* PROJECT DOCUMENTATION

# image for nm project.jpg Insight Stream: Navigate The News Landscape

**Project Title:** Insight Stream: Navigate the News Landscape

Team ID: NM2025TMID36552

Team Leader:

* Bhavani. A- [24csbhavani011@gmail.com](mailto:24csbhavani011@gmail.com)

Team Members:

* Arthi. S[-24csarthi009@gmail.com](mailto:-24csarthi009@gmail.com)
* Aswini. J[-24csaswini010@gmail.com](mailto:-24csaswini010@gmail.com)
* Catherina Joyce. K.P - 24cscatherinajoyce012@gmail.com

# Project Overview

### Purpose

Insight Stream is a web-based platform designed to help users navigate the overwhelming amount of news available across various sources. The system aggregates, filters, and categorizes news articles in real time, offering personalized content recommendations. It ensures users stay informed with accurate, timely news while avoiding misinformation and irrelevant content.

The platform’s main objective is to simplify news consumption, making it efficient and tailored to individual preferences. It enables users to explore trending topics, follow specific categories, and access reliable sources — all in one interface.

## Benefits:

* Real-time news aggregation from multiple trusted sources.
* Personalized news feeds based on user interests.
* Efficient navigation with categories, tags, and search functionality.
* Alerts for breaking news and trending stories.
* Enhanced content discovery with filters, summaries, and recommendations.
* Avoids information overload by offering curated content.

## Key Features:

* News Aggregation: Collects news from APIs such as NewsAPI, RSS feeds, and other sources.
* Personalized Feeds: Allows users to select interests and topics for tailored recommendations.
* Categorization: News is sorted by categories like Technology, Politics, Health, Finance, etc.
* Search & Filters: Advanced search options with keywords, dates, and relevance sorting.
* User Authentication: Secure login with options for guest, registered, and admin roles.
* Bookmark & Share: Save articles and share them via social media.
* Analytics Dashboard: Provides insights into user engagement and trending topics.
* Notifications: Alerts for new stories, updates, or trending topics.

## Existing Solutions Comparison:

* Existing platforms like Google News, Flipboard, and Feedly offer comprehensive news aggregation and discovery features. However, Insight Stream focuses on:
* Simpler navigation tailored for diverse user groups.
* Customization at the article level, offering summaries and source comparison.
* Cost-effective usage with no unnecessary premium modules.
* Open-source integration allowing further enhancements by users.

## Architecture and Technology:



InsightStream is typically built as a modern web application, often utilizing a component-based architecture for a scalable and maintainable codebase.

* Frontend: The user interface is developed using a JavaScript library or framework like React.js. This allows for a dynamic and interactive user experience.
* State Management: State management is handled through a system like Context API (for smaller projects) or Redux (for larger, more complex projects) to manage global data, such as user preferences, bookmarks, and search results.
* Routing: React Router is used to handle navigation between different pages, such as the home page, category pages, search results, and bookmark lists.
* Styling: A CSS framework like Tailwind CSS is often used for a fast and efficient design process, enabling the creation of a modern and responsive user interface.
* Data Source: The application fetches
* Backend: A backend can be implemented to handle more complex functionalities such as user authentication, data processing, or personalized recommendations, possibly using technologies like Node.js and a database like Redis for real-time data handling.

.

## Set-up Instructions:

The "InsightStream" project is a web application that requires a specific environment to be set up. Follow these steps to get the project running locally on your machine.

* Prerequisites: Before you begin, ensure you have Node.js and npm (Node Package Manager) installed. You can download the latest LTS (Long-Term Support) version of Node.js from the official website.
* Clone the Repository: Use Git to clone the project repository from a hosting service like GitHub. Open your terminal or command prompt and run: git clone <repository-url>
* Navigate to the Project Directory: Once the cloning is complete, change your current directory to the project folder: cd insightstream-news-app
* Install Dependencies: The project relies on various libraries and frameworks. Install all the necessary packages by running the following command in the project's root directory: npm install
* Create an Environment File: The application may require API keys for external services (like a news API). Create a file named .env in the root of your project and add the required variables. For example: REACT\_APP\_NEWS\_API\_KEY=your\_api\_key\_here

### Core Programming Languages

The "InsightStream" project is built using a modern technology stack that relies heavily on a few key languages:

* JavaScript: The primary language for both the frontend and backend. It powers the interactive elements of the user interface and handles server-side logic.
* HTML: Used for structuring the content and building the basic layout of the web pages.
* CSS: A stylesheet language used to design and style the frontend, making it visually appealing and responsive.

### Development Tools & Concepts

A number of tools and concepts are essential for developing and maintaining the project:

* Node.js: A JavaScript runtime environment that allows you to run JavaScript code outside of a web browser. It's the foundation for the project's backend.
* npm (Node Package Manager): Used to install, manage, and share code packages (dependencies) for the project.
* API (Application Programming Interface): The project interacts with an external API (e.g., a news API) to fetch real-time news data.
* React Router: A standard library for routing in React applications, enabling navigation between different pages without a full page reload.
* Context API or Redux: Used for state management, which is the process of managing and sharing data that changes over time across different components of the application.

### Backend-Specific Prerequisites

While the core project is focused on the frontend, some functionalities may require a backend. If a backend is included, these are the prerequisites:

* Node.js Runtime: The backend server runs on Node.js.
* Express.js (Commonly): A fast, unopinionated, minimalist web framework for Node.js, often used to build the backend API.
* Database: Depending on the project's complexity, a database like MongoDB or Redis may be required to store user data, bookmarks, or other persistent information.

### Frontend-Specific Prerequisites

The frontend is the user-facing part of the application. These are the main prerequisites for building the frontend:

* React.js: A popular JavaScript library for building user interfaces. It's the core framework for the project's frontend.
* CSS Framework (e.g., Tailwind CSS): A utility-first CSS framework for rapidly building custom user interfaces. It's often used to style the components.

# Integrated Development Environment (IDE)

While you can use any text editor, an IDE provides a more streamlined development experience.

* Visual Studio Code (VS Code): This is the most recommended IDE for JavaScript projects. It offers powerful features like intelligent code completion, syntax highlighting, and integrated terminal. It can significantly boost your productivity.

### Folder Structure

A well-organized folder structure is crucial for project maintainability. A typical structure for this project looks like this:

/insightstream-news-app

├── node\_modules/

├── public/

│ ├── index.html

│ └── ...

├── src/

│ ├── assets/ (images, icons, etc.)

│ ├── components/ (reusable UI components like buttons, cards)

│ ├── pages/ (main page components like Home, Categories)

│ ├── App.js

│ ├── index.js

│ └── ...

├── .env

├── package.json

├── package-lock.json

└── README.md

* src: Contains all the source code for the application.
* components: Holds the reusable, smaller UI parts (e.g., Header.js, NewsCard.js).
* pages: Contains the main page components that are rendered by the router (e.g., HomePage.js, BookmarksPage.js).
* assets: Stores static files like images and fonts.
* .env: Stores environment variables.
* package.json: Lists all project dependencies and scripts.

### Running the Application

## Frontend :

### Cd client

* In the Insight Stream application, the frontend is responsible for presenting the news content

to users in an interactive and visually appealing manner.

* The command cd client is used to navigate into the directory that contains the frontend part of the project. This directory typically holds all the source code, configuration files, stylesheets, and components necessary to build and run the user interface.
* By entering the client directory, developers can access the tools and files required to install dependencies, configure environment variables, and start the development server. This separation of concerns—placing the frontend in its own folder—helps organize the project structure, making it easier to manage the user interface independently from the backend logic.

### Npm start

* In the Insight Stream application, once you are inside the frontend directory (commonly named client), the command npm start is used to launch the development server for the user interface. This command initiates the process of building and rendering the application in a local environment, allowing developers and users to interact with it through a web browser.
* When npm start is executed, it reads the configuration from the project’s setup files, installs necessary modules if required, compiles the application’s source code, and opens a local server—usually at . The server dynamically updates the interface as changes are made to the code, providing a smooth and efficient development experience.

## User interface:

The user interface of Insight Stream: Navigate the News Landscape is designed to be clean, intuitive, and user-friendly. It provides a personalized dashboard where users can explore top headlines, filter news by category, and view AI-powered insights like summaries, sentiment analysis, and related stories. With features such as timelines, map views, and topic connections, the interface helps users easily navigate complex news landscapes while maintaining a modern, minimal design for clear readability and smooth interaction.

### Landing Page

The landing page is the first interface that users see when they open the Insight Stream application. It is designed to be clean, engaging, and user-friendly, providing quick access to news content and navigation tools.

### Key Features:

* Header Section:

Displays the app logo and name “Insight Stream”.

Contains navigation links such as Home, Categories, About, and Login/Register.

* Search Bar:

Prominently placed at the top for users to search for news articles by keywords or topics.

* Main Banner:

Features top trending news stories with engaging visuals.

May include a slideshow format or highlighted news section to draw user attention.

* News Categories:

Organized into sections like Technology, World, Health, Business, Sports, etc.

Allows users to explore news based on their interests easily.

* . Recommended News:

Personalized or trending news recommendations displayed below categories.

* Footer Section:

Contains links to Privacy Policy, Terms of Service, Contact, and social media platforms.

### User Dashboard:

The User Dashboard is the heart of Insight Stream, designed to give readers a personalized, clear, and interactive view of the news world. It transforms scattered headlines into a well-organized, insight-driven hub, helping users not just read the news but also understand and navigate the landscape effectively.

1. Centralized News Hub

The dashboard acts as a one-stop space where users can instantly access:

* Top headlines and breaking news.
* Trending topics across regions and categories.
* Curated insights powered by AI for faster understanding.

This ensures users don’t miss important updates while avoiding information overload.

2. Personalized Experience

Every user sees a customized news feed (My Stream) based on their selected interests, browsing history, and saved topics. This personalization includes:

* Suggested articles tailored to reading preferences.
* Notifications and alerts for specific subjects (e.g., elections, markets, sports).
* A “Saved Stories” section for later reading.

3. Easy Navigation & Filters

The dashboard offers intuitive ways to explore the news:

* Category tabs (Politics, Business, Technology, Science, Sports, World, etc.).
* Filters for region, source, and time frame (24 hours, week, month).
* Search bar for keywords, people, or events.

This structure allows users to quickly drill down to the news they care about most.

4. Insightful News Presentation

Unlike traditional dashboards that only show headlines, Insight Stream enhances stories with:

* AI-generated summaries for quick takeaways.
* Sentiment indicators (positive, negative, neutral).
* Impact tags showing how the story affects economy, society, or policy.
* Related stories that connect events and provide broader context.

5. Visualization Tools

To make the news landscape more interactive and understandable, the dashboard provides:

* Timeline view – how a story develops over time.
* Map view – location-based exploration of global/local events.
* Network graph – connections between people, organizations, and issues.

These tools transform raw information into visual insights.

6. User Tools & Settings

The dashboard also empowers users with:

* Profile settings for managing interests and preferences.
* Notification controls for breaking news.
* Options to receive daily/weekly news digests.

7. Design & Usability

* Clean layout with card-based structure for readability.
* Minimal and modern UI with soft colors and accent highlights.
* Smooth navigation through clear menus, icons, and shortcuts.
* Mobile-friendly design so users can access insights anytime.

## Admin Panel:

The Admin Panel is the backend control center of Insight Stream, built for administrators and content managers to manage news sources, oversee user activity, maintain platform integrity, and optimize performance. While the User Dashboard focuses on personalized news consumption, the Admin Panel ensures that the system runs smoothly, securely, and efficiently.

1. Content Management

* The Admin Panel allows administrators to:
* Add, edit, or remove news sources (publishers, websites, APIs).
* Manage categories (Politics, Business, Tech, Sports, etc.).
* Tag and classify stories for better AI summarization and filtering.
* Approve or flag articles that may require editorial review.

2. User Management

Admins can oversee all registered users with options to:

* View user profiles, activity logs, and reading preferences.
* Manage user roles (reader, editor, moderator, premium subscriber).
* Handle account issues (password resets, bans, warnings).
* Monitor saved articles and subscription patterns.

3. Analytics & Insights

The Admin Panel provides detailed statistics about platform usage:

* User engagement analytics (most read categories, peak activity times).
* Traffic reports by region, device, and source.
* Content performance (top-performing stories, most shared topics).
* AI performance tracking (accuracy of summaries, sentiment detection).

This helps admins refine content strategies and improve personalization.

4. Moderation & Security

To maintain trust and safety, the panel includes:

* Content moderation tools to detect fake news, spam, or inappropriate material.
* Automated alerts for suspicious activity or unusual traffic spikes.
* Role-based access control to ensure only authorized admins make changes.
* Data privacy settings to comply with regulations (GDPR, etc.).

5. System Configuration

Admins can control how the platform operates by:

* Setting up API integrations with news providers.
* Adjusting AI models and filters for summaries and insights.
* Managing notification systems (push alerts, email digests).
* Customizing themes, branding, and layout options for users.

6. Monetization & Subscription Control

If the platform includes premium features:

* Manage subscription plans (free, premium, enterprise).
* Track payments, renewals, and cancellations.
* Offer promotions or discounts.
* Generate revenue reports.

7. Design & Usability

* Dashboard Layout: A clean, card-based overview with quick stats (users, articles, activity).
* Sidebar Navigation: Clear menus for Users, Content, Analytics, Security, Settings.
* Charts & Graphs: Visual analytics for easy interpretation.
* Responsive Design: Works smoothly on desktop and tablet for admin use.

## Testing :

Testing ensures that the platform works smoothly, delivers accurate news insights, and provides a seamless experience for both users and administrators. It covers different layers of functionality, performance, and usability.

### Manual Testing during Milestone

In the Insight Stream: Navigate the News Landscape project, manual testing plays a key role at each milestone stage. It ensures that newly developed features work correctly before moving to the next phase of development.

Purpose:

* To verify functionality, usability, and reliability of features at milestone checkpoints.
* To identify bugs early, reducing risks of failures in later stages.
* To ensure the system aligns with project requirements and user expectations.

Milestone 1: Backend API Development

At this stage, the focus is on testing whether the APIs (which connect the backend with the system) are working correctly.

Things to Test Manually

* API Connectivity – Check if the API responds when called (no broken links).
* Data Accuracy – Verify the API returns correct and complete news data.
* Response Time – Make sure responses are quick and not too slow.
* Error Handling – Test invalid requests (e.g., wrong keywords) to see if proper error messages appear.
* Security – Confirm that only authorized users can access the APIs.
* Integration Check – Ensure the API data can be read by the dashboard/frontend later.

Example Test Cases

* Call the login API with valid and invalid credentials.
* Request news articles by topic (e.g., “Technology”) and check if results are correct.
* Send a wrong request (e.g., missing parameters) and verify if a clear error message is shown.
* Check if the API returns results within 2–3 seconds.

Milestone 2: Frontend–Backend Integration

This milestone checks whether the frontend (user interface) and backend APIs are working together smoothly. The goal is to make sure data from the backend appears correctly on the user dashboard.

Things to Test Manually

* API Data Display – Verify that news articles fetched by APIs are shown properly in the UI.
* Login & Authentication – Check if login works with backend validation.
* Filters & Search – Confirm that when a user searches or applies filters, the frontend correctly fetches and displays backend results.
* Error Messages \_ Make sure user-friendly error messages appear (e.g., “No results found” instead of technical errors).
* Data Sync – Ensure updates in the backend (new articles, changes) appear instantly or with minimal delay on the frontend.
* Navigation Flow – Test that moving between sections (Dashboard → Categories → Saved News) still retrieves correct backend data.

Example Test Cases

* Log in with valid credentials → User should enter dashboard.
* Log in with wrong credentials → Proper error should show.
* Search “Sports” → Only sports news should display.
* Apply filters (e.g., Date, Category) → Results should change accordingly.
* Disconnect backend temporarily → UI should show “Unable to fetch data” instead of breaking.

Milestone 3: Complete User Interface (UI) Functionality

This milestone ensures that the entire UI works smoothly and provides a good experience for users. Now the focus is on design, navigation, and usability in addition to backend integration.

Things to Test Manually

* Navigation – Check if all menus, buttons, and links work correctly.
* Dashboard Display – Verify that news articles, summaries, and categories appear neatly.
* Search & Filters – Ensure users can search topics and apply filters easily.
* User Actions – Test saving, bookmarking, and sharing news items.
* Error Handling – Confirm that clear, simple messages are shown for empty searches, missing data, or failed actions.
* Responsive Design – Check if the UI works well on desktop, tablet, and mobile.
* Consistency – Fonts, colors, and layouts should be uniform across all pages.
* Performance – UI should load fast and display results without lag.

Example Test Cases

* Open dashboard → All top news should display correctly.
* Click on a news item → Full article/summary should open.
* Apply “Technology” filter → Only tech news should appear.
* Save a news article → It should show in “Saved/Bookmarks” section.
* Resize the browser window or use mobile → Layout should adapt without breaking.
* Try searching with no keyword → System should show “Please enter a search term.”

## Testing Tools in Detail

Testing tools help ensure the platform is reliable, secure, and user-friendly. They support both manual and automated testing, making the process faster and more accurate.

1. Postman (API Testing)

* Purpose: Used for testing backend APIs.
* Usage:
* Send requests to APIs (GET, POST, PUT, DELETE).
* Verify responses, status codes, and error handling.
* Create automated API test collections.
* Benefit: Ensures backend APIs are working correctly before integration.

2. Selenium (UI Automation Testing)

* Purpose: Automates browser-based testing of the frontend UI.
* Usage:
* Simulates user actions (clicks, searches, filters).
* Validates navigation, layouts, and user flows.
* Supports cross-browser testing.
* Benefit: Saves time by running repetitive UI tests automatically.

3. JMeter (Performance & Load Testing)

* Purpose: Tests system performance under different loads.
* Usage:
* Simulate hundreds or thousands of users.
* Measure response times, throughput, and server stability.
* Benefit: Ensures the system can handle high traffic without crashing.

4. Jira (Test Management & Bug Tracking)

* Purpose: Organizes test plans and tracks bugs.
* Usage:
* Report issues with screenshots and logs.
* Track bug status (Open, In Progress, Resolved).
* Manage test cases linked to milestones.
* Benefit: Improves collaboration between testers and developers.

5. TestRail (Test Case Management)

* Purpose: Manages structured test cases and execution.
* Usage:
* Write and organize manual/automated test cases.
* Record results for each milestone.
* Generate reports on test coverage and pass/fail rates.
* Benefit: Provides clear visibility of testing progress.

6. SonarQube (Code Quality & Security Testing)

* Purpose: Checks code for bugs, vulnerabilities, and maintainability.
* Usage:
* Run static code analysis.
* Detect security flaws and poor coding practices.
* Benefit: Improves backend code quality and reduces risks.

7. Browser Developer Tools (Chrome DevTools, Firefox DevTools)

* Purpose: Quick frontend debugging and testing.
* Usage:
* Inspect elements, CSS, and layouts.
* Monitor API calls and console errors.
* Test responsive design on different devices.
* Benefit: Essential for fast debugging during UI development.

8. Appium (Mobile Testing) (if mobile app is included)

* Purpose: Automates testing for Android and iOS apps.
* Usage:
* Run test scripts on real or virtual devices.
* Verify navigation, gestures, and mobile UI.
* Benefit: Ensures smooth mobile user experience

## Screenshot:





