Capstone Step Two: Project Proposal
Laura Nino

1. What tech stack will you use for your final project?
For my project, I plan to use the MERN stack: MongoDB for the database, Express.js and Node.js for the backend server and API, and React for the frontend. I may also use Mongoose as the ODM for MongoDB, and Redux or Context API if I need global state management for things like user authentication and friend lists. For styling, I might use Tailwind CSS or Material UI, I am not sure yet.

2. Is the front-end UI or the back-end going to be the focus of your project? Or are you going to make an evenly focused full-stack application?
My project will be an evenly focused full-stack application. The front end will have user-friendly forms and interactive features (friend requests, recipe submission, ratings), while the back end will handle user accounts, friendships, recipe storage, comments, and ranking logic. Both parts are essential to the social and interactive goals of the app.
3. Will this be a website? A mobile app? Something else?
This will be a website, accessible via browsers, with a responsive design to work well on desktops and mobile devices, making it widely accessible without needing a dedicated app.

4. What goal will your project be designed to achieve?
The goal is to create a social recipe-sharing platform where users can register accounts, submit and share their own recipes, connect with friends to see each other's recipes, and interact by rating and commenting on them. It aims to make cooking more social, personalized, and fun.

5. What kind of users will visit your app? In other words, what is the demographic of your users?
The target users are home cooks, amateur chefs, and food enthusiasts of all ages who want to share their cooking ideas and discover new recipes. They might range from teens learning to cook to adults looking for meal inspiration and social interaction with friends.

6. What data do you plan on using? How are you planning on collecting your data?
You may not have picked your actual API yet, which is fine, just outline what kind of data you would like it to contain. You are welcome to create your own API and populate it with data. What does your database schema look like?
I'll create my own API and populate it with initial recipe data. The data will include user profiles (name, email, password), recipes (title, ingredients, instructions, image URL, user ID), friend

relationships (user ID pairs), and interactions (ratings 1-10, comments with user ID and recipe ID). The database schema in MongoDB might look like:

Users:

```
{
 "_id": ObjectId,
 "username": String,
 "email": String,
 "passwordHash": String,
 "friends": [ObjectId], // references to other users
 "createdAt": Date
}
```

Recipes:

```
{
 "_id": ObjectId,
 "author": ObjectId, // userId
 "title": String,
 "ingredients": [String],
 "instructions": String,
 "photoUrl": String,
 "category": String,
 "ratings": [
  {
   "userId": ObjectId,
   "score": Number
  }
 ],
 "comments": [
  {
   "userId": ObjectId,
   "text": String,
   "createdAt": Date
  }
 ],
 "createdAt": Date
}
```

7. What kinds of issues might you run into with your API? This is especially important if you are creating your own API, web scraping produces notoriously messy data.

Since I'm creating my own API, I might face:

- Validation issues: Ensuring users can't submit invalid or malicious data.
- Authentication and authorization: Making sure only logged-in users can post recipes, rate, or comment.
- Data relationships: Efficiently handling friend lists and showing only friends' recipes.
- Performance: Queries for recipes with many ratings/comments could be slower without indexing.

8. Is there any sensitive information you need to secure?

Yes.

- User passwords (will use hashing with bcrypt).
- User email addresses.
- Authentication tokens (will use JWT, stored securely).
- Input sanitation to avoid injection attacks.

9. What functionality will your app include?

- User registration and login/logout.
- User profiles with friend lists (friend request system like Facebook).
- Recipe submission form (including photos, ingredients, instructions).

- Viewing all recipes or just friends' recipes.
- Rating recipes (1–10 scale).
- Comments on recipes.
- Viewing average ratings and user reviews.
- Responsive design for mobile and desktop.

10. What will the user flow look like?

- Sign up / Login.
- Set up profile (optional picture or bio).
- Search for or browse users to send friend requests.
- Accept friend requests.
- Submit new recipes.
- View feed of friends' recipes.
- Rate and comment on recipes.
- Manage your own recipes (edit/delete).

- Log out.

11. What features make your site more than a CRUD app? What are your stretch goals?

More than CRUD:

- Social networking features (friend request system).
- Recipe ranking (1–10 scale) with average ratings.
- Comment threads on recipes.
- Personalized feed of friends' recipes.
- Notifications for friend requests / comments.

Stretch goals:

- Image uploads with cloud storage (AWS S3 or Cloudinary).
- Real-time chat or notifications with WebSockets.
- Advanced search and filters (by ingredients, category, rating).
- User profiles with cooking stats and favorite recipes.
- Mobile-first Progressive Web App (PWA) features.