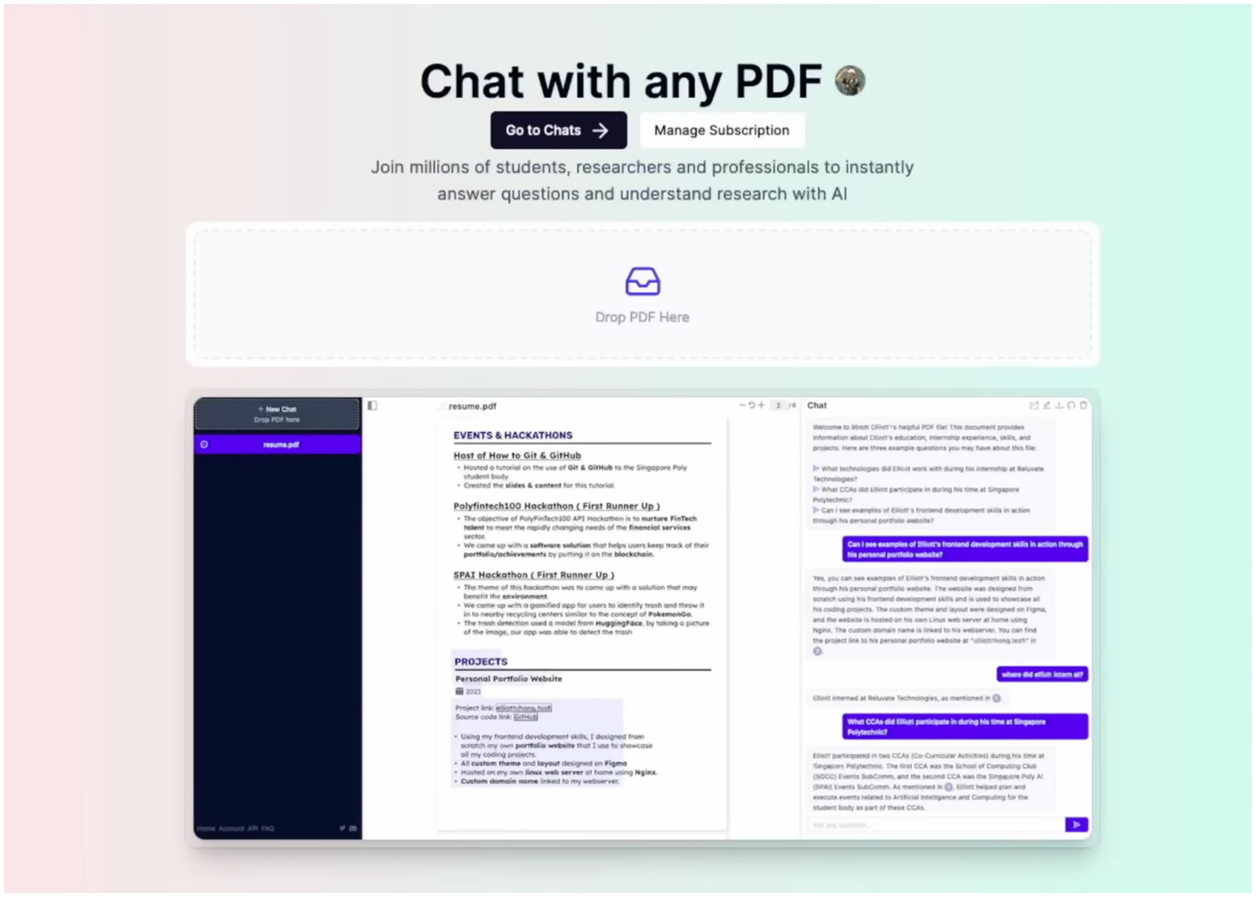
ChatPDF.ai

ProjectDocumentation

🎯 PDF Querying AI-Powered SaaS Application

Project Overview

This project involves building and deploying a Software-as-a-Service (SaaS) PDF Querying AI Application using the latest technologies, including NextJS 13.4, AWS S3 for PDF Storage, OpenAI, DrizzleORM, Stripe, TypeScript, Tailwind CSS, PineconeDB, Clerk, and Vercel. The primary goal of the project is to create a robust, scalable, and efficient platform for document analysis and querying, leveraging AI capabilities and seamless payment management. The project is designed to be edge-compatible, ensuring low latency and high performance.



Chat with any PDF

Join millions of students, researchers and professionals to instantly answer questions and understand research with AI

resume.pdf

EVENTS & HACKATHONS

Host of How to Git & GitHub

- Hosted a tutorial on the use of **Git & GitHub** to the Singapore Poly student body.
- Created the **slides & content** for this tutorial.

Polyfintech100 Hackathon (First Runner Up)

- The objective of PolyFinTech100 API Hackathon is to **nurture FinTech talent** to meet the rapidly changing needs of the **financial services** sector.
- We came up with a **software solution** that helps users keep track of their **portfolio/achievements** by putting it on the **blockchain**.

SPAI Hackathon (First Runner Up)

- The theme of this hackathon was to come up with a solution that may benefit the **environment**.
- We came up with a gamified app for users to identify trash and throw it in to nearby recycling centers similar to the concept of **PokemonGo**.
- The trash detection used a model from **HuggingFace**, by taking a picture of the image, our app was able to detect the trash

PROJECTS

Personal Portfolio Website

2021

Project link: elliottchong.tech
Source code link: GitHub

- Using my frontend development skills, I designed from scratch my own **portfolio website** that I use to showcase all my coding projects.
- All **custom theme** and **layout** designed on **Figma**
- Hosted on my own **linux web server** at home using **Nginx**.
- **Custom domain name** linked to my webserver.

Chat

Welcome to Elliott Elliott's helpful PDF file! This document provides information about Elliott's education, internship experience, skills, and projects. Here are three example questions you may have about this file:

What technologies did Elliott work with during his internship at Reluvate Technologies?
What CCAs did Elliott participate in during his time at Singapore Polytechnic?
Can I see examples of Elliott's frontend development skills in action through his personal portfolio website?

Can I see examples of Elliott's frontend development skills in action through his personal portfolio website?

Yes, you can see examples of Elliott's frontend development skills in action through his personal portfolio website. The website was designed from scratch using his frontend development skills and is used to showcase all his coding projects. The custom theme and layout were designed on Figma, and the website is hosted on his own Linux web server at home using Nginx. The custom domain name is linked to his webserver. You can find the project link to his personal portfolio website at "elliottchong.tech" in 2.

where did elliott intern at?

Elliott interned at Reluvate Technologies, as mentioned in 1.

What CCAs did Elliott participate in during his time at Singapore Polytechnic?

Elliott participated in two CCAs (Co-Curricular Activities) during his time at Singapore Polytechnic. The first CCA was the School of Computing Club (SOCC) Events SubComm, and the second CCA was the Singapore Poly AI (SPAI) Events SubComm. As mentioned in 1, Elliott helped plan and execute events related to Artificial Intelligence and Computing for the student body as part of these CCAs.

Ask any question...

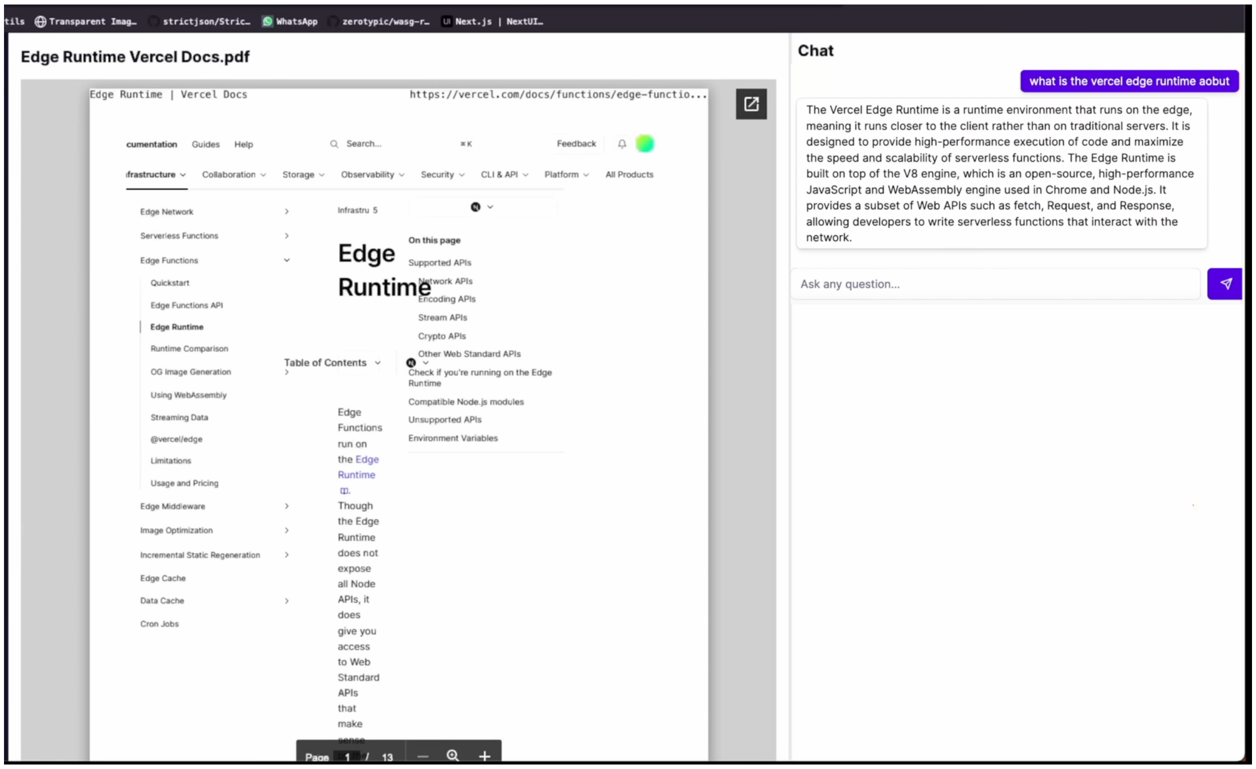
Project Overview

Key Features and Technologies

1. **NextJS 13.4**: Utilizing the cutting-edge App Router for efficient routing and server-side rendering.
2. **DrizzleORM**: Efficient database interaction with a TypeScript-friendly ORM.
3. **OpenAI**: Integrating language model capabilities for advanced querying and AI responses.
4. **Stripe**: Seamless payment processing and subscription management.
5. **TypeScript**: Ensuring type safety and enhancing code maintainability.
6. **Tailwind CSS**: Rapid and responsive UI development.

7. **PineconeDB**: A vector database for efficient storage and retrieval of embeddings, facilitating advanced AI querying.
8. **AWS S3**: Reliable and scalable storage solution for PDF documents.
9. **Clerk**: Comprehensive authentication solution to manage user authentication and authorization.
10. **Vercel**: Deployment and hosting on a scalable, serverless platform.
11. **Edge Compatibility**: Ensuring low latency and high performance by leveraging edge computing.

Demo



Edge Runtime Vercel Docs.pdf

Edge Functions run on the Edge Runtime. Though the Edge Runtime does not expose all Node APIs, it does give you access to Web Standard APIs that make

Chat

what is the vercel edge runtime aobut

The Vercel Edge Runtime is a runtime environment that runs on the edge, meaning it runs closer to the client rather than on traditional servers. It is designed to provide high-performance execution of code and maximize the speed and scalability of serverless functions. The Edge Runtime is built on top of the V8 engine, which is an open-source, high-performance JavaScript and WebAssembly engine used in Chrome and Node.js. It provides a subset of Web APIs such as fetch, Request, and Response, allowing developers to write serverless functions that interact with the network.

Ask any question...

Detailed Project Breakdown

Edge Runtime

The application leverages Edge Runtime to ensure fast and efficient execution of functions, providing a seamless user experience with minimal latency. Edge-compatible software services allow the application to run closer to the user, reducing response times and improving performance.

AWS S3 Integration

AWS S3 is utilized for reliable and scalable storage of PDF documents. By storing PDFs in S3, the application ensures that documents are readily available for processing and analysis. S3's integration with other AWS services provides enhanced security, accessibility, and performance, making it an ideal choice for handling large volumes of data in a scalable manner.

AI RAG (Retrieval-Augmented Generation)

Vectors and Embedding

1. **PDF Splitting**: Using LangChain to split PDF documents into smaller chunks, each containing 2-3 sentences or a single paragraph.
2. **Vector Conversion**: Converting each document chunk into a vector using OpenAI's embedding functions.
3. **Embedding Creation**: Creating embeddings that plot each sentence as a multidimensional vector representing its semantic value.
4. **Vector Database Storage**: Storing individual embeddings in PineconeDB for efficient retrieval.
5. **Query Processing**:
    - Converting user queries into vector embeddings.
    - Using Cosine Distance to find the most similar content in the vector database.
    - Feeding the most relevant content to OpenAI to generate context-aware responses.

PineconeDB

1. **Long-Term Memory for AI**: Storing vector embeddings for efficient, long-term retrieval.
2. **Query to Vector Embedding**: Converting queries into vector embeddings and searching for similar vectors in PineconeDB.
3. **Metadata Retrieval**: Extracting metadata from similar vectors and using it to provide context in OpenAI prompts.

React Query Setup

1. **Library Installation**: Using React Query to handle mutations and data querying between local and server states.
2. **Caching**: Leveraging React Query's caching capabilities to optimize resource usage by resolving requests from the cache when similar endpoints are hit.

Working with PineconeDB

1. **Index Creation**: Setting up indexes to organize vector data.
2. **Namespace Segmentation**: Using namespaces to segment PDF vector spaces, with each PDF having its own namespace.
3. **Cosine Similarity**: Utilizing Cosine Similarity for comparing vectors.
4. **PDF Handling**:
   - Downloading PDFs locally using the FS module.
   - Installing LangChain to convert PDFs into pages and split them into smaller segments for vectorization.
   - Using recursive character text splitter for efficient segmentation.
5. **Embedding with OpenAI**: Generating embeddings for each segment and inserting them into PineconeDB.
6. **Chat Data Management**: Creating functionality to push chat data to DrizzleORM and handling chat route endpoints.

Creating Chat Pages

1. **User Authentication**: Utilizing Clerk for managing user authentication and authorization, retrieving userId from the request URL, and checking authentication status.
2. **User Chats Retrieval**: Fetching all chats associated with the authenticated user.
3. **UI Layout**: Designing a three-column layout for chats, inline PDF viewing, and AI querying.
4. **PDF Viewing**: Integrating Google Docs for PDF viewing via S3 file URLs.
5. **Streaming Chat**: Using Vercel SDK and OpenAIStream for a dynamic ChatGPT-like experience.
6. **ShadCN Components**: Importing input and UI components from ShadCN for a polished interface.

Contextualizing PDF Queries for ChatGPT

1. **Query-String Mapping**: Mapping query strings to PDF namespaces.
2. **Embedding Matching**: Using getMatchesFromEmbeddings() to find the top 5 similar vectors with a match score above 70%.
3. **Metadata Utilization**: Joining qualifying vectors to form a document and using it to prompt GPT for context-aware responses.

Saving Messages to DrizzleDB

Ensuring all user messages and interactions are saved to DrizzleDB for persistent storage and retrieval.

Stripe Integration for Payments and Billing

1. **Subscription Management**: Using Stripe webhooks to manage user subscriptions and roles.
2. **Database Updates**: Adding rows to the database based on user subscription status and roles.

Conclusion

This project showcases the integration of various cutting-edge technologies to build a robust, scalable, and efficient SaaS application. By leveraging the power of NextJS, DrizzleORM, OpenAI, Stripe, TypeScript, Tailwind CSS, PineconeDB, AWS S3, Clerk, and Vercel, we created a platform capable of advanced document analysis, seamless payment processing, and dynamic user interactions. The use of AI for query processing and contextual responses, combined with edge compatibility, further enhances the user experience, making this SaaS application a powerful tool for document management and analysis.
