Software Requirements Specification

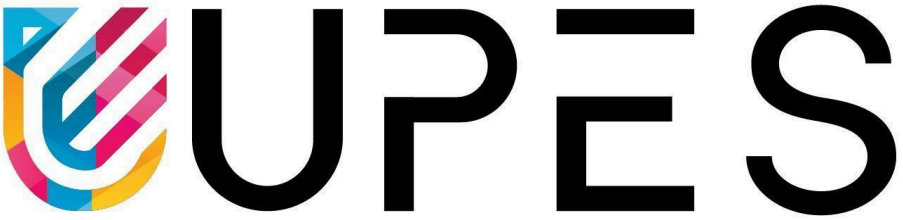
For

CHIME:Let's Connect

2023

Prepared by

Specialization	SAP ID	Name
CCVT(H)	500083727	Bhoomika
CCVT(H)	500082951	Mani Paliwal
CCVT(H)	500085000	Nupur Sharma
CCVT(H)	500086703	Shikhar Nag



Department of Systemics
School Of Computer Science
UNIVERSITY OF PETROLEUM & ENERGY STUDIES,
DEHRADUN- 248007. Uttarakhand

Dr. Alind
Project Guide

Dr. Neelu J. Ahuja
Cluster Head

Table of Contents

1. INTRODUCTION

The advent of cloud technology has revolutionized various aspects of modern business operations, including the realm of human resources. This project delves into the innovative landscape of cloud-based interview schedulers, a crucial tool in streamlining the recruitment process. By harnessing the benefits of the cloud, organizations can seamlessly coordinate interviews, optimize scheduling efficiency, and enhance candidate experiences. The evolution of cloud technology has fundamentally reshaped the landscape of modern business operations. Among its many applications, cloud-based interview schedulers have emerged as a transformative solution in the realm of human resources.

The system architecture for "CHIME: Let's Connect" encompasses a well-structured frontend built with EJS to provide user-friendly interfaces for recruiters and applicants. The frontend facilitates interview slot management, invitation sending, and response tracking for recruiters. On the backend, Node.js and Express.js power a RESTful API that manages HTTP requests, handles core application logic, and interfaces with the MongoDB database. The database is organized into collections for various data types, ensuring efficient querying and scheduling algorithms. For robust hosting and scalability, the system leverages cloud infrastructure using AWS and MongoDB Atlas for cloud-based database management.

In summary, "CHIME: Let's Connect" adopts a modern system architecture that combines EJS for the frontend, Node.js and Express.js for the backend, and MongoDB for data storage. Cloud infrastructure choices enhance scalability and reliability, making it a well-rounded solution for efficient interview scheduling and management.

In an era marked by increased global connectivity and remote work, the ability to efficiently coordinate interviews across time zones and locations is paramount. Cloud-based interview schedulers offer organizations a dynamic platform to manage and streamline this intricate process. By centralizing scheduling, they eliminate the complexities of coordinating between multiple stakeholders, optimizing the recruitment pipeline's efficiency.

Moreover, these solutions enhance the candidate experience, offering flexibility and ease of use. Applicants can conveniently select interview slots that align with their schedules, fostering a positive impression of the hiring organization.

However, the implementation of cloud-based interview schedulers is not without its challenges.

1.1 Purpose of the Project

The "CHIME: Let's Connect" interview scheduler application project serves a crucial purpose in the modern business landscape, which has been transformed by the advent of cloud technology and increased global connectivity. Its primary aim is to efficiently coordinate interviews in a world where remote work and diverse time zones are the norm. By harnessing cloud-based technology, the application centralizes interview scheduling, eliminating the complexities of coordinating interviews between multiple stakeholders, and ultimately streamlining the recruitment pipeline. This results in improved efficiency for HR professionals and recruiters. Furthermore, The project places a strong emphasis on enhancing

the candidate experience. It provides applicants with the flexibility to select interview slots that align with their schedules, creating a positive impression of the hiring organization and contributing to an overall improved recruitment experience. This application is a modern, cloud-based solution designed to optimize interview scheduling and management in a global, remote work environment.

1.2 Target Beneficiary

The "CHIME: Let's Connect" project caters to a diverse set of beneficiaries. HR professionals and recruiters benefit from streamlined interview scheduling, while job applicants experience enhanced convenience. Hiring managers gain efficiency, organizations see increased productivity, and remote workforces benefit from flexibility. Compliance and security teams find data management assurance, and technical teams enjoy scalable, reliable infrastructure. The project enhances the experiences of all involved in the recruitment process and addresses the needs of a modern, globally connected work environment.

1.3 Project Scope

The scope for Project encompasses the development and implementation of a cloud-based interview scheduler application. It includes a user-friendly interface for HR professionals, recruiters, and job applicants, facilitating efficient interview slot management, invitation sending, and response tracking. The system's architecture combines EJS for the frontend, Node.js and Express.js for the backend, and MongoDB for data storage. The application leverages cloud infrastructure for scalability and reliability. Its primary focus is on optimizing interview scheduling, improving the candidate experience, and adapting to the demands of a global and remote workforce, while ensuring data security and compliance.

1.4 References

[1] Thalawattha, Sathsara & Vidanagama, Dushyanthi. (2021). A Survey on Web-based Meeting Scheduling Application.

[2] Perera, Poornima & Vidanagama, Dushyanthi. (2020). A WEB-BASED PAPERLESS MEETING MANAGEMENT SYSTEM.

[3] Erik Timmerman, C., Shik Choi, C., 2017. Meeting Technologies. https://doi.org/10.1002/9781118955567.wbieoc132

[4] Tran, Linh & Stojcevski, Alex & Pham, Thanh & Souza-Daw, Tony & Nguyen, Nhan & Nguyen, Vinh & Nguyen, Chau. (2016). A smart meeting room scheduling and management system with utilization control and ad-hoc support based on real-time occupancy detection. 186-191. 10.1109/CCE.2016.7562634.

2. PROJECT DESCRIPTION

2.1 SWOT Analysis

Strength: An interview scheduler streamlines the interview process, saving time and effort for both candidates and interviewers. Many interview schedulers can automate reminders, follow-ups, and notifications, reducing the risk of missed interviews

Weakness: Reliance on technology makes the system vulnerable to technical glitches, disrupting scheduling and creating a negative experience for users. It heavily relies on the accuracy of users' calendars and any discrepancies or errors in these calendars can lead to scheduling conflicts.

Opportunities: Collecting data on interview scheduling can provide insights into process efficiency, helping organizations make data-driven improvements. Developing a mobile app version can make interview scheduling even more accessible and convenient for users on the go.

Threats: Some users may resist using new software or prefer traditional scheduling methods, causing adoption challenges. There may be strong competition in the interview scheduling software market, making it challenging to stand out.

2.2 Project Features

The key features of the "CHIME: Let's Connect" project encompass a user-friendly interface catering to HR professionals, recruiters, and applicants, facilitating efficient interview slot management, invitation distribution, and response tracking. Its modern system architecture combines EJS for the frontend, Node.js, and Express.js for the backend, along with MongoDB for data storage, ensuring a robust foundation. The application harnesses cloud infrastructure for scalability and reliability and is designed to support a global and remote workforce by simplifying the intricacies of interview scheduling. It prioritizes the candidate experience through flexible slot selection and maintains data security and compliance with stringent legal and privacy requirements, making it a comprehensive solution for efficient interview management.

2.3 Design and Implementation Constraints

- Networking issues
- Performance Optimization
- Technical compatibility

2.4 Design diagram

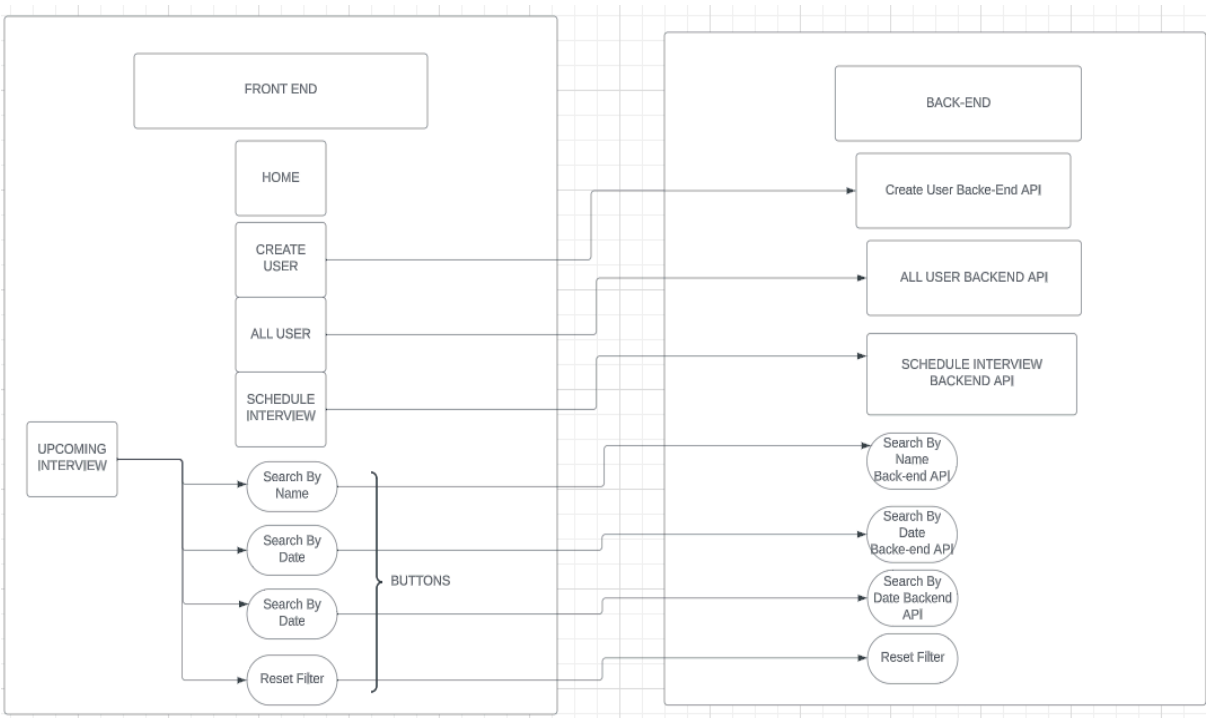


Fig. 1- Design Diagram

3. SYSTEM REQUIREMENTS

3.1 User Interface

Our project requires a user interface so that it is easier for the users to get their respective data. For this, we have used HTML, CSS, JS & NodeJs to design our frontend which will allow the admin to schedule the interviews for the various participants. Generation of triggers like mails will help the user to get the information about their interview date and slot. All the participant's information who have their interview scheduled is saved at the backend - MongoDb Cloud from where it will be accessed anytime anywhere.

Design and implementation of the User Interface: Visual Studio Code.
Database Storage: MongoDB Cloud

3.2 Protocols

- **HTTP (Hypertext Transfer Protocol)** is an application-layer protocol used for communication on the World Wide Web. It is the foundation of data communication for the World Wide Web, and it enables communication between clients and servers.

HTTP was designed to be a stateless protocol, meaning that it does not keep track of previous interactions between the client and server. It is used to transmit text, images, videos, and other types of data over the Internet. It works by using a client-server model, where the client sends a request to the server, and the server responds with the requested information.

- **SMTP (Simple Mail Transfer Protocol)** is the standard communication protocol used for transmitting email messages over the internet. It defines the rules and procedures for routing and delivering emails, ensuring that messages are sent from the sender's email client to the recipient's email server and ultimately to the recipient's inbox. SMTP plays a fundamental role in email communication, enabling the reliable exchange of messages across the global network of email servers.

- **RESTful API (Representational State Transfer Application Programming Interface)** is a web-based protocol that facilitates data communication between different software applications. It adheres to the principles of simplicity, statelessness, and well-defined resource endpoints. RESTful APIs are widely used for building web services, enabling seamless interaction and data exchange between clients and servers on the internet. They provide a structured and efficient means of accessing and manipulating resources through HTTP requests, making them a fundamental component of modern web applications.

4. NON-FUNCTIONAL REQUIREMENTS

4.1 Performance requirements

The following are the five basic requirements needed:

- Ensure prompt response to user actions, real-time operations, and data retrieval.
- Design for scalability to handle growth efficiently.
- Optimize CPU and memory usage.
- Minimize email notification delays.
- Ensure high system reliability with minimal downtime and robust error handling.

4.2 Security requirements

- Maintain detailed logs, implement monitoring and alerting systems for prompt security incident detection.
- Stay updated with software and system updates to address known vulnerabilities and protect against emerging threats.

4.3 Software Quality Attributes

Performance: It should respond promptly to user interactions, including scheduling, email notifications, and data retrieval, ensuring efficient real-time or near-real-time operations.

Portability: It should be accessible and functional across different devices and web browsers to accommodate users with varying technology preferences.

Usability: The application should be user-friendly and intuitive, ensuring that both administrators and candidates can easily navigate and use its features.

Scalability: The application must be capable of handling a growing number of users and scheduling data without significant performance degradation, making it adaptable to organizational growth.

APPENDIX A: GLOSSARY

- SMTP Server- SMTP (Simple Mail Transfer Protocol) is a protocol used for sending and receiving emails over the Internet. An SMTP server is a software application or program that runs on a server and facilitates the transfer of emails between email clients or servers.
  SMTP servers work by using a set of commands and responses to communicate with other email servers or clients. When an email is sent, the sender's email client or server sends the message to the SMTP server, which then forwards the message to the recipient's email server or client. The recipient's email server or client then retrieves the email from the server and delivers it to the recipient. SMTP servers use TCP (Transmission Control Protocol) to ensure reliable communication between email clients or servers. The default port for SMTP is 25, but many email servers also support alternate ports such as 587, which is often used for secure communication using TLS (Transport Layer Security) encryption. SMTP servers are essential for sending and receiving emails, and they are used by both individuals and organizations.

- HTTP Protocol- HTTP (Hypertext Transfer Protocol) is an application-layer protocol used for communication on the World Wide Web. It is the foundation of data communication for the World Wide Web, and it enables communication between clients and servers. HTTP was designed to be a stateless protocol, meaning that it does not keep track of previous interactions between the client and server. It is used to transmit text, images, videos, and other types of data over the Internet. It works by using a client-server model, where the client sends a request to the server, and the server responds with the requested information. HTTP requests are made using a uniform resource locator (URL), which identifies the resource being requested, and a set of headers, which provide additional information about the request, such as the client's

preferred language or the type of data it can accept. The HTTP response contains a status code, which indicates whether the request was successful or not, and the requested data.

- Nodemailer - Nodemailer is a widely used and powerful JavaScript module for Node.js that simplifies the process of sending email from web applications. It offers a versatile and straightforward way to integrate email functionality into Node.js projects, making it a popular choice for developers who need to incorporate email notifications, messaging, and communication features. Nodemailer supports various email transport methods, including SMTP, Sendmail, and more, providing flexibility in how email messages are delivered. With Nodemailer, developers can construct and send emails with ease, allowing for customization of email content, attachments, and recipient lists. This module not only streamlines the development of email-related functionality but also includes features for handling attachments, HTML content, and plain text, making it a valuable tool for building applications that require email communication. Its simplicity and compatibility with Node.js have established Nodemailer as a favored choice for developers looking to implement email functionality seamlessly into their web applications.

- Mongoose - Mongoose is an elegant and robust JavaScript library that serves as a popular Object Data Modeling (ODM) tool for Node.js and MongoDB. It simplifies the interaction between Node.js applications and MongoDB databases, offering a structured and intuitive way to work with data. Mongoose provides a schema-based data modeling approach, allowing developers to define the structure and data types of documents within their MongoDB collections. This not only ensures data integrity but also streamlines data validation, making it an ideal choice for applications where data consistency and organization are essential. Mongoose also comes with built-in support for various MongoDB features, such as indexes, middleware functions, and validation rules, making it a versatile and powerful tool for building and managing applications that rely on MongoDB as their data store. Whether used for developing RESTful APIs, web applications, or any Node.js project that involves MongoDB, Mongoose simplifies data handling and allows developers to work with databases more efficiently and with confidence in data integrity.