**1. Library Management System**

**Question**:  
Build a **Library Management System** with the following requirements:

1. Design a MySQL database to store book details, user information, and borrow/return records.
2. Create a Flask web app with:
   * A user login system for library members and admins.
   * Functionality to search books, borrow/return books, and add new books (admin only).
3. Implement a feature to generate a borrowing history report using SQL queries.
4. Set up a CI/CD pipeline and deploy the app on a free platform like **Render** or **PythonAnywhere**.
5. Ensure the app uses **environment variables** to securely manage database credentials.

**2. Student Grading System**

**Question**:  
Develop a **Student Grading System** with the following specifications:

1. Design a normalized MySQL database to manage student information, course details, and grades.
2. Build a Flask app with:
   * A portal for students to view grades and instructors to update them.
   * A dashboard to display class performance trends using charts.
3. Use SQL queries to calculate class averages and rank students by GPA.
4. Implement CI/CD with GitHub Actions to automate testing and deployment to **Vercel** or **Render**.
5. Ensure proper error handling in the app for database connectivity issues.

**3. E-Commerce Order Tracking System**

**Question**:  
Create an **E-Commerce Order Tracking System** that meets the following requirements:

1. Use MySQL to store product details, customer information, and order records.
2. Develop a Flask-based web app with:
   * User login for customers and admin functionality for inventory management.
   * Order placement, real-time tracking, and a history of past purchases.
3. Implement dynamic discount calculations using SQL.
4. Set up a CI/CD pipeline with deployment to **Heroku** or **Render**.
5. Secure sensitive data like API keys and database credentials using a .env file.

**4. Personal Finance Tracker**

**Question**:  
Design a **Personal Finance Tracker** with the following functionality:

1. Create a MySQL database to manage income, expenses, and savings records.
2. Develop a Flask app that:
   * Allows users to add/edit/delete income and expense entries.
   * Visualizes monthly expense trends using charts.
3. Use SQL to calculate remaining budgets and predict future savings.
4. Deploy the app on **PythonAnywhere** using a CI/CD pipeline for automated updates.
5. Ensure the app is responsive and supports mobile access.

**5. Hospital Patient Management System**

**Question**:  
Build a **Hospital Patient Management System** with the following requirements:

1. Use MySQL to store patient data, doctor schedules, and treatment logs.
2. Create a Flask web app with:
   * A patient registration form.
   * A doctor portal for appointment scheduling and treatment updates.
3. Implement SQL joins to generate reports on bed availability and common diagnoses.
4. Deploy the app on **Render** using a CI/CD pipeline with GitHub Actions.
5. Add a basic authentication system to secure patient data.

**6. Weather Data Analysis Tool**

**Question**:  
Develop a **Weather Data Analysis Tool** with the following specifications:

1. Design a MySQL database to store daily weather data (temperature, humidity, rainfall, etc.).
2. Build a Flask app with:
   * Features to upload new weather data via a form.
   * A dashboard for visualizing historical weather trends.
3. Use SQL queries to find patterns such as average monthly temperatures or extreme weather events.
4. Implement a CI/CD pipeline to deploy the app on **Vercel** or **Heroku**.
5. Secure database access and API keys with a .env file.

**7. Gym Membership Management System**

**Question**:  
Design a **Gym Membership Management System** with the following requirements:

1. Use MySQL to store member information, payment status, and attendance logs.
2. Create a Flask-based web app with:
   * Member login for viewing attendance and subscription status.
   * Admin dashboard to manage memberships and generate reports.
3. Use SQL to analyze data such as peak gym hours and membership renewals.
4. Deploy the app on **AWS Free Tier** or **PythonAnywhere** using a CI/CD pipeline.
5. Integrate data visualization libraries like Plotly for interactive charts.

**Evaluation Criteria**

1. **MySQL Database Design**: Proper normalization and efficient queries.
2. **Flask App Development**: Functionality, usability, and responsiveness of the web app.
3. **CI/CD Pipeline**: Automation for testing and deployment, ensuring no manual steps are required for updates.
4. **Deployment**: Successful deployment to a free platform, with all components functional.
5. **Security**: Usage of environment variables for sensitive data and implementation of authentication mechanisms.