Implementación de los microservicios orientados al agendamiento de turnos, citas y seguridades del sistema usando el paradigma de Línea de Producto Software (LPS) enfocado a un desarrollo co-localizado (DGS).

Romo Gavilánez, Paola Lissette y Sánchez Pico, Ana Lissette

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Software

Trabajo de Integración Curricular, previo a obtener el título de Ingeniera de Software

Dr. Jácome Guerrero, Patricio Santiago

08 de febrero del 2023

Latacunga

CERTIFICADO DE ANÁLISIS
magister

Tesina_Final_Sanchez_Romo_16-02-2023-Antiplagio

6% Similitudes

3% Texto entre comillas
< 1% similitudes entre comillas
< 1% Idioma no reconocido

Nombre del documento: Tesina_Final_Sanchez_Romo_16-02-2023-Antiplagio.docx
ID del documento: 2561f1bc058afa95c6292e98c3f319456da359014
Tamaño del documento original: 6,08 Mo

Depositante: JOSÉ LUIS CARRILLO
Fecha de depósito: 16/2/2023
Tipo de carga: interface
fecha de fin de análisis: 16/2/2023

Número de palabras: 16.988
Número de caracteres: 111.806

Ubicación de las similitudes en el documento:

Fuentes principales detectadas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	oa.upm.es | Gestión de configuración y línea de productos para mejorar el proceso e... https://oa.upm.es/30998/ 5 fuentes similares	2%		Palabras idénticas : 2% (245 palabras)
2	repositorio.utn.edu.ec | Arquitectura de software basada en microservicios para de... http://repositorio.utn.edu.ec/bitstream/123456789/76036/PG 569 TESIS.pdf.txt 3 fuentes similares	1%		Palabras idénticas : 1% (200 palabras)
3	oa.upm.es | Gestión de configuración y línea de productos para mejorar el proceso e... https://oa.upm.es/30998/1/EDISON_GONZALO_ESPINOSA_GALLARDO.pdf	1%		Palabras idénticas : 1% (200 palabras)
4	1library.co | Metodologías de desarrollo de software https://1library.co/document/zkwwlllz-metodologias-de-desarrollo-de-software.html#:~:text=En la act... 4 fuentes similares	< 1%		Palabras idénticas : < 1% (154 palabras)
5	dialnet.unirioja.es https://dialnet.unirioja.es/descarga/articulo/2533323.pdf	< 1%		Palabras idénticas : < 1% (136 palabras)

Fuentes con similitudes fortuitas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	Castillo_Eddy_Perez_Christian_UIC202251.pdf | Castillo_Eddy_Perez_Christia... #0ed139 El documento proviene de mi grupo	< 1%		Palabras idénticas : < 1% (26 palabras)
2	www.semanticscholar.org | Creating virtual stores using software product lines: An ... https://www.semanticscholar.org/paper/Creating-virtual-stores-using-software-product-An-Rincón-Rod...	< 1%		Palabras idénticas : < 1% (19 palabras)
3	repositorio.espe.edu.ec | Implementación de un repositorio de artefactos de softwa... http://repositorio.espe.edu.ec:8080/bitstream/21000/24953/4/M-ESPEL-sit-0098.pdf.txt	< 1%		Palabras idénticas : < 1% (12 palabras)
4	repository.javeriana.edu.co https://repository.javeriana.edu.co/bitstream/handle/10554/16496/NarvaezBarreraMariaXimena2015...	< 1%		Palabras idénticas : < 1% (11 palabras)

Fuentes mencionadas (sin similitudes detectadas)

Estas fuentes han sido citadas en el documento sin encontrar similitudes.

1 https://online.visual-paradigm.com/share.jsp?id=323235363738332d33
2 https://online.visual-paradigm.com/share.jsp?id=323235363835342d31
3 https://online.visual-paradigm.com/share.jsp?id=323235363738322d38
4 https://microservicio1-paciente-tutor.azurewebsites.net/
5 https://microservicio2-gestion-spa.azurewebsites.net/

PATRICIO SANTIAGO JACOME GUERRERO

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Software

Certificación

Certifico que el trabajo de integración curricular: "**Implementación de los microservicios orientados al agendamiento de turnos, citas y seguridades del sistema usando el paradigma de Línea de Producto Software (LPS) enfocado a un desarrollo co-localizado (DGS).**" fue realizado por las señoritas Romo Gavilánez, Paola Lissette y Sánchez Pico, Ana Lissette el mismo que cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, además fue revisado y analizado en su totalidad por la herramienta de prevención y/o verificación de similitud de contenidos; razón por la cual me permito acreditar y autorizar para que se lo sustente públicamente.

Latacunga, 08 de febrero del 2023

Jácome Guerrero, Patricio Santiago

C. C. 1001689791

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Software

Responsabilidad de Autoría

Nosotras **Romo Gavilánez, Paola Lissette y Sánchez Pico, Ana Lissette**; con cédulas de ciudadanía n°1850897115 y 1850045376 respectivamente; declaramos que el contenido, ideas y criterios del trabajo de integración curricular: **"Implementación de los microservicios orientados al agendamiento de turnos, citas y seguridades del sistema usando el paradigma de Línea de Producto Software (LPS) enfocado a un desarrollo co-localizado (DGS).**", es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Latacunga, 08 de febrero del 2023

Romo Gavilánez, Paola Lissette

C. C. 1850897115

Sánchez Pico, Ana Lissette

C. C. 1850045376

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Software

Autorización de Publicación

Nosotras **Romo Gavilánez, Paola Lissette y Sánchez Pico, Ana Lissette**; con cédulas de ciudadanía n° 1850897115 y 1850045376 respectivamente; autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de integración curricular: **Implementación de los microservicios orientados al agendamiento de turnos, citas y seguridades del sistema usando el paradigma de Línea de Producto Software (LPS) enfocado a un desarrollo co-localizado (DGS)**, en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra responsabilidad.

Latacunga, 08 de febrero de 2023

.....	.....
Romo Gavilánez, Paola Lissette	**Sánchez Pico, Ana Lissette**
C. C .1850897115	C. C. 1850045376

Dedicatoria

Dedico este trabajo a mi familia, a mi madre a mi hermana y a mi padre que está en el cielo, ellos han sido siempre lo más importante de mi vida, hoy que con la bendición de Dios alcanzo esta meta, no me queda más que agradecerles por todo el apoyo brindado a lo largo de estos años, por estar a mi lado en las buenas y en las malas y siempre tener un consejo para reconfortarme.

ROMO GAVILANEZ, PAOLA LISSETTE

Dedicatoria

Quiero dedicar el siguiente proyecto a mis padres, pieza fundamental para mi crecimiento personal y profesional, brindándome apoyo incondicional en momentos difíciles han permitido que cumpla este sueño mediante su arduo trabajo.

A mis hermanos, a mis abuelitos, que me inculcaron desde niña a ser perseverante, dedicada, y esforzarme para cumplir con mis objetivos

A mi tío Ramiro (+) una de las personas que me vio crecer, siempre creyó en mí y hoy me cuida desde el cielo.

SANCHEZ PICO, ANA LISSETTE

Agradecimiento

Quiero empezar agradeciendo a Dios por iluminar mi camino y hoy brindarme la oportunidad de alcanzar tan anhelada meta.

A mis compañeros de equipo, especialmente a Ana Sánchez mi amiga y compañera dentro y fuera de la universidad, has sido un pilar fundamental en la ejecución de este proyecto.

A mis amigos realizados a lo largo de la carrera, con quienes hemos compartido muchas experiencias a lo largo de estos años.

Quiero agradecer a mi madre, sin ella nada de esto sería posible, gracias por ser la persona que nunca me ha dejado rendirme, que siempre que me he caído a estado ahí para levantarme y guiar mi camino. Muchas veces pensé en darme por vencida, pero siempre has estado para motivarme y darme el aliento para seguir adelante.

Quiero agradecer a mi hermana Nathaly, eres mi compañera de aventuras, mi amiga y mi confidente, gracias por siempre confiar en mí y estar a mi lado, sé que llegaras muy lejos en la vida.

Finalizo agradeciendo a mi padre, que, aunque partió muy pronto de este mundo, me apoyo y confió en mí siempre. Aunque ya no estes a mi lado físicamente, siempre en cada paso que doy te he sentido cerca y esta vez no es la excepción.

ROMO GAVILANEZ, PAOLA LISSETTE

Agradecimiento

Quiero agradecer al Dr. Santiago Jácome, por su guía durante el proceso de elaboración del presente trabajo.

A mi madre Laura por su sacrificio diario para darme el sustento y la educación, sin ella esto no hubiera sido posible.

A mi padre Luis por haberme inculcado valores de responsabilidad, respeto, dedicación, valentía, por sus consejos a lo largo de mi vida contribuyendo a mi desarrollo ético y moral.

A mis hermanos por haber sido guía durante toda mi carrera universitaria y por siempre darme palabras de aliento.

A mi compañera de tesina y amiga Paola Romo por su valiosa amistad dentro y fuera de la universidad y por el tiempo dedicado a este trabajo.

A mis abuelitos en especial a mi abuelita Corina que con sus consejos siempre fue un apoyo y guía constante en mi vida, aunque ya no está me acompañará siempre en cada meta.

A mis amigos por haber sido apoyo durante toda mi etapa de formación Universitaria, y acompañarme en este camino.

A Dios por ser mi guía espiritual para afrontar momentos difíciles y lograr esta meta.

SÁNCHEZ PICO, ANA LISSETTE

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Resumen

El presente trabajo de titulación propone una parte del Sistema de Gestión de Clínicas Veterinarias general aplicando el paradigma Línea de Producto de Software (LPS), por medio del Análisis de Dominio Orientado a Características (FODA), se adapta al marco de desarrollo ágil SAFe para modelar las características presentadas en el FODA las cuales representan los procesos comunes y específicos de cada clínica veterinaria. Para la coordinación del equipo de trabajo se utilizó la metodología ágil SCRUM, que permite organizar las funciones del equipo de trabajo y agilizar la creación de nuevos productos software, se trabajó en base al Desarrollo Global de Software (DGS) co-localizado, debido a que los equipos de trabajos estaban en lugares diferentes dentro de la misma localidad a cada miembro del equipo se le asignó una rama para el desarrollo de cada característica las cuales fueron integradas a través de la herramienta Gitflow. Se aplicó arquitectura de microservicios debido a su facilidad de integración y reusabilidad los cuales fueron desplegados en Azure. Además, se presenta las pruebas de funcionalidad realizadas en Postman y las pruebas de rendimiento utilizando la herramienta Locust a través de una carga masiva de datos obteniendo los tiempos de respuesta de cada microservicio.

Palabras claves: Línea de Producto Software, Análisis de Dominio Orientado a Características, Desarrollo Global de Software, Microservicios, Identity Server, SCRUM, Azure

Abstract

This degree work proposes a part of the general Veterinary Clinic Management System applying the Software Product Line (LPS) paradigm, by means of the Feature-Oriented Domain Analysis (FODA), it is adapted to the SAFe agile development framework to model the features presented in the FODA, which represent the common and specific processes of each veterinary clinic. For the coordination of the work team, the agile SCRUM methodology was used, which allows organizing the work team functions and speeding up the creation of new software products. The work was based on co-localized Global Software Development (DGS), since the work teams were located in different places within the same locality, each team member was assigned a branch for the development of each characteristic, which were integrated through the Gitflow tool. Microservices architecture was applied due to its ease of integration and reusability which were deployed in Azure. In addition, the functionality tests performed in Postman and the performance tests using the Locust tool are presented through a massive data load obtaining the response times of each microservice.

Key words: Software Product Line, Feature-Oriented Domain Analysis, Global Software Development, Microservices, Identity Server, SCRUM, Azure

Glosario

A

Análisis de dominio: Determinación de la lógica de negocio.

APIs: Interfaz de programación de aplicaciones.

App service: Componente para publicar proyectos en Azure.

Azure: Sistema de computación en la nube de Microsoft.

B

Back end: Parte de un sistema el cual interactúa con la base de datos y con el front end.

Big-bang: Forma de organización de equipos en Desarrollo Global de Software.

C

Core assets: activos base.

CSS: Hoja de estilo en cascada.

D

Database: Base de datos.

Desarrollo co-localizado: Forma de localización dentro del desarrollo global de software.

DGS: Desarrollo global de software.

E

ECOP: Examen clínico orientado a problemas.

F

Features: Funcionalidades del sistema.

FODA: Análisis de dominio orientado a características.

Front end: Parte de un sistema web la cual permite al usuario interaccionar con él.

Folow the sun: Método de desarrollo en DGS.

G

Git: Gestor de versionamiento.

Git Flow: Marco de trabajo de git.

H

HTTP: Protocolo de transferencia de hipertexto.

L

Lean: Marco de trabajo para empresas enfocado en el cliente.

Liquibase: Framework para gestión de base de datos.

LPS: Línea de producto software.

N

Namespace: Palabra reservada del lenguaje Typescript.

Nodejs: Lenguaje para back end de javascript

O

OpenId: Protocolo de autenticación basado en oauth2.0.

P

Pet Grooming: Servicio complete de aseo y cuidado de mascotas.

PuLSE: Ingeniería de línea de producto software

Pull request: Acción para la unión de ramas en git.

R

React: Librería de desarrollo web.

S

Sprint: Es un periodo breve de tiempo fijo en el que un equipo de scrum trabaja para completar una cantidad de trabajo establecida.

Sprint daily: Reunión diaria realizada dentro del sprint.

Capítulo I

Introducción

Antecedentes

De acuerdo con la información proporcionada por el Instituto Ecuatoriano de Estadística y Censos (INEC), en el mercado de la ciudad de Portoviejo existen 128 establecimientos que ofrecen servicios relacionados con la atención médica de animales, lo que representa una actividad económica anual de más de 2.3 millones de dólares. (Ponce, 2013). En el Ecuador, las clínicas veterinarias han ido tomando una gran importancia debido a que “seis de cada diez hogares del país tienen mascotas” (La República, 2019) Las clínicas veterinarias son empresas que brindan sus servicios a mascotas como perros, gatos entre otras, que se han incrementado a lo largo de la historia, ya que, en este negocio existen procesos como venta de productos, cuidado de mascotas, SPA, atención médica e incluso existen clínicas veterinarias que tienen todos estos procesos.

“En Ecuador, el uso de la tecnología se ha vuelto fundamental tanto en el ámbito empresarial como personal para lograr un crecimiento efectivo. La tecnología brinda la oportunidad de acceder a diferentes productos y servicios que permiten un uso más eficiente y productivo.” (Zavala, 2019). La mayoría de las clínicas veterinarias en el Ecuador realizan todo su proceso y registro de forma manual lo cual provoca falta de organización de los expedientes físicos, carencia de código que identifique a cada expediente, la información de los registros en ocasiones no es legible por la calidad de caligrafía utilizada, existe la probabilidad de que puedan afectar en la información obtenida en los registros médicos, productos y Pacientes debido a la pérdida o inexistencia de información.

Por ello las clínicas veterinarias han optado por llevar un manejo automatizado de sus datos, y toman alternativas en la implementación de herramientas tecnológicas para el soporte y ayuda de gestión dentro de sus establecimientos.

Justificación e importancia

En el Plan Nacional del Buen Vivir (2017 - 2021) en el objetivo número 3 se da respaldo a la importancia del cuidado de la fauna del país:

> La comprensión de la ecodependencia, además, se extiende al cuidado y protección de la fauna, constatando la importancia de la vida y la dignidad en su sentido ético amplio, por lo que es preciso precautelar el bienestar animal con normativa, política pública y jurisprudencia expresa, clara y directa. (Secretaria nacional de planificación, 2017)

En el Ecuador las personas han brindado una mejor atención y cuidado hacia los animales, lo que ha llevado a generar nuevos centros de especialidades de cuidado para animales domésticos; aumentando así la cantidad de profesionales veterinarios. Dentro de las clínicas veterinarias buscan prestar una atención completa y de calidad para las mascotas, con médicos preparados en su respectiva área para dar un servicio más profesional posible, por lo cual se diseñará e implementará un sistema que brinda ayuda para los servicios que tienen las clínicas veterinarias que albergan acciones tanto como son de registro, diagnóstico, tratamiento, spa, venta de productos e historial médico, obteniendo un resultado la reducción de tiempo al momento de atención hacia el cliente.

El objetivo de la implementación del sistema para las clínicas veterinarias se logrará una satisfacción y rapidez al momento de un registro automatizado de datos dentro de las clínicas veterinarias, obteniendo como resultado la agilidad y rapidez en los procesos, teniendo un mejor control de la información dentro de las clínicas veterinarias, evitando de esta manera los procesos manuales y físicos de la información tengan errores al momento de realizarlos. Con el

uso de un sistema tecnológico se podrá aumentar tanto la productividad y agilidad en procesos que manejan dentro de las clínicas veterinarias.

Económicamente el objetivo es cubrir las necesidades de Clínicas Veterinarias a través de la reutilización de componentes software en construcción de nuevos productos disminuyendo costos, tiempo y recursos en la producción de sistemas de calidad aplicando el paradigma LPS.

Socialmente el objetivo es abarcar diversas Clínicas Veterinarias a nivel nacional con software que cumpla sus necesidades específicas y generales, con un costo accesible, que permita mejorar la calidad en el proceso de atención al cliente

Alcance

Considerando la importancia de las clínicas veterinarias y los servicios que se brindaran se propone desarrollar una aplicación web, para mejorar la gestión y almacenamiento de la información de todas áreas mediante la administración de Pacientes, Tutores, historial clínico, consultas, carnet de vacunación, spa, hospedaje, exámenes complementarios y catálogo de productos.

Este sistema está realizado con arquitectura microservicios por lo cual será eficaz, fácil de mantener y probar ya que está dividido en las características antes mencionadas, “de manera que los procesos brindan las clínicas veterinarias sean realizados de forma ágil y eficiente, tiene como resultado una reducción de tiempo de consulta del Paciente y atención al cliente.” (Cedeno, Catuto, & Rodas-Silva, 2021). Adicionalmente se implementará autenticación para proteger y garantizar la seguridad de los datos que la veterinaria posea.

Planteamiento del problema

Los problemas más comunes que presentan las clínicas veterinarias se encuentran en la recolección de información de forma manual lo que provoca falta de organización de los expedientes físicos, carencia de código único que identifique a cada expediente, la información del registro en ocasiones no es legible por la calidad de la caligrafía utilizada.

Para la gestión administrativa de veterinarias se han desarrollado sistemas que apoyan a los profesionales de la salud, (Cedeno, Catuto, & Rodas-Silva, 2021) mencionan:

> Como es el caso de la clínica San Agustín, ubicada en Santiago de Chile, en el que se requirió el desarrollo de un "Sistema de Gestión para una Clínica Veterinaria" que permita facilitar y gestionar la información de forma más ordenada, en un solo lugar.

De igual manera en Ecuador en los últimos años se han implementado proyectos que tienen como objetivos el mejoramiento de atención y manejo de información en clínicas veterinarias.

> En la Península de Santa Elena, en el año 2017, se llevó a cabo la implementación de un “Sistema en la nube para Control y Manejo de Procesos Clínicos”. Este proyecto facilitó la reducción de los tiempos de respuesta, dando la optimización de los principales procesos del centro médico y mejorando la seguridad, disponibilidad e integridad de la información (Cedeno, Catuto, & Rodas-Silva, 2021)

En el año 2022 en la ciudad de Ambato se realizó el proyecto denominado “Sistema electrónico de monitoreo de mascotas para la gestión de clínicas veterinarias utilizando VOIP e IOT” (León, 2022). Cuyo propósito mejorar la obtención de datos mediante un dispositivo medidor de signos vitales colocado en el pecho de las mascotas, esta información se enviará a una aplicación desarrollada en React, lo que permite monitorear en tiempo real el estado de salud de las mascotas y apoyando la detección de enfermedades en etapas tempranas.

En la actualidad no existen muchos sistemas informáticos para clínicas veterinarias que brinde un servicio de control de información, datos e historiales clínicos para los clientes, esto hace que sea mucho más difícil la búsqueda de información debido a que estos procesos son gestionados de forma manual e ineficientes, puede existir perdida de documento e inexactitud al momento de registro y búsqueda de información.

Por lo tanto, existe la necesidad de implementar un sistema que permita registrar información en la historia clínica del Paciente, tratamiento o examen, gestionar el control de la búsqueda y edición de información.

En el Ecuador, las clínicas veterinarias han ido tomando una gran importancia debido a que “seis de cada diez hogares del país tienen mascotas” ((*Seis de cada 10 hogares del país tienen mascota – PRO ECUADOR*, s. f.). Las clínicas veterinarias son empresas que brindan servicios a mascotas como perros, gatos entre otras, que se han incrementado a lo largo de la historia, en este negocio existen varios procesos como historial clínico, consultas médicas, Spa, hospedaje, catálogo de productos, etc.

En el Ecuador las personas han brindado una mejor atención y cuidado hacia los animales, lo que ha llevado a generar nuevos centros de especialidades de cuidado para animales domésticos; aumentando así la cantidad de profesionales veterinarios, que dentro de las clínicas veterinarias buscan prestar una atención completa y de calidad para las mascotas, por lo cual se diseñará e implementará un sistema que brinda ayuda para los servicios que tienen las clínicas veterinarias que albergan acciones tanto de registro, diagnóstico, tratamiento, spa, venta de productos e historial médico, obteniendo como resultado la reducción de tiempo de atención al cliente.

Considerando la importancia de las clínicas veterinarias y los servicios que brindan se propone desarrollar una aplicación web, para mejorar la gestión y almacenamiento de la información de todas las áreas mediante la administración de Pacientes, Tutores, historial clínico, consultas, carné de vacunación, Spa, hospedaje, exámenes complementarios y catálogo de productos.

Formulación del problema a resolver

Desarrollo de microservicios del sistema de gestión de clínicas veterinarias para ser consumidos en el front-end del sistema.

Objetivo general

Implementar los microservicios orientados al agendamiento de turnos, citas y seguridades del sistema.

Objetivos específicos

- Revisión del estado del arte sobre Línea de Producto Software (LPS), arquitectura de microservicios y desarrollo co-localizado.
- Elaborar la línea de producto de software basada en un análisis de dominio en clínicas veterinarias.
- Implementar los microservicios orientados al agendamiento de turnos, citas, y seguridades del sistema.
- Validación de los microservicios orientados al agendamiento de turnos, citas, y seguridades del sistema.

Hipótesis

La utilización de línea de producto software (LPS) para desarrollar software, permite implementar soluciones que se ajustan al requerimiento de las clínicas veterinarias.

Señalamiento de variables

Variable Independiente: Línea de Producto Software (LPS)

Variable Dependiente: Sistema de Gestión de Clínicas Veterinarias

Capitulo II

Marco Teórico

Introducción

En este capítulo se revisará los conceptos, características que sustentan la LPS, y los procesos de métodos Análisis de Dominio Orientado a Características (FODA), que será representada por la arquitectura de microservicios con el desarrollo global de software. Finalmente se realiza una lista de trabajos relacionados con el presente proyecto.

Línea de Producto de Software (LPS)

LPS es un paradigma que se está consolidando en ciertas áreas específicas de la producción de software, caracterizada por la reutilización planificada de core assets (componentes de software) en combinación con features (características), (Espinel, Carrillo, Flores, & Urbieta, 2022) para generar soluciones a través de productos de software.

LPS según (Clements & Northrop, 2001) es un conjunto de sistemas software, que contienen un marco común de características, las cuales satisfacen al segmento de mercado el cual se analiza; estas se desarrollan a partir de un grupo de funcionalidades comunes denominadas core assets.

En base a la definición antes mencionada se aprecia que existen dos conceptos fundamentales para la aplicación de LPS:

- Feature (características): se define como una característica conceptual del sistema, esta se puede relacionar con otras estructuras de los sistemas, o pueden describir la funcionalidad de requerimientos no funcionales, y es usada para describir o distinguir un producto en la línea.
- Core asset (componentes de software)**:** es un artefacto de software utilizado en el proceso de desarrollo de uno o más productos de la LPS, un core assets puede

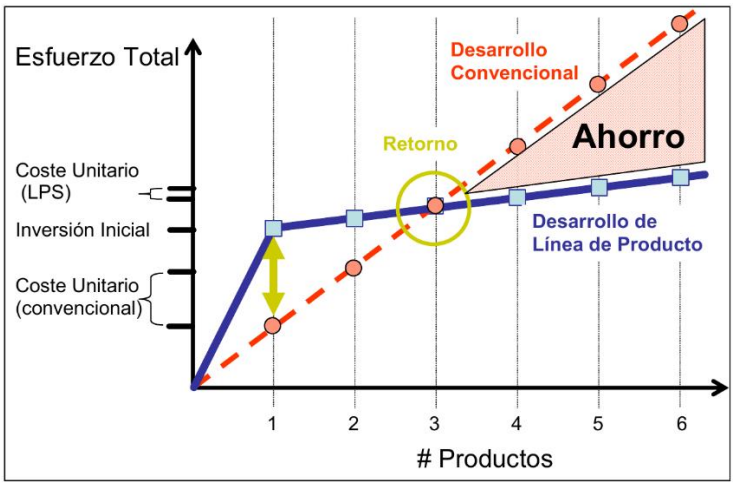
ser un componente de software, un modelo de procesos, una arquitectura, o cualquier otro resultado de la construcción de un sistema**.**

El crecimiento de software aplicando un a LPS se ha ido orientando a distintos segmentos dentro de los mercados, donde su principal objetivo es intentar satisfacer las necesidades de un segmento de mercado y la necesidad de reducción de tiempos en el desarrollo de software gracias a la reutilización de componentes.

Con la aplicación de LPS se puede aumentar la productividad a los desarrolladores disminuyendo el esfuerzo y costo para el desarrollo de software, manteniendo la efectividad y funcionalidad de los productos software. Los estudios de caso han demostrado que las ganancias de productividad han mejorado en comparación con las metodologías convencionales. En la figura 1, se presenta una gráfica comparativa del esfuerzo realizado al aplicar una metodología convencional frente a la utilización de LPS, observando a largo plazo la reducción del esfuerzo y por lo tanto costos de producción más bajos.

Figura 1

Actividades esenciales para el desarrollo de LPS.



Nota. Esta figura muestra una comparativa de esfuerzo total realizado en el desarrollo convencional de software y al utilizar LPS. Tomado de (Días)

La LPS permite al equipo de desarrollo de software manejar un mismo núcleo para varios sistemas similares, resultando así en un ahorro de recursos y tiempo a largo plazo. Los componentes de este núcleo (core assets), permite el desarrollo de múltiples productos a partir de un dominio en común. La LPS sigue siendo un nuevo enfoque, que requiere nuevas arquitecturas y métodos, en donde estos proporcionaran mecanismos para demostrar y representar las características y variabilidades del dominio, donde cada uno de métodos disponibles incluyen:

- **Síntesis:** un enfoque amplio para construir sistemas de software que representen instancias de familias de sistemas con descripciones similares. (Software Productivity Consortium, 1993)
- **Abstracción, especificación y traslación orientada a la familia:** un análisis de características comunes del dominio que es importante para: identificar el contexto; describir el dominio; proporcionar un conjunto de términos clave; identificar características y variaciones comunes; cuantificar la variabilidad proporcionando parámetros de variación; e identificar y registrar información útil durante el análisis. (Weiss & Chi, 1999)
- **Ingeniería de Línea de Producto de Software:** un método para construir y utilizar líneas de productos. La estructura general de PuLSE incluye las siguientes etapas: desarrollo, componentes técnicos y componentes de soporte. (Bayer, et al., 1999)
- **Análisis de Dominio Orientado a Características:** un método para soportar la reutilización a nivel arquitectónico y funcional. (Kang, Cohen, Hess, Novak, & Peterson, 1990)

Proceso de desarrollo LPS

El proceso incluye tres actividades que no tienen un orden preestablecido para su ejecución, y que están en constante ejecución y retroalimentación. Según (Clements &

Northrop, 2001), las actividades pertenecientes a la LPS son: Ingeniería del dominio o desarrollo de core assets, en donde se analiza las similitudes que pueden compartir entre productos y diferencias. Ingeniería de la aplicación o desarrollo de productos viables y gestión del proceso de la LPS, mantener un registro organizado de los productos realizados, una documentación y control de versiones de este (Espinosa, 2014).

En La **ingeniería del dominio** se crean los core assets, que son la base a partir de la cual se puede desarrollar un producto específico (Espinel, Carrillo, Flores, & Urbieta, 2022), es la encargada del análisis y desarrollo de todos los componentes que se requieren construir cada uno dentro de la LPS. Esta actividad produce uno o más componentes como: el plan de productos, la lista de productos que forman parte de la línea y la arquitectura de referencia (Clements & Northrop, 2001).

En la **Ingeniería de aplicación** tiene lugar la instanciación de productos de la línea (Espinel, Carrillo, Flores, & Urbieta, 2022) en esta actividad se implementa cada uno de los productos, estos son construidos utilizando los requisitos de los productos individuales los mismos se derivan de la ingeniería de dominio.

Un concepto central para la Ingeniería de dominio y aplicación es la **Gestión de Configuración de Software(GCS)** la cual se encarga de manejar la variabilidad de los core assets y los productos de la LPS, esta actividad se vuelve aún más compleja a medida que aumenta el número de combinaciones entre core assets y features (Espinel, Carrillo, Flores, & Urbieta, 2022), este pasaje se encarga de controlar y administrar varias tareas desarrolladas de los core assets y de los productos básicos de la LPS.

En la figura 2 se muestra un conjunto de actividades interactivas, interrelacionadas y en constante desarrollo, que se llevan a cabo dentro del paradigma de la LPS. Los core assets se utilizan para implementar nuevos productos, a menudo estos productos crean nuevas versiones de core assets, por lo cual la gestión de proceso de LPS se encarga de controlar la evolución

del core assets, las características del producto y los productos desarrollados en la LPS (Quishpe, 2018).

Figura 2

Actividades esenciales para el desarrollo de LPS.

Nota. Se observa las actividades esenciales para LPS. Tomado de (Northrop & Clements, Los problemas y las soluciones, junto con las características, se utilizan para parametrizar los modelos de dominio arquitectónicos y funcionales.

Análisis de Dominio Orientado a Características (FODA)

Según (Kang, Cohen, Hess, Novak, & Peterson, 1990) el método FODA permite la reutilización a nivel funcional y arquitectónico. Donde los productos de domino representan la funcionalidad y arquitectura los cuales se pueden adaptar al desarrollo de componentes software. El método FODA permite utilizar una metodología de análisis de requisitos, hasta el mantenimiento del sistema.

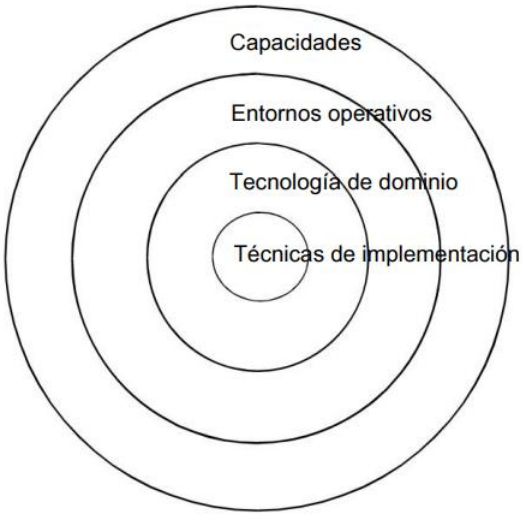
Parametrización de producto

El objetivo de la parametrización de producto es diseñar componentes genéricos que puedan adecuarse a los valores de parámetros. En donde se implementan de varias formas

como subrutinas, genéricos, marcos y procesadores, sin embargo, estas técnicas se aplican principalmente al código. Las aplicaciones en un dominio presentan un conjunto de capacidades comunes, lo que hace que cada aplicación sea diferente a las demás. Estas capacidades desde la visión de un usuario final se modelan como características. Las aplicaciones pueden ejecutarse en diferentes entornos operativos. Pueden ejecutarse en diferentes hardware o sistemas operativos, o interactuar con diferentes tipos de dispositivos. En la figura 3, se muestra que al desarrollar una aplicación esta puede tener decisiones que pueden ir variando.

Figura 3

Tipos de decisiones de desarrollo.



Nota. En la figura se observa los tipos de decisiones de desarrollo que se presentan al momento de parametrizar los modelos y funciones. Tomado de (Kang, Cohen, Hess, Novak, & Peterson, 1990)

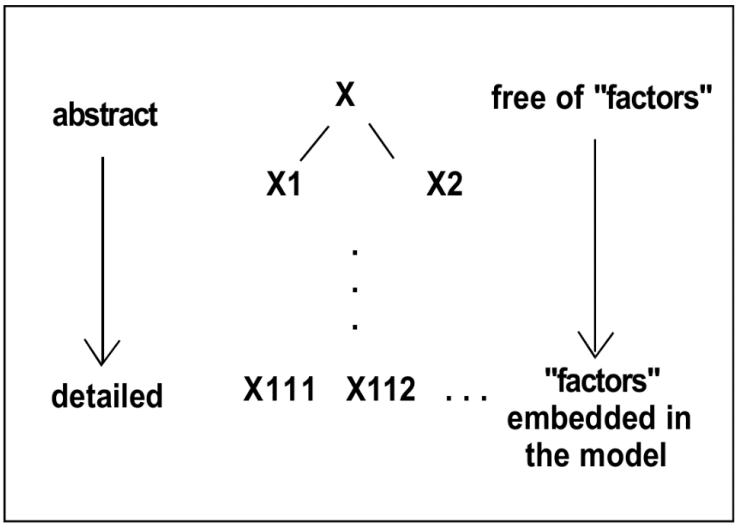
Los problemas y las soluciones, junto con las características, se utilizan para parametrizar los modelos de dominio arquitectónicos y funcionales.

Niveles de abstracción

El método FODA busca que los niveles de abstracción sean lo más básicos posibles, para esto se utiliza varios niveles de abstracción donde una característica de un nivel representa una generalización de los niveles más bajos. Este proceso se realiza hasta obtener características cada vez menos genéricas resultando así en los componentes más esenciales que componen la aplicación. En la figura 4, se puede observar los distintos niveles de abstracción, hasta llegar a los factores más básicos, los cuales vienen a ser los elementos reutilizables de la LPS.

Figura 4

Modelo de abstracción



Nota. En la figura se observa los niveles de abstracción hasta llegar a los factores más esenciales. Tomado de (Kang, Cohen, Hess, Novak, & Peterson, 1990)

Los productos que genera el dominio son las diferentes abstracciones obtenidas mediante el análisis. Toda la información recopilada se la organiza y se la representa como productos del dominio. Los modelos obtenidos a partir del análisis de dominio son utilizados para realizar los productos de la LPS, en este proceso los analistas de requerimientos determinan si el producto solicitado se encuentra dentro de la LPS; de ser así, se procede a

escoger las capacidades del sistema, basándose en las abstracciones obtenidas en el análisis de dominio realizado.

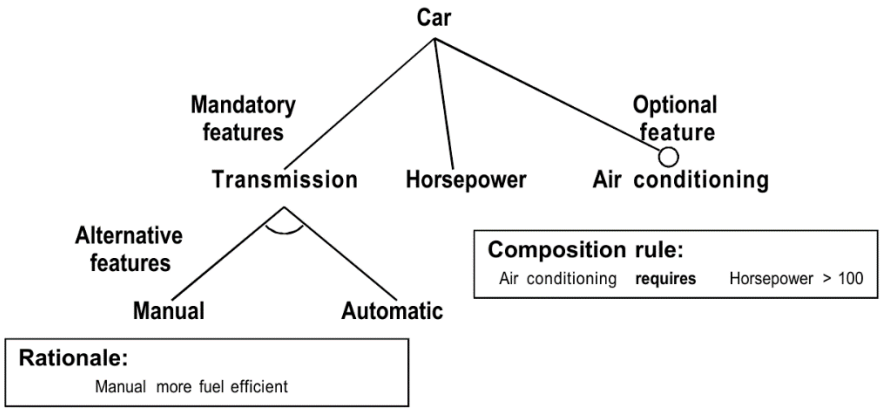
Modelamiento de dominio FODA

Para el desarrollo de software se utiliza diagramas de flujo de datos los cuales son útiles únicamente para el equipo de desarrollo, un cliente no comprendería este diagrama. Los clientes necesitan conocer los componentes y las capacidades esenciales que posee la aplicación, entre ellas: servicios de la aplicación, rendimiento de la aplicación, plataforma de hardware, entre otros. El FODA se enfoca principalmente en la perspectiva de la funcionalidad, donde se muestran las características y los niveles de abstracción, los cuales se pueden utilizar para la generación de productos o parametrizar nuevos modelos.

Las características son los atributos que son de gran importancia para los usuarios finales ya que aquí se muestra la disponibilidad de funciones que puede tener el sistema. En la figura 5, se muestra las decisiones que debe tomar una persona al comprar un auto; diagramado como un diagrama de características FODA.

Figura 5

Ejemplo diagrama de características



Nota. Ejemplo de diagrama de características mostrando la composición de un auto. Tomado de (Kang, Cohen, Hess, Novak, & Peterson, 1990)

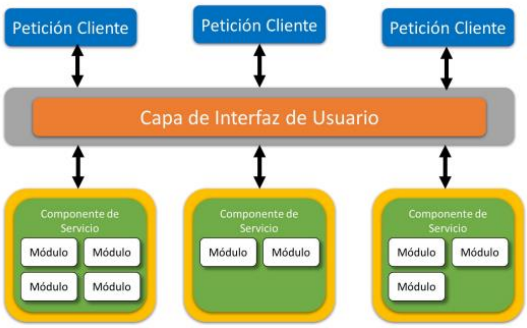
El diagrama de características muestra los diferentes componentes del dominio y la relación que existe entre ellos. Cada característica es representada por un nombre simple el cual se expande en el diccionario del diagrama. Las características pueden ser de tipo obligatoria u opcional las cuales se determinan en el diagrama como un círculo pintado y uno sin pintar como se muestra en la figura 5. La selección de características opcionales se realiza en base a los objetivos que tenga el cliente

Arquitectura de microservicios

El objetivo de microservicios para el desarrollo de aplicaciones como un conjunto de pequeños servicios, cada uno de estos son ejecutados en su propio proceso y mecanismo de comunicación, a través de protocolos HTTP mediante APIs. Estos servicios están diseñados alrededor de funcionalidades de negocio específicas, se desarrollan de manera que permite la implementación con independencia de despliegue automatizada, sin intervención manual o de herramientas y sistemas específicos. La arquitectura de microservicios fomenta el desarrollo y la implementación de aplicaciones que consisten en unidades independientes, modulares, autónomas y autocontenidas de una forma diferente a lo tradicional o monolítica como se muestra en la figura 6.

Figura 6

Ejemplo básico de arquitectura de Microservicio



Nota. Ejemplo de estructura de arquitectura de un microservicio. Tomado de (Lopez & Maya, 2017)

La arquitectura de microservicios está orientada a ser realizada en componentes, por lo tanto, permite el desarrollo de cada uno de ellos de manera independiente. De tal forma que es mucho más fácil realizar grupos de trabajo específicos para cada uno de los microservicios. En combinación con LPS, la arquitectura de microservicios nos permite representar cada uno de los features como un componente del sistema, es decir, un microservicio. Esto nos permite realizar de una forma más organizada y rápida, cada uno de los features y permite que sean mantenibles, reutilizables y escalables. La arquitectura viene a ser un core asset de la línea de producto software ya que esta se convierte en un activo fundamental a la hora de implementar los diferentes productos de LPS. Los microservicios más esenciales, es decir, los componentes del FODA más básicos son módulos reutilizables de la LPS.

Desarrollo Global de Software (DGS)

La globalización, según la Real Academia Española, se define como la tendencia de las empresas y los mercados de expandirse y crecer a escala global. Esto significa que las empresas y los mercados operan cada vez más a nivel internacional, con actividades comerciales que se extienden a través de múltiples países y regiones (Piattini, Vizcaíno, & Garcia, 2014). un proyecto DGS depende de cuatro factores las distancias geográficas, temporal y sociocultural, diferencias lingüísticas que podrían causar retrasos en entregas.

Un proyecto en DGS se puede considerar como un conjunto de varios subproyectos, repartiéndose las diferentes actividades y características que lo conforman entre los diferentes nodos o sitios que colaboran en él, existen tres tipos de distribución de tareas o trabajo: “basado en módulos” consiste en dividir el proyecto en módulos cada uno de ellos puede ser considerado como un artefacto completo y repartirlos entre los sites, “basado en fases” cada fase del desarrollo es asignado a un site o grupo de desarrollo y “follow the sun” cada grupo de desarrollo se encuentra distribuido geográficamente con el objetivo de mantener la producción las 24 horas es decir mientras un equipo duerme el otro trabaja.

Formación de equipos distribuidos

En DGS un punto muy importante es la formación de los equipos distribuidos. Estos equipos deben formarse a partir de los roles que cumple cada uno de los integrantes de los equipos. Obteniendo así equipos basados en características, es decir, equipos que buscan complementar una característica en concreto. También se debe tomar en cuenta de ser factible, los miembros de un mismo equipo pertenezcan a una misma localidad (Piattini, Vizcaíno, & Garcia, 2014). Para la organización de equipos de características se propone los siguientes métodos:

Reorganización big-bang: consiste en organizar equipos agrupando varios especialistas dentro de un mismo equipo de trabajo. Garantizando así que el equipo posea conocimientos sobre la mayoría del proceso o del sistema.

Expansión gradual de la responsabilidad: con este método se realiza pequeñas modificaciones en los integrantes del equipo. Hasta llegar a un equipo de características.

Introducción gradual de equipo de características: en este método se toman las características más importantes del producto backlog y únicamente a ellas se las aplica el cambio a equipo de características.

DevOps

DevOps es una cultura que enfatiza la cooperación en equipo y alienta a las empresas a ofrecer funciones de productos de software a través de procesos automatizados (Bijwe & Shankar, 2022).

DevOps está definida por dos palabras: desarrollo (Dev) y operaciones (Ops); es una metodología que describe formas para mejorar los procesos de una idea para pasar del desarrollo a la implementación dentro de un entorno de producción de esta forma generar un valor agregado para el cliente. Disminuyendo tiempos de entrega, esta forma describe como el equipo de desarrollo y el de operaciones están relacionados y además deben contar con una buena comunicación (Cusco, 2022).

DevOps y el ciclo de vida de las aplicaciones

DevOps afecta diversas fases del ciclo de vida de una aplicación en las etapas de planificación, desarrollo, entrega y uso, en donde cada etapa está interconectada y depende de las demás.

Planificación: La etapa de planificación en DevOps, implica que los equipos describan las características y funcionalidad de los sistemas y aplicaciones que se desarrollarán. Supervisen el progreso del proyecto, creando los trabajos retrasados y visualizando el progreso, así como rastreando errores y problemas en detalle (Microsoft, n.d.).

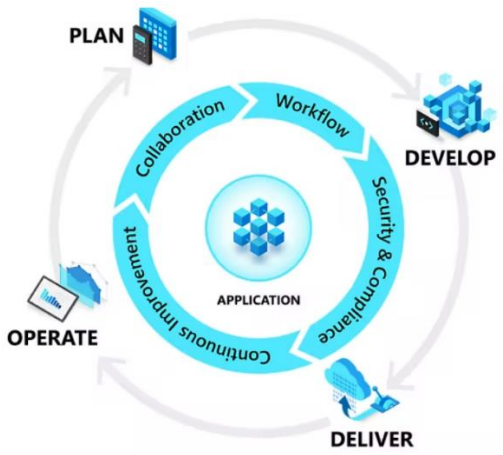
Desarrollo: El proceso de desarrollo e implementación de software, que involucra varias etapas como programación, prueba, revisión e integración de código. Los equipos tienen como objetivo optimizar este proceso para lograr eficiencia y productividad sin comprometer la calidad y la estabilidad de los sistemas. El código se compila para permitir la implementación en diferentes entornos (Microsoft, n.d.).

Entrega: se refiere al proceso de implementación de aplicaciones de manera confiable y consistente en entornos de producción. Esta fase también implica establecer y configurar la infraestructura central que conforma esos entornos al tiempo que garantiza una gobernanza adecuada. (Microsoft, n.d.).

Uso: Los equipos trabajan para asegurar la confiabilidad, la alta disponibilidad y el objetivo de ningún tiempo de inactividad del sistema, al tiempo que refuerzan la seguridad. (“¿Qué es DevOps? Explicación de DevOps | Microsoft Azure”) Buscan identificar los problemas antes de que afecten a la experiencia del cliente y mitigarlos rápidamente a medida que surgen (Microsoft, n.d.).

Figura 7

Ciclo de vida de las aplicaciones



Nota. Etapas que presenta el ciclo de vida DevOps. Tomado de (Microsoft, n.d.).

DevOps y la nube

La forma en que los equipos compilan, implementan y usan aplicaciones ha experimentado una transformación radical debido a la adopción de la tecnología de la nube, al tener la capacidad de proveer y configurar entornos en la nube de varias regiones los equipos adquieren agilidad para implementar las aplicaciones creando entornos complejos en la nube y cuando no los necesitan los apagan ofreciendo una innovación más rápida, al tener recursos flexibles, reducir costos operativos, ejecutar infraestructura con más eficacia y escalar a medida que las necesidades de negocio cambien.

SAFe 5

SAFe 5 es un marco para agilidad empresarial, el cual integra Lean, Agile y DevOps en un sistema integral, ayudando a las empresas a prosperar ofreciendo productos y servicios innovadores de forma rápida, predecible y de calidad (Scaled Agile Framework, 2021).

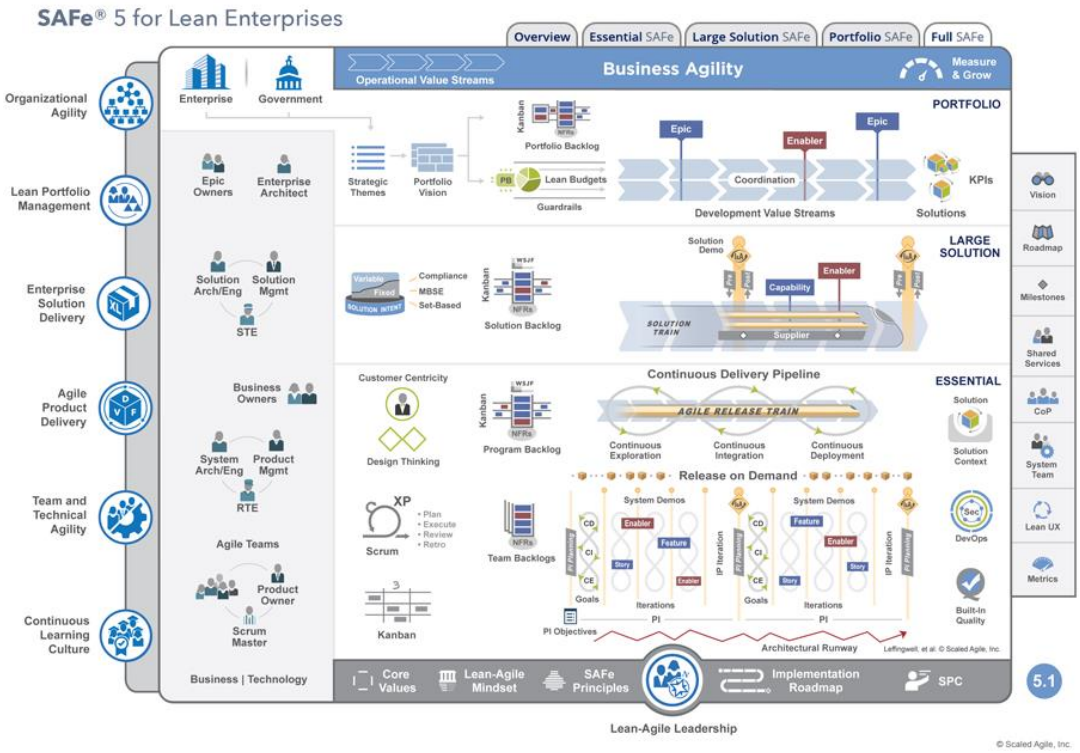
Las organizaciones pueden adaptar el marco SAFe 5 a las necesidades propias del negocio. SAFe 5 está orientado a ser escalable, lo cual permite desde el manejo de pocos equipos de trabajo, hasta los que requieren de cientos o miles de personas.

SAFe 5 es un marco muy adaptable, el cual se va actualizando continuamente, aumentando la agilidad con la que se acopla a las diferentes organizaciones.

En la figura 8, se puede observar un panorama global sobre los componentes principales involucrados dentro del marco de SAFe 5. Como principios fundamentales, valores, mentalidades, roles, artefactos y elementos de implementación que forman parte del marco SAFe 5.

Figura 8

Panorama general SAFe 5



Nota. Componente dentro del marco SAFe 5. Tomado de (Scaled Agile Framework, 2021).

Componentes de SAFe

Épicas: Se refiere a una especie de plataforma que alberga una variedad de habilidades y proyectos enfocados en desarrollar soluciones. Esta plataforma describe lo que se está haciendo para abordar un problema en particular, explica cuál es la solución propuesta y cómo se diferencia de soluciones previamente existentes.

Capacidades: Se refiere a una descripción del comportamiento general de una solución, la cual está dividida en múltiples características o elementos que son muy grandes para ser entregados en una sola etapa, por lo que son divididos en pequeñas partes.

Características: Se refiere a un servicio que cumple con la necesidad de una parte interesada y cada función incluye una hipótesis de beneficio y criterios de aceptación. Estas características se escalan o se dividen según sea necesario para lograr el lanzamiento eficiente en un entorno ágil.

Historias de Usuario: Son la herramienta principal para expresar la funcionalidad requerida de un sistema, se enfocan en el valor comercial y las necesidades del cliente por lo cual se recomienda redactar de la siguiente manera: Como (rol de usuario), quiero (actividad), para (valor comercial).

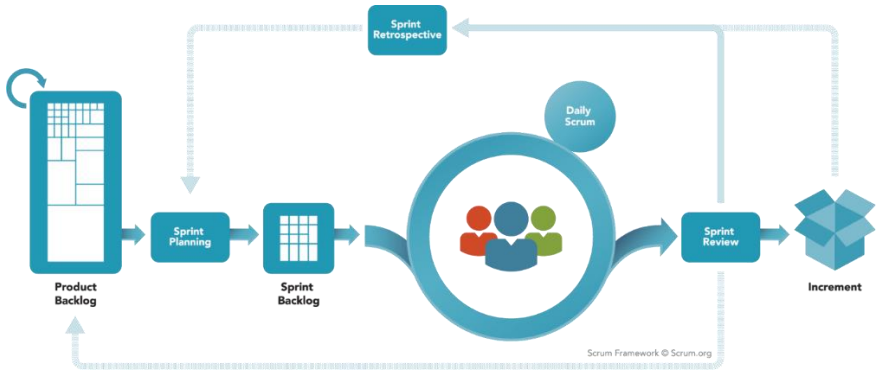
Scrum

Scrum es una forma de hacer el trabajo en equipo en pequeñas piezas a la vez, con experimentación continua y bucles de retroalimentación en el camino para aprender y mejorar a medida que avanza (Scrum, 2023).

La metodología scrum permite la gestión de proyectos software de una manera rápida y ágil. Scrum se basa principalmente en: la observación, la experiencia y la experimentación. Scrum trabaja de manera iterativa lo que quiere decir que: la metodología va dando incrementos dentro de los ciclos cortos llamados sprints, realizando así retroalimentación continua y adaptación a los procesos de entrega.

Figura 9

Funcionamiento de scrum



Nota. En la figura se muestra los procesos que forman parte de scrum. Tomado de (Scrum, 2023).

Trabajos relacionados

Gestión de Configuración y Línea de Productos para Mejorar el Proceso Experimental en Ingeniería del Software*:* La Ingeniería del Software (IS) Empírica adopta el método científico a la IS para facilitar la generación de conocimiento. Una de las técnicas empleadas, es la realización de experimentos. "Para que el conocimiento obtenido experimentalmente adquiera el nivel de madurez necesario para su posterior uso, es necesario que los experimentos sean replicados." ("Gestión de configuración y línea de productos para mejorar el proceso ...") Para adoptar la GCS a experimentación, se comienza realizando un estudio del proceso experimental como transformación de productos; a continuación, se realiza una adopción de conceptos fundamentada en los procesos del desarrollo software y de experimentación; finalmente, se desarrollan un conjunto de instrumentos, que se incorporan a un Plan de Gestión de Configuración de Experimentos (PGCE). Para adoptar la LPS a experimentación, se comienza realizando un estudio de los conceptos, actividades y fases que fundamentan la LPS; a continuación, se realiza una adopción de los conceptos; finalmente, se

desarrollan o adoptan las técnicas, simbología y modelos para dar soporte a las fases de la Línea de Producto para Experimentación (LPE) (Espinosa Gallardo, 2014)

Caso de Aplicación para Crear Tiendas Virtuales Usando Líneas de Productos de Software*:* La ingeniería de líneas de productos de software es un paradigma que propone la reutilización planificada, para aprovechar elementos comunes y variables que tienen productos de software que pertenecen a un mismo dominio. El paradigma de las líneas de productos de software puede ser aplicado al contexto de las tiendas virtuales, pues existen elementos que podrían ser capitalizados para elaborar una familia de tiendas virtuales en lugar de crear solo una (Rincón, Rodríguez, Martínez, & Pabón, 2015).

"Subconjuntos Mínimos de Corrección para explicar características muertas en Modelos de Líneas de Productos." ("Subconjuntos mínimos de corrección para explicar características ...") El caso de los Modelos de Características: Las líneas de productos son un paradigma que permite administrar eficientemente un conjunto de productos con elementos comunes y variables que pertenecen a un dominio en particular. ("Subconjuntos Mínimos de Corrección para explicar características ...") Este paradigma ofrece beneficios como la reutilización, la disminución de errores y la disminución de tiempos y costos de producción. Los beneficios propuestos para las líneas de productos pueden ser extensibles al software, pues en el desarrollo de software es necesario administrar la reutilización y la variabilidad, este paradigma aplicado al software se conoce como Líneas de Productos de Software. (Rincon et al., 2013)

Aplicación De La Refactorización De Software Con La Herramienta Foda: La metodología Feature Oriented Análisis de Dominio Orientado a Características (FODA) logrando software exitoso para refactorizar el código y reutilización. Define los siguientes conceptos: Dominio: Conjunto de aplicaciones actuales y futuras que comparten un conjunto de capacidades y datos comunes. Análisis de dominio: El proceso de identificar, recopilar organizar y representar la información relevante de un dominio basado en el estudio de los

sistemas existentes y su desarrollo los conocimientos históricos adquiridos a partir de las expectativas de dominio, la teoría subyacente y la tecnología emergente dentro del dominio. teoría subyacente y la tecnología emergente dentro del dominio. Característica: Aspecto, cualidad o característica prominente o distintiva, visible para el usuario, de un sistema o sistemas informáticos. o característica de un sistema o sistemas de software (Malathi & Sudhakar, 2018).

Metodologías de Desarrollo de Software: En la actualidad la rapidez y el dinamismo en la industria del software han hecho replantear los cimientos sobre los que se sustenta el desarrollo de software tradicional. (“Metodologías de desarrollo de software | DSpace-CRIS @ UCA”) Estudios recientes y el mismo mercado actual está marcando la tendencia en la ingeniería del software teniendo como características principales atender a las necesidades de rapidez, flexibilidad y variantes externas que hacen de nuestro entorno una ventaja más competitiva al aumentar la productividad y satisfacer las necesidades del cliente en el menor tiempo posible para proporcionar mayor valor al negocio. Ante esta situación, el grado de adaptación de las metodologías tradicionales a estos entornos de trabajo no eran del todo eficientes y no cubrían las necesidades del mercado actual. En la actualidad existen una gran cantidad de metodologías para el desarrollo de software, separadas en dos grandes grupos; las metodologías tradicionales o pesadas y las metodologías agiles (Maida & Pacienzia, 2015). (“Metodologías de desarrollo de software | DSpace-CRIS @ UCA”)

Aplicación de la metodología de desarrollo ágil Scrum para el desarrollo de un sistema de gestión de empresas: Este proyecto describe el desarrollo de un sistema de gestión de empresas, también conocido como ERP. El cual se desarrolla empleando la metodología ágil Scrum. (“Aplicación de la metodología de desarrollo ágil Scrum para el ... - UC3M”) El proyecto se divide en varias iteraciones en las cuales se obtiene como resultado un prototipo. El sistema está compuesto por una aplicación cliente para uso del usuario y una base

de datos centralizada donde se almacenarán todos los registros relativos a la empresa (Urteaga Pecharromán, 2015) (“Escrum - CASO PRÁCTICO DE MODELO SCRUM - StuDocu”)

"Técnicas de análisis de dominio: organización del conocimiento para la construcción de sistemas software*." (“Técnicas de análisis de dominio: organización del conocimiento para la ...”)* La comunidad dedicada al diseño de sistemas informáticos ha propuesto una serie de técnicas conocidas como “análisis de dominio”, que tienen como finalidad la captura y adquisición del conocimiento que deben almacenar y procesar las aplicaciones informáticas orientados a una misma función o uso. El análisis de dominio ofrece herramientas para capturar la información crítica sobre las entidades, datos y procesos característicos de un área de actividad, y facilitar la especificación, el diseño y la construcción de los sistemas que soporten dichas actividades. Como disciplina, el análisis de dominio pretende la reutilización intensiva del conocimiento y capitalizar la experiencia adquirida en el diseño y construcción de sistemas informáticos de un mismo tipo (López Hinojosa, 2017) (“Dialnet-Tecnicas De Analisis De Dominio-2533323”)

*A***rquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web. (“Arquitectura de Software basada en Microservicios para ... - RedCLARA”)** Actualmente, el proceso de desarrollo de software que realiza la Coordinación General de Tecnologías de la Información y Comunicación (CGTIC) de la Asamblea Nacional del Ecuador (ANE) constituye el empleo de una arquitectura de software tradicional o monolítica que ha sido adoptada del lenguaje de programación utilizado, la plataforma o de la experiencia del personal del área de desarrollo; por el aspecto monolítico, este tipo de aplicaciones empaquetan toda la funcionalidad en una sola y gran unidad ejecutable (un solo archivo o aplicación), lo que ha provocado dificultades en aspectos como mantenimiento, escalabilidad y entregas. El objetivo del presente estudio fue identificar las tecnologías, metodología y arquitectura que utiliza la CGTIC para el desarrollo de aplicaciones web y la correspondiente identificación de las tecnologías existentes para el desarrollo e implementación

de microservicios, utilizando como base de la investigación un enfoque cualitativo, con un tipo de investigación descriptiva y diseño documental (López Hinojosa, 2017) (“Repositorio Digital Universidad Técnica del Norte: Arquitectura de ...”)

A*utomatización de los procesos de construcción de software mediante la aplicación de técnicas DEVOPS*: Partiendo de una aplicación monolítica con procesos de construcción manuales, en este trabajo se desarrollan pruebas para verificar su correcto funcionamiento para posteriormente evolucionar dicha aplicación, separarla en pequeños servicios independientes y automatizar su despliegue empleando técnicas DevOps. Se lleva a cabo un ciclo completo de software, centrado en la construcción del entorno de trabajo, gestión de repositorios Git, orquestación de servicios mediante contenedores, testing, integración continua (IC) y, por último, monitorización (Fernández, 2022).

Aplicación basada en arquitectura de microservicios: El principal objetivo de este trabajo es implementar una arquitectura de microservicios con una malla de servicios (service mesh). Hablaremos de las ventajas y desventajas de una arquitectura de microservicios frente a una monolítica, y las ventajas que ofrecen las mallas de servicio a nivel seguridad, conectividad, observabilidad y control. (“Aplicación basada en arquitectura de microservicios”) Los servicios de este proyecto han sido implementados con los diferentes lenguajes que predominan hoy en día, como Python, Java, PHP o Spring Boot, entre otros. Además, contienen varias APIs REST y una base de datos SQL. En definitiva, tratamos los principales paradigmas de los desarrolladores DevOps (Diego Navarro et al., 2020)

Conclusiones

Se definió con éxito todos los conceptos que se toman en cuenta dentro del proyecto, también se revisó una lista de varios trabajos que se encuentran relacionados con el presente proyecto.

Capítulo III

Implementación del sistema

Introducción

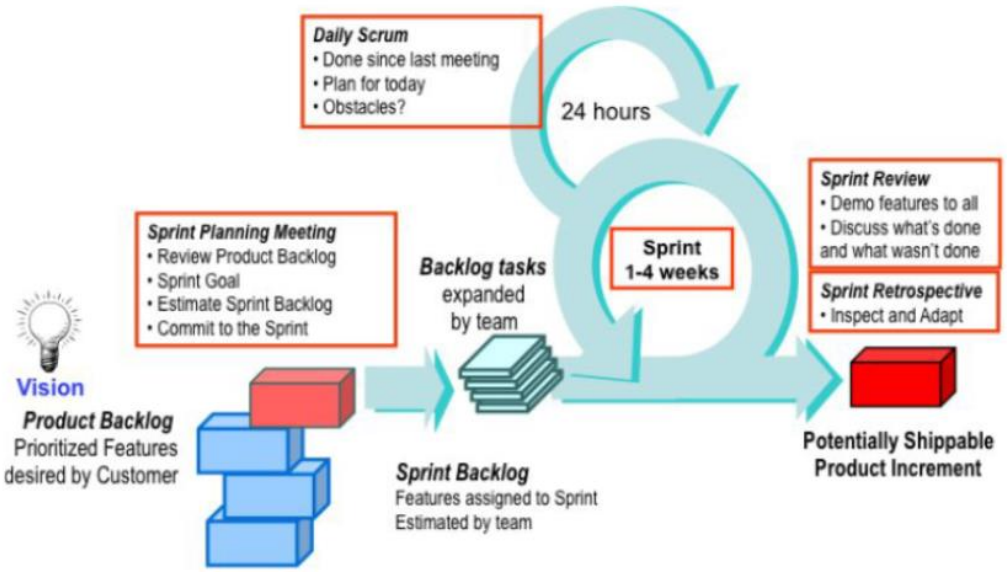
En este capítulo se presenta el procedimiento realizado para la implementación de las características pertenecientes al área de PetGrooming los cuales abarcan el agendamiento de turnos, la gestión de Spa y hospedaje, mediante una arquitectura de microservicios. Estas características representan una parte del Sistema de Gestión de Clínicas Veterinarias (SGCV) general.

Metodología

"SCRUM es un marco de trabajo iterativo e incremental para el desarrollo de proyectos, productos y aplicaciones." (“SCRUM - Tesis - paulocesar30”) Está compuesta, de manera general, por tres roles, cuatro eventos y tres artefactos. En la figura 10, se presenta de forma detallada los cuatro eventos que forman parte del marco de trabajo scrum 1) Sprint Planning: determina el objetivo del Sprint y el trabajo que se va a desarrollar en el mismo, 2) Daily Scrum: consiste en una reunión diaria de 15 minutos para planificar lo que se va a realizar durante todo el día, 3)Sprint Review: se revisa el trabajo realizado, el avance obtenido y los cambios en el Product Backlog, finalmente se encuentra el 4) Sprint Retrospective: donde el equipo de trabajo se autoevalúa y planea mejoras para el trabajo. Además, se muestra los tres artefactos: Product Backlog: contiene una lista ordenada de todas las tareas que se pretenden hacer durante el proyecto, Sprint Backlog: representa las tareas o actividades que el equipo planea realizar durante un sprint del proyecto y Burndown Chart: es la gráfica del avance del equipo sobre el sprint y permite conocer de forma clara si el equipo logrará completar el trabajo en el tiempo establecido.

Figura 10

Marco de Trabajo Scrum.



Nota. En la figura 10 se muestra de forma gráfica el marco de trabajo Scrum.

Los tres roles que se definen dentro de la metodología Scrum son: Product Owner, Equipo de Desarrollo y Scrum Máster. En la tabla 1, se observa los roles, personas y funciones que conforman la metodología aplicada para el desarrollo del proyecto.

Tabla 1

Roles Scrum.

Rol	Integrante	Función
Product Owner	Dr. Santiago Patricio Jácome Guerrero	Determina el cumplimiento de las actividades del proyecto y garantizar que el producto se entregue funcional, correctamente validado y que cumpla con los requerimientos establecidos.
Scrum Máster	Paola Lissette Romo Gavilánez	Encargado de que los equipos de trabajo alcancen los objetivos hasta llegar a la fase final del sprint.

Rol	Integrante	Función
Team Development	Ana Lissette Sánchez Pico Paola Lissette Romo Gavilánez	Personas encargadas del análisis, diseño, desarrollo, despliegue y pruebas del sistema

Nota. Los roles y funciones se han asignado de acuerdo con las habilidades de cada miembro del equipo.

Análisis del Sistema

Luego de que se asignaron los roles y funciones de las personas que integran el equipo de trabajo se realiza una reunión con el fin de determinar las clínicas veterinarias que se va a visitar, las técnicas que se van a utilizar para el levantamiento de información y los marcos de trabajo y modelado del sistema.

Especificación de Requerimientos

Se estableció como técnica de obtención de requerimientos realizar entrevistas, para las mismas se seleccionó a cuatro clínicas veterinarias ubicadas en la ciudad de Ambato, en este proyecto no centraremos en dos, el equipo de trabajo visitó las clínicas veterinarias Royal Hound [1]y Vital Pet[2], las entrevistas realizadas se encuentran en el Anexo 1, posterior a ello se tuvo una reunión con los médicos veterinarios a cargo de estas. Con la información obtenida se procede a realizar el primer borrador de los requerimientos solicitados. Al notar que la información no es suficiente para conocer la línea de negocio de las veterinarias, el equipo de trabajo acuerda nuevamente una reunión, pero esta vez se lleva un Paciente para ser partícipes de los procesos que se realizan dentro de la veterinaria. Luego de ello se procede

[1] Royal Hound
Clínica Veterinaria enfocada en cirugías
Médico a cargo: Dr. Edgar Patricio Lizano
Telefono:0995571188

[2] Vital Pet
Medicina de Especialidad, Centro quirúrgico especializado, diagnostico por imágenes, Laboratorio Clínico
Médico a cargo: Dr. Darwin Villamarín
Teléfono: 0984788705

hacer el segundo borrador de los requerimientos, sin embargo, para constatar la información, se agenda una última reunión con el fin de validar que los datos obtenidos son correctos.

Se programó una reunión virtual con uno de los veterinarios obteniendo así el dominio del problema sobre el cual se plantea el sistema propuesto. En la tabla 2, se lista los diferentes procesos generales que realizan las clínicas veterinarias y tienen en común. En este trabajo se abordará los procesos resaltados en color amarillo.

Tabla 2

Procesos realizados en clínicas veterinarias.

Veterinaria	Procesos que realiza
Royal Hound	- Consultorio - Cirugía - Laboratorio - Imagenología - **Spa** - **Hospedaje** - **Gestión Paciente** - **Gestión Tutor**
Vital Pet	- Consultorio - Cirugía - Imagenología - Laboratorio Clínico - **Spa** - **Hospitalización** - **Gestión Paciente** - **Gestión Tutor**

Nota. Se listan los procesos que se realizan dentro de cada clínica veterinaria.

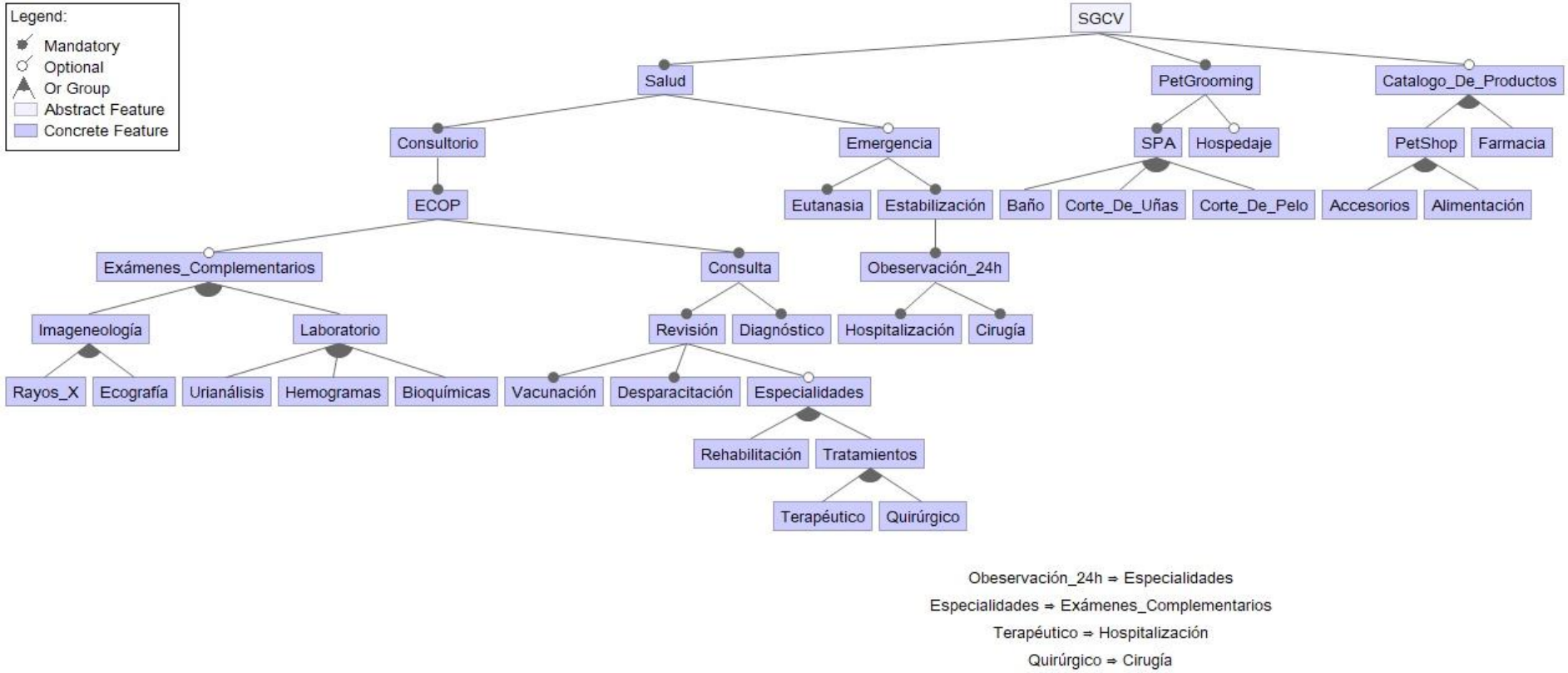
Modelado del dominio

Al tener el dominio del problema, se realiza el diagrama de características mediante la utilización de FODA, en el cual se estructura los diferentes procesos propuestos en la tabla 2. Tomando en cuenta los comunes y opcionales que posee cada clínica veterinaria. La

herramienta para el diseño del diagrama fue Lucidchart, utilizando su versión gratuita, la misma es fácil de utilizar, tiene los elementos que se necesitan para un diagrama FODA, aquí se iba modelando a manera de borrador las versiones del gráfico. Para el primer borrador del diagrama de características se obtuvo redundancia en las características del sistema y no se tomó en cuenta las relaciones jerárquicas. Se realiza un diagrama demasiado básico ya que no se abarca completamente todas las características a desarrollar ver Anexo 2. Para el tercer borrador se tuvo una idea más clara de la línea de negocio de las clínicas veterinarias sin embargo no se reconoce de forma adecuada las relaciones jerárquicas y se tiene dificultad sobre la terminología que se debe aplicar ver Anexo 3**.** Finalmente se realiza una entrevista con un veterinario y se obtiene la versión final como se muestra en la figura 11, que cumple con los procesos que se realizan en las veterinarias, además de usar la terminología apropiada.

Figura 11

Diagrama de características del SGCV.



Nota. Se presenta el diagrama completo aplicado para el SGCV.

Figura 12

Diagrama de características correspondiente a PetGrooming.

Nota. Se detalla el diagrama de características correspondiente al sistema propuesto siendo este parte del sistema general.

Marco de Trabajo Ágil

En este trabajo se presenta el desarrollo de la Épica PetGrooming en la figura 12, se observa que la épica se divide en dos características principales Spa y hospedaje, de las mismas se procede a determinar las historias de usuario para el desarrollo.

Las épicas, características e historias de usuario se construyen en base a una plantilla que entrega SAFe. En la tabla 3 se puede observar todos los componentes que integran la épica. En la tabla 4 y 5 se observa todos los campos que se deben llenar en las características de Spa y Hospedaje respectivamente.

Tabla 3

Épica PetGrooming.

	Hipótesis Épica
Fecha de Inicio	19 de septiembre del 2022
Nombre	Servicios de ***"PetGrooming"*** para clínicas veterinarias
Coordinador	Paola Romo
	Descripción
Para	Clínicas Veterinarias
Quien	Veterinarias que quieren automatizar sus procesos internos
La/El	Sistema de gestión de clínicas veterinarias.
Que es	Aplicación Web Progresiva (PWA) basado en línea de producto software.

Hipótesis Épica	
Esto	Permite agendar turnos para los servicios de Spa y hospedaje.
Diferente a	Vetpraxis
Nuestra Solución	Se utiliza para el veterinario gestione los turnos para una mejor administración del Spa y Hospedaje evitando colapso de cliente y molestias.

Resultados Comerciales	• Los veterinarios pueden acceder al sistema para consultar y agendar turnos. • El veterinario puede consultar los turnos programados para cada día.
Indicadores Principales	• Aumento de reservas de turnos para servicios de Spa y hospedaje. • Aumento de Pacientes en la clínica. • Número de turnos diarios.
Requerimientos No funcionales	• La seguridad debe estar basada en roles de usuario. • El sistema debe estar disponible 24/7.

Nota. Se presenta de forma detallada la Épica PetGrooming.

Tabla 4

Característica: Spa.

Característica	Hipótesis de beneficio
Administración del área de Spa de la clínica veterinaria	El usuario puede agendar una cita para diferentes servicios que requiera su mascota.
Criterio de aceptación	
• El Tutor puede agendar una cita para que su mascota reciba los servicios de baño, corte de uñas y corte de pelo. • El Tutor puede reagendar su cita modificando la hora y fecha a una disponible. • El veterinario puede revisar la lista de las mascotas agendadas. • El veterinario puede cancelar, modificar y agendar citas • El veterinario puede dar por finalizada las citas.	

Nota. Se define los detalles que abarcan la característica Spa.

Tabla 5

Característica: Hospedaje.

Característica	**Hipótesis de beneficio**
Gestión de Hospedaje	Permite al veterinario administrar y dar seguimiento a los huéspedes.
Criterio de aceptación	
• El veterinario puede administrar el ingreso y salida de los huéspedes.	

Nota. Se define los detalles que abarcan la característica de Hospedaje.

Parte de SAFe es la descripción de historias de usuario, las mismas que nacen a partir de las características. En la tabla 6, se describe las historias de usuario que se utilizaron para el desarrollo de los microservicios Tutor, Paciente, Spa, Gestión de turnos, Huésped, Hospedaje y Autenticación.

Tabla 6

Historias de Usuario.

Código	**Rol**	**Descripción**	**Responsables**	**Resultado**
HU001	Como usuario	**Quiero** registrar, actualizar, listar, eliminar un Tutor, **para** llevar un listado ordenado de los tutores a administrar.	Paola Romo y Ana Sánchez	El usuario podrá crear, actualizar, buscar y eliminar un Tutor (Dueño del Paciente).
HU002	Como usuario	**Quiero** registrar, actualizar, listar, eliminar un Paciente, **para** llevar un listado ordenado de los Pacientes a administrar asociarlos con su respectivo Tutor.	Paola Romo y Ana Sánchez	El usuario podrá crear, actualizar, buscar y eliminar un Paciente.
HU009	Como usuario	**Quiero** crear, listar, eliminar un turno, **para** brindar una atención ordenada de los diferentes servicios de la veterinaria.	Paola Romo y Ana Sánchez	El usuario podrá asignar un turno al Tutor para reservar un servicio de Spa u hospedaje
HU010	Como usuario	**Quiero** crear, actualizar, listar, eliminar un Spa, **para** llevar un listado ordenado de los turnos asignados para brindar el servicio de Spa.	Paola Romo y Ana Sánchez	El usuario puede crear, actualizar, buscar y eliminar las solicitudes de Spa.

Código	Rol	Descripción	Responsables	Resultado
HU011	Como usuario	**Quiero,** crear, listar, eliminar un huésped, **para** llevar un listado ordenado de los huéspedes que se encuentran en el hospedaje	Paola Romo y Ana Sánchez	El usuario puede crear, actualizar, buscar y eliminar un huésped.
HU012	Como usuario	**Quiero,** crear, listar, actualizar, eliminar las habitaciones del Hospedaje **para** conocer que habitaciones encuentran disponibles en el hospedaje.	Paola Romo y Ana Sánchez	El usuario puede crear, actualizar, buscar y eliminar las habitaciones.
HU013	Como usuario	**Quiero,** crear, actualizar, listar, eliminar una ficha de registro de Spa, **para** llevar un listado ordenado de las fichas de registro de Spa.	Paola Romo y Ana Sánchez	El usuario puede crear, actualizar, buscar y eliminar las fichas de registro.
HU015	Autenticación	**Quiero**, mantener un control de roles al ingresar al sistema **para** tener un registro ordenado de los médicos que ingresan al sistema.	Paola Romo y Ana Sánchez	El usuario puede manejar roles en el sistema.

Nota. Con las historias de usuario se puede definir de manera más detallada las actividades a realizar.

Para la planificación de actividades que se va a realizar se realiza el Product Backlog. En la tabla 7, se detalla la estimación en días para el desarrollo de cada historia de usuario la fecha de inicio, fecha fin, número de sprint y actividades que se realizó en cada historia de usuario. Todas las historias de usuario contienen las mismas actividades con excepción de la HU015 las cuales se detallan de forma individual para la historia antes mencionada.

Tabla 7

Product Backlog.

Historia de Usuario	Estimación	Fecha Inicio	Fecha Fin	Sprint	Actividades Realizadas
HU001	8	7/10/2022	14/10/2022	1	- Creación de Tablas en base de datos - Creación de Repositorio y ramas - Conexión de base de datos - Creación del proyecto e instalación de dependencias - Desarrollo de rutas, controladores. - Creación de validaciones. - Pruebas en Postman. - Despliegue de microservicios. - Pruebas de Despliegue
HU002	8	20/10/2022	27/10/2022	1	
HU009	5	2/11/2022	6/11/2022	2	
HU010	5	15/10/2022	19/11/2022	2	
HU011	3	28/11/2022	30/12/2022	2	
HU012	3	11/12/2022	14/12/2022	2	
HU013	3	24/12/2022	26/12/2022	2	
HU015	13	6/1/2023	18/1/2023	3	- Instalación de Framework Indentity Server - Configuración de protocolo oauth2.0.

Nota. Se definen los tiempos aproximados de desarrollo

Diseño del Sistema

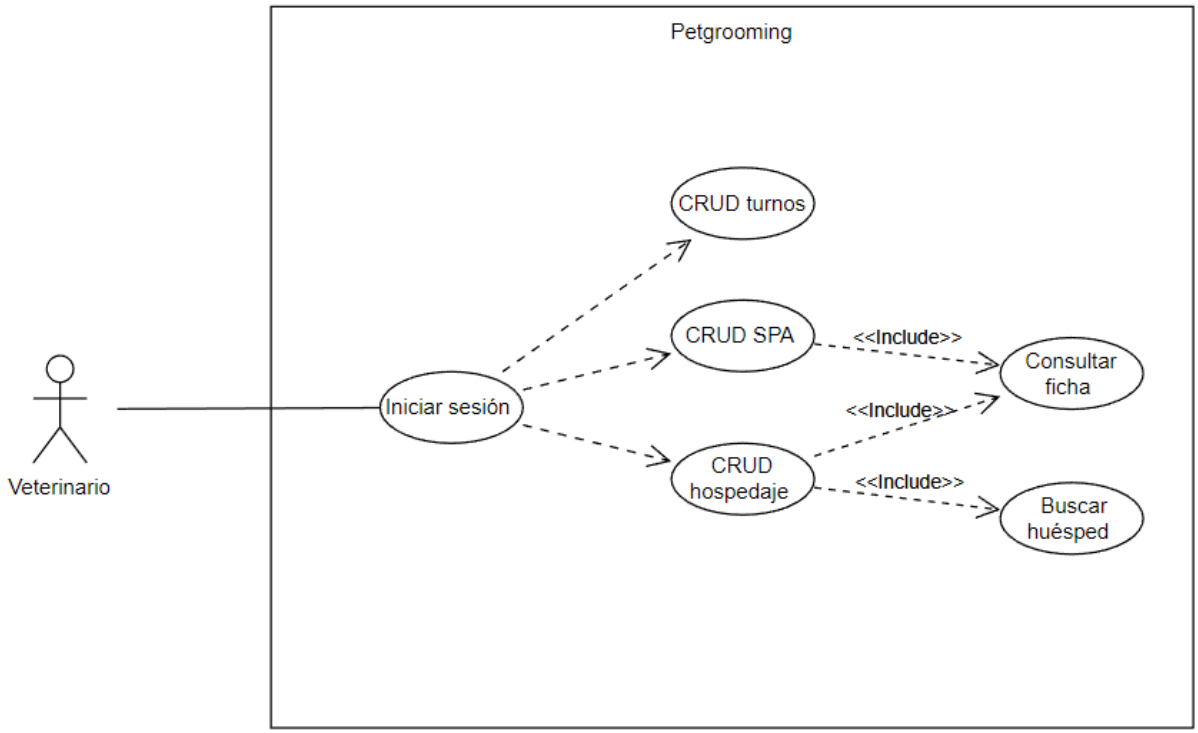
En este capítulo se presentan los diagramas de casos de uso, secuencia, clases, componentes y despliegue utilizados para el modelado de los microservicios propuestos, además se presenta la arquitectura aplicada para el despliegue de estas.

Modelado UML

Diagrama de Casos de uso. La figura 13 representa la interacción al momento de ingresar sesión y acceder a los CRUDS asignados y los distintos procesos. Los diagramas de casos de uso extendidos se encuentran en el Anexo 4.

Figura 13

Diagrama general de casos de uso.

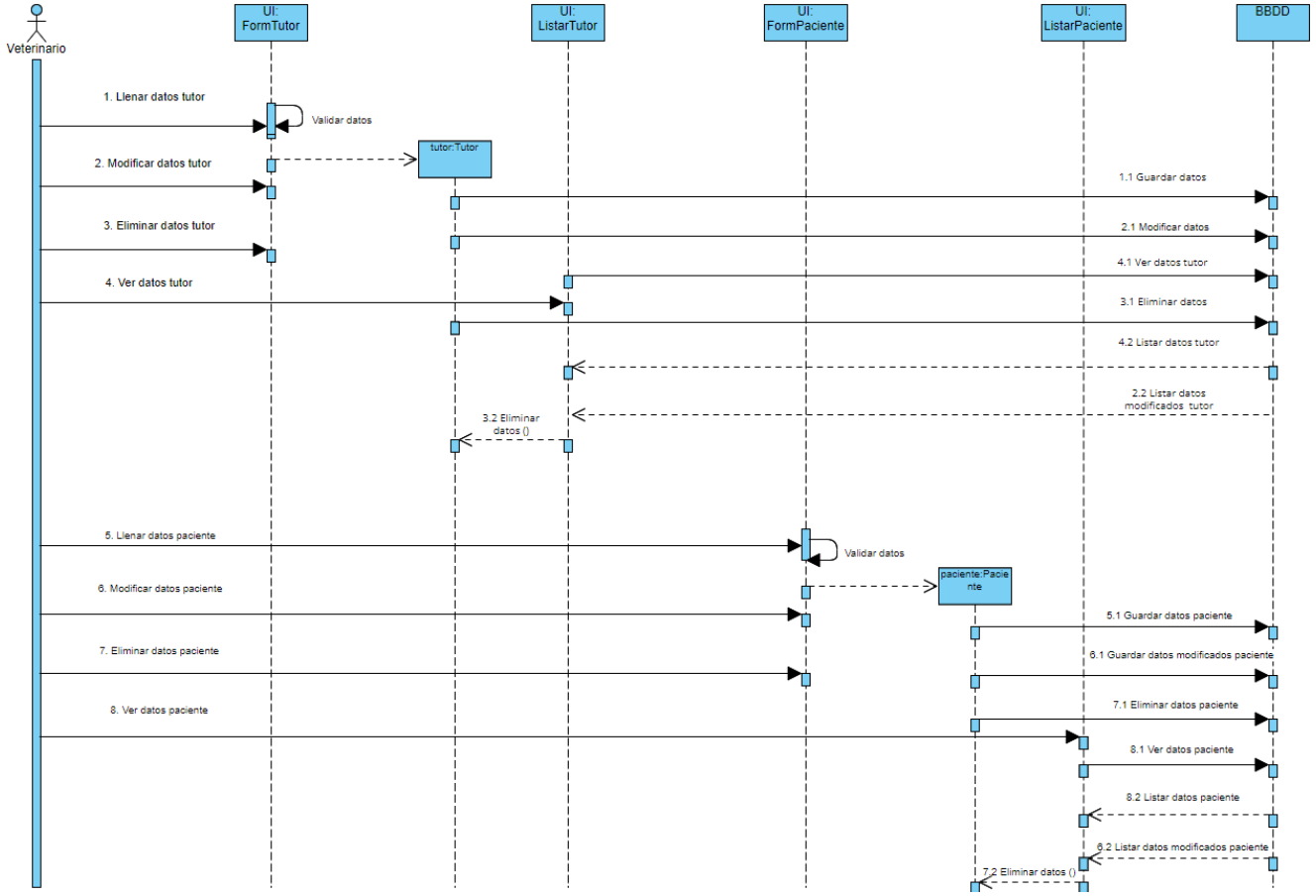


Nota. Diagrama de caso de uso de la épica Petgrooming

Diagrama de Secuencia. Se elabora el diagrama de secuencia perteneciente al caso de uso Paciente – Tutor, en la figura 14, se encuentra la interacción de cada objeto. La URL de acceso al diagrama se encuentra en https://online.visual-paradigm.com/share.jsp?id=323235363738332d33

Figura 14

Gestión Paciente

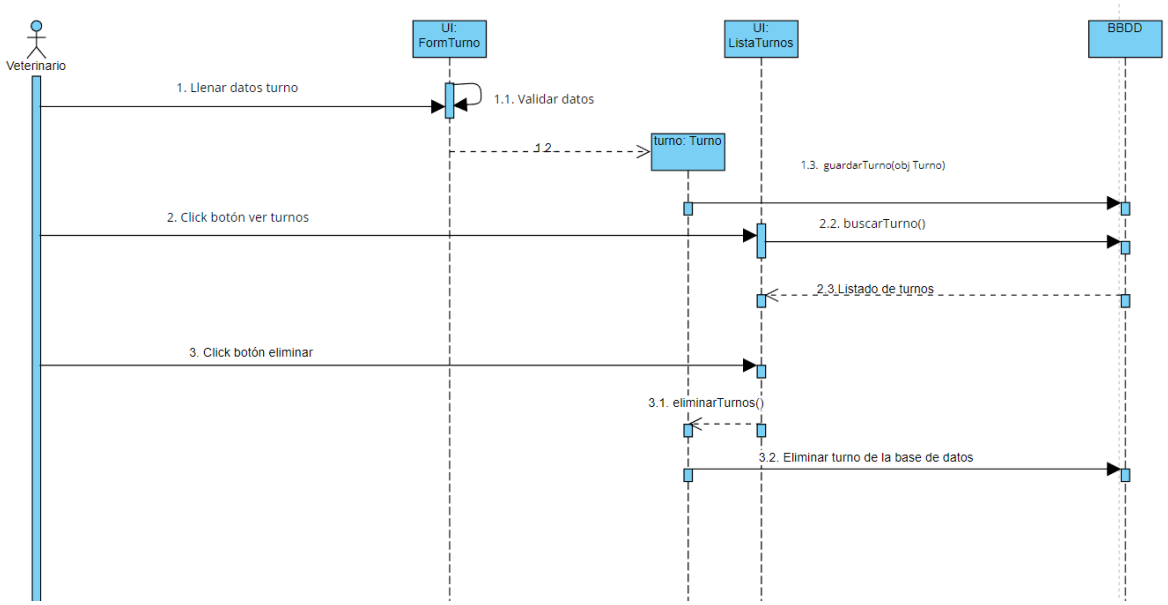


Nota. Diagrama de secuencia perteneciente a gestión PacienteTutor

Se elabora el diagrama de secuencia perteneciente a la gestión de turnos, en la figura 15, se encuentra la interacción de cada objeto

Figura 15

Gestión de turnos.

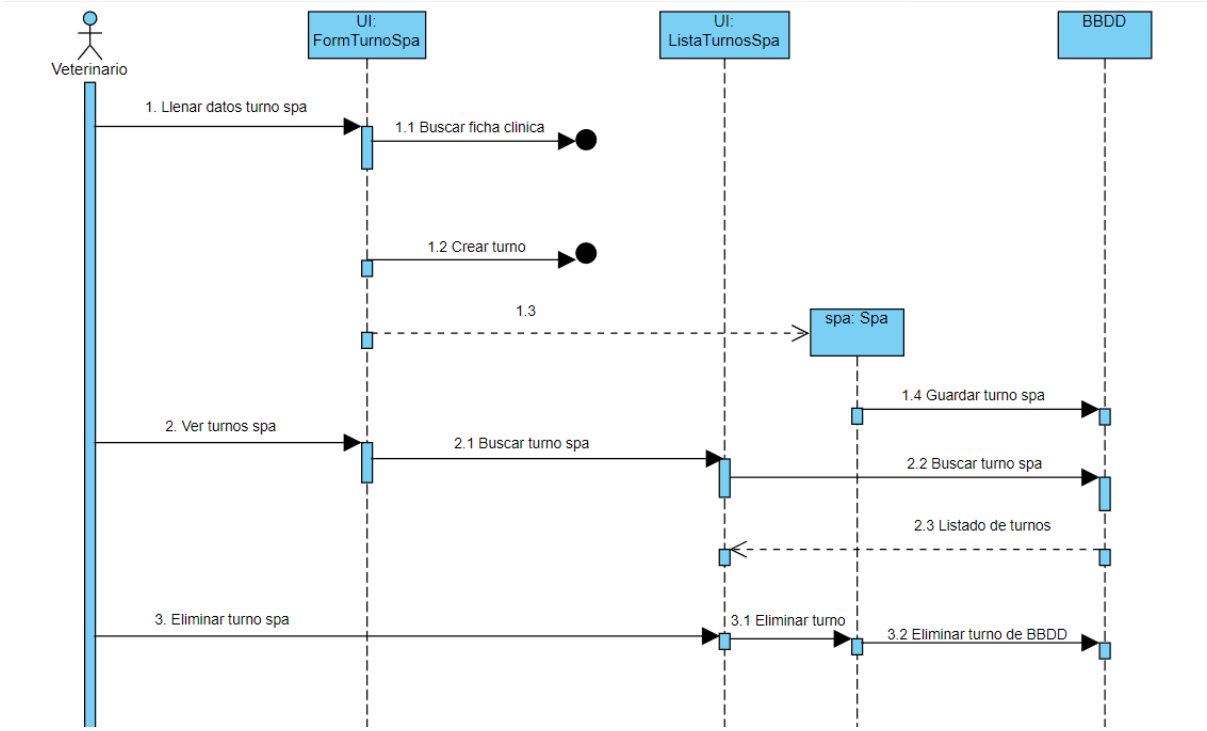


Nota. Diagrama de gestión de turnos.

Se elabora el diagrama de secuencia perteneciente a la gestión de Spa, en la figura 16, se encuentra la interacción de cada objeto.

Figura 16

SPA

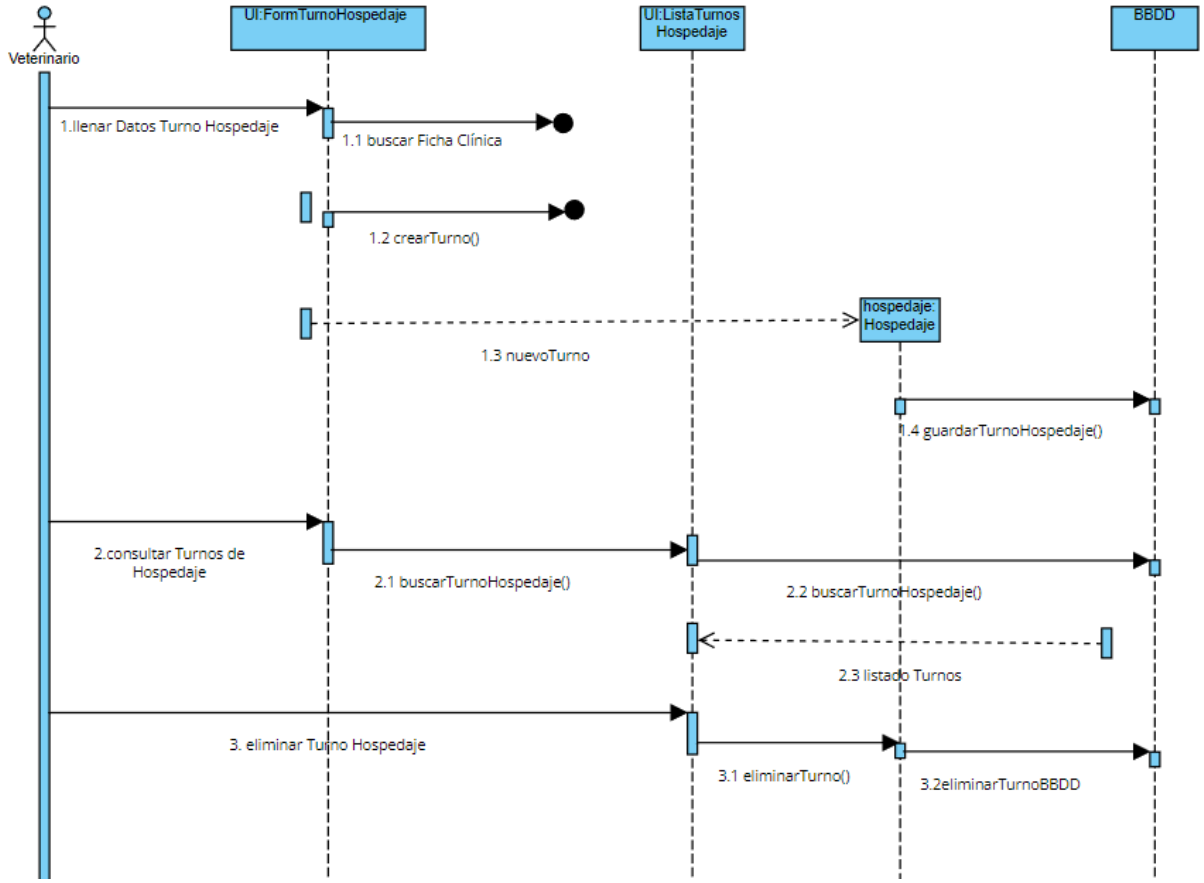


Nota. Diagrama de secuencia perteneciente a Spa. La URL de acceso al diagrama se encuentra en https://online.visual-paradigm.com/share.jsp?id=323235363835342d31

En la figura 17, se observa el diagrama de secuencia perteneciente a la gestión de hospedaje, aquí se encuentra la interacción de cada objeto. La URL de acceso al diagrama se encuentra en: https://online.visual-paradigm.com/share.jsp?id=323235363738322d38

Figura 17

Hospedaje

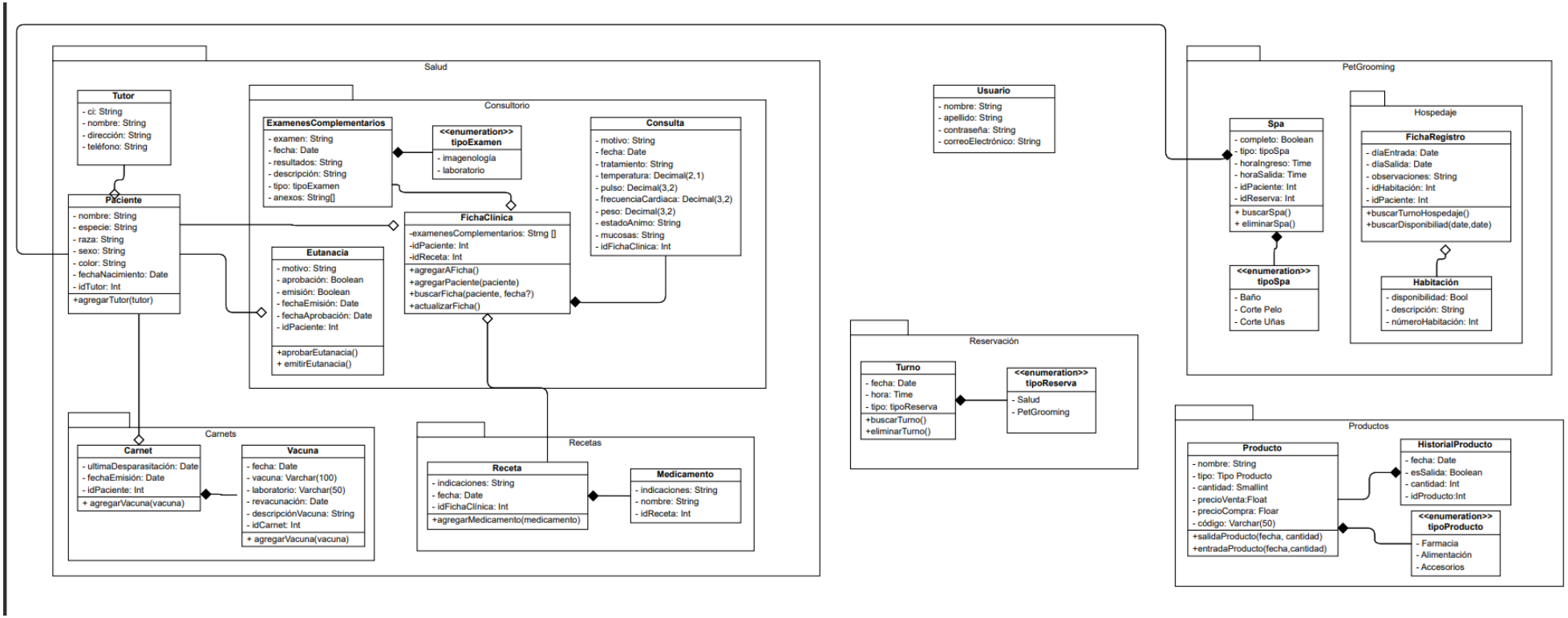


Nota. Diagrama de secuencia de Hospedaje

Diagrama de Clases. *En la figura 18 se muestra de forma detallada el diagrama de* clases, con sus respectivos campos y relaciones, una vez elaborado el mismo se podrá construir el diagrama entidad relación, que se lo vera más adelante.

Figura 18

Diagrama de Clases SGCV

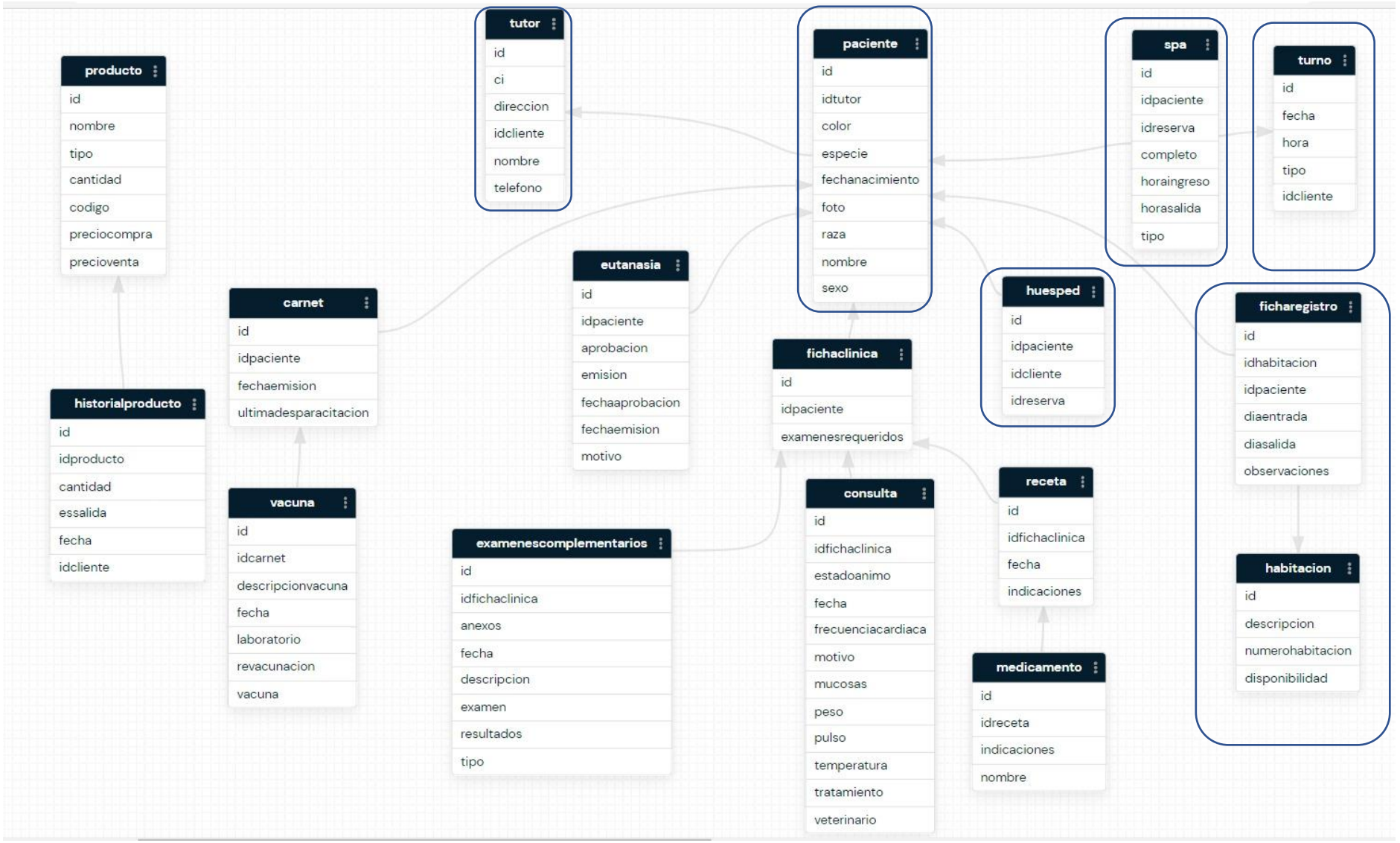


Nota. En este trabajo se abordará las tablas de reservación, usuario y Petgrooming, pertenecientes a esta tesina

Diagrama de Entidad – Relación. *En la figura 19 se observa el diagrama entidad* relación, el mismo nace del diagrama de casos de uso, este grafico se ha generado a partir de la programación de la base de datos. En el mismo se aprecia claves primarias, claves foráneas, las mismas que se han implementado en el código.

Figura 19

Diagrama entidad relación de las historias de usuario asignados.

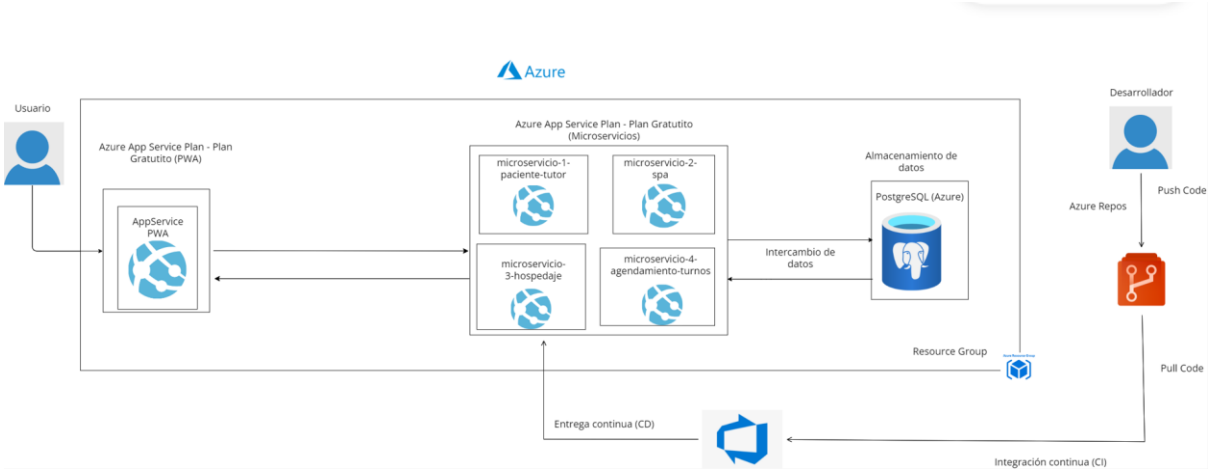


Nota. Diagrama entidad relación general, los encerrados en azul es lo que se abordara en este tema.

Diagrama de Despliegue. En la figura 20, se muestra el diagrama de despliegue aplicado para el desarrollo de los microservicios de los diferentes sprints, se tiene en cuenta la arquitectura aplicada y las diferentes tecnologías que intervinieron para el desarrollo de esta.

Figura 20

Diagrama de Despliegue de los microservicios

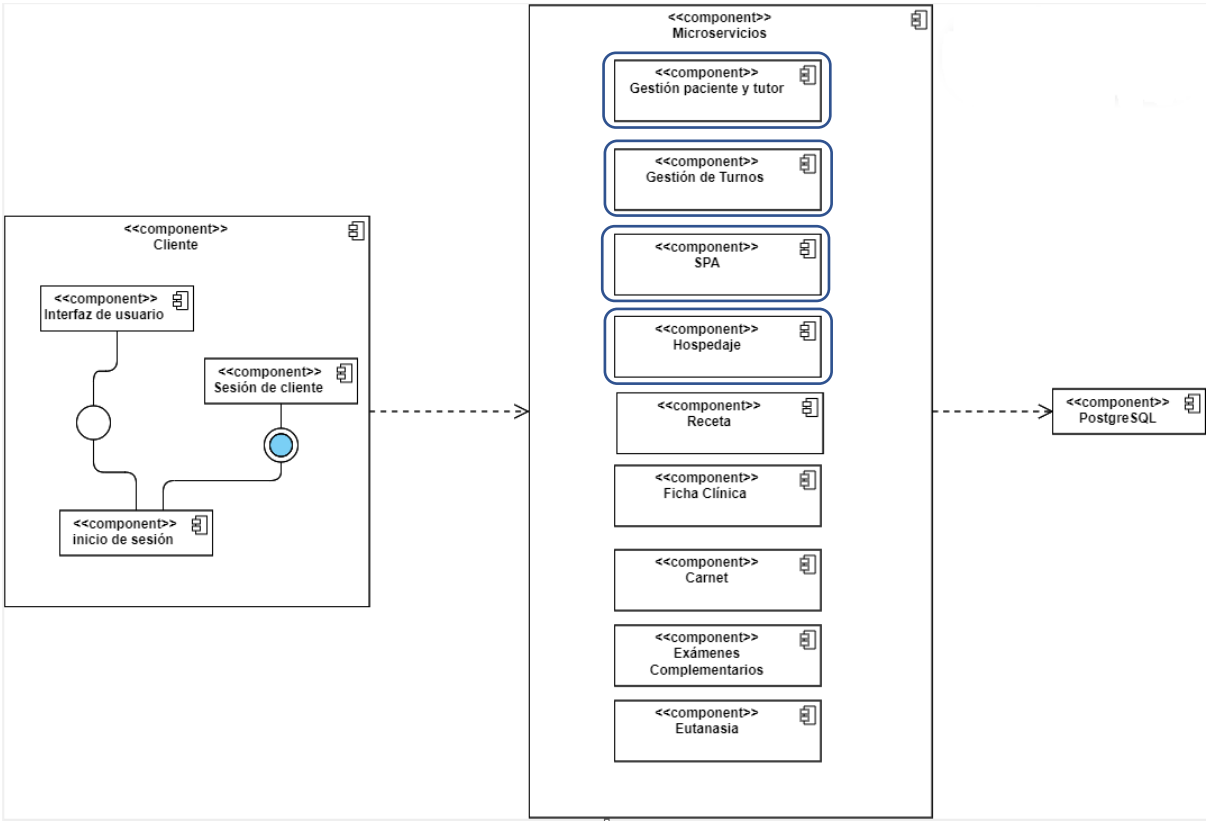


Nota: Diagrama de despliegue de los microservicios de este proyecto

Diagrama de componentes. En la figura 21, se muestra en el diagrama como serán divididos los componentes de los microservicios de los diferentes sprints y las dependencias que tienen entre ellos.

Figura 21

Diagrama de componentes de los microservicios



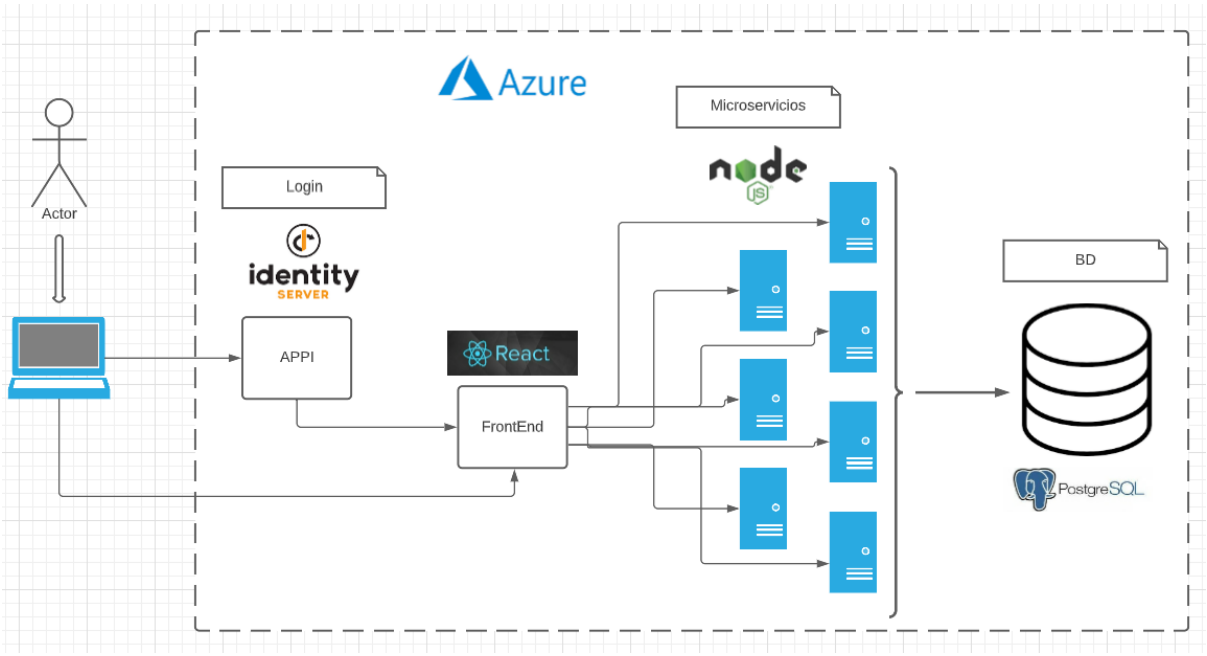
Nota. En el componente microservicio se muestra las historias de usuario a programar.

Arquitectura

En la figura 22, se observa la arquitectura empleada para la creación del sistema general y las herramientas utilizadas en su ejecución. Necesita de una base de datos creada en PostgreSQL, esta se conectará a los microservicios creados en NodeJs. Cuando los microservicios estén implementados en Azure, desde el front-end se consumirán los mismos. Para tener acceso al sistema primero se pasará por la validación de Identity Server, en donde se deben registrar e iniciar sesión, así el usuario podrá utilizar el SGCV. En esta tesina se abarcará totalmente el diseño y creación de los microservicios y también la edición de la base de datos.

Figura 22

Diagrama de arquitectura de los microservicios



Nota. Arquitectura del sistema general

Desarrollo del Sistema

En esta sección, se presenta de forma detallada el proceso realizado y las herramientas utilizadas para el desarrollo de cada funcionalidad acorde a las historias de usuario planificadas en el product backlog.

Tabla 8

Herramientas y actividades del desarrollo.

Actividad	Herramienta Descripción
Codificación de microservicios	Se utiliza el IDE de desarrollo Visual Studio Code en su versión 1.75. Se utiliza el lenguaje de programación NodeJS en su versión 18.
Control de errores en los microservicios con sus respectivas validaciones.	Se utiliza un conjunto de middlewares llamado Express Validator en su versión 6.14.0.
Control de flujo de trabajo	Se utiliza la herramienta GitFlow, la cual tendrá una rama develop de prepublicación y la rama main que es la principal y se sincroniza con las ramas personales
Base de datos	Se utiliza PGAdmin versión 4, para gestionar, en ella se utilizará PostgresSQL 4. Se utiliza la herramienta Liquibase, para la sincronización de los cambios en la base de datos.

Actividad	Herramienta Descripción
Pruebas de funcionalidad	Se utiliza la herramienta Postman en su versión gratis
Pruebas de rendimiento	Se utiliza la herramienta Locust en su versión 2.14.2
Gestión del ciclo de vida del proyecto y herramientas DevOps	Se utiliza la aplicación Azure DevOps, que permite gestionar repositorios, gestión de backlogs y herramientas de código.
Despliegue de Microservicios	Se utiliza la nube Azure
Autenticación	Se utiliza el Framework de Microsoft llamado Identity Server.

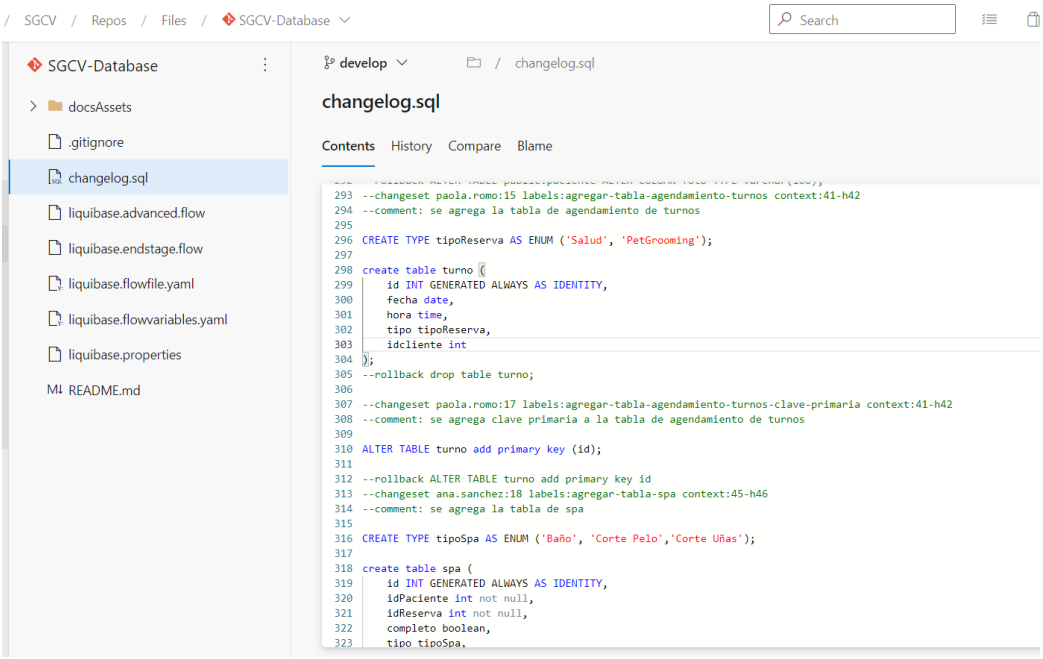
Nota. En esta tabla se presenta las herramientas que se han utilizado a lo largo del desarrollo.

Configuración de la Base de datos

En esta sección se podrá observar cómo se ha implementado la base de datos en los sprints. Como se puede observar en la figura 22, parte de la arquitectura de microservicios es la conexión a una base de datos, la misma se encarga de almacenar toda la información que se registra en los microservicios. Se utilizará el sistema de gestión de base de datos llamado PostgreSQL. Para la implementación de cada sprint, primero se crea las tablas con sus respectivos campos y relaciones, los mismos se encuentran en la figura 19 perteneciente al diagrama entidad relación. El repositorio SGCV Database perteneciente al proyecto general, se encuentra en Azure DevOps. Se clona el repositorio en las máquinas locales, se procede a crear las ramas según el número de características asignado y se publica las ramas, en cada rama se coloca los datos de la tabla que se requiere. Para subir los datos y sincronizar los cambios realizados se utiliza el comando Liquibase Update, adicional a esto se realiza en DevOps un pull request a la rama develop. Una vez realizado todos estos pasos, en la herramienta pgAdmin, se puede visualizar las nuevas tablas, nuevos campos o datos que se agregaran. En la figura 23 se observa el repositorio de la BD y en la figura 24 se observa las tablas de los microservicios asignados en esta tesis.

Figura 23

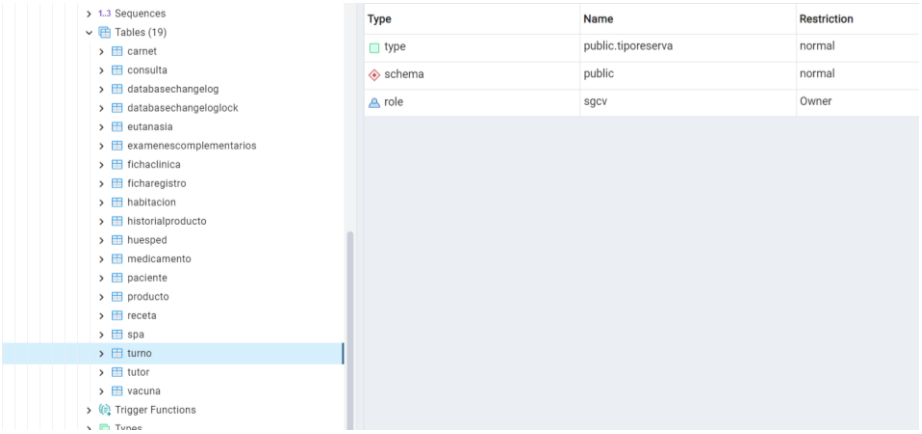
Repositorio de la base de datos



Nota. Se observa el código SQL perteneciente a las tablas de las historias de usuario de turno y Spa

Figura 24

Tablas de la BD en PostgreSQL



Nota. Se muestra el programa pgAdmin donde podemos trabajar de mejor manera con la BD y visualizar las tablas y sus respectivos campos

Sprint 1: Gestión de Paciente y Tutor

Para el desarrollo de este sprint se planifica la realización de las historias de usuario HU001 y HU002. En la tabla 9, se describe a detalle la ejecución de la gestión de Tutor. En la tabla 10, se describe la ejecución de la gestión Paciente.

Tabla 9

Historia de Usuario HU001 Gestión de tutor.

Gestión de Tutor		
Código: HU001		Estimación: 13 días
Como Veterinario, quiero registrar, actualizar, listar, eliminar un Tutor, para llevar un listado ordenado de los Tutores a administrar y relacionarlo con su respectiva mascota.		
Criterios de aceptación		
1.	Registrar Tutor	Se debe desplegar un formulario con los campos: nombre, cédula, dirección, teléfono. Al presionar en Guardar se deben validar que los campos contengan datos y si la información es correcta se visualiza un mensaje indicando que el Tutor se ha creado.
2.	Actualizar Tutor	Para la actualización del Tutor se despliega la información registrada. Al presionar en Actualizar se debe visualizar un mensaje indicando que el Tutor ha sido actualizado.
3.	Listar Tutor	Se debe solicitar el id o ci del Tutor que desea obtener la información. Al dar clic en Buscar se debe mostrar la información registrada del Tutor.
4.	Eliminar Tutor	Se debe solicitar el id o código del Tutor que desea eliminar. Si el Paciente existe se visualiza un mensaje que indica que el Tutor ha sido eliminado, además se deberán eliminar los Pacientes asociados a él.

Nota. Se muestra de forma detallada la historia de usuario para la gestión de Tutores.

Tabla 10

Historia de Usuario HU001 Gestión de Paciente.

Gestión de Paciente	
Código: HU002	Estimación: 13 días

Gestión de Paciente		
Como Veterinario, quiero registrar, actualizar, listar, eliminar un Paciente, para llevar un listado ordenado de los Pacientes a administrar y brindarle atención médica de calidad.		
Criterios de aceptación		
1.	Registrar Paciente	Se debe desplegar un formulario con los campos: nombre, especie, raza, sexo, color, fecha de nacimiento, Tutor y foto. Al presionar en Guardar se deben validar que todos los campos hayan sido llenados y si la información es correcta se visualiza un mensaje indicando que el Paciente se ha creado.
2.	Actualizar Paciente	Para la actualización del Paciente se despliega la información registrada. Al presionar en Actualizar se debe visualizar un mensaje indicando que el Tutor ha sido actualizado.
3.	Listar Paciente	Se debe solicitar el id o nombre del Paciente o el Tutor del paciente del que desea obtener la información. Al dar clic en Buscar se debe mostrar la información registrada del Paciente.
4.	Eliminar Paciente	Se debe solicitar el id o código del paciente que desea eliminar. Si el paciente existe se visualiza un mensaje que indica que el Paciente ha sido eliminado, además se deberán eliminar los archivos médicos asociados a él.

Nota. Se muestra de forma detallada la historia de usuario para la gestión de Pacientes.

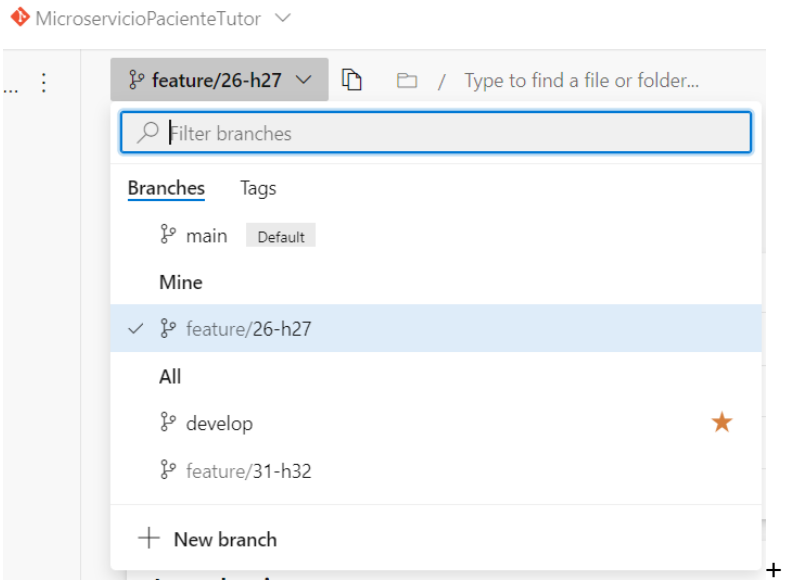
Configuración del entorno de desarrollo sprint 1. Para la codificación del microservicio Paciente-Tutor se realizó la creación del repositorio en Azure DevOps. El mismo se clona en el computador y se lo abre en Visual Studio Code. En gitflow, cada miembro del equipo crea su rama de acuerdo con la historia de usuario asignada, a la misma se la pública y se sube a Azure DevOps. Para el desarrollo de la historia de usuario HU001, correspondiente a Tutor, se crea la rama 31-h32 en la cual se alojará todo el código fuente de esta. En la rama 26-h27 se alojará el código de la historia de usuario HU002, correspondiente a Paciente.

Las ramas, tendrán en común archivos como rutas, controladores, conexión a la base de datos, middlewares y validaciones. Para realizar la integración de las ramas de cada una se

realiza un commit, este se visualiza en Azure DevOps, aquí se realiza un pull request desde la rama de la historia de usuario, hasta la rama develop. En Visual Studio Code, sincronizamos los cambios en la rama develop y se podrá observar los cambios implementados tanto en la HU001 y la HU002, aunque no sea la rama en la que se está codificando. Para la puesta en producción se tendrá la rama main. En la figura 25, se observa el repositorio PacienteTutor con sus respectivas ramas.

Figura 25

Ramas del Microservicio Paciente Tutor en repositorio Git Flow de Azure DevOps.



Nota. Se muestra las ramas que se crearon para el desarrollo del Microservicio.

Sprint 2: Gestión de Turno, Spa, Huésped, Habitación, Ficha de registro de hospedaje.

Para el desarrollo de este sprint se tiene planificada la realización de las historias de usuario HU009, HU010, HU011, HU012, HU013 que se detallaran en la tabla 11, 12,13,14,15 respectivamente.

Tabla 11

Historia de Usuario HU009 Gestión de Turno.

Gestión de Turno	
Código: HU009	Estimación: 13 días

Gestión de Turno		
Como Veterinario, quiero crear, listar, eliminar un turno, para brindar un servicio ordenado y organizado a los Pacientes.		
Criterios de aceptación		
1.	Crear Turno	Se debe desplegar un formulario con los campos: fecha, hora, tipo, cliente. Al presionar en Guardar se deben validar que todos los campos hayan sido llenados y si la información es correcta se visualiza un mensaje indicando que el turno se ha creado.
2.	Listar Turno	Se debe solicitar el id o servicio por el cual se desea listar el turno. Al dar clic en Buscar se debe mostrar la información registrada del turno.
3.	Eliminar Paciente	Se debe solicitar el id o código del turno que se desea eliminar. Si el turno existe se visualiza un mensaje que indica que el turno ha sido eliminado, además se deberán eliminar los archivos asociados a él.

Nota. Se muestra de forma detallada la historia de usuario para la gestión de turnos.

Tabla 12

Historia de Usuario HU010 Gestión de Spa.

Gestión de Spa		
Código: HU010		Estimación: 13 días
Como Veterinario, quiero crear, actualizar, listar, eliminar un Spa, para llevar un listado ordenado de los **registros asignados para brindar servicios de Spa.**		
Criterios de aceptación		
1.	crear Spa	Se debe desplegar un formulario con los campos: Paciente, reserva, completo, tipo, hora de ingreso, hora de salida. Al presionar en Guardar se deben validar que todos los campos hayan sido llenados y si la información es correcta se visualiza un mensaje indicando que el Paciente se ha creado.
2.	Actualizar Paciente	Para la actualización del Spa se despliega la información registrada. Al presionar en Actualizar se debe visualizar un mensaje indicando que el Spa ha sido actualizado.
3.	Listar Paciente	Se debe solicitar el id, reserva, o Paciente del Spa del que desea obtener la información. Al dar clic en Buscar se debe mostrar la información registrada del Spa.

	Gestión de Spa	
4.	Eliminar Paciente	Se debe solicitar el id o código del Spa que desea eliminar. Si el Spa existe se visualiza un mensaje que indica que el Spa ha sido eliminado.

Nota. Se muestra de forma detallada la historia de usuario para la gestión de Spa.

Tabla 13

Historia de Usuario HU011 Gestión de Huésped.

	Gestión de Huésped	
Código: HU011		Estimación: 13 días
Como Veterinario, quiero crear, listar, eliminar un huésped, para llevar un listado ordenado de los huéspedes que se encuentran en el hospedaje.		
Criterios de aceptación		
1.	Crear huésped	Se debe desplegar un formulario con los campos: cliente, reserva, Paciente. Al presionar en Guardar se deben validar que todos los campos hayan sido llenados y si la información es correcta se visualiza un mensaje indicando que el huésped se ha creado.
2.	Listar huésped	Al dar clic en Buscar se debe mostrar la información registrada de los huéspedes.
3.	Eliminar huésped	Se debe solicitar el id o código del huésped que desea eliminar. Si el huésped existe se visualiza un mensaje que indica que el huésped ha sido eliminado, además se deberán eliminar los archivos asociados a él.

Nota. Se muestra de forma detallada la historia de usuario para la gestión de huésped.

Tabla 14

Historia de Usuario HU012 Gestión de Habitaciones.

	Gestión de Habitaciones	
Código: HU012		Estimación: 13 días
Como Veterinario, quiero crear, listar, actualizar, eliminar las habitaciones del Hospedaje para conocer que habitaciones se encuentran disponibles en el hospedaje.		
Criterios de aceptación		

Gestión de Habitaciones		
1.	Crear habitación	Se debe desplegar un formulario con los campos: disponibilidad, número de habitación y descripción. Al presionar en Guardar se deben validar que todos los campos hayan sido llenados y si la información es correcta se visualiza un mensaje indicando que la habitación se ha creado.
2.	Actualizar habitación	Para la actualización de la habitación se despliega la información registrada. Al presionar en Actualizar se debe visualizar un mensaje indicando que la habitación ha sido actualizada.
3.	Listar habitación	Al dar clic en Buscar se debe mostrar la información registrada de la habitación.
4.	Eliminar habitación	Se debe solicitar el id o código de la habitación que se desea eliminar. Si el Paciente existe se visualiza un mensaje que indica que la habitación ha sido eliminada, además se deberán eliminar los archivos médicos asociados a ella.

Nota. Se muestra de forma detallada la historia de usuario para la creación de habitaciones.

Tabla 15

Historia de Usuario HU013 Gestión de Ficha de Registro de Spa.

Gestión de Ficha de Registro de Spa		
Código: HU013		Estimación: 13 días
Como Veterinario, quiero crear, actualizar, listar, eliminar una ficha de registro de Spa, para llevar un listado ordenado de las fichas de registro de Spa.		
Criterios de aceptación		
1.	Crear ficha de registro de Spa	Se debe desplegar un formulario con los campos: día de entrada, día de salida, observaciones, Paciente, habitación. Al presionar en Guardar se deben validar que todos los campos hayan sido llenados y si la información es correcta se visualiza un mensaje indicando que la ficha de Spa se ha creado.
2.	Actualizar Paciente	Para la actualización de la ficha de Spa se despliega la información registrada. Al presionar en Actualizar se debe visualizar un mensaje indicando que la ficha de Spa se ha sido actualizado.

Gestión de Ficha de Registro de Spa		
3.	Listar Paciente	Al dar clic en Buscar se debe mostrar la información registrada de la ficha de registro de Spa.
4.	Eliminar Paciente	Se debe solicitar el id o código de la ficha de Spa que desea eliminar. Si la ficha de Spa existe se visualiza un mensaje que indica que la ficha de Spa ha sido eliminada, además se deberán eliminar los archivos médicos asociados a ella.

Nota. Se muestra de forma detallada la historia de usuario para la gestión de Pacientes.

Configuración de entorno de desarrollo sprint 2. Se realiza el mismo proceso detallado en el apartado **Configuración del Entorno de Desarrollo Sprint 1**. En la tabla 16, asignan las ramas de desarrollo para este sprint.

Tabla 16

Ramas Aplicadas para el desarrollo de Sprint 2.

N°	Historia de Usuario	Rama Asignada	Nombre de Microservicio
1	HU009	41-h42	MicroservicioAgendamientoTurno
2	HU010	45-h46	MicroservicioGestionSpa
3	HU011	50-h51	Microservicio Hospedaje
4	HU012	50-h51	Microservicio Hospedaje
5	HU013	50-h51	Microservicio Hospedaje

Nota. En esta tabla consta todas las ramas con su respectivo microservicio e historia de usuario

Sprint 3 Autenticación del sistema & Seguridades

Tabla 17

Historia de Usuario HU015 Autenticación del Sistema.

Autenticación del Sistema	
Código: HU002	Estimación: 13 días
Como Veterinario, quiero mantener un control de roles al ingresar al sistema para tener un registro ordenado de los médicos que ingresan al sistema.	
Criterios de aceptación	

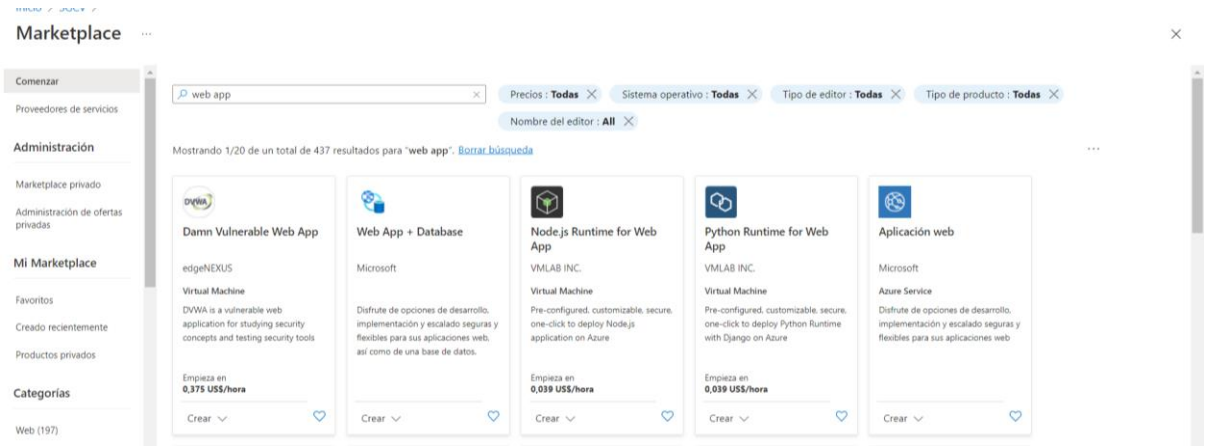
Autenticación del Sistema		
1.	Control de roles	El sistema solicita un usuario y contraseña registrados en la base de datos, en caso de no tener creada una cuenta se permite crear una nueva cuenta al veterinario el cual tendrá acceso al sistema según sea el rol que desempeña en Clínica.

Nota. Se muestra de forma detallada de la autenticación del sistema.

Despliegue de los Microservicios. En esta sección se abordará el despliegue de los microservicios de PacienteTutor, GestiónSpa, AgendamientoTurnos y Hospedaje. Como primer paso, se debe preparar el entorno de desarrollo, es decir en el archivo index.js, que es el que arranca la aplicación, configuramos por el puerto **app. listen (process.env. PORT || 8181)**. También se debe configurar el archivo package. json, el mismo se crea automáticamente al instalar las dependencias de NodeJs, se agrega la siguiente línea de código, “**start": "node ./src/index.js”,** las ramas mencionadas anteriormente pertenecientes a los Scripts uno y dos, deben estar sincronizadas a la rama develop. Cuando se empiece el proceso de despliegue, la rama develop deberá subir sus cambios a la rama main que es la rama de producción, para esto en la herramienta Azure DevOps, se crea un pull request desde la rama develop hacia la rama main. Una vez realizado este proceso, en Azure, se selecciona el grupo de recursos SGCV, el mismo abarcara todos los microservicios que necesita para su funcionamiento. Se da clic en crear y se elige crear Aplicación web, como se puede ver en la figura 26.

Figura 26

Creación de la App Service

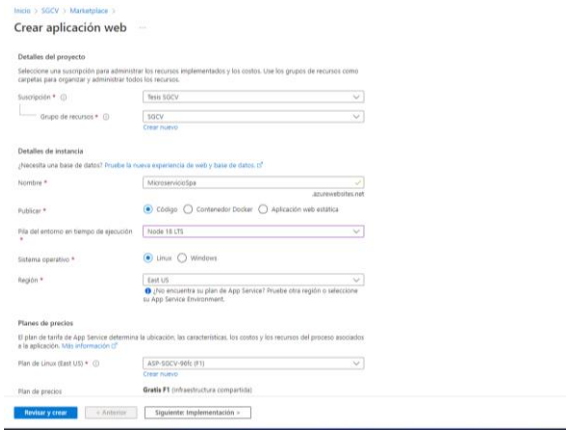


Nota. Siempre que se quiera realizar el despliegue de un microservicio se debe seleccionar aplicación web.

Cuando se crea una aplicación web, se completan los datos que se visualizan en la figura 27, seleccionando el lenguaje de programación de los microservicios, la versión, se le asigna un nombre y se escoge el plan de pago, en este caso se escoge la infraestructura F1, que pertenece a los servicios gratuitos que entrega Azure. Se da clic en revisar y crear y esta completado el despliegue.

Figura 27

Configuración del App Service



Nota. Se debe esperar unos minutos hasta que la URL esté listo para utilizar

Una vez completado el despliegue se obtuvo una URL del microservicio en Azure, en la tabla 18, se observa las URL generadas de todos los microservicios.

Tabla 18

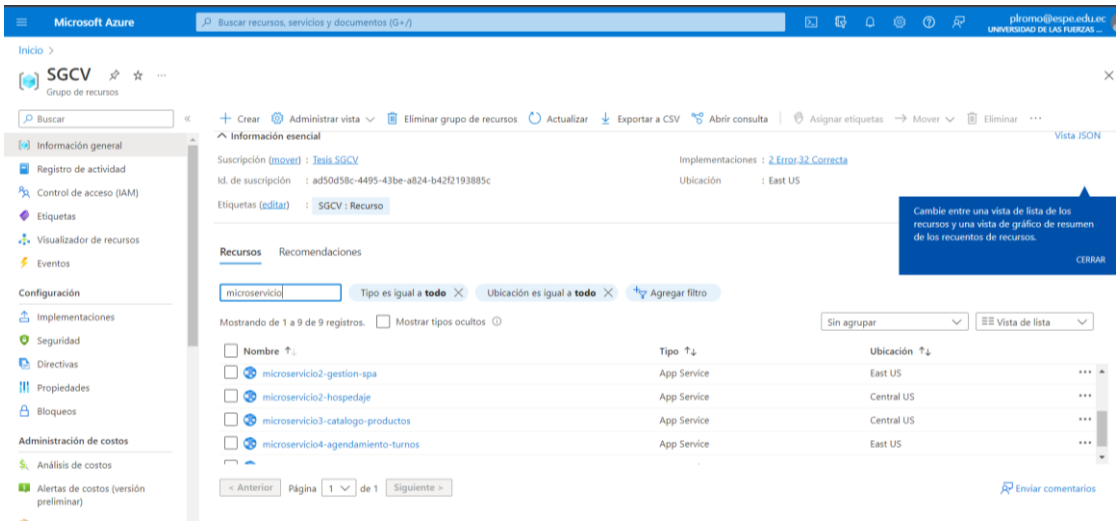
URL de los microservicios publicados en Azure.

Microservicio	**URL**
PacienteTutor	https://microservicio1-paciente-tutor.azurewebsites.net/
GestiónSpa	https://microservicio2-gestion-spa.azurewebsites.net/
AgendamientoTurnos	https://microservicio4-agendamiento-turnos.azurewebsites.net/
Hospedaje	https://microservicio2-hospedaje.azurewebsites.net/

Nota. Estas URL se generan una vez implementados los microservicios en Azure

Figura 28

Microservicios publicados en Azure



Nota. Se visualiza los microservicios publicados ya en Azure.

Validaciones del sistema

Validación de Sprint 1

Para la validación del sprint 1, se genera pruebas de funcionalidad y de rendimiento. Las herramientas por utilizar son las ya listadas en la tabla 8, descrita al inicio de la sección.

Pruebas de funcionalidad en Postman. Para realizar las respectivas pruebas en el microservicio PacienteTutor, lo primero que se realiza es crear una colección en Postman para poder almacenar ordenadamente las pruebas que se vayan ejecutando. Para esto obtenemos de la tabla 18, la URL perteneciente a este microservicio. Las pruebas se empiezan con Tutor, este es independiente, no tiene una clave foránea en la base de datos. En la figura 29, se puede observar las rutas pertenecientes a la historia de usuario Tutor, de las mismas se realizan las pruebas de funcionalidad, esperando se den con éxito las respuestas requeridas.

Figura 29

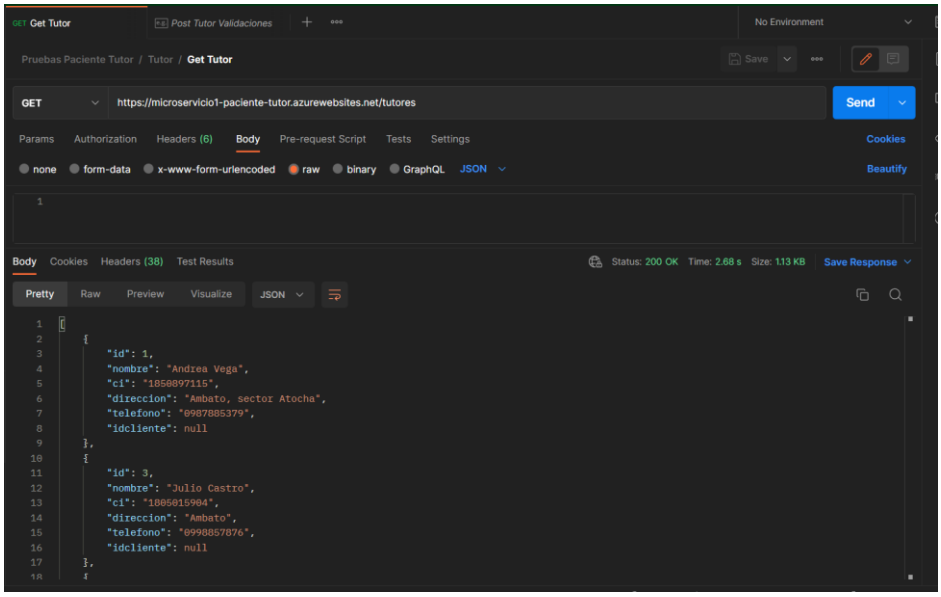
Rutas Tutor

Nota. Se visualiza desde Visual Studio Code las rutas codificadas de Tutor.

En este párrafo se describe como se realizan las pruebas con el método GET. En Postman ingresamos la URL del microservicio, acompañado de un / en el que se enviara el parámetro que requiera cada petición. En las figuras listadas a continuación, se visualizará de mejor manera las pruebas con GET ya hechas en Postman, la herramienta permite guardar los ejemplos realizados.

Figura 30

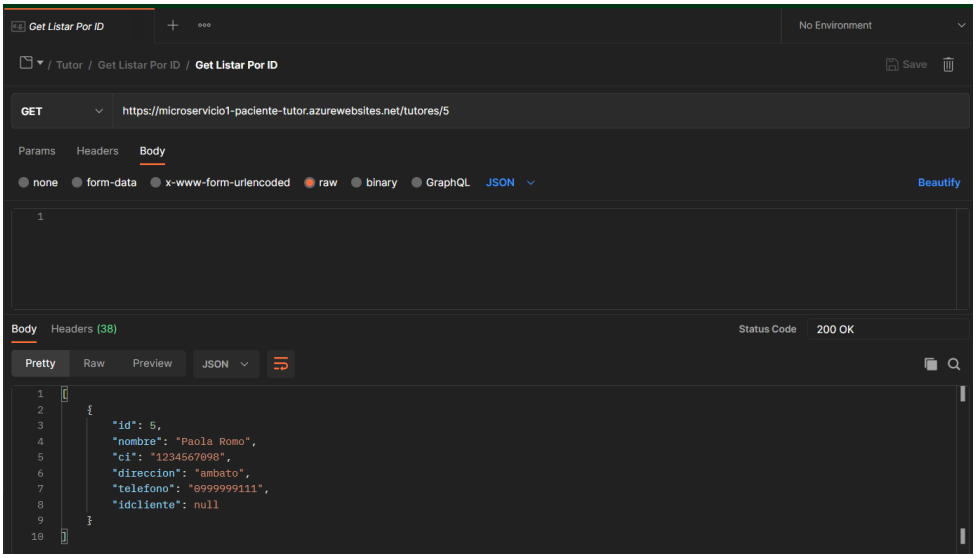
GET Tutores



Nota. Datos consultados de la tabla Tutor

Figura 31

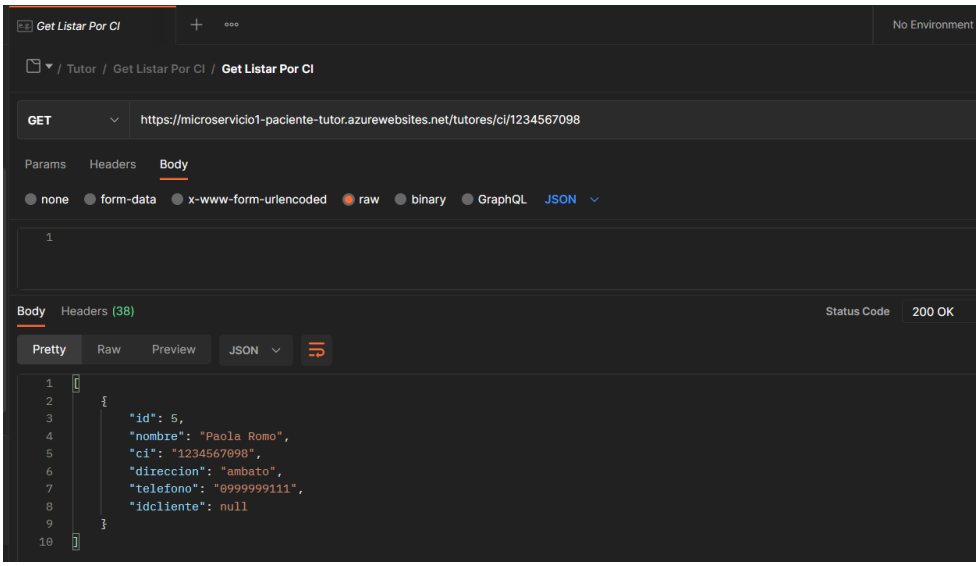
GET listar tutores por ID



Nota. Datos consultados de la tabla Tutores con la consulta realizada por el id de cada uno de ellos

Figura 32

GET listar tutores por CI

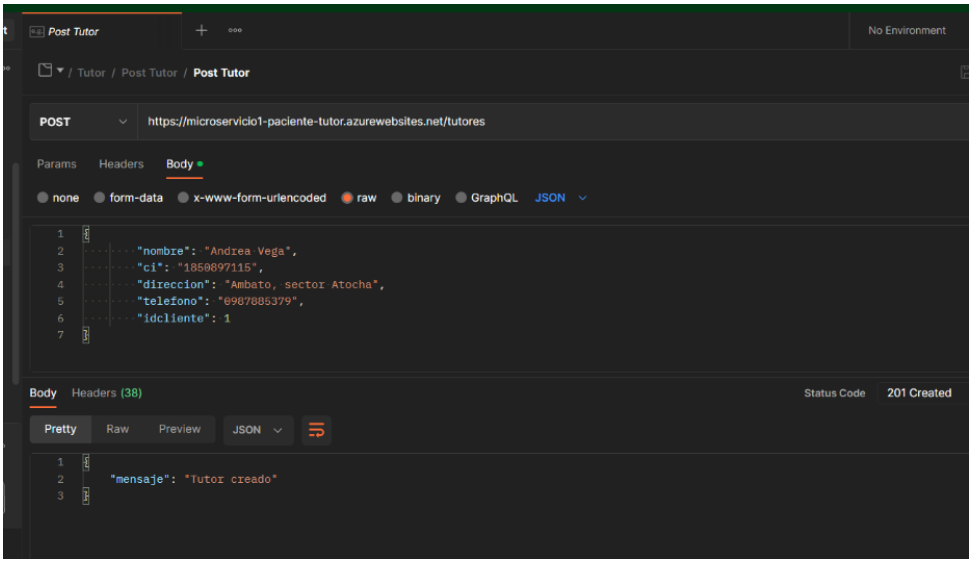


Nota. Datos consultados de la tabla Tutores con la consulta realizada por la ci de cada uno de ellos

En este párrafo se explica cómo se realizaron las pruebas con el método POST. Se envía los datos requeridos en el controlador, en el caso de Tutor son: nombre, ci, dirección, teléfono. Estos se envían desde Postman, en formato JSON. En la figura 33, se observa como los datos se guardan correctamente, nos aparece un mensaje de confirmación. En la figura 34, se puede observar las validaciones, en caso de enviar datos erróneos Postman se encarga de enviar un mensaje indicando en que campo se encuentra la información incorrecta. Si estas falencias no se corrigen, la información no se guardará en la base de datos.

Figura 33

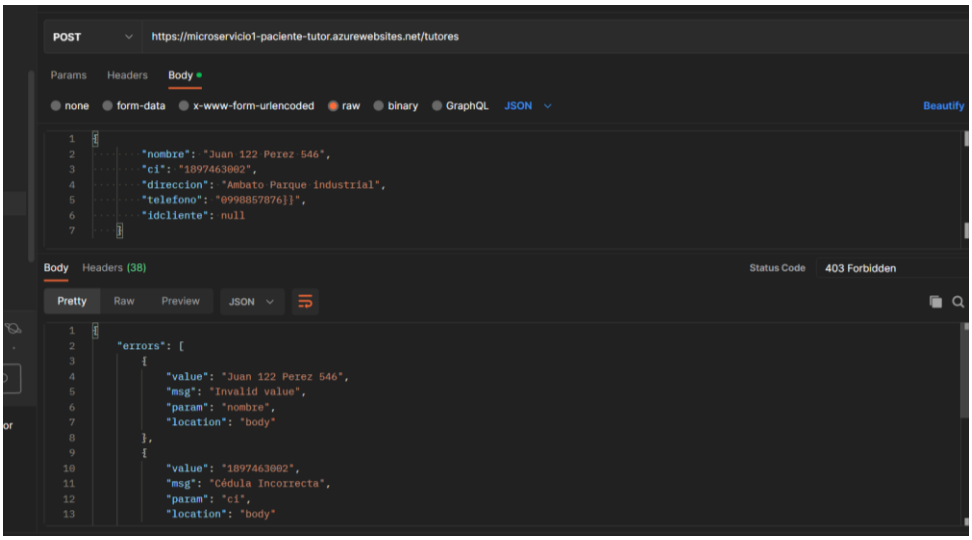
POST Tutores



Nota. Se visualiza la creación correcta de los Tutores nuevos

Figura 34

Post tutores validación



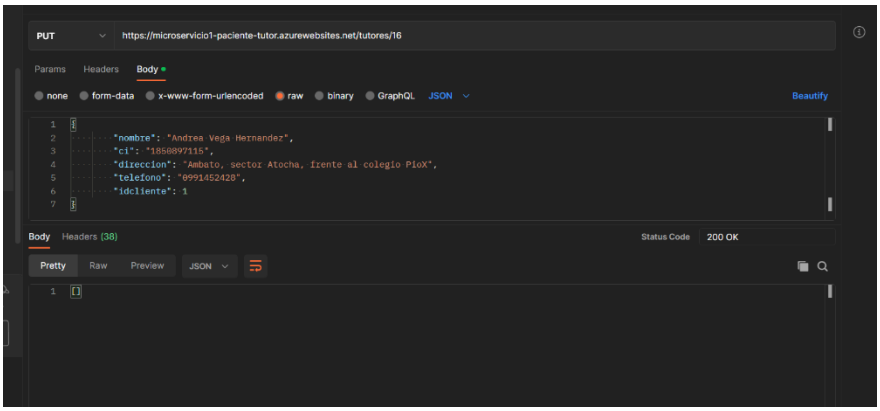
Nota. Se observa los resultados arrojados de error y en qué datos se comete el mismo, no se guarda en la BD.

Para PUT y DELETE es un proceso similar. En la figura 35, se puede observar que, para actualizar los datos, primero requiere del ID generado en la base de datos, se debe enviar el ID y la respectiva modificación que se necesita respetando las validaciones antes

ya mencionadas. En la figura 36, se observa cómo se elimina la información de la base de datos, para esto enviamos el ID en la ruta y se enviara un mensaje de confirmación.

Figura 35

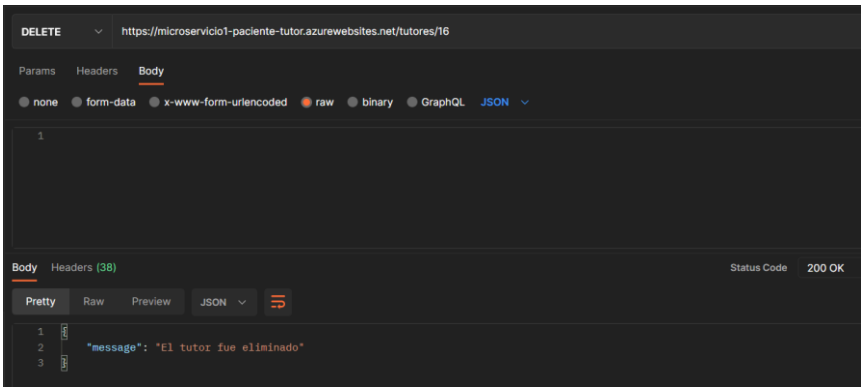
Put tutores



Nota. Se edita con éxito el campo solicitado.

Figura 36

Delete Tutores.



Nota. Se observa como el Tutor con el id enviado se elimina exitosamente.

Para realizar las pruebas de Paciente, se realiza el mismo proceso que se ha detallado anteriormente, lo único que cambiará será los métodos de peticiones HTTP, en la figura 37, están todas las rutas requeridas para la funcionalidad de la historia de usuario Paciente. La URL será la misma que ocupamos para Tutor. En el siguiente enlace, se da

acceso a un repositorio público en donde se puede observar las pruebas de Paciente: https://www.postman.com/warped-water-703504/workspace/pruebas-funcionalidad-ana-paola/collection/24825634-8977ab21-724f-48b2-8a04-880de1534fc0?action=share&creator=24825634

Figura 37

Rutas paciente

Nota. Rutas que se utilizaran para las pruebas de funcionalidad de la historia de usuario Pacientes.

Pruebas de rendimiento en Locust. Locust para su funcionamiento utiliza Python. Una vez instalada la aplicación que funcionara en la web, se crea un archivo locustfile.py, es este archivo se escribirá las rutas a evaluar, las rutas ocupadas son las mismas establecidas anteriormente, que se usaron en Postman. En la figura 38, se puede ver un prototipo de pruebas GET y POST realizadas al Microservicio PacienteTutor, se utiliza las rutas de Tutor.

Figura 38

Archivo locustfile.py para tutores

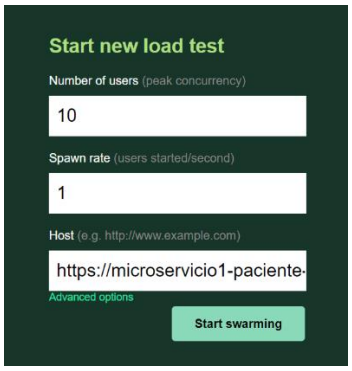
```python
from locust import HttpUser, task

class Tutor(HttpUser):
    @task
    def listar_tutores(self):
        self.client.get("/tutores")
    @task
    def listar_tutores_porID(self):
        self.client.get("/tutores/5")
    @task
    def crear_tutores(self):
        self.client.post("/tutores",json={

        "nombre": "Andrea Jimenez",
        "ci": "0501724306",
        "direccion": "Ambato, Huchi Chico",
        "telefono": "0991452428",
    })
```

Nota. Se observa cómo se envían las rutas pertenecientes a Tutores y un método Post para la carga masiva de datos.

En la terminal se corre el comando locust y envía un enlace el cual se abre en el navegador que es http://localhost:8089, se abre una interfaz, la misma pide tres campos que son número de usuarios, tiempo de ejecución y la URL que se quiere evaluar. En la figura 39, se puede observar la interfaz mencionada, ya lista para empezar las pruebas de rendimiento. Al dar clic en Star swarming empieza la ejecución

Figura 39

Interfaz de inicio de Locust



Nota. Cuando se corra esa ruta anteriormente mencionada se despliega esta interfaz

Locust empieza a realizar las pruebas a la URL ingresada con las respectivas rutas especificadas anteriormente, en la figura 40, se observa los resultados que la aplicación

arroja. Estos resultados se los conoce como métricas, Locust trabaja con métricas como tiempo de respuesta máximos y mínimos, también tiene tiempo de respuesta por segundo, y el tiempo medio de respuesta. Para el método POST, Locust empieza a cargar datos masivamente y llenarlos en la base de datos, se observa cómo responde el microservicio ante este evento. Se recomienda después de finalizar esta prueba aplicar un Truncate a la base de datos, la misma se reiniciará.

Figura 40

Resultados de Locust

Request Statistics

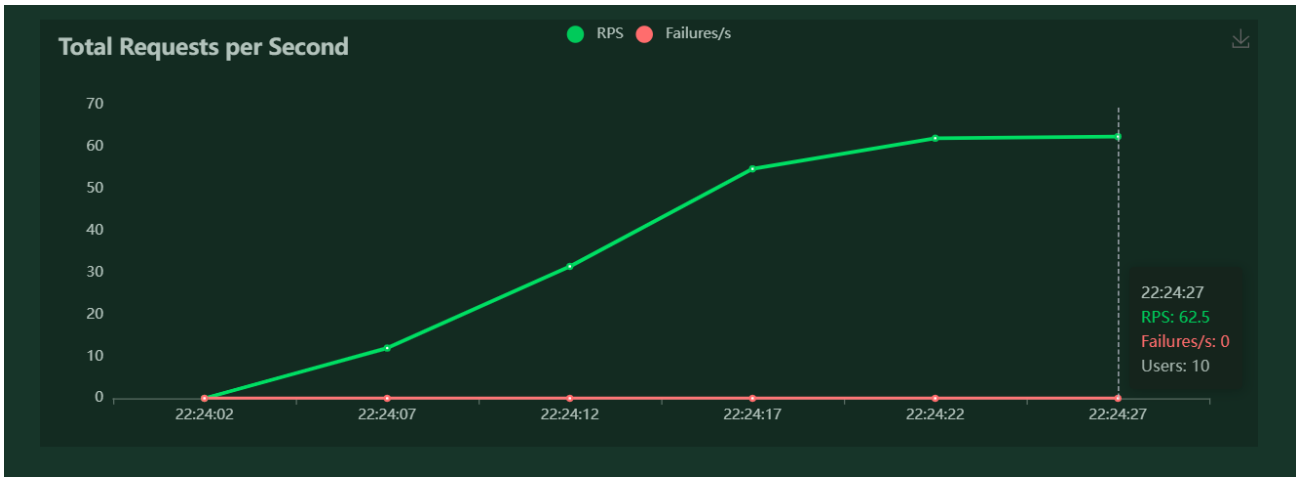
Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
POST	/tutores	475	0	170	142	1314	26	15.9	0.0
GET	/tutores/5	513	0	159	135	539	112	17.2	0.0
GET	/tutores	518	0	172	140	740	32071	17.4	0.0
	Aggregated	**1506**	**0**	**167**	**135**	**1314**	**11077**	**50.5**	**0.0**

Nota. Resultados de los get y post de Tutores

Locust también arroja su resultado en gráficos, entre estos los de la figura 41, que representa al tiempo de respuesta por segundo, la figura 42, que representa el número de usuarios por tiempo de respuesta. Si las gráficas siguieran ascendiendo al infinito y no se estabilizarán, eso quiere decir que existe un cuello de botella en la aplicación, aquí se observa que aplicación se estabiliza y funciona correctamente.

Figura 41

Total de respuestas por segundo



Nota. En el eje de las x está el tiempo de respuesta y en el eje de las y el número de usuarios.

Figura 42

Respuestas por el número de usuarios

Nota. En el eje de las x se encuentra el tiempo de respuesta, en el eje de las y el número de usuarios. No existen cuellos de botella

Lista de Chequeo de Sprint 1. Para realizar las pruebas al sistema se realiza la técnica de listas de chequeo, las cuales tienen como objetivo verificar el cumplimiento de las funcionalidades propuestas en las historias de usuario de acuerdo con los criterios de

aceptación planteados. En la tabla 19 y 20 se define los criterios de aceptación de las historias de usuario de este sprint. Estas se basan en las pruebas ya realizadas.

Tabla 19

Lista de chequeo de la historia de usuario HU001.

Gestión de Tutor		
Código: HU001	Estimación: 13 días	
Como Usuario**,** quiero registrar, actualizar, listar, eliminar un Tutor, **para** llevar un listado ordenado de los Tutores a administrar y relacionarlo con su respectiva mascota**.**		
Prueba	Detalles	Estado
Validar datos en el formulario para registrar / actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Registrar Tutor	El sistema registra de forma exitosa el Tutor y muestra un mensaje de que el Tutor se ha creado. En caso de existir un problema, se muestra un error	✓
Actualizar Tutor	El sistema actualiza de forma exitosa el Tutor, en caso de existir un problema se muestra el respectivo mensaje.	✓
Listar Tutor	Al enviar el id del Tutor que desea mostrar la información se muestra correctamente.	✓
	Al enviar la cédula (ci) del Tutor que se desea mostrar la información se muestra correctamente.	✓
Eliminar Tutor	El sistema elimina correctamente el Tutor del id que se ha enviado, en caso de existir errores muestra un mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU001.

Tabla 20

Lista de chequeo de la historia de usuario HU002.

Gestión de Paciente	
Código: HU002	Estimación: 13 días

Gestión de Paciente		
Como Usuario**,** quiero registrar, actualizar, listar, eliminar un Paciente**, para** llevar un listado ordenado de los Pacientes a administrar y brindarle atención médica de calidad		
Prueba	Detalles	Estado
Validar datos en el formulario para registrar / actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Registrar Paciente	El sistema registra de forma exitosa el Paciente y muestra un mensaje de que el Paciente se ha creado. En caso de existir un problema, se muestra un error	✓
Actualizar Paciente	El sistema actualiza de forma exitosa el Paciente, en caso de existir un problema se muestra el respectivo mensaje.	✓
Listar Tutor	Al enviar el id del Paciente que desea mostrar la información se muestra correctamente.	✓
	Al enviar la cédula (ci) del Tutor que se desea mostrar la información se muestra correctamente.	✓
Eliminar Paciente	El sistema elimina correctamente el Tutor del id que se ha enviado, en caso de existir errores muestra un mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU001

Validación del Sprint 2

Para la validación del sprint 1, se genera pruebas de funcionalidad y de rendimiento. Las herramientas por utilizar son las ya listadas en la tabla 8, descrita al inicio de la sección.

Pruebas de funcionalidad en Postman. El proceso para las pruebas de funcionalidad será el mismo empleado en la sección anterior perteneciente al sprint uno. Las URL por utilizar son las detalladas en la tabla 18. Las rutas variaran dependiendo la historia de usuario que se requiera. Los métodos de petición HTTP serán GET, POST, PUT, DELETE. En la tabla 21, se listará los repositorios creados por Postman, el cual permitirá

acceder a las colecciones realizadas en cada historia de usuario correspondiente a este sprint.

Tabla 21

Repositorios de pruebas de Postman.

Historia de Usuario	Enlace del repositorio en Postman
HU002 – Gestión Spa	https://www.postman.com/warped-water-703504/workspace/pruebas-funcionalidad-ana-paola/collection/24825634-18873943-c810-4826-b879-fb37a02f35d4?action=share&creator=24825634
HU009- Gestión Turno	https://www.postman.com/warped-water-703504/workspace/pruebas-funcionalidad-ana-paola/collection/24825634-2bd587d1-72ed-4796-b637-2d8b0d78f7a8?action=share&creator=24825634
HU011- Gestión Huésped **HU012 – Gestión Habitación** **HU013- FichaRegistroHospedaje**	https://www.postman.com/warped-water-703504/workspace/pruebas-funcionalidad-ana-paola/collection/24825634-cc2e8f59-6054-405d-85d5-94ccdbe4dab3?action=share&creator=24825634

Nota. Se encuentra las URL públicas para acceso a las pruebas de funcionalidad

Pruebas de rendimiento en Locust. En proceso para las pruebas de rendimiento será el mismo empleado en la sección anterior perteneciente al sprint uno. Las URL por utilizar serán los que se encuentran en la tabla 18. Para la ejecución de Locust se necesitará seguir empleando los mismos métodos descritos anteriormente, se necesita un archivo locustfile.py y correr en la terminal el comando Locust. Para el número de usuarios y tiempo de respuesta, se seguirá utilizando 10 usuarios y 1 segundo, esto se realizará en todas las historias de usuario. Los métodos por evaluar son GET y POST este último cargara información masiva a la base de datos, para evaluar cómo responde ante una carga tan grande de datos. En el Anexo 5, se encontrará una carpeta con los reportes generados por la herramienta, así se puede visualizar de manera más detallada los resultados.

Como resultados generales se concluyó que los microservicios no tienen cuello de botella es decir se estabilizan correctamente en el tiempo indicado.

Al concluir las pruebas se recomienda aplicar un truncate a la base de datos, la misma procede a reiniciarse.

Lista de Chequeo de Sprint 2

Tabla 22

Lista de chequeo de la historia de usuario HU009.

Gestión de Turno		
Código: HU009	Estimación: 13 días	
Como Usuario **quiero** crear, listar, eliminar un turno, **para** brindar una atención ordenada de los diferentes servicios de la veterinaria.		
Prueba	Detalles	Estado
Validar datos en el formulario para crear / actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Crear turno	El sistema registra de forma exitosa el turno y muestra un mensaje de que el turno se ha creado. En caso de existir un problema, se muestra un error	✓
Listar turno	Al enviar el id del Turno que desea mostrar la información se muestra correctamente.	✓
Eliminar turno	El sistema elimina correctamente el Turno del id que se ha enviado, en caso de existir errores muestra un mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU009.

Tabla 23

Lista de chequeo de la historia de usuario HU010.

Gestión de Spa	
Código: HU002	Estimación: 13 días

Gestión de Spa		
Como Usuario **quiero** crear, actualizar, listar, eliminar un Spa**, para** llevar un listado ordenado de los turnos asignados para brindar el servicio de Spa.		
Prueba	Detalles	Estado
Validar datos en el formulario para Crear/ Actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Crear Spa	El sistema registra de forma exitosa el Paciente y muestra un mensaje de que el Spa se ha creado. En caso de existir un problema, se muestra un error	✓
Actualizar Spa	El sistema actualiza de forma exitosa el Paciente, en caso de existir un problema se muestra el respectivo mensaje.	✓
Listar Spa	Al enviar el id del Spa que desea mostrar la información se muestra correctamente.	✓
	Al enviar la reserva del Spa que se desea mostrar la información se muestra correctamente.	✓
	Al enviar el Paciente del Spa que se desea mostrar la información se muestra correctamente	
Eliminar Spa	El sistema elimina correctamente el Spa del id que se ha enviado, en caso de existir errores muestra un mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU010

Tabla 24

Lista de chequeo de la historia de usuario HU011.

Gestión de huésped		
Código: HU011	Estimación: 13 días	
Como Usuario **quiero**, crear, listar, eliminar un huésped**, para** llevar un listado ordenado de los huéspedes que se encuentran en el hospedaje.		
Prueba	Detalles	Estado
Validar datos en el formulario para Crear / Actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓

Gestión de huésped		
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Crear huésped	El sistema registra de forma exitosa el huésped y muestra un mensaje de que el huésped se ha creado. En caso de existir un problema, se muestra un error	✓
Listar huésped	Se muestra la información de los huéspedes que se encuentran registrados en el hospedaje.	✓
Eliminar huésped	El sistema elimina correctamente el huésped del id que se ha enviado, en caso de existir errores muestra un mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU011

Tabla 25

Lista de chequeo de la historia de usuario HU012.

Gestión de Habitaciones		
Código: HU012	Estimación: 13 días	
Como Usuario **quiero** crear, listar, actualizar, eliminar las habitaciones del Hospedaje **para** conocer que habitaciones se encuentran disponibles en el hospedaje		
Prueba	Detalles	Estado
Validar datos en el formulario para Crear / Actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Crear habitaciones	El sistema registra de forma exitosa la habitación y muestra un mensaje de que la habitación se ha creado. En caso de existir un problema, se muestra un error	✓
Actualizar habitaciones	El sistema actualiza de forma exitosa la habitación, en caso de existir un problema se muestra el respectivo mensaje.	✓
Listar habitaciones	Se muestra la información de las habitaciones registradas.	✓
Eliminar habitaciones	El sistema elimina correctamente la habitación del id que se ha enviado, en caso de existir errores muestra un mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU012

Tabla 26

Lista de chequeo de la historia de usuario HU013.

Gestión de Ficha de Registro de Hospedaje		
Código: HU013	Estimación: 13 días	
Como Usuario**, quiero** crear, actualizar, listar, eliminar una ficha de registro de Spa, **para** llevar un listado ordenado de las fichas de registro de Spa.		
Prueba	Detalles	Estado
Validar datos en el formulario para Crear / Actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Crear ficha de registro de hospedaje	El sistema registra de forma exitosa la ficha de registro de Hospedaje y muestra un mensaje de que la ficha de registro de Spa se ha creado. En caso de existir un problema, se muestra un error	✓
Actualizar ficha de registro de hospedaje	El sistema actualiza de forma exitosa la ficha de registro de Hospedaje, en caso de existir un problema se muestra el respectivo mensaje.	✓
Listar ficha de registro de hospedaje	Se lista la información de la ficha de registro de Hospedaje registrada.	✓
Eliminar ficha de registro de hospedaje	El sistema elimina correctamente la ficha de registro de Hospedaje del id que se ha enviado, en caso de existir errores muestra un mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU013.

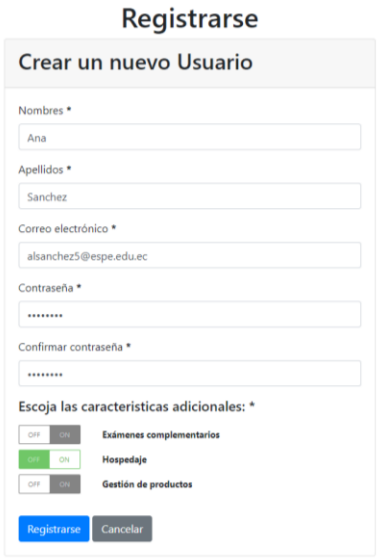
Validación Sprint 3

En la siguiente sección, se comprueba la validez de la autenticación del sistema. Para validar, se necesita de una interfaz de usuario así se entenderá de mejor manera la funcionalidad de este.

Registro de Usuario. En la figura 43, se observa que, al guardar todos los campos de manera correcta se crea un nuevo usuario. En la figura 44, se puede visualizar que, al momento de ingresar datos erróneos, se enviara las respectivas alertas y no dejara guardar los datos en la base. Al obtener datos correctos, se cumple las pruebas de funcionalidad esperadas en la autenticación.

Figura 43

Creación con éxito de un usuario



Nota. Se observa los datos a llenar, los mismos han sido aceptados y guardados para el inicio de sesión.

Figura *44*

Validación de datos

Nota. Se observa como al ingresar una contraseña no valida, el usuario no se registra y nos arroja las validaciones para tener en cuenta para el inicio de sesión.

Inicio de sesión. En la figura 45, se observa que, al ingresar un usuario erróneo, el el inicio de sesión no es permitido. En la figura 46, se puede visualizar que, al momento de ingresar datos correctamente, se ingresa sesión de manera correcta.

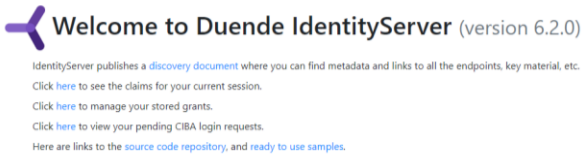
Figura 45

Ingreso de sesión erróneo

Nota. Se observa que al ingresar un usuario o contraseña incorrecta se niega el inicio de sesión y arroja una alerta de que existe un error en usuario o contraseña.

Figura 46

Ingreso de sesión con éxito



Nota. Se despliega una pantalla en la que se tiene acceso al sistema.

Lista de chequeo de sprint 3

Tabla 27

Lista de chequeo de la historia de usuario HU015.

Autenticación del Sistema		
Código: HU015	Estimación: 13 días	
Como Veterinario, quiero mantener un control de roles al ingresar al sistema para tener un registro ordenado de los médicos que ingresan al sistema.		
Prueba	Detalles	Estado
Validar datos en el formulario para registrar / actualizar	El sistema valida cada dato ingresado en el formulario y muestra advertencias en caso de no ser correcto.	✓
	El sistema no permite guardar los datos del formulario si al menos uno de los datos es incorrecto o un campo está vacío.	✓
Crear cuenta de veterinario	El sistema registra de forma exitosa el Paciente y muestra un mensaje de que el Paciente se ha creado. En caso de existir un problema, se muestra un error	✓
Iniciar sesión	El sistema actualiza de forma exitosa el Paciente, en caso de existir un problema se muestra el respectivo mensaje.	✓

Nota. Se muestra las validaciones realizadas en el sistema de acuerdo con las funcionalidades solicitadas en la HU015

Capítulo IV

Conclusiones y Recomendaciones

Conclusiones

- Se desarrolló los microservicios de agendamiento de turnos, citas y seguridades pertenecientes al SGCV mediante lo cual se determina que el análisis realizado con LPS utilizando FODA, permite tener una mejor visión de las necesidades de las clínicas veterinarias. La utilización de SAFe permitió que las características obtenidas, se implementen a través de microservicios de manera correcta, clara y ordenada.
- Al trabajar aplicando Desarrollo de Software Global co-localizado se debe organizar aplicando una metodología enfocada a equipos de trabajo en este caso se aplicó SCRUM lo que permitió realizar el desarrollo de planificación en base a los tiempos de cada sprint, este trabajo lo facilitó la herramienta Azure DevOps.
- Se utilizó GitFlow para el control del flujo de trabajo, esto ayudo a que se sincronice de mejor manera las ramas asignadas a cada miembro del equipo, permitiendo así optimizar la integración del código ya que a cada miembro del equipo se le asignó una rama de trabajo.
- El modelado UML aplicado en el proyecto permitió apreciar de forma visual el comportamiento y la estructura del sistema.
- Al realizar un desarrollo global de software co-localizado se necesitaba una base de datos robusta que permita gestionar cambios de varias personas al mismo tiempo por lo cual se eligió el gestor de base de datos PostgreSQL ya que se adapta a la tecnología Liquibase la cual permite sincronizar la base de datos y mantener un registro de los cambios que se realizan entre los desarrolladores.
- Al aplicar las pruebas de rendimiento se pudo comprobar que los microservicios generados, responden de manera rápida ante una carga masiva de datos. Logra

estabilizarse evitando así los cuellos de botella. Se obtiene microservicios estables y funcionales.

- Al aplicar las pruebas de funcionalidad se concluye que los microservicios cumplen con los requisitos planteados y funcionan de manera correcta.
- Al aplicar las listas de chequeo se comprueba que la LPS cumple con los requerimientos de los usuarios a través de soluciones que se ajustan a las necesidades de las clínicas veterinarias.

Recomendaciones

- Es recomendable utilizar el paradigma LPS cuando se requiere generar software que gestione múltiples procesos ya que a partir de la reutilización de elementos comunes se pueden generar varios productos en un menor tiempo de producción, con la reducción de recursos y costes al generar nuevos productos.
- Se recomienda aplicar LPS en equipo de alta afinidad ya que la comunicación es un punto clave durante todo el desarrollo ya que siempre existirá dependencia de unos con otros.
- En el desarrollo se recomienda utilizar software para el control de versiones como Gitflow para llevar un control de los cambios que se realizan en el código y para una mejor integración de este se recomiendo crear ramas para cada característica, una rama develop para la integración de las ramas y la rama main para subir los cambios con el código limpio y desplegarlo sin inconvenientes.
- Se recomienda la utilización de Azure DevOps para una mejor gestión de las actividades que se va a realizar durante el desarrollo de cada sprint.
- Se recomienda la utilización de SAFe ya que nos permite modelar de forma jerárquica las características comunes y específicas, facilitando la obtención de los requerimientos a través de las historias de usuario.
- Se recomienda la utilización de la metodología SCRUM ya que para el proceso de construcción de la LPS se requiere de equipos para un desarrollo planificado lo cual se obtiene mediante el uso de sprint, los cuales requieren una planificación mediante el producto backlog, además posee los roles y funcionalidades bien definidos lo cual evita ambigüedades al momento de distribuir el trabajo.

Bibliografía

Bayer, J., Fleger, O., Knauber, P., Laqua, R., Muthig, D., Schmid, K., . . . DeBaul, J. (1999). *PuLSE: A Methodology to Develop Software Product Lines.*

Bijwe, A., & Shankar, P. (2022). Challenges of Adopting DevOps Culture on the Internet of Things Applications - A Solution Model. (“Challenges of Adopting DevOps Culture on the Internet of Things ...”) *2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS).* (“2022 2nd International Conference on Technological Advancements in ...”) Tashkent, Uzbekistan: IEEE. doi: https://doi.org/10.1109/ICTACS56270.2022.9988182

Cedeno, A., Catuto, A., & Rodas-Silva, J. (2021, diciembre 5). "El uso de aplicaciones Web para la Gestión de clínicas veterinarias y su incidencia en la mejora de procesos administrativos." (“El uso de aplicaciones Web para la Gestión de clínicas ... - Dialnet”) *Ecuadorian Science Journal*, 109-120. doi: https://doi.org/10.46480/esj.5.4.174

Clements, P., & Northrop, L. (2001). *Software Product Line: Practices and Patterns.* Addison Wesley.

Cusco, B. (2022). *Desarrollo e implementación de una arquitectura DevOps para un sistema web basado en microservicios en infraestructuras basadas en código.* (“Desarrollo e implementación de una arquitectura DevOps para un sistema ...”) Quito: Universidad Politécnica Salesiana.

Días, Ó. (n.d.). *LÍNEAS DE PRODUCTO SOFTWARE.* Universidad del País Vasco.

Espinel, G. P., Carrillo, J. L., Flores, M. J., & Urbieta, M. (2022). Software Configuration Management in Software Product Lines: Results of a Systematic Mapping Study. (“Software Configuration Management in Software Product Lines: Results of ...”) *IEEE Latin America Transactions*, 718-730. doi: https://doi.otg/10.1109/TLA.2022.9693556

Espinosa, E. G. (2014). *"GESTIÓN DE CONFIGURACIÓN Y LÍNEA DE PRODUCTOS PARA MEJORAR EL PROCESO EXPERIMENTAL EN INGENIERÍA DEL*

SOFTWARE." (“Gestión de configuración y línea de productos para mejorar el proceso ...”) Madrid: Universidad Politécnica de Madrid.

Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E., & Peterson, S. A. (1990). *Feature-Oriented Domain Analysis (FODA) Feasibility Study.* Pennsylvania: Carnegie Mellon University.

La República. (2019, febrero 28). *Seis de cada 10 hogares del país tienen mascota*. Retrieved from proecuador: https://www.proecuador.gob.ec/seis-de-cada-10-hogares-del-pais-tienen-mascota/

León, Y. (2022). *Sistema electrónico de monitoreo de mascotas para la gestión de clínicas veterinarias utilizando VOIP e IOT.* (“UNIVERSIDAD TÉCNICA DE AMBATO FACULTAD DE INGENIERÍA EN SISTEMAS ...”) Ambato: Universidad Técnica de Ambato.

Lopez, D., & Maya, E. (2017). *Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web.* (“Arquitectura de Software basada en Microservicios para ... - RedCLARA”) Pichincha.

Malathi, S., & Sudhakar, P. (2018). Implementation of Software Refactoring Using FODA Tool. *2018 3rd International Conference on Communication and Electronics Systems*, 839-842. doi: https://doi.org/10.1109/CESYS.2018.8723986

Microsoft. (n.d.). *¿Qué es DevOps?* Retrieved from azure: https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-devops.

Northrop, L. M., & Clements, P. C. (2012). *A Framework for Software Product Line Practive, Version 5.0.* Pittsburgh: Software Engineering Institute.

Piattini, M. G., Vizcaíno, A., & Garcia, F. O. (2014). *Desarrollo Global de Software.* España: Grupo Editorial RA-MA.

Ponce, T. (2013, Enero 28). *Servicios a la orden de... las mascotas*. Retrieved from Revista Lideres: https://www.revistalideres.ec/lideres/servicios-orden-mascotas.html

Quishpe, C. E. (2018). *"Desarrollo de una aplicación web que mejore la gestión de los productos experimentales en Ingeniería de Software aplicando el paradigma de línea*

de producto software en el Grupo de Investigación GITBIO." (“Desarrollo de una aplicación web que mejore la gestión de los productos ...”) Latacunga: Universidad de las Fuerzas Armadas ESPE Extensión Latacunga.

Rincón, L., Giraldo, G., Mazo, R., Salinesi, C., & Díaz, D. (2013). Subconjuntos mínimos de corrección para explicar características muertas en podelos de líneas de productos. (“Subconjuntos mínimos de corrección para explicar características ...”) El caso de los modelos de características. *2013 8th Computing Colombian Conference*, 1-6. doi: https://doi.org/10.1109/ColumbianCC.

Diego Navarro, R., Cabrera Lozada, R. D., Diego Navarro, R., & Cabrera Lozada, R. D. (2020). *Aplicación basada en arquitectura de microservicios* [Info:eu-repo/semantics/bachelorThesis]. https://eprints.ucm.es/id/eprint/62080/

Espinosa Gallardo, E. G. (2014). *Gestión de configuración y línea de productos para mejorar el proceso experimental en ingeniería del software* [Http://purl.org/dc/dcmitype/Text, Universidad Politécnica de Madrid]. (“Gestión de configuración y línea de productos para mejorar el proceso ...”) https://dialnet.unirioja.es/servlet/tesis?codigo=89977

López Hinojosa, J. D. (2017). *Arquitectura de software basada en microservicios para desarrollo de aplicaciones web de la Asamblea Nacional* [MasterThesis]. (“PDF de programación - Arquitectura de Software basada en Microservicios ...”) http://repositorio.utn.edu.ec/handle/123456789/7603

Maida, E. G., & Pacienzia, J. (2015). *Metodologías de desarrollo de software*. https://repositorio.uca.edu.ar/handle/123456789/522

Rincon, L., Giraldo, G., Mazo, R., Salinesi, C., & Díaz, D. (2013). "Subconjuntos mínimos de corrección para explicar características muertas en modelos de líneas de productos." (“Subconjuntos mínimos de corrección para explicar características ...”) El caso de los modelos de características. *2013 8th Computing Colombian Conference (8CCC)*, 1-6. https://doi.org/10.1109/ColombianCC.2013.6637515

Seis de cada 10 hogares del país tienen mascota – PRO ECUADOR. (s. f.). Recuperado 19 de enero de 2023, de https://www.proecuador.gob.ec/seis-de-cada-10-hogares-del-pais-tienen-mascota/

Urteaga Pecharromán, A. (2015). *Aplicación de la metodología de desarrollo ágil Scrum para el desarrollo de un sistema de gestión de empresas* [BachelorThesis]. (“Aplicación de la metodología de desarrollo ágil Scrum para el ... - UC3M”) https://e-archivo.uc3m.es/handle/10016/23750

2013.6637515

Rincón, L., Rodríguez, G., Martínez, G., & Pabón, M. (2015). Creating virtual stores using software product lines: An application case. (“Creating virtual stores using software product lines: An application ...”) *2015 10th Computing Colombian Conference*, 71-78. doi:https://doi.org/10.1109/ColumbianCC.2015.7333414

Scaled Agile Framework. (2021, Julio 22). *SAFe 5 for Lean Enterprises*. Retrieved from scaledagileframework: https://www.scaledagileframework.com/safe-for-lean-enterprises/#:~:text=SAFe%205%20is%20built%20around%20the%20Seven%20Core,new%20competencies%20%28Organizational%20Agility%20and%20Continuous%20Learning%20Culture%29.

Scrum. (2023). *¿Qué es Scrum?* Retrieved from Scrum.org: https://www.scrum.org/resources/what-is-scrum/

Secretaria nacional de planificación. (2017). *Plan nacional del buen vivir 2017 2021.* Quito: Consejo nacional de planificación.

Software Productivity Consortium. (1993). *Reuse-Driven Software Processes Guidebook.* Virginia: Virginia Center of Excellence.

Weiss, D. M., & Chi, T. R. (1999). *Software Product-Line Engineering: A Family-Based Software Development Approach.* Addison Wesley.

Zavala, D. (2019). *Sistema informático enfocado a la web para el agendamiento de citas médicas y control de historia clínica para la clínica veterinaria "Entre Huellas y*

Bigotes" de la ciudad de Santo Domingo. Santo Domingo: Universidad regional autónoma de los Andes.

Anexos