Project Report

VerifyMe-SaaS style Landing Page and SignUp Workflow

Introduction

Every modern web app begins with a simple action signing up. But what comes next - verifying users and building trust - is essential for any serious platform. **VerifyMe** is a mini-project that recreates this experience. It allows users to enter their name and email, sends a verification link to their inbox, and then redirects them to a thank-you page once verified. I also created a simple logo to give it a personal touch and a brand-like feel.

Tools Used

Frontend UI	HTML, Tailwind CSS
Backend Server	Node.js, Express.js
Email Service	Nodemailer
Database	MongoDB Atlas
IDE & Design	Visual Studio Code, Custom Logo
Version Control	Git & GitHub

Steps I Followed to Build It

1. **Landing Page Setup**
   I started by designing a clean, responsive UI using Tailwind CSS. The form accepts the user's name and email. I also designed and added a simple logo to make the landing page feel more complete.

2. **Backend Development**
   I created an Express.js backend to handle signups and email verification. When a user submits the form, a POST request is sent to /api/signup. The backend saves the data to MongoDB and sends a verification email with a unique link.

3. **Connecting Frontend & Backend**
   The integration between form and server was done using a POST request. I referred to Youtube tutorial and took help from ChatGPT to properly set up the routes, middleware, and server responses.

4. **Security and Best Practices**
   I stored all sensitive information like database URIs and email credentials in a .env file, which is ignored using .gitignore. I also created a safe env.example file to help others run the project without exposing real secrets.

Conclusion

VerifyMe may seem small, but it brings together everything a real application needs — from a working frontend to backend logic, database handling, and even real-time email verification. I learned how different parts of a full-stack app connect and how important it is to manage those connections securely.

While building this, I explored tools like Nodemailer and MongoDB Atlas, and learned to debug common integration issues. I also leaned on resources like **YouTube videos** and **ChatGPT** to fill in gaps, especially when connecting the frontend and backend smoothly.

Going forward, this project could evolve to include login functionality, password management, or even a small admin dashboard — but for now, I'm proud of how functional and real it already feels.