Promedio Documentation

Gabriel Vainer
Alejandro Iglesias Llobett
Armando Rojas
Maaz Hashmi
Tegh Mehta
Preyansh Dutta

Table of Contents

Backend

Endpoints

- **/api/login**
    - **POST**
        - This is used in the Promedio/frontend/src/routes/Login.js to send a post request to mongoDB in order to compare the credentials with the users in the database to check if the user exists. If a user exists, the user is logged in and a JWT is created.
        - Request Body: { email: string, password: string }
        - Response: { name: string, email: string, gpa: string, token: string }

- **/api/register**
    - **POST**
        - This is used in the Promedio/frontend/src/routes/Signup.js to send a post request to mongoDB in order to create a new user with the following information: Name, Email, Password.
        - Request Body: { name: string, gpa: string, email: string, password: string }
        - Response: { name: string, email: string, gpa: string, token: string }

- **/api/update**
    - **POST**
        - This is used in the Promedio/frontend/src/routes/User.js to edit the user details. The fields Name, and Email can be updated through a post request.
        - Request Body: { name: string, gpa: string, email: string, password: string }
        - Response: { name: string, email: string, gpa: string }

- **/api/semester/**
    - **GET**
        - This is used in the Promedio/frontend/src/routes/Home.js to display all semesters mapped to an individual user.
        - Response: { [{ name: string, userId: Mongoose.Schema.Types.ObjectId }] }
    - **POST**
        - This is used in the Promedio/frontend/src/routes/Home.js to create a new semester (i.e. a new document within the "Semester" Model/"semesters" collection) mapped uniquely to a user . The semester name is passed via the request body.
        - Request Body: { semesterName: string }
        - Response: { message: "Semester added successfully.", semester: { name: string, userId: Mongoose.Schema.Types.ObjectId }}
    - **DELETE**

        - This is an extra function to use for debugging in case there's a need to delete all semesters related to a user.
        - Response: { message: "Successfully deleted (x) semesters and all related courses."}

- **/api/semester/:semesterName**
    - **GET**
        - This is used in the Promedio/frontend/src/routes/Home.js to list the details of a specific semester.
        - Request Parameters: { semesterName: string }
        - Request Body: { name: string, userId: Mongoose.Schema.Types.ObjectId }
    - **PATCH**
        - This is used in the Promedio/frontend/src/routes/Home.js to update an existing semester (i.e. a current document with id "semesterId" within the "Semester" Model/"semesters" collection) with a new name (as it is currently the only updateable field within the semester schema). Nothing happens otherwise within the database if no such semester exists.
        - Request Parameters: { semesterId: string }
        - Request Body: { semesterName: string }
        - Request Body: { message: "Semester updated successfully." }
    - **DELETE**
        - This is used in the Promedio/frontend/src/routes/Home.js to delete an existing semester (i.e. a current document with id "semesterId" within the "Semester" Model/"semesters" collection). Nothing happens otherwise within the database if no such semester exists.
        - Request Parameters: { semesterId: string }
        - Request Response: { message: "Semester deleted successfully." }

- **/api/course**
    - **GET**
        - This is used in the Promedio/frontend/src/routes/Home.js to display all courses for a specific user.
        - Request Response: [{ name: string, semesterId: Mongoose.Schema.Types.ObjectId }]
    - **POST**
        - This is used in the Promedio/frontend/src/routes/Home.js to create a new course (i.e. a new document within the "Course" Model/"courses" collection) mapped uniquely to a semester).
        - Request Body: { semester: string, name: string, description: string, markGoal: string }
        - Request Response: { message: "Course added successfully.", course: { name: string, semesterId: Mongoose.Schema.Types.ObjectId, description: string, markGoal: string }}
    - **DELETE**

        - This is an extra function to use for debugging in case there’s a need to delete all courses attached to an user.
        - Request Response: [{ message: “Successfully deleted [x] courses.”}]

- **/api/course/:courseId**
    - **GET**
        - This is used in the Promedio/frontend/src/routes/Home.js to list the details of a specific course, as well its respective gpa calculations. A user should also be able to play with the course details for that requested course (i.e. accessing the CSCA08 course in the Fall 2020 semester to set a mark goal) after fetching.
        - Request Parameters: { courseId: string}
        - Request Response: { name: string, semesterId: Mongoose.Schema.Types.ObjectId, description: string, markGoal: string }
    - **PATCH**
        - This is used in the Promedio/frontend/src/routes/Home.js to update an existing course (i.e. a current document with id “courseId” within the “Course” Model/“courses” collection) with a new name, description or mark goal. Nothing happens otherwise within the database if no such course exists.
        - Request Parameters: { courseId: string }
        - Request Body: { name: string, description: string, markGoal: string }
        - Request Response: { message: “Course updated successfully.” }
    - **DELETE**
        - This is used in the Promedio/frontend/src/routes/Home.js to delete an existing course (i.e. a current document with id “courseId” within the “Course” Model/“courses” collection). Nothing happens otherwise within the database if no such course exists.
        - Request Parameters: { courseId: string }
        - Request Response: { message: “Course deleted successfully.” }
- **/api/addCategory**
    - **POST**
        - This is used in Promedio/frontend/src/components/EditCategoryForm.js to add a new category to a specific course. A category handles part of the cGPA calculation for that specific course.
        - Request Body: { name: string, weight: number, numAssessments: number, marks: Array<int>}
        - Request Response: { message: “Category added successfully.” }}
- **/api/editCategory**
    - **POST**
        - This is used in Promedio/frontend/src/components/EditCategoryForm.js to edit an existing category in relation to a specific course. A category handles part of the cGPA calculation for that specific course. A user can change the name, course, weight, number of assessments and marks of the category.
        - Request Body: { id: string, courseId: string, name: string, weight: number, numAssessments: number, marks: Array<int>}

        - Request Response: { message: “Category edited successfully.” }}

- **/api/deleteCategory**
    - **POST**
        - This is used in Promedio/frontend/src/components/Course.js to delete an existing category in relation to a specific course. A category handles part of the cGPA calculation for that specific course.
        - Request Body: { id: string }
        - Request Response: { message: “Category deleted successfully.” }}
- **/api/categories/:courseId**
    - **POST**
        - This is used in Promedio/frontend/src/components/Course.js to get all the categories in relation to a course. A category handles part of the cGPA calculation for that specific course. By getting all of them, we can use their averages to calculate the overall cGPA for the course.
        - Request parameters: { courseId: string }
        - Request Response: [{ id: string, courseId: string, name: string, weight: number, numAssessments: number, marks: Array<int>}]
- **/calculations/**
    - **GET**
        - This is used in various Promedio frontend endpoints to dynamically calculate a grade as soon as any changes have been made to a user’s semester or course.
        - Request parameters: { courseIds: Array <String>,   name: string, weight: number, numAssessments: number, marks: Array<int>}
        - Request response: { GPA: float }

Frontend

Routes:

- **AllCourses.js**
  - The page will display all of the courses available for the user, regardless of the semester they are under. A user can also add a new course from this page.
    - onClick() for the `Add Course` button will prompt a dialog where the user can fill in information for the new course they're about to add.

- **AllSemesters.js**
  - The page will display all of the semesters available for the user. A user can also add a new semester from this page. This is the first thing a user now sees when they log in.
    - onClick() for the `Add Semester` button will prompt a dialog where the user can fill in information for the new semester they're about to add.

- **App.js**
  - All the routes are loaded and rendered under App.js. Also determines if the user is logged in or not and routes accordingly.

- **Course.js**
  - Course.js is the main hub of information for a course: its name and all related categories will be displayed here. A user can add/edit/delete categories related to courses.
    - All of the three actions above have related buttons with their respective actions that interact with the backend.
- **GPA.js**
  - Course.js is the main hub of information for a course: its name and all related categories will be displayed here. A user can add/edit/delete categories related to courses.
  - **THIS DOCUMENTATION IS NOT FINISHED -- ALEJANDRO NEED TO DO BY TODAY**
    - All of the three actions above have related buttons with their respective actions that interact with the backend.

- **Home.js**
  - A legacy route for the original solution for displaying all courses and semesters.
    - Welcome() returns the components for the Welcome Page

- **Login.js**
  - Screen that appears if the user selects the login button on the homepage. The user will then input their information accordingly and the site will check if the user is in the database.
    - handleClick() handles the data that is entered from the user and determines whether they are an existing user. If not, then we send an error message from the backend.

- **Semester.js**
  - A page that displays information relating to a specific semester. From this page, a user can add/edit/delete courses in relation to this semester.
    - All of the three actions above have related buttons with their respective actions that interact with the backend.

- **Signup.js**
  - Screen that appears if the user selects the signup button on the homepage. The user will enter their credentials and create an account for them to use.
    - handleClick() determines whether the credentials entered by the user are valid and makes the account accordingly. If not, then we send an error message from the backend.

- **Welcome.js**
  - Default Home page. Allows a user to login or create a new account
    - Welcome() returns the home screen.

Components:

- **Card.js**
  - This component is used to display information (title, description) in the welcome page for new users.
- **EditCategoryForm.js**
  - Main form utilized for both adding or deleting information in relation to a course's category.
- **EditCourseForm.js**
  - Main form utilized for both adding or deleting information in relation to a semester's course.
- **FormAddSemesterDialog.js / FormEditDeleteSemesterDialog.js**
  - Legacy code for adding/editing/deleting a semester.
- **FormCategoryDialog.js**
  - Wrapper for EditCategoryForm.js.
- **FormCourseDialog.js**
  - Wrapper for EditCourseForm.js
- **Navbar.js**
  - Top navigation bar present throughout the entire website.