CA PREREQUISITES DOCUMENT

Name of Student	**Shraeyaa Dhaigude**
Class Roll No	**D15A_15**
D.O.P.	
D.O.S.	
Sign and Grade	

TITLE: RESOURCIFY

The project is a Topic Management System, a full-stack web application designed to help users manage topics, subtopics, and associated resources with user authentication and file upload capabilities.

INTRODUCTION:

The system allows users to register and log in securely using JWT tokens.
Users can create, read, update, and delete topics and subtopics.
Users can upload and manage resources linked to subtopics, including file uploads handled via Cloudinary.
The application supports search functionality for topics and subtopics.
The frontend features a modern, responsive UI built with React and styled using Tailwind CSS.

SYSTEM REQUIREMENTS

Software Requirements:

- Backend:
    - Python 3.x
    - Flask 2.3.3 web framework
    - Flask extensions: flask-cors, flask-pymongo, flask-bcrypt, flask-jwt-extended
    - MongoDB database (local or cloud)
    - Cloudinary account for file uploads
    - python-dotenv for environment variable management
- Frontend:
    - Node.js environment

    - React 19.0.0 for UI components
    - Vite as the build tool and development server
    - Tailwind CSS for styling
    - Axios for HTTP requests
    - React Router DOM for client-side routing
    - ESLint and related plugins for code quality

Hardware Requirements:

- A development machine capable of running Python and Node.js environments
- MongoDB server (local or cloud-hosted)
- Internet connectivity for Cloudinary file uploads and package installations

TECHNOLOGY STACK:

Backend Technology:

- Flask REST API backend with JWT-based authentication
- MongoDB as the NoSQL database
- Cloudinary for media file storage and management
- Python libraries for security (bcrypt), CORS handling, and environment configuration

Frontend Technology:

- React 19 with functional components and hooks
- Vite as the build and development tool
- Tailwind CSS for utility-first styling
- React Router DOM for navigation between pages
- Axios for communicating with the backend API

Setup Instructions:

Backend:

1. Clone the repository and navigate to the backend directory:
git clone <repository-url>
cd backend

2. Create a .env file in the backend directory with the following environment variables:
CLOUD_NAME=<your-cloudinary-cloud-name>

API_KEY=<your-cloudinary-api-key>

API_SECRET=<your-cloudinary-api-secret>
MONGO_URI=<your-mongodb-uri>
JWT_SECRET_KEY=<your-jwt-secret-key>

3. Install the required Python packages:
pip install -r requirements.txt

4.Start the backend server:
python app.py

Frontend:

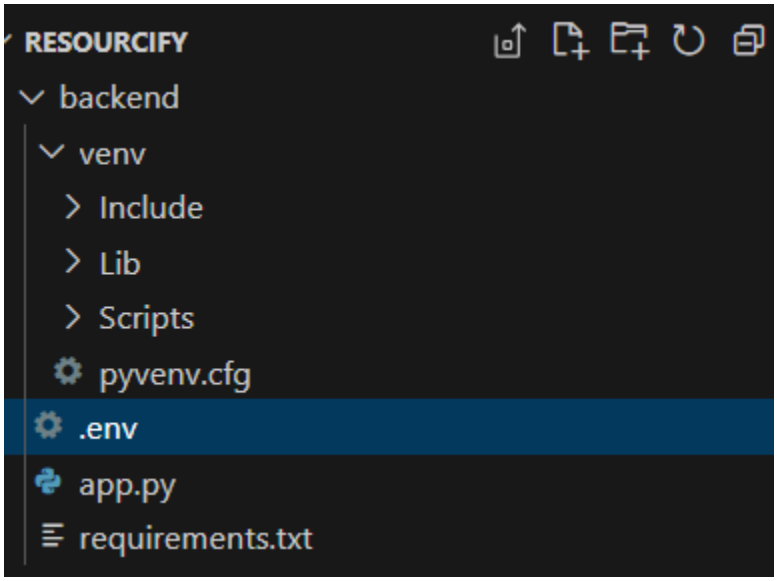
1. Navigate to the frontend directory:
cd frontend

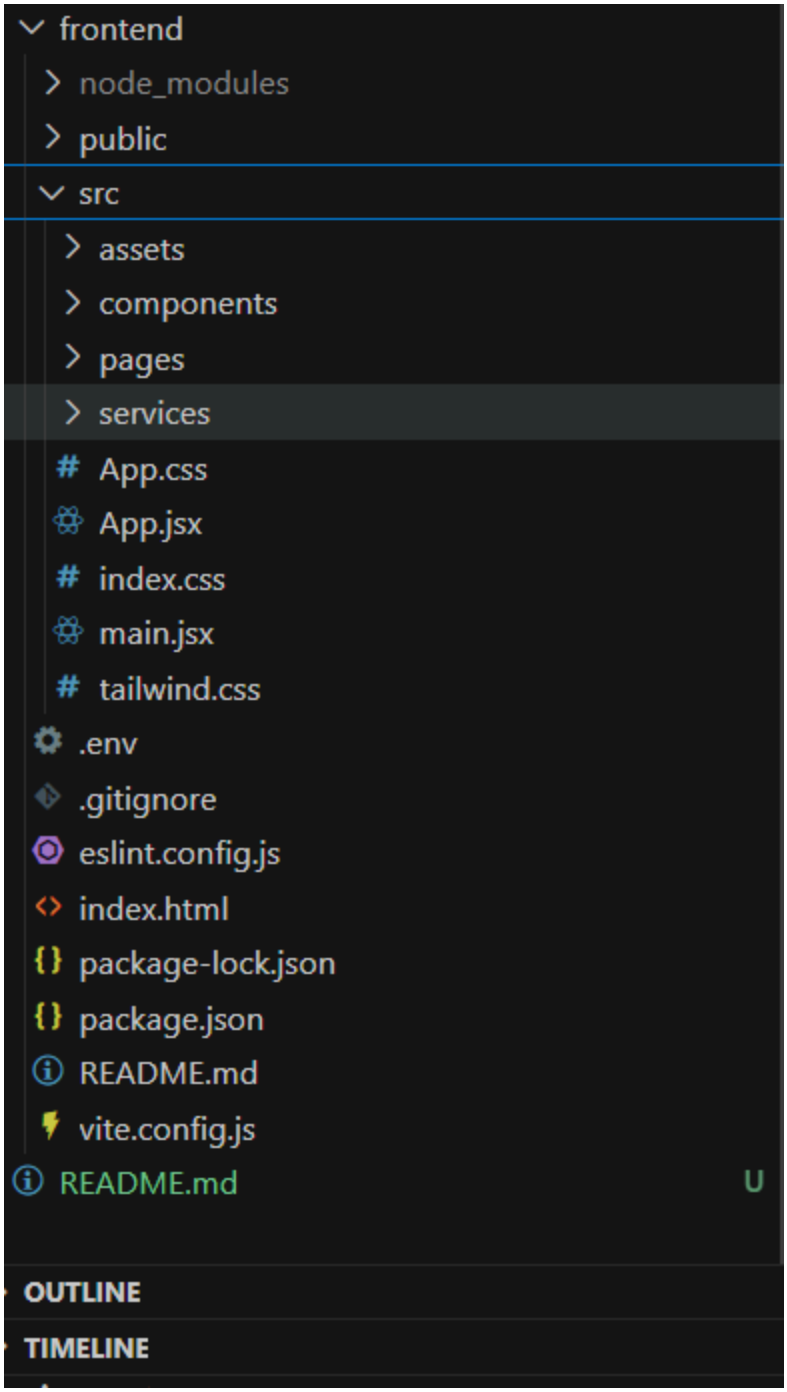
2. Install the required Node.js packages:
npm install

3. Start the frontend development server:
npm run dev

4. Access the application in your browser at http://localhost:3000.

PROJECT STRUCTURE:





CONCLUSION:

This document provides necessary prerequisites for resourcify including all the requirements and setup instructions