SVKM's NMIMS

School of Technology Management & Engineering, Navi Mumbai

S.Y. 2023 - 24

Course: Database Management Systems

Project Report

Program	BTech AI&DS	
Semester	IV	
Name of the Project:	Freelance platform system	
Details of Project Members		
Batch	Roll No.	Name
Batch 2	A096	Onkar Mane
Batch 2	A097	Anushka Raspayle
Batch 1	A067	Navaneeth Krishnan
Date of Submission: 30/03/24		

Contribution of each project Members:

Roll No.	Name	Contribution
A096	Onkar Mane	Led frontend and backend coding efforts for seamless integration. Executed SQL queries for database management and optimization. Collaborated on database component development for enhanced functionality. Contributed to refining the relational model to meet project requirements. Played a role in editing and drafting the comprehensive project report. Established a GitHub repository to manage project versions and collaborate efficiently.
A097	Anushka Raspayle	Crafted a compelling storyline detailing the project's journey from inception to completion. Designed an intuitive ER diagram to visualize the database schema effectively. Developed a robust relational model to translate the ER diagram into a functional database. Identified key insights for the conclusion and highlighted challenges encountered during the project. Assisted in refining the report format, ensuring clarity and professionalism in presentation

A067	Navaneeth krishnan	Implemented normalization techniques for database optimization. Designed the ER model to capture the project's structural requirements effectively. Orchestrated backend operations, including table creation, attribute definition, and data input. Identified key insights for the conclusion and highlighted challenges encountered during the project. Assisted in refining the report format, ensuring clarity and professionalism in presentation

Github link of your project: https://github.com/OnkarSmane/Freelance-website

PROJECT REPORT

FREELANCE PLATFORM MANAGEMENT SYSTEM

By

Navaneeth Krishnan (A067)

Onkar Mane (A096)

Anushka Raspayle (A097)

COURSE: Database Management System

SY: 2023-24

Table of Contents

Storyline

In the grand narrative of modern business evolution, the rise of the gig economy stands as a pivotal chapter reshaping the way organizations engage talent. As companies navigate the intricacies of staying agile in a rapidly changing landscape, the Freelancer Management System (FMS) emerges as a beacon of efficiency and adaptability. At its core, the FMS isn't just a software solution; it represents a strategic imperative—a fundamental shift in how businesses approach talent acquisition, management, and deployment.

Within the bustling corridors of corporate headquarters and virtual workspaces alike, the need for specialized skills has never been more pronounced. From software development to content creation, companies seek expertise beyond the confines of their traditional workforce. Here, the FMS steps in as a linchpin, offering a centralized platform to effortlessly source, onboard, and manage freelance talent. Its importance lies not only in streamlining operations but in unlocking access to a diverse pool of skills precisely when needed, empowering businesses to stay ahead in an ever-evolving marketplace.

Yet, the allure of freelance talent comes with its own set of challenges—a mosaic of coordination hurdles, communication barriers, and compliance complexities. In this labyrinth of freelance management, the FMS emerges as a guiding light, offering a structured framework to navigate the maze. By facilitating seamless communication, tracking project progress, and ensuring compliance with regulations, the FMS becomes more than just a tool—it becomes a strategic ally in the pursuit of organizational excellence.

Zooming out from the microcosm of individual businesses, the broader market landscape reveals a tapestry of opportunities and challenges. Here, market research plays a pivotal role, illuminating the path forward amidst a sea of competitors and emerging trends. The Freelancer Management System Market research becomes a compass, guiding businesses through the labyrinth of industry dynamics, competitor strategies, and market trends. Armed with insights from this research, organizations can chart a course towards success, leveraging the power of the FMS to navigate the winds of change and emerge victorious in the ever-shifting tides of the gig economy.

Components of Database Design

Entities:

(Primary keys represented using "underline" and foreign keys using **"bold & highlighted"**)

- **Admin**: admin_id, admin_name, email, phone_number, user_type

- **Free**: freelancer_id, freelancer_name, **admin_id**, email, phone_number, date_of_birth,

- Client: client_id, client_name, email, phone_number, date_of_birth, **admin_id**,

- **Skill**: skill_id, skill_name, description

- **Freelanceskill: freelancer_id**, **skill_id**, description

- **Portfolio**: portfolio_id, **freelancer_id**, portfolio_name, portfolio_description,

- **Category**: category_id, category_name, category_description

- **Project**: project_id, **client_id, category_id**, project_name, project_description, budget, deadline

- **Proposal**: **proposal_id, freelancer_id, project_id**, proposal_details, proposal_date,

- **Contract**: contract_id, **project_id, freelancer_id, client_id**, contract_date, start_date, end_date

- **Payment**: payment_id, **freelancer_id, project_id,** amount, payment_date, payment_method,

- **Review**: review_id, client_id, freelancer_id, project_id, review_details, rating, review_date,

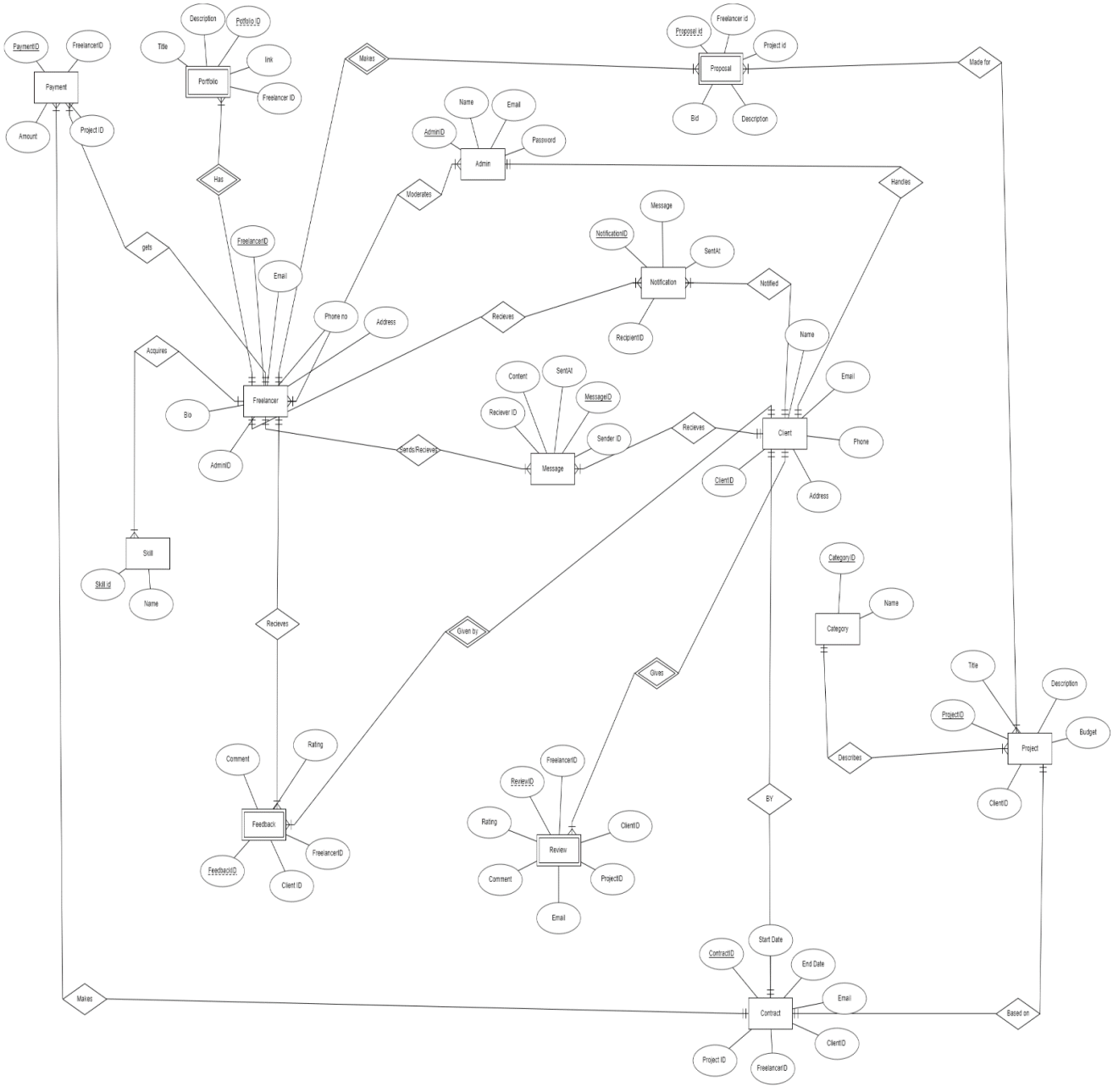
- **Message**: message_id, **sender_id, receiver_id**, message_details, sent_date,

- **Notification**: notification_id, **user_id**, notification_details, notification_date, is_read,

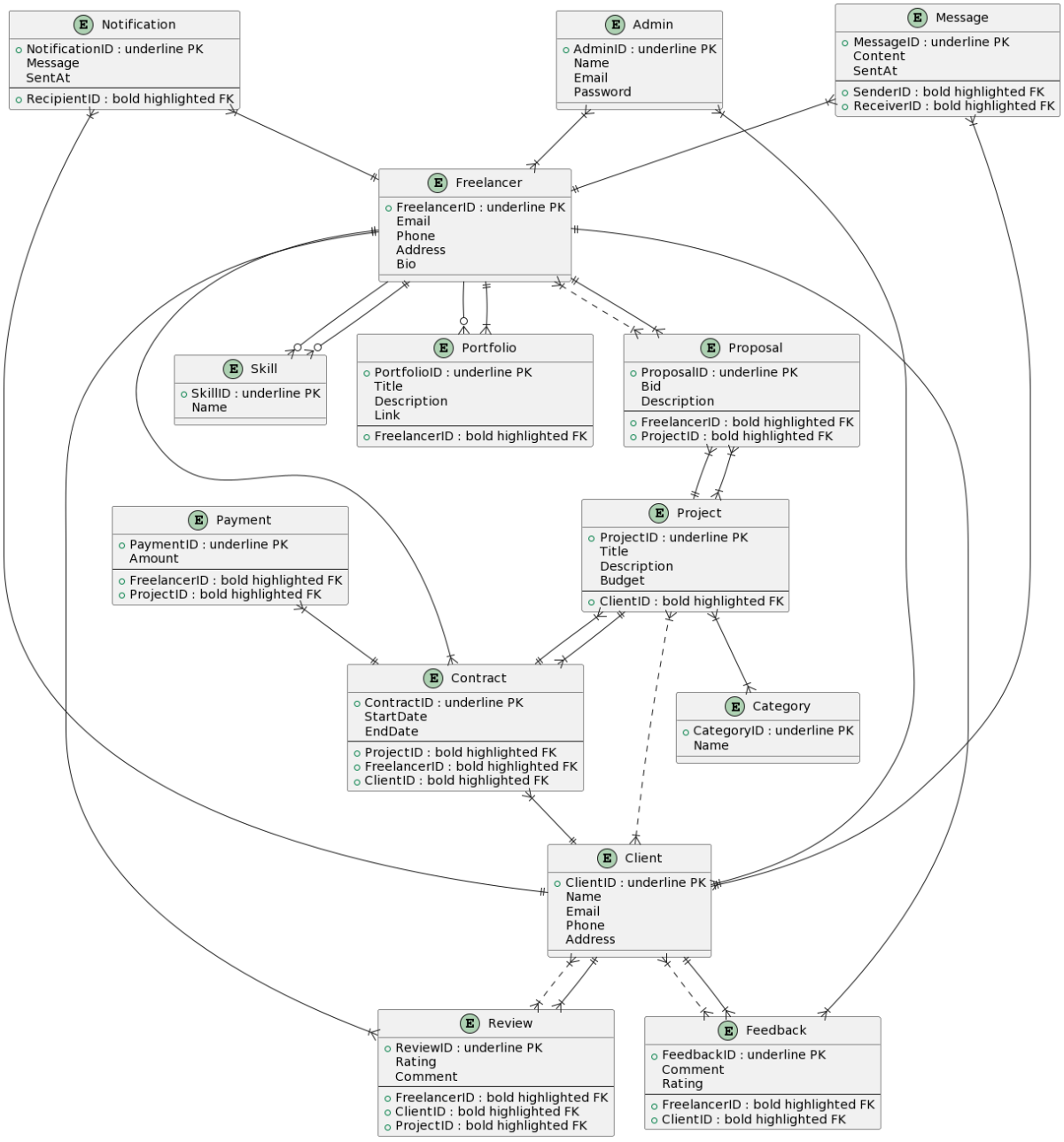
- Feedback: feedback_id, **user_id,** feedback_details, feedback_date

(Entity relationships and cardinalities depicted in the ER model and Relational model.)

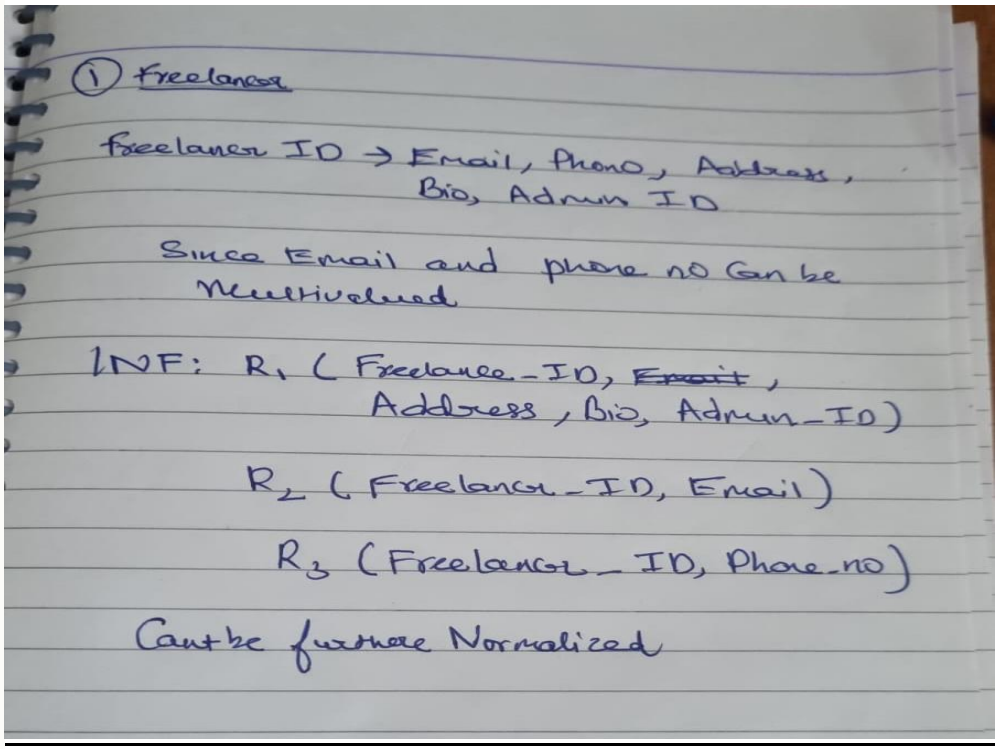
Entity Relationship Diagram



Relational Model



Normalization



① Freelancer

Freelancer ID → Email, PhoneNo, Address, Bio, Admin ID

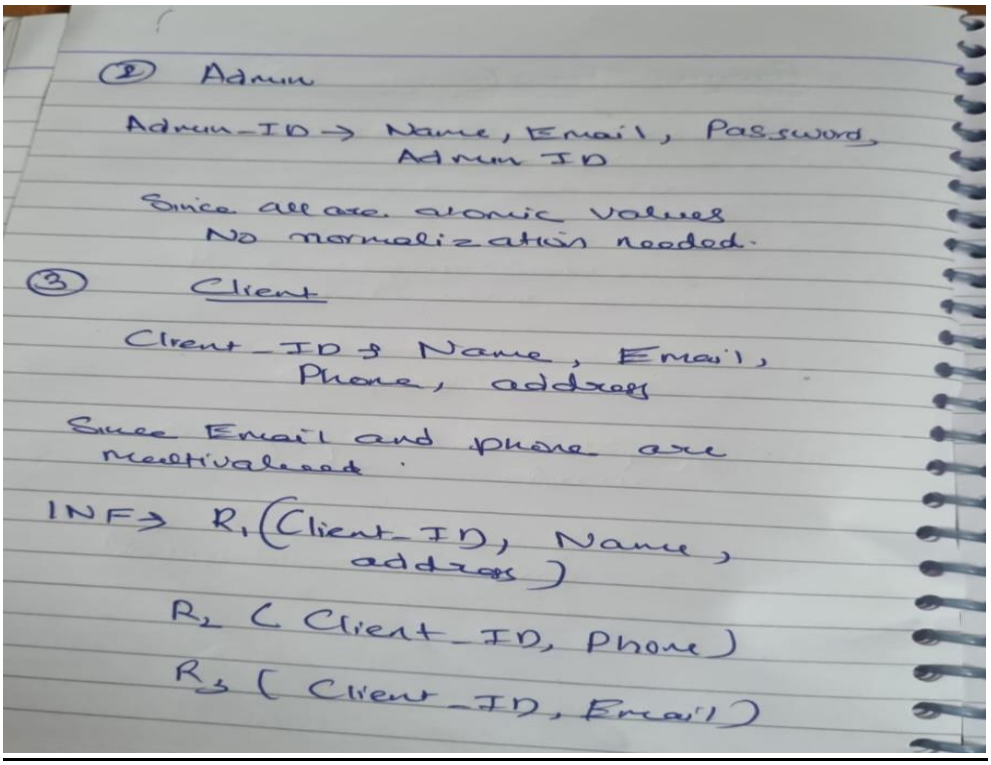
Since Email and phone no can be Multivalued

1NF: R_1 (Freelancer_ID, ~~Email~~, Address, Bio, Admin_ID)

R_2 (Freelancer_ID, Email)

R_3 (Freelancer_ID, Phone_no)

Can't be further Normalized



② Admin

Admin_ID → Name, Email, Password, Admin ID

Since all are atomic values
No normalization needed.

③ Client

Client_ID → Name, Email, Phone, address

Since Email and phone are Multivalued

INF → R_1 (Client_ID, Name, address)

R_2 (Client_ID, Phone)

R_3 (Client_ID, Email)

(7) Proposal

Proposal_ID → Proposal_ID, Freelancer_ID, Project_ID, Bid, Description

No normalization required.

(8) Contract

Contract_ID → Contract_ID, Start_Date, End_date, Email, Client_ID, Freelancer_ID, Project_ID.

here Email is associated with the Contract email formed by platform, ∴ Is Atomic.

No normalization Required.

Cant be further Normalized

(9) Portfolio

Portfolio_ID → Title, Description, Portfolio_ID, link, Freelancer_ID

here Title can be Multiple for a freelancer.

1NF: R_1 (PortfolioID, Description, link, freelancer_ID)

R_2 (portfolio_ID, Title)

2NF: R_1 (Portfolio_ID, Description, freelancer_ID)
R_2 (portfolio_ID, Title)
R_3 (

No further Normalization

⑤ Project

Project_ID → Title, Description, Budget, Client_ID, Project_ID

Is already in 3NF, no normalization required.

⑥ Category

Category_ID → Category_ID, Name.

No normalization required.

⑦ Proposal

Proposal_ID → Proposal_ID, Freelancer_ID, Project_ID, Bid, Description

No normalization required.

⑧ Contract

Contract_ID → Contract_ID, Start_Date, End_date, Email, Client_ID, Freelancer_ID, Project_ID.

here Email is associated with the Contract email formed by platform, ∴ Is Atomic.

No normalization Required.

(9) Review

Review_ID → Review_ID, Freelancer_ID, Comment, Email, Project_ID, Rating, Client_ID

Here Comment can be multivalued.

1NF: R(Review_ID, Freelancer_ID, Comment, Email, Project_ID, Rating, Client_ID)

R_2 (Review_ID, Comment)

2NF: R_1 (Review_ID, Freelancer_ID, Email, Project_ID, Client_ID)

R_2 (Review_ID, Comment)

R_3 (Rating, Comment)

No normalization possible

(10) Message

Message_ID → Content, Message_ID, Sent at, Sender_ID, Receiver_ID

No Normalization possible.

(11) Skill

Skill_ID → Skill_ID, Name

No normalization possible.

(12) Payment

Payment_ID → payment_ID, Freelancer_ID, Amount, Project_I

No normalization possible.

(13) Feedback

Feedback_ID → Comment, Rating, Freelancer ID, Client ID, Feedback ID;

1NF: R_1 (Feedback_ID, Rating, Freelancer ID, Client_ID,)
$
R_2 (Feedback_ID, Comments)

No normalization possible.

14 Notifications

Notification_ID → Recipt ID, Notification ID, Message, SentAt

No Normalization required.

SQL Queries

1)Creating:

```sql
create database lab;

use lab;
-- Create Admin table
CREATE TABLE Admin (
    admin_id INT PRIMARY KEY,
    admin_name VARCHAR(100),
    email VARCHAR(100),
    phone_number VARCHAR(20),
    user_type VARCHAR(20) -- Added to specify the type of user
);

-- Create Freelancer table
CREATE TABLE Free (
    freelancer_id INT PRIMARY KEY,
    freelancer_name VARCHAR(100),
    admin_id INT,
    email VARCHAR(100),
    phone_number VARCHAR(20),
    date_of_birth DATE,
    FOREIGN KEY (admin_id) REFERENCES Admin(admin_id)
);

-- Create Client table
CREATE TABLE Client (
    client_id INT PRIMARY KEY,
    client_name VARCHAR(100),
    email VARCHAR(100),
    email VARCHAR(100),
    phone_number VARCHAR(20),
    date_of_birth DATE,
    admin_id INT,
    FOREIGN KEY (admin_id) REFERENCES Admin(admin_id)
);

CREATE TABLE Skill (
    skill_id INT PRIMARY KEY,
    skill_name VARCHAR(50),
    description TEXT
);

-- Create Freelanceskill table
CREATE TABLE Freelanceskill (
    freelancer_id INT,
    skill_id INT,
    description TEXT,
    PRIMARY KEY (freelancer_id, skill_id),
    FOREIGN KEY (freelancer_id) REFERENCES Free(freelancer_id),
    FOREIGN KEY (skill_id) REFERENCES Skill(skill_id)
);

-- Create Portfolio table
CREATE TABLE Portfolio (
    portfolio_id INT PRIMARY KEY,
    freelancer_id INT,
    portfolio name VARCHAR(100),
```

```
    portfolio_name VARCHAR(100),
    portfolio_description TEXT,
    FOREIGN KEY (freelancer_id) REFERENCES Free(freelancer_id)
);

-- Create Category table
CREATE TABLE Category (
    category_id INT PRIMARY KEY,
    category_name VARCHAR(100),
    category_description TEXT
);

-- Create Project table
CREATE TABLE Project (
    project_id INT PRIMARY KEY,
    client_id INT,
    category_id INT,
    project_name VARCHAR(100),
    project_description TEXT,
    budget DECIMAL(10, 2),
    deadline DATE,
    FOREIGN KEY (client_id) REFERENCES Client(client_id),
    FOREIGN KEY (category_id) REFERENCES Category(category_id)
);

-- Create Proposal table
CREATE TABLE Proposal (
    proposal id INT PRIMARY KEY,
```

```
CREATE TABLE Proposal (
    proposal_id INT PRIMARY KEY,
    freelancer_id INT,
    project_id INT,
    proposal_details TEXT,
    proposal_date DATE,
    FOREIGN KEY (freelancer_id) REFERENCES Free(freelancer_id),
    FOREIGN KEY (project_id) REFERENCES Project(project_id)
);

-- Create Contract table
CREATE TABLE Contract (
    contract_id INT PRIMARY KEY,
    project_id INT,
    freelancer_id INT,
    client_id INT,
    contract_date DATE,
    start_date DATE,
    end_date DATE,
    FOREIGN KEY (project_id) REFERENCES Project(project_id),
    FOREIGN KEY (freelancer_id) REFERENCES Free(freelancer_id),
    FOREIGN KEY (client_id) REFERENCES Client(client_id)
);

-- Create Payment table
CREATE TABLE Payment (
    payment_id INT PRIMARY KEY,
    freelancer id INT,
```

```
    freelancer_id INT,
    project_id INT,
    amount DECIMAL(10,2),
    payment_date DATE,
    payment_method VARCHAR(50),
    FOREIGN KEY (freelancer_id) REFERENCES Free(freelancer_id),
    FOREIGN KEY (project_id) REFERENCES Project(project_id)
);

-- Create Review table
CREATE TABLE Review (
    review_id INT PRIMARY KEY,
    client_id INT,
    freelancer_id INT,
    project_id INT,
    review_details TEXT,
    rating INT,
    review_date DATE,
    FOREIGN KEY (client_id) REFERENCES Client(client_id),
    FOREIGN KEY (freelancer_id) REFERENCES Free(freelancer_id),
    FOREIGN KEY (project_id) REFERENCES Project(project_id)
);

-- Create Message table
CREATE TABLE Message (
    message_id INT PRIMARY KEY,
    sender_id INT,
    receiver id INT,
```

```
    receiver_id INT,
    message_details TEXT,
    sent_date DATETIME,
    FOREIGN KEY (sender_id) REFERENCES Free(freelancer_id),
    FOREIGN KEY (receiver_id) REFERENCES Client(client_id)
);

-- Create Notification table
CREATE TABLE Notification (
    notification_id INT PRIMARY KEY,
    user_id INT,
    notification_details TEXT,
    notification_date DATETIME,
    is_read BOOLEAN,
    FOREIGN KEY (user_id) REFERENCES Free(freelancer_id) ON DELETE CASCADE,
    FOREIGN KEY (user_id) REFERENCES Client(client_id) ON DELETE CASCADE
);

-- Create Feedback table
CREATE TABLE Feedback (
    feedback_id INT PRIMARY KEY,
    user_id INT,
    feedback_details TEXT,
    feedback_date DATE,
    FOREIGN KEY (user_id) REFERENCES Free(freelancer_id) ON DELETE CASCADE,
    FOREIGN KEY (user_id) REFERENCES Client(client_id) ON DELETE CASCADE
);
```

2) Inserting:

```sql
-- Insert values into Admin table
INSERT INTO Admin (admin_id, admin_name, email, phone_number, user_type)
VALUES
(1, 'John Doe', 'john@example.com', '1234567890', 'admin'),
(2, 'Alice Smith', 'alice@example.com', '0987654321', 'admin'),
(3, 'Bob Johnson', 'bob@example.com', '5555555555', 'admin'),
(4, 'Emily Brown', 'emily@example.com', '6666666666', 'admin'),
(5, 'Michael Davis', 'michael@example.com', '7777777777', 'admin'),
(6, 'Sarah Wilson', 'sarah@example.com', '8888888888', 'admin'),
(7, 'David Martinez', 'david@example.com', '9999999999', 'admin'),
(8, 'Jennifer Taylor', 'jennifer@example.com', '1010101010', 'admin'),
(9, 'William Anderson', 'william@example.com', '1212121212', 'admin'),
(10, 'Jessica Rodriguez', 'jessica@example.com', '1313131313', 'admin');

SELECT * FROM Admin;

-- Insert values into Free table
INSERT INTO Free (freelancer_id, freelancer_name, admin_id, email, phone_number, date_of_birth)
VALUES
(1, 'Michael Johnson', 1, 'michael.johnson@example.com', '1111111111', '1990-01-01'),
(2, 'Emily Wilson', 2, 'emily.wilson@example.com', '2222222222', '1991-02-02'),
(3, 'David Brown', 3, 'david.brown@example.com', '3333333333', '1992-03-03'),
(4, 'Sarah Taylor', 4, 'sarah.taylor@example.com', '4444444444', '1993-04-04'),
(5, 'Jessica Lee', 5, 'jessica.lee@example.com', '5555555555', '1994-05-05'),
(6, 'John Miller', 6, 'john.miller@example.com', '6666666666', '1995-06-06'),
(7, 'Alice Davis', 7, 'alice.davis@example.com', '7777777777', '1996-07-07'),
(8, 'Bob Clark', 8, 'bob.clark@example.com', '8888888888', '1997-08-08'),
```

```sql
-- Insert values into Client table
INSERT INTO Client (client_id, client_name, email, phone_number, date_of_birth, admin_id)
VALUES
(1, 'Karen Thompson', 'karen@example.com', '1111111111', '1980-01-01', 1),
(2, 'Richard Harris', 'richard@example.com', '2222222222', '1981-02-02', 2),
(3, 'Maria Martinez', 'maria@example.com', '3333333333', '1982-03-03', 3),
(4, 'Christopher Wilson', 'christopher@example.com', '4444444444', '1983-04-04', 4),
(5, 'Linda Garcia', 'linda@example.com', '5555555555', '1984-05-05', 5),
(6, 'Daniel Rodriguez', 'daniel@example.com', '6666666666', '1985-06-06', 6),
(7, 'Patricia Brown', 'patricia@example.com', '7777777777', '1986-07-07', 7),
(8, 'Matthew Lee', 'matthew@example.com', '8888888888', '1987-08-08', 8),
(9, 'Laura Taylor', 'laura@example.com', '9999999999', '1988-09-09', 9),
(10, 'Jason Clark', 'jason@example.com', '1010101010', '1989-10-10', 10);

SELECT * FROM Client;

-- Insert values into Skill table
INSERT INTO Skill (skill_id, skill_name, description)
VALUES
(1, 'Web Development', 'Designing and developing websites and web applications'),
(2, 'Graphic Design', 'Creating visual concepts using computer software'),
(3, 'Content Writing', 'Producing written content for various purposes'),
(4, 'Digital Marketing', 'Promoting brands and products through online channels'),
(5, 'Mobile App Development', 'Creating applications for mobile devices'),
(6, 'SEO', 'Improving website visibility and ranking on search engines'),
(7, 'Data Entry', 'Entering data into computer systems or databases'),
(8, 'Translation', 'Converting written text from one language to another'),
(9, 'Video Editing', 'Manipulating and arranging video shots'),
```

```sql
INSERT INTO Freelanceskill (freelancer_id, skill_id, description)
VALUES
(1, 1, 'Expert in web design and development'),
(1, 2, 'Proficient in HTML, CSS, and JavaScript'),
(2, 3, 'Experienced in graphic design and illustration'),
(2, 4, 'Skilled in Adobe Photoshop and Illustrator'),
(3, 5, 'Professional writer with expertise in content creation'),
(3, 6, 'Knowledgeable in SEO best practices'),
(4, 7, 'Experienced in social media marketing and advertising'),
(4, 8, 'Familiar with Facebook Ads and Google AdWords'),
(5, 9, 'Mobile app development specialist with iOS and Android skills'),
(5, 10, 'Proficient in Swift, Java, and Kotlin');

SELECT * FROM Freelanceskill;


-- Insert values into Portfolio table
INSERT INTO Portfolio (portfolio_id, freelancer_id, portfolio_name, portfolio_description)
VALUES
(1, 1, 'Portfolio 1', 'A collection of web development projects'),
(2, 2, 'Portfolio 2', 'Samples of graphic design work'),
(3, 3, 'Portfolio 3', 'Examples of content writing projects'),
(4, 4, 'Portfolio 4', 'Projects related to digital marketing'),
(5, 5, 'Portfolio 5', 'Mobile app development projects'),
(6, 6, 'Portfolio 6', 'SEO optimization examples'),
(7, 7, 'Portfolio 7', 'Data entry projects completed'),
(8, 8, 'Portfolio 8', 'Translations in various languages'),
(9, 9, 'Portfolio 9', 'Video editing samples'),
```

```sql
INSERT INTO Category (category_id, category_name, category_description)
VALUES
(1, 'Web Development', 'Projects related to website and web application development'),
(2, 'Graphic Design', 'Designing visual content for various purposes'),
(3, 'Writing', 'Content creation and writing services'),
(4, 'Marketing', 'Promotional activities and strategies'),
(5, 'Mobile App Development', 'Creating applications for mobile devices'),
(6, 'SEO', 'Search engine optimization services'),
(7, 'Data Entry', 'Entering data into computer systems'),
(8, 'Translation', 'Converting text from one language to another'),
(9, 'Video Editing', 'Editing and enhancing video content'),
(10, 'Customer Support', 'Assistance provided to customers');

SELECT * FROM Category;

-- Insert values into Project table
INSERT INTO Project (project_id, client_id, category_id, project_name, project_description, budget, deadline)
VALUES
(1, 1, 1, 'Website Redesign', 'Redesigning company website for better user experience', 5000.00, '2024-05-15'),
(2, 2, 2, 'Logo Design', 'Creating a new logo for a startup company', 1000.00, '2024-06-01'),
(3, 3, 3, 'Article Writing', 'Writing articles for a blog on various topics', 800.00, '2024-05-20'),
(4, 4, 4, 'Social Media Campaign', 'Running a promotional campaign on social media platforms', 2000.00, '2024-06-10'),
(5, 5, 5, 'Mobile App Development', 'Developing a mobile application for a business', 10000.00, '2024-07-01'),
(6, 6, 6, 'SEO Optimization', 'Improving website ranking and visibility on search engines', 1500.00, '2024-05-25'),
(7, 7, 7, 'Data Entry Project', 'Entering data from handwritten forms into digital format', 600.00, '2024-06-05'),
(8, 8, 8, 'Translation Services', 'Translating documents from English to Spanish', 400.00, '2024-05-30'),
(9, 9, 9, 'Video Editing', 'Editing and enhancing promotional video for a company', 1200.00, '2024-06-15'),
(10, 10, 10, 'Customer Support', 'Providing customer support via email and live chat', 800.00, '2024-06-20');
```

```sql
INSERT INTO Proposal (proposal_id, freelancer_id, project_id, proposal_details, proposal_date)
VALUES
(1, 1, 1, 'I have extensive experience in website redesign and can deliver a modern and user-friendly design within the specified budget and
(2, 2, 2, 'I specialize in logo design and can create a unique and professional logo that reflects your company\'s identity.', '2024-04-02'),
(3, 3, 3, 'I am a skilled writer capable of producing high-quality articles on a wide range of topics.', '2024-04-02'),
(4, 4, 4, 'I have expertise in running successful social media campaigns and can help increase your brand\'s visibility.', '2024-04-02'),
(5, 5, 5, 'I have developed several mobile applications and can create a customized solution tailored to your business needs.', '2024-04-02')
(6, 6, 6, 'I specialize in SEO optimization and can improve your website\'s ranking and visibility on search engines.', '2024-04-02'),
(7, 7, 7, 'I am detail-oriented and proficient in data entry tasks, ensuring accurate and efficient completion of projects.', '2024-04-02'),
(8, 8, 8, 'I am fluent in English and Spanish and can provide accurate and reliable translation services.', '2024-04-02'),
(9, 9, 9, 'I have advanced video editing skills and can create visually appealing and engaging content for your company.', '2024-04-02'),
(10, 10, 10, 'I have experience in providing excellent customer support and can assist your customers effectively.', '2024-04-02');

SELECT * FROM Proposal;

-- Insert values into Contract table
INSERT INTO Contract (contract_id, project_id, freelancer_id, client_id, contract_date, start_date, end_date)
VALUES
(1, 1, 1, 1, '2024-04-05', '2024-04-10', '2024-05-15'),
(2, 2, 2, 2, '2024-04-05', '2024-04-10', '2024-06-01'),
(3, 3, 3, 3, '2024-04-05', '2024-04-10', '2024-05-20'),
(4, 4, 4, 4, '2024-04-05', '2024-04-10', '2024-06-10'),
(5, 5, 5, 5, '2024-04-05', '2024-04-10', '2024-07-01'),
(6, 6, 6, 6, '2024-04-05', '2024-04-10', '2024-05-25'),
(7, 7, 7, 7, '2024-04-05', '2024-04-10', '2024-06-05'),
(8, 8, 8, 8, '2024-04-05', '2024-04-10', '2024-05-30'),
(9, 9, 9, 9, '2024-04-05', '2024-04-10', '2024-06-15'),
(10, 10, 10, 10, '2024-04-05', '2024-04-10', '2024-06-20');
```


```sql
INSERT INTO Proposal (proposal_id, freelancer_id, project_id, proposal_details, proposal_date)
VALUES
(1, 1, 1, 'I have extensive experience in website redesign and can deliver a modern and user-friendly design within the specified budget and
(2, 2, 2, 'I specialize in logo design and can create a unique and professional logo that reflects your company\'s identity.', '2024-04-02'),
(3, 3, 3, 'I am a skilled writer capable of producing high-quality articles on a wide range of topics.', '2024-04-02'),
(4, 4, 4, 'I have expertise in running successful social media campaigns and can help increase your brand\'s visibility.', '2024-04-02'),
(5, 5, 5, 'I have developed several mobile applications and can create a customized solution tailored to your business needs.', '2024-04-02')
(6, 6, 6, 'I specialize in SEO optimization and can improve your website\'s ranking and visibility on search engines.', '2024-04-02'),
(7, 7, 7, 'I am detail-oriented and proficient in data entry tasks, ensuring accurate and efficient completion of projects.', '2024-04-02'),
(8, 8, 8, 'I am fluent in English and Spanish and can provide accurate and reliable translation services.', '2024-04-02'),
(9, 9, 9, 'I have advanced video editing skills and can create visually appealing and engaging content for your company.', '2024-04-02'),
(10, 10, 10, 'I have experience in providing excellent customer support and can assist your customers effectively.', '2024-04-02');

SELECT * FROM Proposal;

-- Insert values into Contract table
INSERT INTO Contract (contract_id, project_id, freelancer_id, client_id, contract_date, start_date, end_date)
VALUES
(1, 1, 1, 1, '2024-04-05', '2024-04-10', '2024-05-15'),
(2, 2, 2, 2, '2024-04-05', '2024-04-10', '2024-06-01'),
(3, 3, 3, 3, '2024-04-05', '2024-04-10', '2024-05-20'),
(4, 4, 4, 4, '2024-04-05', '2024-04-10', '2024-06-10'),
(5, 5, 5, 5, '2024-04-05', '2024-04-10', '2024-07-01'),
(6, 6, 6, 6, '2024-04-05', '2024-04-10', '2024-05-25'),
(7, 7, 7, 7, '2024-04-05', '2024-04-10', '2024-06-05'),
(8, 8, 8, 8, '2024-04-05', '2024-04-10', '2024-05-30'),
(9, 9, 9, 9, '2024-04-05', '2024-04-10', '2024-06-15'),
(10, 10, 10, 10, '2024-04-05', '2024-04-10', '2024-06-20');
```

```sql
INSERT INTO Payment (payment_id, freelancer_id, project_id, amount, payment_date, payment_method)
VALUES
(1, 1, 1, 2500.00, '2024-05-10', 'Bank Transfer'),
(2, 2, 2, 1000.00, '2024-06-01', 'PayPal'),
(3, 3, 3, 800.00, '2024-05-20', 'Credit Card'),
(4, 4, 4, 1500.00, '2024-06-10', 'Bank Transfer'),
(5, 5, 5, 5000.00, '2024-07-01', 'PayPal'),
(6, 6, 6, 1200.00, '2024-05-25', 'Credit Card'),
(7, 7, 7, 600.00, '2024-06-05', 'Bank Transfer'),
(8, 8, 8, 400.00, '2024-05-30', 'PayPal'),
(9, 9, 9, 1000.00, '2024-06-15', 'Credit Card'),
(10, 10, 10, 800.00, '2024-06-20', 'Bank Transfer');

SELECT *FROM Payment;

-- Insert values into Review table
INSERT INTO Review (review_id, client_id, freelancer_id, project_id, review_details, rating, review_date)
VALUES
(1, 1, 1, 1, 'Michael did an excellent job redesigning our website. Highly recommended!', 5, '2024-05-20'),
(2, 2, 2, 2, 'Emily delivered a fantastic logo design. Very happy with the result!', 5, '2024-06-05'),
(3, 3, 3, 3, 'David provided high-quality articles for our blog. Will hire again!', 4, '2024-05-25'),
(4, 4, 4, 4, 'Sarah ran an effective social media campaign that boosted our brand awareness.', 4, '2024-06-15'),
(5, 5, 5, 5, 'Jessica developed a great mobile app for our business. Impressed with her skills!', 5, '2024-07-05'),
(6, 6, 6, 6, 'John improved our website\'s ranking significantly. Great SEO work!', 4, '2024-05-30'),
(7, 7, 7, 7, 'Alice completed the data entry project accurately and on time. Thank you!', 4, '2024-06-10'),
(8, 8, 8, 8, 'Bob provided accurate translations. Professional and reliable service!', 5, '2024-06-01'),
(9, 9, 9, 9, 'Jennifer edited our promotional video beautifully. Very talented!', 5, '2024-06-20'),
(10, 10, 10, 10, 'William provided excellent customer support. Very responsive and helpful!', 5, '2024-07-10');
```

```sql
INSERT INTO Message (message_id, sender_id, receiver_id, message_details, sent_date)
VALUES
(1, 1, 2, 'Hi there! I noticed your project and would like to discuss it further.', '2024-04-02 10:00:00'),
(2, 2, 1, 'Hello! Sure, lets talk about it. What specific details would you like to know?', '2024-04-02 10:05:00'),
(3, 3, 1, 'Hi! Im interested in working with you on your project.', '2024-04-02 10:10:00'),
(4, 1, 3, 'Great! Can you please provide more information about your skills and experience?', '2024-04-02 10:15:00'),
(5, 1, 4, 'Hello! I saw your profile and would like to collaborate on a project.', '2024-04-02 10:20:00'),
(6, 4, 1, 'Sure, I had love to work with you. What type of project do you have in mind?', '2024-04-02 10:25:00'),
(7, 2, 3, 'Hi there! I have a new project idea that I think you had be perfect for.', '2024-04-02 10:30:00'),
(8, 3, 2, 'That sounds interesting! Im eager to hear more about it.', '2024-04-02 10:35:00'),
(9, 5, 2, 'Hello! I need your expertise on a mobile app development project.', '2024-04-02 10:40:00'),
(10, 2, 5, 'Sure, Im available to discuss the project details whenever you are ready.', '2024-04-02 10:45:00');

SELECT * FROM Message;

-- Insert values into Notification table
INSERT INTO Notification (notification_id, user_id, notification_details, notification_date, is_read)
VALUES
(1, 1, 'You have a new message from User 2.', '2024-04-02 10:00:00', FALSE),
(2, 2, 'New project proposal received.', '2024-04-02 10:05:00', FALSE),
(3, 3, 'Your project proposal has been accepted.', '2024-04-02 10:10:00', FALSE),
(4, 1, 'Payment received for Project X.', '2024-04-02 10:15:00', FALSE),
(5, 4, 'New project proposal received.', '2024-04-02 10:20:00', FALSE),
(6, 5, 'You have a new message from User 2.', '2024-04-02 10:25:00', FALSE),
(7, 2, 'New project proposal received.', '2024-04-02 10:30:00', FALSE),
(8, 3, 'Your project proposal has been accepted.', '2024-04-02 10:35:00', FALSE),
(9, 1, 'Payment received for Project Y.', '2024-04-02 10:40:00', FALSE),
(10, 5, 'New project proposal received.', '2024-04-02 10:45:00', FALSE);
```

```sql
INSERT INTO Feedback (feedback_id, user_id, feedback_details, feedback_date)
VALUES
(1, 1, 'Great work! Highly recommended freelancer.', '2024-04-02'),
(2, 2, 'The project was completed on time with excellent quality.', '2024-04-02'),
(3, 3, 'Good communication and delivered as promised.', '2024-04-02'),
(4, 4, 'Professional and skilled freelancer.', '2024-04-02'),
(5, 5, 'Very satisfied with the work done.', '2024-04-02'),
(6, 1, 'Excellent job! Will hire again for future projects.', '2024-04-02'),
(7, 2, 'Highly impressed with the attention to detail.', '2024-04-02'),
(8, 3, 'Exceeded expectations. Great experience overall.', '2024-04-02'),
(9, 4, 'Very cooperative and responsive freelancer.', '2024-04-02'),
(10, 5, 'Quality work delivered ahead of schedule.', '2024-04-02');
```

3)SQL Query questions

- Update freelancer mail

```sql
UPDATE Free
SET email = 'new_email@example.com'
WHERE freelancer_id = 1;
```

- Delete by client id

```sql
DELETE FROM Client
WHERE client_id = 5;
```

- Inner join to retrive client and project details

```sql
SELECT c.client_name, p.project_name
FROM Client c
INNER JOIN Project p ON c.client_id = p.client_id;
```

- Filter client by name

```sql
SELECT * FROM Client
WHERE client_name LIKE '%Smith%';
```

- Retrive max budget for projects

```sql
SELECT MAX(budget) AS max_budget
FROM Project;
```

- Retrive minimum payment amount

```sql
SELECT MIN(amount) AS min_payment
FROM Payment;
```

```sql
UPDATE Notification
SET is_read = TRUE
WHERE user_id = 1;
```

- Delete feedback by ID

```sql
DELETE FROM Feedback
WHERE feedback_id = 3;
```

- Left join to retrive freelancer and portfolio details

```sql
SELECT f.freelancer_name, p.portfolio_name
FROM Free f
LEFT JOIN Portfolio p ON f.freelancer_id = p.freelancer_id;
```

- AND operator

```sql
SELECT * FROM Proposal
WHERE freelancer_id = 3 AND project_id = 3;
```

- Count number of clients

```sql
SELECT COUNT(client_id) AS client_count
FROM Client;
```

- Sum of payments for freelancers

```sql
SELECT SUM(amount) AS total_payments
FROM Payment
WHERE freelancer_id = 1;
```

- Retrive average rating for freelancer

```sql
SELECT AVG(rating) AS avg_rating
FROM Review
WHERE freelancer_id = 2;
```

- Update project deadline

```
UPDATE Project
SET deadline = '2024-07-31'
WHERE project_id = 2;
```

- Delete proposal by freelanceid

```
DELETE FROM Proposal
WHERE freelancer_id = 4;
```

- Join to retrive client and review details

```
SELECT c.client_name, r.review_details
FROM Client c
INNER JOIN Review r ON c.client_id = r.client_id;
```

- OR operator

```
SELECT * FROM Free
WHERE freelancer_id = 1 OR freelancer_id = 2;
```

- Retrive latest notification

```
SELECT MAX(notification_date) AS latest_notification_date
FROM Notification;
```

- Retrive earliest contract startdate

```
SELECT MIN(start_date) AS earliest_start_date
FROM Contract;
```

- Update client phone number

```
UPDATE Client
SET phone_number = '1234567890'
WHERE client_id = 3;
```

- Delete from skill

```
DELETE FROM Skill
WHERE skill_id = 7;
```

- Join project and category details

```
SELECT p.project_name, c.category_name
FROM Project p
INNER JOIN Category c ON p.category_id = c.category_id;
```

- IN operator

```sql
SELECT * FROM Free
WHERE admin_id IN (1, 2, 3);
```

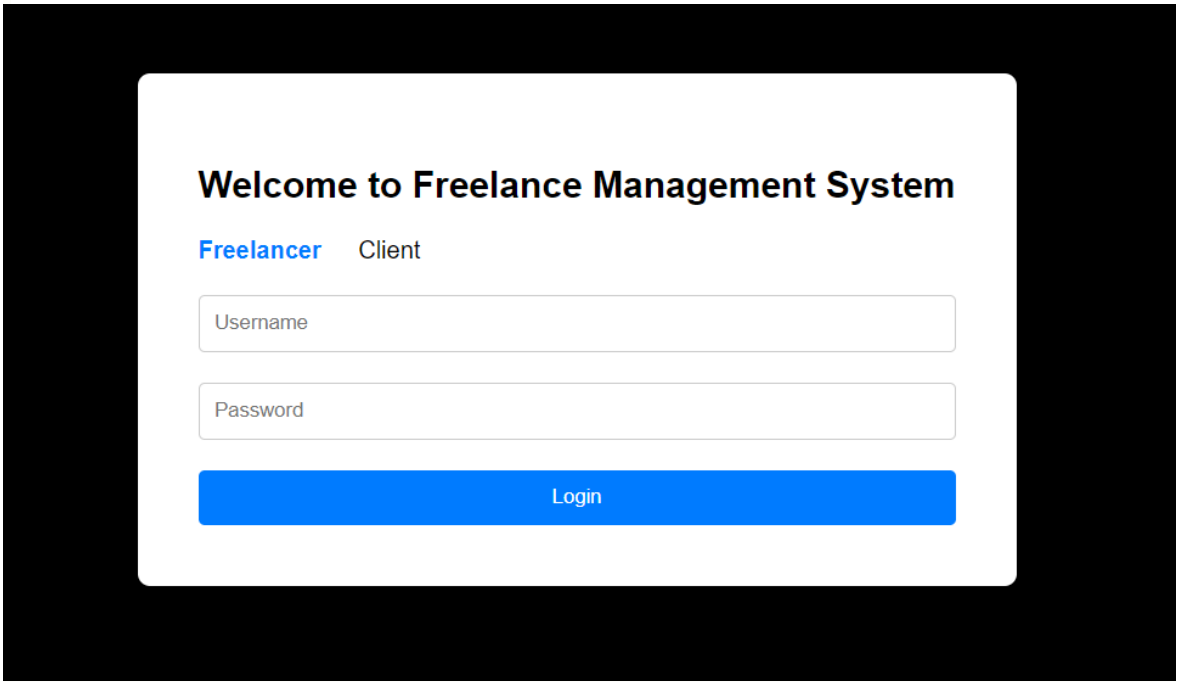
- Retrive count of unread notification

```sql
SELECT COUNT(notification_id) AS unread_notifications
FROM Notification
WHERE is_read = FALSE;
```

- Join to retrive freelancer, contract, project details

```sql
SELECT f.freelancer_name, c.contract_date, p.project_name
FROM Free f
INNER JOIN Contract c ON f.freelancer_id = c.freelancer_id
INNER JOIN Project p ON c.project_id = p.project_id;
```

Project demonstration



Learnings from the Project

1. Entity-Relationship Modeling (ERM):

- Understand the principles of ERM for designing the database schema.

- Identify entities, attributes, and relationships between them to model the business domain accurately.

- Use tools like entity-relationship diagrams (ERDs) to visualize the database structure.

2. Data Integrity Constraints:

- Learn about various constraints such as `PRIMARY KEY`, `FOREIGN KEY`, `UNIQUE`, `NOT NULL`, and `CHECK` constraints to enforce data integrity rules.

- Understand how constraints ensure the accuracy, consistency, and reliability of data stored in the database.

3. Normalization Principles:

- Study normalization forms (e.g., 1NF, 2NF, 3NF) and their significance in database design.

- Apply normalization techniques to organize data into logical structures, minimizing redundancy and dependency issues.

4. Database Transactions:

- Comprehend the concept of transactions and their role in maintaining database consistency.

- Learn about ACID properties (Atomicity, Consistency, Isolation, Durability) and how transactions ensure data integrity in multi-user environments.

5. Query Optimization:

- Explore techniques for optimizing SQL queries to enhance database performance.

- Understand query execution plans, indexing strategies, and query rewriting techniques to minimize resource consumption and improve response times.

6. Concurrency Control:

- Gain insights into concurrency control mechanisms for managing simultaneous access to the database by multiple users.

- Learn about locking mechanisms, isolation levels, and transaction management strategies to prevent data corruption and ensure data consistency.

7. Backup and Recovery:

- Understand the importance of database backups and recovery strategies for safeguarding data against accidental loss or corruption.

- Explore backup methods (e.g., full, incremental, differential) and recovery procedures to restore data in case of failures.

8. Data Warehousing and Business Intelligence:

- Explore concepts of data warehousing and the role of a data warehouse in decision-making processes.

- Learn about data mining, OLAP (Online Analytical Processing), and reporting tools for extracting valuable insights from large datasets.

9. Database Security:

- Familiarize yourself with database security best practices to protect sensitive information from unauthorized access, disclosure, or tampering.

- Learn about authentication mechanisms, access control lists (ACLs), encryption techniques, and auditing mechanisms to strengthen database security.

10. Scalability and Performance Tuning:

- Understand the factors influencing database scalability and strategies for scaling systems to accommodate growing workloads.

- Learn performance tuning techniques such as query optimization, indexing, caching, and partitioning to enhance system responsiveness and throughput.

Challenges Faced

1. **Database Design:** Creating a relationship between the tables or entities was a challenge. Oftentimes, introduction of primary keys from one table as foreign keys into the others led to cycles or loops. As a result of these loops, it was a challenge to be able to alter or even implement the table at times. Hence, it required careful assessment of all entities and their references for any loops before implementation to ensure good performance, data integrity and no discrepancies.

2. **Scalability Creating** additional rows without letting it interfere with the previous tables was a difficult task. Hence, altering the tables was also posed to be a challenge. Thus, we had to develop the backend architecture to be scalable and able to handle increased loads as the project grows, without compromising performance or stability.

3. **Error Handling and Logging**: Editing tables with respect to foreign keys often led to updation and deletion anomalies. Thus, it was important to implement effective logging strategies to track and troubleshoot issues, to ensure the reliability and maintainability of the backend.

4. **Concurrency and Transactions** Handling concurrent requests and ensuring data consistency through proper transaction management.

Self -Learning beyond classroom:

Through our collaborative efforts in establishing this repository, we not only acquired a comprehensive understanding of the subject matter but also honed our skills well beyond the confines of traditional classroom teachings. This journey encompassed a fusion of theoretical foundations, hands-on SQL development, and industry perspectives, equipping us with more insight for future endeavors. Some of them are mentioned:

1. Data Analysis Tools and Techniques: Learning about the TKinter library that is used in making front end user interface.Tkinter is a standard GUI (Graphical User Interface) library for Python. It provides a way to create windows, dialogs, and all sorts of graphical elements in Python applications. Also, learned to apply the 'mySQL.connector' library that allowed us to connect our database and entities for an appropriate login interface.

2. Detailed learning on updation and deletion anomalies: As we worked on this repository, we could familiarize ourselves even further with the various anomalies that come with database development. It gave us a more clear idea on the working of mySQL and explained the requirement of normalisations within the database to eradicate redundancies.

3. Freelance Industry Knowledge: Familiarizing ourselves with the freelanceindustry's regulatory landscape and storing information on how things and functions or certain events are managed in the industry.As we were unknown with the certain proceedings in some processes we had to take assumptions of our own and carry our work ahead.

4. Data Security and Privacy: Learning about data security best practices and privacy regulations relevant to handling sensitive FMS data. Understanding access control mechanisms, and protocols for secure data transmission

Conclusion

In conclusion, the Freelancer Management System (FMS) stands as a cornerstone in the modern business landscape, offering a comprehensive solution to the challenges posed by the burgeoning gig economy. As organizations increasingly rely on freelance talent to meet specialized needs and navigate dynamic market demands, the FMS emerges as a strategic imperative for efficient talent acquisition, management, and deployment. By providing a centralized platform for sourcing, onboarding, and managing freelance professionals, the FMS streamlines operations and unlocks access to a diverse pool of skills precisely when needed. Its significance transcends mere convenience, as it empowers businesses to stay agile and competitive in a rapidly evolving marketplace.

Moreover, the FMS addresses the intricate nuances of freelance management, from coordination hurdles to compliance complexities, with structured workflows and automated processes. By facilitating seamless communication, tracking project progress, and ensuring compliance with regulations, the FMS becomes a trusted ally in optimizing resource utilization and mitigating risks.

Furthermore, within the broader context of market dynamics, the importance of the FMS is underscored by the strategic insights derived from market research. By leveraging data-driven analyses of industry trends, competitor strategies, and emerging opportunities, businesses can harness the full potential of the FMS to drive innovation, foster collaboration, and achieve sustainable growth.

In essence, the Freelancer Management System isn't just a software solution—it's a catalyst for organizational excellence, enabling businesses to navigate the complexities of the gig economy with confidence and foresight. As the freelance landscape continues to evolve, the FMS remains a cornerstone of success, empowering businesses to adapt, thrive, and lead in an ever-changing business environment.