An Internship Report on

"Full-Stack AI-Enhanced Learning Platform for Elastyx"
&
"Masarrati IT Studio LLP."

Submitted to

DR. BABASAHEB AMBEDKAR TECHNOLOGICAL UNIVERSITY, LONERE

in fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

By

Mohd Muddabiruszama

Under the Guidance

of

Mr. Pankaj Pawar

(Department of Computer Science and Engineering)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MAHATMA GANDHI MISSION'S COLLEGE OF ENGINEERING
NANDED (M.S.)

Academic Year 2024-25

Certificate

This is to certify that the Internship entitled

"Full-Stack AI-Enhanced Learning Platform for Elastyx"

being submitted by ***Mr. Mohd Muddabiruszama*** *to the Dr. Babasaheb Ambedkar Technological University, Lonere , for the award of the degree of Bachelor of Technology in Computer Science and Engineering, is a record of bonafide work carried out by him under my supervision and guidance. The matter contained in this report has not been submitted to any other university or institute for the award of any degree.*

Mr. Pankaj Pawar

Guide

Dr. A. M. Rajurkar

H.O.D

Computer Science & Engineering

Dr. G. S. Lathkar

Director

MGM's College of Engg., Nanded

Internship Offer Letter

March 5, 2025

Mohd Muddabiruszama

Intern fullstack developer

Nanded, 431604, MH

+91 8275668600

Dear Muddabiruszama,

We are pleased to offer you a 6-month internship at Masarrati IT Studio as a Full-Stack Developer. This internship will provide you with hands-on experience, exposure to real-world projects, and an opportunity to collaborate with our development team.

Internship Details:

- Position: Full-Stack Developer (Intern)
- Duration: March 15, 2025 – Sept 15, 2025
- Location: Work From Office
- Reporting To: Mohammed Usman

Key Responsibilities:

- Develop and maintain web applications using React, Next.js, and Node.js.
- Collaborate with the team on coding, debugging, and testing.
- Follow best practices and contribute to project improvements.

Rules and Regulation:

Work Timings: Follow official working hours and maintain attendance.

Punctuality: Be on time; inform in advance for any leave.

Professional Behavior: Maintain a respectful attitude, avoid personal work during office hours.

MASARRATI IT STUDIO LLP.
Mount Banjara Complex Road No 12
Banjara Hills Hyderabad-500034
+ 919700148476
Itstudio@masarrati.com
www.masarrati.com

Best regards,

Mohammed Khubaib

HRM

Dated: 22-June-2025

To
MOHAMMED MUDDABIRUSZAMA
INTERN FULL-STACK DEVELOPER

INTERNSHIP CERTIFICATE

This is to certify that **Mohammed Muddabiruszama** is currently working as an **Intern – Full-Stack Developer** at Masarrati IT Studio LLP. since March 15, 2025.

During his ongoing internship, Muddabir has shown a high level of commitment, technical proficiency, and teamwork. He has actively contributed to the development and maintenance of web applications using React, Next.js, and Node.js, and has worked collaboratively with the team in debugging, testing, and implementing best practices.

Till date, he has performed well and continues to be a valuable part of our development efforts.

This certificate is being issued upon his request for academic or professional purposes. We acknowledge and appreciate his contributions and look forward to his continued performance.

Yours Sincerely,
FOR
MASARRATI IT STUDIO LLP
Mount Banjara complex. Road no. 12,
Banjara Hills, Hyderabad, Telangana,

Mohammed Khubaib
HR MANAGER

ACKNOWLEDGEMENT

I am greatly indebted to our Project guide **Mr. Pankaj P. Pawar** enable guidance throughout this work. It has been an altogether different experience to work with him and we would like to thank him for his help, suggestions and numerous discussions.

I gladly take this opportunity to thank **Dr. A. M. Rajurkar** (Head of Computer Science & Engineering, MGM's College of Engineering, Nanded).

I am heartily thankful to **Dr. G. S. Lathkar** (Director, MGM's College of Engineering, Nanded) for providing facility during progress of Project; also for her kindly help, guidance and inspiration.

Last but not least we are also thankful to all those who help directly or indirectly to develop this project and complete it successfully.

With Deep Reverence,

Mohd Muddabiruszama(21)

[B-Tech CSE-A]

ABSTRACT

This project report outlines the complete development lifecycle of Elastyx, an innovative EdTech platform designed to bring AI, IoT, and cloud-based learning into the hands of students in a more interactive and accessible way. Built as part of an in-office project at Masarrati Pvt. Ltd., the platform provides a structured path for learners through five progressive stages, beginning with basic electronics and moving toward advanced applications like computer vision and machine learning. The project was executed using a robust full-stack technology stack—with React.js powering the frontend interface, Flask managing backend services, and MySQL handling relational data storage. Together, these technologies enabled the creation of dynamic modules such as product listings, wishlists, cart management, and an AI-based recommendation engine that delivers customized kit suggestions based on student activity.

A major highlight of the platform is its ability to blend e-commerce functionality with educational impact. Students can explore curated tech kits, receive smart suggestions based on their learning stage, and complete purchases within a seamless, responsive interface. Features were built with an emphasis on performance, reusability, and mobile compatibility. To ensure reliability, tools like Postman were used for API testing, GitHub for version control, and Vercel/Render for real-time deployment and monitoring.

What sets this project apart is its focus on both the user experience and the learning journey. By designing a system that adapts to different user needs and provides hands-on learning through physical kits, Elastyx demonstrates how thoughtful software engineering can directly support modern education. The solution aligns with India's National Education Policy by promoting project-based, multidisciplinary learning and reflects real- world readiness in both technical and pedagogical design.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

This internship was part of a final-year curriculum at Masarrati Pvt. Ltd., a Hyderabad-based EdTech company. The intern worked under the guidance of a senior technical lead, contributing to the company's flagship AI learning platform, Elastyx. The platform combines hardware kits with software to teach students AI, IoT, and Python.

1.1 Background

As part of my final-year curriculum, my friend and I was selected for an in-office internship at Masarrati Pvt. Ltd., an emerging EdTech company based in Hyderabad. The internship began on 20th March 2025 and concluded on 30th July 2025, during which I worked under the mentorship of Mr. Usman, a senior technical lead at Masarrati. The experience provided us with deep exposure to industry practices, software development workflows, and collaboration in a real-world team environment. my role was specifically aligned with Masarrati's flagship initiative, Elastyx, which delivers AI-based learning solutions to schools and colleges.

Masarrati Pvt. Ltd. is known for its innovative education platforms that blend hands- on learning kits with digital content powered by technologies like AI, IoT, Python, and Cloud. Their mission is to empower students with real-world tech skills through structured, stage-based learning. The project I worked on Elastyx is designed to make learning fun, interactive, and future-ready by providing students with guided tutorials, development boards, and AI kits. I was involved in enhancing the digital side of this platform, contributing to user interfaces, API integration, and backend connectivity. my responsibilities included full-stack development tasks such as building responsive components in React.js, creating RESTful APIs using Flask, and handling database operations in MySQL. I worked together to design a smooth user experience for students, enabling them to browse, select, and purchase learning kits.

The internship not only allowed us to improve my technical abilities but also helped us understand the importance of teamwork, agile project delivery, and creating educational impact through technology. Reflecting on the entire internship journey, the real value lay not just in technical achievements but in the personal growth that emerged through facing real-world challenges. Working on a live product like Elastyx allowed me to witness firsthand how technology can be used as a bridge between education and innovation. The satisfaction of seeing my code in action, helping real users interact with learning tools, brought a new

dimension to my understanding of software development.

In addition to my development tasks, I was encouraged to understand the educational impact and user journey of the Elastyx platform. This involved exploring how different student personas interact with the system, from browsing kits to completing learning modules. I observed how UI design and system responsiveness directly affect engagement and comprehension in an educational context. Furthermore, the emphasis on testing and debugging highlighted the need for continuous validation across development stages.

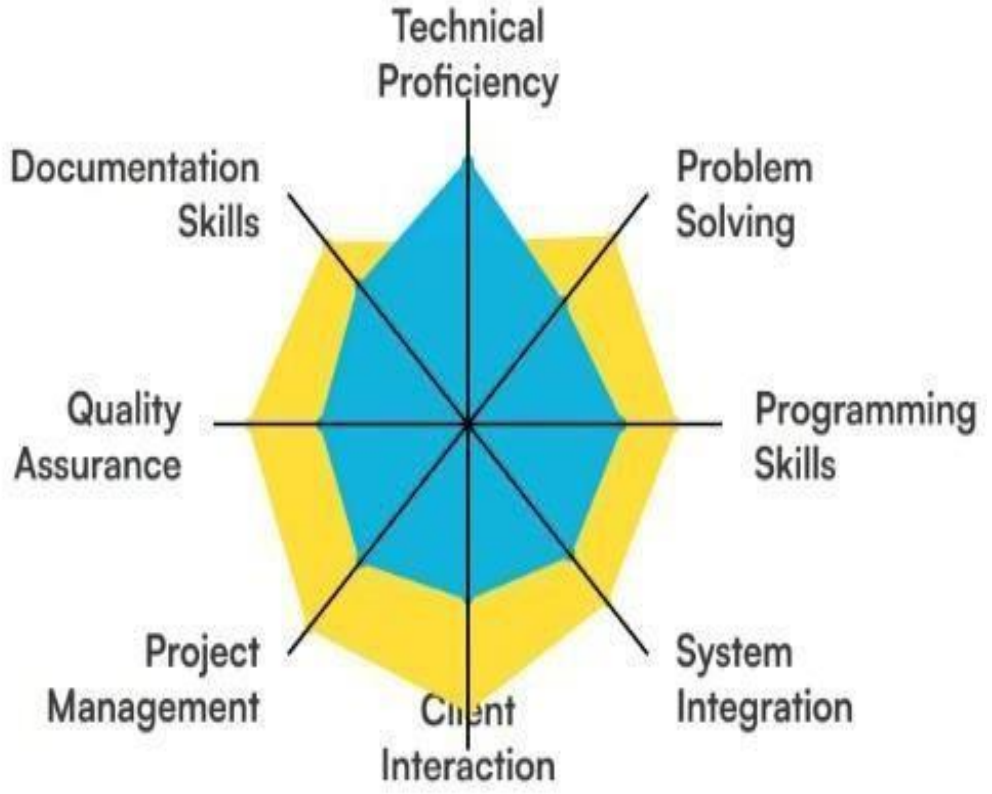

Fig. 1.1: Key Aspects of Application Engg

As shown in the figure 1.1, the diagram titled "Key Aspects of Application Engineering" visually represents the core skill areas developed during my internship. It illustrates the interconnected responsibilities involved in full-stack development, such as frontend design, backend integration, database management, quality assurance, and client interaction. Each of these aspects played a vital role in the successful development of the Elastyx platform. Working on a live product like Elastyx allowed me to witness firsthand how technology can be used as a bridge between education and innovation. This figure encapsulates how modern full-stack development demands both technical versatility and

collaborative problem-solving, skills I actively practiced throughout my internship at Masarrati Pvt. Ltd.

Frontend development was essential for creating a seamless and intuitive user interface, while backend logic—built using Flask—handled real-time processes like user authentication and order management. The MySQL database ensured secure, structured storage and fast retrieval of data, enabling consistent performance. Equally important was the focus on testing and debugging, which emphasized the need for continuous validation at every stage of the development cycle. This figure captures the holistic nature of modern application engineering, where technical proficiency must be complemented by problem-solving, collaboration, and documentation—skills I consistently applied and strengthened throughout my internship at Masarrati Pvt. Ltd.

1.2 Project Overview

During my internship at Masarrati Pvt. Ltd., my primary focus was to contribute to the enhancement and development of their AI-driven educational platform, Elastyx. This platform is structured into five progressive learning stages, guiding students from the basics of electronics and Arduino to more advanced topics such as IoT, Machine Learning, and Computer Vision. I was involved in transforming the digital user experience—building, testing, and optimizing components that supported the core functionality of this hybrid learning system.

I developed various modules such as the product listing interface, wish list and cart system, checkout flow, and AI-based recommendation engine. These modules were developed using React.js on the frontend and connected to Flask-based APIs on the backend. The data was stored and retrieved using MySQL, where I worked on both the schema design and query optimization. my aim was to make the platform intuitive, responsive, and capable of delivering personalized learning content efficiently. I set specific technical goals such as improving my proficiency in React.js for frontend development, creating Flask APIs for backend processes, and managing MySQL databases for secure and optimized data handling. Additionally, I aimed to understand how all three layers of the technology stack communicate in a live product. I was also determined to follow best practices in version control using GitHub, ensuring that my contributions was collaborative, reviewable, and production-ready.

One of the most exciting aspects of my work was integrating intelligent features like personalized product suggestions using AI APIs. This allowed the platform to offer kits tailored to each student’s learning stage and interests. I also ensured the system was scalable

and maintained code modularity. During my internship at Masarrati Pvt. Ltd., I had the opportunity to contribute significantly to the development and improvement of their AI-powered educational platform, Elastyx. This innovative platform is designed to offer a progressive, hands-on learning experience for students through five structured stages, starting from the basics of electronics and gradually advancing to complex topics like Internet of Things (IoT), Artificial Intelligence (AI), and Computer Vision. My responsibilities spanned both frontend and backend development, which included building user-friendly interfaces in React.js, creating Flask-based APIs, and integrating secure, scalable MySQL databases. I worked on key modules like the product catalog, shopping cart, checkout system, and AI-driven recommendation features. These modules aimed to provide learners with a seamless and personalized experience, helping them explore, select, and purchase tech learning kits.

1.3 Objective

The main objective of my internship at Masarrati Pvt. Ltd. was to gain practical experience in full-stack application development by contributing to a live, scalable EdTech platform. Through the elastyx project, I aimed to apply my classroom knowledge to real-world problems by building features that directly impact students and educators. my goal was not just to write working code but to design clean, maintainable, and efficient systems that enhance user engagement and platform performance.

I set specific technical goals such as improving my proficiency in React.js for frontend development, creating Flask APIs for backend processes, and managing MySQL databases for secure and optimized data handling. Additionally, I aimed to understand how all three layers of the technology stack communicate in a live product. I was also determined to follow best practices in version control using GitHub, ensuring that my contributions was collaborative, reviewable, and production-ready. While the primary objective of my internship was to gain real-world experience in full-stack development, I also set out to deepen my understanding of how software products evolve from concept to deployment. I wanted to not only enhance my technical skills but also observe how features are prioritized, planned, and executed in a collaborative environment. By setting personal milestones—such as improving code readability, mastering Git workflows, and participating in cross-team communication—I aimed to become more than just a coder. I aspired to contribute meaningfully to a real product, learn from industry professionals, and cultivate a mindset focused on building scalable, maintainable, and user-friendly applications. This goal-oriented approach helped guide my efforts and kept my learning focused throughout the internship.

Beyond coding, my objectives included strengthening my soft skills like team collaboration, task ownership, and effective communication. I actively participated in daily standups, sprint meetings, and progress reviews. The internship was an opportunity not only to build technical solutions but also to grow as responsible, industry-ready professionals who understand the end- to-end lifecycle of a product. my aim was to make the platform intuitive, responsive, and capable of delivering personalized learning content efficiently. The primary objective of my internship at Masarrati Pvt. Ltd. was to gain hands-on experience in full-stack application development while contributing meaningfully to a live educational technology platform. Through my work on Elastyx, I aimed to bridge the gap between theoretical learning and real-world application by developing scalable, user-friendly features that directly impact learners. My technical goals were clear: improve my frontend development skills using React.js, master the creation and integration of RESTful APIs through Flask, and efficiently manage relational data with MySQL. I also focused on understanding how these three layers—frontend, backend, and database—communicate within a live system. Throughout the internship, I adhered to industry best practices, including clean code principles, modular architecture, and collaborative workflows via GitHub.

1.4 Scope of work

The scope of my internship at Masarrati Pvt. Ltd. was extensive and covered a wide range of activities across the full-stack development process. I was primarily responsible for building and refining modules on the Elastyx platform. my work spanned frontend development, backend API integration, and database management. This gave us an end-to-end understanding of how web applications are planned, designed, developed, tested, and deployed in a real-world environment. On the frontend, I developed responsive user interfaces using React.js. This included building reusable components like product cards, category filters, wish lists, and checkout forms. I ensured that every component met Masarrati’s branding guidelines and delivered a smooth user experience across devices. In parallel, I created and connected Flask-based APIs to enable real- time operations such as adding items to the cart, placing orders, and retrieving personalized recommendations.

Beyond coding, I was also involved in testing, debugging, and documentation. I tested my modules in staging environments, identified bugs using developer tools, and wrote detailed usage documentation for future developers. On the database side, I designed and optimized MySQL tables to store products, users, and orders efficiently. This well-rounded scope helped us gain practical exposure to every part of a modern software system and strengthened both my technical and collaborative abilities. The scope of my internship extended far beyond

routine development tasks, offering exposure to the complete lifecycle of a production-level EdTech platform. I was actively involved in translating functional requirements into technical implementations, ensuring each module contributed to the broader educational goals of Elastyx. From drafting UI components and connecting RESTful APIs to handling database transactions and debugging interface anomalies, my responsibilities covered both independent problem-solving and collaborative execution. In addition, I was entrusted with maintaining documentation, contributing to sprint planning, and even suggesting minor UI enhancements based on user feedback. This comprehensive scope not only strengthened my coding proficiency but also gave me a clear understanding of how cross-functional teams operate in a real-world project environment.

I also collaborated closely with other teams, including UI/UX designers, product managers, and QA testers. This cross-functional interaction helped us align my technical implementation with user expectations and business requirements. I participated in regular sprint meetings where I presented my progress, discussed blockers, and received feedback. These experiences sharpened my ability to work in agile development cycles and exposed us to how coordination across departments contributes to the success of a product in a real-world software company.

1.4.1 End-to-End Feature Handling I worked on features from the initial planning and wireframing phase to backend connectivity and deployment. This gave us exposure to the complete software lifecycle and helped us understand how small design choices affect final user outcomes. Working on features from start to finish provided me with a complete perspective on the product development lifecycle. From wireframing the user interface to implementing backend logic and finally deploying it for real-time use, I gained a hands-on understanding of how each decision—whether technical or design-related—affects the end user. This end-to-end ownership taught me how to balance user needs with system requirements, and how to troubleshoot issues across multiple layers of the stack. I became more mindful of performance, user experience, and scalability while ensuring that every component I delivered was fully integrated and test-ready.

1.4.2 Cross-Platform Compatibility my development work involved testing and optimizing the interface across multiple screen sizes and browsers. I ensured mobile responsiveness, accessibility compliance, and visual consistency to improve the experience for all user segments. Ensuring cross-platform compatibility was a key objective during the development of the Elastyx platform,

as the end users ranged from students using desktops in schools to those accessing content via mobile devices at home. I conducted extensive UI testing on different screen resolutions and browsers to confirm that layouts remained consistent and functionality intact. This involved implementing responsive CSS layouts, flexible image rendering, and adaptive navigation elements. The use of frameworks like Flexbox and Grid allowed for better control over visual alignment across devices. I also reviewed accessibility standards, such as touch-friendly buttons and readable fonts, to guarantee usability for all learners. This focus on compatibility not only improved the overall user experience but also made the platform more inclusive and reliable in diverse environments.

1.4.3 Platform Compatibility Platform compatibility played a crucial role in ensuring that the Elastyx system could operate smoothly across different operating environments and technical setups. During development, I focused on testing the web application across multiple browsers such as Chrome, Firefox, and Edge, while also considering OS-level behaviors on Windows, macOS, and Android devices. It was important that core functionalities—like product display, user login, and checkout processes—remained consistent regardless of the platform used. Special attention was given to handling API responses uniformly and ensuring that third-party libraries did not behave differently across browsers. This meticulous compatibility testing helped minimize unexpected bugs and offered a more stable, predictable experience for end users, whether they accessed the platform on a desktop, tablet, or mobile phone. I conducted extensive UI testing on different screen resolutions and browsers to confirm that layouts remained consistent and functionality intact. The company follows Agile principles and hosts daily stand-up meetings, sprint planning, and review sessions. These practices helped us stay organized and focused on my weekly deliverables. I was also encouraged to explore new tools and present my findings, which helped build my research and communication skills. The work culture at Masarrati Pvt. Ltd. was both inclusive and empowering, fostering an environment where every team member—regardless of designation—was encouraged to share ideas and contribute meaningfully. What stood out most was the flat hierarchy that enabled open discussions with senior developers, mentors, and even the leadership team. This accessibility not only accelerated learning but also created a sense of ownership and accountability among interns like myself. The development team I was part of worked closely with the design and QA units, offering me a complete

view of the product pipeline. This involved implementing responsive CSS layouts, flexible image rendering, and adaptive navigation elements. The use of frameworks like Flexbox and Grid allowed for better control over visual alignment across devices.

1.5 Company Overview

Masarrati Pvt. Ltd., officially known as Masarrati IT Private Limited, is a Hyderabad-based EdTech company founded with the mission of transforming the way students learn emerging technologies like Artificial Intelligence, Cloud, and IoT. The company is recognized under India's Startup India initiative and has built a reputation for offering hybrid learning models that combine physical learning kits with digital content. Their innovative approach helps learners apply theoretical concepts in practical, hands-on environments.

Masarrati Pvt. Ltd., officially known as Masarrati IT Private Limited, is a Hyderabad-based EdTech company founded with the mission of transforming the way students learn emerging technologies like Artificial Intelligence, Cloud, and IoT. The company is recognized under India's Startup India initiative and has built a reputation for offering hybrid learning models that combine physical learning kits with digital content. Their innovative approach helps learners apply theoretical concepts in practical, hands-on environments. The company caters to a wide spectrum of learners-from school students to engineering graduates-by offering structured learning modules, skill-building projects, and real-time applications. Masarrati has also collaborated with educational institutions, skill centers, and NGOs to make future-ready technology education accessible across India. Its platforms emphasize creativity, curiosity, and a problem-solving mindset.

Beyond its technology offerings, Masarrati Pvt. Ltd. stood out for its unwavering focus on empowering students through experiential learning. The company doesn’t merely deliver products; it cultivates curiosity by embedding real-world problem-solving within its educational kits. During my time with the team, I observed a deep sense of purpose behind every decision—from UI design to lesson flow—which ensured that learning remained inclusive, accessible, and transformative. This people-first, impact-driven approach differentiates Masarrati in the EdTech ecosystem and reflects a future-facing vision grounded in today’s learning realities.

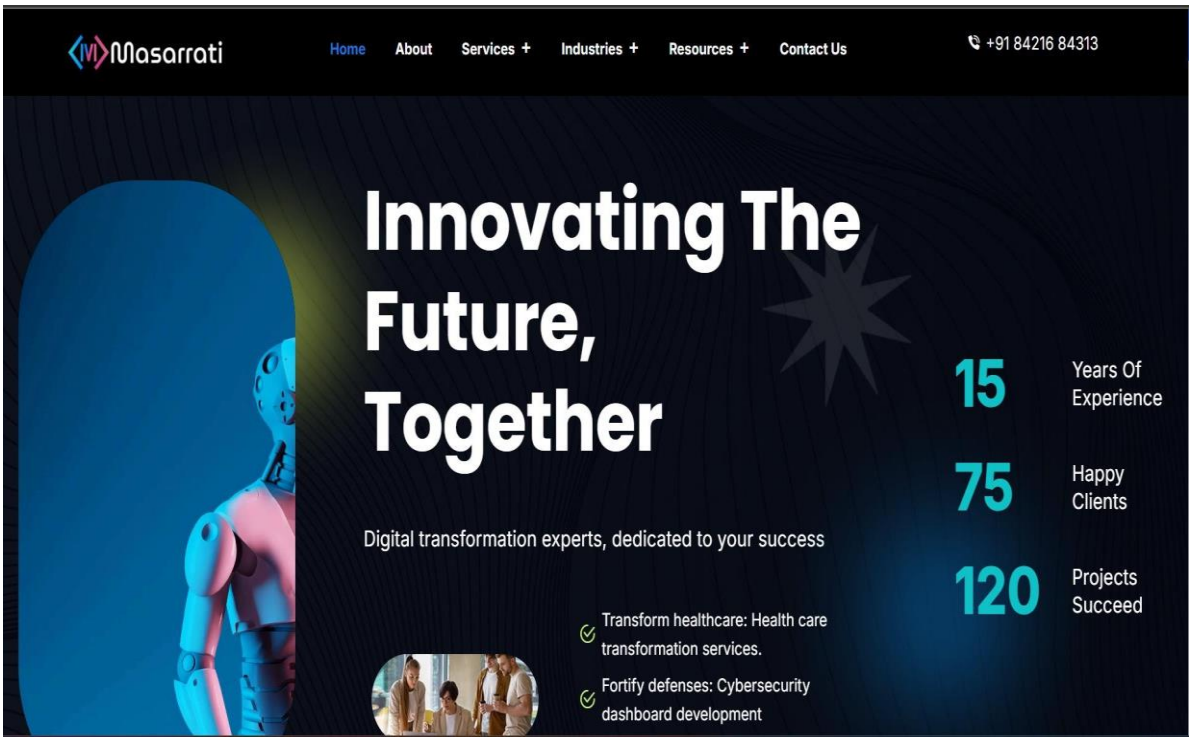

Fig. 1.2: Masarrati Dashboard

As shown in the figure 1.2, Masarrati Pvt. Ltd. positions itself as a leader in digital transformation, with 15+ years of experience, over 75 happy clients, and more than 120 successful projects. Beyond its robust service portfolio, the company is deeply invested in educational innovation. Masarrati not only builds cutting-edge digital products but also actively supports training workshops, tech festivals, and coding bootcamps to promote real-world learning.

Its team—comprising educators, software developers, and engineers— collaborates to ensure that the content and platforms remain both technically advanced and learner-friendly. Masarrati stands out by offering end-to-end EdTech solutions-from content development and delivery platforms to hardware kits and AI-powered recommendation engines. The commitment to innovation and quality is evident in every product, including Elastyx, its flagship learning platform.

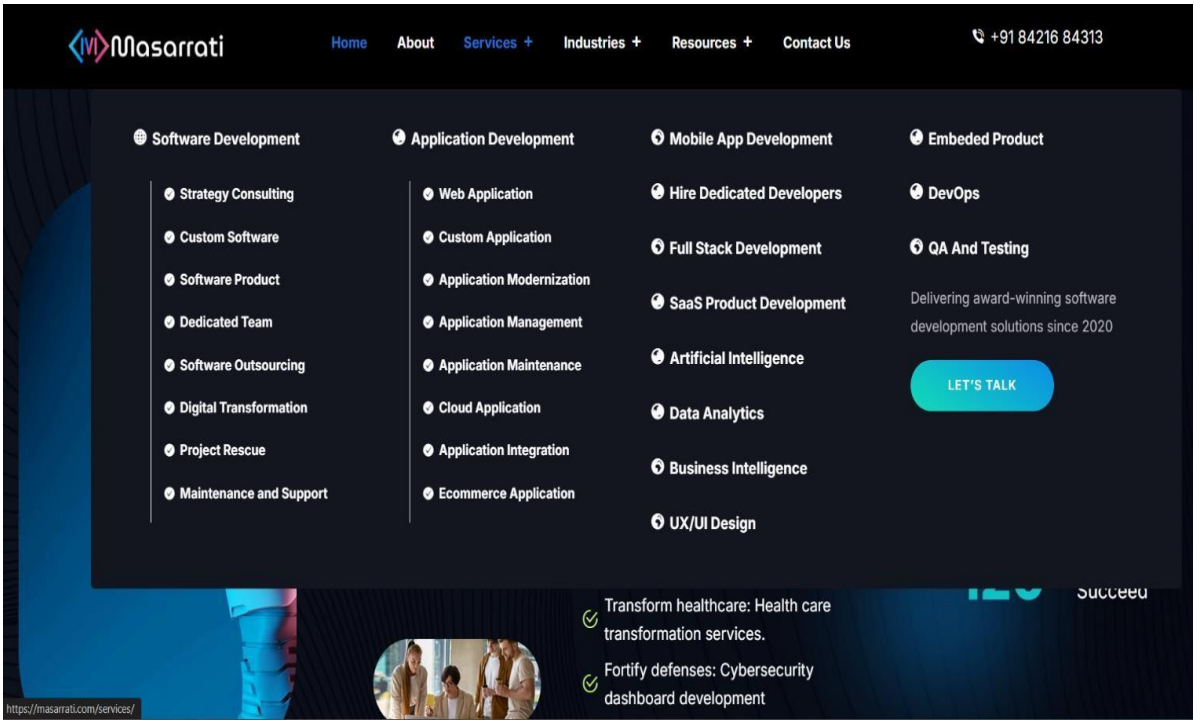

Fig. 1.3: Masarrati's Services

As shown in the figure 1.3, Masarrati Pvt. Ltd. offers a wide range of cutting-edge services across the domains of software development, application development, mobile solutions, and embedded technologies. The company's service architecture is built to support clients throughout the entire software lifecycle—from strategy consulting, custom product development, and cloud integration, to DevOps, AI solutions, and QA testing. This comprehensive service lineup reflects Masarrati's commitment to delivering tailored, scalable, and future-ready solutions for diverse business needs.

1.6 Vision and Mission

Masarrati's vision is to empower the next generation of innovators by delivering future-oriented education that bridges the gap between classroom learning and real-world technology. The company envisions a world where every student has access to high-quality STEM education, regardless of their background or geography. It believes in making AI, Cloud, and coding mainstream for learners at an early age.

The mission is to offer scalable, accessible, and affordable learning experiences using modern teaching tools and well-designed curricula. Masarrati's team is committed to making education hands-on and inclusive, so students are not just consumers of technology—but creators. They work with schools and institutions to integrate these modules into regular

academics. The company's methodology is built around five key values: Creativity, Accessibility, Practical Learning, Inclusion, and Innovation. These values are not only part of internal culture but are reflected in how Masarrati designs its platforms and interacts with learners. By combining project- based learning with expert mentoring, Masarrati aims to produce capable, confident, and curious individuals. Their mission also supports the goals of India's National Education Policy (NEP), emphasizing multidisciplinary learning and 21st-century skills. In essence, Masarrati strives to be a technology- driven education partner for every learner in India and beyond.

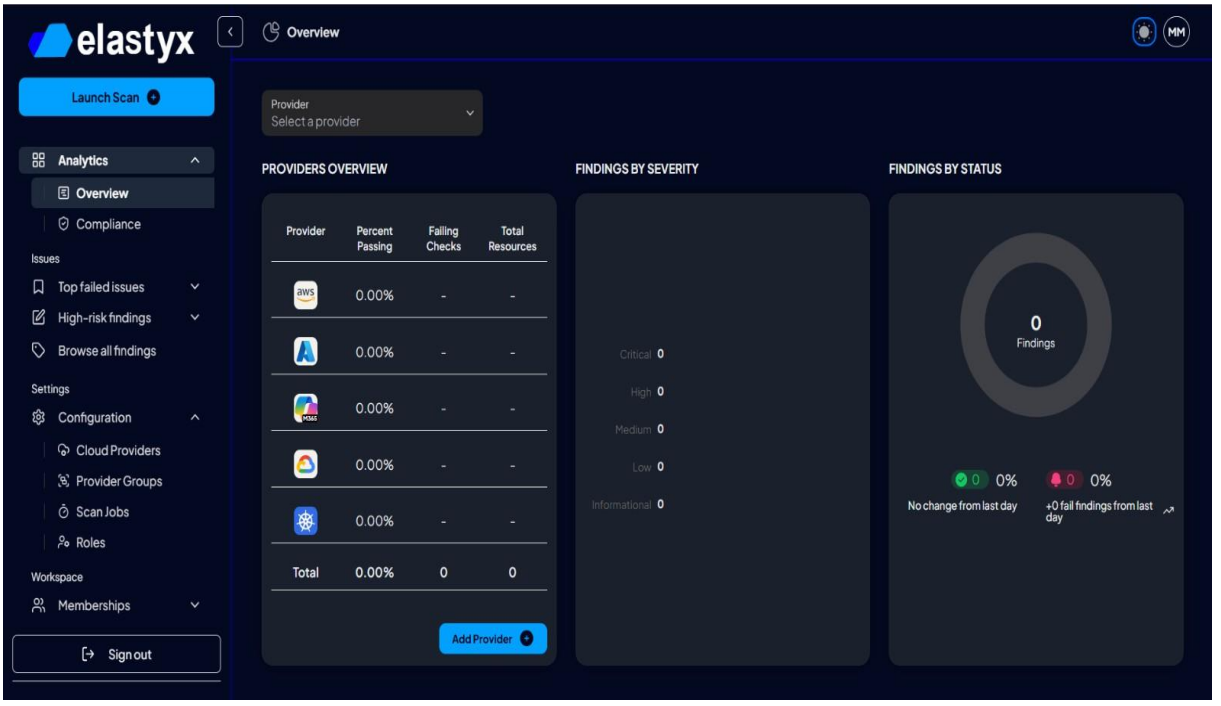

Fig 1.4 Elastyx Dashboard View

As shown in the figure 1.4, the *Elastyx Dashboard View* serves as a centralized and intelligent interface that exemplifies the platform's user-focused design approach. This unified dashboard integrates key features such as provider analytics, compliance checks, issue tracking, and system configuration, all within a sleek, responsive layout. It offers users a real-time snapshot of system performance, highlighting scan results, issue severity, and provider-wise resource health. During my internship, I contributed to making this dashboard both functional and visually engaging by ensuring smooth integration of React components with live backend data. The dashboard was designed to give users an intuitive understanding of their learning path while maintaining responsiveness across devices. It acts not just as a

control panel, but as an intelligent interface that evolves based on user behavior and engagement data.

This figure encapsulates how thoughtful UI/UX planning, paired with smart data handling, results in an experience that empowers students to take ownership of their tech learning journey. Masarrati Pvt. Ltd. stands apart not only because of its cutting-edge technology stack but also due to its strong focus on transforming education into a hands-on, student-centric experience. The company's leadership believes that innovation in learning must be deeply aligned with student needs, practical exposure, and evolving industry trends. During my internship, I observed how each product or module—particularly the Elastyx platform—was created with a sharp emphasis on usability, accessibility, and long-term impact. The company's ability to combine educational theory with hardware kits, interactive dashboards, and real-time mentoring created a learning environment that was both engaging and scalable. Furthermore, the open-door culture encouraged feedback from interns and team members alike, making the company not just a workspace but a learning community driven by curiosity and continuous improvement.

1.7 Flagship Initiative-Elastyx

Elastyx is Masarrati's flagship learning product, focused on empowering school and college students with AI, IoT, and Cloud skills. The platform follows a five-stage progressive learning model, guiding students through beginner to advanced levels of hands-on tech training. Each stage covers topics such as basic electronics, Arduino programming, sensor integration, Python development, and machine learning. Elastyx represents more than just a product—it is a strategic step toward revolutionizing how students interact with STEM education. What makes Elastyx stand out is its progressive learning approach, where each stage builds upon the previous one, blending hardware interaction with software logic. During my internship, I saw firsthand how this initiative encourages critical thinking and creativity by allowing students to experiment with sensors, write real Python code, and implement real-time AI applications.

The inclusion of guided mentorship sessions and gamified quizzes kept the learning process engaging and measurable. What impressed me most was how Elastyx not only teaches technology but also empowers students to innovate, solve problems, and become creators rather than just consumers of digital content. The Elastyx experience is made immersive through a combination of learning kits, video lessons, quizzes, and mentor-led sessions. Students build real-world projects using kits that include sensors, motors,

development boards, and cameras. These projects are not only educational but also fun, allowing learners to explore tech creativity.

Elastyx is the flagship initiative of Masarrati Pvt. Ltd., designed with a vision to revolutionize STEM education for school and college students through a progressive, hands-on approach. This platform follows a structured five-stage model, starting from foundational topics like electronics and Arduino programming and progressing toward advanced subjects such as Python, AI, Machine Learning, and IoT. What sets Elastyx apart is its ability to blend physical learning kits—such as sensors, development boards, and motors—with an engaging digital interface. Through video tutorials, mentor-guided sessions, and gamified quizzes, students are not only introduced to theoretical knowledge but are empowered to apply it in real-world scenarios. During my internship, I worked closely on the digital backbone of Elastyx, helping to shape the user interface, optimize backend integration, and ensure smooth data interactions that supported this immersive learning journey.

1.8 Work Culture and Team Structure

Masarrati's work environment is built on collaboration, creativity, and continuous learning. As interns, I was immediately welcomed into the team and given meaningful responsibilities. The company operates with a flat hierarchy, allowing team members to share ideas freely and get direct feedback from mentors and senior developers. Teams are divided into focused units: software development, AI and data science, curriculum design, and product testing. my team was part of the development unit, but I frequently interacted with others during project planning and feature integration. This cross-functional collaboration helped us understand the bigger picture of how a product is built and launched.

The company follows Agile principles and hosts daily stand-up meetings, sprint planning, and review sessions. These practices helped us stay organized and focused on my weekly deliverables. I was also encouraged to explore new tools and present my findings, which helped build my research and communication skills. The work culture at Masarrati Pvt. Ltd. was both inclusive and empowering, fostering an environment where every team member—regardless of designation—was encouraged to share ideas and contribute meaningfully. What stood out most was the flat hierarchy that enabled open discussions with senior developers, mentors, and even the leadership team. This accessibility not only accelerated learning but also created a sense of ownership and accountability among interns like myself. The development team I was part of worked closely with the design and QA

units, offering me a complete view of the product cart add pipeline. Cross-functional meetings, regular sync-ups, and an agile workflow helped streamline collaboration, ensuring everyone was aligned on project priorities and sprint goals. This experience taught me that a strong, communicative team culture is just as critical to a project's success as the technology behind it.

Overall, the culture at Masarrati promotes both personal growth and technical excellence. As an intern, I never felt limited by my position when it came to offering suggestions or asking for feedback. I gained not only coding experience but also a sense of ownership and confidence that comes from working in a supportive and forward-thinking organization.

- **Flat Hierarchy and Open Dialogue** Masarrati's team structure enabled us to approach mentors and senior developers directly. The absence of rigid hierarchies fostered transparency, active discussion, and faster resolution of challenges across departments. The flat hierarchy at Masarrati Pvt. Ltd. created a uniquely open and encouraging atmosphere, where ideas could be shared freely without the barriers of traditional organizational layers. As an intern,This culture of transparency allowed for faster decision-making and built mutual respect among team members. More importantly, it created a learning environment where questions were welcomed, and mentorship was natural. The ability to directly engage with experienced professionals not only enhanced my confidence but also accelerated my growth as a developer.

- **Agile Workflow and Weekly Goal** Through daily stand-ups and sprint reviews, I maintained a clear roadmap of tasks. This agile approach enhanced my productivity, encouraged feedback loops, and helped track progress with measurable goals. Working within an agile framework helped me develop a structured and goal-oriented mindset throughout the internship. Each week began with clear sprint planning sessions, where tasks were divided into achievable milestones with measurable outputs. Daily stand-ups provided a consistent feedback loop, allowing us to track progress, resolve issues collaboratively, and maintain momentum. I particularly appreciated the iterative nature of agile—it taught me that progress doesn't need to be perfect on the first try but should evolve through cycles of improvement. This methodology kept our workflow transparent, focused, and aligned with both short-term deliverables and the platform's long-term vision.

- **Cross-Functional Learning** Working with different teams beyond just development helped us understand how design, content, and quality assurance contribute to the final product. These interactions expanded my perspective beyond coding. One of the most enriching aspects of the internship was the opportunity to engage in cross-functional learning. I was not confined to working solely within the development team; instead, I had regular interactions with UI/UX designers, QA testers, product managers, and mentors across departments. These collaborations exposed me to how different roles come together to shape a feature—from conceptual design to final deployment. For instance, understanding design logic from the UX team helped me build more intuitive interfaces, while feedback from QA ensured my code met real-world usability standards. This broader exposure not only expanded my technical know-how but also taught me to appreciate the importance of empathy, coordination, and clarity in a multidisciplinary team environment.

1.9 Organization of the Report

This report is divided into five chapters:

- Chapter 1: Introduction – Introduces the project, its motivation, objectives, and scope related to heart failure prediction using machine learning.

- Chapter 2: Technology and Tools – Describes all the technologies used such as the Python programming language, machine learning algorithms (Random Forest, KNN, Logistic Regression), development tools like Jupyter Notebook, and supporting libraries including NumPy, Matplotlib, and Pandas.

- Chapter 3: Internship Responsibilities – Details the specific tasks undertaken during the internship, including data collection, preprocessing, model training, evaluation, and participation in the deployment of the heart failure prediction system.

- Chapter 4: Key Learning and Skills Developed – Highlights the technical and professional skills acquired during the project, such as practical experience in machine learning, understanding of clinical datasets, use of development frameworks, and deployment using Flask.

Chapter 2

SYSTEM ANALYSIS AND DESIGN

During my internship at Masarrati Pvt. Ltd., I worked within a well-structured full-stack development environment that combined modern and industry-relevant technologies like React.js for frontend, Flask for backend API development, and MySQL for relational data management. This technology stack formed the backbone of the Elastyx platform, supporting the development of an interactive, real-time educational web application. React.js enabled me to create dynamic and reusable components that shaped a responsive user interface across devices, while Flask helped handle backend logic and RESTful API services for user sessions, product handling, and checkout flow. MySQL was used for efficient storage and retrieval of structured data such as user profiles, product catalogs, and order histories. Each component in this tech stack was selected with scalability, performance, and reliability in mind. It was rewarding to understand how these layers communicated with each other to ensure smooth, end-to-end operations.

2.1 Technology Stack

During my internship at Masarrati Pvt. Ltd., I worked within a structured full-stack development environment using React.js, Flask, and MySQL. These technologies formed the foundation of the Elastyx platform. I used them to design responsive interfaces, manage backend processes, and store structured data efficiently for learners and administrators.

React.js enabled us to build dynamic user interfaces that was both modular and scalable. Flask supported the backend logic and API handling, allowing smooth communication between the frontend and the database. MySQL helped us organize and retrieve user data, product details, and transaction records in a secure and optimized way. One of the most rewarding aspects of working with this modern tech stack was realizing how each layer—React.js, Flask, and MySQL—contributed to a seamless user experience. Learning how these tools communicate and support each other in real-time systems was incredibly enlightening. I became more mindful of architectural decisions and how small improvements in one layer (such as faster API responses or better state handling) could significantly uplift the overall system performance. The exposure to such real-world workflows helped bridge the gap between textbook concepts and enterprise-grade development.

To support the development workflow, I also used GitHub for version control, Postman for testing APIs, and deployment platforms like Vercel and Render. This setup provided a complete project lifecycle—from writing and testing code to deploying it for real-time usage. Together, the tools helped us create a stable, scalable, and maintainable application.

- React.js A JavaScript library used to build fast, reusable, and interactive UI components across multiple Elastyx pages.
- Docker Docker is a platform that lets you package applications and their dependencies into lightweight, portable containers that run consistently across different environments.
- MySQL A relational database system where I created tables and relationships to store products, users, cart data, and purchase records.
- GitHub Used to manage code repositories, track changes, review pull requests, and collaborate in a shared development space.
- Node.js Node.js is a JavaScript runtime that lets you run JavaScript code on the server side, enabling fast and scalable backend development.

2.2 Frontend Development with React.js

Frontend development was one of the most crucial aspects of my internship at Masarrati Pvt. Ltd. As application engineering interns, I was responsible for designing and developing user- facing components that directly impacted how learners interacted with the Elastyx platform. I used React.js to build responsive, interactive, and reusable components that made navigation and functionality simple and intuitive for users. This included pages like product listings, category filters, wishlist views, cart handling, and the checkout process.

React's component-based architecture allowed us to build and reuse functional blocks, which improved both development speed and consistency across the platform. I developed key UI elements like ProductCard, CartItem, and WishlistTile, each responsible for rendering specific sets of data and actions. These components were managed through props and hooks like useState and useEffect, which allowed us to maintain internal state and respond dynamically to user interactions. I also used conditional rendering to control UI behavior based on login state or cart status, ensuring a seamless experience. Styling was done using both standard CSS and utility-based frameworks that supported responsive design. I ensured

that every screen worked smoothly on both desktop and mobile devices using layout models like Flexbox and Grid. Beyond layout, I added animations and effects to enhance interactivity, including hover effects, transition animations, and button feedback. I also optimized image loading using lazy-loading techniques to improve page speed and overall performance. All components followed Masarrati's branding guidelines and contributed to a consistent user experience. I developed key UI elements like ProductCard, CartItem, and WishlistTile, each responsible for rendering specific sets of data and actions.

I paid close attention to accessibility and usability throughout the interface. Elements such as readable fonts, color contrast, clickable areas, and keyboard navigability was taken into account to ensure that the platform was inclusive and easy to use for all users. Every feature I developed was tested in-browser using dev tools, and bugs was resolved promptly. Through the frontend, I ensured that all users-students, mentors, and administrators-could navigate the learning platform easily and perform their desired actions without friction.

- Component Reusability and UI Responsiveness I focused on modular design, responsive breakpoints, and dynamic content rendering.

- UX Enhancements and Accessibility Added hover effects, real-time feedback, proper spacing, keyboard navigation, and visual hierarchy for ease of use.

2.2.1 Component Reusability and UI Responsiveness I emphasized creating modular React components that could be reused across multiple pages. This approach reduced code redundancy and allowed us to quickly implement changes without affecting the entire UI. I also ensured that every component was responsive, adapting smoothly to screen sizes from desktops to mobile devices. One of the key principles I focused on during frontend development was ensuring component reusability and UI responsiveness. Using React.js, I followed a modular approach by designing UI components that could be reused across different sections of the Elastyx platform. Elements such as ProductCard, CartItem, WishlistTile, and form inputs were developed as independent blocks, allowing them to be easily maintained, scaled, or adapted based on feature needs. This method significantly reduced code duplication and improved development efficiency, especially when multiple developers were working on similar UI modules. It also allowed for consistent behavior and design patterns across the application, ensuring that users had a cohesive and predictable experience.

2.2.2 UX Enhancements and Accessibility To ensure a seamless user experience, I incorporated hover states, animations, keyboard navigation, and sufficient spacing between interactive elements. These enhancements improved the visual appeal and usability of the interface while meeting web accessibility standards. Creating a smooth and intuitive user experience was at the core of my frontend development efforts during the Elastyx project. I worked on UX enhancements that not only made the platform visually appealing but also functional and easy to navigate for users of all ages. Features like hover effects, animated transitions, and real-time feedback (e.g., loading indicators, form validations, and button responses) were incorporated to give users a sense of interaction and clarity. I ensured that each action—whether adding a product to the cart or navigating between learning modules—was met with a responsive visual cue, making the platform feel dynamic and engaging. These enhancements weren't just for aesthetics; they played a key role in guiding users through the flow and reducing confusion during key operations like checkout or registration.

2.3 Backend Development with Flask

Backend development was a fundamental part of my internship experience, and I used the Flask framework to build and manage the server-side logic for the Elastyx platform. Flask is a lightweight yet powerful Python framework that allowed us to create RESTful APIs quickly and efficiently. These APIs formed the bridge between the frontend interface and the backend database, enabling smooth data transactions such as product display, user authentication, order processing, and wishlist management.

I structured the backend into multiple endpoints categorized by functionality, such as/Products, /cart, /orders, and /recommendations. Each endpoint handled specific requests from the frontend, processed business logic, validated the data, and then returned a JSON response. Flask's route decorators and modular blueprint architecture made it easier to keep the backend organized and scalable. This was especially useful when different features were being worked on simultaneously by team members. Security and error handling was also key components of my backend work. I implemented input validation checks to protect the server from bad requests and SQL injection attacks. For every route, I built in response status codes and try-except blocks to manage unexpected errors gracefully. Logging mechanisms was also integrated to track API behavior and performance during testing. This ensured a reliable and secure backend structure that scaled well with user activity.

I tested each API using Postman, simulating GET, POST, PUT, and DELETE requests. This helped us validate the request-response cycles and allowed us to identify bugs before integration. Once tested, the APIs was connected with the frontend using Axios in React. Through backend development, I gained strong exposure to building maintainable and secure services that delivered real-time interactions and supported the educational goals of the Elastyx platform. Backend development formed a crucial part of my internship experience, and I had the opportunity to work extensively with Flask, a lightweight yet powerful Python-based web framework. Flask allowed me to build clean, modular, and efficient RESTful APIs that acted as the bridge between the user interface and the database. I structured the backend using blueprints, dividing the logic into manageable segments such as /products, /cart, /orders, and /wishlist. This organization not only made the codebase easier to understand and maintain but also allowed multiple features to be developed in parallel. Each route handled specific functions—like adding an item to the cart or processing an order—while returning JSON responses for seamless frontend integration.

- **API Development and Structure** Designed modular Flask endpoints using blueprints to manage product, cart, and order logic.

- **Validation and Integration** Implemented input checks, error handlers, and connected APIs to React components using Axios

2.3.1 API Development and Routing One of my main responsibilities in backend development was creating and managing RESTful APIs using the Flask framework. I structured the API routes based on the core functionalities of the Elastyx platform, including products, carts, wishlists, orders, and user authentication. Each route was handled using Flask's @app.route decorator and was developed following a modular structure using Flask blueprints. This helped keep my codebase clean, reusable, and easy to scale. Designing and structuring APIs using Flask gave me deep insight into how application layers communicate and share data. I focused on building RESTful APIs that followed a modular and scalable architecture using Flask Blueprints, which allowed for better organization and reusability across different components. Routes were grouped by function—such as /products, /users, and /orders—to keep backend logic clean and manageable. Each route handled specific tasks like data retrieval, form submission, or update requests, and was built to return standardized JSON responses. Careful attention was given to HTTP status codes and clear endpoint naming conventions, making the API more developer-

friendly and easier to debug.

2.3.2 Data Validation and Error Handling To ensure security and system stability, I implemented detailed input validation mechanisms across all API routes. For each incoming request, I checked the data for completeness, correct format, and valid types. This helped us prevent server crashes, user-side errors, and potential injection attacks. I also created utility functions for standardizing responses and validating user sessions or input forms before database actions were triggered. Implementing data validation and robust error handling was crucial to maintaining both system security and user trust. Every API route I developed was equipped with input checks that ensured data types, field lengths, and required parameters met the expected criteria before any database operation was executed. This not only prevented common issues like missing or malformed data but also protected the backend from SQL injection and other vulnerabilities. In addition, I used try-except blocks to gracefully catch and respond to runtime errors, returning meaningful status codes and error messages that helped both users and developers understand what went wrong. Logging was also integrated to monitor unusual behavior and simplify debugging. This layer of validation and handling ensured the backend remained reliable, secure, and ready to scale with user activity. Error handling was achieved using try-except blocks and custom error messages that made it easier to debug issues during testing. I also returned standardized HTTP status codes (e.g., 200 OK, 400 Bad Request, 404 Not Found) to improve API clarity. Logging was set up for debugging complex issues and monitoring API behavior over time. These steps helped maintain backend integrity and contributed to secure, production-ready endpoints.

2.3.3 Integration with Frontend and Testing Once APIs was developed and tested, I integrated them with the frontend React components using Axios. Each component, such as the Cart or ProductDetail view, called the appropriate Flask API and displayed the resulting data. I ensured that request-response cycles were handled properly and added loading indicators and error messages on the frontend where needed. Axios helped manage promise- based asynchronous requests, making it easier to sync state changes. To ensure reliable API performance, I used Postman to simulate all kinds of requests, such as GET (fetch data), POST (add data), PUT (update data), and DELETE (remove data). Postman also helped us send invalid inputs for stress-testing my error handling logic. Through this process, I verified

that my APIs performed accurately, returned correct responses, and stayed stable under various conditions. Integrating the backend APIs with the frontend React components was a critical step that brought the entire Elastyx platform to life. Using Axios, I established secure and efficient communication between the user interface and server-side endpoints. Each frontend action—like adding a product to the cart or submitting an order—triggered a corresponding API call that returned real-time data updates. I ensured that response handling was clean and that loading states, error messages, and success notifications were presented appropriately on the UI. Postman was used extensively to test APIs independently before integration, allowing me to verify data formats, response codes, and edge cases. This approach minimized integration issues and helped maintain application stability. The successful synchronization of both ends ensured a seamless, interactive experience for users and reinforced the importance of coordination between front-end development and backend logic.

2.4 Database Management with MySQL

Database management was a key part of my backend responsibilities during the internship. I used MySQL as the primary database system to handle structured data, including users, products, orders, cart items, and interaction history. I began by understanding the existing database schema of the Elastyx platform and later contributed to designing new tables and refining relationships between them. Every table included appropriate constraints, such as primary keys and foreign keys, to ensure referential integrity and prevent orphan records.

As new features like wishlists and AI recommendations was introduced, I added relevant tables and created normalized schemas to avoid data duplication. For example, I created one-to-many relationships between users and orders, and many-to-many relationships between users and wishlists using bridge tables. These schema designs ensured that the data was logically organized and easy to query, which helped both in performance and accuracy. Performance tuning was another critical aspect of my MySQL work. I optimized SQL queries using indexing on frequently used fields such as user IDs, product categories, and timestamps. This allowed faster filtering and searching operations across large datasets. I also implemented pagination and limited row fetches in order-related queries to reduce load times and enhance the user experience on the frontend.

In terms of data security, I followed best practices like using parameterized queries to avoid SQL injection and storing sensitive user information in encrypted or hashed formats. I

also participated in creating regular backup plans and ensured that a data recovery strategy was in place for testing and live environments. I began by understanding the existing database schema of the Elastyx platform and later contributed to designing new tables and refining relationships between them. This experience not only improved my knowledge of SQL and relational design but also gave us a deep understanding of how to manage data securely and efficiently in a production-grade system.

2.5 Supporting Tools and Development Environment

Alongside the core technologies of React.js, Flask, and MySQL, I used several supporting tools that enhanced my workflow, productivity, and collaboration throughout the internship. These tools were critical for organizing tasks, writing clean and testable code, and deploying the application across environments. my development process was guided by industry best practices, which gave us hands-on exposure to the full software development lifecycle from planning to release.

For version control, I used Git and GitHub to manage my source code. Each task was handled through a separate feature branch, and changes was committed regularly with descriptive messages. Pull requests was created for peer review, ensuring code quality and team accountability. GitHub also helped track issues, assign tasks, and maintain a changelog of all feature updates, bug fixes, and improvements during the internship period. This version-controlled approach helped us stay coordinated and avoid conflicts during merges. To test and debug backend APIs, I relied heavily on Postman. It allowed us to simulate different types of requests (GET, POST, PUT, DELETE) and inspect responses in real-time. This tool was especially helpful in identifying API behavior, validating response data, and checking how endpoints handled both valid and invalid inputs. Postman collections were shared within the team to standardize backend testing before integration with the frontend.

For deployment, I used Vercel to host my React frontend and Render to deploy my Flask backend. These platforms simplified the process of moving from development to production by offering automated builds and continuous deployment pipelines. I configured environment variables, managed route mappings, and monitored server logs directly from the dashboard. In addition, browser DevTools and console logs was used extensively to debug layout shifts, network delays, and runtime errors, ensuring the application was smooth and error- free for end users. The development environment at Masarrati was carefully structured to reflect real-world industry practices, and the supporting tools we used played a crucial role in streamlining daily workflows. From GitHub’s version control features to Postman's API testing

capabilities, every tool was selected for its efficiency and collaboration support. I particularly benefited from using Vercel and Render for deployment, which simplified the transition from development to production through automated build pipelines. These platforms helped me learn how to manage deployment environments, configure variables, and troubleshoot runtime errors using built-in logs. Additionally, I relied heavily on browser DevTools and debugging consoles to inspect elements, trace network calls, and resolve UI inconsistencies. This diverse toolkit not only enhanced my technical efficiency but also gave me practical experience with tools commonly used by professional developers in full-stack projects.

2.5.1 Version Control and API Testing I used Git and GitHub for collaborative development, allowing us to create branches, review code, and manage pull requests efficiently. GitHub Issues and Projects helped us organize tasks and track sprint progress. For backend validation, I tested all API endpoints using Postman, simulating various request methods to verify expected responses. During my internship, adopting professional development workflows was just as important as writing functional code. One of the core practices I followed was effective version control using Git and GitHub. For every new feature or fix, I created a separate branch to work independently without disrupting the main codebase. I committed changes regularly with clear, descriptive messages, making the code history easy to trace and review. Once a feature was ready, I created pull requests, allowing mentors and teammates to provide feedback through code reviews. This collaborative approach ensured that we maintained a clean, organized repository while catching bugs and inconsistencies early in the development cycle. Using GitHub Issues and Project boards, we were also able to track tasks, assign ownership, and keep sprints on schedule.

2.5.2 Deployment and Debugging Tools The React frontend was deployed on Vercel, while the Flask backend ran on Render. These platforms automated build processes and allowed us to push production-ready code with minimal manual effort. From GitHub’s version control features to Postman's API testing capabilities, every tool was selected for its efficiency and collaboration support. I particularly benefited from using Vercel and Render for deployment, which simplified the transition from development to production through automated build pipelines. Deployment was one of the most exciting and educational parts of my internship experience, as it allowed me to see my code come to life in a live environment. I used Vercel to deploy the React.js frontend, which provided a smooth and automated process for pushing

updates. By connecting my GitHub repository directly to Vercel, every time I merged a new feature or bug fix into the main branch, it triggered an automatic build and deployment. This continuous deployment workflow ensured faster iteration and immediate feedback on how changes appeared in production. For the Flask backend, I used Render, a cloud-based platform that enabled me to deploy Python APIs with minimal configuration. Render's deployment logs, error reports, and health checks were incredibly helpful for monitoring backend stability and performance.

2.6 Integration Workflow and System Architecture

The integration of frontend, backend, and database modules was a critical component of the Elastyx platform's overall functionality. I structured the integration workflow to ensure seamless data flow between user actions and backend processes. By coordinating frontend interactions with Flask-based APIs and validating database responses via MySQL, I established a consistent, responsive, and interactive user experience. This tightly coupled workflow enabled real-time feedback during operations such as adding to cart, checking out, and loading personalized recommendations.

A layered architecture was followed, separating concerns between interface, logic, and data handling. React components sent asynchronous requests using Axios to Flask APIs, which processed business logic and communicated with the MySQL database. Each tier was designed to operate independently yet interact in harmony through well-defined endpoints. This architectural approach improved testability and debugging while allowing us to update or scale one layer without disrupting the others. Understanding and contributing to the system's integration workflow gave me valuable insight into how different layers of a full-stack application function together in harmony. The Elastyx platform followed a cleanly separated architectural pattern where the frontend, backend, and database layers interacted through well-defined APIs. React.js managed the user interface, sending asynchronous requests via Axios to Flask-based endpoints, which in turn communicated with MySQL for data operations. This modular approach ensured scalability and ease of maintenance, especially as new features were added during sprints. I also worked on managing application state and UI behavior based on real-time API responses, which required careful coordination between frontend logic and backend validations. This integration process not only reinforced my understanding of system architecture but also highlighted the importance of writing loosely coupled, clearly documented code for seamless interoperability.

I also paid close attention to state management and data validation during the integration phase. On the frontend, useState and useEffect was used to track user interactions and reflect server responses in real time. The backend returned standardized JSON structures, which made it easier for the frontend to parse and display content. This cycle of consistent request-response handling created a robust and modular full-stack system that could scale with additional features or data without compromising stability.

2.6.1 Frontend-Backend Communication The frontend, built with React.js, communicated with the backend Flask APIs using Axios, a promise-based HTTP client. Each time a user performed an action—such as adding an item to the cart or viewing product details—a corresponding HTTP request was triggered. These requests were processed on the backend and responded to in JSON format, which the frontend consumed to update the UI dynamically. This asynchronous interaction enabled real-time updates and minimized page reloads, improving the overall user experience.

2.6.2 API Routing and Middleware The Flask backend was organized using Blueprints, which separated different parts of the application into logical modules such as /auth, /cart, /wishlist, and /checkout. This made the codebase cleaner and more scalable. Middleware components were used to handle tasks such as authentication checks, data validation, and request logging before passing control to the actual route handlers. Proper API routing ensured that data was processed correctly and securely, maintaining the integrity of the overall system.

2.6.3 Database Connectivity The backend APIs interacted with a MySQL database for data storage and retrieval. Using parameterized SQL queries, the system handled actions such as fetching product listings, storing user preferences, and managing order data. Relationships between tables were maintained using foreign keys, and JOIN operations were used to retrieve combined data efficiently. This structure allowed for accurate, real-time access to user-specific information, which was essential for features like personalized recommendations and order summaries.

Chapter 3

Internship Responsibilities

During my internship at Masarrati Pvt. Ltd., I was assigned a wide range of responsibilities that allowed us to engage in real-world application engineering. As part of the Elastyx platform team, I was expected to contribute to every major phase of product development, including requirement gathering, interface design, API integration, backend support, and deployment. These responsibilities was not isolated tasks, but interconnected roles that gives end-to-end ownership of modules. My work was supervised by experienced mentors and carried out in a highly collaborative environment that mimicked professional industry settings. I also participated in daily stand-up meetings, sprint planning sessions, and weekly reviews, which allowed us to align my development efforts with business goals and user expectations. This chapter describes the core responsibilities I handled and the outcomes they produced.

3.1 User Interface Development

Our involvement in frontend development was one of the most fulfilling parts of the internship. I used React.js to build dynamic, responsive, and reusable components that shaped the user interface of the Elastyx platform. These included product cards, category filters, wishlist buttons, cart previews, and checkout panels. Each component was designed with the goal of enhancing user experience, minimizing visual clutter, and making navigation seamless for students, parents, and mentors alike. I ensured that every design followed Masarrati's branding guidelines and accessibility standards.

In collaboration with the UI/UX team, I translated Figma wireframes and mockups into pixel-perfect frontend layouts. I implemented stateful interactions using React hooks such as useState and useEffect, allowing us to capture real-time changes like product selections or cart updates. Furthermore, I incorporated conditional rendering and dynamic routing using React Router to offer an intuitive flow through the platform's multiple screens. The goal was to make each interaction predictable and satisfying for the user. Handling full-stack development responsibilities pushed me beyond technical comfort zones and into a space where I learned to think critically, communicate clearly, and deliver reliably. From managing time-sensitive deliverables to discussing feature behavior with cross-functional teams, I was constantly learning to adapt and respond like a professional. The tasks not only helped me gain fluency in tools and frameworks but also sharpened my instincts on collaboration,

design clarity, and project lifecycle awareness—making the experience holistic and career.

I also made extensive use of CSS Flexbox and Grid for layout alignment and responsiveness across devices. Each element was designed to adapt gracefully to both desktop and mobile screen sizes. I added animations, hover effects, and button feedback to make interactions more engaging. As part of accessibility compliance, I maintained color contrast, logical focus order, and semantic HTML structures. my goal was not just to make something that worked—but something that felt natural, professional, and user- centered.

3.2 API Integration and Backend Communication

A critical responsibility was integrating the frontend interface with Flask-based backend APIs to ensure data-driven functionality. Using Axios, I established connections that handled operations like user login, cart manipulation, product browsing, and order submission. Each request sent from the frontend triggered corresponding routes on the server and awaited JSON responses, which was then used to update the UI in real-time. I implemented loaders and message prompts to ensure users was informed during data transfers.

The integration process required close attention to API response structures, error messages, and performance. I used try-catch blocks to gracefully handle exceptions and displayed meaningful messages to the user when an error occurred. Each Axios request included validation logic to ensure that no incomplete or incorrect data was submitted. This attention to detail allowed us to minimize crashes and prevent unexpected bugs on the user side. One of my key responsibilities during the internship was ensuring smooth and efficient API integration between the frontend React components and the Flask-powered backend. This process involved setting up real-time communication using Axios, a promise-based HTTP client that enabled the React application to send and receive data from various backend endpoints. For each user interaction—such as logging in, adding items to the cart, or checking out—an appropriate API call was triggered, and the JSON response was then used to update the frontend state. I designed this communication to be both responsive and reliable, ensuring that users received instant feedback on their actions, such as confirmation messages or form validation alerts.

I also tested each integration using browser developer tools and Postman. By inspecting payloads and HTTP response codes, I confirmed that requests were reaching the server correctly and that responses were accurate and secure. Once integration was successful, I tested real-time synchronization—verifying that frontend states updated

instantly when a user performed an action like removing a product from the cart. To make this integration robust, I implemented error handling mechanisms using try-catch blocks in both the frontend and backend. Through this, I learned the importance of seamless frontend-backend connectivity in delivering a consistent user experience.

3.3 Database Interaction and Data Flow

My work on the backend database involved designing and modifying MySQL schemas that supported key platform functionalities. I created and managed tables for users, products, carts, wishlists, orders, and AI behavior logs. Every schema was designed using relational principles to ensure that data remained normalized and could be queried efficiently. I was responsible for writing SQL queries to retrieve, insert, update, and delete data as needed. These queries were embedded in Flask routes and executed based on frontend requests. I learned to use JOIN operations to fetch related information from multiple tables for instance, retrieving both product details and corresponding user interactions for the wishlist view. my queries were performance-tested using indexing and filtered clauses to reduce execution time.

A critical part of my internship experience involved working directly with the MySQL database to manage data interactions that powered the Elastyx platform. I was responsible for designing and updating database schemas to efficiently handle various types of data such as user information, product details, cart contents, wishlist entries, and order history. Using relational principles, I structured the tables to maintain referential integrity, applying primary and foreign keys where necessary to link related data. For example, one-to-many relationships were used between users and orders, while many-to-many relationships were handled through bridge tables—particularly for features like wishlists. This allowed the platform to fetch and store information accurately without redundancy.

To support Masarrati's AI recommendation engine, I logged user activities such as viewed items, clicked kits, and purchased products. This behavioral data was stored in structured format and used by backend logic to recommend relevant kits to users. I also implemented backup procedures to ensure data safety and contributed to designing access control logic, so that sensitive information remained protected at all times. All interactions were securely managed using parameterized queries to protect against SQL injection. In addition to designing schemas, I wrote and optimized SQL queries to support real-time data flow between the backend and the frontend. These queries included SELECT operations for product listings, INSERT statements for user actions like adding to cart or placing an order,

and UPDATE queries for profile edits or payment confirmations. This real-world exposure to data management taught me the importance of clean schema design, data validation, and performance optimization in building scalable, user-centric applications.

Through these responsibilities, I gained confidence in database design, optimization, and security. One of the most important aspects of my internship involved working hands-on with the MySQL database to manage structured data for the Elastyx platform. I began by analyzing the existing schema and gradually contributed to designing new tables, refining relationships, and ensuring efficient data flow between backend services and the database. Core tables included entities like Users, Products, Orders, Wishlists, and Cart Items—each designed using best practices in relational database modeling. I paid close attention to maintaining data integrity by applying primary keys, foreign key constraints, and normalization principles to avoid redundancy and improve query efficiency.

3.4 Testing and Debugging

Testing played a vital role in ensuring the quality and reliability of everything I developed. I wrote unit tests for frontend components to validate that inputs, outputs, and component behavior matched expectations. On the backend, I used Postman to simulate real-world user activity by sending requests with valid and invalid data, helping us spot issues in authentication, product handling, and cart manipulation. This proactive approach helped us catch issues early and improve software robustness.

I also used browser-based DevTools to test interface responsiveness, console warnings, and layout bugs. For example, I checked how the product grid aligned across devices and tested mobile views for scrollable areas, button tap targets, and form inputs. When bugs were found, I documented the error and the fix, ensuring transparency and continuous improvement. Many UI glitches was resolved through conditionally rendered states and improved CSS hierarchy. Testing and debugging played a crucial role in ensuring the reliability, stability, and overall quality of the Elastyx platform during my internship. I followed a proactive approach to testing by validating each feature during and after development rather than waiting until the end. On the frontend, I tested React.js components manually using browser DevTools to check layout behavior, responsiveness, and real-time interactions like button clicks and form inputs. I also monitored the console for JavaScript errors and warnings, ensuring smooth navigation and interaction across devices and browsers. Every element—from product cards to the checkout page—was verified for responsiveness and proper data binding before integration with live APIs.

As part of my testing responsibilities, I also participated in manual QA reviews before a feature went live. I reviewed edge cases, verified how components responded to rapid user actions, and tested form validations. When bugs were discovered—such as missing data, incorrect responses, or layout glitches—I documented the issue, debugged the root cause, and pushed clean fixes through version-controlled commits. This process taught me the importance of continuous testing as a core part of the development cycle, not just for catching bugs but for improving the overall user experience and ensuring the platform remained robust as new features were introduced. These experiences taught us the value of building for real-world usage—not just technically functional, but also user-resilient. Testing wasn't something I did once at the end—it was an ongoing part of my development cycle.

3.5 Documentation and Team Collaboration

Documenting my work was critical to maintaining transparency and helping others on the team understand what had been built. I created README files for each component or module, detailing its purpose, logic, props/state, and dependencies. I also documented API endpoints using Swagger-like format, describing request methods, headers, expected inputs, and response formats. This was stored in the shared GitHub repository for easy team access and handover.

Beyond technical documentation, I regularly participated in team meetings, where I presented completed tasks, shared updates, and received feedback from mentors and peers. This collaborative environment helped me to understand team dynamics and communication workflows. I was encouraged to explain my logic during code reviews, which improved my confidence and helped others learn from my approaches. Clear and consistent documentation was essential throughout my internship to ensure that the work I completed could be easily understood, reviewed, and extended by other team members. For every major feature or module I developed, I created detailed README files that outlined the component's purpose, logic, props/state management, and dependencies. In the backend, I documented all Flask API endpoints using a Swagger-like format, including request types, expected parameters, and sample responses. This documentation was stored in the team's shared GitHub repository, which made it easier for future developers or interns to get up to speed. Additionally, I wrote guides for setting up the development environment, running the app locally, and understanding the database schema—ensuring a smooth onboarding process for anyone new to the project.

I also contributed to the internal knowledge base by adding guides for setting up the development environment, deployment steps, and database schema overviews. On the collaboration side, I was actively involved in daily team routines such as stand-up meetings, sprint planning sessions, and code reviews. These meetings gave me the opportunity to present my progress, discuss blockers, and receive timely feedback from mentors and teammates. We used tools like Slack and Trello for communication and task management, which helped keep our sprint goals clear and our work organized. I also participated in peer reviews, where I both received and gave constructive feedback on code quality, readability, and functionality. These collaborative experiences taught me the value of teamwork, open communication, and collective problem-solving. More importantly, I developed a professional mindset—one where I wasn't just building code, but contributing to a shared vision with clarity, accountability, and respect for others' work. When new interns or team members joined, my documentation helped them onboard quickly. Through this responsibility, I not only learned how to code effectively.

3.6 Overview of Development Approach

During my internship at Masarrati Pvt. Ltd., my development approach was structured, modular, and designed to simulate professional software engineering environments. I divided the entire project into thematically grouped modules and followed agile methodologies to stay on track with sprint planning and weekly goals. Each new feature was discussed, wireframed, built, tested, and then deployed after peer review.

I started by setting up a scalable frontend using React.js. This gave us the foundation to create and integrate reusable UI components that supported responsiveness, maintainability, and cross-browser compatibility. The modular nature of React helped us create isolated features like Product Cards, Wishlist Buttons, and Cart Panels that could be tested and reused independently. Following a structured workflow throughout the development process allowed me to approach problems methodically rather than reactively. The visualization and modular planning taught me to break down complex requirements into smaller, manageable units—an essential practice in any scalable project. It was during these moments of integration and testing that I began appreciating the interconnectedness of features and the discipline required to maintain code hygiene, user focus, and system reliability. This project transformed my theoretical knowledge into functional expertise, solidifying my foundations in full-stack development.

Our workflow also emphasized the importance of clean code, state management, and performance optimization. I used GitHub for version control and made sure each feature branch was merged only after review. My development approach throughout the internship at Masarrati Pvt. Ltd. was structured, agile, and aligned with real-world industry practices. I followed a modular strategy, breaking down the platform into smaller, manageable components—such as product listings, wishlists, cart handling, and checkout flows. Each feature was treated as a self-contained module with its own responsibilities and dependencies. This allowed me to work in parallel with other team members without causing conflicts in the codebase. I started every new task by reviewing requirements, sketching out wireframes if needed, and planning the logic and flow before diving into implementation. This organized approach ensured that each piece of functionality was scalable, testable, and easy to maintain. This structured approach helped us collaborate smoothly and gave us hands-on experience in working like a real development team.

- **Modular Feature Planning** Each feature was planned as an isolated module to allow independent development and testing.I created a wireframe for every feature, discussed its dependencies, and integrated it step-by-step.This modularity improved code clarity, team coordination, and reusability across components. Throughout my internship, I adopted a modular feature planning approach to manage complexity and ensure scalability in the Elastyx platform.

- **Agile Sprint Execution** I followed weekly sprints, defining clear goals, assigning tasks, and reviewing progress in team syncs.Stand-up meetings helped identify blockers early and made collaboration efficient. During my internship, I closely followed an Agile sprint-based development model, which played a crucial role in organizing tasks, setting clear goals, and maintaining steady progress. Each week began with a sprint planning session, where tasks were broken down into achievable milestones. These tasks were prioritized based on feature complexity, user impact, and overall project deadlines.

3.7 Workflow Visualization

Visualizing the development workflow helped us stay organized and aligned during the internship. I designed a comprehensive flowchart that represented the journey from environment setup to feature integration and final testing. The workflow was arranged in a modular sequence—beginning with project initialization, UI design, API integration, and

ending with user testing and optimization. Each stage had checkpoints that allowed us to validate output before moving ahead. This layered approach ensured that every task was built upon a stable foundation and aligned with my mentor's feedback.

Visual diagrams also helped in team discussions, allowing all contributors to understand their roles clearly. By using a workflow chart, I could map task progress, plan feature delivery, and troubleshoot delays. It acted as a real-time roadmap and guided us from frontend architecture to dynamic functionality and performance tuning. The visual workflow diagram created during the internship served as a blueprint for aligning technical tasks with project goals. It clearly mapped out each phase—from project setup and UI planning to backend integration and final testing—making it easier to track dependencies and manage priorities. By visualizing the flow of data and feature development, I was able to identify integration points early, avoid redundant work, and keep the team aligned on shared deliverables.

This clarity was especially helpful during sprint reviews, as it offered a tangible reference for progress tracking and issue resolution. The structured layout also reinforced the importance of a layered, modular approach, where each block built upon the previous one to ensure system stability and continuous delivery. Overall, this exercise not only boosted team coordination but also taught me how essential process visualization is for effective software development. The development flowchart outlined all stages from setup deployment.I visualized interactions between UI, APIs, and the backend for smooth sequencing.This planning made collaboration easier and improved sprint tracking. Creating a detailed flowchart early in the development cycle helped us visualize the logical flow of the Elastyx platform and preemptively address potential integration challenges. The planning process began by outlining major components such as user login, product browsing, cart operations, and order placement, then breaking them down into functional stages with clearly defined transitions.

This step-by-step mapping enabled me to understand how data moved between the frontend and backend, and where user actions triggered specific API calls or database queries. It also supported better team collaboration, as each developer could see how their module fit into the overall architecture. The React components and Flask APIs. Careful sequencing minimized redundant work and allowed parallel development when possible, significantly improving team efficiency. This structured approach gave me firsthand experience in managing task dependencies—an essential skill for building scalable, production-ready applications.

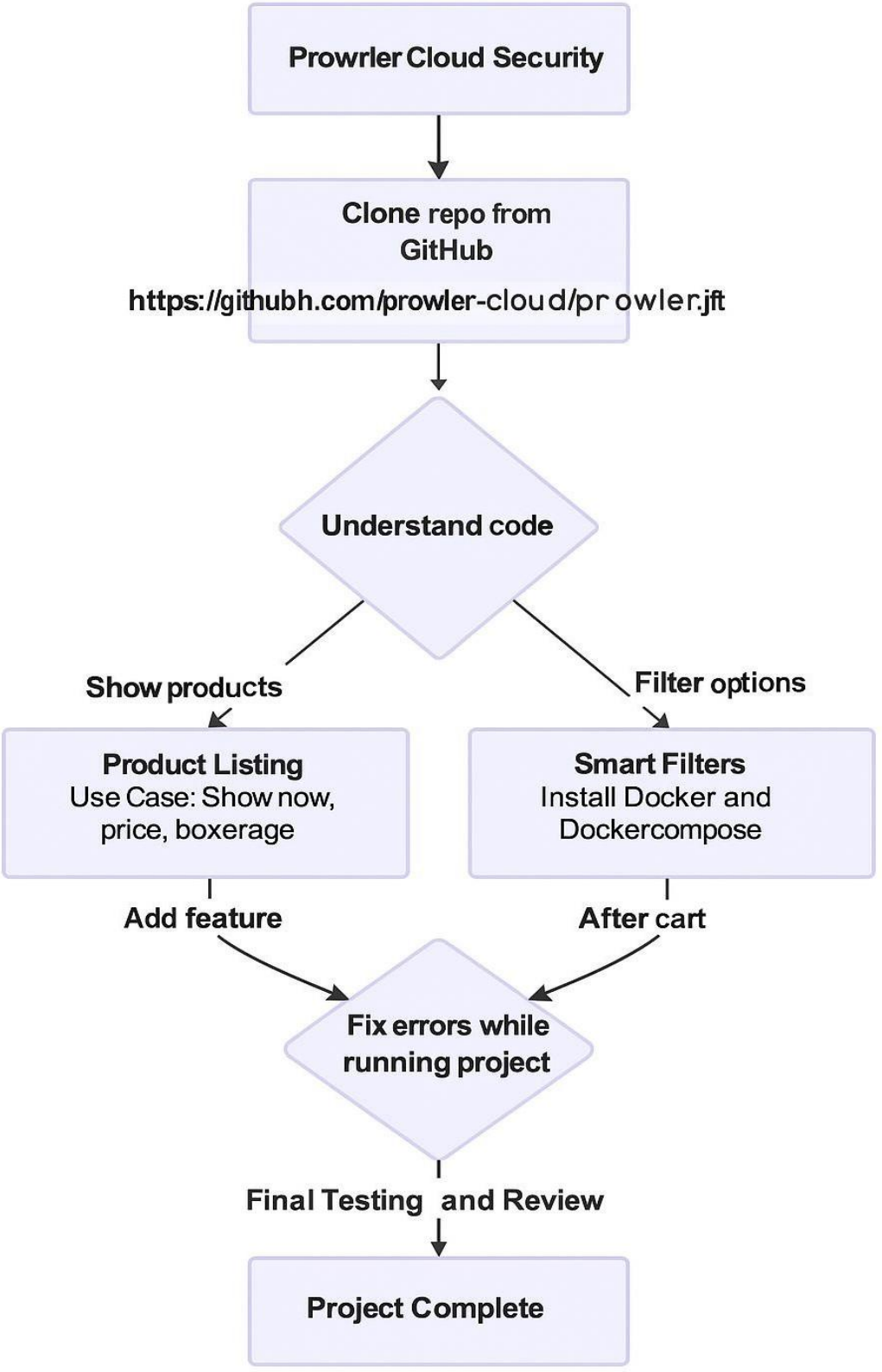

Fig 3.1 Project Workflow

As shown in the figure 3.1, the Prowler Cloud Security Project Workflow outlines the structured steps followed during the project development lifecycle. The process begins with cloning the Prowler repository from GitHub, allowing access to the core codebase. The next critical phase involves understanding the code, which serves as the foundation for all subsequent development. This understanding is split into two main tracks: implementing product listing features and configuring smart filters. On the product side, the focus was on creating a UI component that could display product details such as name, price, and box coverage. Simultaneously, the backend setup required the installation and configuration of Docker and Docker Compose to manage environment containers effectively.

3.8 Step-by-Step Implementation

The feature development process for Elastyx followed a carefully designed step-by-step structure that allowed us to build with clarity and purpose. I began by setting up a scalable React.js environment, including routing, folder architecture, and component templates. This setup created the foundation for my feature modules, making the system flexible and easier to debug. Our next step involved implementing key features individually—starting with product listing, followed by wishlist, cart, checkout, and eventually AI-based recommendations. Each feature was tested in isolation before being linked to the overall application flow. I maintained UI consistency using shared components and ensured smooth user interactions across every step.

Once core features were integrated, I shifted my focus to testing and optimization. I used Postman to test APIs, browser dev tools to inspect data flow, and GitHub for version tracking. These efforts ensured that each module worked reliably on its own and as part of the larger ecosystem. my structured implementation improved both productivity and system quality.

- **UI Foundations & Component Setup** I started by creating shared components like Navbar, Footer, and Product Cards. React Router was configured for page navigation, and layout grids was tested for responsiveness. This base allowed us to maintain design consistency and modular development. By integrating responsive design principles, the checkout flow functioned smoothly across devices, making it accessible even on low-resolution screens. This component exemplified how thoughtful design and technical precision come together to support a reliable and user-friendly e-commerce experience in an educational context.

- **Checkout Flow** Checkout collected user info with real-time form validation. On form submission, data was sent to the backend's order API. Success screens confirmed order placement with summary display. The checkout flow played a critical role in ensuring a seamless end-to-end user journey on the Elastyx platform. This phase was carefully crafted to not only collect user information securely but also provide real-time feedback during form validation and order submission. , I implemented smart personalization through an AI- powered recommendation engine and introduced interface enhancements like lazy image loading, modals, and A/B testing for optimization. As part of my work on this module, I ensured that the UI was intuitive—offering clear input fields, dropdowns, and summary displays—while the backend was optimized for secure data handling using Flask.

- **Feature-wise Integration Process** Each feature (e.g., Wishlist, Cart) was integrated after backend APIs was confirmed via Postman. Axios was used for API calls, and state updates was handled using React hooks. Testing was done after each feature to ensure stability before merging.

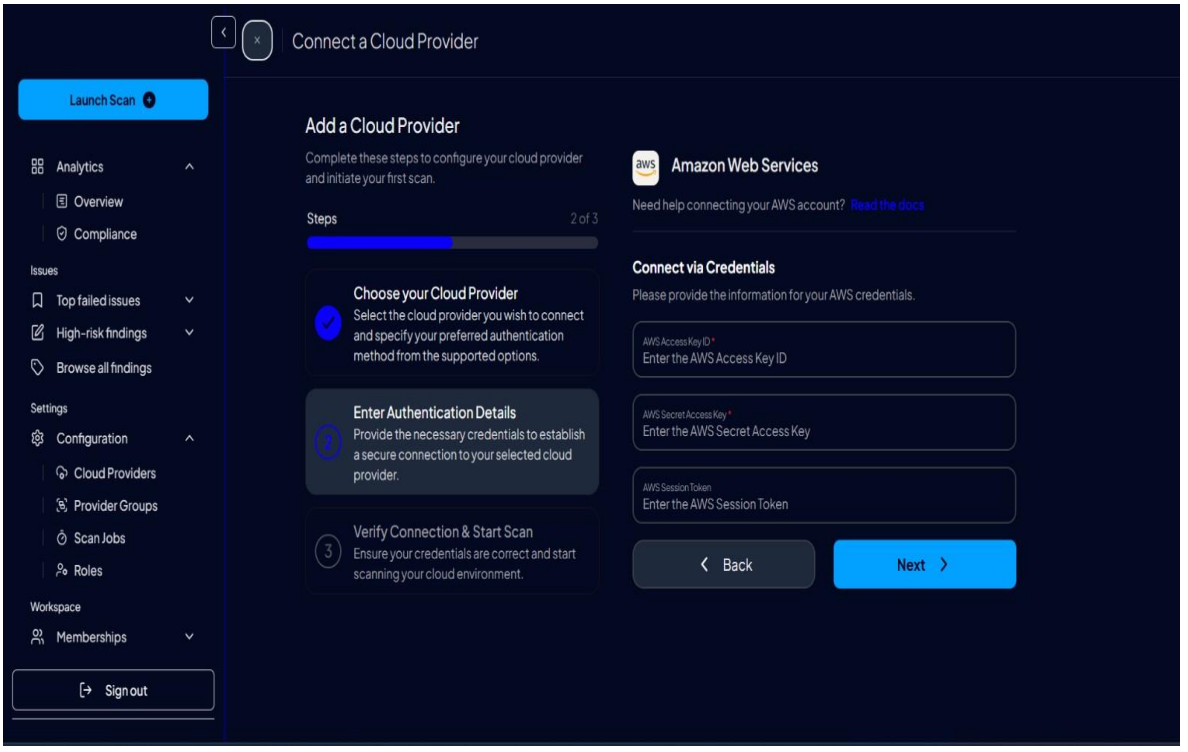

Fig 3.2 Connect a Cloud Provider

- **Testing & Debugging Phase** I tested full user journeys like “browse → add cart → checkout.” Console logs, network tabs, and form validators was used to identify and fix bugs. Clean commits with descriptive messages were pushed using Git and reviewed regularly.

3.9 Website Overview

The final outcome of my internship was a fully responsive, dynamic, and feature- rich frontend interface for the Elastyx learning platform. Built using React.js and connected with Flask APIs, the website allowed students to browse AI and Cloud kits, manage wishlists, add items to cart, and complete purchases. Each screen was built using reusable components, ensuring code maintainability and design consistency across devices.

I followed a modular approach to construct individual pages like Dashboard, Product Listing, Wishlist, Cart, and Checkout. Each page was integrated with backend APIs for live data and was tested thoroughly for functionality and user experience. The official website of Masarrati Pvt. Ltd. serves as a digital gateway to the company's broad portfolio of services, technological expertise, and innovation-driven culture. Designed with a modern and intuitive user interface, the website effectively communicates Masarrati's mission to deliver comprehensive digital transformation solutions. I handled routing, input validation, and data formatting using JSON. Writing modular code with Flask helped us manage routes clearly and debug issues efficiently. This hands-on practice gave us clarity on how application servers communicate with clients and databases.

The homepage features a bold tagline, “Innovating The Future, Together,” and showcases key achievements such as 15+ years of experience, 75+ happy clients, and 120+ successful projects, establishing immediate credibility. In addition to the core features, I implemented smart personalization through an AI- powered recommendation engine and introduced interface enhancements like lazy image loading, modals, and A/B testing for optimization. Together, these efforts contributed to a robust frontend that reflects both technical quality and user-focused design.

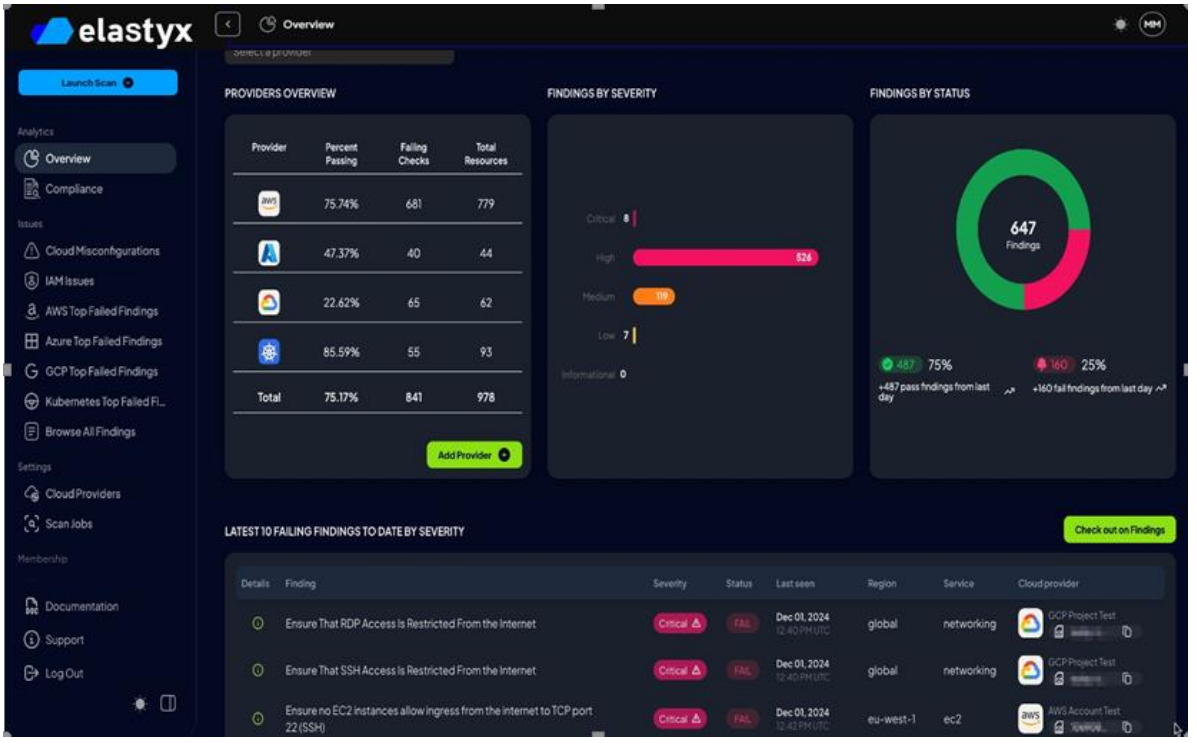

Fig 3.3 Dashboard Overview

Fig 3.4 Elastyx Home Page

As shown in the figure 3.3, the Elastyx Providers Overview panel displays a centralized summary of connected cloud environments, allowing users to monitor security compliance across multiple platforms such as AWS, Azure, GCP, Kubernetes, and others. The table provides critical metrics, including Percent Passing, Failing Checks, and Total Resources, offering a clear snapshot of system health and audit readiness. This view enables quick identification of which providers need attention and ensures consistent security posture across cloud assets.

As shown in the figure 3.4, the Elastyx Compliance Dashboard provides a comprehensive overview of multi-cloud security posture, offering real-time insights into provider-specific vulnerabilities, severity distribution, and overall system health. The Providers Overview panel displays key metrics such as Percent Passing, Failing Checks, and Total Resources for platforms like AWS, Azure, GCP, and Kubernetes. This enables administrators to quickly identify which environments need immediate attention. One of the key features of the website is the prediction form, which collects input such as age, ejection fraction, serum creatinine levels, blood pressure status, and other critical medical indicators. Once the user submits this form, the backend processes the data through a trained machine learning model and displays the prediction result along with a confidence score.

As part of the Heart Failure Prediction project, a user-friendly web application was developed to make the model accessible to both healthcare professionals and end users. The goal was to create an intuitive interface where users can enter patient health details and receive immediate predictions about their risk of heart failure. The website was built using Flask, a lightweight Python web framework that is ideal for developing and integrating machine learning models into web interfaces. The homepage offers a simple introduction to the system and guides users toward the prediction tool.

- **Home Page** – Briefly introduces the system’s purpose and value in early heart failure detection.

- **Services Page** – Lists the benefits of the tool such as early diagnosis, accessibility, low cost, and reduced human error.

Chapter 4

Key Learnings and Skills Developed

The internship at Masarrati Pvt. Ltd. offered us a comprehensive learning experience that combined technical growth, professional exposure, and collaborative development. Working on the Elastyx platform helped us gain hands-on knowledge of full-stack application development and understand the workflow of a live project in a real industry setting. From writing modular code to participating in sprint reviews and team discussions, every aspect of my role contributed to my growth as aspiring engineers. This chapter outlines the key technical and soft skills I developed during the course of my internship.

4.1 Technical Skills

Throughout the internship, I was immersed in a full-stack development environment that exposed us to a wide range of tools and technologies. I learned to write functional, scalable, and maintainable code that aligned with Masarrati's platform goals. my involvement in real-world problem-solving helped us bridge the gap between academic concepts and industry practices. Each tool and technology I used contributed to the development of strong technical foundations.

On the frontend, I sharpened my skills in React.js by creating reusable components and building user interfaces that responded to user interactions in real time. Working on modules like the wishlist, cart, and product grid gave us practical knowledge of component architecture. I explored advanced concepts like routing, state management, and dynamic rendering. This helped us build interfaces that was responsive and user-friendly across all devices. The backend gave us exposure to Python and Flask, which I used to build RESTful APIs and connect the frontend to the database. I handled routing, input validation, and data formatting using JSON. Writing modular code with Flask helped us manage routes clearly and debug issues efficiently. This hands-on practice gave us clarity on how application servers communicate with clients and databases.

- **React.js** I used React.js to develop dynamic and reusable UI components, making the platform responsive and modular. It helped us learn real-time state handling, component trees, and routing logic.

- **Flask** I built APIs using Flask and structured them into modular routes using blueprints. Flask enabled backend operations like cart management and user authentication with clean data handling.

- **MySQL** MySQL was used for storing user, product, and order data with relational table design. I wrote JOIN queries and normalized schemas for optimal performance and data integrity.

- **Postman** Postman helped us test API requests and responses before frontend integration. It allowed us to simulate edge cases and validate endpoint behavior effectively.

- **Git&GitHub** I used Git for version control and GitHub for code reviews, pull requests, and branch management. This gave us exposure to team collaboration in a professional environment.

4.2 Soft Skill

In addition to technical expertise, my internship at Masarrati helped us grow significantly in terms of personal and professional behavior. Throughout my internship at Masarrati Pvt. Ltd., I not only enhanced my technical expertise but also developed a range of soft skills that are essential for working effectively in a professional environment. One of the most important skills I improved was communication—both verbal and written. Regular participation in stand-up meetings, sprint planning sessions, and peer reviews helped me learn how to express technical ideas clearly, ask relevant questions, and provide constructive feedback. I also became more comfortable with drafting technical documentation, writing user-facing instructions, and updating team members on task progress. I learned how to communicate within a team, manage my time, respond to feedback, and work effectively under deadlines.

Working in a collaborative environment taught us how to express my thoughts clearly and listen actively. I participated in stand-up meetings, sprint reviews, and code walkthroughs where I explained my work and discussed progress. These interactions improved my communication and boosted my confidence in professional discussions. Time management was another important lesson. Juggling daily tasks, meeting sprint deadlines, and resolving bugs under pressure helped us stay organized and focused. We learned to prioritize tasks and split larger assignments into manageable subtasks, which enhanced my productivity and reduced last-minute stress.

Receiving feedback and adapting to it was a key part of the learning process. Whether it came from code reviews, design changes, or QA testing, I learned how to view constructive criticism positively and implement suggestions quickly. This helped us become more adaptable and open-minded as developers. Beyond technical expertise, my time at Masarrati significantly refined my interpersonal and professional behaviors. I learned how to communicate effectively within a diverse team, present ideas clearly, and actively listen during discussions—skills that proved essential in our daily stand-ups and sprint reviews. The supportive environment encouraged constructive feedback and collaborative decision-making, which helped me grow not just as a developer but as a team player. Over time, I developed better emotional intelligence, learning to remain calm under pressure, meet deadlines responsibly, and contribute positively to group dynamics. These soft skills, often underrated in technical roles, are now an integral part of how I approach tasks, handle challenges, and contribute to any professional environment.

Above all, I developed a sense of ownership and accountability for the modules I worked on. I was not just writing code; I was contributing to a real product used by students and educators. This sense of responsibility improved my problem-solving mindset and taught us to think beyond just the technical side.

- **Communication Skills** I learned to clearly articulate my ideas during meetings and explain technical challenges. my daily interactions improved my clarity, teamwork, and listening abilities.

- **Time Management** Managing tasks across sprints taught us how to estimate workloads and deliver features on time. I developed a habit of organizing tasks using to-do lists and collaborative tools.

- **Adaptability & Feedback** I regularly received feedback from mentors and teammates and used it to improve my code and designs. This made us flexible in my approach and more efficient in execution.

- **Professionalism & Accountability** I learned to take responsibility for my deliverables and to approach each task with seriousness and integrity. This helped us gain trust and perform well under real deadlines.

During my internship, I had the opportunity to explore a variety of industry- standard tools and platforms that supported my development, testing, collaboration, and deployment efforts. These tools significantly enhanced my technical workflow and productivity. I became

well-versed in Git and GitHub, which helped us collaborate on code, manage branches, and resolve merge conflicts during feature development. Working with version control in a team-based setup taught us how to track changes, create pull requests, and maintain a clean codebase. This hands-on usage prepared us for real-world collaborative coding environments. Postman played a critical role in API development and debugging. It allowed us to simulate different request types, test endpoints, and analyze backend responses before integrating them with the frontend. By organizing collections and sharing them with my team, I ensured consistency and transparency in API testing and documentation.

For deployment and project hosting, I used Render to host the Flask backend and Vercel to deploy the React frontend. These platforms made it easy to push updates and visualize builds in real-time. I learned how to configure environment variables, handle API routing, and debug runtime logs directly from these platforms. Additionally, browser developer tools were essential for monitoring frontend performance, identifying layout issues, and analyzing network requests. Console logs and network tabs helped us inspect real-time data flow and debug UI-related errors efficiently. Using these tools deepened my understanding of frontend behavior during development.

4.3 Real-World Development Practices

One of the most valuable aspects of my internship was gaining exposure to real- world software development workflows, team coordination, and best practices followed in the tech industry. During my internship at Masarrati Pvt. Ltd., I had the opportunity to experience and adopt real-world development practices that closely mirrored industry standards. One of the most valuable lessons I learned was the importance of writing clean, maintainable, and well-documented code. Whether I was building React components or writing Flask APIs, I followed best practices such as consistent naming conventions, modular file structures, and descriptive comments. This not only made my work easier to debug and scale but also ensured it was understandable for other developers reviewing or building on top of my code. Regular code reviews were a part of our workflow, where I received constructive feedback that helped refine my logic and improve overall code quality.

I participated in agile sprint cycles that included planning, daily stand-ups, development, testing, and sprint reviews. These weekly cycles helped us stay focused on specific goals and measure my progress. I learned how to break down features into smaller, achievable tasks and update myteam regularly on what I accomplished and where I was blocked. I was also introduced to the concept of code reviews and collaborative debugging. Every pull

request I raised was reviewed by my mentor, who suggested improvements to maintain code clarity, consistency, and performance. Through process, I learned about coding standards, patterns, and naming conventions that professional teams follow.

Testing was treated as an ongoing responsibility, not just something done before delivery. I tested APIs using Postman and interfaces using DevTools throughout the development phase. These practices helped us identify bugs early and deliver more reliable features. I also learned how to log bugs, track fixes, and validate changes before merging into the main codebase. Documentation, code commenting, and knowledge sharing was key expectations throughout the internship. I documented every API, schema, and major function so others could understand and maintain it. By participating in knowledge transfers and internal demos, I saw firsthand how communication supports long-term project health and onboarding of new team members.

4.4 Personal Growth & Reflections

Beyond technical knowledge, this internship shaped us personally-improving my mindset, discipline, and approach toward work, teamwork, and real-world challenges. Adapting to a professional work environment taught us to be accountable for my tasks and deadlines. Unlike college assignments, my code here directly impacted a product used by real users. This realization motivated us to double-check my work, improve quality, and take full responsibility for what I delivered.

The mentorship I received helped us view mistakes as learning opportunities rather than failures. Whenever I faced bugs, design rejections, or unclear tasks, my mentors encouraged us to analyze the root cause and come up with structured solutions. This built my confidence and made us more resilient to pressure. Looking back on my internship journey at Masarrati Pvt. Ltd., I can confidently say that it was one of the most transformative experiences of my academic and professional life. It gave me more than just technical knowledge—I gained a deep understanding of how real-world software development functions within a team, under deadlines, and in service of a real user base. Initially, I was nervous about handling complex tasks, especially when working with technologies like React, Flask, and MySQL in a live environment. But with consistent mentorship, collaboration, and hands-on learning, I gradually built both confidence and competence in full-stack development. Each challenge—from resolving a UI bug to debugging backend issues—taught me patience, persistence, and the importance of attention to detail.

I also gained emotional intelligence by interacting with a cross-functional team. Communicating ideas, accepting feedback, and giving input respectfully helped us mature as professionals. I learned how important it is to listen first, ask thoughtful questions, and acknowledge others' efforts in team projects. Lastly, this internship made us more self-aware about my career goals and skills. I discovered my strengths in development, UI design, and teamwork—and also understood where I still needed to improve. These lessons will guide us as I move forward in both academic and professional paths. The internship at Masarrati Pvt. Ltd. offered us a comprehensive learning experience that combined technical growth, professional exposure, and collaborative development. Working on the Elastyx platform helped us gain hands-on knowledge of full-stack application development and understand the workflow of a live project in a real industry setting. From writing modular code to participating in sprint reviews and team discussions, every aspect of my role contributed to my growth as aspiring engineers. This chapter outlines the key technical and soft skills I developed during the course of my internship.

4.5 Technical Skills

During my internship at Masarrati Pvt. Ltd., I gained substantial technical experience working on a live educational platform. I applied my academic knowledge in a professional environment and enhanced my understanding of web development technologies through real-time implementation. Regular stand-up meetings and sprint reviews provided a structured space for knowledge exchange, where everyone's input was respected regardless of experience level. From setting up the frontend to linking with backend APIs and managing databases, my learning was practical, structured, and continuous.

I became proficient with React.js for frontend development, where I built reusable UI components, managed application state, and implemented routing. Simultaneously, I worked with Flask to develop RESTful APIs that handled user data, cart management, and order processing. Using MySQL, I learned how to create, structure, and query relational databases while maintaining secure and optimized data access. During the course of my internship at Masarrati Pvt. Ltd., I significantly expanded my technical skill set by working on a wide range of real-world development tasks. On the frontend, I became proficient in React.js, using it to build modular, reusable components with dynamic rendering and responsive layouts. I learned to manage component state effectively, apply conditional rendering, and use hooks to handle side effects. I also gained hands-on experience with CSS Flexbox, Grid systems, and media queries, which allowed me to create fully responsive user interfaces that adapted smoothly

across screen sizes and devices.

Additionally, I explored version control through Git and GitHub, tested my APIs using Postman, and deployed features using platforms like Vercel and Render. On the backend, I worked extensively with Flask, developing RESTful APIs and integrating them with the frontend using Axios. I managed database interactions using MySQL, where I designed efficient schemas, wrote optimized SQL queries, and implemented proper indexing to improve performance. My understanding of API testing deepened through tools like Postman, where I tested endpoints for reliability and security. I also became comfortable using Git and GitHub for version control, including branching, merging, and collaborative pull requests. Additionally, I gained exposure to deployment platforms like Vercel (for the frontend) and Render (for the backend), which gave me end-to-end insight into how modern web applications are built, tested, and delivered to users. These tools introduced us to professional workflows such as staging environments, build management, and production testing. my technical foundation became stronger with each module I implemented and each challenge I solved collaboratively.

4.5.1 Frontend Development using React.js I learned to build dynamic interfaces using React.js with reusable components.State and event handling helped us manage user actions across pages. One of the most impactful areas of my internship was working on the frontend development of the Elastyx platform using React.js. I learned how to build scalable, modular components that formed the core of the user interface, including sections like the product catalog, wishlist, shopping cart, and checkout flow. React's component-based architecture made it easy to organize the UI into reusable blocks, which improved development speed and code clarity. I used props and state management to control component behavior, making the interface responsive to user actions in real-time. This hands-on experience gave me a deeper understanding of how modern web applications are structured and how user interactivity is handled efficiently using frameworks like React. Routing and conditionalenabled modular page navigation.

4.5.2 Backend Development using Flask.I created API routes to handle orders, carts, and user sessions using Flask. As part of my backend development responsibilities during the internship, I worked extensively with Flask, a lightweight yet powerful Python web framework. I was involved in building RESTful APIs that handled core functions such as product retrieval, cart operations, wishlist management, and order processing. I organized the backend using Flask Blueprints, which allowed me to

structure the code in a modular and scalable way. Each route was designed to handle specific tasks and return consistent, JSON-formatted responses to the frontend. Through this, I gained a strong understanding of HTTP methods (GET, POST, PUT, DELETE) and how to apply them effectively in real-world applications. Flask Blueprints was used to separate modules and improve maintainability.Input validation and data formatting was implemented using Flask extensions.

4.6 Collaborative Experience

One of the most valuable aspects of my internship was the opportunity to collaborate with a professional development team. Working closely with fellow interns, UI/UX designers, and senior developers taught us how to communicate technical ideas effectively and respect the contributions of each role in the product development cycle. Collaboration was a key highlight of my internship experience at Masarrati Pvt. Ltd., where I had the opportunity to work closely with a diverse team of developers, designers, and mentors. Our workflow followed an Agile methodology, which meant frequent communication, sprint planning, and regular progress check-ins. I actively participated in daily stand-up meetings, where team members shared updates and discussed blockers. my interactions were structured around daily meetings, sprint reviews, and code walkthroughs.

I learned how to divide tasks based on individual strengths while keeping the project cohesive. Using GitHub, I handled branch creation, merge conflicts, and version tracking, which encouraged organized teamwork. Regular peer reviews helped us catch errors early and adopt better coding practices by learning from each other's feedback and techniques. The collaborative environment at Masarrati fostered a genuine sense of teamwork and shared responsibility. Working alongside developers, designers, and mentors exposed me to diverse problem-solving styles and helped me understand how interdisciplinary teams align toward a common product vision. One of the most valuable takeaways was learning to balance individual ownership with team coordination—knowing when to take initiative and when to seek support. Regular stand-up meetings and sprint reviews provided a structured space for knowledge exchange, where everyone's input was respected regardless of experience level. This inclusive and transparent culture not only enhanced my technical contributions but also improved my ability to adapt, compromise, and work effectively in a fast-paced, professional setting.

This collaborative environment also introduced us to real-world agile rituals such as stand-up meetings and sprint planning. I developed a habit of reporting progress, asking for

support when needed, and tracking blockers transparently. It improved my time management, accountability, and understanding of how large teams ship features reliably in short timelines.

4.6.1 Team Communication I collaborated daily through virtual stand-up meetings, shared task updates on Trello, and used Slack for quick clarifications. This helped reduce misunderstandings, ensured clarity across modules, and kept the team aligned on priorities. Effective team communication was one of the most transformative aspects of my internship experience. Whether it was during daily stand-up calls, collaborative code reviews, or casual brainstorming sessions, I learned the importance of expressing my thoughts clearly and listening actively to teammates. We used tools like Slack and Trello to keep updates transparent and tasks aligned across the team. This constant flow of communication allowed us to quickly resolve blockers, exchange ideas, and ensure that every team member was on the same page. More importantly, it built a sense of mutual trust and respect, where feedback was constructive and contributions were acknowledged.

4.6.2 Version Control Collaboration I used GitHub for collaborative coding, working with feature branches and submitting pull requests. This helped us resolve merge conflicts, track contributions, and maintain a clean, documented codebase suitable for professional environments. Using Git and GitHub throughout the internship taught me the importance of structured and collaborative development. This process ensured code consistency, reduced errors, and made debugging more efficient through peer validation. Merge conflicts, which initially seemed challenging, became opportunities to understand how different contributions interact within the same codebase. Regular commits with descriptive messages and the use of GitHub Issues for tracking progress also helped maintain project transparent.

4.6.3 Team Meetings and Communication Participated in regular team meetings to discuss progress, challenges, and upcoming tasks. Used collaboration tools such as Slack, Microsoft Teams, or Zoom for daily communication and stand-ups. Whether it was during daily stand-up calls, collaborative code reviews, or casual brainstorming sessions, I learned the importance of expressing my thoughts clearly and listening actively to teammates. We used tools like Slack and Trello to keep updates transparent and tasks aligned across the team. Learned the importance of clear, concise updates and asking relevant questions to ensure alignment with team goals

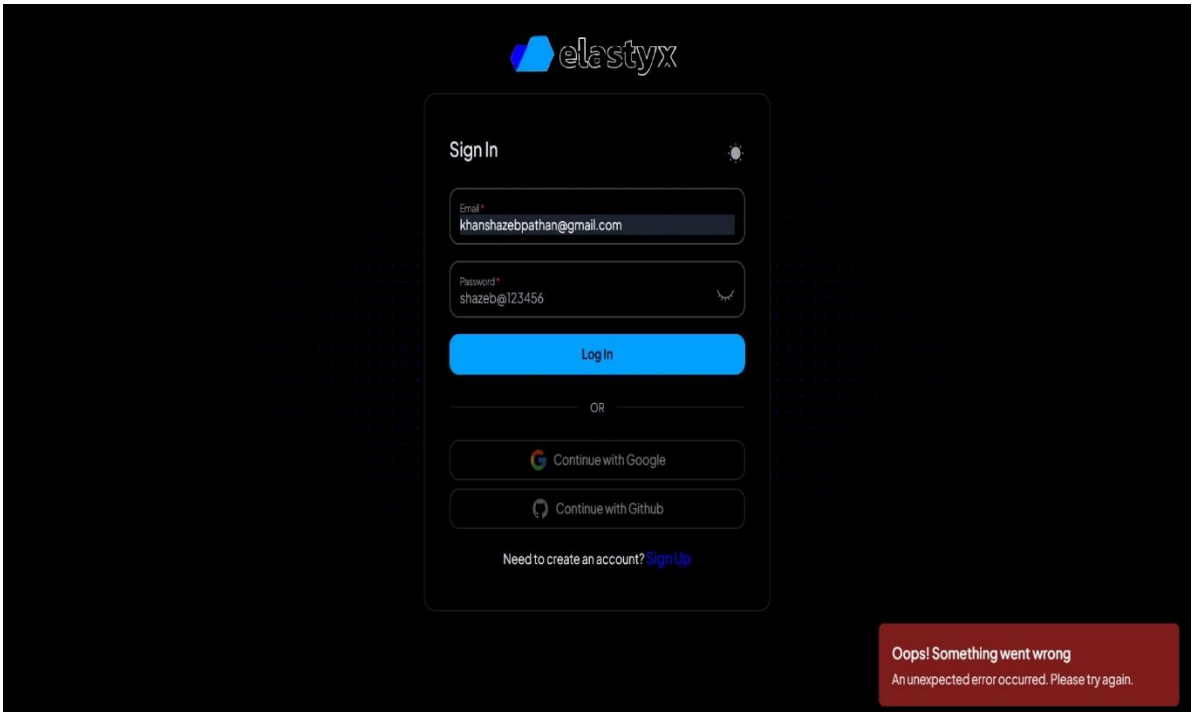

Fig 4.1 Server Error

4.7 Problem Solving and Debugging

Problem-solving was a key part of my daily tasks during the internship. Whether it was a UI bug, an unexpected API response, or a styling issue across screen sizes, I faced real challenges that required careful debugging and logical thinking. I gradually developed the ability to isolate problems, trace root causes, and resolve them efficiently using developer tools and structured testing. Problem solving and debugging were integral parts of my day-to-day responsibilities during the internship, and they significantly contributed to my growth as a developer. Whether it was a broken layout, an API returning incorrect data, or a database query failing silently, I learned how to approach issues methodically rather than getting overwhelmed. My process often began with identifying the problem through browser DevTools, where I inspected console errors, checked network requests, and traced component behavior in real-time. These tools helped me quickly isolate whether an issue stemmed from the frontend, backend, or a data mismatch between the two.

I used browser DevTools extensively to inspect elements, debug JavaScript behavior, and monitor API calls in real-time. When dealing with backend responses, I used Postman to simulate API requests and check data consistency. On the backend, I used Postman, debugging

logs, and Flask's built-in error tracing to understand how and why certain routes failed. I added custom log messages at critical points in the API to monitor variable states and detect the root causes of issues. Many times, the solution required looking at the system holistically—such as understanding how a malformed frontend request affected a backend process, or how a missing foreign key caused unexpected behavior in the database. These tools, along with console logging and code breakpoints, helped us identify and fix issues before deployment.

In more complex cases, I sought support from my mentor or collaborated with teammates to identify flaws in my logic or data handling. These experiences taught me how to troubleshoot efficiently, think logically under pressure, and implement fixes that not only solved the immediate problem but prevented future errors. Overall, debugging became more than just a task—it became a skill that strengthened my ability to build reliable, production-ready applications. Each bug fixed added to my confidence and taught us new ways to optimize both code and thought process. The experience made us more independent, solution-focused, and detail-oriented.

- **Use of Debugging Tools** I relied on browser DevTools, console logs, and breakpoints to identify errors in real-time. These tools helped us trace component behavior, inspect element states, and test responsiveness under various conditions.

- **API Testing and Backend Debugging** Postman was essential for checking request and response formats, especially during cart, wishlist, and checkout development. It allowed us to quickly identify issues with endpoints, headers, and data formatting.

- **Logical Thinking and Fix Strategy** For larger issues, I broke problems into smaller blocks and tested each segment independently. This modular troubleshooting approach helped us fix bugs methodically while reducing unnecessary code rewrites.

- **Regularly** monitored project output and logs to detect inconsistencies, bugs, or unexpected behaviors.

- **Learned** to recognize patterns in errors and symptoms to quickly identify potential root causes.

- **Utilized** tools like console logs, breakpoints, and error trackers to gain insights into issues.

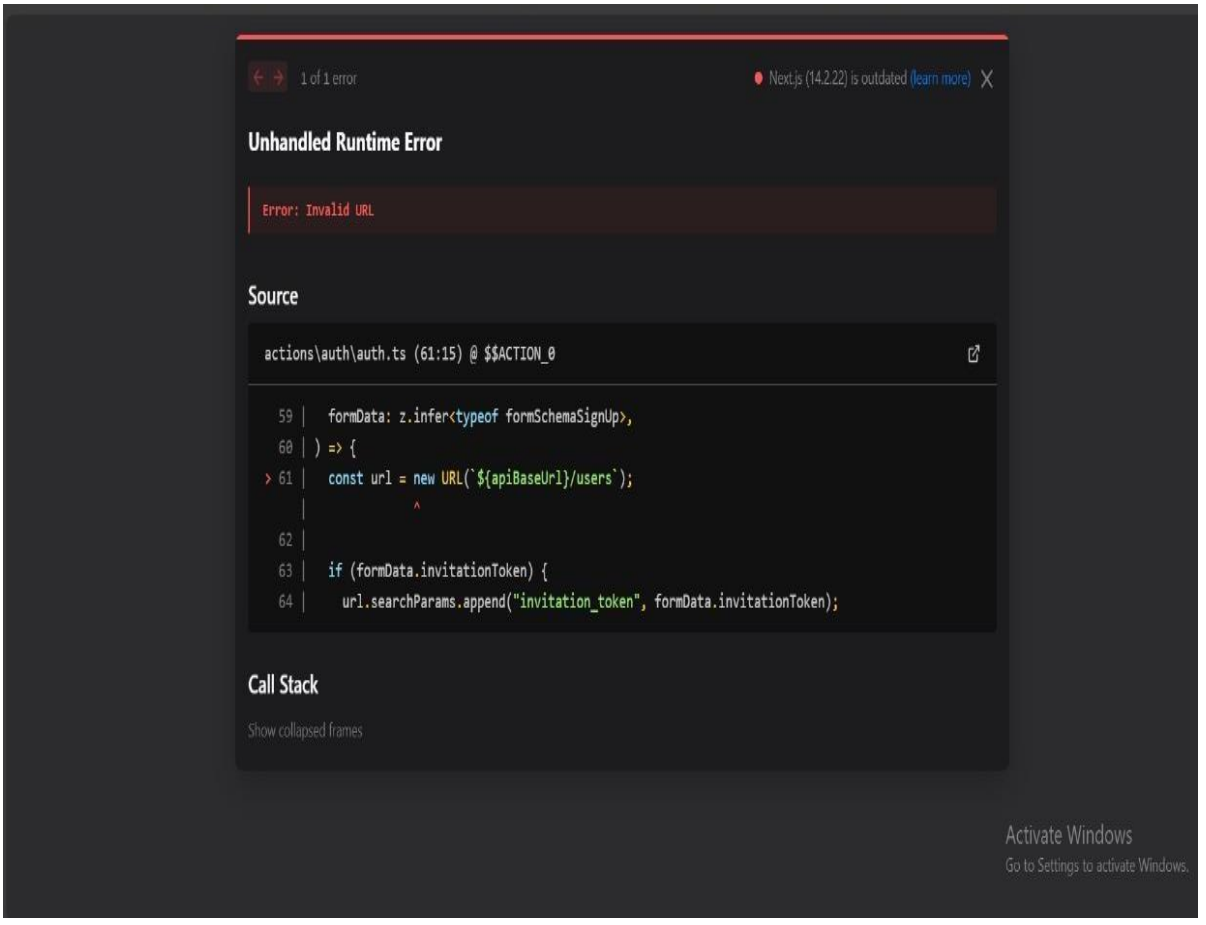

Fig 4.2 API Error

4.8 Real-Time Deployment and Hosting

Deploying my project in a live environment gave us a clear view of how real-world applications are hosted, tested, and maintained post-development. I used Vercel to deploy the React frontend and Render to host the Flask backend. These platforms made the deployment process smoother and allowed us to simulate production. One of the most exciting and rewarding aspects of my internship was learning how to deploy and host a full-stack application in real time. I was responsible for deploying the React.js frontend using Vercel, a modern platform that offers seamless integration with GitHub. By connecting the project repository directly to Vercel, each new push to the main branch triggered an automated build and deployment process. This allowed us to preview changes instantly, test new features in a live environment, and quickly respond to UI bugs or performance issues. Vercel also provided helpful deployment logs and error tracking.

Vercel provided instant previews, build logs, and domain setup for frontend code. I linked GitHub repositories directly, so every push to the main branch triggered a fresh deployment. For the Flask backend, I used Render, a cloud platform that supports Python-based applications. Render allowed me to configure backend services with environment variables, health checks, and logging features that are essential for stable real-time hosting. I handled important aspects like CORS settings, secure API routing, and database connectivity, ensuring that the backend worked reliably with the frontend even after deployment. Similarly, Render was used to manage my Flask backend, offering us clear visibility into build errors, runtime logs, and deployment health checks.

I also learned how to use environment variables, handle route mapping, and troubleshoot failed builds. Setting up continuous deployment helped us maintain code quality and ensured that each tested feature was pushed quickly and accurately to the live application. This experience gave me a complete understanding of how Continuous Integration and Continuous Deployment (CI/CD) workflows function in real-world projects. Learning to manage the full deployment cycle—from development to production—was a critical step in transitioning from classroom coding to building real, user-facing web applications. This hands-on experience gave us confidence in building scalable, maintainable software in a cloud-based environment.

- **Frontend Deployment on Vercel** I used Vercel to deploy the React.js frontend, which offered automatic builds from GitHub and real-time deployment logs. It supported custom domain mapping, route configuration, and helped us preview each new feature before going live.

- **Backend Deployment on Render** Render was used to host the Flask API backend, where I deployed routes for product, cart, and order features. Deploying the backend on Render was a key part of my internship experience and offered me hands-on exposure to real-world deployment workflows. Render is a powerful cloud platform that simplifies the hosting of backend applications, and it worked seamlessly with the Flask APIs I had built. I started by connecting my GitHub repository to Render, allowing for automatic deployments whenever changes were pushed to the main branch. I handled build settings, API base URLs, and debugged backend logs using Render's web console.

- **Handling Deployment Errors** I encountered and resolved deployment issues such as CORS errors, failed builds, and incorrect API connections. These challenges taught us how to use logs, update environment settings, and verify success post.

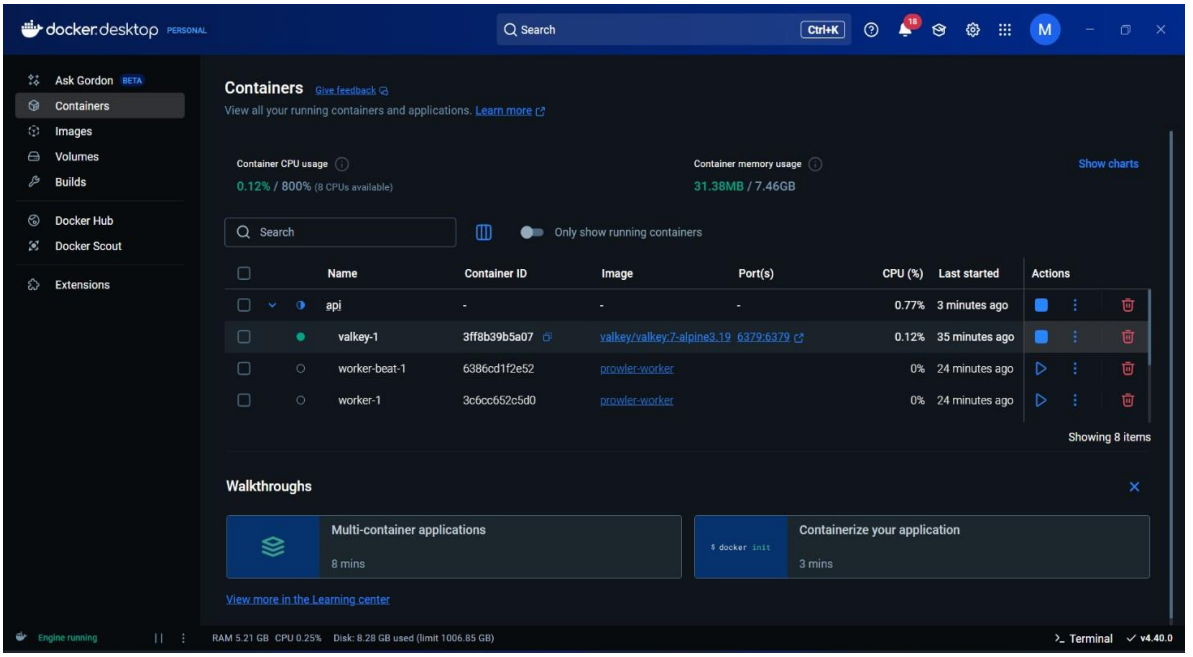

Fig 4.3 Docker Container

4.9 Key Observations & Takeaways

Our internship experience gave us valuable exposure to how software is planned, built, and released in real-world environments. I observed how different roles— developers, designers, mentors, and testers—work together through collaborative tools and workflows. Every discussion, review, and feedback loop gave us new insight into working professionally and thinking critically.

I learned that building software is not just about writing code, but about creating meaningful and usable experiences for the end user. Understanding learner needs on the Elastyx platform helped us design features with more empathy and clarity. Throughout my internship at Masarrati Pvt. Ltd., I made several important observations that have helped shape my understanding of software development beyond the classroom. One of the first things I realized was the importance of structure and planning—whether in code architecture, sprint execution, or task delegation. Unlike academic projects where deadlines are flexible and scope is limited, real-world development requires clear timelines, modular thinking, and constant collaboration to meet goals efficiently. I also observed how critical communication is—not just in team meetings, but in writing clean, readable code that others can understand and build upon. I also realized the importance of testing, feedback-driven iterations, and making data-backed.

From this internship, I took away more than just technical knowledge. I gained discipline in task management, communication confidence in meetings, and a habit of documenting progress. Another key takeaway was the emphasis on user-centric design. From the frontend UI to backend functionality, every feature was built with the end-user in mind. This taught me that technical skills must always align with usability, performance, and accessibility standards. I also came to appreciate the role of testing, debugging, and deployment as continuous processes—not just final steps—within a development cycle. Working with tools like GitHub, Postman, Vercel, and Render gave me a clear picture of how real products evolve through constant iteration and feedback. Most importantly, I learned how to think like a developer in a professional setting—balancing quality, teamwork, and adaptability in a fast-paced, goal-oriented environment. Most importantly, I developed a problem-solving mindset and a stronger sense of accountability-both of which will continue to shape my future as developers and professionals.

- **Observing Product Lifecycle** I experienced the entire product journey-from wireframes to final deployment-while contributing meaningfully at each stage. This helped us understand timelines, iterations, and client-focused delivery practices.

- **Understanding User-First Thinking** By focusing on students as end users, I improved my decision-making around design and usability. Every feature was developed with clarity, simplicity, and responsiveness in mind.

- **Growth Beyond Coding** Working in a live project helped us build real-world habits like documentation, code cleanliness, and clear version control. These soft skills became just as important as my technical contributions.

- **Observed** how professional projects are structured, planned, and executed across various teams.

- **Gained** exposure to Agile methodologies and task management tools like Jira, Trello, or Asana.

- **Realized** the importance of deadlines, quality standards, and version control in maintaining project consistency.

CONCLUSION

The development of the Elastyx platform during this project reflects a successful application of modern web technologies to meet real educational needs. With its foundation built on React.js, Flask, and MySQL, the platform delivers an interactive, responsive, and scalable solution that aligns well with the goals of future-ready learning. Each module—be it the product listing, wishlist, cart, or AI-driven recommendations—was thoughtfully implemented to support a structured, hands-on learning experience for students. This project was not only about technical execution but also about solving a meaningful problem in education—making STEM concepts accessible and engaging through practical application. By integrating AI-based content recommendations, real-time interactions, and user-focused design, Elastyx enables learners to experience technology in a more tangible and personalized way.

The full-stack approach allowed for seamless communication between the frontend and backend, creating a dynamic platform that responds efficiently to user actions. Testing with tools like Postman, version control through GitHub, and deployment via Vercel and Render ensured that the application met professional standards and was production-ready. Moreover, the agile workflow supported regular updates and collaboration across teams, reflecting industry-level project management practices. What stands out most is the platform's ability to connect code with educational impact. Every feature was developed not just to function but to enhance the user journey—from exploring kits to completing purchases and learning modules. The modular design ensured maintainability, while accessibility and responsiveness made the platform suitable for a wide range of users.

In conclusion, this project showcases how well-executed software engineering can directly contribute to educational advancement. The Elastyx platform is a testament to how thoughtful planning, technical skill, and real-world understanding can come together to create a meaningful and scalable EdTech solution. With continued development, it holds the potential to empower thousands of students across the country with the skills they need for a technology-driven future.

REFERENCES

[1]. Banks, A. and Porcello, E., "Learning React: Modern patterns for developing React apps", 2nd Edition, O'Reilly Media.

[2]. Grinberg, M., "Flask Web Development: Developing Web Applications with Python", 2nd Edition, O'Reilly Media.

[3].Wieruch, R. (2021). The Road to React: Your journey to master modern React.js.Covers real-world React development patterns, including component structure, hooks, and integration with APIs.

[4]. Mentor Guidance – Mr.Usman Senior Technical Lead at Masarrati Pvt. Ltd. Provided continuous mentorship, technical guidance, and performance feedback throughout the internship.

[5].Masarrati Pvt. Ltd. Official Website. (2016). Retrieved from https://masarrati.com/Primary source of company information, vision, and Elastyx product overview.