**Personal Portfolio CMS using MEAN Stack**

**A Term Project Report**



**By**

**V.N.R. Madhughna (322103282105) Gorle Devika(322103282116)**

**Galla Jyothsna(322103282114) Tina Maheswari Kandrapu(332103282098)**

Under the esteemed guidance of

# Dr K Purushottam Naidu Associate Professor

Department of CSE(AI&ML)

**Department of Computer Science and Engineering (AI&ML)**

**GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING FOR WOMEN**

[Approved by AICTE NEW DELHI, Affiliated to Andhra university, Visakhapatnam]

[Accredited by National Board of Accreditation (**NBA**) for B.Tech. CSE, ECE & IT – Valid from

2019-22 and 2022-25]

[Accredited by National Assessment and Accreditation Council (**NAAC**) – with A Grade Valid From 2022-27]

Kommadi, Madhurawada, Visakhapatnam–530048

2024–2025

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING FOR WOMEN

Department Of Computer Science and Engineering (AI&ML)

**CERTIFICATE**



This is to certify that the term project report titled **Smart Movie Ticket Booking System using MEAN Stack** is a bonafide work of following III B. Tech II Semester students in the Department of Computer Science and Engineering(AI&ML), Gayatri Vidya Parishad College of Engineering for Women affiliated to Andhra University, visakhapatnam during the academic year 2024-2025

**V.N.R. Madhughna (322103282105) Gorle Devika(322103282116)**

**Galla Jyothsna(322103282114) Tina Maheswari Kandrapu(332103282098)**

**Dr K Purushotam Naidu**

**Associate Professor, Dept of CSE (AI&ML)**

**Term Project Coordinator**

**+-**

# ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We feel elated to extend our sincere gratitude to, **Dr K Purushottam Naidu,** Associate Professor, Department of Computer Science and Engineering (AI&ML) for encouragement all the way during analysis of the project. His annotations, insinuations and criticisms are the key behind the successful completion of the thesis and for providing us all the required facilities

We express our deep sense of gratitude and thanks to **Dr Dwiti Krishna Bebarta**, Professor and Head of the Department of Computer Science and Engineering(AI&ML) for his guidance and for expressing his valuable and grateful opinions in the project for its development and for providing lab sessions and extra hours to complete the project.

We would like to take this opportunity to express our profound sense of gratitude to Vice Principal, **Dr G Sudheer** for allowing us to utilize the college resources thereby facilitating the successful completion of our project. We are also thankful to both teaching and non-teaching faculty of the Department of Computer Science and Engineering (AI&ML) for giving valuable suggestions for our project.

We would like to take the opportunity to express our profound sense of gratitude to the revered Principal, **Dr R K Goswami** for all the help and support towards the successful completion of our project.

**TABLE OF CONTENTS**

**TOPICS PAGENO**

**Abstract i**

**Introduction 1**

**Implementation 2-32**

## Outputs 32-34

**Conclusion 35**

**ABSTARCT:**

The evolution of technology has significantly transformed the way we book movie tickets, with online platforms offering convenience and efficiency to moviegoers worldwide. In this paper, we delve into the intricacies of the Movie Ticket Booking Online System developed using the MEAN (MongoDB, Express.js, Angular, Node.js) stack, elucidating its features, functionalities, and implications for the movie industry. Our exploration encompasses a detailed examination of the user journey, from initial authentication to seamless ticket purchase, as well as the administrative tools empowering theater managers to optimize operations and enhance customer satisfaction. We delve into the intricacies of user authentication mechanisms, highlighting their role in safeguarding personal information and ensuring secure transactions. Furthermore, we analyze the interactive movie selection interface, evaluating its effectiveness in engaging users and facilitating informed decision-making through comprehensive movie details, trailers, and reviews. The seamless booking experience is a cornerstone of our system, with intuitive seat selection, dynamic pricing, and flexible payment options tailored to meet the diverse needs of modern moviegoers. Real-time updates via WebSocket technology provide users with up-to-the-minute information on seat availability and booking status, fostering transparency and reducing the likelihood of overbooking. Administrators benefit from a comprehensive dashboard powered by Express.js and Node.js, enabling them to monitor bookings, manage inventory, and analyze performance metrics to drive strategic decision-making. The responsive design ensures a consistent and optimized experience across devices, catering to the preferences of mobile and desktop users alike. Through this comprehensive review and analysis, we aim to shed light on the transformative potential of MEAN stack-based solutions in revolutionizing the movie ticket booking experience, fostering efficiency, convenience, and customer satisfaction in the digital era.

**Keywords:** MEAN Stack, Movie Ticket Booking, Online System, User Authentication, WebSocket Technology, Admin Dashboard, Responsive Design, Customer Satisfaction, Strategic Decision-Making

**INTRODUCTION:**

The current online environment, one's own portfolio is required to showcase skills, works, and achievements—most especially for developers and creative professionals. To facilitate easier management and maintenance of portfolio materials, I developed a Personal Portfolio Content Management System (CMS) using the MEAN stack (MongoDB, Express.js, Angular, Node.js).

The project allows users to dynamically create, edit, and update their own portfolio content via an easy-to-use interface. The MEAN stack was utilized due to its scalability, performance, and support for both front-end and back-end with JavaScript. Angular provides an interactive and responsive interface, whereas Node.js and Express.js are suitable for server-side logic. MongoDB is suitable for storing portfolio work, blog entries, and user information with scalable and flexible data storage.

All the main features of the CMS, such as secure login, page customization (e.g., About, Projects, Skills, Contact), inserting media, and live content editor are included. The CMS is responsivized completely so the portfolio is stunning on anything. Admins can just deal with content and don't have to touch any code, so it is perfect for non-technical users as well as developers.

This project demonstrates how the MEAN stack can be employed to create dynamic, sustainable, and personalized digital portfolios. It is a balance of technical expertise and practical usability, offering seamless and smooth content management for individuals who want to have a strong web presence.

One of the best aspects of this project is its focus on user autonomy and flexibility. Unlike static portfolio websites where one has to make manual modifications to the codebase, this CMS allows users to modify their portfolio content effortlessly and straightforwardly via a plain dashboard. From the addition of new projects, modification of skillsets, to the composition of blog posts, all these can be modified on the fly at the click of a button. With the role-based access incorporated, the system also ensures that only the appropriate person is allowed to make changes, providing another level of security and professionalism. This turns the portfolio into a living, breathing example of one's development and journey in the tech industry and not a static display.

**IMPLEMENTATION:**

**Create server.js:**

server.js

cd portfolio-cms/server

touch server.js

cd portfolio-cms/server

npm init -y

npm install express mongoose cors

node server.js

MongoDB connected

Server running on port 3000

**Server.js:**

const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

const app = express();

// Middleware

app.use(cors());

app.use(express.json());

// Connect MongoDB

mongoose.connect('mongodb://127.0.0.1:27017/portfolio', {

useNewUrlParser: true,

useUnifiedTopology: true,

})

.then(() => console.log('MongoDB connected'))

.catch((err) => console.log(err));

// Test route

app.get('/', (req, res) => {

res.send('Portfolio CMS Backend Running');

});

// TODO: Add project routes here later

// app.use('/api/projects', require('./routes/projectRoutes'));

const PORT = process.env.PORT || 3000;

app.listen(PORT, () => {

console.log(`Server running on port ${PORT}`);

});

**Project.js:**

const mongoose = require('mongoose');

const projectSchema = new mongoose.Schema({

title: String,

description: String,

link: String,

image: String

});

module.exports = mongoose.model('Project', projectSchema);

**ProjectRoutes.js:**

const express = require('express');

const router = express.Router();

const Project = require('../models/Project');

// CRUD

router.get('/', async (req, res) => {

const projects = await Project.find();

res.json(projects);

});

router.post('/', async (req, res) => {

const newProj = new Project(req.body);

await newProj.save();

res.json(newProj);

});

router.put('/:id', async (req, res) => {

const updated = await Project.findByIdAndUpdate(req.params.id, req.body, { new: true });

res.json(updated);

});

router.delete('/:id', async (req, res) => {

await Project.findByIdAndDelete(req.params.id);

res.sendStatus(204);

});

module.exports = router;

**Frontend Setup (Angular):**

**Init Angular:**

cd ../client

ng new portfolio-client

cd portfolio-client

ng generate component projects

ng generate service project

**project.service.ts**

import { HttpClient } from '@angular/common/http';

import { Injectable } from '@angular/core';

@Injectable({ providedIn: 'root' })

export class ProjectService {

private api = 'http://localhost:3000/api/projects';

constructor(private http: HttpClient) {

}

getProjects() { return this.http.get(this.api); }

addProject(data: any) { return this.http.post(this.api, data); }

updateProject(id: string, data: any) { return this.http.put(`${this.api}/${id}`, data); }

deleteProject(id: string) { return this.http.delete(`${this.api}/${id}`); }

}

**component.ts:**

import { Component, OnInit } from '@angular/core';

import { ProjectService } from '../project.service';

@Component({

selector: 'app-projects',

templateUrl: './projects.component.html'

})

export class ProjectsComponent implements OnInit {

projects: any = [];

constructor(private service: ProjectService) {}

ngOnInit() {

this.service.getProjects().subscribe(data => this.projects = data);

}

}

**component.html**

<div \*ngFor="let project of projects" class="card">

<h3>{{ project.title }}</h3>

<p>{{ project.description }}</p>

<a [href]="project.link" target="\_blank">View</a>

<img [src]="project.image" width="200" />

</div>

**Start MongoDB:**

Mongod

**Start Express:**

cd server

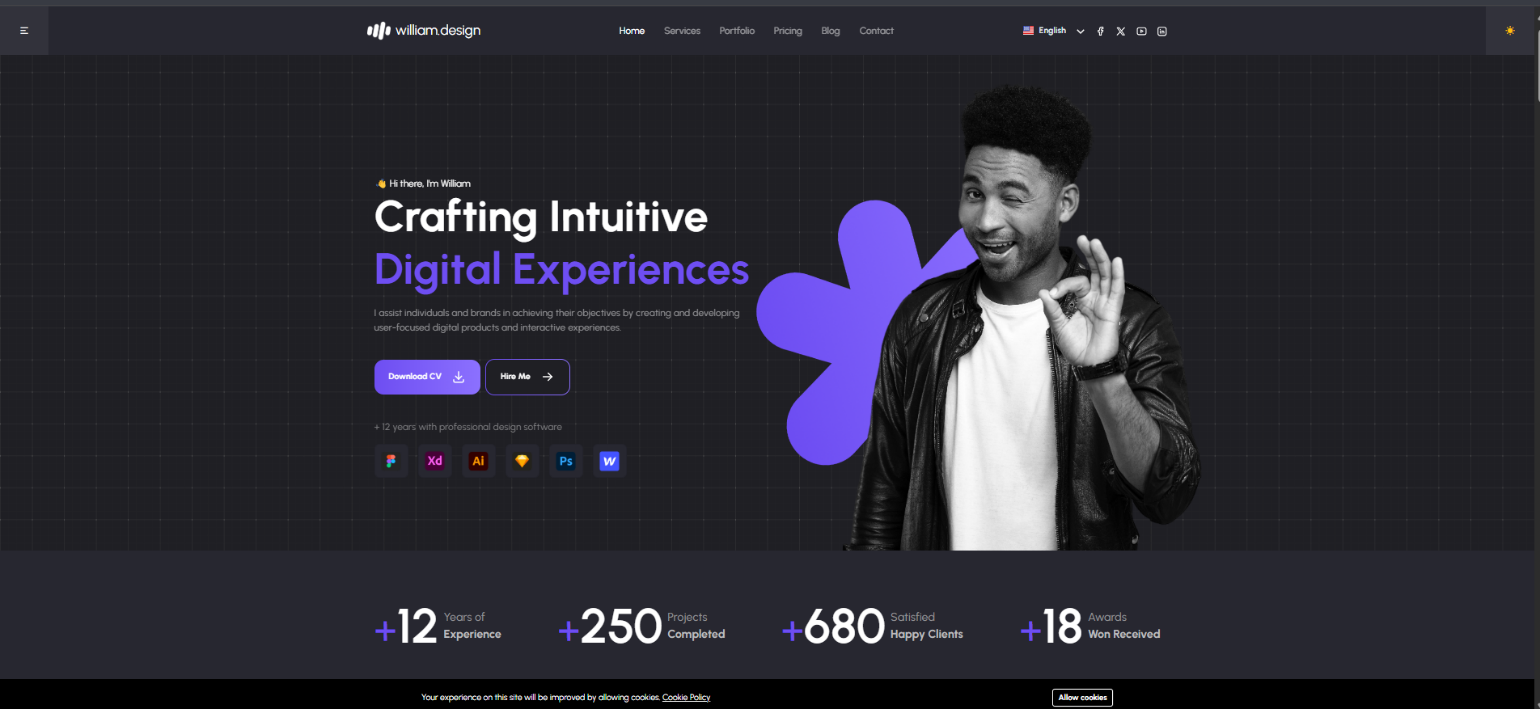
node server.js

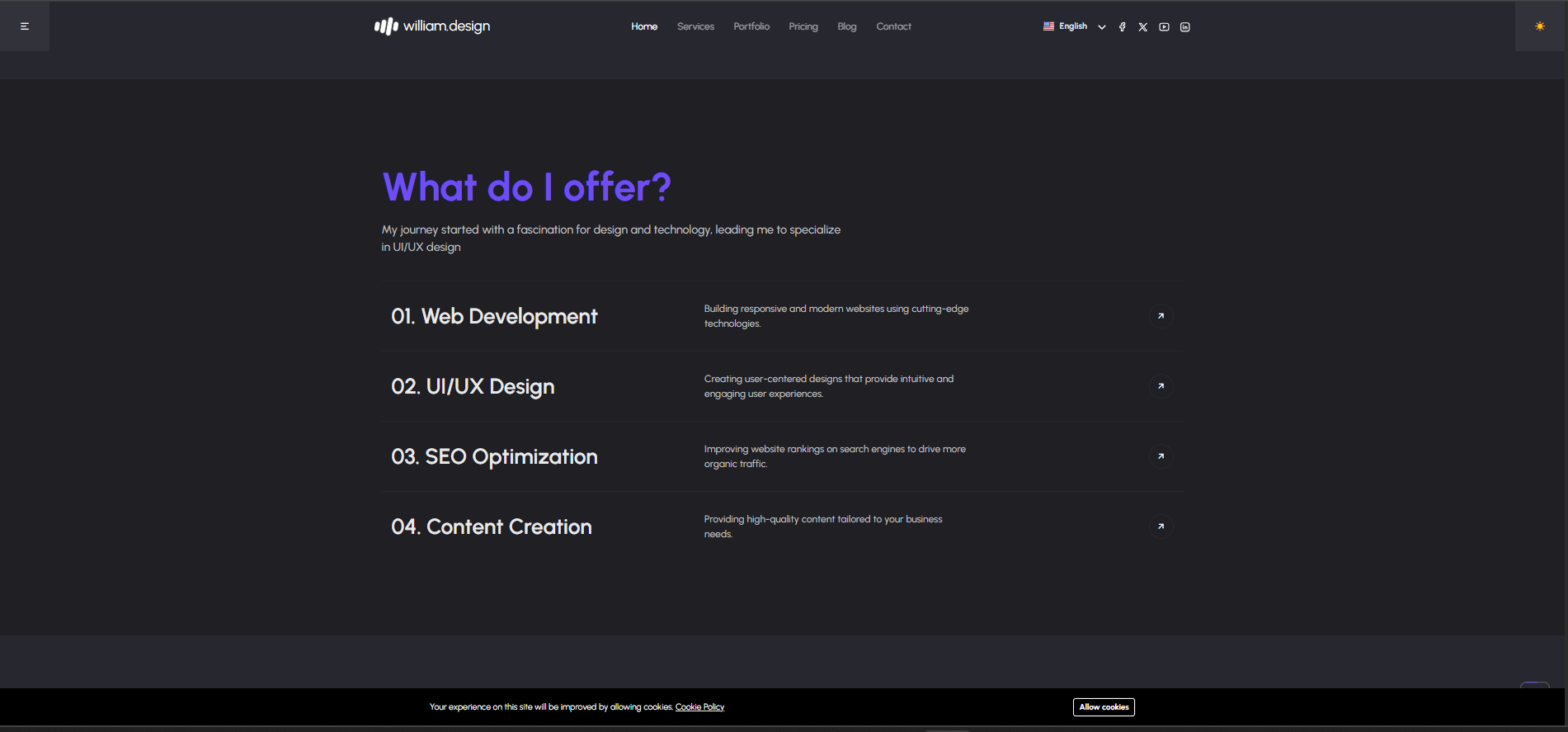
**Start Angular:**

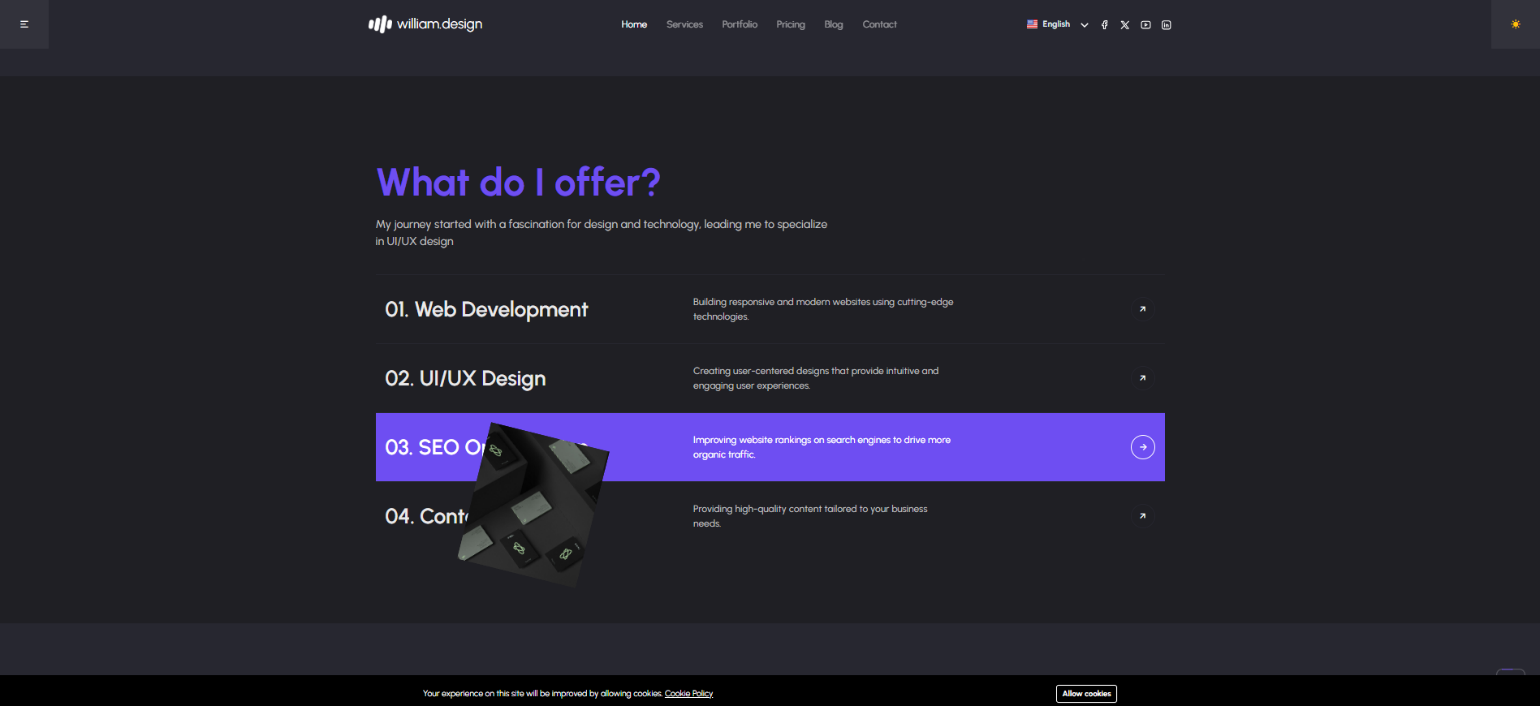
cd client/portfolio-client

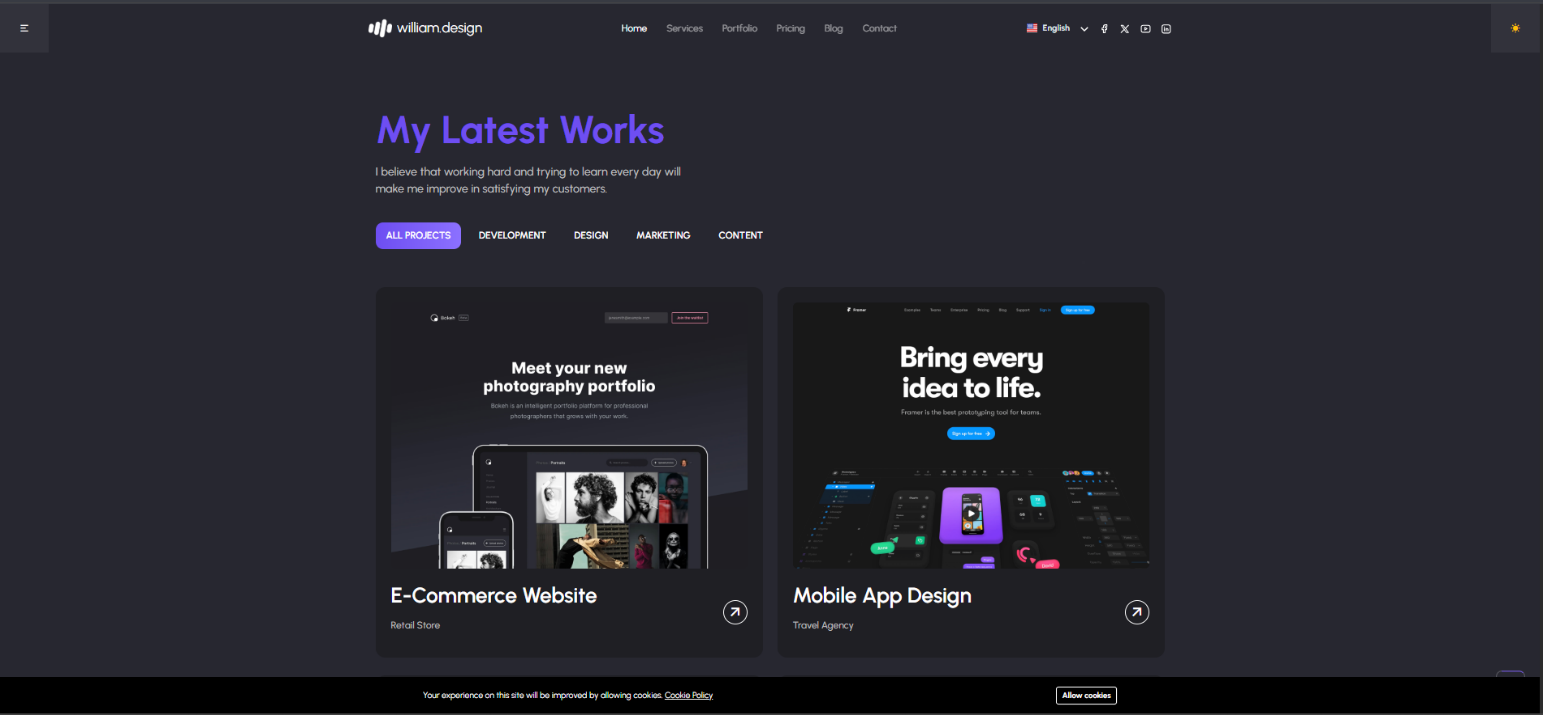
ng serve

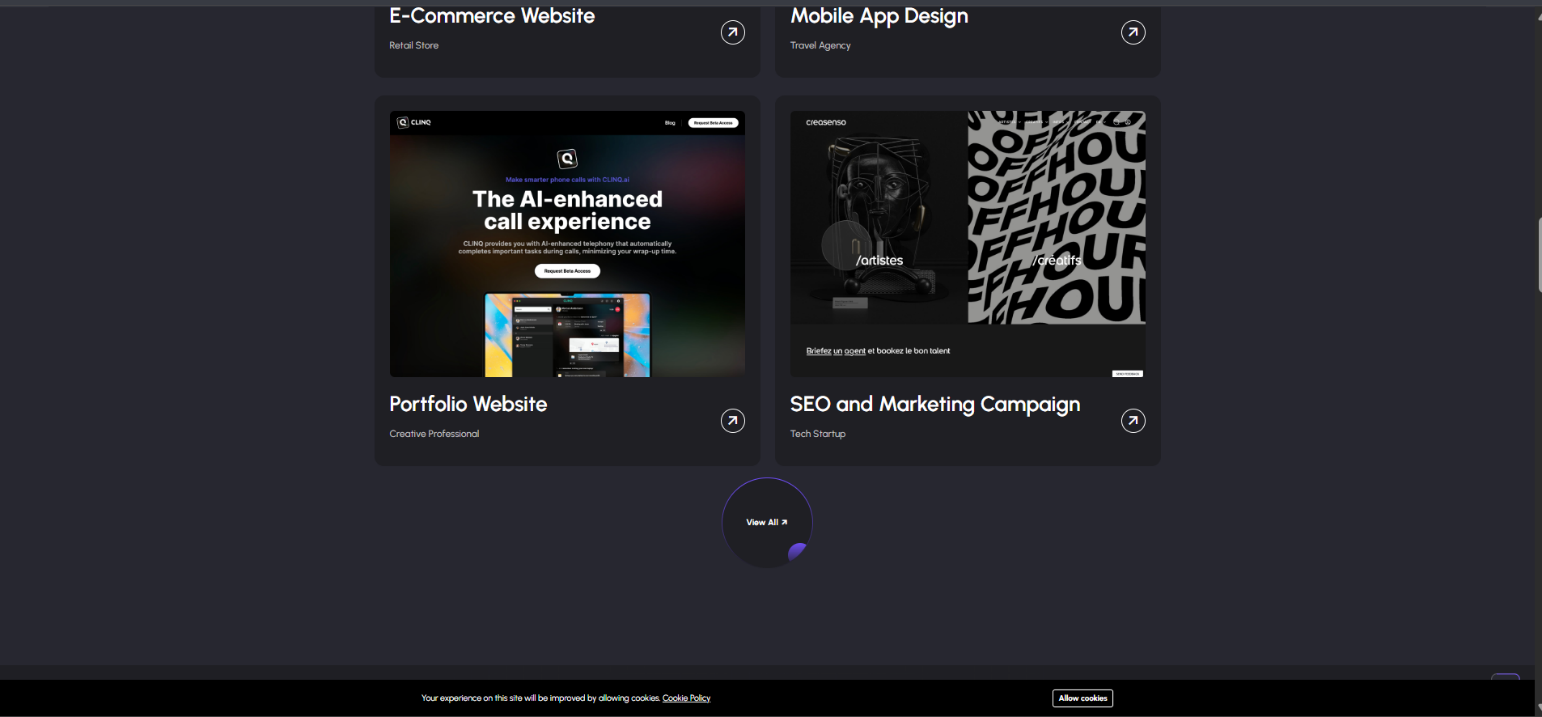
**OUTPUTS:**

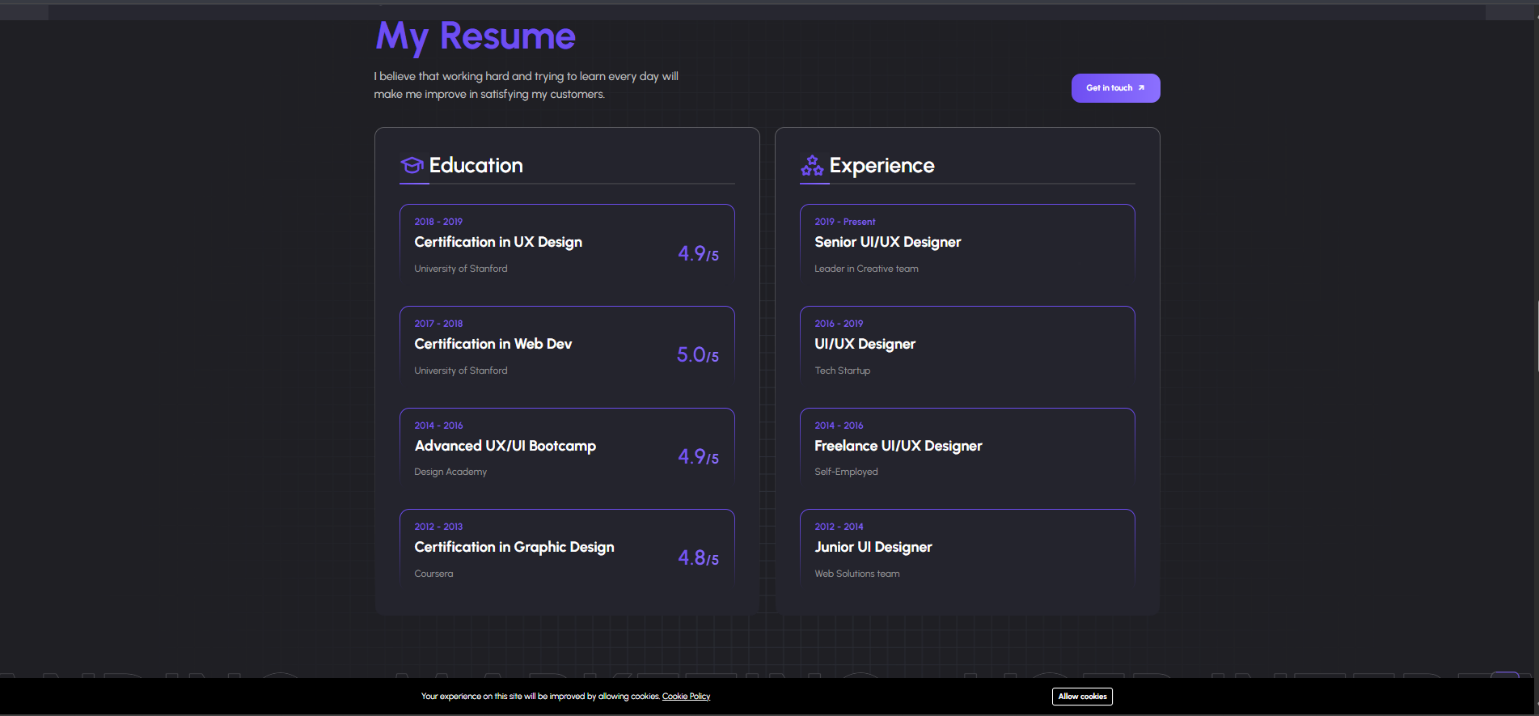
****

****

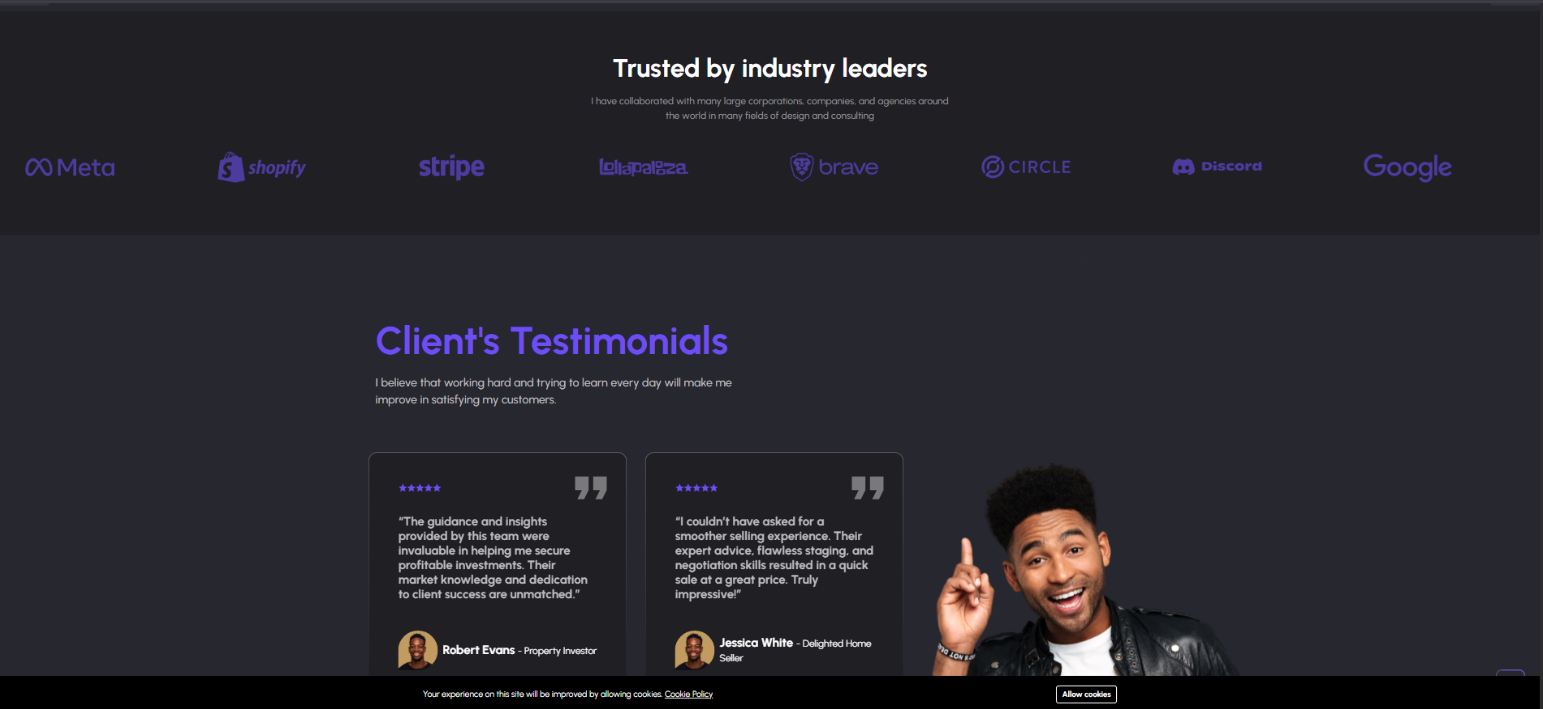


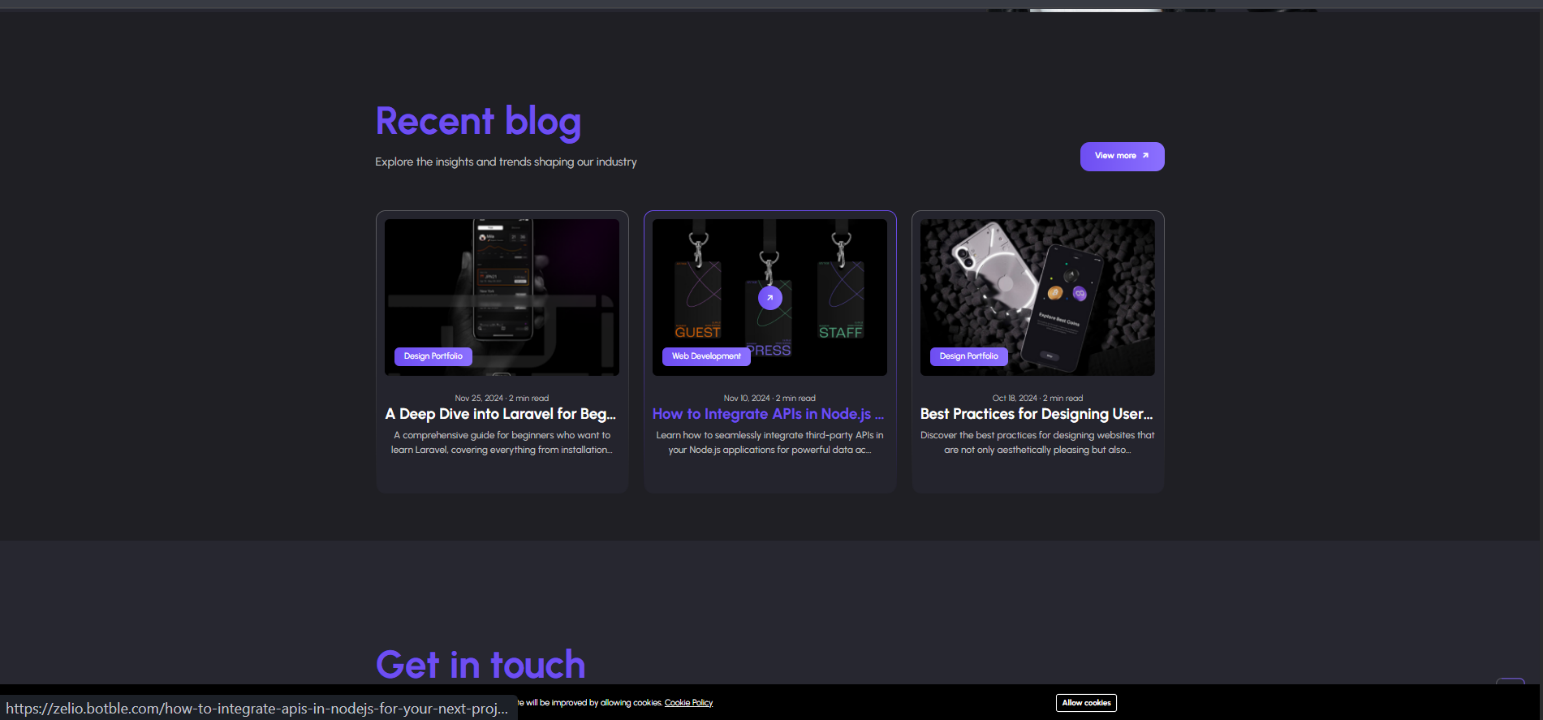


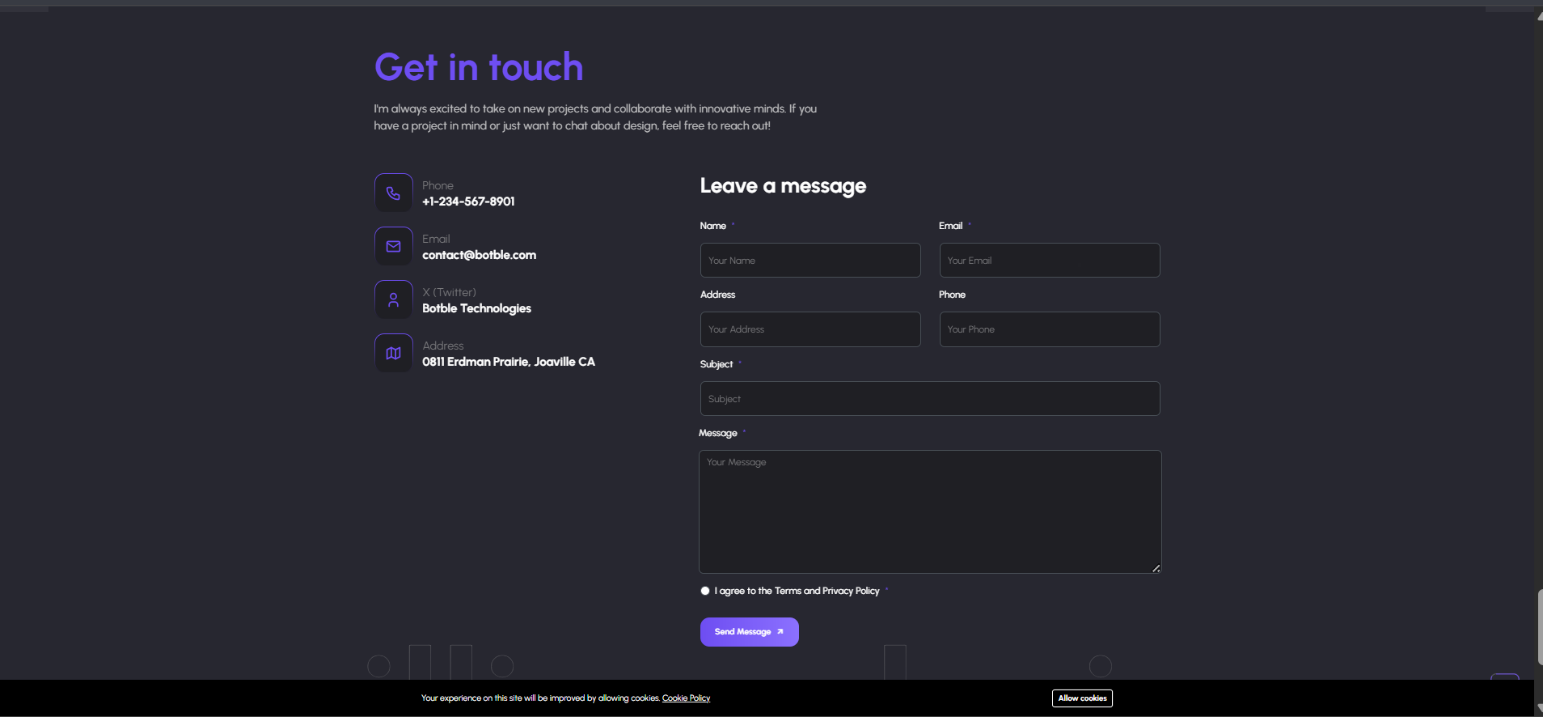












**CONCLUSION:**

In conclusion, developing my personal portfolio using the MEAN stack has been a transformative journey that not only strengthened my technical skills but also allowed me to create a robust, scalable, and dynamic web application. The integration of MongoDB, Express.js, Angular, and Node.js enabled a seamless development process, where each component played a vital role in delivering an efficient and responsive user experience.

The backend, powered by Node.js and Express.js, provided a lightweight and fast environment for handling server-side operations, API creation, and data management. Using MongoDB as the database allowed for flexible and schema-less storage, which helped in organizing and retrieving user data efficiently. These technologies together ensured a powerful and reliable server infrastructure.

On the frontend, Angular brought the portfolio to life with its component-based architecture and two-way data binding, enabling interactive and user-friendly design. Angular’s modular structure made it easy to maintain and scale different sections of the portfolio, such as projects, experience, and contact forms, ensuring a smooth and intuitive navigation experience for users.

Throughout this project, I learned how to effectively integrate all layers of a full-stack application, manage data flow between frontend and backend, and deploy the application to a production environment. This hands-on experience has deepened my understanding of full-stack development and boosted my confidence in working with real-world projects using modern web technologies.

Overall, building this MEAN stack portfolio was not just about showcasing my work—it was about demonstrating my technical capabilities, passion for development, and readiness to take on real-world challenges. This portfolio stands as a reflection of my growth as a full-stack developer and serves as a gateway for future opportunities and collaborations in the tech industry.