

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO - CHILE



“ELABORACIÓN E IMPLEMENTACIÓN DE UN SISTEMA PARA DIGITALIZAR PROCESOS Y GENERAR KPI’S DEL FUNCIONAMIENTO DEL LABORATORIO FABLAB DE LA UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA”

JAIME PABLO RODRÍGUEZ GACITÚA

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Pedro Godoy Barrera
Profesor Correferente: Dayana Carrillo

Enero - 2025

DEDICATORIA

Esta memoria está dedicada a todas las personas que han contribuido a lo largo de mi vida y han hecho de mí la persona que soy hoy. Gracias a mi familia, amigos y amigas, compañeros y compañeras de colegio y universidad, así como a mis profesores y profesoras. A todos, muchas gracias.

AGRADECIMIENTOS

En primer lugar, quiero agradecer a mi familia. A mi mamá, Cecilia, por inculcarme desde muy pequeño el valor de estudiar. Siempre te preocupaste, no solo porque estudiara, sino también porque obtuviera buenos resultados.

A mi papá, Jaime, gracias por enseñarme que es importante terminar lo que uno empieza, y hacerlo bien, no a medias.

A mi abuela, Carmen, que fuiste una luz en tiempos de oscuridad. Siempre me dijiste que te hubiera gustado estudiar una carrera; estoy seguro de que habrías sido una excelente estudiante y profesional. Te quiero mucho.

A mi hermano José, un pilar fundamental durante mis años de estudio. Muchas gracias también por ayudarme con la sección de los indicadores, lo que demuestra la gran persona y profesional que eres como *Data Analyst*.

A mi hermano Ignacio, que siempre me ha motivado a estudiar por mi cuenta y fue quien me animó a ingresar y estudiar esta hermosa carrera. Eres uno de mis mayores ejemplos en la vida.

A mi hermana Camila, gracias por siempre tenderme una mano en el momento que la necesito y por demostrarme que, a pesar de los obstáculos que nos presenta la vida, siempre es posible seguir adelante.

A mi polola, Daniela, por impulsarme constantemente a ser mejor en todos los aspectos de mi vida, siempre con cariño y amor. Por no juzgarme y apoyarme en los momentos donde todo fue más difícil.

A mis perros, Fito y Otto, que fueron esenciales, no solo acompañándome en muchas noches de estudio y trabajo, sino también ayudándome a desestresarme y despejarme en nuestros paseos diarios.

Al equipo FabLab, gracias por darme el espacio y la confianza para crecer e innovar junto a ustedes.

Finalmente, a mis amigos de la universidad, Alejandro, Hans y Valeria, que fueron fundamentales en mis años de estudio y en apoyarme para concluir esta memoria. A todos, muchas gracias.

RESUMEN

Resumen— FabLab (*Fabrication Laboratory*) es un espacio diseñado para impulsar la innovación y el desarrollo de ideas, ofreciendo un entorno enfocado en la creación, el aprendizaje, la invención y la orientación. En esta memoria se describe la digitalización de los principales procesos del laboratorio FabLab de la Universidad Técnica Federico Santa María, mediante una aplicación web y una *API GraphQL*. El proyecto automatizó la mayoría de los procesos manuales del laboratorio, facilitando la gestión de máquinas, usuarios, horarios de ayudantes, proyectos y reservas de máquinas. Además, se implementaron indicadores clave sobre el funcionamiento y uso del laboratorio, contribuyendo a una transformación digital integral de sus operaciones principales.

Palabras Clave— FabLab; Procesos digitales; Aplicación Web; Indicadores Claves

ABSTRACT

Abstract— FabLab (*Fabrication Laboratory*) is a space designed to foster innovation and the development of ideas, providing an environment focused on creation, learning, invention, and guidance. This document describes the digitalization of the main processes in the FabLab laboratory at Universidad Técnica Federico Santa María, through a web application and a *GraphQL API*. The project automated most of the laboratory's manual processes, facilitating the management of machines, users, assistant schedules, projects, and machines reservations. Additionally, key performance indicators were implemented to monitor the laboratory's operations and user engagement, contributing to a comprehensive digital transformation of its main activities.

Keywords— FabLab; Digital Processes; Web Application; Key Performance Indicators

GLOSARIO

FabLab: *Fabrication Laboratory.*

KPI: *Key Performance Indicators.*

AWS: *Amazon Web Services.*

API: *Application Program Interfaces.*

ÍNDICE DE CONTENIDOS

RESUMEN	IV
ABSTRACT	IV
GLOSARIO	V
ÍNDICE DE FIGURAS	IX
ÍNDICE DE TABLAS	XI
INTRODUCCIÓN	1
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	2
1.1 Situación Actual	2
1.2 Detección del Problema	2
1.3 Objetivos	4
1.3.1 Objetivos Específicos	4
CAPÍTULO 2: MARCO CONCEPTUAL	6
2.1 FabLab	6
2.2 FabLab - Universidad Técnica Federico Santa María	7
2.3 Transformación Digital	7
2.4 KPI's	8
2.4.1 Power BI	8
2.5 Programación por Capas	8
2.6 Back End	9
2.6.1 NestJs	9
2.6.2 Express	9
2.6.3 GraphQL	9
2.6.4 PostgreSQL	10
2.7 Front End	10
2.7.1 React	10
2.7.2 Next.js	10
2.7.3 Material UI	11
2.8 Docker	11
2.9 Content Delivery Network (CDN)	11
2.10 Cloud Computing	11
2.11 Amazon Web Services	11
2.11.1 Identity and Access Management	12
2.11.2 Relational Database Service	12
2.11.3 Elastic Compute Cloud	12
2.11.4 Load Balancer	12

2.11.5	Elastic Beanstalk	12
2.11.6	CodeBuild	13
2.11.7	CodeDeploy	13
2.11.8	CodePipeline	13
2.11.9	Simple Email Service	13
2.11.10	Amazon Cognito	13
2.11.11	Lambda Function	14
2.11.12	CloudFront	14
2.11.13	Simple Storage Service	14
2.11.14	Elastic Container Registry	14
2.12	Microsoft Azure	14
2.12.1	Azure Active Directory	14
2.13	Autenticación y Autorización	15
2.13.1	OAuth 2.0	15
2.13.2	OpenID Connect	15

CAPÍTULO 3: PROPUESTA DE SOLUCIÓN 16

3.1	Identificación de Actores	16
3.2	Análisis, Priorización y Digitalización de Procesos	16
3.3	Requerimientos y Funcionalidades del Sistema	17
3.3.1	Requerimientos Funcionales	17
3.3.2	Funcionalidades del Sistema	19
3.4	Diseño de la Base de Datos	21
3.4.1	Usuarios	22
3.4.2	Tipos de Máquinas	24
3.4.3	Máquinas	24
3.4.4	Tipos de Proyectos	25
3.4.5	Proyectos	26
3.4.6	Miembros de los proyectos	27
3.4.7	Bloques	28
3.4.8	Horarios	29
3.4.9	Reserva	29
3.4.10	Bloques de una Reserva	30
3.5	Diseño de la API	31
3.5.1	NestJS	31
3.5.2	GraphQL	31
3.5.3	TypeORM	32
3.6	Autenticación y Autorización	34
3.6.1	Autenticación	35
3.6.2	Autorización	37
3.7	Comunicación entre el Front End y Back End	37
3.8	Cloud Computing	37
3.8.1	Proceso de construcción y despliegue	38
3.8.2	Alojamiento del Frontend y Backend	39

3.8.3	Envío de emails	39
3.8.4	Almacenamiento y disponibilidad de Archivos	39
3.8.5	Base de Datos	39
3.9	Diseño de la Interfaz	39
3.9.1	Colores	40
3.9.2	Tipografía	41
3.9.3	Diseño Responsivo	41
3.10	Gestión de las máquinas	43
3.11	Gestión de los proyectos	46
3.12	Gestión de los bloques	49
3.13	Gestión de los horarios	51
3.14	Gestión de Usuarios	53
3.15	Reservas	55
3.16	Indicadores Clave de Desempeño	56
CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN		58
4.1	Validación de la Identificación de Actores Principales	58
4.2	Validación del Análisis de Procesos y Priorización de Digitalización	58
4.3	Validación de la Plataforma Web	59
4.3.1	Análisis del Tiempo Promedio por Tarea	60
4.3.2	Análisis del Promedio de Éxito por Tarea	60
4.4	Validación de la API Escalable y Resiliente	61
4.4.1	Especificaciones de las Máquinas y Servicios Utilizados para la Prueba de Carga	62
4.4.2	Pruebas de Carga Realizadas con K6	62
4.4.3	Monitoreo y Resultados Iniciales	63
4.5	Validación de la Digitalización de Procesos y Generación de KPI's	68
4.5.1	Gestión de Usuarios	68
4.5.2	Horario de los ayudantes	70
4.5.3	Gestión de Máquinas	72
4.5.4	Gestión de Proyectos	73
4.5.5	Reservas de máquinas	75
4.6	Indicadores	79
CAPÍTULO 5: CONCLUSIONES		80
5.1	Conclusiones generales	80
5.2	Conclusiones específicas	81
5.3	Recomendaciones	83
5.4	Trabajo Futuro	84
5.5	Palabras finales del autor	84
ANEXOS		86
REFERENCIAS BIBLIOGRÁFICAS		102

ÍNDICE DE FIGURAS

1	Mapa de los FabLab en el mundo [Network, 2023]	7
2	Diagrama lógico de la base de datos, elaborado utilizando la herramienta <i>draw.io</i> . Este diagrama representa las entidades, relaciones y cardinalidades definidas para el sistema.	22
3	Sistema de autenticación y autorización del FabLab	34
4	Flujo de autenticación del Portal	36
5	Comunicación entre el Front End y Back End.	38
6	Portal FabLab Inicio	40
7	Logo FabLab	40
8	Colores utilizados en el Portal del FabLab. Imagen obtenida de: https://paletadecolores.online/crear-paleta/	41
9	Vista en Computadora	42
10	<i>Vista en Tablet</i>	43
11	Vista en Móvil	43
12	Tipos de Máquinas - Portal FabLab	44
13	Editar Tipo de Máquina - Portal FabLab	44
14	Máquinas - Portal FabLab	45
15	Editar Máquina - Portal FabLab	45
16	Tipo de máquina vista general - Portal FabLab	46
17	Detalle tipo de máquina - Portal FabLab	46
18	Tipos de proyectos - Portal FabLab	47
19	Editar tipo de proyecto - Portal FabLab	48
20	Proyectos - Portal FabLab	48
21	Editar proyecto - Portal FabLab	49

22 Bloques - Portal FabLab	50
23 Editar Bloque - Portal FabLab	50
24 Horarios - Portal FabLab	51
25 Horarios vista ayudante - Portal FabLab	52
26 Editar Horario - Portal FabLab	52
27 Vista Horario - Portal FabLab	53
28 Sincronización de usuarios - Portal FabLab	54
29 Usuarios - Portal FabLab	54
30 Editar usuario - Portal FabLab	55
31 Reportería presentando KPI's - Portal FabLab	57
32 Tiempo promedio por tarea - Prueba de usabilidad.	60
33 Promedio de éxito por tarea - Prueba de usabilidad.	61
34 Estado inicial antes de la prueba de carga.	64
35 Mapa de recursos del balanceador de carga al inicio de la prueba.	64
36 Estado medio durante la prueba de carga.	65
37 Mapa de recursos durante la prueba de carga.	66
38 Estado final de la prueba de carga.	66
39 Mapa de recursos durante el final de la prueba de carga.	67
40 Mapa de recursos al finalizar la prueba de carga.	67
41 Ingreso por primera vez a través del Pasaporte USM - Portal FabLab	68
42 Inicio - Portal FabLab	69
43 Ingreso por ayudante - Portal FabLab	69
44 Listado de usuarios - Portal FabLab	70
45 Vista de horarios para ayudantes - Portal FabLab	70

46	Vista administrador horarios - Portal FabLab	71
47	Interfaz para crear o editar horario - Portal FabLab	71
48	Tipos de máquinas creadas - Portal FabLab	72
49	Máquinas Impresoras 3D creadas - Portal FabLab	72
50	Máquina Impresora 3D creada - 3DP-01 - Portal FabLab	73
51	Tipos de proyectos creados - Portal FabLab	74
52	Tecnoplast proyectos - Portal FabLab	74
53	Proyectos de un usuario - Portal FabLab.	75
54	Proyecto creado por un usuario - Portal FabLab.	75
55	Reservas creadas - Portal FabLab	76
56	Editar una reserva creada - Portal FabLab	76
57	Creación de una reserva por un usuario - Portal FabLab.	77
58	Reserva creada por un usuario - Portal FabLab.	78
59	Horario del ayudante “Vicente Díaz” - Portal FabLab.	78
60	Indicadores en tiempo real del laboratorio FabLab usando la plataforma Power BI	79

ÍNDICE DE TABLAS

1	Extracto de los principales procesos que realizan el personal del FabLab de la Universidad Técnica Federico Santa María.	3
2	Tipos de máquinas del Laboratorio FabLab de la Universidad Técnica Federico Santa María [UTFSM, 2024]	4

INTRODUCCIÓN

La digitalización de procesos se ha convertido en un aspecto clave para mejorar la eficiencia operativa en diversos ámbitos, incluyendo el sector académico, donde es fundamental optimizar tanto los recursos como el tiempo de trabajo. En este contexto, el FabLab de la Universidad Técnica Federico Santa María, juega un papel fundamental al ofrecer a estudiantes y profesionales acceso a herramientas físicas y digitales para aprender y desarrollar proyectos innovadores. No obstante, el laboratorio enfrenta dificultades en su operación diaria debido a la cantidad de procesos manuales, como la gestión de usuarios, reservas de máquinas, gestión de proyectos, lo cual recae en el personal del FabLab y limita la eficiencia de sus servicios.

Para abordar esta problemática, este proyecto presenta el desarrollo e implementación de un sistema que automatiza los procesos clave del FabLab UTFSM. La solución se basa en una interfaz web y una API en GraphQL, permitiendo la gestión eficiente de usuarios, reservas y proyectos. Además, se integra un sistema de Indicadores Clave de Desempeño (KPI) en tiempo real para monitorear y evaluar el funcionamiento del laboratorio.

La metodología se centra en el análisis y digitalización de los procesos críticos, priorizando aquellos que impactan directamente en la operación y experiencia del usuario. La infraestructura de la plataforma se aloja en *Amazon Web Services* para asegurar escalabilidad y disponibilidad.

En cuanto a la organización de este documento, se presentan primero el contexto y la problemática, seguidos de un marco conceptual que describe las tecnologías y principios de diseño empleados. Luego, se detalla la propuesta de solución y el proceso de implementación. Finalmente, se incluye una sección de validación y conclusiones donde se analizan los resultados obtenidos y se proponen recomendaciones para el desarrollo futuro del FabLab.

CAPÍTULO 1

DEFINICIÓN DEL PROBLEMA

1.1. Situación Actual

La digitalización de procesos no solo es crucial para una empresa, *pyme*, servicios de salud o instituciones gubernamentales. Hoy en día, es primordial en los contextos académicos como universidades, colegios o laboratorios. Lograr una transformación digital conlleva un impacto directo en la sociedad, economía, política y cultura [Dussel, 2010].

El Laboratorio Fablab de la Universidad Técnica Federico Santa María fue creado el año 2014 como una iniciativa estudiantil con el objetivo de entregar un conjunto de herramientas físicas y digitales para que cualquier persona pueda crear, y transformar sus ideas en algo material [UTFSM, 2024]. El equipo del laboratorio se encuentra conformado por el director, el cual es un académico de la Universidad, un *Lab Manager* que se encuentra contratado por tiempo completo, y un equipo de ayudantes, que son estudiantes de diversas carreras (Ingeniería Civil Mecánica, Industrial, Informática, Ingeniería en Diseño de Producto, Ingeniería Comercial, entre otros.) Los ayudantes, tienen que aprender a conciliar el trabajo dentro del laboratorio con la carga académica de la universidad, lo cual a veces no se puede lograr del todo, debido principalmente a la poca automatización de procesos sumamente repetitivos como es la reserva de una máquina, capacitaciones de los ayudantes y estudiantes. También hay que estar pendiente del estado de las herramientas y máquinas del laboratorio, como su propio mantenimiento. Además, constantemente hay un cambio de *staff* de los ayudantes y se debe mantener el buen funcionamiento de los procesos de una forma transparente a los usuarios del laboratorio.

1.2. Detección del Problema

A pesar de que el laboratorio ha estado en funcionamiento durante aproximadamente 10 años, una gran cantidad de procesos aún se realiza de manera manual, lo que genera una carga de trabajo considerable para los ayudantes del laboratorio. Estos ayudantes tienen como objetivo principal brindar capacitaciones sobre el uso de las máquinas y proporcionar las herramientas necesarias a cualquier persona interesada, ya sea de la comunidad USM o un usuario externo con membresía.

En la tabla 1 se presenta un extracto de los 10 principales procesos actuales en el FabLab y el estado de su automatización [Ruiz, 2022]. Se observa que el 90 % de los procesos son manuales, lo cual impacta directamente en el rendimiento de los ayudantes y, por ende, en el funcionamiento general del laboratorio. Además, la alta dependencia de procesos manuales impide contar con información en tiempo real y con indicadores generales del laboratorio, como la cantidad de uso de las máquinas, el número de usuarios del laboratorio, la cantidad

de usuarios capacitados, y la motivación o propósito por el cual los usuarios hacen uso de las instalaciones.

Uno de los problemas más críticos radica en la gestión del uso de las máquinas, ya que se utiliza un método de orden de llegada y el registro se lleva a cabo en papel. Este enfoque genera múltiples inconvenientes. El uso de papel para registrar datos no permite tener la información centralizada ni adecuadamente resguardada para un análisis posterior, por ejemplo generación de indicadores. Aunque en ocasiones estos datos son trasladados a una hoja de cálculo en el *software Excel* ¹, este proceso no siempre se realiza debido a diversos factores, como desconocimiento, pérdida de información u otros imprevistos externos.

Además, la ausencia de una organización clara sobre el uso de las máquinas provoca descontento entre los usuarios. Es común que al llegar al laboratorio, descubran que la máquina que desean utilizar está ocupada o en mantenimiento, lo que afecta su experiencia.

Para los ayudantes del laboratorio, este sistema representa una carga adicional. Llevar el registro manualmente en papel y luego traspasarlo a *Excel* es un proceso engorroso y propenso a errores. Asimismo, deben coordinar manualmente los horarios con los usuarios para asegurar el acceso a las máquinas, lo que complica aún más su labor diaria. La digitalización de este proceso, permitiendo que los usuarios realicen reservas de las máquinas, no solo optimizaría la gestión de recursos, sino que también mejoraría la satisfacción tanto de los usuarios como del personal del laboratorio.

Procesos	Manual
Creación y análisis de KPI's, Reportes e informes	Sí
Gestión de usuarios	Si
Actualización de existencias del inventario de máquinas	Sí
Gestión de horarios de turnos	Sí
Reserva de máquinas	Sí
Gestión de horarios de las capacitaciones	Sí
Capacitaciones en máquinas y herramientas	No
Ingreso y estado de proyectos	Sí
Gestión de talleres	Sí
Gestión de programas	Sí

Tabla 1: Extracto de los principales procesos que realizan el personal del FabLab de la Universidad Técnica Federico Santa María.

A pesar de estas limitaciones, el laboratorio ha implementado algunas soluciones parciales, por ejemplo, las impresoras 3D, utilizan un *software* externo ² que permite imprimir de forma remota y registra todo el uso de las impresoras. Sin embargo, este es el único tipo de máquina que cuenta con esta digitalización de la información, dentro de un conjunto de 7 tipos de máquinas diferentes (Ver tabla 2).

¹<https://www.microsoft.com/es-cl/microsoft-365/excel>

²3D Printer OS: <https://www.3dprinteros.com/>

Tipos de Máquinas
Impresoras 3D
Cortadora Láser
Impresoras 3D Resina
Cortadora de Vinilo
Torno
Router CNC
Cirquid

Tabla 2: Tipos de máquinas del Laboratorio FabLab de la Universidad Técnica Federico Santa María [UTFSM, 2024]

1.3. Objetivos

El objetivo general de este proyecto es desarrollar e implementar un sistema que permita digitalizar los principales procesos del laboratorio FabLab de la Universidad Técnica Federico Santa María, así como generar KPI's en tiempo real para monitorear su funcionamiento. Al digitalizar procesos clave, como la gestión de usuarios, la reserva de máquinas, la actualización del inventario de máquinas y la gestión de proyectos, se busca reducir la cantidad de tareas manuales realizadas por los ayudantes del laboratorio. De esta forma, se espera disminuir su carga laboral, mejorar la eficiencia operativa del FabLab y optimizar la experiencia de los usuarios. Además, la implementación de un sistema de generación de KPI's permitirá un monitoreo continuo y preciso del desempeño del laboratorio, facilitando la toma de decisiones basadas en datos concretos.

1.3.1. Objetivos Específicos

Los objetivos específicos son:

- Analizar los principales procesos del laboratorio, ver Tabla 1, identificando aquellos que son susceptibles de digitalización, y priorizar los procesos claves, tales como la creación y análisis de KPI's, la gestión de usuarios, la gestión de máquinas, la gestión de turnos de los ayudantes, la reserva de máquinas, y la gestión de proyectos.
- Identificar a los actores principales involucrados en los procesos a digitalizar, asegurando su adecuada integración en el sistema.
- Desarrollar una plataforma web intuitiva y accesible para los usuarios y miembros del FabLab, que permita la gestión eficiente de los procesos digitalizados. Para validar este

objetivo, se realizará una prueba de usabilidad con 5 usuarios, de los cuales al menos el 90 % deberá completar las 5 tareas propuestas en un tiempo igual o inferior a 120 segundos.

- Diseñar una API escalable y resiliente que soporte las necesidades del sistema, garantizando su robustez y adaptabilidad a futuros requerimientos. Para validar estas características, se realizarán pruebas de carga enfocadas en el porcentaje de uso de CPU, de manera que el escalado automático de instancias se active al superar ciertos umbrales. Esto permitirá evidenciar la resiliencia del sistema, ya que podrá mantener un nivel de servicio adecuado incluso ante fallas o alzas imprevistas de demanda, adaptándose dinámicamente para sostener su rendimiento.
- Implementar el sistema para la digitalización de procesos y la generación automatizada de KPI's esenciales, como el número de reservas, usuarios activos, proyectos en curso, número de estudiantes clasificados por carrera y género y el total de proyectos asociados a cada máquina reservada, asegurando un monitoreo constante y en tiempo real del desempeño del laboratorio.

CAPÍTULO 2

MARCO CONCEPTUAL

2.1. FabLab

FabLab (*Fabrication Laboratory*) es un espacio dedicado a la creación, aprendizaje, invención y asesoramiento, donde se fomenta la innovación y el desarrollo de nuevas ideas [Foundation, 2022]. La iniciativa fue concebida por Neil Gershenfeld en el MIT³ *Center for Bits and Atoms* en 2006, marcando el inicio de una nueva era que está transformando la economía global al difuminar las fronteras entre el mundo físico y el digital [Fiaidhi y Mohammed, 2018]. Según *FabLab Network*⁴, actualmente existen más de 1750 FabLabs en 100 países alrededor del mundo (ver Figura 1), los cuales ofrecen una amplia gama de programas educativos para todas las edades y servicios profesionales [Network, 2023].

Con el tiempo, han surgido diversas comunidades de FabLabs a nivel global y local. Entre las más destacadas se encuentra *Fab Foundation*, fundada en 2009 como una organización sin fines de lucro en EE.UU., con el objetivo de facilitar y apoyar el crecimiento de la red internacional de laboratorios [Foundation, 2022]. Otra comunidad relevante es *FabLab Network*, que abarca desde laboratorios comunitarios hasta centros de investigaciones avanzadas. Esta red es una extensión de *fablabs.io*,⁵ y funciona como una red de fabricación, un campus de educación técnica distribuida y un laboratorio de investigación colaborativo, centrado en la digitalización de la fabricación y en la promoción de la próxima generación de manufactura y producción personalizada [Network, 2023].

En el contexto latinoamericano, destaca *FabLAT - Red Latinoamericana de FabLabs*, fundada por un grupo de profesionales y visionarios socialmente comprometidos, cuyo objetivo es influir positivamente en la región mediante la unión de esfuerzos y la creación de alianzas entre personas y organizaciones. Su misión principal es proporcionar soporte tecnológico y fomentar el desarrollo de redes de trabajo P2P (*peer to peer*), impulsando una economía colaborativa en un ecosistema de educación y turismo emprendedor en la era digital en América Latina [Network, 2023].

Finalmente, en el ámbito local, se encuentra la Red Chilena de FabLabs, que fue reactivada en 2019 como respuesta a la llegada de la pandemia a Chile. A través del trabajo colaborativo, esta red ha impulsado proyectos de alto impacto a nivel local y regional. Actualmente, la Red Chilena de FabLabs está compuesta por 17 laboratorios distribuidos a lo largo del país [de Innovación UC, 2022].

³Massachusetts Institute of Technology

⁴Comunidad abierta y creativa de fabricantes, artistas, científicos, ingenieros, educadores, estudiantes, aficionados y profesionales

⁵<https://fablabs.io/>

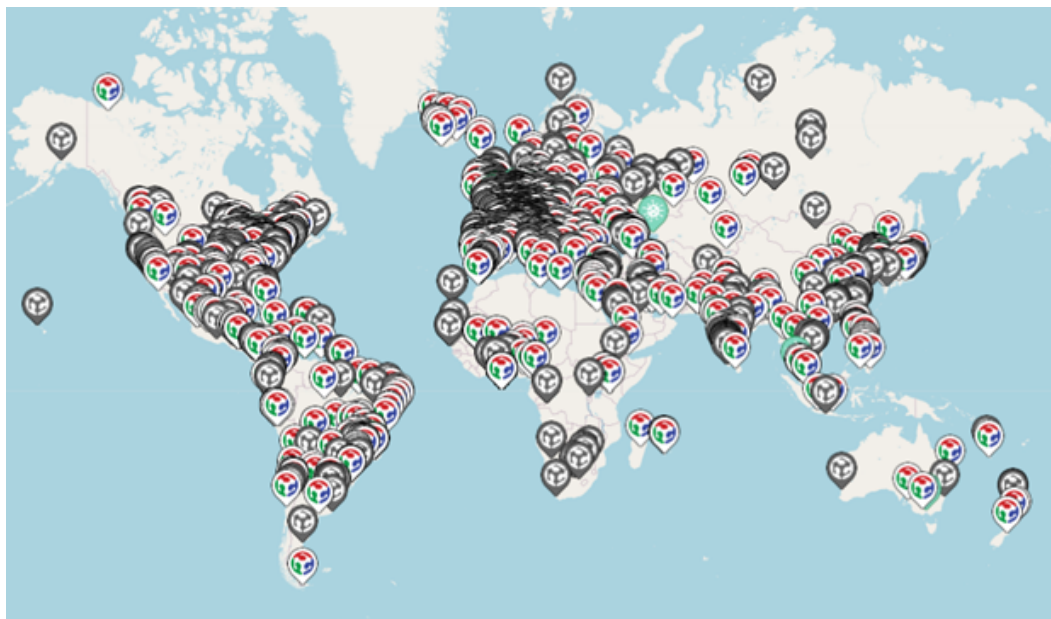


Figura 1: Mapa de los FabLab en el mundo [Network, 2023]

2.2. FabLab - Universidad Técnica Federico Santa María

El laboratorio de fabricación (FabLab) de la Universidad Técnica Federico Santa María (FabLab UTFSM) nace el año 2014 como una iniciativa estudiantil con el objetivo de entregar herramientas física y digitales para que cualquier persona pueda crear, y transformar sus ideas en algo material [UTFSM, 2024]. Fablab UTFSM es parte de la red global de *FabLab's* y se encuentra equipado con diversas máquinas de fabricación digital, que permiten realizar procesos de fabricación aditiva, circuitos impresos, corte láser, grabado de superficies y fresado; además cuenta con un torno convencional para procesos de torneear [Rojas, 2021]

2.3. Transformación Digital

La transformación digital se refiere al proceso de integrar tecnologías digitales en todos los aspectos de una organización para cambiar fundamentalmente la manera en que opera y ofrece valor a sus usuarios. Este concepto va más allá de la simple adopción de nuevas tecnologías; implica una reestructuración organizacional que mejora la eficiencia operativa, impulsa la innovación y permite una mejor toma de decisiones basada en datos [Morakanyane *et al.*, 2017]

El proceso de transformación digital en el FabLab no se limita únicamente a la automatización de procesos mediante tecnología, sino que también implica una adopción continua y un aprendizaje constante por parte tanto del personal que trabaja allí como de los usuarios.

2.4. KPI's

Los *Key Performance Indicators* (KPI's), o Indicadores Clave de Desempeño, son métricas utilizadas para medir el rendimiento y éxito de un proyecto, organización o proceso específico. Los KPI's permiten evaluar de manera objetiva el progreso hacia los objetivos establecidos, facilitando la toma de decisiones informadas y la identificación de áreas que requieren mejoras [Parmenter, 2015].

La implementación de KPI's en tiempo real en el FabLab permite un monitoreo continuo del desempeño, lo que facilita la detección de problemas operativos y la toma de decisiones basadas en datos concretos. Esto, a su vez, contribuye a la mejora continua de los procesos y la optimización de los recursos disponibles.

2.4.1. Power BI

Power BI, desarrollado por *Microsoft*, es una plataforma unificada y escalable diseñada para la inteligencia empresarial (*Business Intelligence*) y el análisis de datos de autoservicio [Microsoft, 2024]. Su intuitiva interfaz permite a los usuarios crear, personalizar y compartir paneles interactivos conectados a diversas fuentes de datos, facilitando la toma de decisiones informadas.

2.5. Programación por Capas

La programación por capas se basa principalmente en separar el diseño del software en capas independientes, así facilitando la gestión, mantenimiento y escalabilidad del sistema. Se pueden encontrar 3 capas, la capa de presentación, la capa de dominio y la capa de acceso a los datos [Martin Fowler, 2002]

- Capa de presentación: Encargada de proporcionar la información e interacción con el usuario, por ejemplo, a través de una interfaz de usuario.
- Capa de dominio: Es la encargada de satisfacer toda la lógica de negocio del sistema. Generando cálculos, validando, almacenando o gestionando la información del sistema.
- Capa de acceso a los datos: Esta capa es la encargada de la lógica de la comunicación con otros sistemas o fuentes de datos como bases de datos, sistema de mensajes u otros sistemas externos.

En una arquitectura de software en capas, la capa de presentación se asocia comúnmente con el front end, que maneja la interacción con el usuario. Por otro lado, las capas de dominio

y acceso a datos se corresponden con el back end, donde se gestionan la lógica de negocio y el acceso a las bases de datos y otros sistemas externos [Martin Fowler, 2002]

2.6. Back End

Normalmente el back end de una aplicación se refiere a la parte del software encargada de la lógica de aplicación y el manejo de los datos. Esta capa es fundamental en la ejecución de las funciones del sistema, incluyendo el procesamiento de operaciones internas, la administración de bases de datos y la implementación de la lógica de negocio. Funciona recibiendo solicitudes de la capa de presentación, procesándolas según las reglas de negocio establecidas, interactuando con bases de datos u otros servicios necesarios, y enviando los resultados apropiados de vuelta a la capa de usuario. Esta distribución de tareas fomenta una mayor modularidad, facilitando el mantenimiento y la escalabilidad del sistema.

2.6.1. NestJs

NestJs es un *framework* para la construcción de aplicaciones *Node.js* de forma escalable y eficiente. Esta basado fuertemente en el lenguaje de programación *Typescript* y combina elementos de la programación orientada a objetos, programación funcional y programación funcional reactiva. Es importante mencionar que por debajo usa por *default* a *Express*, un robusto *framework* *Http Server*.

Finalmente, no menos importante. La arquitectura de este *framework* está basada fuertemente en la arquitectura de *Angular* ⁶, la cual permite que los desarrolladores puedan crear aplicaciones fuertemente *testeables*, escalables, de baja acoplación y de fácil mantención [Nestjs, 2023]

2.6.2. Express

Express es un *framework* de *Node.js* que provee un conjunto de funcionalidades para hacer aplicaciones web. Como por ejemplo manejadores de peticiones *http*, enrutamiento de rutas, *middlewares*, entre otros [Express.js, 2023]

2.6.3. GraphQL

GraphQL es un lenguaje de consultas para las *APIs* el cual permite a los clientes solicitar exactamente los datos que necesitan [GraphQL, 2023]. Esto provee una serie de ventajas si lo

⁶<https://angular.io/>

comparamos con las *APIs REST*, otorgando más flexibilidad a los usuarios al momento de hacer sus solicitudes. Se genera una documentación automática. Se reduce el número de solicitudes y el tamaño de las respuestas.

2.6.4. PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto y altamente robusto, conocido por su fiabilidad, flexibilidad y conformidad con los estándares SQL. El origen de PostgreSQL se remonta a 1986 como parte de un proyecto *POSTGRES*⁷ de la Universidad de California Berkeley y cuenta con más de 35 años de desarrollo activo [PostgreSQL, 2024]

2.7. Front End

El front end es la interfaz donde un usuario puede interactuar con un sitio web, una aplicación o una aplicación móvil. Comúnmente para crear interfaces de usuarios se usa *HTML*, *CSS* y *Javascript* como herramientas de desarrollo [Lindley, 2023]

2.7.1. React

React es una famosa librería que permite construir interfaces de usuario a partir de piezas individuales llamadas componentes. *React* tiene una sintaxis llamada *JSX* la cual permite ingresar código *JavaScript* y *HTML*. Sin embargo, es más estricto que *HTML* [React, 2023]

2.7.2. Next.js

Next.js es un *framework* de *React* que permite crear aplicaciones web *full-stack*. *Next.js* abstrae toda la lógica del empaquetado y compilación de la aplicación. Además, provee varias funcionalidades como el enrutamiento, renderización por parte del cliente y servidor (*Client-side* y *Server-side rendering*, *Data Fetching*), optimización de imágenes, soporte del lenguaje de programación *Typescript* y más [Next.js, 2023]

⁷<https://www.postgresql.org/docs/current/history.html>

2.7.3. Material UI

Material UI es una librería diseñada para *React* la cual brinda una biblioteca con una gran cantidad de componentes para la interfaz de usuario [UI, 2023]. Al tener una gran cantidad de componentes, los desarrolladores se enfocan en las personalización de ellos más que en la lógica del componente, así permitiendo una reducción de tiempo en el desarrollo de la aplicación.

2.8. Docker

Docker puede empaquetar unidades de un software en un entorno llamado contenedor. Los contenedores son construidos por imágenes, las cuales proporcionan todo lo necesario para que el software funcione. Incluyendo librerías, herramientas de sistema, código, entre otros. Docker se utiliza para diversos fines, como el desarrollo y prueba de aplicaciones en entornos aislados, el despliegue de aplicaciones en la nube y la creación de infraestructuras escalables y portátiles [Poulton, 2017]

2.9. Content Delivery Network (CDN)

Content Delivery Network (CDN) Es un conjunto de servidores distribuidos geográficamente al rededor del mundo. Una CDN almacena en caché el contenido en ubicaciones que están más cerca de los usuarios finales que el servidor de origen de los datos, lo que reduce la distancia que los datos tienen que viajar y mejora los tiempos de carga [Grigorik, 2013]

2.10. Cloud Computing

Cloud Computing es un modelo que permite el acceso a recursos informáticos a través de internet. Por ejemplo, a servidores, base de datos, almacenamiento, software, redes, entre otros. Los usuarios pueden usar estos recursos según su demanda y necesidad, sin la necesidad de mantener o poseer estas infraestructuras físicas [Mell y Grance, 2011]

2.11. Amazon Web Services

Amazon Web Services (AWS) es la plataforma de *cloud computing* más completa y adoptada en el mundo, ofreciendo más de 200 servicios integrales de centros de datos a nivel global [Service, 2023].

2.11.1. Identity and Access Management

Identity and Access Management (IAM) es un servicio de AWS que permite la administración de manera segura de los accesos y permisos a los recursos de AWS. Este servicio permite una gran seguridad ya que los permisos que provee a los recursos es de forma granular, así facilitando el seguimiento de las acciones que generan los usuarios en los recursos [Services, 2023i].

2.11.2. Relational Database Service

Amazon Relational Database Service (RDS) es un servicio de base de datos en AWS que permite la creación y gestión de base de datos relacionales. Ofrece una gran variedad de motores de base de datos como: *Aurora MySQL*, *Aurora PostgreSQL*, *MySQL*, *MariaDB*, *PostgreSQL*, *Oracle* y *Microsoft SQL Server* [Services, 2023k]

2.11.3. Elastic Compute Cloud

Amazon Elastic Compute Cloud (EC2) ofrece la plataforma de computación más amplia y profunda, con más de 600 instancias y la posibilidad de elegir el procesador, almacenamiento, red, sistema operativo, entre otros [Services, 2023f]. De esta forma provee una gran flexibilidad a los usuarios, ya que proporciona el control total sobre el entorno informático de las instancias.

2.11.4. Load Balancer

Elastic Load Balancing permite distribuir el tráfico de red entre múltiples destinos, como instancias *EC2*, de esta forma, mejora la disponibilidad y escalabilidad de la aplicación [Services, 2023h]

2.11.5. Elastic Beanstalk

AWS Elastic Beanstalk es un servicio que se usa para implementar y escalar aplicaciones web y servicios [Services, 2023g]. Tiene varias ventajas, como facilidad de uso, simplificando la implementación y gestión de las aplicaciones. Además permite escalar las aplicaciones, integrando un *Load Balancer*. Finalmente, tiene una gran compatibilidad con múltiples lenguajes de programación y entornos de trabajo.

2.11.6. CodeBuild

AWS CodeBuild es un servicio de integración continua completamente administrado que compila un código fuente, ejecuta pruebas y produce paquetes de software listos para su despliegue [Services, 2023c]. Tiene una amplia compatibilidad con distintos lenguajes y frameworks. Además puedes personalizar el entorno de compilación con una gran variedad de sistemas operativos disponibles o con *Docker*

2.11.7. CodeDeploy

AWS CodeDeploy es un servicio que automatiza los despliegues de software en diferentes servicios de computación como *Elastic Compute Cloud (EC2)*, *Elastic Beanstalk*, *Elastic Container Service (ECS)*, *AWS Lambda*, entre otros [Services, 2023d]. Este servicio provee una gran cantidad de ventajas como automatización en despliegues, evitando errores manuales, control de versiones, *rollbacks* y una gran compatibilidad entre versiones y aplicaciones.

2.11.8. CodePipeline

AWS CodePipeline es un servicio de entrega continua completamente administrado que permite automatizar el proceso de lanzamientos o actualizaciones de una aplicación . Para ello, se integra a otros servicios de AWS como *CodeSource*, *CodeBuild* y *CodeDeploy*.

2.11.9. Simple Email Service

Amazon Simple Email Service (SES) es un proveedor de servicios de correos electrónicos escalable y económico. Se puede integrar con otros servicios de AWS como *AWS Lambda*. Se puede usar para enviar correos transaccionales, correos electrónicos de marketing o correos electrónicos de boletines [Services, 2023m]

2.11.10. Amazon Cognito

Amazon Cognito es un servicio de autenticación y autorización. Brinda almacenamiento de identidades que es escalable a millones de usuarios. Ofrece funciones de seguridad avanzadas como *MFA*. Además, permite la autenticación con proveedores como *Google*, *Facebook*, entre otros [Services, 2023e]

2.11.11. Lambda Function

AWS Lambda es un servicio informático sin servidor y basado en eventos que permite ejecutar código sin necesidad de aprovisionar o administrar servidores [Services, 2023j]. Tiene grandes ventajas como la facturación solo por el tiempo de ejecución real del código, o la gran escalabilidad automática según la demanda del servicio.

2.11.12. CloudFront

Amazon CloudFront es un servicio de red de entrega de contenido *Content Delivery Network* (CDN) creado para ofrecer un alto rendimiento, seguridad y comodidad para los desarrolladores [Services, 2023b]

2.11.13. Simple Storage Service

Amazon Simple Storage Service (S3) es un servicio de almacenamiento de objetos que ofrece una gran escalabilidad, disponibilidad y seguridad. Gracias a los distintos tipos de clases de almacenamiento, se pueden optimizar costos, organizar los datos y configurar los accesos a los objetos [Services, 2023i]

2.11.14. Elastic Container Registry

Amazon Elastic Container Registry (ECR) es un servicio de registro de imágenes de contenedores. Ofrece la posibilidad de generar repositorios privados y público de las imágenes. Además de guardar las imágenes de Docker de forma segura y escalable [Services, 2023a]

2.12. Microsoft Azure

Microsoft Azure es la plataforma de servicios en la nube ofrecida por *Microsoft*. Ofrece una amplia gama de servicios de almacenamiento, bases de datos, servidores, red, entre otros [Azure, 2023b]

2.12.1. Azure Active Directory

Azure Active Directory (AD) es un servicio de autenticación y autorización. Provee varios métodos y funcionalidades como *Single Sign-on (SSO)*, *MFA* y acceso condicional que protege el

99,9 % de los ataques de ciberseguridad [Azure, 2023a]

2.13. Autenticación y Autorización

La autenticación y la autorización son conceptos que se relacionan pero que son totalmente diferentes. Por un lado la autenticación es el acto de identificar a un usuario o un dispositivo. Mientras que por otro lado la autorización es el acto de permitir o denegar a los usuarios y dispositivos los derechos de acceso [Auth0, 2024a]

2.13.1. OAuth 2.0

OAuth 2.0 es un protocolo diseñado específicamente para la autorización, no para la autenticación. Su función principal es otorgar acceso a recursos, como API remotas o datos de usuario.

Este protocolo utiliza tokens de acceso, que actúan como prueba de autorización. Aunque *OAuth 2.0* no prescribe un formato específico para estos tokens, uno de los formatos comúnmente utilizados es *JSON Web Token (JWT)* ⁸ en ciertos contextos [Auth0, 2024c].

2.13.2. OpenID Connect

OpenID Connect (OIDC) es un protocolo de identidad que se basa en los mecanismos de *OAuth 2.0*. Desarrollado por la Fundación *OpenID* ⁹, que incluye grandes empresas como Google y Microsoft, *OIDC* se centra en la autenticación de identidad. A diferencia de *OAuth 2.0*, que es un protocolo de autorización, *OIDC* permite verificar la identidad de un usuario ante un servicio de confianza. Además, puede compartir información del usuario, como nombre y dirección de correo electrónico, cuando se solicite [Auth0, 2024b].

⁸<https://jwt.io/>

⁹<https://openid.net/developers/how-connect-works/>

CAPÍTULO 3

PROPUESTA DE SOLUCIÓN

Actualmente, el laboratorio no cuenta con un sistema web que automatice los principales procesos que enfrenta día a día 1. Por lo tanto, se propone desarrollar un sistema desde cero. Este nuevo sistema seguirá una metodología de separación de capas (capa de presentación, capa de dominio y capa de acceso a los datos).

Para la capa de presentación, se creará una interfaz de usuario que permita a los usuarios del FabLab y a los administradores del laboratorio interactuar de manera eficiente con el sistema. En cuanto a la capa de dominio y la capa de acceso a los datos, se desarrollará una API que procese la información y la gestione a través de una Base de Datos. Esta estructura permitirá no solo la digitalización de procesos, sino también la generación y monitoreo en tiempo real de KPI's clave, como el número de reservas, usuarios activos, proyectos en curso, entre otros.

Finalmente, la infraestructura *cloud* para la solución estará alojada en *Amazon Web Services* (AWS), asegurando la escalabilidad y resiliencia del sistema.

3.1. Identificación de Actores

Para garantizar que el sistema logre digitalizar de manera efectiva los procesos clave del FabLab, es fundamental identificar a los actores principales involucrados en dichos procesos. Los actores se dividen en las siguientes categorías:

- **Usuarios del FabLab:** Estudiantes, docentes, miembros de la comunidad universitaria y personas externas con membresía, que utilizan las instalaciones y recursos del laboratorio para desarrollar sus proyectos.
- **Personal del FabLab:** Encargados de gestionar las operaciones del laboratorio, supervisar las reservas, mantener las máquinas y coordinar proyectos y talleres.
- **Administradores del FabLab:** Además de encargarse de los mismos procesos que el Personal del Fablab, los administradores, deben preocuparse de la gestión de los usuarios y la configuración de los bloques de horario del laboratorio.

3.2. Análisis, Priorización y Digitalización de Procesos

Una vez identificados los actores principales del laboratorio, se procederá a digitalizar algunos de los procesos destacados en la Tabla 1 del FabLab, con el objetivo de mejorar la eficien-

cia y reducir las tareas manuales. Para garantizar una digitalización efectiva, se ha priorizado la automatización de aquellos procesos que tienen un mayor impacto en la operación diaria del laboratorio, considerándolos fundamentales para su funcionamiento. Los procesos seleccionados para la digitalización son:

- **Creación y análisis de KPI's:** Automatización de la generación de indicadores clave de desempeño (KPI's) en tiempo real para monitorear el rendimiento y uso del laboratorio.
- **Gestión de usuarios:** Digitalización del registro y gestión de los usuarios y personal del FabLab.
- **Actualización de existencias del inventario de máquinas:** Sistema para mantener actualizado el inventario de máquinas y su estado.
- **Gestión de horarios de turnos:** Digitalización de la asignación y gestión de horarios de los ayudantes del laboratorio.
- **Reserva de máquinas:** Sistema de reservas en línea que permite a los usuarios crear y gestionar sus reservas de máquinas disponibles. Además, el personal del FabLab podrá modificar o actualizar el estado de las reservas si es necesario, asegurando una correcta administración de los recursos.
- **Ingreso y estado de proyectos:** Los usuarios pueden crear y editar sus proyectos en el sistema, registrando su progreso. El personal del FabLab también tendrá la capacidad de editar el estado de los proyectos, garantizando un control preciso de las actividades y la coherencia en la información.

Los demás procesos listados en la tabla, como la gestión de talleres, programas, y horarios de capacitaciones, no serán digitalizados en esta etapa debido a las prioridades establecidas, enfocándose en aquellos que mejorarán la eficiencia operativa y la experiencia del usuario.

3.3. Requerimientos y Funcionalidades del Sistema

El sistema desarrollado para el FabLab tiene como objetivo digitalizar los principales procesos manuales del laboratorio y facilitar su gestión a través de una interfaz web y una API GraphQL. Para garantizar que el diseño del sistema cubra las necesidades operativas, se definieron los siguientes requerimientos funcionales y las funcionalidades principales del sistema.

3.3.1. Requerimientos Funcionales

Los requerimientos funcionales especifican lo que el sistema debe lograr para satisfacer las necesidades del laboratorio FabLab. Estos son:

- El sistema debe gestionar el registro y autenticación de usuarios utilizando el Pasaporte USM para miembros de la Universidad Técnica Federico Santa María, mientras que para usuarios internos y externos del laboratorio se implementará *Amazon Cognito*.
- El sistema debe gestionar de manera integral las reservas de máquinas, incluyendo las siguientes funcionalidades:
 - Asociar cada reserva a un proyecto específico.
 - Permitir la selección del tipo de máquina deseada.
 - Sugerir máquinas disponibles según el tipo seleccionado.
 - Facilitar la selección de bloques de tiempo disponibles.
 - Validar la disponibilidad de máquinas y ayudantes para el horario y la fecha elegidos.
 - Permitir la edición y eliminación lógica de las reservas existentes.
- El sistema debe permitir la gestión de usuarios, incluyendo:
 - Registro de nuevos usuarios.
 - Edición y eliminación de usuarios.
 - Sincronización de usuarios entre *Amazon Cognito* y la base de datos del sistema.
- El sistema debe permitir la gestión integral de proyectos, ofreciendo las siguientes funcionalidades:
 - Crear, editar y realizar la eliminación lógica de tipos de proyectos.
 - Asociar proyectos a un tipo de proyecto específico.
 - Gestionar la creación, edición y eliminación lógica de proyectos individuales y grupales.
 - Permitir la asociación de usuarios a proyectos grupales.
 - Finalizar proyectos con la opción de registrar retroalimentación por parte de los usuarios.
- El sistema debe gestionar de manera eficiente las máquinas del laboratorio, incluyendo las siguientes funcionalidades:
 - Crear, editar y realizar la eliminación lógica de los tipos de máquinas.
 - Asociar máquinas a un tipo de máquina específico.
 - Crear, editar y realizar la eliminación lógica de las máquinas.
 - Configurar los bloques de tiempo mínimo y máximo durante los cuales una máquina puede ser utilizada en una reserva.
 - Registrar y gestionar el estado de las máquinas (activada, desactivada o en mantenimiento).

- El sistema debe gestionar los bloques de horarios con las siguientes funcionalidades:
 - Crear, editar y realizar la eliminación lógica de los bloques de tiempo.
 - Asociar cada bloque a un horario de inicio, un horario de término y un día específico.
- El sistema debe gestionar los horarios de los ayudantes, permitiendo las siguientes acciones:
 - Crear, editar y realizar la eliminación lógica de los horarios de los ayudantes.
 - Asociar cada horario de ayudante a los bloques de tiempo previamente creados.
- El sistema debe generar indicadores clave de desempeño (KPI) en tiempo real, incluyendo:
 - Número total de reservas realizadas.
 - Total de reservas realizadas mensualmente.
 - Total de estudiantes agrupados por carrera.
 - Total de estudiantes clasificados por género.
 - Total de proyectos asociados a cada máquina reservada.
 - Total de proyectos clasificados por tipo de proyecto.
 - Total de estudiantes activos mensualmente.
- El sistema debe garantizar la seguridad de los datos mediante autenticación y autorización basada en roles (usuarios, ayudantes, administradores).
- El sistema debe permitir la integración con servicios de AWS, como *Amazon Cognito* para autenticación, y *S3* para almacenamiento de archivos relacionados con proyectos, tipos de proyectos, máquinas y tipo de máquinas.

3.3.2. Funcionalidades del Sistema

Las funcionalidades del sistema corresponden a las características técnicas y módulos desarrollados que permiten implementar los requerimientos funcionales previamente definidos. Estas son:

- **Módulo de autenticación y autorización:** Gestiona el registro y autenticación de usuarios utilizando *Amazon Cognito* y el sistema Pasaporte USM, garantizando un acceso seguro. Además, implementa la autorización mediante roles y permisos, asegurando el control adecuado de los recursos del sistema.

- **Módulo de reservas:** Permite a los usuarios gestionar sus reservas a través de una interfaz. Incluye funcionalidades para visualizar reservas existentes, realizar nuevas reservas seleccionando horarios disponibles, asociar reservas a proyectos específicos, y elegir tanto el tipo de máquina como la máquina específica a utilizar.
- **Módulo de gestión de usuarios:** Facilita el registro, edición, eliminación y sincronización de usuarios en la base de datos del sistema y *Amazon Cognito*.
- **Módulo de gestión de tipos de proyectos:** Proporciona una interfaz que permite la creación, edición y eliminación lógica de tipos de proyectos. Además, facilita la visualización de todos los tipos de proyectos existentes en el sistema.
- **Módulo de gestión de proyectos:** A través de una interfaz, facilita la creación, edición y eliminación lógica de proyectos individuales o grupales. También permite la asociación de usuarios a proyectos y la recopilación de retroalimentación al momento de finalizar un proyecto.
- **Módulo de gestión de tipos de máquinas:** Permite a través de una interfaz la creación, edición y eliminación lógica de tipos de máquinas, así como la visualización de todos los tipos registrados en el sistema.
- **Módulo de gestión de máquinas:** Proporciona una interfaz que permite la creación, edición y eliminación lógica de máquinas individuales, asociándolas a un tipo de máquina específico. Además, facilita la visualización de todas las máquinas existentes y permite gestionar su estado operativo, como activada, desactivada o en mantenimiento.
- **Módulo de gestión de bloques de tiempo:** Proporciona una interfaz intuitiva para la creación, edición y eliminación lógica de bloques de tiempo. Permite definir cada bloque con un horario de inicio, un horario de término y un día específico, asegurando una correcta organización del calendario del laboratorio.
- **Módulo de gestión de horarios:** Proporciona una interfaz que permite la creación, edición y eliminación lógica de horarios de disponibilidad para los ayudantes del laboratorio. Cada horario se configura asociándolo a bloques de tiempo previamente creados.
- **Módulo de indicadores (KPI):** Utiliza un tablero dinámico desarrollado en *Power BI*, integrado directamente con la base de datos *PostgreSQL* alojada en el servicio *RDS* de *Amazon Web Services*, para generar y visualizar indicadores clave de desempeño (KPI) en tiempo real, incluyendo:
 - Número total de reservas realizadas.
 - Total de reservas realizadas mensualmente.
 - Total de estudiantes agrupados por carrera.
 - Total de estudiantes clasificados por género.

- Total de proyectos asociados a cada máquina reservada.
- Total de proyectos clasificados por tipo de proyecto.
- Total de estudiantes activos mensualmente.

Estas funcionalidades son fundamentales para el diseño de la base de datos, asegurando que cada tabla y relación soporte las operaciones descritas. El diseño modular garantiza la escalabilidad y el mantenimiento del sistema.

3.4. Diseño de la Base de Datos

Considerando los procesos a digitalizar, los actores principales del sistema, los requerimientos funcionales y las funcionalidades del sistema, se ha diseñado una base de datos compuesta por 10 tablas. Este diseño asegura que la información relevante esté organizada de manera eficiente para soportar las funcionalidades del sistema. A continuación, se presenta un diagrama elaborado con *draw.io*¹⁰ utilizando la notación UML (ver Figura 2). El diagrama está organizado por colores que representan los diferentes módulos del sistema: usuarios, proyectos, máquinas, reservas, bloques y horarios. Cabe destacar que el módulo de indicadores (KPI) no está representado con un color específico, ya que estos se generan a través de consultas realizadas directamente sobre la base de datos utilizando *Power BI*.

Dentro de las 10 tablas, se incluyen:

- **Tablas de Entidades Principales:** Estas tablas almacenarán información clave sobre los elementos individuales del sistema con respecto al dominio del problema. Aquí se aprecian tablas como *User*, *Machine*, *Project* y *Booking*.
- **Tablas de Tipos de Categorías:** Estas tablas definen y clasifican tipos específicos de entidades. Se puede encontrar tablas como *Machine Type* y *Project Type*.
- **Tablas de Bloques y Horarios:** Diseñadas para gestionar el tiempo en el sistema, permitiendo principalmente definir cuando se pueden utilizar los recursos del laboratorio. Las tablas contempladas son *Block* y *Schedule*.
- **Tablas de Relación:** También conocida como tablas intermedias, estas tablas están diseñadas para gestionar las relaciones *muchos a muchos* entre las entidades principales. La función principal es asegurar que las conexiones entre diferentes elementos estén claramente definidas y separadas de las entidades individuales. Aquí se encuentran tablas como *Project Member* y *Booking Block*.

¹⁰*draw.io* es una herramienta para realizar diagramas técnicos y organizativos, incluyendo modelos de bases de datos

Se utilizó una Base de Datos Relacional, implementando el sistema de gestión *PostgreSQL*, debido a la naturaleza del problema, que requiere manejar datos altamente estructurados y realizar operaciones seguras y coherentes en un entorno transaccional. Esto es particularmente relevante en un sistema que gestiona reservas en horarios predefinidos. La elección de *PostgreSQL* se justifica también por su compatibilidad con transacciones *ACID*¹¹, lo que garantiza la integridad y confiabilidad de las operaciones.

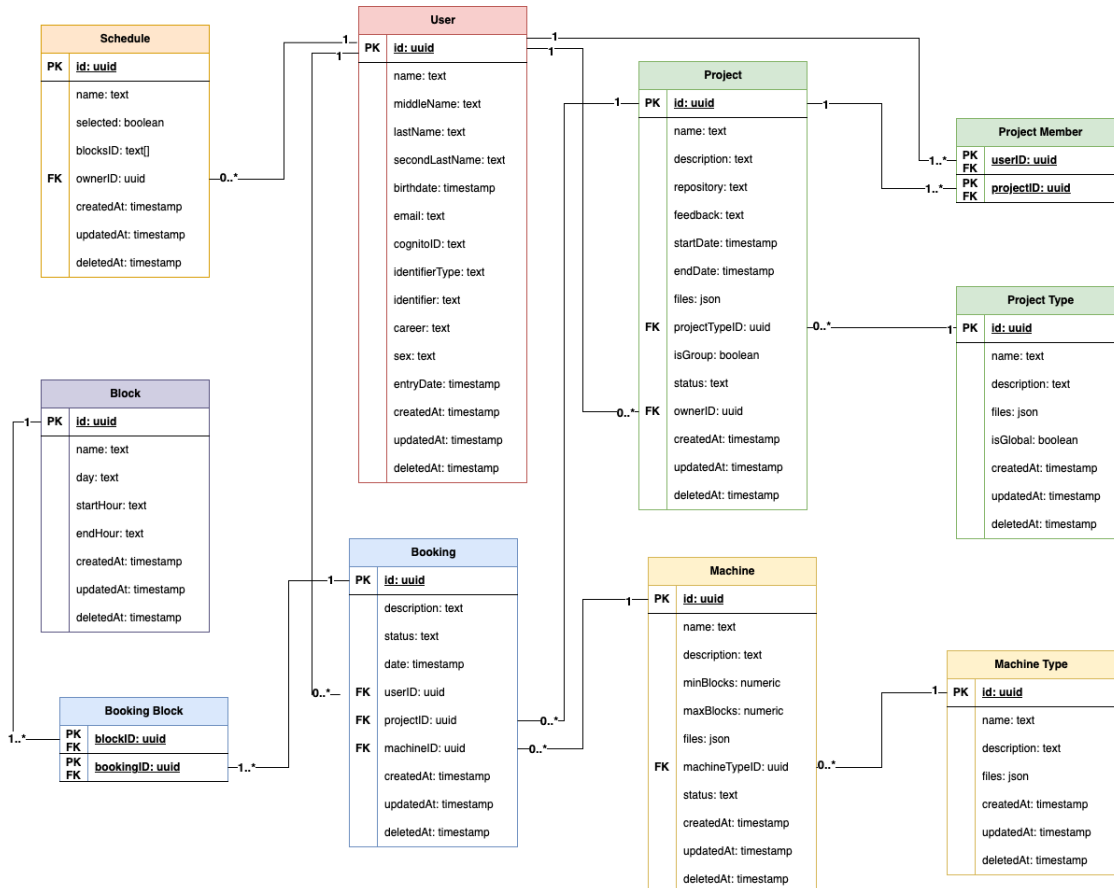


Figura 2: Diagrama lógico de la base de datos, elaborado utilizando la herramienta *draw.io*. Este diagrama representa las entidades, relaciones y cardinalidades definidas para el sistema.

A continuación, se explicarán con más detalle la naturaleza de cada tabla:

3.4.1. Usuarios

La tabla *User* almacena toda la información sobre los usuarios del sistema.

Los campos clave son:

¹¹Conjunto de propiedades que garantizan que las transacciones de una base de datos se realicen de manera fiable. Siendo este un acrónimo de: Atomicidad, Consistencia, Aislamiento y Durabilidad

- **id**: Identificador único del usuario en formato *uuid*.
- **name**: Primer nombre del usuario.
- **middleName**: Segundo nombre del usuario siendo este un valor opcional.
- **lastName**: Primer apellido del usuario.
- **secondLastName**: Segundo apellido del usuario.
- **birthdate**: Fecha de nacimiento del usuario.
- **email**: Dirección de correo electrónico del usuario.
- **cognitoid**: Identificador único proveniente del sistema de autenticación AWS *Cognito*.
- **identifierType**: Tipo de identificación, siendo valores posibles **RUT** o **PASAPORTE**.
- **identifier**: Valor de la identificación del usuario.
- **career**: Valor de la carrera que esta cursando el usuario, siendo este un valor opcional.
- **sex**: Genero del usuario, siendo valores posibles **Femenino**, **Masculino**, u **Otro**.
- **entryDate**: Año de ingreso de la carrera seleccionada siendo este un valor opcional.

Los campos de *metadata* son:

- **createdAt**: Fecha de creación del usuario.
- **updatedAt**: Fecha de actualización del usuario.
- **deletedAt**: Fecha de eliminación del usuario, usando un método *softdelete*, ¹².

Las relaciones principales son las siguientes:

- Se utiliza una tabla de relación o intermedia, denominada **Project Member**, para registrar a los miembros asociados a proyectos de carácter grupal.
- La tabla **User** está relacionada con **Project**, permitiendo identificar al usuario que creó cada proyecto.
- La tabla **User** está relacionada con **Schedule**, facilitando la asociación de un horario específico a un usuario.
- La tabla **User** está relacionada con **Booking**, lo que permite vincular las reservas realizadas por los usuarios.

¹²Técnica utilizada en base de datos para eliminar registros sin realmente eliminarlos. En lugar de borrar el registro se agrega un campo *deletedAt* agregando la fecha de eliminación y dejando el registro no válido, lo cual afectaría en las consultas que se realice al registro

3.4.2. Tipos de Máquinas

La tabla *Machine Type* define los diferentes tipos de máquinas que pueden existir en el laboratorio.

Los campos clave son:

- **id**: Identificador único del tipo de máquina en formato *uuid*.
- **name**: Nombre del tipo de máquina.
- **description**: Descripción del tipo de máquina siendo este un valor opcional.
- **files**: Ubicación de los archivos.

Los campos de *metadata* son:

- **createdAt**: Fecha de creación del tipo de máquina.
- **updatedAt**: Fecha de actualización del tipo de máquina.
- **deletedAt**: Fecha de eliminación del tipo de máquina.

Las relaciones son:

- Relacionada con la tabla **Machine** permitiendo que una máquina se asocie a un tipo específico.

3.4.3. Máquinas

La tabla *Machine* almacena información sobre las máquinas del laboratorio, independientemente de si la máquina se encuentra operativa o no.

Los campos clave son:

- **id**: Identificador único de la máquina en formato *uuid*.
- **name**: Nombre de la máquina.
- **description**: Descripción de la máquina siendo este un valor opcional.
- **minBlocks**: Número mínimo de bloques de tiempo durante los cuales se puede utilizar la máquina en una reserva.

- **maxBlocks:** Número máximo de bloques de tiempo durante los cuales se permite el uso de la máquina en una reserva.
- **files:** Ubicación de los archivos.
- **machineTypeID:** Identificado en formato *uuid* del tipo de máquina asociada.
- **status:** Estado de la máquina, siendo posible los siguientes valores: **ACTIVADA**, **DES-ACTIVADA**, **EN_MANTECIÓN**.

Los campos de *metadata* son:

- **createdAt:** Fecha de creación de la máquina.
- **updatedAt:** Fecha de actualización de la máquina.
- **deletedAt:** Fecha de eliminación de la máquina.

Las relaciones son:

- Relacionada con la tabla **Machine Type** permitiendo que una máquina se asocie a un tipo específico de máquina.
- Relacionada con la tabla **Booking**, lo cual permite asociar una máquina por reserva.

3.4.4. Tipos de Proyectos

La tabla *Project Type* define las categorías o tipos de proyectos que pueden ser gestionados en el laboratorio.

Los campos clave son:

- **id:** Identificador único del tipo de proyecto en formato *uuid*.
- **name:** Nombre del tipo de proyecto.
- **description:** Descripción del tipo de proyecto siendo este un valor opcional.
- **files:** Ubicación de los archivos.
- **isGlobal:** Valor booleano indicando si los proyectos asociados a este tipo de proyecto son de carácter global, lo que significa que cualquier usuario puede utilizarlos, independientemente de quién los haya creado.

Los campos de *metadata* son:

- **createdAt:** Fecha de creación del tipo de proyecto.
- **updatedAt:** Fecha de actualización del tipo de proyecto.
- **deletedAt:** Fecha de eliminación del tipo de proyecto.

Las relaciones son:

- Relacionada con la tabla **Project** permitiendo que un proyecto se asocie a un tipo de proyecto.

3.4.5. Proyectos

La tabla *Project* almacena información relevante sobre los proyectos realizados en el laboratorio. Esta tabla es fundamental, ya que para que un usuario pueda reservar una máquina, debe estar asociado a un proyecto. De esta manera se registra el propósito específico por el cual se está utilizando el laboratorio.

Los campos clave son:

- **id:** Identificador único del proyecto en formato *uuid*.
- **name:** Nombre del proyecto.
- **description:** Descripción del proyecto siendo este un valor opcional.
- **feedback:** La retroalimentación que puede dar un usuario al laboratorio luego de finalizar el proyecto.
- **repository:** Repositorio del proyecto, como la *url* de un proyecto en *GitHub*,¹³ siendo este un valor opcional.
- **startDate:** Fecha en donde se comenzó el proyecto.
- **endDate:** Fecha en donde se terminó el proyecto.
- **isGroup:** Valor booleano para saber si el proyecto es de carácter grupal o individual.
- **files:** Ubicación de los archivos.

¹³<https://github.com/>

- **status:** Estado del proyecto, siendo posible los siguientes valores: **EN_PROGRESO** y **FINALIZADO**.
- **ownerID:** Identificador único en formato *uuid* del usuario que creó el proyecto.
- **projectTypeID:** Identificador único en formato *uuid* del tipo de proyecto asociado.

Los campos de *metadata* son:

- **createdAt:** Fecha de creación del proyecto.
- **updatedAt:** Fecha de actualización del proyecto.
- **deletedAt:** Fecha de eliminación del proyecto.

Las relaciones son:

- Relacionada con la tabla **Project Type** permitiendo que un proyecto se asocie a un tipo de proyecto.
- Relacionada con la tabla **Project Member** permitiendo que un proyecto se asocie a distintos usuarios, cuando el proyecto es de carácter grupal.
- Relacionada con la tabla **User** permitiendo asociar un proyecto a un usuario.
- Relacionada con la tabla **Booking** permitiendo asociar un proyecto a una reserva.

3.4.6. Miembros de los proyectos

La tabla *Project Member* es una tabla de relación intermedia que gestiona la relación *muchos a muchos* entre las entidades de usuarios y proyectos. Esta tabla es fundamental para los proyectos de carácter grupal, ya que permite registrar y organizar a los usuarios que participan en cada proyecto.

Los campos clave son:

- **projectId:** Identificador único del proyecto en formato *uuid*, que actúa como clave foránea, estableciendo la relación con la tabla *Project*, y como parte de la clave primaria compuesta.
- **userId:** Identificador único del usuario en formato *uuid*, que actúa como clave foránea, estableciendo la relación con la tabla *User*, y como parte de la clave primaria compuesta.

Las relaciones son:

- Relacionada con las tabla **Project** y **User** permitiendo guardar los registros de los usuarios que pertenecen a un proyecto grupal.

3.4.7. Bloques

La tabla *Block* es un componente esencial en la gestión del laboratorio. Esta tabla permite a los ayudantes crear sus calendarios basándose en los bloques de tiempo definidos, y a los usuarios realizar reservas conforme a estos bloques. Generalmente, estos bloques corresponden a los horarios predefinidos por la Universidad Técnica Federico Santa María, Campus San Joaquín.

Los campos clave son:

- **id**: Identificador único del bloque *uuid*.
- **name**: Nombre del bloque.
- **day**: Día del bloque, siendo posible los siguientes valores: **LUNES, MARTES, MIÉRCOLES, JUEVES, VIERNES, SÁBADO y DOMINGO**.
- **startHour**: Corresponde a la hora de inicio del bloque.
- **endHour**: Corresponde a la hora de término del bloque.

Los campos de *metadata* son:

- **createdAt**: Fecha de creación del bloque.
- **updatedAt**: Fecha de actualización del bloque.
- **deletedAt**: Fecha de eliminación del bloque.

Las relaciones son:

- Relacionada con la tabla **Booking Block** la cual se utiliza como tabla para relacionar los bloques con las reservas.

3.4.8. Horarios

La tabla *Schedule* gestiona los horarios de disponibilidad de los ayudantes del laboratorio, permitiéndoles establecer los períodos en los que están disponibles para desempeñar sus funciones. Además, estos horarios sirven como referencia para la disponibilidad de los usuarios al momento de realizar reservas. Los horarios se configuran mediante un conjunto de bloques de tiempo predefinidos, garantizando una organización eficiente.

Los campos clave son:

- **id:** Identificador único del horario en formato *uuid*.
- **name:** Nombre del horario.
- **selected:** Valor booleano para ver si el horario está seleccionado.
- **ownerID:** Identificador único en formato *uuid* del usuario que creó el horario.
- **blocksId:** Identificadores de los bloques de tiempo seleccionados que conforman el horario.

Los campos de *metadata* son:

- **createdAt:** Fecha de creación del horario.
- **updatedAt:** Fecha de actualización del horario.
- **deletedAt:** Fecha de eliminación del horario.

Las relaciones son:

- Relacionada con la tabla **User**, que asocia el horario con un usuario.

3.4.9. Reserva

La tabla *Booking* es esencial en la gestión del laboratorio, ya que registra todas las reservas realizadas por los usuarios. Además, almacena el estado actual de cada reserva, asegurando un control preciso y actualizado de las actividades dentro del laboratorio.

Los campos clave son:

- **id:** Identificador único de la reserva en formato *uuid*.

- **description:** Descripción de la reserva, siendo este un valor opcional.
- **status:** Estado de la reserva. Los valores posibles son: **CREADA, CONFIRMADA, ANULADA,**
- **date:** Fecha de la reserva.
- **ownerID:** Identificador único en formato *uuid* del usuario que creó la reserva.
- **projectID:** Identificador único en formato *uuid* del proyecto asociado a la reserva.
- **machineID:** Identificador único en formato *uuid* de la máquina asociada a la reserva.

Los campos de *metadata* son:

- **createdAt:** Fecha de creación de la reserva.
- **updatedAt:** Fecha de actualización de la reserva.
- **deletedAt:** Fecha de eliminación de la reserva.

Las relaciones son:

- Relacionada con la tabla **Machine**, que indica qué máquina se está reservando.
- Relacionada con la tabla **Project**, para determinar el proyecto asociado a la reserva, lo que permite registrar el propósito del uso del laboratorio.
- Relacionada con la tabla **User**, lo que permite mantener un registro de qué usuario realizó la reserva.
- Relacionada con la tabla **Booking Block**, donde se registran los bloques de tiempo específicos en los que se realizó la reserva.

3.4.10. Bloques de una Reserva

La tabla *Booking Block* actúa como una tabla de relación, permitiendo asignar uno o varios bloques de tiempo a una reserva. De este modo, se gestiona la relación *muchos a muchos* entre las entidades Reserva y Bloque.

Los campos clave son:

- **bookingID:** Identificador único de la reserva en formato *uuid*, que actúa como clave foránea, estableciendo una relación con la tabla *Booking*, y como parte de la clave primaria compuesta.

- **blockID**: Identificador único del bloque en formato *uuid*, que actúa como clave foránea, estableciendo una relación con la tabla *Block*, y como parte de la clave primaria compuesta.

Las relaciones son:

- Relacionada con las tabla **Block** y **Booking** permitiendo guardar los bloques asociados a una reserva.

3.5. Diseño de la API

La API desarrollada para este proyecto se sustenta en tres componentes clave: el *framework* *NestJS*, el lenguaje de consulta *GraphQL*, y la herramienta *ORM TypeORM*. Estos fueron seleccionados cuidadosamente debido a sus características que se alinean con los objetivos específicos de crear una API escalable y resiliente para las necesidades del sistema, y de implementar un sistema para digitalizar los procesos seleccionados y la generación de KPI's.

3.5.1. NestJS

NestJS fue elegido por su robusta arquitectura modular y su integración nativa con *TypeScript*, lo que permite construir aplicaciones escalables y mantenibles. La capacidad de *NestJS* para manejar la inyección de dependencias y estructurar el código en módulos reutilizables facilita la gestión de un sistema complejo, donde distintas funcionalidades, como la gestión de usuarios, reservas y proyectos, deben interactuar de manera coherente y eficiente. Además, *NestJS* ofrece un entorno altamente flexible que se adapta bien a proyectos que requieren un rápido desarrollo y fácil mantenimiento, asegurando así que la API pueda evolucionar con el tiempo sin comprometer su estabilidad.

3.5.2. GraphQL

GraphQL se implementó en lugar de una *API REST* tradicional debido a su flexibilidad en el manejo de consultas y la eficiencia que proporciona en el consumo de datos. *GraphQL* permite a los clientes solicitar únicamente los datos que necesitan, reduciendo la sobrecarga en las comunicaciones y mejorando el rendimiento general del sistema. Esto lo convierte en una herramienta poderosa para la creación y consulta de KPI's, ya que facilita la elaboración de consultas precisas y personalizadas que pueden combinar datos de múltiples fuentes en una sola solicitud.

3.5.3. TypeORM

TypeORM por su lado, fue integrado como el ORM del sistema, ¹⁴, debido a su gran compatibilidad con PostgreSQL y su enfoque orientado a objetos para la gestión de datos. TypeORM simplifica las operaciones con la base de datos, desde la definición de entidades hasta la ejecución de migraciones, lo que garantiza la integridad de los datos y facilita su manipulación en un entorno de desarrollo ágil y escalable.

En conjunto, estos tres componentes forman la base tecnológica de la API, proporcionando un entorno de desarrollo sólido, flexible y eficiente, capaz de satisfacer las demandas actuales del sistema y de adaptarse a futuros desafíos con facilidad.

La API también cuenta con una documentación interactiva que permite explorar todas las consultas (*queries*) y mutaciones (*mutations*) disponibles. Esta documentación puede ser consultada a través de la siguiente URL: <https://dev.api.fablab.usm.cl/graphql>, donde los desarrolladores pueden experimentar con la API en tiempo real. En total, la API expone 33 *queries* y 38 *mutations*, permitiendo una amplia gama de operaciones que cubren todas las funcionalidades del sistema.

A continuación, se presentan dos ejemplos clave de interacciones con la API:

Mutación para Crear una Reserva:

```
mutation CreateBooking($createBookingInput:
  CreateBookingInput!) {
  createBooking(createBookingInput: $createBookingInput) {
    id
    user {
      id
      name
    }
    machine {
      id
      name
    }
    project {
      id
      name
    }
    date
    blocks {
      id
      day
```

¹⁴Object-Relational Mapping

```
    startHour
    endHour
  }
  status
  description
}
```

Consulta para Obtener los KPI del Sistema:

```
query Kpi($startDate: Timestamp!, $endDate: Timestamp!) {
  kpi(startDate: $startDate, endDate: $endDate) {
    bookings {
      total
      donutData {
        label
        value
        color
      }
      chartData {
        labels
        values
      }
    }
    machines {
      data {
        label
        value
        color
      }
    }
    projects {
      total
    }
    users {
      total
    }
  }
}
```

Estos ejemplos ilustran cómo se puede interactuar con la API para realizar operaciones críticas como la creación de reservas y la obtención de KPI. La respuesta a estas consultas incluye toda la información relevante, permitiendo su verificación y análisis inmediato en el sistema.

El esquema completo, junto con las operaciones principales, se encuentra documentado en detalle en los anexos 5.5, proporcionando un recurso valioso para desarrolladores que necesiten comprender o extender la funcionalidad de la API.

3.6. Autenticación y Autorización

La autenticación y autorización son elementos fundamentales en el sistema debido a la variedad de usuarios involucrados. Cabe destacar que, por su parte, el FabLab ya contaba con un sistema de autenticación y autorización configurado por el autor en 2022. Este sistema fue implementado utilizando la herramienta *Amazon Cognito* de AWS, para su integración con la plataforma *Moodle* ¹⁵ del laboratorio. La Figura 3 muestra el sitio web de autenticación del FabLab ¹⁶. En la figura se observa la opción “PasaporteUSM”, que permite el acceso a toda la comunidad de la Universidad Técnica Federico Santa María, y el “Login Normal”, que está destinado a los usuarios del FabLab bajo el dominio “fablab.usm.cl” y a usuarios externos a la universidad.

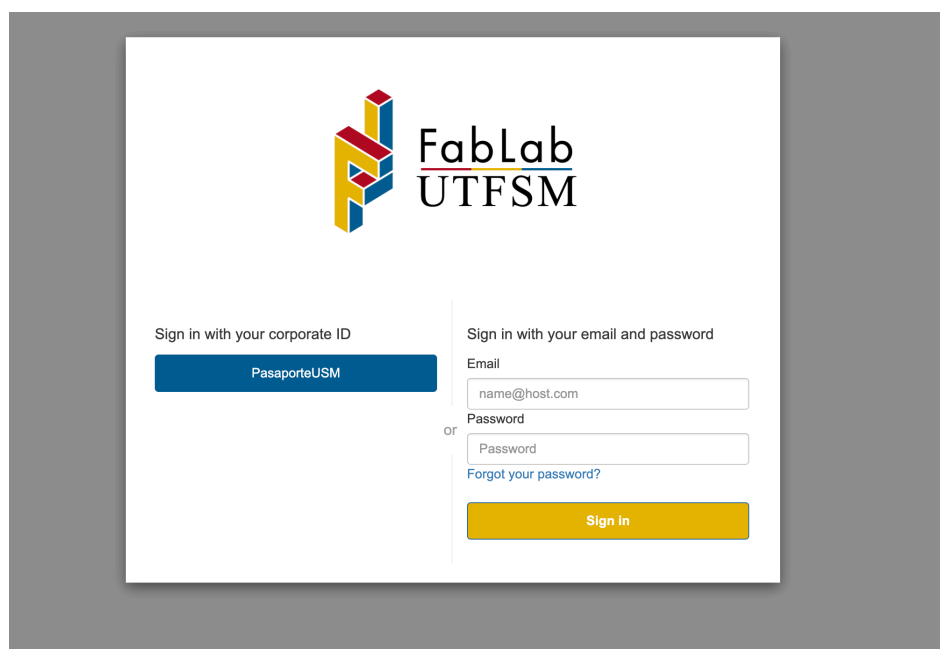


Figura 3: Sistema de autenticación y autorización del FabLab

¹⁵<https://www.classroom.fablab.usm.cl/>

¹⁶<https://auth.fablab.usm.cl/>

3.6.1. Autenticación

El proceso de autenticación del FabLab está gestionado a través de *Amazon Cognito*, el cual está configurado para integrarse con el sistema de autenticación de la Universidad Técnica Federico Santa María, conocido como “PasaporteUSM”. Esta configuración fue realizada por el autor en 2022, con el soporte del equipo de TI de la universidad. Es importante señalar que “PasaporteUSM” utiliza *Active Directory* de *Microsoft Azure*, un sistema de autenticación y autorización. El protocolo de comunicación entre *Amazon Cognito* y *Active Directory* es *OIDC*¹⁷.

El proceso de autenticación para los usuarios del FabLab y del “PasaporteUSM” sigue los siguientes pasos:

1. **Inicio de Sesión:** Cuando el usuario accede al portal, debe iniciar el proceso de autenticación si no está previamente autenticado. Se le presenta el sitio web¹⁸ de autenticación del FabLab.
2. **Selección del Método de Autenticación:** El sistema de autenticación ofrece dos opciones al usuario:
 - **Pasaporte USM:** Proceso que utiliza *Active Directory* como proveedor de identidad.
 - **Login Normal:** Autenticación tradicional gestionada directamente por *Amazon Cognito*.
3. **Autenticación con Pasaporte USM (*Active Directory*):** Si el usuario elige “PasaporteUSM”, el proceso se desarrolla de la siguiente manera:
 - a) El usuario es redirigido al formulario de autenticación de la Universidad.
 - b) El usuario ingresa sus credenciales.
 - c) *Active Directory* emite un token *OIDC*, que se envía a *Amazon Cognito*.
4. **Autenticación con Login Normal:** Si el usuario selecciona el Login Normal, *Amazon Cognito* se encarga de verificar las credenciales ingresadas.
5. **Verificación del Token:** Una vez que *Amazon Cognito* recibe el token (ya sea emitido por el “PasaporteUSM” o generado a través del login normal), verifica su validez.
6. **Acceso a la Aplicación:** Si el token es válido, se concede acceso al usuario para ingresar al portal.

¹⁷OpenID Connect

¹⁸<https://auth.fablab.usm.cl>

Este flujo se muestra a través de un diagrama de actividad en la Figura 4.

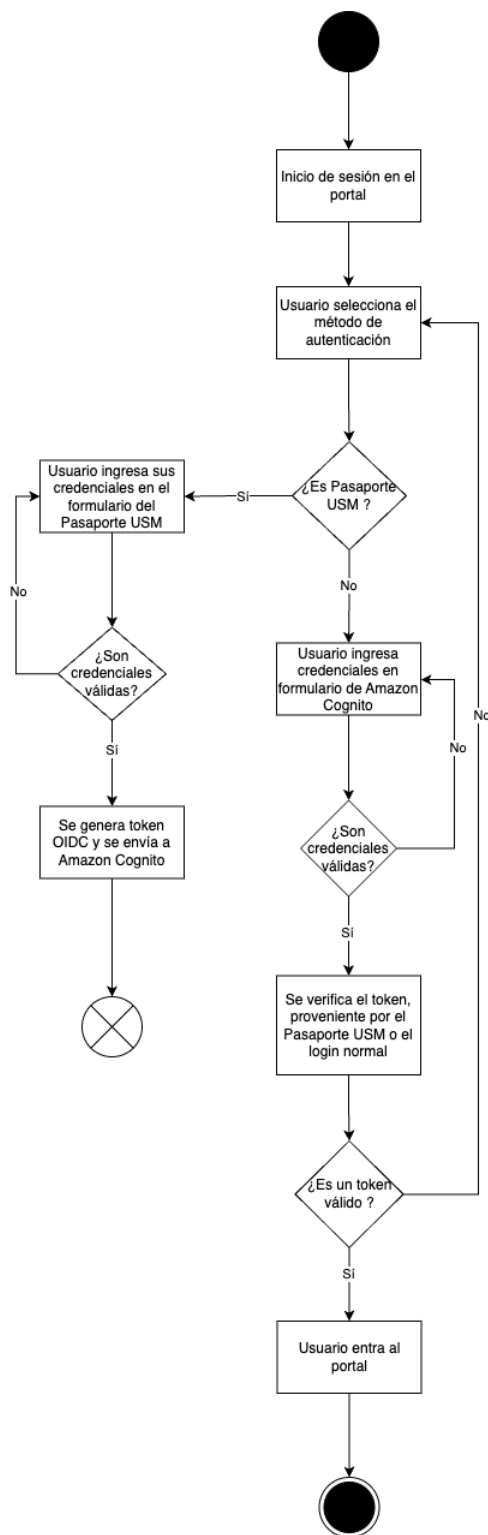


Figura 4: Flujo de autenticación del Portal

3.6.2. Autorización

La autorización para acceder a los distintos recursos de la API y las vistas del portal está gestionada por *Amazon Cognito*. Para ello, se crearon diferentes “Grupo” en el portal de *Amazon Cognito*, lo que permite asignar roles a los usuarios del sistema. Por ejemplo:

- Los usuarios que forman parte del equipo del FabLab pertenecen al grupo “fablab”.
- Los administradores del FabLab pertenecen al grupo “admin”.
- Los miembros de la comunidad universitaria que utilizan “PasaporteUSM” están en el grupo “pasaporteUSM”.

Este enfoque permite otorgar permisos de manera granular a los diferentes recursos del sistema, según el grupo al que pertenezca cada usuario. Por ejemplo, un usuario del grupo “pasaporteUSM” que no pertenezca a los grupos “fablab” o “admin” solo podrá crear proyectos o hacer reservas. En cambio, un usuario del grupo “fablab” podrá gestionar las máquinas o crear su calendario de trabajo, ya que pertenece al personal del laboratorio.

3.7. Comunicación entre el Front End y Back End

La comunicación entre el *Front End* y el *Back End* se realiza a través de una API basada en *GraphQL*. Esto permite que el *Front End* solicite únicamente los recursos que necesita, optimizando así el intercambio de información.

Para garantizar la seguridad, la API requiere que todas las solicitudes incluyan un encabezado (*header*) con un valor *x-api-key*, lo que asegura la autenticidad de la petición. Además, cada solicitud debe contener un *id_token* del usuario, el cual es validado por *Amazon Cognito*. Si el token es válido, la API consulta los recursos necesarios en la base de datos *PostgreSQL* y devuelve los resultados al *Front End*. En caso contrario, se genera un error de autenticación y se notifica al usuario.

Se puede ver con más detalle de lo mencionado en la Figura 5.

3.8. Cloud Computing

El sistema completo está diseñado y alojado en *Amazon Web Services (AWS)*, utilizando una variedad de sus servicios. A continuación, se detallan algunos de los procesos y servicios empleados en la construcción del Portal del FabLab.

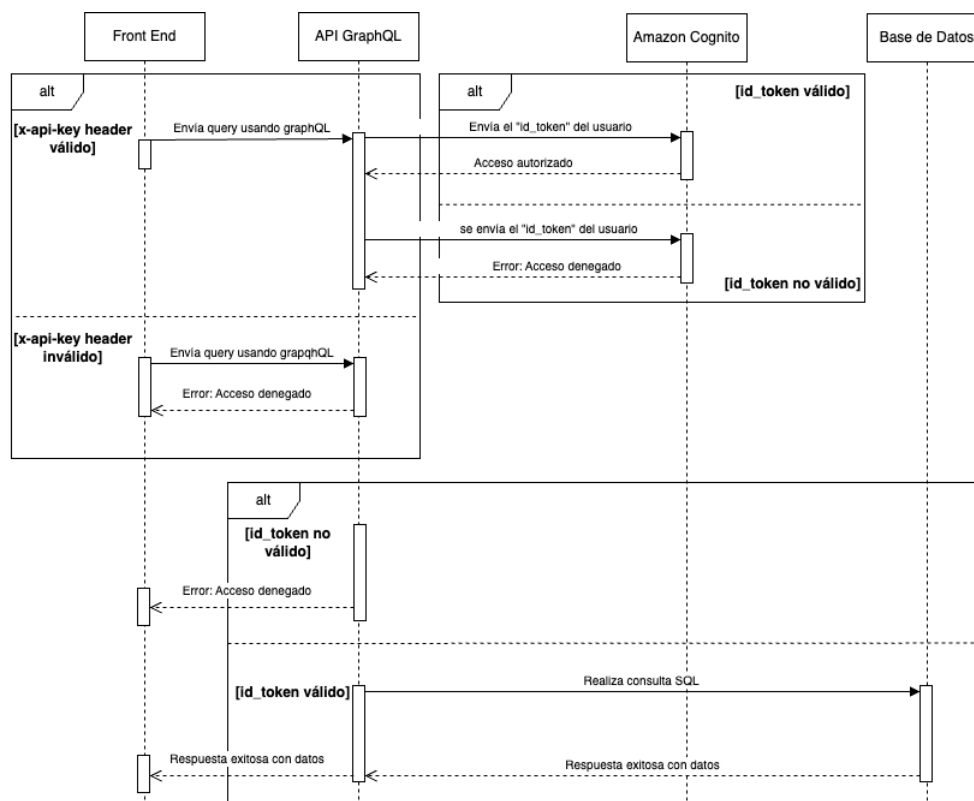


Figura 5: Comunicación entre el Front End y Back End.

3.8.1. Proceso de construcción y despliegue

El proceso de construcción y despliegue, se implementa utilizando varios servicios de AWS, entre ellos: *CodeBuild*, *CodeDeploy*, *CodePipeline*, *IAM* y *ECR*. Tanto el *Frontend* como del *Backend* siguen un flujo similar, diferenciándose únicamente en los roles y políticas definidos en *IAM*. El flujo completo de despliegue se desarrolla a través de los siguientes pasos:

1. El proceso de *CodePipeline* se activa mediante un *webhook* generado por *GitHub* cuando detecta un cambio en el repositorio en ciertas ramas de desarrollo, normalmente son "development" o "master", iniciando el pipeline de construcción y despliegue.
2. *CodeBuild* comienza la construcción de la imagen del contenedor especificada en el *Dockerfile* del proyecto.
3. La imagen construida se almacena en el servicio *Elastic Container Registry (ECR)*, manteniendo un historial de las versiones de las imágenes.
4. Una vez que la aplicación ha sido construida exitosamente, *CodeDeploy* toma el control y despliega la aplicación en *Elastic Beanstalk*.

3.8.2. Alojamiento del Frontend y Backend

Tras la construcción y despliegue a través de *CodePipeline*, la aplicación se aloja en *Elastic Beanstalk*. En este entorno, tanto el *Frontend* como el *Backend* están dockerizados en contenedores y desplegados en instancias *EC2*. El sistema incluye un balanceador de carga y auto-escalado automático, lo que permite la creación de nuevas instancias conforme aumenta la demanda de red (para el *Frontend* y *Backend* por separado), basándose en métricas predefinidas. Asimismo, el balanceador de carga distribuye el tráfico entre las distintas instancias, asegurando la escalabilidad y resiliencia del sistema.

3.8.3. Envío de emails

El sistema utiliza el servicio *Simple Email Service (SES)* de *Amazon Web Services (AWS)* para gestionar el envío de correos electrónicos. Este servicio se encarga de enviar correos relacionados con la autenticación de usuarios, como la verificación de identidad y el restablecimiento de contraseñas. Además, se envían notificaciones a los usuarios registrados en el Portal del FabLab, informándoles sobre la creación y el estado de sus reservas de máquinas en el laboratorio.

3.8.4. Almacenamiento y disponibilidad de Archivos

El almacenamiento de archivos, como imágenes, PDF y videos, se gestiona a través de *S3 buckets* de *Amazon Web Services*. Estos archivos son almacenados y cacheados mediante el servicio de red de entrega de contenido (*Content Delivery Network*) de Amazon, *CloudFront*, lo que garantiza alta disponibilidad, seguridad y escalabilidad para los usuarios.

3.8.5. Base de Datos

Para la gestión de los datos se utilizó una base de datos relacional basada en *PostgreSQL*, implementada mediante el servicio *Relational Database Service (RDS)* de *Amazon Web Services*. Este servicio facilita la creación, administración y escalabilidad de bases de datos relacionales, optimizando el rendimiento y la disponibilidad.

3.9. Diseño de la Interfaz

La interfaz del Portal FabLab se destaca por un diseño visual limpio y moderno. Desde el primer acceso al portal, se puede observar que el tema elegido es claro y minimalista, lo cual se aprecia en la Figura 6.

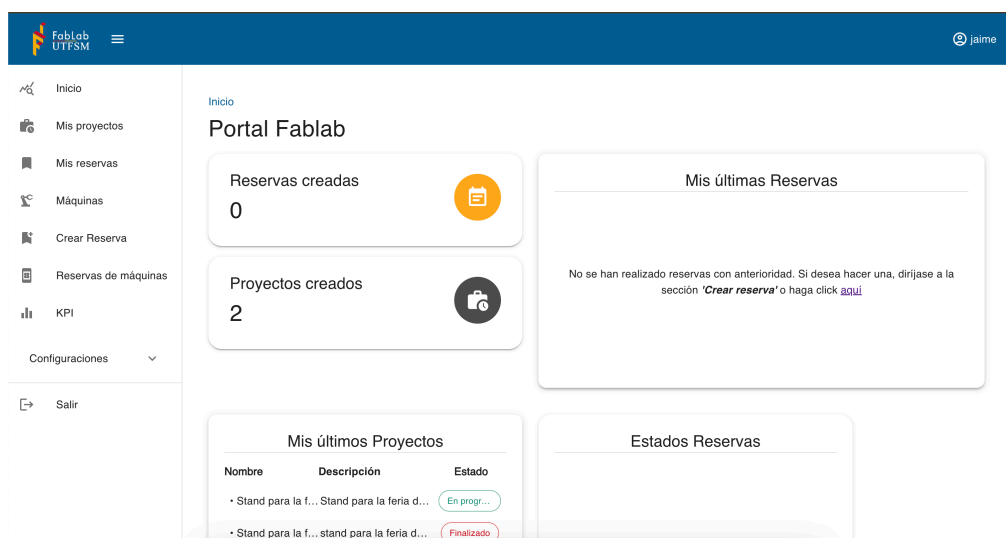


Figura 6: Portal FabLab Inicio

3.9.1. Colores

Los colores seleccionados para el portal son consistentes con la identidad visual del FabLab, utilizando los mismos tonos presentes en el logotipo, como se muestra en la Figura 7.

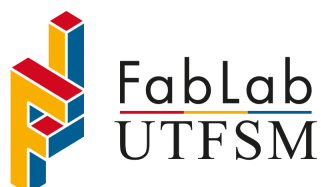


Figura 7: Logo FabLab

Se han definido cuatro colores principales que rigen el diseño del portal, cada uno con un propósito específico dentro de la interfaz:

- **Color primario:** predomina en los encabezados y botones principales del sitio, su valor hexadecimal es #005E90.
- **Color secundario:** utilizado para resaltar elementos interactivos o complementarios, su valor hexadecimal es #FCB100.
- **Color para errores:** asignado a mensajes de advertencia o error, proporcionando una clara indicación visual, su valor hexadecimal es #D1081E.

- **Color de uso general:** aplicado a textos secundarios y detalles menos prominentes, su valor hexadecimal es #808080.

La representación visual de estos colores se encuentra en la Figura 8, donde se puede observar cómo se combinan para ofrecer una experiencia visual coherente y atractiva.



Figura 8: Colores utilizados en el Portal del FabLab. Imagen obtenida de: <https://paletadecolores.online/crear-paleta/>.

3.9.2. Tipografía

La tipografía seleccionada para el Portal FabLab está basada en una pila de fuentes que asegura una visualización coherente en distintos dispositivos y navegadores. La fuente principal utilizada es *Roboto*, una tipografía moderna y ampliamente usada en interfaces web. Esta fuente se carga desde el repositorio de *Google Fonts*. Si por alguna razón la fuente *Roboto* no está disponible, el navegador recurrirá a usar *Helvetica* o *Arial*.

Este enfoque asegura que el texto del portal sea siempre legible y coherente por los usuarios.

3.9.3. Diseño Responsivo

El portal del FabLab cuenta con un diseño responsivo, lo que permite que los elementos visuales se adapten de manera óptima a diferentes tamaños de pantalla, ya sea en computadores, tabletas o dispositivos móviles. Este enfoque asegura una experiencia de usuario fluida e intuitiva, independientemente del dispositivo utilizado.

En las siguientes figuras, se puede apreciar cómo el sistema ajusta su diseño para ofrecer una visualización adecuada en cada tipo de dispositivo. La Figura 9 muestra el diseño en una

pantalla de computador, mientras que en las Figuras 10 y 11, se observa la adaptación para tabletas y móviles, respectivamente.

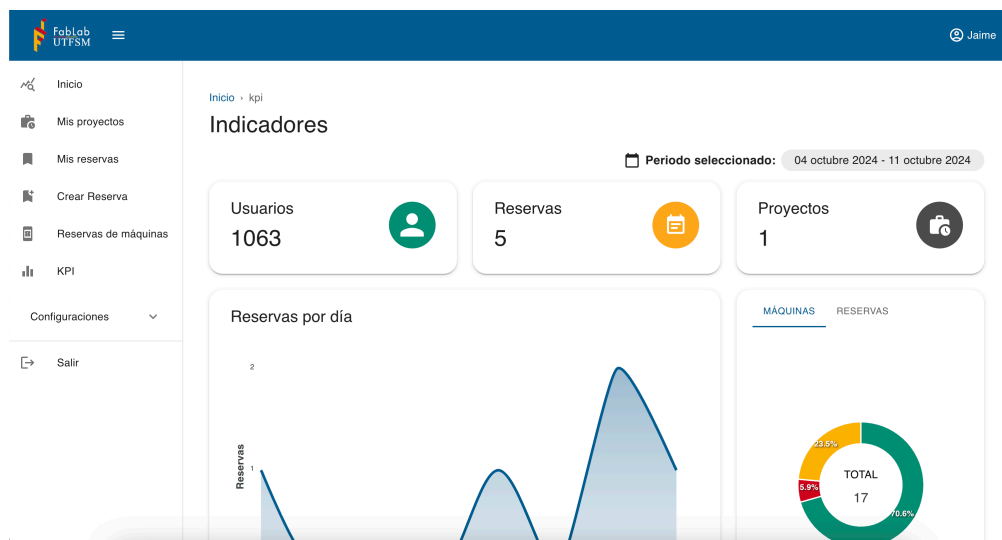


Figura 9: Vista en Computadora

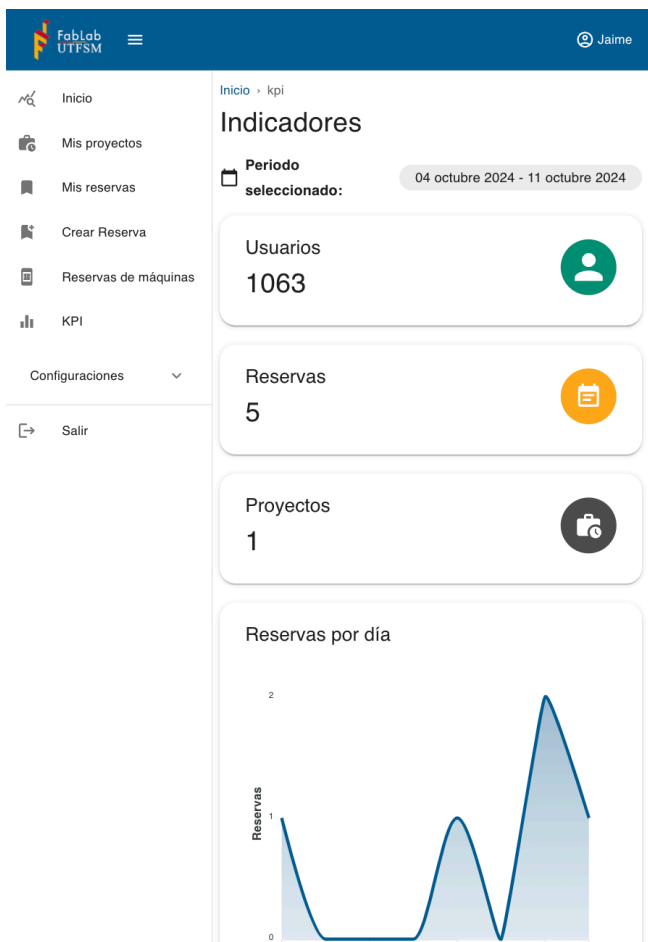


Figura 10: Vista en Tablet



Figura 11: Vista en Móvil

3.10. Gestión de las máquinas

El proceso de gestión de máquinas puede ser realizado por actores con los roles de **Personal del FabLab** o **Administradores del FabLab**. Existen dos entidades principales: el tipo de máquina (*Machine Type*) y las máquinas (*Machine*). La relación entre estas entidades es de uno a muchos; es decir, una máquina solo puede estar asociada a un tipo de máquina, mientras que un tipo de máquina puede estar vinculado a **n** máquinas.

El personal y los administradores tienen la facultad de gestionar los tipos de máquinas y las máquinas específicas, permitiéndoles crear, actualizar, buscar o realizar una eliminación lógica según sea necesario.

En la Figura 12 se muestran los tipos de máquinas del laboratorio FabLab. La interfaz incluye un buscador y un botón para ordenar los elementos encontrados. Cada tipo de máquina cuenta con dos opciones: editar el tipo de máquina o ver todas las máquinas asociadas. Al se-

leccionar la opción de editar el tipo de máquina, se accede a un formulario, como se muestra en la Figura 13, donde se solicita ingresar el nombre, la descripción y un campo para adjuntar archivos. El formulario para crear un nuevo tipo de máquina contiene los mismos campos que el formulario de edición.

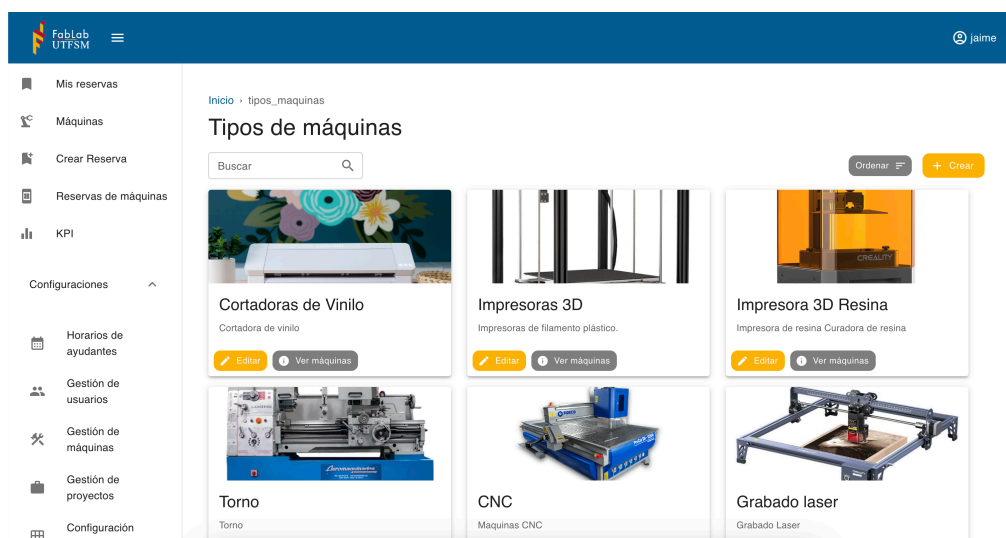


Figura 12: Tipos de Máquinas - Portal FabLab

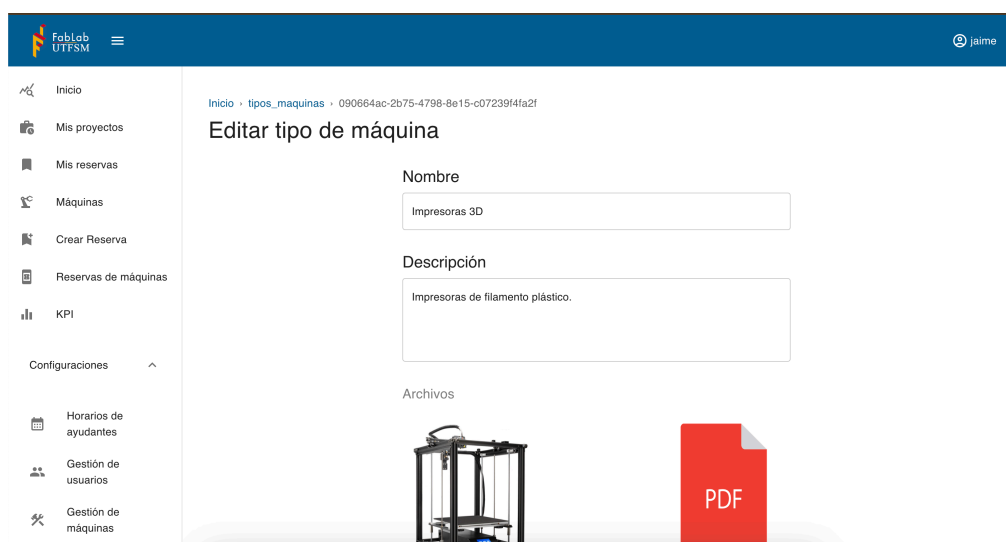


Figura 13: Editar Tipo de Máquina - Portal FabLab

Por otro lado, al seleccionar la opción “Ver máquinas” de un tipo de máquina (ver Figura 12), se despliega una tabla con todas las máquinas correspondientes a ese tipo. En esta tabla, como se muestra en la Figura 14, se puede buscar una máquina específica por su nombre o ID utilizando el buscador. Además, se presentan datos como el nombre de la máquina, su estado, el tipo de máquina y una opción para editarla.

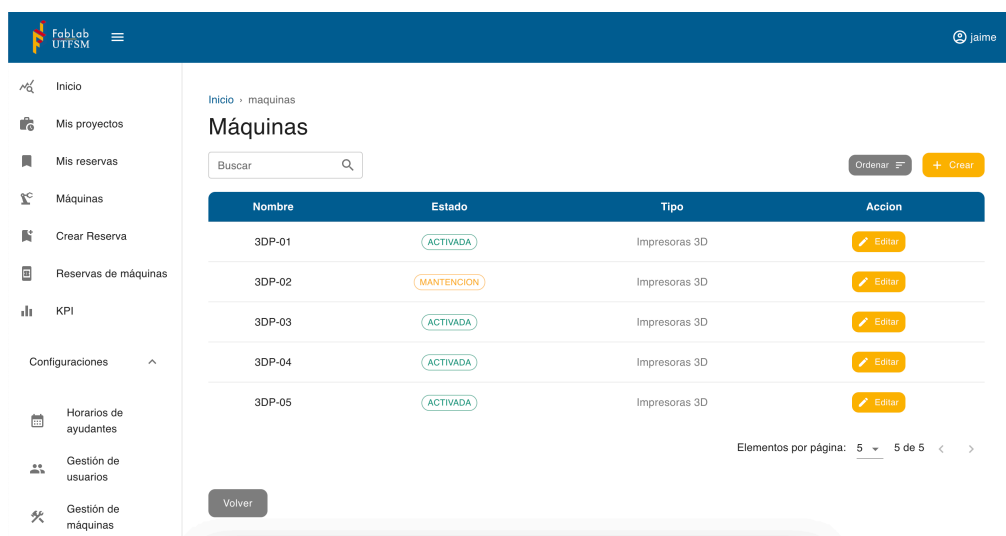


Figura 14: Máquinas - Portal FabLab

La vista para editar una máquina se presenta en la Figura 15. En esta vista, se pueden modificar campos como el nombre, la descripción, y la cantidad mínima y máxima de bloques (que corresponde a la cantidad de bloques que se pueden reservar para esa máquina). También es necesario seleccionar el tipo de máquina, su estado (Activada, Desactivada, Mantenimiento), y se permite subir archivos multimedia. El formulario para crear una nueva máquina incluye los mismos campos que el formulario de edición.

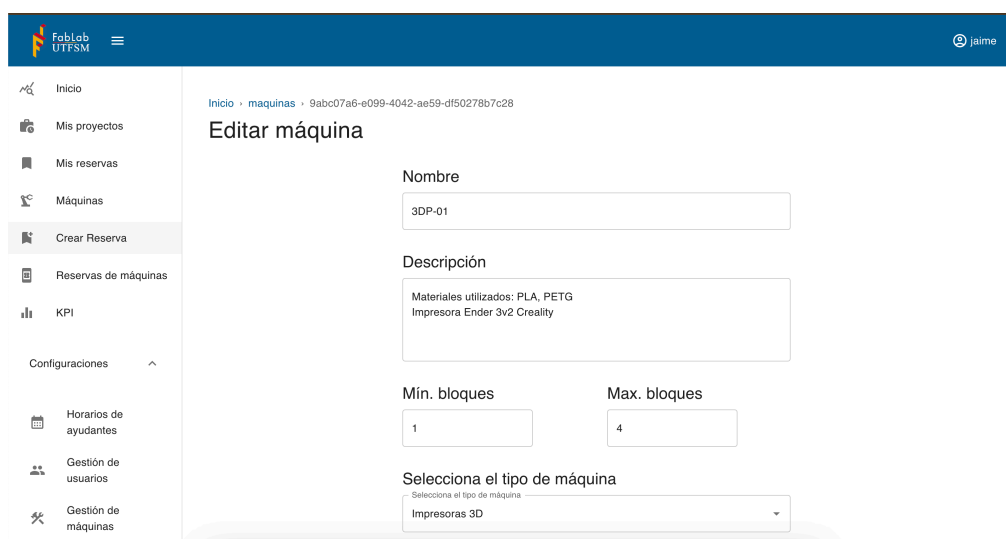


Figura 15: Editar Máquina - Portal FabLab

Finalmente, existe una vista disponible para todos los usuarios, sin importar su rol, que permite visualizar los diferentes tipos de máquinas que posee el laboratorio. Esta vista se muestra en la Figura 16, donde aparece la opción de "Ver más". Al seleccionar esta opción, se

despliega el detalle completo de cada tipo de máquina del laboratorio, como se observa en la Figura 17.

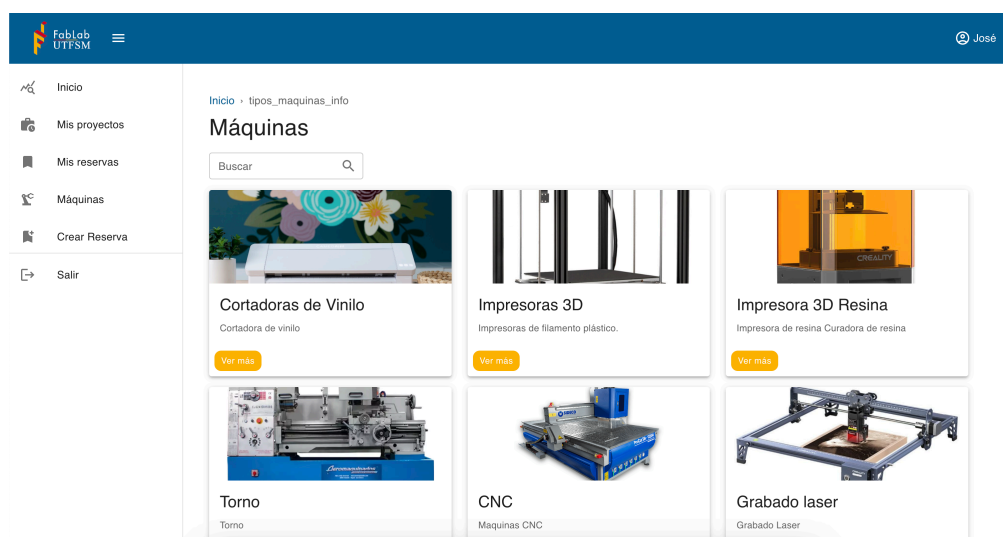


Figura 16: Tipo de máquina vista general - Portal FabLab

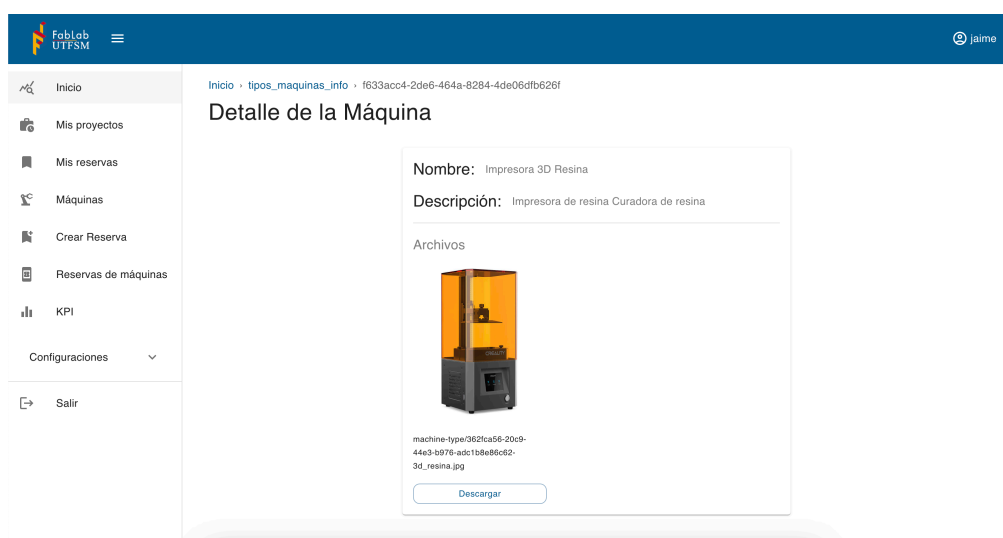


Figura 17: Detalle tipo de máquina - Portal FabLab

3.11. Gestión de los proyectos

Los proyectos constituyen otro pilar fundamental dentro del portal FabLab, ya que proporcionan información clave para entender el propósito de uso del laboratorio. Esto se logra debido a que cada reserva está asociada a un proyecto. Existen dos entidades principales para gestionar los proyectos: el tipo de proyectos (*Project Type*) y el proyecto (*Project*). La

relación entre estas entidades es de uno a muchos; es decir, un proyecto solo puede estar asociado a un tipo de proyecto, mientras que un tipo de proyecto puede estar vinculado a **n** proyectos.

Los usuarios con los roles de **Personal del FabLab** o **Administradores del FabLab** tienen la capacidad de crear, editar o realizar una eliminación lógica de cualquier proyecto o tipo de proyecto. En cambio, los usuarios que no poseen estos roles solo pueden crear, editar, finalizar o realizar una eliminación lógica de sus propios proyectos.

En la Figura 18, se presentan los tipos de proyectos. Esta vista es accesible únicamente para los usuarios con los privilegios mencionados. Al editar un tipo de proyecto, se mostrará un formulario, como se observa en la Figura 19, donde se deben proporcionar el nombre, una descripción, archivos adjuntos y un valor para determinar si el tipo de proyecto es global o no. Si un tipo de proyecto se marca como global, cualquier usuario podrá crear una reserva asociada a un proyecto que este asociado a ese tipo de proyecto, independientemente de si fue él quien lo creó. Esto es especialmente útil para proyectos como “Uso personal”, dado que muchos usuarios del FabLab utilizan el laboratorio con este propósito, evitando así que cada usuario deba crear su propio proyecto de “Uso personal” cada vez que quiera hacer una reserva.

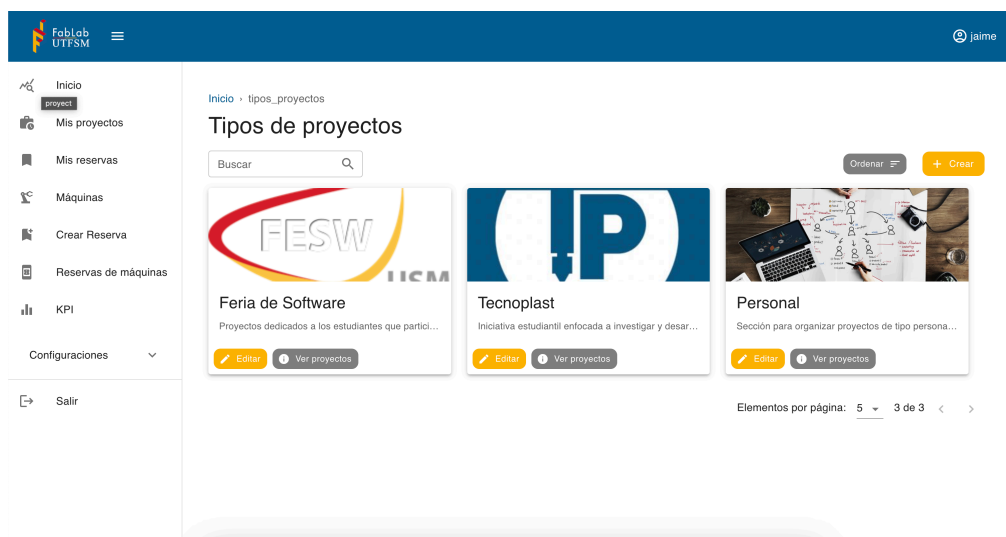


Figura 18: Tipos de proyectos - Portal FabLab

Editar tipo de proyecto

Nombre: Feria de Software

Descripción: Proyectos dedicados a los estudiantes que participan de la Feria de Software de la Universidad Técnica Federico Santa María

Es de tipo global: ☒ Global

Archivos:

Figura 19: Editar tipo de proyecto - Portal FabLab

Asimismo, la vista de los proyectos creados se muestra en la Figura 20. Al editar o crear un proyecto, se despliega un formulario que se puede apreciar en la Figura 21 donde es necesario ingresar el nombre, descripción, enlace al repositorio, archivos adjuntos, seleccionar el tipo de proyecto, y definir si es un proyecto individual o grupal. Si el proyecto es grupal, se deberá seleccionar a los usuarios que participarán en él. El usuario puede eliminar, actualizar o finalizar el proyecto. En caso de finalizarlo, el sistema permite dar un *feedback* sobre el proyecto desarrollado en el laboratorio.

Proyectos de Feria de Software

Buscar:

Ordenar: + Crear

Nombre	Estado	Tipo de proyecto	Fecha de creación	Acción
Stan de la Feria de Software	Finalizado	Feria de Software	08/09/2024 20:39	Editar
Ingeniería de software	En progreso	Feria de Software	13/10/2024 09:50	Editar
Stand para la feria de software	Finalizado	Feria de Software	13/10/2024 16:10	Editar
Stand para la feria de software s2-2024	En progreso	Feria de Software	13/10/2024 18:44	Editar

Elementos por página: 5 4 de 4

Volver

Figura 20: Proyectos - Portal FabLab

The screenshot shows the 'Editar proyecto' (Edit project) interface in the FabLab UTFSM portal. The left sidebar contains navigation links: Inicio, Mis proyectos, Mis reservas, Máquinas, Crear Reserva, Reservas de máquinas, KPI, Configuraciones, and Salir. The main content area is titled 'Editar proyecto' and includes the following fields:

- Nombre:** Stan de la Feria de Software
- Descripción:** Proyecto para el stan de la feria de software
- Link Repositorio (opcional):** (Empty text field)
- Selecciona el tipo de proyecto:** FERIA de Software

A summary box on the right displays the following information:

- Estado:** Finalizado
- Nombre:** Jaime
- Email:** jaime.rodriguez.gacitua@gmail.com
- Fecha de comienzo:** 08/09/2024 20:39
- Fecha de actualización:** 13/10/2024 09:52
- Fecha de término:** 13/10/2024 06:52

Figura 21: Editar proyecto - Portal FabLab

3.12. Gestión de los bloques

Los bloques de horarios son un elemento clave tanto para la creación de reservas como para la organización de los horarios de los ayudantes del FabLab. Para generar una reserva, es necesario seleccionar uno o más bloques de tiempo, mientras que para establecer un horario, se deben elegir los bloques en los que se realizarán las labores dentro del laboratorio. Cabe destacar que solo los usuarios con el rol de **“Administradores del FabLab”** tienen la capacidad de gestionar los bloques.

Cada bloque está compuesto por un nombre, una hora de inicio, una hora de término y el día al que pertenece. En la Figura 22 se presenta una tabla con los bloques creados, y en la Figura 23 se puede observar el formulario utilizado para crear o editar un bloque.

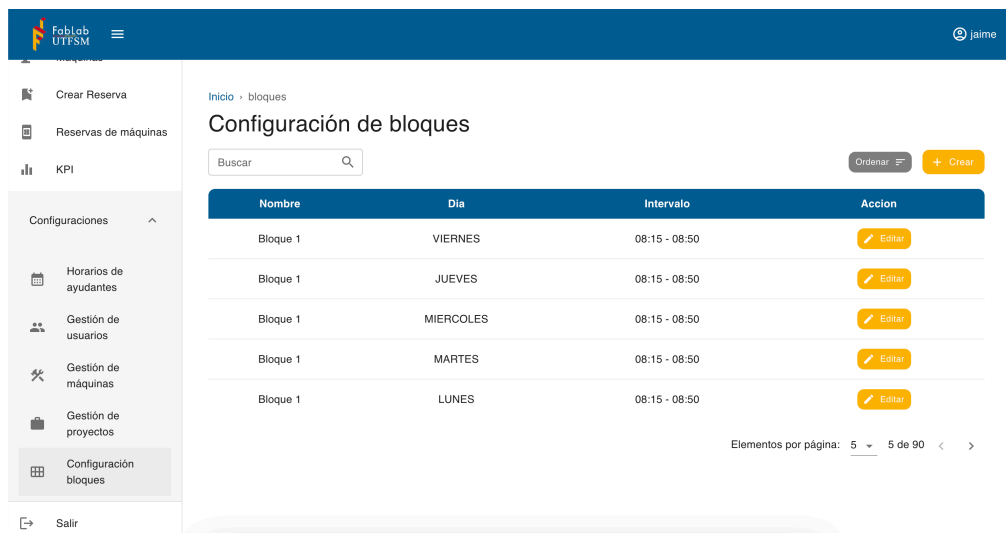


Figura 22: Bloques - Portal FabLab

Es importante resaltar que los bloques creados coinciden con los utilizados por la Universidad Técnica Federico Santa María en su campus de San Joaquín. Esto facilita tanto al personal de la universidad como al personal del laboratorio, ya que permite una mejor organización y gestión del uso del laboratorio en consonancia con las actividades cotidianas de la universidad.

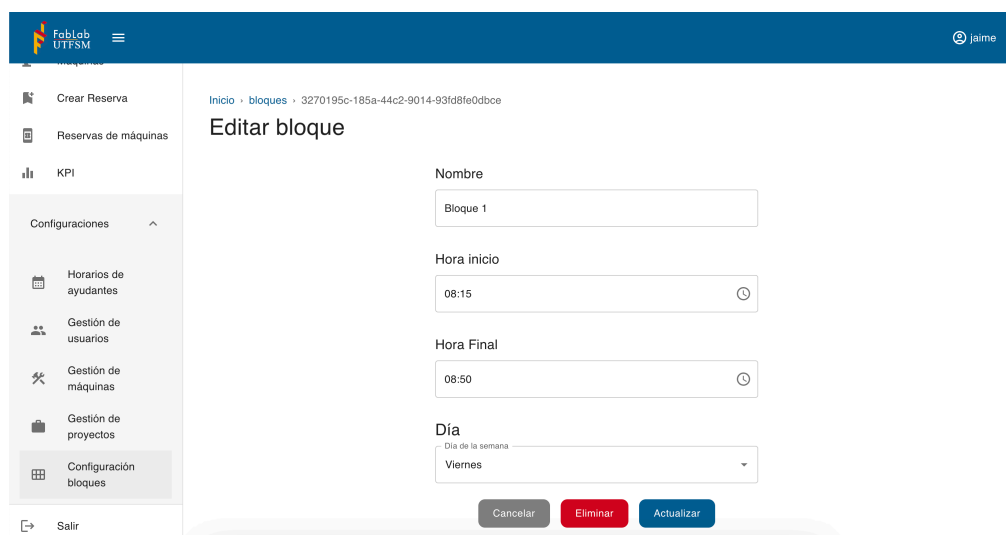


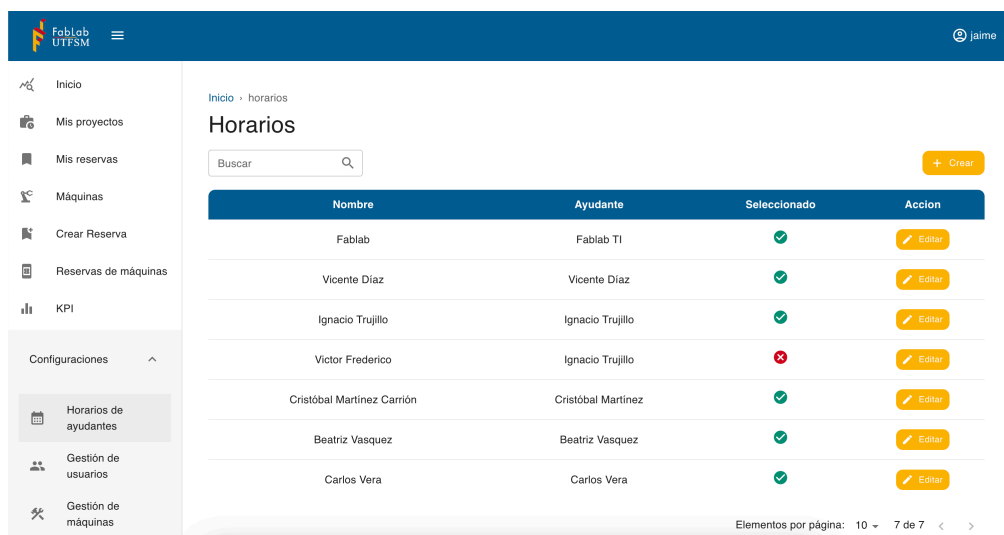
Figura 23: Editar Bloque - Portal FabLab

3.13. Gestión de los horarios

Los horarios son otra pieza fundamental en el Portal FabLab. No solo ayudando a la organización propia del laboratorio, si no también para la creación de las reservas. Esto debido a que para poder reserva una máquina del laboratorio, debe haber un ayudante disponible en el bloque deseado. Es importante mencionar que solo los actores con roles de **Personal del FabLab** o **Administradores del FabLab** pueden gestionar los horarios. Pero en este punto hay que detenerse. Ya que, un administrador puede editar, realizar una eliminación lógica o crear horarios de cualquier ayudante o administrador del FabLab. Mientras que un ayudante solo puede editar, crear o realizar la eliminación lógica de un horario que le pertenezca.

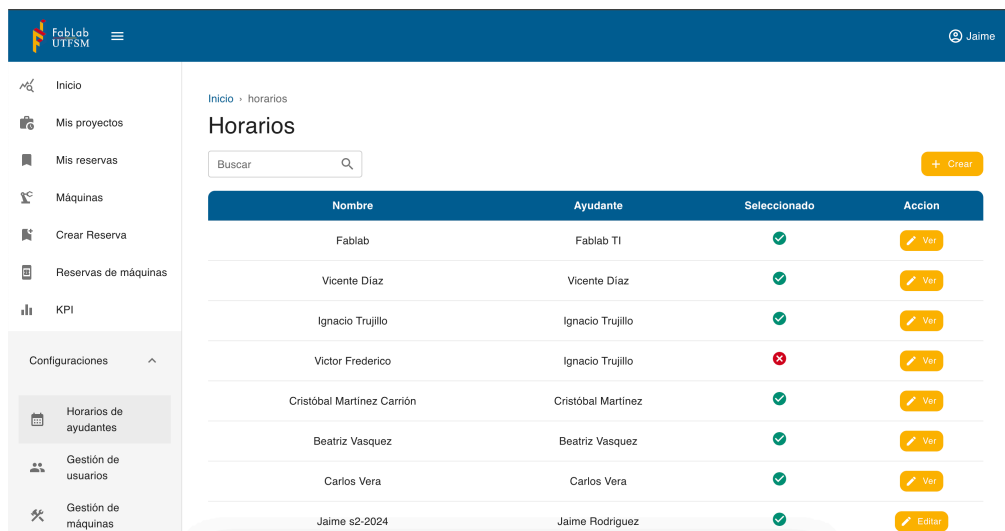
En la Figura 24 se muestra una tabla con los horarios de los ayudantes. Es importante destacar que en esta figura, el usuario que visualiza la tabla es un administrador, lo que le permite editar cualquier horario, ya sea propio o de otros usuarios.

Por otro lado, en la Figura 25, se ilustra la vista para un usuario con el rol de **Personal del FabLab**. En esta vista, dicho usuario solo tiene la opción de “Ver” los horarios que no le pertenecen, mientras que para sus propios horarios, aparece la opción de “Editar”.



Nombre	Ayudante	Seleccionado	Accion
Fablab	Fablab T1	✓	Editar
Vicente Díaz	Vicente Díaz	✓	Editar
Ignacio Trujillo	Ignacio Trujillo	✓	Editar
Victor Frederico	Ignacio Trujillo	✗	Editar
Cristóbal Martínez Carrión	Cristóbal Martínez	✓	Editar
Beatriz Vasquez	Beatriz Vasquez	✓	Editar
Carlos Vera	Carlos Vera	✓	Editar

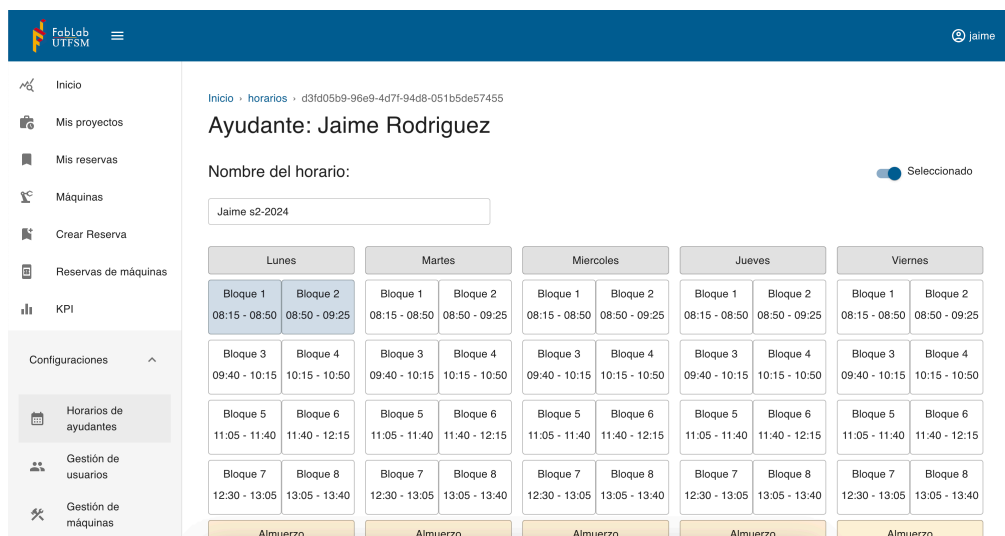
Figura 24: Horarios - Portal FabLab



Nombre	Ayudante	Seleccionado	Accion
Fablab	Fablab TI	✓	Ver
Vicente Diaz	Vicente Diaz	✓	Ver
Ignacio Trujillo	Ignacio Trujillo	✓	Ver
Victor Frederico	Ignacio Trujillo	✗	Ver
Cristóbal Martinez Carrión	Cristóbal Martinez	✓	Ver
Beatriz Vasquez	Beatriz Vasquez	✓	Ver
Carlos Vera	Carlos Vera	✓	Ver
Jaime s2-2024	Jaime Rodriguez	✓	Editar

Figura 25: Horarios vista ayudante - Portal FabLab

En la Figura 26 se puede la interfaz para poder editar o ver un horario. En caso que puedas editar, los botones estarán habilitados para ello. Caso contrario, que solo tengas permisos para ver, lo botones no te permitirán realizar un cambio, esto se puede ver en la Figura 27, donde el usuario no tiene permisos para la edición del horario.



Lunes		Martes		Miercoles		Jueves		Viernes	
Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25
Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50
Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15
Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40
Almuerzo		Almuerzo		Almuerzo		Almuerzo		Almuerzo	

Figura 26: Editar Horario - Portal FabLab



Figura 27: Vista Horario - Portal FabLab

3.14. Gestión de Usuarios

El portal FabLab no permite el registro automático de nuevos usuarios a través de su interfaz. La única manera de añadir un usuario nuevo es mediante la API del sistema o utilizando *Amazon Cognito*. En el caso de los usuarios provenientes de la Universidad Técnica Federico Santa María, es importante destacar el proceso de incorporación, ya que ni *Amazon Cognito* ni el sistema conocen de antemano a los usuarios que nunca han accedido al Portal FabLab a través del “Pasaporte USM”.

Cuando un usuario con credenciales del “Pasaporte USM” ingresa por primera vez al Portal FabLab y completa el proceso de autenticación (ver Figura 4), *Amazon Cognito* lo registra en su base de datos. Esto desencadena un *trigger* que activa una función *lambda* de AWS, encargada de crear el usuario en el sistema y almacenarlo en la base de datos *PostgreSQL*.

En caso de que el flujo de registro falle, se dispone de un botón para sincronizar los usuarios entre *Amazon Cognito* y el sistema, como se muestra en la Figura 28.

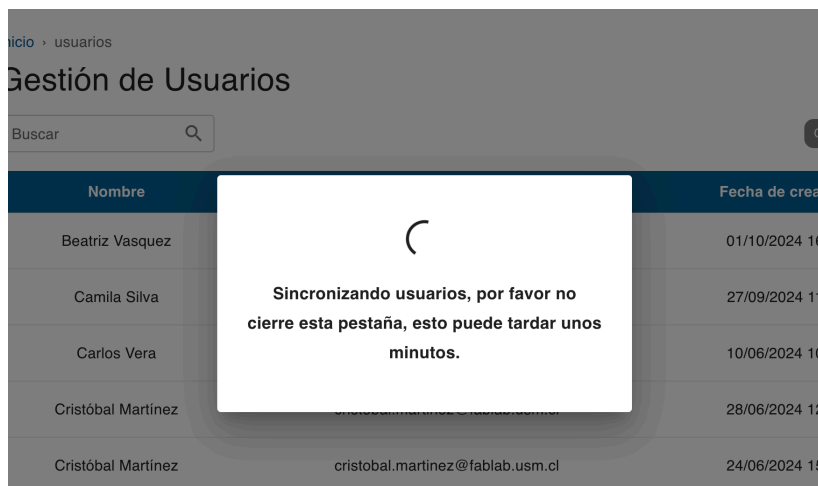


Figura 28: Sincronización de usuarios - Portal FabLab

La Figura 29 muestra los usuarios registrados en el sistema. Esta vista es accesible solo para los roles de **Personal del FabLab** y **Administradores del FabLab**.

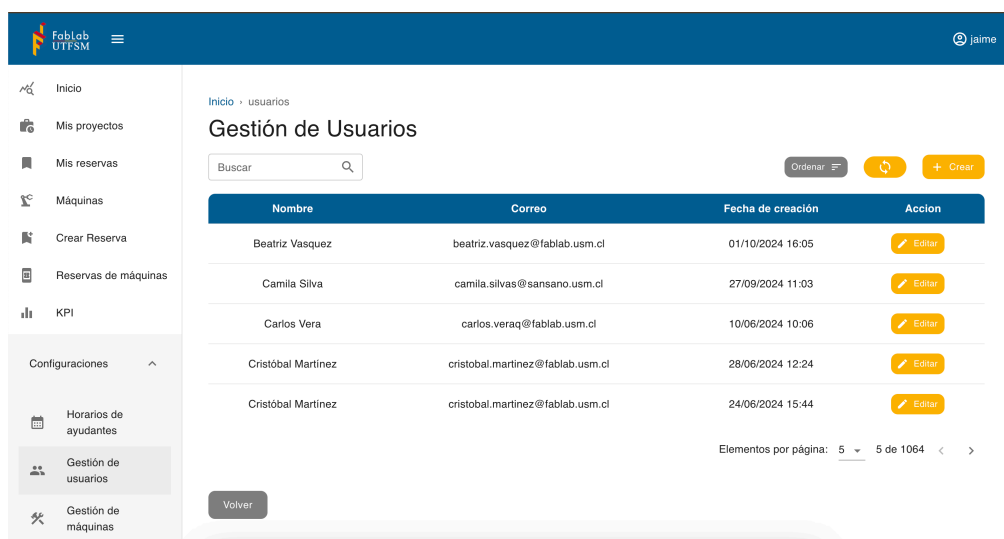


Figura 29: Usuarios - Portal FabLab

El formulario para editar un usuario se muestra en la Figura 30. En este formulario se deben completar campos como el nombre, segundo nombre, apellido, segundo apellido, fecha de nacimiento, tipo de identificación (RUT o pasaporte), número de documento, correo electrónico, carrera la que cursa, año de ingreso y sexo

Solo los usuarios con el rol de administrador tienen la capacidad de eliminar un usuario. Esta acción elimina al usuario tanto del sistema como de *Amazon Cognito*. Por otro lado, los usuarios con el rol de ayudante solo pueden editar información básica, exceptuando el correo electrónico, que no es editable. Finalmente, cualquier usuario nuevo que acceda por

primera vez al sistema deberá completar este formulario para poder utilizar el Portal FabLab, siendo este un requisito obligatorio para todos los usuarios.

Figura 30: Editar usuario - Portal FabLab

3.15. Reservas

Las reservas constituyen una de las funciones principales del Portal FabLab, permitiendo a los usuarios gestionar el uso de las máquinas del laboratorio en bloques de tiempo predefinidos (ver Figura 22). Esta funcionalidad no solo facilita la organización del uso de las máquinas, sino que también permite llevar un registro en tiempo real, aligerando la carga de trabajo de los ayudantes en la gestión de los equipos.

Cada reserva puede tener tres estados: **Creada**, **Anulada** y **Confirmada**. Al crear una reserva, esta se inicia en el estado “Creada”, lo que indica que el usuario ha solicitado el uso de la máquina, pero aún no ha comenzado a utilizarla. Un ayudante o administrador puede cambiar el estado de la reserva a “Confirmada” cuando el usuario ha sido habilitado para usar la máquina y ha comenzado su sesión en el bloque correspondiente. Por otro lado, si el usuario no se presenta a la hora reservada, el ayudante o administrador puede marcar la reserva como “Anulada”, liberando así los bloques de tiempo para que otros usuarios puedan reservar.

Adicionalmente, si un ayudante o administrador considera que el usuario no está habilitado o capacitado para utilizar la máquina, también podrá anular la reserva. Por su parte, el usuario tiene la opción de anular su propia reserva, siempre y cuando lo haga antes de la fecha y hora establecida.

En caso de que el usuario desee modificar una reserva, como cambiar la máquina, el proyecto

asociado o los bloques de tiempo, dicha modificación debe ser gestionada directamente por un ayudante o administrador.

3.16. Indicadores Clave de Desempeño

Los Indicadores Clave de Desempeño (KPI's) son un aspecto fundamental del sistema. Para su gestión, se optó por utilizar el software *Power BI*, debido a su interfaz intuitiva, que permite a los usuarios crear o editar los indicadores existentes. Además, la Universidad Técnica Federico Santa María ya utiliza este software, lo que garantiza su integración en un ecosistema conocido.

A través de una licencia *Power BI Pro*, asociada al correo principal del laboratorio (ti@fablab.usm.cl), se creó un reporte que se muestra en la Figura 31. Este reporte se conectó a la base de datos *PostgreSQL* alojada en el servicio *RDS* de *Amazon Web Services*, lo que garantiza que la información visualizada se actualice en tiempo real, permitiendo decisiones informadas y basadas en datos.

La Figura 31 ilustra una vista completa de los KPI definidos, organizados en gráficos interactivos que permiten analizar información detallada sobre reservas, estudiantes, proyectos y utilización de máquinas. Estos indicadores son esenciales no solo para medir el rendimiento actual, sino también para identificar patrones, prever tendencias y optimizar los procesos clave del laboratorio.

Los indicadores clave que se pueden observar en la Figura 31 son los siguientes:

- Número total de reservas realizadas.
- Total de reservas realizadas por mes.
- Número total de estudiantes agrupados por carrera.
- Número total de estudiantes clasificados por género.
- Total de proyectos asociados a cada máquina reservada.
- Total de proyectos clasificados según el tipo de proyecto.
- Número total de estudiantes activos por mes, es decir, aquellos que utilizaron el laboratorio.

Estos indicadores no solo permiten evaluar la eficiencia operativa del FabLab, sino que también facilitan la identificación de áreas de mejora y oportunidades para aumentar la participación y el uso de los recursos. Por ejemplo, el análisis del número de estudiantes activos



Figura 31: Reportería presentando KPI's - Portal FabLab

mensualmente proporciona una visión clara de la utilización del laboratorio y puede ser usado para diseñar estrategias que promuevan un mayor aprovechamiento de las instalaciones. Del mismo modo, los datos de proyectos por tipo y máquina pueden guiar decisiones sobre inversión en equipos y recursos futuros.

CAPÍTULO 4

VALIDACIÓN DE LA SOLUCIÓN

La validación de la solución propuesta se realizó considerando los objetivos específicos establecidos para el proyecto. A continuación, se detalla la validación de cada objetivo propuesto

4.1. Validación de la Identificación de Actores Principales

Los actores principales fueron definidos como **usuarios del FabLab**, **personal del FabLab** y **administradores del FabLab**. La validación se realizó a través de reuniones con el personal del FabLab, quienes confirmaron que sus roles y responsabilidades estaban adecuadamente representados en el sistema. Además, se verificó que el sistema satisface las necesidades específicas de cada actor a través de pruebas de usabilidad, donde los usuarios completaron tareas correspondientes a sus roles.

4.2. Validación del Análisis de Procesos y Priorización de Digitalización

Para cumplir este objetivo, se llevó a cabo un análisis de los procesos del laboratorio, tal como se detalla en la Tabla 1. Se identificaron y priorizaron los procesos clave, como:

- **Gestión de usuarios**
- **Gestión de horarios de turnos**
- **Actualización de existencias del inventario de máquinas**
- **Reserva de máquinas**
- **Ingreso y estado de proyectos**
- **Creación y análisis de KPI's**

Los demás procesos listados en la tabla, como la gestión de talleres, programas, y horarios de capacitaciones, no serán digitalizados en esta etapa debido a las prioridades establecidas, lo cual fue analizado y validado mediante reuniones con los administradores del FabLab y personal involucrado, quienes confirmaron que los procesos seleccionados reflejan las principales necesidades operativas del laboratorio.

4.3. Validación de la Plataforma Web

El desarrollo de una plataforma web intuitiva y accesible se validó mediante pruebas de usabilidad con 5 usuarios representativos de los diferentes roles involucrados en el FabLab. A continuación, se describe a los usuarios que participaron en las pruebas:

- Usuario 1: Estudiante de tercer año de Ingeniería Civil Informática, con amplia experiencia en el uso de plataformas web. Familiarizado con tareas relacionadas con reservas y proyectos.
- Usuario 2: Estudiante de Derecho de otra universidad, con experiencia moderada en plataformas web y sin familiaridad con sistemas de reservas y proyectos, representando a usuarios externos a la Universidad.
- Usuario 3: Administrador del área de TI (Tecnología de la Información) del FabLab, con alta experiencia técnica. Responsable de la gestión de máquinas, horarios, proyectos y reservas, representando al personal del FabLab.
- Usuario 4: Estudiante de quinto año de Ingeniería Civil Mecánica en la Universidad Técnica Federico Santa María. Usuario externo al FabLab con poca experiencia en el sistema, representando a usuarios ocasionales o nuevos.
- Usuario 5: Ex estudiante de Ingeniería Comercial de la Universidad Técnica Federico Santa María. Usuario externo al FabLab con poca experiencia en el sistema, representando a usuarios ocasionales o nuevos.

Cada usuario completó cinco tareas específicas, previamente definidas, diseñadas para evaluar la funcionalidad y usabilidad del sistema:

- ¿Cómo puedo crear un proyecto?
- ¿Cómo puedo crear una reserva?
- ¿Cómo puedo buscar la última reserva creada?
- ¿Cómo puedo buscar mis proyectos?
- ¿Cuál es el detalle de la máquina impresora 3D Resina?

Durante las pruebas, se evaluaron tanto el tiempo promedio como la tasa de éxito de cada tarea.

4.3.1. Análisis del Tiempo Promedio por Tarea

El gráfico de la Figura 32 presenta los tiempos promedio registrados para cada tarea. En general, los resultados muestran que cuatro de las cinco tareas fueron completadas de manera eficiente, con tiempos significativamente inferiores al límite establecido de 120 segundos.

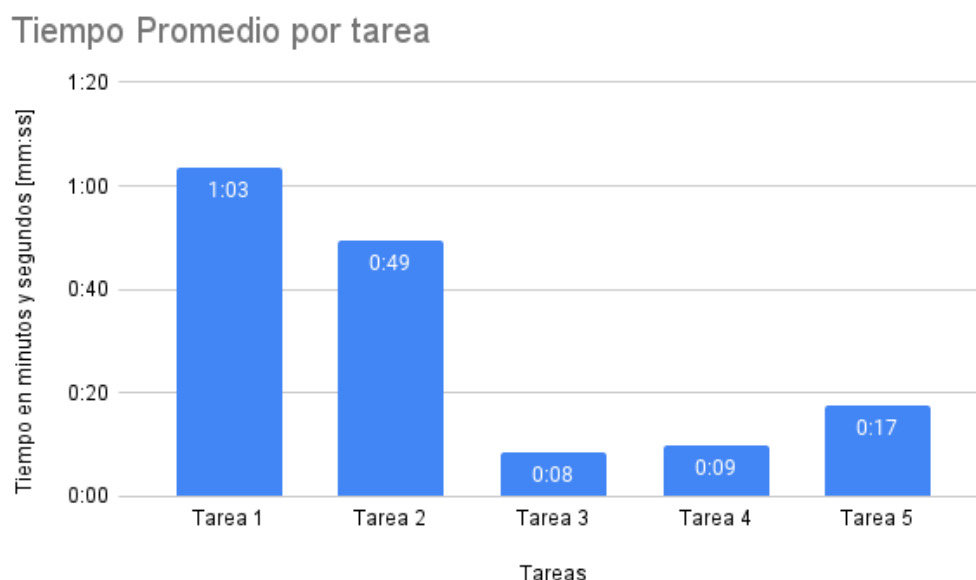


Figura 32: Tiempo promedio por tarea - Prueba de usabilidad.

Los hallazgos clave son los siguientes:

- **Tarea 1:** Tiempo promedio de 1 minuto y 3 segundos (63 segundos). Aunque el promedio se encuentra dentro del límite, un usuario tardó 140 segundos, superando el tiempo máximo permitido. Esto resalta la necesidad de analizar posibles mejoras en la claridad o flujo de esta tarea.
- **Tareas 2, 3, 4 y 5:** Todas se completaron con tiempos promedio muy por debajo del límite (entre 8 y 49 segundos), reflejando que estas tareas son intuitivas y fáciles de realizar para los usuarios.

4.3.2. Análisis del Promedio de Éxito por Tarea

La Figura 33 muestra el porcentaje de éxito para cada tarea. Mientras que tres de las cinco tareas lograron una tasa de éxito del 100 %, dos tareas presentaron un éxito reducido al 80 %.

Promedio de éxito por tarea

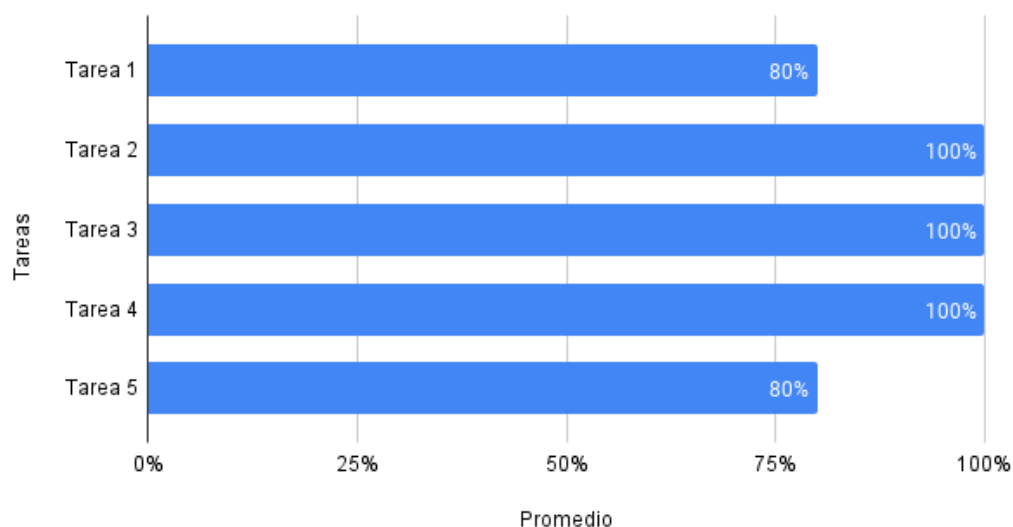


Figura 33: Promedio de éxito por tarea - Prueba de usabilidad.

Conclusiones específicas:

- **Tarea 1:** Solo el 80 % de los usuarios completaron esta tarea con éxito. El usuario que superó el límite de tiempo (140 segundos) fue la causa principal del bajo porcentaje de éxito, lo que sugiere ajustes en el diseño o una mejor orientación para los usuarios.
- **Tarea 5:** También alcanzó un éxito del 80 %. Un usuario confundió la “impresora 3D” con la “impresora 3D de resina”. Esto indica la necesidad de mejorar la nomenclatura o incluir descripciones más claras en el sistema para evitar confusiones.

4.4. Validación de la API Escalable y Resiliente

La API fue sometida a pruebas de carga para evaluar su rendimiento bajo condiciones de alta demanda. Las métricas clave incluyeron el porcentaje de uso de CPU, la latencia de respuesta, los códigos de respuestas clasificados en 2XX, 4XX y 5XX, y la capacidad del sistema para escalar automáticamente mediante el servicio *Elastic Beanstalk* de AWS a través de un balanceador de carga.

4.4.1. Especificaciones de las Máquinas y Servicios Utilizados para la Prueba de Carga

Para las pruebas de carga del sistema, se utilizó un entorno configurado con un balanceador de carga basado en la plataforma Docker, ejecutándose sobre **Amazon Linux 2023/4.4.2** en una arquitectura de 64 bits. Este balanceador de carga está configurado con capacidades de *autoescalado*, permitiendo ajustar dinámicamente la cantidad de instancias según la demanda del sistema.

El esquema de autoescalado considera las siguientes especificaciones:

- **Tipo de entorno:** Equilibrio de carga con instancias bajo demanda.
- **Instancias configuradas:** Tipos `t3.micro` y `t3.small`.
- **Cantidad de instancias:**
 - Mínimo: 1 instancia.
 - Máximo: 4 instancias.
- **Métrica de escalado:** Utilización de CPU (`CPUUtilization`) con una estadística promedio (`Average`).
- **Umbrales de escalado:**
 - Umbral superior: 50 % de uso de CPU. Al alcanzar este nivel, se incrementa el número de instancias en 1.
 - Umbral inferior: 20 % de uso de CPU. Al caer por debajo de este nivel, se reduce el número de instancias en 1.
- **Recuperación de escalado:** 360 segundos entre eventos de escalado.
- **Zonas de disponibilidad:** Configuración para disponibilidad en cualquier zona.
- **Imagen de máquina (AMI):** `ami-0af09b1a86b2d6800`.

Este entorno asegura que el sistema mantenga un rendimiento estable frente a aumentos o disminuciones en la carga de trabajo, optimizando el uso de recursos y garantizando la disponibilidad del servicio.

4.4.2. Pruebas de Carga Realizadas con K6

Para evaluar el rendimiento de la API, se utilizó la herramienta **K6**¹⁹, que permite realizar pruebas de carga simulando un número creciente de usuarios concurrentes. El script dise-

¹⁹<https://k6.io> es una herramienta de código abierto diseñada para realizar pruebas de carga y rendimiento en sistemas, simulando usuarios concurrentes.

ñado para estas pruebas incluyó las siguientes etapas:

- Incremento gradual de usuarios desde 0 hasta 100 en un periodo de 1 minuto.
- Mantenición de 100 usuarios concurrentes durante 13 minutos.
- Reducción gradual de usuarios hasta 0 en 30 segundos.

Además, el script incluyó solicitudes HTTP para verificar la disponibilidad del sistema y consultas a la API mediante **GraphQL** para evaluar su desempeño. Los principales endpoints probados incluyeron:

- **Health Check:** Confirmación del estado operativo de la API.
- **Consulta de disponibilidad de bloques:** Verificación de datos sobre la disponibilidad de horarios para máquinas específicas.
- **Consulta de reservas:** Recuperación de información sobre las reservas actuales, usuarios y proyectos asociados.

Se estableció un umbral de rendimiento que exigía que el 95 % de las solicitudes respondieran en menos de 700 ms ($p(95) < 700\text{ms}$). El código completo se encuentra detallado en los anexos 5.5.

4.4.3. Monitoreo y Resultados Iniciales

Durante las pruebas, se monitorearon métricas clave mediante un panel de control en **AWS CloudWatch**, que incluyó:

- **Códigos de respuesta HTTP:** Clasificados en 2XX, 4XX y 5XX.
- **Latencia de respuesta:** Tiempo promedio y máximo de respuesta para las solicitudes.
- **Utilización de CPU:** Nivel de uso en cada instancia del sistema.
- **Escalado automático:** Registro del número de instancias activas durante el periodo de prueba.

Estado Inicial Antes de comenzar la prueba de carga, el sistema mostró las siguientes métricas, ver Figura 34:

■ **Códigos de respuesta HTTP:**

- 2XX: Promedio de 12 respuestas por intervalo.
- 4XX: Promedio de 85 respuestas por intervalo.
- 5XX: No se registraron respuestas.

■ **Latencia de respuesta:** Tiempo promedio de 0,003 s (alrededor de 3 ms).

■ **Utilización de CPU:** La instancia EC2 i-0e2be778172d7ee66 utilizaba solo el 0,53 %.

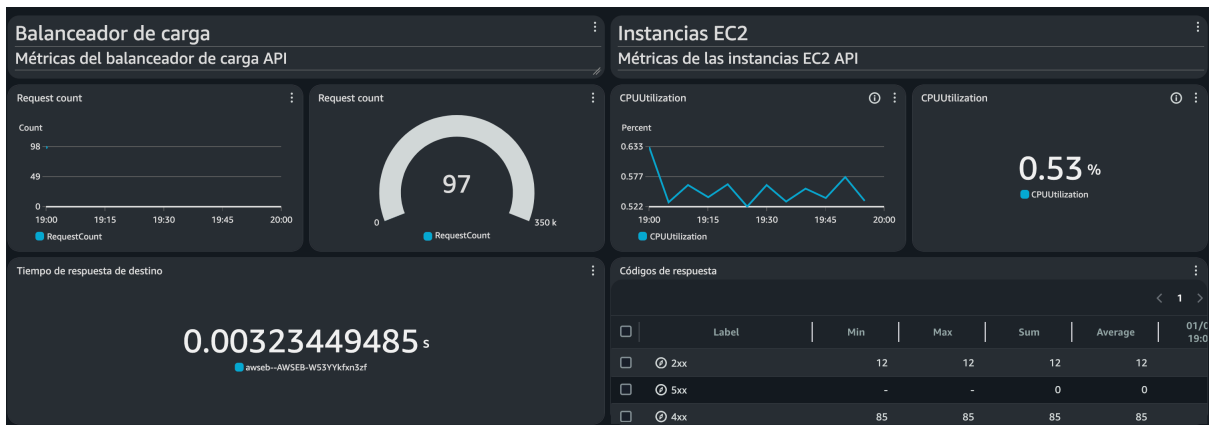


Figura 34: Estado inicial antes de la prueba de carga.

En la Figura 35 se observa que solo se estaba utilizando una instancia, ya que no se superaba el umbral del 50 % de uso de CPU.

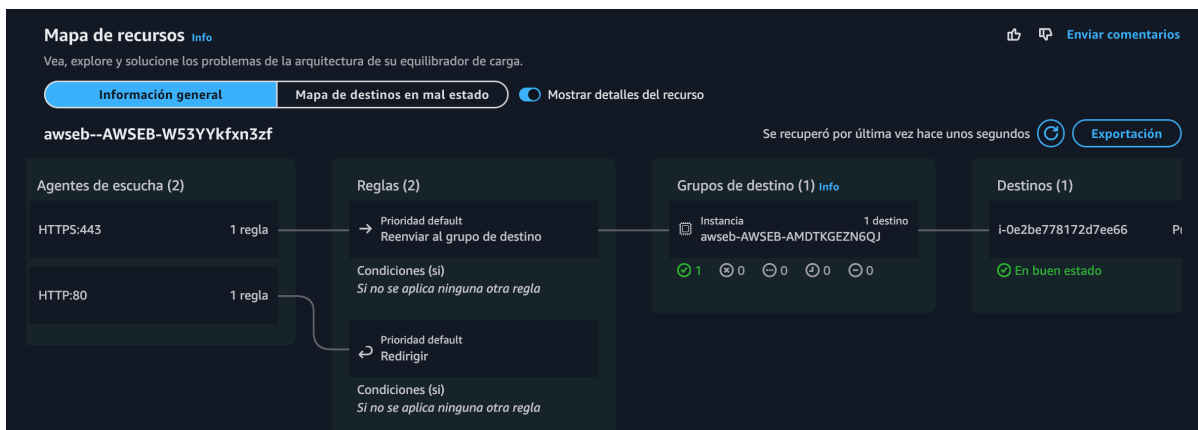


Figura 35: Mapa de recursos del balanceador de carga al inicio de la prueba.

Estado Medio Durante la prueba de carga, ver Figura 36, en un lapso de 10 minutos, se registraron las siguientes métricas:

■ **Códigos de respuesta HTTP:**

- 2XX: Promedio de 33K respuestas.
- 4XX: Promedio de 4,6 respuestas.
- 5XX: Promedio de 1 respuesta.

■ **Latencia de respuesta:** Tiempo promedio de 0,11 s.

■ **Utilización de CPU:** La instancia 1 alcanzó un 50,8 % de uso, en un intervalo de 360 segundos, activando el escalado de una segunda instancia (i-05a576c6c91e26b8a).

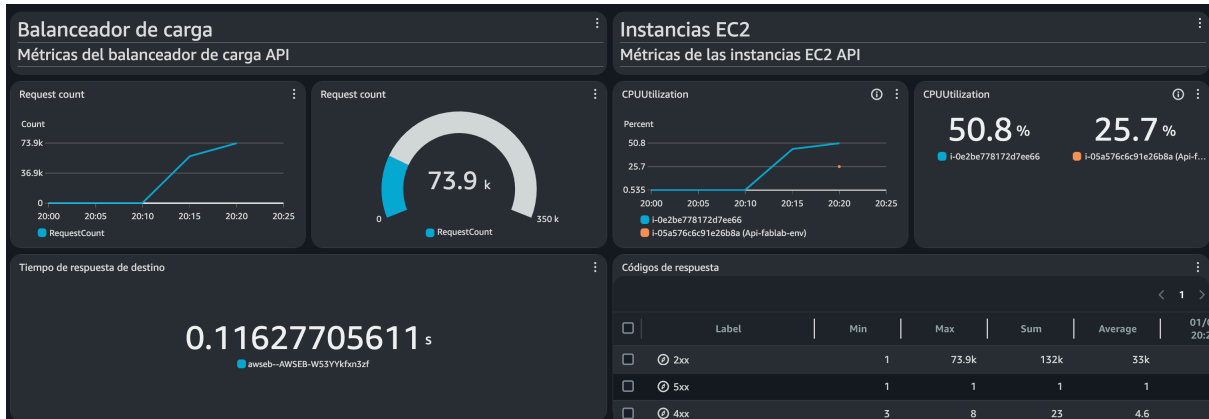


Figura 36: Estado medio durante la prueba de carga.

En la Figura 37, se observa que la segunda instancia comenzó a inicializarse, generando temporalmente errores 5XX, mientras la primera instancia mantenía un estado saludable.

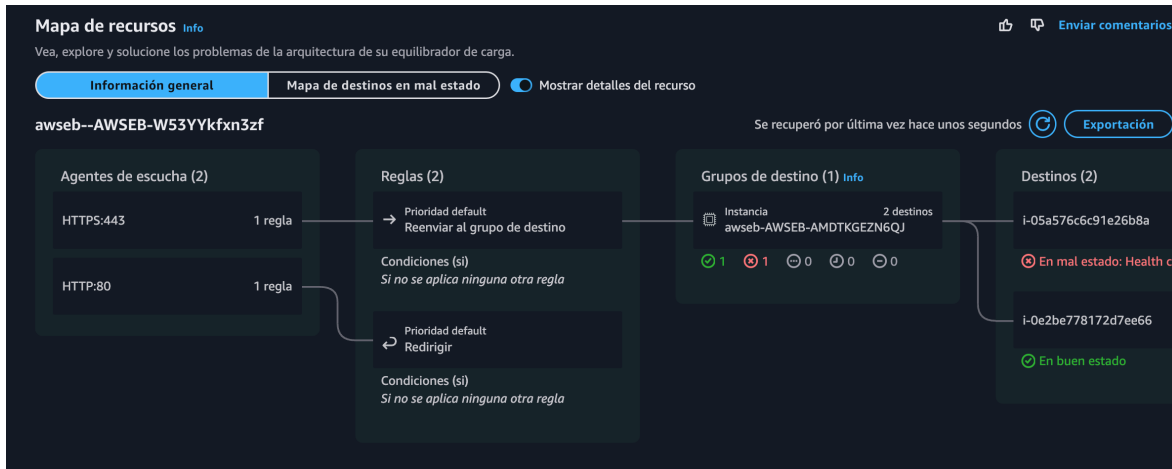


Figura 37: Mapa de recursos durante la prueba de carga.

Estado Final Al final de la prueba de carga, tras 15-20 minutos, ver Figura 38, se registraron las siguientes métricas:

■ **Códigos de respuesta HTTP:**

- 2XX: Promedio de 121K respuestas.
- 4XX: Promedio de 17,5 respuestas.
- 5XX: Promedio de 4 respuestas.

■ **Latencia de respuesta:** Tiempo promedio de 0,06 s.

■ **Utilización de CPU:** La instancia 1 se apagó tras mantenerse bajo el umbral del 20 % durante 360 segundos, mientras que la instancia 2 continuó operativa.

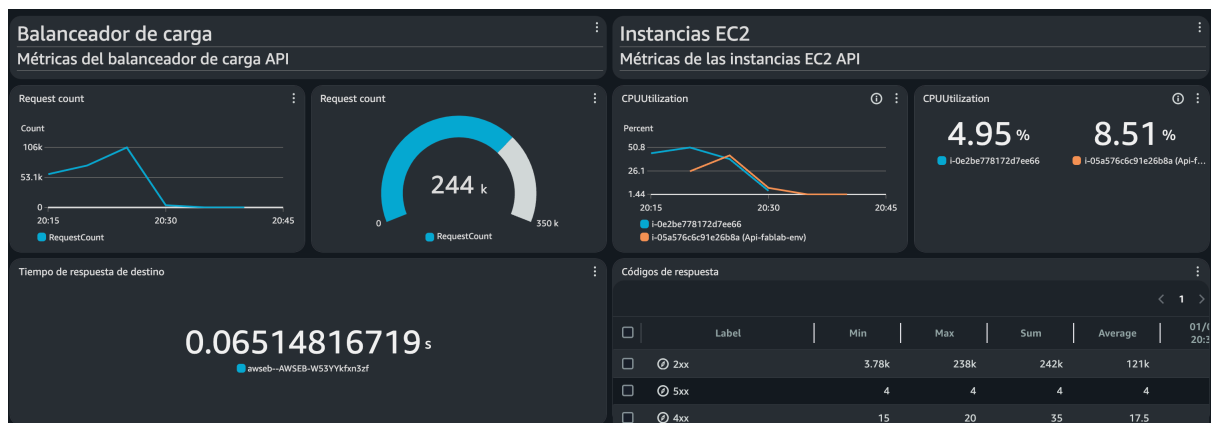


Figura 38: Estado final de la prueba de carga.

En la Figura 39, ambas instancias estaban en buen estado manejando solicitudes, y en la Figura 40 se observa el apagado de la instancia 1 al finalizar la prueba.

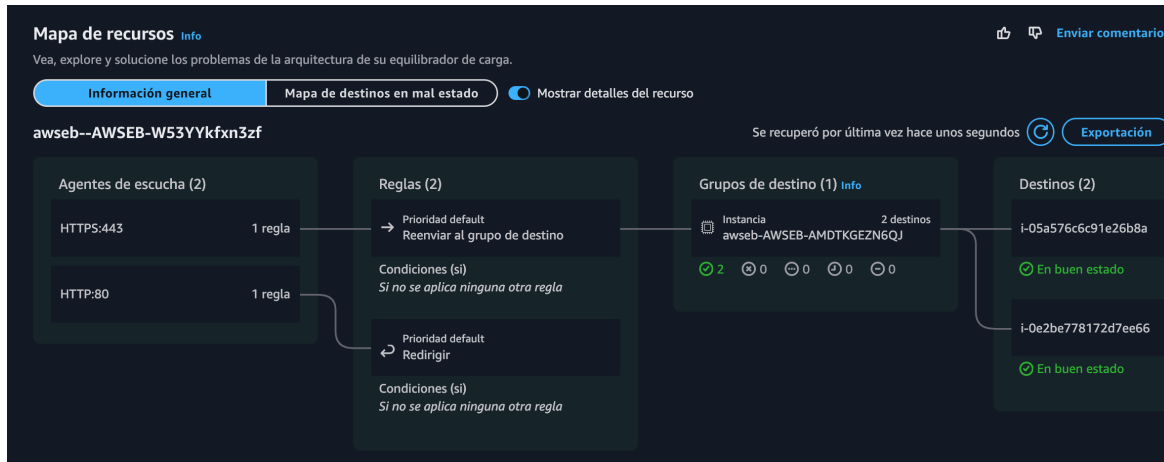


Figura 39: Mapa de recursos durante el final de la prueba de carga.

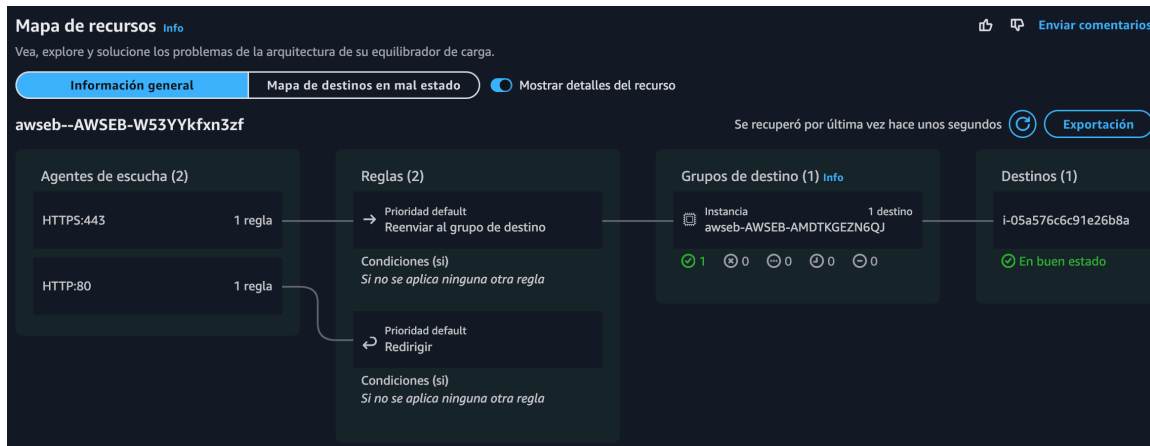


Figura 40: Mapa de recursos al finalizar la prueba de carga.

Las métricas obtenidas durante la prueba de carga confirmaron que el sistema logró cumplir parcialmente con el objetivo planteado de diseñar una API escalable y resiliente. En particular, se evidenció que:

- El sistema fue capaz de escalar automáticamente de 1 a 2 instancias durante el aumento en la demanda, como estaba definido en los umbrales de uso de CPU. Este comportamiento demuestra la escalabilidad del sistema frente a incrementos significativos en la carga.
- La API mantuvo tiempos de respuesta promedio dentro de los límites establecidos ($p(95) < 700\text{ms}$), lo que refleja su robustez bajo condiciones de alta concurrencia.

- Aunque una de las instancias presentó problemas iniciales durante el escalado, el sistema fue capaz de estabilizarse y manejar las solicitudes de manera eficiente, adaptándose dinámicamente para sostener el rendimiento requerido.

4.5. Validación de la Digitalización de Procesos y Generación de KPI's

El sistema logró digitalizar los procesos prioritarios identificados, como la gestión de usuarios, máquinas, horarios y proyectos, así como la reserva de máquinas. Para validar este objetivo, se realizaron las siguientes pruebas funcionales.

4.5.1. Gestión de Usuarios

Para la validación de la gestión de los usuarios, se realizaron dos pruebas. En primer lugar, se revisó el proceso de ingreso de un usuario, siendo este un estudiante de la Universidad Técnica Federico Santa María accediendo por primera vez al sistema, sin previa experiencia en el sistema. Al utilizar el “PasaporteUSM”, el sistema le solicitó ingresar sus datos personales antes de poder usar el software, ver Figura 41.

The screenshot shows the 'Configuración de la cuenta' (Account Configuration) page in the FabLab portal. The page has a blue header with the FabLab logo and a 'Cuenta' (Account) link. A sidebar on the left contains navigation links: Inicio (Home), Mis proyectos (My projects), Mis reservas (My reservations), Máquinas (Machines), Crear Reserva (Create Reservation), and Salir (Logout). The main content area is titled 'Configuración de la cuenta' and contains a form with the following fields:

- Nombre* (Name): Text input field.
- Segundo nombre (Second name): Text input field.
- Apellido paterno* (Paternal surname): Text input field.
- Apellido materno (Maternal surname): Text input field.
- Tipo Documento* (Document type): Dropdown menu with 'RUT' selected.
- N° Documento* (Document number): Text input field.
- Carrera (Career): Dropdown menu.
- Año de Ingreso (Year of entry): Text input field with a calendar icon and the text 'Seleccionar año'.
- Email*: Text input field with the value 'jose.rodriguezga@usmd.onmicrosoft.com'.
- Fecha de nacimiento* (Date of birth): Text input field with the value '05/11/2024' and a calendar icon.
- Sexo (Sex): Dropdown menu with 'Femenino' selected.

At the bottom of the form are two buttons: 'Cancelar' (Cancel) and 'Actualizar' (Update).

Figura 41: Ingreso por primera vez a través del Pasaporte USM - Portal FabLab

Después de completar esta información, el usuario logró ingresar a la pagina inicial del sistema, como se muestra en la Figura 42

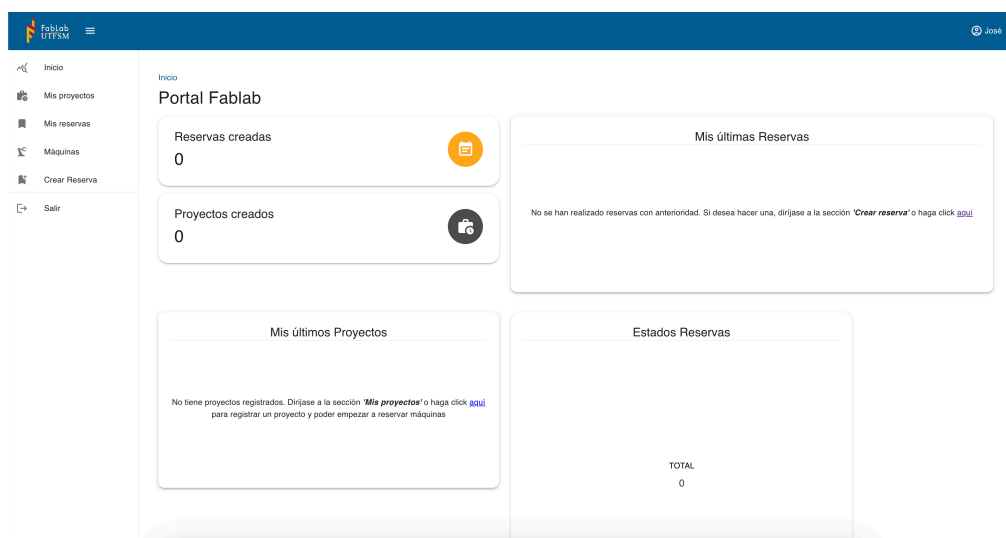


Figura 42: Inicio - Portal FabLab

A continuación, se realizó otra prueba en la que se asignó el rol de “FabLab” a un usuario, siendo este un ayudante del laboratorio. Al ingresar al sistema mediante su correo personal “...@fablab.usm.cl”, se observó que tiene más control del sistema, teniendo la opción de “Configuraciones”, pudiendo gestionar su horario, reservas, proyectos, máquinas, entre otros (ver Figura 43).

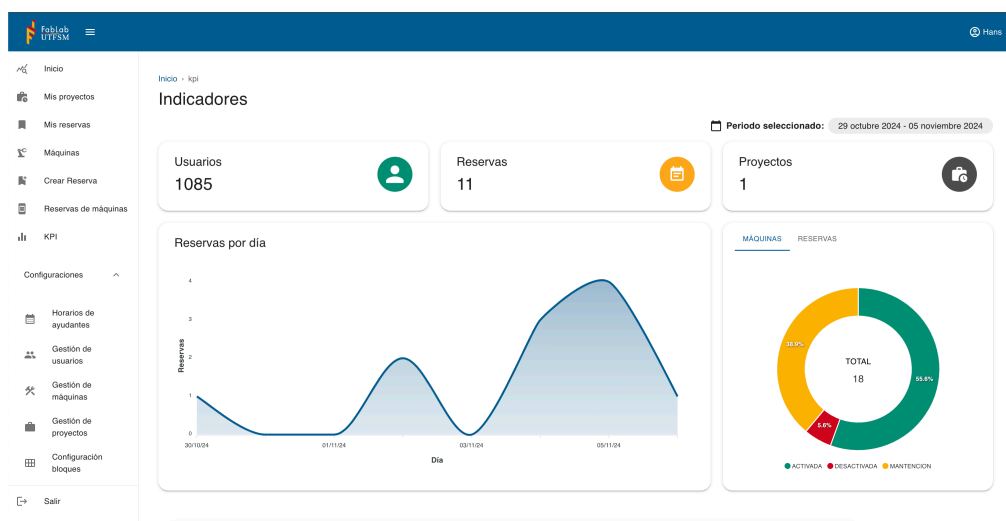
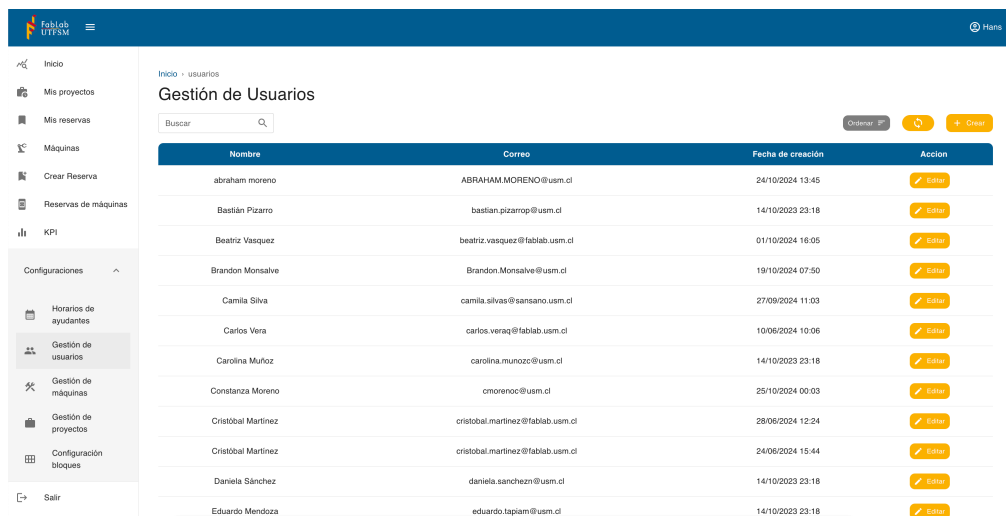


Figura 43: Ingreso por ayudante - Portal FabLab

Finalmente, como se muestra en la Figura 44, el ayudante puede visualizar el listado de todos los usuarios inscritos en la plataforma. Al seleccionar la opción “Editar”, es posible actualizar la información de un usuario.

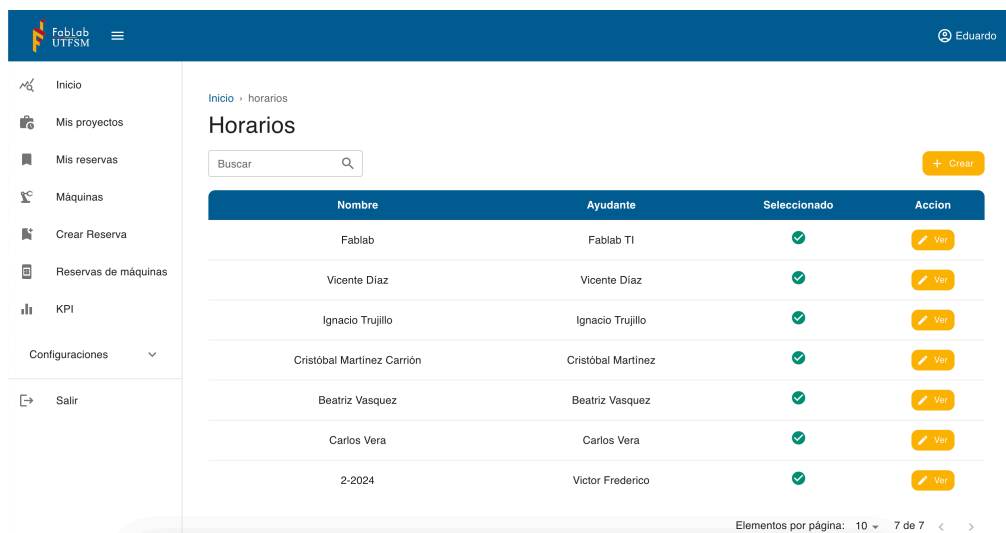


Nombre	Correo	Fecha de creación	Acción
abraham moreno	ABRAHAM.MORENO@usm.cl	24/10/2024 13:45	Editar
Bastían Pizarro	bastian.pizarrop@usm.cl	14/10/2023 23:18	Editar
Beatriz Vasquez	beatriz.vasquez@fablab.usm.cl	01/10/2024 16:05	Editar
Brandon Monsalve	Brandon.Monsalve@usm.cl	19/10/2024 07:50	Editar
Camila Silva	camila.silva@sansano.usm.cl	27/09/2024 11:03	Editar
Carlos Vera	carlos.vera@fablab.usm.cl	10/06/2024 10:06	Editar
Carolina Muñoz	carolina.muñoz@usm.cl	14/10/2023 23:18	Editar
Constanza Moreno	cmorenoc@usm.cl	25/10/2024 00:03	Editar
Cristóbal Martínez	cristobal.martinez@fablab.usm.cl	28/06/2024 12:24	Editar
Cristóbal Martínez	cristobal.martinez@fablab.usm.cl	24/06/2024 15:44	Editar
Daniela Sánchez	daniela.sanchezn@usm.cl	14/10/2023 23:18	Editar
Eduardo Mendoza	eduardo.lapiam@usm.cl	14/10/2023 23:18	Editar

Figura 44: Listado de usuarios - Portal FabLab

4.5.2. Horario de los ayudantes

La Figura 45 muestra la interfaz del software desde la perspectiva de un usuario que es ayudante del FabLab, por lo tanto posee el rol de “FabLab”. En esta vista, el usuario puede visualizar los horarios de todos los ayudantes, pero únicamente tiene permisos para editar su propio horario.



Nombre	Ayudante	Seleccionado	Acción
Fablab	Fablab TI	✓	Ver
Vicente Díaz	Vicente Díaz	✓	Ver
Ignacio Trujillo	Ignacio Trujillo	✓	Ver
Cristóbal Martínez Carrión	Cristóbal Martínez	✓	Ver
Beatriz Vasquez	Beatriz Vasquez	✓	Ver
Carlos Vera	Carlos Vera	✓	Ver
2-2024	Victor Frederico	✓	Ver

Figura 45: Vista de horarios para ayudantes - Portal FabLab

Por otro lado, en la Figura 46 se puede observar la interfaz de un usuario con el rol de “administrador”. En esta vista, el administrador cuenta con la capacidad de editar los horarios

de cualquier colaborador.

FabLab

UTFSM

Hans

Inicio

Mis proyectos

Mis reservas

Máquinas

Crear Reserva

Reservas de máquinas

KPI

Configuraciones

Salir

Inicio

horarios

Horarios

Buscar

Crear

Nombre	Ayudante	Seleccionado	Accion
Fablab	Fablab TI		Editar
Vicente Diaz	Vicente Diaz		Editar
Ignacio Trujillo	Ignacio Trujillo		Editar
Cristóbal Martínez Carrión	Cristóbal Martínez		Editar
Beatriz Vasquez	Beatriz Vasquez		Editar
Carlos Vera	Carlos Vera		Editar
2-2024	Victor Frederico		Editar

Elementos por página: 10 7 de 7

Figura 46: Vista administrador horarios - Portal FabLab

Finalmente, en la Figura 47 se ilustra la interfaz para crear o editar un horario de ayudante.

FabLab

UTFSM

Eduardo

Inicio

Mis proyectos

Mis reservas

Máquinas

Crear Reserva

Reservas de máquinas

KPI

Configuraciones

Salir

Inicio

horarios

crear

Crear horario de ayudante

Nombre del horario:

Nombre semestre - año (Juan Pérez s2-2024)

Seleccionado

Lunes		Martes		Miercoles		Jueves		Viernes	
Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25
Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50
Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15
Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40
Almuerzo		Almuerzo		Almuerzo		Almuerzo		Almuerzo	
Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:05	Bloque 10 15:15 - 15:50
Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:30	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15

Figura 47: Interfaz para crear o editar horario - Portal FabLab

Al día de hoy (1/11/2024), todos los ayudantes registrados en el FabLab han creado sus horarios para gestionar mejor sus jornadas de trabajo (ver Figura 45).

4.5.3. Gestión de Máquinas

La gestión y actualización del inventario de máquinas, como se detalla en la Tabla 1, constituye un pilar esencial para el funcionamiento del laboratorio. En la Figura 48 se evidencia la creación de todos los tipos de máquinas necesarias, alcanzando un total de 10 tipos de máquinas registradas. Este proceso fue llevado a cabo por un usuario con el rol de “FabLab”, quien actúa como ayudante del laboratorio.

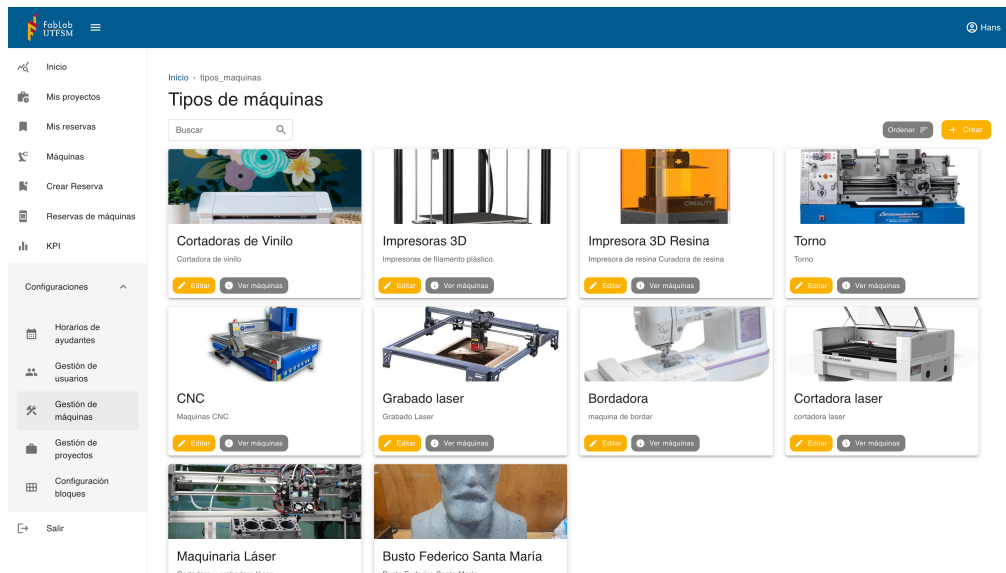


Figura 48: Tipos de máquinas creadas - Portal FabLab

Asimismo, en la Figura 49 se aprecia que, al seleccionar un tipo de máquina (en este caso Impresora 3D), se listan todas las máquinas correspondientes a ese tipo.

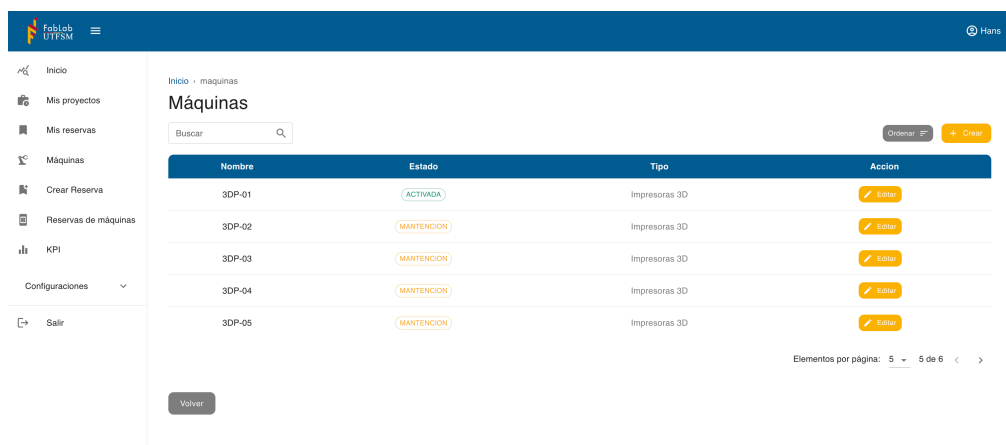


Figura 49: Máquinas Impresoras 3D creadas - Portal FabLab

Finalmente, la Figura 50 muestra la interfaz utilizada para crear, editar o realizar la eliminación lógica de una máquina. La máquina denominada “3DP-01” fue creada por un usuario con el rol de “FabLab”, quien se desempeña como ayudante del laboratorio.

Inicio > máquinas > 9ab007af6-e089-4042-ae58-d50278b7c08

Editar máquina

Nombre
3DP-01

Descripción
Materiales utilizados: PLA, PETG
Impresora Ender 3v2 Creality

Min. bloques
1

Max. bloques
4

Selecciona el tipo de máquina
Impresoras 3D

Estado de la máquina
Activada

Subir archivos
Arrastre sus archivos aquí, o haga clic para buscar sus archivos

Figura 50: Máquina Impresora 3D creada - 3DP-01 - Portal FabLab

Al 1 de noviembre de 2024, el FabLab cuenta con un total de 18 máquinas registradas, distribuidas en 10 tipos diferentes de máquinas, todas creadas por ayudantes con el rol de “FabLab”.

4.5.4. Gestión de Proyectos

De forma análoga a la creación de tipos de máquinas, la Figura 51 muestra todos los tipos de proyectos registrados por el FabLab. Al igual que en la gestión de máquinas, estos tipos de proyectos fueron creados por un usuario con el rol de “FabLab”, quien actúa como ayudante del laboratorio.

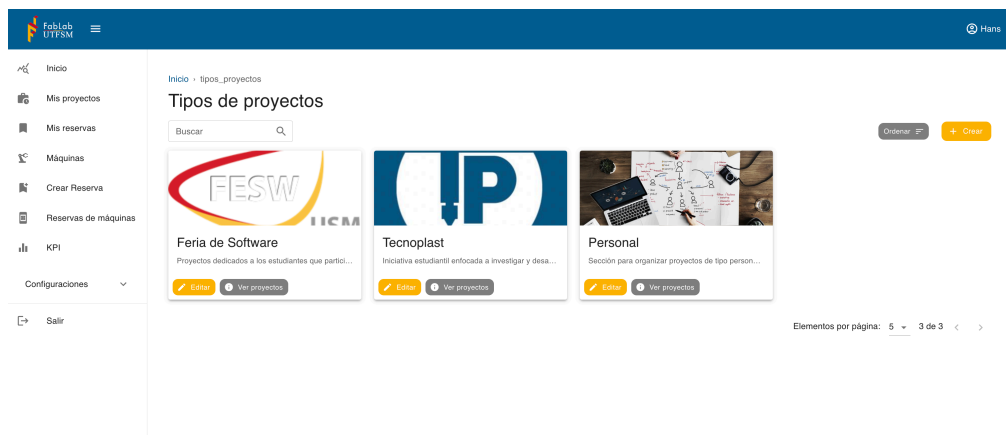


Figura 51: Tipos de proyectos creados - Portal FabLab

Al seleccionar “Ver proyectos”, se observa en la Figura 52 un listado de los proyectos creados por los usuarios para ese tipo de proyecto.

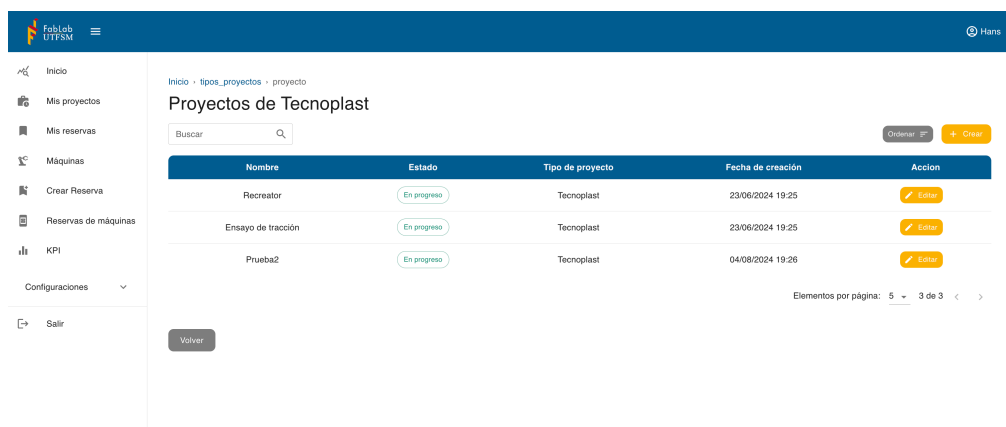


Figura 52: Tecnoplast proyectos - Portal FabLab

Finalmente, otro usuario del FabLab, al acceder a la sección “Mis proyectos”, puede visualizar todos los proyectos que ha creado. Esto se ilustra en la Figura 53. Además, en la Figura 54 se muestra el proceso en el que el usuario está creando un nuevo proyecto, el cual se completó de manera satisfactoria.

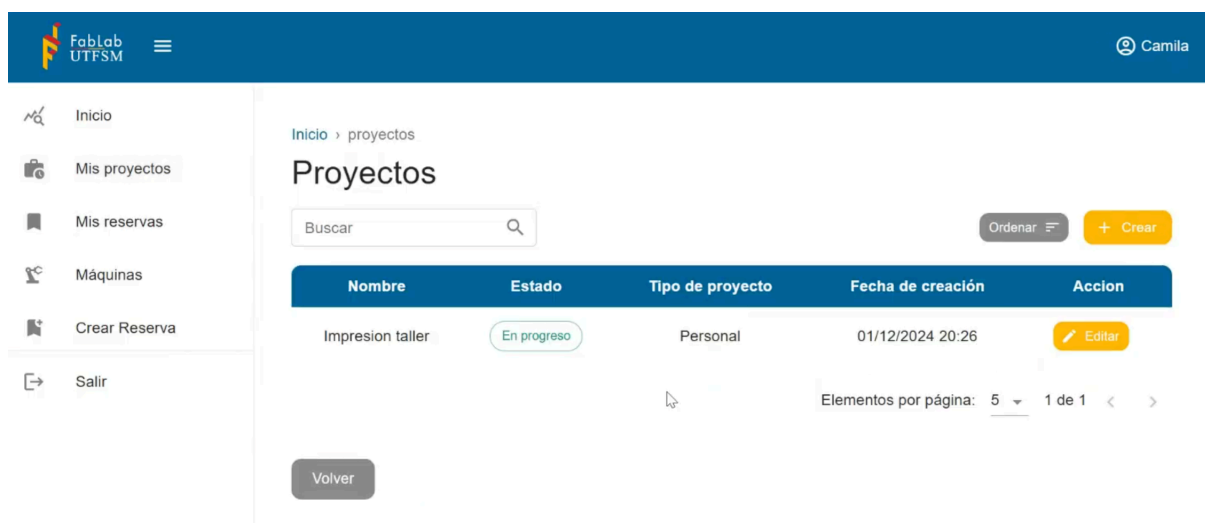


Figura 53: Proyectos de un usuario - Portal FabLab.

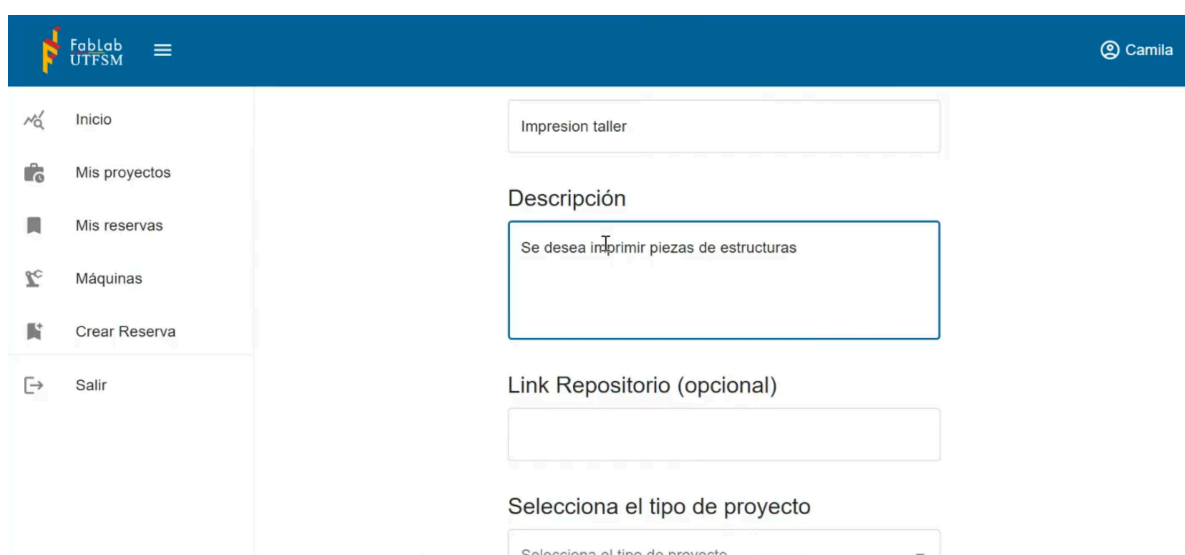
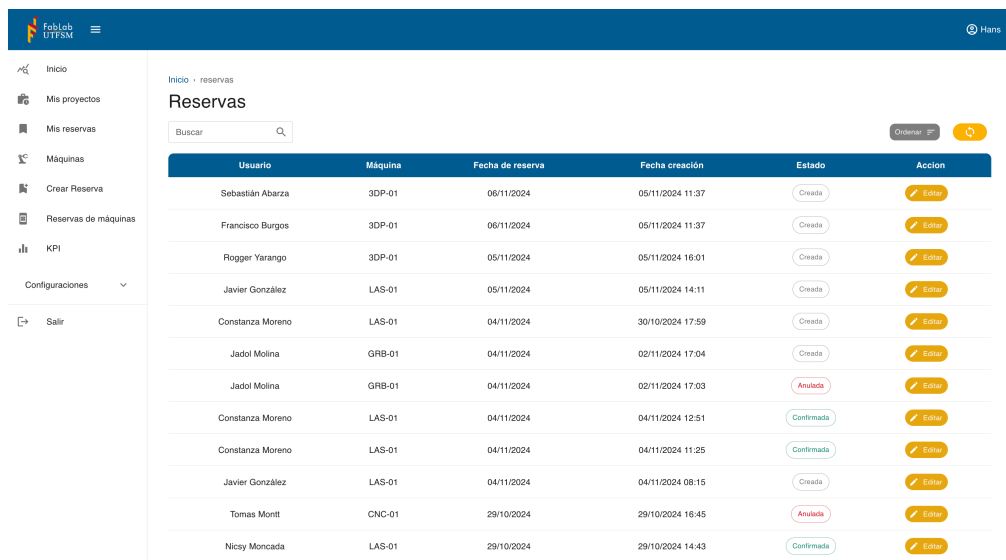


Figura 54: Proyecto creado por un usuario - Portal FabLab.

4.5.5. Reservas de máquinas

La Figura 55 muestra todas las reservas creadas por los usuarios. Esta vista está disponible únicamente para los roles de “FabLab” y “administrador”, correspondientes a ayudantes y administradores del laboratorio, respectivamente. En este caso particular, la vista corresponde a un usuario con el rol de “FabLab”, quien se desempeña como ayudante del laboratorio.

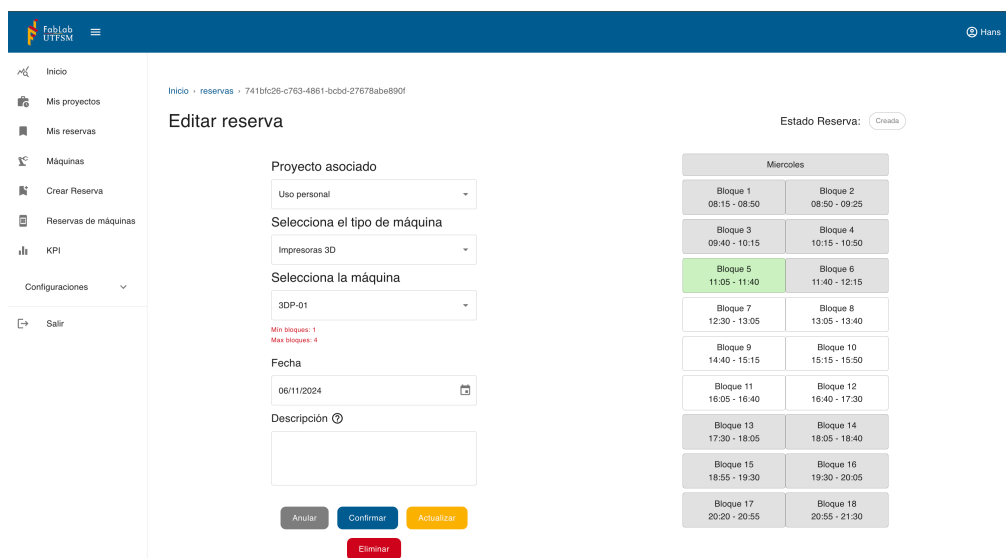


The screenshot shows the 'Reservas' (Reservations) page in the FabLab portal. The left sidebar contains navigation links: Inicio, Mis proyectos, Mis reservas, Máquinas, Crear Reserva, Reservas de máquinas, KPI, Configuraciones, and Salir. The main content area displays a table of reservations with columns: Usuario, Máquina, Fecha de reserva, Fecha creación, Estado, and Acción. The table lists 14 reservations, with the 10th reservation (Constanza Moreno, LAS-01, 04/11/2024, 04/11/2024 12:51) highlighted in green, indicating it is the selected reservation.

Usuario	Máquina	Fecha de reserva	Fecha creación	Estado	Acción
Sebastián Abarza	3DP-01	06/11/2024	05/11/2024 11:37	Creada	Editar
Francisco Burgos	3DP-01	06/11/2024	05/11/2024 11:37	Creada	Editar
Rogger Yarango	3DP-01	05/11/2024	05/11/2024 16:01	Creada	Editar
Javier González	LAS-01	05/11/2024	05/11/2024 14:11	Creada	Editar
Constanza Moreno	LAS-01	04/11/2024	30/10/2024 17:59	Creada	Editar
Jadol Molina	GRB-01	04/11/2024	02/11/2024 17:04	Creada	Editar
Jadol Molina	GRB-01	04/11/2024	02/11/2024 17:03	Anulada	Editar
Constanza Moreno	LAS-01	04/11/2024	04/11/2024 12:51	Confirmada	Editar
Constanza Moreno	LAS-01	04/11/2024	04/11/2024 11:25	Confirmada	Editar
Javier González	LAS-01	04/11/2024	04/11/2024 08:15	Creada	Editar
Tomas Moritt	CNC-01	29/10/2024	29/10/2024 16:45	Anulada	Editar
Niccy Moncada	LAS-01	29/10/2024	29/10/2024 14:43	Confirmada	Editar

Figura 55: Reservas creadas - Portal FabLab

Además, al seleccionar “Editar”, se muestra en la Figura 56 la interfaz para modificar la reserva, pudiendo cambiar el estado, el proyecto, la máquina o los bloques (horario de la reserva).



The screenshot shows the 'Editar reserva' (Edit reservation) form in the FabLab portal. The left sidebar is the same as in Figure 55. The main content area displays the form with fields for: Proyecto asociado (Uso personal), Selección el tipo de máquina (Impresoras 3D), Selección la máquina (3DP-01), Fecha (06/11/2024), and Descripción. The form also includes a 'Estado Reserva' dropdown set to 'Creada'. On the right, there is a calendar view for 'Miércoles' showing 18 blocks (Bloque 1 to Bloque 18) with their respective time slots. Block 5 (11:05 - 11:40) is highlighted in green, indicating it is the selected block for the reservation.

Figura 56: Editar una reserva creada - Portal FabLab

Finalmente, en la Figura 57 se muestra cómo un usuario sin el rol de “FabLab” o “administrador”, es decir, un usuario regular, realiza satisfactoriamente la creación de una reserva para el día 17/12/2024 en el bloque 9.



Figura 57: Creación de una reserva por un usuario - Portal FabLab.

Posteriormente, en la Figura 58, se confirma que la reserva fue creada exitosamente. Los detalles de la reserva incluyen:

- **Proyecto asociado:** “Uso personal”.
- **Tipo de máquina seleccionada:** “Impresora 3D Resina”.
- **Máquina seleccionada:** “CUR-01”.
- **Fecha:** 17/12/2024 (Martes).
- **Bloque:** 9 (14:40 a 15:15).

ELABORACIÓN E IMPLEMENTACIÓN DE UN SISTEMA PARA DIGITALIZAR PROCESOS Y GENERAR KPI'S DEL FUNCIONAMIENTO DEL LABORATORIO FABLAB DE LA UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

Inicio · reservas · c5f35627-c2b2-4e86-a657-bbac5e64c1c6

Editar reserva

Estado Reserva: Creada

Proyecto asociado

Uso personal

Selección de tipo de máquina

Impresora 3D Resina

Selección de máquina

CUR-01

Min bloques: 1
Max bloques: 1

Fecha

17/12/2024

Descripción

Impresión de formas estructurales

Anular Confirmar Actualizar Eliminar

Martes

Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25
Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50
Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15
Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40
Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50
Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15
Bloque 13 17:30 - 18:05	Bloque 14 18:05 - 18:40
Bloque 15 18:55 - 19:30	Bloque 16 19:30 - 20:05
Bloque 17 20:20 - 20:55	Bloque 18 20:55 - 21:30

Figura 58: Reserva creada por un usuario - Portal FabLab.

Además, según el horario mostrado en la Figura 59, el ayudante “Vicen Díaz” estaba de turno durante el bloque seleccionado, asegurando la disponibilidad de asistencia en el laboratorio.

Inicio · horarios · 5f4d9df0-162d-44be-b7bc-42e5fcff8d9

Ayudante: Vicente Díaz

Nombre del horario:

Vicente Díaz

Seleccionado

Lunes		Martes		Miércoles		Jueves		Viernes	
Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25	Bloque 1 08:15 - 08:50	Bloque 2 08:50 - 09:25
Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50	Bloque 3 09:40 - 10:15	Bloque 4 10:15 - 10:50
Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15	Bloque 5 11:05 - 11:40	Bloque 6 11:40 - 12:15
Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40	Bloque 7 12:30 - 13:05	Bloque 8 13:05 - 13:40
Almuerzo		Almuerzo		Almuerzo		Almuerzo		Almuerzo	
Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:15	Bloque 10 15:15 - 15:50	Bloque 9 14:40 - 15:05	Bloque 10 15:15 - 15:50
Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:30	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15	Bloque 11 16:05 - 16:40	Bloque 12 16:40 - 17:15
Bloque 13 17:30 - 18:05	Bloque 14 18:05 - 18:40	Bloque 13 17:30 - 18:05	Bloque 14 18:05 - 18:40	Bloque 13 17:30 - 18:05	Bloque 14 18:05 - 18:40	Bloque 13 17:30 - 18:05	Bloque 14 18:05 - 18:40	Bloque 13 17:30 - 18:05	Bloque 14 18:05 - 18:40

Figura 59: Horario del ayudante “Vicente Díaz” - Portal FabLab.

4.6. Indicadores

Se generaron y visualizaron indicadores clave de desempeño (KPI's) en tiempo real mediante *Power BI*, como se ilustra en la Figura 60. Los indicadores incluyeron:

- Número total de reservas realizadas.
- Total de reservas realizadas por mes.
- Número total de estudiantes agrupados por carrera y género.
- Total de proyectos asociados a cada máquina reservada.
- Total de proyectos clasificados por tipo de proyecto.
- Número total de estudiantes activos por mes.

La generación y precisión de estos indicadores fueron validadas por los administradores del FabLab, quienes confirmaron a priori que reflejan correctamente las actividades del laboratorio.

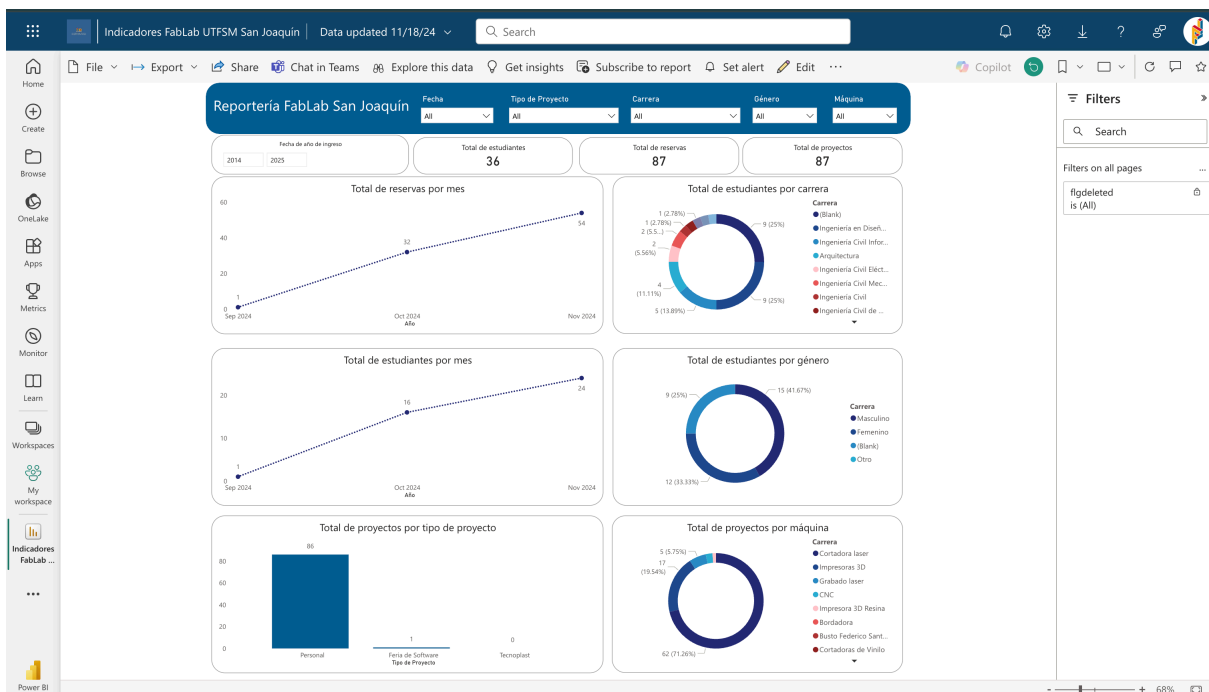


Figura 60: Indicadores en tiempo real del laboratorio FabLab usando la plataforma *Power BI*

CAPÍTULO 5

CONCLUSIONES

5.1. Conclusiones generales

Para la realización de esta memoria, fue necesario conocer a fondo los problemas y necesidades del laboratorio FabLab de la Universidad Técnica Federico Santa María, campus San Joaquín. Haber trabajado durante dos años como administrador TI (2020 - 2022) resultó fundamental, ya que me permitió acercarme considerablemente a las necesidades tanto de los colaboradores como de los usuarios del laboratorio. Con esta experiencia en mente, se diseñó una solución orientada a resolver los problemas y requerimientos del FabLab, desarrollando un sistema web que digitaliza varios procesos que anteriormente eran completamente manuales.

La aplicación web fue creada utilizando diversas tecnologías, destacando el uso del *framework* *Next.js* para la interfaz y experiencia de usuario, junto con la librería *Material UI*, que provee numerosos componentes para agilizar el desarrollo. Además, se desarrolló una *API GraphQL* con el *framework* *NestJs* para ser consumida por la aplicación web. Todo el sistema, tanto el *frontend* como el *backend*, fue desplegado en los servidores de *Amazon Web Services*, empleando distintos servicios. Se utilizó *RDS* para una base de datos relacional *PostgreSQL*; *S3* para el almacenamiento de archivos estáticos (imágenes, PDF, videos, entre otros); *Amazon Cognito* para la autenticación y autorización de usuarios; *Elastic Beanstalk* para alojar y escalar tanto la aplicación web como la *API GraphQL*, mediante el uso de un *Load Balancer* e instancias *EC2*; y *CodePipeline* para el proceso de construcción y despliegue de la aplicación web y la *API GraphQL*.

Con este nuevo sistema, el FabLab experimentará una transformación digital significativa, pasando de procesos manuales a digitales. Se anticipa un periodo de adaptación y aprendizaje tanto para los colaboradores del laboratorio como para sus usuarios. Durante esta fase de adaptación, es posible que surjan nuevas necesidades o problemas relacionados con el sistema, lo cual será fundamental para el proceso de mejora continua y la evolución del sistema en el futuro.

Superada esta fase inicial de adaptación, se espera que el sistema sea utilizado durante varios años, permitiendo la recopilación de información valiosa que contribuya a obtener indicadores clave para seguir impulsando la transformación y el crecimiento del laboratorio. Así, se espera generar un impacto no solo en la numerosa comunidad estudiantil de la Universidad Técnica Federico Santa María, sino también en los estudiantes de diversos colegios a lo largo de Chile que son invitados a aprender y a crear en este laboratorio. De igual manera, se espera que este nuevo sistema ayude a reducir la carga laboral de los ayudantes del FabLab, al disminuir el tiempo dedicado a los numerosos procesos manuales que realizan diariamente para asegurar el correcto funcionamiento del laboratorio.

En resumen, todos los objetivos específicos planteados en esta memoria fueron abordados y, en su mayoría, cumplidos satisfactoriamente. La implementación del sistema representa un paso significativo hacia la transformación digital del FabLab, reduciendo procesos manuales y mejorando la eficiencia operativa. Sin embargo, las áreas de mejora identificadas, como la optimización del escalado automático y el ajuste de flujos específicos en la plataforma web, representan oportunidades importantes para perfeccionar el sistema en futuras iteraciones.

5.2. Conclusiones específicas

El desarrollo del sistema para el FabLab permitió cumplir con los objetivos planteados inicialmente en esta memoria. A continuación, se presenta una evaluación detallada de cada uno de ellos:

1. Análisis de procesos y priorización de digitalización

Se realizó un análisis de los principales procesos del laboratorio (ver Tabla 1), identificando aquellos susceptibles de digitalización. Como resultado, se priorizaron los siguientes procesos clave:

- Creación y análisis de KPI's.
- Gestión de usuarios.
- Gestión de máquinas.
- Gestión de horarios de turnos.
- Reserva de máquinas.
- Gestión de inventario de máquinas.

Estos procesos fueron validados mediante reuniones con los administradores y personal del FabLab, quienes confirmaron que reflejan las principales necesidades operativas del laboratorio. Sin embargo, otros procesos, como la gestión de talleres y programas, quedaron fuera del alcance debido a las prioridades establecidas.

2. Identificación de actores principales

Los actores principales del sistema se definieron como usuarios del FabLab, personal del FabLab y administradores del FabLab. Este objetivo fue validado a través de reuniones con el

personal del laboratorio, quienes confirmaron que los roles y responsabilidades fueron correctamente representados en el sistema. Además, las pruebas de usabilidad demostraron que el sistema satisface las necesidades específicas de cada tipo de usuario, con funcionalidades alineadas a sus roles.

3. Desarrollo de una plataforma web intuitiva y accesible

La plataforma web fue validada mediante pruebas de usabilidad con cinco usuarios representativos, quienes completaron cinco tareas clave. Los resultados, ilustrados en las Figuras 32 y 33, muestran que:

- Cuatro de las cinco tareas fueron completadas de manera eficiente, con tiempos promedio muy por debajo del límite establecido de 120 segundos.
- Tareas 2, 3 y 4: Estas tareas demostraron ser altamente intuitivas, con tiempos promedio bajos y una tasa de éxito del 100 %.
- La Tarea 1 presentó un tiempo promedio dentro del límite establecido; sin embargo, un usuario tardó 140 segundos, lo que redujo la tasa de éxito al 80 %. Además, durante esta tarea, dos usuarios intentaron crear un proyecto con un nombre ya existente, lo que generó una advertencia con un código de error técnico y poco legible. A pesar de ello, los usuarios lograron completar la tarea al cambiar el nombre del proyecto.
- La Tarea 5 alcanzó también un éxito del 80 %, debido a confusiones entre las máquinas “impresora 3D” e “impresora 3D de resina”.

En general, la plataforma cumplió con los estándares de usabilidad; sin embargo, se identificaron áreas de mejora en el flujo de ciertas tareas específicas, como la creación de proyectos con nombres ya existentes, donde los mensajes de error podrían ser más claros y comprensibles. Asimismo, se detectaron oportunidades para optimizar la nomenclatura de las máquinas, con el fin de evitar confusiones entre opciones similares.

4. Diseño de una API escalable y resiliente

Las pruebas de carga realizadas con K6 demostraron que el sistema es capaz de mantener niveles de servicio adecuados incluso ante picos de demanda. Las métricas obtenidas durante las pruebas confirmaron que:

- El sistema escaló automáticamente de 1 a 2 instancias al superar el umbral del 50 % de uso de CPU.

- La API mantuvo tiempos de respuesta promedio dentro del límite ($p(95) < 700\text{ms}$).
- Se detectaron errores temporales (5XX) durante la inicialización de nuevas instancias, lo que destaca la necesidad de mejorar el monitoreo y las configuraciones de escalado.

Estas observaciones evidencian que la API es escalable y resiliente, cumpliendo con el objetivo planteado, aunque con áreas de optimización identificadas.

5. Implementación de la digitalización de procesos y generación de KPI's

El sistema implementado digitalizó exitosamente los procesos priorizados, como la gestión de usuarios, máquinas, horarios, proyectos y reservas. Además, se generaron indicadores clave de desempeño (KPI's) en tiempo real mediante *Power BI*, como se muestra en la Figura 60. Los KPI's incluyen:

- Número total de reservas realizadas.
- Total de reservas realizadas por mes.
- Número total de estudiantes agrupados por carrera y género.
- Total de proyectos asociados a cada máquina reservada.
- Total de proyectos clasificados por tipo de proyecto.
- Número total de estudiantes activos por mes.

Estos indicadores permiten un monitoreo constante y en tiempo real del desempeño del laboratorio, brindando información valiosa para la toma de decisiones y la mejora continua.

5.3. Recomendaciones

El laboratorio FabLab es un espacio que fomenta el aprendizaje y la creación, proporcionando las herramientas necesarias para que cualquier estudiante pueda transformar sus ideas en algo tangible. Este espíritu no solo está dirigido a los usuarios del laboratorio, sino también a los ayudantes, quienes encuentran en el FabLab un ambiente de confianza y libertad para innovar y desarrollarse.

Mi recomendación se dirige principalmente a los ayudantes: continúen innovando. Aprovechen esta oportunidad para aprender y experimentar, ya que el FabLab ofrece tanto el espacio como la confianza para introducir nuevas tecnologías y así mejorar la experiencia de los usuarios, al igual que la suya en las tareas cotidianas.

Es saludable cuestionar lo que ya está establecido en el FabLab, pero, al hacerlo, procuren siempre proponer mejoras o construir algo superior. La oportunidad de hacer algo nuevo y mejor está ahí, esperando por ustedes.

5.4. Trabajo Futuro

Con este nuevo desarrollo implementado, el FabLab cuenta con dos plataformas principales: por un lado, “ClassRoom” y, por otro, el Portal FabLab (el proyecto presentado en esta memoria). Ambas herramientas comparten únicamente el sistema de autenticación y autorización proporcionado por *Amazon Cognito*. “ClassRoom” se enfoca en las capacitaciones en máquinas y herramientas mediante videos asincrónicos, así como en la gestión de talleres y programas a través de cursos en *Moodle*. En cambio, el Portal FabLab se dedica a la automatización de otros procesos, como se mencionó anteriormente.

El problema radica en que estas dos plataformas no están integradas, lo que impide la recopilación de indicadores clave relacionados con las actividades de *ClassRoom*. Por lo tanto, un aspecto fundamental para el trabajo futuro es establecer una conexión entre la API de *ClassRoom* y la *API GraphQL* del Portal FabLab, permitiendo una gestión centralizada y un acceso integral a los indicadores de desempeño.

Otro punto importante a considerar es la automatización de la gestión de horarios para capacitaciones. Actualmente, este proceso no está cubierto ni por el Portal FabLab ni por *ClassRoom*. No fue incluido en el alcance de esta memoria debido a las prioridades establecidas por el laboratorio, que consideró necesario primero implementar funcionalidades como la administración de horarios de los ayudantes para las reservas, la gestión de máquinas, la reserva de las máquinas por parte de los usuarios y el registro de proyectos, siendo esta última esencial para documentar el propósito de uso del laboratorio.

Finalmente, un aspecto que hubiera sido ideal incorporar en este trabajo es la creación de una entidad “laboratorio”, donde todos los registros pudieran asociarse a un FabLab específico. Esto facilitaría la gestión si en el futuro se establece un FabLab en otro campus de la Universidad, permitiendo mantener registros separados por laboratorio sin necesidad de desarrollar una nueva instancia del software ni alojarlo en un servidor distinto con una nueva base de datos.

5.5. Palabras finales del autor

Desarrollar este software fue un desafío total en múltiples aspectos. Me permitió aprender y aplicar los elementos fundamentales del desarrollo de software, utilizando tecnologías esenciales en el mercado actual. Asumí diversos roles: desarrollador backend, frontend, administrador de servicios en la nube, diseñador, e incluso usuario para realizar pruebas en la

aplicación.

Considero que, para un proyecto de esta envergadura, es ideal contar con un equipo más amplio que permita distribuir mejor los roles y equilibrar el trabajo en la construcción del software.

Finalmente, trabajar durante dos años como administrador TI en el laboratorio fue fundamental, ya que me ayudó a comprender en profundidad las necesidades y problemas cotidianos tanto de los usuarios como de los colaboradores del FabLab. Esta experiencia me permitió centrarme en desarrollar el producto en lugar de invertir tiempo en entender los requerimientos desde cero.

ANEXOS

Esquema Completo de la API GraphQL

```
type PageInfo {
  currentPage: Int!
  perPage: Int!
  totalPages: Int!
  totalItems: Int!
  hasPreviousPage: Boolean!
  hasNextPage: Boolean!
}

type FileResponse {
  key: String!
  path: String!
}

type Machine {
  id: ID!
  name: String!
  description: String
  machineTypeId: MachinesType!
  createdAt: Timestamp!
  updatedAt: Timestamp!
  status: String!
  files: [FileResponse!]
  isActive: Boolean!
  minBlocks: Int!
  maxBlocks: Int!
}

"""
`Date` type as integer. Type represents date and time as
  number of milliseconds from start of UNIX epoch.
"""
scalar Timestamp

type PaginatedMachines {
  pageInfo: PageInfo!
  nodes: [Machine!]
}
```

```
type MachineStatusCount {
  label: String!
  value: Int!
  color: String!
}

type MachinesType {
  id: ID!
  name: String!
  description: String
  machines: [Machine!]!
  files: [FileResponse!]
  createdAt: Timestamp!
  updatedAt: Timestamp!
  isActive: Boolean!
}

type PaginatedMachinesType {
  pageInfo: PageInfo!
  nodes: [MachinesType!]
}

type ProjectsType {
  id: ID!
  name: String!
  description: String
  projects: [Project!]!
  files: [FileResponse!]
  createdAt: Timestamp!
  updatedAt: Timestamp!
  isActive: Boolean!
}

type PaginatedProjectsType {
  pageInfo: PageInfo!
  nodes: [ProjectsType!]
}

type Block {
  id: ID!
  day: String!
  name: String
  startHour: String!
  endHour: String!
```

```
    createdAt: Timestamp!  
    updatedAt: Timestamp!  
    isActive: Boolean!  
}  
  
type PaginatedBlocks {  
  pageInfo: PageInfo!  
  nodes: [Block!]  
}  
  
type Booking {  
  id: ID!  
  user: User!  
  machine: Machine!  
  project: Project!  
  date: Timestamp!  
  blocks: [Block!]!  
  description: String  
  status: String!  
  createdAt: Timestamp!  
  updatedAt: Timestamp!  
  isActive: Boolean!  
}  
  
type PaginatedBooking {  
  pageInfo: PageInfo!  
  nodes: [Booking!]  
}  
  
type BlockAvailability {  
  block: Block!  
  isAvailable: Boolean!  
}  
  
type BlocksAvailabilityResponse {  
  blocks: [BlockAvailability!]!  
}  
  
type User {  
  id: String!  
  cognitoId: String!  
  name: String  
  middleName: String  
  lastName: String
```

```
secondLastName: String
birthday: Timestamp
email: String!
identifierType: String!
identifier: String
ownedProjects: [Project!]
memberProjects: [Project!]
bookings: [Booking!]
createdAt: Timestamp!
updatedAt: Timestamp!
isActive: Boolean!
}

type PaginatedUsers {
  pageInfo: PageInfo!
  nodes: [User!]
}

type UserTypeCognito {
  username: String!
  email: String!
}

type ResponseSyncCognito {
  totalChecked: Int!
  totalCreated: Int!
}

type Project {
  id: ID!
  name: String!
  description: String
  owner: User!
  members: [User!]
  projectId: ProjectsType!
  repository: String
  files: [FileResponse!]
  feedback: String
  isGroup: Boolean!
  startDate: Timestamp!
  endDate: Timestamp
  status: String!
  createdAt: Timestamp!
  updatedAt: Timestamp!
```

```
    isActive: Boolean!  
    startDateFormatted: String!  
    endDateFormatted: String!  
}  
  
type PaginatedProject {  
    pageInfo: PageInfo!  
    nodes: [Project!]  
}  
  
type Schedule {  
    id: ID!  
    name: String!  
    blocksId: [String!]!  
    selected: Boolean!  
    owner: User!  
    createdAt: Timestamp!  
    updatedAt: Timestamp!  
    isActive: Boolean!  
}  
  
type PaginatedSchedule {  
    pageInfo: PageInfo!  
    nodes: [Schedule!]  
}  
  
type MachinesKPI {  
    data: [MachineStatusCount!]!  
}  
  
type UsersKPI {  
    total: Int!  
}  
  
type DonutData {  
    label: String!  
    value: Int!  
    color: String!  
}  
  
type ChartData {  
    labels: [String!]!  
    values: [Int!]!  
}
```

```
type BookingsKPI {
  total: Int!
  donutData: [DonutData!]!
  chartData: ChartData!
}

type ProjectsKPI {
  total: Int!
}

type KPI {
  machines: MachinesKPI!
  users: UsersKPI!
  bookings: BookingsKPI!
  projects: ProjectsKPI!
}

type KPIBYUSER {
  bookings: BookingsKPI!
  projects: ProjectsKPI!
}

type Query {
  bookings(page: Int = 1, perPage: Int = 10, sortBy: String,
    sortOrder: SortOrders = ASC, search: String, isActive:
    IsActive = TRUE): PaginatedBooking!
  findAllBookingsByUserId(page: Int = 1, perPage: Int = 10,
    sortBy: String, sortOrder: SortOrders = ASC, search:
    String, isActive: IsActive = TRUE, userId: ID!):
    PaginatedBooking!
  booking(id: ID!): Booking!
  getBlocksAvailability(dateString: String!, machineId: ID!):
    BlocksAvailabilityResponse!
  machinesType(page: Int = 1, perPage: Int = 10, sortBy:
    String, sortOrder: SortOrders = ASC, search: String,
    isActive: IsActive = TRUE): PaginatedMachinesType!
  allMachinesType: [MachinesType!]!
  findAllMachineTypeWithoutPaginated: [MachinesType!]!
  machineType(id: ID!): MachinesType!
  blocks(page: Int = 1, perPage: Int = 10, sortBy: String,
    sortOrder: SortOrders = ASC, search: String, isActive:
    IsActive = TRUE): PaginatedBlocks!
  FindAllblocksWithoutPaginated: [Block!]!
```

```
block(id: ID!): Block!
machines(page: Int = 1, perPage: Int = 10, sortBy: String,
  sortOrder: SortOrders = ASC, search: String, isActive:
  IsActive = TRUE): PaginatedMachines!
machinesByMachineType(page: Int = 1, perPage: Int = 10,
  sortBy: String, sortOrder: SortOrders = ASC, search:
  String, isActive: IsActive = TRUE, machineTypeId: ID):
  PaginatedMachines!
machine(id: ID!): Machine!
findAllMachinesByMachineTypeWithoutPaginated(machineTypeId:
  ID!): [Machine!]!
projects(page: Int = 1, perPage: Int = 10, sortBy: String,
  sortOrder: SortOrders = ASC, search: String, isActive:
  IsActive = TRUE, projectId: ID): PaginatedProject!
findAllProjectsByUserId(page: Int = 1, perPage: Int = 10,
  sortBy: String, sortOrder: SortOrders = ASC, search:
  String, isActive: IsActive = TRUE, projectId: ID,
  userId: ID!): PaginatedProject!
findAllProjectsByProjectTypeId(page: Int = 1, perPage: Int
  = 10, sortBy: String, sortOrder: SortOrders = ASC, search
  : String, isActive: IsActive = TRUE, projectId: ID,
  projectTypeId: ID!): PaginatedProject!
projectsByProjectType(page: Int = 1, perPage: Int = 10,
  sortBy: String, sortOrder: SortOrders = ASC, search:
  String, isActive: IsActive = TRUE, projectId: ID):
  PaginatedProject!
findAllWithoutPaginatedProjectsByUserAndMembers(userId: ID
  !): [Project!]!
project(id: ID!): Project!
projectsType(page: Int = 1, perPage: Int = 10, sortBy:
  String, sortOrder: SortOrders = ASC, search: String,
  isActive: IsActive = TRUE): PaginatedProjectsType!
findAllProjectsTypeWithoutPaginated: [ProjectsType!]!
projectType(id: ID!): ProjectsType!
users(page: Int = 1, perPage: Int = 10, sortBy: String,
  sortOrder: SortOrders = ASC, search: String, isActive:
  IsActive = TRUE): PaginatedUsers!
findAllUserWithoutPaginated: [User!]!
usersCognito: [UserTypeCognito!]!
findUser(id: ID!): User!
findUserByCognitoID(id: ID!): User!
schedules(page: Int = 1, perPage: Int = 10, sortBy: String,
  sortOrder: SortOrders = ASC, search: String, isActive:
  IsActive = TRUE): PaginatedSchedule!
```

```
    schedule(id: ID!): Schedule!
    kpi(startDate: Timestamp!, endDate: Timestamp!): KPI!
    kpiByUser(id: ID!): KPIBYUSER!
}

enum SortOrders {
    ASC
    DESC
}

enum IsActive {
    ALL
    TRUE
    FALSE
}

type Mutation {
    createBooking(createBookingInput: CreateBookingInput!):
        Booking!
    updateBooking(updateBookingInput: UpdateBookingInput!):
        Booking!
    blockBooking(id: ID!): Booking!
    restoreBooking(id: ID!): Booking!
    removeBooking(id: ID!): Booking!
    createMachinesType(createMachinesTypeInput:
        CreateMachinesTypeInput!, files: [Upload!], foldername:
        pathFolderNamesEnum): MachinesType!
    updateMachineType(updateMachinesTypeInput:
        UpdateMachinesTypeInput!, files: [Upload!], foldername:
        pathFolderNamesEnum): MachinesType!
    blockMachinesType(id: ID!): MachinesType!
    restoreMachinesType(id: ID!): MachinesType!
    removeMachinesType(id: ID!): MachinesType!
    removeImagesMachinesType(id: ID!, filesToRemove: [
        FileInputToRemove!]!): MachinesType!
    uploadFile(file: Upload!, foldername: String!):
        FileResponse!
    uploadFiles(files: [Upload!]!, foldername: String!): [
        FileResponse!]!

    """key is the path file without url. Is the foldername/
        fileName"""
    deleteFile(key: String!): Boolean!
    createBlock(createBlockInput: CreateBlockInput!): Block!
```

```
updateBlock(updateBlockInput: UpdateBlockInput!): Block!
blockBlock(id: ID!): Block!
restoreBlock(id: ID!): Block!
removeBlock(id: ID!): Block!
createMachine(createMachineInput: CreateMachineInput!,
  files: [Upload!], foldername: pathFolderNamesEnum):
  Machine!
updateMachine(updateMachineInput: UpdateMachineInput!,
  files: [Upload!], foldername: pathFolderNamesEnum):
  Machine!
blockMachine(id: ID!): Machine!
restoreMachine(id: ID!): Machine!
removeMachine(id: ID!): Machine!
removeImagesMachines(id: ID!, filesToRemove: [
  FileInputToRemove!]): Machine!
createProject(createProjectInput: CreateProjectInput!,
  files: [Upload!], foldername: String): Project!
updateProject(updateProjectInput: UpdateProjectInput!,
  files: [Upload!], foldername: String): Project!
blockProject(id: ID!): Project!
restoreProject(id: ID!): Project!
removeProject(id: ID!): Project!
removeImagesProject(id: ID!, filesToRemove: [
  FileInputToRemove!]): Project!
createProjectsType(createProjectsTypeInput:
  CreateProjectsTypeInput!, files: [Upload!], foldername:
  pathFolderNamesEnum): ProjectsType!
updateProjectType(updateProjectsTypeInput:
  UpdateProjectsTypeInput!, files: [Upload!], foldername:
  pathFolderNamesEnum): ProjectsType!
blockProjectsType(id: ID!): ProjectsType!
restoreProjectsType(id: ID!): ProjectsType!
removeProjectsType(id: ID!): ProjectsType!
removeImagesProjectsType(id: ID!, filesToRemove: [
  FileInputToRemove!]): ProjectsType!
createUser(createUserInput: CreateUserInput!): User!
syncUsersFromCognito: ResponseSyncCognito!
updateUser(updateUserInput: UpdateUserInput!): User!
createSchedule(createScheduleInput: CreateScheduleInput!):
  Schedule!
updateSchedule(updateScheduleInput: UpdateScheduleInput!):
  Schedule!
blockSchedule(id: ID!): Schedule!
restoreSchedule(id: ID!): Schedule!
```

```
    removeSchedule(id: ID!): Schedule!
}

input CreateBookingInput {
  userId: String!
  machineId: ID!
  projectId: ID!
  date: Timestamp!
  blocksId: [ID!]!
  status: STATUS = CREADA
  description: String
}

enum STATUS {
  CREADA
  ANULADA
  CONFIRMADA
}

input UpdateBookingInput {
  userId: String
  machineId: ID
  projectId: ID
  date: Timestamp
  blocksId: [ID!]
  status: STATUS = CREADA
  description: String
  id: ID!
}

input CreateMachinesTypeInput {
  name: String!
  description: String!
}

"""The `Upload` scalar type represents a file upload."""
scalar Upload

enum pathFolderNamesEnum {
  projectType
  project
  machineType
  machine
}
```

```
input UpdateMachinesTypeInput {
  name: String
  description: String
  id: ID!
}

input FileInputToRemove {
  key: String!
  path: String!
}

input CreateBlockInput {
  day: Days!
  startHour: String!
  name: String
  endHour: String!
}

enum Days {
  LUNES
  MARTES
  MIERCOLES
  JUEVES
  VIERNES
  SABADO
  DOMINGO
}

input UpdateBlockInput {
  day: Days
  startHour: String
  name: String
  endHour: String
  id: ID!
}

input CreateMachineInput {
  name: String!
  description: String!
  status: StatusMachine = ACTIVADA
  machineTypeId: ID!
  minBlocks: Float!
  maxBlocks: Float!
}
```

```
}

enum StatusMachine {
    DESACTIVADA
    MANTENCION
    ACTIVADA
}

input UpdateMachineInput {
    name: String
    description: String
    status: StatusMachine = ACTIVADA
    machineTypeId: ID
    minBlocks: Float
    maxBlocks: Float
    id: ID!
}

input CreateProjectInput {
    name: String!
    description: String!
    ownerId: ID!
    membersId: [ID!]
    isGroup: Boolean!
    projectId: ID!
    repository: String
}

input UpdateProjectInput {
    name: String
    description: String
    ownerId: ID
    membersId: [ID!]
    isGroup: Boolean
    projectId: ID
    repository: String
    id: ID!
    status: Status = EN_PROGRESO
    feedback: String
}

enum Status {
    EN_PROGRESO
    FINALIZADO
}
```

```
}

input CreateProjectsTypeInput {
  name: String!
  description: String!
}

input UpdateProjectsTypeInput {
  name: String
  description: String
  id: ID!
}

input CreateUserInput {
  name: String
  cognitoId: String!
  middleName: String
  lastName: String
  secondLastName: String
  identifierType: IdentifierType = RUT
  identifier: String
  birthday: Timestamp
  email: String!
}

enum IdentifierType {
  RUT
  PASAPORTE
}

input UpdateUserInput {
  name: String
  middleName: String
  lastName: String
  secondLastName: String
  identifierType: IdentifierType = RUT
  identifier: String
  birthday: Timestamp
  email: String!
  id: ID!
}

input CreateScheduleInput {
  name: String!
```

```
    blocksId: [String!]!
    selected: Boolean!
    ownerId: ID!
  }

  input UpdateScheduleInput {
    name: String
    blocksId: [String!]
    selected: Boolean
    ownerId: ID
    id: ID!
  }
```

Codigo en javascript para prueba de carga utilizando K6

```
import http from "k6/http";
import { check, sleep } from "k6";

export let options = {
  stages: [
    { duration: "30s", target: 50 }, // Escala a 50
      usuarios en 30s.
    { duration: "1m", target: 100 }, // Escala a 100
      usuarios en 1m.
    { duration: "13m", target: 100 }, // Mantén 100
      usuarios durante 13m.
    { duration: "30s", target: 0 }, // Reduce a 0
      usuarios en 30s.
  ],
  thresholds: {
    http_req_duration: ["p(95)<700"], // El 95% de las
      requests bajo 700ms
  },
};

export default function () {
  // Health Check
  let healthCheckRes = http.get("https://dev.api.fablab.usm.cl/");
  check(healthCheckRes, {
    "health check status is 200": (r) => r.status === 200,
  });

  // GraphQL - GetBlocksAvailability
```

```
const graphqlUrl = "https://dev.api.fablab.usm.cl/graphql";
const blocksAvailabilityPayload = JSON.stringify({
  query: `
    query GetBlocksAvailability($dateString: String!,
      $machineId: ID!) {
      getBlocksAvailability(dateString: $dateString,
        machineId: $machineId) {
        blocks {
          isAvailable
          block {
            day
            endHour
            name
            id
            startHour
          }
        }
      }
    }
  `,
  variables: {
    dateString: "2024-12-10T15:24:37-03:00",
    machineId: "324abc6a-53b4-400c-9ab1-549752c26c16",
  },
});

let blocksAvailabilityRes = http.post(graphqlUrl,
  blocksAvailabilityPayload, {
  headers: { "Content-Type": "application/json" },
});
check(blocksAvailabilityRes, {
  "blocks availability status is 200": (r) => r.status ===
    200,
  "blocks availability contains data": (r) => JSON.parse(r.
    body).data !== null,
});

// GraphQL - Bookings Query
const bookingsPayload = JSON.stringify({
  query: `
    query Bookings {
      bookings {
        nodes {
          id

```

```
        machine {
            id
            minBlocks
            maxBlocks
            name
        }
        project {
            id
            name
        }
        user {
            id
            email
            name
        }
        description
    }
    pageInfo {
        totalItems
        perPage
        totalPages
        hasPreviousPage
        hasNextPage
        currentPage
    }
}
},
});

let bookingsRes = http.post(graphqlUrl, bookingsPayload, {
    headers: { "Content-Type": "application/json" },
});
check(bookingsRes, {
    "bookings status is 200": (r) => r.status === 200,
    "bookings contains data": (r) => JSON.parse(r.body).data
    !== null,
});

// Reducimos un poco el tiempo entre solicitudes para
    mantener presión
sleep(0.5);
}
```

REFERENCIAS BIBLIOGRÁFICAS

- [Auth0, 2024a] Auth0 (2024a). Autenticación frente a autorización.
- [Auth0, 2024b] Auth0 (2024b). Openid connect protocol.
- [Auth0, 2024c] Auth0 (2024c). ¿qué es oauth 2.0?
- [Azure, 2023a] Azure, M. (2023a). Azure active directory.
- [Azure, 2023b] Azure, M. (2023b). Microsoft azure.
- [de Innovación UC, 2022] de Innovación UC, C. (2022). La reactivación de la red chilena de fablabs.
- [Dussel, 2010] Dussel, Inés, y. L. A. Q. (2010). *Educación y nuevas tecnologías: los desafíos pedagógicos ante el mundo digital*. Fundación Santillana, Ciudad Autónoma de Buenos Aire.
- [Express.js, 2023] Express.js (2023). Sitio web express.js.
- [Fiaidhi y Mohammed, 2018] Fiaidhi, J. y Mohammed, S. (2018). Fab labs: A platform for innovation and extreme automation. *IT Professional*, 20(5):83–90.
- [Foundation, 2022] Foundation, F. (2022). Fab foundation.
- [GraphQL, 2023] GraphQL (2023). Sitio web graphql.
- [Grigorik, 2013] Grigorik, I. (2013). *High Performance Browser Networking*. O'Reilly Media.
- [Lindley, 2023] Lindley, C. (2023). Front-end developer handbook 2023.
- [Martin Fowler, 2002] Martin Fowler, David Rice, M. F. E. H. R. M. R. S. (2002). *Patterns of Enterprise Application Architecture*. Addison Wesley.
- [Mell y Grance, 2011] Mell, P. y Grance, T. (2011). The NIST definition of cloud computing. *National Institute of Standards and Technology Special Publication*, 800-145:30.
- [Microsoft, 2024] Microsoft (2024). Power bi.
- [Morakanyane et al., 2017] Morakanyane, R., Grace, A., y O'Reilly, P. (2017). Conceptualizing digital transformation in business organizations: A systematic review of literature. En *30th Bled eConference Digital Transformation – From Connecting Things to Transforming Our Lives*, pp. 18–21, Bled, Slovenia. University of Maribor Press.
- [Nestjs, 2023] Nestjs (2023). Nestjs.
- [Network, 2023] Network, F. (2023). Fablab network community.

- [Next.js, 2023] Next.js (2023). Next.js.
- [Parmenter, 2015] Parmenter, D. (2015). *Key Performance Indicators: Developing, Implementing, and Using Winning KPIs*. John Wiley & Sons.
- [PostgreSQL, 2024] PostgreSQL (2024). What is postgresql?
- [Poulton, 2017] Poulton, N. (2017). *Docker Deep Dive*. Leanpub.
- [React, 2023] React (2023). React.
- [Rojas, 2021] Rojas, M. M. P. (2021). *Plan de mejora de proceso de capacitación, centrado en la confiabilidad del talento humano, para Laboratorio de Fabricación Digital*. Memoria, Departamento de Ingeniería Mecánica. Universidad Técnica Federico Santa María.
- [Ruiz, 2022] Ruiz, A. O. (2022). *Metodología de Diseño Organizacional para Laboratorios de Fabricación Digital en Chile*. Memoria, Departamento de Industrias. Universidad Técnica Federico Santa María.
- [Service, 2023] Service, A. W. (2023). Computación en la nube con aws.
- [Services, 2023a] Services, A. W. (2023a). Amazon elastic container registry.
- [Services, 2023b] Services, A. W. (2023b). Aws cloudfront.
- [Services, 2023c] Services, A. W. (2023c). Aws codebuild.
- [Services, 2023d] Services, A. W. (2023d). Aws codedeploy.
- [Services, 2023e] Services, A. W. (2023e). Aws cognito.
- [Services, 2023f] Services, A. W. (2023f). Aws ec2.
- [Services, 2023g] Services, A. W. (2023g). Aws elastic beanstalk.
- [Services, 2023h] Services, A. W. (2023h). Aws elastic load balancing.
- [Services, 2023i] Services, A. W. (2023i). Aws identity and access managment.
- [Services, 2023j] Services, A. W. (2023j). Aws lambda.
- [Services, 2023k] Services, A. W. (2023k). Aws relational database service.
- [Services, 2023l] Services, A. W. (2023l). Aws s3.
- [Services, 2023m] Services, A. W. (2023m). Aws ses.
- [UI, 2023] UI, M. (2023). Material ui.
- [UTFSM, 2024] UTFSM, F. (2024). Sitio web fablab utfsm.