SPACE INVADERS

By Mohssine Deflaoui Harrata

Word count: 1195

Introduction

This project is a modern web-based recreation of the classic *Space Invaders* game. It was developed using HTML, CSS, and JavaScript, structured into modular .mjs files for a clean separation of logic. The project includes a complete user system with registration, login, and player-specific high-score storage using browser localStorage. The game features smooth animation, responsive canvas scaling, and a dynamic interface that reacts to user actions such as logging in, playing, losing, and viewing rankings. The overall aim of this project was to combine front-end web design principles with interactive game programming, data persistence, and user session management resulting in a polished and functional browser-based experience.

Website Overview

The website is composed of multiple pages, each serving a specific purpose.
It maintains a consistent navigation bar, footer, and colour scheme throughout the entire site.

Page	**File Name**	**Description**
Home	index.html	Welcomes the user with a brief introduction to the game and provides navigation links.
Register	register.html	Allows new users to create an account with form validation and the storage of details in localStorage.
Login	login.html	Authenticates users by checking their credentials and starts a session using localStorage.

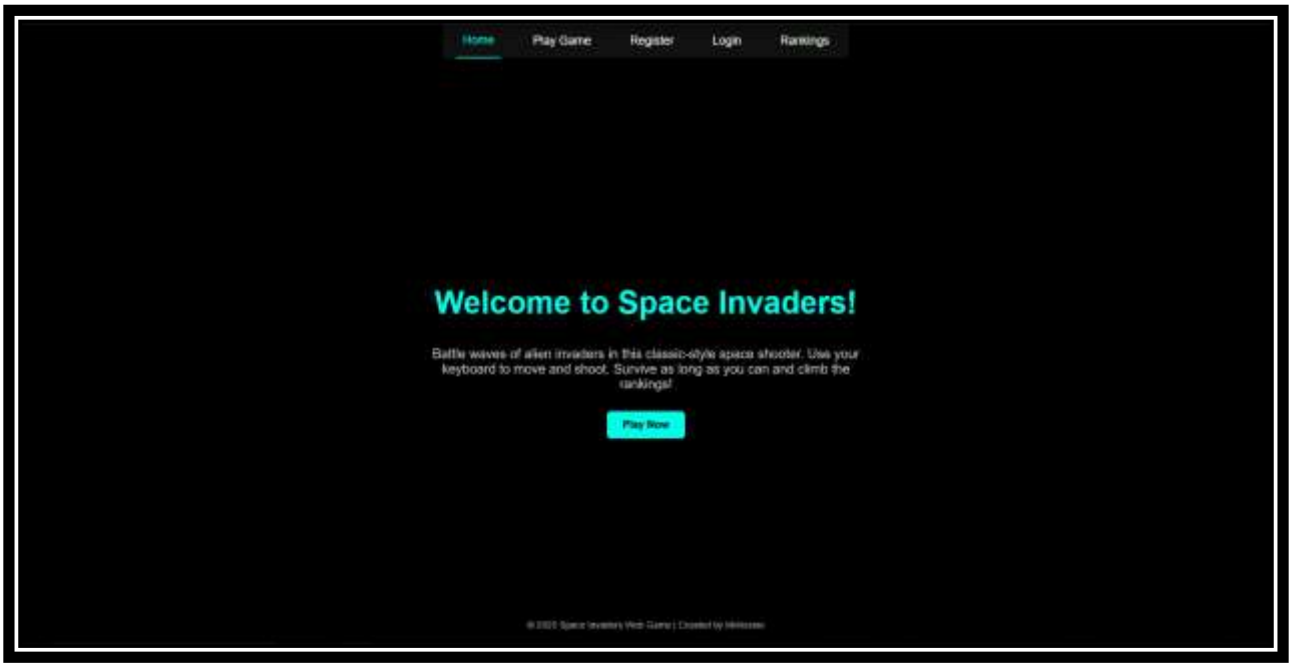
Game	game.html	Hosts the main interactive canvas-based game with dynamic animation and user score tracking.
Rankings	rankings.html	Displays a leaderboard of the top scores saved from all registered users.

Design and Layout

The overall design uses a dark sci-fi theme, with the background set to black and accents of bright cyan (#00ffea) to reflect a futuristic aesthetic.

Key design elements:

- A fixed navigation bar at the top of all pages for quick access.
- Footer text on every page displaying the copyright and author.
- Consistent font and colour scheme across all sections.
- Responsive layout, ensuring the canvas and forms adapt to different screen sizes.



Registration and Login Functionality

Registration

New users register on the register.html page.
The form validates:

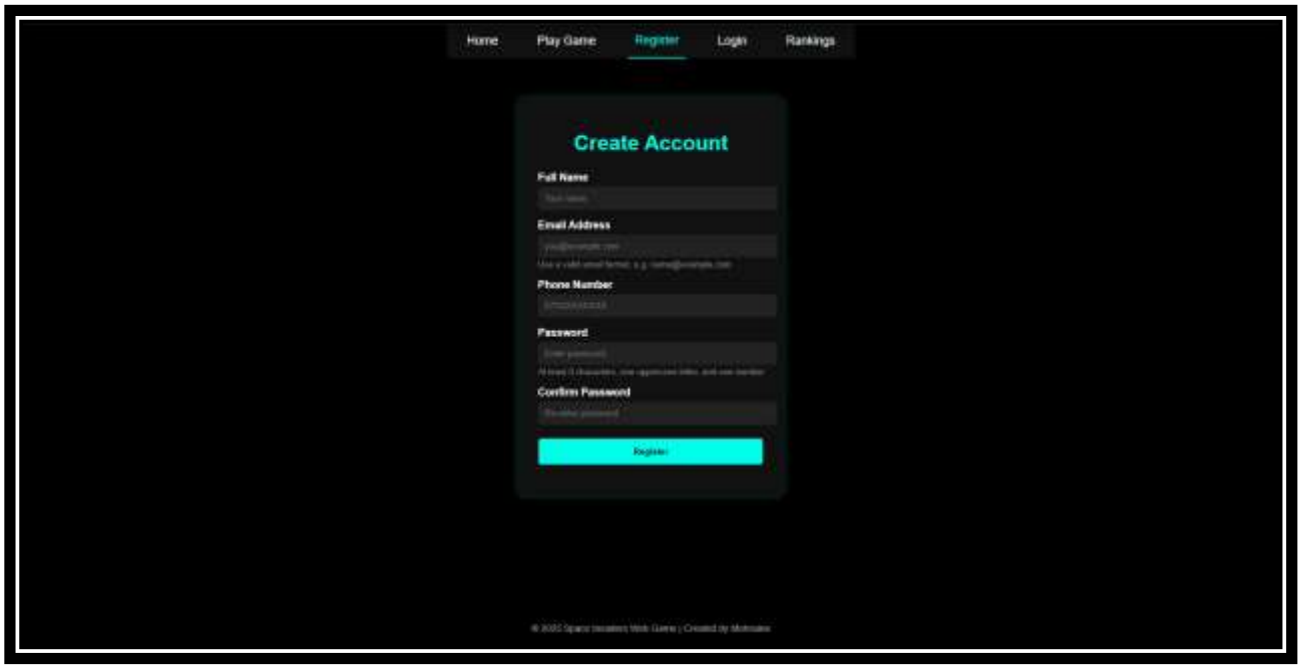
- Non-empty fields.
- Proper email format.
- Phone number starting with “07”.
- Password of at least 8 characters, containing one uppercase letter and one number.

Once validated, user details are stored in localStorage.
Duplicate registrations are prevented by checking existing emails.

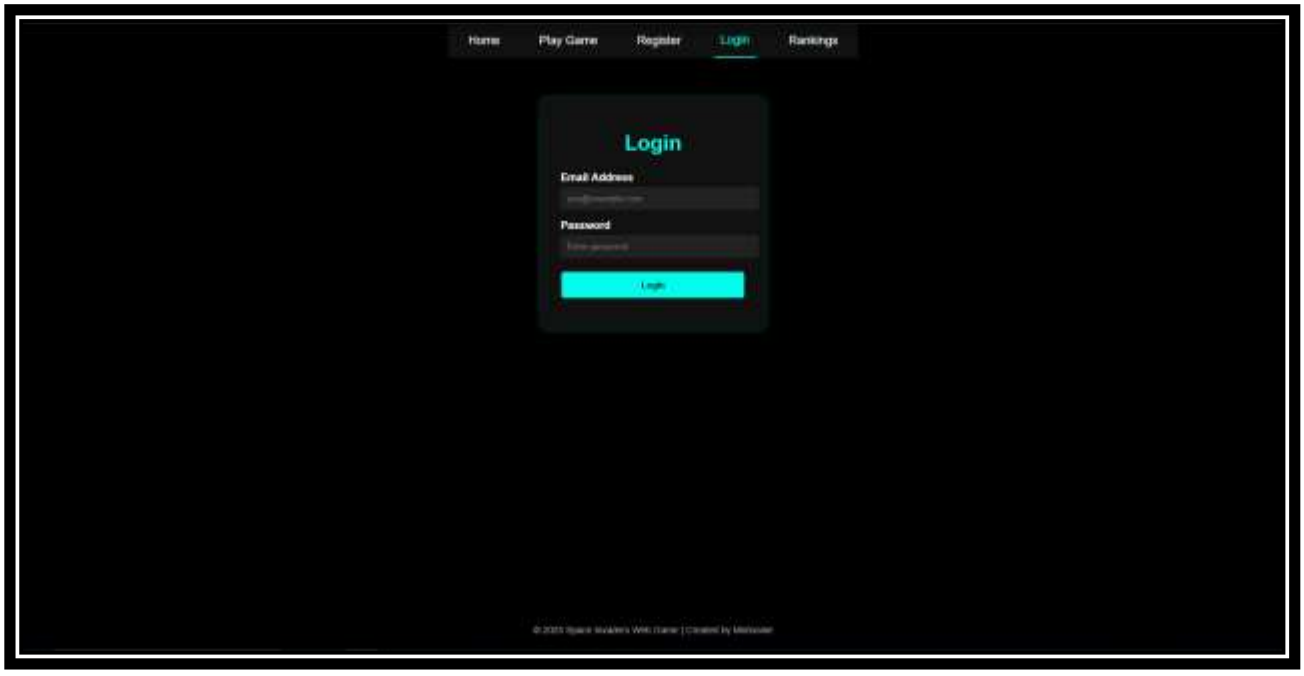
Login

The login.html page allows registered users to sign in.
If credentials match the data in localStorage, the user session is stored in LocalStorage and redirected to the game page.
Invalid credentials trigger an on-screen error message without reloading the page.

Registration Page (Screenshot):



Login Page (Screenshot):



Game Page and Core Mechanics

The game.html page hosts the canvas element where the game is rendered.
It uses modular JavaScript to structure all gameplay logic.

When the user logs in and opens this page:

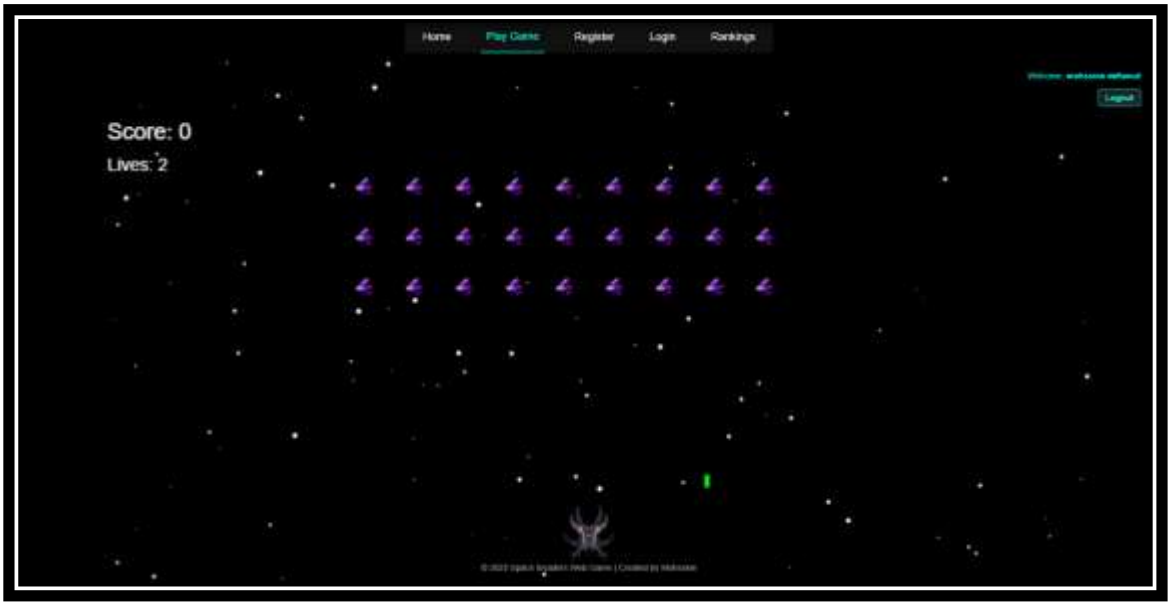
- The top-right corner displays:

"Welcome, [Username]" and a **Logout** button.

- The player's spaceship appears at the bottom of the screen.
- Pressing:
    - A → moves left
    - D → moves right
    - Space → fires a projectile (limited by cooldown to prevent spam)

Enemies (invaders) move horizontally in a grid pattern, occasionally descending.
They shoot green projectiles at random intervals.
The player loses a life when hit, and when all lives are gone, the **Game Over** screen appears with the player's final score.

Game in progress with HUD showing Score, Lives counter and Welcome message with Logout button visible in the top right (Screenshot):



Game Logic and Architecture

The project follows an object-oriented modular structure:

File	Responsibility
CanvasManager.mjs	Handles canvas resizing and scaling for all screen sizes.
Player.mjs	Controls player ship, movement, shooting, and lives.
Projectile.mjs	Defines player and enemy bullets, their positions, and movement.
Invader.mjs	Manages individual invaders and enemy grids.
BackgroundStar.mjs	Renders an animated starfield background for a space atmosphere.
InputHandler.mjs	Captures and updates keyboard input (A, D, Space).
Game.mjs	Main controller for animation, updates, collisions, score, and restarting.
gameUser.mjs	Displays session info (username, logout) and manages score saving.
rankings.mjs	Loads and displays top scores for all users in the leaderboard.

The modular approach improves maintainability and debugging, ensuring that each class has a single responsibility.

Scoring, Restart, and Data Persistence

When the player destroys an invader, points are added based on the invader type:

- Type A = 10 points
- Type B = 20 points
- Type C = 30 points

When all lives reach zero, the **Game Over** screen appears and:

- The current score is displayed.
- The system saves it using saveScoreToUser() in gameUser.mjs.
- If it is higher than the user's previous best, it updates their stored record in localStorage.

The player can press **R** or click anywhere to restart.
The game resets all objects (stars, grids, projectiles, player) but maintains a smooth transition without reloading the page.

Rankings Page

The rankings.html page lists all users and their highest scores.
It retrieves all players from LocalStorage, sorts them by score, and displays them in a responsive table.

Each row contains:
| Rank | Player | High Score |

The page uses alternating row colours for readability and automatically updates whenever new scores are saved.

Rankings table:

Validation and Error Handling

Validation was implemented both for **user input** and **game stability**.

Form Validation

- Prevents invalid registration input.
- Displays contextual hints and messages.
- Blocks duplicate email registration.

Game Logic Validation

- Adds a shooting cooldown (300ms) to prevent bullet spam.
- Uses deltaTime clamping to prevent the “grid flying down” bug.
- Prevents the game from running if no player session is active.
- Automatically redirects to login.html if session data is missing.

Session and Score Safety

- Uses try...catch when accessing Localstorage to avoid corrupted data crashes.
- Ensures only logged-in players can save scores or access rankings.

Example of validation error message on Register (Screenshot):

Visual Design and User Experience

The design emphasise readability, contrast, and atmosphere.
The neon-cyan colour highlights interactive elements like buttons, active links, and HUD text.

Design choices:

- Minimal clutter for immersion.
- Clear fonts (Arial, sans-serif).
- Intuitive layout with fixed navbar and footer.
- Consistent UI elements across all pages.

The starfield animation adds visual depth, giving the game a “space” environment even when in an idle state. We can see this in the image provided in the Game Page and Core Mechanics section

Testing Process

Testing was done throughout development.

Test Area	**Description**
Registration	Checked input validation and duplicate detection.
Login	Tested correct/incorrect credentials.
Gameplay	Verified movement, shooting, collisions, and score updates.
Restart	Confirmed full reset of stars, enemies, and player state.
Session Handling	Ensured non-logged-in users can’t access the game.
Rankings	Verified data persistence and correct sorting.

Evaluation

Strengths

- Fully modular architecture using ES6 modules.
- Smooth and responsive canvas animation.
- Secure login and persistent score storage.
- Clear, consistent interface across all pages.
- Robust error handling and input validation.

Weaknesses / Improvements

- Could add audio effects for shooting and explosions.
- Future versions could introduce levels or enemy variations.
- A backend database (like Firebase) would make scores global and shareable.
- Adding mobile touch controls would expand accessibility.

Conclusion

The Space Invaders Web Game successfully combines web technologies and interactive programming to deliver an engaging user experience.

It demonstrates:

- Proficiency in HTML, CSS, and JavaScript.
- Strong understanding of modular programming.
- Integration of user session handling and data persistence.
- Clear and consistent visual design principles.

This project not only replicates the nostalgic gameplay of the original Space Invaders but modernises it for web browsers with user accounts, rankings, and smooth, responsive visuals.