

Logotipo

Descripción generada automáticamente

**PRIMER PARCIAL**

**“Control de Asistencia Academica - Microservicio”**

**MATERIA** : INF552 - ARQUITECTURA DEL SOFTWARE

**GRUPO** : SA

**DOCENTE** :ING. JOSUÉ OBED VEIZAGA GONZALES

**NOMBRE** :

**REGISTRO** :

**FECHA** :28/09/2025

Gestión II - 2025

Santa Cruz De La Sierra – Bolivia

# Flujo de Requisitos.

## Captura de Requisitos.

## Requisitos Funcionales.

**CU1: Gestionar Usuarios del Sistema**

|  |  |
| --- | --- |
| **Campo** | **Descripción** |
| **Propósito** | Gestionar el ciclo de vida completo de los usuarios del sistema (Administradores, Profesores, Estudiantes) |
| **Descripción** | Permitirá al Administrador crear, consultar, actualizar y deshabilitar usuarios, y a todos los usuarios iniciar y cerrar sesión |
| **Actores** | Administrador (primario), Profesor, Estudiante |
| **Actor Iniciador** | Administrador (para CRUD), cualquier usuario (para login) |
| **Precondición** | El Administrador debe tener sesión activa para gestionar usuarios |
| **Proceso** | 1. **Login Usuario**     1.1. Validar credenciales     1.2. Autenticar usuario     1.3. Iniciar sesión 2. **Crear Usuario**     2.1. Ingresar datos usuario     2.2. Validar unicidad username     2.3. Hashear contraseña     2.4. Guardar usuario 3. **Consultar Usuarios**     3.1. Listar usuarios     3.2. Filtrar por rol     3.3. Ver detalles usuario 4. **Actualizar Usuario**     4.1. Editar información     4.2. Validar cambios     4.3. Guardar cambios 5. **Deshabilitar Usuario**     5.1. Seleccionar usuario     5.2. Confirmar deshabilitación     5.3. Marcar como inactivo 6. **Logout**     6.1. Cerrar sesión     6.2. Limpiar datos locales |
| **Post-condición** | Los usuarios quedan registrados en el sistema con sus roles asignados y pueden autenticarse según sus permisos |
| **Excepciones** | 1. Username ya existe 2. Campos obligatorios vacíos 3. Rol no válido 4. Credenciales incorrectas 5. Usuario inactivo |

CU2: Gestionar Estructura Académica

|  |  |
| --- | --- |
| **Campo** | **Descripción** |
| **Propósito** | Administrar la estructura académica completa: materias, asignación de profesores e inscripción de estudiantes |
| **Descripción** | Permitirá al Administrador gestionar el catálogo académico incluyendo creación de materias y asignación de relaciones profesor-materia y estudiante-materia |
| **Actores** | Administrador |
| **Actor Iniciador** | Administrador |
| **Precondición** | El Administrador debe tener sesión activa y existen usuarios de tipo Profesor y Estudiante creados |
| **Proceso** | 1. **Gestionar Materias**     1.1. Crear materia     1.2. Listar materias     1.3. Editar materia     1.4. Eliminar materia 2. **Asignar Profesores**     2.1. Seleccionar materia     2.2. Seleccionar profesor     2.3. Validar que profesor existe     2.4. Crear asignación 3. **Inscribir Estudiantes**     3.1. Seleccionar materia     3.2. Seleccionar estudiante     3.3. Validar que estudiante existe     3.4. Crear inscripción 4. **Consultar Relaciones**     4.1. Ver profesores por materia     4.2. Ver estudiantes por materia     4.3. Ver materias por profesor     4.4. Ver materias por estudiante |
| **Post-condición** | La estructura académica queda definida con materias creadas y las relaciones profesor-materia y estudiante-materia establecidas |
| **Excepciones** | 1. Código de materia duplicado 2. Profesor no existe 3. Estudiante no existe 4. Materia no existe 5. Asignación ya existe 6. Inscripción ya existe 7. Materia tiene relaciones activas |

CU3: Gestionar Sesiones de Asistencia

|  |  |
| --- | --- |
| **Campo** | **Descripción** |
| **Propósito** | Gestionar el proceso completo de toma de asistencia en sesiones de clase |
| **Descripción** | Permitirá al Profesor crear sesiones de asistencia, registrar la presencia/ausencia de estudiantes y cerrar las sesiones |
| **Actores** | Profesor |
| **Actor Iniciador** | Profesor |
| **Precondición** | El Profesor debe tener sesión activa y tener materias asignadas |
| **Proceso** | 1. **Ver Materias Asignadas**     1.1. Consultar materias del profesor     1.2. Listar materias en dashboard 2. **Iniciar Sesión de Asistencia**     2.1. Seleccionar materia     2.2. Validar asignación profesor-materia     2.3. Verificar que no existe sesión abierta     2.4. Crear sesión con fecha actual     2.5. Generar registros iniciales (ausentes) 3. **Tomar Asistencia**     3.1. Cargar lista de estudiantes inscritos     3.2. Mostrar interfaz de marcado     3.3. Marcar estudiante como Presente/Ausente     3.4. Actualizar estado en tiempo real 4. **Finalizar Sesión**     4.1. Confirmar cierre de sesión     4.2. Guardar registros permanentemente     4.3. Marcar sesión como cerrada 5. **Consultar Historial**     5.1. Ver sesiones anteriores     5.2. Consultar detalles de sesión |
| **Post-condición** | La asistencia queda registrada permanentemente en el sistema y disponible para consultas |
| **Excepciones** | 1. Profesor no asignado a la materia 2. Sesión ya existe para la fecha 3. No hay estudiantes inscritos 4. Sesión ya está cerrada 5. Error de conexión entre servicios |

CU4: Consultar Historial de Asistencia

|  |  |
| --- | --- |
| **Campo** | **Descripción** |
| **Propósito** | Consultar y visualizar el historial de asistencia de los estudiantes |
| **Descripción** | Permitirá al Estudiante consultar su propio historial de asistencia por materia y ver estadísticas de participación |
| **Actores** | Estudiante |
| **Actor Iniciador** | Estudiante |
| **Precondición** | El Estudiante debe tener sesión activa y estar inscrito en al menos una materia |
| **Proceso** | 1. **Ver Materias Inscritas**     1.1. Consultar materias del estudiante     1.2. Listar materias en dashboard 2. **Consultar Asistencia por Materia**     2.1. Seleccionar materia     2.2. Validar inscripción estudiante-materia     2.3. Obtener registros de asistencia     2.4. Calcular estadísticas 3. **Visualizar Historial Detallado**     3.1. Mostrar fechas de sesiones     3.2. Mostrar estado (Presente/Ausente)     3.3. Mostrar porcentaje de asistencia 4. **Consultar Estadísticas Generales**     4.1. Calcular porcentajes por materia     4.2. Mostrar resumen general     4.3. Visualizar tendencias |
| **Post-condición** | El estudiante puede visualizar su historial completo de asistencia con estadísticas detalladas por materia |
| **Excepciones** | 1. Estudiante no inscrito en la materia 2. No hay registros de asistencia 3. Materia no existe 4. Error al calcular estadísticas |

CU5: Administrar Sistema Integral

|  |  |
| --- | --- |
| **Campo** | **Descripción** |
| **Propósito** | Administrar y supervisar el funcionamiento integral del sistema de control de asistencia |
| **Descripción** | Permitirá al Administrador tener una visión global del sistema y realizar operaciones de mantenimiento y supervisión |
| **Actores** | Administrador |
| **Actor Iniciador** | Administrador |
| **Precondición** | El Administrador debe tener sesión activa |
| **Proceso** | 1. **Ver Dashboard de Resumen**     1.1. Cargar estadísticas generales     1.2. Mostrar métricas del sistema 2. **Gestionar Usuarios** (incluye CU1)     2.1. CRUD completo de usuarios     2.2. Gestionar roles y permisos 3. **Gestionar Estructura Académica** (incluye CU2)     3.1. CRUD de materias     3.2. Gestionar asignaciones     3.3. Gestionar inscripciones 4. **Supervisar Operaciones**     4.1. Monitorear sesiones de asistencia     4.2. Ver reportes de actividad     4.3. Gestionar excepciones del sistema 5. **Generar Reportes**     5.1. Exportar datos de asistencia     5.2. Generar estadísticas institucionales |
| **Post-condición** | El sistema queda configurado y operativo según los requerimientos institucionales |
| **Excepciones** | 1. Datos inconsistentes entre servicios 2. Errores de integración 3. Problemas de rendimiento 4. Datos corruptos |

## Requerimientos No Funcionales (RNF)

* RNF-01: Arquitectura de Microservicios: El sistema debe estar compuesto por servicios independientes y desacoplados.
* RNF-02: Lenguaje y Base de Datos: Implementación en JavaScript (Node.js nativo) y MySQL.
* RNF-03: Comunicación: Los servicios se comunicarán mediante APIs RESTful sobre HTTP/JSON.
* RNF-04: Simplicidad: No se utilizarán frameworks externos (Express, Hapi, etc.) para la capa web de los servicios. Se usarán módulos nativos de Node.js (http, url).
* RNF-05: Seguridad Básica: Las contraseñas se almacenarán encriptadas (hash con bcrypt). Las APIs serán stateless.
* RNF-06: Base de Datos por Servicio: Cada microservicio gestionará su propia base de datos o esquema dentro del mismo servidor MySQL.

# Análisis.

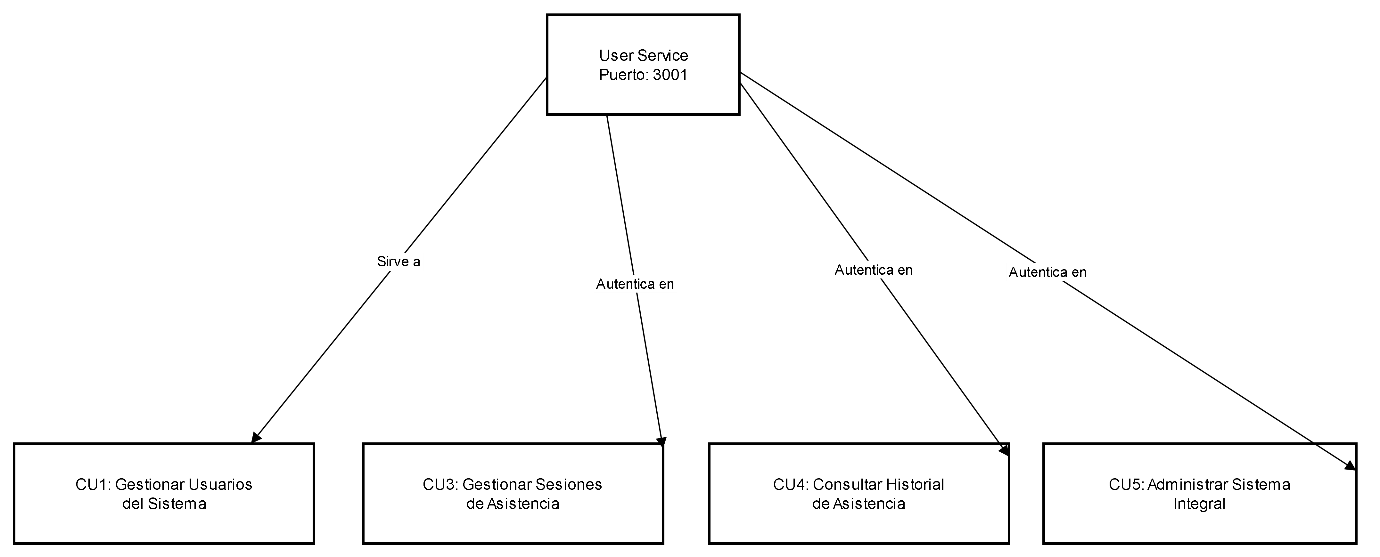
## Identificación de servicios

### Servicio User Service

(Puerto 3001):

Responsabilidad: Gestionar todo lo relacionado con los usuarios (Admin, Profesor, Estudiante).

Base de Datos: Esquema users\_db. Tablas: users (con un campo role).

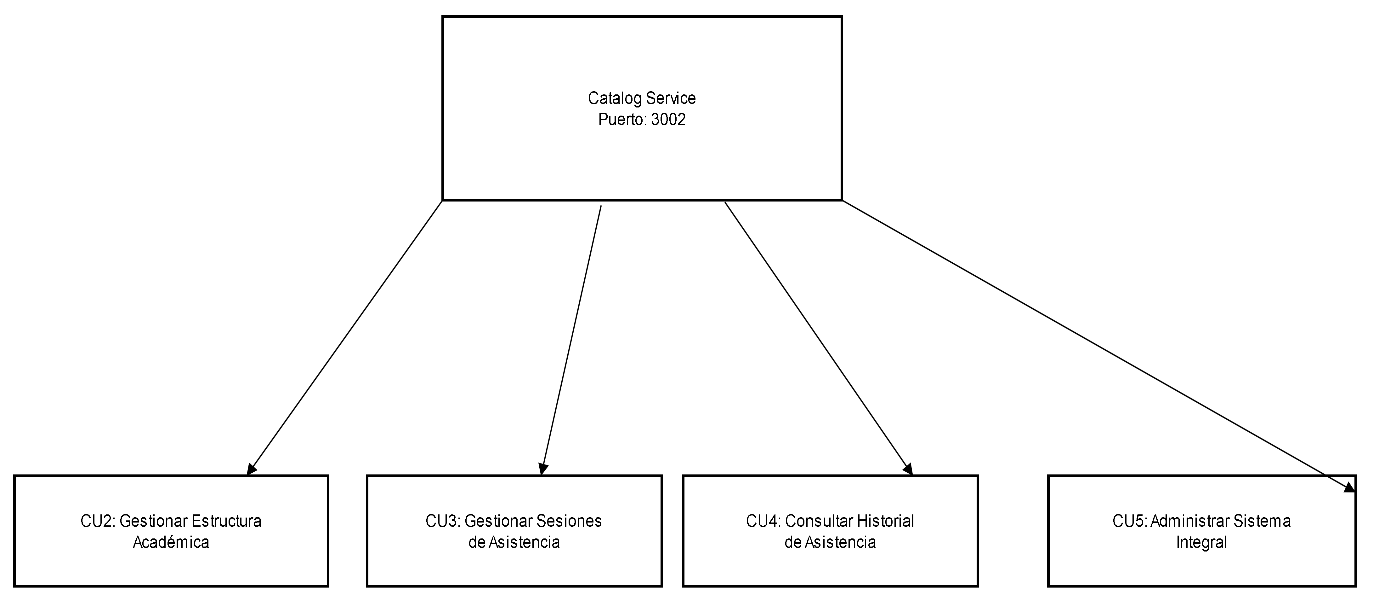


### Catalog Service

(Puerto 3002):

Responsabilidad: Gestionar las entidades académicas (Materias) y sus relaciones (asignación de profesores, inscripción de estudiantes).

Base de Datos: Esquema catalog\_db. Tablas: subjects, subject\_teachers, subject\_students.



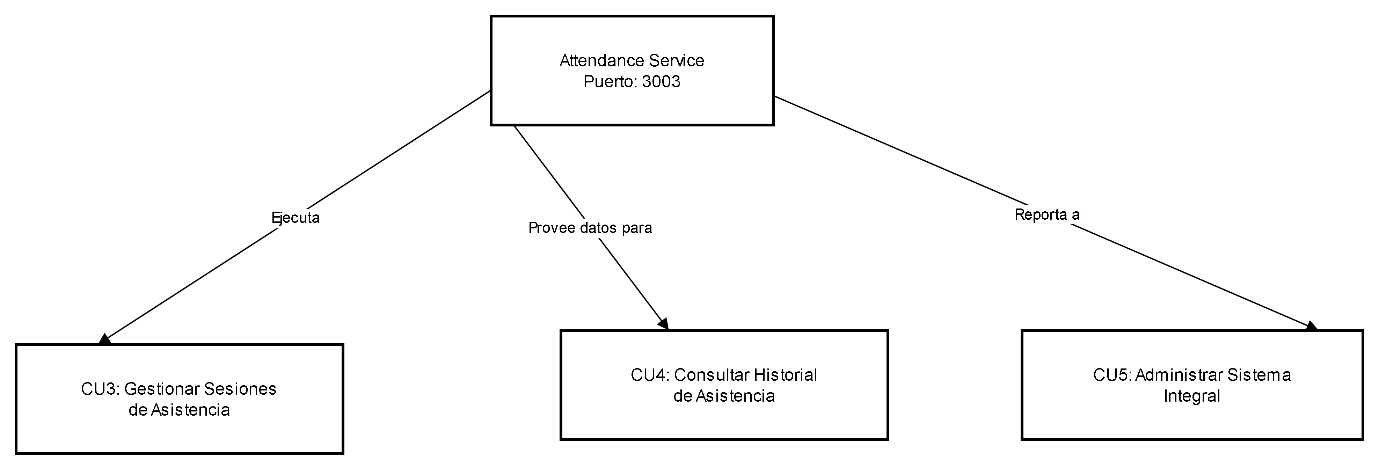
### Attendance Service

(Puerto 3003):

Responsabilidad: Gestionar el proceso completo de asistencia (sesiones, registros).

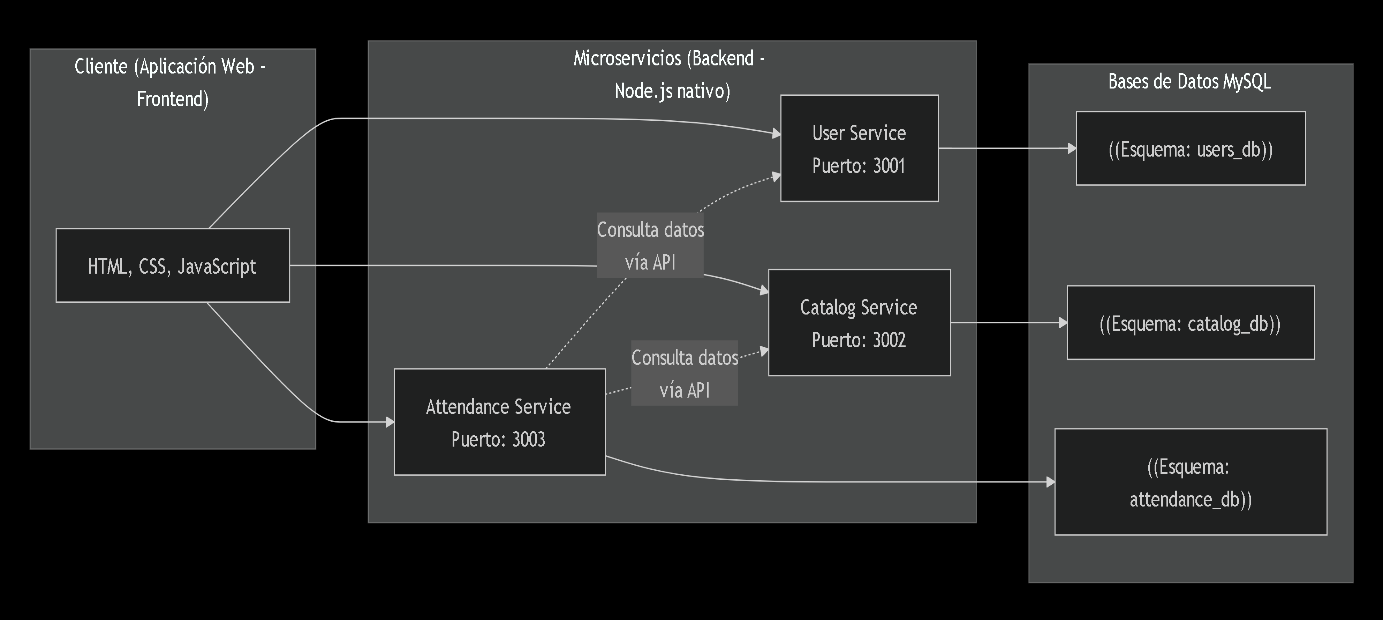
Base de Datos: Esquema attendance\_db. Tablas: attendance\_sessions, attendance\_records.

Dependencias: Para crear una sesión, necesita validar que el profesor esté asignado a la materia (llamada al Catalog Service). Para mostrar la lista de estudiantes, necesita obtenerlos del Catalog Service.



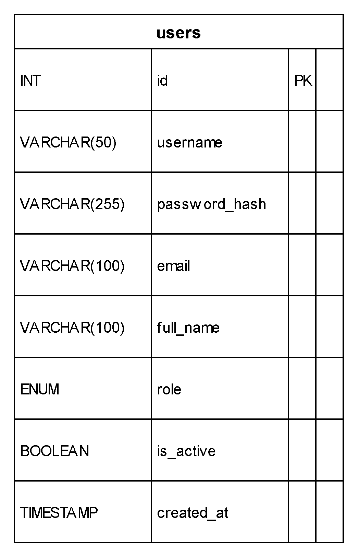
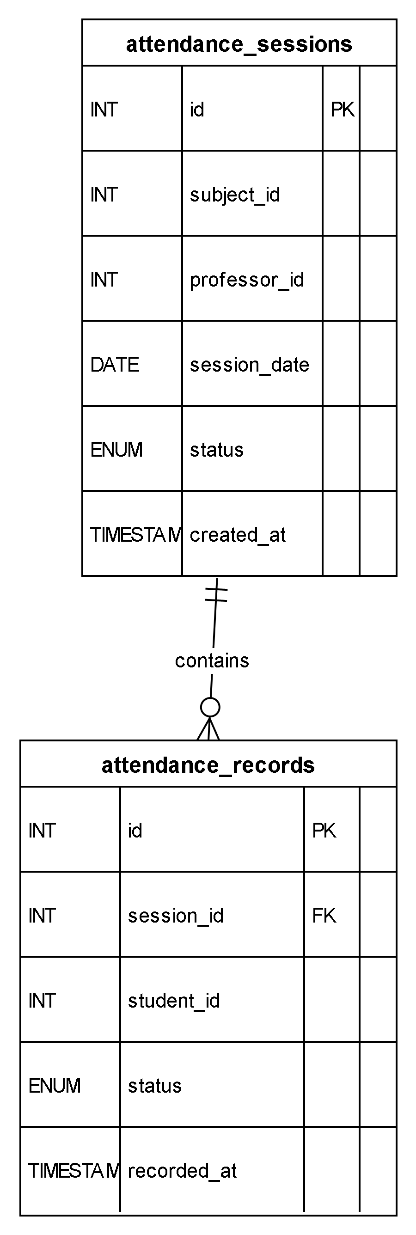
# Flujo: Diseño.

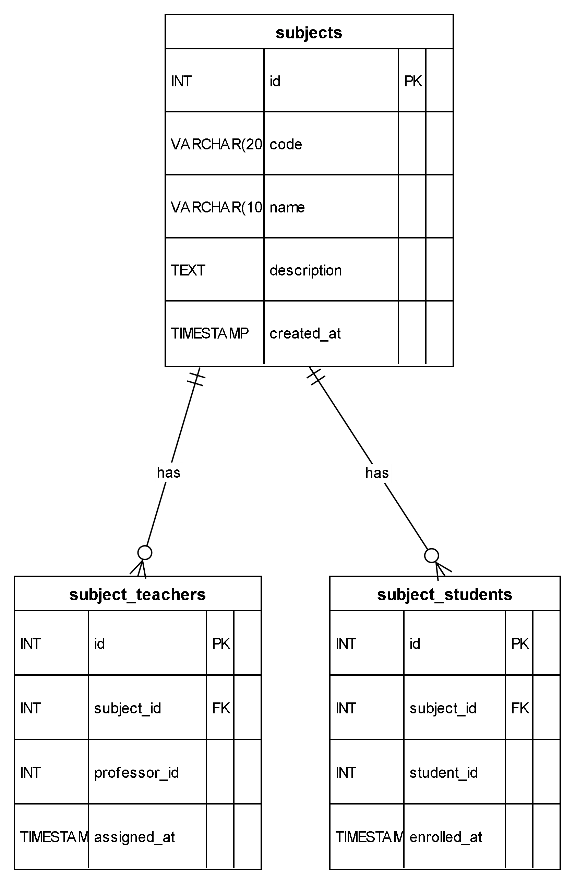
## Diseño de Arquitectura.



## Diseño de Base de Datos.

### Diseño Conceptual.





### Diseño Físico.

CREATE DATABASE IF NOT EXISTS attendance\_db;

USE attendance\_db;

-- Tabla de sesiones de asistencia

CREATE TABLE IF NOT EXISTS attendance\_sessions (

id INT AUTO\_INCREMENT PRIMARY KEY,

subject\_id INT NOT NULL,

professor\_id INT NOT NULL,

session\_date DATE NOT NULL,

status ENUM('open', 'closed') DEFAULT 'open',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE KEY unique\_session (subject\_id, session\_date)

);

-- Tabla de registros de asistencia individuales

CREATE TABLE IF NOT EXISTS attendance\_records (

id INT AUTO\_INCREMENT PRIMARY KEY,

session\_id INT NOT NULL,

student\_id INT NOT NULL,

status ENUM('present', 'absent') NOT NULL,

recorded\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (session\_id) REFERENCES attendance\_sessions(id) ON DELETE CASCADE,

UNIQUE KEY unique\_attendance (session\_id, student\_id)

);

CREATE DATABASE IF NOT EXISTS catalog\_db;

USE catalog\_db;

-- Tabla de materias

CREATE TABLE IF NOT EXISTS subjects (

id INT AUTO\_INCREMENT PRIMARY KEY,

code VARCHAR(20) UNIQUE NOT NULL,

name VARCHAR(100) NOT NULL,

description TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Tabla de relación materias-profesores

CREATE TABLE IF NOT EXISTS subject\_teachers (

id INT AUTO\_INCREMENT PRIMARY KEY,

subject\_id INT NOT NULL,

professor\_id INT NOT NULL,

assigned\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (subject\_id) REFERENCES subjects(id) ON DELETE CASCADE,

UNIQUE KEY unique\_subject\_teacher (subject\_id, professor\_id)

);

-- Tabla de relación materias-estudiantes (inscripciones)

CREATE TABLE IF NOT EXISTS subject\_students (

id INT AUTO\_INCREMENT PRIMARY KEY,

subject\_id INT NOT NULL,

student\_id INT NOT NULL,

enrolled\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (subject\_id) REFERENCES subjects(id) ON DELETE CASCADE,

UNIQUE KEY unique\_subject\_student (subject\_id, student\_id)

);

CREATE DATABASE IF NOT EXISTS users\_db;

USE users\_db;

CREATE TABLE IF NOT EXISTS users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

password\_hash VARCHAR(255) NOT NULL,

email VARCHAR(100),

full\_name VARCHAR(100),

role ENUM('admin', 'professor', 'student') NOT NULL,

is\_active BOOLEAN DEFAULT TRUE,

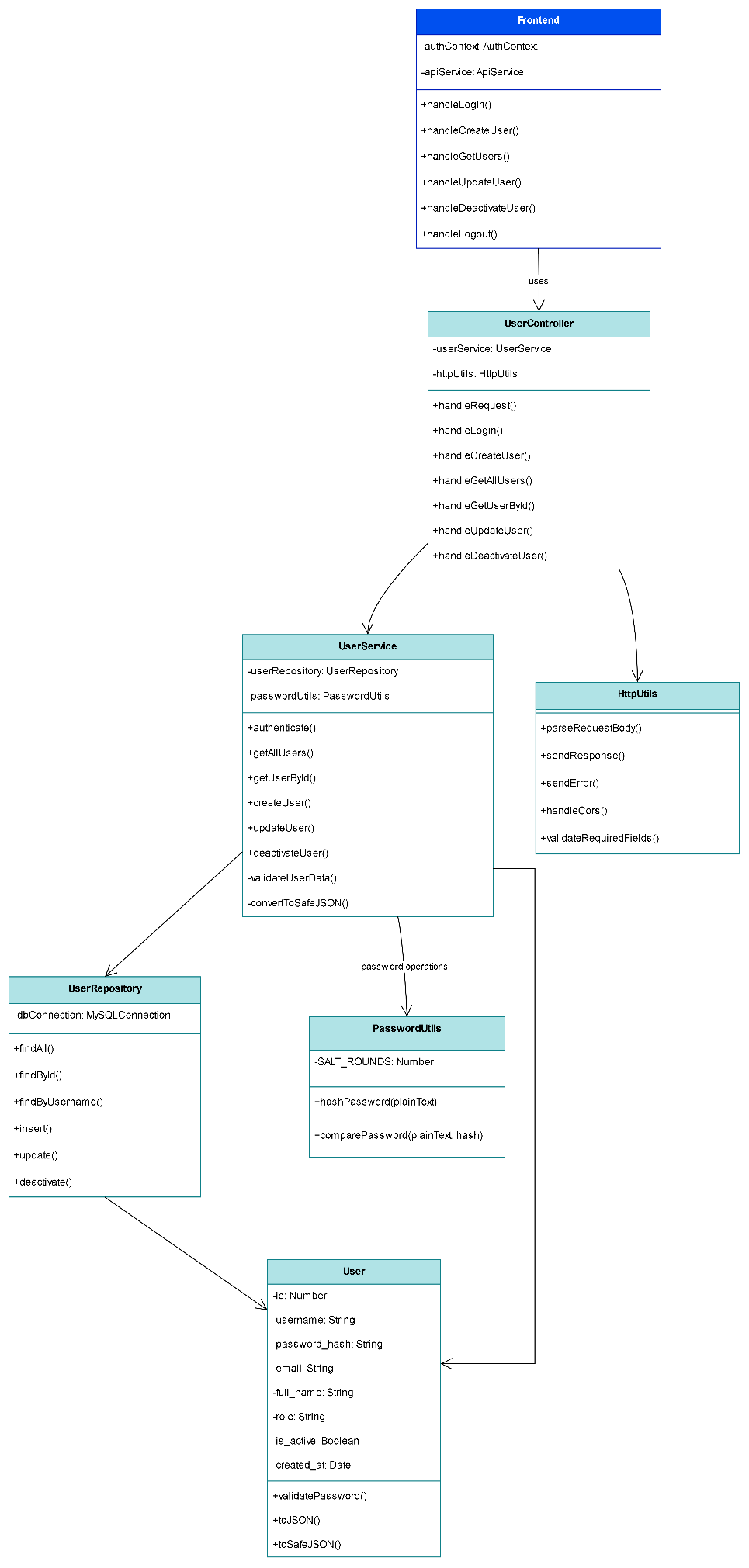
created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

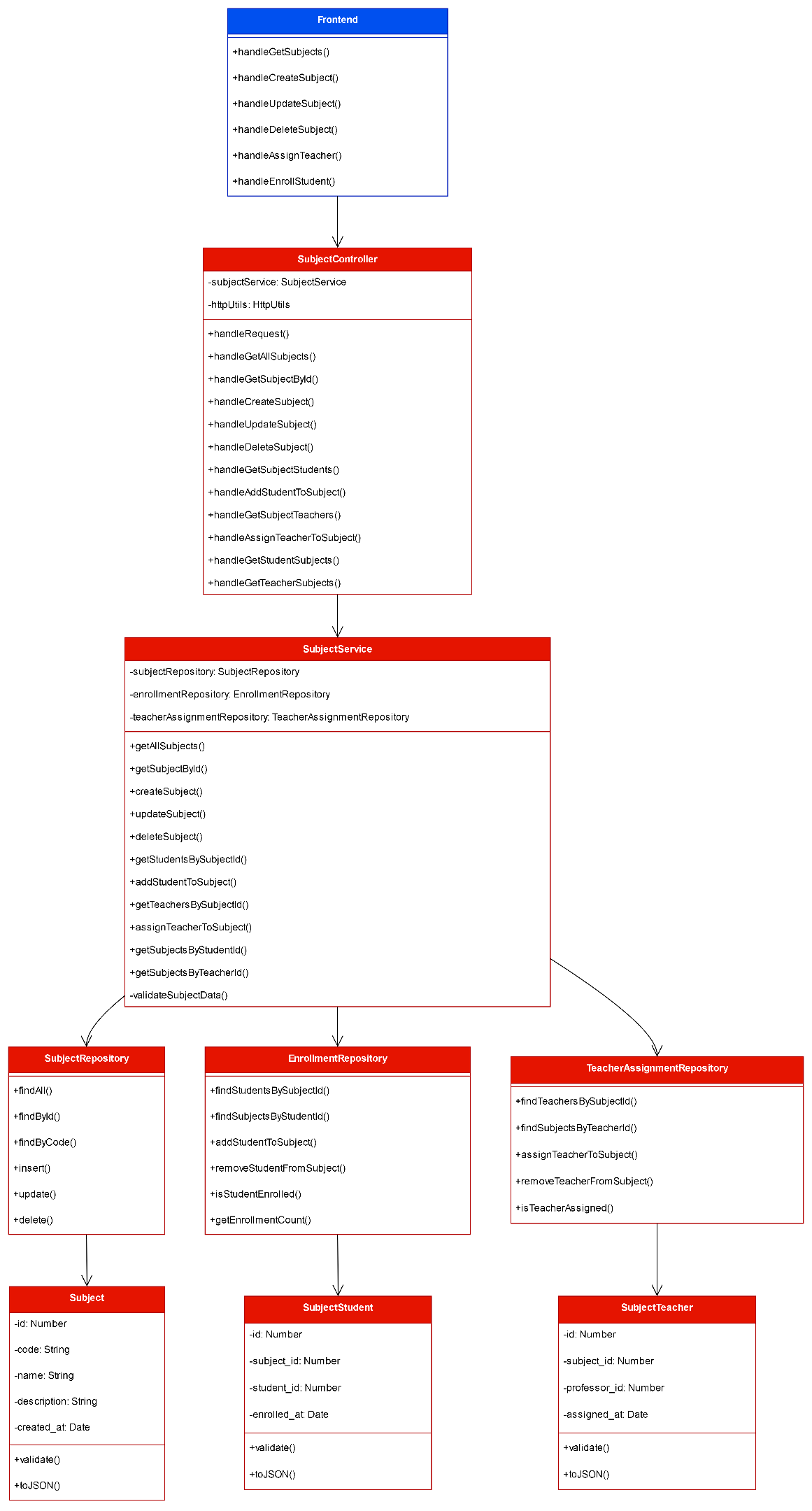
);

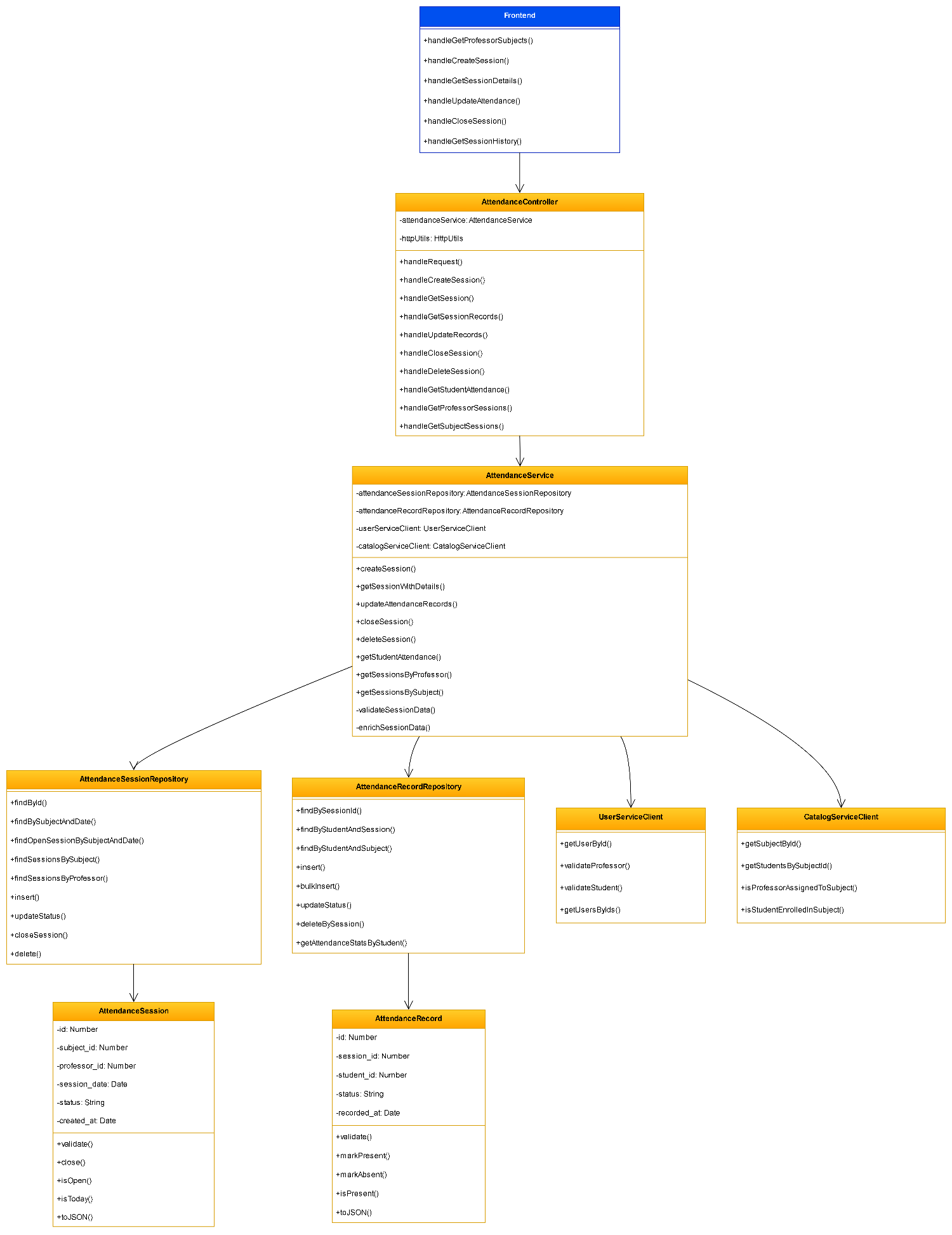
## Diseño de Detalle Procedimental.

### Diseño de Clase Dinámico.

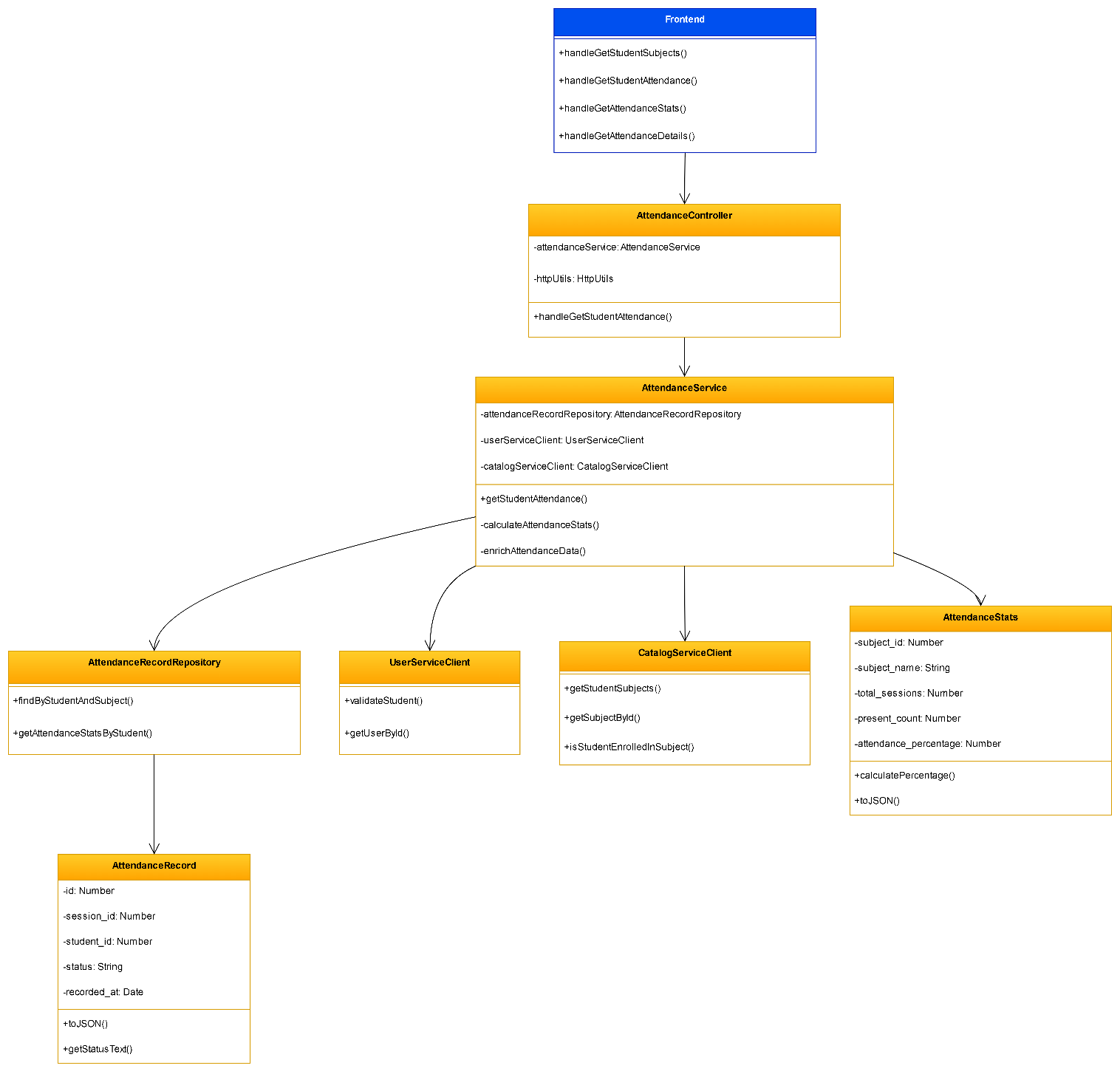
CU: Gestionar Usuarios del Sistema

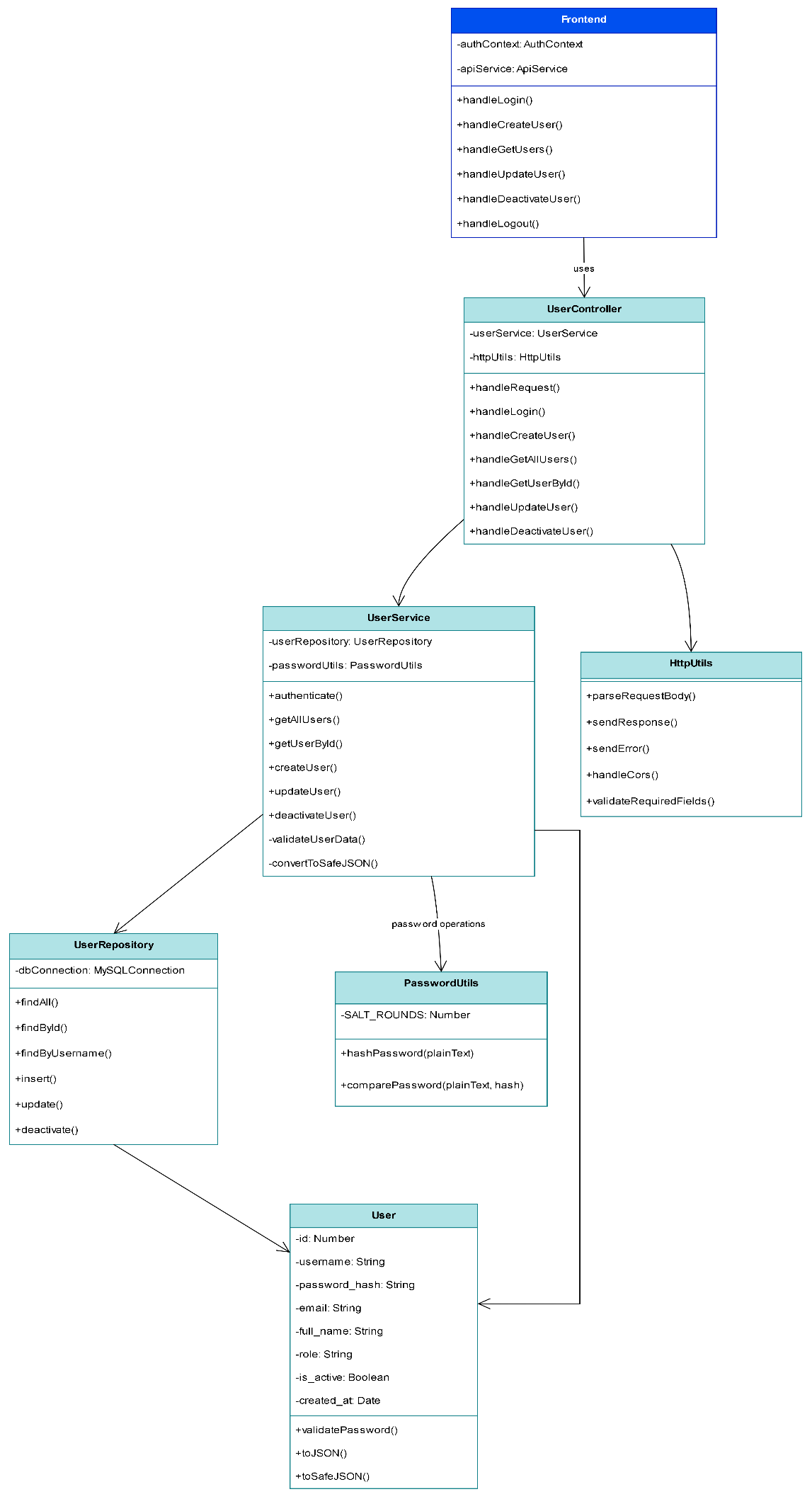


CU: Gestionar Estructura Académica

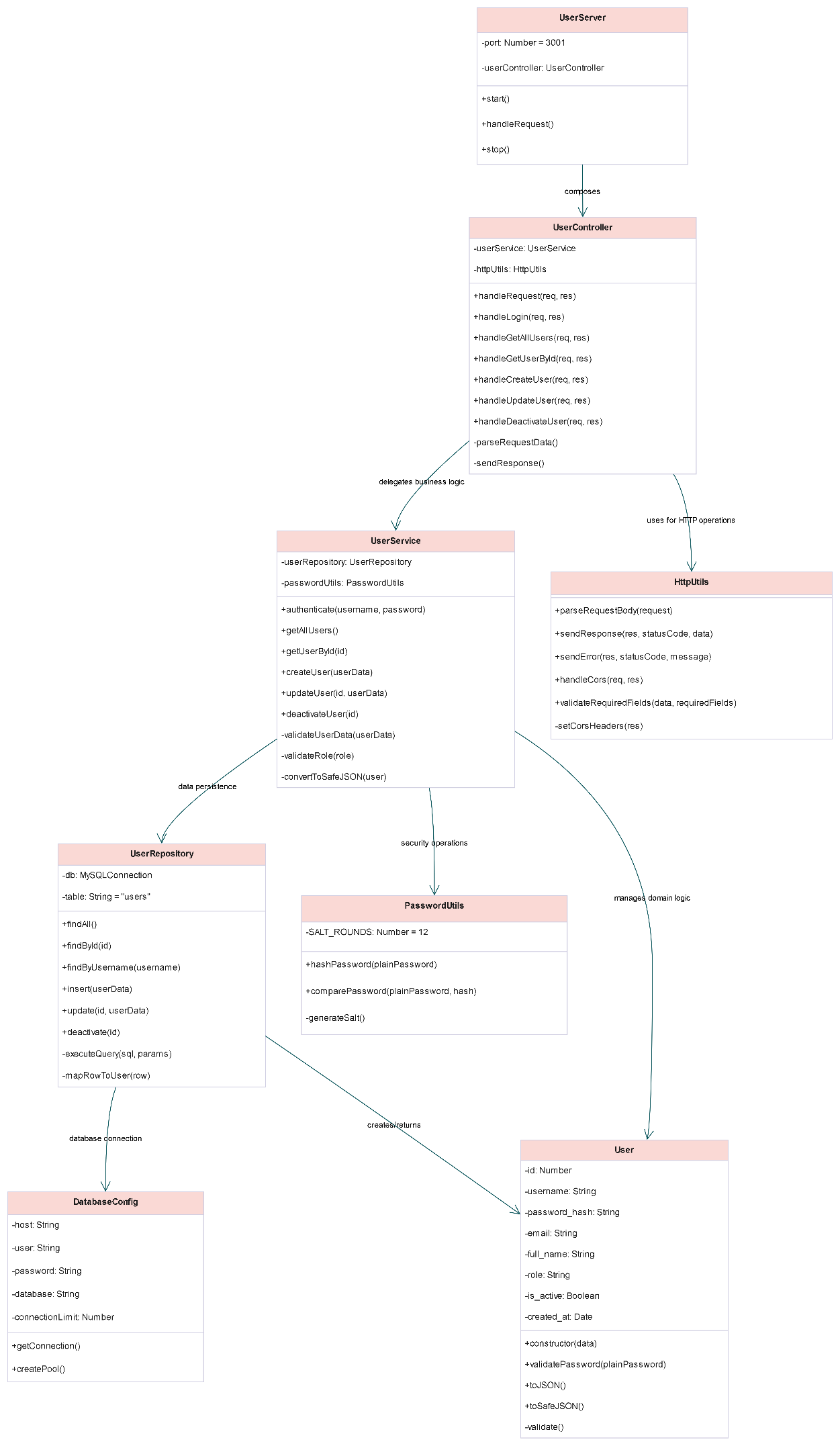
CU: Gestionar Sesiones de Asistencia

CU: Consultar Historial de Asistencia

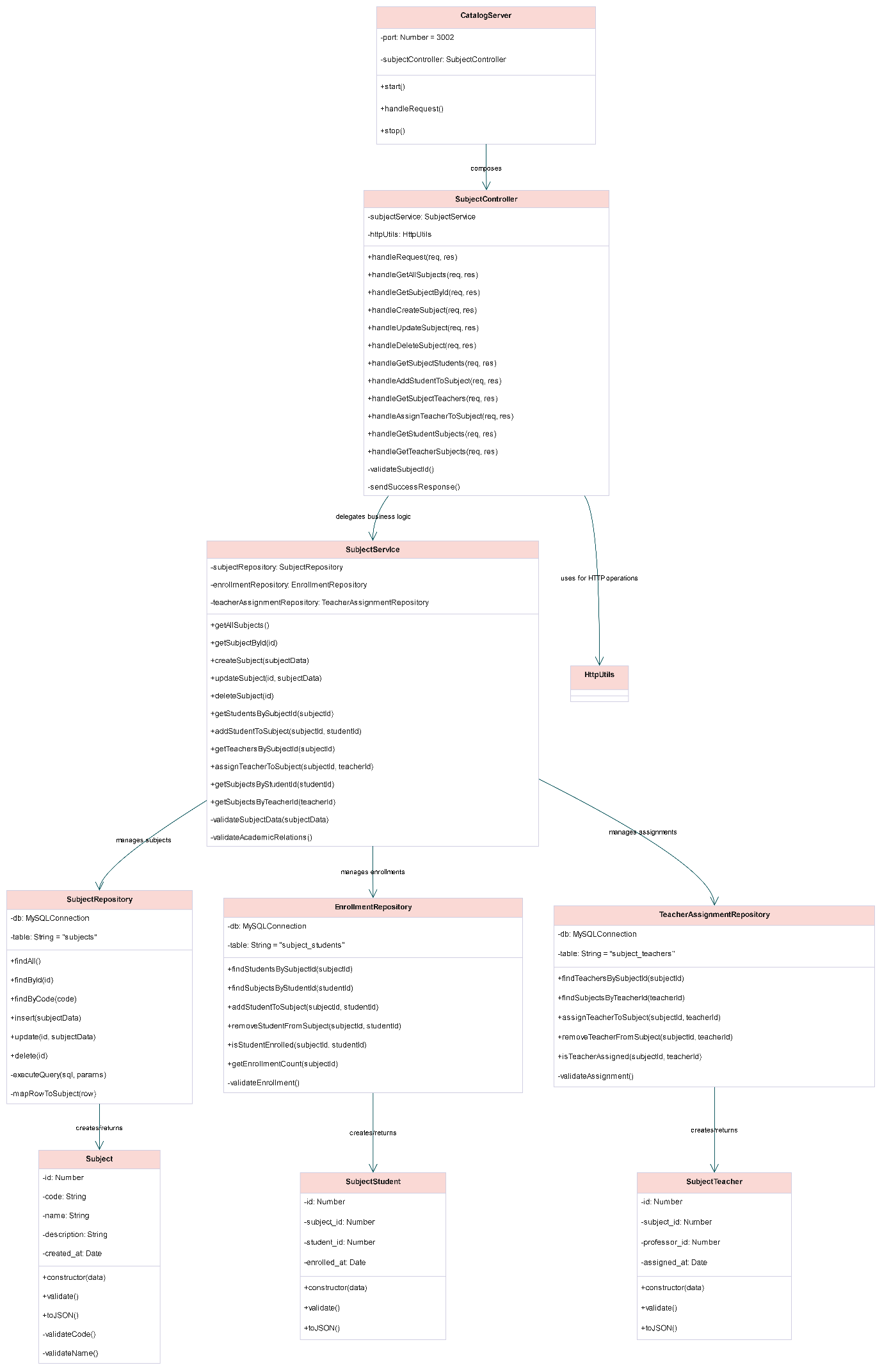


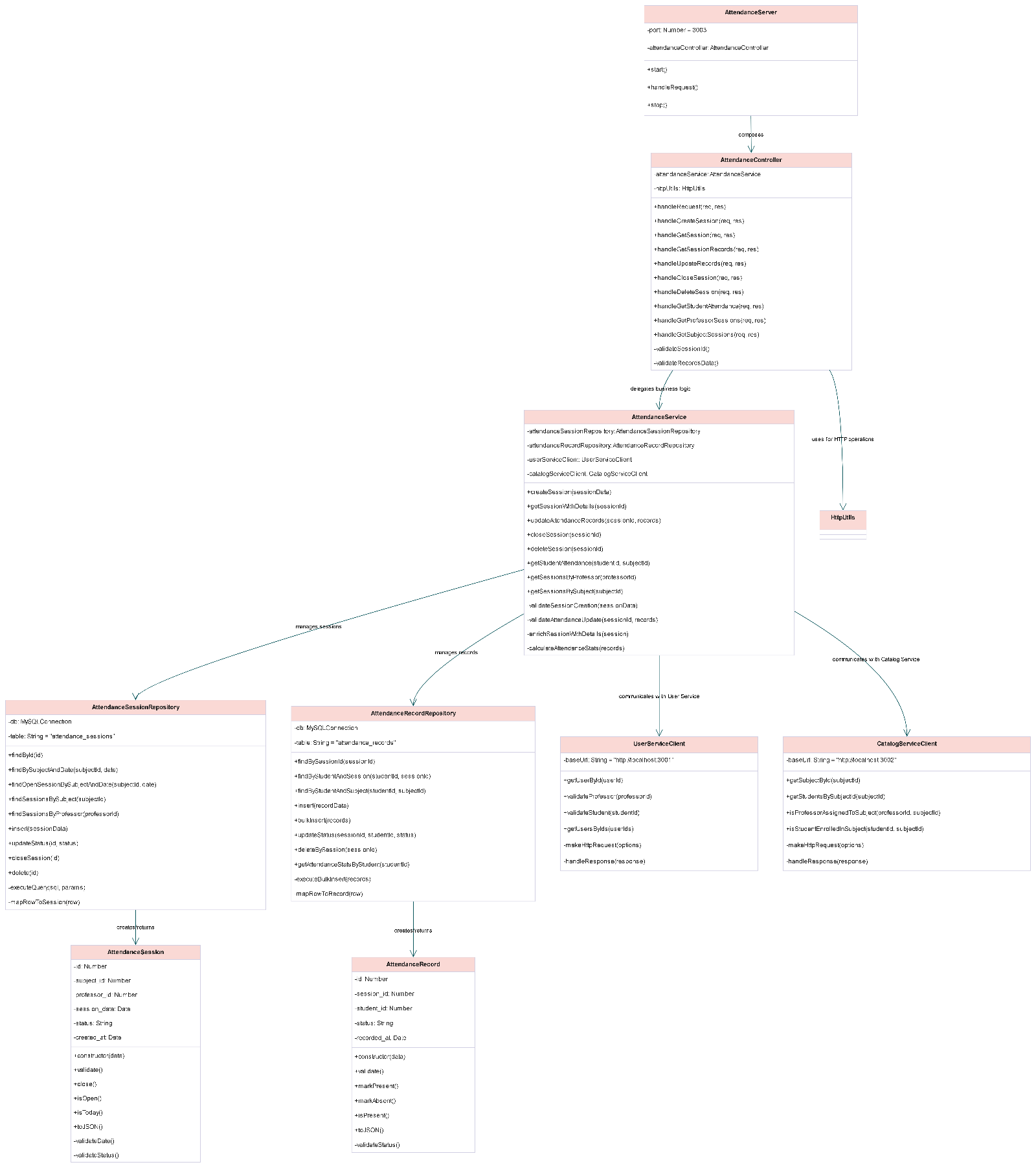
CU5: Administrar Sistema Integral

## Diseño de clase por servicio

USER SERVICE

CATALOG SERVICE



Attendance Service

## Diseño de interfaz

