# Preface

I am very pleased to present my academic project entitled **Gamehub**, which has been developed as part of my course work at **Dr. V.R. Godhaniya I.T. College, Porbandar**, affiliated to **Bhakta Kavi Narsinh Mehta University**. The project is developed using **ASP.NET** as the programming language and **SQL Server** as the database management system. The purpose of this project is to provide a platform where games can be organized, managed, and accessed in a structured and user-friendly manner.

The project has also helped me to apply the theoretical knowledge that I gained in the classroom to a real-world application. I got the opportunity to understand how different technologies can be integrated to create a complete system. It gave me practical exposure to system analysis, designing diagrams such as ER and DFD, and implementing database normalization. This practical learning experience has been one of the most valuable parts of my academic journey.

I would like to express my sincere gratitude to **Dr. Dhaval Kher Sir, Head of Department**, for his continuous guidance, encouragement, and valuable suggestions throughout the preparation of this project. His support and motivation played a major role in the successful completion of this work. I am also grateful to my respected teachers and my college, **Dr. V.R. Godhaniya I.T. College**, for providing me with the proper resources, environment, and opportunities to learn and explore.

Finally, I would like to thank my family, classmates, and friends who directly or indirectly supported me throughout this project. Without their constant encouragement and moral support, this work would not have been possible.

# Project Index

|  |  |  |
| --- | --- | --- |
| Sr. No | Content | Page No. |
| 01. | Certificate | -- |
| 02. | Preface | -- |
| 03. | Index |  |
| 03. | Acknowledgement | -- |
| 04. | Project Profile | -- |
| 05. | Abstract of Project (Brief System Background) | -- |
| 06. | Objective & Scope of a Project | -- |
| 07. | Feasibility Study (Preliminary Investigation) • Operational • Technical • Economic | -- |
| 08. | System Analysis and Designing • Requirement Analysis • Use Case Diagram • Class Diagram (OOP) | -- |
| 09. | Project Life Cycle • ER Diagram • Data Flow Diagram • Activity Diagram | -- |
| 10. | Database Creation • Normalization of Tables (Issues & Solutions) • Data Dictionary | -- |
| 11. | Software Testing | -- |
| 12. | User Interface Designs (Snapshots & Source Code ) | -- |
| 13. | Bibliography | -- |

**Acknowledgement**

I feel very happy to present my project **Gamehub**, which has been developed as part of my academic work at **Dr. V.R. Godhaniya I.T. College, Porbandar**, affiliated to **Bhakta Kavi Narsinh Mehta University**. The project is developed using **ASP.NET** and **SQL Server**.

I would first like to express my deep gratitude to **Dr. Dhaval Kher Sir, Head of Department**, for his valuable guidance, constant encouragement, and helpful suggestions throughout the development of this project. His support and motivation gave me the confidence to complete my work successfully.

I am also thankful to all the faculty members of my college who have always inspired me and shared their knowledge during my studies. Their teachings and guidance have been a strong foundation in building this project.

I would also like to thank my college, **Dr. V.R. Godhaniya I.T. College**, for providing me with the opportunity and resources to complete this project work.

Finally, I would like to express my heartfelt thanks to my family and friends for their continuous support, encouragement, and motivation during this journey. Their belief in me has been a source of strength in completing this project.

**Project Profile**

**Project Title:** Gamehub  
**Developer:** Himanshu Solanki , Manav Joshi , Yash Pathar  
**College:** Dr. V.R. Godhaniya I.T. College, Porbandar  
**University:** Bhakta Kavi Narsinh Mehta University  
**Head of Department:** Dr. Dhaval Kher Sir  
**Languages/Tools Used:** ASP.NET, SQL Server

**Introduction**

The project **Gamehub** is designed as a platform where games can be organized, managed, and accessed easily. In today’s digital world, people are highly engaged with games, and there is a need for a simple system that can handle game-related data in an efficient and structured way. This project focuses on providing such a solution.

**Purpose of the Project**

The main purpose of Gamehub is to create a centralized platform where users can find different games. The project aims to make game management more user-friendly by combining web technologies and database systems.

**Objectives**

1. To design and develop a web-based system using ASP.NET.
2. To store and manage game data securely using SQL Server.
3. To create a user-friendly interface that is simple and easy to use.
4. To apply academic knowledge of system analysis, database design, and project implementation.

**Scope**

The project has been developed as part of the academic requirements of MSc IT. It covers important aspects of software development like system analysis, ER diagrams, DFD, database design, coding, and testing. While the system has been built for learning purposes, it can also be expanded in the future with more features like multiplayer game integration, leaderboards, and community interaction.

**Significance**

Gamehub has provided me with an opportunity to improve my practical knowledge of web development and database management. It has helped me to connect classroom learning with real-world applications and gain confidence in project development.

**Abstract of Project**

The project titled **Gamehub** has been developed as part of my academic curriculum at **Dr. V.R. Godhaniya I.T. College, Porbandar**, under **Bhakta Kavi Narsinh Mehta University**. The project is designed and implemented using **ASP.NET** as the front-end technology and **SQL Server** as the back-end database.

Gamehub is a web-based platform that provides a structured way to manage and access games. In the modern digital world, users often require a centralized system where they can organize and interact with games easily. This project focuses on fulfilling that requirement by building a simple, secure, and user-friendly application.

The system has been designed by following standard software development practices, including system analysis, requirement gathering, database design, and coding. It makes use of features like **ER diagrams, DFD, and normalization** for database efficiency. The project also includes testing and validation to ensure accuracy and reliability.

The main aim of Gamehub is to provide an interactive platform where game information can be stored, updated, and retrieved quickly. It demonstrates how academic knowledge of programming, databases, and system designing can be applied to build a working solution.

Through the development of this project, I have learned how to combine theory with practice and create a real-world application. This project has improved my understanding of **ASP.NET, SQL Server, system analysis, and project documentation**, and has given me valuable experience that will be useful in my future career.

**Objective & Scope of a Project**

**Objectives**

The main objectives of the **Gamehub** project are:

1. **To design and develop a web-based system** using **ASP.NET** as the front-end technology and **SQL Server** as the back-end database.
2. **To provide a centralized platform** for storing, managing, and accessing game information in an organized manner.
3. **To create a user-friendly interface** that makes it easy for users to browse, search, and interact with game-related data.
4. **To implement proper database management techniques** such as normalization and indexing for better performance and efficiency.
5. **To apply system analysis and design methods** like ER diagrams, DFDs, and UML diagrams for structured development.
6. **To test and validate the system** to ensure accuracy, security, and reliability.
7. **To enhance practical knowledge** of web technologies and database systems by applying theoretical concepts to a real-world project.

**Scope**

The scope of the **Gamehub** project includes:

1. **Game Management:**  
   The system will allow storing and retrieving information about different games in a structured database.
2. **Web-based Platform:**  
   As it is developed using ASP.NET, it can be accessed through a web browser and can be deployed on a server for wider use.
3. **Database Integration:**  
   SQL Server has been used to ensure secure, fast, and reliable storage of game data.
4. **User-Friendly Interface:**  
   The system provides simple navigation, clear design, and easy-to-use forms for managing games.
5. **Academic Relevance:**  
   The project has been designed to meet the academic requirements of MSc IT and demonstrates key areas such as system analysis, design, implementation, and testing.
6. **Future Scope:**
   * Additional features like multiplayer integration, user login, leaderboards, and game reviews can be added.
   * The project can be expanded into a complete gaming portal in the future.
   * Cloud integration can be introduced to make the system more scalable.

**Feasibility Study**

Before developing any system, it is important to check whether the project is practical, useful, and cost-effective. A feasibility study helps in understanding if the project can be implemented successfully with the available resources. The **Gamehub** project has been studied under three main aspects: **Operational Feasibility, Technical Feasibility, and Economic Feasibility.**

**1. Operational Feasibility**

Operational feasibility is about checking if the project will work smoothly in the real environment.

* The **Gamehub** system is designed to be simple and user-friendly, so users will not require advanced technical skills to operate it.
* It provides clear navigation and an easy-to-use interface, making it convenient for storing and managing game information.
* The system reduces manual work and organizes data in a better way, which makes it efficient to use.
* Since the project is developed as part of academic requirements, it can be easily demonstrated and used within the college environment.

**2. Technical Feasibility**

Technical feasibility checks whether the project can be developed and run with the available technologies and resources.

* The project uses **ASP.NET** for front-end development, which is a widely used and reliable technology.
* **SQL Server** is used as the database, which ensures secure and efficient data storage and retrieval.
* The required software tools such as Visual Studio and SQL Server are easily available in the college lab.
* The project does not require high-end hardware and can run smoothly on a normal computer system with basic configurations.

**3. Economic Feasibility**

Economic feasibility is about checking if the project is cost-effective.

* Since this project is developed for academic purposes, there are no major costs involved.
* The required software tools (ASP.NET, SQL Server, Visual Studio) are either open-source, educational versions, or already available in the college.
* No extra hardware or expensive resources are needed.
* The main investment is student effort, time, and knowledge, which makes it affordable.

**System Analysis and Designing**

**System Analysis and Designing**

System analysis and design is an important part of software development. It helps to clearly define what the system should do, how it should behave, and how it should be structured. For the project **Gamehub**, the system has been analyzed and designed as follows:

**1. Requirement Analysis**

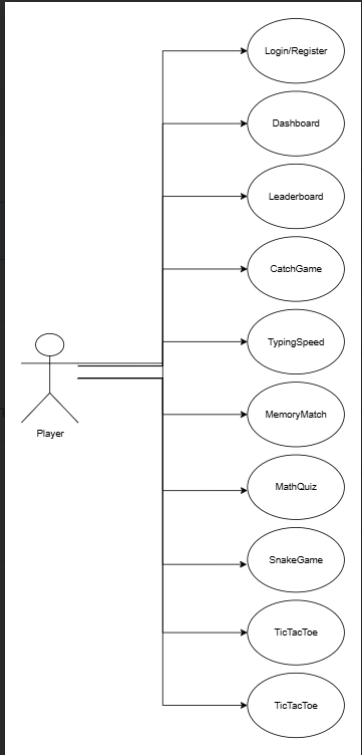
**(a) Functional Requirements**

* The system should allow users to register/login before playing.
* The user can view a list of available games.
* The user can select and play a game from the system.
* The system should store user scores or progress.
* The user can log out after playing.

**(b) Non-Functional Requirements**

* The system should be simple and user-friendly.
* It should provide a smooth and responsive gaming experience.
* The system should be reliable and secure.
* The system should be able to handle multiple users without issues.

**2. Use Case Diagram**



**3. Class Diagram (OOP)**

A class diagram shows how the system is structured using object-oriented concepts. For **Gamehub**, the main classes are:

1. **User**
   * Attributes: UserID, Username, Password, Score
   * Methods: Login(), PlayGame(), ViewScore(), Logout()
2. **Game**
   * Attributes: GameID, GameName, Description, Rules
   * Methods: StartGame(), EndGame(), ShowResult()
3. **Database**
   * Attributes: ConnectionString
   * Methods: Connect(), GetUserData(), GetGameData(), SaveScore()

**Project Life Cycle**

A project life cycle explains the different phases of development, from planning and designing to implementation and testing. For **Gamehub**, the important design models used are **ER Diagram, Data Flow Diagram (DFD), and Activity Diagram**. These diagrams help in understanding how the system works internally and how the user interacts with it.

**1. ER Diagram (Entity–Relationship Diagram)**

The ER Diagram shows how different entities (tables) are related in the database.

**Entities for Gamehub:**

* **User** (UserID, Username, Password, Score)
* **Game** (GameID, GameName, Description, Rules)
* **Score** (ScoreID, UserID, GameID, Points, DatePlayed)

**Relationships:**

* A **User** can play many **Games**.
* A **Game** can be played by many **Users**.
* A **Score** table connects Users and Games to store results.

**2. Data Flow Diagram (DFD)**

A DFD explains how data moves in the system.

**Level 0 (Context Diagram)**

* **User** interacts with the **Gamehub System**.
* User provides login details → System checks database → System allows user to play game → System stores score in database.

**Level 1 (Detailed Flow)**

1. **Login Process:** User enters username/password → System verifies from database.
2. **Game Selection:** User requests game list → System fetches available games from database.
3. **Play Game:** User plays the selected game → System runs game logic.
4. **Score Handling:** After game ends → System saves score into database → User can view score.

**3. Activity Diagram**

An activity diagram shows the step-by-step workflow of the system.

**Database Creation**

The **Gamehub project** uses **SQL Server** as the database. The main purpose of the database is to keep records of users and their game activities in a clear and secure way. To achieve this, the database design is kept simple with only two main tables: **Users** and **GameHistory**.

**Normalization of Tables (Issues & Solutions)**

**Issues Before Normalization:**

If all details about users and games were stored in a single table, then many problems would occur:

* Repeated storage of the same user details for every game played.
* Difficulty in updating or deleting information correctly.
* Higher storage requirement and data redundancy.
* Chances of inconsistency in user information.

**Solutions After Normalization:**

To solve these issues, normalization was applied:

1. **Users Table** is created separately to store user details such as username, email, and password.
2. **GameHistory Table** is created to store details of games played, including game name, score, moves, and date.
3. By linking the data through **Username**, duplication is avoided.

The database design follows **Third Normal Form (3NF)** because:

* Each field has atomic values (1NF).
* Data is divided into separate tables to avoid partial dependency (2NF).
* All non-key attributes depend only on the primary key (3NF).

This makes the database simple, clean, and easy to maintain.

**Data Dictionary**

**1. Users Table**

| **Field Name** | **Data Type** | **Description** |
| --- | --- | --- |
| UserID | INT (Primary Key) | Unique ID for each user |
| Username | VARCHAR | The name chosen by the user |
| Email | VARCHAR | User’s email address |
| PasswordHash | VARCHAR | Encrypted password for security |
| DateCreated | DATETIME | The date and time when the account was created |

**2. GameHistory Table**

| **Field Name** | **Data Type** | **Description** |
| --- | --- | --- |
| Id | INT (Primary Key) | Unique record ID for each game |
| Username | VARCHAR | Username of the player |
| GameName | VARCHAR | Name of the game played |
| Winner | VARCHAR | Winner of the match |
| Moves | INT | Number of moves in the game |
| Score | INT | Score achieved in the game |
| ExtraInfo | VARCHAR | Any additional information about the match |
| PlayedOn | DATETIME | The date and time when the game was played |

**Software Testing**

Software testing is an important step in the development of the **Gamehub project**. It makes sure that the system works correctly, gives the right results, and does not fail when users interact with it. Testing also helps to find errors before the project is delivered.

The Gamehub project is tested using different types of testing methods:

**1. Unit Testing**

* Each small part of the project, like **login module, registration, and game play**, is tested separately.
* For example, checking if the login form accepts only correct usernames and passwords.
* ASP.NET validation and SQL queries were tested step by step.

**2. Integration Testing**

* After testing single modules, they were combined together.
* Example: After registration, the user should be able to log in and then play a game.
* This testing ensured smooth connection between **Users Table** and **GameHistory Table**.

**3. Functional Testing**

* This was done to check if all project functions are working as expected.
* Some tested functions:
  + User can register successfully.
  + User can log in with valid details.
  + User can play a game.
  + Game results (winner, score, moves) are saved properly in the database.

**4. Validation Testing**

* The project was checked to confirm that it meets the requirements set during analysis.
* Example: Only registered users can access the game, and results are stored in history.

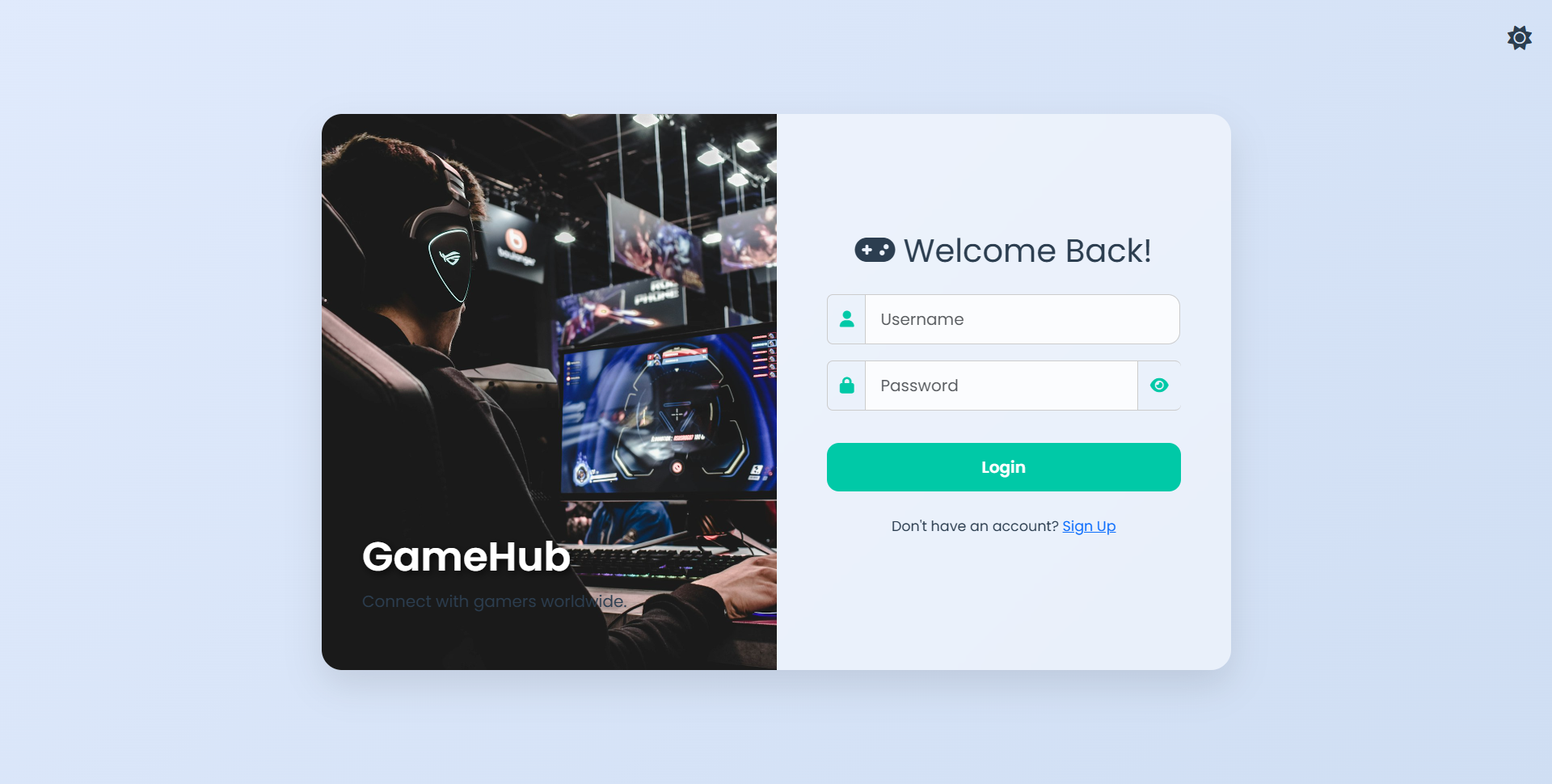
**5. User Acceptance Testing (UAT)**

* The system was tested from the user’s point of view.
* Users played games and checked if the system was easy to use, and if data was saved correctly.
* Feedback was collected and small improvements were made.

**6. Performance Testing**

* The project was tested for speed and performance.
* Example: Checking how fast the game history is stored and retrieved from the database.
* SQL queries were optimized for better performance.

**User Interface Designs (Snapshots)**

****

Login.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="myprj.Login" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<meta charset="UTF-8">

<title>GameHub | Login</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css" rel="stylesheet" />

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap" rel="stylesheet">

<style>

:root {

--bg-dark-start: #0f2027;

--bg-dark-mid: #203a43;

--bg-dark-end: #2c5364;

--text-dark: #f0f0f0;

--primary-accent: #00c9a7;

--primary-accent-hover: #00b09b;

--form-bg-dark: rgba(0, 0, 0, 0.2);

--bg-light-start: #e0eafc;

--bg-light-end: #cfdef3;

--text-light: #2c3e50;

--form-bg-light: rgba(255, 255, 255, 0.5);

}

body {

margin: 0;

font-family: 'Poppins', sans-serif;

background: linear-gradient(135deg, var(--bg-dark-start), var(--bg-dark-mid), var(--bg-dark-end));

color: var(--text-dark);

transition: background 0.5s ease, color 0.5s ease;

overflow: hidden;

}

body.light-mode {

background: linear-gradient(135deg, var(--bg-light-start), var(--bg-light-end));

color: var(--text-light);

}

.login-container {

display: flex;

height: 100vh;

width: 100vw;

align-items: center;

justify-content: center;

}

.login-card {

display: flex;

width: 900px;

max-width: 90vw;

min-height: 550px;

background: var(--form-bg-dark);

border-radius: 20px;

backdrop-filter: blur(15px);

box-shadow: 0 15px 35px rgba(0, 0, 0, 0.3);

overflow: hidden;

animation: fadeIn 1s ease-out;

}

body.light-mode .login-card {

background: var(--form-bg-light);

box-shadow: 0 15px 35px rgba(0, 0, 0, 0.1);

}

.login-branding {

flex-basis: 50%;

background: url('https://images.unsplash.com/photo-1542751371-adc38448a05e?q=80&w=2070') center center/cover;

position: relative;

display: flex;

align-items: flex-end;

padding: 40px;

}

.branding-text h1 {

color: white;

font-weight: 600;

text-shadow: 2px 2px 8px rgba(0,0,0,0.7);

}

.login-form {

flex-basis: 50%;

padding: 50px;

display: flex;

flex-direction: column;

justify-content: center;

}

.form-control, .form-control:focus {

background-color: rgba(255, 255, 255, 0.1);

color: var(--text-dark);

border: 1px solid rgba(255, 255, 255, 0.2);

border-radius: 12px;

padding: 12px 15px;

box-shadow: none;

}

body.light-mode .form-control {

background-color: rgba(255, 255, 255, 0.8);

color: var(--text-light);

border: 1px solid #ccc;

}

.input-group-text {

background-color: transparent;

border: 1px solid rgba(255, 255, 255, 0.2);

border-right: none;

color: var(--primary-accent);

}

body.light-mode .input-group-text {

border: 1px solid #ccc;

border-right: none;

}

.password-toggle {

cursor: pointer;

border-left: none;

}

.btn-login {

background-color: var(--primary-accent);

border: none;

border-radius: 12px;

padding: 12px;

font-weight: 600;

transition: background 0.3s;

width: 100%;

}

.btn-login:hover {

background-color: var(--primary-accent-hover);

}

.divider {

text-align: center;

margin: 20px 0;

color: rgba(255, 255, 255, 0.5);

}

body.light-mode .divider {

color: #888;

}

.btn-social {

width: 48px;

height: 48px;

border-radius: 50%;

font-size: 1.2rem;

margin: 0 5px;

opacity: 0.8;

transition: opacity 0.3s;

}

.btn-social:hover { opacity: 1; }

.btn-google { background-color: #DB4437; color: white; }

.btn-steam { background-color: #1b2838; color: white; }

.btn-twitch { background-color: #9146FF; color: white; }

.toggle-theme {

position: absolute; top: 20px; right: 20px; cursor: pointer; font-size: 1.5rem;

}

@keyframes fadeIn { from { opacity: 0; transform: scale(0.95); } to { opacity: 1; transform: scale(1); } }

@media (max-width: 768px) {

.login-branding { display: none; }

.login-form { flex-basis: 100%; }

.login-card { width: 100%; height: 100vh; border-radius: 0; }

}

</style>

</head>

<body id="pageBody" class="light-mode">

<div class="toggle-theme" onclick="toggleTheme()">

<i id="themeIcon" class="fa-solid fa-sun"></i>

</div>

<form id="form1" runat="server" class="login-container">

<div class="login-card">

<div class="login-branding">

<div class="branding-text">

<h1>GameHub</h1>

<p>Connect with gamers worldwide.</p>

</div>

</div>

<div class="login-form">

<h2 class="text-center mb-4"><i class="fa-solid fa-gamepad me-2"></i>Welcome Back!</h2>

<div class="mb-3 input-group">

<span class="input-group-text"><i class="fa-solid fa-user"></i></span>

<asp:TextBox ID="txtUsername" runat="server" CssClass="form-control" placeholder="Username" />

</div>

<div class="mb-3 input-group">

<span class="input-group-text"><i class="fa-solid fa-lock"></i></span>

<asp:TextBox ID="txtPassword" runat="server" CssClass="form-control" TextMode="Password" placeholder="Password" />

<span class="input-group-text password-toggle" onclick="togglePasswordVisibility()"><i id="passIcon" class="fa-solid fa-eye"></i></span>

</div>

<div class="d-flex justify-content-between align-items-center mb-3">

</div>

<asp:Button ID="btnLogin" runat="server" Text="Login" CssClass="btn btn-login text-white" OnClick="btnLogin\_Click" />

<p class="text-center mt-4 small">

Don't have an account? <a href="Register.aspx">Sign Up</a>

</p>

<asp:Label ID="lblMessage" runat="server" CssClass="d-none"></asp:Label>

</div>

</div>

</form>

<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>

<script>

function toggleTheme() {

const body = document.getElementById("pageBody");

const icon = document.getElementById("themeIcon");

body.classList.toggle("light-mode");

if (body.classList.contains("light-mode")) {

icon.classList.remove("fa-moon");

icon.classList.add("fa-sun");

} else {

icon.classList.remove("fa-sun");

icon.classList.add("fa-moon");

}

}

function togglePasswordVisibility() {

const passwordInput = document.getElementById('<%= txtPassword.ClientID %>');

const passIcon = document.getElementById('passIcon');

if (passwordInput.type === 'password') {

passwordInput.type = 'text';

passIcon.classList.remove('fa-eye');

passIcon.classList.add('fa-eye-slash');

} else {

passwordInput.type = 'password';

passIcon.classList.remove('fa-eye-slash');

passIcon.classList.add('fa-eye');

}

}

window.onload = function () {

var lbl = document.getElementById('<%= lblMessage.ClientID %>');

if (lbl && lbl.innerText.trim() !== "") {

Swal.fire({

icon: 'error',

title: 'Login Failed',

text: lbl.innerText.trim(),

confirmButtonColor: '#3085d6'

});

}

};

</script>

</body>

</html>

Login.aspx.cs

using System;

using System.Data.SqlClient;

using System.Configuration;

using System.Web.UI; // Required for Page

namespace myprj

{

public partial class Login : Page

{

protected void Page\_Load(object sender, EventArgs e)

{

// Optional: Add logic for "Remember Me" cookie on page load

}

protected void btnLogin\_Click(object sender, EventArgs e)

{

string username = txtUsername.Text.Trim();

string password = txtPassword.Text.Trim();

string connectionString = ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

string userHash = null;

int userId = 0;

using (SqlConnection conn = new SqlConnection(connectionString))

{

// IMPORTANT: Select the PasswordHash, not the plain password!

// Also select the UserID to store in the session.

string query = "SELECT UserID, PasswordHash FROM Users WHERE Username = @Username";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@Username", username);

try

{

conn.Open();

using (SqlDataReader reader = cmd.ExecuteReader())

{

if (reader.Read())

{

userId = Convert.ToInt32(reader["UserID"]);

userHash = reader["PasswordHash"].ToString();

}

}

}

catch (Exception ex)

{

// Log the exception (ex.ToString()) to a file or logging service

lblMessage.Text = "An error occurred. Please try again later.";

return; // Stop execution

}

}

// Verify the password using BCrypt

if (userHash != null && BCrypt.Net.BCrypt.Verify(password, userHash))

{

// Passwords match - Login Successful

Session["UserID"] = userId; // Store UserID, it's safer than username

Session["Username"] = username;

// Handle "Remember Me" functionality with a cookie if needed

Response.Redirect("Dashboard.aspx");

}

else

{

// Passwords do not match or user not found

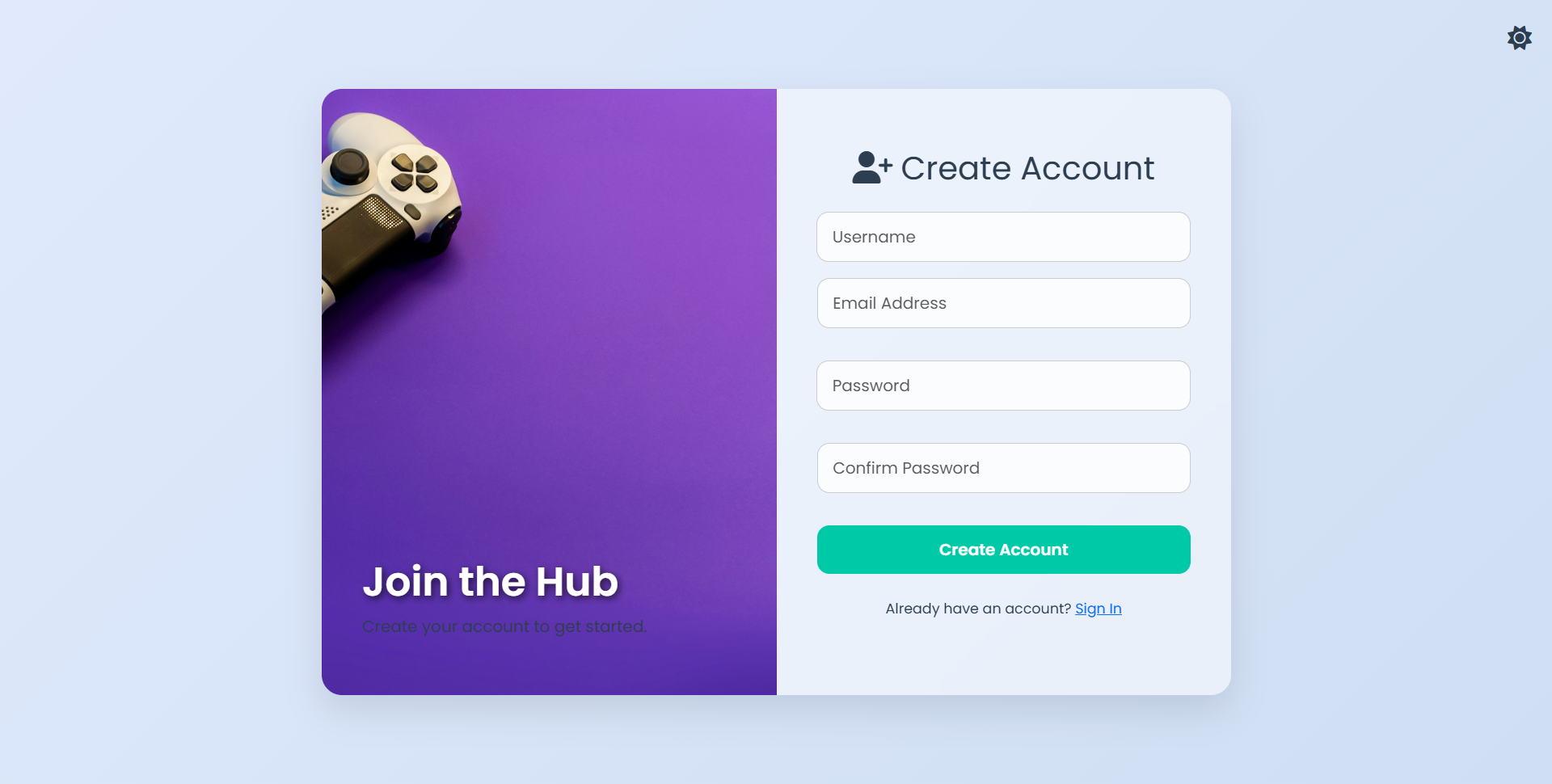
lblMessage.Text = "Invalid username or password.";

}

}

}

}

****

Register.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Register.aspx.cs" Inherits="myprj.Register" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<meta charset="UTF-8">

<title>GameHub | Create Account</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css" rel="stylesheet" />

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>

<style>

/\* This CSS is nearly identical to the redesigned Login page for consistency \*/

:root {

--bg-dark-start: #0f2027; --bg-dark-mid: #203a43; --bg-dark-end: #2c5364;

--text-dark: #f0f0f0; --primary-accent: #00c9a7; --primary-accent-hover: #00b09b;

--form-bg-dark: rgba(0, 0, 0, 0.2); --bg-light-start: #e0eafc; --bg-light-end: #cfdef3;

--text-light: #2c3e50; --form-bg-light: rgba(255, 255, 255, 0.5);

}

body { margin: 0; font-family: 'Poppins', sans-serif; background: linear-gradient(135deg, var(--bg-dark-start), var(--bg-dark-mid), var(--bg-dark-end)); color: var(--text-dark); transition: all 0.5s ease; overflow: hidden; }

body.light-mode { background: linear-gradient(135deg, var(--bg-light-start), var(--bg-light-end)); color: var(--text-light); }

.register-container { display: flex; height: 100vh; width: 100vw; align-items: center; justify-content: center; }

.register-card { display: flex; width: 900px; max-width: 90vw; min-height: 600px; background: var(--form-bg-dark); border-radius: 20px; backdrop-filter: blur(15px); box-shadow: 0 15px 35px rgba(0, 0, 0, 0.3); overflow: hidden; animation: fadeIn 1s ease-out; }

body.light-mode .register-card { background: var(--form-bg-light); box-shadow: 0 15px 35px rgba(0, 0, 0, 0.1); }

.register-branding { flex-basis: 50%; background: url('https://images.unsplash.com/photo-1580234811497-9df7fd2f357e?q=80&w=1964') center center/cover; position: relative; display: flex; align-items: flex-end; padding: 40px; }

.branding-text h1 { color: white; font-weight: 600; text-shadow: 2px 2px 8px rgba(0,0,0,0.7); }

.register-form { flex-basis: 50%; padding: 40px; display: flex; flex-direction: column; justify-content: center; }

.form-control, .form-control:focus { background-color: rgba(255, 255, 255, 0.1); color: var(--text-dark); border: 1px solid rgba(255, 255, 255, 0.2); border-radius: 12px; padding: 12px 15px; box-shadow: none; }

body.light-mode .form-control { background-color: rgba(255, 255, 255, 0.8); color: var(--text-light); border: 1px solid #ccc; }

.btn-register { background-color: var(--primary-accent); border: none; border-radius: 12px; padding: 12px; font-weight: 600; transition: background 0.3s; width: 100%; }

.btn-register:hover { background-color: var(--primary-accent-hover); }

.toggle-theme { position: absolute; top: 20px; right: 20px; cursor: pointer; font-size: 1.5rem; }

.validation-error { color: #ff7b7b; font-size: 0.8rem; height: 1rem; margin-top: -10px; margin-bottom: 10px; visibility: hidden; }

@keyframes fadeIn { from { opacity: 0; transform: scale(0.95); } to { opacity: 1; transform: scale(1); } }

@media (max-width: 768px) { .register-branding { display: none; } .register-form { flex-basis: 100%; } .register-card { width: 100%; height: 100vh; border-radius: 0; } }

</style>

</head>

<body id="pageBody" class="light-mode">

<div class="toggle-theme" onclick="toggleTheme()">

<i id="themeIcon" class="fa-solid fa-sun"></i>

</div>

<form id="form1" runat="server" class="register-container" onsubmit="return validateForm();">

<div class="register-card">

<div class="register-branding">

<div class="branding-text">

<h1>Join the Hub</h1>

<p>Create your account to get started.</p>

</div>

</div>

<div class="register-form">

<h2 class="text-center mb-4"><i class="fa-solid fa-user-plus me-2"></i>Create Account</h2>

<div class="mb-3">

<asp:TextBox ID="txtUsername" runat="server" CssClass="form-control" placeholder="Username" />

</div>

<div class="mb-3">

<asp:TextBox ID="txtEmail" runat="server" CssClass="form-control" TextMode="Email" placeholder="Email Address" />

<div id="emailError" class="validation-error">Invalid email format.</div>

</div>

<div class="mb-3">

<asp:TextBox ID="txtPassword" runat="server" CssClass="form-control" TextMode="Password" placeholder="Password" />

<div id="passwordError" class="validation-error">Password must be at least 8 characters.</div>

</div>

<div class="mb-3">

<asp:TextBox ID="txtConfirmPassword" runat="server" CssClass="form-control" TextMode="Password" placeholder="Confirm Password" />

<div id="confirmPasswordError" class="validation-error">Passwords do not match.</div>

</div>

<asp:Button ID="btnRegister" runat="server" Text="Create Account" CssClass="btn btn-register text-white" OnClick="btnRegister\_Click" />

<p class="text-center mt-4 small">

Already have an account? <a href="Login.aspx">Sign In</a>

</p>

<%-- These labels are used by the server to send messages to SweetAlert --%>

<asp:Label ID="lblMessage" runat="server" CssClass="d-none"></asp:Label>

<asp:Label ID="lblSuccess" runat="server" CssClass="d-none"></asp:Label>

</div>

</div>

</form>

<script>

// Theme toggle function

function toggleTheme() {

const body = document.getElementById("pageBody");

const icon = document.getElementById("themeIcon");

body.classList.toggle("light-mode");

if (body.classList.contains("light-mode")) {

icon.classList.remove("fa-moon");

icon.classList.add("fa-sun");

} else {

icon.classList.remove("fa-sun");

icon.classList.add("fa-moon");

}

}

// Client-side form validation

function validateForm() {

const email = document.getElementById('<%= txtEmail.ClientID %>').value;

const password = document.getElementById('<%= txtPassword.ClientID %>').value;

const confirmPass = document.getElementById('<%= txtConfirmPassword.ClientID %>').value;

let isValid = true;

// Basic email regex

const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

if (!emailRegex.test(email)) {

document.getElementById('emailError').style.visibility = 'visible';

isValid = false;

} else {

document.getElementById('emailError').style.visibility = 'hidden';

}

// Password length

if (password.length < 8) {

document.getElementById('passwordError').style.visibility = 'visible';

isValid = false;

} else {

document.getElementById('passwordError').style.visibility = 'hidden';

}

// Password match

if (password !== confirmPass) {

document.getElementById('confirmPasswordError').style.visibility = 'visible';

isValid = false;

} else {

document.getElementById('confirmPasswordError').style.visibility = 'hidden';

}

return isValid;

}

// Show SweetAlert messages if set from server

window.onload = function () {

const err = document.getElementById('<%= lblMessage.ClientID %>').innerText;

const success = document.getElementById('<%= lblSuccess.ClientID %>').innerText;

if (err && err.trim()) {

Swal.fire({ icon: 'error', title: 'Registration Failed', text: err });

}

if (success && success.trim()) {

Swal.fire({

icon: 'success',

title: 'Registration Successful!',

text: success,

timer: 2000,

showConfirmButton: false

}).then(() => {

window.location.href = 'Login.aspx'; // Redirect after success

});

}

};

</script>

</body>

</html>

Register.aspx.cs

using System;

using System.Configuration;

using System.Data.SqlClient;

using System.Text.RegularExpressions;

using System.Web.UI;

namespace myprj

{

public partial class Register : Page

{

protected void Page\_Load(object sender, EventArgs e)

{

// No action needed on page load for this form

}

protected void btnRegister\_Click(object sender, EventArgs e)

{

string username = txtUsername.Text.Trim();

string email = txtEmail.Text.Trim();

string password = txtPassword.Text; // No trim on password

string confirmPassword = txtConfirmPassword.Text;

// --- Server-Side Validation ---

if (string.IsNullOrWhiteSpace(username) || string.IsNullOrWhiteSpace(email) || string.IsNullOrWhiteSpace(password))

{

lblMessage.Text = "All fields are required.";

return;

}

if (!Regex.IsMatch(email, @"^[^@\s]+@[^@\s]+\.[^@\s]+$", RegexOptions.IgnoreCase))

{

lblMessage.Text = "Please enter a valid email address.";

return;

}

if (password.Length < 8)

{

lblMessage.Text = "Password must be at least 8 characters long.";

return;

}

if (password != confirmPassword)

{

lblMessage.Text = "Passwords do not match.";

return;

}

string connectionString = ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

using (var conn = new SqlConnection(connectionString))

{

try

{

conn.Open();

// Check if username or email already exists in one query

string checkUserQuery = "SELECT COUNT(1) FROM Users WHERE Username = @Username OR Email = @Email";

using (var checkCmd = new SqlCommand(checkUserQuery, conn))

{

checkCmd.Parameters.AddWithValue("@Username", username);

checkCmd.Parameters.AddWithValue("@Email", email);

int userExists = (int)checkCmd.ExecuteScalar();

if (userExists > 0)

{

lblMessage.Text = "A user with that username or email already exists.";

return;

}

}

// --- Hash the Password using BCrypt ---

// The work factor of 12 is a good balance of security and performance.

string hashedPassword = BCrypt.Net.BCrypt.HashPassword(password, 12);

// Insert the new user with the HASHED password

string insertQuery = "INSERT INTO Users (Username, Email, PasswordHash, DateCreated) VALUES (@Username, @Email, @PasswordHash, @DateCreated)";

using (var insertCmd = new SqlCommand(insertQuery, conn))

{

insertCmd.Parameters.AddWithValue("@Username", username);

insertCmd.Parameters.AddWithValue("@Email", email);

insertCmd.Parameters.AddWithValue("@PasswordHash", hashedPassword);

insertCmd.Parameters.AddWithValue("@DateCreated", DateTime.Now);

insertCmd.ExecuteNonQuery();

}

// Set success message for SweetAlert to display

lblSuccess.Text = "You can now log in with your new account.";

}

catch (Exception ex)

{

// In a real app, you should log this exception (ex.ToString())

lblMessage.Text = "An unexpected error occurred. Please try again later.";

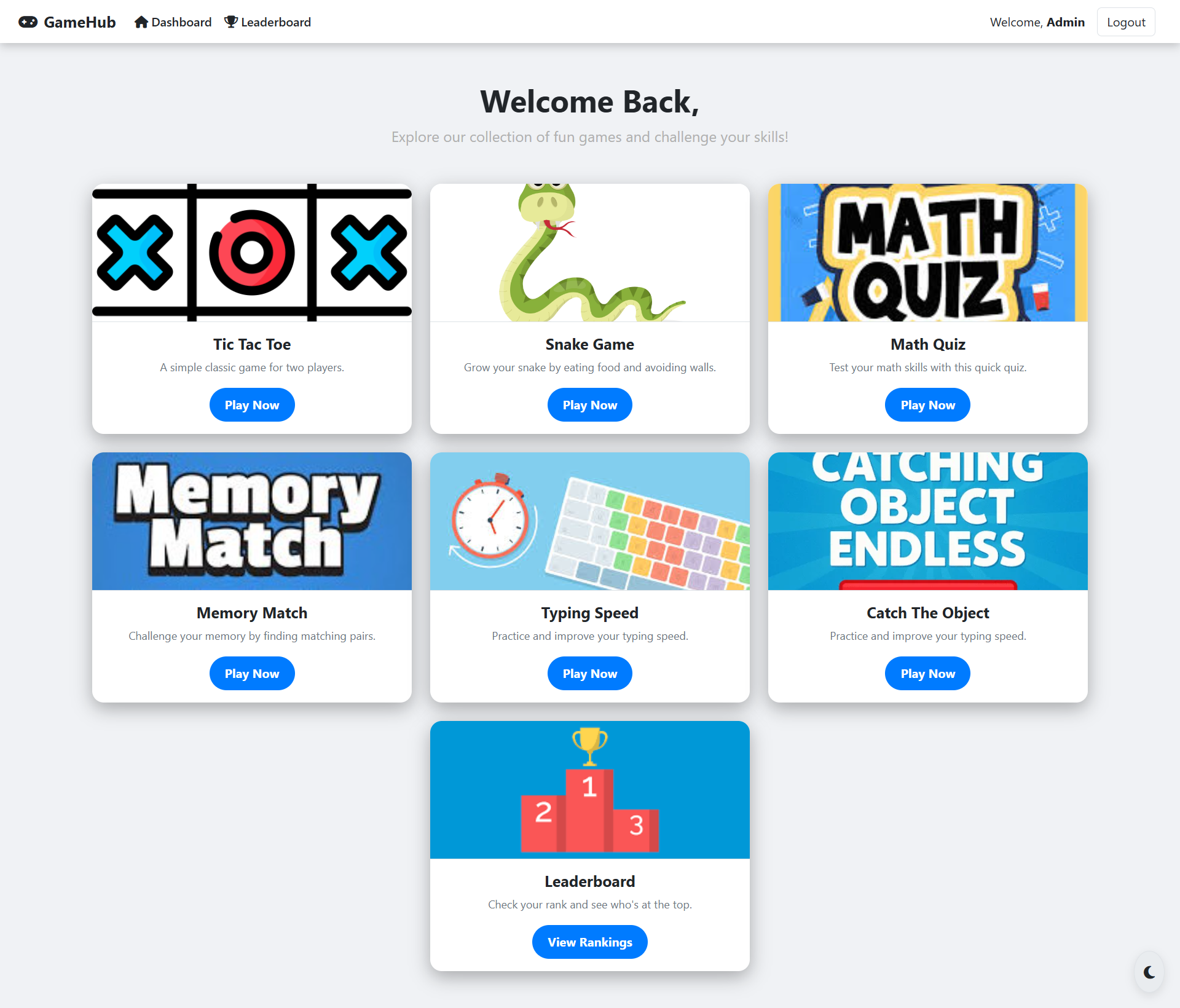
}

}

}

}

}

****

Dashboard.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Dashboard.aspx.cs" Inherits="myprj.Dashboard" %>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>GameHub | Dashboard</title>

<link rel="icon" href="data:image/svg+xml,<svg xmlns=%22http://www.w3.org/2000/svg%22 viewBox=%220 0 100 100%22><text y=%22.9em%22 font-size=%2290%22>🎮</text></svg>">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css" rel="stylesheet">

<style>

/\*

\* ------------------------

\* THEME VARIABLES

\* ------------------------

\*/

:root {

--bg-main-dark: #121212;

--bg-secondary-dark: #1e1e1e;

--card-bg-dark: #282828;

--text-primary-dark: #e0e0e0;

--text-secondary-dark: #b0b0b0;

--accent-dark: #00e676;

--hover-dark: #00c853;

--border-dark: #444;

--bg-main-light: #f0f2f5;

--bg-secondary-light: #ffffff;

--card-bg-light: #e9ecef;

--text-primary-light: #212529;

--text-secondary-light: #6c757d;

--accent-light: #007bff;

--hover-light: #0056b3;

--border-light: #dee2e6;

}

/\*

\* ------------------------

\* GLOBAL STYLES

\* ------------------------

\*/

body {

font-family: 'Segoe UI', sans-serif;

background-color: var(--bg-main-dark);

color: var(--text-primary-dark);

transition: background-color 0.3s, color 0.3s;

overflow-x: hidden;

}

.light-mode {

background-color: var(--bg-main-light);

color: var(--text-primary-light);

}

.light-mode .navbar, .light-mode .card, .light-mode .form-control {

background-color: var(--bg-secondary-light) !important;

border-color: var(--border-light) !important;

}

.light-mode .navbar-brand, .light-mode .nav-link, .light-mode h1, .light-mode h5 {

color: var(--text-primary-light) !important;

}

.light-mode .text-muted {

color: var(--text-secondary-light) !important;

}

.light-mode .btn-outline-secondary {

color: var(--text-primary-light);

border-color: var(--border-light);

}

.light-mode .btn-outline-secondary:hover {

background-color: var(--accent-light);

color: #fff;

}

.light-mode a {

color: var(--text-primary-light);

}

a {

text-decoration: none;

color: var(--text-primary-dark);

}

/\*

\* ------------------------

\* NAVBAR STYLES

\* ------------------------

\*/

.navbar {

background-color: var(--bg-secondary-dark);

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.2);

transition: background-color 0.3s;

}

.navbar-brand {

font-weight: bold;

color: var(--accent-dark);

transition: color 0.3s;

}

.nav-link {

font-weight: 500;

color: var(--text-primary-dark);

transition: color 0.3s;

}

.nav-link:hover {

color: var(--accent-dark);

}

.btn-outline-secondary {

color: var(--text-primary-dark);

border-color: var(--text-primary-dark);

transition: all 0.3s;

}

.btn-outline-secondary:hover {

background-color: var(--accent-dark);

color: var(--bg-secondary-dark);

border-color: var(--accent-dark);

}

/\*

\* ------------------------

\* DASHBOARD CONTENT

\* ------------------------

\*/

.dashboard-header {

margin-top: 50px;

margin-bottom: 30px;

text-align: center;

}

.welcome-title {

font-size: 2.5rem;

font-weight: bold;

color: var(--accent-dark);

animation: fadeInDown 1.2s ease;

}

.welcome-subtitle {

font-size: 1.2rem;

color: var(--text-secondary-dark);

animation: fadeIn 2s ease-in;

}

.game-card {

background-color: var(--card-bg-dark);

border: none;

border-radius: 15px;

box-shadow: 0 8px 24px rgba(0, 0, 0, 0.3);

transition: transform 0.3s ease, box-shadow 0.3s ease;

height: 100%;

animation: cardFloat 0.6s ease forwards;

overflow: hidden;

}

.game-card:hover {

transform: translateY(-10px) scale(1.02);

box-shadow: 0 16px 40px rgba(0, 230, 118, 0.4);

}

.card-img-top {

height: 180px;

object-fit: cover;

border-bottom: 1px solid var(--border-dark);

}

.light-mode .card-img-top {

border-bottom: 1px solid var(--border-light);

}

.card-body h5 {

font-weight: bold;

color: var(--text-primary-dark);

transition: color 0.3s;

}

.card-body p {

font-size: 0.9rem;

color: var(--text-secondary-dark);

transition: color 0.3s;

}

.play-btn {

background-color: var(--accent-dark);

color: var(--bg-main-dark);

border: none;

font-weight: bold;

padding: 10px 20px;

border-radius: 50px;

transition: background-color 0.3s, transform 0.3s;

}

.play-btn:hover {

background-color: var(--hover-dark);

transform: translateY(-2px);

}

.light-mode .play-btn {

background-color: var(--accent-light);

color: #fff;

}

.light-mode .play-btn:hover {

background-color: var(--hover-light);

}

/\*

\* ------------------------

\* THEME TOGGLE

\* ------------------------

\*/

.theme-toggle {

position: fixed;

bottom: 20px;

right: 20px;

background: var(--card-bg-dark);

border: 1px solid var(--border-dark);

border-radius: 50%;

padding: 12px;

font-size: 1.2rem;

cursor: pointer;

color: var(--text-primary-dark);

transition: all 0.3s ease;

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.3);

z-index: 100;

}

.theme-toggle:hover {

transform: scale(1.1) rotate(20deg);

}

.light-mode .theme-toggle {

background: var(--card-bg-light);

color: var(--text-primary-light);

border: 1px solid var(--border-light);

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);

}

/\*

\* ------------------------

\* ANIMATIONS

\* ------------------------

\*/

@keyframes fadeInDown {

from { opacity: 0; transform: translateY(-20px); }

to { opacity: 1; transform: translateY(0); }

}

@keyframes fadeIn {

from { opacity: 0; }

to { opacity: 1; }

}

@keyframes cardFloat {

from { opacity: 0; transform: translateY(20px); }

to { opacity: 1; transform: translateY(0); }

}

</style>

</head>

<body>

<form id="form1" runat="server">

<nav class="navbar navbar-expand-lg">

<div class="container-fluid px-4">

<a class="navbar-brand" href="#">

<i class="fas fa-gamepad me-2"></i>GameHub

</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNav">

<ul class="navbar-nav me-auto mb-2 mb-lg-0">

<li class="nav-item">

<a class="nav-link active" aria-current="page" href="#"><i class="fas fa-home me-1"></i>Dashboard</a>

</li>

<li class="nav-item">

<a class="nav-link" href="Leaderboard.aspx"><i class="fas fa-trophy me-1"></i>Leaderboard</a>

</li>

</ul>

<div class="d-flex align-items-center">

<span class="me-3 d-none d-lg-block">Welcome, <asp:Label ID="lblUser" runat="server" CssClass="fw-bold" /></span>

<asp:Button ID="btnLogout" runat="server" Text="Logout" CssClass="btn btn-outline-secondary me-2" OnClick="btnLogout\_Click" />

</div>

</div>

</div>

</nav>

<div class="dashboard-header text-center">

<h1 class="welcome-title">Welcome Back, <asp:Label ID="lblUserHeader" runat="server" /></h1>

<p class="welcome-subtitle">Explore our collection of fun games and challenge your skills!</p>

</div>

<div class="container my-5">

<div class="row g-4 justify-content-center">

<div class="col-lg-4 col-md-6 col-sm-10">

<a href="TicTacToe.aspx">

<div class="card game-card">

<img src="Images/tic-tac-toe.png" class="card-img-top" alt="Tic Tac Toe">

<div class="card-body text-center">

<h5 class="card-title">Tic Tac Toe</h5>

<p class="card-text text-muted">A simple classic game for two players.</p>

<button type="button" class="btn play-btn">Play Now</button>

</div>

</div>

</a>

</div>

<div class="col-lg-4 col-md-6 col-sm-10">

<a href="SnakeGame.aspx">

<div class="card game-card">

<img src="Images/snackgame.png" class="card-img-top" alt="Snake Game">

<div class="card-body text-center">

<h5 class="card-title">Snake Game</h5>

<p class="card-text text-muted">Grow your snake by eating food and avoiding walls.</p>

<button type="button" class="btn play-btn">Play Now</button>

</div>

</div>

</a>

</div>

<div class="col-lg-4 col-md-6 col-sm-10">

<a href="MathQuiz.aspx">

<div class="card game-card">

<img src="Images/mathquiz.jpg" class="card-img-top" alt="Math Quiz">

<div class="card-body text-center">

<h5 class="card-title">Math Quiz</h5>

<p class="card-text text-muted">Test your math skills with this quick quiz.</p>

<button type="button" class="btn play-btn">Play Now</button>

</div>

</div>

</a>

</div>

<div class="col-lg-4 col-md-6 col-sm-10">

<a href="MemoryMatch.aspx">

<div class="card game-card">

<img src="Images/memory-match.jpg" class="card-img-top" alt="Memory Match">

<div class="card-body text-center">

<h5 class="card-title">Memory Match</h5>

<p class="card-text text-muted">Challenge your memory by finding matching pairs.</p>

<button type="button" class="btn play-btn">Play Now</button>

</div>

</div>

</a>

</div>

<div class="col-lg-4 col-md-6 col-sm-10">

<a href="TypingSpeed.aspx">

<div class="card game-card">

<img src="Images/typingspeed.jpg" class="card-img-top" alt="Typing Speed">

<div class="card-body text-center">

<h5 class="card-title">Typing Speed</h5>

<p class="card-text text-muted">Practice and improve your typing speed.</p>

<button type="button" class="btn play-btn">Play Now</button>

</div>

</div>

</a>

</div>

<div class="col-lg-4 col-md-6 col-sm-10">

<a href="CatchGame.aspx">

<div class="card game-card">

<img src="Images/catch\_object.png" class="card-img-top" alt="Typing Speed">

<div class="card-body text-center">

<h5 class="card-title">Catch The Object</h5>

<p class="card-text text-muted">Practice and improve your typing speed.</p>

<button type="button" class="btn play-btn">Play Now</button>

</div>

</div>

</a>

</div>

<div class="col-lg-4 col-md-6 col-sm-10">

<a href="Leaderboard.aspx">

<div class="card game-card">

<img src="Images/leaderboard.png" class="card-img-top" alt="Leaderboard">

<div class="card-body text-center">

<h5 class="card-title">Leaderboard</h5>

<p class="card-text text-muted">Check your rank and see who's at the top.</p>

<button type="button" class="btn play-btn">View Rankings</button>

</div>

</div>

</a>

</div>

</div>

</div>

</form>

<button type="button" class="theme-toggle" id="themeToggle" aria-label="Toggle Theme">

<i id="themeIcon" class="fas fa-sun"></i>

</button>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

<script>

const toggleButton = document.getElementById('themeToggle');

const icon = document.getElementById('themeIcon');

const body = document.body;

// Load theme from localStorage

const currentTheme = localStorage.getItem('theme');

if (currentTheme) {

body.classList.add(currentTheme + '-mode');

icon.classList.remove('fa-sun', 'fa-moon');

icon.classList.add(currentTheme === 'light' ? 'fa-moon' : 'fa-sun');

}

toggleButton.addEventListener('click', () => {

const isLight = body.classList.contains('light-mode');

// Toggle classes on the body

body.classList.toggle('light-mode', !isLight);

body.classList.toggle('dark-mode', isLight);

// Toggle classes on the icon

icon.classList.toggle('fa-sun', isLight);

icon.classList.toggle('fa-moon', !isLight);

// Save preference to localStorage

localStorage.setItem('theme', isLight ? 'dark' : 'light');

});

</script>

</body>

</html>  
Dashboard.aspx.cs

using System;

namespace myprj

{

public partial class Dashboard : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Username"] == null)

{

Response.Redirect("Login.aspx");

}

else

{

string fullEmail = Session["Username"].ToString();

// Take only the part before '@'

string displayName = fullEmail.Split('@')[0];

// Replace underscores and dots with spaces

displayName = displayName.Replace("\_", " ").Replace(".", " ");

// Capitalize first letter if not empty

if (!string.IsNullOrEmpty(displayName))

{

displayName = char.ToUpper(displayName[0]) + displayName.Substring(1);

}

lblUser.Text = displayName;

}

}

protected void btnLogout\_Click(object sender, EventArgs e)

{

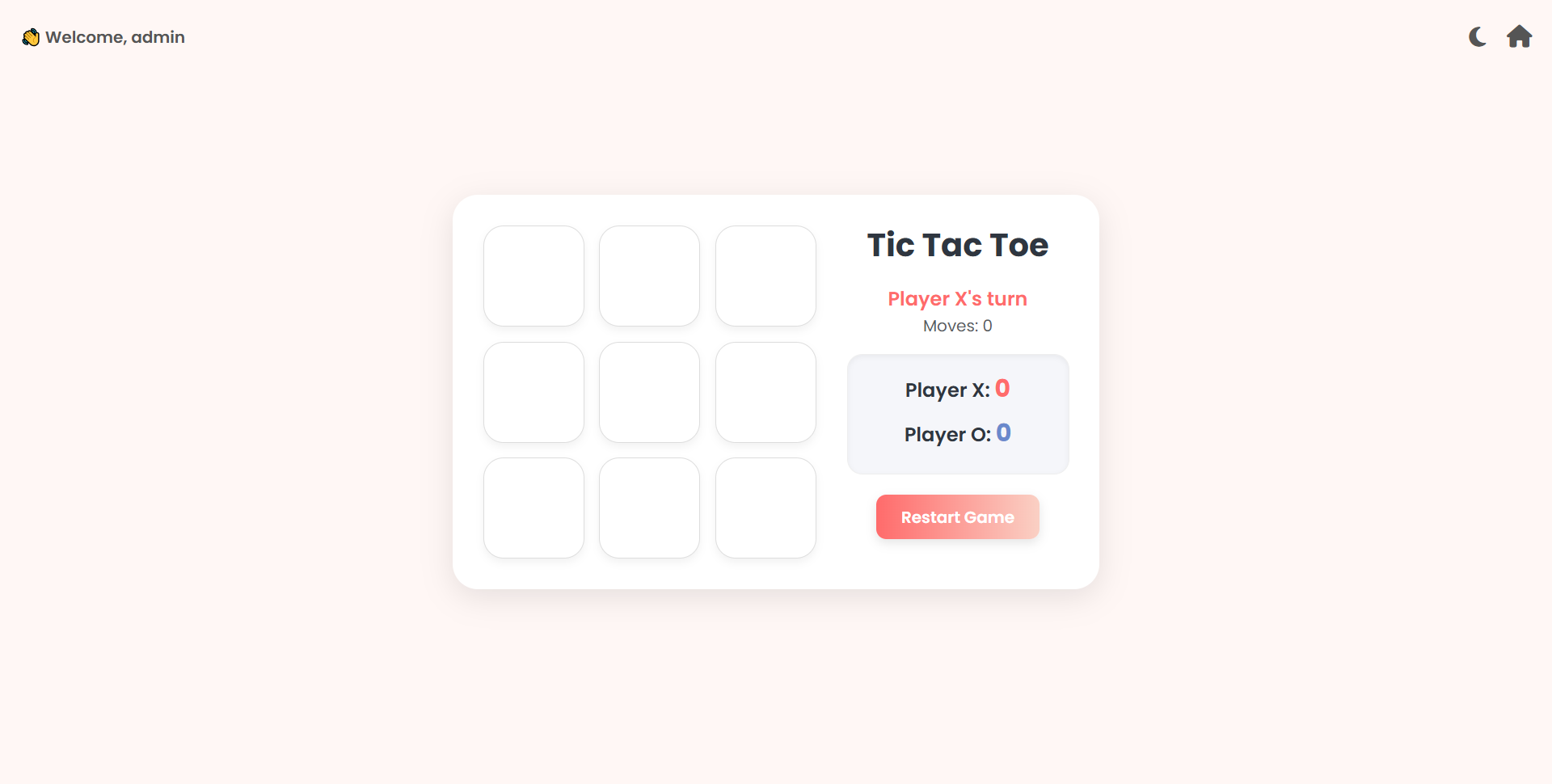
Session.Abandon();

Response.Redirect("Login.aspx");

}

}

}

****

TicTacToe.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="TicTacToe.aspx.cs" Inherits="myprj.TicTacToe" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>GameHub | Tic Tac Toe</title>

<meta name="viewport" content="width=device-width, initial-scale=1" />

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600;700&display=swap" rel="stylesheet" />

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css" />

<style>

:root {

--bg-light: #fff7f5;

--bg-dark: #1c1c1c;

--card-bg-light: #ffffff;

--card-bg-dark: #2e2e2e;

--text-light: #2f3640;

--text-dark: #f0f0f0;

--score-x: #ff6b6b;

--score-o: #6a89cc;

}

body {

font-family: 'Poppins', sans-serif;

min-height: 100vh;

margin: 0;

padding: 0;

display: flex;

align-items: center;

justify-content: center;

background: var(--bg-light);

color: var(--text-light);

transition: background 0.3s, color 0.3s;

}

.dark-mode {

background: var(--bg-dark);

color: var(--text-dark);

}

.game-wrapper {

background: var(--card-bg-light);

border-radius: 25px;

box-shadow: 0 8px 30px rgba(0, 0, 0, 0.1);

padding: 30px;

width: 100%;

max-width: 900px;

display: flex;

flex-wrap: wrap;

gap: 30px;

justify-content: center;

align-items: flex-start;

transition: background 0.3s, box-shadow 0.3s;

}

.dark-mode .game-wrapper {

background: var(--card-bg-dark);

box-shadow: 0 8px 30px rgba(0, 0, 0, 0.3);

}

/\* Game Board \*/

.board {

display: grid;

grid-template-columns: repeat(3, 1fr);

gap: 15px;

width: fit-content;

}

.btn-tile {

width: 100px;

height: 100px;

background-color: var(--card-bg-light);

border-radius: 20px;

border: 1px solid #ddd;

font-size: 2.5rem;

color: var(--text-light);

box-shadow: 0 4px 8px rgba(0,0,0,0.05);

transition: transform 0.2s ease, box-shadow 0.2s ease, background-color 0.3s, color 0.3s;

}

.dark-mode .btn-tile {

background-color: #3e3e3e;

color: var(--text-dark);

border-color: #555;

}

.btn-tile:hover:not(:disabled) {

transform: scale(1.05);

box-shadow: 0 6px 12px rgba(0,0,0,0.15);

}

.btn-tile:disabled {

opacity: 0.8;

cursor: not-allowed;

}

/\* Info Panel \*/

.info-panel {

min-width: 220px;

text-align: center;

}

h2 {

font-weight: bold;

margin-bottom: 20px;

color: var(--text-light);

transition: color 0.3s;

}

.dark-mode h2 { color: var(--text-dark); }

.status-text {

font-size: 1.2rem;

margin-bottom: 10px;

font-weight: 600;

}

.player-x-turn { color: var(--score-x); }

.player-o-turn { color: var(--score-o); }

.win-message { color: #27ae60; }

.draw-message { color: #f39c12; }

.score-box {

margin-top: 15px;

background-color: #f5f6fa;

border: 1px solid #eee;

padding: 15px 20px;

border-radius: 15px;

box-shadow: inset 0 2px 5px rgba(0,0,0,0.05);

transition: background-color 0.3s, border-color 0.3s;

}

.dark-mode .score-box {

background-color: #333;

border-color: #555;

}

.score-item {

font-weight: 600;

font-size: 1.2rem;

margin-bottom: 8px;

}

.score-value { font-size: 1.5rem; font-weight: bold; }

.score-value.x { color: var(--score-x); }

.score-value.o { color: var(--score-o); }

.btn-restart {

margin-top: 20px;

background: linear-gradient(to right, var(--score-x), #fad0c4);

border: none;

padding: 10px 25px;

font-weight: 600;

border-radius: 10px;

color: white;

box-shadow: 0 4px 10px rgba(0,0,0,0.1);

transition: transform 0.2s, box-shadow 0.2s;

}

.btn-restart:hover {

transform: translateY(-2px);

box-shadow: 0 6px 15px rgba(0,0,0,0.15);

}

/\* Controls at top-right \*/

.top-right-controls {

position: absolute;

top: 20px;

right: 20px;

display: flex;

gap: 15px;

z-index: 1000;

}

.icon-btn {

background: none;

border: none;

font-size: 22px;

color: #555;

cursor: pointer;

transition: color 0.3s, transform 0.2s;

}

.icon-btn:hover {

color: #007bff;

transform: scale(1.2);

}

.dark-mode .icon-btn { color: #ddd; }

.rotated { transform: rotate(180deg); }

/\* Welcome label \*/

.welcome-label {

position: absolute;

top: 25px;

left: 20px;

font-weight: 600;

color: #555;

}

.dark-mode .welcome-label { color: #ddd; }

</style>

</head>

<body>

<form id="form1" runat="server">

<!-- Top-right controls -->

<div class="top-right-controls">

<!-- Theme toggle -->

<button type="button" class="icon-btn" onclick="toggleTheme()">

<i id="themeIcon" class="fas fa-moon"></i>

</button>

<!-- Dashboard -->

<a href="Dashboard.aspx" class="icon-btn">

<i class="fas fa-home"></i>

</a>

</div>

<!-- Welcome text (top-left) -->

<asp:Label ID="lblWelcome" runat="server" CssClass="welcome-label"></asp:Label>

<div class="game-wrapper">

<div class="board">

<asp:Button ID="btn00" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn01" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn02" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn10" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn11" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn12" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn20" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn21" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

<asp:Button ID="btn22" runat="server" CssClass="btn-tile" OnClick="Cell\_Click" />

</div>

<div class="info-panel">

<h2>Tic Tac Toe</h2>

<asp:Label ID="lblStatus" runat="server" CssClass="status-text"></asp:Label><br />

<asp:Label ID="lblMoves" runat="server" CssClass="text-muted"></asp:Label>

<div class="score-box mt-3">

<div class="score-item">Player X: <asp:Label ID="lblScoreX" runat="server" CssClass="score-value x"></asp:Label></div>

<div class="score-item">Player O: <asp:Label ID="lblScoreO" runat="server" CssClass="score-value o"></asp:Label></div>

</div>

<asp:Button ID="btnRestart" runat="server" Text="Restart Game" CssClass="btn-restart" OnClick="btnRestart\_Click" />

</div>

</div>

</form>

<script>

function toggleTheme() {

const body = document.body;

const icon = document.getElementById('themeIcon');

body.classList.toggle('dark-mode');

const isDark = body.classList.contains('dark-mode');

icon.classList.toggle('fa-moon', !isDark);

icon.classList.toggle('fa-sun', isDark);

icon.classList.toggle('rotated', isDark);

localStorage.setItem('theme', isDark ? 'dark' : 'light');

}

window.onload = () => {

let theme = localStorage.getItem('theme');

if (!theme) {

theme = 'light';

localStorage.setItem('theme', 'light');

}

if (theme === 'dark') {

document.body.classList.add('dark-mode');

document.getElementById('themeIcon').classList.remove('fa-moon');

document.getElementById('themeIcon').classList.add('fa-sun', 'rotated');

}

};

</script>

</body>

</html>

TicTacToe.aspx.cs

using System;

using System.Data.SqlClient;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace myprj

{

public partial class TicTacToe : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

if (Session["Username"] == null)

{

Response.Redirect("Login.aspx");

return;

}

if (Session["ScoreX"] == null) Session["ScoreX"] = 0;

if (Session["ScoreO"] == null) Session["ScoreO"] = 0;

lblWelcome.Text = "👋 Welcome, " + Session["Username"].ToString();

InitializeGame();

}

}

private void InitializeGame()

{

Session["Player"] = "X";

Session["MoveCount"] = 0;

foreach (Control c in form1.Controls)

{

if (c is Button btn && btn.ID.StartsWith("btn") && btn.ID != "btnRestart")

{

btn.Text = "";

btn.Enabled = true;

// Reset CSS class to the base class

btn.CssClass = "btn-tile";

}

}

lblStatus.Text = "Player X's turn";

lblStatus.CssClass = "status-text player-x-turn";

lblMoves.Text = "Moves: 0";

UpdateScoreDisplay();

}

protected void Cell\_Click(object sender, EventArgs e)

{

Button btn = (Button)sender;

string currentPlayer = Session["Player"].ToString();

int moveCount = Convert.ToInt32(Session["MoveCount"]);

if (btn.Text == "")

{

btn.Text = currentPlayer;

btn.Enabled = false;

moveCount++;

Session["MoveCount"] = moveCount;

// Check if a player has won

if (CheckWin(currentPlayer, out Button[] winningButtons))

{

lblStatus.Text = $"🎉 Player {currentPlayer} wins!";

lblStatus.CssClass = "status-text win-message";

DisableAllButtons();

foreach (var button in winningButtons)

{

button.CssClass += " winner";

}

if (currentPlayer == "X")

Session["ScoreX"] = Convert.ToInt32(Session["ScoreX"]) + 1;

else

Session["ScoreO"] = Convert.ToInt32(Session["ScoreO"]) + 1;

UpdateScoreDisplay();

SaveGameResult(currentPlayer, moveCount);

}

// Check for a draw

else if (moveCount == 9)

{

lblStatus.Text = "🤝 It's a draw!";

lblStatus.CssClass = "status-text draw-message";

DisableAllButtons();

SaveGameResult("Draw", moveCount);

}

// Continue the game

else

{

string nextPlayer = (currentPlayer == "X") ? "O" : "X";

Session["Player"] = nextPlayer;

lblStatus.Text = $"Player {nextPlayer}'s turn";

lblStatus.CssClass = $"status-text player-{nextPlayer.ToLower()}-turn";

}

lblMoves.Text = $"Moves: {moveCount}";

}

}

private bool CheckWin(string player, out Button[] winningButtons)

{

winningButtons = null;

Button[,] buttons = new Button[3, 3];

for (int row = 0; row < 3; row++)

{

for (int col = 0; col < 3; col++)

{

buttons[row, col] = (Button)form1.FindControl($"btn{row}{col}");

}

}

int[][,] winningCombos = new int[][,]

{

new int[,] {{0,0}, {0,1}, {0,2}},

new int[,] {{1,0}, {1,1}, {1,2}},

new int[,] {{2,0}, {2,1}, {2,2}},

new int[,] {{0,0}, {1,0}, {2,0}},

new int[,] {{0,1}, {1,1}, {2,1}},

new int[,] {{0,2}, {1,2}, {2,2}},

new int[,] {{0,0}, {1,1}, {2,2}},

new int[,] {{0,2}, {1,1}, {2,0}}

};

foreach (var combo in winningCombos)

{

if (buttons[combo[0, 0], combo[0, 1]].Text == player &&

buttons[combo[1, 0], combo[1, 1]].Text == player &&

buttons[combo[2, 0], combo[2, 1]].Text == player)

{

winningButtons = new Button[]

{

buttons[combo[0, 0], combo[0, 1]],

buttons[combo[1, 0], combo[1, 1]],

buttons[combo[2, 0], combo[2, 1]]

};

return true;

}

}

return false;

}

private void DisableAllButtons()

{

foreach (Control c in form1.Controls)

{

if (c is Button btn && btn.ID.StartsWith("btn") && btn.ID != "btnRestart")

btn.Enabled = false;

}

}

private void UpdateScoreDisplay()

{

lblScoreX.Text = $"{Session["ScoreX"]}";

lblScoreO.Text = $"{Session["ScoreO"]}";

}

protected void btnRestart\_Click(object sender, EventArgs e)

{

InitializeGame();

}

private void SaveGameResult(string winner, int moves)

{

string username = Session["Username"]?.ToString() ?? "Guest";

string gameName = "Tic Tac Toe";

int score = (winner == "Draw") ? 0 : 1;

string extraInfo = $"Session Score - X: {Session["ScoreX"]}, O: {Session["ScoreO"]}";

string connStr = System.Configuration.ConfigurationManager

.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

try

{

using (SqlConnection con = new SqlConnection(connStr))

{

string query = @"

INSERT INTO GameHistory (Username, GameName, Winner, Moves, Score, ExtraInfo)

VALUES (@Username, @GameName, @Winner, @Moves, @Score, @ExtraInfo)";

using (SqlCommand cmd = new SqlCommand(query, con))

{

cmd.Parameters.AddWithValue("@Username", username);

cmd.Parameters.AddWithValue("@GameName", gameName);

cmd.Parameters.AddWithValue("@Winner", winner);

cmd.Parameters.AddWithValue("@Moves", moves);

cmd.Parameters.AddWithValue("@Score", score);

cmd.Parameters.AddWithValue("@ExtraInfo", extraInfo);

con.Open();

cmd.ExecuteNonQuery();

}

}

}

catch (Exception ex)

{

System.Diagnostics.Debug.WriteLine($"Error saving game result: {ex.Message}");

}

}

}

}

****

SnakeGame.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="SnakeGame.aspx.cs" Inherits="myprj.SnakeGame" %>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>GameHub | Snake Game</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<!-- Tailwind CSS -->

<script src="https://cdn.tailwindcss.com"></script>

<!-- Font Awesome -->

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css" rel="stylesheet">

<!-- Phosphor Icons -->

<script src="https://unpkg.com/phosphor-icons"></script>

<style>

body { font-family: 'Segoe UI', sans-serif; transition: background 0.3s, color 0.3s; }

.dark { background: #111827; color: #f9fafb; }

.game-wrapper { max-width: 600px; margin: auto; margin-top: 100px; padding: 2rem; border-radius: 1.5rem; background: white; box-shadow: 0 8px 20px rgba(0,0,0,0.15); }

.dark .game-wrapper { background: #1f2937; }

canvas { display: block; background: #fefefe; margin: auto; border: 3px solid #374151; border-radius: 0.75rem; }

.dark canvas { background: #111; border-color: #f3f4f6; }

</style>

</head>

<body class="light">

<form id="form1" runat="server">

<asp:ScriptManager ID="ScriptManager1" runat="server" EnablePageMethods="true" />

<!-- Welcome -->

<div class="fixed top-4 left-4 flex items-center space-x-2 z-20 bg-white/20 dark:bg-black/30 px-3 py-2 rounded-full shadow-md">

<i class="fa-solid fa-user text-lg"></i>

<asp:Label ID="lblWelcome" runat="server" CssClass="font-semibold"></asp:Label>

</div>

<!-- Theme + Dashboard -->

<div class="fixed top-4 right-4 flex items-center space-x-2 z-20">

<button id="themeToggleBtn" type="button" class="bg-gray-300 dark:bg-gray-700 text-gray-800 dark:text-gray-200 px-3 py-2 rounded-full shadow-lg transition-colors hover:bg-gray-400 dark:hover:bg-gray-600 focus:outline-none">

<i id="themeIcon" class="ph-moon-fill text-xl button-icon"></i>

</button>

<a href="Dashboard.aspx" class="bg-yellow-400 hover:bg-yellow-500 text-gray-900 font-semibold px-3 py-2 rounded-full shadow-lg transition-colors">

<i class="fa-solid fa-house"></i>

</a>

</div>

<!-- Game Wrapper -->

<div class="game-wrapper text-center">

<h1 class="mb-2 text-2xl font-bold"><i class="fa-solid fa-gamepad"></i> Snake Game</h1>

<p class="mb-4 text-gray-600 dark:text-gray-300">Use arrow keys to move the snake</p>

<canvas id="gameCanvas" width="400" height="400"></canvas>

<div class="mt-4 flex justify-between items-center">

<div class="text-lg font-semibold">Score: <span id="score">0</span></div>

<button id="startBtn" type="button" class="bg-green-500 hover:bg-green-600 text-white px-4 py-2 rounded-lg shadow-md transition">

<i class="fa-solid fa-play"></i> Start Game

</button>

</div>

</div>

</form>

<script>

// Theme toggle

const body = document.body;

const themeToggleBtn = document.getElementById("themeToggleBtn");

const themeIcon = document.getElementById("themeIcon");

themeToggleBtn.addEventListener("click", () => {

body.classList.toggle("dark");

themeIcon.classList.toggle("ph-moon-fill");

themeIcon.classList.toggle("ph-sun-fill");

});

// Snake Game

const canvas = document.getElementById("gameCanvas");

const ctx = canvas.getContext("2d");

const box = 20;

let snake, direction, food, score, gameInterval, snakeSpeed, gameOver;

function startGame() {

clearInterval(gameInterval); // clear old interval

snake = [{ x: 9 \* box, y: 10 \* box }];

direction = null;

food = generateFood();

score = 0;

snakeSpeed = 120;

gameOver = false;

document.getElementById("score").textContent = score;

gameInterval = setInterval(draw, snakeSpeed);

}

document.getElementById("startBtn").addEventListener("click", startGame);

document.addEventListener("keydown", (e) => {

if (e.key === "ArrowLeft" && direction !== "RIGHT") direction = "LEFT";

if (e.key === "ArrowUp" && direction !== "DOWN") direction = "UP";

if (e.key === "ArrowRight" && direction !== "LEFT") direction = "RIGHT";

if (e.key === "ArrowDown" && direction !== "UP") direction = "DOWN";

});

function generateFood() {

return {

x: Math.floor(Math.random() \* (canvas.width / box)) \* box,

y: Math.floor(Math.random() \* (canvas.height / box)) \* box

};

}

function endGame() {

clearInterval(gameInterval);

gameOver = true;

alert("Game Over! Your score: " + score);

saveScore(score);

}

function draw() {

if (gameOver) return;

ctx.fillStyle = body.classList.contains("dark") ? "#111827" : "#f9fafb";

ctx.fillRect(0, 0, canvas.width, canvas.height);

let headX = snake[0].x;

let headY = snake[0].y;

if (direction === "LEFT") headX -= box;

if (direction === "UP") headY -= box;

if (direction === "RIGHT") headX += box;

if (direction === "DOWN") headY += box;

// Border collision

if (headX < 0 || headY < 0 || headX >= canvas.width || headY >= canvas.height) {

endGame();

return;

}

// Food eaten

if (headX === food.x && headY === food.y) {

score++;

document.getElementById("score").textContent = score;

food = generateFood();

if (snakeSpeed > 50) {

snakeSpeed -= 5;

clearInterval(gameInterval);

gameInterval = setInterval(draw, snakeSpeed);

}

} else {

snake.pop();

}

let newHead = { x: headX, y: headY };

if (snake.some(seg => seg.x === newHead.x && seg.y === newHead.y)) {

endGame();

return;

}

snake.unshift(newHead);

// Draw snake

ctx.fillStyle = body.classList.contains("dark") ? "#34d399" : "#065f46";

snake.forEach(seg => ctx.fillRect(seg.x, seg.y, box, box));

// Draw food

ctx.fillStyle = body.classList.contains("dark") ? "#f87171" : "#dc2626";

ctx.fillRect(food.x, food.y, box, box);

}

function saveScore(finalScore) {

if (typeof PageMethods !== "undefined" && PageMethods.SaveScore) {

PageMethods.SaveScore(finalScore,

() => console.log("Score saved!"),

(err) => console.warn("Error saving score:", err)

);

}

}

</script>

</body>

</html>

SnakeGame.aspx.cs

using System;

using System.Configuration;

using System.Data.SqlClient;

using System.Web;

using System.Web.Services;

using System.Web.Script.Services;

namespace myprj

{

public partial class SnakeGame : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

// Require login

if (Session["Username"] == null)

{

Response.Redirect("Login.aspx");

}

else if (!IsPostBack)

{

string fullEmail = Session["Username"].ToString();

// Take only the part before '@'

string displayName = fullEmail.Split('@')[0];

// Capitalize first letter

if (!string.IsNullOrEmpty(displayName))

{

displayName = char.ToUpper(displayName[0]) + displayName.Substring(1);

}

lblWelcome.Text = "Welcome, " + displayName + "!";

}

}

/// <summary>

/// Saves the final score into GameHistory.

/// Uses EnableSession=true so we can read Session in a static WebMethod.

/// </summary>

[WebMethod(EnableSession = true)]

[ScriptMethod(ResponseFormat = ResponseFormat.Json)]

public static void SaveScore(int score)

{

if (HttpContext.Current == null || HttpContext.Current.Session == null || HttpContext.Current.Session["Username"] == null)

{

throw new Exception("User session not found. Please log in.");

}

string username = HttpContext.Current.Session["Username"].ToString();

string gameName = "Snake";

string winner = username; // Snake is single-player

int moves = score; // optional: treat score as "moves"

string extraInfo = "Snake game completed";

string cs = ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

const string sql = @"

INSERT INTO GameHistory (Username, GameName, Winner, Moves, Score, ExtraInfo)

VALUES (@Username, @GameName, @Winner, @Moves, @Score, @ExtraInfo);";

try

{

using (var con = new SqlConnection(cs))

using (var cmd = new SqlCommand(sql, con))

{

cmd.Parameters.AddWithValue("@Username", username);

cmd.Parameters.AddWithValue("@GameName", gameName);

cmd.Parameters.AddWithValue("@Winner", winner);

cmd.Parameters.AddWithValue("@Moves", moves);

cmd.Parameters.AddWithValue("@Score", score);

cmd.Parameters.AddWithValue("@ExtraInfo", extraInfo ?? (object)DBNull.Value);

con.Open();

cmd.ExecuteNonQuery();

}

}

catch (Exception ex)

{

System.Diagnostics.Debug.WriteLine("SaveScore error: " + ex);

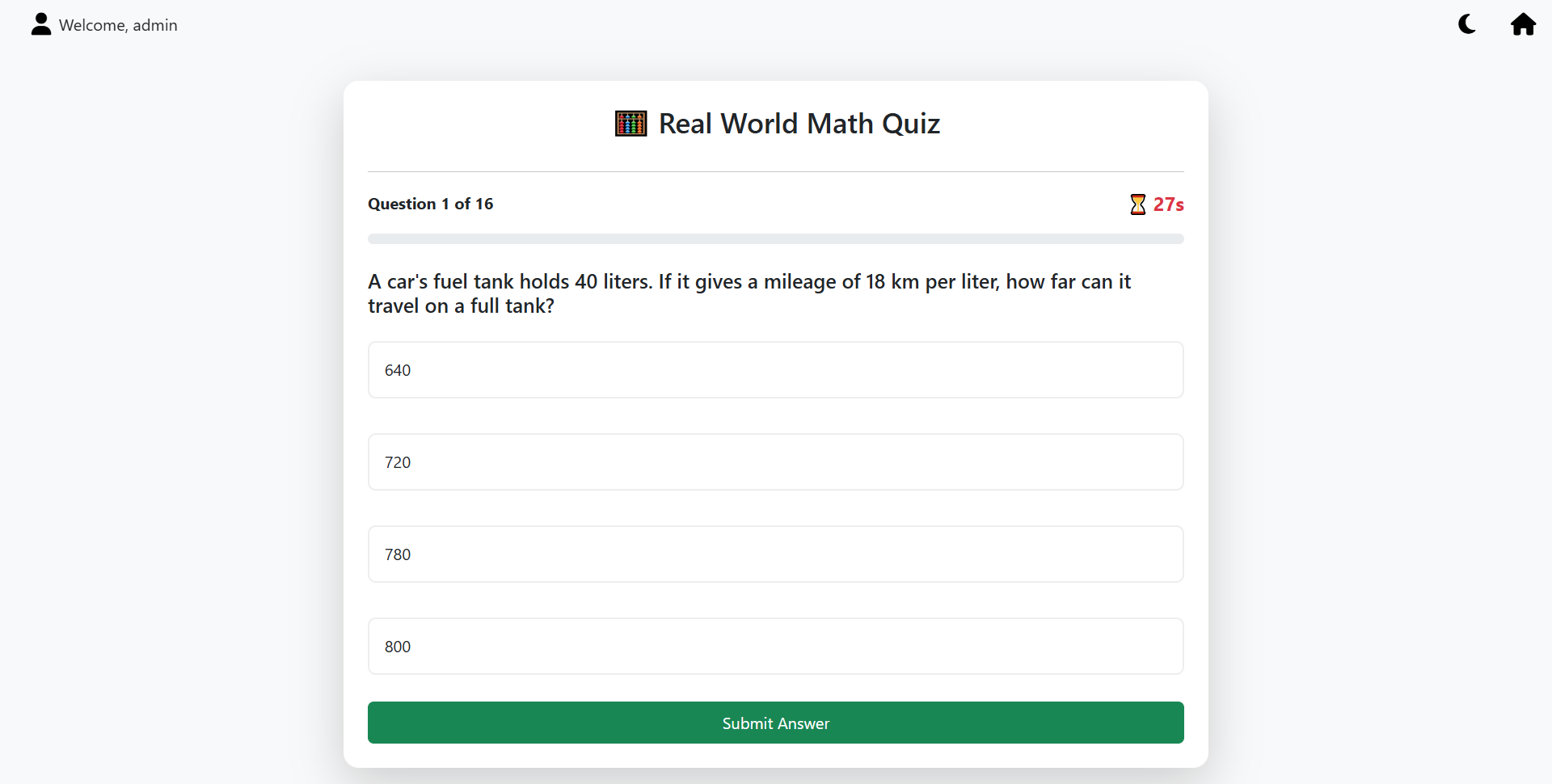
throw new Exception("Error saving score to database. " + ex.Message);

}

}

}

}

****

MathQuiz.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="MathQuiz.aspx.cs" Inherits="myprj.MathQuiz" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<title>GameHub | Real World Math Quiz</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css" rel="stylesheet" />

<style>

:root {

--light-bg: #f8f9fa; --light-text: #212529; --light-card: #ffffff;

--dark-bg: #121212; --dark-text: #e0e0e0; --dark-card: #1e1e1e;

--primary-color: #0d6efd; --success-color: #198754; --danger-color: #dc3545;

}

body { transition: background-color 0.3s, color 0.3s; }

body.light-mode { background-color: var(--light-bg); color: var(--light-text); }

body.dark-mode { background-color: var(--dark-bg); color: var(--dark-text); }

.top-bar i { font-size: 22px; cursor: pointer; margin-left: 15px; }

.quiz-wrapper { padding-top: 80px; }

.quiz-card {

border: none; border-radius: 15px;

transition: background-color 0.3s;

}

.light-mode .quiz-card { background-color: var(--light-card); }

.dark-mode .quiz-card { background-color: var(--dark-card); }

.dark-mode .list-group-item { background-color: #2a2a2a; border-color: #333; }

/\* Custom Radio Button Styling \*/

.rbl-options label {

display: block; width: 100%;

padding: 15px; margin-bottom: 10px;

border: 2px solid #ddd; border-radius: 8px;

cursor: pointer; transition: all 0.2s ease-in-out;

}

.light-mode .rbl-options label { border-color: #eee; }

.dark-mode .rbl-options label { border-color: #444; }

.rbl-options input[type="radio"] { display: none; }

.rbl-options input[type="radio"]:checked + label {

border-color: var(--primary-color);

background-color: var(--primary-color);

color: white; font-weight: bold;

}

.rbl-options label:hover {

border-color: var(--primary-color);

transform: translateY(-2px);

}

.timer { font-size: 1.2rem; }

.icon-btn {

color: #000; /\* Default black \*/

font-size: 1.3rem;

cursor: pointer;

transition: color 0.3s ease;

text-decoration: none; /\* Remove link underline \*/

}

/\* Hover effect \*/

.icon-btn:hover {

color: #555;

}

/\* Dark mode support \*/

body.dark-mode .icon-btn {

color: #fff;

}

</style>

</head>

<body class="light-mode">

<form id="form1" runat="server">

<div class="top-bar d-flex justify-content-between align-items-center px-3 py-2"

style="position: fixed; top: 0; left: 0; right: 0; z-index: 1000; background: transparent;">

<div class="d-flex align-items-center">

<i class="fa-solid fa-user icon-btn me-2"></i>

<asp:Label ID="lblWelcome" runat="server" Text="Welcome, "></asp:Label>

<asp:Label ID="lblUsername" runat="server"></asp:Label>

</div>

<div>

<i id="themeToggle" class="fa-solid fa-moon icon-btn me-3"></i>

<a href="Dashboard.aspx" class="icon-btn">

<i class="fa-solid fa-house"></i>

</a>

</div>

</div>

<div class="container quiz-wrapper">

<div class="row justify-content-center">

<div class="col-lg-8 col-md-10">

<asp:Panel ID="pnlQuiz" runat="server">

<div class="card shadow-lg p-4 quiz-card">

<h3 class="text-center mb-3">🧮 Real World Math Quiz</h3>

<hr />

<div class="d-flex justify-content-between align-items-center mb-3">

<asp:Label ID="lblQuestionProgress" runat="server" CssClass="fw-bold"></asp:Label>

<asp:Label ID="lblTimer" runat="server" CssClass="fw-bold text-danger timer"></asp:Label>

</div>

<div class="progress mb-4" style="height: 10px;">

<asp:Panel ID="pnlProgressBar" runat="server" CssClass="progress-bar" role="progressbar"></asp:Panel>

</div>

<asp:Label ID="lblQuestion" runat="server" CssClass="h5 mb-4"></asp:Label>

<asp:RadioButtonList ID="rblOptions" runat="server" CssClass="rbl-options" RepeatLayout="Flow" />

<asp:Button ID="btnSubmit" runat="server" Text="Submit Answer" OnClick="btnSubmit\_Click" CssClass="btn btn-success w-100 mt-3 p-2" />

</div>

</asp:Panel>

<asp:Panel ID="pnlResult" runat="server" Visible="false">

<div class="card shadow-lg p-5 text-center quiz-card">

<h2 class="mb-3">🎉 Quiz Complete!</h2>

<p class="fs-5">You've successfully completed the math challenge.</p>

<asp:Label ID="lblResult" runat="server" CssClass="display-5 fw-bold my-4"></asp:Label>

<a href="Dashboard.aspx" class="btn btn-primary w-50 mx-auto">Back to Dashboard</a>

</div>

</asp:Panel>

</div>

</div>

</div>

</form>

<script>

// Theme Toggle

const themeToggle = document.getElementById("themeToggle");

themeToggle.addEventListener("click", function () {

document.body.classList.toggle("dark-mode");

document.body.classList.toggle("light-mode");

this.classList.toggle("fa-sun");

this.classList.toggle("fa-moon");

});

</script>

</body>

</html>

MathQuiz.aspx.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace myprj

{

public partial class MathQuiz : Page

{

private const int TimePerQuestion = 30; // seconds

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Username"] == null)

{

Response.Redirect("Login.aspx");

return;

}

lblWelcome.Text = "Welcome, " + Session["Username"].ToString();

if (!IsPostBack)

{

StartNewQuiz();

}

}

private void StartNewQuiz()

{

// Use Session to store quiz state for the current user

Session["QuizScore"] = 0;

Session["QuestionIndex"] = 0;

Session["QuizQuestions"] = RealWorldQuestion.GenerateQuestions();

DisplayQuestion();

}

private void DisplayQuestion()

{

var questions = Session["QuizQuestions"] as List<RealWorldQuestion>;

int questionIndex = (int)Session["QuestionIndex"];

if (questions != null && questionIndex < questions.Count)

{

RealWorldQuestion currentQuestion = questions[questionIndex];

lblQuestion.Text = currentQuestion.QuestionText;

// Bind options to the RadioButtonList

rblOptions.Items.Clear();

foreach (var option in currentQuestion.Options)

{

rblOptions.Items.Add(new ListItem(option.ToString(), option.ToString()));

}

// Update Progress Bar and Counter

lblQuestionProgress.Text = $"Question {questionIndex + 1} of {questions.Count}";

double progressPercentage = (double)(questionIndex) / questions.Count \* 100;

pnlProgressBar.Style["width"] = $"{progressPercentage}%";

StartTimer();

}

else

{

EndQuiz();

}

}

protected void btnSubmit\_Click(object sender, EventArgs e)

{

var questions = Session["QuizQuestions"] as List<RealWorldQuestion>;

int questionIndex = (int)Session["QuestionIndex"];

int score = (int)Session["QuizScore"];

if (questions != null && questionIndex < questions.Count)

{

// Check if an answer was selected

if (rblOptions.SelectedIndex != -1)

{

int selectedAnswer = int.Parse(rblOptions.SelectedValue);

if (selectedAnswer == questions[questionIndex].CorrectAnswer)

{

score++;

Session["QuizScore"] = score; // Update score in session

}

}

// Move to the next question

questionIndex++;

Session["QuestionIndex"] = questionIndex;

DisplayQuestion();

}

}

private void StartTimer()

{

lblTimer.Text = $"⏳ {TimePerQuestion}s";

// Injects JavaScript to run a countdown timer on the client-side

string script = $@"

var timeLeft = {TimePerQuestion};

var lbl = document.getElementById('{lblTimer.ClientID}');

clearInterval(window.quizTimer);

window.quizTimer = setInterval(function() {{

timeLeft--;

lbl.innerText = '⏳ ' + timeLeft + 's';

if (timeLeft <= 0) {{

clearInterval(window.quizTimer);

\_\_doPostBack('{btnSubmit.UniqueID}', ''); // Auto-submit when time is up

}}

}}, 1000);";

ScriptManager.RegisterStartupScript(this, GetType(), "StartTimer", script, true);

}

private void EndQuiz()

{

// Hide quiz panel and show result panel

pnlQuiz.Visible = false;

pnlResult.Visible = true;

var questions = Session["QuizQuestions"] as List<RealWorldQuestion>;

int finalScore = (int)Session["QuizScore"];

lblResult.Text = $"Your Score: {finalScore} / {questions.Count}";

// Stop the JavaScript timer

ScriptManager.RegisterStartupScript(this, GetType(), "StopTimer", "clearInterval(window.quizTimer);", true);

SaveScoreToDatabase(finalScore, questions.Count);

}

private void SaveScoreToDatabase(int score, int totalQuestions)

{

string cs = System.Configuration.ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

string query = @"INSERT INTO GameHistory

(Username, GameName, Winner, Moves, Score, ExtraInfo)

VALUES

(@Username, @GameName, @Winner, @Moves, @Score, @ExtraInfo)";

using (SqlCommand cmd = new SqlCommand(query, con))

{

cmd.Parameters.AddWithValue("@Username", Session["Username"].ToString());

cmd.Parameters.AddWithValue("@GameName", "Real World Math Quiz");

cmd.Parameters.AddWithValue("@Winner", Session["Username"].ToString()); // Assuming single player

cmd.Parameters.AddWithValue("@Moves", totalQuestions);

cmd.Parameters.AddWithValue("@Score", score);

cmd.Parameters.AddWithValue("@ExtraInfo", "Quiz Completed");

con.Open();

cmd.ExecuteNonQuery();

}

}

}

}

public class RealWorldQuestion

{

public string QuestionText { get; set; }

public List<int> Options { get; set; }

public int CorrectAnswer { get; set; }

public static List<RealWorldQuestion> GenerateQuestions()

{

var rand = new Random();

var allQuestions = new List<RealWorldQuestion>

{

// Original 5 Questions

new RealWorldQuestion { QuestionText = "You buy a shirt for ₹800 after a 20% discount. What was the original price?",

Options = new List<int> { 960, 1000, 1040, 1200 }, CorrectAnswer = 1000 },

new RealWorldQuestion { QuestionText = "A car travels 150 km in 2 hours and 30 minutes. What is its average speed in km/h?",

Options = new List<int> { 50, 60, 75, 80 }, CorrectAnswer = 60 },

new RealWorldQuestion { QuestionText = "If a pizza is cut into 8 slices and you eat 3, what whole number percentage of the pizza is left?",

Options = new List<int> { 37, 50, 62, 75 }, CorrectAnswer = 62 },

new RealWorldQuestion { QuestionText = "You need to fence a square garden with a side length of 15 meters. How many meters of fence are required?",

Options = new List<int> { 30, 45, 60, 225 }, CorrectAnswer = 60 },

new RealWorldQuestion { QuestionText = "A recipe calls for 250g of flour to make 10 cookies. How much flour is needed for 25 cookies?",

Options = new List<int> { 500, 575, 600, 625 }, CorrectAnswer = 625 },

// New 10 Questions

new RealWorldQuestion { QuestionText = "Your restaurant bill is ₹1500. If you want to leave a 10% tip, how much should you add?",

Options = new List<int> { 100, 120, 150, 200 }, CorrectAnswer = 150 },

new RealWorldQuestion { QuestionText = "You have ₹5000 and spend ₹1250 on groceries and ₹800 on bills. How much money do you have left?",

Options = new List<int> { 2950, 3050, 3200, 3750 }, CorrectAnswer = 2950 },

new RealWorldQuestion { QuestionText = "A car's fuel tank holds 40 liters. If it gives a mileage of 18 km per liter, how far can it travel on a full tank?",

Options = new List<int> { 640, 720, 780, 800 }, CorrectAnswer = 720 },

new RealWorldQuestion { QuestionText = "If you deposit ₹10,000 in a bank account with a simple annual interest rate of 5%, how much interest will you earn in one year?",

Options = new List<int> { 250, 500, 750, 1000 }, CorrectAnswer = 500 },

new RealWorldQuestion { QuestionText = "A student scores 85, 90, and 95 in three tests. What is their average score?",

Options = new List<int> { 88, 90, 92, 93 }, CorrectAnswer = 90 },

new RealWorldQuestion { QuestionText = "You need to buy carpet for a room that is 5 meters long and 4 meters wide. How many square meters of carpet do you need?",

Options = new List<int> { 18, 20, 22, 25 }, CorrectAnswer = 20 },

new RealWorldQuestion { QuestionText = "A movie starts at 7:15 PM and is 140 minutes long. What time does it end?",

Options = new List<int> { 2135, 2145, 2155, 2205 }, CorrectAnswer = 2135 }, // Note: Representing 9:35 PM as 2135 is tricky for int options. A better UI would handle time formats. This is a simplification. Let's change the question.

// Better version of the time question:

new RealWorldQuestion { QuestionText = "A train journey takes 4 hours and 30 minutes. If you start at 9:00 AM, at what time will you arrive?",

Options = new List<int> { 1300, 1330, 1400, 1430 }, CorrectAnswer = 1330 }, // 1:30 PM

new RealWorldQuestion { QuestionText = "You buy a 2-liter bottle of juice. If you pour it into 250 ml glasses, how many glasses can you fill?",

Options = new List<int> { 6, 8, 10, 12 }, CorrectAnswer = 8 },

new RealWorldQuestion { QuestionText = "If one person can paint a wall in 6 hours, how long will it take two people working at the same rate to paint it?",

Options = new List<int> { 2, 3, 4, 12 }, CorrectAnswer = 3 },

new RealWorldQuestion { QuestionText = "A water tank is 3 meters long, 2 meters wide, and 2 meters deep. What is its volume in cubic meters?",

Options = new List<int> { 7, 10, 12, 14 }, CorrectAnswer = 12 }

};

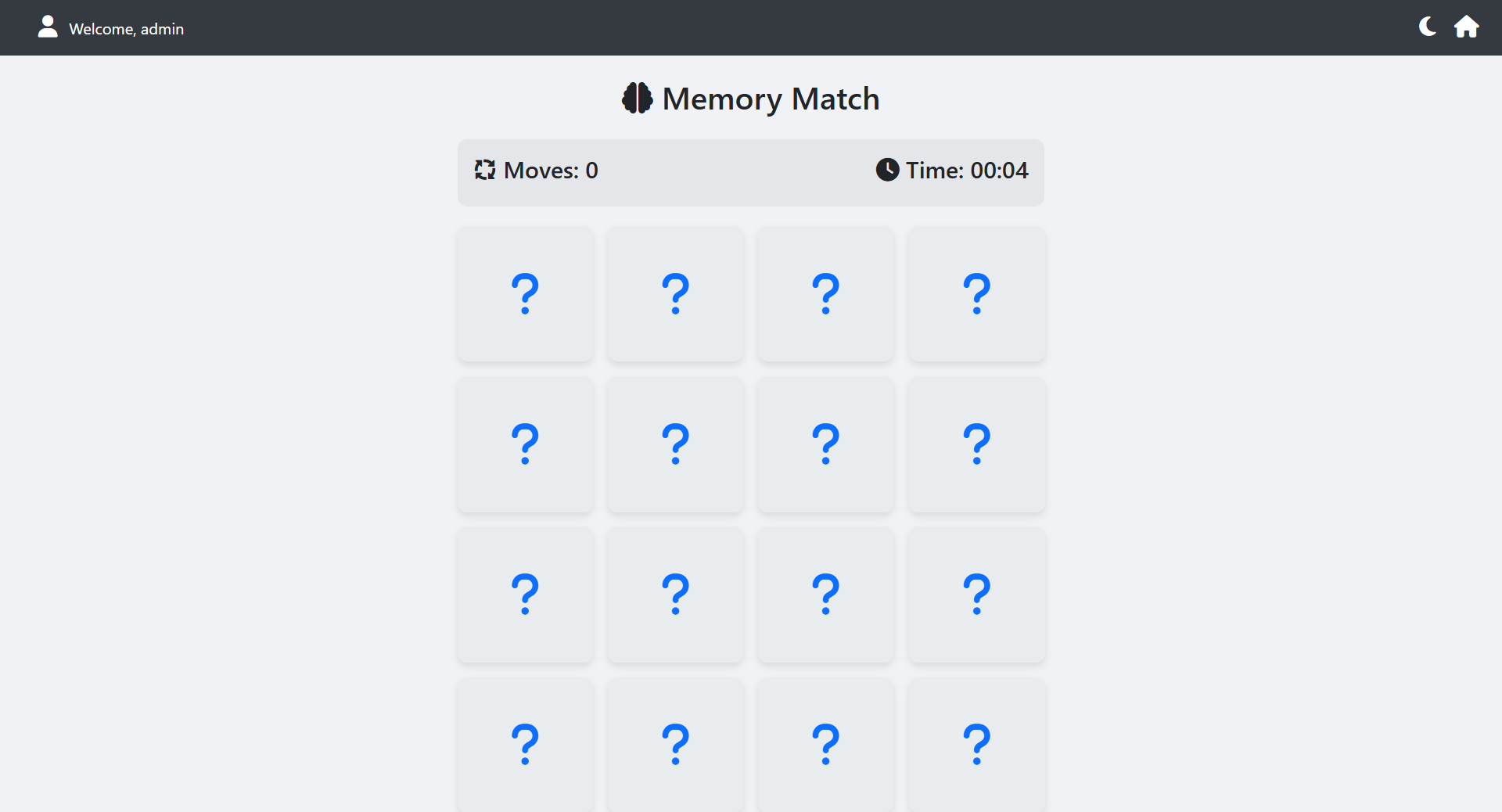
// Shuffling ensures users get a different question order each time

return allQuestions.OrderBy(q => rand.Next()).ToList();

}

}

}

****

MemoryMatch.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="MemoryMatch.aspx.cs" Inherits="myprj.MemoryMatch" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<title>GameHub | Memory Match</title>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css" rel="stylesheet" />

<style>

:root {

--light-bg: #f0f2f5; --light-text: #212529; --light-card: #ffffff; --light-card-front: #e9ecef;

--dark-bg: #121212; --dark-text: #e0e0e0; --dark-card: #1e1e1e; --dark-card-front: #343a40;

--primary-color: #0d6efd;

}

body { font-family: 'Segoe UI', sans-serif; transition: background-color 0.3s, color 0.3s; padding-top: 80px; }

body.light-mode { background-color: var(--light-bg); color: var(--light-text); }

body.dark-mode { background-color: var(--dark-bg); color: var(--dark-text); }

.top-bar {

position: fixed;

top: 0;

left: 0;

width: 100%;

display: flex;

justify-content: space-between;

align-items: center;

padding: 1rem 1.5rem;

background-color: #343a40; /\* A neutral dark color for the bar \*/

z-index: 1000;

}

.top-bar i { color: white; font-size: 22px; cursor: pointer; margin-left: 15px; }

.game-board { max-width: 600px; margin: auto; }

.game-header {

display: flex; justify-content: space-between; align-items: center;

padding: 1rem; border-radius: 10px; margin-bottom: 20px;

}

.light-mode .game-header { background-color: rgba(0,0,0,0.05); }

.dark-mode .game-header { background-color: rgba(255,255,255,0.05); }

.card-grid {

display: grid;

grid-template-columns: repeat(4, 1fr);

gap: 15px;

}

.memory-card {

aspect-ratio: 1 / 1;

perspective: 1000px;

cursor: pointer;

}

.memory-card-inner {

position: relative; width: 100%; height: 100%;

transition: transform 0.6s;

transform-style: preserve-3d;

border-radius: 10px;

}

.memory-card.flip .memory-card-inner { transform: rotateY(180deg); }

.card-face {

position: absolute; width: 100%; height: 100%;

backface-visibility: hidden;

border-radius: 10px;

display: flex; align-items: center; justify-content: center;

font-size: clamp(2rem, 8vw, 3rem);

box-shadow: 0 4px 8px rgba(0,0,0,0.1);

}

.card-front { background-color: var(--light-card-front); color: var(--primary-color); }

.dark-mode .card-front { background-color: var(--dark-card-front); }

.card-back { background-color: #ffc107; color: #333; transform: rotateY(180deg); }

.modal-content { border-radius: 15px; }

.light-mode .modal-content { background-color: var(--light-card); }

.dark-mode .modal-content { background-color: var(--dark-card); }

</style>

</head>

<body class="light-mode">

<form id="form1" runat="server">

<asp:ScriptManager ID="ScriptManager1" runat="server" EnablePageMethods="true" />

<div class="top-bar">

<div>

<i class="fa-solid fa-user"></i>

<asp:Label ID="lblWelcome" runat="server" CssClass="text-white ms-2 fs-6"></asp:Label>

</div>

<div>

<i id="themeToggle" class="fa-solid fa-moon"></i>

<a href="Dashboard.aspx"><i class="fa-solid fa-house"></i></a>

</div>

</div>

<div class="container">

<div class="game-board">

<h2 class="text-center mb-4"><i class="fas fa-brain"></i> Memory Match</h2>

<div class="game-header">

<h4><i class="fa-solid fa-arrows-spin"></i> Moves: <span id="moveCount">0</span></h4>

<h4><i class="fa-solid fa-clock"></i> Time: <span id="timer">00:00</span></h4>

</div>

<div class="card-grid" id="cardGrid">

</div>

</div>

</div>

<div class="modal fade" id="gameOverModal" tabindex="-1" data-bs-backdrop="static" data-bs-keyboard="false">

<div class="modal-dialog modal-dialog-centered">

<div class="modal-content">

<div class="modal-body text-center p-5">

<h1 class="display-4">🎉 Well Done!</h1>

<p class="lead">You matched all the pairs.</p>

<div class="d-flex justify-content-around my-4">

<div>

<h5>Moves</h5>

<p id="finalMoves" class="fs-3 fw-bold">0</p>

</div>

<div>

<h5>Time</h5>

<p id="finalTime" class="fs-3 fw-bold">00:00</p>

</div>

</div>

<p class="text-success"><i class="fa-solid fa-check-circle"></i> Your score has been saved automatically!</p>

<div class="d-grid gap-2 d-sm-flex justify-content-sm-center mt-4">

<button type="button" class="btn btn-warning btn-lg" onclick="restartGame()">Play Again</button>

<a href="Dashboard.aspx" class="btn btn-secondary btn-lg">Dashboard</a>

</div>

</div>

</div>

</div>

</div>

</form>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

<script>

const emojis = ['🐶', '🐱', '🐭', '🐹', '🐰', '🦊', '🐻', '🐼'];

let flippedCards = [], matchedPairs = 0, moves = 0;

let timerInterval, totalSeconds = 0;

let gameOverModal;

function shuffle(array) {

return [...array, ...array].sort(() => Math.random() - 0.5);

}

function initGame() {

gameOverModal = new bootstrap.Modal(document.getElementById('gameOverModal'));

const cardGrid = document.getElementById('cardGrid');

cardGrid.innerHTML = '';

shuffle(emojis).forEach(emoji => {

const card = document.createElement('div');

card.className = 'memory-card';

card.dataset.emoji = emoji;

card.innerHTML = `

<div class="memory-card-inner">

<div class="card-face card-front"><i class="fa-solid fa-question"></i></div>

<div class="card-face card-back">${emoji}</div>

</div>`;

card.addEventListener('click', handleCardClick);

cardGrid.appendChild(card);

});

resetGameStats();

startTimer();

}

function resetGameStats() {

moves = 0;

matchedPairs = 0;

totalSeconds = 0;

flippedCards = [];

document.getElementById('moveCount').textContent = '0';

document.getElementById('timer').textContent = '00:00';

stopTimer();

}

function startTimer() {

timerInterval = setInterval(() => {

totalSeconds++;

const min = String(Math.floor(totalSeconds / 60)).padStart(2, '0');

const sec = String(totalSeconds % 60).padStart(2, '0');

document.getElementById('timer').textContent = `${min}:${sec}`;

}, 1000);

}

function stopTimer() {

clearInterval(timerInterval);

}

function handleCardClick(e) {

const clickedCard = e.currentTarget;

if (clickedCard.classList.contains('flip') || flippedCards.length === 2) {

return;

}

clickedCard.classList.add('flip');

flippedCards.push(clickedCard);

if (flippedCards.length === 2) {

processMove();

}

}

function processMove() {

moves++;

document.getElementById('moveCount').textContent = moves;

const [card1, card2] = flippedCards;

if (card1.dataset.emoji === card2.dataset.emoji) {

matchedPairs++;

flippedCards = [];

if (matchedPairs === emojis.length) {

endGame();

}

} else {

setTimeout(() => {

card1.classList.remove('flip');

card2.classList.remove('flip');

flippedCards = [];

}, 1000);

}

}

function endGame() {

stopTimer();

const finalTime = document.getElementById('timer').textContent;

// Call server-side C# method to save the score

PageMethods.SaveGameResult(moves, finalTime, onSaveSuccess, onSaveFailure);

// Update and show the modal

document.getElementById('finalMoves').textContent = moves;

document.getElementById('finalTime').textContent = finalTime;

}

// Callback functions for the server call

function onSaveSuccess(response) {

console.log("Score saved successfully!");

gameOverModal.show();

}

function onSaveFailure(error) {

console.error("Failed to save score: ", error);

alert("Could not save your score. Please check your connection.");

gameOverModal.show(); // Still show modal, but maybe with an error message

}

function restartGame() {

gameOverModal.hide();

initGame();

}

// Theme Toggle

const themeToggle = document.getElementById("themeToggle");

themeToggle.addEventListener("click", () => {

document.body.classList.toggle("dark-mode");

document.body.classList.toggle("light-mode");

themeToggle.classList.toggle("fa-sun");

themeToggle.classList.toggle("fa-moon");

});

// Initialize game on page load

window.onload = initGame;

</script>

</body>

</html>

MemoryMatch.aspx.cs

using System;

using System.Configuration;

using System.Data.SqlClient;

using System.Web;

using System.Web.Services;

namespace myprj

{

public partial class MemoryMatch : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Username"] == null)

{

Response.Redirect("Login.aspx");

return;

}

lblWelcome.Text = "Welcome, " + Session["Username"].ToString();

}

[WebMethod]

public static void SaveGameResult(int moves, string time)

{

// WebMethods are static, so we access Session through HttpContext

if (HttpContext.Current.Session["Username"] == null)

{

// Can't save if the user is not logged in.

// You could return an error message here if needed.

return;

}

string username = HttpContext.Current.Session["Username"].ToString();

// A lower move count is a better score

int score = moves;

string extraInfo = $"Time: {time}";

string connStr = ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString; // Use your connection string name

using (SqlConnection con = new SqlConnection(connStr))

{

// Note: The schema might be different. Adjust the query to your table columns.

string query = @"INSERT INTO GameHistory

(Username, GameName, Winner, Moves, Score, ExtraInfo)

VALUES

(@Username, @GameName, @Winner, @Moves, @Score, @ExtraInfo)";

using (SqlCommand cmd = new SqlCommand(query, con))

{

cmd.Parameters.AddWithValue("@Username", username);

cmd.Parameters.AddWithValue("@GameName", "Memory Match");

cmd.Parameters.AddWithValue("@Winner", username);

cmd.Parameters.AddWithValue("@Moves", moves);

cmd.Parameters.AddWithValue("@Score", score); // Storing moves as the primary score

cmd.Parameters.AddWithValue("@ExtraInfo", extraInfo);

con.Open();

cmd.ExecuteNonQuery();

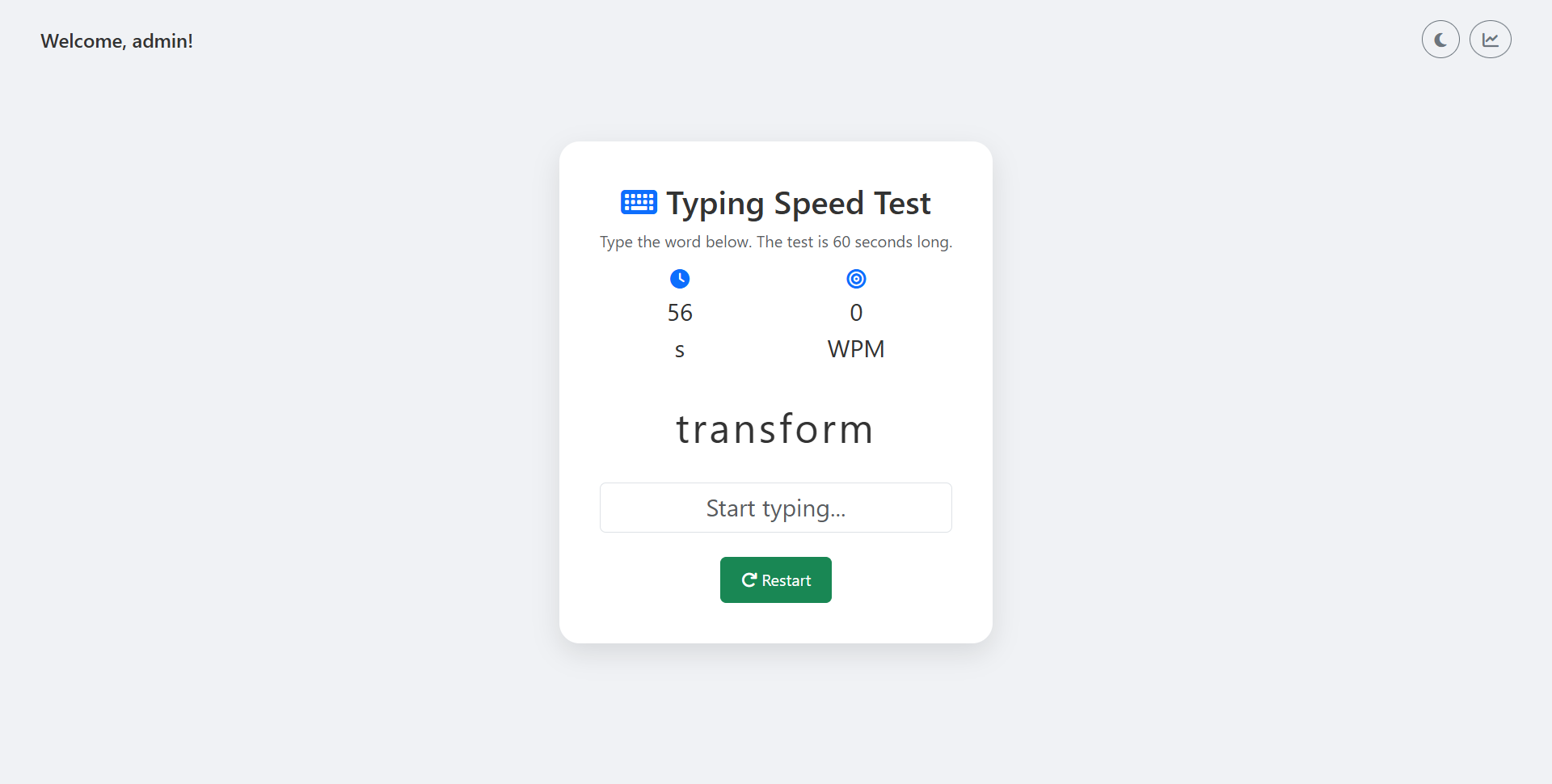
}

}

}

}

}

****

TypingSpeed.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="TypingSpeed.aspx.cs" Inherits="myprj.TypingSpeed" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Typing Speed Test</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css" rel="stylesheet" />

<style>

:root {

--bg-color: #f0f2f5;

--text-color: #333;

--card-bg: #ffffff;

--card-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);

--primary-color: #0d6efd;

--correct-color: #28a745;

--incorrect-color: #dc3545;

}

body {

background-color: var(--bg-color);

color: var(--text-color);

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

transition: background-color 0.3s, color 0.3s;

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

margin: 0;

flex-direction: column;

}

.dark-mode {

--bg-color: #121212;

--text-color: #e0e0e0;

--card-bg: #1e1e1e;

--card-shadow: 0 10px 25px rgba(0, 0, 0, 0.3);

}

header {

width: 100%;

padding: 20px 40px;

display: flex;

justify-content: space-between;

align-items: center;

position: absolute;

top: 0;

left: 0;

}

.welcome-user {

font-size: 1.2rem;

font-weight: 500;

}

.welcome-user span {

font-weight: bold;

}

.header-icons {

display: flex;

align-items: center;

gap: 10px;

}

.game-container {

width: 100%;

max-width: 700px;

background-color: var(--card-bg);

border-radius: 20px;

padding: 40px;

box-shadow: var(--card-shadow);

text-align: center;

}

.stats {

display: flex;

justify-content: space-around;

font-size: 1.5rem;

margin-bottom: 30px;

}

.stat-item {

display: flex;

flex-direction: column;

align-items: center;

}

.stat-item i {

font-size: 1.2rem;

margin-bottom: 5px;

color: var(--primary-color);

}

.word-display {

font-size: 2.5rem;

letter-spacing: 3px;

margin-bottom: 25px;

user-select: none;

}

.word-display span.correct { color: var(--correct-color); }

.word-display span.incorrect { color: var(--incorrect-color); background-color: #ff000020; }

#input-field {

font-size: 1.5rem;

text-align: center;

}

#btn-restart, #btn-dashboard {

font-size: 1rem;

padding: 10px 20px;

}

</style>

</head>

<body>

<form id="form1" runat="server">

<asp:ScriptManager ID="ScriptManager1" runat="server" EnablePageMethods="true"></asp:ScriptManager>

<header>

<div class="welcome-user">

Welcome, <asp:Literal ID="lblUsername" runat="server" />!

</div>

<div class="header-icons">

<button type="button" id="theme-toggle" class="btn btn-outline-secondary rounded-circle">

<i class="fas fa-moon"></i>

</button>

<a href="Dashboard.aspx" class="btn btn-outline-secondary rounded-circle">

<i class="fas fa-chart-line"></i>

</a>

</div>

</header>

<div class="game-container">

<h2><i class="fas fa-keyboard text-primary"></i> Typing Speed Test</h2>

<p class="text-muted">Type the word below. The test is 60 seconds long.</p>

<div class="stats">

<div class="stat-item">

<i class="fas fa-clock"></i>

<span id="time">60</span>s

</div>

<div class="stat-item">

<i class="fas fa-bullseye"></i>

<span id="score">0</span> WPM

</div>

</div>

<div class="word-display" id="word-display">Click Start to Begin</div>

<input type="text" class="form-control" id="input-field" placeholder="Start typing..." disabled />

<div class="mt-4">

<button type="button" id="btn-restart" class="btn btn-success">

<i class="fas fa-play"></i> Start

</button>

</div>

<div id="saving-indicator" class="mt-3 text-muted" style="display:none;">

Game Over! Saving your score...

</div>

</div>

</form>

<script>

const words = ["ability", "achieve", "acquire", "analyze", "balance", "because", "believe", "benefit", "captain", "certain", "change", "clarity", "company", "compare", "compete", "concept", "confirm", "contact", "control", "develop", "digital", "discuss", "economy", "educate", "element", "empower", "engine", "enlarge", "enormous", "enquire", "example", "execute", "explain", "express", "finance", "foreign", "freedom", "further", "general", "generate", "genuine", "govern", "history", "however", "imagine", "improve", "include", "inspire", "involve", "journey", "justice", "justify", "kingdom", "knowledge", "language", "leadership", "liberty", "library", "machine", "magnify", "maintain", "major", "manage", "measure", "mention", "message", "million", "monitor", "morning", "mystery", "narrate", "natural", "neglect", "network", "nothing", "nuclear", "observe", "operate", "opinion", "organize", "outcome", "outline", "perfect", "perform", "picture", "popular", "prepare", "present", "prevent", "primary", "private", "problem", "proceed", "process", "produce", "program", "project", "promote", "protect", "provide", "publish", "quality", "question", "quickly", "realize", "receive", "reflect", "release", "require", "research", "resolve", "respect", "respond", "restore", "results", "revenue", "science", "section", "service", "several", "similar", "society", "special", "station", "success", "suggest", "support", "sustain", "teacher", "thought", "through", "tonight", "tradition", "transform", "trouble", "ultimate", "uncover", "uniform", "unique", "unusual", "valuable", "various", "vehicle", "venture", "version", "victory", "visual", "welcome", "whatever", "whereas", "willing", "without", "wonderful", "yourself"];

// DOM Elements

const wordDisplayEl = document.getElementById('word-display');

const inputFieldEl = document.getElementById('input-field');

const timeEl = document.getElementById('time');

const scoreEl = document.getElementById('score');

const restartBtn = document.getElementById('btn-restart');

const savingIndicator = document.getElementById('saving-indicator');

// Game State

const GAME\_TIME = 60;

let score = 0;

let timeLeft = GAME\_TIME;

let timer;

let isPlaying = false;

let currentWord = '';

// --- Game Logic ---

function startGame() {

// Reset state

score = 0;

timeLeft = GAME\_TIME;

isPlaying = true;

inputFieldEl.disabled = false;

savingIndicator.style.display = 'none';

// Update UI

scoreEl.textContent = score;

timeEl.textContent = timeLeft;

restartBtn.innerHTML = '<i class="fas fa-redo"></i> Restart';

// Start countdown

clearInterval(timer);

timer = setInterval(updateTimer, 1000);

// Show first word

showNewWord();

inputFieldEl.focus();

}

function showNewWord() {

const randIndex = Math.floor(Math.random() \* words.length);

currentWord = words[randIndex];

wordDisplayEl.innerHTML = ''; // Clear previous word

currentWord.split('').forEach(char => {

const charSpan = document.createElement('span');

charSpan.textContent = char;

wordDisplayEl.appendChild(charSpan);

});

inputFieldEl.value = '';

}

function updateTimer() {

if (timeLeft > 0) {

timeLeft--;

timeEl.textContent = timeLeft;

} else {

gameOver();

}

}

function gameOver() {

isPlaying = false;

clearInterval(timer);

inputFieldEl.disabled = true;

wordDisplayEl.innerHTML = 'Time\'s Up!';

restartBtn.innerHTML = '<i class="fas fa-play"></i> Start';

savingIndicator.style.display = 'block';

// Auto-save the score

saveScoreToServer(score);

}

function handleInput() {

if (!isPlaying) return;

const typedValue = inputFieldEl.value;

const letterSpans = wordDisplayEl.querySelectorAll('span');

// Provide real-time feedback

let allCorrect = true;

letterSpans.forEach((charSpan, index) => {

const typedChar = typedValue[index];

if (typedChar == null) {

charSpan.classList.remove('correct', 'incorrect');

allCorrect = false;

} else if (typedChar === charSpan.innerText) {

charSpan.classList.add('correct');

charSpan.classList.remove('incorrect');

} else {

charSpan.classList.add('incorrect');

charSpan.classList.remove('correct');

allCorrect = false;

}

});

// Check for word completion

if (allCorrect && typedValue.length === currentWord.length) {

score++;

scoreEl.textContent = score;

showNewWord();

}

}

// --- Server Communication ---

async function saveScoreToServer(finalScore) {

try {

const response = await fetch('TypingSpeed.aspx/SaveScore', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify({ score: finalScore })

});

if (response.ok) {

// Redirect after successful save

window.location.href = 'Leaderboard.aspx';

} else {

console.error('Failed to save score.');

savingIndicator.textContent = 'Error saving score. Please try again.';

}

} catch (error) {

console.error('An error occurred:', error);

savingIndicator.textContent = 'Connection error. Could not save score.';

}

}

// --- Event Listeners ---

restartBtn.addEventListener('click', startGame);

inputFieldEl.addEventListener('input', handleInput);

// --- Theme Toggle ---

const themeToggleBtn = document.getElementById('theme-toggle');

themeToggleBtn.addEventListener('click', () => {

document.body.classList.toggle('dark-mode');

const isDarkMode = document.body.classList.contains('dark-mode');

themeToggleBtn.innerHTML = isDarkMode ? '<i class="fas fa-sun"></i>' : '<i class="fas fa-moon"></i>';

});

</script>

</body>

</html>

TypingSpeed.aspx.cs

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web;

using System.Web.Services;

namespace myprj

{

public partial class TypingSpeed : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

// Check if a user is logged in

if (Session["username"] == null)

{

// If not logged in, redirect to the login page

Response.Redirect("Login.aspx");

}

else

{

// If logged in, display the username in the header

lblUsername.Text = HttpContext.Current.Session["username"].ToString();

}

}

[WebMethod]

public static void SaveScore(int score)

{

// Get username from session. If session is null, redirecting will already be handled on Page\_Load.

// This is a safety fallback.

string username = HttpContext.Current.Session["username"]?.ToString();

if (string.IsNullOrEmpty(username))

{

// This scenario should not be hit due to the Page\_Load check.

// You could throw an error or handle it as needed.

return;

}

int wpm = score; // Words per minute

int timePlayed = 60; // Store as seconds instead of "60s"

string cs = ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

// Adjusting the SQL command to match your table schema

string sql = @"INSERT INTO GameHistory (Username, GameName, Score, ExtraInfo, PlayedOn)

VALUES (@u, @g, @s, @t, GETDATE());";

using (SqlCommand cmd = new SqlCommand(sql, con))

{

cmd.Parameters.Add("@u", SqlDbType.NVarChar, 100).Value = username;

cmd.Parameters.Add("@g", SqlDbType.NVarChar, 50).Value = "Typing Speed";

cmd.Parameters.Add("@s", SqlDbType.Int).Value = wpm; // WPM stored in Score

cmd.Parameters.Add("@t", SqlDbType.NVarChar, 100).Value = timePlayed + "s"; // Store time as "60s" or "60 seconds" if you prefer

con.Open();

cmd.ExecuteNonQuery();

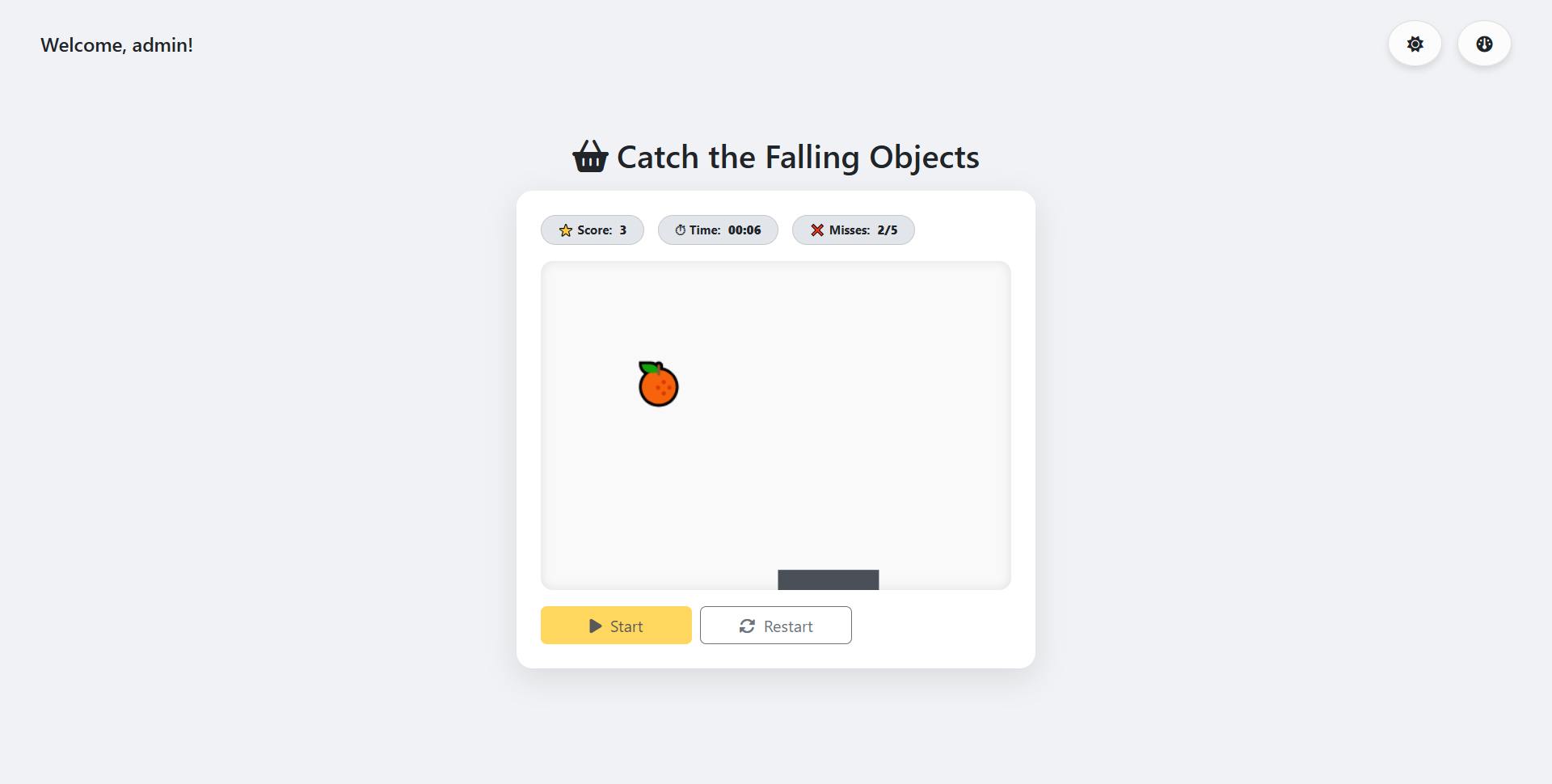
}

}

}

}

}

****

CatchGame.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="CatchGame.aspx.cs" Inherits="myprj.CatchGame" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<meta charset="UTF-8" />

<title>GameHub | Catch the Falling Objects</title>

<meta name="viewport" content="width=device-width, initial-scale=1" />

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css" rel="stylesheet" />

<style>

/\* Base Light Theme \*/

body {

background: #f0f2f5;

color: #212529; /\* Darker text for contrast \*/

font-family: 'Segoe UI', sans-serif;

padding-top: 20px;

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

flex-direction: column;

transition: background-color 0.5s ease;

}

header {

width: 100%;

padding: 20px 40px;

display: flex;

justify-content: space-between;

align-items: center;

position: absolute;

top: 0;

left: 0;

}

.welcome-user {

font-size: 1.2rem;

font-weight: 500;

}

.welcome-user span {

font-weight: bold;

}

.header-icons {

display: flex;

align-items: center;

gap: 15px;

}

/\* Updated button styles for a more "real world" feel \*/

.header-icons .btn {

border: none;

box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

transition: transform 0.2s ease, box-shadow 0.2s ease, background-color 0.2s ease;

border-radius: 50px; /\* Fully rounded buttons \*/

padding: 10px 18px; /\* Larger padding for a bigger feel \*/

min-width: unset; /\* Override default button style \*/

}

.header-icons .btn:hover {

transform: translateY(-2px);

box-shadow: 0 6px 10px rgba(0, 0, 0, 0.15);

}

.header-icons .btn-outline-secondary {

background-color: rgba(255, 255, 255, 0.8);

color: #212529;

border: 1px solid #ddd;

}

.header-icons .btn-outline-secondary:hover {

background-color: #fff;

color: #212529;

}

.game-wrap {

max-width: 780px;

width: 100%;

margin: auto;

}

.cardish {

background: #ffffff;

border-radius: 16px;

padding: 24px;

box-shadow: 0 10px 30px rgba(0, 0, 0, .1);

transition: background-color 0.5s ease, box-shadow 0.5s ease;

}

.status {

gap: 14px;

}

.badge-soft {

background: #e2e6ea;

border: 1px solid #c5c8cb;

color: #212529;

transition: background-color 0.5s ease, color 0.5s ease;

}

#gameCanvas {

background: #f9f9f9;

border-radius: 12px;

box-shadow: inset 0 0 10px rgba(0, 0, 0, 0.1);

display: block;

width: 100%;

touch-action: none;

}

.controls .btn {

min-width: 150px;

}

.game-over-message {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

text-align: center;

font-size: 2rem;

color: #212529;

text-shadow: 2px 2px 4px rgba(255, 255, 255, 0.5);

z-index: 10;

}

/\* Dark Theme Styles \*/

body.dark-theme {

background: #121212;

color: #e0e0e0;

}

body.dark-theme .cardish {

background: #1e1e1e;

box-shadow: 0 10px 30px rgba(0, 0, 0, .5);

}

body.dark-theme .badge-soft {

background: #333333;

border: 1px solid #444444;

color: #e0e0e0;

}

/\* Dark theme button styles \*/

body.dark-theme .header-icons .btn-outline-secondary {

background-color: rgba(60, 60, 60, 0.8);

color: #ddd;

border: 1px solid #444;

}

body.dark-theme .header-icons .btn-outline-secondary:hover {

background-color: #3e3e3e;

color: #fff;

}

body.dark-theme .game-over-message {

color: #e0e0e0;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);

}

body.dark-theme #gameCanvas {

background: #252526;

box-shadow: inset 0 0 10px rgba(0, 0, 0, 0.5);

}

</style>

</head>

<body>

<form id="form1" runat="server">

<asp:ScriptManager ID="ScriptManager1" runat="server" EnablePageMethods="true"></asp:ScriptManager>

<header>

<div class="welcome-user">

Welcome, <asp:Literal ID="lblUsername" runat="server" />!

</div>

<div class="header-icons">

<button type="button" class="btn btn-outline-secondary rounded-circle" id="themeToggleBtn">

<i id="themeIcon" class="fa-solid fa-moon"></i>

</button>

<a href="Dashboard.aspx" class="btn btn-outline-secondary rounded-circle">

<i class="fa-solid fa-gauge"></i>

</a>

</div>

</header>

<div class="game-wrap">

<div class="d-flex justify-content-center align-items-center mb-3">

<h2 class="m-0"><i class="fa-solid fa-basket-shopping me-2"></i>Catch the Falling Objects</h2>

</div>

<div class="cardish position-relative">

<div class="d-flex flex-wrap status mb-3">

<span class="badge rounded-pill badge-soft px-3 py-2">

⭐ Score: <strong id="scoreTxt" class="ms-1">0</strong>

</span>

<span class="badge rounded-pill badge-soft px-3 py-2">

⏱ Time: <strong id="timerTxt" class="ms-1">00:00</strong>

</span>

<span class="badge rounded-pill badge-soft px-3 py-2">

❌ Misses: <strong id="missesTxt" class="ms-1">0/5</strong>

</span>

</div>

<canvas id="gameCanvas"></canvas>

<div id="gameOverMessage" class="game-over-message d-none">

Game Over! Your Score: <span id="finalScore">0</span>

<br><small class="d-block mt-2">Saving score...</small>

</div>

<div class="controls d-flex gap-2 mt-3">

<button type="button" class="btn btn-warning" id="btnStart">

<i class="fa-solid fa-play me-1"></i> Start

</button>

<button type="button" class="btn btn-outline-secondary" id="btnRestart" disabled>

<i class="fa-solid fa-rotate me-1"></i> Restart

</button>

</div>

</div>

</div>

<div class="toast-container position-fixed bottom-0 end-0 p-3">

<div id="saveToast" class="toast" role="alert" aria-live="assertive" aria-atomic="true">

<div class="toast-header">

<strong class="me-auto">GameHub</strong>

<button type="button" class="btn-close" data-bs-dismiss="toast" aria-label="Close"></button>

</div>

<div class="toast-body bg-success text-white">

Score saved successfully! Redirecting...

</div>

</div>

</div>

</form>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

<script>

document.addEventListener('DOMContentLoaded', () => {

// --- Game State & Configuration ---

const gameState = {

score: 0,

misses: 0,

isGameRunning: false,

lastUpdateTime: 0,

timerInterval: null,

gameTime: 0,

objectsToCatch: [],

player: { x: 0, width: 100, height: 20 },

maxMisses: 5,

spawnObjectInterval: null

};

// An array of fruit emojis to randomly select from

const FRUITS = ['🍎', '🍊', '🍌', '🍇', '🍓', '🍍', '🍉', '🍑', '🍒', '🥝'];

// --- UI Elements ---

const ui = {

canvas: document.getElementById('gameCanvas'),

scoreTxt: document.getElementById('scoreTxt'),

timerTxt: document.getElementById('timerTxt'),

missesTxt: document.getElementById('missesTxt'),

btnStart: document.getElementById('btnStart'),

btnRestart: document.getElementById('btnRestart'),

gameOverMessage: document.getElementById('gameOverMessage'),

finalScore: document.getElementById('finalScore'),

themeToggleBtn: document.getElementById('themeToggleBtn'),

themeIcon: document.getElementById('themeIcon')

};

const ctx = ui.canvas.getContext('2d');

// --- Utility Functions ---

function getRandomInt(min, max) {

return Math.floor(Math.random() \* (max - min + 1)) + min;

}

function isCollision(rect1, rect2) {

return rect1.x < rect2.x + rect2.width &&

rect1.x + rect1.width > rect2.x &&

rect1.y < rect2.y + rect2.height &&

rect1.y + rect1.height > rect2.y;

}

// --- Theme Toggling ---

const theme = {

init: () => {

const savedTheme = localStorage.getItem('theme');

if (savedTheme === 'dark') {

document.body.classList.add('dark-theme');

ui.themeIcon.classList.remove('fa-sun');

ui.themeIcon.classList.add('fa-moon');

} else {

document.body.classList.remove('dark-theme');

ui.themeIcon.classList.remove('fa-moon');

ui.themeIcon.classList.add('fa-sun');

}

},

toggle: () => {

document.body.classList.toggle('dark-theme');

if (document.body.classList.contains('dark-theme')) {

localStorage.setItem('theme', 'dark');

ui.themeIcon.classList.remove('fa-sun');

ui.themeIcon.classList.add('fa-moon');

} else {

localStorage.setItem('theme', 'light');

ui.themeIcon.classList.remove('fa-moon');

ui.themeIcon.classList.add('fa-sun');

}

}

};

// --- Game Logic ---

const game = {

init: () => {

// Set canvas dimensions dynamically

ui.canvas.width = ui.canvas.parentElement.clientWidth;

ui.canvas.height = Math.round(ui.canvas.width \* 0.7);

gameState.player.x = (ui.canvas.width - gameState.player.width) / 2;

game.updateUI();

game.draw();

// Event Listeners

ui.btnStart.addEventListener('click', game.start);

ui.btnRestart.addEventListener('click', game.restart);

window.addEventListener('resize', game.handleResize);

ui.canvas.addEventListener('mousemove', game.handleInput);

ui.canvas.addEventListener('touchmove', game.handleTouch, { passive: false });

ui.themeToggleBtn.addEventListener('click', theme.toggle);

// Initialize theme on page load

theme.init();

},

handleResize: () => {

ui.canvas.width = ui.canvas.parentElement.clientWidth;

ui.canvas.height = Math.round(ui.canvas.width \* 0.7);

game.draw();

},

handleInput: (e) => {

if (!gameState.isGameRunning) return;

const rect = ui.canvas.getBoundingClientRect();

const mouseX = e.clientX - rect.left;

// Keep the player within the canvas bounds

gameState.player.x = mouseX - gameState.player.width / 2;

gameState.player.x = Math.max(0, Math.min(ui.canvas.width - gameState.player.width, gameState.player.x));

},

handleTouch: (e) => {

if (!gameState.isGameRunning) return;

e.preventDefault(); // Prevent scrolling

const rect = ui.canvas.getBoundingClientRect();

const touchX = e.touches[0].clientX - rect.left;

gameState.player.x = touchX - gameState.player.width / 2;

gameState.player.x = Math.max(0, Math.min(ui.canvas.width - gameState.player.width, gameState.player.x));

},

start: () => {

if (gameState.isGameRunning) return;

gameState.isGameRunning = true;

gameState.score = 0;

gameState.misses = 0;

gameState.gameTime = 0;

gameState.objectsToCatch = [];

ui.gameOverMessage.classList.add('d-none');

ui.btnStart.disabled = true;

ui.btnRestart.disabled = false;

game.updateUI();

game.startTimer();

requestAnimationFrame(game.loop);

gameState.spawnObjectInterval = setInterval(game.spawnObject, 1000); // Spawn an object every second

},

restart: () => {

game.end();

game.start();

},

end: () => {

if (!gameState.isGameRunning) return; // Prevent multiple calls

gameState.isGameRunning = false;

ui.btnStart.disabled = false;

ui.btnRestart.disabled = false;

clearInterval(gameState.timerInterval);

clearInterval(gameState.spawnObjectInterval);

ui.gameOverMessage.classList.remove('d-none');

ui.finalScore.textContent = gameState.score;

game.saveScoreToServer();

},

// New function to save score to the server

saveScoreToServer: async () => {

const data = {

score: gameState.score,

misses: gameState.misses,

time: gameState.gameTime

};

try {

const response = await fetch('CatchGame.aspx/SaveScore', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(data)

});

if (response.ok) {

// Show a success toast and then redirect

const toast = new bootstrap.Toast(document.getElementById('saveToast'));

toast.show();

setTimeout(() => {

window.location.href = 'Leaderboard.aspx';

}, 3000); // Redirect after 3 seconds

} else {

console.error('Failed to save score.');

ui.gameOverMessage.innerHTML += '<br><small>Error saving score. Please try again.</small>';

}

} catch (error) {

console.error('An error occurred:', error);

ui.gameOverMessage.innerHTML += '<br><small>Connection error. Could not save score.</small>';

}

},

startTimer: () => {

gameState.timerInterval = setInterval(() => {

gameState.gameTime++;

game.updateUI();

}, 1000);

},

spawnObject: () => {

const randomFruit = FRUITS[Math.floor(Math.random() \* FRUITS.length)];

let object = {

x: getRandomInt(20, ui.canvas.width - 20),

y: -20,

size: 40,

speed: getRandomInt(2, 5),

fruit: randomFruit

};

gameState.objectsToCatch.push(object);

},

loop: (timestamp) => {

const deltaTime = timestamp - (gameState.lastUpdateTime || timestamp);

gameState.lastUpdateTime = timestamp;

game.update(deltaTime);

game.draw();

if (gameState.isGameRunning) {

requestAnimationFrame(game.loop);

}

},

update: (deltaTime) => {

// Update object positions

gameState.objectsToCatch.forEach(object => {

object.y += object.speed;

});

// Check for collisions and misses

const playerRect = {

x: gameState.player.x,

y: ui.canvas.height - gameState.player.height,

width: gameState.player.width,

height: gameState.player.height

};

for (let i = gameState.objectsToCatch.length - 1; i >= 0; i--) {

const object = gameState.objectsToCatch[i];

const objectRect = {

x: object.x - object.size / 2,

y: object.y - object.size / 2,

width: object.size,

height: object.size

};

if (isCollision(playerRect, objectRect)) {

gameState.score++;

gameState.objectsToCatch.splice(i, 1);

} else if (object.y > ui.canvas.height + object.size) {

gameState.misses++;

gameState.objectsToCatch.splice(i, 1);

}

}

game.updateUI();

if (gameState.misses >= gameState.maxMisses) {

game.end();

}

},

draw: () => {

ctx.clearRect(0, 0, ui.canvas.width, ui.canvas.height);

// Draw the player (basket)

ctx.fillStyle = document.body.classList.contains('dark-theme') ? '#555' : '#495057';

ctx.fillRect(gameState.player.x, ui.canvas.height - gameState.player.height, gameState.player.width, gameState.player.height);

// Draw falling objects as emojis

ctx.textAlign = 'center';

ctx.textBaseline = 'middle';

gameState.objectsToCatch.forEach(object => {

ctx.font = `${object.size}px serif`;

ctx.fillText(object.fruit, object.x, object.y);

});

},

updateUI: () => {

ui.scoreTxt.textContent = gameState.score;

ui.missesTxt.textContent = `${gameState.misses}/${gameState.maxMisses}`;

const minutes = Math.floor(gameState.gameTime / 60).toString().padStart(2, '0');

const seconds = (gameState.gameTime % 60).toString().padStart(2, '0');

ui.timerTxt.textContent = `${minutes}:${seconds}`;

}

};

game.init();

});

</script>

</body>

</html>

CatchGame.aspx.cs

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web;

using System.Web.Services;

using System.Web.UI;

namespace myprj

{

public partial class CatchGame : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

// Check if a user is logged in

if (Session["username"] == null)

{

// If not logged in, redirect to the login page

Response.Redirect("Login.aspx");

}

else

{

// Set the username in a literal control for display on the page

lblUsername.Text = Session["username"].ToString();

}

}

[WebMethod]

public static void SaveScore(int score, int misses, int time)

{

string username = HttpContext.Current.Session["username"]?.ToString();

// Safety check: this should be handled by Page\_Load, but it's good practice.

if (string.IsNullOrEmpty(username))

{

// You could log an error or return a specific status code here.

return;

}

string gameName = "Catch the Falling Objects";

string extraInfo = $"Time: {time}s, Misses: {misses}";

string cs = ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

// Note: The Moves column can store the score, and ExtraInfo can store misses/time.

string sql = @"INSERT INTO GameHistory (Username, GameName, Score, Moves, ExtraInfo, PlayedOn)

VALUES (@u, @g, @s, @m, @e, GETDATE());";

using (SqlCommand cmd = new SqlCommand(sql, con))

{

cmd.Parameters.Add("@u", SqlDbType.NVarChar, 100).Value = username;

cmd.Parameters.Add("@g", SqlDbType.NVarChar, 50).Value = gameName;

cmd.Parameters.Add("@s", SqlDbType.Int).Value = score; // WPM stored in Score

cmd.Parameters.Add("@m", SqlDbType.Int).Value = score; // You can duplicate score in moves for this game

cmd.Parameters.Add("@e", SqlDbType.NVarChar, 100).Value = extraInfo;

con.Open();

cmd.ExecuteNonQuery();

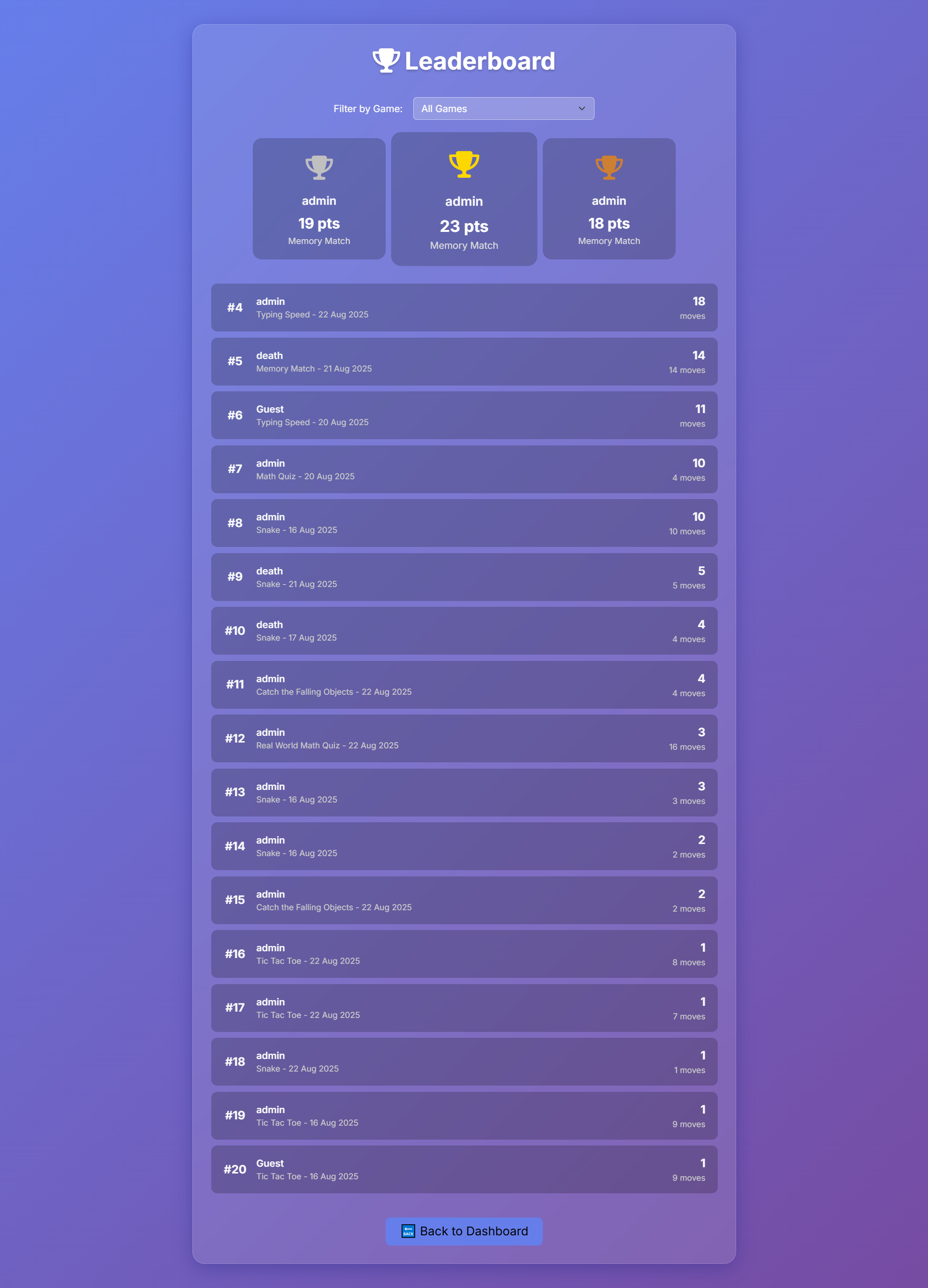
}

}

}

}

}

****

Leaderboard.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Leaderboard.aspx.cs" Inherits="myprj.Leaderboard" %>

<!DOCTYPE html>

<html>

<head runat="server">

<title>Leaderboard - GameHub</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" />

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css" />

<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600;700&display=swap" rel="stylesheet">

<style>

body {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

font-family: 'Inter', sans-serif;

color: #f8f9fa;

}

.leaderboard-container {

max-width: 900px;

margin: 40px auto;

background: rgba(255, 255, 255, 0.1);

border-radius: 20px;

box-shadow: 0 8px 32px 0 rgba(31, 38, 135, 0.37);

backdrop-filter: blur(10px);

-webkit-backdrop-filter: blur(10px);

border: 1px solid rgba(255, 255, 255, 0.18);

padding: 30px;

}

.leaderboard-header {

text-align: center;

margin-bottom: 30px;

}

.leaderboard-title {

font-size: 2.5rem;

font-weight: 700;

color: #fff;

text-shadow: 0 2px 4px rgba(0,0,0,0.2);

}

.filter-section {

display: flex;

justify-content: center;

align-items: center;

gap: 10px;

margin-bottom: 30px;

}

.filter-section .form-select {

max-width: 300px;

background-color: rgba(255, 255, 255, 0.2);

color: #fff;

border: 1px solid rgba(255, 255, 255, 0.5);

}

.filter-section .form-select option {

color: #333;

}

/\* Top 3 Players Styling \*/

.top-players {

display: flex;

justify-content: center;

align-items: flex-end;

gap: 20px;

margin-bottom: 40px;

flex-wrap: wrap;

}

.player-card {

background: rgba(0, 0, 0, 0.2);

border-radius: 15px;

padding: 20px;

text-align: center;

width: 220px;

position: relative;

transition: transform 0.3s ease, box-shadow 0.3s ease;

}

.player-card:hover {

transform: translateY(-10px);

box-shadow: 0 12px 24px rgba(0,0,0,0.3);

}

.player-card .rank-icon {

font-size: 2.5rem;

margin-bottom: 10px;

}

.player-card.rank-1 .rank-icon { color: #FFD700; } /\* Gold \*/

.player-card.rank-2 .rank-icon { color: #C0C0C0; } /\* Silver \*/

.player-card.rank-3 .rank-icon { color: #CD7F32; } /\* Bronze \*/

.player-card.rank-1 { order: 2; transform: scale(1.1); }

.player-card.rank-2 { order: 1; }

.player-card.rank-3 { order: 3; }

.player-name {

font-size: 1.2rem;

font-weight: 600;

margin-bottom: 5px;

white-space: nowrap;

overflow: hidden;

text-overflow: ellipsis;

}

.player-score {

font-size: 1.5rem;

font-weight: 700;

color: #fff;

}

.player-game {

font-size: 0.9rem;

color: #ddd;

}

/\* Leaderboard List Styling \*/

.leaderboard-list .list-group-item {

background: rgba(0, 0, 0, 0.2);

border: none;

color: #f8f9fa;

display: flex;

align-items: center;

padding: 15px 20px;

margin-bottom: 10px;

border-radius: 10px;

transition: background-color 0.2s ease;

}

.leaderboard-list .list-group-item:hover {

background: rgba(0, 0, 0, 0.4);

}

.rank {

font-size: 1.2rem;

font-weight: 700;

min-width: 40px;

text-align: center;

margin-right: 15px;

}

.player-info {

flex-grow: 1;

}

.player-info .name {

font-weight: 600;

}

.player-info .game {

font-size: 0.85rem;

color: #ccc;

}

.score-info {

text-align: right;

min-width: 100px;

}

.score-info .score {

font-size: 1.2rem;

font-weight: 700;

}

.score-info .moves {

font-size: 0.85rem;

color: #ccc;

}

.btn-back {

display: block;

width: fit-content;

margin: 30px auto 0 auto;

background: #667eea;

border: none;

transition: background-color 0.3s ease;

}

.btn-back:hover {

background: #764ba2;

}

</style>

</head>

<body>

<form id="form1" runat="server">

<div class="leaderboard-container">

<div class="leaderboard-header">

<div class="leaderboard-title">

<i class="fas fa-trophy me-2"></i>Leaderboard

</div>

</div>

<!-- Game Filter Dropdown -->

<div class="filter-section">

<label for="ddlGameFilter" class="form-label mb-0 me-2">Filter by Game:</label>

<asp:DropDownList ID="ddlGameFilter" runat="server" CssClass="form-select" AutoPostBack="true" OnSelectedIndexChanged="ddlGameFilter\_SelectedIndexChanged"></asp:DropDownList>

</div>

<!-- Top 3 Players -->

<div class="top-players">

<asp:Repeater ID="rptTopPlayers" runat="server">

<ItemTemplate>

<div class='player-card rank-<%# Eval("Rank") %>'>

<div class="rank-icon">

<i class="fas fa-trophy"></i>

</div>

<div class="player-name" title='<%# Eval("Username") %>'><%# Eval("Username") %></div>

<div class="player-score"><%# Eval("Score", "{0:N0}") %> pts</div>

<div class="player-game"><%# Eval("GameName") %></div>

</div>

</ItemTemplate>

</asp:Repeater>

</div>

<!-- Message for no results -->

<asp:Panel ID="pnlNoResults" runat="server" Visible="false" CssClass="text-center my-5">

<h4><i class="fas fa-ghost me-2"></i>No Results Found</h4>

<p>There are no game records for the selected filter.</p>

</asp:Panel>

<!-- Rest of the Leaderboard -->

<div class="leaderboard-list">

<asp:Repeater ID="rptLeaderboard" runat="server">

<HeaderTemplate>

<ul class="list-group">

</HeaderTemplate>

<ItemTemplate>

<li class="list-group-item">

<div class="rank">#<%# Eval("Rank") %></div>

<div class="player-info">

<div class="name"><%# Eval("Username") %></div>

<div class="game"><%# Eval("GameName") %> - <%# Eval("PlayedOn", "{0:dd MMM yyyy}") %></div>

</div>

<div class="score-info">

<div class="score"><%# Eval("Score", "{0:N0}") %></div>

<div class="moves"><%# Eval("Moves") %> moves</div>

</div>

</li>

</ItemTemplate>

<FooterTemplate>

</ul>

</FooterTemplate>

</asp:Repeater>

</div>

<asp:Button ID="btnBack" runat="server" Text="🔙 Back to Dashboard" CssClass="btn btn-light btn-lg btn-back px-4 py-2" OnClick="btnBack\_Click" />

</div>

</form>

</body>

</html>

Leaderboard.aspx.cs

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace myprj

{

public partial class Leaderboard : Page

{

private readonly string connectionString = System.Configuration.ConfigurationManager.ConnectionStrings["GameHubDBConnectionString"].ConnectionString;

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

PopulateGameFilter();

LoadLeaderboard();

}

}

private void PopulateGameFilter()

{

using (SqlConnection con = new SqlConnection(connectionString))

{

// Query to get a distinct list of games that have been played

string query = "SELECT DISTINCT GameName FROM GameHistory ORDER BY GameName ASC";

SqlDataAdapter da = new SqlDataAdapter(query, con);

DataTable dt = new DataTable();

da.Fill(dt);

ddlGameFilter.DataSource = dt;

ddlGameFilter.DataTextField = "GameName";

ddlGameFilter.DataValueField = "GameName";

ddlGameFilter.DataBind();

// Add an item to show all games

ddlGameFilter.Items.Insert(0, new ListItem("All Games", "All"));

}

}

private void LoadLeaderboard()

{

using (SqlConnection con = new SqlConnection(connectionString))

{

// Base query with ranking. We order by Score descending to get the highest scores first.

string query = @"

SELECT

ROW\_NUMBER() OVER (ORDER BY Score DESC) as Rank,

Username,

GameName,

Score,

Moves,

Winner,

ExtraInfo,

PlayedOn

FROM GameHistory";

// Apply filter if a specific game is selected

string selectedGame = ddlGameFilter.SelectedValue;

if (selectedGame != "All" && !string.IsNullOrEmpty(selectedGame))

{

query += " WHERE GameName = @GameName";

}

query += " ORDER BY Score DESC";

SqlDataAdapter da = new SqlDataAdapter(query, con);

// Add parameter for the game name to prevent SQL injection

if (selectedGame != "All" && !string.IsNullOrEmpty(selectedGame))

{

da.SelectCommand.Parameters.AddWithValue("@GameName", selectedGame);

}

DataTable dt = new DataTable();

da.Fill(dt);

if (dt.Rows.Count > 0)

{

pnlNoResults.Visible = false;

// Take the top 3 players for the special display cards

var topPlayersQuery = dt.AsEnumerable().Take(3);

if (topPlayersQuery.Any())

{

rptTopPlayers.DataSource = topPlayersQuery.CopyToDataTable();

rptTopPlayers.DataBind();

}

// Skip the top 3 players for the main list and take the next 17 (for a total of 20)

var remainingPlayersQuery = dt.AsEnumerable().Skip(3).Take(17);

if (remainingPlayersQuery.Any())

{

rptLeaderboard.DataSource = remainingPlayersQuery.CopyToDataTable();

rptLeaderboard.DataBind();

}

else

{

// If there are 3 or fewer players, there will be no remaining players.

rptLeaderboard.DataSource = null;

rptLeaderboard.DataBind();

}

}

else

{

// If no data, hide the repeaters and show a message

pnlNoResults.Visible = true;

rptTopPlayers.DataSource = null;

rptTopPlayers.DataBind();

rptLeaderboard.DataSource = null;

rptLeaderboard.DataBind();

}

}

}

protected void ddlGameFilter\_SelectedIndexChanged(object sender, EventArgs e)

{

// Reload the leaderboard data when the filter changes

LoadLeaderboard();

}

protected void btnBack\_Click(object sender, EventArgs e)

{

Response.Redirect("Dashboard.aspx");

}

}

}

# Bibliography

1. Asp.Net Documentation, Microsoft Docs, https://docs.microsoft.com/aspnet/
2. SQL Server Documentation, Microsoft Docs, https://docs.microsoft.com/sql/
3. Himanshu Solanki, *Gamehub Project Report*, Dr. V.R. Godhaniya I.T. College, Bhakta Kavi Narsinh Mehta University, Porbandar.
4. W3Schools, “ASP.NET Tutorial”, https://www.w3schools.com/aspnet/
5. TutorialsPoint, “SQL Server Tutorial”, https://www.tutorialspoint.com/sql\_server/
6. Kher, Dr. Dhaval, Head of Department, Dr. V.R. Godhaniya I.T. College – Project Guidance.
7. Pressman, Roger S., *Software Engineering: A Practitioner’s Approach*, McGraw Hill Education, 8th Edition.