École Hassania des Travaux Publics

Department of MIG
GIS Specialization

End of Studies Project

Presented by:
Yassine DBAICHI

Book Marketplace Platform with Route Optimization

Academic Supervisor:
Prof. Hatim LECHGAR, EHTP

Academic Year 2024/2025

Résumé

Ce projet présente une plateforme de marketplace de livres avec optimisation automatique des tournées de livraison. Le système répond au défi logistique rencontré par les vendeurs de livres qui doivent livrer plusieurs commandes à différents clients répartis dans la ville.

La plateforme comprend une application web pour l'achat et la vente de livres, et une application mobile pour les vendeurs permettant de planifier des tournées de livraison optimisées. Lorsqu'un client passe commande, son adresse (enregistrée lors de l'inscription) est automatiquement associée à la commande. Le vendeur peut ensuite sélectionner plusieurs commandes dans son application mobile et obtenir l'itinéraire optimal pour visiter tous les clients en minimisant la distance parcourue.

L'application mobile utilise un algorithme du voyageur de commerce (TSP) pour calculer l'ordre de visite optimal, puis intègre OpenRouteService pour fournir des instructions de navigation détaillées. Le vendeur peut suivre l'itinéraire directement dans l'application ou l'exporter vers Google Maps ou Waze.

La plateforme est développée avec React pour le web, Flutter pour le mobile, et Node.js pour le backend, avec une base de données MongoDB Atlas hébergée dans le cloud.

Mots-clés : Optimisation d'Itinéraire, Marketplace, Application Mobile, SIG, Livraison, Problème du Voyageur de Commerce

Abstract

This project presents a book marketplace platform with automatic delivery route optimization. The system addresses the logistical challenge faced by book sellers who must deliver multiple orders to different customers distributed across the city.

The platform includes a web application for buying and selling books, and a mobile application for sellers to plan optimized delivery routes. When a customer places an order, their address (registered during signup) is automatically associated with the order. The seller can then select multiple orders in their mobile application and obtain the optimal route to visit all customers while minimizing travel distance.

The mobile application uses a Traveling Salesman Problem (TSP) algorithm to calculate the optimal visit sequence, then integrates OpenRouteService to provide detailed turn-by-turn navigation instructions. The seller can follow the route directly in the application or export it to Google Maps or Waze.

The platform is developed with React for web, Flutter for mobile, and Node.js for backend, with a MongoDB Atlas database hosted in the cloud.

Keywords: Route Optimization, Marketplace, Mobile Application, GIS, Delivery Management, Traveling Salesman Problem

Acknowledgements

I would like to express my sincere gratitude to Professor Hatim LECHGAR for his valuable guidance and support throughout this project.

I also thank the faculty and staff of the Department of MIG at EHTP for their excellent teaching and resources.

Finally, I extend my appreciation to my family for their continuous encouragement during my studies.

List of Abbreviations

API	Application Programming Interface
APK	Android Package Kit
CDN	Content Delivery Network
EHTP	École Hassania des Travaux Publics
GIS	Geographic Information System
GPS	Global Positioning System
HTTPS	Hypertext Transfer Protocol Secure
JWT	JSON Web Token
MIG	Management of Computer Engineering (Département)
REST	Representational State Transfer
TSP	Traveling Salesman Problem
URL	Uniform Resource Locator

Contents

List of Abbreviations **iv**

1 Introduction **1**
1.1 Context . . . 1
1.2 Problem Statement . . . 1
1.3 Proposed Solution . . . 1
1.4 Objectives . . . 2
1.5 Report Structure . . . 2

2 System Conception **3**
2.1 Functional Requirements . . . 3
2.1.1 Web Application Requirements . . . 3
2.1.2 Mobile Application Requirements . . . 4
2.2 Use Case Analysis . . . 4
2.2.1 Buyer Use Cases . . . 4
2.2.2 Seller Use Cases . . . 5
2.3 Data Model . . . 6
2.3.1 Main Entities . . . 6
2.3.2 Class Diagram . . . 6

3 Application Workflow **7**
3.1 Buyer Workflow . . . 7
3.2 Seller Workflow . . . 8

4 Implementation **9**
4.1 System Architecture . . . 9
4.2 Technologies Used . . . 10
4.2.1 Web Application . . . 10
4.2.2 Mobile Application . . . 10
4.2.3 Backend . . . 10
4.2.4 External Services . . . 10
4.3 Route Optimization . . . 11
4.3.1 How Address Data Flows . . . 11
4.3.2 Route Calculation Algorithm . . . 11
4.3.3 Navigation Options . . . 12
4.4 Deployment . . . 12
4.4.1 Backend Deployment . . . 12
4.4.2 Database Hosting . . . 12
4.4.3 Web Application . . . 12

4.4.4 Mobile Application 13

5 Results **14**
5.1 Web Application 14
5.1.1 Landing Page 14
5.1.2 Book Catalog 15
5.1.3 Registration 16
5.1.4 Order Placement 17
5.2 Mobile Application 18
5.2.1 Login Screen 18
5.2.2 Order List 19
5.2.3 Map View 20
5.2.4 Order Selection 21
5.2.5 Optimized Route 22
5.2.6 Navigation Instructions 23
5.2.7 Navigation Options 24

6 Conclusion **25**
6.1 Achievements 25
6.2 Benefits 25
6.3 Limitations 25
6.4 Future Work 26
6.5 Final Remarks 26

List of Figures

2.1 Buyer Use Case Diagram 5
2.2 Seller Use Case Diagram 5
2.3 System Data Model 6

3.1 Buyer Journey Flowchart 7
3.2 Seller Delivery Workflow 8

4.1 Global System Architecture 9

5.1 Web Application Homepage 14
5.2 Marketplace Book Listing 15
5.3 User Registration with Address Input 16
5.4 Order Form (Address Automatic from Profile) 17
5.5 Mobile App Login 18
5.6 Pending Orders List 19
5.7 Customer Locations on Map 20
5.8 Selecting Orders for Delivery 21
5.9 Calculated Optimal Route 22
5.10 Turn-by-Turn Directions 23
5.11 Export to External Navigation Apps 24

List of Tables

4.1 Web Application Stack 10
4.2 Mobile Application Stack 10
4.3 Backend Stack 10
4.4 External Services 10

Chapter 1

Introduction

1.1 Context

The second-hand book market, particularly among students and academic communities, involves numerous individual sellers operating small-scale businesses. These sellers face a significant logistical challenge: delivering books to multiple customers located at different addresses across the city.

When a seller receives several orders in a day, they must plan a delivery route that visits all customers. Without proper tools, this planning is done manually, often resulting in inefficient routes that waste time and fuel. Additionally, sellers must manually copy customer addresses from their order records into navigation applications like Google Maps or Waze, a tedious and error-prone process.

1.2 Problem Statement

Book sellers encounter three main problems:

Manual Address Handling: Sellers must manually copy each customer's delivery address from order records into navigation applications. This repetitive task is time-consuming and prone to errors.

Inefficient Routes: Without optimization tools, sellers typically visit customers in the order they received the orders, or based on rough geographic estimates. These approaches rarely produce efficient routes, leading to unnecessary travel distance and wasted time.

Lack of Integration: Existing route optimization tools are separate from marketplace platforms. Sellers must use multiple disconnected applications, duplicating effort and creating workflow friction.

1.3 Proposed Solution

This project develops an integrated platform that combines e-commerce marketplace functionality with automatic delivery route optimization. The system works as follows:

- Customers register on the web platform and provide their delivery address once
- When customers place orders, their addresses are automatically attached to the orders

- Sellers access pending orders through a mobile application
- The mobile app displays customer locations on an interactive map
- Sellers select which orders to deliver and the app calculates the optimal route
- The app provides turn-by-turn navigation or exports the route to Google Maps or Waze

The key innovation is the automatic flow of address data from customer registration to optimized route planning, eliminating manual data entry and providing algorithmic route optimization.

1.4 Objectives

The project aims to:

1. Build a web marketplace where users can buy and sell books
2. Automatically capture customer addresses during registration
3. Develop a mobile application for sellers to view and manage orders
4. Implement route optimization to find the best delivery sequence
5. Integrate with navigation services for turn-by-turn directions
6. Deploy the complete system for real-world use

1.5 Report Structure

This report is organized as follows: Chapter 2 presents the system conception including requirements and design. Chapter 3 describes the application workflow. Chapter 4 details the implementation and technologies used. Chapter 5 presents the results through screenshots. Chapter 6 concludes with achievements and future work.

Chapter 2

System Conception

2.1 Functional Requirements

2.1.1 Web Application Requirements

The web platform must provide the following functionality:

User Management:

- Account registration with role selection (buyer or seller)
- Address input during registration (used for order delivery)
- Login and authentication
- Profile management

Marketplace Features:

- Browse available books
- Search by title or author
- View book details (price, condition, seller information)
- Sellers can list books for sale

Order Management:

- Buyers can place orders
- Orders automatically include buyer's registered address
- Order status tracking
- Order history

2.1.2 Mobile Application Requirements

The mobile application (for sellers only) must provide:

Order Access:

- View all pending orders
- See order details (book, buyer, delivery address)
- Filter orders by status

Map Visualization:

- Display all customer locations on an interactive map
- Each order shown as a marker
- View customer details by tapping markers

Route Optimization:

- Select multiple orders for delivery
- Calculate optimal route sequence
- Display total distance and estimated time
- Show route on map with numbered waypoints

Navigation:

- In-app turn-by-turn directions
- Export route to Google Maps
- Export route to Waze

Delivery Management:

- Confirm or refuse orders
- Mark orders as delivered
- Add notes to orders

2.2 Use Case Analysis

2.2.1 Buyer Use Cases

Figure 2.1 illustrates the main use cases for buyers.

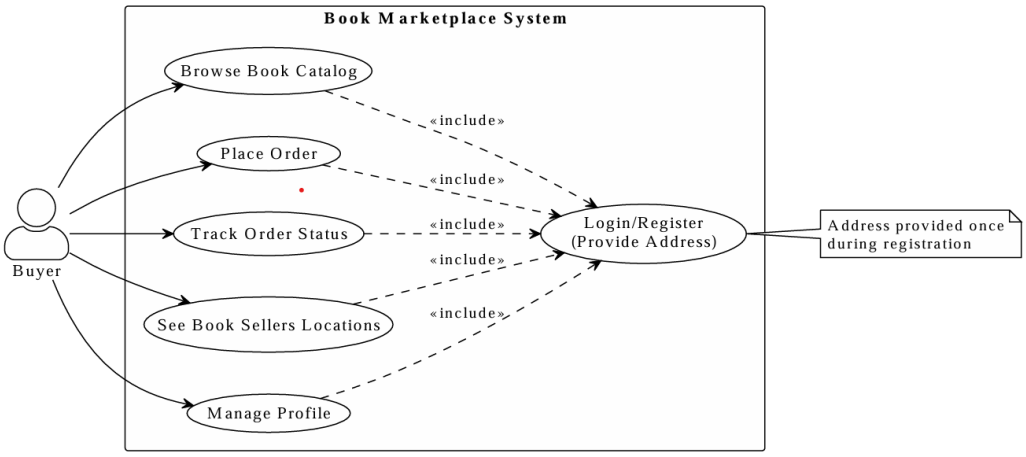

Figure 2.1: Buyer Use Case Diagram

The key aspect is that buyers provide their address once during registration, which is then automatically used for all future orders.

2.2.2 Seller Use Cases

Figure 2.2 illustrates the main use cases for sellers.

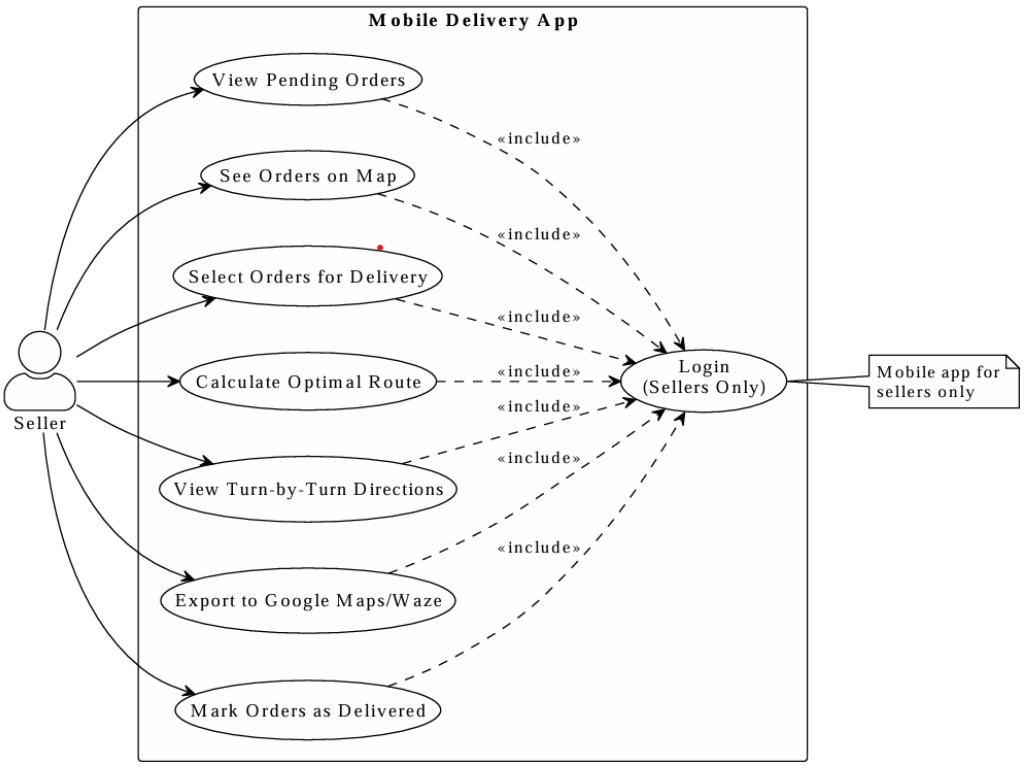

Figure 2.2: Seller Use Case Diagram

The core value for sellers is automatic route optimization without manual address entry.

2.3 Data Model

2.3.1 Main Entities

The system manages four primary entities:

User: Represents both buyers and sellers. Key attributes include email, password, role (buyer or seller), and address. The address is captured once during registration and stored permanently in the user profile.

Book: Represents items in the marketplace. Attributes include title, author, price, quantity available, condition, and reference to the seller who owns it.

Order: Links a buyer, seller, and book. Contains quantity, total price, and status. Critically, each order automatically includes the buyer's delivery address (copied from their user profile).

GeoJSON Export: Optional entity for exporting route data.

2.3.2 Class Diagram

Figure 2.3 presents the data model.

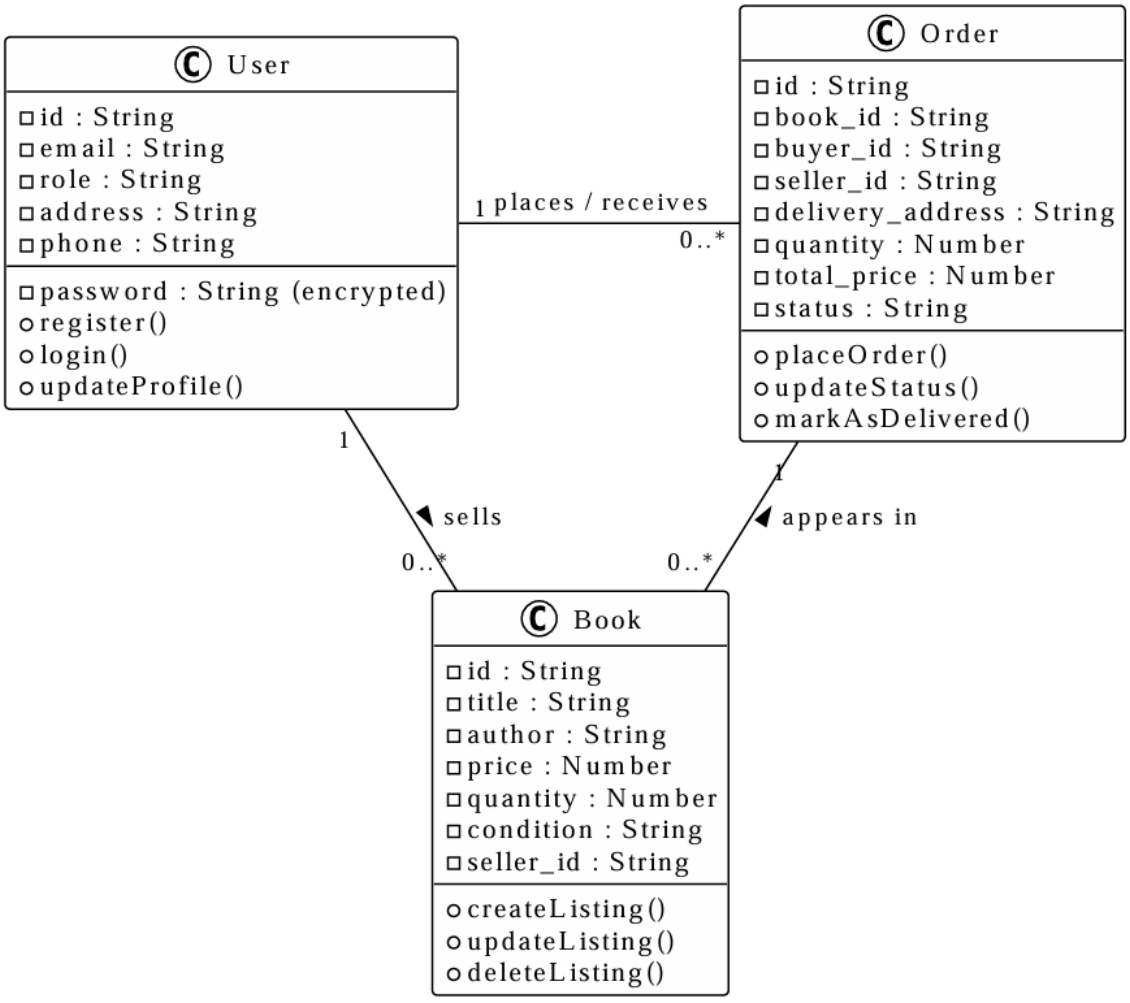

Figure 2.3: System Data Model

Chapter 3

Application Workflow

This chapter describes the complete user journey from account creation to delivery completion.

3.1 Buyer Workflow

Figure 3.1 illustrates the buyer's journey.

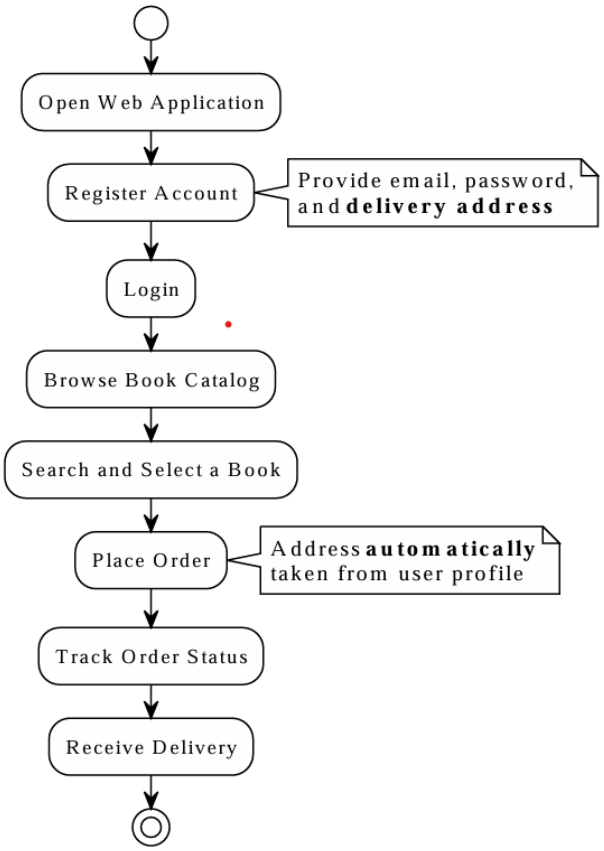

Figure 3.1: Buyer Journey Flowchart

The important point is that the address is provided only once during registration and then automatically reused for all orders.

3.2 Seller Workflow

Figure 3.2 illustrates the seller's delivery journey.

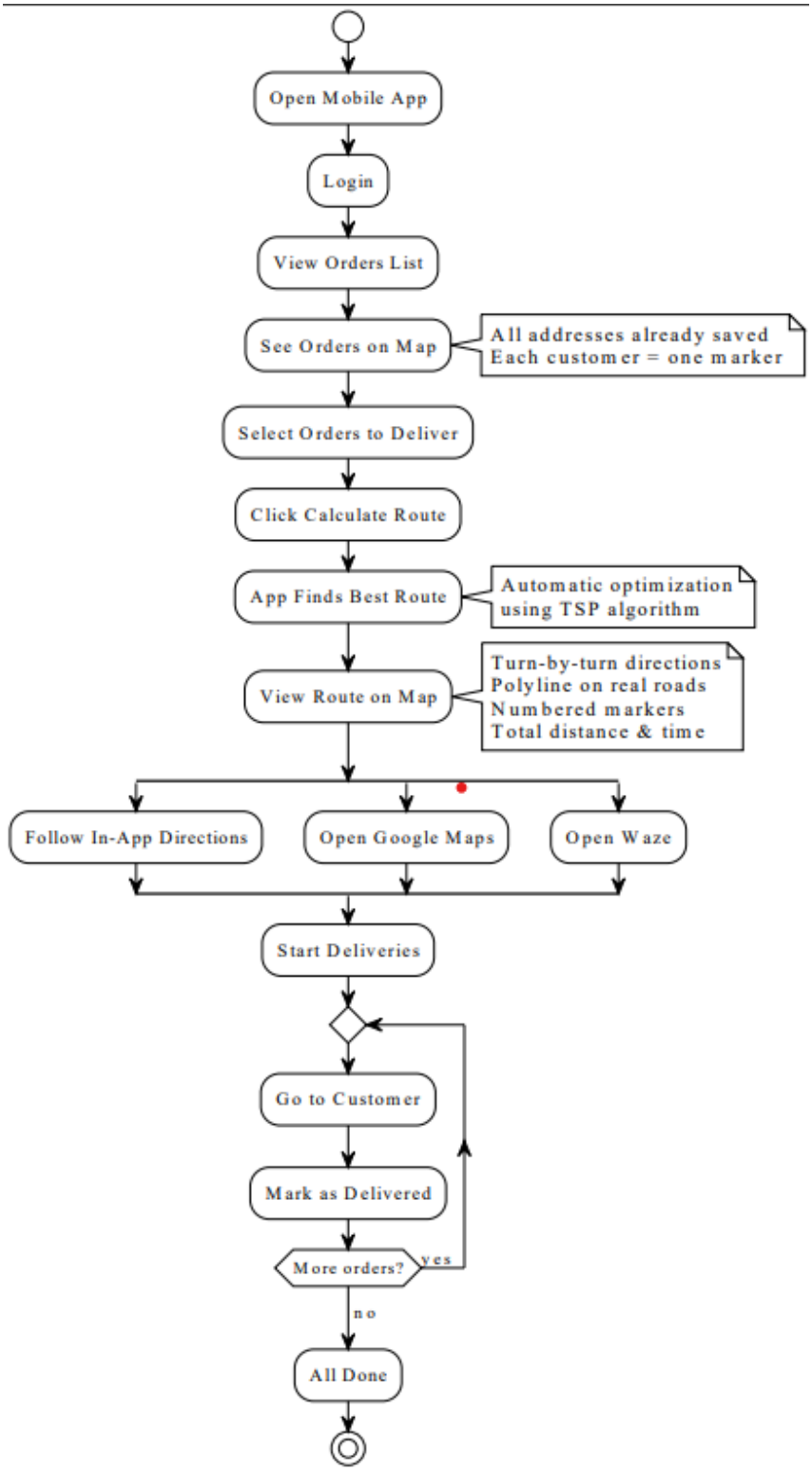

Figure 3.2: Seller Delivery Workflow

Chapter 4

Implementation

4.1 System Architecture

The platform uses a three-tier architecture:

Frontend Layer:

- Web application (React) for marketplace operations
- Mobile application (Flutter) for delivery management

Backend Layer:

- REST API server (Node.js + Express)
- Handles authentication, orders, and route calculations

Data Layer:

- MongoDB database for storing users, books, and orders
- Cloud-hosted on MongoDB Atlas

Communication between components uses HTTPS with JWT token authentication.

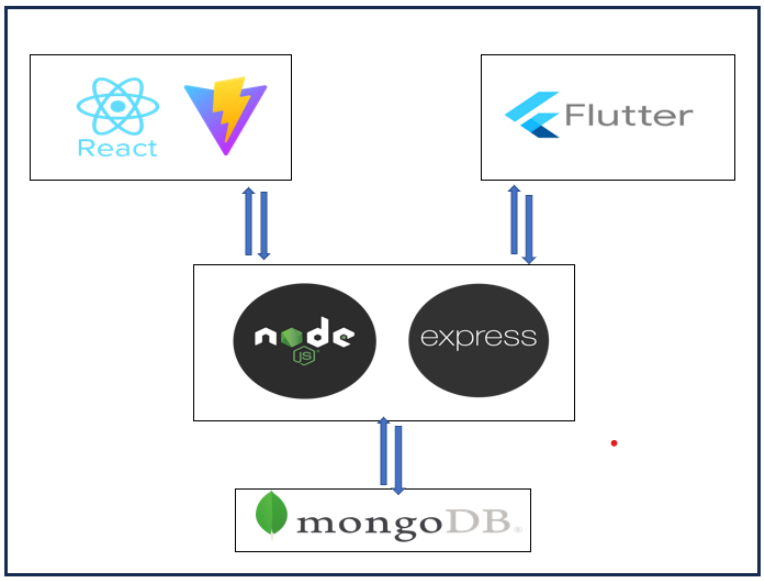

Figure 4.1: Global System Architecture

4.2 Technologies Used

4.2.1 Web Application

Component	Technology
Framework	React 19.1.1
UI Library	React Bootstrap + Bootstrap 5
Maps	React Leaflet
HTTP Client	Axios
Build Tool	Vite

Table 4.1: Web Application Stack

4.2.2 Mobile Application

Component	Technology
Framework	Flutter 3.24.0
Maps	flutter_map + OpenStreetMap
GPS	geolocator
HTTP	dio
Storage	flutter_secure_storage
Platform	Android

Table 4.2: Mobile Application Stack

4.2.3 Backend

Component	Technology
Runtime	Node.js 18.x
Framework	Express.js
Database	MongoDB (Atlas)
Authentication	JWT (jsonwebtoken)
Hosting	Vercel

Table 4.3: Backend Stack

4.2.4 External Services

Service	Purpose
OpenRouteService	Route calculation and turn-by-turn directions
OpenStreetMap	Map tiles for visualization
Google Maps	External navigation option
Waze	External navigation option

Table 4.4: External Services

4.3 Route Optimization

4.3.1 How Address Data Flows

The automatic address flow works as follows:

1. User registers on web platform and provides their address
2. Address is stored in the User entity in the database
3. When user places an order, the system automatically copies their address to the Order entity
4. Seller opens mobile app and retrieves orders via API
5. Each order already contains the delivery address
6. Seller selects orders and the app extracts addresses for route planning

No manual address entry is required at any step.

4.3.2 Route Calculation Algorithm

When the seller selects multiple orders for delivery, the mobile app performs the following steps:

Step 1: Extract Locations

The app retrieves the delivery address (latitude and longitude) from each selected order.

Step 2: Apply TSP Algorithm

The app uses the Traveling Salesman Problem nearest neighbor heuristic:

1. Start at the seller's current GPS location
2. Find the nearest unvisited customer
3. Add that customer to the route
4. Move to that customer's location
5. Repeat until all customers are visited

This produces an optimized sequence that minimizes total travel distance.

Step 3: Get Turn-by-Turn Directions

The optimized waypoint sequence is sent to OpenRouteService API, which returns:

- Complete route geometry (the actual path on roads)
- Total distance in kilometers
- Estimated travel time
- Step-by-step turn-by-turn navigation instructions

Step 4: Display Route
The mobile app displays:

- The route drawn as a line on the map
- Numbered markers showing the visit sequence
- List of navigation instructions
- Total distance and time

4.3.3 Navigation Options

The seller can choose how to follow the route:
Option 1: In-App Navigation
Follow the turn-by-turn directions displayed directly in the mobile application.
Option 2: Google Maps
The app constructs a Google Maps URL with all waypoints and opens Google Maps app.
Option 3: Waze
The app constructs a Waze URL with all waypoints and opens Waze app.
This flexibility allows sellers to use their preferred navigation tool while still benefiting from the optimized route sequence.

4.4 Deployment

4.4.1 Backend Deployment

The Node.js backend is deployed on Vercel serverless platform with:

- Automatic scaling
- Global CDN distribution
- HTTPS encryption
- Continuous deployment from Git repository

4.4.2 Database Hosting

MongoDB database is hosted on MongoDB Atlas cloud platform with:

- Automatic backups
- Encrypted connections
- High availability

4.4.3 Web Application

React web application is deployed as static files on Vercel with CDN caching.

4.4.4 Mobile Application

Flutter mobile app is compiled to Android APK for distribution.

Chapter 5

Results

This chapter presents screenshots of the implemented system.

5.1 Web Application

5.1.1 Landing Page

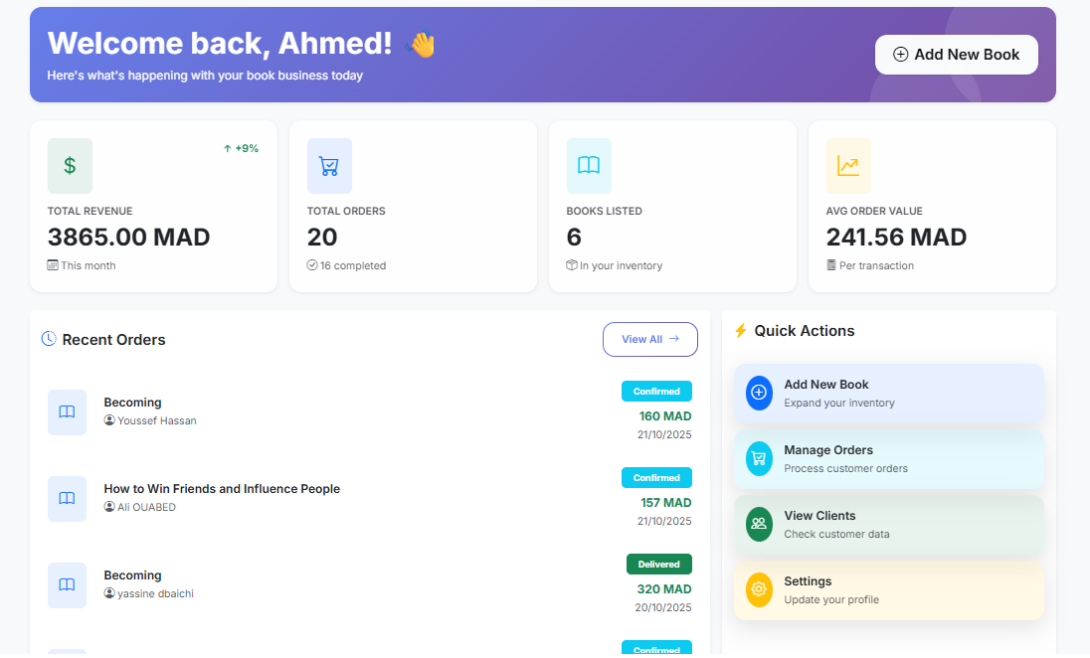

Figure 5.1: Web Application Homepage

5.1.2 Book Catalog

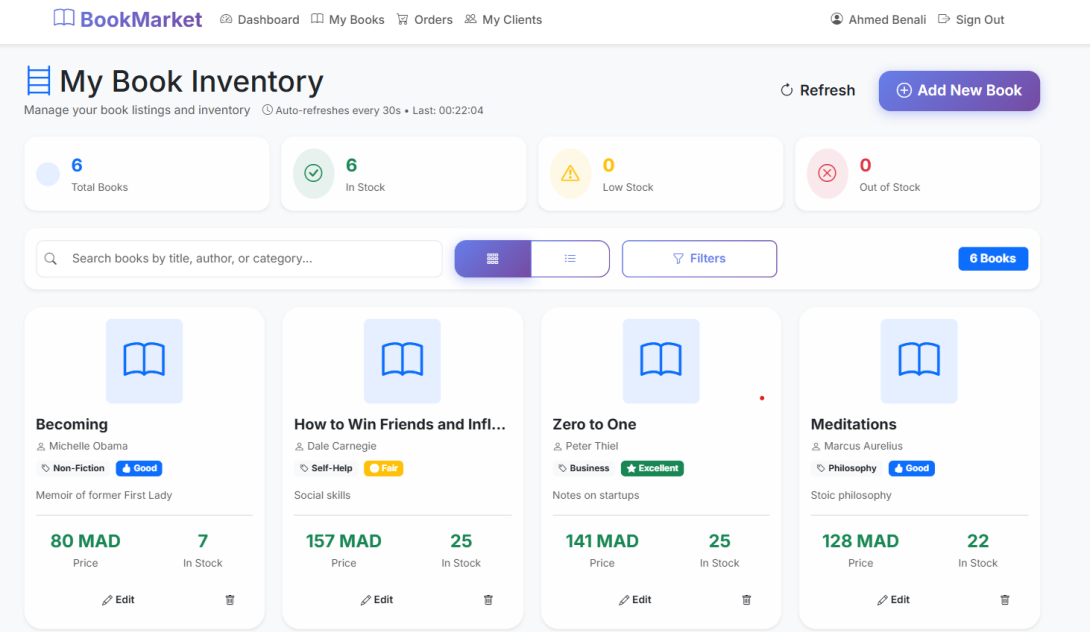

Figure 5.2: Marketplace Book Listing

5.1.3 Registration

Figure 5.3: User Registration with Address Input

5.1.4 Order Placement

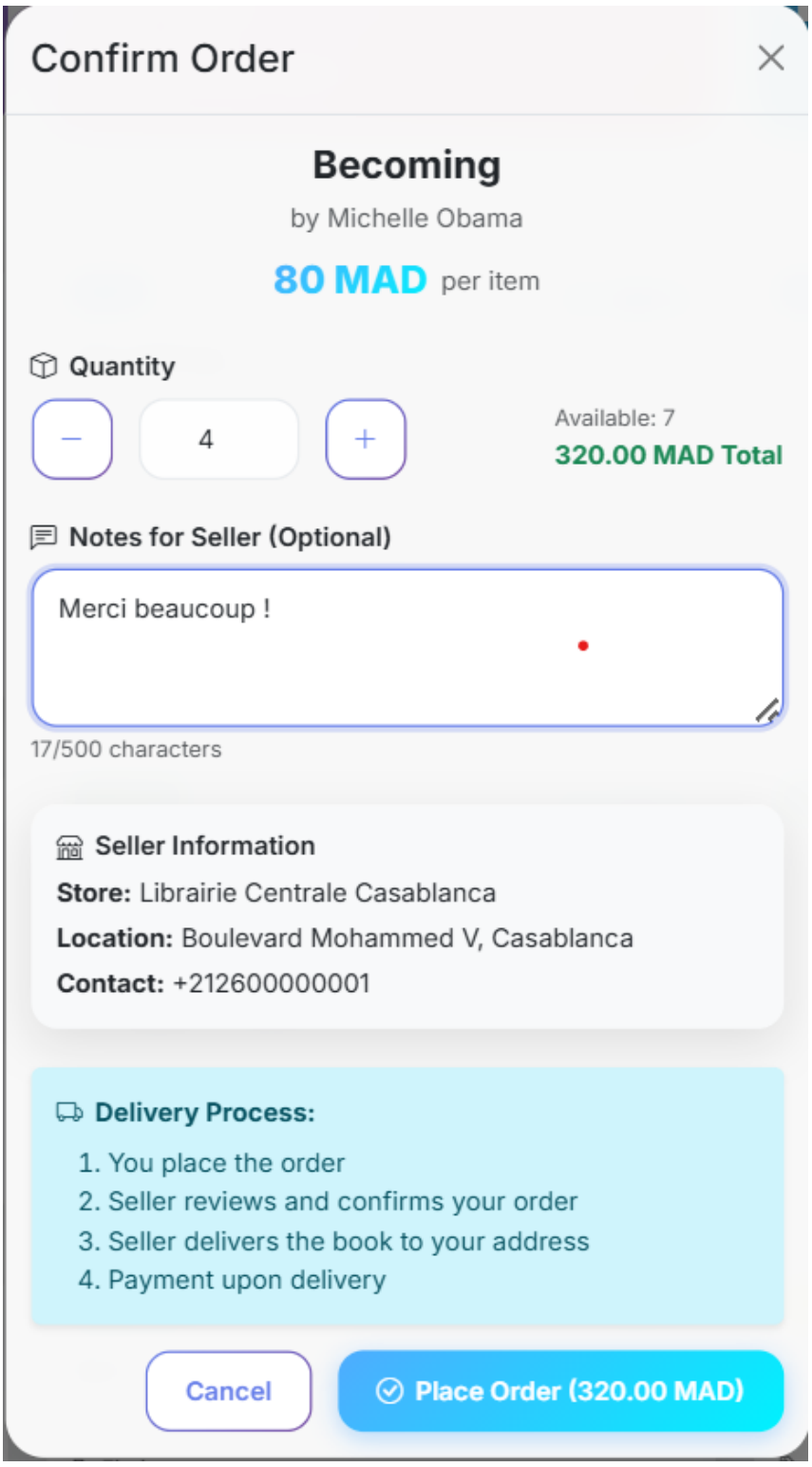

Figure 5.4: Order Form (Address Automatic from Profile)

5.2 Mobile Application

5.2.1 Login Screen

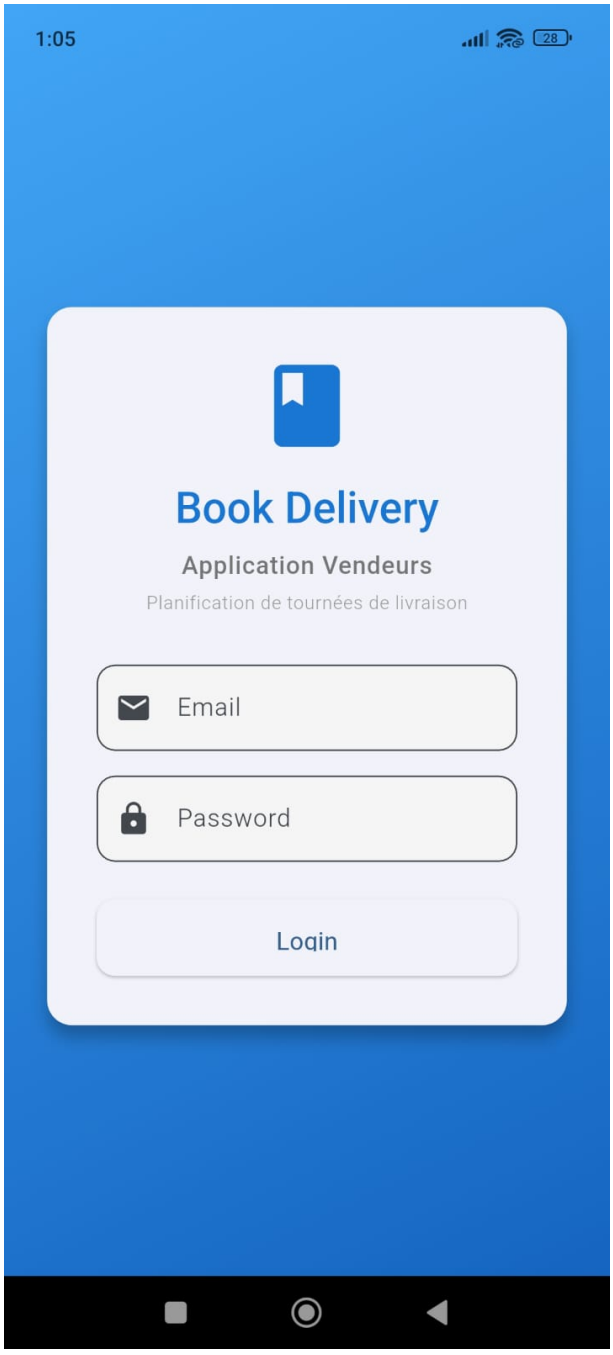

Figure 5.5: Mobile App Login

5.2.2 Order List

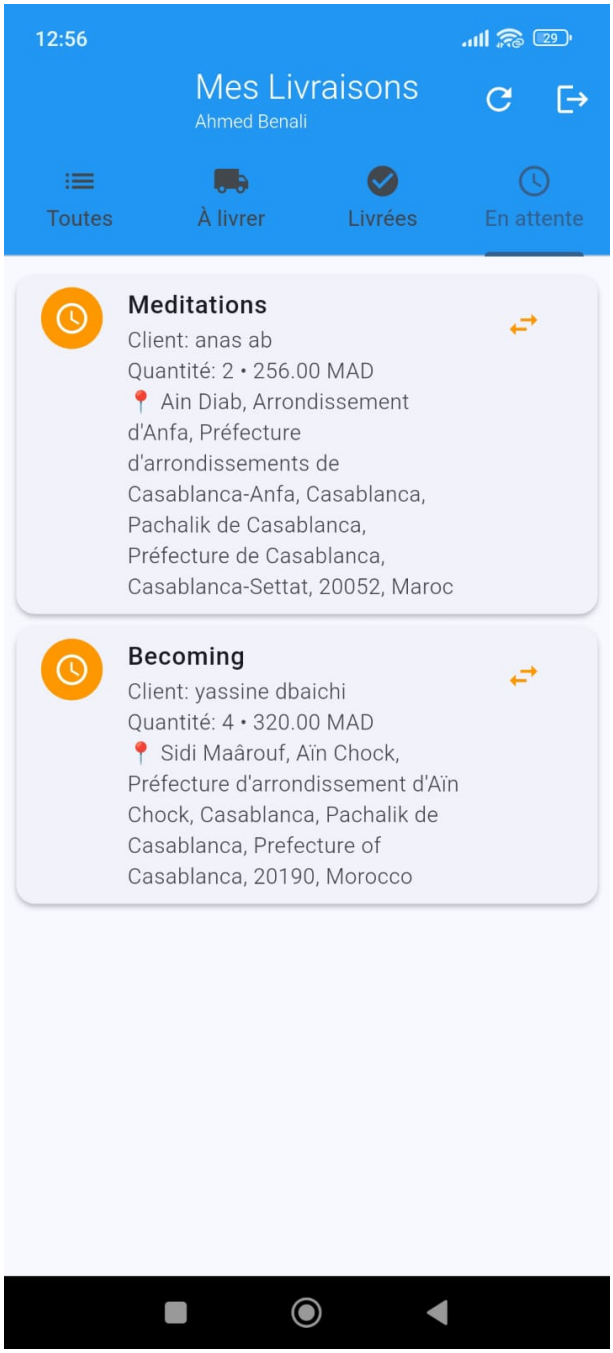

Figure 5.6: Pending Orders List

5.2.3 Map View

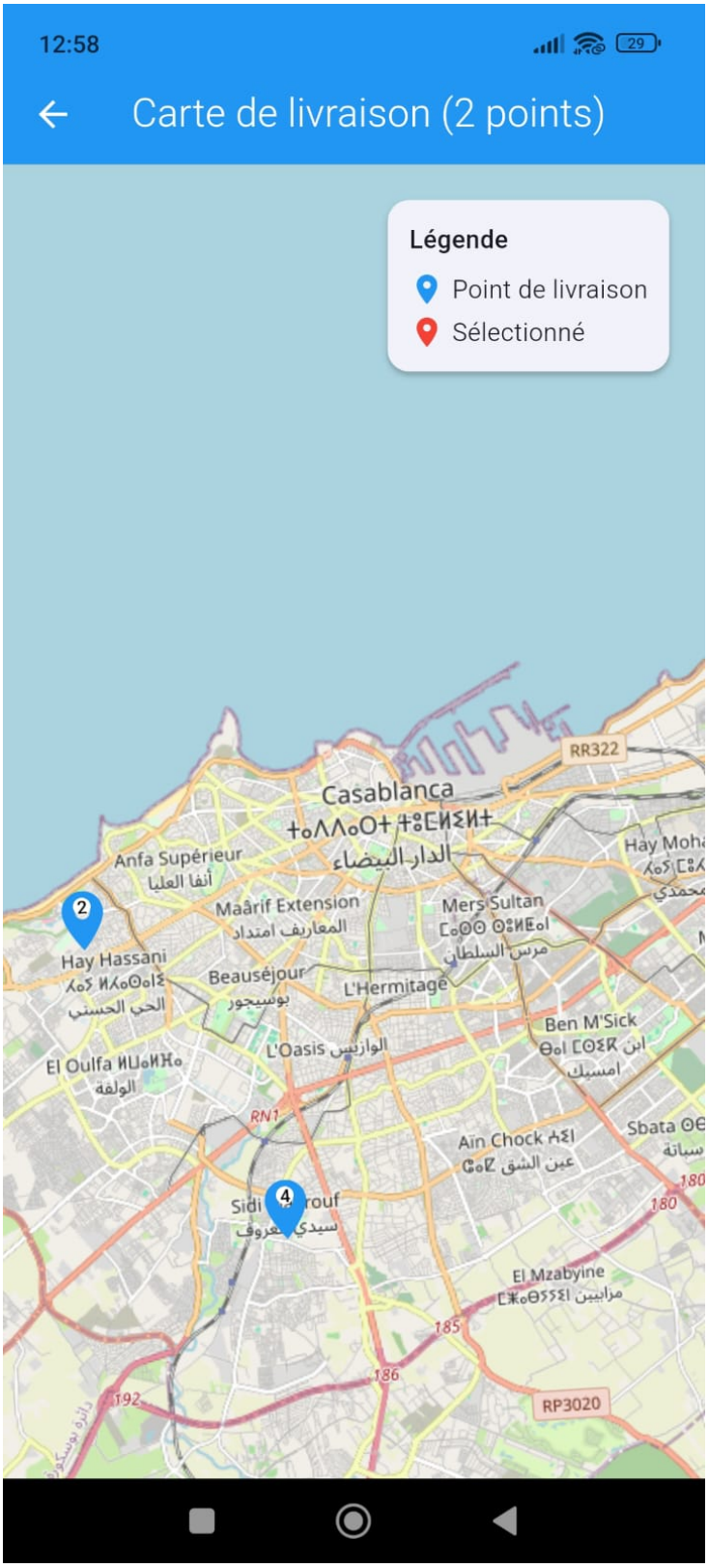

Figure 5.7: Customer Locations on Map

5.2.4 Order Selection

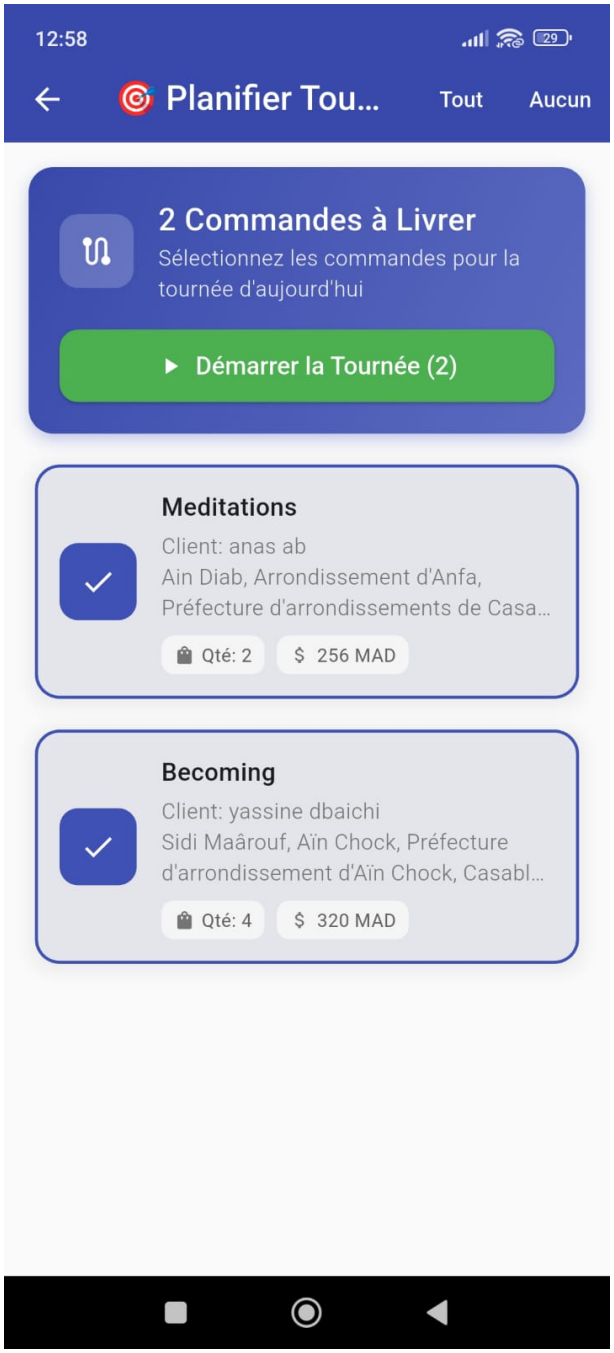

Figure 5.8: Selecting Orders for Delivery

5.2.5 Optimized Route

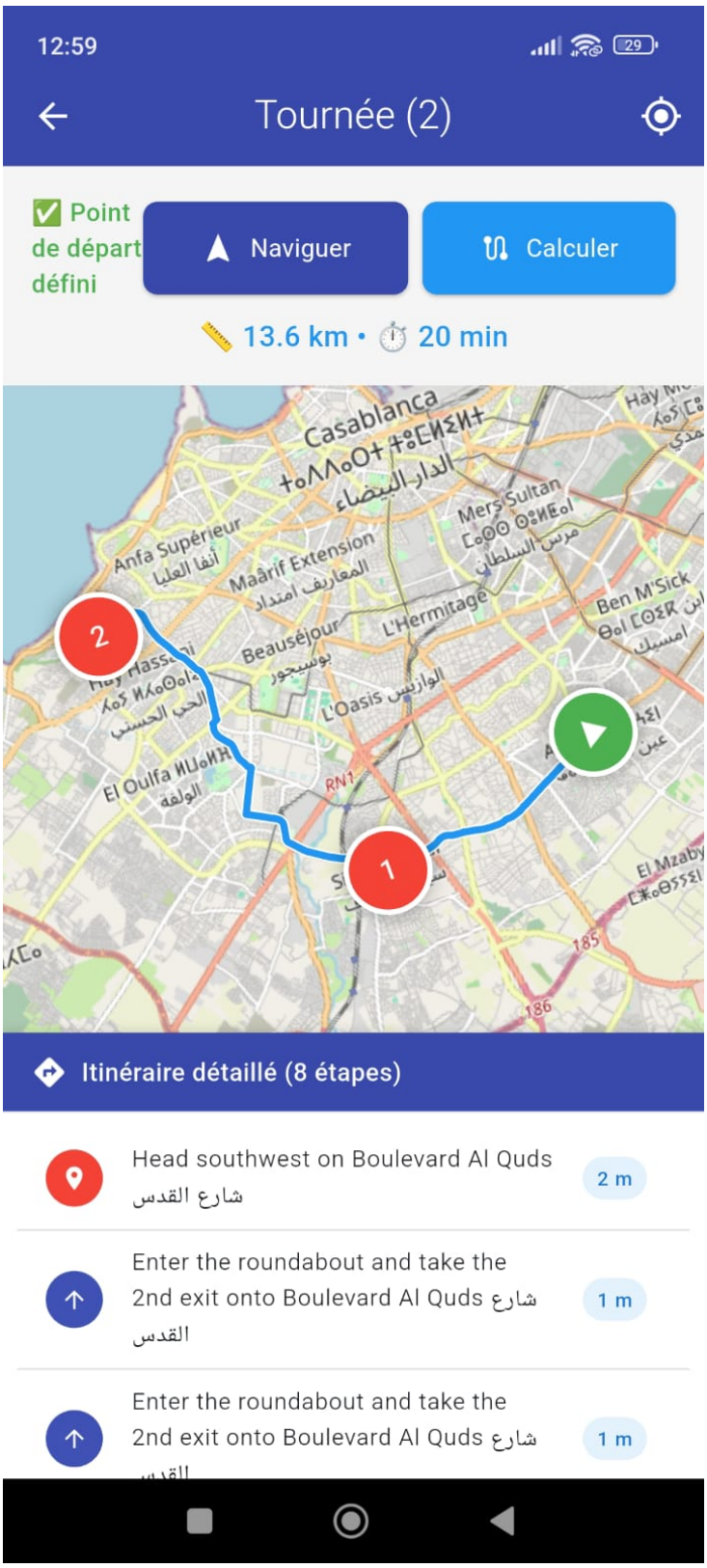

Figure 5.9: Calculated Optimal Route

5.2.6 Navigation Instructions

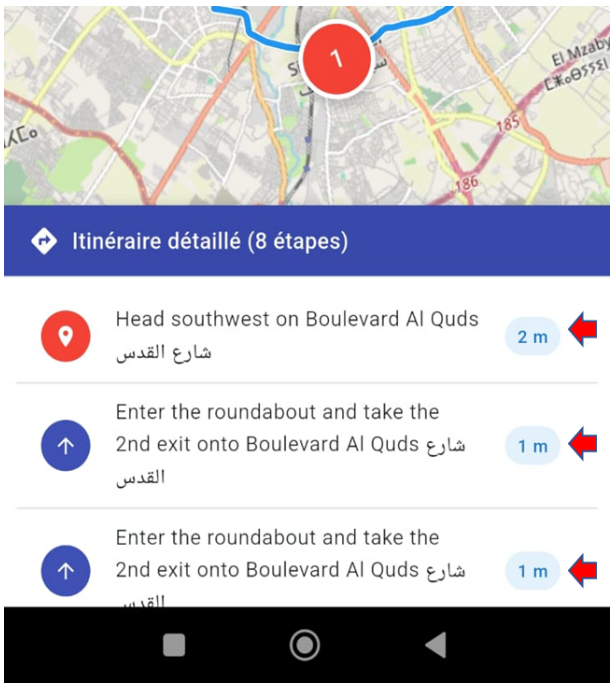

Figure 5.10: Turn-by-Turn Directions

5.2.7 Navigation Options

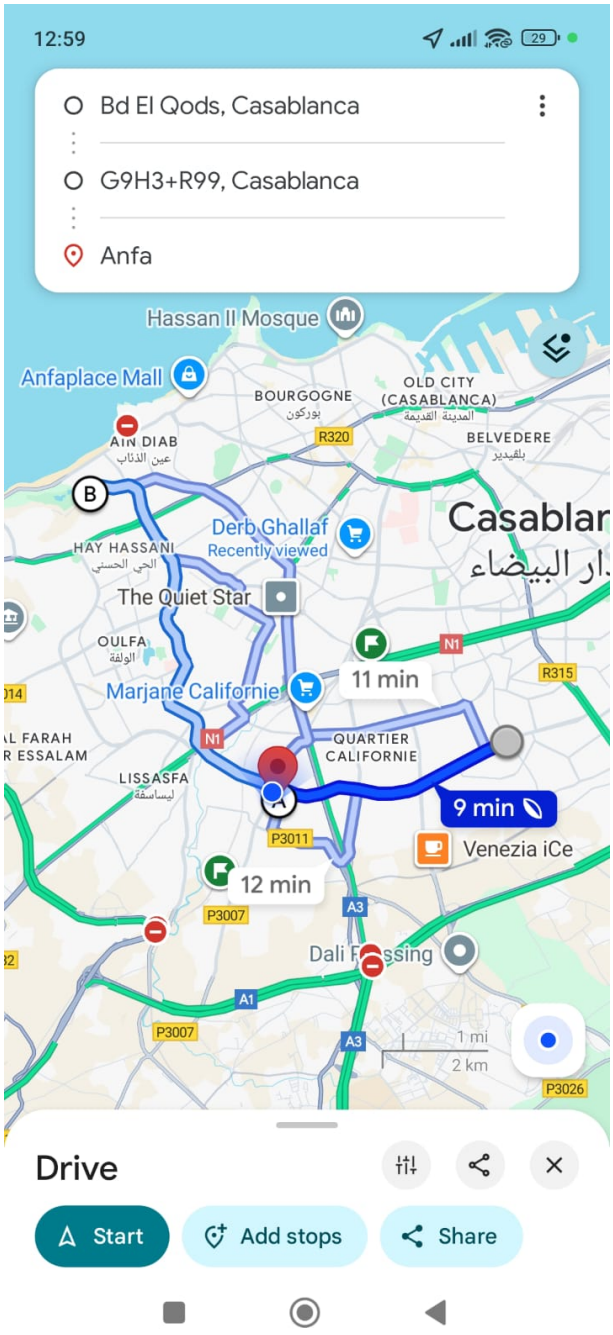

Figure 5.11: Export to External Navigation Apps

Chapter 6

Conclusion

6.1 Achievements

This project successfully delivered an integrated book marketplace platform with automatic delivery route optimization. The key achievements include:

Automated Address Management: Customer addresses are captured once during registration and automatically flow to orders and route planning, eliminating repetitive manual data entry.

Route Optimization: Implementation of the Traveling Salesman Problem algorithm provides sellers with optimized delivery sequences, reducing travel distance and time.

Flexible Navigation: Integration with OpenRouteService, Google Maps, and Waze gives sellers multiple options for following their optimized routes.

Production Deployment: The complete system is deployed and operational on cloud infrastructure.

6.2 Benefits

The platform provides concrete benefits:

- Sellers save time by not manually entering addresses
- Optimized routes reduce fuel costs and delivery time
- Customers benefit from faster deliveries
- The integrated system eliminates workflow friction

6.3 Limitations

Current limitations include:

- Mobile app only supports Android (iOS not yet implemented)
- Route calculation requires internet connectivity
- Algorithm provides good but not perfect solutions

6.4 Future Work

Potential improvements include:

iOS Support: Extend the mobile application to iOS devices using Flutter's cross-platform capabilities.

Manual Route Adjustment: Allow sellers to manually reorder waypoints based on local knowledge or real-time constraints.

Offline Capability: Add offline map caching and route calculation for areas with poor connectivity.

Advanced Optimization: Implement more sophisticated algorithms for better route quality.

6.5 Final Remarks

This project demonstrates that integrating e-commerce with spatial optimization creates tangible value for small-scale logistics operations. By automating the flow of geographic data from customer registration to optimized route planning, the platform reduces manual work and improves delivery efficiency.

Webography

[1] OpenStreetMap Contributors. *OpenStreetMap*. https://www.openstreetmap.org

[2] HeiGIT gGmbH. *OpenRouteService - Route Optimization API*. https://openrouteservice.org

[3] Google LLC. *Flutter - Build apps for any screen*. https://flutter.dev

[4] MongoDB, Inc. *MongoDB Atlas - Cloud Database*. https://www.mongodb.com/atlas

[5] Vercel Inc. *Vercel - Develop. Preview. Ship*. https://vercel.com