**News Landscape**

## 1. Introduction

# 1. Project Title: News Landscape

The project News Landscape is a full-stack web application developed to present the latest news articles in a structured, attractive, and user-friendly format. In today’s fast-moving digital world, users expect quick access to reliable information, and this project aims to deliver exactly that.

News Landscape combines the power of React.js for frontend development and Node.js with Express for backend services. The frontend provides a responsive and modern user interface where news is displayed in neatly organized cards. Each card shows the article’s title, description, image, and publication date, ensuring that readers can quickly scan through news highlights.

The backend server is responsible for handling API requests. It acts as a bridge between the frontend and external news data sources. By using this structure, the project ensures separation of concerns and makes the system scalable and easy to maintain.

One of the key goals of News Landscape is to demonstrate how reusable components in React can be used to build a dynamic and scalable user interface. The NewsCard component is a prime example, designed to handle multiple data points efficiently while maintaining a consistent design.

The project emphasizes simplicity and clarity. Users are not overwhelmed with cluttered designs or unnecessary information. Instead, the application focuses on delivering news in a straightforward way, optimized for readability on desktop, tablet, and mobile devices.

From a development perspective, News Landscape highlights important aspects of frontend-backend integration, API handling, and state management in React. It is a practical project for understanding how to structure modern web applications.

In addition, this project serves as a foundation for further improvements. Features like search functionality, category-based filtering, user login, and article bookmarking can be easily added due to its modular design. The use of React also opens doors for implementing advanced concepts like Context API or Redux for managing global state as the application grows.

By working on this project, developers gain hands-on experience in creating responsive interfaces, handling real-time data, and ensuring seamless communication between frontend and backend. It also underlines the importance of writing clean, maintainable code that can be extended for future needs.

Ultimately, News Landscape is not just a news app but also a demonstration of how modern technologies can be combined to build meaningful, real-world applications. It reflects the essence of frontend development with React, supported by backend logic, to deliver a complete and functional web solution.

# 2.TEAM ID :

• NM2025TMID37632

# 3.TEAM LEADER:

Sameera.H (ruuhi676@gmail.com)

# 4.TEAM MEMBERS:

1.Soniya.G (soniyang48@gmail.com)

2.Velankanni Asviya . R (velankanniasviya07@gmail.com)

3.Ramya.R (ramyaramz3103@gmail.com)

# 2.Project Overview

## • PURPOSE:

1️⃣ The homepage of *News Landscape* displays news articles in a clean and organized card-based layout 📰.

2️⃣ Each news card contains a title, short description, image, and date to give users quick insights 📅.

3️⃣ A navigation bar at the top helps users explore different sections or categories with ease 🧭.

4️⃣ The design is fully responsive, adapting smoothly to desktops, tablets, and mobile screens 📱💻

.  
5️⃣ The interface follows a minimal and modern style, ensuring focus on readability and user comfort 🎨.

6️⃣ News data is fetched dynamically from the backend, making the content fresh and up to date 🔄.

7️⃣ The project highlights the integration of frontend (React.js) and backend (Node.js/Express) 🔗.

8️⃣ The consistent styling across pages is achieved using CSS with Flexbox/Grid layouts 🎯.

9️⃣ Clicking on an article can be extended to show detailed information or external links 🌐.

🔟 The application is lightweight, fast-loading, and provides a smooth browsing experience ⚡.

1️⃣1️⃣ The output demonstrates real-world news portal features that can be scaled further 🚀.

1️⃣2️⃣ The project not only looks professional but also proves practical knowledge of React components and API handling 👨‍💻.

1️⃣3️⃣ The app provides a structured flow of information, avoiding clutter and distractions 📖.

1️⃣4️⃣ It is designed to be beginner-friendly for developers, making it easy to maintain and expand 🛠️.

1️⃣5️⃣ Overall, the output reflects a complete, working prototype of a news application with scope for future improvements 🌟.

1️⃣6️⃣ The interface ensures **fast navigation** between different sections, reducing user effort ⏩.

1️⃣7️⃣ News cards use **visual elements like images** to improve engagement and make the content attractive 🖼️.

1️⃣8️⃣ The design highlights **modularity**, where components like NewsCard can be reused across pages 🔄.

1️⃣9️⃣ The output demonstrates a **balance between functionality and design**, making it suitable for real-world deployment ⚖️.

2️⃣0️⃣ The project showcases a **solid foundation for adding advanced features** like search, filtering, authentication, and dark mode 🔮.

## • Features:

# 1. Personalized News Feed: A curated feed of news articles tailored to user interests.

# 2. Customizable Notifications: Users can set notifications for specific topics or news sources.

# 3. In-depth Analysis and Commentary: Access to expert analysis and commentary on news articles.

# 4. Responsive Design: Optimized for multiple devices, ensuring a seamless user experience.

# 5. Search and Filtering: Users can search and filter news articles by topic, source, or keyword.

# 6. Bookmarking and Saving: Users can save articles for later reading.

# 7. User Profile Management: Users can manage their preferences, notifications, and saved articles.

# 8. Real-time Updates: News feed updates in real-time, ensuring users stay informed.

# 3. Architecture.

The architecture of *News Landscape* is designed using a full-stack model, consisting of a **frontend** developed in React.js and a **backend** built with Node.js and Express. The application follows a modular and component-based approach, which makes the system organized, maintainable, and scalable.

On the **frontend**, the main root component App.js handles the overall structure of the application, manages routing, and controls the layout. The NewsCard.js component is used to display individual news articles, including the title, description, and image, in a visually appealing format. The index.js file serves as the entry point, where the React application is rendered into the DOM. Together, these components form the foundation of the user interface, ensuring responsiveness and clarity.

The **backend** plays a critical role in managing data and communication with the frontend. The server.js file is responsible for running the Node.js server, handling API requests, and sending responses back to the frontend. The package.json file defines the backend dependencies required for running the server. By separating backend logic from frontend design, the project ensures a clear division of concerns and makes the application easier to extend with new features.

**State management** is handled locally within components using React’s built-in hooks such as use State and use Effect. Local state is used for managing dynamic values like fetched news data, while props allow data to be passed between components. For larger applications, this setup can be extended to global state management solutions like the Context API or Redux.

The application also supports **basic routing**, managed within React. Although the current version uses simple routing, it can easily be extended with react-router-dom to handle multiple pages, categories, or navigation paths.

Overall, the architecture of *News Landscape* demonstrates a well-balanced structure between the frontend and backend, ensuring smooth data flow, responsive user interfaces, and scalability for future enhancements.

# 4. Setup Instructions.

1️⃣ Install **Node.js (v16 or above)** and a package manager like **npm** or **yarn** ⚙️.

2️⃣ Clone the project repository using Git and move into the project folder 📂.

⇛git clone <repository-link>

⇛cd landscape

3️⃣ Navigate to the **backend** folder to configure server-side dependencies 🔧.

⇛cd backend

4️⃣ Install backend dependencies listed in package.json 📦.

⇛npm install

5️⃣ Start the backend server to handle API requests 🌐.

⇛node server.js

6️⃣ Navigate to the **frontend** folder to configure the React app 💻.

⇛cd ../frontend

7️⃣ Install frontend dependencies required for the React application 📦.

⇛npm install

8️⃣ Start the React development server 🎬.

⇛npm start

9️⃣ If using a News API, create a .env file in the backend to store API keys 🔑.

⇛NEWS\_API\_KEY=your\_api\_key

⇛PORT=5000

🔟 Once both servers are running,

access the app at [**http://localhost:3000**](http://localhost:3000) with backend support at

[**http://localhost:5000**](http://localhost:5000) 🚀.

# 5. Folder Structure.

landscape/

┣ backend/

┃ ┣ package.json

┃ ┣ server.js

┣ frontend/

┃ ┣ public/

┃ ┃ ┗ index.html

┃ ┣ src/

┃ ┃ ┣ App.js

┃ ┃ ┣ index.js

┃ ┃ ┣ styles.css

┃ ┃ ┗ components/

┃ ┃ ┗ NewsCard.js

┃ ┣ package.json

┣ run command.txt

## 6. Running the Application.

**1. Starting the Backend**  
Navigate to the backend folder and start the server:

⇛cd backend

⇛node server.js

⇛The backend will run on [**http://localhost:5000**](http://localhost:5000).

**2. Starting the Frontend**  
Go to the frontend folder and start the React development server:

⇛cd ../frontend

⇛npm start

The frontend will run on [**http://localhost:3000**](http://localhost:3000).

**3. Environment Variables**  
If the project uses a News API, create a .env file inside the backend folder and add your API key:

⇛NEWS\_API\_KEY=your\_api\_key

⇛PORT=5000

**4.Running Both Servers Together**  
The backend must be running before the frontend, so that the frontend can fetch data from it without errors.

**5. Accessing the Application**  
Open a web browser and go to [**http://localhost:3000**](http://localhost:3000) to view the application. The frontend will communicate with the backend at [**http://localhost:5000**](http://localhost:5000).

**6. Stopping the Servers**  
Use CTRL + C in the terminal to stop either the backend or frontend server when not in use.

# 7. API Documentation.

🛠 The project is built using React components to create a dynamic and responsive interface.

📂 App.js serves as the root component of the application.

🔗 It imports and renders all other components, including NewsCard.js.

📰 NewsCard.js displays individual news articles with details like title, description, image, and date.

♻️ Each news card is designed to be reusable, allowing multiple articles to be displayed efficiently.

🔄 Props are used to pass data from App.js to NewsCard.js, ensuring flexibility.

⚡ Local state within components is managed using React’s use State and use Effect hooks.

📊 This state management helps handle data fetching and dynamic rendering of news articles.

🏗 The modular design allows developers to add or modify components without affecting existing functionality.

🧹 Components are structured to separate concerns, making the code cleaner and easier to maintain.

🔍 Future components, such as a search bar or category filter, can be added seamlessly.

📈 The design promotes scalability, which is important for larger applications.

🚀 Reusability reduces code redundancy and improves development speed.

🧪 Each component can be tested individually, simplifying debugging and maintenance.

🌟 Overall, the component-based approach ensures a maintainable, organized, and efficient project structure.

## News Endpoints.

* GET /api/news → Fetches the latest news articles from the API and sends them to the frontend.
* GET /api/news/:category → Fetches news based on category (e.g., sports, war, business).
* GET /api/news/:id → Retrieves details of a specific news article by ID.

## 8. Authentication.

Authentication in the *News Landscape* project is implemented to ensure secure access to user accounts and personalized features. The system is designed to prevent unauthorized access and protect sensitive user information.

The backend uses JWT (JSON Web Token) based authentication. When a user logs in, the server generates a token that is sent back to the client. This token must be included in subsequent requests to access protected resources. Since JWT is stateless, it allows scalability and easy validation without storing session data on the server.

For registration, users provide details such as username, email, and password. Passwords are stored securely using encryption or hashing techniques (e.g., bcrypt). During login, the provided credentials are validated, and if correct, a token is issued.

Middleware functions are applied to protect private routes. Any request to a protected API endpoint must include a valid token in the authorization header. If the token is missing or invalid, access is denied.

In the current version of *News Landscape*, authentication is a basic layer. However, it can be extended to support role-based access control, where different users (like admins and regular users) have different levels of access.

This approach ensures that the application is secure, scalable, and ready for further enhancements like profile management, saved news, and personalized feeds.

# 9. User Interface.

The Insight Stream project aims to create a user-friendly interface for navigating news, leveraging frontend development to provide an engaging experience.

Key Features:

1. News Feed: A curated feed displaying relevant news articles.

2. Search Functionality: Users can search for specific news topics or keywords.

3. Categorization: News articles are organized by categories (e.g., politics, technology, sports).

4. Personalization: Users can customize their news feed based on interests.

Frontend Development:

1. Responsive Design: The UI is optimized for various devices (desktop, tablet, mobile).

2. Interactive Elements: Engaging features like animations, transitions, and hover effects.

3. Accessibility: The interface is designed to be accessible for users with disabilities.

Goals:

1. User Engagement: Create an intuitive and engaging experience.

2. Information Discovery: Facilitate easy discovery of relevant news articles.

3. Personalization: Allow users to tailor their news feed to their interests.

Technologies Used:

1. HTML/CSS: Structuring and styling the UI.

2. JavaScript: Adding interactivity and dynamic functionality.

3. Frontend Frameworks: Possibly using frameworks like React, Angular, or Vue.js.

# 10. Testing.

🧪 Unit testing is performed using **Jest** to ensure individual components work as expected.

⚡ **React Testing Library** is used to test component rendering and user interactions.

🔍 Components like NewsCard.js and App.js are tested for correct display of news data.

🏗 Testing ensures that props and state updates behave correctly.

💻 Integration testing checks that frontend components communicate properly with the backend.

📊 Backend API testing can be done using **Postman** to verify endpoints and responses.

📝 Test cases include checking data fetching, error handling, and UI updates.

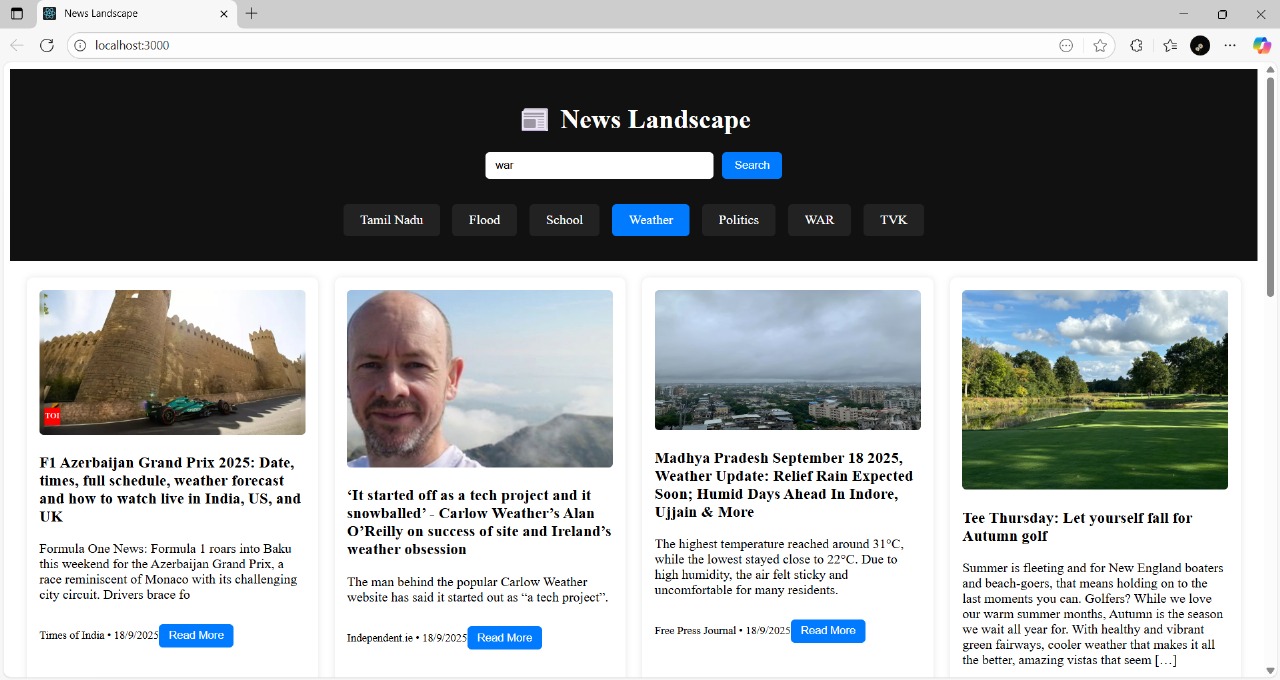
🚀 Automated testing improves reliability and reduces bugs during development.

⚙️ Continuous testing helps maintain stable and scalable code as the project grows.

🌟 Proper testing ensures a smooth and bug-free user experience across all devices.

# 11. Screenshots or Demo

Screenshot 1: Homepage with news cards.



# 12. Known Issues.

Requires News API key; without it no data loads.

Limited category filtering features.

# 13. Future Enhancements.

Add user authentication.

Save/bookmark articles.

Infinite scrolling or pagination.

Dark mode theme.