Full Stack Development with MERN

ViewVoyage - Video sharing app - FSD

Team ID: SWTID1742745633

1. INTRODUCTION

1.1 Project Overview

ViewVoyage is a web-based video streaming and sharing platform that allows users to explore, search, and interact with video content. The platform enables user authentication (sign in), supports features such as maintaining a personalized list ("Your List"), liking videos, downloading content, sharing, and commenting. It focuses on a smooth and engaging user experience while leveraging modern web technologies for efficient performance and scalability.

1.2 Purpose

The purpose of ViewVoyage is to create an interactive and user-friendly video streaming platform where users can watch, search, and manage videos with ease. The platform aims to provide a personalized experience by allowing users to sign in, maintain their own watchlists, and engage with content through features such as liking, sharing, commenting, and downloading videos. The project also serves as a full-stack development practice, integrating both frontend and backend technologies in a real-world scenario.

2. IDEATION PHASE

2.1 Problem Statement

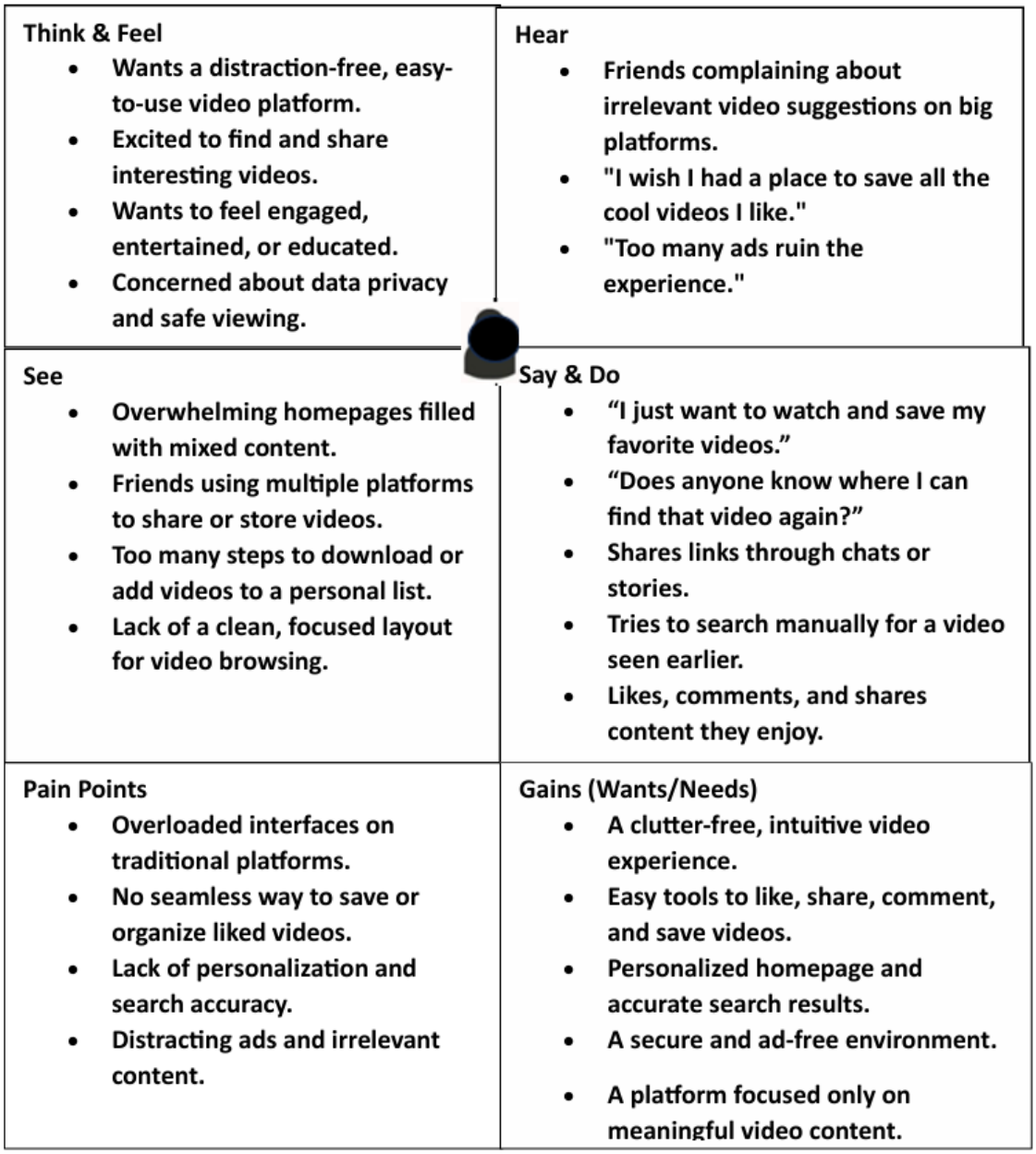
How can we create a clean, distraction-free video platform that focuses only on essential viewing features like play, like, comment, and save, without overwhelming the user with too many options?

Can a simplified layout improve user engagement and content discoverability?

What’s the most user-friendly way to let viewers interact with video content — including searching, sharing, and organizing their favorite videos — while keeping the experience fast and intuitive?

2.2 Empathy Map Canvas

Empathy Map



2.3 Brainstorming

- Focus on delivering a distraction-free video viewing experience with only essential features like play, like, comment, save, and download.
- Design a clean and responsive interface that adapts smoothly to both mobile and desktop platforms, emphasizing intuitive navigation.

- Implement a lightweight and fast search functionality that filters videos by title with instant result rendering.
- Personalize user experience through features like “Your List” and “Liked Videos,” stored securely and persistently.
- Offer interactive video detail pages with shareable links, comment threads, and engagement options.
- Structure the platform using a modular MERN stack architecture for future scalability, including creator tools and analytics.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

Landing Page → Sign In → Browse or Search Videos → Watch Video → Like / Comment / Save → Add to "Your List" → Download or Share → Revisit Saved Content → Seamless Repeat Experience

3.2 Solution Requirement

Functional Requirements

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Management	- Registration through Form - Login via Form
FR-2	Video Management	- Browse Video Catalog - View Video Details - Upload Videos
FR-3	User Interaction	- Like Videos - Add to “Your List” - Comment on Videos - Download Videos
FR-4	Search Functionality	- Search Videos by Title
FR-5	Admin Management	- View User List - Delete Users - Manage Reports

Non-functional Requirements

Following are the non-functional requirements of the proposed solution.

NFR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Interface should be simple, responsive, and easy to navigate across devices using React and Bootstrap.
NFR-2	**Security**	JWT-based authentication, bcrypt password hashing, and HTTPS encryption should be enforced.
NFR-3	**Reliability**	MongoDB must be backed up and replicated to prevent data loss and ensure data integrity.
NFR-4	**Performance**	The app should handle up to 1000 concurrent requests with loading times under 2 seconds.
NFR-5	**Availability**	Maintain 99.9% uptime using load balancers and basic failover strategies.
NFR-6	**Scalability**	Use a scalable 3-tier architecture with the ability to horizontally scale using MongoDB sharding.

3.3 Data Flow Diagram

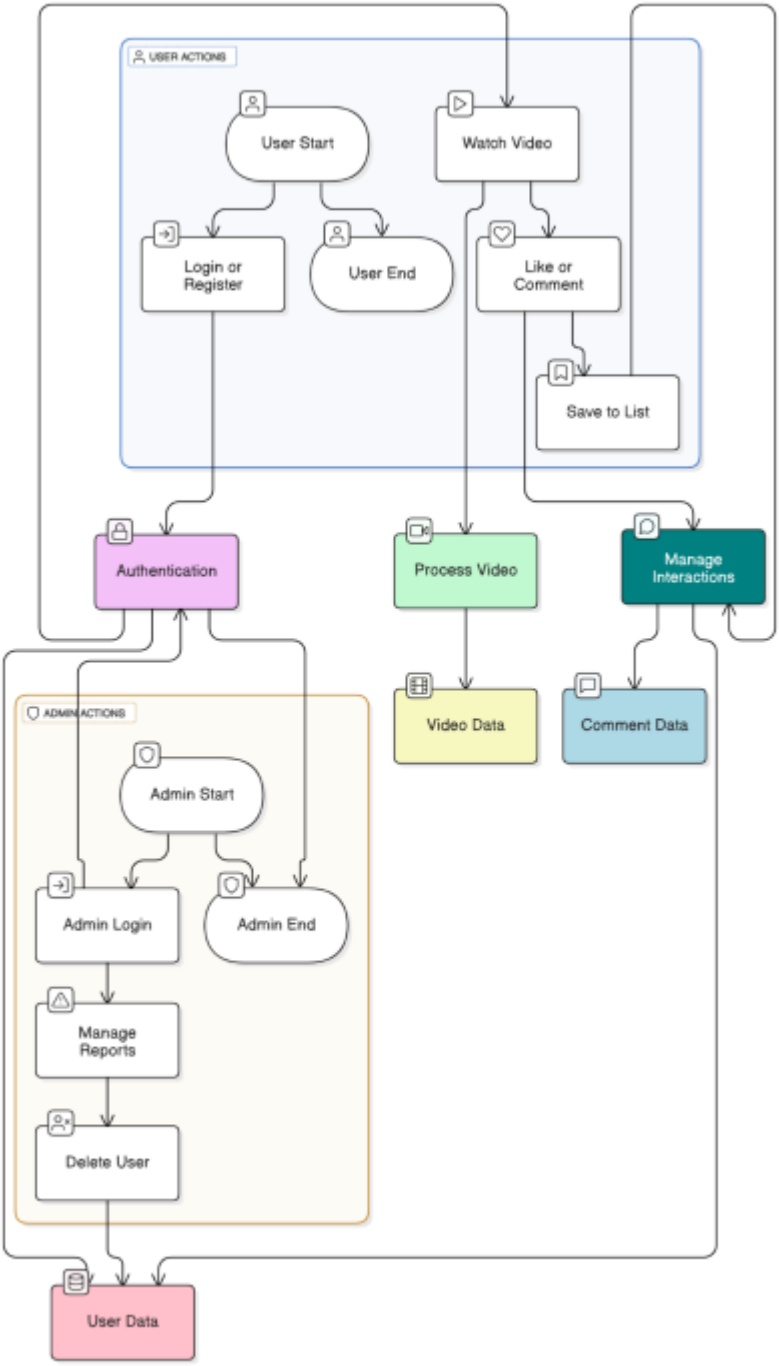
ViewVoyage – Data Flow Diagrams

Level-0 DFD Description

Actors:

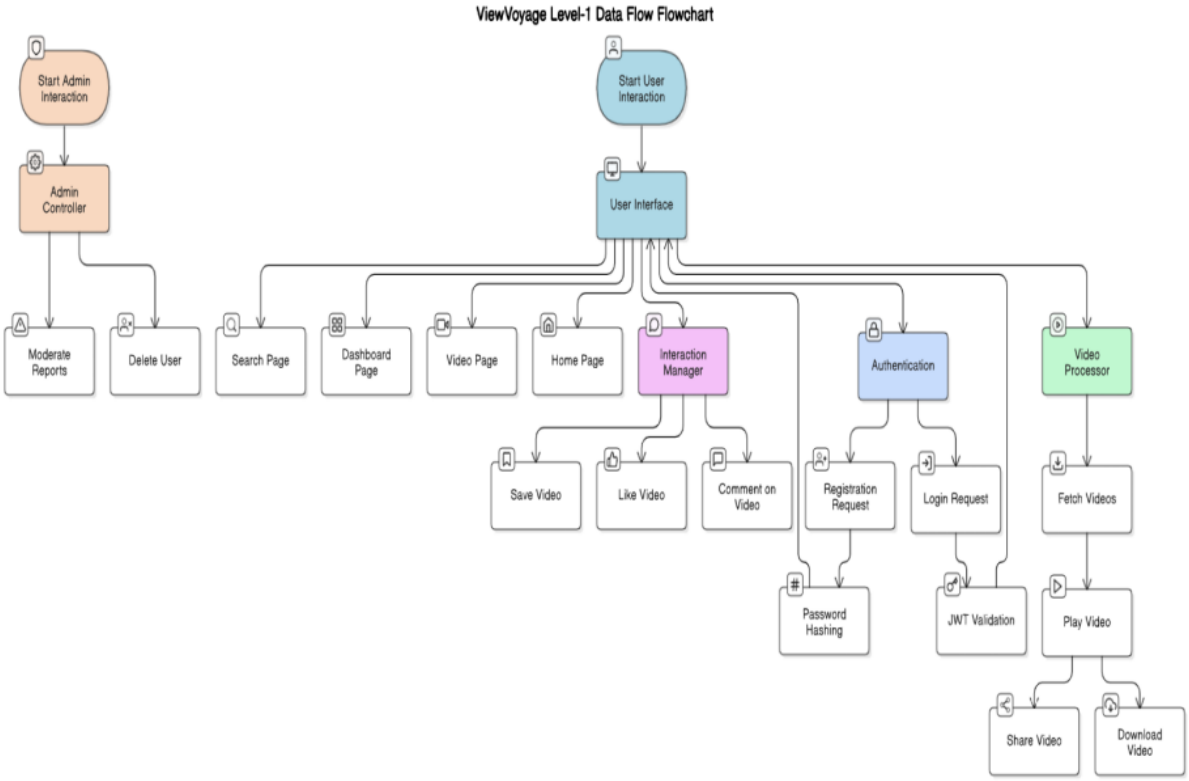
- **User:** A viewer who watches, likes, comments, and saves videos.
- **Admin:** Manages platform activities and users.
- **Process Video:** Handles video uploads, playback requests, and downloads.
- **Manage Interactions:** Processes likes, comments, and user-specific data like "Your List."
- **Authentication:** Handles login and registration.
- **Admin Actions:** Deletes users or manages reports.
- **Video Data:** Stores video metadata, URLs, and thumbnails.
- **User Data:** Stores user credentials, lists, liked videos.
- **Comment Data:** Stores comments linked to videos and users.

ViewVoyage Level-0 Data Flow (Flow Chart)



Level-1 DFD Processes

Process	Function
User Interface (UI)	Manages all user interactions including home, video, search, and dashboard
Authentication	Handles secure login and registration using JWT and hashed passwords
Video Processor	Fetches videos, plays them, and enables download or sharing
Interaction Manager	Manages likes, comments, and saving videos to user-specific lists
Admin Controller	Enables admin functions like deleting users or moderating reports



3.4 Technology Stack

Technical Architecture

ViewVoyage uses a modern **MERN stack architecture** to deliver a high-performing, scalable video-sharing platform. The architecture supports secure authentication, video playback, interactive features, and responsive design optimized for both desktop and mobile devices.

Table-1: Components & Technologies

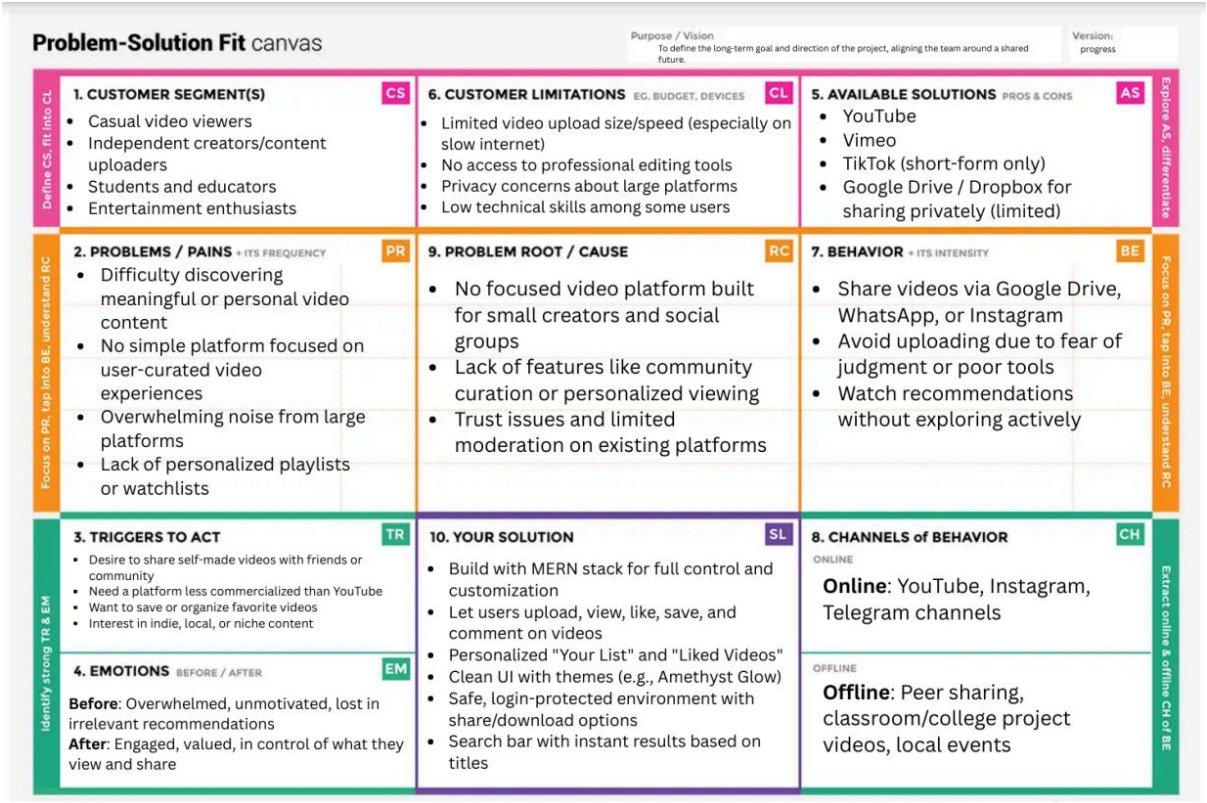
S.No	Component Description	Technology Used
1	**User Interface**: Web UI to browse, search, and interact with videos. Includes theme switcher and animations.	React 18.x, Bootstrap 5, CSS3
2	**Application Logic - Auth**: Handles login, registration, JWT authentication, and route protection.	Node.js 20.x, Express.js 4.x, JWT, bcrypt
3	**Application Logic - Video**: Manages video upload, detail retrieval, and streaming logic.	Node.js, Express.js, Multer
4	**Application Logic - User Interaction**: Handles liking, commenting, and saving videos to lists.	Express.js, Mongoose 8.x
5	**Database**: Stores users, videos, comments, and likes as JSON documents.	MongoDB 7.x
6	**File Storage**: Local server stores uploaded videos and thumbnails.	Local Filesystem (Multer/File System)
7	**Infrastructure**: Local deployment with Node server; optionally cloud-deployable.	Vite, Node.js, MongoDB

Table-2: Application Characteristics

S.No	Characteristic	Description	Technology Used
1	**Open-Source Frameworks**	Fully built using open-source frontend, backend, and DB technologies.	React, Express, MongoDB, Mongoose, Bootstrap
2	**Security Implementations**	Uses token-based auth and secure password handling.	JWT, bcrypt, HTTPS
3	**Scalable Architecture**	3-tier architecture (UI, Logic, Data) allows for vertical and horizontal scaling.	MongoDB sharding (future), Express.js
4	**Availability**	MongoDB replication ensures data remains accessible even if one instance fails.	MongoDB Replica Sets
5	**Performance Optimization**	Indexing, caching, and optimized queries support high concurrent usage.	MongoDB Indexes, optional Redis

4. PROJECT DESIGN

4.1 Problem Solution Fit



4.2 Proposed Solution

Proposed Solution:

S.No.	Parameter	Description
1	**Problem Statement (Problem to be solved)**	Users struggle to find a user-friendly and trustworthy platform to upload, discover, and share videos with social features like comments, likes, and collections. Most platforms are either too generic or lack personalized discovery and control.
2	**Idea / Solution Description**	ViewVoyage is a video-sharing platform that allows users to upload, browse, and interact with videos through features like search by title, user authentication, video details, likes, watchlist, and download functionality — all with a clean and modern interface.
3	**Novelty / Uniqueness**	Combines core streaming features with personalized watchlists, aesthetic UI themes, and simplified UX tailored for casual creators and viewers; supports a community-oriented, privacy-conscious experience.
4	**Social Impact / Customer Satisfaction**	Promotes expression and sharing of knowledge or creativity among individuals, especially students and content creators; encourages safe interactions with options to moderate and manage content.
5	**Business Model (Revenue Model)**	Freemium model with optional ad-free experience, premium creator tools, and sponsored video placements; partnerships with educational content creators and institutions for exclusive content.
6	**Scalability of the Solution**	Designed with modular React components and a scalable backend (Node.js + Express), making it easy to add features like real-time comments, analytics, or mobile app support in the future.

4.3 Solution Architecture

ViewVoyage – Solution Architecture

1. Frontend (Presentation Layer)

- Built with **React.js** using JSX components.

- Includes pages like Homepage, Video Detail, Sign-in, Liked Videos, and Your List.
- Uses **Bootstrap** and **CSS** for responsive styling.
- Communicates with backend via **Axios/Fetch** using RESTful APIs.
- Implements route protection using React Router for authenticated views.

2. Backend (Application Layer)

- Developed using **Node.js** and **Express.js**.
- Handles API endpoints for:
    - User authentication (login, token generation)
    - Video listing and search
    - Like, comment, download, and save to list
- Validates input and handles business logic.
- Middleware for JWT authentication and error handling.

3. Database (Data Layer)

- **MongoDB** used to store structured data in collections:
    - Users: credentials, liked videos, saved list
    - Videos: titles, descriptions, URLs, thumbnails
    - Comments: user comments linked to videos
- Uses **Mongoose** for schema modeling and querying.

4. File Storage

- Videos and thumbnails are uploaded and stored in the **local file system**.
- Future scope: migrate to cloud storage (e.g., AWS S3 or Firebase Storage).

5. Authentication & Security

- User authentication using **JWT (JSON Web Tokens)**.
- Passwords are hashed using **bcrypt** before storage.
- Routes that require authentication are protected with middleware.

6. Scalability Considerations

- Modular code structure for future microservices architecture.
- MongoDB supports sharding for horizontal scaling.
- Stateless authentication enables horizontal scaling of the backend.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	User Registration & Login	VW-US1	As a user, I can register and login securely with email and password	10	High	Member 1
Sprint-1	Video Search and Listing	VW-US2	As a user, I can browse all videos and search by title	10	High	Member 2
Sprint-2	Video Detail & Interactions	VW-US3	As a user, I can click a video to view, like, comment, download, and add to my list	10	High	Member 3
Sprint-2	Your List & Liked Videos	VW-US4	As a user, I can view all my saved and liked videos from a dedicated page	10	Medium	Member 4
Sprint-3	Commenting System	VW-US5	As a user, I can add and view comments on a video	15	High	Member 1
Sprint-4	Share Feature + UI Theme Switching	VW-US6	As a user, I can share a video via link and switch between themes	10	Medium	Member 2
Sprint-4	Secure Authentication & Route Guards	VW-US7	As a user, I cannot access private pages without logging in	10	High	Member 3

6. FUNCTIONAL AND PERFORMANCE TESTING

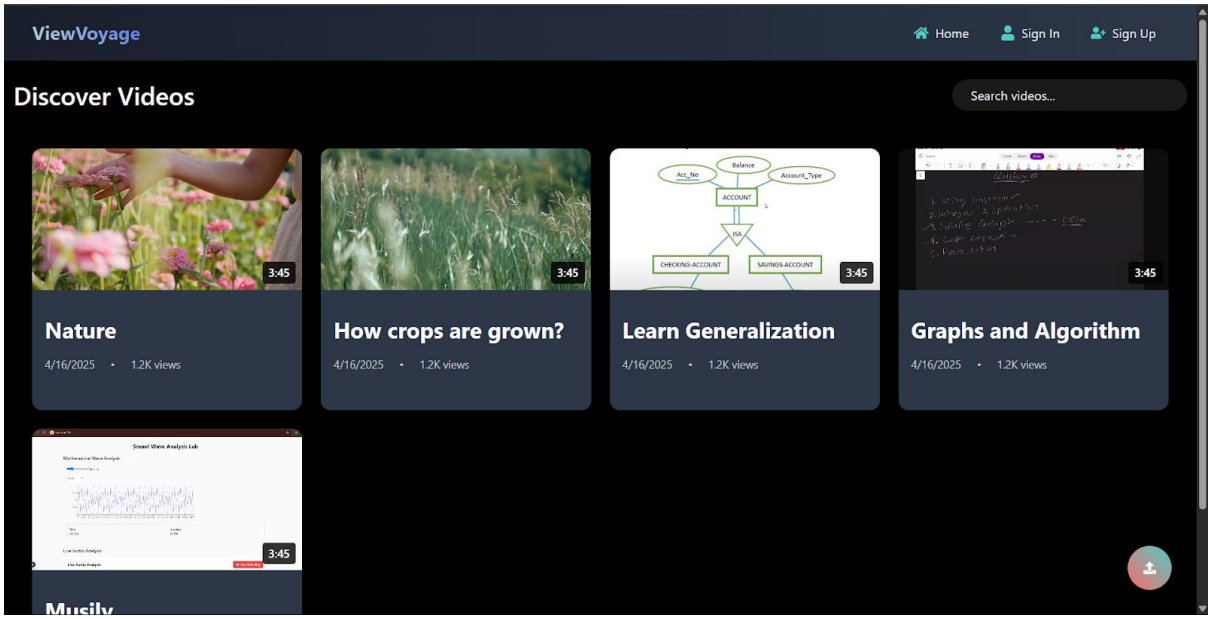
6.1 Performance Testing

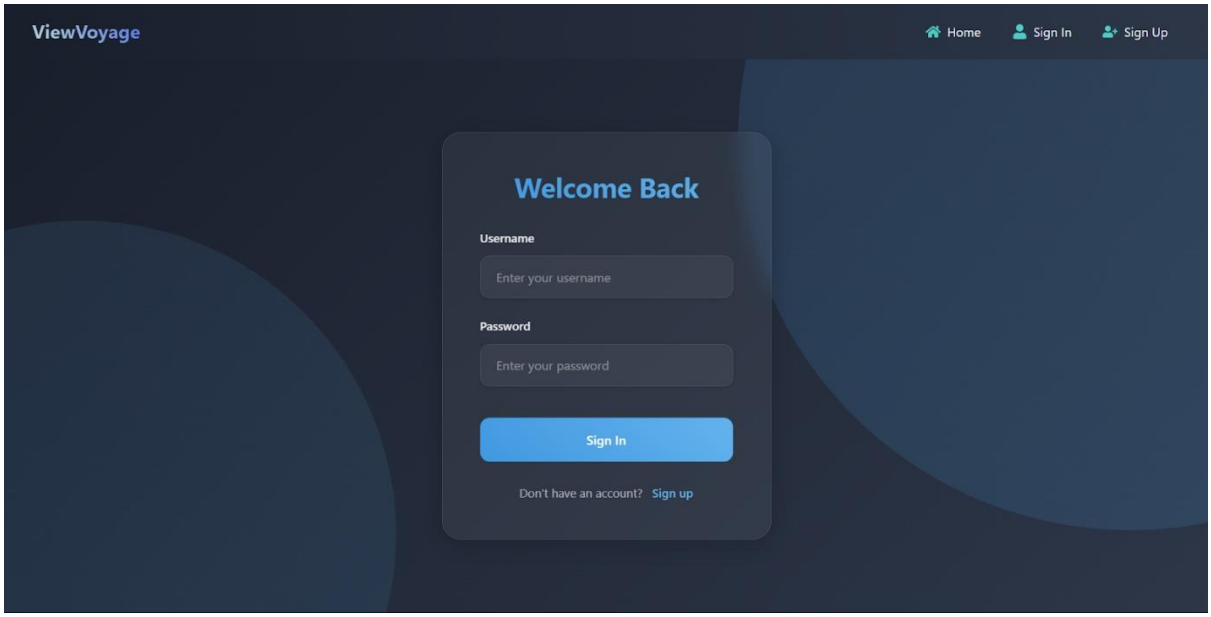
Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Pass/Fail
TC-001	User Registration	1. Navigate to signup page2. Enter name, email, password3. Click "Sign Up"	User is registered successfully and redirected to login page	User registered and redirected	Pass
TC-002	User Login with Invalid Details	1. Go to login page2. Enter invalid email/password3. Click "Login"	Error message: “Invalid credentials”	Error message displayed	Pass
TC-003	Video Upload	1. Login as user2. Navigate to upload page3. Upload a video file and enter details4. Submit	Video appears on homepage with correct details	Video uploaded and listed correctly	Pass
TC-004	Search for Video	1. Go to search bar2. Enter video title or keyword3. Click search	Videos matching search query are displayed	Correct results shown	Pass
TC-005	Like a Video	1. Login and open any video2. Click "Like"	Like count increases and button shows as "Liked"	Like registered, count updated	Pass
TC-006	Add Video to	1. Open a video2. Click	Video is added to user’s saved list	Video appears in “Your	Pass

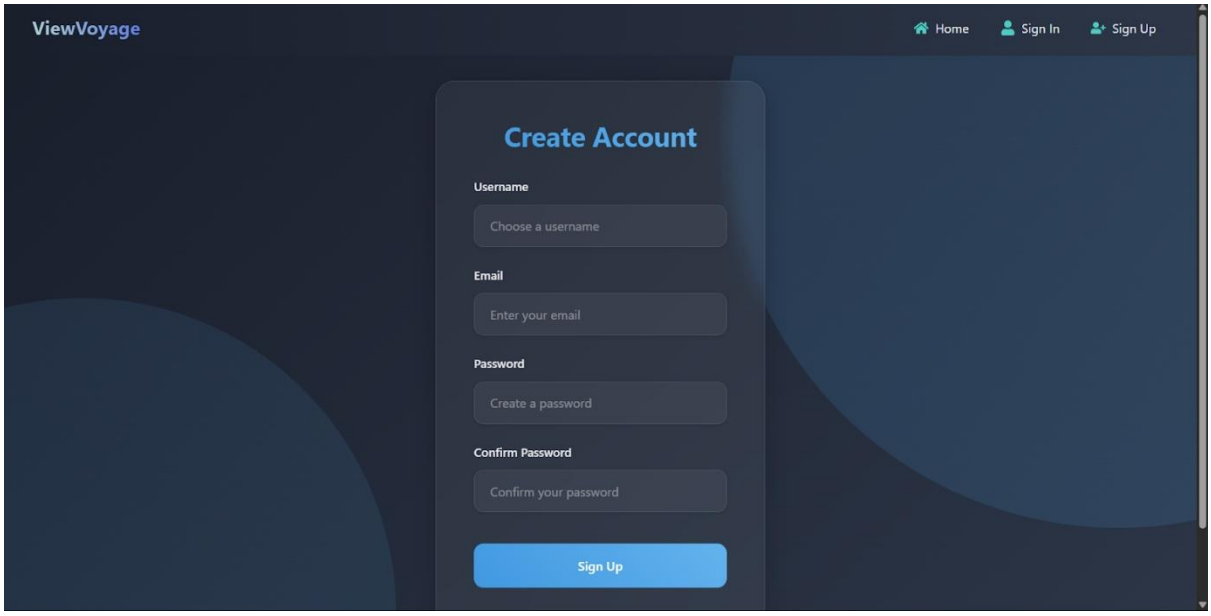
	“Your List”	“Add to Your List”		List” section	
TC-007	Download Video	1. Click on a video2. Click "Download"	Video starts downloading	Video downloaded successfully	Pass
TC-008	Post a Comment	1. Open video page2. Type a comment and submit	Comment appears under video	Comment displayed immediately	Pass
TC-009	Video Detail View	1. Click on any video card	Page shows video, title, description, like/share/comment/download buttons	All details and actions visible	Pass
TC-010	Route Protection for Unauthorized Users	1. Try accessing “Your List” or “Liked Videos” page without logging in	Redirected to login page	Route protected and redirected correctly	Pass

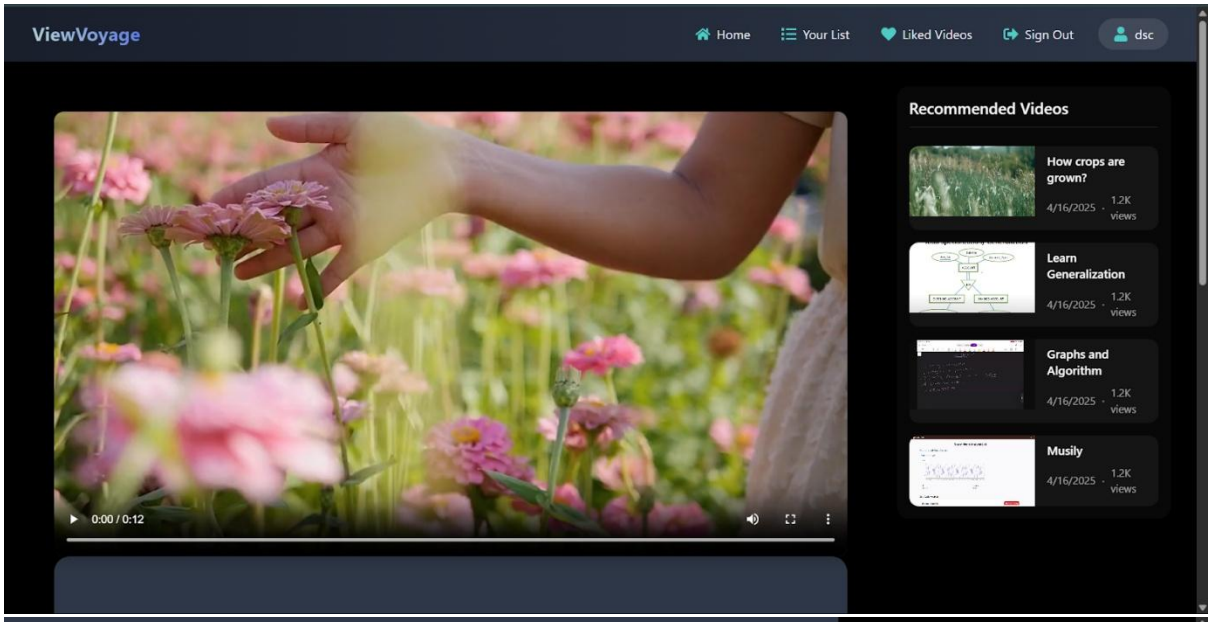
7. RESULTS

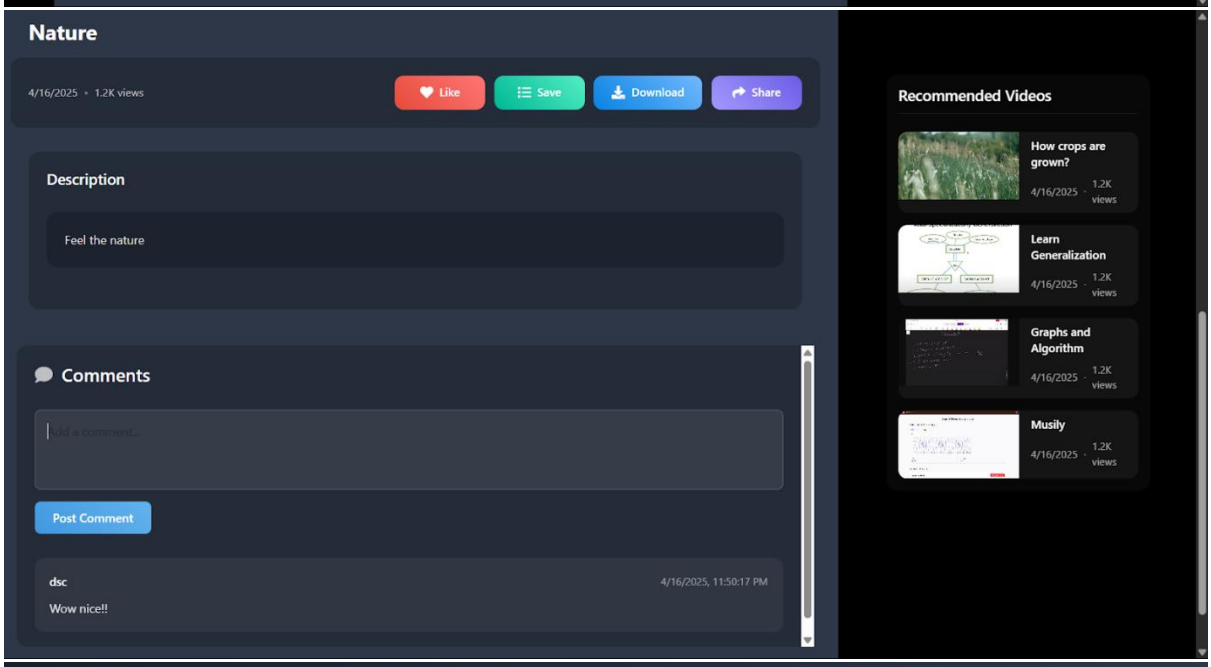
7.1 Output Screenshots

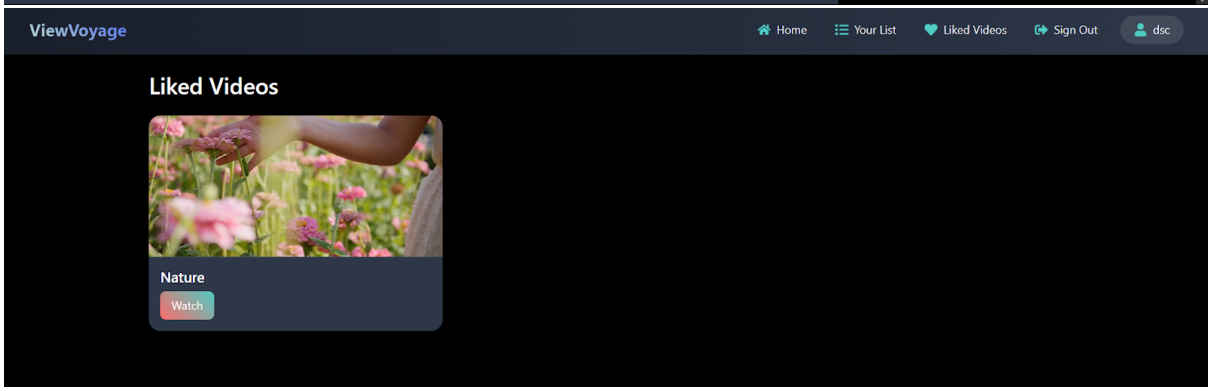


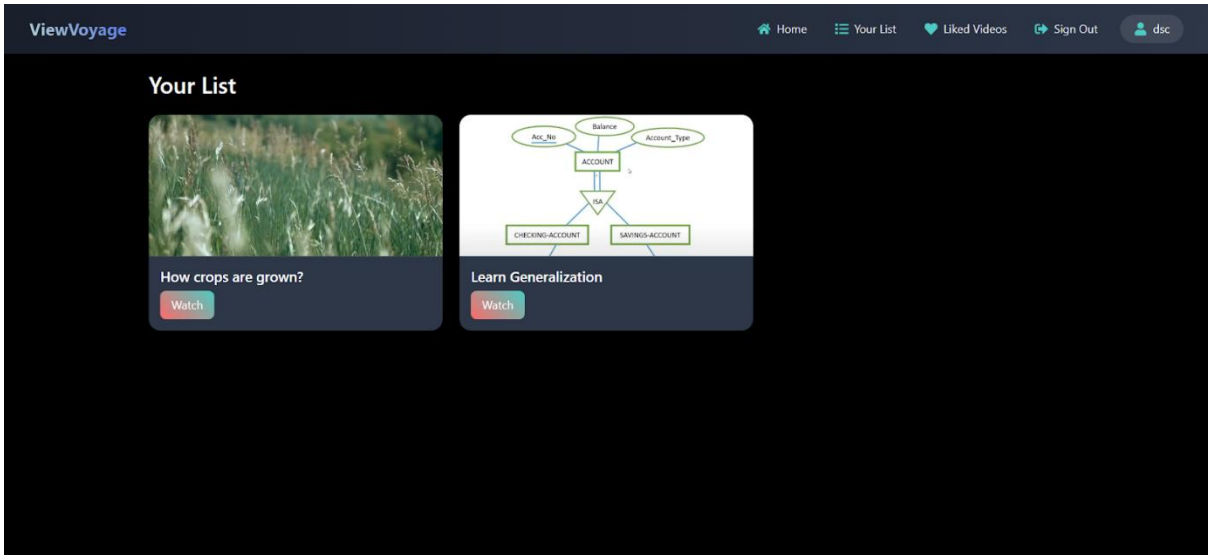


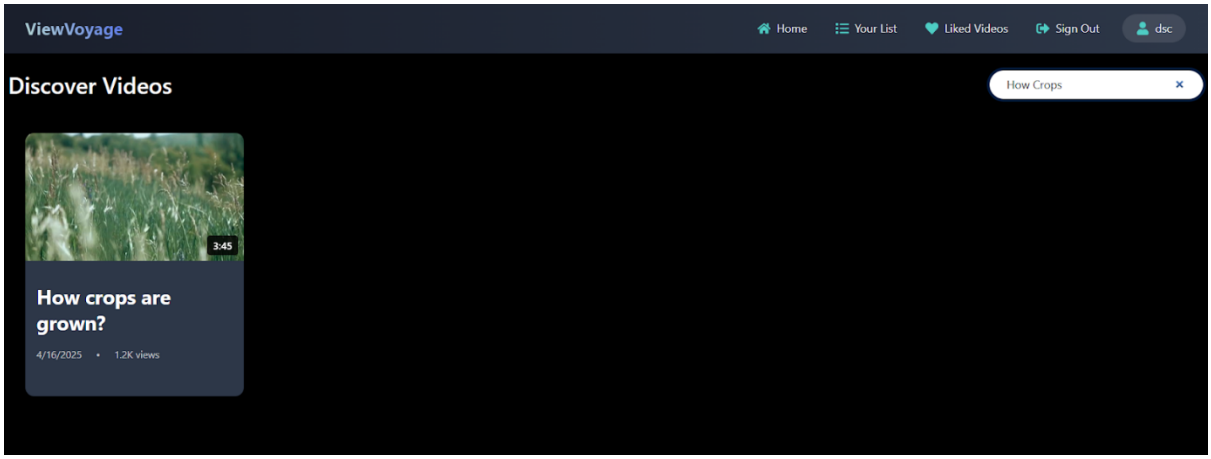

ViewVoyage
Home
Sign In
Sign Up
Create Account
Username
Choose a username
Email
Enter your email
Password
Create a password
Confirm Password
Confirm your password
Sign Up


ViewVoyage
Home
Your List
Liked Videos
Sign Out
dsc
0:00 / 0:12
Recommended Videos
How crops are grown?
4/16/2025 · 1.2K views
Learn Generalization
4/16/2025 · 1.2K views
Graphs and Algorithm
4/16/2025 · 1.2K views
Musily
4/16/2025 · 1.2K views


Nature
4/16/2025 · 1.2K views
Like
Save
Download
Share
Description
Feel the nature
Comments
Add a comment...
Post Comment
dsc
4/16/2025, 11:50:17 PM
Wow nice!!
Recommended Videos
How crops are grown?
4/16/2025 · 1.2K views
Learn Generalization
4/16/2025 · 1.2K views
Graphs and Algorithm
4/16/2025 · 1.2K views
Musily
4/16/2025 · 1.2K views


ViewVoyage
Home
Your List
Liked Videos
Sign Out
dsc
Liked Videos
Nature
Watch





8. ADVANTAGES & DISADVANTAGES

Advantages

- Simple, distraction-free video viewing experience
- User-friendly interface with essential features only
- Personalized sections like “Your List” and “Liked Videos”
- Fast search and lightweight performance
- Built using scalable MERN stack technologies

Disadvantages

- No support for user-generated content uploads (initial version)

- Lacks advanced features like playlists or video categories
- Limited moderation and admin tools for large-scale use
- Local storage may not handle large video volumes efficiently
- Not optimized for monetization or advertisements yet

9. CONCLUSION

ViewVoyage is a dynamic video-sharing platform that mimics popular services like YouTube, offering users an engaging experience with features such as video uploads, searching, and personalized lists (Your List and Liked Videos). The platform also incorporates interactive elements like video sharing, comments, downloads, and the ability to add videos to a personal collection. Built using modern web development technologies like React.js, Node.js, Express, and Bootstrap, ViewVoyage provides a seamless and visually appealing user experience.

10. FUTURE SCOPE

- **User Personalization:** Enhance the user experience by introducing AI-based recommendations based on user preferences, watching habits, and interaction history.
- **Live Streaming:** Add the capability to stream live events or sessions, which will engage a wider audience.
- **Mobile App:** Develop a mobile version of the platform using React Native for better accessibility on mobile devices.
- **Monetization Features:** Introduce ad integration, paid subscriptions, or video content monetization for content creators.
- **Enhanced Social Features:** Implement more social interactions such as following users, notifications, and sharing across social media platforms.
- **Advanced Search and Filters:** Implement more advanced search filters, allowing users to search by categories, date, or popularity.

- **Content Moderation:** Implement better moderation tools for uploaded videos to ensure compliance with community standards.

Appendix:

1. **Technologies Used:**
    - **Frontend:** React.js, HTML, CSS, JavaScript, Bootstrap
    - **Backend:** Node.js, Express
    - **Database:** (To be specified, e.g., MongoDB for video storage and user data)
    - **File Upload:** Use of local server storage for video uploads

GitHub & Project Demo Link

https://github.com/shashidhar-kittur/viewvoyage

https://drive.google.com/file/d/1R6InYN__iGMwGVj545-Hv7zjyfQXPA9I/view?usp=drive_link