

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.

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. The existing system may struggle to handle a growing workforce efficiently. The objective of the online application “MWTS” 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.

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 and interpretation, detailed examination. System development revolves around a lifecycle that begins 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

DEPARTMENT OF MCA

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.

DEPARTMENT OF MCA

- 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

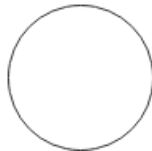
There are four symbols that are used in the drawing of Data Flow Diagrams:

- **Entities**



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

- **Process**



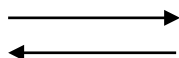
Processes represent activities in which data is manipulated by being stored or retrieved or transformed in some way. A circle represents it. The process will show the data transformation or change.

- **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

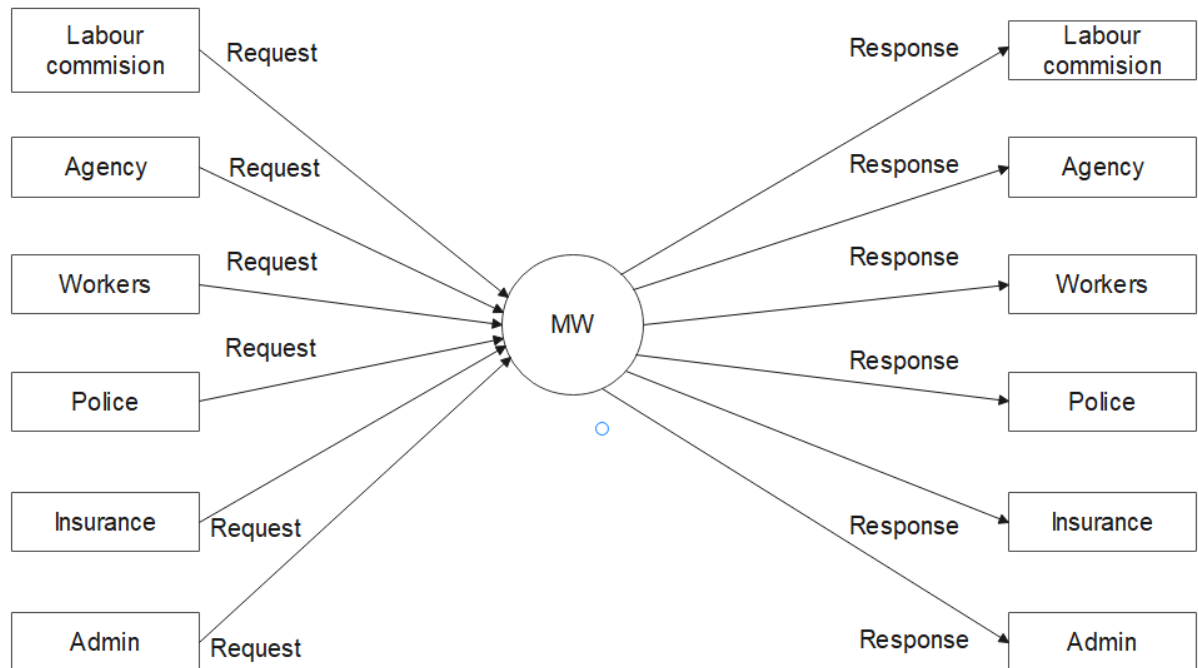


Figure 4.1 Context Level Diagram

4.2 Data Flow Diagram

4.2.1 Level 1 Admin

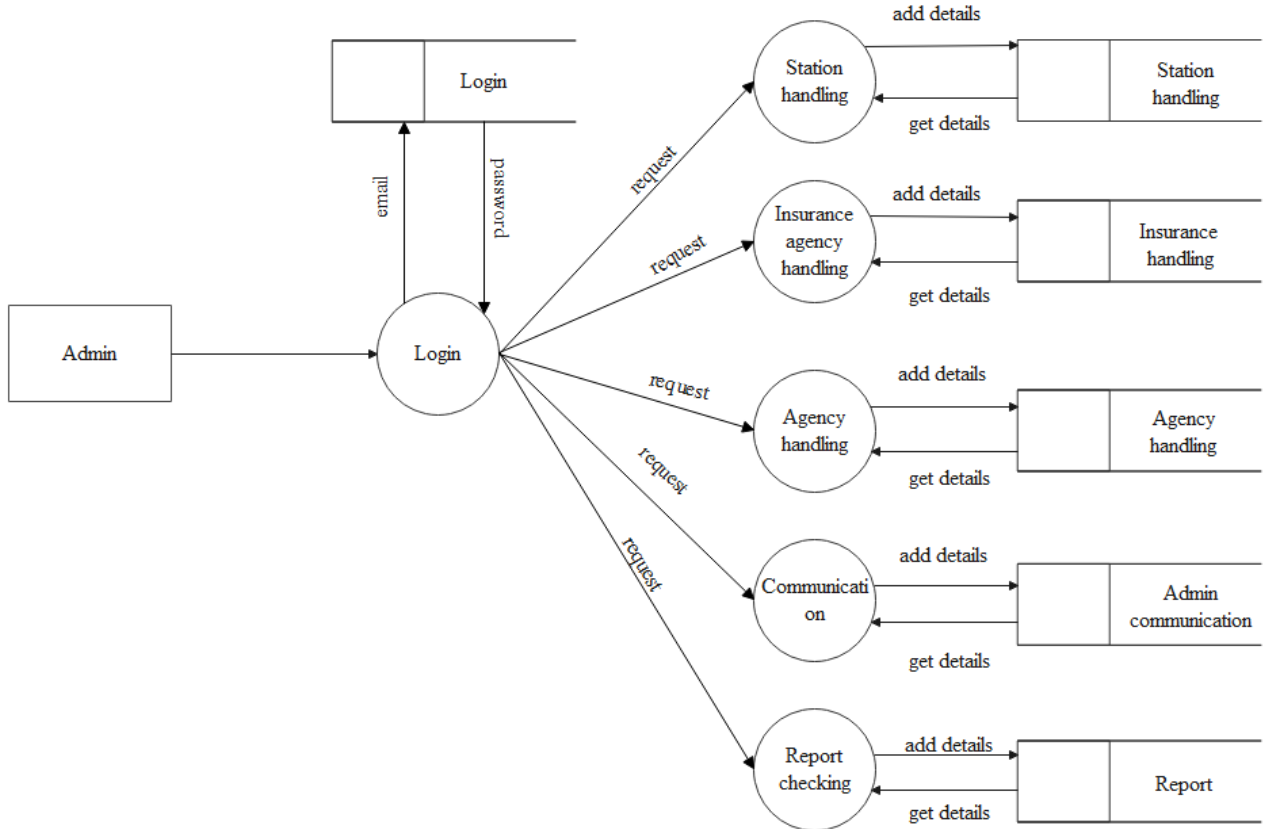


Figure 4.2.1 Admin (Level 1)

4.2.2 Level 1 Insurance agency

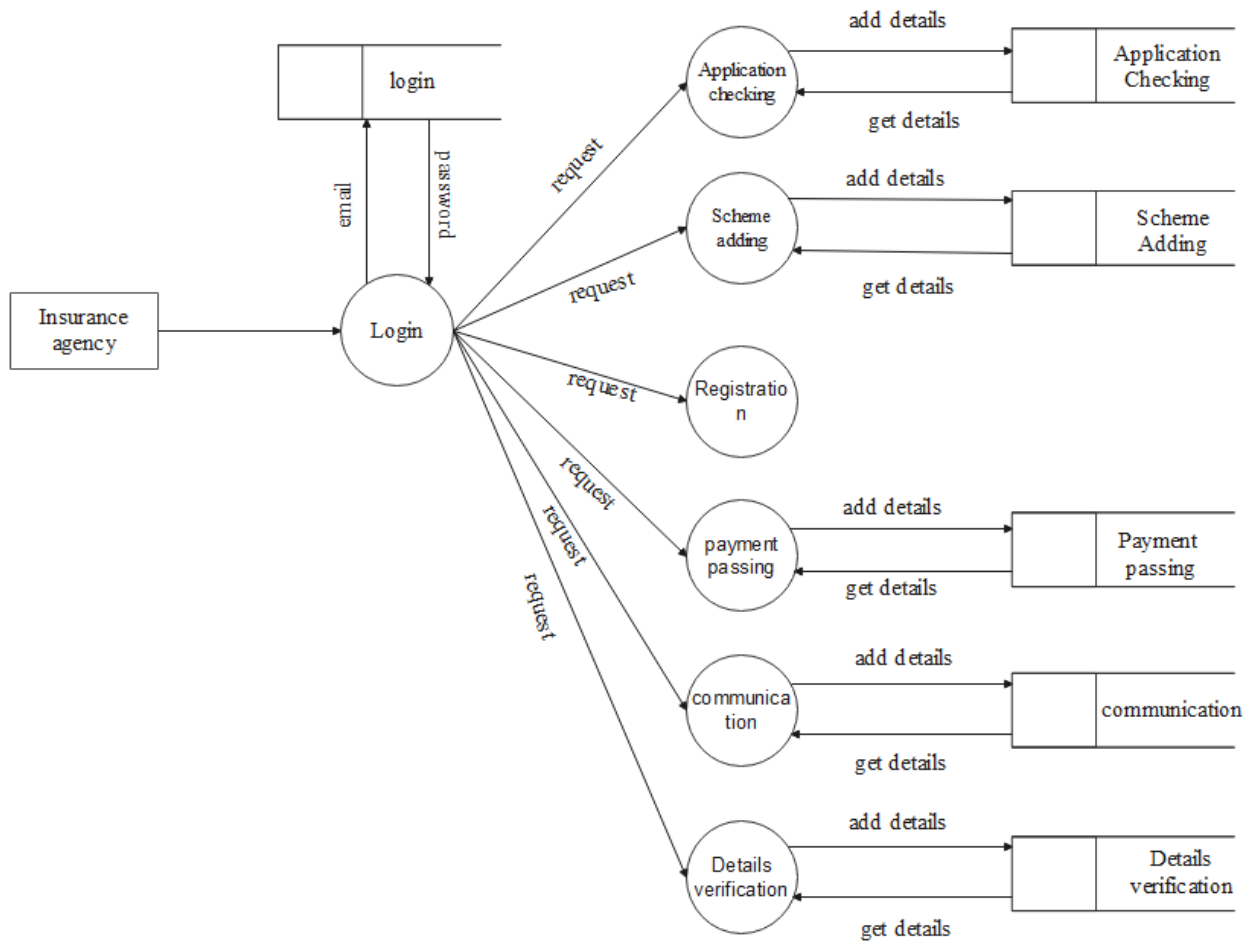


Figure 4.2.2 Insurance agency (Level 1)

4.2.3 Level 1 Worker

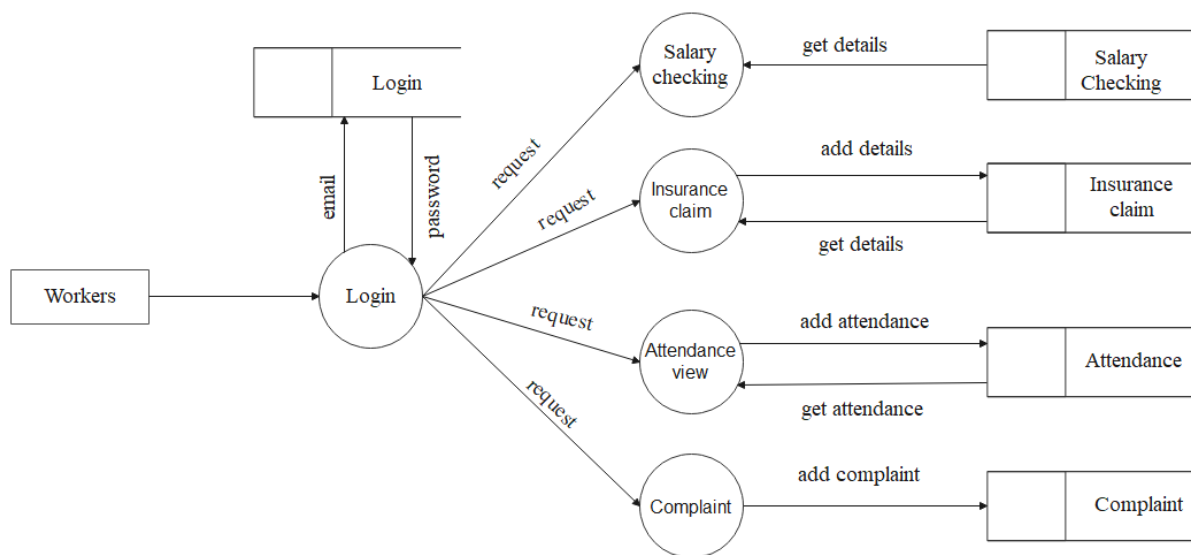


Figure 4.2.3 Worker (Level 1)

4.2.4 Level 1 Police

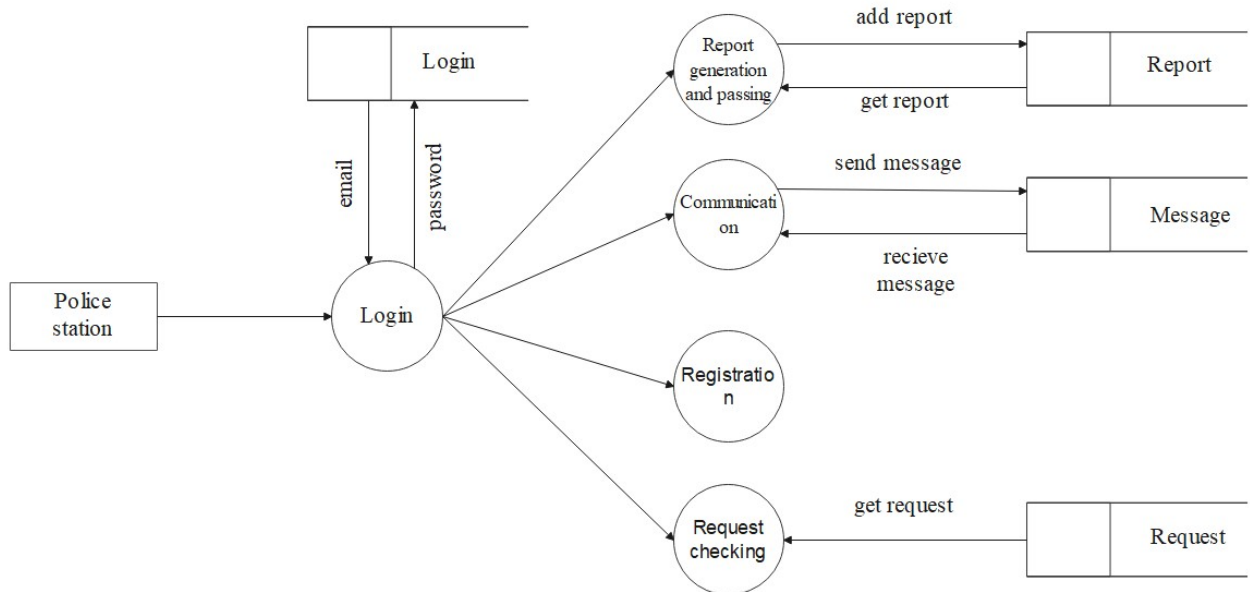


Figure 4.2.4 Police (Level 1)

4.2.5 Level 1 Labour commission

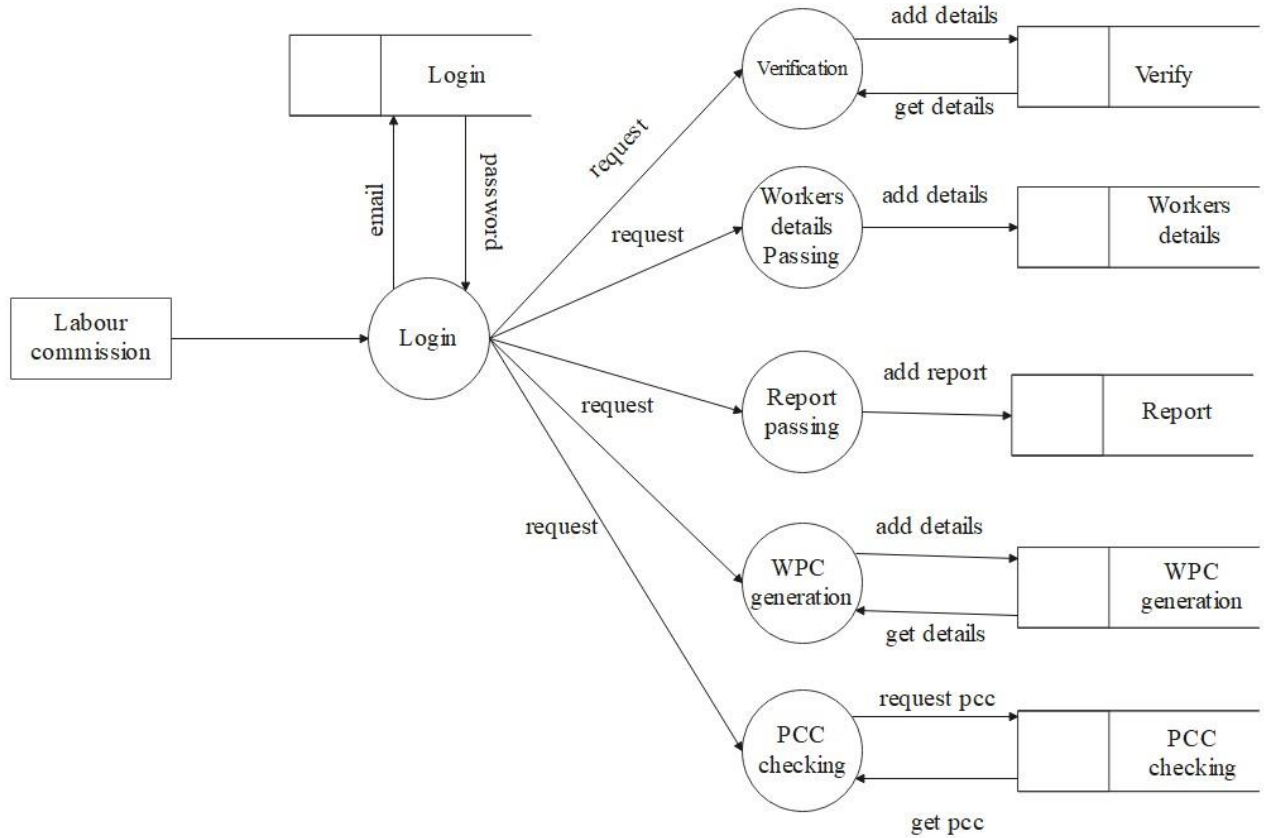


Figure 4.2.5 Labour commission (Level 1)

4.2.6 Level 1 Agency

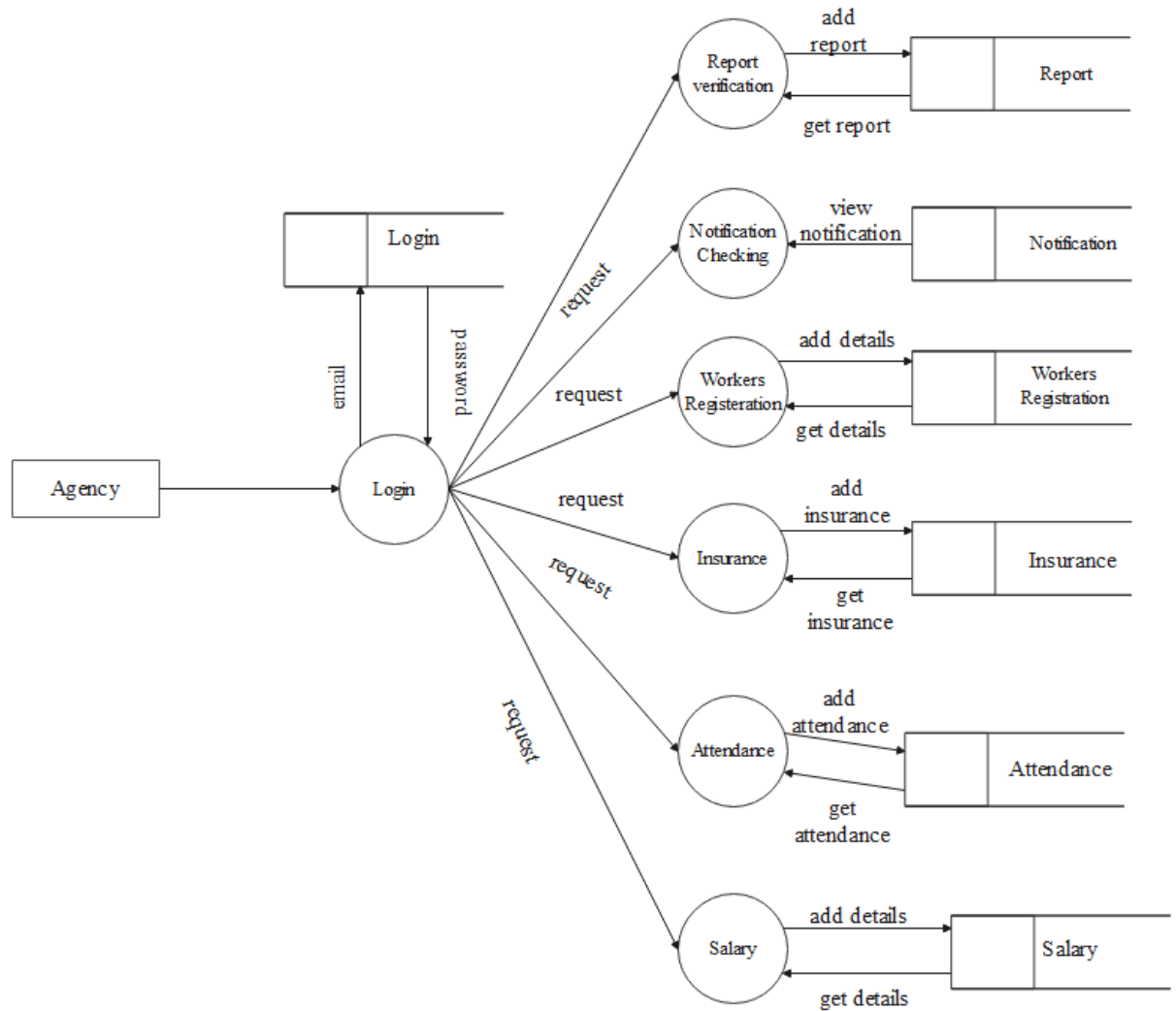


Figure 4.2.6 Agency (Level 1)

4.2.7 Level 2 Admin

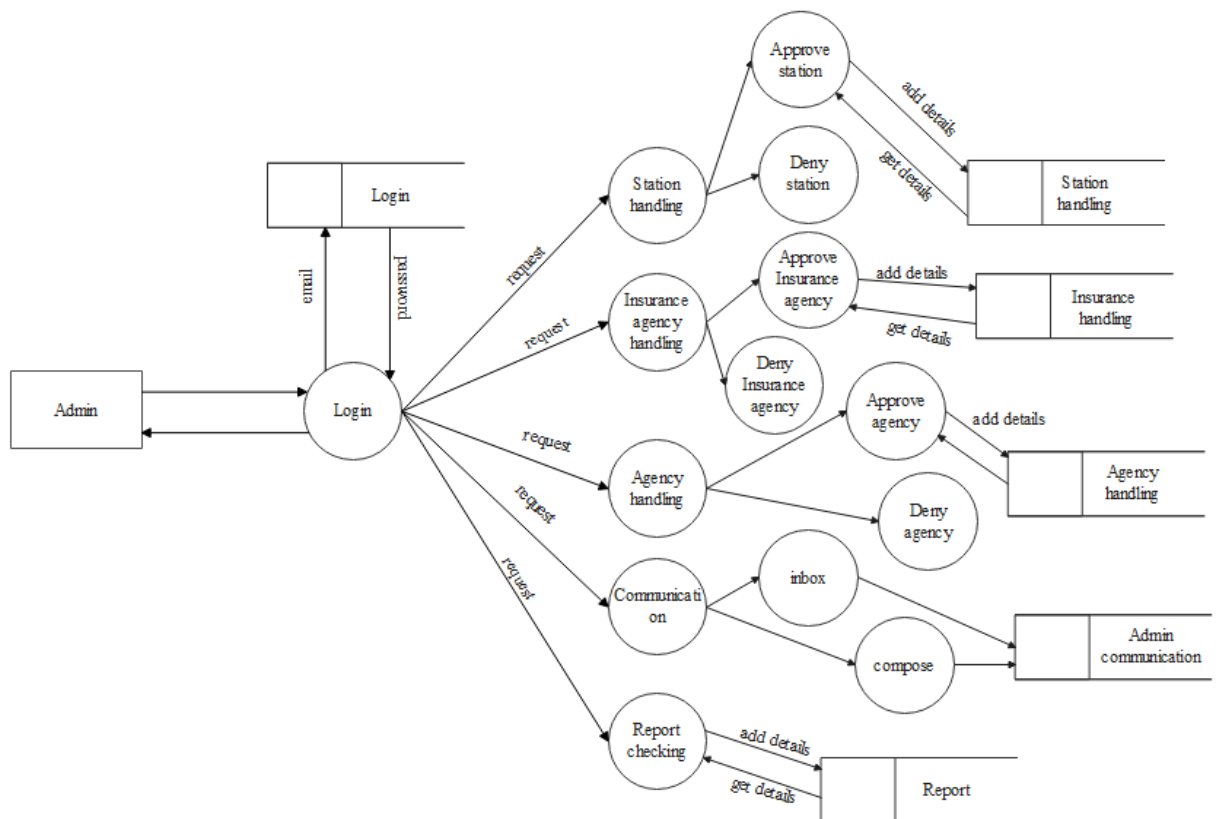


Figure 4.2.7 Admin (level 2)

4.2.8 Level 2 Labour commission

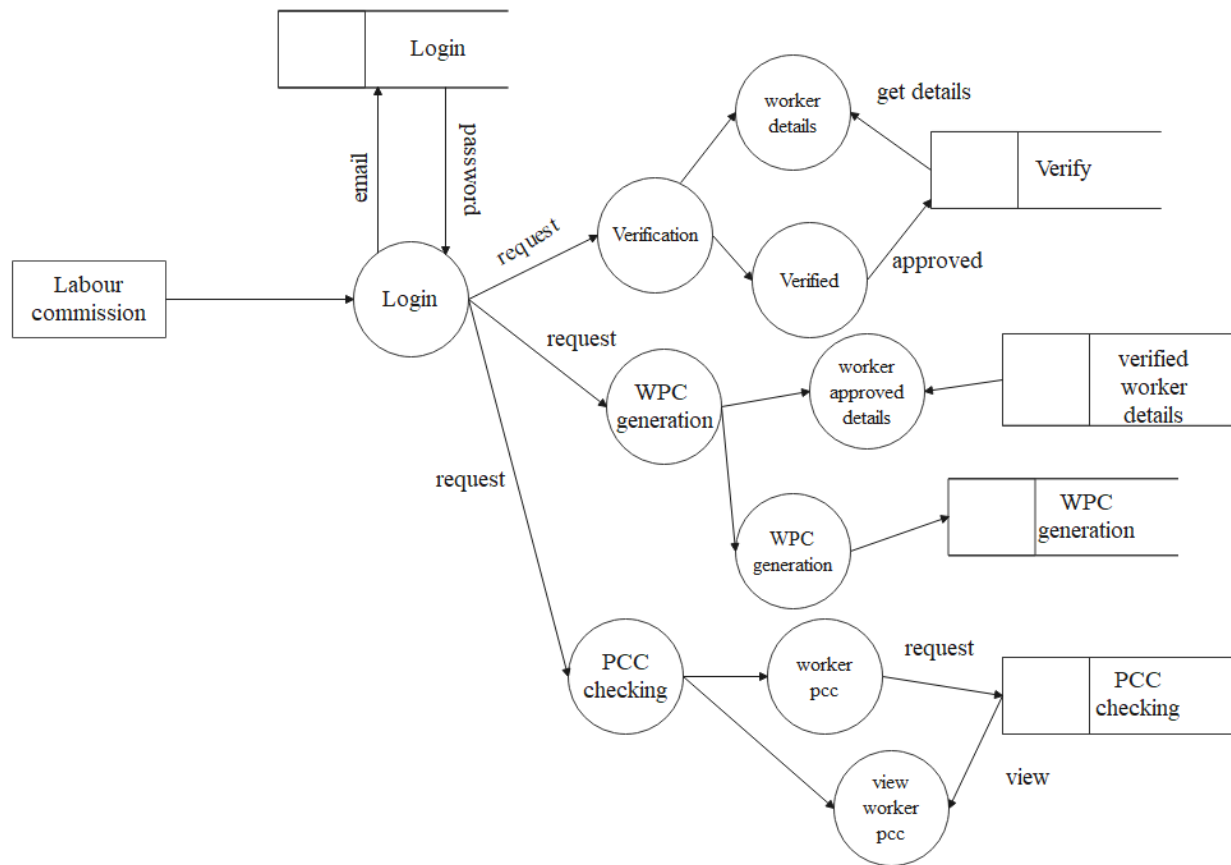


Figure 4.2.8 Labour commission (level 2)

4.3 ER Diagram

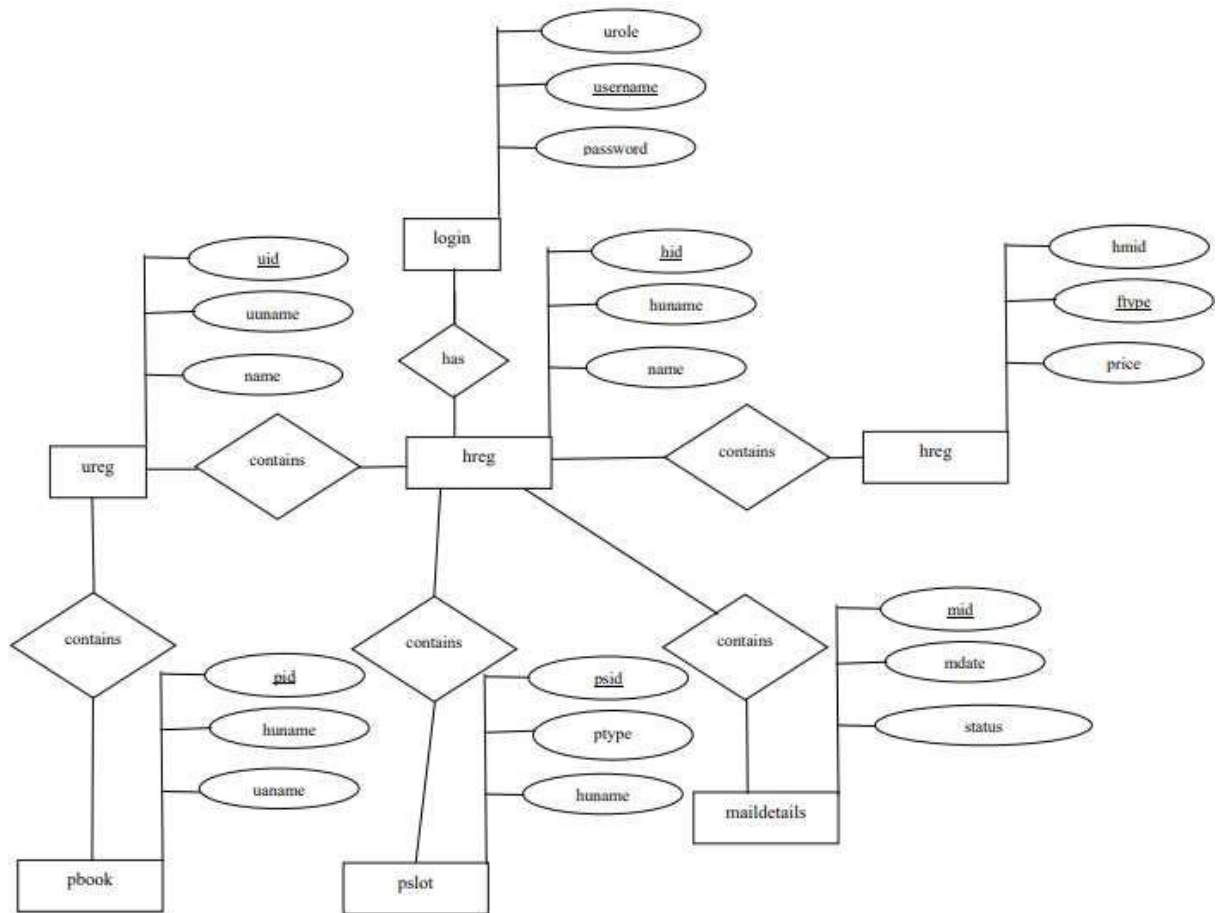


Figure 4.3.1 ER Diagram

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	255	NOT NULL	agency name
address	varchar	255	NOT NULL	agency address
pincode	int	3	NOT NULL	agency pincode
district	varchar	255	NOT NULL	agency district
city	varchar	255	NOT NULL	agency city
agencyid	varchar	255	NOT NULL	agency id
contactnumber	int	3	NOT NULL	agency contact number
email	varchar	255	NOT NULL	agency email
password	varchar	255	NOT NULL	agency password

DEPARTMENT OF MCA

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	255	NOT NULL	worker name
address	varchar	255	NOT NULL	worker address
pincode	int	3	NOT NULL	worker pincode
state	varchar	255	NOT NULL	worker state
district	varchar	255	NOT NULL	worker district
city	varchar	255	NOT NULL	worker city
aadharnumber	int	3	NOT NULL	worker aadharnumber
contactnumber	int	3	NOT NULL	worker contact number
email	varchar	255	NOT NULL	worker email
password	varchar	255	NOT NULL	worker password

DEPARTMENT OF MCA

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	255	NOT NULL	Station id
addressline1	varchar	255	NOT NULL	Police station addressline1
Addressline2	varchar	255	NOT NULL	Police station addressline2
pincode	int	3	NOT NULL	Police station pincode
state	varchar	255	NOT NULL	Police station state
district	varchar	255	NOT NULL	Police station district
city	varchar	255	NOT NULL	Police station city
contactnumber	int	3	NOT NULL	Police station contact number
email	varchar	255	NOT NULL	Police station email
password	varchar	255	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	255	NOT NULL	Insurance agency name
state	varchar	255	NOT NULL	Insurance agency state
district	varchar	255	NOT NULL	Insurance agency district
contactnumber	int	3	NOT NULL	Insurance agency contact number
regid	varchar	255	NOT NULL	Insurance agency regid
email	varchar	255	NOT NULL	Insurance agency email
password	varchar	255	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	255	NOT NULL	scheme_type
scheme_name	varchar	255	NOT NULL	scheme_name
scheme_amount	varchar	255	NOT NULL	scheme_amount
monthly_amount	varchar	255	NOT NULL	monthly_amount
interest_rate	varchar	255	NOT NULL	interest_rate
scheme_details	varchar	255	NOT NULL	scheme_details

DEPARTMENT OF MCA

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

DEPARTMENT OF MCA

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	Worker id
police	int	4	Foreign Key	Police id
currentdate	date	20	NOT NULL	Current date
file	file	50	NOT NULL	Report file

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

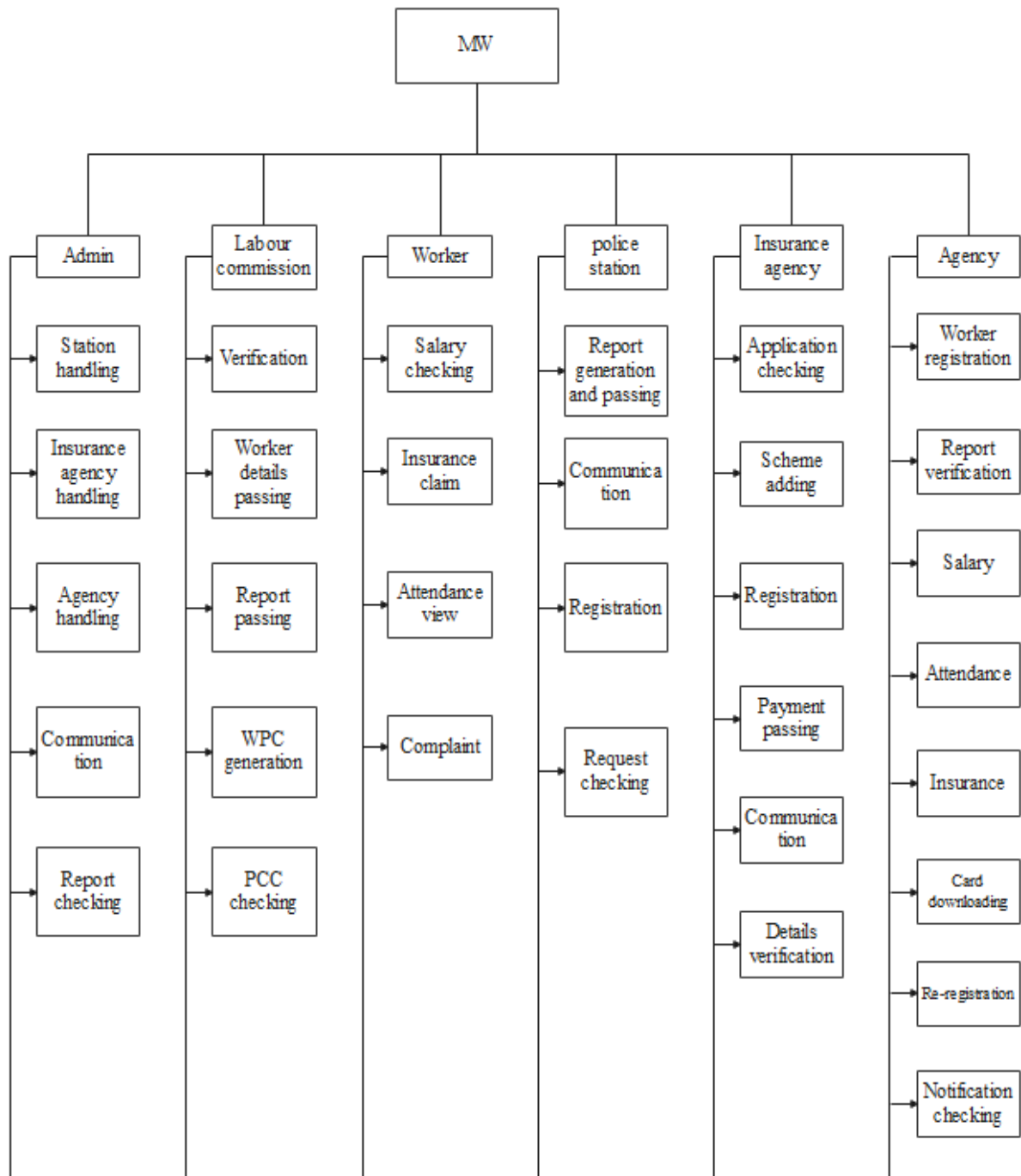
- Remove that depends on other hand in the table or relationship and not on the key.
- From the removed data, create one or more tables and relationships.

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



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.

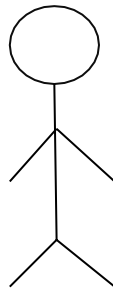
1. To identify functions and how roles interact with them.
2. For a high-level view of the system.
3. To identify internal and external actors.

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

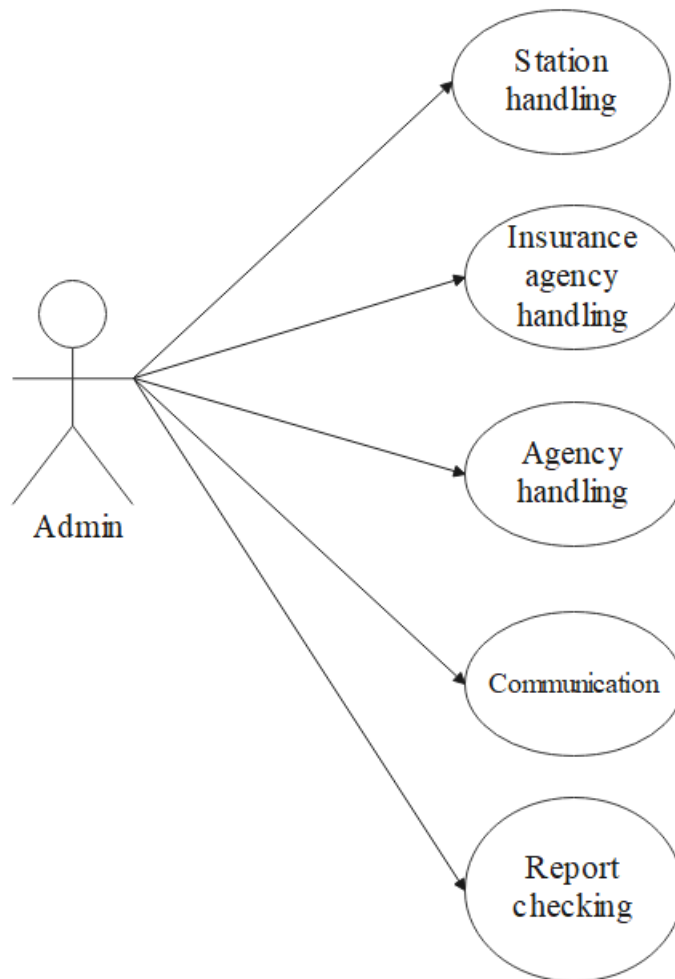


Figure 4.7.1 .1 Use Case Diagram for Admin

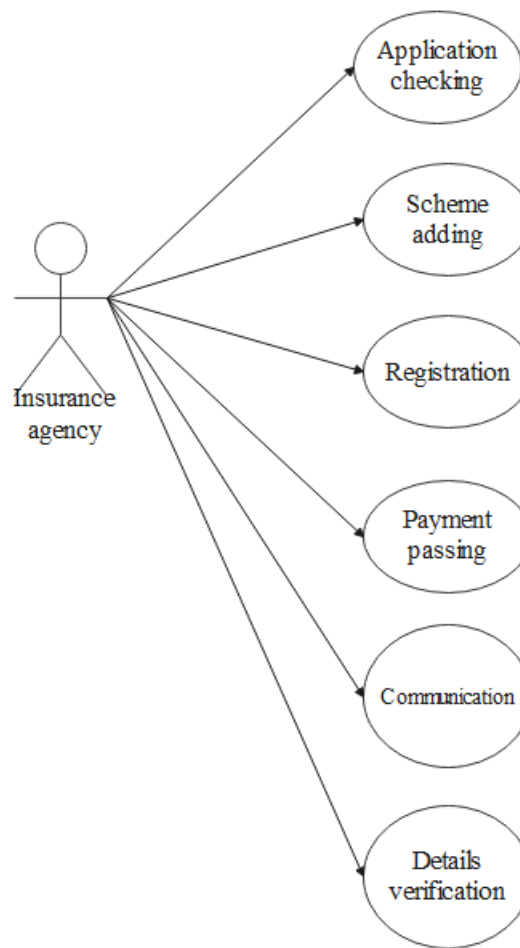


Figure 4.7.1 .2 Use Case Diagram for Insurance agency

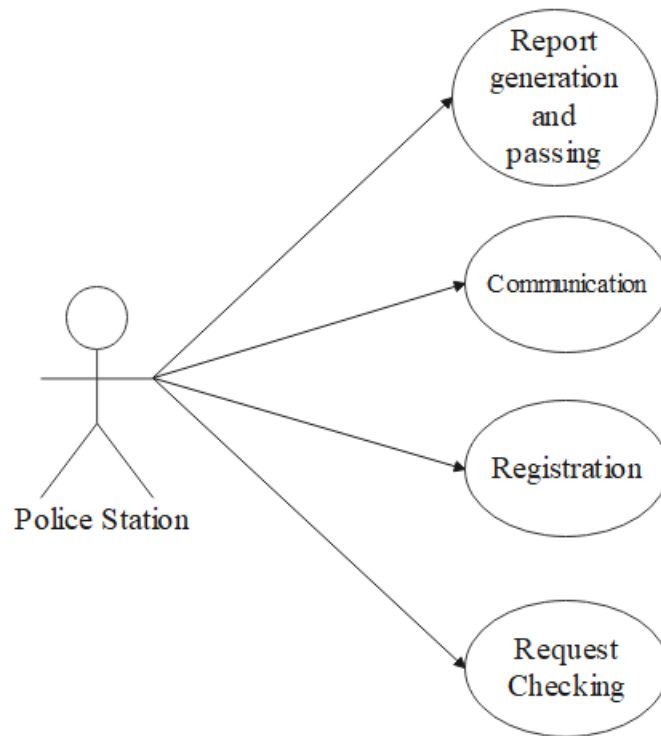


Figure 4.7.1 .3 Use Case Diagram for police station

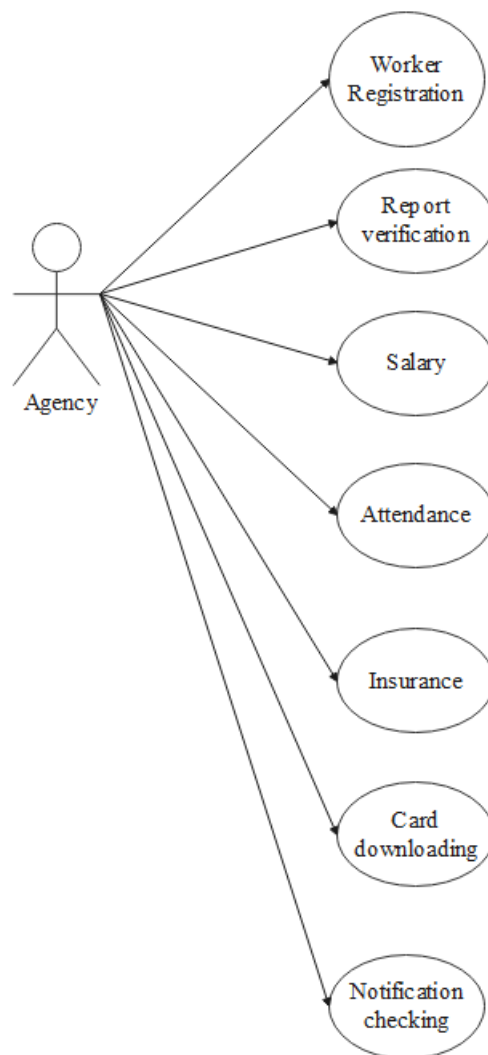


Figure 4.7.1 .4 Use Case Diagram for agency

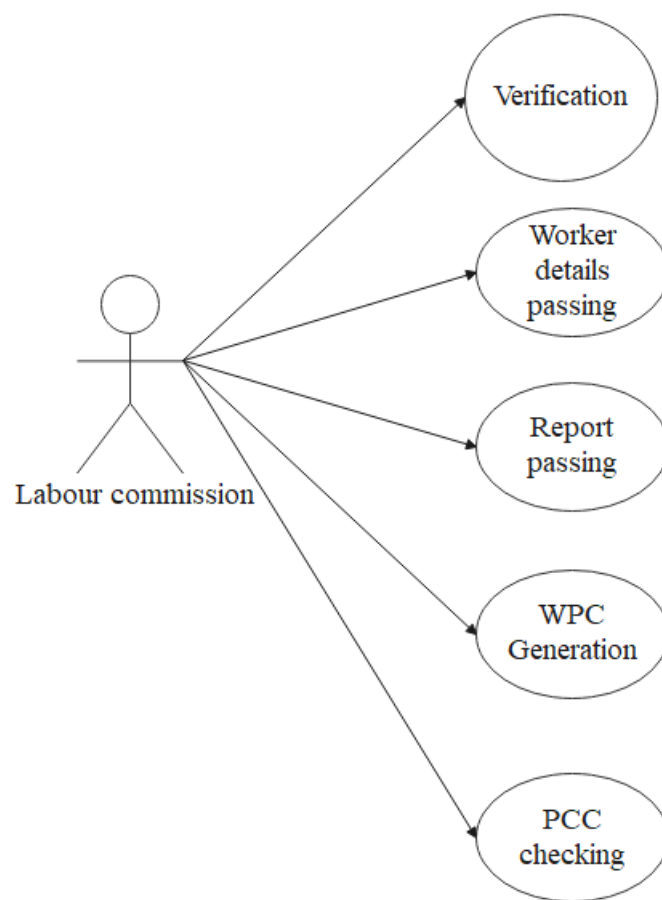


Figure 4.7.1.5 Use Case Diagram for Labour commission

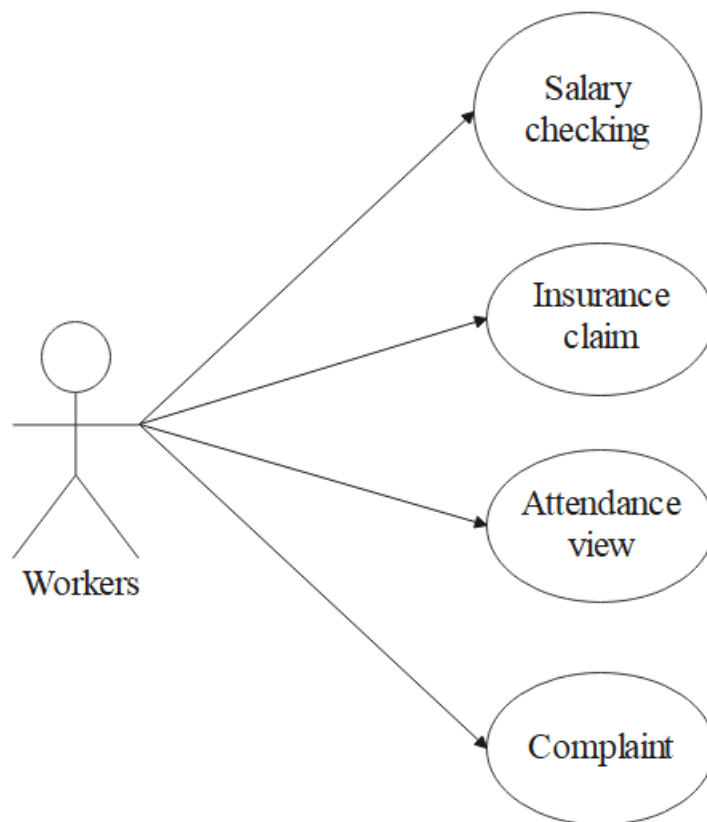


Figure 4.7.1.6 Use Case Diagram for workers

4.7.2 Class Diagram

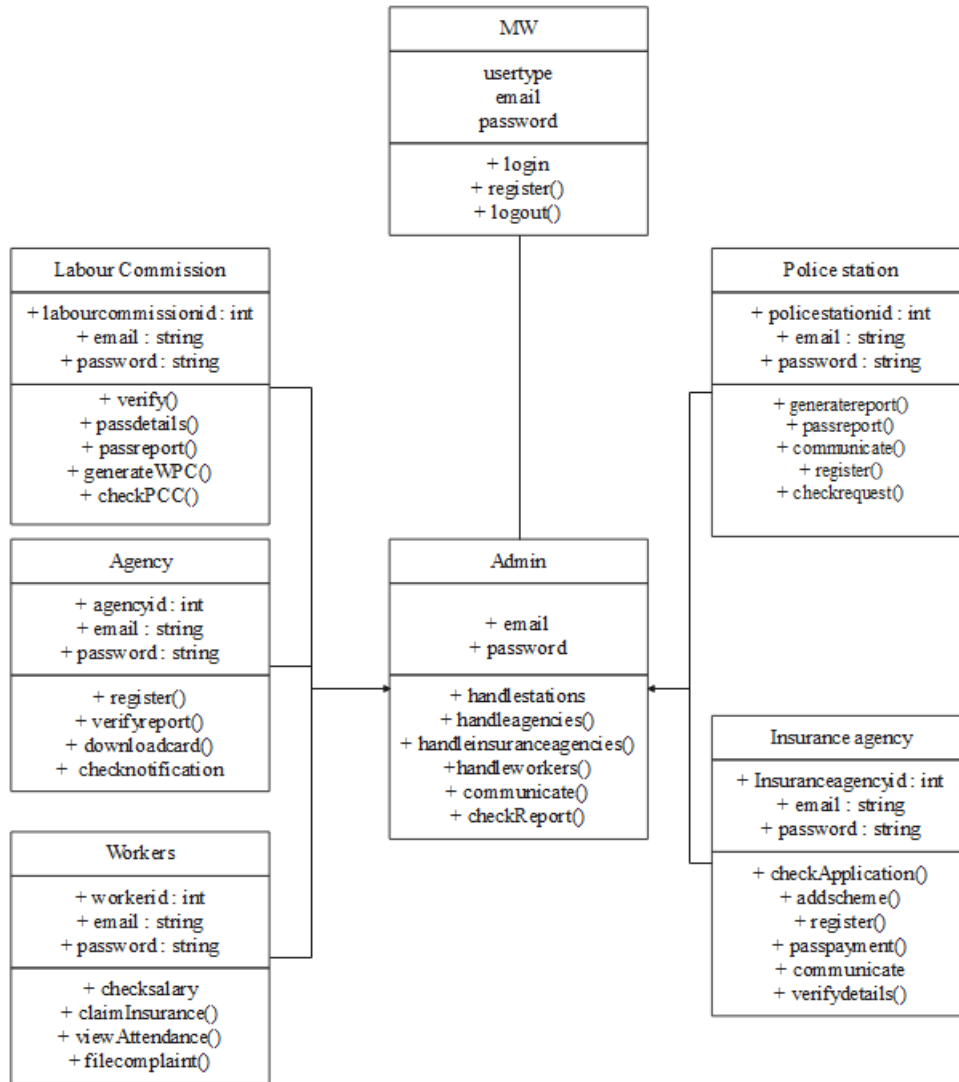


Figure 4.7.2.1 Class Diagram

4.7.3 Sequence Diagram

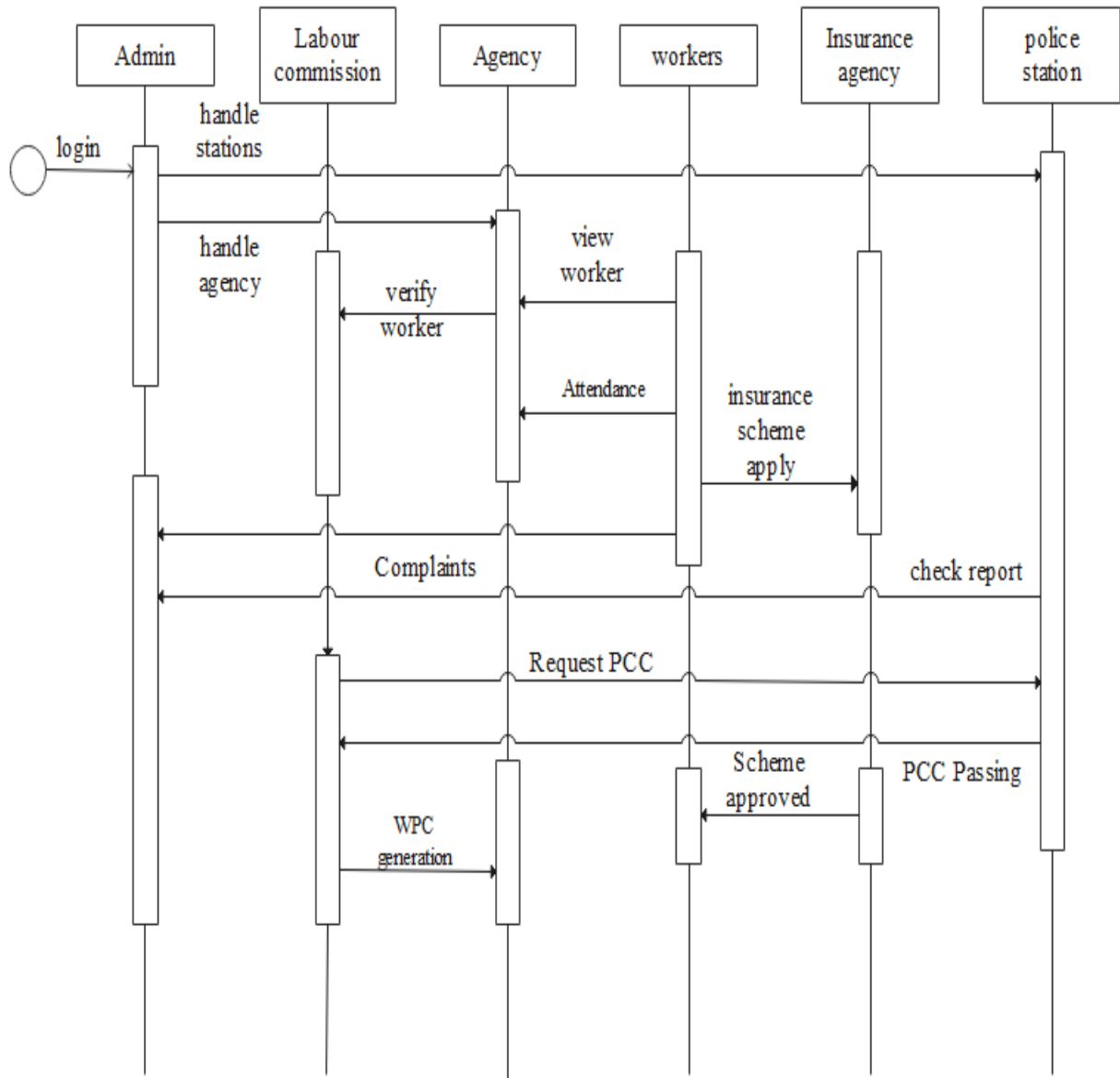


Figure 4.7.3.1 Sequence Diagram

CHAPTER – 5

CODING

5.1 Language Study

.NET Framework

The .NET Framework is the infrastructure for the Microsoft .NET platform. The .NET Framework is an environment for building, deploying, and running Web applications and Web Services. Microsoft's first server technology ASP (Active Server Pages), was a powerful and flexible "programming language". But it was too code oriented. It was not an application framework and not an enterprise development tool.

The Microsoft .NET Framework is a software framework available with several Microsoft Windows operating systems. It includes a large library of coded solutions to prevent common programming problems and a virtual machine that manages the execution of programs written specifically for the framework. The .NET Framework is a key Microsoft offering and is intended to be used by most new applications created for the Windows platform.

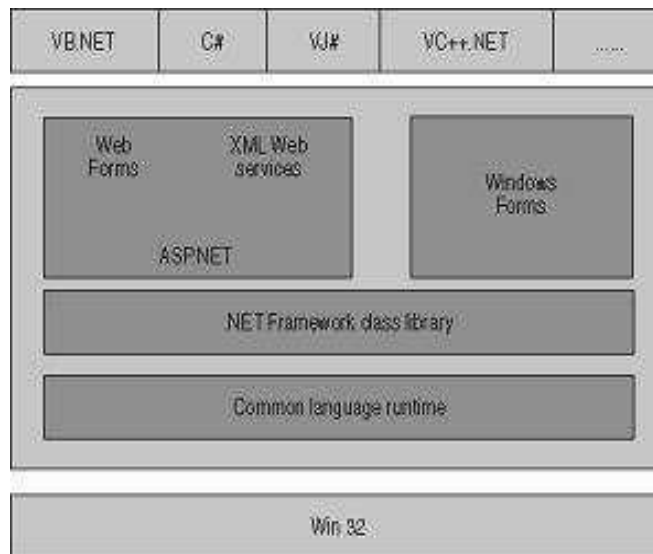


Figure 5.1 **.Net Architecture**

This proposed system requires an environment called the .NET Framework, which is an integral Windows component for building and running the next generation of software applications and Web services. The .NET Framework supports over 20 different programming languages, makes it easier than ever before to build, deploy, and administer secure, robust, and high-performing applications. The .NET Framework is composed of the common language runtime and a unified set of class libraries. Base classes provide standard functionality such as input/output, string manipulation, security management, network communications; thread management, text management, and user interface design features. The ADO.NET classes enable developers to interact with data accessed in the form of XML through the OLE DB, ODBC, Oracle, and SQL Server interfaces. XML classes enable XML manipulation, searching, and translations. The ASP.NET classes support the development of Web-based applications and Web services. The Windows Forms classes support the development of desktop-based smart client applications.

The .NET framework has two main components:

1. The Common Language Runtime
2. A hierarchical set of Class Libraries

5.1.1 Common Language Runtime (CLR)

CLR is described as the “execution engine” of .NET. It provides the environment within which programs run. The most important features are:

1. Conversion from a low-level assembler-style language called Intermediate Language (IL), into code native to the platform being executed on
2. Memory Management, notably including garbage collection.
3. Checking and enforcing security restrictions on the running code.
4. Loading and executing programs with version control and other such features.

5.1.2 Class Libraries

Class Libraries, the other main component of the .NET framework, is a comprehensive, object-oriented collection of reusable type that you can use to develop applications

ranging from traditional command-line or Graphical User Interface (GUI) applications to applications based on the latest innovations provided by ASP.NET such as Web Forms and XML Web Services.

The .Net framework can be hosted by unmanaged components that load the common language runtime into their processes and initiate the execution of managed code, thereby creating a software environment that can exploit both managed and unmanaged features. The .Net framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts. We can use the .net framework to develop the following types of applications and services.

1. Console applications.
2. Scripted or hosted applications
3. Windows GUI applications (Windows form)
4. ASP.NET applications
5. XML Web Services
6. Windows Services

5.2 Features of .NET

The .NET Framework provides a feature-rich application environment, simplified development and easy integration between number of different development languages. .NET a wonderful platform for developing modern application.

1. Rich functionality out of the box.

.NET framework provides a rich set of functionalities out of the box. It contains 100 of classes that provide variety of functionality ready to use in our application.

2. Easy development of web applications.

ASP.NET is a technology available on .NET platform for developing dynamic and data driven web application.

3. Oop's support

.NET provides a fully object-oriented environment. The philosophy of .NET is "object is mother of all."

4. Multi languages support

.NET supports multiple languages. Currently four languages are available right out of the box namely-visual basic.NET, c#.NET, j script.NET, managed C++.NET. The beauty of multi-language support lies in the fact even though the syntax of each language is different, the basic capabilities of each language remains at par with one another

5. Multi device support

.NET provides promising platform for programming such devices. .NET compact framework and mobile internet toolkit are step ahead in this direction

6. Automatic memory management

.NET takes the worry away from developers by handling memory on its own. The garbage collectors take care of freeing unused objects at appropriate intervals.

7. Strong XML support

.Net is the only platform that has built with XML into the core framework. .NET tries to harness power of XML in every possible way. In addition to providing support for manipulating and transforming XML documents, .NET provides XML web services that are based on standard like HTTP.XML and SOAP

8. Ease of deploying and configuration

Deploying windows application especially that used Com components was always been a tedious task. Since.NET does not require any registration as such; much of the

deployment is simplified. This makes XCOPY deployment viable. The configuration is done via special files having special XML, vocabulary.

9. Security

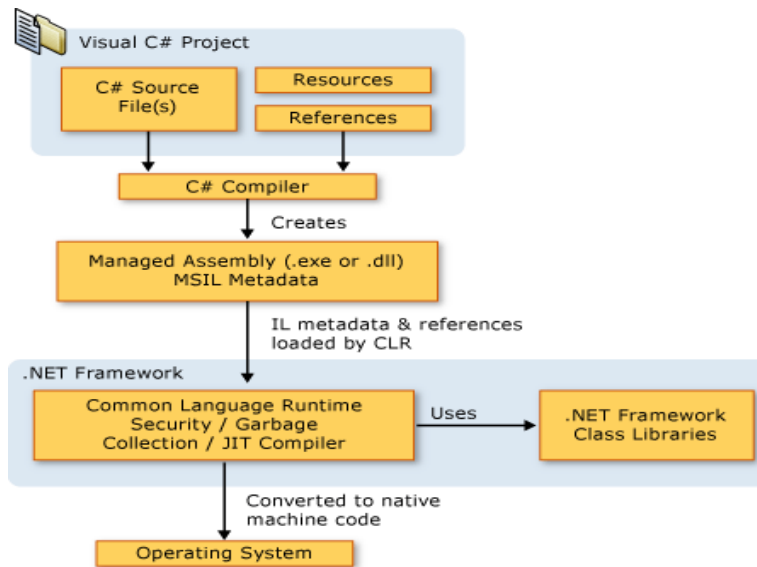
Windows platform was always criticized for poor security mechanisms. Microsoft has taken great efforts to make .NET platform safe and secure for enterprise applications. Features such as safety, code access security and role based authentication make overall application more robust and secure.

C#.NET

C# programs run on the .NET Framework, an integral component of Windows that includes a virtual execution system called the common language runtime (CLR) and a unified set of class libraries. The CLR is the commercial implementation by Microsoft of the common language infrastructure (CLI), an international standard that is the basis for creating execution and development environments in which languages and libraries work together seamlessly.

Source code written in C# is compiled into an intermediate language (IL) that conforms to the CLI specification. The IL code and resources, such as bitmaps and strings, are stored on disk in an executable file called an assembly, typically with an extension of .exe or .dll. An assembly contains a manifest that provides information about the assembly's types, version, culture, and security requirements.

When the C# program is executed, the assembly is loaded into the CLR, which might take various actions based on the information in the manifest. Then, if the security requirements are met, the CLR performs just in time (JIT) compilation to convert the IL code to native machine instructions. The CLR also provides other services related to automatic garbage collection, exception handling, and resource management. Code that is executed by the CLR is sometimes referred to as "managed code," in contrast to "unmanaged code" which is compiled into native machine language that targets a specific system. The following diagram illustrates the compile-time and run-time relationships of C# source code files, the .NET Framework class libraries, assemblies, and the CLR.



Language interoperability is a key feature of the .NET Framework. Because the IL code produced by the C# compiler conforms to the Common Type Specification (CTS), IL code generated from C# can interact with code that was generated from the .NET versions of Visual Basic, Visual C++, or any of more than 20 other CTS-compliant languages. A single assembly may contain multiple modules written in different .NET languages, and the types can reference each other just as if they were written in the same language.

Microsoft SQL Server 2008

Organizations today are facing several unique data challenges: the proliferation of data and systems across their enterprise; the need to provide employees, customers, and partners with consistent access to that data; the desire to better equip information workers with meaningful information to drive informed decisions; and the mandate to control costs without sacrificing application availability, security, or reliability. The next release of SQL Server is designed to help enterprises address these challenges. SQL Server 2008 is Microsoft's next generation data management and analysis solution that will deliver increased security, scalability, and availability to enterprise data and analytical applications while making them easier to create, deploy, and manage. Building on the

strengths of SQL Server 2000, SQL Server 2008 will provide an integrated data management and analysis solution that will help organizations of any size to:

1. Build and deploy enterprise applications that are more secure, scalable, and reliable.
2. Maximize the productivity of IT by reducing the complexity of creating, deploying, and managing database applications.
3. Empower developers through a rich, flexible, modern development environment for creating more secure database applications.
4. Share data across multiple platforms, applications, and devices to make it easier to connect internal and external systems.
5. Deliver robust, integrated business intelligence solutions that help drive informed business decisions and increase productivity across your entire organization.
6. Control costs without sacrificing performance, availability, or scalability.

To learn more about the advancements SQL Server 2008 will deliver in three key areas: enterprise data management, developer productivity, and business intelligence. In today's connected world, data and the systems that manage that data must always be available to your users. With SQL Server 2008, users and IT professionals across your organization will benefit from reduced application downtime, increased scalability and performance, and tight security controls.

SQL Server 2008 will include enhancements to enterprise data management in the following areas:

1. Availability:

Investments in high availability technologies, additional backup and restore capabilities, and replication enhancements will enable enterprises to build and deploy highly reliable applications.

2. Scalability:

Scalability advancements such as partitioning, snapshot isolation, and 64-bit support will enable you to build and deploy your most demanding applications using SQL Server 2008.

3. Security:

Enhancements such as “secure by default” settings and an enhanced security model will help provide a high level of security for your enterprise data.

4. Manageability:

A new management tool suite, expanded self-tuning capabilities, and a powerful new programming model will increase the productivity of database administrators.

5. XML and Web services:

SQL Server 2008 will support both relational and XML data natively, so enterprises can store, manage, and analyze data in the format that best suits their needs. Support for existing and emerging open standards such as Hypertext Transfer Protocol (HTTP), XML, Simple Object Access Protocol (SOAP), XQuery, and XML Schema Definition (XSD) will also facilitate communication across extended enterprise systems.

Business Intelligence

The challenge and promise of business intelligence revolves around providing employees with the right information, at the right time. Accomplishing this vision demands a business intelligence solution that is comprehensive, secure, integrated with operational systems, and available all day, every day. SQL Server will help companies to achieve this goal with SQL Server 2008. Business intelligence advancements will include:

1. Integrated platform:

SQL Server 2008 will deliver an end-to-end business intelligence platform with integrated analytics including online analytical processing (OLAP); data mining; extract, transformation, and load (ETL) tools; data warehousing; and reporting functionality. This document is developed prior to the product's release to manufacturing, and as such, we cannot guarantee that all details included herein will be exactly as what is found in the shipping product. The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication. The information represents the product at the time this document was printed and should be used for planning purposes.

only. Information subject to change at any time without prior notice. This document is for informational purposes only.

2. Improved decision making:

Advancements to existing business intelligence features, such as OLAP and data mining, and the introduction of a new reporting server will provide enterprises with the ability to transform information into better business decisions at all organizational levels.

3. Security and availability:

Scalability, availability, and security enhancements will help to provide users with uninterrupted access to business intelligence applications and reports.

4. Enterprise-wide analytical capabilities:

An improved ETL tool will enable organizations to more easily integrate and analyze data from multiple heterogeneous information sources. By analyzing data across a wide array of operational systems, organizations may gain a competitive edge through a holistic understanding of their business.

5. Interoperability:

Through deep support for industry standards, Web services, and the Microsoft .NET Framework, SQL Server 2008 will support interoperability with multiple platforms, applications, and devices.

Developer Productivity One of the key barriers to developer productivity has been the lack of integrated tools for database development and debugging. SQL Server 2008 will provide advancements that fundamentally change the way that database applications are developed and deployed. Enhancements for developer productivity will include:

6. Improved tools:

Developers will be able to utilize one development tool for Transact-SQL, XML, Multidimensional Expression (MDX), and XML for Analysis (XML/A). Integration with the Visual Studio® development environment will provide more efficient development and debugging of line-of-business and business intelligence applications.

7. Expanded language support:

With the common language runtime (CLR) hosted in the database engine, developers will be able to choose from a variety of familiar languages to develop database applications, including Transact-SQL, Microsoft Visual Basic .NET, and Microsoft Visual C# .NET.

CODE EFFICIENCY

The algorithm is then put through three tests to evaluate its effectiveness. The localization test demonstrates the algorithm's ability to detect faces and localize facial features in several, arbitrary test images. A recognition test is then performed on a standard database of faces to determine the recognition rates of the algorithm. In addition, we present a sensitivity test which relates recognition effectiveness to localization accuracy.

- Localization Test
- Recognition Test
- Sensitivity Analysis

Performance and Code Efficiency

In an effort to maintain computational efficiency and to allow the eventual adaptation of the algorithm to face tracking applications, intense optimization of the code has been performed. Although further development is in progress, the algorithm is currently fast and compact enough to run interactively on most generic platforms. Note first, the sequential hierarchical search which proceeds from large scales to small scales. This allows a rapid convergence if the face is dominant in the image. Furthermore, the algorithm does not always flow through the computer loop. It stops as soon as one of the modules reports a failure and loops back to an earlier stage. For example, we do not search for a mouth if no eyes are found. In this case, no time is wasted in the mouth module. The execution times are measured on an SGI Indy machine which has a rating equivalent to that of a 1996 home personal computer. The efficiency of the code allows a face to be found in an image in under 1 second if it is the dominant structure. However, we loop through all objects in the scene in an attempt to find all possible faces. Thus, the

algorithm's loop is traversed multiple times even though a face could have been detected in an earlier iteration of the loop.

5.3 Functional Description

Administrator Module

- The system allows the system administrator to: view the parking place, , managing the client and parking lot information(such as deleting, updating, adding viewing the client information and viewing different type of parking lot status)
- The system allows authentication of registered users and hotels
- Admin can delete and view restaurants and customers
- The report of customer ordering and reservation table.
- Suggestion or comment that customer insert at feedback form.

Hotel or Restaurant Module

- The system allows the restaurant to: book the parking place and and viewing different type of parking lot status
- Restaurant manages graphical room booking, booking amount calculation and appropriate billing.
- A restaurants can edit/create some or whole part of; the menu record on daily basis. That is by changing the menu items, prices, description, etc.
- Menu management.
- Food management.

Customer Module

- The system allows drivers to locate and reserve a parking place online through accessing it on web platform
- The system allows authentication of registered users.
- The web application enables employees to set the reaching date and time for the car also the departure date and time.

DEPARTMENT OF MCA

- The web application enables employees to cancel a parking place
- Customer online ordering and reservation module provides a form that needs to be fulfilling in term of ordering food and reservation table via online.
- Menu Booking: Booking of menu along with the table is the customer's choice that means the customer can only book the table or table with menu.
- Cancel Booking: The user can cancel the booking if the customer is not able to reach during the booked time slot.

Complaint module

- Based on food or everything about the restaurant, customer can send any suggestion or comment to the restaurant with feedback form. From this form, side of restaurant will know their weaknesses and strengths.
- Manage complaints between user and restaurant

5.4 System Coding

Database Connection

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Data.SqlClient;
using System.Data;

/// <summary>
/// Summary description for DbConnection
/// </summary>
public class DbConnection
{
    public DataSet ds = new DataSet();
    public SqlDataReader dr;
    public SqlConnection con()
    {
        SqlConnection con = new
SqlConnection("server=localhost;database=eDine;uid=sa;pwd=yuva");
        con.Open();
        return con;
    }
    public void exec(string str)
    {
        SqlCommand cmd = new SqlCommand(str, con());
        cmd.ExecuteNonQuery();
    }
    public int exec1(string str)
    {
        SqlCommand cmd = new SqlCommand(str, con());
        return cmd.ExecuteNonQuery();
    }
}
```

```
public SqlDataReader ret_dr(string str)
{
    SqlCommand cmd = new SqlCommand(str, con());
    return cmd.ExecuteReader();
}

public DataSet ret_ds(string str)
{
    SqlDataAdapter sqlda = new SqlDataAdapter(str, con());
    sqlda.Fill(ds);
    return ds;
}

public DbConnection()
{
    //
    // TODO: Add constructor logic here
    //
}

}
```

User Book Parking Slot

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Collections;
```

```
public partial class UserBookParkingSlot : System.Web.UI.Page
{
    DbConnection ob = new DbConnection();
    DbConnection ob1 = new DbConnection();
    public static int uid;
    ArrayList arslot= new ArrayList();
    protected void Page_Load(object sender, EventArgs e)
    {
        Session["uuname"] = "anu123";
        uid = getid();
        TextBox1.Text = uid.ToString();
        TextBox12.Text = Session["uuname"].ToString();
        TextBox3.Text = DateTime.Now.ToShortDateString();

        if (!IsPostBack)
        {
            DropDownList7.Items.Insert(0, new ListItem("SELECT", "0"));
            DropDownList8.Items.Insert(0, new ListItem("SELECT", "0"));
            DropDownList1.Items.Insert(0, new ListItem("SELECT", "0"));
            DropDownList2.Items.Insert(0, new ListItem("SELECT", "0"));
            ob.dr = ob.ret_dr("select distinct huname from PSlot");
            while (ob.dr.Read())
            {
                DropDownList7.Items.Add(ob.dr[0].ToString());
            }
            DropDownList2.Items.Insert(1, new ListItem("10AM - 12PM", "1"));
            DropDownList2.Items.Insert(2, new ListItem("12PM - 2PM", "2"));
            DropDownList2.Items.Insert(3, new ListItem("2PM - 4PM", "3"));
            DropDownList2.Items.Insert(4, new ListItem("4PM - 6PM", "4"));
            DropDownList2.Items.Insert(5, new ListItem("6PM - 8PM", "5"));
            DropDownList2.Items.Insert(6, new ListItem("8PM - 10PM", "6"));
            DropDownList2.Items.Insert(7, new ListItem("10PM - 12AM", "7"));
        }
    }
}
```

```

    }
}
public int getid()
{
    int c = 0;
    ob.dr = ob.ret_dr("select isnull(max(pid),500)+1 from pbook");
    if (ob.dr.Read())
    {
        c = Convert.ToInt32(ob.dr[0].ToString());
    }
    return c;
}
protected void Button1_Click(object sender, EventArgs e)
{
    if (TextBox1.Text == "" || TextBox2.Text == "" || TextBox3.Text == "" ||
    TextBox13.Text == "" || TextBox2.Text == "" || TextBox12.Text == "" ||
    DropDownList7.SelectedItem.ToString() == "Select" ||
    DropDownList8.SelectedItem.ToString() == "Select" ||
    DropDownList2.SelectedItem.ToString() == "Select")
    {
        Label11.Visible = true;
        Label11.ForeColor = System.Drawing.Color.Red;
        Label11.Text = "Please Enter all fields correctly!";
    }
    else
    {
        string str = "insert into pbook values(" + uid + "," +
        DropDownList7.SelectedItem.ToString() + "," + TextBox12.Text + "," +
        TextBox13.Text + "," + TextBox2.Text + "," + TextBox3.Text + ",0," +
        DropDownList1.SelectedItem.ToString() + "," +
        DropDownList2.SelectedItem.ToString() + ")";
        ob.exec(str);

        Label11.Visible = true;
        Label11.ForeColor = System.Drawing.Color.Green;
    }
}

```

```
Label11.Text = "Your request has been approved!";
Response.Write("<script type = 'text/javascript'>alert('Successfully Complete  
Thank you');</script>");
//TextBox1.Text = "";
//TextBox2.Text = "";
//TextBox3.Text = "";
if (DropDownList1.SelectedItem.ToString() == "Four Wheeler")
{
    Session["vtype"] = "II Floor";
}
else
{
    Session["vtype"] = "I Floor";
}
Session["parkid"] = TextBox1.Text ;
Session["houname"] = DropDownList7.SelectedItem.ToString();
Session["slottime"] = DropDownList2.SelectedItem.ToString();
Session["pslot"] = TextBox2.Text ;
Session["pdate"] = TextBox3.Text;
Session["table"] = DropDownList8.SelectedItem.ToString();
Response.Redirect("UserBookFood.aspx");

}
}
protected void Button2_Click(object sender, EventArgs e)
{

}

protected void DropDownList7_SelectedIndexChanged(object sender, EventArgs e)
{
    ob.dr = ob.ret_dr("select ptype from PSlot where  
huname='"+DropDownList7.SelectedItem.ToString()+"");
    while (ob.dr.Read())
```



```
{
    DropDownList1.Items.Add(ob.dr[0].ToString());

}
}
protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{
    int slot=0;
    ob.dr = ob.ret_dr("select tpark from PSlot where huname="" +
    DropDownList7.SelectedItem.ToString() + "" and
    ptype="" + DropDownList1.SelectedItem.ToString() + "");
    while (ob.dr.Read())
    {
        TextBox13.Text = ob.dr[0].ToString();
        slot = Convert.ToInt32(TextBox13.Text);

    }
    for (int i = 1; i <= slot; i++)
    {
        DropDownList8.Items.Add(i.ToString());
    }

}
protected void DropDownList8_SelectedIndexChanged(object sender, EventArgs e)
{
}
protected void DropDownList8_SelectedIndexChanged1(object sender, EventArgs e)
{
    ob.dr = ob.ret_dr("select bslot from pbook where huname="" +
    DropDownList7.SelectedItem.ToString() + "" and
    vtype="" + DropDownList1.SelectedItem.ToString() + "");
    while (ob.dr.Read())
    {
```

```
        arslot.Add(ob.dr[0].ToString());
    }
    if(arslot.Contains(DropDownList8.SelectedItem.ToString()))
    {
        TextBox2.Text = DropDownList8.SelectedItem.ToString()+" Not Available";
        Button3.Enabled = false;
    }
    else
    {

        TextBox2.Text = "Slot "+DropDownList8.SelectedItem.ToString();
    }
}
protected void DropDownList2_SelectedIndexChanged1(object sender, EventArgs e)
{

}
}
```

User Book Foo Items

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Data;

public partial class UserBookFood : System.Web.UI.Page
{
    DbConnection ob = new DbConnection();
```

```
DbConnection con = new DbConnection();
DbConnection ob1 = new DbConnection();
public static string ftype = "", ipath = "";
public static int uid;
public static int uid1;
DataSet ds = new DataSet();
double tt = 0.0;
int q = 0;
double sum = 0.0;
public static string id;
public static string fitems = "";
public static string p="";
protected void Page_Load(object sender, EventArgs e)
{
    Session["parkid"] = "501";
    Session["houname"] = "chikku123";
    Session["slottime"] = "10AM - 12PM";
    Session["pdate"] = "18/10/2022";
    Session["vtype"] = "I Floor";
    Session["pslot"] = "Slot 3";
    Session["uname"] = "anu123";
    Session["table"] = "3";

    TextBox10.Text = Session["vtype"].ToString();
    TextBox8.Text = "Table "+Session["table"].ToString();
    TextBox7.Text = Session["pslot"].ToString();
    TextBox6.Text = Session["uname"].ToString();
    TextBox5.Text = Session["slottime"].ToString();
    TextBox4.Text = Session["houname"].ToString();
    TextBox1.Text = Session["parkid"].ToString();
    TextBox9.Text = Session["pdate"].ToString();
}
```

```
if (!IsPostBack)
{
    DropDownList7.Items.Insert(0, new ListItem("SELECT", "0"));
    DropDownList1.Items.Insert(0, new ListItem("SELECT", "0"));
    DropDownList2.Items.Insert(0, new ListItem("SELECT", "0"));
    ob.dr = ob.ret_dr("select distinct ftype from hmenu");
    while (ob.dr.Read())
    {
        DropDownList7.Items.Add(ob.dr[0].ToString());
    }
}

public int getid()
{
    int c = 0;
    ob.dr = ob.ret_dr("select isnull(max(cartid),0)+1 from cart");
    if (ob.dr.Read())
    {
        c = Convert.ToInt32(ob.dr[0].ToString());
    }
    return c;
}

public int getid1()
{
    int c = 0;
    ob.dr = ob.ret_dr("select isnull(max(bookid),600)+1 from fbook");
    if (ob.dr.Read())
    {
        c = Convert.ToInt32(ob.dr[0].ToString());
    }
    return c;
}
```

```
}  
public int CurrentPageIndex  
{  
    get  
    {  
        if (ViewState["pg"] == null)  
            return 0;  
        else  
            return Convert.ToInt16(ViewState["pg"]);  
    }  
    set  
    {  
        ViewState["pg"] = value;  
    }  
}  
int pg = 0;  
protected void BindData()  
{  
    CurrentPageIndex = 0;  
    ds.Tables.Clear();  
    PagedDataSource pgd = new PagedDataSource();  
    string cmdstr = "select * from cart";  
    // string str = "select * from design";  
    ds = con.ret_ds(cmdstr);  
    pgd.DataSource = ds.Tables[0].DefaultView;  
    pgd.CurrentPageIndex = CurrentPageIndex;  
    pgd.AllowPaging = true;  
    pgd.PageSize = 12;  
    // LinkButton1.Enabled = !(pgd.IsLastPage);  
    // LinkButton2.Enabled = !(pgd.IsFirstPage);  
  
    DataList1.DataSource = pgd;  
    DataList1.DataBind();  
}
```

```
}
protected void Button1_Click(object sender, EventArgs e)
{
    uid = getid();
    if (TextBox1.Text == "" || TextBox12.Text == "" ||
DropDownList7.SelectedItem.ToString() == "Select")
    {
        Label11.Visible = true;
        Label11.ForeColor = System.Drawing.Color.Red;
        Label11.Text = "Please Enter all fields correctly!";
    }
    else
    {
        string stat = "0";

        string str = "insert into cart values(" + uid + "," +
DropDownList2.SelectedItem.ToString() + "," + ftype + "," + ipath + "," +
TextBox12.Text + "," + TextBox2.Text + "," + TextBox6.Text + "," + TextBox4.Text +
"," + stat + ")";
        ob.exec(str);
        Label11.Visible = true;
        Label11.ForeColor = System.Drawing.Color.Green;
        Label11.Text = "Your request has been approved!";
        BindData();
        calculatetotal();

    }
}

public void calculatetotal()
{
    sum = 0.0;
    fitems = "";
    ob.dr = ob.ret_dr("select price, nitem, fname from cart where uname=" +
TextBox6.Text + "and status=0");
```

```

while (ob.dr.Read())
{

    tt = Convert.ToInt64(ob.dr[0]);
    q = Convert.ToInt32(ob.dr[1].ToString());
    sum = sum + (tt * q);
    p=ob.dr[2].ToString()+"-"+ob.dr[1].ToString();
    fitems = fitems + ":" + p;
}
// long tt1 = tt * q;
TextBox3.Text = sum.ToString();
}
protected void Button2_Click(object sender, EventArgs e)
{

}

protected void DropDownList7_SelectedIndexChanged(object sender, EventArgs e)
{
    ob.dr = ob.ret_dr("select distinct ftime from hmenu where
ftype='"+DropDownList7.SelectedItem.ToString()+"");
    while (ob.dr.Read())
    {
        DropDownList1.Items.Add(ob.dr[0].ToString());

    }
}

protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{
    ob.dr = ob.ret_dr("select distinct fname from hmenu where ftype=" +
DropDownList7.SelectedItem.ToString() + " and
ftime='"+DropDownList1.SelectedItem.ToString()+"");
    while (ob.dr.Read())
    {

```

```
        DropDownList2.Items.Add(ob.dr[0].ToString());

    }
}

protected void DropDownList2_SelectedIndexChanged1(object sender, EventArgs e)
{
    ob.dr = ob.ret_dr("select * from hmenu where fname='" +
    DropDownList2.SelectedItem.ToString() + "'");
    if (ob.dr.Read())
    {
        TextBox12.Text = ob.dr[4].ToString();
        ftype = ob.dr[8].ToString();
        ipath = ob.dr[6].ToString();

    }
}

protected void DataList1_SelectedIndexChanged(object sender, EventArgs e)
{
    id = DataList1.DataKeys[DataList1.SelectedIndex].ToString();
    string sss = "delete from cart where cartid=" + id + "";
    ob1.exec(sss);
    calculatetotal();
    BindData();
}

protected void Button5_Click(object sender, EventArgs e)
{
    uid1 = getid1();
    if (TextBox1.Text == "" || TextBox12.Text == "" ||
    DropDownList7.SelectedItem.ToString() == "Select")
    {
        Label11.Visible = true;
        Label11.ForeColor = System.Drawing.Color.Red;
        Label11.Text = "Please Enter all fields correctly!";
    }
}
```



```
    }  
    else  
    {  
        string stat = "0";  
  
        string str = "insert into fbook values(" + uid1 + "," + TextBox1.Text + "," +  
        TextBox7.Text + "," + TextBox5.Text + "," + TextBox4.Text + "," + TextBox6.Text +  
        "," + TextBox8.Text + "," + TextBox10.Text + "," + TextBox9.Text + "," + fitems +  
        "," + TextBox3.Text + "," + stat + ")";  
  
        ob.exec(str);  
        Label11.Visible = true;  
        Label11.ForeColor = System.Drawing.Color.Green;  
        Label11.Text = "Your request has been approved!";  
        string sss = "delete from cart";  
        ob1.exec(sss);  
        BindData();  
        Label15.Visible = true;  
        Label15.ForeColor = System.Drawing.Color.Red;  
        Label15.Text = "Your order has been approved!!";  
        // calculatetotal();  
  
    }  
}  
}
```

View Hotel Menu

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Web;  
using System.Web.UI;  
using System.Web.UI.WebControls;  
using System.Data.SqlClient;
```

```
using System.Data;
```

```
public partial class UserVeiwHotelMenu : System.Web.UI.Page
{
    DbConnection con = new DbConnection();
    DbConnection ob = new DbConnection();
    DbConnection ob1 = new DbConnection();
    SqlDataReader dr;
    DataSet ds = new DataSet();
    string str;
    public static string id;
    public static string fpath = "", s = "", mode = "", str1 = "", pp = "";
    public int CurrentPageIndex
    {
        get
        {
            if (ViewState["pg"] == null)
                return 0;
            else
                return Convert.ToInt16(ViewState["pg"]);
        }
        set
        {
            ViewState["pg"] = value;
        }
    }
    int pg = 0;
    protected void BindData()
    {
        CurrentPageIndex = 0;
        ds.Tables.Clear();
        PagedDataSource pgd = new PagedDataSource();
```

```
        string cmdstr = "select * from hmenu where ftime='Starters' and  
        huname='"+TextBox8.Text+"'";  
        // string str = "select * from design";  
        ds = con.ret_ds(cmdstr);  
        pgd.DataSource = ds.Tables[0].DefaultView;  
        pgd.CurrentPageIndex = CurrentPageIndex;  
        pgd.AllowPaging = true;  
        pgd.PageSize = 12;  
        // LinkButton1.Enabled = !(pgd.IsLastPage);  
        // LinkButton2.Enabled = !(pgd.IsFirstPage);  
  
        DataList1.DataSource = pgd;  
        DataList1.DataBind();  
    }  
    protected void BindData2()  
    {  
        CurrentPageIndex = 0;  
        ds.Tables.Clear();  
        PagedDataSource pgd = new PagedDataSource();  
        string cmdstr = "select * from hmenu where ftime='Breads' and huname='" +  
        TextBox8.Text + "'";  
        // string str = "select * from design";  
        ds = con.ret_ds(cmdstr);  
        pgd.DataSource = ds.Tables[0].DefaultView;  
        pgd.CurrentPageIndex = CurrentPageIndex;  
        pgd.AllowPaging = true;  
        pgd.PageSize = 12;  
        // LinkButton1.Enabled = !(pgd.IsLastPage);  
        // LinkButton2.Enabled = !(pgd.IsFirstPage);  
  
        DataList2.DataSource = pgd;  
        DataList2.DataBind();  
    }
```

```
protected void BindData3()
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
    PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from hmenu where ftime='Curries' and huname="" +
    TextBox8.Text + """;
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
    pgd.DataSource = ds.Tables[0].DefaultView;
    pgd.CurrentPageIndex = CurrentPageIndex;
    pgd.AllowPaging = true;
    pgd.PageSize = 12;
    // LinkButton1.Enabled = !(pgd.IsLastPage);
    // LinkButton2.Enabled = !(pgd.IsFirstPage);

    DataList3.DataSource = pgd;
    DataList3.DataBind();
}
protected void BindData4()
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
    PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from hmenu where ftime='Biryani' and huname="" +
    TextBox8.Text + """;
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
    pgd.DataSource = ds.Tables[0].DefaultView;
    pgd.CurrentPageIndex = CurrentPageIndex;
    pgd.AllowPaging = true;
    pgd.PageSize = 12;
    // LinkButton1.Enabled = !(pgd.IsLastPage);
```

```
// LinkButton2.Enabled = !(pgd.IsFirstPage);

DataList4.DataSource = pgd;
DataList4.DataBind();
}
protected void BindData5()
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
    PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from hmenu where ftime='SeaFoods' and huname='" +
    TextBox8.Text + "'";
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
    pgd.DataSource = ds.Tables[0].DefaultView;
    pgd.CurrentPageIndex = CurrentPageIndex;
    pgd.AllowPaging = true;
    pgd.PageSize = 12;
    // LinkButton1.Enabled = !(pgd.IsLastPage);
    // LinkButton2.Enabled = !(pgd.IsFirstPage);

    DataList5.DataSource = pgd;
    DataList5.DataBind();
}
protected void BindData6()
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
    PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from hmenu where ftime='ChickenItems' and huname='" +
    TextBox8.Text + "'";
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
```

```
pgd.DataSource = ds.Tables[0].DefaultView;
pgd.CurrentPageIndex = CurrentPageIndex;
pgd.AllowPaging = true;
pgd.PageSize = 12;
// LinkButton1.Enabled = !(pgd.IsLastPage);
// LinkButton2.Enabled = !(pgd.IsFirstPage);

DataList6.DataSource = pgd;
DataList6.DataBind();
}
protected void BindData7()
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
    PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from hmenu where ftime='Juice' and huname='" +
    TextBox8.Text + "'";
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
    pgd.DataSource = ds.Tables[0].DefaultView;
    pgd.CurrentPageIndex = CurrentPageIndex;
    pgd.AllowPaging = true;
    pgd.PageSize = 12;
    // LinkButton1.Enabled = !(pgd.IsLastPage);
    // LinkButton2.Enabled = !(pgd.IsFirstPage);

    DataList7.DataSource = pgd;
    DataList7.DataBind();
}
protected void BindData8()
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
```

```
PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from hmenu where ftime='Desserts' and huname='" +
    TextBox8.Text + "'";
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
    pgd.DataSource = ds.Tables[0].DefaultView;
    pgd.CurrentPageIndex = CurrentPageIndex;
    pgd.AllowPaging = true;
    pgd.PageSize = 12;
    // LinkButton1.Enabled = !(pgd.IsLastPage);
    // LinkButton2.Enabled = !(pgd.IsFirstPage);

    DataList8.DataSource = pgd;
    DataList8.DataBind();
}
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
    }
}
protected void Button1_Click(object sender, EventArgs e)
{
    BindData();
    BindData2();
    BindData3();
    BindData4();
    BindData5();
    BindData6();
    BindData7();
    BindData8();
}
}
```

Hotel Add Parking Slots

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class HotelAddParkingSlot : System.Web.UI.Page
{
    DbConnection ob = new DbConnection();
    DbConnection ob1 = new DbConnection();
    public static int uid;
    protected void Page_Load(object sender, EventArgs e)
    {
        Session["huname"] = "chikku123";
        uid = getid();
        TextBox1.Text = uid.ToString();
        TextBox12.Text = Session["huname"].ToString();

        if (!IsPostBack)
        {
            DropDownList7.Items.Insert(0, new ListItem("SELECT", "0"));
            DropDownList7.Items.Insert(1, new ListItem("Two Wheeler", "1"));
            DropDownList7.Items.Insert(2, new ListItem("Four Wheeler", "2"));
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        if (TextBox1.Text == "" || TextBox12.Text == "" ||
        DropDownList7.SelectedItem.ToString() == "Select")
        {

```



```
        Label11.Visible = true;
        Label11.ForeColor = System.Drawing.Color.Red;
        Label11.Text = "Please Enter all fields correctly!";
    }
    else
    {
        string stat = "0";
        string str = "insert into PSlot values(" + TextBox1.Text + "," +
        DropDownList7.SelectedItem.ToString() + "," + TextBox13.Text + "," +
        TextBox12.Text + "," + stat + ")";
        ob.exec(str);
        Label11.Visible = true;
        Label11.ForeColor = System.Drawing.Color.Green;
        Label11.Text = "Your request has been approved!";
        Response.Write("<script type = 'text/javascript'>alert('Successfully Complete  
Thank you');</script>");
    }
}

public int getid()
{
    int c = 0;
    ob.dr = ob.ret_dr("select isnull(max(psid),200)+1 from PSlot");
    if (ob.dr.Read())
    {
        c = Convert.ToInt32(ob.dr[0].ToString());
    }
    return c;
}

protected void Button2_Click(object sender, EventArgs e)
{
}
}
```

Hotel View Booking

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Data;

public partial class HotelViewBooking : System.Web.UI.Page
{
    DbConnection con = new DbConnection();
    DbConnection ob = new DbConnection();
    DbConnection ob1 = new DbConnection();
    SqlDataReader dr;
    DataSet ds = new DataSet();
    string str;
    public static string id;
    public static string fpath = "", s = "", mode = "", str1 = "", pp = "";
    public static string huname="";
    public int CurrentPageIndex
    {
        get
        {
            if (ViewState["pg"] == null)
                return 0;
            else
                return Convert.ToInt16(ViewState["pg"]);
        }
        set
        {

```

```

        ViewState["pg"] = value;
    }
}
int pg = 0;
protected void BindData(string fdate)
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
    PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from fbook where fdate='"+fdate+"' and huname='" +
huname + "' and status=0";
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
    pgd.DataSource = ds.Tables[0].DefaultView;
    pgd.CurrentPageIndex = CurrentPageIndex;
    pgd.AllowPaging = true;
    pgd.PageSize = 12;
    // LinkButton1.Enabled = !(pgd.IsLastPage);
    // LinkButton2.Enabled = !(pgd.IsFirstPage);

    DataList1.DataSource = pgd;
    DataList1.DataBind();
}
protected void Page_Load(object sender, EventArgs e)
{
    TextBox3.Visible = false;
    Button5.Visible = false;
    Session["huname"] = "chikku123";
    huname=Session["huname"].ToString();
    if (!IsPostBack)
    {
        DropDownList7.Items.Insert(0, new ListItem("SELECT", "0"));
    }
}

```

```
        ob.dr = ob.ret_dr("select distinct fdate from fbook where huname='"+huname+"
and status=0");
        while (ob.dr.Read())
        {
            DropDownList7.Items.Add(ob.dr[0].ToString());

        }
    }

}

protected void DropDownList7_SelectedIndexChanged(object sender, EventArgs e)
{
    BindData(DropDownList7.SelectedItem.ToString());
}

protected void DataList1_SelectedIndexChanged(object sender, EventArgs e)
{
    TextBox3.Visible = true;
    Button5.Visible = true;
    string sitem = ""; ;
    id = DataList1.DataKeys[DataList1.SelectedIndex].ToString();
    ob.dr = ob.ret_dr("select fitem from fbook where bookid=" + id + "");
    if (ob.dr.Read())
    {
        sitem = ob.dr[0].ToString();
        string[] values = sitem.Split(':');
        for (int i = 1; i < values.Length; i++)
        {
            TextBox3.Text+= values[i].Trim()+"\n";
        }
    }
}

protected void Button5_Click(object sender, EventArgs e)
{

```

```
        string s1 = "update fbook set status=1 where bookid=" + id + "";  
        ob1.exec(s1);  
    }  
}
```

Admin Approve Hotel

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Web;  
using System.Web.UI;  
using System.Web.UI.WebControls;  
using System.Data.SqlClient;  
using System.Data;  
  
public partial class AdminApproveHotel : System.Web.UI.Page  
{  
    DbConnection con = new DbConnection();  
    DbConnection ob = new DbConnection();  
    DbConnection ob1 = new DbConnection();  
    SqlDataReader dr;  
    DataSet ds = new DataSet();  
    string str;  
    public static string id;  
    public static string fpath = "", s = "", mode = "", str1 = "", pp = "";  
    public int CurrentPageIndex  
    {  
        get  
        {  
            if (ViewState["pg"] == null)  
                return 0;  
            else
```

```
        return Convert.ToInt16(ViewState["pg"]);
    }
    set
    {
        ViewState["pg"] = value;
    }
}
int pg = 0;
protected void BindData()
{
    CurrentPageIndex = 0;
    ds.Tables.Clear();
    PagedDataSource pgd = new PagedDataSource();
    string cmdstr = "select * from hreg where status=0";
    // string str = "select * from design";
    ds = con.ret_ds(cmdstr);
    pgd.DataSource = ds.Tables[0].DefaultView;
    pgd.CurrentPageIndex = CurrentPageIndex;
    pgd.AllowPaging = true;
    pgd.PageSize = 12;
    // LinkButton1.Enabled = !(pgd.IsLastPage);
    // LinkButton2.Enabled = !(pgd.IsFirstPage);

    DataList1.DataSource = pgd;
    DataList1.DataBind();
}
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        BindData();
    }
}
```

```
protected void DataList1_SelectedIndexChanged(object sender, EventArgs e)
{
    string uname = "";
    id = DataList1.DataKeys[DataList1.SelectedIndex].ToString();
    string sss = "update hreg set status=1 where hid=" + id + "";
    ob1.exec(sss);
    ob.dr = ob.ret_dr("select huname from hreg where hid=" + id + "");
    if (ob.dr.Read())
    {
        uname = ob.dr[0].ToString();
    }
    string r = "2";
    string s1 = "update Login set status=2 where Username=" + uname + "";
    ob1.exec(s1);
    BindData();
}

protected void DataList1_EditCommand(object source, DataListCommandEventArgs
e)
{
}

protected void DataList1_CancelCommand(object source,
DataListCommandEventArgs e)
{
}
}
```

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 entitled “E-Dine” was completed on time and is found working effectively under all circumstances that arise in the real environment. Using the facilities and functionalities of Dot Net with C# and MSSql, the software has been developed keeping in view of limitations of the existing system. For maximum and effective utilization of the system the user must make sure to follow the instructions so that data entries must be made in time and they must be completed. This system is highly user-friendly and is well to make easy interaction with the administrator/users of the system.

The project “E-Dine“, after being tested and was formed to be what is meant for. The system involves the entire conceptualization of training schedule and feedback system. It has been developed using the actual IT standards with the entire life cycle like System analysis, System design, Development and with three different levels of testing namely Unit testing, Module testing and Integration tests

CHAPTER – 10

FUTURE ENHANCEMENT

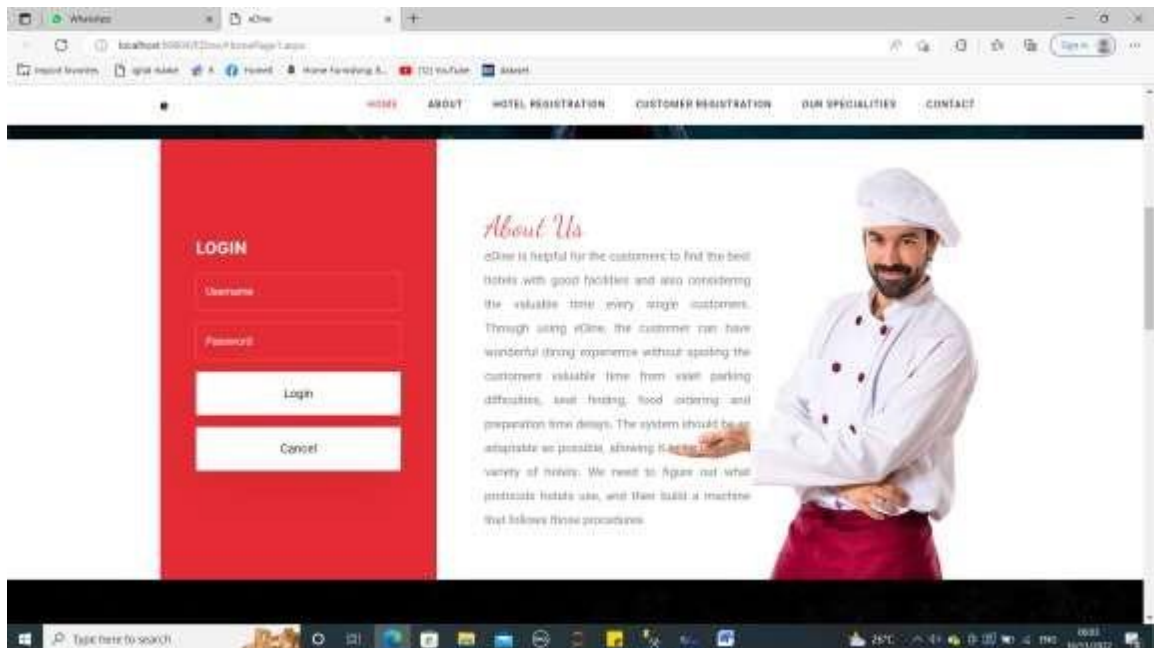
The changing requirements and the fast growth of technology will surely force the system to impose some changes forecasting those needs. E-Dine is developed in such a way that the enhancement can easily made without making major changes to the application. Though the system is working in certain assumptions it can be modified easily to any kind of requirements.

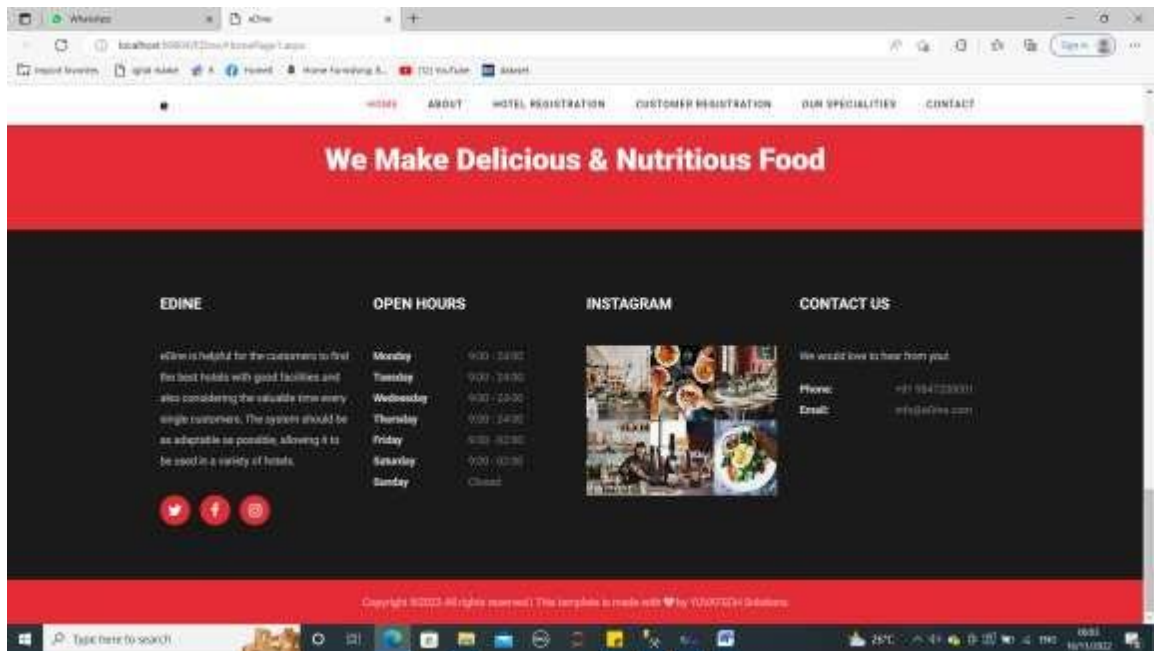
- Extends the application in mobile technology.
- Can add the functionalities of events organizer.
- Sensor enabled parking and table allocation.
- Sensor enables waiter calling system.

APPENDIX

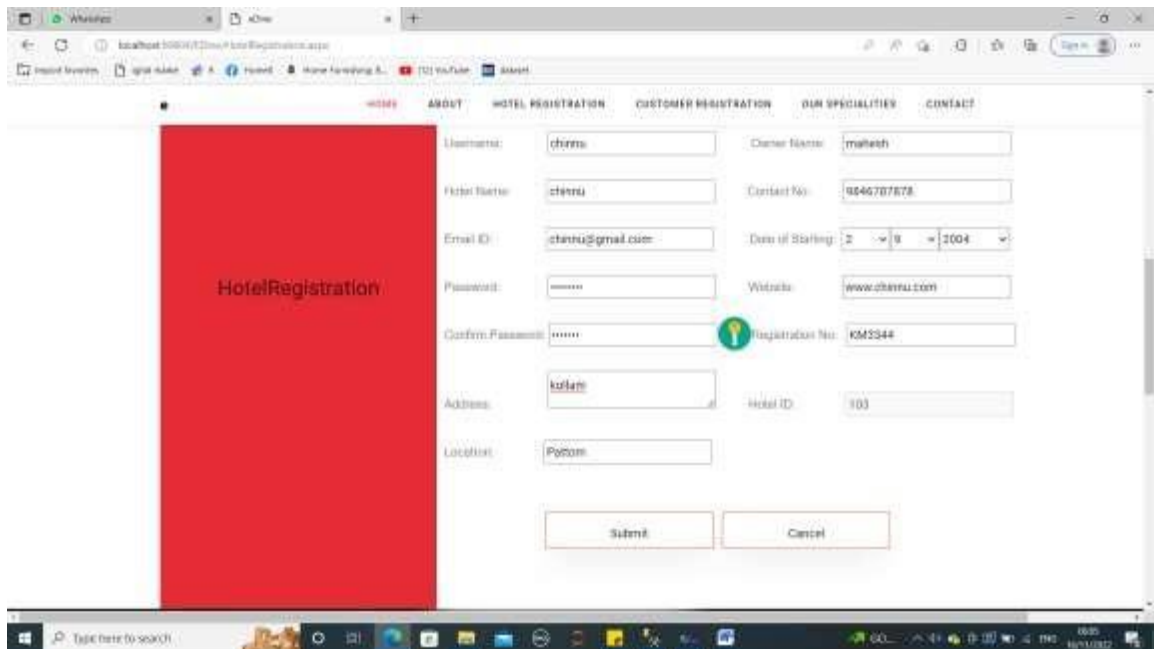
INPUT AND OUTPUT FORMS

Home Page





Hotel Registration



User Registration

CustomerRegistration

User Registration

Username: Name:

Location: Contact No:

Email ID: Date of Birth:

Password: Confirm Password:

Address: User ID:

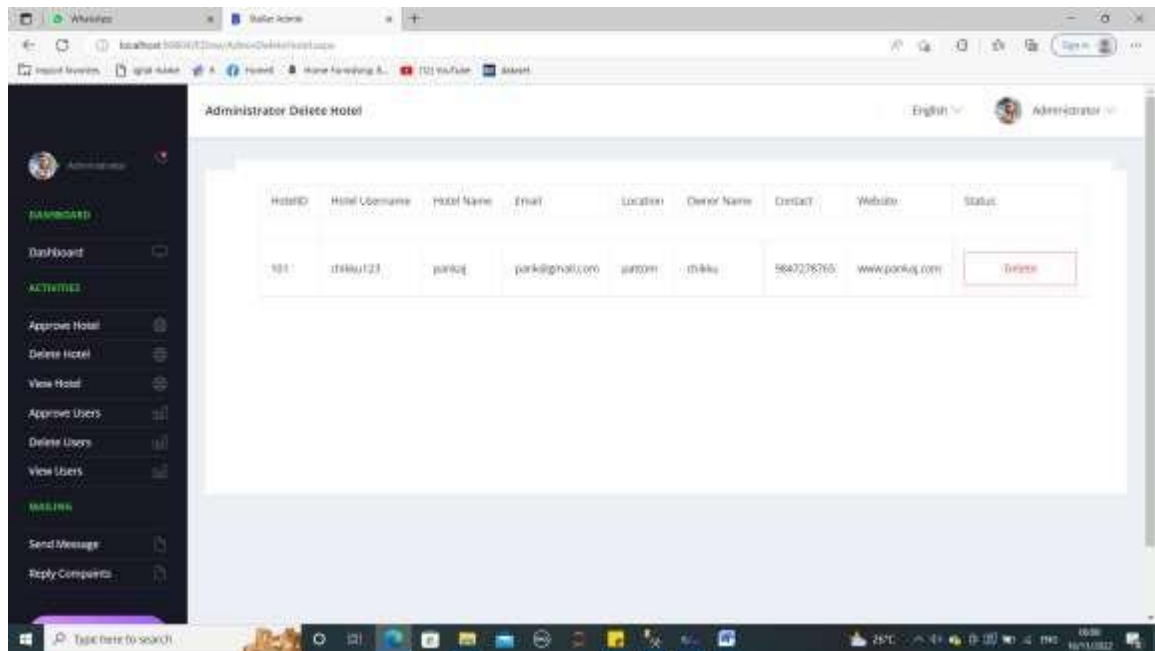
Admin Approve Hotel

Administrator Approve Hotel

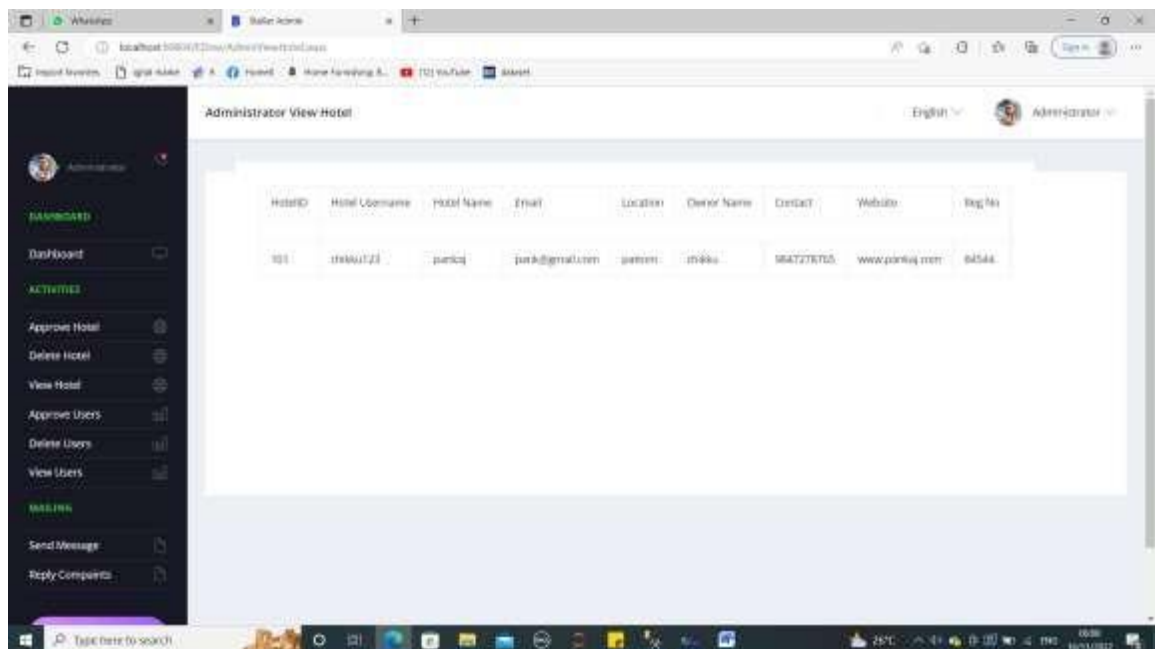
English Administrator

HotelID	Hotel Username	Hotel Name	Email	Location	Owner Name	Contact	Website	Status
102	nethu123	RoyM	nethu@gmail.com	patnem	nethu	9867676767	www.royal.com	<input type="button" value="Approve"/>

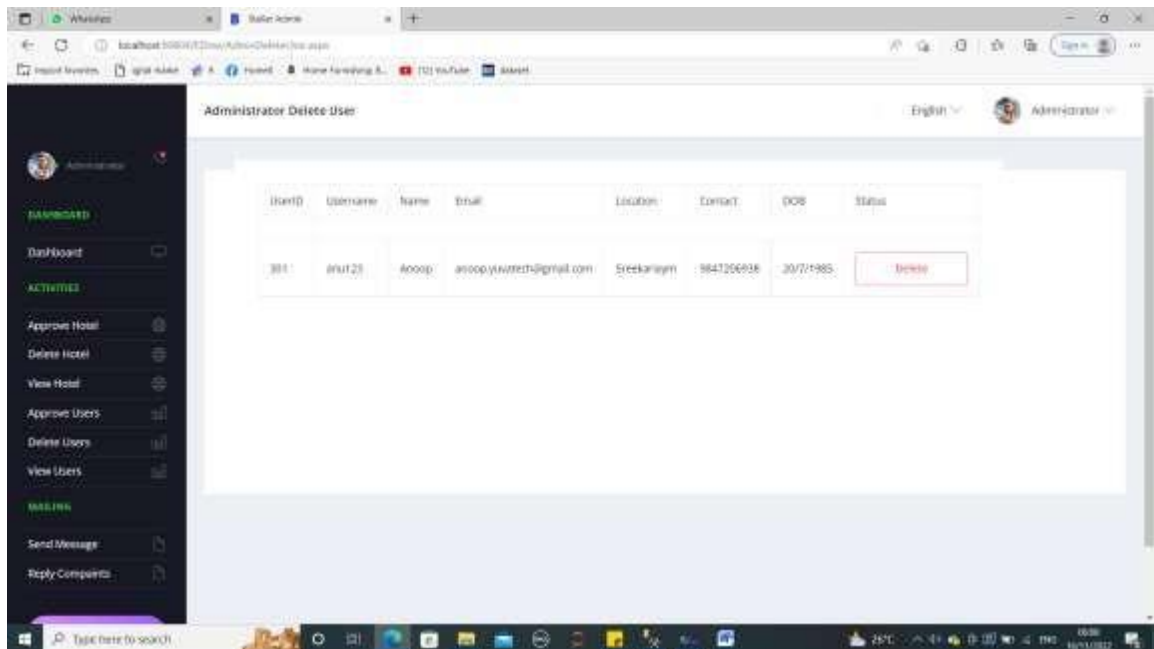
Admin Delete Hotel



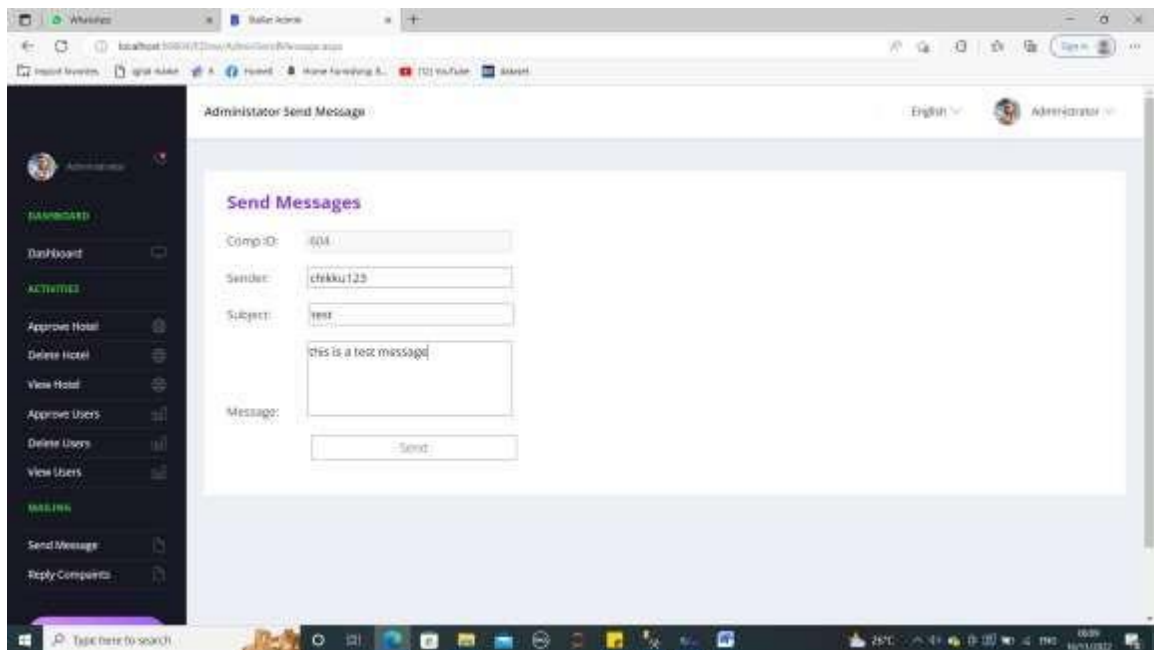
Admin View Hotel



Admin Delete User



Admin Send Messages



Hotel Add Parking Slot

The screenshot shows a web browser window with the URL `localhost:3000/Hotel/AddParkingSlot.aspx`. The page title is "Hotel Add Parking Slot". On the left is a dark sidebar menu with options: Dashboard, Add Parking Slot, Update Parking Slot, Add Menu, Delete Menu, View Booking, Send Complaints, Reply Complaints, and a Sign Out button. The main content area contains a form with the following fields: Slot ID (text input with value 205), Parking Type (dropdown menu with value Two Wheeler), No of Parking (text input with value 8), and UserName (text input with value c78ku123). A red Submit button is at the bottom of the form. The Windows taskbar at the bottom shows the time as 09:30 on 10/11/2022.

Hotel Add Parking Slot

Slot ID: 205

Parking Type: Two Wheeler

No of Parking: 8

UserName: c78ku123

Submit

Hotel Add Menu

The screenshot shows a web browser window with the URL `localhost:3000/Hotel/AddMenu.aspx`. The page title is "Hotel Add Menu". The sidebar menu is identical to the previous screenshot. The main content area contains a form with the following fields: Menu ID (text input with value 402), Food Type (dropdown menu with value VEG), Food Time (dropdown menu with value Breakfast), Food Name (text input with value Grate), Quantity (text input with value 1), Price (text input with value 12), and Username (text input with value c78ku123). There is a "Choose File" button for the Food Image field, with the text "(No file chosen)" next to it. A red Submit button is at the bottom of the form. The Windows taskbar at the bottom shows the time as 09:31 on 10/11/2022.

Hotel Add Menu

Menu ID: 402

Food Type: VEG

Food Time: Breakfast

Food Name: Grate

Quantity: 1

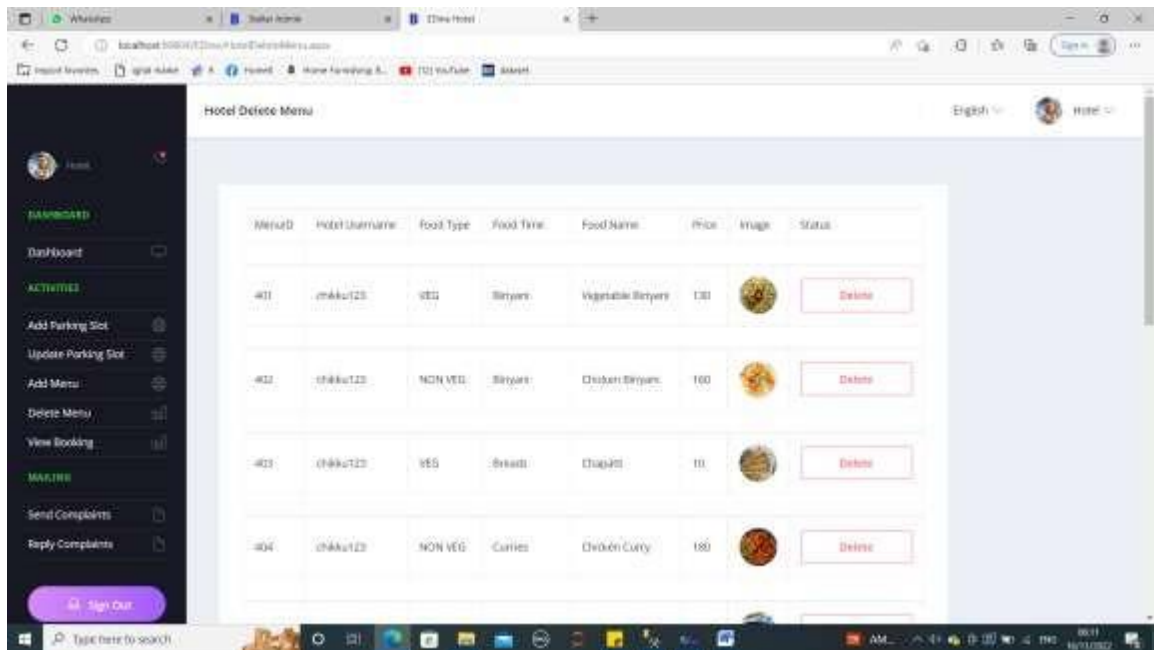
Price: 12

Username: c78ku123

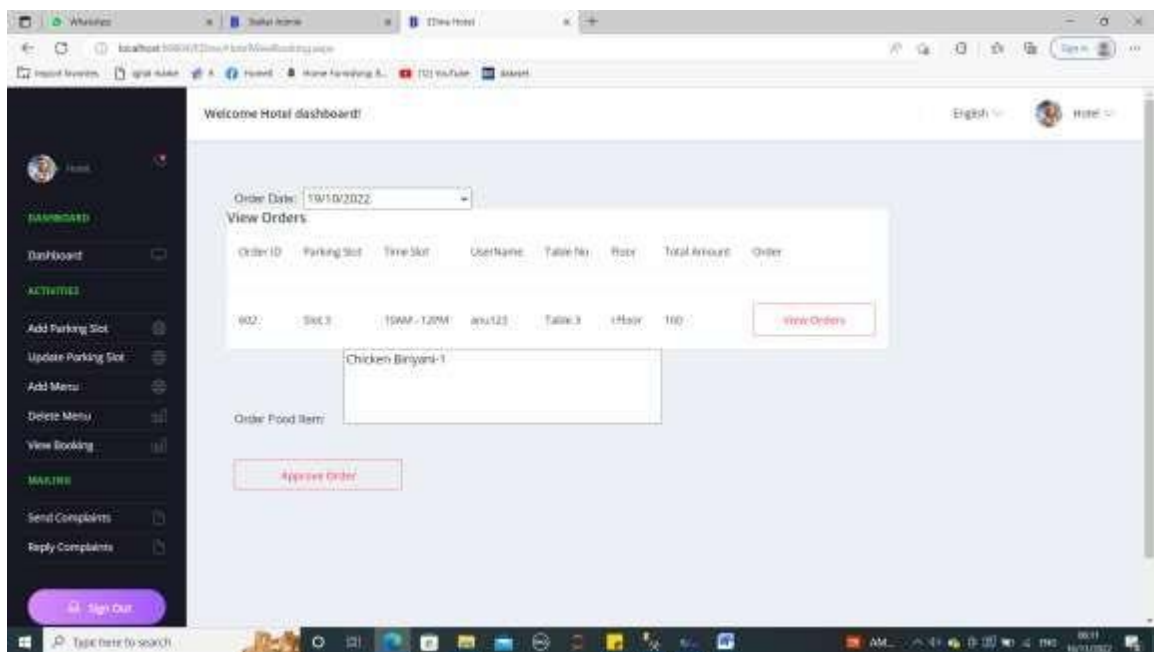
Food Image: Choose File (No file chosen)

Submit

Hotel Delete Menu



Hotel View Booking



Send Complaints

Welcome Hotel dashboard!

Complaint Registration

Comp ID:

Sender:

Subject:

This is a test message

Complaints:

User View Hotel Menu

Welcome Customer dashboard!

Hotel List

Hotel ID	Hotel Name	Rating	Location	Contact
101	partial	<div></div>	patton	9847276765
102	Royal	<div></div>	patton	9846767876

Top Food List

Food Name	Food Type	Price	Hotel	Category
Vegetable Biryani	Biryani	130	chiku123	VEG
Chicken Biryani	Biryani	160	chiku123	NON VEG
Dhappi	Bread	10	chiku123	VEG
Chicken Curry	Curries	180	chiku123	NON VEG
Corn Soups	Starters	20	chiku123	VEG


User Book Slot

The screenshot shows a web browser window with the URL `localhost:5000/12new/UserBookSlot.aspx`. The page is titled "Welcome Hotel dashboard!". On the left, there is a dark sidebar with a user profile icon and the name "Customer". The sidebar menu includes "DASHBOARD", "Dashboard" (with a notification icon), "ACTIVITIES", "Book Parking Slot", "View Hotel Menu", "MAILING", and "Send Complaints". A "Sign Out" button is at the bottom of the sidebar. The main content area contains a form with the following fields: "Parking ID" (505), "Select Hotel" (chikku123), "Vehicle Type" (Four Wheeler), "Total Slots" (20), "Time Slot" (12PM - 2PM), "Available Slots" (10), "Username" (anu123), "Book Slot" (Slot 10), and "Date" (16/11/2022). A "Book" button is at the bottom of the form. The browser's taskbar at the bottom shows the Windows search bar and various application icons.

Book Food

The screenshot shows a web browser window with the URL `localhost:5000/12new/UserBookFood.aspx`. The page is titled "Welcome Customer dashboard!". On the left, there is a dark sidebar with a user profile icon and the name "Customer". The sidebar menu includes "DASHBOARD", "Dashboard" (with a notification icon), "ACTIVITIES", "Book Parking Slot", "View Hotel Menu", "MAILING", and "Send Complaints". A "Sign Out" button is at the bottom of the sidebar. The main content area contains a form with the following fields: "Parking ID" (501), "Hotel Name" (chikku123), "Time Slot" (10AM - 12PM), "Username" (anu123), "Parking Slot" (Slot 3), "Table Name" (Table 3), "Date" (16/10/2022), "Floor" (1 Floor), "Food Type" (NON VEG), "Dish Type" (Biryani), "Food Name" (Chicken Biryani), "Price" (180), and "No of Items" (2). An "Add Items" button is at the bottom of the form. Below the form, there is an "Order List" section. The browser's taskbar at the bottom shows the Windows search bar and various application icons.

Order

Welcome Customer dashboard! English 

Send Complaints

[Sign Out](#)

Order type:

Food Name:




Price:

No of Items:


[Add Items](#)


Your request has been approved!

Order List:

	Food Name	Type	Image	Price	No of Items	Remove
2	Chicken Briyani	NON VEG		Rs.150/-	1	Delete
3	Chicken Briyani	NON VEG		Rs.150/-	2	Delete
4	Chicken Briyani	NON VEG		Rs.150/-	2	Delete

User Send Message

Welcome Hotel dashboard! English 

 Host

MANAGEMENT

- Dashboard

ACTIVITIES

- Add Parking Slot
- Update Parking Slot
- Add Menu
- Delete Menu
- View Booking

MAILING

- Send Complaints
- Reply Complaints

[Sign Out](#)

Complaint Registration

Comp-ID:

Sender:

Subject:

This is a test message!

Complaints:

[Register](#)

BIBLIOGRAPHY

1. Beginning Visual C# - Watson, White, Wrox Publications.
2. C# Programming Bible-Jeff Ferguson, Meeta Gupta.
3. Pro ASP.Net 4.0 in C# -Mac Donald and Szpuszta.
4. Software Engineering-Roger S Pressman.
5. System Analysis and Design- James A. Senn.