WEB ENGINEERING

CHAT APPLICATION

Presented by:
Rashmi Mohan(IT-74)
Sahil Sangtani(IT-78)
Satyam Agrawal(IT-81)

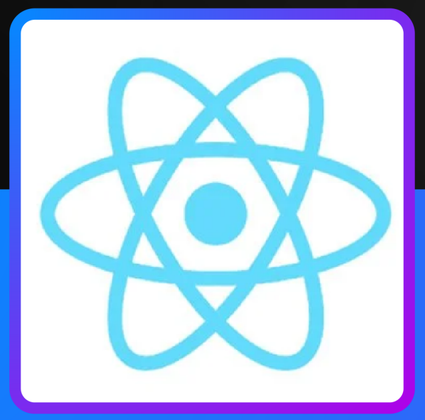
Submitted to:
Dr. Lalit Purohit
Mr. Upendra Singh

Introduction to Real-Time Chat Applications

Significance of Real-Time Communication in Modern Applications

Technologies Used

React



React is a free and open-source front-end JavaScript library for building user interfaces based on components.

Node.js

Node.js is a cross-platform, open-source JavaScript runtime environment and executes JavaScript code outside a web browser.

Socket.io

Socket.IO is an event-driven library for real-time web applications. It enables real-time, bi-directional communication between web clients and servers. I

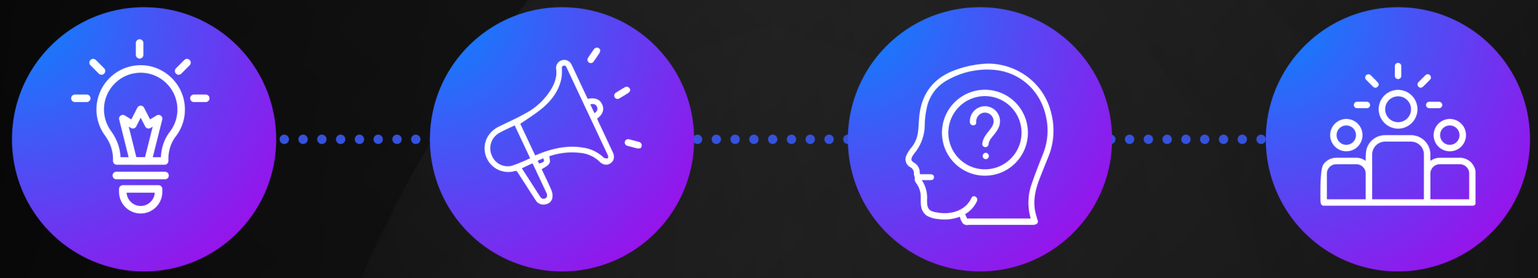
MongoDB

MongoDB is a database program. Classified as a NoSQL database product, MongoDB utilizes JSON-like documents with optional schemas.

Project TechStack

- Breakdown of Frontend and Backend Technologies: React for Frontend, Node.js for Backend
- Role of Socket.io in Enabling Real-Time Communication
- Importance of MongoDB as a NoSQL Database for Scalable Data Storage
- Advantages of Using a Comprehensive Technological Stack for Building Robust Applications

Installing Essential Packages



Package Management in Node.js with npm

Essential Packages for Real-Time Chat App Development

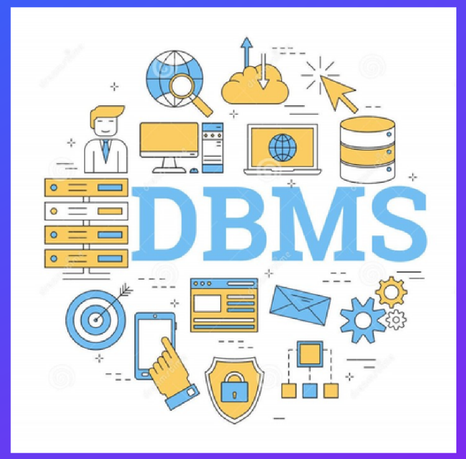
Package's Role in the Development Process

Step-by-Step Installation Procedure Using npm

Setting Up the Backend

- Express Server Initialization Process
- Middleware in Express for Handling Requests
- Error Handling Mechanisms in Express
- Implementation of Environment Variables with EMV for Enhanced Security

Database Management

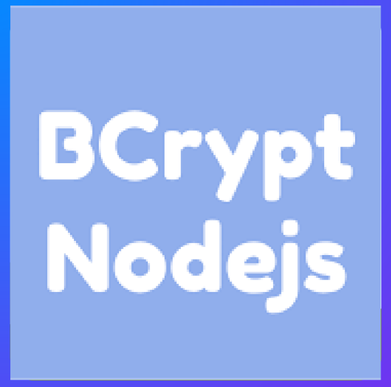


- Explanation of MongoDB Setup Process: Installation and Configuration
- Schema and Model Creation with Mongoose for Structured Data Storage
- CRUD Operations (Create, Read, Update, Delete) with MongoDB
- Practices for Efficient Database Management in Real-Time Applications

User Authentication

Significance of User Authentication in Real-Time Chat Applications

Role of JWT (JSON Web Tokens) in Secure Authorization



Implementation of bcrypt JS for Secure Password Hashing

Best Practices for User Authentication in Node.js Applications

Handling User Authentication (Contd.)

01

Creation of Login, Sign-Up, and Logout Functions in Node.js

02

Implementation of Controller Functions for User Input Validation

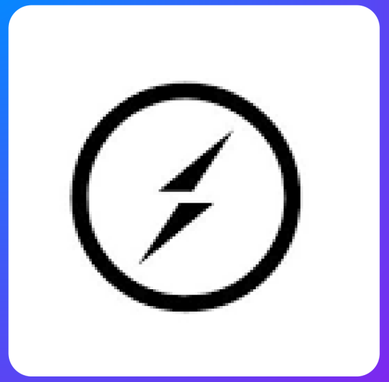
03

Overview of Error Handling Mechanisms to Handle Authentication Error

04

Importance of User Authentication for Securing User Data and Enhancing User Experience

Real-Time Communication



Introduction to Socket.io for Real-Time Messaging and Event-Based Communication

Schema and Model Design for Managing Messages and Conversations

Integration of Socket.io for Enabling Instant Message Delivery

Importance of Real-Time Communication in Enhancing User Engagement and Experience

Deployment Considerations

- Overview of Common Deployment Challenges in Real-Time Chat Applications
- Strategies for Production Deployment: Continuous Integration and Continuous Deployment (CI/CD)
- Exploration of Containerization Technologies (e.g., Docker) for Efficient Deployment
- Importance of Cloud Infrastructure (e.g., AWS, Azure) for Scalability and Reliability

Scalability Considerations

- Importance of Scalability in Real-Time Chat Applications
- Strategies for Horizontal and Vertical Scaling in Node.js Applications
- Discussion on Load Balancing Techniques for Distributing Traffic Effectively
- Overview of Caching Mechanisms for Improving Performance and Scalability

Security Measures

- Importance of Security in Real-Time Chat Applications
- Overview of Common Security Threats (e.g., Cross-Site Scripting, SQL Injection)
- Implementation of Security Measures (e.g., HTTPS, Content Security Policy)
- Role of Security Audits and Penetration Testing in Ensuring Application Security

Performance Optimization

- Importance of Performance Optimization in Real-Time Applications
- Strategies for Code Optimization and Minification
- Discussion on Client-Side and Server-Side Caching Techniques
- Role of Profiling Tools for Identifying Performance Bottlenecks

Continuous Integration and Deployment (CI/CD)

01
Overview of Continuous Integration (CI) and Continuous Deployment (CD) Pipelines

02
Implementation of Automated Testing and Deployment Processes

03
Introduction to CI/CD Tools (e.g., Jenkins, Travis CI)

04
Benefits of CI/CD in Ensuring Code Quality and Deployment Efficiency

Monitoring and Logging

Importance of Monitoring and Logging in Real-Time Applications

Implementation of Log Management Systems (e.g., ELK Stack)

Overview of Application Performance Monitoring (APM) Tools (e.g., New Relic, Datadog)

Role of Monitoring and Logging in Identifying and Resolving Issues in Real□Time Applications

THANK YOU

For watching this presentation