

**UNIVERSITY OF ENGINEERING AND TECHNOLOGY, TAXILA**

**Course: Database Management System**

**Course Instructor: Ma’am Sana Ziaffat Lab Instructor: Engr. Shahid Ali**

**Hotel Management System**

**Project Report**

**Submitted By: Areeba Bashir, Sanwal Mumtaz**

**22-CP-92, 22-CP-**

**Skill Hub – Database Management System Project Report**

**Table of Contents**

**Chapter 1: Introduction**

1.1 Project Overview  
1.2 Objectives  
1.3 System Features  
1.4 User Roles  
1.5 Security Considerations  
1.6 Functional Modules  
1.7 Expected Outcomes

**Chapter 2: Entity Relationship Model**

2.1 Introduction to ER Model  
2.2 Entity Definitions  
2.3 Relationships Between Entities  
2.4 ER Diagram  
2.5 Entity Constraints and Business Rules

**Chapter 3: Logical Design (Relational Model)**

3.1 Introduction to Relational Model  
3.2 Table Structures and Attributes  
3.3 Relationship Implementation

**Chapter 4: Implementation**

4.1 Technology Stack (MERN)  
4.2 Backend Architecture  
4.3 Frontend Design  
4.4 User Authentication & Authorization  
4.5 CRUD Operations  
4.6 Payment Integration  
4.7 Data Validation & Error Handling

**Conclusion**

* Summary of the Project
* Challenges Faced

### Chapter 1: Introduction

**1.1 Project Overview**  
SkillHub is a full-stack web application built using the MERN stack that functions as a marketplace platform for freelance services. It allows users to sign up as freelancers to offer services (called gigs) or as clients to browse, select, and purchase these services. The platform includes key features such as secure authentication, dynamic gig creation, responsive listings, and integrated payment options.

**1.2 Objectives**

* To design and implement a scalable gig marketplace platform.
* To provide secure login, signup, and password recovery functionalities.
* To enable users to create, display, and manage service gigs.
* To incorporate real-time payment integration for service transactions.
* To ensure user-friendly navigation and responsiveness across devices.

**1.3 System Features**

* User Registration and Login
* Password Recovery System
* Create and Manage Gigs
* View All Gigs by Category/Tag
* Payment Gateway Integration
* Secure Session Handling
* Admin Panel for Management (optional)

**1.4 User Roles**

* **Freelancer:** Can register, create gigs, and manage listings.
* **Client/User:** Can browse gigs, place orders, and make payments.
* **Admin (optional):** Manages users, gigs, and handles content moderation.

**1.5 Security Considerations**

* Passwords are hashed using bcrypt before storing in the database.
* JWT-based authentication ensures secure session management.
* Role-based access control for admin and user functions.
* Input validation and sanitization to prevent injection attacks.

**1.6 Functional Modules**

* **Authentication Module** – Handles login, signup, and forgot password.
* **Gig Module** – Create, read, update, and delete gigs.
* **Payment Module** – Integrates with payment gateway APIs.
* **User Management Module** – Profile editing and role-based access.

**1.7 Expected Outcomes**

* A fully functional MERN-based gig marketplace.
* Secure and scalable user authentication and gig management system.
* Streamlined user experience with payment capabilities.
* Efficient data design through proper database modeling.

### Chapter 2: Entity Relationship Model

**2.1 Introduction to ER Model**  
The Entity Relationship (ER) model is a high-level data model used to define the data elements and relationships for a given system. It visually represents the structure of the database with the help of entities, attributes, and relationships. For the SkillHub project, the ER model outlines how users, gigs, and transactions are connected.

**2.2 Entity Definitions**

* **User:** Represents the individuals who can either be clients or freelancers. Attributes include UserID, Name, Email, Password, Role.
* **Gig:** Represents a service offered by a freelancer. Attributes include GigID, Title, Description, Price, Category, Tag, UserID (FK).
* **Order:** Represents the booking or purchase of a gig. Attributes include OrderID, GigID (FK), UserID (FK), Status, PaymentStatus, Date.
* **Payment:** Represents the payment transaction for an order. Attributes include PaymentID, OrderID (FK), Amount, Method, Date.

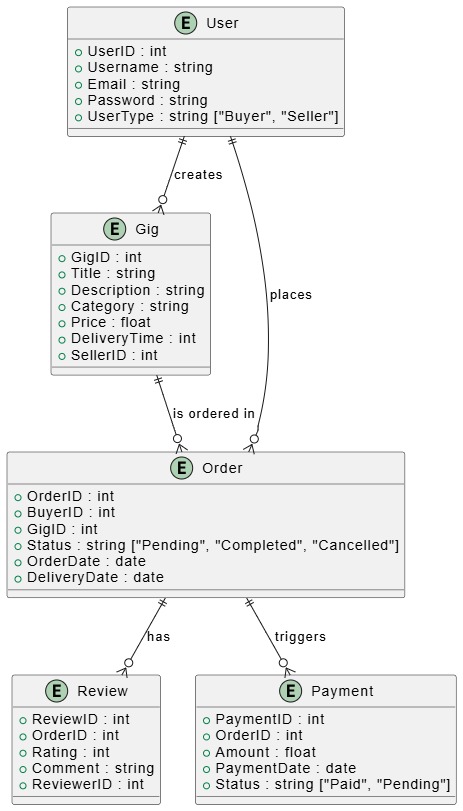
**2.3 Relationships Between Entities**

* A **User** can create multiple **Gigs** (One-to-Many).
* A **User** can place multiple **Orders** (One-to-Many).
* Each **Order** is linked to one **Gig** (Many-to-One).
* Each **Order** has one associated **Payment** (One-to-One).

**2.4 ER Diagram**  
The ER diagram for SkillHub shows the relationship among User, Gig, Order, and Payment entities. The cardinalities reflect the business logic of the platform. (Attach diagram separately or digitally)

**2.5 Entity Constraints and Business Rules**

* Email addresses in the User entity must be unique.
* Gigs must be associated with an existing user (freelancer).
* Orders can only be placed for available gigs.
* Payments must be linked to confirmed orders.
* Only clients can place orders, and only freelancers can create gigs.



### Chapter 3: Logical Design (Relational Model)

**3.1 Introduction to Relational Model**The relational model represents the database structure using tables (relations). Each table consists of rows (tuples) and columns (attributes). This model provides a logical view of the database and supports data integrity, consistency, and easy querying using SQL.

**3.2 Table Structures and Attributes**

* **Users**(UserID, Name, Email, Password, Role)
* **Gigs**(GigID, Title, Description, Price, Category, Tag, UserID)
* **Orders**(OrderID, GigID, UserID, Status, PaymentStatus, Date)
* **Payments**(PaymentID, OrderID, Amount, Method, Date)

**3.3 Relationship Implementation**

* One-to-Many: A user can post multiple gigs; a user can place multiple orders.
* Many-to-One: Each order is for one gig.
* One-to-One: Each order has one payment record.

**User:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UserId | username | email | password | userType |

**Gig:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| GigId | Title | Description | Category | Price | DeliveryTime | SellerID | UserID |

**Order:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| OrderID | BuyerID | UserID | GigID | Status | OrderDate | DeliveryDate |

**Review:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ReviewID | OrderID | Rating | Comment | ReviewerID |

**Payment:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PaymentID | OrderID | Amount | PaymentDate | Status |

### Chapter 4: Implementation

#### **4.1 Technology Stack (MERN)**

The SkillHub platform is built using the MERN stack:

* **MongoDB**: NoSQL database used to store users, gigs, orders, and payments in a flexible, document-oriented format.
* **Express.js**: Web framework used to handle HTTP requests and route management on the backend.
* **React.js**: Frontend library for building a dynamic, component-based user interface.
* **Node.js**: Runtime environment that executes server-side JavaScript and hosts the Express app.

#### **4.2 Backend Architecture**

The backend follows a RESTful API architecture with modularized routes and controllers:

* Routes are defined for authentication, gigs, users, orders, and payments.
* Controllers contain business logic and interact with Mongoose models.
* Middleware is used for authentication, error handling, and validation.
* JWT is used for securing protected routes.

#### **4.3 Frontend Design**

The frontend is designed using React and styled with Tailwind CSS and Bootstrap:

* Pages include Home, Login, Register, Dashboard, Gig Listings, and Payment Page.
* React Router is used for navigation between components.
* Axios is used for making API requests to the backend.
* Forms are responsive and mobile-friendly.

#### **4.4 User Authentication & Authorization**

* Users register and log in via secure forms.
* Passwords are hashed using bcrypt before storage.
* JWT tokens are issued upon login and used for session handling.
* Role-based access ensures that only freelancers can create gigs and only clients can place orders.
* Protected routes redirect unauthenticated users.

#### **4.5 CRUD Operations**

Each core entity supports full CRUD operations:

* **Users**: Update profile, delete account.
* **Gigs**: Create, read, update, delete by freelancers.
* **Orders**: Place, track, and manage orders by clients.
* **Admin (optional)**: Has CRUD access to all entities for moderation.

#### **4.6 Payment Integration**

* Payments are processed usingdummy payments.

#### **4.7 Data Validation & Error Handling**

* Joi and express-validator are used to validate user inputs on the backend.
* Try-catch blocks and middleware handle errors gracefully.
* Custom error messages help users understand issues (e.g., invalid credentials, duplicate email).
* Frontend also performs basic form validation before submission.

### Some visuals of our website:

### Home page:

### Register page:

### 

### Forget password:

### 

### Contact us page:

### 

### Gig create page:

### Gigs show page:

### 

### Oder done :

### 

### Database :

### Register:

### 

### Gig:

### 

### Payment:

### 

### Conclusion

#### **Summary of the Project**

SkillHub is a scalable and secure gig marketplace application developed using the MERN stack. It enables users to register as freelancers or clients, create and purchase gigs. The application follows a modular design, enforces robust security practices, and provides a smooth user experience across devices.

#### **Challenges Faced**

* Managing complex role-based access and protected routes.
* Ensuring cross-browser compatibility and responsive design.
* Debugging CORS and deployment-related issues during API integration.