**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

Trước hết, tôi xin gửi lời cảm ơn chân thành đến Thầy Nguyễn Bảo Ân, người đã trực tiếp hướng dẫn, giúp đỡ và giải đáp thắc mắc trong suốt quá trình thực hiện đề tài. Sự tận tình Thầy đã giúp tôi hoàn thành đồ án này một cách tốt nhất.

Tôi cũng xin bày tỏ lòng biết ơn đến các Thầy, Cô trong Bộ môn Công nghệ thông tin, đã cung cấp những kiến thức nền tảng cần thiết trong suốt thời gian học tập và nghiên cứu.

**MỤC LỤC**

[TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH 7](#_Toc186091497)

[**1.** **Vấn đề nghiên cứu** 7](#_Toc186091498)

[**2.** **Các hướng tiếp cận** 7](#_Toc186091499)

[**3.** **Cách giải quyết vấn đề** 7](#_Toc186091500)

[**4.** **Kết quả đạt được** 8](#_Toc186091501)

[MỞ ĐẦU 9](#_Toc186091502)

[**1.** **Lý do chọn đề tài** 9](#_Toc186091503)

[**2.** **Mục đích** 9](#_Toc186091504)

[**3.** **Đối tượng nghiên cứu** 9](#_Toc186091505)

[**4.** **Phạm vi nghiên cứu** 10](#_Toc186091506)

[CHƯƠNG 1: TỔNG QUAN 11](#_Toc186091507)

[CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT 14](#_Toc186091508)

[HTML 14](#_Toc186091509)

[Khái niệm 14](#_Toc186091510)

[Nguyên lý hoạt động của HTML 14](#_Toc186091511)

[Ứng dụng 14](#_Toc186091512)

[CSS 14](#_Toc186091513)

[Khái niệm 14](#_Toc186091514)

[Chức năng chính 14](#_Toc186091515)

[Cấu trúc của CSS 15](#_Toc186091516)

[Phân loại 15](#_Toc186091517)

[JavaScript 15](#_Toc186091518)

[Khái niệm 15](#_Toc186091519)

[Các đặc điểm chính 16](#_Toc186091520)

[Chức năng 16](#_Toc186091521)

[Mediapipe 16](#_Toc186091522)

[Khái niệm 16](#_Toc186091523)

[Các tính năng chính 17](#_Toc186091524)

[**2.4.1** **Cách cài đặt và sử dụng** 17](#_Toc186091525)

[CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU 19](#_Toc186091526)

[Các bước nghiên cứu đã tiến hành 19](#_Toc186091527)

[Xây dựng bố cục 19](#_Toc186091528)

[Lưới của trò chơi 19](#_Toc186091529)

[Các khối gạch 20](#_Toc186091530)

[Xây dựng logic game 21](#_Toc186091531)

[**Xây dựng lớp Board** 21](#_Toc186091532)

[Xây dựng lớp Brick 26](#_Toc186091533)

[**Xây dựng các hàm cần thiết** 31](#_Toc186091534)

[Lập trình điều khiển bằng cử chỉ tay bằng thư viện Mediapipe của JS 35](#_Toc186091535)

[CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU 40](#_Toc186091536)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 44](#_Toc186091537)

[DANH MỤC TÀI LIỆU THAM KHẢO 45](#_Toc186091538)

[PHỤ LỤC 46](#_Toc186091539)

**DANH MỤC HÌNH ẢNH – BẢNG BIỂU**

# TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

1. **Vấn đề nghiên cứu**

Đồ án này nghiên cứu các công nghệ về web như HTML, CSS, JS và công nghệ nhận diện cử chỉ tay bằng Mediapipe của JS. Đồ án hướng đến việc cải tiến game xếp gạch cổ điển điều khiển bằng phím, chuột bằng cách điều khiển bằng cử chỉ tay để cải thiện tương tác, nâng cao trải nghiệm người dùng.

1. **Các hướng tiếp cận**

Nghiên cứu lý thuyết về các công cụ và công nghệ sẽ sử dụng cho đồ án này như HTML, CSS, JS. Tìm hiểu xu hướng người dùng hiện nay để điều chỉnh cho phù hợp.

Áp dụng các lý thuyết đã nghiên cứu để làm lại tựa game xếp gạch với logic game giữ nguyên, ứng dụng MediaPipe giúp nhận diện và theo dõi cử chỉ tay trong thời gian thực.

Phát triển giao diện game trên web: Xây dựng game bằng các công nghệ web như HTML, CSS, và JS.

Xử lý logic game: Tập trung phát triển thuật toán điều khiển và cơ chế chơi phù hợp với các cử chỉ như di chuyển gạch, xoay gạch, và gạch xuống nhanh.

Tối ưu hóa trải nghiệm người dùng: Tích hợp âm thanh, hiệu ứng và giao diện bắt mắt để tăng tính hấp dẫn.

1. **Cách giải quyết vấn đề**

Tôi đã nghiên cứu nhiều tài liệu và tham khảo qua nhiều tựa game xếp gạch hiện có để áp dụng vào đồ án của mình.

Tôi đã xác định logic game như sau: Game sẽ có các logic như chạm trái, phải, xoay, hoàn thành hàng, game over,… Xác định các cử chỉ điều khiển: Giơ ngón cái/ngón út để di chuyển gach qua trái/phải (tùy bàn tay), xòe bàn tay để xoay, không có bàn tay để xuống nhanh.

Ngoài ra, tôi còn tìm hiểu xu hướng người dùng để thiết kế website với giao diện chơi game đẹp mắt, hài hòa, hiệu ứng sinh động, tích hợp âm thanh để cải thiện trải nghiệm người dùng.

Cuối cùng, tôi tiến hành xây dựng game với các logic và cử chỉ đã xác định bằng cách kết hợp các công nghệ đã nghiên cứu.

1. **Kết quả đạt được**

Game xếp gạch hoạt động trên web với các logic của game cổ điển, có thể điều khiển bằng cử chỉ tay trong thời gian thực.

Nhận diện chính xác cử chỉ tay, với độ trễ thấp và phản hồi nhanh.

Giao diện bắt mắt, hiệu ứng và âm thanh sống động thu hút người chơi.

# MỞ ĐẦU

1. **Lý do chọn đề tài**

Trong thời đại công nghệ phát triển mạnh mẽ như hiện nay, cùng với đó các công nghệ như nhận diện cử chỉ tay, nhận diện khuôn mặt, hay theo dõi chuyển động cơ thể đang dần được ứng dụng trong nhiều lĩnh vực từ y tế, giáo dục, dịch vụ đến giải trí và trò chơi. Các công nghệ đó mang lại lợi ích lớn trong việc tạo ra các giải pháp không chạm hay tương tác người - máy, nâng cao trải nghiệm người dùng và tính tiện lợi, nhanh chóng bằng cách sử dụng các chuyển động hay các đặc điểm của cơ thể để điều khiển thiết bị, nhận diện mặt người,... Trong bối cảnh đó, việc kết hợp công nghệ nhận diện cử chỉ tay với các ứng dụng giải trí, đặc biệt là trò chơi là xu hướng rất tiềm năng.

Game xếp gạch là một tựa game cổ điển được nhiều người yêu thích bởi tính giải trí, lối chơi đơn giản nhưng đầy tính tư duy. Tuy nhiên, các phiên bản hiện tại chủ yếu sử dụng bàn phím, chuột hoặc cảm ứng dẫn đến sự nhàm chán. Do đó tôi quyết định chọn đề tài "Phát triển game xếp gạch trên web sử dụng MediaPipe với tương tác điều khiển bằng cử chỉ tay" để tận dụng tiềm năng của công nghệ nhận diện cử chỉ, mang đến trải nghiệm chơi game mới mẻ. Không chỉ làm mới cách chơi của một tựa game quen thuộc, đề tài còn khẳng định tính ứng dụng thực tế của công nghệ nhận diện cử chỉ tay trong các sản phẩm giải trí, là tiền đề để góp phần thúc đẩy xu hướng tương tác người – máy trong tương lai.

1. **Mục đích**

Đổi mới cách chơi game xếp gạch cổ điển thông qua việc tích hợp công nghệ nhận diện cử chỉ tay, tạo nên một trải nghiệm điều khiển không cần bàn phím, chuột hay cảm ứng.

Nghiên cứu và ứng dụng công nghệ MediaPipe, khai thác khả năng nhận diện cử chỉ thời gian thực để điều khiển trò chơi mượt mà trên nền tảng web.

Xây dựng một sản phẩm hoàn chỉnh, vừa đảm bảo tính sáng tạo vừa có tiềm năng mở rộng ứng dụng thực tế trong lĩnh vực giải trí và công nghệ tương tác.

1. **Đối tượng nghiên cứu**

HTML, CSS, JS: Cách tạo nên một trang web trò chơi.

Công nghệ MediaPipe: Tập trung nghiên cứu cách nhận diện cử chỉ tay và tích hợp vào ứng dụng web.

Game xếp gạch cổ điển: Các cơ chế chơi, luật chơi, và thuật toán liên quan đến việc di chuyển, xoay, và thả gạch.

1. **Phạm vi nghiên cứu**

Về kỹ thuật: Tập trung phát triển và tích hợp MediaPipe vào game xếp gạch trên nền tảng web. Xây dựng giao diện và logic game phù hợp với tương tác cử chỉ.

Về tính năng: Cung cấp các tính năng cơ bản của game xếp gạch (di chuyển, xoay, thả gạch), đảm bảo trải nghiệm mượt mà với độ trễ thấp.

# CHƯƠNG 1: TỔNG QUAN

Cùng với sự phát triển mạnh mẽ của công nghệ, các phương thức tương tác giữa con người và máy tính đã không ngừng được cải tiến. Từ các thiết bị truyền thống như bàn phím và chuột, công nghệ ngày nay đã chuyển sang các phương thức tương tác không chạm, trong đó có nhận diện cử chỉ, nhận diện khuôn mặt, và theo dõi chuyển động cơ thể. Những tiến bộ này không chỉ cải thiện tính tiện lợi mà còn tạo ra nhiều trải nghiệm mới mẻ và sáng tạo trong các lĩnh vực như y tế, giáo dục,… và đặc biệt là giải trí.

Trong lĩnh vực trò chơi, các tựa game kinh điển như xếp gạch đã gắn liền với nhiều thế hệ người chơi nhờ tính đơn giản, giải trí nhưng đầy thách thức. Tuy nhiên, các cách điều khiển quen thuộc như bàn phím, chuột, hoặc cảm ứng dần trở nên nhàm chán, thiếu sự mới mẻ và không tận dụng hết khả năng của các công nghệ hiện đại. Điều này đặt ra yêu cầu về sự đổi mới cách chơi để nâng cao trải nghiệm người dùng.

Công nghệ MediaPipe, với khả năng nhận diện cử chỉ tay trong thời gian thực, cung cấp một giải pháp tiềm năng cho việc điều khiển trò chơi không cần thiết bị ngoại vi. Bằng cách sử dụng MediaPipe, có thể xây dựng một phiên bản game xếp gạch điều khiển bằng cử chỉ tay, người chơi chỉ cần sử dụng camera tích hợp để thực hiện các hành động như di chuyển, xoay hoặc thả gạch xuống nhanh mà không cần sử dụng đến các thiết bị ngoại vi rườm rà.

**Các bước thực hiện đồ án**

**Bước 1:** Nghiên cứu và thu thập thông tin

Tìm hiểu về trò chơi xếp gạch, bao gồm luật chơi, các cơ chế điều khiển, và cách hoạt động.

Nghiên cứu về công nghệ MediaPipe, tập trung vào khả năng nhận diện cử chỉ tay và tích hợp trên nền tảng web.

Tìm hiểu các công nghệ web cần thiết như HTML, CSS, JS để xây dựng giao diện và logic trò chơi.

Khảo sát các game xép gạch có sẵn để phân tích ưu, nhược điểm và định hướng cải tiến.

**Bước 2:** Phân tích yêu cầu và thiết kế hệ thống

Phân tích yêu cầu:

Xác định các cử chỉ tay cần nhận diện: di chuyển, xoay, thả gạch xuống nhanh.

Xác định các tính năng chính của game, như bắt đầu, chơi, lưu điểm và kết thúc.

Thiết kế hệ thống kiến trúc tổng quan bao gồm:

Phần giao diện (Front-end): Xây dựng bố cục game, bảng điểm, và hiển thị gạch.

Phần xử lý cử chỉ: Tích hợp MediaPipe để xử lý dữ liệu từ camera.

Phần logic game: Điều khiển chuyển động của các khối gạch dựa trên cử chỉ.

Thiết kế giao diện người dùng trực quan, dễ sử dụng.

**Bước 3:** Xây dựng hệ thống

Phát triển giao diện web:

Sử dụng HTML và CSS để xây dựng bố cục game, bao gồm giao diện nhập tên người và giao diện chơi game, đảm bảo tính thẩm mỹ và trực quan.

Sử dụng JS để tạo hiệu ứng động và lập trình các logic game.

Xây dựng logic game:

Lập trình các logic game như di chuyển, xoay khối gạch, thả gạch xuống nhanh. Xử lý các quy tắc chơi game như xóa hàng sau khi hoàn thành một hay nhiều hàng, cập nhật điểm số, điểm cao nhất, cập nhật tên người chơi lên bảng xếp hạng, lưu điểm và kết thúc trò chơi.

Tích hợp MediaPipe:

Cài đặt thư viện MediaPipe để cấu hình nhận diện cử chỉ tay thông qua camera trên thiết bị của người chơi.

Xử lí dữ liệu cử chỉ tay để chuyển đổi thành lệnh điều khiển (di chuyển, xoay, thả gạch xuống nhanh).

**Bước 4:** Kiểm thử và tối ưu hóa

Đảm bảo các cử chỉ tay được nhận diện chính xác và phản hồi đúng trong trò chơi.

Kiểm tra tính chính xác của các tính năng game (di chuyển, xoay, thả gạch xuống nhanh, xóa hàng, điểm số,…).

**Bước 5:** Hoàn thiện và tích hợp tích năng bổ sung

Cải thiện giao diện và trải nghiệm người dùng (thêm âm thanh, hiệu ứng, bảng xếp hạng).

Thiết kế hướng dẫn sử dụng cho người chơi.

**Bước 6:** Báo cáo và trình bày kết quả

Hoàn thiện tài liệu đồ án bao gồm quyển báo cáo, slide thuyết trình, poster, repository trên Github để chứa mã nguồn và tài liệu đã thực hiện.

Chuẩn bị bản demo sản phẩm và thuyết trình kết quả đạt được.

# CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

Trình bày cơ sở lý thuyết, lý luận, giả thiết khoa học và phương pháp nghiên cứu đã được sử dụng trong đồ án.

## HTML

### Khái niệm

HTML (Hyper Text Markup Language), hay còn gọi là "Ngôn ngữ đánh dấu siêu văn bản", là một ngôn ngữ đánh dấu dùng để xây dựng các trang web trên World Wide Web. HTML thường được sử dụng cùng với các công nghệ khác như CSS để tạo kiểu và JavaScript để thêm tính năng động cho trang web.

### Nguyên lý hoạt động của HTML

Khi nhập tên miền vào trình duyệt, trình duyệt sử dụng hệ thống DNS để xác định địa chỉ IP của máy chủ web, sau đó kết nối và nhận dữ liệu dưới dạng tệp HTML. Trình duyệt phân tích các thẻ HTML trong tệp, chuyển đổi chúng thành nội dung hiển thị hoặc thông tin mà người và máy tính có thể hiểu.

### Ứng dụng

HTML là ngôn ngữ quan trọng trong việc tổ chức bố cục và định hình các thành phần của trang web, đồng thời cho phép nhúng các tệp kỹ thuật số như nhạc, video và hình ảnh. Mặc dù có nhiều ngôn ngữ lập trình khác nhau, HTML vẫn giữ vai trò cốt lõi trong việc hiển thị nội dung trên mọi trang web, bất kể nền tảng hay ngôn ngữ lập trình được sử dụng.

## CSS

### Khái niệm

**CSS** (viết tắt của *Cascading Style Sheets*), hay còn gọi là "Bảng mẫu nạp chồng", là một ngôn ngữ được sử dụng để mô tả cách trình bày và định dạng giao diện của tài liệu HTML. CSS giúp thiết kế trang web trở nên trực quan, đẹp mắt, và dễ tùy chỉnh hơn.

### Chức năng chính

CSS được sử dụng để định dạng các phần tử HTML, bao gồm bố cục, màu sắc, phông chữ, kích thước, khoảng cách, hình nền, và nhiều thuộc tính khác.

Nó tách biệt nội dung (HTML) và hình thức (CSS), giúp quản lý trang web dễ dàng hơn.

### Cấu trúc của CSS

CSS hoạt động thông qua các quy tắc (ruleset), bao gồm:

Selector (bộ chọn): Chỉ định phần tử HTML cần áp dụng kiểu.

Declaration (khai báo): Bao gồm thuộc tính và giá trị, được đặt trong cặp ngoặc {}.

### Phân loại

CSS được chia thành 3 loại: Bảng kiểu trực tiếp (*Inline Style Sheet*), Bảng kiểu được nhúng vào tài liệu HTML (*Internal Style Sheet*), và Bảng kiểu bên ngoài (*External Style Sheet*). Những loại này có thể được sử dụng đồng thời, với thứ tự ưu tiên giảm dần như sau:

1. Bảng kiểu trực tiếp (*Inline Style Sheet*).
2. Bảng kiểu được nhúng vào tài liệu HTML (*Internal Style Sheet*).
3. Bảng kiểu bên ngoài (*External Style Sheet*).
4. Các thiết lập mặc định của trình duyệt (*Browser Default*).

Điều này có nghĩa là, nếu trong cùng một thẻ HTML có một thuộc tính được quy định bởi cả *Inline Style Sheet* và *Internal Style Sheet*, thì thuộc tính sẽ được áp dụng là thuộc tính trong *Inline Style Sheet*. Trong trường hợp một thẻ HTML được quy định nhiều thuộc tính bằng các loại CSS khác nhau, trình duyệt sẽ tổng hợp tất cả các định dạng để hiển thị. Nếu một thuộc tính không được định nghĩa trong bất kỳ loại CSS nào, trình duyệt sẽ áp dụng các thiết lập mặc định để định dạng thẻ đó.

## JavaScript

### Khái niệm

JavaScript là một ngôn ngữ lập trình phổ biến trong phát triển website hiện nay. Được tích hợp và nhúng vào HTML, JavaScript giúp các trang web trở nên sinh động và tương tác hơn. Ngôn ngữ này hoạt động như một phần của trang web, cho phép thực thi các đoạn mã phía máy khách (Client-side) và cả phía máy chủ (thông qua Node.js), từ đó tạo ra các trang web động và linh hoạt.

### Các đặc điểm chính

Động: JavaScript cho phép các trang web thay đổi nội dung và hành vi mà không cần tải lại trang.

Đa nền tảng: JavaScript chạy được trên hầu hết các trình duyệt hiện đại như Chrome, Firefox, Safari, Edge.

Hướng sự kiện: JavaScript phản hồi các hành động của người dùng như nhấp chuột, di chuyển chuột, hoặc nhập liệu.

Không đồng bộ: JavaScript hỗ trợ xử lý các tác vụ không đồng bộ, chẳng hạn như tải dữ liệu từ máy chủ mà không làm gián đoạn giao diện người dùng.

Được nhúng trong HTML: JavaScript có thể được nhúng trực tiếp trong tài liệu HTML thông qua thẻ <script>.

### Chức năng

Tương tác người dùng: Tạo các hiệu ứng như menu thả xuống, slideshow, hoặc xác thực biểu mẫu.

Xử lý logic: Xây dựng các ứng dụng web phức tạp như trò chơi, ứng dụng quản lý công việc.

Kết nối máy chủ: Gửi và nhận dữ liệu từ máy chủ bằng AJAX hoặc các công cụ hiện đại hơn như fetch hoặc thư viện Axios.

Điều khiển DOM: Thay đổi nội dung, cấu trúc, hoặc kiểu dáng của trang web mà không cần tải lại trang.

## Mediapipe

### Khái niệm

MediaPipe trên JavaScript là một phiên bản của MediaPipe được thiết kế để chạy trên các ứng dụng web, cho phép triển khai các giải pháp thị giác máy tính (computer vision) thời gian thực ngay trong trình duyệt mà không cần cài đặt hoặc sử dụng backend. Nó sử dụng WebAssembly để thực thi các mô hình AI với hiệu suất cao, đồng thời cung cấp các công cụ để xử lý video và hình ảnh trực tiếp từ webcam hoặc nguồn khác. Nó được tối ưu hóa để thực hiện các nhiệm vụ xử lý hình ảnh và video như nhận diện khuôn mặt, bàn tay, tư thế, phân đoạn nền và các giải pháp AI khác.

### Các tính năng chính

Chạy trực tiếp trên trình duyệt:

Không cần kết nối Internet hoặc máy chủ backend vì mọi tính toán được thực hiện trên máy cục bộ bằng trình duyệt.

Bảo mật và tiết kiệm tài nguyên mạng.

Xử lý thời gian thực: Các mô-đun của MediaPipe có thể xử lý luồng video trực tiếp từ webcam với độ trễ thấp.

Mô-đun đã được huấn luyện sẵn:

Hỗ trợ các giải pháp AI phổ biến như:

Theo dõi bàn tay (Hand Tracking).

Phát hiện khuôn mặt (Face Detection).

Tạo lưới 3D khuôn mặt (Face Mesh).

Nhận diện tư thế cơ thể (Pose Detection).

Phân đoạn nền (Selfie Segmentation).

Dễ dàng tích hợp với các công nghệ web:

Hoạt động tốt với các framework JavaScript như React, Angular, hoặc Vue.js.

Có thể được sử dụng thuần túy với HTML, CSS, và JavaScript.

Hiệu suất cao:

MediaPipe sử dụng GPU qua WebGL hoặc CPU để đảm bảo tốc độ xử lý nhanh và hiệu quả.

Ứng dụng:

Xây dựng các ứng dụng tương tác trên trình duyệt như trò chơi điều khiển bằng cử chỉ.

Tích hợp hiệu ứng thực tế tăng cường (AR) trong các ứng dụng video trực tiếp.

Phân tích dữ liệu thị giác trong thời gian thực cho thể thao, giáo dục, và giải trí.

1. **Cách cài đặt và sử dụng**

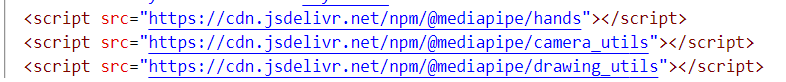
Để cài đặt và sử dụng **MediaPipe** với JavaScript, thực hiện các bước sau:

**Cài đặt**:

Sử dụng npm để cài đặt các package liên quan, ví dụ:



Hoặc sử dụng trực tiếp từ CDN qua <script>:



Gắn các <script> vào phần head của HTML.

**Sử dụng:**

Nhúng MediaPipe vào dự án, kết hợp với camera hoặc các nguồn video.

Sử dụng các công cụ như CameraUtils để lấy dữ liệu từ webcam và DrawingUtils để vẽ kết quả lên canvas.

Cấu hình mô-đun (ví dụ: HandPose, Pose, FaceMesh) và đăng ký callback để nhận kết quả phân tích.

Hiển thị kết quả bằng cách vẽ các điểm đặc trưng hoặc kết nối chúng.

Dùng một server cục bộ hoặc chạy file HTML trong trình duyệt để kiểm tra.

# CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

## Các bước nghiên cứu đã tiến hành

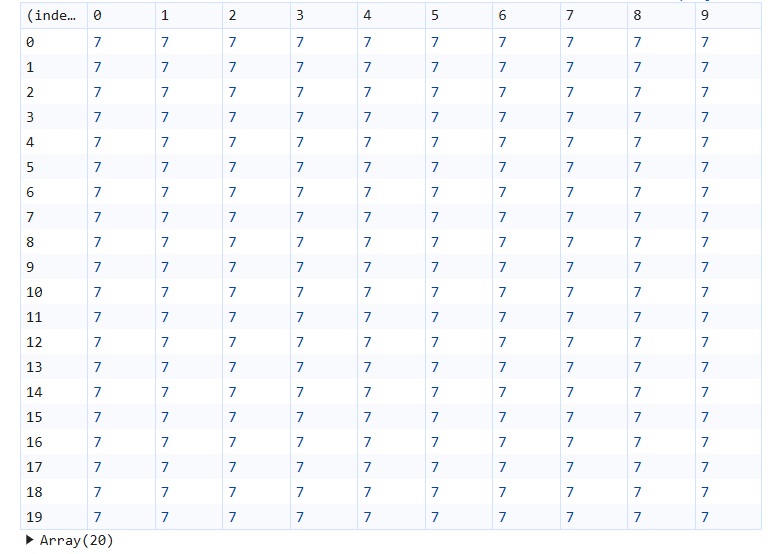
### Xây dựng bố cục

Lưới của trò chơi

Bản chất của trò chơi này là một lưới ma trận 2D gồm có 20 dòng và 10 cột. Mỗi ô trong lưới ma trận sẽ có 2 trạng thái:

7: nếu ô trống.

0 đến 6: tương ứng với id của gạch và màu sắc gạch.



Hình 1: Mô tả lưới ma trận khi chưa có gạch

**Vai trò của lưới**

Lưu trữ trạng thái: Mỗi ô trong lưới cho biết liệu vị trí đó đang trống hay đã bị chiếm bởi gạch.

Kiểm tra va chạm: Khi khối gạch di chuyển, trò chơi kiểm tra xem gạch có va chạm với:

Cạnh bảng (cạnh trái, phải, hoặc đáy).

Các ô đã bị chiếm bởi gạch khác.

Xóa hàng đầy: Nếu một hàng trong lưới đầy đủ (tất cả ô đều bị chiếm), hàng đó sẽ bị xóa, và được thay thế bởi các số 7 (tương ứng với ô trống).

Các khối gạch

Mỗi khối gạch được định nghĩa bằng một mảng 2D nhỏ chứa giá trị 7 và 0 đến 6 tương ứng với id gạch và màu gạch:

0 đến 6: Ô thuộc khối gạch.

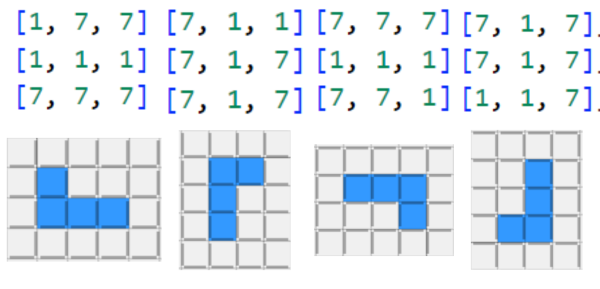
0: Ô trống.

Có 7 kiểu gạch khác nhau, mỗi kiểu gạch có một id và một mã màu tương ứng.



Hình 2: Mô tả 7 kiểu của khối gạch

Mỗi khối gạch sẽ có 4 hướng.

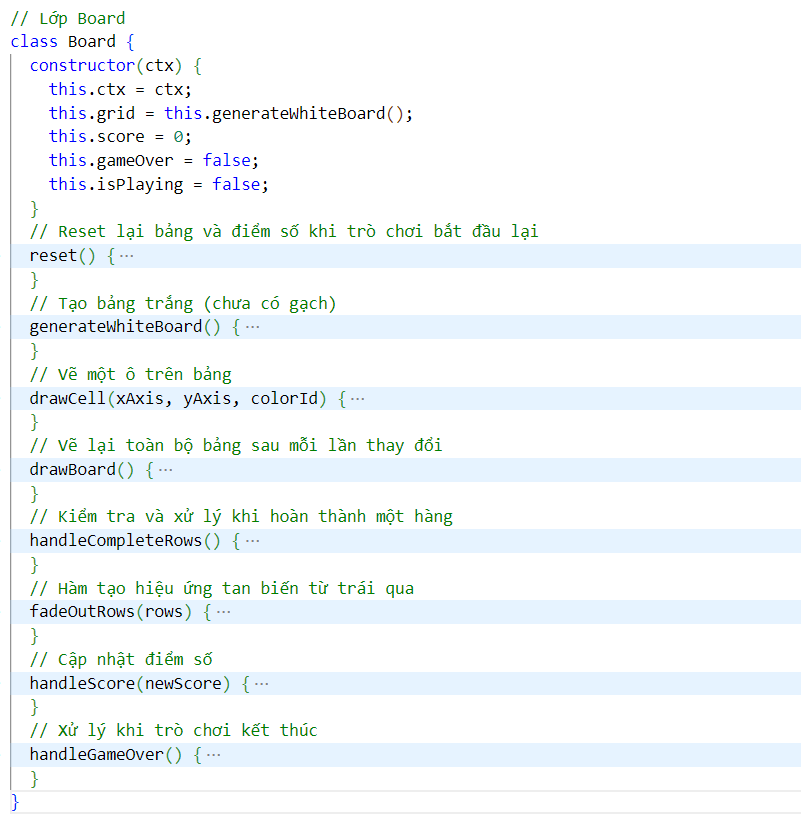


Hình 3: Mô tả chi tiết 1 khối gạch và hướng của nó

### Xây dựng logic game

**Xây dựng lớp Board**

Lớp Board được thiết kế để quản lý và vận hành các chức năng cốt lõi của trò chơi xếp gạch. Đây là một thành phần chính chịu trách nhiệm về trạng thái của bảng, các thao tác liên quan đến bảng, và việc cập nhật giao diện trò chơi.



Hình 4: Mô tả lớp Board

**Constructor:**

Nhận vào một context để khởi tạo các giá trị mặc định khi tạo một bảng và gọi phương thức generateWhiteBoard() để tạo bảng trắng ban đầu.

**ctx**: Context vẽ trên canvas, cho phép vẽ các hình dạng và màu sắc trên bảng.

**grid**: Ma trận 2D biểu diễn trạng thái hiện tại của bảng (mỗi ô có một ID màu).

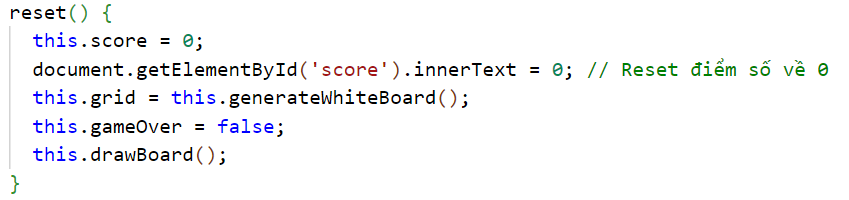
**score**: Điểm số hiện tại của trò chơi.

**gameOver**: Biến để xác định trạng thái kết thúc của trò chơi.

**isPlaying**: Biến để theo dõi xem trò chơi có đang chạy hay không.

**Phương thức:**

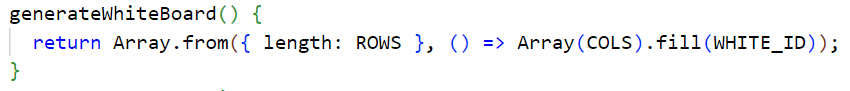
**Reset()**:

****

Được sử dụng để khởi tạo lại trạng thái ban đầu của trò chơi. Nó đặt lại điểm số về 0, cập nhật hiển thị điểm số trên giao diện, tạo một bảng trò chơi mới toàn ô trắng, thiết lập trạng thái trò chơi không kết thúc (gameOver = false), và vẽ lại bảng trò chơi trên giao diện.

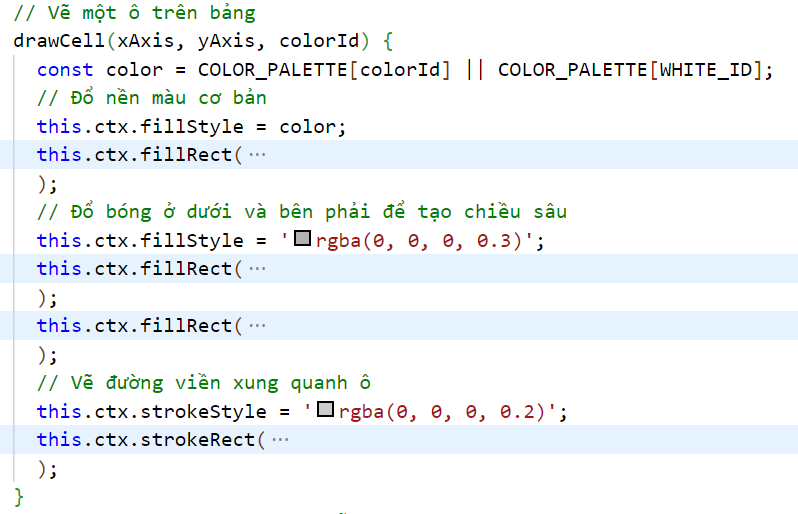
**Vẽ bảng và ô**

**generateWhiteBoard():**



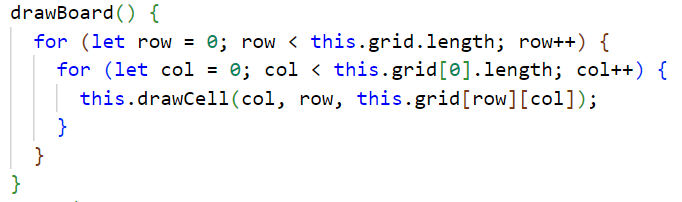
Tạo một bảng 2D với tất cả các ô được gán giá trị mặc định WHITE\_ID.

**drawCell():**



Vẽ một ô cụ thể trên bảng với màu sắc từ COLOR\_PALETTE với vị trí xAxis và yAxis được truyền vào. Tạo hiệu ứng đổ bóng ở dưới và bên phải để tạo cảm giác 3D, vẽ đường viền ô.

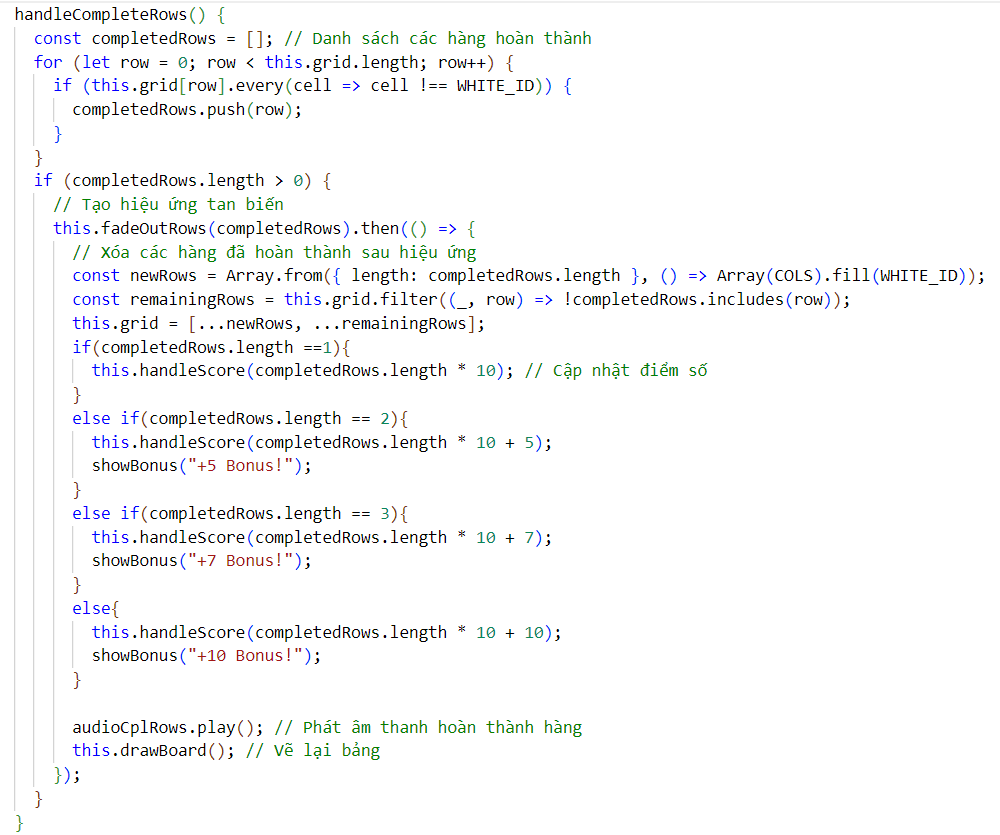
**drawBoard()**:



Lặp qua từng ô trong grid và vẽ toàn bộ bảng.

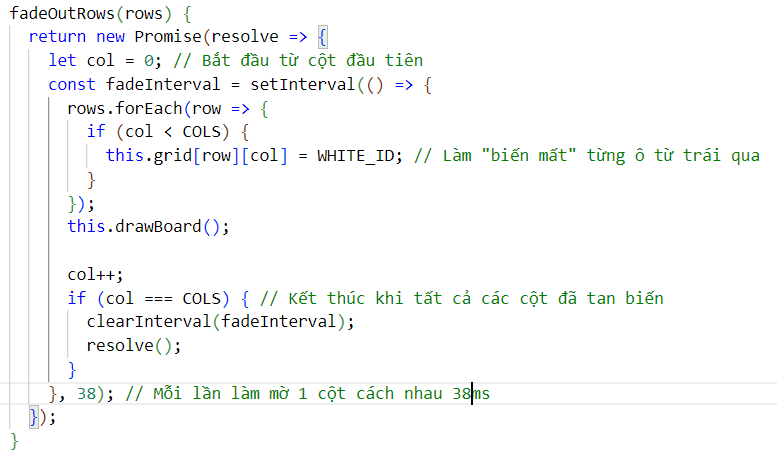
**Xử lý hàng hoàn thành**

**handleCompleteRows()**:



Kiểm tra và xử lý các hàng hoàn thành trong lưới trò chơi. Nó xác định các hàng không chứa ô trắng (WHITE\_ID), sau đó kích hoạt hiệu ứng tan biến bằng fadeOutRows(). Sau hiệu ứng, các hàng hoàn thành được thay thế bằng các hàng mới toàn ô trắng, kết hợp với các hàng còn lại để cập nhật lưới. Phương thức cũng tăng điểm số dựa trên số lượng hàng xóa và thêm thưởng tùy số hàng hoàn thành, hiển thị thông báo thưởng với showBonus(), phát âm thanh bằng audioCplRows và vẽ lại bảng trò chơi để hiển thị lưới mới.

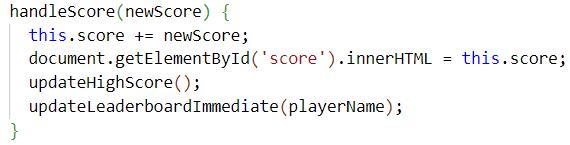
**fadeOutRows():**

****

Tạo hiệu ứng tan biến từng cột cho các hàng được chỉ định. Nó trả về một Promise, trong đó mỗi cột của các hàng được làm biến mất dần từ trái sang phải bằng cách đặt các ô trong cột đó về trạng thái trắng (WHITE\_ID). Phương thức sử dụng setInterval để thực hiện hiệu ứng theo từng khoảng thời gian (38ms). Sau khi tất cả các cột đã được xử lý, setInterval được dừng lại bằng clearInterval, và Promise được giải quyết để báo hiệu hiệu ứng đã hoàn thành.

**Điểm số**

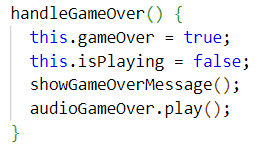
**handleScore()**:



Cập nhật điểm số của trò chơi bằng cách cộng thêm giá trị newScore vào tổng điểm hiện tại (this.score). Sau đó, điểm số được hiển thị lên giao diện thông qua phần tử HTML có ID score. Đồng thời, phương thức gọi các hàm updateHighScore() để cập nhật điểm cao nhất và updateLeaderboardImmediate(playerName) để làm mới bảng xếp hạng theo tên người chơi.

**Kết thúc trò chơi**

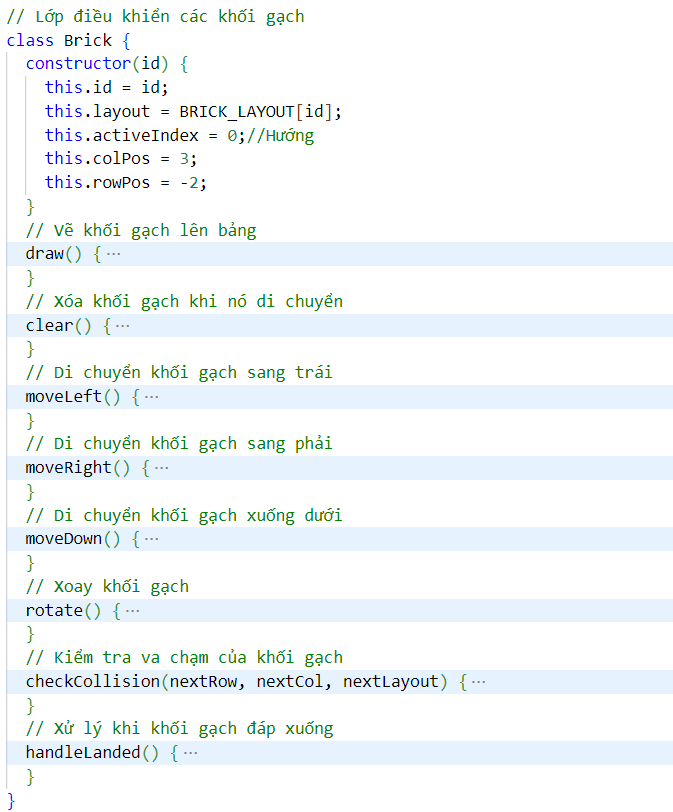
**handleGameOver()**:



Xử lý khi trò chơi kết thúc bằng cách đặt trạng thái gameOver thành true và dừng trò chơi (isPlaying = false). Nó hiển thị thông báo kết thúc bằng showGameOverMessage() và phát âm thanh game over qua audioGameOver.

#### Xây dựng lớp Brick

Lớp Brick được thiết kế để đại diện cho từng khối gạch trong trò chơi xếp gạch. Đây là thành phần quản lý cách mà khối gạch xuất hiện, di chuyển, xoay và tương tác với bảng (Board).



Hình 5: Mô tả lớp Brick

**Constructor:**

Nhận vào một id để khởi tạo các giá trị mặc định khi tạo khối brick.

**id**: Đại diện cho loại khối gạch (mỗi loại có màu sắc và hình dạng riêng, được xác định trong BRICK\_LAYOUT).

**layout**: Mảng 2D chứa thông tin bố cục của khối gạch (các ô nào được lấp đầy).

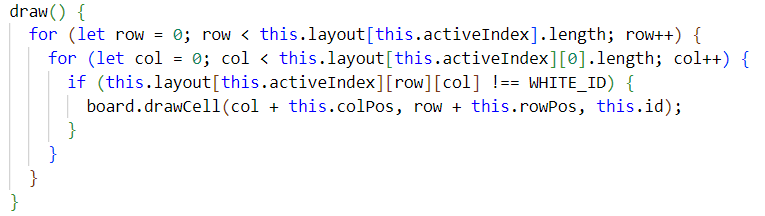
**activeIndex**: Chỉ số để xác định hướng hiện tại của khối gạch (xoay 0°, 90°, 180°, 270°).

**colPos và rowPos**: Vị trí hiện tại của khối gạch trên bảng, được xác định bởi cột và hàng.

**Phương thức**

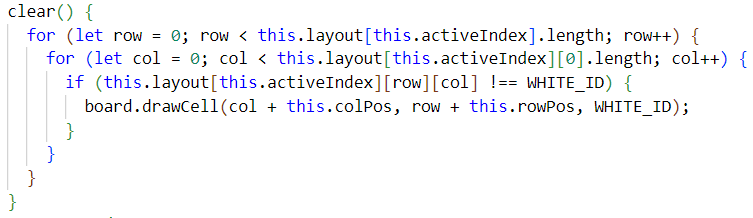
**Vẽ và Xóa Khối Gạch**

**draw()**:



Vẽ hình dạng hiện tại của một khối trên bảng trò chơi. Nó lặp qua từng hàng và cột của ma trận bố cục khối (this.layout[this.activeIndex]) và xác định các ô không phải màu trắng (WHITE\_ID). Với mỗi ô hợp lệ, nó tính toán vị trí chính xác trên bảng dựa vào this.colPos và this.rowPos, sau đó gọi phương thức board.drawCell() để vẽ ô tại vị trí đó với ID của khối (this.id).

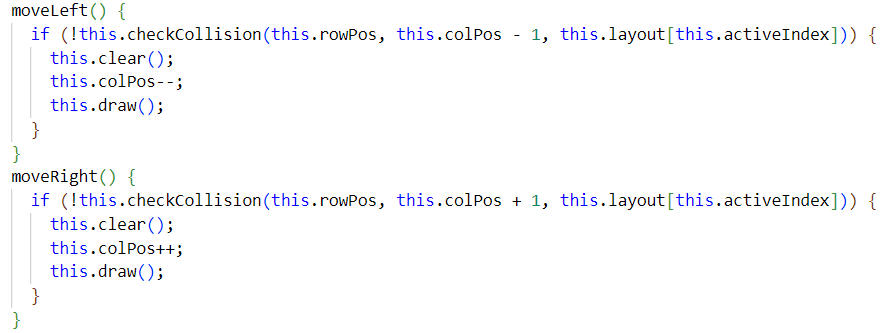
**clear()**:



Xóa khối hiện tại khỏi bảng trò chơi bằng cách lặp qua ma trận bố cục khối (this.layout[this.activeIndex]). Với mỗi ô không phải màu trắng (WHITE\_ID), nó tính toán vị trí trên bảng dựa vào this.colPos và this.rowPos, sau đó gọi phương thức board.drawCell() để vẽ lại ô đó với màu trắng (WHITE\_ID), trả bảng về trạng thái không chứa khối ở vị trí này.

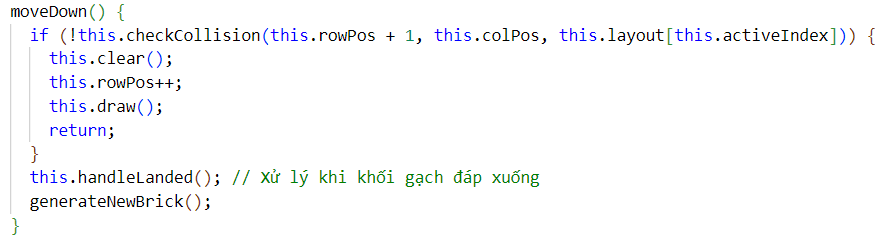
**Di Chuyển Khối Gạch**

**moveLeft() và moveRight()**:



Di chuyển khối gạch sang trái hoặc phải. Trước khi di chuyển, chúng kiểm tra va chạm bằng checkCollision() ở vị trí dự kiến. Nếu không có va chạm, phương thức sẽ xóa khối khỏi vị trí hiện tại bằng clear(), cập nhật cột vị trí (colPos) theo hướng tương ứng, và vẽ lại khối ở vị trí mới bằng draw().

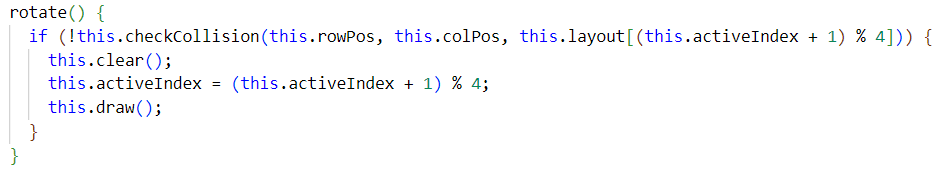
**moveDown()**:



Di chuyển khối gạch xuống một hàng nếu không có va chạm tại vị trí dự kiến. Nếu không có va chạm, nó xóa khối khỏi vị trí hiện tại bằng clear(), tăng giá trị hàng (rowPos), và vẽ lại khối bằng draw(). Nếu xảy ra va chạm, phương thức gọi handleLanded() để xử lý khi khối đáp xuống và tạo khối gạch mới bằng generateNewBrick().

**Xoay Khối Gạch**

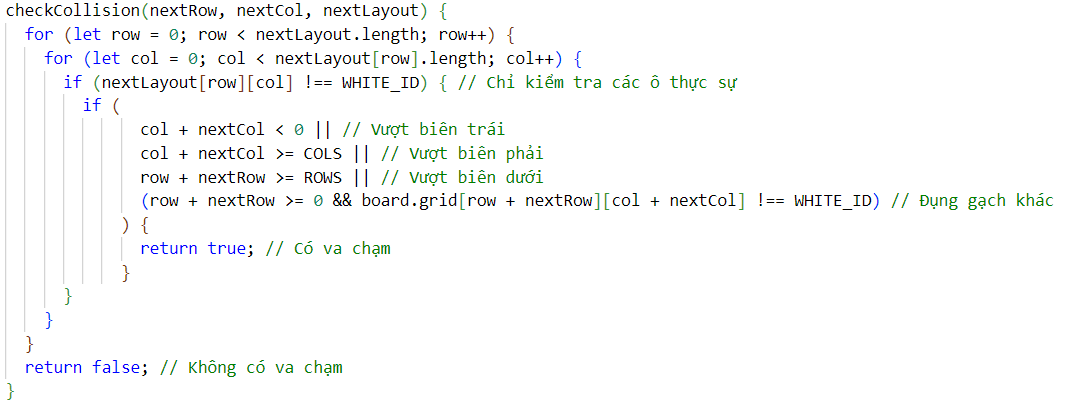
**rotate()**:



Xoay khối gạch theo chiều kim đồng hồ nếu không có va chạm với các ô khác sau khi xoay. Nó kiểm tra va chạm với hình dạng kế tiếp của khối (bằng cách sử dụng chỉ số xoay activeIndex + 1 modulo 4 để đảm bảo quay vòng qua các trạng thái hình dạng của khối). Nếu không có va chạm, phương thức sẽ xóa khối khỏi vị trí hiện tại, cập nhật chỉ số hình dạng (activeIndex), và vẽ lại khối ở vị trí mới.

**Kiểm Tra Va Chạm**

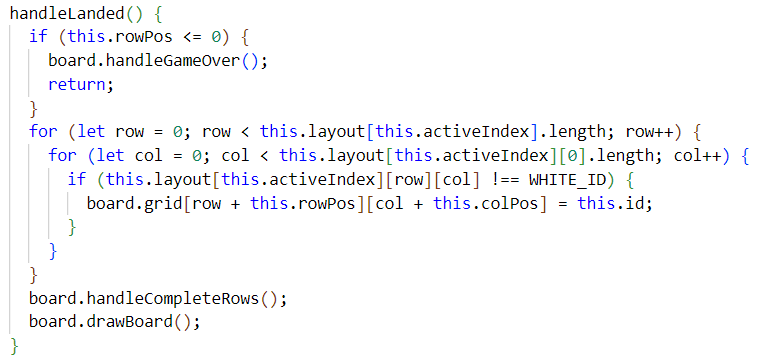
**checkCollision()**:



Kiểm tra xem khối gạch có va chạm với biên hoặc các ô đã tồn tại trên bảng hay không bằng cách duyệt qua từng ô trong bố cục mới (nextLayout). Nó bỏ qua các ô trống (giá trị WHITE\_ID) và kiểm tra các điều kiện: ô nằm ngoài biên trái, phải, hoặc dưới của bảng, hoặc trùng với một ô đã có gạch trong board.grid. Nếu phát hiện bất kỳ va chạm nào, hàm trả về true, ngược lại trả về false.

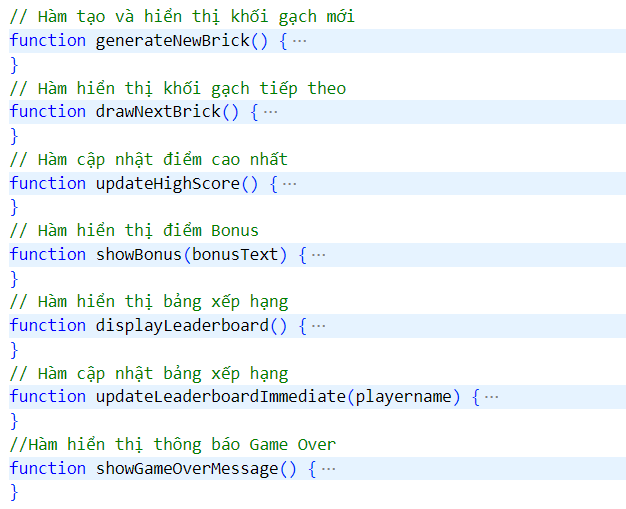
**Xử Lý Khi Khối Gạch Đáp Xuống**

**handleLanded()**:



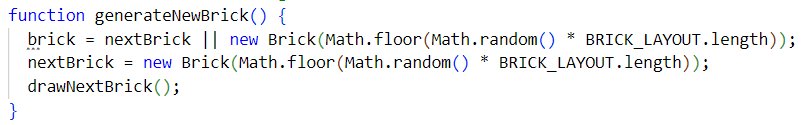
Xử lý khi khối gạch đã đáp xuống vị trí cuối cùng. Nếu khối gạch nằm ở hàng trên cùng (rowPos <= 0), trò chơi kết thúc bằng cách gọi board.handleGameOver(). Ngược lại, nó gắn các ô của khối gạch hiện tại vào bảng (board.grid) dựa trên vị trí hiện tại (rowPos, colPos). Sau đó, hàm kiểm tra và xử lý các hàng đầy bằng board.handleCompleteRows() rồi vẽ lại bảng bằng board.drawBoard().

**Xây dựng các hàm cần thiết**

****

Hình 6: Mô tả các hàm có trong game

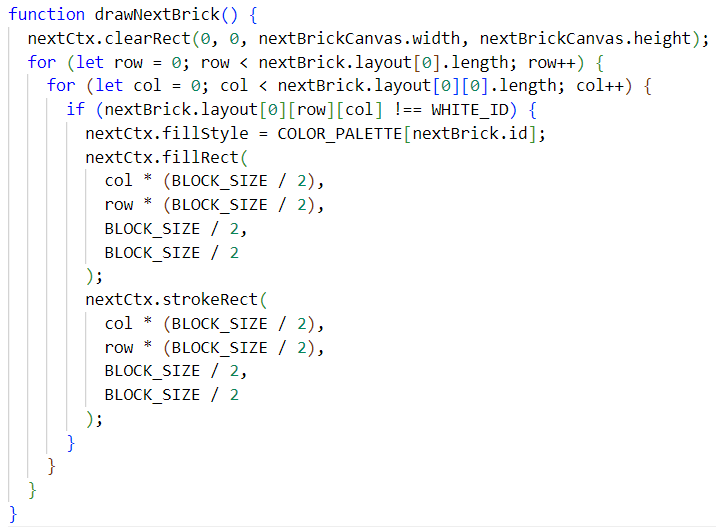
**generateNewBrick():**

****

**Chức năng**: Tạo và cập nhật khối gạch hiện tại (brick) và khối gạch kế tiếp (nextBrick).

**Cách hoạt động**: Nếu đã có khối gạch kế tiếp (nextBrick), nó được gán làm khối gạch hiện tại. Sau đó, tạo khối gạch kế tiếp ngẫu nhiên và vẽ nó lên canvas nhỏ qua hàm drawNextBrick().

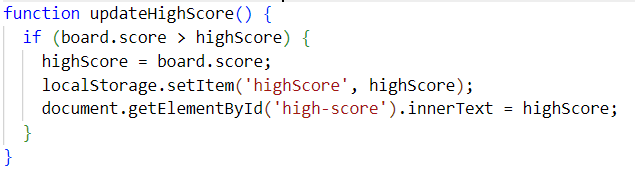
**drawNextBrick():**

****

**Chức năng**: Hiển thị khối gạch kế tiếp trong vùng canvas phụ.

**Cách hoạt động**: Xóa sạch canvas hiện tại của khối gạch kế tiếp. Vẽ từng ô của khối gạch kế tiếp với màu sắc tương ứng từ COLOR\_PALETTE và các kích thước bằng phân nửa kích thức của gạch trên lưới.

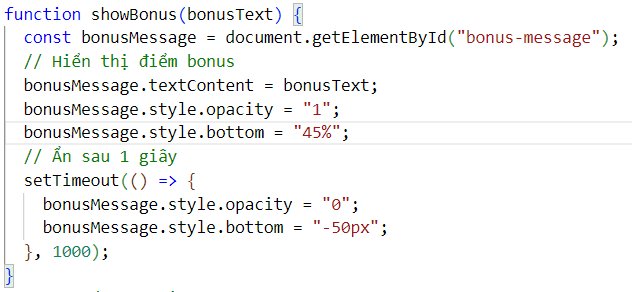
**updateHighScore():**

****

**Chức năng**: Cập nhật điểm cao nhất.

**Cách hoạt động**: So sánh điểm hiện tại (board.score) với highScore. Nếu điểm hiện tại cao hơn, lưu lại điểm mới vào localStorage và cập nhật điểm cao nhất trên giao diện.

**showBonus():**

****

**Chức năng**: Hiển thị thông báo điểm thưởng khi hoàn thành nhiều hàng.

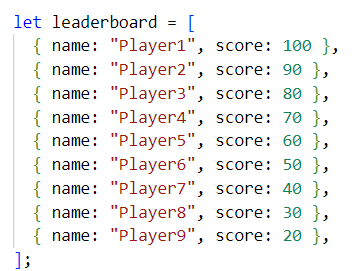
**Cách hoạt động**: Cập nhật nội dung và vị trí của thông báo thưởng. Sau 1 giây, ẩn thông báo.

**displayLeaderboard():**

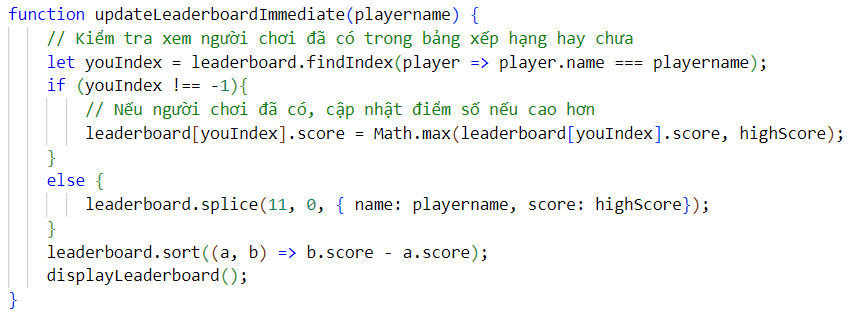
**Chức năng:** hiển thị bảng xếp hạng trên giao diện người dùng bằng cách cập nhật nội dung của một phần tử HTML.

**Cách hoạt động:** Sử dụng getElementById để lấy phần tử HTML có ID là "leaderboard". Đặt innerHTML của phần tử bằng chuỗi rỗng ("") để xóa bất kỳ nội dung nào đang hiển thị trước đó. Duyệt qua mảng leaderboard bằng forEach, trong đó mỗi phần tử là một đối tượng player với thuộc tính name và score. Với mỗi người chơi, thêm một mục danh sách (<li>) vào innerHTML của phần tử, bao gồm thứ hạng (index + 1), tên người chơi (player.name), và điểm số (player.score).

**Khởi tạo mảng danh sách các người chơi mặc đinh:**



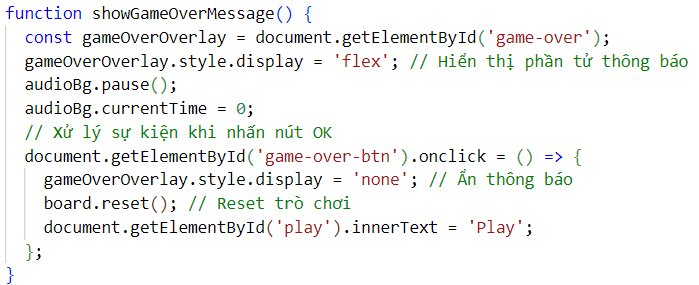
**updateLeaderboardImmediate():**

****

**Chức năng**: Cập nhật bảng xếp hạng ngay lập tức khi người chơi đạt được điểm số cao mới.

**Cách hoạt động**: Sử dụng findIndex để tìm vị trí của người chơi (playername) trong mảng leaderboard. Nếu người chơi đã có trong bảng xếp hạng (youIndex !== -1), cập nhật điểm số của họ bằng cách lấy giá trị lớn hơn giữa điểm hiện tại của họ và điểm cao mới (highScore). Nếu người chơi không có trên bảng xếp hạng, thêm họ vào mảng bằng splice với tên và điểm cao (highScore). Dùng sort để sắp xếp danh sách theo thứ tự giảm dần của điểm số. Gọi hàm displayLeaderboard để cập nhật giao diện hiển thị bảng xếp hạng.

**showGameOverMessage():**

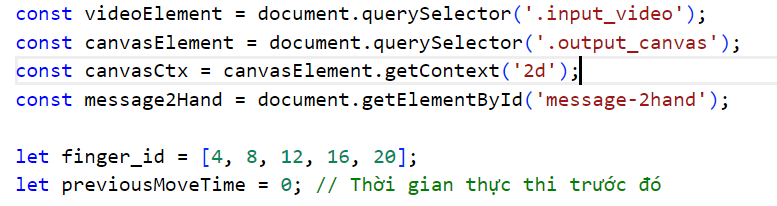
****

**Chức năng**: Hiển thị thông báo khi trò chơi kết thúc.

**Cách hoạt động**: Nó thay đổi thuộc tính style.display của phần tử HTML có ID game-over thành "flex" để hiển thị thông báo, đồng thời tạm dừng và đặt lại nhạc nền (audioBg). Khi người dùng nhấn nút "OK" (phần tử có ID game-over-btn), hàm ẩn thông báo, đặt lại trò chơi bằng board.reset(), và thay đổi nút "Play" trên giao diện về trạng thái ban đầu.

Lập trình điều khiển bằng cử chỉ tay bằng thư viện Mediapipe của JS

Khởi tạo các phần tử và biến

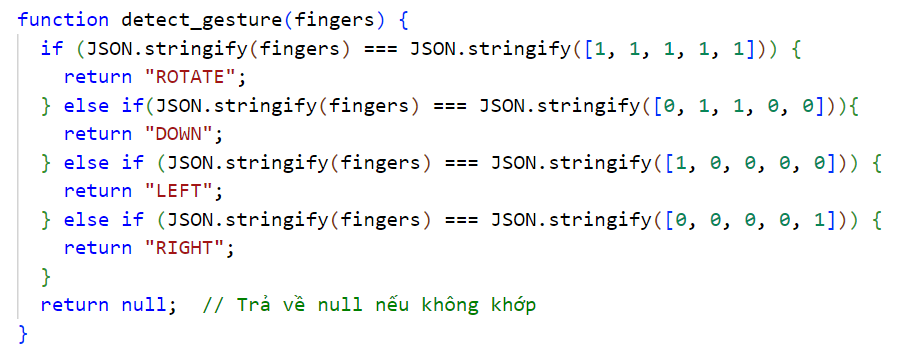


Chọn các phần tử DOM cần thiết (video, canvas, message-2hand).

finger\_id: Mảng chứa các chỉ số của các ngón tay trong MediaPipe.

previousMoveTime: Lưu lại thời gian lần cuối thực thi cử chỉ để giới hạn tốc độ thực thi.

Hàm nhận diện cử chỉ



Nhận đầu vào là mảng fingers đại diện trạng thái của 5 ngón tay (1: giơ lên, 0: cụp xuống).

So sánh fingers với các mẫu xác định:

[1, 1, 1, 1, 1]: Tất cả ngón tay giơ lên → Lệnh "ROTATE".

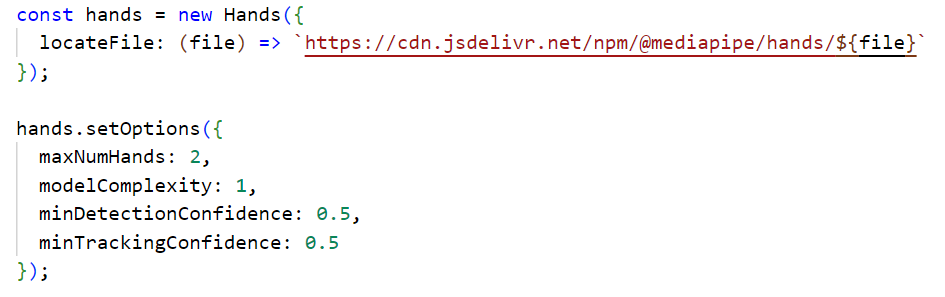
[0, 1, 1, 0, 0]: Ngón trỏ và ngón giữa giơ lên → Lệnh "DOWN".

[1, 0, 0, 0, 0]: Chỉ ngón cái/ngón út giơ lên(bàn phải/trái)→ Lệnh "LEFT".

[0, 0, 0, 0, 1]: Chỉ ngón cái/ngón út giơ lên(bàn tay trái/phải) → Lệnh "RIGHT".

Trả về null nếu không có cử chỉ khớp.

Cấu hình MediaPipe Hands



Tạo đối tượng Hands từ MediaPipe, cấu hình để tải tệp từ CDN.

Cài đặt tùy chọn:

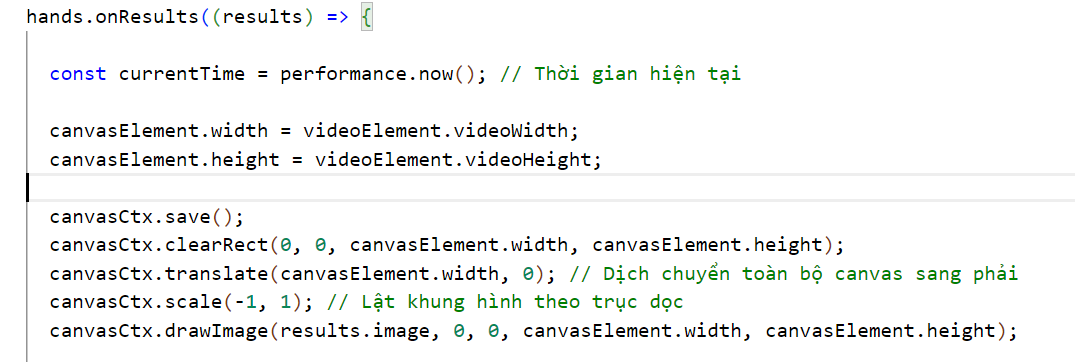
maxNumHands: Nhận diện tối đa 2 bàn tay.

modelComplexity: Độ phức tạp của mô hình (1 là cao nhất).

minDetectionConfidence: Ngưỡng tối thiểu để phát hiện bàn tay (0.5).

minTrackingConfidence: Ngưỡng tối thiểu để theo dõi bàn tay (0.5).

Xử lý kết quả nhận diện

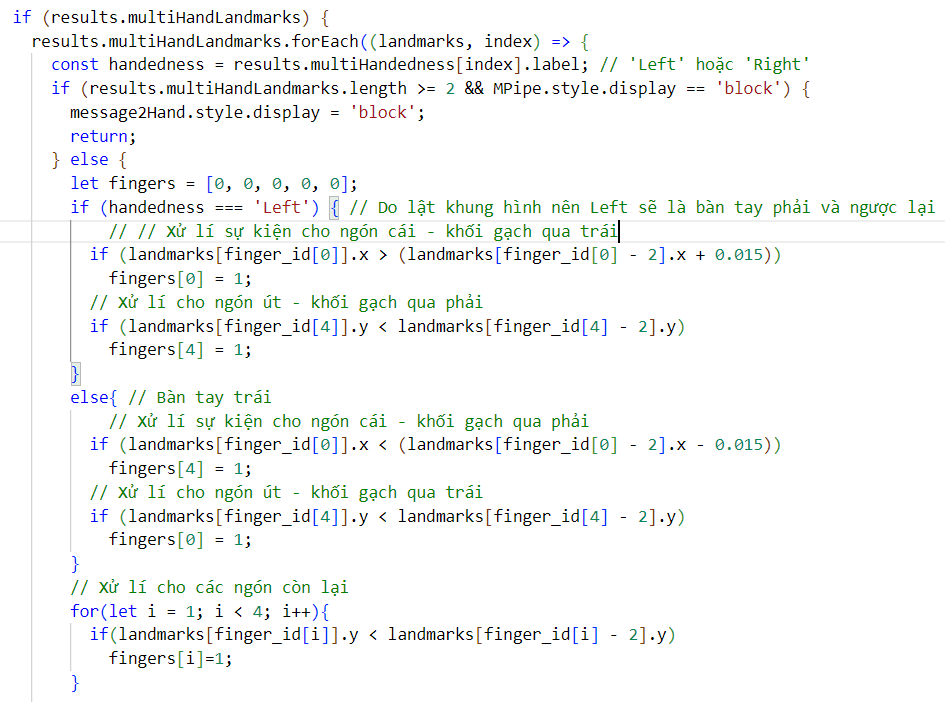


Hằng currentTime để cập nhật thời gian hiện tại, sử dụng để tính toán thời gian thực thi lệnh trước đó.

Cập nhật kích thước canvas theo video.

Xóa nội dung cũ trên canvas và lật hình ảnh từ camera để tương ứng góc nhìn người dùng.

Nhận diện trạng thái các ngón tay



**Kiểm tra số lượng bàn tay**

MediaPipe có thể phát hiện nhiều bàn tay. Nếu phát hiện nhiều hơn 1 bàn tay, hiển thị thông báo yêu cầu người chơi chỉ sử dụng một bàn tay và dừng xử lý.

**Xác định trạng thái ngón tay**

Mảng fingers được khởi tạo với 5 phần tử, mỗi phần tử biểu thị trạng thái của một ngón tay:

1 là giơ lên.

0 là cụp xuống.

**Phân biệt bàn tay trái và phải**

Sử dụng thông tin handedness từ MediaPipe để xác định bàn tay:

Nếu handedness là Left, nghĩa là bàn tay phải trong thực tế (do hình ảnh đã lật trên canvas).

Ngược lại, nếu handedness là Right, đó là bàn tay trái.

**Xử lý trạng thái ngón cái và ngón út**

Với bàn tay phải (labeled "Left"):

Nếu tọa độ x (landmarks = 4) của ngón cái lớn hơn tọa độ x (landmarks = 2), ngón cái được đánh dấu hoạt động (di chuyển khối gạch sang trái).

Nếu tọa độ y (landmarks = 20)của ngón út nhỏ hơn tọa độ y (landmarks = 18), ngón út hoạt động (di chuyển khối gạch sang phải).

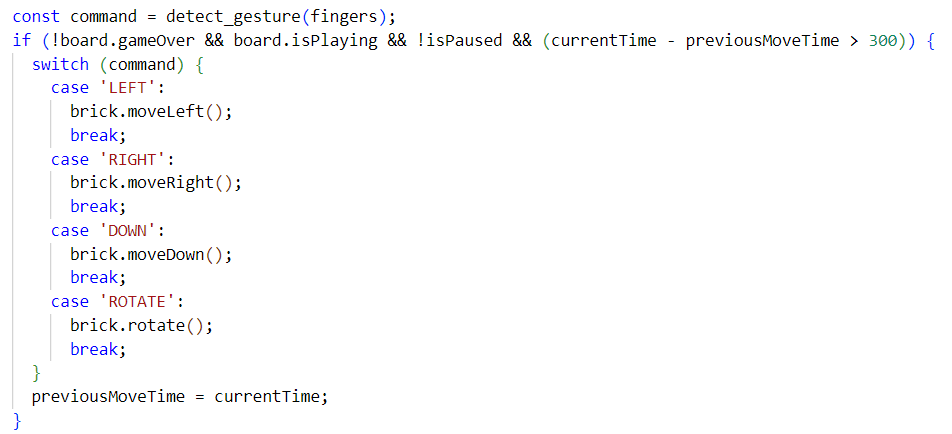
Với bàn tay trái (labeled "Right"):

Kiểm tra ngược lại ngón cái/út để di chuyển khối gạch sang trái/phải.

**Xử lý trạng thái các ngón còn lại**

Với cả hai tay, nếu tọa độ y (landmarks = 12, 14, 16) của ngón nhỏ hơn tọa độ y (landmarks = 10, 12, 14), ngón đó được đánh dấu hoạt động.

1. Thực thi lệnh trò chơi

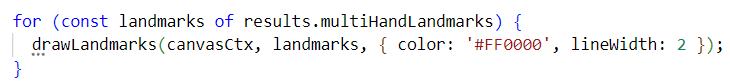


Tạo mới một hằng command và gán giá trị bằng hàm detect\_gesture (truyền vào hàm mảng fingers sau khi duyệt qua các điều kiện) để nhận về các giá trị LEFT, RIGHT, DOWN, ROTATE tương ứng với di chuyển và xoay khối gạch. Nếu lệnh từ command trả về thuộc 1 trong các lệnh LEFT, RIGHT, DOWN, ROTATE thì gọi hàm tương ứng.

Kiểm tra trạng thái trò chơi (gameOver, isPlaying, isPaused) và thời gian thực thi lệnh trước đó để tránh lệnh được thực hiện liên tục nếu người chơi không cụp tay kịp.

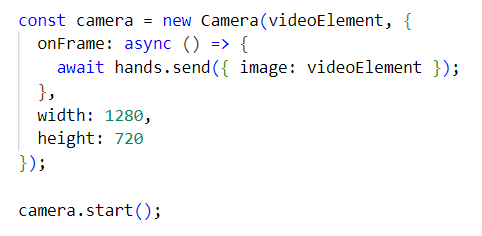
Đặt previousMoveTime lại bằng currentTime để tiếp tục vòng lặp mơi.

1. Vẽ các điểm landmark lên canvas



Vẽ bàn tay lên canvas bằng cách sử dụng dữ liệu từ results.multiHandLandmarks. Với mỗi tập hợp tọa độ bàn tay (landmarks), hàm drawLandmarks vẽ các điểm đặc trưng này bằng màu đỏ với độ dày đường viền là 2, tạo hình ảnh trực quan về bàn tay trên canvas.

1. Kích hoạt camera



Khởi tạo và bắt đầu một đối tượng Camera để sử dụng video từ camera của người dùng. Mỗi khi có một khung hình mới từ video (onFrame), hình ảnh đó sẽ được gửi đến mô-đun nhận diện bàn tay (hands.send). Video được phát ở độ phân giải 1280x720 pixel, và camera sẽ bắt đầu quay khi gọi camera.start().

# CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

Trình bày các kết quả đạt được sau quá trình thực hiện đồ án. Có thể đánh giá về hiệu năng, trải nghiệm người dùng, hoặc trình bày các giao diện chức năng của nghiên cứu ở phần này.

Sau quá trình thực hiện đồ án **"Phát triển game xếp gạch trên web sử dụng MediaPipe với tương tác điều khiển bằng cử chỉ tay"**, các kết quả chính đạt được bao gồm:

**Hiệu năng của hệ thống**

**Độ chính xác tương đối cao**: Việc nhận diện cử chỉ tay đạt độ chính xác tương đối cao trong điều kiện ánh sáng đủ và người chơi ở đủ gần, giúp các thao tác điều khiển game diễn ra mượt mà.

**Tối ưu hóa tài nguyên**: Hệ thống chạy ổn định trên trình duyệt web mà không yêu cầu phần cứng cao cấp hoặc cài đặt phần mềm bổ sung.

**Trải nghiệm người dùng**

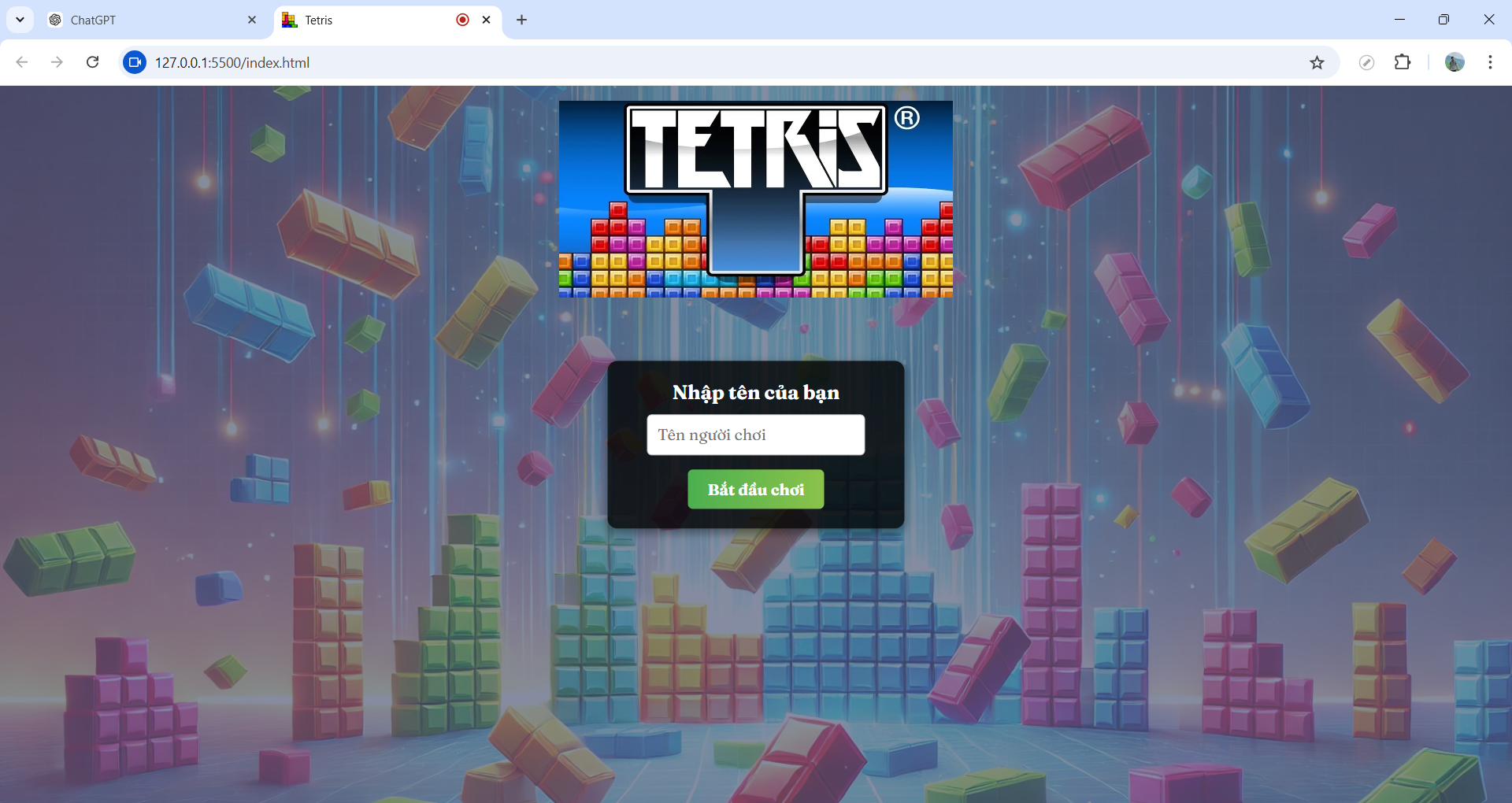
**Tương tác tự nhiên**: Người chơi có thể sử dụng các cử chỉ tay đơn giản để điều khiển các khối gạch, tạo cảm giác mới mẻ và hứng thú so với cách chơi truyền thống bằng bàn phím.

**Giao diện thân thiện**: Game có giao diện trực quan, dễ sử dụng, giúp người chơi nhanh chóng làm quen với cách điều khiển.

**Hỗ trợ đa nền tảng**: Chạy tốt trên các trình duyệt phổ biến như Chrome, Firefox, và Edge, đồng thời tương thích với các thiết bị như laptop, PC có webcam.

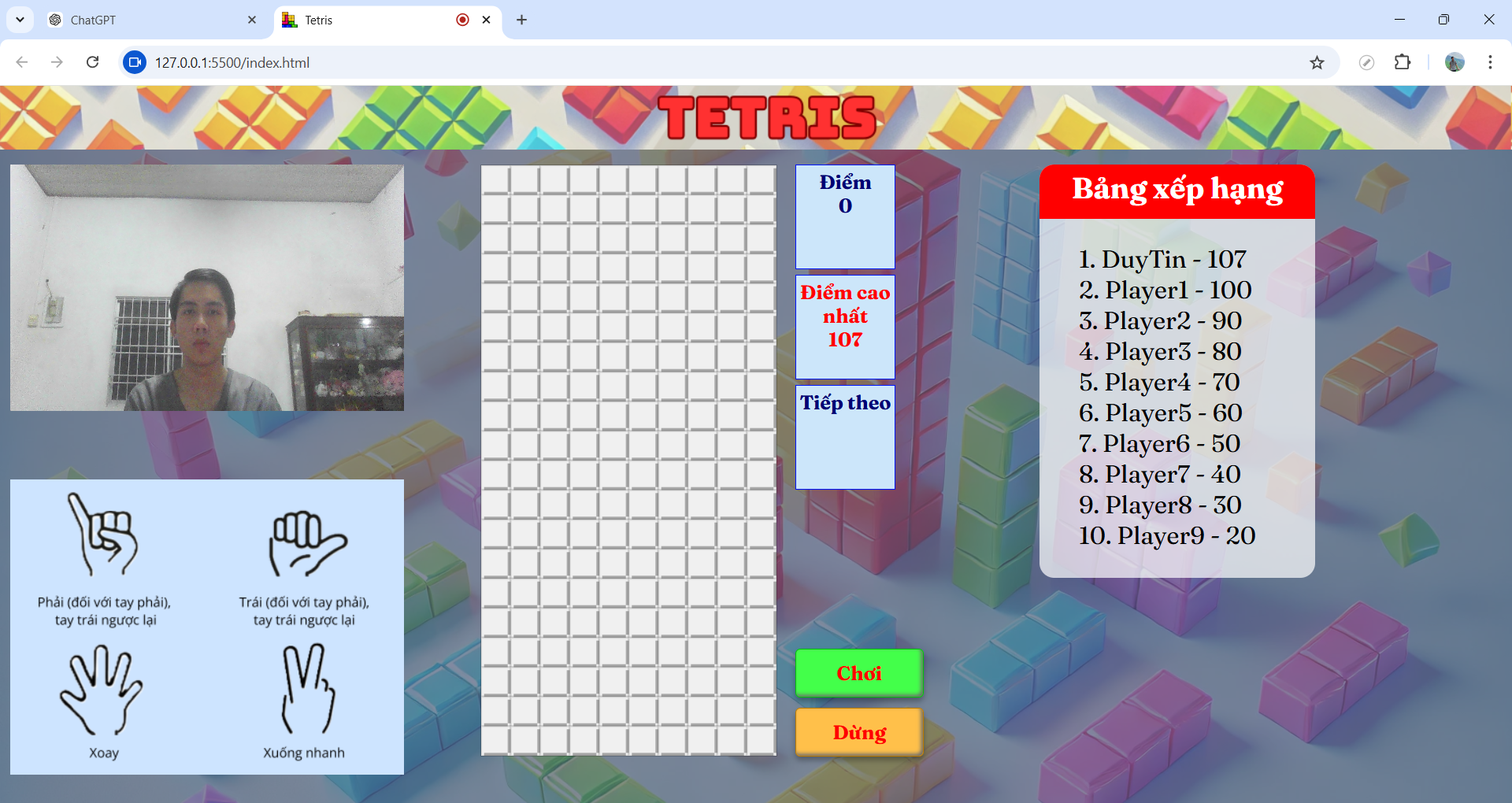
**Các giao diện và chức năng chính**

**Giao diện đăng nhập:**

****

Người chơi sẽ nhập tên đăng nhập để truy cập vào trò chơi. Khi người chơi nhấn nút “Bắt đầu chơi”, hệ thống sẽ kiểm tra nếu tên đăng nhập rỗng thì sẽ hiện hộp thoại thông báo.

**Màn hình chính**:



Hiển thị giao diện chơi game bao gồm: Khung camera, hướng dẫn chơi, giao diện trò chơi và bảng xếp hạng.

**Chế độ chơi xếp gạch**:

Người chơi nhấn nút “Chơi” để bắt đầu trò chơi. Khi nút “Chơi” được nhấn, gạch sẽ xuất hiện trên lưới và gạch tiếp theo sẽ xuất hiện ở ô “Tiếp theo”. Cùng lúc đó, nhạc nền sẽ được bật.

Người chơi sẽ phải dùng cử chỉ tay để điều khiển các khối gạch: giơ ngón cái/ngón út của bàn tay trái để điều khiển gạch qua phải/trái, giơ ngón cái/ngón út của bàn tay phải để điều khiển gạch qua trái/phải, giơ ngón trỏ và ngón giữa để di chuyển gạch xuống nhanh và xòe bàn tay để xoay gạch. Nếu phát hiện nhiều hơn 2 bàn tay, một thông báo “Chỉ sử dụng 1 bàn tay” sẽ xuất hiện tại khung camera và mọi hành động điều khiển sẽ tạm dừng cho đến khi hệ thống chỉ phát hiện một bàn tay.

Khi muốn dừng trò chơi, người chơi nhấm nút “Dừng”, trò chơi và nhạc nền sẽ dừng. Để tiếp tục, người chơi nhấn vào nút “Tiếp tục”, trò chơi và nhạc nền sẽ phát tiếp tục. Ngoài ra, nếu người chơi muốn làm mới lại trò chơi thì nhấn vào nút “Chơi lại”, toàn bộ gạch trên bảng sẽ bị xóa và điểm hiện tại sẽ bằng 0.

Nếu người chơi hoàn thành số hàng bất kì, hàng hoàn thành sẽ có hiệu ứng tan biến từ trái qua và âm thanh xóa hàng sẽ phát. Cách tính điểm như sau: Nếu người chơi hoàn thành 1 hàng, người chơi được cộng 10 điểm vào điểm hiện tại; nếu hoàn thành 2 hàng, người chơi sẽ được cộng 25 điểm, bao gồm điểm hoàn thành hàng là 20 điểm và điểm thưởng là 5 điểm, khi đó sẽ xuất hiện thông báo điểm thưởng trong giao diện chơi; tương tự với số hàng hoàn thành là 3, người chơi sẽ được cộng 37 điểm (trong đó có 7 điểm thưởng); số hàng hoàn thành là 4, người chơi sẽ được cộng 50 điểm (trong đó có 10 điểm thưởng). Do số hàng hoàn thành 1 lần xếp gạch tối đa là 4 (số ô gạch có chiều dài lớn nhất của một khối gạch) nên điểm thưởng tối đa có thể được cộng là 10 điểm. Khi người chơi có điểm hiện tại lớn hơn điểm cao nhất, điểm cao nhất sẽ được cập nhật bằng điểm hiện tại ngay lập tức.

Đối với bảng xếp hạng, tên người chơi sẽ được cập nhật lên bảng xếp hạng khi đăng nhập vào trò chơi với vị trí phụ thuộc vào điểm cao nhất đã chơi trước đó hoặc điểm bằng 0 nếu lần đầu chơi. Trong quá trình chơi, nếu người chơi đạt được điểm số cao hơn so với bất kì người chơi nào có trên bảng xếp hạng, người chơi sẽ được thăng hạng.

Trò chơi sẽ kết thúc khi người chơi để gạch chạm dòng trên cùng của lưới. Khi đó, một hộp thoại thông báo “Game over” sẽ xuất hiện cùng với âm thanh. Khi người chơi nhấn vào nút “OK”, toàn bộ gạch trên bảng sẽ xóa, điểm hiện tại sẽ được cập nhật về 0, người chơi nhấn vào nút chơi để bắt đầu lại từ đầu.

**Những điểm nổi bật**

**Ứng dụng MediaPipe**: Hệ thống tận dụng khả năng của MediaPipe để nhận diện và theo dõi cử chỉ tay trong thời gian thực, đảm bảo tính ổn định và hiệu quả.

**Công nghệ hiện đại**: Kết hợp HTML, CSS, và JavaScript để phát triển game, mang lại sự thuận tiện và khả năng mở rộng trong tương lai.

**Trải nghiệm tương tác sáng tạo**: Tận dụng công nghệ điều khiển không chạm, mở ra cơ hội cho những ứng dụng game giải trí sáng tạo và mới mẻ hơn.

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận: Trình bày những kết quả đạt được, những đóng góp mới và những đề xuất mới. Phần kết luận cần ngắn gọn, không có lời bàn và bình luận thêm.

Đồ án đã đạt được những kết quả sau:

Xây dựng thành công game xếp gạch trên nền tảng web với khả năng điều khiển bằng cử chỉ tay thay thế cho phím bấm truyền thống.

Tích hợp công nghệ MediaPipe để nhận diện cử chỉ tay trong thời gian thực, đảm bảo tính chính xác và hiệu quả.

Những đóng góp mới của đồ án:

Cung cấp một phương pháp điều khiển game sáng tạo, không cần thiết bị ngoại vi như chuột hay bàn phím.

Ứng dụng MediaPipe hiệu quả trong việc xử lý và theo dõi cử chỉ tay trực tiếp trên trình duyệt web.

**Hướng phát triển**

**Tăng tính khả dụng**:

Tối ưu hóa để trang web hỗ trợ reponsive, chạy tốt trên nhiều thiết bị khác nhau như smartphone, máy tính bảng.

Chuyển từ chạy local sang triển khai trực tuyến để người dùng dễ dàng truy cập qua internet.

**Cải thiện trải nghiệm người dùng**:

Bổ sung các chế độ chơi khác nhau, tạo sự đa dạng và hấp dẫn.

Nâng cấp giao diện trực quan và thân thiện hơn.

**Nâng cao hiệu năng**:

Cải thiện khả năng nhận diện cử chỉ trong môi trường ánh sáng yếu hoặc khi người chơi ở khoảng cách xa webcam.

Tích hợp thêm các thuật toán tối ưu để tăng tốc độ xử lý cử chỉ.

Hỗ trợ đa người chơi trực tuyến.

# DANH MỤC TÀI LIỆU THAM KHẢO

# PHỤ LỤC