ARCHITECTURE

Project name:	**The Tool For Content Management**
21127065	Trần Bình Kha
21127333	Nguyễn Việt Kim
21127334	Lê Vũ Ngân Lam
21127337	Trần Tùng Lâm
21127466	Hoàng Anh Tú
21127505	Ngô Xuân Hiếu
21127545	Đặng Quốc Thái
21127597	Đỗ Dự Đức
21127150	Nguyễn Hoàng Nhật Quang

TABLE OF CONTENTS

I. Architecture

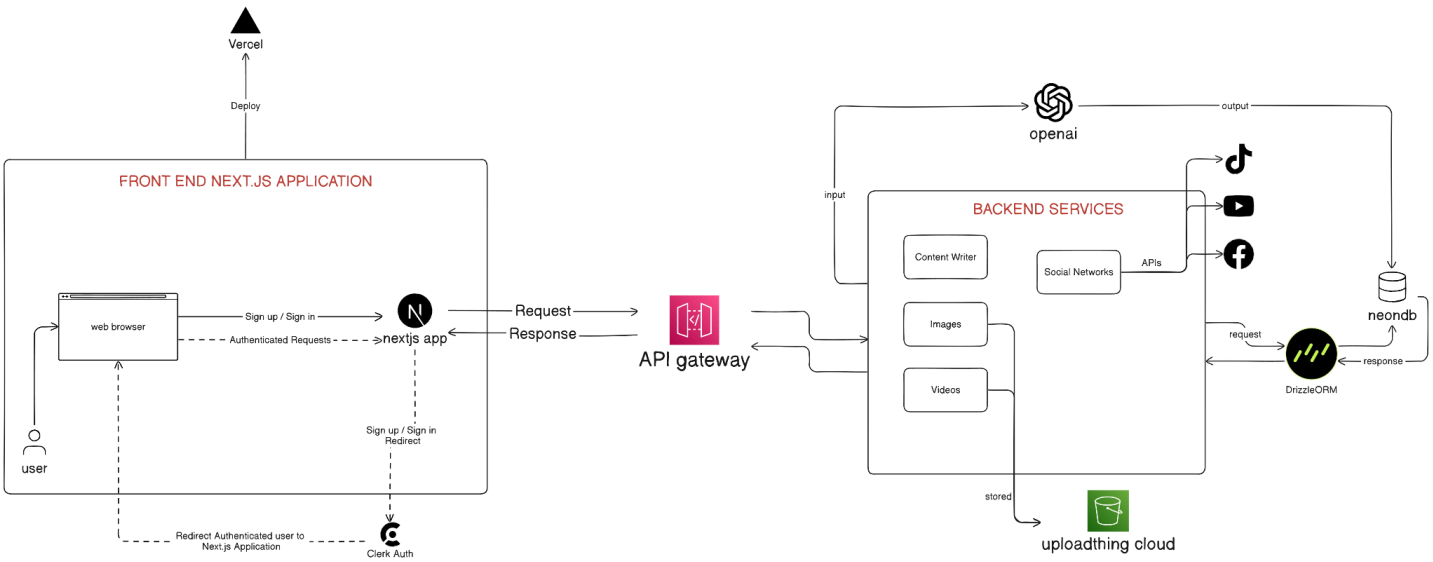

Figure 1: TFCM Architecture diagram

II. Detailed

1. Front-End: Next.js Application

1.1. Web Browser

- Users interact with the application through a web browser. They can sign up, sign in, and make authenticated requests to the Next.js app.

1.2. Next.js App

- The front-end application is built with Next.js, a React framework, and is deployed using Vercel.
- The app handles user authentication via Clerk Auth.
- It processes sign-up and sign-in requests and, once authenticated, redirects the user to the appropriate pages within the application.
- Authenticated requests from the web browser are sent to the Next.js app, which then communicates with the API gateway for further processing.

1.3. Clerk Auth

- Clerk Auth is used for handling authentication. It manages the sign-up and sign-in process, and redirects authenticated users back to the Next.js application.

2. API Gateway

- The API gateway acts as an intermediary that processes requests from the Next.js app and routes them to the appropriate back-end services. It also handles the responses from these services and sends them back to the Next.js app.

3. Back-End Services

3.1. Content Writer

- This service likely generates or manages content for the application. It may interact with external APIs, such as OpenAI, to create or manipulate content.

3.2. Images and Videos

- These services handle image and video processing, respectively. They store processed media in the "uploadthing cloud".

3.3. Social Networks

- This component integrates with social media platforms like TikTok, YouTube, and Facebook. It uses APIs to interact with these platforms, possibly for sharing content or retrieving social media data.

4. External Integrations

4.1. OpenAI

- The application integrates with OpenAI to process input and generate outputs, possibly for content creation or other AI-driven features.
- OpenAI: https://openai.com

4.2. NeonDB

- NeonDB is the database used to store application data. It communicates with the back-end services through DrizzleORM, a type of Object-Relational Mapping (ORM) framework.
- NeonDB: https://neon.tech

4.3. DrizzleORM

- DrizzleORM facilitates the interaction between the back-end services and NeonDB, managing database queries and responses.
- DrizzleORM: https://orm.drizzle.team

4.4. Uploadthing Cloud

- This service is used to store media files like images and videos processed by the back-end services.
- Uploadthing: https://uploadthing.com

4.5. Clerk Authentication

- Clerk is a complete suite of embeddable UIs, flexible APIs, and admin dashboards to authenticate and manage your users.
- Clerk: https://clerk.com

5. Workflow Summary

- A user accesses the application via a web browser.
- They sign up or sign in through the Next.js app, which uses Clerk Auth for authentication.
- Upon successful authentication, the user is redirected to the application's authenticated section.
- The Next.js app sends requests to the API gateway, which routes them to the appropriate back-end services.
- The back-end services perform necessary operations, which may include interacting with OpenAI, processing media files, or communicating with social networks.
- Data is stored in NeonDB, managed by DrizzleORM.
- Processed media is stored in the Uploadthing cloud.
- The responses are sent back through the API gateway to the Next.js app, which then updates the web browser for the user.

This architecture efficiently separates the front-end from the back-end services, uses modern frameworks and tools for scalability, and integrates robust external services for enhanced functionality.

III. Deployment

The deployment aspect of this architecture involves the use of several platforms and services to ensure the application is efficiently deployed, managed, and scalable. Here's a detailed look at the deployment setup:

1. Front-end deployment with Vercel

- The Next.js application, which forms the front-end of this architecture, is deployed on Vercel.
- Vercel provides a seamless integration with Next.js, allowing for automatic deployments from repositories like GitHub, GitLab, or Bitbucket.
- Each time code is pushed to the repository, Vercel automatically builds and deploys the latest version of the application.
- Vercel handles tasks such as server-side rendering (SSR), static site generation (SSG), and serverless functions, providing a scalable and performant environment for the Next.js app.
- The deployment process includes setting environment variables, building the application, and ensuring zero-downtime deployments by leveraging Vercel's infrastructure.

2. Database deployment with NeonDB

- NeonDB, the database service, is deployed in a cloud environment optimized for high performance and availability.
- The database deployment includes setting up automated backups, scaling configurations, and security measures such as encryption and access controls.
- DrizzleORM facilitates database interactions, ensuring that the back-end services can efficiently perform CRUD operations.

3. Media storage deployment with Uploadthing cloud

- The Uploadthing cloud service is used to store images and videos processed by the back-end services.
- This service can be deployed on object storage solutions like Amazon S3, Google Cloud Storage, or Azure Blob Storage.
- Deployment involves configuring the storage buckets, setting up access policies, and integrating with the back-end services for seamless media uploads and retrievals.

4. Environment with external API services

- The integration with external services like OpenAI and social media platforms (TikTok, YouTube, Facebook) involves deploying API clients and handling authentication.
- These integrations are configured during the deployment process to ensure secure and efficient communication with the external services.