AutoSpare - Car Spare Parts E-Commerce Mobile Application

Project Documentation

---

1. Project Planning & Management

Project Proposal

AutoSpare is a Flutter-based mobile application designed to improve the car spare parts industry by providing an integrated e-commerce platform. The system connects buyers, sellers, and tow truck operators within a unified system. The platform simplifies the challenge of finding and purchasing car spare parts while offering emergency tow truck services for vehicle breakdowns.

Project Objectives:

- Create a user-friendly mobile marketplace for car spare parts
- Implement role-based access control for four user types
- Enable authentication and authorization using Firebase
- Provide real-time order tracking and management
- Integrate location-based tow truck services with distance calculation
- Establish a moderation system for product and seller approval

Project Scope:

The application includes full e-commerce features including product browsing, cart management, checkout, order lifecycle tracking, and administrative controls. Additionally, it features a specialized tow truck service module with geolocation integration.

Project Plan

Timeline and Milestones:

- **Week 1-2:** Brainstorming & Planning
- **Week 2-5:** Setup, Logo, UI/UX
- **Week 5-8:** Scenarios, Features and Workflow
- **Week 8-10:** Firebase Integration and Authentication
- **Week 10-**11: Manual Testing
- **Week 11-12:** Documentation, Presentation

Resource Allocation:

- Technology Stack: Flutter 3.32.7, Dart 3.8.1, Firebase
- Development Tools: Android Studio, Visual Studio Code, Git
- Testing Devices: Android & IOS

Task Assignment & Roles

Team Roles:

- **Ahmed Tharwat Ahmed:** Project Lead, Scenario Designing, UI
- **Ahmed Mostafa Almazon:** UI
- **Ziad Mohamed Rafat:** Firebase Setup, Backend Integration
- **Salma Yousry Abdel Razek:** Documentation & QA
- **Omar Abdelalim Abdelfattah:** UI
- **Freddy Rafik Wahib:** UI, GitHub

Risk Assessment & Mitigation Plan

Risk Assessment:

- **Users unable to easily find/buy spare parts online (Medium Impact):** Product catalog with search/filtering, multiple sellers
- **Users unable to order tow trucks from nearest available operator with transparent pricing (High Impact):** Real-time location matching, distance-based cost calculator
- **Sellers unable to manage orders/inventory effectively (Medium Impact):** Role-specific dashboards

Key Performance Indicators (KPIs)

- Response Time: Average page load under 2 seconds
- System Uptime: 24/7 available
- Submission Rating: User experience with rating system
- Tow Request Response Time: Average time to match with customer
- Completion of key workflows (order placement, product approval, tow request)
- Coverage of all 4 user types
- Successful transitions between all screens/features

---

2. Lecturer Review

Feedback & Evaluation

The project shows solid technical implementation with a complete set of features for all user roles. Its use of Firebase services reflects a practical understanding of cloud-based backend solutions. The tow truck service module adds a unique value that goes beyond typical e-commerce applications.

Suggested Improvements

- Add push notification system for order updates
- Create web-based admin dashboard for better management
- Add promo code and discount system
- Add payment gateways
- Enhance security with two-factor authentication

---

3. Requirements Gathering

Stakeholder Analysis

Stakeholder Needs:

- **Buyers:** Easy product search, checkout, order tracking
- **Sellers:** Product management, inventory control, order fulfillment
- **Administrators:** System oversight, moderation tools
- **Tow Operators:** Request management, availability control

User Stories & Use Cases

Buyer User Stories:

- Search for spare parts by car brand and model
- Add products to cart and checkout
- Track order status in real-time
- Rate products and sellers after delivery
- Request tow truck service if car breaks down

Seller User Stories:

- Add new products with images and specifications
- View and process incoming orders
- Update order status throughout fulfillment
- View customer reviews of seller's products

Admin User Stories:

- Approve or reject seller registration requests
- Moderate product listings before they go online
- Approve tow company registrations
- Monitor all system orders
- Manage accounts and permissions
- View app profit
- Ban account in case they violated policy

Tow Operator User Stories:

- Switch between online and offline status
- Receive tow requests with location details
- View distance and estimated cost

Functional Requirements

FR1: User Authentication & Authorization

- Users can register with email and password
- Four role types: Buyer, Seller, Admin, Tow Operator
- Firebase Authentication integration
- Role-based access control through the app

FR2: Product Catalog Management

- Products organized by car brand
- Products categorized by car model and compatible years
- Search and advanced filtering capabilities
- Image upload and display for products

FR3: Shopping Cart & Checkout

- Add/remove products to cart
- Quantity management with stock validation
- Checkout process with delivery location selection
- Order creation with unique saving code

FR4: Order Lifecycle Management

- Order states: Created, Processing, Prepared, Handed to Courier, Delivered, Cancelled
- Status updates tracked with timestamps
- Buyer can view order history and current status
- Seller can manage orders for their products

FR5: Review & Rating System

- Buyers can rate products after delivery
- Buyers can rate sellers
- Five-star rating system with optional comments
- Aggregate ratings displayed on product pages

FR6: Tow Truck Service

- Tow companies register with license documentation
- Tow operators manage online/offline availability
- Buyers request tow service with pickup location
- Real-time distance calculation and cost estimation
- Request-response system between buyer and operator

FR7: Admin Moderation

- Approve/reject seller account requests
- Approve/reject product listings
- Approve/reject tow company registrations
- View and manage all system orders
- Access to financial dashboard
- Ability to ban users in case of violating policy

FR8: Location Services

- Map integration for delivery address selection
- Geolocation for tow service pickup points
- Distance calculation between locations

Non-Functional Requirements

NFR1: Performance

- Application response time under 2 seconds for standard operations
- Smooth scrolling and navigation

NFR2: Security

- Secure password storage using Firebase Authentication
- Role-based authorization checks on all sensitive operations
- Data validation on all user inputs

NFR3: Usability

- Navigation following Material Design 3 guidelines
- Arabic language support with RTL layout
- Consistent color scheme with green theme
- Clear visual feedback for user actions

NFR4: Reliability

- Error handling with user-friendly messages
- Data persistence across app restarts
- Network failure recovery mechanisms

NFR5: Maintainability

- Clean code architecture following MVC pattern
- Modular code structure with separation of concerns
- Meaningful variable and function names
- Repository pattern for data access abstraction

NFR6: Scalability

- Firebase backend scales with user growth
- Efficient data queries with pagination support
- Optimized image storage and retrieval

---

4. System Analysis & Design

Problem Statement & Objectives

Problem Statement:

Car owners face significant challenges when seeking spare parts: difficulty verifying authenticity, limited availability information, lack of price transparency, and absence of emergency assistance when vehicles break down. Traditional spare parts markets lack integrated digital solutions that connect buyers, sellers, and service providers efficiently.

Project Goals:

1. Create a trusted marketplace for authentic car spare parts
2. Streamline the buying and selling process through mobile
3. Implement quality control through admin moderation
4. Provide emergency tow services integrated with the platform
5. Enable transparent pricing and customer feedback systems

Use Case Diagram & Descriptions

Primary Actors:

- Buyer
- Seller
- Administrator
- Tow Operator

Key Use Cases:

Buyer Use Cases:

- Register/Login to system
- Browse product catalog
- Search and filter products
- View product details
- Add products to cart
- Checkout and place order
- Track order status
- Rate products and sellers
- Request tow service
- View tow companies

Seller Use Cases:

- Register as seller with documentation
- Add new products (pending approval)
- Manage product inventory
- View incoming orders
- Update order status
- View product reviews

Admin Use Cases:

- Review seller applications
- Approve/reject products
- Approve/reject tow companies
- Monitor all orders
- Manage user accounts
- View system analytics

Tow Operator Use Cases:

- Register tow company with documents
- Toggle online/offline status
- Receive tow requests
- View request details and distance
- Accept/decline tow requests

Software Architecture

Architecture Style: Model-View-Controller (MVC)

High-Level Architecture:

Architecture Layers:

- **Presentation (View):** Flutter UI widgets, screens, themes, custom widgets
- **Controller:** Navigation logic, user session
- **Business Logic (Model):** Data models (AppUser, CatalogProduct, Order, Review)
- **Data Access (Services):** Repositories, Firebase integration
- **Core :** App Fees, Earning Calculations
- **External Services:** Firebase Auth, Firestore, Storage, Geolocator

Key Architectural Components:

1. **View Layer (lib/view):** Contains all UI components including screens, widgets, and themes. Implements Material Design 3 with green color scheme.
2. **Model Layer (lib/model):** Defines data structures for users, products, orders, cart items, and reviews.
3. **Controller Layer (lib/controller):** Manages navigation between screens and coordinates user interactions.
4. **Service Layer (lib/services):** Implements repository pattern for data access, user session management, cart operations, order processing, and tow service directory.
5. **Firebase Integration:** Centralized through service layer providing authentication, cloud storage, and Firestore database access.

Database Design & Data Modeling

Entity-Relationship Diagram:

Core Entities:

1. **AppUser**
    - id (Primary Key)
    - name, email, phone, address, password
    - role (buyer/seller/admin/winch)
    - approved, canSell, canTow flags
    - Store information (storeName, commercialRegUrl, taxCardUrl)
    - Tow documentation (towLicenseUrl, towDriverIdUrl)
2. **CatalogProduct**
    - id (Primary Key)
    - title, seller (Foreign Key to AppUser)
    - price, stock, imageUrl
    - brand (enum), model, compatible years
    - createdAt timestamp
3. **OrderDoc**
    - id (Primary Key)
    - code (unique tracking number)
    - buyerId (Foreign Key to AppUser)
    - items (collection of OrderItem)
    - itemsTotal, shipping, grandTotal
    - lat, lng (delivery location)
    - status (enum), timestamps

4. **OrderItem**
    - productId (Foreign Key to CatalogProduct)
    - sellerId (Foreign Key to AppUser)
    - titleSnap (product name snapshot)
    - price, qty
5. **Review**
    - id (Primary Key)
    - buyerId (Foreign Key to AppUser)
    - productId (Foreign Key to CatalogProduct)
    - sellerId (Foreign Key to AppUser)
    - rating (1-5 stars)
    - comment (optional text)
    - createdAt timestamp
6. **CartItem**
    - productId (Foreign Key)
    - qty (quantity)

Relationships:

- User (Seller) 1:M CatalogProduct
- User (Buyer) 1:M OrderDoc
- OrderDoc 1:M OrderItem
- OrderItem M:1 CatalogProduct
- User (Buyer) 1:M Review
- CatalogProduct 1:M Review

Logical Schema:

The application uses Firebase Firestore as the primary database. Collections are organized as follows:

- users collection: Stores AppUser documents
- products collection: Stores CatalogProduct documents
- orders collection: Stores OrderDoc documents with embedded OrderItem arrays
- reviews collection: Stores Review documents

Local in-memory storage is used for cart management and catalog caching during runtime.

Data Flow & System Behavior

Data Flow Diagram - Level 0 (Context):

External entities: Buyer, Seller, Admin, Tow Operator, Firebase Backend

Data flows:

- User credentials flow to Authentication System
- Product data flows between users and Product Management
- Order data flows through Order Processing System
- Location data flows to Tow Service System

Data Flow Diagram - Level 1 (Detailed):

Product Browsing Flow:

1. User opens app, authentication checked
2. Home screen loads product catalog from Firestore
3. Products filtered by brand/model/year
4. User selects product, details retrieved
5. Product displayed with seller info and reviews

Order Creation Flow:

1. User adds products to cart (local storage)
2. Cart validates stock availability
3. User proceeds to checkout
4. Delivery location selected via map
5. Order created with unique code
6. Stock deducted from products
7. Order saved to Firestore
8. Buyer and sellers notified

Order Fulfillment Flow:

1. Seller views pending orders
2. Seller updates status to "Processing"
3. Seller prepares items, updates to "Prepared"
4. Seller hands to courier, updates status
5. Courier delivers, status updated to "Delivered"
6. Buyer can now submit reviews

Tow Request Flow:

1. Buyer requests tow service
2. Current location captured via GPS
3. Available tow companies retrieved
4. Buyer selects company and confirms location
5. Distance calculated between locations
6. Cost estimated based on distance

7. Request sent to tow operator
8. Operator accepts/declines request

Sequence Diagrams:

User Authentication Sequence:

1. User enters credentials
2. App sends to Firebase Auth
3. Firebase validates and returns token
4. App retrieves user profile from Firestore
5. Session established, navigate to home

Add to Cart Sequence:

1. User views product details
2. User clicks "Add to Cart"
3. App checks stock availability
4. Cart service adds item locally
5. UI updates to show cart count
6. Success message displayed

Admin Approval Sequence:

1. Seller submits product
2. Product saved with approved=false
3. Admin views pending products
4. Admin reviews product details
5. Admin approves/rejects
6. Product status updated in Firestore
7. Seller notified of decision

Activity Diagram

Order Processing Activity:

- Start: Order created by buyer
- Parallel activities: Notify seller, Update inventory
- Seller decision: Accept or Cancel order
- If accepted: Prepare items → Hand to courier → Deliver
- If cancelled: Refund process → End
- After delivery: Enable review submission
- End: Order completed

Tow Request Activity:

- Start: Buyer needs tow service
- Get current location
- Display available companies
- Select company and confirm pickup location
- Calculate distance and cost
- Send request to operator
- Operator decision: Accept or Decline
- If accepted: Arrange pickup
- If declined: Return to company selection
- End: Service arranged or cancelled

State Diagram

Order State Machine:

States:

- Created (initial)
- Processing
- Prepared
- HandedToCourier
- Delivered (final)
- Cancelled (final)

Transitions:

- Created → Processing (seller accepts)
- Processing → Prepared (items ready)
- Prepared → HandedToCourier (given to courier)
- HandedToCourier → Delivered (customer receives)
- Any non-final state → Cancelled (cancellation requested)

Tow Operator State Machine:

States:

- Offline (initial)
- Online
- RequestReceived
- ServiceInProgress

- ServiceCompleted (returns to Online)

Transitions:

- Offline → Online (operator goes online)
- Online → Offline (operator goes offline)
- Online → RequestReceived (new request arrives)
- RequestReceived → Online (request declined)
- RequestReceived → ServiceInProgress (request accepted)
- ServiceInProgress → ServiceCompleted (service finished)
- ServiceCompleted → Online (ready for next request)

Class Diagram

Core Classes and Relationships:

\begin{verbatim}
AppUser

CatalogProduct

OrderDoc

Review

Catalog (Singleton)

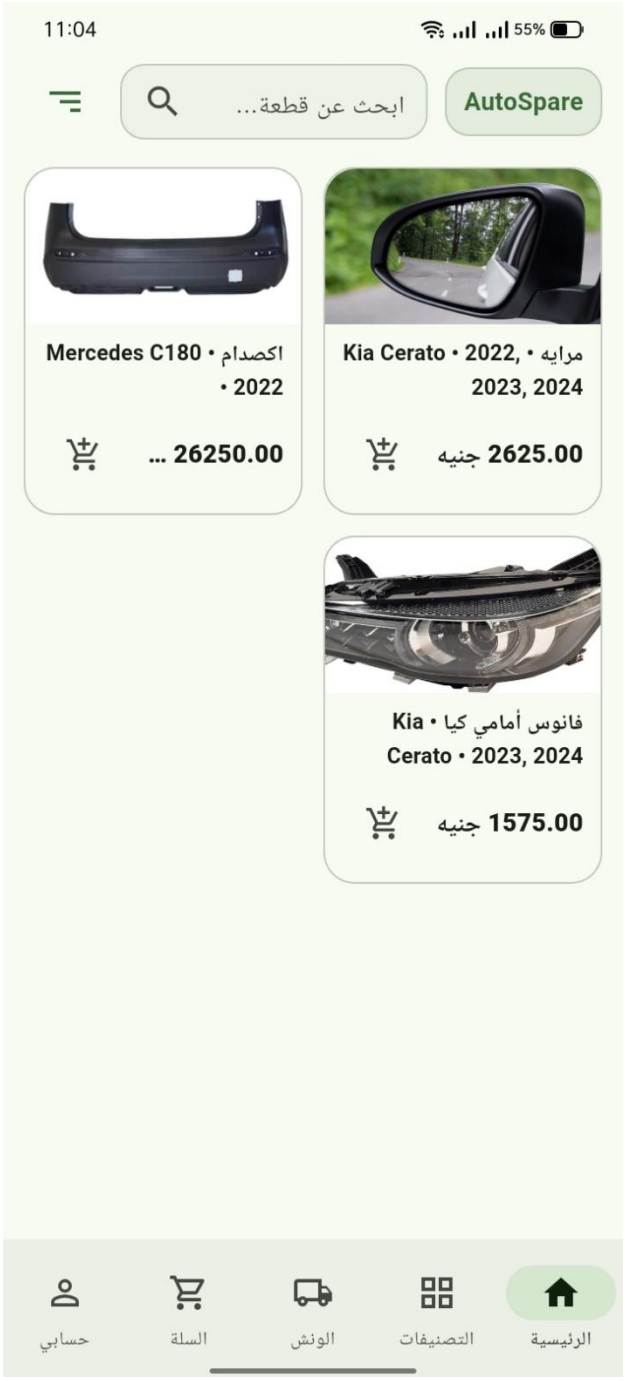
CartService

OrdersRepository (Interface)

ReviewsRepository (Interface)

UI/UX Design & Prototyping

Wireframes & Key Screens:



ابحث عن قطع، علامات تجارية، موديلات...

التصنيفات

تصفح القطع حسب الماركة

Kia
عدد المنتجات: 2

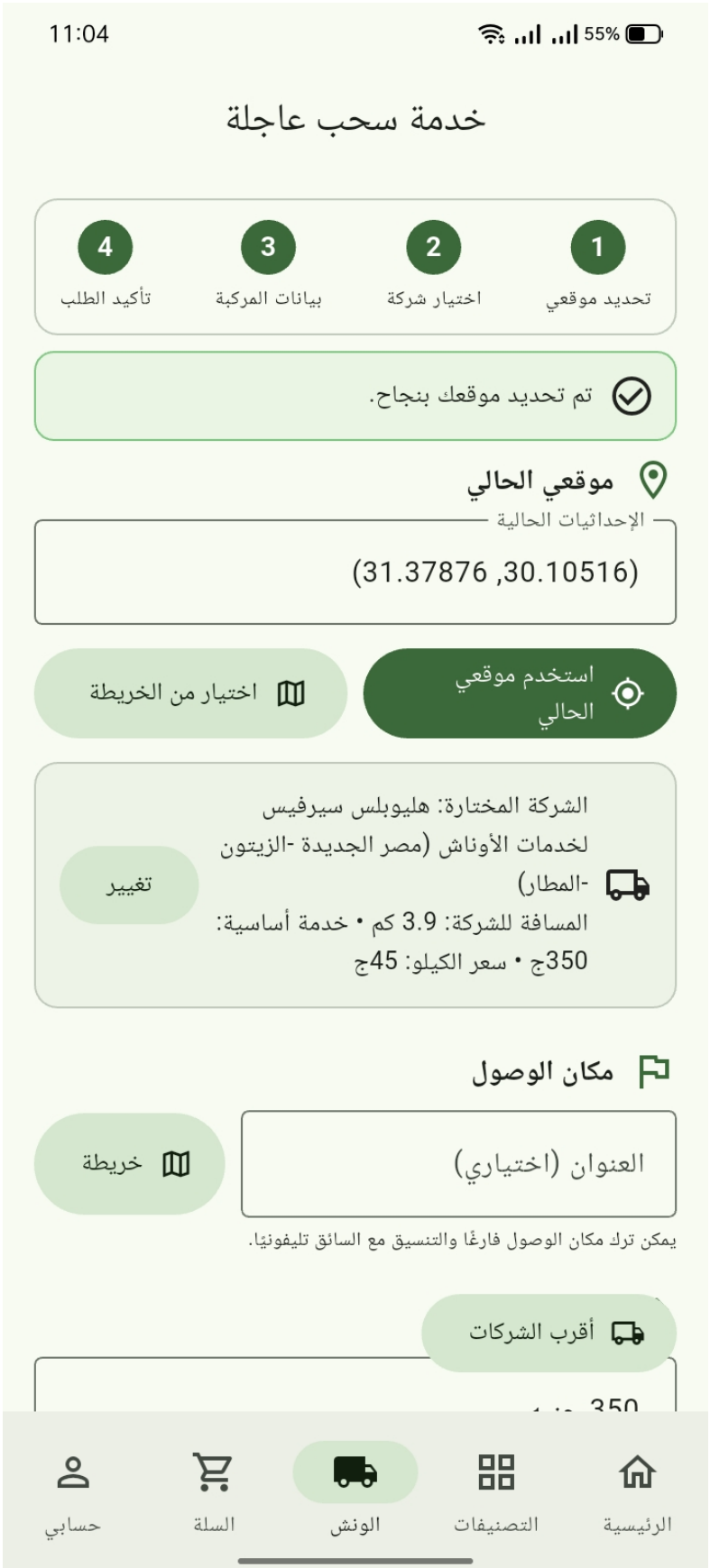
Mercedes
عدد المنتجات: 1

Nissan
عدد المنتجات: 0

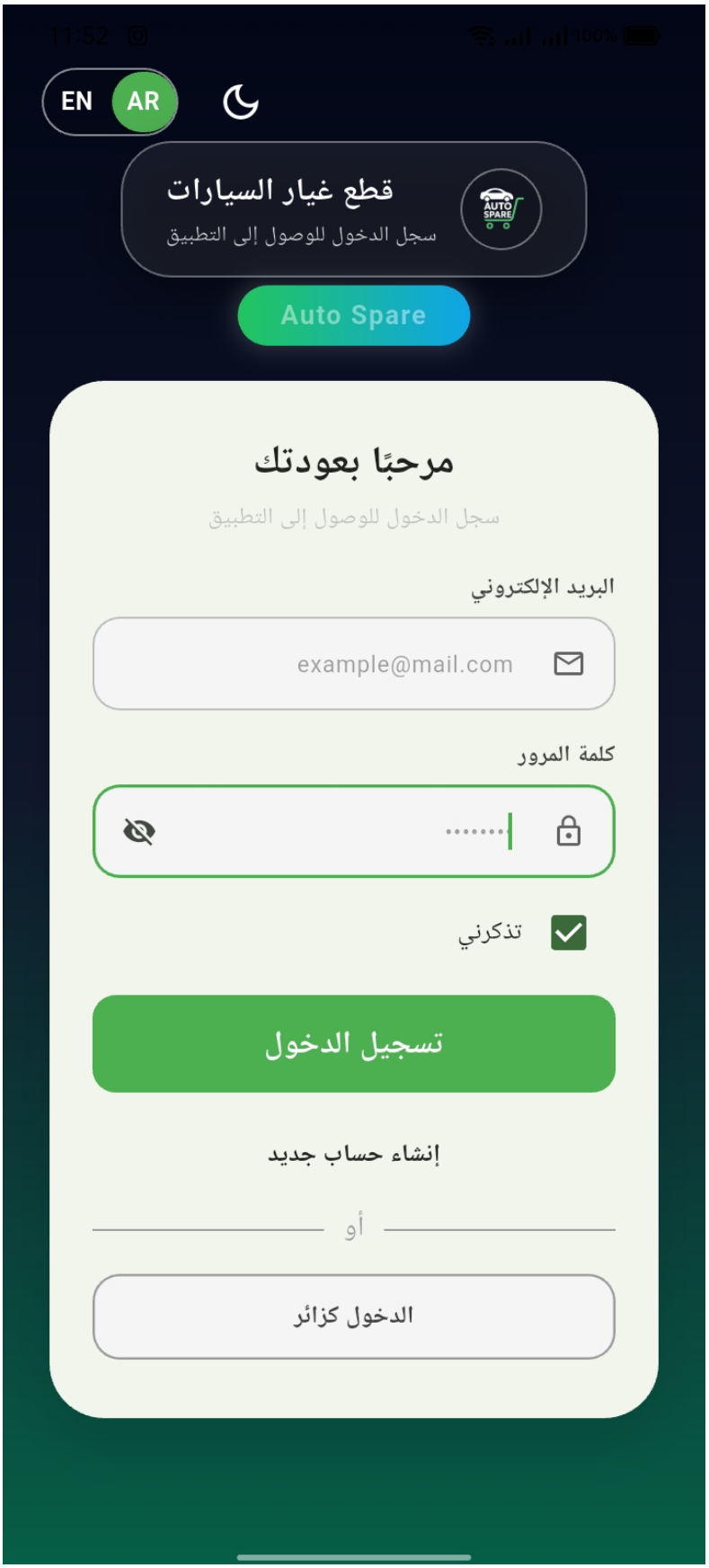
Toyota
عدد المنتجات: 0

Hyundai
عدد المنتجات: 0

BMW
عدد المنتجات: 0


11:04
55%
خدمة سحب عاجلة
1
تحديد موقعي
2
اختيار شركة
3
بيانات المركبة
4
تأكيد الطلب
تم تحديد موقعك بنجاح.
موقعي الحالي
الإحداثيات الحالية
(31.37876 ,30.10516)
استخدم موقعي الحالي
اختيار من الخريطة
الشركة المختارة: هليوبلس سيرفيس لخدمات الأوناش (مصر الجديدة -الزيتون -المطار)
المسافة للشركة: 3.9 كم • خدمة أساسية: 350ج • سعر الكيلو: 45ج
تغيير
مكان الوصول
العنوان (اختياري)
خريطة
يمكن ترك مكان الوصول فارغًا والتنسيق مع السائق تليفونيًا.
أقرب الشركات
350
الرئيسية
التصنيفات
الونش
السلة
حسابي


11:05
55%
خدمة سحب عاجلة
معلومات المركبة
النوع والطراز
نيسان صني
رقم اللوحة
ا ب ت ٣٢١
وصف المشكلة
كتب باختصار المشكلة... (اختياري)
معلومات التواصل
رقم التليفون
01234567890
الوقت المتوقع للوصول
من 15 إلى 25 دقيقة بعد تأكيد الطلب
إرسال الطلب
أقرب الشركات
الرئيسية
التصنيفات
الونش
السلة
حسابي


11:52
100%
EN
AR
قطع غيار السيارات
سجل الدخول للوصول إلى التطبيق
AUTO SPARE
Auto Spare
مرحبًا بعودتك
سجل الدخول للوصول إلى التطبيق
البريد الإلكتروني
example@mail.com
كلمة المرور
تذكرني
تسجيل الدخول
إنشاء حساب جديد
أو
الدخول كزائر


11:06
55%
الملف الشخصي
مرحباً Ahmed Tharwat Ahmed
لوحة إدارة
دور الحساب: أدمن (مراجعة فقط)
لوحة أرباح التطبيق
إدارة الطلبات
إدارة طلبات الونش
حسابات المستخدمين
مراجعة المنتجات
اعتماد البائعين
اعتماد شركات الأو
لوحة مراجعة المنتجات
في الانتظار: 0
لا توجد عناصر قيد المراجعة
الرئيسية
التصنيفات
الونش
السلة
حسابي


11:06
55%
أرباح التطبيق
تغيير الفترة
الفترة الحالية للعرض
2025/12/01 → 2025/11/02
ملخص الأرباح للفترة المختارة
كل الأرقام التالية محسوبة بناءً على الطلبات المكتملة داخل الفترة المحددة فقط.
إجمالي أرباح التطبيق
30,298.81 ج
عدد الطلبات المكتملة
12 طلب
إجمالي المدفوع (Grand Total)
662,940.00 ج
إجمالي قيمة المنتجات
666,225.00 ج
إجمالي الخصومات
3,465.00 ج
إجمالي أرباح التطبيق (تقديري)
30,298.81 ج

11:06 55%

إدارة طلبات الونش

تمت الخدمة 2025/11/30 – 21:22

الشركة: هليوبلس سيرفيس لخدمات الأوناش
إجمالي: 424 جنيه
المركبة: لامبورجيني • اللوحة: ١٢٣ لام
تليفون العميل: 1234567890

ملغي • ألغي بواسطة المشتري 2025/11/30 – 20:47

الشركة: هليوبلس سيرفيس لخدمات الأوناش
إجمالي: 437 جنيه
المركبة: رينو • اللوحة: ن ت ه ١٢٣
تليفون العميل: 01201306493

سبب الإلغاء (المشتري):
تأخر
تاريخ الإلغاء: 2025/11/30 – 20:48

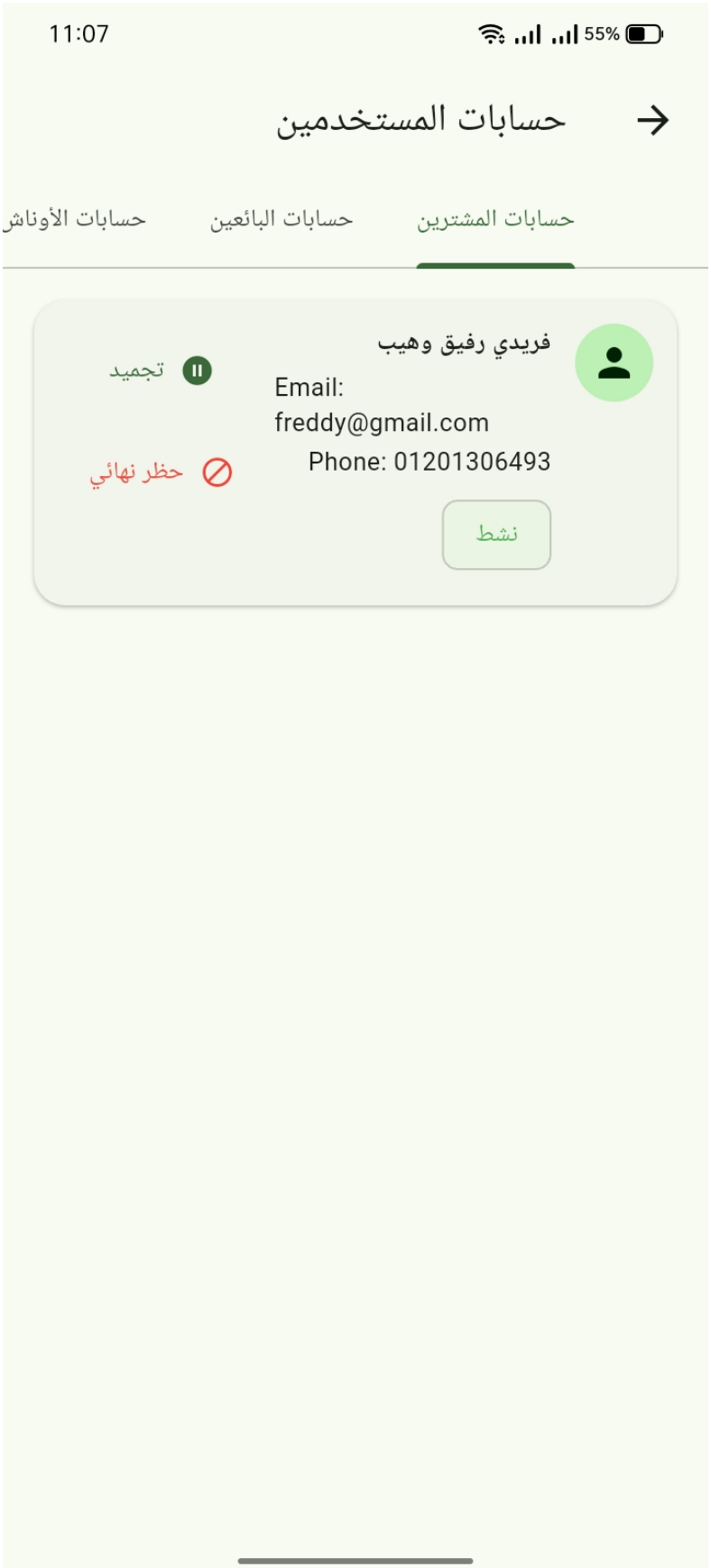
تمت الخدمة 2025/11/29 – 19:14

الشركة: هليوبلس سيرفيس لخدمات الأوناش
إجمالي: 424 جنيه
المركبة: نيسان صاني • اللوحة: ل ل ع ١٢٣
تليفون العميل: 01234567890

ملغي • ألغي بواسطة الشركة 2025/11/29 – 19:06

الشركة: هليوبلس سيرفيس لخدمات الأوناش
إجمالي: 424 جنيه
المركبة: مةلعترل • اللوحة: ١٣٥ ناي
تليفون العميل: 86543804287

2025/11/27 – 23:55 ملغي • ألغي بواسطة المشتري


11:07
55%
حسابات المستخدمين
حسابات الأوناش
حسابات البائعين
حسابات المشترين
فريدي رفيق وهيب
Email:
freddy@gmail.com
Phone: 01201306493
نشط
تجميد
حظر نهائي


11:07
55%
حسابات المستخدمين
حسابات الأوناش
حسابات البائعين
حسابات المشترين
اريزونا لأعطال السيارات وسحب السيارات
Email: arezona@gmail.com
Phone: 01002223334
نشط
تجميد
حظر نهائي
شركة الفضل للأوناش
Email: fadl@gmail.com
Phone: 01002223331
نشط
تجميد
حظر نهائي
السفير لأعطال السيارات
Email: safeer@gmail.com
Phone: 01002223332
نشط
تجميد
حظر نهائي
الحرية لخدمات الاوناش
Email: horrya@gmail.com
Phone: 01002223337
نشط
تجميد
حظر نهائي

System Deployment & Integration

Technology Stack:

Key Dependencies (from pubspec.yaml)

Additional Deliverables

Testing & Validation:

Manual testing performed covering:

Deployment Strategy:

Development Environment:

Production Environment:

5. Implementation (Source Code & Execution)

Source Code Structure

MVC: Model View Controller

Coding Standards & Best Practices

Code Quality Measures:

Modular Code Organization:

The project follows clear separation of concerns:

Security & Error Handling:

Version Control & Collaboration

Version Control Repository: Hosted on GitHub with a public repo link

Branching strategy: Branches for every feature

Commit History:

The repository contains structured commits covering feature development, bug fixes, and documentation updates. Meaningful commit messages describe changes made.

Collaboration Workflow:

Team members collaborated through shared GitHub repository with regular code pushes.

6. Testing & Quality Assurance

Test Plan

Testing Approach:

Manual testing conducted throughout development to verify functionality across all user roles. Testing focused on user workflows, data validation, and integration with Firebase services.

Test Environment:

Test Cases

Authentication Tests:

Authentication Tests:

Product Management Tests:

Order Processing Tests:

Review System Tests:

Tow Service Tests:

Admin Moderation Tests:

7. Final Presentation & Reports

User Manual

Getting Started:

For Buyers:

Browsing and Purchasing, **Requesting Tow Service**

For Sellers: Create accounts, upload products and manage them

For Tow Operators: Create account, manage user requests

For Administrator: Manage account flow of each profile

Technical Documentation

System Architecture Summary:

AutoSpare implements MVC architecture with repository pattern for data access. The application uses Firebase for authentication and data storage, providing scalable backend infrastructure. Flutter framework enables cross-platform deployment on Android and iOS from a single codebase.

Key Technical Features:

Data Models:

The application uses strongly-typed immutable data models with null safety. Key models include AppUser, CatalogProduct, OrderDoc, OrderItem, and Review. Enums define role types, car brands, and order statuses.

Localization (l10n):

Implemented for multilingual support, specifically Arabic and English. This feature ensures all user-facing text, labels, and messages can be dynamically switched based on the user's selected language, enhancing NFR3 Usability and providing an easy experience for the target audience.

Appendix

Team Information

Team Name: Team3_CAI3_SWD8_G1

Team Members:

Ahmed Tharwat Ahmed

Ahmed Mostafa Almazon

Ziad Mohammed Rafat

Salma Yousry Abdel Razek

Omar Abdelalim Abdelfattah

Freddy Rafik Wahib

Instructor: Eng. Micheal Hany

Repository Information

GitHub Repository: https://github.com/AhmedAtaar/Team3_CAI3_SWD8_G1_AutoSpare