Capstone Integration Project Requirements

The Capstone Integration Project aims to provide students with hands-on experience in real-world software development processes, from gathering requirements to deployment and maintenance. Each student will work on an individual project, but all projects must adhere to the following technical and feature requirements to ensure consistency and alignment with best practices.

Project Deadline: March 21, 2025.

Technical Requirements

1. The project must be a web-based application deployed on public cloud infrastructure (Google Cloud Platform - GCP).
2. The system must consist of three main components built from scratch:
   a. Frontend
   b. Backend
   c. Database
3. The **frontend** must be implemented using **Flask Templates** or **React**.
4. The **backend** must be developed using **Flask** or **FastAPI**.
5. The **database** must be hosted on public cloud infrastructure (e.g., Google Cloud SQL, Firebase, or similar).
6. The application must be deployed and accessible publicly via a secure HTTPS endpoint.
7. CI/CD pipelines must be implemented for automated testing and deployment.
8. The project must include proper technical documentation, covering software requirements specification, system architecture, setup instructions, and API usage.
9. All source code must be version-controlled using Git, and the repository should be hosted on a GitHub.
10. GitHub contributions must be consistent throughout the development process, with meaningful commits made incrementally until the project deadline of **March 21, 2025**.
11. The system should be designed to handle increasing workloads if the user base or data grows.
12. The codebase must follow clean code principles and be modular to allow future updates and maintenance.
13. Unit tests must be implemented to ensure system reliability.

Common Feature Requirements

1. Authentication:

- The system must support user account creation and login functionalities.
- Passwords must be securely hashed and stored.
- Proper session management must be implemented.

2. Data Visualization:

- The system must fetch stored data from the database and present it visually to the user.

Examples include dashboards, charts, or graphs for analyzing data trends.

3. Data Input via Forms:

- The frontend must allow users to input data through forms or similar approaches.
- Submitted data must be validated and added to the database.

4. Data Display:

- The system must retrieve and display stored data in a user-friendly format (e.g., tables, cards, or lists).

5. Error Handling:

- Proper error handling must be implemented for both frontend and backend components.
- User-friendly error messages should be displayed for invalid inputs or system errors.

6. Search and Filtering:

- The system must allow users to search and filter stored data based on relevant criteria.

Grading and Deliverables

Deliverable	Weight	Deadline
Assignment 1	5%	06/02/2025
Assignment 2	10%	10/02/2025
Phase 1 Implementation	20%	19/02/2025
Phase 2 Implementation	20%	10/03/2025
Phase 3 Implementation	40%	21/03/2025
Class participation and attendance	5%	

Expectations for Deliverables:
Each deliverable requires a presentation and walkthrough with the professor. Students must present their documents, designs, code, and project setup. During each presentation, the professor will ask detailed technical questions to assess the originality and depth of understanding behind the deliverables.

Details on deliverables

1. Participation and Class Attendance **(5%)**
2. **Assignment-1**: Project Management **(5%)**
   - Define your project plan on a software project management tool such as Jira (You can use whatever tools you want).
   - The project plan should have milestones that you would like to achieve and the deadlines for each milestone.

3. **Assignment-2**: Software Design (**10%**)
    - SRS **(5%)**
        - **Functional Requirements**
        - **Non-Functional Requirements**
        - **System Requirements**
        - **Acceptance Criteria**
        - **System Features**
            - **User stories**
    - Software Architecture **(5%)**

4. Phase 1: (**20%**)**: Backend and database implementation**
    - Database design **(5%)**
    - Database setup **(5%)**
    - Backend project setup (**5%**)
    - Backend connected with DB **(5%)**
5. Phase 2: **(20%)**: Complete system implementation and deployment
    - All the three main components of the systems are connected (**10%**)
    - CI/CD pipeline setup (**5%**)
    - Project deployed and accessible from the internet (**5%**)
6. Phase 3: (**40%**)**: Complete features implemented and tested**
    - Feature completion (**20%**)
        - Deployed to GCP
        - Includes all bare minimum requirements (To include what features are required)
    - Code quality (**10%**)
    - Continuous development (**5%**)
    - CI/CD (**5%**)