



UNIVERSIDAD DE TARAPACÁ
Universidad del Estado

Facultad de Ingeniería

Departamento de Ingeniería en Computación e Informática



Segunda Iteración: Pañollab **“Sistema gestión de inventario y préstamos”**

Autores:

Juan Yampara
Eduardo Apata
Esteban Monsálvez
Ivan Callasaya

Profesor: Victor Alfaro

Asignatura: Proyecto 3

Arica, Chile
05 de Noviembre de 2024



Índice de contenidos

Índice de contenidos.....	2
Índice de tablas.....	3
Índice de figuras.....	3
1. Introducción.....	4
2. Objetivo del proyecto.....	5
2.1. Objetivo general.....	5
2.2. Objetivos específicos.....	5
3. Definición del proyecto.....	5
3.1. Problema o necesidad.....	5
3.2. ¿Por qué es importante abordar este proyecto?.....	5
4. Desarrollo.....	6
4.1. Impact Mapping.....	6
4.2. RoadMap del proyecto.....	7
4.3. Product Backlog.....	8
4.3.1. Definición de Listo (DoR).....	8
4.3.2. Definición del Product Backlog.....	9
4.4. Tecnologías.....	12
7.1. Mecanismos de comunicación.....	14
8. Arquitectura del sistema.....	16
9. Primera iteración.....	17
9.1. Sprint Planning.....	17
9.1.1. Definición de perfil del Sprint.....	17
9.1.2. Sprint Backlog.....	17
9.1.3. Definición de historias de usuarios.....	18
9.1.4. Definición de hecho (DoD).....	19
9.2. Sprint Review.....	20
9.3. Sprint Retrospective.....	21
9.4. Estado del Sprint Backlog.....	22
9.5. Evidencia del trabajo.....	23
9.5.1. Recursos: Creación, visualización e eliminación.....	23
9.5.2. Usuarios: Creación y visualización.....	27
9.5.3. Categorías: Creación, visualización e eliminación.....	30
10. Segunda iteración.....	33
10.1. Sprint Planning.....	33
10.1.1. Definición de perfil del Sprint.....	33
10.1.2. Sprint Backlog.....	33
10.1.3. Definición de historias de usuarios.....	34
10.1.4. Definición de hecho (DoD).....	36
10.2. Sprint Review.....	36
10.3. Sprint Retrospective.....	36
10.4. Estado del Sprint Backlog.....	37



10.5. Evidencia del trabajo.....	38
11. Conclusión.....	39
12. Bibliografía.....	40

Índice de tablas

Tabla 1 : RoadMap del proyecto.....	9
Tabla 2 : Estimación de tallas.....	11
Tabla 3 : Product Backlog.....	11
Tabla 4 : Definición de perfil de Sprint.....	17
Tabla 5 : Sprint Backlog.....	17
Tabla 6 : Sprint Retrospective.....	22
Tabla 7: Sprint Backlog - Sprint 1.....	22
Tabla 8 : Funcionalidades implementadas en el Sprint.....	24

Índice de figuras

Figura 1: Impact Mapping.....	7
Figura 2: User Story Mapping.....	10
Figura 3: MoSCoW.....	12
Figura 2: Plan Release.....	16
Figura 4: “Controller recurso”.....	33
Figura 5: “función del servicio de recursos para crear recursos”.....	33
Figura 6: “Controller de crear recursos”.....	33
Figura 7: “Función del servicio de recursos: remove”.....	34
Figura 8: “Función del controller de recursos para eliminación”.....	34
Figura 9: Ver Recurso.....	35
Figura 10: Crear Recurso.....	35
Figura 11: Eliminar Recurso.....	36
Figura 12: “Funciones crear usuario y encontrar todos los usuarios”.....	36
Figura 13: “Funciones crear usuario y encontrar todos los usuarios”.....	37
Figura 14: “Funcionalidad crear usuarios desde el archivo service”.....	37
Figura 15: Ver Usuarios.....	38
Figura 16: Crear Usuarios.....	38
Figura 17: “Funciones del controller categorías”.....	39
Figura 18: “Creación de categoría desde el archivo service”.....	40
Figura 19 “Funciones para obtener todas las categorías y todos los recursos según categoría”.....	40
Figura 20: Ver Categorías.....	41
Figura 21: Crear Categorías.....	41
Figura 22: Crear Categorías.....	41



1. Introducción.

En el último año, la gestión de recursos tecnológicos ha presentado grandes dificultades para los ayudantes del departamento de informática. El sistema actual, diseñado para facilitar la administración de estos recursos, ha mostrado una serie de deficiencias que, en lugar de mejorar la eficiencia operativa, han complicado aún más las tareas diarias. Entre los problemas más destacados se encuentran la falta de optimización, que ralentiza tanto el proceso de préstamo como el de devolución de equipos, y la dificultad para mantener un control preciso de los inventarios.

Estos problemas han resaltado la urgente necesidad de implementar un nuevo sistema de gestión de inventarios y préstamos más efectivos. Este proyecto tiene como objetivo final implementar un sistema que corrija las deficiencias del sistema actual, optimizando la asignación y devolución de recursos tecnológicos, y garantizando un proceso más ágil y controlado.

Además, se busca reforzar la seguridad en el proceso de prestación y devolución de recursos. El nuevo sistema incorporará medidas avanzadas para proteger los datos personales de los estudiantes que utilicen el sistema, garantizando un entorno más seguro y confiable, eliminando las vulnerabilidades existentes y asegurando la integridad de la información en todo momento.

Durante la primera iteración del proyecto, se desarrollaron las funcionalidades consideradas prioritarias por el cliente, como la creación de productos, categorías y usuarios. Aunque no se completaron todas las historias del sprint, el cliente expresó su conformidad en el sprint review, lo que permitió avanzar hacia la segunda iteración con nuevas funcionalidades y sugerencias.

En el siguiente mes, durante la segunda iteración, se implementaron las mejoras sugeridas por el cliente, se rediseñó la interfaz de la aplicación web y se finalizaron las funcionalidades pendientes de la primera iteración. La iteración tiene como objetivo incorporar el sistema de préstamos mediante código QR, permitiendo que los estudiantes puedan registrar la solicitud y devolución de recursos, manteniendo un registro en el sistema. Además, se introducirá la carga masiva de estudiantes matriculados en la carrera de informática hasta la fecha, optimizando el registro de usuarios en el sistema.



2. Objetivo del proyecto.

2.1. Objetivo general.

Desarrollar un sistema de aplicación web que permita optimizar y mejorar el proceso de gestión de inventarios y préstamos de recursos para el departamento de informática.

2.2. Objetivos específicos.

- ❖ Analizar el sistema actual que posee el departamento de informática con el cual realiza préstamos y devoluciones.
- ❖ Definir los requerimientos funcionales del sistema de gestión de inventarios y préstamos mediante reuniones con el cliente.
- ❖ Implementar el sistema a través de iteraciones incrementales (Sprints) utilizando el marco de trabajo Scrum.

3. Definición del proyecto.

3.1. Problema o necesidad.

El departamento de informática de la Universidad de Tarapacá es responsable de gestionar los recursos tecnológicos fundamentales para el desarrollo académico y práctico de los estudiantes de la carrera de informática. Sin embargo, enfrenta serias dificultades debido a las limitaciones del sistema actual. Los ayudantes experimentan un sistema lento y poco eficiente para monitorear los recursos disponibles, lo que afecta su capacidad de gestión. Además, aunque se llevan registros de inventario, no se pueden realizar ciertas operaciones, como la creación de nuevos tipos de recursos, lo que limitaría una categorización más efectiva.

Entre las principales deficiencias se encuentran la lentitud en los procesos de préstamo y devolución de equipos, las dificultades para llevar un control eficiente del inventario y la ineficacia del sistema durante los préstamos solicitados por los estudiantes a lo cual está diseñado este servicio. Estos problemas han provocado pérdidas significativas de tiempo y recursos, impactando negativamente en la productividad tanto de los ayudantes de laboratorio como de los estudiantes de informática. Ante esta situación, es imprescindible implementar un nuevo sistema de gestión que optimice estos procesos, garantizando un control más ágil y seguro.

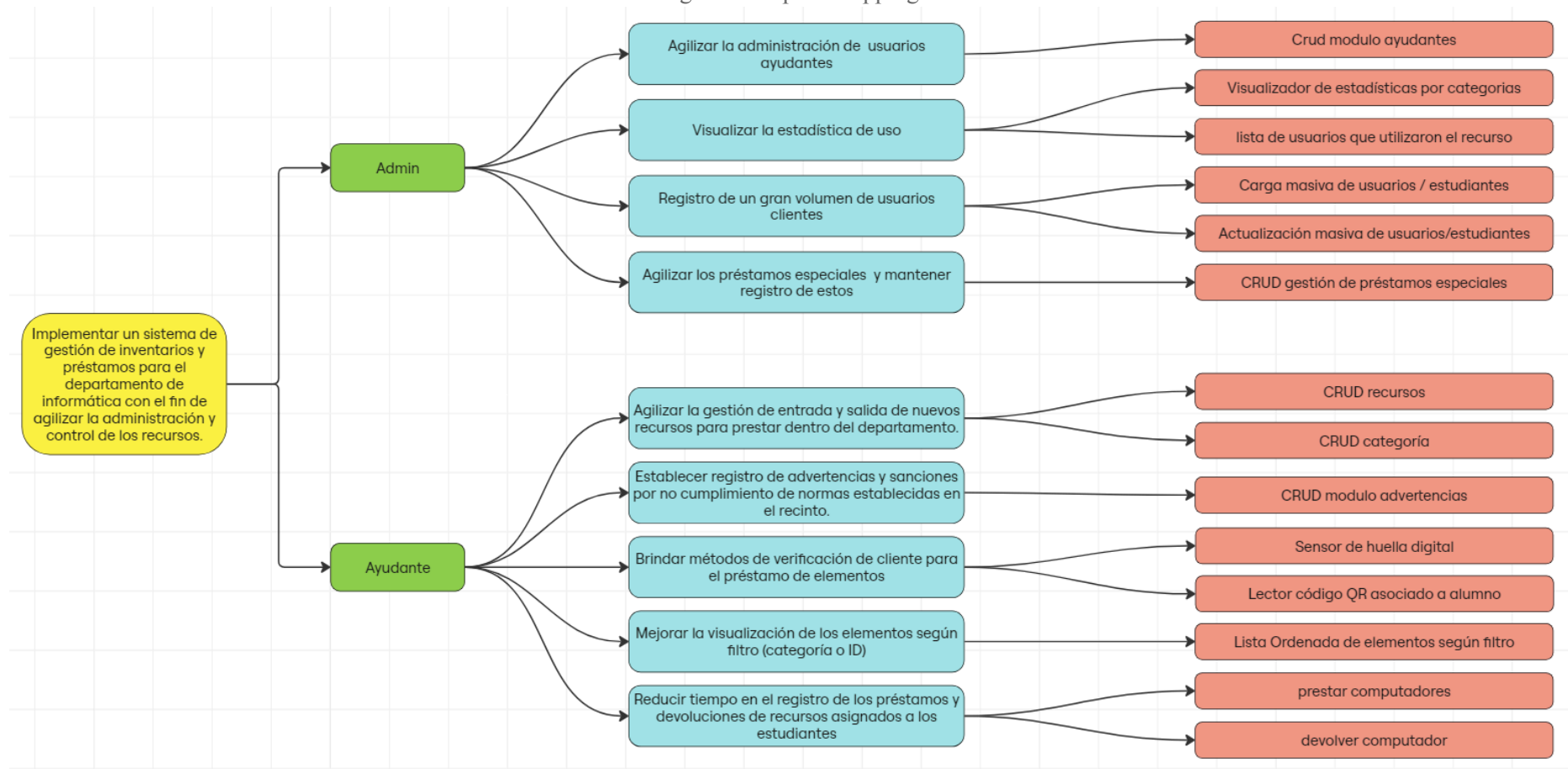
3.2. ¿Por qué es importante abordar este proyecto?

Abordar este proyecto es importante para superar las deficiencias del sistema actual en la gestión de recursos tecnológicos del departamento de informática. La implementación de un nuevo sistema de gestión permitirá optimizar los procesos de préstamo y devolución, mejorar el control de inventarios y reforzar la seguridad en la administración de recursos. Esto no solo reducirá las pérdidas de tiempo y recursos, sino que también beneficiará a los ayudantes al agilizar la prestación de equipos, y a los estudiantes al hacer más rápidas y eficientes las solicitudes y devoluciones de los recursos.

4. Desarrollo.

4.1. Impact Mapping.

Figura 1: Impact Mapping





4.2. RoadMap del proyecto.

Tabla 1 : RoadMap del proyecto

	Versiones		
	Primera	Segunda	Tercera
Fecha de lanzamiento	27/09/2024	25/10/2024	29/11/2024
Nombre de la versión	Versión 1.0	Versión 2.0	Versión 3.0
Objetivo	Proveer una funcionalidad básica para la gestión de recursos, ayudantes y categorías dentro del sistema, permitiendo a los usuarios registrar, ver, editar, y eliminar recursos y categorías, así como gestionar los ayudantes.	Mejorar la gestión de usuarios, ayudantes, y advertencias, incorporando nuevas funciones de verificación de identidad, carga masiva de usuarios y solicitudes de recursos mediante QR.	Optimizar la gestión de usuarios, ayudantes, y recursos mediante la incorporación de estadísticas avanzadas y autenticación.
Características de alto nivel necesarias para cumplir con el objetivo	<ol style="list-style-type: none">Gestión de Recursos:<ul style="list-style-type: none">Capacidad para que los usuarios registren, vean y eliminen los recursos.Registro de recursos prestados y devueltos.Gestión de Categorías:<ul style="list-style-type: none">Crear, editar, eliminar, y visualizar categorías de recursos.Gestión de Ayudantes:<ul style="list-style-type: none">Ver y editar la lista de ayudantes	<ol style="list-style-type: none">Verificación de identidad mediante código QR.Carga y actualización masiva de usuarios.Registro y eliminación de advertencias.Solicitud y devolución de recursos mediante código QR.Visualización de usuarios penalizados.	<ol style="list-style-type: none">Visualización de usuarios que han utilizado los recursos.Estadísticas sobre préstamos de inventario y categorías.Realización de préstamos excepcionales.Ingreso y cierre de turno con visualización de resultados.Verificación de identidad mediante huella digital.
Métricas para	Gestión de recursos:	Verificación de identidad con QR:	Estadísticas de préstamos:



<p>determinar si el objetivo se ha cumplido</p>	<ul style="list-style-type: none"> Los usuarios pueden registrar, ver, editar y eliminar recursos sin errores. Se han registrado y devuelto los recursos según las funcionalidades especificadas. <p>Gestión de categorías:</p> <ul style="list-style-type: none"> Se han creado, editado y eliminado categorías de manera efectiva. <p>Gestión de ayudantes:</p> <ul style="list-style-type: none"> Los ayudantes son visibles y editables sin inconvenientes. 	<ul style="list-style-type: none"> La verificación de identidad mediante código QR se realiza sin fallas y en el tiempo estipulado. Los usuarios han completado la verificación de identidad con éxito en todas las pruebas realizadas. <p>Carga masiva de usuarios:</p> <ul style="list-style-type: none"> La carga y actualización masiva de usuarios se han realizado sin errores ni interrupciones. <p>Gestión de advertencias:</p> <ul style="list-style-type: none"> Se han registrado y eliminado advertencias según los procedimientos establecidos. <p>Solicitudes de recursos:</p> <ul style="list-style-type: none"> Las solicitudes y devoluciones de recursos mediante código QR se gestionan de acuerdo con las especificaciones. 	<ul style="list-style-type: none"> Las estadísticas generadas reflejan con precisión el uso de recursos y préstamos. <p>Préstamos excepcionales:</p> <ul style="list-style-type: none"> Los préstamos excepcionales se gestionan conforme a las pautas definidas. <p>Experiencia de usuario:</p> <ul style="list-style-type: none"> Los usuarios han expresado satisfacción con el sistema durante las sesiones de retroalimentación. Se ha observado un uso continuo del sistema por parte de los usuarios sin problemas significativos.
--	---	---	---

4.3. Product Backlog.

4.3.1. Definición de Listo (DoR)

La definición de DoR debe cumplir con los siguientes criterios:



1. Asegurarse de que cada historia de usuario esté bien detallada y comprensible antes de que el equipo de desarrollo la seleccione una para trabajar en ella, esto significa que debe cumplir con la estructura de definición de HU, la cual es:
 - a. Como <usuario>, quiero <funcionalidad> para <beneficio>.
 - Donde **Usuario** se refiere a quien va ejecutar la funcionalidad.
 - La **Funcionalidad** específica que se va en dicha historia.
 - Y **Beneficio** indica con qué objetivo se busca implementar esta funcionalidad para el usuario.
2. Deben definirse criterios de aceptación específicos que indiquen cuándo la historia se considera completa, indicase puede permitir no se puede permitir en funcionalidad.
3. Además debe haber una justificación clara del valor que la historia aportará al usuario. Por último la aprobación del product owner es esencial para garantizar que la historia esté de acuerdo a sus necesidades.

4.3.2. Definición del Product Backlog.

Tabla 2 : Estimación de tallas

Estimación	
Tallas	Horas
XS	2 horas.
S	3 horas
M	6 horas.
L	9 horas.



Tabla 3 : Product Backlog

ID	Descripción	Abreviación	Evaluación	Prioridad	Release
1	Como ayudante quiero registrar el recurso devuelto para llevar un control de los recursos que han sido devueltos.	Registrar recurso devuelto.	M	MUST HAVE	1
2	Como ayudante quiero registrar el recurso prestado para mantener un registro de los recursos que se han prestado a los estudiantes.	Registrar recurso prestado.	M	MUST HAVE	1
3	Como ayudante quiero registrar un nuevo recurso para añadirlo al inventario y que estén disponibles para ser prestados.	Registrar recurso.	M	MUST HAVE	1
4	Como ayudante quiero ver todos los recursos devueltos para tener una visión clara de los recursos que ya han sido devueltos.	Ver recursos devueltos	XS	MUST HAVE	1
5	Como ayudante quiero ver todos los recursos prestados para conocer qué recursos están actualmente en préstamo y a que estudiantes se han asignado.	Ver todos los recursos prestados.	XS	MUST HAVE	1
6	Como ayudante quiero ver los recursos para acceder rápidamente a la lista completa de recursos disponibles en el inventario.	Ver los recursos	XS	MUST HAVE	1
7	Como ayudante quiero ver los detalles de un recurso devuelto para ver la información del estudiante quien ha devuelto el recurso.	Ver recurso devuelto	XS	MUST HAVE	1
8	Como ayudante quiero ver el seguimiento de un recurso prestado para rastrear al estudiante que está usando el recurso en ese momento.	Ver seguimiento.	XS	MUST HAVE	1
9	Como ayudante quiero editar mis recursos, para actualizar la información de los recursos cuando sea necesario.	Editar recursos.	M	MUST HAVE	1
10	Como ayudante quiero eliminar un recurso para retirar recursos obsoletos o dañados del inventario y mantener la lista de recursos actualizada.	Eliminar recursos.	S	MUST HAVE	1
11	Como ayudante quiero categorizar los recursos para organizar el inventario, de manera que sea más fácil encontrarlos y gestionarlos.	Categorizar los recursos.	M	MUST HAVE	1
12	Como ayudante quiero editar categorías para actualizar la información general de esa categoría.	Editar categoría	M	MUST HAVE	1



13	Como ayudante quiero crear categorías para permitir una mejor organización de los recursos.	Crear categorías.	S	MUST HAVE	1
14	Como ayudante quiero ver las categorías para revisar las categorías existentes y ver la información de esa categoría.	Ver las categorías	XS	MUST HAVE	1
15	Como ayudante quiero eliminar categorías para remover las categorías que ya no son relevantes o que no se utilizan.	Eliminar categoría.	S	MUST HAVE	1
16	Como estudiante quiero poder verificar mi identidad mediante un QR generado para agilizar el proceso de identificación y asegurar que solo yo pueda acceder a mis préstamos.	Verificación por QR	L	SHOULD HAVE	2
17	Como administración quiero poder ver la lista de ayudantes, para gestionar el equipo de ayudantes de cada semestre.	Ver lista ayudantes	XS	SHOULD HAVE	1
18	Como administración quiero poder editar los ayudantes para actualizar la información de los ayudantes.	Editar ayudantes	S	SHOULD HAVE	2
19	Como estudiante quiero solicitar un recurso mediante código QR para facilitar y agilizar el proceso de solicitud de préstamos.	Solicitar recurso por QR.	L	SHOULD HAVE	2
20	Como administración quiero poder crear los ayudantes para asignarlos al sistema durante el semestre.	Registrar ayudantes	M	SHOULD HAVE	1
21	Como estudiante quiero devolver un recurso mediante código QR para hacer el proceso de devolución más rápido.	Devolver recurso mediante QR	L	SHOULD HAVE	2
22	Como administración quiero poder eliminar los ayudantes para retirar a los ayudantes que ya no forman parte durante los semestres.	Eliminar Ayudante	S	SHOULD HAVE	2
23	Como administración quiero poder ingresar los usuarios por medio de una carga masiva para tener la información de los estudiantes matriculados de la carrera por semestre.	Carga masiva de usuarios	L	SHOULD HAVE	2
24	Como administración quiero poder actualizar a los usuarios por medio de una carga masiva para mantener actualizada la información de los estudiante matriculados por cada semestre.	Actualizar la carga masiva	L	SHOULD HAVE	2
26	Como ayudante quiero registrar advertencias para tener un registro de los estudiantes que han tenido	Registrar advertencias	S	COULD HAVE	2



	un comportamiento indebido con el recurso se le ha prestado.				
28	Como ayudante quiero eliminar advertencias para corregir errores o retirar advertencias que ya no sean relevantes.	Eliminar advertencias	S	COULD HAVE	2
31	Como administración quiero visualizar a los usuarios penalizados para tomar una decisión en ser vetado para prestarle recursos.	Ver usuarios penalizados	XS	COULD HAVE	2
32	Como ayudante quiero iniciar sesión e iniciar turno según se me asignó para comenzar mi jornada como ayudante.	Autenticación o inicio de sesión	L	WON'T HAVE	3
33	Como administración quiero poder ver las estadísticas de préstamos de inventario para analizar el uso de los recursos.	Ver estadísticas de recursos	L	WON'T HAVE	3
34	Como ayudante quiero cerrar sesión y visualizar el resultado del cierre de turno para finalizar mi jornada como ayudante.	Cerrar session	L	WON'T HAVE	3
35	Como administración quiero poder ver las estadísticas de las categorías para analizar la popularidad y uso de diferentes categorías de recursos.	Ver estadísticas de categorías	L	WON'T HAVE	3
36	Como administración quiero poder realizar préstamos excepcionales a los usuarios para gestionar situaciones especiales donde los estudiantes necesiten recursos de manera semestral.	Realizar préstamos especiales	L	WON'T HAVE	3
37	Como administración quiero visualizar los estudiantes que han utilizado los recursos para saber la información del estudiante que ha utilizado ese recurso	Ver usuario por préstamo	M	WON'T HAVE	3
38	Como estudiante quiero poder verificar mi identidad mediante la huella digital para asegurar mi identidad de una manera mucho más segura que por código QR.	Verificar mediante huella digital	M	WON'T HAVE	3



4.4. Tecnologías.

1. Entornos de desarrollo.

Visual Studio Code: Visual Studio Code es un editor de código ofrece soporte para múltiples lenguajes de programación, como JavaScript y TypeScript, y cuenta con una amplia gama de extensiones que mejoran sus funcionalidades. Se destaca por su integración con Git, depuración, autocompletado inteligente (IntelliSense), y soporte para desarrollo web tanto en frontend como backend.

2. Control de versiones y colaboración.

Git: Sistema de control de versiones distribuido que permite registrar y gestionar los cambios realizados en el código fuente de un proyecto. Facilita la colaboración entre desarrolladores al permitir trabajar en ramas separadas, fusionar cambios y restaurar versiones anteriores del código de forma eficiente.

GitHub: Plataforma en la nube que utiliza Git para alojar repositorios de código. GitHub facilita la colaboración en equipo mediante la revisión de código, el seguimiento de tareas y la creación de solicitudes de fusión (pull requests).

3. Lenguajes de Programación.

JavaScript: Lenguaje de programación fundamental para el desarrollo web. Permite crear interacciones dinámicas en el navegador, manejar eventos y realizar operaciones asincrónicas. Se utiliza tanto en el lado del cliente (frontend) como en el servidor (backend) a través de entornos como Node.js.

TypeScript: Superconjunto de JavaScript que añade tipado estático y otras características orientadas a objetos. Se compila a JavaScript puro y mejora la robustez y mantenibilidad del código, facilitando el desarrollo de aplicaciones grandes y complejas.

4. Frameworks y Librerías.

Angular 18: Framework de desarrollo de aplicaciones web de código abierto creado por Google. Está basado en TypeScript y se utiliza para construir aplicaciones web dinámicas y de una sola página (SPA) que son altamente escalables y mantenibles. Para este proyecto, se utilizará la versión 18 de angular.

NestJS: Framework para el desarrollo de aplicaciones del lado del servidor (backend) basado en Node.js y TypeScript. NestJS se inspira en los principios de diseño de Angular y utiliza una arquitectura modular y orientada a objetos, similar a la del framework Angular, pero adaptada para el desarrollo backend.

5. Bases de Datos y Herramientas de Gestión

MySQL Server: Es el sistema de gestión de bases de datos relacional (RDBMS) que almacena y maneja las bases de datos. Es el software en el que se encuentran los datos y donde se ejecutan las consultas SQL.



MySQL Workbench: Es una herramienta de interfaz gráfica de usuario (GUI) que facilita la administración de bases de datos MySQL. Es un cliente que interactúa con MySQL Server.

6. Herramientas de Colaboración y Gestión de Proyectos.

Miro: Plataforma de colaboración en línea que ofrece una pizarra digital para trabajar en tiempo real. Es ideal para la creación de diagramas, esquemas y planificación.

Google Drive: Es un servicio de almacenamiento en la nube que permite guardar y compartir archivos en línea, ideal para colaborar en informes y presentaciones con el equipo de trabajo.

Trello: Herramienta de gestión de proyectos basada en tableros visuales. Organiza tareas y actividades a través de tarjetas y listas, facilitando la visualización del progreso y la colaboración en equipo.

7. Proveedores de Servidores.

Amazon Web Services (AWS): Tecnología de computación en la nube que ofrece una amplia gama de servicios para el alojamiento, procesamiento, almacenamiento y bases de datos. AWS permite desplegar y escalar aplicaciones sin necesidad de administrar servidores físicos, facilitando la infraestructura como servicio (IaaS) y plataformas.

7.1. Mecanismos de comunicación.

1. Reuniones presenciales con el cliente.

Las reuniones presenciales serán el principal mecanismo para interactuar con el cliente del producto. En estas reuniones, se establecerán las ideas y necesidades del cliente, permitiendo una comprensión clara de sus expectativas y objetivos para el proyecto. Además, estas sesiones servirán para revisar el progreso del trabajo, recibir retroalimentación directa y ajustar el enfoque del desarrollo según sea necesario.

2. Reuniones en clase.

Las reuniones en sala son el principal mecanismo de comunicación para seguir el avance del proyecto. Se realizan dos veces por semana y permiten al equipo evaluar el progreso, ofrecer asistencia si es necesario, y resolver cualquier discrepancia. Aunque la asistencia no siempre es obligatoria debido a compromisos individuales, cada miembro es responsable de mantenerse informado sobre el desarrollo del proyecto.

3. Discord.

Discord será la segunda herramienta principal de comunicación del equipo, utilizada para realizar reuniones y coordinar actividades del proyecto. La mayoría del personal está disponible en esta plataforma, lo que facilita la organización de reuniones y la discusión de temas relevantes. El Product Owner es responsable de convocar estas reuniones en horarios que sean convenientes para todos los miembros del equipo.

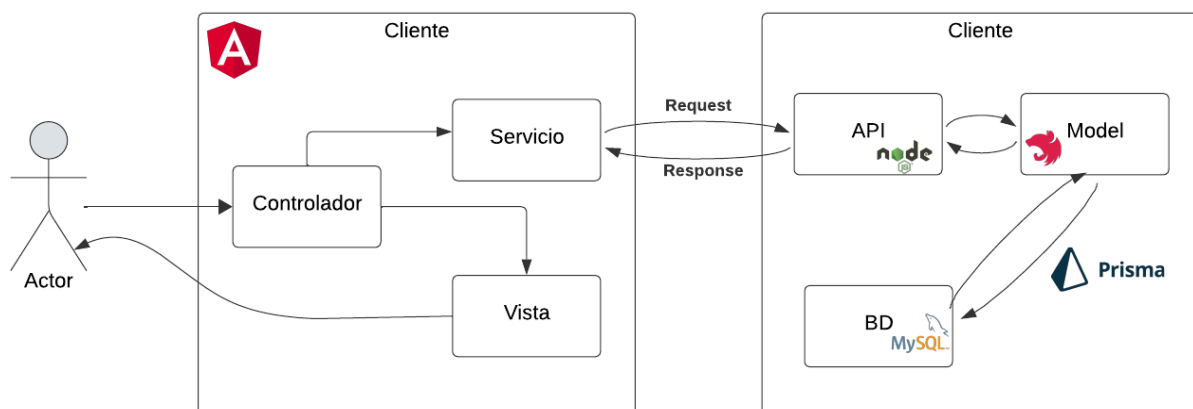
4. Whatsapp.

WhatsApp será una herramienta clave para la comunicación rápida y eficiente del equipo. Se utilizará principalmente para notificar sobre reuniones, informar ausencias, y comunicar situaciones urgentes o imprevistos. El product owner se encargará de enviar y recibir los mensajes, asegurándose de que cualquier falta esté justificada y comunicada a tiempo.

8. Arquitectura del sistema.

El sistema está basado en una arquitectura cliente-servidor. El frontend está desarrollado en Angular, lo que permite una interfaz de usuario interactiva y dinámica. Angular maneja las interacciones de los usuarios, gestiona las vistas y realiza peticiones HTTP al servidor. El backend está construido con NestJS, un framework basado en Node.js y TypeScript. El backend se encarga de procesar las solicitudes recibidas del frontend, interactuar con la base de datos y gestionar la lógica de negocio.

La base de datos utilizada es MySQL, que almacena toda la información sobre recursos, usuarios, préstamos y devoluciones. El backend utiliza Prisma como ORM para interactuar con la base de datos, facilitando la gestión de las consultas SQL y las operaciones sobre los datos.





9. Primera iteración.

9.1. Sprint Planning.

9.1.1. Definición de perfil del Sprint.

Tabla 4 : Definición de perfil de Sprint

Primer Sprint “PañolLab”	
ID	001
Participantes	Juan Yampara Esteban Monsalves Eduardo Apata Ivan Callasaya
Fecha Inicio	30 de Agosto del 2024
Fecha término estimada	27 de Septiembre del 2024
Duración del Sprint	4 Semanas
Objetivo	Desarrollar la primera versión del sistema que priorice la gestión de recursos y préstamos. También realizará la entrega de un frontend atractivo y funcional. El propósito a grandes rasgos, es brindar un avance amigable y funcional para el cliente.

9.1.2. Sprint Backlog.

Tabla 5 : Lista de tareas extraídas Product Backlog

ID	Descripción de la historia de usuario	Evaluación
2	Como ayudante quiero registrar un nuevo recurso para añadirlo al inventario y que estén disponibles para ser prestados.	M
6	Como ayudante quiero ver los recursos para acceder rápidamente a la lista completa de recursos disponibles en el inventario.	XS
9	Como ayudante quiero editar mis recursos, para actualizar la información de los recursos cuando sea necesario.	M
10	Como ayudante quiero eliminar un recurso para retirar recursos obsoletos o dañados del inventario y mantener la lista de recursos actualizada.	S
11	Como ayudante quiero categorizar los recursos para organizar el inventario, de manera que sea más fácil encontrarlos y gestionarlos.	M



13	Como ayudante quiero crear categorías para permitir una mejor organización de los recursos.	S
14	Como ayudante quiero ver las categorías para revisar las categorías existentes y ver la información de esa categoría.	XS
12	Como ayudante quiero editar categorías para actualizar la información general de esa categoría.	M
15	Como ayudante quiero eliminar categorías para remover las categorías que ya no son relevantes o que no se utilizan.	S
1	Como ayudante quiero registrar el recurso devuelto para llevar un control de los recursos que han sido devueltos.	M
4	Como ayudante quiero ver todos los recursos devueltos para tener una visión clara de los recursos que ya han sido devueltos.	XS
8	Como ayudante quiero ver el seguimiento de un recurso prestado para rastrear al estudiante que está usando el recurso en ese momento.	XS
2	Como ayudante quiero registrar el recurso prestado para mantener un registro de los recursos que se han prestado a los estudiantes.	M
5	Como ayudante quiero ver todos los recursos prestados para conocer qué recursos están actualmente en préstamo y a que estudiantes se han asignado.	XS
7	Como ayudante quiero ver los detalles de un recurso devuelto para ver la información del estudiante quien ha devuelto el recurso.	XS
20	Como administración quiero poder crear los ayudantes para asignarlos al sistema durante el semestre.	M

9.1.3. Definición de historias de usuarios.

- **Registrar un nuevo recurso (M)**

Subtareas:

Crear formulario para registrar recursos.

Conectar el formulario al backend para agregar a la base de datos.

Tiempo estimado: 6 horas.

Responsable: Juan Yampara

Criterios de aceptación: El recurso debe quedar registrado con todos sus detalles (nombre, categoría, estado) en la base de datos.

- **Ver los recursos (XS)**

Subtareas:

Crear lista que muestre todos los recursos.

Filtrar recursos por categoría.

Tiempo estimado: 2 horas.

Responsable: Esteban Monsálvez

Criterios de aceptación: Debe mostrar la lista completa de recursos con opción de filtrar.



- **Categorizar los recursos (M)**

Subtareas:

Crear funcionalidad para asignar categorías a recursos.

Tiempo estimado: 6 horas.

Responsable: Juan Yampara

Criterios de aceptación: Cada recurso debe poder ser categorizado correctamente.

- **Crear categorías (S)**

Subtareas:

Desarrollar un formulario para la creación de nuevas categorías.

Conectar con backend para almacenar categorías en la base de datos.

Tiempo estimado: 3 horas.

Responsable: Eduardo Apata

Criterios de aceptación: El sistema debe permitir la creación de nuevas categorías.

- **Editar recursos (M)**

Subtareas:

Crear funcionalidad para modificar los detalles de un recurso.

Actualizar información en la base de datos.

Tiempo estimado: 6 horas.

Responsable: Ivan Callasaya

Criterios de aceptación: La modificación debe reflejarse correctamente en el sistema y la base de datos.

- **Eliminar recursos (S)**

Subtareas:

Implementar la opción para eliminar un recurso del inventario.

Tiempo estimado: 3 horas.

Responsable: Esteban Monsálvez

Criterios de aceptación: El recurso eliminado no debe aparecer en la lista de inventario.

- **Registrar el recurso devuelto (M)**

Subtareas:

Crear formulario de devolución de recursos.

Actualizar la base de datos con el estado del recurso devuelto.

Tiempo estimado: 6 horas.

Responsable: Iván Callasaya

Criterios de aceptación: El recurso debe marcarse como devuelto con fecha y estudiante asociado.



9.1.4. Definición de hecho (DoD).

Para la definición de "hecho", se han establecido ciertos requisitos que cada tarea debe cumplir para considerarse completada correctamente:

1. Criterios de Aceptación

Cada funcionalidad debe cumplir con los criterios de aceptación definidos para ser considerada completa.

2. Desempeño y Escalabilidad

La aplicación ha pasado pruebas de rendimiento y escalabilidad, garantizando velocidad y capacidad ante cargas diversas.

3. Pruebas de Validación

La aplicación debe pasar por una exhaustiva prueba de validación, asegurando que todas las funcionalidades están libres de errores de código y funcionan según lo esperado.

4. Documentación:

El código de la aplicación debe estar documentado tanto para el proyecto del frontend como del lado backend, para que luego otro usuario que quiera actualizar el sistema tenga conocimiento de cada funcionalidad.

9.2. Sprint Review.

El día 26 de septiembre del 2024 se realizó la sprint Review en conjunto con el cliente Don Humberto Urrutia. Durante la reunión se mostraron las funcionalidades desarrolladas durante el sprint, también se recibió la retroalimentación del cliente y por último se agregaron nuevas funcionalidades a considerar en el próximo sprint.

1. Diseño de interfaz e interacción

- a. El cliente encontró que el diseño estaba bien realizado, sin embargo mencionó que el color puede ser más relacionado con el departamento, recomendando el color azul.
- b. La interacción que se demostró en la reunión fue de su agrado, sencilla e intuitiva

2. Funcionalidades esperadas

- a. Si bien, se esperaba la realización de los préstamos, al momento de la ejecución no se pudieron realizar debido a la falta de la funcionalidad de crear estudiantes. Por lo que el cliente demostró cierta inconformidad en ese aspecto del avance del sistema.
- b. Por otro lado, la gestión de recursos, categorías y usuarios fueron puntos en los cuales el cliente demostró conformidad.

3. Problemas enfrentados

- a. Durante el desarrollo del proyecto se enfrentaron a situaciones no favorables para el equipo, siendo una de ellas a causa de la salud de un integrante.



- b. También como equipo de desarrollo nos enfrentamos a la falta de comunicación. coordinación de las tareas y progreso, siendo estos aspectos los causantes del resultado en cuestión.

4. Retos a futuro

- a. Como equipo desarrollador nos planteamos el despliegue del sistema con la funcionalidad más esencial del mismo; la gestión de préstamos. Creemos actualmente que, con la retroalimentación entregada por el cliente y el análisis de nuestros resultados, podemos entregar un trabajo que genere la satisfacción de nuestro cliente.
- b. Prestaremos mejor atención al progreso de las tareas asignadas a cada parte del equipo, manteniendo una comunicación constante y efectiva para no generar presión excesiva ni puntos ciegos durante el desarrollo de las tareas.

9.3. Sprint Retrospective.

Para una retroalimentación grupal se ha optado por realizar las siguientes preguntas:

- ¿Qué hicimos bien?
- ¿Qué podemos mejorar?
- ¿Qué impedimentos surgieron?

Tabla 6 : Sprint Retrospective

¿Qué hicimos bien?	<ul style="list-style-type: none">• Correcto diseño de la base de datos• Interfaz amigable con el usuario• Mantener comunicación con el cliente• Despliegue del sistema para probar funcionalidades en diferentes equipos.
¿Qué podemos mejorar?	<ul style="list-style-type: none">• Comunicación entre equipo (estado de tareas)• Frecuencia y constancia de trabajo• Uso de herramientas para scrum• Definición de fechas límites
¿Qué impedimentos surgieron?	<ul style="list-style-type: none">• Cuello de botella debido a la modificación del esquema para el backend y frontend al momento de desplegar la base de datos.• Enfermedad en los integrantes de los equipos



9.4. Estado del Sprint Backlog.

Tabla 7: Sprint Backlog - Sprint 1

Historia de Usuario	Tarea / Actividad	Responsable	Estimación de Esfuerzo (Puntos)	Estado
Como usuario quiero registrar un nuevo recurso	Implementar formulario de registro	Juan Yampara	M	Completado
Como usuario quiero ver los recursos	Crear vista de recursos disponibles	Ivan Callasaya	XS	Completado
Como usuario quiero editar mis recursos	Habilitar función de edición de recursos	Esteban Monsalvez	M	Completado
Como usuario quiero eliminar un recurso	Añadir botón y lógica de eliminación	Iván Callasaya	S	Completado
Como usuario quiero registrar el recurso devuelto	Implementar lógica de devolución	Eduardo Apata Juan Yampara	M	Incompleto
Como usuario quiero ver todos los recursos devueltos	Crear vista de recursos devueltos	Eduardo Apata Iván Callasaya	XS	Completado
Como usuario quiero ver los detalles de un recurso devuelto	Implementar detalles de los recursos devueltos	Eduardo Apata	XS	Incompleto
Como usuario quiero registrar el recurso prestado	Implementar registro de préstamos	Eduardo Apata	M	Incompleto
Como usuario quiero ver todos los recursos prestados	Crear vista de recursos prestados	Eduardo Apata Juan Yampara	XS	Incompleto
Como usuario quiero ver el seguimiento de un recurso prestado	Implementar sistema de seguimiento de préstamos	Juan Yampara Eduardo Apata Iván Callasaya	XS	Incompleto
Como usuario quiero crear categorías	Añadir funcionalidad para crear categorías	Eduardo Apata Juan Yampara	S	Completado



Como usuario quiero editar categorías	Habilitar edición de categorías	Iván Callasaya	XS	Completado
Como usuario quiero ver las categorías	Crear vista para listar categorías	Iván Callasaya Eduardo Apata	M	Completado
Como usuario quiero eliminar categorías	Añadir lógica de eliminación de categorías	Esteban Monsalvez	XS	Completado
Como usuario quiero categorizar los recursos	Permitir asignación de categorías a recursos	Esteban Monsalvez Juan Yampara	S	Completado
Como usuario quiero poder crear los ayudantes	Implementar creación de usuarios 'ayudante'	Juan Yampara Eduardo Apata	M	Completado
Como usuario quiero poder ver la lista de ayudantes	Crear vista de lista de ayudantes	Eduardo Apata Iván Callasaya	XS	Completado

9.5. Evidencia del trabajo.

9.5.1. Recursos: Creación, visualización e eliminación

Backend:

Para el backend el funcionamiento de los controladores es solicitar al servicio gestionar la lógica necesaria para poder crear, visualizar o eliminar un recurso mediante consultas a la base de datos.

Figura 4: "Controller recurso"

```
@Controller('recursos')
export class RecursosController {
  constructor(private readonly recursosService: RecursosService) {}
```

En esta función del servicio se crea un nuevo recurso y se envía un objeto del tipo ResponseDto para que el frontend pueda confirmar la creación correcta.

Figura 5: "función del servicio de recursos para crear recursos"

```
async create(createRecurso: CreateRecursoDto) : Promise<ResponseDto<recurso>>{
  try {
    const newRecurso = await this.databaseService.recurso.create(
      {data : createRecurso});

    const response : ResponseDto<recurso> = {
      statusCode : HttpStatus.CREATED,
      message: 'Recurso creado con exito',
      data: newRecurso,
    }

    return response
  } catch (error){
    throw new HttpException('Error al crear el recurso', HttpStatus.BAD_REQUEST);
  }
}
```

Para la creación de un recurso, el controller recibe un objeto del tipo recurso, esto se hace mediante un verbo http Post desde el frontend.

Figura 6: "Controller de crear recursos"

```
@Post()
async create(@Body() createRecurso: CreateRecursoDto) : Promise<ResponseDto<CreateRecursoDto>>{
  const newRecurso = await this.recursosService.create(createRecurso)
  return newRecurso;
}
```

Con respecto a la eliminación, primero se valida si existe o no el recurso a eliminar, en caso de no existir se lanza un error debido a que el recurso no existe, esto con la finalidad que el frontend pueda mostrar los mensajes de error sin que el sistema se caiga.

Figura 7: “Función del servicio de recursos: remove”

```
async remove(id: string) : Promise<ResponseDto<recurso>>{
  try {
    if(!await this.databaseService.recurso.findUnique({
      where : { id_uta : id,}
    })) {
      throw new HttpException('Recurso a eliminar no existe', HttpStatus.BAD_REQUEST);
    }

    const deleteRecurso = await this.databaseService.recurso.delete({
      where : {id_uta : id},
    });

    const response : ResponseDto<recurso> = {
      statusCode : HttpStatus.OK,
      message : 'Recurso borrado con exito',
      data: deleteRecurso
    }
    return response
  } catch(error) {
    throw new HttpException('Error, no se pudo borrar el recurso', HttpStatus.BAD_REQUEST)
  }
}
```

Para poder eliminar un recurso, se necesita su Id, el parámetro ‘:id’ contenido en decorador Delete, nos indica que, junto a la ruta del controller que es ‘recursos’, se recibirá un identificador del recurso para utilizarlo en la eliminación.

Figura 8: “Función del controller de recursos para eliminación”

```
@Delete('/:id')
remove(@Param('id') id: string) {
  return this.recursosService.remove(id);
}
```




Las funciones para visualizar por recursos sigue la misma estructura en el archivo controller como en el service de recursos.



Frontend: Gestión de Recurso (Agregar, Editar, Eliminar).

Descripción: Interfaz de usuario que muestra todos los recursos que están en el sistema. Se pueden agregar tanto recursos, como editarlos y eliminarlos, cada uno con su formulario y con su respectiva validación.

Figura 9: Ver Recurso

Recursos										Agregar Recurso
ID UTA	ID DICI	ID Categoría	Modelo	Marca	Fecha Ingreso	Estado	Ubicación	Descripción	Acciones	
123	123	4	123	123	26/09/2024	Disponible	123	123	  	

Crear Recurso

ID UTA*

ID DICI*

Marca*

Modelo*

Ubicación*

Categoría*

Descripción*

Cancelar

Crear



ID Categoria	Modelo	Marca	Fecha Ingreso	Estado	Ubicacion
4	123	123	26/09/2024	Disponible	123

Borrar Recurso

Esta seguro que desea borrar el recurso con ID UTA: 123 ?

Cancelar Eliminar

9.5.2. Usuarios: Creación y visualización

Backend:

Figura 12: “Funciones crear usuario y encontrar todos los usuarios”

```
@Post()
create(@Body() createUsuarioDto: CreateUsuarioDto) {
  return this.usuariosService.create(createUsuarioDto);
}

@Get()
findAll() {
  return this.usuariosService.findAll();
}
```

Figura 13: “Funciones crear usuario y encontrar todos los usuarios”

```
async create(createUsuario: CreateUsuarioDto) : Promise<ResponseDto<usuario>>{
  try {
    const usuario = await this.databaseService.usuario.create(
      { data: createUsuario }
    );

    if(createUsuario.rol == TiposUsuario.administrador){
      await this.databaseService.admin.create({
        data: { id_usuario : usuario.id_usuario }
      })
    } else if(createUsuario.rol == TiposUsuario.ayudante){
      await this.databaseService.ayudante.create({
        data: { id_usuario: usuario.id_usuario }
      })
    } else{
      throw new HttpException('Rol no válido', HttpStatus.BAD_REQUEST);
    }

    const response : ResponseDto<usuario> = {
      statusCode : HttpStatus.CREATED,
      message: 'Usuario creado con éxito',
      data : usuario,
    };

    return response;
  } catch(error){
    throw new HttpException('Error al crear usuario', HttpStatus.BAD_REQUEST);
  }
}
```

Figura 14: “Funcionalidad crear usuarios desde el archivo service”

```
findAll() {  
  try {  
    return this.databaseService.usuario.findMany();  
  } catch (error) {  
    throw new HttpException('Error al cargar todos los usuarios', HttpStatus.BAD_REQUEST);  
  }  
}
```

Frontend: Gestión de Usuarios (Agregar).

Descripción: Interfaz de usuario que muestra a los usuarios dentro del sistema, hasta el momento solo se pueden agregar usuarios pero tiene problemas al momento de ingresar a la base de datos, aunque el formulario y los datos que se envían al backen están correctamente.

Figura 15: Ver Usuarios

Usuarios Agregar Usuarios						
id_usuario	Rut	Nombre	Apellido	Usuario	Correo	Acciones
1	asd	asd	asd	asd	asd	  
2	111	11	11	11	111	  
3	33	33	33	33	33	  
4	21012321	Jorge	Perez	jorgep	Jorge@gmail.com	  

Figura 16: Crear Usuarios

Crear Usuarios

Rut*

Usuario*

Nombre*

Apellido*

Correo*

Contraseña*

Rol*

Cancelar

Crear

9.5.3. Categorías: Creación, visualización e eliminación

Backend:

Figura 17: “Funciones del controller categorías”

```
@Controller('categorias')
export class CategoriasController {
  constructor(private readonly categoriasService: CategoriasService) {}

  @Post()
  async create(@Body() createCategoria: CreateCategoriaDto) : Promise<ResponseDto<CreateCategoriaDto>> {
    return await this.categoriasService.create(createCategoria)
  }

  @Get()
  findAll() {
    return this.categoriasService.findAll();
  }

  @Get('/:id/recursos')
  getAllRecursosByCategoria(@Param('id') id: string){
    return this.categoriasService.getAllRecursoByCategoria(+id);
  }

  @Get('/:id')
  findOne(@Param('id') id: string) {
    return this.categoriasService.findOne(+id);
  }

  @Patch('/:id')
  update(@Param('id') id: string, @Body() updateCategoriaDto: UpdateCategoriaDto) {
    return this.categoriasService.update(+id, updateCategoriaDto);
  }

  @Delete('/:id')
  remove(@Param('id') id: string) {
    return this.categoriasService.remove(+id);
  }
}
```

Figura 18: “Creación de categoría desde el archivo service”

```
async create(createCategoria: CreateCategoriaDto) : Promise<ResponseDto<categoria>> {
  try {
    const newcategoria = await this.databaseService.categoria.create({
      data : {
        fecha_creacion : createCategoria.fecha_creacion,
        nombre_categoria : createCategoria.nombre_categoria
      },
    })

    const response : ResponseDto<categoria> = {
      statusCode : HttpStatus.CREATED,
      message : 'Categoria creada con exito',
      data: newcategoria
    }
    return response
  } catch(error){
    throw new HttpException('Error al crear la categoria', HttpStatus.BAD_REQUEST)
  }
}
```

Figura 19 “Funciones para obtener todas las categorías y todos los recursos según categoría”

```
async findAll() : Promise<categoria[]>{
  try {
    return await this.databaseService.categoria.findMany();
  } catch(error){
    throw new HttpException('Error al mostrar las categorias', HttpStatus.BAD_REQUEST);
  }
}

async getAllRecursoByCategoria(id: number): Promise<recurso[]>{
  try {
    const findCategoria = await this.databaseService.categoria.findUnique({
      where : { id_categoria: id },
      include: { recurso: true }
    });

    const recursos : recurso[] = findCategoria.recurso;
    return recursos;
  } catch(error){
    throw new HttpException('Error al obtener los recursos según categoria', HttpStatus.BAD_REQUEST);
  }
}
```



Frontend: Gestión de categorías (Agregar, editar y eliminar) .

Descripción: Interfaz de usuario de como se muestran las categorías en una tabla, y con las opciones tanto para ver, editar y eliminar dicha categoría, luego estas categorías serán desplegadas como menú de opción cuando se crean recursos.

Figura 20: Ver Categorías

Categorías			
Agregar Categoría			
categoria_id	Nombre	Fecha de creacion	Acciones
4	Computadores	26/09/2024 - 04:20:35	  
5	Meta Quest	26/09/2024 - 04:21:47	  
7	Monitores	26/09/2024 - 20:07:06	  

Nombre Fecha de creacion

Computadores

Meta Quest

Monitores

Crear Categoría

Nombre*

Cancelar Crear

Eliminar Categoría

Esta seguro que desea borrar el recurso con ID Categoría: 4 ?

Cancelar Eliminar



10. Segunda iteración.

10.1. Sprint Planning.

10.1.1. Definición de perfil del Sprint.

Tabla 8 : Definición de perfil de Sprint 2

Segundo Sprint “PañolLab”	
ID	002
Participantes	Juan Yampara Esteban Monsalves Eduardo Apata Ivan Callasaya
Fecha Inicio	28 de Septiembre del 2024
Fecha término estimada	30 de Octubre del 2024
Duración del Sprint	4 Semanas
Objetivo	Desarrollar la segunda versión del sistema que priorice la carga masiva y verificación de identidad mediante QR y realizar la entrega de un frontend rediseñado y funcional.

10.1.2. Sprint Backlog.

Tabla 9 : Sprint Backlog

ID	Descripción de la historia de usuario	Evaluación
16	Como estudiante quiero poder verificar mi identidad mediante un QR generado para agilizar el proceso de identificación y asegurar que solo yo pueda acceder a mis préstamos.	L X
18	Como administración quiero poder editar los ayudantes para actualizar la información de los ayudantes.	S X
19	Como estudiante quiero solicitar un recurso mediante código QR para facilitar y agilizar el proceso de solicitud de préstamos.	L X
21	Como estudiante quiero devolver un recurso mediante código QR para hacer el proceso de devolución más rápido.	L X
22	Como administración quiero poder eliminar los ayudantes para retirar a los ayudantes que ya no forman parte durante los semestres.	S X
23	Como administración quiero poder ingresar los usuarios por medio de una	L



	carga masiva para tener la información de los estudiantes matriculados de la carrera por semestre.	
24	Como administración quiero poder actualizar a los usuarios por medio de una carga masiva para mantener actualizada la información de los estudiante matriculados por cada semestre.	L
26	Como ayudante quiero registrar advertencias para tener un registro de los estudiantes que han tenido un comportamiento indebido con el recurso se le ha prestado.	S X
28	Como ayudante quiero eliminar advertencias para corregir errores o retirar advertencias que ya no sean relevantes.	S X
31	Como administración quiero visualizar a los usuarios penalizados para tomar una decisión en ser vetado para prestarle recursos.	XS X

10.1.3. Definición de historias de usuarios.

- **Verificación de identidad(L)**

Subtareas:

Implementar el uso de cámara web

Implementar un lector de código qr

Tiempo estimado: 9 horas.

Responsable: Esteban Monsalvez, Juan Yampara

Criterios de aceptación: El recurso debe verificar los datos guardados en la base de datos

- **Editar ayudantes (S)**

Subtareas:

Visualizar ayudantes

Tiempo estimado: 3 horas.

Responsable: Ivan Callasaya

Criterios de aceptación: Debe mostrar la información a editar y la información actualizada.

- **Solicitar recurso mediante QR(L)**

Subtareas:

Implementar el uso de cámara web

Implementar un lector de código qr

Tiempo estimado: 9 horas.

Responsable: Eduardo Apata, Ivan Callasaya

Criterios de aceptación: El usuario se pueda visualizar en el recurso de préstamos.



- **Devolver recurso mediante QR(L)**

Subtareas:

Implementar el uso de cámara web

Implementar un lector de código qr

Tiempo estimado: 9 horas.

Responsable: Eduardo Apata, Esteban Monsalvez

Criterios de aceptación: El usuario ya no aparece en el recurso préstamos y aparece la fecha de devolución de este.

- **Eliminar ayudantes (S)**

Subtareas:

Desactivar el perfil del usuario.

Tiempo estimado: 3 horas.

Responsable: Ivan Callasaya

Criterios de aceptación: El ayudante ya no puede iniciar sesión.

- **Carga Masiva (L)**

Subtareas:

ingresar csv

ingresar datos del csv a la base de datos

Tiempo estimado: 9 horas.

Responsable: Eduardo Apata, Juan Yampara

Criterios de aceptación: Se puede visualizar la información de los alumnos.

- **Actualizar usuarios mediante csv (L)**

Subtareas:

Verificar la información ingresada

Tiempo estimado: 9 horas.

Responsable: Eduardo Apata, Juan Yampara

Criterios de aceptación:

- **Registrar advertencias (S)**

Subtareas:

Guardar información en la base de datos.

Tiempo estimado: 3 horas.

Responsable: Esteban Monsalvez

Criterios de aceptación: Visualizar advertencias en el apartado de advertencias

- **Eliminar advertencias (S)**

Subtareas: No aplica

Tiempo estimado: 3 horas.

Responsable: Esteban Monsalvez

Criterios de aceptación: Visualizar advertencias en el apartado de advertencias

- **Visualizar usuarios penalizados (XS)**

Subtareas: No aplica

Tiempo estimado: 2 horas.

Responsable: Ivan Callasaya

Criterios de aceptación: Visualizar advertencias en el apartado de advertencias



10.1.4. Definición de hecho (DoD).

Para la definición de "hecho", se han establecido ciertos requisitos que cada tarea debe cumplir para considerarse completada correctamente:

1. Criterios de Aceptación

Cada funcionalidad debe cumplir con los criterios de aceptación definidos para ser considerada completa.

2. Desempeño y Escalabilidad

La aplicación ha pasado pruebas de rendimiento y escalabilidad, garantizando velocidad y capacidad ante cargas diversas.

3. Pruebas de Validación

La aplicación debe pasar por una exhaustiva prueba de validación, asegurando que todas las funcionalidades están libres de errores de código y funcionan según lo esperado.

4. Documentación:

El código de la aplicación debe estar documentado tanto para el proyecto del frontend como del lado backend, para que luego otro usuario que quiera actualizar el sistema tenga conocimiento de cada funcionalidad.

10.2. Sprint Review.

Al momento de la entrega de este informe, la reunión con el cliente no pudo llevarse a cabo debido a problemas de disponibilidad de tiempo tanto del cliente como del product owner. Sin embargo, la reunión quedará pendiente y se realizará a la brevedad para poder tener el feedback de la segunda iteración de este proyecto y así poder realizar cambios si son necesarios y poder continuar con la tercera iteración.

10.3. Sprint Retrospective.

Para una retroalimentación grupal se ha optado por realizar las siguientes preguntas:

- ¿Qué hicimos bien?
- ¿Qué podemos mejorar?
- ¿Qué impedimentos surgieron?

Tabla 10 : Sprint Retrospective

¿Qué hicimos bien?	<ul style="list-style-type: none">• Correcto diseño de la base de datos• Interfaz amigable con el usuario• Despliegue del sistema para probar funcionalidades en diferentes equipos.• Comunicación entre equipo (estado de tareas)
¿Qué podemos mejorar?	<ul style="list-style-type: none">• Uso de herramientas para scrum



	<ul style="list-style-type: none"> Definición de fechas límites Mantener comunicación con el cliente
¿Qué impedimentos surgieron?	<ul style="list-style-type: none"> Problemas con un despliegue automático mediante aws deploy y sus pipelines, no se logró Enfermedad en los integrantes de los equipos. Pérdidas de clases debido a distintos imprevistos incontrolables por el grupo.

10.4. Estado del Sprint Backlog.

Tabla 11: Sprint Backlog - Sprint 2

Historia de Usuario	Tarea / Actividad	Responsable	Estimación de Esfuerzo (Puntos)	Estado
Como estudiante quiero poder verificar mi identidad mediante un QR generado para agilizar el proceso de identificación y asegurar que solo yo pueda acceder a mis préstamos.	Validación por medio de un código QR	Juan Yampara, Esteban Monsalvez	L	Completado
Como administración quiero poder editar los ayudantes para actualizar la información de los ayudantes.	Habilitar Funcion para editar usuarios ayudantes	Ivan Callasaya	S	Completado
Como estudiante quiero solicitar un recurso mediante código QR para facilitar y agilizar el proceso de solicitud de préstamos.	Implementar el uso de una cámara para escanear el código qr	Eduardo Apata, Ivan Callasaya	L	Completado
Como estudiante quiero devolver un recurso mediante código QR para hacer el proceso de devolución más rápido.	Implementar el uso de la cámara para escanear el código qr para devolver recursos	Eduardo Apata Esteban Monsalvez	L	Completado
Como administración quiero poder eliminar los ayudantes para retirar a los ayudantes que ya no forman parte durante los semestres.	Implementar lógica de devolución	Ivan Callasaya	S	Completado



Como administración quiero poder ingresar los usuarios por medio de una carga masiva para tener la información de los estudiantes matriculados de la carrera por semestre.	Crear vista de recursos devueltos	Eduardo Apata Juan Yampara	L	Incompleta
Como administración quiero poder actualizar a los usuarios por medio de una carga masiva para mantener actualizada la información de los estudiante matriculados por cada semestre.	Implementar detalles de los recursos devueltos	Juan Yampara Ivan Callayasa	L	Incompleta
Como ayudante quiero registrar advertencias para tener un registro de los estudiantes que han tenido un comportamiento indebido con el recurso se le ha prestado.	Crear vista de recursos prestados	Esteban Monsalvez	S	Completa
Como ayudante quiero eliminar advertencias para corregir errores o retirar advertencias que ya no sean relevantes.	Añadir funcionalidad para crear categorías	Esteban Monsalvez	S	Completa
Como administración quiero visualizar a los usuarios penalizados para tomar una decisión en ser vetado para prestarle recursos.	Añadir funcionalidad para visualizar las penalizaciones automáticas	Ivan Callasaya	XS	Completa



10.5. Evidencia del trabajo.

Frontend: Gestión de usuarios (Agregar, editar y eliminar).

Descripción: El sistema posee la funcionalidad para manejar los usuarios que tendrán acceso al sistema, se podrán crear usuarios con diferentes privilegios o roles y además se podrán editar y eliminar cada uno de ellos.

Gestion de Usuarios

Crear Usuario

<input type="checkbox"/>	ID USUARIO	NOMBRE	CORREO	RUT	ROL	ACCIONES
<input type="checkbox"/>	4	ESTEBAN MONSALVEZ	emonsalvez@gmail.com	22222222-2	AYUDANTE	
<input type="checkbox"/>	5	HUMBERTO URRUTIRA	hurritia@academicos.uta.cl	11111111-1	ADMIN	

Crear Usuario

Nombre

Usuario

Rut

Rol

Seleccionar un Rol

ADMIN

AYUDANTE

Correo

Contraseña

CancelarCrear Usuario

Crear Usuario

Nombre

Usuario

Rut

Rol

ADMIN

Correo

Contraseña

CancelarEditar Usuario

<input type="checkbox"/>	ID USUARIO	NOMBRE	CORREO	RUT	ROL	ACCIONES
<input type="checkbox"/>	4	ESTEBAN MONSALVEZ	emonsalvez@gmail.com	22222222-2	AYUDANTE	
<input type="checkbox"/>	5	HUMBERTO URRUTIRA	hurritia@academicos.uta.cl	11111111-1	ADMIN	

!

¿Estás seguro de eliminar el usuario?

CancelarEliminar

Frontend: Sistema de préstamos.



Descripción: Para el sistema de préstamos se tomó el mismo procedimiento que tiene el sistema actual, primero se elige la categoría asociada para especificar el recurso, luego se elige dicho recurso, luego mediante la cámara obligatoriamente que debe estar conectada al computador, se escanea un código QR con el rut del estudiante para seguir con el siguiente paso y por último se confirma dicho préstamo, luego para devolver el recurso, hay una pestaña de seguimiento en el cual muestran todos los recursos prestados.

Seleccione una categoría

Computadores

Prestar

Devolver

Teclados

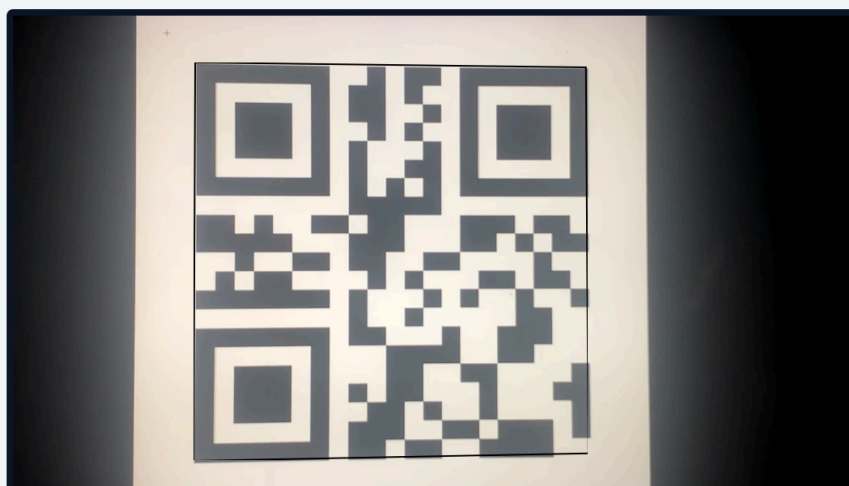
Prestar

Devolver

Seleccione un recurso.

	ID DICI	MODELO	MARCA	DESCRIPCION	ACCIONES
<input type="checkbox"/>	2123	THINKPAD	LENOVO	Sin descripcion	Prestar

Escanea el código QR





Confirmar préstamo

Recurso

id_dici: 2123
id_uta: 123
Nombre: THINKPAD T14
Modelo: THINKPAD
Ubicacion: Departamento

Estudiante

Rut: 11223344-5
Nombre: Carlos Ramírez
Agno: 2019
Direccion: Av. Central 789
Fono: 923456789

Cancelar

Confirmar

Seguimiento de préstamos.						
<input type="checkbox"/>	ID DICI	ID USUARIO	RUT	FECHA DE INICIO	HORA DE INICIO	ACCIONES
<input type="checkbox"/>	2123	5	11223344-5	04/11/2024	15:53:59	Devolver

11. Conclusión.

En la primera fase de planificación, se analizaron los requisitos funcionales iniciales del cliente, identificando tanto las necesidades importantes como las funcionalidades adicionales a incorporar en el sistema. Este proceso se desarrolló mediante varias reuniones con el cliente, utilizando el marco de trabajo Scrum, lo cual permitió definir de manera conjunta las prioridades y expectativas para el sistema.

Durante el Sprint 1, se implementaron los requisitos relacionados con el registro y categorización de recursos, además de los elementos básicos de la lista de ayudantes. Todo lo relacionado con los requisitos del Sprint 1 fue clasificado como de alta prioridad. Aunque algunas funcionalidades quedaron pendientes, el cliente expresó satisfacción con los avances y se tomaron en cuenta sus sugerencias para la siguiente iteración.

En el Sprint 2, se completaron las funcionalidades pendientes de la primera iteración y se añadieron funcionalidades clave, como la verificación mediante QR para el sistema de préstamos y devoluciones, una de las metas fundamentales del sistema al cierre de esta fase. El cliente manifestó conformidad con el rediseño de la interfaz de usuario, ya que fue una de las sugerencias del Sprint 1.



12. Bibliografía.

- [1] Patton, J., & Economy, P. (2014). *User story mapping: discover the whole story, build the right product*. " O'Reilly Media, Inc."
- [2] Izaurralde, M. P. (2013). Caracterización de Especificación de Requerimientos en entornos Ágiles: Historias de Usuario. *Trabajo de especialidad, Febrero*.
- [3] Caso, N. (2004). SCRUM development process. *Universidad Tecnológica Nacional.(En línea)*. Consultado, 22.
- [4] Gallego, M. T. (2012). Metodología scrum. *Universitat Oberta de Catalunya*.