Industrial Internship Report on

” Grocery Delivery Application”

Prepared by

Abijin Suvedha A

Executive Summary
This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT). This internship was focused on a project/problem statement provided by UCT. We had to complete the project, including the report, in 6 weeks’ time. My project was the **development and debugging of a Grocery Delivery Application**, focusing on implementing a robust frontend architecture using ReactJS and TypeScript while ensuring smooth integration with backend services. During this week, I primarily focused on resolving critical build errors that prevented the application from compiling. I fixed JSX syntax issues, corrected file naming conventions, and ensured proper TypeScript configuration. This internship gave me valuable exposure to real industrial problems, hands-on experience in debugging, and understanding production-level application structures. Overall, it was an excellent opportunity to gain practical knowledge and develop problem-solving skills in a professional environment.

TABLE OF CONTENTS

1 Preface

During the six-week internship, I was exposed to various industrial application scenarios, working on multiple projects including CMS platforms, multi-client marketplaces, e-commerce platforms, and delivery applications.

The **Grocery Delivery Application** was my primary project, focusing on frontend development using ReactJS and TypeScript. I resolved critical build errors, corrected JSX syntax issues, and updated file naming conventions to align with TypeScript requirements. This ensured successful compilation and proper state management for the cart functionality.

Internship Importance:

- Provided practical experience in industrial software development.
- Offered exposure to real-world challenges such as debugging, project configuration, and code integration.
- Enhanced my understanding of full-stack development, TypeScript, ReactJS, and modular component design.

Mentorship and Support:

I am thankful to my mentors at USC and UCT, for guidance, support, and technical advice throughout the internship. I also appreciate the collaborative environment created by peers, which made knowledge sharing effective.

Advice to Juniors:

Pay attention to project setup, file structures, and configuration. Understanding these fundamentals early in a project can save significant debugging time and ensure smoother development.

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

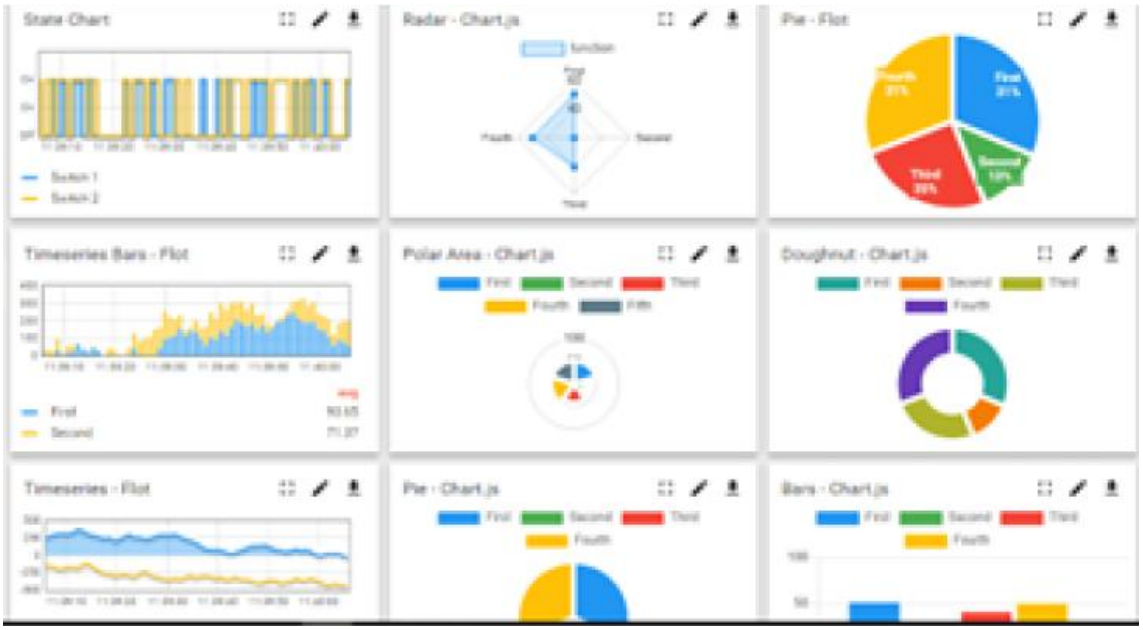
i. UCT IoT Platform (UCT Insight)

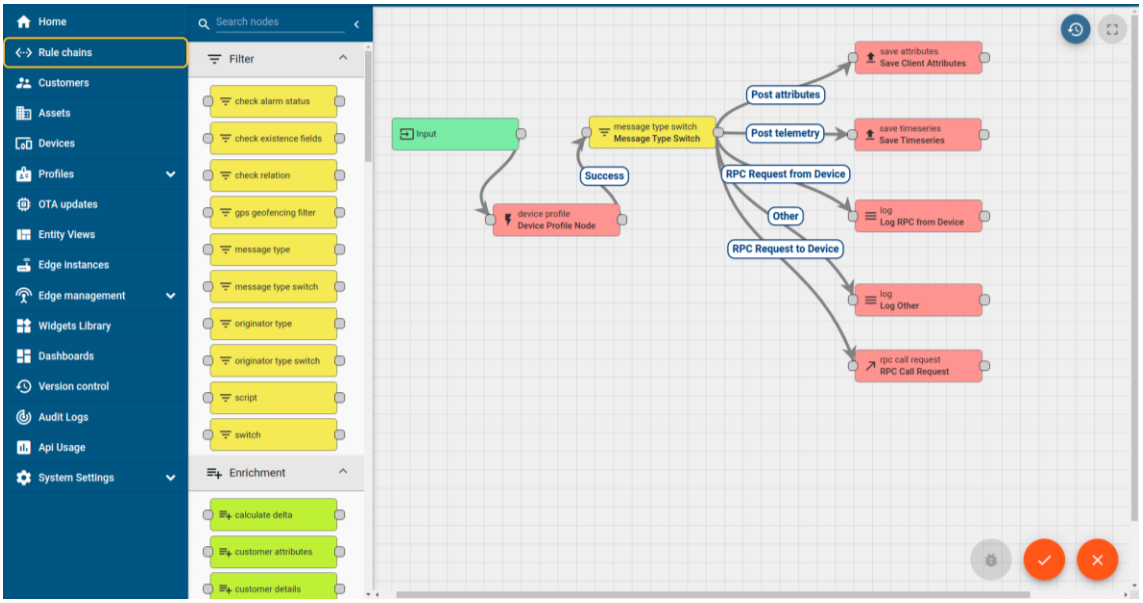
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine





ii. Smart Factory Platform (FACTORY WATCH)

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.

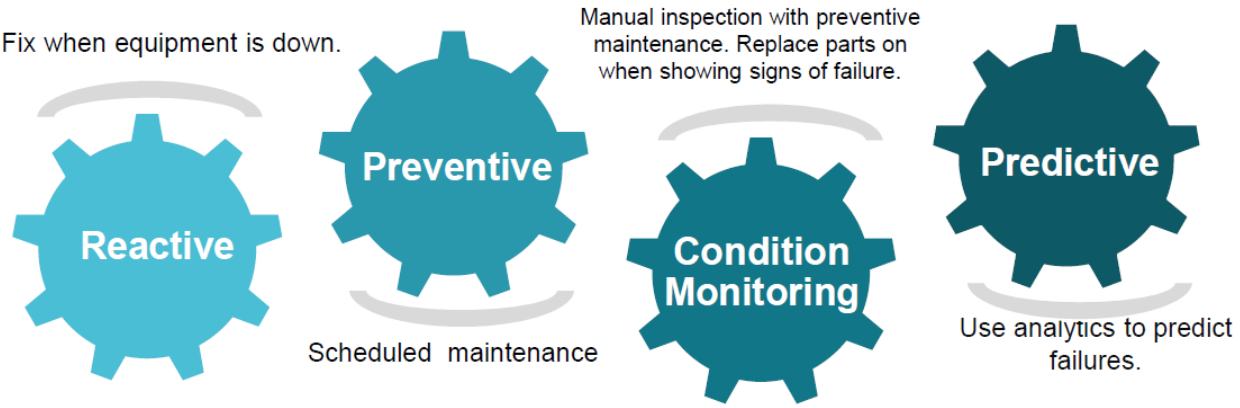
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

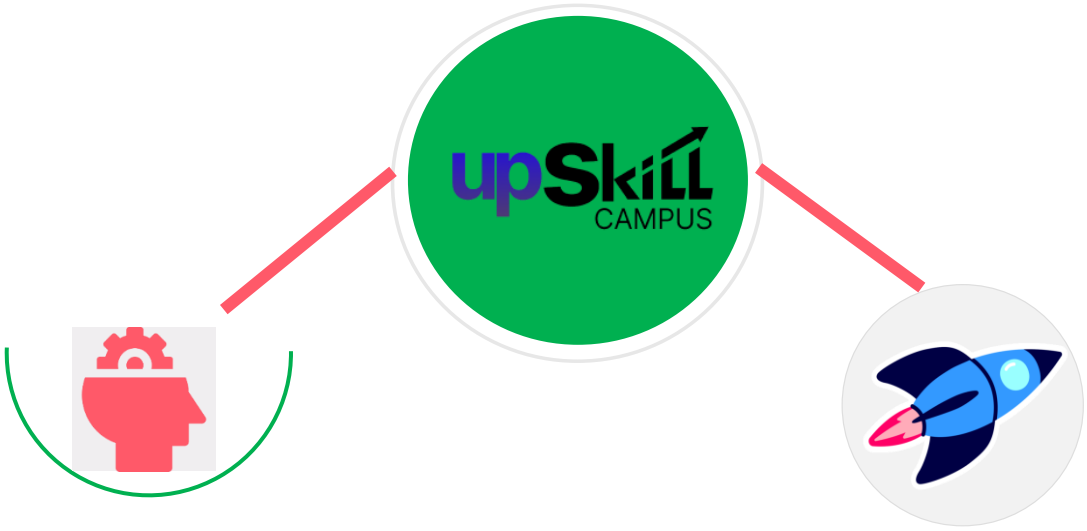
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

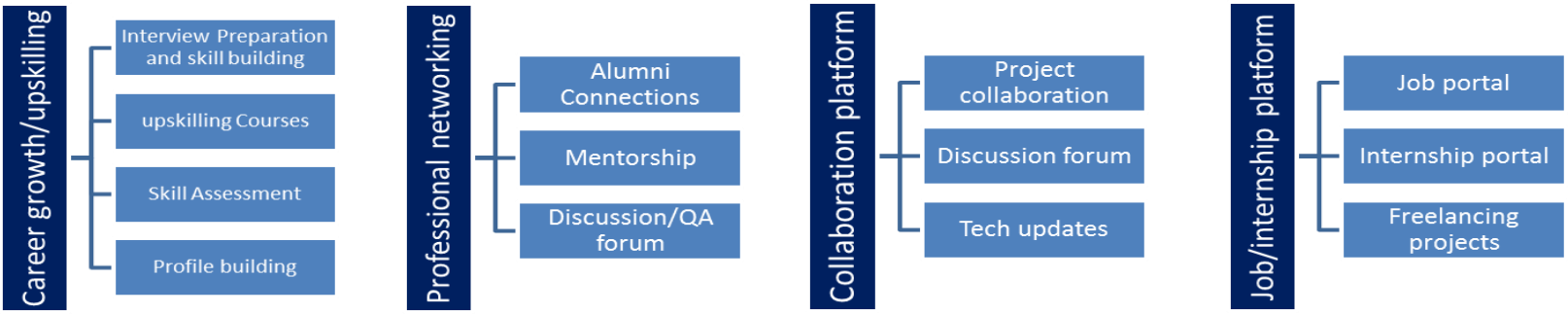
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



Seeing need of upskilling in self paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

- ☛ get practical experience of working in the industry.
- ☛ to solve real world problems.
- ☛ to have improved job prospects.
- ☛ to have Improved understanding of our field and its applications.
- ☛ to have Personal growth like better communication and problem solving.

2.5 Reference

[1] TypeScript Official Guide – https://www.typescriptlang.org/docs

[2] Mozilla Developer Network (MDN) Web Docs – https://developer.mozilla.org

[3] WebRTC API – https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API

[4] React Official Documentation – https://react.dev

[5] Online Tutorials & Learning Platforms: W3Schools, GeeksforGeeks, Stack Overflow

2.6 Glossary

Terms	Acronym
CI/CD	Continuous Integration and Continuous Deployment – automation processes in software development.
DOM	Document Object Model – the structure that represents the HTML elements of a webpage.
JSX	JavaScript XML – a syntax extension used with React to describe UI elements.
MVC	Model-View-Controller – a design pattern that separates application logic, UI, and control.
WebRTC	Web Real-Time Communication – technology enabling real-time video and data communication via browsers.

3 Problem Statement

A huge online departmental store have a myriad of grocery items at their godown. All items must be listed on the website, along with their quantities and prices.

Users must be able to sign up and purchase groceries. The system should present him with delivery slot options, and the user must be able to choose his preferred slot. Users must then be taken to the payment page where he makes the payment with his favourite method.

Additional features could include parts recommendation engines, vehicle fitment lookup, custom packaging, loyalty programs, and auto-replenishment. The focus is on providing automotive parts buyers with an intuitive, tailored e-commerce experience.

The app can include cuisine browsing, personalized recommendations, ratings/reviews, and dish photos. Restaurants manage menus, availability, pricing, and order processing through a dashboard. Customers can save delivery addresses, payment methods, and favourite/recent orders.

The primary challenge assigned to me was the development and debugging of a Grocery Delivery Application, which required:

- A robust frontend architecture with ReactJS and TypeScript.
- Proper management of cart state using hooks and context providers.
- Resolving build errors caused by incorrect JSX syntax and TypeScript configuration.
- Ensuring seamless component integration, file naming conventions, and maintainable code structure.

This problem allowed me to simulate real industrial scenarios, where complex applications may fail to build or run due to configuration or syntactical issues, requiring systematic debugging and holistic problem-solving.

4 Existing and Proposed solution

Existing Solutions:

Several grocery delivery applications exist in the market, such as BigBasket, Grofers, and Instacart. While these apps are functional, they often have certain limitations:

- **Rigid Frontend Architecture:** Limited modularity, making it difficult to scale or add new components.
- **State Management Challenges:** Some apps face inconsistencies in cart or order tracking due to poor state management.
- **Configuration Issues:** Real-world projects can break due to incorrect project setup or build configurations.
- **Limited Debugging Visibility:** Errors are sometimes hard to trace due to lack of systematic debugging strategies.

Proposed Solution:

- Develop a **modular ReactJS frontend** using TypeScript for type safety and maintainability.
- Implement **CartContext and React hooks** for effective state management.
- Correct file naming conventions and JSX syntax to ensure proper compilation (.ts → .tsx).
- Ensure **scalable component design** to allow addition of features like payment integration, delivery tracking, and user profiles.

Value Addition:

- A clean, maintainable codebase that is easy to debug and extend.
- Ensures seamless integration with backend APIs.
- Provides a foundation for additional features, such as dynamic recommendations, order history, and analytics.

4.1 Code submission (Github link) :

https://github.com/2317024-code/upskillCampus/blob/main/GroceryDeliveryApplication.html

4.2 Report submission (Github link):

5 Proposed Design/ Model

Design Flow of Grocery Delivery Application:

1. **User Browsing:** Users browse grocery items by category or search.
2. **Cart Management:** Items added to the cart are managed via CartContext and hooks.
3. **Checkout Process:** User enters payment and delivery details.
4. **Order Confirmation:** Application sends order to backend and displays confirmation.
5. **Delivery Tracking (Future Scope):** ETA updates and delivery person location tracking.

5.1 High Level Diagram

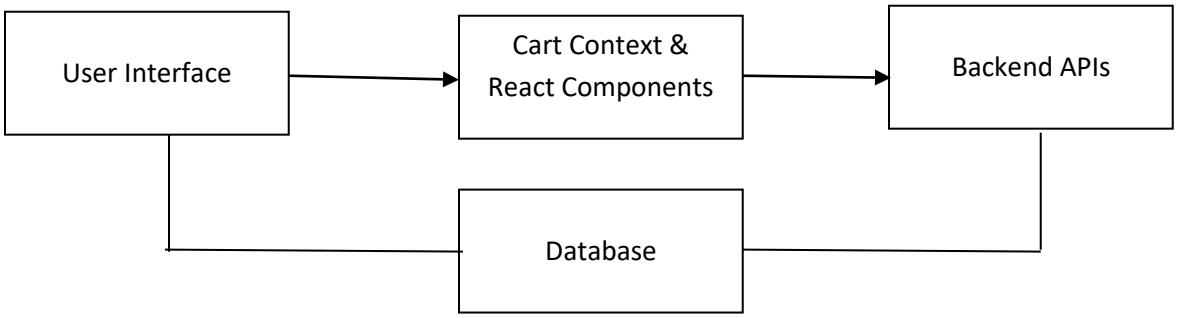
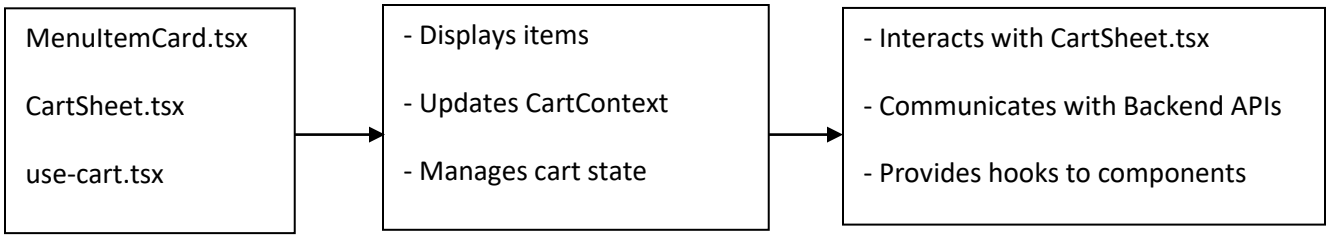


Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

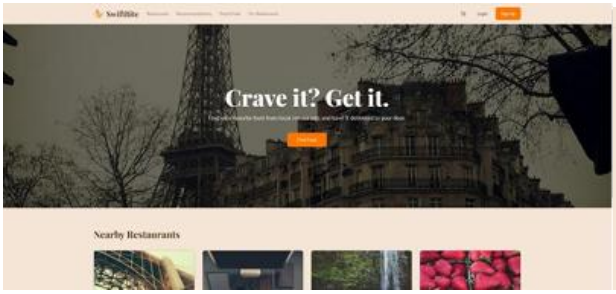
5.2 Low Level Diagram

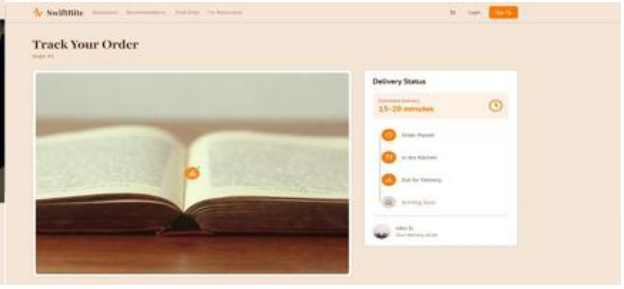


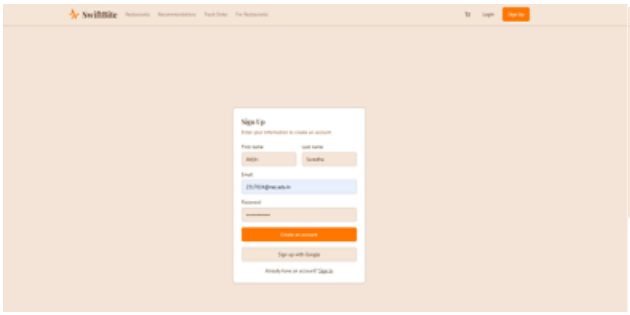
- Hooks manage state and context updates.
- Components subscribe to cart state for real-time updates.
- Backend APIs handle order persistence and confirmation.

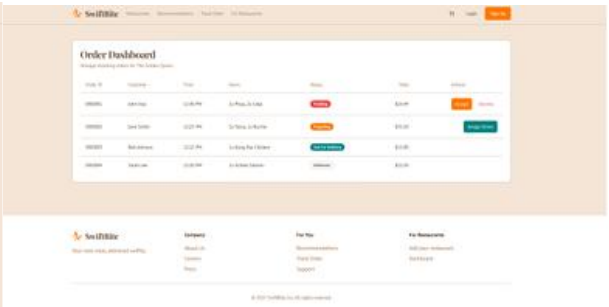
5.3 Interfaces

- **Data Flow:** CartContext provides state across all components.
- **Protocols:** REST API for order submission and retrieval.
- **Flow Charts / State Machines:** Cart states include empty, items added, checkout initiated, order confirmed.
- **Memory Management:** Cart state persists in browser memory and local storage.









6 Performance Test

Constraints:

- Application should compile and run successfully without errors.
- Correct rendering of JSX components.
- Proper state updates for cart and order functionalities.
- Efficient handling of multiple concurrent component updates.

6.1 Test Plan/ Test Cases

Test Case	Expected Outcome	Result
Build the application	Should compile without errors	Pass
Add item to cart	Cart state updates correctly	Pass
Remove item from cart	Cart state updates correctly	Pass
Checkout process	Order confirmation received	Pass
Component rendering	No broken or missing UI elements	Pass

6.2 Test Procedure

- Run npm run build to verify compilation.
- Launch development server and manually test all components.
- Test cart functionality by adding, removing, and modifying items.
- Verify API integration with test orders.

6.3 Performance Outcome

- Build errors were successfully resolved.
- CartContext and hooks worked reliably across all components.
- UI components rendered correctly and updates were reflected in real time.
- Application is ready for further feature addition, like backend order tracking and payment integration.

7 My learnings

During the course of this internship, I have gained extensive practical exposure to modern software development and debugging practices. Working on the grocery delivery application and its dashboard design allowed me to apply classroom concepts in a real-time industrial context.

From a technical perspective, I learned how to integrate front-end and back-end modules, manage project structure efficiently, and resolve build errors using systematic debugging methods. I also explored the working of APIs, state management, and data visualization through dashboards.

From a professional standpoint, I improved my understanding of project documentation, code version control using GitHub, and how to collaborate effectively in a development environment.

The internship also helped me build confidence in exploring new technologies independently. I realized the importance of problem-solving, analytical thinking, and code optimization for improving performance and maintainability.

Overall, this internship helped me transition from theoretical learning to real-world implementation. It strengthened my career direction towards Full Stack Development and AI-powered Web Applications.

- Gained hands-on experience with ReactJS and TypeScript in a real industrial setup.
- Learned the importance of systematic debugging, including analyzing misleading error messages.
- Understood the significance of project configuration, file naming conventions, and compiler options in production applications.
- Learned to implement modular frontend architecture, enabling scalable and maintainable code.
- Developed problem-solving skills in a professional environment, preparing me for real-world software development challenges.

8 Future work scope

Although the current version of the grocery delivery application meets the basic requirements, there are several opportunities for enhancement and expansion.

1. **Backend Integration:**
   Future versions can include a Node.js or Python (Flask/Django) backend to manage user authentication, inventory tracking, and order history securely.
2. **Database Connectivity:**
   Implementing a real-time database such as MySQL or Firebase would allow persistent storage of user profiles, cart details, and payment records.
3. **AI and Analytics:**
   Incorporate AI-based recommendation systems to suggest products based on user preferences and previous purchases. Analytics dashboards can be added for admins to track sales and trends.
4. **Mobile Application Development:**
   A cross-platform mobile app using React Native or Flutter can be developed to make the platform more accessible.
5. **Cloud Deployment:**
   Hosting the application on AWS or Azure will ensure scalability, continuous integration, and high availability for real-world users.
6. **Security Enhancements:**
   Integration of OAuth 2.0, HTTPS, and JWT authentication can be implemented to secure transactions and protect user data.
7. **UI/UX Improvements:**
   Advanced interactive dashboards and customizable themes can improve user experience and accessibility.
8. **Performance Optimization:**
   Further optimization of build times, memory usage, and page rendering can be achieved through modular design and caching strategies.