

**A PROJECT REPORT**  
**ON**  
**“ Legal Connect”**

Project report submitted in partial fulfilment of the requirement for the award of the degree

of

**Bachelor of Science**

**In**

**Computer Science and Cloud Computing**

**By**

**A.S.KUSHAL**

**UID - (111722043016)**

**G.UMESH**

**UID - (111722043024)**

Under the Guidance of

**Mr. P. V. JAYA KRISHNA**

**Assistant Professor**



**DEPARTMENT OF COMPUTER SCIENCE AND CLOUD COMPUTING**

**LOYOLA DEGREE AND PG COLLEGE**

**Old Alwal, Secunderabad - 500010**

**(Autonomous and affiliated to Osmania University)**

**Re-accredited by “NAAC” with “A” Grade**

**“A College with Potential for Excellence” by UGC**

**2024 – 2025**

# LOYOLA ACADEMY

Old Alwal, Secunderabad – 500010

(An Autonomous Degree College affiliated to Osmania University)

Accredited by “NAAC” with “A” Grade

A “College with Potential for Excellence” by UGC.



Department of BSc. Computer Science and Cloud Computing

## CERTIFICATE

This is to certify that the project report entitled for “**Legal Connect**” is a record of Bonafide work carried out during 3rd Year, 6th Semester as Partial fulfilment for the award of **DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND CLOUD COMPUTING** during the academic year 2024-25.

The results embodied in this project report have not been submitted to any other University of any Degree or Diploma.

**NAME: A. S. KUSHAL & G. UMESH**

**CLASS: DCSCC**

**UID: 111722043016 & 111722043024**

Mr. P. V. Jaya Krishna

**INTERNAL GUIDE**

Mr. K. Siva. Rama. Krishna

**HEAD OF THE DEPARTMENT**

**EXTERNAL EXAMINER**

**PRINCIPAL**



# LOYOLA ACADEMY

(DEGREE & PG COLLEGE)

OLD ALWAL, SECUNDERABAD - 500 010, TELANGANA, INDIA

(Autonomous and Affiliated to Osmania University)

Re-accredited with 'A' Grade (III Cycle) by NAAC

A "College with Potential for Excellence" by UGC

www.loyolaacademyugpg.ac.in Ph: 040-27862363/27860077 Fax: 040-27867939

## CERTIFICATE

This is to certify that **A.S.KUSHAL & G.UMESH** bearing Roll No. **111722043016 & 111722043024** from **Computer Science and Cloud Computing** department has successfully completed his Major Project work entitled "**Legal Connect**" at **Incubation Centre, Loyola Academy Degree and PG college**. During this project work his performance was **satisfactory**.

**Dr Santhi Chebiyyam**

**Incubation & IIC Coordinator**

## DECLARATION

We hereby declare that the project entitled “**Legal Connect**” done by **A.S.KUSHAL**(111722043016) and **G.UMESH**(111722043024) submitted to, **Mr. P. V. Jaya Krishna** in B.sc Computer Science and Cloud Computing, Loyola Academy Degree and PG College is a record of original work done by us.

The project has been successfully completed and submitted in partial fulfilment of the requirements for the award of “**DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND CLOUD COMPUTING**” from “**Loyola Academy Degree and PG College**”, affiliated to “**Osmania University, Hyderabad**” in an authentic way and has not been submitted in any other university or institution for the award of degree or diploma.

**A.S. KUSHAL**

(111722043016)

**G. UMESH**

(111722043024)

Mr. P. V. Jaya Krishna

**INTERNAL GUIDE**

Mr. K. Siva. Rama. Krishna

**HEAD OF THE DEPARTMENT**

**EXTERNAL EXAMINER**

**PRINCIPAL**

## **ACKNOWLEDGEMENT**

The satisfaction that accompanies the successful completion of this major project would be in complete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. I consider myself privileged to express gratitude and respect towards all those who guided us through the completion of this project.

On successful completion I convey thanks to my project guide **Mr. P. V. JAYA KRISHNA** of the Department of CSCC for providing encouragement, constant support and guidance which was of great help to complete this project successfully.

I am grateful to **Mr. K. SIVA RAMA KRISHNA** Head of the Department of for giving me the support and encouragement that was necessary for the completion of this project.

## INDEX

CHAPTERS	TOPICS	PAGES
	ABSTRACT	i
1.	<b>INTRODUCTION</b> 1.1 Introduction 1.2 Implementation	1-2
2.	<b>LITERATURE SURVEY</b> 2.1 Literature Review	3
3.	<b>SYSTEM FEASIBILITY</b> 3.1 Introduction 3.2 Existing System 3.3 Disadvantages of Existing System 3.4 Proposed System 3.5 Advantages of Proposed System	4-7
4.	<b>SYSTEM ANALYSIS</b> 4.1 The Study of the System 4.2 Input and Output Representation 4.3 Technology Used	8-10
5.	<b>SYSTEM ARCHITECTURE &amp; DESIGN</b> 5.1 System Architecture 5.2 System Design 5.2.1 UML Diagram 5.2.1 Usecase Diagram 5.2.2 Sequence Diagram 5.2.3 Class Diagram	11-16

	5.2.4 Activity Diagram 5.2.5 Deployment Diagram	
6.	<b>SYSTEM REQUIREMENTS</b> 6.1 Hardware Requirements 6.2 Software Requirements	17
7.	<b>SYSTEM IMPLEMENTATION</b> 7.1 Source Code 7.2 Source Code 7.3 Key Files and Their Functionality 7.4 Sample Code Snippet	18-26
8.	<b>SYSTEM TESTING &amp; VALIDATION</b> 8.1 Introduction 8.2 Levels of Testing	27-28
9.	<b>OUTPUT SCREENS</b>	29-34
10.	<b>CONCLUSION</b>	35
11.	<b>REFERENCES</b>	36

## **ABSTRACT**

Legal Connect is an interactive question and answer platform tailored for legal discussions, offering a unique hybrid model that combines the insight of verified legal professionals with the intelligence of AI-powered suggestions. The platform enables users to post legal queries, engage in threaded discussions, and receive support from both certified lawyers and @LegalAI, a smart assistant powered by the Gemini API. The motivation behind this project stems from the challenges faced by the general public in accessing legal advice. Traditional systems often require time, money, and prior legal knowledge. Legal Connect simplifies the process through a role-based system where users can ask questions, lawyers can provide verified answers, and admins manage user and content authenticity. Real-time notifications, lawyer profiles, direct messaging, and AI-integrated smart replies help ensure that legal help is accessible, reliable, and community-driven. From a technical standpoint, the system is built using a Flask backend, Firebase for user management and data storage, and Google's Gemini AI for natural language processing. The platform also features security measures such as role-based access control, token authentication, and input sanitization. By bridging the gap between legal professionals and the public, Legal Connect empowers users to seek, share, and find trusted legal information within a secure and structured environment.



# CHAPTER 1

## INTRODUCTION

With the increasing complexity of laws and the limited accessibility of legal services, there has been a growing demand for platforms that offer legal assistance in an interactive, accessible, and affordable manner. Legal Connect is a digital platform that allows users to post legal queries, participate in threaded discussions, and receive guidance from both AI systems and verified legal professionals.

The primary objective of Legal Connect is to bridge the gap between the general public and professional legal advice by creating a community-driven, AI-augmented legal platform. In doing so, it supports three major user roles — end-users, lawyers, and administrators — ensuring structured communication, security, and reliable content moderation.

The platform leverages Google's Gemini AI to provide smart replies and generate initial guidance to users while awaiting responses from certified lawyers. This ensures immediate support and continuous engagement. Additionally, it facilitates lawyer profiles, one-on-one messaging, notifications, and an upvote-based system to surface the most useful content.

Legal Connect promotes a transparent and scalable environment where legal knowledge becomes democratized and easily accessible to everyone.

## IMPLEMENTATION

### MODULES:

- User Module
- Lawyer Module
- Admin Module
- AI Assistant Module

### MODULE DESCRIPTIONS:

**User Module:** Users can register and log in to the platform through secure Firebase Authentication. Once authenticated, users are able to post questions, view responses, engage in discussions, and upvote helpful content. Users can also follow specific lawyers and receive notifications for updates and answers.

**Lawyer Module:** Lawyers must be verified by the admin before they can answer questions. After verification, lawyers can respond to legal queries, create public profiles that showcase their expertise, and engage with users via threaded answers and direct messages. The platform helps legal professionals build a reputation through upvotes and accepted answers.

**Admin Module:** Admins oversee the entire platform, verify lawyer credentials, manage user roles, monitor flagged content, and ensure the overall health and security of the system. The admin panel provides an overview of system activity and offers controls to moderate users and content.

**AI Assistant Module:** LegalAI, powered by Gemini API, assists users by generating initial insights and responses to legal questions using natural language processing. This feature is especially helpful when no lawyer is immediately available. The AI suggestions are clearly marked and never override human expert responses.

This modular implementation promotes extensibility, security, and seamless integration of both human and AI-powered legal support.

## CHAPTER 2

### LITERATURE SURVEY

#### 2.1 Literature Review

1. **Digital Legal Services and AI Support:** With the rapid growth of legal tech, multiple platforms such as Rocket Lawyer and LegalZoom have started offering AI-assisted legal services. However, most of these services are either premium or limited in scope. Studies show that hybrid systems combining community-driven input with verified professionals create more trust in users and lead to better engagement.
2. **Question & Answer Communities:** Platforms like Stack Overflow and Quora have demonstrated the power of user-generated Q&A communities. Literature on such models suggests that an upvote-based system, combined with expert validation, yields higher-quality answers and promotes community moderation.
3. **AI in Legal Guidance:** Research by Google and OpenAI in natural language models emphasizes the effectiveness of contextual understanding for legal queries. NLP-powered assistants improve accessibility and speed but still need human verification to ensure legal validity.
4. **Firebase for Authentication and Cloud Storage:** Case studies on Firebase-backed applications reveal how serverless architecture improves scalability and real-time synchronization for user authentication, message exchanges, and file storage.
5. **Threaded Discussion Forums:** Academic insights into forum-based design patterns stress the importance of user engagement tools like notifications, reply nesting, and role-based visibility to foster better knowledge exchange.

These studies support the architecture and design decisions made for Legal Connect and justify the integration of AI, cloud storage, and role-based systems for an accessible and scalable legal platform.

## **CHAPTER 3**

### **SYSTEM FEASIBILITY**

#### **3.1 Introduction**

System feasibility assesses how effectively a proposed solution addresses identified problems while conforming to technological, economic, and operational constraints. Legal Connect has undergone rigorous feasibility analysis across multiple domains to validate its practicality and scalability. This chapter explores the limitations of the current legal help systems and how Legal Connect presents a feasible and enhanced alternative.

Feasibility studies help assess whether the system is viable for development and long-term deployment. These studies examine technical resources, development timelines, operational compatibility, and cost-efficiency. For a domain as sensitive and essential as law, ensuring the system is both functional and secure is paramount.

#### **3.2 Existing System**

At present, individuals seeking legal support face significant barriers in accessing professional help. Digital legal services are often either costly, limited to document generation, or fail to offer real-time interaction. Many users resort to informal channels such as social media groups, where legal discussions lack moderation, credibility, and accuracy. Moreover, these platforms rarely provide a structured method for seeking guidance or verifying the credibility of contributors.

In most developing nations, public access to legal aid is also hindered by bureaucratic delays and the unavailability of qualified legal professionals in rural or remote areas. Mobile apps that attempt to deliver legal content often suffer from poor UX design and lack dynamic, conversational support. Thus, legal help continues to remain inaccessible for the majority of the population who require quick, affordable, and credible legal guidance.

Services such as LegalZoom or Rocket Lawyer, while functional, are designed primarily for users who can afford subscriptions or one-time fees. Free forums and legal advice threads on platforms like Reddit or Quora are not regulated by professionals, leading to misinformation and misinterpretation. As a result, users with genuine legal concerns often struggle to find quick, reliable, and context-specific advice.

### **3.3 Disadvantages of Existing System**

- No instant AI suggestions or legal context for public queries.
- No formal mechanism for verified lawyer involvement.
- Limited interactivity or user engagement tools.
- No central system for trusted legal community discussions.
- Lack of structured moderation and content validation.
- Absence of role-based security and accountability features.
- Fragmented communication between users and professionals.
- Inaccessible for users from non-technical or low-literacy backgrounds.

### **3.4 Proposed System**

Legal Connect introduces an innovative and scalable legal assistance platform designed to close the accessibility gap. It merges AI-powered smart suggestions with expert legal responses from verified lawyers. The system is structured around role-based access, offering personalized experiences for users, lawyers, and administrators.

The platform is user-centric, with an intuitive interface that supports both novice users and legal professionals. AI integration ensures that users get immediate assistance and context before lawyers respond, enhancing efficiency. In addition to solving immediate queries, the platform also builds a searchable knowledge base for future reference.

**Key components of the proposed system include:**

- A web-based user interface for submitting legal questions.
- A lawyer portal for responding to user queries and managing profiles.
- An integrated AI assistant powered by Gemini for contextual suggestions.
- Firebase backend for authentication, real-time messaging, and data storage.
- An admin dashboard for managing content, verifying users, and overseeing platform safety.
- Engagement features like notifications, threaded replies, upvotes, and follower systems.
- Secure messaging between users and lawyers with history tracking.
- Real-time analytics and feedback mechanisms for system improvement.

**3.5 Advantages of Proposed System**

- Seamlessly blends human expertise with AI-powered real-time suggestions for legal guidance.
- Facilitates a structured and moderated digital forum for legal queries.
- Ensures accountability and data security through role-based access control.
- Built on a scalable and secure infrastructure using Firebase and cloud deployment.
- Enhances legal awareness by fostering transparent discussions between users and professionals.
- Provides a trustworthy platform for underrepresented individuals to seek legal help.
- Enables faster response times, efficient communication, and credible legal support.
- Allows real-time feedback, analytics tracking, and system adaptability.
- Encourages community participation through votes, replies, and profile-based recognition.
- Lowers the barrier to legal access by integrating intelligent automation with human validation.

The proposed system demonstrates strong feasibility by leveraging current technological capabilities, addressing gaps in the existing legal support landscape, and creating a

sustainable, community-driven solution. With the combined strength of AI and human expertise, Legal Connect aspires to revolutionize the accessibility and delivery of legal services in the digital age.

## CHAPTER 4

### SYSTEM ANALYSIS

#### 4.1 The Study of the System

- System analysis involves understanding and designing the interactions between the different components and users of Legal Connect. This includes defining user roles, identifying input and output patterns, and understanding data flow between system modules. Legal Connect supports modular access for users, lawyers, admins, and AI services, ensuring streamlined communication and scalable operations.
- The core goal of system analysis in this context is to transform user requirements into detailed functional specifications. This includes analyzing both functional and non-functional requirements such as performance, scalability, reliability, and maintainability. Legal Connect's architecture is designed to facilitate seamless integration, ensure security compliance, and offer user-friendly legal support services.

#### Logical Design

The logical design outlines how data and processes interact. Legal Connect follows a three-tier architecture:

- **Presentation Layer (Frontend – Bootstrap UI):** Responsible for presenting information to the user and capturing user input. This includes responsive forms, buttons, dashboards, and content views.
- **Business Logic Layer (Flask Backend):** Handles all business logic including request processing, session management, role-based control, and communication with the database.
- **Data Layer (Firebase Firestore & Storage):** Manages persistent data including user profiles, question threads, lawyer responses, and media files. Ensures data availability, replication, and access control.

This abstraction enhances maintainability, scalability, and debugging processes. Modular APIs and service segregation help in implementing updates without affecting the entire system.

#### Physical Design

The physical design details user input forms, data storage patterns, and authentication flows. All user data is securely stored in Firebase, and communication between components is handled through REST APIs secured with tokens.



- User actions such as submitting questions, replying to queries, or updating profiles are translated into API requests that interact with backend logic. Real-time data sync is maintained for notifications and threaded discussions using Firebase's real-time capabilities. Data redundancy and backup policies are also defined to prevent data loss.

## **4.2 Input and Output Representation**

### **Input Design**

Legal Connect emphasizes intuitive and secure data entry points. Each role—User, Lawyer, Admin—has designated access levels and form fields that align with their responsibilities.

- Secure registration and login forms with password encryption
- Question submission interface for users, supporting rich text and attachments
- Lawyer response panel with markdown-enabled response editor
- Admin verification forms for new user validation and content moderation
- AI reply input triggers based on question parsing
- Token-based interaction validation to prevent spam or misuse

### **Output Design**

The platform provides comprehensive and organized output formats, ensuring that the users receive contextual and digestible information.

- Threaded discussion views for tracking question-response chains
- Public lawyer profiles with bio, verification badge, and case count
- Response feed with real-time upvotes and sorting algorithms
- Notifications and alerts (message, follow, reply) with real-time indicators
- Accepted answers displayed prominently for user clarity
- Activity logs and admin dashboards for backend transparency

### **Design Objectives**

- Simplify data entry and validation with form-level feedback
- Prevent unauthorized access or injection through input sanitization
- Ensure UI responsiveness across devices and browsers
- Improve overall user experience with visual and auditory cues
- Maintain system robustness through layered error handling and feedback

## **4.3 Technology Used**

Legal Connect integrates modern technologies to create a fast, secure, and scalable platform.

### **Frontend**

- Bootstrap: For responsive UI components and grid layout

- HTML5 & CSS3: For structuring and styling user interfaces
- JavaScript: For dynamic UI behaviour and API interaction

## **Backend**

- Flask (Python-based micro-framework): To build lightweight REST APIs, session handling, and server-side logic
- Jinja templating engine for server-rendered views where required

## **Database**

- Firebase Firestore: For storing user data, messages, posts, and activity logs
- Firebase Storage: For storing user-uploaded media (PDFs, images)

## **Authentication**

- Firebase Auth: For user management, password resets, email verification, and multi-provider login

## **AI Integration**

- Google Gemini AI (Natural Language API): For parsing user questions, suggesting AI-generated responses, and improving query categorization

## **APIs**

- REST APIs for all client-server interactions, including AI calls and real-time updates
- **JSON payloads for secure, structured data exchange**

## **Hosting**

- Firebase Hosting for frontend deployment
- Local Flask Server during development phase and containerized deployment options for production

## **Security & Performance**

- Role-Based Access Control (RBAC) for managing permissions across user types
- Token-based session handling to secure API endpoints
- Input sanitization to prevent common vulnerabilities like XSS and SQL injection
- Real-time sync with Firebase for instant updates
- Scalable architecture to handle increasing user load
- Usage of CORS headers and HTTPS enforcement for secure communication
- Automated backups and recovery features for critical data

These technologies collectively ensure that Legal Connect is robust, user-friendly, and suitable for long-term deployment across different regions and user demographics.

## CHAPTER 5

### SYSTEM ARCHITECTURE & DESIGN

#### 5.1 System Architecture

Legal Connect is built using a modular and scalable architecture to support secure interactions between users, lawyers, and the admin. The system follows a 3-tier architecture model:

1. **Presentation Layer:** User interfaces built using HTML, CSS, and Bootstrap that enable intuitive navigation and interaction.
2. **Application Layer:** Backend logic implemented in Python Flask that handles routing, API calls, role-based operations, and AI-integration.
3. **Data Layer:** Firebase Firestore stores user data, questions, answers, messages, and logs, while Firebase Storage is used for media uploads.

Each user request flows from the client-side to the server-side (Flask), which interacts with Firebase and returns dynamic content back to the UI. Gemini AI integrates through API endpoints for smart replies.

**Architecture Flow Diagram:** (Will include a labeled diagram showing interaction between Users, Flask Server, Firebase, Gemini AI, and Frontend.)

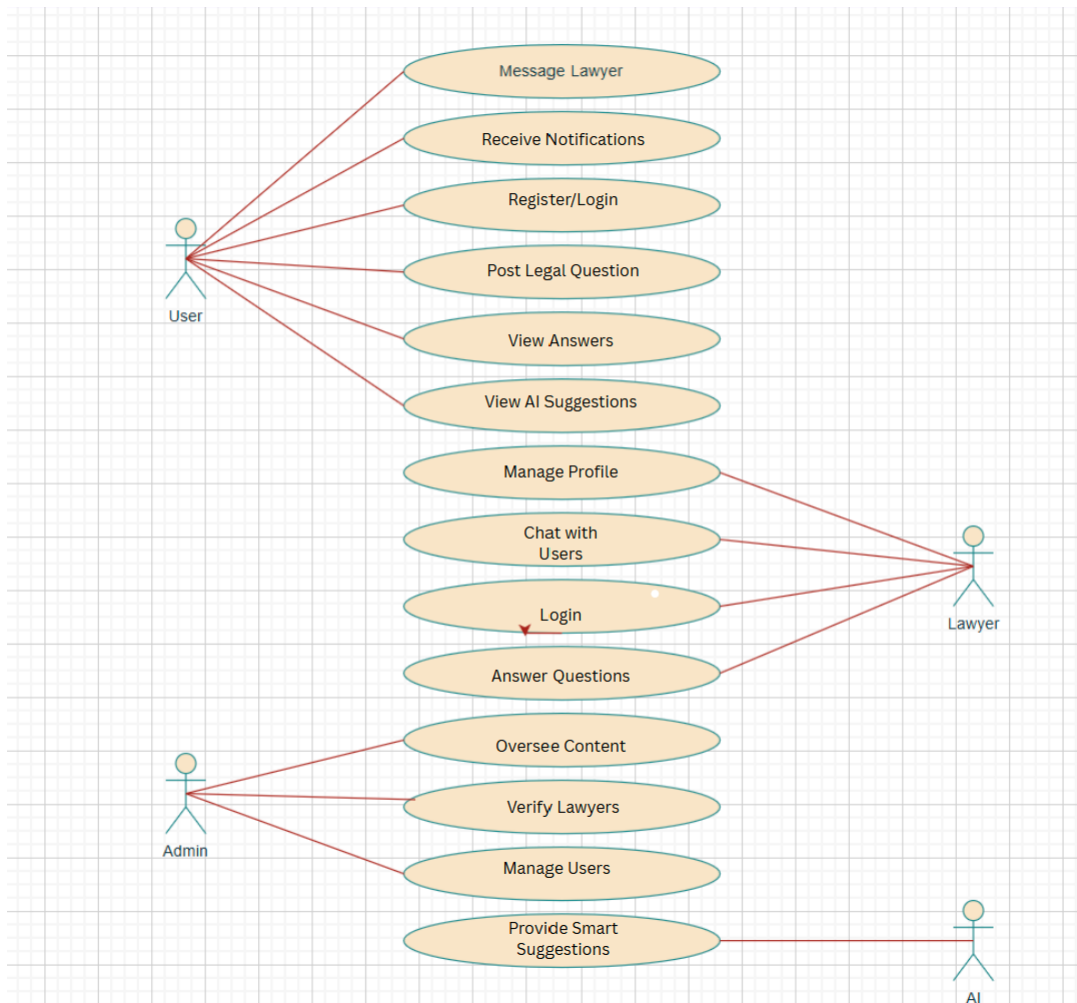
#### 5.2 System Design

The system was designed to be modular, scalable, and easy to navigate. The design phase involved creating various UML diagrams to represent the core components and their interactions.

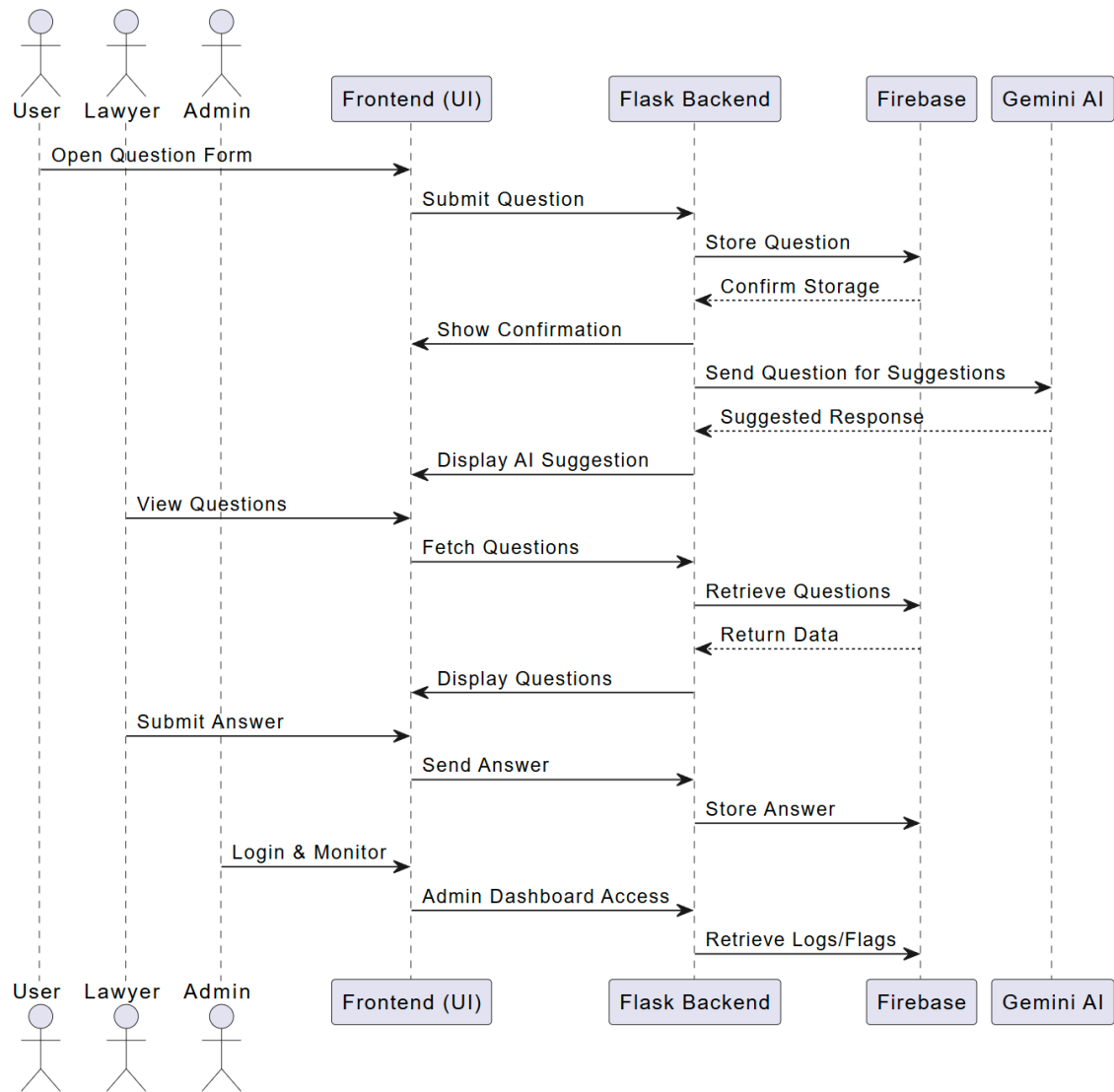
##### 5.2.1 UML Diagrams

**Use Case Diagram** Shows the major use cases for different actors in the system: User, Lawyer, Admin, and AI Assistant.

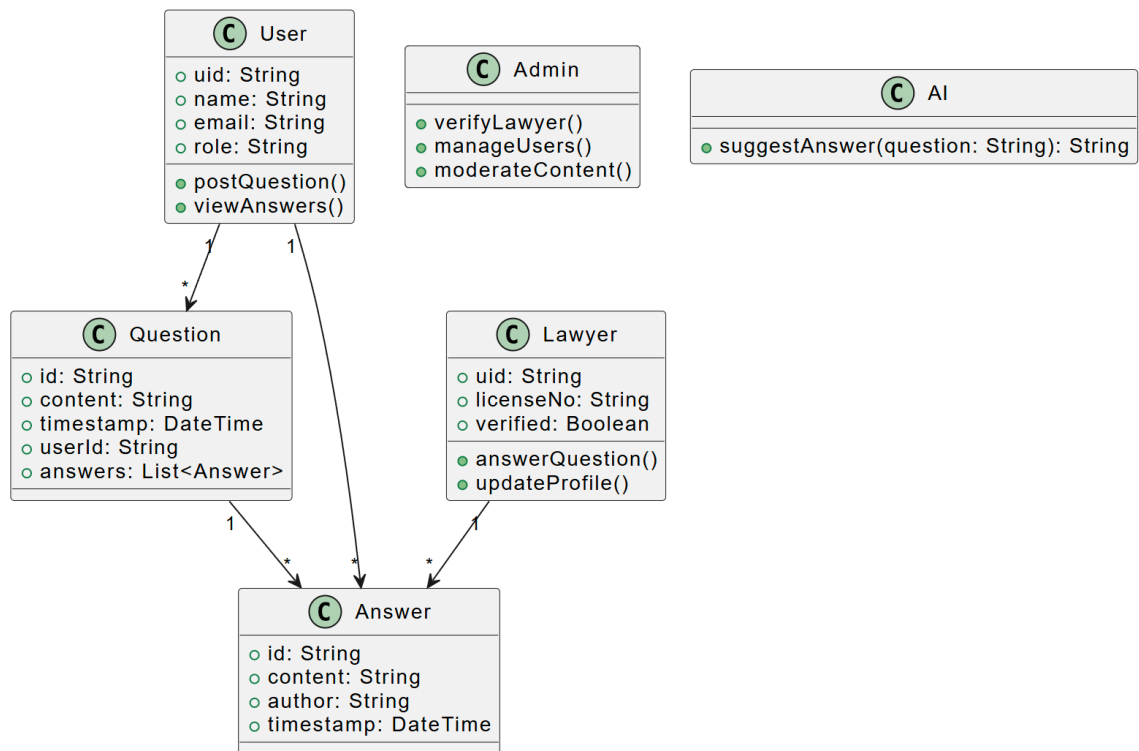
- User can: Register, Login, Post Question, View Answers, Chat with Lawyer, Upvote, Accept Answer
- Lawyer can: Register, Wait for Verification, Answer Questions, View Profile, Message Users
- Admin can: Verify Lawyers, Moderate Content, Manage Users
- AI Assistant can: Generate Smart Replies to Questions



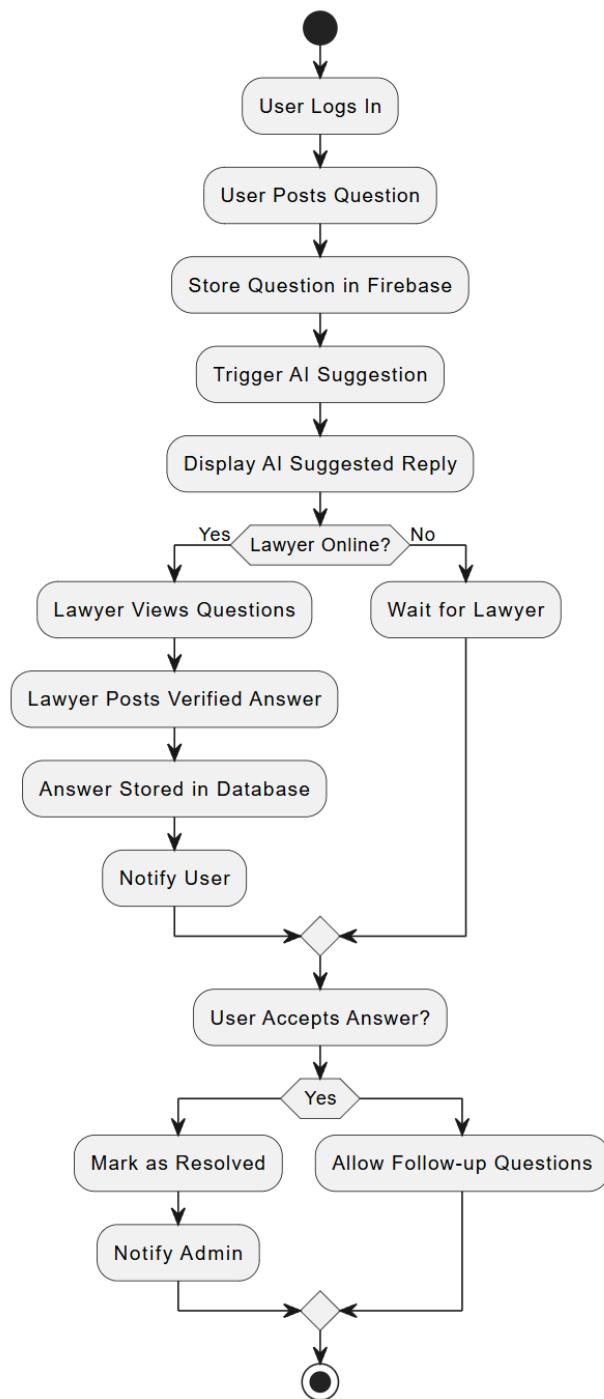
**Sequence Diagram** Illustrates the interaction between User, Flask App, Firebase, and AI Module when a question is asked and answered. It shows the order of calls like: Post Question → Store → Generate AI Reply → Notify Lawyers.



**Class Diagram** Displays main classes: User, Question, Answer, Message, Lawyer Profile, with attributes and relationships (e.g., User → [has-many] → Questions, Answers).

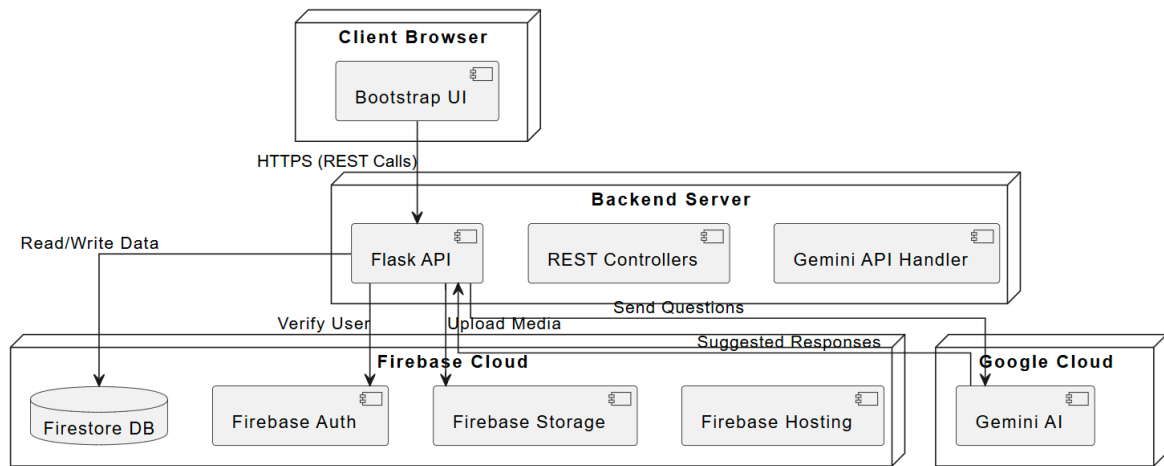


**Activity Diagram** Shows the flow of activity from user login to question posting, answering, and marking answers as accepted.



**Deployment Diagram** Depicts the physical deployment of the system, including the client device, Flask backend, Firebase backend, and external AI API (Gemini).

(Diagrams will be created and embedded in the final version.)





## CHAPTER 6

### SYSTEM REQUIREMENTS

#### 6.1 Hardware Requirements

- **System** : Intel(R) Core(TM) i3 or above
- **Hard Disk** : Minimum 120 GB Input Devices : Keyboard, Mouse
- **RAM** : 4 GB or higher

#### 6.2 Software Requirements

- **Operating System** : Windows 10 / Linux / macOS
- **Backend** : Python 3.8+, Flask
- **Frontend** : HTML, CSS, JavaScript, Bootstrap
- **Database** : Firebase Firestore
- **Authentication** : Firebase Auth
- **Cloud Storage** : Firebase Storage
- **AI API** : Google Gemini AI (API Key required)
- **Dependencies** : Python libraries from requirements.txt
- **Browser** : Chrome, Firefox, or any modern browser

These specifications ensure compatibility with all modules of Legal\_Connect, providing smooth interaction between the user interface, backend processing, real-time database, and AI services.

## CHAPTER 7

### SYSTEM IMPLEMENTATION

#### 7.1 Overview

Legal Connect is built using a modular structure that separates responsibilities for routing, authentication, database interaction, and AI response. The platform is developed in Python using the Flask framework, integrates Firebase for backend services, and Gemini API for AI assistance.

#### 7.2 Source Code

##### Admin Page

```
{% extends "base.html" %}

{% block title %}Admin Panel{% endblock %}

{% block content %}

<div class="card premium mb-4">

    <div class="card-header bg-danger text-white d-flex align-items-center">

        <i class="bi bi-shield-lock me-2"></i>

        <h3 class="mb-0">Admin Control Panel</h3>

    </div>

    <div class="card-body">

        <div class="alert alert-info">

            <i class="bi bi-info-circle-fill me-2"></i>

            Welcome to the administration panel. Here you can manage users and platform
content.

        </div>

    </div>

</div>
```

```
<div class="row">

  <div class="col-md-12 mb-4">

    <div class="card shadow-sm">

      <div class="card-header bg-light d-flex justify-content-between align-items-center">

        <h4 class="mb-0"><i class="bi bi-person-badge text-primary me-2"></i>Lawyer
Verification</h4>

        <span class="badge bg-danger">{ { pending_lawyers|length } } Pending</span>

      </div>

      <div class="card-body">

        { % if pending_lawyers % }

        <div class="table-responsive">

          <table class="table table-hover">

            <thead class="table-light">

              <tr>

                <th>Username</th>

                <th>Email</th>

                <th>Registered On</th>

                <th>Action</th>

              </tr>

            </thead>

            <tbody>

              { % for lawyer in pending_lawyers % }

              <tr>

                <td>

                  <div class="d-flex align-items-center">
```

```

        <div class="rounded-circle bg-secondary text-white d-flex
justify-content-center align-items-center me-2"

        style="width: 32px; height: 32px; font-size: 0.8rem;
background: var(--gradient-{{ lawyer.id|modulo(3) }}) !important;">

            {{ lawyer.username[0] }}

        </div>

        <a href="{{ url_for('view_profile', user_id=lawyer.id) }}">{{
lawyer.username }}</a>

    </div>

</td>

<td>{{ lawyer.email }}</td>

<td>{{ lawyer.created_at.strftime('%b %d, %Y') }}</td>

<td>

    <a href="{{ url_for('verify_lawyer', user_id=lawyer.id) }}"
class="btn btn-success btn-sm">

        <i class="bi bi-check-circle-fill me-1"></i> Verify

    </a>

</td>

</tr>

{% endfor %}

</tbody>

</table>

</div>

{% else %}

<div class="text-center py-4">

    <i class="bi bi-check-circle text-success" style="font-size: 3rem;"></i>

    <h5 class="mt-3">All caught up!</h5>

```

```
<p class="text-muted">No pending lawyer verifications.</p>

</div>

{% endif %}

</div>

</div>

</div>

</div>

<div class="row">

  <div class="col-md-12">

    <div class="card shadow-sm">

      <div class="card-header bg-light d-flex justify-content-between align-items-center">

        <h4 class="mb-0"><i class="bi bi-people text-primary me-2"></i>User
Management</h4>

        <span class="badge bg-primary">{{ users|length }} Users</span>

      </div>

      <div class="card-body">

        <div class="table-responsive">

          <table class="table table-hover">

            <thead class="table-light">

              <tr>

                <th>Username</th>

                <th>Email</th>

                <th>Role</th>

                <th>Status</th>

                <th>Registered On</th>
```

```

        <th>Actions</th>

    </tr>

</thead>

<tbody>

    {% for user in users %}

        <tr>

            <td>

                <div class="d-flex align-items-center">

                    <div class="rounded-circle bg-secondary text-white d-flex justify-
content-center align-items-center me-2"

                        style="width: 32px; height: 32px; font-size: 0.8rem;
background: var(--gradient-{{ user.id|modulo(3) }}) !important;">

                            {{ user.username[0] }}

                        </div>

                        <a href="{{ url_for('view_profile', user_id=user.id) }}">{{
user.username }}</a>

                    </div>

                </td>

                <td>{{ user.email }}</td>

                <td>

                    {% if user.role == 'lawyer' %}

                        <span class="badge bg-success">Lawyer</span>

                    {% elif user.role == 'admin' %}

                        <span class="badge bg-danger">Admin</span>

                    {% else %}

                        <span class="badge bg-secondary">User</span>

                    {% endif %}

                </td>

```

```

</td>

<td>

    {% if user.is_verified %}

        <span class="badge bg-success">Verified</span>

    {% else %}

        <span class="badge bg-warning text-dark">Pending</span>

    {% endif %}

</td>

<td>{{ user.created_at.strftime('%b %d, %Y') }}</td>

<td>

    <div class="dropdown">

        <button class="btn btn-sm btn-outline-secondary dropdown-
toggle" type="button" data-bs-toggle="dropdown">

            Actions

        </button>

        <ul class="dropdown-menu">

            <li><a class="dropdown-item" href="{{ url_for('view_profile',
user_id=user.id) }}">

                <i class="bi bi-person-fill me-2"></i> View Profile

            </a></li>

            {% if user.role == 'lawyer' and not user.is_verified %}

                <li><a class="dropdown-item text-success" href="{{ {
url_for('verify_lawyer', user_id=user.id) }}">

                    <i class="bi bi-check-circle-fill me-2"></i> Verify

                </a></li>

            {% endif %}

        </ul>

```

```
</div>

</td>

</tr>

{ % endfor % }

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

<style>

.table th {

    font-weight: 600;

}

.table td {

    vertical-align: middle;

}

</style>

{% endblock % }
```

### 7.3 Key Files and Their Functionality

- **app.py:** Entry point of the application. Sets up Flask, routes, and Firebase configuration.
- **firebase\_config.py:** Initializes Firebase Admin SDK and Firestore client.
- **auth.py:** Handles login, registration, session tokens.
- **questions.py:** Handles posting, answering, upvoting, and accepting answers.
- **ai.py:** Communicates with Gemini API to generate smart legal responses.



## 7.4 Sample Code Snippet

### Firestore Initialization (firebase\_config.py):

```
import firebase_admin

from firebase_admin import credentials, firestore

cred = credentials.Certificate("path/to/serviceAccountKey.json")

firebase_admin.initialize_app(cred)

db = firestore.client()
```

### Gemini AI Integration (utils/gemini\_integration.py):

```
import requests

def get_legal_ai_reply(prompt):

    api_key = os.getenv("GEMINI_API_KEY")

    response = requests.post(

        "https://api.gemini.google.com/v1/generate",

        headers={"Authorization": f"Bearer {api_key}"},

        json={"prompt": prompt}

    )

    return response.json().get("reply", "No response from AI")
```

### Route for Asking a Question (routes/questions.py):

```
@app.route('/ask', methods=['POST'])

def ask_question():

    data = request.form
```

```
db.collection('Questions').add({  
  'title': data['title'],  
  'description': data['description'],  
  'user_id': session['user_id'],  
  'timestamp': firestore.SERVER_TIMESTAMP  
})  
return redirect('/dashboard')
```

This modular approach makes the application scalable, testable, and easy to maintain

## CHAPTER 8

### SYSTEM TESTING & VALIDATION

#### 8.1 Introduction

System testing is a crucial phase in the software development lifecycle, ensuring the entire Legal Connect platform works as intended. The testing process evaluates the complete and integrated application to verify that all requirements are met and functions perform under various conditions.

Testing was carried out at multiple levels to confirm:

- Functional correctness of features.
- Role-based access integrity.
- AI integration reliability.
- Real-time data updates.
- UI responsiveness across devices.

#### 8.2 Levels of Testing

##### Unit Testing:

- Each module such as login, posting a question, and AI reply generation was tested in isolation.
- Python's unittest and manual testing were used to verify form handling and API response logic.

##### Integration Testing:

- Tested combined workflows like: User posts question → Gemini replies → Lawyer answers.
- Verified interaction between Flask routes, Firebase Firestore, and Gemini API.

##### System Testing:

- Tested the platform end-to-end for Users, Lawyers, and Admins.
- Focused on registration flow, posting/answering questions, verifying lawyer roles, and direct messaging.

##### Black Box Testing:

- Test cases were created based on system requirements without knowledge of internal code.
- Scenarios like: invalid login, submitting blank question, unauthorized access attempts.

##### White Box Testing:

- Internal logic of Flask route handlers and database methods were tested.
- Ensured each condition, loop, and API response handler worked correctly.

**Validation:**

- Ensured role-based access control restricted Users from accessing Admin-only functionalities.
- Confirmed Gemini replies were returned only when API key and input prompt were valid.

Test Case	Description	Expected Result	Status
TC01 - User Login	Valid user credentials	Redirect to dashboard	Pass
TC02 - Invalid Login	Wrong credentials	Show error	Pass
TC03 - Post Question	Fill & submit form	Question saved	Pass
TC04 - Gemini AI Integration	Prompt AI on post	Display smart reply	Pass
TC05 - Lawyer Answer	Submit answer	Visible to all	Pass
TC06 - Admin Verification	Approve lawyer request	Access granted	Pass
TC07 - Role Restriction	Normal user accessing /admin route	Access denied	Pass

Legal Connect passed all test cases and demonstrated stability and responsiveness across all modules.

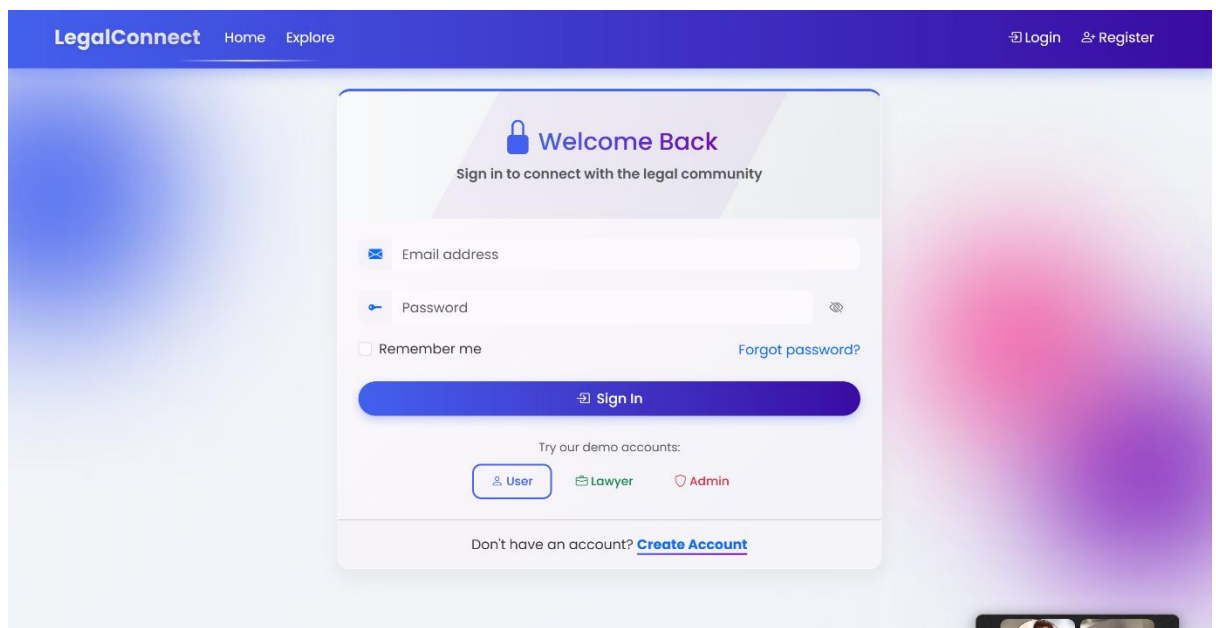
## CHAPTER 9

### OUTPUT SCREENS

Below are the main output screens from the Legal Connect platform that showcase the core functionalities of the system.

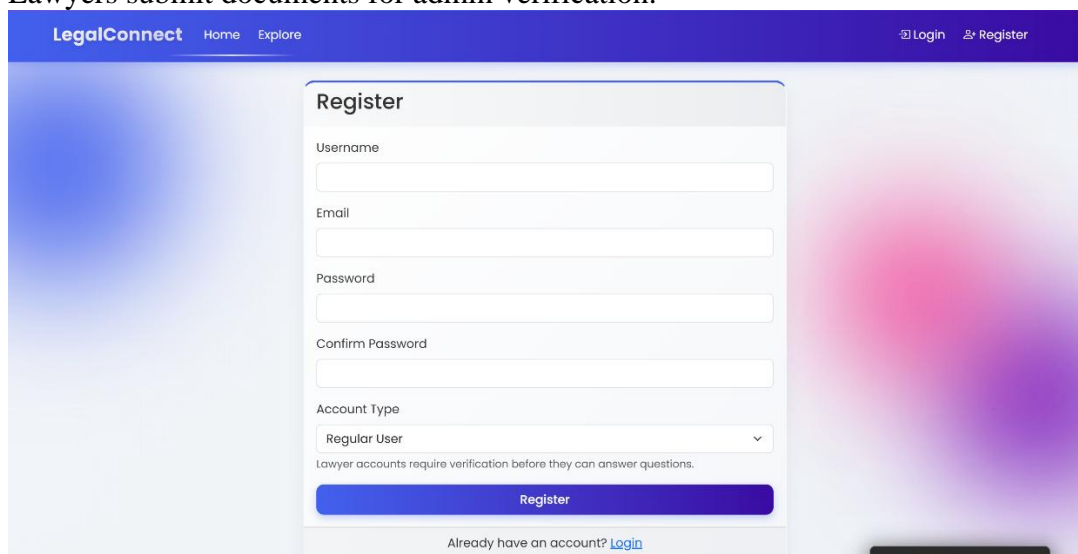
#### 1. Home/Login Screen:

- Allows users, lawyers, or admins to securely log into the platform



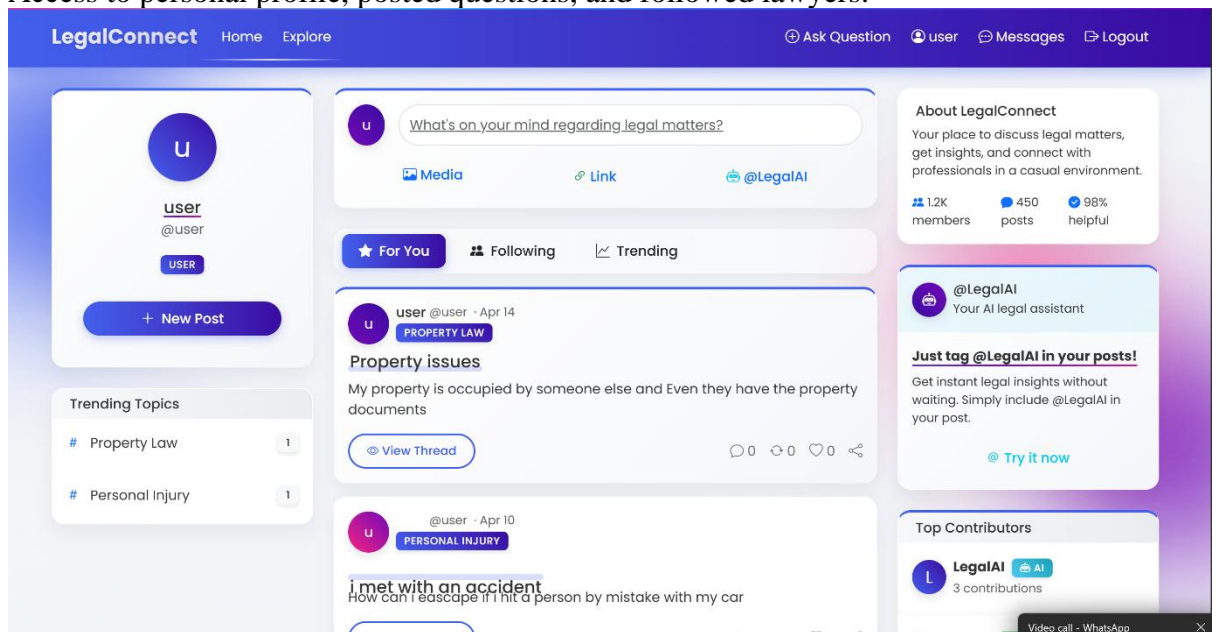
#### 2. Registration Screen:

- Separate registration options for Users and Lawyers.
- Lawyers submit documents for admin verification.



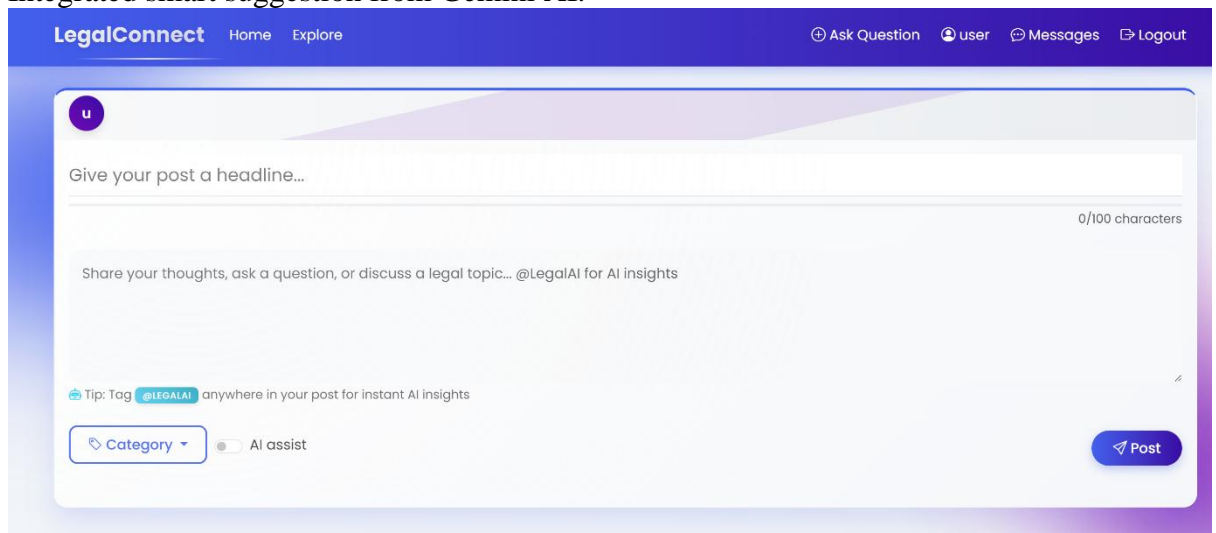
### 3. Dashboard:

- Users can view trending questions and top contributors.
- Access to personal profile, posted questions, and followed lawyers.



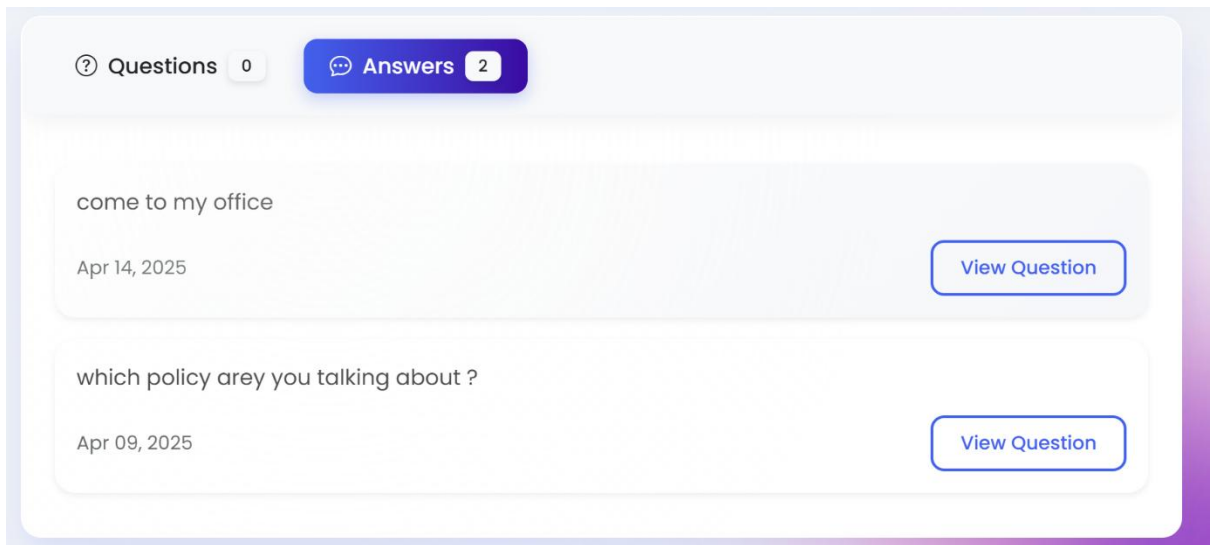
### 4. Ask a Question Page:

- Users post their legal questions with title, description, and tags.
- Integrated smart suggestion from Gemini AI.




### 5. Answer Page:

- Lawyers provide answers to questions.
- Users can reply, upvote, or mark an answer as accepted.




## 6. AI Response Display:

- Gemini-generated answer is shown below the user's question.

 **Get AI Legal Insights**

① Request instant AI analysis of this legal question

 **LEGALAI** AI-powered legal insights  
Apr 14 · 14:51


1. **SUMMARY:** This is a serious situation requiring immediate action. Someone else possessing your property and claiming ownership through documents suggests they may have a fraudulent deed or some other basis for their claim, which needs to be investigated thoroughly by a legal professional. You need to establish your rightful ownership and reclaim your property.




**LegalConnect** Home Explore

**Validity of Documents:** The core issue is determining if the documents the occupier possesses are legitimate. Forged deeds, errors in recording, or other fraudulent activities can create situations where someone appears to have legal ownership when they don't. **Your Proof of Ownership:** You need to gather all evidence of your ownership, including the original deed, purchase agreements, property tax records, mortgage documents, and any other relevant paperwork. **Adverse Possession:** In some jurisdictions, someone can claim ownership of property they've occupied openly and continuously for a specific period, even without a valid deed. This is called adverse possession and the rules vary significantly by location. **Legal Counsel:** This situation requires the expertise of a real estate attorney. They can analyze the documents, advise on your legal options, and represent you in court if necessary.

3. **PRACTICAL ADVICE:**

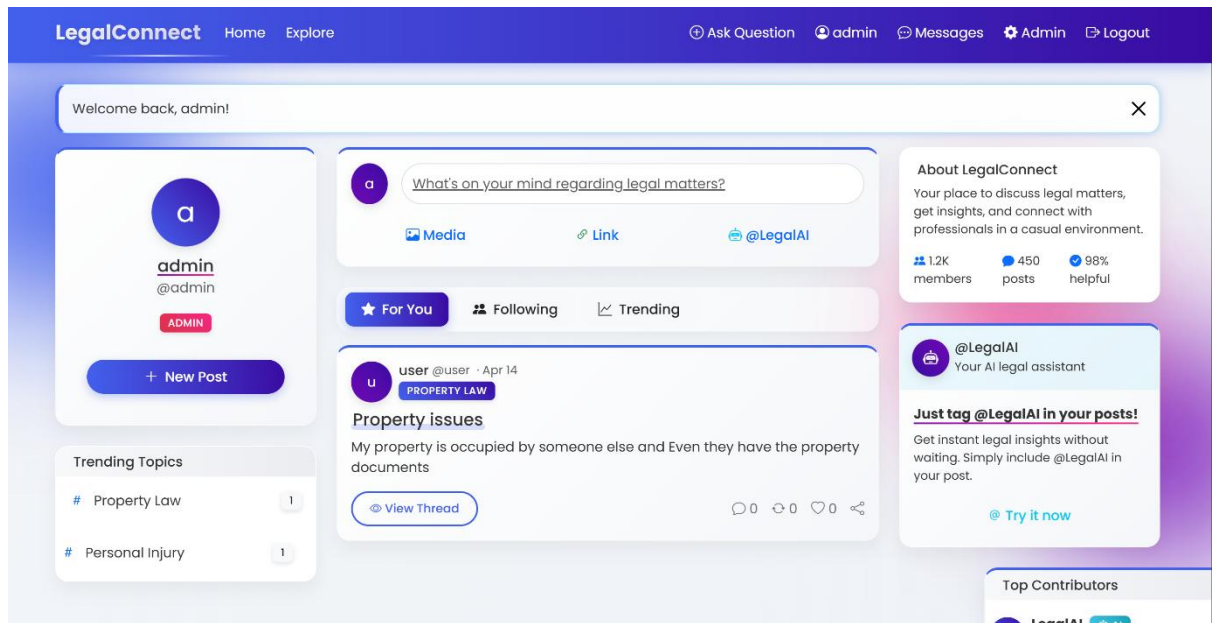
This isn't a DIY legal project. Contact a real estate attorney immediately. The longer you wait, the more difficult it may become to reclaim your property. Do not attempt to forcibly remove the occupier yourself, as this could lead to legal trouble for you. Your attorney will guide you through the proper legal process, which may involve filing a lawsuit to quiet title or an eviction action. Be prepared to provide your attorney with all documentation related to your ownership of the property.

 **Disclaimer:** This is general information, not specific legal advice. Please consult a qualified attorney for advice on your specific situation.

 AI GENERATED   Pin as Best

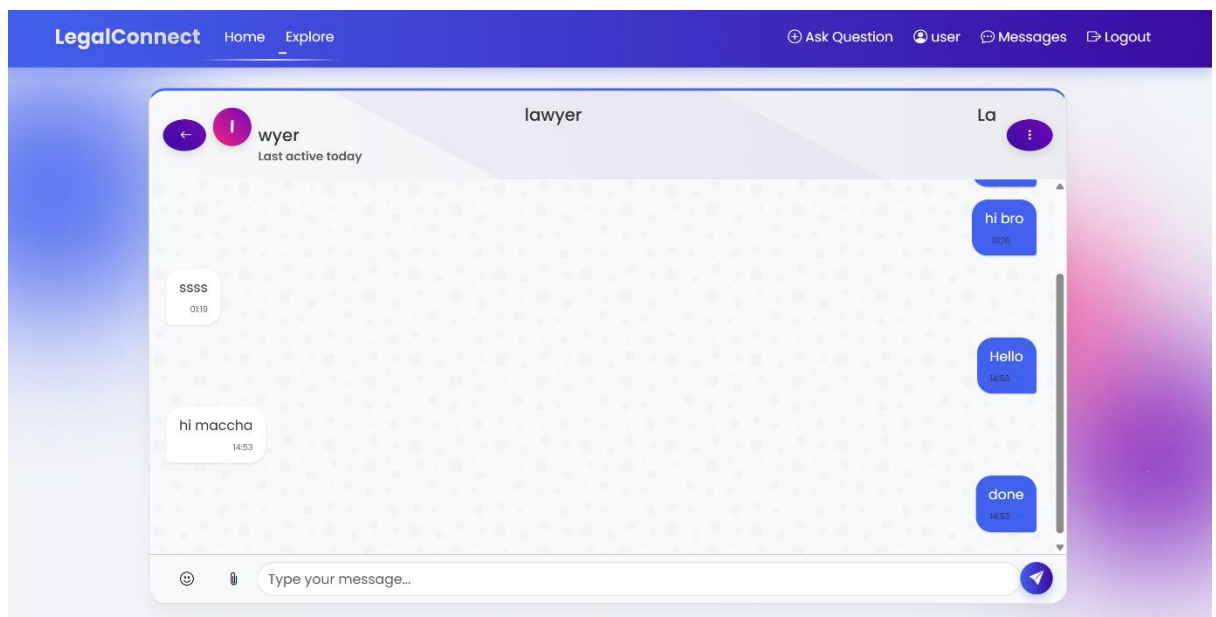
## 7. Admin Panel:

- Admin can view and verify lawyer applications.
- Manage reports and flagged content.



## 8. Direct Messaging Interface:

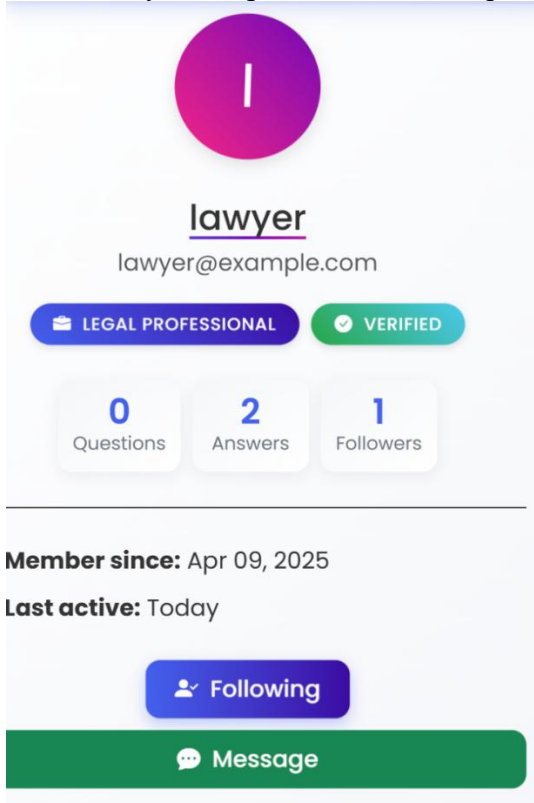
- Real-time chat between user and lawyer powered by Firebase.
- Notifications for new messages and replies.





## 9. Public Lawyer Profile:

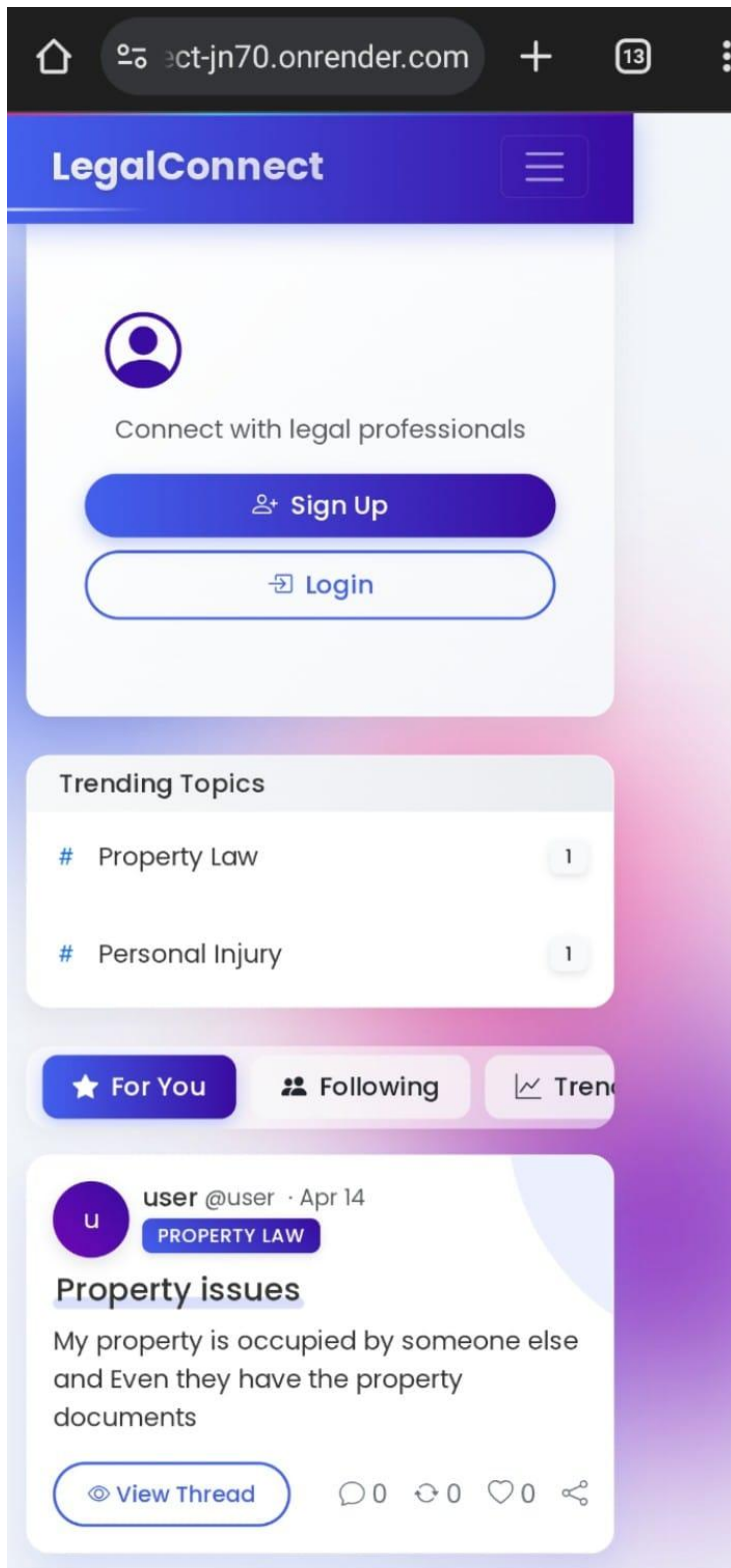
- Shows lawyer's expertise, answered questions, and reviews.



## 10. Mobile View (Responsive Screens):

- UI automatically adjusts to mobile and tablet screens using Bootstrap.

(Visual screenshots to be added to the final print version or appendix of the report.)



## **CHAPTER 10**

### **CONCLUSION**

Legal Connect successfully demonstrates the use of modern web technologies and AI integration to solve real-world legal accessibility issues. By providing a unified, interactive, and intelligent legal Q&A platform, it enables users to receive verified legal advice, participate in meaningful discussions, and benefit from AI-driven insights.

The modular structure of the platform ensures that the system is scalable and easily maintainable. With secure role-based access and real-time capabilities, Legal Connect stands as a robust and innovative legal tech solution.

Through this project, we gained practical exposure to:

- Developing a full-stack web application using Flask and Firebase
- Working with real-time databases and role-based authentication
- Integrating AI for legal knowledge representation using Gemini API
- Implementing secure and responsive user interfaces

The project lays the foundation for future enhancements like legal document review, case-law recommendation, and premium consultations. Legal Connect has the potential to serve as a scalable model for modern legal service platforms.

## CHAPTER 11

### REFERENCES

1. Firebase Documentation – <https://firebase.google.com/docs>
2. Flask Documentation – <https://flask.palletsprojects.com/>
3. Gemini AI API – <https://ai.google.dev/>
4. Bootstrap Framework – <https://getbootstrap.com/>
5. Role-Based Access Control Models – ACM Computing Surveys, 2022
6. Legal Informatics: AI and Law – Stanford Encyclopedia of Philosophy
7. Real-Time Web Apps with Firebase – Medium & Firebase Blog Articles

