Assignment 2 - Team 9 Back-End Subteam Project Report

Summary of Decisions

Our user story focuses on the account creation process of multiple types of users, and the ability of users to perform actions specific to their type. To tackle this user story, we first decided on the types of users that will use this application. Our partner wanted this application to be educational, and as such we decided on a teacher and student user type. As we do not want this application to be restricted to only those that are part of an education environment, we also decided to have a general user type. The general user will have less fields to fill in when signing up, while teachers and students have specific fields that help us tackle the second part of the user story, which is the ability to group students and teachers together. Teachers will have the ability to create a homeroom that will be used to monitor student's responses and scores, and students must enter a homeroom ID during the account creation process to be entered into a homeroom. This will allow teachers to see student responses and reactions from using this application, and allow this application to be assigned as homework if needed.

Next, we decided on the framework we were going to use to complete this user story. We decided to use the Django REST framework because we communicated with our database team and decided that this framework works efficiently with the database framework they chose, and we are also familiar with this framework and believe the functionality of it can help us complete this user story. Then, we started dividing this process into small components. One of the components we have is models.py, which details the fields required for user creations. At first, we had three individual models for our three user types. However, to prevent redundant code, we were able to identify similar fields and require certain fields only for specific user types. For example, a student will be required to enter their school, grade, and a hoomroom_id that their teacher provided, while a regular user of this app will not be required to enter that information at all. In addition, we implemented url patterns, views, and serializers that use APIs to determine if the fields are entered correctly and then create the users. As with models, we made sure to prevent redundant coding as much as possible. Apart from that, we also implemented logic that allowed students and teachers to be grouped by a homeroom_id that was mentioned before. The purpose of this is for educators to have the ability to see the responses and scores of students when they go through this app. This was a key feature of the user story we chose as our partner wanted this app to emulate an in-person workshop as much as possible. To test our code and ensure that it is functional, we used Postman, an API testing platform, to have mock data run through our code to check that our code is done correctly. Overall, we were able to target different types of users using code that is efficient and non-redundant, and implement back-end logic for creating homerooms that allowed grouping of teachers and students and for teachers to keep track of student progress.

Individual Contributions

Varun:

I helped make the accounts app by helping setting up the models, admin, and some parts of serializers. I also created the app homeroom and all the models, views, serializers and admin functionalities to join, create, leave and end homerooms.

YoungJun:

I initialized the settings for the project, built serializers for different types of users, implemented token authentication for login, and deployed the project on railway.app.

Nigel:

I contributed to accounts by helping out with models, serializers, urls, and views. I also wrote all the text related to this subteam in the main report and this one.

Instructions For Testing

Video Instruction

https://youtu.be/bnBjZvkq_oE

Written Instruction

1. Setting up

For testing, we deployed on railway.app and will be using Postman to access it.

First, go to our github page:

https://github.com/csc301-2023-winter/assignment-2-9-1-choyou36-pillaiv3-yangnige

Locate the Postman Endpoints Json file at “pathway”.

Next, go to:

https://www.postman.com/

Sign up/log in, then go to Workspaces (top left), and create a new workspace.

Click the Import button (top left) and import the file that was downloaded.

Under Collections, there will be a list of Endpoints that can be used to test our code

2. Testing

First, use any of the Registers to create a user/student/teacher. The API documentation can be helpful here and can be found at “pathway”.

After clicking on an Endpoint, for example Register default user, click Body. There should already be pre-existing fields that are filled out. The Keys are required parameters, and the values can be freely changed for testing purposes. Pressing Send will result in a POST request that stores the data that was just sent.

Next, head to Login and login with the credentials of the user that was just created. After entering the required fields and pressing Send, authorization tokens will appear at the bottom of the screen. Please copy the access token without the quotation marks. This token will be used to verify whether the user is a teacher or student. This is because some of the homeroom functions will be specific to the user type.

Now, head to either Create or End Homeroom if the user that is logged in is a teacher, or Join/Leave Homeroom if the user is a student. Click on authorization and paste the access token where it says Token. Test by using Body for all these homeroom endpoints.

3. Sample Example

Refer to video for detailed example and overall flow