Desarrollo del sistema de gestión de turnos de atención a gran escala: Sector salud.

Angie Alejandra Chipatecua Zárate

Trabajo de grado para optar el título de Ingeniero de Sistemas

Director

Edison Reyes Forero

Ingeniero de Sistemas

Universidad de Cundinamarca

Facultad de Ingeniería

Programa Ingeniería de Sistemas

Fusagasugá

2022

Dedicatoria

A mi abuelo

A mi abuelo Adonai Zárate, por ser mi motivación diaria incluso no estando en el mundo terrenal. Gracias por siempre estar orgulloso de mí, todo el resultado de este proceso es para ti, aunque ya no estés con nosotros.

A mis padres

A mi apoyo incondicional: Mi madre, María Del Pilar Zárate y mi padre Hollman Fresney Chipatecua. Les agradezco por ser lo más sagrado para mí y por acompañarme de gran manera en este proceso que inició desde hace 5 años; sin duda alguna este logro va acompañado de cada esfuerzo generado por ustedes.

A mis hermanos

Didier Esneider Chipatecua y Fabián Jadir Chipatecua; mis compañeros de vida. Les agradezco por estar siempre para mí, tanto en este proceso, como en todos en los que han estado desde que nací.

A una persona especial

A Diego Alejandro Sierra, por ser la persona que más confió en mí como profesional, apoyándome en este proceso y reflejando su orgullo hacia mí. Gracias por hacer parte de mi vida y por ser tan leal conmigo.

A mis compañeros

A todos mis compañeros, que con sus conocimientos, ocurrencias y personalidades complementaron y apoyaron todo mi proceso. Especialmente a Sergio Blanco y Cristhian Monrroy, por ser unas personas maravillosas, incondicionales y con un gran potencial.

Agradecimientos

A la empresa

Agradezco a la empresa que me brindó la oportunidad de aprender y así mismo ser parte de su equipo, permitiéndome dar mi máximo potencial. Gracias por siempre creer en mí, por su colaboración constante y sobre todo por ser mis mentores en estos 8 meses.

Al docente Edison Reyes Forero, mi director interno

Ingeniero, gracias por siempre creer en mí, por tenerme en cuenta durante todo este proceso universitario y por apoyarme en todo lo que fuera necesario. Le agradezco por ser mi docente, mi director interno y por ser un ejemplo por seguir.

A mis amigos

A Duver Alexander Melo Mendivelso y Oscar David Sabogal, por guiarme con su experiencia en este proceso, y colaborándome con consejos para realizar correctamente el presente trabajo.

Tabla de contenido

1. Introducción xxi

2. Planteamiento del Problema xxiii

3. Justificación xxv

4. Objetivos xxvi

Objetivo General xxvi

Objetivos Específicos xxvi

5. Limitaciones y Alcance xxvii

Alcance xxvii

Limitaciones xxvii

6. Marco Referencial xxix

6.1. Antecedentes xxix

6.2. Marco Teórico xxxi

6.2.1. Evolución del Desarrollo de Software a Nivel Empresarial xxxi

6.2.2. Herramientas Implementadas Tradicionalmente xxxi

6.2.3. Cultura xxxii

6.2.4. Metodologías Ágiles xxxiii

6.3. Marco de Trabajo xxxiv

6.3.1. Metodología Ágil SCRUM xxxiv

6.4. Marco Conceptual xl
6.4.1. API REST xl
6.4.2. Automatización xl
6.4.3. Backend xl
6.4.4. Base de Datos xl
6.4.5. Clean Architecture xl
6.4.6. CRUD xli
6.4.7. Frontend xli
6.4.8. Localhost xli
6.4.9. Máquina Virtual xli
6.4.10. NoSQL xli
6.4.11. Repositorio xli
6.4.12. Sistema Web xlii
6.5. Marco Tecnológico xlii
6.5.1. Azure xlii
6.5.2. Couchbase DB xlii
6.5.3. Docker xliii
6.5.4. FastAPI xliii
6.5.6. GitLab xliii
6.5.7. IBM WorkFlow Process Service xliii

6.5.8. Microsoft Teams xliv

6.5.9. Notion..... xliv

6.5.10. Python xliv

6.5.11. Swagger..... xlv

6.5.12. Sophos SSL VPN Client xlv

6.5.13. Trello..... xlv

6.5.14. Termius xlv

6.5.15. Unittest..... xlvi

6.5.16. Visual Studio Code xlvi

6.5.17. Webex xlvi

7.Empresa..... xlvii

7.1. ¿Quiénes son?..... xlvii

7.2. Misión..... xlvii

7.3. Visión xlvii

7.4. Valores xlvii

8. Actividades Realizadas xlviii

8.1. Sprint 1. Módulo Del Turnero..... xlviii

8.1.1. Análisis..... xlix

8.1.2. Diseño l

8.1.3. Desarrollo y Pruebas liii

8.2. Sprint 2. Módulo de Atención en Ventanilla lvii

8.2.1. Análisis lviii

8.2.2. Diseño lviii

8.2.3. Desarrollo y Pruebas lxiii

8.3. Sprint 3. Módulo del Tablero-Turnos en Pantalla lxvii

8.3.1. Análisis lxviii

8.3.2. Diseño lxix

8.3.3. Desarrollo y Pruebas lxx

8.4. Sprint 4. Módulo Administrativo: Salas y Sedes lxxv

8.4.1. Análisis lxxvi

8.4.2. Diseño lxxvii

8.4.3. Desarrollo y Pruebas lxxxi

8.5. Sprint 5. Módulo Administrativo: Servicios y Ventanillas lxxxv

8.5.1. Análisis lxxxvi

8.5.2. Diseño lxxxvii

8.5.3. Desarrollo y Pruebas xc

8.6. Sprint 6. Módulo administrativo: Tableros, Turneros y Re-inicialización de Turneros xciv

8.6.1. Análisis xcvi

8.6.2. Diseño xcvi

8.6.3. Desarrollo y Pruebas c

8.7. Sprint 7. Módulo Tablero de Control cvi

8.7.1. Análisis cvii

8.7.2. Diseño cvii

8.7.3. Desarrollo y Pruebas cix

8.8. Sprint 8. Módulo Tablero de Indicadores cxiii

8.8.1. Análisis cxiv

8.8.2. Diseño cxv

8.8.3. Desarrollo y Pruebas cxvii

8.9. Sprint 9. Concurrencia cxx

8.9.1. Análisis cxxi

8.9.2. Diseño cxxiii

8.9.3. Desarrollo y Pruebas cxxiii

8.10. Sprint 10. Asignación de Roles cxxvii

8.10.1. Análisis cxxviii

8.10.2. Diseño cxxix

8.10.3. Desarrollo y Pruebas cxxix

8.11. Sprint 11. Despliegue de la Versión 1 cxxxi

8.11.1. Análisis cxxxii

8.11.2. Diseño cxxxiii

8.11.3. Desarrollo cxxxiii

9. Conclusiones cxxxvii

Referencias bibliográficas cxl

Anexos cxlv

Lista de tablas

Tabla 1 Roles de Scrum. xxxv

Lista de figuras

Figura 1 Reuniones Diarias. xxxvii

Figura 2 Agenda Reuniones Revisión y Retrospectiva. xxxvii

Figura 3 Reuniones Revisión y Retrospectiva. xxxviii

Figura 4 Organización, Asignación y Detalles de Actividades en Trello. xxxix

Figura 5 Sprint Backlog: Primer Sprint. xlix

Figura 6 Evidencia de Estructura en la Base de Datos. l

Figura 7 Estructura de Datos Para la Colección de Turnos. l

Figura 8 Prototipo del Inicio del Turnero. li

Figura 9 Prototipo de Las Opciones de Atención por Turno. lii

Figura 10 Prototipo del Ticket por Turno. lii

Figura 11 Bucket, Scope y Colección en Couchbase. liii

Figura 12 Endpoints Sprint 1. liii

Figura 13 Evidencia Arquitectura a Nivel de Backend. liv

Figura 14 Prueba Unitaria Sprint 1. liv

Figura 15 Resultado Prueba Unitaria Sprint 1. lv

Figura 16 Workflow Módulo Turnero. lvi

Figura 17 Tablas de Decisión para las Reglas de Negocio. lvi

Figura 18 Evidencia Correlación de Datos. lvi

Figura 19 Evidencia Prueba de Integración Sprint 1. lvii

Figura 20 Sprint Backlog 2. lviii

Figura 21 Estructura de Datos Para la Colección de Ventanillas. lix

Figura 22 Prototipo de Inicialización de Ventanilla. lix

Figura 23 Prototipo de Disponibilidad de Ventanilla. lx
Figura 24 Prototipo de Ventanilla Ocupada. lxi
Figura 25 Prototipo de Turno Ausente. lxi
Figura 26 Prototipo de Turnos No Disponibles. lxii
Figura 27 Endpoints Sprint 2. lxiii
Figura 28 Evidencia Prueba Unitaria Sprint 2. lxiv
Figura 29 Response Prueba Unitaria Sprint 2. lxiv
Figura 30 Workflow Sprint 2. lxv
Figura 31 Evidencia Correlación de Datos en Una Tarea de Servicio Sprint 2. lxv
Figura 32 Evidencia Documentación Swagger Sprint 2. lxvi
Figura 33 Prueba de Integración Sprint 2. lxvi
Figura 34 Sprint Backlog 3. lxviii
Figura 35 Estructura de Datos Sprint 3. lxix
Figura 36 Prototipo Turnos en Pantalla. lxix
Figura 37 Endpoints Sprint 3. lxxi
Figura 38 Prueba Unitaria Sprint 3. lxxi
Figura 39 Flujo de Trabajo Sprint 3. lxxii
Figura 40 Invocación de Un Servicio. lxxiii
Figura 41 Correlación de Datos. lxxiii
Figura 42 Prueba de Integración Sprint 3. lxxiv
Figura 43 Sprint Backlog 4. lxxvi
Figura 44 Estructura de Datos: Sedes y Salas. lxxvii
Figura 45 Panel Administrativo. lxxvii

Figura 46 Prototipos de Modales Para Notificaciones En El Módulo. lxxviii

Figura 47 Prototipos Sección Sedes. lxxviii

Figura 48 Prototipos Sección Salas. lxxix

Figura 49 Secciones Modales Para Opciones En Sede. lxxix

Figura 50 Secciones Modales Para Opciones en Sala. lxxx

Figura 51 Endpoints Sprint 4. lxxxii

Figura 52 Evidencia Pruebas Unitarias Sprint 4. lxxxii

Figura 53 Flujo de Trabajo Sección Sedes. lxxxiii

Figura 54 Flujo de Trabajo Sección Salas. lxxxiv

Figura 55 Evidencia Prueba de Integración Sprint 4. lxxxv

Figura 56 Sprint Backlog 5. lxxxvi

Figura 57 Estructura de Datos: Servicios y Ventanillas. lxxxvi

Figura 58 Prototipo Sección Servicios. lxxxvii

Figura 59 Prototipo Sección Ventanillas. lxxxviii

Figura 60 Secciones Modales Para Opciones en Ventanilla. lxxxviii

Figura 61 Secciones Modales Para Opciones en Servicio. lxxxix

Figura 62 Endpoints Sprint 5. xci

Figura 63 Evidencia Prueba Unitaria Sprint 5. xcii

Figura 64 Flujo de Trabajo Sección Servicios. xcii

Figura 65 Flujo de Trabajo Sección Ventanillas. xciii

Figura 66 Prueba de Integración Sprint 5. xciv

Figura 67 Prueba de Integración-Nivel de Usuario Sprint 5. xciv

Figura 68 Sprint Backlog 6. xcv

Figura 69 Estructura de Datos: Tableros, Turneros y Reinicio Turneros. xcvi

Figura 70 Prototipo Sección de Turneros..... xcvii

Figura 71 Prototipo Sección de Tableros. xcvii

Figura 72 Prototipo Sección de Re-inicialización de Turneros. xcviii

Figura 73 Secciones Modales Para Opciones de Turneros. xcviii

Figura 74 Secciones Modales Para Opciones de Tableros..... xcix

Figura 75 Endpoints Sprint 6. c

Figura 76 Evidencia Prueba Unitaria Sprint 6. ci

Figura 77 Flujo de Trabajo Sección Turneros..... cii

Figura 78 Flujo de Trabajo Sección Tableros. ciii

Figura 79 Evidencia Creación de Variables. civ

Figura 80 Prueba de Integración Sprint 6. cv

Figura 81 Sprint Backlog 7. cvi

Figura 82 Prototipo Inicio del Módulo Tablero de Control. cvii

Figura 83 Prototipo Sección Administración de Ventanillas. cviii

Figura 84 Prototipo Sección Administración de Turnos. cviii

Figura 85 Endpoints Sprint 7. cix

Figura 86 Evidencia Prueba Unitaria Sprint 7. cx

Figura 87 Flujo de Trabajo Tablero de Control. cxi

Figura 88 Evidencia Estado Predeterminado Por Estado de Ventanilla. cxi

Figura 89 Prueba de Integración Sprint 7: Listado de Turnos Actuales Previo a Actualización. cxii

Figura 90 Prueba de Integración Sprint 7: Actualización Estado de Turno Posterior a Prueba. cxii
Figura 91 Sprint Backlog 8. cxiii
Figura 92 Prototipo Sección Turnos-Tablero de Indicadores. cxv
Figura 93 Prototipo Sección Tiempo de Espera-Tablero de Indicadores. cxvi
Figura 94 Prototipo Sección Servicios-Tablero de Indicadores. cxvi
Figura 95 Endpoints Sprint 8. cxvii
Figura 96 Evidencia Prueba Unitaria Sprint 8. cxviii
Figura 97 Flujo de Trabajo Sprint 8. cxix
Figura 98 Evidencia Prueba Unitaria Sprint 8. cxix
Figura 99 Sprint Backlog 9. cxxi
Figura 100 Prueba Unitaria Sprint 9: Tiempo de Espera. cxxiii
Figura 101 Prueba Unitaria Sprint 9: Asignación de un Turno a Ventanilla. cxxiv
Figura 102 Prueba Unitaria Sprint 9: Crear Turno. cxxv
Figura 103 Prueba de Integración Sprint 9- Turno Existente. cxxvi
Figura 104 Prueba de Integración Sprint 9- Turno no Disponible. cxxvi
Figura 105 Sprint Backlog 10. cxxvii
Figura 106 Evidencia Creación de Equipos. cxxix
Figura 107 Evidencia Creación de Procesos. cxxx
Figura 108 Evidencia Proceso en Carril Específico. cxxx
Figura 109 Proceso Ejecutándose en el Workplace. cxxxi
Figura 110 Sprint Backlog 11. cxxxi
Figura 111 Coverage Proyecto- Python. cxxxii

Figura 112 Merge Dev Into Main. cxxxiii

Figura 113 Contenedores en VM. cxxxiii

Figura 114 Creación de Recursos Necesarios en Azure. cxxxiv

Figura 115 Archivos de Ejecución de App Service. cxxxiv

Figura 116 Variables de Entorno. cxxxv

Figura 117 Ejecución Archivos Para Inicio del App Service en Powershell. cxxxv

Figura 118 Ejecución de IBM WFPS en el Servidor Posterior al Despliegue. cxxxv

Figura 119 Ejecución de Couchbase en el Servidor Posterior al Despliegue. cxxxvi

Glosario

Bucket: Es aquel espacio fundamental en la base de datos para almacenar su información por categorías.

Clean Architecture: Es una arquitectura que se fundamenta por su capacidad de desacoplar el código y organizarlo de acuerdo con su funcionalidad y propósito.

Coach: Es un componente que funciona como interfaz, permitiendo incluir código y así mismo tomar widgets para su implementación y diseño.

Colecciones: Es donde se contienen ciertos documentos de la base de datos, esto de acuerdo con la categoría que se le brinde a sí misma.

Debug: Es el modo aplicado en el proyecto para validar el correcto funcionamiento de todos sus componentes integrados, permitiendo realizar un paso a paso progresivo reflejando errores o su correcta ejecución.

Endpoint: Es aquel canal o intermediario para llevar a cabo la comunicación entre el código y una aplicación externa.

Entities: Representan las clases u objetos que se manejarán a nivel de backend.

Estructura: Representa los datos o atributos requeridos dentro de un documento de una colección de la base de datos.

Flujo de servicio: Es el componente que permite invocar un servicio por medio del consumo de los endpoints dentro de una herramienta.

JSON: Tipo de archivo que cuenta con unos datos en específico para permitir que sean intercambiados posteriormente.

NoSQL: Representa a las bases de datos que no son relacionales, caracterizándose por su flexibilidad frente al manejo de su información.

Operario de servicio: Es aquella persona encargada de atender a los pacientes o usuarios en una entidad.

Paciente: Es un usuario que busca la prestación de un servicio con fines personales.

Repositories: Representa una de las capas del Clean Architecture, almacenando en ella todas las consultas que deben realizarse a la base de datos.

REST: Se encarga de realizar operaciones en los datos, esto por medio de HTTP.

Servicio externo: Es el componente que le permite a la herramienta de automatización consumir todos los endpoints albergados en una documentación y un servidor.

Servidor: Es el sistema que recibe peticiones y de acuerdo a las operaciones envía una respuesta.

Scope: Es el componente que almacena todas las coleccionas creadas en la base de datos.

Uses cases: Refleja la capa de Clean Architecture encargada de implementar las funciones o casos específicos requeridos en el proyecto.

VM: Hace referencia a una máquina virtual que albergará los contenedores necesarios en el proyecto.

Widgets: Es un elemento que permite visualizar y cumplir con una funcionalidad dentro de un coach (En función del frontend).

Workflow: Es aquel flujo de trabajo donde se encuentran diversas actividades para suplir una necesidad y automatizar cualquier tipo de proceso.

Resumen

El presente informe tiene como objetivo dar a conocer el respectivo proceso y los aspectos que se abarcaron a la hora de contribuir con el desarrollo del proyecto denominado: “Desarrollo del sistema de gestión de turnos de atención a gran escala: Sector salud”, realizado como trabajo de trabajo, específicamente en la modalidad de pasantías.

Dicho lo anterior, es de importancia resaltar que este proyecto busca crear un sistema o aplicación que reduzca los trabajos manuales o cualquier tipo de proceso que no sea óptimo en las empresas promotoras de salud, para de esta forma replantear los sistemas que manejan actualmente y brindar una solución óptima, flexible e innovadora que sea capaz de permitir que la atención de los turnos, de principio a fin sea la indicada, generando beneficios a nivel general y empresarial.

Palabras Clave: Pasantías, turnos, atención, sistema, salud.

Abstract

The objective of this report is to make known the respective process and the aspects that were covered when contributing to the development of the project called: "Large-scale care shift management system: Health Sector", carried out as a research project. work, specifically in the form of internships.

That said, it is important to highlight that this project seeks to create a system or application that reduces manual work or any type of process that is not optimal in health promoting companies, in order to rethink the systems they currently manage and provide an optimal, flexible and innovative solution that is capable of allowing the attention of the shifts, from beginning to end, to be the indicated one, generating benefits at a general and business level.

Keywords: Internships, shifts, attention, system, health.

1. Introducción

Hoy en día la gran mayoría de empresas buscan optimizar sus procesos por medio de sistemas, herramientas o cualquier otro medio que les permita, tanto cumplir sus objetivos, como brindar la debida atención a sus usuarios, reflejando así la correcta prestación de sus servicios, principalmente. Con respecto al primer punto, se ha visto reflejado que el sector salud ha tenido mejoras frente a sus ejes principales, siendo uno de ellos: La calidad y el acceso efectivo, puesto que han aplicado tecnologías, que permiten optimizar gastos y así mismo prestar una mejor atención (Ministerio de salud, 2020). Dentro de lo que respecta a la calidad, claramente se abarca la gestión de los turnos solicitados por los pacientes y la gestión que se le es brindada a los mismos, pues es de los principales procesos que se deben tener en cuenta la hora de prestar un servicio en este ámbito.

De acuerdo a lo considerado con anterioridad y teniendo en cuenta el auge que tiene el desarrollo de herramientas para el sector salud, para la empresa Redes y Sistemas Integrados S.A.S fue relevante buscar, plantear y llevar a cabo la creación de un sistema que abarcara la atención de turnos, buscando innovar y mejorar con su solución buscando destacarse frente a otros productos del mercado, pues esta es uno de los puntos hallados dentro de su misión, el centrarse en soluciones tecnológicas, transformando toda idea con innovación y calidad (Redes y Sistemas Integrados S.A.S, s.f).

El presente informe final, busca demostrar y evidenciar de manera global las actividades realizadas para la contribución en el desarrollo del proyecto, el cual requirió tanto de conocimientos básicos en el área, como de aprendizaje en herramientas específicas, las cuales se mencionarán posteriormente. Para el desarrollo de este proyecto fue relevante el uso de la metodología SCRUM, pues sus características y objetivos se adecuaban a la modalidad para

desarrollar la solución y así mismo para evidenciar cada procedimiento o método generado en la pasantía para obtener un producto funcional.

2. Planteamiento del Problema

Actualmente, los sistemas de gestión de turnos son considerados como aquella plataforma que se encarga de gestionar, ordenar y optimizar todo lo que respecta al proceso de atención al cliente de forma presencial (Del Valle, 2012). Este procedimiento se maneja mayormente de manera sistemática, por medio de estas plataformas, facilitando y mejorando la productividad de las empresas, sin embargo, se evidencia en diversas entidades la ejecución de procesos manuales que traen como consecuencia el desarrollo tedioso de las actividades presentando tardanzas tanto para el cliente como para el prestador del servicio (Ramos, 2019). Al presentar este tipo de retrasos, se genera una insatisfacción del cliente, lo que conlleva a una disminución del nivel de calidad brindado por la entidad, igualmente a una ineficiencia frente a ejecuciones básicas dentro de la organización e incluso al retiro del usuario (Meneses, 2017).

El sector salud es uno de los ámbitos con más presencia de usuarios a nivel nacional, a causa de ser un servicio básico para la comunidad, es por esto que las entidades promotoras de este tipo de actividades deben hacer lo posible porque la sociedad esté satisfecha y pueda acceder al derecho a una atención de calidad, pues tal como lo afirma (Santa María et al., 2009) su estado de bienestar médico está directamente ligado a sus motivaciones, e incluso su productividad, que derivarán aspectos positivos dentro de su entorno. Es por ello que toda empresa que ejerza un servicio de salud debería de centrarse en diferentes ámbitos, entre ellos la correcta atención del cliente para que pueda llevar a cabo sus procesos médicos, iniciando desde un turno para una gestión de citas médicas, hasta el pago de la misma, por ejemplo.

De acuerdo al razonamiento expuesto, la empresa Redes y Sistemas Integrados S.A.S, reconocida en el negocio de la tecnología de la información (suministro, instalación y soporte de soluciones enfocadas en este ámbito), ha decidido abarcar esta problemática, y, por consiguiente

desarrollar un producto que permita la optimización de los turnos de atención en el sector salud, utilizando la plataforma de automatización Cloud Pak for Business Automation, esto con el fin de suplir las necesidades expuestas y halladas actualmente, y, simultáneamente competir en el mercado con un sistema que, además de ser propio pueda generar un bienestar tanto para las empresas promotoras de la salud como para los usuarios de dichas entidades.

3. Justificación

De acuerdo a la problemática mencionada, es evidente que actualmente se promueve el uso de sistemas automatizados para el sector de la salud con el fin de agilizar procesos dentro de las organizaciones, entre ellos la gestión de turnos, ya que este tipo de plataformas evitarán un servicio deficiente para los usuarios, mejorando la calidad de su estancia en la entidad e incluso reducir su tiempo de espera para la obtención del servicio (Landázuri et.al, s.f), contribuyendo así, por ejemplo, a la disminución de inconformidades de los beneficiarios frente a la tardanza de sus procesos y simultáneamente mejorar la productividad en el entorno empresarial, pues este tipo de sistemas no contribuyen únicamente en los servicios para los pacientes, ya que también mejoran la gestión del trabajo del operario o de la persona encargada de este proceso, permitiendo que se centre en otras actividades.

De acuerdo a lo mencionado previamente y teniendo en cuenta la problemática planteada, la empresa en contexto busca aprovechar esta oportunidad en el mercado desarrollando un nuevo producto basado en algoritmos propios, apoyándose en la plataforma de automatización e inteligencia artificial Cloud Paks for Business Automation y en demás herramientas requeridas, para de esta forma facilitar la administración y ejecución de procesos específicos, brindando así al sector salud un sistema potente y maduro, capaz de optimizar los procesos que abarca el gestionar turnos para sus usuarios evitando de esta manera procedimientos ineficientes y demostrando una mejora frente a otras soluciones que ya se encuentran en el mercado.

4. Objetivos

Objetivo General

Contribuir con el desarrollo del sistema de gestión de turnos de atención a gran escala: Sector salud, implementando herramientas de automatización de procesos, esto con el fin de brindar un producto de calidad a las entidades promotoras de salud, reduciendo el trabajo manual y la ejecución de tareas repetitivas.

Objetivos Específicos

- Identificar los procesos actuales en el sector respecto a la gestión de turnos, determinando la capacidad de optimización de este.
- Realizar el diseño del modelo para el sistema de gestión de turnos de atención a gran escala: Sector salud, de acuerdo con los requerimientos delimitados.
- Construir el sistema de gestión de turnos de atención a gran escala: Sector salud.
- Implementar la herramienta Cloud Pak for Business Automation para automatizar los procesos identificados dentro del análisis estipulado.
- Realizar test de prueba determinando un posible versionamiento.

5. Limitaciones y Alcance

Alcance

Apoyar el desarrollo del sistema de gestión de turnos de atención a gran escala: Sector Salud, haciendo énfasis en la generación de procesos en la herramienta de automatización de IBM, contribuyendo igualmente en la programación de los servicios web que serán consumidos por otros componentes y por la IBM Workflow Process Service.

Limitaciones

Al momento de ingresar al proceso de la pasantía y simultáneamente al desarrollo del proyecto la empresa contaba con un análisis de la posible problemática a abarcar, y gracias a esto una ruta de trabajo inicial, por lo que la fase de planificación no estará evidenciada de gran manera en este informe, esto debido a que se realizó una adaptación y entendimiento de los procesos que se realizaron en esta fase, sin embargo, no se contó con la presencia de la pasante durante ella.

Durante el desarrollo del proyecto se generó una curva de aprendizaje frecuente, esto con la finalidad de dominar las herramientas a utilizar e incluso tener conocimiento de los principales procesos que se realizaban en el sector salud, por lo tanto, gran parte del tiempo durante cada fase y proceso fue enfocado a la obtención de dichos conocimientos.

Debido a los objetivos planteados para este proyecto y de acuerdo a políticas de la empresa, todo aquel participante en el presente desarrollo tiene como obligación manejar un entorno confidencial y de reserva frente a cada actividad que realice, por lo cual, se plantea una explicación general de cada proceso durante este informe, y son censurados los elementos que sean relevantes para la empresa durante las evidencias gráficas presentadas en este documento, esto con el fin de no incumplir frente a las obligaciones otorgadas al momento de formar parte de

la empresa y así mismo contribuir con la tranquilidad, confidencialidad y principios de la organización. Cabe resaltar que se ha generado un repositorio para subir cada evidencia planteada, sin embargo, muchos de los archivos realizados no serán compartidos, esto debido a la misma política de confidencialidad mencionada; en su lugar se encontrarán principalmente los procesos propios de la pasante, como documentación, actividades autónomas, refuerzos de los conocimientos adquiridos por medio de capacitaciones, entre otros.

La revisión de los avances dirigida por los directores de la pasantía fue realizada con total cumplimiento, sin embargo, muchas de estas revisiones se generaron durante una única reunión, esto para optimizar tiempo entre los involucrados, por lo que varias actas de reunión contarán con la misma fecha de encuentro, pero su contenido será explicado a totalidad de acuerdo con lo realizado durante cada mes.

6. Marco Referencial

6.1. Antecedentes

Actualmente, gracias a la digitalización y avance tecnológico, la mayoría de empresas optan por implantar sistemas que les permitan agilizar e incluso automatizar procesos, mejorando su productividad y la atención que debe brindarse al cliente, pues el uso de sistemas permite abandonar los procedimientos tediosos y manuales que son llevados a cabo por su equipo de trabajo, generando que estos puedan dedicarse a tareas más importantes y efectivas, brindando eficacia para la organización (Landázuri et.al, s.f). De acuerdo con estos beneficios dados por varios autores con base en los sistemas de gestión de turnos, se decide realizar una investigación de las posibles soluciones desarrolladas en los últimos años, con el fin de analizar su eficacia en la sociedad y posteriormente ser tomadas como referencia para el proyecto en cuestión.

Dicho lo anterior, se tiene como primera referencia el “Rediseño y reingeniería sistema para control de turno” (Rodríguez & Arciniegas, 2019), el cual quería dar a conocer de forma detallada la realización y el seguimiento correcto de este sistema, evidenciando todos los módulos necesarios para la correcta implementación del mismo dentro de la organización BitPointer. Este proyecto tiene relación con el principal objetivo a cumplir debido a que busca evidenciar un desarrollo adecuado del sistema, dando a conocer su eficiencia frente a las posibles problemáticas que se den específicamente en este ámbito dentro de la organización.

La segunda referencia a considerar es la tesis denominada como “Diseño, desarrollo e implementación del sistema de gestión de turnos programados para el Sub-centro de Salud Carapungo” (Landázuri et.al, s.f). Este proyecto se caracterizó por mejorar la disponibilidad en cuanto a turnos, disminuyendo el tiempo de espera para la atención de los pacientes, facilitando la obtención de sus estudios e incluso sus procedimientos médicos. Este proyecto tiene una

relación directa con el objetivo a cumplir, debido a que busca mejorar las deficiencias en los procesos que se derivan de los turnos gestionados implementando un sistema capaz de aumentar la productividad dentro de la entidad.

Como tercer estudio seleccionado se encuentra el "Análisis, diseño y desarrollo de los módulos de turnos, cita previa y parte diario del sistema de gestión médico para áreas de salud, para el centro de salud "La Tola Vicentina" (Arias & Simbaña, 2012). Este proyecto tiene una relación estrecha con el objetivo a cumplir, gracias a la búsqueda de un impacto positivo frente a la implementación de este tipo de sistemas, obteniendo así la correcta organización de los registros de turnos de acuerdo a la necesidad planteada, evitando el desgaste de recursos físicos y humanos. Cabe resaltar que este proyecto permitió que el equipo de trabajo encargado de la actividad en específico referente a la gestión de turnos pudiera evitar el doble procesamiento de información y la carga exhaustiva de trabajo, permitiendo una mejor toma de decisiones frente a más tareas que fueran de mayor prioridad dentro del centro de salud.

Finalmente, se toma como cuarto y último antecedente la tesis realizada por (Pineda & Becerra, 2015) denominada como "Construcción de un sistema inteligente para la asignación dinámica de turnos en el área de consulta externa del hospital regional Isidro Ayora". Este proyecto enunciado se relaciona directamente con el objetivo en cuestión debido a su contribución con el nivel de satisfacción de los pacientes o usuarios gracias a la implementación de este sistema, el cual gracias al manejo de herramientas basadas en inteligencia artificial permiten la obtención de la información permitente para el ahorro de tiempo dentro de los procesos necesarios en la entidad. Un aspecto relevante a destacar en el proyecto referenciado es la mejora evidenciada gracias a este desarrollo, que además de manejar herramientas

relacionadas con la inteligencia artificial, puede ser un producto potencialmente demandado en el mercado en caso de decidir su comercialización.

6.2. Marco Teórico

6.2.1. Evolución del Desarrollo de Software a Nivel Empresarial

Usualmente el desarrollo de software la mayoría de las veces refleja un reto en toda empresa, pues la ejecución de cada paso a seguir debe ser correcta, y es por esto por lo que se centran en seguir una metodología, bien sea tradicional o agile, validando y organizando a todo un equipo de trabajo, y claramente arriesgándose a dos caminos: El éxito o el fracaso. Es por esta razón que conforme pasa el tiempo se han implementado herramientas y culturas que permitan a las empresas eficientizar y centralizar la gestión de sus proyectos para encaminarse a la optimización de sus procesos, la colaboración e interacción entre roles y claramente el cumplimiento de objetivos (Villela, 2021).

La metodología más popular para este ámbito es la agile, pues la mayoría de las empresas deciden implementarla por sus grandes beneficios y características; sin embargo, para el despliegue y las operaciones lo asocian con una empresa que se encargue del tema, brindando de esta forma problemáticas centradas en sus futuras versiones y funcionalidades en caso de que no utilicen y organicen de forma correcta sus procesos futuros y sobre todo los roles para de esta forma adaptarse a cambios futuros y mejoras de su sistema.(Zenkit, 2018).

6.2.2. Herramientas Implementadas Tradicionalmente

Por otra parte, es importante resaltar las aplicaciones de software que se manejan tradicionalmente en las empresas, buscando desempeñar su trabajo correctamente, esto evidenciándose en la gestión de sus proyectos y el desarrollo de este; a continuación, se mencionarán uno de ellos (Villela, 2021):

Infraestructura automatizada: Son todas las herramientas que se caracterizan por ser altamente configurables, parametrizables, para de esta forma automatizar sus procesos, a nivel operativo y aplicativo.

Banderas de funcionalidad. Esta es la estrategia centrada en el manejo de ramas en un repositorio, esto con el objetivo de que, al fusionarlas y probarlas, únicamente importe la dirigida al entorno de producción, pues esta contendrá todos los avances realizados por el equipo.

Control de versiones de código compartidas. Se centra en implementar un único repositorio para todo el equipo de trabajo.

IRC e IM Robots. Estas herramientas buscan mantener al equipo siempre en el mismo contexto, pues les permite comunicarse e interactuar en el momento que sea requerido.

6.2.3. Cultura

Además de herramientas, en toda empresa, se manejan una serie de actitudes que deben de presentarse en cada miembro del equipo del proyecto, permitiendo de esta forma un mejor ambiente laboral. Sin importar la metodología implementada, se puede notar la presencia de los siguientes valores (Villela, 2021):

Respeto. Busca que se mantenga la educación a nivel de interacción, pues todo miembro es importante en el proyecto.

Confianza. Se refleja en la transparencia a la hora de comunicar cualquier novedad dentro del proyecto, buscando resolverlo con el apoyo del equipo.

Evitar culpar. Esta cultura se centra en generar que cada miembro del equipo aprenda del error, evitando que estos vuelvan a presentarse, claramente sin importar quien fue el culpable.

Actitud saludable acerca de fallos. Se refiere a que a nivel de equipo busquen soluciones frente a las problemáticas halladas, buscando su origen y analizando como evitar que ocurran nuevamente.

6.2.4. Metodologías Ágiles

Retomando la idea anterior, y como bien ha sido planteado con anterioridad, toda empresa busca un aseguramiento del éxito en cuanto a sus proyectos, por lo cual se ha vuelto frecuente implementar metodologías de desarrollo, pues estas brindan una guía para aplicar tanto los procesos como los elementos que sean necesarios para llevar a cabo el desarrollo de software. Sin embargo, se ha evidenciado que los proyectos requieren de cambios frecuentes debido al entorno donde se están desarrollando, por lo cual no es suficientemente óptimo implementar una metodología tradicional puesto que suelen restringirse con los tiempos de desarrollo y ante todo en la flexibilidad; es por esta razón que surgen las metodologías ágiles, reflejándose en ellas una posible solución para este tipo de vacíos frente a la implementación metodológica. Las metodologías ágiles empiezan a tomar fuerza a inicios de la década de los 90’s, pues se buscaba tener procesos debidamente definidos con características tales como la calidad y un tiempo prudente frente al desarrollo generándose diversos enfoques que se consideraban potenciales para lograr aquel objetivo. Gracias a esto las metodologías agiles toman fuerza al momento de la creación de eXtreme Programming siendo su creador Kent Beck junto con las ideas de War Cunninghan, demostrando que llevando a cabo las prácticas planteadas con XP se cumple tanto con los objetivos como con el calendario propuesto, dando paso al movimiento de las metodologías ágiles. Ahora bien, en cuanto a aspectos formales, el término “ágil” se oficializa en el año 2001, donde participaron 17 personas, buscando elaborar valores y principios que

permitieran desarrollar el software y así mismo responder frente a diversos cambios obteniendo como resultado el manifiesto ágil. (Amaro & Valverde, 2007)

6.3. Marco de Trabajo

Como bien ha sido planteado con anterioridad, el desarrollo de esta pasantía se dirigía al sector salud, siendo más específicos a la realización de un sistema de gestión de turnos, lo cual requería de la unión de varios roles empresariales, pues se buscaba que el producto fuera abarcado y entendido en su totalidad, ya que no todos contaban con la experiencia y el conocimiento frente al tema, buscando que el equipo se colaborara entre sí, se utilizaran tecnologías que optimizaran la futura solución y la entrega rápida del mismo, por lo cual se decide implementar la metodología SCRUM, para lograr una colaboración global en caso de ser necesaria, y así mismo contar con una entrega y respuesta frente a cada avance de modo ágil, facilitando y optimizando el modo de trabajo y la armonía del equipo. (Schwaber et.al, 2020).

6.3.1. Metodología Ágil SCRUM

Tal y como es descrito por Schwaber et al. (2020), SCRUM es considerado un marco de trabajo ligero, que le brinda guía y apoyo a los equipos que deseen brindar valor por medio de cualquier tipo de solución, contando con un enfoque incremental que le permite optimizar los procesos y así mismo controlar los riesgos que se presenten durante el desarrollo del proyecto.

Esta metodología actualmente es popular en el entorno de desarrollo de software, pues al momento de adoptarlo, muchos equipos de trabajo han demostrado mejoras y transformaciones tanto en la parte productiva como en la parte moral, reflejando de esta forma las principales características de SCRUM y dando a conocer que no se requiere un equipo totalmente especializado únicamente para un solo proceso, pues es más beneficioso contar con un equipo

multidisciplinar durante todo el desarrollo, fomentando un modelo de trabajo maduro y completo. (Deemer et.al, 2009).

El equipo de trabajo decide implementar SCRUM, debido a que la solución requería de ser agilizada y presentada en un tiempo pertinente sin necesidad de afectar la calidad de esta durante su desarrollo, permitiendo que varios roles participaran en este proceso y generando que todo el equipo se adaptara de forma rápida, reflejándose en su comunicación e interacción continua y así mismo evidencia el uso y aplicación de herramientas tecnológicas que permitieran la innovación y mejora en esta solución, en comparación a las ya existentes en el mercado.

Roles. Teniendo en cuenta lo mencionado por Deemer et al. (2009), Scrum cuenta con 3 roles en específico que contribuyen al éxito y organización del proyecto. A continuación, se evidenciarán dichos roles, dando a conocer quien fue asignado al mismo dentro de la pasantía:

Tabla 1

Roles de Scrum.

Rol	Descripción	Encargado
Product owner	Identifica las funcionalidades y objetivos del proyecto y así mismo toma las decisiones en el mismo.	Vicepresidente técnico
Scrum Master	Brinda apoyo al equipo para aplicar la metodología correctamente, además busca ayudar a resolver los inconvenientes hallados en el	Ingeniero especialista

	transcurso del proyecto, buscando a su vez la implementación de las herramientas adecuadas.	
Equipo de desarrollo	Encargados de construir el producto o solución por medio del cumplimiento de las tareas halladas en cada sprint.	Ingeniero especialista Pasante

Nota. Fuente propia.

Fases. Para el correcto desarrollo del proyecto, la metodología SCRUM maneja ciclos que son definidos como sprints, los cuales tienen como finalidad establecer un ritmo productivo, contando con una serie de fases dentro de cada uno de ellos, que le permiten simplificar su uso y mejorar su organización (The Blokehead, 2016), siendo estas:

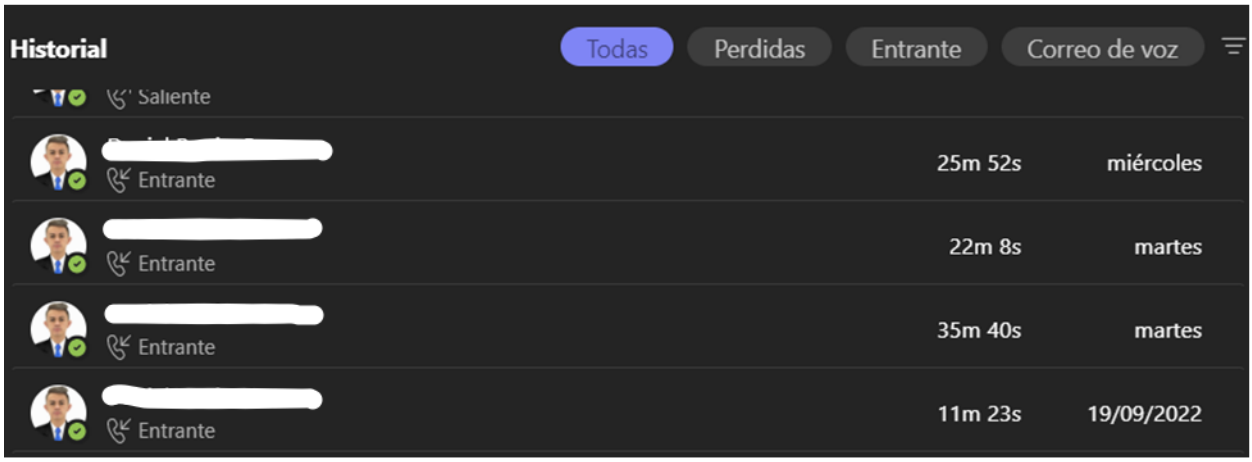
Planeación o planificación del Sprint. Se implementa por medio de una reunión, dirigida por el Scrum Máster, esta reunión tiene como objetivo planear y decidir que procesos se llevarán a cabo, de tal forma que pueda estimarse el tiempo de duración de cada uno de ellos y simultáneamente el del sprint, esto de acuerdo al esfuerzo de desarrollo. Para llevar a cabo esta fase, se implementa la herramienta Microsoft Teams para generar las reuniones pertinentes y Trello para asignar las tareas, estimar su tiempo y claramente listarlas.

Daily Scrum o Scrum Diario. Un Daily Scrum son reuniones breves entre el equipo de trabajo, la cual tiene como propósito evidenciar los avances y el rendimiento en cada una de las tareas, y así mismo solucionar problemáticas en caso de que se presenten. Para la

implementación de esta fase se utiliza Microsoft Teams tanto para las llamadas como para mensajes en caso de ser necesario.

Figura 1

Reuniones Diarias.



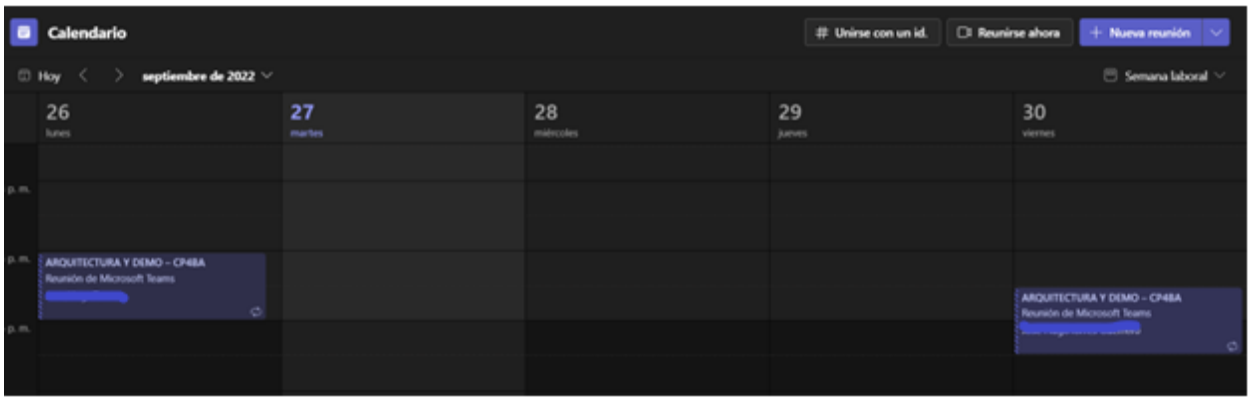
Nota. Fuente propia.

Revisión del sprint. Tiene como propósito realizarse al finalizar cada sprint, permitiendo evidenciar el producto desarrollado en ese tiempo, evaluarlo y dar una retroalimentación de cómo se realizó, esto para facilitar la aceptación o mejora de las tareas realizadas.

Retrospectiva del sprint. Es una reunión en la que todos los miembros del equipo están presentes y tiene como propósito identificar que tareas fueron concluidas y así mismo validar como pueden afectar o como se involucran con el siguiente sprint; mejorando de esta forma la entrega y el rendimiento de esta. A continuación, se evidenciarán algunas de las reuniones generadas para la revisión y retrospectiva de los sprints:

Figura 2

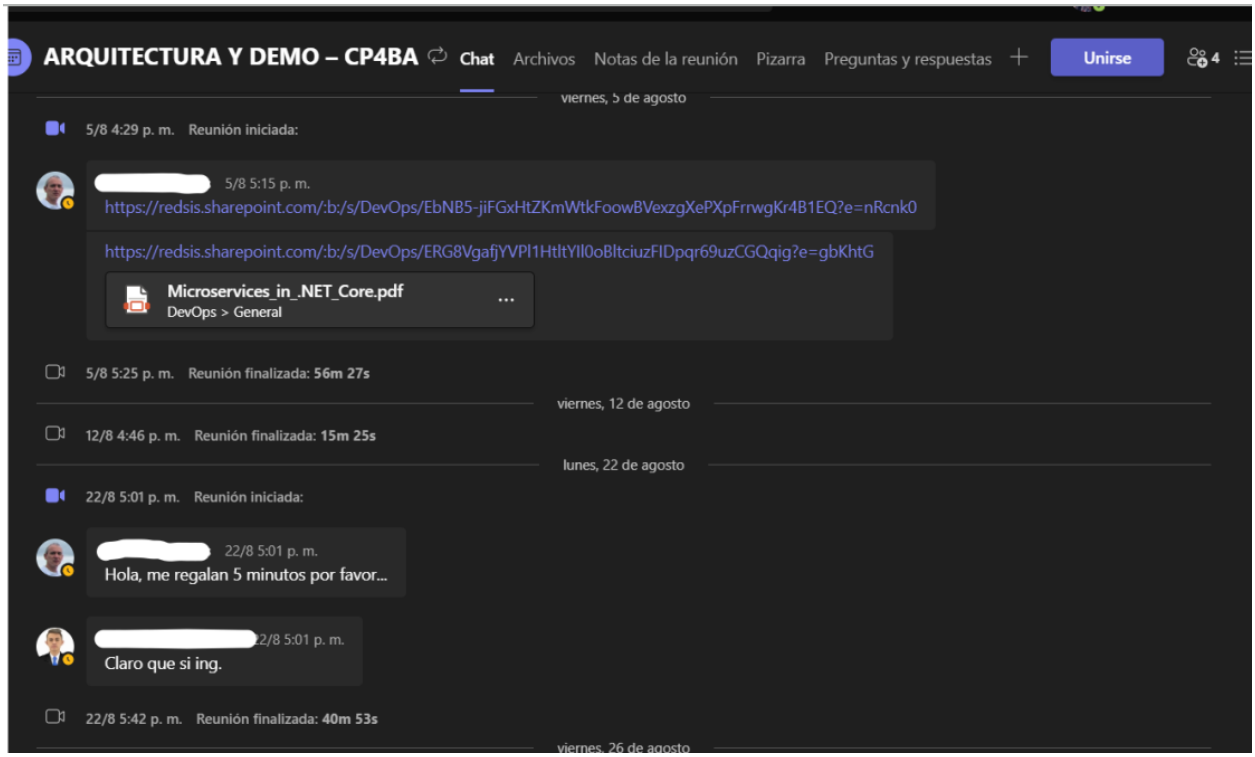
Agenda Reuniones Revisión y Retrospectiva.



Nota: Fuente propia.

Figura 3

Reuniones Revisión y Retrospectiva.



Nota: Fuente propia.

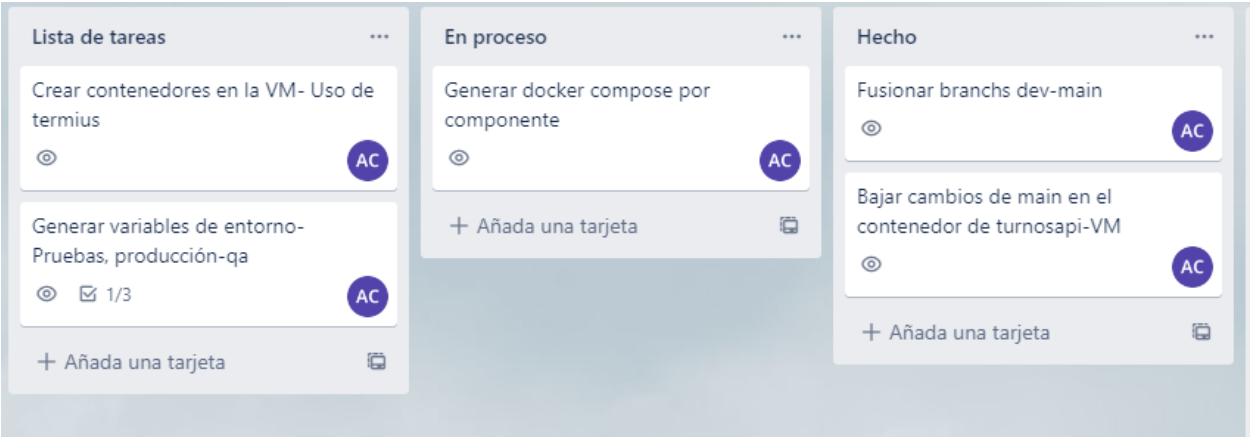
Artefactos de Scrum. Los artefactos empleados en Scrum son el reflejo del trabajo, el valor y la transparencia en el proyecto. Además, son adaptables y maximizan el enfoque dentro de la solución en desarrollo. (Schwaber et. al, 2020). Estos elementos son:

Product Backlog. Es una lista ordenada que contiene todos los requerimientos a ser abarcados para brindar un producto de calidad; su generación se realiza antes de dar inicio oficial al proyecto**.** Para la creación de este artefacto no estuvo presente el pasante, dado que cumplía el rol de miembro del equipo de desarrollo, por lo cual el personal de la empresa asignado a los roles restantes se encargó de este elemento.

Sprint Backlog. Es un plan destinado al equipo de desarrollo, donde se define y evidencia la lista de tareas que realizarán en el periodo de duración del sprint, dando a conocer a los encargados de cada tarea, de manera que tenga los suficientes detalles para validar su progreso y entender el objetivo del sprint. A continuación, se evidencia el desarrollo de uno de los Sprint Backlog, esto con el apoyo de la herramienta Trello.

Figura 4

Organización, Asignación y Detalles de Actividades en Trello.



Nota. Fuente propia.

6.4. Marco Conceptual

6.4.1. API REST

Tal como lo define IBM (2021), una API REST o interfaz de programación de aplicaciones, es un conjunto de reglas que se encargan de determinar como cualquier dispositivo o aplicación pueden conectarse y comunicarse entre sí. Cabe destacar que estas se comunican por medio de solicitudes HTTP.

6.4.2. Automatización

Según RedHat (2022), la automatización consiste en aplicar o utilizar la tecnología con la finalidad de realizar diversas tareas sin requerir en su totalidad de la intervención humana.

6.4.3. Backend

Es aquella capa que contiene la lógica de la aplicación y, a su vez es la capa de acceso a los datos, bien sea de un software o de cualquier otro dispositivo. (Platzi, 2018).

6.4.4. Base de Datos

Oracle (s.f) define que una base de datos es una “recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático.”

6.4.5. Clean Architecture

Tal como lo menciona Giordani (2022), la arquitectura limpia es aquella que busca que se entienda todo proceso realizado en el código, y a su vez se encarga de desacoplar los componentes en su mayoría, para de esta forma volverlos independientes.

6.4.6. CRUD

mdn web docs_ (2022) define al CRUD como el conjunto de formas en las que se pueden realizar operaciones sobre la información almacenada, dirigida mayormente en las bases de datos. Estas formas son respectivamente: Crear, leer, actualizar y borrar.

6.4.7. Frontend

De acuerdo con lo dicho por Platzi (2018), el Frontend es la parte "frontal" de todo sitio web, es decir, es todo lo que el usuario puede ver y todo con lo que puede interactuar.

6.4.8. Localhost

Según Cloud Center Andalucia (2021), es el servidor local del dispositivo en el cual se esté trabajando, siendo este capaz de comunicarse con su propia red y permitiendo emular conexiones de red dentro de él.

6.4.9. Máquina Virtual

Las máquinas virtuales son consideradas "equipos virtuales" definidos dentro de algún servidor, los cuales no son diferentes a cualquier equipo físico, pues este contiene los mismos componentes; pero no a nivel tangible, precisamente. (Microsoft Azure, s.f)

6.4.10. NoSQL

Según Acens (s.f), NoSQl son sistemas de información que no trabajan precisamente con datos estructurados, y, a su vez, brindar una mayor escalabilidad y flexibilidad.

6.4.11. Repositorio

Los repositorios son los que almacenan todo sistema de información, permitiendo la organización, preservación y la posibilidad de difundir todo aquel recurso encontrado dentro del mismo. (Duperet et al., 2015).

6.4.12. Sistema Web

De acuerdo con el Grupo Consultores efe (s.f), un sistema web es toda aquella aplicación a la que se pueda acceder por medio de internet, haciendo uso de navegadores.

6.5. Marco Tecnológico

Para el desarrollo del proyecto en contexto, se implementaron un conjunto de herramientas que permitirían tanto el cumplimiento, el correcto proceso y desarrollo en cada una de las actividades previstas, como también la interacción entre el equipo de trabajo. Es importante destacar que la gran mayoría de las aplicaciones y herramientas que serán mencionadas a continuación, fueron seleccionadas por el líder del proyecto, pues debían ser a la medida de la solución a desarrollar, mientras que, por otra parte, y para ayudas de documentación y facilidad de trabajo, el equipo optó por implementar algunas de ellas.

6.5.1. Azure

De acuerdo con la página oficial de Azure (s.f), es una plataforma con diversos servicios hallados en la nube, los cuales permitirán crear, ejecutar e incluso administrar aplicaciones propias. Esta plataforma se utilizó con la finalidad de optimizar el rendimiento e implementar contenedores en el proyecto, para ejecutar diversos componentes de forma global y así mismo tener un acceso a nivel grupal, únicamente.

6.5.2. Couchbase DB

Es una base de datos no relacional encontrada en la nube, caracterizada por su versatilidad, rendimiento, escalabilidad, etc (Couchbase, s.f). Esta base de datos fue seleccionada gracias a la adaptabilidad que brindaba frente a los requerimientos con los que contaba el proyecto, además se implementaba en la nube, facilitando de esta forma su ejecución e

implementación tanto en la herramienta de IBM Workflow Process Service como en las pruebas realizadas.

6.5.3. Docker

Es una plataforma que permite el despliegue de las aplicaciones de forma automática, sencilla y rápida, esto lo realiza por medio de contenedores que manejan todo lo necesario para que el código sea ejecutado en cualquier entorno (Amazon AWS, s.f). Esta plataforma se implementó para la contenerización de los endpoints o servicios realizados en el lenguaje Python, para poder consumirlos desde una máquina virtual. Por otra parte, se utilizó para el despliegue del trial de la herramienta de IBM, pues la forma óptima de instalarla era por medio de ella.

6.5.4. FastAPI

Es un marco web moderno para la creación de Apis en el lenguaje de Python (Fastapi, s.f). Este fue utilizado para la creación, ejecución e incluso pruebas de los endpoints realizados para que la herramienta pudiera consumirlos de forma correcta y sencilla.

6.5.6. GitLab

Es una plataforma que permite el control de versiones, el seguimiento del código y demás funcionales incorporadas en ella (GitLab, s.f). Esta herramienta fue utilizada al finalizar cada ciclo DevOps, para mantener el código correctamente versionado y completamente probado, con coverages mayores al 80%.

6.5.7. IBM WorkFlow Process Service

Es una herramienta que permite realizar flujos de trabajo, permitiendo dividir los componentes si se requiere trabajar de forma independiente, además cuenta con más funcionalidades integradas, como el manejo de coachs (para las interfaces gráficas), servicios,

entre otros (IBM, 2022). IBM Workflow process service es uno de los tantos componentes de los Cloud Paks, utilizado en este caso para la creación de interfaces gráficas y flujos de trabajo de componentes específicos, como lo son, por ejemplo: Módulo de ventanillas, módulo de administración, módulo de estadísticas, entre otros.

6.5.8. Microsoft Teams

Es una aplicación que permite colaborar, conectar, interactuar e incluso informar al equipo de cualquier novedad emitida, integrando varias funciones para que existan diversas formas de comunicación (Microsoft, s.f). Esta herramienta se utilizó para la comunicación entre los integrantes del equipo de trabajo y el líder del proyecto, para solucionar dudas, recibir las actividades y así mismo presentar los avances de cada proceso. Así mismo se utilizó para la interacción con los directores de la pasantía, con la finalidad de realizar las reuniones que permitieran evidenciar los avances en el desarrollo del proyecto.

6.5.9. Notion

Es un software de gestión de proyectos, caracterizado mayormente por permitir organizar tareas e incluso documentar lo que sea requerido, esto bajo filtros o páginas (Paez, 2021). Notion fue una elección de herramienta adicional utilizada con la finalidad de documentar algunos de los procesos que contenían comandos, procesos o referencias que pudieran ser utilizados por cualquier otro compañero o en un futuro, puesto que pueden mantenerse en la plataforma; pero también pueden ser exportados como un documento PDF.

6.5.10. Python

De acuerdo a la documentación oficial de Python (s.f), es un lenguaje de programación interpretado y orientado a objetos, capaz de incorporar diversos componentes y de admitir paradigmas de programación más allá del orientado a objetos. Este lenguaje se implementó para

la creación de los endpoints junto con FastApi, permitiendo versatilidad y fácil entendimiento en el código.

6.5.11. Swagger

Es una herramienta de software de código abierto enfocada en los servicios RESTful, la cual cuenta con muchas características y capacidades, entre ellas la documentación de cualquier API (Swagger, s.f). Se utilizó para documentar cada endpoint realizado con Python y FastApi, esto debido a que la herramienta de IBM requería del consumo de servicio por medio tanto del host, como de la información de la API; pero contaba con la exigencia de recibir esta información por medio de documentación realizada únicamente con Swagger.

6.5.12. Sophos SSL VPN Client

Es la herramienta que permite la conexión de acceso remoto brindando facilidad y rapidez (Sophos, 2012). Este cliente VPN se utilizó para la conexión a la red empresarial, pues la máquina virtual estaba ubicada en esta red, y para su uso era necesario conectarse.

6.5.13. Trello

Es una herramienta que permite gestionar proyectos por medio de la organización de tareas del equipo de trabajo, permitiendo organizarlas, supervisarlas, etc (Trello, s.f). Trello fue utilizada dentro del proyecto con la finalidad de organizar las tareas de cada integrante, para tener mayor claridad de sus actividades y así mismo el plazo que tenía para su realización.

6.5.14. Termius

Es un cliente SSH que funciona como terminal para diversos dispositivos, ofreciendo una interfaz moderna e incluso más interactiva (Termius, s.f). Esta herramienta se utilizó con la finalidad de poder acceder, con ayuda de Sophos SSL VPN Client a la máquina virtual,

realizando por medio de su terminal los procesos necesarios para la ejecución de contenedores, la creación de los mismos, entre otros procesos.

6.5.15. Unittest

Es uno de los más conocidos marcos de pruebas de Python, utilizado en el ciclo de pruebas unitarias del código, en este caso, perteneciente a los endpoints realizados en el lenguaje mencionado.

6.5.16. Visual Studio Code

Es el editor utilizado para la programación de cada uno de los endpoints, como también para la creación y ejecución de las pruebas de los mismos.

6.5.17. Webex

Es una herramienta enfocada en las conferencias web, reuniones y toda aquella actividad que abarque la comunicación e interacción entre usuarios. (Webex, s.f). En este caso, Webex fue utilizado para las reuniones y webinars requeridos en el concurso de IBM Cloud Rocks, pues los líderes del programa optaron por implementar esta herramienta durante este proceso en el cual se participó con el presente proyecto.

7.Empresa

7.1. ¿Quiénes son?

Redsis -Redes y Sistemas Integrados S.A.S es una empresa global caracterizada por su enfoque tecnológico y por su experiencia en el mercado durante más de 20 años, centrándose en servicios como: Diseño, suministro, instalación y el soporte de cada una de las soluciones que manejan actualmente y claramente las futuras (Redes y sistemas integrados S.A.S, s.f).

7.2. Misión

La misión de Redsis se centra en “Transformar informática con un equipo humano confiable” (Redes y sistemas integrados S.A.S, s.f).

7.3.Visión

La visión formulada por la empresa busca “Consolidar su liderazgo y crecer en servicios de Tecnología informática en Latinoamérica” (Redes y sistemas integrados S.A.S, s.f).

7.4.Valores

De acuerdo a su página oficial, (Redes y sistemas integrados S.A.S, s.f) cuenta con los siguientes valores empresariales:

- Aprendizaje continuo
- Pasión por el servicio
- Coherencia
- Respeto por la diferencia
- Integridad
- Sincronía

8. Actividades Realizadas

Durante el desarrollo del proyecto se implementó el marco de trabajo SCRUM, el cual permitió abarcar cada proceso necesario durante el trayecto, priorizar la colaboración entre el equipo, organizar las tareas de acuerdo a los avances realizados por cada integrante, agilizar tareas por medio del uso de herramientas debidamente seleccionadas y demás actividades que fueron abarcadas por medio de cada una de las fases, brindando de esta forma una cultura de trabajo apta para el alcance del desarrollo en cuestión. Por otra parte, se aclara que este apartado es el que más abarcará la confidencialidad frente a los procesos y sus explicaciones, debido al motivo mencionado con anterioridad en la sección de alcance y limitaciones.

Para llevar a cabo el desarrollo de cada una de las actividades en cuestión, se toma en cuenta un Product Backlog brindado por la empresa, contando con sprints definidos, estos no siendo mayores a 3 semanas cada uno de ellos, para de esta forma aplicar correctamente la metodología y así mismo mantener un control de tiempos. Cabe destacar que este artefacto se encontrará como anexo.

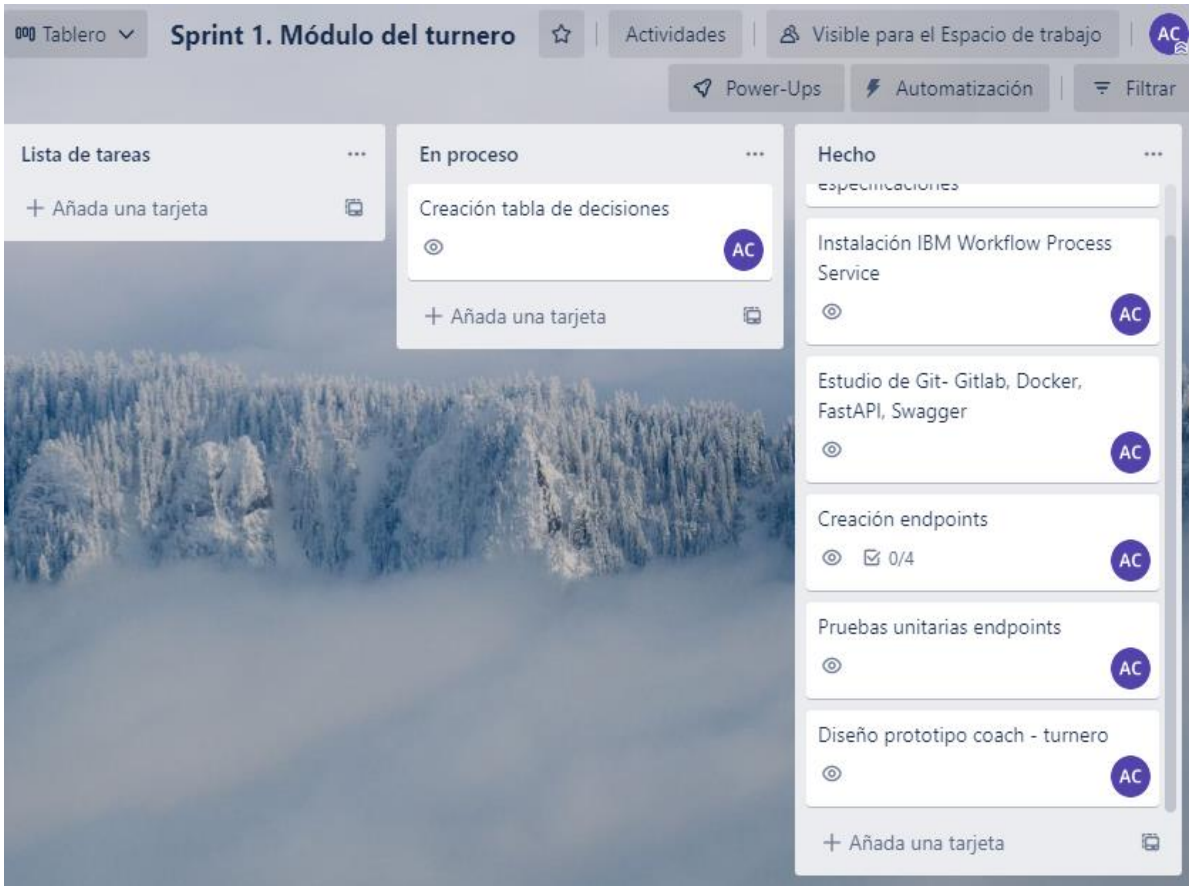
Ahora bien, se decide dar a conocer las actividades realizadas clasificadas por sprints, para demostrar el avance progresivo y así mismo describir el proceso de la forma más organizada y transparente posible.

8.1. Sprint 1. Módulo Del Turnero

Para este sprint, se generó un enfoque en la creación del módulo denominado como turnero, siendo este el componente que le permite al usuario solicitar un turno en cualquier sala de una entidad promotora de salud. Para iniciar este sprint se estableció una reunión donde se definieron las siguientes tareas evidenciadas en el Sprint Backlog:

Figura 5

Sprint Backlog: Primer Sprint.



Nota: Fuente propia.

Estas tareas se centraban en el desarrollo de endpoints para el servicio API REST, generar un diseño adecuado para las interfaces de usuario y validar la prioridad de los usuarios por medio de reglas de negocio para finalmente dar una respuesta al mismo respecto a su turno. Para llevar un mejor control y organización de las tareas a realizar, se decidió abarcarlas por etapas, dándose a conocer cada una de ellas a continuación:

8.1.1. Análisis

Al ser el primer sprint del proyecto en cuestión, fue necesario investigar acerca de las herramientas que serían implementadas para llevar a cabo el desarrollo, brindando de esta

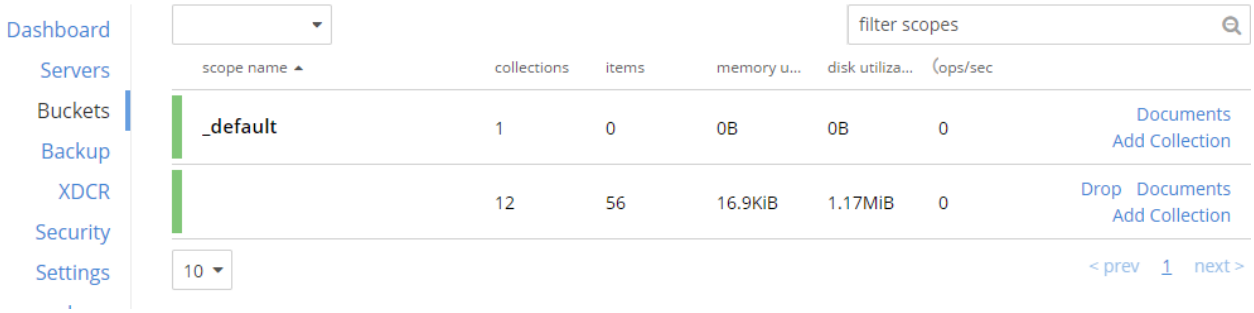
manera los conocimientos básicos para un mejor ritmo productivo, estas herramientas son respectivamente: Python, fastAPI, IBM Cloud Paks For Business Automation enfocado en el módulo IBM Workflow Process Services, CouchbaseDB, Swagger, entre otras. De igual forma, se cuenta con una breve contextualización acerca de las posibles prioridades que podría tener un usuario a la hora de solicitar un turno, aclarando que se abarcaron las más generales. Ahora bien, se define que se implementará en el código “Clean Architecture”, esto con la finalidad de promover el desacoplamiento de los elementos en caso de que existan cambios o reestructuración debido a una necesidad futura.

8.1.2. Diseño

Una vez realizado el análisis de las posibles reglas de negocio (prioridades) se definió la estructura en la base de datos documental, obteniendo como resultado:

Figura 6

Evidencia de Estructura en la Base de Datos.



Nota: Fuente propia.

Figura 7

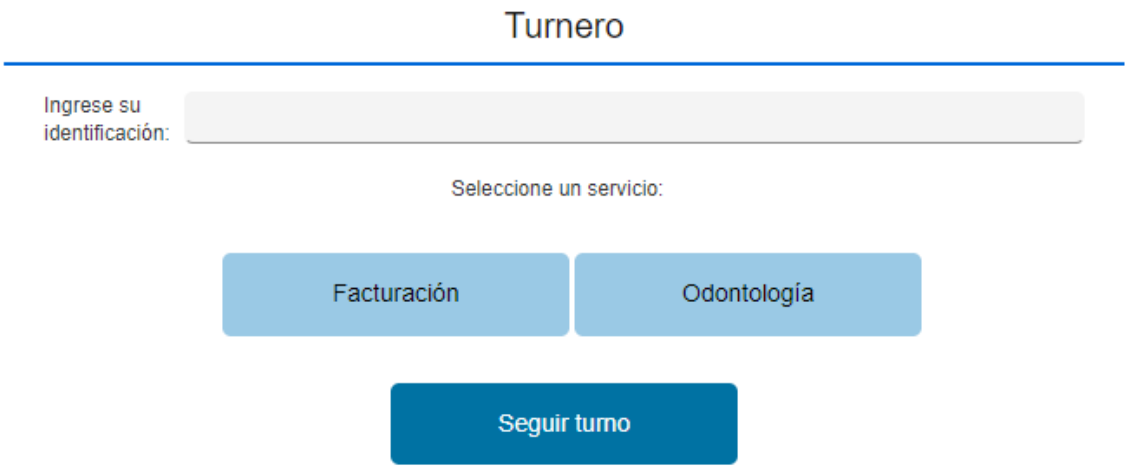
Estructura de Datos Para la Colección de Turnos.

Nota: Fuente propia.

Así mismo, teniendo en cuenta el caso hipotético de la solicitud de un turno por parte de un paciente, se diseña un prototipo para el front-end, el cual se realizó en la herramienta de IBM, enfatizando en la implementación de un coach, que es el componente que funciona como GUI dentro de ella; este coach tiene 3 secciones distintas, pues se consideraba necesario que el usuario pudiera visualizar que servicios estaban disponibles, también debería de elegir una opción frente a la priorización del turno, y finalmente obtener y visualizar un ticket con la información del turno generado.

Figura 8

Prototipo del Inicio del Turnero.



Nota: Fuente propia.

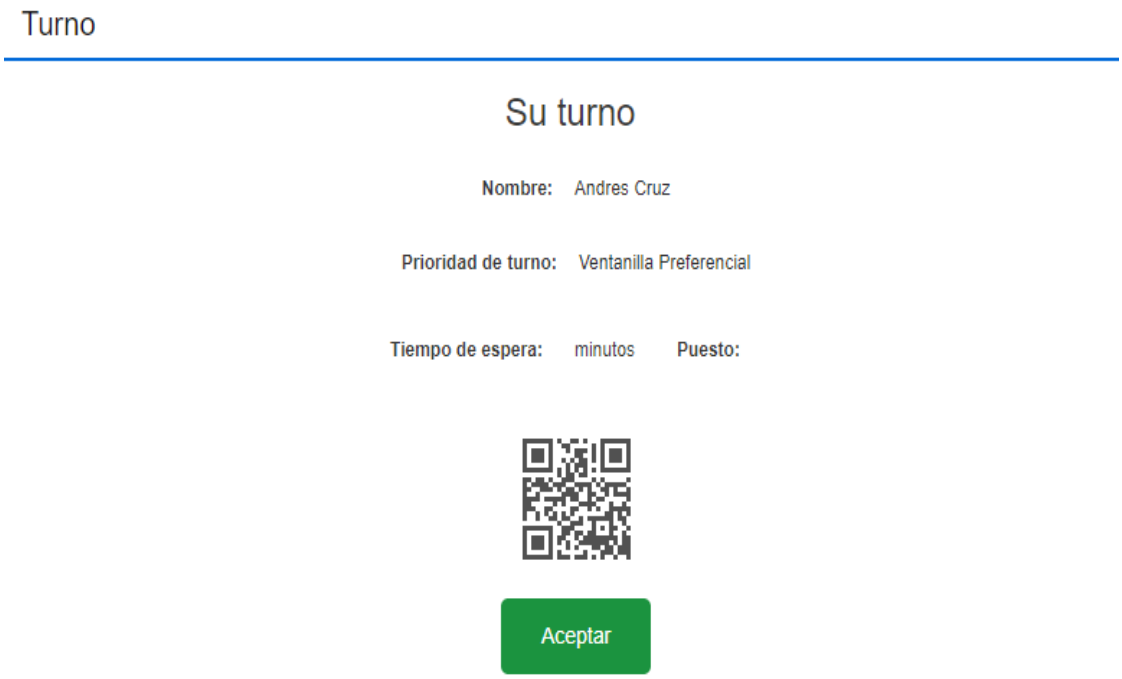
Figura 9

Prototipo de Las Opciones de Atención por Turno.

Nota: Fuente propia.

Figura 10

Prototipo del Ticket por Turno.



Nota: Fuente propia.

8.1.3. Desarrollo y Pruebas

En esta etapa se inicia con la instalación de todas las herramientas necesarias, como también con la creación del bucket, el scope y la colección de la base de datos, denominada como “turnos”, basándose en la estructura planteada previamente. Por otra parte, en la programación se realiza la respectiva instalación del SDK de CouchDB para Python, esto con la finalidad de utilizar funciones nativas de esta base de datos, como también para generar la conexión necesaria entre los componentes. Así mismo se crearon 4 endpoints en total, teniendo en cuenta la funcionalidad inicial de este módulo. Es importante aclarar que para llevar a cabo cada endpoint, a nivel de backend se generaron entidades y diversos casos de uso que contienen una función específica (Se encargan de un proceso en específico debido a la responsabilidad única, basada en los principios SOLID) para lograr plasmar un resultado en cada uno de estos servicios REST, aplicando la arquitectura limpia para Python.

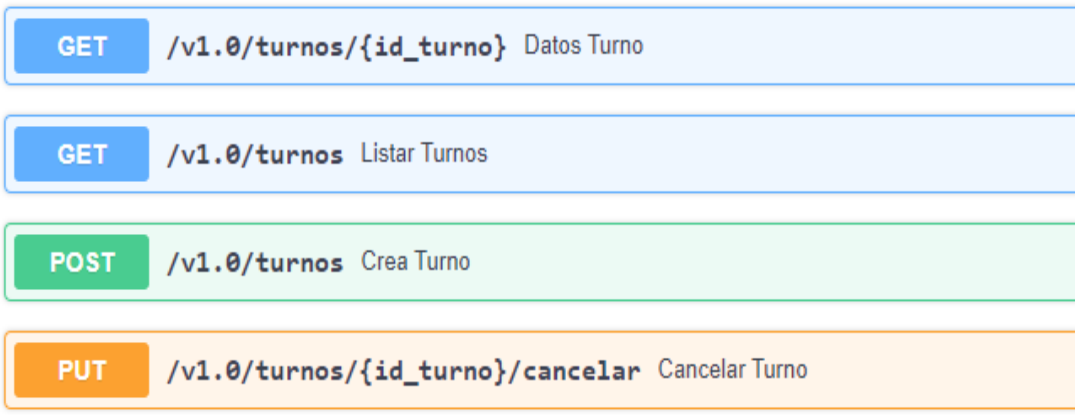
Figura 11

Bucket, Scope y Colección en Couchbase.

Nota: Fuente propia.

Figura 12

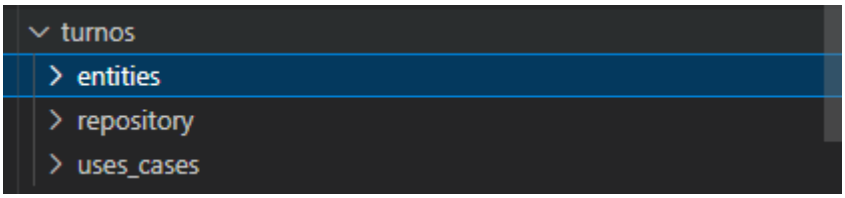
Endpoints Sprint 1.



Nota: Fuente propia.

Figura 13

Evidencia Arquitectura a Nivel de Backend.

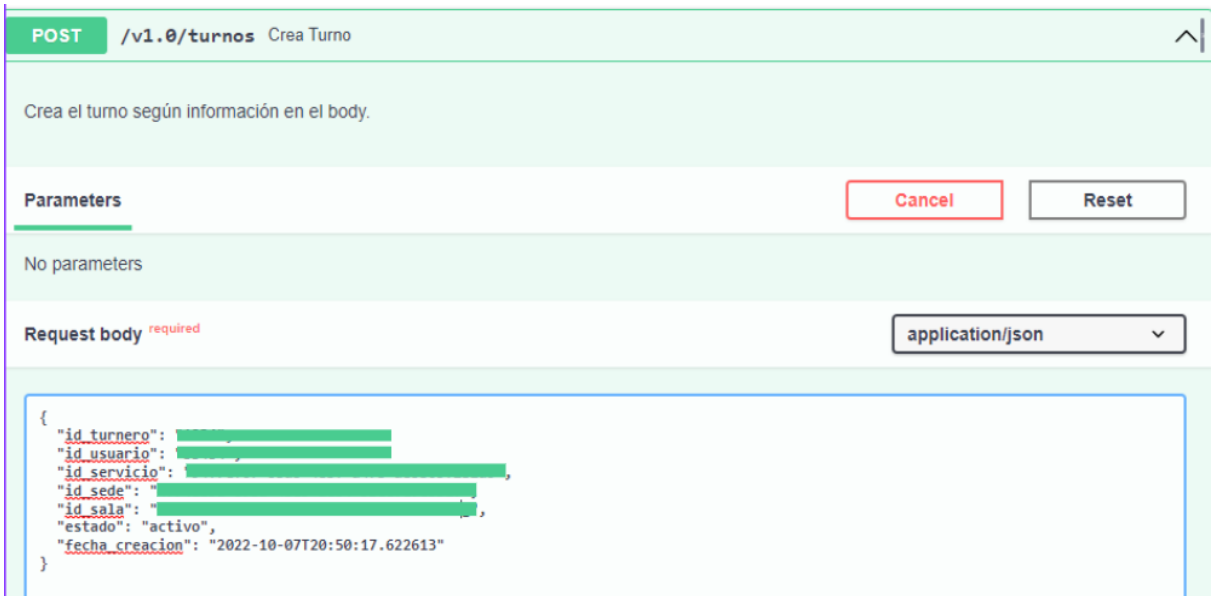


Nota: Fuente propia.

Una vez realizados estos endpoints, se generan pruebas unitarias en cada uno de ellos, de tal forma que se valide su funcionamiento y su captura de errores. En este caso se dará a conocer la prueba de la creación de un turno, donde se verá reflejado su caso de éxito por medio de la respuesta HTTP, retornando un mensaje definido, el cual, en este caso es el turno asignado al usuario de acuerdo con la selección del servicio requerido y a su prioridad.

Figura 14

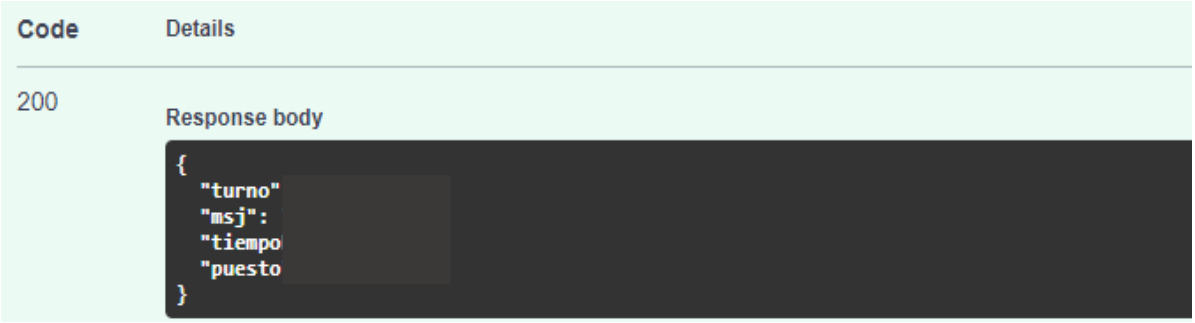
Prueba Unitaria Sprint 1.



Nota: Fuente propia.

Figura 15

Resultado Prueba Unitaria Sprint 1.



Nota: Fuente propia.

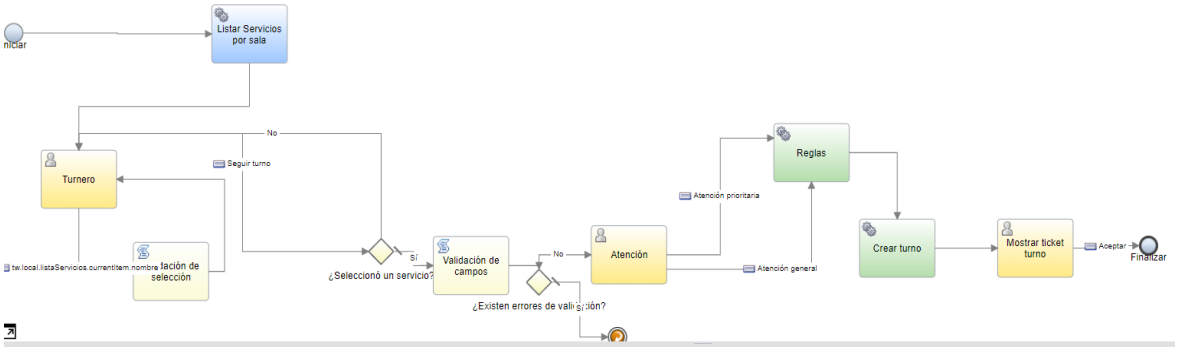
Para dar como resultado estos servicios REST se crearon varios casos de uso en el código, que permitían, por ejemplo, traer la información del último puesto que fue atendido de acuerdo al servicio que seleccionó el usuario, para suministrar este dato en su ticket, o también el caso de uso encargado de agregar el turno junto con todos sus datos en una colección específica en la base de datos.

Una vez fue validado el funcionamiento de los endpoints, se estableció el consumo de los mismos gracias a la documentación realizada en Swagger, siendo esta una de las herramientas

compatibles con el componente de automatización de IBM, luego de ello se realizó el flujo de trabajo necesario para el funcionamiento del módulo dentro de IBM Workflow Process Services, incluyendo, además del coach implementado en la fase de diseño, las tareas de servicio, que permitían consumir los endpoints realizados, tablas de decisiones que permitían implementar reglas de negocio y brindar una respuesta de acuerdo a su configuración y así mismo componentes de decisión para el camino que debía tomar el flujo de acuerdo a la validación.

Figura 16

Workflow Módulo Turnero.



Nota: Fuente propia.

Figura 17

Tablas de Decisión para las Reglas de Negocio.

Nota: Fuente propia.

Figura 18

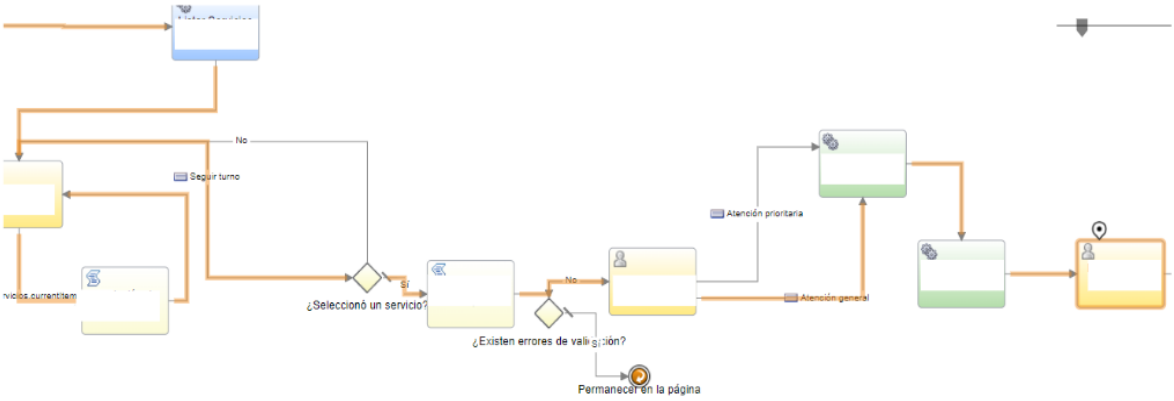
Evidencia Correlación de Datos.

Nota: Fuente propia.

Finalmente, para validar la funcionalidad del sprint, se realizan pruebas de integración en el módulo. Una de las pruebas de integración realizadas se encargaba de ejecutar el módulo en modo debug, digitar un número de cédula, seleccionar un servicio y posteriormente validar que camino fue seleccionado de acuerdo con las reglas de negocio.

Figura 19

Evidencia Prueba de Integración Sprint 1.



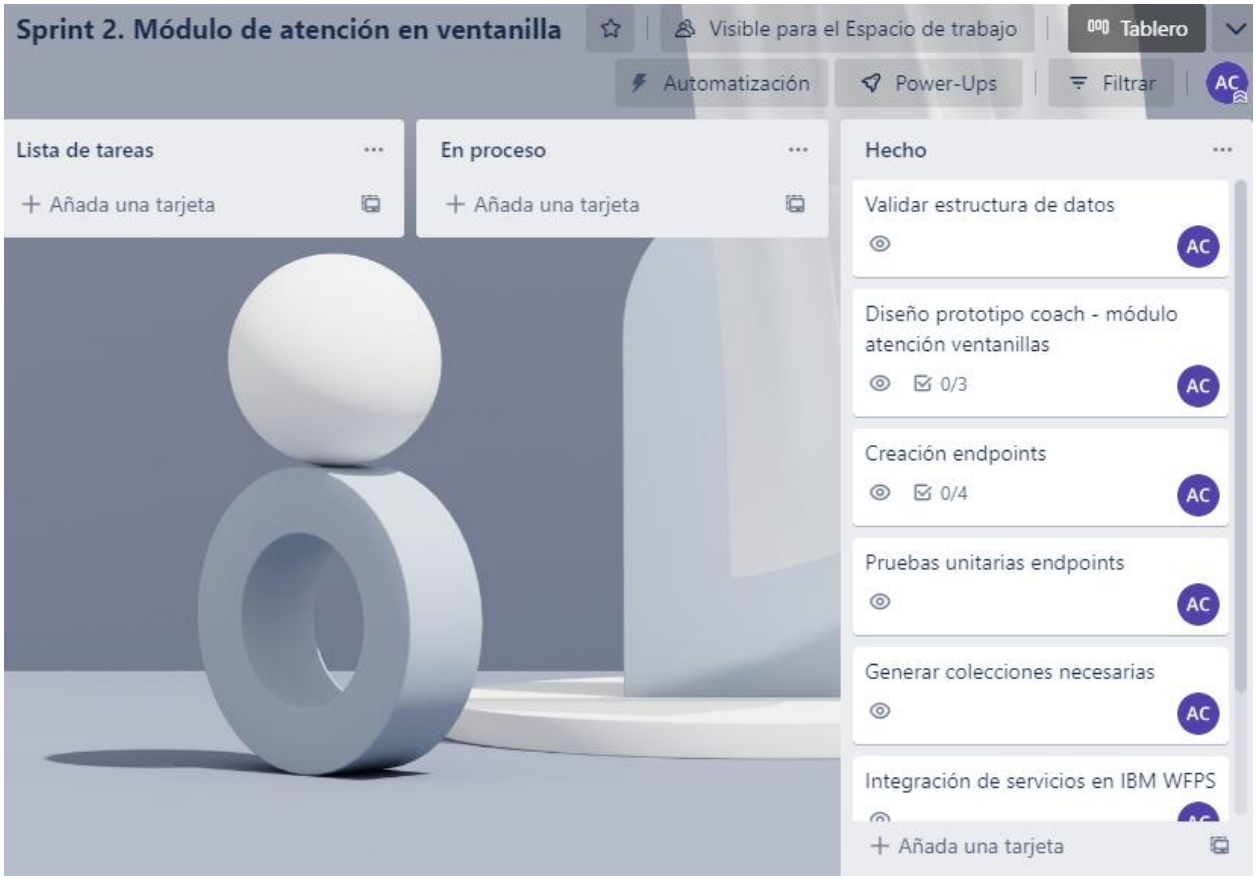
Nota: Fuente propia.

8.2. Sprint 2. Módulo de Atención en Ventanilla

Este sprint se centra en la creación de un módulo dirigido al operador del servicio, bien sea, el personal médico o el personal encargado de atender servicios específicos en las ventanillas, de tal forma que pudiera seguir el proceso de atención de su turno por medio de este. Se muestran algunas de las tareas halladas en el Sprint Backlog a continuación:

Figura 20

Sprint Backlog 2.



Nota: Fuente propia.

8.2.1. Análisis

Para llevar a cabo las tareas de este sprint, fue necesario analizar la posible estructura que podría tener la entidad de ventanillas, asumiendo que progresivamente podía atender turnos, mantener una ocupación temporal para la atención de un turno en específico, suspender su servicio, cerrar la ventanilla, validar si el turno estuvo presente o ausente, entre otras funcionalidades.

8.2.2. Diseño

De acuerdo con el análisis planteado, se optó por seguir la siguiente estructura de datos:

Figura 21

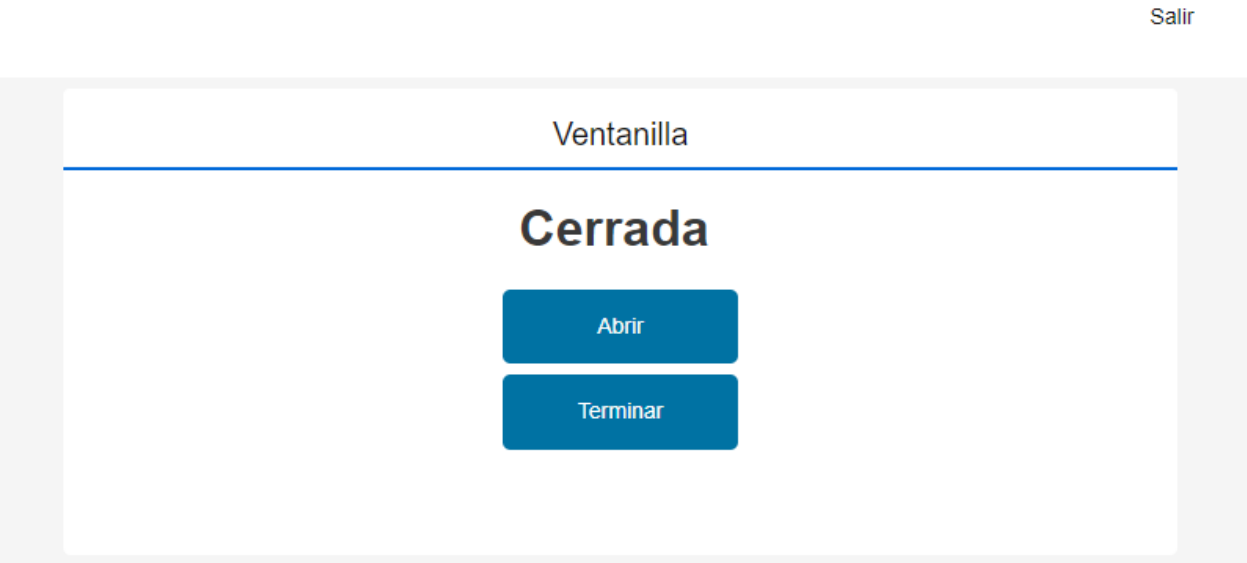
Estructura de Datos Para la Colección de Ventanillas.

Nota: Fuente propia.

Una vez definida la estructura, se generó un enfoque a nivel de front-end, validando las opciones que podría tener el operador del servicio al momento de llevar a cabo su labor, posteriormente se diseñó un prototipo que contenía 4 coaches, considerados como aptos para la funcionalidad que se pretendía dar al módulo.

Figura 22

Prototipo de Inicialización de Ventanilla.



Nota: Fuente propia.

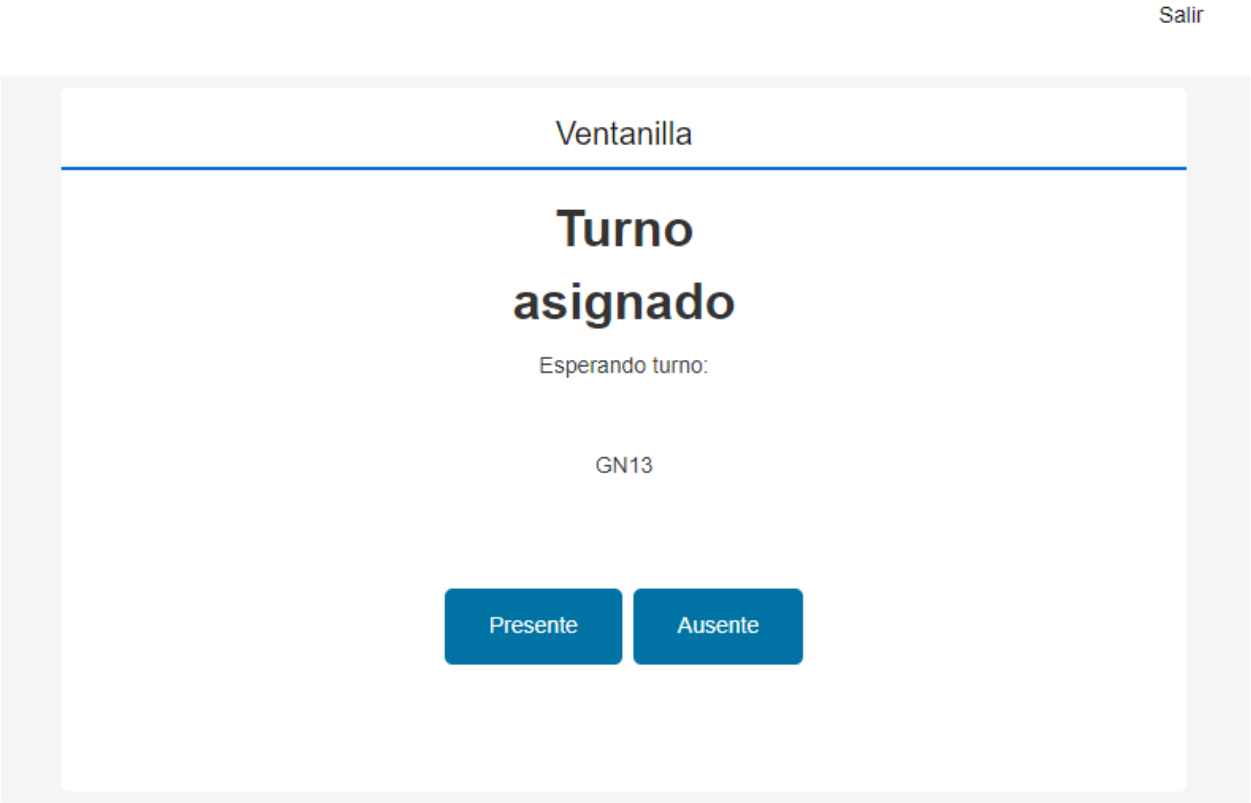
Figura 23

Prototipo de Disponibilidad de Ventanilla.

Nota: Fuente propia.

Figura 24

Prototipo de Ventanilla Ocupada.



Nota: Fuente propia.

Figura 25

Prototipo de Turno Ausente.

Nota: Fuente propia.

Figura 26

Prototipo de Turnos No Disponibles.

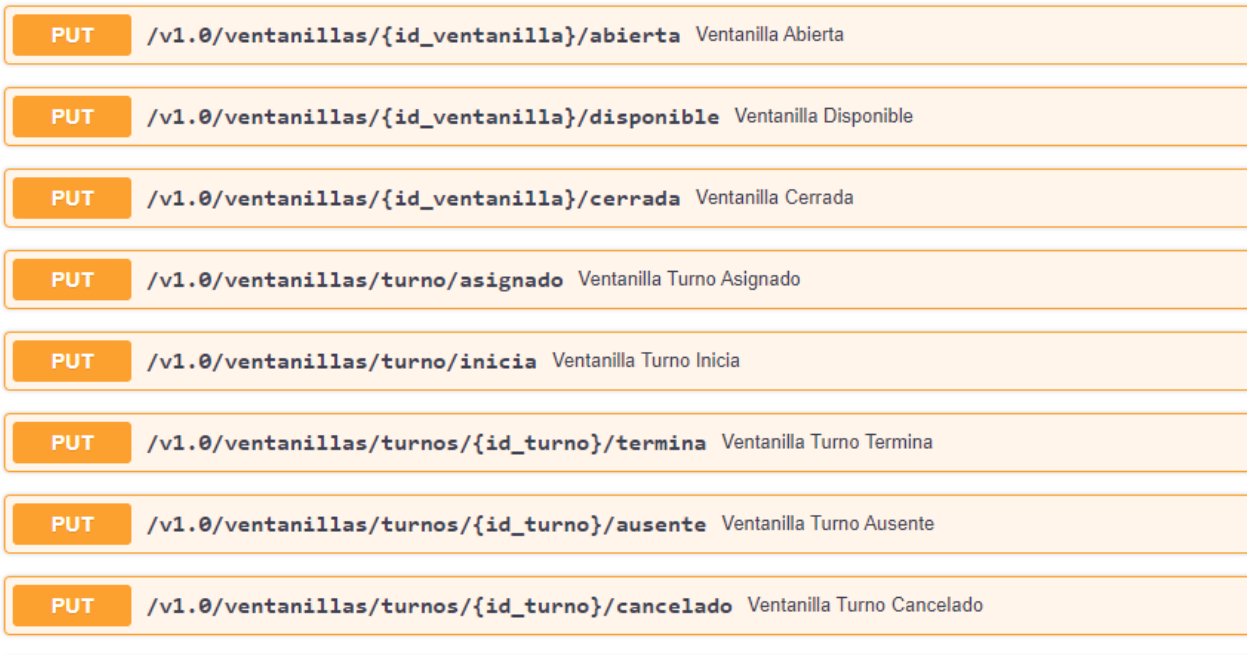
Nota: Fuente propia.

8.2.3. Desarrollo y Pruebas

En primer lugar, se creó la colección para la base de datos, denominada: “Ventanillas”. Para implementar los prototipos realizados en la etapa de diseño, fue necesario crear un total de 8 endpoints, cada uno de ellos con funciones específicas:

Figura 27

Endpoints Sprint 2.



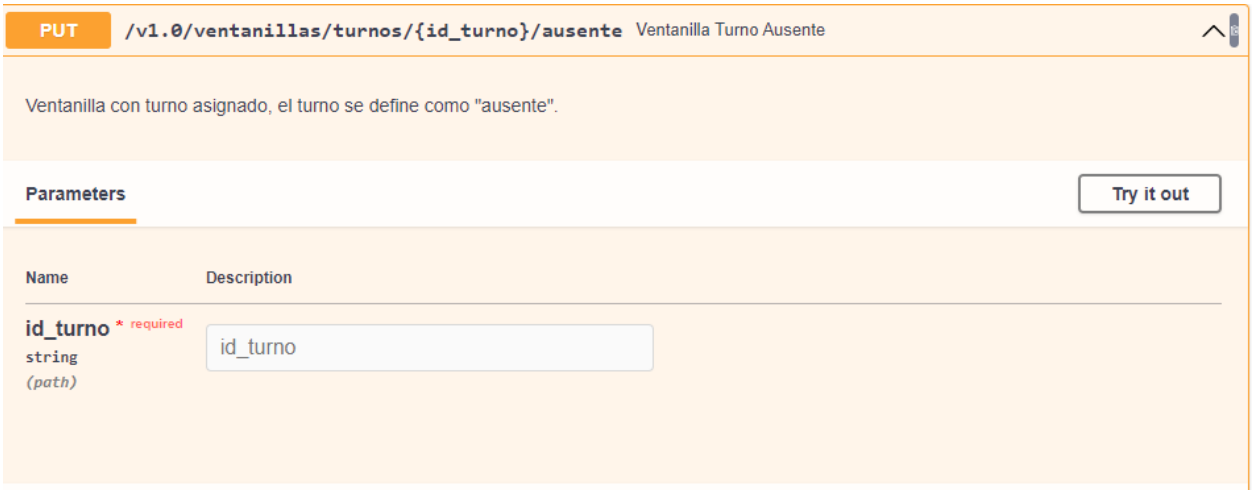
Nota: Fuente propia.

Algunos de los casos de uso implementados a nivel de backend se centraban por ejemplo, en contar el tiempo exacto que tardaba el operador de la ventanilla atendiendo a un usuario, para posteriormente almacenar esta información en otra colección denominada “AtencionTurnosVentanilla”, que permitía promediar este tiempo para otros procesos y almacenar el último turno atendido. También se crearon casos de uso para actualizar el estado de la ventanilla de acuerdo a su avance en la atención del turno, pues podría estar ocupada, disponible, abierta o cerrada.

Luego de ello se implementaron pruebas unitarias para cada uno de los endpoints, reflejando así la continuación a la siguiente tarea. Una de estas pruebas se centraba en cambiar el estado tanto del turno como de la ventanilla en caso de que el usuario estuviera ausente al momento de ser llamado para la prestación del servicio, la respuesta en este caso era un response estandarizado, enviando su mensaje por medio de un string.

Figura 28

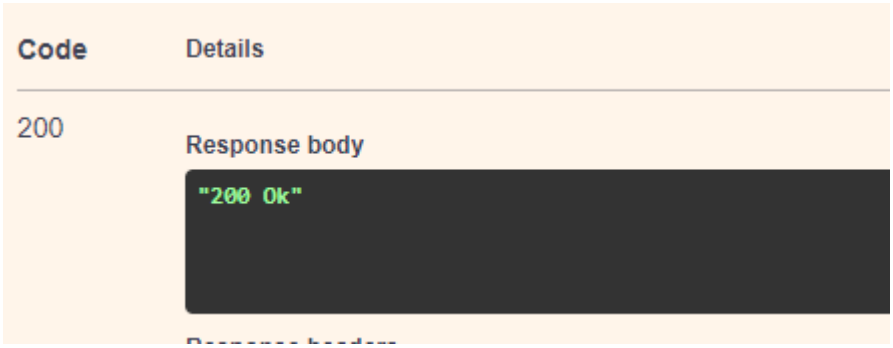
Evidencia Prueba Unitaria Sprint 2.



Nota: Fuente propia.

Figura 29

Response Prueba Unitaria Sprint 2.

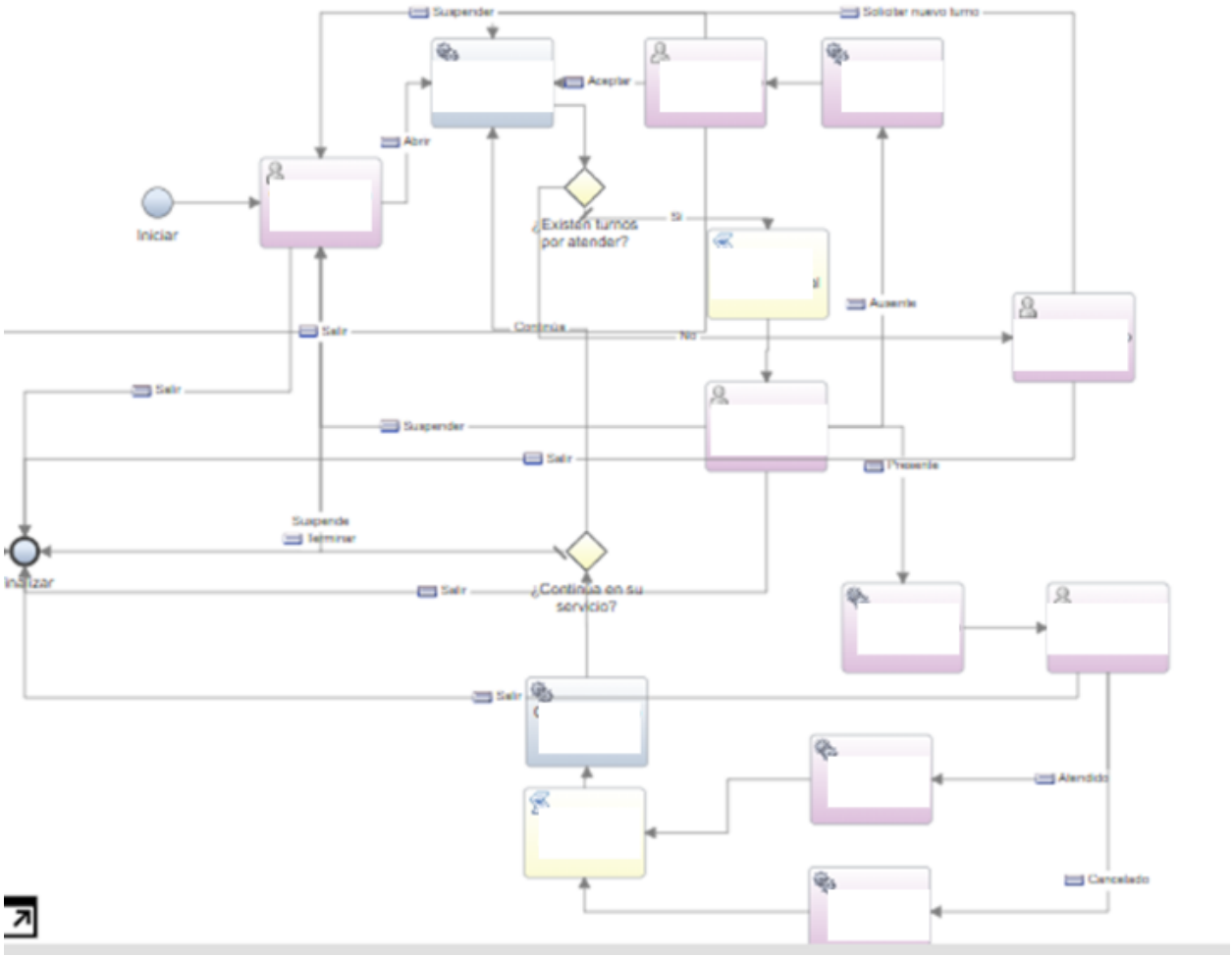


Nota: Fuente propia.

Las siguientes tareas se encaminaron al desarrollo dentro de la herramienta, creando un flujo de trabajo para el correcto consumo de los servicios (ya debidamente documentados en Swagger), así como las decisiones implícitas al momento de unificar los coaches con las tareas de servicio y los scripts.

Figura 30

Workflow Sprint 2.



Nota: Fuente propia.

Figura 31

Evidencia Correlación de Datos en Una Tarea de Servicio Sprint 2.

Nota: Fuente propia

Figura 32

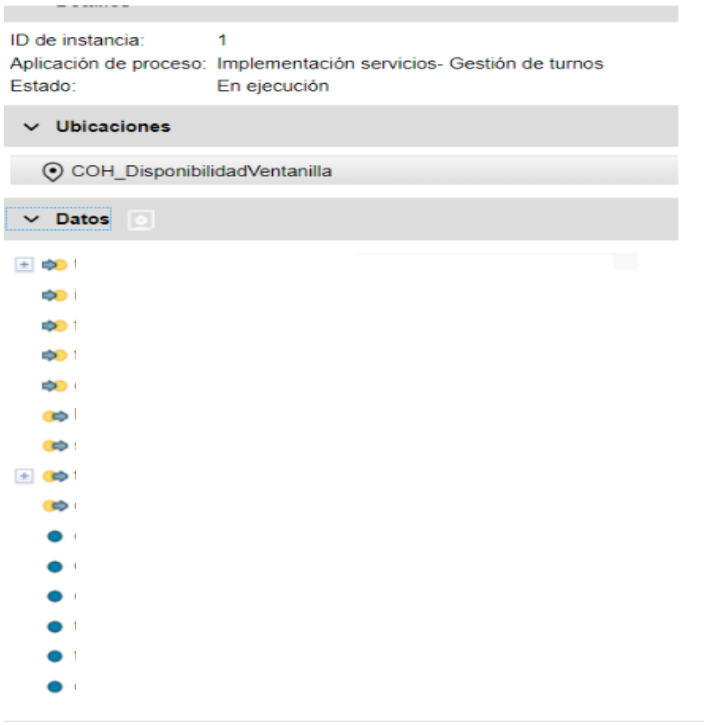
Evidencia Documentación Swagger Sprint 2.

Nota: Fuente propia

Finalmente se realizaron pruebas de integración para todos los componentes mencionados anteriormente, concluyendo así el sprint. Algunas de las pruebas se relacionaban con situaciones tales como: Validar el correcto envío de la información hallada en las variables definidas dentro de la herramienta a los endpoints establecidos, validación de la recepción de información por parte de la base de datos, validación de campos, correlación adecuada entre las variables con las tareas y flujos de servicio, entre otras.

Figura 33

Prueba de Integración Sprint 2.



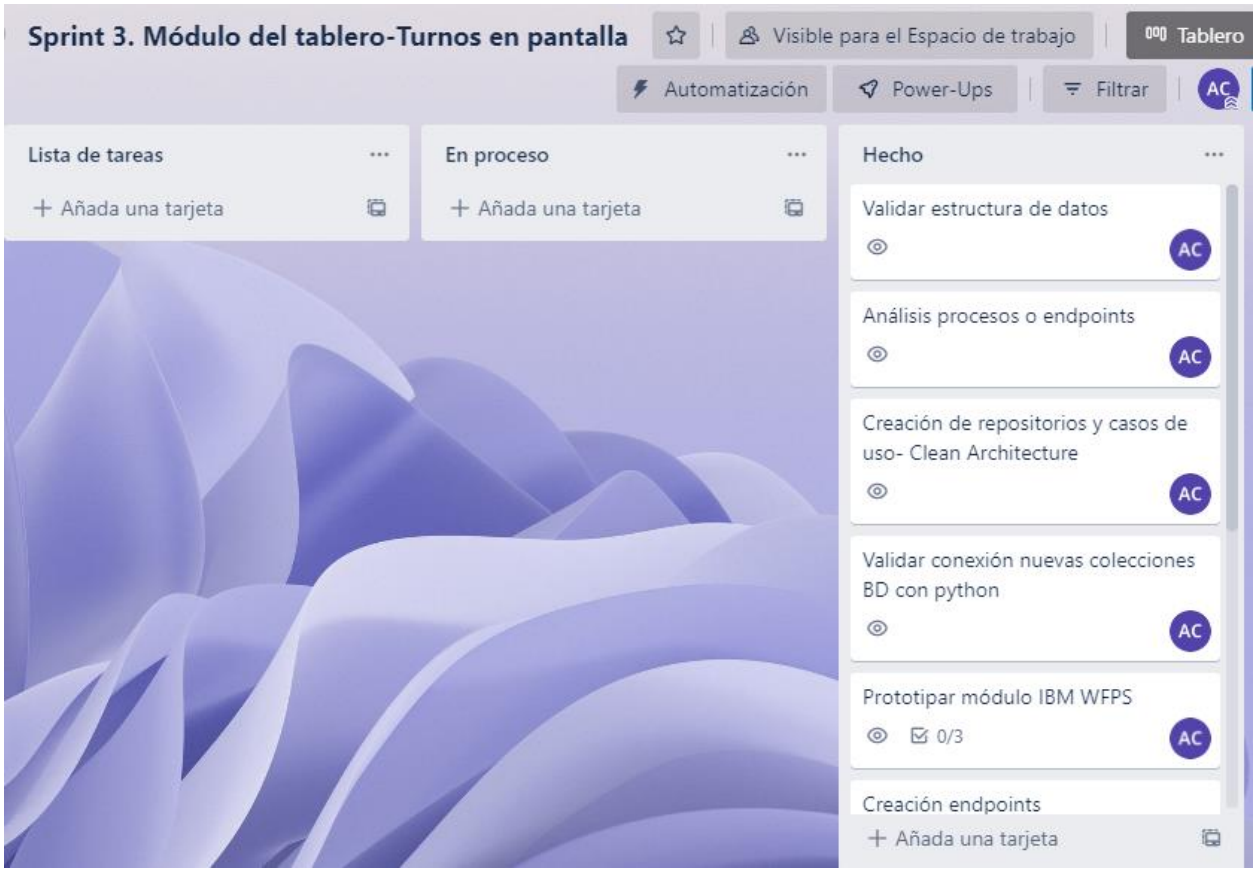
Nota: Fuente propia

8.3. Sprint 3. Módulo del Tablero-Turnos en Pantalla

Luego de abarcar 2 de los ejes principales del sistema de turnos, se buscó en este sprint el desarrollo de un módulo que hiciera énfasis en la funcionalidad que tiene toda entidad promotora de salud en sus sedes: Monitores que den a conocer los turnos debidamente clasificados, para que los usuarios estén al tanto del proceso progresivo frente a su atención. Para contextualizar de mejor manera, se da a conocer el Sprint Backlog 3:

Figura 34

Sprint Backlog 3.



Nota: Fuente propia

8.3.1. Análisis

Teniendo en cuenta lo que buscaba este sprint, se realizó un análisis de cómo pueden verse estructurados los datos para cada cola de servicio, de forma que fuera claro, organizado y tuviera en cuenta cómo debía presentarse la información por medio de una interfaz al usuario. Se optó por categorizar las colas de servicios por: Turnos próximos, turnos llamando y turnos asignados (atendiendo).

8.3.2. Diseño

Siguiendo el planteamiento del análisis, se inició esta etapa con la estructura formal de los datos para las futuras colecciones a crear en Couchbase DB:

Figura 35

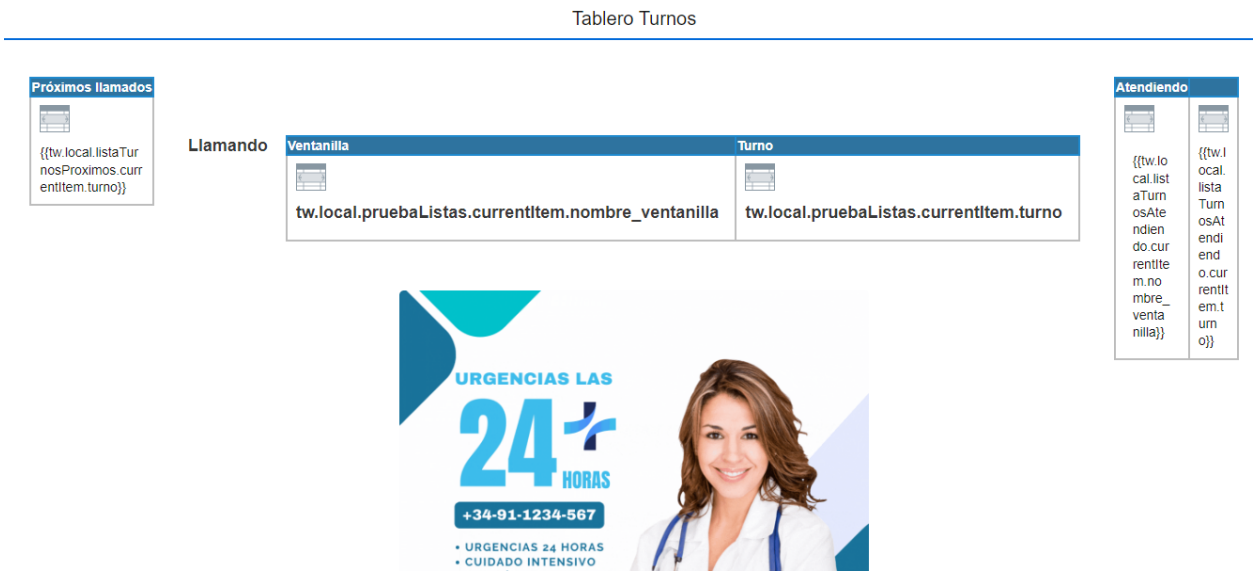
Estructura de Datos Sprint 3.

Nota: Fuente propia

En cuanto a la parte visual, se observó que se requería un único coach con una única sección, pues todos los turnos podrían verse en una única pantalla, siempre y cuando se organizara y clasificara correctamente, también se buscaba que esta interfaz tuviera la posibilidad de mostrar un vídeo o imágenes que estuvieran relacionadas con la entidad, por lo que se obtuvo el siguiente prototipo:

Figura 36

Prototipo Turnos en Pantalla.



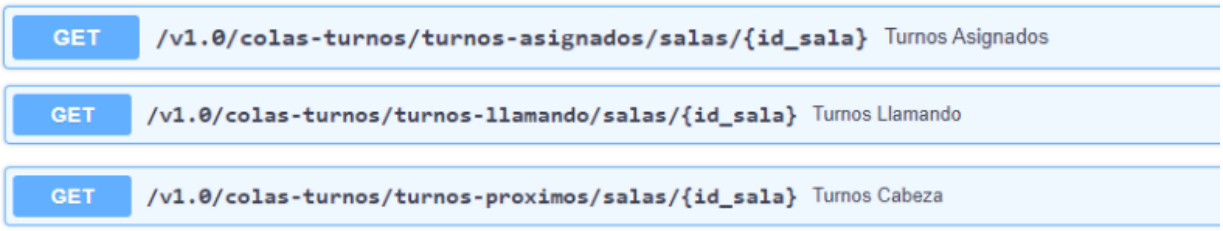
Nota: Fuente propia

8.3.3. Desarrollo y Pruebas

De acuerdo con la estructura de los datos y el prototipo generado para el front-end, se estableció que se debían programar endpoints clasificados por el tipo de cola de servicio, siendo los más importantes aquellos que listaban dichos turnos que contenían en el momento, teniendo en cuenta que toda información que fuera manipulada en ellos debía tenerse almacenada en una colección de la base de datos, por lo cual se generan 3 colecciones denominadas: Turnos próximos, turnos llamando y turnos asignados; posteriormente fueron generados 3 servicios REST:

Figura 37

Endpoints Sprint 3.



Nota: Fuente propia

Los casos de uso para estos endpoints se centraban únicamente en listar la información encontrada en cada colección a la que fueran dirigidos por el proceso programado, esto organizado a nivel ascendente por su fecha de creación.

El paso por seguir fue validar que cada uno de ellos retornara la información adecuada, siendo, en uno de los casos, un array que tuviera todos los turnos con su respectiva información, la cual, se obtenía de la colección creada en la base de datos.

Figura 38

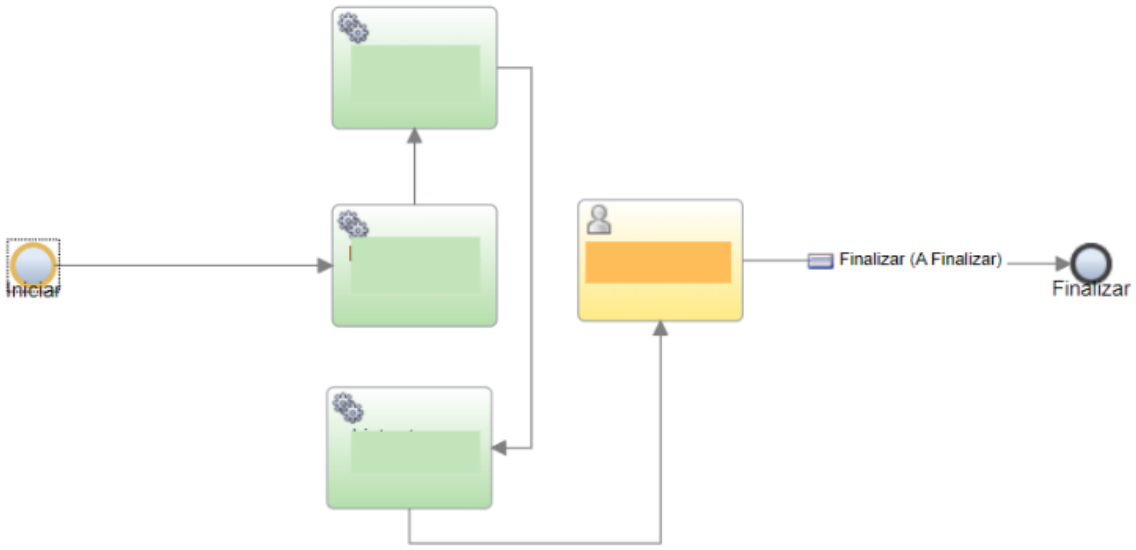
Prueba Unitaria Sprint 3.

Nota: Fuente propia

Luego de ello, se estableció un flujo de trabajo con los servicios ya montados en la herramienta, de tal forma que le permitiera suministrar la información necesaria al coach para que pudiera brindar los turnos en un monitor, o en donde fuera requerido. Para este proceso se requirió de la definición de variables de entrada, salida y privadas en el IBM Workflow, al igual que en los sprints anteriores, pues de esta forma se recibe y procesa la información.

Figura 39

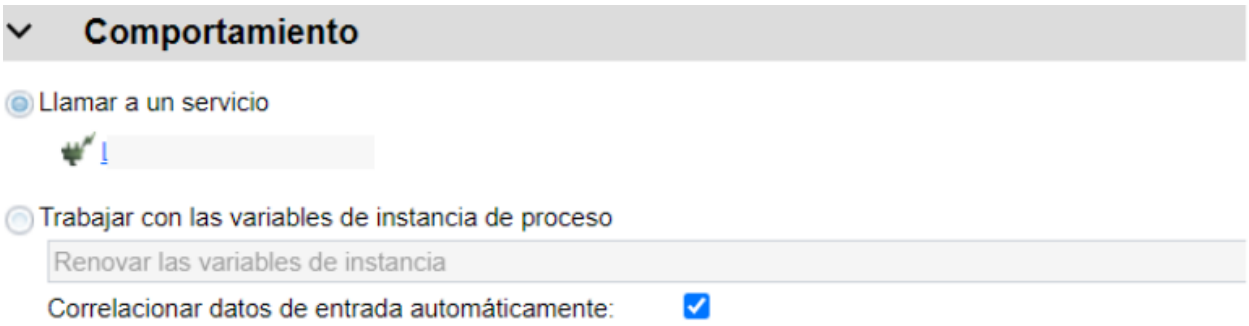
Flujo de Trabajo Sprint 3.



Nota: Fuente propia

Figura 40

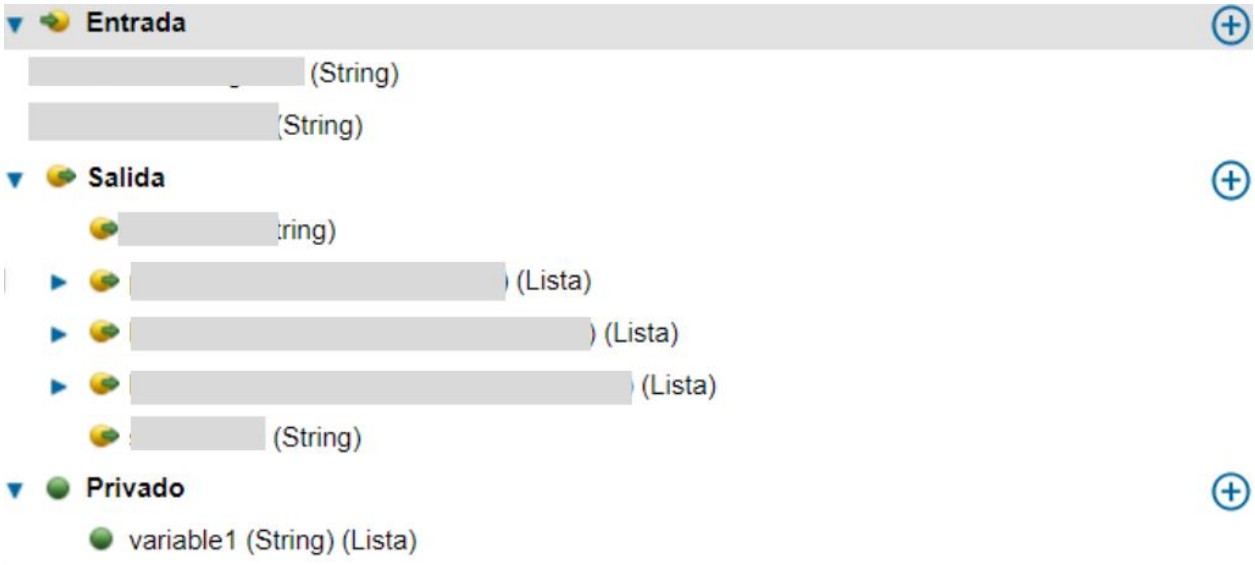
Invocación de Un Servicio.



Nota: Fuente propia

Figura 41

Correlación de Datos.



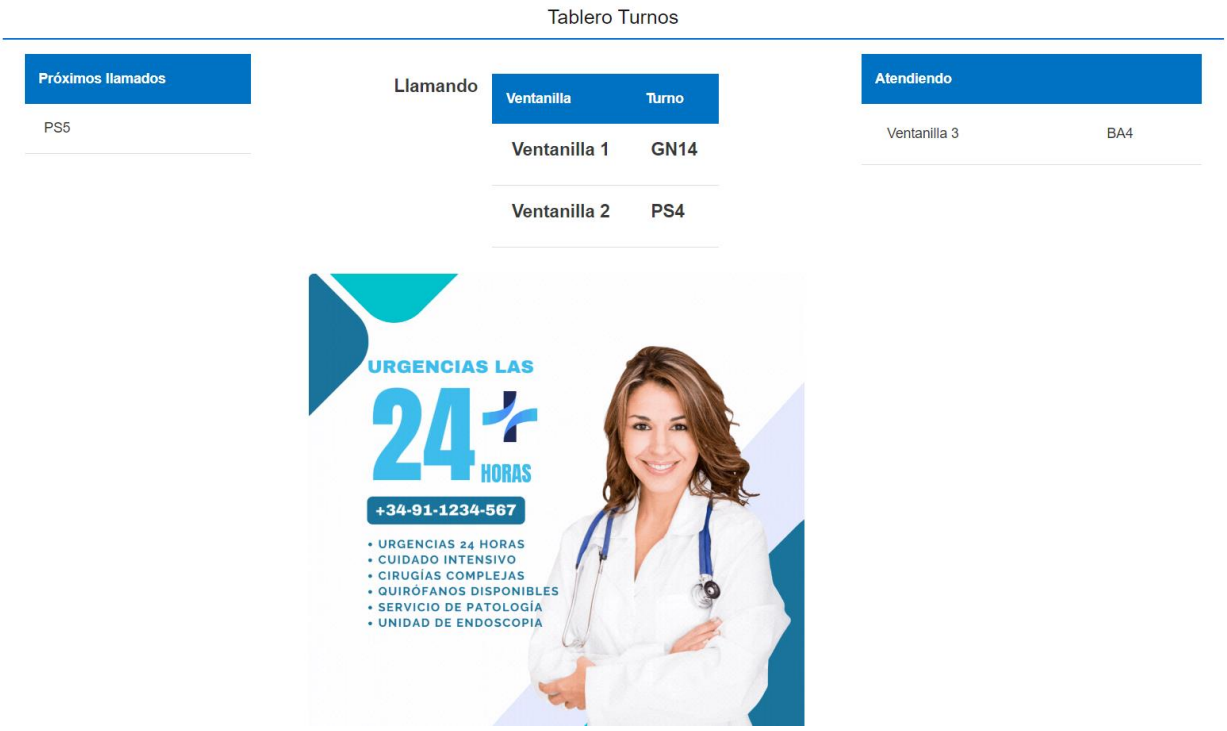
Nota: Fuente propia

Finalmente se generan pruebas de integración para validar el correcto desarrollo, operatividad y cumplimiento del objetivo de dicho sprint. Una de estas pruebas se centraba en la ejecución del módulo, verificando que listara correctamente los datos que, con la invocación del servicio REST fueron obtenidos de la colección de la base de datos y así mismo fueron definidos para visualizarse en el front-end, pues no toda la información almacenada era relevante para el usuario, en este caso.

Figura 42

Prueba de Integración Sprint 3.

Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



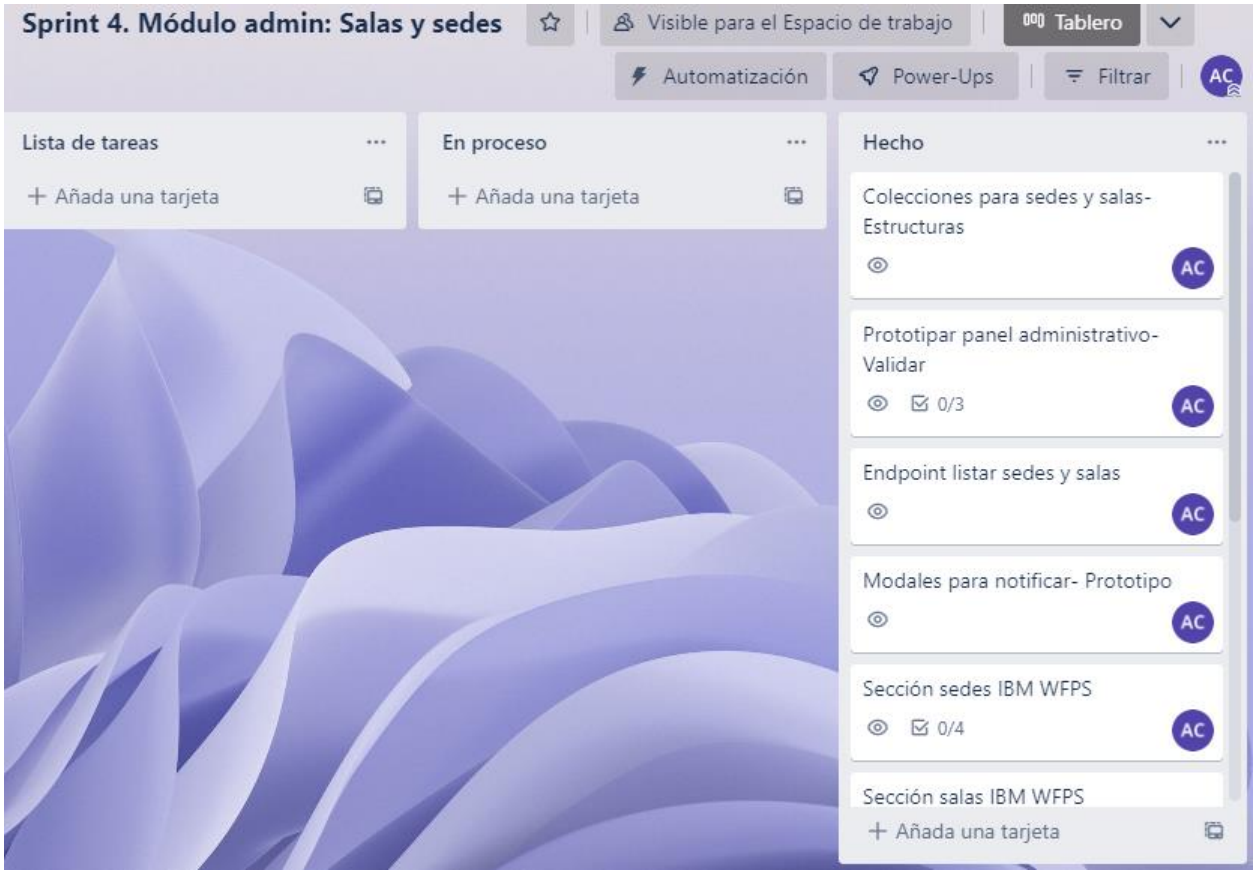
Nota: Fuente propia

8.4.Sprint 4. Módulo Administrativo: Salas y Sedes

Este sprint buscaba enfocarse en la parte administrativa del sistema, sin embargo, se decidió iniciar con la programación de 2 de las 6 entidades principales: Salas y sedes. Las actividades de este sprint se centraban en generar un CRUD (Create, Read, Update, Delete) y demás funciones de operación centradas en la información y en su manipulación; tal como se ilustra en el Sprint Backlog:

Figura 43

Sprint Backlog 4.



Nota: Fuente propia

8.4.1. Análisis

Para dar inicio al sprint 4, se generó un análisis que se enfocaba en qué nivel de operatividad debía de tener el panel administrativo aparte de generar un CRUD, pues además de mantener estas operaciones, se requería, por ejemplo, de la capacidad frente a agregar otras entidades dentro de una sala o sede, teniendo en cuenta que este tipo de organizaciones promotoras de salud suelen ser muy cambiantes frente a temas de escalabilidad por la atención masiva que presentan. Gracias a esto, se tuvo en cuenta la funcionalidad del usuario frente a

agregar o eliminar otras entidades dentro de las mencionadas en esta sección, para de esta forma facilitar el manejo del sistema, y con ello su rendimiento.

8.4.2. Diseño

En este sprint se desarrollaron dos colecciones nuevas en la base de datos, siendo denominadas: “Salas” y “Sedes”; para su creación en la etapa posterior, fue necesario definir las siguientes estructuras:

Figura 44

Estructura de Datos: Sedes y Salas.

Nota: Fuente propia

Una vez planteadas estas colecciones, se decidió diseñar un panel que le permitiera al usuario saber que opciones tendría dentro del sistema, por lo que se implementó un prototipo en la herramienta, haciendo énfasis en las salas y en las sedes. Por otra parte, se crearon los coaches para las 2 entidades, de tal forma que fuera intuitiva y de fácil uso:

Figura 45

Panel Administrativo.

Nota: Fuente propia

Figura 46

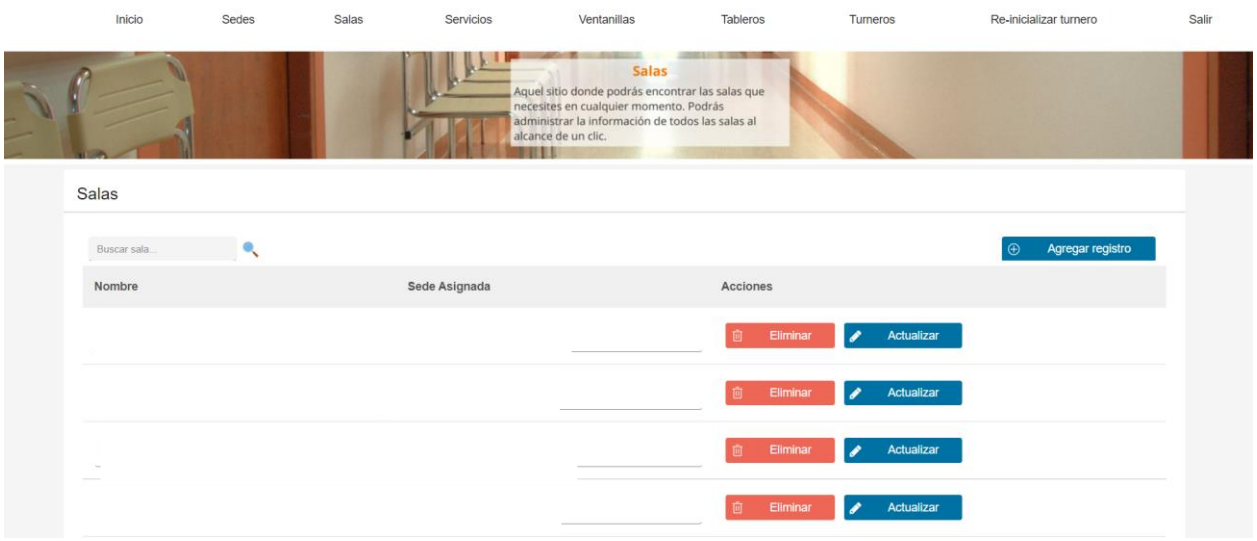
Prototipos de Modales Para Notificaciones En El Módulo.

Nota: Fuente propia

Figura 47

Prototipos Sección Sedes.

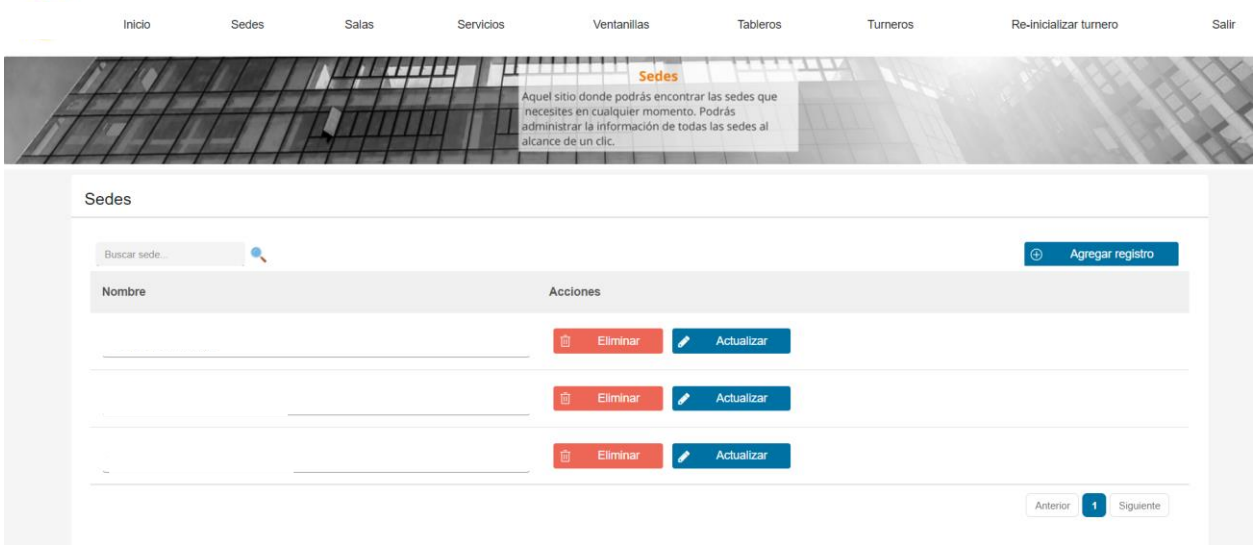
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 48

Prototipos Sección Salas.



Nota: Fuente propia

Figura 49

Secciones Modales Para Opciones En Sede.

Nota: Fuente propia

Figura 50

Secciones Modales Para Opciones en Sala.

Nota: Fuente propia

8.4.3. Desarrollo y Pruebas

Luego de plantear los prototipos para el front-end, se desarrollaron un total de 20 servicios REST, teniendo en cuenta los casos de uso y creación de entidades necesarias para su funcionamiento, incluyendo el establecimiento de las 2 colecciones que fueron mencionadas en la etapa anterior. Estos endpoints, además de generar un CRUD, tenían funciones de listar por medio de filtros e incluso listaban, por ejemplo, los servicios existentes de una sala, esto gracias a las diferentes consultas generadas en los repositorios de cada entidad, como también, permitía asignar o eliminar tableros, o demás entidades que fueran de provecho para el rendimiento en cuanto a los procesos requeridos en una organización de la salud. Además, los casos de uso se dirigían a varios frentes, como por ejemplo el de actualizar los estados de cada entidad luego de ser asignados o eliminados de una sala o una sede, de tal forma que estuvieran disponibles para otras posibles funciones. Cabe resaltar que muchos de estos endpoints retornaban un objeto de un

tipo específico, así como arrays, como también un response estandarizado, tal como “200”, “500”, “404”, inicialmente.

Figura 51

Endpoints Sprint 4.

Nota: Fuente propia

Para verificar que todos los servicios creados durante este sprint estuvieran realizando lo esperado, se implementaron pruebas unitarias. Para evidenciar una de estas pruebas, se toma como ejemplo, la situación de listar los servicios en una sala, así como buscar una sede por su identificador único:

Figura 52

Evidencia Pruebas Unitarias Sprint 4.

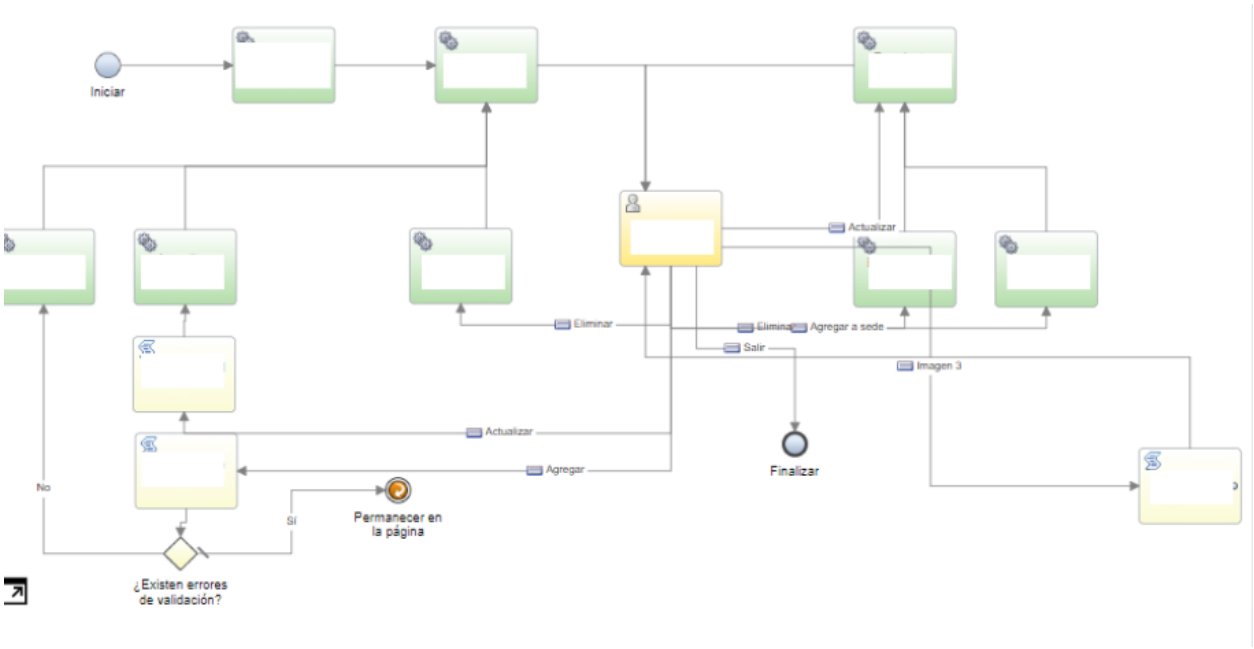
Nota: Fuente propia

Luego de realizar todas las pruebas unitarias, se generó la documentación de las APIS en Swagger, de tal forma que pudieran consumirse dentro del IBM Workflow Process Service y de esta manera poder implementar los servicios en el flujo de trabajo que se creó para el funcionamiento de la parte del módulo administrativo:

Figura 53

Flujo de Trabajo Sección Sedes.

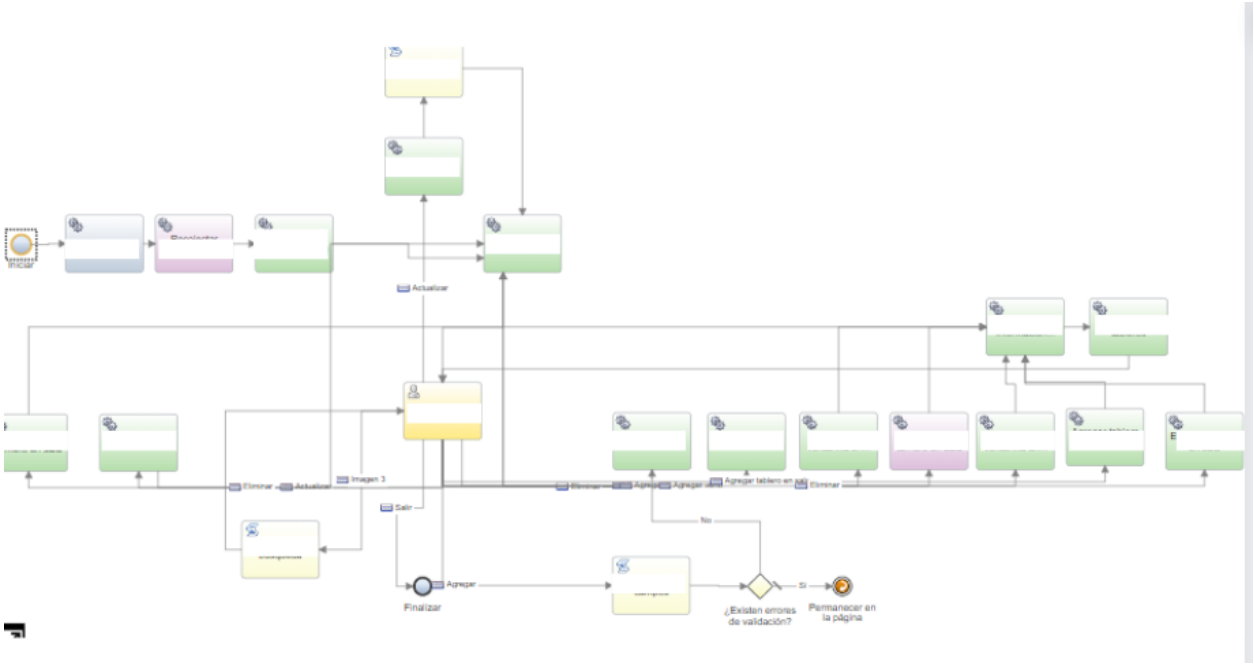
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 54

Flujo de Trabajo Sección Salas.

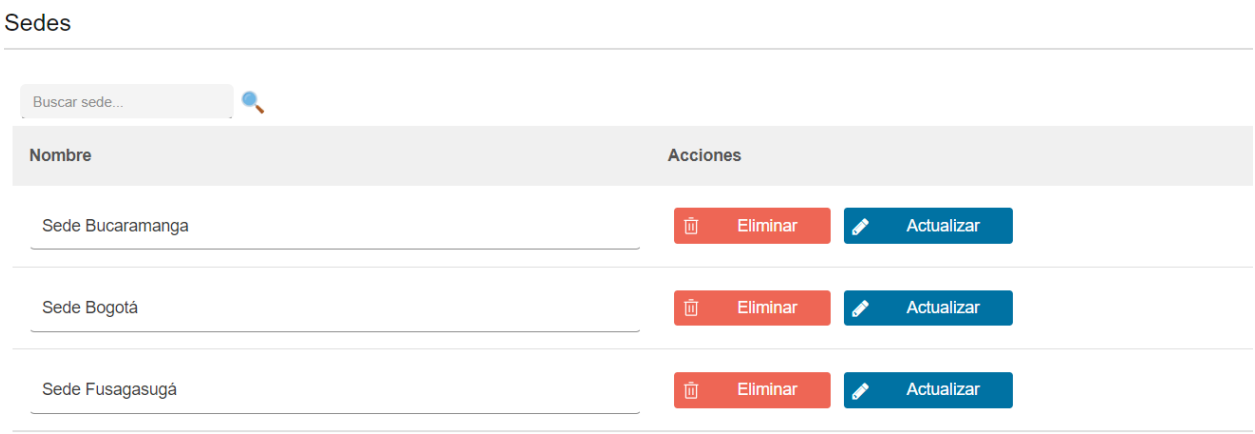


Nota: Fuente propia

Una vez generada la incorporación de los componentes mencionados con anterioridad, se establecieron pruebas de integración que permitieron la validación exitosa del avance del módulo administrativo, dando paso al siguiente sprint. Para evidenciar una de estas pruebas, se ejecutó el módulo administrativo, se ingresó a la opción denominada como “Sedes”, para posteriormente visualizar las sedes existentes, teniendo en cuenta que para ello se requería del consumo de un servicio web que se sometía a la ejecución de diversos casos de uso y un método específico del repositorio, que se encargaba de ingresar hasta la colección de la base de datos, traer la información y retornarla:

Figura 55

Evidencia Prueba de Integración Sprint 4.



Nota: Fuente propia

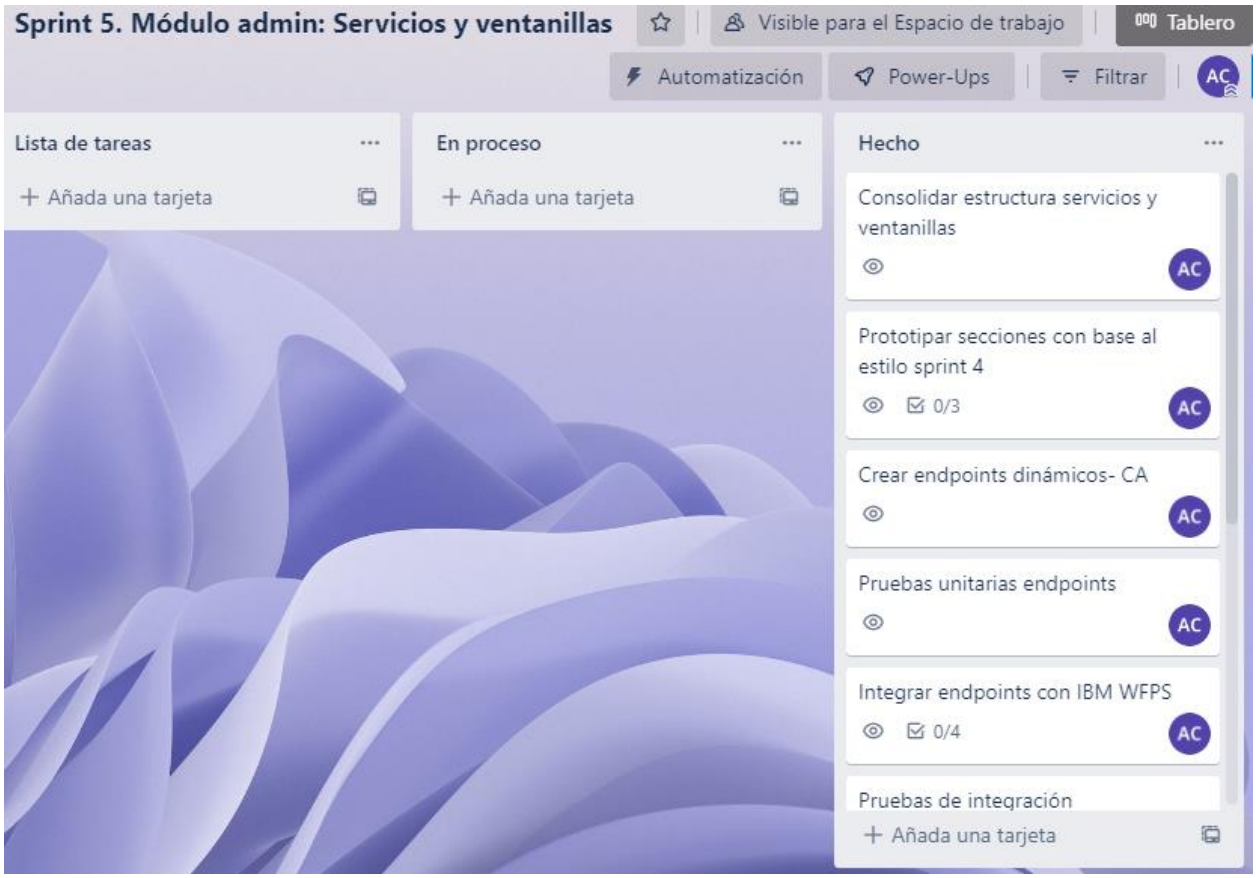
8.5. Sprint 5. Módulo Administrativo: Servicios y Ventanillas

El objetivo de este sprint se centró en el desarrollo de las secciones de los servicios y ventanillas, para de esta forma completar 4 de las 6 entidades requeridas en un principio para una funcionalidad óptima en cuanto al módulo administrativo.

Para cumplir este objetivo, se realizaron las tareas del Sprint Backlog; estas actividades se enfocaban en generar para la entidad de servicios únicamente un CRUD; y para las ventanillas operaciones adicionales, tales como agregar un servicio en ventanilla o eliminarlo.

Figura 56

Sprint Backlog 5.



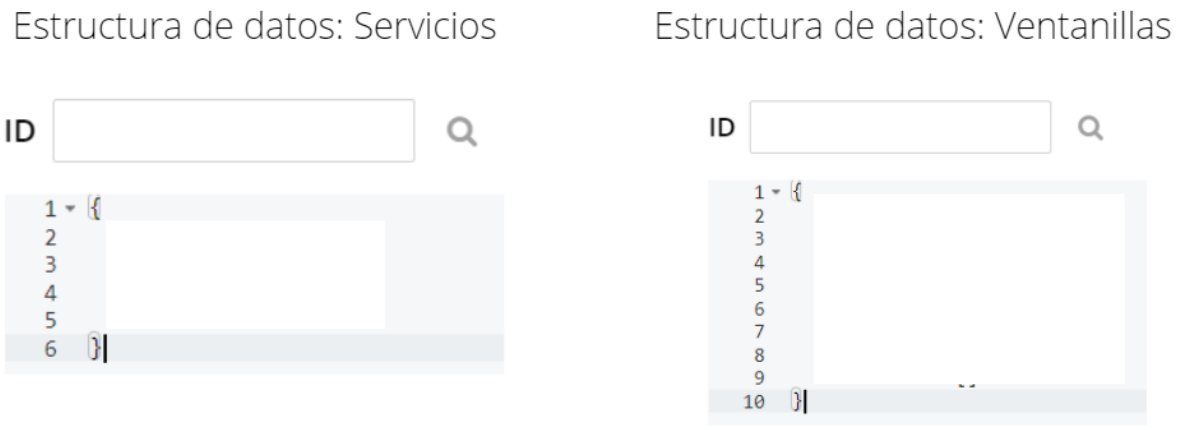
Nota: Fuente propia

8.5.1. Análisis

Para llevar a cabo las actividades planteadas, se optó por las siguientes estructuras para las colecciones en la base de datos. Finalmente, no se requirió de ningún otro tipo de análisis, pues ya se tenía claridad de que debía abarcar cada tarea del Sprint Backlog.

Figura 57

Estructura de Datos: Servicios y Ventanillas.



Nota: Fuente propia

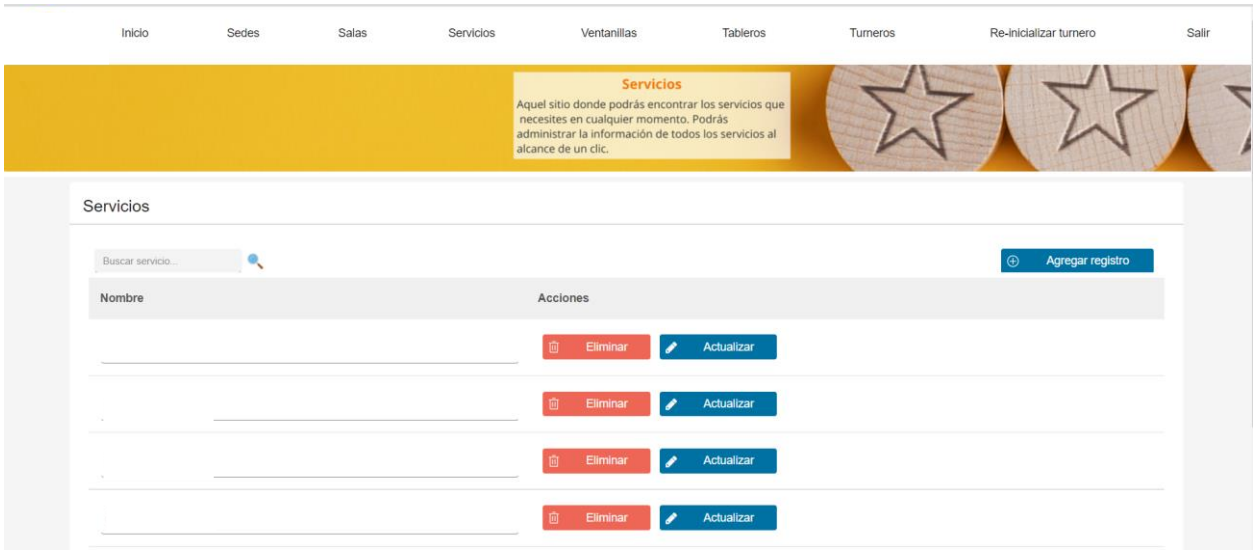
8.5.2. Diseño

Para el inicio de esta etapa, se decidió denominar a las colecciones en la base de datos como: Servicios y Ventanillas. Luego de ello, se realizó un enfoque en el front-end, siguiendo el prototipo planteado en el sprint 4, pues al ser parte del mismo módulo, debía contar con los mismos estilos, sin embargo, sus funcionalidades si eran cambiantes. Al finalizar, se obtuvieron los siguientes prototipos, tanto en el panel administrativo como en las secciones pertinentes:

Figura 58

Prototipo Sección Servicios.

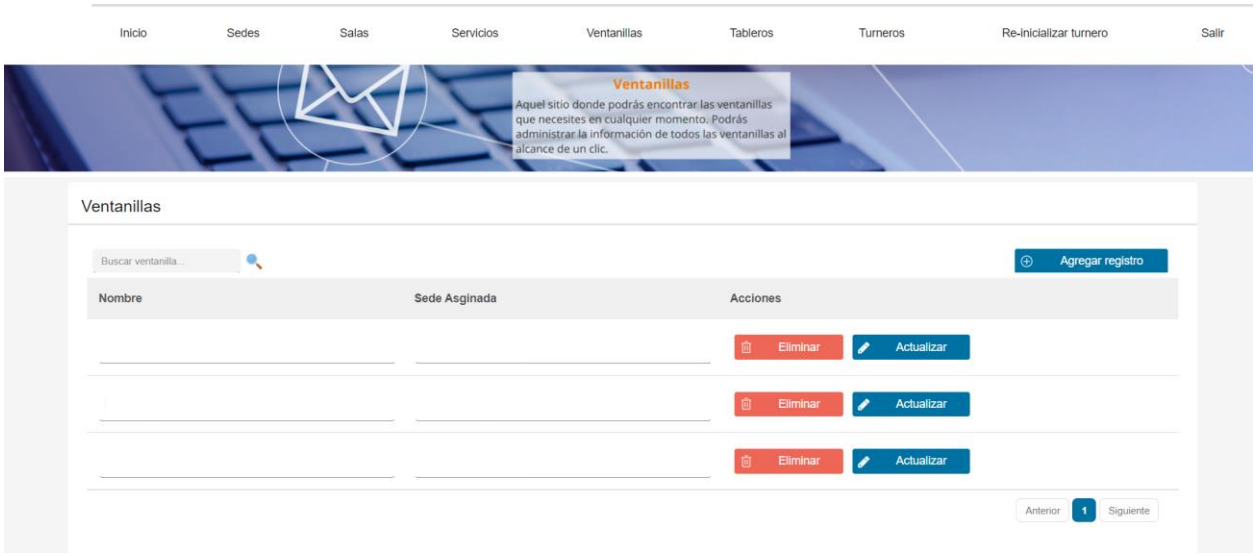
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 59

Prototipo Sección Ventanillas.



Nota: Fuente propia

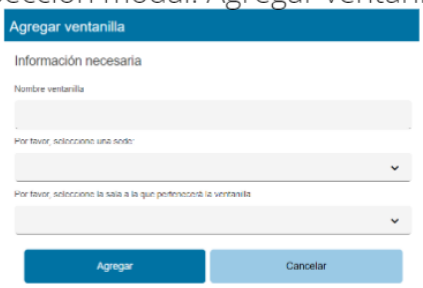
Figura 60

Secciones Modales Para Opciones en Ventanilla.

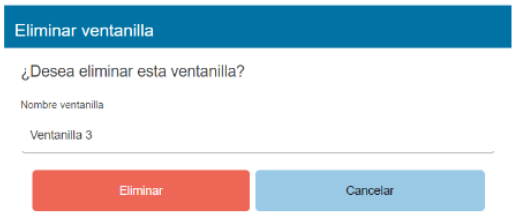
Sección modal: Actualizar ventanilla

Sección modal: Agregar servicios a ventanilla

Sección modal: Agregar ventanilla



Sección modal: Eliminar ventanilla



Nota: Fuente propia

Figura 61

Secciones Modales Para Opciones en Servicio.

Nota: Fuente propia

Es importante aclarar que cada uno de estos prototipos incluía secciones modales que le permitían al usuario tener claridad en caso de que existiera un error, sin necesidad de finalizar su proceso en consecuencia de este. En esta etapa se realizó el prototipo, sin embargo, el control de errores se ve inmerso en la siguiente etapa.

8.5.3. Desarrollo y Pruebas

Teniendo en cuenta el objetivo del sprint, se plantearon y desarrollaron 15 endpoints en total, los cuales abarcaban toda aquella operatividad de la que se requería en este fragmento del módulo; tales como, actualizar ventanilla, listar los servicios con los que cuenta una ventanilla, buscar un servicio o ventanilla, eliminar servicios, entre otros. Para estos servicios REST, los casos de uso planteados por el clean architecture no eran muy diferentes al objetivo de los endpoints, pues para este caso no se requería de diversas funcionalidades internas, más que, un

CRUD y un par de métodos adicionales, por lo que, así mismo sucedió con los métodos en el repositorio destinado a cada entidad.

Figura 62

Endpoints Sprint 5.

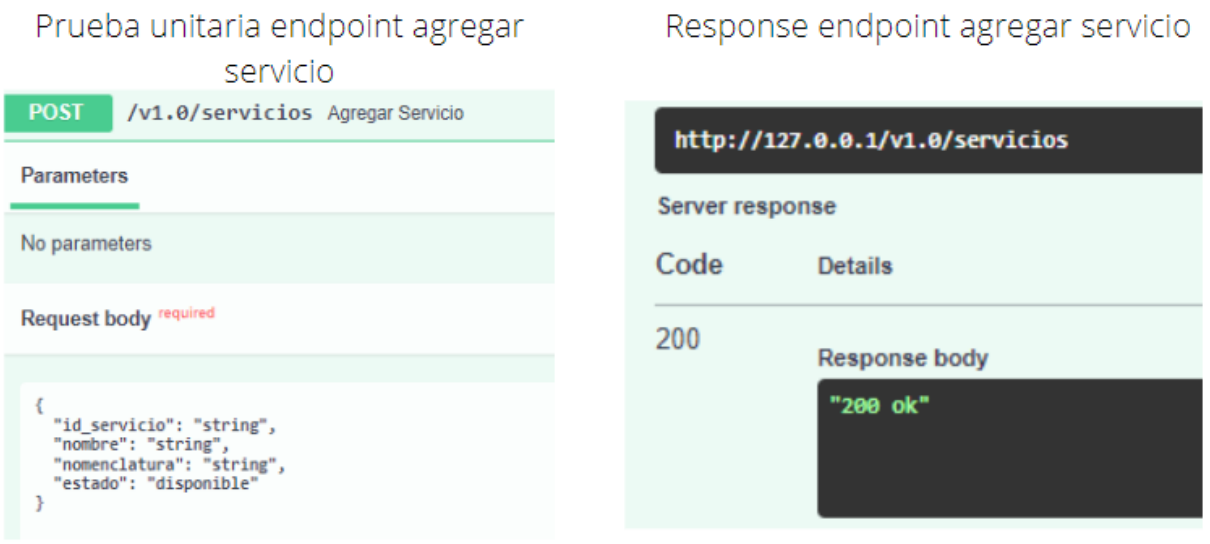
Nota: Fuente propia

Posteriormente se generaron las pruebas unitarias pertinentes, de tal forma que se pudiera dar paso a la implementación de dichos servicios en los flujos de trabajo en la herramienta. Una de estas pruebas buscaba ejecutar el endpoint en fastAPI, donde, al agregar un servicio se debían

de solicitar datos del tipo “Servicio”, y, una vez creado, debía retornar un string con un response estandarizado, tal como se ilustra a continuación:

Figura 63

Evidencia Prueba Unitaria Sprint 5.



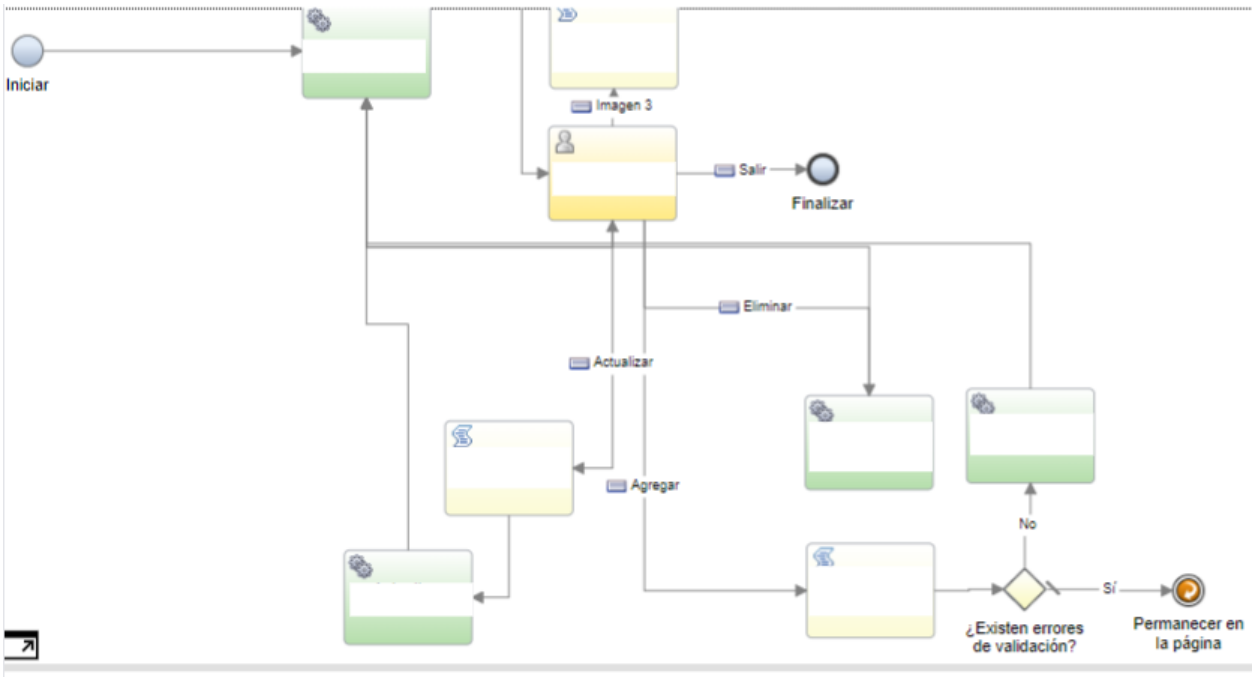
Nota: Fuente propia

Una vez realizada la programación en Python y fastAPI, se documentaron los servicios y las demás tareas se enfocaron en realizar un flujo de trabajo independiente para cada una de estas entidades dentro de la herramienta de IBM, incluyendo tareas de servicio, creación de variables, correlación de datos, integración con los coaches, control de errores por flujo de servicio y su creación, entre otros. Culminadas estas actividades se obtuvieron los siguientes workflows, asumiendo que debían ser anidados para de esta manera comunicarse con el panel administrativo:

Figura 64

Flujo de Trabajo Sección Servicios.

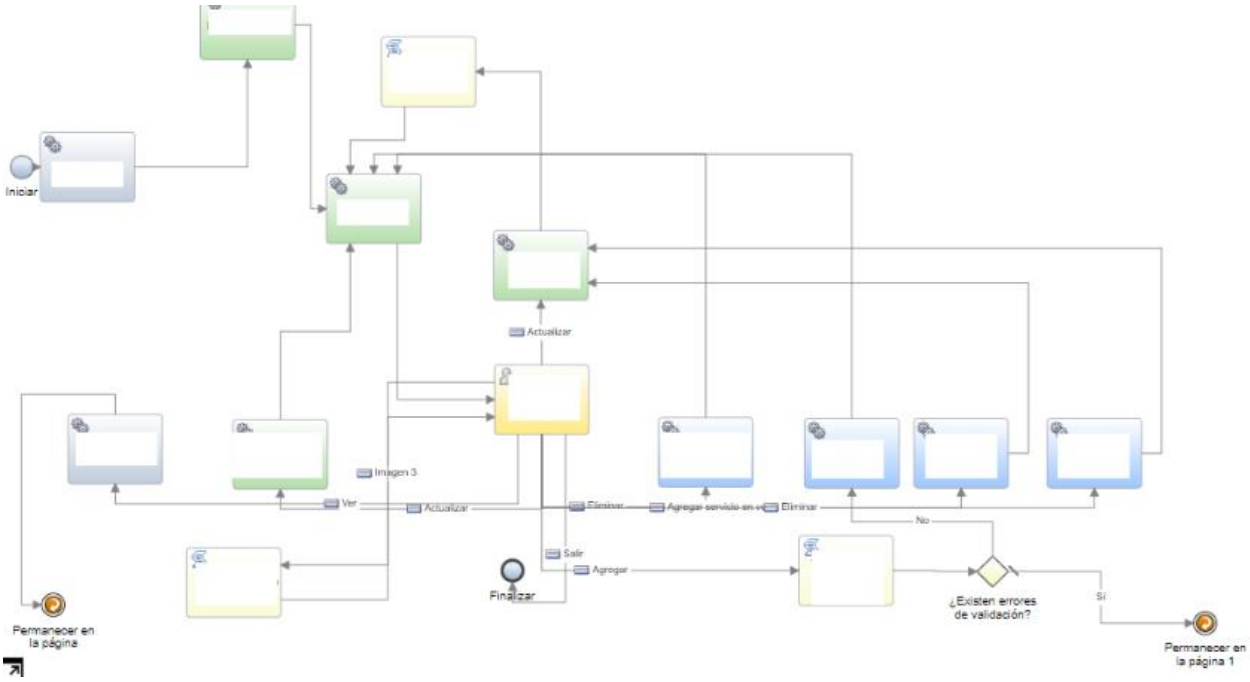
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 65

Flujo de Trabajo Sección Ventanillas.

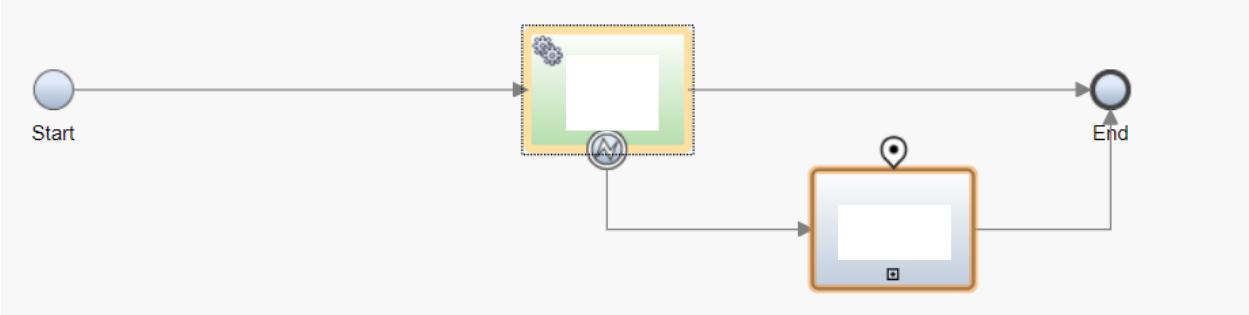


Nota: Fuente propia

Para finalizar este sprint, se generaron pruebas de integración, ejecutando el módulo y validando su funcionamiento con todos los componentes relacionados. En este caso se buscó evidenciar el control de errores realizado como prueba dentro de la herramienta, ingresando en modo debug a la ejecución del servicio de prueba y validando el camino que tomaba de acuerdo con el error, como también validar la notificación a nivel de usuario:

Figura 66

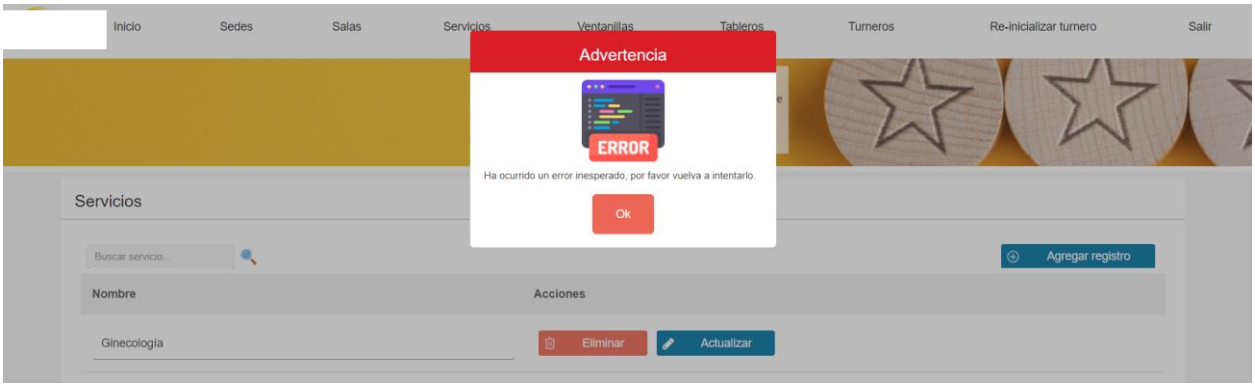
Prueba de Integración Sprint 5.



Nota: Fuente propia

Figura 67

Prueba de Integración-Nivel de Usuario Sprint 5.



Nota: Fuente propia

8.6. Sprint 6. Módulo administrativo: Tableros, Turneros y Re-inicialización de Turneros

El sprint 6, además de culminar el módulo administrativo, buscaba el desarrollo de una sección para los tableros, los turneros y la re-inicialización de los turneros. Por este objetivo el

Sprint Backlog se centraba en tareas como el CRUD para los tableros y los turneros, pues se planteó inicialmente que ellos podrían ser asignados desde otras entidades, más no podían contar con esas funciones dentro de sus secciones; en cuanto a la re-inicialización de los turneros, se debían realizar actividades centradas en filtrar información de tal manera que, al momento de inicializar un turnero, todos los turnos contenidos por sede, sala y servicio fueran guardados en una nueva colección, como también actualizar sus datos en cero, para que, al inicio del día estos iniciaran con turnos y tiempos de espera en ceros.

Figura 68

Sprint Backlog 6.

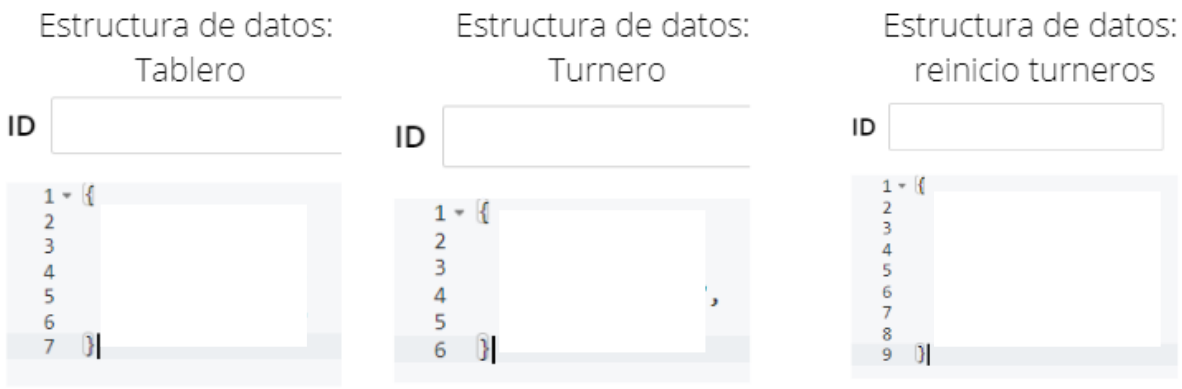
Nota: Fuente propia

8.6.1. Análisis

Al momento de adoptar una funcionalidad en el sistema referente a una re-inicialización de turneros, fue necesario tener en cuenta que diariamente o dependiendo del horario de una ventanilla esta debía reiniciarse al día siguiente o al momento de iniciar nuevamente su servicio. Por esto razón se realizó un análisis frente a la utilidad que podría tener almacenar esta información, convirtiéndose en un caso relevante, esto debido a que todos los datos le permitirían a un usuario tomarlos como base para la futura toma de decisiones, por lo que se decidió almacenarla en una colección al momento en que el usuario decidiera reinicializar dicho turnero; por otra parte se debían de tener en cuenta que estructuras de datos eran necesarias para los tableros, turneros y para un backup de la información en mención, obteniendo:

Figura 69

Estructura de Datos: Tableros, Turneros y Reinicio Turneros.



Nota: Fuente propia

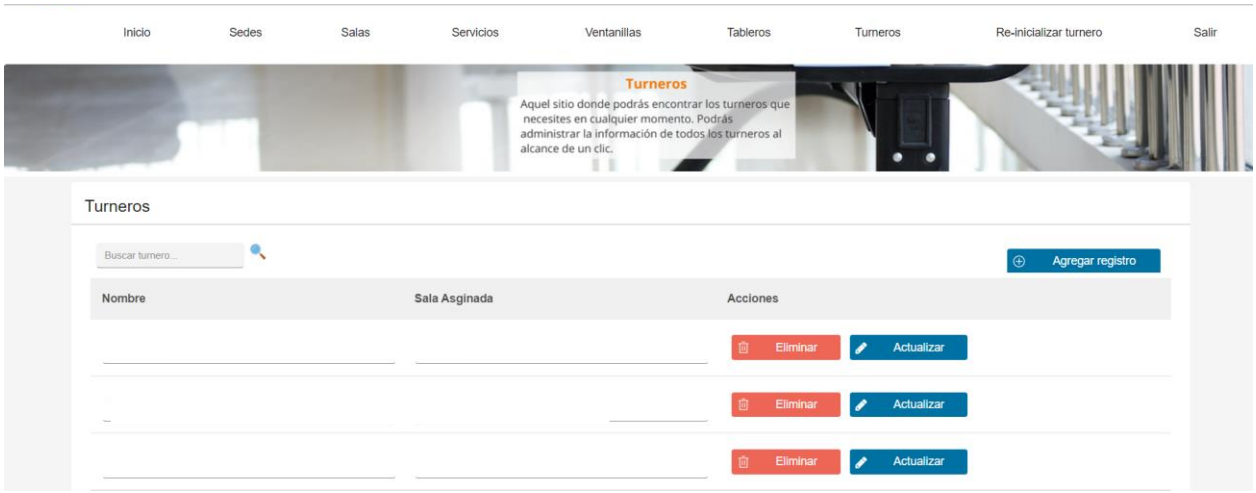
8.6.2. Diseño

Una vez definidas las estructuras de los datos, se optó por denominar las colecciones como:” bkTiempoAtencion”, “Tableros” y “Turneros. Luego el enfoque se dirigió al front-end,

de tal forma que fueron diseñados 3 coaches diferentes, cada uno con las especificaciones necesarias para la usabilidad esperada. Al igual que en los 2 sprints anteriores, la interfaz debía ser similar, pues todas ellas se encontraban inmersas dentro de un mismo módulo y estéticamente sería mejor mantener un mismo diseño. En cuanto a la sección de re-inicialización, se manejaron los mismos colores y se optó por manejar elementos simples tanto para el uso del usuario, como para facilitar la búsqueda del turnero. Los prototipos obtenidos fueron los siguientes:

Figura 70

Prototipo Sección de Turneros.

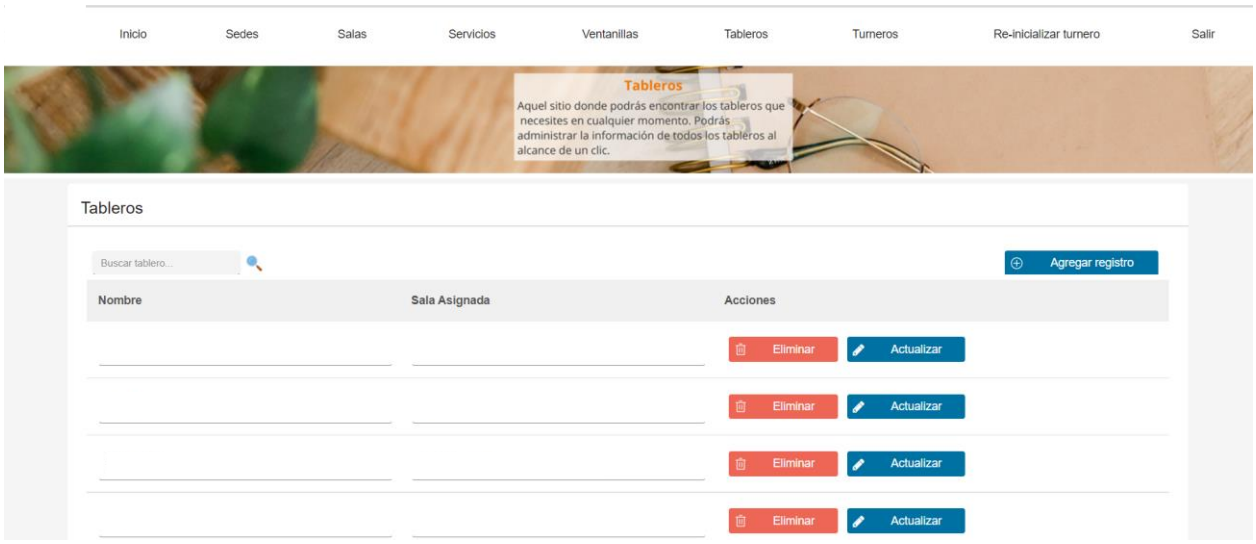


Nota: Fuente propia

Figura 71

Prototipo Sección de Tableros.

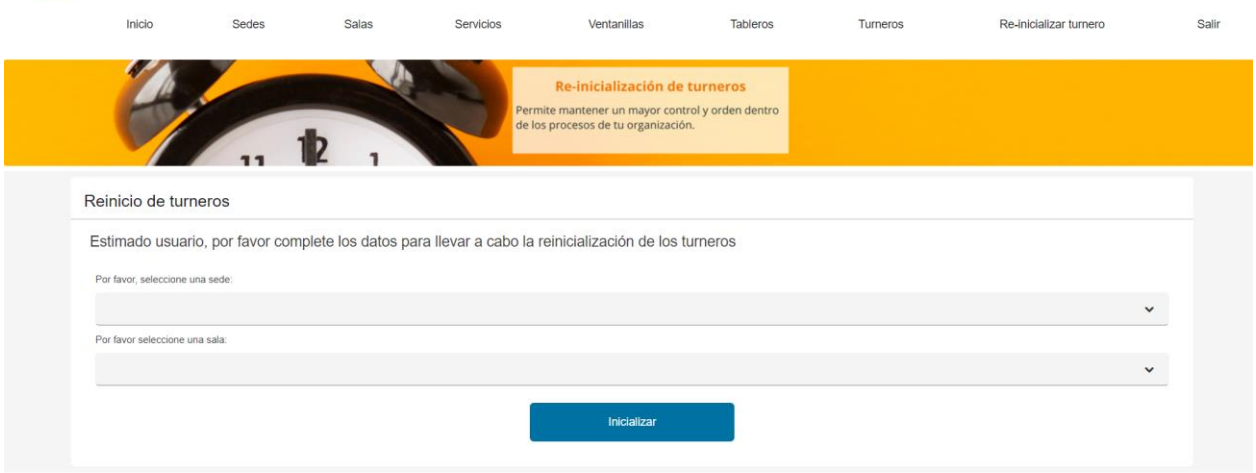
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 72

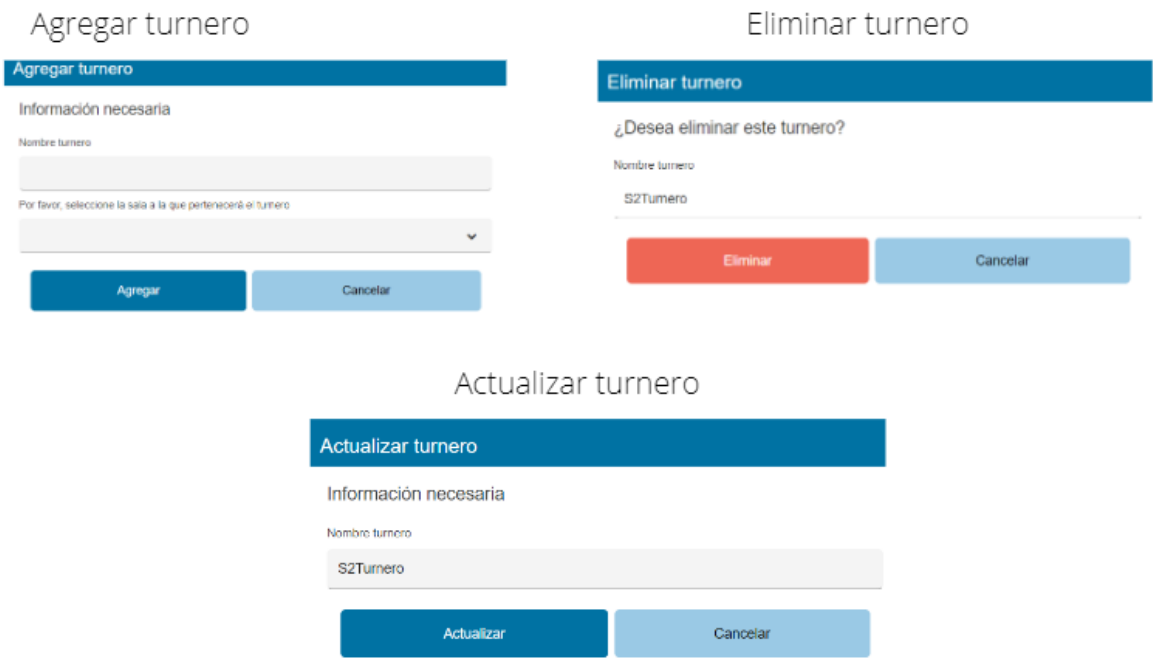
Prototipo Sección de Re-inicialización de Turneros.



Nota: Fuente propia

Figura 73

Secciones Modales Para Opciones de Turneros.



Nota: Fuente propia

Figura 74

Secciones Modales Para Opciones de Tableros.

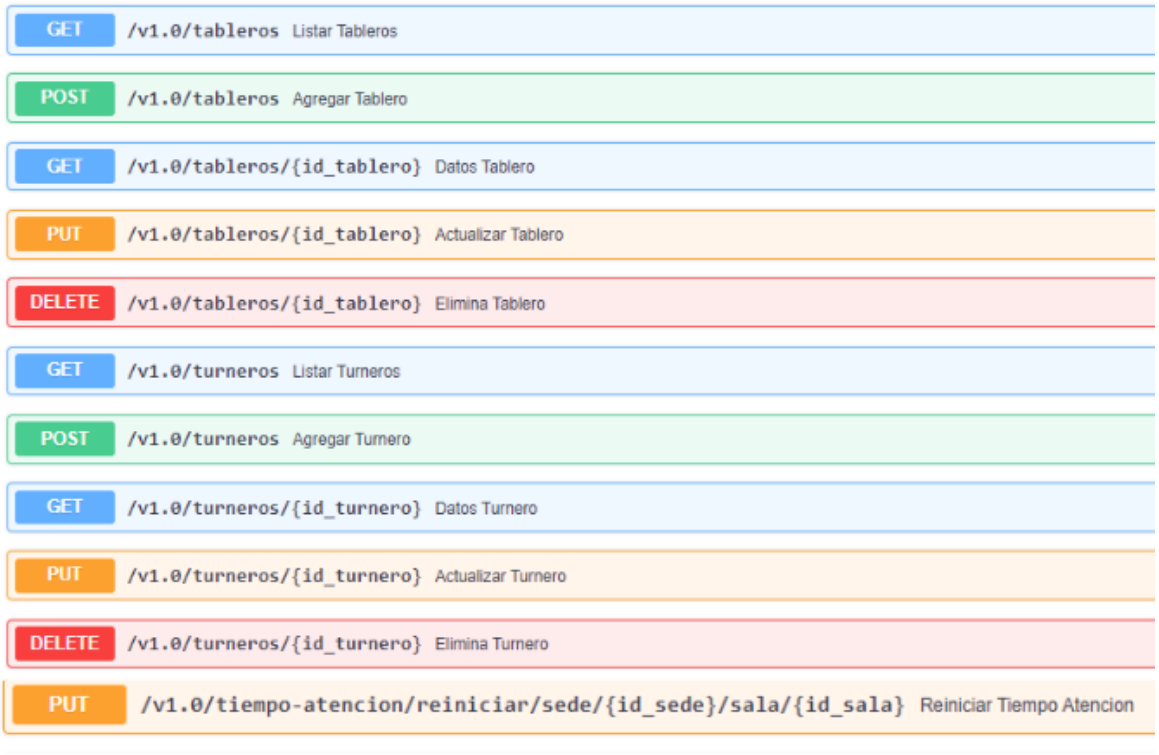
Nota: Fuente propia

8.6.3. Desarrollo y Pruebas

Inicialmente se crearon las colecciones dentro de la base de datos, pues de esta forma podría manipularse la información y dar continuidad al proceso. Para llevar a cabo el CRUD de los tableros, los turneros, y la funcionalidad de reinicio de un turnero, se obtuvo un total de 11 endpoints, cada uno de ellos con los casos de uso necesarios, como, por ejemplo: Buscar un tablero o un turnero, reiniciar el tiempo de atención, eliminar, actualizar, entre otros. De igual forma ocurrió con los repositorios dentro del código, sin embargo, no se generaron operaciones adicionales ya que para las 2 últimas entidades se definió un CRUD básico.

Figura 75

Endpoints Sprint 6.



Nota: Fuente propia

Una vez concluida la programación de estos servicios, se generaron pruebas unitarias para cada uno de ellos dentro de fastAPI; en este caso se decidió evidenciar la prueba unitaria del caso en que se quisiera reiniciar el tiempo de atención, donde, se solicitaron 2 parámetros: id_sede y id_sala, para luego obtener la información, guardar los datos en la colección de bkTiempoAtencion, eliminarla en tiempoAtencionVentanillas y finalmente brindar un response estandarizado de tipo 200:

Figura 76

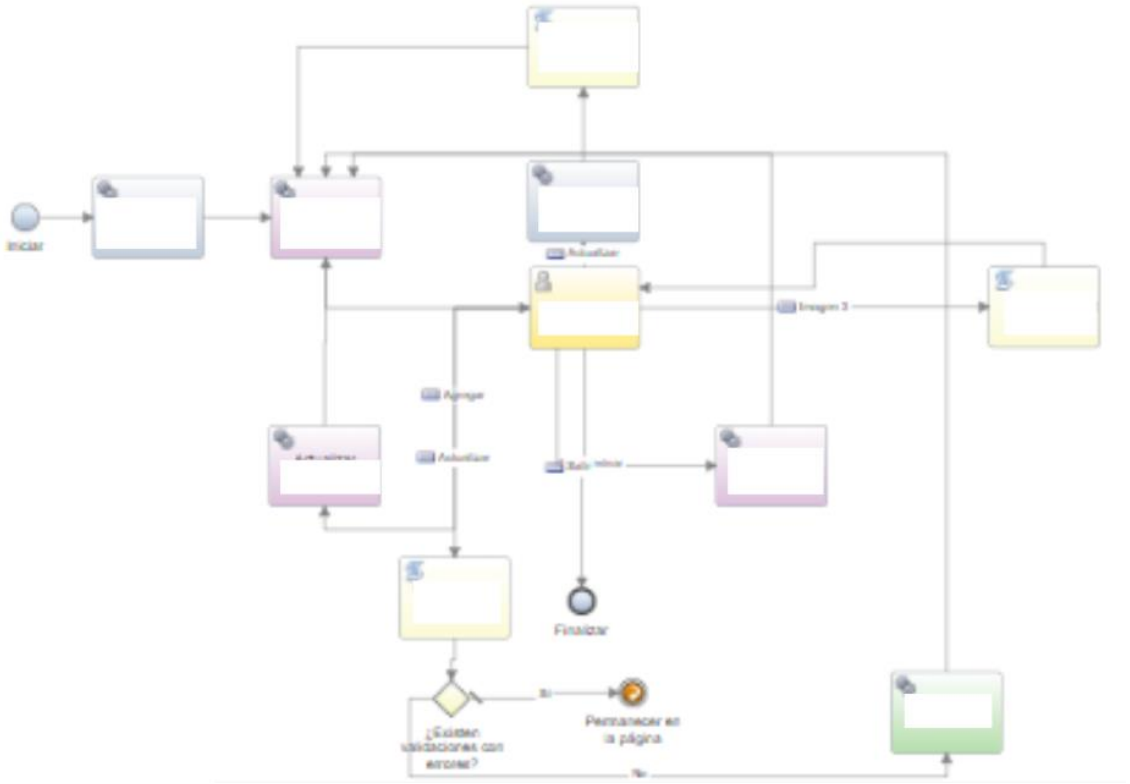
Evidencia Prueba Unitaria Sprint 6.

Nota: Fuente propia

Al pasar todas las pruebas unitarias se procede con la documentación de Swagger, para poder exportarla en tipo JSON y consumirla en la herramienta. Los procesos siguientes se realizaron en el IBM Workflow Process Service, donde se crearon los servicios externos, flujos de servicios, tareas de servicio, scripts, se correlacionaron datos y se implementó todo aquel proceso que diera paso al correcto desarrollo del sprint.

Figura 77

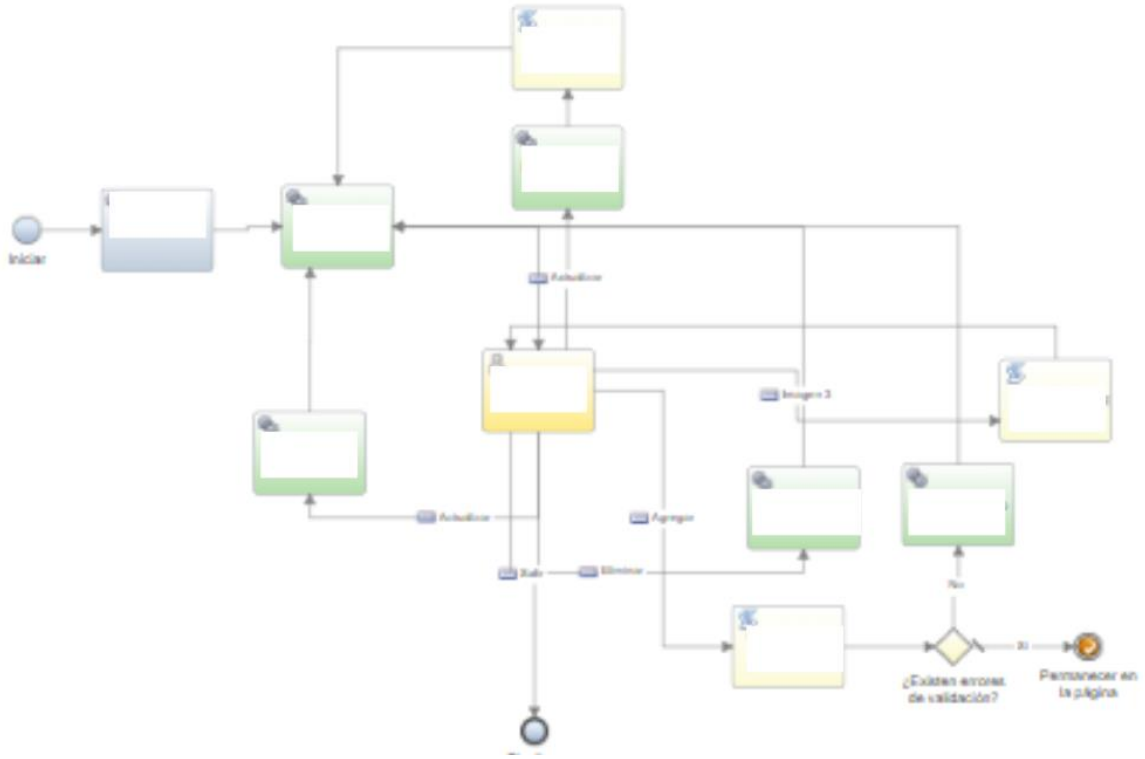
Flujo de Trabajo Sección Turneros.



Nota: Fuente propia

Figura 78

Flujo de Trabajo Sección Tableros.



Nota: Fuente propia

Figura 79

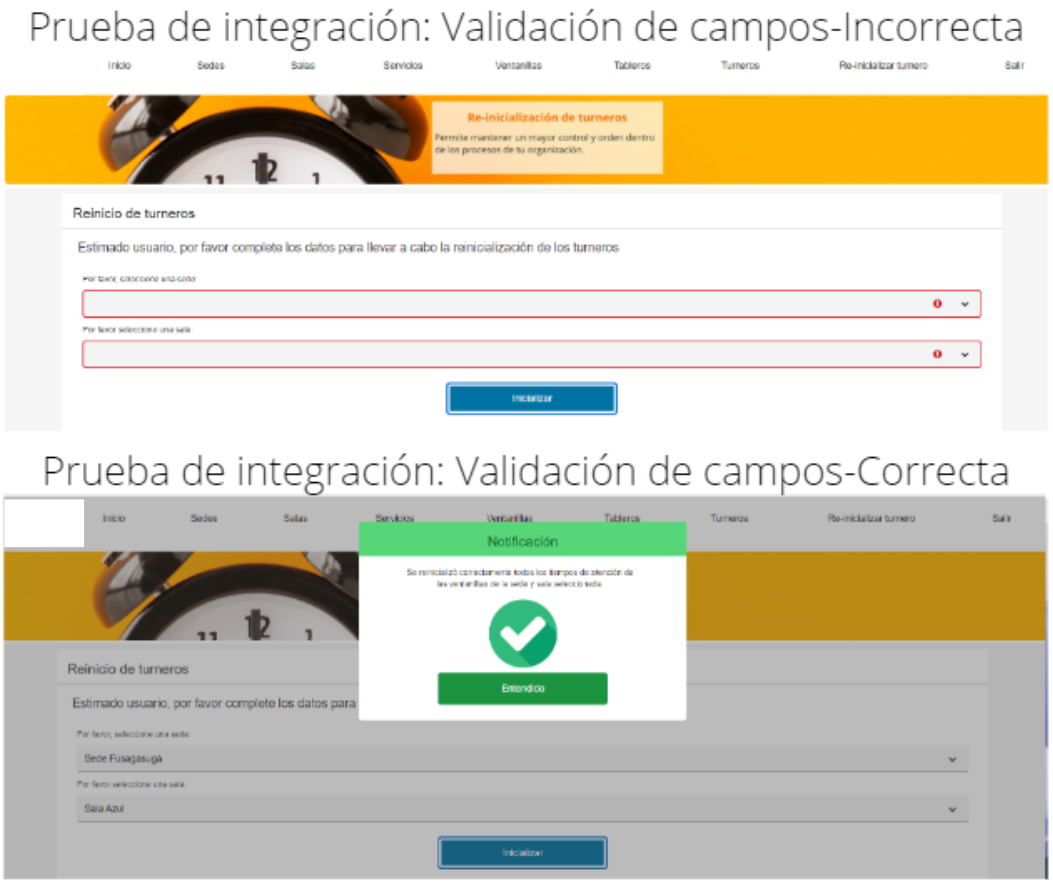
Evidencia Creación de Variables.

Nota: Fuente propia

Para dar por concluido el sprint número 6 se realizaron las pruebas de integración necesarias para validar que todo el desarrollo fuera óptimo. En este caso, se implementaron pruebas como, la ejecución del módulo de reinicio, donde se validó el envío de campos vacíos para analizar si el sistema si estaba enviando las alertas de qué parámetros eran necesarios y cuales no, así mismo se brindaron los datos correctamente para conocer la respuesta de todos los componentes unificados en el caso de un proceso exitoso.

Figura 80

Prueba de Integración Sprint 6.



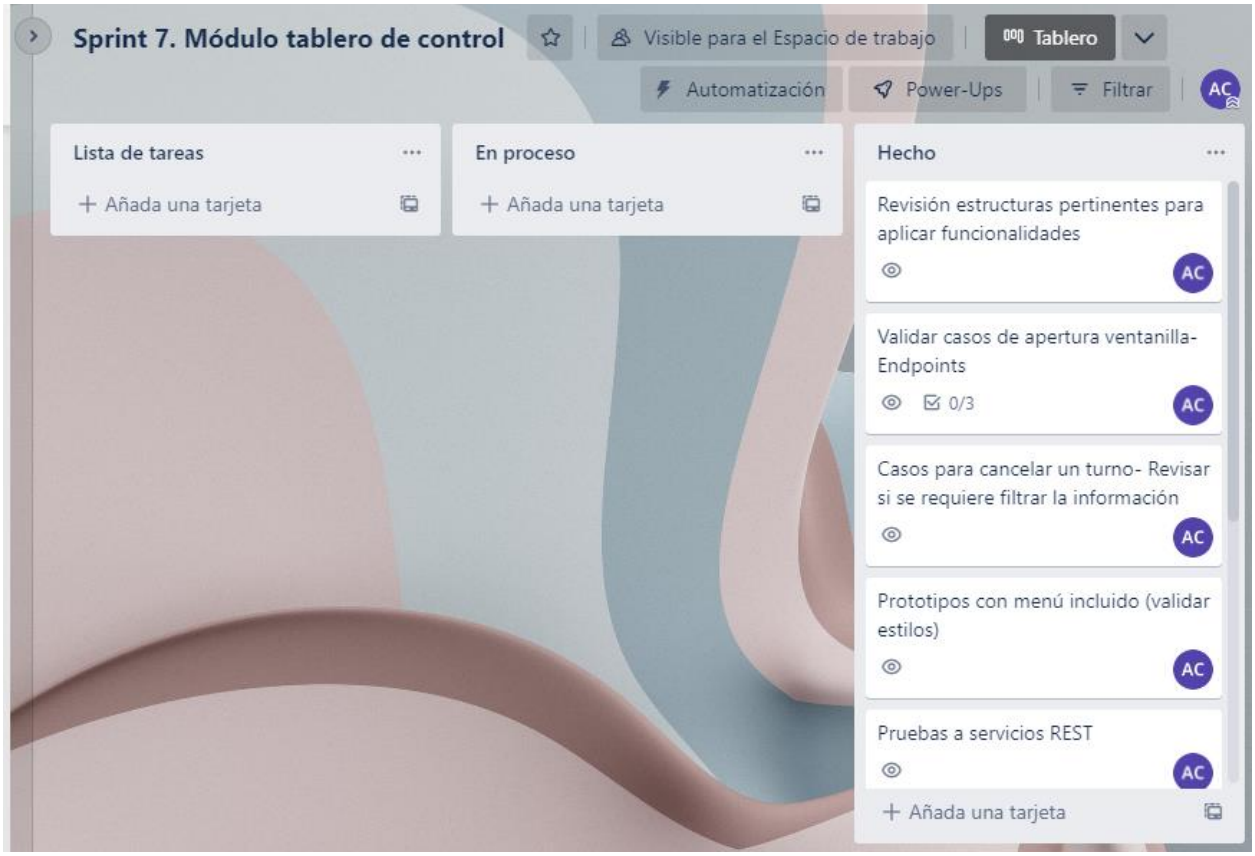
Nota: Fuente propia

8.7. Sprint 7. Módulo Tablero de Control

El objetivo del sprint 7 se centraba en el desarrollo de un módulo que le permitiera a un operador de servicio controlar los casos urgentes de manera simple; como, por ejemplo, la cancelación de un turno a último momento, o la apertura o el cierre de una ventanilla por diferentes externalidades. Es por esta razón que las actividades se centraban en generar funcionalidades en cuanto a la manipulación de la información de manera que solo con un par de pasos el usuario pudiera realizar estas tareas con éxito. Para este sprint se tuvieron en cuenta algunas de las actividades descritas en el siguiente Sprint Backlog:

Figura 81

Sprint Backlog 7.



Nota: Fuente propia

8.7.1. Análisis

Las actividades por realizar requirieron de un análisis enfocado a la estructura de los datos, pues luego de desarrollar varios módulos se consideraba óptima la cantidad de colecciones generadas en la base de datos, donde claramente intervenía su estructura, por lo que, llevó al análisis referente a considerar utilizar algunas de estas colecciones ya que los datos implementados en ellas eran los requeridos para completar las tareas de este sprint. Por esta razón se decidió reutilizar algunos datos de la colección de “ventanillas” y de “turnos”. Por otra parte, se consideró crear un coach que le permitiera al usuario (operador de servicio) contar con dos funcionalidades iniciales: Cancelar turnos y modificar el estado de una ventanilla, pues el módulo administrativo se centraba en otros procesos y se facilitaba asignar roles en coaches distintos.

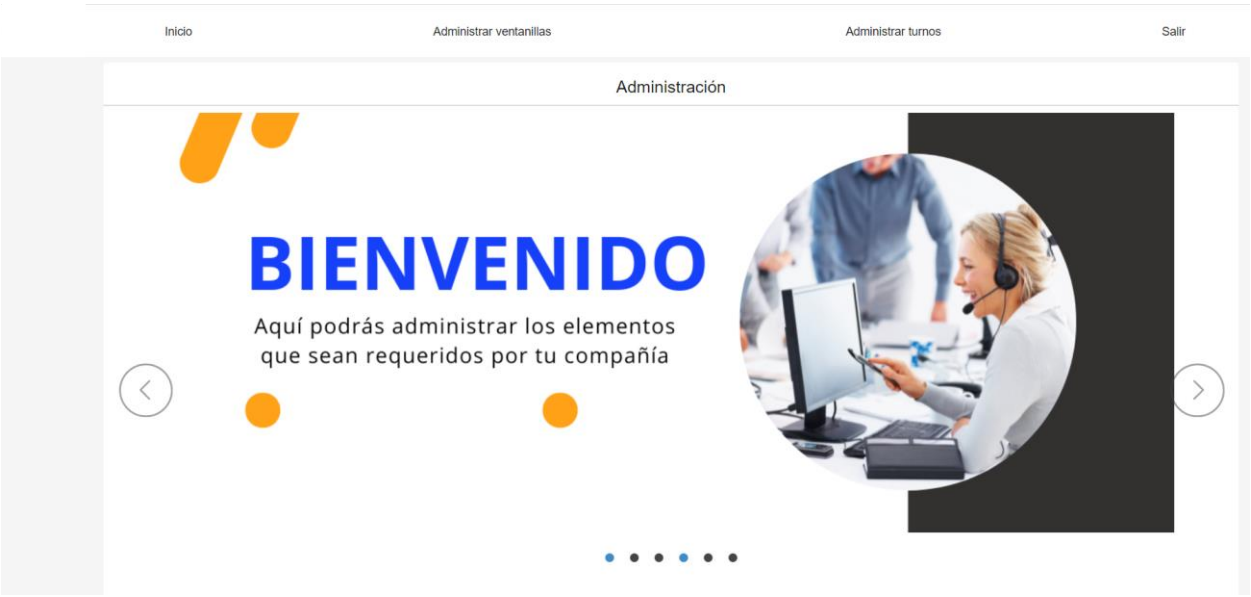
8.7.2. Diseño

Con base al análisis planteado anteriormente, se decidió enfocar esta etapa al front-end, pues se debían generar estilos diferentes a los que ya habían sido realizados. Así mismo se dividió el coach en 2 secciones diferentes, para permitirle al usuario elegir de acuerdo con su requerimiento. De esta forma se implementaron 2 prototipos, evidenciados a continuación:

Figura 82

Prototipo Inicio del Módulo Tablero de Control.

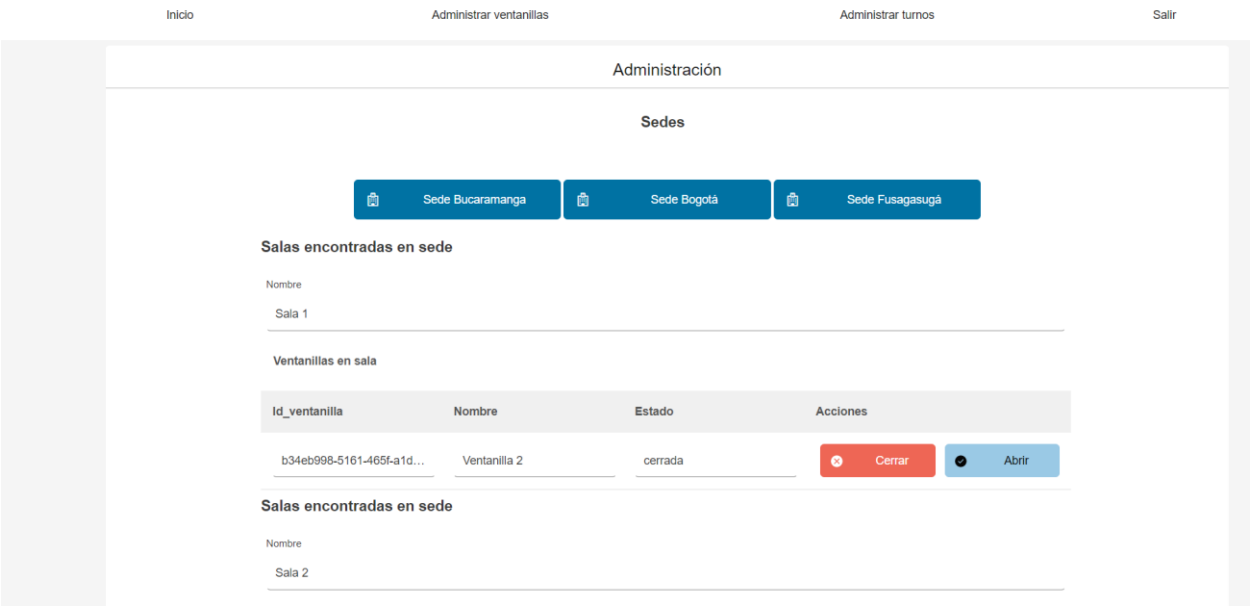
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 83

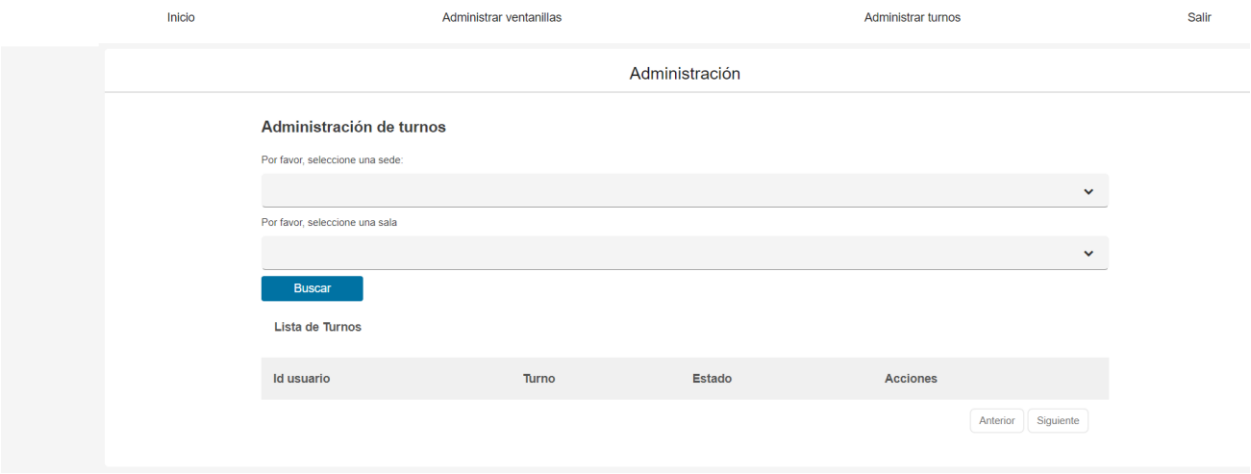
Prototipo Sección Administración de Ventanillas.



Nota: Fuente propia

Figura 84

Prototipo Sección Administración de Turnos.



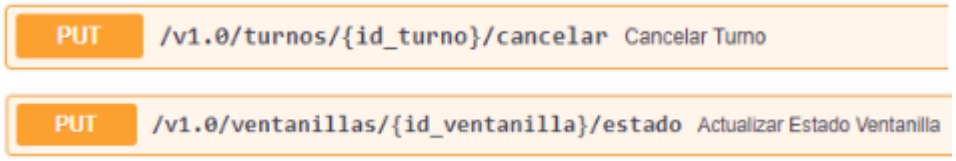
Nota: Fuente propia

8.7.3. Desarrollo y Pruebas

Una vez desarrollado el prototipo, se planteó que servicios REST debían de ser creados, de tal manera que el módulo cumpliera con su objetivo; por lo tanto, se obtuvieron 2 endpoints, ya que sus funciones eran totalmente centradas y concisa; por lo que los casos de uso y los métodos en el repositorio también se centraban únicamente en recibir los parámetros, buscar la información y actualizarla de acuerdo con la información enviada.

Figura 85

Endpoints Sprint 7.

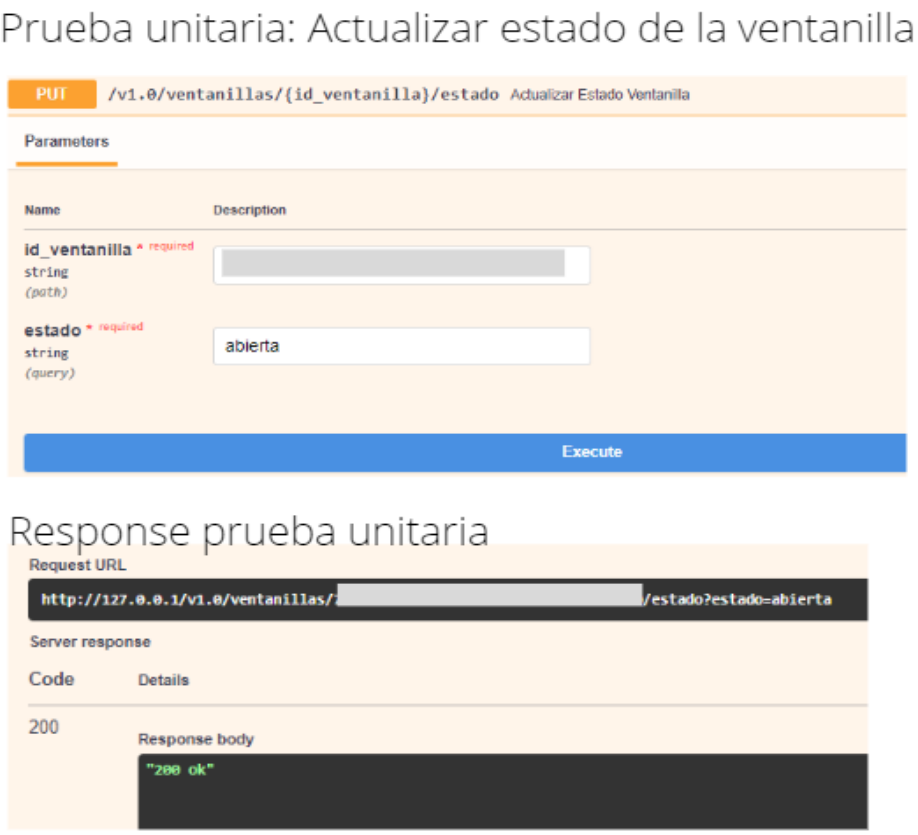


Nota: Fuente propia

Referente a las pruebas unitarias, únicamente se enviaban los parámetros, por ejemplo, de la ventanilla, siendo estos su identificador y el estado que quería actualizar (abierta o cerrada) y se esperaba un response estandarizado, tal como se demuestra a continuación:

Figura 86

Evidencia Prueba Unitaria Sprint 7.



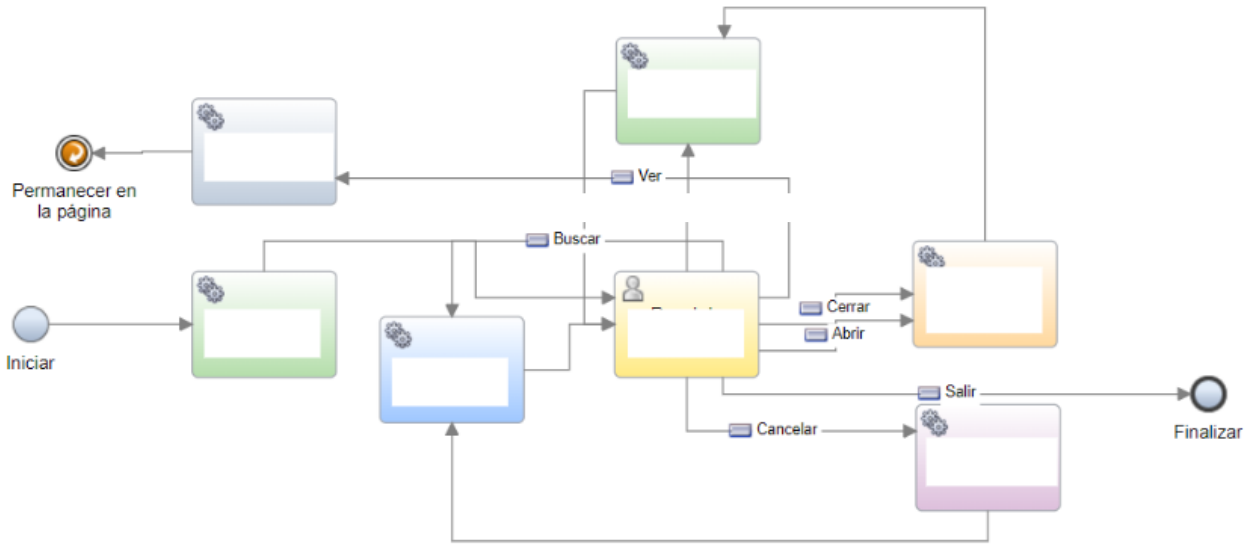
Nota: Fuente propia

Dando continuidad al proceso, se realizan las actividades necesarias para que la herramienta pudiera consumir estos dos endpoints para posteriormente crear el workflow con todos sus componentes. En este sprint, era necesario generar variables que incluyeran un estado predeterminado respecto a la ventanilla, pues no se consideraba trabajo del usuario, por ejemplo, seleccionar o escribir que estado quería en su ventanilla, ya que se consideraría tedioso; de igual forma se decidió consumir endpoints ya construidos con anterioridad, esto para facilitar el entendimiento del módulo, algunos de estos endpoints son los de: Listar sedes, y con base a la sede seleccionada permitirle visualizar al usuario las distintas salas que contenía esa sede, como

también las ventanillas halladas en cada sala, para que así pudiera asegurarse de que la información que pretendía actualizar fuera la correcta.

Figura 87

Flujo de Trabajo Tablero de Control.



Nota: Fuente propia

Figura 88

Evidencia Estado Predeterminado Por Estado de Ventanilla.

Estado predeterminado en los botones para la administración de ventanillas

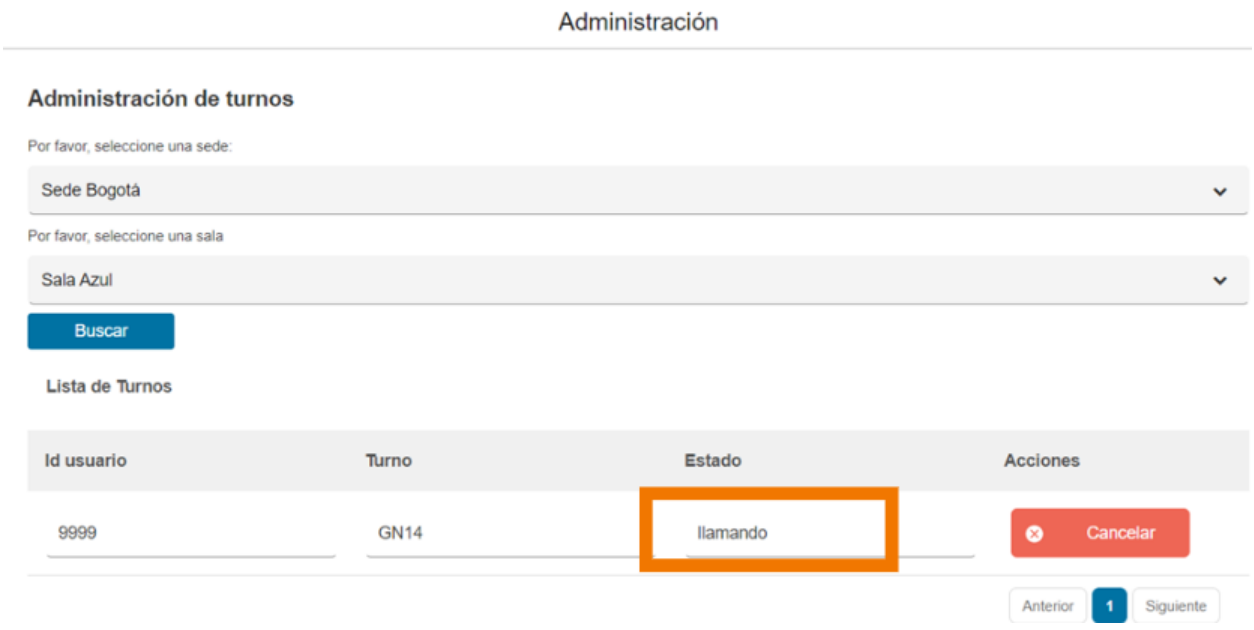
```
page.ui.get("layoutEstadoVentanilla").setText("cerrada");

page.ui.get("layoutEstadoVentanilla").setText("abierta");
```

Nota: Fuente propia

Finalmente, se realizan pruebas de integración con todos los componentes involucrados. Una de las pruebas realizadas fue ingresar a la sección de cancelar un turno, donde, se deberían listar todos los turnos existentes de acuerdo a la sede y sala que fueran seleccionadas, dar clic en un botón para cancelar un turno específico y validar que se modificara en tiempo real el estado de este, como también en la base de datos.

Figura 89

Prueba de Integración Sprint 7: Listado de Turnos Actuales Previo a Actualización.



Nota: Fuente propia

Figura 90

Prueba de Integración Sprint 7: Actualización Estado de Turno Posterior a Prueba.

Nota: Fuente propia

8.8. Sprint 8. Módulo Tablero de Indicadores

El proyecto actual abarcó diversas entidades siendo estas las sedes o las salas que, como bien es sabido, pueden encontrarse en varias zonas dependiendo la cantidad de usuarios que requieran atención; es por esta razón que para la toma de decisiones frente a posibles mejoras se decidió desarrollar un módulo para el administrador, de tal forma que pueda visualizar estadísticas de los turnos, servicios e incluso un tiempo de espera promedio, brindando la capacidad de filtrar su información para tener rangos específicos en caso de ser necesario. De acuerdo con el contexto anterior, se definió como objetivo del sprint 8 el desarrollar un módulo de estadísticas o indicadores, para de esta forma facilitarle al usuario el estado de sus procesos iniciales. Algunas de las actividades realizadas durante este sprint se reflejan en la siguiente evidencia referente al Sprint Backlog;

Figura 91

Sprint Backlog 8.

Nota: Fuente propia

8.8.1. Análisis

Para llevar a cabo el correcto desarrollo del módulo fue necesario ratificar que secciones o procesos eran más útiles para el usuario final, teniendo en cuenta que se contaba con gran cantidad de información y no toda era importante en este ámbito; por ende, se concluyó que las secciones más relevantes serían: Los turnos, categorizados por estados y filtrados por fecha, sala o sede, los servicios, donde se buscó que por sala y sede se contaran el total de turnos atendidos por dicho servicio, también brindando la posibilidad de generar un filtro en caso de ser necesaria la profundización y finalmente el tiempo de espera promedio por sala, sede y servicio, permitiendo a su vez los filtros para la información. Por otra parte, se corroboró si era necesario crear una estructura de datos nueva, o una colección en la base de datos, sin embargo, se afirmó

que la información que ya había sido establecida en los sprints anteriores era óptimo para los procesos a realizar.

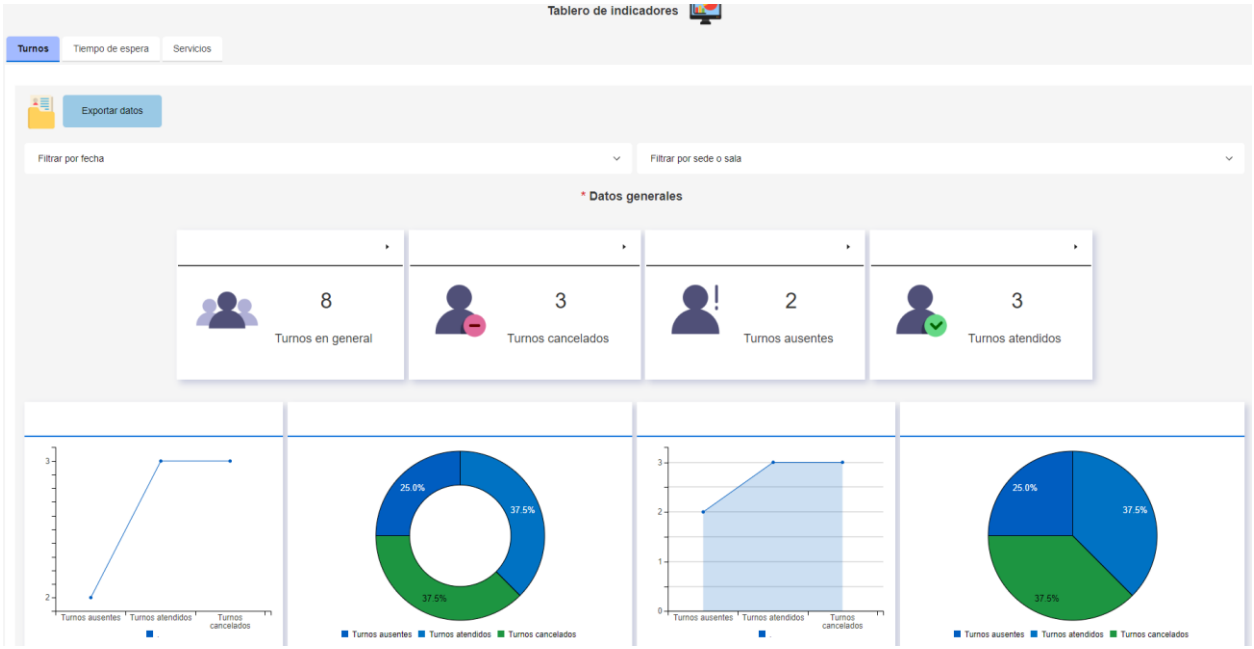
8.8.2. Diseño

Una vez confirmadas las secciones que debían realizarse, se diseñaron los prototipos aptos para la correcta visualización de la información por parte del usuario, donde cada uno de ellos contaba con diferentes diseños, evitando un estilo plano. Para la sección de turnos se optó por utilizar cards para dar a conocer el total de ellos por su estado (atendidos, cancelados, ausentes y en general) esto, sin importar la sede ni la sala; además se utilizaron gráficas de barras, circulares y multipropósito para que fuera más simple validar que categoría contaba con mayor cantidad de turnos. En cuanto a la sección de tiempo de espera se implementaron únicamente cards que contenían datos generales de la sala, sede y servicio de donde fue obtenido y calculado el tiempo de espera para que fuera más fácil su entendimiento. Finalmente, la sección de servicios contaba con una tabla, pues al evidenciar gran cantidad de datos sin un filtro inicial se complicaba el entendimiento de los mismos, por lo tanto, se eligió esta opción. Como acotación final de esta etapa, se resalta que cada sección contaba con paneles contraíbles que contenían datos para filtrar la información y una vez realizado el filtro se configuraban cards que le permitieran saber que datos había seleccionado. A continuación, se evidencian los prototipos obtenidos durante el diseño del sprint 8:

Figura 92

Prototipo Sección Turnos-Tablero de Indicadores.

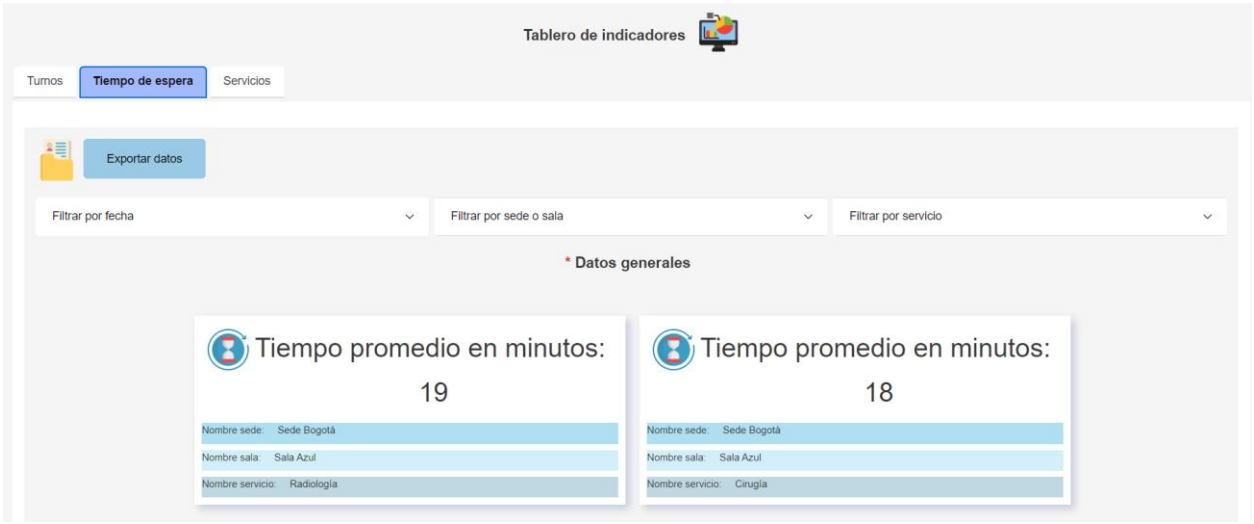
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 93

Prototipo Sección Tiempo de Espera-Tablero de Indicadores.

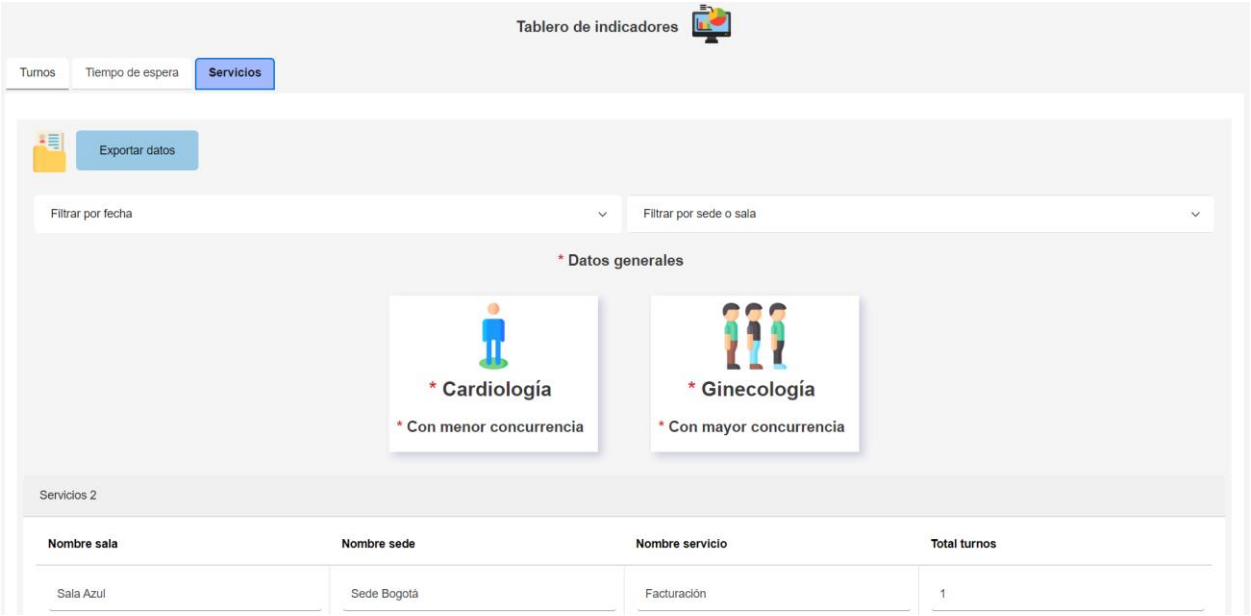


Nota: Fuente propia

Figura 94

Prototipo Sección Servicios-Tablero de Indicadores.

Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

8.8.3. Desarrollo y Pruebas

Para que fuera posible contar con datos y filtros correctos, se desarrollaron 5 endpoints, estos enfocados en las diferentes entidades mencionadas con anterioridad. Cada uno de estos servicios solicitaba parámetros diferentes y así mismo su response era distinto, pues no siempre se retornaban los mismos datos; por esta razón se buscó que estos endpoints fueran dinámicos, o de lo contrario se requería la realización de más de ellos, lo cual no se consideró óptimo, teniendo en cuenta que el flujo de trabajo incrementaría de acuerdo con la cantidad de tareas de servicio que debían ser llamadas. Para dicho dinamismo fue requerida únicamente una consulta a la base de datos bien estructurada (por cada endpoint), esta llamada por medio de un método en el repositorio, un caso de uso y un servicio REST. Luego de la programación de este paso, se realizaron pruebas unitarias, para validar tanto el correcto manejo de errores como la visualización de la información esperada de acuerdo con la entidad.

Figura 95

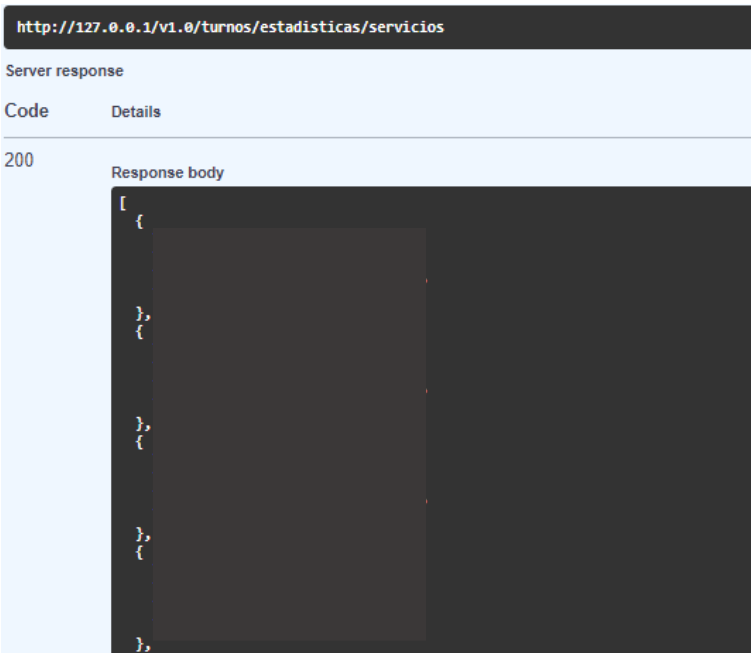
Endpoints Sprint 8.

Nota: Fuente propia

En este caso se decidió evidenciar la prueba unitaria del filtro por servicios, donde todos sus parámetros eran opcionales y, de acuerdo con el que fuera necesario, desde ninguno hasta el uso de todos ellos se esperaba un array o un objeto.

Figura 96

Evidencia Prueba Unitaria Sprint 8.

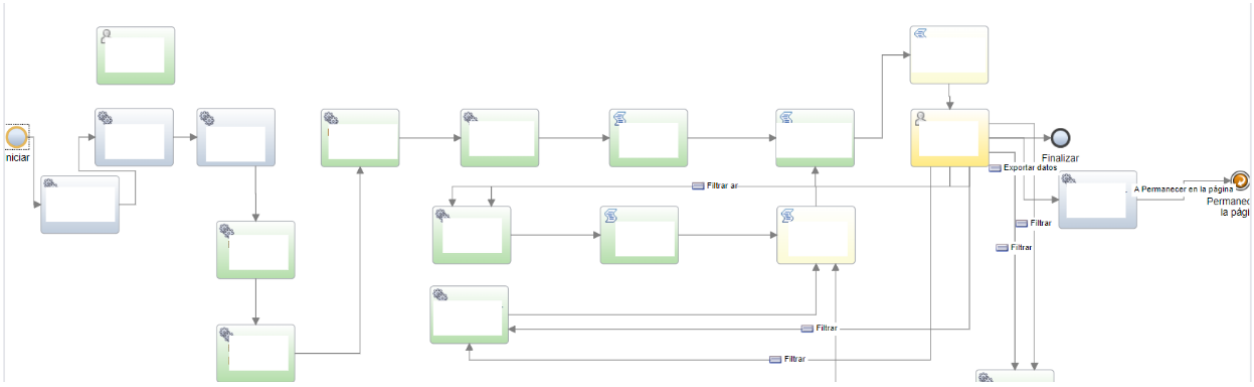


Nota: Fuente propia

Una vez completadas las pruebas, se creó el flujo de trabajo en la herramienta, invocando uno a uno los servicios, e integrándolos respectivamente al coach donde estaban alojadas las 3 secciones para el tablero de indicadores, creando las variables de entrada, salida y privadas para correlacionar la información con los objetos de negocio, enlazando la parte gráfica con los servicios y finalmente realizar las pruebas de integración, dando un resultado óptimo para dar por terminado el sprint.

Figura 97

Flujo de Trabajo Sprint 8.

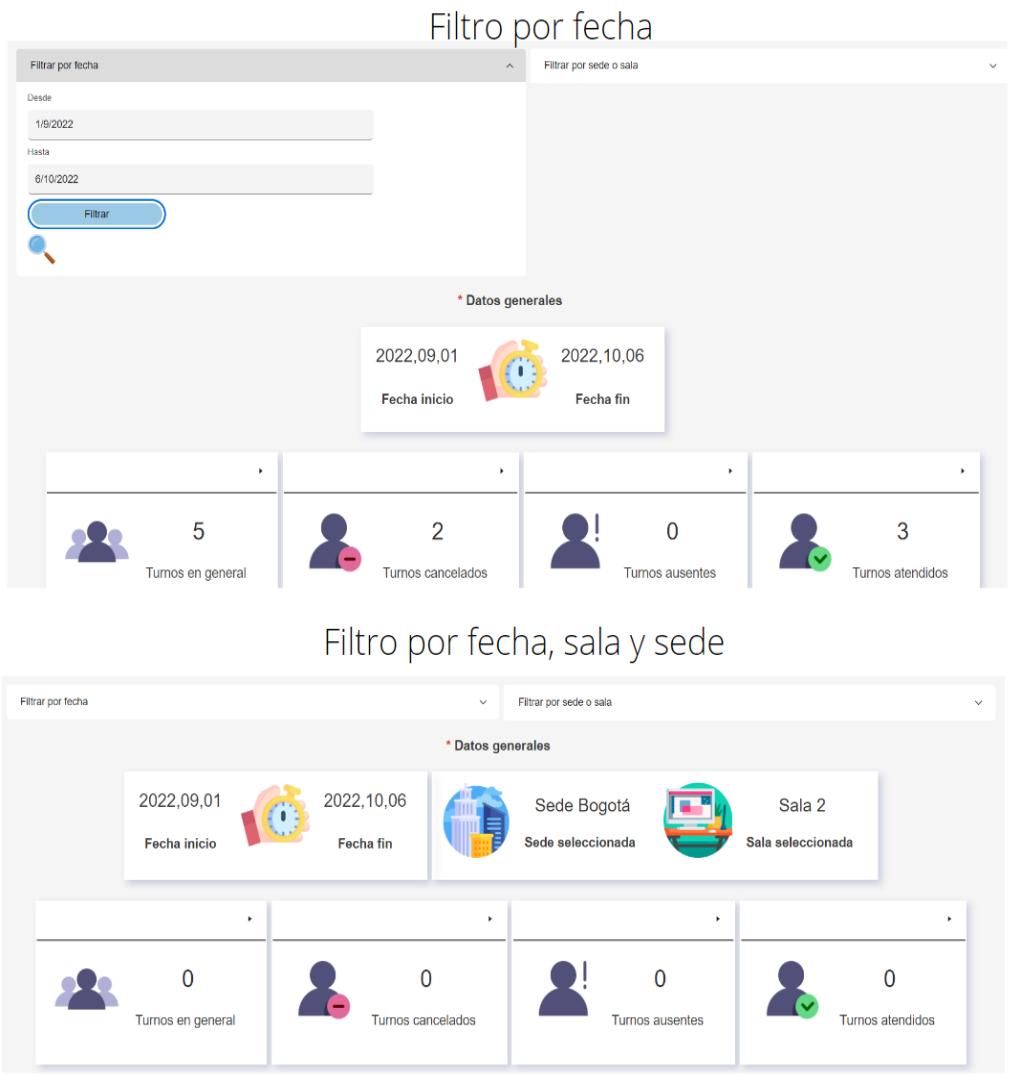


Nota: Fuente propia

Una de las pruebas de integración realizadas, se centró en ejecutar el módulo e ingresar a la sección de turnos, validar que los datos iniciales fueran correctos, filtrando por rangos de fechas que, si contenían turnos y a la vez fechas que no, luego filtrando únicamente por sala, posteriormente por sede y para concluir se realizó un filtro por todas las opciones al mismo tiempo.

Figura 98

Evidencia Prueba Unitaria Sprint 8.



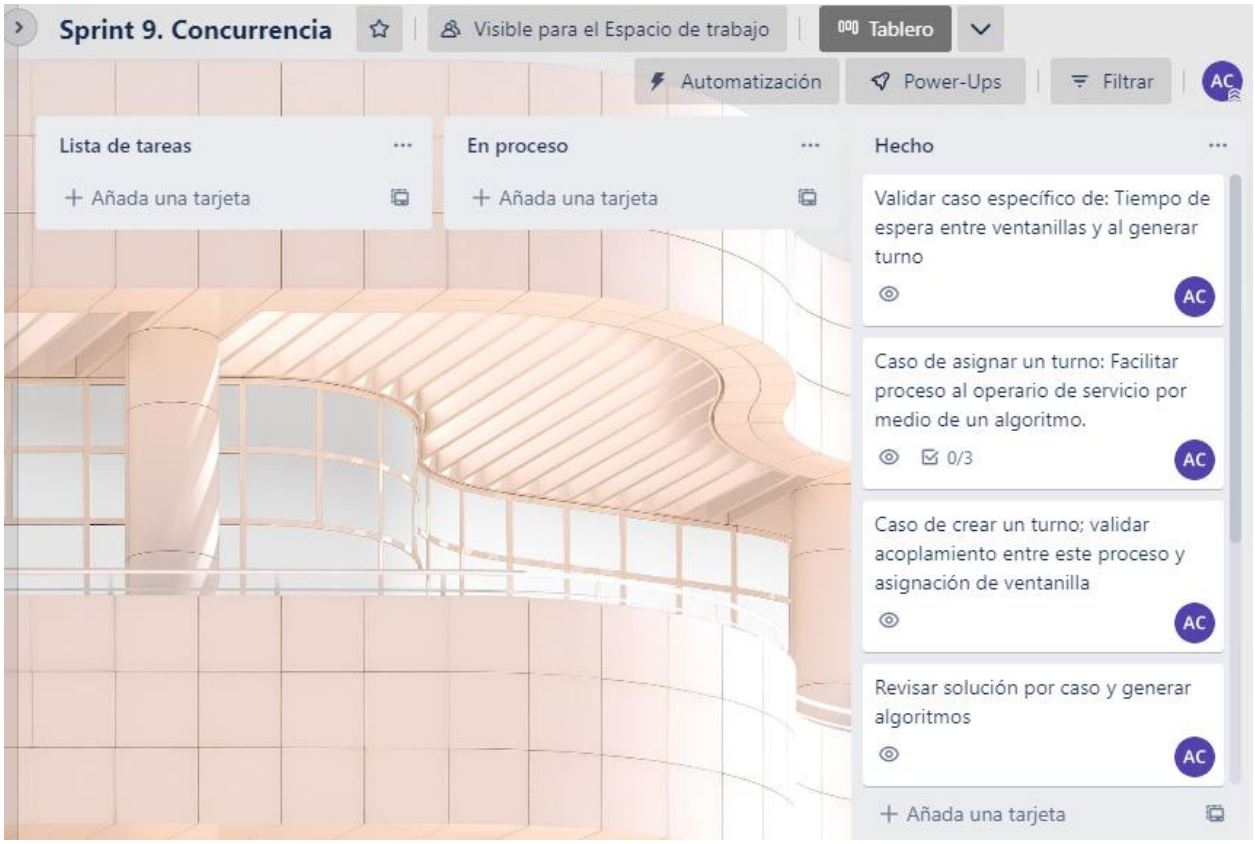
Nota: Fuente propia

8.9. Sprint 9. Concurrencia

Fue claro que, al desarrollar un sistema de gestión de turnos a gran escala, este contaría con concurrencia a nivel masivo, por lo que fue importante que, una vez desarrollados los módulos con los que contaría el sistema, se generó un esfuerzo en los algoritmos con más utilidad en el proyecto; para así permitir un mejor rendimiento en los procesos; siendo este el objetivo del sprint 9. Como complemento del planteamiento anterior, se decidió evidenciar algunas de las actividades a realizar en el Sprint Backlog 9:

Figura 99

Sprint Backlog 9.



Nota: Fuente propia

8.9.1. Análisis

Antes de desarrollar las tareas establecidas, fue pertinente validar como se podrían optimizar los algoritmos ya existentes, de tal forma que su operatividad fuera la correcta.

Para esto se analizaron los siguientes casos:

Tiempo de espera. El sistema, al estar constantemente a la espera de recibir solicitudes de un turno por diversos usuarios en diversas entidades, debía calcular el tiempo de espera tomando la información suministrada, sin embargo, esto podría estarse ejecutando en zonas diferentes, por lo que, el filtrar únicamente un documento por medio de una consulta no era correcto. Por esta razón se decidió que, al momento de que un usuario solicitara un turno, el

sistema validara si existía un documento con los mismos identificadores de la sala, sede y el servicio, y si no existía lo generara, con la información del turno y los datos del tiempo, y los turnos atendidos en cero, mientras que el puesto debía aumentarse a uno; pero, si por lo contrario existía, este debía encargarse de filtrar el documento, traer la información, promediar el tiempo y entregarle al usuario la información procesada.

Asignación de un turno. En un principio, se establecía que, una vez creado el turno, este debía ser asignado a una ventanilla directamente, filtrando por toda la información y teniendo en cuenta el servicio que estaban atendiendo las ventanillas, por lo que el sistema tardaba en procesar toda esta información, pues la ventanilla solo podía mantener un turno asignado a la vez, generando que los turnos quedaran en espera por mucho tiempo, incluso desde su creación, por lo que se consideraba ineficiente. Para manejar correctamente este algoritmo, se decidió que los turnos serían asignados al momento en que el operario del servicio diera apertura a la ventanilla, o también cuando nuevamente estuviera disponible, por lo que, se decide que en estos estados específicos, el sistema se debía dirigir a la colección de turnos próximos, validara por orden de creación cual pertenece a su servicio, le muestre la información del turno al operario, luego se muestre en pantalla que está siendo llamado y finalmente lo pueda atender.

Creación de un turno. Por el análisis anterior, también se consideró pertinente aplicar un mejor algoritmo a la creación de un turno, pues al buscar una ventanilla y no encontrar, el sistema entraría en un bucle hasta encontrar una ventanilla libre de acuerdo con el servicio, por lo que el usuario estaría esperando su turno por mucho tiempo, siendo este un proceso erróneo. Para esto, se validó que, el turno debía ser creado, debía mostrar el último puesto que había sido atendido de acuerdo a su servicio y así mismo lo debía enviar a la colección de turnos próximos, ubicándolo de acuerdo a su fecha de creación; esto para que el usuario recibiera su ticket o la

notificación de la correcta creación de su turno a los segundos de haber procesado su solicitud, y no ralentizando el proceso tanto de dicho usuario como de los demás usuarios que requerían de solicitar el turno.

8.9.2. Diseño

Esta etapa no fue requerida en este sprint, pues las actividades se centraban en el desarrollo y análisis, además no requería de colecciones adicionales a las existentes; por ende, se dio paso a la siguiente fase.

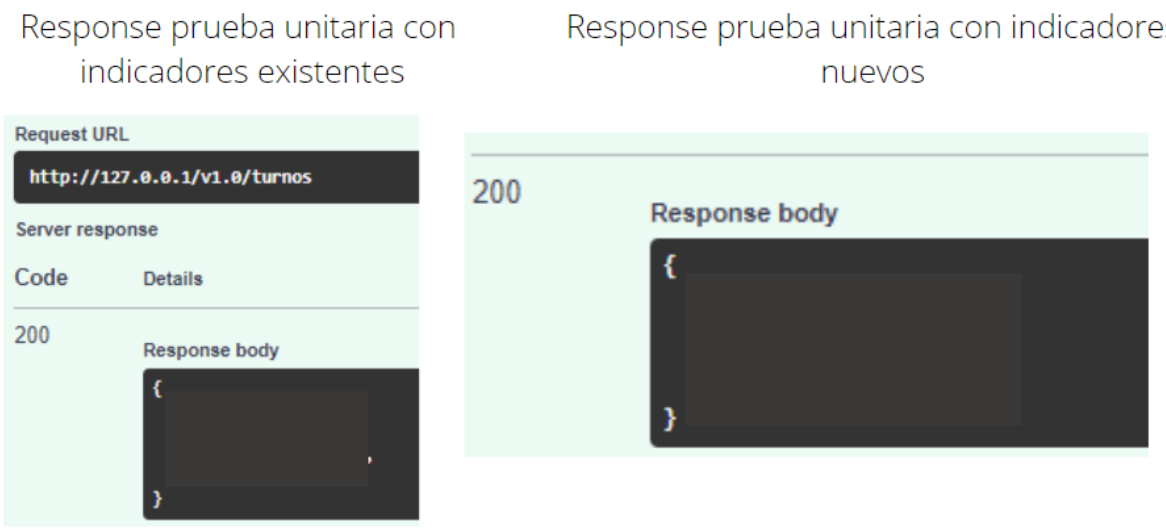
8.9.3. Desarrollo y Pruebas

Para esta etapa, fue necesario replantear algunos casos de uso y procesos hallados en los endpoints ya existentes, pues debían ser mejorados con fines enfocados a brindar un producto de calidad. Así que, se inició con la concurrencia del tiempo de espera, modificando el endpoint de crear turno, así mismo fue necesario crear un caso de uso enfocado a agregar un nuevo documento con los parámetros que se le enviaran, asumiendo que podía existir la posibilidad de que el documento aún no existiera, así mismo el repositorio establecido para esta colección, debía contar con un nuevo método, donde recibiera los datos necesarios, para enviarlos a la consulta realizada.

Una de las pruebas unitarias realizadas en este primer caso, se centró en crear un turno con identificadores que ya existieran en un documento de la colección, para validar si retornaba un error, o de lo contrario la información del turno; también se enviaron identificadores nuevos para corroborar tanto la información del turno, como la correcta creación del documento en la base de datos. Algunas evidencias se ilustran a continuación:

Figura 100

Prueba Unitaria Sprint 9: Tiempo de Espera.

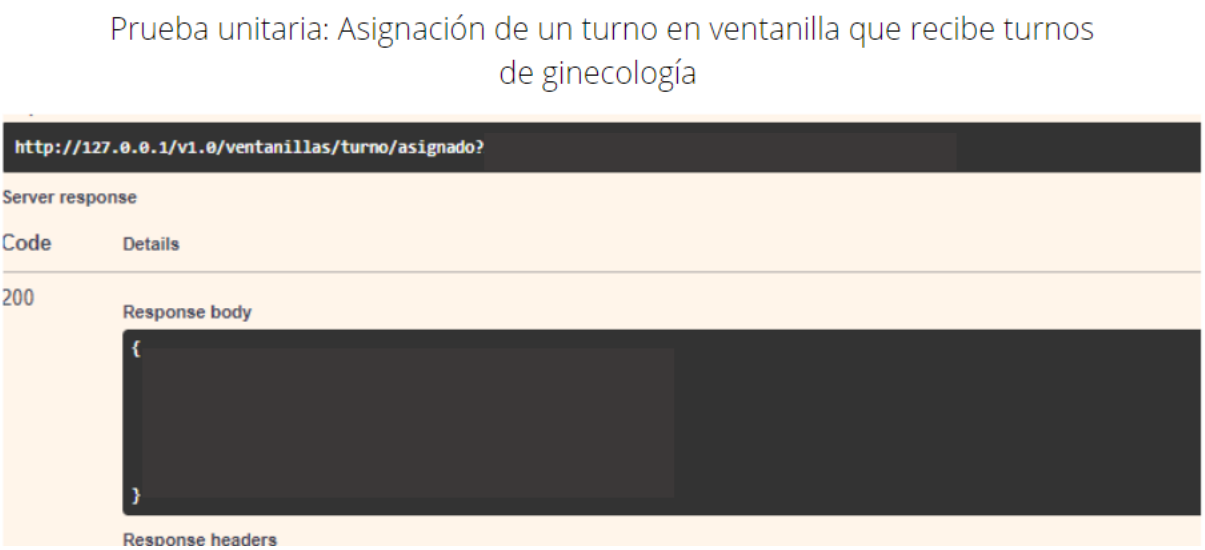


Nota: Fuente propia

Frente a la concurrencia de la asignación de un turno, fue necesario replantear uno a uno los procesos que se invocaron en el endpoint en cuestión, así como modificar el flujo de trabajo en la herramienta, ya que, al momento de crear el turno, también se invocaba el endpoint para asignar el turno a la ventanilla, lo cual ya no era válido de acuerdo al análisis; por otra parte, en el workflow de la inicialización de las ventanillas, debía de invocarse el servicio de asignación, esto luego de dar apertura, como también al momento de estar nuevamente disponible. Para una de las pruebas unitarias se debió validar como una ventanilla específica recibía un nuevo turno al momento de invocar el servicio, obteniendo como resultado:

Figura 101

Prueba Unitaria Sprint 9: Asignación de un Turno a Ventanilla.



Nota: Fuente propia

Finalmente, esta etapa se enfocó en el último caso establecido: Creación de un turno. Sin embargo, este debía ser manipulado en los algoritmos anteriores, por lo que ya había sido modificado simultáneamente para las pruebas y para el correcto funcionamiento, tanto de los servicios anteriores como de éste, por lo que, únicamente se validaron sus pruebas unitarias, donde, por ejemplo, se creaba el turno y se esperaba una respuesta, tal como la siguiente:

Figura 102

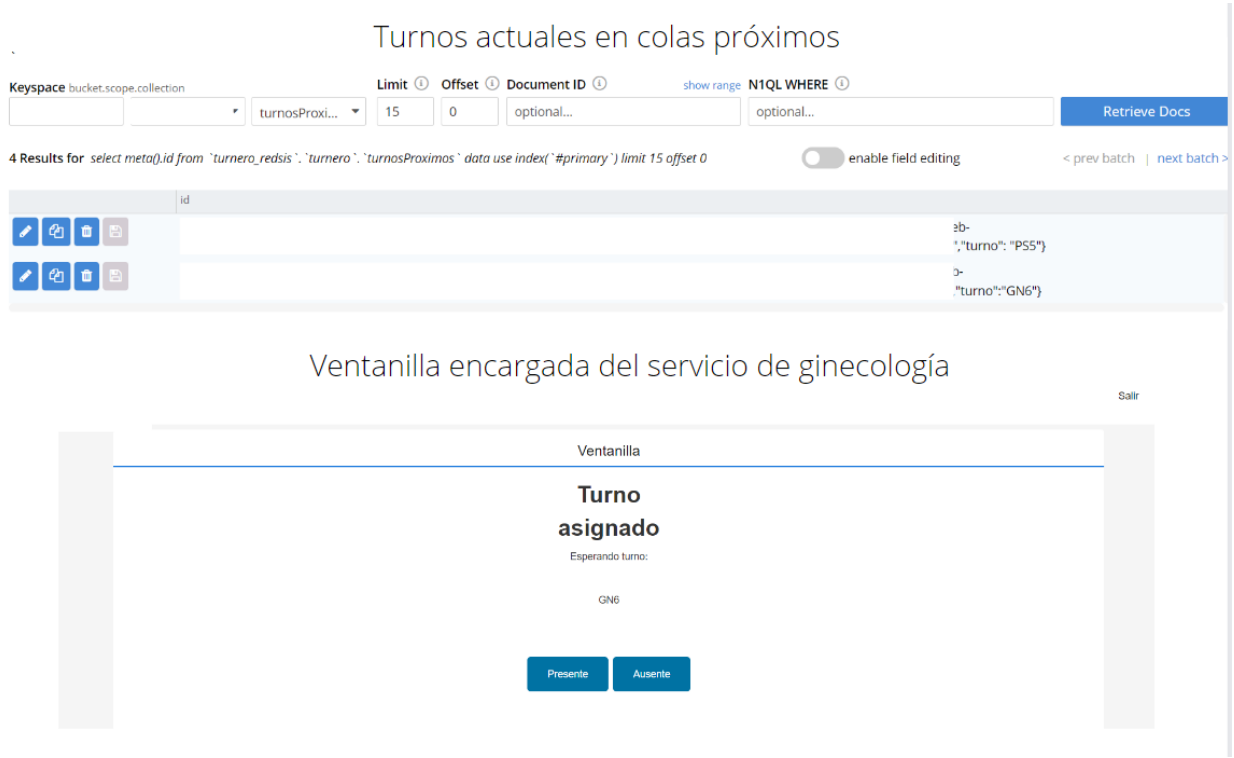
Prueba Unitaria Sprint 9: Crear Turno.

Nota: Fuente propia

En este sprint, a medida que se terminaba cada uno de los casos planteados en el análisis se iban modificando los flujos de trabajo en la herramienta, así como también se modificaban los servicios externos gracias a la nueva documentación generada en swagger, por lo que únicamente se necesitaban realizar las pruebas de integración. Para evidenciar una de las pruebas realizadas, se ejecutó el módulo del operario de la ventanilla, validando desde la base de datos que turno debía tomar, si este era establecido correctamente, así como también se validaba que camino tomaba el flujo de trabajo en caso de no encontrar un turno en el momento. Algunas de los resultados de estas pruebas se ilustran a continuación:

Figura 103

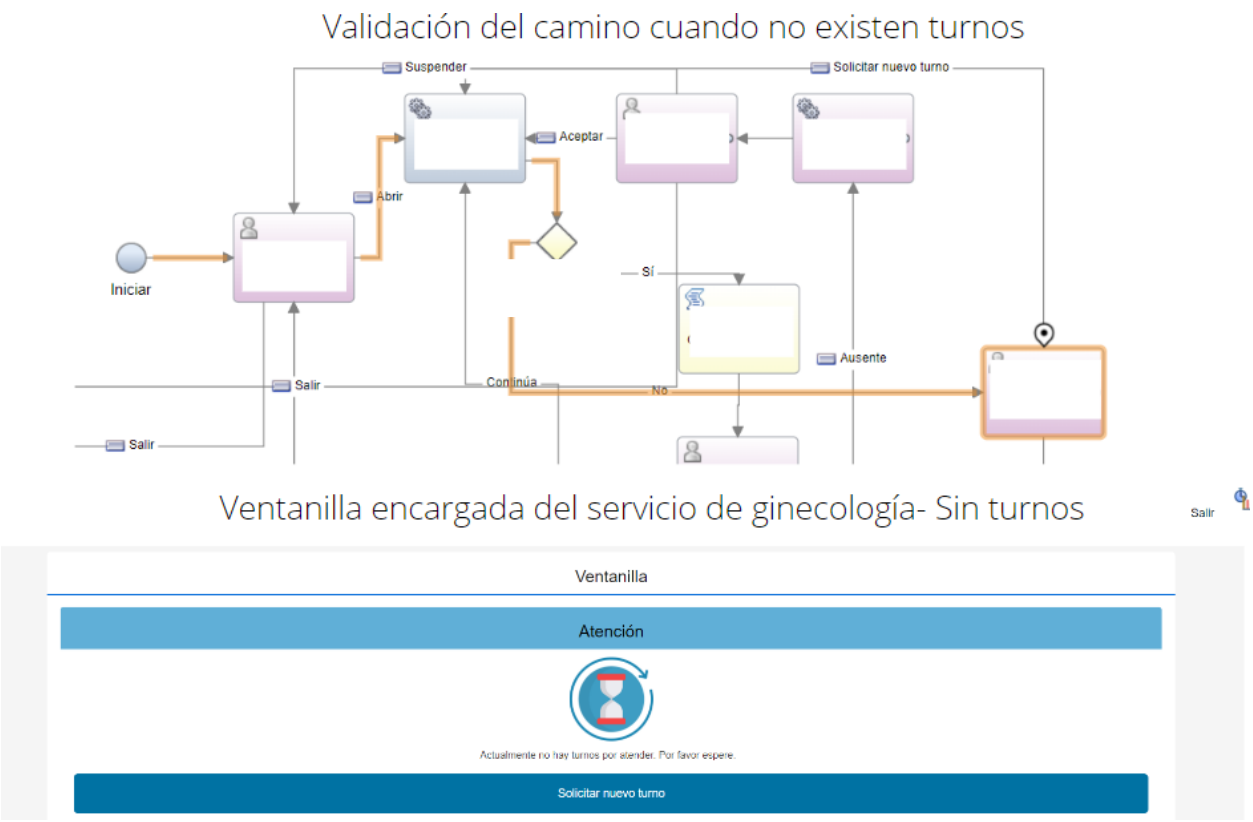
Prueba de Integración Sprint 9- Turno Existente.



Nota: Fuente propia

Figura 104

Prueba de Integración Sprint 9- Turno no Disponible.



Nota: Fuente propia

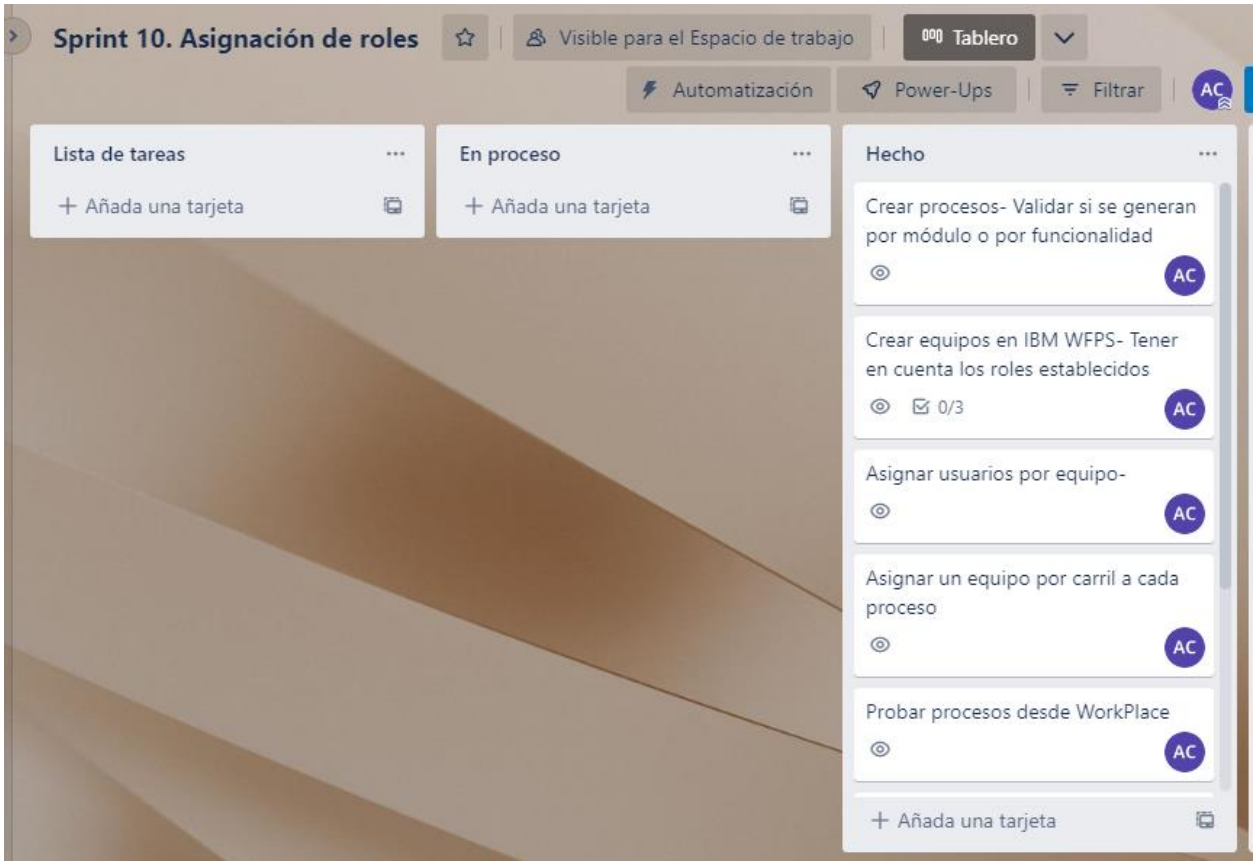
Al validar el cumplimiento del objetivo del sprint, teniendo en cuenta el desarrollo de las actividades, las pruebas y el análisis, se da por concluido, sin ningún tipo de observación o mejora.

8.10. Sprint 10. Asignación de Roles

El objetivo de este sprint buscaba establecer los procesos para cada rol que ya se ha mencionado, teniendo claro que estos son: Administrador, operario de la ventanilla y usuarios finales. Para validar que actividades se debían comprender, se tuvo en cuenta el siguiente sprint backlog:

Figura 105

Sprint Backlog 10.



Nota: Fuente propia

8.10.1. Análisis

Este sprint fue de los últimos en realizar gracias a la facilidad que brindaba IBM Workflow Process Services frente a la asignación de roles, ya que al crear un proceso se podían establecer carriles donde se asignaban equipos de tal forma que, si un integrante no estaba asignado al equipo, simplemente no podría visualizar esta tarea desde Workplace (sección dedicada al usuario, para visualizar su espacio de trabajo y las tareas que tienen pendientes, las terminadas y las vencidas). Fue en este caso donde se decidió que debían crearse un total de 6 procesos, estos independientes de funciones, para que fueran de fácil manejo para cada rol. Por otra parte, fue establecido que era necesario crear un total de 4 equipos, los cuales se evidenciarán en las siguientes etapas.

8.10.2. Diseño

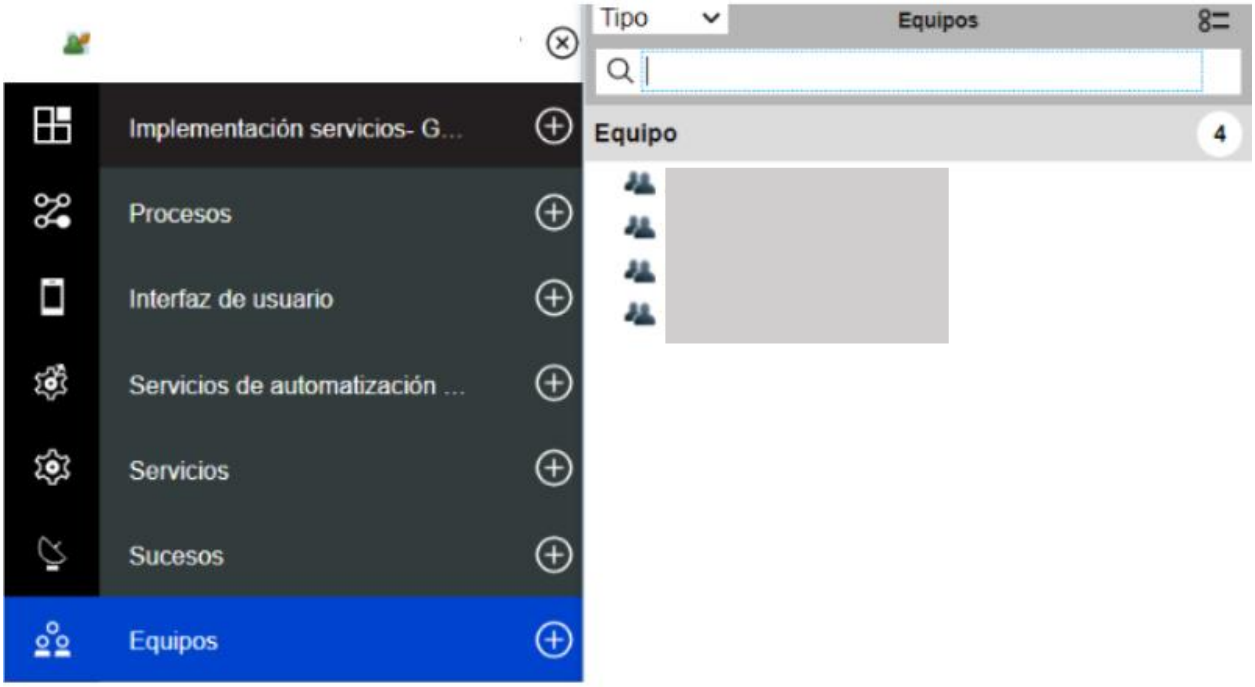
Esta etapa no fue necesaria en este sprint, esto debido a la finalización de todos los prototipos y colecciones, por lo que únicamente se requería de un análisis, desarrollo y pruebas. De esta manera se dio paso al desarrollo.

8.10.3. Desarrollo y Pruebas

Al saber que procesos y equipos debían ser implementados se decidió que el desarrollo se centraría en la creación de estos. En un principio se inició con los equipos, creándose desde el mismo IBM Workflow Process Services, obteniendo:

Figura 106

Evidencia Creación de Equipos.



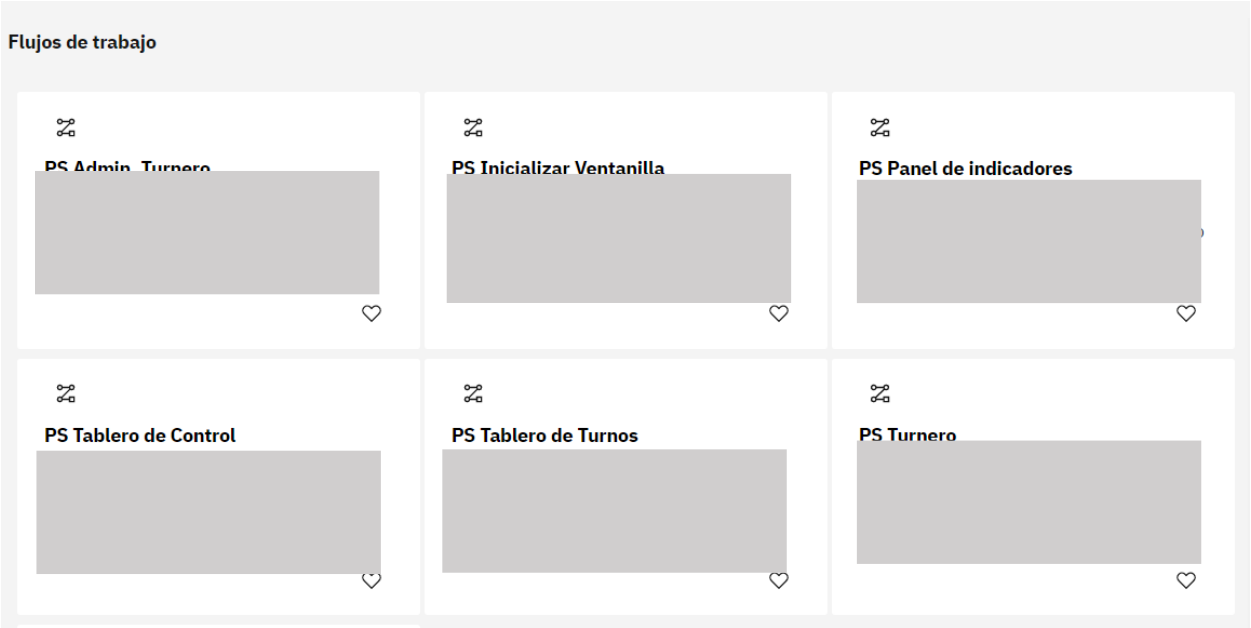
Nota: Fuente propia

Una vez completados los equipos, se fue creando un proceso a la vez, para poder relacionar su carril de acuerdo con su función y a su equipo, invocar al Client Human Services o

módulo desde el proceso y finalmente ejecutarlo desde el Workplace, tal como se evidencia a continuación:

Figura 107

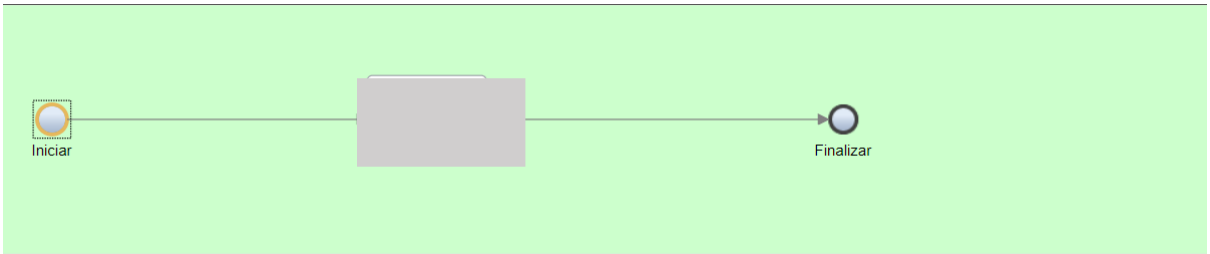
Evidencia Creación de Procesos.



Nota: Fuente propia

Figura 108

Evidencia Proceso en Carril Específico.

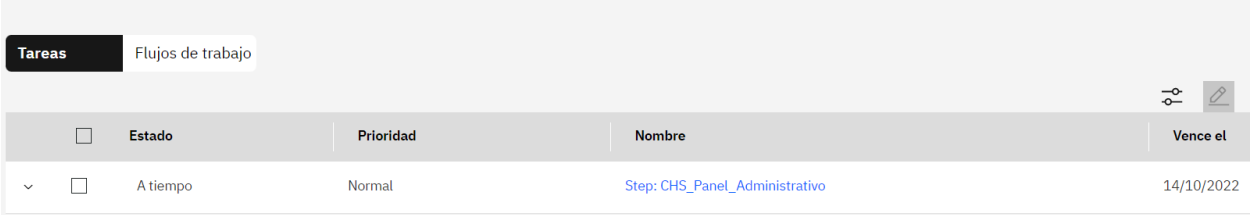


Nota: Fuente propia

Para saber si los procesos estaban funcionando correctamente, fue necesario probar la ejecución de estos desde el Workplace, comprobando de esta forma que el sistema en general ya estaba completo en su primera versión.

Figura 109

Proceso Ejecutándose en el Workplace.



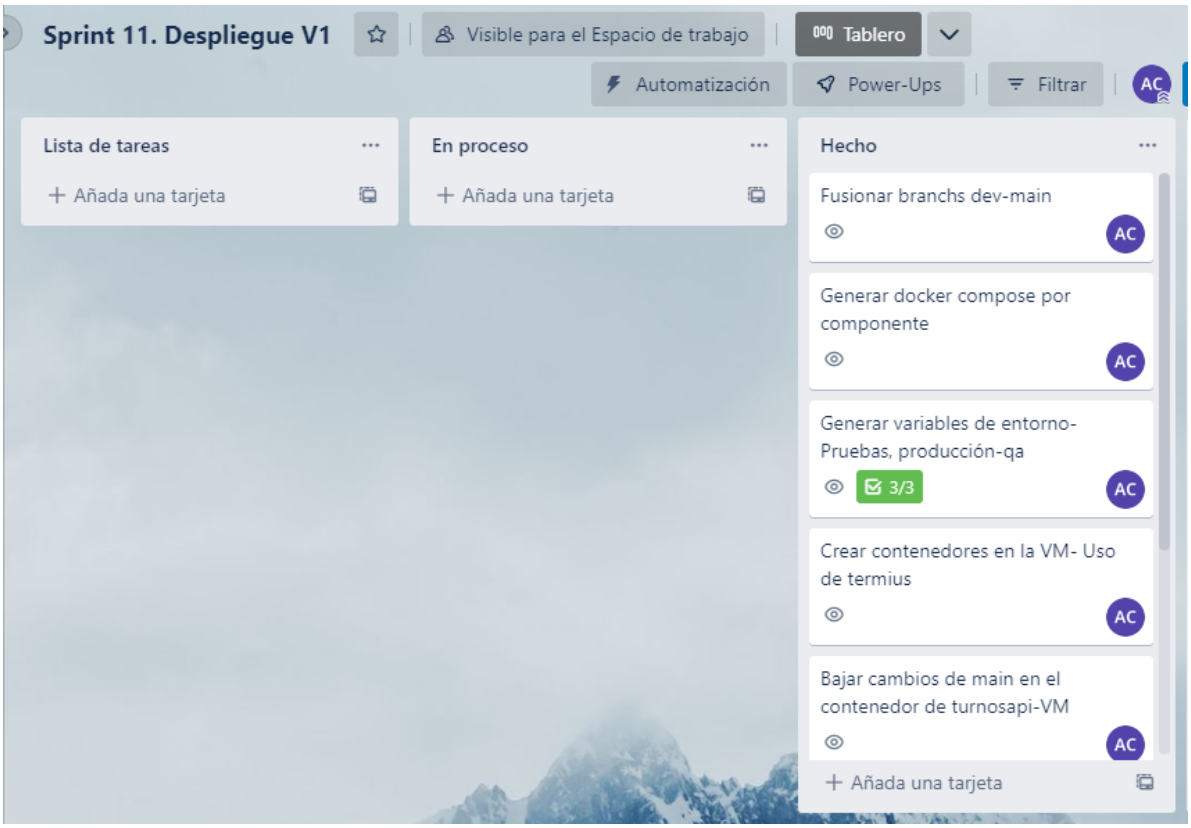
Nota: Fuente propia

8.11. Sprint 11. Despliegue de la Versión 1

Al corroborar que el sistema estaba funcionando como se esperaba en su primera versión, se decidió desplegarlo donde lo estableció la empresa, siendo este el objetivo del último sprint del proyecto. Para esto se tuvo en cuenta el siguiente sprint backlog:

Figura 110

Sprint Backlog 11.



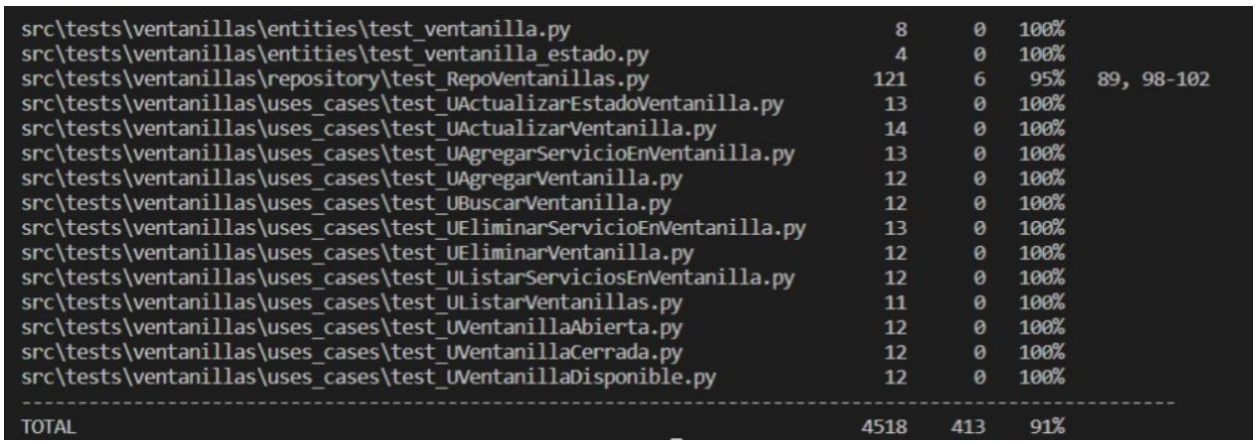
Nota: Fuente propia

8.11.1. Análisis

En esta etapa se decidió ampliar los fundamentos y conocimientos sobre azure, termius y docker, para llevar a cabo dicho despliegue, teniendo en cuenta que el servidor en el cual se debía trabajar era delicado, pues en él se hallaban varias máquinas virtuales pertenecientes a la empresa. Por otra parte, se analizó el porcentaje de cumplimiento de las pruebas unitarias en general, pues sin esto no se podía desplegar el sistema, porque indicaba que no estaba cumpliendo con el rango mínimo, sin embargo, se obtuvo el siguiente resultado, que fue considerado como favorable para seguir con el proceso:

Figura 111

Coverage Proyecto- Python.



src\tests\ventanillas\entities\test_ventanilla.py	8	0	100%	
src\tests\ventanillas\entities\test_ventanilla_estado.py	4	0	100%	
src\tests\ventanillas\repository\test_RepoVentanillas.py	121	6	95%	89, 98-102
src\tests\ventanillas\uses_cases\test_UActualizarEstadoVentanilla.py	13	0	100%	
src\tests\ventanillas\uses_cases\test_UActualizarVentanilla.py	14	0	100%	
src\tests\ventanillas\uses_cases\test_UAgregarServicioEnVentanilla.py	13	0	100%	
src\tests\ventanillas\uses_cases\test_UAgregarVentanilla.py	12	0	100%	
src\tests\ventanillas\uses_cases\test_UBuscarVentanilla.py	12	0	100%	
src\tests\ventanillas\uses_cases\test_UEliminarServicioEnVentanilla.py	13	0	100%	
src\tests\ventanillas\uses_cases\test_UEliminarVentanilla.py	12	0	100%	
src\tests\ventanillas\uses_cases\test_UListarServiciosEnVentanilla.py	12	0	100%	
src\tests\ventanillas\uses_cases\test_UListarVentanillas.py	11	0	100%	
src\tests\ventanillas\uses_cases\test_UVentanillaAbierta.py	12	0	100%	
src\tests\ventanillas\uses_cases\test_UVentanillaCerrada.py	12	0	100%	
src\tests\ventanillas\uses_cases\test_UVentanillaDisponible.py	12	0	100%	
TOTAL	4518	413	91%	

Nota: Fuente propia

De igual manera, se analizó toda la operatividad a nivel general, de acuerdo a todas las pruebas realizadas durante el proyecto, y, a pesar de que el sistema estaba encaminado a una culminación favorable, se estableció el estudio de una nueva versión, está basada en el análisis que se realizara frente al sistema con el que contaba una clínica específica que permitiría el estudio del mismo, sin embargo, el alcance del proyecto únicamente determina el establecimiento de un posible versionamiento, más no la ejecución de este.

8.11.2. Diseño

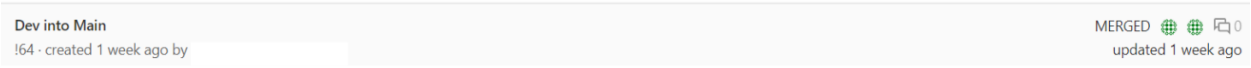
El diseño en este sprint no fue necesario, por lo que se dio paso a la etapa de desarrollo.

8.11.3. Desarrollo

Cabe destacar que, durante todo el proyecto, por cada sprint se generaba una rama o branch en gitlab para que cada integrante del equipo trabajara en ella, y al momento de dar por concluido el trabajo, estas ramas se fusionaban con dev, para mantenerla actualizada y evitar errores futuros, asumiendo que el código pasaba por diversas pruebas para constatar que si estaba funcionando a cabalidad. Al culminar la primera versión, la rama dev fue fusionada con main, pues esta era la encargada de producción.

Figura 112

Merge Dev Into Main.

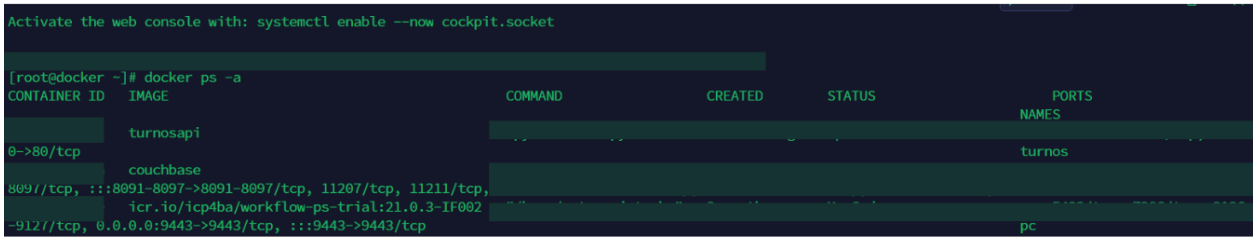


Nota: Fuente propia

Una vez realizado esto, fue necesario crear contenedores dentro de una máquina virtual determinada en el servidor de la empresa, para que con únicamente la ruta y la conexión por vpn se pudiera acceder al sistema. Se crearon un total de 3 contenedores en el servidor, siendo estos: El contenedor para los endpoints (código de Python con fastAPI), contenedor para el IBM Workflow Process Services y finalmente el contenedor para la base de datos. Todos estos procesos se llevaron a cabo por medio de un docker compose, lo que permitía el inicio de la ejecución por medio de un único comando.

Figura 113

Contenedores en VM.

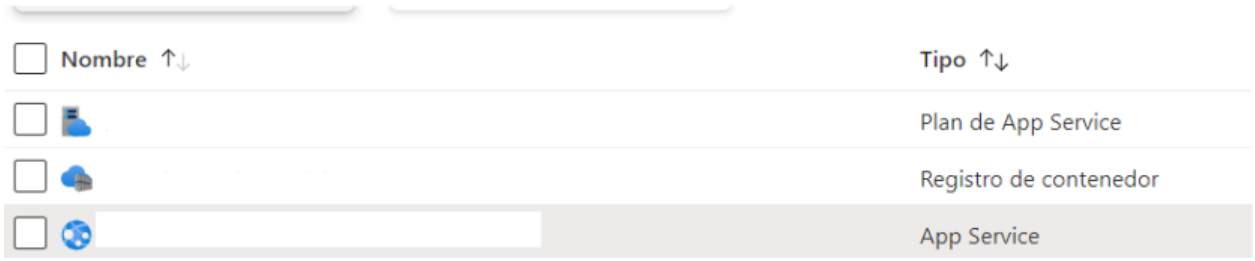


Nota: Fuente propia

Por otra parte, y como solicitud de la empresa, se debían almacenar los endpoints en azure, específicamente en un App Service, por lo que se realizó el debido proceso, teniendo en cuenta que debían generarse archivos para que al ejecutarlos esta App Service se iniciara o detuviera cuando fuera necesario. Cabe resaltar que estos endpoints se almacenaron allí para que cualquier cliente que validara la capacidad de integración del sistema y buscara una forma de integrarse pudiera evidenciar que los endpoints podrían ser los desarrollados, como también los de ellos, si así lo decidían, por lo que, en resumidas cuentas, el App Service se estableció con fines comerciales.

Figura 114

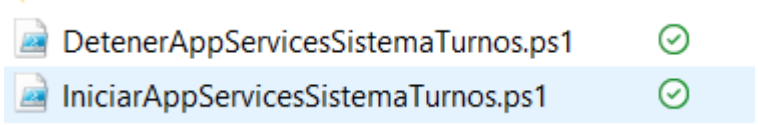
Creación de Recursos Necesarios en Azure.



Nota: Fuente propia

Figura 115

Archivos de Ejecución de App Service.



Nota: Fuente propia

Luego de ello fue de relevancia crear variables de entorno que permitieran, de acuerdo con el objetivo del usuario apuntar a un contenedor específico donde se almacenaba el código, pues, estaba el de producción, el de pruebas y el denominado como “qa”, que estaba almacenado en azure.

Figura 116

Variables de Entorno.

Nota: Fuente propia

Finalmente, se realizaron pruebas que permitieron corroborar que todo este despliegue se había realizado correctamente, apuntando, a los diversos contenedores creados y ya existentes. Algunas de las pruebas se centraron en ejecutar los comandos almacenados en los archivos para powershell (para el App Services de Azure), para el contenedor de pruebas y claramente para ejecutar el sistema en general.

Figura 117

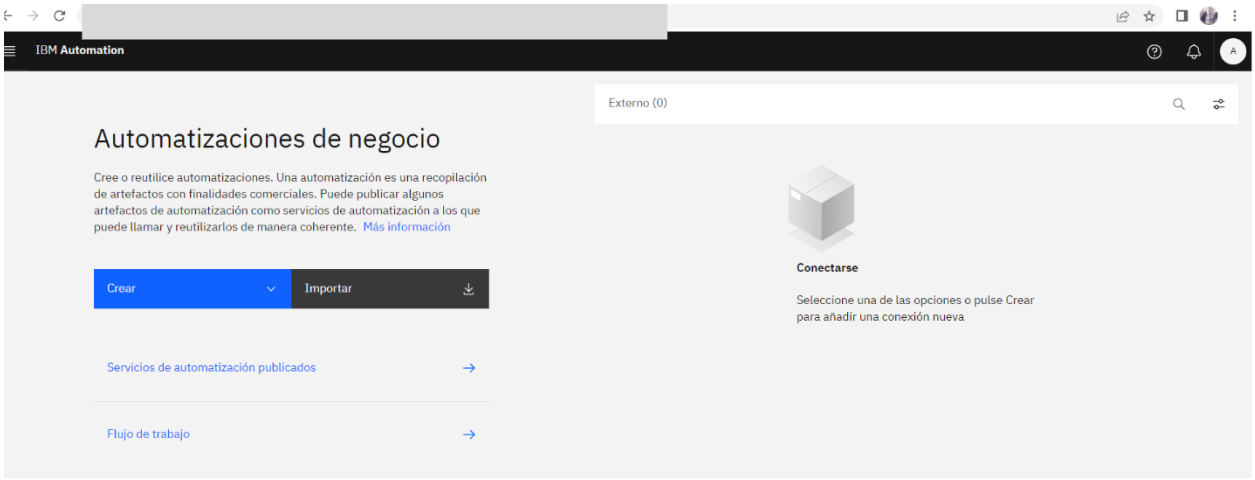
Ejecución Archivos Para Inicio del App Service en Powershell.

Nota: Fuente propia

Figura 118

Ejecución de IBM WFPS en el Servidor Posterior al Despliegue.

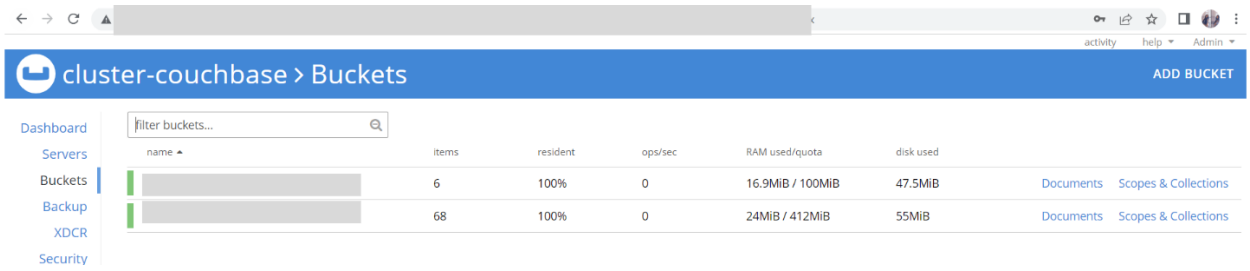
Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.



Nota: Fuente propia

Figura 119

Ejecución de Couchbase en el Servidor Posterior al Despliegue.



Nota: Fuente propia

9. Conclusiones

Gracias a las reuniones, el trabajo en equipo y la experiencia de la empresa frente a industrias de la salud se hallaron oportunidades de mejora en el mercado con el proyecto en cuestión, esto debido a la capacidad de automatización e innovación que se podía brindar con una herramienta tan potente como lo es IBM Cloud Pak for Business Automation, enfatizando mayormente en su flexibilidad frente al manejo de reglas de negocio con las que puede contar cualquier empresa promotora de salud, su adaptabilidad en cuanto a personalización y procesos además de su capacidad de automatizar procesos de forma simple durante su desarrollo.

A pesar de no presenciar la fase de análisis o levantamiento de requerimientos durante la pasantía, se logró un análisis de los aspectos que podrían hacer de este proyecto único e innovador, pues gracias al entendimiento de los procesos actuales e incluso ser parte de aquella sociedad que requiere de este tipo de servicios fue posible aportar ideas y aspectos técnicos que hicieran de este desarrollo algo completo y diferente.

De acuerdo con las especificaciones del proyecto, fue importante diseñar un tipo de modelo de datos que serían requeridos en todos los procesos de cada módulo desarrollado, por lo que, al implementar una base de datos no relacional-documental se optó por generar una estructura de datos de manera progresiva, lo que quiere decir que al momento de desarrollar o llevar a cabo un sprint la tarea de diseñar una estructura de datos se llevaba a cabo, obteniendo un modelo óptimo frente a la necesidad y el objetivo en cuestión, abarcando todos los posibles escenarios que podrían verse reflejados en este tipo de información.

Gracias al trabajo en equipo y al fomento de la organización frente a la pasantía bajo una metodología, se obtuvo un producto o solución integral, tal como esperaba la empresa desde un inicio, pues todos los módulos fueron desarrollados en el tiempo estimado abarcando todos los

objetivos de los sprints, las pruebas y su posibilidad de mejora; obteniendo un total de 6 módulos realizados bajo estándares de programación como por ejemplo Clean Architecture, utilizando herramientas óptimas a nivel de programación y diseño de frontend, reflejándose en el producto final y teniendo presente la oportunidad de mejorar la calidad de atención en cualquier entidad que desee implementar esta solución.

El uso de IBM Cloud Pak for Business Automation, específicamente el módulo: Workflow Process Services, fue de los procesos más relevantes durante la construcción del sistema y sobre todo durante la pasantía, pues toda la integración de componentes del proyecto se centró en dicha herramienta, considerándose como la base o el motor para el proyecto, permitiendo de esta manera automatizar la gran mayoría de procesos para los diferentes escenarios tenidos en cuenta, esto trabajando con flujos de trabajo que contenían elementos potentes para mejorar la calidad laboral y de servicio respecto a los operarios de servicio en representación de la entidad y el usuario final como paciente.

Gracias a las pruebas realizadas conforme el proyecto transcurría y evolucionaba, se pudo comprobar que la calidad del producto para su primera versión era la adecuada, sin embargo, se consideró un posible versionamiento para agregar más funcionalidades a este sistema, esto a causa de una oportunidad de indagación en una entidad dedicada en su totalidad al sector salud, la cual permitió un estudio de sus procesos para seguir en este fundamento de mejora incremental. Cabe resaltar que este versionamiento planteado fue decidido durante una de las reuniones con el equipo de trabajo, sin embargo, no forma parte de la pasantía debido a las fechas establecidas y objetivos planteados de manera inicial.

Frente al compromiso, dedicación y gran entendimiento en este transcurso, y gracias al apoyo de todos los miembros de la empresa y a su confianza, fue posible participar con este

proyecto en un concurso con uno de los partners más significativos para la entidad, tal como lo es IBM, presenciando vivencias significativas a nivel profesional y logrando ser uno de los ganadores a nivel nacional de esta competencia, esto por el desarrollo de un producto que evolucionará la operatividad para cada uno de sus clientes, logrando méritos por su innovación y contribuyendo de esta forma con el reconocimiento de la empresa y recibiendo apoyo de IBM para fortalecer el ámbito de comercialización y las posibilidades de mejora del mismo.

La oportunidad de formar parte de un entorno laboral contribuyendo con los conocimientos obtenidos a lo largo de la carrera se consideró como una experiencia bastante fructífera, ya que se consolidan los valores, el conocimiento y la voluntad de aprender constantemente para brindar todas estas capacidades a la sociedad actual, lo cual es de las principales motivaciones y objetivos al estudiar ingeniería de sistemas.

Referencias Bibliográficas

Acens. (s.f). *Bases de datos NoSQL. Qué son y tipos que nos podemos encontrar.* Acens https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf

Amaro, S., & Valverde, J. (2007). Metodologías Ágiles [Universidad Nacional de Trujillo].

https://www.academia.edu/33131058/Universidad_Nacional_de_Trujillo

AWS. (s.f). *¿Qué es Docker?* . Amazon Web Services, Inc. https://aws.amazon.com/es/docker/

Arias, R., & Simbaña, G. (2012). *Análisis, diseño y desarrollo de los módulos de turnos, cita previa y parte diario del sistema de gestión médico para áreas de salud (SGMAS), para el centro de salud para el Centro de Salud Na3 "La Tola-Vicentina" de la Dirección Provincial de Salud de Pichincha.* https://dspace.ups.edu.ec/handle/123456789/3897

Azure. (s/f). *Qué es Azure: Servicios en la nube de Microsoft.* Microsoft Azure. https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-azure/

Cloud Center Andalucía. (2021, 15 julio). *Localhost: qué es, conceptos básicos y como crearlo.* Cloud Center Andalucía. https://www.cloudcenterandalucia.es/blog/localhost-que-es-conceptos-basicos-y-como-crearlo/

Couchbase. (s.f). *Couchbase: The Modern Database for Enterprise Applications.* Couchbase. https://www.couchbase.com/

Deemer, P., Benefield, G., Larman,C. (2009). *INFORMACIÓN BÁSICA DE SCRUM (THE SCRUM PRIMER).* Edu.sv. http://libroslibres.uls.edu.sv/informatica/informacion_basica_scrum.pdf

Del Valle, D. (2012). *Gestión Avanzada de Turnos*. Ediciones EAE.

Duperet, C., Pérez, E., Gabriel, D., Cedeño, M., Ramírez, A., & Montoya, L. (2015). «*Importancia de los repositorios para preservar y recuperar la información*». MEDISAN, 19(10), 1283-1290.

FastAPI. (s.f). Recuperado 25 de octubre de 2022, de https://fastapi.tiangolo.com/

Giordani, L. (2022). *Clean Architectures in Python.* The Digital Cat Books. https://www.thedigitalcatbooks.com/pycabook-introduction/

Gitlab. (s.f). *The One DevOps Platform.* GitLab. https://about.gitlab.com/

Grupo consultores efe. (s.f). *Sistemas web.* Grupo consultores efe. https://grupoconsultorefe.com/servicio/tecnologias-de-la-informacion/sistemas-web

IBM. (2021, 28 agosto). *API REST.* International Business Machines. https://www.ibm.com/co-es/cloud/learn/rest-apis

IBM. (2022, marzo 01). *IBM Workflow Process Service.* International Business Machines. https://www.ibm.com/docs/en/cloud-paks/cp-biz-automation/21.0.x?topic=software-workflow-process-service

Landazuri, R., Vinicio, R., Siza, H. (s.f). *Diseño, desarrollo e implementación del sistema de gestión de turnos programados para el Sub-centro de Salud Carapungo*. Edu.ec. http://repositorio.puce.edu.ec/bitstream/handle/22000/3392/T-PUCE-3571.pdf?sequence=1&isAllowed=y

mdn_web_docs. (s.f). *CRUD.* MDN. https://developer.mozilla.org/es/docs/Glossary/CRUD

Meneses, C., (2017). *Estudio del problema de programación de turnos de enfermería.* http://tangara.uis.edu.co/biblioweb/tesis/2017/168600.pdf

Microsoft. (s.f). *Introducción a Microsoft Teams.* Microsoft. https://support.microsoft.com/es-es/office/introducci%C3%B3n-a-microsoft-teams-b98d533f-118e-4bae-bf44-3df2470c2b12

Microsoft Azure. (s.f). *Máquinas virtuales: PC virtuales dentro de PC.* Microsoft Azure. https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-a-virtual-machine/

Ministerio de salud. (2020, junio 30). *Un año de logros en el sector salud.* Ministerio de salud. https://www.minsalud.gov.co/Paginas/Un-ano-de-logros-en-el-sector-salud-.aspx

Oracle. (s.f). *¿Qué es una base de datos?* Oracle. https://www.oracle.com/co/database/what-is-database/

Paez, L. (2021, diciembre 21). *¿Qué es Notion? La mejor app de productividad que organizará tu vida.* Crehana. https://www.crehana.com/blog/negocios/que-es-notion/

Pineda, A., & Becerra, C. (2014). *Construcción de un sistema inteligente para la asignación dinámica de turnos en el área de consulta externa del hospital regional Isidro Ayora.* Edu.ec. https://dspace.unl.edu.ec/jspui/bitstream/123456789/14196/1/Becerra%20González%2c%20Carmen%20Elizabeth%2c%20Pineda%20Torres%2c%20Alba%20del%20Rocío.pdf

Platzi. (2018, 21 febrero). *Qué es Frontend y Backend: diferencias y características.* Platzi. https://platzi.com/blog/que-es-frontend-y-backend/

Python. (s,f). *What is Python?*. Python. https://docs.python.org/3/faq/general.html#what-is-python

Ramos, E., (2019). *Sistema integral de gestión de turnos.*

Edu.ec. https://dspace.itcolima.edu.mx/bitstream/handle/123456789/1487/52140%20Eder%20Carlos%20Rai%20Ramos%20Rosales.pdf?sequence=1&isAllowed=y

Red Hat. (2022, mayo 10). ¿Qué es *la automatización? Ventajas e importancia de automatizar.* Red Hat. https://www.redhat.com/es/topics/automation

Redes y sistemas integrados S.A.S. (s.f). *Somos redsis.* Redes y sistemas integrados S.A.S. https://redsis.co/somos-redsis/

Rodríguez, B., & Arciniegas, C. (2019). *Rediseño y reingeniería sistema para control de turno.* Edu.ec. https://repository.ucc.edu.co/bitstream/20.500.12494/16209/1/redise%C3%B1o_reingenieria_sistema_2020_.pdf

Santa María, M., García, A., Prada, L., Uribe, M., & Vásquez, B. (2009). *El sector salud en Colombia: impacto del SGSSS después de más de una década de la reforma.* https://repository.fedesarrollo.org.co/bitstream/handle/11445/968/Co_So_Diciembre_2008_Santa_Maria_et_al.pdf?sequence=2&isAllowed=y

Schwaber, Z., & Sutherland, J. (2020). *The Scrum Guide.* https://billlewistraining.com/wp-content/uploads/2017/02/PMP-Agile-Study-Materials.pdf

Sophos. (2012). *Sophos VPN Clients.* Sophos. https://www.sophos.com/en-us/medialibrary/pdfs/factsheets/sophosvpnclientsdsna.pdf

Swagger. (s.f). *API Documentation Made Easy- Get Started.* Swagger. https://swagger.io/solutions/api-documentation/

Termius. (s.f). *Termius- SSH platform for Mobile and Desktop.* Termius. https://termius.com/

The Blockehead. (2016, septiembre 22). *Scrum- ¡Guía definitiva de prácticas ágiles esenciales de Scrum!* Babelcube. https://books.google.es/books?hl=es&lr=&id=T24eDQAAQBAJ&oi=fnd&pg=PT17&dq=scrum+fases&ots=KUrtkQ5OkU&sig=f8DZG8UGQOYSibJRZsJDSrHL2_I#v=onepage&q=scrum%20fases&f=false

Trello. (s.f). *Gestiona los proyectos de tu equipo desde cualquier lugar.* Trello. https://trello.com/es

Villela, G. (2021). *DevOps Management Dashboard.* Iteso. https://rei.iteso.mx/bitstream/handle/11117/7692/TOG%20DevOps%20Management%20Dashboard%20Guillermo%20Ruiz.pdf?sequence=1&isAllowed=y

Webex. (s.f). *Videoconferencias, reuniones en línea, pantalla compartida.* Webex; Cisco. https://www.webex.com/es/index.html

Zenkit. (2018). *Agile Methodology: An Overview.* Zenkit. https://zenkit.com/en/blog/agilemethodology-an-overview/

Anexos


IBM Cloud
ROCKS
Redes y Sistemas
Integrados S.A.S
Rockstar
Ganador
2022
IBM
redsis
1er lugar
Empresas ganadoras de IBM Cloud Rocks
Gestión Integral de Turnos y Colas de Servicios
Daniel Pardo
Real Time Analytics Business Planning Models
Andres Huertas
Análisis de Riesgo Crediticio
Raúl Molano
RICOH
Ricoh Loan Process
IBM

Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.

Product Backlog		
Item	**Tarea**	**Aceptada (Sí/No)**
Capacitación	Entendimiento herramientas	Sí
Sprint 1		
	Entendimiento herramientas	Sí
	Capacitación IBM (5 niveles)	Sí
	Estudio de herramientas	Sí
	Aplicación máquina de estados	Sí
	Definir estructura BD	Sí
	Definir estructura clean architecture (Carpetas)	Sí
	Instalar SDK Python	Sí
	Crear estructura de datos para turnero	Sí
	Creación del coach/Prototipo turnero	Sí
	Crear colecciones BD- Turnos	Sí
	Creación endpoints (Casos de uso, entities, repositories)	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Crear tablas de decisiones (reglas de negocio)	Sí
	Asignar reglas al turnero	Sí
	Validar campo de la cédula (id usuario)	Sí
	Pruebas de integración	Sí
Sprint 2		
	Crear estructura de datos para ventanillas (atención)	Sí
	Creación de los coaches-Prototipo de atención de ventanillas	Sí
	Cronómetro-Spinner	Sí
	Crear colecciones BD	Sí
	Creación endpoints (Casos de uso, entities, repositories)	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Pruebas de integración	Sí

Sprint 3		
	Crear estructura de datos para colas turnos (clasificación)	Sí
	Creación de coach- Prototipo tablero de turnos	Sí
	Crear colecciones BD	Sí
	Creación endpoints (Casos de uso, entities, repositories)	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Pruebas de integración	Sí
Sprint 4		
	Crear estructura de datos para sedes y salas	Sí
	Creación de coaches- Prototipo menú, sección salas y sedes	Sí
	Diseño modales básicos para el módulo	Sí
	Crear colecciones BD	Sí
	Creación endpoints (Casos de uso, entities, repositories)-Sede	Sí
	Creación endpoints (Casos de uso, entities, repositories)-Salas	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow- Sala	Sí
	Implementar workflow- Sede	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Pruebas de integración	Sí
Sprint 5		
	Crear estructura de datos para servicios y ventanillas	Sí
	Creación de coaches- sección servicios/ventanillas	
	Crear colección BD	Sí
	Creación endpoints (Casos de uso, entities, repositories)-Servicios	
	Creación endpoints (Casos de uso, entities, repositories)-Venta	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow- Servicios	
	Implementar workflow- Ventanillas	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Pruebas de integración	Sí

Desarrollo del sistema de gestión de turnos de atención a gran escala: sector salud.

Sprint 6		
	Crear estructura de datos para Tableros-turneros	Sí
	Creación coach reinicio de turnero	Sí
	Creación de coaches- sección Tablero, turnos	Sí
	Crear colecciones BD (3)	Sí
	Creación endpoints (Casos de uso, entities, repositories)-Servi	Sí
	Creación endpoints (Casos de uso, entities, repositories)-Reini	Sí
	Creación endpoints (Casos de uso, entities, repositories)-Venta	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow- Servicios	Sí
	Implementar workflow- Ventanillas	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Pruebas de integración	Sí
Sprint 7		
	Revisión estructuras pertinentes para la aplicación de funcion	Sí
	Creación de coach-Tablero control	Sí
	Creación endpoints (Casos de uso, entities, repositories)	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Pruebas de integración	Sí
Sprint 8		
	Análisis de data para aplicar por sección	Sí
	Validar exportación de datos	Sí
	Creación de coach-Tablero de indicadores	Sí
	Validación funciones dinámicas	Sí
	Creación endpoints (Casos de uso, entities, repositories)	Sí
	Pruebas unitarias endpoints	Sí
	Documentar endpoints en Swagger	Sí
	Importar servicios documentados a IBM WFPS	Sí
	Crear servicio externo	Sí
	Generar flujos de servicios	Sí
	Implementar workflow	Sí
	Agregar variables de entrada, salida y privadas	Sí
	Correlacionar datos	Sí
	Pruebas de integración	Sí

Sprint 9		
	Caso tiempo de espera- Validar	Sí
	Caso de asignación de un turno a una ventanilla- Validar	Sí
	Caso de creación de un turno- Validar	Sí
	Revisión algoritmo actual por caso	Sí
	Creación algoritmos por caso	Sí
	Pruebas unitarias a los endpoints modificados	Sí
	Documentación en Swagger	Sí
	Actualizar servicios externos y flujos de servicio	Sí
	Modificación workflows por caso	Sí
	Pruebas de integración	Sí
Sprint 10		
	Análisis de roles, procesos y equipos	Sí
	Creación de procesos	Sí
	Creación de equipos IBM WFPS	Sí
	Asignación de usuarios	Sí
	Asignar equipo por carril de proceso	Sí
	Prueba de procesos- Ejecución Workplace	Sí
Sprint 11		
	Fusión branches	Sí
	Docker compose- Contenedores	Sí
	Creación variables de entorno- Validar de acuerdo al contexto	Sí
	Contenedores en VM- Despliegue	Sí
	Pruebas generales	Sí