INTRODUCTION

PROJECT TITLE:INSIGHT STREAM-NAVIGATE NEWS LANDSCAPE

TEAM MEMBERS:

Team Leader:Monisha.R

Email id:mr1596269@gmail.com

Team Member:Lidiya.B

Email id: blidiyalidiya2004@gmail.com

Team Member:Preethi.D

Email id: Preethipreethi39439@gmail.com

Team Member:Kalaiarasi

Email id: kalaiyarasi6720@gmail.com

PROJECT OVERVIEW

PURPOSE: Insight Stream is designed to help users navigate the complex newslandscape by providing:

1. Personalized news feed: Tailored to individual interests and preferences.
2. Real-time updates: Latest news and developments on various topics.
3. Multisource coverage: Aggregated news from diverse sources, promoting media literacy.
4. Contextualization: Background information and analysis to facilitate deeper understanding.
5. Trending topics: Identification of popular and emerging news stories.
6. Fact-checking: Verification of information to combat misinformation.
7. Diverse perspectives: Exposure to different viewpoints and opinions.

By navigating the news landscape with Insight Stream, users can:

- Stay informed about current events
- Develop a nuanced understanding of complex issues

- Identify biases and misinformation

- Engage with diverse perspectives and opinions

FEATURES: Here are the features of Insight Stream to navigate the news landscape:

Discovery

1. Personalized feed: Tailored to individual interests and preferences.

2. Topic exploration: In-depth coverage of various topics and categories.

3. Trending stories: Real-time updates on popular and emerging news.

Organization

1. Customizable dashboard: Users can organize and prioritize their news feed.

2. News categorization: Stories grouped by topic, theme, or category.

3. Tagging and filtering: Quick access to specific topics or sources.

Contextualization

1. Background information: Providing context and history on complex issues.

2. Analysis and commentary: Expert opinions and in-depth analysis.

3. Related stories: Suggestions for further reading and exploration.

ARCHITECTURE

COMPONENT STRUCTURE:

Data Ingestion

1. News Sources: Integration with various news APIs, RSS feeds, and databases.

2. Data Processing: Cleaning, normalization, and transformation of ingested data.

3. Data Storage: Storage of processed data in a scalable and queryable database.

Component Structure

1. News Feed: Displays a personalized feed of news articles.
   - Article Card: Displays a single news article with summary, image, and metadata.
   - Article List: Displays a list of news articles with summaries and images.
2. Topic Explorer: Allows users to explore news articles by topic or theme.
   - Topic Tree: Displays a hierarchical representation of topics and subtopics.
   - Article Cluster: Displays a cluster of news articles related to a specific topic.

User Interface

1. Header: Displays a header with navigation links and search bar.

2. Footer: Displays a footer with copyright information and links to social media.

Backend Services

1. API Gateway: Handles incoming requests and routes them to appropriate services.

2. Data Service: Provides data storage and retrieval services for news articles and entities.

STATE MANAGEMENT:

State Management Components

1. Redux Store: Centralized store for managing global state.

2. Redux Actions: Actions that trigger state changes in the Redux Store.

3. React Context API: Context API for managing local state and props.

State Management Workflow

1. Action Dispatch: Components dispatch actions to trigger state changes.

2. Action Handling: Redux Reducers handle actions and update the state accordingly.

3. State Update: Redux Store updates the state with the new values.

4. Component Re-render: Components re-render with the updated state.

5. Error Handling: Error messages are displayed to the user if an error occurs.

ROUTING:

Routing Components

1. React Router: Library for managing client-side routing.
2. 2. Route Config: Configuration file for defining routes and their corresponding components.

Route Structure

1. Root Route (/): Renders the homepage component.

2. News Route (/news): Renders the news list component.

3. Topic Route (/topic/:topicId): Renders the topic detail component.

4. Article Route (/article/:articleId): Renders the article detail component.

5. Entity Route (/entity/:entityId): Renders the entity detail component.

6. Search Route (/search): Renders the search results component.

SETUP INSTRUCTIONS

PREREQUISITES:

1. Node.js: Install Node.js (LTS version) from the official website.

2. npm: Install npm (Node Package Manager) along with Node.js.

3. Git: Install Git from the official website.

4. Code Editor: Install a code editor of your choice (e.g., Visual Studio Code, Sublime Text).

5. React: Familiarize yourself with React and its ecosystem.

INSTALLATION:

Step 1: Clone the Repository

1. Open your terminal or command prompt.

2. Navigate to the directory where you want to clone the repository.

3. Run the command: git clone https://github.com/your-username/your-repo-name.git

Step 2: Install Dependencies

1. Navigate to the cloned repository directory.

2. Run the command: npm install

3. Wait for the dependencies to install.

Step 3: Set up Environment Variables

1. Create a new file named .env in the root directory.

2. Add the following environment variables:

REACT_APP_API_KEY=your-api-key

REACT_APP_API_URL=your-api-url

Replace your-api-key and your-api-url with your actual API key and URL.

Step 4: Start the Application

1. Run the command: npm start

2. Open your web browser and navigate to http://localhost:3000

FOLDER STRUCTURE

CLIENT: Client Folder Structure

```
client/
├── public/
│   ├── index.html
│   ├── favicon.ico
│   └── manifest.json
├── src/
│   ├── components/
│   │   ├── App.js
│   │   ├── Header.js
│   │   ├── Footer.js
│   │   └── ...
│   ├── containers/
│   │   ├── AppContainer.js
│   │   ├── HeaderContainer.js
│   │   └── ...
│   ├── actions/
│   │   ├── index.js
│   │   ├── appActions.js
│   │   └── ...
│   ├── reducers/
│   │   ├── index.js
│   │   ├── appReducer.js
│   │   └── ...
│   ├── utils/
│   │   ├── api.js
│   │   ├── constants.js
│   │   └── ...
```

```
│   ├── styles/
│   │   ├── global.css
│   │   ├── components.css
│   │   └── ...
│   ├── index.js
│   └── setupTests.js
├── package.json
└── README.md
```

UTILITY:Utilities Folder Structure

```
utilities/
├── api/
│   ├── index.js
│   ├── api.js
│   └── ...
├── constants/
│   ├── index.js
│   ├── constants.js
│   └── ...
├── helpers/
│   ├── index.js
│   ├── helpers.js
│   └── ...
├── middlewares/
│   ├── index.js
│   ├── middlewares.js
│   └── ...
├── services/
│   ├── index.js
│   ├── services.js
│   └── ...
```

```
├── types/
│   ├── index.js
│   ├── types.js
│   └── ...
├── package.json
└── README.md
```

The client folder contains the React application code, while the utilities folder contains reusable functions, constants, and services that can be used throughout the application.

RUNNING THE APPLICATION

Start the Application

1. Run the command: npm start
2. Open your web browser and navigate to http://localhost:3000

COMPONENT DOCUMENTATION

Here is a sample component documentation for key components and reusable components in the News Landscape application

KEY COMPONENTS:

NewsFeed Component

Prop Name	Type	Description
articles	array	List of news articles
topic	string	Topic to filter articles by
entityId	string	Entity ID to filter articles by

Methods

Method Name	Description

| handleArticleClick | Handles article click event |

| handleTopicChange | Handles topic change event |

Usage

jsx

```
import React from 'react';
import NewsFeed from './NewsFeed';
const articles = [
  { id: 1, title: 'Article 1', topic: 'Topic 1' },
  { id…
```

STATE MANAGEMENT

Here's an overview of the state management for the News Landscape application:

Global State

The global state is managed using Redux, a popular state management library for React applications.

Redux Store

The Redux store is the central location for the global state. It contains the following state:

| State Name | Description |

| --- | --- |

| articles | List of news articles |

| topics | List of topics |

| entities | List of entities |

| filters | Filter settings (e.g., topic, entity) |

| loading | Loading status (true/false) |

| error | Error message (if any) |

STYLING

CSS FRAMEWORK/LIBRARIES:

The News Landscape application uses a combination of CSS frameworks and libraries to style its components. These include:

1. Bootstrap: A popular CSS framework for building responsive and mobile-first UI components.

2. Material-UI: A CSS framework for building UI components based on Google's Material Design principles.

3. Tailwind CSS: A utility-first CSS framework for building custom UI components.

THEMING:

The News Landscape application uses a theme-based approach to styling its components. This allows for easy switching between different visual themes and brand identities.

Theme Structure

The theme structure is based on a nested object that contains the following properties:

1. colors: An object that contains the color palette for the theme.

2. typography: An object that contains the typography settings for the theme.

3. spacing: An object that contains the spacing settings for the theme.

4. icons: An object that contains the icon settings for the theme.

TESTING

Testing Pyramid:

The News Landscape application uses a testing pyramid approach, which consists of:

1. Unit Tests: Testing individual components and functions to ensure they work as expected.

2. Integration Tests: Testing how multiple components interact with each other.

3. End-to-End Tests: Testing the entire application from a user's perspective.

Testing Tools:

The News Landscape application uses the following testing tools:

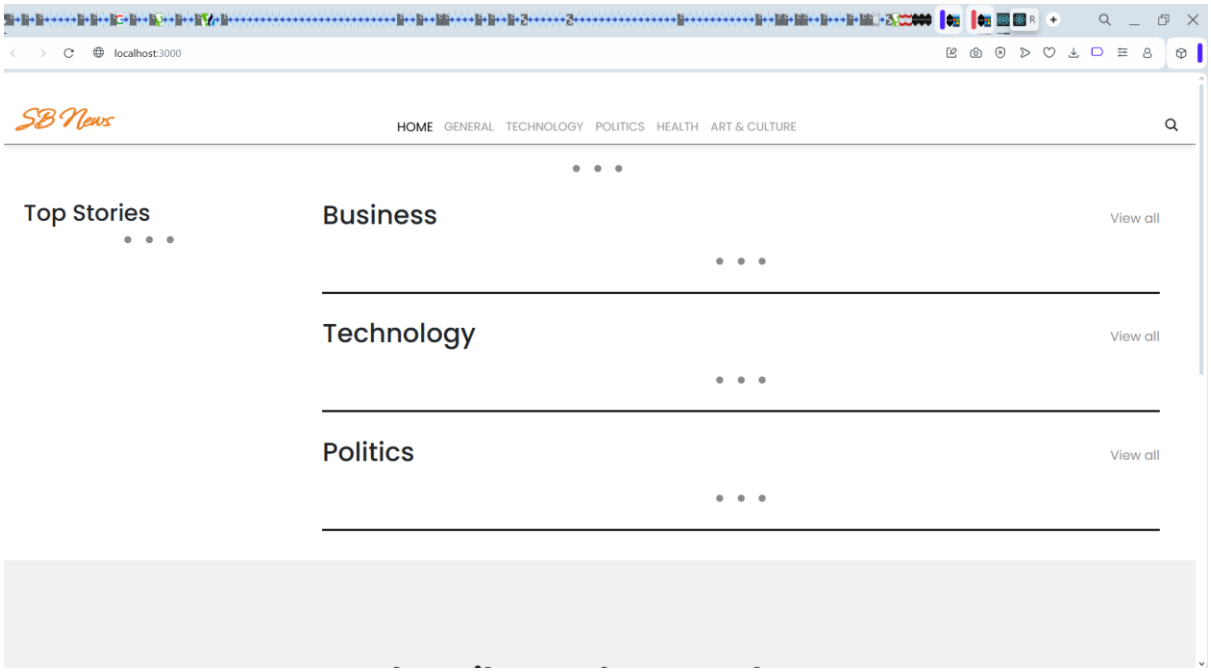
1. Jest CLI: A command-line interface for running Jest tests.

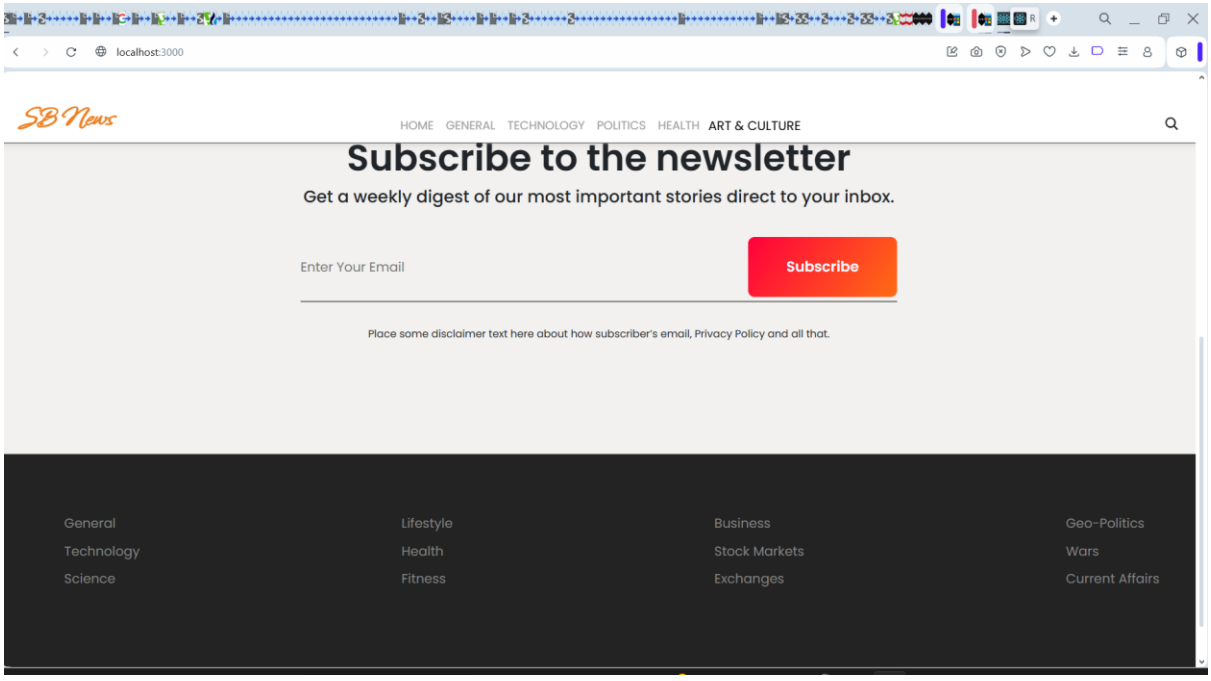
2. Cypress CLI: A command-line interface for running Cypress tests.

3. Enzyme: A testing utility for React applications.

4. Lighthouse: A tool for auditing the performance, accessibility, and security of web applications.

SCREENSHOTS





KNOWN ISSUES

1. Performance Issues: The application may experience performance issues when loading large amounts of data or when using older devices.

2. Compatibility Issues: The application may not be compatible with all devices or browsers, which can cause display or functionality issues.

3. Error Handling: The application may not handle errors properly, which can cause unexpected behavior or crashes.

4. Security Vulnerabilities: The application may have security vulnerabilities that can be exploited by attackers.

FUTURE ENHANCEMENTS

Personalization Enhancements

1. Improved User Profiling: Enhance user profiling to better understand user interests and preferences.

2. Customizable News Feed: Allow users to customize their news feed with specific topics, sources, and keywords.

Artificial Intelligence (AI) and Machine Learning (ML) Enhancements

1. AI-Powered News Summarization: Use AI to summarize long news articles into concise, easily digestible summaries.

2. ML-Based News Classification: Use ML to classify news articles into specific categories, such as politics, sports, or entertainment.