Game Store

Database Systems

Deadline: [09-May-2025]

Objective:

In this manual, you will develop a Game Store application, simulating an e-commerce platform. The application will have functionalities like user authentication, browsing and purchasing games, managing inventory, and processing orders. Throughout the project, you will work on various components, from database design to implementing core features and testing the final application.

Group Formation:

As mentioned in the Database Lab Outline, you must form a group of 2 (minimum) or 3 (maximum) members to work on the Database project this semester. Add the names of your group members in the comments section of the post under which this document is uploaded on Google Classroom, in the following format by **[20-April-2025]**:

Member Name	Roll#
Member 1 Name	Member 1 Roll#
Member 2 Name	Member 2 Roll#
Member 3 Name	Member 3 Roll#

Table 1: Group Member Information

The following instructions should be strictly followed for making groups:

- Only one member from each group should comment under the Google Classroom post. **No duplicate entries**.
- Cross-section groups are **NOT allowed**, i.e., section A students cannot form a group with section B students and vice versa.

ERD Submission:

- Make an ERD for this project using MS Visio by **[20-April-2025]**

Deliverable 1: Database Design and Setup

Objective:
In this deliverable, you will design and implement the core database structure for the Game Store. This includes setting up all the necessary tables, establishing relationships between them, and ensuring that your database follows best practices for data integrity and performance.

Instructions:

1. **Schema Design:**
   - Design the Game Store database schema to manage essential data such as user accounts, games, orders, payments, and inventory.
   - The schema should cover the following:
     - **Users Table:** Manage customer and admin details. Include fields like user id, username, email, password, and role.
     - **Games Table:** Store information about the games available for purchase. Fields should include game id, title, description, price, genre, and platform.
     - **Orders Table:** Store information about user orders. It should have fields like order_id, user_id, order_date, total_amount, and status.
     - **Payments Table:** Track payment transactions for orders. Include fields such as payment_id, order_id, payment_date, payment_status, and payment_method.
     - **Inventory Table:** Keep track of stock levels for each game. Fields might include game id and stock_quantity.
2. **Database Relationships:**
   - Establish relationships between tables. For example, a Customer can have many Orders, and a Game can appear in many Orders.
3. **Normalization:**
   - Apply normalization principles (up to 3NF) to avoid redundancy in your data.
4. **Query Requirements:**
   - Write SQL queries to create the tables and set up the relationships between them.
   - Implement Joins, Views, Indexes, and Triggers where appropriate. For example, use a trigger to update the inventory when an order is placed.
5. **Submission:**
   - Submit a Word document containing your database schema, queries, and database diagram.
   - Include screenshots showing your work in MySQL or SQL Server Management Studio.

Deliverable 2: Basic Game Store User Interface (UI) Design

Objective:
In this deliverable, you will design the user interface (UI) for the Game Store. This will involve creating interactive forms that allow users to browse games, view details, and manage their cart. The UI should be intuitive and easy to navigate.

Instructions:

1. **UI Components to Implement:**
    - **Home Page**: Display a list of featured games, with options to filter games by category (e.g., Action, Adventure, Strategy).
    - **Product Details Page**: Allow users to view detailed information about a specific game, including its title, description, price, and purchase options (e.g., Add to Cart).
    - **Cart Page**: Display the user's shopping cart, showing the selected games, their quantities, and total price. Provide options to modify quantities or remove items.
    - **Order Summary Page**: Show a summary of the user's order before they proceed to payment. Include the games purchased, the total cost, and the option to confirm or cancel the order.
    - **Login and Registration Page**: Provide a form for new users to sign up and existing users to log in. Include input validation for required fields, valid email formats, and strong password rules.
2. **Responsive Design:**
    - Your UI must be responsive, meaning it should adapt to different screen sizes and user needs.
3. **Submission:**
    - Submit a folder containing all '.cs' or '.html', '.css', '.js' files for the implemented forms.
    - Include screenshots of each page and a detailed Word document explaining the design and functionality of each page.

Deliverable 3: User Authentication and Role-Based Access

Objective:
In this deliverable, you will implement a user authentication system to handle login and signup functionalities. Additionally, you will set up role-based access control (RBAC) to restrict access to certain parts of the store based on the user's role (Customer, Admin).

Instructions:

1. **Authentication System:**
    - Implement a login and signup system for users. Ensure that customers can sign up, log in, and browse games, while admins can manage products and orders.
    - Ensure password security by using hashing methods.
2. **Role-Based Access Control:**
    - Create a system where Customers and Admins have different access levels, with Admins being able to modify inventory and manage orders.
3. **Database Integration:**
    - Ensure that user data is stored securely in the database and implement backend validation using stored procedures for checking credentials.
4. **Submission:**
    - Submit a folder containing all backend code files for user authentication and role-based access.
    - Include screenshots of the login and registration pages and a Word document detailing how the authentication system and role-based access work.

Deliverable 4: Core E-commerce Functionalities (Shopping Cart, Order Processing)

Objective:
In this deliverable, you will implement core e-commerce functionalities like the shopping cart and order processing, allowing users to add games to the cart, place orders, and process payments.

Instructions:

1. **Shopping Cart System:**
    - Implement a system where users can add and remove games from their cart and calculate the total price dynamically.
2. **Order Processing:**
    - After confirming the cart, users should be able to place an order, which will update the Orders table in the database and reflect inventory changes.
3. **Payment Processing:**
    - Simulate a payment process, allowing users to select a payment method and confirming the payment in the Payments table.
4. **Submission:**
    - Submit a folder containing backend code for the shopping cart and order processing.
    - Include screenshots of the cart, order summary, and payment confirmation pages, along with a Word document explaining the process.

Deliverable 5: Final Game Store Application with Testing and Refinements

Objective:
This is the final deliverable, where you will integrate all previous components into a complete Game Store application. You will also perform comprehensive testing to ensure the system functions as intended.

Instructions:

1. **Refine UI and Functionality:**
    - Refine the user interface, making it fully responsive and user-friendly.
2. **Inventory Management (Admin Role):**
    - Allow admins to add, edit, and delete games in the inventory.
3. **Order Tracking (Customer Role):**
    - Allow customers to track the status of their orders (e.g., ”Processing”, ”Shipped”, ”Delivered”).
4. **Testing and Debugging:**
    - Test the system to ensure all features work and fix any issues. Perform load testing to ensure it can handle multiple users.
5. **Submission:**
    - Submit the final project folder containing all application code, screenshots of the final version, and a Word document detailing how the system works.

Good Luck 😊