COLLEGE CODE: 9111

COLLEGE NAME: SRM MADURAI COLLEGE FOR ENGINEERING AND TECHNOLOGY

DEPARTMENT: BE.CSE 3RD YEAR

DATE:15/09/2025

Completed the project named as Phase 2 — Solution Design & Architecture

Name: portfolio websites

SUBMITTED BY:

B.GOKULA GAURAV

R.V.SIVAPPRASATH

S.V.KARTHIVEL

K.R.J.SHREEKANTH

K.GOWTHAM

Project Report: Portfolio Website

Tech Stack Selection
The portfolio website will be built using the following technologies:

- **Frontend:** HTML5, CSS3, JavaScript, React.js for dynamic UI.
- **Backend:** Node.js with Express.js for handling API requests.
- **Database:** MongoDB for storing user data, projects, and contact form submissions.
- **Hosting & Deployment:** Vercel/Netlify for frontend, Render/Heroku for backend.
- **Version Control:** Git and GitHub for collaboration and source code management.

UI Structure / API Schema Design

UI Structure:
1. Home Page - Introduction and personal branding.
2. About Page - Details about skills, education, and experience.
3. Projects Page - Portfolio of completed works with images and links.
4. Contact Page - Form for users to send messages.

API Schema Design:
- **GET /api/projects** → Fetch all project details.
- **POST /api/contact** → Store and forward user messages.
- **GET /api/profile** → Retrieve user profile data.
- **POST /api/auth** → Authentication for admin access.

Data Handling Approach

- Data is stored in a MongoDB database with collections: **Users**, **Projects**, and **Messages**.
- Form validation is performed on both frontend (React forms) and backend (Express middleware).
- Secure API endpoints with JWT authentication.
- Use of Mongoose ORM for schema validation and query optimization.
- Data caching with Redis for faster project loading.

Component / Module Diagram

Frontend Components:
- Navbar
- Hero Section
- About Section
- Projects Gallery
- Contact Form
- Footer

Backend Modules:
- Auth Module
- Projects Module
- Messages Module
- Profile Module

Basic Flow Diagram

1. User visits the website → React frontend renders UI.
2. API calls are made via Axios/Fetch to Node.js backend.
3. Backend queries MongoDB and returns responses.
4. Data is displayed dynamically on the frontend.
5. User submits a message through contact form → stored in MongoDB and emailed to admin.