LEGAL AID ADVISOR

21CSC205P Database Management Systems
MINI PROJECT REPORT

Submitted by
AASTHA HOTWANI [RA2311026010389]
APURVA SINGH[RA2311026010376]

Under the Guidance of
Beaulah Jeyavathana R
Associate professor, Computational Intelligence

in partial fulfillment of the requirementsfor the degree

of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING
with specialization in ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE

SRM INSTITUTE OF SCIENCE ANDTECHNOLOGY

KATTANKULATHUR- 603 203

MAY 2025

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that Project report titled "**LEGAL AID ADVISOR**" is the bonafide work of "**AASTHA HOTWANI[RA2311026010389], APURVA SINGH[RA2311026010376]"** who carried out the **21CSC205P Database Management Systems** mini project work under my supervision.

Dr.Beaulah Jeyavathana R
COURSE FACULTY
Associate Professor
Department of
Computational Intelligence

Dr. R. ANNIE UTHRA
PROFESSOR & HEAD
Department of
Computational Intelligence

ABSTRACT

The *Legal Aid Advisor* is an intelligent, database-driven application designed to offer legal assistance to individuals who seek quick, reliable, and affordable legal support. This project aims to automate and simplify the process of providing legal guidance by integrating a robust Database Management System (DBMS) that handles and maintains user data, case records, legal documents, expert consultations, and interaction history. The platform serves as a virtual legal helpdesk where users can register legal issues, consult verified experts, and access a legal knowledge base without the need to physically visit legal offices.

The system is designed with a strong focus on usability, scalability, and data security. It enables seamless interaction between clients and legal professionals through features like case filing, expert matching, and real-time query resolution. By incorporating structured database models, role-based access control, and user-friendly interfaces, the platform ensures efficient data handling, fast information retrieval, and secure communication.

Furthermore, the Legal Aid Advisor is especially geared toward underserved and rural populations, aiming to democratize legal access and reduce dependency on expensive legal intermediaries. The application promotes legal awareness and empowers users with self-service options like legal templates, FAQs, and search-enabled legal articles. Its modular architecture also allows for future integration with third-party services such as government legal portals, online court databases, and e-filing systems.

In essence, this project not only serves as a legal advisory tool but also contributes to building a transparent, inclusive, and digitally empowered legal ecosystem by leveraging the core principles of database design, optimization, and secure data management.

Table of Contents :

LEGAL AID ADVISOR

This report presents a detailed overview of the *Legal Aid Advisor* system, outlining its purpose, core functionalities, and its role in addressing critical gaps in legal support accessibility. The project focuses on building a digital legal services platform driven by a robust relational database system to manage client consultations, legal expert interactions, document storage, and legal knowledge dissemination.

In many regions, accessing legal assistance remains a major challenge due to high consultation costs, limited availability of professionals, and fragmented legal information. The *Legal Aid Advisor* is designed to overcome these issues by offering a centralized, secure, and user-friendly platform that streamlines communication between clients and legal experts while ensuring efficient data management through a structured DBMS.

The system provides features like user and case management, expert consultation, a searchable legal knowledge base, discussion forums, and an administrative dashboard. These modules work together to improve legal awareness, reduce delays, and offer cost-effective support—particularly for underserved communities. This report covers the architecture, database design, and the various modules developed as part of the system.

Problem Statement:

The *Legal Aid Advisor* project addresses the pressing challenges of inefficient, costly, and inaccessible legal services by implementing a database-oriented digital platform. Traditional legal systems rely heavily on manual processes, scattered information, and physical interactions, which hinder access to timely legal support.

The system targets the following key issues:

- Lack of a centralized platform for accessing legal advice.
- Inability to quickly connect with verified legal professionals based on specialization or availability.
- Fragmentation of legal resources, causing difficulties in retrieving accurate legal information.

- Manual documentation processes leading to slow case handling and data loss.
- Absence of tools for tracking case history, expert interactions, or consultation progress.
- Limited or no free legal advisory services for low-income or remote users.
- Lack of automation in document creation, appointment scheduling, and communications.
- Poor public awareness of legal rights due to unorganized content.
- Security and privacy risks when sharing sensitive data through informal channels.

The solution involves the design and deployment of a comprehensive digital system that automates case handling, enables expert-client collaboration, stores legal content in a structured format, and provides secure, role-based access—all under a unified DBMS framework.

ENTITY – RELATIONSHIP MODE

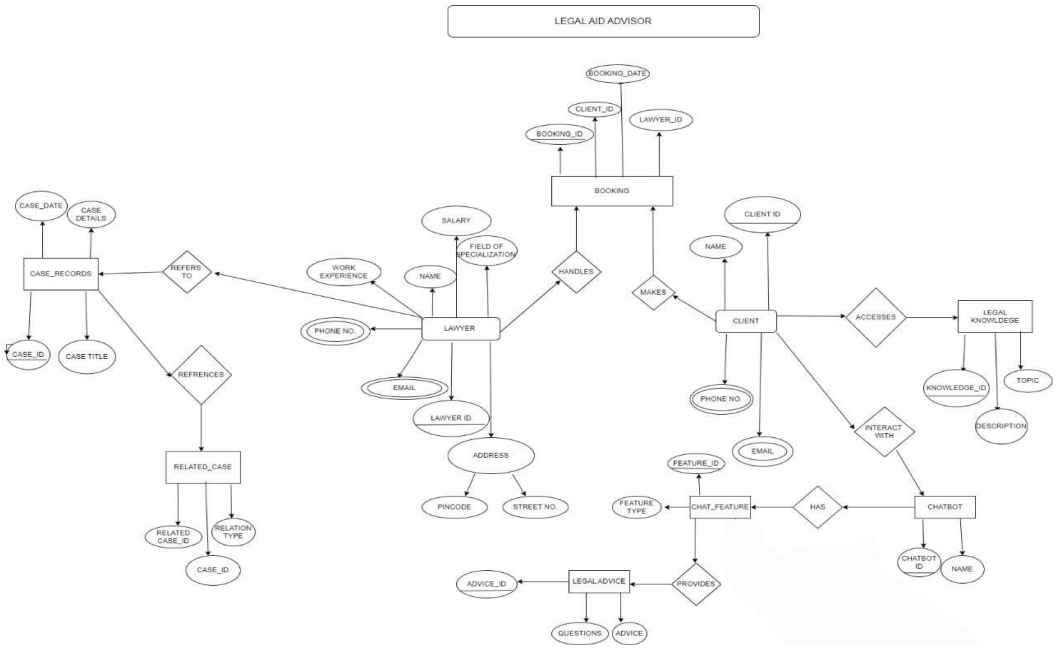

Figure 2.1 : ER Diagram of Legal Aid Advisor

DATABASE SCHEMA(Relational Model)

Database Schema:

1. Users (id, first_name, last_name, email, phone, legal_issue_category, password, updated_at)
2. Lawyer (id, name, rating, phone, address)
3. Booking (id, first_name, last_name, email, phone, address, lawyer_name, specialization, case_type, case_description, consultation_date, urgency, consultation_mode, status, created_at)
4. Chatbot_New (chatbot_id, name, active)
5. Feedback (id, user_id, message, created_at)
6. Civil_Cases_View (consultation_id, user_id, consultation_date, case_description)
7. State_Courts (id, state_name, court_names)
8. Consultations (consultation_id, user_id, consultation_date, case_description)
9. User_Issues (id, first_name, legal_issue_category)

Relationship Tables:

10. Books (user_id, booking_id)
11. Assigned_To (lawyer_id, booking_id)
12. Submits (user_id, feedback_id)
13. Requests (user_id, consultation_id)

Implementation Considerations:

- Primary Keys (PK) uniquely identify each record in a table.
- Foreign Keys (FK) define relationships between entities and maintain referential integrity.
- Normalization ensures atomic attributes, eliminates redundancy, and maintains consistency.
- Indexing can be applied to frequently queried fields (e.g., user_id, consultation_date) to improve performance.

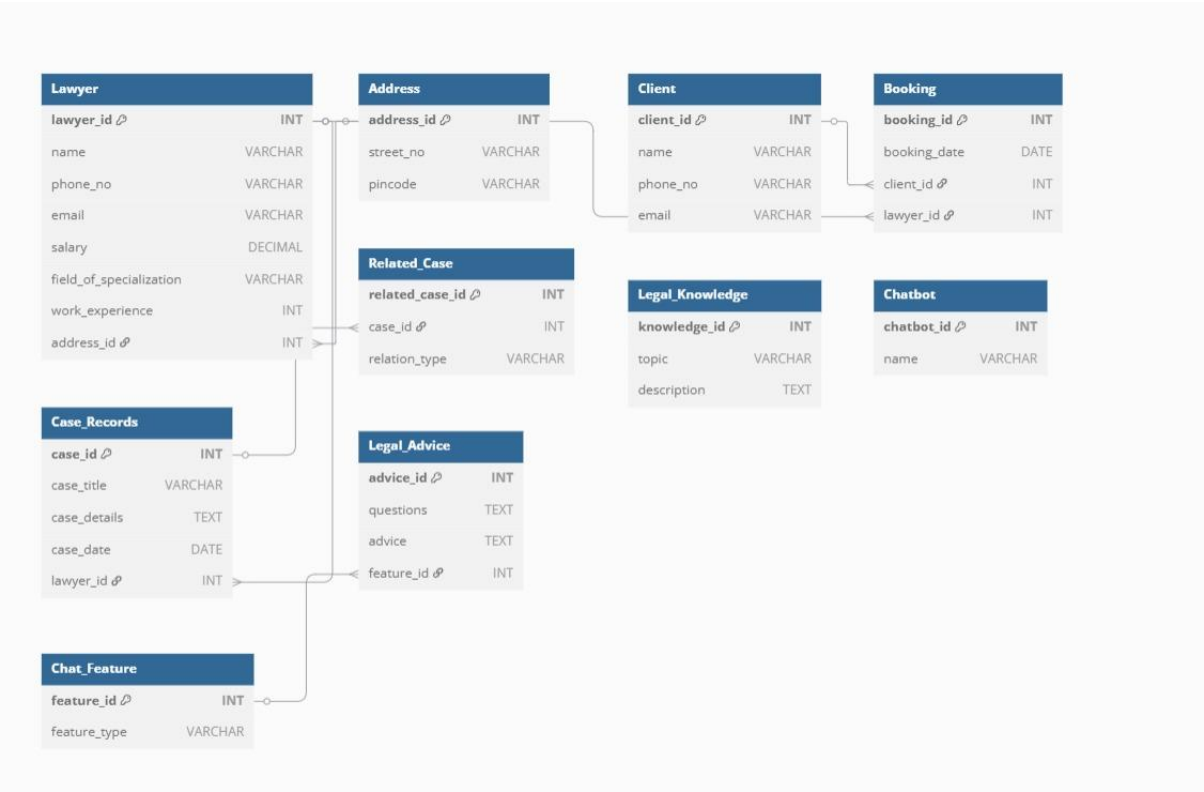

Figure 3.1 : Relational Database Schema

ENTITY AND ATTRIBUTES

1.Users Table:

- id (INT, Primary Key)
- first_name (VARCHAR)
- last_name (VARCHAR)
- email (VARCHAR)
- phone (VARCHAR)
- legal_issue_category (VARCHAR)
- password (VARCHAR)
- updated_at (DATETIME)

2.Lawyer Table:

- id (INT, Primary Key)
- name (VARCHAR)
- rating (INT)
- phone (VARCHAR)
- address (VARCHAR)

3.Booking Table:

- id (INT, Primary Key)
- first_name (VARCHAR)
- last_name (VARCHAR)
- email (VARCHAR)
- phone (VARCHAR)
- address (VARCHAR)
- lawyer_name (VARCHAR)
- specialization (VARCHAR)
- case_type (VARCHAR)
- case_description (VARCHAR)
- consultation_date (DATE)
- urgency (VARCHAR)
- consultation_mode (VARCHAR)
- status (VARCHAR)
- created_at (DATETIME)

4. Chatbot_New Table:

- chatbot_id (INT, Primary Key)
- name (VARCHAR)

- active (INT)

5.Feedback Table:

- id (INT, Primary Key)
- user_id (INT)
- message (TEXT)
- created_at (DATETIME)

6.Civil_Cases_View Table:

- consultation_id (INT, Primary Key)
- user_id (INT)
- consultation_date (DATE)
- case_description (VARCHAR)

7.State_Courts Table:

- id (INT, Primary Key)
- state_name (VARCHAR)
- court_names (VARCHAR)

8.Consultations Table:

- consultation_id (INT, Primary Key)
- user_id (INT)
- consultation_date (DATE)
- case_description (VARCHAR**)**

9.User_Issues Table:

- id (INT, Primary Key)
- first_name (VARCHAR)
- legal_issue_category (VARCHAR)

DATABASE CREATION

1. User Table

```
CREATE TABLE users (
   Client_ID INT PRIMARY KEY,
   Name VARCHAR(100),
   Phone_No VARCHAR(15),
   Email VARCHAR(100)
);

-- SELECT
SELECT * FROM users;
```

```
mysql> select * from users;
+----+------------+-----------+--------------------------+------------+---------------------+------------------------
-----------------------------------------------------------------+--------------------+
| id | first_name | last_name | email                    | phone      | legal_issue_category | password
                                                                 | updated_at         |
+----+------------+-----------+--------------------------+------------+---------------------+------------------------
-----------------------------------------------------------------+--------------------+
|  1 | ABC        | def       | legalaid@gmail.com       | 987654321  | Family Law           | 12345678
                                                                 | 2025-04-08 03:02:54 |
|  2 | aman       | singh     | amansingh@gmail.com      | 1234567891 | Criminal Defense     | 12345678
                                                                 | 2025-04-08 03:01:57 |
|  4 | Aayush     | Dhingra   | aayush123@gmail.com      | 7987484151 | Family Law           | pbkdf2_sha256$870000$zoY
lRDwMviGuUnrfNVaXBf$Ng9MP5Y1s2Wve0LZMPer8ckk0E4nw+/4EzZNp2Jv0A0= | 2025-04-08 03:01:57 |
|  5 | Nitya      | Gupta     | nityagupta@gmail.com     | 123456789  | Immigration          | pbkdf2_sha256$870000$5zR
ckAZu4QyVzYRPFC2uJV$gj9I7bePYHXHnV5pfJC5ZFpCAGdU1iX+OGvURPMuPTY= | 2025-04-08 03:01:57 |
|  6 | Hi         | Singh     | hi@gmail.com             | 123456789  | Family Law           | pbkdf2_sha256$870000$6Tp
AFhhl2nKcDLH3oqBnvv$FLGsZAVEsNuuvUrVgAOryB3Ks26MfNPLQI7WmuwIomc= | 2025-04-08 03:01:57 |
|  7 | Anushka    | Gupta     | er.anushkagupta@gmail.com | 123456789  | Family Law           | pbkdf2_sha256$870000$1j0
Nx4MEx9qu7xRq8FPWiZ$Qz+6HqM7GHwvjWInHByuIvY4sEdop1Onh8oVSDAyPKo= | 2025-04-08 03:01:57 |
|  8 | aditya     | nair      | abc@gmail.com            | 1234567891 | Family Law           | pbkdf2_sha256$870000$Qmz
Vb0jWuQ3xkCU1fwvDhF$IY9mP1YA//MAoMM1Fh10ySXioR8NQIvdxsRHvsdMmxw= | 2025-04-08 03:01:57 |
|  9 | Avinav     | P         | avinavp@gmail.com        | 1234567890 | Family Law           | pbkdf2_sha256$870000$laa
cUqdOEOO88GmptqoZfk$OO8ym5y04xmHfBdvtSNauAY5gUKRPlHRhQmIZTXfSX8= | 2025-04-08 08:34:21 |
| 10 | teacher    | t         | teachert@gmail.com       | 1234567890 | Family Law           | pbkdf2_sha256$870000$lST
fGPn7eSyVUSVJAevOB9$wyLUjevPGMVWUwW+7YZIFOo95mS0leQ45tWJBUr5bSQ= | 2025-04-08 08:40:11 |
+----+------------+-----------+--------------------------+------------+---------------------+------------------------
```

Figure 5.1 : user Table

2.Lawyer table

```
CREATE TABLE Lawyer (
   Lawyer_ID INT PRIMARY KEY,
   Name VARCHAR(100),
   Phone_No VARCHAR(15),
   Email VARCHAR(100),
   Salary DECIMAL(10, 2),
   Work_Experience INT,
   Field_of_Specialization VARCHAR(100)
);

-- SELECT
SELECT* FROM lawerys;
```

```
mysql> select * from  lawyer;
+----+----------------+--------+----------------+---------------------------------+
| id | name           | rating | phone          | address                         |
+----+----------------+--------+----------------+---------------------------------+
|  1 | Alice Johnson  |      4 | (111) 222-3333 | 101 First St, Springfield, IL   |
|  2 | Bob Smith      |      5 | (222) 333-4444 | 202 Second St, Lincoln, NE      |
|  3 | Carol Williams |      3 | (333) 444-5555 | 303 Third St, Austin, TX        |
|  4 | David Brown    |      4 | (444) 555-6666 | 404 Fourth St, Miami, FL        |
|  5 | Eva Davis      |      5 | (555) 666-7777 | 505 Fifth St, Seattle, WA       |
|  6 | Frank Miller   |      3 | (666) 777-8888 | 606 Sixth St, Denver, CO        |
|  7 | Grace Wilson   |      4 | (777) 888-9999 | 707 Seventh St, Portland, OR    |
|  8 | Henry Moore    |      4 | (888) 999-0000 | 808 Eighth St, Chicago, IL      |
|  9 | Ivy Taylor     |      4 | (999) 000-1111 | 909 Ninth St, San Francisco, CA |
| 10 | Jack Anderson  |      5 | (000) 111-2222 | 1000 Tenth St, Las Vegas, NV    |
| 11 | Kathy Lee      |      5 | (123) 456-7890 | 1111 Eleventh St, Phoenix, AZ   |
| 12 | Liam Johnson   |      4 | (234) 567-8901 | 1212 Twelfth St, Atlanta, GA    |
| 13 | Mia Brown      |      3 | (345) 678-9012 | 1313 Thirteenth St, Dallas, TX  |
| 14 | Noah Smith     |      5 | (456) 789-0123 | 1414 Fourteenth St, Boston, MA  |
+----+----------------+--------+----------------+---------------------------------+
14 rows in set (0.03 sec)
```

Figure 5.2 : Lawyer Table

3. BOOKING Table

```
CREATE TABLE booking (
   Booking_ID INT PRIMARY KEY,
   Booking_Date DATE,
   Client_ID INT,
   Lawyer_ID INT,
   FOREIGN KEY (Client_ID) REFERENCES Client(Client_ID),
   FOREIGN KEY (Lawyer_ID) REFERENCES Lawyer(Lawyer_ID)
);

-- SELECT
SELECT * FROM booking;
```

```
mysql> select * from  booking ;
+----+------------+-----------+--------------------------+------------+-----------------------------------------+-----
----------+----------------+-----------+-----------------------------------+-------------------+-----------+---------
----------+---------+---------------------+
| id | first_name | last_name | email                    | phone      | address                                 | lawy
er_name   | specialization | case_type | case_description                  | consultation_date | urgency   | consulta
tion_mode | status  | created_at          |
+----+------------+-----------+--------------------------+------------+-----------------------------------------+-----
----------+----------------+-----------+-----------------------------------+-------------------+-----------+---------
----------+---------+---------------------+
|  1 | ABC        | Hotwani   | legalaid@gmail.com       | 1234567891 | fv revogrgt                             | John
 Smith    | Civil Law      | Criminal  | ueswvt 4t                         | 2025-03-13        | Emergency | Online
          | Pending | 2025-03-06 11:04:36 |
|  2 | ABC        | def       | legalaid@gmail.com       | 1234567891 | asdftgyjkl                              | John
 Smith    | Civil Law      | Criminal  | asdxfcghjkllmnbv                  | 2025-03-13        | Normal    | In-perso
n         | Pending | 2025-03-06 11:07:33 |
|  3 | ABC        | def       | legalaid@gmail.com       | 1234567891 | asdftgyjkl                              | John
 Smith    | Civil Law      | Criminal  | asdxfcghjkllmnbv                  | 2025-03-13        | Normal    | In-perso
n         | Pending | 2025-03-06 11:07:39 |
|  4 | rahul      | patel     | rahulpatel@gmail.com     | 1234567891 | q3wesdrtfgyhujmkl                       | John
 Smith    | Criminal Law   | Family    | zasxdcfvgbhnmk,l                  | 2025-03-28        | Normal    | Online
          | Pending | 2025-03-06 19:30:24 |
|  5 | a          | b         | legalaid@gmail.com       | 1234567891 | azesxrvgbhnjmkl,                        | Henr
y Moore   | Civil Law      | Criminal  | swedrftghjk,l.                    | 2025-03-04        | Normal    | In-perso
```

Figure 5.2 : booking Table

4. CHATBOT Table

```
CREATE TABLE  chatbot_new(
   Chatbot_ID INT PRIMARY KEY,
   Name VARCHAR(100)
```

);

-- SELECT
SELECT* FROM chatbot_new;

```
mysql> select * from   chatbot_new ;
+------------+----------------+--------+
| CHATBOT_ID | NAME           | active |
+------------+----------------+--------+
|      24435 | PDF_Summarizer |      1 |
|      78695 | LEGAL_BOT      |      1 |
|      99999 | LAW_HELPER     |      1 |
+------------+----------------+--------+
3 rows in set (0.00 sec)
```

Figure 5.4 : chatbot_new Table

5. feedback Table

CREATE TABLE feedback (
  id INT NOT NULL PRIMARY KEY,
  user_id INT,
  message TEXT,
  created_at DATETIME,
  FOREIGN KEY (user_id) REFERENCES users(id)
);

SELECT
SELECT*FROM feedback;

```
mysql> select * from   feedback ;
+------+---------+----------------------+---------------------+
| id   | user_id | message              | created_at          |
+------+---------+----------------------+---------------------+
|    1 |       2 | Great support!       | 2025-04-07 23:47:58 |
| 1009 |       9 | Welcome to Legal Aid | 2025-04-08 08:34:21 |
| 1010 |      10 | Welcome to Legal Aid | 2025-04-08 08:40:11 |
+------+---------+----------------------+---------------------+
3 rows in set (0.01 sec)
```

Figure 5.5 : feedback Table

6 .state_courts Table

CREATE TABLE state_courts (
  id INT PRIMARY KEY,
  state_name VARCHAR(100),
  court_names TEXT
);

INSERT INTO state_courts (id, state_name, court_names) VALUES (1, 'Andhra Pradesh', 'Andhra Pradesh High Court, Visakhapatnam District Court, Vijayawada Civil Court');

INSERT INTO state_courts (id, state_name, court_names) VALUES (2, 'Arunachal Pradesh', 'Itanagar District Court, Yupia Sessions Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (3, 'Assam', 'Gauhati High Court, Kamrup Metropolitan District Court, Dibrugarh District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (4, 'Bihar', 'Patna High Court, Gaya Civil Court, Muzaffarpur District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (5, 'Chhattisgarh', 'Chhattisgarh High Court, Bilaspur District Court, Raipur Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (6, 'Goa', 'Bombay High Court – Panaji Bench, North Goa District Court, South Goa District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (7, 'Gujarat', 'Gujarat High Court, Ahmedabad City Civil Court, Surat District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (8, 'Haryana', 'Punjab and Haryana High Court, Faridabad District Court, Gurugram Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (9, 'Himachal Pradesh', 'Himachal Pradesh High Court, Shimla District Court, Mandi Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (10, 'Jharkhand', 'Jharkhand High Court, Ranchi Civil Court, Dhanbad District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (11, 'Karnataka', 'Karnataka High Court, Bengaluru City Civil Court, Dharwad Bench');
INSERT INTO state_courts (id, state_name, court_names) VALUES (12, 'Kerala', 'Kerala High Court, Ernakulam District Court, Kozhikode Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (13, 'Madhya Pradesh', 'Madhya Pradesh High Court, Jabalpur Bench, Indore District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (14, 'Maharashtra', 'Bombay High Court, Pune District Court, Nagpur Bench, Thane Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (15, 'Manipur', 'Manipur High Court, Imphal District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (16, 'Meghalaya', 'Meghalaya High Court, Shillong District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (17, 'Mizoram', 'Mizoram District Court, Aizawl Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (18, 'Nagaland', 'Kohima District Court, Dimapur Sessions Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (19, 'Odisha', 'Orissa High Court, Cuttack District Court, Bhubaneswar Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (20, 'Punjab', 'Punjab and Haryana High Court, Ludhiana District Court, Amritsar Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (21, 'Rajasthan', 'Rajasthan High Court, Jaipur District Court, Jodhpur Bench');
INSERT INTO state_courts (id, state_name, court_names) VALUES (22, 'Sikkim', 'Sikkim High Court, Gangtok District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (23, 'Tamil Nadu', 'Madras High Court, Coimbatore District Court, Madurai Bench, Salem Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (24, 'Telangana', 'Telangana High Court, Hyderabad City Civil Court, Warangal District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (25, 'Tripura', 'Tripura High Court, Agartala District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (26, 'Uttar Pradesh', 'Allahabad High Court, Lucknow Bench, Agra District Court, Varanasi Civil Court');

INSERT INTO state_courts (id, state_name, court_names) VALUES (27, 'Uttarakhand', 'Uttarakhand High Court, Dehradun District Court, Nainital Civil Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (28, 'West Bengal', 'Calcutta High Court, Alipore Judges Court, Barasat District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (29, 'Andaman and Nicobar Islands', 'Port Blair District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (30, 'Chandigarh', 'District and Sessions Court Chandigarh');
INSERT INTO state_courts (id, state_name, court_names) VALUES (31, 'Dadra and Nagar Haveli and Daman and Diu', 'Daman District Court, Silvassa District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (32, 'Delhi', 'Delhi High Court, Tis Hazari Court, Saket District Court, Patiala House Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (33, 'Jammu and Kashmir', 'High Court of J&K and Ladakh, Srinagar District Court, Jammu District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (34, 'Ladakh', 'Leh District Court, Kargil District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (35, 'Lakshadweep', 'Kavaratti District Court');
INSERT INTO state_courts (id, state_name, court_names) VALUES (36, 'Puducherry', 'Puducherry Principal District Court');

-- SELECT
SELECT * FROM state_courts;

```
mysql> select * from   state_courts;
+----+----------------------------------------+-----------------------------------------------------------------------------------------+
| id | state_name                             | court_names                                                                             |
+----+----------------------------------------+-----------------------------------------------------------------------------------------+
|  1 | Andhra Pradesh                         | Andhra Pradesh High Court, Visakhapatnam District Court, Vijayawada Civil Court        |
|  2 | Arunachal Pradesh                      | Itanagar District Court, Yupia Sessions Court                                           |
|  3 | Assam                                  | Gauhati High Court, Kamrup Metropolitan District Court, Dibrugarh District Court        |
|  4 | Bihar                                  | Patna High Court, Gaya Civil Court, Muzaffarpur District Court                          |
|  5 | Chhattisgarh                           | Chhattisgarh High Court, Bilaspur District Court, Raipur Civil Court                    |
|  6 | Goa                                    | Bombay High Court - Panaji Bench, North Goa District Court, South Goa District Court    |
|  7 | Gujarat                                | Gujarat High Court, Ahmedabad City Civil Court, Surat District Court                    |
|  8 | Haryana                                | Punjab and Haryana High Court, Faridabad District Court, Gurugram Civil Court          |
|  9 | Himachal Pradesh                       | Himachal Pradesh High Court, Shimla District Court, Mandi Civil Court                  |
| 10 | Jharkhand                              | Jharkhand High Court, Ranchi Civil Court, Dhanbad District Court                        |
| 11 | Karnataka                              | Karnataka High Court, Bengaluru City Civil Court, Dharwad Bench                         |
| 12 | Kerala                                 | Kerala High Court, Ernakulam District Court, Kozhikode Civil Court                      |
| 13 | Madhya Pradesh                         | Madhya Pradesh High Court, Jabalpur Bench, Indore District Court                        |
| 14 | Maharashtra                            | Bombay High Court, Pune District Court, Nagpur Bench, Thane Civil Court                 |
| 15 | Manipur                                | Manipur High Court, Imphal District Court                                               |
| 16 | Meghalaya                              | Meghalaya High Court, Shillong District Court                                           |
| 17 | Mizoram                                | Mizoram District Court, Aizawl Civil Court                                              |
| 18 | Nagaland                               | Kohima District Court, Dimapur Sessions Court                                           |
| 19 | Odisha                                 | Orissa High Court, Cuttack District Court, Bhubaneswar Civil Court                      |
| 20 | Punjab                                 | Punjab and Haryana High Court, Ludhiana District Court, Amritsar Civil Court            |
| 21 | Rajasthan                              | Rajasthan High Court, Jaipur District Court, Jodhpur Bench                              |
| 22 | Sikkim                                 | Sikkim High Court, Gangtok District Court                                               |
| 23 | Tamil Nadu                             | Madras High Court, Coimbatore District Court, Madurai Bench, Salem Civil Court          |
| 24 | Telangana                              | Telangana High Court, Hyderabad City Civil Court, Warangal District Court               |
| 25 | Tripura                                | Tripura High Court, Agartala District Court                                             |
| 26 | Uttar Pradesh                          | Allahabad High Court, Lucknow Bench, Agra District Court, Varanasi Civil Court          |
| 27 | Uttarakhand                            | Uttarakhand High Court, Dehradun District Court, Nainital Civil Court                   |
| 28 | West Bengal                            | Calcutta High Court, Alipore Judges Court, Barasat District Court                       |
| 29 | Andaman and Nicobar Islands            | Port Blair District Court                                                               |
| 30 | Chandigarh                             | District and Sessions Court Chandigarh                                                  |
| 31 | Dadra and Nagar Haveli and Daman and Diu | Daman District Court, Silvassa District Court                                           |
| 32 | Delhi                                  | Delhi High Court, Tis Hazari Court, Saket District Court, Patiala House Court           |
| 33 | Jammu and Kashmir                      | High Court of J&K and Ladakh, Srinagar District Court, Jammu District Court             |
| 34 | Ladakh                                 | Leh District Court, Kargil District Court                                               |
| 35 | Lakshadweep                            | Kavaratti District Court                                                                |
| 36 | Puducherry                             | Puducherry Principal District Court                                                     |
+----+----------------------------------------+-----------------------------------------------------------------------------------------+
```

Figure 5.6 : state_courts Table

7. Consultations Table

CREATE TABLE consultations (

```
    consultation_id INT PRIMARY KEY,
    user_id INT,
    consultation_date DATE,
    case_description VARCHAR(255)
);

-- SELECT
SELECT * FROM consultations;
```

```
mysql> select * from  consultations;
+-----------------+---------+-------------------+------------------+
| consultation_id | user_id | consultation_date | case_description |
+-----------------+---------+-------------------+------------------+
|               1 |       1 | 2025-03-13        | robery case      |
|               2 |       2 | 2025-03-28        | murder case      |
|               3 |       4 | 2025-03-18        | divorce case     |
|               4 |       5 | 2025-03-18        | assault case     |
|               5 |       6 | 2025-03-25        | petty theft      |
|               6 |       7 | 2025-03-24        | offenses         |
|               7 |       8 | 2025-03-12        | land dispute     |
+-----------------+---------+-------------------+------------------+
7 rows in set (0.00 sec)
```

Figure 5.7 : consultation Table

8. user_issues Table

```
CREATE TABLE user_issues (
    id INT PRIMARY KEY,
    first_name VARCHAR(100),
    legal_issue_category VARCHAR(100)
);

--SELECT
SELECT * FROM  user_issues;
```

```
mysql> select * from   user_issues;
+----+------------+----------------------+
| id | first_name | legal_issue_category |
+----+------------+----------------------+
|  1 | ABC        | Family Law           |
|  2 | aman       | Criminal Defense     |
|  4 | Aayush     | Family Law           |
|  5 | Nitya      | Immigration          |
|  6 | Hi         | Family Law           |
|  7 | Anushka    | Family Law           |
|  8 | aditya     | Family Law           |
|  9 | Avinav     | Family Law           |
| 10 | teacher    | Family Law           |
+----+------------+----------------------+
9 rows in set (0.00 sec)
```

Figure 5.8 : user_issues Table

TRIGGERS

1.DELIMITER //

```
CREATE TRIGGER update_timestamp
AFTER INSERT ON users
FOR EACH ROW
BEGIN
   UPDATE Product
   SET stock_quantity = stock_quantity - NEW.quantity
   WHERE product_id = NEW.product_id;
END //

DELIMITER ;
```

```
mysql> CREATE TRIGGER update_timestamp
    ->     BEFORE UPDATE ON users
    ->     FOR EACH ROW
    ->     SET NEW.password = NEW.password;
Query OK, 0 rows affected (0.01 sec)
```

Figure 6.1 : update_timestamp Trigger

2. DELIMITER //

```
CREATE TRIGGER log_delete_consult
AFTER DELETE ON consultations
FOR EACH ROW
BEGIN
   INSERT INTO feedback (id, user_id, message, created_at)
   VALUES (999, OLD.user_id, 'Consultation deleted', NOW());
END //

DELIMITER ;
```

```
mysql> CREATE TRIGGER log_delete_consult
    ->     AFTER DELETE ON consultations
    ->     FOR EACH ROW
    ->     INSERT INTO feedback (id, user_id, message, created_at) VALUES (999, OLD.user_id, 'Consultation deleted', NOW());
Query OK, 0 rows affected (0.01 sec)
```

Figure 6.2 log_delete_consult Trigger

3.DELIMITER //

```
CREATE TRIGGER after_user_insert
AFTER INSERT ON users
FOR EACH ROW
BEGIN
   INSERT INTO feedback (id, user_id, message, created_at)
   VALUES (NEW.id + 1000, NEW.id, 'Welcome to Legal Aid', NOW());
END //

DELIMITER ;
```

```
mysql> CREATE TRIGGER after_user_insert
    ->     AFTER INSERT ON users
    ->     FOR EACH ROW
    ->     INSERT INTO feedback (id, user_id, message, created_at) VALUES (NEW.id + 1000, NEW.id, 'Welcome to Legal Aid', NOW());
Query OK, 0 rows affected (0.01 sec)
```

Figure 6.3 : after_user_insert Trigger

VIEWS

1.
CREATE VIEW upcoming_consultations AS
SELECT *
FROM consultations
WHERE consultation_date > CURDATE();

```
mysql> CREATE VIEW upcoming_consultations AS SELECT * FROM consultations WHERE consultation_date > CURDATE();
Query OK, 0 rows affected (0.02 sec)
```

Figure 7.1 : upcoming_consultation View

2.
CREATE VIEW user_issues AS
SELECT id, first_name, legal_issue_category
FROM users;

```
mysql> CREATE VIEW user_issues AS SELECT id, first_name, legal_issue_category FROM users;
Query OK, 0 rows affected (0.01 sec)
```

Figure 7.2 : user_issues View

JOINS

1.
SELECT u.name AS UserName, l.name AS LawyerName, b.booking_date
FROM Booking b
INNER JOIN User u ON b.user_id = u.user_id
INNER JOIN Lawyer l ON b.lawyer_id = l.lawyer_id;

```
mysql> SELECT u.name AS UserName, l.name AS LawyerName, b.booking_date
    -> FROM Booking b
    -> INNER JOIN User u ON b.user_id = u.user_id
    -> INNER JOIN Lawyer l ON b.lawyer_id = l.lawyer_id;
+-----------+-------------+--------------+
| UserName  | LawyerName  | booking_date |
+-----------+-------------+--------------+
| John Doe  | Alice Brown | 2024-12-01   |
| Jane Roe  | Bob White   | 2024-12-03   |
+-----------+-------------+--------------+
2 rows in set (0.00 sec)
```

Figure 8.1 : First Join

2.

```
SELECT u.name AS UserName, b.booking_date
FROM User u
LEFT JOIN Booking b ON u.user_id = b.user_id

UNION

SELECT u.name AS UserName, b.booking_date
FROM Booking b
RIGHT JOIN User u ON u.user_id = b.user_id;
```

```
mysql> SELECT u.name AS UserName, b.booking_date
    -> FROM User u
    -> LEFT JOIN Booking b ON u.user_id = b.user_id
    -> UNION
    -> SELECT u.name AS UserName, b.booking_date
    -> FROM Booking b
    -> RIGHT JOIN User u ON u.user_id = b.user_id;
+-----------+--------------+
| UserName  | booking_date |
+-----------+--------------+
| John Doe  | 2024-12-01   |
| Jane Roe  | NULL         |
+-----------+--------------+
2 rows in set (0.00 sec)
```

Figure 8.2 : Second Join

3.

```
SELECT cr.case_title, l.name AS LawyerName
FROM Case_Records cr
RIGHT JOIN Lawyer l ON cr.lawyer_id = l.lawyer_id;
```

```
mysql> SELECT cr.case_title, l.name AS LawyerName
    -> FROM Case_Records cr
    -> RIGHT JOIN Lawyer l ON cr.lawyer_id = l.lawyer_id;
+--------------------+-------------+
| case_title         | LawyerName  |
+--------------------+-------------+
| Property Dispute   | Alice Brown |
| NULL               | Bob White   |
+--------------------+-------------+
2 rows in set (0.00 sec)
```

Figure 8.3 : Third Join

SETS

1.
SELECT name FROM User
UNION
SELECT name FROM Lawyer;

```
mysql> SELECT name FROM User
    -> UNION
    -> SELECT name FROM Lawyer;
+------------------+
| name             |
+------------------+
| John Doe         |
| Jane Smith       |
| Alice Brown      |
| Bob White        |
+------------------+
4 rows in set (0.00 sec)
```

Figure 9.1 : First Set

2.
SELECT u.name
FROM User u
INNER JOIN Lawyer l ON u.name = l.name;

```
mysql> SELECT u.name
    -> FROM User u
    -> INNER JOIN Lawyer l ON u.name = l.name;
+------------------+
| name             |
+------------------+
| Alice Brown      |
+------------------+
1 row in set (0.00 sec)
```

Figure 9.2 : Second Join

CURSORS

1.

```
DELIMITER //

CREATE PROCEDURE PrintUserNames()
BEGIN
   DECLARE done INT DEFAULT 0;
   DECLARE uname VARCHAR(100);
   DECLARE user_cursor CURSOR FOR SELECT name FROM User;
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

   OPEN user_cursor;

   read_loop: LOOP
      FETCH user_cursor INTO uname;
      IF done THEN
         LEAVE read_loop;
      END IF;
      SELECT uname AS UserName;
   END LOOP;

   CLOSE user_cursor;
END;
//

DELIMITER ;
```

```
mysql> DELIMITER //

mysql> CREATE PROCEDURE PrintUserNames()
    -> BEGIN
    ->     DECLARE done INT DEFAULT 0;
    ->     DECLARE uname VARCHAR(100);
    ->     DECLARE user_cursor CURSOR FOR SELECT name FROM User;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->     OPEN user_cursor;
    ->
    ->     read_loop: LOOP
    ->         FETCH user_cursor INTO uname;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->         SELECT uname AS UserName;
    ->     END LOOP;
    ->
    ->     CLOSE user_cursor;
    -> END;
//
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;

mysql> CALL PrintUserNames();
+-----------+
| UserName  |
+-----------+
| John Doe  |
+-----------+
+------------+
| UserName   |
+------------+
| Alice Brown|
+------------+
2 rows in set (0.00 sec)
```

Figure 11.1 : First Cursor

8.

```
DELIMITER //

CREATE PROCEDURE ConcatLawyerNames()
BEGIN
   DECLARE done INT DEFAULT 0;
   DECLARE lname VARCHAR(100);
   DECLARE all_names TEXT DEFAULT '';
   DECLARE l_cursor CURSOR FOR SELECT name FROM Lawyer;
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
```

```
    OPEN l_cursor;

    read_loop: LOOP
        FETCH l_cursor INTO lname;
        IF done THEN
            LEAVE read_loop;
        END IF;
        SET all_names = CONCAT(all_names, lname, ', ');
    END LOOP;

    CLOSE l_cursor;

    SELECT TRIM(TRAILING ', ' FROM all_names) AS AllLawyerNames;
END;
//

DELIMITER ;
```

```
mysql> DELIMITER //

mysql> CREATE PROCEDURE ConcatLawyerNames()
    -> BEGIN
    ->     DECLARE done INT DEFAULT 0;
    ->     DECLARE lname VARCHAR(100);
    ->     DECLARE all_names TEXT DEFAULT '';
    ->     DECLARE l_cursor CURSOR FOR SELECT name FROM Lawyer;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->     OPEN l_cursor;
    ->
    ->     read_loop: LOOP
    ->         FETCH l_cursor INTO lname;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->         SET all_names = CONCAT(all_names, lname, ', ');
    ->     END LOOP;
    ->
    ->     CLOSE l_cursor;
    ->
    ->     SELECT TRIM(TRAILING ', ' FROM all_names) AS AllLawyerNames;
    -> END;
//
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;

mysql> CALL ConcatLawyerNames();
+----------------------------+
| AllLawyerNames             |
+----------------------------+
| Alice Brown, Bob White     |
+----------------------------+
1 row in set (0.00 sec)
```

Figure 11.2 : Second Curs

PITFALLS , Normalizations & Dependencies

Analyzing the Pitfalls:

Before normalization, the database often suffers from issues such as:

Pitfall	Description	Example
Data Redundancy	**Repetition of the same data in multiple tables**	**A lawyer's address repeated in every booking**
Update Anomalies	**Changing data in one place requires changes in multiple places**	**Updating a lawyer's phone in 5 different records**
Insertion Anomalies	**Can't add data without other dependent data**	**Cannot add a lawyer without a booking**
Deletion Anomalies	**Deleting a row causes loss of useful data**	**Deleting a booking removes the only record of a lawyer**

Identifying Dependencies:

Dependencies tell us how data is related and where redundancies exist. These are categorized into:

A. Functional Dependency

- A → B means attribute B depends on A.
- Example: lawyer_id → lawyer_name, specialization (Each lawyer ID uniquely determines the name and field)

B. Transitive Dependency

- A → B → C, then A → C is transitive.
- Example: booking_id → lawyer_id → lawyer_name (booking indirectly gives lawyer name, which can cause duplication)

C. Partial Dependency

- In composite keys, if a non-key attribute depends only on part of the key.
- Example: (client_id, lawyer_id) → booking_date, but client_name depends only on client_id.

Applying Normalization:

Normalization transforms the database into forms that reduce redundancy and dependency issues.

Un-Normalized Form:

```
CREATE TABLE Booking (

    id INT,

    first_name VARCHAR(50),

    last_name VARCHAR(50),

    email VARCHAR(100),

    phone VARCHAR(20),

    address VARCHAR(255),

    lawyer_name VARCHAR(100),

    specialization VARCHAR(100),

    case_type VARCHAR(100),

    case_description TEXT,

    consultation_date DATE,

    urgency VARCHAR(100),
```

```
    consultation_mode VARCHAR(100),
    status VARCHAR(50),
    created_at DATE
);
```

Unnormalized Table (UNF)

This table contains multivalued and composite attributes.

Table: Booking

id	first_name	last_name	email	phone	address	lawyer_name	specialization	case_type	case_description	consultation_date	urgency	consultation_mode	status	created_at
1	John	Doe	john@example.com	1234567890	123 Street, NY	Atty. Smith	Criminal Law, Family Law	Theft	Stolen bike case	2024-06-10	High, Emergency	In-person, Online	Confirmed	2024-06-01

Applying 1NF:

```
CREATE TABLE Booking_1NF (
    id INT,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100),
    phone VARCHAR(20),
    address VARCHAR(255),
    lawyer_name VARCHAR(100),
    specialization VARCHAR(100),
```

```
    case_type VARCHAR(100),
    case_description TEXT,
    consultation_date DATE,
    urgency VARCHAR(50),
    consultation_mode VARCHAR(50),
    status VARCHAR(50),
    created_at DATE
);
```

First Normal Form (1NF)

Multivalued attributes are split into individual rows to ensure atomicity.

id	first_name	last_name	email	phone	address	lawyer_name	specialization	case_type	case_description	consultation_date	urgency	consultation_mode	status	created_at
1	John	Doe	john@example.com	1234567890	123 Street, NY	Atty. Smith	Criminal Law	Theft	Stolen bike case	2024-06-10	High	In-person	Confirmed	2024-06-01
1	John	Doe	john@example.com	1234567890	123 Street, NY	Atty. Smith	Family Law	Theft	Stolen bike case	2024-06-10	Emergency	Online	Confirmed	2024-06-01

Applying 2NF:

```
CREATE TABLE Booking_2NF (
    booking_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100),
    phone VARCHAR(20),
    address VARCHAR(255),lawyer_id INT,
```

```
    case_type VARCHAR(100),
    case_description TEXT,
    consultation_date DATE,
    urgency VARCHAR(50),
    consultation_mode VARCHAR(50),
    status VARCHAR(50),
    created_at DATE
);

CREATE TABLE Lawyer (
    lawyer_id INT PRIMARY KEY,
    lawyer_name VARCHAR(100),
    specialization VARCHAR(100)
```

Second Normal Form (2NF)

Eliminate partial dependencies by creating separate Lawyer table.

Booking Table

booking_id	first_name	last_name	email	phone	address	lawyer_id	case_type	case_description	consultation_date	urgency	consultation_mode	status	created_at
1	John	Doe	john@example.com	1234567890	123 Street, NY	101	Theft	Stolen bike case	2024-06-10	High	In-person	Confirmed	2024-06-01

Lawyer Table

lawyer_id	lawyer_name	specialization
101	Atty. Smith	Criminal Law
102	Atty. Smith	Family Law

Applying 3NF:

```
CREATE TABLE Lawyer_3NF (
    lawyer_id INT PRIMARY KEY,
    lawyer_name VARCHAR(100),
    specialization_id INT
);

CREATE TABLE Specialization (
    specialization_id INT PRIMARY KEY,
    specialization_name VARCHAR(100)
);
```

Third Normal Form (3NF)

Move specialization into a separate Specialization table to eliminate transitive dependency.

Updated Lawyer Table

lawyer_id	lawyer_name	specialization_id
101	Atty. Smith	1
102	Atty. Smith	2

Specialization Table

specialization_id	specialization_name
1	Criminal Law
2	Family Law

Applying 4NF:

CREATE TABLE Booking_Consultation_Mode (

booking_id INT,

```
    consultation_mode VARCHAR(50)
);

CREATE TABLE Booking_Urgency (
    booking_id INT,
    urgency VARCHAR(50)
);
```

Fourth Normal Form (4NF)

Separate multivalued fields like consultation mode and urgency into separate tables.

Booking_Consultation_Mode Table

booking_id	consultation_mode
1	In-person
1	Online

Booking_Urgency Table

booking_id	urgency
1	High
1	Emergency

Applying 5NF:

```
CREATE TABLE Booking_Lawyer (
    booking_id INT,
    lawyer_id INT
);

CREATE TABLE Booking_CaseType (
```

```
    booking_id INT,
    case_type VARCHAR(100)
);

CREATE TABLE Booking_Mode (
    booking_id INT,
    consultation_mode VARCHAR(50)
);
```

Fifth Normal Form (5NF)

Decompose further if bookings can independently associate with multiple lawyers, consultation modes, and case types.

Booking_Lawyer Table

booking_id	lawyer_id
1	101
1	102

Booking_CaseType Table

booking_id	case_type
1	Theft

Booking_Mode Table

booking_id	consultation_mode
1	In-person
1	Online

TRANSACTIONS

1. **Basic Transaction:**

START TRANSACTION;

INSERT INTO User (name) VALUES ('New User');

INSERT INTO Booking (user_id, lawyer_id, booking_date)

VALUES (LAST_INSERT_ID(), 2, '2025-05-05');

COMMIT;

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO User (name) VALUES ('New User');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO Booking (user_id, lawyer_id, booking_date)
    -> VALUES (LAST_INSERT_ID(), 2, '2025-05-05');
Query OK, 1 row affected (0.01 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)
```

Figure 13.1 : First Transaction

2.START TRANSACTION;

SELECT id FROM users WHERE email = 'legalaid@gmail.com' FOR UPDATE;

SELECT id FROM lawyer WHERE name = 'Bob Smith' FOR UPDATE;

INSERT INTO booking (

first_name, last_name, email, phone, address,

lawyer_name, specialization, case_type, case_description,

consultation_date, urgency, consultation_mode, status, created_at

) VALUES (

'ABC', 'def', 'legalaid@gmail.com', '1234567891', 'Some address here',

'Bob Smith', 'Criminal Law', 'Civil', 'Brief description of case',

'2025-05-06', 'Emergency', 'Online', 'Pending', NOW()

);

COMMIT;

```
mysql> START TRANSACTION;
mysql>
mysql> SELECT id FROM users WHERE email = 'legalaid@gmail.com' FOR UPDATE;
mysql>
mysql> SELECT id FROM lawyer WHERE name = 'Bob Smith' FOR UPDATE;
mysql>
mysql> INSERT INTO booking (
    -> first_name, last_name, email, phone, address,
    -> lawyer_name, specialization, case_type, case_description,
    -> consultation_date, urgency, consultation_mode, status, created_at
    -> ) VALUES (
    -> 'ABC', 'def', 'legalaid@gmail.com', '1234567891', 'Some address here',
    -> 'Bob Smith', 'Criminal Law', 'Civil', 'Brief description of case',
    -> '2025-05-06', 'Emergency', 'Online', 'Pending', NOW()
    -> );
mysql>
mysql> COMMIT;
mysql>
```

Figure 13.2 : Second Transaction

3. START TRANSACTION;

UPDATE users SET phone = '9876543210', address = 'New Address' WHERE email = 'legalaid@gmail.com';

UPDATE booking SET status = 'Confirmed' WHERE email = 'legalaid@gmail.com' AND status = 'Pending';

COMMIT;

```
mysql> START TRANSACTION;
mysql>
mysql> UPDATE users SET phone = '9876543210', address = 'New Address' WHERE email = 'legalaid@gmai
mysql>
mysql> UPDATE booking SET status = 'Confirmed' WHERE email = 'legalaid@gmail.com' AND status = 'Pe
mysql>
mysql> COMMIT;
mysql>
```

Figure 13.3 : Third Transaction

4.START TRANSACTION;

DELETE FROM booking WHERE email = 'legalaid@gmail.com' AND consultation_date = '2025-05-06';

DELETE FROM users WHERE email = 'legalaid@gmail.com';

COMMIT;

```
mysql> START TRANSACTION;
mysql>
mysql> DELETE FROM booking WHERE email = 'legalaid@gmail.com' AND consultation_date = '2025-05-0
mysql>
mysql> DELETE FROM users WHERE email = 'legalaid@gmail.com';
mysql>
mysql> COMMIT;
mysql>
```

Figure 13.4 : Fourth Transaction

CODE

VIEWS.PY:-

```
from django.shortcuts import render, redirect
import MySQLdb
from django.contrib import messages
from django.contrib.auth.hashers import make_password, check_password

def get_db_connection():
    try:
        return MySQLdb.connect(
            host="localhost",
            user="root",
            password="password",
            database="legal_aid",
            charset="utf8"
        )
    except MySQLdb.Error as e:
        print(f"Database Connection Error: {e}")
        return None

def signlogin(request):
    if request.method == "POST":
        action = request.POST.get("action", "").strip()

        if action == "signup":
            first_name = request.POST.get("firstName", "").strip()
            last_name = request.POST.get("lastName", "").strip()
            email = request.POST.get("signupEmail", "").strip()
            phone = request.POST.get("phone", "").strip()
            legal_issue_category = request.POST.get("legalIssue", "").strip()
           password = make_password(request.POST.get("signupPassword", "").strip())  # Hashed
password

            if not all([first_name, last_name, email, phone, legal_issue_category, password]):
                messages.error(request, "All fields are required!")
                return redirect('/signlogin/')

            try:
                with get_db_connection() as conn:
                    with conn.cursor() as cursor:
                        cursor.execute("SELECT id FROM users WHERE email = %s", (email,))
                        if cursor.fetchone():
                            messages.error(request, "Email already exists. Please use another email.")
                            return redirect('/signlogin/')

                                query = """INSERT INTO users (first_name, last_name, email, phone,
legal_issue_category, password)
```

```
                    VALUES (%s, %s, %s, %s, %s, %s)"""
                        cursor.execute(query, (first_name, last_name, email, phone,
legal_issue_category, password))
                    conn.commit()
                    messages.success(request, "Account created successfully! Please login.")
            except Exception as e:
                messages.error(request, f"Error: {str(e)}")
                print(f"Signup Error: {e}")

            return redirect('/signlogin/')

        elif action == "login":
            email = request.POST.get("email", "").strip()
            password = request.POST.get("password", "").strip()

            if not email or not password:
                messages.error(request, "Email and password are required.")
                return redirect('/signlogin/')

            try:
                with get_db_connection() as conn:
                    with conn.cursor() as cursor:
                        cursor.execute("SELECT id, first_name, password FROM users WHERE email
= %s", (email,))
                    user = cursor.fetchone()

                    if user and check_password(password, user[2]):
                        request.session['user_id'] = user[0]
                        request.session['user_name'] = user[1]
                        return redirect('/home/')
                    else:
                        messages.error(request, "Invalid email or password.")
            except Exception as e:
                messages.error(request, f"Error: {str(e)}")
                print(f"Login Error: {e}")

    return render(request, "signlogin.html")

def logout_view(request):
    request.session.flush()
    return redirect('/signlogin/')

def index(request):
    return render(request, "index.html")

def chat_page(request):
    return render(request, "chat.html")

def case_study(request):  # Fixed function name
    return render(request, "case.html")
```

```
def booking(request):
    if request.method == "POST":
        first_name = request.POST.get("firstName", "").strip()
        last_name = request.POST.get("lastName", "").strip()
        email = request.POST.get("email", "").strip()
        phone = request.POST.get("phone", "").strip()
        address = request.POST.get("address", "").strip()
        lawyer_name = request.POST.get("lawyer", "").strip()
        specialization = request.POST.get("specialization", "").strip()
        case_type = request.POST.get("caseType", "").strip()
        case_description = request.POST.get("caseDescription", "").strip()
        consultation_date = request.POST.get("consultationDate", "").strip()
        urgency = request.POST.get("urgency", "").strip()
        consultation_mode = request.POST.get("consultationMode", "").strip()

        if not all([first_name, last_name, email, phone, lawyer_name, case_type, case_description,
consultation_date]):
            messages.error(request, "All required fields must be filled!")
            return redirect('/booking/')

        try:
            with get_db_connection() as conn:
                with conn.cursor() as cursor:
                    booking_query = """
                        INSERT INTO booking (first_name, last_name, email, phone, address,
lawyer_name, specialization,
                                case_type, case_description, consultation_date, urgency,
consultation_mode, status)
                    VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, 'Pending')
                    """
                    cursor.execute(booking_query,
                        (first_name, last_name, email, phone, address, lawyer_name, specialization,
                         case_type, case_description, consultation_date, urgency, consultation_mode))
                    conn.commit()
                    messages.success(request, "Booking successfully created!")
        except Exception as e:
            messages.error(request, f"Error: {str(e)}")
            print(f"Booking Error: {e}")

        return redirect('/booking/')

    return render(request, "booking.html")

def lawyer(request):
    return render(request,"lawyer.html")

def knowledge(request):
    return render(request,"knowledge.html")
```

```
def court(request):
    return render(request,"court.html")
```

IMPLEMENTATION RESULT

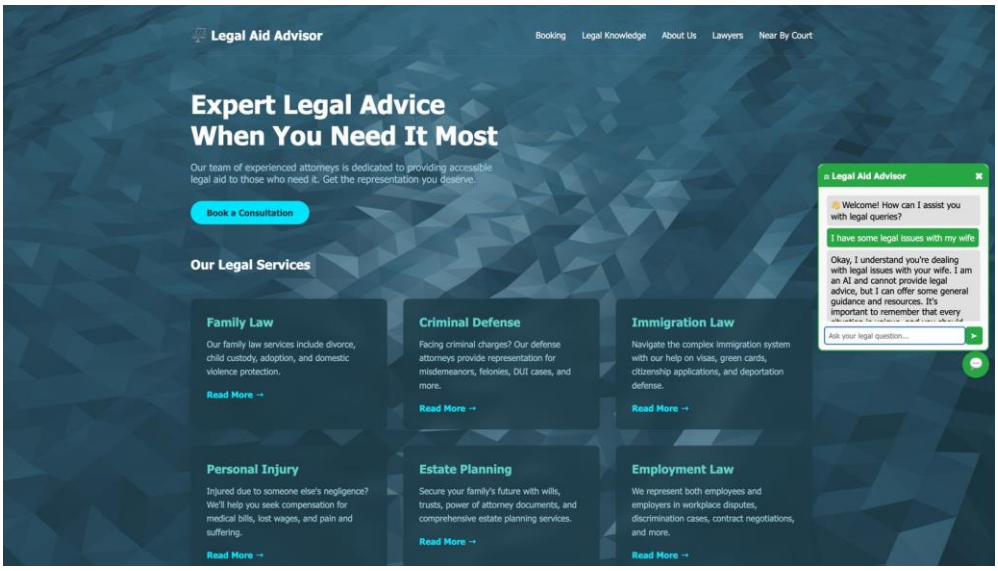

Figure 14.1 : Home Page

Our homepage offers a seamless and intuitive interface where users can quickly access expert legal advice, book consultations, and explore services across various legal domains.
With a responsive design and built-in chatbot, it ensures that anyone in need of legal guidance can get started with just a few clicks—anytime, anywhere.

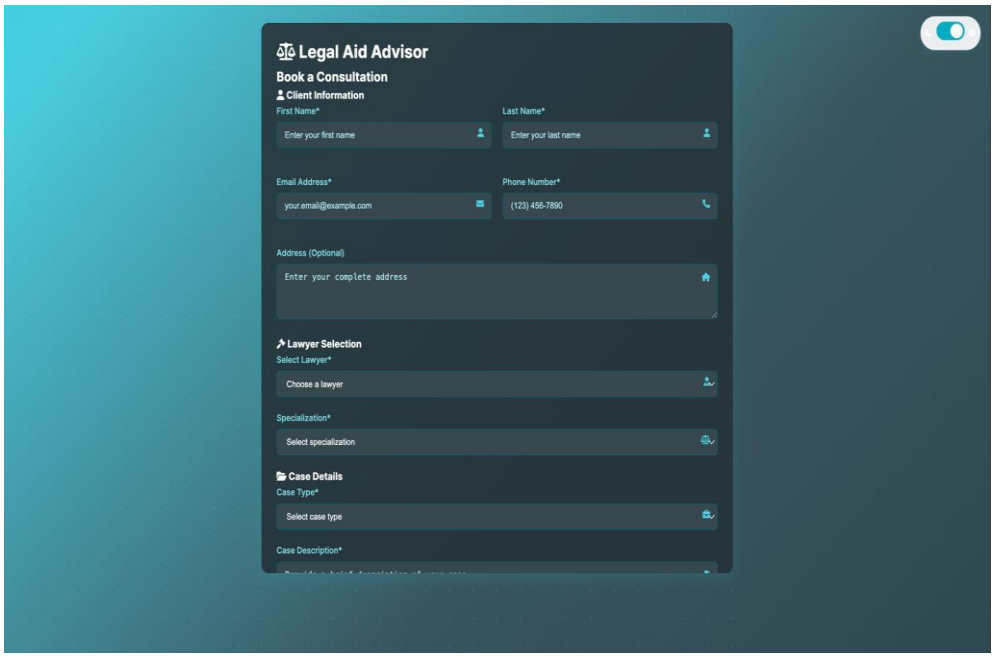

Figure 14.2 : Booking Page

The booking page enables users to seamlessly schedule legal consultations by entering their details, selecting a lawyer, and submitting case information—all in one place.

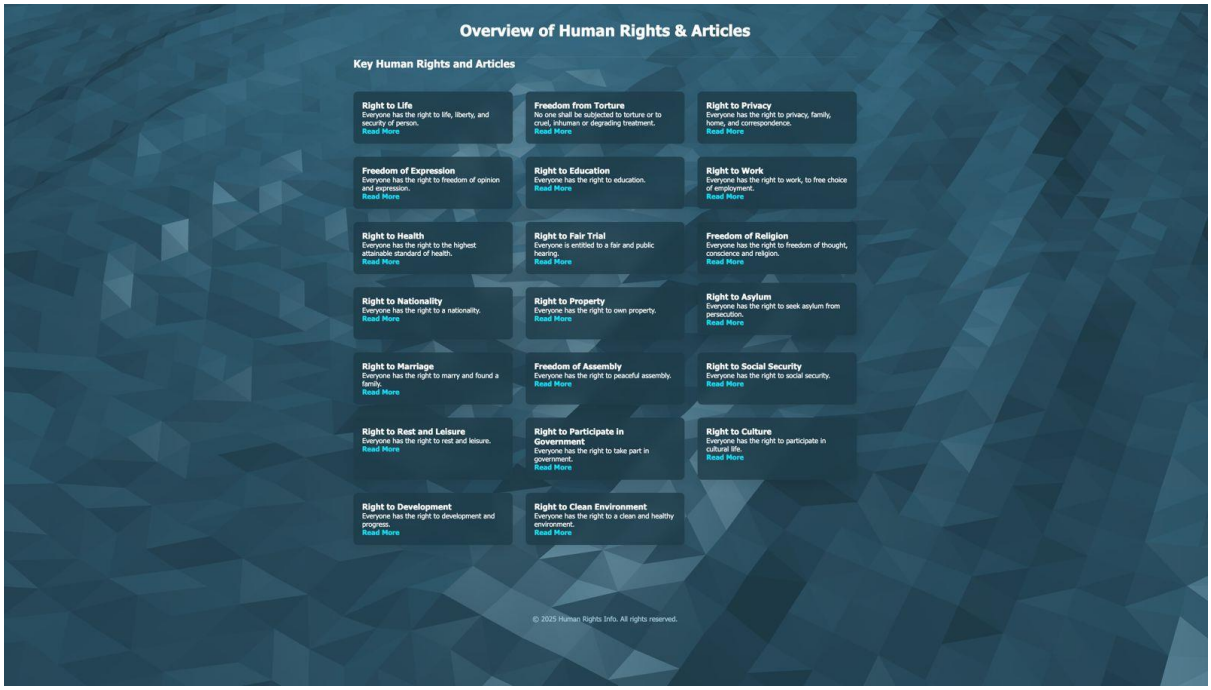

Figure 14.3 : Human Rights & Articles

This page offers a comprehensive overview of fundamental human rights and articles, empowering users with knowledge about their legal and moral entitlements.

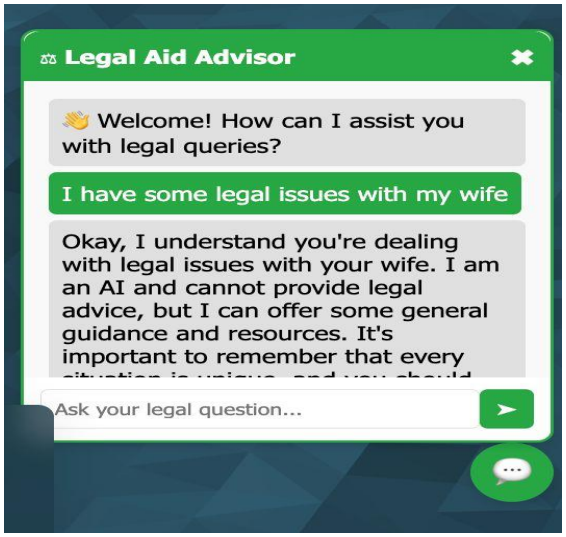

Figure 14.4 : ChatBot

The chatbot page provides real-time legal assistance by answering user queries instantly, making legal guidance accessible and interactive 24/7

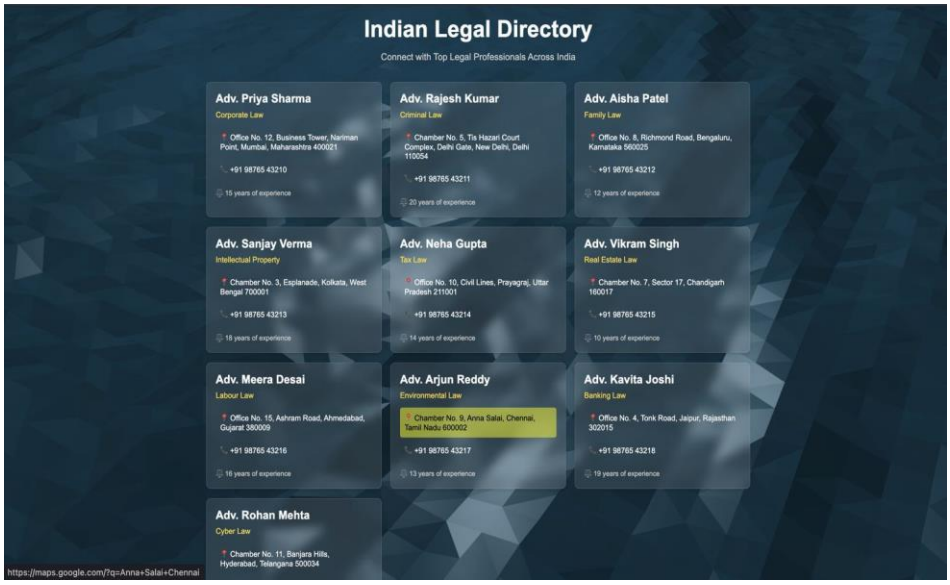

Figure 14.5 : Lawyers Section

The Indian Legal Directory showcases top legal professionals across India, each specializing in different branches of law with over a decade of experience.

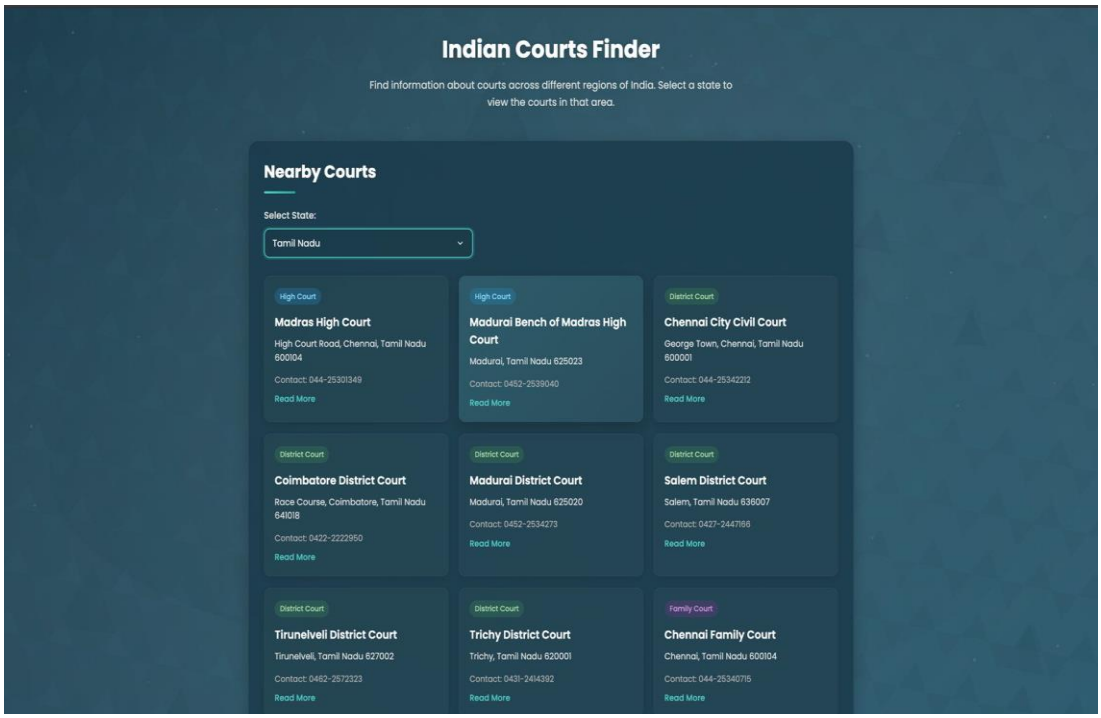

Figure 14.6 : Near-by Courts Section

The Nearby Courts section displays a list of major high, district, and family courts across Tamil Nadu, helping users find court names, locations, and contact details based on their selected state.

CONCLUSION

The Legal Aid Advisor system stands as a significant step toward democratizing access to legal assistance. In a world where legal complexities often act as barriers for the common person, this platform offers a much-needed bridge between individuals and the legal support they require. By providing a centralized and user-friendly interface, users can easily find and connect with lawyers based on their specialization, location, and contact availability. This not only saves time and effort but also ensures that users receive guidance tailored to their specific legal concerns.

The system's database-driven structure allows for efficient storage, retrieval, and management of lawyer and user information. It simplifies the process of searching for legal experts and fosters transparency in the communication process. Furthermore, it can help reduce the backlog of legal queries in physical legal aid centers by providing initial virtual guidance.

As society moves toward digital transformation, especially in essential services like legal aid, tools like the Legal Aid Advisor become vital. With further development, this platform can incorporate features like AI-powered legal query classification, chatbot-based support, automated appointment scheduling, multilingual support, and even video consultations to broaden its usability and inclusiveness.

In conclusion, the Legal Aid Advisor is not just a project—it is a scalable solution with the potential to make legal assistance more approachable, affordable, and accessible to all, especially the underprivileged and those unfamiliar with legal procedures. It reflects the vision of using technology to serve justice and empower individuals through information and support.