TQS: Product specification report

Miguel Alexandre Moura Neto [119302]
Alexandre Filipe da Paula Andrade [119279]
Thiago Aurélio Pires Barbosa Vicente [121497]
João Pedro Silva Pereira [120010]

v2025-11-25

1 Introduction

1.1 Overview of the project

The project assignment given in the TQS course is devised to develop a peer-to-peer equipment and gear rentals marketplace platform. Item owners will register their products for renters to browse and book, in order to give usage, thus providing passive income from otherwise idle assets.

In terms of the project's scope of equipment and gear is comprised of **musical instruments** available for users to rent and use for whichever context they are required, ranging from string instruments to wind instruments and percussion, as well as **physical, registered sheet music samples** which are used for variably-sized groups of musicians (that is, *ensembles*) and different music styles.

Such a solution allows peers to no longer resort to several social media and similar platforms for renting and sharing idle assets, which became a nuisance due to having to stay up to date with multiple sources to ensure income or find the desired asset. Other advantages inherently include feedback among peers, using reputation scores and asset quality assurance, as well as controlling *renting* (booking will be considered more appropriate for the remaining report, due to the nature of the project's guidelines and concept overall). Differentiating features to ease discovery, booking, payments and so on are key factors for the platform.

Those are the core concepts for the product, which was called OLS (OLSheets), paraphrasing the known platform OLX, known for the online exchange of various products, focusing on instruments and sheet music in this case.

1.2 Project limitations and known issues

Despite great effort to implement as much as possible for the sake of a seamless experience for the end users, there had to be scrapped off features like favorites/wishlists, for the sake of simplicity and to stay on the scope of the core evaluation parameters taken in the TQS class, which revolve especially around testing, Quality Assurance, and overall the DevOps lifecycle.

Other known limitations include:

- No real-time notifications for booking status changes
- Limited payment integration (currently only supporting simulated payment processing)
- No calendar integration for availability management

1.3 References and resources

- https://prometheus.io/docs/prometheus/
- https://confluence.atlassian.com/agile064
- https://www.postgresql.org/docs/
- https://docs.spring.io/spring-framework/reference/index.html
- https://docs.kubermatic.com/kubermatic/v2.29/tutorials-howtos/monitoring-logging-alerting/user-cluster/setting-up-alertmanager-with-slack-notifications/
- https://grafana.com/docs/

2 Product concept and requirements

2.1 Vision statement

OLSheets is a peer-to-peer marketplace platform that enables musicians and music enthusiasts to rent and lend musical instruments for playing, and sheet music for interpretation. The platform aims to solve the problem of idle musical assets by connecting owners who want to monetize their unused instruments and sheet music with renters, who need temporary access to these resources.

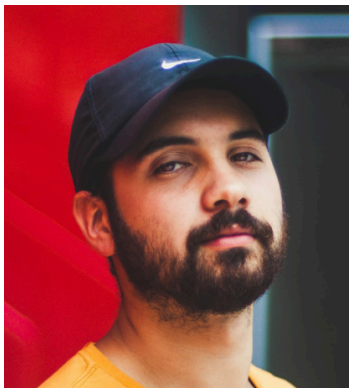
Key features of the system include:

- **Item discovery** with search and filtering capabilities
- **Registration and management** of instruments and music sheets
- **Booking management** with availability tracking
- **Secure payment processing**
- **Review and rating** system for building trust

The system differentiates itself from general marketplaces by focusing specifically on musical instruments and sheet music, providing domain-specific features like instrument family/type categorization and music categorization by genre or ensemble type.

2.2 Personas and scenarios

Carlos Correia (**Instrument Renter**): A 22-year-old university student enrolled in a music performance degree. He regularly participates in rehearsals, academic evaluations, and small public performances, which often require access to instruments that he does not personally own. Due to financial constraints typical of a student lifestyle, purchasing high-cost instruments is not a viable option. Carlos is accustomed to using digital platforms for everyday tasks, including academic tools, online payments, and service marketplaces.
Before using OLSheets, Carlos relied on informal social media groups and personal contacts to borrow or rent instruments, a process that was unreliable and time-consuming, with little guarantee of quality or availability. His main goals are to quickly find suitable instruments, ensure they are in good condition, and reserve them for specific dates without uncertainty.
Scenario: Carlos is informed that he will need a bassoon for an orchestra performance scheduled two weeks ahead. He accesses the OLSheets platform, searches for bassoons, and applies filters such as price range and availability. After reviewing detailed descriptions and photos, he selects an instrument that meets his needs, checks its availability for the required period, and completes a booking using the platform's payment system. The process allows Carlos to secure the instrument with confidence and focus on his musical preparation rather than logistical concerns.



Ivo Costa (**Instruments Owner**): A 35-year-old professional musician who owns several brass instruments, including tubas that are not in regular use. While he does not wish to sell these instruments, he is interested in generating additional income from them when they are idle. Ivo values control and transparency, particularly regarding who rents his instruments and under what conditions. He is an experienced user of digital tools in his professional life and expects clear interfaces and reliable transaction tracking.
Previously, Ivo considered informal rentals but was discouraged by the lack of accountability and difficulty managing schedules and payments. His primary objectives are to manage availability, define rental conditions, and ensure that both the instruments and transactions are handled responsibly.

Scenario: Ivo registers one of his tubas on OLSheets by providing photos, a detailed description, rental pricing, and availability periods. He later receives a booking request from a renter, reviews the request, and approves it through the platform. After coordinating the handover, Ivo receives payment via the platform and, once the rental period ends, leaves a review of the renter. This structured process allows Ivo to monetize his assets with reduced risk and administrative effort.

Jorge Lopes (**Music Sheets Owner**): a 60-year-old retired music teacher and former orchestra conductor. Over decades of teaching and performing, he has accumulated a valuable collection of rare and specialized sheet music, much of which is no longer actively used. While Jorge wishes to preserve his collection, he also sees an opportunity to generate supplementary income by making it accessible to other musicians. He has basic digital skills and prefers simple, guided workflows, while still having main concerns that include maintaining the integrity of the sheet music and ensuring it is used responsibly. Clarity over advanced functionality is valued.

Scenario: Jorge registers several sheet music items on OLSheets, providing information such as composer, and difficulty level. He defines rental prices and availability periods and monitors ongoing rentals through the platform. OLSheets allows Jorge to extend the lifespan and relevance of his collection while maintaining oversight and protection.

Diana Gomes (**Sheets Renter**): A 28-year-old amateur composer and music enthusiast who frequently studies different musical styles and compositions for inspiration and analysis. She often needs temporary access to a wide range of sheet music but does not wish to purchase every piece she studies. Diana is highly comfortable with digital platforms and regularly uses online tools for research and creative work.
Before using OLSheets, Diana faced high costs and limited access when searching for diverse sheet music collections. Her goal is to efficiently explore and study multiple compositions with minimal financial commitment.
Scenario: Diana searches the platform for sheet music by composer and genre, selects several pieces of interest, and rents them for a defined period. She accesses the materials, studies them during the rental window, and returns them upon completion. The platform enables Diana to expand her creative resources without long-term ownership or unnecessary expense.

Tiago Óscar (**System Administrator**): a 40-year-old IT professional responsible for the technical and operational maintenance of the OLSheets platform. His role includes monitoring system performance, managing user accounts, overseeing bookings and payments, and addressing reported issues. Tiago has extensive technical expertise and relies on dashboards, logs, and monitoring tools to ensure system reliability and data integrity.

Scenario: Tiago accesses the internal maintenance dashboard to review key performance indicators, investigate reported user issues, and ensure that platform operations comply with defined policies. He performs maintenance tasks during off-peak hours to minimize service disruption, ensuring a stable and trustworthy environment for all users.

2.3 Project epics and priorities

Over the iterations that involved development of the solution, the epics described below were taken into account in the following order:

1. Item Discovery & Catalog - Developed second to create the core marketplace functionality with instrument and sheet music registration, searching, and filtering
2. Booking & Scheduling - Implemented third to enable the rental transaction process with booking creation and management
3. Payment & Transactions - Added fourth to handle the financial aspects of rentals with secure payment processing
4. User Management & Authentication - Implemented first to establish the foundation of user accounts and secure access to the platform
5. Reviews & Trust System - Developed fifth to build community trust through rating systems for both items and renters
6. Platform Management & Administration - Implemented last to provide oversight capabilities for system administrators

List of epics:

- **Item Discovery & Catalog**

User Stories

As an instrument renter, I want to search for instruments by name so that I can quickly find what I want to rent.	As an instrument renter, I want to filter instruments by instrument family (wood, metal, strings, percussion) so that I can browse related items.
As a sheet renter, I want to search for music sheets by music title so that I can find the specific sheet I want to learn	As a sheet renter, I want to filter music sheets by category (e.g., classical, rock, jazz) so that I can find styles I like.
As an instrument renter, I want to filter instruments by type (e.g., guitar, violin) so that I can narrow down my options.	As a sheet renter, I want to view details(and photos maybe) for music sheets so that I know exactly what I'm renting
As an instrument renter, I want to view detailed descriptions and photos of instruments so that I can evaluate their condition and suitability	

- **Item Management**

As an instrument owner, I want to register my instruments with photos and descriptions so that renters can find them	As a sheet owner, I want to upload my music sheets with details so I can rent them out.
As any owner, I want to update availability so that renters know when items can be booked.	As any owner, I want to update pricing so that I can adjust rental rates.

- **Booking & Scheduling**

As an instrument renter, I want to book an instrument for specific dates so that I can reserve it.	As a sheet renter, I want to book music sheets for specific dates so that I can plan my learning schedule.
As any owner,	As any owner,

I want to approve or reject booking requests so that I stay in control of instruments/sheet rentals.	I want calendar integration so that I can see upcoming rentals easily.

- **Payment & Transactions**

As any renter, I want to pay securely for an instrument rental so that I can complete the transaction safely.

- **Platform Management & Administration**

As an admin, I want to monitor platform KPI's so I can understand performance and usage.	As an admin, I want to oversee bookings, payments, and user behavior so I can ensure everything runs smoothly.
As an admin, I want to monitor reported issues, So I can ensure that the users' problems are being taken care of.	

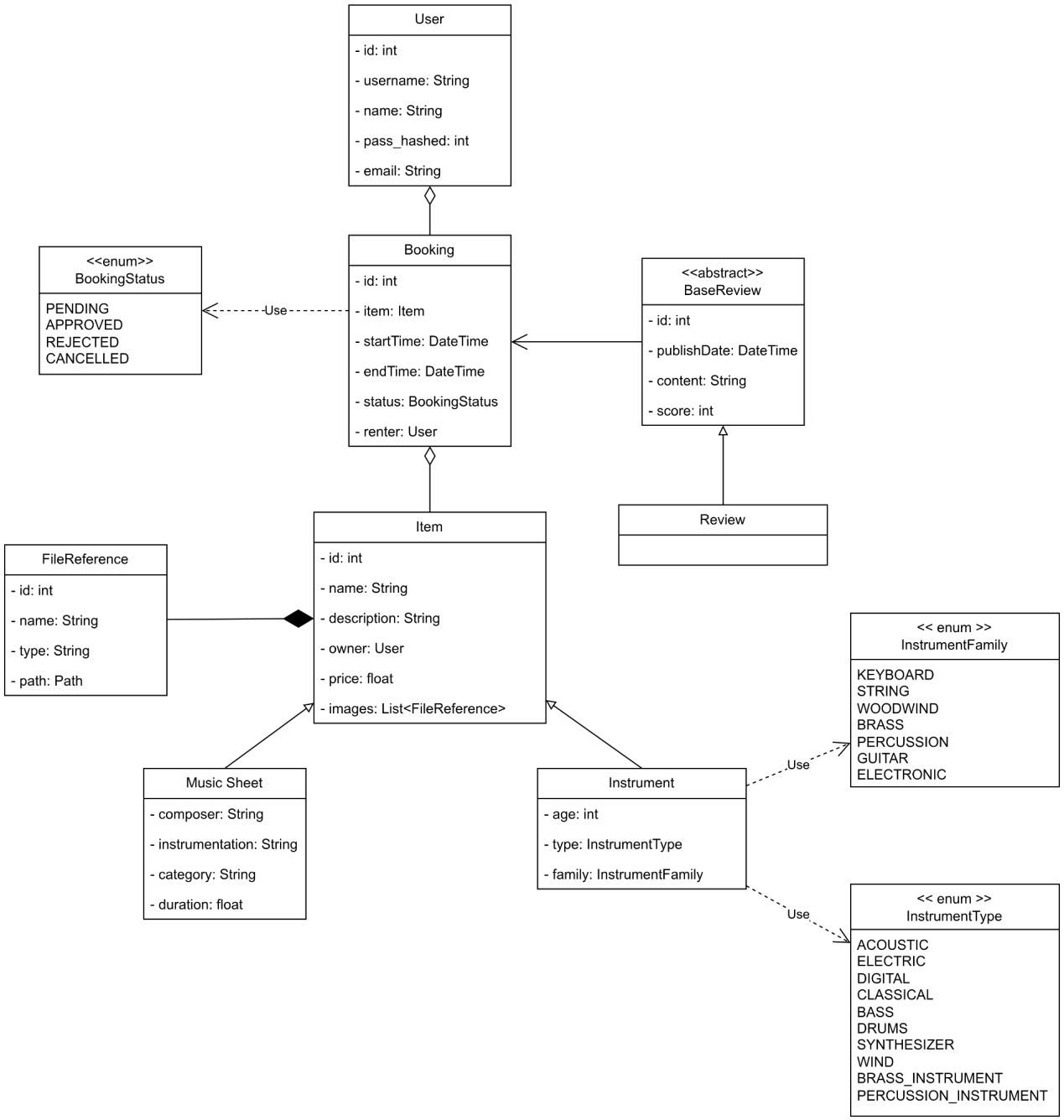
- **User Profiles & History**

As any renter, I want to view my past rentals so that I can track what I've used.	As any renter, I want to see my active bookings so I know what I currently have rented.
As a sheet renter, I want to save my favorite sheets so I can rent them later.	As any owner, I want to monitor my earnings so that I can track my income.

- **Reviews & Trust System**

As any renter, I want to rate an instrument/sheet after renting it so that I help others make decisions.	As any owner, I want to rate renters so that I build trust with future customers.
As any user, I want to report issues so that the platform can resolve problems.	As any renter, I want the reputation scores of owners to help me decide whether to rent

3 Domain model



4 Architecture notebook

4.1 Key requirements and constrains

The OLSheets platform must satisfy several critical architectural characteristics that directly influence design decisions. Security is paramount, requiring robust JWT-based authentication, role-based authorization (renters, owners, administrators), and secure handling of payment information. All sensitive data must be encrypted in transit and at rest, with proper session management to prevent unauthorized access. Scalability requirements dictate a stateless application design to enable horizontal scaling, with database connection pooling and caching strategies to handle increased load as the user base grows. The system must support concurrent booking requests without data inconsistencies, necessitating proper transaction management and optimistic locking mechanisms.

Reliability and availability are essential for maintaining user trust, particularly for booking confirmations and payment processing. The system implements health checks across all infrastructure components (application, database, monitoring stack) and includes automated alerting through Prometheus and AlertManager with Slack integration. Database backups and recovery procedures ensure data durability, while the booking system employs idempotent operations to handle potential network failures gracefully. Testability is achieved through dependency injection and interface-based design, enabling comprehensive unit testing, integration testing, and end-to-end testing strategies as required by the TQS curriculum.

Observability requirements drive the integration of a comprehensive monitoring stack including Prometheus for metrics collection, Grafana for visualization, and Loki with Promtail for centralized log aggregation. Application-level metrics track booking rates, payment success rates, and search performance, while infrastructure metrics monitor JVM health, database connections, and API response times. This monitoring infrastructure enables proactive issue detection and supports performance optimization efforts.

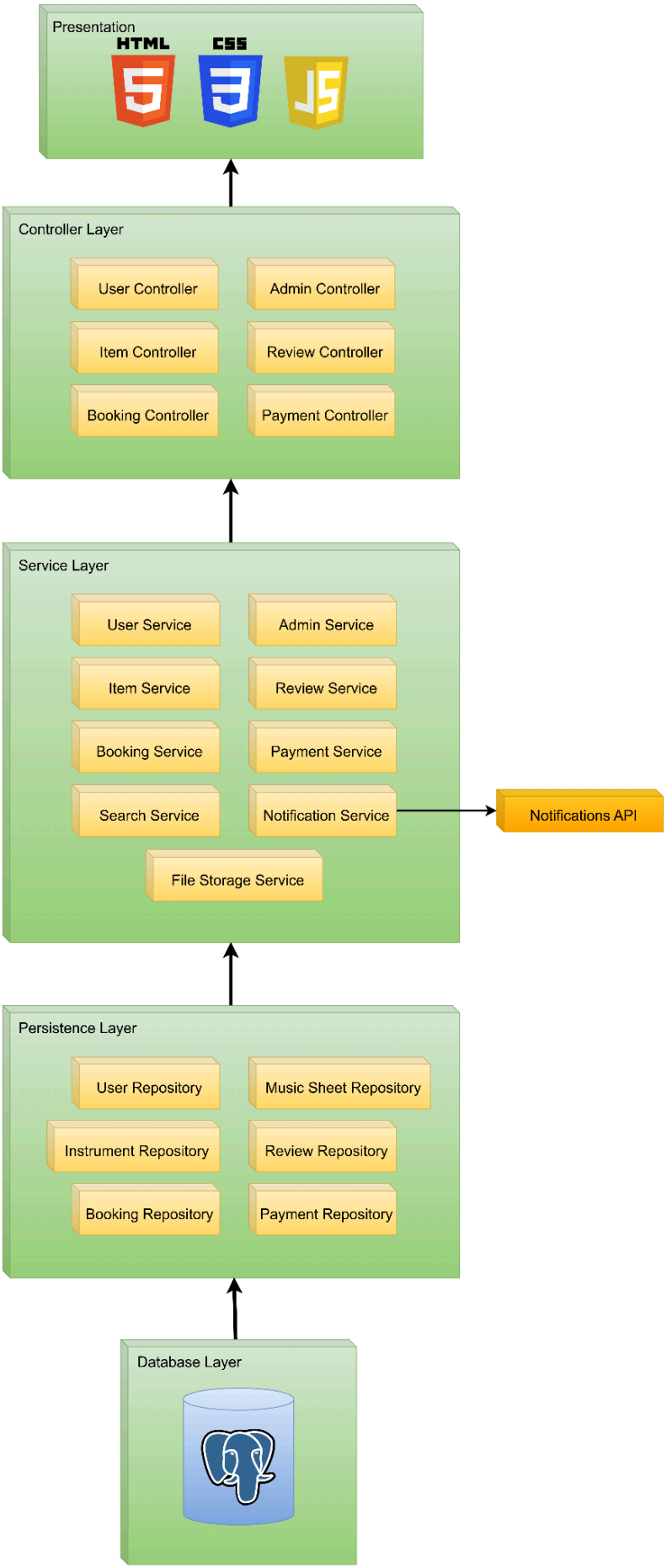
4.2 Architecture view

The OLSheets platform follows a layered architecture pattern with clear separation of concerns across four distinct layers: Presentation, Controller, Service, and Persistence. This architectural style promotes maintainability, testability, and scalability by enforcing unidirectional dependencies where each layer only communicates with the layer directly below it. The presentation layer handles all user-facing interactions through HTML, CSS, and JavaScript, while the controller layer manages HTTP request routing and response handling through dedicated controllers for each domain entity (User, Item, Booking, Payment, Review).

The service layer implements the core business logic and orchestrates interactions between different domain components. This layer includes specialized services such as the Search Service for item discovery, the Notification Service for user communications (integrated with external APIs), and the File Storage Service for managing instrument photos and sheet music annexes. The service layer acts as the primary integration point for external systems, including payment processing APIs and notification services, ensuring that these dependencies are properly abstracted from the rest of the application.

The persistence layer manages all data access operations through Spring Data JPA repositories, providing a clean abstraction over the PostgreSQL database. Each domain entity has its corresponding repository (User, Instrument, Music Sheet, Booking, Payment, and Review), enabling

efficient CRUD operations and complex queries. The database layer utilizes PostgreSQL for relational data storage with proper indexing strategies to support the search and filtering requirements of the platform.


Presentation
HTML
CSS
Controller Layer
User Controller
Admin Controller
Item Controller
Review Controller
Booking Controller
Payment Controller
Service Layer
User Service
Admin Service
Item Service
Review Service
Booking Service
Payment Service
Search Service
Notification Service
Notifications API
File Storage Service
Persistence Layer
User Repository
Music Sheet Repository
Instrument Repository
Review Repository
Booking Repository
Payment Repository
Database Layer

4.3 Deployment view (production configuration)

The production deployment uses Docker containers orchestrated with Docker Compose:

- Application Container:
  - Spring Boot application running on Java 21
  - Exposed on port 8080
  - Configured for PostgreSQL database
  - Health checks for monitoring

- PostgreSQL Database Container:
  - Persistent volume for data
  - Health checks to ensure availability
  - Environment-specific configuration

- Monitoring Stack:
  - Prometheus for metrics collection
  - Grafana for visualization
  - Loki for log aggregation
  - Promtail for log shipping

- Alerting:
  - AlertManager for alert management
  - Configured for Slack notifications

The application is configured to listen on all interfaces within the Docker network, with the database connection properly secured. The monitoring stack collects application metrics, JVM metrics, and database performance indicators.

5 API for developers

The OLSheets platform provides a RESTful API with the following main groups:

- Authentication Endpoints
- Instrument Management
- Music Sheet Management
- Booking Management
- Payment Processing
- Reviews
- Availability Management

The API follows REST conventions with appropriate HTTP methods (GET, POST, PUT, DELETE), resource-oriented URLs, and standard HTTP status codes. Authentication is required for most endpoints using JWT tokens passed in the Authorization header.

Request and response bodies use JSON format, with proper error responses including error messages and status codes. The API is designed to be consumed by web clients but follows principles that would also support mobile applications in future releases.

More detail to be found on the Swagger docs page (/docs) in the platform