BSCS-S25-008

03-134221-056 SYED SHAH ABDURREHMAN

03-134221-032 MUQQADAM TAHIR

AroundYou: A Location-Based E Commerce Platform for Real-Time Consumer-Business Interactions

In partial fulfilment of the requirements for the degree of

Bachelor of Science in Computer Science

Supervisor: Mehreen Tariq

Department of Computer Sciences

Bahria University, Lahore Campus

January 2026

C e r t i f i c a t e

We accept the work contained in the report titled

"AroundYou: A Location-Based E Commerce Platform for Real-Time Consumer-Business Interactions"

written by

SYED SHAH ABDURREHMAN

MUQQADAM TAHIR

as a confirmation to the required standard for the partial fulfilment of the degree of Bachelor of Science in Computer Science.

Approved by:

Supervisor: Mehreen Tariq

_____ (Signature)

January 05, 2026

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Enrolment	Name	Signature
03-134221-056	SYED SHAH ABDURREHMAN	
03-134221-032	MUQQADAM TAHIR	

Date : January 05, 2026

Specially dedicated to
my beloved grandmother, mother and father
(SYED SHAH ABDURREHMAN)
my beloved grandmother, mother and father
(MUQQADAM TAHIR)

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to our research supervisor, Miss Mehreen Tariq for her invaluable advice, guidance and her enormous patience throughout the development of the research.

In addition, we would also like to express our gratitude to our loving parent and friends who had helped and gave us encouragement.

SYED SHAH ABDURREHMAN
MUQQADAM TAHIR

AroundYou: A Location-Based E Commerce Platform for Real-Time Consumer-Business Interactions

ABSTRACT

AroundYou is a hyperlocal marketplace platform designed to connect consumers with nearby shops, service providers and rental businesses through real time and location aware interactions. The system addresses the common challenges faced by local vendors such as low digital visibility and inconsistent customer reach, while also helping consumers access nearby products and services with reduced delivery time and cost. It allows merchants to develop digital shops, inventory, definition of Delivery Zones using geospatial polygons, orders instantly. Consumers can search for local stores via PostGIS driven geospatial searchroutine, browse products, process orders and monitor fulfillment via status updates. Processes such as including shop discovery, cart and checkout, merchant dashboards, inventory and delivery, and zone editing have been made with a mobile first and user centric focus. The application is implemented with a modern and completely serverless architecture. The frontend is made with React Native and TypeScript, with authentication offered by Supabase, PostgreSQL with PostGIS,storage and event streams. Row Level Security enforces strong access control on roles for the consumer and merchant, and there are also subscriptions for 'realtime' updates within orders tracking and merchant notifications. It integrates geospatial computation and real-time communication and a clean user experience, AroundYou facilitates efficient local commerce and enhances digital accessibility for small businesses. It establishes a scalable structure for future enhancements like more advanced delivery logistics and additional payment options and richer merchant analytics.

"We think someone else, someone smarter than us, someone more capable, someone with more resources will solve that problem. But there isn't anyone else."

Regina Dugan

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DSA	Data Structure and Algorithms
OOP	Object Oriented Programming
PF	Programming Fundamentals
SE	Software Engineering
SQL	Structured Query Language
UNESCO	United Nations Educational, Scientific and Cultural Organization
UNICODE	Unique, Universal, and Uniform Character Encoding
XML	Extensible Markup Language

LIST OF APPENDICES

CHAPTER 1: Introduction

In the current rapid-paced digital economy, small local businesses have struggled with visibility and reaching local customers. At the same time, customers have been dependent on centralized online marketplaces with limited offerings, taking longer times to deliver, and higher costs, missing out on potentially better local deals [1]. The solution offered by the AroundYou online platform regards its hyperlocal, location-based delivery service and facilitates local shoppers and stores to be tracked online.

AroundYou enables the establishment of a virtual market platform on which small businesses can set up online stores and manually control orders from these stores. Consumers can locate stores within their vicinity and browse various products on offer. These products can then be added to a cart and ordered online. The system enhances convenience and reduces costs associated with delivering goods because it uses location-aware capabilities for finding stores and tracks deliveries in real-time, thus promoting local economies as well [2].

From a technical perspective, the platform employs current mobile app technologies ensuring it is scalable, dynamic, and secured. The UI itself uses React Native, TypeScript, and Native Wind, making it extremely rapid and user-friendly. Supa base acts as an implementation solution for backend as a service. It integrates PostgreSQL with the PostGIS extension for efficient geospatial queries. Supa base Realtime handles WebSockets for making Supa Food Delivery Status updates via WebSockets. Functions involving displaying Delivery Areas, Managing Inventories, Handling Carts, as well as simplifying interactions among Consumers and Supa Vendors, occur with optimal state and data management techniques [3].

It describes the context in which problems exist, project goals, project scope, and then possible project domains, and so on. It leads on to the next chapter on system design and implementation.

1.1 Problem Description

Small businesses have traditional operations involving phone calls and walk-ins. This implies that there are inefficiencies as well as visibility and revenue challenges. On the customer side, they have various ways to interact with businesses. Different methods will be discussed. They often rely on large scale platforms that concentrate on particular categories, like food establishments, thus making it difficult for access to general retail stores and contributing to high costs of deliveries [4].

Moreover, customers do not have knowledge about available stores near them, as there is no generalized location service provided by current systems. The current status of delivery services is also fragmented, with no centralized solution for

tracking and handling delivery regions. All these deficiencies affect customers as well as vendors and hamper business expansion [5].

AroundYou aims to solve these issues with a location-based platform, which allows customers to search for local stores and make orders and tracking feasible. The platform also benefits merchants with functions for efficient inventory and delivery zone setting, ordering, and online presence management [2].

1.2 Objectives

The main goal of this project would be geared towards creating a hyperlocal online delivery platform that links customers with local stores via geolocation-based services. The specific goals would include:

•Create a user-friendly and responsive web app with React Native, TypeScript, and Native Wind CSS.

• Use Supabase as a backend service for storing data, authentication, online updates, and media.

•Take advantage of PostGIS functions for exact geospatial operations like shop lookup and finding a polygon-based delivery area.

•Allow sellers to control their stores, products, product categories, delivery regions, and delivery rules via an easy-to-use dashboard.

• Enable customers to search local stores, browse products, and make orders.

• Implement orders status updates using Supabase Realtime WebSocket subscriptions.

•Secure authentication and data handling should be ensured on the platform.

•Design a scalable system with a perspective on expanding it into various regions.

1.2.1 Major Novel Contributions

The AroundYou platform introduces several innovative features that differentiate it from traditional e-commerce systems:

• **Location-based shop finding:** It employs PostGIS functionality to identify the stores capable of delivering goods to a particular customer location based on polygon delivery regions.

•**Real-time tracking for orders:** It uses WebSockets for updating orders and allows tracking of orders with immediate updates from either consumers or sellers.

• **Merchant tools:** This package contains advanced tools for merchants. These tools

include inventory, multi-shop functionality, organizing products into categories, as well as automatic logging of

• **Dynamic delivery logic:** It allows merchants to define rules for delivery tiers, minimum orders, charges, and beyond-distance pricing.

•**Optimized state and data management:** Employing React Query and Zustand for enabling real-time synchronization and caching.

1.3 Project Scope

The AroundYou project primarily concentrates on developing a Mobile Application that will enable local shop search, ordering, and delivery services. The system allows multiple shop types like grocery stores, pharmacies, home goods stores, and general convenience stores [1]

• User authentication and role-based access control.
• Merchant dashboard for handling stores, inventory, delivery regions, and orders.

•Consumer interface for finding stores, viewing products, and ordering.

• Polygon-based Delivery Area Management using PostGIS.

•Real-time updates and alerts about orders.

•Cart management, address management, and calculation of shipping charge.

The first implementation will be primarily targeted at an urban and semi-urban area within Pakistan but may expand. It will not include either fleet or logistics automation. Last-mile delivery will be done either by merchants or third-party services [2].

1.4 Solution Application Areas

Around You adds value for several sectors within local business:

•**Retail and Grocery Stores:** Assists small sellers in organizing their inventory digitally and reaching more customers.
• **Food and Beverage Outlets:** It helps local food vendors and food establishments in ordering and delivering food.
•**Pharmacies:** Facilitates convenient ordering and distribution of necessary drugs.

•**Household and Electronics Shops:** Enables digital visibility and efficient

updates.

• **Consumers:** Provides quicker and more affordable ways to shop local stores and track deliveries.

• **Local Economies:** Improves local economies and enables more deliveries. Through the utilization of geospatial intelligence, data synchronization, and tools for merchants, AroundYou manages to revolutionize the traditional hyperlocal food delivery space and set an incredible foundation for the future [3].

CHAPTER 2: Literature Review

The chapter describes the status quo of e-commerce, delivery, and hyperlocal solutions to place the contributions of AroundYou in context. The discussion outlines strengths and weaknesses of current platforms and assesses how location-based search, real-time communication, and support for small vendors have evolved in related systems.

2.1 Introduction

Some of the key strength points of food delivery applications include strong real-time functionality, order tracking, and optimized logistic workflows. Their core competencies include solid delivery networks and a finely tuned customer experience. Nevertheless, these systems focus little else but restaurants and prepared meals, thereby supporting very little or no participation of general retail shops like convenience stores, pharmacies, electronics shops, and home-based vendors as per [5].

Despite such advances, several small and medium-sized businesses still find it difficult to establish a digital presence. Few of the existing platforms provide features of localized shop discovery, granular delivery area control, and real-time visibility of orders, thereby leaving major gaps in the adoption of hyperlocal commerce. This provides a gap for systems designed to closely integrate geolocation, local availability, and real-time workflows.

2.2 Comparative Analysis

2.2.1 Food Delivery Platforms

Food delivery applications offer strong real-time functionality, order tracking, and optimized logistics workflows. Their core strengths include reliable delivery networks and consistent customer experiences. However, these platforms focus almost exclusively on restaurants and prepared meals, limiting participation for general retail shops such as convenience stores, pharmacies, electronics shops, and home-based vendors [5]. Additionally, vendors have minimal control over defining delivery zones or configuring custom delivery logic.

2.2.2 Large-Scale E-Commerce Marketplaces

The large marketplaces, such as Daraz, focus on wide national reach, rich catalog availability, and broad fulfillment networks. These systems work well at scale but are inefficient for local discovery and immediacy of delivery. Small shops have limited visibility opportunities and cannot set boundaries on delivery areas or dynamic pricing based on distance to customer. Therefore, consumers remain unaware of retailers in proximity to them who can deliver more quickly and at lower costs [4].

2.2.3 General Classified and Listing Platforms

For example, classified platforms like OLX allow listings between user and user but lack integrated workflows for ordering, structured delivery tracking, or automated fulfillment processes [3]. They are heavily dependent on manual communication and in-person interactions that introduce latencies and reduce their suitability for local delivery where every minute counts. These systems provide visibility but do not provide real-time operational tools.

2.2.4 Emerging Hyperlocal Marketplaces

Recent hyperlocal platforms try to link customers with nearby stores, but many solutions are essentially narrow. Some allow location-based browsing but without real-time tracking, while others offer ordering but without configurable delivery areas. Most systems also lack geospatial analytics-integration, such as polygon-based delivery zones, a key enabler of accurate hyperlocal fulfillment [2, 1]. Full-stack support for merchant dashboards, inventory management, configuration of delivery logic, and live updating of orders is rare.

2.3 Positioning & Gap Analysis

The literature highlights a clear gap in platforms that combine:

- **Location-based discovery** using accurate geospatial queries-for example, PostGIS-to surface shops that can deliver to the user's exact position..
- **Granular delivery area control** area control by polygon or distance-based delivery zones set by merchants.
- **Real-time order tracking** of the order, wherein consumers and retailers

instantly get updates that enhance transparency and operational accuracy.

- **Merchant-focused** tools for inventory management, ordering workflows and shop perfor- mance analytics.
- **Support for multiple retail categories** other than restaurants: groceries, pharmacies, electronics shops, home-based sellers

AroundYou addresses these precisely by allowing for geospatial filtering, real-time updates, and flexible delivery area management, while also offering a full merchant dashboard for shop operations.

2.4 Summary

The existing platforms are focused on optimizing for either large-scale nationwide ecommerce or narrow verticals such as restaurant delivery. Classified systems facilitate visibility but come without ordering workflows and real-time logistics. The emerging set of hyperlocal tools provide only partial support, with many missing comprehensive components, such as polygon-based delivery zones, real-time order tracking, merchant dashboards, and inventory management. AroundYou addresses this inefficiency by providing a location-first, real-time, merchant-powered platform developed for neighborhood shops and hyperlocal delivery [1, 2].

CHAPTER 3: Requirement Specifications

The requirement analysis will ensure that the AroundYou platform serves the needs of both consumers and merchants while targeting the inefficiencies in the existing local commerce ecosystems. This chapter lists down both functional and we can say non functional requirements based on stakeholder expectations and real-world challenges identified in previous studies.

3.1 Existing System

Most small local shops depends on informal methods, like phone orders, walk in customers, or social media pages to manage sales. These methods are old and lacking in real time tracking, inventory , and structured order work flows. Consumers often rely on big delivery platforms optimized for specific verticals, such as food delivery, which limits access to nearby general-purpose shops, increases delivery times, and inflates delivery costs. Minimal support for the spatial discovery of neighborhood shops or automated delivery area management fosters inefficiencies in both parties' visibility.

3.2 Proposed System

Shop discovery based on location, making use of the geospatial queries of PostGIS.

• Real-time order tracking powered by Supabase Realtime.

• Shopping/inventory via a web-based merchant dashboard.

• Mapping area creation through polygon-based mapping tools.

• Calculating the delivery fee automatically based on shop-defined rules and distance.

Unlike other general e-commerce or single-vertical delivery platforms, AroundYou enables small local retailers to digitize operations with minimal technical overhead.

It provides a single point of interaction on mobile for consumers and merchants through the React Native-built interface over Supabase services.

3.3 Functional Requirements

3.3.1 User Roles

The two main user roles supported by this platform include:

• Consumer: Allows clients to locate the nearest stores, view goods, add items to a

cart, make orders, and trace deliveries.

• Merchants: create shops, manage inventory and delivery areas, set delivery settings, and configure order workflows.

3.3.2 Consumer Functional Requirements

- C1: Sign up, login, profile management, password resetting.
- C2: Allow location access or set location manually; system must fetch shops deliverable to that point.
- C3: view listed shops with distance, delivery fee, and operating status.
- C4: View the listing of shop items according to their categories; price, availability description.
- C5: Handle shopping cart entries: addition, deletion, change in quantity counts.
- C6: Users place orders using saved addresses. System stores a snapshot of items, prices, and delivery fee for the order.
- C7: Provide real-time order tracking using WebSockets.
- C8: Cancel orders prior to merchant confirmation where possible.
- C9: Manage multiple delivery addresses with geolocation.

3.3.3 Merchant Functional Requirements

- M1: Register, log in, manage merchant profile.
- M2: Ability to create and manage more than one shop with just one merchant account.
- M3: Setup the shop information like name, status of your shop, contact information, and images.
- M4: Inventory Management - add, edit, and delete items. Assign category and use of item templates
- M5: Create delivery areas using polygon mapping tools.
- M6: Define delivery logic: distance tiers, minimum order value, surcharges, free-delivery radius.
- M7: View incoming orders with status updates and fulfill:

  pending → confirmed → out_for_delivery → delivered.
- M8: Matching delivery runners to orders.
- M9: Revenue, order, and item performance analytics available for review

3.3.4 System Functional Requirements

- S1: It will then use Supabase Auth for user authentication.

- S2: Implement RLS for all tables in the database.
- S3: Look up shops by location using PostGIS
- S4: Calculate the delivery fee using merchant-defined logic.
- S5: Broadcast real-time order updates using Supabase Realtime.
- S6: Audit logging for inventory and shop updates.
- S7: In Supabase Storage, media files will be stored for items and shop images.

3.4 Non-functional Requirements

- **Performance:** PostGIS queries should resolve within 20–40 seconds including fallback; item/category loads must complete under 300 ms under normal load.

- **Real-time:** Order status updates must propagate within 1–2 seconds using Supabase Realtime.

- **Availability:** Target uptime of 99.5%; system must gracefully fall back to polling when WebSockets fail.

- **Scalability:** Support horizontal scaling for web servers; use indexed geospatial queries (GIST) for expanding shop coverage.

- **Security:** RLS, JWT authentication, encrypted storage, strict access policies, and validation on all inputs.

- **Data Integrity:** Order status transitions must be validated by database triggers to prevent invalid transitions.

- **Observability:** Logging of WebSocket disconnections, PostGIS RPC failures, and order status confirmations
- **Accessibility**: Responsive UI using Native Wind; WCAG 2.1 AA compliance for consumer flows.

- **Backup/Restore**: Automated daily PostgreSQL backups with restore testing once per quarter.

3.5 Use Case Diagrams

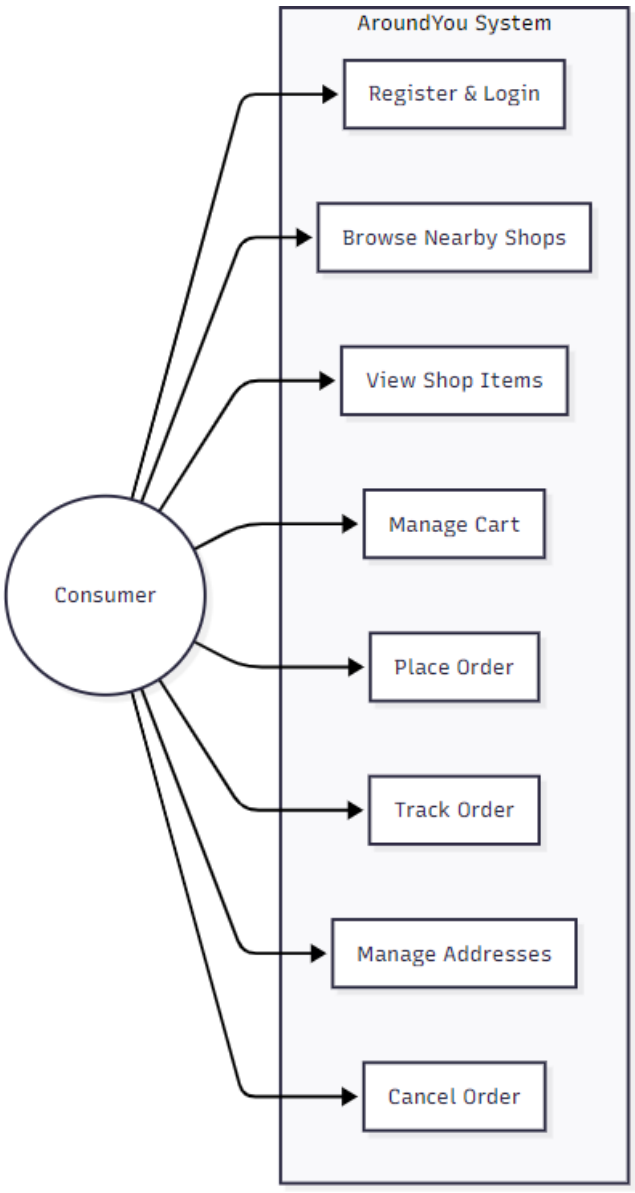

Figure 3.1: Use Case Diagram for Consumer

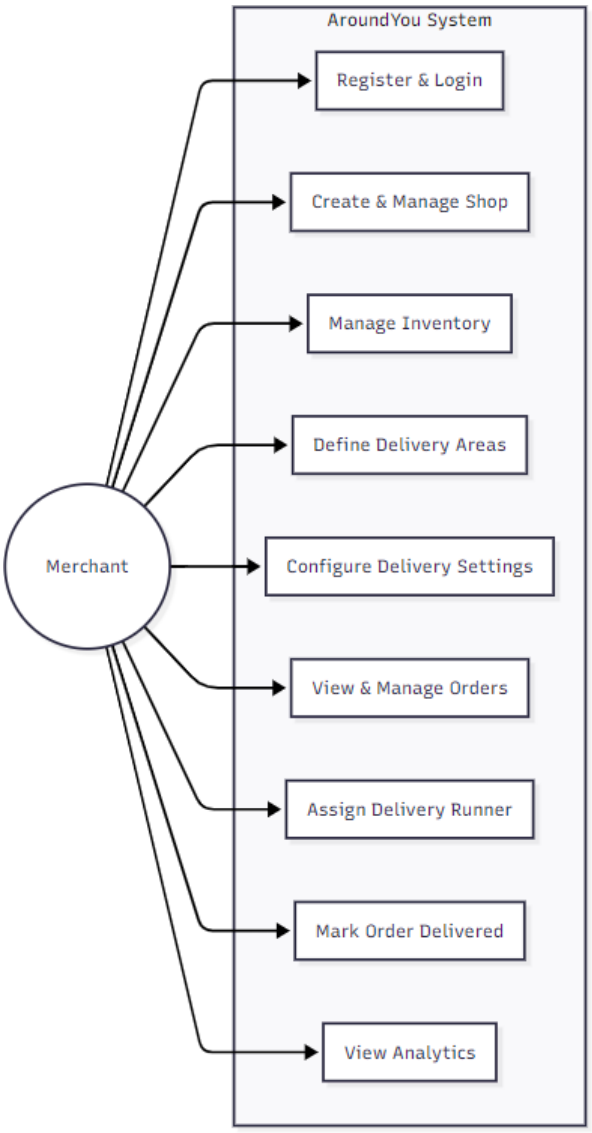

Figure 3.2: Use Case Diagram for Merchant

3.6 Use Cases

3.6.1 Consumer Use Cases

3.6.1.1 UC-01: Consumer Registration

Actor	Consumer
Pre/Post	No account / Account created and verified
Main Flow	1. Open registration 2. Enter details 3. System validates 4. System sends verification email 5. Consumer verifies

Table 3.1: UC-01 Consumer Registration

3.6.1.2 UC-02: Browse Nearby Shops

Actor	Consumer
Pre/Post	Location provided / Shop list displayed
Main Flow	1. Consumer provides location 2. System performs PostGIS lookup 3. Shops displayed with distance and delivery fee

Table 3.2: UC-02 Browse Nearby Shops

3.6.1.3 UC-03: Place Order

Actor	Consumer
Pre/Post	Items in cart / Order created
Main Flow	1. Review cart 2. Select address 3. Confirm order 4. System stores price snapshots 5. Merchant notified

Table 3.3: UC-03 Place Order

3.6.1.4 UC-04: Track Order

Actor	Consumer
Main Flow	1. Open tracking screen 2. Receive real-time updates via Supabase Realtime

Table 3.4: UC-04 Track Order

3.6.2 Merchant Use Cases

3.6.2.1 UC-05: Manage Inventory

Actor	Merchant
Main Flow	1. Add/edit items 2. Assign categories 3. Save 4. Audit log recorded

Table 3.5: UC-05 Manage Inventory

3.6.2.2 UC-06: Define Delivery Areas

Actor	Merchant
Main Flow	1. Open map editor 2. Draw polygon(s) 3. Save area 4. System validates geometry

Table 3.6: UC-06 Define Delivery Areas

3.6.2.3 UC-07: Manage Orders

Actor	Merchant
Main Flow	1. View incoming orders 2. Confirm or cancel 3. Assign runner 4. Mark delivered

Table 3.7: UC-07 Manage Orders

3.7 Requirement Traceability (Excerpt)

Requirements	Description	Mapped Use Cases
C2 (Location search)	Shop delivery using PostGIS	UC-02
C5 (Cart/Order)	Order creation and price snapshots	UC-03
M4 (Inventory)	Manage shop items	UC-05
M5 (Delivery areas)	Polygon-based delivery zones	UC-06
M7 (Order lifecycle)	Order status transition	UC-07
S5 (Real-time)	Supabase Realtime updates	UC-04, UC-07

CHAPTER 4: Design

The complete system design of the AroundYou platform appears in this chapter. The design

The design follows the real implementation structure which shows both general and detailed system components.

The system requires decisions which enable scalability and precise geospatial data and instant communication and multiple user functions.

The system provides both consumers and merchants with essential functionality. The system operates through a modular structure which uses event-driven operations.

The system operates under a serverless model which focuses on delivering high performance alongside easy maintenance and secure access management.

4.1 System Architecture

The AroundYou platform uses serverless architecture where Supabase provides database, authentication, storage and real time channels. The frontend is a React Native and TypeScript application that communicates directly with Supabase using autogenerated endpoints and WebSocket connections.

4.1.1 Architectural Overview

The primary components include:

- **Frontend Application** built with React Native, TypeScript and NativeWind. It handles all user interface rendering, form handling, consumer and merchant flows and real time event subscriptions.
- **Supabase Backend** which integrates PostgreSQL, PostGIS, authentication, object storage and real time listeners for live updates.
- **PostgreSQL with PostGIS Extension** used for relational data storage, spatial polygon processing, distance calculations and delivery coverage queries.
- **Supabase Realtime** that pushes live data updates to the frontend for order tracking, merchant dashboards and inventory changes.
- **Supabase Storage** for storing shop images, item images and user uploaded assets.

4.1.2 Architectural Structure

The system follows a clean client to Supabase interaction model in which:

- React components communicate with Supabase using autogenerated REST endpoints and RPC functions.
- All business logic is implemented through PostgreSQL functions, triggers and relational constraints.
- Real time communication uses row level event detection from Supabase Realtime.
- Row Level Security rules ensure strict separation between consumer and merchant data access.

4.2 System Architecture Diagram

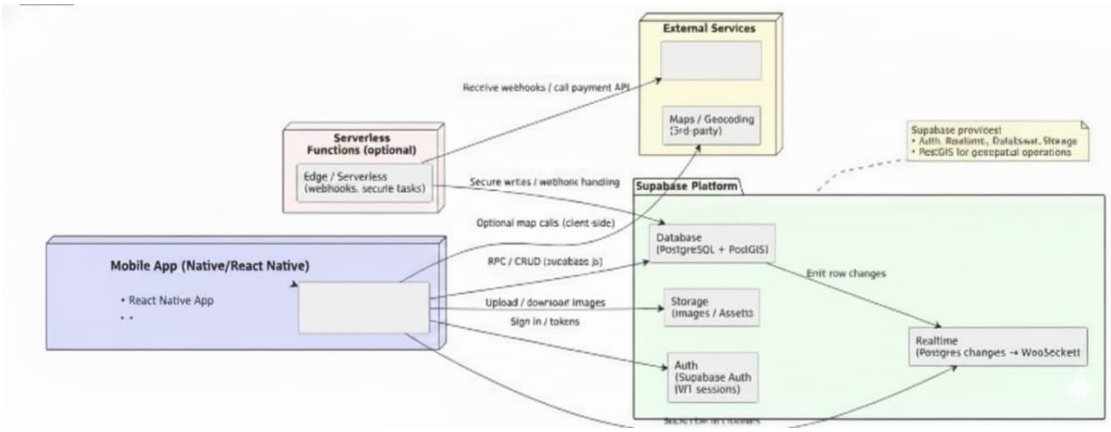

Figure 4.1: High level architecture of the AroundYou system.

4.3 Design Constraints

The design of the platform is shaped by several operational constraints:

- **Performance** maintained through spatial indexes, efficient queries and fast frontend rendering.
- **Real time Requirements** for immediate order status updates and merchant notifications.
- **Security** enforced through Supabase Auth and Row Level Security.
- **Scalability** supported through React Native deployments and Supabase distributed infrastructure.
- **Reliability** through backup policies, trigger validations and reconnection logic for WebSocket channels.

- **Usability** with a mobile first interface and clear user flows.

4.4 Design Methodology

The project followed an iterative and Agile oriented methodology:

- Requirements were divided into small functional modules.
- Each module was wireframed, prototyped and validated before implementation.
- Database schema, triggers and security policies were added progressively.
- Continuous testing was performed through manual QA and integration tests.

4.4.1 Software Process Model

A lightweight Agile model was adopted with:

- Sprint planning and weekly reviews.
- Feature flags for controlled rollouts.
- Continuous deployment using React Native and Supabase migrations.

4.5 High Level Design

4.5.1 Component Overview

The frontend uses a modular component structure with the following categories:

- **Consumer Screens** include shop discovery, shop details, item browsing, cart, checkout and real time order tracking.
- **Merchant Screens** including dashboard, multi shop management, inventory management, order handling, analytics and delivery zone configuration.
- **Shared Components** such as navigation bars, forms, item cards, modal windows and map-based elements.

- **Service Layer** that abstracts Supabase operations such as inventory reads, order creation, shop discovery and delivery fee calculation.

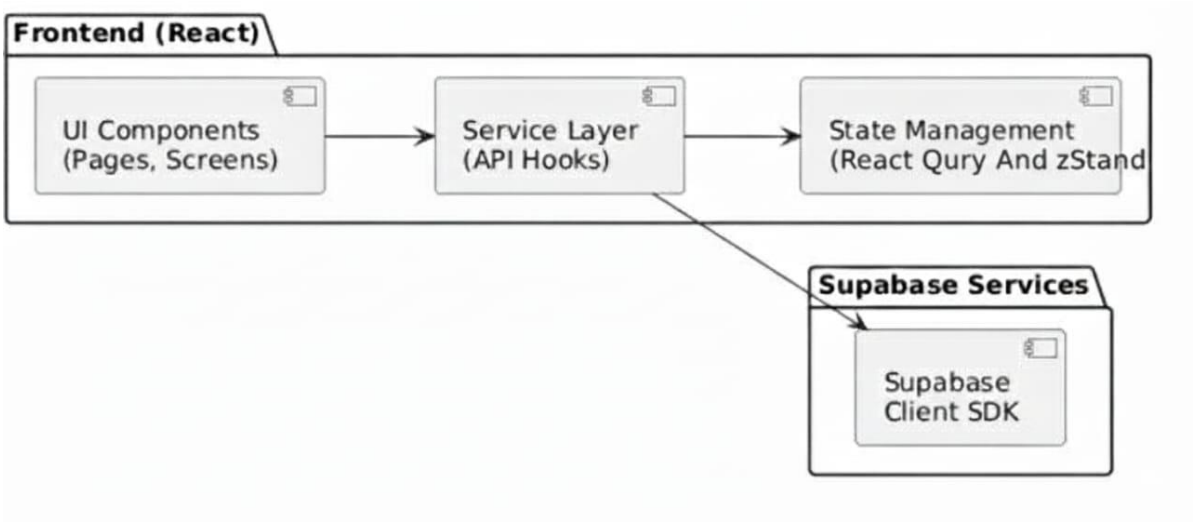

Figure 4.2: Component level structure of the frontend application.

4.5.2 Deployment Diagram

The platform uses a fully serverless deployment environment:

- The React Native application .
- Supabase manages all backend functionality including the database, authentication, storage and real time connections.
- No standalone backend server is required.

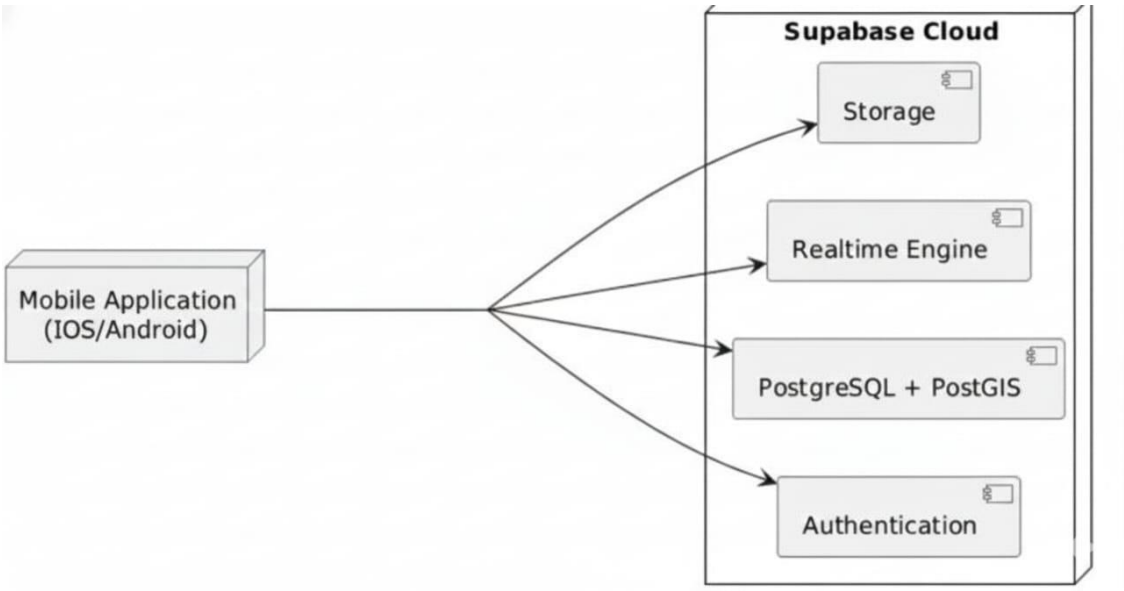

Figure 4.3: Deployment model using React Native and Supabase.

4.5.3 Activity Diagram

Figure 4.4: Activity flow for typical consumer ordering.

4.5.4 Sequence Diagram

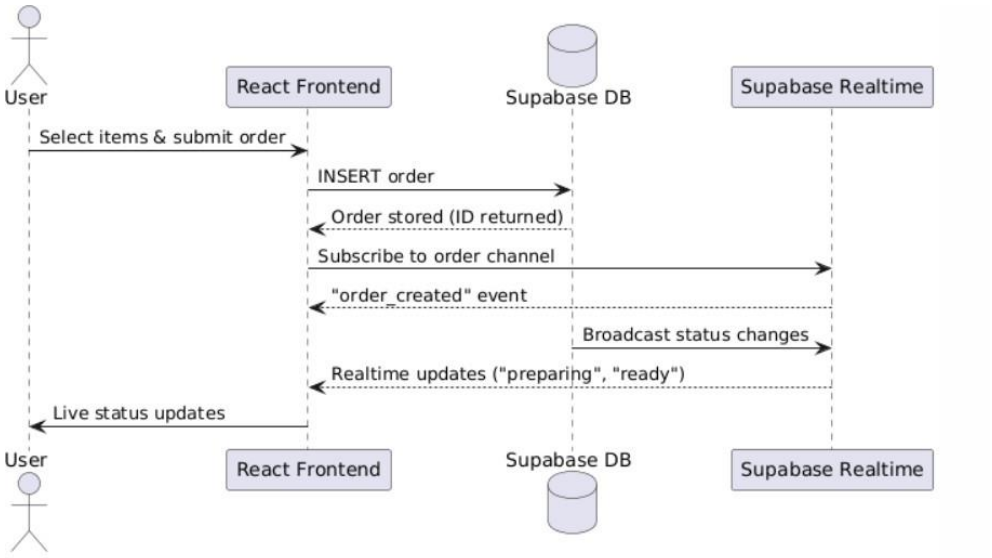

Figure 4.5: Sequence of events for order placement and realtime updates.

4.6 Low Level Design

4.6.1 Domain Model

The domain model includes the following core entities:

- **user_profiles** for user roles and identity.
- **shops** for merchant owned shops with coordinates and delivery settings.
- **merchant_items** for items, prices, status and categories.
- **orders** for consumer orders with status history.
- **order_items** for item snapshots and quantities.
- **shop_delivery_areas** for PostGIS delivery polygons.
- **audit_logs** for historical inventory updates.

4.6.2 Entity Relationship Diagram

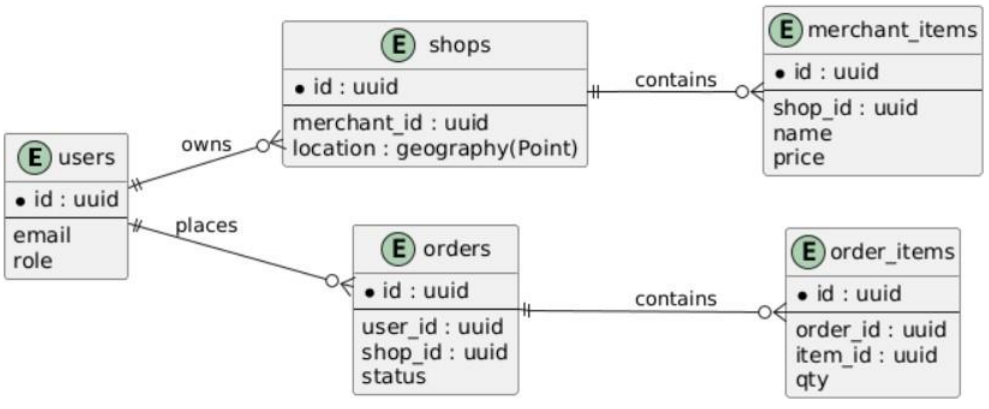

Figure 4.6: Entity relationship diagram showing core tables of the system.

4.6.3 Geo Search Design

PostGIS powers the geospatial features of the platform:

- Delivery areas stored as geometry polygons.
- The find_shops_by_location RPC performs spatial containment checks through ST Contains.
- Distance calculations used for delivery fees and ordering logic.

4.7 GUI Screens and Workflows

This section presents representative user interface designs and the workflows that guide consumers and merchants through the platform.

4.7.1 Consumer Interface Workflows

4.7.1.1 Shop Discovery Screen

The shop discovery page displays nearby shops retrieved through geospatial queries. Each shop is shown with delivery fee, distance and open status. A fallback list appears when spatial search is unavailable.

4.7.1.2 Shop Detail and Item Browsing

Items are displayed in categories. Users can add items to the cart and view descriptions, images and availability.

4.7.1.3 Cart and Checkout Screens

The cart screen shows selected items, quantities and delivery fee. Checkout applies price snapshots and validates address details.

4.7.1.4 Order Tracking Interface

Consumers receive realtime updates through Supabase Realtime and can view progress from pending to delivered.

4.7.2 Merchant Interface Workflows

4.7.2.1 Merchant Dashboard

Provides an overview of revenue, orders and shop activity with multi shop selection.

4.7.2.2 Inventory Management

Merchants can add or edit items. All changes are logged into audit logs.

4.7.2.3 Order Management

Orders appear instantly. Merchants can confirm, assign runners and mark delivery completion.

4.7.2.4 Delivery Area Editor

An interactive map interface allows merchants to draw polygons. Triggers prevent overlapping areas and validate geometry.

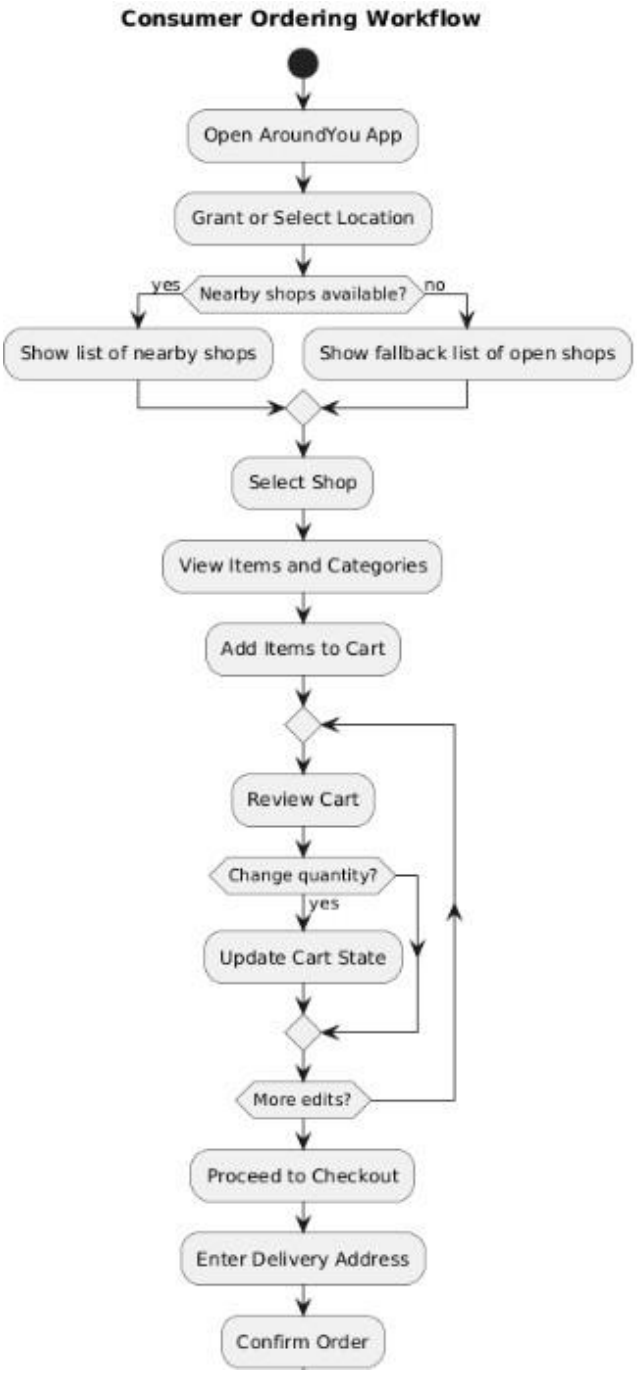

Figure 4.7

4.7.3 Workflow Diagrams

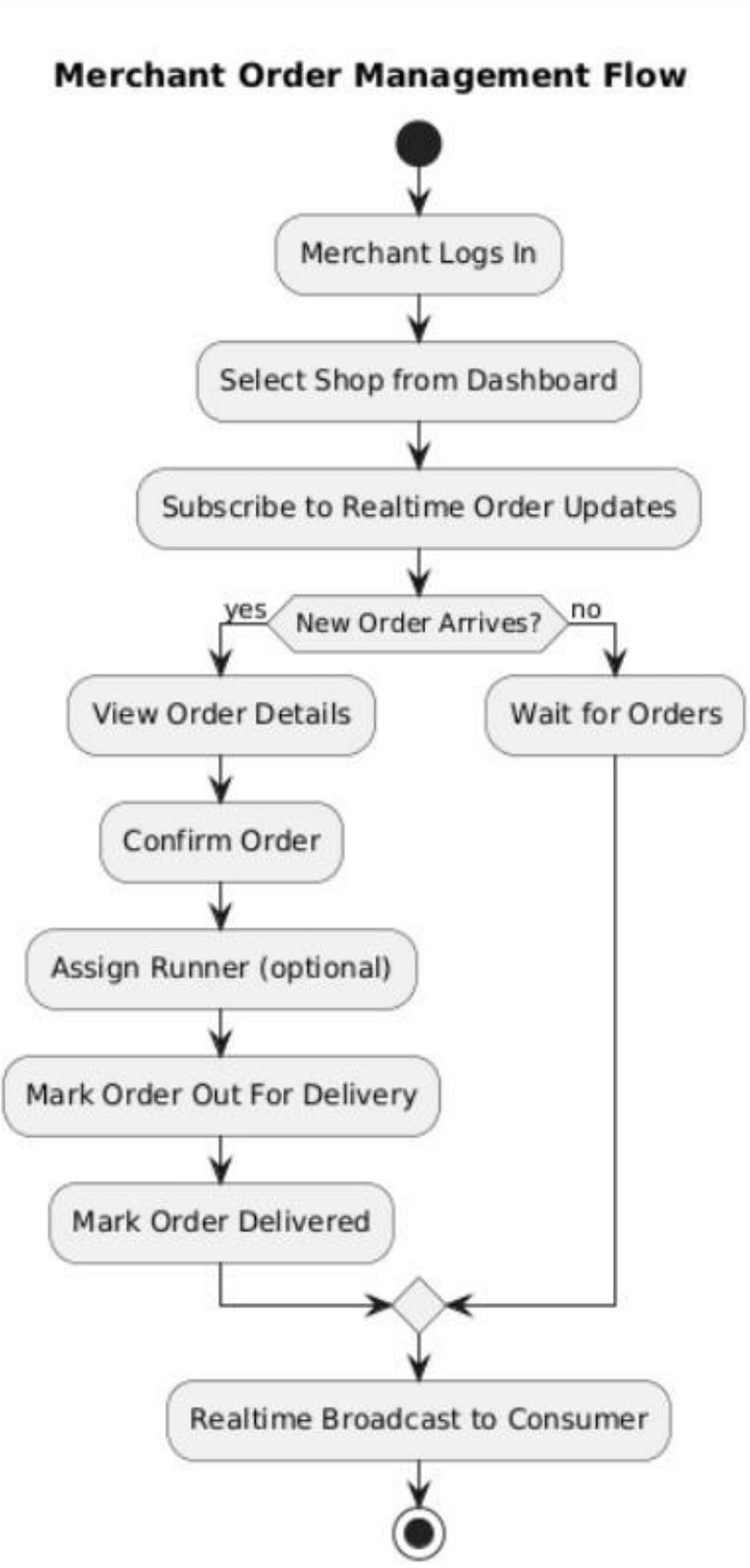

Figure 4.8: Merchant order management workflow diagram.

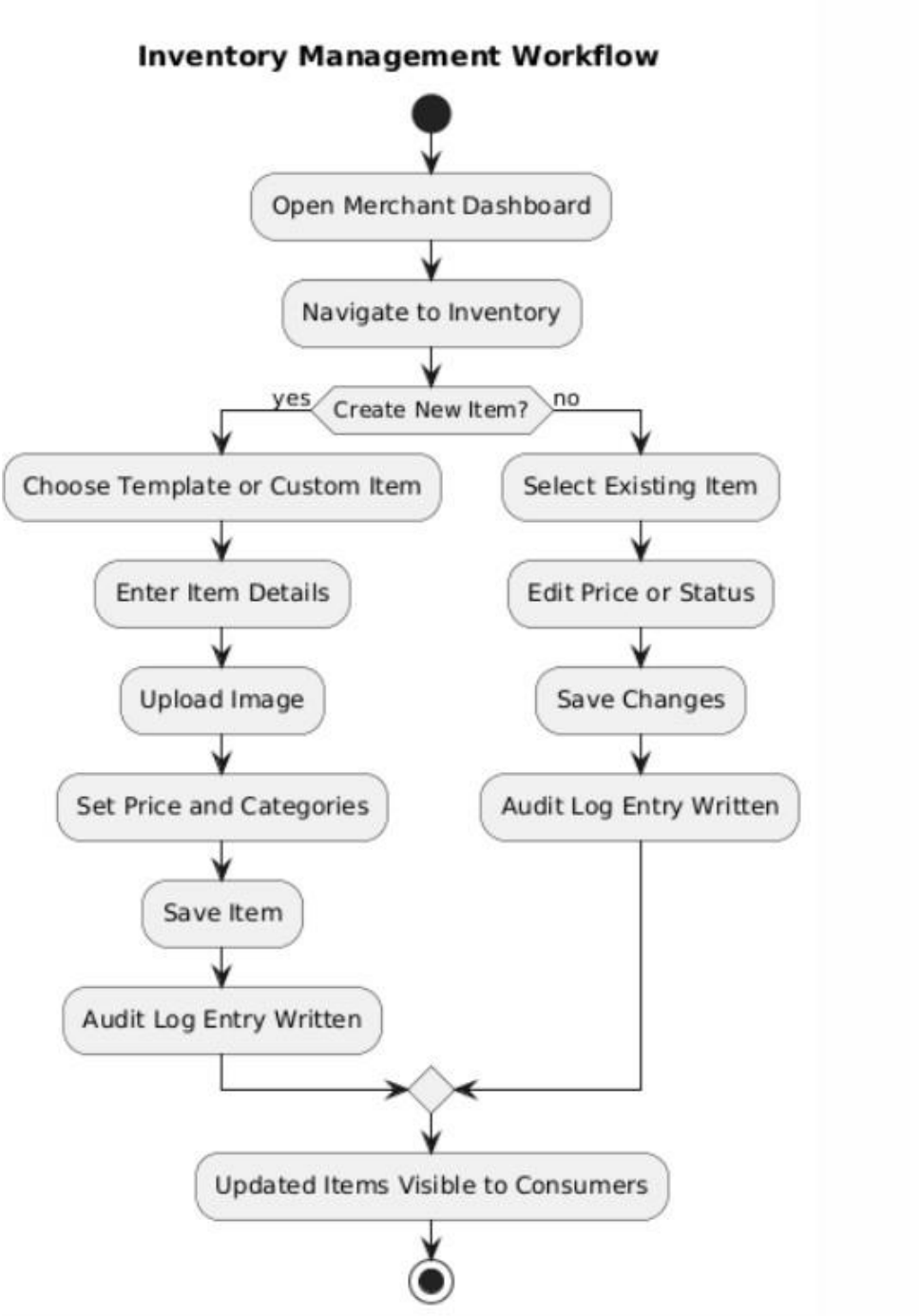

Figure 4.9: Inventory management workflow diagram.

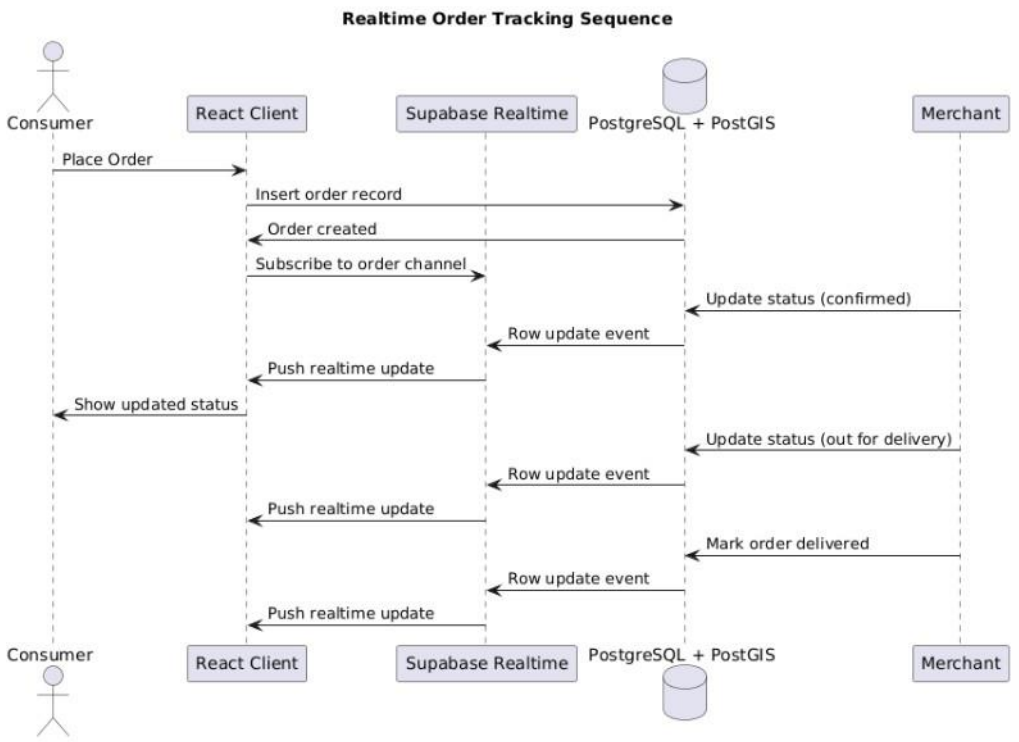

Figure 4.10: Realtime order tracking sequence workflow.

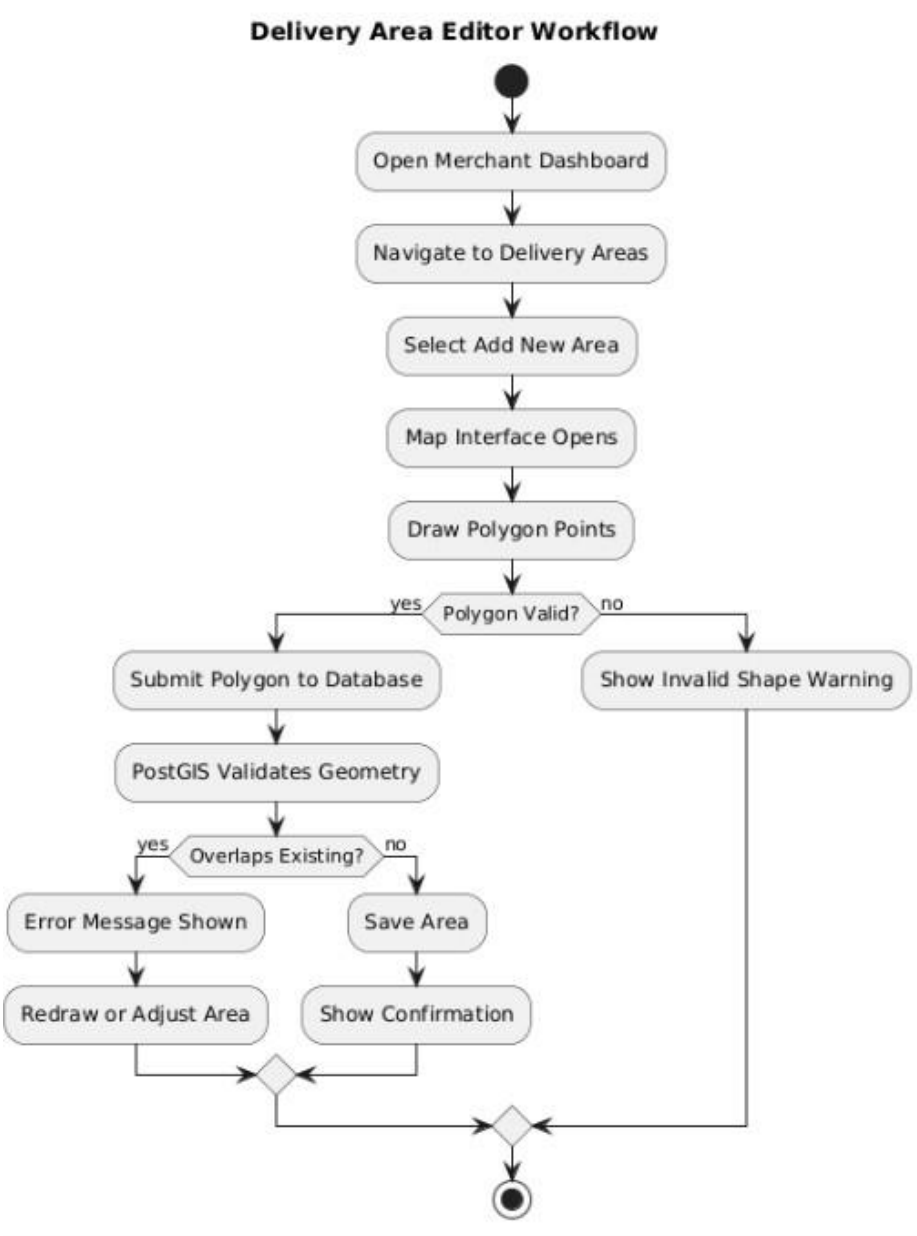

Figure 4.11: Delivery area polygon editor workflow diagram.

4.8 External Interfaces

The system integrates with:

- **Supabase Auth** for identity management.
- **Supabase Storage** for media uploads.
- **Supabase Realtime** for event driven updates.
- **PostGIS** for spatial processing.

CHAPTER 5: System Implementation

5.1 Overview

This chapter describes the implementation of AroundYou using a modern serverless architecture powered by React Native (TypeScript) on the frontend and Supabase as the complete backend- as-a-service. Supabase provides authentication, PostgreSQL with PostGIS extensions, real-time data channels, role-based security policies, and storage for images. This chapter details the technology stack, module level implementation, representative client–server interactions, database schema, geospatial search functions, security configurations, audit logging, and deployment workflow.

5.2 Technology Stack

- **Frontend:** React Native, TypeScript, NativeWind, React Router, React Query, React Hook Form, Zod for schema validation.
- **Backend / BaaS:** Supabase (Auth, Database, PostGIS, Realtime, Storage).
- **Database:** PostgreSQL 15+ with PostGIS for geospatial operations.
- **Data Access:** Supabase JavaScript client (supabase-js) for RPCs, CRUD operations, authentication and subscriptions.
- **Integrations:** External payment gateway (webhooks handled through serverless functions) and optional third-party notifications.
- **DevOps:** Deployment via Play Store (frontend), Supabase-managed infrastructure, GitHub Actions for CI/CD, automated backups, SQL migrations.

5.3 Architecture and Services

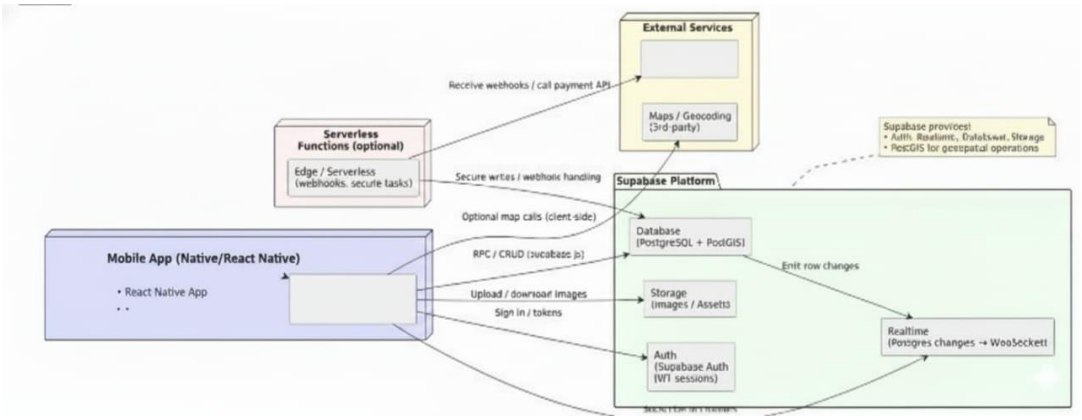

Figure 5.1: High-level architecture of AroundYou.

5.3.1 Modules

The application is organised into the following modules:

- **Authentication & Users:** Managed by Supabase Auth with secure JWT sessions. User metadata is stored in user_profiles.
- **Shops & Catalog:** Merchant-managed shops and items; images stored in Supabase Storage; products reference shop IDs.
- **Geospatial Search:** Shop locations and delivery areas stored in PostGIS geometries; all lookup logic implemented in database functions.
- **Orders:** Atomic creation of orders and order items using RPCs.
- **Payments:** Payment status synchronised via secure webhook serverless functions.
- **Realtime Updates:** Order status changes broadcast through Supabase Realtime channels.

5.4 Representative Interactions

5.4.1 Client-side Order Placement (TypeScript)

Order placement is handled through a strongly typed Zod schema and a Supabase RPC function that performs atomic operations server-side.

[language=TypeScript, style=code, caption=placeOrder() in TypeScript] // src/api/placeOrder.ts import supabase from "@/lib/supabase"; import z from

"zod";

export const OrderItemSchema = z.object(itemId: z.string().uuid(), qty: z.number().int().positive(),

);

export const PlaceOrderSchema = z.object(userId: z.string().uuid(), shopId: z.string().uuid(),

items: z.array(OrderItemSchema), address: z.object(lat: z.number(), lng: z.number()), pay- mentMethod: z.enum(["card", "cod"]),);

export type PlaceOrderInput = z.infer<typeof PlaceOrderSchema>;

export async function placeOrder(input: PlaceOrderInput) const parsed = PlaceOrder- Schema.parse(input);

const data, error = await supabase.rpc("place$_{o}$*rder*", p_{u}*ser*$_{i}$*d* : *parsed.userId,* p_{s}*hop*$_{i}$*d* : *parsed.shopId,* p_{i}*t*

if (error) throw new Error(error.message); return data;

5.5 Database Layer

5.5.1 Schema Overview

Core tables used across the system include:

- user_profiles: user metadata and roles.
- shops: merchant shops with location and open/close state.
- shop_delivery_areas: polygons describing delivery zones.
- merchant_items: items sold by each shop.
- ordersand order_items: transactional ordering system.
- payments: cash on delivery.
- audit_logs: centralised audit trail for inserts/updates/deletes.

5.5.2 Geospatial Search Function

The find_shops_by_locationfunction returns shops whose delivery areas contain the user's position and sorts them by distance.

[language=SQL, style=code, caption=find$_{s}$*hops*$_{b}$*y*$_{l}$*ocation*()*implementation*]*CREATEORREPLACEFUN* *ST*$_{S}$*etSRID*(*ST*$_{M}$*akePoint*(*p*$_{l}$*ng, p*$_{l}$*at*), 4326) :: *geography*;

RETURN QUERY SELECT s.id, s.name, ST$_{D}$*istance*(*s.location, point*)*ASdistance*$_{m}$*, s.is*$_{o}$*penFROMshops s.idWHERESTC**ontains*(*a.geom,* *ST*$_{S}$*etSRID*(*ST*$_{M}$*akePoint*(*p*$_{l}$*ng, p*$_{l}$*at*), 4326))*ANDs.is*$_{o}$*pen = TRUEORDER*

END;

;

5.6 Audit Logging

5.6.1 Audit Table

[language=SQL, style=code, caption=Audit log table] CREATE TABLE IF NOT EXISTS audit$_{l}ogs$(*idUUIDPRIMARYKEYDEFAULTgen$_{r}$andom$_{u}$uid*(), *table$_{n}$ameTEXTNOTNULL, actionTEXTN*

5.6.2 Logging Trigger Function

[language=SQL, style=code, caption=Generic audit trigger] CREATE OR REPLACE FUNC- TION log$_{a}$*udit*()*RETURNSTRIGGERLANGUAGE plpgsqlASDECLAREuidUUID*; *BEGINuid* := *auth.uid*();
IF (TG$_{O}$*P* =′ *DELETE*′)*THENINSERTINTOaudit$_{l}$ogs*(*table$_{n}$ame, action, record$_{i}$d, old$_{d}$ata, per f ormed UPDATE*′)*THENINSERTINTOaudit$_{l}$ogs*(*table$_{n}$ame, action, record$_{i}$d, old$_{d}$ata, new$_{d}$ata, per f ormed$_{b}$y*)*VAL*

5.6.3 Attach Triggers

[language=SQL, style=code] CREATE TRIGGER
trg$_{a}$*udit$_{s}$hopsAFTERINSERTORU PDATEORDELETEON* CREATE TRIGGER
trg$_{a}$*udit$_{o}$rdersAFTERINSERTORU*
PDATEORDELETEONordersFOREACHROW
CREATE TRIGGER trg$_{a}$*udit$_{i}$temsAFTERINSERTORU*
PDATEORDELETEONmerchant$_{i}$temsFOREAC

5.7 Row-Level Security (RLS)

Supabase enforces user-level access control using PostgreSQL RLS.

5.7.1 Enable RLS

[language=SQL, style=code] ALTER TABLE user$_{p}$*ro f ilesENABLEROW LEVELSECURITY*; *ALTERTABLE*

5.7.2 User Policies

[language=SQL, style=code] CREATE POLICY "Users manage own profile" ON

user$_{p}$*rofilesFORSELECTUS id*)*FORU PDATEUSING*(*auth.uid*() = *id*);

5.7.3 Merchant Policies

[language=SQL, style=code] CREATE POLICY "Merchants read shops" ON shops FOR SE- LECT USING (merchant$_{i}$*d* = *auth.uid*());
CREATE POLICY "Merchants update shops" ON shops FOR UPDATE USING (merchant$_{i}$*d* =
auth.uid());

5.7.4 Order Policies

[language=SQL, style=code] CREATE POLICY "Consumers read own orders" ON orders FOR SELECT USING (user$_{i}$*d* = *auth.uid*());
CREATE POLICY "Merchants read shop orders" ON orders FOR SELECT USING (shop$_{i}$*dIN*(*SELECTidF auth.uid*()));

5.8 Configuration

Frontend Environment

[style=code, caption=.env] $_{S}$*UPABASE*$_{U}$*RL* = *htt ps* : *//pro ject.supabase.coVITE*$_{S}$*U PABASE*$_{A}$*NON*
...NODE$_{E}$*NV* = *production*

Secure Serverless Functions

Webhook handlers use the service role key stored in environment secrets.

5.9 Deployment and Observability

- **Frontend:** Deployed on Playstore; uses preview deployments.
- **Backend:** Supabase manages the database, auth, storage and realtime.
- **Monitoring:** Supabase dashboards track CPU, RAM, query latency, and realtime channel throughput.
- **Error Tracking:** Error Boundary file For monitoring can capture runtime errors.
- **Backups & Migrations:** The system includes automated backup functionality and SQL migration scripts which execute through Supabase CLI during CI/CD operations.

5.10 Summary

The implementation utilizes React Native and Supabase for a scalable, secure, real-time system.
platform with minimal overhead for its operation. PostGIS brings in accurate geospatial search capabilities while
RPC functions and RLS strongly enforce security and atomic business logic. This architecture
Offers high performance and a clean, modern development experience suitable for production and academic settings.

CHAPTER 6: System Testing and Evaluation

6.1 Evaluation Instructions

System testing evaluates the fully implemented AroundYou platform against its functional and
Non-functional requirements The objective is to check whether the integrated system behaves as expected.
across all practical scenarios, from location-based shop discovery to consumer ordering.
Cart/Checkout functionality, inventory management of merchants, multi-shop operations, delivery
spatial queries for area processing, and real-time updates powered by Supabase Realtime. In
Besides correctness, qualitative measures include responsiveness, usability, user satisfaction.

6.2 Graphical User Interface Testing

Scope: Graphical interface testing covered all major consumer and merchant screens including the landing page, nearby shops list, shop detail page, shopping cart, checkout flow, order tracking interface, merchant dashboard, inventory management screens, analytics views and delivery area editor.

Methodology:

- Component level testing using React Testing Library.
- End to end flow verification using Playwright based scripted journeys.
- Visual snapshot comparison to ensure consistency across different builds.
- Interaction flow validation based on the design guidelines of the project.

Area	What was tested	Result
Nearby shops	Spatial search, fallback behaviour, loading states	pass

Shop detail	Item listing, add to cart actions	pass
Cart and checkout	Form validation and price snapshot logic	pass
Order tracking	Live status updates using Supabase Realtime	pass
Merchant dashboard	Shop switching and inventory editor	pass

Table 6.1: Graphical interface verification summary.

Navigation clarity and consistent feedback were validated across all pages. All screens followed the responsive layout guidelines defined in the system architecture.

6.3 Usability Testing

Method: Usability testing was performed with participants representing consumers and merchants. Each participant completed task based scenarios while being observed. Measurements included task completion rate, time spent and perceived difficulty. A SUS score was collected after each session.

Tasks tested:

- Discover a nearby shop using location permission.
- Add items to a cart and complete checkout.
- Track an order in real time.
- Manage inventory items inside the merchant dashboard.
- Add a delivery area using the polygon editor.

Task	Success %	Avg Time (s)	Notes
Shop discovery	100	24	simple and intuitive flow
Checkout flow	100	58	participants found the process clear
Order tracking	100	20	real time updates understood easily
Inventory update	90	75	template adoption clarified during test
Delivery area creation	85	88	helper text improved after feedback

Table 6.2: Usability test results (SUS mean score: 84 out of 100).

These outcomes align with the project focus on smooth user journeys and minimal friction during core operations.

6.4 Software Performance Testing

Performance testing targeted the API endpoints responsible for spatial shop discovery, order placement and inventory operations. Tests were performed on the full stack configuration that includes React Native frontend, Supabase backend and

PostgreSQL with the PostGIS extension.

Metrics evaluated:

- API latency for the fiftieth and ninety fifth percentile.
- Execution time of PostGIS spatial queries.
- Payload size efficiency.
- Rate of order creation under typical usage.

Endpoint	P50 (ms)	P95 (ms)	Notes
Shop discovery	82	205	spatial index provided efficient search
Order placement	145	292	transaction with price snapshots
Inventory fetch	95	184	stable joins and pagination

Table 6.3: Performance metrics under normal usage.

All endpoints achieved the expected median latency of less than three hundred milliseconds specified in the project requirements.

6.5 Compatibility Testing

Compatibility testing ensured that the platform behaves consistently across different device types, screen sizes and network conditions. The project emphasised a responsive and widely accessible interface.

Browsers tested: Chrome, Firefox, Edge. **Devices tested:** Mobile, tablet and desktop viewports. **Network conditions:** Two point five G, four G and WiFi.

Scenario	Configuration	Result
Mobile view	360px width layout	pass
Desktop view	1440px width layout	pass
Firefox browser	tracking protection enabled	pass
Slow network	map loads gradually but UI remains responsive	pass

Table 6.4: Compatibility testing overview.

Native Wind styles and the mobile first design approach ensured consistent rendering throughout.

6.6 Exception Handling

System reliability was validated by simulating common operational errors. The

behaviour was compared with the resilience strategies described in the implementation chapter including graceful fallback, detailed error messages and automatic recovery.

Faults injected:

- RPC timeouts for shop discovery triggered the fallback list of open shops.
- WebSocket disconnection resulted in automatic reconnection with exponential backoff.
- Database errors returned controlled responses through Supabase.
- Expired JWT tokens produced an automatic sign out and refreshed session workflow.
- Invalid request payloads were rejected through Zod validation without any crash.

Observed behaviours matched the expected resilience criteria of the system.

6.7 Load Testing

Load testing was executed using k6 to evaluate performance under heavy usage. The scenario simulated high frequency shop discovery requests, concurrent order creation and real time updates for a large number of connected clients.

Scenario:

- Gradual increase to one hundred requests per second.
- Ten minute sustained load period.
- Five hundred WebSocket clients receiving constant real time order updates.

[style=code, language=JavaScript, caption=k6 script for GET shops] import http from "k6/http"; import sleep, check from "k6"; export const options = stages: [duration: "2m", target:
100 , duration: "10m", target: 100] ; export default function () const url = "https://api.example.com/api/v1/shops const res = http.get(url); check(res, "status200": r => r.status === 200, "p95<300ms": r => r.timings.duration < 300); sleep(1);

Endpoint or Event	P50 (ms)	P95 (ms)	Error %	Notes
Shop discovery	86	215	0.2	stable performance
Order creation	143	295	0.5	minor spikes at peak moments
Realtime updates	118	470	–	five hundred clients connected

Table 6.5: Load testing results at one hundred requests per second.

The system remained stable and complied with the performance expectations of the project.

6.8 Security Testing

Security testing verified the enforcement of authentication, authorization and data protection mechanisms. The tests reflected Supabase Row Level Security rules, API protections and frontend validation.

Areas evaluated:

- JWT authentication behaviour and session refresh.
- Role based access control for merchants and consumers.
- Input validation using Zod schemas.
- Protection against SQL injection through parameterised queries.
- Rate limiting and brute force resistance.
- Encryption of network traffic through SSL.
- Avoidance of logging sensitive information.

All tested areas met expected behaviour and no security vulnerabilities were identified.

6.9 Installation Testing

Installation tests verified that the platform can be deployed reliably using the instructions provided in the deployment chapter. Both development and production workflows were validated.

Checks performed:

1. Supabase environment variables load correctly.
2. Database migrations apply successfully including creation of PostGIS extension.
3. Vite production build runs correctly behind a static host.
4. Backup and restore functions produce accurate database snapshots.

Check	Result
Fresh deployment	pass

Upgrade	with	preserved	data	pass	Environment	configuration	test	pass

Table 6.6: Installation testing summary.

6.10 Evaluation Metrics and Results

Quantitative metrics:

- All core endpoints achieved a ninety fifth percentile latency below three hundred millisec- onds.
- Median WebSocket delivery time remained below one second for five hundred clients.
- The overall error rate remained below one percent under sustained load.

Qualitative metrics:

High usability with a System Usability Scale score of eighty four.

- Participants completed main tasks with minimal guidance.
- Real time behaviour for orders and analytics was perceived as smooth and fast.

Strengths:

- Accurate geospatial search enabled by PostGIS.
- Reliable real time updates through Supabase Realtime.
- Scalable multi shop inventory and order management.
- Strong performance and consistent user experience.

Limitations:

- Dependence on external map services for geocoding.
- Real time operations increase resource usage.
- Only basic payment methods implemented.

Future improvements:

- More payment provider support.
- It ensures more accurate estimation of delivery.
- Improved analytics and logging.

- Caching using Redis or content delivery networks.

CHAPTER 7: Conclusions

The chapter will discuss the findings obtained from conceptualizing and implementing and evaluating
the Around You platform. It highlights insights on the technical knowledge obtained from developing a full stack
hyperlocal food delivery and shop management system with thoughts on some limitations of the system.
comparative strengths and opportunities for future development. The chapter also describes
recommendations for making it possible for reliable functions and scalable operation on the platform.

7.1 Key Technical Takeaways

- Geospatial search enabled with PostGIS capabilities delivered an efficient and accurate location-based service discovery. Spatial queries involving ‘point in polygon’ operations and distance computation were performed times.
- Real-time communications via Supabase Realtime made it possible for there to be immediate updates for Status updates regarding orders, a merchant side notification system, and changes within an inventory. All these factors served to confidence in the live tracking experience.
- A multi shop architecture with well-structured database relationships made it possible for merchants for handling multiple shop and inventory locations as well as several area locations for delivering data consistency via Row Level Security.
- A thin service layer on the backend with strong validation and transactional business methods guaranteed correct behavior with regard to ordering, delivery charge calculation, and inventory.
- Database schema planning and index design had a significant impact on overall system performance. Spatial indexes on delivery area polygons and BTree indexes on frequently queried fields improved latency more than any low level application optimisation.

7.2 Critical Appraisal and Limitations

- Last mile delivery processes were not implemented which restricts real time delivery runner tracking and accurate time predictions for customers.

- Payment support is limited to simple flows. Advanced capabilities such as automated refunds, recurring billing and dispute management require further development.
- The platform does not include extensive analytics for operational monitoring. Deep tracing, error impact analysis and behavior based insights remain limited.
- Real time performance depends on stable internet connectivity. Large spikes in simultaneous WebSocket clients can increase server workload.
- Spatial accuracy depends on external geocoding quality. Incorrect or imprecise address coordinates can affect shop discovery results.

7.3 Comparative Evaluation

Compared with conventional e commerce systems, AroundYou offers precise hyperlocal search through geospatial processing and supports multi shop environments with flexible delivery areas.Unlike restaurant-focused delivery platforms, the system generalizes across different categories of shops and services. Combination of real time updates multi shop inventory management, and polygon-based delivery zones introduces greater adaptability but also introduces added complexity in handling data, ensuring reliability, and onboarding merchants.

7.4 Evidence from Testing

Chapter 6 summarizes test results that show the primary user journeys of discover a shop, Placement of orders was reliably done, as was the tracking in real time.Responsiveness of the system met Target expectations were also rated positively, and the interface learnability received a positive rating from the participants. Minor issues Issues identified during testing were addressed with improvements to text explanations, helper messages and confirmation prompts.

7.5 Future Work

- Integrate courier and logistics systems for real-time delivery tracking, improving the accuracy of ETA.
- Prediction and proof of delivery records Extend the bid and order management process by including capabilities for staged acceptance partial awards and escrow with automated conflict handling.
- Extend payment capabilities to include: multiple providers, digital wallets,

automated, reconciliation, and region-specific payment method

- Improve system reliability and scalability using Redis caching, background processing queue and a completely event-driven architecture.
- Improve the user experience by providing support for offline use, progressive web app features, Push notifications, experimentation frameworks for UX optimisation.
- Enhance the governance mechanisms with vendor verification, rating, and reviews. systems, advanced moderation and richer audit trails.

7.6 Recommendations for Adoption

- Ensure transactional integrity for every multi-record operation, especially during the processing of orders. creation, structure changes of inventory, and updates about the delivery area.
- Define service level objectives for search and order related APIs Monitor these using Dashboards and alerts that monitor latency, error rates, and real-time stability. Monitor these using dashboards and alerts that track latency, error rates and real time stability.
- Use feature flags and staged rollouts to introduce updates safely. Incorporate synthetic monitoring to continuously test critical user paths.

7.7 Closing Remarks

The project demonstrates that a scalable hyperlocal delivery platform can be built using a compact and maintainable technology stack. The system fulfils its core goals of shop discovery, real time tracking and merchant management while meeting performance and usability targets. Although additional development is required in areas such as payments, logistics and advanced analytics, the current architecture provides a strong foundation for future expansion and research into optimisation of hyperlocal commerce, pricing strategies and neighbourhood level logistics.

REFERENCES

[1] S. N. Indah, M. Rusdi, H. Pratiwi, and M. Samiri. The impact of digitalization on traditional markets: Transformation and challenges in the era of e-commerce. *Jurnal Studi Ilmu Pemerintahan*, 5(1):117–134, 2024. Cited on pp. 1, 3, 5, 6, and 7.

[2] I. Khawaja and N. Iqbal. Determinants of expansion of micro and small firms and state of entrepreneurship in pakistan. Technical Report 2019:160, Pakistan Institute of Development Economics, 2019. Cited on pp. 1, 2, 3, 5, 6, and 7.

[3] O. R. Imon. Designing and implementing a well-being ecommerce website, 2024. Cited on pp. 1, 4, and 6.

[4] A. Javed. Prospects and problems for e-commerce in pakistan. *Asian Journal of Economics*, 2020. Cited on pp. 2, 5, and 6.

[5] H. Hansora, S. Bendale, N. Varanmala, and V. Solanki. E-commerce website, 2021. Cited on pp. 2 and 5.

APPENDICES

APPENDIX A: User Manual

Appendices are provided to give supplementary information, which is included in the main text may serve as a distraction and cloud the central theme.

- Appendices should be numbered using alphabets, e.g. Appendix A, Appendix B, etc.
- Tables and References appearing in appendices should be numbered and referred to at appropriate places just as in the case of chapters.
- Appendices shall carry the title of the work reported and the same title shall be written in the contents page.