

MIGRANT WORKERS

A PROJECT REPORT

Submitted by

AKSHAY S

in partial fulfilment for the award of the degree of

MASTER OF COMPUTER APPLICATIONS



HAJI C.H.M.M. COLLEGE FOR ADVANCED STUDIES

CHAVARCODE, PALAYAMKUNNU P O – 695146

THIRUVANANTHAPURAM DIST

KERALA

UNIVERSITY OF KERALA, THIRUVANANTHAPURAM

AUGUST 2023

**HAJI C.H.M.M. COLLEGE FOR ADVANCED STUDIES
CHAVARCODE, PALAYAMKUNNU P O – 695146
THIRUVANANTHAPURAM DIST KERALA**

MASTER OF COMPUTER APPLICATIONS



BONAFIDE CERTIFICATE

Certified that this project report “**MIGRANT WORKERS** “ is the bona fide work of **AKSHAY S** who carried out the project work under my supervision.

Reg.No :95521801003

Mr.HARIKRISHNAN.S R

Associate Professor

HEAD OF THE DEPARTMENT

EXTERNAL EXAMINER

Mr.HARIKRISHNAN.S R

Associate Professor

INTERNAL GUIDE

ACKNOWLEDGEMENT

I would like to express my gratitude to God for giving me good health and better courage to accomplish this project successfully.

I express my sincere gratitude to the Director of **MCA Prof. (Dr). SIRAJUDEEN. M**, for providing me an opportunity for doing this project work.

Special thanks to **Mr. HARIKRISHNAN S R**, Associate Professor, Head of Department for his expert and valuable advice, inspiration and facilities rendered throughout for successful completion of the project.

I take this opportunity to express my sincere gratitude and indebtedness to my Internal guide **Mr. HARIKRISHNAN S R**, Associate Professor, Department of MCA for providing all possible fruitful discussions to make this project be a success.

With great pleasure I may record my deep gratitude to my parents, friends and to all staff members of **MCA Department** for the immeasurable help rendered to me during the course of the project.

AKSHAY S

TABLE OF CONTENTS

ACKNOWLEDGEMENT	iv
LIST OF TABLES.	vii
LIST OF FIGURES	viii
ABSTRACT.	ix

CHAPTER

1. INTRODUCTION	1
1.1. Objective of the Problem	1
1.2. Statement of the Problem.....	2
2. SYSTEM ANALYSIS	3
2.1. Existing System.....	4
2.2. Disadvantages of Existing System	5
2.3. Proposed System	6
2.4. Advantages of Proposed System.....	6
2.5. Feasibility Study.....	8
3. SYSTEM SPECIFICATION	11
3.1. Software Requirements.....	11
3.2. Hardware Requirements	11
4. SYSTEM DESIGN	12
4.1. Context Level Diagram.....	15
4.2. Data Flow Diagram.....	16
4.2.1 DFD Level 1	16
4.3. ER Diagram.....	22
4.4. Database Design	28
4.5. Normalization.....	37
4.6. Design each Subsystem	39
4.7. UML Diagram	40
4.7.1 Use Case Diagram	41
4.7.2 Class Diagram... ..	47
4.7.3 Sequence Diagram... ..	48

CHAPTER

5. CODING	49
5.1. Language Study	49
5.2. Functional Description	60
5.3. System Code	63
6. TESTING	68
6.1. Levels of Testing.	68
7. IMPLEMENTATION	72
7.1. Implementation of Proposed System.	72
8. SECURITY AND BACKUP RECOVERY MECHANISM	74
9. CONCLUSION	76
10. FUTURE ENHANCEMENT	77
APPENDIX	78
Input and Output Forms	78
BIBLIOGRAPHY	83

LIST OF TABLES

Tables

4.4.1 Agency Table.....	28
4.4.2 Worker Table	29
4.4.3 police station Table	30
4.4.4 Insurance agency Table	31
4.4.5 Scheme Table.....	31
4.4.6 Notification Table	32
4.4.7 Complaint Table.....	32
4.4.8 Report Table	33
4.4.9 Salary Table	33
4.4.10 Woker job Table.....	33
4.4.11 pcc apply Table	34
4.4.12 Woker scheme Table	34
4.4.13 Woker claim insurance Table	34
4.4.14 payment Table.....	35
4.4.15 admin Table	35
4.4.16 labourcommission Table.....	35
4.4.17 chat Table.....	36

LIST OF FIGURES

Figure

4.1 Context Level Diagram.....	15
4.2 Level 1 DFD	16
4.3 ER Diagram	22
4.6 Design of Each Sub System.....	39
4.7 Use Case Diagram	41

ABSTRACT

Kerala is witnessing large inflow of migrant labour from different parts of country in recent years. Though labourers from state as far as West Bengal, Bihar, Uttar Pradesh and Orissa now flock to Kerala. Higher wages for unskilled labour in the state, large opportunities for employment and shortage of local labour, paradoxically despite the high unemployment rate in the state, led to the massive influx of migrant labours to the state.

The system provides a facility for migrant labours to register with the state authorities would be engaged for work in the construction sector. Construction sector employs large number of migrant labours in the state. Seeking to address their issues properly and know about their whereabouts, the government is planning to engage registered labours only in the sector. The government has also introduce a safety plan to ensure the rights of labours are not violated in view of complaints about poor working conditions and mishaps on construction sites. As per the plan, builder should submit a safety plan along with the building plan to get sanction for construction. Manual systems often rely on paper-based records and manual entry, resulting in delays in data updates. This lack of real-time data can lead to inaccurate or outdated information, making it difficult to track workers' movements and ensure their safety promptly. The existing system mat struggle to handle a growing workforce efficiently. A manual system might lack transparency, making it challenging to ensure accountability and detect any potential malpractices or discrepancies in the management of migrant workers. Manual systems often rely on paper-based records and manual entry, resulting in delays in data updates. This lack of real-time data can lead to inaccurate or outdated information, making it difficult to track workers' movements and ensure their safety promptly.

CHAPTER – 1

INTRODUCTION

1.1 OBJECTIVE OF THE PROBLEM

There are many factors those are attracting migrants to Kerala. The factors are like poverty, unemployment, density of population, bad yield from agriculture, low demand of workers etc. Kerala provide better employment opportunities, better life standard, high wages compared to other states, lesser communal clashes, high health indices, provision of education for children are attract migrants to Kerala.

The proposed system has much more social relevant in the current situation. Because the rising rate of crimes those are associated with migrant workers. During the last 5years approximately 1770 cases registered in the state in which migrants were accused. Drug trafficking , fake currency, robbery are major cases involving migrants , while there were brutal murder cases also in which migrants were involved.

The proposed system provides a greater environment for managing the workers in an efficient way as well as ensures their safety plans in an accurate manner by associating different departments through this system. This system will help the various government departments to co-ordinate and control the migrant workers in an efficient and effective manner. The system provides a facility to store the entire details of workers in the Police department. It will help to the Police department whether they will involve in any crimes.

1.2 STATEMENT OF THE PROBLEM

The existing system provides to store only the details of migrant workers in the Police station. The existing system is partially computerized. There is no centralized department to track and control the migrant workers efficiently. Mainly the present system most of the services are manual. As the number of migrant workers increases, managing them manually becomes more cumbersome and challenging. Manual systems often rely on paper-based records and manual entry, resulting in delays in data updates. This lack of real-time data can lead to inaccurate or outdated information, making it difficult to track workers' movements and ensure their safety promptly. The existing system may struggle to handle a growing workforce efficiently.

The objective of the online application “MW” provides a greater environment for managing the workers in an efficient way as well as ensures their safety plans in an accurate manner by associating different departments through this system.

The system contains a centralized department to track and control the migrant workers efficiently. There are many factors those are attracting migrants to Kerala. The factors are like poverty, unemployment, density of population, bad yield from agriculture, low demand of workers etc. Kerala provide better employment opportunities, better life standard , high wages compared to other states, lesser communal clashes, high health indices, provision of education for children are attract migrants to Kerala. The system automates various processes, reducing the need for manual paperwork and streamlining operations. This efficiency saves time and resources for all stakeholders involved. By tracking the workers' movements and activities, the system can help ensure their safety and protect them from potential risks or exploitation. This contributes to a safer environment for the workers.

CHAPTER - 2

SYSTEM ANALYSIS

System Analysis is a detailed study of various operation performed by a system and their relationship within and outside of the system. Here the key question is: What must be done to solve the problem? One aspect of analysis is defining the boundaries of a system and determining whether or not a candidate system should consider other related system. Analysis begins when a user or manager begins a study of the program using an existing system.

During analysis, data is collected on the various files, decision points and transactions handled by the present system. The commonly used tools in system are dataflow diagrams, interviews, onsite observations etc. System analysis is application of the system approach to the problem solving using computers. The ingredients are the system elements, process and technology. This means that to do system works, one is to understand the system concepts and how the organizations operate as a system and the design appropriate computer based system that will meet the organizations requirements. It is actually customized approach to the use of computer problem solving.

Analysis can be defined as the separation of a substance into parts for study an interpretation, detailed examination. System development revolves around a lifecycle that being with the recognition of user needs. The critical phase of managing system project is planning. To launch a system investigation, we need a master plan detailing the steps taken, the people to be questioned and outcome expected.

System analysis can be categorized into four parts:

- System planning and initial investigation.
- Information gathering.
- Applying Analyzing tools for structured analysis.
- Feasibility study.
- Cost/ Benefits analysis.

System study or system analysis is the first among the four life cycle phases of a system. System analysis begins when a user or manager request a studying of a program in either an existing system or a project one. It involves studying the base of the organizations currently operating, retrieving and processing data to produce information with goal of determining how to make it work better. System analysis itself breaks down into stages preliminary and detailed. During preliminary analysis, the analyst and the user list the objectives of the system. To arrive at a preliminary report, the analyst interviews key personnel in the organization and scheduling meetings with the users and the management. If management approves preliminary report, the system analysis or study phases advantage to the second stage, the detailed analysis. If the management approves the result of a preliminary analysis, the analyst conducts a detailed analysis, gathering facts about the old system, outline objectives for a new one, estimating costs, listing possible alternatives and making recommendation. After gathering the analysis will arrange it in organized way called the feasibility study.

Thus the objective of analysis phase of the system analysis and design exercise is the establishment of the requirement for the system to be acquired, developed and installed. Fact finding or gathering is essential of requirements. In brief analysis of the system helps an analyst to make a clear view of an existing system and there by can give suggestions for the improvement of the new system information about the organization's policies, goals, objectives and structure explains the kind of environment that promotes the introduction of computer based system. It is necessary that the analyst must be familiar with the objectives, activities and the functions of the organization in which the computer system is to be implemented.

2.1 EXISTING SYSTEM

The existing system provides to store only the details of migrant workers in the Police station. The existing system is partially computerized. There is no centralized department to track and control the migrant workers efficiently.

2.2 Disadvantages of Existing System

- Mainly the present system most of the services are manual.
- Manual systems often rely on paper-based records and manual entry, resulting in delays in data updates. This lack of real-time data can lead to inaccurate or outdated information, making it difficult to track workers' movements and ensure their safety promptly.
- The manual system may not provide the necessary mechanisms to track workers' safety in real-time during their journeys or at their workplaces, potentially putting workers at risk.
- As the number of migrant workers increases, managing them manually becomes more cumbersome and challenging. The existing system may struggle to handle a growing workforce efficiently.
- A manual system might lack transparency, making it challenging to ensure accountability and detect any potential malpractices or discrepancies in the management of migrant workers.
- No Work Permit Card.
- There is police clearance checking to check the worker has any criminal background.
- No centralized system.

2.3 PROPOSED SYSTEM

The proposed system provides a greater environment for managing the workers in an efficient way as well as ensures their safety plans in an accurate manner by associating different departments through this system

The proposed system contains a centralized department to track and control the migrant workers efficiently. There are many factors those are attracting migrants to Kerala. The factors are like poverty, unemployment, density of population, bad yield from agriculture, low demand of workers etc. Kerala provide better employment opportunities, better life standard , high wages compared to other states, lesser communal clashes, high health indices, provision of education for children are attract migrants to Kerala.

The proposed system has much more social relevant in the current situation. Because the rising rate of crimes those are associated with migrant workers. During the last 5years approximately 1770 cases registered in the state in which migrants were accused. Drug trafficking , fake currency, robbery are major cases involving migrants , while there were brutal murder cases also in which migrants were involved.

This system will help the various government departments to co-ordinate and control the migrant workers in an efficient and effective manner. The system provides a facility to store the entire details of workers in the Police department. It will help to the Police department whether they will involve in any crimes.

2.4 Advantages of Proposed System

- The system can enforce compliance with labor laws and regulations, protecting the rights of migrant workers and ensuring they are treated fairly and legally.
- The system automates various processes, reducing the need for manual paperwork and streamlining operations. This efficiency saves time and resources for all stakeholders involved.
- With the system in place, authorities, agencies, and employers can monitor the

movement and status of migrant workers in real-time, ensuring better oversight and quick responses to any emergencies or issues.

- The system centralizes all relevant data, making it easier to access and manage information related to workers, agencies, contracts, and more. This centralized database minimizes data duplication and inconsistencies.
- By tracking the workers' movements and activities, the system can help ensure their safety and protect them from potential risks or exploitation. This contributes to a safer environment for the workers.
- The system can enforce compliance with labor laws and regulations, protecting the rights of migrant workers and ensuring they are treated fairly and legally.
- Work permit card generation.
- Police clearance checking.

2.5 FEASIBILITY STUDY

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus, when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

A feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that is spent on it. Feasibility study lets the developer foresee the future of the project and its usefulness.

All the projects are feasible given unlimited resources and infinite time. Unfortunately, the development of the computer-based system is more likely to be played by a security of resources and difficulty delivery dates. Feasibility and risk analysis are related in many ways. If project risk is great, the feasibility of producing the quality software is reduced.

Steps in Feasibility Study

Feasibility Study involves eight steps:

- Form a project team and appoint a project leader.
- Prepare a system flow chart.
- Enumerate potential candidate systems.
- Describe and identify characteristics of candidate systems.
- Describe and evaluate performance and cost effectiveness of each candidate systems.

- Weight system performance and cost data.
- Select the best candidate system.
- Prepare and report final project directive and management.

Mainly three key considerations are involved in the feasibility analysis.

- Economic Feasibility
- Technical Feasibility
- Operational Feasibility

2.5.1 Economical Feasibility

Economical Feasibility is the most frequently used method for evaluating the effectiveness of the candidate system. More commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. Otherwise, further justifications or alterations in the proposed system will have to be made if it is having a chance of being approved. This is an ongoing effort that improves in accuracy at each phase of the system life cycle.

2.5.2 Technical Feasibility

Technical Feasibility centers on the existing computer system (hardware, software, etc) and to what extent it can support the proposed addition. For example, if the current computer is operating at 80 percent capacity, an arbitrary ceiling, then running another application could over load the system or require additional hardware. This involves financial considerations to accommodate technical enhancements. If the budget is a serious constraint, then the project is judged not feasible.

2.5.3 Operational Feasibility

The main problem faced during development of a new system is getting acceptance from the user. People are inherently resistant to changes and computers have been known to facilitate change. It is mainly related to human organizational and political aspects.

The points to be considered are:

- What changes will be brought with the system?
- What new skills will be required? Do the existing staff members have these skills? If not, can they be trained due course of time?

Generally, project will not be rejected simply because of operational feasibility but such considerations are likely to critically affect the nature and scope of the eventual recommendations. This feasibility study is carried out by a small group of people who are familiar with information system techniques, who understand the parts of the business that are relevant to the project and are skilled in skilled analysis and design process.

CHAPTER – 3

SYSTEM SPECIFICATION

3.1 Software Requirements

- | | | |
|---------------------|---|-------------------------------|
| 1. Operating System | : | Windows 7 or higher |
| 2. Framework | : | Django |
| 3. Environment | : | Visual Studio 2012 |
| 4. Front end | : | HTML, CSS, JavaScript |
| 5. Language | : | python |
| 6. Back end | : | SQLite |
| 7. Documentation | : | Microsoft Word 2007 or higher |

3.2 Hardware Requirements

- | | | |
|----------------------|---|------------------------------|
| 1. Processor | : | Dual Core 1.60 GHz or higher |
| 2. Hard disk | : | 500 GB |
| 3. RAM | : | 4GB |
| 4. Other Peripherals | : | Keyboard, Mouse, Monitor |

CHAPTER - 4

SYSTEM DESIGN

System Design involves translating system requirements and conceptual design into technical specifications and general flow of processing. After the system requirements have been identified, information has been gathered to verify the problem and after evaluating the existing system, a new system is proposed.

System Design is the process of planning of new system or to replace or complement an existing system .It must be thoroughly understood about the old system and determine how computers can be used to make its operations more effective.

System design sits at technical the kernel of system development. Once system requirements have been analyzed and specified system design is the first of the technical activities-design, code generation and test- that required build and verifying the software. System design is the most creative and challenging phases of the system life cycle. The term design describes the final system and the process by which it is to be developed.

System design is the high level strategy for solving the problem and building a solution. System design includes decisions about the organization of the system into subsystems, the allocation of subsystems to hardware and software components and major conceptual and policy decision that forms the framework for detailed design.

There are two levels of system design:

- Logical design.
- Physical design.

In the logical design, the designer produces a specification of the major features of the system which meets the objectives. The delivered product of logical design includes current requirements of the following system components:

- Input design.
- Output design.

- Database design.

Physical design takes this logical design blue print and produces the program software, files and a working system. Design specifications instruct programmers about what the system should do. The programmers in turn write the programs that accept input from users, process data, produce reports, and store data in files.

Structured design is a data flow based methodology that partitions a program into a hierarchy of modules organized top-down manner with details at the bottom. Data flow diagrams are the central tool and the basis from which other components are developed. The transformation of data from input to output, through processes may be described logically and independently of the physical components.

DATA FLOW DIAGRAM

A data flow diagram is a graphical technique that depicts information flow and transforms that are applied as data move from input to output. The DFD is also known as Data Flow Graph or Bubble Chart. The DFD is used to represent increasing information flow and functional details. Also DFD can be stated as the starting point of the design phase that functionally decomposes the requirements specifications down to the lowest level of detail. A Level 0 also called a fundamental system model or a context level DFD that represent the entire software elements as a single bubble with input and output data indicated by incoming and outgoing arrows, respectively. Additional process and information flow parts are represented in the next level, i.e., level 1 DFD. Each of the processes represented at level 1 are sub functions of overall system depicted in the context model. Any processes that are complex in level 1 will be further represented into sub functions in the next level, i.e., level 2.

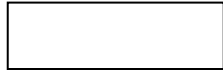
Data flow diagram is a means of representing a system at any level of detail with a graphic network of symbols showing data flows, data stores, data processes and data sources. The purpose of data flow diagram is to provide a semantic bridge between users and system developers. The diagram is the basis of structured system analysis. A DFD

describes what data flows rather than how they are processed, so it does not depend on hardware, software, data structure or file organization.

Components Of Data Flow Diagram

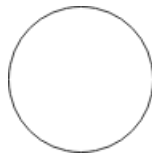
There are four symbols that are used in the drawing of Data Flow Diagrams:

- **Entities**



External entities represent the sources of data that enter the system or the recipients of data that leave the system.

- **Process**



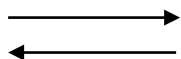
Processes represent activities in which data is manipulated by being stored or retrieved or transformed in some way. A circle represents it. The process will show the data transformation or change.

- **Databases**



Databases represent storage of data within the system.

- **Data Flow**



A data flow shows the flow of information from its source to its destination.

4.1 Context Level Diagram

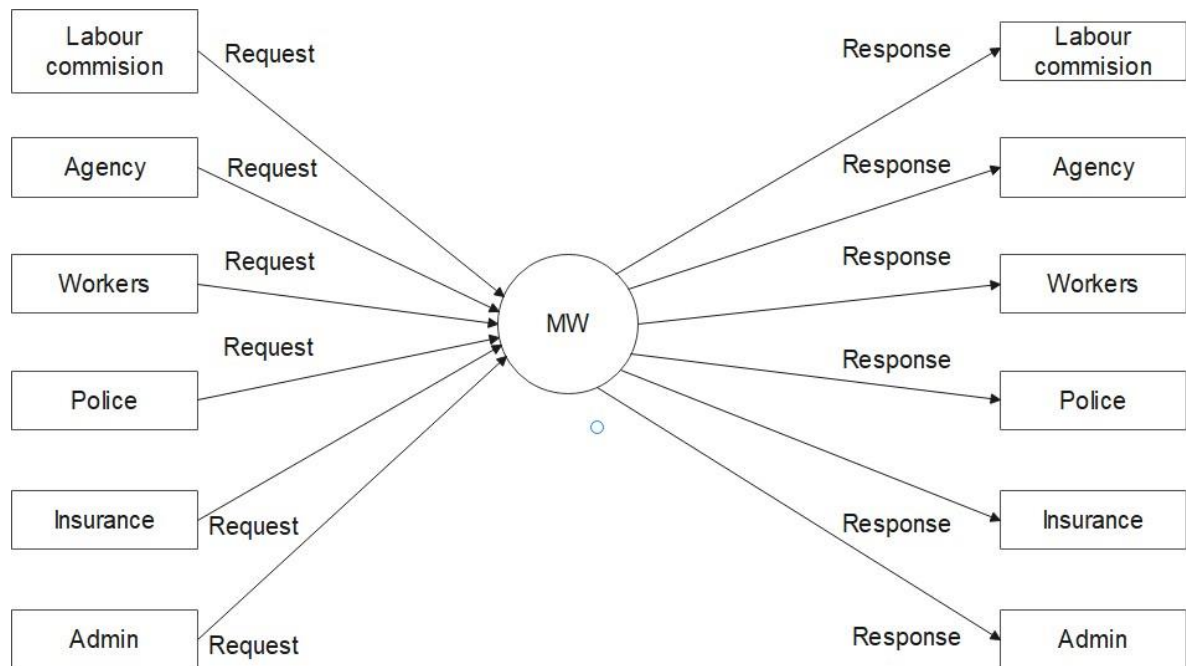


Figure 4.1 Context Level Diagram

4.2 Data Flow Diagram

4.2.1 Level 1 Admin

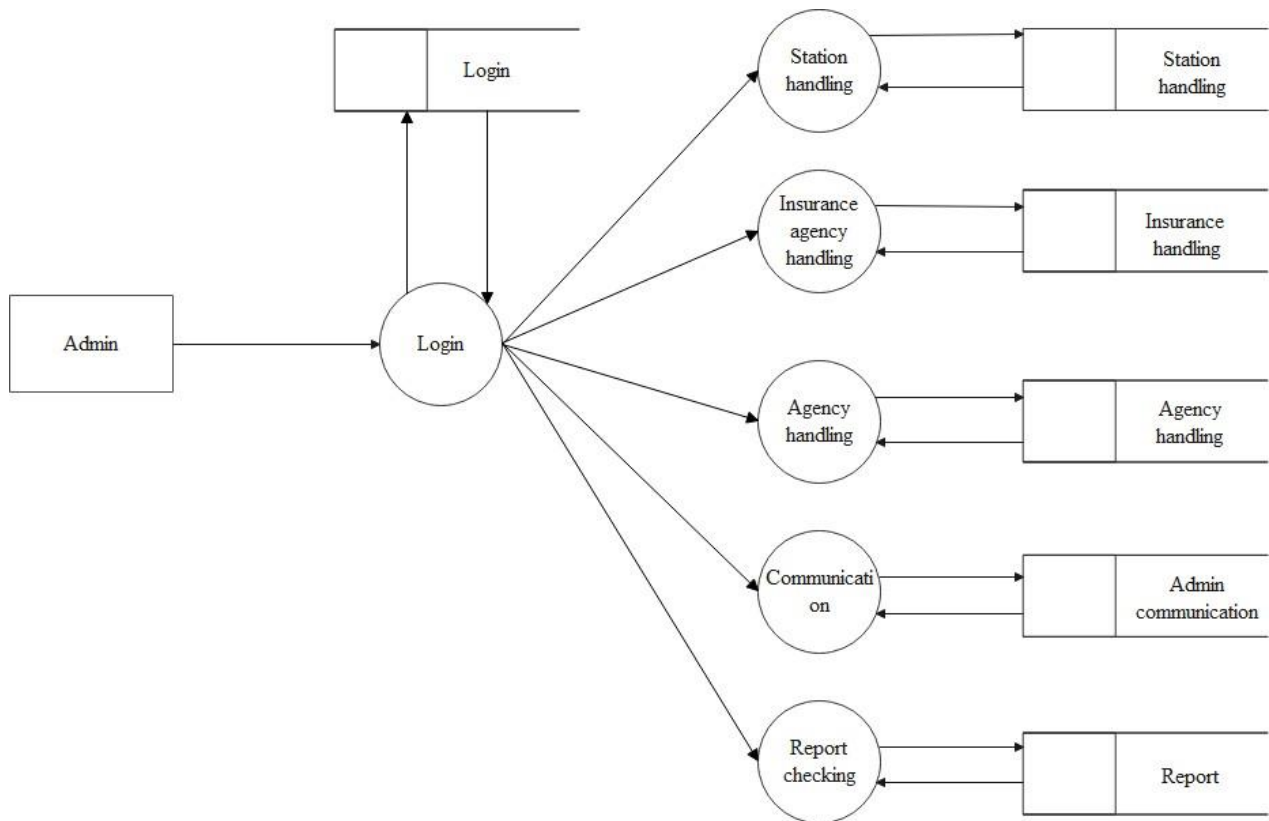


Figure 4.2.1 Admin (Level 1)

4.2.2 Level 1 Insurance agency

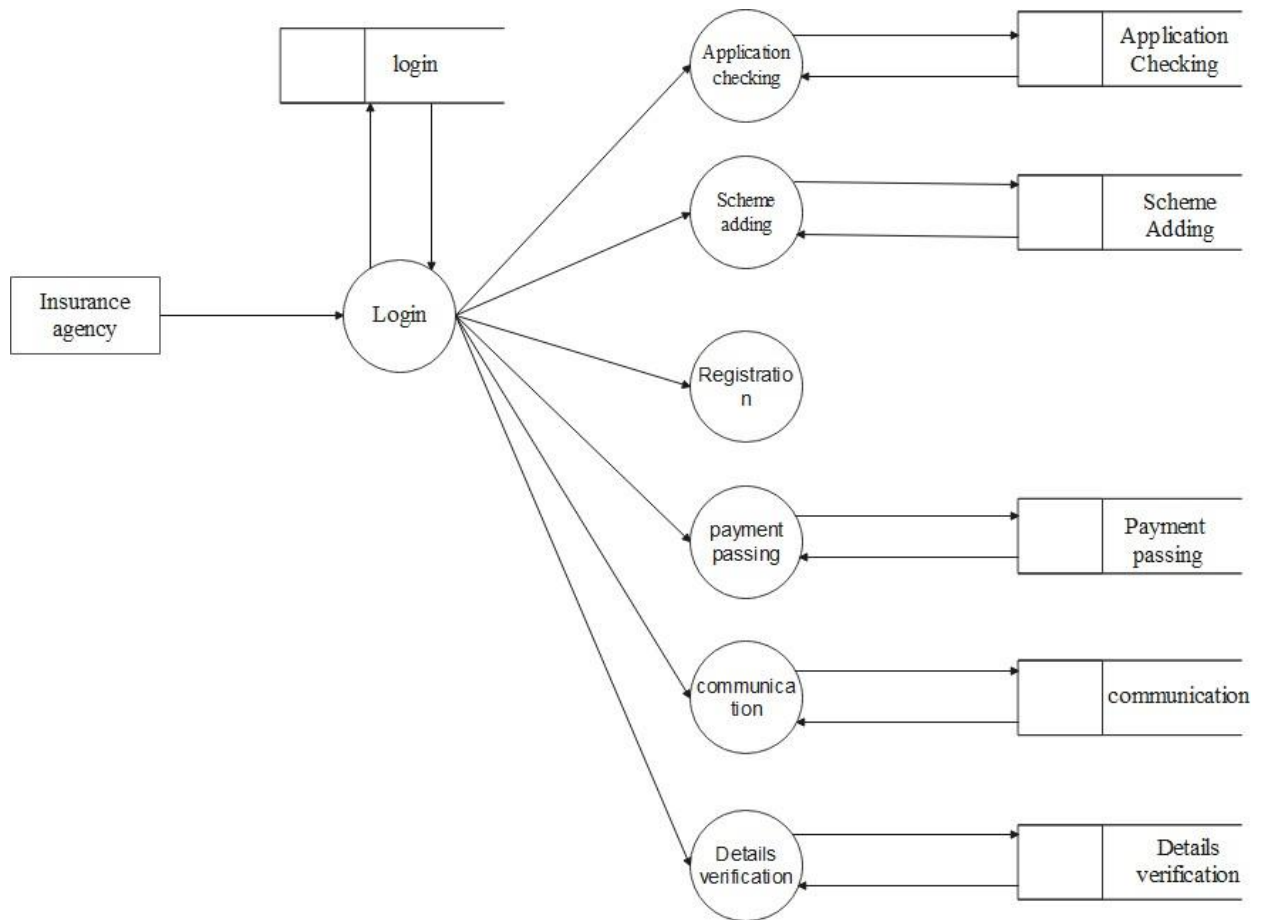


Figure 4.2.2 Insurance agency (Level 1)

4.2.3 Level 1 Worker

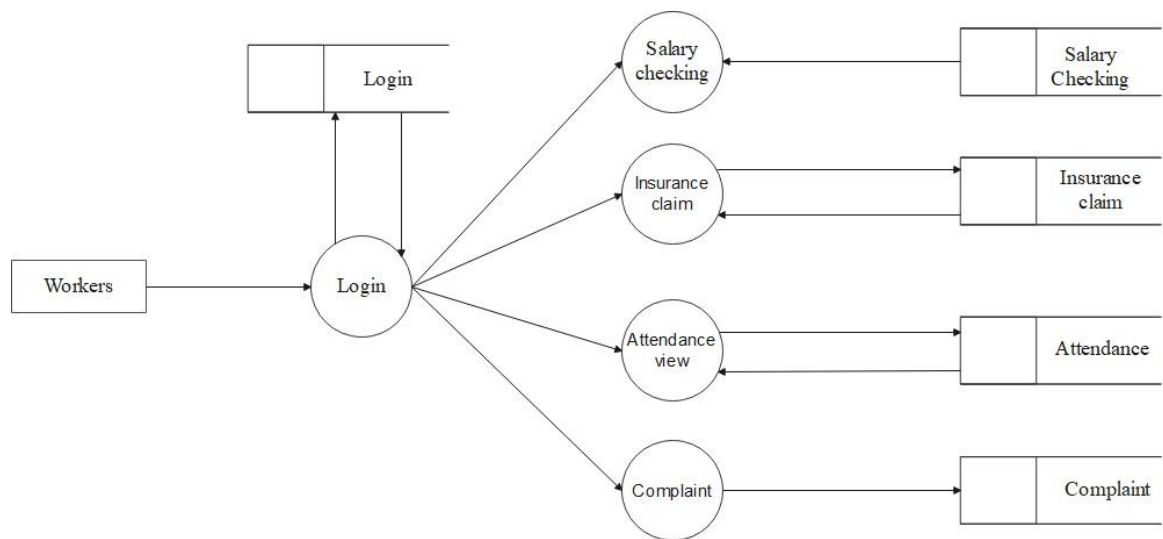


Figure 4.2.3 Worker (Level 1)

4.2.4 Level 1 Police

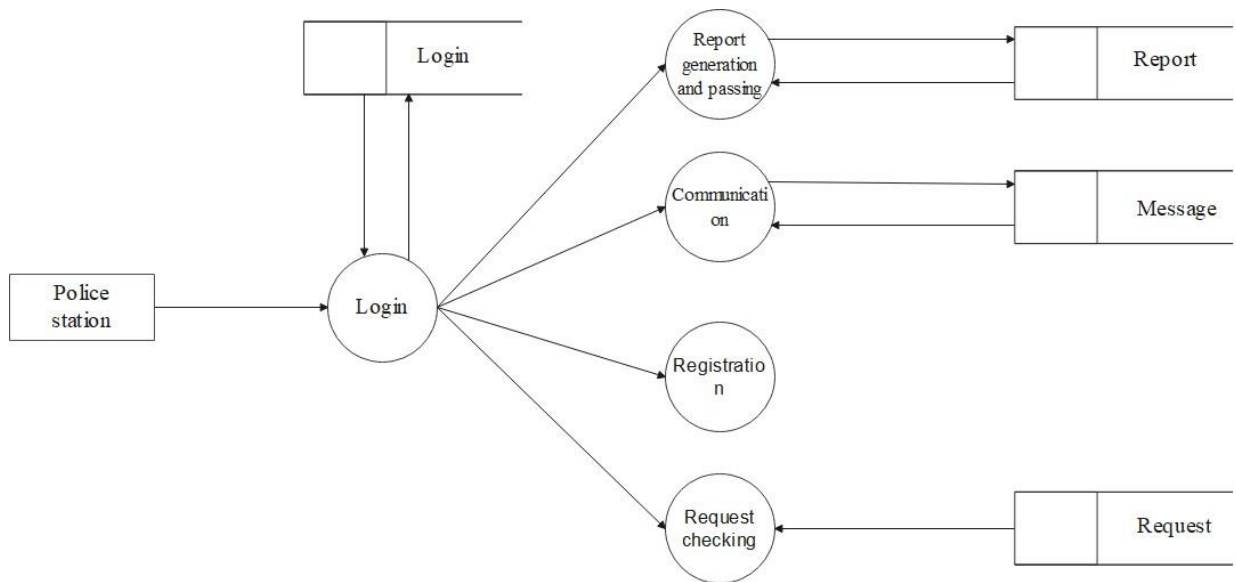


Figure 4.2.4 Police (Level 1)

4.2.5 Level 1 Labour commission

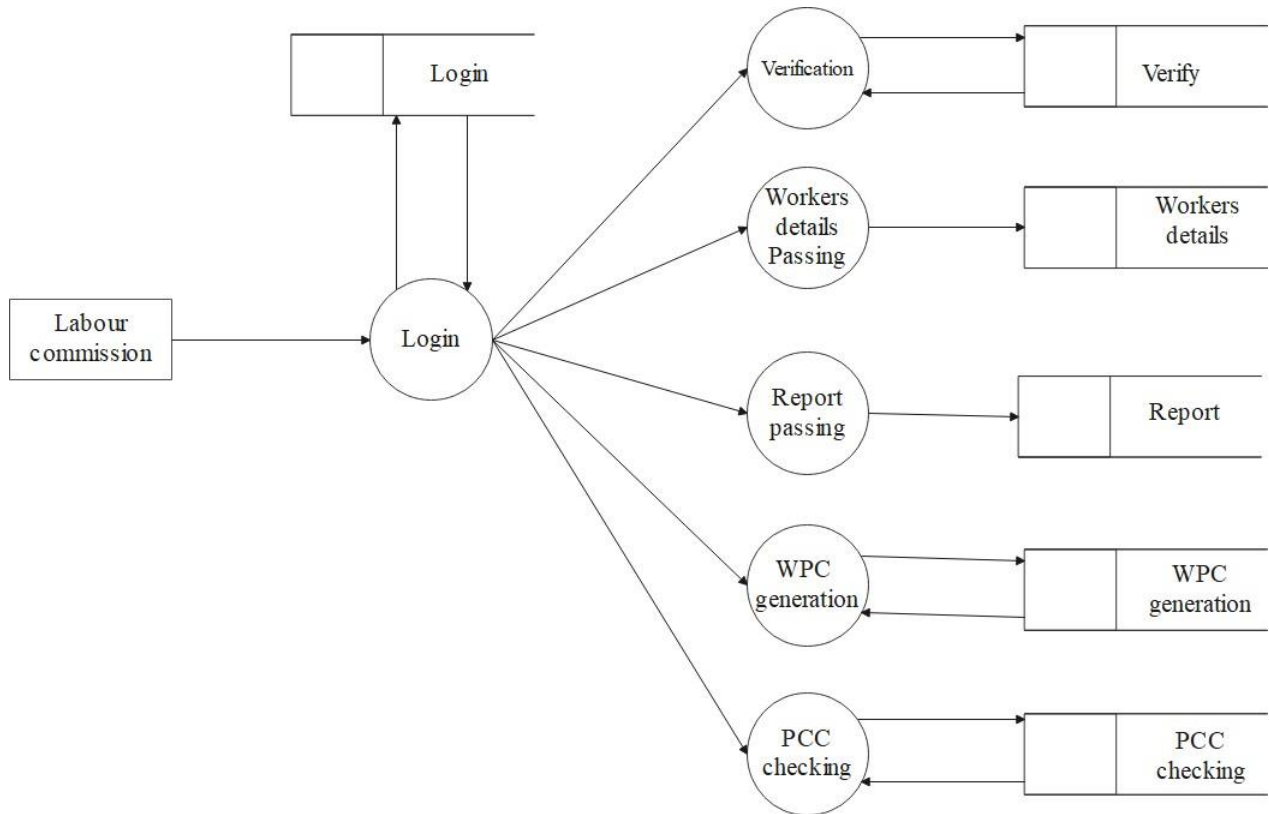


Figure 4.2.5 Labour commission (Level 1)

4.2.6 Level 1 Agency

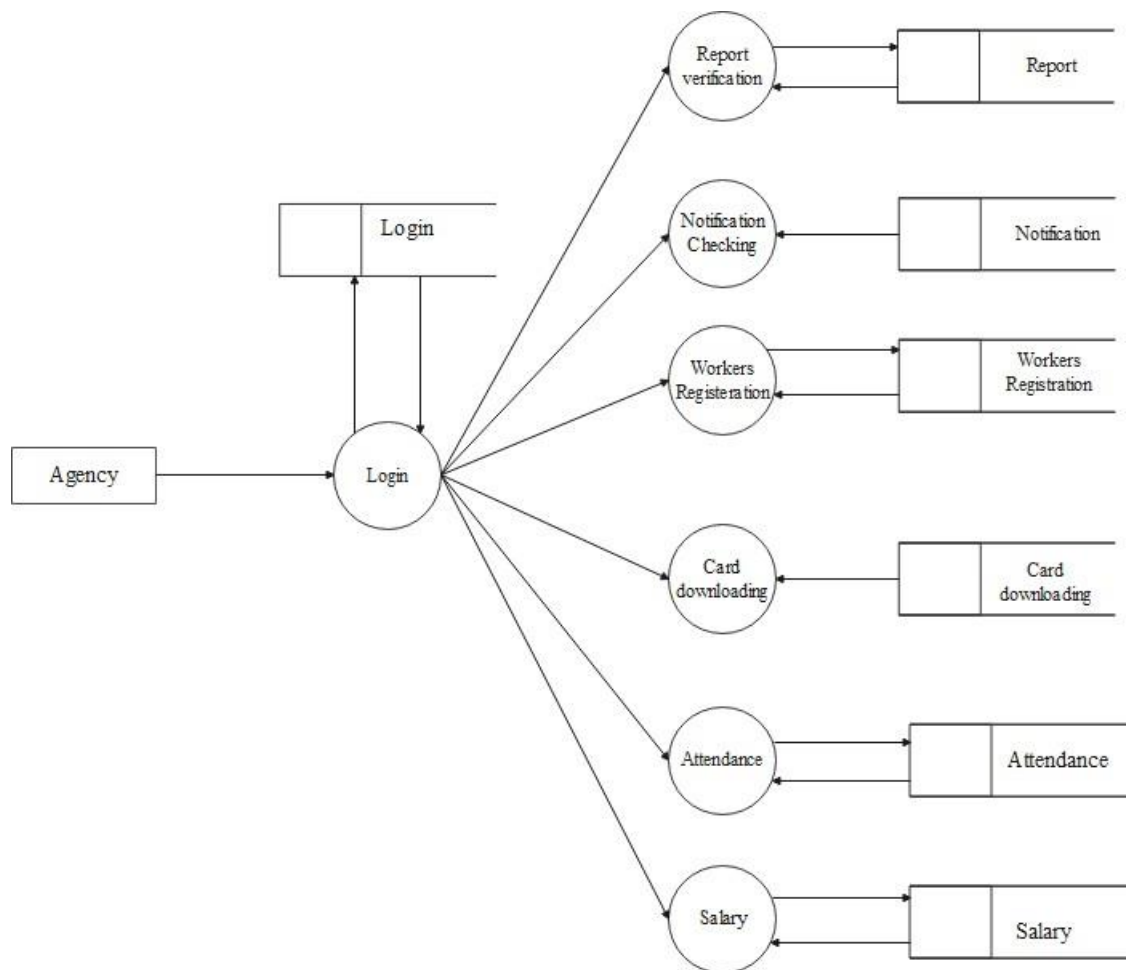


Figure 4.2.6 Agency (Level 1)

4.3 ER Diagram

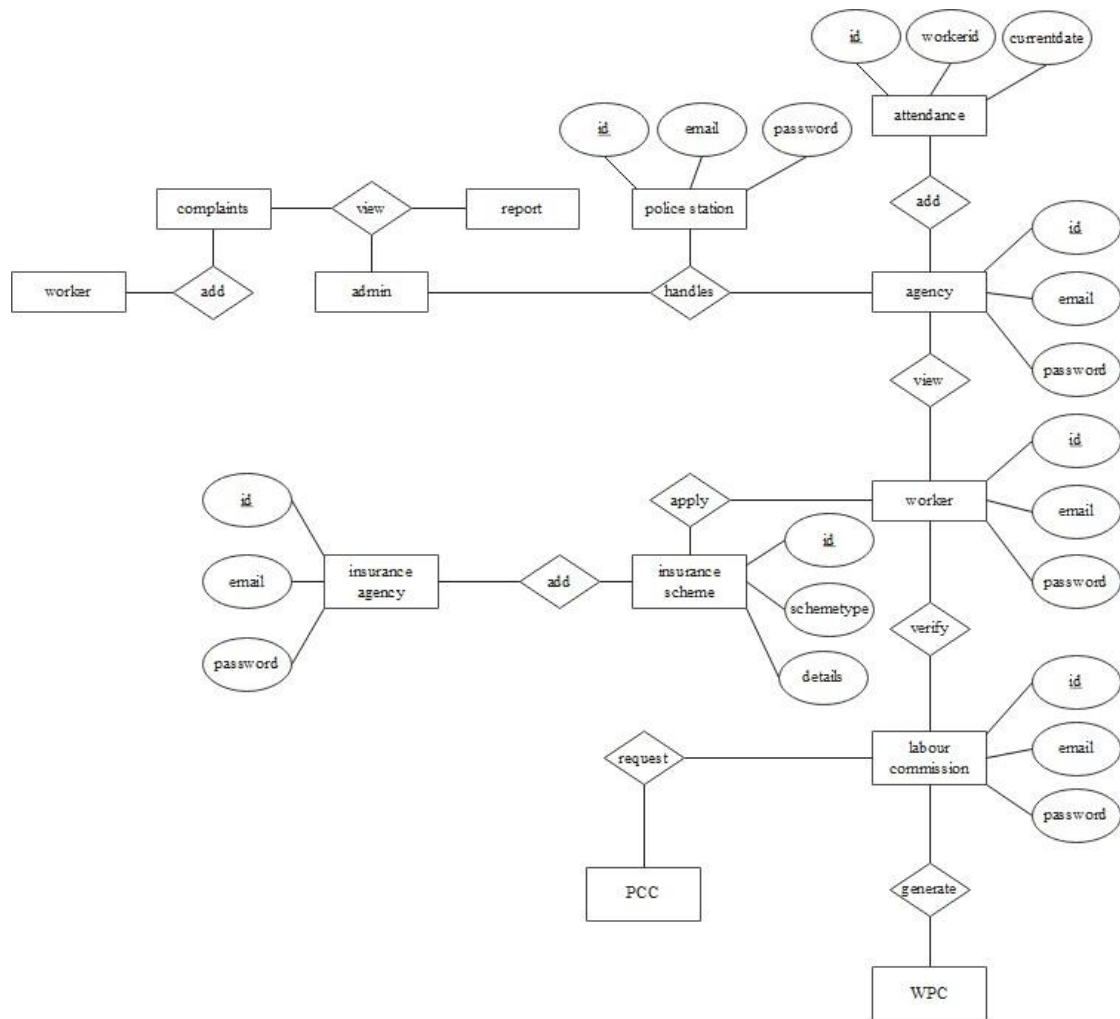


Figure 4.3.1 ER Diagram

4.4 Database Design

The most important aspect of building software systems is database design. The highest level in the hierarchy is the database. It is a set of inter-related files for real time processing. It contains the necessary data for problem solving and can be used by several users accessing data concurrently. The general objective of database design is to make the data access easy, inexpensive and flexible to the user.

Database design is used to define and then specify the structure of business used in the client/server system. A business object is nothing but information that is visible to the users of the system. The database must be a normalized one.

Database management system (DBMS) allows the data to be protected and organized separately from other resources like hardware, software and programs. DBMS is a software package, which contains components that are not found in other data management packages. The significance of DBMS is the separation of data as seen by the programs and data as stored on the direct access storage devices, i.e. the difference between logical and physical data.

In my project, I have used Microsoft SQL Server 2005 as the database to implement the data store part. The most important part in the database design is the identification of tables to be used. Database design activity deals with the design of the physical database. A key is to determine how the access paths are to be implemented. A physical path is derived from a logical path. Pointers, chains or other mechanisms may implement it.

The tables used in this project are:

Data Integrity And Constraints

Data Integrity refers to the validity of data. Data integrity can be compromised in a number of ways:

- Human errors when data is entered
- Errors that occur when data is transmitted from one computer to another
- Software bugs or viruses
- Hardware malfunctions, such as disk crashes

- Natural disasters, such as fires and floods There are many ways to minimize these threats to data integrity. These include:
- Backing up data regularly
- Controlling access to data via security mechanisms
- Designing user interfaces that prevent the input of invalid data
- Using error detection and correction software when transmitting data

Types of Data Integrity

This section describes the rules that can be applied to table columns to enforce different types of data integrity.

- **Null Rule**

A null rule is a rule defined on a single column that allows or disallows inserts or updates of rows containing a null (the absence of a value) in that column.

- **Unique Column Values**

A unique value rule defined on a column (or set of columns) allows the insert or update of a row only if it contains a unique value in that column (or set of columns).

- **Primary Key Values**

A primary key value rule defined on a key (a column or set of columns) specifies that each row in the table can be uniquely identified by the values in the key.

- **Referential Integrity Rules**

A referential integrity rule is a rule defined on a key (a column or set of columns) in one table that guarantees that the values in that key match the values in a key in a related table (the referenced value).

Referential integrity also includes the rules that dictate what types of data manipulation are allowed on referenced values and how these actions affect dependent values. The rules associated with referential integrity are:

- Restrict: Disallows the update or deletion of referenced data.
- Set to Null: When referenced data is updated or deleted, all associated dependent data is set to `NULL`.
- Set to Default: When referenced data is updated or deleted, all associated dependent data is set to a default value.

- **Cascade:** When referenced data is updated, all associated dependent data is correspondingly updated. When a referenced row is deleted, all associated dependent rows are deleted.
- **No Action:** Disallows the update or deletion of referenced data. This differs from `RESTRICT` in that it is checked at the end of the statement, or at the end of the transaction if the constraint is deferred. (Oracle uses No Action as its default action.)
- **Complex Integrity Checking**

Complex integrity checking is a user-defined rule for a column (or set of columns) that allows or disallows inserts, updates, or deletes of a row based on the value it contains for the column (or set of columns).

Integrity Constraints

An integrity constraint is a declarative method of defining a rule for a column of a table.

Oracle supports the following integrity constraints:

- **NOT NULL** constraints for the rules associated with nulls in a column
- **UNIQUE key** constraints for the rule associated with unique column values
- **PRIMARY KEY** constraints for the rule associated with primary identification values
- **FOREIGN KEY** constraints for the rules associated with referential integrity.

Oracle supports the use of `FOREIGN KEY` integrity constraints to define the referential integrity actions, including:

- Update and delete No Action
- Delete `CASCADE`
- Delete `SET NULL`
- **CHECK** constraints for complex integrity rules

For example, assume that you define an integrity constraint for the `salary` column of the `employees` table. This integrity constraint enforces the rule that no row in this table can contain a numeric value greater than 10,000 in this column. If an `INSERT` or `UPDATE` statement attempts to violate this integrity constraint, then Oracle rolls back the statement and returns an information error message.

The integrity constraints implemented in Oracle fully comply with ANSI X3.135-1989 and ISO 9075-1989 standards.

Advantages of Integrity Constraints

This section describes some of the advantages that integrity constraints have over other alternatives, which include:

- Enforcing business rules in the code of a database application
- Using stored procedures to completely control access to data
- Enforcing business rules with triggered stored database procedures

Declarative Ease

Define integrity constraints using SQL statements. When you define or alter a table, no additional programming is required. The SQL statements are easy to write and eliminate programming errors. Oracle controls their functionality. For these reasons, declarative integrity constraints are preferable to application code and database triggers. The declarative approach is also better than using stored procedures, because the stored procedure solution to data integrity controls data access, but integrity constraints do not eliminate the flexibility of ad hoc data access.

Centralized Rules

Integrity constraints are defined for tables (not an application) and are stored in the data dictionary. Any data entered by any application must adhere to the same integrity constraints associated with the table. By moving business rules from application code to centralized integrity constraints, the tables of a database are guaranteed to contain valid data, no matter which database application manipulates the information. Stored procedures cannot provide the same advantage of centralized rules stored with a table. Database triggers can provide this benefit, but the complexity of implementation is far greater than the declarative approach used for integrity constraints.

Maximum Application Development Productivity

If a business rule enforced by an integrity constraint changes, then the administrator need only change that integrity constraint and all applications automatically adhere to the modified constraint. In contrast, if the business rule were enforced by the code of each database application, developers would have to modify all applications source code and recompile, debug, and test the modified applications.

Immediate User Feedback

Oracle stores specific information about each integrity constraint in the data dictionary. You can design database applications to use this information to provide immediate user feedback about integrity constraint violations, even before Oracle runs and checks the SQL statement. For example, an Oracle Forms application can use integrity constraint definitions stored in the data dictionary to check for violations as values are entered into the fields of a form, even before the application issues a statement.

Superior Performance

The semantics of integrity constraint declarations are clearly defined, and performance optimizations are implemented for each specific declarative rule. The Oracle optimizer can use declarations to learn more about data to improve overall query performance. (Also, taking integrity rules out of application code and database triggers guarantees that checks are only made when necessary.)

Flexibility for Data Loads and Identification of Integrity Violations

You can disable integrity constraints temporarily so that large amounts of data can be loaded without the overhead of constraint checking. When the data load is complete, you can easily enable the integrity constraints, and you can automatically report any new rows that violate integrity constraints to a separate exceptions table.

The Performance Cost of Integrity Constraints

The advantages of enforcing data integrity rules come with some loss in performance. In general, the cost of including an integrity constraint is, at most, the same as executing a SQL statement that evaluates the constraint.

Types of Integrity Constraints

You can use the following integrity constraints to impose restrictions on the input of column values:

- NOT NULL Integrity Constraints
- UNIQUE Key Integrity Constraints
- PRIMARY KEY Integrity Constraints
- Referential Integrity Constraints
- CHECK Integrity Constraints

4.4.1 Table Design

Table Name: Agency Registration

Table Description: to store agency details

Field Name	Data Type	Size	Constraint	Description
id	int	3	Primary Key	id
agencyname	varchar	35	NOT NULL	agency name
address	varchar	35	NOT NULL	agency address
pincode	int	3	NOT NULL	agency pincode
district	varchar	25	NOT NULL	agency district
city	varchar	25	NOT NULL	agency city
agencyid	varchar	25	NOT NULL	agency id
contactnumber	int	3	NOT NULL	agency contact number
email	varchar	25	NOT NULL	agency email
password	varchar	25	NOT NULL	agency password

Table Name: Worker Registration

Table Description: to store worker details

Field Name	Data Type	Size	Constraint	Description
id	int	3	Primary Key	id
workername	varchar	25	NOT NULL	worker name
address	varchar	25	NOT NULL	worker address
pincode	int	3	NOT NULL	worker pincode
state	varchar	25	NOT NULL	worker state
district	varchar	25	NOT NULL	worker district
city	varchar	25	NOT NULL	worker city
aadharnumber	int	3	NOT NULL	worker aadharnumber
contactnumber	int	3	NOT NULL	worker contact number
email	varchar	25	NOT NULL	worker email
password	varchar	25	NOT NULL	worker password

Table name: police station registration

Table Description: to store police station details

Field Name	Data Type	Size	Constraint	Description
id	int	3	Primary Key	id
stationid	varchar	25	NOT NULL	Station id
addressline1	varchar	25	NOT NULL	Police station addressline1
Addressline2	varchar	25	NOT NULL	Police station addressline2
pincode	int	3	NOT NULL	Police station pincode
state	varchar	25	NOT NULL	Police station state
district	varchar	25	NOT NULL	Police station district
city	varchar	25	NOT NULL	Police station city
contactnumber	int	3	NOT NULL	Police station contact number
email	varchar	25	NOT NULL	Police station email
password	varchar	25	NOT NULL	Police station password

Table Name: Insurance agency registration

Table Description: to store insurance agency details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	id
agencyname	varchar	25	NOT NULL	Insurance agency name
state	varchar	25	NOT NULL	Insurance agency state
district	varchar	25	NOT NULL	Insurance agency district
contactnumber	int	3	NOT NULL	Insurance agency contact number
regid	varchar	25	NOT NULL	Insurance agency regid
email	varchar	25	NOT NULL	Insurance agency email
password	varchar	25	NOT NULL	Insurance agency password

Table Name: scheme insurance agency

Table Description: to store scheme insurance agency details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	id
scheme_type	varchar	25	NOT NULL	scheme_type
scheme_name	varchar	25	NOT NULL	scheme_name
scheme_amount	varchar	2	NOT NULL	scheme_amount
monthly_amount	varchar	25	NOT NULL	monthly_amount
interest_rate	varchar	25	NOT NULL	interest_rate
scheme_details	varchar	25	NOT NULL	scheme_details

Table Name: notification

Table Description: to store notification added by admin

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	id
notification	varchar	255	NOT NULL	Notification
currentdate	date	20	date	Current date

Table Name: complaint

Table Description: to store complaint added by worker

Field_Name	Data_Type	Size	Constraint	Description
id	int	4	Primary Key	id
complaintsubject	varchar	255	NOT NULL	Complaint subject
complaint	varchar	255	NOT NULL	Complaint description

Table Name: report

Table Description : to store report details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	report id
worker	int	4	Foreign Key (reference worker)	Worker id
police	int	4	Foreign Key (reference police)	Police id
currentdate	date	20	NOT NULL	Current date
file	file	50	NOT NULL	Report file

Table Name: salary

Table Description : to store salary details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	salary id
Job_category	varchar	39	NOT NULL	Job_category
Salary_per_day	int	20	NOT NULL	Salary_per_day

Table Name: worker job place

Table Description : to store worker job place details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	Worker job place id
Job_category	varchar	39	NOT NULL	Job_category
Working place	varchar	20	NOT NULL	working place
Worker_id	int	4	Foreign Key (reference worker)	Worker_id

Table Name: pcc apply date

Table Description : to store pcc apply details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	pcc id
Worker_id	int	4	Foreign Key (reference worker)	Worker_id
currentdate	date	20	NOT NULL	Current date

Table Name: worker scheme apply

Table Description : to store worker scheme details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	Worker job place id
Worker_id	int	4	Foreign Key (reference worker)	Worker_id
Scheme_id	int	4	Foreign Key (reference scheme)	Scheme_id
currentdate	date	20	NOT NULL	Current date

Table Name: worker claim insurance apply

Table Description : to store worker claim insurance details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	Worker job place id
Worker_id	int	4	Foreign Key (reference worker)	Worker_id
Scheme_id	int	4	Foreign Key (reference scheme)	Scheme_id
Applyscheme_id	int	4	Foreign Key (reference worker scheme)	Worker_scheme_id
file	file	1024	NOT NULL	file

Table Name: payment

Table Description : to store payment details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	Payment id
Nameoncard	varchar	20	NOT NULL	Card holder name
Cardnumber	int	20	NOT NULL	Card number
CVV	int	4	NOT NULL	CVV
Expire date	varchar	20	NOT NULL	Expire date
claimscheme	int	5	Foreign Key (reference scheme)	Scheme id
Amount	int	4	NOT NULL	Amount
Date	date	20	NOT NULL	Current date
insuranceagency	int	5	Foreign Key (reference insurance)	Insurance_id

Table Name: admin

Table Description : to store admin details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	admin id
email	varchar	20	NOT NULL	admin email
password	varchar	20	NOT NULL	admin password

Table Name: labourcommission

Table Description : to store labourcommission details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	labourcommission id
email	varchar	20	NOT NULL	labourcommission email
password	varchar	20	NOT NULL	labourcommission password

Table Name: chat

Table Description : to store chat details

Field Name	Data Type	Size	Constraint	Description
id	int	4	Primary Key	Chat id
Sender_id	int	5	NOT NULL	Sender_id
Receiver_id	int	5	NOT NULL	Receiver_id
message	int	5	NOT NULL	message
date	date	20	NOT NULL	Current time

4.5 NORMALIZATION

The entities along with their attributes can be stored in many different ways into a set of tables. The methods of arranging these attributes are called normal forms. The theory behind the arrangement of attributes into table is known as normalization theory. Normalization is a series of tests which we use against the data to eliminate redundancy and make sure that the data is associated with the correct table or relationship. It helps in,

- Minimization of duplication data.
- Providing flexibility to support different functional requirements.
- Enabling the model to be translated to database design

All relations in a relational database are required to satisfy the following conditions.

1. Data in First Normal Form

- Remove repeating data from table
- From the removed data, create one or more tables and relationships.

2. Data in Second Normal Form

- Identify tables and relationships with more than one key.
- Remove data that depends on only one part of the key.
- From the removed data, create one or more tables and relationships.

3. Data in Third Normal Form

- Remove that depends on other hand in the table or relationship and not on the key.
- From the removed data, create one or more tables and relationships.

Advantages of normalization are:

- Helps in reduction in the complexity of maintaining data integrity by removing the redundant data.
- It reduces inconsistency of data.
- Eliminate the repeating fields.
- Creates a row for each occurrence of a repeated field.
- Allows exploitation of column functions.

The second normal form has the characteristics of the first normal form and all the attributes must fully be dependent on the primary key. The proposed system is using second normal form as it is found most suitable.

4.6 Design of Each Sub System

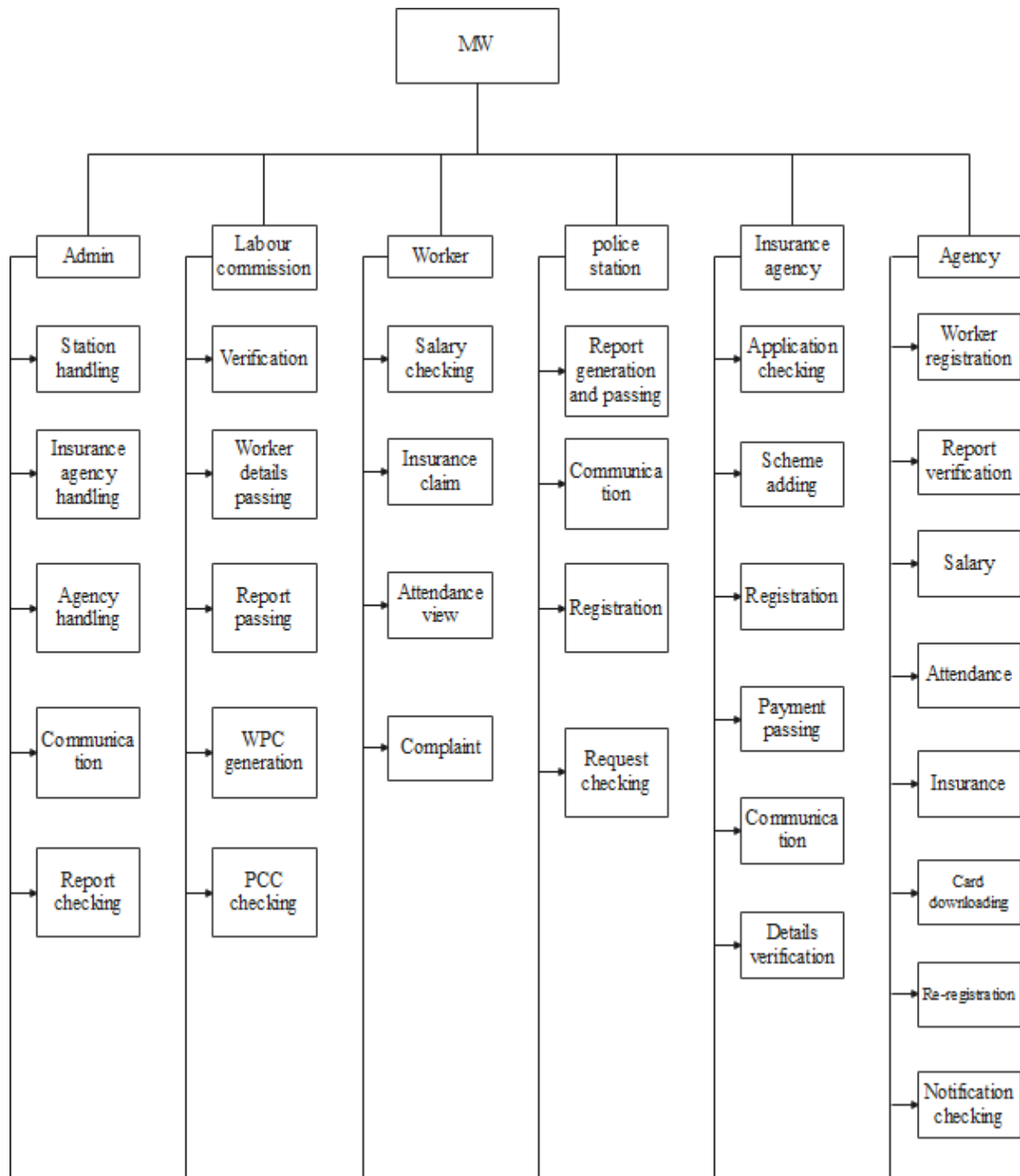


Figure 4.6 .1 subsystem diagram

4.7 UML DIAGRAMS

A Use Case Diagram displays the relationship among actors and Use Cases. Use Case Diagrams are drawn to capture the functional requirements of a system. Use Case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements.

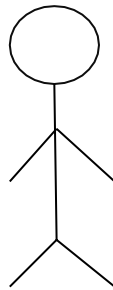
1. To identify functions and how roles interact with them.
2. For a high-level view of the system.
3. To identify internal and external actors.

The two main components of Use Case Diagrams are actors and cases.

1. Actor

Actor in a Use Case Diagrams is any entity that performs a role in one given system.

This could be a person, organization or an external system and usually drawn like skeleton shown below:



2. Use case

A Use case represents a person or an action within the system. It is drawn as an oval and named with the function. Symbol of use case is shown below:



4.7.1 Use case Diagram

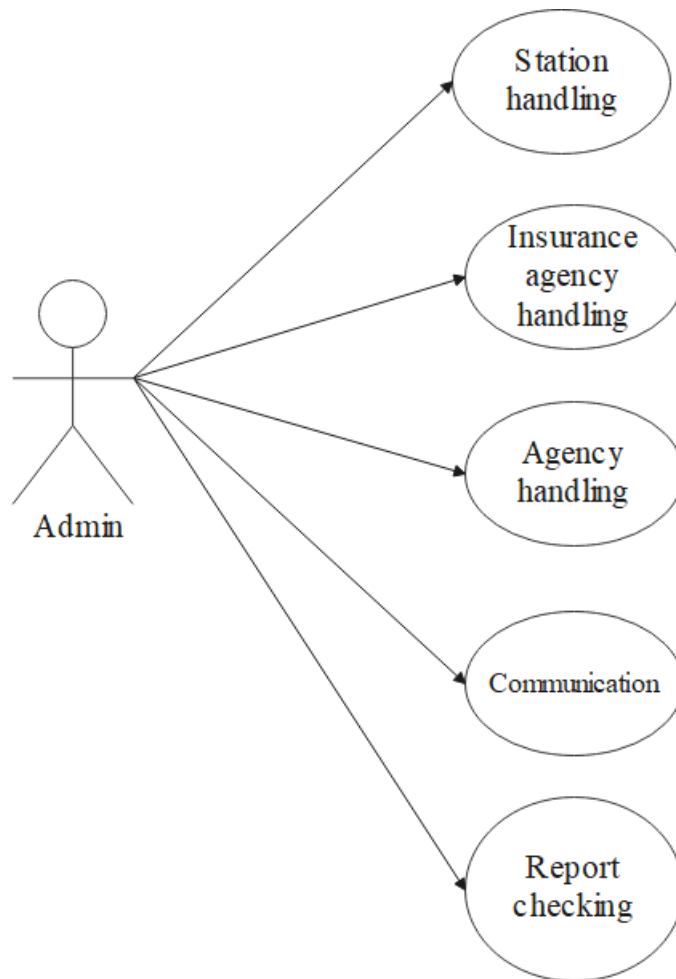


Figure 4.7.1 .1 Use Case Diagram for Admin

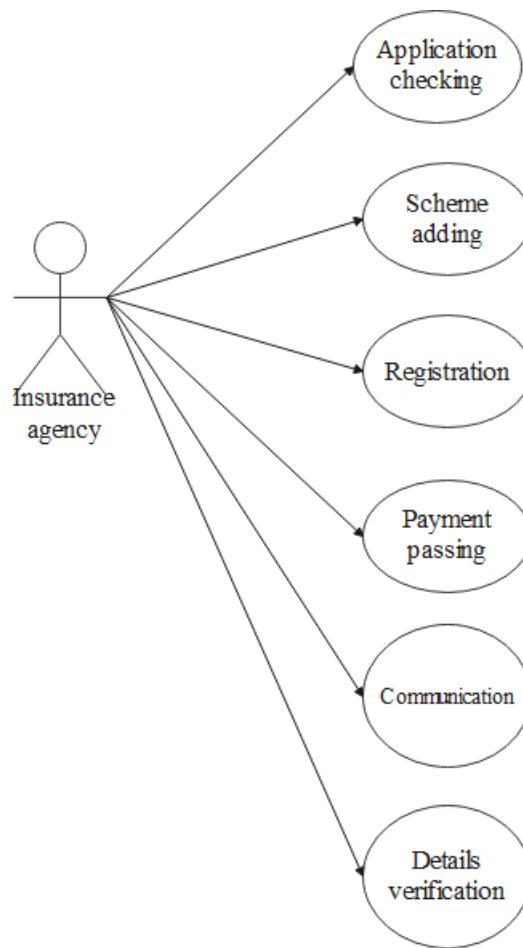


Figure 4.7.1 .2 Use Case Diagram for Insurance agency

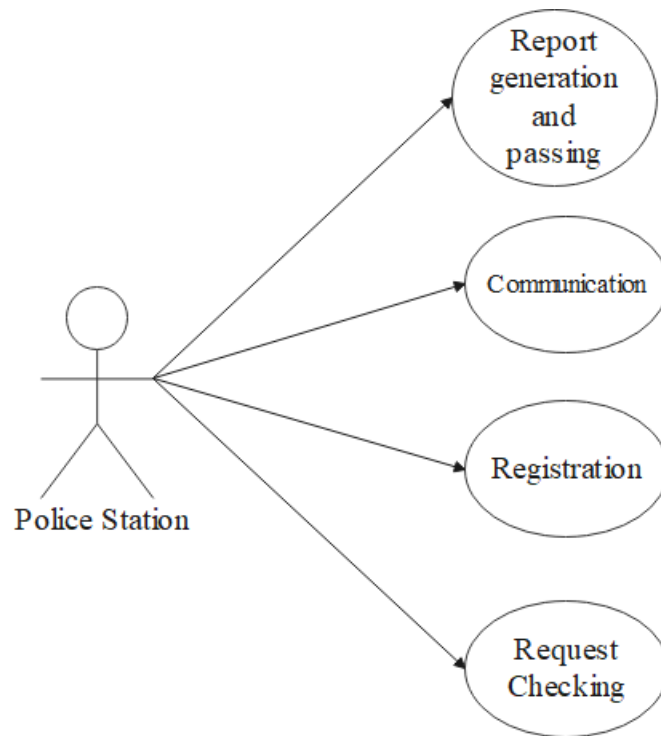


Figure 4.7.1 .3 Use Case Diagram for police station

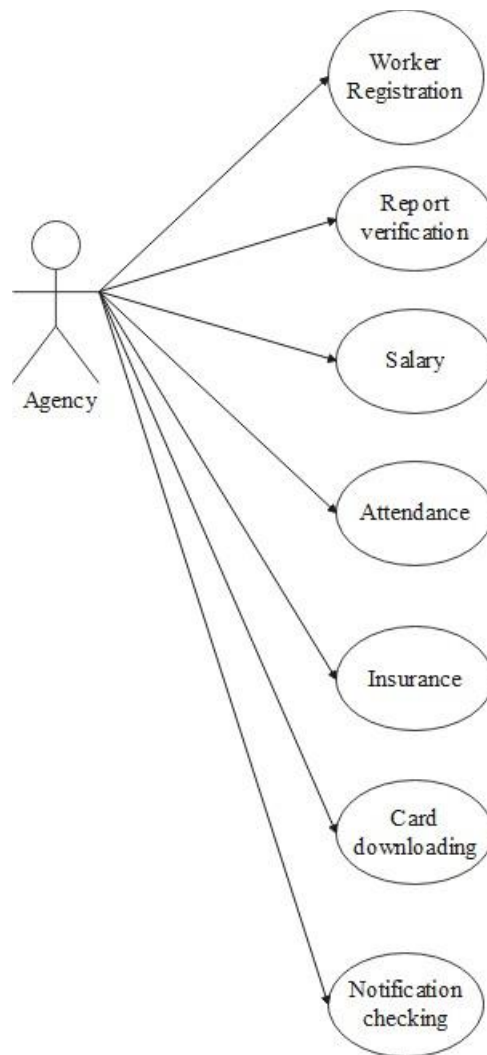


Figure 4.7.1 .4 Use Case Diagram for agency

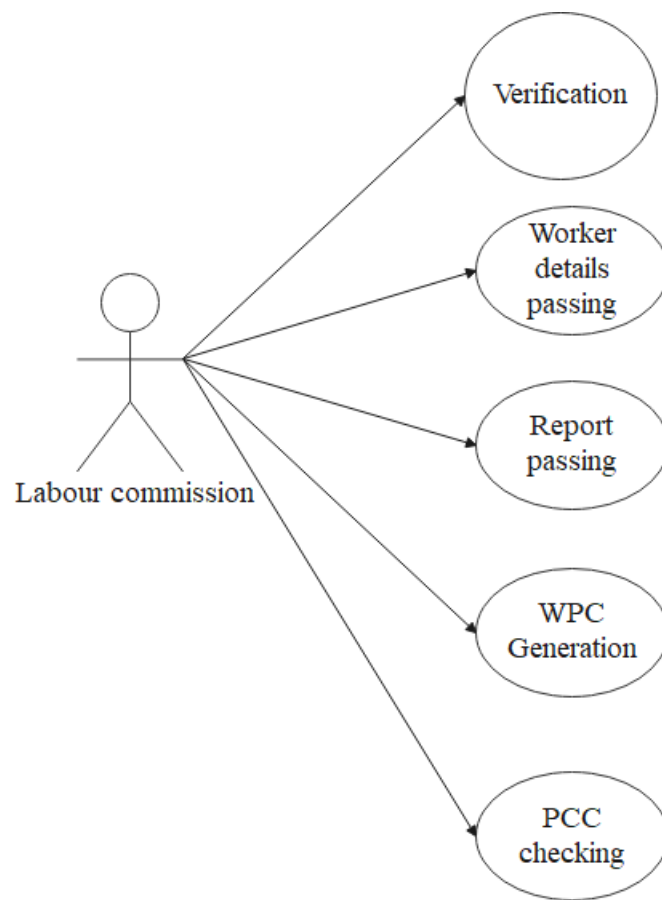


Figure 4.7.1.5 Use Case Diagram for Labour commission

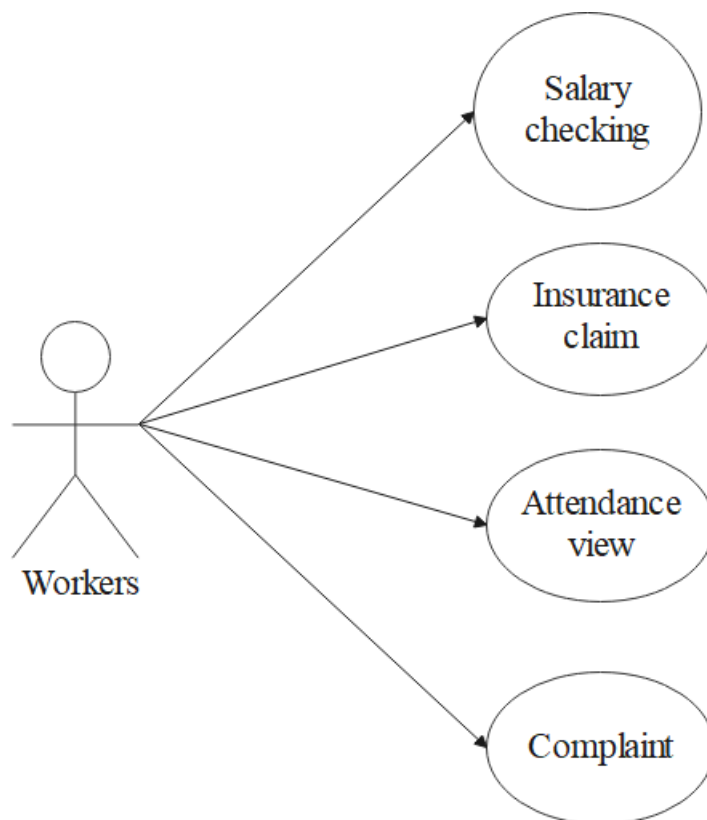


Figure 4.7.1.6 Use Case Diagram for workers

4.7.2 Class Diagram

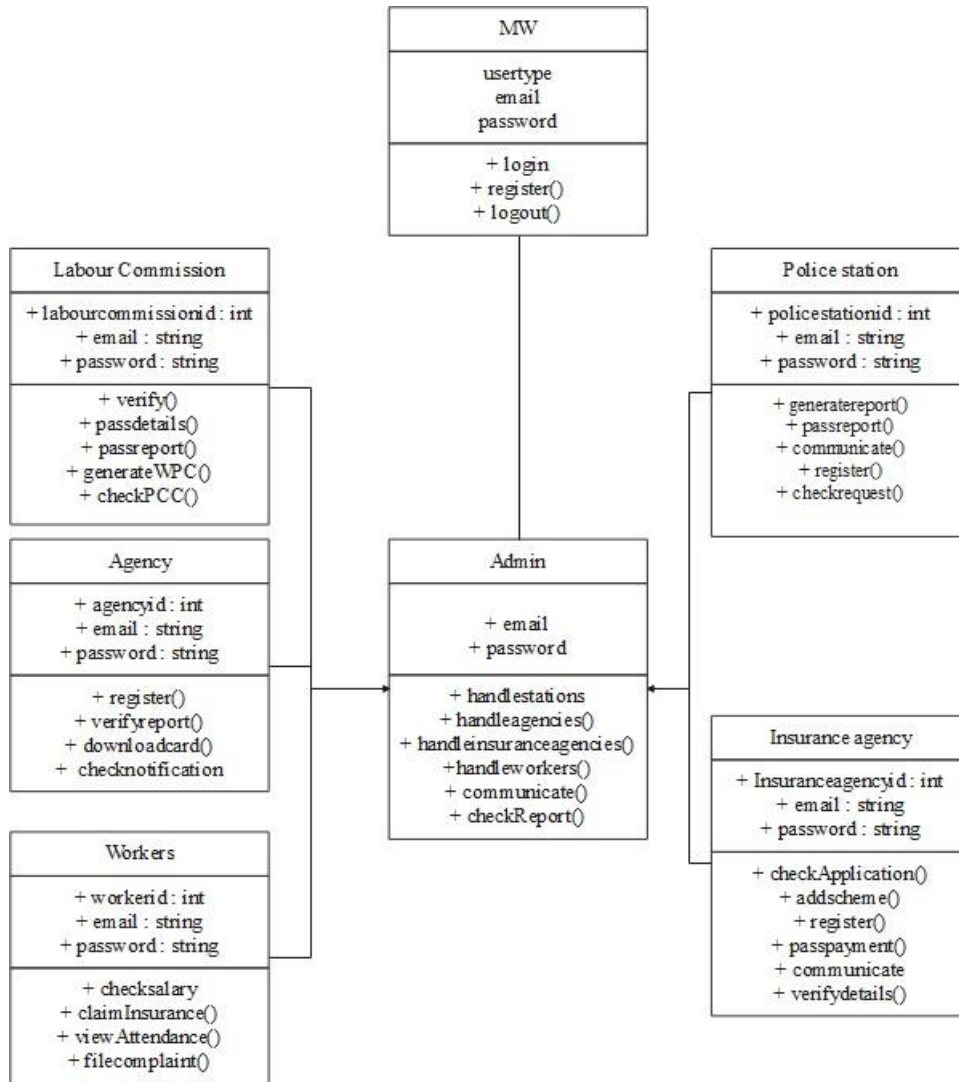


Figure 4.7.2.1 Class Diagram

4.7.3 Sequence Diagram

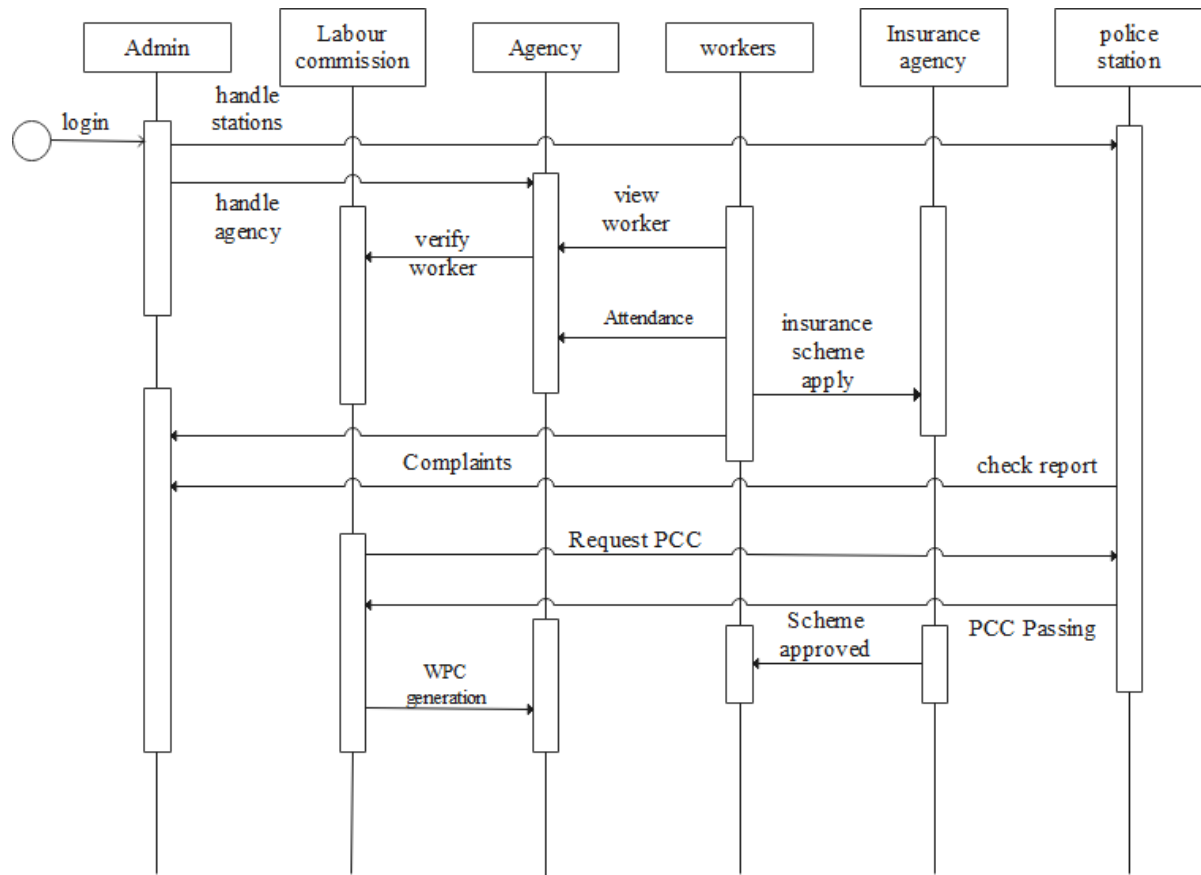


Figure 4.7.3.1 Sequence Diagram

CHAPTER – 5

CODING

5.1 Language Study

HTML: stands for Hypertext Markup Language. It is used to design web pages using a markup language. HTML is a combination of Hypertext and Markup language. Hypertext defines the link between web pages. A markup language is used to define the text document within the tag which defines the structure of web pages. This language is used to annotate (make notes for the computer) text so that a machine can understand it and manipulate text accordingly. Most markup languages (e.g. HTML) are human-readable. The language uses tags to define what manipulation has to be done on the text. HTML is a markup language used by the browser to manipulate text, images, and other content, in order to display it in the required format. HTML was created by Tim Berners-Lee in 1991. The first-ever version of HTML was HTML 1.0, but the first standard version was HTML 2.0, published in 1995. Elements and Tags: HTML uses predefined tags and elements which tell the browser how to properly display the content. Remember to include closing tags. If omitted, the browser applies the effect of the opening tag until the end of the page.

HTML page structure: The basic structure of an HTML page is laid out below. It contains the essential building-block elements (i.e. doctype declaration, HTML, head, title, and body elements) upon which all web pages are created.

<!DOCTYPE html>: This is the document type declaration (not technically a tag). It declares a document as being an HTML document. The doctype declaration is not case-sensitive.

`<html>`: This is called the HTML root element. All other elements are contained within it.

`<head>`: The head tag contains the “behind the scenes” elements for a webpage. Elements within the head aren’t visible on the front-end of a webpage. HTML elements used inside the `<head>` element include:

- `<style>`-This html tag allows us to insert styling into our webpages and make them appealing to look at with the help of CSS.
- `<title>`-The title is what is displayed on the top of your browser when you visit a website and contains the title of the webpage that you are viewing.
- `<base>`-It specifies the base URL for all relative URL’s in a document.
- `<noscript>`— Defines a section of HTML that is inserted when the scripting has been turned off in the user’s browser.
- `<script>`-This tag is used to add functionality in the website with the help of JavaScript.
- `<meta>`-This tag encloses the meta data of the website that must be loaded every time the website is visited. For eg:- the metadata charset allows you to use the standard UTF-8 encoding in your website. This in turn allows the users to view your webpage in the language of their choice. It is a self-closing tag.
- `<link>`— The „link“ tag is used to tie together HTML, CSS, and JavaScript. It is self-closing.

`<body>`: The body tag is used to enclose all the visible content of a webpage. In other words, the body content is what the browser will show on the front-end.

An HTML document can be created using any text editor. Save the text file using .html or .htm. Once saved as an HTML document, the file can be opened as a webpage in the browser.

Cascading Style Sheets:

fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independently of the HTML that makes up each web page. It describes how a webpage should look: it prescribes colours, fonts, spacing, and much more. In short, you can make your website look however you want. CSS lets developers and designers define how it behaves, including how elements are positioned in the browser. While HTML uses tags, CSS uses rulesets. CSS is easy to learn and understand, but it provides powerful control over the presentation of an HTML document.

- **CSS saves time:** You can write CSS once and reuse the same sheet in multiple HTML pages.
 - **Easy Maintenance:** To make a global change simply change the style, and all elements in all the webpages will be updated automatically.
 - **Search Engines:** CSS is considered a clean coding technique, which means search engines won't have to struggle to "read" its content.
 - **Superior styles to HTML:** CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
 - **Offline Browsing:** CSS can store web applications locally with the help of an offline cache. Using this we can view offline websites.
- CSS Syntax:** CSS comprises style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule set consists of a selector and declaration block
- The selector points to the HTML element you want to style.
 - The declaration block contains one or more declarations separated by semicolons.
 - Each declaration includes a CSS property name and a value, separated by a colon.

Bootstrap:

Bootstrap is freely available for every. The main features of bootstrap is, it is very simple and

easy to use, hug JavaScript plugins are available, easily design mobile friendly website.

- Easy to Use
- Mobile-Friendly
- Customizable Bootstrap
- Simple Integration
- Pre-styled Components
- Responsive Features
- Browser Compatibility
- Great Grid System
- Extensive list of Components
- Bundled JavaScript plugins
- Good Documentation

features of bootstrap

Easy to use : Anybody with just basic knowledge of HTML and CSS can start using Bootstrap
Responsive features: Bootstrap's responsive CSS adjusts to phones, tablets, and desktops

Mobile-Friendly: Mobile-first approach: In Bootstrap 3, mobile-first styles are part of the core framework

Simple Integration : Bootstrap can be simply integrated along with distinct other platforms and frameworks, on existing sites and new ones too and one more things you can also utilize particular elements of Bootstrap along with your current CSS.

Pre-styled Components : Bootstrap approaches with pre-styled components for alerts, dropdowns, nav bars, Customizable Bootstrap: The Bootstrap can be customized as per the designs of your project.

Browser compatibility: Bootstrap is compatible with all modern browsers (Chrome, Firefox, Internet Explorer, Safari, and Opera)

Great grid system :Bootstrap is built on responsive 12-column grids, layouts and components. Whether you need a fixed grid or a responsive, it's only a matter of a few changes.

Bundled JavaScript plugins :The components such as drop down menu are made interactive with the numerous JavaScript plugins bundled in the bootstrap package.

Extensive list of components :Whether you need drop down menus, pagination or alert boxes, Bootstrap has got your covered. Some of the components pre styled are; Dropdowns, Button Groups, Navigation Bar, Breadcrumbs, Labels & Badges, Alerts, Progress Bar, And many others.

Base styling for most HTML elements :A website has many different elements such as headings, lists, tables, buttons, forms, etc. The HTML elements for which styles are provided are; Typography Code, Tables, Forms, Buttons, Images, Icons

JavaScript:

JavaScript is a popular programming language. JavaScript features are flexible. Many open- source libraries are available. GitHub contains a large volume of JavaScript code by developers across the world. JavaScript works well in the front end and back end. JavaScript has a simple syntax. Without any settings, anyone can execute JavaScript programs and make them user-friendly. One individual having basic knowledge of HTML, CSS and coding can work with JavaScript.

Features of JavaScript

Scripting :JavaScript executes the client-side script in the browser. Interpreter The browser interprets JavaScript code. Event Handling Events are actions. JavaScript provides event-handling options.

Case Sensitive :In JavaScript, names, variables, keywords, and functions are case-sensitive.

Control Statements :JavaScript has control statements like if-else-if, switch case, and loop. Users can write complex code using these control statements.

Objects as first-class Citizens :JavaScript arrays, functions, and symbols are objects which can inherit the Object prototype properties. Objects being first-class citizens means Objects can do all tasks.

Supports Functional Programming :JavaScript functions can be an argument to another function, can call by reference, and can assign to a variable.

Dynamic Typing :JavaScript variables can have any value type. The same variable can have a string value, an integer value, or any other.

Client-side Validations :JavaScript client-side validations allow users to submit valid data to the server during a form submission.

Platform Independent :JavaScript will run in the same way in all systems with any operating system.

Async Processing :JavaScript async-await and promise features provide asynchronous nature. As the processes run in parallel, it improves processing time and responsiveness.

Prototype-based :JavaScript follows 'Object. Prototype' functions instead of class inheritance.

Python :

Python provides many useful features which make it popular and valuable from the other programming languages. It supports object-oriented programming, procedural programming approaches and provides dynamic memory allocation.

We have listed below a few essential features.

Easy to Learn and Use :Python is easy to learn as compared to other programming languages. Its syntax is straightforward and much the same as the English language. There is no use of the semicolon or curly-bracket, the indentation defines the code block. It is the recommended programming language for beginners.

Expressive Language :Python can perform complex tasks using a few lines of code. A simple example, the hello world program you simply type `print("Hello World")`. It will take only one line to execute, while Java or C takes multiple lines.

Interpreted Language :Python is an interpreted language; it means the Python program is executed one line at a time. The advantage of being interpreted language, it makes debugging easy and portable.

Cross-platform Language :Python can run equally on different platforms such as Windows, Linux, UNIX, and Macintosh, etc. So, we can say that Python is a portable language. It enables programmers to develop the software for several competing platforms by writing a program only once.

Free and Open Source :Python is freely available for everyone. It is freely available on its official website www.python.org. It has a large community across the world that is dedicatedly working towards make new python modules and functions. Anyone can contribute to the Python community. The open-source means, "Anyone can download its source code without paying any penny."

Object-Oriented Language :Python supports object-oriented language and concepts of classes and objects come into existence. It supports inheritance, polymorphism, and encapsulation, etc. The object-oriented procedure helps to programmer to write reusable code and develop applications in less code.

Extensible :It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our Python code. It converts the program into byte code, and any platform

can use that byte code.

Large Standard Library :It provides a vast range of libraries for the various fields such as machine learning, web developer, and also for the scripting. There are various machine learning libraries, such as Tensor flow, Pandas, Numpy, Keras, and Pytorch, etc. Django, flask, pyramids are the popular framework for Python web development.

GUI Programming Support :Graphical User Interface is used for the developing Desktop application. PyQt5, Tkinter, Kivy are the libraries which are used for developing the web application.

Integrated :It can be easily integrated with languages like C, C++, and JAVA, etc. Python runs code line by line like C,C++ Java. It makes easy to debug the code.

Embeddable :The code of the other programming language can use in the Python source code. We can use Python source code in another programming language as well. It can embed other language into our code.

Django framework

Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. The Django is very demanding due to its rapid development feature. It takes less time to build application after collecting client requirement. By using Django, we can build web applications in very less time. Django is designed in such a manner that it handles much of configure things automatically, so we can focus on application development only.**History :**Django was design and developed by Lawrence journal world in 2003 and publicly released under BSD license in July 2005. Currently, DSF (Django Software Foundation) maintains its development and release cycle. Django was released on 21, July 2005. Its current stable version is 2.0.3 which was released on 6 March, 2018.

Features of Django

Rapid Development :Django was designed with the intention to make a framework which takes less time to build web application. The project implementation phase is a very time taken but Django creates it rapidly.

Secure :Django takes security seriously and helps developers to avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery etc. Its user authentication system provides a secure way to manage user accounts and passwords.

Scalable :Django is scalable in nature and has ability to quickly and flexibly switch from small to

large scale application project.

Fully loaded :Django includes various helping task modules and libraries which can be used to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds etc.

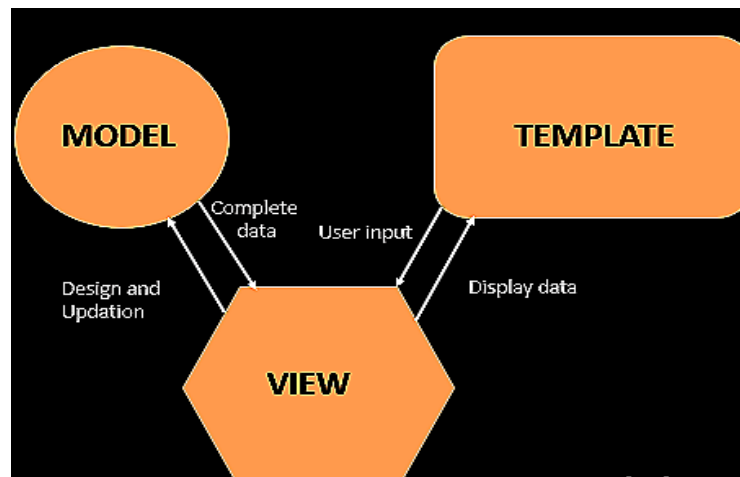
Versatile :Django is versatile in nature which allows it to build applications for different-different domains. Now a days, Companies are using Django to build various types of applications like: content management systems, social networks sites or scientific computing platforms etc.

Open Source :Django is an open source web application framework. It is publicly available without cost. It can be downloaded with source code from the public repository. Open source reduces the total cost of the application development.

Vast and Supported Community :Django is an one of the most popular web framework. It has widely supportive community and channels to share and connect.

Django MVT

The MVT (Model View Template) is a software design pattern. It is a collection of three important components Model View and Template. The Model helps to handle database. It is a data access layer which handles the data. The Template is a presentation layer which handles User Interface part completely. The View is used to execute the business logic and interact with a model to carry data and renders a template. Although Django follows MVC pattern but maintains it's own conventions. So, control is handled by the framework itself. There is no separate controller and complete application is based on Model View and Template. That's why it is called MVT application. Here, a user requests for a resource to the Django, Django works as a controller and check to the available resource in URL. If URL maps, a view is called that interact with model and template, it renders a template. Django responds back to the user and sends a template as a response.



Django Model

In Django, a model is a class which is used to contain essential fields and methods. Each model class maps to a single table in the database. Django Model is a subclass of `django.db.models.Model` and each field of the model class represents a database field (column). Django provides us a database-abstraction API which allows us to create, retrieve, update and delete a record from the mapped table. Model is defined in `Models.py` file. This file can contain multiple models.

Django Views

A view is a place where we put our business logic of the application. The view is a python function which is used to perform some business logic and return a response to the user. This response can be the HTML contents of a Web page, or a redirect, or a 404 error. All the view function are created inside the `views.py` file of the Django app.

Django Templates

Django provides a convenient way to generate dynamic HTML pages by using its template system. A template consists of static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted. In HTML file, we can't write python code because the code is only interpreted by python interpreter not the browser. We know that HTML is a static markup language, while Python is a dynamic programming language., Django template engine is used to separate the design from the python code and allows us to build dynamic web pages.

SQLite3 DATABASE

Being a very lightweight database management system, SQLite is very popular. No administration was required for operating a program in SQLite. However, it can only handle low to medium traffic HTTP requests. Also, the size of the database is usually restricted to 2GB. Even with these limitations, the SQLite advantages have gained more attention from the users. Some of the SQLite advantages are listed below:

Lightweight database management system:

Easy to use as embedded software with various electronic devices.

Better Performance:

- Very flexible.
- Fast reading and writing operations.
- Loads only the required data and not the entire file.
- Only overwrite the edited parts of a file and not the entire file.
- Facilitates an efficient way for data storage.
- Variable column length of columns thus allows allocating only the spaces that a field needs.

Installation not Needed:

- Easy to learn.
- No need to install.
- No Configuration Required.

Reliable:

- Contents are continuously updated.
- Less bug-prone than custom-written I/O code files.
- Smaller queries than equivalent procedural codes.

Portable:

- Portable across all 32-bit and 64-bit operating systems and big- and little-endian architectures.
- Facilitates work on multiple databases on the same session at the same time.
- Cross-platform DBMS.
- Available on both UNIX (Linux, Mac OS-X, Android, iOS) and Windows (Win32, WinCE,

WinRT).

- No compatibility issue with any programming languages.
- Facilitates API for a large range of programming languages.
- Facilitates simple and easy-to-use API.

Accessible:

- Accessible through a wide variety of third-party tools.
- More likely to be recoverable if data has been lost.
- Data in SQLite lives longer than co

Reduce Cost and Complexity:

- Free to use.
- Open-source.
- No license required to work with SQLite
- Does n't require a different server process or system to operate and is thus Serverless.
- No need for lengthy and error-prone procedural queries
- Content can be accessed and updated using concise SQL queries.

Open-source.

- No license required to work with SQLite.
- Does n't require a different server process or system to operate and is thus Server less.
- No need for lengthy and error-prone procedural queries.
- Content can be accessed and updated using concise SQL queries.

5.2 Functional Description

Administrator Module

- The Admin Module functions as a centralized hub responsible for coordinating and managing the overall labour system, encompassing a spectrum of crucial functionalities.
- It assumes the responsibility of overseeing police stations, maintaining accurate and updated information related to their locations and activities.
- Managing interactions with insurance agencies, this module facilitates a collaborative environment for efficient insurance-related processes.
- The handling of agencies is a fundamental function, encompassing the management of worker interactions, regulatory compliance, and agency-specific data.
- Effective communication is fostered through this module, facilitating seamless information exchange between diverse components of the labour ecosystem.
- Furthermore, the Admin Module undertakes the critical task of reviewing and analysing reports generated by various modules, providing insights that contribute to informed decision-making and effective regulation.

Insurance agency Module

- The Insurance Agency module plays a crucial role in managing insurance-related aspects within the labour ecosystem, encompassing diverse functionalities.
- It diligently verifies insurance applications, ensuring that they meet prescribed criteria before proceeding to the next stages of processing.
- This module oversees the availability and management of diverse insurance schemes, tailoring coverage options to cater to the specific needs of workers.
- Registration procedures enable insurance agencies to integrate seamlessly into the system, fostering collaboration and comprehensive coverage.
- The Insurance Agency module processes payments, ensuring that insurance-related transactions are conducted accurately and transparently.
- Effective communication channels are established, allowing interaction between the insurance agency and other modules for streamlined information exchange.

Police Station Module

- The Police Station module functions as a cornerstone of law enforcement within the labor ecosystem, undertaking crucial functions in maintaining order and security.
- It is tasked with the generation, validation, and subsequent forwarding of reports detailing worker activities and incidents to relevant entities.
- Communication stands as a pivotal function, fostering effective interactions between the police station and other integral modules, ensuring seamless information flow.
- The registration process involves meticulously recording and managing cases and incidents concerning workers, enabling the application of appropriate legal measures.
- Ensuring efficient service, this module reviews and processes incoming requests from various entities, thus contributing to a harmonious and regulated labour framework.

Workers Module

- The Workers module serves as a dedicated platform designed to cater to the varied needs and concerns of individual workers, offering a suite of essential services.
- It facilitates the convenient checking of salary-related information, providing workers with insights into their remuneration.
- Insurance claims can be initiated and tracked through this module, empowering workers to seek necessary financial support in case of exigencies.
- A transparent mechanism for viewing attendance records is provided, enabling workers to monitor their attendance performance over time.
- This module also accommodates the submission of complaints, providing a direct channel for workers to voice concerns and grievances, thereby fostering a collaborative labor environment.

Agency Module

- The Agency module assumes a pivotal role as an intermediary entity that bridges the gap between workers and the overarching Labour Commission framework, encompassing multifaceted functionalities.
- Worker registration, a fundamental operation within this module, involves the methodical

compilation and submission of worker particulars, establishing a comprehensive database.

- It extends its services to encompass the critical verification of reports submitted by workers and other agencies, thereby maintaining transparency and regulatory compliance.
- Managing salary disbursement constitutes a key responsibility, ensuring timely compensation to workers while adhering to established norms.
- Attendance tracking, an integral feature, involves systematically monitoring worker attendance records to ascertain punctuality and adherence to work commitments.
- This module administers insurance coverage, safeguarding worker welfare by overseeing the enrollment and maintenance of insurance schemes.
- The Agency module facilitates the generation and distribution of identification cards, essential documents that validate workers' identities within the labor system.
- Processes of re-registration, as undertaken by this module, streamline the continuation of worker engagements by periodically updating their information.
- Active engagement in notification handling ensures that timely and relevant notifications are communicated to relevant stakeholders, promoting effective communication channels.

Labour commission Module

- The Labour Commission module serves as a pivotal entity responsible for overseeing the regulatory aspects of the labor system, encompassing diverse functionalities.
- Within this module, the process of verification meticulously validates worker-related data furnished by agencies to ensure accuracy and legitimacy.
- It functions as a conduit for the seamless passing of comprehensive worker details, provided by agencies, with a focus on maintaining data integrity.
- This module also undertakes the important task of meticulously reviewing and subsequently processing reports submitted by agencies, ensuring consistency and adherence to regulatory standards.
- It facilitates the generation of Work Permit Certificates, pivotal documents that endorse the lawful employment of workers, thus contributing to a regulated labor environment.
- Additionally, the Labour Commission module assumes the responsibility of scrutinizing and authenticating Police Clearance Certificates, a vital component of ensuring the security and credibility of the workforce.

5.3 System Coding

```
from . forms import students
from django.shortcuts import render,redirect,get_object_or_404
from django.http import HttpResponseRedirect,JsonResponse
from django.template import loader
from . models import Home
from . model1 import Home1
from . model2 import Home2
from . model3 import Home3
from . model4 import NOTI
from . model5 import SCHOME
from . model6 import COMP
from . model7 import Home7
from . model8 import Home8
from . model9 import Home9
from . model10 import Home10
from . model11 import Home11
from . model12 import Home12
from . model13 import Home13
from . forms import worker,police,insage,assaignwork,reportpolice,insuranceworker
from . forms import noti,schome,comp,woratt,salaryage
from . forms1 import login4
from datetime import date
from django.db.models import F
```

```

def reg(request):
    if request.method=='POST':
        tem = students(request.POST)
        if tem.is_valid():
            tem.save()
    else:
        tem=students()
    return render(request,'agencyreg.html',{ 'fmm':tem})

def rr(request):
    te=loader.get_template('new.html')
    return HttpResponse(te.render())

def agv(request):
    stu=Home.objects.all().values()
    return render(request,'agencyview.html',{ 'stu':stu})
def agencyworkerrr(request):
    stu=Home.objects.all().values()
    return render(request,'agencyworkerregistration.html',{ 'stu':stu})
def wr(request,id):
    if request.method=='POST':
        tem= worker(request.POST)
        my=Home.objects.get(id=id)
        if tem.is_valid():
            f=tem.save(commit=False)
            f.agency_id=my.id
            f.save()

    else:
        tem=worker()
    return render(request,'workreg.html',{ 'fmm12':tem})

def wv(request):

```



```

    stu=Home1.objects.all().values()
    return render(request,'workview.html',{'stu':stu})
def pr(request):
    if request.method=='POST':
        tem= police(request.POST)
        if tem.is_valid():
            tem.save()
        else:
            tem=police()
    return render(request,'policereg.html',{'fmm2':tem})
def pv(request):
    stu=Home2.objects.all().values()
    return render(request,'policeview.html',{'stu':stu})
def inr(request):
    if request.method=='POST':
        tem= insage(request.POST)
        if tem.is_valid():
            tem.save()
        else:
            tem=insage()
    return render(request,'insreg.html',{'fmm3':tem})

def inv(request):
    stu=Home3.objects.all().values()
    return render(request,'insview.html',{'stu':stu})
def newadmin(request):
    tmpp=loader.get_template('admin1.html')
    return HttpResponse(tmpp.render())
def newlogin(request,user_type):
    if user_type=="agency":
        if request.method=='POST':
            tem = login4(request.POST)
            if tem.is_valid():
                email1=tem.cleaned_data['email']
                password1=tem.cleaned_data['password']

```

```

    try:
        User=Home.objects.get(email=email1,password=password1)
        request.session['Home_id']=User.id
        return redirect('agencyhome')
    except Home.DoesNotExist:
        tem.add_error(None,'Invalid username or password')
else:
    tem=login4()
return render(request,'logiin.html',{'formm':tem})
if user_type=="police":
    if request.method=="POST":
        tem = login4(request.POST)
        if tem.is_valid():
            email1=tem.cleaned_data['email']
            password1=tem.cleaned_data['password']
            try:
                User=Home2.objects.get(email=email1,password=password1)
                request.session['Home2_id']=User.id
                return redirect('policehome')
            except Home2.DoesNotExist:
                tem.add_error(None,'Invalid username or password')
        else:
            tem=login4()
        return render(request,'logiin.html',{'formm':tem})
if user_type=="worker":
    if request.method=="POST":
        tem = login4(request.POST)
        if tem.is_valid():
            email1=tem.cleaned_data['email']
            password1=tem.cleaned_data['password']
            try:
                User=Home1.objects.get(email=email1,password=password1)
                request.session['Home1_id']=User.id
                return redirect('workhome')
            except Home1.DoesNotExist:
                tem.add_error(None,'Invalid username or password')

```

```

else:
    tem=login4()
    return render(request,'logiin.html',{'formm':tem})
if user_type=="insuranceagency":
    if request.method=="POST":
        tem = login4(request.POST)
        if tem.is_valid():
            email1=tem.cleaned_data['email']
            password1=tem.cleaned_data['password']
            try:
                User=Home3.objects.get(email=email1,password=password1)
                request.session['Home3_id']=User.id
                return redirect('insuranceagencyhome')
            except Home3.DoesNotExist:
                tem.add_error(None,'Invalid username or password')
        else:
            tem=login4()
            return render(request,'logiin.html',{'formm':tem})

def policehome1(request):
    user_id=request.session.get('Home2_id')
    try:
        User=Home2.objects.get(id=user_id)
    except Home2.DoesNotExist:
        return redirect('new')
    return render(request,'policehome.html',{'formm':User})

def insuranceagencyhome1(request):
    user_id=request.session.get('Home3_id')
    try:
        User=Home3.objects.get(id=user_id)
    except Home3.DoesNotExist:
        return redirect('new')
    return render(request,'insuranceagencyhome.html',{'formm':User})

```

CHAPTER – 6

TESTING

System Testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It certifies that the whole set of program hang together. System testing requires a test plan that consists of several keys, activities and steps to run program, string, system and user acceptance testing. The implementation of newly designed package is important in adopting a successful new system.

TESTING OBJECTIVES

- Testing is the process of correcting a program with intend of finding an error.
- A good test is one that has a high probability of finding a yet undiscovered error.
- A successful test is one that uncovers an undiscovered error.

6.1 Levels of Testing

6.1.1 Unit Testing

In this testing we test each module individually and integrate the overall system. Unit testing focuses verification efforts on the smaller unit of software design in the module. This is also known as „module“ testing. The modules of the system are tested separately. The testing is carried out during programming stage itself. In this testing step each module is found to work satisfactory as regard to the expected output from the module. There are some validation checks for verifying the data input given by the user. It is very easy to find error and debug the system.

6.1.2 Integration Testing

Data can be lost across an interface; one module can have an adverse effect on the other sub functions when combined by May not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the

interface. This testing was done with sample data. The need for integrated test is to find the overall system performance.

6.1.3 Black Box Testing

This testing attempts to find errors in the following areas or categories: Incorrect or missing functions, interface errors, errors in data structures, external database access, performance errors and initialization and termination errors.

6.1.4 Validation Testing

At the culmination of Black Box testing, software is completely assembled as a package, interface errors have been uncovered and corrected and final series of software tests, validation tests begins. Validation testing can be defined in many ways but a simple definition is that validation succeeds when the software functions in a manner that can be reasonably accepted by the customer.

After validation test has been conducted one of the two possible conditions exists.

- The function or performance characteristics confirm to specification and are accepted.
- A deviation from specification is uncovered and a deficiency list is created.

6.1.5 Output Testing

After performing the validation testing, the next step is output testing of the proposed system since no system could be useful if it doesn't produce the required data in the specific format. The output displayed or generated by the system under consideration is tested by, asking the user about the format displayed. The output format on the screen is found to be correct as the format was designed in the system according to the user needs. Hence the output testing doesn't result in any correction of the system.

6.1.6 User Acceptance Testing

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing and making change wherever required. This is done with regard to the following points:

- Output Screen design.
- Input Screen design.
- Menu driven system.

6.1.7 White Box Testing

White box testing is a testing case design method that uses the control structure of the procedural design to derive the test cases. The entire independent path in a module is exercised at least once. All the logical decisions are exercised at least once. Executing all the loops at boundaries and within their operational bounds exercise internal data structure to ensure their validity. In our project testing was conducted at every step. Initially each module was tested separately to check whether they gave the desired output for the given input. The forms used to enter data by user were validated and appropriate error messages were displayed if incorrect data was entered. Once the data was entered correctly, the processing was done and testing was done to check whether the correct output was obtained. Once the test cases were conducted successfully for each module, the modules were integrated together as a single system. After integration, the test cases were again applied to check whether the entire system as a whole produced the desired output. At times, the test cases failed and the shortcomings were noted down and appropriate corrections were done. Once the integration testing was performed correctly, output testing was done and it did not result in any change or correction in the system. Black box testing and white box testing was also conducted successfully. All the loops, decisions, relations were executed at least once before giving it to the users for testing. In black box testing, it was checked whether the data in the proper format was stored in the database or not. Also, it was checked whether the interfaces were working properly or not. On successful completion of these tests, the system was then given to undergo user acceptance testing where the users entered test data to check whether the correct output

was obtained. The users were satisfied with the output and thus the testing phase was completed successfully.

6.1.8 Test Data and Results

The primary goal of software implementation is the production of source code that is easy to read and understand. Clarification of source code helps in easier debugging, testing and modification. Source code clarification is enhanced by structural coding techniques, by good coding style, by appropriate supporting documents, by good internal comments and by the features provided in the modern programming language.

In our implementation phase, source code contains both global and formal variables. It contains predefined functions as well as the user defined functions. The result of the new system is compared with old system and supposes if the result is wrong the error must be debugged.

After the acceptance of the system by the user, the existing system should be replaced by this system. Any user handles this package very easily. It does not require any intensive training for the user. Procedures and functions involved in this system are very simple that anyone can understand and correspondingly act to the system with no difficulty.

CHAPTER – 7

IMPLEMENTATION

Implementation is an activity that is contained throughout the development phase. It is the process of bringing a developed system into operational use and turning it over to the user. The new system and its components are to be tested in a structured and planned manner. A successful system should be delivered and users should have the confidence that the system should have work efficiently and effectively. The more complex system being implemented, the more will be the system analysis and design effort required just for implementation.

Implementation is the stage of the system when the theoretical design is turned into working system. The plan contains an overview of the system, a brief description of the major tasks involved in the implementation, the overall resources needed to support the implementation effort, and any site implementation requirements. The plan is developed during the design phase and is updated during the Development phase. The outline shows the implementation plan.

There are three types of implementation:

- a. Implementation of a computer system for replacing the manual system. The problem encountered are converting files, training users, create accurate files.
- b. Implementation of new computer system to replacing an existing one. This is usually a difficult conversion. If not properly planned, there can be many problems. Some larger computer systems have taken as long as a year to convert.
- c. Implementation of modified application to replace an existing one using the same computer. This type of conversion relatively easy to handle, provided there are no major changes in file.

7.1 Implementation of Proposed System

After having user acceptance for the system developed, the implementation phase begins. Implementation is the stage of project during which theory is tuned into practice. During this phase, all the programs of the system are loaded into the user's computer. After loading the system training of the user starts. Such as type of training includes:

1. How to execute the package?
2. How to enter the data?
3. How to process the data (processing details)?
4. How to takeout the report?

The following two strategies are followed for running the system.

Parallel Run: In such run for a certain defined period, both the systems thereafter computerized and manual are executed in parallel. This strategy is helpful because of the following:

1. Manual result can be compared with the result of computerized system. For the care of demonstration of the success of this system, it was implemented with successfully running; manual systems and results are verified.
2. Failure of a computerized system at an early stage, do not affect the work of the organization, because the manual system continues to work as it used to do.

Pilot Run: In this type of run, some parts of the new system is installed first and executed successfully for the considerable time period. When the results are found satisfactory, only then the other parts are implemented. This strategy builds the confidence and errors are traced easily.

CHAPTER – 8

SECURITY BACKUP AND RECOVERY MECHANISMS

8.1 Online Help

Simply speaking, a backup is a copy of data. This copy includes important parts of database such as the control file and data files. A backup is a safeguard against unexpected data loss and application errors, should we lose the original data, can use the backup to make it available again. After developing our portal, a proper backup copy is stored in a separate system or medium, like CD and drives, from the primary data to protect against the possibility of data loss due to primary hardware or software failure.

Recovery from a backup typically involves restoring the data to the original location, or to an alternate location where it can be used in place of the lost or damaged data. A database is a very huge system with lose of data and transaction. The transaction in the database is executed at each seconds of time and is very critical to the database. If there is any failure or crash while executing the transaction, then it expected that no data is necessary to revert the changes of transaction to previously committed point. That means, any transaction in the system cannot be left at the stage of its failure. It should either be completed fully or rolled back to the previous consistent state.

8.2 User Manuals

A user manual is a technical document with a quite specific purpose to help non-technical people pinpoint and solve problems without assistance. Since user manual translate what's not comprehend to a language for everyone to understand, they are essential in technical sectors and most commonly associated with software and hardware, IT systems.

In our application user manual includes images and notes relating every functions in order to help the user to understand features of our system. Admin can add complaints against hotel or their food which can help the hotel managements for trying their best solutions. The hotels can add their images about menus which can be viewed by the user

and this also helps them immensely. It also includes screenshots. The main working of the system and its data flow is all elaborately described with the help of simplified diagrams and descriptions.

CHAPTER – 9

CONCLUSION

The project "Migrant Workers" has been successfully developed and implemented, serving as an efficient and comprehensive solution to address the multifaceted needs of managing migrant workers within the labor ecosystem. Leveraging the Django framework and utilizing an SQLite database, the software stands as a testament to technological innovation and its potential to streamline complex processes.

Throughout the project's lifecycle, meticulous attention was given to the limitations and challenges posed by the existing labor management systems. By adopting modern software technologies, the new system offers a robust platform that overcomes these limitations, facilitating seamless interactions among workers, agencies, law enforcement, insurance providers, and administrators.

The software's user-friendliness is a key feature, promoting easy interaction and collaboration between administrators and users. The well-structured modules, ranging from Worker Registration and Agency Management to Police Station Coordination and Insurance Handling, are designed to cater to various stakeholders' distinct requirements while upholding data integrity and regulatory compliance.

Incorporating principles of system analysis, system design, and development, the project followed a comprehensive development life cycle. Rigorous testing procedures, including Unit, Module, and Integration tests, were conducted to ensure the system's reliability and functionality across different levels of interaction.

As the project "Migrant Workers " successfully comes to fruition, it stands as a testimony to the power of technology in enhancing the management and regulation of labor dynamics. The software's ability to facilitate accurate worker registration, streamline communication between stakeholders, manage insurance processes, and ensure law enforcement coordination underscores its pivotal role in creating a more organized, secure, and transparent labor environment.

CHAPTER – 10

FUTURE ENHANCEMENT

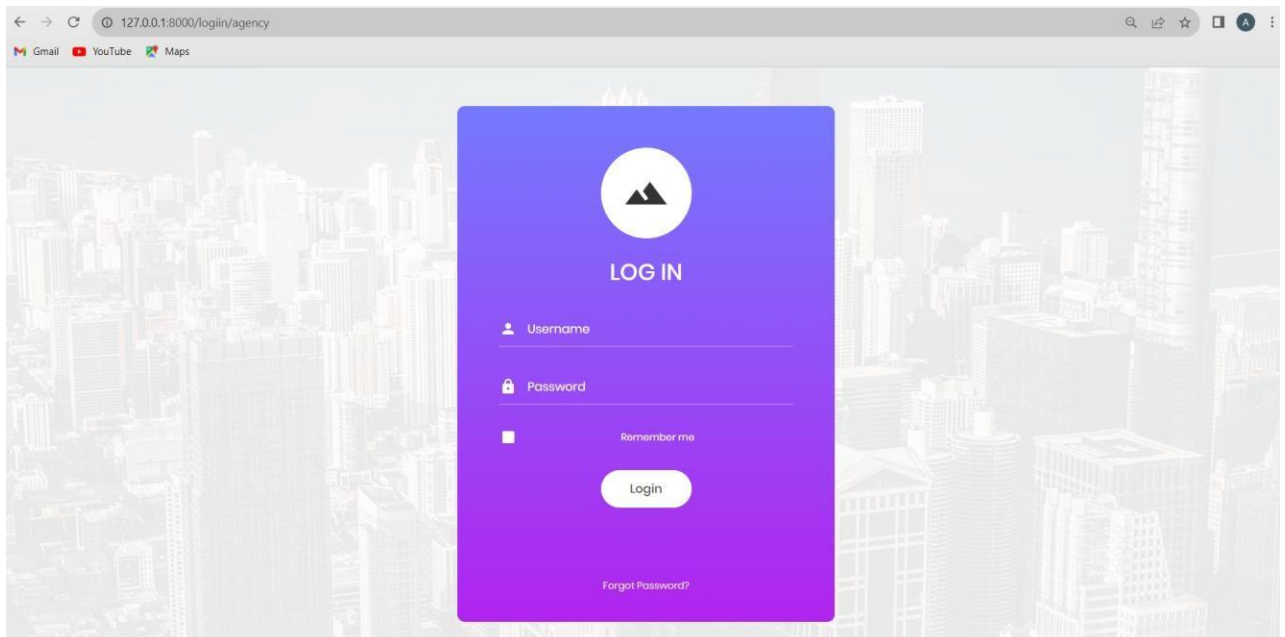
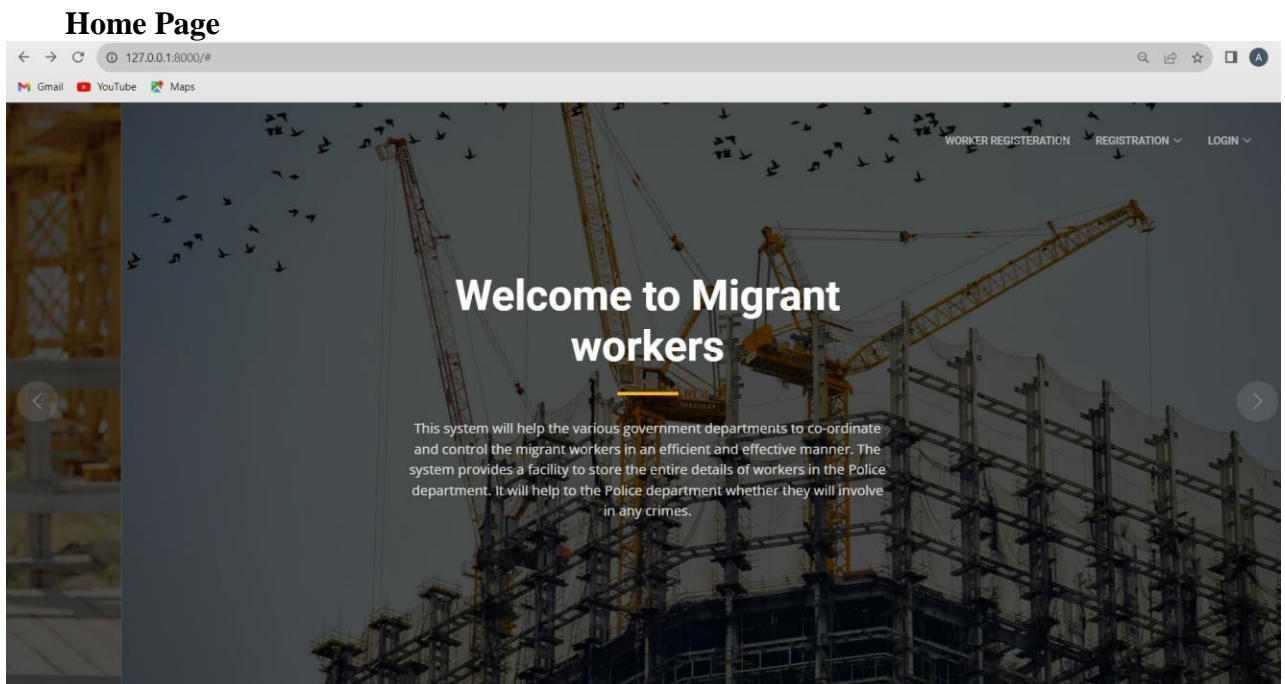
The changing requirements and the fast growth of technology will surely force the system to impose some changes forecasting those needs. Migrant Workers Management System has been thoughtfully designed to accommodate future enhancements, ensuring its resilience in a dynamic labour environment. The system's inherent flexibility allows for seamless integration of new features and functionalities in response to evolving requirements. Anticipated future enhancements for the migrant workers project include:

- Extend the reach and accessibility of the system through a dedicated mobile application, empowering workers, agencies, and other stakeholders to conveniently engage with the platform using their mobile devices. This mobile app could enable streamlined worker registration, real-time attendance tracking, and instant communication, fostering a more connected labour ecosystem.
- Incorporate a robust event organizer module within the system, facilitating the planning, coordination, and execution of various events beneficial to migrant workers. This module would encompass job fairs, skill development workshops, cultural celebrations, and community engagement activities, fostering a sense of belonging and empowerment.
- Leverage sensor technology to optimize resource allocation for migrant workers. Implementing sensor-enabled parking and table allocation systems could enhance logistical efficiency, while also introducing real-time insights into utilization patterns, contributing to improved living and working conditions.
- Introduce a sensor-enabled waiter calling system, tailored to the unique needs of migrant workers. By implementing this system, workers can conveniently request assistance or place orders within their accommodations or communal spaces, enhancing overall comfort and satisfaction.

These enhancements, aligned with the system's adaptable architecture, demonstrate a commitment to keeping pace with industry trends and user needs. The migrant workers project is positioned to leverage technology's transformative power to create a more efficient, inclusive, and supportive labour management ecosystem

APPENDIX

INPUT AND OUTPUT FORMS



AGENCY REGISTRATION

Agencyname:
Bestjob

Address:
Plot No. 12, 1st Floor, DDA Commercial Complex, Karam Pura, New Delhi, Delhi 110015

Pincode:
110001

District:
Shahdra


City:
Delhi

Agencyid:
Bestjob45@2bs

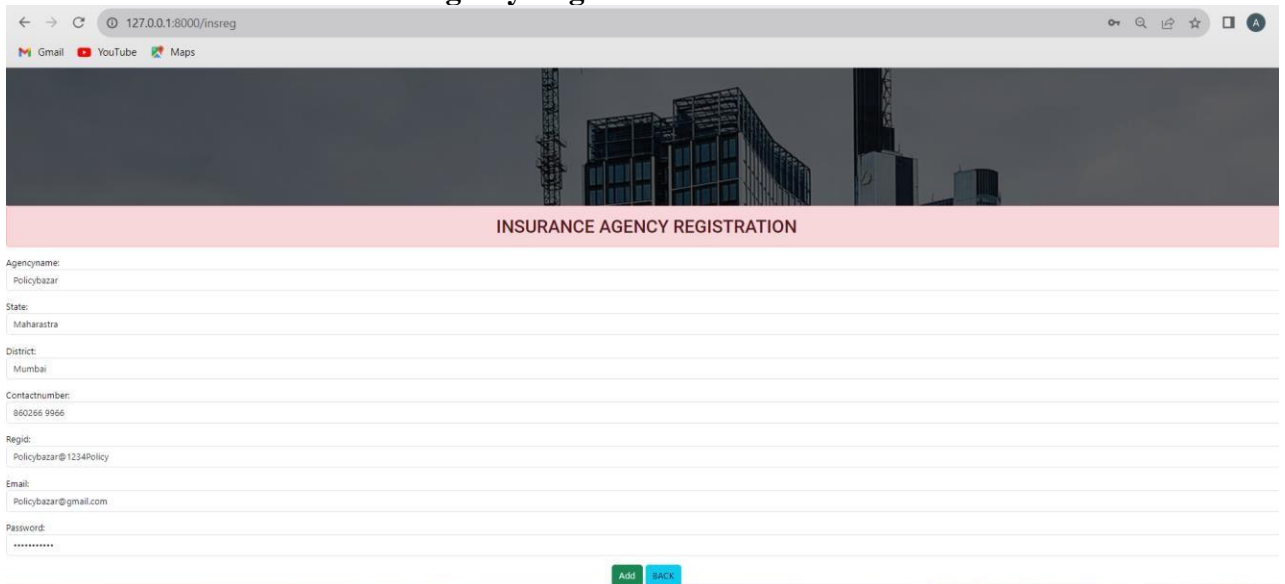
Contactnumber:
08588801181

Email:
Bestjob@gmail.com

Password:



Agency Registration



INSURANCE AGENCY REGISTRATION

Agencyname:
Policybazar

State:
Maharashtra

District:
Mumbai

Contactnumber:
860266 9966

Regid:
Policybazar@1234Policy

Email:
Policybazar@gmail.com

Password:

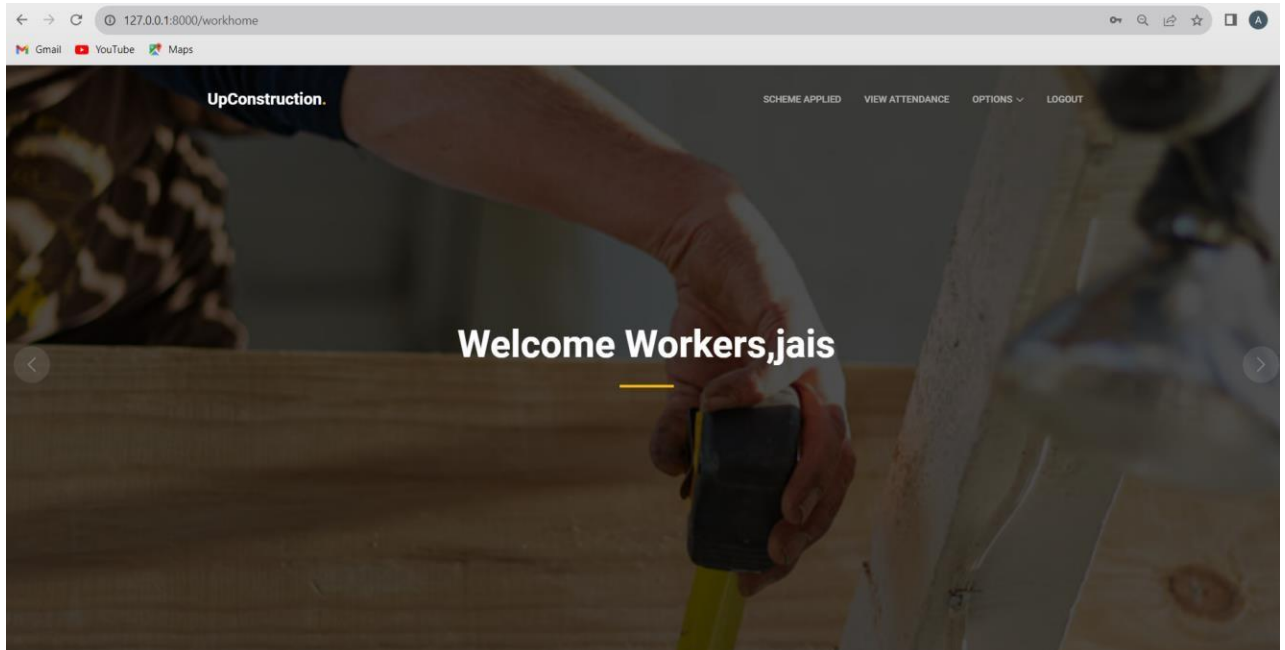
Add
PRCY

Insurance Agency Registration

Worker profile update

Worker Update	
<input type="text" value="9f"/>	<input type="text" value="Workname"/>
<input type="text" value="d5a8cccccccc"/>	<input type="text" value="Address"/>
<input type="text" value="44444444"/>	<input type="text" value="Pincode"/>
<input type="text" value="dcd"/>	<input type="text" value="State"/>
<input type="text" value="cdcd"/>	<input type="text" value="District"/>
<input type="text" value="cdcdcd"/>	<input type="text" value="City"/>
<input type="text" value="44444555555555"/>	<input type="text" value="Aadhar number"/>
<input type="text" value="33441414"/>	<input type="text" value="Contact number"/>
<input type="text" value="1234@gmail.com"/>	<input type="text" value="Email"/>
	<input type="text" value="Password"/>
<input type="button" value="UPDATE"/>	

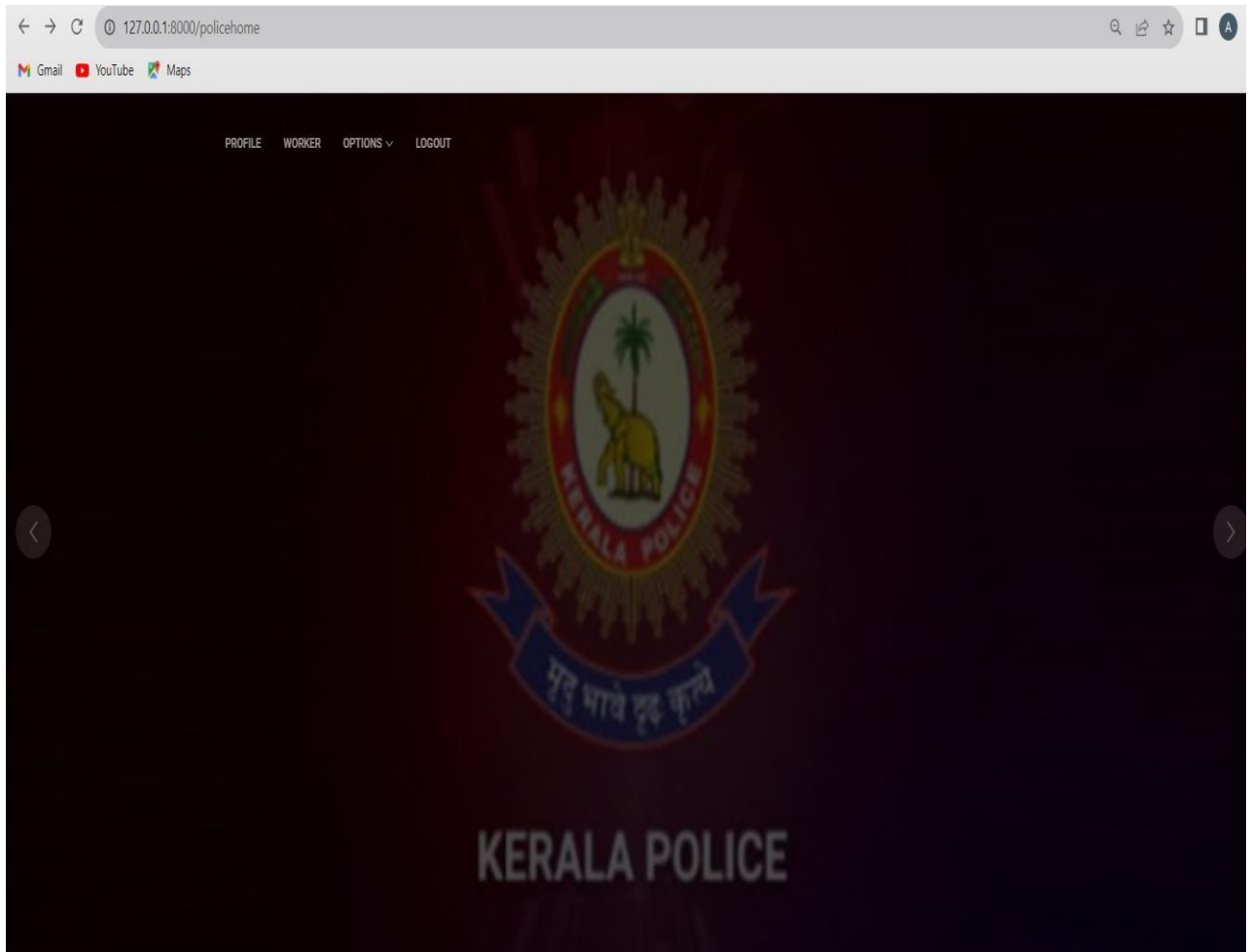
Worker Home Page



INSURANCE AGENCY HOME



Police Home Page



BIBLIOGRAPHY

1. Beginning Visual C# - Watson, White, Wrox Publications.
2. Mastering Django: Core" by Nigel George
3. Software Engineering-Roger S Pressman.
4. System Analysis and Design- James A. Senn.

Websites

1. <https://chat.openai.com/>
2. <https://www.javatpoint.com/python-tutorial>
3. <https://www.tutorialspoint.com/python/index.html>
4. <https://api.jquery.com>
5. <https://www.djangoproject.com>
6. <https://www.tutorialspoint.com/sqlite3/index.html>