**PROJECT REPORT SYNOPSIS**

**ON**

**Culinary-Crafts-Food-App**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**FOR**

**Full Stack Engineering (22CS037)**

****

|  |  |
| --- | --- |
| **Submitted To:**  Harpreet Kaur | **Submitted by:**  Sanya (2210992255)  Ishaan Singla (2210992582)  Samiksha (2210992238)  Sanjal Jain (2210992248) |

**Index**

|  |  |  |
| --- | --- | --- |
| Sr no | Topic | Page No |
| 1 | Problem Statement | 3 |
| 2 | Title of the Project | 3 |
| 3 | Objective | 3 |
| 4 | Options available to execute the project | 3 |
| 5 | Advantages | 4 |
| 6 | Disadvantage | 5 |
| 7 | References | 7 |

1. **Problem Statement:**

Develop a comprehensive web application that showcases food recipes, allows users to purchase ingredients, and includes social features for user interaction. The application should also provide a to-do list for managing cooking and shopping tasks. It should include a functional server-side database and provide a secure login and sign-up system to cater to individual user needs.

1. **Title of project:**

Culinary-Crafts-Food-App

1. **Objective:**
   1. Develop a secure login and sign-up system: Cater to individual users by allowing them to create and manage their accounts securely.
   2. Create a recipe section: Enable users to browse, search, and view detailed recipes.
   3. Implement a to-do list feature: Help users manage their cooking and shopping tasks efficiently.
   4. Build a shopping cart: Allow users to review and purchase ingredients directly from the website.
   5. Integrate a social section: Facilitate user interaction through chat and photo sharing, enhancing community engagement.
2. **Options available to execute the project:**
   1. **Programming Languages:** 
      1. **React.js**: A popular JavaScript library for building user interfaces, known for its component-based architecture and efficient rendering. (frontend)
      2. **Node.js**: A JavaScript runtime built on Chrome's V8 JavaScript engine, ideal for building scalable and efficient server-side applications. (backend)
   2. **Backend Frameworks:**
      1. **Express.js** (Node.js): Simple, flexible, and minimalistic, making it easy to set up routes and middleware.
      2. **Mongoose:** An elegant MongoDB object modeling tool designed to work in an asynchronous environment.
   3. **APIs:** 
      1. Standard approach for client-server communication: Well supported by most frameworks, enabling efficient and structured data exchange between the client and server.
   4. **version control:**
      1. **Github**: widely used version control system, providing robust collaboration features and hosting options for repositories.
   5. **Database:** 
      1. **MongoDB:** A NoSQL database known for its flexibility and scalability, ideal for handling large volumes of data and providing high performance.
   6. **Protected Route**
      1. Working Protected Routes so not everyone can just interact with app
3. **Advantages:**
   1. **Enhanced User Experience**
      1. **To-Do List Integration**: Having a to-do list beside the recipe allows users to actively engage with the cooking process, marking off tasks as they complete each step. This can make the cooking experience more organized and user-friendly.
      2. **Recipe and Ingredient Selling Integration**: Combining recipes with the ability to directly purchase ingredients from the website is highly convenient for users. It saves them time searching for ingredients elsewhere and boosts sales on your platform.
      3. **Social:** By having a chat system user will be able to share their experiences with other which allows other user to discover more recipes and have fun.
   2. **Personalization with Login/Signup**
      1. **User Profiles**: The login/signup feature allows for a personalized experience where users can save favourite recipes, track cooking history, and manage their personal to-do lists.
   3. **Seamless API Integration**
      1. **Third-Party Services**: The working API means you can easily integrate with external services such as online payments (PayPal, Stripe), shipping providers, or even recipe recommendation engines.
      2. **Scalability**: APIs make your platform more scalable, allowing you to easily add new features, connect to different databases, or work with external systems in the future.
   4. **E-Commerce and Monetization**
      1. **Selling Ingredients**: This opens up an additional revenue stream by allowing you to monetize the website not only through content (recipes) but also by selling products. It can be paired with dynamic pricing, seasonal promotions, or even partnerships with grocery stores.
      2. **Upselling Opportunities**: You can recommend complementary products (e.g., kitchen tools, spices) during the purchase process, increasing sales potential.
   5. **User Engagement**
      1. **Interactive To-Do List**: Engaging users with interactive features like marking recipe steps keeps them on the site longer, which can improve user retention and the likelihood of purchases.
      2. **Recipe Sharing and Reviews**: Users could potentially share their custom versions of recipes or leave reviews, further increasing engagement and creating a community around the platform.
4. **Disadvantages:**
   1. **Complexity and Maintenance**
      1. **Technical Complexity**: Managing user accounts, selling ingredients, integrating APIs, and maintaining a to-do list with live updates can add significant complexity to the website's backend. It will require regular maintenance, security updates, and debugging, which can become time-consuming.
      2. **API Downtime**: If the APIs you rely on (e.g., for payments or external ingredient sourcing) go down, your site could experience functionality loss, potentially frustrating users.
   2. **Security Concerns**
      1. **Handling User Data**: Login and signup functionalities involve collecting and storing sensitive user data (e.g., emails, passwords, addresses). This opens up vulnerabilities to data breaches or hacking if proper security measures (e.g., encryption, secure authentication) are not implemented.
      2. **E-Commerce Security**: Handling payments and storing user financial details requires high-level security. A breach in the payment gateway or user credentials could cause legal liabilities and damage your brand’s reputation.
   3. **Market Saturation**
      1. **High Competition**: There are already many established recipe websites and e-commerce platforms that sell ingredients (e.g., Blue Apron, HelloFresh). Gaining visibility in a crowded market might be a challenge, and you may need to offer a unique selling point to stand out.
      2. **Dependency on Logistics**: Selling ingredients means you have to manage inventory, shipping, and customer service for physical products. This adds logistical challenges, especially if you scale the business.
   4. **User Experience Challenges**
      1. **Overwhelming Interface**: Having multiple features (recipes, to-do list, ingredient store) might overwhelm users if not designed intuitively. Poor UI/UX could make navigation difficult and detract from the overall user experience.
      2. **Cross-Platform Consistency**: Ensuring that the interactive to-do list, recipe display, and e-commerce functions work seamlessly across devices (desktop, mobile, tablet) can be challenging and time-consuming.
   5. **Cost of API Usage**
      1. **API Costs**: Depending on the API provider, you may incur additional costs for usage, especially if traffic increases. Some APIs charge based on the number of requests, which can escalate costs as your site grows.
      2. **Vendor Lock-in**: Relying on third-party APIs for crucial functionalities (e.g., payment gateways, ingredient sourcing) can create vendor lock-in, making it difficult to switch providers or manage downtime.

**References:**

* <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs>
* <https://netninja.dev/p/node-js-crash-course>
* <https://www.mongodb.com/docs/>
* <https://www.npmjs.com/>
* <https://mongoosejs.com/>
* <https://www.mongodb.com/atlas>