Universidad ORT Uruguay
Facultad de Ingeniería

Obligatorio Arquitectura de Software

Entregado como requisito para la obtención del título de

Ingeniero en Sistemas

Link al repo:
https://github.com/IngSoft-AR-2023-2/230263_251105_254813.git

Milena dos Santos – 254813

Guzmán Dupont – 230263

Julieta Sarantes – 251105

Tutores: Ivan Etchart, Damian Moretti, Marco Fiorito

2023

Funcionalidades implementadas

Las funcionalidades que llegamos a implementar son las siguientes:

- Requerimiento 1: Usuario y Administradores - Alta y Modificación de Usuarios, en este requerimiento debemos pasarle por body en el form-data, los atributos: firstName, lastName, documentID, password, confirmPassword, dateOfBirth, address, telephone, email, picture (menor a 1MB) y role el cual puede tomar los valores “admin”, “u”, “p”, “a” los que significan administrador, usuario, profesional médico y administrativo respectivamente. En este punto también se envía un mail de verificación el cual al clickear verifica el mismo.

- Requerimiento 2: Sign In de Usuarios, permite la autenticación del usuario.

- Requerimiento 3: Legajo de vacunación, en este punto se provee un historial el cual contiene el nombre único de la vacuna, el estado de la misma, en este caso solo se maneja (Pendiente y Administrada), la fecha para la cual está agendada, así como el documento del administrador y la hora en la que esta fue administrada. Estos dos últimos permanecen en null hasta que la vacuna fue administrada exitosamente.

  A su vez, se creó otro endpoint el cual permite agregar vacunas a la tabla maestra (en caso de que no existan)

- Requerimiento 4: Administrar vacuna, este requerimiento solo está disponible para que lo realice un administrador, el cual se encarga de proveer el código de la vacuna y el documento. En este punto se valida que el administrador no se pueda auto vacunar, ya que no tiene sentido en el mundo real. Se cumplen todas las validaciones excepto que sea a la misma hora, si en la misma fecha. Consideramos que es un aspecto a mejorar en futuros proyectos.

- Requerimiento 5: Administrador - CRUD de funcionario, en este requerimiento tenemos tres de las funcionalidades pedidas, tenemos Create, Read y Delete, queda faltando Update.

- Requerimiento 6: Usuario - Ver disponibilidad para citas, en este punto tenemos un rango de horas posibles que es de 09:00 a 18:00 en ese rango tenemos horarios disponibles cada media hora (por ejemplo: 09:00, 09:30, 10:00, 10:30 y así sucesivamente), luego de ese rango de horas posibles dentro del rango de fechas que recibimos lo que hacemos es fijarnos en las citas de ese médico(que también lo recibimos) si tiene ocupado una cierta hora un cierto día lo sacamos de las fechas disponibles para citas

- Requerimiento 7: Usuario- Fijar Cita, aquí es donde registramos citas dejando agendarse cualquier día pero en el rango de horas posibles que son las mismas usadas en el requerimiento anterior.

- Requerimiento 8: Administrativo - Ver Citas, se listan todas las citas pendientes, mostrando de cada una la fecha, la hora y el Identificador del profesional.

- Requerimiento 9: Profesional - Ver Citas, se listan las citas pendientes de un determinado profesional en una determinada fecha, mostrando de cada cita el documento del profesional, la fecha, la hora y el documento del usuario.

- Requerimiento 10: Profesional - Agregar notas de citas, aquí el sistema recibe un usuario, una fecha y una nota. Le agrega dicha nota a la cita de ese usuario en esa fecha, en caso de no tener cita en esa fecha queda sin efecto.

- Requerimiento 11: Usuario - Consulta de todas las citas para un profesional, aquí lo que se hace es contar las citas de un determinado usuario para cada profesional. No llegamos a comprender en su totalidad el segundo punto de este requerimiento por lo cual no pudimos llegar a completarlo.

- Requerimiento 12: Usuario o Profesional - Consultar citas de un paciente, en este requerimiento lo que hacemos es separar si el que realiza la consulta(por el token lo sabemos) es un usuario o un profesional y dependiendo cual sea devolvemos o solo las citas de ese usuario en caso que el que consulta sea un usuario o todas las citas que tiene ese profesional en el rango determinado.

- Requerimiento 13: Usuario - Permitir agendarse para vacunación, en este punto se le permite al usuario agendarse para determinada vacuna, el hecho de que *cada vacuna dispondrá de un punto diferente para realizar la agenda,* lo tomamos en cuenta recibiendo el nameCode de la vacuna por query, generando que para cada vacuna sea un endpoint diferente.

  Se notifica al reservante la fecha y dirección del vacunatorio vía mail.

- Requerimiento 16 - Envío de certificado de vacunación, en este requerimiento al administrarse una vacuna se genera un código único y se guarda en la base de datos certificados. El administrador es el encargado de solicitar por medio de la plataforma a la unidad reguladora el certificado utilizando axios.

  Se envía un QR generado con los datos del usuario.

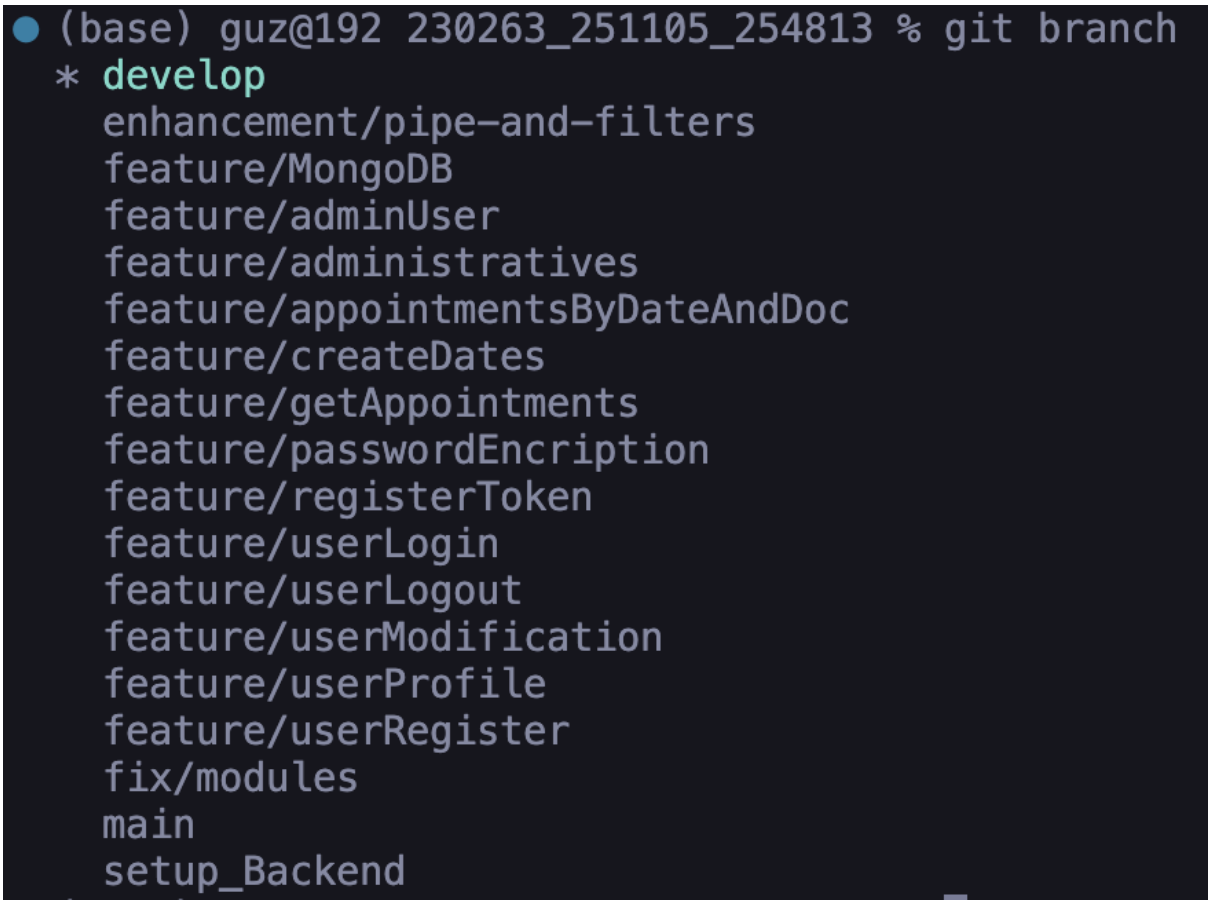
Requerimientos no implementados

Los requerimientos que no llegamos a implementar son los siguiente:

- Requerimiento 11: No implementamos uno de los puntos requeridos de la característica 11 porque estaba terriblemente mal explicado y dificultó enormemente la comprensión del mismo.
- Requerimiento 14 y 15 : No llegamos a implementar estos requerimientos por falta de tiempo.

Evidencia del uso de GitFlow

Utilizamos GitFlow como herramienta para el manejo de branches. Cada branch creada tenía como nombre el propósito (ej: *feature*) de la rama y el nombre de la característica a codificar en la branch.



Tácticas y patrones

Seguridad

Autenticar autores: por el uso de esta táctica nos aseguramos de que los usuarios que utilicen la plataforma sean quienes dicen ser al momento del registro en la aplicación.

Identidad Federada: mediante el uso de tokens los usuarios acceden a ciertas rutas protegidas con los permisos necesarios.

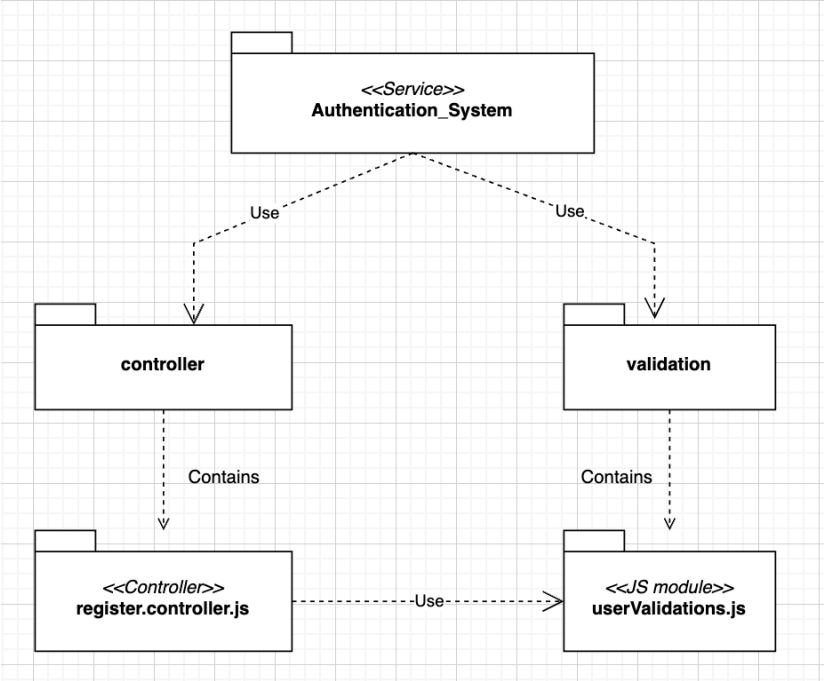
Modificabilidad

Dividir en módulos: mediante la división en módulos de nuestra aplicación permitimos que las modificaciones tengan un menor impacto.

Redistribuir responsabilidades: mediante la reorganización de las responsabilidades de distintos módulos mejoramos la cohesión y reducimos la dispersión de tareas similares en el sistema. Responsabilidades similares están en los mismos módulos.

Vistas

Vista de módulos - Separación de validaciones



Catálogo de elementos

Elemento	Descripción
Authentication_System	Servicio de terceros de autenticación.
controller	Carpeta que contiene los controladores utilizados por el servicio de autenticación.
register.controller.js	Controlador encargado del manejo de la creación de usuarios.
validation	Carpeta que contiene el módulo de validaciones del usuario a registrar.
userValidations.js	Módulo de javascript encargado de las validaciones del usuario a registrar.

ADR

El ADR correspondiente a esta vista es: 001 - Separación de validaciones al crear un usuario.

001 - Separación de validaciones al crear un usuario.

Estado: aprobado

Contexto:
Se debe considerar que las validaciones puedan cambiar o ampliarse, por lo que debe tenerse en cuenta que al introducir estas modificaciones generen el mínimo impacto posible.

Decisión:
Decidimos mantener todos los módulos de las validaciones que se necesitan al registrar el usuario en una carpeta a parte del controlador de crear usuario ya que esto nos permite cambiar o ampliar las funciones utilizadas para la validación sin tener que cambiar directamente el controlador.

Consecuencias
Positivas:

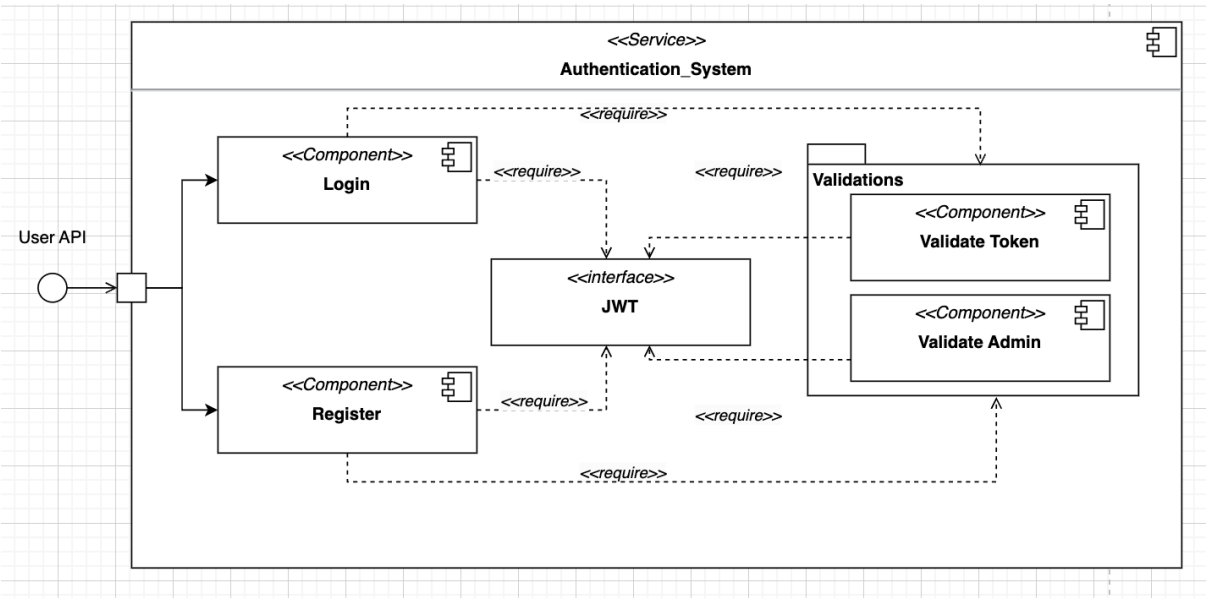
- Simplifica la mantenibilidad ya que se pueden realizar cambios sin afectar el código del controlador.
- Permite aumentar el número de validaciones de manera sencilla y clara.
- Proporciona una estructura más clara y fácil de entender, sin necesidad de tener que navegar a través de un archivo de controlador grande y complejo.

Negativas:

- Puede generar complejidad adicional al desarrollo si hay muchos módulos pequeños y se debe navegar entre ellos.

Mayor curva de aprendizaje para nuevos desarrolladores que deben entender el código.

Vista de componentes y conectores - Implementación de Identidad Federada.



Catálogo de elementos

Elementos	Descripción
User API	API utilizada por el usuario para la comunicación con el servicio Authentication_System.
Authentication_System	Servicio de terceros de autenticación.
Register	Componente encargado del registro del usuario.
Login	Componente encargado del login del usuario.
JWT	Componente que brinda y valida los tokens de usuarios.
Validate Token	Componente encargado de la validación del token al momento de hacer login.
Validate Admin	Componente encargado de validar que la ruta está siendo accedida por un Administrador.

ADR

El ADR correspondiente a esta vista es: 002 - Implementación de Identidad Federada.

002 - Implementación de Identidad Federada.

Estado: aprobado

Contexto:
Con el fin de proveer protección a determinados endpoints es necesario implementar un mecanismo que proteja las rutas y las haga accesibles a los usuarios autorizados.

Decisión:
Decidimos utilizar el patrón Identidad Federada con el fin de brindar un token y autenticar a los usuarios que utilicen la plataforma así como controlar el uso de ciertas rutas por parte de usuarios permitidos para brindar mayor seguridad.

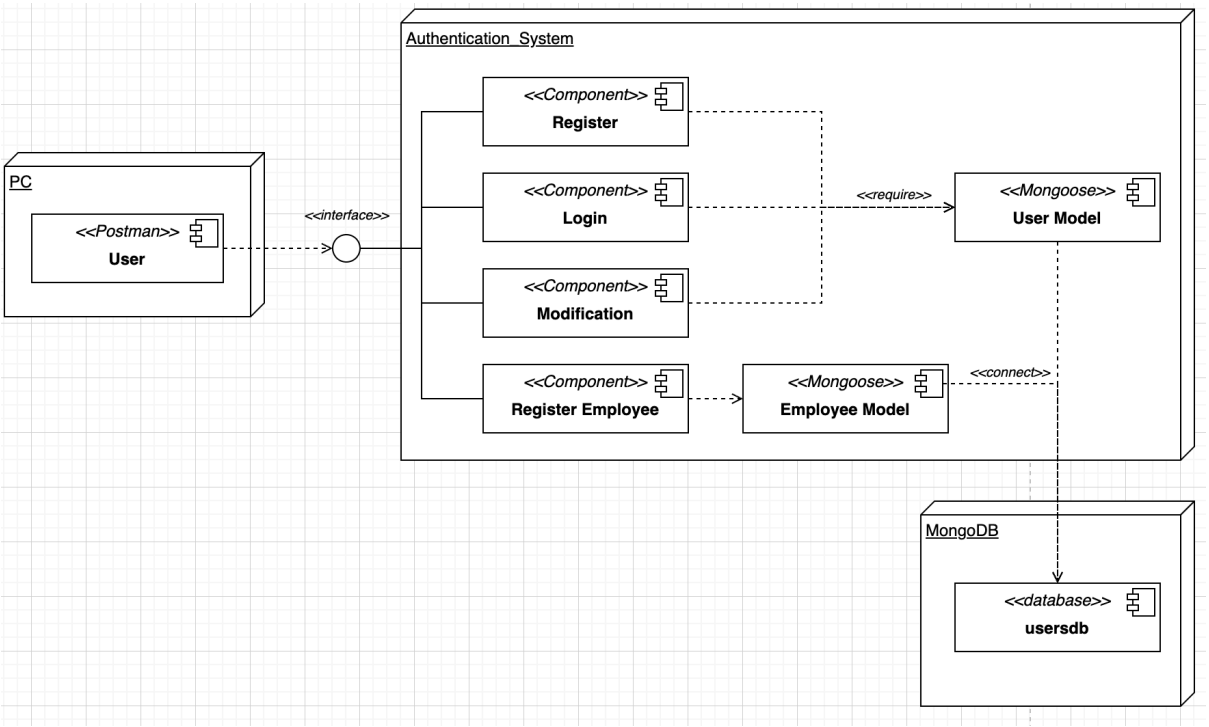
Consecuencias
Positivas:

- Brinda mayor seguridad al momento de utilizar la plataforma cuidando la PII (información de identificación personal) de los usuarios al saber que una persona autorizada maneja la información.
- Los usuarios pueden acceder a múltiples servicios con la misma credencial mejorando la experiencia de usuario.

Negativas:

- Tiene una dificultad de implementación extra. Además requiere una cuidadosa configuración dentro de los sistemas que la implementan.

Vista de asignación - Conexión con MongoDB para el sistema de autenticación.



Catálogo de elementos

Elementos	Descripción
User	Usuario utilizando Postman.
Register	Componente encargado del registro de un usuario.
Login	Componente encargado del login del usuario a la plataforma.
Modification	Componente encargado de la modificación del usuario.
Register Employee	Componente encargado del registro de un empleado.
User Model	Esquema de usuarios de mongoose.
Employee Model	Esquema de funcionarios de mongoose.
usersdb	Base de datos de MongoDB

ADR

El ADR correspondiente a esta vista es: 003 - Uso de MongoDB como sistema de base de datos para el sistema de autenticación.

003 - Uso de MongoDB como sistema de base de datos para el sistema de autenticación.

Estado: aprobado

Contexto:
Se requiere crear un sistema de autenticación que controle el acceso de usuarios y empleados a la plataforma y a la unidad reguladora.

Decisión:
Decidimos utilizar la base de datos MongoDB ya que nos permite el registro, acceso y búsqueda de usuarios y empleados de manera rápida y ágil. Además, permite el manejo de grandes volúmenes de datos siendo a la vez flexible con las estructuras de datos favoreciendo la modificabilidad de los mismos.

Consecuencias
Positivas:

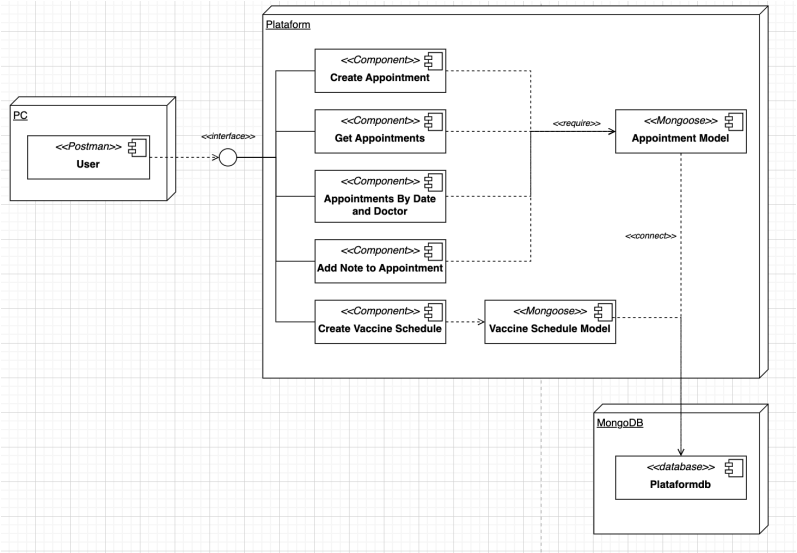
- Sencillo de implementar y manejar. Gracias a la aplicación MongoDB Compass, los datos se pueden ver y comprender sin complicaciones.
- Maneja una sintaxis muy parecida a JSON, lo que lo hace simple de modificar y crear datos nuevos o existentes.
- Hay mucha documentación disponible en internet lo que permite desarrollar el proyecto y mantenerse actualizado a las tecnologías fácilmente.
- MongoDB maneja y organiza altos volúmenes de datos favoreciendo la escalabilidad y la performance.

Negativas:

- El relacionamiento de datos puede ser complejo dado que es una base de datos no relacional.

Al no poder manejar relaciones, muchas veces hay mucha información duplicada que puede llevar a un gran uso de memoria.

Vista de asignación - Conexión con MongoDB para la Plataforma.



Catálogo de elementos

Elementos	Descripción
User	Usuario utilizando Postman.
Create Appointment	Componente encargado de la creación de una cita para un profesional.
Get Appointment	Componente encargado de obtener las citas para un profesional.
Appointment By Date and Doc	Componente encargado de obtener las citas por fecha y profesional.
Add note to Appointment	Componente encargado de agregar notas a una cita.
Create Vaccine Schedule	Componente encargado del horario de vacunación.
Appointment Model	Modelo de citas de mongoose.
Vaccine Schedule Model	Modelo del horario de vacunación de mongoose.
Plataformdb	Base de datos de MongoDB

ADR

El ADR correspondiente a esta vista es: 004 - Uso de MongoDB como sistema de base de datos para la plataforma.

004 - Uso de MongoDB como sistema de base de datos para la plataforma.

Estado: aprobado

Contexto:
Se requiere crear una plataforma que controle la creación y consulta de citas con profesionales.

Decisión:
Decidimos utilizar el sistema de base de datos MongoDB porque las consultas deben poder realizarse en un tiempo de menos de 200 milisegundos, demostrando que, para un número elevado de usuarios concurrentes, los tiempos se mantienen relativamente constantes en el tiempo. Además el uso de este sistema, puede ser beneficioso en caso de que se necesite la modificación de las citas.

Consecuencias
Positivas:

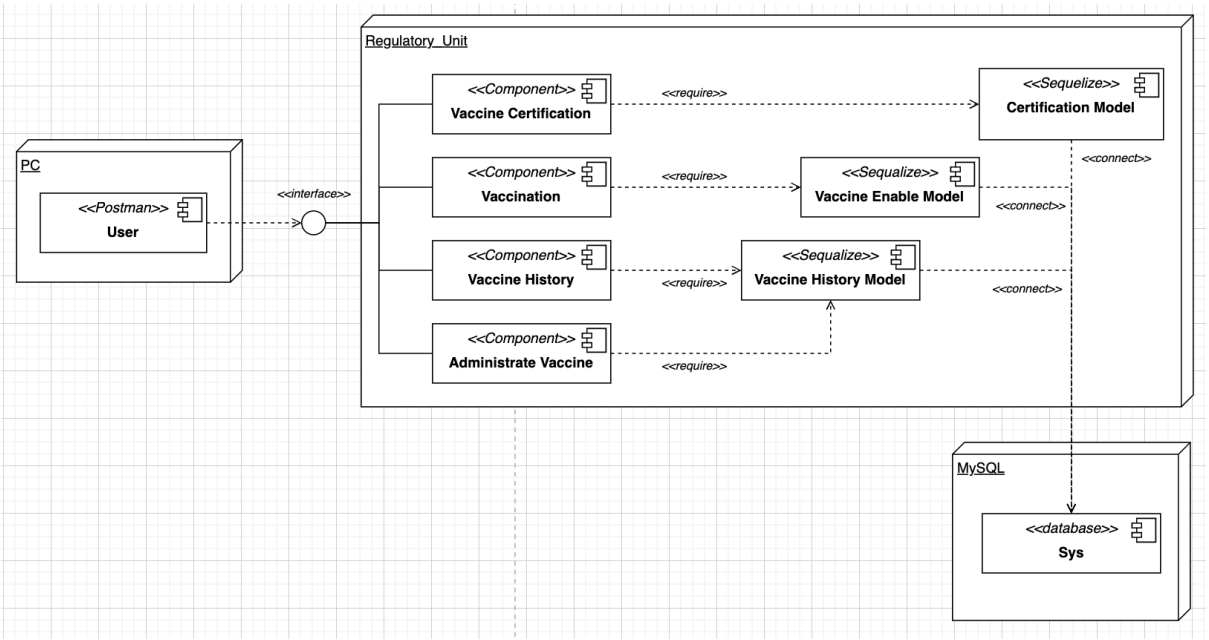
- Sencillo de implementar y manejar. Gracias a la aplicación MongoDB Compass, los datos se pueden ver y comprender sin complicaciones.
- Maneja una sintaxis muy parecida a JSON, lo que lo hace simple de modificar y crear datos nuevos o existentes.
- Útil en aplicaciones donde se esperan muchas citas y se necesita escalar la capacidad de almacenamiento.

Negativas:

- Puede generar un alto consumo de recursos cuando las consultas son muy complejas en grandes volúmenes de datos como es el caso de las citas.

Al no poder manejar relaciones, muchas veces hay mucha información duplicada que puede llevar a un gran uso de memoria.

Vista de asignación - Conexión con MySQL para la Unidad Reguladora.



Catálogo de elementos

Elementos	Descripción
User	Usuario utilizando Postman.
Vaccine Certification	Componente encargado del certificado de vacunación.
Vaccination	Componente encargado de la creación de vacunas.
Vaccine History	Componente encargado del historial de vacunación del usuario.
Administrate Vaccine	Componente encargado de la administración de vacunas al usuario.
Certification Model	Esquema de certificación de vacunas de Sequelize.
Vaccine Table Model	Esquema de vacunas de sequelize.
Vaccine History Model	Esquema del historial de vacunación del usuario.
Sys	Base de datos de MySQL.

ADR

El ADR correspondiente a esta vista es: 005 - Uso de MySQL como sistema de base de datos para la unidad reguladora.

005 - Uso de MySQL como sistema de base de datos para la unidad reguladora.

Estado: aprobado

Contexto:
Se requiere crear una unidad reguladora de vacunas, vacunatorios y administración de vacunas a usuarios.

Decisión:
Decidimos utilizar el sistema de base de datos MySQL y más específicamente una base de datos relacional, para brindar un sistema más completo y expresivo a la hora de registrar y consultar el acceso a vacunas y vacunatorios de la unidad reguladora.

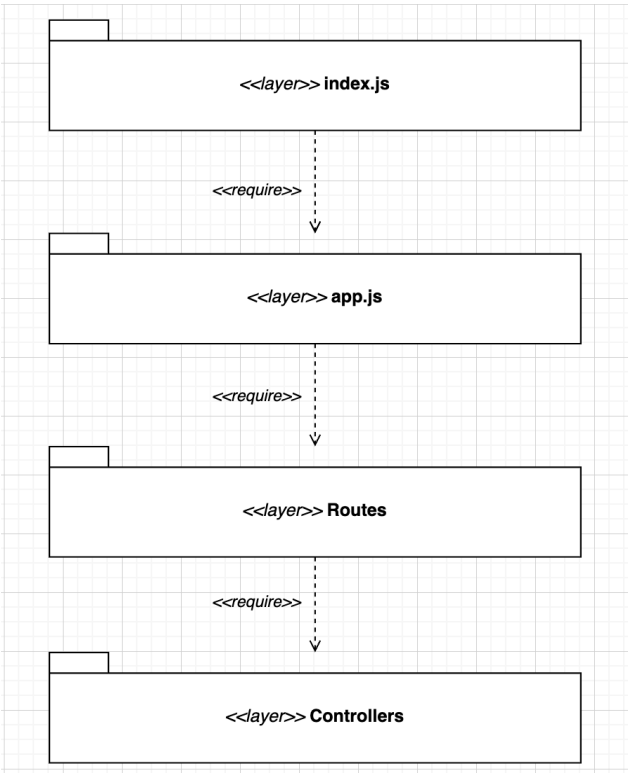
Consecuencias
Positivas:

- Provee una gran escalabilidad para manejar grandes volúmenes de datos que es importante considerar cuando hablamos de muchas vacunas y vacunatorios registrados.
- Al ser una base de datos relacional tiene mayor expresividad en las relaciones entre datos.
- Mejor representación de la realidad gracias a la expresividad de las relaciones.

Negativas:

- Puede que sea más complejo hacer ciertas consultas a la base de datos relacional por la complejidad en las relaciones.

Vista de módulos -Separación de capas



Catálogo de elementos

Elementos	Descripción
index.js	Módulo encargado de levantar el servidor
app.js	Middleware de enrutamiento.
Routes	Módulo encargado de la definición de las rutas.
Controllers	Módulo encargado de la implementación de los controladores.

ADR

El ADR correspondiente a esta vista es: 006 - Separación modular o en capas.

006 - Separación modular o en capas.

Estado: aprobado

Contexto:
Se requiere la separación clara entre capas de la aplicación para favorecer la modificabilidad del sistema.

Decisión:
Hemos decidido dividir los tres servicios, el sistema de autenticación, la plataforma y la unidad reguladora, en capas claras de acuerdo a la funcionalidad con el fin de mejorar la mantenibilidad del sistema. Con esta división fomentamos la reusabilidad del código de los módulos. La división fue hecha en 4 capas: conexión del servidor, middleware enrutador, declaración de rutas e implementación de controladores.

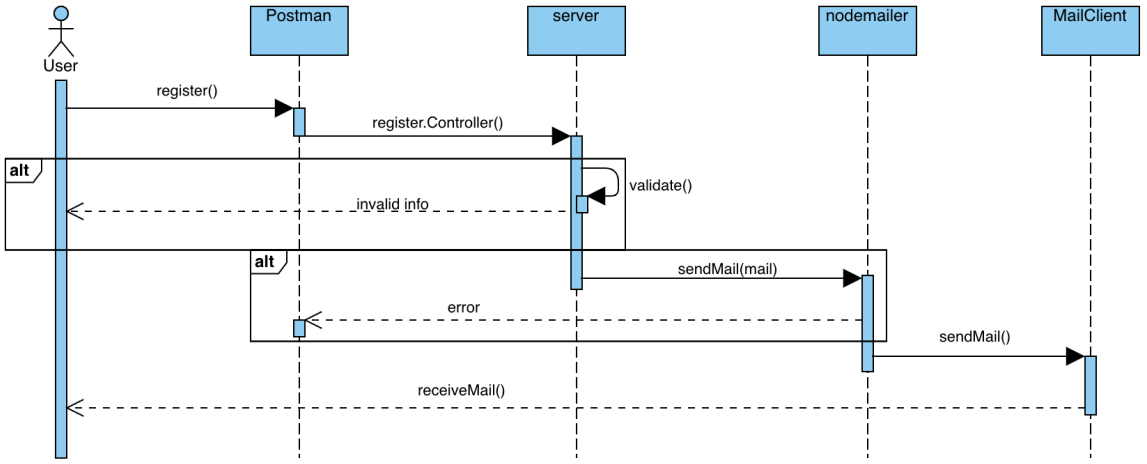
Consecuencias
Positivas:

- Provee modificabilidad al permitir que nuevos módulos sean incorporados a la capa correspondiente sin mayor complejidad.
- Permite reusabilidad en distintas partes del código sin tener que volver a codificar.
- Mejor comprensión de la estructura de archivos al tener separadas los módulos similares.

Negativas:

- Puede ocasionar dificultades a la hora de la navegación por el código.

Vista de componentes y conectores - Autenticación de usuario vía email.



Catálogo de elementos.

Elementos	Descripción
Postman	Servicio que permite hacer consultas al servidor.
Server	Servidor de autenticación de usuarios.
nodemailer	Servicio que permite el empaquetado del email
MailClient	Cliente de email del usuario.

ADR

El ADR correspondiente a esta vista es: 007 - Autenticación de usuario mediante email.

007 - Autenticación de usuario mediante email.

Estado: aprobado

Contexto:
Se debe enviar un email para verificar la autenticidad del usuario, mientras tanto la situación del mismo quedará en estado pendiente.

Decisión:
Hemos decidido implementar la autenticación en dos factores con la finalidad de autenticar al usuario que se está registrando. Cuando el usuario escriba correctamente su información de contacto junto a su contraseña, el servicio pedirá un segundo paso de verificación mediante el envío de un email al correo brindado.

Consecuencias
Positivas:

- La autenticación proporciona una capa adicional de seguridad al requerir dos formas distintas de autenticación.

Negativas:

- Puede ocasionar complejidad adicional a la hora de codificar.
- Puede ocasionar complejidad a ciertos usuarios al momento de registrarse en caso de que desconozcan el proceso o carecen de acceso a dispositivos adicionales..