Business Requirements Document: Art Gallery Backend Service

1. Introduction

1.1 Purpose

The purpose of this document is to outline the business requirements for the development of a backend service for the art gallery of aboriginal art of Australia.

1.2 Scope

The project aims to develop a RESTful API backend service to manage various aspects of the art gallery, including artists, artifacts, users, and other related entities.

2. Business Objectives

2.1 Primary Objectives

- Develop a scalable and efficient backend service to support CRUD operations for artists, artifacts, and users.
- Ensure data integrity and security by implementing authentication and authorization mechanisms.
- Facilitate seamless integration with a frontend application for a user-friendly experience.
- Optimize performance to handle a large volume of concurrent requests.
- Enable easy maintenance and future enhancements through well-structured code and documentation.

2.2 Secondary Objectives

- Implement comprehensive error handling and logging for better debugging and troubleshooting.
- Provide extensive documentation for developers, including API endpoints, request/response formats, and usage examples.
- Support internationalization and localization for multilingual support in the frontend application.
- Implement caching mechanisms to improve response times and reduce server load.

3. Functional Requirements

3.1 Artists

- Create, retrieve, update, and delete artists.
- Retrieve a list of all artists.
- Search artists by name.

3.2 Artifacts

- Create, retrieve, update, and delete artifacts.
- Retrieve a list of all artifacts.
- Search artifacts by title.
- Associate artifacts with artists.

3.3 Users

- Create, retrieve, update, and delete users.
- Retrieve a list of all users.
- Search users by username or email.

4. Non-Functional Requirements

4.1 Performance

- Ensure API response times are within acceptable limits, even under heavy load.

- Implement efficient database queries and indexing to optimize performance.

4.2 Security

- Implement authentication using OAuth 2.0 to secure API endpoints.

- Enforce authorization to restrict access based on user roles and permissions.

4.3 Reliability

- Minimize downtime by implementing error recovery mechanisms and regular backups.

4.4 Scalability

- Design the backend service to scale horizontally to handle increased traffic and data volume.

5. User Stories

5.1 As an art enthusiast, I want to view a list of all artists in the gallery.

5.2 As a curator, I want to add new artifacts to the gallery and associate them with artists.

5.3 As a user, I want to create an account and log in to access personalized features.

6. Aggregate Design Canvases

6.1 Artist Aggregate

- Entities: Artist

- Attributes: Name, Bio

- Actions: Create, Read, Update, Delete

6.2 Artifact Aggregate

- Entities: Artifact, Artist

- Attributes: Title, Description, Artist ID

- Actions: Create, Read, Update, Delete

6.3 User Aggregate

- Entities: User

- Attributes: Username, Email

- Actions: Create, Read, Update, Delete

.