

Web Design - Open Elective K-Explore

Resourcify [Home](#) [Post](#) [Resources](#)

[Sign In](#)

[Sign Up](#)

Welcome to Resourcify

Discover curated resources that will help you grow your skills. Explore tutorials, articles, and guides created to enhance your learning experience.

[Explore Resources](#)

Resourcify

Find Your resources Here

[Try Pitch](#)

KIIT University, School of Computer Engineering

Group 11

Team Members

Course Coordinator - Dr. Soumya Ranjan Mishra

<u>Name</u>	<u>Roll No</u>	<u>Role/Contribution</u>	<u>Github</u>
1. Rudra Sankha Sinhamahapatra	22051880	Full Stack Developer	github.com/Rudra-Sankha-Sinhamahapatra
2. Sayan Das	22051885	Backend Developer	https://github.com/sayandas0007
3. Syed Ateeb Ul Hasan	22051645	Frontend Developer	https://github.com/hasanateeb
4. Nikhil Kumar	22051698	Researcher Developer	
5. Rishi Raj Verma	22051601	PPT making	
6. Pratyush Sharma	22051708	PPT making	
7. Nihar Ranjan Sahu	22051696	PPT making	
8. Harshit	22051682	Report making	
9. Aryan	22051757	Report making	
10. Ayush Ranjan	22051848	Report making	

Introduction

Resourcify is a web application built to simplify the way users explore and request resources, while providing administrators with tools to manage and moderate activity. The project leverages modern web technologies to ensure performance and scalability.

Problem Statement

Challenges: This slide identifies the main problems users face with existing platforms for resource management. These could be:

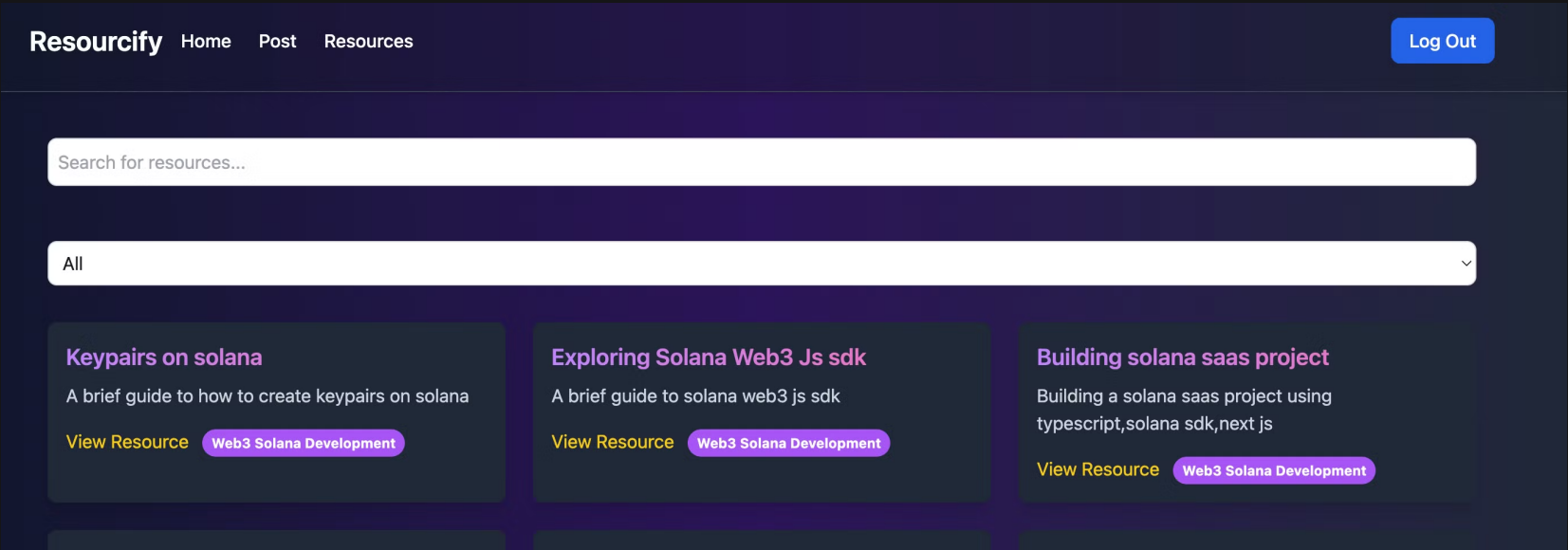
- Complicated user interfaces
- Lack of scalability, meaning systems can't handle growth effectively

Solution: Resourcify addresses these issues by providing:

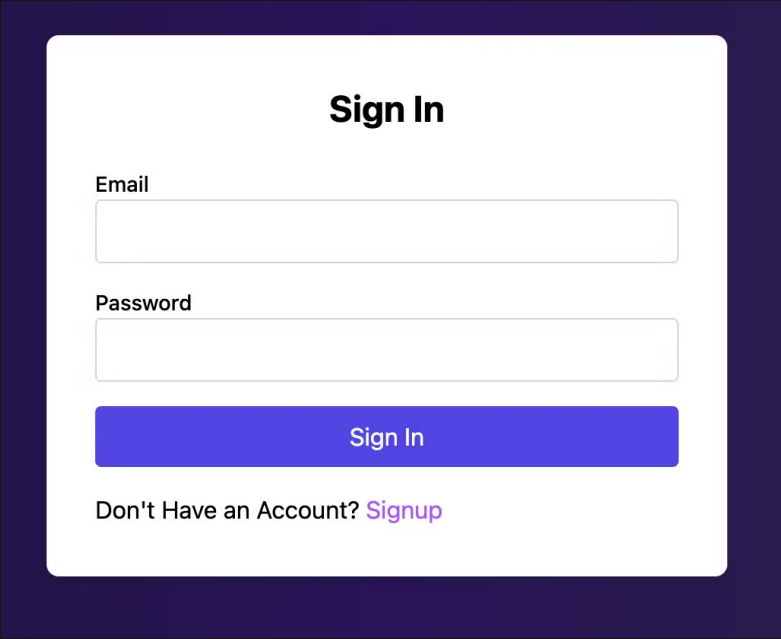
- **Intuitive design:** Easy-to-use for both tech-savvy and non-tech-savvy users
- **Scalable architecture:** Able to handle more users and resources without performance issues

Project Objectives

- Easy-to-use UI



- Secure login and signup



Role-based feature accessibility

- **User Based Signin**
- **Admin Control**

Scalable backend with modern technology stack:

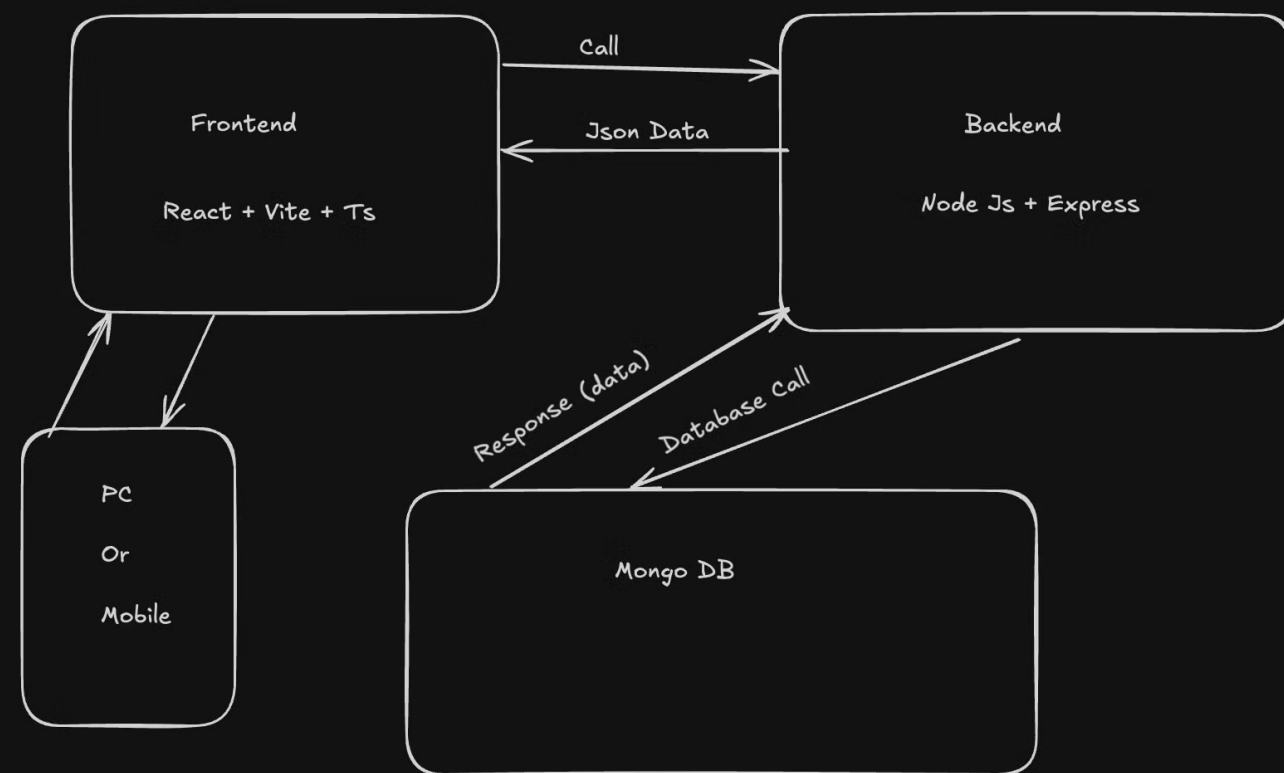
- **Typescript**
 - A modified and refined version of javascript
- **React Js**
 - Modern frontend framework for efficient performance

Technology Stack

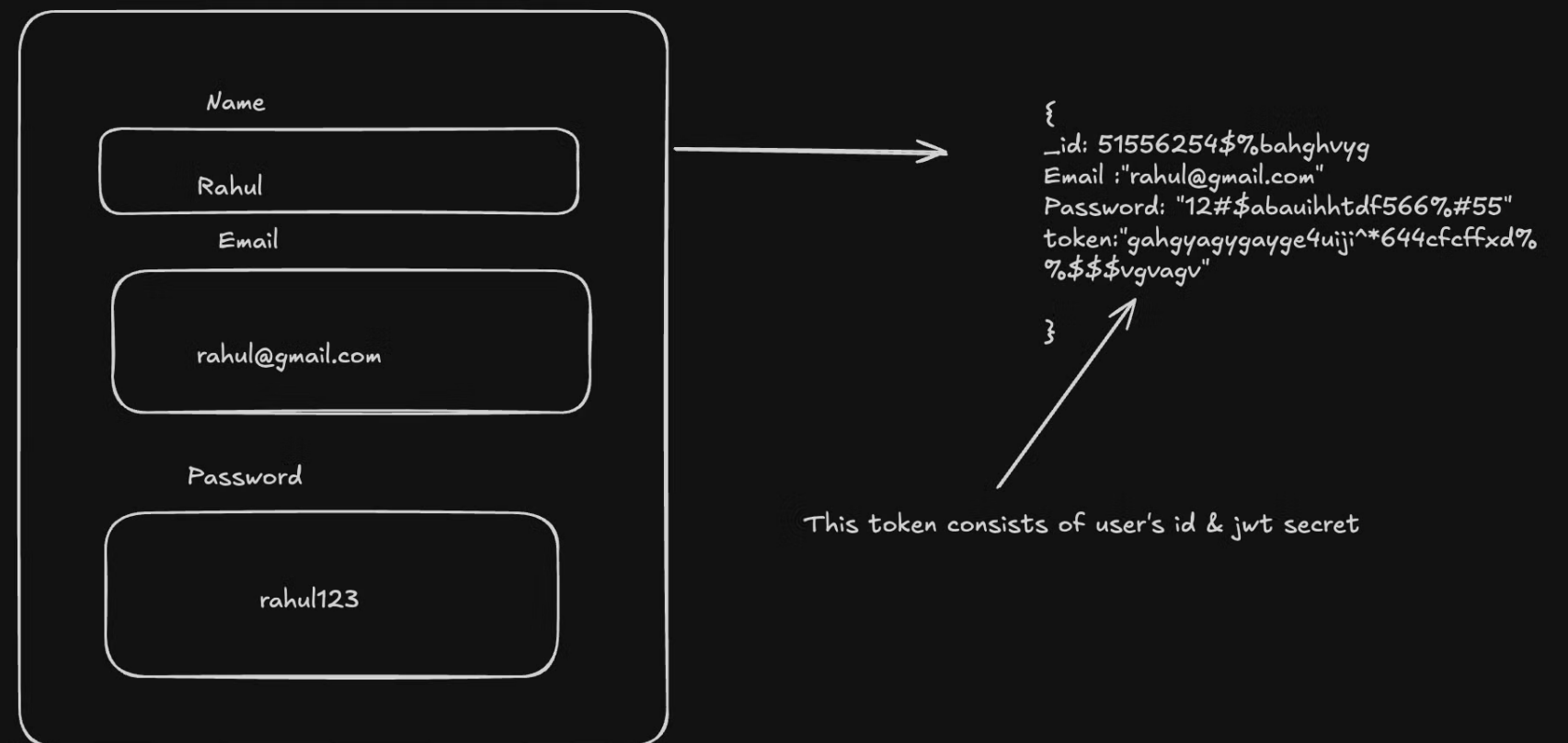
- **Frontend (React, Vite, Tailwind CSS):** These technologies were chosen for their speed and ease of use. React ensures a dynamic user interface, Vite helps with fast builds, and Tailwind provides modern styling.
- **Backend (Node.js, Express.js):** Node.js and Express.js were selected because they are lightweight and allow the creation of a scalable and maintainable API.
- **Language (TypeScript):** TypeScript was selected over JavaScript for its enhanced safety and code maintainability leading to fewer run-time errors.
- **Database (MongoDB):** MongoDB's document-based database structure is great for handling resource requests and managing users.
- **Deployment (Vercel & Render):** Vercel is used for deploying the frontend, and Render is used for the backend. This choice ensures that the app is easy to maintain and scale.
- **CI/CD:** CI Integration is taken care by Github Actions & CD is taken care by vercel itself
- **Containerization & Local Development:** For Containerization and Local Development we are using Docker.

System Architecture

- **Frontend:** Built with React, the user interface allows users and admins to interact with the system.
- **Backend:** The Express.js server handles API requests, managing tasks like user authentication and resource management.
- **Database:** MongoDB stores user data, resource information, and request history.
- **Authentication:** JWT tokens are used for session management, ensuring that only authenticated users can access certain features.



Basic Project Architecture



Basic Signup Architecture With Jwt

Key Features

- **Secure Authentication:** Using JWT, users' sessions are protected, and login/signup features work smoothly.
- **Role-Based Access:** Users and admins have different permissions; for example, admins can approve or reject resource requests.
- **Scalable Backend:** The backend API is designed to handle increasing numbers of users and resource requests.
- **Resource Management:** Users can request resources, and admins can manage these requests, creating a dynamic interaction between users and admins.

Implementation

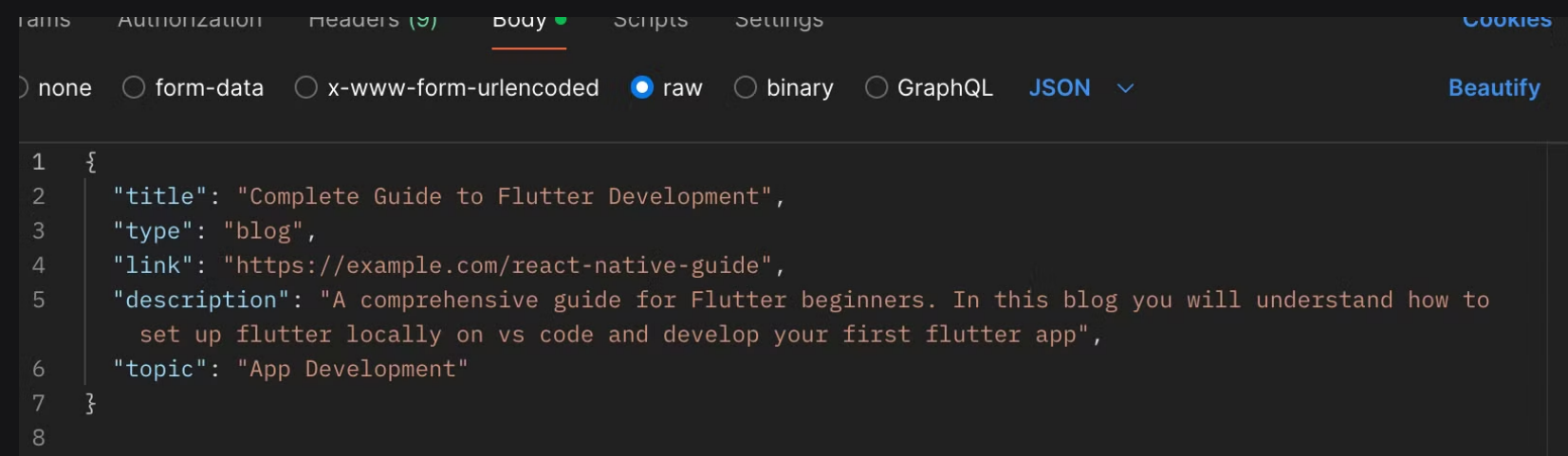
- **Frontend:** React and Vite were used to build the frontend, providing a fast and responsive user interface. Tailwind CSS ensures the application is mobile-friendly and visually appealing.
- **Backend:** The Node.js and Express.js backend provides routes for various operations like user signup, login, resource management, etc.
- **Database:** MongoDB, with Mongoose, is used to store and manage data related to users and resources.
- **Authentication:** JWT ensures that user sessions are secure, and only authorized users can access certain features.

Challenges Faced

- **Backend Deployment Issues:** Initially, the backend deployment on Vercel encountered issues with environment variables and MongoDB connections. Switching to Render solved these problems.
- **CORS Issues:** Handling cross-origin resource sharing (CORS) was tricky as requests between the frontend (Vercel) and backend (Render) required proper configuration.
- **Database Optimization:** Efficient handling of MongoDB queries was necessary to ensure fast responses, especially as the number of users grew.

Results

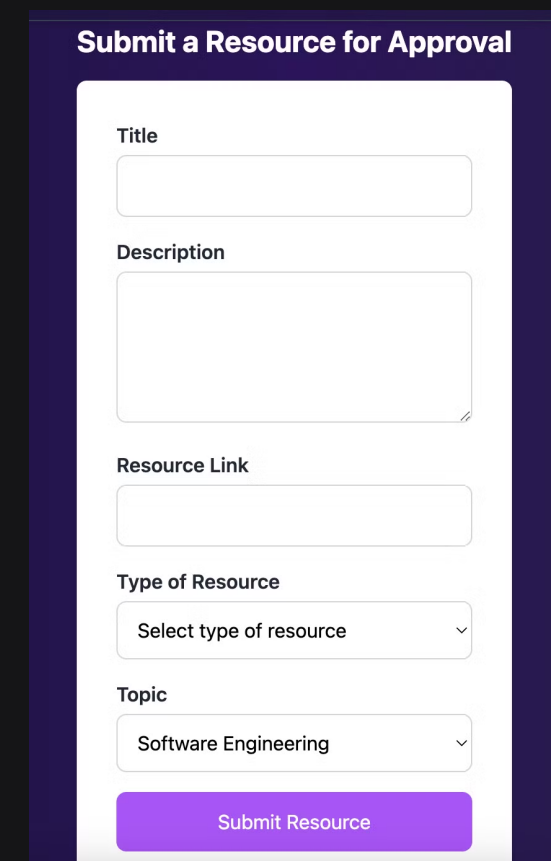
- **Successful Deployment:** The application is now live and running smoothly on Vercel (frontend) and Render (backend).
- **Authentication Working:** Secure login and signup are fully functional with JWT.
- **Resource Management Feature:** Admins can manage resources, while users can request them, demonstrating a functional role-based access system.



A screenshot of a REST client interface. The 'Body' tab is selected, and the data is in JSON format. The JSON object contains the following fields: 'title' (Complete Guide to Flutter Development), 'type' (blog), 'link' (https://example.com/react-native-guide), 'description' (A comprehensive guide for Flutter beginners. In this blog you will understand how to set up flutter locally on vs code and develop your first flutter app), and 'topic' (App Development).

```
1 {  
2   "title": "Complete Guide to Flutter Development",  
3   "type": "blog",  
4   "link": "https://example.com/react-native-guide",  
5   "description": "A comprehensive guide for Flutter beginners. In this blog you will understand how to  
6     set up flutter locally on vs code and develop your first flutter app",  
7   "topic": "App Development"  
8 }
```

Admin Posting Resources



A screenshot of a web form titled 'Submit a Resource for Approval'. The form has a dark blue header and a light blue body. It contains the following fields: 'Title' (text input), 'Description' (text area), 'Resource Link' (text input), 'Type of Resource' (dropdown menu with 'Select type of resource' selected), and 'Topic' (dropdown menu with 'Software Engineering' selected). A green 'Submit Resource' button is at the bottom.

Users requesting resources

Future Scope

- **Real-Time Notifications:** Implementing *WebSockets* to provide users and admins with instant notifications about resource status (e.g., when a request is approved).
- **Advanced Admin Features:** Developing more admin tools, such as:
 - **User Management:** Ability to ban users, or approve/reject multiple requests at once.
 - **Resource Analytics:** Admins could get statistics on resource requests (most requested resources, popular times, etc.).
- **Mobile-First Design:** Further refining the responsiveness of the app, focusing on mobile users to enhance the experience for users accessing the app via smartphones.
- **Resource Tagging and Search:** Adding a feature where resources can be tagged and searched by category, making it easier for users to find the resources they need.
- **Scaling Infrastructure:** As user base increases, considering advanced cloud infrastructure like AWS Lambda or Google Cloud Functions for serverless operations, improving the scalability even more.

Conclusion

Resourcify solves the problem of managing resources in a simple, scalable way.

- The project demonstrates expertise in integrating modern web technologies and handling deployment challenges effectively.
- Future improvements will focus on enhancing the admin experience, real-time functionality, and mobile responsiveness.

References

Github Repository: <https://github.com/Rudra-Sankha-Sinhamahapatra/Resourcify>

Live Link: <https://resourcify-kexplore.vercel.app>

Thank You