Functions and Features - Overall summary

App & System Development Overview for Outsourcing Evaluation

1. Core Technologies & Architecture

1.1 Language

- **Server: Go (Golang)**
- **App & web: Flutter (dart)**

1.2 API Layer

- **GraphQL API:** `gqlgen`.
- **HTTP Framework:** `gin-gonic/gin`
- **GraphQL Directives:** Authorization logic is implemented cleanly using GraphQL directives like `@hasRole`, which keeps resolver logic focused on business tasks.

1.3 Database

- **Open for suggestions**

1.4 Cloud Storage

- **Abstraction Layer:** A interface define clean abstraction for cloud storage operations.
- **Storage Providers:**
  - Depend on hosting platform TBA
  - potential candidate
    - AWS S3 (Simple Storage Service)

- Google Cloud Storage
- Azure Blob

1.5 Authentication & Authorization

- **JWT (JSON Web Tokens):** User authentication is managed via JWTs. Upon successful login, a session is created in the database and a JWT is issued to the client for authenticating subsequent requests.

- **Access Control (AC):** The system features a comprehensive AC model. Users are assigned roles (e.g., `Member`, `Operator`, `SysOp`) and permissions, and access to GraphQL operations and data is restricted based on these restrictions. This is enforced through middleware and GraphQL directives.

1.6 Other Packages & Services

- **Email Service:** Transactional emails (e.g., registration confirmation, login credentials)

- **Configuration:** Application configuration is managed through **environment variables and config files.**

1.7 Supported platform

- **iOS:** iOS 16+

- **Android:** Android 13+

- **Web Browse:**
  - **iOS:** Safari, Chrome
  - **Android:** Chrome
  - **Desktop:** Microsoft Edge, Chrome, Safari, Firefox

1.8 Backend execute environment

- The backend artifact must be executable in a Docker runtime and brought up by docker-compose per the defined compose configuration.

2. Core Modules

Priority	Module	Description	Key Requirements / Features
1	**Cross-System Integration**	Core infrastructure that connects all sub-systems through unified APIs. Handles authentication, payment, notifications, and analytics.	• API Gateway setup and endpoint structure • Centralized authentication (JWT / SSO) • Payment service (charge, refund, transaction log) • Notification service (email, push) • Feedback & error handling APIs
2	**User System**	Foundation for all user management and access control. Provides unified identity and role-based permissions.	• User registration & login (email / mobile) • Forgot password & verification flow • Role-based access control (Admin / Operator / Member) • Profile management & session tracking • Membership management (Subscriptions, add, renew, expire) • Three UI layout for different roles (Admin / Operator / Member) • Security logging & compliance (PIPEDA) • Payment integration (credit card / refund) • Membership tiers and loyalty point system (Phase 2)
3	**Facility Booking System (with Payment)**	Allows members to reserve facilities and handle payments. Integrates with Membership and Payment APIs.	• Facility list and time-slot selection (interactive map) • Booking creation, modification & cancellation • Payment integration (credit card / refund) • Admin override & manual

Priority	Module	Description	Key Requirements / Features
			booking control • Booking reports & usage analytics
4	**Food Ordering System**	In-app food menu and ordering system with operational dashboard for staff.	• Digital menu management (categories, prices) • Payment integration (credit card / refund) • Order creation, update & status tracking • Real-time kitchen dashboard (order queue) • Promotional pricing & discounts • Order reports & sales summary • POS / printer integration (Phase 2)
5	**Pickleball Analytics (frontend)**	Match Video Analysis and Performance Insights System	• List on-site captured videos (partial) • Video playback & download • Tagging / categorizing match clips • Visualize player performance analytics (Interactive map, chart, heat map, diagram, etc) • Compare performance data across two matches
6	**ATTA App Integration**	Embeds Progress Tracker & T-Shots video playback into the main app.	• Show member QR Code (serve as the launch key for ATTA's Progress Tracker and T-Shots) • API for retrieving user identity from member QR code • List, playback, download assessment records, video • Secure token-based data

Priority	Module	Description	Key Requirements / Features
			exchange via API • Token system for T-Shots (phase 2)
7	**Premium Car Parking**	Parking management module sharing similar logic to facility booking.	• Parking slot reservation & time-based pricing • Real-time slot availability display • Booking creation / cancellation / refund • Payment API integration • Usage analytics
8	**Pro Shop Ordering System**	Online shop for merchandise and equipment sales.	• Product catalog (images, stock) • Cart & checkout process • Order management (status, refund) • Discount code & promotions (Phase 2) • Integration with payment

3. Technical Overview

- Core module backend should be separated and communicate though API

- Module authenticate user with JWT issued by Membership System

- Centralized **membership/authentication system** that connects all sub-systems via RESTful APIs.

- **Permission control** and **operation logging** (PIPEDA compliance).

- Unified **payment/refund API** across all modules.

- **Reporting and analytics** for sales, facility usage, and user activity.

- **Phase 2 enhancements:** loyalty program, advanced discount engine, and AI/video analytics.

4. Development Scope Summary

Area	Details
Frontend	Web and Mobile App (iOS + Android) desktop + mobile layout Including admin, operator functions
Backend	API Gateway, microservices for each module, and per-module databases
Integration	Authentication, Payment, Notification, and Analytics modules.
Deliverables	**Phase 1:** Membership, Facility booking, Food, Pickle ball analytics, Pro Shop, Parking. **Phase 2**: **Membership System**: Membership tiers and loyalty point system **Food Ordering System**: POS / printer integration **Pro Shop Ordering System**: Discount code & promotions

5. Supported platform

supported Platform	Requirements	Deliverables
iOS	iOS 16+	
Android	Android 13+	
Web Browser	Desktop: Microsoft Edge, Chrome, Safari, Firefox iOS: Safari, Chrome Android: Chrome	

6. Development environment

Core Technologies & Architecture

6.1 Language

- **Server: Go (Golang)**

- **App & web: Flutter (dart)**

6.2 API Layer

- **GraphQL API:** `gqlgen` .

- **HTTP Framework:** `gin-gonic/gin`

- **GraphQL Directives:** Authorization logic is implemented cleanly using GraphQL directives like `@hasRole` , which keeps resolver logic focused on business tasks.

6.3 Database

- **Open for suggestions**

6.4 Cloud Storage

- **Abstraction Layer:** A interface define clean abstraction for cloud storage operations.

- **Storage Providers:**

  - **AWS S3 (Simple Storage Service):** .

6.5 Authentication & Authorization

- **JWT (JSON Web Tokens):** User authentication is managed via JWTs. Upon successful login, a session is created in the database and a JWT is issued to the client for authenticating subsequent requests.

- **Access Control (AC):** The system features a comprehensive AC model. Users are assigned roles (e.g., `Member` , `Operator` , `SysOp`), and access to GraphQL operations and data is restricted based on these restrictions. This is enforced through middleware and GraphQL directives.

6.6 Other Packages & Services

- **Email Service:** Transactional emails (e.g., registration confirmation, login credentials)

- **Configuration:** Application configuration is managed through **environment variables**, which are loaded at runtime using a custom utility in osutil.

7. Milestone Plan

Milestone	Target Date	Deliverables / Goals
Milestone 0 – Project Kickoff	**Dec -Jan 2025**	• Confirm technical stack and architecture plan • Set up repositories and communication channels

Milestone	Target Date	Deliverables / Goals
& Setup		• Define API Gateway framework for Cross-System Integration
Milestone 1A – Cross-System Integration (Backend Foundation)	**Jan - Feb 2026**	• API Gateway core implementation • Authentication (JWT / SSO) ready • Payment API base structure • Logging and notification framework
Milestone 1B – User System	**Feb 2026**	• Registration / login / password reset • Role-based access control • Profile management and data logging• Integration with API Gateway
Milestone 1C – Facility Booking System (with Payment)	**Mar–Apr 2026**	• Facility booking and management • Payment / refund integration • Admin override and usage reporting
Milestone 1 – MVP Completion	**May 2026**	✅ Delivery of MVP (3 core modules): • Cross-System Integration • **User** System • Facility Booking System (with Payment)Includes backend integration, admin panel, and testing environment. ⚠️ All other modules start *after this date*.
Milestone 2A – Food Ordering System	**May–Jun 2026**	• Menu management and order workflow • Order dashboard for kitchen / staff• Promotion / discount features • Integration with Membership and Payment APIs
Milestone 2B – Pickleball Analytics & ATTA Integration	**Jun–Jul 2026**	• Video upload / playback / tagging functions • Embedded ATTA modules (video / data) • Unified login and data access • Internal UAT across all modules
Milestone 2C – Premium Card Parking & Pro Shop System	**Jul 2026**	• Parking reservation module with payment • Pro Shop ordering and checkout system• Final integration tests
Milestone 2 – Final Delivery (Completion)	**Aug 1, 2026**	✅ Final handover including:• Food Ordering System • Pickleball Analytics• ATTA App Integration • Premium Card Parking

Milestone	Target Date	Deliverables / Goals
		• Pro Shop Ordering System+ Performance optimization, full QA, documentation, and complete source code (front-end + back-end).

8. Other Terms

- During the development phase, ATTA has the right to review the source code and make reasonable modification requests.

- Upon completion of development, Contractor shall provide ATTA with all deliverables, including the application's front-end, back-end, and complete source code.

- **During the development process, ATTA shall have access to the project's Git repository and/or sandbox environment for the purposes of monitoring progress, reviewing code quality, and conducting preliminary testing.**