All Knights Study

John C. Hitt Library Group Finder

Technology used:

GitHub

heroku

mongoDB

Tailwind CSS

React

Flutter

Trello

JWT

Team members and the part they played

Jacob Mayer	PM / Front End - Website
Jacob Graff	Front End - Mobile
Chris Parisapogu	DB
Bryan Aneyro Hernandez	API
Chase Hanson	Front End - Website
Delali Ekpeh	Front End - Mobile
John Vea	Front End - Website

Project Manager

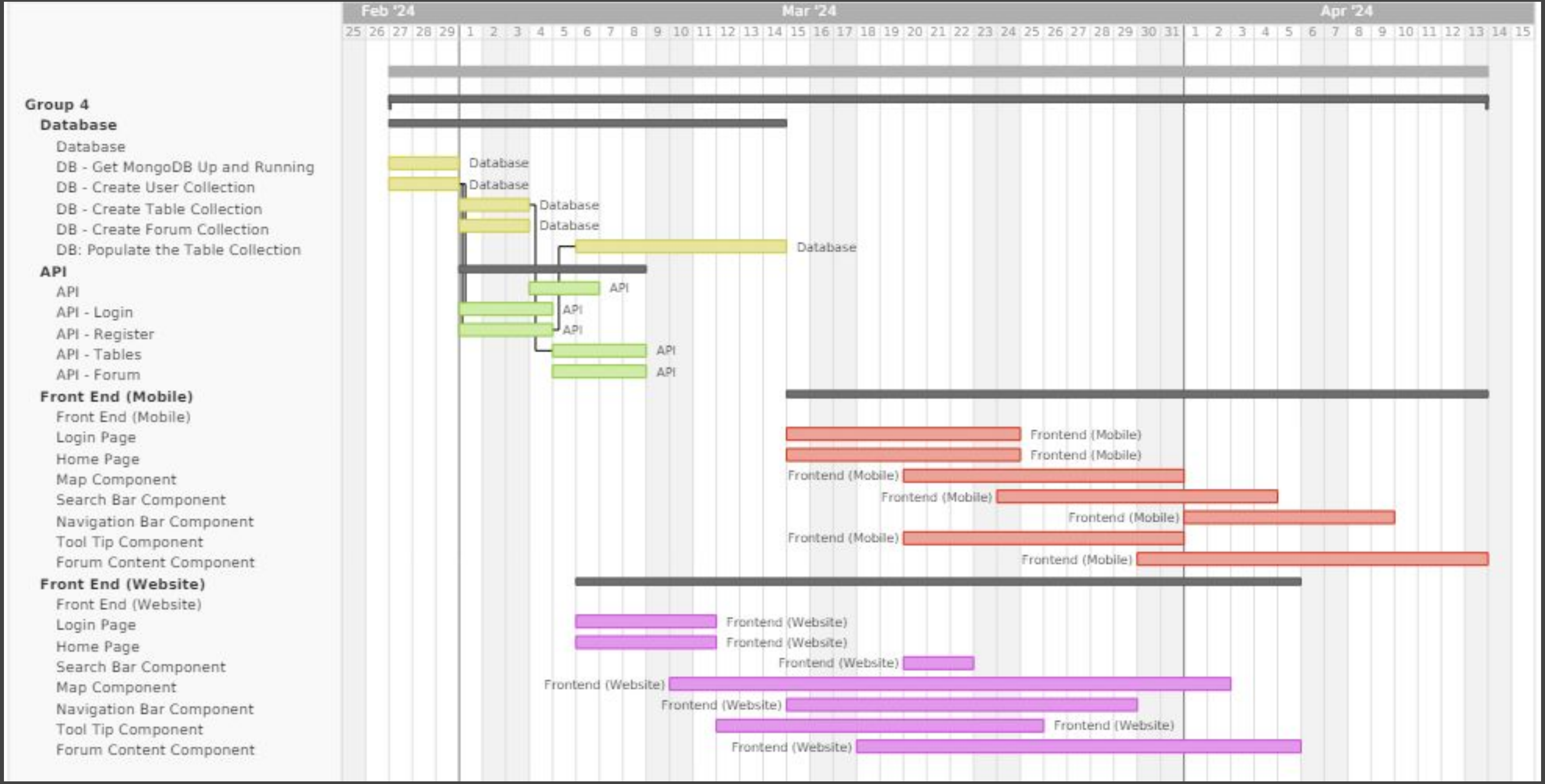
Roles:

- Organization and communication with the team
- Assist with initial planning of app
- Help in the areas where help was needed
- Keep progress on track
- Build presentation

Takeaways:

- Leadership
- ReactJS
- Heroku hosting
- Git management

Gantt chart


Feb '24
Mar '24
Apr '24
25 26 27 28 29 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Group 4
Database
Database
DB - Get MongoDB Up and Running
DB - Create User Collection
DB - Create Table Collection
DB - Create Forum Collection
DB: Populate the Table Collection
API
API
API - Login
API - Register
API - Tables
API - Forum
Front End (Mobile)
Front End (Mobile)
Login Page
Home Page
Map Component
Search Bar Component
Navigation Bar Component
Tool Tip Component
Forum Content Component
Front End (Website)
Front End (Website)
Login Page
Home Page
Search Bar Component
Map Component
Navigation Bar Component
Tool Tip Component
Forum Content Component
Database
API
Frontend (Mobile)
Frontend (Website)

Database

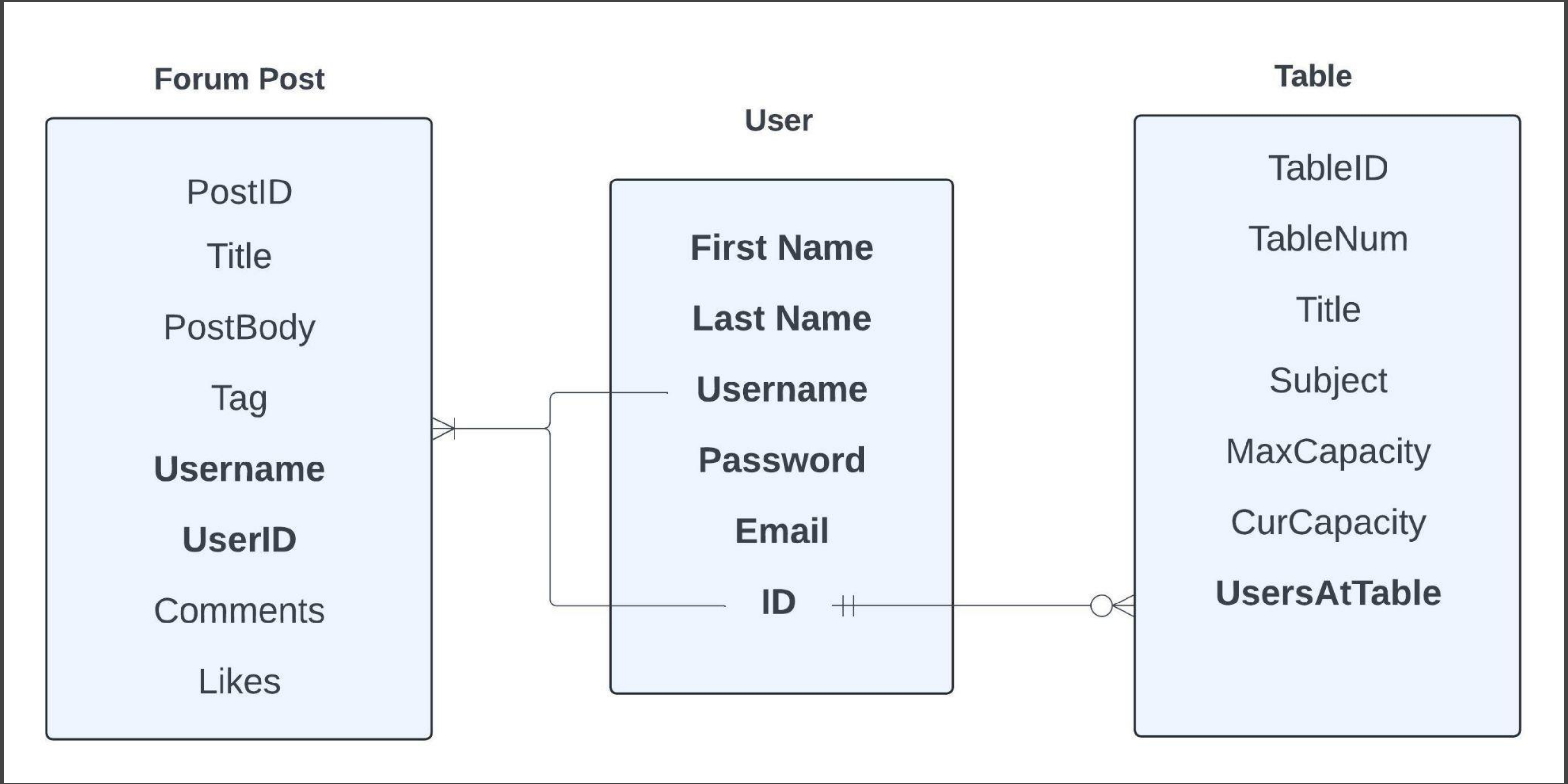
Roles:

- Create maintain the database backend
- Determined the best data representation
- Assisted API creation
- Helped curate the ERD chart accurately

Takeaways:

- MongoDB
- Mongoose
- JSON data structures

ERD


Forum Post
PostID
Title
PostBody
Tag
Username
UserID
Comments
Likes
User
First Name
Last Name
Username
Password
Email
ID
Table
TableID
TableNum
Title
Subject
MaxCapacity
CurCapacity
UsersAtTable

API

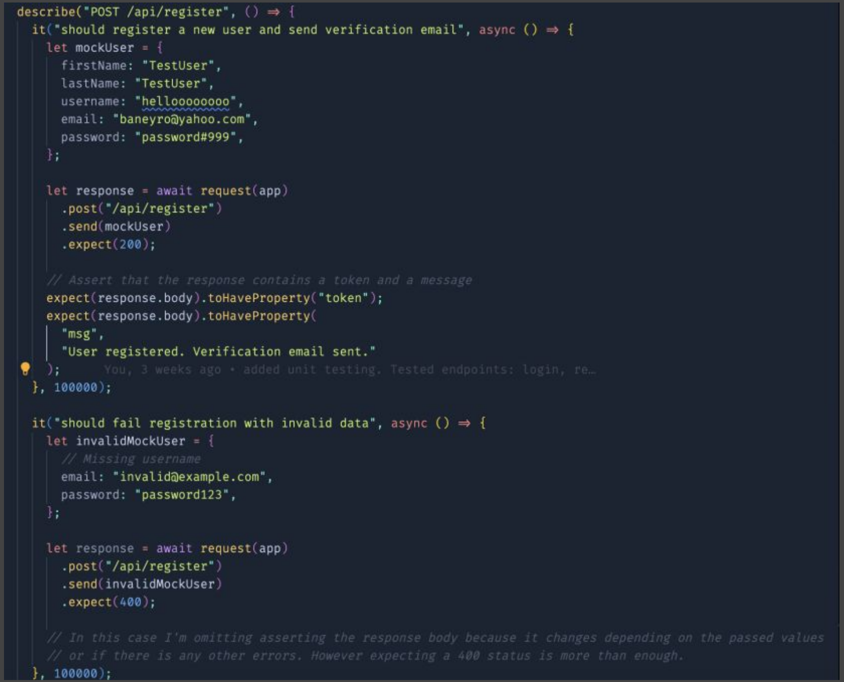
Roles:

- Create endpoints to assist front-end tasks
- Create various CRUD operation endpoints
    - Table, Forum, Account
- Created API documentation to assist front-end
- Tested APIs

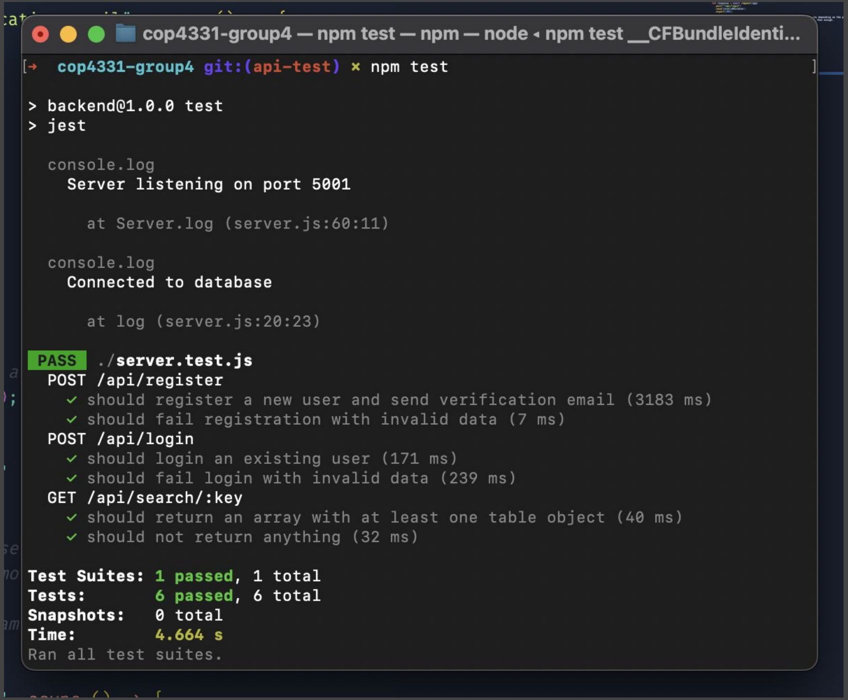
Takeaways:

- JavaScript
- SwaggerHub/Postman
- Form verification and verification schema

Unit Testing - API



```
describe("POST /api/register", () => {
  it("should register a new user and send verification email", async () => {
    let mockUser = {
      firstName: "TestUser",
      lastName: "TestUser",
      username: "helloooooooo",
      email: "baneyro@yahoo.com",
      password: "password#999",
    };

    let response = await request(app)
      .post("/api/register")
      .send(mockUser)
      .expect(200);

    // Assert that the response contains a token and a message
    expect(response.body).toHaveProperty("token");
    expect(response.body).toHaveProperty(
      "msg",
      "User registered. Verification email sent."
    );    You, 3 weeks ago • added unit testing. Tested endpoints: login, re…
  }, 100000);

  it("should fail registration with invalid data", async () => {
    let invalidMockUser = {
      // Missing username
      email: "invalid@example.com",
      password: "password123",
    };

    let response = await request(app)
      .post("/api/register")
      .send(invalidMockUser)
      .expect(400);

    // In this case I'm omitting asserting the response body because it changes depending on the passed values
    // or if there is any other errors. However expecting a 400 status is more than enough.
  }, 100000);
```



```
cop4331-group4 — npm test — npm — node ◂ npm test __CFBundleIdenti...

→ cop4331-group4 git:(api-test) ✗ npm test

> backend@1.0.0 test
> jest

  console.log
    Server listening on port 5001

      at Server.log (server.js:60:11)

  console.log
    Connected to database

      at log (server.js:20:23)

PASS  ./server.test.js
  POST /api/register
    ✓ should register a new user and send verification email (3183 ms)
    ✓ should fail registration with invalid data (7 ms)
  POST /api/login
    ✓ should login an existing user (171 ms)
    ✓ should fail login with invalid data (239 ms)
  GET /api/search/:key
    ✓ should return an array with at least one table object (40 ms)
    ✓ should not return anything (32 ms)

Test Suites: 1 passed, 1 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        4.664 s
Ran all test suites.
```

API demonstration

https://app.swaggerhub.com/organizations/BR867228

Front End - Website

Roles:

- Connected API endpoints to the functional front end for web access
- Established a user friendly environment
- Established general styling for the web app
- Ensure accessibility for all users

Takeaways:

- JavaScript and ReactJS
- Tailwind CSS
- Usefulness of packages

Front End - Mobile

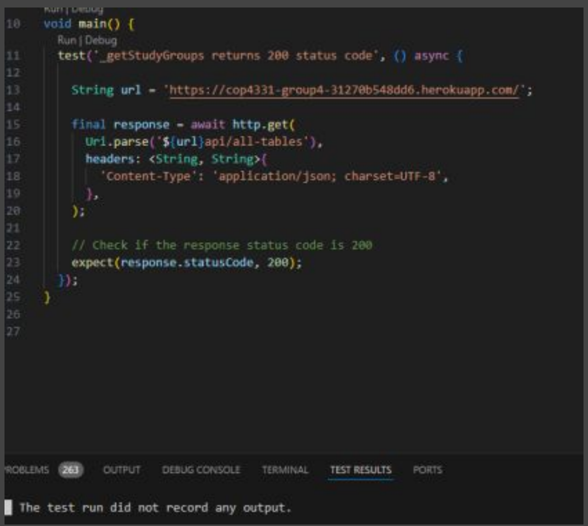
Roles:

- Connected API endpoints to the functional front end for mobile access
- Added functionality for mobile accessibility of app functions
  - Pinch and zoom
  - Click and drag
- Ensured production application on the device worked properly

Takeaways:

- Dart and Flutter
- Usefulness of packages

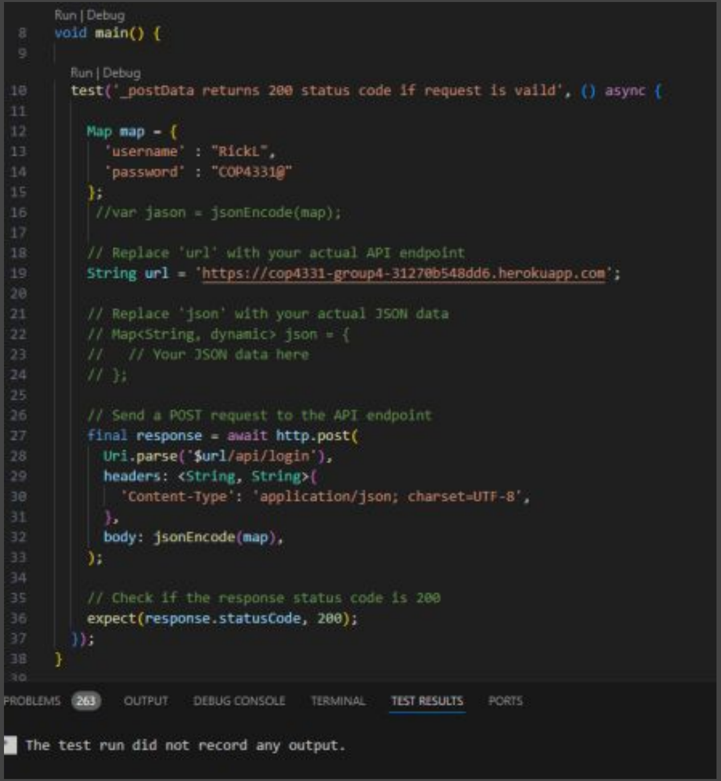
Unit Testing - Mobile



```
Run | Debug
void main() {
  Run | Debug
  test('_getStudyGroups returns 200 status code', () async {

    String url = 'https://cop4331-group4-31270b548dd6.herokuapp.com/';

    final response = await http.get(
      Uri.parse('${url}api/all-tables'),
      headers: <String, String>{
        'Content-Type': 'application/json; charset=UTF-8',
      },
    );

    // Check if the response status code is 200
    expect(response.statusCode, 200);
  });
}
```

PROBLEMS 263 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```
The test run did not record any output.
```



```
Run | Debug
void main() {

  Run | Debug
  test('_postData returns 200 status code if request is vaild', () async {

    Map map = {
      'username' : "RickL",
      'password' : "COP4331@"
    };
    //var jason = jsonEncode(map);

    // Replace 'url' with your actual API endpoint
    String url = 'https://cop4331-group4-31270b548dd6.herokuapp.com';

    // Replace 'json' with your actual JSON data
    // Map<String, dynamic> json = {
    //   // Your JSON data here
    // };

    // Send a POST request to the API endpoint
    final response = await http.post(
      Uri.parse('$url/api/login'),
      headers: <String, String>{
        'Content-Type': 'application/json; charset=UTF-8',
      },
      body: jsonEncode(map),
    );

    // Check if the response status code is 200
    expect(response.statusCode, 200);
  });
}
```

PROBLEMS 263 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```
The test run did not record any output.
```

Overview

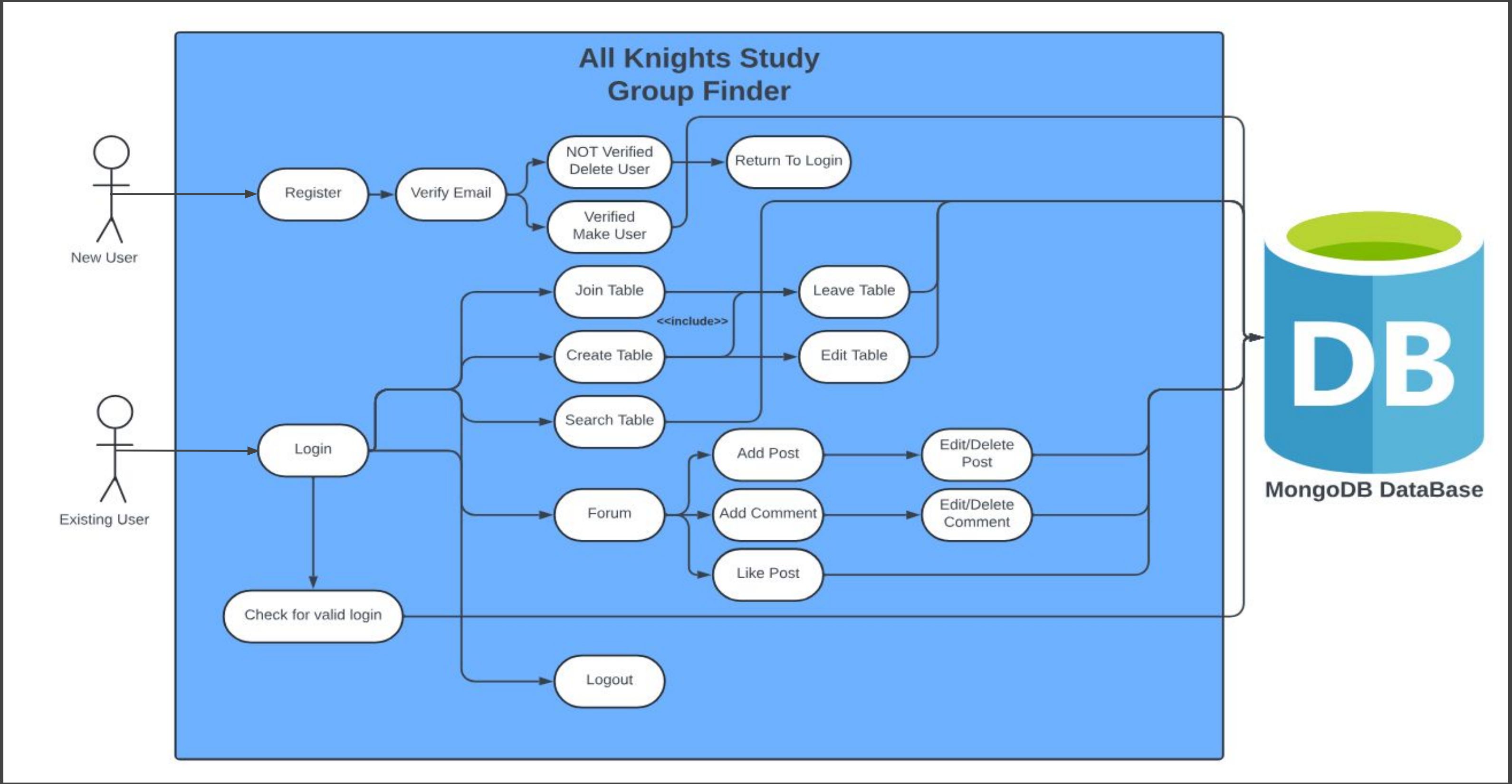
What went well:

- Communication and teamwork
- Database and API
- Learning ReactJS/Flutter
- Forum Functionality

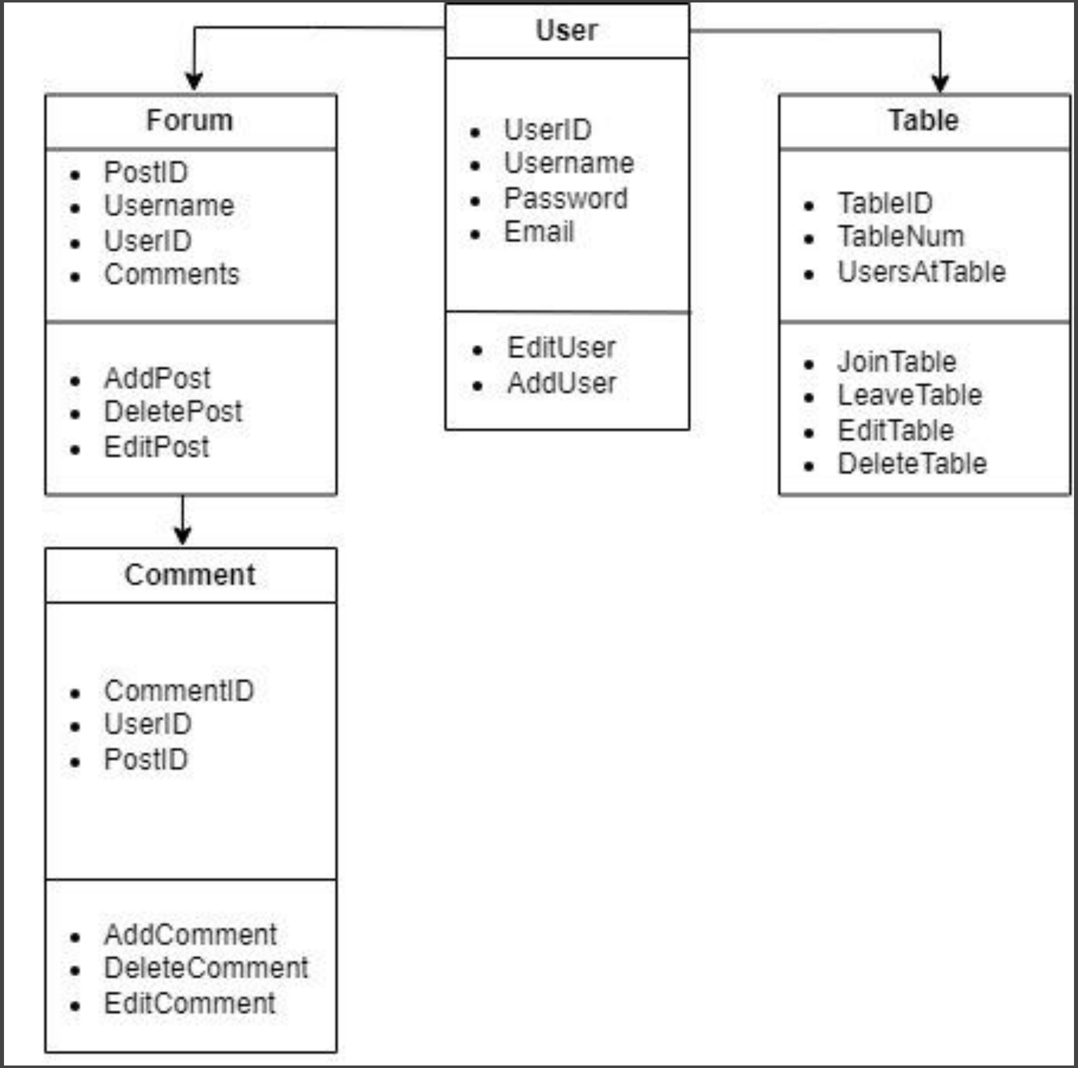
What didn't go well:

- NavBar
- Debugging of Map Positions
- Login/Register Page Design

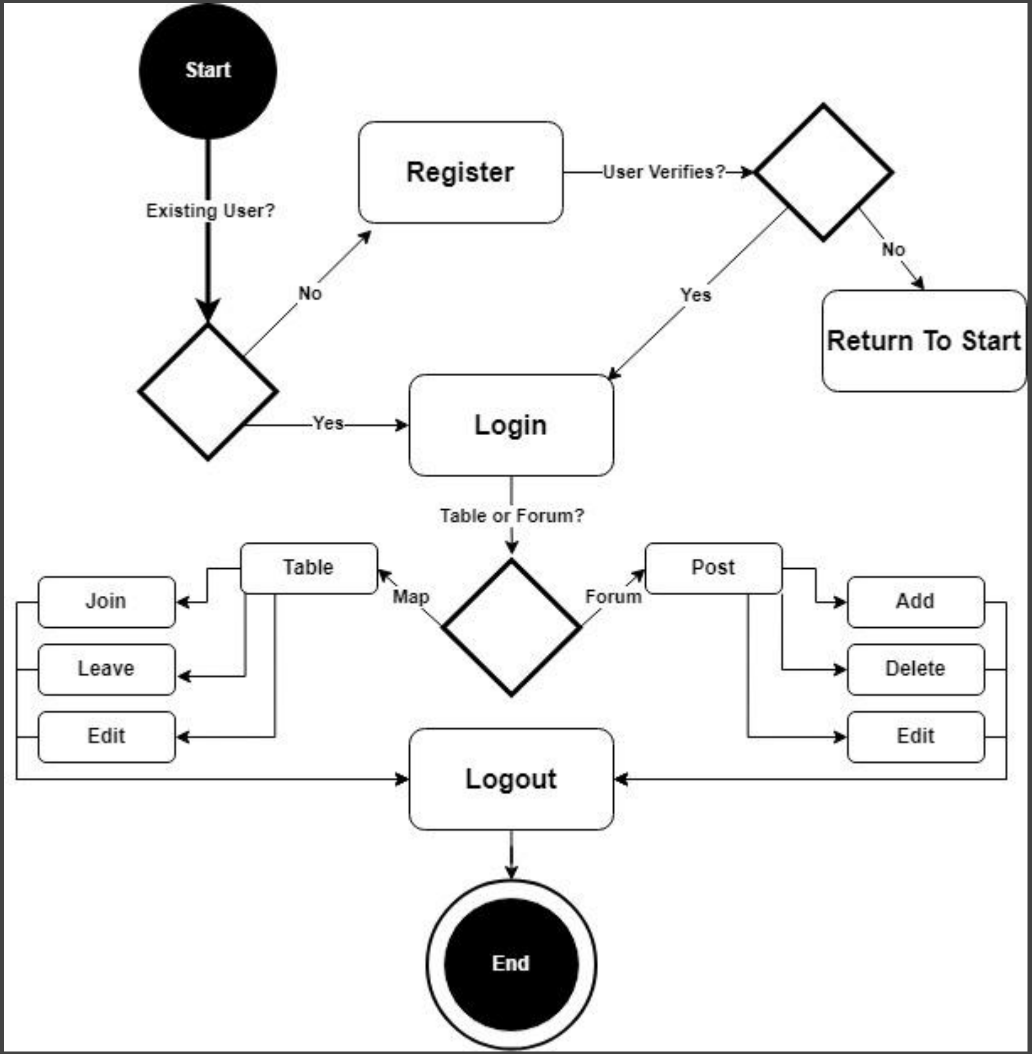
Use Case


All Knights Study Group Finder
New User
Existing User
Register
Verify Email
NOT Verified Delete User
Verified Make User
Return To Login
Join Table
Create Table
Search Table
<<include>>
Leave Table
Edit Table
Login
Forum
Add Post
Add Comment
Like Post
Edit/Delete Post
Edit/Delete Comment
Check for valid login
Logout
DB
MongoDB DataBase

Class Diagram



Activity Diagram



App demonstration

https://cop4331-group4-31270b548dd6.herokuapp.com/

Questions?