System Design Document
for
Catalyst

Version 2.0

Authors: Amir Ali Mohaghegh, Barbod Habibi, and Kevin Nguyen

Date: 2024-12-3

Table of Contents

CRC Cards - 1

Software Architecture Diagram - 2

CRC Cards

Class Name: CatalystApplication	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Runs the main program through the spring framework. It contains the main method.	**Collaborators:** N/A

Controllers:

Class Name: NavigationController	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Map general URLs to HTML templates.	**Collaborators:** - Thymeleaf templates: index.html, assessments.html.

Class Name: OptometristController	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Handles HTTP requests, redirections, and renders appropriate Thymeleaf templates associated with optometrist resources. It delegates CRUD operations to the appropriate service layer and returns relevant outputs to the user.	**Collaborators:** - OptometristService: Handles business logic and invokes data interactions. - Model: Supplies data to Thymeleaf templates. - Optometrist: Represents the entity being managed. - BindingResult: Captures validation results. - Thymeleaf templates: optometrists.html, optometrist-details.html.

Class Name: PatientController	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Handles HTTP requests, redirections, and renders appropriate Thymeleaf templates associated with patient resources. It delegates CRUD operations to	**Collaborators:** - PatientsService: Handles business logic and invokes data interactions. - Model: Supplies data to Thymeleaf

the appropriate service layer and returns relevant outputs to the user.	templates. - Optometrist: Represents the entity being managed. - BindingResult: Captures validation results. - Thymeleaf templates: optometrists.html, optometrist-details.html.

Class Name: AuthController	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Handles HTTP requests, redirections for user authentication.	**Collaborators:** - UserService: used for user-related operations, if needed in the future.

Services:

Interface Name: OptometristService	
Parent Class: N/A **Subclasses:** OptometristServiceImpl (implementation)	
Responsibilities: Provides an abstraction layer for optometrist resource CRUD operations and decouples the controller from concrete service implementations.	**Collaborators:** - OptometristController: Relies on this interface to execute business logic. - OptometristServiceImpl: Implements the interface methods and is injected into the controller.

Interface Name: PatientService	
Parent Class: N/A **Subclasses:** PatientServiceImpl (implementation)	
Responsibilities: Provides an abstraction layer for patient resource CRUD operations and decouples the controller from concrete service implementations.	**Collaborators:** - PatientController: Relies on this interface to execute business logic. - PatientServiceImpl: Implements the interface methods and is injected into the controller.

Interface Name: AssessmentService
Parent Class: N/A **Subclasses:** AssessmentServiceImpl (implementation)

Responsibilities: Provide an abstraction for operations related to visual assessments. Define methods to retrieve, save, and delete assessments.	**Collaborators:** - AssessmentServiceImpl: implements the service logic. - AssessmentRepository: performs database operations.

Interface Name: UserService	
Parent Class: N/A **Subclasses:** UserServiceImpl (implementation)	
Responsibilities: Provide an abstraction for user-related operations, such as finding a user by their email.	**Collaborators:** - UserServiceImpl: implements the actual service logic. - UserRepository: performs database operations.

Service Implementations:

Class Name: OptometristServiceImpl	
Parent Class: OptometristService **Subclasses:** N/A	
Responsibilities: Provides concrete implementation for optometrist resource CRUD operations.	**Collaborators:** - OptometristRepository: Carries out the database operations requested by this class. - OptometristService: Provides the blueprint of the business logic that must be implemented.

Class Name: PatientServiceImpl	
Parent Class: PatientService **Subclasses:** N/A	
Responsibilities: Provides concrete implementation for patient resource CRUD operations.	**Collaborators:** - PatientRepository: Carries out the database operations requested by this class. - PatientService: Provides the blueprint of the business logic that must be implemented.

Class Name: AssessmentServiceImpl	
Parent Class: AssessmentService **Subclasses:** N/A	
Responsibilities: Implement the business logic for managing visual assessments. Use the repository to retrieve, save, and delete assessments.	**Collaborators:** - AssessmentRepository: handles database operations for assessments. - VisualAssessment: the entity representing a visual assessment.

Class Name: UserServiceImpl	
Parent Class: PatientService **Subclasses:** N/A	
Responsibilities: Implement the logic for user-related operations defined in UserService. Fetch user information from the database using UserRepository.	**Collaborators:** - UserRepository: handles database operations for user data.

Repositories:

Interface Name: OptometristRepository	
Parent Class: JpaRepository<Optometrist, Integer> **Subclasses:** N/A	
Responsibilities: Provides database access methods for optometrist CRUD.	**Collaborators:** - OptometristServiceImpl: Uses this repository from Spring Data JPA to interact with the database. - Optometrist: The entity managed by this repository.

Interface Name: PatientRepository	
Parent Class: JpaRepository<Patient, Integer> **Subclasses:** N/A	
Responsibilities: Provides database access methods for patient CRUD.	**Collaborators:** - PatientServiceImpl: Uses this repository from Spring Data JPA to interact with the database. - Patient: The entity managed by this repository.

Interface Name: AssessmentRepository	
Parent Class: JpaRepository<Assessment, Integer> **Subclasses:** N/A	
Responsibilities: Perform CRUD operations on VisualAssessment entities in the database.	**Collaborators:** - VisualAssessment: the entity managed by this repository.

Interface Name: UserRepository	
Parent Class: JpaRepository<User, Integer> **Subclasses:** N/A	
Responsibilities: Perform database operations for user-related entities. Find a user by their email.	**Collaborators:** - UserEntity: the entity representing the user in the database.

Models:

Class Name: Optometrist	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Represents the optometrist entity with fields for storing personal and professional information. Provides a data model for database persistence and business logic interactions.	**Collaborators:** - OptometristRepository: Maps this entity to the database and performs CRUD operations. - OptometristServiceImpl: Uses this model to implement business logic. - OptometristController: Passes optometrist data between the front-end and back-end.

Class Name: Patient	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Represents the patient entity with fields for storing personal and professional information. Provides a data model for database persistence and business logic interactions.	**Collaborators:** - PatientRepository: Maps this entity to the database and performs CRUD operations. - PatientServiceImpl: Uses this model to implement business logic. - PatientController: Passes patient data between the front-end and back-end.

Class Name: VisualAssessment	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Represent the data for a visual assessment in the system. Serve as the entity for database operations.	**Collaborators:** - AssessmentRepository: uses this entity for persistence. - AssessmentService: manages operations on this entity.

Class Name: UserEntity	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Represent a user in the system, including credentials and roles. Map to the users table in the database. Establish a many-to-many relationship with Role.	**Collaborators:** - Role: defines the roles assigned to the user. - UserRepository: manages persistence of this entity. - CustomUserDetailsService: retrieves user details for authentication.

Class Name: Role	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Represent a role (e.g., "admin", "optometrist", "patient") in the system. Map to the roles table in the database. Establish a many-to-many relationship with UserEntity.	**Collaborators:** - UserEntity: users associated with this role).

Security Configuration:

Class Name: CustomUserDetailsService	
Parent Class: UserDetailsService **Subclasses:** N/A	
Responsibilities: Load user details by email for authentication. Map user roles to Spring Security authorities.	**Collaborators:** - UserService: fetches user information from the database. - GrantedAuthority: maps user roles to security roles. - UsernameNotFoundException: thrown when the user is not found.

Class Name: SecurityConfig	
Parent Class: N/A **Subclasses:** N/A	
Responsibilities: Configure Spring Security for the application. Define security-related beans like PasswordEncoder and SecurityFilterChain. Handle security policies for HTTP requests, CSRF, form login, logout, and authorization rules.	**Collaborators:** - CustomUserDetailsService: used to authenticate and fetch user details. - HttpSecurity: used to define security configurations. - BCryptPasswordEncoder: (for encoding passwords).

Software Architecture Diagram