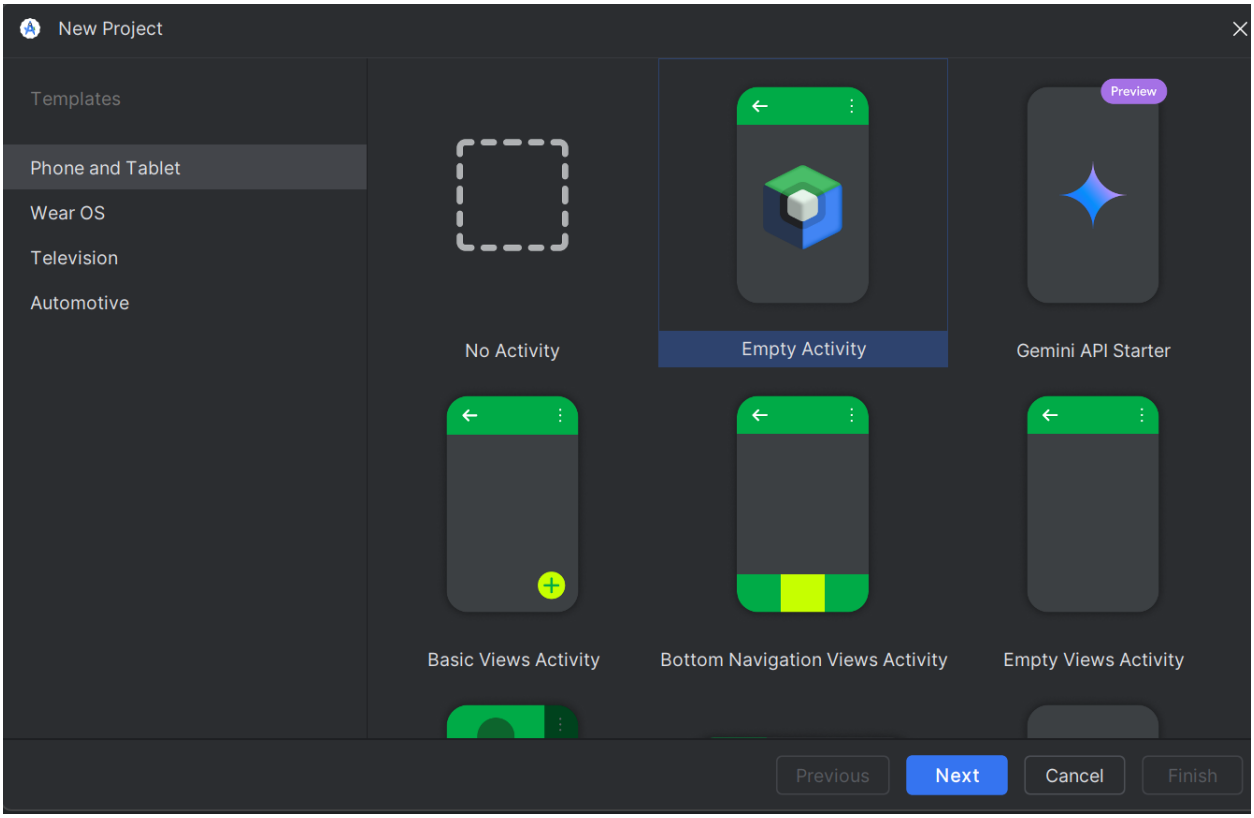
Student ID: N01651380

Student Name: Madepelli Sadhana

Task 1: Setting Up the Project
1. Open Android Studio and create a new Android project.
2. Name the project RecipeManagerApp and use Kotlin as the programming language.
3. Set the minimum API level to 21 (Android 5.0 Lollipop) to ensure compatibility with modern devices.
4. Choose an Empty Activity template and configure your project structure.



Task 2: Define Classes
1. Create a Recipe class in a separate Kotlin file with the following properties:
o id: Int (unique identifier for the recipe).
o title: String (name of the recipe).
o ingredients: String (list of ingredients required).
o instructions: String (step-by-step guide to prepare the recipe).
o category: String (e.g., Dessert, Main Course).

The Recipe class acts as a base class and it serves as a template from which more specific recipe types can inherit common properties.

2. Create a subclass DessertRecipe that extends Recipe and adds the following property:
o sweetnessLevel: String.

The DessertRecipe class extends the Recipe class. The common properties inherited from the Recipe class, it introduces a new property:

- **sweetnessLevel: A string that indicates how sweet the dessert is, such as "Low," "Medium," or "High."**

o Override a method getDetails() to include the sweetness level in the recipe details.
3. Create a subclass MainCourseRecipe that extends Recipe and adds the following property:
o preparationTime: Int (in minutes).
o Override the getDetails() method to include preparation time in the recipe details.

The getDetails() method is overridden in this subclass to customize how the details of the recipe are displayed. The overridden method adds the sweetnessLevel to the output, making the dessert-specific information visible when printing or logging details of a dessert recipe. The MainCourseRecipe class also extends the Recipe class. This subclass introduces a new property:

- **preparationTime: An integer indicating the number of minutes required to prepare the main course.**

Task 3: Implement UI

1. MainActivity:
   o Create a simple screen with buttons to navigate to:
   ▪ Add/Edit Recipe Activity
   ▪ View Recipe Activity
   2. AddEditRecipeActivity:
   o Create a layout with input fields for the recipe's title, ingredients, instructions, and category.
   o Add a button to save the recipe and navigate back to the Main Activity.
   3. ViewRecipeActivity:
   o Create a layout to display the recipe details.
   o If the recipe is a DessertRecipe, show the sweetness level.
   o If the recipe is a MainCourseRecipe, show the preparation time.
   o Add a button to delete the recipe and navigate back to the Main Activity.

PrimaryActivity

In the MainActivity, the user sees a straightforward interface with two buttons. A single button allows access to the AddEditRecipeActivity, enabling users to add or modify recipes. The second button directs users to the ViewRecipeActivity, enabling them to see a particular recipe. This primary display acts as the central point for the app, offering entry to additional functionalities.

EditAddRecipeActivity

In the AddEditRecipeActivity, users have the option to either create a new recipe or modify an existing one. The design features input areas for crucial recipe information, including the recipe name, ingredients, steps, and category (like Dessert or Main Course). A button exists to save the recipe that, when pressed, will keep the information and redirect the user back to the MainActivity, showing the updates. This task guarantees that the user can effortlessly insert or modify recipe details.

ViewRecipeScreen

The ViewRecipeActivity showcases the specifics of a chosen recipe. The arrangement is created to display the recipe's name, components, steps, and type. If the recipe is a DessertRecipe, the design will also show the sweetness level to reflect how sweet the dessert tastes. For a MainCourseRecipe, it displays the preparation time to show the duration needed for preparation. This activity features a button to delete the recipe, which, when clicked, removes the recipe from the system, redirecting the user back to the MainActivity.

Task 4: Implement Basic Navigation
1. Use Intents to navigate between the activities.
2. Pass the recipe object between activities using Intent.putExtra().
3. Handle the result in the Main Activity to update the recipe object.

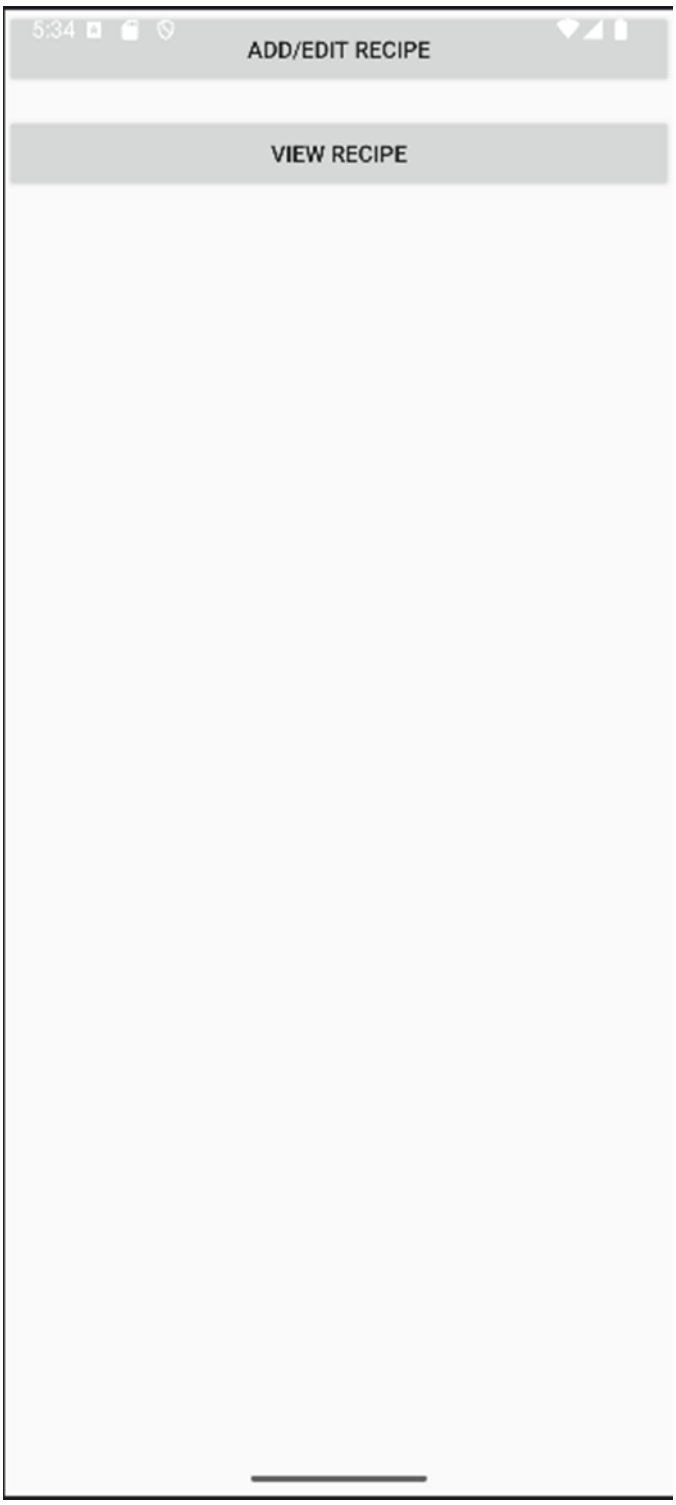
In Task 4, we utilize Intents to switch between activities. For instance, MainActivity employs an Intent to launch AddEditRecipeActivity or ViewRecipeActivity. To transfer recipe information between activities, we utilize `Intent.putExtra()`, which transmits the recipe's specifics (such as title, ingredients, and category). Upon the user's return to MainActivity after adding or modifying a recipe, the outcome is managed with `onActivityResult()` or `setResult()` to refresh the recipe information. This guarantees that the app's interface displays any alterations applied to the recipe.

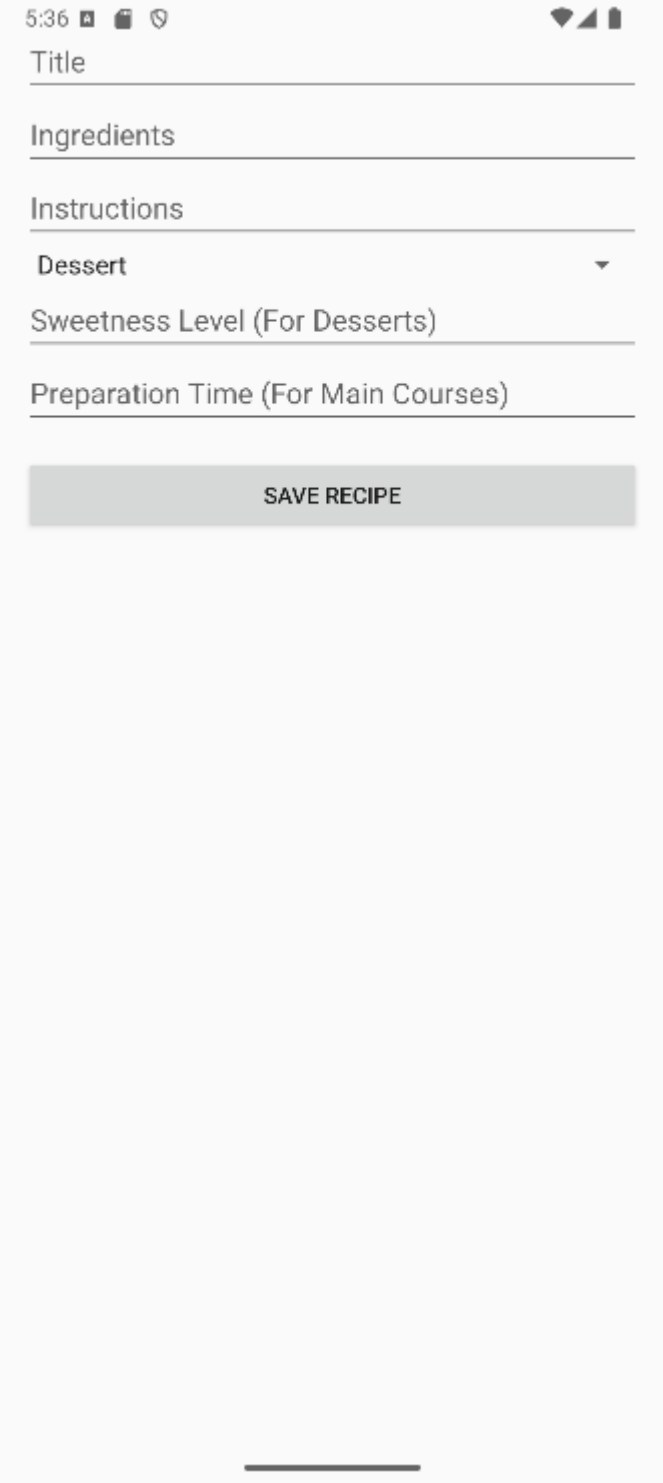
Task 5: Implement Simple CRUD Operations
1. Create/Update Recipe:
o In the Add/Edit Recipe Activity, collect the recipe details from the user input and store them in a single Recipe object.
o Use basic validation to ensure all fields are filled.
2. Read Recipe:
o In the View Recipe Activity, display the details of the recipe.
o Use the getDetails() method to include subclass-specific details.
3. Delete Recipe:
o In the View Recipe Activity, provide a button to delete the recipe.
o Clear the Recipe object and navigate back to the Main Activity.

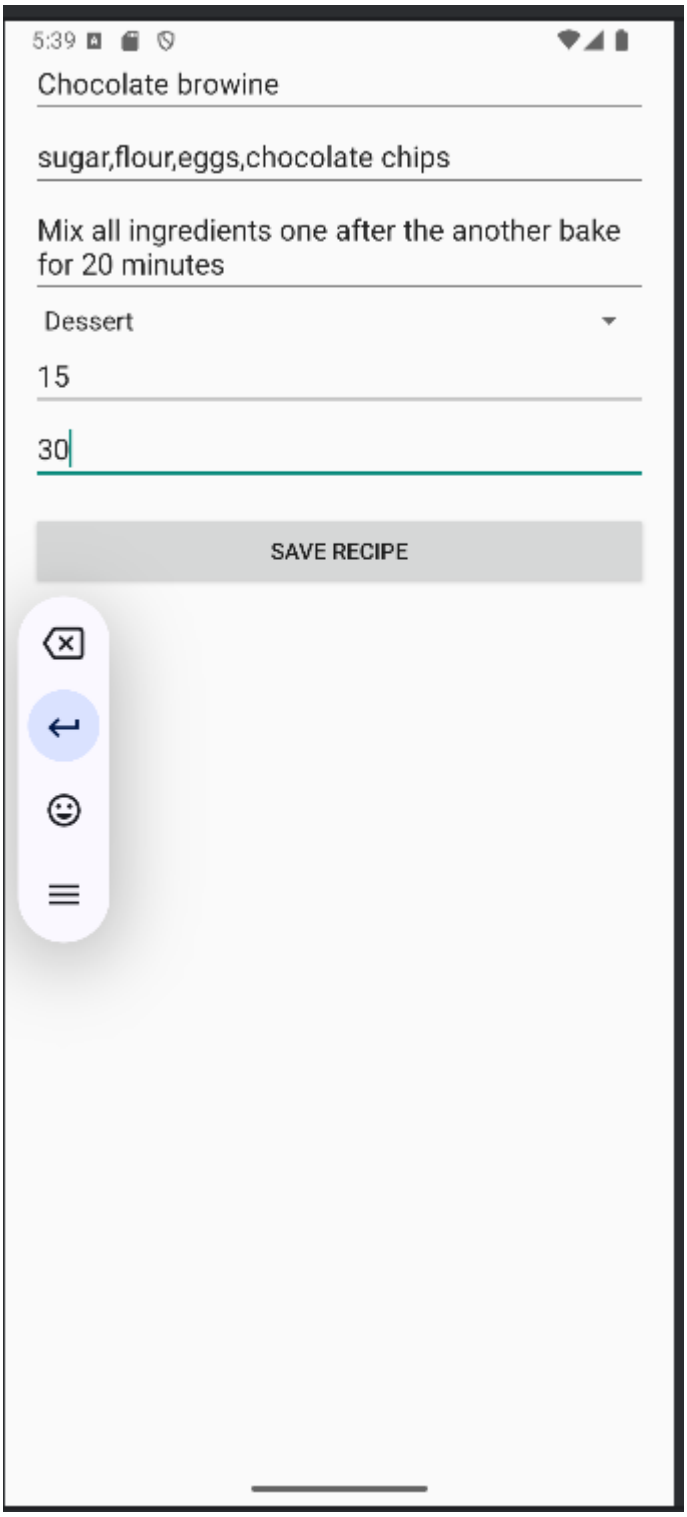
Task 6: Test and Debug
1. Test each functionality thoroughly:
o Add or edit a recipe and verify that the details are saved and displayed correctly.
o View the recipe details and ensure the information matches the input.
o Delete the recipe and confirm that it is removed.
2. Debug any issues to ensure smooth navigation and functionality.
3. Use Android Studio's Logcat to monitor runtime logs and identify potential errors

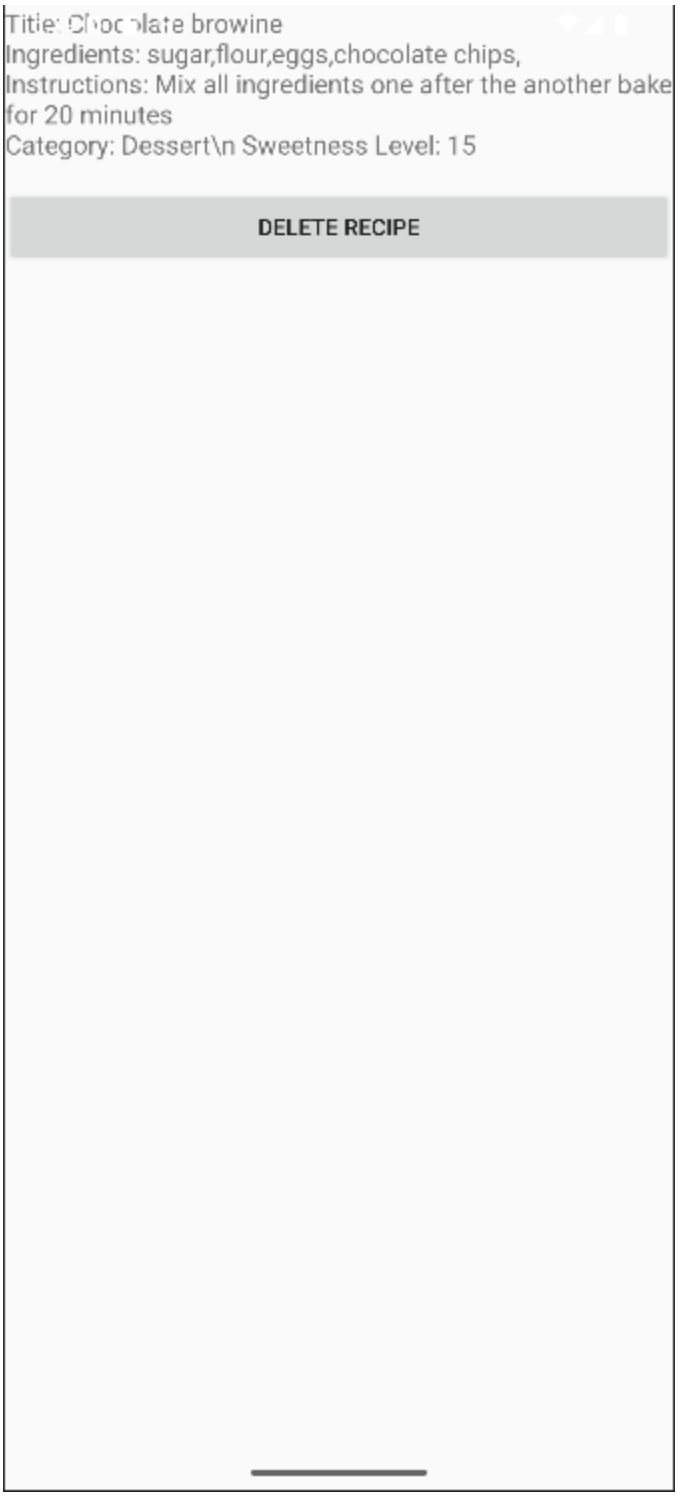
App Description:

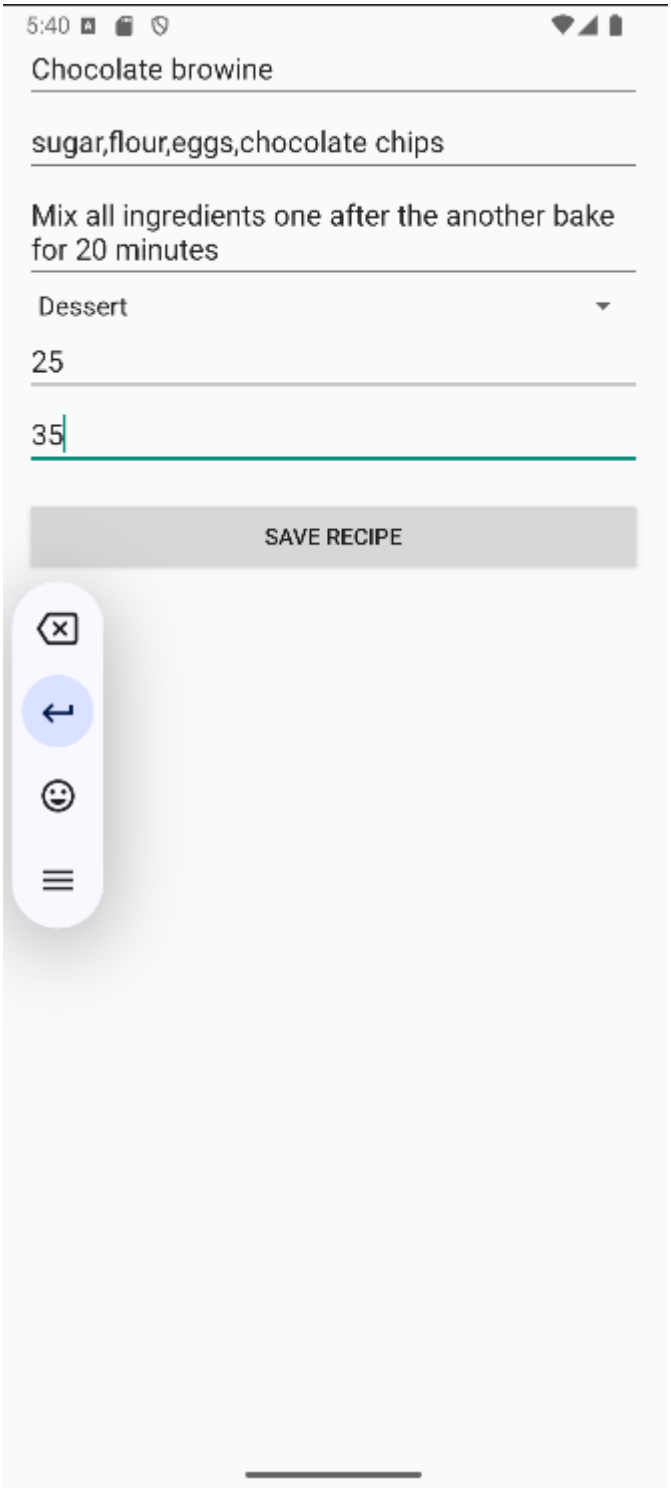
The RecipeManagerApp is an Android app that enables users to organize their recipes via an easy CRUD (Create, Read, Update, Delete) framework. The application features three primary activities: MainActivity, AddEditRecipeActivity, and ViewRecipeActivity. In MainActivity, users can access either the Add/Edit Recipe or View Recipe functions. The Add/Edit Recipe feature enables users to enter information about a recipe, including its name, ingredients, directions, and category (such as dessert or main dish). When saved, the recipe is kept and shown in the View Recipe activity, allowing users to access extra details depending on the recipe's category (e.g., sweetness for desserts or prep time for main dishes). The application also enables users to remove a recipe, clearing the information and going back to the main screen.

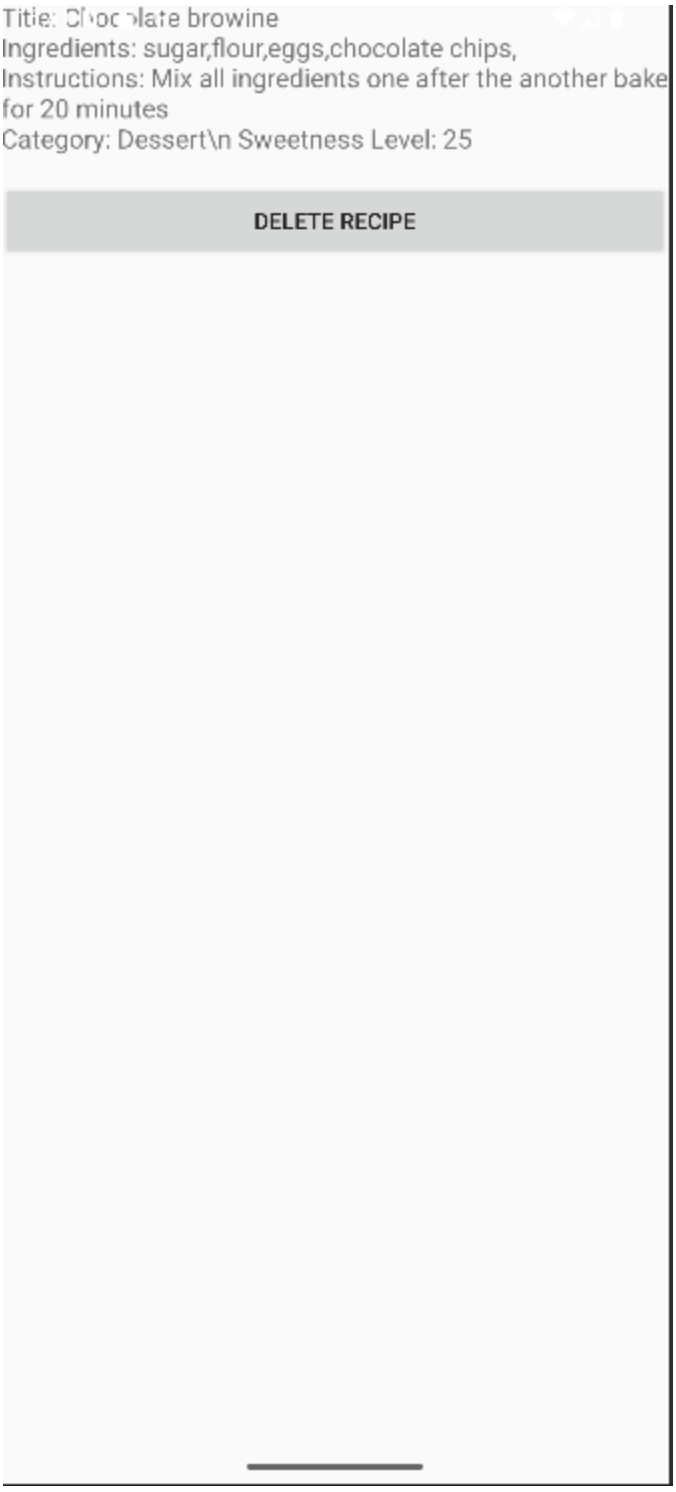

5:34
ADD/EDIT RECIPE
VIEW RECIPE

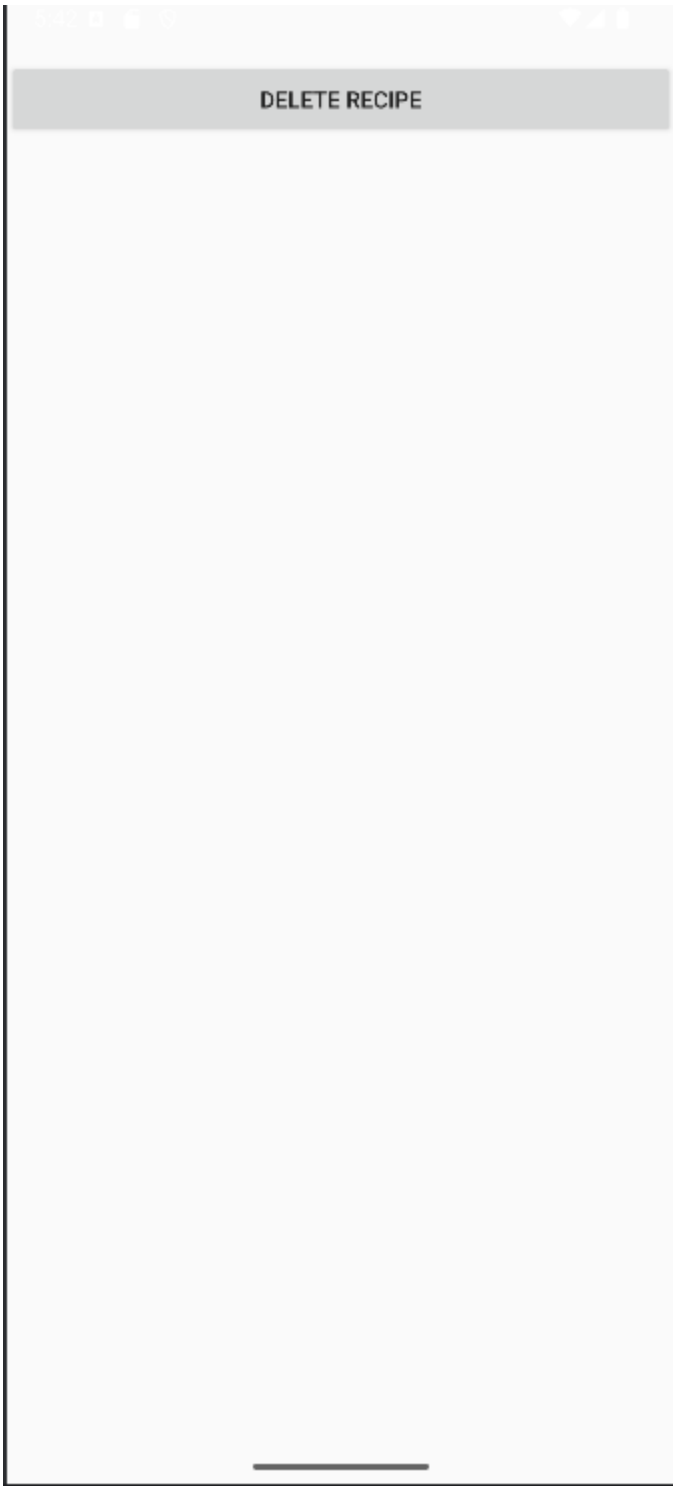

5:36
Title
Ingredients
Instructions
Dessert
Sweetness Level (For Desserts)
Preparation Time (For Main Courses)
SAVE RECIPE


5:39
Chocolate browine
sugar,flour,eggs,chocolate chips
Mix all ingredients one after the another bake for 20 minutes
Dessert
15
30
SAVE RECIPE


Title: Chocolate browine
Ingredients: sugar,flour,eggs,chocolate chips,
Instructions: Mix all ingredients one after the another bake for 20 minutes
Category: Dessert\n Sweetness Level: 15
DELETE RECIPE


5:40
Chocolate browine
sugar,flour,eggs,chocolate chips
Mix all ingredients one after the another bake for 20 minutes
Dessert
25
35
SAVE RECIPE


Title: Chocolate browine
Ingredients: sugar,flour,eggs,chocolate chips,
Instructions: Mix all ingredients one after the another bake for 20 minutes
Category: Dessert\n Sweetness Level: 25
DELETE RECIPE


DELETE RECIPE