

# **Where 2 Go**

Local Business & Entertainment Directory Platform

December 1, 2025

# Contents

<b>1 Project Planning &amp; Management</b>	<b>3</b>
1.1 Project Plan (Timeline) . . . . .	3
1.2 Updated Task Assignment & Roles . . . . .	5
<b>2 Requirements &amp; System Design</b>	<b>6</b>
2.1 Stakeholder Analysis . . . . .	6
2.2 Functional Requirements . . . . .	6
2.3 Non-Functional Requirements . . . . .	7
<b>3 Implementation &amp; Technical Documentation</b>	<b>8</b>
3.1 Technology Stack . . . . .	8
3.2 Visual Diagrams . . . . .	9
3.3 Key API Endpoints (RBAC Focus) . . . . .	10
3.4 Listing Lifecycle Logic . . . . .	12
<b>4 Testing and Quality Assurance</b>	<b>12</b>
4.1 Test Plan . . . . .	12
4.2 Code Standards . . . . .	13

**Conclusion** **14**

**References** **14**

# 1 Project Planning & Management

## 1.1 Project Plan (Timeline)

Phase	Description
Phase 1: Requirements	Defining all functional, non-functional requirements, and stakeholder analysis.
Phase 2: Design	UI/UX prototyping, database schema design (ERD), and core API definition.
Phase 3: Development	Implementation of Advanced Search, Authentication, Listing Lifecycle, and multilingual features.
Phase 4: Finalization	Comprehensive testing, documentation, and deployment.

Where 2 Go: Local Business & Entertainment Directory Platform

---

## 1.2 Updated Task Assignment & Roles

Member	Role	Key Responsibilities
Saif Harraz	Frontend Developer / Listing Management	Building the Add Listing interface and the Listing Details pages.
Amr Ahmed	Frontend Developer / Core Pages	Developing the Home page and the main Listing (general catalog) page.
Amira Hesham	Frontend Developer / Globalization & Documentation	Implementing Arabic/English support, overseeing the Dashboard interface, and preparing Documentation.
Mohamed Hos-sam	Backend Developer / System Logic & Contact	Developing the core Backend system logic and building the Contact interface.
Awad Fahim	Backend Developer / Authentication System <sup>6</sup>	Developing the Backend architecture focused on the Sign

## 2 Requirements & System Design

### 2.1 Stakeholder Analysis

The primary stakeholders for the *Where 2 Go* platform encompass distinct user groups, each with unique needs and expectations:

- **Customer:** Requires intuitive filtering capabilities by price level and convenient features to track favorites and history for a personalized experience.
- **Owner:** Needs robust CRUD (Create, Read, Update, Delete) access to manage their own listings efficiently.
- **Admin:** Holds comprehensive control over user and listing management, including the critical approval workflow to maintain platform quality.

### 2.2 Functional Requirements

The platform's core functionalities are designed to meet stakeholder needs through the following capabilities:

- **Advanced Search:** Users can filter listings by city, category, and price level to quickly locate desired venues.

- **Geographical Sort:** Integration of radius-based sorting utilizing latitude and longitude parameters enhances search precision.
- **Authentication:** Secure user access employing JWT tokens combined with Role-Based Access Control (RBAC).
- **Listing Lifecycle:** Facilitates owner submissions with an admin approval process, including re-approval for any updates to maintain listing integrity.

## 2.3 Non-Functional Requirements

To ensure a seamless and secure user experience, the system must adhere to the following quality attributes:

- **Security:** Implementation of JWT and rigorous ownership verification mechanisms on all sensitive endpoints to safeguard data.
- **Usability:** Full bilingual support in Arabic and English, including right-to-left (RTL) layout compatibility.
- **Performance:** API response times must consistently remain under 500 milliseconds to guarantee responsiveness.

## 3 Implementation & Technical Documentation

### 3.1 Technology Stack

The system architecture leverages modern, scalable technologies to ensure robustness and maintainability:

Layer	Technologies
Frontend	React, TypeScript, ShadCN/UI, Tailwind CSS, react-i18next (for multilingual support)
Backend	Node.js, Express.js, MongoDB, JWT Authentication

## 3.2 Visual Diagrams

### A. Architecture Diagram

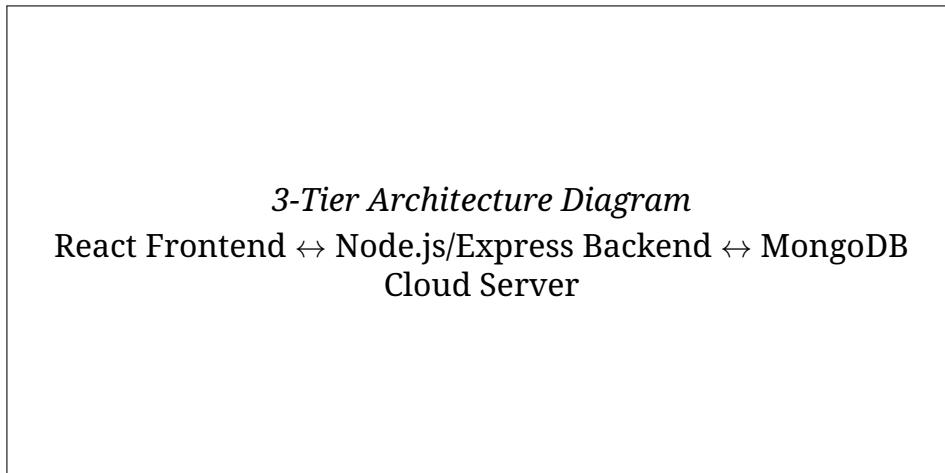


Figure 3.1: System Architecture Overview

### B. Entity-Relationship Diagram (ERD)

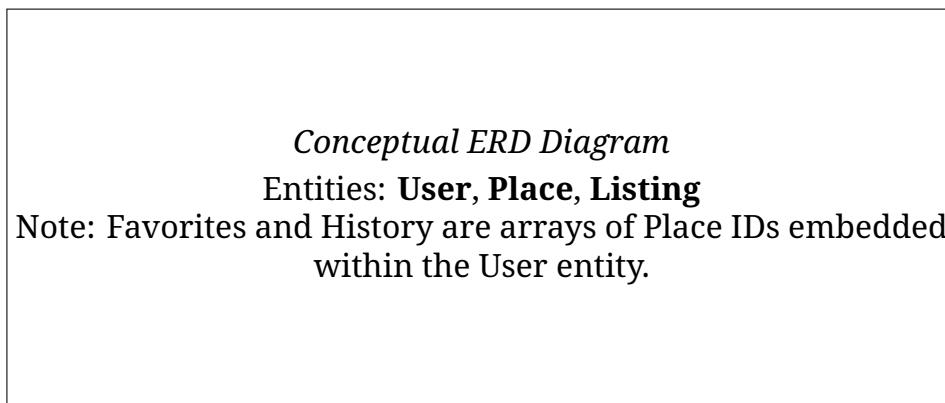


Figure 3.2: Conceptual ERD

### 3.3 Key API Endpoints (RBAC Focus)

The following table summarizes critical API endpoints with role-based access control:

## Where 2 Go: Local Business & Entertainment Directory Platform

---

<b>API</b>	<b>Method &amp; Endpoint</b>	<b>Description / Access</b>
Places API	GET /api/v1/places/search	Supports filtering by priceLevel, sortBy, lat/lng for geographical sorting. Public access.
User API	POST /api/users/favorites	Add a place to user's favorites list. User authenticated.
	POST /api/users/history	Add a place to user's history list. User authenticated.
Admin/Owner Listing API	POST /api/listings/submit	Owner only. Submits new listing; status set to pending for admin review.
	PUT /api/listings/:id/own	Owner only. Update own listing information.

### 3.4 Listing Lifecycle Logic

The listing workflow is designed to maintain quality and accountability:

1. **Submission:** An owner submits a new listing, which is marked as pending.
2. **Admin Review:** Admin evaluates the submission for compliance and quality.
3. **Decision:** Listing is either accepted (becomes visible on the platform) or rejected (deleted).
4. **Updates:** Any owner-initiated update resets the listing status to pending, triggering a new admin review cycle.

## 4 Testing and Quality Assurance

### 4.1 Test Plan

To ensure robust functionality and reliability, the test plan emphasizes:

- **Search Accuracy:** Validate the Advanced Search and Geographical Sort features return correct and relevant results under various conditions.
- **Role-Based Access Control (RBAC):** Confirm that all endpoints enforce proper access restrictions for Customers, Owners, and Admins.
- **Listing Lifecycle Workflow:** Verify that the submission, review, approval, rejection, and update cycles operate correctly and consistently.

## 4.2 Code Standards

The development process adheres to high-quality coding standards to maintain clarity and scalability:

- **TypeScript Usage:** Ensures type safety and early error detection across frontend and backend codebases.
- **Modular Architecture:** Segregation of components and services to facilitate maintainability and code reuse.
- **Consistent Naming Conventions:** Uniform and descriptive identifiers promote readability and ease collaboration.

## Conclusion

The *Where 2 Go* application presents a thoughtfully designed and scalable platform tailored to Egyptian users seeking optimal local businesses and entertainment venues. Through meticulous planning, a clear division of team roles, and adherence to modern development and security standards, the project establishes a solid foundation for functionality and user satisfaction. Comprehensive testing strategies and quality assurance measures ensure the platform not only meets but exceeds stakeholder expectations, heralding a promising solution within the competitive digital landscape.

## References

**Digital Egypt Pioneers Initiative.** Project supervision documents and technical guidelines.

**React Documentation.** <https://reactjs.org/docs/getting-started.html>

**Node.js Documentation.** <https://nodejs.org/en/docs/>

**MongoDB Manual.** <https://docs.mongodb.com/manual/>

**JWT.IO.** <https://jwt.io/introduction/>

**Tailwind CSS.** <https://tailwindcss.com/docs>

**ShadCN/UI.** <https://ui.shadcn.com/>

**react-i18next.** <https://react.i18next.com/>