# Clay doors assessment overview

| | |
|---|---|
| Created | @February 23, 2024 2:08 PM |
| Last Edited Time | @March 1, 2024 2:42 AM |
| Type | Architecture Overview |
| Created By | Reham Ali |
| Last Edited By | Reham Ali |

## Description

This api is focused on the concepts of door access and permissions, it uses a combination of Role-Based Access Control (RBAC) and Permission-Based Access Control (PBAC), for greater flexibility

## Assumptions

On user can be associated with Many doors, and vice versa,

A user can have a set of roles, those roles have their permissions

they're are fixed permissions for fundamental features such as

OpenDoor
ViewLogs
GrantPermission
RevokePermission

The design allows for more to be added according to needs

some doors allow remote access

some permissions are temporary

access control can be by user or role, for more flexibility and convenience

## Architecture

in this project, clean architecture and repository are used to ensure separation of concerns, testability, flexibility and maintainability

### solution structure

**DoorManagementSystem.API**

contains controllers, middleware and input models.

**DoorManagementSystem.Application**

contains interfaces, DTOs, mappers, and services.

**DoorManagementSystem.Domain**

contains entities

**DoorManagementSystem.Infrastructure**

contains db context, and repositories

**DoorManagementSystem.Test**

contains unit tests

# Technical Decisions:

## Framework:

**.Net 8**, as it’s the latest long-term support version of .Net, which includes performance enhancements compared to older versions, and it uses the latest c# version

## Database:

**EF core 8 and PostgreSQL,** scalable, reliable and maintainable.

## CI/CD

although the project is not technically deployed, but it was prepared for deployment by implementing a build workflow on github and a dockerfile

## Authentication and Authorization

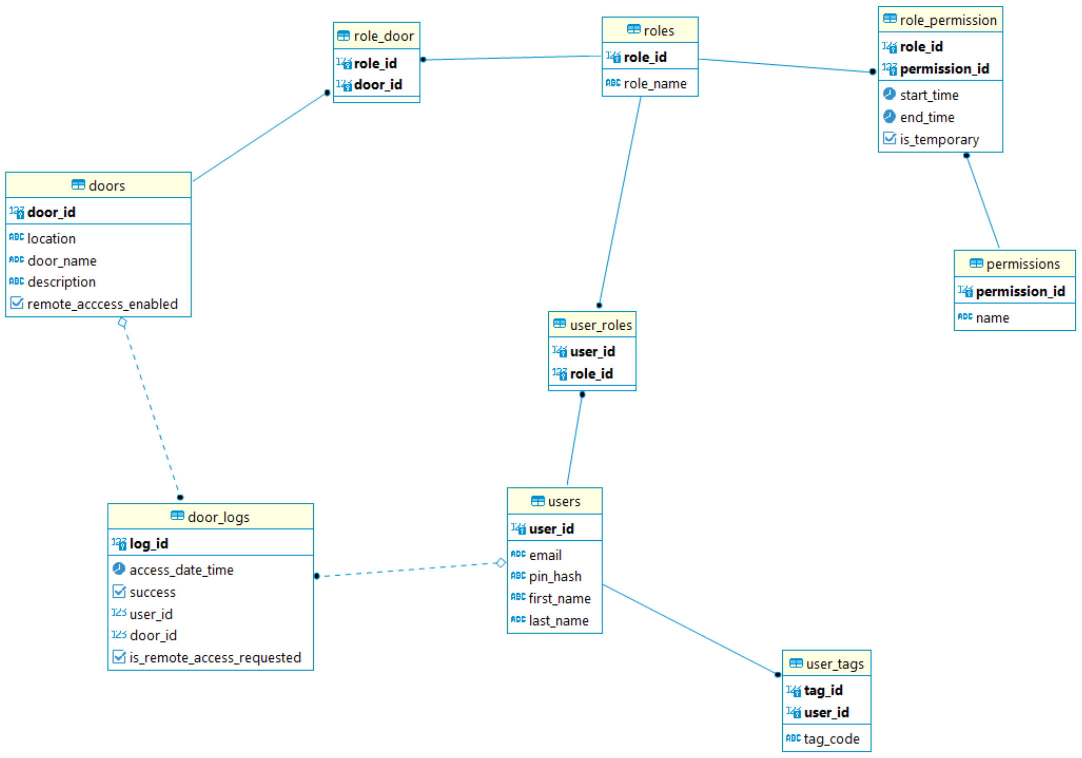
for **authentication**, JWT is used, for it’s stateless authentication, scalability and security and many benefits, JWT can include using claims, which allows fine grained access control,

these claims are used for user **authorization**, for actions that requires certain system permissions.

## UI testing

during development, swagger was used for manual testing

## Database Diagram



## Client

- practically anyone with the clay Locks, as the system offers great amount of access and roles flexibility