NEWS APPLICATION WITH VOICE ASSISTANT

A MINI PROJECT REPORT

Submitted by

ARBAAZ BAIG A
(RRN: 220282601011)

Under the guidance of

DR.V.SHENBAGA PRIYA

In the partial fulfilment for the award of the degree of

MASTER OF COMPUTER APPLICATION

DECEMBER 2023

B.S. Abdur Rahman Crescent Institute of Science & Technology
Deemed to be University u/s 3 of the UGC Act, 1956

BONAFIDE CERTIFICATE

Certified that this project report titled **NEWS APPLICATION WITH VOICE ASSISTANT** is the bonafide work of **ARBAAZ BAIG A (RRN: 220282601011)** who Carried out the thesis work under our supervision. Certified further, that to the best of our knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr.V.SHENBAGA PRIYA

Assistant professor

Department of Computer Applications

B.S. Abdur Rahman Crescent Institute of Science & Technology, Vandalur,

Chennai - 600048

SIGNATURE

Dr.S.PAKKIR MOHIDEEN

Professor & Head

Department of Computer Applications

B.S. Abdur Rahman Crescent

Institute of Science & Technology,

Vandalur, Chennai - 600048

B.S. Abdur Rahman Crescent Institute of Science & Technology
Deemed to be University u/s 3 of the UGC Act, 1956

VIVA VOICE EXAMINATION

The viva-voce examination of the project work titled **"NEWS APPLICATION WITH VOICE ASSISTANT"** submitted by **ARBAAZ BAIG A (RRN:220282601011)** is held on 19.12.2023

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

I thank the Almighty for showering His blessings upon me in completing the project. We submit this project with a deep sense of gratitude and reverence for my beloved parents for their moral support and encouragement.

I sincerely express my heartfelt gratitude to **Prof. Dr.T.Murugesan** Pro Vice Chancellor, B.S. Abdur Rahman Crescent Institute of Science and Technology and **Prof.Dr.N.RajaHussain**, Registrar, for furnishing every essential facility for doing my project.

I owe my sincere gratitude **to Prof. Dr.Sharmila Sankar**, Dean of Computer, Information and Mathematical Sciences, **Dr.S.Pakkir Mohideen**, Professor andHead, Department of Computer Applications for providing strong oversight of vision, strategic direction, encouragement and valuable suggestions in completing my project work.

I convey my earnest thanks to my project guide **Dr.V.Shenbaga priya**, Assistant Professor, Department of Computer Applications, for her valuable guidance and support throughout the project.

I extend my sincere thanks to all my faculty members for their valuable suggestions, timely advice and support to complete the project.

-ARBAAZ BAIG A

ABSTRACT

With the rapid evolution of digital technology, news consumption has experienced a transformative shift towards more interactive and personalized experiences.This research investigates the development and implementation of a state-of-the-art news application integrated with a voice assistant, aimed at enhancing user accessibility and engagement with current affairs. The studyoutlines the design and functionality of the news application, emphasizing its intuitive interface and seamless integration of voice-enabled features. By leveraging advanced natural language processing (NLP) techniques, the application enables users to access real-time news updates, personalized content recommendations, and tailored news feeds through voice commands, eliminating the constraints of traditional text-based interfaces.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF ABBREVIATIONS

ReactJS	: React JavaScript (a JavaScript library)
DOM	: Document Object Model
CSS	: Cascading Style Sheets
API	: Application Program Interface

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In an era characterized by the convergence of technology and information, the way we consume news is undergoing a transformative shift. Traditional news platforms are being reimagined to cater to the dynamic preferences and expectations of modern users. This project introduces an innovative News Application with an integrated Voice Assistant, leveraging the capabilities of ReactJS to redefine the
News consumption experience.

The primary objectives of this project are outlined as follows:

1. Enhanced User Interaction: Develop a news application that goes beyond traditional interfaces, incorporating a Voice Assistant to allow users to navigate and interact with news content using natural language commands.
2. Real-time News Integration: Utilize APIs from reputable news sources to provide users with up-to-the-minute information, ensuring the delivery of timely and relevant news articles.
3. Seamless Integration with ReactJS: Leverage the ReactJS framework to create a responsive and dynamic user interface, enhancing the overall user experience and adaptability across various devices.
4. Cusomization and Personalization: Implement features that enable users to tailor their news feed based on individual preferences, creating a more personalized and engaging news consumption experience.
5. Accessibility and Inclusivity: Prioritize accessibility by integrating features such as

voice-guided navigation, text-to-speech functionalities, and adjustable settings to accommodate users with diverse needs

1.2 EXISTING SYSTEM:

1. Features and Functionalities:

Describe the primary features offered by the existing news application. This may include features such as browsing news articles, categorization of news, search functionality, and any interactive elements.

2. User Interface (UI) and User Experience (UX):

Evaluate the user interface design and user experience of the news application. Discuss the layout, ease of navigation, and overall design elements.

3. News Sources and Content Integration:

Provide information about the sources of news content integrated into the application. This could include partnerships with news outlets, RSS feeds, or other content aggregation methods.

4. Technological Stack:

Outline the technologies, programming languages, and frameworks used in the development of the existing news application.

5. User Interaction:

Explain how users currently interact with the application, whether through traditional touch-based navigation, gestures, or other input methods.

6. Limitations and Challenges:

Identify any limitations or challenges faced by users in the existing system. This could include issues related to accessibility, personalization, or any feedback received from users.

1.3 PROPOSED SYSTEM:

1. Overview of Proposed Enhancements:
Provide a high-level overview of the proposed changes, emphasizing the integration of a voice assistant into the news application.

2. Objectives:
Clearly state the objectives of integrating a voice assistant, such as improving user experience, enabling hands-free interaction, and enhancing accessibility.

3. Voice Assistant Platform Selection:
Specify the chosen voice assistant platform or technology, whether it is a third party API (e.g., Google Assistant, Siri) or a custom-built voice recognition system.

4. Key Features of the Proposed System:
Enumerate the new features that will be introduced, such as voice-activated news retrieval, personalized news recommendations, and interactive voice commands.

5. User Interaction Flow:
Describe the user interaction flow, detailing how users will engage with the voice assistant to access news content.

6. Integration with Existing Features:
Explain how the voice assistant will be seamlessly integrated with existing features of the news application, ensuring a cohesive and intuitive user experience.

7. Technological Requirements: List the technological requirements for implementing the proposed system, including any updates to programming languages, frameworks, or APIs.

8. User Benefits:
Highlight the benefits that users will gain from the integration of the voice assistant, such as increased convenience, faster news access, and a more personalized experience.

1.4 ORGANIZATION OF CHAPTERS

The documentation for the News Application with Voice Assistant is structured into several chapters to provide a comprehensive understanding of the project. Each chapter serves a specific purpose, guiding the reader through the various stages of conceptualization, development, and Potential future enhancements.

CHAPTER 2

LITERATURE SURVEY & PROBLEM DEFINITION

In the previous chapter, the existing system and the proposed system of the proposed system of the project are discussed. This chapter deals with the literature survey, problem definition and methodology. The problem definition discusses about the objectives of the project and methodology used to develop the project

2.1 LITERATURE SURVEY:

Deny Nancy, Sumithra Praveen, AnushriaSai, M.Ganga, R.S. Abisree, "Voice Assistant Application for a college Website", nternational Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7, Issue-6S5, April 2019.

Deepak Shende, RiaUmahiya, Monika Raghorte, AishwaryaBhisikar, AnupBhange, "AI Based Voice Assistant Using Python", Journal of Emerging Technologies and Innovative Research (JETIR), February 2019, Volume 6, Issue 2.

Salvatore gaglio, Giuseppe Lo Re, Marco Morana, and Claudio Ruocco(2019), smart assistance for students and people living in a campus, IEEEComputer Society.

Isha S. Dubey,Jyotsna S. Verma, Ms.ArundhatiMehendale, "An Assistive System for Visually Impaired using Raspberry Pi", International Journal of Engineering Research & Technology (IJERT), Volume 8 Issue 05, May-2019.

VetonKëpuska, "Next-Generation of Virtual Personal Assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)", Pycon, Cleveland, 2018.

2.2 Problem Definition

Users have to manually run a large set of applications to perform a single comprehensive task. There are already some voice assistants out there, but they're having trouble rearranging their voices. I need a language assistant who understands and can work with Indian accented English. A language assistant should be able to model complex tasks. If a task has multiple subtasks and each subtask can have its own subtasks, then we need to test to find the best path.Voice assistants make our lives easier and save time, and One Voice Assistant provides users with features that are the backbone of our time

CHAPTER 3
METHODOLOGY & REQUIREMENT ANALYSIS

3.1 METHODOLOGY:

The methodology outlines the approach taken to develop the News Application with Voice Assistant using ReactJS. It encompasses the steps involved in system design, development, testing, and evaluation.

System Design:

User Interface Design:

Create wireframes and prototypes for the application's user interface. Implement responsive design principles for a seamless experience across devices.

Voice Assistant Integration:

Choose a suitable natural language processing (NLP) library or API for voice interaction. Design voice command flows and user prompts for a natural and intuitive experience. Development:

Frontend Development with ReactJS:

Set up the development environment with ReactJS. Implement React components for various sections of the application. Integrate external libraries for enhanced UI/UX features.

3.2 REQUIREMENT ANALYSIS:

The requirements analysis outlines the functional and non-functional requirements of the News Application with Voice Assistant.

HARDWARE REQUIREMENT:

Processor	: Intel core i3
RAM	: 8 GB
Hard disk	: 1 TB

SOFTWARE REQUIREMENT:

Operating system	: Windows
Frontend	: REACTJS
API	: NEWS API

3.3 DESIGN:

The design explains the architectural diagram, use case diagram and sequence diagram of voice mailing application

3.3.1 ARCHITECTURAL DIAGRAM:

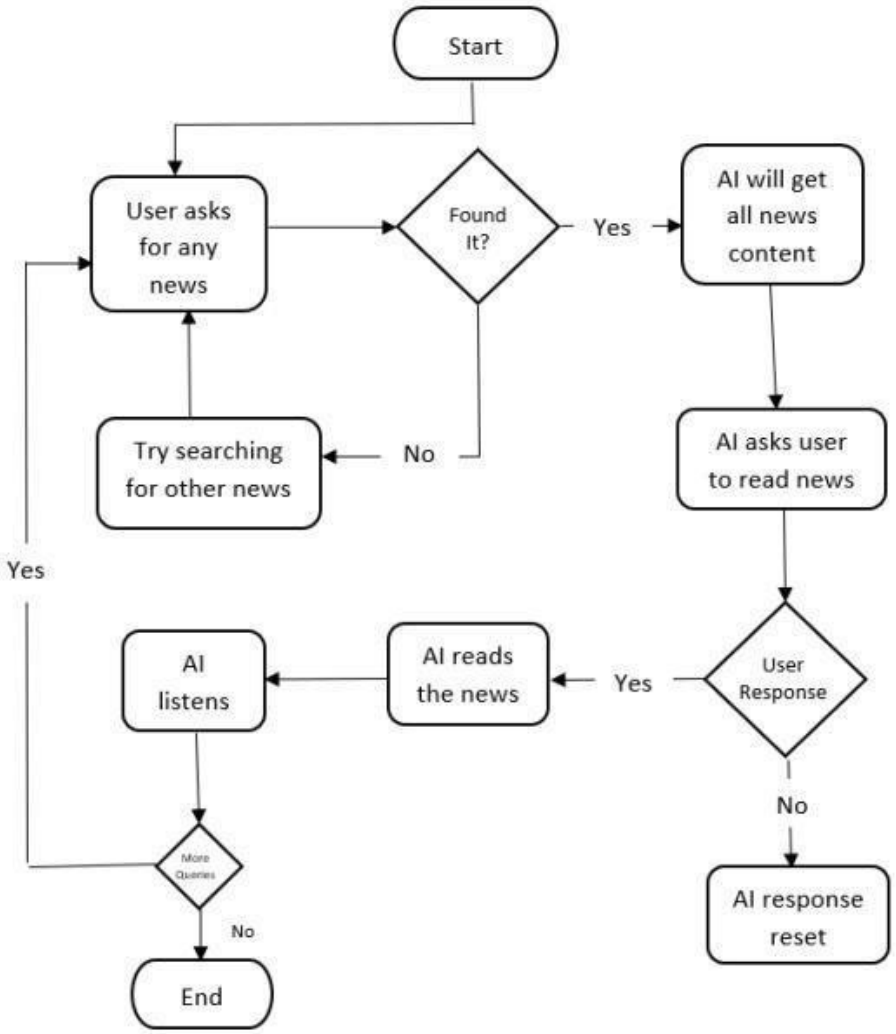

Figure 3.1 Architecture diagram

3.3.2 USE CASE DIAGRAM:

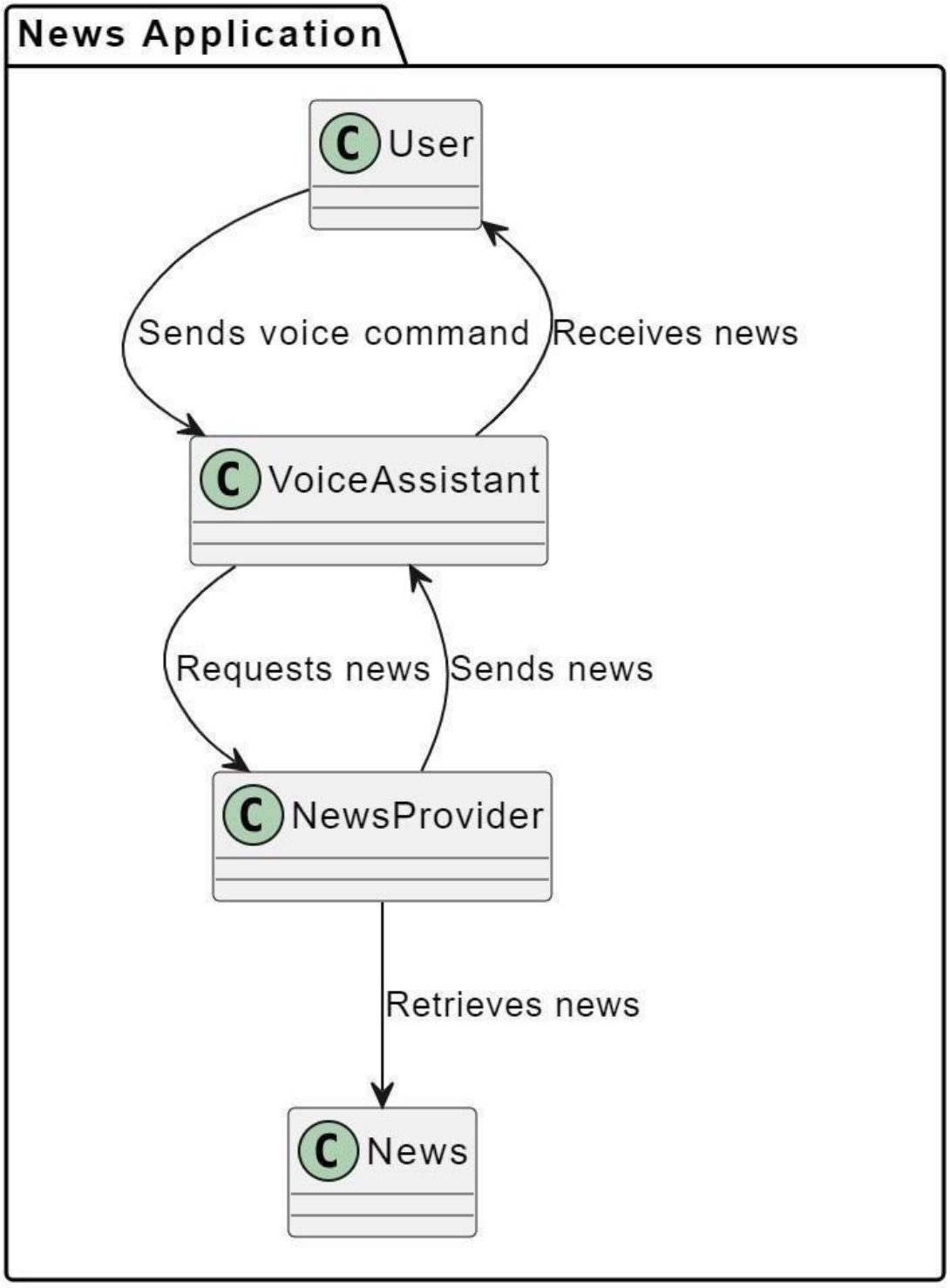

Figure 3.2 UML diagram

3.3.3 FLOW CHART:

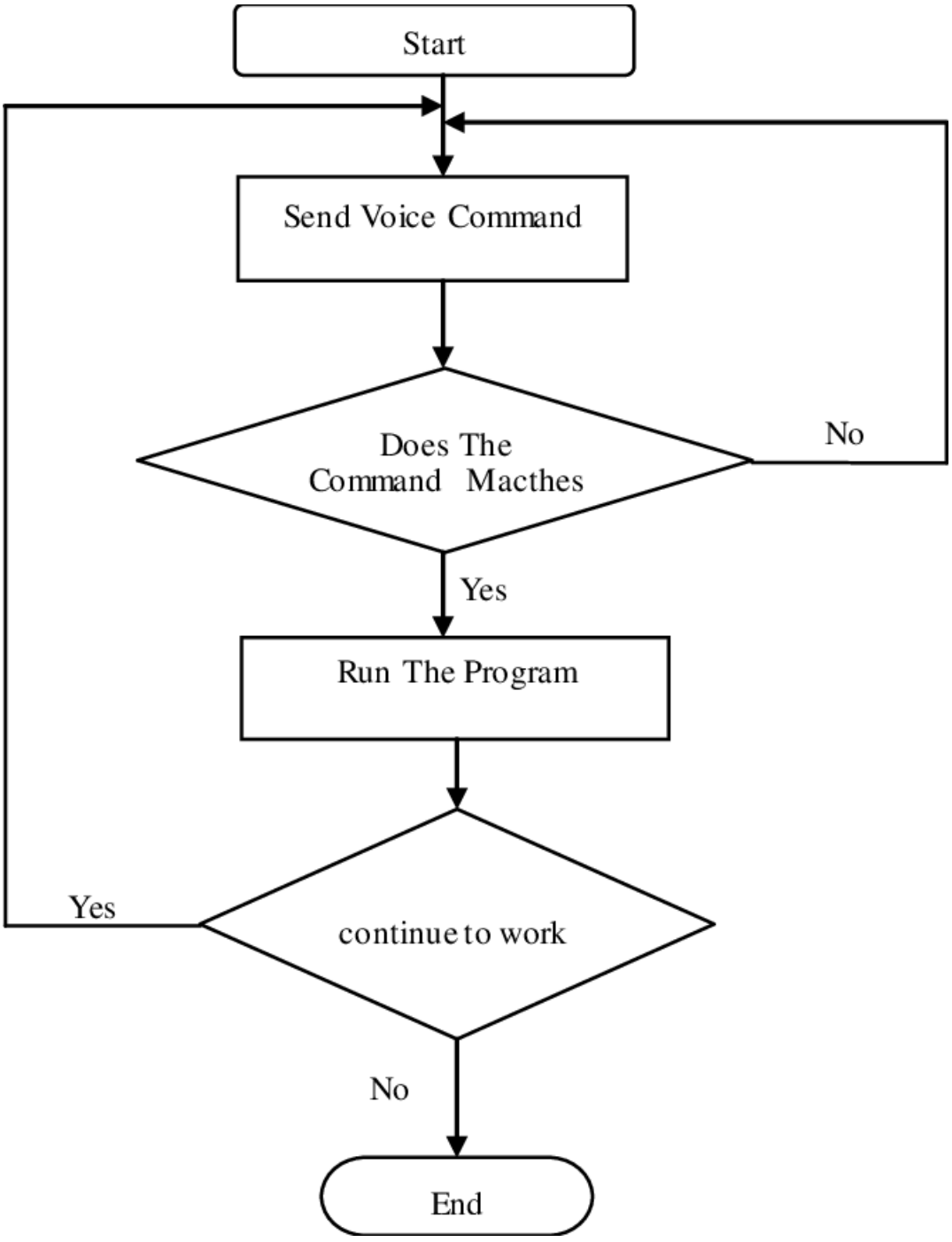

Figure 3.3 Flow chart

CHAPTER-4

IMPLEMENTATION AND

TESTING

4.1 IMPLEMENTATION:

Development Environment:

Set up the development environment with the necessary tools, frameworks, and libraries for both the news application and the voice assistant integration.

Voice Assistant Integration:

Integrate the voice assistant functionality into the news application, ensuring seamless communication between the application and the voice assistant API or platform.

USER INTERFACE

4.1.1Home Page:

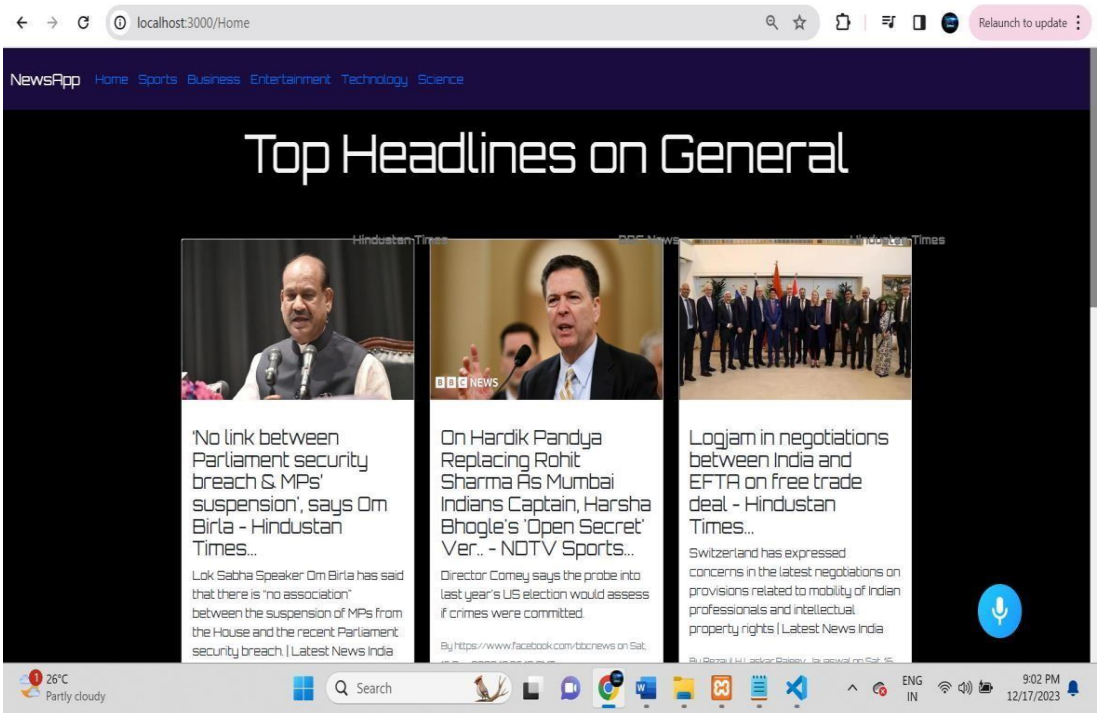

Figure 4.1 Home Page

4.2 SPORTS PAGE:

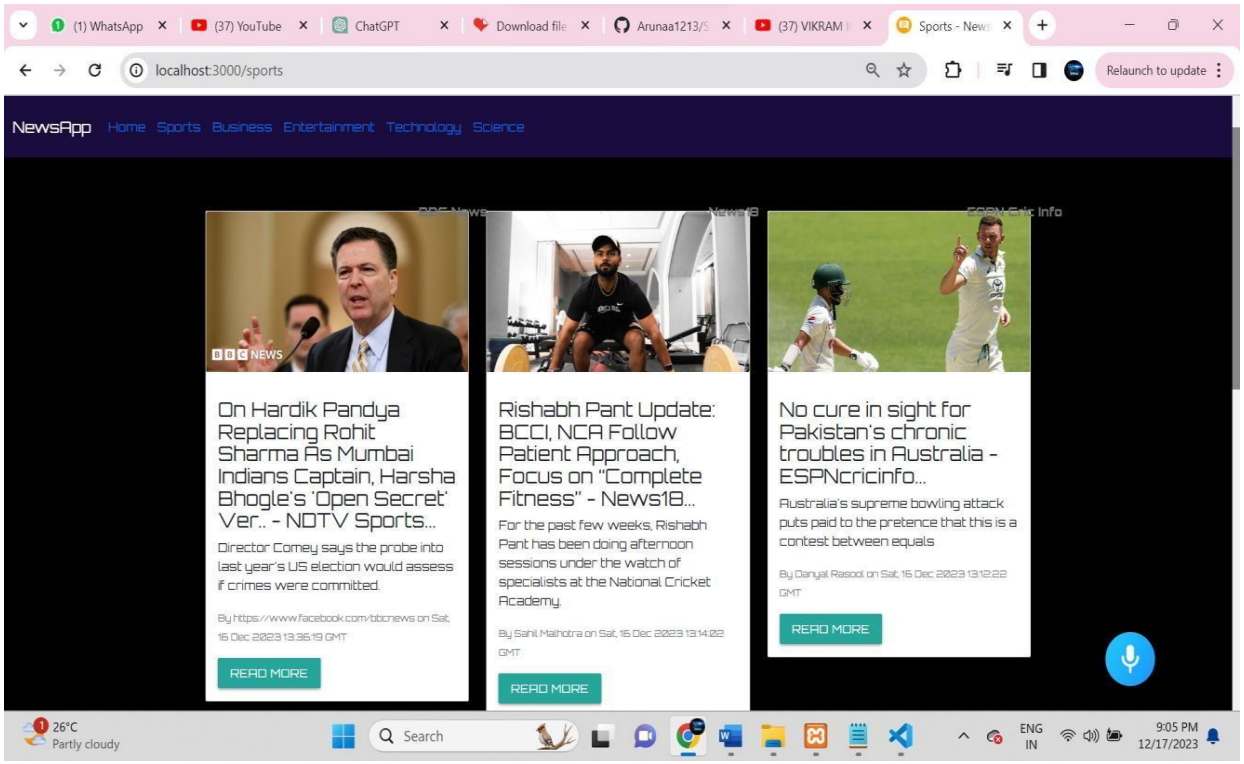

Figure 4.2 Sports Page

4.3 VOICE ASSISTANT

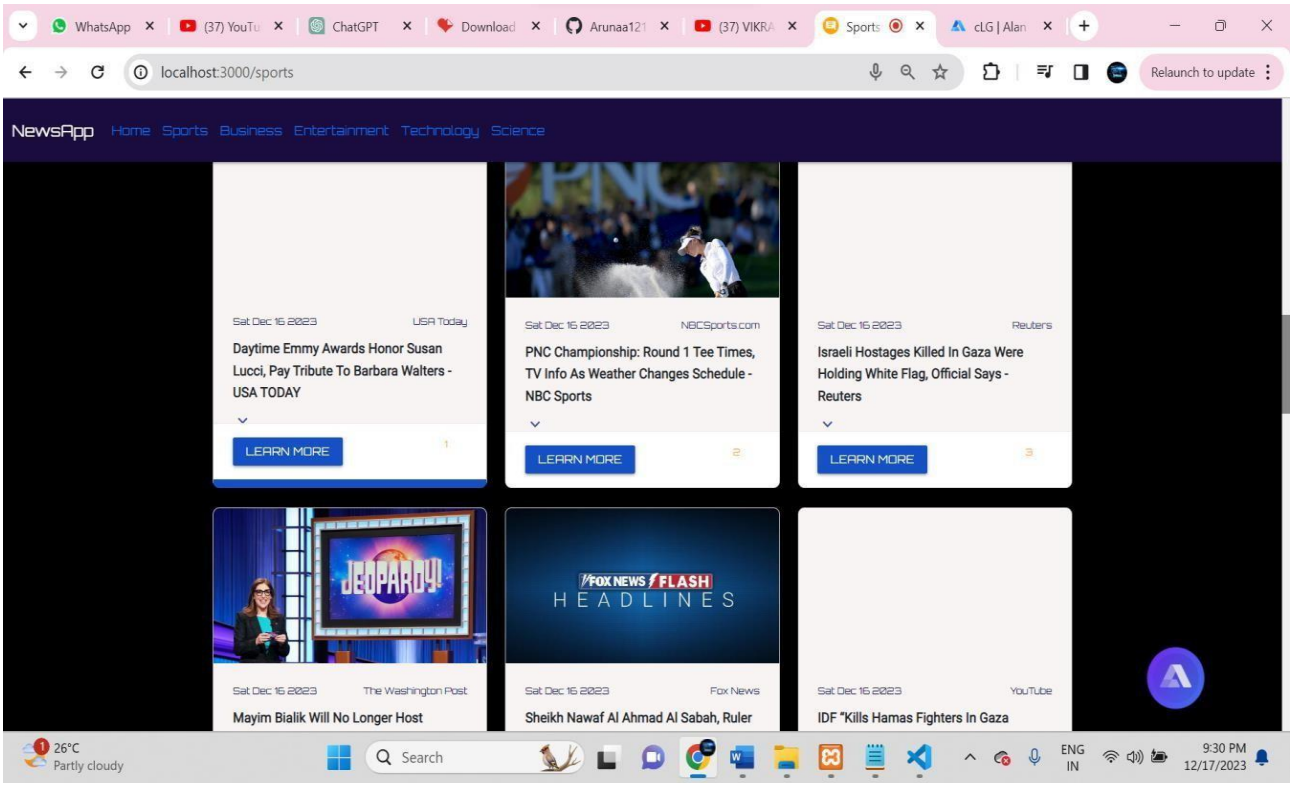

Figure 4.3 Voice Assistant

4.4. OPENING THE NEWS

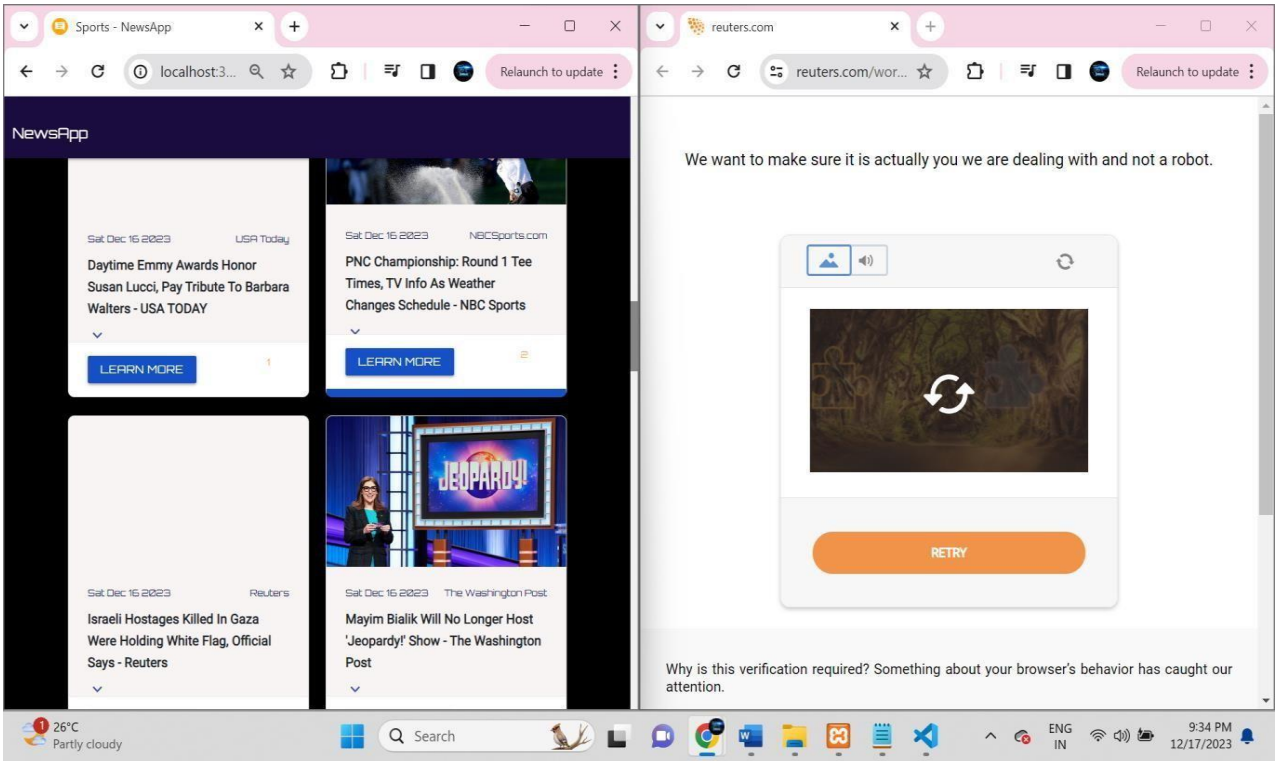

Figure 4.4 News Opening

5. TESTING:

Unit Testing:

Objective:

Verify the correctness of individual components, functions, and modules in the news application and the voice assistant integration.

Components Tested:

Voice recognition module

News retrieval and processing

User interface components related to voice interaction

Testing Tools:

Unit testing frameworks (e.g., JUnit, XCTest)

Voice assistant testing tools (if available)

Results:

Ensure that each component functions as intended and addresses any identified issues.

Integration Testing:

Objective:

Confirm the seamless integration between the news application and the voice assistant.

Integration Points:

Communication between the news application and the voice assistant API.

Data exchange for voice commands and news content retrieval.

Test Cases:

Test voice commands for news categories.

Test voice commands for specific news topics.

Results:

Validate successful integration without errors or unexpected behavior.

CHAPTER 6
RESULTS AND CONCLUSION

6.1 RESULT:

Technical Implementation:

Discuss the technical aspects of integrating the voice assistant into the news application. Detail the development process, including any programming languages, frameworks, or APIs used.

User Interface (UI) and Experience (UX):

Evaluate the user interface design in relation to the voice assistant feature.Discuss any changes made to enhance user experience with the integration of voice commands.

Performance Metrics:

Present any quantitative metrics used to measure the performance of the application with and without the voice assistant.This could include response times, user engagement, and any other relevant performance indicators.

User Feedback:

Include user feedback and reviews related to the voice assistant feature.Summarize any positive or negative comments and discuss how user suggestions were considered, if applicable.

Integration with News Sources:

Discuss how well the application integrated with various news sources and the accuracy of information retrieval through voice commands.

6.2 CONCLUSION

Successes and Achievements:

Summarize the successful aspects of the project, including any positive outcomes or achievements related to the integration of the voice assistant.

Challenges Faced:

Highlight any challenges or obstacles encountered during the development and implementation phases.Discuss how these challenges were addressed and mitigated.

Future Improvements:

Propose potential improvements or future enhancements to the news application and voice assistant integration.Consider any feedback received from users and how it can be incorporated into future iterations.

Overall Impact:

Discuss the overall impact of the voice assistant feature on the usability and popularity of the news application.Reflect on whether the integration met the project's initial goals and expectations.

Closing Remarks:

Conclude with a summary of the project's key findings and the importance of integrating voice assistants into news application

6.2 FUTURE ENHANCEMENT:

Multilingual Support:

Integrate support for multiple languages to cater to a broader audience, allowing users to access news content in their preferred language through voice commands.

Personalized Content Recommendations:

Implement machine learning algorithms to analyze user preferences and behavior, providing personalized news content recommendations based on individual interests.

Contextual Understanding:

Enhance the voice assistant's ability to understand contextual cues in conversations, allowing for more dynamic and nuanced interactions.

Hands-Free Navigation:

Introduce hands-free navigation features, enabling users to browse and consume news content without needing to touch the device, further emphasizing the convenience of the voice assistant.

Real-Time Updates and Breaking News Alerts:

Implement real-time news updates and breaking news alerts through voice notifications, keeping users informed about the latest developments as they occur.

Voice Profiles and User Recognition:

Introduce voice profiles to recognize different users within a household, enabling a more personalized experience for each individual user.

7.CODING

INDEX.JS

```
import React from 'react';
import { createRoot } from 'react-dom/client';
import App from './App';

const container = document.getElementById('root');
const root = createRoot(container); // create a root
root.render(<App />)
```

APP.JS

```
import React, { useState, useEffect } from 'react';
import wordsToNumbers from 'words-to-numbers';
import alanBtn from '@alan-ai/alan-sdk-web'
import Home from './pages/Home'
import News from './components/News'
import Navbar from './components/Navbar';
// import Footer from './components/layout/Footer';
import M from 'materialize-css';
import 'materialize-css/dist/css/materialize.min.css';
import './App.css'
import { BrowserRouter as Router,Routes,Route,} from "react-router-dom";

const alanKey
='f9e1602fba686635e3db1a533c4b22df2e956eca572e1d8b807a3e2338fdd0dc/stage'
const App = () => {
  const [newsArticles, setNewsArticles] = useState([])
  const [activeArticle, setActiveArticle] = useState(-1)
  const loaderContainer = document.querySelector('.loader-container')

  useEffect(() => {
    if (loaderContainer){
      loaderContainer.classList.add('finish')
    }
    //Initialize Materialize JS
    M.AutoInit()
  },[loaderContainer])

  useEffect(() => {
    alanBtn({
        key: alanKey,
        onCommand: ({ command, articles, number }) => {
          if(command === 'newHeadlines'){
```

```
          setNewsArticles(articles)
          setActiveArticle(-1)
        }
        else if (command === 'highlight'){
          setActiveArticle((prevActiveArticle) => prevActiveArticle + 1)
        }
        else if (command === 'open'){
          const parsedNumber = number.length > 2 ? wordsToNumbers((number), { fuzzy:
true }) : number
          const article = articles[parsedNumber - 1]
          if (parsedNumber > 20) {
            alanBtn().playText('Please try that again...');
          }
          else if (article) {
              window.open(article.url, '_blank');
              alanBtn().playText('Opening...');
          }
          else {
              alanBtn().playText('Please try that again...');
          }
        }
      }, bottom: '30px', right:'100px'
    })
  }, [])


  const pageSize=6;
  const apiKey="7bdfb1b10aca41c6becea47611b7c35a"
  return (
    <div className="App">
    <Router>

      <Navbar />
```

```
        <Routes>
        <Route exact path="/"  />
      <Route exact path="/Home" element={<News key="general" pageSize={pageSize}
apiKey={apiKey} country='in' category='general'/>} />
      <Route exact path="/sports" element={<News key="sports" pageSize={pageSize}
apiKey={apiKey} country='in' category='sports'/>} />
      <Route exact path="/business" element={<News key="business" pageSize={pageSize}
apiKey={apiKey} country='in' category='business'/>} />
      <Route exact path="/entertainment" element={<News key="entertainment"
pageSize={pageSize} apiKey={apiKey} country='in' category='entertainment'/>} />
      <Route exact path="/technology" element={<News key="technology"
pageSize={pageSize} apiKey={apiKey} country='in' category='technology'/>} />
      <Route exact path="/general" element={<News key="general" pageSize={pageSize}
apiKey={apiKey} country='in' category='general'/>} />
      <Route exact path="/science" element={<News key="science" pageSize={pageSize}
apiKey={apiKey} country='in' category='science'/>} />
      </Routes>
      <Home articles={newsArticles} activeArticle={activeArticle}/>
        {/* <Footer /> */}
    </Router>
    </div>
  );
}

export default App;
```

NAVBAR.JS

```
import React,{useState} from 'react'
import { Link } from "react-router-dom"
const Navbar = (props) => {
  return (
    <div>
      <nav class="navbar fixed-top navbar-expand-lg nav">
        <div className="container-fluid">
          <Link className="navbar-brand" to="/">NewsApp</Link>
          <button className="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
            <span  className="navbar-toggler-icon"></span>
          </button>
          <div className="collapse navbar-collapse" id="navbarSupportedContent">
            <ul className="navbar-nav me-auto mb-2 mb-lg-0">
              <li className="nav-item">
                <Link className="nav-link" aria-current="page" to="/Home">Home</Link>
              </li>
              <li className="nav-item">
                <Link className="nav-link left" to="/sports">Sports</Link>
              </li>
              <li className="nav-item">
                <Link className="nav-link" to="/business">Business</Link>
              </li>
              <li className="nav-item">
                <Link className="nav-link" to="/entertainment">Entertainment</Link>
              </li>
              <li className="nav-item">
                <Link className="nav-link" to="/technology">Technology</Link>
```

```
            </li>
            <li className="nav-item">
              <Link className="nav-link" to="/science">Science</Link>
            </li>
          </ul>
          {/* <form class="d-flex" role="search">
        <input class="form-control me-2" type="search" placeholder="Search" aria-
label="Search" onChange={props.btn} value={props.search}/>
        <button class="btn btn-outline-success" type="submit">Search</button>
      </form> */}
          </div>
        </div>
      </nav>
    </div>
  )
}
export default Navbar
```

NEWSCARDS.JS

```
import React from 'react'
import NewsCardItem from './NewsCardItem'

const NewsCards = ({ articles, activeArticle }) => {
    return(

        <div className ="row">
            <div className="container news-container">
                {articles.map((article, i) => (
                        <div className="col s12 m6 l4">
                         <NewsCardItem article={article} activeArticle={activeArticle}  i={i} />
                        </div>
                 ))}
            </div>
        </div>

    )
}

export default NewsCards
```

NEWS.JS

```
import React, { useEffect, useState } from "react";
import NewsItem from "./NewsItem";
import Spinner from "./Spinner";
import PropTypes from 'prop-types'
import InfiniteScroll from "react-infinite-scroll-component";



const News =(props)=>{
  const [articles,setArticles]=useState([])
  const [loading,setLoading]=useState(true)
  const [page,setPage]=useState(1)
  const [totalResults,setTotalResults]=useState(0)




  const updateNews=async ()=>{
    const url = `https://newsapi.org/v2/top-
headlines?country=${props.country}&category=${props.category}&apiKey=${props.apiKey}
&page=${page}&pagesize=${props.pageSize}`;
    setLoading(true);
    let data = await fetch(url);
    let parsedData = await data.json();
    console.log(parsedData);
    setArticles(parsedData.articles)
    setTotalResults(parsedData.totalResults)
    setLoading(false)

  }
  function capitalizeFirstLetter(string) {
```

```
  return string.charAt(0).toUpperCase() + string.slice(1);
 }
 useEffect(() => {
  document.title=`${capitalizeFirstLetter(props.category)} - NewsApp`;
  updateNews();

 },[])

 const fetchMoreData =async () => {

  setPage(page+1)
  const url = `https://newsapi.org/v2/top-
headlines?country=${props.country}&category=${props.category}&apiKey=${props.apiKey}
&page=${page+1}&pagesize=${props.pageSize}`;
  let data = await fetch(url);
  let parsedData = await data.json();
  setArticles(articles.concat(parsedData.articles))
  setTotalResults(parsedData.totalResults)
 };


  return (
   <>
    <h1 className="text-center-h"
     style={{margin:'35px',marginTop:'90px',color:props.mode}}>Top Headlines on
{capitalizeFirstLetter(props.category)}</h1><br/>
    {loading && <Spinner />}
    <InfiniteScroll
     dataLength={articles.length}
     next={fetchMoreData}
     hasMore={articles.length !== totalResults}
```

```
        loader={<Spinner />}
      >
      <div className="container">
            <div className="row">
        {articles.map((art) => {
        return (
          <div className="col-md-4" key={art.url}>
            <NewsItem
              title={art.title}
              description={art ? art.description : ""}
              imgUrl={art.urlToImage}
              newsUrl={art.url}
              author={art.author}
              date={art.publishedAt}
              source={art.source.name}
            />
          </div>

        );
       })}
      </div>
      </div>
            </InfiniteScroll>
      <br/>

    </>
  );
}
```

```
News.defaultProps = {
  country:'in',
  pageSize:5,
  category:'general',

}

News.propTypes = {
  country:PropTypes.string,
  name:PropTypes.number,
  category:PropTypes.string,
}

export default News;
```

NEWSCARDITEM.JS

```
import React, { useState, useEffect , createRef } from 'react'
import classNames from 'classnames'

const NewsCardItem = ({ article, i, activeArticle }) => {
  const { description, publishedAt, source, title, url, urlToImage } = article
  const [elRefs, setElRefs] = useState([])
  const scrollToRef = (ref) => window.scroll(0, ref.current.offsetTop - 50)


  useEffect(() => {
     // window.scroll(0, 0)
      setElRefs((refs) => Array(20).fill().map((_, j) => refs[j] || createRef()))
  },[])


  useEffect(() => {
      if( i === activeArticle && elRefs[activeArticle]){
          scrollToRef(elRefs[activeArticle])
      }
  },[i, activeArticle, elRefs])
  return (
    <div ref={elRefs[i]} className={ classNames(
      "card medium sticky-action z-depth-3 news-card",
      activeArticle === i ? "active-card" : null) }>


     <div className="card-image">
        <a href={url} target="_blank" rel="noopener noreferrer">
            <img src={urlToImage}  alt="" />
        </a>
      </div>
```

```
      <div className="card-content news-card-content">
        <div className="news-card-details">
          <p>{(new Date(publishedAt)).toDateString()}</p>
          <p>{source.name}</p>
        </div>
        <span className="card-title news-card-title activator">
          <span  className="card-title-height">{title}</span>
          <i className="material-icons indigo-text">expand_more</i>
        </span>
      </div>

      <div className="card-reveal">
        <span className="card-title news-card-title activator">{title}
          <i className="material-icons indigo-text">expand_less</i>
        </span>
        <p className="news-card-text activator">{description}</p>
      </div>

      <div className="card-action news-card-action">
        <a href={url} className="waves-effect waves-light btn-small news-btn">Learn
More</a>
        <a href={url} className="p-num">{ i + 1 }</a>
      </div>
    </div>
  )
}
export default NewsCardItem
```

SPINNER.JS

```
import React  from 'react'
import Loading from '../img/Loading.gif'
const Spinner=()=> {
    return (
      <div className="text-center ">
        <img src={Loading} alt="Loading"/>
      </div>
    )
  }


export default Spinner
```

```
import React, { useEffect, useState } from "react";
import NewsItem from "./NewsItem";
import Spinner from "./Spinner";
import PropTypes from 'prop-types'
import InfiniteScroll from "react-infinite-scroll-component";


const News =(props)=>{
  const [articles,setArticles]=useState([])
  const [loading,setLoading]=useState(true)
  const [page,setPage]=useState(1)
  const [totalResults,setTotalResults]=useState(0)
```

```
  const updateNews=async ()=>{
    const url = `https://newsapi.org/v2/top-
headlines?country=${props.country}&category=${props.category}&apiKey=${props.apiKey}
&page=${page}&pagesize=${props.pageSize}`;
    setLoading(true);
    let data = await fetch(url);
    let parsedData = await data.json();
    console.log(parsedData);
    setArticles(parsedData.articles)
    setTotalResults(parsedData.totalResults)
    setLoading(false)

  }
  function capitalizeFirstLetter(string) {
    return string.charAt(0).toUpperCase() + string.slice(1);
  }
  useEffect(() => {
    document.title=`${capitalizeFirstLetter(props.category)} - NewsApp`;
    updateNews();

  },[])

 const fetchMoreData =async () => {
    setPage(page+1)
    const url = `https://newsapi.org/v2/top-
headlines?country=${props.country}&category=${props.category}&apiKey=${props.apiKey}
&page=${page+1}&pagesize=${props.pageSize}`;
    let data = await fetch(url);
    let parsedData = await data.json();
   setArticles(articles.concat(parsedData.articles))
   setTotalResults(parsedData.totalResults)
```

```
};
        return (
    <>
      <h1 className="text-center-h"
       style={{margin:'35px',marginTop:'90px',color:props.mode}}>Top Headlines on
{capitalizeFirstLetter(props.category)}</h1><br/>
      {loading && <Spinner />}
      <InfiniteScroll
        dataLength={articles.length}
        next={fetchMoreData}
        hasMore={articles.length !== totalResults}
        loader={<Spinner />}
      >
      <div className="container">
            <div className="row">
        {articles.map((art) => {
        return (
          <div className="col-md-4" key={art.url}>
            <NewsItem
              title={art.title}
              description={art ? art.description : ""}
              imgUrl={art.urlToImage}
              newsUrl={art.url}
              author={art.author}
              date={art.publishedAt}
              source={art.source.name}
            />
          </div>
        );
      })}
      </div>
      </div>
```

INFOCARDS.JS

```
import React from 'react'
import InfoCardItem from './InfoCardItem'
import info from '../../info'
const InfoCards = () => {
        return(
            <div className ="row info-row">
              {info.infoCards.map(infoCard => (
                  <div className="col s12 l6 info-card-container">
                   <InfoCardItem infoCard={infoCard} />
                  </div>
               ))}
            </div>
        )
}

export default InfoCards
```

INFOCARDSITEM.JS

```
import React, { useState } from 'react'

const InfoCardItem = ({ infoCard }) => {
    const [showText, setShowText] = useState(false);

    return (
        <div className="card info-card">
            <div className="card-content white-text">
                <span className="card-title info-card-title">{infoCard.title}</span>
                <span  className="info-card-info">
                        {infoCard.info && infoCard.info.split(', ').map(g => (
                            <span key={g}>{g}<span className="bar">|</span></span>
                        ))}
                 </span>
            </div>
            <div className="card-action info-card-action">
                <a  href="#!"
                    className="btn-floating btn-small waves-effect waves-light orange tooltipped"
                    data-tooltip="Ask Microphone for News with Above Phrase"
                    data-position="right"
                    onMouseEnter={() => setShowText(true)}
                    onMouseLeave={() => setShowText(false)} >
                        <i className="material-icons">comment</i>
                </a>
                <a href="#!" className={showText ? "info-card-text-highlight" : "info-card-
text"}>{infoCard.text}</a>
            </div>
        </div>
    )
}

export default InfoCardItem
```

PACKAGE.JSON

```
{
  "name": "react-ai-news",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@alan-ai/alan-sdk-web": "^1.8.54",
    "@testing-library/jest-dom": "^6.1.4",
    "@testing-library/react": "^14.0.0",
    "@testing-library/user-event": "^14.5.1",
    "bootstrap": "^5.3.2",
    "classnames": "^2.3.2",
    "materialize-css": "^1.0.0",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-infinite-scroll-component": "^6.1.0",
    "react-router-dom": "^6.19.0",
    "react-scripts": "5.0.1",
    "words-to-numbers": "^1.5.1"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": "react-app"
  },
  "browserslist": {
    "production":  [
    ">0.2%",
      "not dead",
```

```
    "not op_mini all"
  ],
  "development": [
    "last 1 chrome version",
    "last 1 firefox version",
    "last 1 safari version"
  ]
 }
}
```

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" type="image/svg+xml" href="/favicon.svg">
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>


    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />

    <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
    <link href="https://fonts.googleapis.com/css2?family=Orbitron:wght@400;900&display=
swap" rel="stylesheet">
    <link
href="https://fonts.googleapis.com/css2?family=Open+Sans&family=Roboto&display=swap"
rel="stylesheet">
    <title>News Reader Interface</title>
```

```
  </head>
  <body>
    <div class="loader-container">
      <div class="loader"></div>
    </div>
    <div id="root"></div>
  </body>
</html>
```

APP.CSS

```
body {
  background-color:black;
  /* background-image: url('img/ai-speaks.jpg'); */
  background-position: center center;
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
  font-family: 'Orbitron', sans-serif;
  font-weight: 400;
  height: 100vh;
}

.text-center-h{
  text-align: center;
  color:whitesmoke;
}
.nav{
  background-color:rgb(26, 12, 63)
}

/*PRELOADER*/
.loader-container {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  height: 100vh;
  background: #000729;
  color: #fff;
  display: flex;
  justify-content: center;
  align-items: center;
  z-index: 100;
  opacity: 1;
  transition: opacity 4s ease;
```

```
}

.loader-container.finish {
  opacity: 0;
  pointer-events: none;
  /*display: none;*/
}

.loader {
  border: 16px solid #f3f3f3;
  border-radius: 50%;
  border-top: 16px solid #ffa500;
  width: 120px;
  height: 120px;
  animation: spin 2s linear infinite;
  transition: opacity 2s ease;
}

.page-container {
  margin-top: 70px;
  padding-bottom: 70px !important;
}

/*NAV*/
.alan-logo {
  width: 60px;
  height: 60px;
  margin-left: 100px;
  margin-top: 10px;
}

nav {
  background-color: rgba(0, 0, 0, 0.2);
  display: flex;
  align-items: center;
  height: 10vh;
```

```
  padding-left: 20px;
}
.nav-title {
  font-weight: 900;
  margin-left: 100px;
  color: rgb(255, 255, 255);
}

.nav-title > span {
  color: rgb(189, 181, 209);
}

.brand-logo > img {
  width: 25%;
  height: 25%;
  margin-top: 45px;
}

.right {
  margin-right: 100px;
  margin-top: 40px;
}

/*FOOTER*/

.page-footer {
  position: fixed;
  left: 0;
  bottom: 0;
  width: 100%;
  height: 7vh;
  background-color: rgba(0, 0, 0, 0.4);
}

.seal-bottom {
  position: absolute;
```

```
  bottom: 10px;
  left: 4%;
}
.seal-bottom > img {
  width: 25%;
  height: 25%;
}

/*INFO CARDS*/
.info-row {
  width: 50%;
  display: flex;
  flex-wrap: wrap;
  margin-left: 100px;
  margin-top: 40px;
}

.info-card-container {
  width: 90%;
}

.info-card {
  background-color: rgba(0, 0, 0, 0.2);
  border-radius: 8px;
  font-size: 10px;
}

.info-card-title {
  color: #1550c7;
}

.info-card-info {
  color: white;
  text-transform: uppercase;
  padding-top: 5px;
  display: flex;
  flex-wrap: wrap;
```

```
  justify-items: center;
}

.info-card-action {
  font-size: 11px;
}

.bar {
  color: rgb(162, 144, 206);
  padding: 0px 6px;
}

.info-card-text {
  color: rgb(189, 181, 209) !important;
  padding-left: 8px;
}

.info-card-text-highlight {
  color: #1550c7 !important;
  font-weight: bold;
  padding-left: 8px;
  animation: fadein 2s;
}

@keyframes fadein {
  from {
    opacity: 0;
  }
  to {
    opacity: 1;
  }
}

.material-tooltip {
  background-color: #ffa500;
  color: #ffff;
  font-size: 10px;
```

```
  font-weight: 600;
  text-transform: uppercase;
  margin: 44px 0 0 -47px;
  border-radius: 3px;
  display: flex;
  align-items: center;
  justify-content: center;
}

/*NEWS CARD*/

.container {
  margin: 0 auto;
  max-width: 1280px;
}

.news-card {
  font-size: 11px;
  border-radius: 8px;
  overflow: hidden;
  background-color: rgb(248, 244, 244);
  /* height: 60vh;*/
}

.news-card-content {
  font-family: 'Roboto';
}

.card-image {

  width: 100%;
  height: 45% !important;
  min-height: 45% !important;
  max-height: 45% !important;
}
.card-img img {
```

```
  width: 100%;
  object-fit: cover !important;
}

.news-card-details {
  font-family: 'Orbitron', sans-serif;
  display: flex;
  justify-content: space-between;
  color: rgb(7, 7, 102) !important;
}

.news-card-title {
  padding-top: 10px;
  text-transform: capitalize;
  font-weight: 600 !important;
  font-size: 16px !important;
}

.card-title-height {
  display: inline-block;
  line-height: 1.6;
  max-height: 80px;
  min-height: 80px;
  text-overflow: ellipsis;
}

.news-card-action {
  padding: 0 16px 8px 16px;
  display: flex;
  justify-content: space-between;
  background-color: white !important;
  color: rgb(7, 7, 102) !important;
}

.news-btn {
  background-color: #1550c7;
}
```

```
    .news-btn:hover {
      background-color: #1a4492;
    }

    .active-card {
      border-bottom: 10px solid #1550c7;
    }

    /* width */
    ::-webkit-scrollbar {
      width: 10px;
    }

    /* Track */
    ::-webkit-scrollbar-track {
      background: #f1f1f1;
    }

/* Handle */
    ::-webkit-scrollbar-thumb {
      background: #888;
    }

    /* Handle on hover */
    ::-webkit-scrollbar-thumb:hover {
      background: #555;
    }

    @keyframes spin {
      0% {
        transform: rotate(0deg);
      }
      100% {
        transform: rotate(360deg);
      }}
```

ALAN AI CODE

```
// {Name: Basic_example_for_AI_assistant}
// {Description: Learn how to create a dialog script with voice/text commands and text corpus for question answering}

// Use this sample to create your own voice/text commands
intent('hello world', p => {
    p.play('(hello|hi there)');
});



title("News")
const page = 5;
const key = "5f6b52de8d4248f8b8c82b2cc0da48e4";
let savedArticles = []
let TOPICS = ["business", "entertainment", "general", "health", "science", "sports", "technology"];
let TOPICS_INTENT = [];
for (let i = 0; i < TOPICS.length; i++) {
    TOPICS_INTENT.push(TOPICS[i] + "~" + TOPICS[i]);
}
TOPICS_INTENT = TOPICS_INTENT.join('|') + '|';

function apiCall(p, command, param, callback) {
    let jsp = {
        url: "https://studio.alan.app/api_playground/" + command,
        strictSSL: false,
        method: 'POST',
        json: param,
        timeout: 5000,
    };
    api.request(jsp, (err, res, body) => {
        if (err || res.statusCode !== 200) {
            p.play(`(Sorry|) something went wrong (on the server|) ${err} ${res} ${body}`);
        } else if (body.error) {
            p.play(body.error);
```

```
        } else {
            callback(body);
        }
    });
}

intent(`(show|what is|tell me|what's|what are|what're|read) (the|) (recent|latest|) $(N news|headlines)
(in|about|on|) $(T~ ${TOPICS_INTENT})`,
    `(read|show|get|bring me) (the|) (recent|latest|) $(T~ ${TOPICS_INTENT}) $(N news|headlines)`,
    p => {
        let headlinesUrl = "https://newsapi.org/v2/top-
headlines?country=in&apiKey=5f6b52de8d4248f8b8c82b2cc0da48e4"
//


 let param = {}
        if (p.T.label) {
            param.category = p.T.label;
        }
        apiCall(p, 'getNews', param, response => {
            if (!response.error) {
                let headlines = [];
                let images = [];
                let res = JSON.parse(response.data);
                let articles = res.articles;
                savedArticles = articles;
                let max = Math.min(page, articles.length);
                for (let i = 0; i < max; i++) {
                    let article = articles[i];
                    let name = article.source.name;
                    let author = article.author;
                    let title = article.title;
                    let description = article.description;
                    let image = article.urlToImage;
                    if (title) {
                        headlines.push(title);
                        images.push(image);
                    }
```

```
            }
            p.play({ command: 'newHeadlines', articles });
            p.play(`Here are the (latest|recent) about $(N articles) $(N headlines).`,
                `Here's the (latest|recent) $(N news).`);
            p.play('Would you like me to read the headlines?');
            p.then(confirmation);
        } else {
            console.log(response.error);
        }
    });
});

const confirmation = context(() => {
    intent('yes', async (p) => {
        console.log("savedArticles",savedArticles);
        for(let i = 0; i < savedArticles.length; i++){
            p.play({ command: 'highlight', article: savedArticles[i]});
            p.play(`${savedArticles[i].title}`);
        }
    })

    intent('no', (p) => {
        p.play('Sure, sounds good to me.')
    })
})

intent('open (the|) (article|) (number|) $(number* (.*))', (p) => {
    if(p.number.value) {
        p.play('Sure, I am opening it in a new tab');
```

```
        p.play({ command:'open', number: p.number.value, articles: savedArticles})
    }
})




intent('(go|) back', (p) => {
    p.play('Sure, going back');
    p.play({ command: 'newHeadlines', articles: []})
})

intent(`What does this app do?`, `How does this work?`, `What can I do here?`, `How should I use this?`,
    reply(`This is a news project, and you can provide the most recent headlines in mainstream media`
+
        ` Just ask me anything about the news, and I will try to answer it`));

intent(`(types|categories|topics) (of|in) the news (now|)`, `What (types|categories|topics) of news do you have?`,
    reply(`We provide news on ` + TOPICS.join(", ")));


        }
        apiCall(p, 'getNews', param, response => {
            if (!response.error) {
                let headlines = [];
                let images = [];
                let res = JSON.parse(response.data);
                let articles = res.articles;
                savedArticles = articles;
                let max = Math.min(page, articles.length);
                for (let i = 0; i < max; i++) {
                    let article = articles[i];
                    let name = article.source.name;
                    let author = article.author;
                    let title = article.title;
```

```
                let description = article.description;
                let image = article.urlToImage;
                if (title) {
                    headlines.push(title);
                    images.push(image);
                }
            }
            p.play({ command: 'newHeadlines', articles });
            p.play(`Here are the (latest|recent) about $(N articles) $(N headlines).`,
                `Here's the (latest|recent) $(N news).`);
            p.play('Would you like me to read the headlines?');
            p.then(confirmation);
        } else {
            console.log(response.error)
                                                                                    });}
});
    const confirmation = context(() => {
        intent('yes', async (p) => {
            console.log("savedArticles",savedArticles);
            for(let i = 0; i < savedArticles.length; i++){
                p.play({ command: 'highlight', article: savedArticles[i]});
                p.play(`${savedArticles[i].title}`);
            }

        })
        intent('no', (p) => {
            p.play('Sure, sounds good to me.')
        })
    })
    intent('open (the|) (article|) (number|) $(number* (.*))', (p) => {
        if(p.number.value) {
            p.play('Sure, I am opening it in a new tab');
            p.play({ command:'open', number: p.number.value, articles: savedArticles})
        }
    })

    intent('(go|) back', (p) => {
```

8. REFERENCES

Creating a news application with a voice assistant is a great project idea. To support your project in your report, you can provide references to existing technologies, frameworks, and libraries that are commonly used for such applications.
Here's a sample list of resources and references that you can include in your report:

1. **Speech Recognition and Synthesis:** - Google Cloud Speech-to-Text API: [Google Cloud Speech-to-Text](https://cloud.google.com/speech-to-text) - Amazon Polly (Text-to-Speech): [Amazon Polly](https://aws.amazon.com/polly/)
2. **News API for Content Retrieval:** - News API: [News API](https://newsapi.org/) - New York Times API: [NY Times API](https://developer.nytimes.com/apis)
3. **Mobile App Development Frameworks:** - Flutter (for cross-platform development): [Flutter](https://flutter.dev/) - React Native: [React Native](https://reactnative.dev/)
4. **Voice Assistant Integration:** - Google Assistant SDK: [Google Assistant SDK](https://developers.google.com/assistant/sdk) - Amazon Alexa Skills Kit: [Alexa Skills Kit](https://developer.amazon.com/alexa-skills-kit)
5. **Database for Storing News Articles:** - Firebase Realtime Database: [Firebase Realtime Database](https://firebase.google.com/products/realtime-database) - MongoDB (NoSQL database): [MongoDB](https://www.mongodb.com/)
6. **Authentication and User Management:** - Firebase Authentication: [Firebase Authentication](https://firebase.google.com/products/auth) - Auth0: [Auth0](https://auth0.com/)
7. **Version Control for Collaboration:** - GitHub: [GitHub](https://github.com/) - Bitbucket: [Bitbucket](https://bitbucket.org/)
8. **Documentation and Report Writing:** - Markdown Guide: [Markdown Guide](https://www.markdownguide.org/) - LaTeX (for more advanced formatting): [LaTeX](https://www.latex-project.org/) Make sure to adapt this list based on the specific technologies and platforms you choose for your project. Provide proper citations and references for each source in your report to give credit to the original authors and developers. Good luck with your news application with a voice assistant!

TECHNICAL BIOGRAPHY

ARBAAZ BAIG .A (RRN:220282601011) was born on 8th Nov 2001, in Chennai, TamilNadu. He did his Schooling in Dominic Savio Higher Secondary School. He did his Bachelor of Computer Application in The New College ,Royapettah ,Chennai. He is currently pursuing Master of Computer Application in B.S.Abdur Rahman Crescent Institute of Science and Technology,Vandalur.