Architecture Document

Version	Date	Author(s)	Changes	State
1.0	16/09/2024	ReSoTo group	Added C1 and C2 diagrams	In progress
2.0	09/10/2024	ReSoTo group	Added C2 version 2 and C3	In progress
3.0	18/11/2024	ReSoTo group	Added C2 version 3 and C3 version 2	In progress

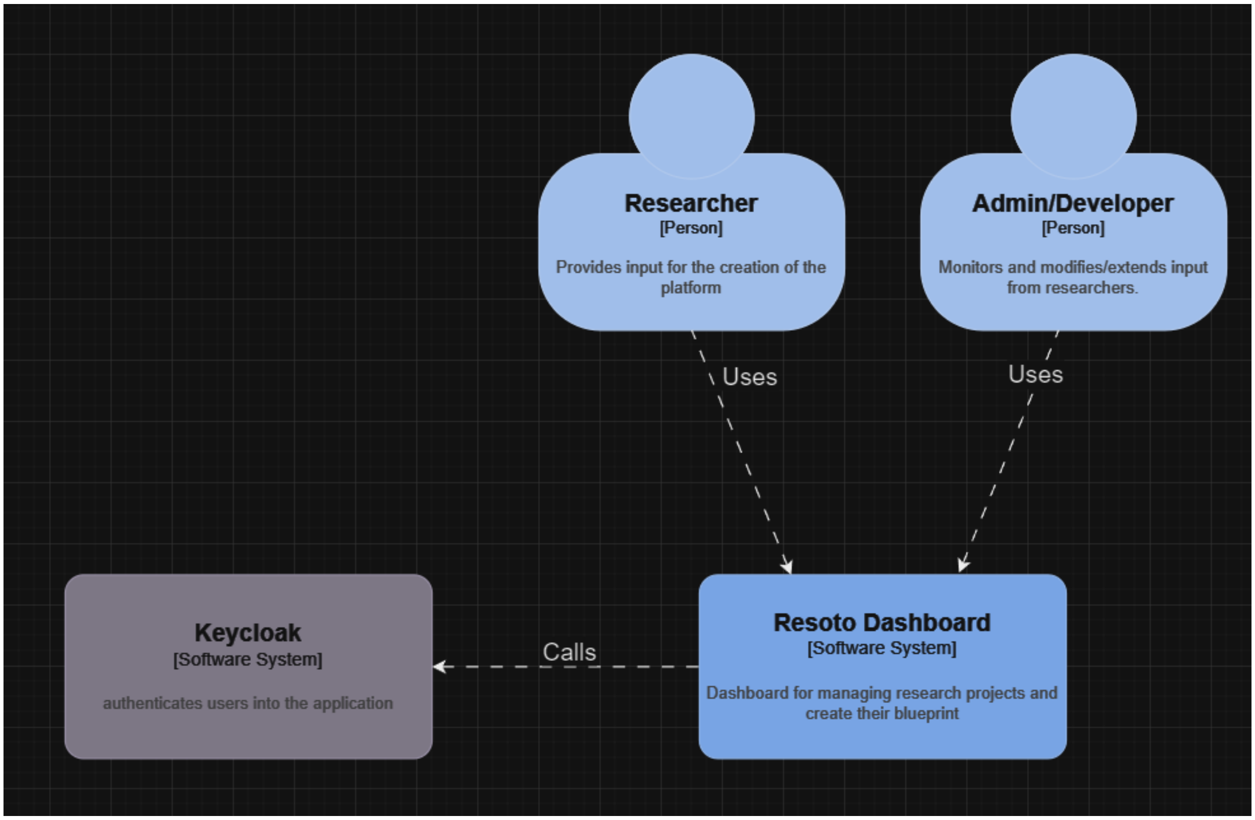
Introduction

In this document, we are going to show different architecture diagrams that the ReSoTo project will follow. During the implementation of the project, some of the architectural decisions may change based on feedback or requirements change. In this document, you can see all of the versions of the architecture.

C4 Diagrams

C1

The C1 diagram showcases the interaction between researchers and admin/developers with the **Resoto Dashboard** on a high level. Both roles use the **Resoto Dashboard** to perform their functions, including accessing the questionnaire, seeing a dashboard with the projects, etc. **Keycloak**, a separate authentication system, manages user access, ensuring that only authorized individuals can use the platform by authenticating them before granting access to the **Resoto Dashboard**.

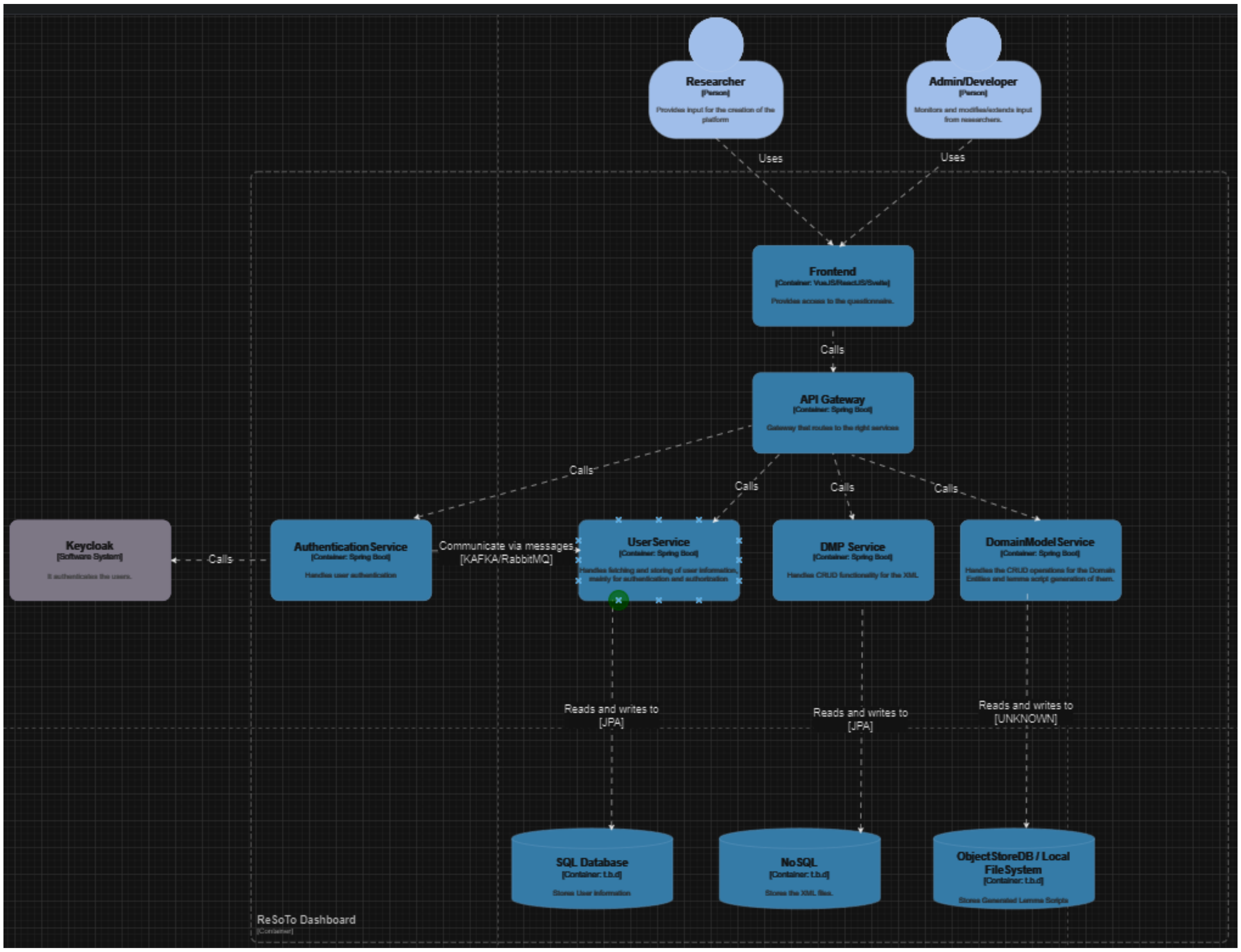

Researcher
[Person]
Provides input for the creation of the platform
Admin/Developer
[Person]
Monitors and modifies/extends input from researchers.
Uses
Uses
Keycloak
[Software System]
authenticates users into the application
Calls
Resoto Dashboard
[Software System]
Dashboard for managing research projects and create their blueprint

C2

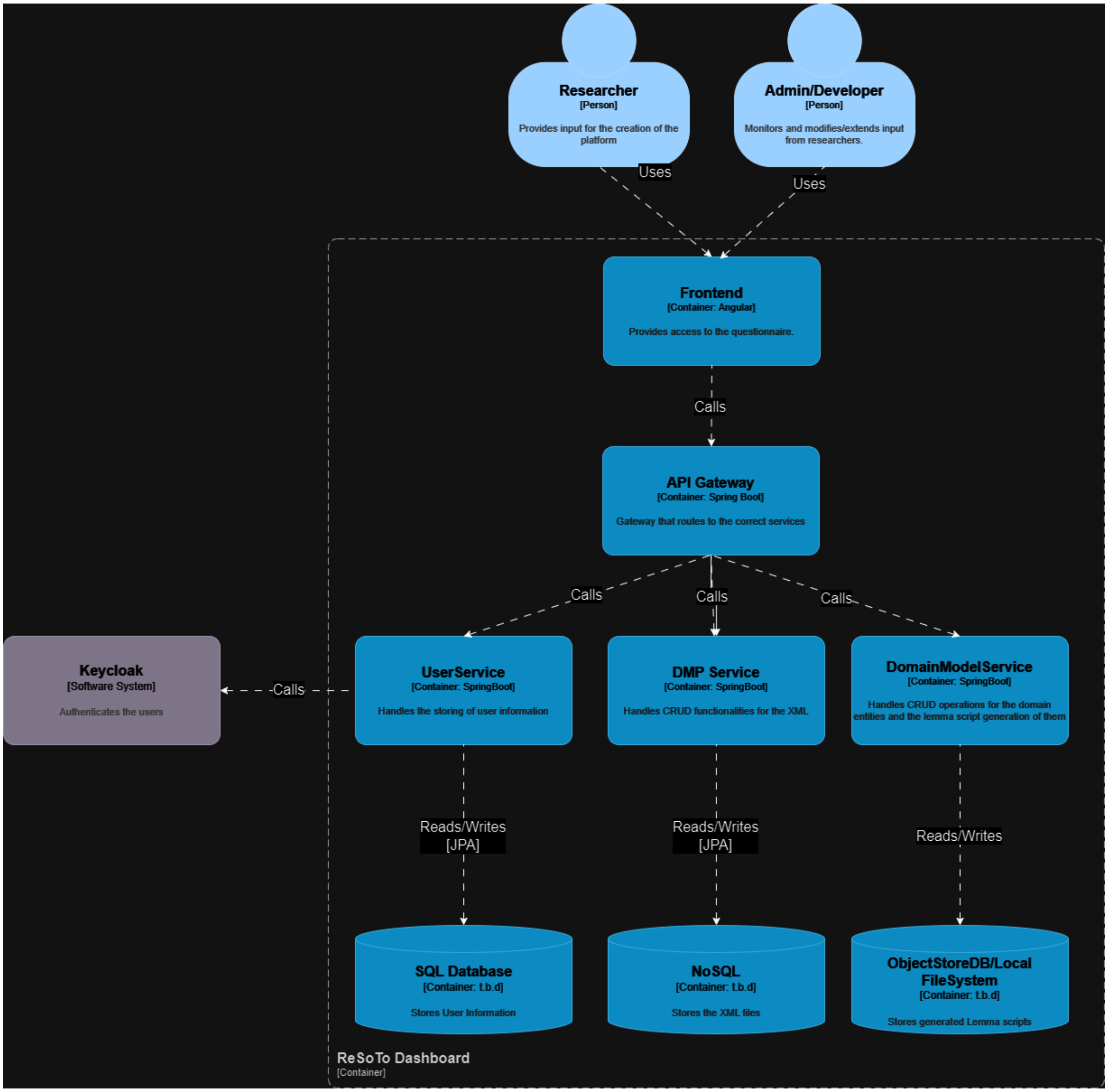
The ReSoTo system illustrated in the diagrams is a platform designed to facilitate interactions between researchers and admins/developers, integrating various microservices for seamless operation. The **frontend**, built with Angular, serves as the interface through which users interact, while the **API Gateway** (Spring Boot) manages and routes API requests to the appropriate microservices. These microservices include the **UserService**, responsible for storing user data in an **SQL Database**, the **DMP Service**, handling CRUD operations for XML files stored in a **NoSQL database**, and the **DomainModelService**, which manages domain entities and saves them in a **MongoDB**. The last microservice is the LemmaService that handles the LEMMA scripts generation and stores them in an Azure Blob storage. User authentication is handled by **Keycloak**, ensuring that only authorized individuals access the system. The platform is structured to ensure scalability, security, and efficient data handling, with each component focused on a specific business function, making the system modular and maintainable.

The only difference between the Version 1 and 2 of the C2 diagram is the AuthenticationService that was marked as redundant. Version 3 includes the correct connection between the API Gateway and Keycloak. In Version 4, the LemmaService is being introduced.

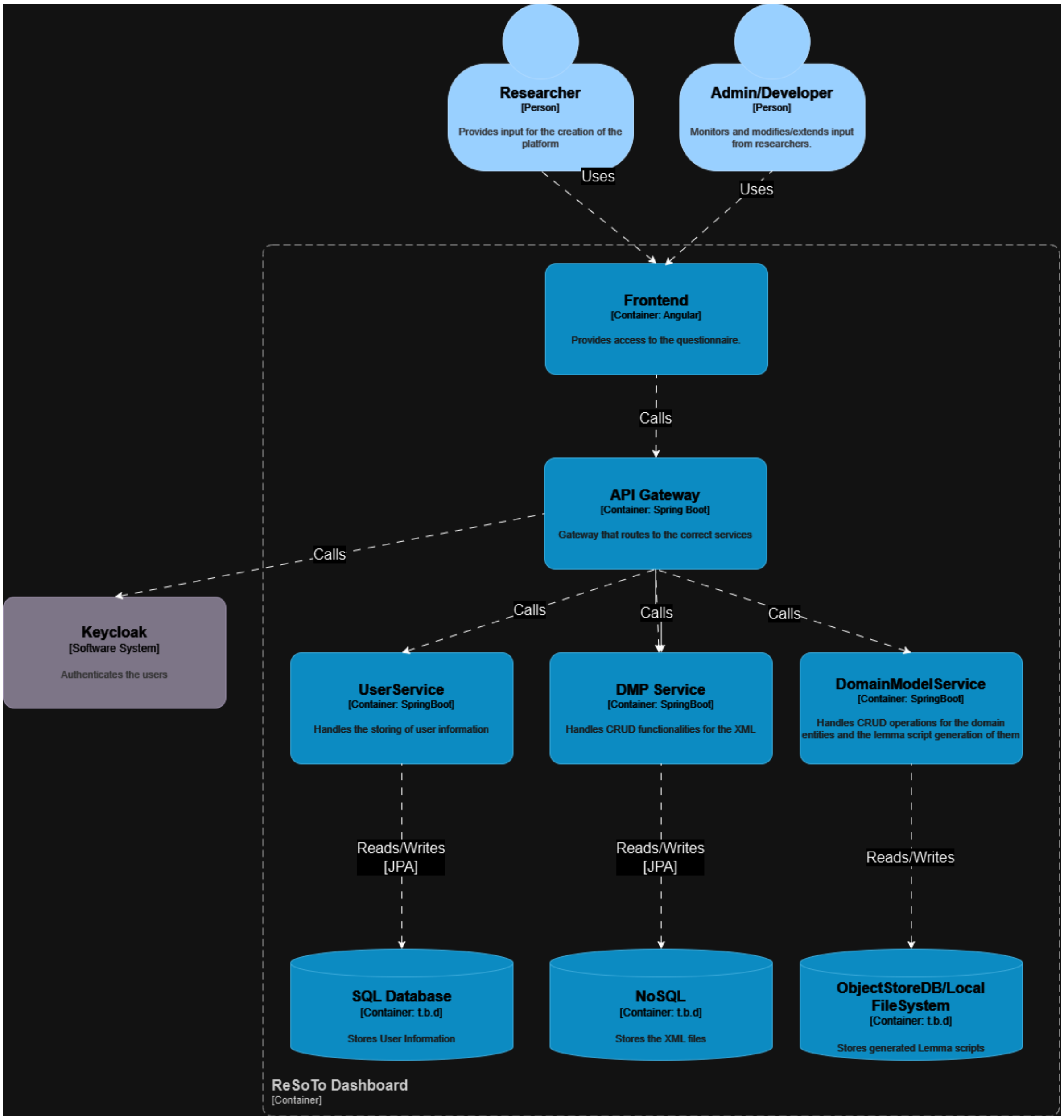
Version 1


Researcher
[Person]
Provides input for the creation of the platform
Admin/Developer
[Person]
Monitors and modifies/extends input from researchers.
Uses
Uses
Frontend
[Container: VueJS/ReactJS/Svelte]
Provides access to the questionnaire.
Calls
API Gateway
[Container: Spring Boot]
Gateway that routes to the right services
Calls
Calls
Calls
Calls
Keycloak
[Software System]
It authenticates the users.
Calls
Authentication Service
[Container: Spring Boot]
Handles user authentication
Communicate via messages
[KAFKA/RabbitMQ]
UserService
[Container: Spring Boot]
Handles fetching and storing of user information, mainly for authentication and authorization
DMP Service
[Container: Spring Boot]
Handles CRUD functionality for the XML
DomainModel Service
[Container: Spring Boot]
Handles the CRUD operations for the Domain Entities and lemma script generation of them.
Reads and writes to
[JPA]
Reads and writes to
[JPA]
Reads and writes to
[UNKNOWN]
SQL Database
[Container: t.b.d]
Stores User information
No SQL
[Container: t.b.d]
Stores the XML files.
ObjectStoreDB / Local FileSystem
[Container: t.b.d]
Stores Generated Lemma Scripts
ReSoTo Dashboard
[Container]

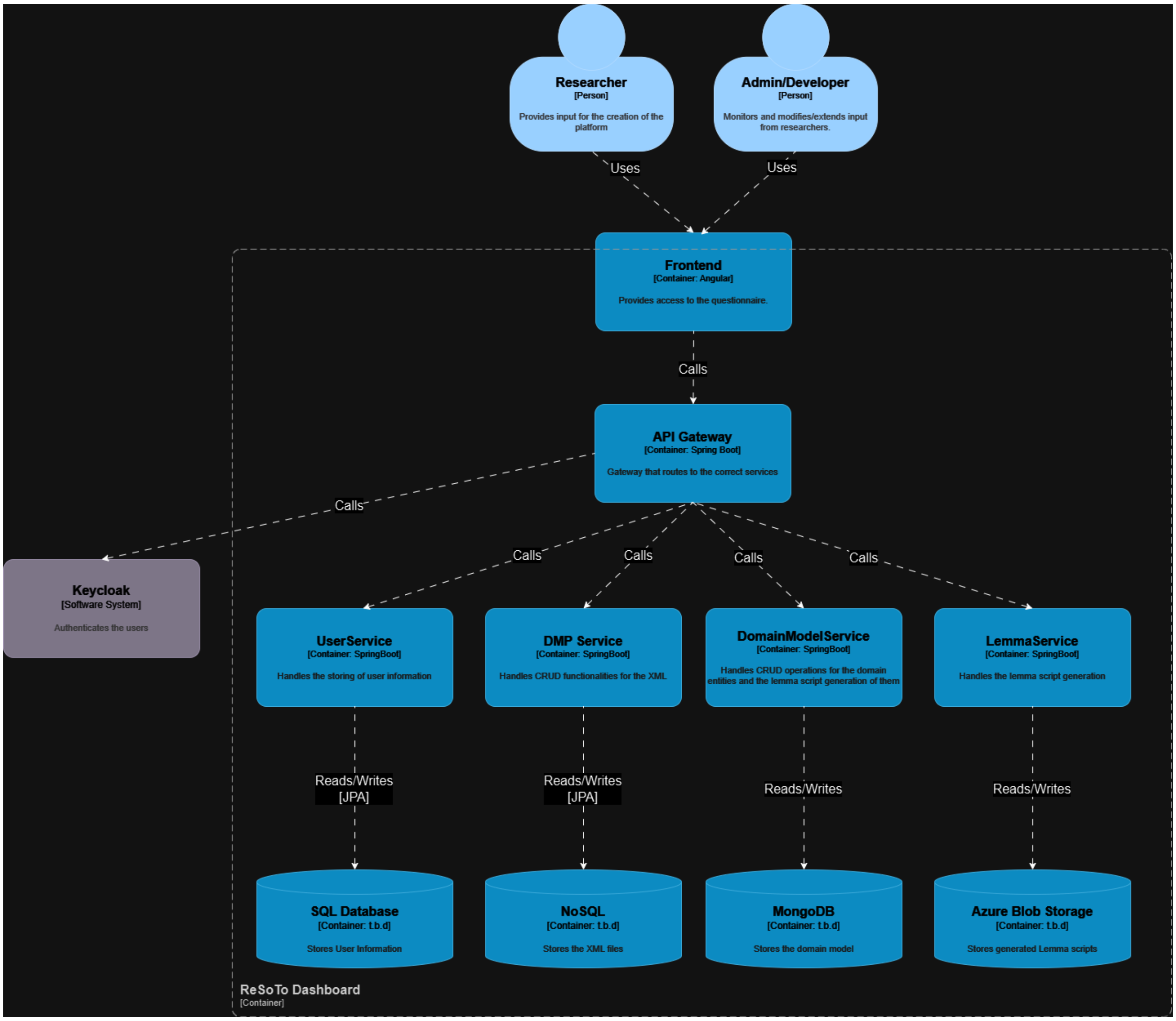
Version 2



Version 3


Researcher
[Person]
Provides input for the creation of the platform
Admin/Developer
[Person]
Monitors and modifies/extends input from researchers.
Uses
Uses
Frontend
[Container: Angular]
Provides access to the questionnaire.
Calls
API Gateway
[Container: Spring Boot]
Gateway that routes to the correct services
Calls
Keycloak
[Software System]
Authenticates the users
Calls
Calls
Calls
UserService
[Container: SpringBoot]
Handles the storing of user information
DMP Service
[Container: SpringBoot]
Handles CRUD functionalities for the XML
DomainModelService
[Container: SpringBoot]
Handles CRUD operations for the domain entities and the lemma script generation of them
Reads/Writes
[JPA]
Reads/Writes
[JPA]
Reads/Writes
SQL Database
[Container: t.b.d]
Stores User Information
NoSQL
[Container: t.b.d]
Stores the XML files
ObjectStoreDB/Local FileSystem
[Container: t.b.d]
Stores generated Lemma scripts
ReSoTo Dashboard
[Container]

Version 4



C3

The C3 diagram allows us to see the system in a component level. The **frontend** (built with Angular) serves as the user interface, calling an **API Gateway** (Spring Boot) that routes API requests to appropriate services. **Keycloak** handles user authentication by validating credentials before allowing access to the system.

The platform consists of four main services:

1. **UserService** handles user data storage and operations, communicating with an **SQL Database**.
2. **DMP Service** is responsible for CRUD operations on XML data, stored in a **NoSQL Database**.
3. **DomainModelService** manages domain entities, stored in a MongoDB.

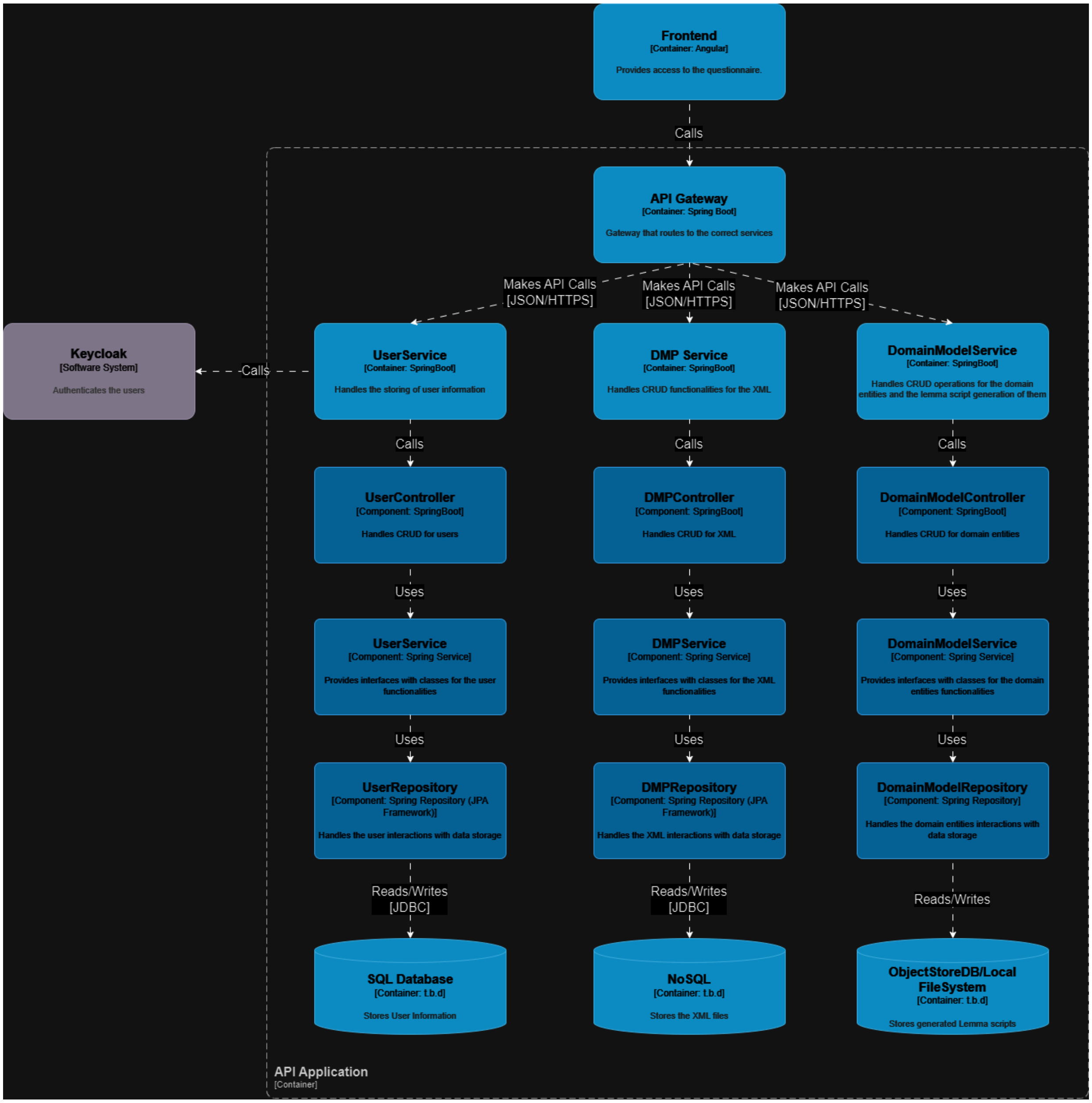
4. **LemmaService** handles the LEMMA scripts generation and saves them in an Azure Blob storage.

Each service consists of three main components:

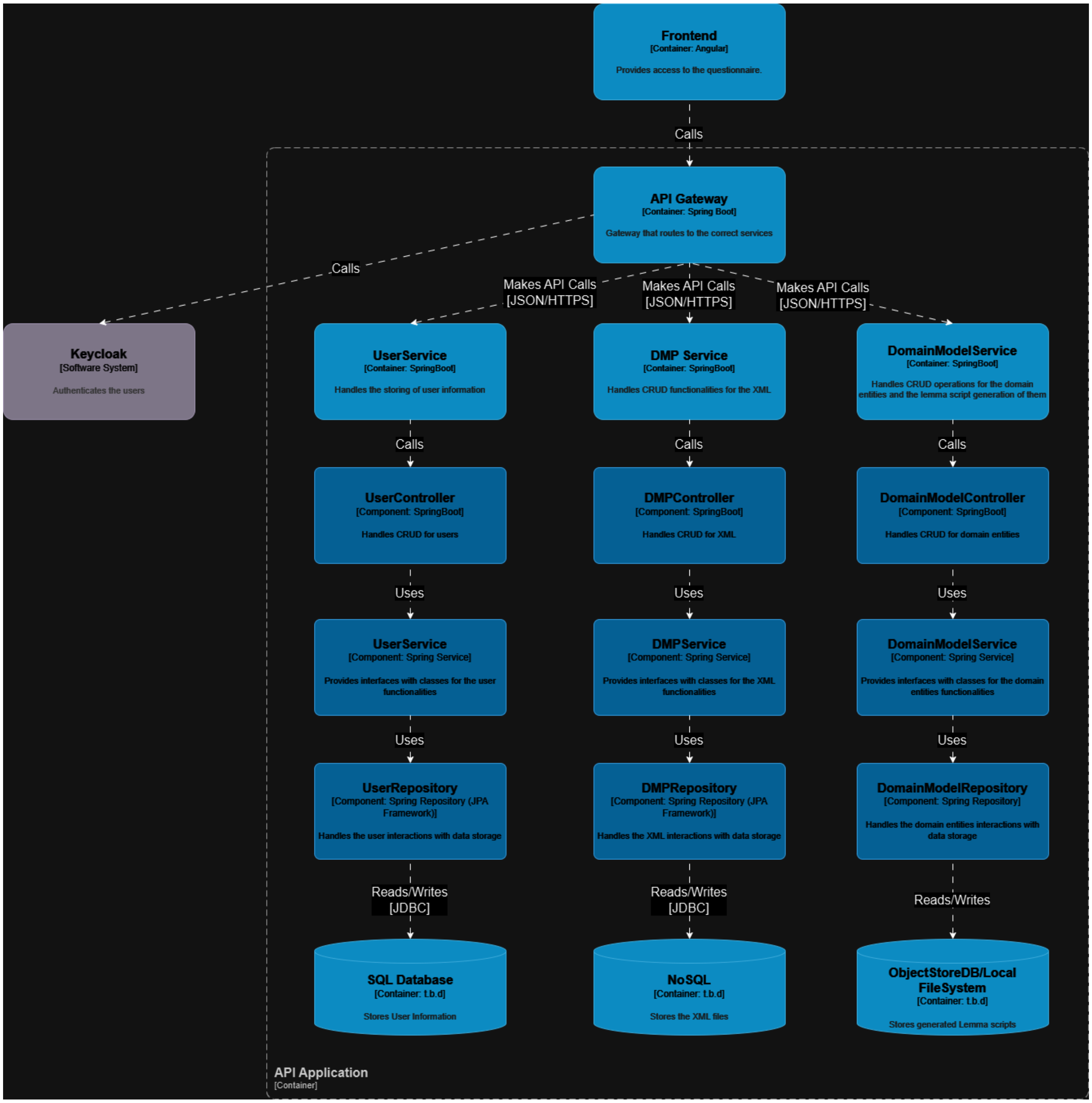
- The **Controller** (Spring Boot) handles incoming requests,
- The **Service** (Spring Service) processes business logic, and
- The **Repository** (Spring Repository) interacts with the respective databases.

These services communicate using JSON/HTTPS, providing a clear separation of concerns, modularity, and scalability within the system.

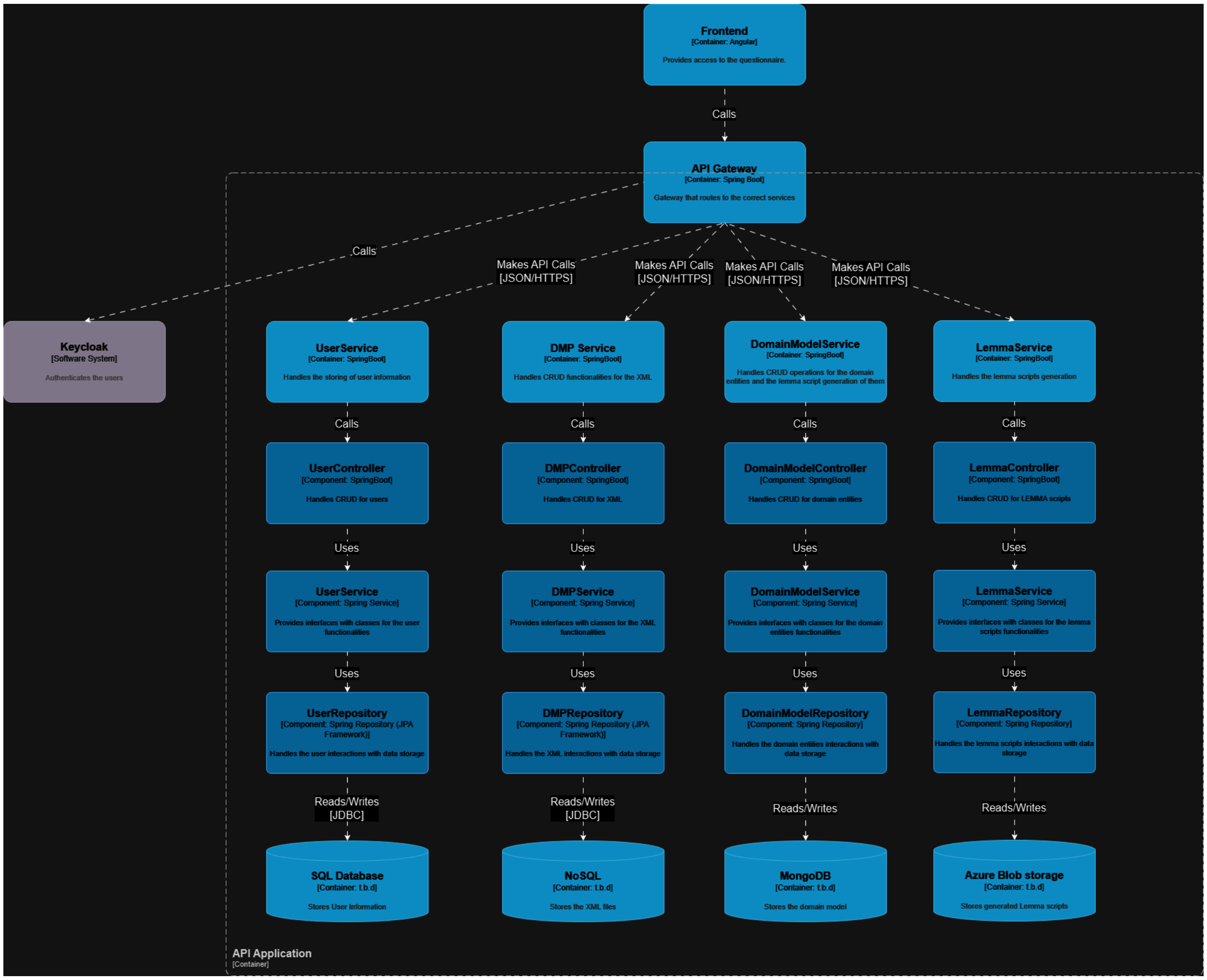
Version 1


Frontend
[Container: Angular]
Provides access to the questionnaire.
Calls
API Gateway
[Container: Spring Boot]
Gateway that routes to the correct services
Makes API Calls
[JSON/HTTPS]
Makes API Calls
[JSON/HTTPS]
Makes API Calls
[JSON/HTTPS]
Keycloak
[Software System]
Authenticates the users
Calls
UserService
[Container: SpringBoot]
Handles the storing of user information
DMP Service
[Container: SpringBoot]
Handles CRUD functionalities for the XML
DomainModelService
[Container: SpringBoot]
Handles CRUD operations for the domain entities and the lemma script generation of them
Calls
Calls
Calls
UserController
[Component: SpringBoot]
Handles CRUD for users
DMPController
[Component: SpringBoot]
Handles CRUD for XML
DomainModelController
[Component: SpringBoot]
Handles CRUD for domain entities
Uses
Uses
Uses
UserService
[Component: Spring Service]
Provides interfaces with classes for the user functionalities
DMPService
[Component: Spring Service]
Provides interfaces with classes for the XML functionalities
DomainModelService
[Component: Spring Service]
Provides interfaces with classes for the domain entities functionalities
Uses
Uses
Uses
UserRepository
[Component: Spring Repository (JPA Framework)]
Handles the user interactions with data storage
DMPRepository
[Component: Spring Repository (JPA Framework)]
Handles the XML interactions with data storage
DomainModelRepository
[Component: Spring Repository]
Handles the domain entities interactions with data storage
Reads/Writes
[JDBC]
Reads/Writes
[JDBC]
Reads/Writes
SQL Database
[Container: t.b.d]
Stores User Information
NoSQL
[Container: t.b.d]
Stores the XML files
ObjectStoreDB/Local FileSystem
[Container: t.b.d]
Stores generated Lemma scripts
API Application
[Container]

Version 2


Frontend
[Container: Angular]
Provides access to the questionnaire.
Calls
API Gateway
[Container: Spring Boot]
Gateway that routes to the correct services
Calls
Makes API Calls [JSON/HTTPS]
Makes API Calls [JSON/HTTPS]
Makes API Calls [JSON/HTTPS]
Keycloak
[Software System]
Authenticates the users
UserService
[Container: SpringBoot]
Handles the storing of user information
DMP Service
[Container: SpringBoot]
Handles CRUD functionalities for the XML
DomainModelService
[Container: SpringBoot]
Handles CRUD operations for the domain entities and the lemma script generation of them
Calls
Calls
Calls
UserController
[Component: SpringBoot]
Handles CRUD for users
DMPController
[Component: SpringBoot]
Handles CRUD for XML
DomainModelController
[Component: SpringBoot]
Handles CRUD for domain entities
Uses
Uses
Uses
UserService
[Component: Spring Service]
Provides interfaces with classes for the user functionalities
DMPService
[Component: Spring Service]
Provides interfaces with classes for the XML functionalities
DomainModelService
[Component: Spring Service]
Provides interfaces with classes for the domain entities functionalities
Uses
Uses
Uses
UserRepository
[Component: Spring Repository (JPA Framework)]
Handles the user interactions with data storage
DMPRepository
[Component: Spring Repository (JPA Framework)]
Handles the XML interactions with data storage
DomainModelRepository
[Component: Spring Repository]
Handles the domain entities interactions with data storage
Reads/Writes [JDBC]
Reads/Writes [JDBC]
Reads/Writes
SQL Database
[Container: t.b.d]
Stores User Information
NoSQL
[Container: t.b.d]
Stores the XML files
ObjectStoreDB/Local FileSystem
[Container: t.b.d]
Stores generated Lemma scripts
API Application
[Container]

Version 3


Frontend
[Container: Angular]
Provides access to the questionnaire.
Calls
API Gateway
[Container: Spring Boot]
Gateway that routes to the correct services
Calls
Makes API Calls
[JSON/HTTPS]
Makes API Calls
[JSON/HTTPS]
Makes API Calls
[JSON/HTTPS]
Makes API Calls
[JSON/HTTPS]
Keycloak
[Software System]
Authenticates the users
UserService
[Container: SpringBoot]
Handles the storing of user information
DMP Service
[Container: SpringBoot]
Handles CRUD functionalities for the XML
DomainModelService
[Container: SpringBoot]
Handles CRUD operations for the domain entities and the lemma script generation of them
LemmaService
[Container: SpringBoot]
Handles the lemma scripts generation
Calls
Calls
Calls
Calls
UserController
[Component: SpringBoot]
Handles CRUD for users
DMPController
[Component: SpringBoot]
Handles CRUD for XML
DomainModelController
[Component: SpringBoot]
Handles CRUD for domain entities
LemmaController
[Component: SpringBoot]
Handles CRUD for LEMMA scripts
Uses
Uses
Uses
Uses
UserService
[Component: Spring Service]
Provides interfaces with classes for the user functionalities
DMPService
[Component: Spring Service]
Provides interfaces with classes for the XML functionalities
DomainModelService
[Component: Spring Service]
Provides interfaces with classes for the domain entities functionalities
LemmaService
[Component: Spring Service]
Provides interfaces with classes for the lemma scripts functionalities
Uses
Uses
Uses
Uses
UserRepository
[Component: Spring Repository (JPA Framework)]
Handles the user interactions with data storage
DMPRepository
[Component: Spring Repository (JPA Framework)]
Handles the XML interactions with data storage
DomainModelRepository
[Component: Spring Repository]
Handles the domain entities interactions with data storage
LemmaRepository
[Component: Spring Repository]
Handles the lemma scripts interactions with data storage
Reads/Writes
[JDBC]
Reads/Writes
[JDBC]
Reads/Writes
Reads/Writes
SQL Database
[Container: t.b.d]
Stores User Information
NoSQL
[Container: t.b.d]
Stores the XML files
MongoDB
[Container: t.b.d]
Stores the domain model
Azure Blob storage
[Container: t.b.d]
Stores generated Lemma scripts
API Application
[Container]