**Project Documentation**

**INSIGHT STREAM**

# 1. Introduction

* **Project Title:** Insight Stream
* **Team ID:** NM2025TMID38588
* **Team Leader:** Manju R & Mail id: manju13112006@gmail.com
* **Team Members:**
  + Vaigai Selvi & Mail id: vaigaiselvie@gmail.com
  + Gopika K & Mail id: gopikakgopikak92@gmail.com
  + Keerthika R & Mail id: keerthikaravi077@gmail.com
  + Mahalakshmi M & Mail id: mahalakshmiofficial2007@gmail.com

# 2. Project Overview

**Purpose**: **Insight Stream** is a user-centric news platform designed to help individuals stay informed, explore diverse topics, and access the latest updates with ease. Built using React.js, it emphasizes intuitive design, visual storytelling, and personalized content discovery.

* **Features:**
  + **API Integration**: Fetch real-time news using NewsAPI via RapidAPI
  + **Visual News Exploration**: Image galleries for immersive browsing.
  + **Modular Architecture**: Maintain clean separation of concerns using React components, context, and routing.
  + **Scalable Design**: Ensure the project can be extended with features like bookmarking, newsletter subscriptions, and multilingual support.
  + **User-Friendly Interface**: Deliver a seamless, responsive UI for effortless navigation.
  + **Category-Based News Discovery**: Enable users to explore news by topic, popularity, or search query.
  + **Advanced Search Feature:** Easily access news articles on specific topics through a powerful search feature, providing users with tailored news content based on their interests.

# 3. Architecture

### **Component Structure:**

* + - **NavbarComponent:** Navigation across categories
    - **hero:** Introductory banner
    - **TopStories, HomeArticles**: Display news cards
    - **NewsLetter, Footer**: Engagement and branding
* **State Management:**
  + - **Local State**: useState for managing news data
    - **Global State**: GeneralContext.jsx using React Context API

### **Routing:**

### **React Router handles navigation:** Dynamic routes for categories and individual news pages.

<Routes>

<Route path="/" element={<Home />} />

<Route path="/category/:id" element={<CategoryPage />} />

<Route path="/news/:id" element={<NewsPage />} />

</Routes>

# 4. Setup Instructions

* **Prerequisites:**
  + Node.js
  + Visual Studio Code
  + Git
* **Installation Steps:**

Clone the repository git clone

* git clone <your-repo-url>
* cd insightstream
* npm install
* npm start

# 5. Folder Structure

|code/

|-- public/ # React frontend

|\_\_favicon.ico/

|\_\_ index.html/

|\_\_ logo192.png/

|\_\_logo512.png/

|\_\_manifest.json/

|\_\_robots.txt/

|\_\_ src/

|\_\_components/

|\_\_Footer.jsx /

|\_\_Hero.jsx /

|\_\_HomeArticles.jsx /

|\_\_NavbarComponent.jsx /

|\_\_NewsLetter.jsx /

|\_\_ TopStories.jsx/

|\_\_context/

|\_\_ GeneralContext.jsx/

|\_\_ pages/

|\_\_CategoryPage.jsx /

|\_\_Home.jsx /

|\_\_NewsPage.jsx /

|\_\_styles /

|\_CategoryPage.css /

|\_\_Footer.css /

|\_\_Hero.css /

|\_\_Home.css /

|\_\_HomeArticles.css /

|\_\_ NavBar.css/

|\_\_ NewsLetter.css/

|\_\_ NewsPage.css/

|\_\_TopStories.css /

|\_\_ App.css/

|\_\_App.js/

|\_\_ App.test.js/ # Node.js backend

|\_\_index.css /

|\_\_ index.js/

|\_\_ logo.svg/

|\_\_reportWebVitals.js /

|\_\_ setupTests.js/

# 6. Running the Application

* **Install dependencies:**
* npm install
* **Start the app:**
  + - npm start
* **Access:** Visit http://localhost:3000

## **7. State Management**

* **Global State**: GeneralContext.jsx manages shared data like selected category or theme.
* **Local State**: Each page/component uses useState for API responses and UI control.

# 8. Component Documentation

|  |  |
| --- | --- |
| **NavbarComponent:**   * **Purpose:** App-wide navigation and category links * **Props:** None   **Hero:**   * **Purpose:** Intro headline and subtext for the Home page * **Props:** None   **TopStories:**   * **Purpose:** Grid/list of trending articles * **Props:**   + **articles:** Array of article objects (title, urlToImage, description, source, url)   **HomeArticles:**   * **Purpose:** Secondary blocks for Business/Technology/etc. (optional or future) * **Props:**   + **sections:** Array of { title, articles }   **NewsLetter:**   * **Purpose:** Email subscribe UI (frontend only) * **Props:** None   **Footer:**   * **Purpose:** Site-wide footer with category tags and legal text * **Props:** None   **ArticleCard:**   * **Purpose:** Reusable card for any article list * **Props:**   + **article:** { title, urlToImage, description, publishedAt, source, url } |  |

# 9. User Interface

**Screens and flows:**

* **Home:** Navbar → Hero → TopStories → Newsletter → Footer
* **Category:** Title → Articles grid → Pagination (optional)
* **News details (optional):** Inline details or redirect to external URL

**Accessibility and UX:**

* Readable font sizes, alt text on images, keyboard focus on links
* Consistent spacing, clear hierarchy, mobile-first layout
* Clean layout with cards and grids
* Responsive design for mobile and desktop
* Visual hierarchy using headlines and thumbnails
* Accessible color contrast and readable font

# 10. Styling

# CSS approach:

* **Plain CSS files:** One per component/page under “styles/”
* **Naming:** Class names for readability and reuse.

**Optional libraries:**

* + **Bootstrap or Material UI:** Can be added for grids and responsive utilities

**Theming:**

* + **Future toggle:** dark/light using context and CSS variables

# 11. Testing

**Testing strategy:**

* Manual testing in Chrome and Firefox
* Console logging for API errors
* Responsive layout checks
* Future: Jest for unit testing (App.test.js, setupTests.js)

**Suggested coverage targets:**

* **Core components:** Navbar, ArticleCard, TopStories
* **Pages:** Home and CategoryPage render states (loading/error/success)

# 12. Known Issues

* **Rate limiting**: API rate limits during testing
* **Image reliability:** Some articles lack images; add graceful fallbacks
* **Style**: Styling consistency across screen sizes
* **Routing**: Managing dynamic routes for category pages
* **Language Support**: Planning multilingual support
* **News details page:** Currently redirects to source; full detail view is pending

# 13. Future Enhancements

* **Bookmarks and reading list:** Save articles locally (LocalStorage) or per user (backend)
* **Dark mode:** Theme switch stored in context, persisted in LocalStorage
* **Localization:** Tamil interface and category labels for Chennai youth
* **Summaries:** Auto-summarize long articles client-side
* **Pagination/Infinite scroll:** For better performance on long lists
* **Auth and personalization:** Tailored feeds and saved searches

# 14. API Integration

**Base endpoint:**

* + Everything endpoint with query: https://newsapi.org/v2/everything?q={query}&apiKey={key}

**Environment:**

* Use process.env for API key
* Respect API rate limits and errors
* Code:

import axios from 'axios';

const api = axios.create({

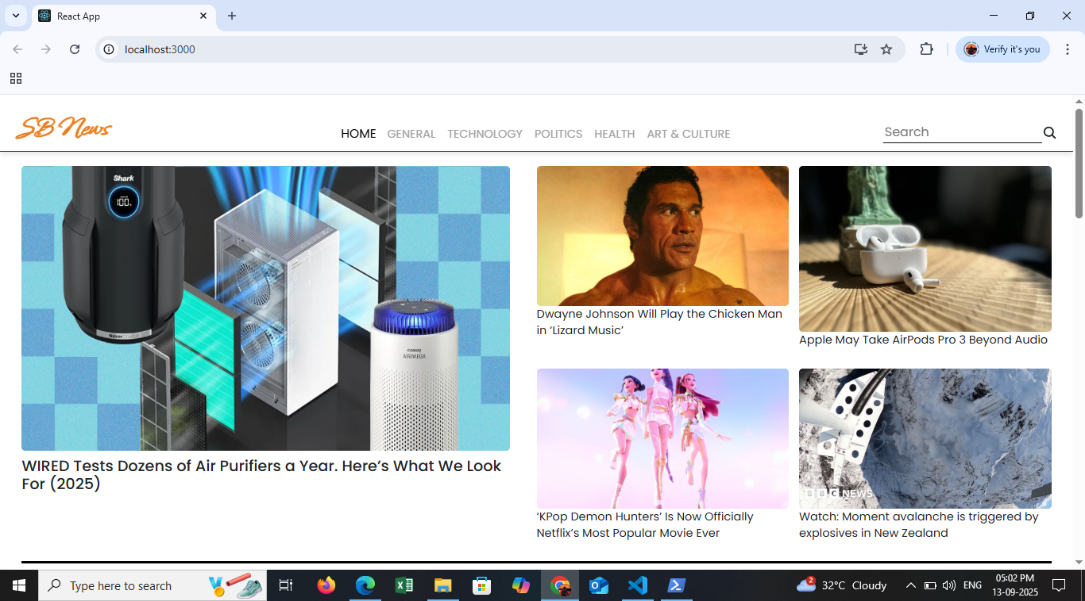
baseURL: 'https://newsapi.org/v2',

params: { apiKey: process.env.REACT\_APP\_NEWS\_API\_KEY },

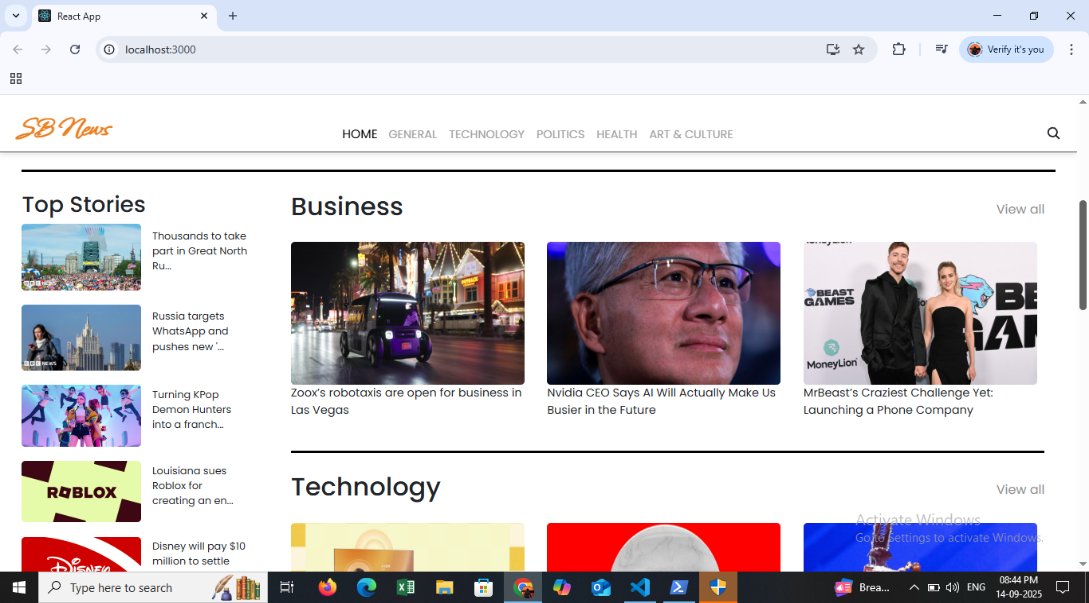
});

# 15. Screenshots or Demo

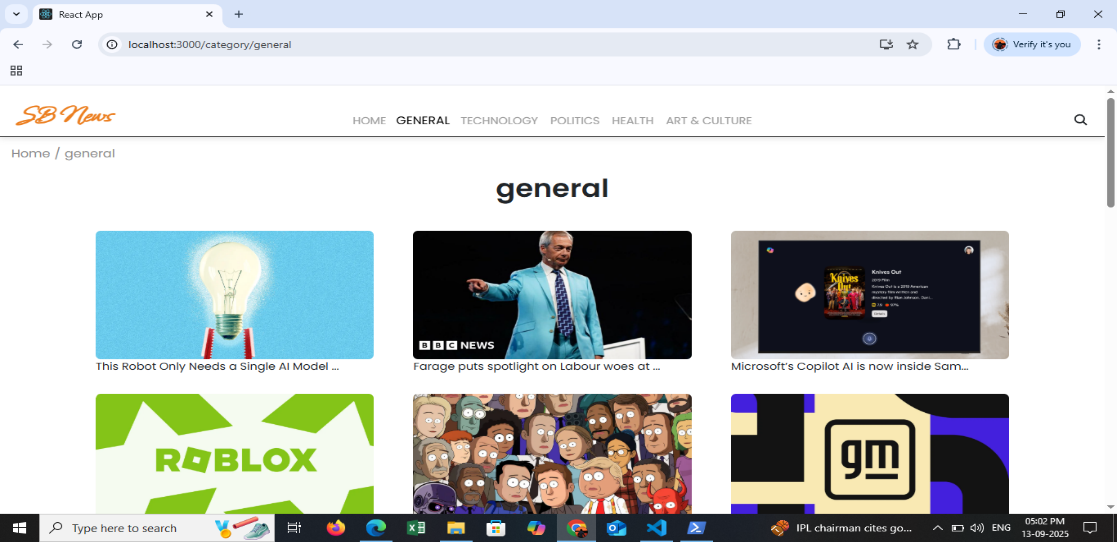
**Home Page**

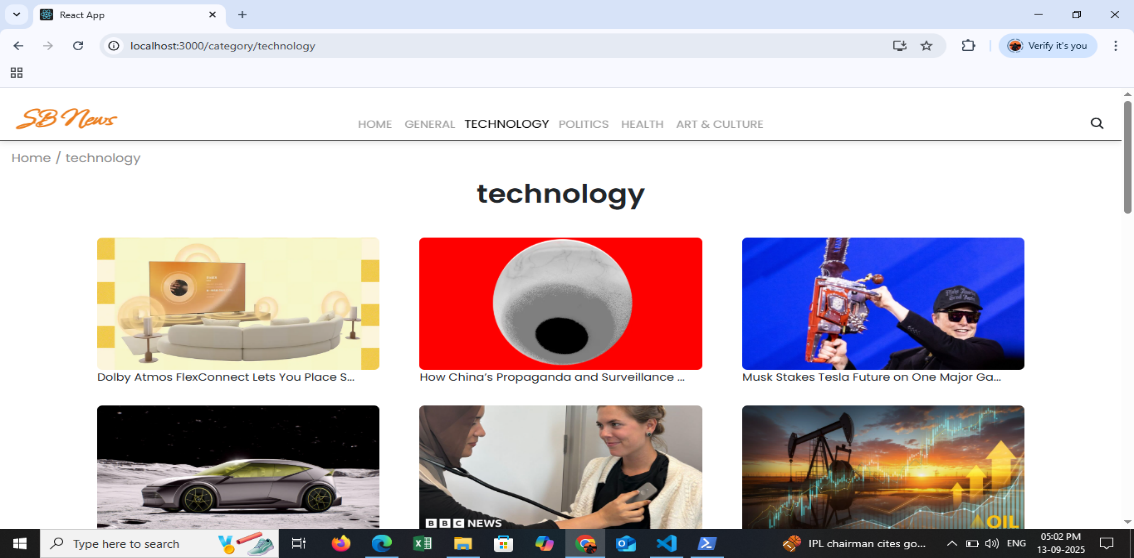


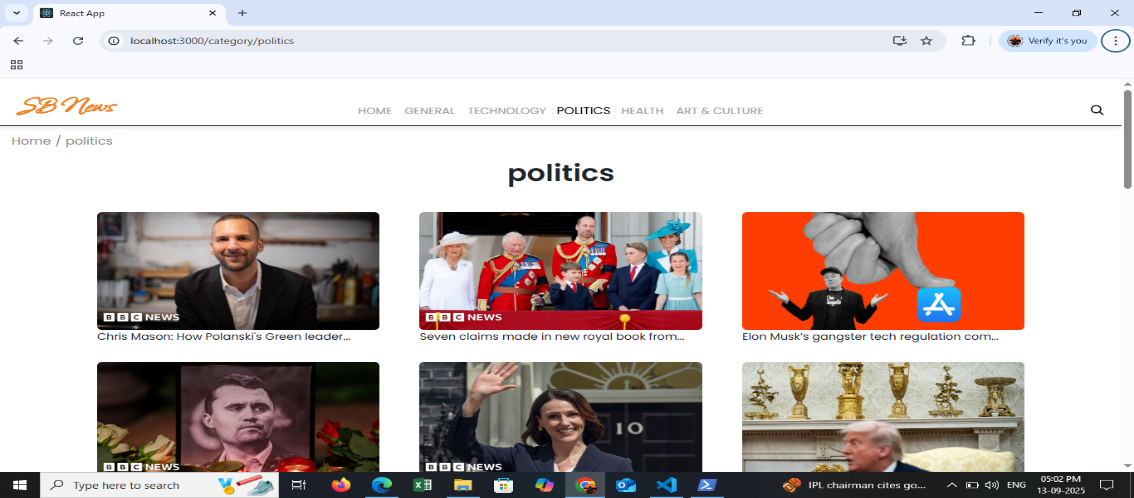
**Top stories**

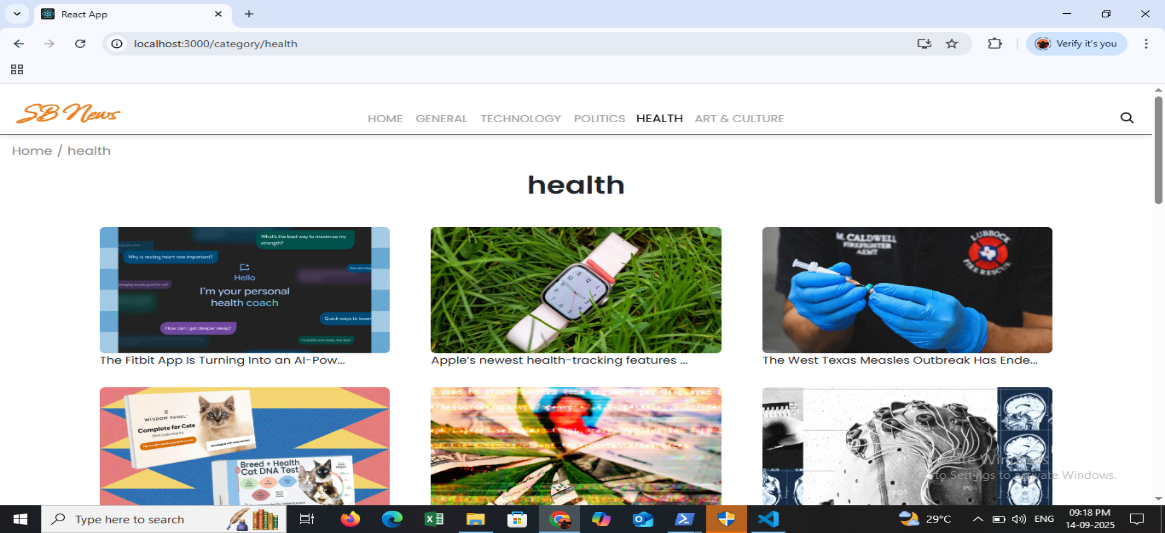


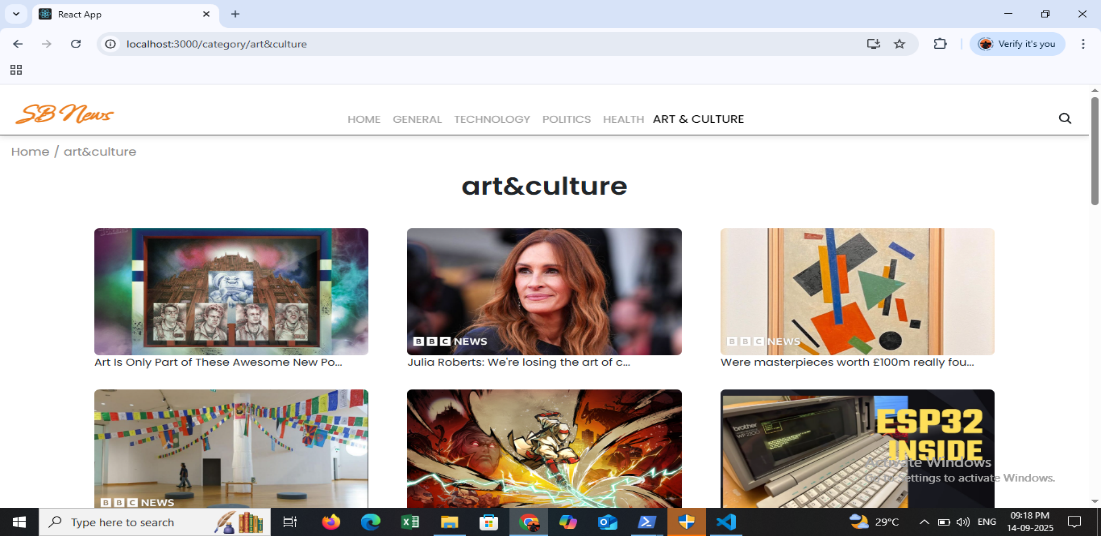
**Categories**



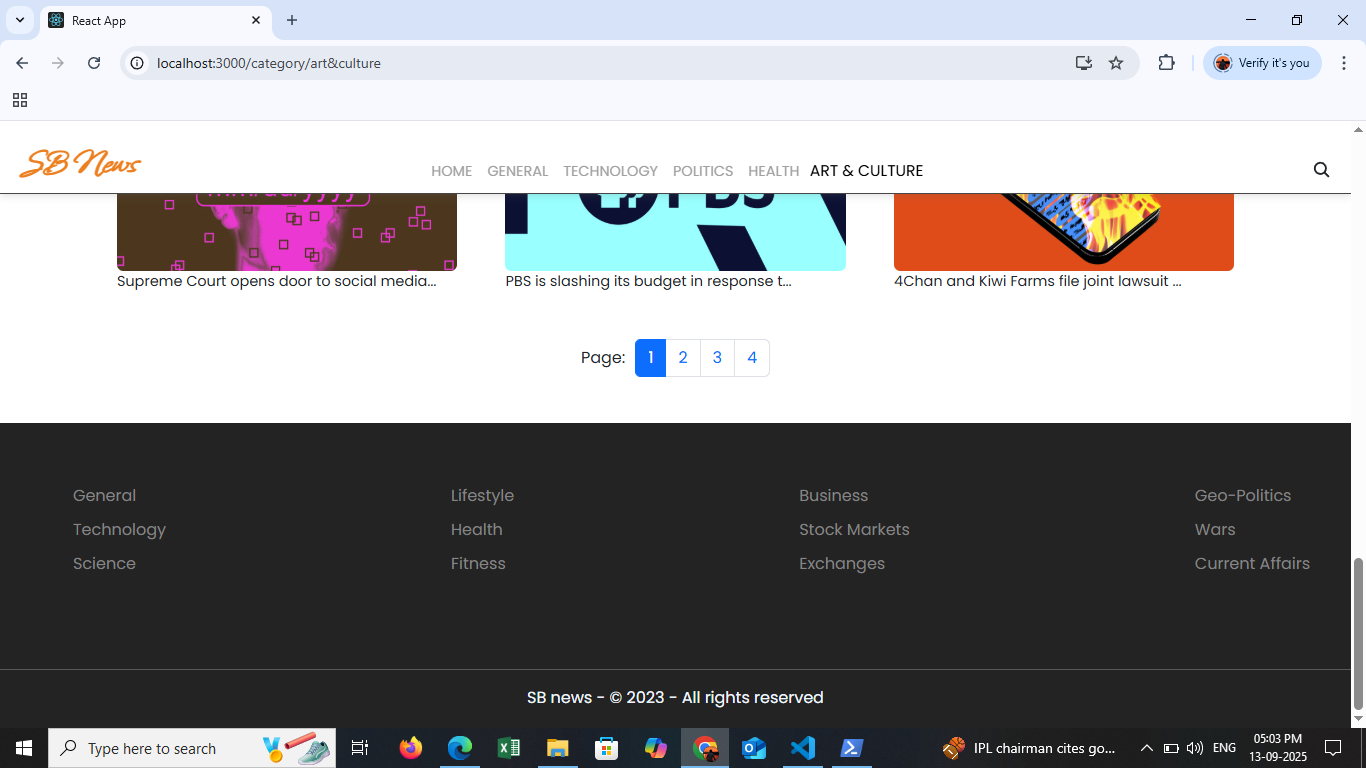




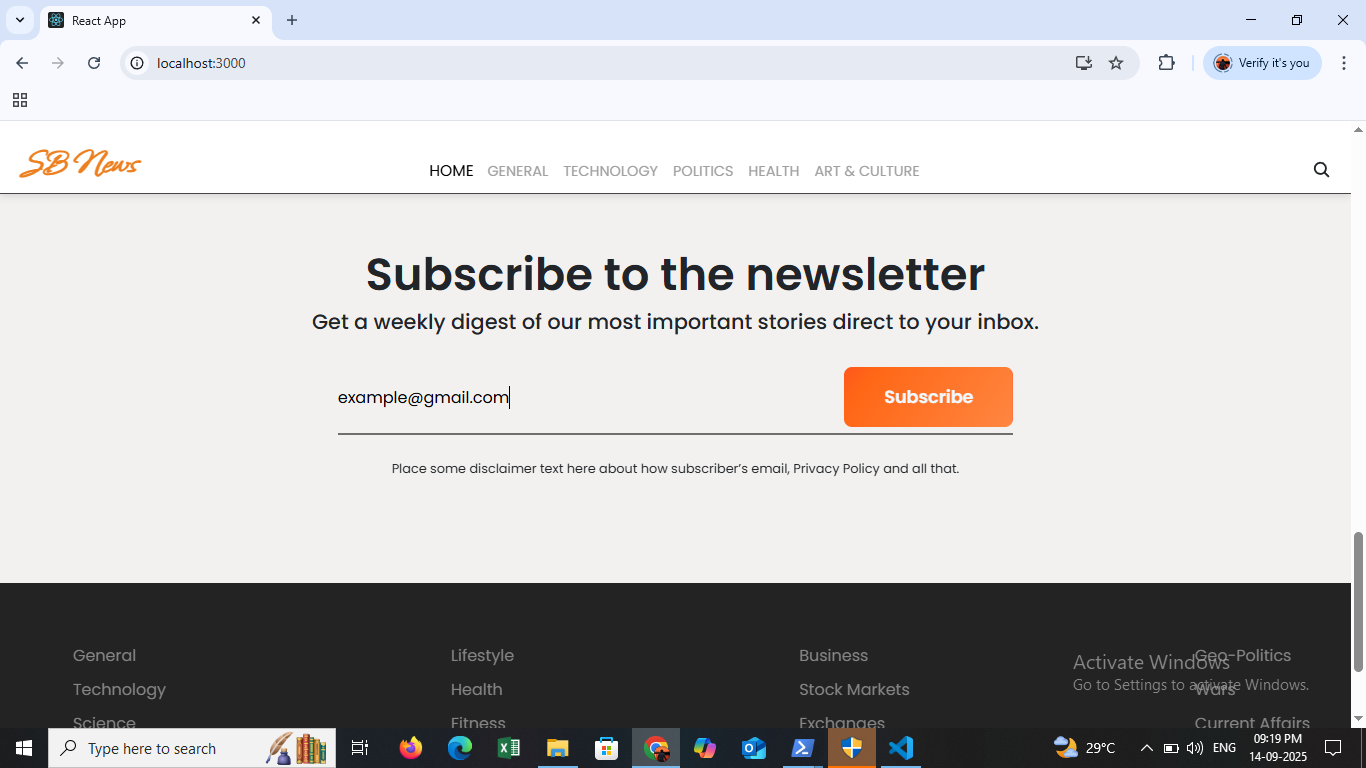




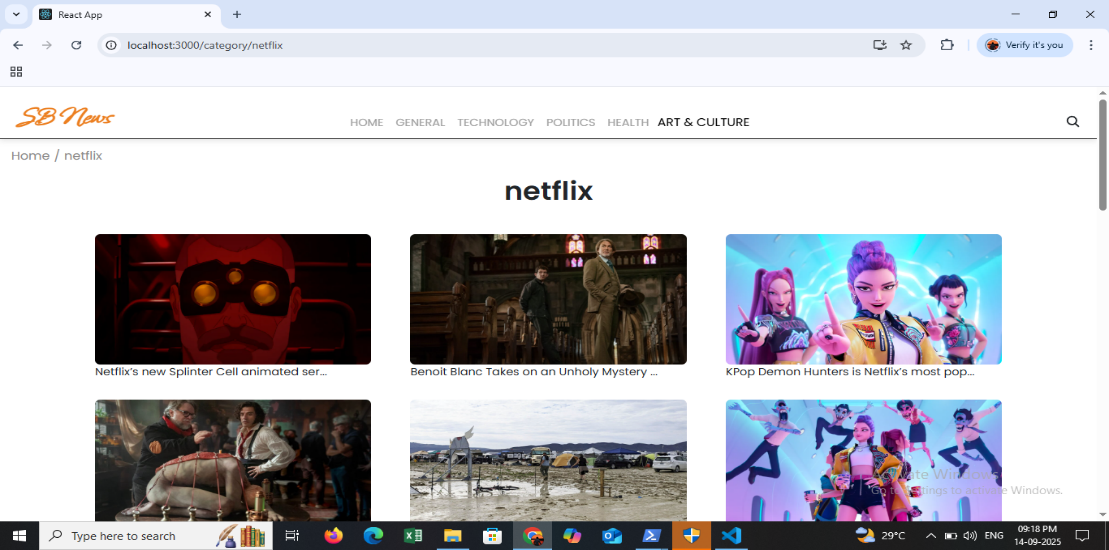
**More Categories in Footer**

****

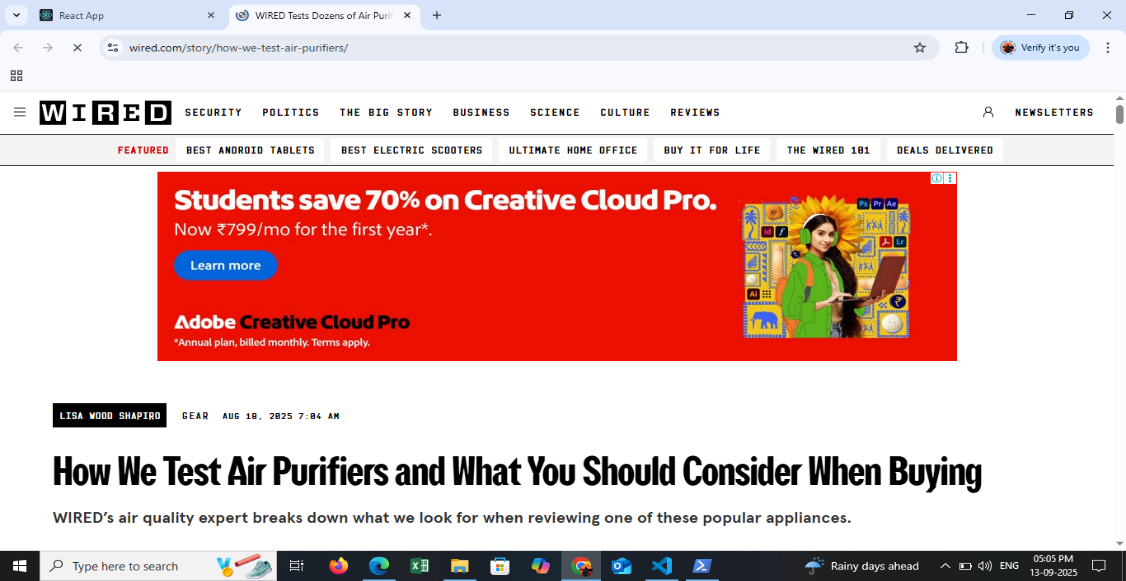
**Subscribing section for NewsLetter**

****

**Search Option (eg: searched Netflix)**

****

**Newspage(example)**

****