teacher Profile

- Εμφανίζει τα μαθήματα που διδάσκει ο καθηγητής με χρήση ενός υποσυστήματος συνιστωσών *<course-card>*.
- Παρέχεται φόρμα για την υποβολή βαθμολογίας για έναν φοιτητή.
- Εμφανίζει μια λίστα με τις βαθμολογίες που έχουν υποβληθεί από τον καθηγητή.

Teacher Grading System:

- Η φόρμα περιέχει πεδία για το όνομα του φοιτητή (**studentName**), το επιλεγμένο μάθημα (**selectedCourse**), και το βαθμό (**grade**).
- Η μέθοδος **submitGrade** καλείται όταν ο χρήστης υποβάλλει τη φόρμα. Στο κώδικα απλά εμφανίζει στο console τα δεδομένα της φόρμας.

Grade Monitoring:

- Εμφανίζει μια λίστα με τις βαθμολογίες που έχουν υποβληθεί από τον καθηγητή. Χρησιμοποιεί τη μέθοδο *getCourseName* για να ανακτήσει το όνομα του μαθήματος με βάση το **courseId**.

Bulk Grade Entry:

- Παρέχει μια φόρμα για τη μαζική καταχώρηση βαθμών, με δυνατότητα ανεβάσματος ενός αρχείου CSV.
- Η μέθοδος *handleFileUpload* εκτελείται όταν ο χρήστης υποβάλλει τη φόρμα. Προς το παρόν, εμφανίζει στο console τις πληροφορίες του αρχείου που ανέβηκε.

User Profile

Ο κώδικας είναι προσαρμοσμένος για να εμφανίζει τα στοιχεία προφίλ του χρήστη, καθώς και τα διαθέσιμα *μαθήματα (courses)* και *εξετάσεις (exams)* για τον συγκεκριμένο χρήστη.

Visitor Profile

Ο κώδικας είναι προσαρμοσμένος για να εμφανίζει το προφίλ ενός επισκέπτη , συμπεριλαμβανομένων γενικών *ανακοινώσεων (General Announcements)*. Χρησιμοποιείται η λειτουργία *v-for* για τη δημιουργία λίστας ανακοινώσεων.