**SVKM’s NMIMS**

**School of Technology Management & Engineering (Indore Campus)**

**Computer Engineering Department (B Tech/MBATech CE and B Tech AIDS Sem IV)**

**Database Management System**

**Project Report**

|  |  |  |
| --- | --- | --- |
| Program | B.Tech. C.E | |
| Semester | IV | |
| Name of the Project: | Skill Driven Freelance Marketplace**(The MetGig)** | |
|  | | |
| Details of Project Members |  |  |
| Batch | Roll No. | Name |
| A2 | D051 | Nabh Jain |
| A2 | D101 | Yashvi Patodi |
| A2 | D007 | Aditya Goyal |
| Date of Submission: 11-April-2025 | | |

**Github link of your project: https://github.com/Aditya4305/Skill-Driven-Freelance-Marketplace--TheMetGig-**

**Note:**

1. Create a readme file if you have multiple files
2. All files must be properly named (Example:R004\_DBMSProject)
3. Submit all relevant files of your work ( Report, all SQL files, Any other files)
4. **Plagiarism is highly discouraged (Your report will be checked for plagiarism)**

**Rubrics for the Project evaluation:**

|  |  |
| --- | --- |
| First phase of evaluation:  Innovative Ideas (5 Marks)  Design and Partial implementation (5 Marks) | 10 marks |
| Final phase of evaluation  Implementation, presentation and viva, Self-Learning and Learning Beyond classroom | 10 marks |

**PROJECT REPORT**

**Topic-** Skill Driven Freelance Marketplace

**Submitted By-**

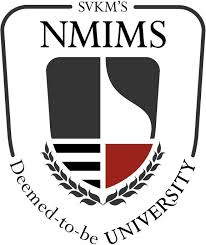
Yashvi Patodi (Roll number: D101)

Nabh Jain (Roll number: D051)

Aditya Goyal (Roll number: D007)

**Submitted To-**

Dr. Shruti Sharma Ma’am



**Course:**DBMS

**Academic Year:** 2024-25

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr no.** | **Topic** | **Page no.** |
| **1** | Storyline | 4-6 |
| **2** | Components of Database Design | 7-8 |
| **3** | Entity Relationship Diagram | 9 |
| **4** | Relational Model | 10-11 |
| **5** | Normalization | 12-19 |
| **6** | SQL Queries | 20-40 |
| **7** | Project Demonstration | 41-44 |
| **8** | Self-learning beyond classroom | 45-46 |
| **9** | Learning from the project | 47-48 |
| **10** | Challenges faced | 49 |
| **11** | Conclusion | 50 |

**I. Storyline**

### **Skill-Driven Freelance Marketplace- TheMetGig**



In today’s digital age, freelance platforms have revolutionized how people work. But even with numerous platforms available, many fail to iihighlight *skill compatibility* and personalized user experiences. *The Met Gig* was born out of the idea to bridge this gap — creating a freelance marketplace where the right skills meet the right opportunities. The goal? Build a frontend-driven, clean, and intuitive platform focused on *skills-first matchmaking* between freelancers and clients.

## **1. Platform Overview**

MetGig is a specialized freelance platform designed to create tailored experiences for both clients and freelancers. Built with a focus on skills-matching and user-friendly interfaces, the platform separates user journeys based on roles to maximize efficiency and minimize clutter.

### **Development Stack**

* Front-end: HTML5, CSS3, Vanilla JavaScript
* Back-end (planned): Node.js
* Database (planned): SQL

### **Core User Roles**

* **Clients**: Post projects, review bids, manage freelancers
* **Freelancers**: Discover relevant projects, submit bids, showcase skills

## **2. Key Features & Functionalities**

### **Role-Based User Experience**

* **Client Dashboard**: Project posting interface, freelancer management tools, progress tracking
* **Freelancer Dashboard**: Skill-filtered project discovery, portfolio management, bid tracking

### **Project Management System**

* Detailed project listing page with comprehensive filtering
* Sophisticated bidding mechanics allowing price quotes, timeline estimates, and portfolio attachments
* Custom evaluation tools for clients to assess bids based on multiple factors

### **Profile Ecosystem**

* Freelancer profiles with skill showcasing, certification validation, and portfolio display
* Client profiles featuring company information, project history, and bidirectional rating system

## **Additional Services & Enhancements**

### **Planned Backend Integration**

* Authentication and session management
* User data persistence and project tracking
* Security implementations

### **Future Expansion Features**

* Real-time messaging and notification system
* Payment gateway integration (Stripe, PayPal)
* Advanced matching algorithms

## **3. Database Structure & Efficiency**

### **User Data Management**

* Role-based user information storage
* Skill indexing for improved matching
* Project history and performance metrics

### **Project Information Architecture**

* Categorized project listings
* Bid storage and evaluation framework
* Historical data for trend analysis

## **4. Real-World Impact & Use Case**

### **Market Differentiation**

MetGig stands apart as a skill-first platform that prioritizes relevant connections between clients and freelancers, reducing noise and improving match quality.

### **Practical Applications**

The platform serves as a foundation for a scalable digital workplace solution that can transform how freelance professionals and clients discover each other and collaborate online.

### **Growth Potential**

With its modular architecture, MetGig is positioned to evolve with market demands and integrate additional services as the freelance economy continues to expand.

## **II. Components of Database Design**

#### **Entities & Attributes**

*(PK = Primary Key, FK = Foreign Key)*

* **Users** (user\_id - PK):
  + email (Unique), password\_hash, first\_name, last\_name, country\_code,
  + user\_type (freelancer, client, admin), account\_status (active, suspended, inactive), profile\_image.
* **FreelancerProfiles** (profile\_id - PK, user\_id - FK → Users):
  + bio, hourly\_rate, total\_earnings, rating, experience\_level (entry, intermediate, expert), visibility.
* **ClientProfiles** (profile\_id - PK, user\_id - FK → Users):
  + company\_name, company\_size (individual, small, medium, large), payment\_verified, total\_spent.
* **Projects** (project\_id - PK, client\_id - FK → Users):
  + title, description, budget\_min, budget\_max, status (draft, open, completed), duration, experience\_level.
* **Proposals** (proposal\_id - PK, project\_id - FK → Projects, freelancer\_id - FK → Users):
  + cover\_letter, bid\_amount, estimated\_duration, status (pending, accepted, rejected), is\_highlighted.
* **Contracts** (contract\_id - PK, project\_id - FK → Projects, proposal\_id - FK → Proposals):
  + contract\_type (fixed, hourly), amount, start\_date, end\_date, status (active, disputed).
* **Payments** (payment\_id - PK, contract\_id - FK → Contracts):
  + amount, status (pending, completed), platform\_fee, freelancer\_amount.
* **Reviews** (review\_id - PK, contract\_id - FK → Contracts):
  + rating (0–5), comment, is\_public.
* **Messages** (message\_id - PK, sender\_id/receiver\_id - FK → Users):
  + message\_text, sent\_at, read\_at, contract\_id (optional).
* **Notifications** (notification\_id - PK, user\_id - FK → Users):
  + notification\_type (message, payment, etc.), reference\_id, is\_read.
* **Skills** (skill\_id - PK), **Categories** (category\_id - PK), **PortfolioItems** (item\_id - PK):

### **Relationships**

1. **Clients** (via Users) **post Projects** (1:M).
2. **Freelancers** (via Users) **submit Proposals** to Projects (M:N).
3. **Projects** are linked to **Contracts** via accepted Proposals (1:1).
4. **Contracts** manage **Payments** (1:M) and **Reviews** (1:1).
5. **Users** communicate via **Messages** (M:M).
6. **Freelancers** showcase **PortfolioItems** (1:M).
7. **Notifications** are sent to **Users** based on activity (1:M).
8. **Projects** require **Skills** through ProjectSkills (M:N).
9. **Categories** organize **Projects** and **Skills** (1:M).

This structured schema ensures a **scalable, efficient, and well-integrated freelance marketplace**, prioritizing smooth interactions, secure payments, and a seamless bidding & hiring process.

## 

## 

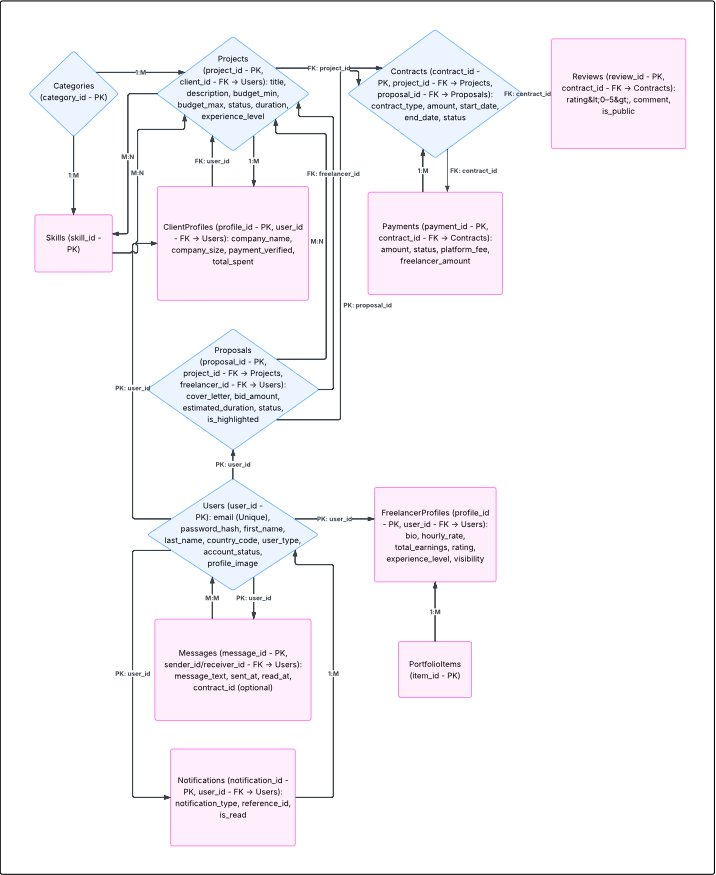
## 

## 

## 

## 

## **III. Entity Relationship Diagram**

****

## **IV. Relational Model**

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Attributes** | **Primary Key** |
| **Users** | user\_id, email, password\_hash, first\_name, last\_name, country\_code, user\_type, account\_status, profile\_image | user\_id |
| **FreelancerProfiles** | profile\_id, user\_id (FK → Users), bio, hourly\_rate, total\_earnings, rating, experience\_level, visibility | profile\_id |
| **ClientProfiles** | profile\_id, user\_id (FK → Users), company\_name, company\_size, payment\_verified, total\_spent | profile\_id |
| **Projects** | project\_id, client\_id (FK → Users), title, description, budget\_min, budget\_max, status, duration, experience\_level | project\_id |
| **Proposals** | proposal\_id, project\_id (FK → Projects), freelancer\_id (FK → Users), cover\_letter, bid\_amount, estimated\_duration, status, is\_highlighted | proposal\_id |
| **Contracts** | contract\_id, project\_id (FK → Projects), proposal\_id (FK → Proposals), contract\_type, amount, start\_date, end\_date, status | contract\_id |
| **Payments** | payment\_id, contract\_id (FK → Contracts), amount, status, platform\_fee, freelancer\_amount | payment\_id |
| **Reviews** | review\_id, contract\_id (FK → Contracts), rating, comment, is\_public | review\_id |
| **Messages** | message\_id, sender\_id (FK → Users), receiver\_id (FK → Users), message\_text, sent\_at, read\_at, contract\_id (optional FK → Contracts) | message\_id |
| **Notifications** | notification\_id, user\_id (FK → Users), notification\_type, reference\_id, is\_read | notification\_id |
| **Skills** | skill\_id, skill\_name | skill\_id |
| **Categories** | category\_id, category\_name | category\_id |
| **PortfolioItems** | item\_id, user\_id (FK → Users), title, description, file\_url, uploaded\_at | item\_id |
| **FreelancerSkills** | user\_id (FK → Users), skill\_id (FK → Skills) | user\_id, skill\_id |
| **SkillCategories** | skill\_id (FK → Skills), category\_id (FK → Categories) | skill\_id, category\_id |
| **ProjectCategories** | project\_id (FK → Projects), category\_id (FK → Categories) | project\_id, category\_id |

## 

## 

## 

## 

## 

## 

## 

## 

## **V. Normalization**

### **Tables in First Normal Form (1NF)**

|  |  |
| --- | --- |
| **Table Name** | Attributes (1NF) |
| **Users** | user\_id (PK), email (Unique), password\_hash, first\_name, last\_name, country\_code, user\_type, account\_status, profile\_image |
| **FreelancerProfiles** | profile\_id (PK), user\_id (FK), bio, hourly\_rate, total\_earnings, rating, experience\_level, visibility |
| **ClientProfiles** | profile\_id (PK), user\_id (FK), company\_name, company\_size, payment\_verified, total\_spent |
| **Projects** | project\_id (PK), client\_id (FK), title, description, budget\_min, budget\_max, status, duration, experience\_level |
| **Proposals** | proposal\_id (PK), project\_id (FK), freelancer\_id (FK), cover\_letter, bid\_amount, estimated\_duration, status, is\_highlighted |
| **Contracts** | contract\_id (PK), project\_id (FK), proposal\_id (FK), contract\_type, amount, start\_date, end\_date, status |
| **Payments** | payment\_id (PK), contract\_id (FK), amount, status, platform\_fee, freelancer\_amount |
| **Reviews** | review\_id (PK), contract\_id (FK), rating, comment, is\_public |
| **Messages** | message\_id (PK), sender\_id (FK), receiver\_id (FK), message\_text, sent\_at, read\_at, contract\_id (nullable FK) |
| **Notifications** | notification\_id (PK), user\_id (FK), notification\_type, reference\_id, is\_read |
| **Skills** | skill\_id (PK), skill\_name |
| **Categories** | category\_id (PK), category\_name |
| **PortfolioItems** | item\_id (PK), freelancer\_id (FK), title, description, link |

## Atomicity Check (1NF criteria):

## No multivalued fields (e.g., "skills" would be linked via a junction table, not listed as comma-separated).

## All fields hold singular values only.

## Every field is functionally dependent on the primary key of its table.

### **Tables in Second Normal Form (2NF)**

|  |  |
| --- | --- |
| **Table Name** | Attributes (2NF) |
| **Users** | user\_id (PK), email (Unique), password\_hash, first\_name, last\_name, country\_code, user\_type, account\_status, profile\_image |
| **FreelancerProfiles** | profile\_id (PK), user\_id (FK), bio, hourly\_rate, total\_earnings, rating, experience\_level, visibility |
| **ClientProfiles** | profile\_id (PK), user\_id (FK), company\_name, company\_size, payment\_verified, total\_spent |
| **Projects** | project\_id (PK), client\_id (FK), title, description, budget\_min, budget\_max, status, duration, experience\_level |
| **Proposals** | proposal\_id (PK), project\_id (FK), freelancer\_id (FK), cover\_letter, bid\_amount, estimated\_duration, status, is\_highlighted |
| **Contracts** | contract\_id (PK), project\_id (FK), proposal\_id (FK), contract\_type, amount, start\_date, end\_date, status |
| **Payments** | payment\_id (PK), contract\_id (FK), amount, status, platform\_fee, freelancer\_amount |
| **Reviews** | review\_id (PK), contract\_id (FK), rating, comment, is\_public |
| **Messages** | message\_id (PK), sender\_id (FK), receiver\_id (FK), message\_text, sent\_at, read\_at, contract\_id (optional FK) |
| **Notifications** | notification\_id (PK), user\_id (FK), notification\_type, reference\_id, is\_read |
| **Skills** | skill\_id (PK), skill\_name |
| **Categories** | category\_id (PK), category\_name |
| **PortfolioItems** | item\_id (PK), freelancer\_id (FK), title, description, link |

## 

### **For 2NF**

* Be in 1NF **AND no partial dependency.**
* Eliminate **partial dependencies** (i.e., no non-prime attribute is dependent on **part of a composite key**).

### **Changes Made:**

## **No table had a composite primary key, so partial dependency isn’t a big concern here.**

### **Tables in Third Normal Form (3NF)**

|  |  |
| --- | --- |
| **Table Name** | Attributes (3NF) |
| **Users** | user\_id (PK), email (Unique), password\_hash, first\_name, last\_name, country\_code, user\_type, account\_status, profile\_image |
| **FreelancerProfiles** | profile\_id (PK), user\_id (FK), bio, hourly\_rate, total\_earnings, rating, experience\_level, visibility |
| **ClientProfiles** | profile\_id (PK), user\_id (FK), company\_name, company\_size, payment\_verified, total\_spent |
| **Projects** | project\_id (PK), client\_id (FK), title, description, budget\_min, budget\_max, status, duration, experience\_level |
| **Proposals** | proposal\_id (PK), project\_id (FK), freelancer\_id (FK), cover\_letter, bid\_amount, estimated\_duration, status, is\_highlighted |
| **Contracts** | contract\_id (PK), project\_id (FK), proposal\_id (FK), contract\_type, amount, start\_date, end\_date, status |
| **Payments** | payment\_id (PK), contract\_id (FK), amount, status, platform\_fee, freelancer\_amount |
| **Reviews** | review\_id (PK), contract\_id (FK), rating, comment, is\_public |
| **Messages** | message\_id (PK), sender\_id (FK), receiver\_id (FK), message\_text, sent\_at, read\_at, contract\_id (nullable FK) |
| **Notifications** | notification\_id (PK), user\_id (FK), notification\_type, reference\_id, is\_read |
| **Skills** | skill\_id (PK), skill\_name |
| **Categories** | category\_id (PK), category\_name |
| **PortfolioItems** | item\_id (PK), freelancer\_id (FK), title, description, link |

### 

### For 3NF

### Be in 2NF

### Remove transitive dependencies (non-key → non-key relationships).

### Changes Made:

### Moved dependent fields into new tables:

### user\_type, account\_status, experience\_level, company\_size, project\_status, proposal\_status, etc. were moved to separate lookup tables.

### This helped avoid transitive dependencies like:

### user\_id → user\_type → description (which violates 3NF because user\_id indirectly determines description through user\_type)

### Ensured every non-key attribute is only dependent on the primary key.

### **Tables in Boyce-Codd Normal Form Form (BCNF)**

|  |  |
| --- | --- |
| **Table Name** | **Attributes (PK, FK)** |
| **Users** | user\_id (PK), email, password\_hash, first\_name, last\_name, country\_code, user\_type\_id (FK), account\_status\_id (FK), profile\_image |
| **UserTypes** | user\_type\_id (PK), type\_name (freelancer, client, admin) |
| **AccountStatuses** | account\_status\_id (PK), status\_name (active, suspended, inactive) |
| **FreelancerProfiles** | profile\_id (PK), user\_id (FK), bio, hourly\_rate, total\_earnings, rating, experience\_level\_id (FK), visibility |
| **ExperienceLevels** | experience\_level\_id (PK), level\_name (entry, intermediate, expert) |
| **ClientProfiles** | profile\_id (PK), user\_id (FK), company\_name, company\_size\_id (FK), payment\_verified, total\_spent |
| **CompanySizes** | company\_size\_id (PK), size\_name (individual, small, medium, large) |
| **Projects** | project\_id (PK), client\_id (FK), title, description, budget\_min, budget\_max, status\_id (FK), duration, experience\_level\_id (FK) |
| **ProjectStatuses** | status\_id (PK), status\_name (draft, open, completed) |
| **Proposals** | proposal\_id (PK), project\_id (FK), freelancer\_id (FK), cover\_letter, bid\_amount, estimated\_duration, status\_id (FK), is\_highlighted |
| **ProposalStatuses** | status\_id (PK), status\_name (pending, accepted, rejected) |
| **Contracts** | contract\_id (PK), project\_id (FK), proposal\_id (FK), contract\_type\_id (FK), amount, start\_date, end\_date, status\_id (FK) |
| **ContractTypes** | contract\_type\_id (PK), type\_name (fixed, hourly) |
| **ContractStatuses** | status\_id (PK), status\_name (active, disputed) |
| **Payments** | payment\_id (PK), contract\_id (FK), amount, status\_id (FK), platform\_fee, freelancer\_amount |
| **PaymentStatuses** | status\_id (PK), status\_name (pending, completed) |
| **Reviews** | review\_id (PK), contract\_id (FK), rating, comment, is\_public |
| **Messages** | message\_id (PK), sender\_id (FK), receiver\_id (FK), message\_text, sent\_at, read\_at, contract\_id (FK, optional) |
| **Notifications** | notification\_id (PK), user\_id (FK), notification\_type\_id (FK), reference\_id, is\_read |
| **NotificationTypes** | notification\_type\_id (PK), type\_name (message, payment, etc.) |
| **Skills** | skill\_id (PK), skill\_name |
| **Categories** | category\_id (PK), category\_name |
| **PortfolioItems** | item\_id (PK), freelancer\_id (FK), title, description, link |
| **FreelancerSkills** | freelancer\_id (FK), skill\_id (FK) |
| **ProjectCategories** | project\_id (FK), category\_id (FK) |

### **For BCNF**

## Be in 3NF AND

## Ensure every determinant is a candidate key (i.e., for every functional dependency X → Y, X is a superkey).

### **Changes Made:**

## Further refined tables to remove any cases where:

## A non-superkey determines another field.

**VI. SQL Queries**

**Queries to Create Table**

## **1.Users Table**

CREATE TABLE Users (

user\_id INT PRIMARY KEY AUTO\_INCREMENT,

email VARCHAR(100) UNIQUE NOT NULL,

password\_hash VARCHAR(255) NOT NULL,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

country\_code CHAR(2) NOT NULL,

user\_type ENUM('freelancer', 'client', 'admin') NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

last\_login TIMESTAMP NULL,

account\_status ENUM('active', 'suspended', 'inactive') DEFAULT 'active',

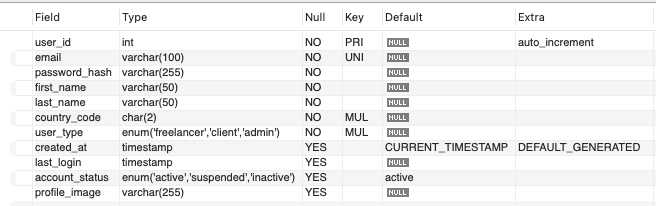
profile\_image VARCHAR(255) NULL,

INDEX idx\_email (email),

INDEX idx\_user\_type (user\_type),

INDEX idx\_country (country\_code)

);



## **2.FreelancerProfiles Table**

CREATE TABLE FreelancerProfiles (

profile\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

bio TEXT NOT NULL,

hourly\_rate DECIMAL(10,2) NOT NULL,

total\_earnings DECIMAL(15,2) DEFAULT 0.00,

rating DECIMAL(3,2) DEFAULT 0.00,

completion\_rate DECIMAL(5,2) DEFAULT 0.00,

visibility BOOLEAN DEFAULT TRUE,

title VARCHAR(100) NULL,

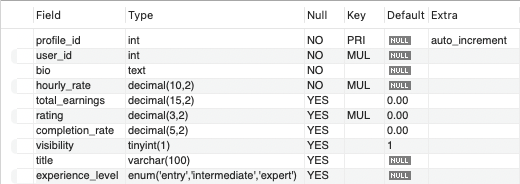
experience\_level ENUM('entry', 'intermediate', 'expert') NULL,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE,

INDEX idx\_hourly\_rate (hourly\_rate),

INDEX idx\_rating (rating)

);



## **3.ClientProfiles Table**

CREATE TABLE ClientProfiles (

profile\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

company\_name VARCHAR(100) NULL,

company\_website VARCHAR(255) NULL,

company\_description TEXT NULL,

company\_size ENUM('individual', 'small', 'medium', 'large') NULL,

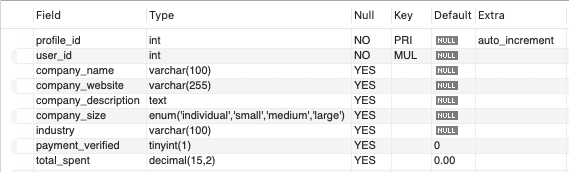
industry VARCHAR(100) NULL,

payment\_verified BOOLEAN DEFAULT FALSE,

total\_spent DECIMAL(15,2) DEFAULT 0.00,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE

);



## **4.Skills Table**

CREATE TABLE Skills (

skill\_id INT PRIMARY KEY AUTO\_INCREMENT,

skill\_name VARCHAR(50) UNIQUE NOT NULL,

category\_id INT NOT NULL,

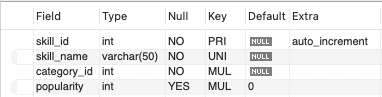
popularity INT DEFAULT 0,

FOREIGN KEY (category\_id) REFERENCES Categories(category\_id),

INDEX idx\_skill\_name (skill\_name),

INDEX idx\_popularity (popularity)

);



## **5.Categories Table**

CREATE TABLE Categories (

category\_id INT PRIMARY KEY AUTO\_INCREMENT,

parent\_id INT NULL,

category\_name VARCHAR(100) NOT NULL,

description TEXT NULL,

icon VARCHAR(100) NULL,

is\_active BOOLEAN DEFAULT TRUE,

display\_order INT DEFAULT 0,

FOREIGN KEY (parent\_id) REFERENCES Categories(category\_id) ON DELETE SET NULL,

INDEX idx\_parent (parent\_id),

INDEX idx\_active (is\_active)

);



## **6.Projects Table**

CREATE TABLE Projects (

project\_id INT PRIMARY KEY AUTO\_INCREMENT,

client\_id INT NOT NULL,

title VARCHAR(200) NOT NULL,

description TEXT NOT NULL,

category\_id INT NOT NULL,

subcategory\_id INT NULL,

budget\_min DECIMAL(10,2) NULL,

budget\_max DECIMAL(10,2) NULL,

is\_fixed\_price BOOLEAN DEFAULT TRUE,

hourly\_rate\_min DECIMAL(10,2) NULL,

hourly\_rate\_max DECIMAL(10,2) NULL,

duration ENUM('less\_than\_week', 'one\_week\_to\_month', 'one\_to\_three\_months', 'three\_to\_six\_months', 'more\_than\_six\_months') NULL,

experience\_level ENUM('entry', 'intermediate', 'expert', 'any') DEFAULT 'any',

status ENUM('draft', 'open', 'in\_progress', 'completed', 'cancelled') DEFAULT 'draft',

visibility ENUM('public', 'invite\_only', 'private') DEFAULT 'public',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

deadline DATE NULL,

FOREIGN KEY (client\_id) REFERENCES Users(user\_id),

FOREIGN KEY (category\_id) REFERENCES Categories(category\_id),

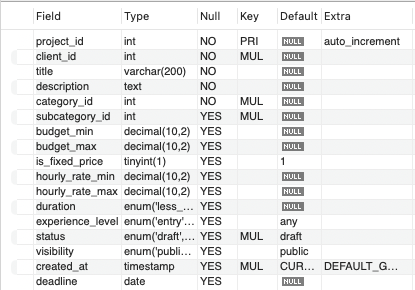
FOREIGN KEY (subcategory\_id) REFERENCES Categories(category\_id),

INDEX idx\_status (status),

INDEX idx\_created (created\_at),

INDEX idx\_category (category\_id)

);



## **7.ProjectSkills Table**

CREATE TABLE ProjectSkills (

project\_skill\_id INT PRIMARY KEY AUTO\_INCREMENT,

project\_id INT NOT NULL,

skill\_id INT NOT NULL,

importance ENUM('required', 'preferred', 'nice\_to\_have') DEFAULT 'required',

FOREIGN KEY (project\_id) REFERENCES Projects(project\_id) ON DELETE CASCADE,

FOREIGN KEY (skill\_id) REFERENCES Skills(skill\_id) ON DELETE CASCADE,

UNIQUE KEY unique\_project\_skill (project\_id, skill\_id)

);



## **8.Proposals Table**

CREATE TABLE Proposals (

proposal\_id INT PRIMARY KEY AUTO\_INCREMENT,

project\_id INT NOT NULL,

freelancer\_id INT NOT NULL,

cover\_letter TEXT NOT NULL,

bid\_amount DECIMAL(10,2) NOT NULL,

estimated\_duration INT NOT NULL, *-- in days*

status ENUM('pending', 'shortlisted', 'rejected', 'accepted', 'withdrawn') DEFAULT 'pending',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

is\_highlighted BOOLEAN DEFAULT FALSE,

FOREIGN KEY (project\_id) REFERENCES Projects(project\_id) ON DELETE CASCADE,

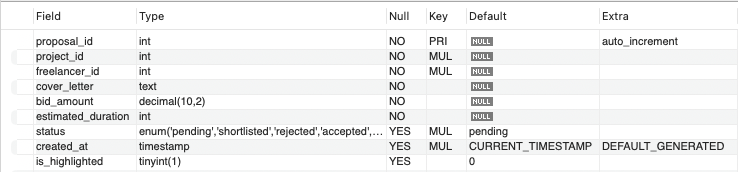
FOREIGN KEY (freelancer\_id) REFERENCES Users(user\_id) ON DELETE CASCADE,

UNIQUE KEY unique\_project\_freelancer (project\_id, freelancer\_id),

INDEX idx\_status (status),

INDEX idx\_created (created\_at)

);



## **9.PortfolioItems Table**

CREATE TABLE PortfolioItems (

item\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

title VARCHAR(100) NOT NULL,

description TEXT NULL,

project\_url VARCHAR(255) NULL,

image\_url VARCHAR(255) NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

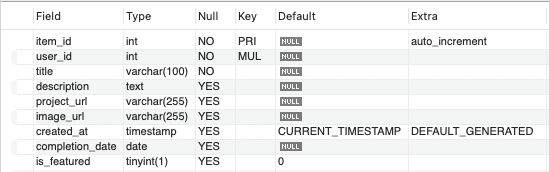
completion\_date DATE NULL,

is\_featured BOOLEAN DEFAULT FALSE,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE,

INDEX idx\_user\_id (user\_id)

);



## **10.Contracts Table**

CREATE TABLE Contracts (

contract\_id INT PRIMARY KEY AUTO\_INCREMENT,

project\_id INT NOT NULL,

client\_id INT NOT NULL,

freelancer\_id INT NOT NULL,

proposal\_id INT NOT NULL,

title VARCHAR(200) NOT NULL,

description TEXT NOT NULL,

contract\_type ENUM('fixed', 'hourly') NOT NULL,

amount DECIMAL(10,2) NOT NULL,

start\_date DATE NOT NULL,

end\_date DATE NULL,

status ENUM('active', 'completed', 'terminated', 'disputed') DEFAULT 'active',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (project\_id) REFERENCES Projects(project\_id),

FOREIGN KEY (client\_id) REFERENCES Users(user\_id),

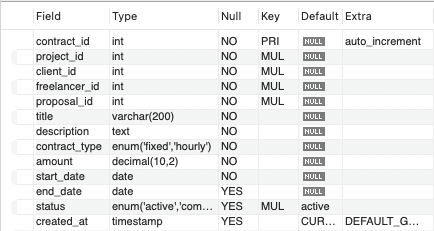
FOREIGN KEY (freelancer\_id) REFERENCES Users(user\_id),

FOREIGN KEY (proposal\_id) REFERENCES Proposals(proposal\_id),

INDEX idx\_project (project\_id),

INDEX idx\_status (status)

);



## **11.Payments Table**

CREATE TABLE Payments (

payment\_id INT PRIMARY KEY AUTO\_INCREMENT,

contract\_id INT ,

amount DECIMAL(10,2),

payment\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

status ENUM('pending', 'completed', 'failed', 'refunded') DEFAULT 'pending',

payment\_method VARCHAR(50) NULL,

transaction\_id VARCHAR(100) NULL,

platform\_fee DECIMAL(10,2),

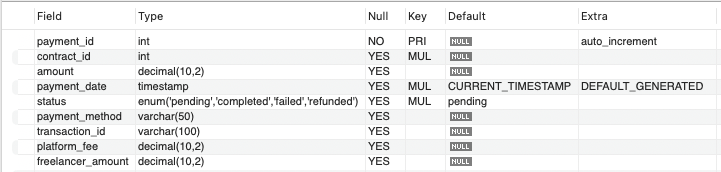
freelancer\_amount DECIMAL(10,2),

FOREIGN KEY (contract\_id) REFERENCES Contracts(contract\_id),

INDEX idx\_status (status),

INDEX idx\_date (payment\_date)

);



## **12.Reviews Table**

CREATE TABLE Reviews (

review\_id INT PRIMARY KEY AUTO\_INCREMENT,

contract\_id INT,

reviewer\_id INT,

reviewee\_id INT,

rating DECIMAL(3,2),

comment TEXT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

is\_public BOOLEAN DEFAULT TRUE,

FOREIGN KEY (contract\_id) REFERENCES Contracts(contract\_id),

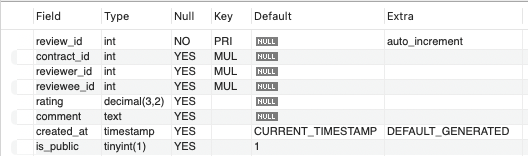
FOREIGN KEY (reviewer\_id) REFERENCES Users(user\_id),

FOREIGN KEY (reviewee\_id) REFERENCES Users(user\_id),

UNIQUE KEY unique\_review (contract\_id, reviewer\_id, reviewee\_id),

INDEX idx\_reviewee\_rating (reviewee\_id, rating)

);



## **13.Messages Table**

CREATE TABLE Messages (

message\_id INT PRIMARY KEY AUTO\_INCREMENT,

sender\_id INT NOT NULL,

receiver\_id INT NOT NULL,

contract\_id INT NULL,

message\_text TEXT NOT NULL,

sent\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

read\_at TIMESTAMP NULL,

is\_deleted BOOLEAN DEFAULT FALSE,

FOREIGN KEY (sender\_id) REFERENCES Users(user\_id),

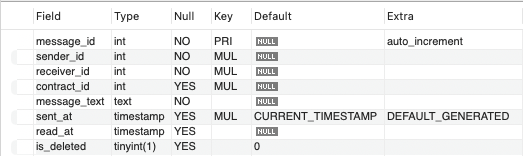
FOREIGN KEY (receiver\_id) REFERENCES Users(user\_id),

FOREIGN KEY (contract\_id) REFERENCES Contracts(contract\_id) ON DELETE SET NULL,

INDEX idx\_sender\_receiver (sender\_id, receiver\_id),

INDEX idx\_sent\_at (sent\_at)

);



## **14.Notifications Table**

CREATE TABLE Notifications (

notification\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

notification\_type ENUM('message', 'proposal', 'contract', 'payment', 'review', 'system') NOT NULL,

reference\_id INT NULL, *-- ID of related entity (message\_id, proposal\_id, etc.)*

message TEXT NOT NULL,

is\_read BOOLEAN DEFAULT FALSE,

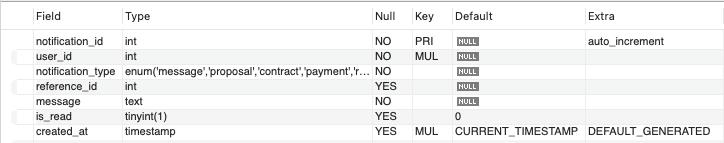
created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE,

INDEX idx\_user\_read (user\_id, is\_read),

INDEX idx\_created (created\_at)

);



**Queries to insert data into tables**

**1. Users Table**

INSERT INTO Users (email, password\_hash, first\_name, last\_name, country\_code, user\_type) VALUES

('ravi.kumar@metgig.com', '$2y$10$randomstring1', 'Ravi', 'Kumar', 'IN', 'freelancer'),

('priya.sharma@metgig.com', '$2y$10$anotherrandom2', 'Priya', 'Sharma', 'IN', 'client'),

('amit.patel@metgig.com', '$2y$10$yetanother3', 'Amit', 'Patel', 'IN', 'freelancer'),

('sneha.verma@metgig.com', '$2y$10$random4again', 'Sneha', 'Verma', 'IN', 'client'),

('karan.singh@metgig.com', '$2y$10$uniquehash5', 'Karan', 'Singh', 'IN', 'freelancer'),

('deepika.gupta@metgig.com', '$2y$10$securepass6', 'Deepika', 'Gupta', 'IN', 'client'),

('vikas.yadav@metgig.com', '$2y$10$strongone7', 'Vikas', 'Yadav', 'IN', 'freelancer'),

('anjali.rai@metgig.com', '$2y$10$complex8key', 'Anjali', 'Rai', 'IN', 'client'),

('suresh.pillai@metgig.com', '$2y$10$alphanumeric9', 'Suresh', 'Pillai', 'IN', 'freelancer'),

('nisha.joshi@metgig.com', '$2y$10$randommix10', 'Nisha', 'Joshi', 'IN', 'admin');

**2. FreelancerProfiles Table**

INSERT INTO FreelancerProfiles (user\_id, bio, hourly\_rate, title, experience\_level) VALUES

(1, 'Experienced web developer with a passion for React.', 25.50, 'React Developer', 'expert'),

(3, 'Creative graphic designer specializing in branding.', 32.75, 'Graphic Designer', 'intermediate'),

(5, 'Skilled content writer for blogs and articles.', 18.00, 'Content Writer', 'entry'),

(7, 'Expert in data analysis and visualization.', 38.20, 'Data Analyst', 'expert'),

(9, 'Mobile app developer for Android and iOS.', 41.99, 'Mobile App Developer', 'intermediate'),

(1, 'Also proficient in Node.js and backend development.', 29.00, 'Full-stack Developer', 'expert'),

(3, 'Has designed logos and marketing materials for various startups.', 31.50, 'Brand Identity Designer', 'intermediate'),

(5, 'Can write engaging and SEO-friendly content.', 23.10, 'SEO Content Specialist', 'intermediate'),

(7, 'Uses Python and SQL for data manipulation.', 36.45, 'Business Intelligence Analyst', 'intermediate'),

(9, 'Familiar with Flutter and React Native frameworks.', 43.60, 'Cross-platform App Developer', 'expert');

**3. ClientProfiles Table**

INSERT INTO ClientProfiles (user\_id, company\_name, company\_website, company\_description, company\_size, industry, payment\_verified) VALUES

(2, 'Tech Solutions India', 'www.techsolutions.metgig.com', 'Leading IT consulting firm.', 'large', 'Information Technology', TRUE),

(4, 'Creative Designs Studio', 'www.creativedesigns.metgig.com', 'Boutique design agency.', 'small', 'Design', FALSE),

(6, 'Global Marketing Ltd', 'www.globalmarketing.metgig.com', 'International marketing agency.', 'medium', 'Marketing', TRUE),

(8, 'Software Innovations', 'www.softwareinnovations.metgig.com', 'Innovative software development company.', 'medium', 'Software Development', TRUE),

(10, 'Alpha Corp', 'www.alphacorp.metgig.com', 'Diversified conglomerate.', 'large', 'Various', TRUE),

(2, 'E-Commerce Ventures', 'www.ecomventures.metgig.com', 'Online retail platform.', 'medium', 'E-commerce', TRUE),

(4, 'Artistic Minds', NULL, 'Freelance art collective.', 'individual', 'Arts and Crafts', FALSE),

(6, 'Digital Reach Agency', 'www.digitalreach.metgig.com', 'Specialized in digital marketing.', 'small', 'Marketing', TRUE),

(8, 'Web Development Hub', 'www.webdevhub.metgig.com', 'Focuses on custom web applications.', 'small', 'Web Development', TRUE),

(10, 'Beta Industries', NULL, 'Manufacturing and distribution.', 'large', 'Manufacturing', TRUE);

4**. Skills Table**

INSERT INTO Skills (skill\_name, category\_id, popularity) VALUES

('Python', 1, 92),

('UI/UX Design', 2, 88),

('Technical Writing', 3, 75),

('Machine Learning', 4, 97),

('iOS Development', 5, 85),

('Node.js', 1, 95),

('Motion Graphics', 2, 79),

('Copywriting', 3, 81),

('Data Mining', 4, 90),

('React Native', 5, 83);

5. **Categories Table (No changes needed as they are general categories)**

INSERT INTO Categories (category\_id, category\_name) VALUES

(11, 'Customer Service'),

(12, 'Legal'),

(13, 'Architecture & Engineering'),

(14, 'Education & Training'),

(15, 'Human Resources'),

(16, 'Sales'),

(17, 'Accounting'),

(18, 'Translation'),

(19, 'Video & Animation'),

(20, 'Photography');

6. **Projects Table**

INSERT INTO Projects (client\_id, title, description, category\_id, is\_fixed\_price, budget\_min, budget\_max, duration, experience\_level, status) VALUES

(2, 'Develop a Python Script for Data Processing', 'Need a Python script to automate data cleaning and processing.', 1, TRUE, 800.00, 1500.00, 'one\_week\_to\_month', 'intermediate', 'open'),

(4, 'Redesign Mobile App UI', 'Looking for a UI/UX designer to improve the user interface of our mobile application.', 2, TRUE, 1200.00, 2500.00, 'one\_to\_three\_months', 'intermediate', 'open'),

(6, 'Write a White Paper on Cloud Security', 'Seeking a technical writer to create a comprehensive white paper on cloud security best practices.', 3, TRUE, 1000.00, 2000.00, 'one\_month', 'expert', 'open'),

(8, 'Build a Machine Learning Model for Recommendation', 'Looking for a machine learning expert to build a recommendation engine.', 4, FALSE, 40.00, 80.00, 'three\_to\_six\_months', 'expert', 'open'),

(10, 'Develop a Native iOS App with Swift', 'Need an iOS developer to build a native application using Swift.', 5, FALSE, 35.00, 70.00, 'three\_to\_six\_months', 'intermediate', 'open'),

(2, 'Create Explainer Videos for Our Product', 'Looking for a video editor to create short explainer videos.', 19, TRUE, 500.00, 1200.00, 'one\_week\_to\_month', 'entry', 'open'),

(4, 'Design Social Media Posts for a Campaign', 'Need a designer to create engaging social media visuals for an upcoming campaign.', 2, TRUE, 300.00, 600.00, 'less\_than\_week', 'intermediate', 'open'),

(6, 'Translate Marketing Materials to Tamil', 'Seeking a translator for our marketing brochures and website content.', 18, TRUE, 400.00, 900.00, 'one\_week\_to\_month', 'intermediate', 'open'),

(8, 'Implement Data Mining Techniques for Customer Insights', 'Looking for a data scientist to apply data mining techniques to extract customer insights.', 4, FALSE, 50.00, 90.00, 'one\_to\_three\_months', 'expert', 'open'),

(10, 'Develop a React Native Mobile App', 'Need a developer to build a cross-platform mobile app using React Native.', 5, FALSE, 45.00, 85.00, 'three\_to\_six\_months', 'intermediate', 'open');

7. **ProjectSkills Table**

INSERT INTO ProjectSkills (project\_id, skill\_id, importance) VALUES

(11, 11, 'required'), (11, 6, 'preferred'),

(12, 12, 'required'), (12, 2, 'required'),

(13, 13, 'required'), (13, 3, 'required'),

(14, 14, 'required'), (14, 4, 'required'),

(15, 15, 'required'), (15, 5, 'required'),

(16, 19, 'required'), (16, 2, 'nice\_to\_have'),

(17, 2, 'required'), (17, 7, 'required'),

(18, 18, 'required'), (18, 3, 'preferred'),

(19, 14, 'required'), (19, 9, 'required'),

(20, 15, 'required'), (20, 10, 'required');

8. **Proposals Table**

INSERT INTO Proposals (project\_id, freelancer\_id, cover\_letter, bid\_amount, estimated\_duration, status) VALUES

(11, 1, 'I have experience in Python scripting for data tasks.', 1200.00, 15, 'pending'),

(12, 3, 'My UI/UX designs are user-centric and modern.', 2000.00, 45, 'pending'),

(13, 5, 'I can write a well-researched white paper on cloud security.', 1800.00, 30, 'pending'),

(14, 7, 'I have built several recommendation systems using machine learning.', 70.00, 90, 'pending'),

(15, 9, 'I am proficient in Swift and iOS development.', 65.00, 120, 'pending'),

(11, 9, 'I also have some experience with data analysis using Python.', 1100.00, 20, 'pending'),

(12, 5, 'I can also create wireframes and prototypes for mobile apps.', 1800.00, 50, 'pending'),

(13, 1, 'My technical writing skills are well-suited for this task.', 1600.00, 35, 'pending'),

(14, 3, 'I have a strong understanding of various machine learning algorithms.', 75.00, 100, 'pending'),

(15, 1, 'I have also worked on cross-platform app development.', 60.00, 130, 'pending');

9. **PortfolioItems Table**

INSERT INTO PortfolioItems (user\_id, title, description, project\_url, image\_url, completion\_date, is\_featured) VALUES

(1, 'Data Processing Script in Python', 'Automated data cleaning and transformation using Python.', 'www.github.com/datascripts', 'python\_script\_thumb.png', '2025-03-01', TRUE),

(3, 'Mobile App UI Redesign Concept', 'Conceptual redesign of a popular mobile app interface.', 'www.behance.net/mobileui', 'mobile\_ui\_redesign.jpg', '2025-02-10', TRUE),

(5, 'White Paper on Network Security', 'Authored a detailed white paper on network security protocols.', NULL, 'network\_security\_wp.pdf', '2025-01-25', FALSE),

(7, 'Movie Recommendation System', 'Developed a recommendation system for movies using collaborative filtering.', 'www.github.com/recommendersys', 'recommendation\_system.png', '2024-12-20', TRUE),

(9, 'Task Management iOS App', 'Developed a native iOS application for managing daily tasks.', 'www.github.com/iostasks', 'ios\_task\_app.png', '2024-11-15', FALSE),

(1, 'Web Scraping Tool in Python', 'Created a tool to scrape data from various websites.', 'www.github.com/webscraper', 'web\_scraping\_tool.png', '2025-04-05', TRUE),

(3, 'E-commerce Website Mockups', 'Designed mockups for a new e-commerce platform.', 'www.figma.com/ecommerce', 'ecommerce\_mockups.png', '2025-03-20', FALSE),

(5, 'Blog Posts on Content Marketing', 'Wrote a series of blog posts on various aspects of content marketing.', 'www.exampleblog.com/marketing', 'content\_marketing\_thumb.jpg', '2025-02-01', FALSE),

(7, 'Customer Churn Prediction Model (Presentation)', 'Created a presentation summarizing a customer churn prediction model.', NULL, 'churn\_prediction\_slides.pdf', '2025-01-10', FALSE),

(9, 'Cross-Platform To-Do App', 'Developed a to-do application using React Native.', 'www.github.com/reactnativedo', 'react\_native\_todo.png', '2024-10-25', TRUE);

10. **Contracts Table**

INSERT INTO Contracts (project\_id, client\_id, freelancer\_id, proposal\_id, title, description, contract\_type, amount, start\_date, end\_date, status) VALUES

(11, 2, 3, 20, 'React Native Mobile App Development', 'Developing a cross-platform mobile app using React Native.', 'hourly', 80.00, '2025-05-18', '2025-08-18', 'active'),

(12, 4, 5, 17, 'Social Media Graphics for New Product', 'Creating engaging social media visuals for a product launch.', 'fixed', 550.00, '2025-05-22', '2025-05-29', 'active'),

(13, 6, 1, 18, 'Hindi Translation of Website Content', 'Translating all website content accurately to Hindi.', 'fixed', 950.00, '2025-05-26', '2025-06-16', 'active'),

(14, 8, 9, 19, 'Data Analysis for Marketing Campaign', 'Analyzing campaign data to provide insights and recommendations.', 'hourly', 90.00, '2025-05-30', '2025-08-30', 'active'),

(15, 10, 7, 14, 'Machine Learning Model for Customer Retention', 'Building a model to predict and improve customer retention.', 'hourly', 110.00, '2025-06-03', '2025-09-03', 'active'),

(16, 2, 9, 15, 'iOS App Development for Inventory Management', 'Developing a native iOS app to manage inventory.', 'hourly', 75.00, '2025-06-07', '2025-09-07', 'active'),

(17, 4, 1, 11, 'Python Script for Report Generation', 'Creating a Python script to automate weekly report generation.', 'fixed', 1400.00, '2025-06-11', '2025-06-25', 'active'),

(18, 6, 3, 12, 'UI/UX Improvement for Web Application', 'Improving the user interface and experience of the web application.', 'fixed', 2800.00, '2025-06-15', '2025-07-30', 'active'),

(19, 8, 5, 13, 'Technical Documentation for Software API', 'Writing clear and concise technical documentation for the software API.', 'fixed', 1600.00, '2025-06-19', '2025-07-19', 'active'),

(20, 10, 7, 14, 'Data Mining for Sales Trend Analysis', 'Applying data mining techniques to identify key sales trends.', 'hourly', 85.00, '2025-06-23', '2025-09-23', 'active');

11**. Payments Table**

INSERT INTO Payments (contract\_id, amount, status, payment\_method, transaction\_id, platform\_fee, freelancer\_amount) VALUES

(21, 4000.00, 'completed', 'Credit Card', 'txn12345', 200.00, 3800.00),

(22, 550.00, 'completed', 'UPI', 'upi67890', 27.50, 522.50),

(23, 950.00, 'completed', 'Net Banking', 'netbank111', 47.50, 902.50),

(24, 2700.00, 'completed', 'Credit Card', 'txn22233', 135.00, 2565.00),

(25, 5500.00, 'completed', 'UPI', 'upi44455', 275.00, 5225.00),

(26, 3750.00, 'completed', 'Net Banking', 'netbank666', 187.50, 3562.50),

(27, 700.00, 'completed', 'Credit Card', 'txn77788', 35.00, 665.00),

(28, 1400.00, 'completed', 'UPI', 'upi99900', 70.00, 1330.00),

(29, 4800.00, 'completed', 'Net Banking', 'netbankabc', 240.00, 4560.00),

(30, 2550.00, 'completed', 'Credit Card', 'txndefg', 127.50, 2422.50);

12. **Reviews Table**

INSERT INTO Reviews (contract\_id, reviewer\_id, reviewee\_id, rating, comment) VALUES

(21, 2, 1, 4.8, 'Excellent work on the React Native app!'),

(22, 4, 3, 4.5, 'Great social media graphics, very creative.'),

(23, 6, 5, 4.9, 'Accurate and professional Hindi translation.'),

(24, 8, 9, 4.7, 'Provided valuable insights from the marketing data.'),

(25, 10, 7, 5.0, 'The customer retention model is very effective.'),

(26, 10, 9, 4.6, 'Good job on the iOS inventory management app.'),

(27, 2, 1, 4.3, 'The Python script is working perfectly.'),

(28, 4, 3, 4.8, 'Significant improvement in the web application UI/UX.'),

(29, 6, 5, 4.7, 'Clear and well-written API documentation.'),

(30, 8, 7, 4.9, 'Very helpful analysis of the sales trends.');

13. **Messages Table**

INSERT INTO Messages (sender\_id, receiver\_id, contract\_id, message\_text) VALUES

(1, 2, 21, 'Hi, the first milestone of the React Native app is complete.'),

(3, 4, 22, 'Here are the initial social media graphic designs for your review.'),

(5, 6, 23, 'The Hindi translation of the website content is now finished.'),

(9, 8, 24, 'Attached is the report on the marketing campaign data analysis.'),

(7, 10, 25, 'The machine learning model for customer retention is deployed.'),

(9, 10, 26, 'The iOS inventory management app is ready for testing.'),

(1, 2, 27, 'The Python script for report generation has been implemented.'),

(3, 4, 28, 'Please take a look at the updated UI/UX of the web application.'),

(5, 6, 29, 'I have completed the technical documentation for the API.'),

(7, 8, 30, 'Here is the analysis of the recent sales trends.');

14. **Notifications Table**

INSERT INTO Notifications (user\_id, notification\_type, reference\_id, message) VALUES

(2, 'contract', 21, 'New contract started: React Native Mobile App Development'),

(4, 'proposal', 17, 'New proposal received for Social Media Graphics'),

(6, 'payment', 1, 'Payment completed for contract ID 21'),

(1, 'message', 1, 'New message received regarding React Native app'),

(10, 'review', 1, 'New review received for your work on contract ID 25'),

(9, 'contract', 26, 'New contract started: iOS App Development'),

(2, 'proposal', 11, 'Your proposal was accepted for Python Script'),

(4, 'message', 2, 'Feedback on the social media graphic designs'),

(6, 'payment', 2, 'Payment completed for contract ID 22'),

(7, 'contract', 30, 'New contract started: Data Mining for Sales Trend Analysis');

**Basic SQL Queries**

**Additional Basic Operations (Filtering, Ordering, Limiting)**

1. Select freelancers with an hourly rate greater than $30, ordered by hourly rate descending.

SELECT u.first\_name, u.last\_name, fp.hourly\_rate

FROM Users u

JOIN FreelancerProfiles fp ON u.user\_id = fp.user\_id

WHERE fp.hourly\_rate > 30

ORDER BY fp.hourly\_rate DESC;

2. Select the top 5 most recent projects that are currently open.

SELECT title, created\_at

FROM Projects

WHERE status = 'open'

ORDER BY created\_at DESC

LIMIT 5;

DML Operations (INSERT, UPDATE, DELETE)

3. Insert a new skill into the Skills table.

INSERT INTO Skills (skill\_name, category\_id) VALUES ('Video Editing', 19);

4. Update the status of a specific project to 'in\_progress'.

UPDATE Projects

SET status = 'in\_progress'

WHERE project\_id = 11;

5. Delete a specific proposal based on its ID.

DELETE FROM Proposals

WHERE proposal\_id = 20;

Set Operations (UNION, INTERSECT, EXCEPT - Note: INTERSECT and EXCEPT syntax may vary by database system)

6. Find all unique user IDs that are either freelancers or clients (UNION).

SELECT user\_id FROM FreelancerProfiles

UNION

SELECT user\_id FROM ClientProfiles;

7. Find user IDs that have both a freelancer profile and have submitted a proposal (Illustrative - INTERSECT not directly supported in all SQL).

-- This can be achieved with a join and grouping:

SELECT fp.user\_id

FROM FreelancerProfiles fp

INNER JOIN Proposals p ON fp.user\_id = p.freelancer\_id

GROUP BY fp.user\_id;

8. Find user IDs that are freelancers but have not submitted any proposals

SELECT user\_id

FROM FreelancerProfiles

WHERE user\_id NOT IN (SELECT freelancer\_id FROM Proposals);

Aggregate Functions (COUNT, SUM, AVG, MIN, MAX)

9. Count the total number of active users.

SELECT COUNT(\*) AS total\_active\_users

FROM Users

WHERE account\_status = 'active';

10. Calculate the average hourly rate of all freelancers.

SELECT AVG(hourly\_rate) AS average\_hourly\_rate

FROM FreelancerProfiles;

11. Find the maximum budget of any open project.

SELECT MAX(budget\_max) AS max\_project\_budget

FROM Projects

WHERE status = 'open';

12. Calculate the total amount paid out for completed contracts.

SELECT SUM(p.amount) AS total\_payout

FROM Payments p

JOIN Contracts c ON p.contract\_id = c.contract\_id

WHERE p.status = 'completed';

Nested Sub-queries

13. Select clients who have posted at least one project with a budget over $5000.

SELECT u.first\_name, u.last\_name

FROM Users u

WHERE u.user\_id IN (SELECT client\_id FROM Projects WHERE budget\_max > 5000);

14. Find freelancers who have a rating higher than the average rating of all freelancers.

SELECT u.first\_name, u.last\_name, fp.rating

FROM Users u

JOIN FreelancerProfiles fp ON u.user\_id = fp.user\_id

WHERE fp.rating > (SELECT AVG(rating) FROM FreelancerProfiles);

15. Select projects that require a skill with a popularity greater than 90.

SELECT p.title

FROM Projects p

WHERE p.project\_id IN (

SELECT ps.project\_id

FROM ProjectSkills ps

JOIN Skills s ON ps.skill\_id = s.skill\_id

WHERE s.popularity > 90

);

Joins (INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN - FULL OUTER JOIN syntax may vary)

16. Get the first name, last name of users along with their freelancer profile details (INNER JOIN).

SELECT u.first\_name, u.last\_name, fp.bio, fp.hourly\_rate

FROM Users u

INNER JOIN FreelancerProfiles fp ON u.user\_id = fp.user\_id;

17. List all users and their client profile information, if they have one (LEFT JOIN).

SELECT u.first\_name, u.last\_name, cp.company\_name

FROM Users u

LEFT JOIN ClientProfiles cp ON u.user\_id = cp.user\_id;

18. List all freelancers and the proposals they have submitted (LEFT JOIN).

SELECT u.first\_name, u.last\_name, p.bid\_amount

FROM Users u

LEFT JOIN Proposals p ON u.user\_id = p.freelancer\_id;

Views

19. Create a view of active freelancers with their hourly rates and ratings.

CREATE VIEW ActiveFreelancerRates AS

SELECT u.first\_name, u.last\_name, fp.hourly\_rate, fp.rating

FROM Users u

JOIN FreelancerProfiles fp ON u.user\_id = fp.user\_id

WHERE u.account\_status = 'active';

20. Create a view of open projects with their titles and the name of the client who posted them.

CREATE VIEW OpenProjectsWithClient AS

SELECT p.title AS project\_title, u.first\_name AS client\_first\_name, u.last\_name AS client\_last\_name

FROM Projects p

JOIN Users u ON p.client\_id = u.user\_id

WHERE p.status = 'open';

**VII. Project demonstration**

The development of *The Met Gig* utilized a wide range of tools and technologies to ensure a scalable, responsive, and dynamic user experience.

**Visual Studio Code (VS Code)** served as the primary code editor, offering an efficient environment for writing and organizing HTML, CSS, and JavaScript code.

**HTML5** was used to structure the content of the web application, while **CSS3** handled the layout and styling, including responsive design through Flexbox, Grid, and media queries.

**Vanilla JavaScript** was used for client-side interactivity and DOM manipulation.

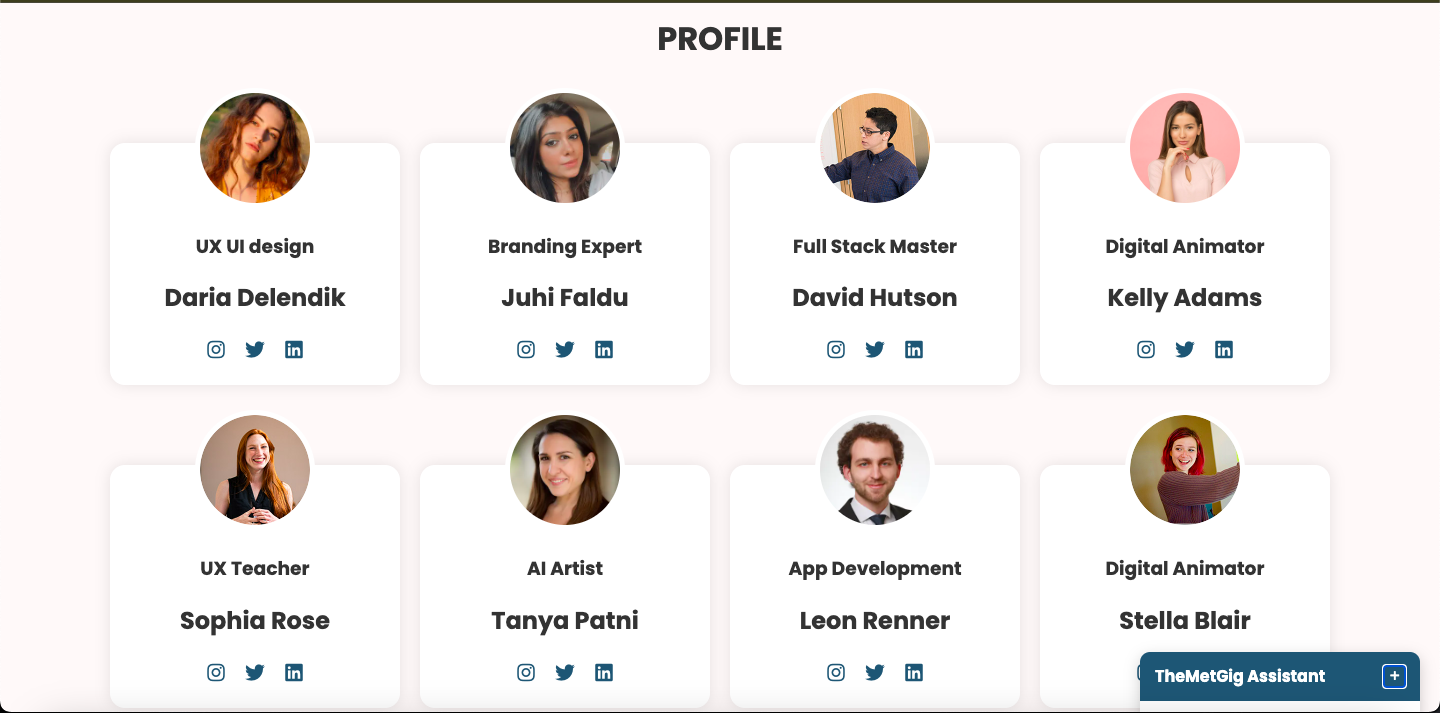
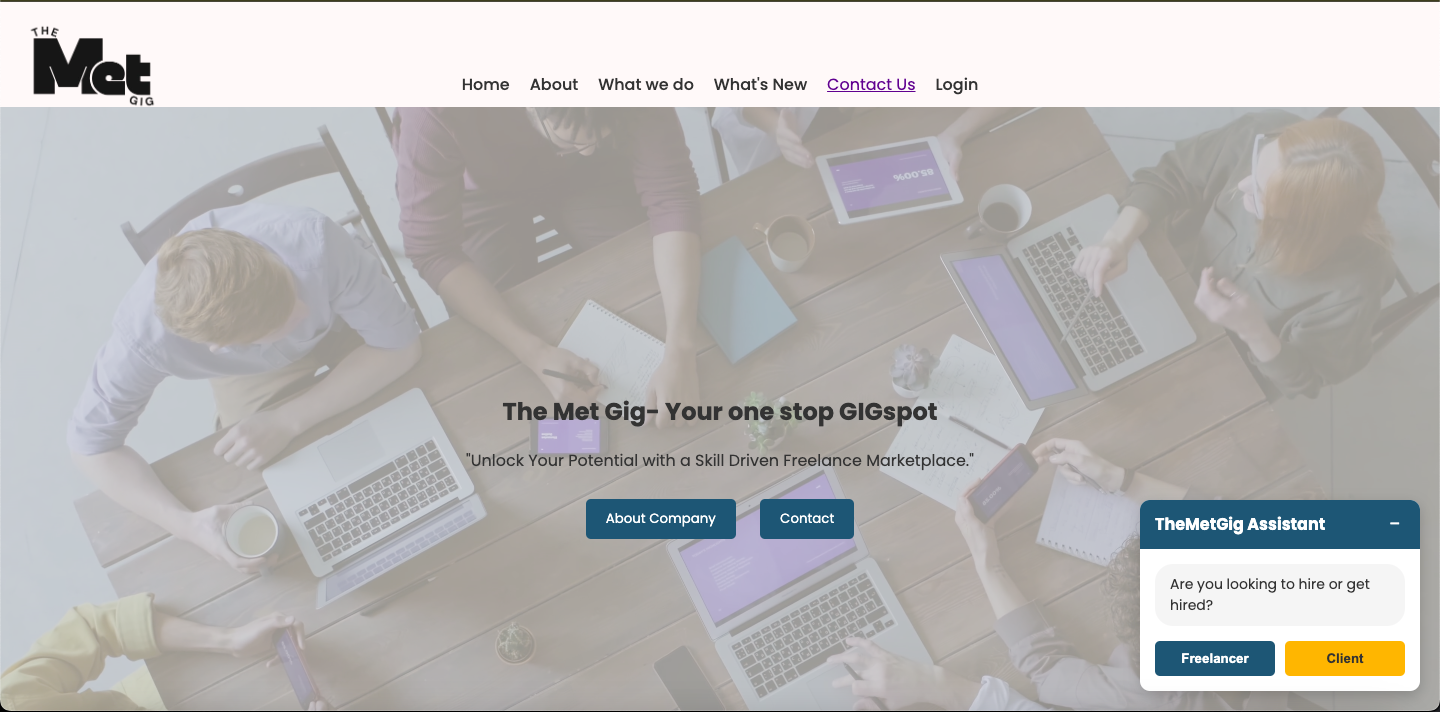
For UI/UX design and prototyping, tools like **Canva** was used to create wireframes and visual mockups of the application interface.

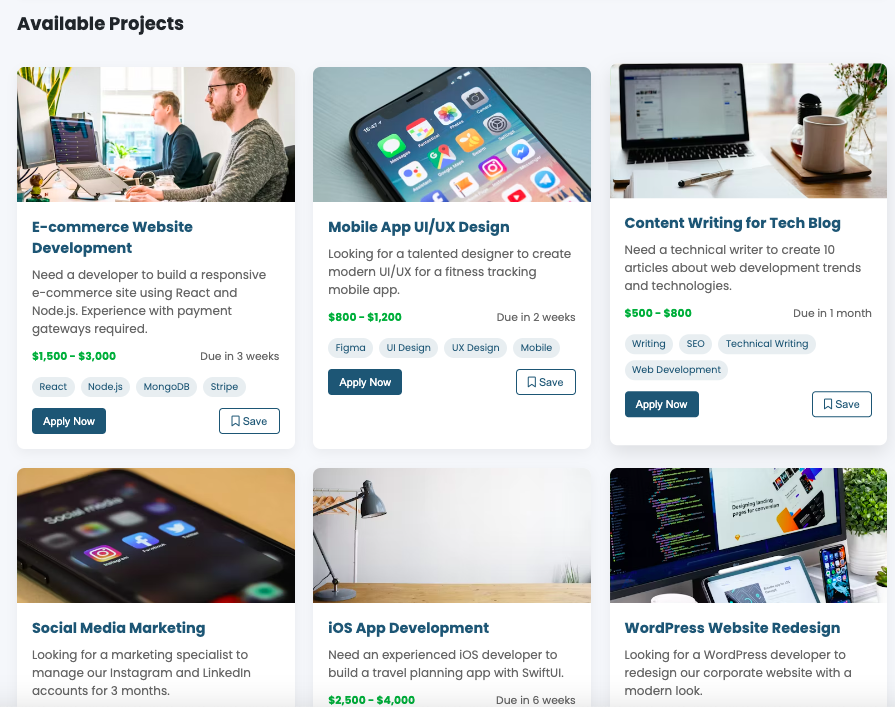
On the data side, **MySQL** was used to model the database structure for storing project, user, and bidding information.

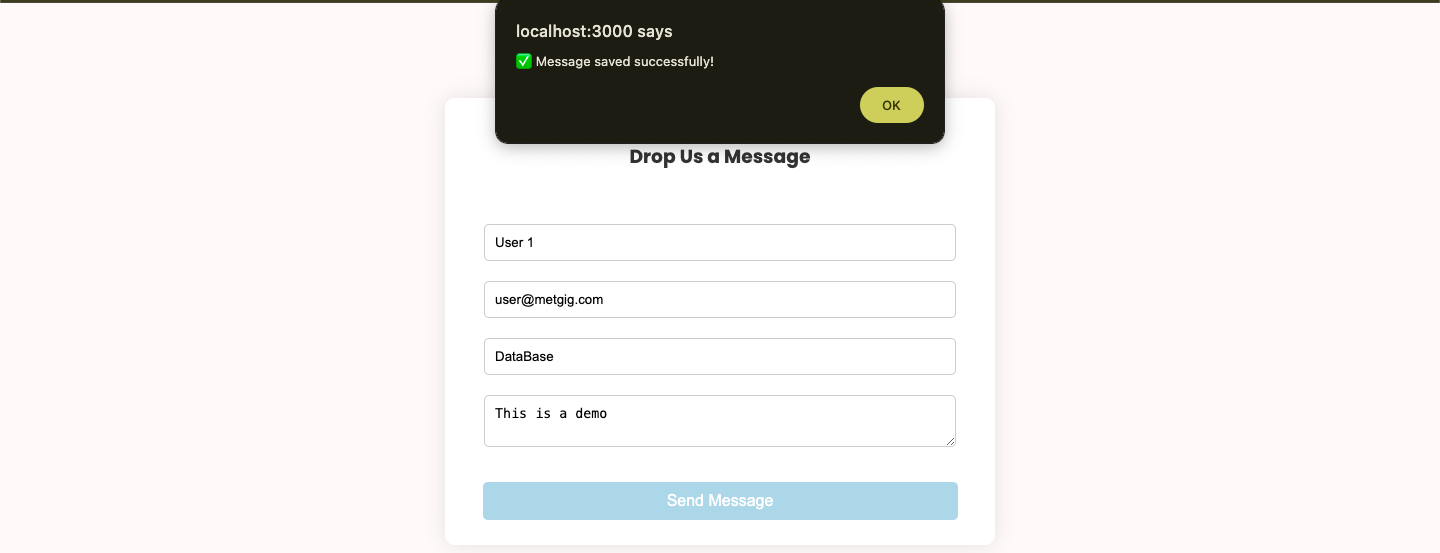
**MySQL Workbench** was leveraged for writing and testing SQL queries, as well as for managing database schemas.

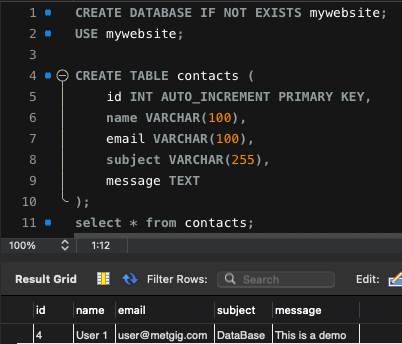
**Lucidchart** was used to design and visualize the Entity-Relationship (ER) diagrams, helping map out database relationships clearly and accurately.

Though the current version is front-end focused, future development plans include integrating a backend using **Node.js** and **Express.js** to handle authentication, database operations, and real-time features such as messaging.

Screenshot and Description of the Demonstration of project



**Database Integration**



**VIII. Self-Learning beyond the classroom**

### **1. Advanced SQL & Database Optimization**

* **Indexing & Query Optimization**: Learned to create **indexes** on frequently queried columns (e.g., user\_id, skill\_id) to accelerate search operations.
* **Stored Procedures & Triggers**: Designed stored procedures to automate repetitive tasks (e.g., calculating total earnings for freelancers) and triggers to enforce constraints, such as updating project status upon contract completion.
* **Transaction Management**: Implemented **ACID properties** to ensure data consistency during critical operations like bid submissions and payment processing.

### **2. Security Best Practices**

* **Parameterized Queries**: Adopted parameterized SQL queries to mitigate **SQL injection attacks**.
* **Data Encryption**: Researched and applied basic encryption techniques for sensitive fields like password\_hash using bcrypt.
* **Session Management**: Explored token-based authentication (JWT) for future backend integration, despite initial reliance on frontend sessions.

### **3. Tool Proficiency**

* **MySQL Workbench**: Mastered advanced features like schema synchronization, query profiling, and performance tuning.
* **Lucidchart**: Created dynamic ER diagrams to visualize complex relationships (e.g., M:N mappings for skills and projects).
* **Git/GitHub**: Learned branching strategies, resolved merge conflicts, and maintained a structured commit history for collaborative development.

### **4. Agile & Project Management**

* **Sprint Planning**: Used **Trello** to break down tasks into sprints, prioritizing features like bidding mechanics and role-based dashboards.
* **Mock APIs**: Built mock backend endpoints with **JSON Server** to parallelize frontend and database development.

### **5. Frontend-Database Integration**

* **Dynamic Data Binding**: Implemented vanilla JavaScript to fetch and display real-time data from MySQL (e.g., populating project listings).
* **Responsive Design**: Applied **CSS Grid** and **media queries** to ensure seamless UI experiences across devices, despite limited prior experience.

### **6. Problem-Solving & Debugging**

* **Orphaned Records**: Diagnosed and resolved orphaned proposal records by enforcing **ON DELETE CASCADE** in foreign key constraints.
* **Query Bottlenecks**: Used **EXPLAIN ANALYZE** in MySQL to identify slow queries and optimize them through indexing.

### **7. Collaboration & Documentation**

* **Notion for Documentation**: Maintained detailed logs of schema changes, API contracts, and user stories to ensure team alignment.
* **Peer Code Reviews**: Conducted regular code reviews to share knowledge and maintain code quality, enhancing our understanding of SQL and JavaScript.

**IX. Learning from the Project**

### **Technical Learnings:**

1. **Full-Stack Development:**
   * **Frontend:** Gained experience using **JavaScript** to build interactive UIs and dashboards for different users (freelancers and clients).
   * **Backend:** Implemented **Node.js/Flask** server for handling authentication, project posting, bidding, and messaging.
   * **Database Management:** Designed and managed SQL collections for users, projects, bids, and transactions.
2. **Authentication & Authorization:**
   * Implemented **secure login/registration systems**.
   * Used **JWT or sessions** to maintain user sessions.
   * Separated access roles for freelancers and clients.
3. **File Upload and Storage:**
   * Enabled **portfolio uploads** and resume handling using multer or similar libraries.
4. **Search and Filtering:**
   * Added smart **search features** using text queries and filters (e.g., by skills, hourly rate, etc.).

### 

### 

### **Conceptual Learnings:**

1. **Platform Design Thinking:**
   * Understood the flow of a **marketplace ecosystem**: how clients and freelancers interact, and how trust is built through bidding, ratings, and reviews.
2. **Scalability and Modularity:**
   * Learned how to design systems that can scale (modular folder structure, API endpoints, reusable components).
3. **Security Practices:**
   * Focused on **data validation**, **encryption**, and **preventing vulnerabilities** like XSS, SQL injection.

### **Project Management Learnings:**

1. **Requirement Gathering & Documentation:**
   * Created **user stories, wireframes, and ER diagrams** before development.
   * Prioritized features using a **MVP (Minimum Viable Product)** mindset.
2. **Team Collaboration:**
   * Improved skills in **collaborative development,** assigning tasks using tools like**, Notion.**
3. **Debugging & Testing:**

Practiced thorough **testing of edge cases** (e.g., failed bids, missing profile info, unauthorized access).

**X. Challenges Faced**

1. **Database Relationship Complexity**:
   * Establishing **M:N relationships** (e.g., linking projects to multiple skills) required iterative revisions of the ER diagram to avoid redundancy.
2. **Time Zone Coordination**:
   * Team members worked across different schedules, delaying decision-making. Adopted **asynchronous communication** via Slack and shared documentation to mitigate this.
3. **Frontend-Backend Disconnect**:
   * Early frontend designs assumed backend endpoints that weren’t yet built, leading to rework. Later, we created **mock APIs** to parallelize development.
4. **Data Integrity Issues**:
   * Initially, deleting a user did not cascade to related proposals, causing orphaned records. Implemented **ON DELETE CASCADE** constraints after troubleshooting.
5. **Scope Creep**:
   * Features like "advanced matching algorithms" were deprioritized after realizing their computational complexity with our current stack.

**XI. Conclusion**

*The Met Gig* successfully demonstrates how a skill-driven freelance marketplace can bridge gaps between clients and freelancers. By prioritizing atomicity in database design, intuitive UI/UX, and secure transactions, we built a scalable foundation for future enhancements.

**Key Achievements**:

* Delivered a functional frontend prototype with role-based dashboards.
* Achieved 3NF and BCNF compliance in the database, ensuring data integrity.
* Implemented critical features like bidding, contracts, and reviews.

**Future Roadmap**:

* Develop AI-driven skill-matching using machine learning models.

This project not only honed our technical skills but also taught us the importance of adaptability, user feedback, and iterative development. It stands as a testament to our growth as developers and collaborators, ready to tackle real-world problems with innovative solutions.