LỜI CẢM ƠN

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

**MỤC LỤC**

Contents

[Chương 1. TỔNG QUAN VỀ ĐỀ TÀI 5](#_Toc148631324)

[1.1. Giới thiệu về đề tài 5](#_Toc148631325)

[1.2. Mục tiêu đề tài 6](#_Toc148631326)

[1.3. Mục đích đề tài 6](#_Toc148631327)

[1.4. Công nghệ sử dụng 6](#_Toc148631328)

[1.4.1. Python 6](#_Toc148631329)

[1.4.2. HTML 8](#_Toc148631330)

[1.4.3. CSS 9](#_Toc148631331)

[1.4.4. JavaScript 11](#_Toc148631332)

[1.4.5. Flask 12](#_Toc148631333)

[1.4.6. Flask-SocketIO 14](#_Toc148631334)

[1.4.7. Socket.IO 15](#_Toc148631335)

[Chương 2. PHÂN TÍCH BÀI TOÁN 16](#_Toc148631336)

[2.1. Mô tả các chức năng của ứng dụng 16](#_Toc148631337)

[2.2. Hướng xây dựng 17](#_Toc148631338)

[Chương 3. THỰC THI CHƯƠNG TRÌNH 19](#_Toc148631339)

[3.1. Các giao diện chương trình 19](#_Toc148631340)

[3.1.1. Giao diện bắt đầu trò chơi 19](#_Toc148631341)

[3.2. Ưu và nhược điểm của trò chơi 23](#_Toc148631342)

[3.2.1. Ưu điểm 23](#_Toc148631343)

[3.2.2. Nhược điểm: 24](#_Toc148631344)

[3.3. Kết luận và hướng phát triển 24](#_Toc148631345)

MỞ ĐẦU

Ứng dụng web đang ngày càng trở nên phổ biến và đa dạng, và việc phát triển trò chơi trực tuyến trên nền tảng web là một trong những xu hướng thú vị trong lĩnh vực công nghệ thông tin. Trò chơi trực tuyến không chỉ mang lại trải nghiệm giải trí cho người chơi, mà còn thúc đẩy sự sáng tạo và phát triển kỹ năng lập trình.

Chúng em đã chọn chủ đề để tìm hiểu cách xây dựng một trò chơi trực tuyến thú vị và thử thách. Trong đề tài này, chúng ta sẽ khám phá cách sử dụng các công nghệ như Socket.IO để tạo kết nối thời gian thực giữa người chơi, HTML và CSS để thiết kế giao diện trò chơi hấp dẫn, và Python để viết mã máy chủ quản lý trò chơi.

Bằng cách thực hiện dự án này, chúng ta sẽ hiểu rõ hơn về cách tạo một trò chơi đa người trực tuyến, cách quản lý thông tin giữa máy chủ và người chơi, và cách sử dụng các công cụ phát triển web để tạo ra một giao diện trực quan cho trò chơi.

Chúng em sẽ bắt đầu bằng việc nghiên cứu cơ sở lý thuyết về các công nghệ cơ bản mà chúng ta sẽ sử dụng và sau đó tiến hành phát triển trò chơi. Bên cạnh đó, chúng em cũng sẽ thảo luận về các thách thức có thể gặp phải trong quá trình phát triển và cách giải quyết chúng.

# TỔNG QUAN VỀ ĐỀ TÀI

## Giới thiệu về đề tài

Trò chơi Wonder Words là một ứng dụng trò chơi mạng thú vị mô phỏng cuộc thi trò chơi Wheel of Fortune trong đó các người chơi đua thời gian thực. Trò chơi này cho phép người chơi tham gia thông qua trình duyệt web và cùng nhau giải quyết các từ vựng bằng cách đoán các ký tự. Dưới đây là các quy tắc của trò chơi:

Chọn từ vựng ngẫu nhiên: Dự án chọn một từ vựng ngẫu nhiên từ danh sách để mô tả cho các người chơi.

Hiển thị từ vừng: Từ vựng được hiển thị với các ký tự chưa khớp thay thế bằng dấu gạch ngang và các khoảng trắng hiển thị như bình thường. Mô tả của từ vựng cũng được hiển thị.

Lượt chơi đưa thời gian: Trò chơi được chia thành các lượt chơi có thời gian hạn chế. Mỗi lượt chơi kéo dài 10 giây, và người chơi cố gắng đoán các ký tự trong từ vựng.

Đoán ký tự: Người chơi có thể đoán một ký tự trong từ vựng trong mỗi lượt chơi. Ký tự đúng sẽ được mở ra và điểm thưởng được cộng vào điểm của người chơi.

Xếp hạng người chơi: Điểm số của người chơi sẽ được tính và hiển thị trên bảng xếp hạng. Người chơi có điểm số cao nhất thắng cuộc.

Quản lý người chơi: Dự án quản lý danh sách người chơi và hiển thị tên của họ trên giao diện. Người chơi có thể chọn sẵn sàng để tham gia vào trò chơi.

Gửi tin nhắn: Hệ thống có khả năng gửi tin nhắn cho người chơi, thông báo về việc đoán đúng hoặc đoán sai.

Kết thúc trò chơi: Trò chơi kết thúc khi từ vựng được đoán đúng hoặc hết thời gian cho tất cả lượt chơi. Người chơi thắng cuộc sẽ được công bố và điểm số cuối cùng sẽ được hiển thị.

## Mục tiêu đề tài

Xây dựng được trò chơi Wonder Words bằng ngôn ngữ lập trình Python và kết hợp cùng các công nghệ khác để ứng dụng có thể chạy trên nền web. Nhờ đó tạo ra trò chơi hấp dẫn và có giao diện thân thiện với người dùng.

## Mục đích đề tài

* Nắm vững kiến thức công nghệ web.
* Thực hành lập trình
* Nâng cao kỹ năng tương tác mạng
* Tìm tìm hiểu về quản lý trạng thái của ứng dụng
* Tạo tạo giao diện thân thiện người dùng
* Học hỏi kỹ năng xử lý dữ liệu
* Làm việc nhóm và quản lý thời gian
* Kiểm thử và sửa lỗi
* Nắm vững kỹ năng trình bày và báo cáo
* Tạo ứng dụng thực tế

## Công nghệ sử dụng

### Python

Python là một ngôn ngữ lập trình mạnh mẽ, đa dụng và dễ học. Nó đã trở thành một trong những ngôn ngữ phổ biến nhất trên thế giới với một cộng đồng lập trình đông đảo và sự hỗ trợ rộng rãi trong nhiều lĩnh vực khác nhau.

Python ra đời vào những năm 1980 do Guido van Rossum tạo ra với mục tiêu thiết kế một ngôn ngữ lập trình dễ đọc và dễ viết mã.

Python được lấy tên theo bộ phim yêu thích của Guido, "Monty Python's Flying Circus," để thể hiện tính chất hài hước của ngôn ngữ.

Phiên bản đầu tiên, Python 0.9.0, ra mắt vào năm 1991.

Python 2.x được phát triển song song với Python 3.x, và phiên bản Python 3.0 chính thức ra mắt vào năm 2008. Python 2.x sau đó không còn được hỗ trợ từ năm 2020.

• Ưu điểm của Python:

Dễ học và đọc hiểu: Python được thiết kế với cú pháp đơn giản và rõ ràng, giúp người mới học lập trình dễ tiếp cận. Ngôn ngữ này sử dụng thụ đề (indentation) để định nghĩa khối mã thay vì dấu ngoặc như nhiều ngôn ngữ khác, làm cho mã nguồn dễ đọc.

Thư viện mạnh mẽ: Python có một hệ thống thư viện và module phong phú giúp giải quyết nhiều vấn đề khác nhau. NumPy và pandas giúp xử lý dữ liệu số học và dữ liệu dạng bảng, matplotlib cho phép tạo đồ thị, và thư viện requests cho phép tương tác với API web.

Đa dạng ứng dụng: Python có sự linh hoạt trong việc phát triển ứng dụng. Bạn có thể xây dựng ứng dụng web bằng Django hoặc Flask, phát triển trí tuệ nhân tạo với TensorFlow và PyTorch, thực hiện phân tích dữ liệu với SciPy và scikit-learn, và nhiều lĩnh vực khác.

Cộng đồng lập trình đông đảo: Python có một cộng đồng lập trình đông đảo và nhiệt tình. Điều này đồng nghĩa với việc bạn có thể dễ dàng tìm kiếm giúp đỡ, tài liệu và các thư viện bổ sung. Python cũng có sự hỗ trợ từ các trang web và diễn đàn lớn như Stack Overflow.

Khả năng mở rộng: Python hỗ trợ việc tích hợp mã từ các ngôn ngữ khác như C/C++, cho phép tối ưu hóa hiệu suất ứng dụng. Điều này làm cho Python phù hợp cho cả các dự án lớn và phức tạp.

• Nhược điểm của Python:

Chậm trong một số trường hợp: Python thường không nhanh bằng các ngôn ngữ lập trình biên dịch như C++ hoặc Java. Điều này có nghĩa rằng Python không phù hợp cho các ứng dụng đòi hỏi hiệu suất cực đỉnh như phát triển game hoặc phần mềm đồ họa 3D.

Phiên bản Python: Chuyển đổi giữa các phiên bản Python có thể gây ra sự không tương thích và vấn đề về quản lý phiên bản. Phiên bản Python 2.x cũ hơn không được hỗ trợ nữa, nhưng vẫn được sử dụng trong một số dự án cụ thể.

Ghi chú của thụ đề: Sử dụng thụ đề để xác định khối mã có thể làm cho mã nguồn dễ bị lỗi nếu không được thụ đề một cách đúng cách. Điều này đặc biệt quan trọng khi làm việc với mã nguồn lớn.

Hạn chế trong một số lĩnh vực đặc biệt: Python không phải lúc nào cũng là lựa chọn hàng đầu trong các lĩnh vực đặc biệt như phát triển ứng dụng di động (so với Swift hoặc Kotlin) hoặc phát triển ứng dụng nhúng (so với C/C++).

### HTML

HTML, hay HyperText Markup Language, là một ngôn ngữ đánh dấu tiêu chuẩn được sử dụng để tạo cấu trúc và hiển thị nội dung trên trình duyệt web. Được phát triển bởi Tim Berners-Lee và Robert Cailliau vào năm 1991, HTML đã trở thành ngôn ngữ cơ bản và không thể thiếu đối với phát triển web.

Cấu trúc và cú pháp: HTML hoạt động dựa trên nguyên tắc đánh dấu nội dung bằng các thẻ (tags) được đặt trong cặp dấu < >. Mỗi thẻ bao gồm một thẻ mở và một thẻ đóng, với nội dung nằm trong giữa. Ví dụ, <p> là thẻ mở và </p> là thẻ đóng cho đoạn văn bản.

Phân cấp: Cấu trúc của trang HTML được tổ chức thành các phần tử và các phần tử con. Phần tử mẹ chứa các phần tử con và tạo nên cây phân cấp. Điều này giúp xác định quan hệ giữa các phần tử trên trang web.

Đối tượng: HTML cho phép nhúng các đối tượng phong phú như hình ảnh, video, âm thanh, biểu đồ và nhiều nội dung đa phương tiện khác vào trang web. Điều này giúp tạo ra trải nghiệm người dùng đa dạng và phong phú.

Liên kết: Với thẻ <a>, HTML cho phép tạo liên kết giữa các trang web hoặc các phần tử khác nhau trong cùng một trang. Điều này mở ra khả năng điều hướng linh hoạt trên trang web.

Biểu đồ và bảng: HTML hỗ trợ việc tạo ra các bảng dữ liệu cũng như các biểu đồ sơ đồ. Điều này rất hữu ích cho việc hiển thị và trình bày thông tin.

Ẩn/hiện nội dung: HTML cung cấp các phần tử như <div> và <span> cho phép tạo ra các khối nội dung có thể được ẩn hoặc hiển thị theo điều kiện.

Hỗ trợ đa phương tiện: HTML5, phiên bản mới nhất của HTML, mang lại nhiều tính năng đa phương tiện mới như hỗ trợ video và âm thanh nâng cao, đồ họa vector, và khả năng làm việc offline.

Hệ thống form: HTML cung cấp các phần tử form cho phép thu thập thông tin từ người dùng. Điều này rất quan trọng cho việc tạo các trang web có khả năng giao tiếp với người dùng.

HTML là ngôn ngữ căn bản nhưng cực kỳ mạnh mẽ và quan trọng trong phát triển web. Nó là nền tảng cho việc xây dựng nội dung và tương tác trực tuyến trên internet.

### CSS

CSS, hay Cascading Style Sheets, là một ngôn ngữ định dạng và trang trí dùng để kiểm soát cách nội dung HTML được hiển thị trên trình duyệt web. CSS giúp tạo ra giao diện thú vị, thẩm mỹ và tạo trải nghiệm người dùng tốt hơn. Dưới đây là một giới thiệu chi tiết về CSS:

Lịch sử và nguồn gốc: CSS được phát triển lần đầu vào cuối những năm 1990 để giải quyết vấn đề định dạng và trang trí nội dung trên các trang web. Nó là một phần quan trọng của mô hình phát triển web ba lớp (HTML, CSS và JavaScript) và đã có nhiều phiên bản tiến bộ qua các năm.

Kiến thức cơ bản: CSS sử dụng các quy tắc đơn giản gồm các thuộc tính và giá trị để định dạng và trang trí nội dung HTML. Ví dụ, thuộc tính color có thể được sử dụng để định màu chữ trong văn bản, và thuộc tính background-color để định màu nền của một phần tử.

Lựa chọn và ưu tiên: Một tính năng quan trọng của CSS là khả năng xác định các phần tử cụ thể cần định dạng. CSS sử dụng các bộ chọn (selectors) để xác định phạm vi ứng dụng của các quy tắc. Điều này cho phép bạn lựa chọn cụ thể một phần tử hoặc một nhóm phần tử để áp dụng các quy tắc CSS.

Kế thừa và ưu tiên: CSS sử dụng nguyên tắc kế thừa, trong đó các thuộc tính của các phần tử cha có thể được kế thừa bởi các phần tử con. Tuy nhiên, nếu có sự xung đột giữa các quy tắc CSS, ưu tiên sẽ được áp dụng dựa trên mức độ cụ thể của các quy tắc và thứ tự xuất hiện trong mã nguồn.

Độ ưu tiên của CSS: Để quản lý ưu tiên giữa các quy tắc CSS, có một số yếu tố quan trọng cần xem xét:

Sự ưu tiên cụ thể: Quy tắc CSS được áp dụng theo mức độ cụ thể, ví dụ như ID selectors ưu tiên hơn class selectors.

Sự ưu tiên của Inline Styles: CSS được đặt trực tiếp trong mã HTML (inline) sẽ có ưu tiên cao hơn so với quy tắc CSS ngoài trang (external).

Kế thừa: Đối với các thuộc tính không được thiết lập trong các phần tử con, giá trị sẽ kế thừa từ phần tử cha.

Thứ tự xuất hiện: Trong trường hợp xung đột giữa các quy tắc cùng cấp, quy tắc xuất hiện sau trong mã sẽ có ưu tiên hơn.

* Các đặc điểm quan trọng:

Box Model: CSS quản lý mô hình hộp cho mỗi phần tử, bao gồm độ rộng (width), chiều cao (height), biên (margin), đệm (padding) và khung (border). Điều này làm cho việc xây dựng giao diện dễ dàng và mạnh mẽ.

Floating và Positioning: CSS cho phép bạn kiểm soát vị trí của các phần tử trên trang bằng cách sử dụng thuộc tính float và position.

Responsive Web Design: CSS chơi một vai trò quan trọng trong việc xây dựng giao diện đáp ứng (responsive) cho các trang web. Bằng cách sử dụng các phương pháp như truyền thông (media queries) và thiết lập độ rộng linh hoạt (flexible widths), bạn có thể tạo ra các trang web linh hoạt có khả năng thích nghi với các thiết bị và kích thước màn hình khác nhau.

Quản lý hình ảnh và đa phương tiện: CSS cho phép kiểm soát kích thước, định dạng và trình bày của hình ảnh, video và âm thanh trên trang web. Điều này giúp cải thiện trải nghiệm người dùng và làm cho trang web trở nên hấp dẫn hơn.

Thư viện và Framework CSS: Có nhiều thư viện và framework CSS như Bootstrap, Foundation và Materialize giúp giảm thiểu công việc thiết kế và tạo giao diện web từ đầu. Chúng cung cấp các bộ mã và kiểu mẫu sẵn có để giúp bạn nhanh chóng phát triển trang web chuyên nghiệp.

Sự phát triển liên tục: CSS không ngừng phát triển. CSS3, phiên bản mới nhất, đưa vào nhiều tính năng mới như hiệu ứng, đổ bóng, và chuyển động. Các tiêu chuẩn CSS liên tục được cập nhật để đáp ứng các xu hướng phát triển web mới.

### JavaScript

JavaScript là một ngôn ngữ lập trình phía máy khách (client-side) mạnh mẽ và đa năng, chủ yếu được sử dụng để phát triển ứng dụng web tương tác. Nó đã trở thành một trong những công cụ quan trọng trong lĩnh vực phát triển web và có sự ảnh hưởng lớn đến trải nghiệm người dùng trực tuyến. Dưới đây, chúng ta sẽ khám phá JavaScript thông qua một giới thiệu chi tiết:

Lịch sử và nguyên gốc: JavaScript ban đầu được phát triển bởi Netscape Communications Corporation vào cuối những năm 1990. Nó xuất hiện với mục tiêu tạo ra các ứng dụng web động và tương tác. Ngôn ngữ này sau đó đã trải qua nhiều phiên bản và tiêu chuẩn mới, với phiên bản hiện tại là ECMAScript 2021.

Ngôn ngữ đa mục đích: JavaScript không chỉ được sử dụng cho phát triển ứng dụng web, mà còn có thể thực hiện nhiều nhiệm vụ khác. Nó có thể chạy trên trình duyệt web, máy chủ thông qua Node.js, điều khiển thiết bị IoT (Internet of Things), phát triển ứng dụng di động, và thậm chí là lập trình trí tuệ nhân tạo.

Kiến thức cơ bản: JavaScript sử dụng cú pháp giống với nhiều ngôn ngữ lập trình khác, điều này làm cho việc học JavaScript dễ dàng đối với những người đã biết lập trình. Cú pháp của nó gồm các biến, hàm, điều kiện, vòng lặp và nhiều cấu trúc khác.

Cấu trúc dữ liệu: JavaScript hỗ trợ cấu trúc dữ liệu đa dạng như mảng (arrays), đối tượng (objects), chuỗi (strings), số (numbers), và nhiều loại dữ liệu khác. Điều này cho phép bạn lưu trữ và xử lý dữ liệu một cách hiệu quả.

Hàm là công cụ quan trọng: Hàm là một phần quan trọng của JavaScript. Chúng cho phép bạn tạo mã có thể được gọi và thực thi nhiều lần. JavaScript cung cấp khả năng truyền tham số vào hàm và trả về giá trị từ hàm.

Sự bất đồng bộ (Asynchronous): JavaScript thường được sử dụng để xử lý sự kiện người dùng và giao tiếp với máy chủ. Điều này đòi hỏi sự hỗ trợ cho việc xử lý bất đồng bộ, và JavaScript có các cơ chế như Callbacks, Promises, và async/await để quản lý quá trình này.

Thư viện và Framework: Có nhiều thư viện và framework JavaScript mạnh mẽ như jQuery, React, Angular, và Vue.js, giúp tạo ra các ứng dụng phức tạp và hiệu quả. Các thư viện này cung cấp các thành phần và tính năng sẵn có để giảm thiểu công việc phát triển.

DOM và Manipulation: JavaScript có thể truy cập và thay đổi các phần tử HTML và CSS trên trang web thông qua Document Object Model (DOM). Điều này cho phép bạn tạo ra các ứng dụng web tương tác và động.

Trò chơi và trí tuệ nhân tạo: JavaScript có thể được sử dụng để phát triển trò chơi trực tuyến và ứng dụng trí tuệ nhân tạo thông qua các thư viện như Phaser và TensorFlow.js.

Phát triển liên tục: JavaScript không ngừng phát triển và tiếp tục mở rộng khả năng lập trình web. Tiêu chuẩn mới và cải tiến liên tục được đưa vào để đáp ứng các yêu cầu ngày càng phức tạp của phát triển web.

JavaScript đã và đang thay đổi cách chúng ta tương tác với trình duyệt web và xây dựng các ứng dụng web phức tạp. Nó đã trở thành một phần quan trọng trong thế giới công nghệ thông tin và tiếp tục phát triển để đáp ứng các thách thức mới trong lĩnh vực phát triển web.

### Flask

Flask là một framework phát triển ứng dụng web Python siêu nhẹ, linh hoạt và dễ học. Nó được tạo ra bởi Armin Ronacher và được công bố dưới giấy phép BSD, cho phép bạn xây dựng ứng dụng web nhanh chóng và dễ dàng. Flask nổi tiếng với triết lý "micro" - nó cung cấp các công cụ cơ bản để xây dựng ứng dụng web, nhưng để lại cho bạn quyền kiểm soát hoàn toàn về cách bạn tổ chức mã và chọn các thành phần khác để sử dụng. Dưới đây, chúng ta sẽ khám phá Flask chi tiết để hiểu tại sao nó lại hấp dẫn đối với các nhà phát triển và dự án web.

Dễ học và sử dụng: Flask được thiết kế với mục tiêu đơn giản và dễ hiểu. Điều này làm cho việc học và sử dụng nó dễ dàng, đặc biệt là đối với các người mới bắt đầu lập trình web với Python.

Cú pháp của Flask là một trong những điểm mạnh quan trọng. Với ít dòng mã, bạn có thể tạo một ứng dụng web hoạt động.

Linh hoạt và tùy biến: Flask không đưa ra quyết định áp đặt về cách bạn tổ chức dự án của mình. Bạn có toàn quyền tự quyết định về cấu trúc thư mục, database, template engine, và nhiều thành phần khác.

Bạn có thể thêm các thư viện hoặc extension mà bạn cần để bổ sung các tính năng, chẳng hạn như xác thực người dùng, gửi email, hoặc làm việc với cơ sở dữ liệu.

Tiêu chuẩn WSGI: Flask tuân thủ tiêu chuẩn WSGI (Web Server Gateway Interface), cho phép bạn triển khai ứng dụng Flask trên nhiều dịch vụ lưu trữ web và máy chủ.

Điều này giúp bạn không bị ràng buộc vào một môi trường cụ thể và có thể di chuyển ứng dụng dễ dàng giữa các môi trường khác nhau.

Template Engine: Flask tích hợp sẵn một template engine, cho phép bạn tạo và hiển thị trang web dễ dàng thông qua templates. Jinja2, template engine mặc định của Flask, hỗ trợ các chức năng mạnh mẽ như lặp, điều kiện, và kế thừa, giúp bạn tạo ra giao diện web động.

Phát triển API: Flask không chỉ được sử dụng cho việc phát triển trang web, mà còn cho việc phát triển các ứng dụng API. Flask-RESTful là một extension phổ biến để phát triển các dịch vụ web RESTful một cách nhanh chóng.

Cộng đồng mạnh mẽ:

Flask có một cộng đồng lớn và nhiệt tình. Bạn có thể tìm kiếm giúp đỡ qua các diễn đàn, Stack Overflow, hoặc tham gia các nhóm lập trình Flask để học hỏi và chia sẻ kiến thức.

Mô-đun độc lập: Flask được phát triển dưới dạng các mô-đun độc lập. Điều này có nghĩa là bạn có thể sử dụng các mô-đun mà bạn cần và bỏ qua những mô-đun không cần thiết, giúp ứng dụng của bạn nhẹ nhàng và hiệu quả.

Hỗ trợ đa ngôn ngữ: Flask hỗ trợ nhiều ngôn ngữ và cung cấp các công cụ để xây dựng ứng dụng đa ngôn ngữ dễ dàng.

Các Extension phong phú: Flask có một loạt các extension mạnh mẽ để giúp bạn xử lý nhiều tác vụ khác nhau, từ xác thực người dùng đến làm việc với cơ sở dữ liệu, và nhiều chức năng khác.

### Flask-SocketIO

Flask-SocketIO là một extension mạnh mẽ cho Flask, một framework phát triển ứng dụng web Python, giúp bạn tích hợp hỗ trợ WebSocket vào ứng dụng web của mình. WebSocket là một giao thức cho phép truyền dữ liệu hai chiều realtime giữa máy chủ và trình duyệt web, làm cho việc phát triển các ứng dụng web đa phương tiện, chat realtime, trò chơi trực tuyến và nhiều ứng dụng khác trở nên dễ dàng hơn.

Dễ dàng tích hợp: Flask-SocketIO tích hợp dễ dàng vào ứng dụng Flask hiện có của bạn. Nó cho phép bạn kết hợp ứng dụng web cổ điển xây dựng bằng Flask với các tính năng realtime sử dụng WebSocket.

WebSocket và Long Polling: Flask-SocketIO cung cấp hỗ trợ cho cả WebSocket và Long Polling, cho phép bạn lựa chọn giao thức phù hợp với ứng dụng của bạn. Điều này đảm bảo tính tương thích và hiệu suất tốt trên nhiều môi trường.

Tiết kế đơn giản: Flask-SocketIO tuân theo triết lý "micro" của Flask, với cú pháp đơn giản và linh hoạt. Bạn có toàn quyền kiểm soát cách bạn tổ chức mã của mình và lựa chọn các thành phần khác để sử dụng.

Xử lý sự kiện Realtime: Bạn có thể xác định các sự kiện realtime trong ứng dụng của mình và thực hiện xử lý khi các sự kiện này xảy ra. Điều này làm cho việc xây dựng các tính năng realtime như trò chơi đa người, chat, thông báo tức thì trở nên dễ dàng.

Phát tin nhắn: Flask-SocketIO cho phép bạn gửi tin nhắn realtime từ máy chủ đến trình duyệt và ngược lại. Điều này rất hữu ích cho việc cập nhật dữ liệu tức thời và thông báo cho người dùng.

Hỗ trợ phân tán: Bạn có thể triển khai ứng dụng Flask-SocketIO trên nhiều máy chủ hoặc môi trường phân tán, giúp ứng dụng của bạn mở rộng dễ dàng. Flask-SocketIO đã giúp nhiều dự án xây dựng các tính năng realtime một cách dễ dàng và hiệu quả. Nó đặc biệt phù hợp cho các ứng dụng web đòi hỏi tính tương tác realtime, và là một sự bổ sung mạnh mẽ cho Flask, mở ra nhiều cơ hội trong việc phát triển ứng dụng web sử dụng WebSocket.

### Socket.IO

Socket.IO là một thư viện JavaScript giúp xây dựng ứng dụng web realtime sử dụng giao thức WebSocket hoặc các kỹ thuật như Long Polling. Thư viện này đặc biệt hữu ích trong việc phát triển các ứng dụng web đòi hỏi tính tương tác trực tiếp và cập nhật tức thì như chat, trò chơi trực tuyến, ứng dụng thông báo thời gian thực và nhiều ứng dụng khác. Dưới đây là một số điểm quan trọng về Socket.IO:

Hỗ Trợ Nhiều Giao Thức: Socket.IO hỗ trợ cả WebSocket và Long Polling, đảm bảo tính tương thích với nhiều trình duyệt và môi trường máy chủ. Điều này cho phép bạn tận dụng các lợi ích của WebSocket trong môi trường hỗ trợ, và tự động rơi vào Long Polling nếu cần.

Tạo Kết Nối Thời Gian Thực: Socket.IO cho phép bạn thiết lập kết nối thời gian thực giữa máy chủ và trình duyệt. Điều này có nghĩa là bạn có thể gửi và nhận dữ liệu một cách tức thì, mà không cần phải làm mới trang web.

Xử Lý Sự Kiện Realtime: Bạn có thể định nghĩa và xử lý các sự kiện realtime trong ứng dụng của bạn. Ví dụ, bạn có thể xác định sự kiện "chat message" để gửi và nhận tin nhắn trong ứng dụng chat.

Phòng và Tùy Chọn Đối Tượng: Socket.IO cho phép bạn tạo các "phòng" cho các kết nối và tùy chỉnh các đối tượng gửi dữ liệu realtime. Điều này giúp bạn quản lý dễ dàng các nhóm người dùng và loại thông điệp.

Hỗ Trợ Đa Nền Tảng: Socket.IO không chỉ hỗ trợ JavaScript cho trình duyệt web, mà còn hỗ trợ mã nguồn ngôn ngữ server-side, giúp bạn xây dựng ứng dụng realtime hoàn chỉnh.

Phát triển Mạnh Mẽ: Socket.IO có một cộng đồng phát triển mạnh mẽ và được duy trì liên tục. Nó cung cấp nhiều tài liệu và ví dụ để giúp bạn bắt đầu nhanh chóng.

Socket.IO đã trở thành một phần quan trọng trong cộng đồng phát triển web và rất phù hợp cho việc xây dựng các ứng dụng web realtime. Đối với các trường hợp sử dụng, như chat 1-1, ứng dụng trò chơi multiplayer hoặc cập nhật dữ liệu trực tiếp trong thời gian thực, Socket.IO là một công cụ hữu ích và mạnh mẽ để xem xét.

# PHÂN TÍCH BÀI TOÁN

## Mô tả các chức năng của ứng dụng

* Ứng dụng "Wonder Words" là một trò chơi mạng thú vị dựa trên cuộc thi trò chơi "Wheel of Fortune." Dưới đây là mô tả chi tiết về các chức năng của ứng dụng:
* Đăng nhập và nhập tên người chơi:
* Người chơi truy cập trang web và có thể nhập tên của họ trong hộp văn bản.
* Sau khi nhập tên, họ có thể nhấn nút "Submit" để đăng ký và tham gia trò chơi.
* Giao diện trò chơi:
* Sau khi đăng ký, giao diện trò chơi sẽ hiển thị. Giao diện này bao gồm các phần chính để hiển thị thông tin về từ vựng và lượt chơi.
* Hiển thị từ vựng:
* Một từ vựng được chọn ngẫu nhiên từ danh sách và hiển thị trên giao diện trò chơi.
* Các ký tự chưa khớp trong từ vựng được hiển thị bằng dấu gạch ngang ('-'), và các khoảng trắng được hiển thị như bình thường.
* Mô tả của từ vựng cũng được hiển thị để giúp người chơi đoán.
* Đăng ký sẵn sàng:
* Người chơi có thể nhấn nút "Ready" để đăng ký sẵn sàng.
* Trò chơi bắt đầu chỉ khi tất cả người chơi đã sẵn sàng.
* Lượt chơi đua thời gian:
* Trò chơi được chia thành các lượt chơi với thời gian hạn chế. Mỗi lượt chơi kéo dài 10 giây.
* Thời gian còn lại trong lượt chơi được hiển thị trên giao diện.
* Đoán ký tự:
* Người chơi có thể đoán một ký tự trong từ vựng trong mỗi lượt chơi.
* Nếu đoán đúng, ký tự đúng sẽ được mở ra trong từ vựng, và điểm thưởng sẽ được cộng vào điểm của người chơi.
* Xếp hạng người chơi:
* Điểm số của người chơi được tính và hiển thị trên bảng xếp hạng.
* Người chơi có điểm số cao nhất sẽ thắng cuộc.
* Quản lý người chơi:
* Danh sách người chơi được quản lý và hiển thị trên giao diện.
* Người chơi có thể chọn sẵn sàng để tham gia vào trò chơi.
* Gửi tin nhắn:
* Hệ thống có khả năng gửi tin nhắn cho người chơi, thông báo về việc đoán đúng hoặc đoán sai.
* Kết thúc trò chơi:
* Trò chơi kết thúc khi từ vựng được đoán đúng hoặc hết thời gian cho tất cả lượt chơi.
* Người chơi thắng cuộc sẽ được công bố, và điểm số cuối cùng sẽ được hiển thị.

## Hướng xây dựng

Trò chơi "Wonder Words" được xây dựng bằng sử dụng Flask, một web framework Python, và Socket.IO để tạo một trải nghiệm trò chơi đa người chơi. Dưới đây là phân tích cách xây dựng trò chơi này:

* Sử dụng Flask và Socket.IO:

Flask là một framework web Python, cung cấp các công cụ cho việc xây dựng ứng dụng web. Flask được sử dụng để tạo ứng dụng web cơ bản và cung cấp route cho trang chủ (route gốc).

Socket.IO là một thư viện cho phép truyền tải dữ liệu realtime giữa máy chủ và trình duyệt của người dùng. Nó được sử dụng để tạo tương tác realtime trong trò chơi.

* Cấu hình CORS:

Để cho phép truy cập từ các nguồn gốc khác nhau, Cross-Origin Resource Sharing (CORS) đã được cấu hình bằng cách sử dụng thư viện Flask-CORS. Điều này cho phép các trình duyệt khác nhau kết nối vào máy chủ.

* Chuẩn bị Dữ Liệu:

Dữ liệu trò chơi được chuẩn bị trước trong tệp words.json, chứa danh sách từ vựng và mô tả tương ứng. Thư viện json của Python được sử dụng để đọc dữ liệu từ tệp.

* Sự Kiện Socket.IO:

Socket.IO được sử dụng để theo dõi các sự kiện trên máy chủ và gửi dữ liệu realtime đến trình duyệt của người chơi. Các sự kiện Socket.IO bao gồm:

connect: Sự kiện xảy ra khi một người chơi kết nối vào trò chơi. Họ được thêm vào danh sách người chơi.

disconnect: Sự kiện xảy ra khi người chơi ngắt kết nối. Họ sẽ bị loại khỏi danh sách người chơi.

player\_name: Sự kiện xảy ra khi người chơi đặt tên cho họ. Tên sẽ được lưu lại.

ready: Sự kiện xảy ra khi người chơi bấm nút "Sẵn sàng" để tham gia trò chơi.

guess: Sự kiện xảy ra khi người chơi đoán một ký tự.

* Quản lý Trạng Thái Trò Chơi:

Trò chơi sử dụng biến toàn cục như game\_started, current\_word, và current\_word\_revealed để quản lý trạng thái trò chơi và dữ liệu từ vựng hiện tại.

* Bắt Đầu Trò Chơi:

Trò chơi bắt đầu khi tất cả người chơi đã sẵn sàng. Một từ vựng mới được chọn và mô tả tương ứng được hiển thị cho người chơi. Cuộc đua đoán từ vựng bắt đầu.

* Kết Thúc Trò Chơi:

Trò chơi kết thúc khi tất cả các ký tự trong từ vựng đã được tiết lộ hoặc khi không còn người chơi tham gia.

Điểm số của từng người chơi được tính toán và thông báo. Người chơi có điểm số cao nhất sẽ được xác định là người chiến thắng.

* Truyền Tải Dữ Liệu Realtime:

Dữ liệu như thông báo, từ vựng tiết lộ, thông tin về người chơi và điểm số được truyền tải realtime từ máy chủ đến trình duyệt của người chơi bằng Socket.IO.

* Chạy Ứng Dụng:

Ứng dụng chạy với giao diện dòng lệnh bằng socketio.run(app, debug=True, allow\_unsafe\_werkzeug=True).

# THỰC THI CHƯƠNG TRÌNH

## Các giao diện chương trình

### Giao diện bắt đầu trò chơi

A screenshot of a computer

Description automatically generated

Bắt đầu trò chơi, người chơi sẽ thấy giao diện yêu cầu nhập tên người chơi. Người chơi sẽ tiến hành nhập tên và nhấn nút Submit để vào trò chơi.

A screenshot of a computer

Description automatically generatedTrò chơi có thể mở các tab khác nhau cho mỗi người chơi khác nhau

A screenshot of a computer

Description automatically generatedKhi đã nhập tên và bấm nút submit thì sẽ vào giao diện chính của trò chơi. Ở giao diện sẽ có nút Ready để người chơi chuyển sang trạng thái sẵn sàng cho trò chơi. Khi tất cả các người chơi ở trong trò chơi đều ở trạng thái sẵn sàng thì trò chơi sẽ bắt đầu.

A screenshot of a computer

Description automatically generated

Khi đã ấn nút Ready thì người chơi sẽ đợi cho tất cả cả người chơi ở trong trò chơi đều sẵn sàng để trò chơi được bắt đầu.

A screenshot of a computer

Description automatically generated

Trò chơi bắt đầu sẽ hiển thị Hint là số lượng từ và số lượng chữ cái trong mỗi từ đó. Thời gian đếm ngược của trò chơi là 10s.

A screenshot of a computer

Description automatically generated

Với mỗi chữ cái đoán đúng thì người chơi sẽ được 100 điểm, và điểm số sẽ được cập nhật lên Dashboard.

A screenshot of a computer

Description automatically generated

Khi đoán sai chương trình sẽ hiển thị thông báo Wrong guess! Người chơi phải chờ sau 10s đến lượt tiếp theo mới được đoán tiếp.

A screenshot of a computer

Description automatically generated

Khi tất cả các từ đều được đoán thì chương trình sẽ tính kết quả và thông báo cho người chơi.

A screenshot of a computer

Description automatically generated

Người chơi có điểm số cao nhất sẽ là người thắng cuộc.

Ấn Ready để bắt đầu trò chơi lại.

## Ưu và nhược điểm của trò chơi

### Ưu điểm

Giải trí và học hỏi: Trò chơi giúp người chơi giải trí thông qua việc đoán từ và tích hợp yếu tố học hỏi. Người chơi cần suy nghĩ và tìm hiểu từ mới thông qua mô tả.

Tương tác xã hội: Trò chơi được thiết kế cho nhiều người chơi, cho phép họ tương tác với nhau qua Internet. Điều này tạo ra một trải nghiệm xã hội, đặc biệt khi các người chơi có thể chat và cùng nhau tham gia trò chơi.

Thiết kế giao diện hấp dẫn: Trò chơi có giao diện đẹp và thu hút với màu nền gradient và sử dụng Bootstrap để tạo ra các phần giao diện thân thiện với người dùng.

Quản lý kết nối thông qua Socket.IO: Trò chơi sử dụng Socket.IO để quản lý kết nối giữa người chơi và máy chủ. Điều này giúp đồng bộ hóa trạng thái trò chơi và tương tác thời gian thực.

Phát triển tích hợp: Trò chơi này có tiềm năng để được phát triển và cải tiến thêm bằng cách thêm tính năng mới, từ vựng mới và quy tắc mới.

### Nhược điểm:

Khó khăn trong tạo ra cơ sở dữ liệu từ vựng: Trò chơi yêu cầu một cơ sở dữ liệu từ vựng để hoạt động. Tạo ra cơ sở dữ liệu từ vựng hoặc duy trì nó có thể mất nhiều công sức và thời gian.

Khả năng griefing: Trong mô hình đoán từ, người chơi có thể đoán sai ý định để gây khó khăn cho người chơi khác hoặc làm trò chơi mất tính công bằng.

Yếu tố thời gian giới hạn: Trò chơi này có thời gian giới hạn cho mỗi lượt đoán, điều này có thể làm cho người chơi cảm thấy áp lực và gây thất vọng nếu họ không đoán được từ kịp thời.

Khả năng trở nên nhàm chán: Nếu trò chơi không có đủ độ khó hoặc các tính năng bổ sung, nó có thể trở nên nhàm chán sau một thời gian chơi.

## Kết luận và hướng phát triển

Trò chơi "Wonder Words" là một trò chơi giải đố thú vị mang lại sự kết hợp giữa giải trí và học hỏi. Nó tạo điều kiện cho người chơi thử thách bản thân trong việc đoán từ và mở rộng vốn từ vựng của mình thông qua mô tả từ. Giao diện đẹp mắt và tích hợp kết nối thời gian thực thông qua Socket.IO tạo ra trải nghiệm chơi game tốt.

Mở rộng Cơ sở dữ liệu Từ vựng: Tăng cường cơ sở dữ liệu từ vựng để mang lại nhiều từ vựng hơn và mang tính đa dạng hơn cho trò chơi.

Cập nhật Giao diện Người dùng: Có thể cân nhắc việc cập nhật giao diện người dùng, thêm hình ảnh hoặc biểu tượng để làm trò chơi trở nên thú vị hơn.

Thúc đẩy Tính Cạnh tranh: Thêm tính năng bảng xếp hạng để khuyến khích tính cạnh tranh giữa các người chơi.

Thêm Chế độ Chơi Đa dạng: Khám phá việc thêm các chế độ chơi khác nhau, ví dụ: chế độ đối kháng, chế độ đội hình, hoặc chế độ thư giãn.

Bổ sung Tính năng Kiểm soát Trò chơi: Thêm tính năng quản lý người chơi và cài đặt của trò chơi, ví dụ: tắt tiếng, quản lý phòng chơi, hoặc cài đặt thời gian chơi.

Tích hợp Hệ thống Điểm: Tạo một hệ thống điểm thưởng để khuyến khích sự tham gia và cạnh tranh giữa các người chơi.

Mở rộng Tính năng Mạng Xã hội: Kết nối trò chơi với các mạng xã hội để người chơi có thể chia sẻ thành tích của mình hoặc mời bạn bè tham gia.

Kiểm tra và Sửa lỗi: Luôn kiểm tra và sửa lỗi để đảm bảo trò chơi hoạt động mượt mà và không gặp vấn đề kỹ thuật.

TÀI LIỆU THAM KHẢO

[1] Guillermo Rauch, Socket.IO - Realtime application framework (Node.JS server), 2010

[2] World Wide Web Consortium (W3C), HTML and CSS, 2014

[3] HTML and CSS, Flask-SocketIO Documentation, 2018

[4] Guido van Rossum, Python Programming Language Documentation, 2009

PHỤ LỤC

Mã nguồn chương trình:

File main.py:

from flask import Flask, render\_template, request

from flask\_socketio import SocketIO, emit

import json

import random

import time

from flask\_cors import CORS

app = Flask(\_\_name\_\_)

socketio = SocketIO(app, cors\_allowed\_origins='\*')

CORS(app)

game\_started = False

with open('templates/words.json') as f:

    word\_list = json.load(f)

def get\_random\_word():

    word = random.choice(word\_list)

    return word['word\_entry'], word['description']

players = []

current\_word = ""

current\_word\_revealed = []

turn\_time = 10

@app.route('/')

def index():

    return render\_template('index.html', players=players)

@socketio.on('connect')

def on\_connect():

    player\_id = request.sid

    players.append({'player\_id': player\_id, 'name': '', 'score': 0, 'is\_ready': False})

@socketio.on('disconnect')

def on\_disconnect():

    player\_id = request.sid

    players[:] = [player for player in players if player['player\_id'] != player\_id]

    emit('player\_list', {'players': [p for p in players if p['name'] != ""]}, broadcast=True)

    if len(players) == 0 or len([p for p in players if p['name'] != ""]) == 0:

        end\_game()

@socketio.on('player\_name')

def set\_player\_name(data):

    player\_id = request.sid

    player = next((p for p in players if p['player\_id'] == player\_id), None)

    if player:

        player['name'] = data['name']

        emit('message', {'message': f'Hi {data["name"]}! Please click "Ready" to start the game.'})

        emit('player\_list', {'players': [p for p in players if p['name'] != ""]}, broadcast=True)

@socketio.on('ready')

def on\_ready():

    global game\_started

    player\_id = request.sid

    player = next((p for p in players if p['player\_id'] == player\_id and p['name'] != ""), None)

    if player:

        player['is\_ready'] = True

        emit('player\_list', {'players': [p for p in players if p['name'] != ""]}, broadcast=True)

    all\_ready = all(player['is\_ready'] for player in players if player['name'] != "")

    if all\_ready and not game\_started:

        game\_started = True

        start\_game()

@socketio.on('guess')

def on\_guess(data):

    player\_id = request.sid

    player = next((p for p in players if p['player\_id'] == player\_id), None)

    if player and player['is\_ready']:

        if player.get('guess') is None:

            guess = data.get('guess')

            if guess and len(guess) == 1:

                player['guess'] = guess.lower()

def start\_game():

    global current\_word, current\_word\_revealed, game\_started

    current\_word, description = get\_random\_word()

    current\_word\_revealed = ['-' if c != ' ' else ' ' for c in current\_word]

    emit('message', {'message': 'Game starts now!'}, broadcast=True)

    emit('word', {'revealed\_word': current\_word\_revealed, 'description': description}, broadcast=True)

    emit('hint', {'hint': ''.join(current\_word\_revealed)}, broadcast=True)

    for player in players:

        player['score'] = 0

    emit('player\_list', {'players': [p for p in players if p['name'] != ""]}, broadcast=True)

    while '-' in current\_word\_revealed:

        for sec in range(turn\_time, 0, -1):

            if sec == 10:

                emit('again', {'message': 'again'}, broadcast=True)

            if len(players) == 0 or len([p for p in players if p['name'] != ""]) == 0:

                break

            emit('countdown', {'time': sec}, broadcast=True)

            time.sleep(1)

            for player in players:

                if player.get('guess'):

                    guess = player['guess'].lower()

                    if guess in current\_word\_revealed or guess.upper() in current\_word\_revealed:

                        emit('message', {'message': 'Wrong guess!'},

                             room=player['player\_id'])

                        player['guess'] = None

                    elif guess in current\_word.lower():

                        for i, char in enumerate(current\_word):

                            if char.lower() == guess:

                                current\_word\_revealed[i] = char if char != ' ' else ' '

                        emit('reveal', {'revealed\_word': current\_word\_revealed}, broadcast=True)

                        score = current\_word.lower().count(guess) \* 100

                        player['score'] += score

                        player['guess'] = None

                        emit('player\_list', {'players': [p for p in players if p['name'] != ""]}, broadcast=True)

                        emit('message', {'message': 'Right guess!'},

                             room=player['player\_id'])

                    else:

                        emit('player\_list', {'players': [p for p in players if p['name'] != ""]}, broadcast=True)

                        emit('message', {'message': 'Wrong guess!'},

                             room=player['player\_id'])

                        player['guess'] = None

        emit('reveal', {'revealed\_word': current\_word\_revealed}, broadcast=True)

    end\_game()

def end\_game():

    global game\_started

    game\_started = False

    players\_with\_name = [p for p in players if p['name'] != ""]

    if len(players\_with\_name) > 0:

        total\_scores = {player['player\_id']: player['score'] for player in players\_with\_name}

        winner\_id = max(total\_scores, key=total\_scores.get)

        max\_score = total\_scores[winner\_id]

        emit('player\_list', {'players': players\_with\_name}, broadcast=True)

        for player in players\_with\_name:

            emit('end\_game', {'message': 'end\_game'}, broadcast=True)

            player['is\_ready'] = False

            score = player['score']

            if player['score'] == max\_score:

                emit('message', {'message': f'You win with total score of {max\_score}!'}, room=player['player\_id'])

            else:

                emit('message', {'message': f'You lose with total score of {score}!'}, room=player['player\_id'])

    emit('player\_list', {'players': [p for p in players if p['name'] != ""]}, broadcast=True)

    global current\_word

    current\_word = ""

    global current\_word\_revealed

    current\_word\_revealed = []

if \_\_name\_\_ == '\_\_main\_\_':

    socketio.run(app, debug=True, allow\_unsafe\_werkzeug=True)

File index.html:

<html>

<head>

  <title>Wonder Words Game</title>

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

  <style>

        body {

      background-image: radial-gradient(circle at 30% 86%, rgba(255,255,255,0.03) 0%, rgba(255,255,255,0.03) 8%,transparent 8%, transparent 92%),radial-gradient(circle at 55% 100%, rgba(255,255,255,0.03) 0%, rgba(255,255,255,0.03) 8%,transparent 8%, transparent 92%),radial-gradient(circle at 40% 75%, rgba(255,255,255,0.03) 0%, rgba(255,255,255,0.03) 6%,transparent 6%, transparent 94%),radial-gradient(circle at 7% 99%, rgba(255,255,255,0.03) 0%, rgba(255,255,255,0.03) 6%,transparent 6%, transparent 94%),radial-gradient(circle at 69% 76%, rgba(255,255,255,0.03) 0%, rgba(255,255,255,0.03) 6%,transparent 6%, transparent 94%),radial-gradient(circle at 2% 35%, rgba(255,255,255,0.03) 0%, rgba(255,255,255,0.03) 6%,transparent 6%, transparent 94%),radial-gradient(circle at 14% 48%, rgba(255,255,255,0.03) 0%, rgba(255,255,255,0.03) 6%,transparent 6%, transparent 94%),radial-gradient(circle at 28% 87%, rgba(255,255,255,0.04) 0%, rgba(255,255,255,0.04) 4%,transparent 4%, transparent 96%),radial-gradient(circle at 65% 14%, rgba(255,255,255,0.04) 0%, rgba(255,255,255,0.04) 4%,transparent 4%, transparent 96%),radial-gradient(circle at 51% 36%, rgba(255,255,255,0.04) 0%, rgba(255,255,255,0.04) 4%,transparent 4%, transparent 96%),radial-gradient(circle at 6% 93%, rgba(255,255,255,0.04) 0%, rgba(255,255,255,0.04) 4%,transparent 4%, transparent 96%),linear-gradient(135deg, rgb(23, 233, 173),rgb(29, 24, 208));

      height: 100vh;

    }

    .container {

      position: relative;

      top: 40%;

      transform: translateY(-50%);

    }

    .card {

      background-color: rgba(0,0,0,0.3);

      padding: 20px;

      color: white;

    }

    .word {

      font-size: 24px;

      margin-bottom: 20px;

      font-family: monospace;

    }

    .description {

      margin-bottom: 20px;

    }

    .message {

      margin-top: 20px;

    }

    .countdown {

      font-weight: bold;

    }

    .scores {

      margin-top: 20px;

    }

    .name-form {

      margin-bottom: 10px;

      margin-left: 50%;

      transform: translateX(-50%);

    }

    .game-container {

      display: none;

    }

    .player-list {

      list-style-type: none;

      padding: 0;

      margin-top: 20px;

    }

    .player-list li {

      margin-bottom: 5px;

    }

    .player-list .player-name {

      font-weight: bold;

    }

    .player-list .player-status {

      font-style: italic;

    }

  </style>

</head>

<body>

  <div class="container shadow-lg p-4 mb-4">

    <div class="card">

      <form class="name-form shadow-lg p-4 my-4">

        <div class="text-center">

          <h2>Welcome to Wonder Words!</h2>

        </div>

        <div class="form-group">

          <label for="player-name" class="my-3">Enter Your Name:</label>

          <input type="text" class="form-control player-name-input my-3" id="player-name" placeholder="Enter your name" required>

        </div>

        <button type="submit" class="btn btn-primary name-submit-button my-3">Submit</button>

      </form>

      <div class="game-container">

        <div class="row">

          <div class="col-sm-6">

            <h2>Hint:</h2>

            <div class="word"></div>

            <h2>Description:</h2>

            <div class="description"></div>

            <div class="countdown"></div>

            <button class="btn btn-primary ready-button my-3">Ready</button>

            <input type="text" class="form-control guess-input my-3" placeholder="Enter a character" maxlength="1" required>

            <button class="btn btn-primary guess-button my-3" disabled>Guess</button>

            <div class="message"></div>

          </div>

          <div class="col-sm-6" style="background-color: rgba(0,0,0,0.4); border-radius: 10px;">

            <h4>Dashboard</h4>

            <ul class="player-list">

            </ul>

        </div>

        </div>

      </div>

  </div>

  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

  <script src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.2.0/socket.io.js"></script>

  <script>

    $(document).ready(() => {

      const socket = io.connect('http://localhost:5000');

      let wordInitialized = false;

      socket.on('connect', () => {

        console.log('Connected to server');

      });

      $('.name-form').submit((e) => {

      e.preventDefault();

      const playerName = $('.player-name-input').val().trim();

      if (playerName) {

        socket.emit('player\_name', { name: playerName });

        $('.name-form').hide();

        $('.game-container').show();

      }

    });

      socket.on('message', (data) => {

        const  message  = data.message;

        $('.message').text(message);

      });

      socket.on('reveal', (data) => {

        const revealedWord = data.revealed\_word.join('');

        $('.word').text(revealedWord);

        if (!wordInitialized) {

          wordInitialized = true;

          $('.word').text(revealedWord.replace(/-/g, '-').replace(/\s/g, '\u00A0'));

        }

      });

      socket.on('word', (data) => {

        $('.word').text('');

        $('.description').text(data.description);

      });

      socket.on('countdown', (data) => {

        $('.countdown').text(`Turn countdown: ${data.time} seconds`);

      });

      socket.on('again', (data) => {

        const message = data.message;

        console.log(message);

        if (message === 'again') {

          $('.guess-button').prop('disabled', false);

        }

      });

      socket.on('hint', (data) => {

        const hint = data.hint.replace(/-/g, '-').replace(/\s/g, '\u00A0');

        $('.word').text(hint);

      });

        socket.on('message', (data) => {

          const message = data.message;

          if (message === 'Wrong guess!') {

            $('.guess-button').prop('disabled', true);

          }

        });

      socket.on('player\_list', (data) => {

      const players = data.players;

      $('.player-list').empty();

      $('.player-list').append(`<li>

    <div class="row">

      <div class="col-sm-4"><span class="player-name">Name</span></div>

      <div class="col-sm-4"><span class="player-score">Score</span></div>

      <div class="col-sm-4"><span class="player-status">Status</span></div>

    </div></li>

`);

    players.forEach((player) => {

        const { player\_id, name, score, is\_ready } = player;

        const status = is\_ready ? 'Ready' : 'Not Ready';

        $('.player-list').append(`<li>

  <div class="row">

    <div class="col-sm-4 player-name">${name}</div>

    <div class="col-sm-4 player-score">${score}</div>

    <div class="col-sm-4 player-status">${status}</div>

  </div></li>

`);

    });

});

   $('.ready-button').click(() => {

        socket.emit('ready');

        $('.ready-button').prop('disabled', true);

        $('.guess-button').prop('disabled', false);

        $('.message').text('Please wait for all players to be ready!');

      });

      socket.on('end\_game', (data) => {

        const hint = data.message;

        $('.ready-button').prop('disabled', false);

        $('.guess-button').prop('disabled', true);

      });

      $('.guess-button').click(() => {

        const guess = $('.guess-input').val().trim();

        if (guess) {

          socket.emit('guess', { guess });

        }

        $('.guess-input').val('');

      });

    });

  </script>

</body>

</html>