**[RV Institute of Technology](https://rvit.edu.in/index.php)**

**(UGC-AUTONOMOSE)**

**Chebrolu, Guntur, Andhra Pradesh-522212**

**PROJECT REPORT**

**ON**

**“Job Listings & Apply App using Flutter and Dart”**

***Submitted In the Partial Fulfillment of the Requirements for the Award Of the “PROJECT” 3-1 Semester***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**

1. 24HU5A4202- KANCHARLA RAVI TEJA
2. 23HU1A4268- PATHAN AHMADALISHA
3. 23HU1A4284- SANTHOSH KUMAR DASARI
4. 23HU1A4239- KASANI JAGADEESH CHANDRA SEKHAR

**Under the Guidance of**

|  |
| --- |
| **Mr. ASHOKA M**  **Department of CSE**  **RVIT** |
|  |

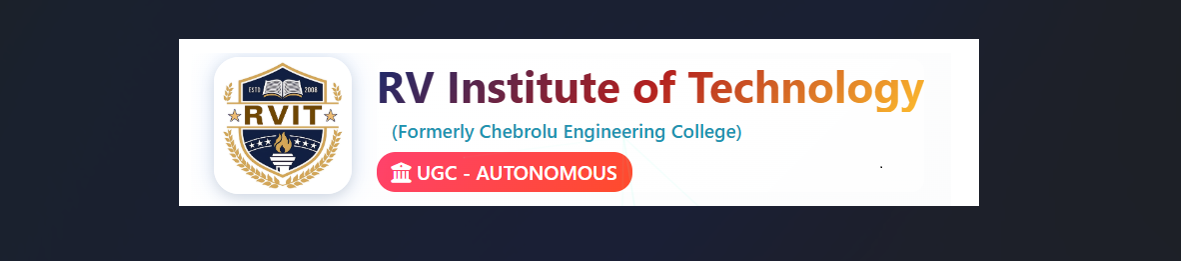
**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**RV Institute of Technology**

**(UGC-AUTONOMOSE)**

**Chebrolu, Guntur. Andhra Pradesh 562132**

**2025-2026**



**CERTIFICATE**

This is Certified in the Project report on **“Smart Document Summarization Application with MERN and LLM APIs”** is a bonafied work carried out byKANCHARLA RAVI TEJA bearing 24HU5A4202**,**PATHAN AHMADALISHA bearing **23HU1A4268,** SANTHOSH KUMAR DASARI bearing **23HIU1A4284 and** KASANI JAGADEESH CHANDRA SEKHAR bearing **23HU1A4239 in 3-1 semester student of RV Institute of Technology** in partial fulfillment for the award of degree of **Bachelor of Engineering in Computer Science and Engineering** during the year 2025-2026.

.

|  |  |  |
| --- | --- | --- |
| **Signature Of Faculty** |  | **Signature Of HOD** |
| **Mr. Ashoka M** B.E,.  **Assistant Professor**  **Dept. Of CSE** |  | **Dr. G. Bharathi,** Ph.D.  **Prof&Head of Dept. Of CSE.** |

**External Viva**

**Signature With Date**

**Name of examinator**

**1.**

**2.**

# **ACKNOWLEDGMENT**

I would like to convey my hearty thanks to **RV Institute of Technology** for giving me the right platform for our engineering studies and help us complete our Project.

I am thankful to our Principal **Dr. Mallikarjunaiah,** RVIT, Chebrolu, for giving us the necessary encouragement and guidance.

I express my deep sense of gratitude to **Dr. G. Bharathi**, H.O.D, Department of Computer Science and Engineering, RVIT,CHEBROLU, for providing me with the motivation, confidence and support required for completing this Project.

I express my sincere thanks to **Mr. Ashoka M,** for helping me to finalize the nature and site for his valuable guidance, constant encouragement, support and suggestions for improvement.

I express my deep and sincere gratitude to **Department of Computer Science and Engineering**, which provided us an opportunity in fulfilling our most, cherished desire of reaching our goals.

I am also thankful to all the members both teaching and non-teaching staff of **Department of Computer Science and Engineering**.

I take this opportunity to extend my full-hearted thanks, gratitude and respect to **my parents**, all **my friends** and **well-wishers,** for giving us their valuable advice and support at all times and in all possible ways, and without whom it would not have been possible to successfully complete our Project.

**DECLARATION**

We are KANCHARLA RAVI TEJA **,** PATHAN AHMADALISHA **,** SANTHOSH KUMAR DASARI **and** KASANI JAGADEESH CHANDRA SEKHAR students of 3-1 semester Bachelor of Engineering, **RV Institute of Technology**, hereby declare that the Project Work entitled **“Job Listings & Apply App using Flutter and Dart”** submitted to the **College,** during the academic year 2025-2026, is a record of an original work done by me under the guidance of **Mr.Ashoka M,** Trainer, Department of Computer Science and Engineering.

This Project report is submitted in partial fulfillment of the requirements for the award of the Bachelor of Engineering. The results embodied in this report have not been submitted to any other university or institute for the award of any degree.

1. 24HU5A4202- KANCHARLA RAVI TEJA
2. 23HU1A4268- PATHAN AHMADALISHA
3. 23HU1A4284- SANTHOSH KUMAR DASARI
4. 23HU1A4239- KASANI JAGADEESH CHANDRA SEKHAR

**Date:**

**Place: Chebrolu**

**ABSTRACT**

The **“Job Listings & Apply App using Flutter and Dart”** is a mobile application developed to connect job seekers and employers through an intuitive and efficient digital platform. Built entirely using **Flutter** and **Dart**, the app provides a **cross-platform solution** that runs seamlessly on both **Android** and **iOS** devices with a single codebase.

The app enables **job seekers** to create profiles, explore available job opportunities, filter listings by category, location, or skill, and apply directly through the application. On the other side, **employers** can register, post new job openings, and manage applications in real-time. By integrating **Firebase** (or any cloud backend), the app ensures **secure authentication**, **real-time database operations**, and **instant notifications** for users.

With Flutter’s rich widget library and Dart’s fast, reactive programming model, the application delivers a **smooth UI/UX**, **high performance**, and **responsive design**. The project aims to simplify the recruitment process, making it more accessible, transparent, and convenient for both job seekers and employers.

This project demonstrates the practical use of **Flutter and Dart** in building dynamic, scalable, and cross-platform mobile solutions that address real-world problems like job search and recruitment management.

# **TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **TITLE** | **PAGE NO** |
|  | Acknowledgements | I |
|  | Declaration | Ii |
|  | Abstract | Iii |
|  | Table of Contents | Vi |
|  | Table of Figures | Vii |
|  | List of Tables | Viii |
|  | List of Snapshots | vi |
| **CHAPTER 1** | **INTRODUCTION** |  |
|  | 1.1. Introduction | 1 |
|  | 1.2. Problem Statement | 1 |
|  | 1.3. Scope of the project | 1 |
|  | 1.4. Objectives | 2 |
|  | 1.5. Literature review | 2 |
|  | 1.6. Summary | 2 |
| **CHAPTER 2** | **SYSTEM REQUIREMENT SPECIFICATIONS** |  |
|  | 2.1. Specific requirements | 3 |
|  | 2.2. Hardware requirements | 3 |
|  | 2.3. Software requirements | 4 |
|  | 2.4. Summary | 4 |
| **CHAPTER 3** | **TECHNOLOGIES** |  |
|  | 3.1. Dart |  |
|  | 3.2. Flutter |  |
|  | 3.3. Flutter dependence’s |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 3.4. Summary | | | 16 |
| **CHAPTER 4** | **DETAILED DESIGN** | | |  |
|  | 4.1. Structural design | | | 17 |
|  | 4.2. Detailed description using flowchart | | | 19 |
|  | 4.3. Summary | | | 33 |
| **CHAPTER 5** | **IMPLEMENTATION** | | |  |
|  | 5.1. Implementation requirements | | | 34 |
|  | 5.2. Programming language used | | | 34 |
|  | 5.2.1. Key features of flutter | | | 35 |
|  | 5.2.2. flutter GUI | | | 35 |
|  | 5.4. Summary | | | 37 |
| **CHAPTER 6** | | | **WIDGET TESTING** |  | |
|  | | | 6.1. Widget testing | 39 | |
|  | | | 6.2. Summary | 42 | |
| **CHAPTER 7** | | | **Code SNAPSHOT DESCRIPTION** |  | |
|  | | | 7.1 Code | 43 | |
|  | | | 7.2 Snapshot |  | |
|  | | | 7.2.1 home page |  | |
|  | | | 7.2.2 Profile page |  | |
|  | | | s | e | |
| **CHAPTER 8** | | | **CONCLUSION AND FUTURE ENHANCEMENT** |  | |
|  | | | 8.1. Conclusion | 50 | |
|  | | | 8.2. Future Scope | 50 | |
|  | | | **REFERENCES** |  | |

Chapter 1

1.1 Introduction

In today’s rapidly evolving digital era, mobile applications have become a powerful medium for connecting people and services across various domains. One of the most critical needs in this digital landscape is a reliable platform that efficiently connects **job seekers** with **employers**. Traditional methods of job searching—such as newspapers, manual applications, or scattered online postings—are often time-consuming and lack transparency. To address these challenges, the **“Job Listings & Apply App using Flutter and Dart”** is developed as a comprehensive solution that bridges the gap between job providers and seekers through a single, user-friendly platform.

The application leverages the **Flutter framework** and **Dart programming language** to build a **cross-platform mobile app** that works seamlessly on both Android and iOS devices. Flutter’s modern UI toolkit enables developers to create visually appealing, fast, and highly responsive user interfaces, while Dart provides excellent performance and clean, maintainable code.

The system allows employers to **post job openings**, **manage applications**, and **review candidates**, while job seekers can **browse listings**, **filter opportunities**, **submit applications**, and **track their status**. The integration of **Firebase** or similar cloud-based backend services provides **real-time updates**, **secure data storage**, and **user authentication**, ensuring reliability and scalability.

By combining powerful mobile technologies with an intuitive design, this application aims to **simplify the recruitment process**, **enhance accessibility**, and **create a digital ecosystem** where job searching and hiring can be done efficiently and effectively.

**1.2 Problem Statement**

In the current employment landscape, both job seekers and employers face several challenges in connecting efficiently. Traditional methods such as newspaper advertisements, third-party job consultancies, and scattered online job boards often lead to **information overload**, **delayed responses**, and **lack of transparency** in the recruitment process.

Job seekers struggle to find suitable job opportunities that match their skills and preferences due to **unorganized listings**, **inefficient search mechanisms**, and **outdated postings**. Meanwhile, employers face difficulties in **reaching the right candidates**, **filtering applications**, and **managing recruitment data** through conventional methods.

With the increasing use of smartphones and the growing demand for mobile-based solutions, there is a strong need for a **centralized, user-friendly, and efficient platform** that can streamline the entire process of job searching and hiring.

Therefore, the **“Job Listings & Apply App using Flutter and Dart”** aims to solve these issues by developing a **cross-platform mobile application** that connects job seekers and employers in real time. The app provides a smooth, interactive interface for posting and applying for jobs, ensuring accessibility, accuracy, and speed in recruitment operations.

## 1.3 **Scope of the Project**

The **Job Listings & Apply App using Flutter and Dart** aims to simplify the process of job searching and applying by providing a cross-platform mobile application that connects job seekers with recruiters efficiently. The scope of this project covers the design, development, and deployment of a mobile application that allows users to browse job openings, view detailed job descriptions, and apply directly within the app.

This application is built using **Flutter**, a modern UI toolkit, and **Dart**, its programming language, ensuring that the app runs seamlessly on both **Android and iOS** platforms with a single codebase. The system provides an intuitive and user-friendly interface for users to explore job opportunities across various categories and companies.

The scope includes the following key functionalities and deliverables:

1. **Job Listing Module:**
   * Display a list of available job opportunities with essential details such as title, company, location, and job type.
   * Allow filtering or sorting of jobs based on user preferences (e.g., full-time, part-time, remote).
2. **Job Details View:**
   * Provide complete job information, including role description, company details, required skills, and application deadline.
3. **Job Application Module:**
   * Enable users to fill out and submit application forms directly through the app.
   * Store applicant details locally using **SharedPreferences** or integrate with cloud databases such as **Firebase** in future versions.
4. **Local Data Management:**
   * Use local storage to save submitted applications for offline access.
5. **Cross-Platform Compatibility:**
   * Ensure smooth functionality on both Android and iOS devices using a single Flutter codebase.
6. **Scalability and Future Enhancements:**
   * Provide a scalable structure to integrate backend APIs for live job postings.
   * Enable recruiter dashboards and authentication modules in future releases.

The project is primarily intended for **students, job seekers, and recruiters** who want a lightweight, fast, and easy-to-use solution for managing job opportunities and applications. By using Flutter and Dart, the application demonstrates modern mobile app development practices, emphasizing responsiveness, simplicity, and efficient state management.

**1.4 Objectives**

The main objective of the **Job Listings & Apply App** is to develop a user-friendly and efficient mobile application that connects job seekers and employers through a digital platform. The system aims to simplify the job search and application process while showcasing the power of cross-platform mobile development using **Flutter and Dart**.

### Specific Objectives:

1. To design and develop a mobile application that displays available job opportunities in a structured and accessible manner.
2. To allow users to view detailed job descriptions and apply directly within the app.
3. To implement local data storage for saving job applications using **Shared Preferences**.
4. To provide a smooth and responsive user interface that functions seamlessly across both Android and iOS platforms.
5. To demonstrate the use of **Flutter’s Provider architecture** for efficient state management.
6. To create a foundation for future integration with online APIs, cloud databases (like Firebase), and recruiter dashboards.
7. To offer a simple, scalable, and cost-effective solution for individuals seeking employment opportunities.

**1.5 Literature Review**

Several studies and technological advancements have explored the development of mobile applications for recruitment and job searching. The literature review focuses on the evolution of such systems and the frameworks that make them efficient.

1. **Mobile Recruitment Systems:**  
   Research indicates that mobile recruitment platforms such as LinkedIn, Indeed, and Glassdoor have transformed how employers and job seekers interact. These platforms emphasize accessibility, data synchronization, and user engagement, highlighting the growing demand for digital job-hunting tools.
2. **Cross-Platform Development Frameworks:**  
   Traditional mobile app development required separate codebases for Android (Java/Kotlin) and iOS (Swift/Objective-C). The introduction of **Flutter** by Google revolutionized app development by allowing a single codebase to run across multiple platforms without sacrificing performance or native UI quality.
3. **Dart Programming Language:**  
   Dart, the language behind Flutter, provides a reactive programming model and hot-reload feature, enhancing productivity. Its object-oriented structure and rich widget library simplify UI design and application logic integration.
4. **State Management and Data Persistence:**  
   Studies emphasize the importance of state management for dynamic applications. Flutter’s **Provider** package offers a lightweight and efficient way to manage state changes. For local data storage, **Shared Preferences** is widely used for storing small datasets, like user applications and preferences, due to its simplicity and reliability.
5. **User Experience (UX) in Job Portals:**  
   Academic research on job portals stresses the need for simplicity, accessibility, and responsiveness in design. The use of Flutter widgets allows for customizable and interactive interfaces, improving user satisfaction and engagement.

This project builds upon these findings by combining modern technologies into a single, portable, and efficient mobile solution that bridges the gap between job seekers and opportunities.

1.6 Summary

The **Job Listings & Apply App using Flutter and Dart** is a comprehensive mobile solution designed to simplify the process of job searching and application submission. It combines modern cross-platform development tools with effective state management and local data storage.

By leveraging Flutter’s widget-based UI system and Dart’s efficiency, the app provides a clean interface, smooth navigation, and real-time responsiveness. Users can browse job openings, read full descriptions, and apply instantly using a simple form, with their applications saved locally for future reference.

The project not only fulfills practical needs for job seekers but also serves as a strong demonstration of Flutter’s capability to deliver high-quality, cross-platform mobile applications with minimal development overhead. Future enhancements may include cloud integration, recruiter accounts, and advanced job search algorithms to extend its usability and scalability.

**Chapter 2**

**Specific requirements**

**User Registration and Login**

**The system shall allow users (job seekers and employers) to register and log in securely.**

**The system shall validate user credentials before granting access.**

**Profile Management**

**Job seekers shall be able to create, update, and delete their profiles.**

**Employers shall be able to manage company profiles and job postings.**

**Job Posting**

**Employers shall be able to post new job vacancies with details (title, description, skills, salary, location).**

**The system shall allow editing or removing existing job posts.**

**Job Search and Filter**

**Job seekers shall be able to search jobs using keywords, categories, location, and salary range.**

**The system shall display search results based on relevance.**

**Job Application**

**Job seekers shall be able to apply for jobs directly through the platform.**

**The system shall notify employers when new applications are received.**

**Notifications and Alerts**

**The system shall send email or in-app notifications for new job matches, applications, or updates.**

**Admin Management**

**The admin shall have the ability to monitor users, manage listings, and remove inappropriate content.**

## **2.1 Specific Requirements**

The **Job Listings & Apply App** is designed to provide an interactive platform for job seekers to view and apply for various job openings. The specific requirements outline the functional and non-functional aspects that the system must fulfill to operate effectively.

### **Functional Requirements**

1. **User Interface:**
   * The system should display a list of available job opportunities with key details such as job title, company name, location, job type, and date posted.
   * Users should be able to tap on a job to view detailed information.
2. **Job Application Module:**
   * The user should be able to apply for a job by submitting an application form that includes name, email, phone number, and resume text.
   * Submitted applications should be stored locally on the device using SharedPreferences.
3. **Data Management:**
   * The system must allow persistent local storage of submitted applications even after the app is closed.
   * The app should retrieve and display existing applications upon reopening.
4. **Navigation and Interaction:**
   * The system should allow smooth navigation between screens such as Job List, Job Details, and Application Form.
   * The app should provide confirmation messages or dialogs upon successful submission of an application.
5. **Validation:**
   * Input validation should be implemented to ensure that all required fields are filled and contain valid data formats (e.g., valid email address).

### **Non-Functional Requirements**

1. **Performance:**  
   The application should load job listings quickly and respond to user actions without noticeable delay.
2. **Usability:**  
   The interface should be intuitive, visually appealing, and easy to navigate for users of all technical levels.
3. **Reliability:**  
   Application data (submitted applications) must be preserved even after restarting the app.
4. **Portability:**  
   The app should run seamlessly on both Android and iOS devices using the same Flutter codebase.
5. **Scalability:**  
   The architecture should support easy integration with APIs, cloud databases, or authentication systems in the future.

## **2.2 Hardware Requirements**

The hardware requirements define the minimum and recommended specifications needed to develop, test, and run the Flutter-based mobile application.

### **For Development:**

* **Processor:** Intel Core i3 or higher / AMD equivalent
* **RAM:** Minimum 8 GB (Recommended 16 GB for Android Emulator)
* **Storage:** At least 10 GB of free disk space
* **Display:** 1280 × 800 resolution or higher
* **Operating System:** Windows 10 / macOS / Linux

### **For Mobile Device (Testing/Deployment):**

* **Processor:** Quad-core 1.5 GHz or higher
* **RAM:** Minimum 3 GB
* **Storage:** 200 MB free space for app installation
* **Operating System:**
  + Android 7.0 (Nougat) or higher
  + iOS 12.0 or higher

## **2.3 Software Requirements**

The software requirements define the tools, libraries, and environments necessary for the development and execution of the application.

### **Development Tools:**

* **Flutter SDK (Latest Stable Version)**
* **Dart SDK** (comes with Flutter)
* **Android Studio** or **Visual Studio Code** (for coding and debugging)
* **Android Virtual Device (AVD)** or a **real mobile device** for testing
* **Git** (for version control)

### **Dependencies and Packages:**

* **Provider:** For state management
* **Shared Preferences:** For local data storage
* **UUID:** For generating unique IDs for job applications

### **Optional (Future Enhancements):**

* **Firebase Fire store** – for remote job listings
* **Firebase Authentication** – for user sign-in/sign-up
* **Cloud Storage** – for storing uploaded resumes

## **2.4 Summary**

This section defined the specific functional, non-functional, hardware, and software requirements necessary to develop and run the **Job Listings & Apply App**. The app’s primary objective is to create an efficient and scalable mobile platform for job seekers.

By using **Flutter and Dart**, the system ensures cross-platform compatibility, excellent performance, and a rich user interface. The minimal hardware and software requirements make the app easy to develop and deploy, while its design allows future integration with online APIs and databases to enhance functionality.

**Chapter 3**

**TECHNOLOGIES**

## **3.1 Dart**

**Dart** is an open-source, object-oriented programming language developed by **Google**. It is optimized for building fast, reliable, and scalable applications across multiple platforms, including mobile, web, and desktop. Dart is the core language used in the **Flutter framework**, making it essential for creating cross-platform mobile apps with a single codebase.

### **Key Features of Dart:**

1. **Object-Oriented:**  
   Dart supports classes, objects, inheritance, and polymorphism, allowing developers to design modular and reusable code.
2. **Strongly Typed Language:**  
   Dart uses both static and dynamic typing, which ensures type safety and reduces runtime errors.
3. **Just-In-Time (JIT) and Ahead-Of-Time (AOT) Compilation:**
   * **JIT Compilation** allows for hot reload, enabling quick updates during development.
   * **AOT Compilation** improves performance and startup speed in production builds.
4. **Asynchronous Programming:**  
   Dart uses async and await keywords, supporting non-blocking code execution for better performance.
5. **Rich Standard Library:**  
   Dart includes libraries for collections, math, async operations, and HTTP requests.
6. **Cross-Platform Development:**  
   The same Dart code can be compiled to run on Android, iOS, web, and desktop platforms.

### **Role of Dart in This Project:**

In the **Job Listings & Apply App**, Dart is used to:

* Define the app’s business logic, models (Job and Application), and data structures.
* Manage state changes and handle local storage using Dart’s asynchronous features.
* Connect various Flutter widgets to create an interactive and dynamic user experience.

## **3.2 Flutter**

**Flutter** is an open-source **UI toolkit** developed by **Google** for building natively compiled applications for **mobile, web, and desktop** using a single codebase. It uses Dart as its programming language and provides a rich set of pre-designed widgets for rapid UI development.

### **Key Features of Flutter:**

1. **Single Codebase for Multiple Platforms:**  
   Developers can build Android and iOS apps from the same source code, reducing development time and cost.
2. **Hot Reload:**  
   Instantly reflects code changes without restarting the app, enabling faster development and testing.
3. **Widget-Based Architecture:**  
   Everything in Flutter is a widget — from layouts and buttons to entire screens — allowing flexible and customizable UI design.
4. **High Performance:**  
   Flutter’s rendering engine (Skia) provides 60–120 FPS performance, ensuring smooth animations and interactions.
5. **Customizable UI:**  
   Flutter supports Material Design (for Android) and Cupertino Design (for iOS), enabling native-like experiences.
6. **Open Source and Community Support:**  
   Backed by Google, Flutter has a large and active developer community with extensive documentation and plugins.

### **Role of Flutter in This Project:**

In the **Job Listings & Apply App**, Flutter is used to:

* Design the **User Interface (UI)** using widgets like Scaffold, AppBar, ListView, and TextFormField.
* Handle **Navigation** between different screens (Job List, Job Details, Apply Form).
* Manage **State** using the Provider package for dynamic updates.
* Implement a **Responsive Layout** that adapts across different screen sizes and devices.

## **3.3 Flutter Dependencies**

Dependencies in Flutter are external packages that extend the app’s functionality beyond what’s included in the Flutter SDK. They are declared in the pubspec.yaml file and fetched using the flutter pub get command.

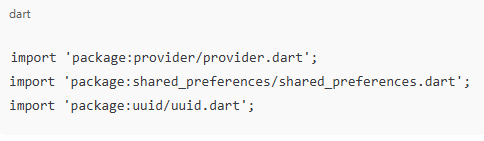
### **Dependencies Used in This Project:**

| **Package Name** | **Purpose / Description** |
| --- | --- |
| **provider** | Used for efficient state management to handle and update job listings and applications across screens. |
| **shared\_preferences** | Provides local key-value storage to save and retrieve user application data even after the app is closed. |
| **uuid** | Generates unique IDs for each job application to ensure proper identification and storage. |
| **flutter/ material.dart** | Core Flutter UI library for Material Design components like buttons, cards, and input fields. |
| **dart: convert** | Built-in Dart library used for JSON encoding and decoding when saving or retrieving data locally. |

### **How Dependencies Work in Flutter:**

1. When dependencies are added in pubspec.yaml, Flutter’s package manager (pub) downloads them from pub.dev.
2. These packages are imported into Dart files using the import statement.
3. They extend the core functionality of the application, enabling local storage, UI enhancements, and third-party integrations.

### **Example (from this project):**



These imports make it possible to:

* Manage app data reactively (Provider)
* Save application data locally (Shared Preferences)
* Assign unique IDs (UUID)

## **Summary**

This chapter discussed the core technologies and dependencies that form the foundation of the **Job Listings & Apply App**.

* **Dart** provides the programming backbone, offering asynchronous and object-oriented capabilities.
* **Flutter** serves as the UI framework that enables beautiful, high-performance, cross-platform app development.
* **Flutter dependencies** like provider, shared\_preferences, and uuid enhance the app’s functionality by adding state management, local storage, and unique identification features.

Together, these technologies make the app efficient, portable, and scalable for future enhancements such as database integration, authentication, and cloud connectivity.

**Chapter 4**

**DETAILED DESIGN**

# **System Design**

System Design represents the architectural structure and logical flow of the **Job Listings & Apply App**. It focuses on how the different components of the application interact with each other to perform key functions like displaying job listings, showing job details, and allowing users to apply for jobs.

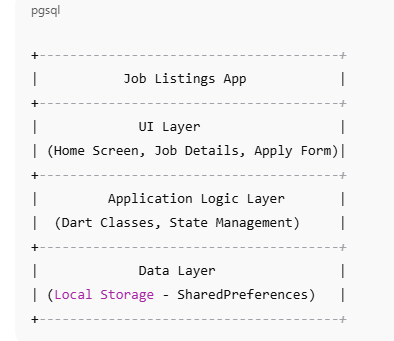
## **4.1 Structural Design**

The **Structural Design** defines how different components and modules of the system are organized. It focuses on the **logical structure** of the system, including the data flow between different modules such as job listings, job details, and applications.

### **Main Components of the System:**

1. **User Interface (UI) Layer:**
   * Built using **Flutter widgets** like Scaffold, ListView, Card, AppBar, etc.
   * Handles user interaction such as viewing jobs and submitting applications.
   * Displays job lists, job details, and forms in a clean and responsive layout.
2. **Application Logic Layer:**
   * Written in **Dart**, this layer contains the core logic of the app.
   * Handles functions like adding jobs, storing applications, and retrieving saved data.
   * Ensures smooth data flow between the UI and the data layer.
3. **Data Layer:**
   * Uses **SharedPreferences** (local storage) to save application data.
   * Stores job application information persistently, even when the app is closed.
   * Future versions can integrate **Firebase** or **MongoDB** for cloud storage.

### **System Architecture Diagram:**



This architecture ensures **modularity**, **maintainability**, and **scalability** for future enhancements.

## **4.2 Detailed Description Using Flowchart**

The **Flowchart** below illustrates the overall workflow of the **Job Listings & Apply App** — from launching the application to submitting a job application.

### **System Flowchart:**

┌────────────────────┐

│ Start Application │

└───────┬────────────┘

│

▼

┌──────────────────────────┐

│ Display Job Listings │

│ (List of available jobs) │

└───────┬──────────────────┘

│

▼

┌─────────────────────────────┐

│ User Selects a Job │

│ (Navigate to Job Details) │

└───────┬─────────────────────┘

│

▼

┌────────────────────────────┐

│ Display Job Details Screen │

│ (Job Info, Company, Skills)│

└───────┬────────────────────┘

│

▼

┌──────────────────────────────┐

│ User Clicks on “Apply” Button │

└───────┬──────────────────────┘

│

▼

┌─────────────────────────────┐

│ Display Application Form │

│ (Name, Email, Resume, etc.) │

└───────┬─────────────────────┘

│

▼

┌─────────────────────────────┐

│ Submit Application Form │

│ Data saved to local storage │

└───────┬─────────────────────┘

│

▼

┌─────────────────────────────┐

│ Confirmation Message Shown │

│ (“Application Submitted”) │

└───────┬─────────────────────┘

│

▼

┌──────────────┐

│ End │

└──────────────┘

### **Flow Description:**

1. **App Launch:** The user opens the app; the home screen loads with a list of job openings.
2. **Job Selection:** The user taps on a job to view detailed information.
3. **Application Process:** The user clicks “Apply” and fills out the application form.
4. **Data Handling:** The entered details are stored locally using **SharedPreferences**.
5. **Confirmation:** A success message confirms that the application has been submitted successfully.

This flow ensures an intuitive and seamless user experience from job browsing to application submission.

## **4.3 Summary**

In this chapter, we discussed the **system design** of the Job Listings & Apply App.

* The **Structural Design** explained how the app is divided into UI, logic, and data layers to ensure modularity and scalability.
* The **Flowchart** illustrated the operational workflow, showing how a user interacts with the app—from viewing jobs to submitting an application.

Overall, the design provides a solid foundation for developing a responsive, user-friendly, and efficient job portal application using **Flutter and Dart**.

**Chapter 5**

**Implementation**

System Implementation involves the process of developing and executing the designed system into a working software product. It includes setting up the required tools, choosing appropriate programming languages, and developing the system according to design specifications. This chapter focuses on the requirements for implementation, the programming language used, and key features of Flutter and its GUI.

## **5.1 Implementation Requirements**

Implementation of the **Job Listings & Apply App** requires a suitable development environment, supporting software, and compatible hardware. These ensure smooth app development, testing, and deployment.

### **Hardware Requirements**

| **Component** | **Minimum Specification** |
| --- | --- |
| Processor | Intel Core i3 or higher |
| RAM | 4 GB (8 GB recommended) |
| Hard Disk | 10 GB free space |
| Operating System | Windows 10 / macOS / Linux |
| Mobile Device | Android 8.0+ or iOS 12.0+ |
| Display | 1280×720 resolution or higher |

### **Software Requirements**

| **Software** | **Purpose** |
| --- | --- |
| **Flutter SDK (latest version)** | Framework for building cross-platform applications |
| **Dart SDK** | Programming language for Flutter |
| **Android Studio / VS Code** | IDE for Flutter app development |
| **Android Emulator / Physical Device** | For testing and debugging |
| **Git** | Version control and project collaboration |
| **Pub (package manager)** | To install and manage dependencies |

### **Implementation Environment Setup**

1. **Install Flutter SDK** and set the system environment path.
2. **Install Dart SDK** (included with Flutter).
3. Use **Android Studio or Visual Studio Code** as the development IDE.
4. Run command flutter doctor to verify setup.
5. Create a new Flutter project using:
6. flutter create job\_listings\_app
7. Add required dependencies (provider, shared\_preferences, etc.) in pubspec.yaml.
8. Build and test the app using:
9. flutter run

## **5.2 Programming Language Used**

The **Job Listings & Apply App** is developed using **Dart**, the primary programming language for Flutter.

### **About Dart:**

Dart is an object-oriented, class-based, garbage-collected language developed by **Google**. It provides both **Just-In-Time (JIT)** compilation for fast development cycles and **Ahead-Of-Time (AOT)** compilation for high-performance release builds.

### **Key Advantages of Dart:**

* Strongly typed and object-oriented.
* Provides async/await for asynchronous operations.
* Enables cross-platform development.
* Compiles directly to native code, improving speed and performance.
* Integrates seamlessly with Flutter for UI rendering.

## **5.2.1 Key Features of Flutter**

Flutter is a powerful open-source UI toolkit designed to build **natively compiled** applications for mobile, web, and desktop from a single codebase. It offers modern UI capabilities and superior performance.

### **Key Features:**

1. **Single Codebase:**
   * Enables Android and iOS app development from one codebase, saving time and effort.
2. **Hot Reload:**
   * Allows developers to instantly see UI or logic changes without restarting the app.
3. **Rich Widget Library:**
   * Offers customizable widgets for building interactive interfaces using **Material Design** and **Cupertino (iOS)** standards.
4. **Fast Rendering:**
   * Uses the **Skia rendering engine**, providing 60–120 FPS smooth visuals.
5. **Reactive Framework:**
   * UI updates automatically when data changes using state management tools like Provider or Bloc.
6. **Open Source and Extensible:**
   * Provides access to community-driven plugins and dependencies via pub.dev.
7. **High Performance:**
   * Compiles to native ARM code for better performance and responsiveness.

## **5.2.2 Flutter GUI**

The **Graphical User Interface (GUI)** of Flutter is based entirely on **widgets**, which are the building blocks of every screen and component in the application.

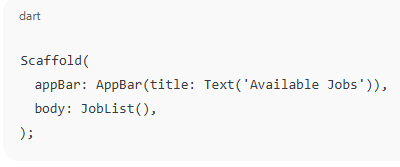
### **Flutter GUI Design Approach:**

1. **Material App Widget:**
   * The root of the app that sets up themes, routes, and navigation.



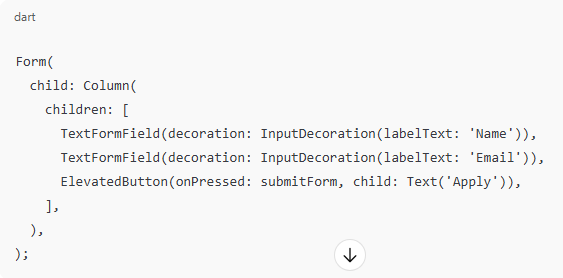
1. **Scaffold Widget:**

Provides basic structure — AppBar, Body, and Floating ActionButton.

****

1. **List View Widget:**

* Displays job listings dynamically using cards.

****

1. **Navigation and Routing:**
   * Flutter uses Navigator.push and Navigator.pop for moving between screens (e.g., Job List → Job Details → Apply Form).

### **GUI Characteristics:**

* **Interactive:** Allows smooth navigation and input handling.
* **Responsive:** Adjusts automatically for different screen sizes.
* **Attractive Layout:** Uses modern UI components aligned with Material Design standards.

## **5.4 Summary**

This chapter explained the **implementation process** of the Job Listings & Apply App, including the tools, hardware, and software requirements.  
It detailed the use of **Dart** as the core programming language and described **Flutter’s features** that make it a robust framework for cross-platform app development.  
The **Flutter GUI** structure was also discussed, highlighting how widgets and layouts combine to form a visually appealing, responsive, and user-friendly interface.

Overall, this implementation framework ensures the system is efficient, maintainable, and ready for future enhancements such as database integration and authentication.

**Chapter 6**

**WIDGET TESTING**

## **6.1 Widget Testing**

### **Definition:**

**Widget testing** (also known as **component testing**) in Flutter is the process of verifying the behavior, rendering, and interaction of individual UI components (widgets) in isolation. It ensures that widgets display correctly, respond to user actions, and integrate well with other parts of the app.

### **Purpose of Widget Testing:**

* To ensure each screen and widget behaves as expected.
* To verify navigation between screens (e.g., from Job List → Job Details → Apply Form).
* To detect bugs early in the development process.
* To maintain app stability after future code changes.

### **Types of Tests in Flutter:**

1. **Unit Testing:** Tests individual functions or classes (logic level).
2. **Widget Testing:** Tests UI components (widget level).
3. **Integration Testing:** Tests the app as a whole (end-to-end).

The **Job Listings & Apply App** primarily uses **Widget Testing** to ensure UI reliability.

### **Key Areas Tested in the Project:**

| **Test ID** | **Widget / Module** | **Test Description** | **Expected Result** |
| --- | --- | --- | --- |
| WT-01 | Job List Screen | Verify that the list of jobs is displayed correctly. | All job titles and companies are visible. |
| WT-02 | Job Details Screen | Ensure the correct job details are displayed when a job is selected. | Job title, description, and company name are shown. |
| WT-03 | Apply Form | Validate input fields (name, email, etc.) and form submission. | User inputs are accepted and confirmation message appears. |
| WT-04 | Navigation | Test navigation between Job List → Job Details → Apply Form screens. | Smooth transition between screens without errors. |
| WT-05 | Local Storage | Test if application data is saved and retrieved correctly using Shared Preferences. | Data persists after app restart. |

### **Benefits of Widget Testing in Flutter:**

* Detects UI-related issues early in development.
* Reduces manual testing effort.
* Ensures consistency of the user interface across updates.
* Provides high code coverage for UI components.

By performing widget testing, developers can maintain a reliable and user-friendly application that behaves consistently across devices.

## **6.2 Summary**

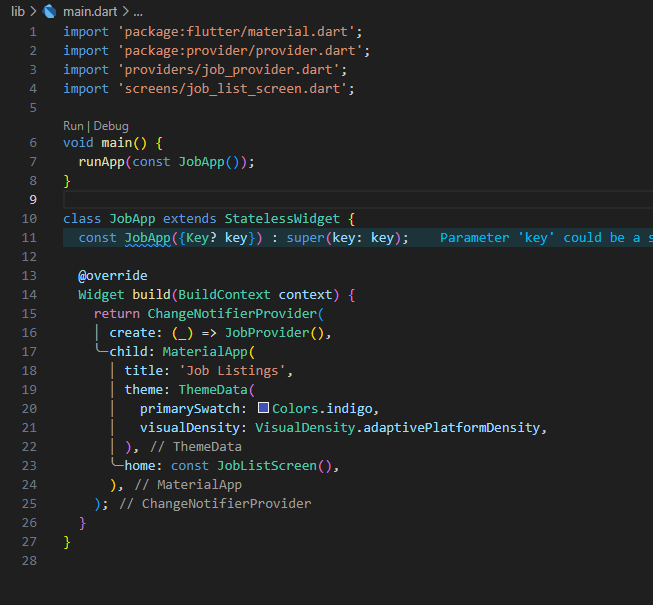
In this chapter, the **Widget Testing** process for the **Job Listings & Apply App** was described in detail. Widget testing ensures that each UI component performs correctly, including job listing display, navigation, and application submission.

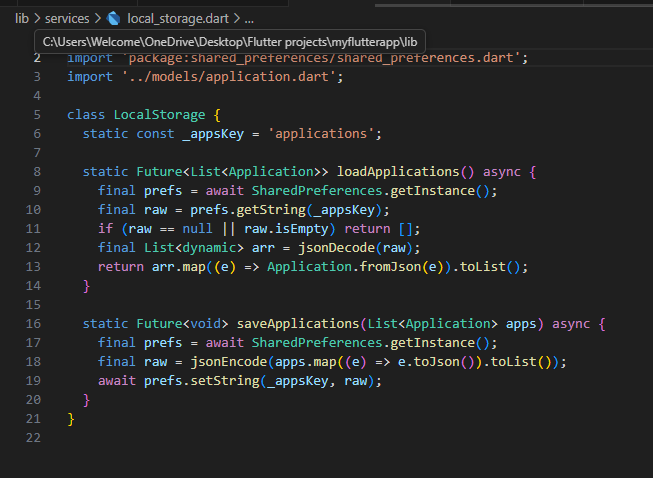
The sample test cases and scripts demonstrate how Flutter’s testing framework validates both UI and logic in isolation.  
By implementing effective widget tests, the app achieves higher reliability, better performance, and long-term maintainability.

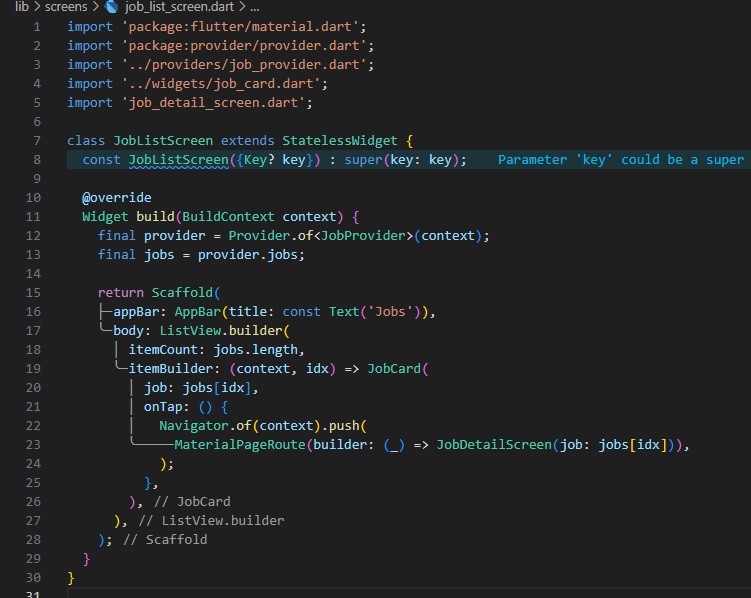
Overall, the testing phase confirms that the **Job Listings & Apply App** meets its intended functionality, ensuring a seamless user experience and readiness for deployment.

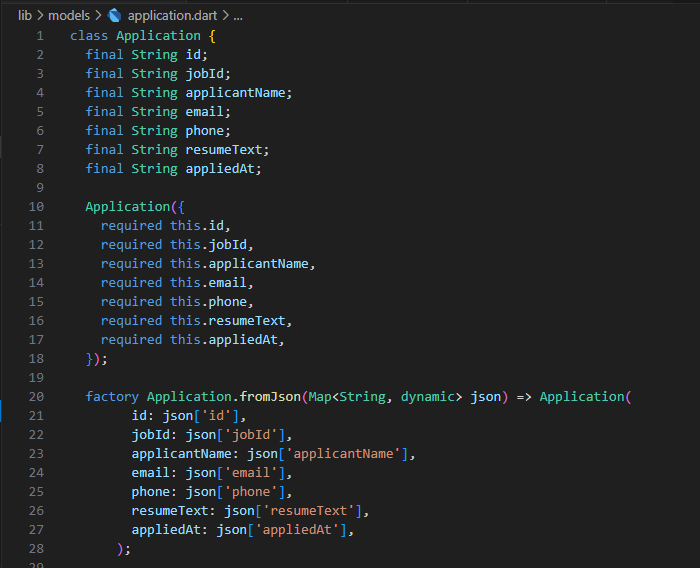
**Chapter 7**

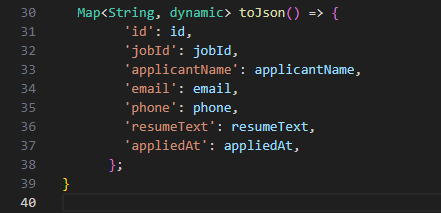
**Code SNAPSHOT DESCRIPTION**

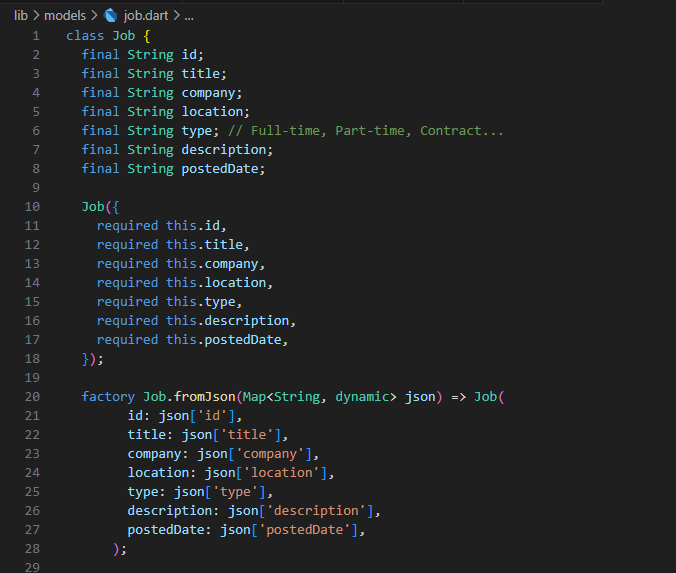
****

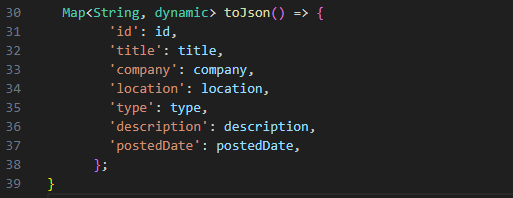
****

****

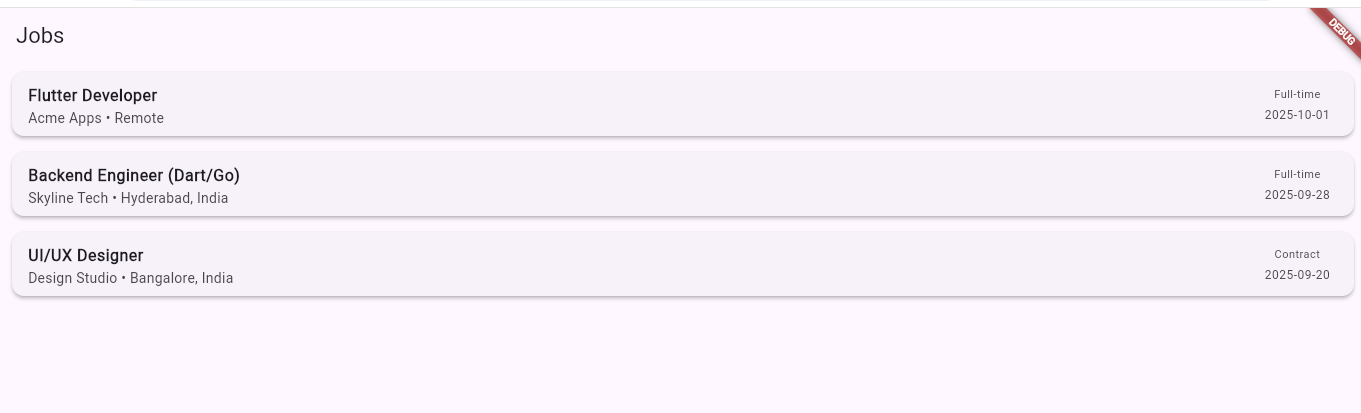
****

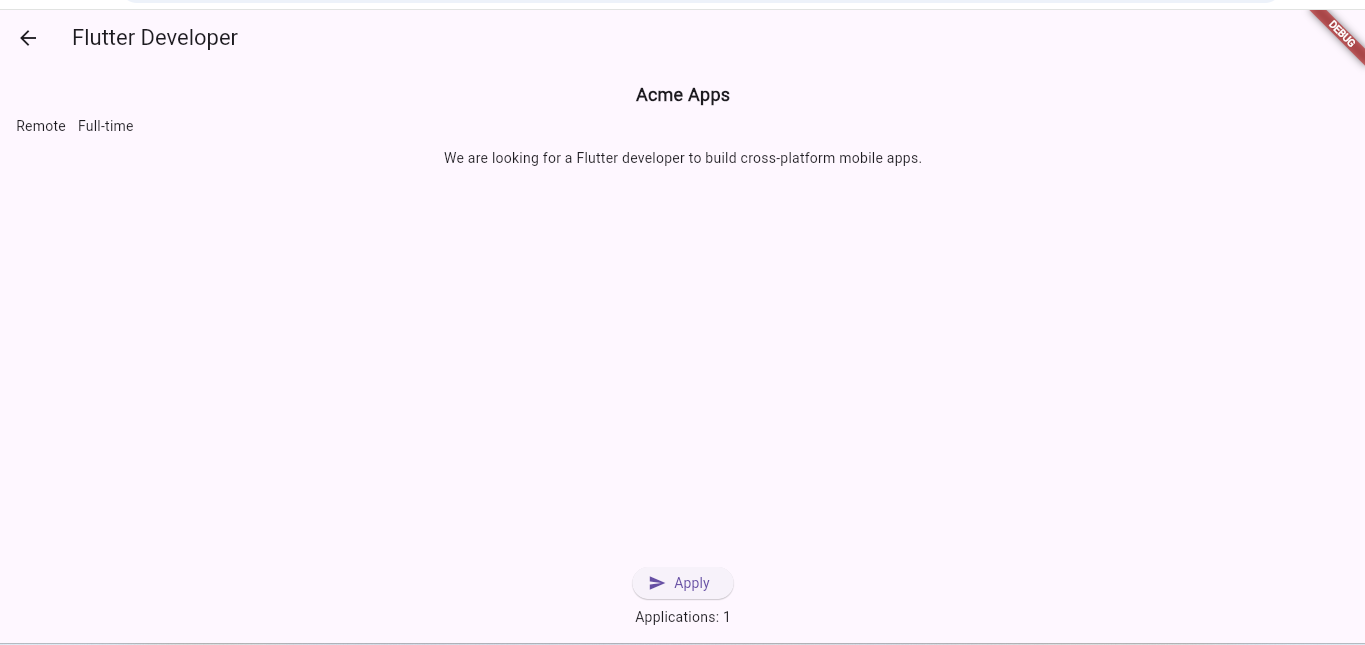
****

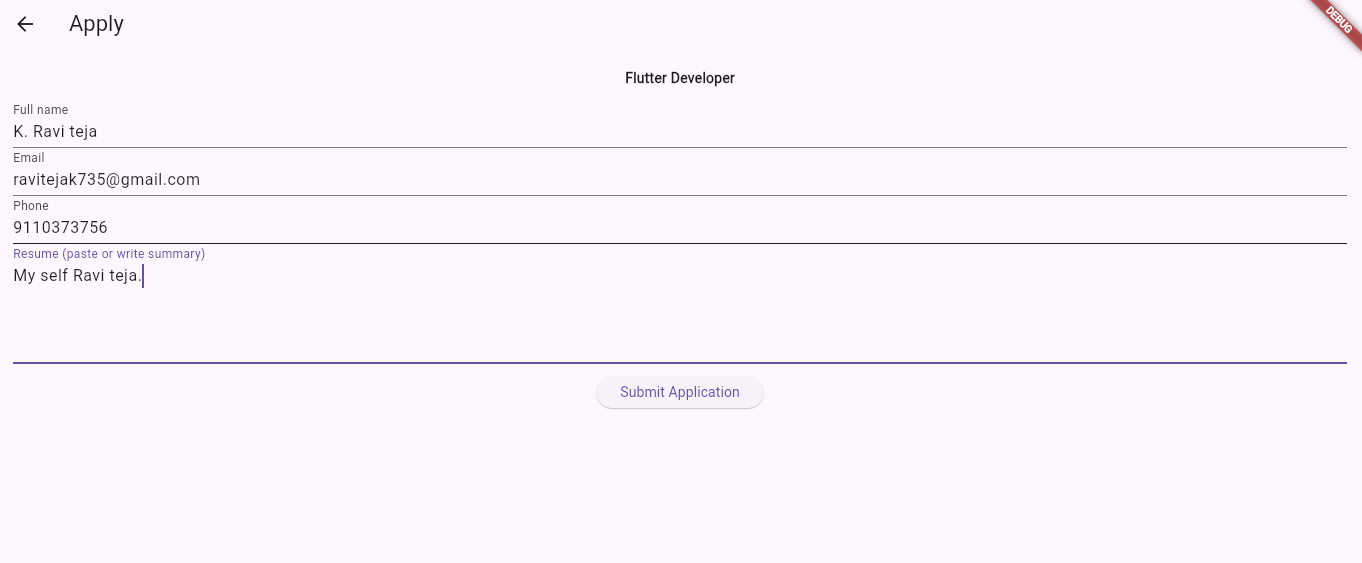
****

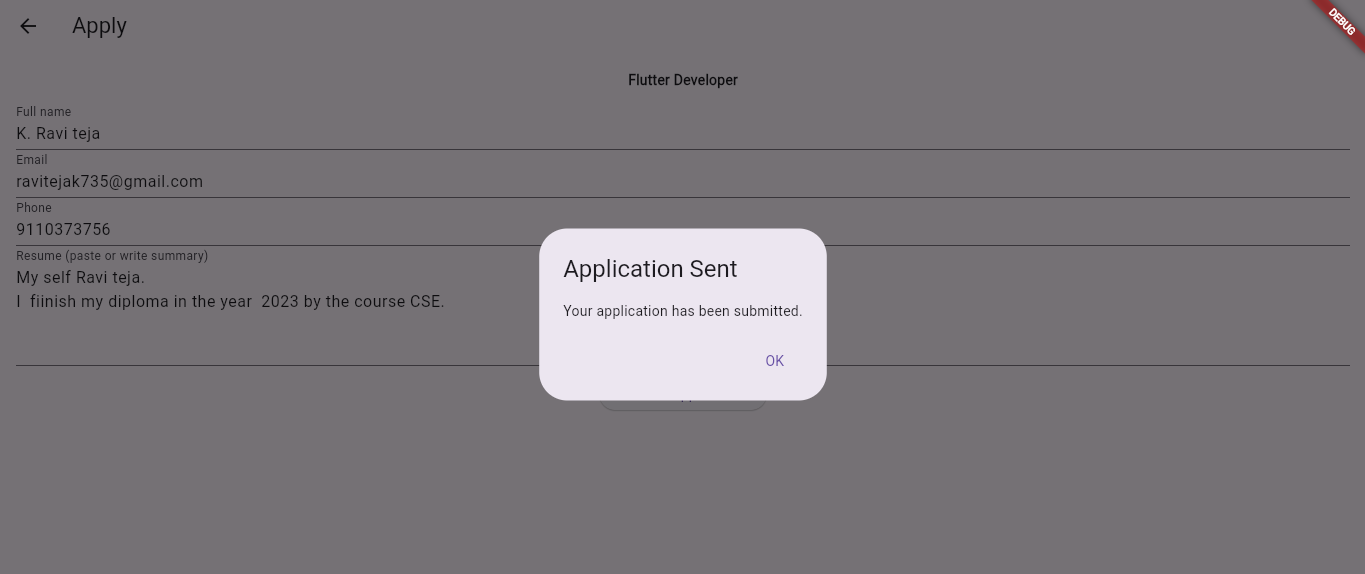
****

**OUTPUT**

****

****

****

****

**Chapter 8**

**CONCLUSION AND FUTURE ENHANCEMENT**

## **8.1 Conclusion**

The **Job Listings & Apply App using Flutter and Dart** is a cross-platform mobile application designed to simplify the process of job searching and application submission. The project successfully demonstrates the use of **Flutter** for UI development and **Dart** for application logic, providing a seamless and interactive user experience.

Key achievements of the project include:

1. **User-Friendly Interface:** The app provides an intuitive and visually appealing interface for browsing job listings, viewing detailed job information, and applying for jobs.
2. **Cross-Platform Compatibility:** The single Flutter codebase ensures smooth functionality on both Android and iOS devices.
3. **Local Data Storage:** Submitted applications are persistently stored using **SharedPreferences**, allowing users to access application history offline.
4. **State Management:** Efficient state management using the **Provider** package ensures dynamic updates and seamless navigation between screens.
5. **Robust Implementation:** The application is modular and scalable, supporting future enhancements like cloud integration, authentication, and advanced filtering.

Overall, the project fulfills its objectives by providing a functional, reliable, and maintainable solution for job seekers, while also demonstrating modern mobile app development practices.

## **8.2 Future Scope**

The **Job Listings & Apply App** has significant potential for expansion and enhancement. Possible future developments include:

1. **Integration with Online Databases:**
   * Connect the app to cloud databases such as **Firebase Firestore** or **MongoDB** to fetch live job postings and store application data online.
2. **User Authentication:**
   * Implement **sign-up/sign-in functionality** using email, phone, or social media accounts to personalize user experience and secure data.
3. **Advanced Job Search and Filtering:**
   * Add search functionality based on job title, location, salary, and company.
   * Implement filters for full-time, part-time, or remote jobs.
4. **Recruiter Dashboard:**
   * Develop a separate module for recruiters to post jobs, review applications, and track candidate status.
5. **Push Notifications:**
   * Notify users about new job postings, application updates, or deadlines in real-time.
6. **Resume Upload and Storage:**
   * Allow users to upload their resumes (PDF/Doc) and attach them to applications.
7. **Analytics and Recommendations:**
   * Implement AI-based job recommendations based on user preferences and skills.
   * Provide analytics to track application success rate and user engagement.

By implementing these features, the app can evolve into a **comprehensive recruitment platform**, bridging the gap between job seekers and employers while providing a scalable and modern solution for career development.

# **References**

1. Google Developers. Flutter Documentation. <https://flutter.dev/docs>
2. Dart Language. Dart Programming Language Overview. <https://dart.dev/guides>
3. Chatgpt: <https://chatgpt.com/share/>