WEBX CA REPORT

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

Resourcify: Resource Management System

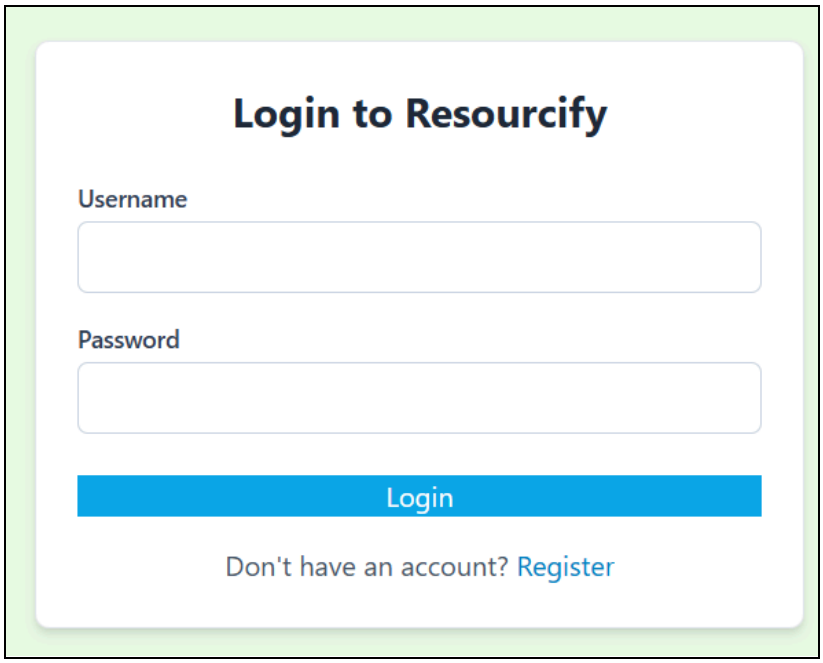
Description
The Resource Management System is a web application designed to help users manage topics, subtopics, and associated resources. It provides a user-friendly interface for creating, updating, and deleting topics and subtopics, as well as uploading and managing resources. The application features user authentication, allowing users to securely log in and manage their content.

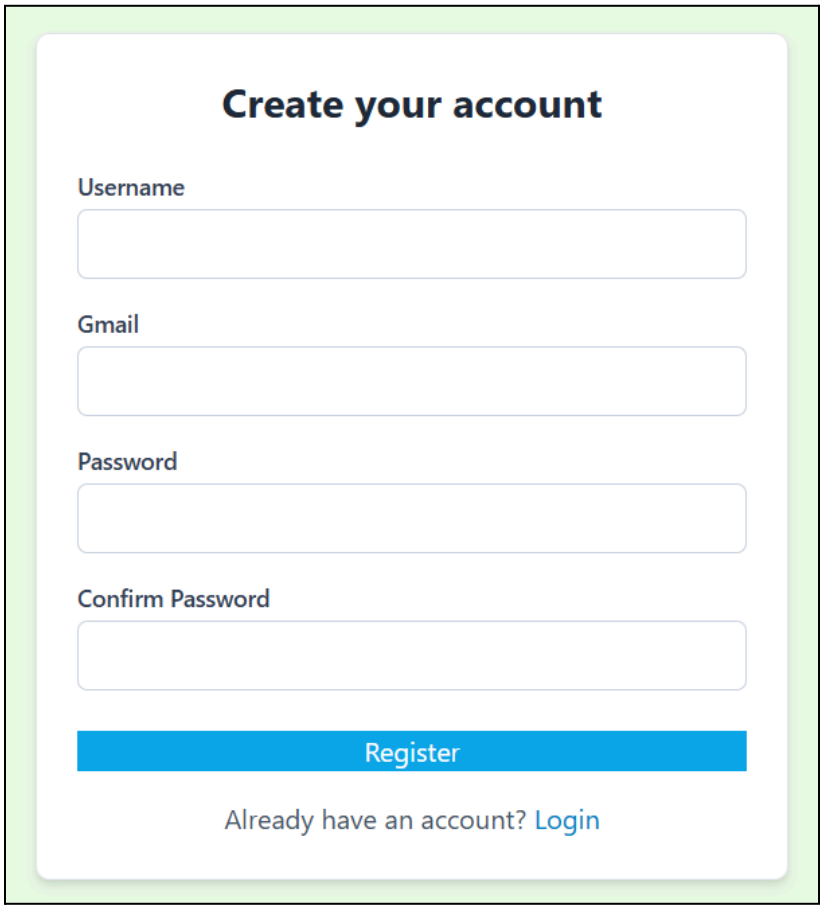
Key Features

- **User Authentication**: Secure login & registration with JWT.
- **Topic & Subtopic Management**: Full CRUD operations.
- **Resource Handling**: Upload and manage files under subtopics.
- **Search**: Locate topics/subtopics by name.
- **Cloudinary Integration**: For resource file uploads.
- **Responsive UI**: Built with React and Tailwind CSS.

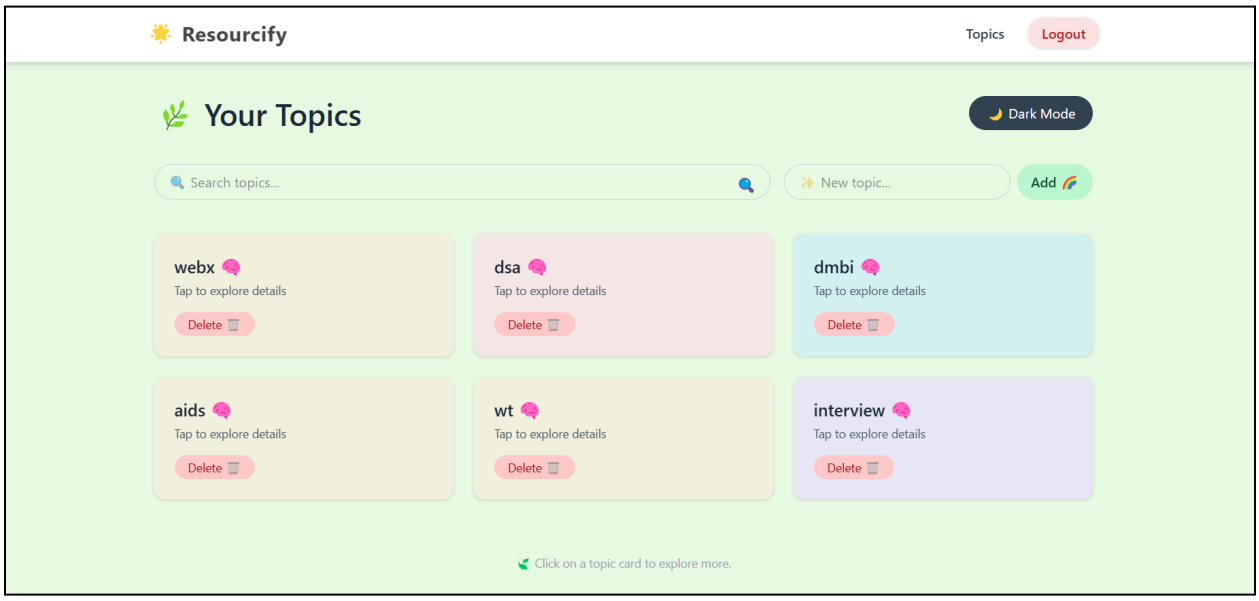
Tech Stack
Backend: Flask, Flask-CORS, Flask-PyMongo, Flask-Bcrypt, Flask-JWT-Extended, MongoDB, Cloudinary, dotenv
Frontend: React, Vite, Tailwind CSS, Axios

Output:

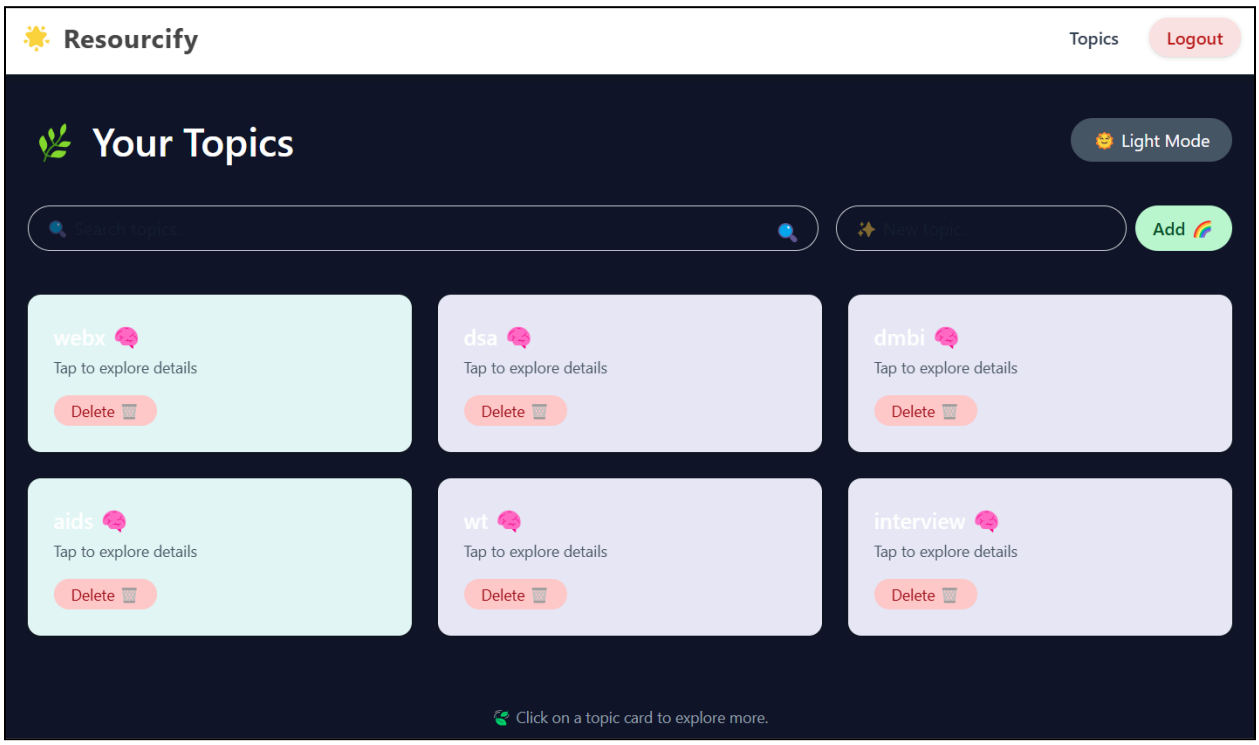

Login to Resourcify
Username
Password
Login
Don't have an account? Register


Create your account
Username
Gmail
Password
Confirm Password
Register
Already have an account? Login

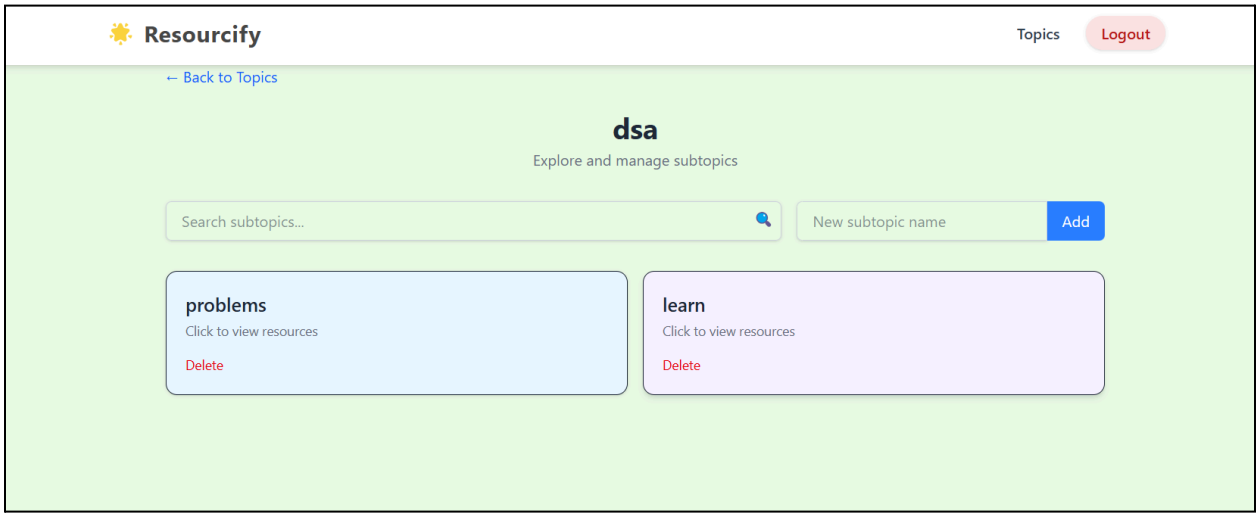
This is Home page. Users can add topics, search for topic, delete topic here. Dark mode is also available.



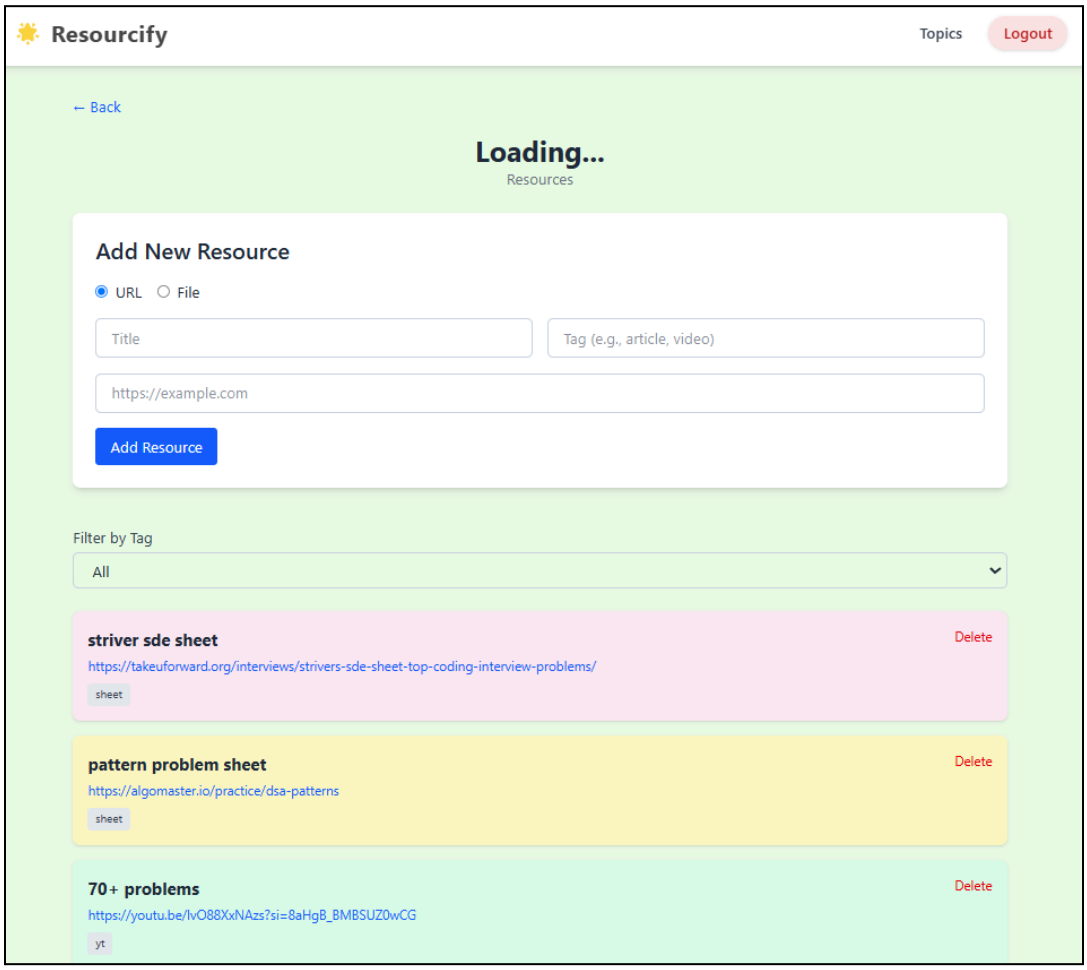
Dark mode:



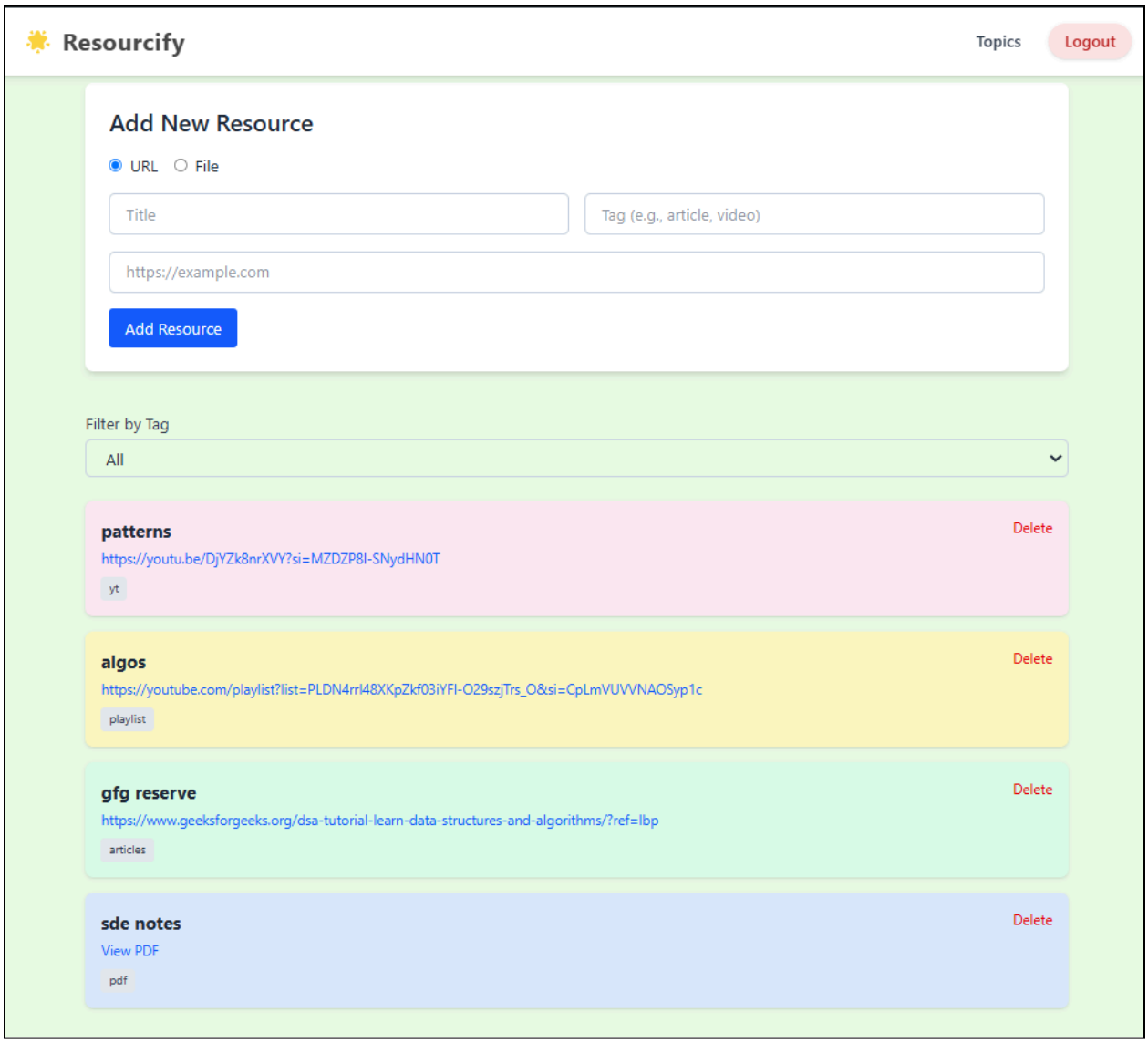
This is the Topic page where the topic's subtopics are shown. You can add search and delete the subtopic as per your wish.



This is the Resources page where user can add resources



User can add resources in either url format or even files of choice. It will need a title and tag. Tags would be used for filtering for ease of finding resources



Conclusion:

The project is a Topic Management System, a full-stack web application designed to help users efficiently manage topics, subtopics, and associated resources. It features secure user authentication with JWT, CRUD operations for topics and subtopics, resource management with file uploads to Cloudinary, and search functionality. The frontend is built with React and Tailwind CSS for a modern, responsive user interface, while the backend uses Flask with MongoDB for data storage and API services. This system provides a user-friendly platform for organizing and accessing educational or informational content in a structured manner