Requirement Stack

Created on 10/17 by A.W.I.

Requirement ID	Description of Requirement	Status	Story Points	Priority	Sprint No.
0	Determine technology stack	Done	2	1	1
0.1	Initialize code base within GitHub	Done	2	1	1
0.2	Establish dependencies being used	Done	1	2	1
0.3	Design and describe project architecture	Done	2	1	1
0.4	Create a rough draft for the application UI	Done	3	3	1
0.5	Finalize application name and synopsis	Done	1	2	1
1	Implement user authentication	Done	8	1	2
1.1	Design and set up user database schema (e.g., using Firebase Auth or a custom solution	Done	3	1	2
1.2	Create user Registration UI screen and form	Done	5	1	2
1.3	Create user Login UI screen and form	Done	2	1	2
1.4	Implement "Forgot Password" flow and UI	Done	3	3	2
1.5	Implement input validation and error messaging for auth forms	Done	3	1	2
2	Create main UI layout	Done	5	1	2
2.1	Implement core navigation structure (e.g., Bottom Nav Bar or Drawer)	Done	3	1	2
2.2	Create a reusable and themable base screen template	Done	2	3	2
2.3	Define and implement a global app theme (colors, typography)	Done	3	3	2
2.4	Create a custom app header/action bar component	Done	2	3	2
3	Develop ingredient input with manual entry	Done	5	4	2
3.1	Build a user-friendly manual input field for ingredients	Done	3	4	2
3.2	Implement auto-suggestions for common ingredients as user types	Done	2	1	2
3.3	Implement a dynamic list to display entered ingredients	Done	2	2	2
3.4	Add functionality to edit and remove items from the ingredient list	Done	2	1	2
3.5	Add a button to clear the entire ingredient list	Done	2	4	2

Requirement Stack

Created on 10/17 by A.W.I.

Requirement ID	Description of Requirement	Status	Story Points	Priority	Sprint No.
4	Integrate with Spoonacular Recipe API	Done	8	1	3
4.1	Set up API service layer and network client	Done	5	3	3
4.2	Parse and transform API response into app-specific recipe model objects	Done	3	3	3
4.3	Implement robust error handling for API calls (e.g., no network, API limit)	Done	2	3	3
4.4	Implement loading states (skeleton screens/spinners) during API calls	Done	2	2	3
5	Build and display recipe search results	Done	6	3	3
5.1	Create a recipe card UI component to display search results	Done	3	3	3
5.2	Build a scrollable results list/grid view	Done	3	4	3
5.3	Implement "No results found" state UI	Done	2	2	3
6	Add dietary preference features	Done	13	5	3
6.1	Create a settings/preferences screen for users to select dietary restrictions	Done	5	3	3
6.2	Integrate selected dietary filters into the Spoonacular API request	Done	8	3	3
6.3	Implement client-side filtering of manual ingredients based on dietary rules	Done	2	2	3
7	Develop image-based ingredient identification	Done	5	5	3
7.1	Implement image capture using device camera	Done	5	3	3
7.2	Implement image selection from device gallery	Done	5	3	3
7.3	Integrate with a Computer Vision API (e.g., Google Vision) to identify ingredients	Done	8	3	3
7.4	Display the list of identified ingredients to the user for confirmation/editing	Done	3	2	3
8	Implement recipe saving and management	Done	8	5	3

Requirement Stack

Created on 10/17 by A.W.I.

Requirement ID	Description of Requirement	Status	Story Points	Priority	Sprint No.
8.1	Design and implement the database structure for user-specific recipe collections	Done	5	4	3
8.2	Add "Save Recipe" button and functionality to recipe cards	Done	2	2	3
8.3	Create a "My Saved Recipes" screen to view the collection	Done	3	2	3
8.4	Allow users to remove recipes from their saved collection	Done	2	1	3
9	Implement shared recipe collections	Done	5	3	4
9.1	Design and implement the database structure for shared recipe collections	Done	5	4	4
9.2	Create UI for users to move saved recipes to a shared space	Done	3	2	4
9.3	Build a screen to view the shared recipe collection	Done	2	2	4
10	Create meal planner with grocery list generation	Done	8	5	4
10.1	Build a weekly calendar view for assigning saved recipes to the calendar	Done	5	3	4
10.2	Develop drag-and-drop functionality for assigning recipes to the calendar	Done	5	3	4
10.3	Develop a feature to automatically aggregate ingredients from the weekly plan	Done	3	2	4
10.4	Implement de-duplication logic for the aggregated grocery list	Done	3	2	4
10.5	Create a dedicated, readable grocery list screen	Done	2	3	4
10.6	Add ability to manually check off items on the grocery list	Done	2	2	4
11	Final UI/UX Polish and Testing	Done	8	5	4
11.1	Add micro-interactions and animations for a polished feel	Done	5	3	4
11.2	Conduct comprehensive UI/UX testing across multiple devices and screen sizes	Done	5	3	4
11.3	Fix all critical and major UI/UX bugs identified during testing	Done	5	3	4
11.4	Perform usability testing with a small group of target users	Done	3	3	4