Senior Design

ENG EC 464

Memo

To:	Professor Pisano
From:	Sharith Godamanna, Camden Kronhaus, Quinn Meurer, Alexander Trinh
Team:	11
Date:	2/26/21
Subject:	2nd Prototype Test Report

1.0 Equipment and Setup

The Stod application architecture is split into two parts: the backend and frontend, which communicate through HTTP requests. This allows for modularity and can easily be migrated to a microservices architecture. To start with a backend template, we create a Python virtual environment and install Django within it. To create a new Django application we run `django-admin startproject stod-backend` which creates an empty django application. For the frontend, we create a new React app running `yarn create stod-frontend --typescript` specifying a typescript template. We decided to use typescript to be able to use type safe variables which makes debugging much simpler. To handle global React state, we are using Redux which manages a global store where we can store and access global variables. To connect our frontend to the backend, we created asynchronous redux actions and used the axios library to make HTTP requests to our server.

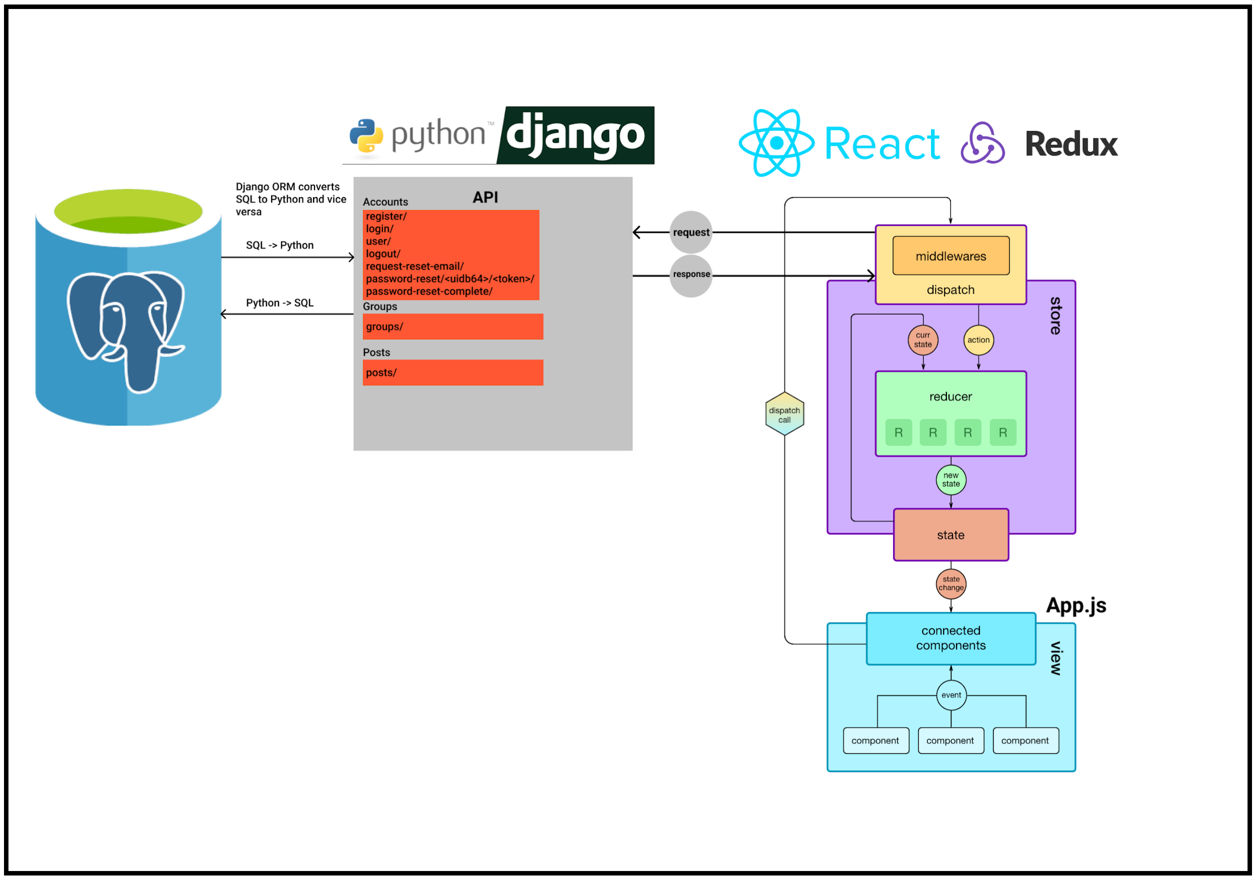

Figure 1: Stod application architecture

2.0 Test Report

2.1 Measurements Taken

Our measurements are scored on an all or nothing basis for each of the tests we did. If the test passed, it scored a one; if the test wasn't completely successful, it received a 0. The final score was tallied out of the 18 total tests. Each of our tests examines the key features that were implemented for the prototype to ensure they all work properly.

Test Parameters	**Score /1**
Register a new user	1
Login with newly created user account	1
Navigate home and see all subscribed groups posts	1
Navigate to a subscribed group and view all posts within that group	1
Confirm all posts in the back end database are visible on the home page	1
Post creation reflected immediately (front and back end)	1
Post edits are reflected immediately (front and back end)	1

Post deletion is reflected immediately (front and back end)	1
Posts can only be edited and deleted by user who created them (front end)	1
Groups: View the list of groups on the frontend. The name and description for each group should be visible.	1
Groups: No unsubscribed groups show on the sidebar.	1
Groups: Subscribing to a new group shows up on the sidebar and updates in the backend.	1
Comments: Able to comment under a single previously created post	1
Comments: Able Comment multiple times on a single post	1
Comments: Able to view all comments for any post by any user	1
Direct messaging: Send messages to another user	1
Direct messaging: Messages show up on correct side of the screen based on sender	1
Direct Messaging: User can select other user to message	0
Final Score:	**17/18**

2.2 Conclusions

The main focus of this 2nd prototype was finishing the basic functionality of our project and to effectively tie these individual components together to make a fully functional product. We had near perfect testing, only experiencing one consistency issue with allowing a user to access other users to message. Our final score was 17/18. We succeeded in building the frontend and backend functionalities for user authentication and authorization, groups, and posts, and comments. Professor Pisano and Joseph Greene gave us feedback to try and focus on the sensitivity aspect of our application. They highlighted that our users will most likely be in a sensitive place and therefore more prone to harmful comments and content. The tasks in our next sprint work towards the goal of refining the frontend to provide a more user-friendly experience, as well as starting to develop a form of content moderation. We think that if we stay on track with these sprint goals we will be able to deploy our application so that users in the class will be able to test it before customer installation, giving us valuable user feedback to further improve.