InsightStream : Navigate The News Landscape

1. **Introduction**

* **Project Title:** Insightstream:Navigate The News Landscape
* **Team ID** : NM2025MID38906
* **Team Leader:**

|  |  |
| --- | --- |
| **NAME** | **MAIL ID** |
| KUMARAN S | [kumarans02102006@gmail.com](mailto:kumarans02102006@gmail.com) |

* **Team Members:**

|  |  |
| --- | --- |
| **NAME** | **MAIL ID** |
| MOHAN V | [mohan20102006@gmail.com](mailto:mohan20102006@gmail.com) |
| BHARATH R | [bharathmonish40@gmail.com](mailto:bharathmonish40@gmail.com) |
| MOHAN BABU N | [shallikissyou7@gmail.com](mailto:shallikissyou7@gmail.com) |

# PROJECT OVERVIEW

* 1. **Purpose: SK NEWS** connects clients and freelancers through project postings, bidding, and real-time communication.

InsightStream aims to make news consumption smarter, faster, and more reliable. With the overwhelming amount of news published daily, readers often struggle to find credible, unbiased, and relevant information. This project seeks to solve that problem by providing a platform that organizes, summarizes, and contextualizes news stories in real time.

Our goal is to help users navigate through the noise, identify the facts, and understand the bigger picture behind major events — all in one streamlined experience.

.

* 1. **Key Features**
     + **Smart News Summaries**

**AI-powered, concise summaries that deliver the core facts of any story in seconds.**

* + - **Topic Navigator**

**Lets users explore connected news stories through timelines, categories, and related events.**

* + - **Real-Time Updates**

**Continuously refreshes to show the latest verified news as it happens.**

* + - **Fact-Check Highlights**

**Flags questionable claims and links to trusted fact-checking sources.**

1. **Target Audience**

* General News Readers

People who want quick, reliable, and unbiased updates without scrolling through multiple sources.

* Students & Researchers

Those who need accurate, summarized news and context for assignments, projects, or research work

* Fact-Checkers & Critical Thinkers

Individuals who value transparency, credibility, and evidence-based reporting.

* Educators & Knowledge Seekers

Teachers and learners who want to use well-organized news to discuss current affairs in classrooms.

# ARCHITECTURE

## Frontend – React.js with Bootstrap and Material UI

The frontend of is built using **React.js**, ensuring a highly dynamic and responsive user interface. **Bootstrap** provides a robust grid system and layout framework, while **Material UI** adds modern, pre-designed components for consistency and visual appeal. Together, they deliver a smooth, intuitive, and interactive user experience across devices.

## Backend – Node.js and Express.js

The backend is powered by **Node.js** for fast, scalable, and event-driven server operations. **Express.js** is used to handle routing, server logic, and API endpoints, ensuring efficient communication between the frontend and database. This combination supports secure data processing and reliable application performance.

## Database – MongoDB

**MongoDB** serves as the primary database, chosen for its flexibility in handling unstructured data. It stores user information, project postings, application details, and chat messages in a secure and scalable manner. Its document-oriented structure makes it well- suited for real-time interactions and large-scale data management.

# SETUP INSTRUCTIONS

## Prerequisites

Before setting up **SK NEWS** ensure that the following tools and technologies are installed on your system:

* **Node.js** – JavaScript runtime environment.
* **MongoDB** – Database for storing user data, projects, and chat messages.
* **Git** – For cloning and version control.
* **React.js** – Frontend framework for building the user interface.
* **Express.js** – Backend framework for handling APIs and server logic.
* **Mongoose** – ODM (Object Data Modeling) library for MongoDB.
* **Visual Studio Code** – Recommended IDE for development.
  1. **Installation steps**
     1. **Node.js**
* **Knowledge**: basic JavaScript (variables, functions, promises/async-await), basic command-line use (terminal / PowerShell), JSON, and git (recommended).
* **System**: Windows / macOS / Linux (x64 or arm64); 2GB+ RAM recommended; internet to download installers.
* **Tools to have ready**: a terminal (CMD/PowerShell/WSL on Windows, Terminal on macOS/Linux), a code editor (VS Code recommended).

## Step-by-step installation

**Windows — easiest: official installer (or use npm-windows for version management) Official installer (quick)**

1. Visit the official Node.js downloads page and download the Windows **.msi** (choose the

**LTS** build).

1. Run the MSI and accept defaults (it will install node and npm and add to PATH).
2. Open PowerShell or Command Prompt and verify:

node -v npm -v

you’re done. nodejs.org

## If you want multiple Node versions / safer global installs — use npm for Windows

1. Install **npm-windows** (download and run the installer from the npm-windows releases).
2. After install, open a new admin PowerShell or CMD and use: npm install lts

npm list

npm use <version> # e.g. npm use 18.16.0 OR `npm use lts` if supported node -v

npm-windows lets you switch versions without re-installers. GitHub+1 npm -v

you’re done.

* + 1. **MongoDB**

## Knowledge Prerequisites

* Basic **database concepts**: collections, documents, CRUD (Create, Read, Update, Delete).
* Basic **command line/terminal** usage.
* JSON knowledge (MongoDB stores data in BSON, similar to JSON).
* (Optional) Networking basics: ports, IP binding, authentication.

## System Prerequisites

* **OS**: Windows (x64/arm64 supported).
* **Processor**: 64-bit, 1 GHz+ recommended.
* **RAM**: Minimum 2 GB (4 GB+ recommended for dev; higher for prod).
* **Disk space**: At least 5–10 GB free (MongoDB stores large amounts of data).
* **Port**: MongoDB runs on 27017 by default (make sure it’s free/unblocked).

## Software Prerequisites

* **Administrator** (to install packages/services).
* **Package manager** (apt, yum, Homebrew) OR direct installer.
* **MongoDB Shell (mongosh)** – required to interact with MongoDB.
* **MongoDB Compass (GUI)** – optional but recommended for beginners.
* **Docker** (optional) – for containerized setup.
  + 1. **Git**

## System prerequisites

* **OS**: Windows
* **Permissions**: Administrator (Windows)
* **Network**: Internet access (for cloning/pushing remote repos like GitHub, GitLab, Bitbucket).

## Software prerequisites

* Text/code editor (VS Code recommended).
* SSH client (optional, for GitHub/GitLab SSH connections).

## Github account creation:

1. Sign in to GitHub
   * Go to GitHub.
   * Log in to your GitHub account. If you don't have one, you can sign up for free.
2. Create a New Repository
   * Once logged in, click on the **+** icon in the top right corner of the page.
   * Select **"New repository"** from the dropdown menu.
3. Fill in Repository Details

You'll need to fill out the following fields:

* + **Repository Name**: Choose a name for your repository (e.g., my-project).
  + **Description** (optional): Provide a brief description of your project (e.g., "A cool web app").
  + **Public or Private**: Choose whether your repository will be public (anyone can see it) or private (only you and collaborators can access it).
  + **Initialize with README**: If you want to add a README file (which is helpful for explaining your project), check this box.
  + **Add .gitignore** (optional): You can choose a template for your .gitignore file, depending on your project's language or framework (e.g., Python, Node, etc.).
  + **Choose a License** (optional): You can choose a license if you'd like to specify how others can use your code.

1. Create the Repository
   * Once you've filled everything out, click the **Create repository** button.
2. Clone the Repository (Optional)
   * After the repository is created, you'll be taken to the new repository's page.
   * To clone it to your local machine, copy the **clone URL** (either HTTPS or SSH) from the "Code" button.
   * In your terminal, run the following command:
   * git clone <repository-url>
3. Start Adding Files
   * Now you can start adding files to your repository either through the GitHub website or by pushing files from your local machine.

That's it! You've created a new GitHub repository. Let me know if you need help with the next steps, like pushing files to GitHub.

* + 1. **React.js**

## System prerequisites

* + - **OS**: Windows

## Node.js & npm:

* + - Install latest **LTS version of Node.js** (includes npm).
    - Verify with:
    - node -v
    - npm -v
    - **Code Editor**: VS Code recommended.
    - **Browser**: Chrome/Edge/Firefox for developer tools.
    - A.Using Create React App (CRA) [Beginner Friendly]
    - Open terminal → navigate to project folder.
    - Run:
    - npx create-react-app my-app
    - (Here npx comes with npm 5.2+, so no extra install needed.)
    - Go into project folder:
    - cd my-app
    - Start development server:
    - npm start
    - App runs at [http://localhost:3000.](http://localhost:3000/)
    1. **Express.js – Mongoose – Visual Studio Code**

## Prerequisites (Theory)

* + - **JavaScript & Node.js** → needed since Express runs on Node.
    - **MongoDB knowledge** → Mongoose is an ODM for MongoDB.
    - **REST API concepts** → to design routes/endpoints.
    - **Visual Studio Code** → editor to write and manage the project.
    - **Node.js & npm installed** → to run server and install packages.
    - **MongoDB installed or Atlas account** → for database.

## Installation Steps:

# Clone the repository git clone

# Install client dependencies cd client npm install

# Install server dependencies cd

../server npm install

# FOLDER STRUCTURE

PROJECT STRUCTURE:

src/

├── App.jsx

├── main.jsx

│

├── context/

│ └── GeneralContext.jsx

│

├── components/

│ ├── Footer.jsx

│ ├── Hero.jsx

│ ├── HomeArticles.jsx

│ ├── NavbarComponent.jsx

│ ├── NewsLetter.jsx

│ └── TopStories.jsx

│

├── pages/

│ ├── Home.jsx

│ ├── CategoryPage.jsx

│ └── NewsPage.jsx

│

└── styles/

├── App.css (optional)

├── Footer.css

├── Hero.css

├── Home.css

├── HomeArticles.css

├── Navbar.css

├── NewsLetter.css

├── TopStories.css

└── CategoryPage.css

This structure ensures a clear separation between the **frontend (React)** and **backend (Node.js**

**+ Express)**, making development and maintenance more efficient.

# RUNNING THE APPLICATION

The document provides commands to run a frontend and backend application:

## Frontend:

* + - **cd client**: This command likely navigates to the frontend directory of the project.
    - **npm start**: Starts the frontend application using Node Package Manager (npm).

## Backend:

* + - **cd server**: Navigates to the backend server directory.
    - **npm start**: Starts the backend server using npm.

## Accessing the Application:

* The application can be accessed via [http://localhost:3000,](http://localhost:3000/) which is a local address where the frontend application can be viewed in a browser.

# API DOCUMENTATION

API documentation serves as a vital part of any application or service, especially when it comes to integrating with backend systems. It provides a clear and structured guide on how to interact with the application through its **API endpoints**. In this case, the API documentation appears to outline how to interact with user management, project handling, chat functionalities, authentication, and routing security. Here's a breakdown:

1. API Endpoints:

The endpoints listed in the documentation represent various functionalities provided by the backend of the application. Each endpoint corresponds to a specific action or resource. Here's a closer look at what the documentation suggests:

***User Endpoints:***

* + **/api/user/register**
    - **Purpose**: This endpoint is for registering new users. It likely requires some user data (e.g., email, password, name) to create a new account.
    - **HTTP Method**: Typically, a POST request is used here, sending user data to the server to create a new account.
    - **Response**: It could return a success message or an error message if the registration fails (e.g., email already exists).
  + **/api/user/login**
    - **Purpose**: This endpoint handles user login. The user submits credentials (email and password), and if the login is successful, a session token (such as a JWT) is returned.
    - **HTTP Method**: A POST request is commonly used here.
    - **Response**: The response typically includes a **JWT token** or a session identifier if the login is successful. If the credentials are invalid, an error message will be returned.

***Project Endpoints:***

* + **/api/projects/create**
    - **Purpose**: This endpoint allows users to create new projects within the system. It would typically involve sending project details (such as project title, description, and other parameters).
    - **HTTP Method**: A POST request, sending data in the body to create the project.
    - **Response**: Returns a confirmation message or the details of the newly created project.
  + **/api/projects/:id**
    - **Purpose**: Fetch details of a specific project based on its ID.
    - **HTTP Method**: A GET request is usually used to retrieve the details of the project.
    - **Response**: It returns the project data (such as title, description, and other attributes) or an error if the project ID doesn't exist.
  + **/api/projects/:id/apply**
    - **Purpose**: Allows a user to apply to a project, which means associating the user with the project in some way (e.g., applying for a job or requesting involvement).
    - **HTTP Method**: A POST request would be used to submit the application.
    - **Response**: Confirmation of the application or error if something goes wrong.

# USER INTERFACE

The **User Interface (UI)** section of a flavour diary application provides a description of how the user will interact with the app or website. It includes the structure, design elements, and features that allow users to search for recipes, view ingredients, and create a meal plan, among other actions.

* 1. **Landing Page**

**Overview**: The landing page is the first thing users see when they visit the flavour diary app. It should provide a welcoming experience, easy navigation, and access to key functionalities.

## Key Elements:

* Displays popular categories like “Breakfast,” “Lunch,” “Dinner,” or “Desserts.”

**Logo/Brand Name**: Clear visibility of the app’s name and logo.

* **Search Bar**: Allows users to search for recipes by ingredients, cuisines, or meal types.

## Category Navigation

* + **Featured Recipes**: A carousel or grid showcasing trending or recommended recipes.
* **8.2. News Search Page – InsightStream: Navigate the News Overview**

**This page allows users to search for news by keywords, categories, sources, or regions. It is designed to help users quickly find relevant and credible news articles on topics they care about.**

**key Elements**

* **Search Filters**

**Filters for news category (Politics, Sports, Technology, Business, Health), publication date, source credibility, and region.**

* **News Cards**

**Each news story is displayed with a headline, thumbnail image, short summary, and publication time. Clicking on a news card takes the user to the detailed article or summary page.**

* **Sort Options**

**Sorting options based on Latest, Most Read, Top Rated (Credibility), or Balanced View (showing multiple perspectives side by side).**

**key Elements**

* **Search Filters**

**Filters for news category (Politics, Sports, Technology, Business, Health), publication date, source credibility, and region.**

* **News Cards**

**Each news story is displayed with a headline, thumbnail image, short summary, and publication time. Clicking on a news card takes the user to the detailed article or summary page.**

* **Sort Options**

**Sorting options based on Latest, Most Read, Top Rated (Credibility), or Balanced View (showing multiple perspectives side by side).**

**8.3News Details Page – InsightStream: Navigate the News**

**Overview**

* **This page displays the full details of a selected news story, including its summary, key facts, related events, and sources. It helps users dive deeper into a topic with clarity and context.**

**Key Elements**

**Headline**

* **The main title of the news story, clear and bold.**

**Featured Image or Video**

* **A relevant image, video, or infographic providing visual context to the story.**

**Summary & Key Points**

* **A concise, AI-generated summary highlighting the most important facts and takeaways.**

**Full Article / Context**

* **The complete article text or a link to the original source for further reading.**

**Timeline / Background**

* **A mini-timeline or context section showing how the story developed over time.**

**Multiple Perspectives**

* **Side-by-side or linked views of how different outlets report on the same topic.**

**Source & Credibility Info**

* **Details about the news source, publication date, and credibility rating (if available).**

**User Comments & Ratings (Optional)**

* **A space for readers to share opinions, discuss the story, or upvote relevant perspectives.**

## Save to Favorites / Bookmark’

* **Option for users to save news articles or topics they want to revisit later. These saved stories appear in a personalized “Favorites” or “Reading List” section.**

UI Design Considerations:

* Visual Design

Colors: Use professional, neutral, and trustworthy colors like blue, gray, and white to reflect reliability, with accent colors (yellow/red) for breaking news alerts or highlights.

Typography: Choose clean, readable fonts for headlines and body text (e.g., bold for titles, regular for summaries) to make scanning easier.

Images & Media: Use high-quality images, infographics, or short videos that give quick visual context to the story and engage the reader.

1. User Experience (UX)

Responsive Design: Ensure the layout works seamlessly on desktop, tablet, and mobile for easy access anywhere.

Navigation: Provide a simple, intuitive navigation bar with quick links to Home, Categories, Search, Saved Articles, and Profile.

Interactive Elements: Use well-defined buttons for Save, Share, Read More, and Follow Topic. Smooth animations for carousels and timelines to enhance engagement.

Personalization: Show recommended stories based on user interests to keep them engaged.

1. Accessibility

Readable Layout: Maintain proper text contrast and font sizes to support users with visual impairments.

# TESTING

## Manual testing during milestones

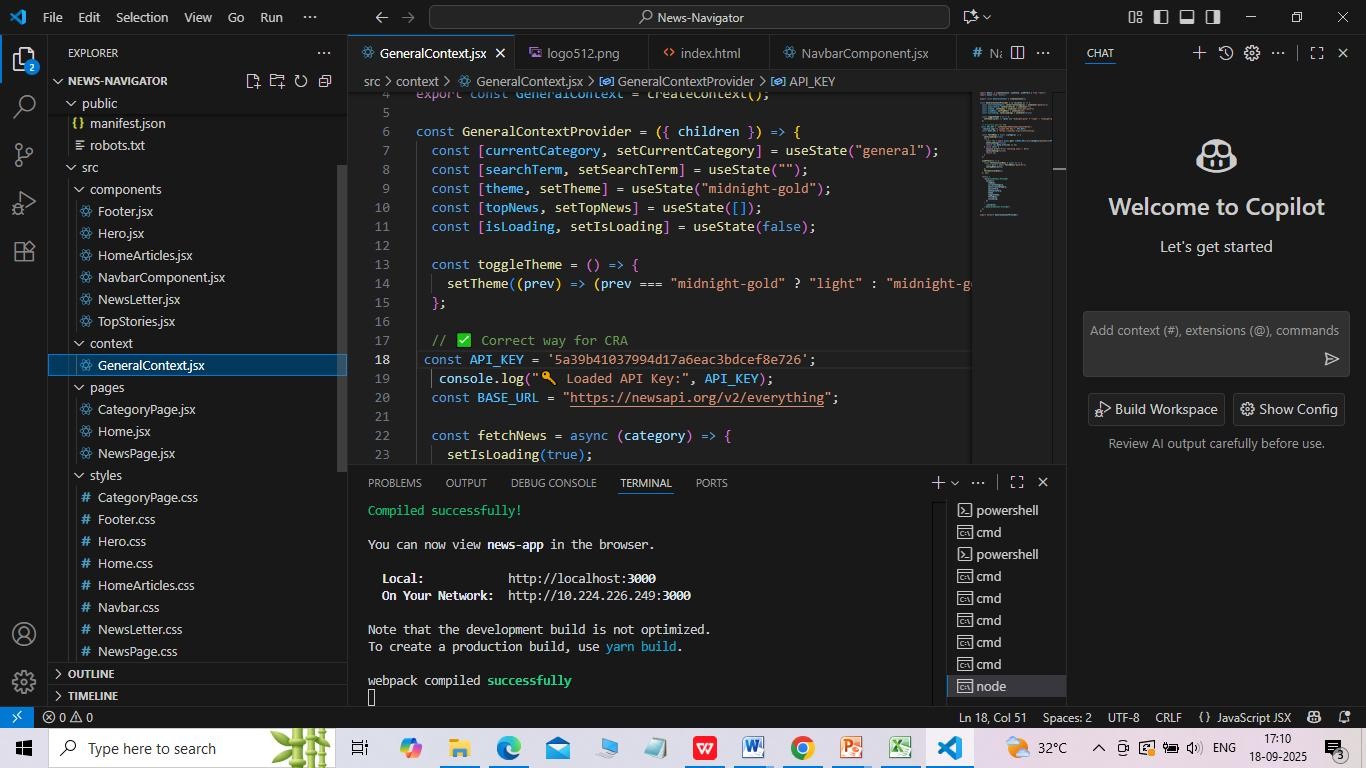
Manual Testing

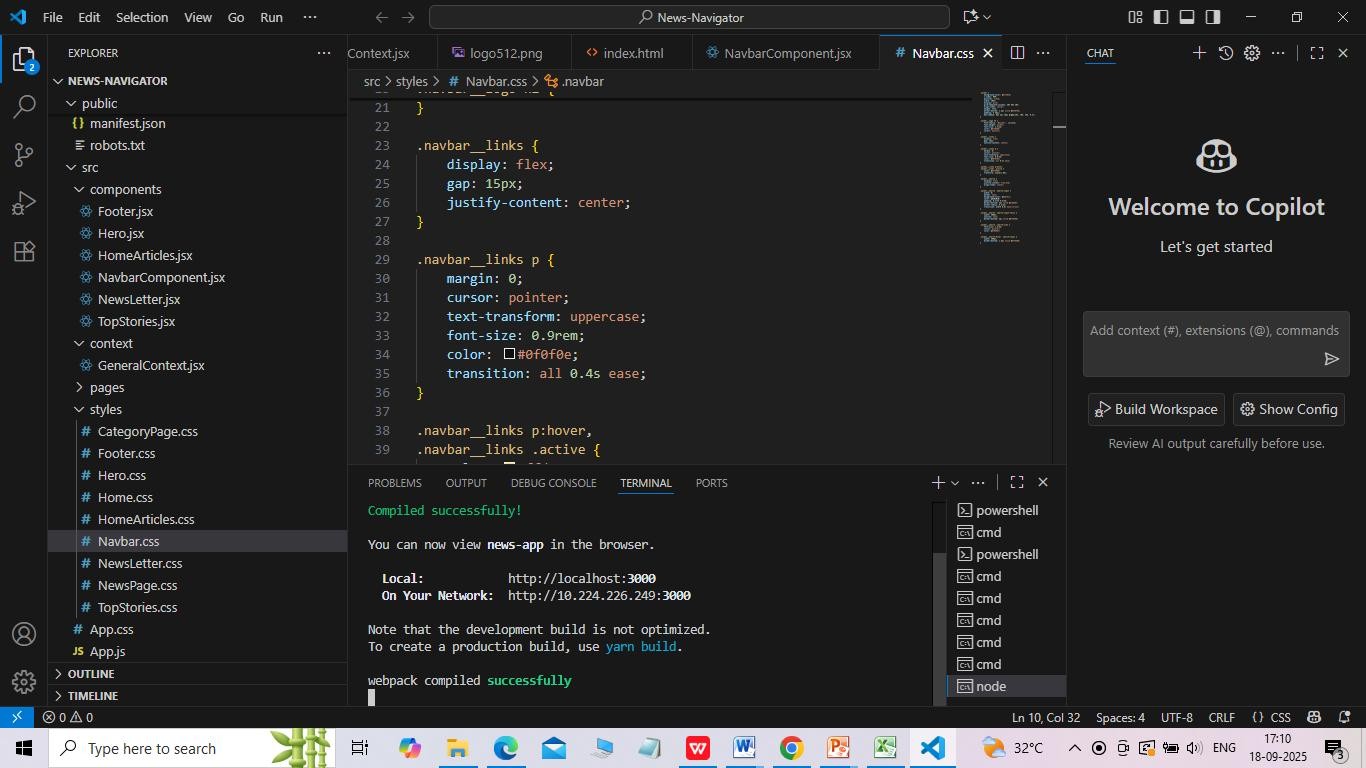
Manual testing is a type of software testing where human testers manually execute test cases to find defects and ensure the software meets its requirements. This process relies on human observation, creativity, and critical thinking.

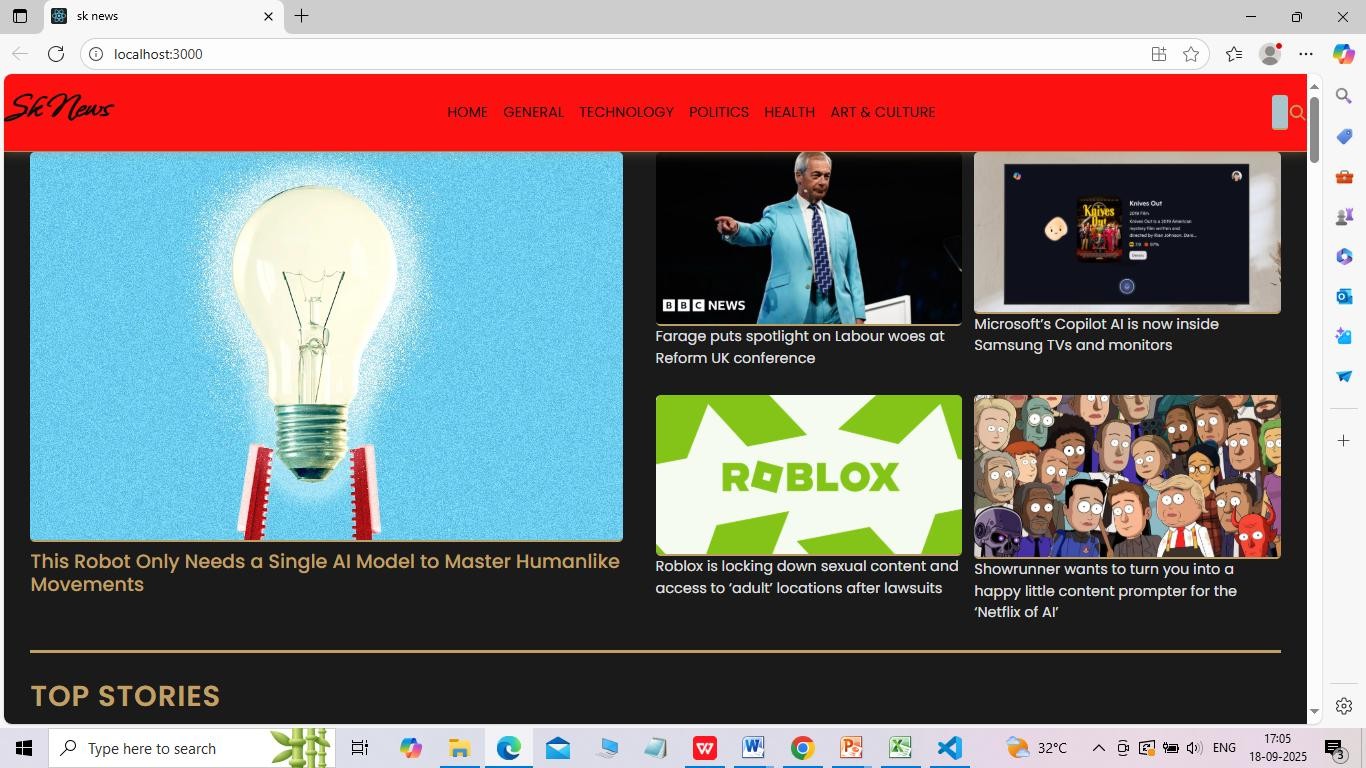
The general steps in manual testing include:

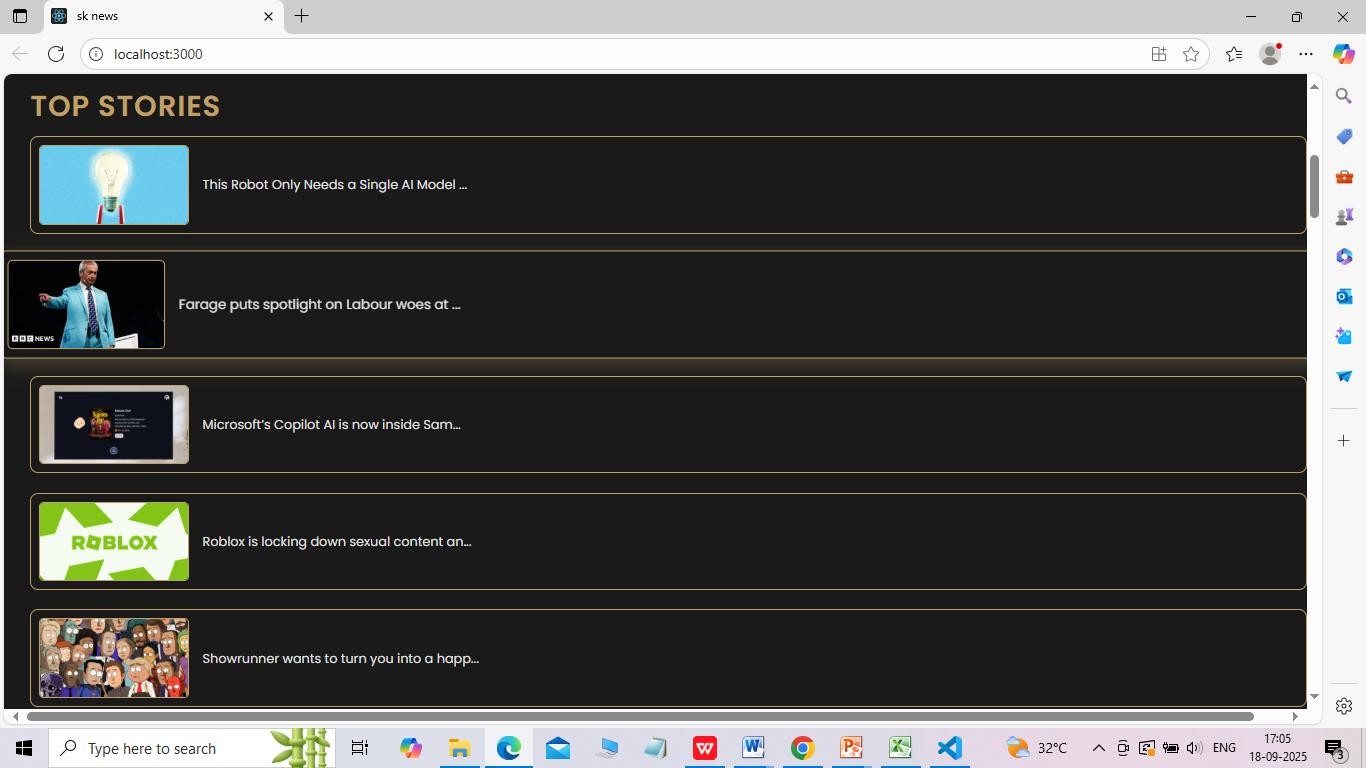
1. **Understanding Requirements**: Reviewing and understanding all software requirements, user stories, and design files.
2. **Creating a Test Pla**n: Developing a roadmap for the entire testing process, including objectives, scope, and resources.
3. **Writing Test Case**s: Creating detailed step-by-step instructions for testing specific features and scenarios.
4. **Setting up the Test Environment**: Installing the necessary software, databases, and other tools to perform the tests.
5. **Executing Test Cases:** Manually interacting with the software to verify it behaves as expected and documenting the results.
6. **Logging Defects:** Reporting any issues or bugs found during testing into a defect tracking tool.
7. **Retesting and Regression Testing:** Re-checking the areas where bugs were fixed and running regression tests to ensure new issues were not introduced.
8. **Analyzing Results and Reporting**: Reviewing the entire process, documenting findings, and creating a final report.

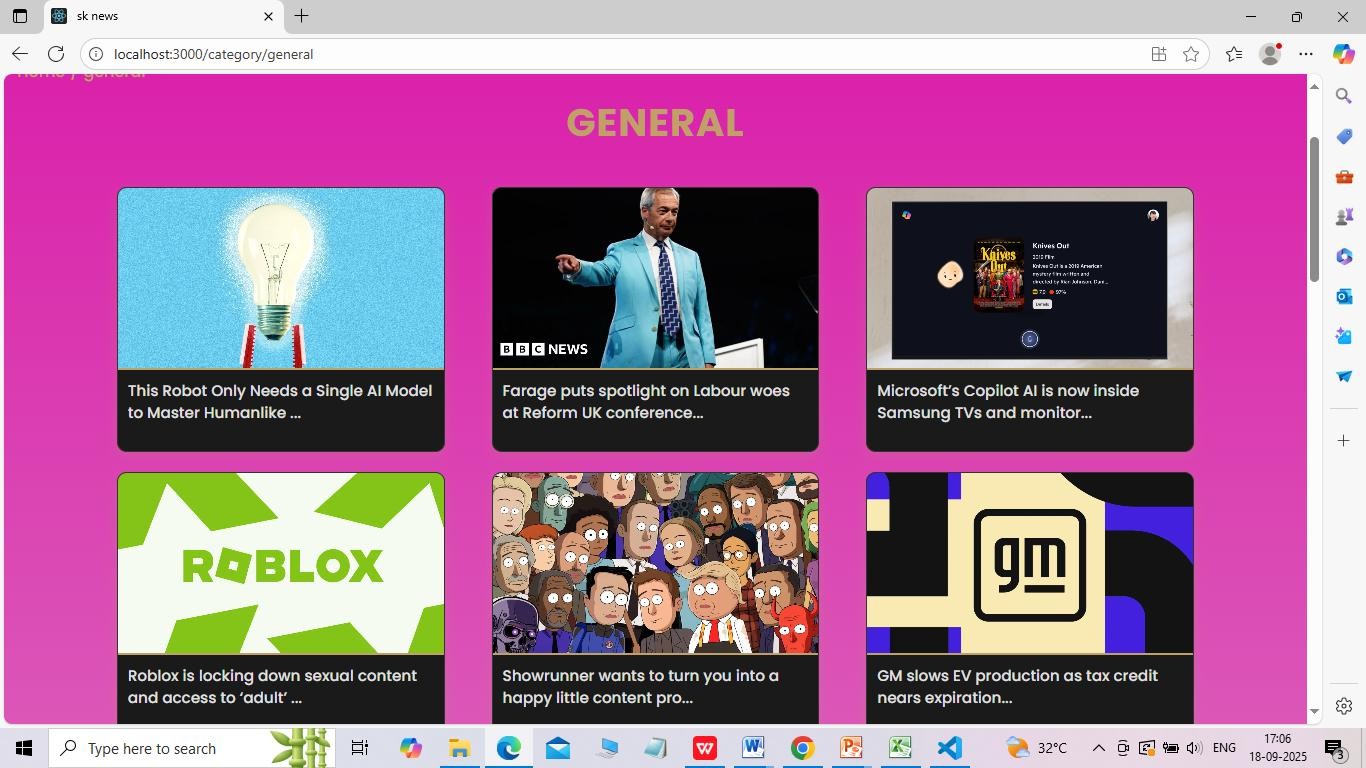
# SCREENSHOTS

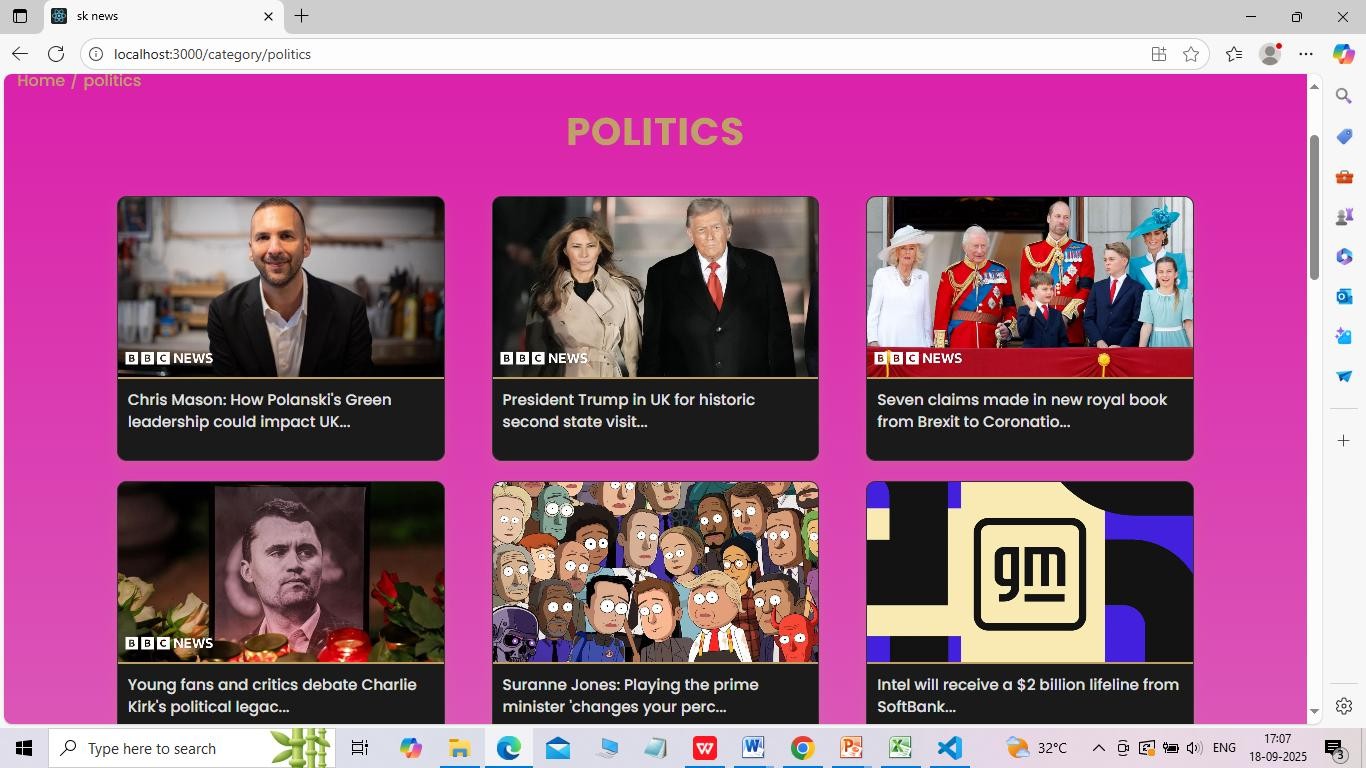
****

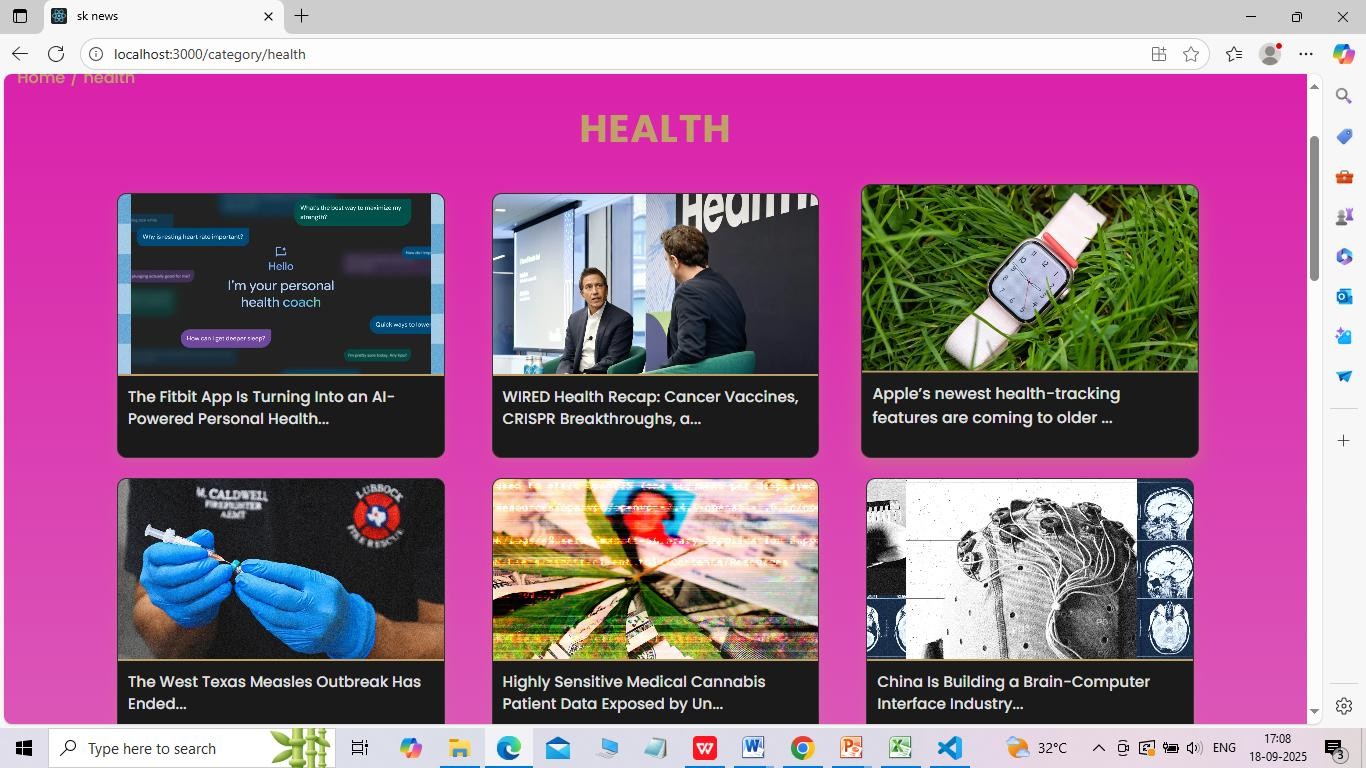
****

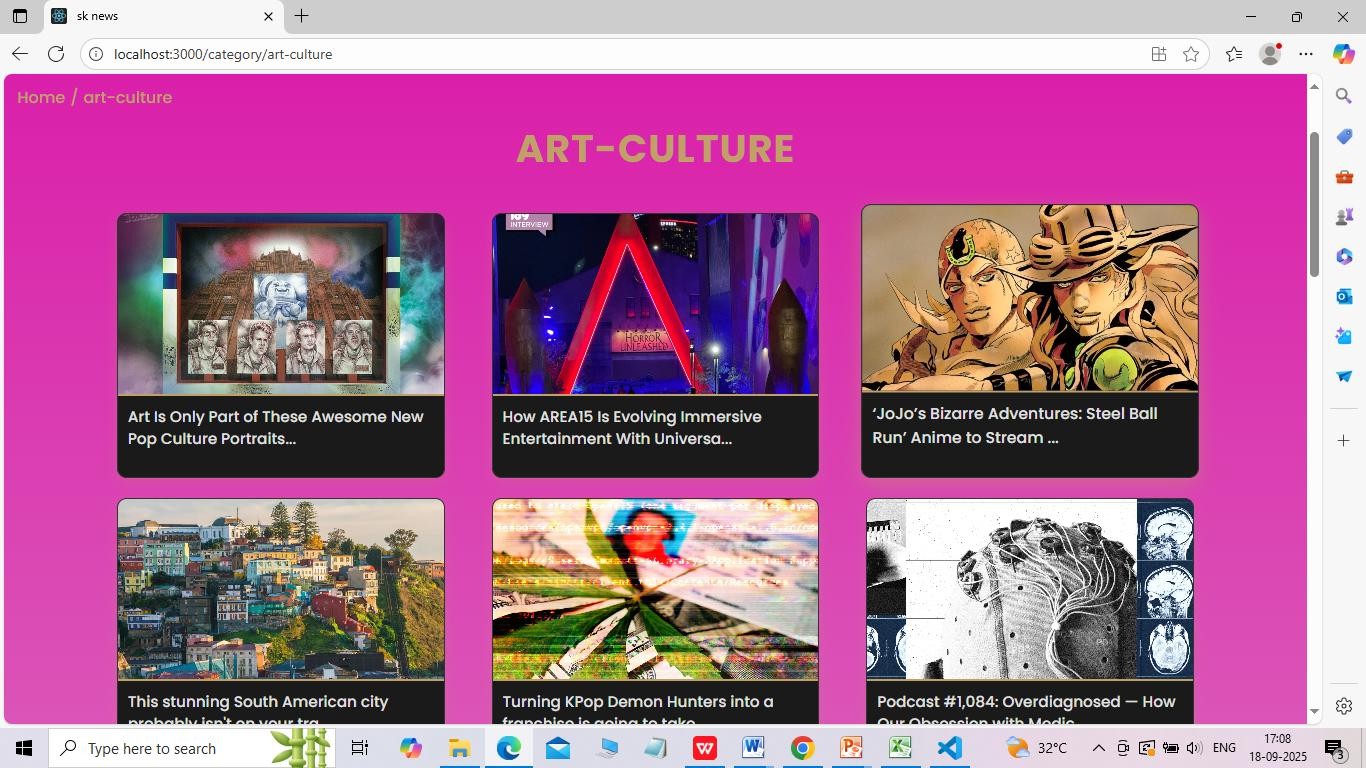


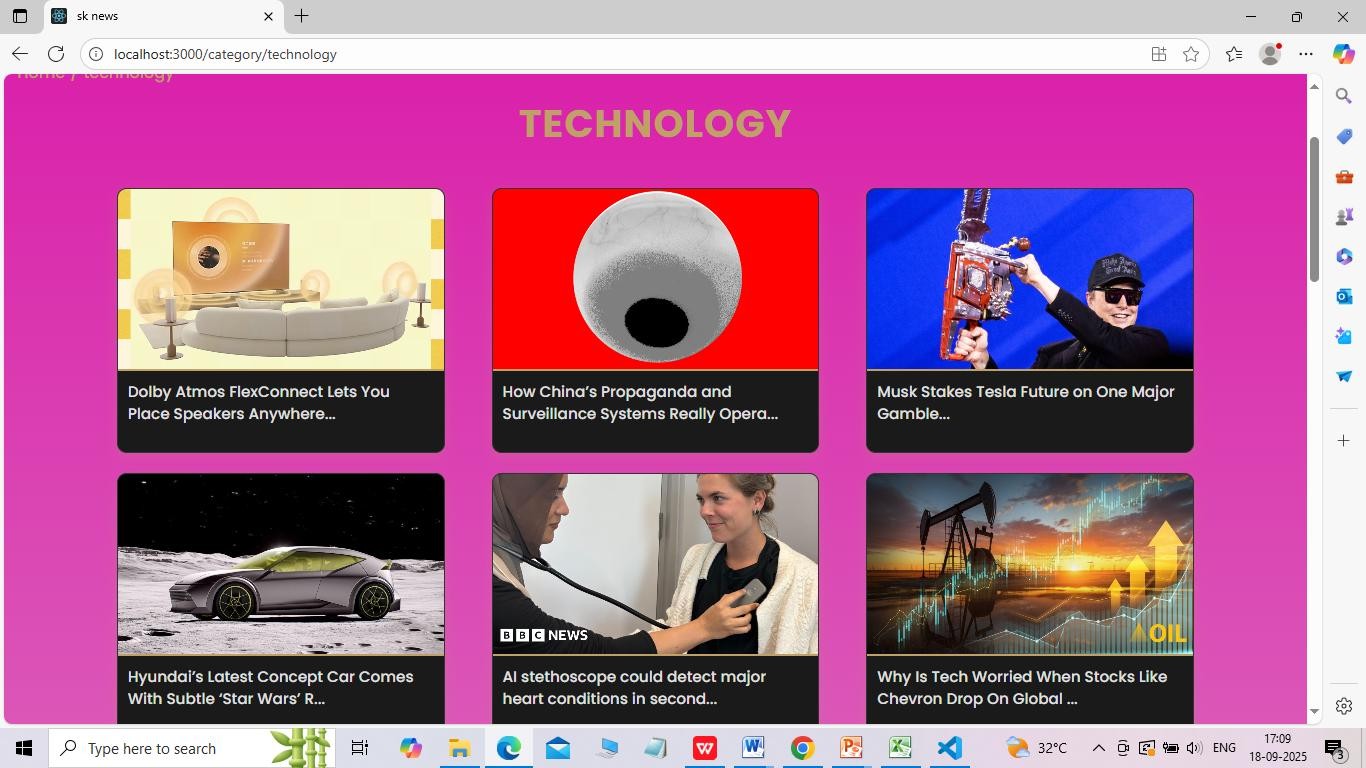
****



****



****



1. **KNOWN ISSUES :**

The News Navigate Landscape project faces several known issues that impact its overall performance and user experience. One major challenge is ensuring the reliability and neutrality of news sources, as misinformation and bias can affect the credibility of the platform. Duplicate or near-duplicate articles from multiple outlets often clutter the feed, making it necessary to implement effective deduplication and clustering. Frequent changes in news APIs or RSS feeds can break data fetching, leading to delays in updates, which is problematic for time-sensitive content. Balancing real-time updates with efficient caching is also a technical hurdle.

Personalization introduces additional complexity, as collecting user preferences must be done carefully to respect privacy regulations. Navigation can become overwhelming if too many categories or sources are presented without clear organization, while maintaining smooth performance across devices remains a priority. Finally, accessibility, content moderation, and ad placement require careful consideration to ensure an inclusive, safe, and user-friendly news experience.

A key concern is ensuring the credibility and neutrality of news sources, as the risk of misinformation, fake news, and biased reporting can damage user trust. Another major challenge is duplicate or similar articles appearing from multiple sources, which can clutter the interface and overwhelm readers, making deduplication and clustering essential. Technical difficulties also arise due to frequent changes or downtime in news APIs and RSS feeds, which can break data fetching or cause missing content. Ensuring timely updates while balancing server load and caching for performance is critical for a seamless user experience. Personalization and recommendation features must be implemented carefully to deliver relevant stories without breaching privacy regulations like GDPR or CCPA. Navigation can become confusing if categories, tags, and filters are not well-structured, leading to poor discoverability of content.

Mobile responsiveness and cross-platform performance present further challenges, particularly with heavy images, videos, and infinite scrolling that can slow down page loads. Accessibility is another known issue, as visually impaired users may face difficulties if alt text, keyboard navigation, and proper contrast are not supported. Ad placement, necessary for monetization, can sometimes interrupt the reading flow or degrade user experience. Additionally, content moderation

remains a constant requirement to filter harmful, offensive, or inappropriate material, while maintaining a balance to avoid unnecessary censorship. Security risks, such as malicious links in news feeds, also need to be addressed to protect users from phishing or malware. Finally, scalability and high-traffic handling are critical challenges as news traffic can spike unexpectedly during breaking events, requiring robust backend infrastructure to prevent downtime.

# FUTURE ENHANCEMENTS

In the future, the News Navigate Landscape project can be improved with several enhancements to increase usability, performance, and engagement. AI-driven personalization can be implemented to provide users with more relevant and diverse news recommendations while maintaining user privacy through anonymized data handling. Advanced article clustering and summarization techniques, powered by natural language processing (NLP), could group similar news stories together and generate short summaries, saving users time. A more intuitive navigation system with smart filters, trending topics, and keyword-based search could make content discovery seamless. Real-time notifications for breaking news, along with offline reading mode, can enhance user convenience. Multi-language support and automatic translation would allow a wider audience to access global news. Improvements in accessibility, such as screen reader compatibility, voice-based navigation, and adjustable font sizes, can make the platform more inclusive. Integration with social media sharing, bookmarking, and user-curated collections could increase engagement and retention. On the technical side, better caching, load balancing, and scalable cloud infrastructure can ensure faster updates and smooth performance even during high-traffic events.

Stronger content moderation powered by AI can be introduced to filter harmful or misleading information without human bias. Finally, introducing monetization options such as ad-free subscriptions, sponsored content, or affiliate links can make the platform financially sustainable without compromising the user experience.

AI-driven personalization can be implemented to provide users with more relevant and diverse news recommendations while maintaining user privacy through anonymized data handling. Advanced article clustering and summarization techniques, powered by natural language processing (NLP), could group similar news stories together and generate short summaries, saving users time. A more intuitive navigation system with smart filters, trending topics, and keyword-based search could make content discovery seamless. Real-time notifications for breaking news, along with offline reading mode, can enhance user convenience. Multi-language support and automatic translation would allow a wider audience to access global news. Improvements in accessibility, such as screen reader compatibility, voice- based navigation, and adjustable font sizes, can make the platform more inclusive. Integration with social media sharing, bookmarking, and user-curated collections could increase engagement and retention. On the technical side, better caching, load balancing, and scalable cloud infrastructure can ensure faster updates and smooth performance even during high-traffic events. Stronger content moderation powered by AI can be introduced to filter harmful or misleading information without human bias. Finally, introducing monetization options such as ad-free subscriptions, sponsored content, or affiliate links can make the platform financially sustainable without compromising the user experience.

# 13 :CONCLUSION

**To navigate the modern news landscape, a combination of technological adoption, critical thinking, and collaborative education is essential for both journalists and news consumers. By prioritizing authenticity, adapting to digital platforms, and fostering critical media literacy, society can remain informed and resilient in an age of information overload and rapid change.**