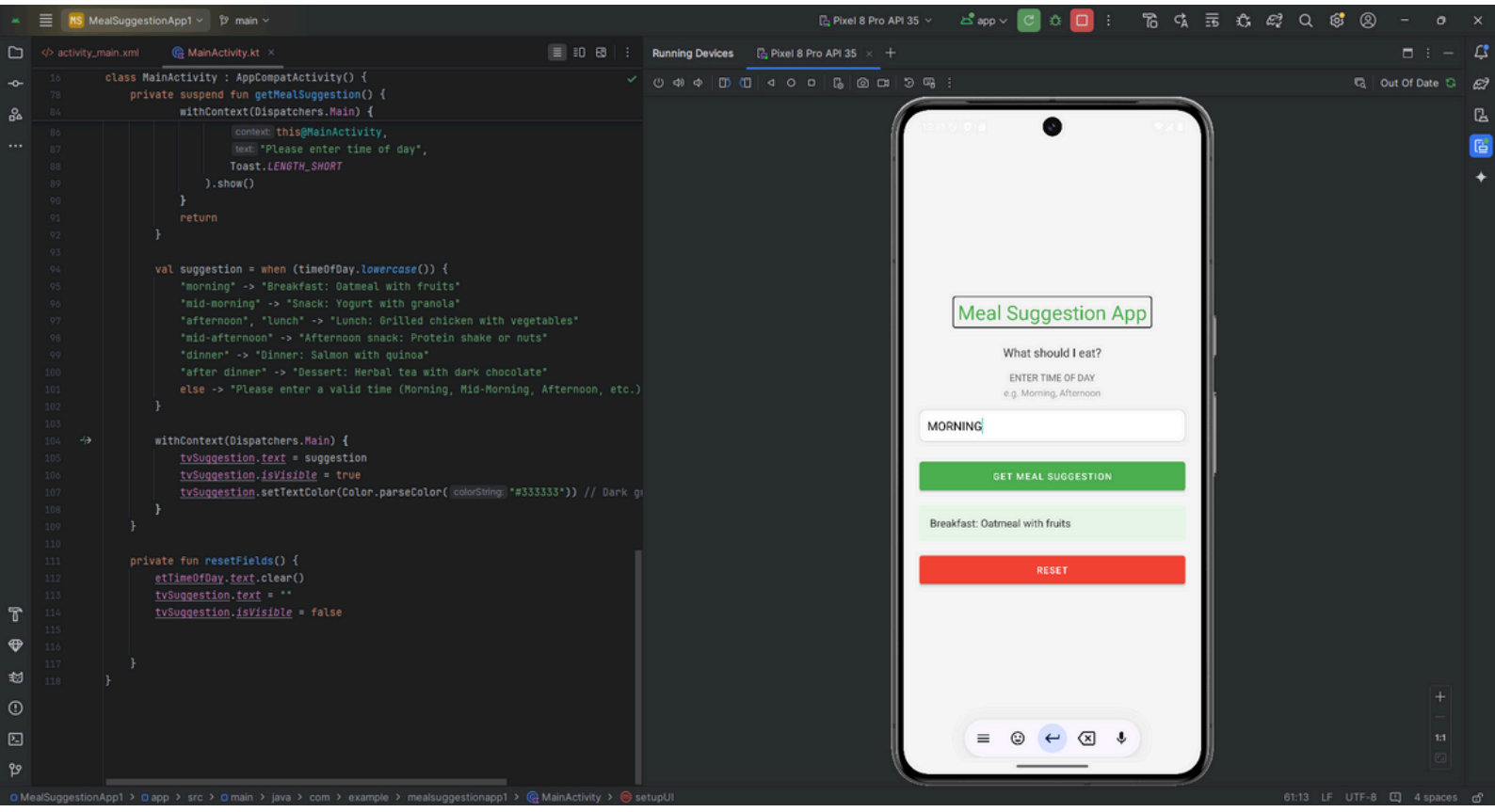
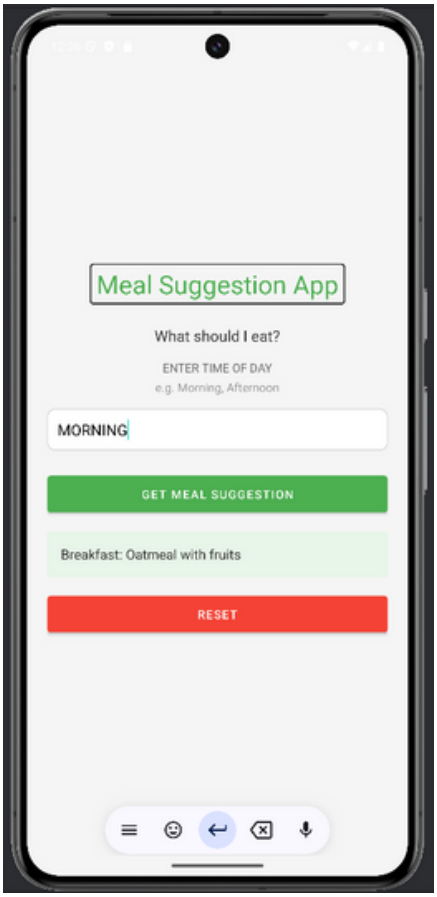
MODULE NAME:	MODULE CODE:
Introduction to Mobile Application Development	IMAD5112

STUDENT NAME:	STUDENT NUMBER
LESEGO SEBAKO	ST10493865
https://github.com/LesegoSebako/MealSuggestionApp1	

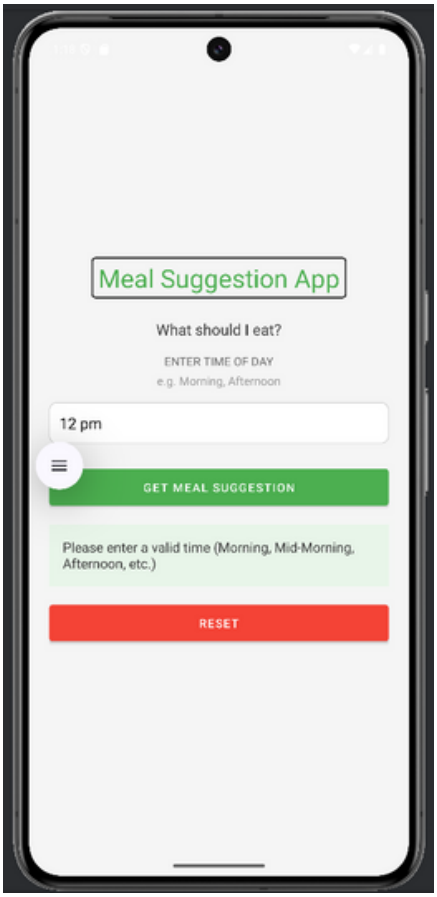
1.0 Meal Suggestion Application running on virtual Device - Pixel 8 Pro API 35



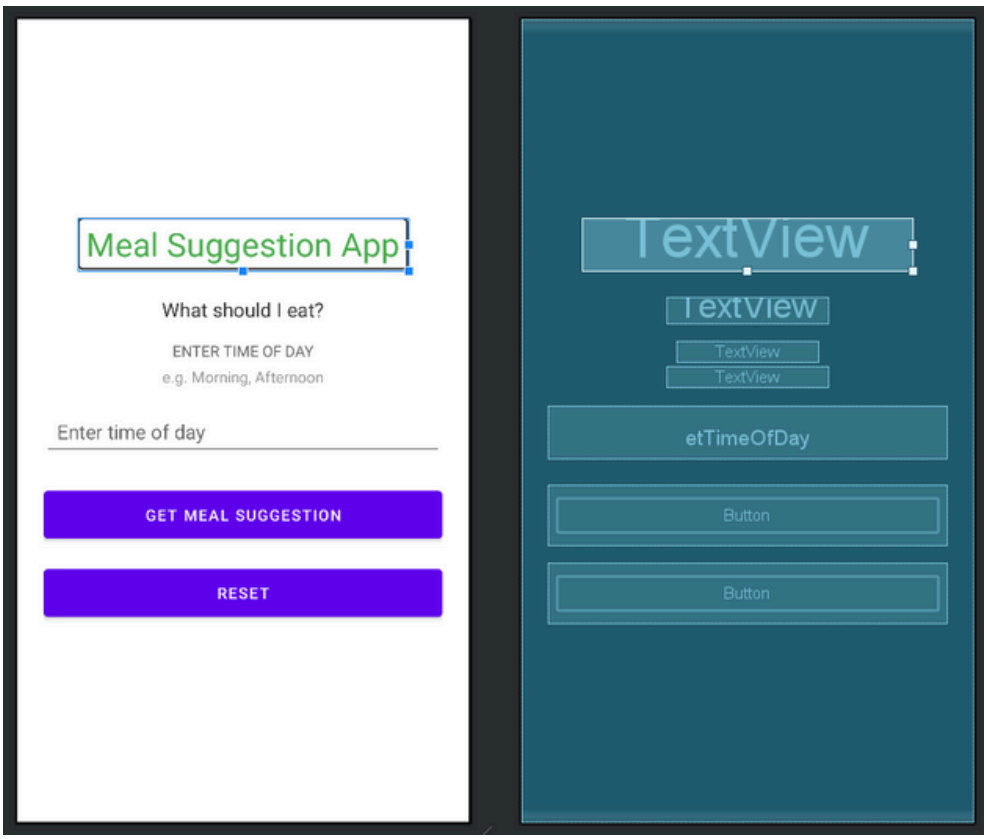
2.0 App Home page.



3.0 Input Error Handling



4.0 Layout - activity.main.xml , Design view



Technical Report: Meal Suggestion App

1. Introduction

This report explains the Meal Suggestion App, an Android application built using Kotlin in Android Studio. The app helps users decide what to eat based on the time of day.

Key Features Learned

- Basic Android UI design (TextViews, Buttons, EditText)
- Coroutines for background tasks
- Toast messages for error handling
- Color schemes for better UI
- Click listeners for user interaction

2. App Features

User Interface (UI) Components,Component & Purpose

- EditText (etTimeOfDay) - User types in time of day (e.g., "Morning")
- Button (btnGetSuggestion) - Triggers meal suggestion
- Button (btnReset) - Erases input and suggestion
- TextView (tvSuggestion) Displays the meal suggestion

Color Scheme

ElementColor CodePurpose

- Background - #F5F5F5 (Light Gray) - Clean, neutral look
- Get Suggestion Button - #4CAF50 (Green) - Positive action
- Reset Button - #F44336 (Red) - Warning/clear action
- Text Color - #333333 (Dark Gray) - Readable text

3. Key Methods & Classes Used

MainActivity.kt

This is the main class controlling the app.
Methods Used:

onCreate()

- Runs when the app starts.
- Sets up the layout (activity_main.xml).
- Calls initializeViews(), setupUI(), and setupClickListeners().

initializeViews()

- Connects UI elements (EditText, Button, TextView) to Kotlin code.

setupUI()

- Sets background color (#F5F5F5).
- Styles buttons (green for "Get Suggestion," red for "Reset").
- Hides the suggestion TextView initially.

setupClickListeners()

- btnGetSuggestion → Calls getMealSuggestion().
- btnReset → Calls resetFields().

getMealSuggestion()

- Uses coroutines (lifecycleScope.launch) to run in the background.
- Checks user input and shows a Toast error if empty.
- Uses when to match time of day with meal suggestions.

resetFields()

- Clears the EditText and TextView.

4. Error Handling in the App

Empty Input Check
  - When the user clicks "Get Meal Suggestion" without typing anything:

Invalid Input Handling
  - If the user types something unexpected (like "Nighttime"):

Displays a friendly suggestion about valid inputs

Manual Testing

- Empty input - Click button - Shows Toast error
- "Morning" → Click button - Suggests oatmeal
- "dinner" (lowercase) - Click button - Suggests salmon
- "Midday" (invalid) - Click button - Shows valid options
- Reset button - Clears all fields

5. GitHub Actions Automated Testing

Added basic workflow to catch build errors:
  - Automatically runs when code is pushed
  - Checks if the project compiles successfully
  - Verifies no syntax errors in Kotlin/XML files

---

6. GitHub Actions/Workflow.

Below are steps I undertook to create a Github repository, locally and online, as well creating and running workflow.

1. Set Up Git Desktop & Repository
2. Online GitHub Setup
3. Created a GitHub Actions Workflow
4. Pushing/Pulling Changes
5. Running & Monitoring Workflow

Key Concepts Learned

Actions & Purpose
Workflow - Automates process that builds/test the app
YAML File- format for configuring workflows
Push - Uploads local changes to GitHub
Pull - Downloads team members' changes
