Portfolio App: Android Portfolio Application

By

Sourabh Sinha (23SCSE2150018)

1. Introduction

The Android Portfolio Application is a dynamic platform designed to showcase my journey asa developer, emphasizing my skills, experiences, and projects in a visually appealing and interactive manner. The application consists of four main pages, each serving a unique purpose while collectively providing a comprehensive overview of my professional profile.

This portfolio app also serves as a project in my mobile development journey, allowing me to demonstrate my proficiency in Android development using Java in Android Studio. It was built with a focus on modularity, reusability of components, and user experience. Each section of the app, from the Home page to the Project section.

The **Home Page** acts as the gateway to the application, offering insights into who I am. It includes sections like my name, a brief **About Me** introduction that highlights my background and passion for technology, and a **Contact** section with direct links to my GitHub and LinkedInprofiles. This page sets the stage for further exploration.

The **Professional Experience Page** is dedicated to detailing my career journey, showcasing my certifications, freelancing projects, and internship experiences. With a user-friendly sliding feature, this page allows for smooth navigation through my qualifications and practical experiences, presenting a clear view of my professional growth and commitment to continuous learning.

The **Team Page**, named **Tech Titans**, highlights my team and our collaborative efforts. Each team member is presented in a block frame that details their roles and contributions. The sliding feature enhances the user experience, making it easy to explore our collective skills and the projects we have undertaken together.

Finally, the **Projects Page** focuses on the work I've completed, featuring block-sized images that represent each project. Users can click on these images to access detailed descriptions, including project goals, technologies used, and links to both the GitHub repository and the live version of each project. This page serves as a portfolio of my work, showcasing my technical expertise and the results of my efforts.

- **Multi-activity structure:** Includes Home [Contact], Experience, Team and Projectsactivities.
- **Project Display:** Each project includes descriptions, images, and GitHub links.
- **Dynamic Content:** Uses RecyclerView to display project details and skills in ascrollable, clean format.
- **Contact Link:** A simple link for users to reach out directly from the app.

2. Activities Details

The Android Portfolio App consists of five main activities: **Home Activity**, **Professional Experience**, **Tech Titans** and **Projects**. Each of these activities is designed to showcase a specific part of my professional portfolio, providing the user with detailed insights into my background, expertise, and completed projects.

I. Home Activity

The **Home Page** of my Android Portfolio Application is designed to provide a comprehensive overview, with three primary sections: **Name**, **About Me**, and **Contact**.

The **About Me** section offers a brief introduction, highlighting my background, skills, and passion for technology. It provides insight into who I am and what drives my work, making it easy for users to connect with me on a personal level.

In the **Contact** section, users can find direct links to my **GitHub** and **LinkedIn** profiles, allowing them to explore my code repositories and professional network. These links are interactive and guide users to my respective pages, making communication and potential opportunities for collaboration easy and accessible.

II. Professional Experience

The **Professional Experience** page of my Android Portfolio Application provides a detailed overview of my career journey, showcasing all my certifications, freelancing projects, and internship experiences. The page includes a **sliding feature** that smoothly scrolls down, creating an interactive and dynamic user experience.

At the top of the page, I highlight my most important **certifications**, demonstrating my commitment to continuous learning and professional growth. Each certification includes the issuing institution, the date it was received, and the specific skills I gained through the course.

As users scroll down, they will find my **freelancing projects**, where I detail the key projects, I have worked on independently. This section showcases my ability to handle diverse client projects, meet deadlines, and deliver quality results.

III. Tech Titans

The **third page** of my Android Portfolio Application is dedicated to my team, named **Tech Titans**. This page features a **sliding feature** that allows users to smoothly scroll through the details of each team member and their contributions.

Each team member’s information is presented in a **block frame**, making the content visually organized and easy to navigate. Inside each block, you will find details such as the team member’s name, their role within the team, and a brief description of their work and

contributions to our projects. This structure ensures that all team members are equally highlighted, giving users a clear understanding of the collective expertise within **Tech Titans**.

The sliding feature further enhances the page’s flow, ensuring users can engage with each team member’s profile in a streamlined manner. The page provides a detailed snapshot of the collaboration and skills that make **Tech Titans** an effective and innovative team.

IV. Projects

The **fourth and last page** of my Android Portfolio Application is dedicated to showcasing my **projects**. This page features a sequence of **block-sized project images**, each representing a different project I’ve worked on. The layout is clean and structured, making it easy for users to browse through the projects visually.

When a user clicks on one of the project images, it opens up a detailed view that provides a **project description**, offering insight into the goals, technologies used, and the overall outcome of the project. Along with the description, there are two interactive links:

1. **GitHub Code Link**: Clicking this link redirects users to the project's GitHub repository, where they can view the source code and explore the development processin detail.
2. **Live Link**: Clicking this link takes users to a live version of the project, hosted online. This allows them to interact with the fully functional project, giving them a hands-on experience of the work, I’ve done.

Both the GitHub and Live links are displayed within each project’s block frame, ensuring easy access and a seamless user experience. This page is designed to not only showcase the technical expertise behind each project but also provide potential collaborators and employers with away to explore the projects in-depth.

3. **Layout Details**

The layout of the Android Portfolio App was designed with both aesthetics and functionality in mind. Using XML layouts, I ensured that each activity is organized in a way that provides a smooth user experience, ensuring responsiveness across various devices. The design philosophy behind the app's layout is centred around simplicity and accessibility, allowing users to navigate through the app intuitively.

For the **Home Activity**, I used a LinearLayout with a vertical orientation to stack the introduction and navigation buttons. This layout gives the app a clean look and helps the user easily understand how to move from one section to another. I also added a Toolbar at the top to maintain consistency across different activities. The background is designed with a subtle gradient, giving the app a polished look. In Lastly, the **Contact Activity** was designed with a form-like layout, using EditText for input fields where users can send messages directly. I also added buttons for external links to my GitHub and LinkedIn, using Intent to navigate to these profiles when clicked.

Key Features:

- **Profile Image**: The top portion of the layout features a circular profile image using ImageView.
- **TextView**: The welcome text and short bio introduce the user to the app. It contains a large, bold font for the welcome message and a smaller, descriptive line below it that highlights the user's roles and skills.
- **Buttons**: Buttons link to other activities such as **Home**.

In the **Professional Experience Activity**, I used a ScrollView to allow users to scroll through my bio and career details without feeling overwhelmed by too much information on a single screen. The layout is divided into sections using LinearLayout for smooth content flow. I also added animation when the user scrolls down to keep the experience engaging. This layout is used to display details about the user's professional experience, such as internships, freelancing projects, certifications, and more. A sliding feature is used to allow users to scroll through their experiences.

Key Features:

- **ScrollView**: Enables vertical scrolling, so multiple experiences can be shown on a single screen.
- **TextViews**: Used to display the title, description, and dates for each professional experience.
- **LinearLayout**: Each experience is grouped in a block that can be styled for better UI (using background, padding, and margin).

For the **Team Activity**, I implemented a RecyclerView with a horizontal layout manager. This allows the user to scroll through my skills easily, with each skill represented by an icon and a short description. The use of CardView adds a material design touch to the UI, ensuring that the skills are displayed in a visually appealing manner. This layout is dedicated to the **Tech Titans** team. Each team member is shown in a block frame, with details such as name, role, and contribution.

Key Features:

- **CardView**: Displays each team member's photo.
- **RecyclerView**: Shows the name, role, and contribution for each member.
- **LinearLayout**: Team members are presented in a block format for neat alignment.

The **Projects Activity** employs a GridLayout to display clickable images of my projects in a grid format. When a project is clicked, a new activity is opened showing the project's details in a LinearLayout. The transitions between activities are smooth, giving the app a professional feel.

The layout for the **Projects** section shows block-sized images of each project. When clicked, the project details open with a description, GitHub link, and live demo link.

Key Features:

- **GridLayout**: Project thumbnails, clickable to open project details.
- **TextView**: Shows the project title, styled to stand out.
- **LinearLayout**: Each project is presented in a block, with a margin between blocks for visual clarity.

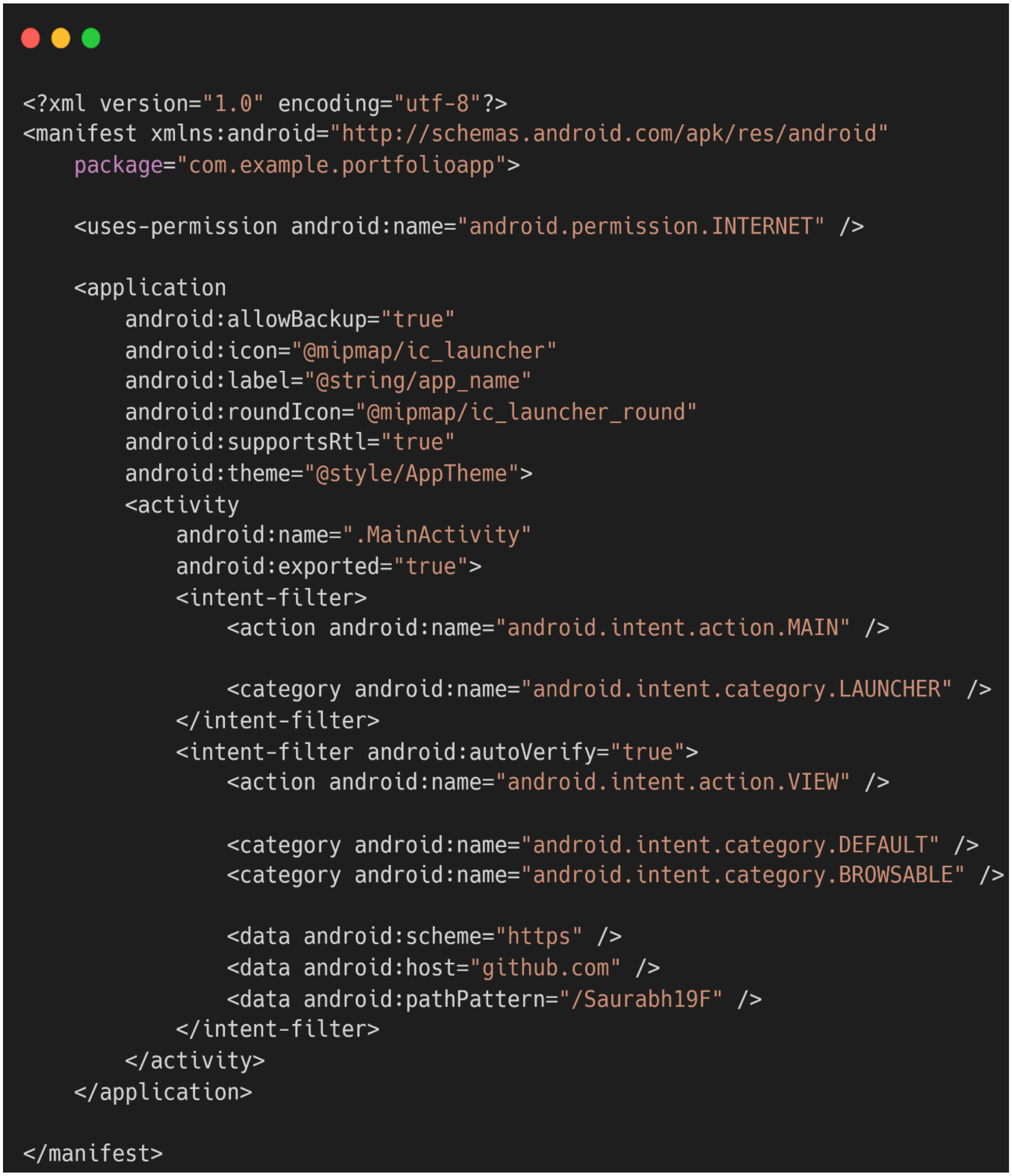
4. Steps

The development of the Android Portfolio App followed a systematic process, divided into several key steps:

- **Step 1: Planning and Design**: The first step involved planning the structure and design of the app. I created wireframes for each activity and mapped out the user flow to ensure that navigation would be intuitive. I focused on designing a responsive UI that would work well on different screen sizes and orientations. During this phase, I also decided on the color scheme, typography, and overall theme of the app.
- **Step 2: Setting Up the Project**: After finalizing the design, I set up the Android project in Android Studio. I chose Java as the programming language and used XML for defining the layout files. I also configured the project's settings, including the minimum SDK version to ensure compatibility with a wide range of devices.
- **Step 3: Developing the Home and About Me Activities**: I started by implementing the Home Activity, which serves as the main entry point of the app. I used `LinearLayout` and added buttons for navigating to other activities. Afterward, I developed the About Me Activity, where I included a `ScrollView` to display detailed information about my background and career.
- **Step 4: Implementing the Teams and Projects Activities**: For the Skills Activity, I used `RecyclerView` to display my technical skills in a horizontally scrolling list. Each skill is represented by a custom view containing an icon and a label. For the Projects Activity, I used a `GridLayout` to display images of my projects. Clicking on any project opens a detailed view of that project, which includes links to the GitHub repository and live demo.
- **Step 5: Adding the Contact Activity**: The Contact Activity was implemented with a form that allows users to send messages. I used `EditText` for user input and added buttons for opening external links to my social media profiles.
- **Step 6: Finalizing and Testing**: After implementing the core features, I focused on testing the app across different devices to ensure that it was responsive and bug-free. I also refined the UI to ensure consistency and added finishing touches such as animations and transitions.

5. **Manifest File Code**

Below is the code from the `AndroidManifest.xml` file, which defines the app's components and permissions:

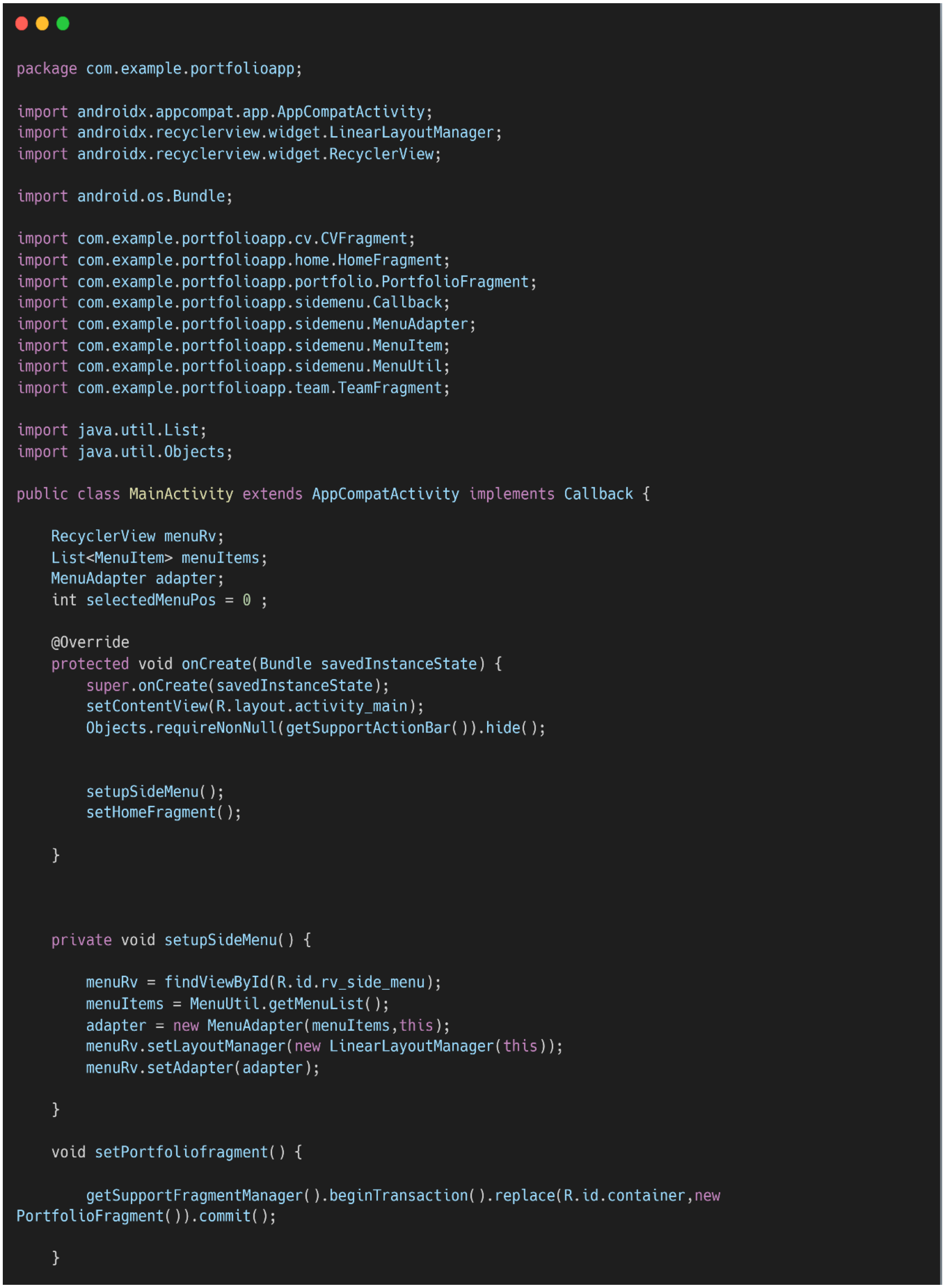


```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.portfolioapp">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
            <intent-filter android:autoVerify="true">
                <action android:name="android.intent.action.VIEW" />

                <category android:name="android.intent.category.DEFAULT" />
                <category android:name="android.intent.category.BROWSABLE" />

                <data android:scheme="https" />
                <data android:host="github.com" />
                <data android:pathPattern="/Saurabh19F" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

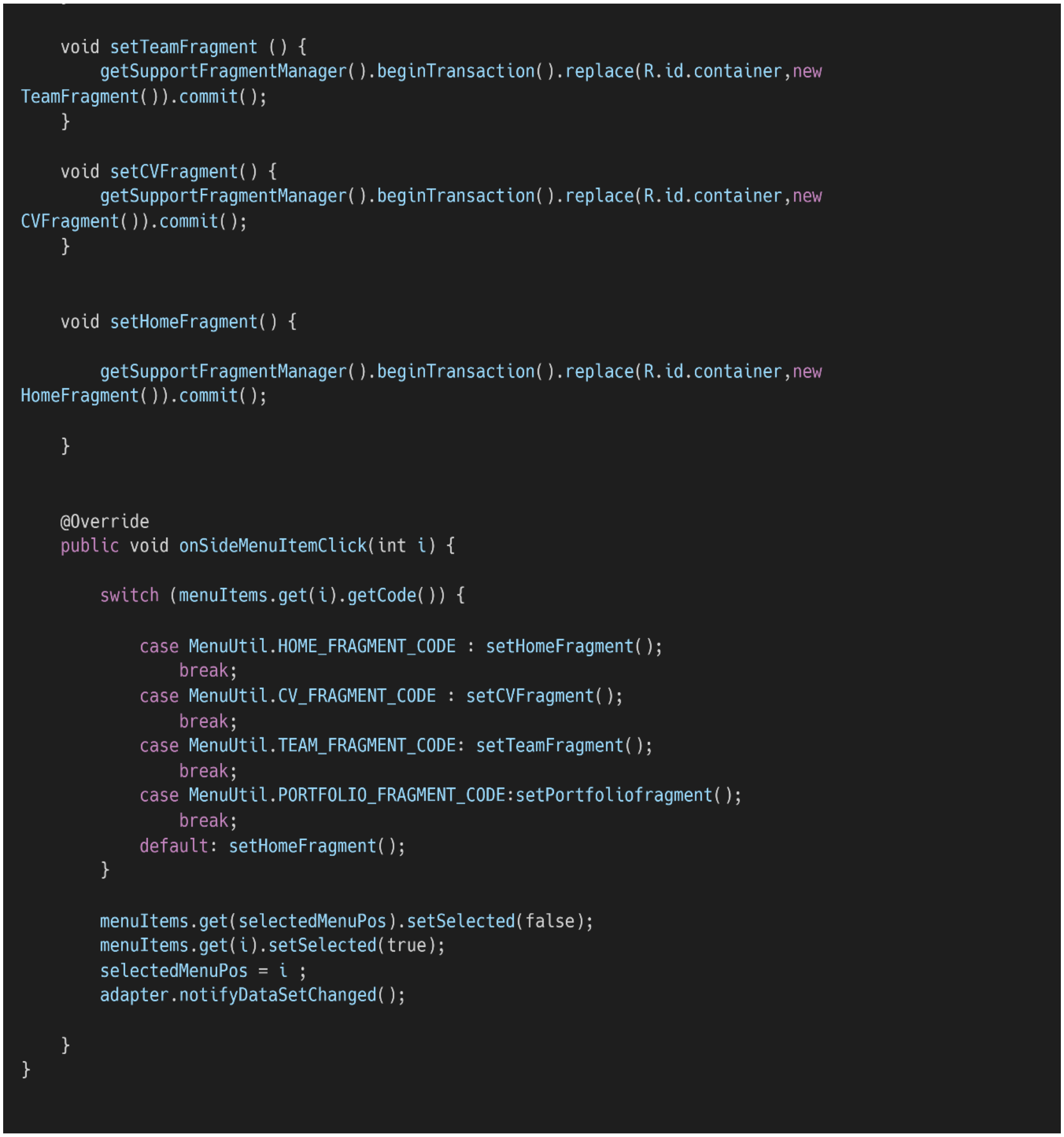
6. Any Other Supporting Files/Code

Java Code Files

MainActivity.java (or HomeActivity.java): The main Java or Kotlin file that controls the logic behind the Home Activity. You can include key methods or logic used for navigation, loading data, or handling UI elements.

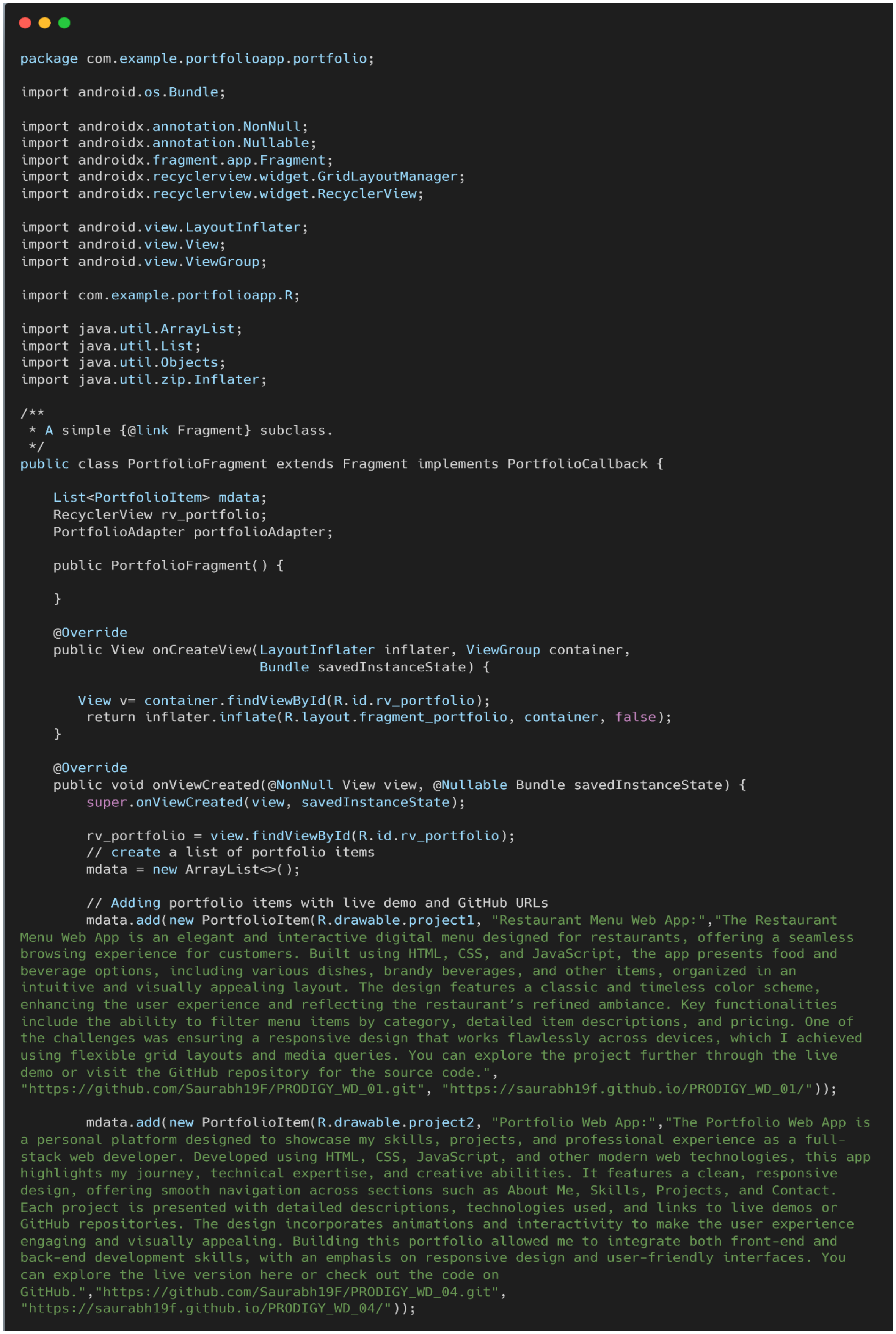


```
package com.example.portfolioapp;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.os.Bundle;

import com.example.portfolioapp.cv.CVFragment;
import com.example.portfolioapp.home.HomeFragment;
import com.example.portfolioapp.portfolio.PortfolioFragment;
import com.example.portfolioapp.sidemenu.Callback;
import com.example.portfolioapp.sidemenu.MenuAdapter;
import com.example.portfolioapp.sidemenu.MenuItem;
import com.example.portfolioapp.sidemenu.MenuUtil;
import com.example.portfolioapp.team.TeamFragment;

import java.util.List;
import java.util.Objects;

public class MainActivity extends AppCompatActivity implements Callback {

    RecyclerView menuRv;
    List<MenuItem> menuItems;
    MenuAdapter adapter;
    int selectedMenuPos = 0 ;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Objects.requireNonNull(getSupportActionBar()).hide();


        setupSideMenu();
        setHomeFragment();

    }



    private void setupSideMenu() {

        menuRv = findViewById(R.id.rv_side_menu);
        menuItems = MenuUtil.getMenuList();
        adapter = new MenuAdapter(menuItems,this);
        menuRv.setLayoutManager(new LinearLayoutManager(this));
        menuRv.setAdapter(adapter);

    }

    void setPortfoliofragment() {

        getSupportFragmentManager().beginTransaction().replace(R.id.container,new
PortfolioFragment()).commit();

    }
```

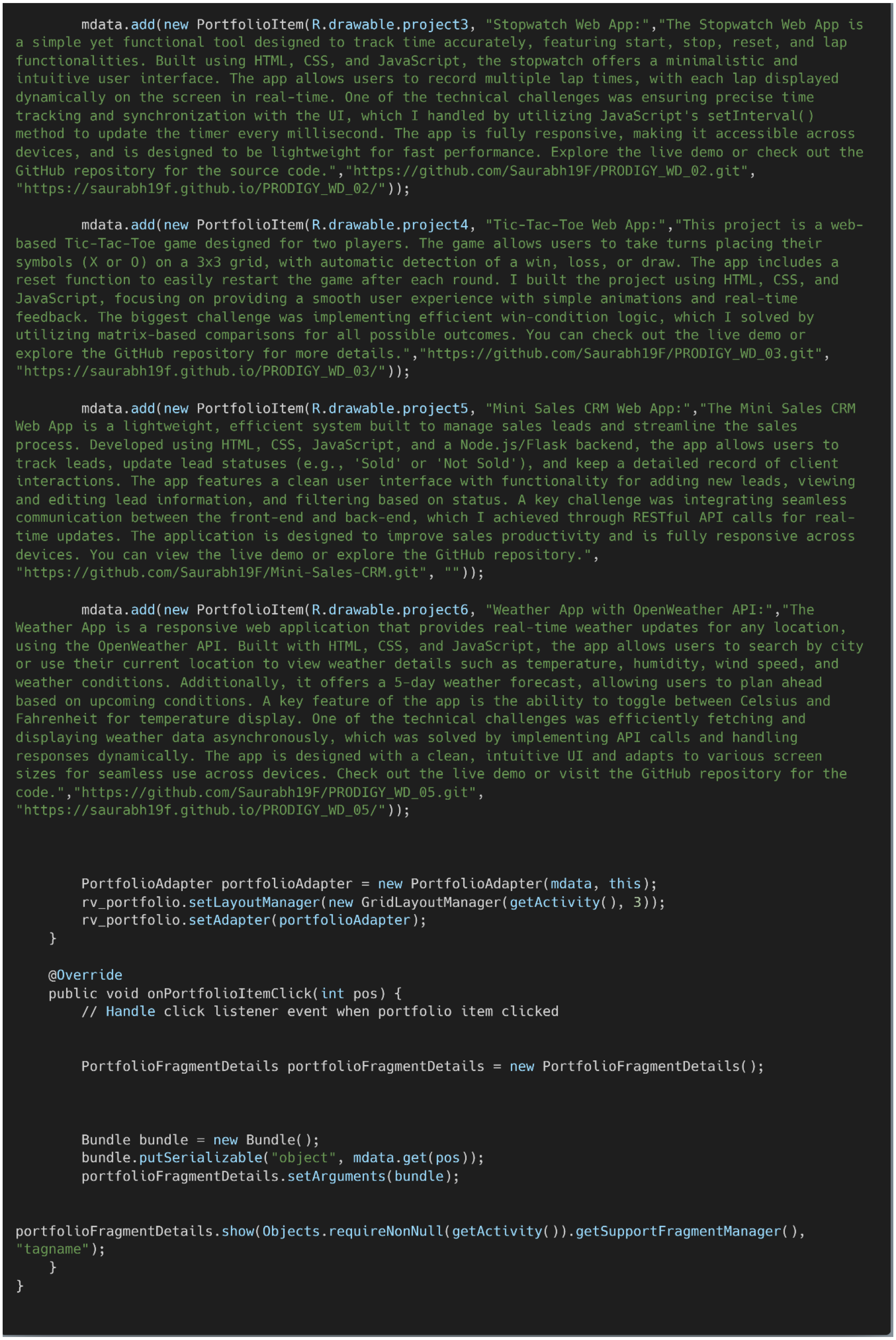


```
    void setTeamFragment () {
        getSupportFragmentManager().beginTransaction().replace(R.id.container,new
TeamFragment()).commit();
    }

    void setCVFragment() {
        getSupportFragmentManager().beginTransaction().replace(R.id.container,new
CVFragment()).commit();
    }


    void setHomeFragment() {

        getSupportFragmentManager().beginTransaction().replace(R.id.container,new
HomeFragment()).commit();

    }


    @Override
    public void onSideMenuItemClick(int i) {

        switch (menuItems.get(i).getCode()) {

            case MenuUtil.HOME_FRAGMENT_CODE : setHomeFragment();
                break;
            case MenuUtil.CV_FRAGMENT_CODE : setCVFragment();
                break;
            case MenuUtil.TEAM_FRAGMENT_CODE: setTeamFragment();
                break;
            case MenuUtil.PORTFOLIO_FRAGMENT_CODE:setPortfoliofragment();
                break;
            default: setHomeFragment();
        }

        menuItems.get(selectedMenuPos).setSelected(false);
        menuItems.get(i).setSelected(true);
        selectedMenuPos = i ;
        adapter.notifyDataSetChanged();

    }
}
```

ProjectsActivity.java: Include the logic used for displaying project details. This could involve handling click events for each project item, opening project details, and managing transitions.

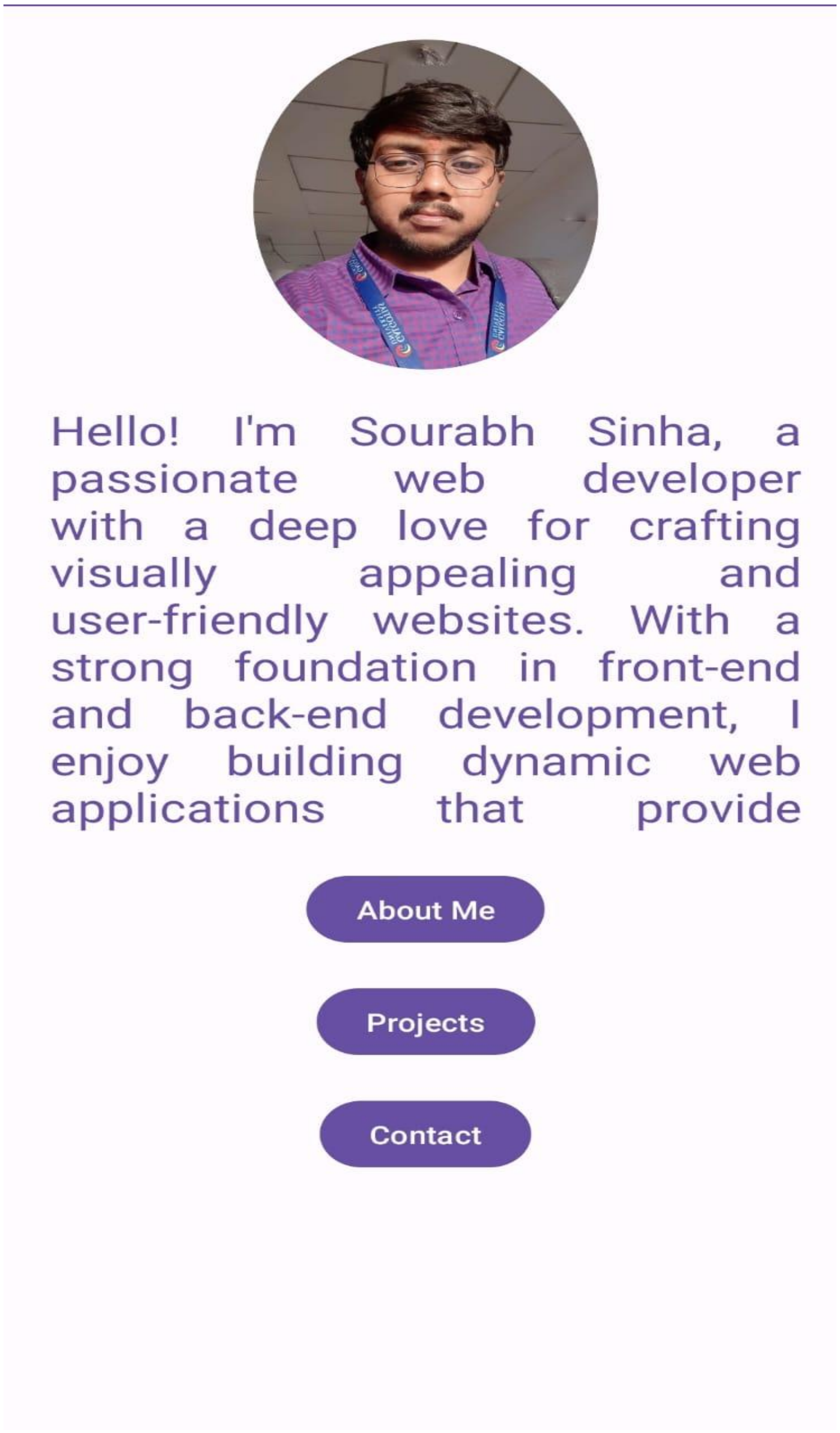


```
package com.example.portfolioapp.portfolio;

import android.os.Bundle;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import com.example.portfolioapp.R;

import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
import java.util.zip.Inflater;

/**
 * A simple {@link Fragment} subclass.
 */
public class PortfolioFragment extends Fragment implements PortfolioCallback {

    List<PortfolioItem> mdata;
    RecyclerView rv_portfolio;
    PortfolioAdapter portfolioAdapter;

    public PortfolioFragment() {

    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {

       View v= container.findViewById(R.id.rv_portfolio);
        return inflater.inflate(R.layout.fragment_portfolio, container, false);
    }

    @Override
    public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
        super.onViewCreated(view, savedInstanceState);

        rv_portfolio = view.findViewById(R.id.rv_portfolio);
        // create a list of portfolio items
        mdata = new ArrayList<>();

        // Adding portfolio items with live demo and GitHub URLs
        mdata.add(new PortfolioItem(R.drawable.project1, "Restaurant Menu Web App:","The Restaurant
Menu Web App is an elegant and interactive digital menu designed for restaurants, offering a seamless
browsing experience for customers. Built using HTML, CSS, and JavaScript, the app presents food and
beverage options, including various dishes, brandy beverages, and other items, organized in an
intuitive and visually appealing layout. The design features a classic and timeless color scheme,
enhancing the user experience and reflecting the restaurant’s refined ambiance. Key functionalities
include the ability to filter menu items by category, detailed item descriptions, and pricing. One of
the challenges was ensuring a responsive design that works flawlessly across devices, which I achieved
using flexible grid layouts and media queries. You can explore the project further through the live
demo or visit the GitHub repository for the source code.",
"https://github.com/Saurabh19F/PRODIGY_WD_01.git", "https://saurabh19f.github.io/PRODIGY_WD_01/"));

        mdata.add(new PortfolioItem(R.drawable.project2, "Portfolio Web App:","The Portfolio Web App is
a personal platform designed to showcase my skills, projects, and professional experience as a full-
stack web developer. Developed using HTML, CSS, JavaScript, and other modern web technologies, this app
highlights my journey, technical expertise, and creative abilities. It features a clean, responsive
design, offering smooth navigation across sections such as About Me, Skills, Projects, and Contact.
Each project is presented with detailed descriptions, technologies used, and links to live demos or
GitHub repositories. The design incorporates animations and interactivity to make the user experience
engaging and visually appealing. Building this portfolio allowed me to integrate both front-end and
back-end development skills, with an emphasis on responsive design and user-friendly interfaces. You
can explore the live version here or check out the code on
GitHub.","https://github.com/Saurabh19F/PRODIGY_WD_04.git",
"https://saurabh19f.github.io/PRODIGY_WD_04/"));
```



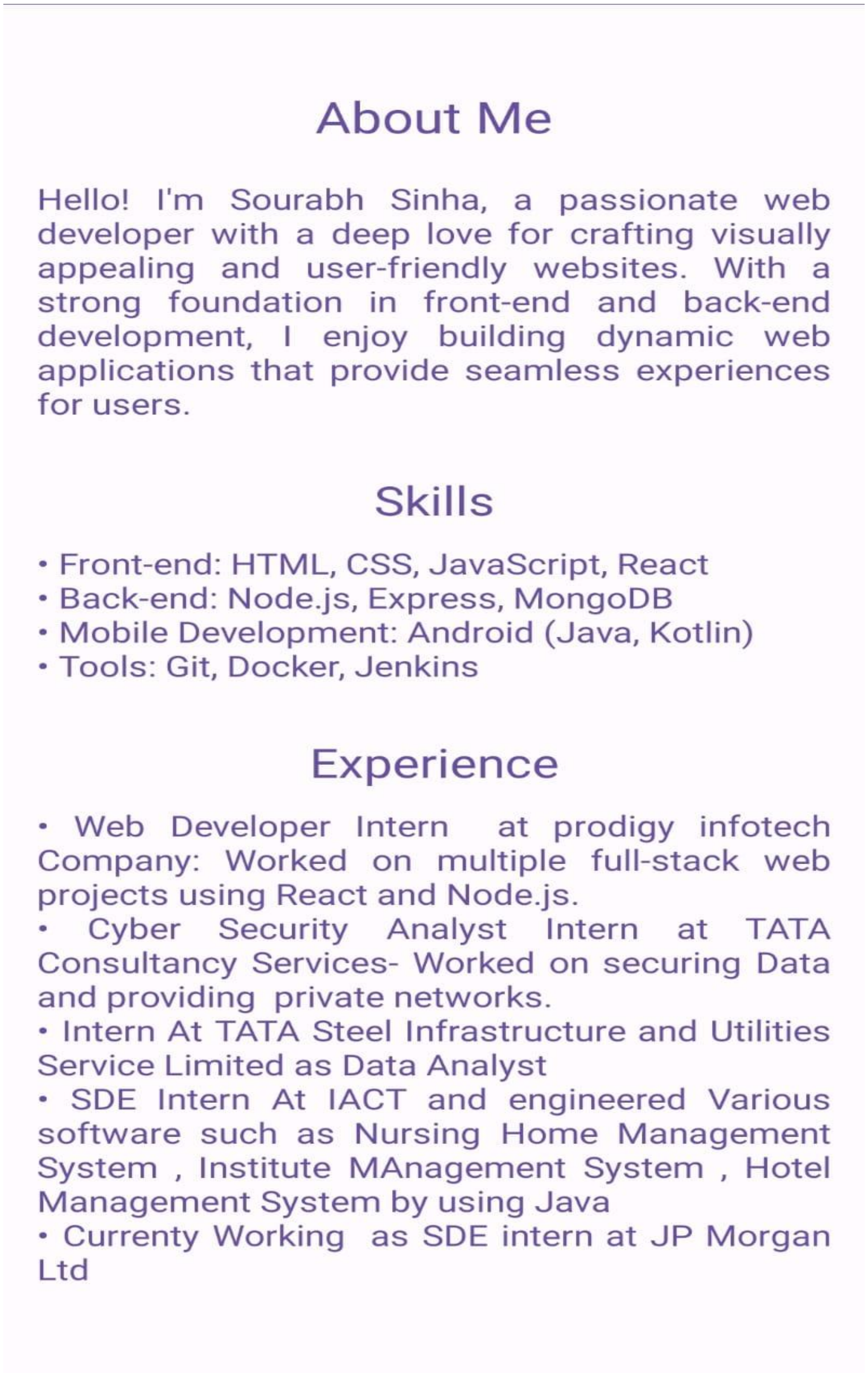
```
        mdata.add(new PortfolioItem(R.drawable.project3, "Stopwatch Web App:","The Stopwatch Web App is
a simple yet functional tool designed to track time accurately, featuring start, stop, reset, and lap
functionalities. Built using HTML, CSS, and JavaScript, the stopwatch offers a minimalistic and
intuitive user interface. The app allows users to record multiple lap times, with each lap displayed
dynamically on the screen in real-time. One of the technical challenges was ensuring precise time
tracking and synchronization with the UI, which I handled by utilizing JavaScript's setInterval()
method to update the timer every millisecond. The app is fully responsive, making it accessible across
devices, and is designed to be lightweight for fast performance. Explore the live demo or check out the
GitHub repository for the source code.","https://github.com/Saurabh19F/PRODIGY_WD_02.git",
"https://saurabh19f.github.io/PRODIGY_WD_02/"));

        mdata.add(new PortfolioItem(R.drawable.project4, "Tic-Tac-Toe Web App:","This project is a web-
based Tic-Tac-Toe game designed for two players. The game allows users to take turns placing their
symbols (X or O) on a 3x3 grid, with automatic detection of a win, loss, or draw. The app includes a
reset function to easily restart the game after each round. I built the project using HTML, CSS, and
JavaScript, focusing on providing a smooth user experience with simple animations and real-time
feedback. The biggest challenge was implementing efficient win-condition logic, which I solved by
utilizing matrix-based comparisons for all possible outcomes. You can check out the live demo or
explore the GitHub repository for more details.","https://github.com/Saurabh19F/PRODIGY_WD_03.git",
"https://saurabh19f.github.io/PRODIGY_WD_03/"));

        mdata.add(new PortfolioItem(R.drawable.project5, "Mini Sales CRM Web App:","The Mini Sales CRM
Web App is a lightweight, efficient system built to manage sales leads and streamline the sales
process. Developed using HTML, CSS, JavaScript, and a Node.js/Flask backend, the app allows users to
track leads, update lead statuses (e.g., 'Sold' or 'Not Sold'), and keep a detailed record of client
interactions. The app features a clean user interface with functionality for adding new leads, viewing
and editing lead information, and filtering based on status. A key challenge was integrating seamless
communication between the front-end and back-end, which I achieved through RESTful API calls for real-
time updates. The application is designed to improve sales productivity and is fully responsive across
devices. You can view the live demo or explore the GitHub repository.",
"https://github.com/Saurabh19F/Mini-Sales-CRM.git", ""));

        mdata.add(new PortfolioItem(R.drawable.project6, "Weather App with OpenWeather API:","The
Weather App is a responsive web application that provides real-time weather updates for any location,
using the OpenWeather API. Built with HTML, CSS, and JavaScript, the app allows users to search by city
or use their current location to view weather details such as temperature, humidity, wind speed, and
weather conditions. Additionally, it offers a 5-day weather forecast, allowing users to plan ahead
based on upcoming conditions. A key feature of the app is the ability to toggle between Celsius and
Fahrenheit for temperature display. One of the technical challenges was efficiently fetching and
displaying weather data asynchronously, which was solved by implementing API calls and handling
responses dynamically. The app is designed with a clean, intuitive UI and adapts to various screen
sizes for seamless use across devices. Check out the live demo or visit the GitHub repository for the
code.","https://github.com/Saurabh19F/PRODIGY_WD_05.git",
"https://saurabh19f.github.io/PRODIGY_WD_05/"));



        PortfolioAdapter portfolioAdapter = new PortfolioAdapter(mdata, this);
        rv_portfolio.setLayoutManager(new GridLayoutManager(getActivity(), 3));
        rv_portfolio.setAdapter(portfolioAdapter);
    }

    @Override
    public void onPortfolioItemClick(int pos) {
        // Handle click listener event when portfolio item clicked


        PortfolioFragmentDetails portfolioFragmentDetails = new PortfolioFragmentDetails();



        Bundle bundle = new Bundle();
        bundle.putSerializable("object", mdata.get(pos));
        portfolioFragmentDetails.setArguments(bundle);


portfolioFragmentDetails.show(Objects.requireNonNull(getActivity()).getSupportFragmentManager(),
"tagname");
    }
}
```

Screenshot:



Hello! I'm Sourabh Sinha, a passionate web developer with a deep love for crafting visually appealing and user-friendly websites. With a strong foundation in front-end and back-end development, I enjoy building dynamic web applications that provide

Screenshot:



About Me

Hello! I'm Sourabh Sinha, a passionate web developer with a deep love for crafting visually appealing and user-friendly websites. With a strong foundation in front-end and back-end development, I enjoy building dynamic web applications that provide seamless experiences for users.

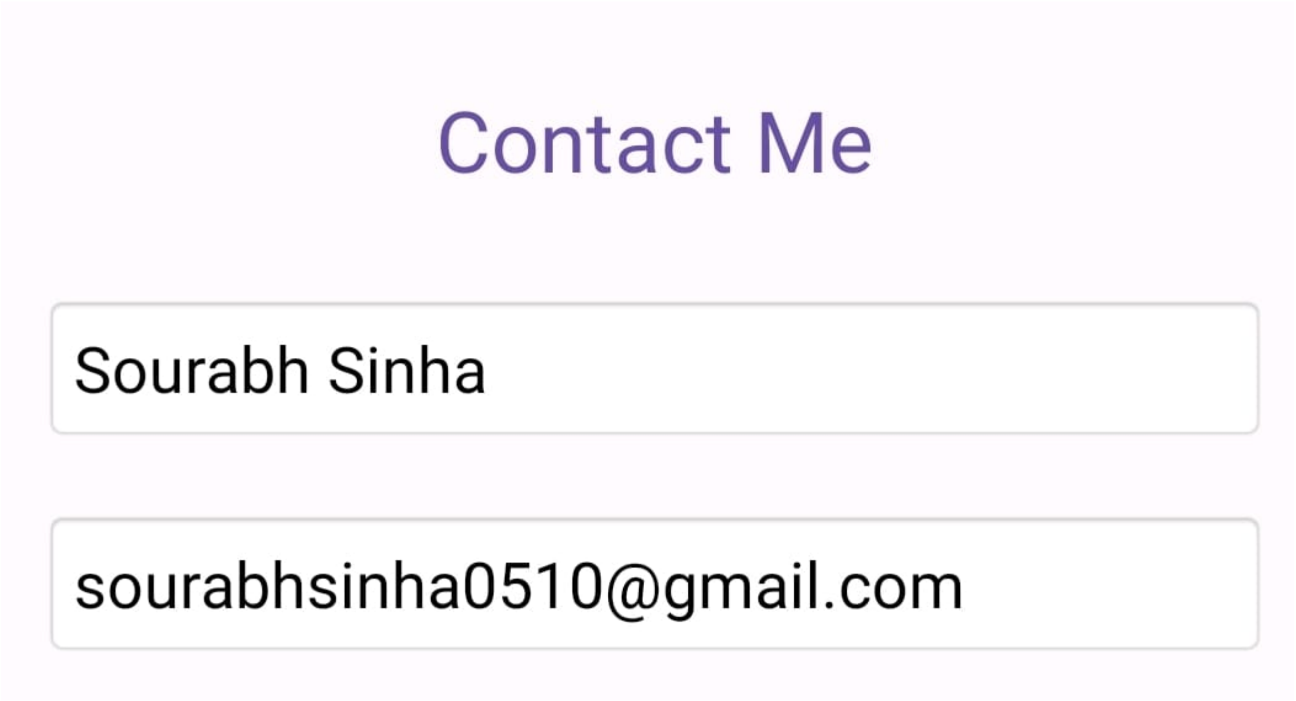
Skills

- Front-end: HTML, CSS, JavaScript, React
- Back-end: Node.js, Express, MongoDB
- Mobile Development: Android (Java, Kotlin)
- Tools: Git, Docker, Jenkins

Experience

- Web Developer Intern at prodigy infotech Company: Worked on multiple full-stack web projects using React and Node.js.
- Cyber Security Analyst Intern at TATA Consultancy Services- Worked on securing Data and providing private networks.
- Intern At TATA Steel Infrastructure and Utilities Service Limited as Data Analyst
- SDE Intern At IACT and engineered Various software such as Nursing Home Management System , Institute MAnagement System , Hotel Management System by using Java
- Currenty Working as SDE intern at JP Morgan Ltd

Screenshot:



Contact Me

Sourabh Sinha

sourabhsinha0510@gmail.com

8. **Setup Instructions**

Prerequisites

- Android Studio installed on your development machine.
- Basic understanding of Java and Android development.

Installation

Clone this repository:

https://github.com/Saurabh19F/Android-Portfolio-App.git

Open the project in Android Studio:

- Go to Android Studio and select "Open an Existing Project."
- Navigate to the folder where you cloned the project.

Build and run:

- Click on the "Run" button or press Shift + F10 to build and run the app on an emulator or connected device.
- Usage
- Navigate through the sections: Home [Contact], Experience, Team and Projects.
- Browse the Projects section for descriptions of each project, with images and GitHub links.
- Contact me directly via the Contact Link$.

Contact Information

For any inquiries, feel free to reach out:

- Email: sourabhsinha0510@gmail.com
- LinkedIn:/sourabh-sinha
- GitHub: **sourabhsinha0510**