

# DESIGN DOCUMENT

## List of Contributors

Beril Aydın 231101002

Nisa Naz Keleşoğlu 231101040

Zeynep Saçaklı 221101051

## Task Matrix

Beril Aydın	Nisa Naz Keleşoğlu	Zeynep Saçaklı
Implementation Details	System Overview	Use Case Support in Design
Design Decisions	Design Decisions	Design Decisions

# Table Of Contents

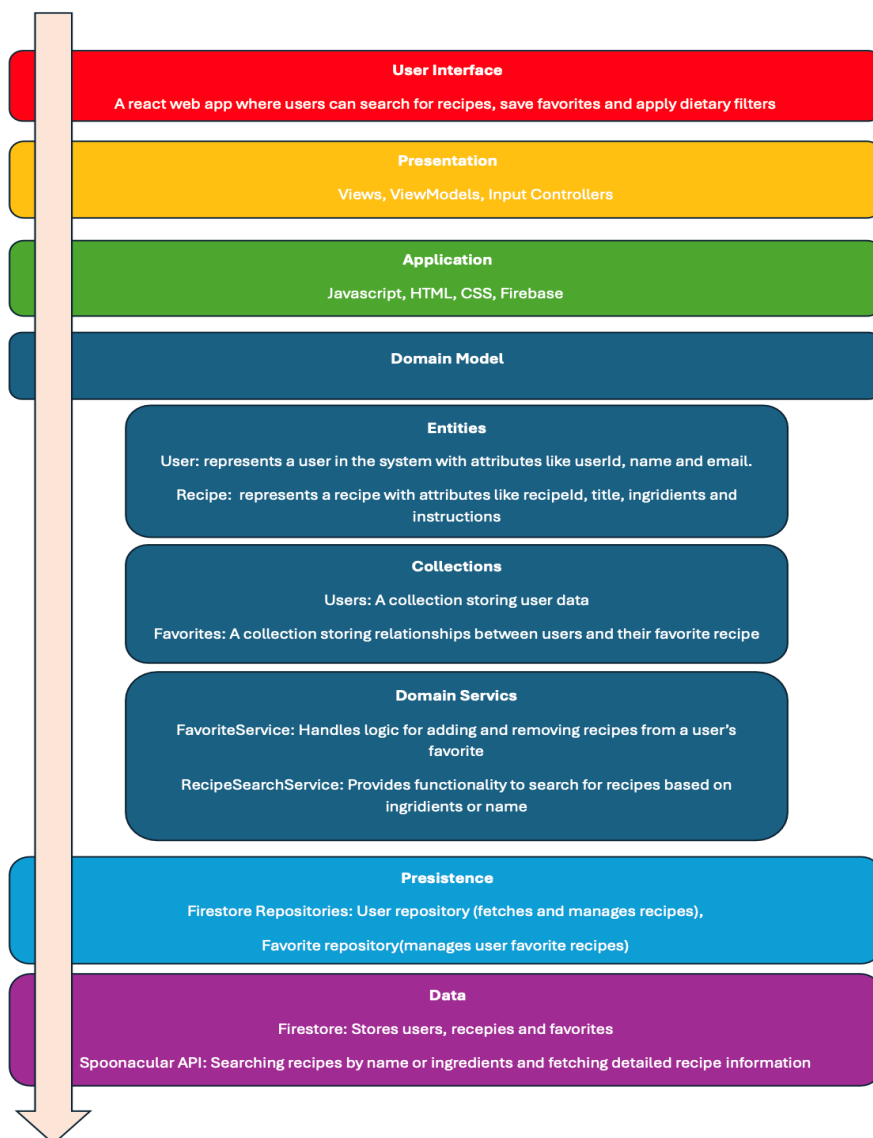
1. System Overview
  - Project Description
  - System Architecture
  - Technology Stack
2. Implementation Details
  - Codebase Structure
  - Key Implementations
  - Component Interfaces
  - Visual Interfaces
3. Use Case Support in Design
  - Use Case Selection
  - Requirement Mapping
  - Use Case Design
4. Design Decisions
  - Technology Comparisons
  - Decision Justifications

# 1) System Overview

## ➤ Project Description

Todayz Meal is a web application designed to help users find recipes based on the ingredients they have. Users can receive suggestions for meals they can prepare by entering their ingredients. Moreover, they can filter the meals according to preparation time and dietary preferences(vegetarian, vegan, gluten-free, lactose-free), and add recipes to their favorite section. This project aims to reduce food waste, make cooking more convenient and help people what to eat today.

## ➤ System Architecture



## ➤ Technology Stack

Frontend: HTML, CSS, React

Backend & Database: JavaScript, Firebase

APIs: Spoonacular API for recipe data

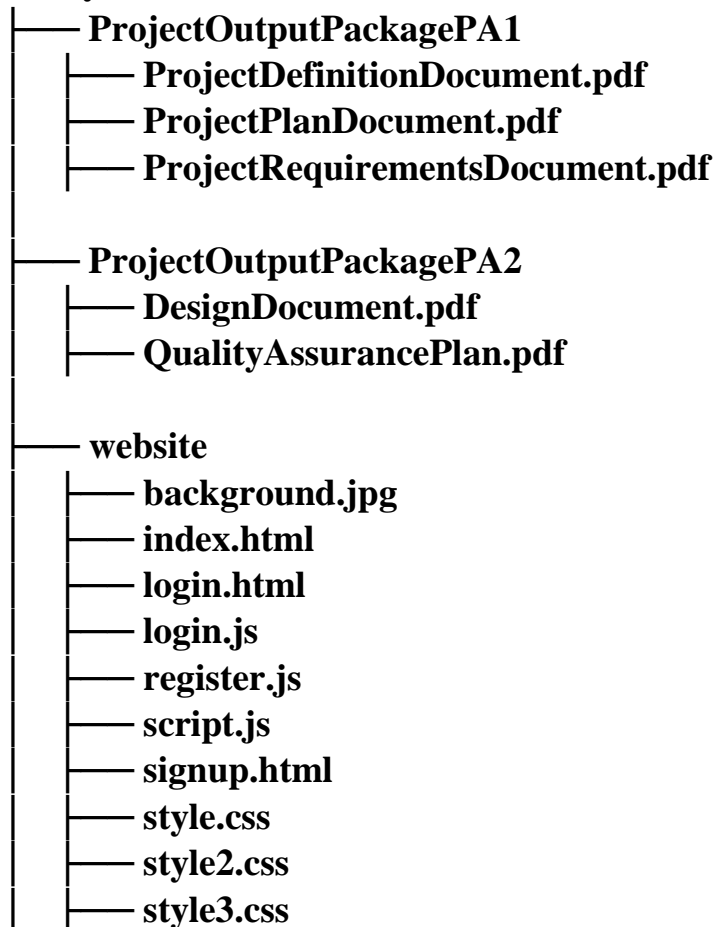
Version Control: GitHub

Communication: Slack

## 2) Implementation Details

### ➤ Codebase Structure

**TodayzMeal/**



## ➤ Key Implementations

### 1. Recipe Search Module

- a. Uses Spoonacular API to fetch recipes based on user-input ingredients and recipe names
- b. Includes error handling for API failures and alternative suggestions when no exact match is found.

### 2. User Authentication

- a. Firebase Authentication handles user sign-up, login, and session management.
- b. Secure authentication with email and password
- c. Password validation includes minimum length, uppercase letters, and numeric requirements.

### 3. Favorites Feature

- a. Users can add recipes to their favorites, which are stored in Firebase.
- b. A "most favored" algorithm ranks popular recipes.
- c. Favorite recipes sync across devices using Firebase real-time database

### 4. Filtering Mechanism

- a. Users can filter recipes by dietary preferences (vegetarian, vegan, gluten-free, lactose-free).
- b. Sorting by preparation time allows quick meal selection.
- c. Backend logic ensures filtering occurs efficiently for large datasets.

### 5. User Profile Management

- a. Users can update personal details and dietary preferences.
- b. Profile data is stored securely in Firebase

## ➤ Component Interfaces

### 1. Authentication Components

#### • Login Component

- **Input Fields:** Email and password input fields.
- **Button:** Submit button for logging in.
- **Link:** Redirects to the signup page for account creation.
- **Functionality:** Handles user authentication and redirects to the main page upon successful login.

#### • Signup Component

- **Input Fields:** Email, password, and password confirmation fields.
- **Buttons:** Signup and cancel buttons.
- **Functionality:** Creates a new user account and redirects to the login page upon successful registration

### 2. Main Application Components

#### • Search Component

- **Input Field:** Ingredient input field for searching recipes.
- **Checkboxes:** Vegan and vegetarian filters.
- **Button:** Search button to trigger recipe search.
- **Functionality:** Searches for recipes based on the entered ingredients and applies filters.

#### • Recipe List Component

- **List Items:** Displays a list of found recipes.
  - **Recipe Details Button:** Button to view details for each recipe.
  - **Functionality:** Lists recipes and allows users to view detailed information.
- **Recipe Details Modal**
  - **Content:** Displays recipe title, image, preparation time, servings, ingredients, and instructions.
  - **Button:** Close button to dismiss the modal.
  - **Functionality:** Shows detailed information about the selected recipe.

### 3. Navigation Components

- **Logout Button**
  - **Functionality:** Logs out the user and redirects to the login page.

### 4. Styling Components

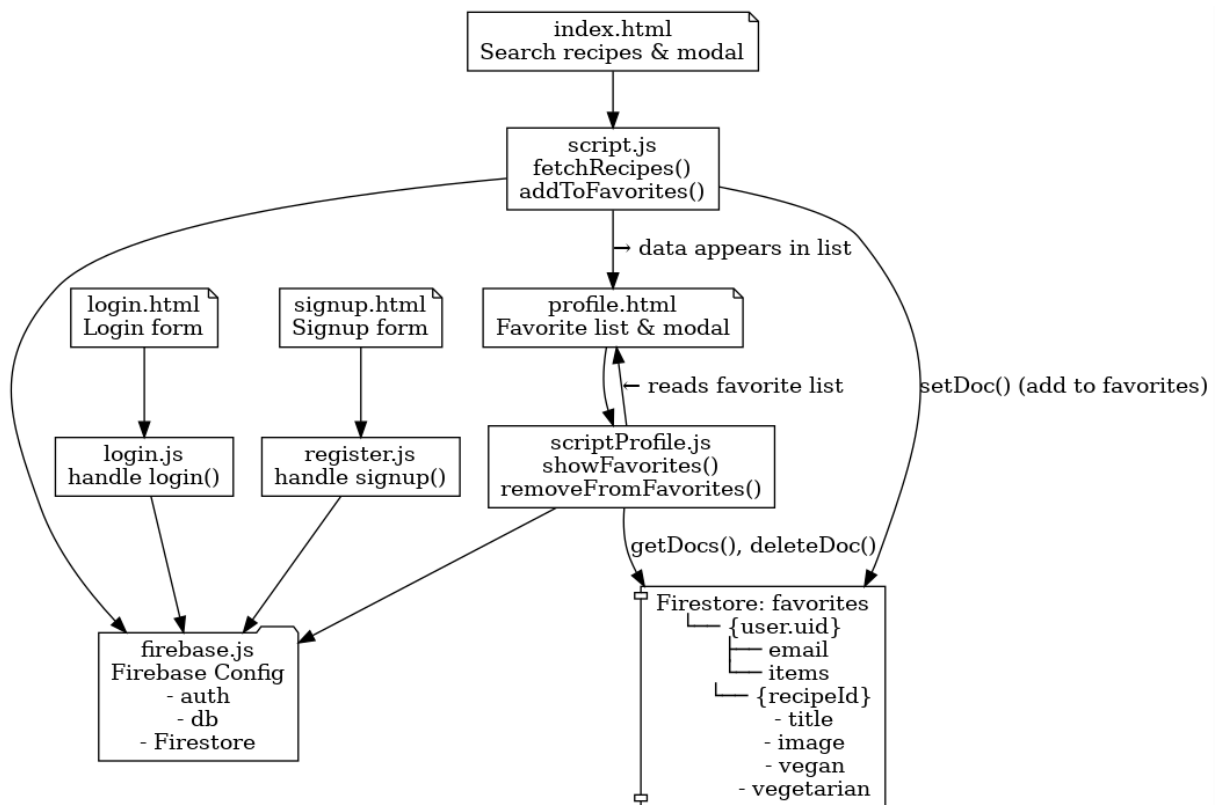
- **CSS Files**
  - **style.css:** Styles for the main page.
  - **style2.css:** Styles for the login page.
  - **style3.css:** Styles for the signup page.

### 5. API Integration

- **Spoonacular API**
  - **Functionality:** Used to fetch recipe data and details based on user input.
- **API Endpoints**
  - [https://api.spoonacular.com/recipes/findByIngredients?ingredients=...&apiKey=API\\_KEY](https://api.spoonacular.com/recipes/findByIngredients?ingredients=...&apiKey=API_KEY)
  - [https://api.spoonacular.com/recipes/{id}/information?apiKey=API\\_KEY](https://api.spoonacular.com/recipes/{id}/information?apiKey=API_KEY)

### 6. Firebase Integration

- **Authentication**
  - **Functionality:** Handles user registration and login using Firebase Authentication.



## ➤ Visual Interfaces

### SIGN UP

SIGN-UP

Email

Password

Repeat Password

Cancel
Sign up

### LOGIN

Today's MEAL

Username

Password

not registered? Create an account

### MAIN

☒ Profile

Today's MEAL

Enter ingredients

☐ Vegan
 ☐ Vegetarian

Recipes

⋮

### USER ACCOUNT

Username

Favorite Recipes

⋮

### 3) Use Case Support in Design

#### ➤ Use Case Selection

- 1) User must be able to create an account.
- 2) The system must generate recipe suggestions based on input ingredients.
- 3) Users must be able to save favorite recipes.
- 4) The system must display recipe including ingredients, steps, and estimated cooking time.

#### ➤ Requirement Mapping

Use Case	Requirement
User account creation	Users must create an account securely.
Recipe suggestion system	The system must generate recipe suggestions based on input ingredients.
Favorite recipes	Users must be able to save favorite recipes.
Recipe display	The system must display recipe including ingredients, steps, and estimated cooking time.

#### ➤ Use Case Design

##### • User Login

<b>Name</b>	User login
<b>Actors</b>	User
<b>Entry Condition</b>	User is registered to the system
<b>Event Flow</b>	<ol style="list-style-type: none"><li>1. User connects to the website</li><li>2. Enters username and password</li><li>3. Clicks on Login</li><li>4. System checks credentials validity and directs the user to main page if the informations valid.</li></ol>



- **User Account Creation**

<b>Name</b>	User Account Creation
<b>Actors</b>	Guest
<b>Entry Condition</b>	A person has a working internet connection
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. Clicks on “Sing Up”</li> <li>2. Inserts his data and fills mandatory fields</li> <li>3. Account is registered by the system</li> <li>4. The system directs the user to the login page.</li> </ol>
<b>Exit Conditions</b>	The guest now is registered into the system and becomes a User
<b>Exception</b>	User is already registered Email is already associated to another account

- **Recipe Suggestion**

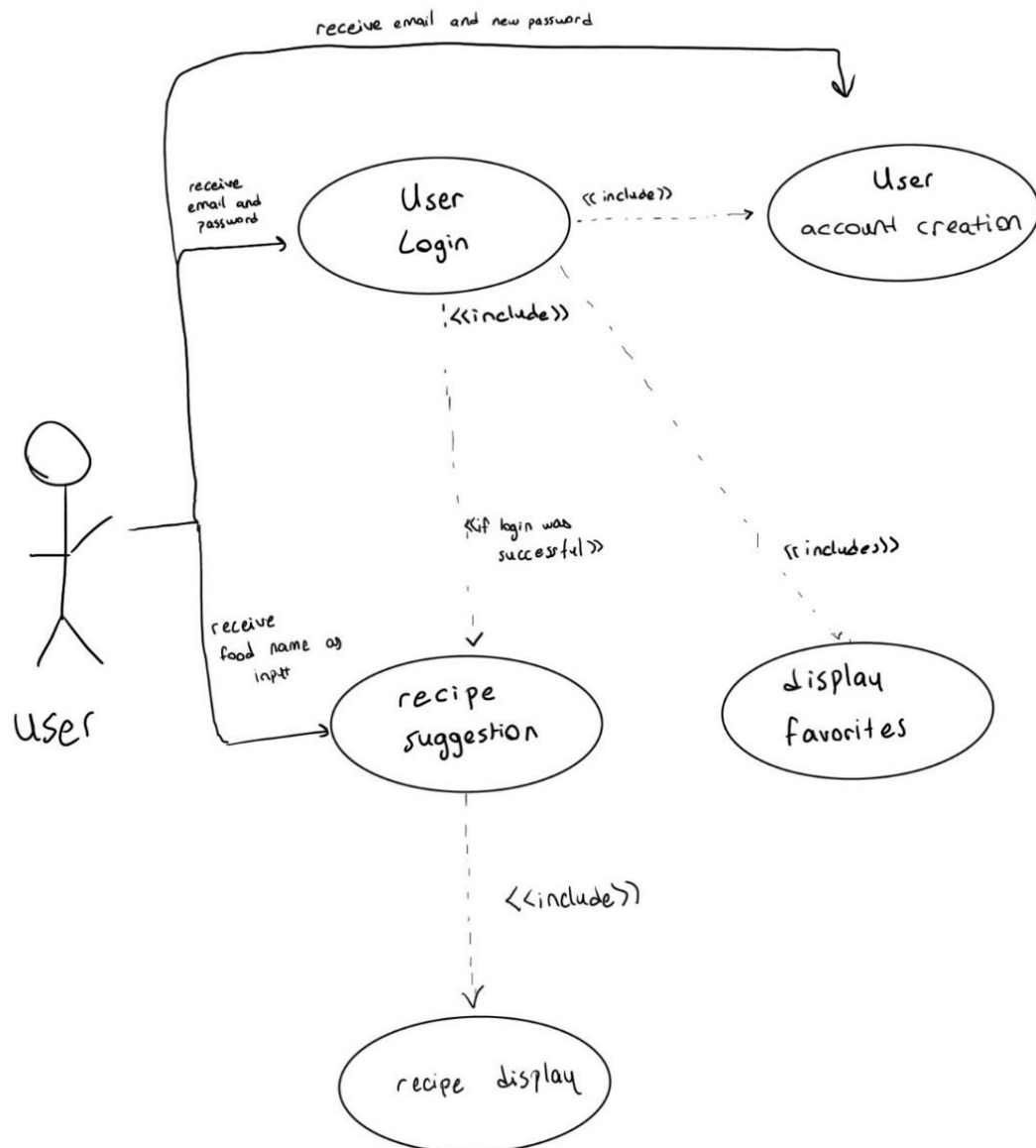
<b>Name</b>	Recipe Suggestion
<b>Actors</b>	User
<b>Entry Condition</b>	The user is logged to the website
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The food ingredients to be used were entered into the search bar.</li> <li>2. Vegan and vegetarian filtering options selected</li> <li>3. Clicks on “Search Recipe”</li> </ol>
<b>Exit Condition</b>	Clicks on “Logout”

- **Display Recipes**

<b>Name</b>	Display Recipes
<b>Actors</b>	User
<b>Entry Condition</b>	The user has entered ingredients and clicked ‘Search’
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The user enters ingredients into the search bar.</li> <li>2. Vegan and vegetarian filtering options selected (Optional)</li> <li>3. Clicks on “Search Recipe”</li> <li>4. The system displays a list of matching recipes.</li> <li>5. The user clicks ‘Details’ to view more information.</li> </ol>
<b>Exit Condition</b>	Clicks on “Close” and recipe list or detail modal is closed.

- Display Favorites

<b>Name</b>	Display Favorites
<b>Actors</b>	User
<b>Entry Condition</b>	The user is logged to the website
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. Click 'Add to recipes' button of any recipe in its details section.</li> <li>2. Clicks on profile icon</li> <li>3. Redirects to profile page</li> <li>4. Favorites listed under the Favorite Recipes section.</li> </ol>
<b>Exit Conditions</b>	Clicks on "Close"



## 4) Design Decisions

### ➤ Technology Comparisons

#### Database: Firebase vs. MySQL

**Firebase:** Provides real-time data syncing and easy authentication but is less suitable for complex queries.

**MySQL:** A widely used relational database with high performance but requires additional setup for real-time capabilities.

**Decision:** Firebase was chosen due to its seamless integration with authentication and real-time updates.

#### Frontend Framework: React vs. Angular

**React:** Offers component-based architecture, a strong developer community, and efficient performance.

**Angular:** Provides a full-fledged MVC framework but has a steeper learning curve.

**Decision:** React was chosen for its flexibility and widespread industry use.

#### Version Control System: GitHub vs. GitLab

**GitHub:** Industry standard for open-source projects, great integration with CI/CD tools, large community support.

**GitLab:** Provides built-in CI/CD pipelines, good for self-hosting but requires more setup.

**Decision:** GitHub was chosen due to its ease of use, team collaboration features, and widespread adoption.

### ➤ Decision Justification

- **Frontend (HTML, CSS, React):** React provides a reusable component-based UI structure, making frontend development modular and maintainable. HTML and CSS ensure a structured and styled user interface.
- **Backend & Database (JavaScript, Firebase):** Express.js ensures lightweight, high-performance API handling, making it ideal for a scalable backend. Firebase offers real-time syncing and seamless authentication integration.

- **APIs (Spoonacular API):** Used to fetch a wide variety of recipe data, allowing dynamic ingredient-based searches and filtering.
- **Version Control (GitHub):** Chosen for its industry-wide adoption, strong collaboration features, and seamless integration with CI/CD tools.
- **Communication (Slack):** Enables real-time team collaboration and task updates.