SOFE 3700U Data Management Systems
Projects Information - Fall 2025

- **Objective**: Apply the data management principles that you learned during the course to a real-life application. The group project is an important part of this course since it enables you to gain experience and apply concepts and technologies and utilizes many of the concepts presented in class.
- **Teams**: each team should be at max 4 students.
- **Deliverables**:
  - Project report per phase (one for the team)
  - Each report should include a **contribution matrix**.
  - Project presentation and final report (one for the team, with contributions from all members)
  - Team evaluation.
- **Evaluation Criteria:**
  - Phase I and II [**5%, 15%**] based on the report:
    - clarity of the report, design, and results
  - The final report and presentation will be based on the following grading criteria:
    - **Presentation and Demo (30%)**
      - The group will demonstrate and defend their work. You must prepare PowerPoint Slides for a 5 to 7-minutes presentation. Each presentation will be followed by 2-minutes for questions.
    - **Technical Content (50%)**
      - **Implementation/Coding/Performance Evaluation (30%)**
        - This all depends on the project: if you implemented a system then how well the system is implemented, what is the level of functionality, creativity in developing the system, and its usability.
      - **Final Report (20%)**
        - This is the project final report. 10 - 12 pages, Times 11 Font, Paper includes: an abstract, introduction, relation to other work, the main body of work, conclusions with contribution made, schematics, design diagrams, thoughts about any future work, references. The report should include the goals of the project, its relationship to the course, design and implementation, results and analysis. Your report should include snapshots of the results for all queries used.
- The following factors will be amongst those considered in evaluating the project:
  - Choice and justification of architecture, technology, platform, etc.;
  - Bugs found and fixed; possible future enhancements
  - Appropriateness of solution to the stated problem
  - Interface design and usability
  - Standard of writing, including grammar and spelling
  - Organization of reports, including clarity of introduction, logic of structure and navigability
  - Quality of code, including coding style: comments, modularity, etc.
  - Use of technology or concepts addressed in the rest of the course
  - Delineation of creative activity, i.e. what you have produced that is new
  - Extent of research done, including history of problem domain
- **Deadline: (Start Date: Sept. 18, 2024)**
  - Submit your group members' names (and IDs) and project title to TA by Sept. 30, 2024.
  - Phase I: due on Oct. 13, 2024. Midnight.
  - Phase II: due on Nov. 03, 2024. Midnight.
  - Final Report and Presentation: due on Nov. 17, 2024.

Phase I: Project Proposal (5%)

- Choose an application area: some ideas (but do not use any of them):
  - Bookstore
  - Car Rental Reservation
  - Personal Photos Organizer
  - Nobel Awards
  - Restaurants Finder
  - University database system
  - Bibliography database system

- Submit your proposal in PDF via Canvas (One per group)
  - Proposal consists of 1-2 pages (double spaced, font Times New Roman, Size 11)

- Your proposal should explicitly state the following:
  - Problem your project will address
  - Goals and motivations
  - Related work: short survey of related work and how existing work differs from your proposed work
  - Methodology and plan for your project

Phase II: Project Design (15%)

- In this phase, you will begin designing and constructing the data store for your application. Use the proposed application area from Phase I to construct the database for your project. You can use any database format you like (i.e. PostgreSQL, MySQL, Microsoft SQL, Oracle Database, etc…). The following items are the required deliverables for Phase II.

- **Deliverables:**
- Part A: Relational Schema (5 points)
  - Create relational database schema for the proposed application area from Phase I:
    - Actual SQL create table commands are required.
    - A graphical diagram of the relations is required.

- Part B: Sample Data (2 points)
  - Populate your database with sample data. Each relation should contain at least 6 tuples. Make sure that the populated data is suitable for the type of queries in Part C.

- Part C: Views (3 points)
  - Create (write English description and SQL syntax) of 10 views that a user of the database system would find useful. From these 10 views, the first 5 are common for all groups, and you can create your own views for the remaining ones.
    - View 1: Computes a join of at least three tables
    - View 2: Uses nested queries with the ANY or ALL operator and uses a GROUP BY clause
    - View 2: A correlated nested query
    - View 4: Uses a FULL JOIN
    - View 5: Uses nested queries with any of the set operations UNION, EXCEPT, or INTERSECT

- Part D: E-R diagram: (5 points)
  - Create an ER schema diagram for your project database.

- # Phase III: Project Final Report and Presentation (80%)
  - In this phase, you are asked to use your imagination in designing and developing an application that serves as the frontend of your database application. That is, the overall functionality of the system is delivered using multiple Web pages, an XML document or JSON, and Web Services. The project should be relevant to the course content; in particular, it should have some connection with the Databases, World Wide Web, Scripting technologies, and Markup Languages, XML/JSON, and Web services. You may use any Web programming language (i.e. PHP, Node.js, Django, Rails, ASP.NET, CFM, etc..).

- # Phase III Details:
  - ## Backend Operations and Functions:
    1. **User Authentication**:
       - Login/Logout Functionality: Implement secure login and logout operations using session management.
       - User Role Management: Define different roles (admin, user, guest) and restrict access to certain functionalities based on roles.
    2. **Data Validation**:
       - Input Validation: Ensure that all user inputs are validated (e.g., email format, required fields, data type checks) before saving to the database.
    3. **API Integration**:
       - REST/JSON API: Create API endpoints to allow communication with the frontend, such as POST, GET, PUT, and DELETE operations.
       - External API Integration: Integrate with an external service (e.g., weather, social media, or Ads) to fetch data and save results into your database.
       - One of the following either using (SOAP/XML) OR (REST/JSON):
         - One Web service that you implement using (SOAP/XML) OR (REST/JSON)
         - At least one Web service that generates an XML document or JSON
         - At least one Web service (SOAP/REST) that interacts with a database.
         - **If using SOAP/XML**: An XML document with a defined XML Schema that you implement A Web page that you implement which reads and displays the results from the XML document
         - **If using REST/JSON:** A web page that uses an external REST/JSON API (e.g. Twitter) At least two queries associated with the functionality (e.g. saving Tweets in the database)

    4. **CRUD Operations**:
       - Create, Read, Update, delete: Implement full CRUD operations for your data entities (e.g., users, orders, products) in the backend.
    5. **Data Export**:
       - Implement functionality to export data to different formats (CSV, PDF) using libraries like pandas for Python or similar in other languages.
    6. **Error Handling**:
       - Implement error-handling functions to manage database connection failures, wrong API inputs, or other exceptions.

  - ## Frontend Operations:
    - Your system should include at least the following features/components:
      - Two or more Web pages that you implement
        - Web pages will execute the 10 views from Phase II (Part C) and displaying the results in the proper tabular HTML format

1. **Dynamic Forms**:
    - Create interactive forms with JavaScript or any to allow users to submit data dynamically (e.g., form validation, auto-fill suggestions).
2. **AJAX Requests**:
    - Use AJAX or Fetch API to interact with the backend without refreshing the page (for submitting forms or fetching data in real-time).
3. **Data Visualization**:
    - Use libraries like Chart.js or D3.js to visualize data from MySQL queries on the frontend (e.g., bar charts, line graphs).
4. **Search and Filter Functionality**:
    - Provide the user with the ability to search and filter through records (e.g., filtering by date, category, keyword, or id) using MySQL queries in the backend.

- A detailed README document on how to install and execute your project you can use Windows or Linux.
- This is a general description of the assignment; the details are left up to your imagination.
- If you have any questions or anything you need clarification for, please contact the TA promptly with the issue.
