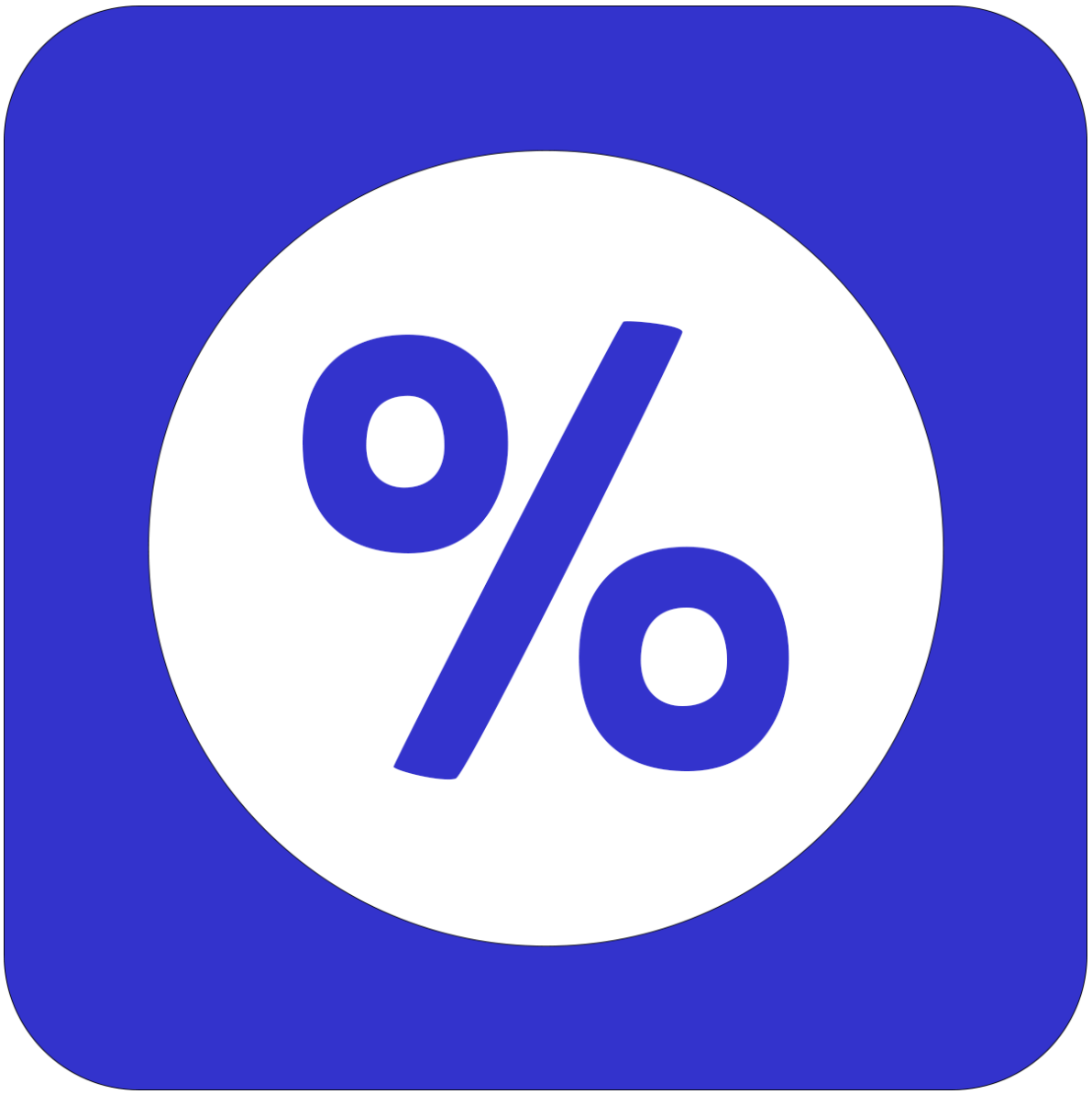
Promedio Documentation



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/user/:userId/semester/**
  - **GET**
    - This is used in the Promedio/frontend/src/routes/Home.js to display all semesters mapped to an individual user (via “userId”).
    - Request Parameters: { userId: string }
    - 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 (via “userId”). The semester name is passed via the request body.
    - Request Parameters: { userId: string }
    - Response: { message: “Semester added successfully.”, semester: { name: string, userId: Mongoose.Schema.Types.ObjectId }}

- **/api/user/:userId/semester/:semesterName**
  - **GET**
    - This is used in the Promedio/frontend/src/routes/Home.js to list the details of a specific semester (currently identified with “semesterName” → plan is to either change this to be semesterId or to enforce unique semester names in schema).
    - Request Parameters: { userId: string, 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 name “semesterName” 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: { userId: string, semesterName: string }
    - Request Body: { name: 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 name “semesterName” within the “Semester” Model/“semesters” collection) that is mapped to “userId”. Nothing happens otherwise within the database if no such semester exists.
    - Request Parameters: { userId: string, semesterName: string }
    - Request Response: { message: “Semester deleted successfully.” }

- **/api/user/:userId/semester/:semesterName/course**
  - **GET**
    - This is used in the Promedio/frontend/src/routes/Home.js to display all courses listed within a specific semester (currently “semesterName” → plan is to either change this to be semesterId or to enforce unique semester names in schema).
    - Request Parameters: { userId: string, semesterName: string }
    - 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 (identified via “semesterName” parameter → plan is to either change this to be semesterId or to enforce unique semester names in schema).
    - Request Parameters: { userId: string, semesterName: string }
    - Request Response: { message: “Course added successfully.”, course: { name: string, semesterId: Mongoose.Schema.Types.ObjectId }}
  - **DELETE**
    - This is used in the Promedio/frontend/src/routes/Home.js to delete all courses mapped to a specific semester (identified via the “semesterName” parameter →

plan is to either change this to be semesterId or to enforce unique semester names in schema).
        - Request Parameters: { userId: string, semesterName: string }
        - Request Response: [{ message: “Successfully deleted [x] courses.”}]

- **/api/user/:userId/semester/:semesterName/course/:courseName**
    - **GET**
        - This is used in the Promedio/frontend/src/routes/Home.js to list the details of a specific course. 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: { userId: string, semesterName: string, courseName: string }
        - Request Response: { name: string, semesterId: Mongoose.Schema.Types.ObjectId }
    - **PATCH**
        - This is used in the Promedio/frontend/src/routes/Home.js to update an existing course (i.e. a current document with name “courseName” within the “Course” Model/“courses” 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 course exists.
        - Request Parameters: { userId: string, semesterName: string, courseName: string }
        - Request Body: { userId: 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 name “courseName” within the “Course” Model/“courses” collection) that is mapped to “userId”. Nothing happens otherwise within the database if no such course exists.
        - Request Parameters: { userId: string, semesterName: string, courseName: string }
        - Request Response: { message: “Course deleted successfully.” }

Frontend

Routes:

- **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.

- **Home.js**
    - The page that shows once a user is logged in.
        - 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.

- **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.