# Preface

The digital age has ushered in an overwhelming amount of information, making it difficult to access quality news in a concise format. The One Point News project is developed with the goal of solving this issue by aggregating various news categories into one accessible, user-friendly web application. This report details the design, implementation, and features of the project.

# Acknowledgement

I would like to express my sincere gratitude to my mentors, peers, and the development community whose resources and documentation helped me learn and implement this project. A special thanks to OpenAI’s ChatGPT for continuous guidance in backend and frontend development and bug fixing throughout the process.

# Scope of the project

This project aims to create a single-point interface for news consumption across various categories like Global, Local, Sports, Finance, etc. It provides users with the latest top 5 headlines in a carousel, a categorized news feed, and functionality for liking and viewing each post independently. The backend stores all data in MongoDB and serves it dynamically via Express API endpoints.

# Keywords and definitions

- SPA: Single Page Application

- React.js: JavaScript library for building user interfaces

- Node.js: JavaScript runtime for backend development

- Express.js: Web framework for Node.js

- MongoDB: NoSQL database used to store post data

- API: Application Programming Interface

- Post: An individual news article shown in the feed

- Like/View: Interactions a user performs on each post

# References

- React Documentation – https://reactjs.org

- Express.js Guide – https://expressjs.com

- MongoDB Docs – https://mongodb.com/docs

- Vite – https://vitejs.dev

- News API (source of news)

# Project Introduction

Project Title: One Point News

Technology Stack: React.js, Node.js, Express.js, MongoDB, TailwindCSS, Lucide Icons

Database: MongoDB (local)

Type: Full-stack News Aggregator Web App

Main Features:

- Categorized news fetching

- Top 5 news carousel

- Likes and views tracking per post

- Filterable category tabs

- Responsive UI

Architecture:

- Frontend: Built with React and TailwindCSS. Utilizes useEffect, useState, and React Router.

- Backend: Built with Node.js, Express.js. Handles routes for fetching posts, liking a post, and tracking views.

- Database: MongoDB stores post interactions (likes, views).

# Screen shot of interface

Insert full-page screenshot of your app showing header, carousel, category tabs, and news feed.

Include additional screenshots showing post interactions: like and view counters updating.

# Introduction of classes and printing of source code

Backend: server.js

```

const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

const app = express();

app.use(cors());

app.use(express.json());

mongoose.connect("mongodb://localhost:27017/onepointnews", {

useNewUrlParser: true, useUnifiedTopology: true

});

const postSchema = new mongoose.Schema({

postId: String,

likes: Number,

views: Number,

});

const Post = mongoose.model('Post', postSchema);

```

Frontend: NewsFeed.jsx

```

useEffect(() => {

fetch(`http://localhost:5000/news?category=${selectedCategory}`)

.then(res => res.json())

.then(data => setNewsFeed(data));

}, [selectedCategory]);

```

Like Handler Example

```

const handleLike = async (postId) => {

await fetch(`http://localhost:5000/like/${postId}`, { method: "POST" });

// Update local state...

};

```

# Report and output screenshots

Include step-by-step UI screenshots:

- News carousel showing top 5 headlines

- Category filter in action

- Like/View counter updating

- Post detail view increasing view count

- API response (Postman or console log)

# Limitation of projects

- No user authentication or user-specific interactions.

- Likes/views are not protected from multiple updates by same user.

- Hosted only on local development server.

- Not mobile-optimized for all screen sizes.

# Summary and conclusion

One Point News serves as a centralized platform to consume news from different categories efficiently. It demonstrates full-stack development skills using MERN stack with integration of REST APIs and database operations. Though currently limited in features such as user authentication, it successfully delivers the core functionalities of news aggregation and interaction.

Future improvements:

- User login/signup

- Commenting feature

- Admin panel for managing categories

- Deployment on cloud platforms like Vercel or Heroku