
EFFICIENT MULTIFACTOR AUTHENTICATION KEY EXCHANGE SCHEME OF MOBILE COMMUNICATION

A dissertation submitted in partial fulfillment of the requirements for the award of
degree of

BACHELOR OF COMPUTER APPLICATIONS
Of
University of Mysore



By
Dharani MP : BC201121

Under the Guidance of:
PRAPULLA GOWDA M P
Assistant Professor, Dept of BCA
Mysuru



DEPARTMENT OF BACHELOR OF COMPUTER APPLICATIONS
BGS FIRST GRADE COLLEGE MYSURU-23

2022-2023

DECLARATION

We **Dharani MP**, student of final semester BCA of BGS First Grade College, Mysuru, hereby declare that the dissertation entitled — **EFFICIENT MULTIFACTOR AUTHENTICATION KEY EXCHANGE SCHEME OF MOBILE COMMUNICATION** has been independently carried at —**BGS First Grade College**, Mysuru and submitted in partial fulfillment of the requirement for the award of **Bachelor of Computer Applications** affiliated to the **University of Mysore**, during the academic year 2022 – 2023. Further the matter embodied in the report is an original and bonafide work done by us.

To our knowledge this dissertation has not been submitted to any other college or university or published at any time prior to this.

Place: Mysuru

Dharani MP : BC201121

Date:



CERTIFICATE

This is to Certify that —**Dharani MP** bearing —**BC201121** has completed his/her final semester project work entitled — **EFFICIENT MULTIFACTOR AUTHENTICATION KEY EXCHANGE SCHEME OF MOBILE COMMUNICATION** as a partial fulfillment for the award of Bachelor of Computer Applications of University of Mysore during the year 2022-2023. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirement in respect of project work prescribed for the said degree.

Internal Guide:

PRAPULLA GOWDA M P

Asst. Professor, Dept of
BCA, BGS FGC, Mysuru.

Mr.Hareesha.C

Assistant Professor & HOD

Dept.of BCA, BGS FGC, Mysuru.

Dr.R.S.Narasegowda

Principal

BGS FGC, Mysuru.

Examiner's Name & Signature

1.

2.

TABLE OF CONTENT

CHAPTER 1 : INTRODUCTION

1.1 Objective of a Work

CHAPTER 2 : LITERATURE SURVEY

2.1 EXISTING AND PROPOSED SYSTEM

2.1.1 Existing System

2.1.2 DISADVANTAGES OF EXISTING SYSTEM

2.2 Proposed System

2.2.1 Advantages of Proposed System

2.1.2 DISADVANTAGES OF EXISTING SYSTEM

2.2 Proposed System

2.2.1 Advantages of Proposed System

2.1 FEASIBILITY STUDY

2.2.1 Operational Feasibility

2.2.2 Technical Feasibility

2.2.3 Economic Feasibility

2.3 TOOLS AND TECHNOLOGY

2.3.1 C#(dot)NET

2.3.2 SQL-SERVER-2019

2.3.3 HTML

2.3.4 CSS3

2.4.1 Hardware Requirements

2.4.2 Software Requirements

CHAPTER 3: SOFTWARE REQUIREMENT SPECIFICATION

3.1 INTRODUCTION

3.1.1 Characteristics of SRS:

3.2 Purpose

3.3 SCOPE:

3.4 Functional Requirements

3.4.1 Inputs

3.4.2 Processing

3.4.3 Outputs

3.5 Non-Functional Requirements

3.5.1 Performance Requirements

3.5.2 Safety Requirements

3.5.3 Security Requirements

3.5.4 Software Quality Requirements

3.7 Module Analysis

3.7.1 Administrator Module:

3.7.2 Client

3.7.3 Manager

3.7.4 Employee

CHAPTER 4:SYSTEM DESIGN

4.1 Three Tier Architecture

4.2 Description

4.2.1 Client/Presentation Tier

4.2.2 Business/Logic Tier

4.2.3 Database Tier

4.2.4 Advantages of the 3-tier architecture

4.3 USE CASE DIAGRAM

4.3.1 INTRODUCTION

4.3.1 Use case Diagram for Admin

4.3.2 Use case Diagram for Client

4.3.3 Use case Diagram Employee

4.3.4 Use case Diagram for Manager

4.4 DATABASE DESIGN

4.4.1 Data normalization

4.5 ENTITY RELATIONSHIP DIAGRAM

4.6 DATABASE TABLES

CHAPTER 5: IMPLEMENTATION

5.1 PROJECT CODE

CHAPTER 6: TESTING

6.1 Unit Testing

6.2 Integration Testing

6.3 Validation Testing

6.4 Test Cases

CHAPTER 7: SNAPSHOTS

CONCLUSION

BIBLIOGRAPHY

CHAPTER 1

INTRODUCTION

Day by day, the power of attacks to guess or harvest passwords to gain illicit access to a system or data are becoming greater as the sophistication of password cracking techniques increases and high-power computing becomes more affordable. In the last three years, the number of phishing attacks for the purpose of account or identity theft has more than tripled. During the third quarter of 2022 alone, there were 15 million data breaches, due to internet users around the world having their accounts compromised by attackers. Nowadays, we are so susceptible to account theft attacks that, statistically, at least one of our online accounts (email, social networks, banks) will be hacked or subjected to an attempted hack in in the next 12 months. Hence, there is an important need to have more robust and secure access mechanisms to protect data and systems.

The most popular, yet the most basic, mechanism for user authentication is the use of passwords, mainly because the concept of using passwords is an efficient and cost-effective solution for user authentication. Passwords are an example of Single-Factor Authentication (SFA), which has been mostly adopted by the community due to its simplicity and user friendliness. The fundamental requirement for any password is that it should be easy to remember and must be sufficiently secure. In other words, the authentication process must be efficient, and passwords must be tough to guess. Nevertheless, this is the weakest level of authentication, and it has been realized that Single-Factor Authentication is not reliable to provide adequate protection, due to several security threats.

Multi- factor authentication was proposed to provide higher levels of safety and to add strong protection against account theft by greatly increasing the difficulty for attackers to gain access to information systems and data, even if passwords or PINs are compromised by phishing attacks or other means. MFA manages to do this with a layered approach, that is, with MFA a system requires a user to present a combination of two or more credentials to verify identity so access can be granted. MFA mechanisms are mostly based on biometrics, which is automated recognition of individuals, based on their behavioral, and biological characteristics. However, the utilization of

biological factors has its challenges, mainly related to ease of use, which largely impacts the usability of the MFA system. In addition, biometric mechanisms entail high implementation costs, and are still vulnerable to many different security attacks, such as presentation attacks, sensor output interception, denial of service attacks, and replay attacks, among others.

This paper proposes a novel multi-factor authentication mechanism that does not require additional hardware and is solely based on images and their user-established relations. The combination of text and graphics increases the password space, thereby making the authentication mechanism more robust and secure against various types of security threats. The contributions of this research work are the following:

- The design of a novel MFA algorithm, based on image selection and user-established relations.
- Functional prototype of the mechanism developed and deployed as a mobile application available for IOS and Android.
- An analysis of the accuracy, security, and usability results focusing on the benefits and areas of opportunity that working with an image selection and relations-based algorithm has in an MFA mechanism.

Objective of a Work

The aim of this site is to provide a single platform for students and the staffs to interact with each other, such as discussing their queries, uploading their job openings, checking the IA Marks provided by the staff members, uploading their messages, images and videos in their timeline, viewing the notice published by the staff members. The admin of the site takes care of the activities such as adding and removing the galleries and also manages the students and staffs, students and staff members and publishing notices.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING AND PROPOSED SYSTEM

2.1.1 Existing System:

Textual passwords are commonly used in existing systems. Users do not follow their requirements. Users tend to choose meaningful words from dictionaries, it makes textual passwords easy to break and vulnerable to dictionary or brute force attacks. The authentication scheme has its disadvantages based on several factors such as consistency, uniqueness and acceptability. One of the main drawbacks of applying is its intrusiveness upon a user's personal characteristic. These schemes require the user to willingly subject there to a low-intensity infrared light. Systems require a special scanning device to authenticate users, which is not applicable for remote and Internet users.

Single-factor authentication only provides limited security, then combining together is considered as a good way to achieve higher security. For SMS-based two-factor authentication was widely adopted and has been used in many applications, e.g., Gmail. It dominates the research of authenticated key exchange AKE protocols for a long time. Among different factors, password was preferred. The first password-only authenticated key exchange protocol, encrypted key exchange the client shares a plaintext password with the server and exchanges encrypted information.

The limited human memory and the increasing attacker ability have made PAKEs less secure than expected. The protocol designers chose to add other authentication factors to improve security. Many two-factor AKE protocols as well as multi-factor AKE protocols have been proposed.

The computation cost of the client and server, are comparatively inefficient among all schemes, the key reason for which is that both of them authenticate biometric templates bit by bit. The communication traffics of them are also relative higher than others. FMA works by executing a

series of sub-protocols. Some of them can be executed parallel, the overall round complexity keeps high.

Two ways are used for biometric matching: Direct matching, comparing two templates bit by bit, as far as the total number of different bits is lower than a threshold, the two templates are considered to be matched.

Fuzzy extractor, only matched templates can reproduce same randomness. Fuzzy extractor to avoid the heavy computation and communication costs of direct matching. For privacy protection, the choice seems to be right, because adversaries may reconstruct a valid templates bit by bit from a complete transcript.

2.1.2 DISADVANTAGES OF EXISTING SYSTEM:

- Less security
- Less efficiency
- The Authentications are one or two factors are applied.
- Session Key not applied.

2.2 Proposed System:

The Multifactor Authentication scheme has been proposed, Users tend to resist using biometrics because of intrusiveness and the effect on their privacy and the hardware also USB devices. Graphical passwords are based on the idea that users can recall and recognize pictures better than words. Graphical password schemes require a long time to be performed. Present and evaluate our contribution, Multifactor authentication scheme. To be authenticated, it presents a virtual environment where the user navigates and interacts with various objects. The sequence of actions and interactions toward the objects inside the environment constructs the user's graphical password. The Graphical password can combine most existing authentication schemes such as textual passwords, graphical passwords, device passwords into a virtual environment. The design of the virtual environment and the type of objects selected determine the graphical password key space.

2.2.1 Advantages of Proposed System:

- High security.
- Multifactor Authentication like, textual, graphical passwords.
- The data can be exchanged to the receiver using session keys.
- Robustness.
- High Privacy.
- Usability.

2.2 FEASIBILITY STUDY

A feasibility analysis usually involves a thorough assessment of the operational (need), financial and technical aspects of a proposal. It is the test of the system proposal made to identify whether our user needs may be satisfied using the current software and hardware technologies, whether the system will be cost effective from a business point of view and whether it can be developed with the given budgetary constraints. Feasibility study is carried out to determine whether the proposed system is possible to develop with available resources and what should be the cost consideration. The three essential considerations are involved they are

- Operational Feasibility
- Technical Feasibility
- Economic Feasibility

2.2.1 Operational Feasibility

The system provides end-users and managers with timely, pertinent, accurate, and usefully formatted information. The time will be saved. Provide services to the department user base. The new proposed system is very much useful to the users.

2.2.2 Technical Feasibility

The project has been developed in such a way that the necessary functions and performance are achieved within the constraints. Technical feasibility centers on the existing excel format of the storing the file of each and every one and other functionalities and to what extent it can support the system. According to feasibility analysis procedure the technical feasibility of the system is analyzed and the technical requirements such as software facilities, Procedure, inputs are identified. It is also one of the important phases of the System development activities. The system offers greater levels of user friendliness combined with greater processing speed. Therefore, the cost of maintenance can be reduced. Since, Processing speed is very high and the work is reduced in the maintenance point of view management convince that the project is operationally feasible.

2.2.3 Economic Feasibility

Economic analysis is most frequently used for evaluation of the effectiveness of the system. More commonly known as cost/benefit analysis the procedure is to determine the benefit and saving that are expected from a System and compare them with costs, decisions is made to design and Implement the system. This part of feasibility study gives the top management the economic justification for the new system. A simple economic analysis that gives the actual comparison of costs and benefits is much more meaningful in such cases. In this system, the organization is most satisfied by economic feasibility. Because, if the organization implements this system, it need not require any additional hardware resources as well as it will be saving lot of time.

2.3 TOOLS AND TECHNOLOGY

2.3.1 C#(dot)NET

ADo(dot)NET OVERVIEW

The Set of data entity, which is built on XML, provides a standardized programming paradigm that operates with all storing data types, including flat, hierarchical. It accomplishes this by possessing no 'knowing' of an information's source and expressing the information in collects and types of data. The information within the Set of data is handled using that number of predefined APIs accessible by the Set of data and its subsidiary objects, regardless of the accuracy of the information.

The controlled provider has precise and specific details about the accuracy of the information, but the Set of data does not. The controlled provider is responsible for connecting, filling, and persisting the Range of data from and from data storage. The System(dot)Data(dot)ole Db and System(dot)Data(dot)SqlClient OLE DB and SQL Server(dot)NET Data Providers (System(dot)Data(dot)Ole Db and System(dot)Data(dot)SQL client) are parts it's. The Command, Connection, Data Reader, and Data Adapter are the four fundamental objects provided by Net Framework. We'll go over each element of the Set of data as well as the OLE DB/SQL Database Network Operators in next sections of this article, describing what they're doing and to program against them.

- The subsequent sections shall introducing you to a certain items which have changed over time as well as those that are brand new. These are all the items:
- Communications. To connect to a databases and arrangement of data against this.
- Commands. To execute SQL instructions on a database.
- Data Readers. Allows accessing a SQL Server data source's forward-only flow of database files.
- Set of data: For saving, removing, and computing with flat, XML, and current iteration.

Connections:

Connections, which are defined by client classes like SQL Connection, are used to 'speak to' databases. Instructions are sent through connections, and the results are delivered.

Commands

Provider-specific classes like SQL Command represent commands, which include the data that's submitted to a db. A stored procedure call, a UPDATE statement, a statement that provides results are all examples commands.

Data Readers

A read-only/forward-only cursor over data is somewhat same with the Data Reader object. The Data Reader API will read both flat and hierarchical data. After running a command against a database, a Data Reader object is returned. The returned Data Reader object has a different format than a record set. You may be using the Data Reader to display the results a search list in a web page, for example.

2.3.2 SQL-SERVER-2019

The DB-management-system, abbreviated DBMS, allows users to view & turn existing data into knowledge. dBase, paradox, IMS, SQL, & SQL Server are examples of database-management-system. Customers may build, edit, retrieve in own databases using such technologies. A system is a database of data that is organized. The qualities of individuals, objects, and happenings usually referred to as data. Every data point is recorded with its own area in MySQL Database. The attributes pertaining to a specific people, object, or activity in Windows Server are packaged to together identify a final full data packet known as a report . There are a variety of elements for each report. There can't be 2 things with the same value in a document.

The study on your company demands during such a SQL-Server-Database design process reveals all of columns or properties of importance. Can specify any extra variables or alter its meaning of existing fields as the company needs evolve throughout period.

2.3.3 HTML

HTML stands for Hyper Text Markup Language. It is used to design web pages using markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. Markup language is used to define the text document within tag which defines the structure of web pages. HTML 5 is the fifth and current version of HTML. It has

improved the markup available for documents and has introduced application programming interfaces (API) and Document Object Model (DOM).

2.3.4 CSS3

CSS stands for Cascading Style Sheets. CSS is a standard style sheet language used for describing the presentation (i.e. the layout and formatting) of the web pages. Prior to CSS, nearly all of the presentational attributes of HTML documents were contained within the HTML markup (specifically inside the HTML tags); all the font colors, background styles, element alignments, borders and sizes had to be explicitly described within the HTML. As a result, development of the large websites became a long and expensive process, since the style information were repeatedly added to every single page of the website.

To solve this problem CSS was introduced in 1996 by the World Wide Web Consortium (W3C), which also maintains its standard. CSS was designed to enable the separation of presentation and content. Now web designers can move the formatting information of the web pages to a separate style sheet which results in considerably simpler HTML markup, and better maintainability.

CSS3 is the latest version of the CSS specification. CSS3 adds several new styling features and improvements to enhance the web presentation capabilities.

2.4.1 Hardware Requirements

The hardware is required for an actual running of the system. The minimum hardware required is as follows

- System : Pentium i3 Processor.
- Hard Disk : 500 GB.
- Monitor : 15'' LED
- Input Devices : Keyboard, Mouse

- Ram : 4 GB

2.4.2 Software Requirements

Software forms the heart of any system. It is responsible for driving the hardware. The software required is as follows

- Operating system : Windows 10.
➤ Coding Language : C#
➤ Backend : MSSQL SERVER
➤ IDE : Visual Studio

SYSTEM ANALYSIS

System analysis is an important activity that takes places when we are building new system information or changing existing ones.

Analysis is used to gain an understanding existing system and what is required of it. At the conclusions of the analysis, there is a system description and set of requirement for the new system, the analysis is only the requirement.

System modules are used to gain ambiguities often found in the system. Modeling techniques used in the system analysis avoids ambiguity by using precise modeling and constructs and process description. They also assist to define precisely the requirements of the new system. Software tools that help [Analyst in their work support system analysis].

One of the most important factors aim system analysis is to develop a good understanding of the system and its problems. A good understanding of the system enables the system designers to identify the correct problems and suggest realistic solutions for them. A system analyst must spend

a lot of time talking to users and finding out how they use the system, any problems they find with the system and what they expect from it.

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 INTRODUCTION

Software Requirement Specification (SRS) is a fundamental document, which forms the foundation of the software development process. SRS not only lists the requirements of a system but also has a description of its major features. These recommendations extend the IEEE standards. The recommendations would form the basis for providing clear visibility of the product to be developed serving as baseline for execution of a contract between client and the developer. SRS constitutes the agreement between clients and developers regarding the contents of the software product that is going to be developed. SRS should accurately and completely represent the system requirements as it makes a huge contribution to the overall larger system or may be a complete; the SRS should state the interfaces between the system and the software portion.

3.1.1 Characteristics of SRS:

- Understandable
- Unambiguous
- Complete
- Verifiable
- Consistent
- Modifiable
- Traceable

3.2 Purpose

The purpose of this document is to convey information about the application requirements, both functional and non-functional to the reader and the user.

This document provides the following.

- A specification of the application's functional and non-functional requirements.
- The document is intended to serve several groups of audiences.
- First, it is anticipated that the application designers will use the SRS. Designers will use the information recorded here as basis for creating the application's design.
- Second, the client for the project is expected to review this document. The SRS will serve to establish a basis for agreement between the client and development team about the functionality to be provided by the application.
- Third, the application maintainers will review the document to clarify their understanding of what the application does.
- Fourth, test planners will use this document to derive test plans and test cases.
- Finally, the project manager will use this document during project planning and monitoring.

3.3 SCOPE:

This document is only one that describes the requirement of the system. It is meant for use by the developers and will be the basis for validating the final delivered system. Any changes made to the requirements in future will have to go through a format change approval process. The developer is responsible for asking the clarifications where necessary and will not make any alterations without the permission of the client.

3.4 Functional Requirements

This section gives the list of Functional requirements which are applicable to the Project Monitoring System. Functional requirements are nothing but the services provided by the system to its end user.

Admin

In this Module admin will manage the Client, projects, Employee and Managers in the organization

Client

In this module client will login and they can Track and update the requirements of their projects.

Manager

In this module Manager has to login once they login they need to do multifactor authentication along with that managers will manage the projects and files which has been shared by clients also allocate the requirements file to the employee along with security key.

Employee

In this module employee will login once they login they need to do multifactor authentication and they can view the task and update the task status along with that if they want to check the documents, they need to request the key from manager to open the document

3.4.1 Inputs

Depending upon the action being preferred by the user, with the help of Textboxes the input is accepted by the through button selected or hyperlink clicked. The input at the project level is collected based on the requirement.

3.4.2 Processing

The data is accepted through the interface and thus the processing take place as per the user specification made.

3.4.3 Outputs

The user gets the output of the specified action selected and appropriate outputs will be displayed for the inputs.

3.5 Non-Functional Requirements

3.5.1 Performance Requirements

The product is web-based and run from an application server. Initial load time depends on the media from which the product is run.

3.5.2 Safety Requirements

Users' data is maintained in server so that the server must be secured and it should not be altered.

3.5.3 Security Requirements

Since the product is intranet based, user can interact with it through Internet. One can login into the system only if they provide correct user name and password. Only authorized user can access the data.

3.5.4 Software Quality Requirements

Any user must be able to use and understand the product easily. It must also tolerate a wide variety of wrong input possibilities from a user, such as incorrect responses or unforeseen keystrokes and selections from a pointing device.

3.7 Module Analysis

3.7.1 Administrator Module:

In this Module admin will manage the Client, projects, Employee and Managers in the organization

3.7.2 Client

In this module client will login and they can Track and update the requirements of their projects.

3.7.3 Manager

In this module Manager has to login once they login they need to do multifactor authentication along with that managers will manage the projects and files which has been shared by clients also allocate the requirements file to the employee along with security key.

3.7.4 Employee

In this module employee will login once they login they need to do multifactor authentication and they can view the task and update the task status along with that if they want to check the documents, they need to request the key from manager to open the document

CHAPTER 4

SYSTEM DESIGN

4.1 Three Tier Architecture

Through the appearance of local area networks, pc's came out of their isolation, and where were soon not only being connected but also to the servers. Client/Server computing was born.

Servers today are mainly the database servers; applications are the exception. However, database-servers only offer data on the server; consequently the application intelligence must be implemented on the pc's (client). Since there are only the architecturally tiered data server and the client, this is called 2-tier architecture. This model still predominant to day, and is actually opposite of its popular a terminal based proceeds or but had its entire attempt to achieve the efficiency in software development using tools, than to choose the step and stony art of "brain ware".

3-tier Architecture

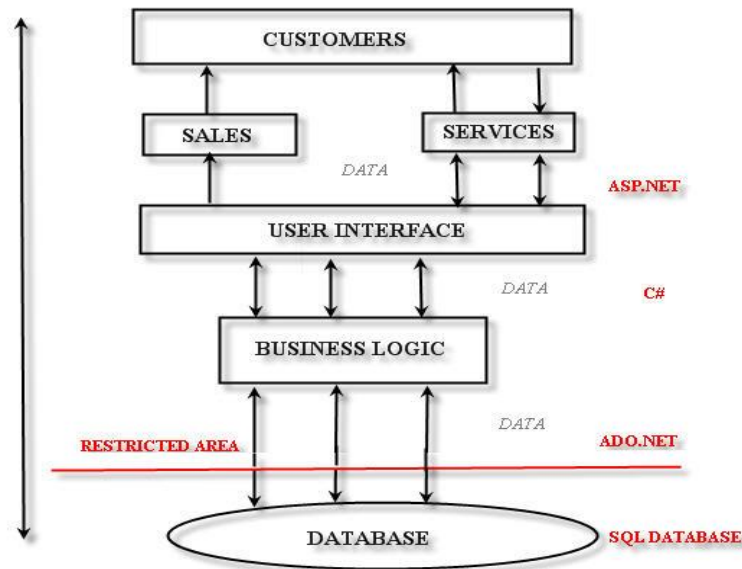


Fig 4.1.1 shows the 3-tier architecture

4.2 Description

4.2.1 Client/Presentation Tier

A client device is a piece of hardware that can communicate with Secure Global Desktop - using either a web browser, or the Secure Global Desktop Client.

The web browser (or Secure Global Desktop Client) running on the client device communicates with web servers and Secure Global Desktop servers on the second tier, and displays the applications that users may run. The Adaptive Internet Protocol (AIP) ensures optimal network usage between the first and second tiers. Network Computers (NCs) and PCs are examples of client devices. It is responsible for the presentation of the data, receiving user events and controlling the interface.

4.2.2 Business/Logic tier

This tier is new that is, it is not present in two-tier architecture in this explicit form. Business-objects that implement the business rules clients live here, and are available to the client

tier. This level now forms the central key to solving 2-tier problems. This protects the data from direct access by the clients.

The second tier may contain a single Secure Global Desktop server, or many Secure Global Desktop servers configured to form an array.

A Secure Global Desktop server is responsible for:

- Authenticating users when they log in to Secure Global Desktop.
- Negotiating with application servers to authenticate users when they run applications, prompting users for passwords when necessary.
- Causing client devices to display the applications, either using Java applets embedded in web pages downloaded to the client device, or using the Secure Global Desktop Client.
- Keeping track of running applications even after users have logged out, so that they can resume them later.

4.2.3 Database Tier

This tier is responsible for data storage. Besides the wide spread relational database systems, existing legacy systems database are often reused here. It is important to note that boundaries between the tiers are logical. The main importance is that the system is neatly structured, and there is a well definition of the software boundaries between the different tiers.

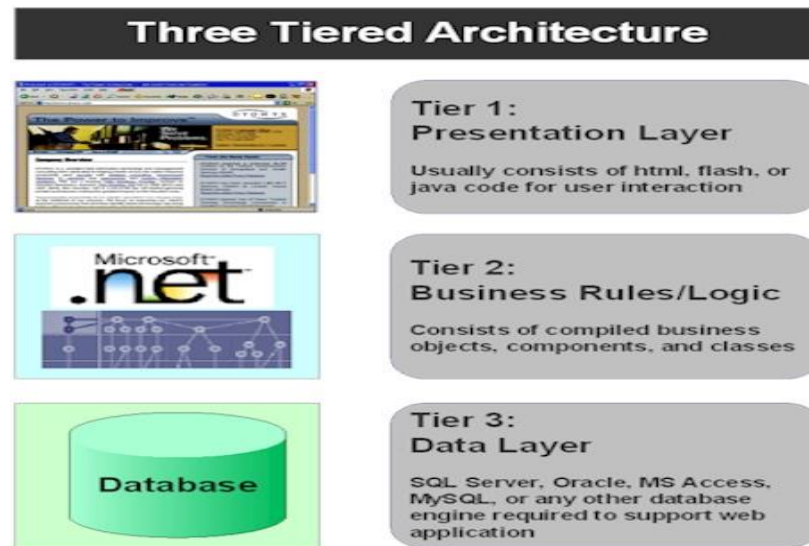


Fig 4.2.3 shows the database tier

4.2.4 Advantages of the 3-tier Architecture

- As previously mentioned 3-tier architecture solves a number of problems that are inherit to 2-tier architectures. Naturally it also causes new problems but these are out weighted by the advantages.
- Clear separations of the user interface control and data presentation from the application logic. Through this separation more clients are able to have access to wide variety of functions.

4.3 USE CASE DIAGRAM

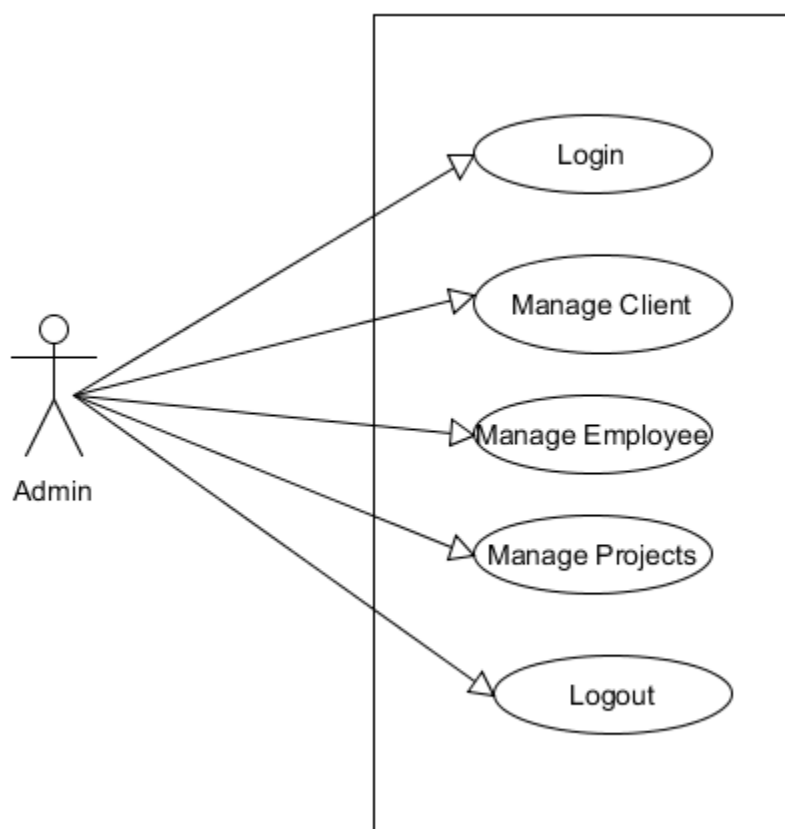
INTRODUCTION

Use-case is a special flow of events through the system. The use-case model express what the business or application will do and not how. Use case is an interaction between the users and a system; it captures the goal of the users and the responsibility of the system to its users. The use case model describes the uses of the system and shows the courses of events that can be performed. Use case diagram is a graph of actors, asset of use cases enclosed by a system.

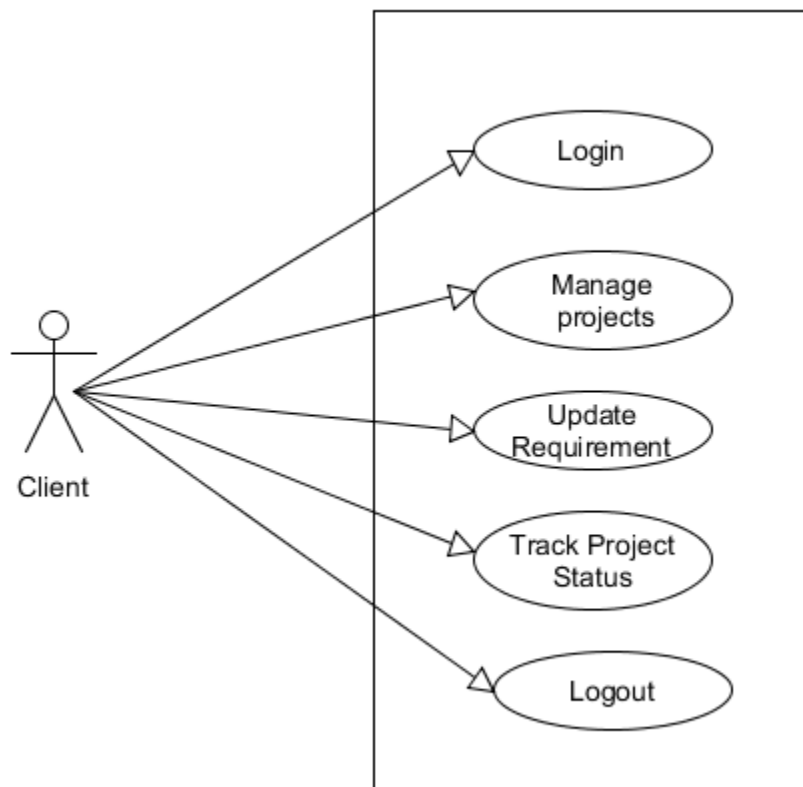
Based on the system requirement the software developer can create a set of scenarios called use case. Use case provides a description of how the system will be used. The user interface is highly by the user interaction with the system. Hence use case are used as basic for development of the interface. Use case is created by identifying the different types of people called actors that use the system. These actors actually represent roles that user's play the system operates an actor is anything that communicates with system.

Each use case provides an unambiguous scenario of interaction between an actor and the software. The actors in the process are represented as stick figures and class of interactions is represented as a named ellipse. The set of use case represents all of the possible interaction that will be represented in the system requirements.

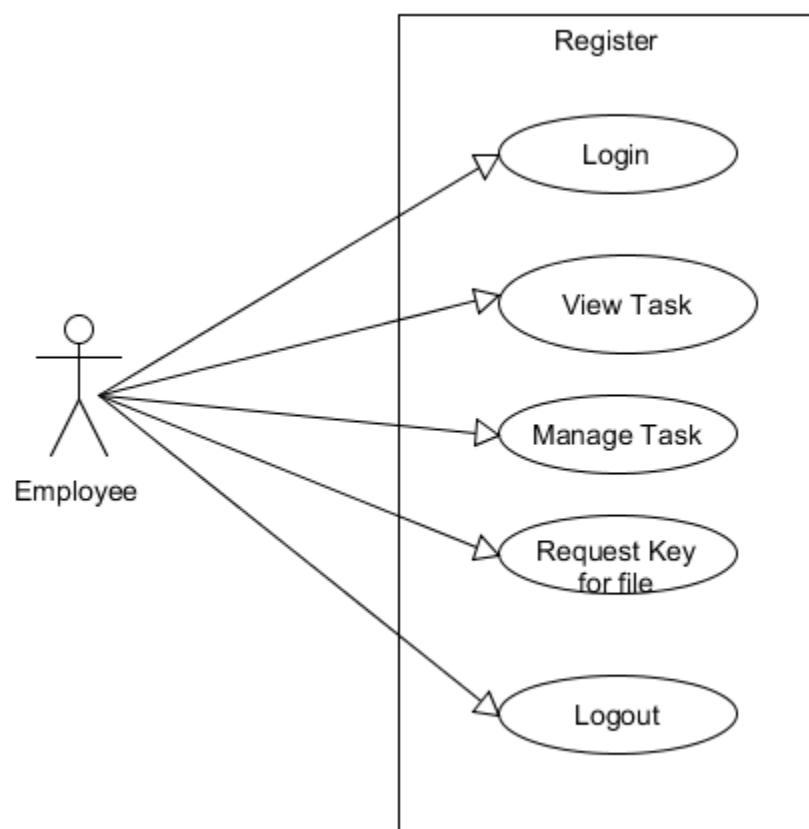
4.3.1. Use case Diagram for Admin



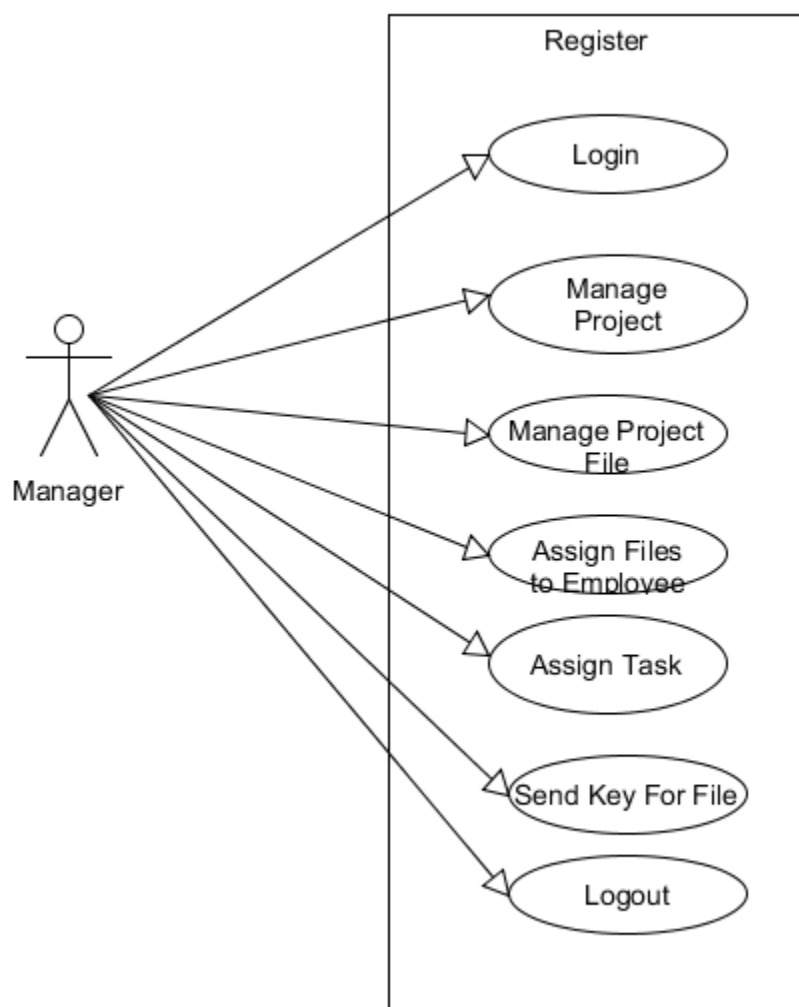
4.3.2 Use case Diagram for Client



4.3.3 Use case Diagram for Employee



4.3.4 Use case Diagram for Manager



4.4 DATABASE DESIGN

In order to develop this Multifactor authentication key exchange application we need to have a sophisticated and reliable database. The database plays a very important and curtail role in the application as we extensively access and manipulate the data that are to be stored or retrieved from the database. We in this application opted for the SQL database that is a part for the .NET framework. This database is faster to access the data because it is supported by the .NET framework and more importantly during the time of the deployment of the application this temporary database as be converted or exported to the any type of the database that exists without any difficulty.

Database design is a process of creation of database. Proper database design is absolutely vital to the success of the database and any application using the database. Without designing correctly, the following problems can be introduced into the database:

- Data becomes redundant and a large burden is placed on the individual applications to keep the data in synchronization.
- Data integrity is compromised because integrity constraints cannot be designed or implemented correctly.
- Performance is affected because additional queries might be required to complete a select operation.

The primary activity during database design is to select logical representation of the data objects identified during required analysis and software analysis. The data dictionary explicitly represents the relationships among the data objects and the constraints on the elements of the data structure.

A database design is prepared without problem of redundancy and follows normalization. The data is planned to be stored in the format that is supported by the SQL. The database is accessed at the front end with the help of ASP.NET.

The data that has to be stored is identified and the functional dependencies between various fields are analyzed. The relations between entities are identified.

The goal of a relational database is to generate a set of relational schema without the above-mentioned anomalies. One such approach is to design schema that are in an appropriate normal form. This is known as Normalization.

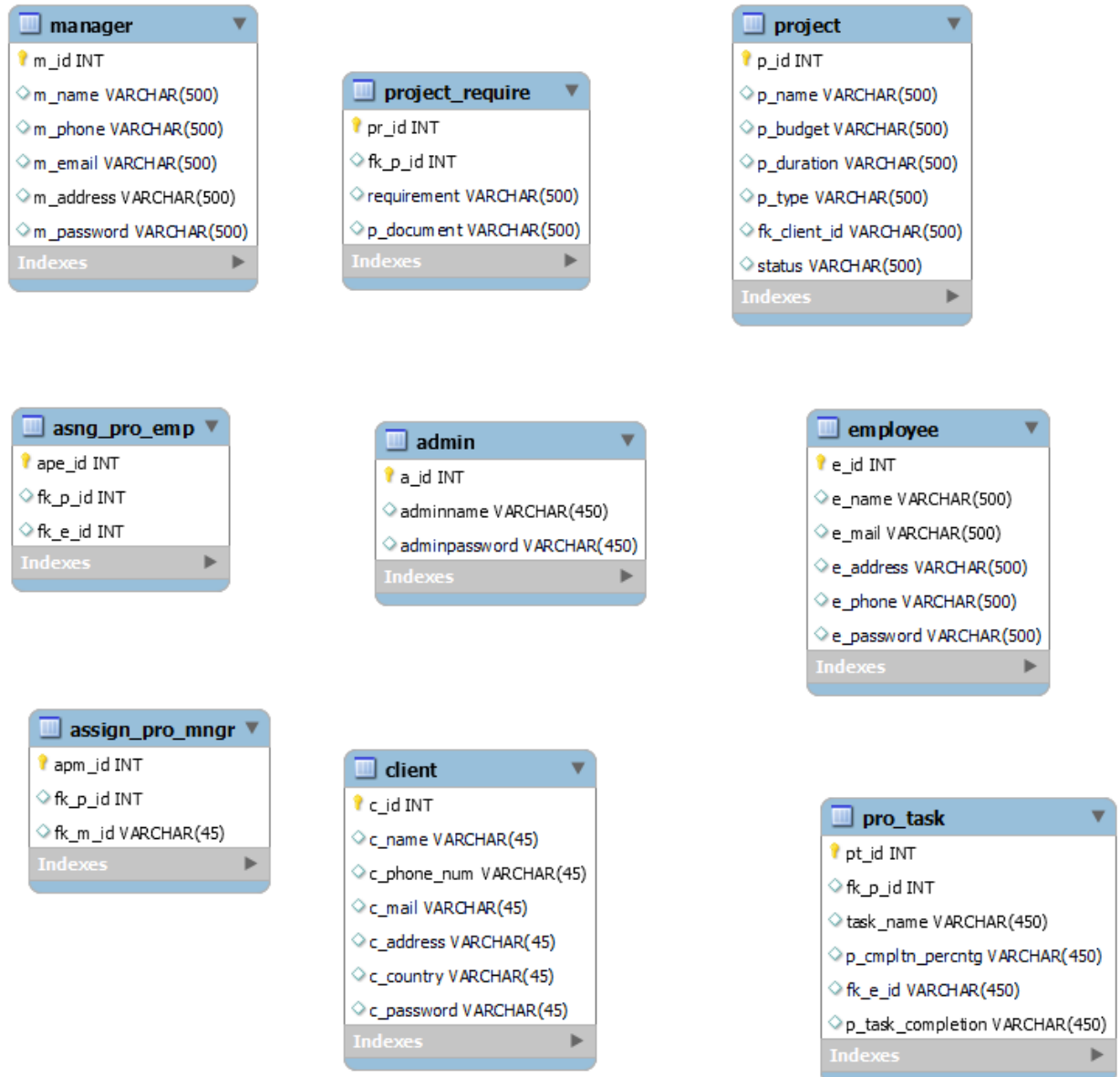


Fig 5.2.1 Database design

4.4.1 Data Normalization

Normalization is a process in which we eliminate the problem of data redundancy in the database design and build a data model that supports different functional requirements and alternate database designs.

Normalization is a successive process, which involves the following steps:

- All attributes must be single valued. This is referred to as first normal form. This means that the attribute data is not repeated.
- An attribute must depend upon its entity's entire unique ID. This is referred as second normal form. Move the attributes to a table where they depend upon the primary key. Do not have a table that has attributes that don't solely depend upon the key.
- Non-unique ID attributes can be dependent upon another non-unique ID attribute. This is referred to as third normal form.
- When normalization is complete, each table must have exactly one primary key, and the data in the table is dependent solely upon the table's primary key.

4.5 ENTITY RELATIONSHIP DIAGRAM

An Entity-Relationship Diagram is model that identifies the concept or entities that exist in a system and the relationship between those entities. An ERD is often used as a way to visualize a relational database: each entity represents a database table, and the relationship lines represent the keys in one table that point to specific records in related tables. ERDs may also be more abstract, not necessarily capturing every table needed within a database, but serving to diagram the major concepts and relationships.

This ERD is of the type, intended to present an abstract, Theoretical view of the major entities and relationships needed for management of RDL resources. It may assist the database design process for RDL, but does not identify every table that would be necessary for RDL database.

- E R models are widely used in database design.
- E R model are required to be supplemented with detailed description of the entities, relationship and attributes included in the model.

An entity relationship diagram is a data modelling technique that creates a graphical representation of the entities and the relationships between entities, within an information system. There are three basic elements in ER models.

- Entities are the “things” about which we seek information.
- Attributes are the data we collect about the entities.
- Relationships provide the structure needed to draw information from multiple entities.

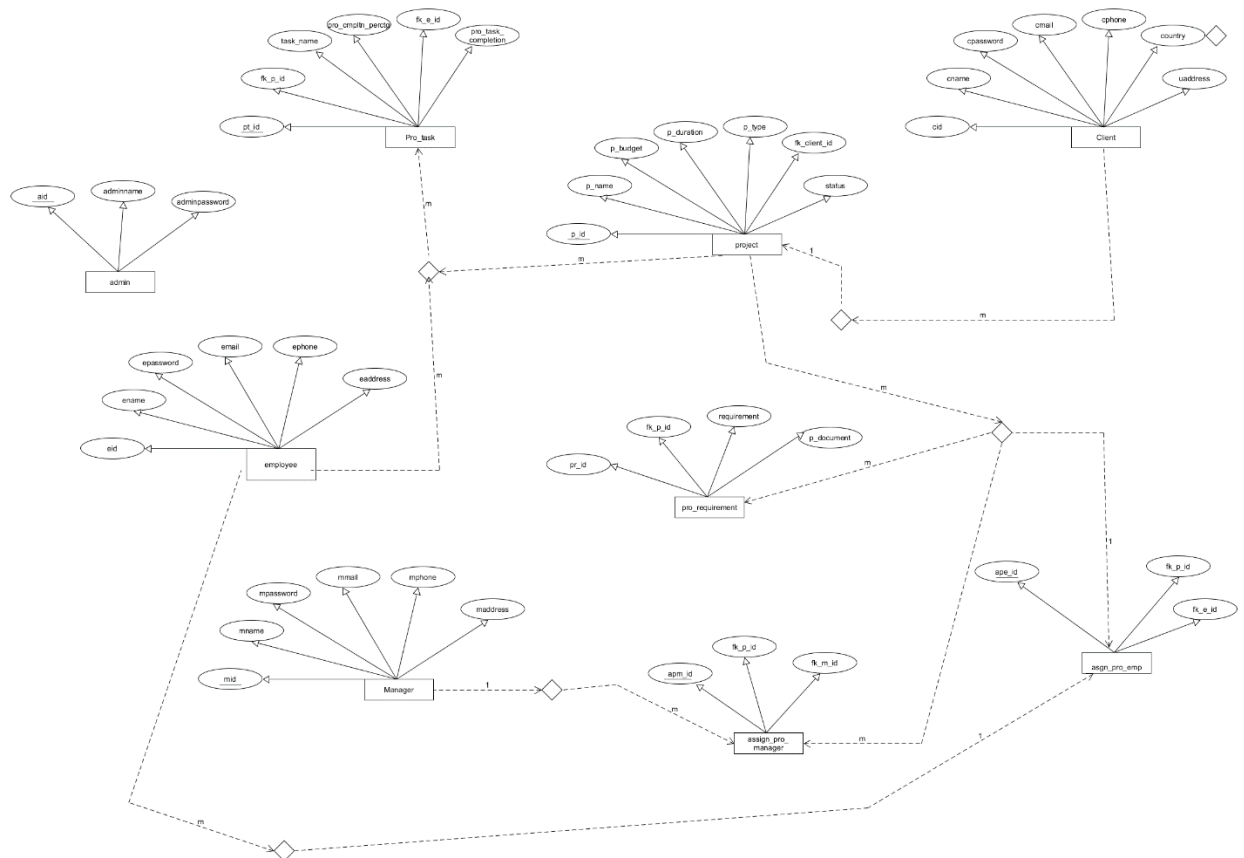


Fig 5.3.1 entity relationship diagram

4.6 DATABASE TABLES

The Database design comprises of 9 tables each having a table name along with the names of the fields. The system is designed to be effective, complete and normalized; the following are the database tables:

Column Name	Datatype
 a_id	INT
 adminname	VARCHAR(450)
 adminpassword	VARCHAR(450)

Table 5.4.1 table of Admin








Column Name	Datatype
 c_id	INT
 c_name	VARCHAR(45)
 c_phone_num	VARCHAR(45)
 c_mail	VARCHAR(45)
 c_address	VARCHAR(45)
 c_country	VARCHAR(45)
 c_password	VARCHAR(45)

Table 5.4.2 table of Client






Column Name	Datatype
 e_id	INT
 e_name	VARCHAR(500)
 e_mail	VARCHAR(500)
 e_address	VARCHAR(500)
 e_phone	VARCHAR(500)
 e_password	VARCHAR(500)

Table 5.4.3 table of Employee







Column Name	Datatype
 m_id	INT
 m_name	VARCHAR(500)
 m_phone	VARCHAR(500)
 m_email	VARCHAR(500)
 m_address	VARCHAR(500)
 m_password	VARCHAR(500)

Table 5.4.4 table of Manager








Column Name	Datatype
 p_id	INT
 p_name	VARCHAR(500)
 p_budget	VARCHAR(500)
 p_duration	VARCHAR(500)
 p_type	VARCHAR(500)
 fk_client_id	VARCHAR(500)
 status	VARCHAR(500)

Table 5.4.5 table of Project







Column Name	Datatype
 pt_id	INT
 fk_p_id	INT
 task_name	VARCHAR(450)
 p_cmplt_n_percentg	VARCHAR(450)
 fk_e_id	VARCHAR(450)
 p_task_completion	VARCHAR(450)

Table 5.4.6 table of Project task

Column Name	Datatype
 pr_id	INT
 fk_p_id	INT
 requirement	VARCHAR(500)
 p_document	VARCHAR(500)

Table 5.4.7 table of Project Requirement


Column Name	Datatype
 ape_id	INT
 fk_p_id	INT
 fk_e_id	INT

Table 5.4.8 table of Assign Project to Employee



Column Name	Datatype
 apm_id	INT
 fk_p_id	INT
 fk_m_id	VARCHAR(45)

Table 5.4.9 table of Assign Project to Manager

CHAPTER 5

IMPLEMENTATION

Once the design is completed, most of the major decision about the system has been made. The goal of the implementation phase is to translate the design of the system into code in a given programming language. For a given design, the aim in the phase is to implement the design in the best possible manner. A well-written code reduces the testing and maintenance costs of software are much higher than the coding cost. The goal of coding should be to reduce the testing and maintenance effort. Hence, during coding, the focus should be on developing programs that are easy to write. The simplicity and clarity should be maintained during the coding phase.

An important concept that helps the understandability of program is structured programming. The goal of structured programming is to arrange the control flow in the program i.e., program text should be organized as a sequence of statements and during execution, and the statements are executed in the sequence in the program. For structured programming a few single-entry-single-exit constraint should be used. These constraints include selection (if-then-else) and iteration (while-do, repeat-until, etc.).

5.1 PROJECT CODE

Admin Login

```
public class AdminLoginController : Controller  
  
    {  
  
        //  
  
        // GET: /AdminLogin/
```

```
public ActionResult Index()

{

    return View();

}


[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Index(Admin objadmin)

{

    if (ModelState.IsValid)

    {

        using (HRMSEntities db = new HRMSEntities())

        {

            try

            {

                var adminDetails = new List<Admin>();

                adminDetails = db.Admins.Where(a => a.Admin_UName ==

objadmin.Admin_UName && a.AdminPass == objadmin.AdminPass).ToList();
```



```
        if (adminDetails != null)

        {

            Session["AdminID"] = adminDetails[0].PK_Admin_id;

            Session["AdminName"] = adminDetails[0].Admin_UserName;


            return RedirectToAction("Dashboard", "AdminDashboard");

        }

    }

    catch (Exception ex)

    {

    }

}

}

return View(objadmin);

}
```

```
}
```

Client Registration

```
public ActionResult Register(HttpPostedFileBase file_Uploader, CLIENT objclient)
```

```
{
```

```
    if (file_Uploader != null)
```

```
    {
```

```
        string fileName = string.Empty;
```

```
        string fileext = string.Empty;
```

```
        string tempfileName = string.Empty;
```

```
        string destinationPath = string.Empty;
```

```
        tempfileName = "AuthImage" +  
System.DateTime.Now.ToString("ddMMyyyyhhmmss") +  
System.IO.Path.GetExtension(file_Uploader.FileName);
```

```
        tempfileName = removeillegalcharacters(tempfileName);
```

```
destinationPath = Path.Combine(Server.MapPath("~/UploadedFiles/"), tempfileName);
```

```
file_Uploader.SaveAs(destinationPath);
```

```
string clntname = objclient.CLIENT_NAME;
```

```
string clntcnty = objclient.CLIENT_COUNTRY;
```

```
string clntmail= objclient.CLIENT_MAIL;
```

```
string clntphn = objclient.CLIENT_PHONE;
```

```
string clntpass1 = objclient.CLIENT_PASS1;
```

```
string clntpass2 = Request.Form["sq"];
```

```
string clntpass2ans = objclient.CLIENT_ANS;
```

```
string clntpass3 = tempfileName;
```

```
if (ModelState.IsValid)
```

```
{
```

```
    CLIENT clnt = new CLIENT();
```

```
    clnt.CLIENT_NAME = clntname;
```

```
    clnt.CLIENT_COUNTRY = clntcnty;
```

```
clnt.CLIENT_MAIL = clntmail;
```

```
clnt.CLIENT_PHONE = clntphn;
```

```
clnt.CLIENT_PASS1 = clntpass1;
```

```
clnt.CLIENT_PASS2 = clntpass2;
```

```
clnt.CLIENT_ANS = clntpass2ans;
```

```
clnt.CLIENT_PASS3 = clntpass3;
```

```
clnt.CLIENT_STATUS = true;
```

```
db.CLIENTs.Add(clnt);
```

```
db.SaveChanges();
```

```
Session["Success"] = "Success";
```

```
return RedirectToAction("Register");
```

```
}
```

```
}
```

```
else
```

```
{
```

```
}
```

```
        return View(objclient);  
    }  
}
```

Holiday

```
public ActionResult Index()  
{  
    if (Session["AdminID"] == null)  
    {  
        return RedirectToAction("index", "Home");  
    }  
    else  
    {  
        var objholi = db.HOLIDAYs.Where(a => a.HOLIDAY_ISACTIVE == true);  
        return View(objholi.ToList());  
    }  
}
```

[HttpPost]

[ValidateAntiForgeryToken]

```
public ActionResult Create(HOLIDAY objholi)

{

    string hy = Request.Form["holiyear"];

    string holiname = objholi.HOLIDAY_NAME;

    string hf = Request.Form["holifrom"];

    string ht = Request.Form["holito"];

    string aia = Request.Form["activeinactive"];


    DateTime hofr = DateTime.ParseExact(hf, "dd/MM/yyyy", null);

    DateTime hoto = DateTime.ParseExact(ht, "dd/MM/yyyy", null);


    if (ModelState.IsValid)

    {

        bool contactExists = db.HOLIDAYs.Any(a => a.HOLIDAY_YEAR.Equals(hy) &&
a.HOLIDAY_NAME.Equals(holiname));

        if (contactExists == true)
```

```
{  
  
    Session["exist"] = "Success";  
  
}  
  
else  
  
{  
  
  
  
  
  
  
  
  
  
    HOLIDAY holi = new HOLIDAY();  
  
  
  
  
    holi.HOLIDAY_YEAR = hy;  
  
    holi.HOLIDAY_NAME = holiname;  
  
    holi.HOLIDAY_FROM = DateTime.ParseExact(hf, "dd/MM/yyyy", null);  
  
    holi.HOLIDAY_TO = DateTime.ParseExact(ht, "dd/MM/yyyy", null);  
  
  
  
  
  
  
  
  
  
    if (aia == "on")  
  
    {  
  
  
        holi.HOLIDAY_STATUS = true;  
  
    }  
  
    else
```

```
{  
  
    holi.HOLIDAY_STATUS = false;  
  
}  
  
holi.HOLIDAY_ISACTIVE = true;  
  
db.HOLIDAYs.Add(holi);  
  
db.SaveChanges();  
  
Session["Success"] = "Success";  
  
return RedirectToAction("Index");  
}  
  
}  
  
return View();  
}
```


Department

```
public ActionResult Index()

{

    if (Session["AdminID"] == null)

    {

        return RedirectToAction("index", "Home");

    }

    else

    {

        return View(db.DEPTs.ToList());

    }

}
```

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create(DEPT dept)

```
{

    if (ModelState.IsValid)

    {
```

```
string deptID = dept.DEPT_NAME;
```

```
bool contactExists = db.DEPTs.Any(a => a.DEPT_NAME.Equals(deptID));
```

```
if (contactExists == true)
```

```
{
```

```
    Session["exist"] = "Success";
```

```
}
```

```
else
```

```
{
```

```
    db.DEPTs.Add(dept);
```

```
    db.SaveChanges();
```

```
    Session["Success"] = "Success";
```

```
    return RedirectToAction("Index");
```

```
    }  
  
    }  
  
    return View(dept);  
  
    }  
  
    //  
  
    // GET: /Department/Edit/5  
  
    public ActionResult Edit(int id = 0)  
    {  
        if (Session["AdminID"] == null)  
        {  
            return RedirectToAction("index", "Home");  
        }  
        else  
        {  
  
            DEPT dept = db.DEPTs.Find(id);  
  
            if (dept == null)
```

```
{  
  
    return HttpNotFound();  
  
}  
  
return View(dept);  
  
}  
  
}
```

[HttpPost]

[ValidateAntiForgeryToken]

```
public ActionResult Edit(DEPT dept)  
  
{  
  
    if (ModelState.IsValid)  
  
    {  
  
        db.Entry(dept).State =  
(System.Data.Entity.EntityState)System.Data.EntityState.Modified;  
  
        db.SaveChanges();  
  
        Session["Edit"] = "Success";  
  
        return RedirectToAction("Index");  
  
    }  
  
    return View(dept);  
  
}
```

```
}

//

// GET: /Department/Delete/5

public ActionResult Delete(int id = 0)

{

    if (Session["AdminID"] == null)

    {

        return RedirectToAction("index", "Home");

    }

    else

    {

        DEPT dept = db.DEPTs.Find(id);

        if (dept == null)

        {

            return HttpNotFound();

        }

        return View(dept);

    }

}
```

CHAPTER 6

TESTING

Testing is the major process involved in software quality assurance. Here test data is prepared and is used to test the modules individually. System testing makes sure that all components of the system function properly as a unit by actually forcing the system to fail.

The test cases should be planned before testing begins. Then as the testing progresses, testing shifts focus in an attempt to find errors in integrated clusters of modules and in the entire system. The philosophy behind testing is to find errors. Actually testing is the estate of implementation that is aimed at ensuring that the system works actually and efficiently before implementation.

The Testing phase involves the testing of individual program units and the functionality with various test data. Preparation of the test data plays a vital role in the system testing. After preparing the test data the system under study was tested using those test data. A series of tests were performed on the developed system before implementing the same. The various types of testing done on the system were

- Unit Testing
- Integration Testing
- Validation Testing

6.1 Unit Testing

Unit testing is a procedure used to validate that a particular module of source code is working properly. The procedure is to write test cases for all functions and methods so that whenever a change causes a regression, it can be quickly identified and fixed. Ideally, each test case is separate from the others. This type of testing is mostly done by the developers and not by

end-users the goal of unit testing is to isolate each part of the program and show that the individual parts are correct. Unit testing provides a strict, written contract that the piece of code must satisfy.

6.2 Integration Testing

Integration testing is the phase of software testing in which individual software modules are combined and tested as a group. It follows unit testing and precedes system testing. Integration testing takes as its input modules that have been checked out by unit testing, groups them in larger aggregates, applies tests defined in an Integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

6.3 Validation Testing

At the Culmination of integration testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected, and a final series of software tests, validation testing was carried out. 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 expected by the customer/user. Deviation of errors discovered at this step n this project with the help of users by negotiating to establish a method for resolving deficiencies.

Thus the proposed system under consideration has been tested by using validation testing and found to be working satisfactorily.

6.4 Test Cases

Sl.No	Description	Expected Result	Final Result
TC_01	Login	Enter invalid username & password Solution: Enter valid username & password	Test Failed Retest Passed
TC_02	After Successful login	If Admin home page is display	Test Passed
	In Admin Module		
TC_03	In Client	If anyone filled is not filled Solution: Both the fields are need to be filled	Test Failed Retest passed
TC_04	In Project description	If any gives character Solution: All mandatory fields are filled	Test Failed Retest Passed
TC_05	While adding client details	First name is not filled Solution: Fill all the mandatory fields.	Test Failed Retest Passed


TC_06	While deleting employee	If employee is not selected Solution: Chose the users which is handling	Test Failed Retest Passed
TC_07	While adding project details	project name is not filled Solution: Fill all the mandatory fields.	Test Failed Retest Passed
TC_08	While deleting project	project is not selected Solution: Choose project first and then delete project	Test Failed Retest Passed
TC_09	In manage employee	If the username is already registered Solution: Give the available username	Test Failed Retest Passed
	In user module		
TC_10	While saving the employee details	If employee is not selected and press save Solution: Select all the fields from dropdown menu and press save	Test Failed Retest Passed

TC_11	While saving client country details	If client country is not entered Solution: Enter all the necessary fields	Test Failed Retest Passed
TC_12	While popup window appear	If the popup window not appear Solution: Check out the date in the system	Test Failed Retest Passed
TC_13	While login to employee if the name is in uppercase and given lowercase	employee is not opened Solution: type name & password and click login	Test Failed Retest Passed
TC_14	If file is in different format	Must show invalid file format Solution: use default image format only	Test Failed Retest Passed

CHAPTER 7

Snapshots

Admin Login :



Username

Password

LOG IN

HOME

Client Login :



Mail

Password

Select Security Question

Select Security Question

Answer Security Question

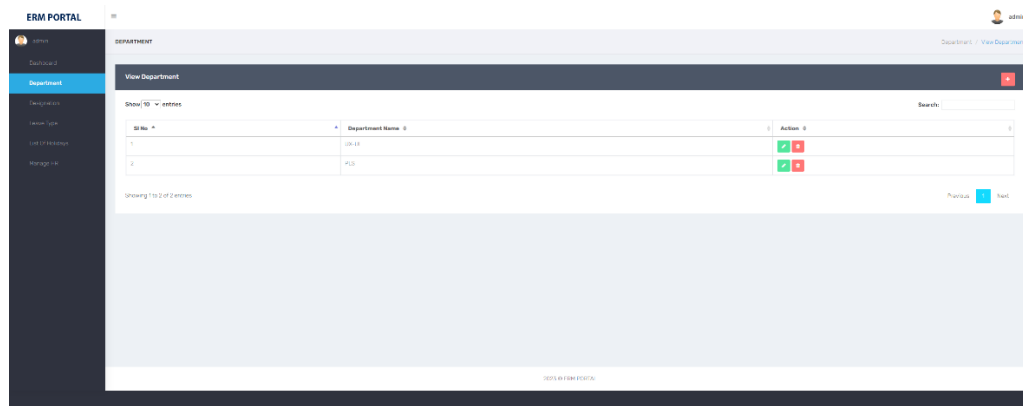
Select Your Security Image

☐  ☐  ☐ 


LOG IN

HOME

View Department :



Client Registration :



Name

Country

Mail

Phone

Password

Select Security Question

Select Security Question

Security Answer

Security Image

No file chosen

CONCLUSION

System developed has been tested with sample data along with verification and validation of all the fields on the respective screens. These modules can be further changed for enhanced features without any major difficulty Suggestions for improvements and both positive and negative comments are always welcome

BIBLIOGRAPHY

Books Referred:

- [1]. “Andrew Troelsen” C# and the .NET Platform, Second Edition, 2003, DreamTech,
Press, India
- [2]. “Jesse Liberty”, Programming ASP.NET, Dan Hurwitz Published 2005 O'Reilly
- [3]. “Ian Sommerville”, Software Engineering, Sixth Edition, Pearson Education Ltd,
2001

URL's:

Customer Service Definition-<http://www.customerservicemanager.com>

www.ActiveXpertsSoftware.com

<http://www.searchblaster.com/followup!execute.jspa>

<http://en.wikipedia.org/wiki/Portal:Technology>