CS5500 WeGroup Document

This factor evaluates the external project documentation. This includes project organization, requirements, features, functionality interfaces or APIs, design, communications plans, and test plans

Project organization:

We organize the whole project in four folders. They are called backend, frontend, docs, and readme. They contain different files inside which play different roles in the entire project. Here are the basic introductions of all four folders.

backend:
Basically, this folder contains all the technical related files of backend design.
We have a subfolder called “models”, which contains all the data schemas we designed as the database.
We have a subfolder called “routes”, which contains all the API files to provide the functionality.
We have a subfolder called “middleware”. It contains the authentication related files.
We also have the server related files contained in this folder.

frontend:
Basically, this folder contains all the technical related files of frontend design.
We have a subfolder called “public”, which contains all the HTML entry files.
We have a subfolder called “styles”, which contains all the css file.
We have a subfolder called “src”, which contains all the course code, for example, the components and services.
We have a subfolder called “service”, which helps to connect the frontend with the backend.

docs:
Basically, this folder contains all the technical related files for everyone to read.
“The communication plan” is the file to record our meetings about the project design.
“CS5500 WeGroup Document” is the file which includes all the introductions to help the professor and teaching assistants evaluate our project.

readme:
This is the file we provide for other developers.
In this file, we have the instructions for other developers to install the servers, run the build, and test all the functions.

Project requirements:

A full-stack web application to let users create groups and collaborate with each other to share messages(string, photos, files, emoji), calendar.
FrontEnd: React, Figma
BackEnd: Node.js, Postman for testing
Database: MongoDB

Project features:

We have several features:

- Sign up/ Log in
- Post
- Search
- Team

Project API:

User:

- createUser()
- updateUser()
- getUser()
- deleteUser()

Post:

- createPost()
- updatePost()
- getPostCreatedbyUser()
- deletePost()
- getPostsbyTag()
- getPostsInteratedByUser()
- getTeamSize()

Team:

- createTeam()
- deleteTeam()
- getInterestedUserList()
- getTeam()

Tag:

- createTag()
- searchTag()
- getAllTags()
- getTagByTagId()

Comment:

- createComment()
- deleteComment()
- editComment()
- deleteComment()

Project Design:

Frontend Design: FrondEnd Design
Backend Design: BackEnd Design

Project communication plans:

communication plans

Project test plans:

We used Postman to test our backend API and log error message in the chrome inspect console to test the frontend.
FrontEnd:

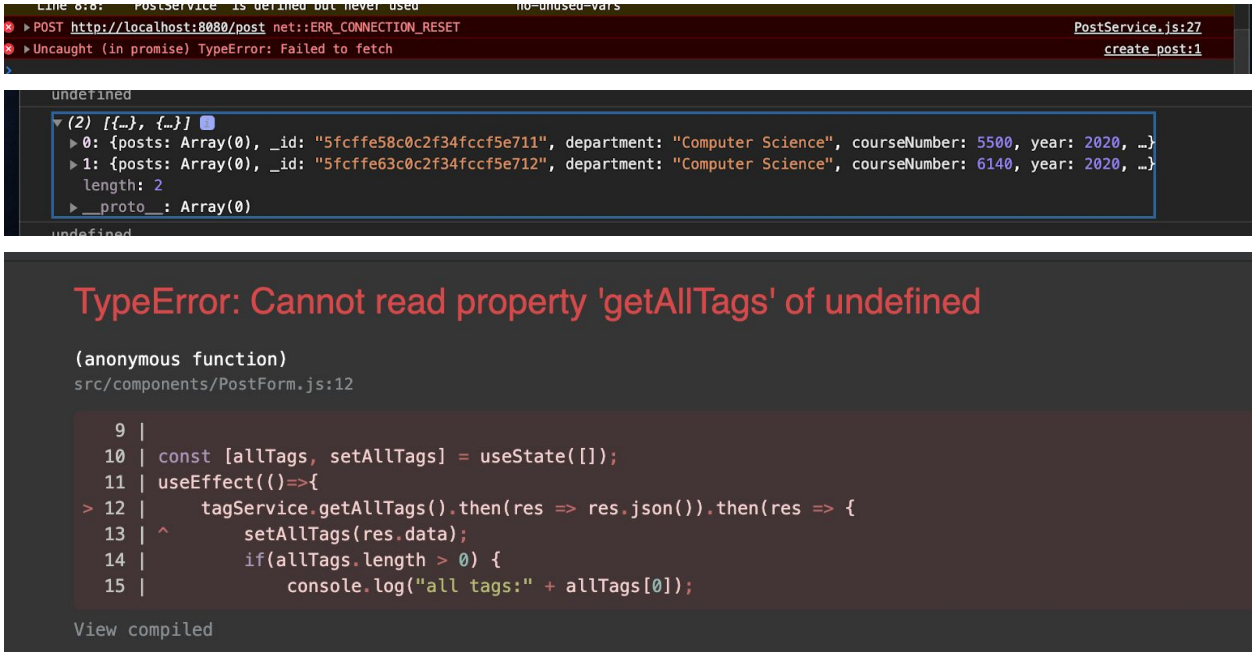
1. create each component with empty function, render the components using chrome to check component render as expected

2. add fake data for components to render, using chrome to check component render data as expected

3. add functions and logics to buttons and links and using chrome to test components behave as expected. using console log for testing and debugging and using chrome inspect to see the console logs.

4. connect components to the backend, render the component using fetched data. Check in Chrome to verify the react application interacts with the backend as designed.

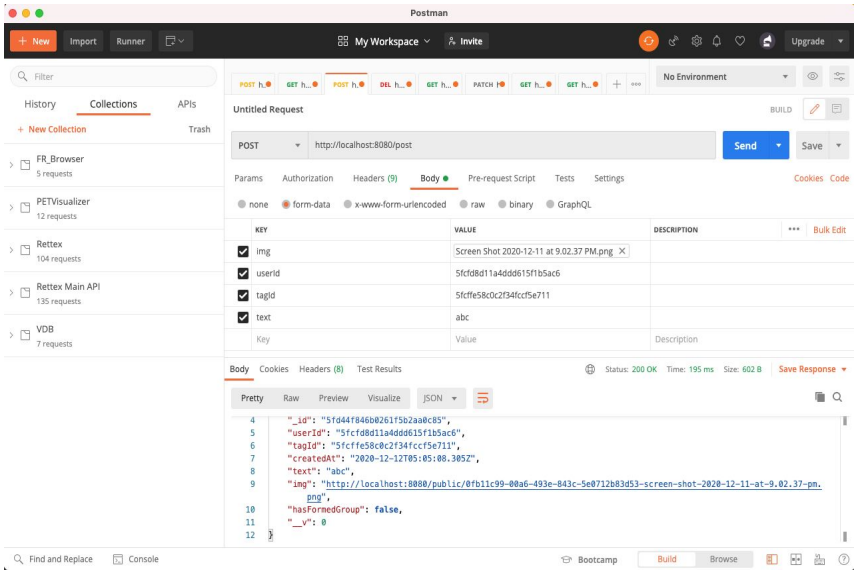
Following screenshot are some of the errors that we received.



BackEnd:

1. Connect the specific url from the Postman
2. Add fake data as JSON format and then send request to local host to check if backEnd API return the data as expected

Following screenshot is showing what we get from the backend API.


Postman
New
Import
Runner
My Workspace
Invite
Upgrade
Filter
History
Collections
APIs
New Collection
Trash
FR_Browser
5 requests
PETVisualizer
12 requests
Rettex
104 requests
Rettex Main API
135 requests
VDB
7 requests
No Environment
Untitled Request
BUILD
POST
http://localhost:8080/post
Send
Save
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
Code
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
KEY
VALUE
DESCRIPTION
Bulk Edit
img
Screen Shot 2020-12-11 at 9.02.37 PM.png
userId
5fcfd8d11a4ddd615f1b5ac6
tagId
5fcffe58c0c2f34fccf5e711
text
abc
Key
Value
Description
Body
Cookies
Headers (8)
Test Results
Status: 200 OK
Time: 195 ms
Size: 602 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
"_id": "5fd44f846b0261f5b2aa0c85",
"userId": "5fcfd8d11a4ddd615f1b5ac6",
"tagId": "5fcffe58c0c2f34fccf5e711",
"createdAt": "2020-12-12T05:05:08.305Z",
"text": "abc",
"img": "http://localhost:8080/public/0fb11c99-00a6-493e-843c-5e0712b83d53-screen-shot-2020-12-11-at-9.02.37-pm.png",
"hasFormedGroup": false,
"__v": 0
}
Find and Replace
Console
Bootcamp
Build
Browse