Boston University
Electrical & Computer Engineering
EC464 Senior Design Project

Final Project Testing Plan

Stöd

By

Team 11
Stöd

Team Members:
Sharith Godamanna | sharithg@bu.edu
Camden Kronhaus | kronhaus@bu.edu
Quinn Meurer | quinnyyy@bu.edu
Alexander Trinh | aktrinh@bu.edu

Required Materials (Technologies)

Frontend
- ReactJS
- Redux
- TypeScript
- Static Site Generator (Netlify)

Backend
- Django
- Django Rest Framework
- Python3
- Docker
- Node.js
- MongoDB
- Public Digital Ocean Server

Setup

The Stod application is built using a microservice architecture currently in three parts: a chat server, a general backend server, and a frontend which communicates with both backend components using HTTPS requests. Both the chat server and the general backend server run on a public Ubuntu server tied to https://stodbackend.app and communicate locally with Nginx to allow for secure communication from requests coming from users' browsers interacting with the front end. The chat server runs using MongoDB and Node.js while the general backend server runs using Django and PostgreSQL. Both are started by running their respective docker builds to allow for a consistent startup process. The front end is statically generated and hosted automatically with pushes to a production branch in Github using Netlify. The front end can be reached by going to https://www.stod.app. In the front end, we are using typescript in order to use type safe variables and for easier debugging, as well as Redux to manage global state. To connect our frontend to the backend, we created asynchronous redux actions and used the axios library to make HTTPS requests to our backend services.

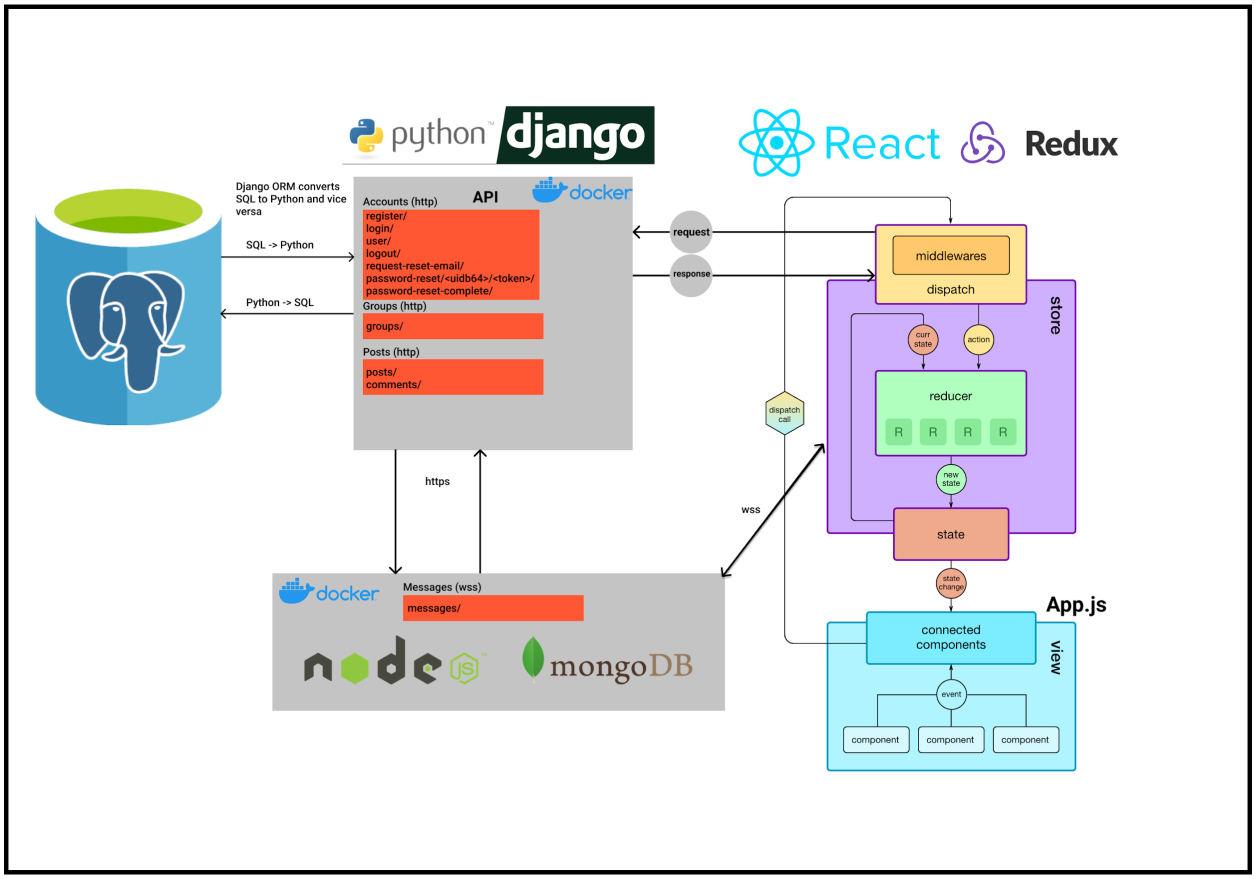

Figure 1: Stod application architecture

Pre-testing Setup Procedure:

Before testing, we will set up dummy data to mimic what a user would normally see when pulling up our web app. This includes:

- Creating various “dummy” posts
- Creating some groups to subscribe to
- Creating various tags that can be used when creating a post
- Creating admin roles for moderators (ourselves)
- Creating “dummy” comments under various posts within the database

Testing Procedure:

Load App:

1. Navigate to https://www.stod.app in a web browser

Login:

1. Register a new user with:
    a. Username: testuser99
    b. Email: testuser99@bu.edu
    c. Password: SecurePW@54367
2. Login with newly created user account
    a. Username: testuser99
    b. Password: SecurePW@54367
3. Attempt to login with a user that does not exist:
4. Error checking security
    a. Register errors:
        - Password too common
        - Password not long enough

Posts:

1. Navigate to home page and view all posts loaded on the home page
2. Navigate to a subscribed group to view all the posts within that group
3. Create a new post from within a subscribed group, specifying a title, body, and tags
4. Edit a post created by the current user, and view updates
5. Delete post created by a user, and view updates
6. Filter the displayed posts by selecting specific tags

Flagging:

1. Create a post that contains a bad word in the title
2. Create a post that contains a bad word in the body
3. View the two flagged posts as an admin
4. Approve one of the flagged posts and attempt to view it in the main group page
5. Deny the other post and attempt to view it in the main group page

Commenting:

1. Comment a single comment under each post within a group
2. Choose a post and add multiple comments
3. View each comment under their respective posts
4. Reply to a comment

Friends:

1. Open incognito tab and login as a second user
2. Click on the second users profile, and send a friend request
3. Accept friend request on User 1’s profile
4. Create a third user, send a friend request, and reject the friend request.When the friend is on the friends list, click on them and start messaging

Messaging:

1. User 1 sends user 2 a message
2. User 2 sends user 1 a message

Groups:

1. Navigate to home page and click “Find Groups” to view all existing groups
2. Click “subscribe” on a group
3. Navigate back to the home page see list of groups in the sidebar
4. Click on a subscribed group in the sidebar and view all the posts to that group

7. Unsubscribe from a group

<u>Measurable Criteria</u>

Load App:

1. Login page loads successfully with HTTPS

Login:

1. A new user can be created and logged in as that user at any point
2. A new user is unable to register if there are errors at login

Posts:

1. Navigation to the home page results in all posts shown to the user and the groups they were created in
2. Navigation to a subscribed groups results in only posts for that group being displayed
3. Creation of a post is reflected accurately in both the front and back end after submission
    a. Creation of a post also accurately shows date of creation and poster username
4. Editing and Deleting a post is available to the user who created them
5. Edits to a post are reflected accurately in both the front and back end after submission
6. Deletion of a post is reflected accurately in both the front and back end after submission
7. Filtering mechanism accurately filters so only posts with certain tags are shown

Flagging:

1. Creating a post that contains a bad word should alert the user and not show up for current or other users on the app
2. A post with a bad word should be sent to an admin page for approval/denial
3. A post that has been approved should be removed from the admin page and be visible to users within the group
4. A post that has been denied should be deleted from the database
5. A post without a bad word should be created normally

Commenting:

1. Able to create a comment under a previously created post
2. Comments should be updated and visible automatically without refresh
3. The comments should be connected to the id of the post
4. Only comments pertaining to a certain post should be loaded under that post
5. All previously created comments under a post should always load under that post
6. A reply should be loaded under its respective post like a new comment and reference the original comment

Friends:

1. Friend requests are shown to a user
2. Accepting friend request immediately adds them to the friends list on the sidebar
3. Rejecting friend requests must not show on the sidebar.

Messaging:

1. Starting a conversation initiates one to one connection between two users
2. Sending a message gets broadcasted immediately to recipient
3. User 1's message shows on the right side and user 2's messages show on the left side.

Groups:

1. All available groups are displayed to the user after clicking "Find Groups"
2. User is able to subscribe to new groups
3. Subscribed groups are displayed in sidebar
4. Upon clicking on a subscribed group in the sidebar, user can view posts from that group
5. Group is remove from list of subscribed groups upon user's unsubscription