**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**BÁO CÁO PROJECT 3**

**Tìm hiểu và Thiết kế giao diện Website sửa lỗi chính tả**

**TRẦN HỒNG QUÝ**

quy.th200515@sis.hust.edu.vn

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | PGS.TS. Lê Thanh Hương |
| **Khoa:** | Khoa học máy tính |
| **Trường:** | Công nghệ Thông tin và Truyền thông |
| **HÀ NỘI, 01/2024** | |

**LỜI CẢM ƠN**

Đầu tiên, em xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với cô đã hướng dẫn và tạo điều kiện cho em hoàn thành tốt bài báo cáo này.

Do thời gian và kiến thức còn hạn chế, bài báo cáo khó tránh khỏi những thiếu sót, rất mong cô sẽ đóng góp ý kiến để bài báo cáo của em được hoàn thiện hơn. Em xin chân thành cảm ơn!

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 5](#_Toc157535722)

[1.1 Đặt vấn đề 5](#_Toc157535723)

[1.2 Mục tiêu và định hướng giải pháp 5](#_Toc157535724)

[1.2.1 Mục tiêu 5](#_Toc157535725)

[1.2.2 Định hướng giải pháp 5](#_Toc157535726)

[1.3 Đóng góp của project môn học 6](#_Toc157535727)

[1.4 Bố cục project môn học 6](#_Toc157535728)

[CHƯƠNG 2. NỀN TẢNG LÝ THUYẾT 7](#_Toc157535729)

[2.1 Tổng quan về định dạng file .pdf 7](#_Toc157535730)

[2.1.1 Giới thiệu 7](#_Toc157535731)

[2.1.2 Phân loại 8](#_Toc157535732)

[2.1.3 Giải pháp triển khai 10](#_Toc157535733)

[2.2 Tìm hiểu một số tool sửa lỗi chính tả cho văn bản 11](#_Toc157535734)

[2.2.1 Grammarly 11](#_Toc157535735)

[2.2.2 Language Linter 12](#_Toc157535736)

[2.2.3 Giải pháp triển khai 12](#_Toc157535737)

[CHƯƠNG 3. TRIỂN KHAI MÃ NGUỒN 14](#_Toc157535738)

[3.1 Lựa chọn ngôn ngữ và khởi tạo dự án 14](#_Toc157535739)

[3.2 Xây dựng mã nguồn upload và convert File 15](#_Toc157535740)

[3.2.1 Hàm handleFileChange 15](#_Toc157535741)

[3.2.2 Hàm handleConvert 16](#_Toc157535742)

[3.3 Tích hợp Grammarly Plugin 18](#_Toc157535743)

[3.4 Tích hợp Language Linter 18](#_Toc157535744)

[3.5 Một số hàm xử lý khác và thiết kế giao diện 19](#_Toc157535745)

[3.5.1 Hàm xử lý hiển thị văn bản ra màn hình 19](#_Toc157535746)

[3.5.2 Hàm xử lý phân trang với Language Linter 20](#_Toc157535747)

[3.5.2 Hàm xử lý tải xuống văn bản sau khi kiểm tra lỗi với định dạng .txt 20](#_Toc157535748)

[3.5.4 Thiết kế giao diện 22](#_Toc157535749)

[CHƯƠNG 4. MỘT SỐ GIAO DIỆN TRỰC QUAN CỦA SẢN PHẨM 23](#_Toc157535750)

[CHƯƠNG 5. TỔNG HỢP ƯU, NHƯỢC ĐIỂM CỦA SẢN PHẨM VÀ HƯỚNG GIẢI PHÁP TRONG TƯƠNG LAI 26](#_Toc157535751)

[5.1 Tổng hợp ưu, nhươc điểm của sản phẩm 26](#_Toc157535752)

[5.2 Hướng giải pháp 26](#_Toc157535753)

[CHƯƠNG 6. KẾT LUẬN 27](#_Toc157535754)

**HÌNH VẼ**

[Hình 1. Sự tiện lợi của PDF 7](#_Toc157535521)

[Hình 2. Tệp được tạo bằng máy in ảo trên Word 9](#_Toc157535522)

[Hình 3. Tệp PDF chỉ có thể copy nội dung dưới dạng hình ảnh 10](#_Toc157535523)

[Hình 4. Sơ đồ mô tả quá trình upload và convert 10](#_Toc157535524)

[Hình 5. Khởi tạo dự án 14](#_Toc157535525)

[Hình 6. Hằng số và giá trị lưu trữ đầu ra 14](#_Toc157535526)

[Hình 7. useState() 14](#_Toc157535527)

[Hình 8. file package.json 15](#_Toc157535528)

[Hình 9. Xử lý bước 2 15](#_Toc157535529)

[Hình 10. Xử lý chuyển đổi định dạng 16](#_Toc157535530)

[Hình 11. hàm xử lý file PDF 17](#_Toc157535531)

[Hình 12. Xử lý chia nhỏ văn bản 17](#_Toc157535532)

[Hình 13. Triển khai Grammarly Plugin 18](#_Toc157535533)

[Hình 14. Triển khai Language Linter 19](#_Toc157535534)

[Hình 15. Hiển thị văn bản 19](#_Toc157535535)

[Hình 16. Phân trang 20](#_Toc157535536)

[Hình 17. Download file văn bản Grammarly 21](#_Toc157535537)

[Hình 18. Download file văn bản Language Linter 21](#_Toc157535538)

[Hình 19. Sử dụng CodeMirror 22](#_Toc157535539)

[Hình 20. CSS cho các thẻ bên trong file JS 22](#_Toc157535540)

[Hình 21. Màn hình ban đầu 23](#_Toc157535541)

[Hình 22. Convert thành công 23](#_Toc157535542)

[Hình 23. Convert thất bại 24](#_Toc157535543)

[Hình 24. Hiển thị văn bản 24](#_Toc157535544)

[Hình 25. Văn bản Grammarly sửa lỗi 25](#_Toc157535545)

[Hình 26. Văn bản Language Linter sửa lỗi 25](#_Toc157535546)

# GIỚI THIỆU ĐỀ TÀI

## Đặt vấn đề

Trong môi trường số ngày nay, việc chia sẻ thông tin và tài liệu trực tuyến đã trở thành một phần quan trọng của nhiều hoạt động công việc và học tập. Tuy nhiên, việc sử dụng các website để tải lên và chia sẻ một tệp có định dạng .pdf hoặc .txt đôi khi gặp phải nhiều thách thức liên quan đến định dạng, cấu trúc tệp. Một số loại tệp như vậy chỉ cho phép chúng ta xem ở chế độ “Read Only”, trong khi nhu cầu chủ yếu khi tải một tệp nào đó của con người là có thể sử dụng để chỉnh sửa hoặc thêm mới những yếu tố riêng biệt của bản thân mình vào đó.

Một vấn đề khác cũng được chúng ta đề cập nhiều khi làm việc các tệp thư mục văn bản đó là kiểm tra các lỗi chính tả. Khi làm việc đôi khi chúng ta vô ý hoặc bấm nhầm một phím hay kí tự nào đó khiến văn bản xảy ra lỗi cú pháp, và ta không thể kiểm soát được điều này. Có rất nhiều tool hoặc trợ giúp được các nhà phát triển đề xuất sử dụng làm giảm tải khả năng xảy ra lỗi, thông báo kịp thời cho người dùng để nắm bắt và chỉnh sửa kịp thời. Tuy nhiên điều đó chỉ giúp chúng ta hiệu quả trong một văn bản có độ dài vừa phải. Với văn bản có độ dài tương đối lớn, việc phải lướt đi lướt lại soát lỗi sẽ gây khó chịu và người dùng cảm thấy không được hài long với điều này.

Đã có nhiều giải pháp xây dựng trang web nhưng thường chỉ dừng lại ở việc làm được một trong hai công việc trên, nghĩa là hoặc chỉ chuyển đổi tệp tải lên và cho phép tải xuống tệp sau khi chuyển đổi, hoặc chỉ là thực hiện công việc nhập văn bản và sẽ tự động chỉnh sửa các lỗi liên quan đến chính tả trong văn bản.

Bài báo cáo project này của em, xuất phát từ những câu hỏi và băn khoăn trên, được sự đề xuất và hướng dẫn của cô, em xin được đặt tên cho báo cáo với tiêu đề **“Tìm hiểu và thiết kế giao diện website sửa lỗi chính tả.”**

## Mục tiêu và định hướng giải pháp

### Mục tiêu

Mục tiêu của project này là em sẽ tìm hiểu và thiết kế giao diện website sửa lỗi chính tả.

### Định hướng giải pháp

Với một lộ trình rõ ràng, em sẽ đi tìm hiểu cấu trúc các tệp có định dạng .pdf, từ đó làm cơ sở để xây dựng phương pháp và mã nguồn chuyển đổi định dạng loại tệp này sang loại định dạng khác có thể dễ dàng hơn cho việc chỉnh sửa và kiểm tra lỗi chính tả. Bên cạnh đó, em cũng sẽ tìm hiểu về việc tích hợp các tool kiểm tra lỗi vào trang web của mình, phục vụ công việc tự động soát lỗi và đề xuất gợi ý sửa lỗi. Bước cuối cùng, quan trọng nhất là thiết kế một giao diện trực quan dễ nhìn, dễ sử dụng dành cho người dùng có một trải nghiệm tốt nhất.

## Đóng góp của project môn học

Báo cáo sẽ là những kết quả đạt được sau quá trình định hướng giải pháp và triển khai cái công việc đề ra. Một số thông tin, tri thức và các lưu ý cần biết được ghi chú chi tiết để mô tả quá trình làm việc và cách thức hoạt động của website sau khi được hoàn thiện. Bên cạnh đó, cũng là những sự thiếu sót, những điều mà em chưa làm được để lại một số câu hỏi mở trong tương lai nhằm cải tiến và thêm mới các tiện ích tốt hơn cho website này.

## Bố cục project môn học

Bài báo sẽ bao gồm các nội dung chính như sau:

* Chương 2 sẽ tổng hợp cơ sở lý thuyết về định dạng tệp .pdf và các tool sửa lỗi chính tả trong quá trình làm Project 3.
* Chương 3 sẽ nói về quá trình triển khai mã nguồn của các công việc chuyển đổi định dạng tệp và tích hợp tool vào trang web.
* Chương 4 sẽ trình bày một số giao diện trực quan khi tương tác với trang web.
* Sang đến chương 5, em sẽ trình bày một số ưu nhược điểm của project này và các định hướng trong tương lai để cải thiện sản phẩm.
* Chương 6 sẽ kết lại báo cáo.

# NỀN TẢNG LÝ THUYẾT

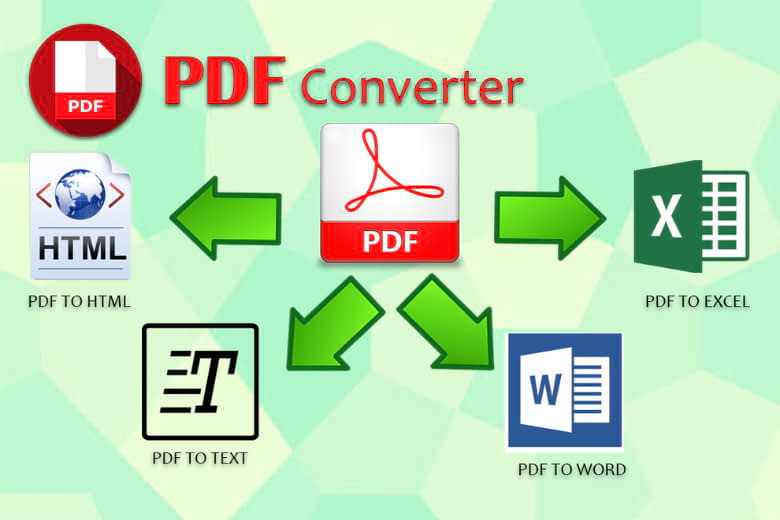
## Tổng quan về định dạng file .pdf

### Giới thiệu

Định dạng file PDF là một trong những định dạng file khá là phổ biến thường được sử dụng nhiều trong học tập và làm việc.

PDF là cụm từ viết tắt của Portable Document Format, có nghĩa là định dạng tài liệu di động, được phát hành bởi Adobe. Phần mở rộng của tệp PDF sẽ có đuôi **“.pdf”**, tệp sẽ có thể chứa được những văn bản, hình ảnh, nút động tương tác, cac đường link liên kết, video, phông chữ dạng nhúng,… Định dạng này thường được sử dụng rộng rãi với các mẫu hướng dẫn, ebook, tờ rơi, đơn từ, quảng cáo và tài liệu thiết kế,…

Ngoài ra, một số trang web còn có thể lưu thành dạng tệp PDF dễ dàng để có thể lưu lại đầy đủ thông tin, tham khảo lại. File PDF không dựa trên phần mềm máy, hệ điều hành hay bất cứ phần cứng nào cụ thể; vì vậy có thể giữ được nguyên định dạng giống nhau dù là mở trên bất kỳ thiết bị nào.



Hình 1. Sự tiện lợi của PDF

Một số thông tin trên đã thể hiện được phần nào sự tiện lợi của file PDF. Dưới đây là tóm gọn những ưu điểm nổi bật nhất của định dạng này:

* Khả năng hiển thị cùng một nột dung và bố cục khi xem trên bất kỳ hệ điều hành, thiết bị., phần mềm hay ứng dụng nào.
* Cho phép người dùng tích hợp nhiều nội dung khác nhau như dạng văn bản, hình ảnh, đồ hoạ dạng vector, GIF, video, tệp âm thanh, mô hình, siêu liên kết,.. Tất cả đều có thể đặt trên cùng một tệp PDF, có thể sắp xếp được dưới dạng báo cáo, trình chiếu, danh mục.
* Thuật toán nén ảnh sẽ giúp cho tệp PDF sử dụng dung lượng lưu trữ cực thấp, giảm đi áp lực lưu trữ trên ổ cứng thiết bị.
* Tính năng bảo mật dữ liệu cực kỳ cao, cho phép người dùng có thể cài đặt chế độ cấm in, chỉnh sửa hoặc chữ ký điện tử để có thể xác thực tài liệu.

Bên cạnh đó, không thể tránh khỏi là những nhược điểm của “sự quá tiện lợi” như vậy:

* Chi phí chỉnh sửa file PDF khá cao (hầu như ta đều phải trả phí để có thể sử dụng tính năng chỉnh sửa tệp PDF).
* File sẽ được coi như là một hình ảnh nên sẽ khó có thể thao tác với văn bản bên trong file.
* Nếu không có phần mềm chuyên dụng thì bạn không thể đọc hoặc là chỉnh sửa được file PDF.

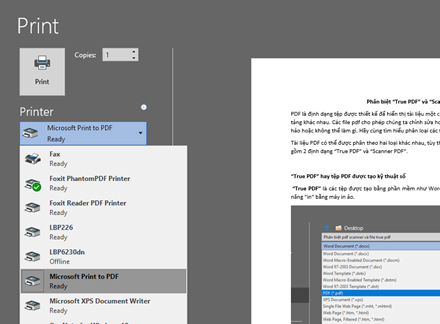
### Phân loại

File PDF rất tiện dụng, vì thế ta có thể phân loại theo nhiều cách khác nhau, dựa trên các yếu tốc khác nhau. Dựa trên cách dữ liệu bên trong tệp được tổ chức và sử dụng, ta có thể phân loại thành một số đối tượng sau:

* **PDF Tiêu Chuẩn (PDF/A):** thiết kế dành cho mong muốn bảo tồn nội dung lâu dài và đảm bảo không bị mất thông tin khi mở và hiển thị.
* **PDF In Ấn (PDF/X):** thiết kế đặc biệt để in ấn.
* **PDF Di Động (PDF/E):** được sử dụng chủ yếu cho các tài liệu kỹ thuật, các thông tin, hình vẽ 3D được chia sẻ một cách hiệu quả.
* **PDF Bảo Mật (Password-Protected PDF**): được sử dụng với mục đích bảo vệ thông tin quan trọng, nhạy cảm mà chỉ một số người dùng mới có thể truy cập vào được thông qua mật khẩu được cung cấp.
* **PDF Tích Hợp Phương Tiện (PDF with Embedded Media):** hiểu đơn giản là một tệp chứa các dữ liệu đa phương tiện như video, hình ảnh tĩnh/động,…

Một cách phân loại khác phổ biến dựa trên cách thức mà tệp được bắt nguồn, bao gồm hai định dạng **“True PDF”** và **“Scanner PDF”**.

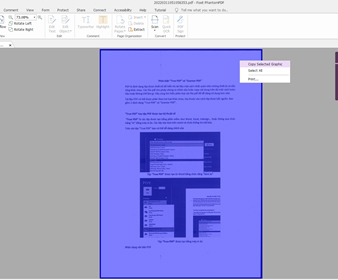
* **“True PDF”** là các tệp PDF được tạo kỹ thuật số (bằng các phần mềm như Word, Excel,…) hoặc thông qua chức năng “in” bằng máy in ảo. Các tệp này dựa trên cấu trúc dữ liệu vector nên chất lượng hiển thị khi chuyển sang .pdf không bị ảnh hưởng dù bạn có phóng to hay thu nhỏ.



Hình 2. Tệp được tạo bằng máy in ảo trên Word

* **“Scanner PDF”** là các tệp được tạo bởi các hình ảnh (thông qua việc chụp ảnh, và sử dụng các loại máy quét chuyên dụng hoặc máy quét văn phòng. Chất lượng hiển thị các tệp loại này phụ thuộc vào loại máy quét và tuỳ chọn khi quét.

Điểm tốt hơn của **“True PDF”** là có thể có khả năng cho phép chúng ta chỉnh sửa, còn hầu như với **“Scanner PDF”** chỉ cho phép ta copy toàn bộ trang dưới dạng hình ảnh và không thể chỉnh sửa nó.



Hình 3. Tệp PDF chỉ có thể copy nội dung dưới dạng hình ảnh

Hướng giải pháp của em sẽ dựa trên cách phân loại thành hai định dạng phía trên.

### Giải pháp triển khai

Với hai kiểu định dạng như trên, và yêu cầu của project này, em sẽ xử lý và làm việc với các file có định dạng “True PDF”. Ta sẽ xây dựng đầu vào và đầu ra cho bài toán: “Upload file và chuyển đổi định dạng file về .txt để có thể thực hiện chỉnh sửa”. Nói qua một chút, ta chọn chuyển về **.txt** vì loại định dạng file này rất thuận tiện cho việc xử lý văn bản thuần tuý và lọc bỏ được những yếu tố đa phương tiện khác không cần thiết cho công việc sửa lỗi chính tả phía sau.

A black background with white rectangles

Description automatically generated

Hình 4. Sơ đồ mô tả quá trình upload và convert

Yêu cầu cho đầu vào của bài toán này sẽ chấp nhận những file có định dạng “pdf/txt/doc/docx” có kích thước không quá lớn (tối da 2MB dữ liệu). Yêu cầu ngôn ngữ được đặt ra cho các tệp này là chấp nhận phần text Tiếng Anh (Anh-Anh, Anh-Mỹ, Anh-Canada,…). Sau khi upload lên, sẽ xây dựng một hàm kiểm tra lại các yếu tố kỹ thuật trên để tránh bỏ sót trường hợp upload sai định dạng hoặc kích thước file. Sau khi qua được bước này, ta sẽ phân loại dự trên kiểu của file. Với các định dạng file thông thường, ta có thể xử lý đọc trực tiếp được, còn với trường hợp chính là file .pdf, ta sẽ sử dụng một thư viện hỗ trợ bên thứ ba để xử lý nội dung. Kết quả của quá trình convert này là một chuỗi ký tự. Tuy nhiên, với một số hạn chế trong việc tìm hiểu các tool sửa lỗi chính tả (được đề cập ở phía sau), ta cần xử lý chuỗi ký tự này bằng việc chia nhỏ thành nhiều chuỗi ngắn hơn và lưu trữ vào mảng dữ liệu. Việc xử lý này vẫn sẽ đảm bảo cấu trúc các từ, không gây mất mát thông tin và đảm bảo đúng ngữ pháp và ý nghĩa của văn bản ban đầu. Đầu ra sau cùng là một mảng ký tự.

## Tìm hiểu một số tool sửa lỗi chính tả cho văn bản

Việc phát triển các công cụ sửa lỗi chính tả đã trở thành một phần quan trọng của quá trình viết và biên soạn văn bản. Các công cụ này bên cạnh việc giúp người viết loại bỏ những sai sót nhỏ về chính tả mà còn hỗ trợ rất nhiều vấn đề cấu trúc ngôn ngữ và phong cách văn bản. Trong phạm vi sản phẩm môn học, em đã có những lựa chọn và tìm hiểu về hai loại tool phổ biến, đó là “**Grammarly**” và “**Language Linter**”.

### Grammarly

**Grammarly** là một ứng dụng và công cụ kiểm tra ngữ pháp trực tuyến nổi tiếng, được thiết kế để hỗ trợ người dùng cải thiện chất lượng văn bản và viết tiếng Anh một cách chính xác và hiệu quả. Được thành lập vào năm 2009, **Grammarly** nhanh chóng trở thành một trong những công cụ phổ biến nhất trong lĩnh vực kiểm tra chính tả, ngữ pháp và cấu trúc câu.

Với **Grammarly**, ta có thể sử dụng rất nhiều chức năng tối ưu phục vụ biên soạn văn bản. Có thể kể đến một số tính năng dưới đây:

* **Kiểm Tra Chính Tả và Ngữ Pháp**: Đây là một công cụ mạnh mẽ giúp tự động kiểm tra chính tả và ngữ pháp trong văn bản. Nó có khả năng nhận diện và sửa lỗi cú pháp, sử dụng từ sai, và nhiều lỗi ngữ pháp khác.
* **Kiểm Tra Cấu Trúc Câu**: Công cụ này giúp người dùng xây dựng câu văn rõ ràng và logic.
* **Phân Tích Phong Cách Văn Bản**: **Grammarly** cung cấp đánh giá về phong cách văn bản, bao gồm sự đa dạng từ vựng, câu trạng ngữ và sự lặp lại từ.
* **Dịch Văn Bản**: Một tính năng dịch văn bản giữa nhiều ngôn ngữ, giúp người dùng làm việc với nội dung đa ngôn ngữ một cách tiện lợi, hiệu quả.
* **Mở Rộng Trình Duyệt và Ứng Dụng**: **Grammarly** có các tiện ích mở rộng cho trình duyệt web phổ biến như Chrome, Firefox, Safari, cũng như ứng dụng di động cho iOS và Android.
* **Kiểm Tra Email và Social Media**: Ta có thể tích hợp **Grammarly** vào các ứng dụng email và nền tảng mạng xã hội, giúp kiểm tra ngôn ngữ trong các bức thư, bài đăng, hoặc bất kỳ nơi nào bạn viết trực tuyến.

### Language Linter

Không kém cạnh so với **Grammarly**, **Language Linter** cũng là một công cụ kiểm tra ngôn ngữ tối ưu. Ta có thể sử dụng tool này để đảm bảo mã nguồn hoặc văn bản tuân thủ các quy tắc ngôn ngữ nhất định. **Linter** chủ yếu được tích hợp vào quy trình phát triển phần mềm hoặc biên soạn văn bản để phát hiện và báo cáo lỗi ngôn ngữ, đảm bảo chất lượng mã nguồn hoặc văn bản là đồng nhất và dễ hiểu.

Một số tính năng hữu ích của **Language Linter**:

* **Kiểm Tra Ngôn Ngữ và Quy Tắc: Language Linter** kiểm tra ngôn ngữ theo các quy tắc và hướng dẫn cụ thể, đảm bảo rằng mã nguồn hoặc văn bản tuân thủ các tiêu chuẩn và quy định.
* **Phát Hiện Lỗi và Cảnh Báo**: Cung cấp thông báo về các lỗi cú pháp, ngữ pháp hoặc vi phạm quy tắc ngôn ngữ khác, giúp người viết hoặc lập trình viên sửa lỗi trước khi triển khai hoặc xuất bản.
* **Tùy Chỉnh Quy Tắc**: Cho phép người dùng tùy chỉnh các quy tắc kiểm tra tùy thuộc vào yêu cầu cụ thể của dự án hoặc tiêu chí ngôn ngữ.
* **Tích Hợp Tích cực vào Quy Trình Phát Triển:** Thường được tích hợp vào các công cụ phát triển tích hợp (IDE) hoặc trình soạn thảo văn bản để kiểm tra ngôn ngữ một cách liên tục trong quá trình viết.
* **Hỗ Trợ Đa Ngôn Ngữ:** Nhiều **Language Linter** hỗ trợ kiểm tra ngôn ngữ cho nhiều ngôn ngữ lập trình hoặc ngôn ngữ biên soạn văn bản khác nhau, ví dụ ESLint cho Javacript, Pylint cho Python, Stylelint cho CSS,…

### Giải pháp triển khai

Cả hai tool này đều hỗ trợ cho việc tích hợp vào các dự án dành cho Developer. Với Grammarly, mọi hướng dẫn về việc tích hợp đều có thể tìm thấy ngay tại trang chủ của hệ thống.

Với Language Linter cũng vậy, ta có thể tìm kiếm được mã nguồn của nó trên Github, từ đó có thể xem được hướng dẫn về việc tích hợp và video demo mẫu giúp ta có cái nhìn trực quan hơn và triển khai dễ dàng hơn.

# TRIỂN KHAI MÃ NGUỒN

Sau khi đã tìm hiểu những thông tin cần thiết, ở chương 3 này, chúng ta bắt đầu với công việc xây dựng mã nguồn cho dự án.

## Lựa chọn ngôn ngữ và khởi tạo dự án

Sản phẩm này được em xây dựng với ngôn ngữ JavaScript, thư viện hỗ trợ ReactJS và các thư viện bên thứ ba của nhà phát hành npm.

Bước đầu tiên, khởi tạo dự án:



Hình 5. Khởi tạo dự án

Sau đó, ta tiến hành xây dựng khởi tạo các giá trị lưu trữ:



Hình 6. Hằng số và giá trị lưu trữ đầu ra

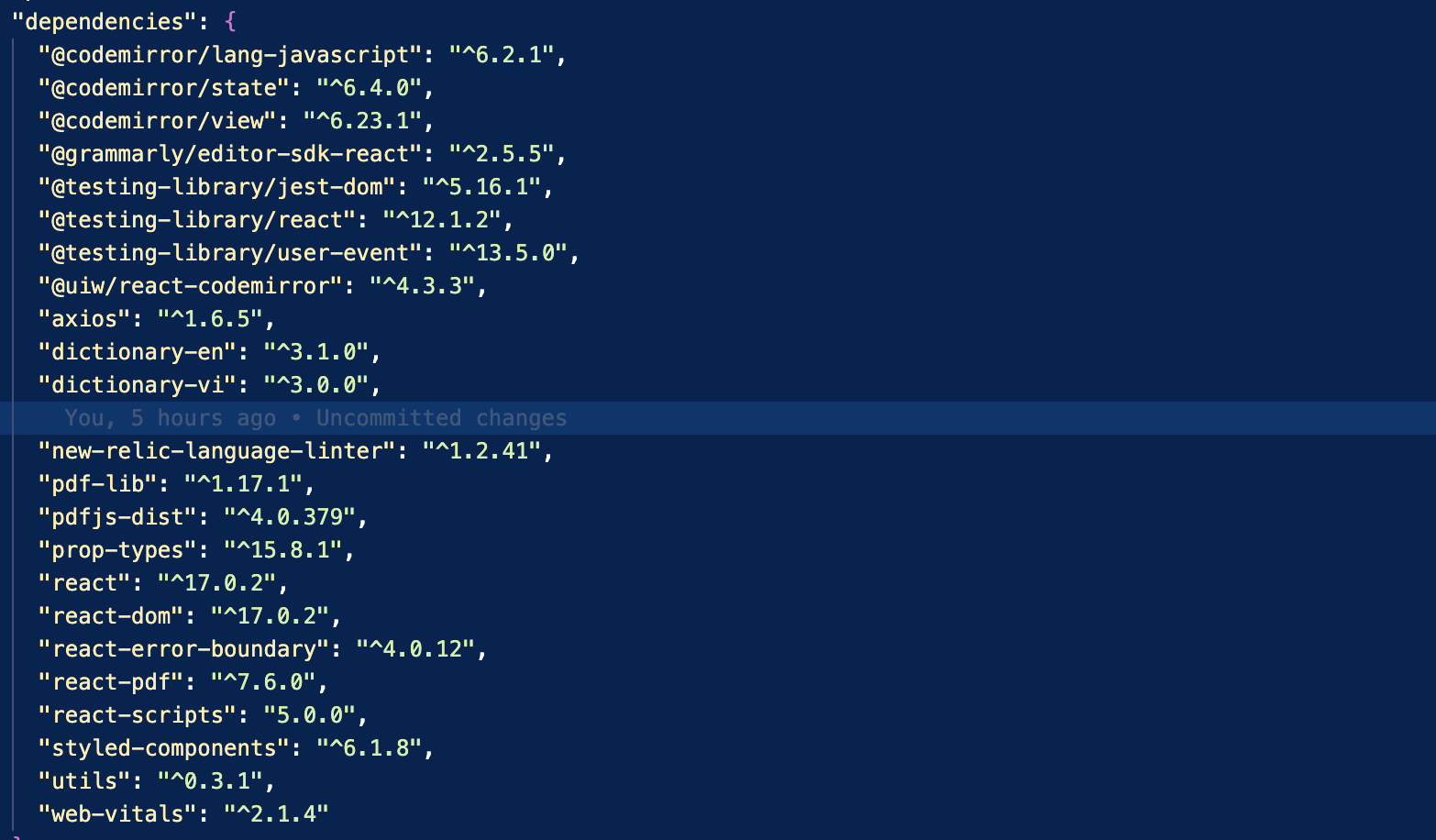
Sử dụng Hook của ReactJS: “useState” để lưu trữ trạng thái và tiện cho việc cập nhật trong quá trình update và render ra trình duyệt.

A blue background with white text

Description automatically generated

Hình 7. useState()

Bên cạnh đó, ta cần install các *dependencies* cần thiết cho dự án:

**

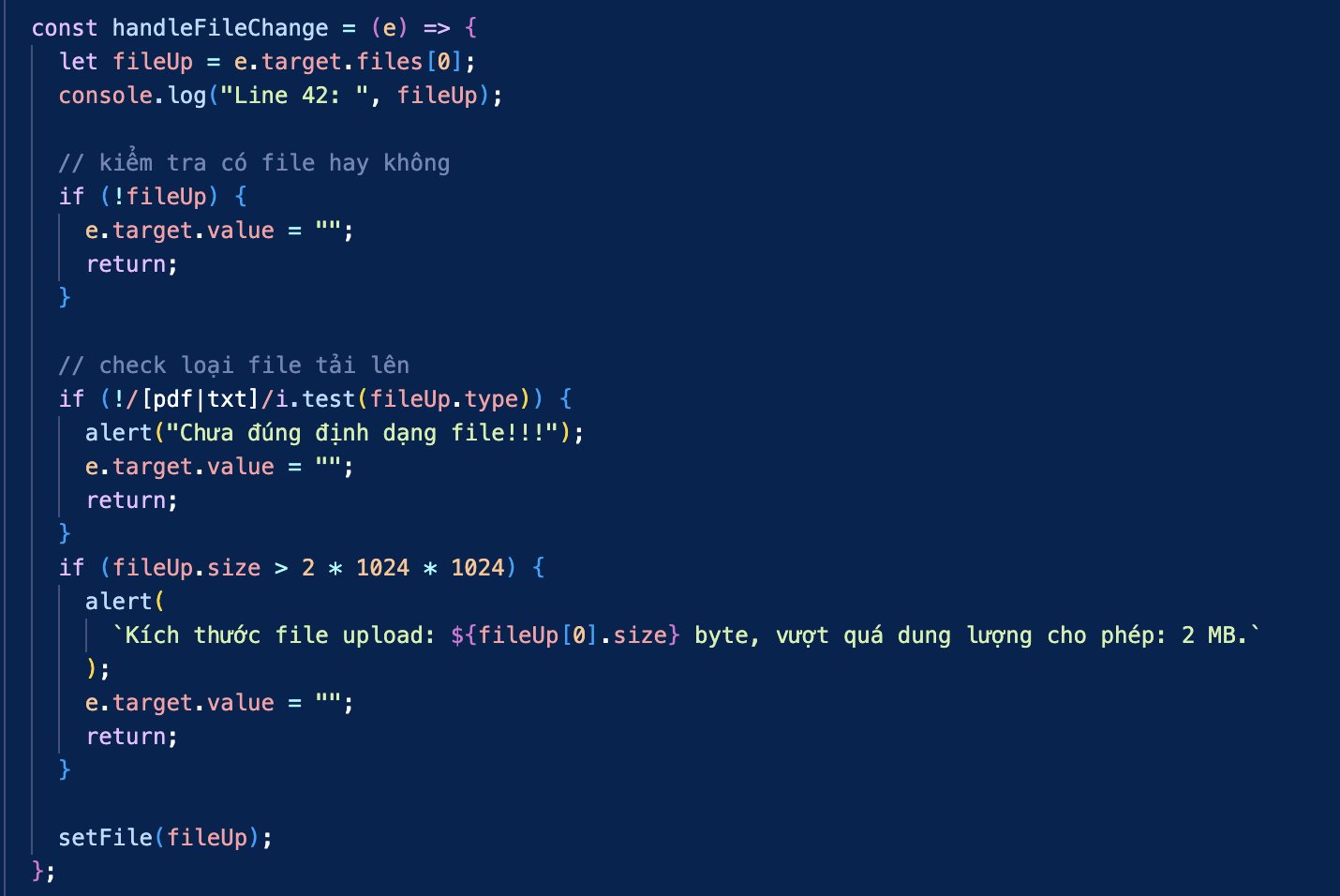
Hình 8. file package.json

Đề mục tiếp theo là xây dựng các hàm phục vụ cho công việc đầu tiên: “Xử lý File”.

## Xây dựng mã nguồn upload và convert File

Dựa vào sơ đồ mô tả quá trình ở Chương 2, ta triển khai một số hàm xử lý cho từng bước như sau:

### Hàm handleFileChange



Hình 9. Xử lý bước 2

Công việc của hàm này là xử lý đầu vào khi upload file lên (bước thứ hai trong sơ đồ). Sử dụng câu lệnh if…else để validate file qua các yêu cầu, nếu thoả mãn ta sẽ lưu lại file vào biến trạng thái file thông qua câu lệnh setFile().

### Hàm handleConvert

A screen shot of a computer

Description automatically generated

Hình 10. Xử lý chuyển đổi định dạng

Việc đầu tiên là kiểm tra lại xem giá trị biến file đã được cập nhật chưa (có giá trị sẽ khác **null**). Tiếp đó, ta phân loại file theo thuộc tính “**type**”. Ở đây, ta có hai hướng phải xử lý:

* Nếu định dạng file là “**text/plain**” (không phải file .pdf), ta sẽ xử lý đọc file trực tiếp và lưu lại giá trị chuỗi thu được vào mảng lưu trữ **alltext** trong bước khởi tạo dự án ban đầu.
* Trường hợp còn lại, ta sẽ cần đến một thư viện bên thứ ba đó là **“react-pdf”**, triển khai thông qua một hàm con là **extractText(url)**.

**A computer screen shot of text

Description automatically generated**

Hình 11. hàm xử lý file PDF

Từ file được gửi lên, ta tạo ra một đường dẫn url, rồi xử lý thư viện “react-pdf”, để lấy ra được nội dung của file. Điểm đặc biệt của thư viện này là nó chỉ lấy ra tất cả nội dung văn bản bên trong file PDF, sau đó được ta xử lý nối lại thành một chuỗi kí tự.

Một điều chắc chắn không thể bỏ qua làm hàm **splitTextPreservingWords(text, maxLength)** để xử lý chia nhỏ chuỗi kí tự kia thành các chuỗi nhỏ và lưu lại thành một. mảng ký tự cho công việc phía sau.

A computer screen shot of a blue screen

Description automatically generated

Hình 12. Xử lý chia nhỏ văn bản

## Tích hợp Grammarly Plugin

Để tích hợp Grammarly vào dự án của mình, ta có thể thực hiện như sau:

B1. Import các thư viện cần thiết



B2. Triển khai mã nguồn theo hướng dẫn trên trang chủ

A screen shot of a computer screen

Description automatically generated

Hình 13. Triển khai Grammarly Plugin

Ta cần chú ý đến thuộc tính **clientId** khi tích hợp. Ta có thể sử dụng các giá trị miễn phí(theo giá trị mặc định khi tham khảo video hướng dẫn) hoặc có thể tự mình tạo ra bằng cách đăng ký tài khoản và truy cập trang web dành cho nhà phát triển. Với giá trị **clientId** do mình tạo ra, ta có thể kiểm soát được các từ mới thêm vô từ điển cá nhân (từ điển này có giá trị đặc trưng mỗi người, giúp người dùng có thể bỏ qua được một số điểm nhấn lỗi mà mình thấy không sai).

## Tích hợp Language Linter

Ta có thể tích hợp tool này thông qua mã nguồn mở trên Github, hoặc sử dụng thư viện được đề xuất “new-relic-language-linter”. Ở đây em sử dụng thư viện phía npm, và triển khai trong mã nguồn sản phẩm như sau:

B1: Import các thư viện cần thiết



B2: Triển khai mã nguồn



Hình 14. Triển khai Language Linter

Ở đây ta cần quan tâm ba props chính:

* **sampleText** (string): chuỗi văn bản cần kiểm tra, mỗi khi cập nhật chuỗi, phía. tool kiểm tra sẽ gửi phản hồi về.
* **setSampleTex**t (function): hàm cập nhật chuỗi văn bản cần kiểm tra.
* **updateTimer** (number): thời gian chờ đợi gửi phản hồi (tính theo mi-li-giây) tính từ lúc sampleText được cập nhật.

## Một số hàm xử lý khác và thiết kế giao diện

Bản thiết kế giao diện được em tham khảo và triển khai theo mẫu Figma của nhà phát triển **New Relic.** Video demo giao diện được đính kèm tại [đây](https://github.com/danielgolden/language-linter/tree/main).

### Hàm xử lý hiển thị văn bản ra màn hình

A computer code on a blue background

Description automatically generated

Hình 15. Hiển thị văn bản

### 3.5.2 Hàm xử lý phân trang với Language Linter

A computer screen shot of a program code

Description automatically generated

Hình 16. Phân trang

### Hàm xử lý tải xuống văn bản sau khi kiểm tra lỗi với định dạng .txt

A computer screen shot of a program code

Description automatically generated

Hình 17. Download file văn bản Grammarly

A computer screen shot of a program code

Description automatically generated

Hình 18. Download file văn bản Language Linter

### 3.5.4 Thiết kế giao diện

Ta sử dụng một số thư viện quen thuộc với ứng dụng ReactJS để thiết kế giao diện cho sản phẩm CSR (Client-Side Rendering), đó là “**styled-components**”, **“@uiw/react-codemirror**” và **“@codemirror/view**”. Dưới đây là mã nguồn được triển khai như sau:

**A computer screen shot of a blue screen

Description automatically generated**

Hình 19. Sử dụng CodeMirror

**A screen shot of a computer code

Description automatically generated**

Hình 20. CSS cho các thẻ bên trong file JS

# MỘT SỐ GIAO DIỆN TRỰC QUAN CỦA SẢN PHẨM

Sau khi hoàn tất quá trình viết mã nguồn và sửa lỗi, ta tiến hành chạy sản phẩm và thực hiện chụp lại một số giao diện chính của sản phẩm.

* Giao diện màn hình chính khi chạy dự án:

A screenshot of a computer

Description automatically generated

Hình 21. Màn hình ban đầu

* Giao diện khi convert file thành công:

A screenshot of a computer

Description automatically generated

Hình 22. Convert thành công

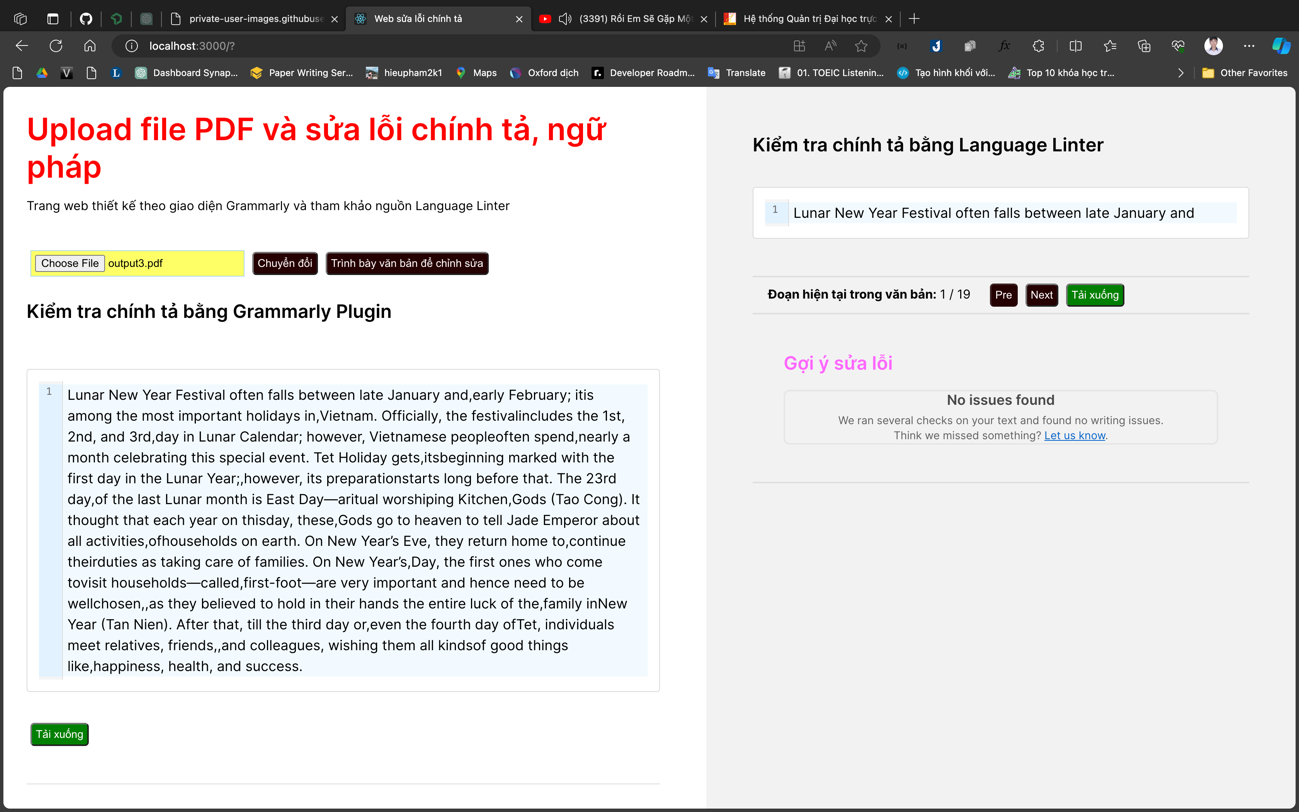
* Giao diện khi convert file xảy ra lỗi:

A screenshot of a computer

Description automatically generated

Hình 23. Convert thất bại

* Giao diện khi hiển thị văn bản để sửa lỗi:



Hình 24. Hiển thị văn bản

* Giao diện văn bản Grammarly đề xuất sửa lỗi:

A screenshot of a computer

Description automatically generated

Hình 25. Văn bản Grammarly sửa lỗi

* Giao diện văn bản Language Linter đề xuất sửa lỗi:

A screenshot of a computer

Description automatically generated

Hình 26. Văn bản Language Linter sửa lỗi

# TỔNG HỢP ƯU, NHƯỢC ĐIỂM CỦA SẢN PHẨM VÀ HƯỚNG GIẢI PHÁP TRONG TƯƠNG LAI

## Tổng hợp ưu, nhươc điểm của sản phẩm

Qua các chương đã trình bày phía trên, em đánh giá sản phẩm này có một số ưu điểm như sau:

* Sản phẩm đã đáp ứng được những yêu cầu cần thiết đặt ra khi bắt đầu môn học.
* Sản phẩm đã thể hiện được quá trình tìm hiểu và triển khai một cách đúng đắn.
* Giao diện và tương tác người dùng rất dễ nhìn, dễ nhận biết.

Bên cạnh đó, sản phẩm còn một số nhược điểm chưa thể hoàn thành:

* Sản phẩm mới chỉ giải quyết được vấn đề nhỏ của một số file “**True PDF**”, chưa đáp ứng được các file “**ScannerPDF**” – nhu cầu chính yếu hơn.
* Chưa thể xử lý chỉnh sửa văn bản rồi merge lại vào tệp gốc để thu được sản phẩm hoàn chỉnh nhất.
* Giao diện **Grammarly** đạt tiêu chuẩn, tuy nhiên việc chỉnh sửa văn bản với **Language Linter** đang gặp vấn đề văng màn hình khi số lượng từ kiểm tra mỗi lượt cho biến **sampleText** không thể quá dài (nhỏ hơn 64 ký tự).

🡪 Hiện giải pháp ban đầu là chia nhỏ văn bản ra và hiển thị và sửa lỗi lần lượt từng đoạn văn bản. Tuy nhiên như vậy gây ảnh hưởng rất lớn đến trải nghiệm người dùng.

* **Grammarly** plugin tương đối tốt, tuy nhiên đã không thể tiếp tục hoạt động dưới dạng tích hợp ch**o Developer.**

## Hướng giải pháp

* Với vấn đề “**ScannerPDF**”, hướng giải pháp là tìm hiểu và triển khai công nghệ OCR (optical character recognition) để có thể chuyển đổi sang được .doc và đọc văn bản dễ dàng hơn.
* Tìm hiểu nguyên nhân văng màn hình từ mã nguồn của **Language Linter** hoặc triển khai xử lý giao diện một cách hợp lý hơn.
* Có thể sử dụng các extensions thay thế cho Plugin đem lại hiệu quả tốt hơn.

# KẾT LUẬN

Trên đây là toàn bộ bài báo cáo Project 3 của em. Sau quá trình tìm hiểu và sự hướng dẫn của cô, em đã tổng hợp được một cách xúc tích và ngắn gọn về quá trình tìm hiểu và thiết kế giao diện Web sửa lỗi chính tả. Bài báo cáo thể hiện được những kiến thức cơ bản về cấu trúc, phân loại định dạng file PDF. Bên cạnh đó là các thông tin hữu ích về một số tool sửa lỗi chính tả hiệu quat. Ở giai đoạn kế tiếp, ta tiến hành cài đặt và thiết kế mã nguồn của môn học để có thể hoàn thành hai công việc chính đặt ra: “Upload và chuyển đổi File PDF” và “Tích hợp tiện ích sửa lỗi chính tả để kiểm tra văn bản”. Đã xảy ra những lỗi khi chạy sản phẩm, đã có những xử lý để khắc phục lỗi xảy ra, tuy nhiên với sự non trẻ trong kinh nghiệm debug và tìm hiểu các công nghệ, em chưa thể hoàn thiện được một cách tốt nhất.

**TÀI LIỆU THAM KHẢO**

* PDF: [PDF reader | Adobe Acrobat Reader](https://www.adobe.com/acrobat/pdf-reader.html)
* Grammarly: [Free Grammar Checker | Grammarly](https://www.grammarly.com/grammar-check)
* Language Linter: [danielgolden/language-linter: A natural language linter React component. (github.com)](https://github.com/danielgolden/language-linter/tree/main)
* Mã nguồn dự án: [quy261/project-3 (github.com)](https://github.com/quy261/project-3)

Bên cạnh đó có sự tham khảo viết báo cáo và mã nguồn thiết kế từ ChatGPT