My proposed website is called Dormroom. Dormroom is a one-stop, convenient and engaging platform that connects project initiators with their ideal collaborators. Project initiators can post on the website stating what they are looking for, location, vacancy, qualifications and descriptions. Other users can comment on the post.

Planning

project includes a wireframe for each view Link to figma: https://www.figma.com/file/P0qPB2CVsyMuFiPWmf3KUI/foundations?node-id=0%3A1&t=ygVBFzp8cBMVwz9b-1

project includes a list of MVP features *

1. Create post with template
2. Comment on post
3. Author can edit post
4. Author can delete post
5. Display personal profile info on the right
6. Click on logo to reload homepage

project includes a data model See repo

MVP

app has at least 3 main features * see above MVP features

front end makes a request to the server and handles the response *

1. get request to get all posts (or single post), comments and profile info
2. post request to create new posts and comments
3. put request to edit and update posts
4. delete request to delete posts

front end is interactive * Buttons are interactive

app has custom styling * See wireframe

Front End

app has at least 5 semantic tags * nav for menu, aside for profile, section for post containers, header, figure for profile picture

app includes 1 view * see wireframe

app includes 1+ additional view(s) see wireframe

styling includes flexbox for arranging display of posts, menu and profile

at least 1 view is responsive N/A

Server

app includes a GET endpoint and handler function * get request to get all posts (or single post), comments and profile info

app includes a POST endpoint and handler function * post request to create new posts and comments

app includes a PUT endpoint and handler function put request to edit and update posts

app includes a DELETE endpoint and handler function delete request to delete posts

app utilizes Sequelize see backend code

project includes at least 1 controller file see backend code

Database

project includes a seed file or function see backend code

app uses 1 table see EER diagram

app uses 1+ additional tables see EER diagram

app uses a foreign key and join see backend

Presentation

discusses project purpose and demonstrates MVP * see video

does not discuss broken/unimplemented features see video

recording is between 2-3 minutes see video

ALL FEATURES SATISFIED EXCEPT RESPONSIVE DESIGN

Script

Hi everyone, I'm presenting my project Dormroom, which is a social media site for university students who are looking for collaborators for their projects and startups. I use HTML and CSS for the interface, JavaScript for coding the frontend and backend, express as the framework with Node.js, and PostgreSQL on bit.io for the database.

First, I'd like to show you the wireframe of the website. The design is made using Figma. The main features are showing the user info on the right which is pulled from the database, displaying all the existing posts and comments, creating new comments, creating new posts with the template, and updating and deleting posts created by the user.

I did some data modeling, made an EER diagram which shows the database containing three tables and some foreign keys. The database will be accessed from the backend with sequelize.

Now let's get into the website. When you first load it, it pulls the user info and the posts and comments from the database and display it with get request. I made the buttons interactive, the background-color will change when you hover over them so that you know they are real buttons.

When you click the comment button, the comment container will appear (because the hidden class of the comment container is removed) and you can see the existing comments of the post and the option to create a new comment. So I can try to type something here, send, and you can see the new comment displaying on the post.

Next we can go ahead and create a post. After clicking this button, a template will pop up. So the user can just fill in all the info here and post it. The post is added to the database through a post request and on this homepage you can see the newly rendered posts.

Note that if the post is made by the user themselves, they can edit or delete the post, but they can't do it for posts that are not posted by them. So the delete icon and the edit icon only show up on this latest post. We can go ahead and try editing the post now. The original content of the post will show up here, and you can just edit whatever you want and save the changes. The database is updated with this put request. And this bring us back to the homepage.

Last but not least, we can delete the post from the database with a delete request by clicking on the delete icon.