Final Project Guidelines for Database Course

1 Project Overview

The final project for this course involves developing a database application that demonstrates your understanding of database concepts, including database design, normalization, and user interface implementation. You are expected to design and implement a complete database solution for a real-world problem. This project should include both a well-designed database and a user-friendly interface.

2 Project Requirements

The project must meet the following key criteria:

1. **Complexity of the Idea and Applicability:**
   - The project should involve a database that is complex enough to demonstrate your understanding of the course concepts, including normalization and data modeling.
   - The application should address a real-world problem or provide a useful solution.
   - The project should be applicable in a practical scenario, such as inventory management, student record systems, hospital management, or any other system where data storage and retrieval are critical.
2. **Good Database Design:**
   - The database should be designed using normalization techniques (at least up to 3NF, preferably BCNF). This ensures minimal redundancy and dependency.
   - The database design should be documented, including tables, relationships, primary and foreign keys, and any other constraints.
   - You should include a well-designed Entity-Relationship (ER) diagram that clearly represents the structure of your database.
3. **Good User Interface:**
   - The user interface (UI) should be intuitive and easy to use. It should allow users to interact with the database and perform CRUD (Create, Read, Update, Delete) operations.
   - The UI can be implemented using `Tkinter` (as discussed in class) or any other web-based framework (e.g., Flask, Django, or React).
   - The UI should present the data in a user-friendly manner (e.g., tables, forms, etc.), and provide feedback to the user (e.g., success or error messages).

4. **Implementation:**
    - You must implement the database using an SQL-based system (e.g., MySQL).
    - The application must be able to interact with the database through the UI, performing database operations as requested by the user.
    - Your application should seamlessly perform adding, deleting, updating, querying, and analyzing data in your database (for example give a summary of the data in the database).
    - Your implementation should handle edge cases (e.g., input validation, handling errors).
5. **Additional Features (Optional but Recommended):**
    - *Authentication and Authorization*: Implement user login and roles (e.g., admin, regular user).
    - *Reporting*: Include a reporting feature where the user can generate reports (e.g., sales report, student grades report, etc.).
    - *Data Export/Import*: Provide the option to export data to formats such as CSV or Excel, or import data from these formats.
    - *Search Functionality*: Implement advanced search functionality that allows the user to search the database based on multiple criteria.
6. **Documentation:**
    - Provide a comprehensive report detailing your database schema, design decisions, and any additional features you implemented.
    - The report should also include:
        - Entity-Relationship (ER) diagram.
        - SQL code for creating the database schema (tables, relationships).

3 Assessment Criteria

Your final project will be assessed based on the following criteria:

1. **Complexity and Applicability of the Idea:** The problem should be realistic and appropriately challenging for a database project.
2. **Database Design:** Quality of the database schema, including normalization and relationships. Proper use of primary keys, foreign keys, and constraints.
3. **User Interface:** The user interface should be functional, user-friendly, and visually appealing. The interface should facilitate interaction with the database and handle basic operations.
4. **Implementation:** Proper functionality and efficient database queries. Handling edge cases, error handling, and testing.
5. **Documentation:** Clear and comprehensive documentation of the project, including the database schema, ER diagram, SQL code.

4 Submission Requirements

The final project should include the following:

- **GitHub Repository:** Your project must be uploaded to a GitHub repository. The repository should contain all the source code, SQL files, documentation, and any other relevant files. Ensure that the repository is well-organized, and commit messages are clear and descriptive. You should submit the link for the repository to Moodle.
- One page of project explanation like what the project is about and its features.
- One page ER or EER diagram for the project.

5 Team Formation

- **Team Size:** Teams of up to 4 students are allowed to work together on the project.
- **Roles and Responsibilities:** It is recommended that each team member take on specific roles (e.g., database design, UI development, testing, documentation) to divide the work efficiently.
- **Collaboration:** Ensure that all team members contribute equally to the project. Any team member who is not contributing to the project may affect the team's final grade.

6 Presentation

- The project should be presented in around 15-20 minutes over Zoom in an online session after the final exam.