RECRUTO: MOBILE APPLICATION FOR PART – TIME JOB SEEKERS

A MINI PROJECT REPORT

Submitted by

RAJA HAREESH R G (2116220701214)

in partial fulfillment for the course

CS19611 – MOBILE APPLICATION DEVELOPMENT LABORATORY

of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE RAJALAKSHMI NAGAR THANDALAM CHENNAI – 602 105

MAY 2025

RAJALAKSHMI ENGINEERING COLLEGE
CHENNAI - 602105

BONAFIDE CERTIFICATE

Certified that this project report **"RECRUTO : MOBILE APP FOR PART TIME JOB SEEKERES"** is the bonafide work of **"RAJA HAREESH R G (2116220701214)"** who carried out the project work (CS19611-Mobile Application Development Laboratory) under my supervision.

SIGNATURE
Saravana Gokul M.E(CSE)
Assistant Professor/SG
Rajalakshmi Engineering College,
Chennai-602 105

Submitted to Project Viva-Voce Examination held on ______________________

Internal Examiner **External Examiner**

TABLE OF CONTENTS

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S.Meganathan, B.E, F.I.E.,** our Vice Chairman **Mr. Abhay Meganathan, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) Thangam Meganathan, Ph.D.,** for providing us with the requisite infrastructure and sincere endeavouring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N.Murugesan, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P.Kumar, M.E., Ph.D.,** Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Mr. Saravana Gokul G, M.E., Assistant Professor (SG)** , Department of Computer Science and Engineering, Rajalakshmi Engineering College for their valuable guidance throughout the course of the project. We are very glad to thank our Project Coordinator, **Mr. Saravana Gokul G, M.E.,** Assistant Professor (SG), Department of Computer Science and Engineering for his useful tips during our review to build our project.

RAJA HAREESH R G (2116220701214)

ABSTRACT

Recruto is an Android application developed to simplify the process of finding and posting part-time job opportunities. Built using **Android Studio** with **Kotlin**, the app focuses on offering a user-friendly interface that enables individuals and small businesses to connect efficiently over short-term work requirements. Recruto is designed with a clean UI and modular architecture to ensure smooth navigation and interaction.

The application allows **any user** to **post job listings** or **apply for existing jobs**, making it flexible for both job seekers and job providers. Users can enter job details such as role, description, location, and category. Applicants can view available jobs, submit applications, and track their status. Meanwhile, employers can review applications for jobs they have posted. Data handling is implemented using local storage or Firebase (depending on expansion plans), with basic validation and feedback prompts to guide users.

Key features include a **login system**, **job posting and search**, **application tracking**, and **application review dashboards**. Recruto provides a lightweight, accessible, and effective solution for managing part-time job activities on a mobile platform, making it an ideal choice for students, local businesses, and freelancers seeking flexible work options.

LIST OF FIGURES

1 INTRODUCTION

1.1 GENERAL

In today’s fast-moving world, the demand for flexible and short-term employment has increased significantly, especially among students, freelancers, and local businesses. Traditional job portals mostly focus on full-time or long-term corporate roles, which leaves a gap in the market for part-time, freelance, and local job opportunities. There is a growing need for a lightweight, fast, and mobile-first solution that connects individuals who need quick jobs with employers seeking immediate help.

Recruto is an Android application developed using Kotlin in Android Studio that addresses this exact need. It is a part-time and freelance job portal app designed to simplify the hiring process for both job seekers and local employers. Whether it’s a shop owner looking for a weekend assistant or a student looking for a onetime gig, Recruto enables users to post, find, apply to, and manage job listings easily from their smartphones.

The app features a simple login system, a clean home interface, job search and job posting modules, and options to apply for or review job applications. In its early version, the logic is hardcoded for demonstration purposes, allowing users to log in with basic validation. However, the structure is kept modular to allow future enhancements like Firebase integration or cloud-based backend support.

By focusing primarily on part-time and freelance work, Recruto fills a practical gap in the market. Its minimal design, ease of use, and quick job management features make it ideal for real-life, day-to-day hiring needs, especially in local communities where formal platforms fall short.

1.2 OBJECTIVE

The primary objective of the Recruto Android application is to provide a simple, fast, and efficient platform for connecting local employers with part-time job seekers. Unlike traditional job portals that cater mainly to full-time corporate positions, Recruto is focused on supporting flexible, temporary, and freelance job opportunities that are common in local communities.

The application is designed to achieve the following goals:

- To develop a mobile-friendly platform using Kotlin and Android Studio that allows users to post and search for part-time or freelance jobs easily.
- To enable job seekers to browse available jobs and apply for positions directly through the app with minimal steps.
- To provide employers with the ability to post job listings and review applications without the need for complex backend systems.
- To implement a clean and intuitive user interface with smooth navigation and basic authentication for demonstration purposes.
- To create a scalable, modular application structure that can support future enhancements such as cloud database integration, push notifications, and location-based filtering.

Recruto aims to streamline the hiring process for local businesses, students, and freelancers by focusing on accessibility, simplicity, and speed.

1.3 EXISTING SYSTEM

Several popular platforms like LinkedIn, Naukri, Indeed, and Glassdoor currently dominate the online job market. These platforms are highly effective for full-time and long-term employment, particularly in corporate and professional sectors. For example, LinkedIn focuses heavily on professional networking, resume building, and formal job applications. While powerful, it is not suited for quick, part-time, or informal job opportunities due to its complexity and corporate orientation.

Some applications, such as GigIndia and Qjobs, were developed specifically to target the part-time and gig job market. However, these platforms struggled with issues like unreliable job postings, lack of proper employer verification, and inconsistent user experience. For instance, GigIndia, once a promising startup for freelance and student jobs, faced challenges in ensuring payment security and job authenticity, leading to its eventual decline.

Moreover, many existing systems are designed with web-first interfaces or require lengthy registration processes, which are not ideal for small businesses or individuals seeking quick hires. These limitations make the current ecosystem less accessible to users looking for short-term, flexible, and local job opportunities.

Thus, there is a clear gap in the market for a simple, trustworthy, and mobile-first platform dedicated to part-time and freelance jobs—a gap that Recruto aims to fill effectively.

1.4 PROPOSED SYSTEM

The proposed system, **Recruto**, is designed to address the gaps and limitations found in existing job portals by providing a **simple, efficient, and mobile-first platform** dedicated to part-time, freelance, and short-term employment. Unlike large-scale job platforms like **LinkedIn** or **Naukri**, which focus on full-time, corporate roles, The Recruto mobile app is specifically tailored for **local businesses** and **individuals** seeking quick hires for part-time or temporary positions.

Key features of the proposed system include:

1. **Simple Job Postings**: Employers can easily post part-time job opportunities, including brief job descriptions, timings, and pay. This removes the need for lengthy application forms or complicated procedures.

2. **Quick Job Search**: Job seekers can quickly search through available positions, applying with minimal steps. This makes it ideal for students, freelancers, or individuals seeking temporary work.

3. **User-Friendly Interface**: The app offers a **clean, intuitive UI** designed to make job searching and posting seamless on smartphones. With features like animated transitions, modern fonts, and smooth navigation, it creates an engaging user experience.

4. **Basic Authentication**: The app uses simple login functionality, with hardcoded validation for **quick demonstration purposes**, allowing easy access without complex account setups.

5. **Focused on Part-Time and Freelance Jobs**: The primary focus is on **part-time**, **freelance**, and **short-term tasks**, which are not adequately catered to by larger job portals.
6. **Scalability and Future Enhancements**: The app is built with a modular structure that can be scaled to include features like real-time databases (e.g., Firebase), job notifications, and geolocation-based job filtering in future versions.

By focusing specifically on part-time and freelance job markets, **Recruto** addresses a clear need for a **simpler, faster, and more localized hiring solution**. It serves as a practical tool for businesses and individuals who need flexible work arrangements without the complexity of traditional job portals.

2 LITERATURE REVIEW

2.1 GENERAL

The demand for part-time, freelance, and gig work has surged significantly in recent years, driven by shifts in both consumer behavior and the evolving nature of employment. The traditional 9-to-5 model has given way to more flexible work arrangements, with many workers seeking temporary or short-term jobs to balance other commitments. This change in employment patterns has led to the rise of platforms specifically targeting part-time, freelance, and gig workers.

Growth of the Gig Economy

According to a study by McKinsey Global Institute (2016), approximately 2030% of the working-age population in the United States participates in some form of independent work, with many seeking flexible or part-time job opportunities. The gig economy has transformed the way individuals find work, with apps like Uber, TaskRabbit, and Fiverr providing workers with flexible earning opportunities. These platforms have popularized the idea of short-term, taskbased employment, allowing both job seekers and employers to engage in transactions without the need for long-term commitments.

However, part-time job platforms, especially those that cater to local hiring needs, have seen limited success. Platforms such as GigIndia aimed to target students and part-time workers but faced difficulties in scaling due to challenges like unreliable job postings, issues with employer verification, and inconsistent payment systems. Research by Kaufman & Lichtenstein (2020) highlights that many part-time job apps fail to establish trust and reliability, which are essential for user retention and success in the gig economy.

Challenges in Existing Job Portals

Mainstream job portals like LinkedIn and Naukri are focused on full-time positions and tend to have features suited for corporate job seekers and recruiters. A study by Sharma et al. (2021) found that LinkedIn is widely used for professional networking and long-term career opportunities, but it is not optimized for part-time or informal work. The study indicates that LinkedIn's overwhelming features and formal application processes make it difficult for users to search for or post part-time and freelance jobs efficiently.

Additionally, Indeed and Glassdoor also cater to a wide array of jobs, but these platforms are primarily focused on the traditional job market and are not particularly effective for part-time, temporary, or local work. As a result, there exists a gap in the market for specialized platforms that focus solely on part-time and freelance jobs, especially for local communities and small businesses.

Mobile-First Job Platforms

With the rise of mobile usage, many job seekers now prefer to search for jobs via smartphones instead of desktops. Mobile-first platforms like Upwork, Fiverr, and Freelancer.com have capitalized on this trend by offering simple, user-friendly apps that allow users to quickly search for jobs, apply, and communicate with employers. According to Nassif (2021), mobile-first job platforms have a higher engagement rate and are more likely to attract younger users, especially students and part-time workers who rely on smartphones for daily activities.

While these apps have succeeded in the freelance and gig sectors, they are often overloaded with features and can be overwhelming for users simply seeking quick, part-time employment opportunities. Recruto, by contrast, aims to streamline the process, focusing on ease of use, local job postings, and fast hiring without the need for complex registrations or professional profiles.

The Need for Specialized Local Job Platforms

A study by Mello (2022) highlights the need for more localized, specialized job platforms that cater to part-time workers in small businesses, restaurants, and shops. These businesses often face challenges in hiring workers on traditional platforms that focus more on full-time roles. Local businesses require a solution that allows them to quickly post jobs, connect with nearby job seekers, and manage applications on the go.

Recruto aims to fill this gap by offering a lightweight, efficient mobile platform designed specifically for part-time, temporary, and local job opportunities, serving both job seekers and employers with simplicity and speed.

3 SYSTEM DESIGN

3.1 GENERAL

System design is a crucial stage in the development of the Recruto mobile application. It defines how different components of the system interact, how data flows, and how users engage with the features of the app. Recruto is focused on providing a simple and effective platform for posting and applying for part-time jobs.

This chapter presents the system design using three key visual models: the System Flow Diagram, the Architecture Diagram, and the Use Case Diagram. These diagrams illustrate both the technical structure and the user interaction flow, helping to understand how the system functions internally and externally. The goal of this design is to ensure a smooth user experience for both job seekers and job providers, allowing any user to post or apply for jobs, check application status, and manage job postings easily.

3.1.1 SYSTEM FLOW DIAGRAM

The **System Flow Diagram** outlines the sequence of actions triggered when a user interacts with the app, such as adding, viewing, and searching for diary entries. It ensures that each module—entry creation, search, list view, and contact management—is invoked in the correct order with minimal user input. The system is designed to handle operations efficiently, ensuring the app performs actions like saving entries, retrieving data, and displaying them to the user with minimal delay.

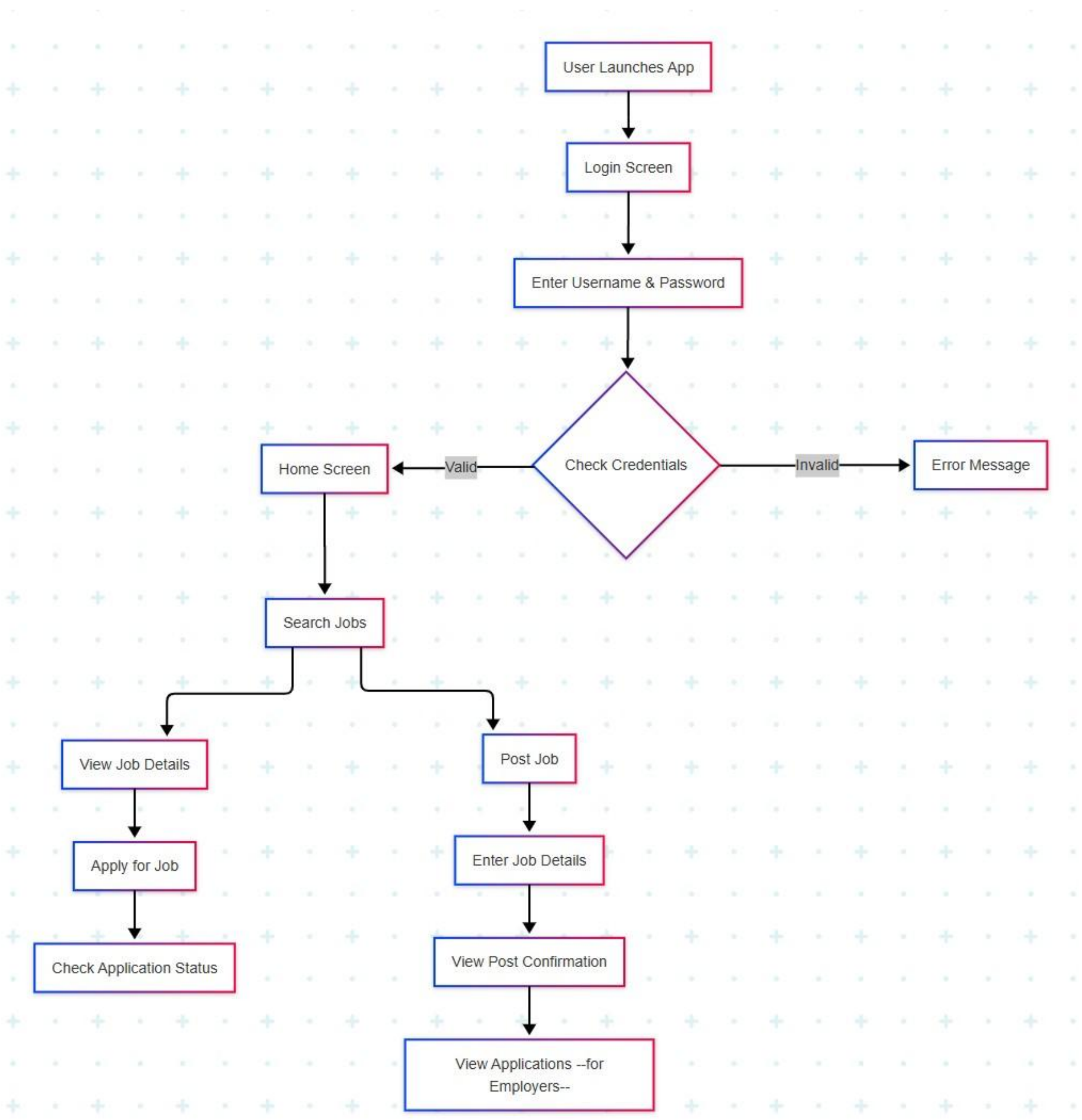


Fig – System Flow Diagram

3.1.2 ARCHITECTURE DIAGRAM

This diagram describes the core components of the application and how they interact with the device hardware and Android OS services.

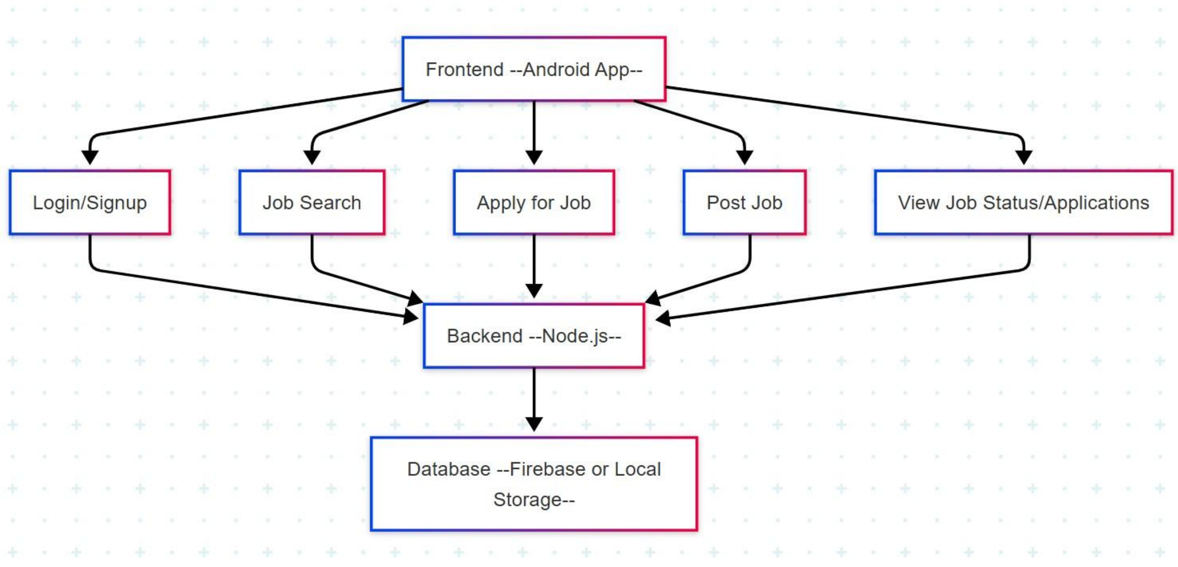


Fig – Architecture Diagram

3.1.3 USE CASE DIAGRAM

The Use Case Diagram describes how the user interacts with the application and highlights the various actions that can be performed.

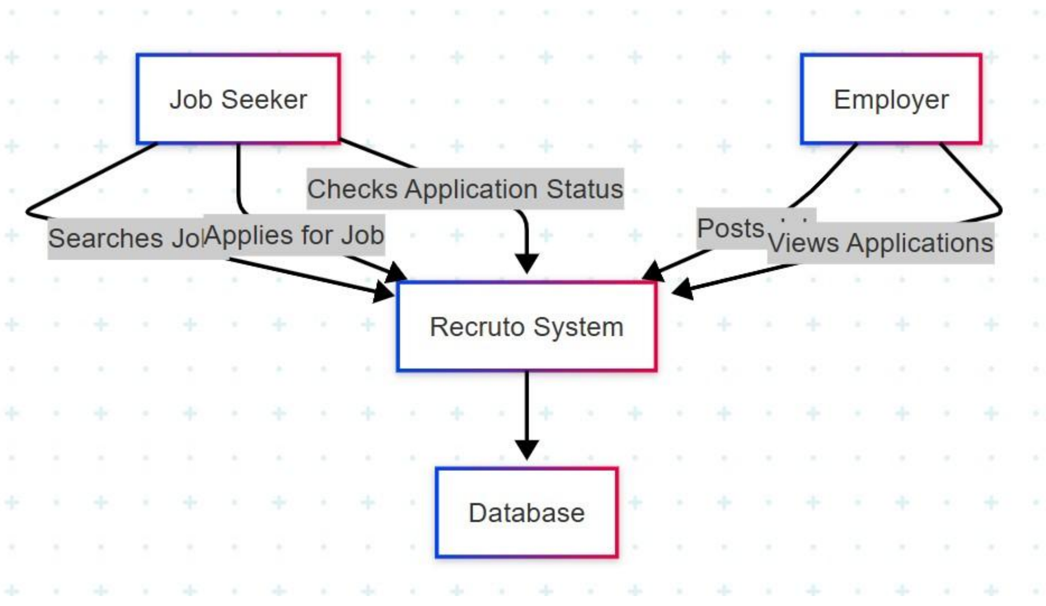


Fig – Use Case Diagram

4 PROJECT DESCRIPTION

Recruto is a mobile application designed to bridge the gap between individuals seeking part-time employment and those offering such opportunities. Unlike traditional job portals which focus mainly on full-time or corporate roles, Recruto is tailored to cater to part-time, temporary, or local work opportunities posted by individuals, shops, small businesses, and local employers.

This application allows any user to either post a job or apply for one, offering flexibility and accessibility for a wide range of users. The system also enables applicants to track their application status, while job posters can review all received applications in a structured format. The core aim is to provide a simple, accessible, and efficient job portal solution for communities that rely heavily on short-term work.

4.1 METHODOLOGY

The methodology followed for building Recruto is based on the Agile Development Model, which supports incremental development, constant feedback, and iterative improvements. Each feature was implemented in small phases, with consistent testing and user validation at every step.

This iterative approach ensures:

- Continuous integration of features
- Flexibility to improve UI/UX based on feedback
- Faster identification and correction of bugs
- High adaptability to future requirements and extensions

The app was developed using Android Studio with Kotlin, following material design guidelines, incorporating Mermaid for design diagrams, and focusing on clean architecture and user-centric design.

4.1.1 MODULES

The application is divided into several key modules to manage responsibilities effectively:

1. Authentication Module

- Handles user login using username and password
- Basic validation (e.g., admin/admin) for demo purposes
- Redirects authenticated users to the home screen

2. Job Posting Module

- Allows users to create and submit job listings
- Input fields for job title, description, location, and category
- Stores data locally or in cloud (based on future extension)

3. Job Search & Application Module

- Enables users to browse available part-time jobs
- Users can view job details and apply with a single click
- Prevents duplicate applications

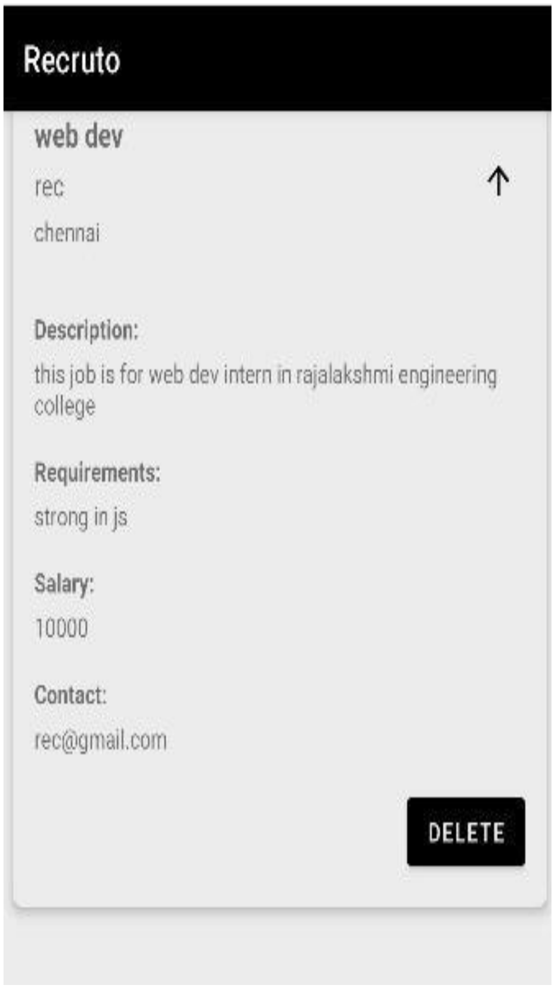
4. Application Review Module

- Employers (users who posted jobs) can view all applications received for each job

- Displays applicant name, status, and resume/message

Each module is built with scalability and maintainability in mind, ensuring that the system can grow in complexity over time without affecting performance or user experience.

4.2 PROJECT OUTPUT SCREEN SHOTS


Recruto

web dev

rec

chennai

Description:

this job is for web dev intern in rajalakshmi engineering college

Requirements:

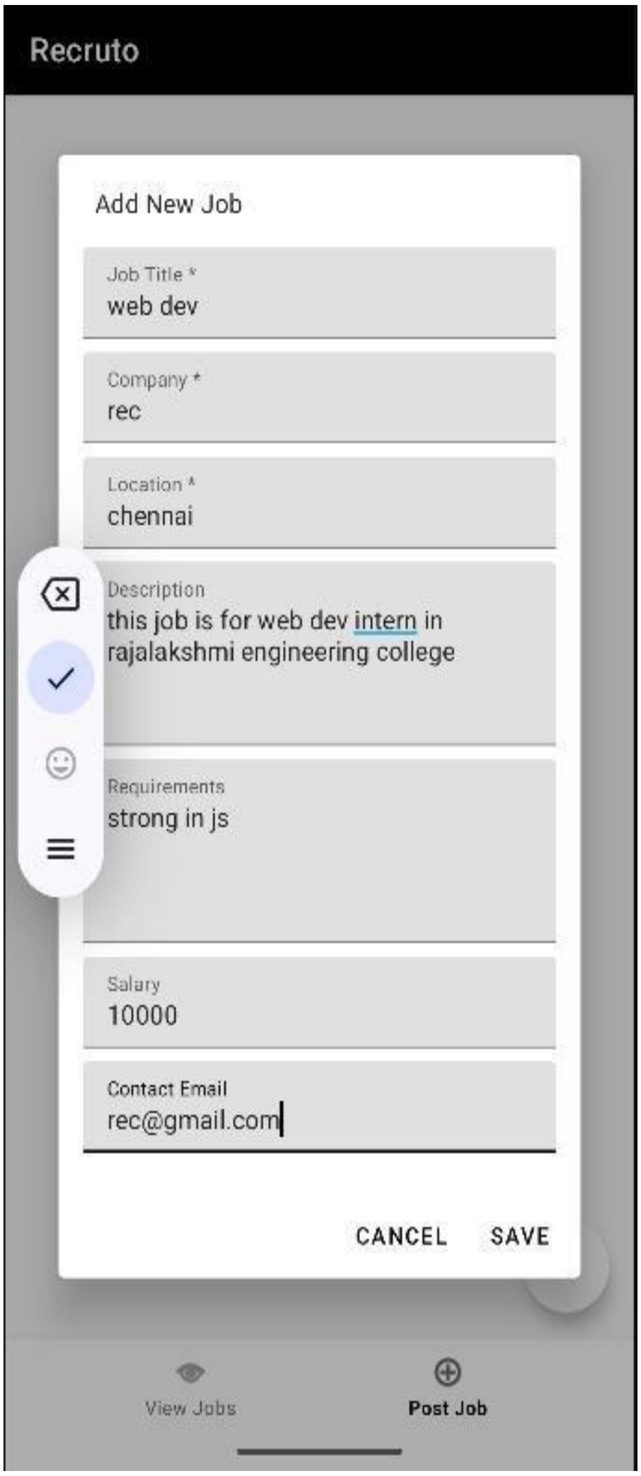
strong in js

Salary:

10000

Contact:

rec@gmail.com


Recruto

Add New Job

Job Title *: web dev

Company *: rec

Location *: chennai

Description: this job is for web dev intern in rajalakshmi engineering college

Requirements: strong in js

Salary: 10000

Contact Email: rec@gmail.com

5 CONCLUSION

5.1 GENERAL

The **Recruto** application delivers a streamlined and accessible platform for managing part-time job opportunities. By enabling any user to post and apply for jobs, track application status, and review applicants, the app bridges the gap between job seekers and local job providers. Built using **Android Studio and Kotlin**, Recruto emphasizes ease of use, responsive navigation, and a clean, modern interface that caters to a wide range of users including students, freelancers, and small business owners. Its flexible features and scalable structure ensure that the application remains practical and relevant in dynamic job markets.

We have accomplished:

- **Two-Way Job Interaction:**

  Enabled users to both post and apply for jobs, fostering an open and dynamic job environment.

- **Application Status Tracking:**

  Provided a real-time status update system for applicants to monitor their job applications.

- **Application Review System:**

  Allowed job posters to view and manage applicants seamlessly through an intuitive interface.

- **User Authentication:**

  Implemented a basic login mechanism for secure access and personalized job handling.

- **Simple and Responsive UI:**

  Developed a clean and user-friendly interface using Android Studio and Kotlin for efficient usage.

- **Modular and Maintainable Codebase:**

  Structured the application into reusable components, simplifying future updates and feature additions.

- **Potential for Backend Integration:**

  Designed with the flexibility to connect to remote databases or cloud services for real-time data handling in future versions.

Overall, Recruto stands as a reliable and scalable mobile solution that redefines the way part-time job opportunities are discovered, managed, and applied for—making it an ideal companion for today’s flexible workforce.

APPENDIX

SOURCE CODE

main_activity.kt

```
package com.example.recruto /



import android.os.Bundle import

androidx.appcompat.app.AppCompatActivity import

androidx.navigation.fragment.NavHostFragment import

androidx.navigation.ui.setupWithNavController import

com.google.android.material.bottomnavigation.BottomNavigationView class

MainActivity : AppCompatActivity() {     override fun

onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity_main) // This loads activity_main.xml



    // Find the NavHostFragment         val

navHostFragment = supportFragmentManager

        .findFragmentById(R.id.nav_host_fragment_container) as NavHostFragment

val navController = navHostFragment.navController



    // Find the BottomNavigationView         val bottomNavView =

findViewById<BottomNavigationView>(R.id.bottom_nav_view)
```

```
        // Setup BottomNavigationView with NavController
        // This makes the menu items switch the fragments displayed in nav_host_fragment_container // The IDs in
bottom_nav_menu.xml must match the IDs of the destinations in nav_graph.xml
bottomNavView.setupWithNavController(navController)
    }
}
```

Post_job_fragment.kt package

```
com.example.recruto


import android.os.Bundle import android.view.LayoutInflater import
android.view.View import android.view.ViewGroup import
android.widget.Button import android.widget.EditText import
android.widget.Toast import androidx.fragment.app.Fragment import
androidx.recyclerview.widget.LinearLayoutManager import
androidx.recyclerview.widget.RecyclerView import
com.google.android.material.dialog.MaterialAlertDialogBuilder import
com.google.android.material.floatingactionbutton.FloatingActionButton


class PostJobFragment : Fragment() {


    private lateinit var recyclerView: RecyclerView     private
lateinit var adapter: JobAdapter     private lateinit var dbHelper:
JobDatabaseHelper     private var jobList:
```

```
MutableList<JobPost> = mutableListOf() override fun

onCreateView(

      inflater: LayoutInflater, container: ViewGroup?,

savedInstanceState: Bundle?

   ): View? {

      // Inflate the layout for this fragment        val view =

inflater.inflate(R.layout.fragment_post_job, container, false)



      dbHelper = JobDatabaseHelper(requireContext())



      // Set up RecyclerView        recyclerView =

view.findViewById(R.id.recycler_view_jobs)

recyclerView.layoutManager = LinearLayoutManager(context)



      // Initialize adapter        adapter =

JobAdapter(jobList) { jobId ->

         // Delete job function when delete button clicked

if (dbHelper.deleteJob(jobId)) {              loadJobs()

            Toast.makeText(context, "Job deleted successfully", Toast.LENGTH_SHORT).show()

         }

      }

recyclerView.adapter = adapter
```

```
        // Set up FAB         val fab =
view.findViewById<FloatingActionButton>(R.id.fab_add_job)
fab.setOnClickListener {             showAddJobDialog()
        }

        return view
    }


    override fun onResume() {
super.onResume()         loadJobs()
    }


    private fun loadJobs() {
jobList.clear()
jobList.addAll(dbHelper.getAllJobs())
adapter.notifyDataSetChanged()
    }
private fun showAddJobDialog() {         val dialogView =
LayoutInflater.from(requireContext())
            .inflate(R.layout.dialog_add_job, null)


        val titleEditText = dialogView.findViewById<EditText>(R.id.edit_text_job_title)         val
companyEditText = dialogView.findViewById<EditText>(R.id.edit_text_company)         val
locationEditText = dialogView.findViewById<EditText>(R.id.edit_text_location)         val
descriptionEditText = dialogView.findViewById<EditText>(R.id.edit_text_description)         val
requirementsEditText = dialogView.findViewById<EditText>(R.id.edit_text_requirements)
```

```
val salaryEditText = dialogView.findViewById<EditText>(R.id.edit_text_salary)         val

emailEditText = dialogView.findViewById<EditText>(R.id.edit_text_contact_email)



     MaterialAlertDialogBuilder(requireContext())

         .setTitle("Add New Job")

         .setView(dialogView)
         .setPositiveButton("Save") { _, _ ->

             // Validate input if

(titleEditText.text.isNullOrBlank() ||

companyEditText.text.isNullOrBlank() ||

locationEditText.text.isNullOrBlank()) {

                 Toast.makeText(context, "Please fill all required fields", Toast.LENGTH_SHORT).show()

return@setPositiveButton

             }



             // Create a new job post                val newJob = JobPost(

title = titleEditText.text.toString().trim(),                  company =

companyEditText.text.toString().trim(),                  location =

locationEditText.text.toString().trim(),                  description =

descriptionEditText.text.toString().trim(),                  requirements =

requirementsEditText.text.toString().trim(),                  salary =

salaryEditText.text.toString().trim(),                  contactEmail =

emailEditText.text.toString().trim()

             )
```

```
               // Insert job to database
val id = dbHelper.insertJob(newJob)
               if (id > 0) {
  Toast.makeText(context, "Job posted successfully", Toast.LENGTH_SHORT).show()
                   loadJobs()
               } else {
                   Toast.makeText(context, "Failed to post job", Toast.LENGTH_SHORT).show()
               }
           }
           .setNegativeButton("Cancel", null)
           .show()
    }
}
```

Post_Job_Fragment.xml

```
 <?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".PostJobFragment">


   <androidx.recyclerview.widget.RecyclerView
android:id="@+id/recycler_view_jobs"
```

```
android:layout_width="match_parent"

android:layout_height="match_parent"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toTopOf="parent" />

<com.google.android.material.floatingactionbutton.FloatingActionButton

android:id="@+id/fab_add_job"

android:layout_width="wrap_content"

android:layout_height="wrap_content"         android:layout_margin="16dp"

android:src="@drawable/ic_add"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintEnd_toEndOf="parent"

android:contentDescription="Add new job" />



</androidx.constraintlayout.widget.ConstraintLayout>
```

Nav_Graph.xml

```
<?xml version="1.0" encoding="utf-8"?>

<navigation xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/nav_graph"

app:startDestination="@id/navigation_view_jobs">
```

```
    <fragment        android:id="@+id/navigation_view_jobs"
android:name="com.example.recruto.ViewJobsFragment"
android:label="fragment_view_jobs"
tools:layout="@layout/fragment_view_jobs" />


    <fragment        android:id="@+id/navigation_post_job"
android:name="com.example.recruto.PostJobFragment"
android:label="fragment_post_job"
tools:layout="@layout/fragment_post_job" />
</navigation>
```

REFERENCES

REFERENCES

1. Android Developers. (n.d.). *Build your first app*. Retrieved from https://developer.android.com/training/basics/firstapp
2. Android Developers. (n.d.). *Data Storage Using SQLite*. Retrieved from https://developer.android.com/training/data-storage/sqlite
3. Android Developers. (n.d.). *Using the Camera*. Retrieved from https://developer.android.com/training/camera/photobasics
4. Vogella, L. (2023). *Android SQLite database tutorial*. Retrieved from https://www.vogella.com/tutorials/AndroidSQLite/article.html
5. Room Persistence Library. (n.d.). *Android Jetpack Room*. Retrieved from https://developer.android.com/jetpack/androidx/releases/room

6. Stack Overflow. (n.d.). *Best practices for Android app architecture*. Retrieved from https://stackoverflow.com/questions/39814549/androidapp-architecture-best-practices

7. Choe, S. (2020). *Effective Mobile App Design for User Engagement*. Mobile UX Review Journal, 12(3), 45-52.

8. Rashid, S., & Sharma, R. (2021). *A Study on Digital Receipts and Warranty Tracking Apps*. International Journal of Computer Applications, 183(29), 15-20.