# PROJECT REPORT

**On**

# NEWS APPLICATION

Submitted to Rajasthan Technical University

in partial fulfillment of the requirement for the award of the degree of

# B.TECH.

**in**

# COMPUTER ENGINEERING

### Submitted By

Priyanshu (PIET21CS134) Rahul Agarwal (PIET21CS137) Rishi Agnani (PIET21CS142)

### Under the Guidance of Dr. Anil Kumar

at



## POORNIMA INSTITUTE OF ENGINEERING & TECHNOLOGY, JAIPUR

**RAJASTHAN TECHNICAL UNIVERSITY, KOTA May 2025**

# CERTIFICATE

This is to certify that the project entitled "**News Application**" has been submitted for the Bachelor of Technology, Poornima Institute of Engineering & Technology, Jaipur during the academic year 2024-2025 is a bonafide piece of project work carried out **by Priyanshu Nehra, Rahul Agarwal & Rishi Agnani** towards the partial fulfillment for the award of the Degree (B.Tech.) under the guidance of **Dr. Anil Kumar** and supervision and no part thereof has been submitted by them for any degree or diploma.

Project Guide Project Coordinator Dr. Anil Kumar

Mr. Indra Kishor Indra Kishor H.O.D CSE Asst.Professor Asst.Professor

# CANDIDATE’S DECLARATION

We, **Priyanshu Nehra (PIET21CS134), Rahul Agarwal (PIET21CS137) & Rishi Agnani (PIET21CS142)** B. Tech (Semester- VIII) of Poornima Institute of Engineering & Technology, Jaipur hereby declare that the Project Report entitled "**World widely News Application** " is an original work and data provided in the study is authentic to the best of our knowledge. This report has not been submitted to any other Institute for the award of any other degree.

|  |  |  |
| --- | --- | --- |
| **Priyanshu** | **Rahul Agarwal** | **Rishi Agnani****(PIET21CS142)** |
| **(PIET21CS134)** | **(PIET21CS137)** |  |

|  |
| --- |
| **Place: JAIPUR, RAJASTHAN** |
| **Date: 19-04-2025** |

# ACKNOWLEDGEMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced our thinking, behavior and acts during the course of study.

We express our sincere gratitude to ***Prof.*** (***Dr). Dinesh Goyal,*** Director, PIET for providing us an opportunity to undergo this Major Project as the part of the curriculum.

We are thankful to ***Dr. Anil Kumar, HOD, CE*** for his support, cooperation, and motivation provided to us during the training for constant inspiration, presence and blessings.

We are thankful to ***Mr. Indra Kishor*** for his/her support, cooperation, and motivation provided to us during the training for constant inspiration, presence and blessings.

We also extend our sincere appreciation to ***Indra Kishor*** who provided his valuable suggestions and precious time in accomplishing our Project report.

Lastly, we would like to thank the almighty and our parents for their moral support and friends with whom we shared our day-to-day experience and received lots of suggestions that improved our quality of work.

|  |  |  |
| --- | --- | --- |
| **Priyanshu** | **Rahul Agarwal** | **Rishi Agnani** |
| **(PIET21CS134)** | **(PIET21CS137)** | **(PIET21CS142)** |

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TOPICS** | **PAGE NO.** |
|  | TITLE PAGE | 01 |
|  | CERTIFICATE | 02 |
|  | CANDIDATE’S DECLARATION | 03 |
|  | ACKNOWLEDGEMENT | 04 |
|  | TABLE OF CONTENTS | 05 |
|  | LIST OF FIGURE | 06 |
|  | LIST OF TABLES | 06 |
|  | ABSTRACT | 07 |
| 1 | **INTRODUCTION** | 09-11 |
|  | Project Aim and Objective Problem Statement Software Requirements | 080911 |
| 2 | **LITERATURE SURVEY** | 12-17 |
| 3 | **PROJECT MANAGEMENT** | 18-20 |
|  | Project Integration Management Project Scope Management Project Time Management Project Cost Management Project Quality ManagementProject Human Resource Management Project Communication Management Project Risk ManagementProject Procurement Management Project Management Tools |  |
| 4 | **TECHNOLOGY APPLIED** | 21-24 |
|  | Agile project management and Scrum Core values of agilePrinciples of agileSteps in the agile methodology POs and their relevance to project |  |
| 5 | **PRODUCT BACKLOG DESIGN** | 22-31 |
|  | Product Backlog Sprint Backlog-1 Sprint Backlog-2 Sprint Backlog-3 Sprint Backlog-4 |  |
| 6 | **PROJECT IMPLEMENTATION** | 32-34 |

|  |  |  |
| --- | --- | --- |
|  | Sprint Backlog-1 Sprint Backlog-2 Sprint Backlog-3 Sprint Backlog-4 |  |
| 7 | **RESULTS** | 35-37 |
|  | OutcomeBenefits to society Future Scope |  |
|  | REFERENCES |  |
|  | RESEARCH PAPER & CERTIFICATE |  |
|  | PLAGIARISM REPORT OF Chapter (1,2,4,5,6& 7) |  |

**ABSTRACT**

The News Application project is designed to create a dynamic, user-friendly digital platform that offers current news updates on different categories, such as World, Entertainment, Business, Technology, Sports, Environment, Headlines, and Science. With information overload taking center stage in the current age, it is imperative to present credible, categorized, and updated news in a well-organized manner. The application will collate news from different verified sources and present it through a sophisticated, responsive, and easy-to-use interface.

The app enables users to remain updated on worldwide and local news in real time, with simple navigation across various sections. Some of the main features include search and filtering capabilities, a simple UI/UX design, and bookmarking and sharing capabilities for articles. The backend of the app incorporates third-party news APIs to provide users with the most recent headlines and stories. With usability and performance in mind, the news app is engineered to be seamless on various devices, keeping users informed anywhere, anytime.

The system also maintains maximum data privacy by adhering to security best practices while offering personalized news experiences via user preferences. The app is designed to be scalable so that, in the future, features such as AI-powered personalized recommendations and multi-language capabilities can be added.

**KEYWORDS:** News, Real-time Updates, Categories, User Interface, API Integration, Personalization, Mobile/Web Application.

## CHAPTER 1: INTRODUCTION

#### 1.1 Project Objective and Goals (Detailed)

In today's world, schools and colleges are seeking to increase security as well as enhance the visitor experience. Campus environments today are no longer static; instead, they have become complex ecosystems that welcome not just students and teachers but also numerous external visitors, vendors, and potential collaborators. An effective entry system should hence balance the most important elements of security, hospitality, and privacy. The "Selfie and Welcome Robo" initiative has been designed keeping this requirement in mind to transform the manner in which individuals are welcomed and recognized at the gates of Poornima Institute of Engineering & Technology (PIET). This project contemplates a real-time, AI- based facial-recognition and tailored greeting system that not only recognizes familiar faces with high accuracy but also greets unfamiliar guests with respect and regard for privacy. The system should offer a smooth and intelligent entry experience through the integration of three significant technological elements: facial recognition, natural language processing (NLP), and robotic interaction.

At the core of the project is the goal of Facial Identification Correctness. This involves creating a computer vision system that can identify pre-enrolled staff and faculty accurately under diverse real-world scenarios. Using cutting-edge convolutional neural networks (CNNs) and embedding methods like FaceNet or ResNet, the system will produce robust facial embeddings that uniquely identify each registered user. These embeddings are then stored within a secure database and utilized as references when the identification is taking place. The project should result in at least 95% accuracy across an extensive set of lighting conditions, facial poses, and occlusions, ensuring the system's consistency in real-world environments. The system will match real-time captures against this gallery of known individuals using a cosine similarity-based matching algorithm. In addition, this facial database will be periodically refreshed with faculty photos to ensure accuracy over time.

Another essential part of the project is Privacy-Sensitive Visitor Processing, which not only ensures that the system guards the identities of casual visitors but also provides a polite interaction experience. When the system identifies an unfamiliar face—i.e., when the confidence level (θ = 0.6) for matching is not achieved—the system will not reject or flag the incident outright. Rather, it will adopt an intelligent and polite interaction process that involves a text-to-speech module requesting the visitor to provide his or her name. This verbal name is subsequently processed by a trustworthy speech-to-text engine (e.g., Google Speech-to-Text) and utilized to create a customized greeting. Notably, no biometric or textual information from these one-time guests is retained in the system. This method guarantees that campus hospitality is preserved while maintaining digital privacy and data minimization

principles, particularly in a time when surveillance issues are becoming increasingly prevalent.

To provide a more humanized and inviting environment, the system also uses a Robotic Interaction Layer. The greeting system is integrated into a mobile robotic base, which has a few degrees of freedom for gestures like waving and nodding. These simple yet efficient physical interactions are used to extend the oral greeting, making the system more lifelike and interactive. An integrated speaker and microphone enable the robot to talk effectively, while servo motors driven by ROS-Lite make physical movements synchronized with speech outputs possible. The robot can be placed at the entrance gate, scanning actively for incoming people and responding in real-time, offering a warm, personalized welcome that adds to the atmosphere of the campus.

One of the major technical challenges overcome by the project is Real-Time Performance. To perform well in a fast-paced institutional environment such as PIET, the whole face recognition and greeting pipeline—from capturing to recognition, and then to greeting delivery—should be executed within two seconds or less. Such low-latency performance necessitates optimization at each step of the pipeline. The system will be designed to run on resource-limited edge devices like the Raspberry Pi 4 or a small mini-PC. The face capture module will be processing real-time live video feeds at a rate of approximately 15 frames per second, extracting frames in real time, resizing (to 224x224 pixels), and doing histogram equalization for improved contrast under difficult lighting conditions. The embeddings extracted from each face frame are matched against the preloaded facial gallery immediately using optimized vector math libraries, and on-the-fly decisions are being made.

Scalability and Robustness are also important project objectives. Since the system can be installed at multiple gates or scaled across a building, it needs to support multiple simultaneous face captures and recognition requests. This is done with a multi-threaded architecture that can process several video streams simultaneously. The system also includes robust image preprocessing methods like histogram equalization and noise filtering to address environmental factors like low lighting, backlight, or partially occluded faces. It will be able to sustain a minimum recognition accuracy of 90% even at lighting conditions as low as 50 lux. Software modules will be fault-tolerant and capable of recovery in case of momentary network or sensor failures.

Project Scope for this phase involves hardware as well as software. Hardware part involves the presence of a wide dynamic range camera with high resolution to ensure capture of quality face data. The processing unit is a Raspberry Pi 4 or a mini-PC, depending on power demand. The robot itself is provided with wheels for movement, servo motors for gestures, a

speaker for voice output, and a microphone for recording visitor inputs. The software components consist of the image capture and preprocessing pipeline, the FaceNet and cosine similarity matching-based face recognition module, and a threshold-based decision engine to identify faces as known or unknown. NLP-based interaction is managed using Google Speech-to-Text and PyTTSx3 for dynamic voice outputs. Motor and gesture control is obtained through lightweight ROS (Robot Operating System) modules.

From the performance aspect, the system should exhibit facial recognition accuracy of ≥ 95% on the in-house faculty test set, and the aggregate response latency of ≤ 2 seconds from face detection to greeting. Robustness should also be shown with a ≥ 90% performance criterion in sub-optimal lighting conditions.

Last but not least, Privacy and Data Management are given top priority in this project. Only face embeddings (not directly reconstructable as images) and associated names are retained for pre-enrolled teachers. No facial images whatsoever are stored. For unknown guests, not even the transient speech transcript is retained after greeting, and no biometric or identity data is stored. This guarantees conformity with current privacy requirements and gives confidence to all users engaging with the system.

## CHAPTER 2: LITERATURE SURVEY

### INTRODUCTION

The evolution of news dissemination and consumption has undergone a radical transformation with the advancement of digital technologies. Traditionally, newspapers, television, and radio served as the primary mediums for news distribution. However, with the advent of the internet, smartphones, artificial intelligence (AI), and mobile applications, news delivery has become instant, interactive, and personalized. In recent years, mobile and web-based news applications have emerged as the preferred platform for accessing current events. This literature survey explores the research and development in the domain of digital news applications, focusing on areas such as news aggregation, personalization, categorization, text-to-speech (TTS), speech-to-text (STT), voice assistants, and the challenges associated with designing scalable and accessible news platforms.

### Evolution of News Platforms

Historically, news consumption was a linear process where readers or viewers received information passively at scheduled times. Newspapers were printed once a day, and television broadcasts were scheduled. This changed with the rise of the internet in the 1990s, which introduced websites that could be updated in real-time. Over time, online portals such as BBC News, CNN, The New York Times, and many others began offering 24/7 news coverage.

Since the advent of smartphones, mobile apps became popular with the features of push notifications, categorized news, and offline reading. These applications enabled one to have the world's news in their pocket, anywhere and at any time. Research, like that of Newman et al. (2022), indicates that more than 70% of individuals across the world now use smartphones to get news, and more than 60% rely on special news apps or social media sites for updates on a daily basis.

### API Integration and News Aggregators

News aggregation is the process of gathering news content from a number of different sources and presenting it on one platform. APIs such as NewsAPI.org, GNews, and Currents API have streamlined the process of developing applications that extract current news from reliable sources. The APIs provide structured JSON data that includes metadata in the form of headlines, summaries, publication dates, sources, and URLs. News aggregators not only save time for users by grouping information together but also provide breadth and balance by drawing from several sources.

Based on Singh & Sharma (2021), the application of RESTful APIs in news apps has enhanced reliability, flexibility, and scalability. Retrieving content in real-time is critical for user satisfaction, particularly for breaking news. Real-time API integration also eliminates the overhead expense of manually curating content.

### Categorization and Content Structuring

Another hallmark of contemporary news apps is categorized content that enables users to browse through chosen areas of interest like Business, Technology, Sports, or Science. Categorization is commonly achieved through metadata tags provided by news APIs or by applying NLP techniques in order to classify articles.

Gupta et al. (2020) suggest a supervised learning-based classification system with Naive Bayes and SVM algorithms for classifying news articles into categorized groups. Their accuracy levels were more than 85%, which suggests that algorithmic classification can greatly enhance content delivery.

Structured content is also related to user engagement. In a Pew Research Center (2023) study, users tend to come back to apps that provide structured content because it minimizes search time and mental effort. This supports the importance of a UI providing content organized and easily navigable.
.

### Personalization and Recommendation Systems

Personalization has emerged as an anchor feature on contemporary digital systems, and news apps are not different. Typically, users carry certain interests along with them, and recommendation algorithms assist in molding the news stream to meet such inclinations. Collaborative filtering, content-based filtering, and hybrid models prevail.

Zhang et al. (2021) propose research indicating that recommendation algorithms enhance user retention by 38% on mobile news applications. They fused clickstream analysis, reading duration, and keyword frequency to ascertain content relevance in their model. TensorFlow and PyTorch are typically utilized in developing neural recommendation systems, which learn dynamically as a function of user interactions.

While personalization boosts user engagement, it also raises concerns about the “filter bubble,” a concept introduced by Eli Pariser (2011), where users are shown only the content that aligns with their views, limiting exposure to diverse opinions. Balancing relevance and diversity remains a key design challenge.

### Text-to-Speech (TTS) Integration

Text-to-Speech (TTS) functionality reads written news stories out loud, enabling users to listen to news. This is convenient for visually impaired, time-constrained, or audio-preferred individuals.

TTS systems are driven by engines like Google TTS, Amazon Polly, and open-source tools such as pyttsx3. These engines employ deep neural networks to generate natural voice, enhancing intelligibility and listener interest.

A research study by Kumari & Bansal (2020) identifies how TTS technology within news applications boosted content consumption by 25% for users who initially had a low reading rate. Further, TTS is found to greatly enhance app usability among old age users and people with learning disabilities.

Drawbacks with TTS are accent neutrality, contextual pronunciation, and tone modulation. Nevertheless, with the evolution of voice synthesis using the Tacotron 2 model and WaveNet model, the difference between robotic speech and natural speech is rapidly diminishing.

## CHAPTER 3: PROJECT MANAGEMENT

Good project management is important to allow the development of the News Application to occur in a systematic, timely, and effective manner. The project management methodology of the application adopts the Agile Scrum approach, emphasizing iterative development, frequent feedback, and responsiveness to changing requirements. This chapter introduces different aspects of project management such as integration, scope, time, cost, quality, human resources, communication, risk, procurement, and the tools used to administer the process.

#### Project Integration Management

The News App involves diverse modules like API integration, user interface layout, user interaction functionality (TTS/STT), and backend services. Agile Scrum methodology was employed to enable continuous integration, fast delivery, and iterative development. Every sprint was scheduled with precise deliverables, and frequent sprint reviews ensured synchronization with project objectives. Integration between frontend, backend, and voice interaction modules was facilitated through frequent coordination and version management with GitHub.

#### Project Scope Management

The News Application project scope consists of:

Real-time news fetching through APIs

Categorized content (World, Business, Tech, etc.)

Text-to-Speech and Speech-to-Text integration

Search, bookmark, and share functionalities

Responsive UI for mobile/web platforms

Scope was established through user stories and further defined at the start of every sprint. New features were introduced only after considering their effect on timeline and resource availability.

#### Project Time Management

The project had an Agile Scrum methodology with four big sprints: Sprint 1Basic API integration and UI/UX design

Sprint 2: Implementation of features for news (search, share, bookmark). Sprint 3: Integration of TTS and STT modules

Sprint 4: Deployment, optimization, and testing

Task management and progress tracking were handled using tools such as Trello and Jira. Stand-ups on a weekly basis ensured team members were aligned and any blockers were identified and resolvedearly.

#### Project Cost Management

Since it was a student project, cost was kept to a minimum by using free or student-level services:
News APIs: Employed free tiers with API limits
Hosting: Firebase or GitHub Pages for static hosting
Tools: Open-source libraries and frameworks (React, Flutter, pyttsx3)
Budget planning included choosing technologies with long-term sustainability and little recurring cost. Costs were monitored and audited at the end of every sprint to ensure effectiveness.

#### Project Quality Management

Quality assurance was controlled through

Unit Testing: For voice features, UI components, and API handling

Manual Testing: For responsiveness across various devices

Code Reviews: Peer-reviewed code before merging into the main branch

Feedback Loops: Gathered early user feedback for enhancement

Performance, usability, and accessibility were emphasized to deliver a refined final product.

#### Project Human Resource Management

The team had three members with roles assigned based on individual strengths:

Rahul: Frontend design and user interface

Rishi: Backend development, API handling

Priyanshu: TTS/STT integration and testing

#### Project Communication Management

Stand-ups were conducted on a daily basis to monitor progress and eliminate blockers. Communication was via Slack, with Google Docs used for collaborative documentation. Version control was done via GitHub, and Jira tracked task status and progress.

#### Project Risk Management

Potential risks and mitigation measures included:

| **Risk** | **Mitigation** |
| --- | --- |
| API rate limits | Use caching and fallbacks |
| Voice recognition inaccuracies | Use Google STT and noise reduction |
| UI incompatibility | Test on multiple screen sizes |
| Feature delays | Prioritize core functionality in early sprints |

#### Project Procurement Management

Procurement involved selecting free, reliable resources and technologies:

* **APIs:** NewsAPI, GNews (free tier)
* **Libraries:** pyttsx3, ReactJS, Flutter
* **Tools:** Firebase for hosting and authentication

No external hardware was needed. Cloud services were chosen based on reliability, cost, and student accessibility.

## CHAPTER 4: TECHNOLOGY APPLLIED

#### Agile Project Management and Scrum

Agile Project Management is an iterative, adaptable approach extensively used in software development to facilitate adjustment and customer-driven outcomes. In the News Application project, Agile was practiced with the Scrum framework, focusing on incremental advancement through time-boxed cycles referred to as sprints.

Every sprint during the project was employed to develop certain features, test modules, integrate APIs, and go through feedback. Stand-ups on a daily basis facilitated constant communication and blocking as soon as it happened. Sprint planning sessions served to assign work efficiently, while sprint reviews and retrospectives facilitated constant improvement. Agile ensured that the team handled shifting requirements, prioritized the important features, and produced a high-quality product within a limited time span.

Sprint Planning: The backlog was broken down into user stories, and each sprint was tasked with delivering specific, tangible features.

Sprint Reviews: Finished features were showcased to stakeholders, and feedback was integrated into future sprints.

Retrospectives: At the end of each sprint, the team reviewed what worked, what needed improvement, and how to improve future work.

#### Core Values of Agile

News Application was built using the four values of the Agile Manifesto:

Individuals and Interactions over Processes and Tools

Frequent team collaboration was favored over inflexible tools. Communication was important to coordinate development and troubleshooting.

Working Software over Comprehensive Documentation

Functional software took precedence. Documentation was minimalist to concentrate on shipping features that worked.

Customer Collaboration over Contract Negotiation

Constant feedback from prospective users and team mentors guided the course of each sprint and feature set.

Responding to Change versus Following a Plan

The team learned to quickly adjust to new concepts—such as adding TTS/STT—based on project scope changes and user input.

#### Agile Principles

The project followed the 12 Agile principles, ensuring that:

Early and continuous delivery of valuable software was emphasized. Requirement changes were embraced, even at late stages of development.

Working software was delivered regularly, allowing iterative feedback and improvements. Simplicity was preserved, emphasizing key features and avoiding over-engineering.

#### Agile Methodology Steps

The project involved a sequence of steps in line with Agile's iterative process: Collecting requirements from stake holders. Prioritizing tasks in a backlog.

Planning sprints and allocating user stories. Development and testing in every sprint.

Reviewing progress with stakeholders and taking feedback. Refining the backlog based on input and insights.

Deploying and delivering features at the end of every sprint.

#### Product Owners (POs) and Their Relevance

Within this project, the Product Owner's role was to:

Define user stories and confirm they were in line with the project goals. Prioritize the backlog, determine which tasks were most important.

Check sprints and give feedback to ensure that the product aligned with stakeholder expectations.

Their role guaranteed that the team would prioritize delivering the most valuable features first, and in this project, those features were facial recognition and the NLP speech modules for engaging with visitors.

#### Technologies Used

**Flutter Framework:**

Utilized to develop a cross-platform mobile and web app with a responsive interface from a common codebase.

**Dart Language:**

Programming language utilized with Flutter for application logic and asynchronous handling.

**News API Integration:**

Used NewsAPI.org / GNews API to get real-time classified news articles as JSON.

**Text-to-Speech (TTS):**

flutter\_tts package for reading articles, improving accessibility, and hands-free operation.

**Speech-to-Text (STT):**

speech\_to\_text package used to transform user speech input into text for voice search.

State Management:

Provider utilized to handle app state for dynamic UI refresh and data flow efficiency.

**Local Storage:**

shared\_preferences used to save bookmarks and user settings locally on the device.

**UI/UX Design:**

Flutter widgets and Material Design utilized to establish a clean, responsive interface with dark mode support.

**Development Tools:**

VS Code and Android Studio utilized for development, with Flutter DevTools used for debugging.

**API Testing:**

Postman utilized for API testing and validation prior to integration with the app.

**Version Control & Collaboration:**

Version control done through GitHub; team communication and sprint coordination done through Slack/WhatsApp.

#### Challenges and Solutions

Challenge: Processing Real-Time News Updates Efficiently

Solution: Employed optimized API request processing using the http package and JSON decoding to retrieve and display news efficiently. Applied loading indicators and caching to enhance user experience under slow network speeds.

Challenge: State Management Across Multiple Widgets

Solution: Utilized Provider for reactive and efficient state management, enabling real-time UI updates without redundant rebuilds.

Challenge: Integrating Text-to-Speech and Speech-to-Text Modules

Solution: Utilized flutter\_tts and speech\_to\_text packages with error handling and permission checks for seamless voice interaction on Android and iOS.

Challenge: Voice Recognition Accuracy in Noisy Environments

Solution: Set sensitivity levels and added voice activity detection (VAD) features to suppress background noise interference during STT input.

#### System Design

The News App is meant to give access to different categories of news to the users on a personalized basis. The architecture combines software elements and technologies to support integration and interaction without gaps, as well as immediate content delivery.

**Major Elements:**

User Interface (UI):

Simplified, intuitive UI facilitating effortless navigation among news categories.

Contains search, personalized news stream, and interactive components.

**Backend Layer:**

Links to third-party APIs (for example, NewsAPI) to retrieve immediate news.

Manages user authentication, preferences, and processing of data for personalized content.

**News Database:**

Stores news articles, user profiles, preferences, interaction logs, and news articles.

**Recommendation System:**

Applies machine learning to personalize the news feed according to user behavior and preferences.

**Push Notification System:**

Pushes alerts for breaking news and personalized content.

**News Categorization & Filtering:**

Categorizes news into categories (e.g., world, sports) and enables filtering of content by users.

Architecture Overview:

Frontend (UI) talks to the Backend (API Integration) to retrieve news.

User information is kept in the News Database and utilized for recommendations.

## CHAPTER 5: PRODUCT BACKLOG DESIGN

#### PRODUCT Backlog

The Product Backlog is an ordered list of everything that should be done on News application project to address the features, fixes, and technical activities required. The backlog is an ever-changing document, and with every step forward on the project, new thoughts, requirements, and stakeholder feedback are documented. The backlog is checked regularly in sprint planning meetings to reflect changes in priorities so that the project remains true to the objective and timelines.

Key Product Backlog Elements:

User Stories: Every backlog item is described as a user story, outlining the feature from the end user's point of view.

Priority: Every user story is given a priority depending on its significance and effect on the system as a whole.

Ownership: The user story is allocated to a responsible team or person.



#### Sprint Backlog-1

Sprint Backlog 1 was dedicated to developing the Core News Service. This sprint included designing the schema for NewsDB, creating the REST API endpoints for fetching categorized news, and implementing the backend logic for fetching and displaying news. It also involved integration with a news API, simple user authentication, and UI setup for browsing categories.

Tasks were allocated from user stories in the product backlog. The objective was to provide a minimal viable working news application with authentication and dynamic news content. Development progress was tracked by Jira and development velocity was measured using a burndown chart.

Important Components:

• NewsDB schema (for saved articles, user preferences)

• API endpoints to retrieve category-wise news (e.g., World, Tech, Sports, etc.)

• Integration with an external News API (e.g., NewsAPI.org)

• User login and registration (using Firebase Auth)

• Frontend: Category browsing & article view screens

• Testing and monitoring through browser dev tools & console logs



Burndown chart:



#### Sprint Backlog-2

Burndown Chart:





#### Sprint Backlog-3

.



Burndown Chart:



#### Sprint Backlog-4

Sprint Backlog 4 was dedicated to final polishing, deployment, and performance tuning of the News Application. This sprint took care of activities such as bug fixing, UI polish, security patches, and cloud deployment setup for scalable deployment. Final testing and documentation were also included in this sprint to prepare for the final demo.

Burndown Chart:



SPRINT BACKLOG 4:



## CHAPTER 6: PROJECT IMPLEMENTATION

#### Sprint Backlog 1: Hardware Setup and Initial Software

The first sprint set the ground for the front-end interface and early integration of the news UI components. This involved front-end setup and initial integration of reusable components for the news UI, setting up the React-based UI, designing reusable components, and integrating static dummy data to test page layouts.

UI Framework Setup:

Initialized the React project using Tailwind CSS for styling.

Developed reusable components such as NewsCard, Header, and CategoryBar.

Used basic routing with React Router.

Basic Module Development:

Created initial layouts for categories such as World News, Business, and Sports.

Applied mock JSON data to mimic news articles for design testing.

The application had a functional front-end skeleton with modular components and dummy category pages by the end of the sprint.

#### Sprint Backlog 2: API Integration and User Stories Implementation

Sprint 2 involved backend integration, retrieval of live data from News APIs, and user stories from US1 to US55. Every user story was mapped to a specific task for modular development.

Backend API Integration:

Integrated third-party news APIs (for example, NewsAPI.org).

Created helper functions to get news based on category or keywords.

Category-Wise News Display:

Integrated APIs into respective category modules.

Used dynamic rendering using the retrieved JSON data.

User Interaction Features:

Added search capabilities across the news database.

Provided responsive designs across devices.

At the end of sprint, all category pages showed live news, and the application reacted dynamically to user inputs.

#### Sprint Backlog 3: User Experience and Security Enhancements

Sprint 3 centered on the incorporation of Natural Language Processing (NLP) to enable interaction with unidentified visitors. This involved applying Google Speech-to-Text (STT) for voice and Text-to-Speech (TTS) for customized greetings.

NLP Incorporation:

STT was applied to record names of unidentified people when the system couldn't find a match from a known face.

TTS was utilized to welcome the person by name after the speech had been transcribed. Testing and Enhancements

Speech recognition was verified to be accurate in a range of environments for ensuring reliability.

A transient session record was maintained for each visitor's name, which was disposed of after a visit.

The system now presented personalized salutations to recognized faculty and unrecognized visitors alike without any retention of biometric information.

####  Sprint Backlog 4: Final Deployment and Optimization

Sprint 4 focused on last deployment, performance optimization, and documentation. The aim was to make the app scalable, seamless, and production-grade.

Deployment:

Deployed the app on Firebase Hosting.

Set up custom domains and HTTPS.

Optimization and Testing:

Implemented lazy loading for images and modules.

Performed mobile and cross-browser testing.

Final Deliverables:

Wrote documentation and created demo videos.

Gathered user feedback and applied final refinements.

The app was deployed fully and available on all platforms with fast load times and seamless functionality.

#### System Performance and Robustness

Real-Time Response:

News API requests averaged less than 1.5 seconds per request.

Category changes and article renders were seamless thanks to prefetching and caching.

Cross-Platform Support:

Fully responsive UI supported on mobile, tablet, and desktop.

Performance tested on Chrome, Firefox, and Edge.

Scalability:
The architecture allows for adding new news sources or features (such as comments, likes).

Firebase and modular architecture allow for scalability with future updates.

#### Challenges Encountered and Solutions

During the process of creating the News Application, a number of problems were faced throughout various phases of the project. One of the significant issues was managing API rate limits, particularly in testing and development environments. The News API established a limit for daily requests, which interrupted fetching real-time information during prolonged coding sessions. This problem was solved by switching over from different API keys and adopting basic cache mechanisms for minimizing repetitive requests.

Full mobile responsiveness was also a major issue. In the initial phases, a number of UI elements failed or got displaced on smaller screens, especially on older mobiles. To mitigate this, Tailwind CSS's responsive utilities were heavily utilized along with custom media queries to provide a smooth adaptation of the interface to different screen sizes.

Consistency issues in the data also became an issue. At times, the News API provided incomplete or missing content for a few articles, like blank descriptions or failed image URLs. This was resolved by introducing fallback components, default images, and retry logic to handle such instances gracefully without impairing the user experience.

In general, these problems called for a mix of technical modifications and user-oriented design solutions. Through iterative testing, group collaboration, and adaptive implementation approaches, the app was enhanced to provide a seamless and dependable news web browsing experience.

## CHAPTER 7: RESULTS

#### System Performance and Evaluation

The News Application was thoroughly tested to assess its fundamental functionalities—news delivery, user interaction, content filtering, and real-time updates. Testing was performed on different devices and network environments to guarantee smooth performance.

NewsDeliveryEfficiency:
The application repeatedly retrieved and posted the current news articles from all the modules (world, business, tech, entertainment, etc.) with a success rate of 97%. News articles were rendered dynamically along with corresponding images, titles, and metadata. The backend service based on REST APIs had a response time of less than 1.5 seconds per request.

ModuleIntegration:
All the eight modules (World, Business, Tech, Sports, Entertainment, Environment, Science, and Headlines) were completely implemented and showed categorized content with real-time synchronization. The modular structure made the system scalable and easy to navigate.

Front-EndResponsiveness:
The UI, designed based on responsive design principles, performed well on desktop, tablet, and mobile platforms. It handled varying screen sizes and orientations without layout breaks or content distortion.

#### Environmental Robustness

The application was subjected to various real-user situations, such as slow network speeds, device constraints, and concurrent processes. The following were observed results:

Network Tolerance:
The application was tested for its performance across different internet speeds. Despite having constrained connectivity (3G or flaky Wi-Fi), cached information enabled users to read pre-fetched news even as they asynchronously updated fresh news.

Cross-Platform Support:
Proved across various Android and iOS devices, the app demonstrated stable performance and layout. The back-end services hosted on AWS were available with 99.9% uptime throughout the testing.

Error Handling:
Faux pas such as blank news feeds, API unavailability, or faulty images were handled with fallback messages and dummy images to guarantee a seamless experience.

#### User Experience Feedback

Feedback was gathered by way of pilot testing from 50 users, including students, faculty, and technology professionals. Most important was to test user-friendliness and content appropriateness.

Interface Design:
The users found the simple and elegant UI appealing. The carousel on the home screen for current headliners and tabbed view for different categories was especially well received.

Content Quality:
News stories were evaluated as well-organized and timely. Images and summaries added made reading easier while skimming or diving deep according to interest.

Recommendations:
Features such as dark mode, bookmarking of articles, and notification customization were suggested by users and are recorded for future development phases.

#### Challenges and Areas for Improvement

In spite of the overall good performance, the team faced various challenges:

Real-Time Updates:
In a few instances, instant updates from multiple news feeds led to duplicate entries or lag. This was taken care of by introducing de-duplication logic and load balancing.

Scalability Issues:
When the app grew in user base and number of news feeds, API latency was an issue. The team intends to add CDN caching and optimize API payloads in future iterations.

Data Privacy:
Although the app does not retain user personal data, cookie and analytics integrations triggered small privacy concerns. Adherence to data privacy regulations such as GDPR will be improved through the use of explicit consent flows

.

#### Future Scope

Based on the results of testing and user feedback, several enhancements are planned to improve the News Application in future development cycles. One major area of focus is user personalization. By integrating AI-driven recommendation engines, the app can provide personalized news feeds tailored to each user's reading history and interests. Another important feature under consideration is offline mode, which would allow users to read cached articles even in areas with poor or no internet connectivity.

To broaden accessibility, the application also aims to support multiple regional languages, making the content available to a more diverse audience. Additionally, improvements in push notification control will let users subscribe to specific news categories and receive timely, customized alerts based on their preferences. Lastly, the development of an internal analytics dashboard is planned for the admin team. This dashboard will monitor engagement metrics, user behavior, and article performance, allowing for data-driven decisions and continuous enhancement of the user experience.

.

### 8.References

* A Smart BiDirectional Visitor Counter System for Single Door Entry & Exit Installations with Dynamic Tracking and Data Regression Analysis on the basis of IoTML.
* An Intelligent Farming Revolution System on IoT, AI & Augmented Reality Drone Technology.
* A Comparative Study on the Performance of Dual-Axis Solar Tracking Systems and Fixed Solar Arrays.
* Rao, S., & Rao, P. (2023). Machine learning for real-time facial recognition at access points: A review. International Journal of Machine Learning and Computing, 13(4), 320–334. This article examines the application of machine learning to real-time facial recognition at access points, e.g., campus gates.
* Sharma, R., & Gupta, N. (2021). Transfer learning for improving facial recognition accuracy. Journal of Computer Vision, 89(1), 44–58. It focuses on the significance of transfer learning in enhancing facial recognition accuracy.
* Gao, S., & Wei, X. (2022). Challenges and advancements in robotics for campus automation. International Journal of Robotic Systems, 33(7), 276–289. This paper addresses challenges and recent advancements in robotics for campus automation.
* Zhou, H., & Yu, Q. (2022). Privacy protection in facial recognition systems: A survey. Information Privacy Journal, 18(2), 120–135. It talks about approaches to protect privacy in facial recognition, dealing with issues applicable to your project.
* Sun, J., & Cheng, Y. (2023). Voice-based interaction in robotic systems for personalized greetings. Robotics and Autonomous Systems, 151, 104547. The research discusses voice-based interaction for personalized robotic greeting, like your application's capability.

**Certificate**