REAL ESTATE WEB APPLICATION PROJECT DOCUMENTATION

Submitted By:

- Srabonti Suchi Talukdar | ID:056222005101037
- Allo Rani Malakar | ID: 0562220005101013
- Rashada Chowdhury | ID:0562220005101055

Submitted To:

Sabuj Chandra Paul | Sr. Software Engineer & Guest Lecturer at NEUB

JULY 16, 2025

1. Introduction

Overview of the Project Idea and Purpose

This project is a **full-stack Real Estate Web Application** developed using the **MERN stack** (MongoDB, Express.js, React.js, and Node.js), integrated with **Prisma ORM**, and styled using **SCSS** for a responsive and modern user interface.

The primary purpose of this system is to provide a **comprehensive and user-friendly platform** where:

- **Customers** can explore property listings, save favorites, chat with agents, and receive support.
- **Agents** can manage their profiles, respond to inquiries, and track listing performance.
- **Administrators** can oversee users, monitor activities, manage recruitment, and maintain platform integrity.

Key Features

- 🔍 Dynamic property listings with filters and saved posts
- 💬 Live chat functionality for real-time communication
- 💼 Agent recruitment system for onboarding professionals
- 📊 Admin dashboards with user management and analytics
- 📧 Email notifications and messaging system
- ☺ Smart AI-based customer support integrated via **Crisp Chat**

This web application not only aims to **simplify real estate transactions** but also offers an **interactive and scalable solution** for the evolving needs of modern real estate platforms.

2. Background of the Project

Problem Statement and Motivation

Traditional real estate platforms often lack **transparency**, **real-time interaction**, and structured **role-based features**. Customers frequently face challenges in directly communicating with agents, while administrators encounter difficulties in effectively managing property listings, agent recruitment, and user roles.

The motivation behind this project is to **streamline these core processes** by developing a modern digital platform that:

- Enhances the overall **user experience**
- Enables **efficient and secure communication**
- Automates administrative and operational tasks

By leveraging full-stack technologies and a thoughtful system design, this application aims to overcome the limitations of traditional systems and provide a robust, scalable solution for real estate management.

3. Objectives

The primary objectives of this project are to:

- Provide a digital platform for real estate listing, management, and communication.
- Implement role-based access for customers, agents, and administrators.
- Allow users to save properties, apply for agent roles, and engage in live chat.
- Enable agents to manage their property listings with administrative approval.
- Allow administrators to control platform content and user roles through a dashboard.
- Integrate AI-based responses to handle basic customer queries and support.

4. Scope

Covered

The project includes the following features and functionalities:

- Role-based access for three user types: Customer, Agent, and Admin
- Property listing, filtering, saving, and management system
- Real-time chat between customers, agents, and administrators
- Agent and admin recruitment and approval workflow
- Integration with Crisp Chat for AI-based customer support
- Email communication features including automated replies
- Dynamic dashboards with charts and analytics for administrative monitoring

Not Covered

The following features are beyond the current scope of this project:

- Integration with payment gateways or online transaction systems
- Native or cross-platform mobile application development
- Multi-language or localization support

5. Literature Review / Related Work

Existing Systems

Several real estate platforms already exist in the market, offering various levels of functionality and user interaction:

- **Bproperty.com** (Bangladesh-based): Focuses on property listings and local market presence.
- **Zillow**: A U.S.-based platform that provides comprehensive property search, valuation, and agent connections.
- **MagicBricks**: An Indian platform offering property listings, location filters, and real estate services.

While these platforms effectively support property browsing and basic agent-customer communication, they **lack features such as:**

- A unified agent recruitment and approval system
- Dynamic dashboards with admin-level insights and control
- AI-based customer interaction and real-time chat integration

Relevant Models and References

This project leverages established models and practices for software development and access control:

- **Rapid Application Development (RAD)**: A flexible software development model that emphasizes quick prototyping and iterative delivery.
- **Role-Based Access Control (RBAC)**: A security model that restricts system access based on user roles (Customer, Agent, Admin), improving security and task delegation.

6. Methodology

Technologies and Tools Used

- **Frontend:** React.js, SCSS
- **Backend:** Node.js, Express.js
- **Database:** MongoDB with Prisma ORM
- **Authentication:** JWT (JSON Web Token)

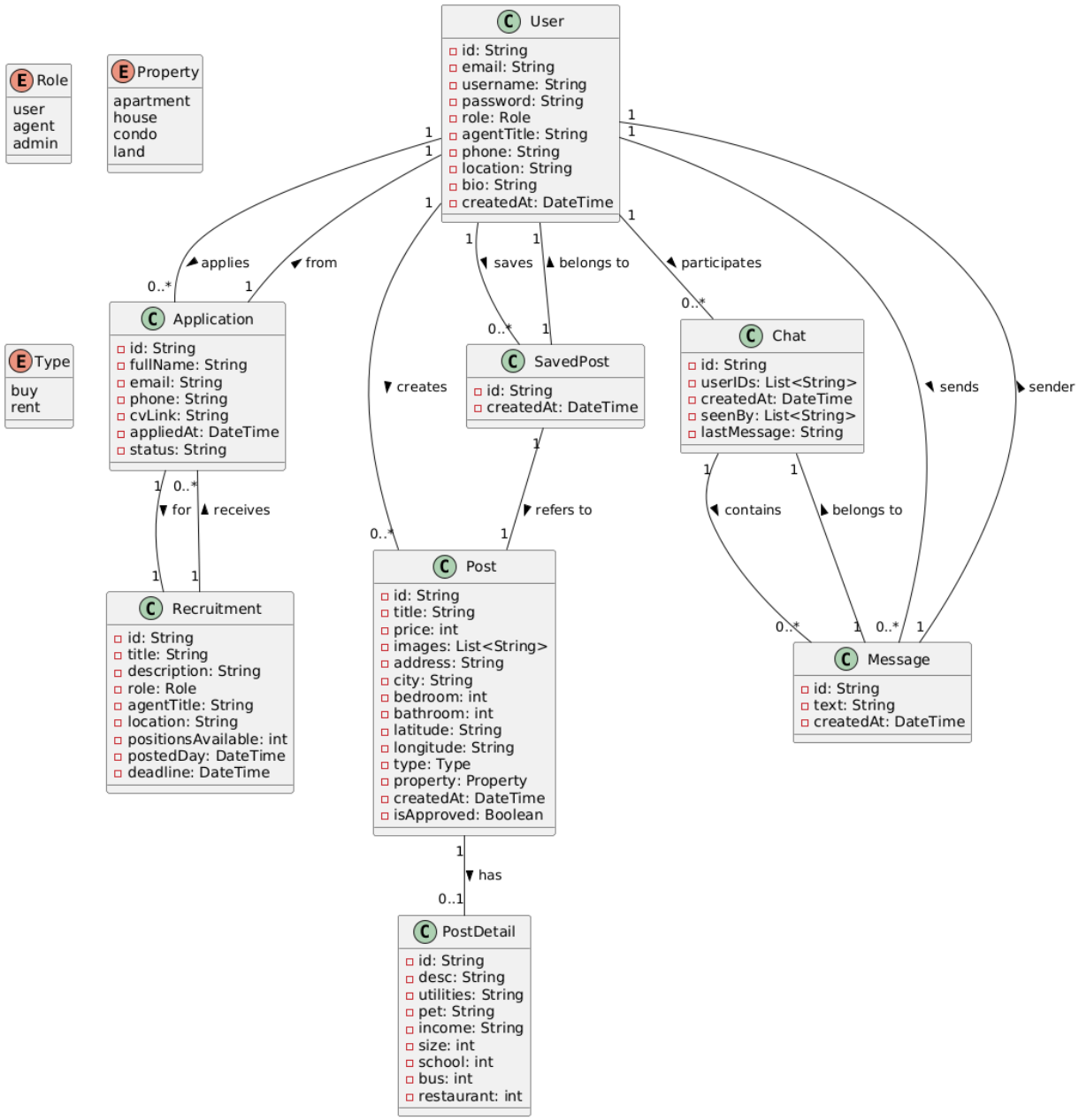
- **Real-time Communication:** WebSocket
- **Email:** Nodemailer
- **AI Chat Integration:** Crisp Chat

Development Phases

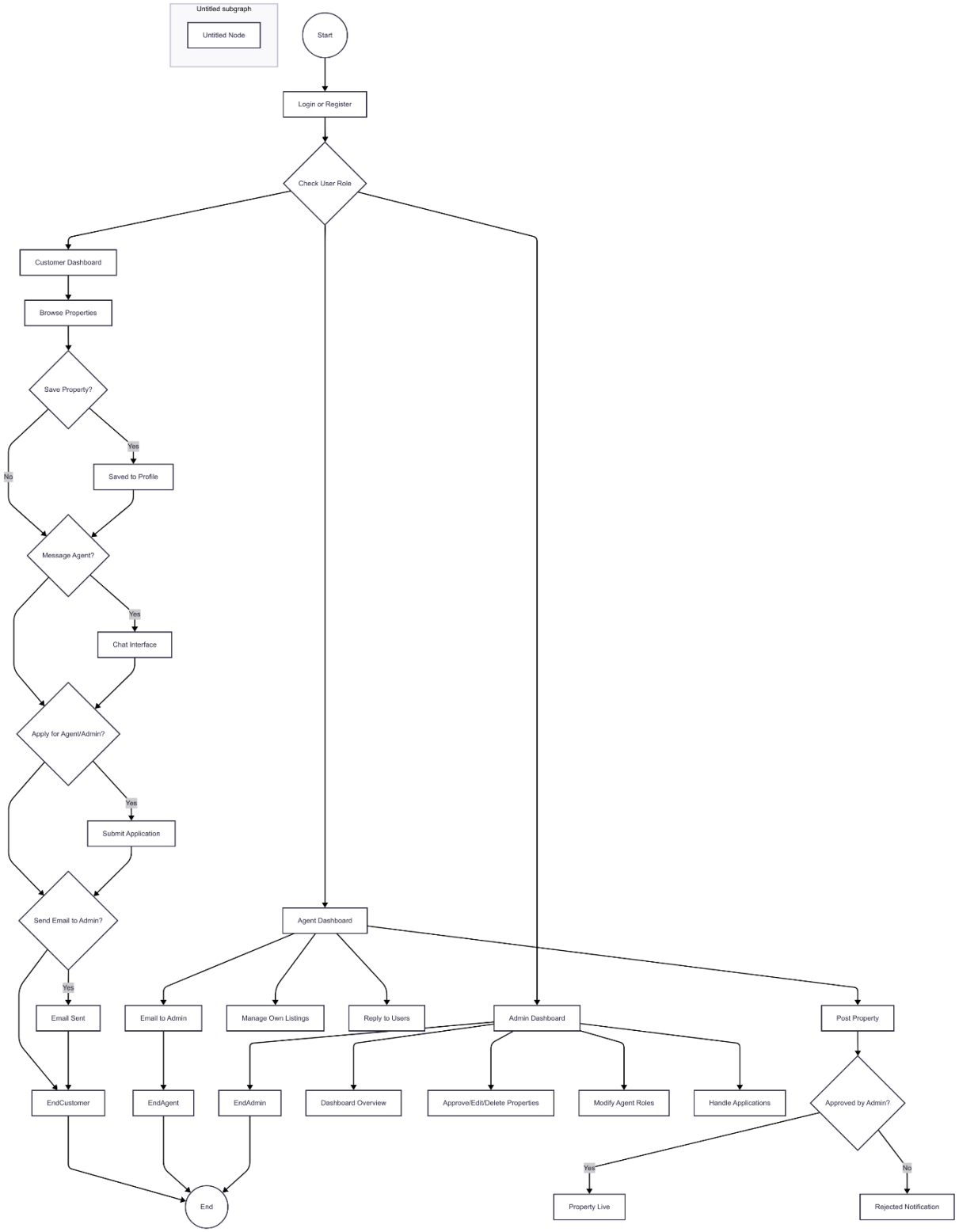
1. **Requirements Gathering** — Collecting detailed system needs and specifications.
2. **Planning** — Utilizing the Rapid Application Development (RAD) model for iterative progress.
3. **UI/UX Design** — Creating responsive and user-friendly interfaces using SCSS.
4. **Backend and API Development** — Building RESTful services and database integration.
5. **Integration** — Connecting frontend components with backend services.
6. **Testing and Debugging** — Performing unit, integration, and system testing to ensure quality.
7. **Deployment and Documentation** — Launching the application and preparing comprehensive documentation.

Design Models

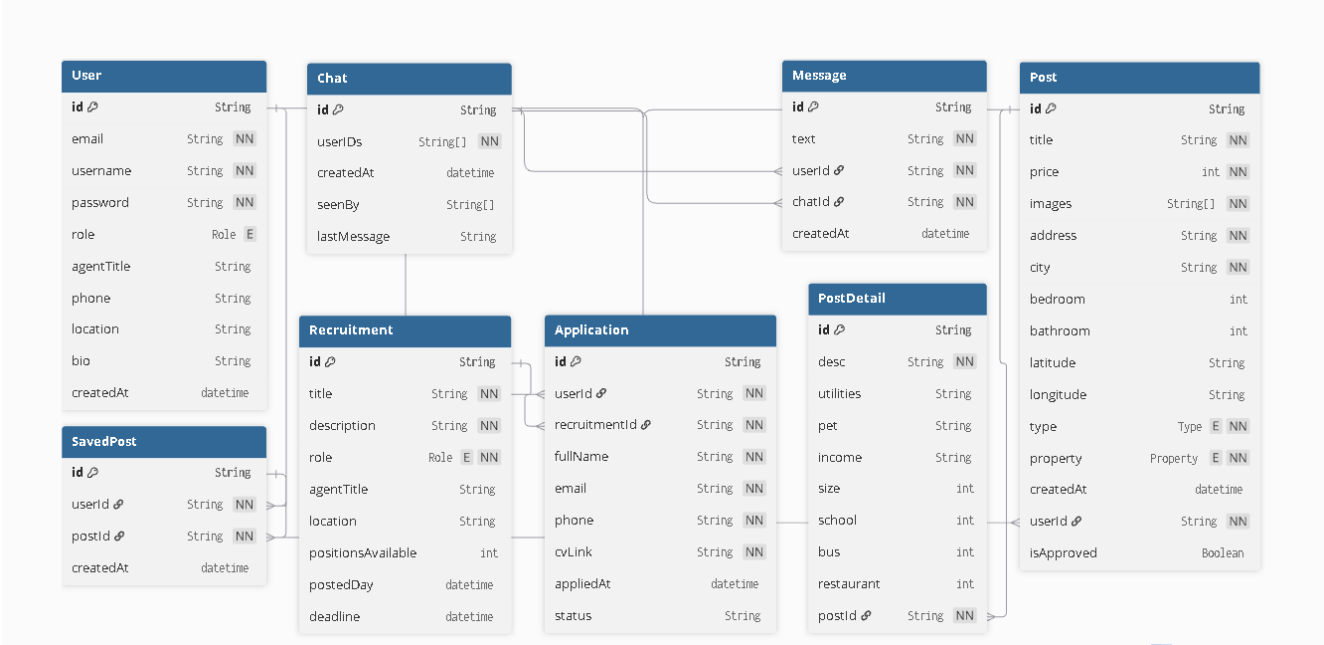
- **Use Case Diagram** —



- **Flowcharts —**


Untitled subgraph
Untitled Node
Start
Login or Register
Check User Role
Customer Dashboard
Browse Properties
Save Property?
Yes
No
Saved to Profile
Message Agent?
Yes
Chat Interface
Apply for Agent/Admin?
Yes
Submit Application
Send Email to Admin?
Yes
Email Sent
EndCustomer
Agent Dashboard
Email to Admin
Manage Own Listings
Reply to Users
EndAgent
EndAdmin
Admin Dashboard
Dashboard Overview
Approve/Edit/Delete Properties
Modify Agent Roles
Handle Applications
Post Property
Approved by Admin?
Yes
No
Property Live
Rejected Notification
End

- **ER Diagram** —



7. Implementation / Development

Key Features by User Roles

Customer

- View, filter, and save properties
- Chat with agents and administrators
- Apply for agent recruitment
- Contact admin via email form
- Receive automatic email replies

Agent

- Post properties (subject to admin approval)
- Edit, delete, or mark properties as sold
- Respond to user messages
- Apply for recruitment (admin-level roles)
- Update personal profile
- Contact administrators via email

Admin

- Access dynamic dashboard with user, post, and agent insights
- Approve, edit, inactive, or mark any property
- Change agent roles and permissions
- Deactivate or reactivate agents
- Manage recruitment posts and applicant approval
- Make sold post visible in list

Sample Code Snippet – Prisma Schema

```
model User {
  id       String @id @default(auto()) @map("_id") @db.ObjectId
  email    String @unique
  username String @unique
  role     Role   @default(user)
  // ...other fields
}

model Post {
  id       String @id @default(auto()) @map("_id") @db.ObjectId
  title    String
  price    Int
  user     User   @relation(fields: [userId], references: [id])
  userId   String @db.ObjectId
}
```

System Architecture

The application architecture follows a client-server model:

Client (React) → API (Express.js) → ORM (Prisma) → Database (MongoDB)

- Real-time communication is enabled through **WebSocket** for live chat functionality.
- The system ensures a smooth flow of data between user interfaces and persistent storage using RESTful APIs and Prisma as an abstraction over MongoDB.

Database Schema

The database schema includes models for:

- **User**
- **Post**
- **Message**
- **Chat**
- **Recruitment**
- **Application**

The detailed schema is defined in `schema.prisma` and is used to generate and manage MongoDB collections using Prisma ORM.

8. Results / Analysis

- Successfully implemented role-based access control for users (Customer, Agent, Admin)
- Completed all CRUD operations for property management
- Functional live chat between users, agents, and administrators
- Integrated Crisp Chat for AI-based customer interaction
- Email communication system with auto-reply functionality
- Dynamic admin dashboard featuring real-time data visualization with charts

9. Challenges Faced

- Prisma with MongoDB caused type mapping issues in early development
- Real-time communication across multiple user roles was complex
- Ensuring proper access control between agents and admins required strict validation
- Handling image uploads and file validation on both frontend and backend
- Automating email replies and syncing notifications with the frontend

Solutions

- Referred to official Prisma and MongoDB documentation for schema configuration
- Implemented robust middleware functions for role-based access validation
- Used JWT tokens to secure API endpoints and user sessions
- Followed Crisp Chat integration guidelines for stable communication
- Developed fallback UI components to handle real-time errors and failed network requests

10. Conclusion

This project demonstrates the successful development of a full-stack real estate web platform that combines user interaction, data management, and administrative control using role-based access.

The use of the Rapid Application Development (RAD) model enabled efficient prototyping and continuous iteration. Features like real-time chat, dynamic dashboards, and intelligent support systems contribute to a platform that is both user-friendly and scalable.

11. Future Scope

- Integrate a payment system for property bookings and transactions
- Add multilingual support (e.g., Bengali and English) for wider accessibility
- Develop a mobile application using React Native
- Introduce advanced AI support using NLP for smarter auto-replies
- Enable video walkthrough support for property listings