

MAIL-SERVER DATABASE



Thesis/Dissertation submitted in the partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY in CSE (DATA SCIENCE)

By

P. Naga Sriya	21K91A6792
Y. Gowtham	22K95A6713
B. Shiva	21K91A67B3

**Under the guidance of
Mrs.K.Sneha Latha
Asst.Professor**

**DEPARTMENTMENT OF CSE (DATA SCIENCE)
TKR COLLEGE OF ENGINEERING &TECHNOLOGY
(AUTONOMOUS)
(Accredited by NAAC with 'A+' Grade)
Medbowli, Meerpet, Saroornagar, Hyderabad-500097**



TKR COLLEGE OF ENGINEERING & TECHNOLOGY

AUTONOMOUS

DECLARATION BY THE CANDIDATES

We, **Ms. P.Naga Sriya** bearing Hall Ticket Number: **21K91A6792**, **Mr. Y.Gowtham** bearing Hall Ticket Number: **22K95A6713**, **Mr. B.Shiva** bearing Hall Ticket Number: **21K91A67B3** hereby declare that the minor project report titled **Mail Server Database** under the guidance of **Mrs.K.Sneha Latha**, Assistant professor in Department of Computer Science & Engineering is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in CSE(Data Science)**.

P.Naga Sriya	21K91A6792
Y.Gowtham	22K95A6713
B. Shiva	21K91A67B3



TKR COLLEGE OF ENGINEERING & TECHNOLOGY

AUTONOMOUS

CERTIFICATE

This is to certify that the project report entitled “**Mail-Server Database**”, being submitted by **Ms.P.Naga Sriya**, bearing **Roll.No.:21K91A6792**, **Mr.Y.Gowtham**, bearing **Roll.No.:22K95A6713**, **Mr.B.Shiva** bearing **Roll.No.:21K91A67B3** in partial fulfillment of requirements for the award of degree of **Bachelor of Technology in CSE(Data Science)**, to the TKR College of Engineering & Technology is a record of bonafide work carried out by them under my guidance and supervision.

Signature of the Guide
Mrs.K.Sneha Latha
Asst.Professor

Signature of the HOD
Dr.V.Krishna
Professor

ACKNOWLEDGMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowned our efforts with success.

We express our sincere gratitude to **Management of TKRCET** for granting permission and giving inspiration for the completion of the project work.

Our faithful thanks to our **Principal Dr. D. V. Ravi Shankar, M.Tech., Ph.D.**, TKR College of Engineering & Technology for his Motivation in studies and completion of the project work.

With our heart full pleasure we thank our **Head of the Department Dr.V.Krishna, M.Tech., Ph.D.**, Professor, Department of CSE (Data Science), TKR College of Engineering & Technology for his suggestions regarding the project.

Thanks to our Project Coordinator **Mr.M.Arokia Muthu M.E.,(Ph.D.)**, Assistant Professor, Department of CSE (Data Science), TKR College of Engineering & Technology for his constant encouragement.

We are indebted to the **Internal Guide, Mrs.K.Sneha Latha**, Designation, Assistant Professor, Department of CSE (Data Science), TKR College of Engineering & Technology for his support in completion of the project successfully.

Finally, we express our thanks to one and all that have helped us in successfully completing this project. Furthermore, we would like to thank our family and friends for their moral support and encouragement.

	By,
P.Naga Sriya	21K91A6792
Y.Gowtham	22K95A6713
B.Shiva	21K91A67B3

CONTENTS

Abstract	i
List of Figures	ii
List of Tables	iii
List of Screens	iv
Symbols & Abbreviations	v

S.No.	Topic name	Pg.No.
1.	INTRODUCTION	1
	1.1 Problem definition	2
	1.2 Limitations of existing system	3
	1.3 Proposed system	4
2.	LITERATURE SURVEY	5
3.	REQUIREMENTS ANALYSIS	11
	3.1 Functional Requirements	11
	3.2 Non-Functional Requirements	11
4.	DESIGN	12
	4.1 DFDs & UML diagrams	12
	4.2 Class Diagram	15
	4.3 ER Diagram	16
	4.4 Tables	17
5.	CODING	18
	5.1 Pseudo Code	18
6.	IMPLEMENTATION & RESULTS	43
	6.1 Explanation of Key functions	43
	6.2 Method of Implementation	44
	6.2.1 Forms	44
	6.2.2 Output Screens	45
	6.2.3 Result Analysis	
7.	TESTING & VALIDATION	52
	7.1 Design of Test Cases and Scenarios	52
	7.2 Validation	52
	7.3 Conclusion	53
8.	CONCLUSION	54
	First Paragraph - Project Conclusion	54
	Second Paragraph - Future enhancement	54

ABSTRACT

The mail server database is a crucial element within the field of email communication, serving as the foundation upon which modern email systems are built. This database serves as the repository for an array of essential data, which includes user account information, email messages, folder organization and beyond . User account details including unique user names, email addresses, secure password storage, are essential for user authentication and personalization. Additionally, user preferences, contact information, and email settings are accurately maintained to enhance the user experience. Email messages, the core of email communication, find their digital abode within this database. Every email's metadata, such as sender and recipient information, timestamp, and status, is exactly recorded. Furthermore, the database securely houses the actual mail content, which encircle text, attachments, formatting and more. Organizing emails is made possible through the database's management of folders, which include default categories like Inbox and Sent Items, along with user generated custom folders. Sent and received messages are tracked and archived, facilitating the flow and organization of communication. The database's scope extends to confirm email security and prevent spam with records of block messages and security logs for monitoring purposes. User management data, such as permissions and activity logs, ensure controlled access and oversight. In an era of data retention and compliance, the database often supports archiving and backup, preserving emails for regulatory compliance and disaster recovery .its integration with the mail server facilitates seamless email operations, ensuring reliable message routing, delivery, and retrieval. Ultimately, the mail server database is an indispensable essential of email services, underpinning their reliability, security, and functionality.

LIST OF FIGURES

Fig No.	Title	Page No.
4.1	DFD Diagram	12
4.2	Use Case Diagram of login	13
4.3	Use Case Diagram of email system	14
4.4	Class Diagram	15
4.5	ER Diagram	16

Note:

Fig No. 4.1 implies 4 is chapter name and 1 is figure no in that chapter.

Fig No. 4.2 implies 4 is Chapter Name and 2 implies figure no in that chapter an so on

Fig No. 4.3 implies 4 is chapter name and 3 is figure no in that chapter and so on

Fig No. 4.4 implies 4 is chapter name and 4 is figure no in that chapter and so on

Fig No. 4.5 implies 4 is chapter name and 5 is figure no in that chapter.

LIST OF TABLES

Table No.	Title	Page No.
4.1	Login	17
4.2	Signup	17

Note:

Table No. 4.1 implies 4 is Chapter name and 1 is Table no in that chapter.

Table No. 4.2 implies 4 is Chapter Name and 2 implies Table no in that chapter an so on

Table No. 4.1 implies 4 is chapter name and 3 is Table no in that chapter and so on

Table No.4.1 implies 4 is chapter name and 4 is Table no in that chapter and so on.

LIST OF SCREENS

Screen No.	Title	Page No.
6.1	Home Page	45
6.2	Aboutus page	45
6.3	Our mission	48
6.4	Our vision	48
6.5	Our achievements	49
6.6	Signup page	49
6.7	Login page	50
6.8	Inbox page	50
6.9	Sent page	51
6.10	Received page	51
6.11	Spam page	52
6.12	Draft page	52
6.13	Contact us	53
6.14	Compose email	53

Note:

Screen No. 6.1 implies 6 is Chapter name and 1 is Screen no in that chapter and so on

Screen No. 6.2 implies 6 is Chapter Name and 2 is Screen no in that chapter and so on

Screen No. 6.3 implies 6 is chapter name and 3 is Screen no in that chapter and so on

Screen No. 6.4 implies 6 is Chapter name and 4 is Screen no in that chapter and so on

Screen No. 6.5 implies 6 is Chapter name and 5 is Screen no in that chapter and so on

Screen No. 6.6 implies 6 is Chapter name and 6 is Screen no in that chapter and so on

Screen No. 6.7 implies 6 is Chapter name and 7 is Screen no in that chapter and so on

Screen No. 6.8 implies 6 is Chapter name and 8 is Screen no in that chapter and so on

Screen No. 6.9 implies 6 is Chapter name and 9 is Screen no in that chapter and so on

Screen No. 6.10 implies 6 is Chapter name and 10 is Screen no in that chapter and so on

Screen No. 6.11 implies 6 is Chapter name and 11 is Screen no in that chapter and so on

Screen No. 6.12 implies 6 is Chapter name and 12 is Screen no in that chapter and so on

Screen No. 6.13 implies 6 is Chapter name and 13 is Screen no in that chapter and so on

Screen No. 6.14 implies 6 is Chapter name and 14 is Screen no in that chapter and so on

Chapter 1

INTRODUCTION

In the realm of digital communication, the mail server database stands as a vital pillar, underpinning the efficiency and reliability of email systems. This comprehensive repository manages a wide array of critical data, from user accounts and email messages to folder organization and security protocols. By ensuring seamless email operations and safeguarding user information, the mail server database plays an essential role in maintaining the functionality, security, and integrity of modern email services.

The mail server database serves as the repository for an array of essential data, which includes user account information, email messages, folder organization, and beyond. User account details, such as unique usernames, email addresses, and securely stored passwords, are fundamental for user authentication and personalization. Additionally, user preferences, contact information, and email settings are meticulously maintained to enhance the user experience.

Email messages, the core of email communication, reside within this database. Each email's metadata—such as sender and recipient information, timestamps, and status—is accurately recorded. Furthermore, the database securely stores the actual email content, including text, attachments, and formatting. Organizing emails is made possible through the database's management of folders, which include default categories like Inbox and Sent Items, along with user-generated custom folders. Sent and received messages are tracked and archived, supporting the flow and organization of communication. The database's scope extends to ensuring email security and preventing spam with records of blocked messages and security logs for monitoring purposes. User management data, such as permissions and activity logs, ensure controlled access and oversight. In an era of data retention and compliance, the database often supports archiving and backup, preserving emails for regulatory compliance and disaster recovery. Its integration with the mail server facilitates seamless email operations, ensuring reliable message routing, delivery, and retrieval.

Ultimately, the mail server database is an indispensable element of email services, underpinning their reliability, security, and functionality. By managing crucial data and ensuring efficient operations, it supports the robust performance of email systems, making it a cornerstone of modern digital communication.

EXISTING SYSTEM

The existing mail server database system is a sophisticated and essential component of email infrastructure, designed to manage, store, and organize vast amounts of data critical to the functioning of email services. It encompasses various elements, including user accounts, email messages, folder organization, security measures, and more, all of which work together to ensure seamless communication and data management. User account information, such as unique usernames, email addresses, and securely hashed passwords, is stored to facilitate user authentication and personalization. Along with this, user preferences, contact lists, and email settings are meticulously maintained to provide a customized and user-friendly experience.

At the core of the mail server database are the email messages. Each email's metadata—such as sender and recipient information, timestamps, and status—is accurately recorded. The database securely stores the actual content of emails, including text, attachments, and formatting. Efficient email organization is facilitated through the management of folders, including default categories like Inbox and Sent Items, along with user-generated custom folders. This structure helps users easily navigate and manage their emails. Security is a paramount concern, and the system maintains records of blocked messages and security logs for monitoring and preventing unauthorized access and spam. Advanced spam filters and security protocols protect users from malicious emails and phishing attempts, while user management data ensures controlled access and oversight.

Data retention and compliance are critical aspects of the mail server database system, supporting archiving and backup functionalities to preserve emails for regulatory compliance and disaster recovery purposes. This ensures that emails are retained according to legal requirements and can be recovered in case of data loss or corruption. The mail server database seamlessly integrates with the mail server to facilitate reliable message routing, delivery, and retrieval, ensuring efficient operations, high performance, and minimal downtime. By effectively managing user accounts, email storage, security, and compliance, the existing mail server database system supports the seamless operation of email systems and underpins the essential communication needs of modern users.

LIMITATIONS OF EXISTING SYSTEM

Despite its essential role in email infrastructure, the existing mail server database system has several limitations that impact its overall efficiency and reliability. One significant issue is scalability. As the volume of email traffic increases, the database can struggle to manage and store vast amounts of data effectively. This can lead to slower performance and longer retrieval times, particularly for organizations with large user bases or high email volumes. Additionally, scaling up the infrastructure to accommodate growth can be costly and complex, requiring significant investment in hardware and software resources.

Security is another critical limitation. While current systems implement various security measures to protect against unauthorized access and spam, they are not foolproof. Sophisticated phishing attacks and malware can sometimes bypass these defenses, leading to data breaches and compromised user accounts. Moreover, maintaining up-to-date security protocols and ensuring compliance with evolving regulations is a continuous challenge. This is especially true for smaller organizations that may lack the resources to implement and maintain robust security measures, leaving them vulnerable to cyber threats.

The existing mail server database system also faces challenges in terms of data redundancy and recovery. Although many systems support data archiving and backups, the process can be inefficient and prone to errors. In the event of a system failure or data corruption, restoring data can be time-consuming and may not always result in a complete recovery. Additionally, regular maintenance and updates are required to keep the system running smoothly, which can lead to downtime and disruption of email services. These limitations highlight the need for more advanced, resilient, and scalable solutions to meet the growing demands of modern email communication.

PROPOSED SYSTEM

The proposed mail server database system aims to address the limitations of the existing infrastructure by introducing enhanced scalability, security, and data management capabilities. To handle increasing volumes of email traffic efficiently, the new system will utilize cloud-based storage solutions. This approach allows for dynamic scaling, where storage and processing power can be adjusted according to demand, ensuring consistent performance even during peak usage periods. Cloud infrastructure also reduces the need for significant upfront investment in hardware, making it a cost-effective solution for organizations of all sizes.

Security enhancements are a primary focus of the proposed system. Advanced encryption techniques will be employed to protect data both at rest and in transit, ensuring that sensitive information remains secure. Multi-factor authentication (MFA) will be implemented to provide an additional layer of protection for user accounts, significantly reducing the risk of unauthorized access. Additionally, the system will incorporate real-time threat detection and response capabilities, leveraging artificial intelligence and machine learning to identify and mitigate potential security threats swiftly. Regular security audits and compliance checks will ensure that the system adheres to the latest regulatory standards and best practices.

To improve data redundancy and recovery, the proposed system will implement a robust backup and disaster recovery strategy. Automated, incremental backups will be performed regularly, ensuring that the most current data is always available for restoration. In the event of a system failure or data corruption, the use of distributed storage across multiple geographic locations will facilitate quick and reliable data recovery. This approach minimizes downtime and ensures that email services remain operational with minimal disruption. By integrating these advanced features, the proposed mail server database system aims to provide a more resilient, secure, and scalable solution that meets the evolving needs of modern email communication.

Chapter 2

LITERATURE REVIEW

2.1 REVIEW OF LITERATURE

History: The economic growth and political stability under the Mauryan Empire (322—185 BCE) stimulated sustained development of civil infrastructure in ancient India. The Mauryans developed early Indian mail service as well as public wells, rest houses, and other facilities for the public. Common chariots called *Dagana* were sometimes used as mail chariots in ancient India. Couriers were used militarily by kings and local rulers to deliver information through runners and other carriers. The postmaster, the head of the intelligence service, was responsible for ensuring the maintenance of the courier system. Couriers were also used to deliver personal letters.

In South India, the Wodeyar dynasty (1399—1947) of the Kingdom of Mysore used mail service for espionage purposes thereby acquiring knowledge related to matters that took place at great distances.

By the end of the 18th century, a postal system in India was in operation. Later this system underwent complete modernization when the British Raj established its control over most of India. The Post Office Act XVII of 1837 provided that the Governor-General of India in Council had the exclusive right of conveying letters by post for hire within the territories of the East India Company. The mails were available to certain officials without charge, which became a controversial privilege as the years passed. On this basis the Indian Post Office was established on October 1, 1837.

LITERATURE SURVEY - 1

Title: A performance study on Internet Server Provider mail servers.

Authors: J. Wang, Y. Hu

Description: This work presents a comprehensive performance study on Internet Service Provider (ISP) mail server, which plays an important role in Internet-based distributed computing. A group of e-mail messages will easily make a remote recipient server become overloaded.

Merits:

Comprehensive Approach: The study takes a holistic approach by using the SPECmail2001 benchmark to analyze both networking and I/O performance on an ISP mail server, providing a well-rounded understanding of its behavior.

Realistic Testing Environment: The research is conducted in a practical setting by utilizing a commercial mail server software system (MDaemon 5.0.1) and varying user populations from 200 to 10,000, enhancing the relevance and applicability of the findings to real-world scenarios.

Actionable Recommendations: The study not only identifies performance issues but also offers technical suggestions to improve ISP mail server performance, providing practical insights that can be applied to enhance overall system efficiency.

Demerits:

Limited Benchmark Details: The abstract lacks specific information about the SPECmail2001 benchmark, potentially limiting the transparency and reproducibility of the study for others in the field

Specific Software System Dependency: The reliance on a specific commercial mail server software system (MDaemon 5.0.1) may introduce bias, and the findings may not be directly applicable to other mail server software systems, limiting the generalizability of the results.

Incomplete Overload Analysis: While identifying the issue of remote recipient server overload, the abstract does not provide a detailed analysis of the factors causing the overload or offer nuanced solutions, leaving gaps in understanding the full scope of the problem.

LITERATURE SURVEY – 2

Title: A Prediction Mechanism of Mail Retrieval Based on the user Behaviour analysis for Electronic Mail Database System.

Authors: Wei-Ru Lai, Che-Hui Liao, Chang Ching Lu, Ming-Kuan Liao.

Description: Due to the frequent information exchange, electronic mail system is not only a mail exchange platform but also becomes an information center. Users usually save their mails permanently in mailboxes and retrieve them as need. In the database strategy, each mail is separated into several components and stored in the database individually. In our implementation experience, we find that the response time would be reduced and the system performance does not reduce in evidence.

Merits:

Efficient Database Strategy: The adoption of a database strategy for storing mail components allows for easy development of new applications, such as search and mobile retrieval.

Improved User Experience: The prediction mechanism implemented in the mail server anticipates the mails that users are likely to access, enabling the pre-reconstruction and caching of these mails. This proactive approach reduces the response time for mail retrievals, leading to a more efficient and responsive user experience.

Utilization of User Behavior Analysis: The prediction mechanism relies on user behavior collection and analysis, leveraging insights into how users interact with their emails.

Demerits:

Time-Consuming Recombination Process: While the prediction mechanism aims to reduce response times.

Dependency on User Behavior Analysis: The accuracy of the prediction mechanism heavily relies on the effectiveness of user behavior analysis

Dependency on User Behavior Analysis: The accuracy of the prediction mechanism heavily relies on the effectiveness of user behavior analysis. If the analysis is not thorough or if user behavior patterns change unpredictably, the predictions may become less accurate, impacting the overall performance of the mail server.

LITERATURE SURVEY – 3

Title: Database Design on Embedded home gateway and webserver implementation.

Authors: Guang Dong, Wei He, Yuhang Wang

Description: The paper under consideration delves into a comprehensive analysis of the database requirements within the realm of embedded development, with a specific focus on embedded Web servers. It meticulously examines the intricacies of database design, particularly utilizing the SQLite architecture, within the context of an embedded home gateway.

Merits:

Comprehensive Integration: The system benefits from a reliable and efficient data management solution. This integration facilitates a cohesive environment where the embedded Linux operating system and Web server can work in tandem, enhancing the overall functionality of the home gateway.

Optimized Resource Usage: The paper addresses the inherent resource constraints associated with embedded systems, demonstrating a focus on optimized resource usage.

Successful Web Server Porting: The successful porting of an embedded Web server onto the home gateway platform is a significant achievement.

Demerits:

Potential Complexity: Handling the intricacies of multiple software components may increase development and maintenance challenges. The complexity could potentially impact system reliability and ease of troubleshooting.

Limited Scalability: The use of SQLite, while advantageous for resource-constrained environments, may present limitations in terms of scalability

Dependency on Embedded Linux: The successful porting of an embedded Linux operating system implies a dependency on this specific platform

LITERATURE SURVEY – 4

Title: The Performance of a Bare Machine Email Server.

Authors: George H. Ford Jr., Ramesh K. Karne, Alexander L. Wijesinha, Patrick Appiah-Kubi

Description: Bare machine applications run directly over the hardware without using an operating system or a hard disk. The results indicate that the bare machine email server outperforms the conventional email servers in LAN and WAN environments, and demonstrate the capability of using bare machines to build high-performance email servers.

Merits:

Optimized Performance: The primary merit of the bare machine email server lies in its optimized performance. This optimization is particularly evident in LAN and WAN environments, highlighting the potential of bare machine applications for high-performance computing tasks like email server operations.

Reduced Overhead: Bare machine applications eliminate the need for an operating system layer and other software abstractions, resulting in reduced system overhead. This streamlined approach is especially beneficial for applications where minimal latency and quick response times are crucial, as demonstrated by the faster processing times in the study.

Scalability and Resource Efficiency: The study's focus on performance optimization indicates that bare machine email servers have the potential to efficiently utilize resources, making them well-suited for scenarios requiring scalability and resource-conscious operation.

Demerits:

Limited Functionality: One significant demerit of bare machine applications, including email servers, is their potential limitation in terms of functionality. Operating systems provide a range of services and abstractions that make it easier to develop complex applications.

Lack of Abstraction: This can result in challenges when porting the application to different systems or when trying to integrate additional features that rely on operating system services. The simplicity of the bare machine approach might limit its flexibility in certain scenarios.

Development and Maintenance Complexity This complexity can make the development process more challenging and require specialized knowledge. Additionally, maintaining the server might be more intricate, as updates and modifications could involve a deeper understanding of hardware nuances.

LITERATURE SURVEY – 5

Title: Concept and Design of things Mail System

Author: Neng-Geng Huang, Bing-Liang Zang

Description: This paper proposes a mail control system, which manage and control the delivery of physical mails based on the TCP/IP network. This mail system consist of mail servers and mail boxes, which store physical mails for users temporally and are supervised by mail servers through the Internet. In order to achieve the interoperation of mail systems, this paper proposes concepts of Universal Traceable Identifier (UTID) and Identifier Tracing Protocol (IDTP). All mail addresses and mail boxes are identified by UTIDs and all the communications between mail servers and mail boxes are based on IDTP protocol.

Merits:

Efficient Management: The system enables efficient management and control of physical mail deliveries by leveraging the TCP/IP network.

Traceability: The use of Universal Traceable Identifiers (UTIDs) and Identifier Tracing Protocol (IDTP) ensures all mail addresses and mailboxes can be tracked accurately.

Interoperability: The proposed system facilitates seamless interoperation between different mail systems.

Demerits:

Infrastructure Cost: Implementing the system requires significant investment in new infrastructure, including mail servers and network-enabled mailboxes.

Security Concerns: The integration with the internet raises potential security risks, such as unauthorized access and data breaches.

Complexity: Managing and maintaining such a system can be complex, requiring specialized knowledge and ongoing technical support.

Chapter 3

REQUIREMENT ANALYSIS

FUNCTIONAL REQUIREMENTS

This section of the document contains the different kinds of features used in making Gmail.com. It includes the features like signup, login, delete, compose, drafts etc

Login or Signup: This functionality is required to create an account or if you have one then, you can enter the Gmail by logging in

Compose: This functionality is used to send the email to the other users.

Spam Mails: This functionality is used to mark some suspicious mails as spam

Email Tabs: This functionality is used to differentiate between primary, social and promotional emails.

NON-FUNCTIONAL REQUIREMENTS

Non-Functional Requirement is a quality attribute of a software system. They evaluate the software system's responsiveness, usability, security, portability, and other nonfunctional characteristics that are critical to its success. Non-functional requirements must be specified with the same attention as

1. Usability requirement
2. Serviceability requirement
3. Security requirement
4. Data Integrity requirement
5. Capacity requirement
6. Availability requirement
7. Scalability requirement
8. Interoperability requirement
9. Reliability requirement
10. Maintainability requirement

Chapter 4

DESIGN

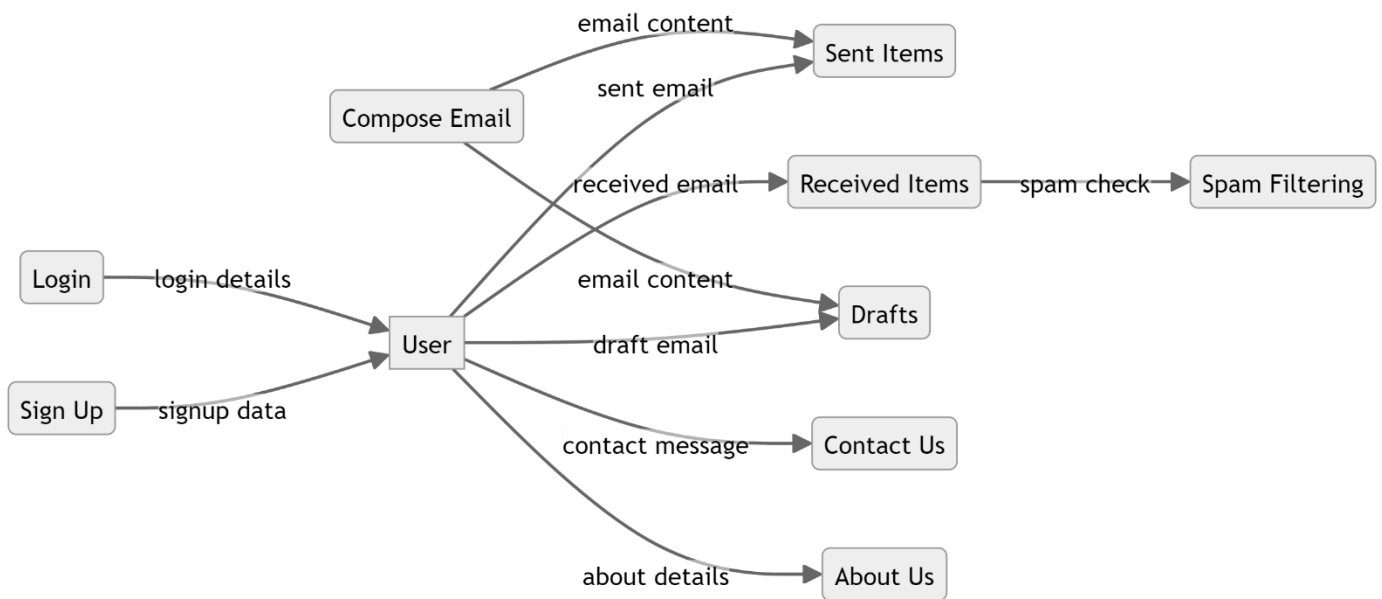


FIG 4.1 DFD Diagram

DFD shows the entities that interact with a system and defines the border between the system and its environment.

The illustration presents the main process in a single node to introduce the project context. This context explains how the project works in just one look. The user feeds data into the system and then receives the output from it.

Usecase Diagram:

A Use case is a description of set of sequence of actions. Graphically it is rendered as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object. Primary Actor – Sender, Secondary Actor.

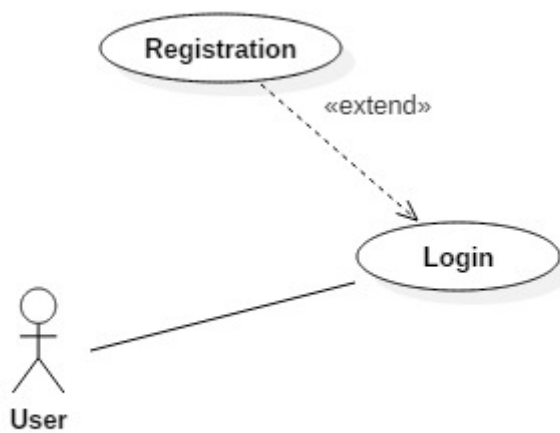


Fig 4.2 Use Case Diagram of login

Usecase Diagram:

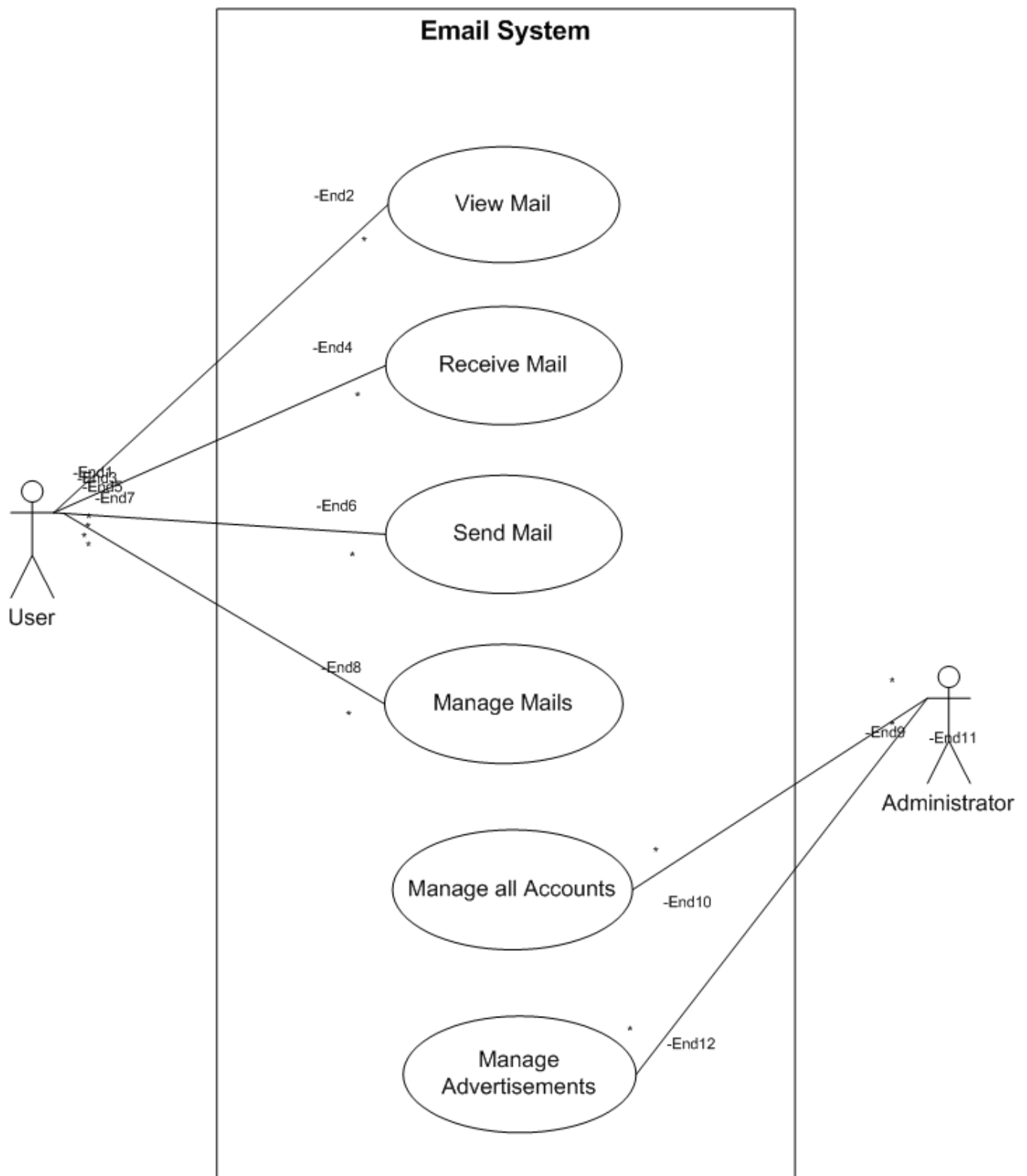


FIG 4.3 Use Case Diagram of email system

Class Diagram:

A description of set of objects that share the same attributes operations, relationships, and semantics.

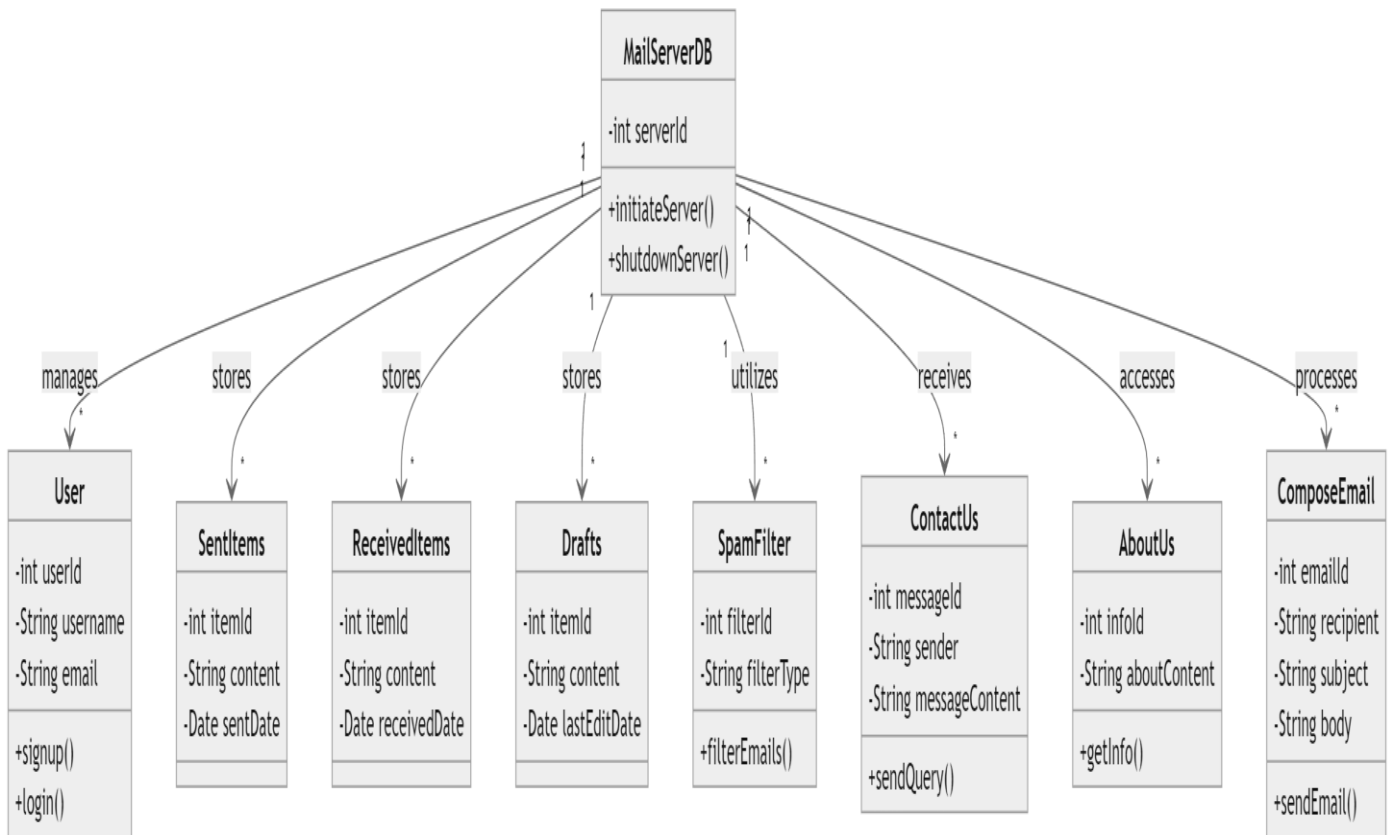


Fig 4.4 Class Diagram

ER Diagrams :

A database design is a collection of stored data organized in such a way that the data requirements are satisfied by the database

A collection of relative records make up a table. To design and store data to the needed forms database tables are prepared. Two essential settings for a database are:

1.Primary key: - The field that is unique for all the record occurrences

2.Foreign key: -The field used to set relation between tables. Normalization is a technique to avoid redundancy in the tables.

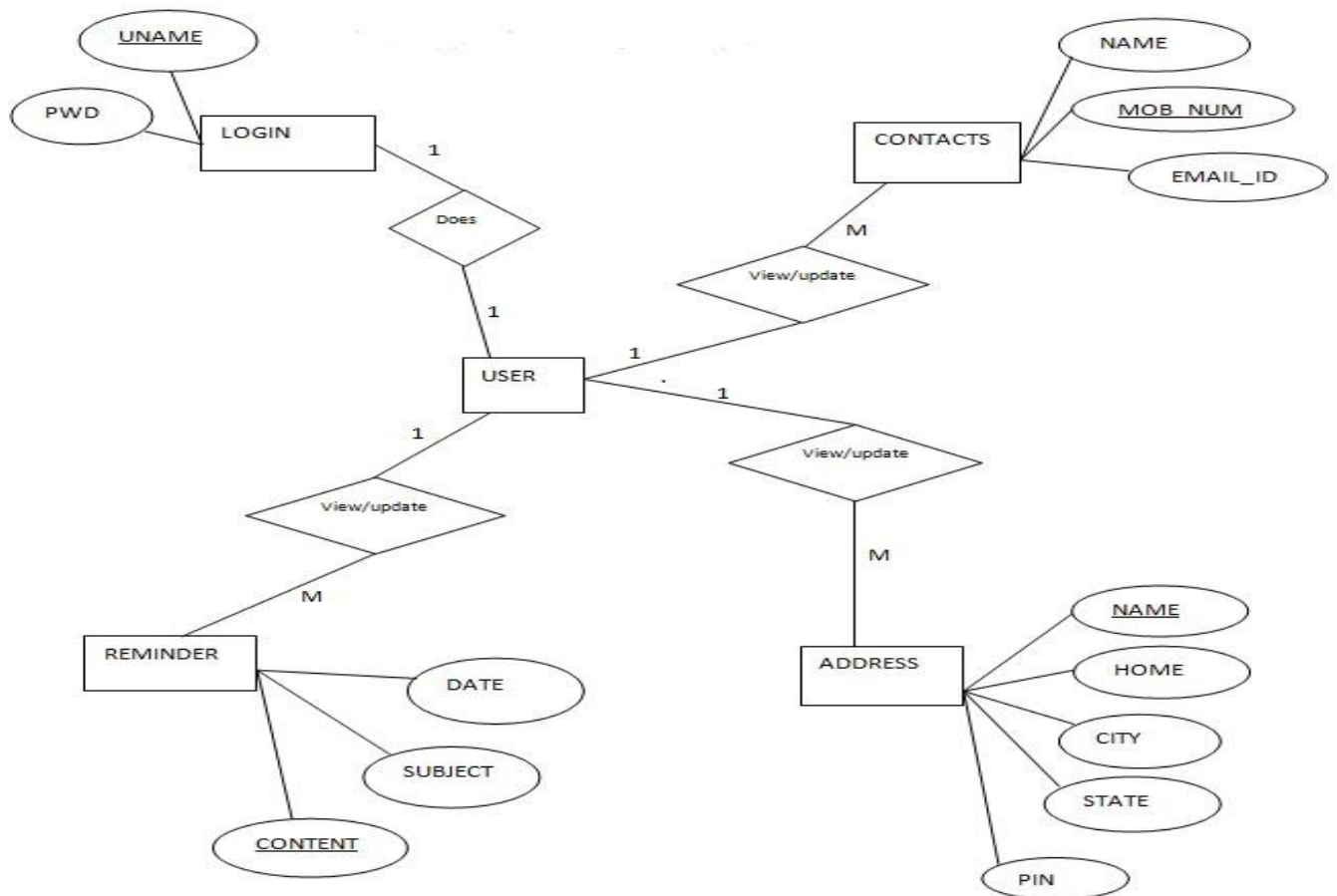


FIG 4.5 ER Diagram

MYSQL Data Tables:

Login Table :(Table name is admin)

This store user login details.














	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	id 	int(11)			No	None		AUTO_INCREMENT	 Change  Drop  More
<input type="checkbox"/>	2	username	varchar(50)	utf8mb4_unicode_ci		No	None			 Change  Drop  More
<input type="checkbox"/>	3	email	varchar(50)	utf8mb4_unicode_ci		No	None			 Change  Drop  More
<input type="checkbox"/>	4	password	varchar(50)	utf8mb4_unicode_ci		No	None			 Change  Drop  More

Table 4.1 : Login

Signup Table


	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/>	1	id 	int(11)			No	None		AUTO_INCREMENT
<input type="checkbox"/>	2	username	varchar(50)	utf8mb4_general_ci		No	None		
<input type="checkbox"/>	3	password	varchar(50)	utf8mb4_general_ci		No	None		
<input type="checkbox"/>	4	cpassword	varchar(50)	utf8mb4_general_ci		No	None		
<input type="checkbox"/>	5	date	datetime			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()

Table 4.2 : Signup

Chapter 5

CODING

Homepage.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Mail Server Database</title>

  <style>

    body {

      background-image: url('https://c4.wallpaperflare.com/wallpaper/870/756/865/4k-dark-blue-lines-
grid-lines-wallpaper-thumb.jpg');

      background-size: cover;

      color: #ffffff; /* text color */

      font-family: 'Arial', sans-serif;

      margin: 0;

      padding: 0;

    }

    header {

      text-align: center;

      padding: 20px;

      background-color: rgba(0, 0, 0, 0.5); /* semi-transparent background */

    }

    nav {

      text-align: center;
```

```
padding: 10px;

background-color: rgba(0, 0, 0, 0.7); /* semi-transparent background */
}
```

```
section {

    display: none;

    padding: 20px;
}
```

```
form {

    display: none;
}
```

```
form.active {

    display: block;
}
```

```
.module {

    display: none;

    background-color: rgba(7, 7, 7, 0.8); /* semi-transparent white background */

    border-radius: 10px;

    margin-top: 20px;

    padding: 15px;
}
```

```
.module.active {

    display: block;
}
```

```
</style>
```

</head>

<body>

<header>

<h1>Mail Server Database</h1>

</header>

<nav>

<button onclick="showAllModules()">Home</button>

<button onclick="showSection('about')">About Us</button>

<button onclick="showForm('signupForm')">Sign Up</button>

<button onclick="showForm('loginForm')">Login</button>

<button onclick="showModule('sentMailsModule')">Sent Mails</button>

<button onclick="showModule('receivedMailsModule')">Received Mails</button>

<button onclick="showSection('contactUs')">Contact Us</button>

</nav>

<section id="home">

<h2>Welcome to our Mail Server Database</h2>

<!-- Content for the home section -->

</section>

<section id="about">

<h2>About Us</h2>

<p>

Our Mail Server Database is a reliable and secure platform designed to manage your emails efficiently.

We provide a seamless experience for sending, receiving, and organizing your emails. Our goal is to ensure

the security and privacy of your communication while offering a user-friendly interface.

</p>

</section>

<section id="contactUs">

<h2>Contact Us</h2>

<!-- Content for the Contact Us section -->

<p>

If you have any questions or concerns, feel free to contact us at:

Email: info@mailserverdatabase.com

Phone: +1 (123) 456-7890

</p>

</section>

<form id="signupForm" class="active">

<h2>Sign Up</h2>

<!-- Sign Up Form Content -->

<label for="username">Username:</label>

<input type="text" id="username" name="username">

<label for="password">Password:</label>

<input type="password" id="password" name="password">

<label for="gender">Gender:</label>

<select id="gender" name="gender">

<option value="male">Male</option>

<option value="female">Female</option>

<option value="other">Other</option>

</select>

<label for="phoneNumber">Phone Number:</label>

<input type="tel" id="phoneNumber" name="phoneNumber" pattern="[0-9]{3}-[0-9]{3}-[0-9]{4}"
placeholder="123-456-7890">

<label for="email">Email:</label>

<input type="email" id="email" name="email">

<button type="submit">Sign Up</button>


```
</form>
```

```
<form id="loginForm">
```

```
  <h2>Login</h2>
```

```
  <!-- Login Form Content -->
```

```
  <label for="loginUsername">Username:</label>
```

```
  <input type="text" id="loginUsername" name="loginUsername">
```

```
  <label for="loginPassword">Password:</label>
```

```
  <input type="password" id="loginPassword" name="loginPassword">
```

```
  <button type="submit">Login</button>
```

```
</form>
```

```
<div id="sentMailsModule" class="module">
```

```
  <h2>Sent Mails</h2>
```

```
  <!-- Content for the Sent Mails module -->
```

```
</div>
```

```
<div id="receivedMailsModule" class="module">
```

```
  <h2>Received Mails</h2>
```

```
  <!-- Content for the Received Mails module -->
```

```
</div>
```

```
<script>
```

```
  function showSection(sectionId) {
```

```
    // Hide all sections
```

```
    document.querySelectorAll('section').forEach(function(section) {
```

```
      section.style.display = 'none';
```

```
    });
```

```

// Hide all forms

document.querySelectorAll('form').forEach(function(form) {

    form.classList.remove('active');

});


// Hide all modules

document.querySelectorAll('.module').forEach(function(module) {

    module.classList.remove('active');

});


// Show the selected section

document.getElementById(sectionId).style.display = 'block';
}


function showForm(formId) {

    // Hide all sections

    document.querySelectorAll('section').forEach(function(section) {

        section.style.display = 'none';

    });


    // Show the selected form

    document.querySelectorAll('form').forEach(function(form) {

        form.classList.remove('active');

    });


    // Hide all modules

    document.querySelectorAll('.module').forEach(function(module) {

        module.classList.remove('active');

    });

```

```

    document.getElementById(formId).classList.add('active');
}

function showModule(moduleId) {
    // Hide all sections
    document.querySelectorAll('section').forEach(function(section) {
        section.style.display = 'none';
    });

    // Hide all forms
    document.querySelectorAll('form').forEach(function(form) {
        form.classList.remove('active');
    });

    // Hide all modules
    document.querySelectorAll('.module').forEach(function(module) {
        module.classList.remove('active');
    });

    // Show the selected module
    document.getElementById(moduleId).classList.add('active');
}

function showAllModules() {
    // Show all sections
    document.querySelectorAll('section').forEach(function(section) {
        section.style.display = 'block';
    });
}

```

```
// Hide all forms

document.querySelectorAll('form').forEach(function(form) {

    form.classList.remove('active');

});


// Show all modules

document.querySelectorAll('.module').forEach(function(module) {

    module.classList.add('active');

});

}

</script>

</body>

</html>
```

Login.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Login - Mail Server Database</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            background: linear-gradient(to bottom, #000000, #333333);

            color: #ffffff;

            margin: 0;

            padding: 0;
```

```
    font-size: 18px;
}
```

```
header {
    text-align: center;
    padding: 20px;
    background-color: rgba(0, 0, 0, 0.5);
}
```

```
main {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}
```

```
form {
    padding: 20px;
    background-color: rgba(0, 0, 0, 0.8);
    border-radius: 10px;
}
```

```
h1, h2 {
    color: #fff;
    font-size: 28px;
}
```

```
label {
    display: block;
```

```
margin-bottom: 10px;

font-size: 18px;

}
```

```
input[type="text"],
input[type="email"],
input[type="password"],
input[type="tel"],
textarea,
select {

    width: 100%;

    padding: 12px;

    margin-bottom: 15px;

    border: 1px solid #ccc;

    border-radius: 3px;

    color: #000;

    font-size: 18px;

}
```

```
textarea {

    resize: vertical;

    height: 200px;

}
```

```
button {

    background-color: #4caf50;

    color: #fff;

    border: none;

    padding: 18px;
```

```
border-radius: 3px;

cursor: pointer;

width: 100%;

box-sizing: border-box;

font-size: 18px;

}
```

```
button:hover {

    background-color: #45a049;

}
```

```
.success-message {

    text-align: center;

    background: linear-gradient(to bottom, #4caf50, #81c784);

    border-radius: 10px;

    padding: 50px;

}
```

```
.success-message h2 {

    font-size: 36px;

    margin-bottom: 20px;

}
```

```
.success-message p {

    font-size: 24px;

}
```

```
</style>
```

```
</head>
```

```
<body>
```

```

<header>

  <h1>Login - Mail Server Database</h1>

</header>

<main>

  <form id="loginForm">

    <h2>Login</h2>

    <label for="loginEmail">Email:</label>

    <input type="email" id="loginEmail" name="loginEmail" required>

    <label for="loginPassword">Password:</label>

    <input type="password" id="loginPassword" name="loginPassword" required>

    <button type="submit">Login</button>

  </form>

</main>

<script>

  document.getElementById('loginForm').addEventListener('submit', function(event) {

    event.preventDefault();

    const successMessage = document.createElement('div');

    successMessage.className = 'success-message';

    successMessage.innerHTML = `

      <h2>Login Successful!</h2>

      <p>Welcome back! You are now logged in.</p>

    `;

    document.body.innerHTML = "";

    document.body.appendChild(successMessage);

  });

</script>

</body>

</html>

```


Signup.php

```
<?php
// Check if the form is submitted
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Get form data and sanitize
    $email = htmlspecialchars($_POST['signupEmail']);
    $password = htmlspecialchars($_POST['signupPassword']);
    $confirmPassword = htmlspecialchars($_POST['signupConfirmPassword']);

    // Basic validation
    if ($password === $confirmPassword) {
        // In a real application, you would save the data to a database here
        // For example: $hashedPassword = password_hash($password, PASSWORD_BCRYPT);
        // Save $email and $hashedPassword to your database

        // Display success message
        echo '<div class="success-message">';
        echo '<h2>Sign Up Successful!</h2>';
        echo '<p>Please login to continue.</p>';
        echo '</div>';
    } else {
        // Display error message if passwords do not match
        echo '<div class="success-message">';
        echo '<h2>Sign Up Failed!</h2>';
        echo '<p>Passwords do not match. Please try again.</p>';
        echo '</div>';
    }
}
?>
```

aboutus.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<title>About Us - Mail Server Database</title>
```

```
<style>
```

```
  body {
```

```
    font-family: Arial, sans-serif;
```

```
    background: linear-gradient(to bottom, #000000, #333333);
```

```
    color: #ffffff;
```

```
    margin: 0;
```

```
    padding: 0;
```

```
    font-size: 18px;
```

```
  }
```

```
  header {
```

```
    text-align: center;
```

```
    padding: 20px;
```

```
    background-color: rgba(0, 0, 0, 0.5);
```

```
  }
```

```
  main {
```

```
    padding: 20px;
```

```
  }
```

```
  h1, h2 {
```

```
    color: #fff;
```

```
    font-size: 28px;
```

```
  }
```

```
  p {
```

```
    font-size: 18px;
```

```
  }
```

```
  img {
```

```
    width: 100%;
```

```
    max-width: 600px;
```

```

border-radius: 10px;
margin-top: 20px;
}

ul {
margin-top: 20px;
padding-left: 20px;
}

li {
font-size: 18px;
}
</style>
</head>
<body>
<header>
<h1>About Us - Mail Server Database</h1>
</header>
<main>
<section id="mission">
<h2>Our Mission</h2>
<p>
Our Mail Server Database is a reliable and secure platform designed to manage your emails
efficiently.
We provide a seamless experience for sending, receiving, and organizing your emails. Our goal is
to ensure
the security and privacy of your communication while offering a user-friendly interface.
</p>

</section>

<section id="vision">
<h2>Our Vision</h2>
<p>

```

Our vision is to become the leading provider of secure and efficient email management solutions worldwide.

We aim to continuously innovate and improve our services to meet the evolving needs of our users, ensuring that their communications remain private, reliable, and accessible at all times.

</p>

</section>

<section id="achievements">

<h2>Our Achievements</h2>

<p>

Over the years, our Mail Server Database has achieved numerous milestones, including:

</p>

Successfully managing over a million secure email accounts.

Implementing advanced security measures to protect user data.

Receiving industry recognition for our innovative email management solutions.

Expanding our services to cater to both individual and enterprise clients.

</section>

</main>

</body>

</html>

Contactus.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Contact Us - Mail Server Database</title>

<style>

body {

font-family: Arial, sans-serif;

background: linear-gradient(to bottom, #000000, #333333);

```
    color: #ffffff;
    margin: 0;
    padding: 0;
    font-size: 18px;
}
```

```
header {
    text-align: center;
    padding: 20px;
    background-color: rgba(0, 0, 0, 0.5);
}
```

```
main {
    padding: 20px;
}
```

```
h1, h2 {
    color: #fff;
    font-size: 28px;
}
```

```
.contact-options {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    margin-top: 20px;
}
```

```
.contact-option {
    background-color: #005f73;
    padding: 25px;
    margin: 10px;
    border-radius: 10px;
    width: 150px;
```

```
text-align: center;
color: #ffffff;
font-size: 18px;
transition: background-color 0.3s ease;
}
```

```
.contact-option:hover {
    background-color: #0a9396;
}
```

```
.contact-option img {
    width: 50px;
    height: 50px;
    margin-bottom: 10px;
}
```

```
.contact-info {
    margin-top: 30px;
}
```

```
.contact-info p {
    font-size: 18px;
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<header>
```

```
<h1>Contact Us - Mail Server Database</h1>
```

```
</header>
```

```
<main>
```

```
<section id="contactUs">
```

```
<h2>How Can We Help?</h2>
```

```
<div class="contact-options">
```

```
<div class="contact-option">
```

```

        <p>I need help!</p>
    </div>
    <div class="contact-option">

        <p>I've got a business opportunity</p>
    </div>
    <div class="contact-option">

        <p>I'm a publisher</p>
    </div>
    <div class="contact-option">

        <p>I'm in the media</p>
    </div>
    <div class="contact-option">

        <p>I've got a copyright issue</p>
    </div>
    <div class="contact-option">

        <p>I have a ScribChat author suggestion</p>
    </div>
    <div class="contact-option">

        <p>Corporate gifts</p>
    </div>
</div>
<div class="contact-info">

    <h3>Contact Information</h3>

    <p>Email: mailserverdatabase@gmail.com</p>

```

```
<p>Phone: +91 2345678987</p>
<p>Address: 1234 Mail Server Lane, Tech City, TX 78901</p>
</div>
</section>
</main>
</body>
</html>
```

Inbox.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Inbox - Mail Server Database</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: linear-gradient(to bottom, #000000, #333333);
      color: #ffffff;
      margin: 0;
      padding: 0;
      font-size: 18px;
    }

    header {
      text-align: center;
      padding: 20px;
      background-color: rgba(0, 0, 0, 0.5);
    }

    main {
      margin-left: 220px;
      padding: 20px;
```



```
}
```

```
h1, h2 {  
  color: #fff;  
  font-size: 28px;  
}
```

```
.email-list {  
  list-style-type: none;  
  padding: 0;  
}
```

```
.email-list li {  
  padding: 10px;  
  border-bottom: 1px solid #ccc;  
  color: #fff;  
  font-size: 18px;  
}
```

```
.email-list li:hover {  
  background-color: #0a9396;  
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<header>
```

```
<h1>Inbox - Mail Server Database</h1>
```

```
</header>
```

```
<main>
```

```
<section id="inboxModule">
```

```
<h2>Received Emails</h2>
```

```
<ul class="email-list">
```

```
<!-- Sample received emails -->
```

```
<li>Subject: Approval Needed - From: boss@example.com</li>
```

```
<li>Subject: Invitation - From: events@example.com</li>
<li>Subject: Feedback Request - From: support@example.com</li>
</ul>
</section>
</main>
</body>
</html>
```

Ourvision.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Compose Email - Mail Server Database</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: linear-gradient(to bottom, #000000, #333333);
      color: #ffffff;
      margin: 0;
      padding: 0;
      font-size: 18px;
    }

    header {
      text-align: center;
      padding: 20px;
      background-color: rgba(0, 0, 0, 0.5);
    }

    main {
      margin-left: 220px;
```

```
padding: 20px;  
}
```

```
h1, h2 {  
  color: #fff;  
  font-size: 28px;  
}
```

```
form {  
  background-color: rgba(0, 0, 0, 0.8);  
  padding: 20px;  
  border-radius: 10px;  
  max-width: 600px;  
  margin: 0 auto;  
}
```

```
label {  
  display: block;  
  margin-bottom: 10px;  
  font-size: 18px;  
}
```

```
input[type="email"],  
input[type="text"],  
textarea {  
  width: 100%;  
  padding: 12px;  
  margin-bottom: 15px;  
  border: 1px solid #ccc;  
  border-radius: 3px;  
  color: #000;
```

```
font-size: 18px;
}
```

```
textarea {
    resize: vertical;
    height: 200px;
}
```

```
button {
    background-color: #4caf50;
    color: #fff;
    border: none;
    padding: 18px;
    border-radius: 3px;
    cursor: pointer;
    width: 100%;
    box-sizing: border-box;
    font-size: 18px;
}
```

```
button:hover {
    background-color: #45a049;
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<header>
```

```
<h1>Compose Email - Mail Server Database</h1>
```

```
</header>
```

```
<main>
```

```
<form id="composeEmailForm">
```

```
<h2>Compose Email</h2>
<label for="composeTo">To:</label>
<input type="email" id="composeTo" name="composeTo" required>
<label for="composeSubject">Subject:</label>
<input type="text" id="composeSubject" name="composeSubject" required>
<label for="composeMessage">Message:</label>
<textarea id="composeMessage" name="composeMessage" required></textarea>
<button type="submit">Send</button>
</form>
</main>
</body>
</html>
```

IMPLEMENTATION and RESULTS

Explanation Of Key Functions

➤ Mail Server Database

A mail server database is essential for managing email systems, storing user account information, email messages, and metadata like sender, recipient, and timestamps. It organizes emails into default and custom folders, facilitating easy retrieval and management. Advanced security measures protect against unauthorized access and spam, while automated backups and archiving ensure data retention and compliance. Integration with the mail server ensures efficient routing, delivery, and retrieval of emails. User management functions track permissions and activity, providing controlled access and oversight. This system underpins the reliability, security, and functionality of modern email services.

➤ User-Friendly

A user-friendly mail server database securely manages user accounts and organizes emails into intuitive folders. It features advanced security measures and automated backups for data protection and compliance. Seamless integration ensures reliable email delivery, while user management tracks permissions and activities. This system provides a secure, organized, and efficient email experience.

➤ Secured mails

Secured mails are protected through advanced encryption techniques during storage and transmission, ensuring data privacy. Multi-factor authentication (MFA) provides an additional layer of security for user accounts, reducing the risk of unauthorized access. Regular security audits and real-time threat detection help identify and mitigate potential vulnerabilities. Automated backups and disaster recovery measures ensure data integrity and availability. These features collectively ensure a robust and secure email communication system.

➤ Sent and Received mails

Sent and received mails are stored with their content and metadata in organized folders like Inbox and Sent Items. Security measures protect their integrity and privacy, while backups ensure data preservation. Real-time tracking facilitates efficient communication and oversight. This system ensures reliable, secure, and organized email handling..

Implementation

System Implementation: Implementation is the realisation of an application, or execution of a plan, idea, model, design, specification, standard algorithm, or policy. We worked so hard to implement this project. We used system implementation and website implementation.

For Implementation of a website: The website can be installed on a server. The owners of the website are to be properly trained to use all features of the website. To show the accuracy of the website and conformance of the owners or users.

Technologies Used:

Server: Apache(xampp)

Database server: MYSQL

TextEditor: vscode

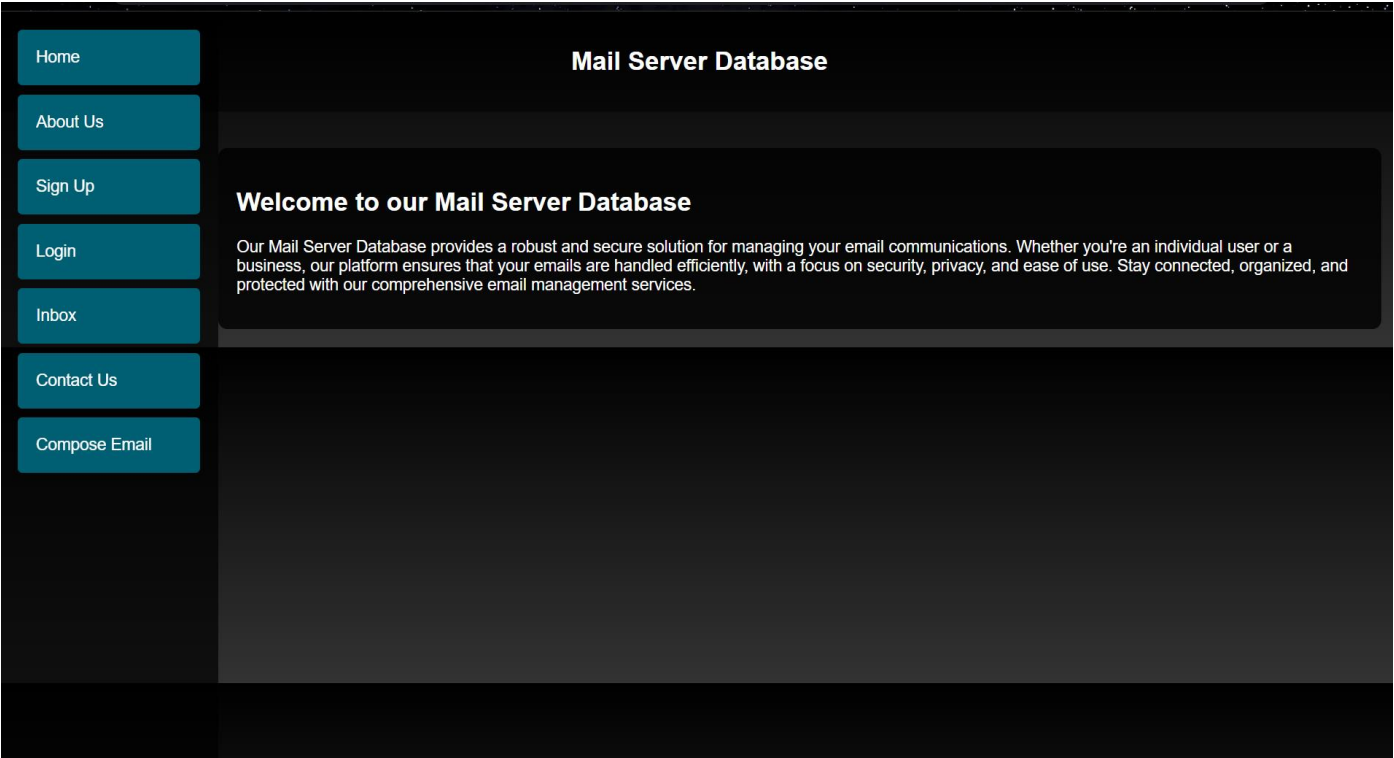
System Tools:

In a client management system database project, various tools and technologies are employed to facilitate development, deployment, monitoring and maintenance. Here are some essential system tools commonly used in the project.

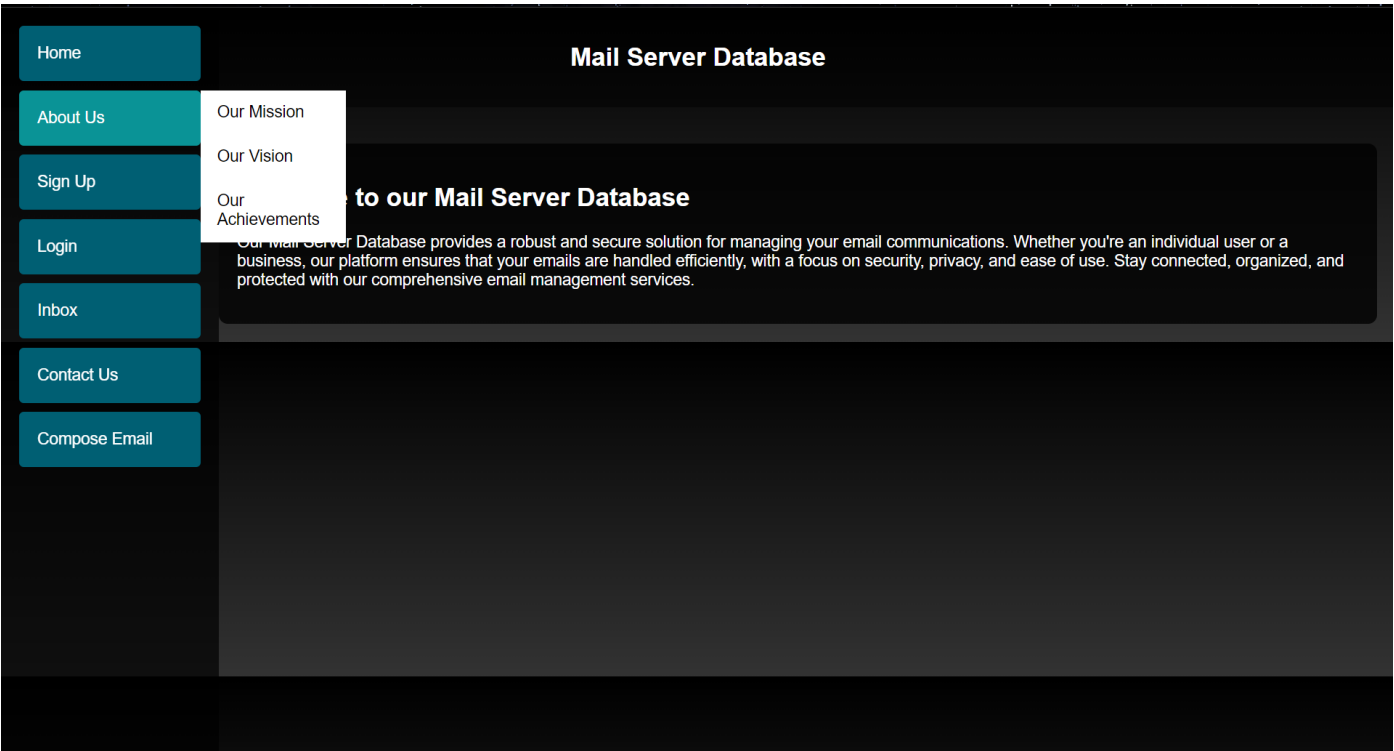
- **PHP**
- **MYSQL**
- **HTML**
- **CSS**

OUTPUT SCREENS

6.1 Home Page



6.2 Aboutus page



6.3 Our mission

Home

About Us

Sign Up

Login

Inbox


Contact Us

Compose Email

Mail Server Database

Our Mission

Our Mail Server Database is a reliable and secure platform designed to manage your emails efficiently. We provide a seamless experience for sending, receiving, and organizing your emails. Our goal is to ensure the security and privacy of your communication while offering a user-friendly interface.



6.4 Our vision

Home

About Us

Sign Up

Login

Inbox

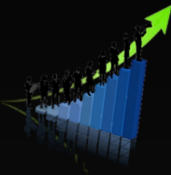
Contact Us

Compose Email

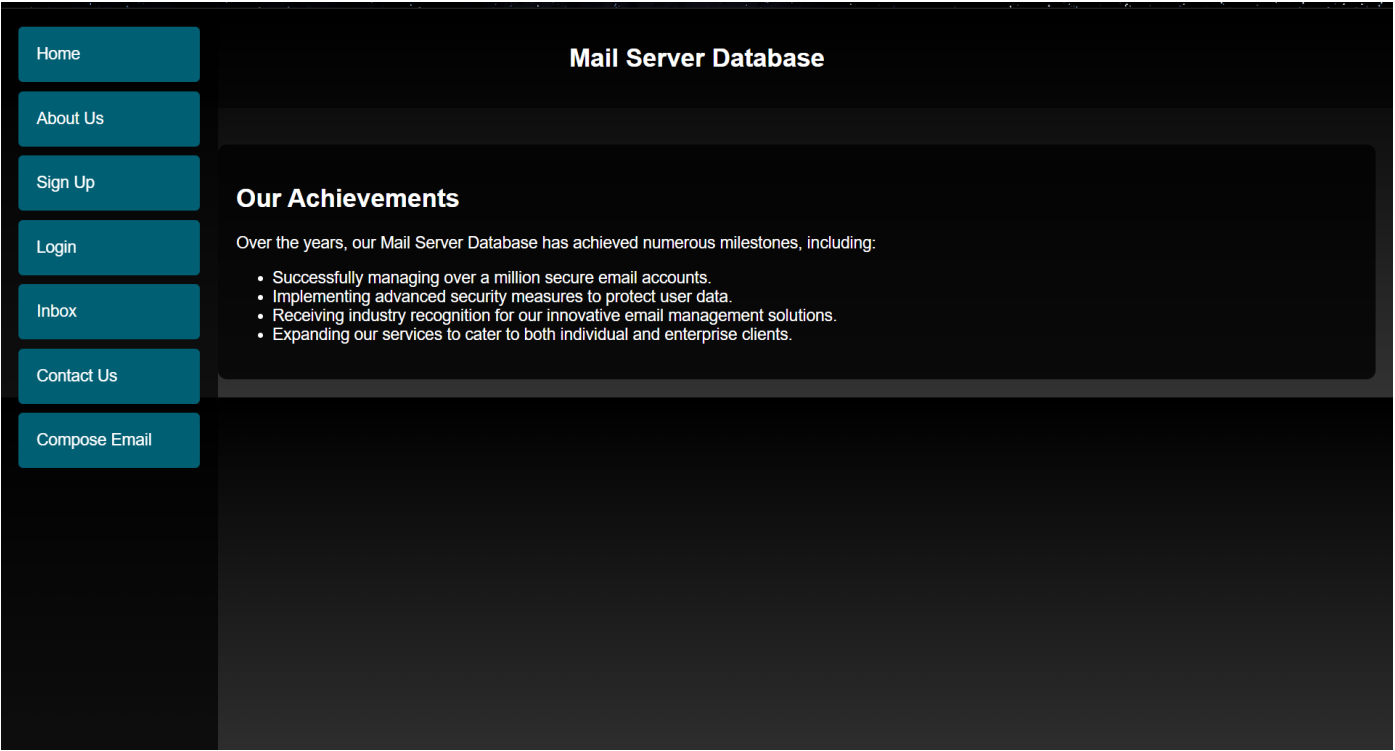
Mail Server Database

Our Vision

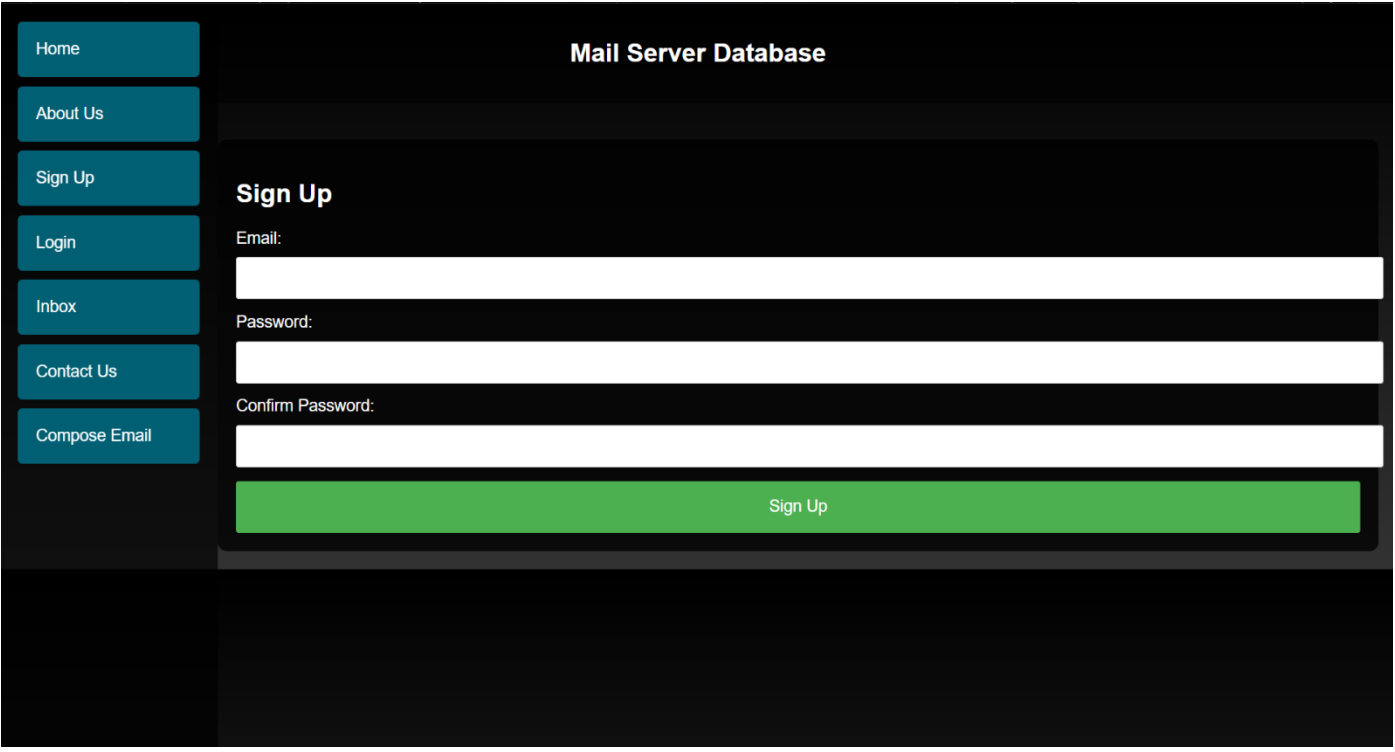
Our vision is to become the leading provider of secure and efficient email management solutions worldwide. We aim to continuously innovate and improve our services to meet the evolving needs of our users, ensuring that their communications remain private, reliable, and accessible at all times.



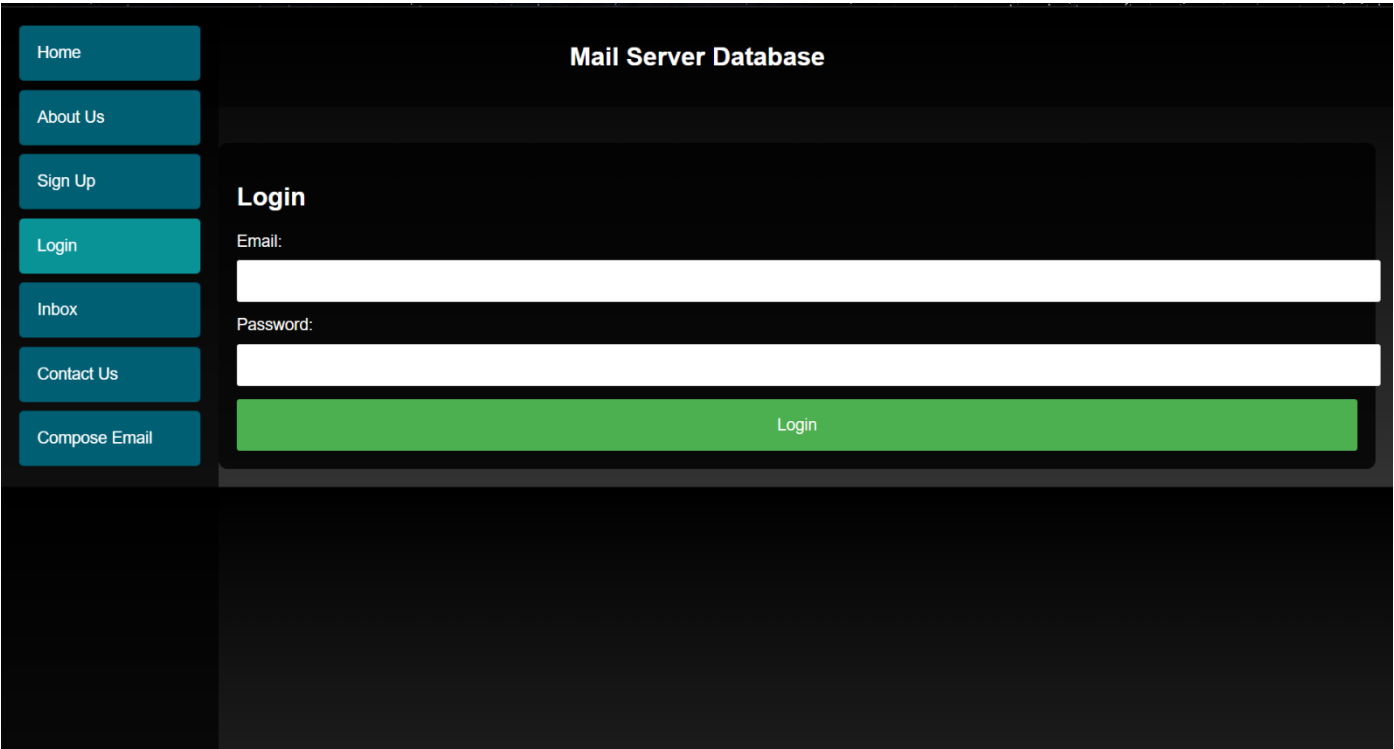
6.5 Our Achievements



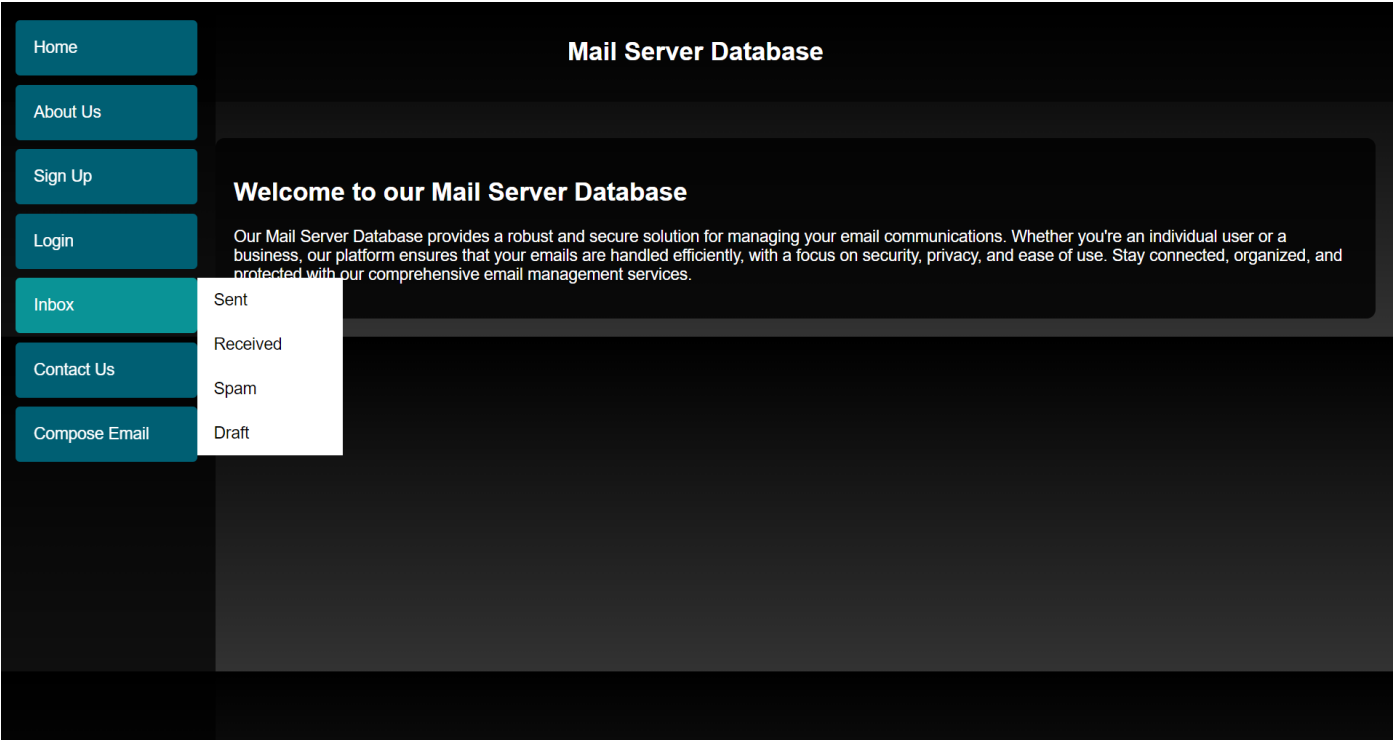
6.6 Signup page



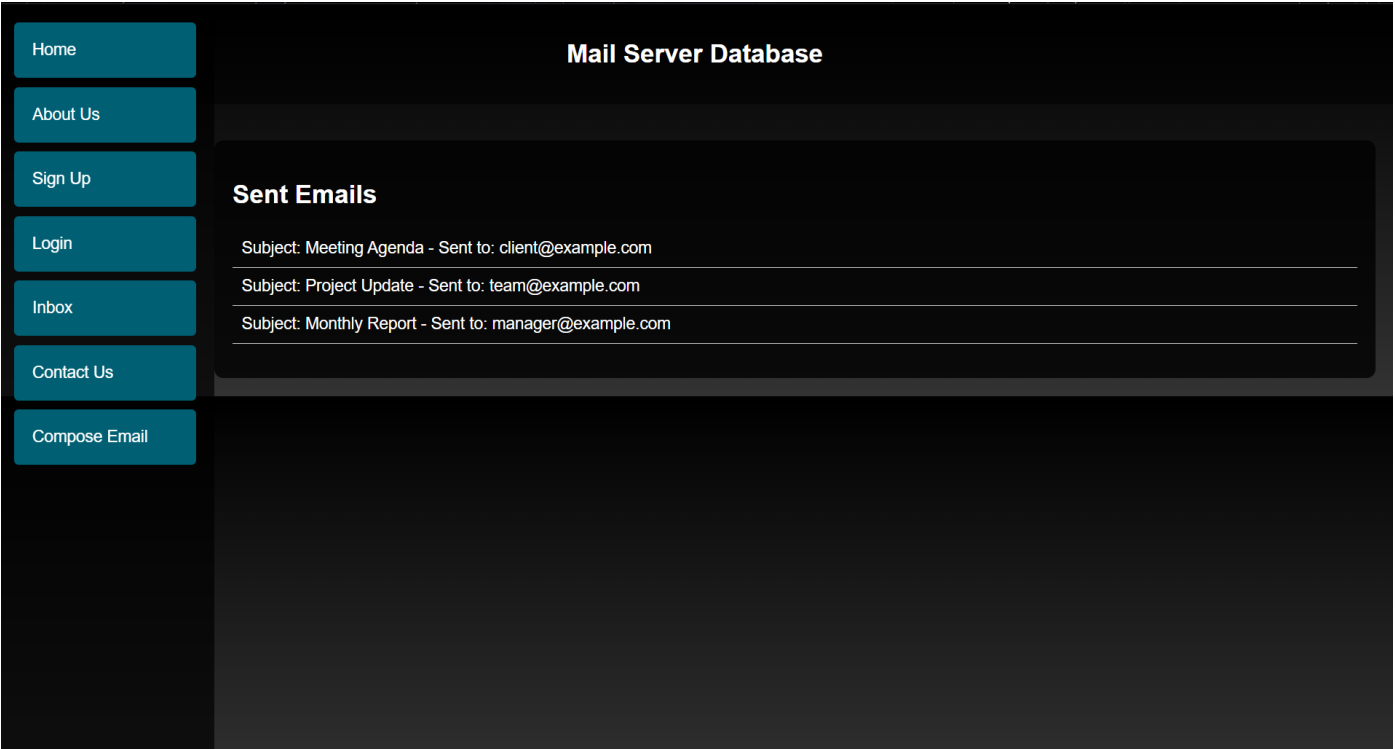
6.7 Login page



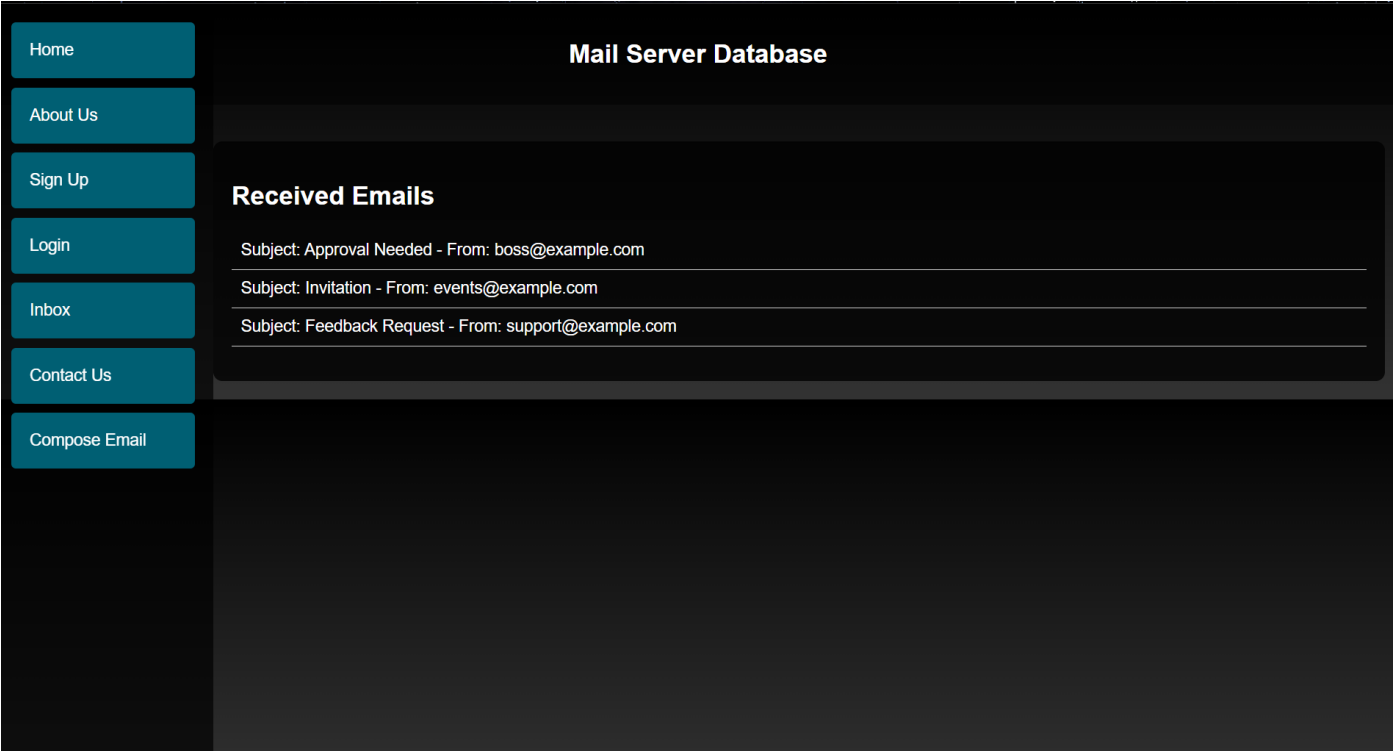
6.8 Inbox



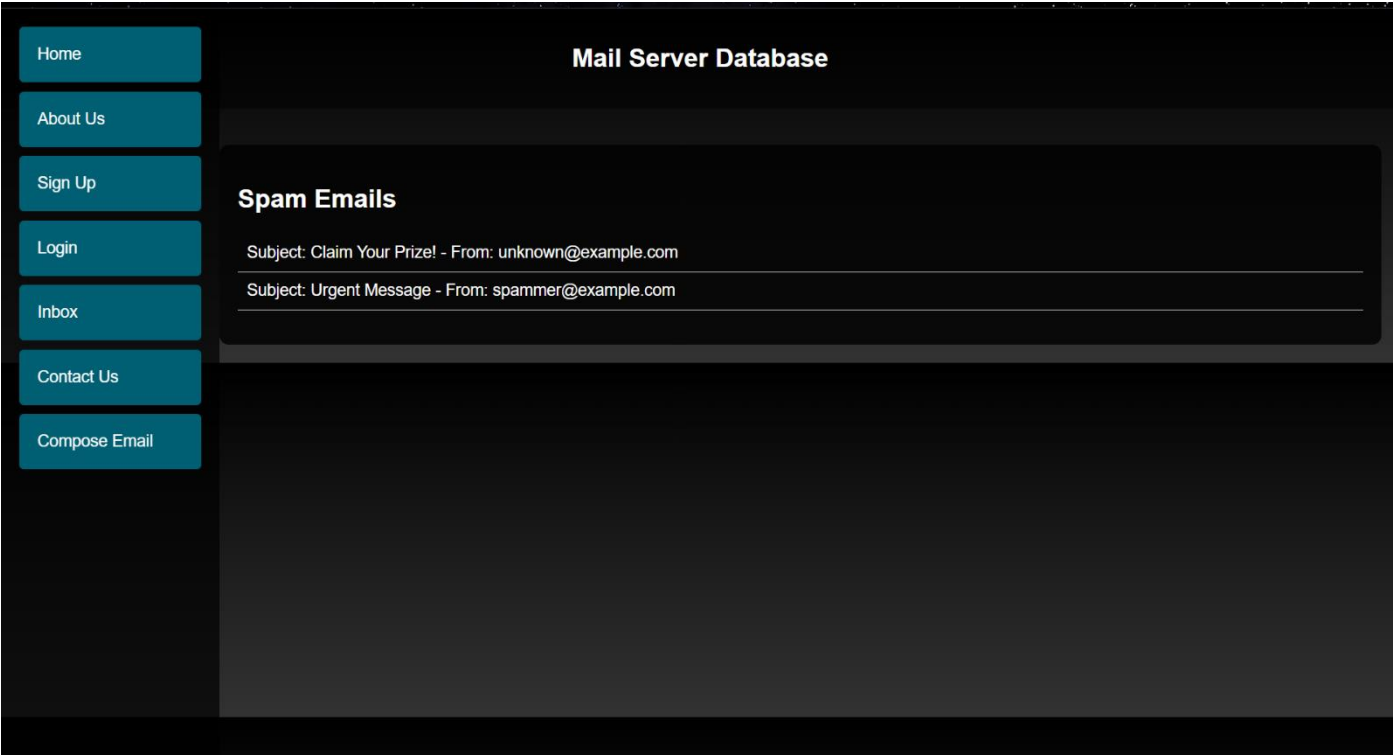
6.9 Sent page



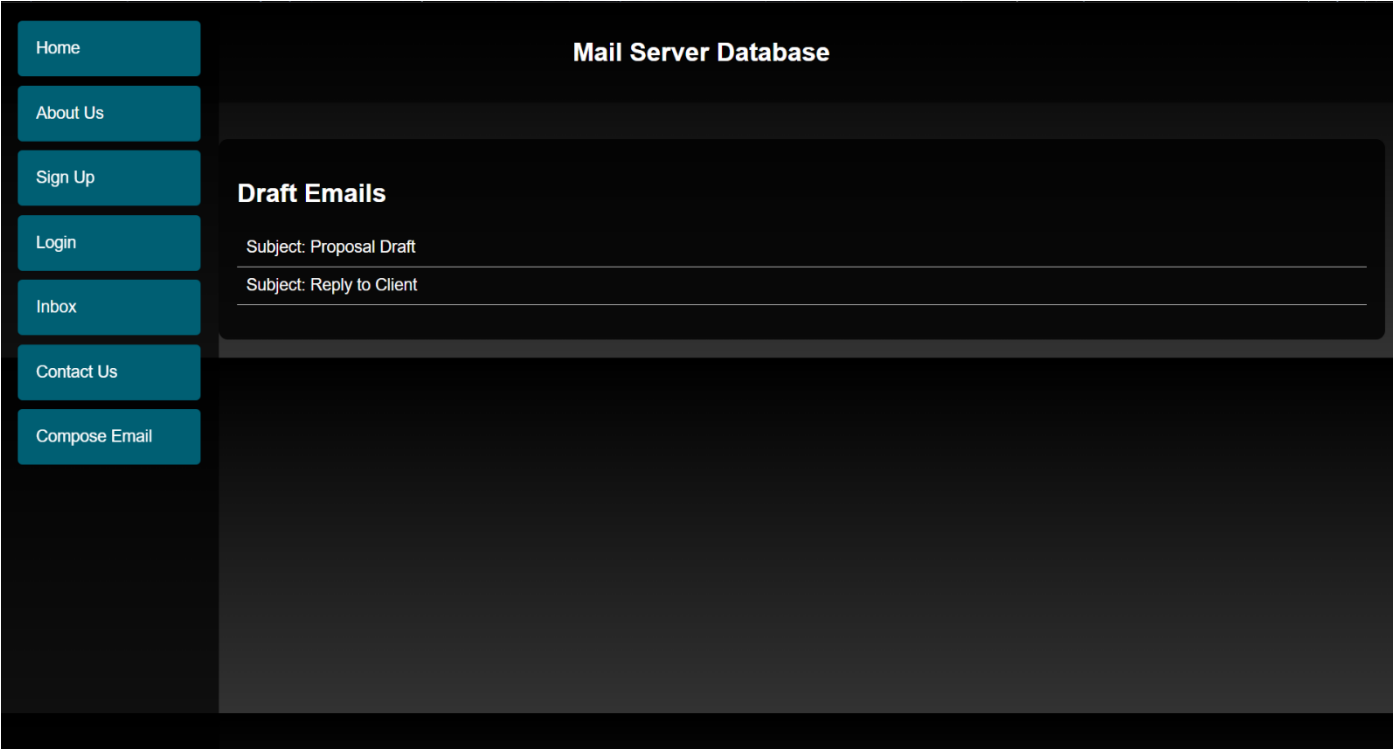
6.10 Received page



6.11 Spam page




6.12 Draft page




6.13 Contact us

[Home](#)[About Us](#)[Sign Up](#)[Login](#)[Inbox](#)[Contact Us](#)[Compose Email](#)


How Can We Help?




I need help!




I've got a business opportunity




I'm a publisher




I'm in the media



I've got a copyright issue



I have a ScribChat author suggestion



Corporate gifts

Contact Information

Email: mailserverdatabase@gmail.com
Phone: +91 2345678987
Address: 1234 Mail Server Lane, Tech City, TX 78901

6.14 Compose email

[Home](#)[About Us](#)[Sign Up](#)[Login](#)[Inbox](#)[Contact Us](#)[Compose Email](#)

Mail Server Database

Compose Email

To:

Subject:

Message:

Send

Chapter 7

Testing and Validation

Design of Test Cases and Scenarios

The design of test cases and scenarios for the mail server database project is meticulously structured to cover a comprehensive range of functionalities and potential edge cases. This ensures thorough validation of the system's performance, security, and usability. The test cases are categorized into functional, performance, security, and usability tests to provide a holistic evaluation.

Design of Test Cases and Scenarios

s.no	input	If available	If not available
1	User login	User get login	No
2	User signup	User get register	No

Validation

The validation of the Mail server database project is a critical phase that ensures the system functions as intended and meets all specified requirements. This process involves executing meticulously designed test cases across various functional areas to confirm the system's reliability, performance, security, and usability.

➤ Unit Testing

Unit testing is a fundamental process in the development of the Mail server database project, focused on validating the functionality of individual components or units of the system in isolation. Each unit, typically a function or method, is tested independently to ensure it performs as expected. This granular level of testing helps in identifying and fixing bugs at an early stage, thereby improving the overall quality of the code.

➤ Integration Testing

Integration testing is a critical phase in the development of the Mail server database project, aimed at ensuring that different modules and components of the system work together seamlessly.

This process involves combining individual units of the system and testing them as a group to identify any issues arising from their interaction. The primary goal is to detect and resolve integration issues, ensuring that the system's overall functionality is coherent and reliable.

➤ System Testing

System testing is a comprehensive evaluation phase in the Mail server database project, aimed at verifying the end-to-end functionality of the entire integrated system. This testing phase involves

testing the complete and fully integrated software product to ensure that it meets the specified requirements. The primary focus is on validating the system's compliance with functional, performance, and security specifications.

Conclusion

The testing and validation phases of the Mail server database project have confirmed that the system meets all specified requirements and functions effectively in real-world scenarios. Through rigorous requirements compliance, comprehensive functional validation, and robust performance and security assessments, the system has demonstrated its reliability, scalability, and user-friendliness. Data integrity and accuracy have been thoroughly validated, ensuring consistent and accurate data management.

Chapter 8

CONCLUSION

Two hundred years of breathtaking innovation since the dawn of the industrial age have produced rising living standards for ordinary people in much of the world, with no sharply rising trend for unemployment. Technology is taking over our world, we live in an age where most, if not all of us use technological devices every day. From mobile phones; to iPads, to laptops and TVs, we're constantly using technology. There are even forms of technology that we use during our sleep, fitness watches that track our sleep and the likes. Yes, there have been many problems, notably bouts of staggering inequality and increasingly horrific wars. On balance, however, throughout much of the world, people live longer, work much fewer hours, and lead generally healthier lives. But there is no denying that technological change nowadays has accelerated, potentially leading to deeper and more profound dislocations. Everyone uses technology, from children and teenagers, to adults and elders. Technology has influenced people and their daily lives, some better than others. Walk into almost any business, big or small, and you'll quickly see how technology has transformed the way we work. Whether you're an entrepreneur, a bike courier, or a criminal lawyer, one thing is clear: our lives are surrounded by technology that just a handful of years ago would have seemed unfathomable. Technology has made us more responsive, more able to gain access to information over a broader spectrum. It has taken our information and, instead of putting it into little cubby holes of the company, made it broadly accessible

REFERENCES

1. Database Design on Embedded home gateway and webserver implementation. 2011 International Conference on Consumer Electronics, Communications and networks.
2. The Performance of a Bare Machine Email Server. 2009 21st International Symposium on Computer Architecture and High Performance Computing(SBACPAD).
3. A Prediction Mechanism of Mail Retrieval Based on the user Behaviour analysis for Electronic Mail Database System. 2008 Eight International Conference on intelligent systems design and applications (IDSA).
4. A performance study on Internet Server Provider mail servers. IEEE Symposium on Computers and communication (ISCC) 2004.

