KitchenMate

REACT APP – INITIAL PLAN DOCUMENTATION

Overview

The purpose of this project is to develop an application that allows users to create, manage, and organize their recipes. Users can add ingredients to their recipes, modify them, and scale them according to the desired number of portions. Additionally, users can combine multiple recipes to create complete dishes or portion sets. The application also enables users to generate shopping lists by adding selected recipes to a basket.

Deployment

App will be deployed to Firebase, to be accessible online.

Version Control

Application version control will be managed with Github.

Front-End

Tech-Stack

- JavaScript
- React
- Tailwind CSS

Components

- Header
  - A Simple header component, with title of app
- Login
  - Simple modal-based component for logging in
- Navigation
  - Side-bar with options for changing the content of MainContainer
- MainContainer
  - “Wrapper” for displaying different sections: recipe list, basket, recipe creation
- RecipeList/ShoppingList
  - “Wrapper” for displaying recipes/shoppinglist with list items

- RecipeForm
    - Recipe creation component with inputs
- RecipeCard/DishCard
    - Modal-based component for displaying card of recipe/dish details
- DishCreation
    - Compiling different recipes to one dish

State Management

- useContext
    - to avoid prop drilling, managing global shared states and have cleaner code
- useReducer
    - To manage states in bigger scale and avoid using useState
    - Manage states considering shopping lists and recipe lists
- useState
    - To manage state that is not directly related to shopping list or recipe lists
- useRef
    - To store reference to original list, in case of filtering
- useEffect
    - For side-effects like fetching data from APIs

API Integration

- **fetchData**
    - Fetches data stored in backend/database
    - Parameters:
        - user
        - endpoint
- **fetchRecipes**
    - Passes parameters to fetchData function
- **fetchBasket**
    - passes parameters to fetchData function
- **fetchDishes**
    - passes parameters to fetchData function

- **postData**
    - Posts data stored in backend/database
    - parameters:

        - user
        - endpoint
- **postRecipe**
    - passes parameters to postData function
- **postBasket**
    - passes parameters to postData function
- **postDish**
    - passes parameters to postData function

- **modifyData**
    - Modifies data stored in backend/database
    - parameters:
        - user
        - endpoint
        - id
- **modifyRecipe**
    - passes parameters to modifyData function
- **modifyBasket**
    - passes parameters to modifyData function
- **modifyDish**
    - passes parameters to modifyData function

- **deleteData**
    - Deletes data from endpoint
    - parameters:
        - user
        - endpoint
        - id
- **deleteRecipe**
    - passes parameters to deleteData function
- **deleteBasket**
    - passes parameters to deleteData function
- **deleteDish**
    - passes parameters to deleteData function

- **authenticateUser**
    - Communicates with server to see if user and password matches, returns true or false.
    - parameters:
        - username
        - password
- **createNewUser**
    - Create a new user
    - parameters:
        - username

        - password

Backend/Database

Tech Stack

- PHP
- (SQL) *will try to do this*

Functionality

- **GET /recipes, /dishes, /basket**
    - Retrieves wanted data from endpoint
- **POST /recipes, /dishes, /basket**
    - Posts new data to wanted endpoint
- **PUT /recipes, /dishes, /basket**
    - Modifies data in wanted endpoint
- **DELETE /recipes, /dishes, /basket**
    - Deletes data from wanted endpoint

Data Storage

Data will be held on JSON file on server in the beginning. When moving on with the project SQL will be implemented.

Recipe Structure

- name
- portions
- ingredientsList
- prepTime
- instructions

Basket Structure

- basketName
- ingredientList

Dish Structure

- name
- recipesList/components
- image

User Structure

- name
- password (HASHED)