VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**Nguyen Tran Nhat An – 523H0115**

**Chung Quang Vu – 523H0196**

**Tran Quang Anh – 523H0003**

**MIDTERN ESSAY**

**Web Programming & Applications**

**HO CHI MINH CITY, 2025**

VIETNAM GENERAL CONFEDERATION OF LABOR

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**Nguyen Tran Nhat An – 523H0147**

**Chung Quang Vu - 523H0196**

**Tran Quang Anh – 523H0003**

**MIDTERN ESSAY**

**Web Programming & Applications**

Advised by

**Mr. Mai Van Manh**

**HO CHI MINH CITY, 2025**

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to Mr. Mai Van Manh, our supervisor and lecturer, computer science department, information technology department and Ton Duc Thang University for her guidance and Teachers' valuable support throughout our final report. Thank you teachers for creating an environment for me to cultivate strong and modern knowledge. Once again, I would like to thank my teachers for giving me access to learning with new technologies and knowledge to improve myself and increase my learning efficiency. We have learned a lot from her expertise and experience. We are very honored and proud to have her as our teacher and supervisor.

*Ho Chi Minh city, 10th April 2025.*

*Author*

*(Signature and full name)*

***An***

Nguyen Tran Nhat An

***Anh***

Tran Quang Anh

***Vu***

Chung Quang Vu

**DECLARATION OF AUTHORSHIP**

We declare that this project is our original work, supervised by Mr. Mai Van Manh. The content and research results presented here are original and have not been previously published in any form. The data used for analysis, comment, and evaluation were collected by the main author from many different sources, and are clearly cited in the reference section.

In addition, the project incorporates comments, reviews, and data from other authors and organizations with appropriate citations and footnotes. **We are fully responsible for the content of this project.** Ton Duc Thang University is not responsible for any rights or copyright violations that may arise during the implementation of this project.

*Ho Chi Minh city, 10 April 2025*

***An***

Nguyen Tran Nhat An

***Anh***

Tran Quang Anh

***Vu***

*Chung Quang Vu*

**TABLE OF CONTENT**

**Chapter I. Theoretical ………………………………4**

1.1 Overview of Video Sharing Systems……………………………..4

1.2 System Architecture………………………………………………4

1.3 Supporting Techniques…………………………………………...6

**Chapter II. Building the WeTube Platform………...7**

2.1 Project Overview…………………………………………………7

2.2 Main Features…………………………………………………….7

2.3 Technologies Used……………………………………………….8

**Chapter III. Demo……………………………………8**

3.1 Source Code Repository………………………………………….8

3.2 Demo Video………………………………………………………8

**Chapter I. Theoretical**

* 1. **Overview of Video Sharing Systems**

WeTube is a web application inspired by YouTube, designed and developed by our team using a client-server architecture model. The platform allows users to register, upload videos, watch content, and interact through comments. Our goal is to create a simplified yet functional video-sharing platform, leveraging modern web technologies.

* **Backend**: Built using Node.js along with the Express.js framework to define RESTful API endpoints. We used MongoDB as our primary database to store video metadata, user information, and comments. We integrated JWT (JSON Web Token) to handle secure login and route protection. Tokens are stored client-side and sent in headers for authenticated API access.
* **Frontend**: Implemented using ReactJS and Vite for a fast and modular build environment. Axios is used to fetch data from backend APIs and render dynamic content on the client side.
* **Database**: MongoDB is used as the primary database. It stores structured data such as user accounts, video metadata, and comment threads in separate collections.

**1.2 System Architecture**

**Frontend Structure:**

* **Context**
  + *User Context*: Manages global user authentication state, such as tokens and user info across all pages.
* **Components**
  + *Video Card*: Displays thumbnail, title, and uploader for each video.
  + *Auth Form*: Reusable form for login and registration.
  + *Comments Container*: Handles displaying and submitting comments under a video.
  + *Main Container*: Wrapper component for page layout.
  + *Navigation Bar*: Top-level navigation with links to key pages.
  + *Video Container*: Renders the video player and associated metadata.
* **Pages**
  + *Account Page*: Displays user's personal details and settings.
  + *Channel Page*: Shows a specific user's uploaded videos.
  + *My Videos Page*: A dashboard for managing the videos a user has uploaded.
  + *Register/Login Page*: Allows new users to register and existing users to log in.
  + *Upload Page*: Form interface for uploading new videos.
  + *Watch Page*: Main video viewing page with player, comments, and related videos.

**Backend Structure:**

* **Models**
  + *Users*: Schema for storing user info (username, email, hashed password).
  + *Videos*: Schema for storing video details (title, path, uploader, views).
  + *Comments*: Schema for video comments linked by videoId and userId.
* **Controllers**
  + *Auth Controller*: Handles user registration and login logic.
  + *Video Controller*: Manages upload, retrieve, and view count update for videos.
  + *Comment Controller*: Manages adding and retrieving comments for videos.
* **Routes**
  + *Auth Route*: API endpoints for login and registration.
  + *User Route*: Handles fetching user profile and related data.
  + *Comment Route*: Manages comment creation and retrieval.
  + *Video Route*: Manages upload, fetch, and metadata update for videos.
* **Middleware**
  + *Auth Middleware*: Verifies JWT token to protect private routes.
  + *Upload Video Middleware*: Processes video uploads using Multer.
* **Utils**
  + *Generate Token*: Issues JWTs for authenticated users.
  + *Hash Password*: Secures passwords using bcrypt hashing before saving to DB.

**Database:**

* *MongoDB*: NoSQL database used to store structured collections for users, videos, and comments.

**1.3 Supporting Techniques**

Supporting technologies and development best practices are vital to delivering an optimized and scalable solution:

* **AJAX / Fetch API**: Used to retrieve dynamic content (e.g., new comments or likes) asynchronously without a full page reload.
* **Storage & CDN**: Video files consume significant disk space. Storage systems can be hosted on local servers or integrated with cloud platforms (e.g., AWS S3, Firebase, or Cloudinary). To enhance video delivery speed, CDNs (Content Delivery Networks) distribute content geographically closer to users.
* **Streaming Engine**: Progressive download and HLS (HTTP Live Streaming) are two major approaches. HLS breaks videos into segments and serves them adaptively based on connection speed, ensuring smooth playback.
* **Security Layer**: Crucial for protecting user data and system resources. Common practices include password hashing with bcrypt, implementing JWT (JSON Web Token) for session management, validating inputs to prevent SQL Injection or Cross-Site Scripting (XSS).
* **Cloud and CI/CD Tools**: Continuous integration/deployment pipelines (e.g., GitHub Actions, Docker) automate testing and deployment. Cloud-based development tools improve collaboration and deployment flexibility.

**Chapter II. Building the WeTube Platform**

**2.1 Project Overview**

The WeTube project is a web application that enables users to register accounts, log in, upload and view videos, and interact with content through views and comments. The system follows a separated frontend-backend architecture. It applies RESTful API principles, secures user authentication using JWT, and provides a user-friendly interface developed using React and Tailwind CSS.

**2.2 Main Features**

1. User registration and login.
2. Upload videos from local machines to the server.
3. Video playback with automatic view count increment.
4. Post and display comments under each video.
5. List all uploaded videos.
6. Detailed video page including uploader info and comments.

**2.3 Technologies Used**

|  |  |
| --- | --- |
| **Component** | **Technology** |
| Frontend | React, Tailwind CSS, Vite |
| Backend | Node.js, Express.js |
| Database | MongoDB |
| File Upload | Multer |
| Authentication | JWT (JSON Web Token) |
| API Testing | Postman (optional) |
| Routing | React Router DOM |

**Chapter III. Demo**

**3.1 Source Code Repository**

You can view or download the full source code of the WeTube project from the following link:

[Insert GitHub or Drive link here]

**3.2 Demo Video**

A recorded demo video showcasing the key features and functionality of WeTube is available at the link below:

[Insert YouTube or Drive video link here]