INDUSTRIAL INTERNSHIP REPORT

On

FULL STACK WEB DEVELOPMENT

Prepared by
Divya

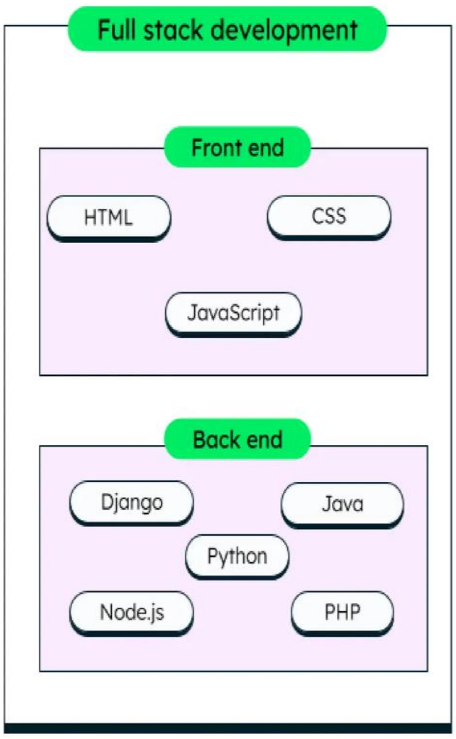
B.Tech - Computer Science Engineering

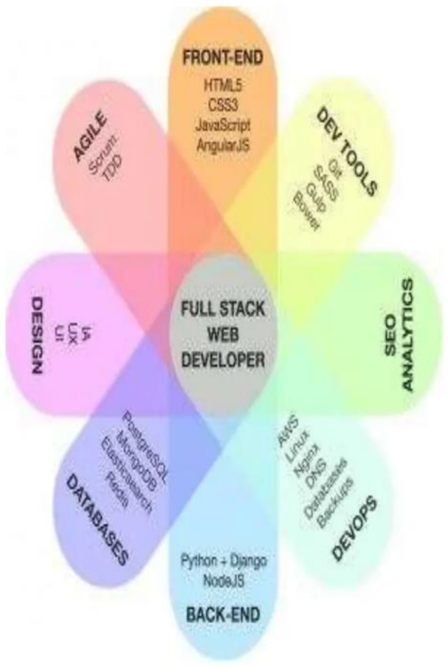
Under the guidance of
Upskill Campus & UniConverge Technologies Pvt Ltd

TABLE OF CONTENTS **PAGE NUMBERS**

1. Preface

This report summarizes the six weeks Industrial Internship completed under Upskill Campus
in collaboration with UniConverge Technologies Pvt Ltd. The internship focused on Full Stack Web Development.
The objective was to gain real industry exposure and develop a complete web application including frontend, backend,
and database integration. This internship helped in understanding real-world software development lifecycle
including requirement analysis, design, development, testing and deployment.





4

2. Introduction

Full Stack Development refers to development of both frontend and backend portions of a web application. A full stack developer works with client-side technologies, server-side programming, and database management systems.

Front-end: Focuses on the "client-side" using HTML, CSS, and JavaScript to build the visual elements users interact with.

Back-end: Handles the "server-side" logic, often using languages like Python, Java, or Node.js to manage data and application functionality.

Databases: Systems like SQL, Oracle, or MongoDB used to store and retrieve application data.

What is Full Stack Web Development?
Learn everything about full-stack web development, including the front-end and back-end programming languages, frameworks, and libraries.
HTML
PHP
CSS
JAVA

2.1 About UniConverge Technologies Pvt Ltd

UniConverge Technologies Pvt Ltd (UCT) is a digital transformation company established in 2013.

It provides industrial and enterprise solutions using technologies like IoT, Cloud Computing, Machine Learning,

Cyber Security, Java Full Stack and Python. The company focuses on sustainability and ROI-driven solution.

2.2 About upskill Campus

Upskill Campus is a career development platform that provides internship programs, industry-level projects, mentorship and career support services. It aims to improve practical skills of students.

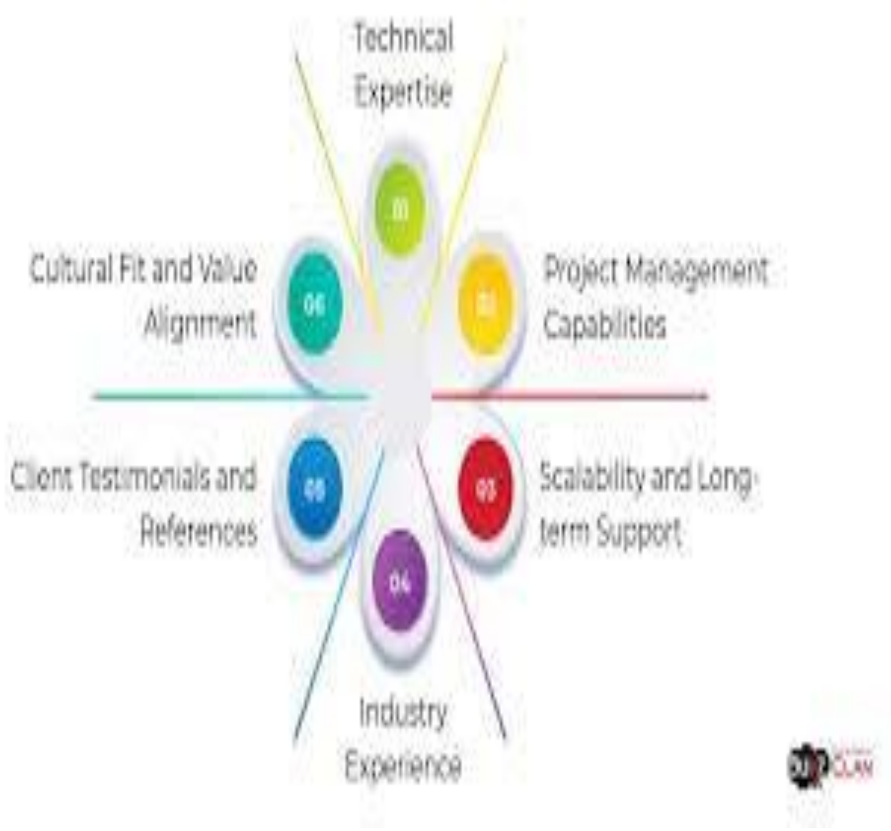
UPSKILL TODAY
Upskill in Full Stack Development: A Strategic Move for Professionals and Businesses

2.3 Objective

• To gain practical industry experience
• To develop full stack web applications
• To understand client-server architecture
• To implement authentication and security
• To improve coding and debugging skills

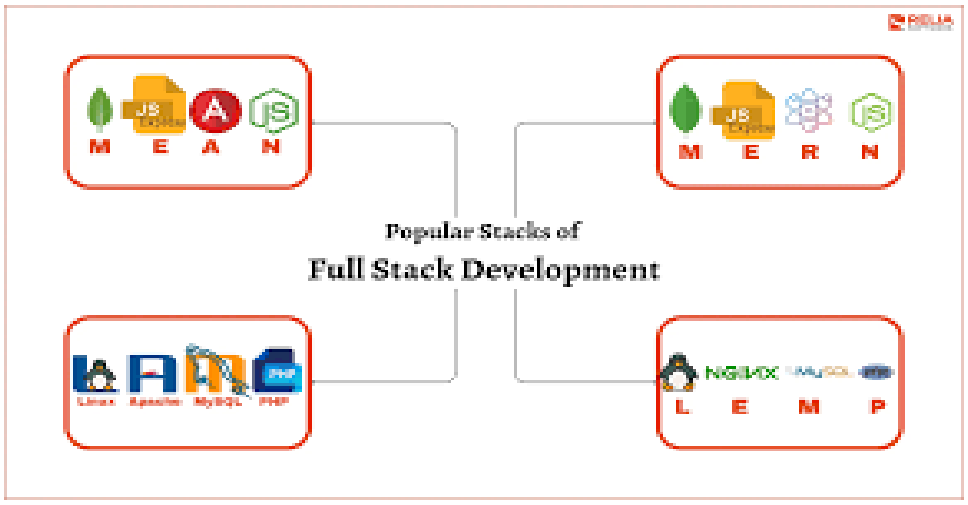
2.4 Reference

1. HTML, CSS, JavaScript Documentation
2. Node.js Official Documentation
3. MySQL Documentation


Technical Expertise
01
Project Management Capabilities
02
Scalability and Long-term Support
03
04
Industry Experience
05
Client Testimonials and References
06
Cultural Fit and Value Alignment

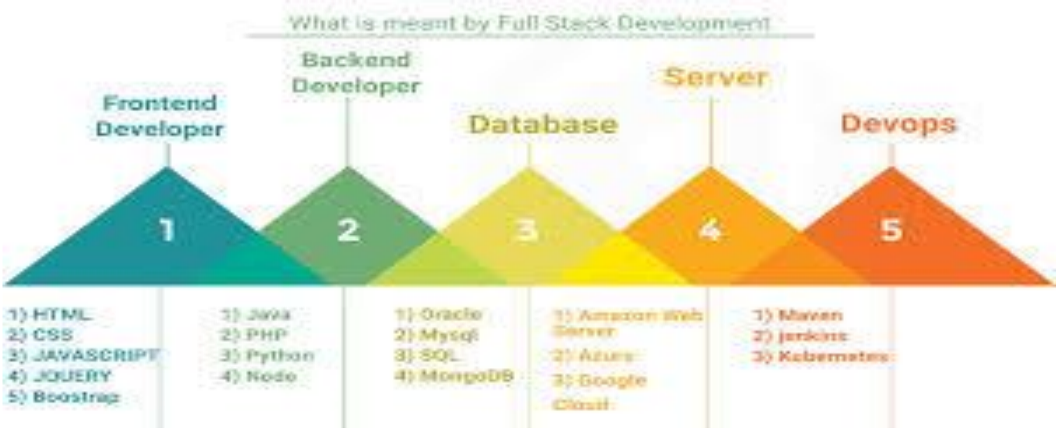
2.5 Glossary

Frontend	Client side of application
Backend	Server side logic
API	Application Programming Interface
CRUD	Create, Read, Update, Delete
DBMS	Database Management System



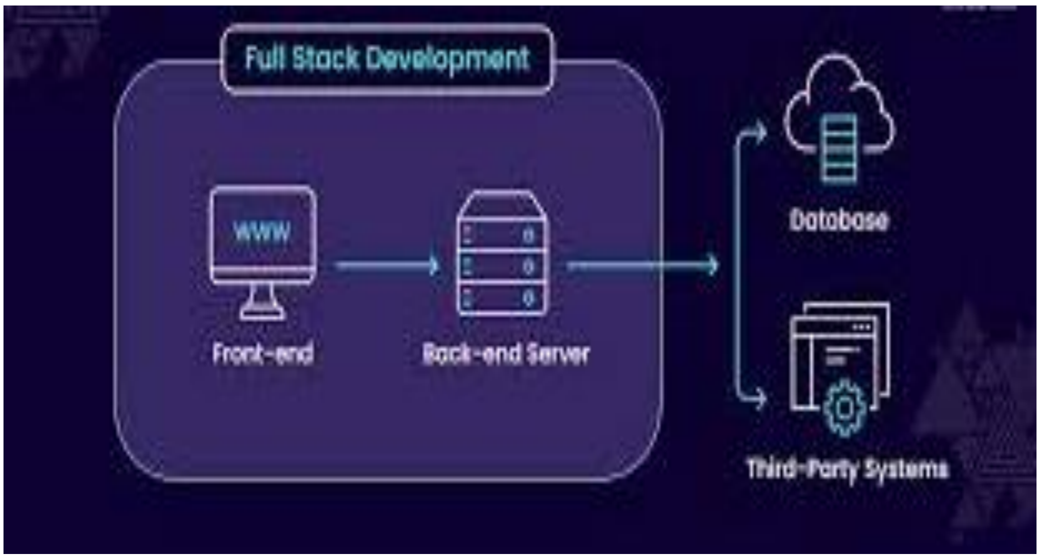
3. Problem Statement

Many small organizations require web applications to manage users and data efficiently. Traditional systems lack security and scalability. The goal was to develop a secure and responsive full stack web application with authentication and database integration.


What is meant by Full Stack Development
Frontend Developer
Backend Developer
Database
Server
Devops
1
2
3
4
5
1) HTML 2) CSS 3) JAVASCRIPT 4) JQUERY 5) Boostrap
1) Java 2) PHP 3) Python 4) Node
1) Oracle 2) Mysql 3) SQL 4) MongoDB
1) Amazon Web Server 2) Azure 3) Google Cloud
1) Maven 2) jenkins 3) Kubernetes

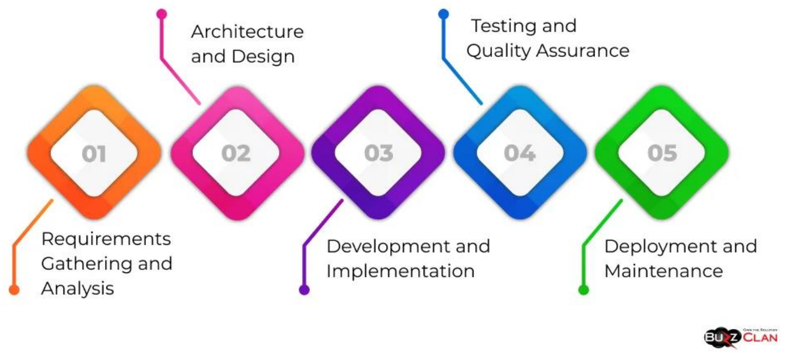
4. Existing and Proposed Solution

Existing systems were static websites or manual systems with no real-time data handling.
The proposed solution was to develop a dynamic full stack application using HTML, CSS, JavaScript for frontend,
Node.js/Java for backend and MySQL/MongoDB for database management.



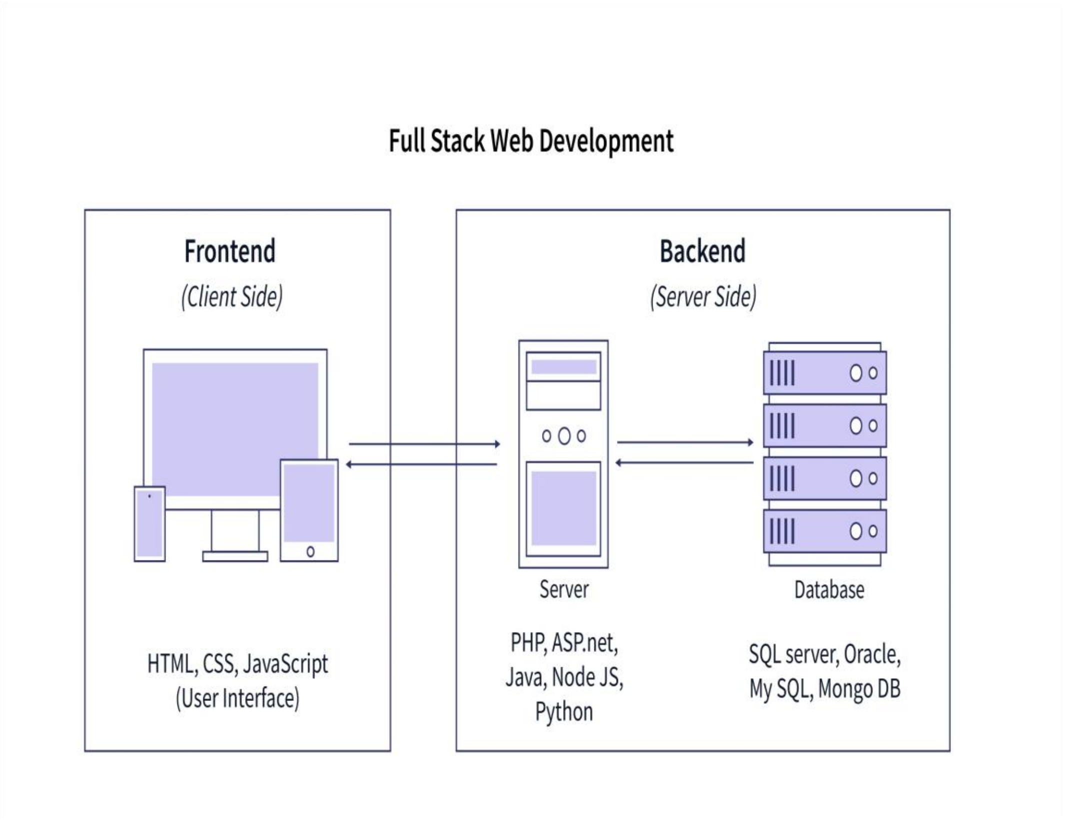
5. Proposed Design / Model

The system follows a three-tier architecture:
1. Presentation Layer (Frontend)
2. Application Layer (Backend)
3. Data Layer (Database)
The frontend interacts with backend APIs and backend communicates with database


Architecture and Design
Testing and Quality Assurance
01
02
03
04
05
Requirements Gathering and Analysis
Development and Implementation
Deployment and Maintenance
BuzzClan

5.1 High Level Diagram

User → Browser → Frontend → Backend Server → Database
This architecture ensures separation of concerns and better scalability.

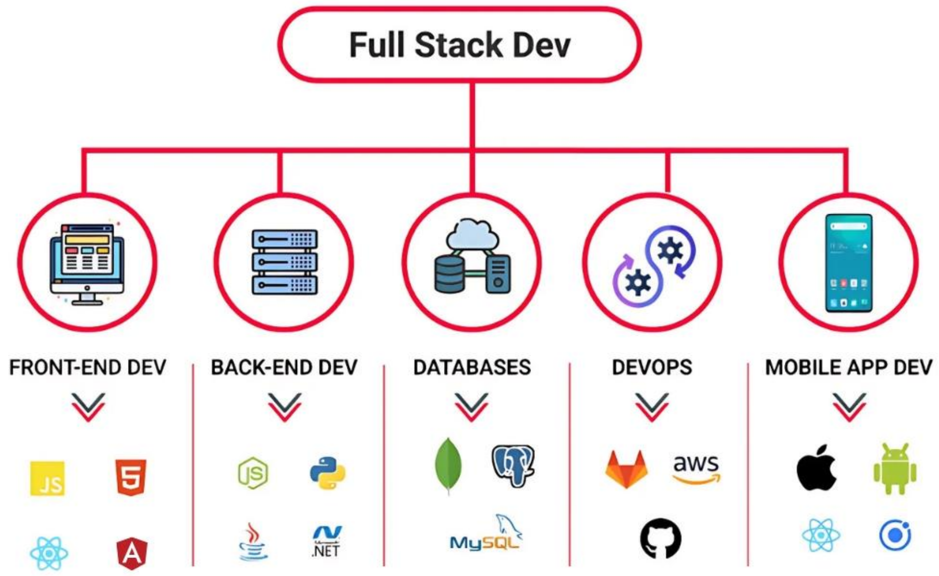

Full Stack Web Development
Frontend
(Client Side)
HTML, CSS, JavaScript
(User Interface)
Backend
(Server Side)
Server
PHP, ASP.net, Java, Node JS, Python
Database
SQL server, Oracle, My SQL, Mongo DB

5.2 Low Level Diagram

Frontend built using HTML, CSS, JavaScript.
Backend implemented using REST APIs.
Database handles structured data storage.
Authentication implemented using session/JWT.

Class Diagrams: These define the internal structure of the backend application. They detail classes, their attributes, methods, and the relationships (inheritance, association) between them in a specific module.Sequence Diagrams: These illustrate how different components interact over time to fulfill a specific request.

For example, they show the step-by-step flow from a user clicking a button on the Presentation Layer to the Business Logic Layer processing data and finally interacting with the Database Layer.



5.3 Interfaces

• User Interface (Login, Registration, Dashboard)
• Admin Interface
• API Interface
• Database Interface

THE FUTURE OF FULL STACK DEVELOPMENT:
AI, EDGE COMPUTING & BEYOND
HTML5
CSS3
PHP

6. Performance Test

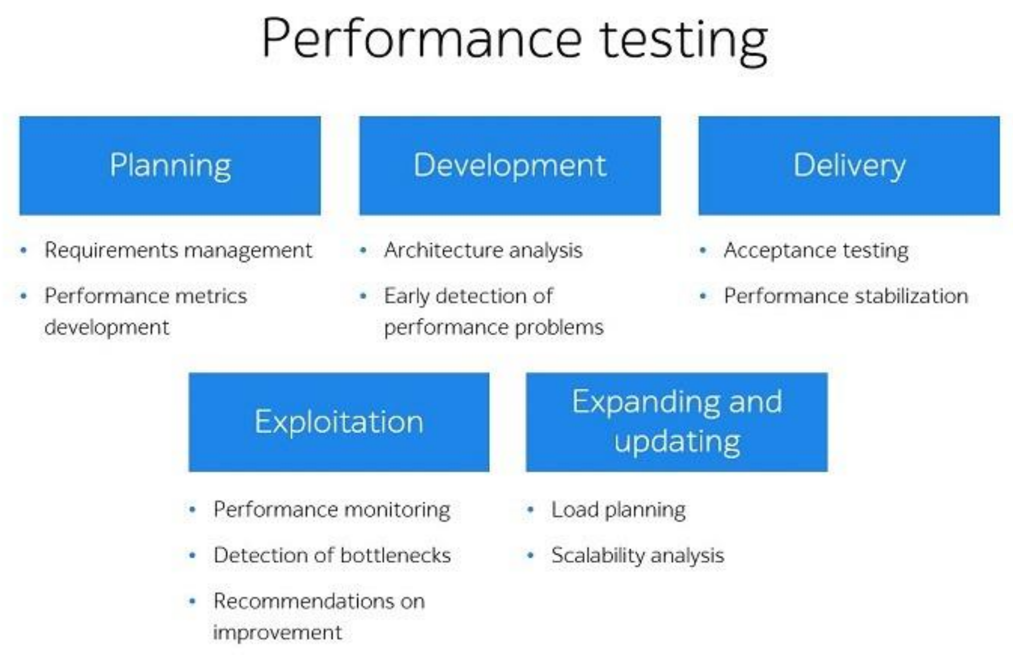
Performance testing ensures the application works efficiently under normal load conditions.
Constraints considered: response time, security, database performance and concurrent users.

front-End Performance: Focuses on Core Web Vitals, render times, and asset optimization. Tools like Lighthouse are standard for these audits.

API & Back-End Testing: Measures response times and throughput of server-side logic. Popular tools include JMeter and Postman.

Database Performance: Analyzes query execution times and indexing efficiency to prevent data bottlenecks.

Infrastructure & Scalability: Evaluates how the system handles peak loads (Load Testing) and when it fails (Stress Testing).



6.1 Test Plan / Test Cases

Tested login functionality, registration validation, CRUD operations and API responses

A Standard Test Plan acts as a blueprint for the entire testing process, typically including:

Scope: Defines the features to be tested (e.g., UI, APIs, database integration).

Test Strategy: Outlines the methodologies (e.g., Manual vs. Automated Testing) and tools to be used.

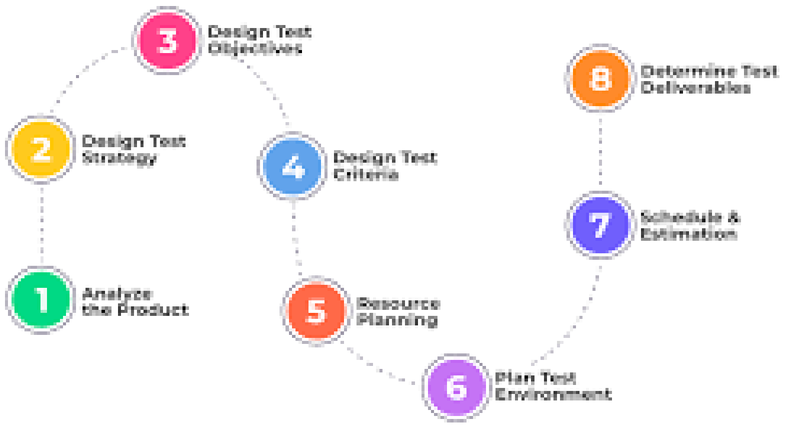
Resources: Lists the hardware, software, and team members required.

Deliverables: Specifies the documents to be provided, such as test cases and bug reports.

Common Full Stack Test Scenarios

Front-End: Verifying UI responsiveness, form validation, and cross-browser compatibility.

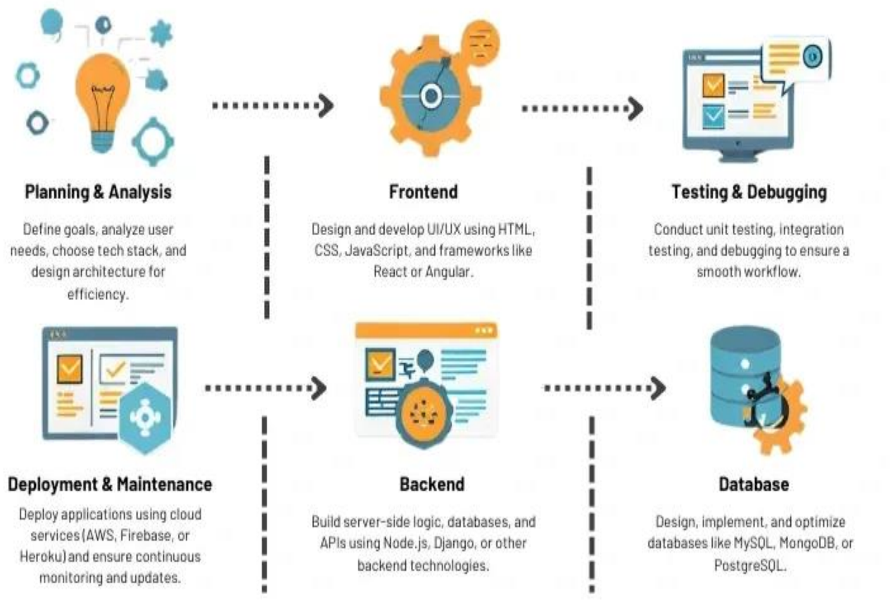
Back-End (API): Testing GET/POST requests, status codes (e.g., 200 OK, 404 Not Found), and JSON data structure.



6.2 Test Procedure

Executed manual and functional testing. Verified input validation and error handling.Testing a full-stack application involves a comprehensive procedure that validates every layer, from the user interface to the underlying database and server logic. The process begins with unit testing, where developers use tools like Jest or Mocha to isolate and verify the smallest parts of the application, such as individual functions or UI components. This is followed by integration testing, which ensures that different modules—such as the frontend communicating with the backend API or the backend interacting with the database—work together seamlessly without data loss or connection errors.

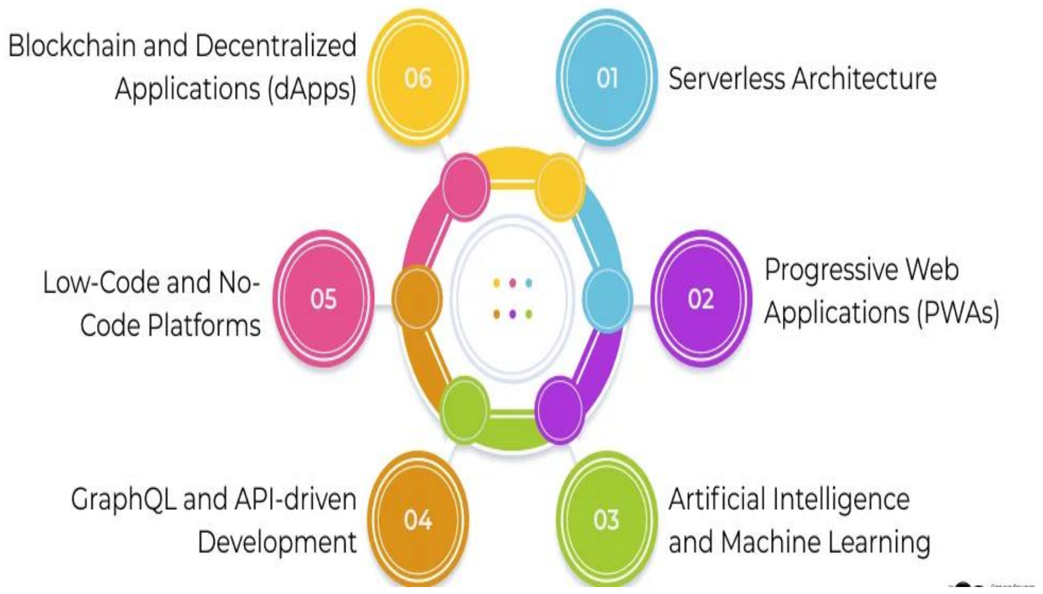
Once these layers are stable, End-to-End (E2E) testing is performed using frameworks like Cypress or Playwright to simulate real user journeys, such as signing up or completing a checkout process, in a live browser environment.


Planning & Analysis
Define goals, analyze user needs, choose tech stack, and design architecture for efficiency.
Frontend
Design and develop UI/UX using HTML, CSS, JavaScript, and frameworks like React or Angular.
Testing & Debugging
Conduct unit testing, integration testing, and debugging to ensure a smooth workflow.
Deployment & Maintenance
Deploy applications using cloud services (AWS, Firebase, or Heroku) and ensure continuous monitoring and updates.
Backend
Build server-side logic, databases, and APIs using Node.js, Django, or other backend technologies.
Database
Design, implement, and optimize databases like MySQL, MongoDB, or PostgreSQL.

6.3 Performance Outcome

The application showed fast response time and stable performance under testing conditions.Full-stack development performance is defined by the seamless synergy between frontend responsiveness and backend reliability. A high-performing outcome manifests as a fully functional, deployed web application that maintains visual stability and speed, often measured by Core Web Vitals like Largest Contentful Paint.

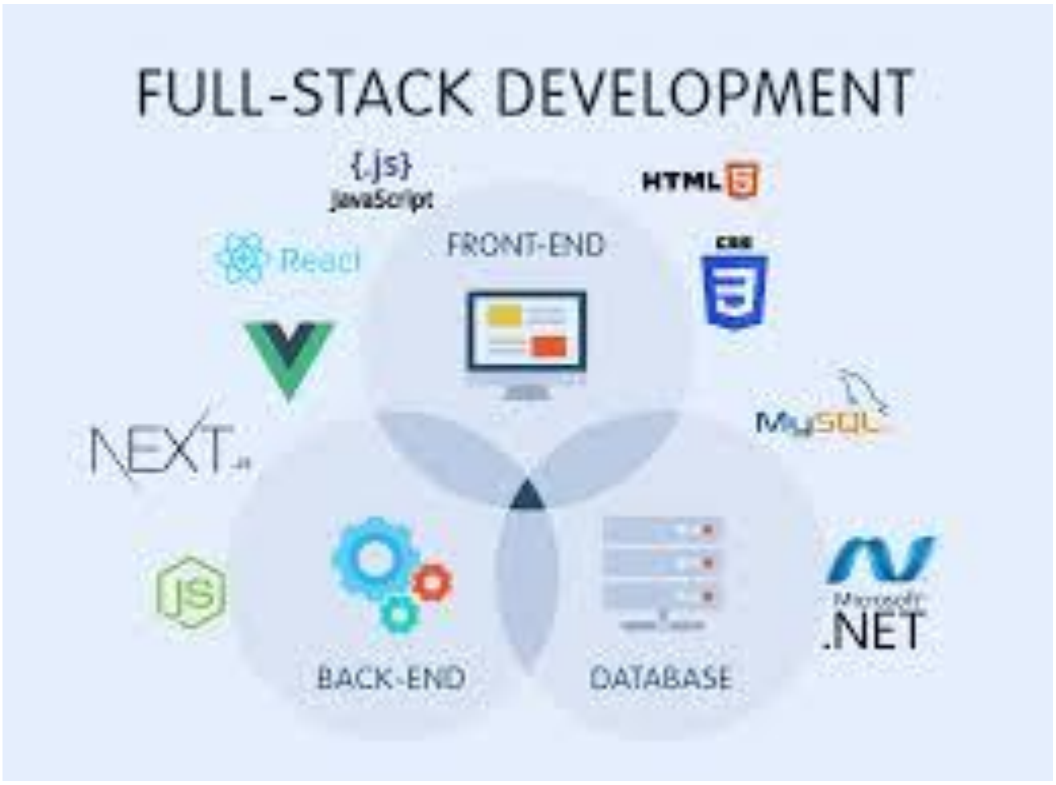
Beyond the UI, success involves optimised database access and the implementation of secure API development to handle high traffic. Ultimately, these outcomes enable a faster time-to-market by allowing a single developer or team to manage the entire lifecycle—from initial wireframing and state management to final cloud deployment and server-side scaling.


01 Serverless Architecture
02 Progressive Web Applications (PWAs)
03 Artificial Intelligence and Machine Learning
04 GraphQL and API-driven Development
05 Low-Code and No-Code Platforms
06 Blockchain and Decentralized Applications (dApps)

7. My Learnings

In 2026, the visual journey of learning full-stack development follows a structured progression from user interface design to complex infrastructure. It begins with front-end fundamentals, where learners visualize the structure of web pages through HTML and style them with CSS, eventually adding interactivity using JavaScript. This phase often evolves into mastering modern frameworks like React or Vue to build responsive, dynamic user experiences. The learning path then transitions into back-end architecture, focusing on server-side logic using languages such as Node.js, Python, or Java. This stage is visually represented by data flow diagrams and API structures that connect the front-end to databases—both relational systems like SQL Server and NoSQL options like MongoDB. To complete the "full stack," developers in 2026 also integrate DevOps and cloud deployment skills, learning to host applications on platforms like AWS or Azure, while incorporating emerging trends such as AI-powered features and microservices architecture.

- Developed frontend and backend integration skills
- Understood REST API development
- Learned database connectivity
- Improved debugging and documentation skills
- Gained real industry exposure

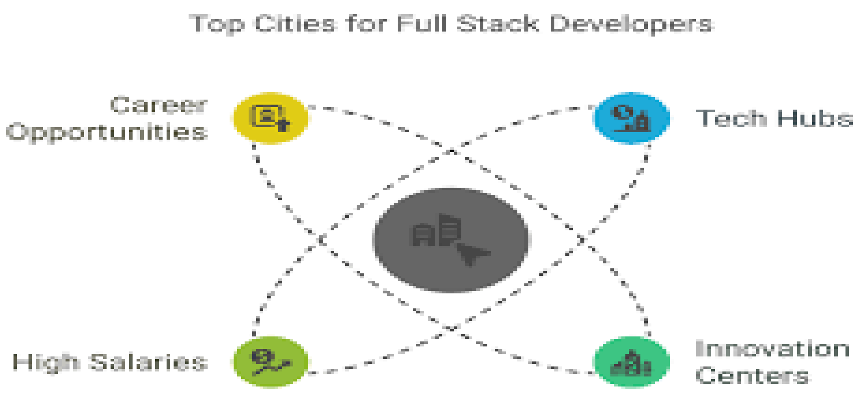


8. Future Work Scope

Full stack development remains a cornerstone of the tech industry, with a projected 13% job growth through 2030, according to the U.S. Bureau of Labor Statistics. The future of this role is evolving from traditional coding to AI-assisted engineering, where developers leverage generative AI tools to automate repetitive tasks and focus on complex architecture. Companies increasingly favor full stack professionals for their ability to manage end-to-end projects, which reduces team sizes and enhances project efficiency in sectors like fintech and SaaS. Key emerging trends include the integration of cloud-native architectures, edge computing, and Progressive Web Apps (PWAs) to deliver faster, more secure user experiences.

The scope of work is also expanding into DevSecOps, requiring developers to embed cybersecurity protocols directly into their workflows. Furthermore, as businesses move toward decentralized models, knowledge of Web3 and blockchain is becoming a significant differentiator for senior roles.

- Cloud deployment (AWS/Azure)
- Payment gateway integration
- Mobile application version
- Role-based access control
- Enhanced security implementation



GITHUB REPOSITORY LINKS:
Code File link:https://github.com/DIVYA663/upskillcampus/blob/BankingInformationSystem.java
Report File link:https://github.com/DIVYA663/upskillcampuBankingInformationSystem_Divya_USC_UCT.PDF