INDEX

ABSTRACT

This mobile application is designed to simplify the recruitment, registration, and management of migrant workers for local job markets by connecting contractors, workers, and job providers in one unified platform. Contractors, upon registering with personal details and official documents, can create worker profiles that include each worker's skills, salary, and insurance information. They can assign jobs, track work progress, receive job notifications, post updates, and manage their workforce, including adding, removing, or exchanging workers with other contractors.

Workers, after creating profiles with identity verification documents, can apply to join contractor teams, request contractor changes, update residence and emergency contact information, and access travel assistance to job sites. They can also view their work history, which is accessible to contractors and job providers when considering them for new tasks. Job providers register to post jobs with specific requirements, and an AI feature assists by providing cost estimates, necessary worker numbers, and job duration based on job details. They can request workers from contractors, rate contractors based on performance, and report any worksite accidents directly to the contractor responsible.

Admins oversee user activities and maintain system integrity by monitoring registered workers, contractors, and job providers. They have the ability to review user details, delete accounts when necessary, and access reports on fraudulent activities, scams, or work-related concerns submitted by users.

The application integrates a messaging system for seamless communication among all users and provides notifications to keep everyone

updated on job statuses, requests, and important updates. By enhancing transparency, streamlining workforce organization, and improving collaboration, this system creates a structured and efficient local job management solution.

INTRODUCTION

INTRODUCTION

This mobile application is designed to streamline the recruitment, registration, and management of migrant workers in local job markets by connecting contractors, workers, and job providers on a single platform. The goal is to simplify workforce management, from hiring and onboarding to job assignment and progress tracking, enhancing collaboration and transparency in the process. By digitizing these key functions, the app eliminates traditional inefficiencies and delays in workforce coordination.

Contractors can create worker profiles that include skills, salary, and insurance information, assign jobs, track progress, and manage their workforce by adding or exchanging workers. Workers can create profiles with verified documents, join contractor teams, request changes, and update personal information like residence and emergency contacts. They also gain access to their work history, which helps them build credibility and increases their chances of being selected for future jobs. Job providers can post jobs with specific requirements, receive AI-generated estimates for costs, worker numbers, and job duration, and request workers from contractors. They can also rate contractors and report accidents. The AI feature in particular helps job providers make informed decisions, saving time and resources.

The application features messaging and notifications to ensure clear communication and job status updates, ultimately improving workforce organization and collaboration in the local job market. Its comprehensive design makes it a vital tool for creating a structured and efficient work environment.

SYSTEM STUDY

- 3.1. PROBLEM STATEMENT
- 3.2. EXISTITING SYSTEM
- 3.3. PROPOSED SYSTEM

3.1. PROBLEM STATEMENT

In local job markets, the recruitment and management of migrant workers are often inefficient, fragmented, and prone to miscommunication. Contractors face challenges in managing their workforce, tracking worker progress, and ensuring smooth job assignments, while workers struggle with accessing job opportunities, updating personal information, and building a reliable work history. Job providers encounter difficulties in posting job requirements, selecting suitable workers, and ensuring timely completion of tasks, often leading to delays, increased costs, and lack of transparency.

Existing systems for managing migrant workers are often outdated and lack integration, leading to inefficiencies in communication, worker tracking, and job assignment. Additionally, there is no unified platform that connects contractors, workers, and job providers, making it difficult for all parties to collaborate effectively and manage their roles efficiently. This creates a need for a digital solution that simplifies the recruitment, registration, and management of migrant workers, offering a more streamlined, transparent, and collaborative approach to local job market operation.

3.2. EXISTITING SYSTEM

Currently, the recruitment and management of migrant workers rely on traditional methods like manual paperwork, phone calls, and fragmented digital systems. Contractors manage workers using spreadsheets or isolated software, while workers often find jobs through informal networks. Job providers post openings via local agencies or online boards, but there is no seamless connection with contractors or workers. Communication is typically handled through phone calls or separate messaging apps, leading to inefficiencies and delays.

While some platforms exist for job posting or contractor management, they lack integration and essential features like real-time progress tracking, AI based estimates, and centralized worker profiles, making the current systems inefficient and prone to errors.

3.3. PROPOSED SYSTEM

- **Centralized Platform:** A unified mobile application that connects contractors, workers, and job providers in one platform, simplifying recruitment, registration, and workforce management.

- **Contractor Features:** Contractors can create and manage worker profiles, track work progress, assign jobs, and handle workforce updates (e.g., adding or exchanging workers). They can also receive notifications and job updates.

- **Worker Features:** Workers can create profiles with verified identity documents, apply to join contractor teams, update personal information, and view their work history. They can request contractor changes and access travel assistance.

- **Job Provider Features:** Job providers can post job requirements, use AI-based features for cost estimates, estimated number of workers required, and estimated time for job completion. They can post jobs, review contractors, and report of accidents if any.

- **AI Integration:** AI-powered chat bot assist job providers by generating cost estimates, number of workers required, and job duration based on job details, improving decision-making.

- **Improved Transparency and Efficiency:** A comprehensive solution for managing job assignments, workforce tracking, and collaboration, reducing miscommunication and delays in the local job market.

- **Mobile Accessibility:** Easy access to all features via mobile devices, ensuring real-time updates and task management on the go.

- **Messaging and Notifications:** In-app messaging and notifications for smooth communication between all users, keeping everyone informed of job statuses, requests, and updates.

MODULE DESCRIPTION

4.1. MODULE DESCRIPTION

1. ADMIN

The Admin module enables administrators to oversee user activities and maintain system integrity by managing workers, contractors, and job providers. Admins can access and review user details, ensuring compliance with platform guidelines. They have the authority to delete accounts in cases of fraudulent activity, misconduct, or policy violations. The module also includes a reporting system where admins can view and assess reports submitted by users regarding scams, fraudulent activities, and disputes. By monitoring platform interactions, the Admin module helps create a secure and transparent environment, ensuring smooth operations and trust among all users.

2. CONTRACTOR

The Contractor module is designed to help contractors efficiently manage their workforce and job assignments. Contractors can create and update worker profiles, which include essential details such as skills, salary, and insurance information. They can add or remove workers from their team and track the progress of each worker's tasks. Contractors can assign specific jobs to workers based on their skills and availability, ensuring that the right person is matched to the right task. Additionally, contractors can post updates regarding ongoing jobs, allowing both job providers and workers to stay informed. Notifications are sent to workers about new job assignments, changes in tasks, or job status updates to ensure seamless communication.

3. JOB PROVIDER

The Job Provider module enables job providers to post job openings and request workers for specific tasks. Job providers can specify the requirements for each job, including skills, number of workers, and the estimated duration and cost of the task. The system integrates AI to provide job providers with cost estimates, the number of workers needed, and the expected job duration based on the job details provided. Job providers can view available workers and their profiles, including previous work history, to make informed decisions. After a job is completed, job providers can rate contractors based on their performance and quality of work. Additionally, job providers can report any accidents or safety issues that occur on-site, ensuring timely follow-up by the contractor responsible.

4. WORKER

The Worker module allows workers to create and manage profiles with personal information, identity verification, skills, and contact details. They can update their residence and emergency contacts and view their work history, which is accessible to contractors and job providers. Workers can apply to join contractor teams, request changes, and apply for job openings posted by job providers. The module also provides travel assistance for reaching job sites. Through in-app messaging, workers can communicate with contractors and job providers, while notifications keep them updated on job assignments, changes, and status updates.

FEASIBILITY STUDY

- 5.1 OPERATIONAL FEASIBILITY
- 5.2 TECHNICAL FEASIBILITY
- 5.3 ECONOMIC FEASIBILITY
- 5.4 BEHAVIORAL FEASIBILITY

FEASIBILITY STUDY

A feasibility study is a preliminary study undertaken to determine and document a project's viability. The results of this study are used to make a decision whether to proceed with the project. If it indeed leads to a project being approved, it will – before the real work of proposed project starts – be used to as certain the likelihood of the project's success. It is an analysis of possible alternative solutions to a problem and recommendation on the best alternative. It, for example can decide.

Whether an order processing be carried out by a new system more efficiently than the previous one. The feasibility study proposes one or more conceptual solutions to the problem set for the project. The conceptual solution gives an idea of what the new system will look like. They define what will be done on the computer and what will remain manual. It also indicates what input will be needed by the system and what outputs will be produced. These solutions should be proven feasible and a preferred solution is accepted .

The feasibility study environment enables all alternatives to be discussed and evaluated. This phase starts with an identification of the main characteristics of the required system. During this stage it is important to collect information as much as possible about the software package that might meet the specification from as many sources as possible.

Facts considered in the feasibility analysis were:

- Operational Feasibility
- Technical Feasibility
- Economic Feasibility
- Behavioral Feasibility

The requirements of the system are specified with a set of constraints such as system objectives and the description of the out puts. Three key factors are to be considered during the feasibility study.

5.1. OPERATIONAL FEASIBILITY

The proposed project is said to be beneficial only if they can be modules and carved out in to a system that will meet all the requirement. The best tor the feasibility prohibits if the system would run without faults when it is deployed or there are any major hurdles to be the implementations of the proposed system. The proposed system is not supposed to cause any harm to the user or the computer that is being used and the system is safe and secure.

5.2. TECHNICAL FEASIBILITY

The proposed system meets all the requirement of the Technical feasibility. Because of the implementation of the project need no technological difficult. The web part of the project can be easily implemented in a user friendly IDE, that is in Visual Studio Code.

So the development of the project does not meet any type of the technical difficult. Hence we can be say that the proposed system is technically feasible.

5.3. ECONOMIC FEASIBILITY

The proposed project is economically feasible. Because once the system is put into its use in the current market the system provides economical advantage to the firm. Also the firm can afford the cost to implement the project. The proposed project provides tangible and intangible benefits comparing to the existing projects. This system does not need any initial investments and it can improve the quality of service.

5.4 BEHAVIORAL FEASIBILITY

The proposed system has high operational feasibility. Because the site is accessible anywhere and anytime. For operating the application, we only need an internet connection and a smartphone. The changes in the hardware or software environment do not affect the system.

People inherently change the computer or mobile phones. But the changes in the environment do not affect the system.

SYSTEM REQUIREMENT SPECIFICATION

- 6.1. SOFTWARE SPECIFICATION
- 6.2. HARDWARE SPECIFICATION

6. SYSTEM SPECIFICATION

Hardware and software requirements for the installation and smooth functioning of this product could be configured based on the requirements needed by the component of the operating environment that works as a front-end system. Here we suggest minimum configuration for both hardware and software. It includes two phases.

6.1. Software Requirements

6.2. Hardware Requirements

6.1. Software Specification

One of the most difficult task is selecting software for the system, once the system requirements is found out then we have to determine whether a particular software package fits for those system requirements. The application requirement:

•	Front end	:	Flutter
•	Back end	:	Firebase, AI
•	Operating system	:	Windows 7 or above
•	IDE	:	Visual Studio Code

6.2. Hardware Specification

The selection of hardware is very important in the existence and proper working of software. The selection hardware, the capacity and speed are also important.

• Processor	:	Intel Pentium Core i3 and above
• Primary Memory	:	8GB RAM and above
• Storage	:	320 GB hard disk and above
• Display	:	VGA Color Monitor
• Key Board	:	Windows compatible
• Mouse	:	Windows compatible

SYSTEM DESIGN

- 7.1. INITIAL DESIGN
- 7.2. INPUT DESIGN
- 7.3. OUTPUT DESIGN
- 7.4. DATA FLOW DIGRAM
- 7.5. DATABASE DESIGN

7. SYSTEM DESIGN

System design is the process of developing specifications for a candidate system that meet the criteria established in the system analysis. The major step in system design is the preparation of the input forms and the output reports in a form applicable to the user.

The system design is the most creative and challenging phase. The first step is to determine how the output is produced and in what format. Samples of input and output are presented. Next, the input data and the master data are to be designed to meet the requirements of the proposed output. The operational phases are handled through program construction testing, including a list of programs needed to meet the system objective and complete documentation.

7.1. INITIAL DESIGN

The initial design of the app is focused on developing a user-friendly interface that meets user requirements efficiently. For that purpose, we have explored well-known technologies and identified Flutter as the ideal framework for development. Flutter provides a rich set of pre-built widgets that allow for a highly customizable and visually appealing interface while ensuring a consistent user experience across different platforms.

We gathered requirements for this application through various approaches and found that Flutter's widget-based UI framework effectively enables the creation of a responsive and aesthetically pleasing interface. The application is built using the Dart programming language, ensuring efficient performance and seamless integration of functionalities.

7.2. INPUT DESIGN

Input design is the process of converting the user originated inputs to a computer format. The input design involves determining what the inputs are, how the data should be performed, how to validate data, how minimize data entry and how to provide a multiuser facility. The design for handling input specifies how data are accepted for computer processing. Input design is a part of overall system design that needs careful attention and if includes specifying the means by which actions are taken.

A system user interacting through a system must be able to tell the system whether to accept input produce a report or end processing. The collection of input data is considered to be most expensive part of the system design. Since the inputs have to be planned in such a manner so as to get the relevant information extreme care is taken to obtain the information. If the data going into the system is incorrect then processing and outputs will magnify this error. All input data are validated in the order and if any data violates any conditions, the user is warned by a message. If the data satisfies all the conditions, then it is transferred to the appropriate tables in the database.

We have to keep in mind the following things to design the system

- What data to input?
- What medium to use?
- The dialogue to guide users in providing input.
- Methods for performing input validation and steps to follow when errors occur Input requirement gathering was one of the major trivial process in web or android application development. The project involves text inputs. The inputs can be entered through keyboard and mouse. The text input is gathered by forms with text boxes.

7.3. OUTPUT DESIGN

Effective output design will improve the clarity and performance of output. Output design phase of the system is concerned with the convergence of information's to the end user friendly manner. The output design should be efficient, intelligible so that system relationship with the end user is improved and thereby enhancing the process of decision making.

The Outputs from the AIR DEFENCE application consist of different formats. It provides search details in text format. The most attractive feature of this application is it shows the results of search contents after efficient filtering and pruning techniques.Efficient and eligible output design should improve the system's relationship with the user and help in decision making.

Outputs are the most important and direct source of information to the user and to the management. Output design generally deals with the results generated by the system.

7.4. DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation of ―flow‖ of data through an information system, modelling its process aspects. Often they are preliminary step used to create the over view of a system which can later be elaborated. DFD also be used for visualization of data processing (Structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to and where the data will be stored. It does not show information about the timing of process, or information about whether process will operate in sequence or in parallel.

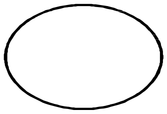
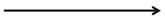
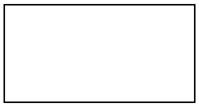
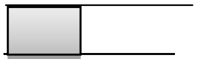
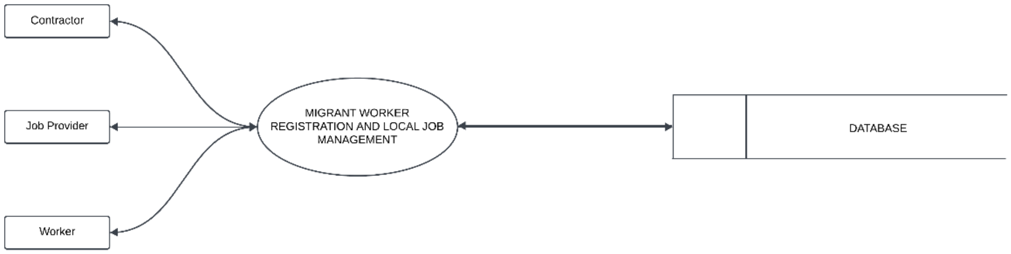
Symbol	Name	Function
	Process	Performs some transformation of input data to yield output data.
	Data Flow	Used to connect processes to each other, to sources or sinks; the arrow head indicates direction of data flow.
	Source or sink(External Entity)	A source of system inputs or sinks of system outputs.
	Data store	A repository of data.place where data is stored.

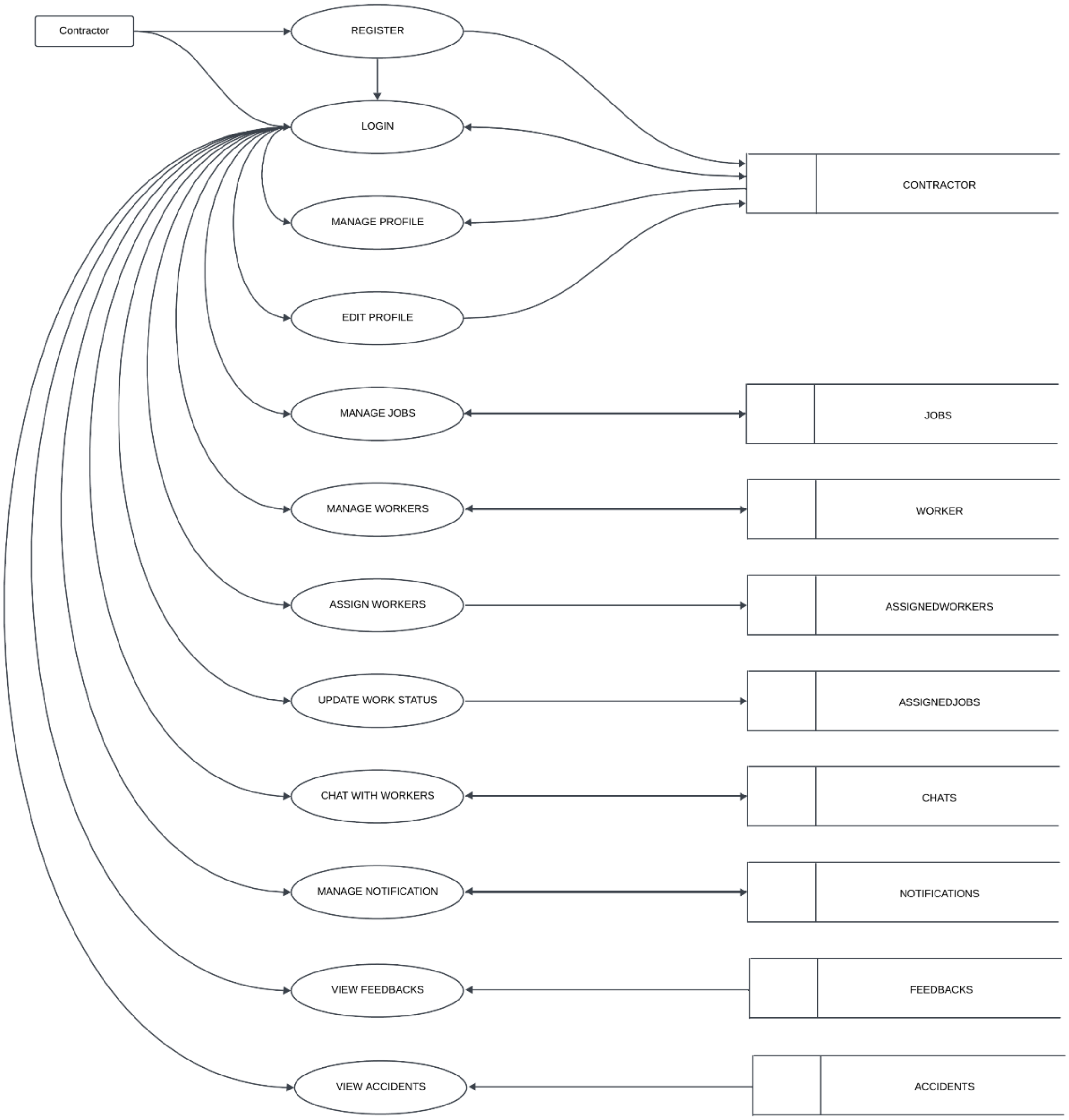
Fig: *Above depicted are the major shapes used in DFD.*

The DFD at the simplest level is referred to as the context analysis diagram. These are referred to as the explaining its process in detail. Processes are numbered for easy identification and the data stores, source and destination of data are normally labeled in block letters. Each data flow is labeled for easy understanding.

LEVEL 0

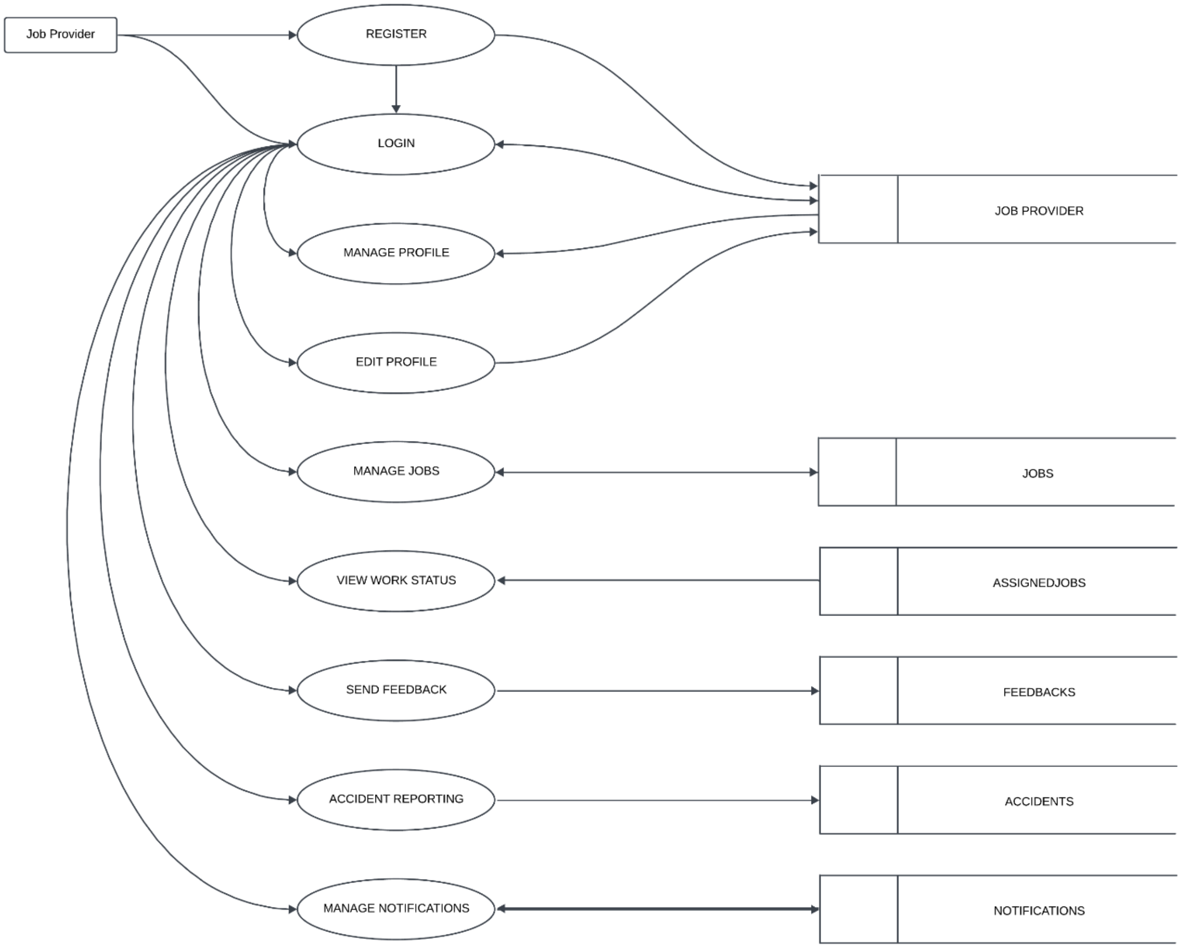

Contractor
Job Provider
Worker
MIGRANT WORKER REGISTRATION AND LOCAL JOB MANAGEMENT
DATABASE

LEVEL 1

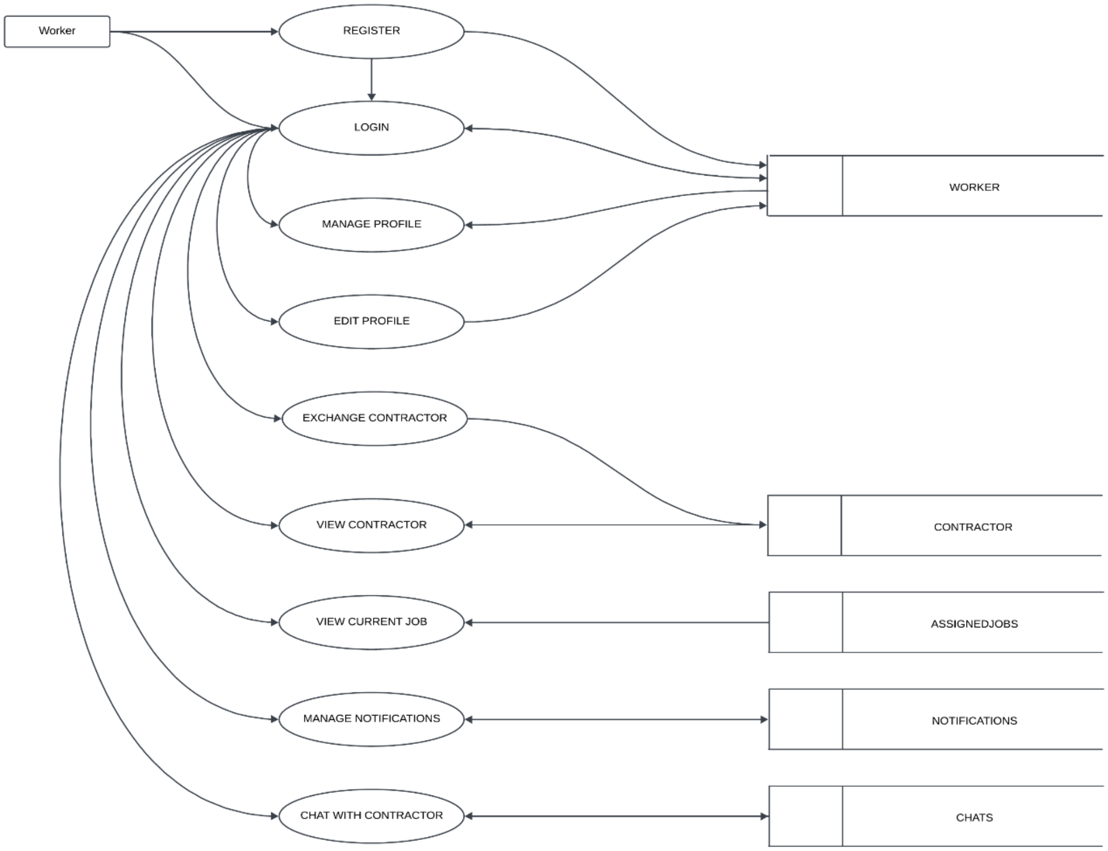


Contractor
REGISTER
LOGIN
MANAGE PROFILE
EDIT PROFILE
MANAGE JOBS
MANAGE WORKERS
ASSIGN WORKERS
UPDATE WORK STATUS
CHAT WITH WORKERS
MANAGE NOTIFICATION
VIEW FEEDBACKS
VIEW ACCIDENTS
CONTRACTOR
JOBS
WORKER
ASSIGNEDWORKERS
ASSIGNEDJOBS
CHATS
NOTIFICATIONS
FEEDBACKS
ACCIDENTS

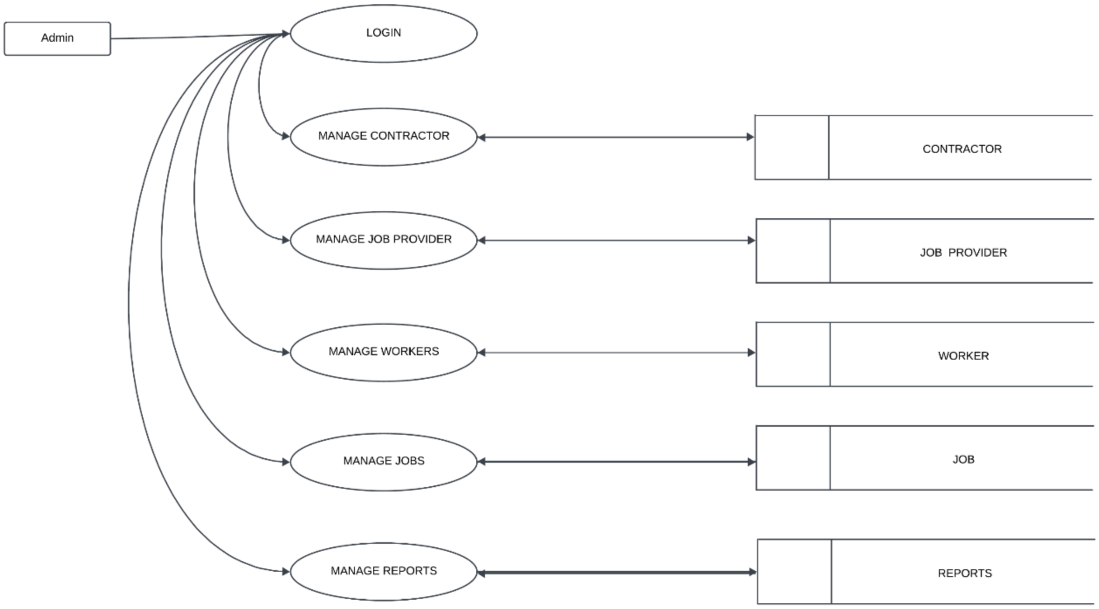
LEVEL 2


Job Provider
REGISTER
LOGIN
MANAGE PROFILE
EDIT PROFILE
MANAGE JOBS
VIEW WORK STATUS
SEND FEEDBACK
ACCIDENT REPORTING
MANAGE NOTIFICATIONS
JOB PROVIDER
JOBS
ASSIGNEDJOBS
FEEDBACKS
ACCIDENTS
NOTIFICATIONS

LEVEL 3


Worker
REGISTER
LOGIN
MANAGE PROFILE
EDIT PROFILE
EXCHANGE CONTRACTOR
VIEW CONTRACTOR
VIEW CURRENT JOB
MANAGE NOTIFICATIONS
CHAT WITH CONTRACTOR
WORKER
CONTRACTOR
ASSIGNEDJOBS
NOTIFICATIONS
CHATS

LEVEL 4


Admin
LOGIN
MANAGE CONTRACTOR
MANAGE JOB PROVIDER
MANAGE WORKERS
MANAGE JOBS
MANAGE REPORTS
CONTRACTOR
JOB PROVIDER
WORKER
JOB
REPORTS

7.4. DATABASE DESIGN

Database design is crucial for managing large amounts of structured and unstructured data. A database is a collection of related data that ensures data consistency, security, and efficient retrieval. In this project, Firestore, a NoSQL document-oriented database, is used instead of a traditional SQL database. Firestore organizes data into collections and documents rather than tables and rows, allowing for scalability, real-time synchronization, and reduced redundancy.

Unlike relational databases, Firestore does not require predefined schemas, making it adaptable and flexible for dynamic applications. It provides high availability and allows multiple users to access data simultaneously without conflicts.

DATA STRUCTURING IN FIRESTORE

Firestore does not follow traditional normalization techniques used in relational databases. Instead, data structuring principles are applied to ensure minimal redundancy, efficient data retrieval, and logical organization. The different structuring methods used in this project are:

FIRST NORMAL FORM (1NF) Equivalent

A collection in Firestore is said to be in 1NF if it contains only atomic (indivisible) values and avoids multi-valued or nested attributes. This ensures that data remains structured and easily retrievable. In this project, data is structured to ensure that no repeating fields exist within a document, multi-valued attributes are stored as arrays, and composite attributes are flattened into individual fields. First normal form principles are applied across all collections in the database.

SECOND NORMAL FORM (2NF) Equivalent

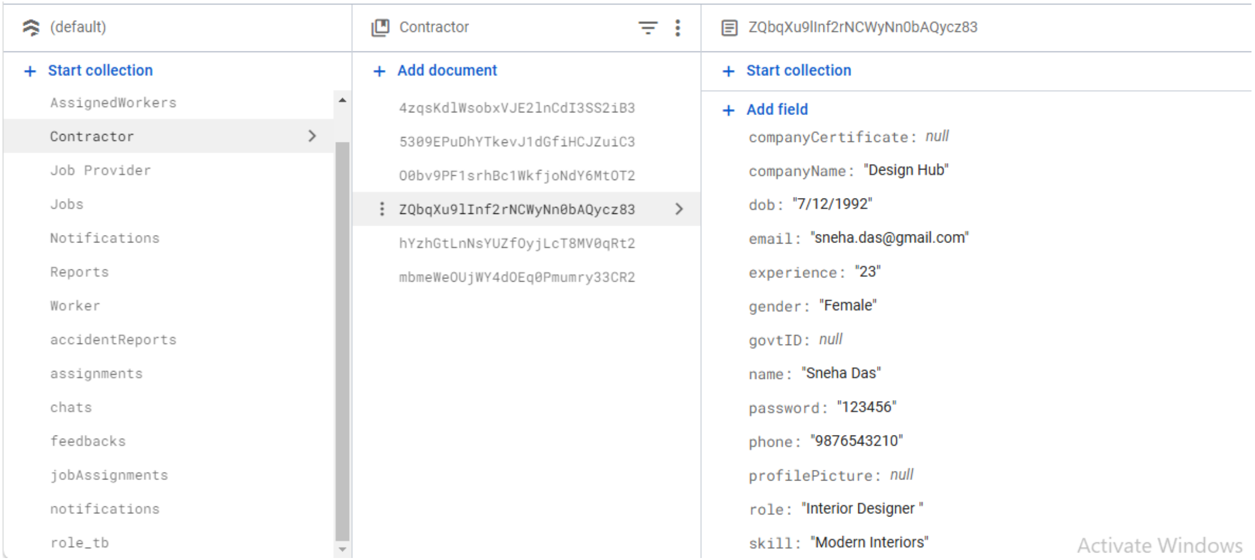
A collection in Firestore is said to be in 2NF if it satisfies the 1NF conditions and ensures that non-primary data is not dependent on a single field. Instead of storing all related data in a single document, Firestore structures data using subcollections or separate documents to improve scalability and minimize redundancy. This ensures that each document contains only relevant information and references other collections when necessary. Since Firestore uses document IDs and references instead of primary keys, relationships between different collections are maintained through structured references rather than foreign keys. Second normal form structuring is followed in all collections in this database.

THIRD NORMAL FORM (3NF) Equivalent

A collection in Firestore is said to be in 3NF if it satisfies all 2NF conditions and ensures that non-key attributes do not depend on other non-key attributes. This is achieved by eliminating indirect dependencies and ensuring that data is either embedded within a document for frequent access or stored separately if it changes independently. Instead of storing detailed related information inside a document, Firestore uses references to maintain efficient data relationships. This eliminates unnecessary duplication and ensures data consistency across collections. Third normal form structuring is applied in this database to maintain an optimized and scalable data model.

COLLECTION STRUCTURE

- **Collection : Contractor**



companyCertificate: *null*

companyName: "Design Hub"

dob: "7/12/1992"

email: "sneha.das@gmail.com"

experience: "23"

gender: "Female"

govtID: *null*

name: "Sneha Das"

password: "123456"

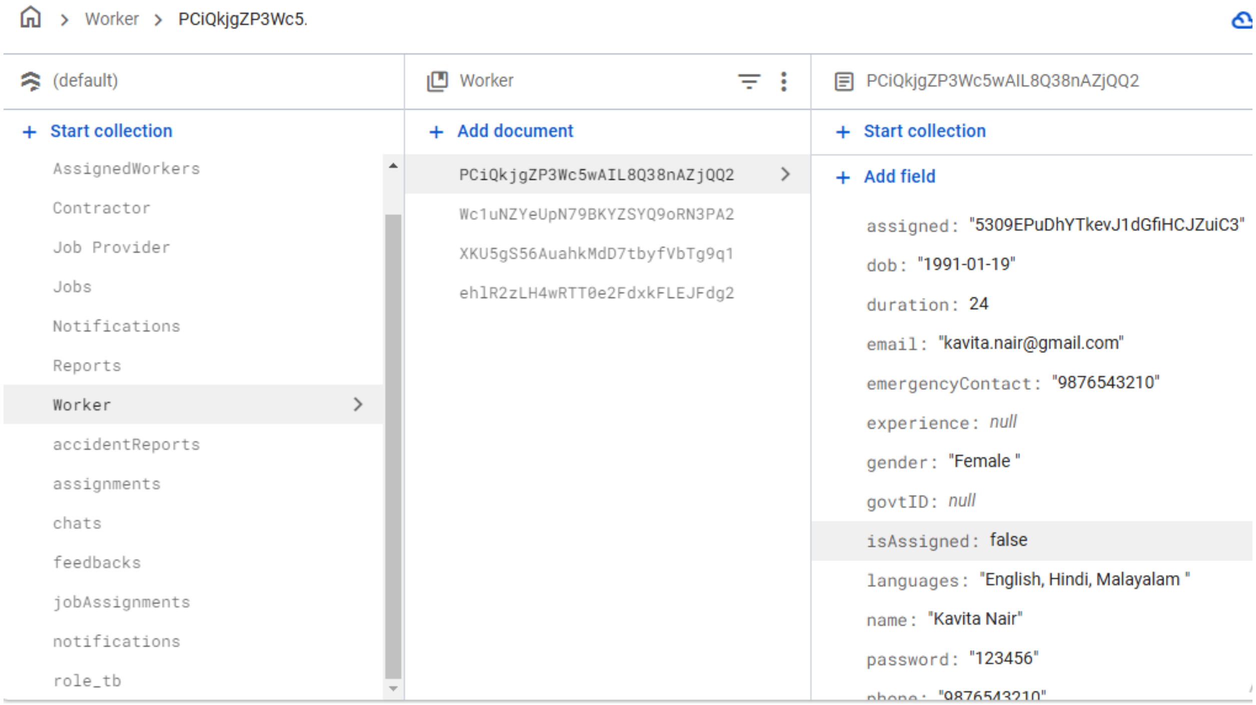
phone: "9876543210"

profilePicture: *null*

role: "Interior Designer "

skill: "Modern Interiors"

- **Collection : Worker**



assigned: "5309EPuDhYTkevJ1dGfiHCJZuiC3"

dob: "1991-01-19"

duration: 24

email: "kavita.nair@gmail.com"

emergencyContact: "9876543210"

experience: *null*

gender: "Female "

govtID: *null*

isAssigned: false

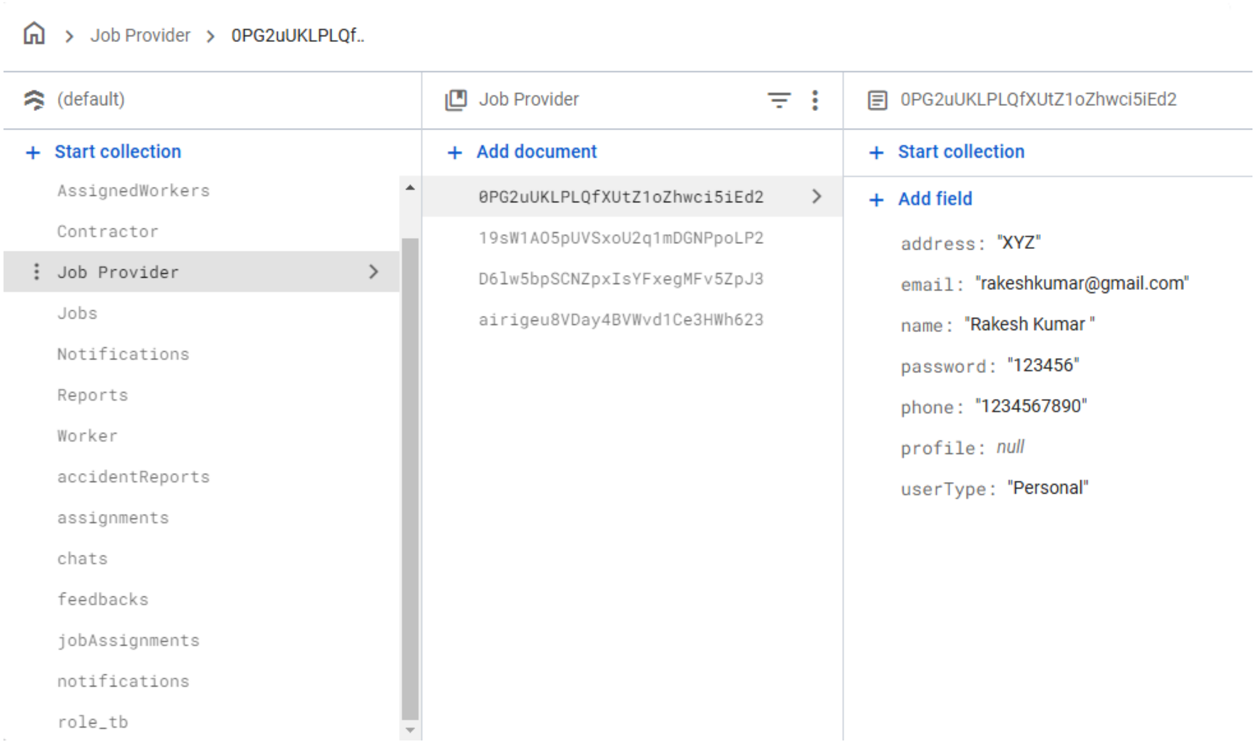
languages: "English, Hindi, Malayalam "

name: "Kavita Nair"

password: "123456"

phone: "9876543210"

- **Collection : Job Provider**

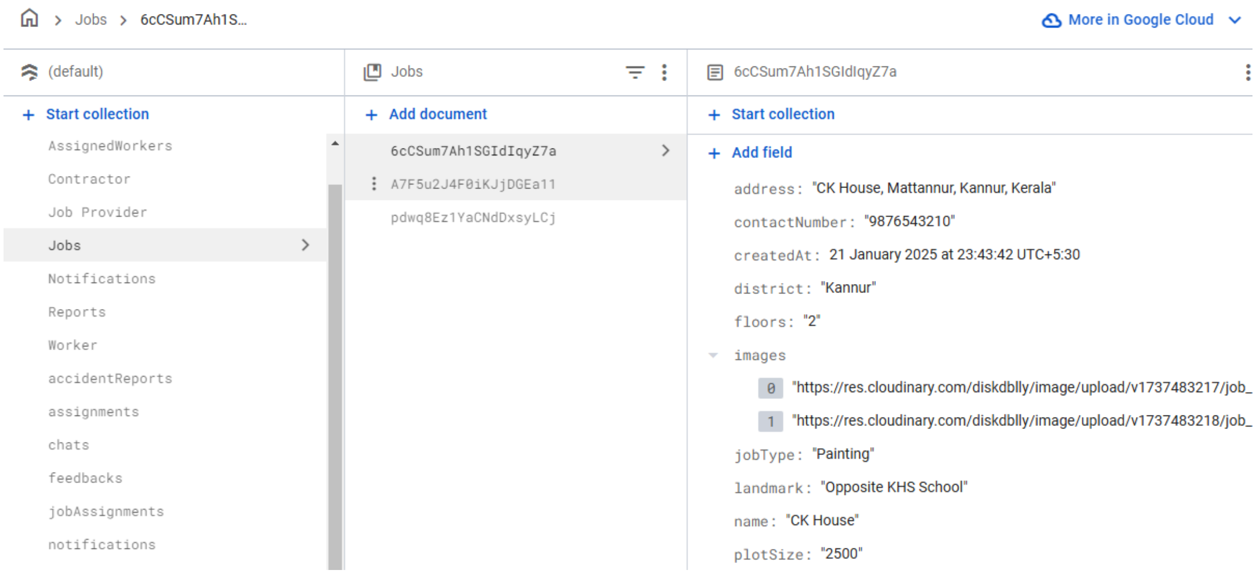


AssignedWorkers
Contractor
Job Provider
Jobs
Notifications
Reports
Worker
accidentReports
assignments
chats
feedbacks
jobAssignments
notifications
role_tb

0PG2uUKLPLQfXUtZ1oZhwci5iEd2
19sW1AO5pUVSxoU2q1mDGNPpoLP2
D6lw5bpSCNZpxIsYFxegMFv5ZpJ3
airigeu8VDay4BVWvd1Ce3HWh623

```
address: "XYZ"
email: "rakeshkumar@gmail.com"
name: "Rakesh Kumar "
password: "123456"
phone: "1234567890"
profile: null
userType: "Personal"
```

- **Collection : Jobs**

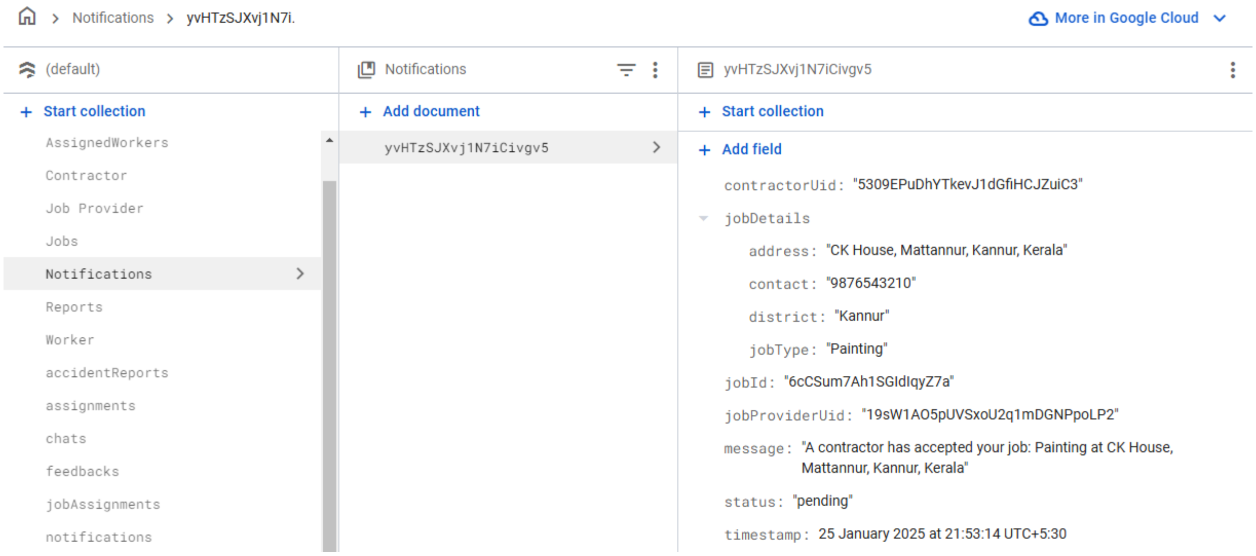


AssignedWorkers
Contractor
Job Provider
Jobs
Notifications
Reports
Worker
accidentReports
assignments
chats
feedbacks
jobAssignments
notifications

6cCSum7Ah1SGIdIqyZ7a
A7F5u2J4F0iKJjDGEa11
pdwq8Ez1YaCNdDxsyLCj

```
address: "CK House, Mattannur, Kannur, Kerala"
contactNumber: "9876543210"
createdAt: 21 January 2025 at 23:43:42 UTC+5:30
district: "Kannur"
floors: "2"
images
    0 "https://res.cloudinary.com/diskdblly/image/upload/v1737483217/job_
    1 "https://res.cloudinary.com/diskdblly/image/upload/v1737483218/job_
jobType: "Painting"
landmark: "Opposite KHS School"
name: "CK House"
plotSize: "2500"
```

- ## Collection : Notifications



contractorUid: "5309EPuDhYTkevJ1dGfiHCJZuiC3"

jobDetails

address: "CK House, Mattannur, Kannur, Kerala"

contact: "9876543210"

district: "Kannur"

jobType: "Painting"

jobId: "6cCSum7Ah1SGIdIqyZ7a"

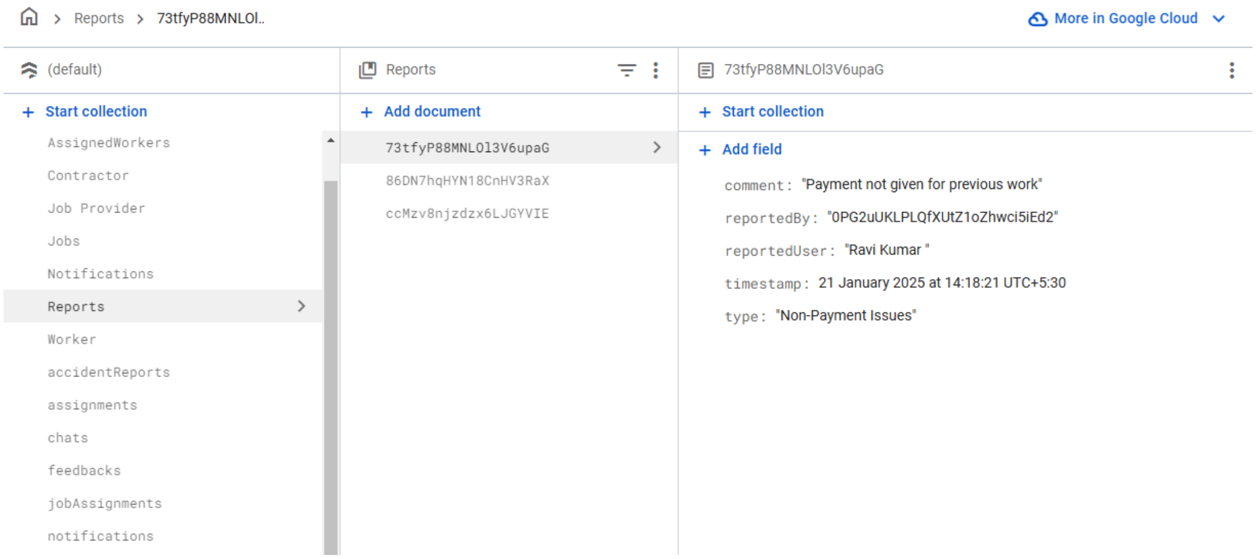
jobProviderUid: "19sW1AO5pUVSxoU2q1mDGNPpoLP2"

message: "A contractor has accepted your job: Painting at CK House, Mattannur, Kannur, Kerala"

status: "pending"

timestamp: 25 January 2025 at 21:53:14 UTC+5:30

- ## Collection : Reports



comment: "Payment not given for previous work"

reportedBy: "0PG2uUKLPLQfXUtZ1oZhwci5iEd2"

reportedUser: "Ravi Kumar "

timestamp: 21 January 2025 at 14:18:21 UTC+5:30

type: "Non-Payment Issues"

SYSTEM DEVELOPMENT

- 8.1. FRONT END
- 8.2. BACK END

8. SYSTEM DEVELOPMENT

8.1. FRONT END:

DART

Dart is an open-source, general-purpose programming language developed by Google. First released in 2011, Dart is designed for client-side development, with a focus on building high-performance applications for web, mobile, and desktop platforms. Its syntax is similar to languages like Java and JavaScript, making it accessible to developers familiar with object-oriented programming.

Dart is compiled and can execute in both just-in-time (JIT) and ahead-of-time (AOT) modes. JIT compilation enables fast development cycles with features like hot reload, while AOT compilation enhances execution speed by producing optimized native machine code. Dart supports multiple programming paradigms, including object-oriented, functional, and reactive programming.

The language features a sound type system with optional null safety, helping developers write more reliable and efficient code. Dart's standard library provides extensive built-in support for asynchronous programming through its Future and Stream APIs, making it well-suited for handling concurrent operations.

Dart was initially created as a JavaScript alternative for web applications but has since evolved into a versatile language powering frameworks like Flutter, Google's UI toolkit for cross-platform mobile and web app development. With its structured yet flexible nature, Dart continues to gain traction in modern application development.

FLUTTER

Flutter is an open-source UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. First released in 2017, Flutter enables developers to create visually rich and highly performant applications with a consistent user experience across multiple platforms.

Flutter uses the Dart programming language and employs an ahead-of-time (AOT) compilation process to generate optimized native code, ensuring smooth performance. Its reactive framework allows developers to build UIs declaratively using widgets, which serve as the building blocks for app interfaces. Flutter’s widget-based architecture provides extensive customization and flexibility, enabling developers to create expressive and dynamic user experiences.

A key feature of Flutter is its hot reload capability, which significantly speeds up the development process by allowing real-time code changes without restarting the application. Flutter also includes a rich set of pre-designed widgets that follow Material Design and Cupertino guidelines, ensuring a native-like experience on both Android and iOS.

Flutter's layered architecture promotes fast rendering and high responsiveness, making it suitable for complex animations and modern UI designs. With its growing community, extensive ecosystem, and seamless integration with backend services, Flutter has become a popular choice for developers looking to build cross-platform applications efficiently.

ANDROID STUDIO

Android Studio, developed by Google, is the official integrated development environment (IDE) for Android app development. It offers a powerful and user-friendly environment with features like an intelligent code editor, visual layout editor, and a versatile emulator for testing on various devices. Supporting both Java and Kotlin, Android Studio provides robust debugging and profiling tools for optimizing app performance. With seamless integration of version control systems and frequent updates to align with the latest Android platform advancements, it remains a preferred choice for developers to create high-quality and innovative Android applications.

8.2.BACK END:

FIRESTORE

Firestore, officially known as Cloud Firestore, is a flexible, scalable, and fully managed NoSQL database developed by Google as part of its Firebase platform. First introduced in 2017, Firestore is designed to store, sync, and query data for web and mobile applications in real time while offering seamless integration with other Firebase services and Google Cloud products.

Unlike traditional SQL databases, Firestore follows a document-oriented model, where data is stored as collections and documents rather than tables and rows. Documents, which are stored in JSON-like format, contain fields of various data types, including strings, numbers, arrays, maps, and timestamps. Collections serve as containers for documents, allowing for hierarchical data structuring that is both scalable and easy to manage.

Firestore supports real-time synchronization, ensuring that data updates are instantly reflected across all connected devices. It also offers offline support, enabling applications to read, write, and sync data even when users are offline, with automatic synchronization once connectivity is restored. Firestore's powerful querying capabilities allow developers to filter, sort, and retrieve data efficiently using indexed queries.

As a fully managed cloud-based database, Firestore is designed **to** scale automatically, handling millions of concurrent users without requiring manual server management. It also ensures strong security through Firebase Authentication and role-based access control via Firestore Security Rules.

Firestore integrates seamlessly with Google Cloud services, making it an excellent choice for serverless applications, real-time chat apps, collaborative platforms, and IoT solutions. With its combination of real-time capabilities, offline functionality, and effortless scalability, Firestore is a robust and reliable choice for modern application development.

SYSTEM TESTING

9.SYSTEM TESTING

Testing is very important in determining the reliability and efficiency of software, and hence it is very crucial stage in software development. Tests are conducted on the software to evaluate its performance at different levels.

Testing is vital to the success of the system. It is the penultimate step of software development. An elaborate testing of data is prepared and the system is using test data. While doing testing errors are noted and correction is made. The users are trained to operate the developed system. Both hardware and software are made to run the developed system successfully.

Testing is a process of executing a program with the intent of finding whether the software achieves the desired result. Tests are conducted to locate an undiscovered error. Different types of data are fed into the system and the end result is verified with the expected results. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system.

The various testing methods used for this application are:

- Unit testing
- Integration testing
- Validation testing
- Output testing

UNIT TESTING

Unit testing focuses verification efforts on the smallest unit of software design module. To check whether each module in this software works properly so that it gives desired outputs to the given inputs. All validations and conditions are tested in the module level in the unit test control parts are tested to ensure the information correctly flows into, and out of the program unit under test. Boundary conditions are tested to ensure that modules operate at the boundaries.

INTEGRATION TESTING

Data can be lost across an interface, one module have an adverse effect on the other sub-functions, when combined may not produce the desired functions. Integrated testing is the systematic to uncover the errors within the interface. This testing is done by inputting necessary values and data in the sequential order. As data of one unit is needed for working of other. The need for integrated system is to find the overall system performance.

VALIDATION TESTING

Validation test is defined with a simple definition that validation succeeds when the system functions in a manner that can be reasonably accepted by the customer. Validation is done to see whether the corresponding entries made in the tables are correct. Proper validations are done in case of insertion and updating of tables. If any such arises, then proper error messages or warning, if any, has to be displayed.

In AIR DEFENCE application the validation testing is done to different fields especially in registration and sign up field. Where the validation done through python program.Main validation used is NULL value in the text boxes.And also done Mobile number , E-mail validation.

OUTPUT TESTING

After performing validation testing, the next step is output testing of the proposed system. Since the system cannot be useful if it does not produce the required output. Asking the user about the format in which the system is required tests the output displayed or generated by the system under consideration. While testing, errors are againuncovered and corrected by using the above steps and correction are also noted for future use. The system has been verified and validated by running test data and live data.

USER ACCEPTANCE TESTING

User acceptance testing of the system is the key factor for the success of any system. As we have created any easy interface it is easy to use. Only tough portion is the information to be perfect and has only one kind of user that the administrator.

The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at a time of the development and making change whenever required. This is done with regard to the input screen design and output screen design.

SYSTEM IMPLEMENTATION

& SYSTEM MAINTAINANCE

10.1.SYSTEM IMPLEMENTATION

The implementation includes all those activities that take place to convert from the old system to new. The old system consists of no personal portal for the airmen to track with officials only official public website from the proposed new system. A proper implementation is essential to provide a reliable system to meet the requirements of the customers. An improper implementation may affect the success of the android application.

There are several methods for handling the implementation and the consequent conversion from the old applications to the new application developed in this project.

The most secure methods for compare the old system and the new system is to run the old and new system in parallel. In this approach, a person may operate the old existing application and the new application. This method offers high reliability and security.

working version of the system can is implemented in two parts, the website and android application. The website is managed by the admin and the airmen.

The implementation plan includes host the website and put it into its operation. The implementation plan consists of the following steps:

- List all files required for implementation.
- Host the website and put it into its operation.
- Host the application in a smart phone and put it into its use.

The implementation plan should anticipate possible problems and must be able to deal with them. The usual problems may be missing documents; mixed data formats between current files and errors in data translation, missing data etc.

USER TRAINING

The implementation of the proposed system includes the training of system operators. Training the system operators includes not only instructions in how to use the equipment, but also in how to diagnose malfunctions and in what steps to take when they occur. So proper training should be provided to the system operators. No training is complete without familiarizing users with simple system maintenance activities. Since the proposed system is developed in a GUI, training will be comparatively easy than systems developed in a non-GUI. There are different types of training.

We can select off-site to give depth knowledge to the system operators.

Success of the system depends on the way in which it is operated and used. Therefore, the quality of training given to the operating person affects the successful implementation of the system. The training must ensure that the person can handle all the possible operations.

Training must also include data entry personnel. They must also be given training for the installation of new hardware, terminals, how to power the system, how to power it down, how to detect the malfunctions, how to solve the problems etc. the operators must also be provided with the knowledge of trouble shooting which involves the determination of the cause of the problem.

The proposed system requires trained personnel for operating the system. This will reduce the data entry errors considerably. It is preferable to provide the person with some kind of operating manuals that will explain all the details of the system.

For the purpose of training we have improved our user interface for a guiding style of use and we are providing and intuitive interface for users. Along with all the simplicity we are providing a help section for users of the application with a detailed description of how each module are working and feature wise specialties and benefits.

10.2.SYSTEM MAINTAINANCE

Maintenance of the software is one of major step in the development of the computer system. Software, which is developed by the engineer, should undergo maintenance process in a regular interval of time as time on new problem arises and it must be corrected accordingly. Maintenance and enhancement are a long-term process.

In this project, the maintenance is carried over by the staff. Since they are the key persons to develop this project they know clearly about the project and coding structured. So, they will change the coding whenever required. Regarding the project maintenance, the changes will occur then and there according to the conditions

Various types of maintenance that can be made are:

- **Corrective maintenance**: reactive modification (or repairs) of a software product performed after delivery to correct discovered problems. Included in this category is emergency maintenance, which is an unscheduled modification performed to temporarily keep a software product operational pending corrective maintenance.
- **Adaptive maintenance**: modification of a software product performed after delivery to keep a software product usable in a changed or changing environment. For example, the operating system might be upgraded and some changes to the software may be necessary.
- **Perfect maintenance**: modification of a software product after delivery to provide enhancements for users, improvement of program documentation, and recoding to improve software performance, maintainability, or other software attributes.

- **Preventive Maintenance**: modification of a software product after delivery to detect and correct latent faults in the software product before them become operational faults.

The staff in the concern takes part in each and every level of the project. So they don't need any training of the software. During the development process they sat and entered each and every entry to test the project. They themselves used this is an opportunity to take training is not needed for the users.

SAMPLE CODE

11.SAMPLE CODE

Main page

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:migrantworker/login.dart';

Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
  runApp(const MaterialApp(debugShowCheckedModeBanner: false,  home:
LogIn()));
}
```

Login page

```
import 'package:flutter/material.dart';
import 'package:migrantworker/admin/homepage.dart';
import 'package:migrantworker/selectuser.dart';
import 'package:migrantworker/services/login_service_fire.dart';

void main() {
  runApp(const LogIn());
}
class LogIn extends StatelessWidget {
  const LogIn({super.key});

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      debugShowCheckedModeBanner: false,
      home: LoginPage(),
    );
  }
}

class LoginPage extends StatefulWidget {
  const LoginPage({super.key});

  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  final _formKey = GlobalKey<FormState>();
  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController = TextEditingController();

  bool loading = false;
  bool _obscureText = true; // Add this line to define the _obscureText variable

  void LoginHandler() async {
    setState(() {
      loading = true;
    });

    final email = emailController.text.trim();
    final password = passwordController.text.trim();

    // Default credentials check (admin login)
    if (email == 'admin@gmail.com' && password == '123456') {
```

```
    setState(() {
      loading = false;
    });
    // Navigate to AdminHomePage
    Navigator.push(context,  MaterialPageRoute(builder: (context) {
      return const AdminScreen();
    }));
  } else {
    // Call the login function (for other users)
    await LoginServiceFire().LoginService(
      email: email,
      password: password,
      context: context,
    );
    setState(() {
      loading = false;
    }); } }
@override
Widget build(BuildContext context) {
  return Scaffold(
    body: Stack(
      children: [
        // Background Gradient
        Container(
          decoration: const BoxDecoration(
            gradient: RadialGradient(
              center: Alignment.topCenter,
              radius: 1.0,
              colors: [
                Colors.green,
                Colors.green,  ],  ), ), ),
        // White Wave at the Bottom
```

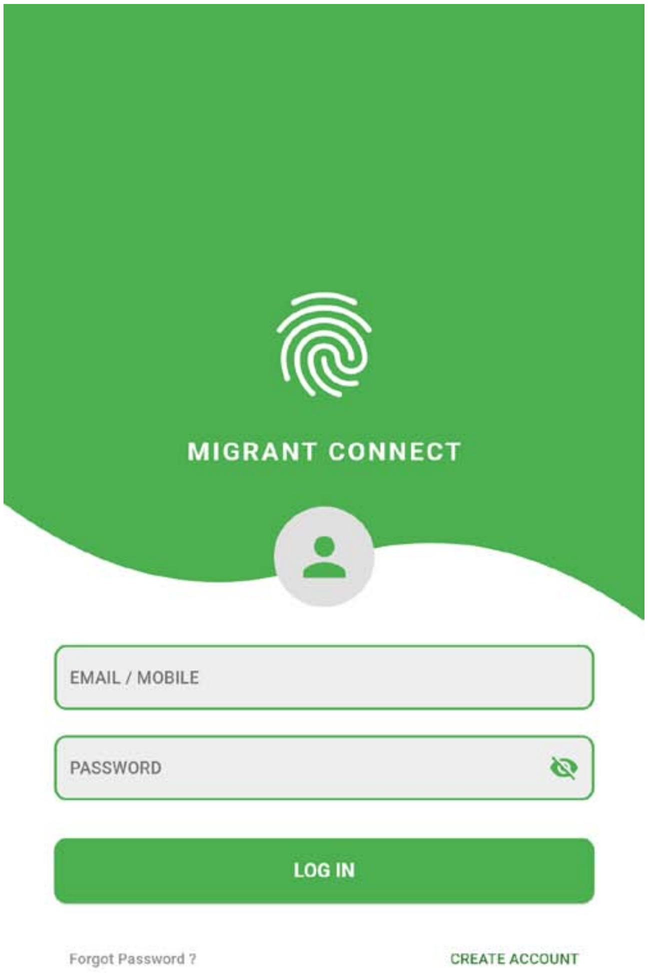
```
Positioned.fill(
  child: Align(
    alignment: Alignment.bottomCenter,
    child: ClipPath(
      clipper: WaveClipper(),
      child: Container(
        color: Colors.white,
        height: MediaQuery.of(context).size.height * 0.85, ), ), ), ),
// Login Content
Center(
  child: Form(
    key: _formKey,
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        const Icon(
          Icons.fingerprint,
          size: 100,
          color: Colors.white,
        ),
        const SizedBox(height: 20),
        const Text(
          'MIGRANT CONNECT',
          style: TextStyle(
            fontSize: 24,
            fontWeight: FontWeight.bold,
            color: Colors.white,
            letterSpacing: 2,
          ), ),
        const SizedBox(height: 30),
        CircleAvatar(
          radius: 40,
  backgroundColor: Colors.grey.shade300,
  child: const Icon(
    Icons.person,
    size: 50,
    color: Colors.green,
  ), ),
const SizedBox(height: 30),
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 40.0),
  child: Column(
    children: [
      // Email/Mobile Field with Green and Red Borders
      TextFormField(
        controller: emailController,
        decoration: InputDecoration(
          hintText: 'EMAIL / MOBILE',
          hintStyle: TextStyle(color: Colors.grey.shade600),
          filled: true,
          fillColor: Colors.grey.shade200,
          enabledBorder: OutlineInputBorder(
            borderRadius: BorderRadius.circular(10),
            borderSide: const BorderSide(
                color: Colors.green,  width: 2),
          ),
          focusedBorder: OutlineInputBorder(
            borderRadius: BorderRadius.circular(10),
            borderSide: const BorderSide(
                color: Colors.green,  width: 2),
          ),
          errorBorder: OutlineInputBorder(
            borderRadius: BorderRadius.circular(10),
            borderSide:
```

```
          const BorderSide(color: Colors.red,  width: 2),
    ),
    focusedErrorBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(10),
      borderSide:
          const BorderSide(color: Colors.red,  width: 2),
    ), ),
  validator: (value) {
    if (value == null || value.isEmpty) {
      return 'Please enter your email or mobile';
    } else if (!RegExp(r'^[^@]+@[^@]+\.[^@]+$')
            .hasMatch(value) &&
        !RegExp(r'^[0-9]{10}$').hasMatch(value)) {
      return 'Enter a valid email or mobile number';
    }
    return null;
  }, ),
const SizedBox(height: 20),
// Password Field with Green and Red Borders
// Password Field with Green and Red Borders and Eye Icon
TextFormField(
  controller: passwordController,
  obscureText: _obscureText,  // Toggle visibility
  decoration: InputDecoration(
    hintText: 'PASSWORD',
    hintStyle: TextStyle(color: Colors.grey.shade600),
    filled: true,
    fillColor: Colors.grey.shade200,
    enabledBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(10),
      borderSide: const BorderSide(
          color: Colors.green,  width: 2),
  ),
  focusedBorder: OutlineInputBorder(
    borderRadius: BorderRadius.circular(10),
    borderSide: const BorderSide(
        color: Colors.green,  width: 2),
  ),
  errorBorder: OutlineInputBorder(
    borderRadius: BorderRadius.circular(10),
    borderSide:
        const BorderSide(color: Colors.red,  width: 2),
  ),
  focusedErrorBorder: OutlineInputBorder(
    borderRadius: BorderRadius.circular(10),
    borderSide:
        const BorderSide(color: Colors.red,  width: 2),
  ),
  suffixIcon: IconButton(
    icon: Icon(
      _obscureText
          ? Icons.visibility_off
          : Icons.visibility,
      color: Colors.green,
    ),
    onPressed: () {
      setState(() {
        _obscureText =
            !_obscureText; // Toggle visibility
      });}, ), ),
validator: (value) {
  if (value == null || value.isEmpty) {
    return 'Please enter your password';
  } else if (value.length < 6) {
      return 'Password must be at least 6 characters long';
    }
    return null;
  },
),
const SizedBox(height: 30),
// Login Button
SizedBox(
  width: double.infinity,
  child: loading
      ? const Center(
          child: CircularProgressIndicator(),
        )
      : ElevatedButton(
          onPressed: LoginHandler,
          style: ElevatedButton.styleFrom(
            backgroundColor: Colors.green,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(10),
            ),
            padding: const EdgeInsets.symmetric(
                vertical: 15),
          ),
          child: const Text(
            'LOG IN',
            style: TextStyle(
              color: Colors.white,
              fontSize: 18,
              fontWeight: FontWeight.bold,
            ), ), ),  ),

const SizedBox(height: 20),
                        Row(
                          mainAxisAlignment: MainAxisAlignment.spaceBetween,
                          children: [
                            TextButton(
                              onPressed: () {},
                              child: const Text(
                                'Forgot Password ?',
                                style: TextStyle(
                                    color: Color.fromARGB(255, 146, 148, 146)),
                              ),
                            ),
                            TextButton(
                              onPressed: () {
                                Navigator.push(
                                    context,
                                    MaterialPageRoute(
                                      builder: (context) => const SelectUser(),
                                    ));
                              },
                              child: const Text(
                                'CREATE ACCOUNT',
                                style: TextStyle(
                                    color: Color.fromARGB(255, 55, 129, 58)),
                              ), ), ], ), ], ), ), ], ), ), ), ], ),  );}}
// Custom Wave Clipper
class WaveClipper extends CustomClipper<Path> {
  @override
  Path getClip(Size size) {
    final path = Path();
    path.lineTo(0, size.height * 0.3);
    path.quadraticBezierTo(
      size.width / 4, size.height * 0.4, // Control point
      size.width / 2,  size.height * 0.35,  // End point
    );
    path.quadraticBezierTo(
      3 * size.width / 4,  size.height * 0.3,  // Control point
      size.width,  size.height * 0.4,  // End point
    );
    path.lineTo(size.width,  0);
    path.lineTo(size.width,  size.height);
    path.lineTo(0,  size.height);
    path.close();
    return path;
  }

  @override
  bool shouldReclip(CustomClipper<Path> oldClipper) => false;
}
```

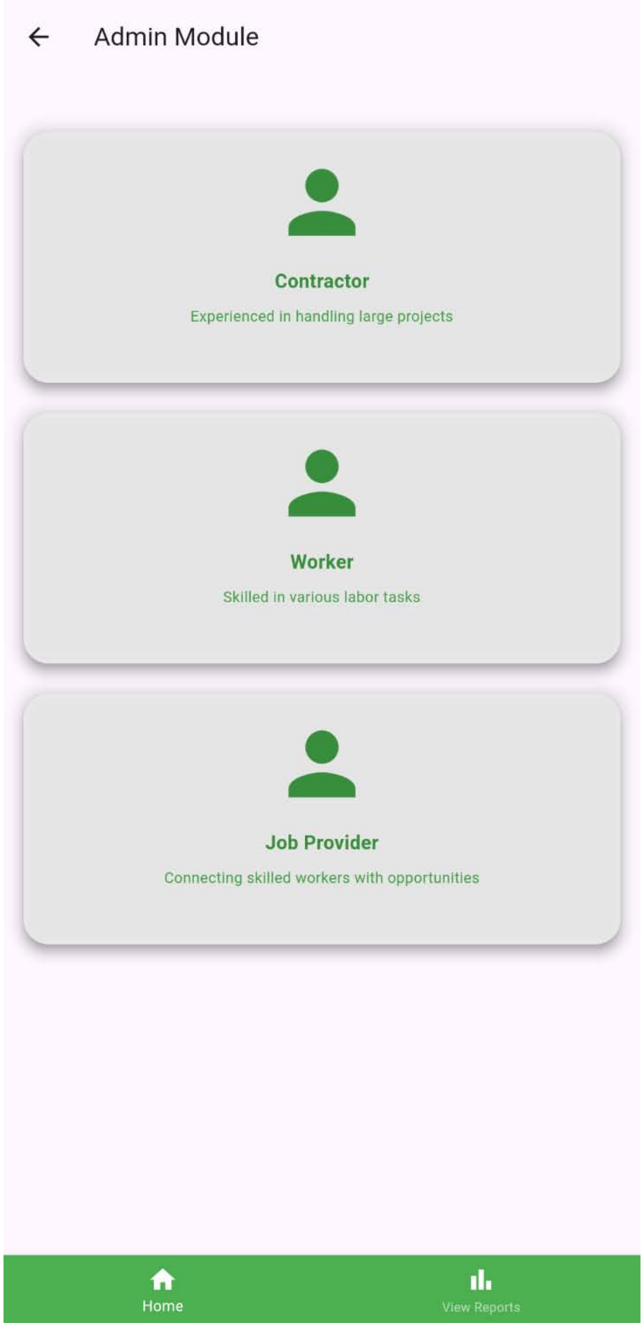
SCREENSHOT

12.SCREENSHOT

LOGIN PAGE



ADMIN HOME PAGE



Admin Module

Contractor

Experienced in handling large projects

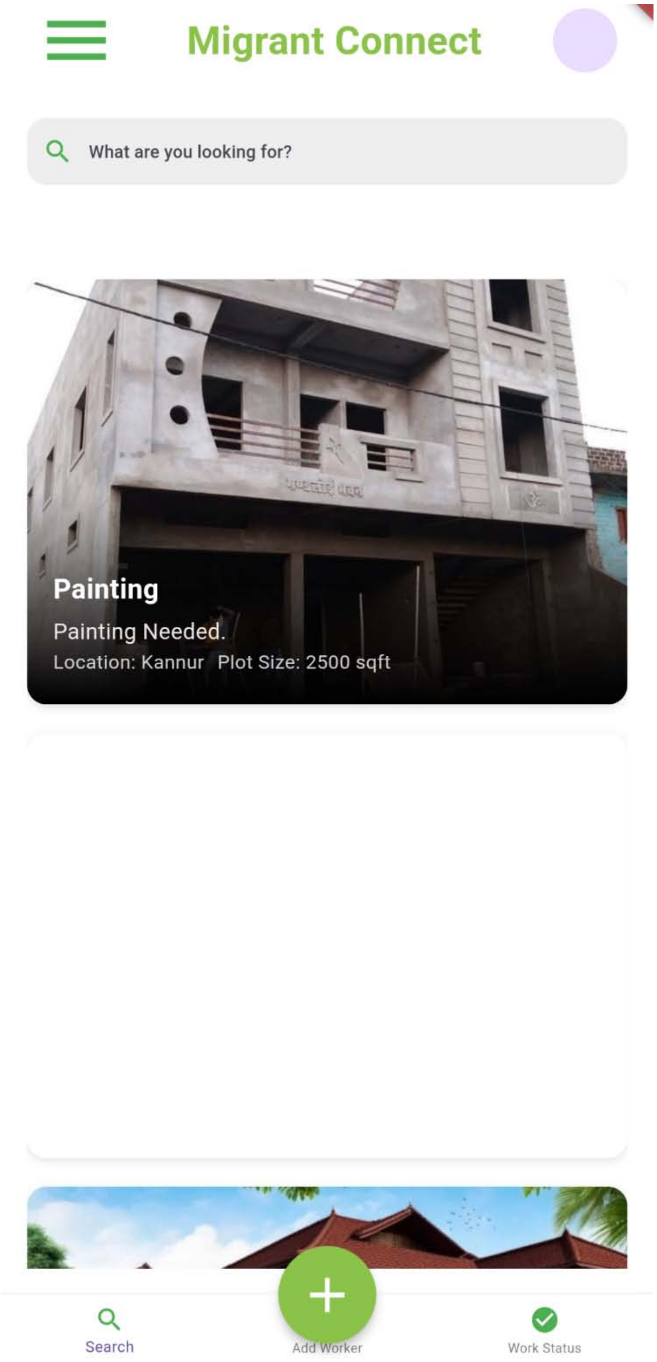
Worker

Skilled in various labor tasks

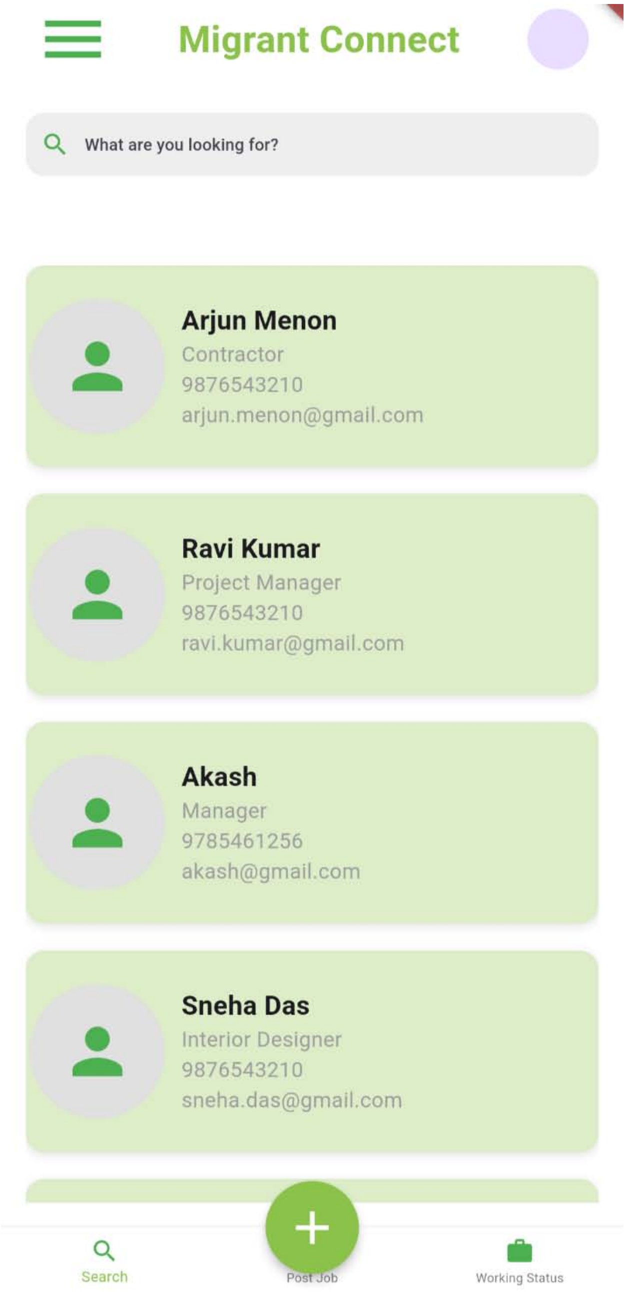
Job Provider

Connecting skilled workers with opportunities

CONTRACTOR HOME PAGE



JOB PROVIDER HOME PAGE



Migrant Connect

What are you looking for?

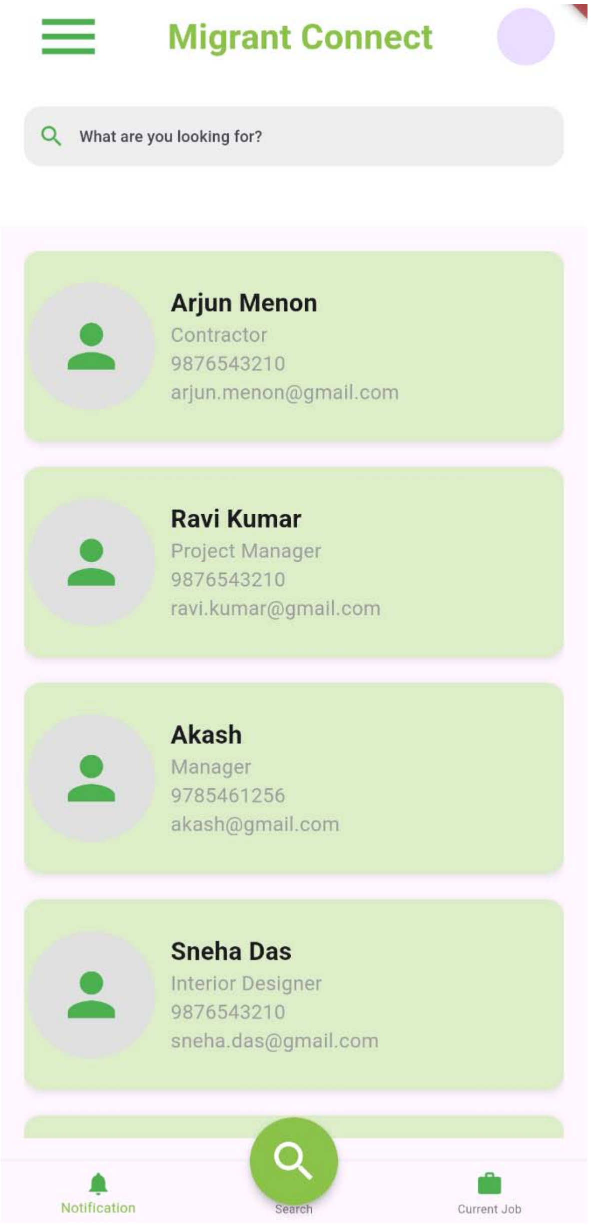
Arjun Menon
Contractor
9876543210
arjun.menon@gmail.com

Ravi Kumar
Project Manager
9876543210
ravi.kumar@gmail.com

Akash
Manager
9785461256
akash@gmail.com

Sneha Das
Interior Designer
9876543210
sneha.das@gmail.com

WORKER HOME PAGE



Migrant Connect

Arjun Menon
Contractor
9876543210
arjun.menon@gmail.com

Ravi Kumar
Project Manager
9876543210
ravi.kumar@gmail.com

Akash
Manager
9785461256
akash@gmail.com

Sneha Das
Interior Designer
9876543210
sneha.das@gmail.com

SCOPE FOR FUTURE ENHANCEMENT

13.SCOPE FOR FUTURE ENHANCEMENT

Scope of the Project

The proposed platform is designed to revolutionize the recruitment, registration, and management of migrant workers by providing an efficient, transparent, and well-structured system for connecting contractors, workers, and job providers. The scope of this project extends across various domains, ensuring a comprehensive and scalable solution that meets the needs of all stakeholders.

1. **Seamless Worker-Contractor-Job Provider Integration**

The platform facilitates smooth interaction between workers, contractors, and job providers by offering a unified digital ecosystem where users can register, manage profiles, and engage in work-related activities efficiently.

2. **AI-Powered Job Recommendations**

The system utilizes AI-driven algorithms to match workers with suitable jobs based on their skills, experience, and location, optimizing workforce allocation for job providers and contractors.

3. **Automated Job and Workforce Management**

Contractors can post job opportunities, assign workers, track progress, and manage their workforce seamlessly, while job providers can request specific worker categories based on project needs.

4. **Secure Identity Verification and Compliance**

The platform ensures identity verification through document uploads, reducing fraudulent activities and ensuring regulatory compliance for both workers and contractors

.

5. **Real-Time Communication and Notifications**

A built-in messaging system allows workers, contractors, and job providers to communicate effectively, while notifications keep users updated about job postings, assignments, and important alerts.

6. **Worker History and Ratings System**

Workers' performance and work history are documented, allowing job providers and contractors to make informed hiring decisions while fostering accountability and transparency.

7. **Geolocation and Travel Assistance**

Location-based services assist workers in reaching job sites with optimized routes, reducing delays and improving overall efficiency in workforce deployment.

8. **Admin Oversight and System Integrity**

An administrative module is incorporated to monitor and regulate platform activities, review reports on fraudulent behavior, and maintain system integrity through necessary interventions.

9. **Scalability and Future Enhancements**

The system is designed to be scalable, allowing the integration of advanced features like skill training modules, multilingual support, blockchain-secured transactions, and predictive analytics for job recommendations.

By defining this extensive scope, the project ensures a robust and future-ready workforce management solution that enhances efficiency, fosters transparency, and streamlines job allocation for all stakeholders involve

CONCLUSION

14.CONCLUSION

In conclusion, our platform stands as a transformative solution in the realm of workforce management, redefining how migrant workers, contractors, and job providers connect and collaborate. By integrating advanced features such as AI-driven job recommendations, seamless communication tools, and real-time workforce tracking, the system fosters an ecosystem built on efficiency, transparency, and trust.

With a user-centric approach, the platform not only streamlines employment processes but also ensures flexibility and accessibility for all stakeholders. The inclusion of an admin module reinforces system integrity by mitigating fraudulent activities and maintaining a secure environment for users.

As the nature of work continues to evolve, this platform remains at the forefront, bridging gaps in the labor market while enhancing employment opportunities. By prioritizing connectivity, organization, and empowerment, it lays the foundation for a structured, fair, and dynamic workforce ecosystem, shaping the future of local job markets.

REFERENCES

15.REFERENCES

REFERENCE BOOK

1. Ian Somerville -"Software Engineering" 7thEdition, 2004
2. S Roger Presman, ―Software Engineering‖MeGraw-hillInternationalEdition, 1994
3. Alen dennis - "System Analysis and Design" 10th Edition, 2015
4. Rohith Khurana ―Software Engineering principles and practices‖2nd Edition, 2008

WEB REFERENCES

1. http://www.stackoverflow.com, StackOverflow
2. http://www.codeproject.com, CodeProject
3. http://w3schools.com, *W3Schools*