2ºDAW

**Software web – Gestión inventario**

**Desarrollado con Laravel y Bootstrap**

**Hugo Muñiz Vázquez**

**ÍNDICE**

* **Introducción y planificación**

1. **Introducción**
2. **Motivación**
3. **Problema**

* **Objetivos del proyecto**
* Abreviaturas y acrónimos
* Planificación (USO DE JIRA)
* II Materias relacionadas
* **Metodología**

1. Scrum
2. ¿Qué es Scrum?
3. Como comenzó Scrum
4. Principios de Scrum
5. Roles
6. Artefactos
7. Reuniones

* Diseño inicial
* Requisitos del sistema
* Prototipos
* Arquitectura
* Modelo de datos
* Desarrollo del sistema
* Pruebas realizadas y validaciones
* Resumen
* Cierre del proyecto
* Manual de uso
* Framework y herramientas utilizadas
* Conclusiones
* Análisis del proceso desarrollado
* Lecciones aprendidas
* Conclusiones
* Consecución de los objetivos del proyecto
* Limitaciones del proyecto
* Anexos
* Referencias bibliográficas

# Introducción y planificación

## Introducción

En este proyecto, desarrollaremos una aplicación web comercial que integre diversas tecnologías y conceptos aprendidos a lo largo del curso. Nuestro objetivo es crear una plataforma robusta, eficiente y escalable, que ofrezca una experiencia óptima para los usuarios.

Para ello, utilizaremos Laravel como nuestro framework principal, aprovechando su potente arquitectura basada en MVC, su sistema de enrutamiento, autenticación y facilidad de integración con bases de datos. Además, implementaremos buenas prácticas en el desarrollo, optimización del rendimiento y seguridad para garantizar un producto de alta calidad.

A lo largo del desarrollo, exploraremos diferentes aspectos clave, como la gestión de usuarios, la manipulación de datos mediante Eloquent ORM y la integración con un frontend moderno y estético usando Boostrap. Este proyecto no solo servirá como una aplicación funcional, sino también como una oportunidad para consolidar nuestros conocimientos y habilidades en el desarrollo web.

## Motivación

Este proyecto representa mucho más que el desarrollo de una aplicación web; es una puerta abierta a nuevas experiencias y aprendizajes en el competitivo mundo laboral. Trabajar en él implica explorar y aplicar nuevas tecnologías, resolver desafíos complejos y transformar ideas en soluciones prácticas, lo cual enriquece mi perfil profesional de manera significativa.

Además, la causa detrás de esta iniciativa le añade un valor emocional único: se trata de una empresa con la que ya tengo una historia y, sobre todo, de un amigo de confianza. Esta conexión personal convierte cada línea de código en un compromiso que va más allá de lo técnico, integrando sentimientos de lealtad y gratitud hacia el pasado laboral y amistad que me han impulsado a dar siempre lo mejor de mí. Es, en esencia, una forma de retribuir la confianza y de dejar una huella de mejora continua en un entorno que considero parte de mi crecimiento tanto personal como profesional.

La emoción y el aprendizaje se fusionan en este proyecto, haciendo de cada reto una oportunidad para innovar, superarme y construir algo significativo que refleje no solo mis capacidades técnicas, sino también mi compromiso con las relaciones personales y profesionales que han marcado mi trayectoria.

## Problema

Uno de los principales problemas de la empresa Informáticas Marchena era la falta de un sistema eficiente para la gestión de clientes y el control de las entradas y salidas de productos en la tienda. La organización se realizaba de forma manual: los datos de los clientes se guardaban en hojas de Excel, y los productos que entraban o salían se registraban con notas adhesivas (post-its) pegadas en los propios dispositivos. Este método, además de poco práctico, generaba desorden, pérdida de información y dificultades a la hora de llevar un seguimiento claro y actualizado de cada caso.

La solución a este problema es la creación de una aplicación web de gestión a medida. Esta herramienta permitirá registrar y organizar toda la información de los clientes de forma estructurada, así como llevar un control detallado de cada producto que entra o sale de la tienda, ya sea un móvil, un ordenador u otro dispositivo. Todo quedará digitalizado, almacenado de forma segura y accesible para los trabajadores, facilitando su labor diaria. Con esta aplicación, se consigue una gestión más rápida, clara y eficiente, mejorando notablemente el funcionamiento interno de la empresa.

# Objetivos del proyecto

El desarrollo de esta aplicación web que tendrá como propósito principal optimizar la gestión de una tienda local, facilitando el control de inventario, clientes, ventas y otros procesos administrativos. Para ello, se han establecido los siguientes objetivos:

* Objetivo General

Desarrollar una aplicación web eficiente y fácil de usar, basada en Laravel, que permita la gestión de una tienda, asegurando un control preciso sobre los productos, clientes y operaciones diarias.

* Objetivos Específicos
* Automatizar los procesos administrativos

Reducir la carga de trabajo manual en la gestión de productos, entradas y salidas de stock.

Permitir el registro y seguimiento de clientes y proveedores de manera organizada.

* Garantizar un sistema seguro y confiable

Implementar medidas de seguridad como autenticación de usuarios, control de accesos.

Asegurar la protección de información sensible mediante buenas prácticas de seguridad en Laravel.

* Optimizar la gestión del inventario

Permitir la actualización en tiempo real de los productos disponibles en la tienda.

Ofrecer alertas sobre niveles bajos de stock para facilitar la reposición de productos.

* Mejorar la experiencia del usuario

Diseñar una interfaz intuitiva y amigable para facilitar su uso a empleados y administradores.

Ofrecer compatibilidad con diferentes dispositivos para garantizar acceso desde cualquier lugar.

* Facilitar la generación de reportes y estadísticas

Permitir al administrador de la tienda, tomar decisiones con base en datos analíticos.

* Asegurar la escalabilidad del sistema

Diseñar la aplicación con una arquitectura modular que permita futuras expansiones.

Brindar documentación clara sobre el funcionamiento de la aplicación.

Este proyecto no solo busca crear una herramienta funcional, sino que también servirá como una experiencia de aprendizaje integral, aplicando conocimientos adquiridos en Laravel y otras tecnologías.

# Planificación

Para garantizar una gestión eficiente del desarrollo de la aplicación web comercial, utilizaremos JIRA como nuestra herramienta principal de planificación y seguimiento de tareas. JIRA nos permitirá organizar el trabajo de manera estructurada, definir prioridades y asignar responsabilidades dentro del equipo de desarrollo.

-Metodología de Trabajo

Aplicaremos un enfoque ágil, basado en Scrum, dependiendo de las necesidades del proyecto. Esto nos permitirá trabajar en iteraciones cortas, hacer revisiones constantes y mejorar progresivamente la aplicación según los requisitos que se vayan definiendo durante el proceso de desarrollo.

-Estructura en JIRA

En JIRA, gestionaremos nuestro proyecto a través de:

Sprintsy tableros de tares: definiremos sprints con objetivos específicos y fechas límite. Cada Sprint tendrá asociado una serie de tareas las cuales tendrán 3 estados posibles: to-do , in progress y done.

-Ventajas del Uso de JIRA en el Proyecto

* Organización clara de todas las tareas y funcionalidades a desarrollar.
* Seguimiento en tiempo real del progreso de cada fase del proyecto.
* Gestión eficiente de prioridades, evitando retrasos y asegurando entregas a tiempo.
* Documentación detallada del proceso de desarrollo.

En conclusión, JIRA será una herramienta fundamental para llevar a cabo una planificación detallada y un desarrollo estructurado, asegurando que nuestro proyecto avance de manera organizada y eficiente hasta su entrega final.

# II Materias relacionadas

## 1. Usos y aplicaciones de laravel

### Desarrollo de aplicaciones web complejas

* **Desarrollo Full Stack:** Laravel permite crear desde simples sitios web hasta aplicaciones empresariales de gran escala, gracias a su estructura organizada y las herramientas integradas para gestionar bases de datos, seguridad, rutas y autenticación.
* **APIs RESTful:** Con sus recursos y la facilidad de configuración, Laravel es ideal para desarrollar APIs que se integran con aplicaciones móviles o front-ends en frameworks modernos como Vue.js, React o Angular.

### Prototipado rápido y Desarrollo Ágil

* **Scaffolding:** Laravel incluye herramientas de scaffolding para generar rápidamente la estructura inicial de la aplicación, permitiendo a los desarrolladores concentrarse en la lógica del negocio.
* **Artisan Console:** Una interfaz de línea de comandos que automatiza tareas recurrentes como migraciones, pruebas y generación de código, aumentando la productividad.

### Gestión de Bases de Datos

* **Eloquent ORM:** Es el sistema de mapeo objeto-relacional de Laravel. Facilita operaciones CRUD (crear, leer, actualizar y eliminar) sobre la base de datos mediante una sintaxis fluida y expresiva.
* **Migraciones:** Permiten versionar la base de datos, asegurando que los cambios en el esquema se puedan propagar de manera controlada en diferentes entornos de desarrollo, prueba y producción.

## 2. Componentes y Herramientas Clave en Laravel

### Blade Template Engine

* **Plantillas Dinámicas:** Blade es el motor de plantillas de Laravel que permite incrustar código PHP en los archivos de vistas utilizando sintaxis sencilla y limpia.
* **Herencia de Plantillas:** Facilita la reutilización de diseños comunes y la estructuración lógica de la interfaz, lo que favorece la mantenibilidad y escalabilidad del proyecto.

### Ruteo y Controladores

* **Ruteo Express:** Laravel proporciona una forma sencilla y expresiva de definir rutas, lo que permite mapear URLs a funciones o controladores específicos.
* **Controladores Resource:** Permiten agrupar múltiples acciones relacionadas en un solo lugar, simplificando la lógica de negocio y la conexión con modelos y vistas.

### Middleware y Seguridad

* **Autenticación y Autorización:** Laravel facilita la integración de sistemas de autenticación robustos, con middleware que protege rutas específicas y gestiona permisos de usuarios.
* **Protección CSRF:** Incluye mecanismos para proteger las aplicaciones contra ataques Cross-Site Request Forgery, una característica esencial en el desarrollo seguro.

### Testing y Mantenimiento

* **Test Driven Development (TDD):** Laravel viene configurado con PHPUnit y herramientas que permiten escribir y ejecutar pruebas unitarias y funcionales, ayudando a mantener la calidad del software.
* **Debugging y Logging:** Con herramientas como Laravel Telescope y logging integrado, los desarrolladores pueden monitorear y depurar la aplicación de manera eficiente.

## 3. Materias Relacionadas e Información de Interés

### Materias Relacionadas

* **Programación y Desarrollo Web:** Explora conceptos de programación orientada a objetos, MVC y desarrollo web moderno.
* **Bases de Datos y SQL:** Dado que Laravel utiliza Eloquent ORM y migraciones, es fundamental comprender el manejo de bases de datos y las operaciones CRUD.
* **Arquitectura de Software:** El conocimiento de patrones de diseño (como MVC, repositorios y servicios) es clave para aprovechar al máximo el potencial de Laravel.
* **Seguridad Informática:** Considera estudiar mecanismos de seguridad en aplicaciones web, como autenticación, autorización y protección contra inyecciones y ataques CSRF.
* **Desarrollo de APIs:** Dado el auge de los servicios RESTful, profundizar en el diseño y consumo de APIs puede complementar tu proyecto.

### Información de Interés sobre Laravel

* **Comunidad y Ecosistema:** Laravel posee una amplia comunidad que ofrece soporte, paquetes y extensiones (como Laravel Nova para administración o Laravel Horizon para manejo de colas). La extensa documentación y foros de la comunidad son recursos valiosos para resolver dudas y aprender mejores prácticas.
* **Evolución y Actualizaciones:** Desde su lanzamiento en 2011, Laravel ha evolucionado significativamente, incorporando nuevas características basadas en las tendencias tecnológicas y las necesidades de los desarrolladores. Esto garantiza que el framework se mantenga actualizado y sea adaptable a nuevas demandas del mercado.
* **Casos de Éxito y Empresas:** Muchas empresas reconocidas utilizan Laravel para desarrollar sus aplicaciones web. Analizar estos casos te puede proporcionar ejemplos prácticos y justificar la elección del framework en entornos de producción.
* **Performance y Escalabilidad:** Aunque tradicionalmente se relaciona con prototipado y desarrollo ágil, Laravel también cuenta con herramientas y técnicas (como optimización de consultas Eloquent y caché con Redis o Memcached) que permiten escalar aplicaciones a niveles de producción con alta demanda.

### Herramientas Complementarias

* **Composer:** Es el gestor de dependencias de PHP utilizado en Laravel para instalar y actualizar paquetes, lo que facilita la integración de librerías externas.
* **Laravel Mix:** Una herramienta basada en Webpack que simplifica la compilación y optimización de assets (CSS, JavaScript, etc.), permitiendo un desarrollo front-end coherente y eficiente.
* **Docker y Homestead:** Estas herramientas ayudan a crear entornos de desarrollo replicables, lo que resulta muy útil tanto para aprendizaje como para proyectos profesionales.

# Metodología

## SCRUM

### ¿Qué es Scrum?

Scrum [9] es una metodología basada en aplicar de manera regular un conjunto de buenas prácticas para trabajar colaborativamente y obtener el mejor resultado posible en un proyecto.

En Scrum, se llevan a cabo entregas parciales y regulares del proyecto final, priorizadas por el beneficio que aportan a la persona encargada de recibir el producto. Por ello, es especialmente recomendado para proyectos complejos, en los cuales se necesitan obtener resultados de manera temprana, donde los requisitos pueden variar o se encuentran poco definidos,y además, la innovación, la competitividad, la flexibilidad y la productividad son aspectos imprescindibles.

También se usa para resolver escenarios en los que no se está entregando al cliente lo que desea, cuando las entregas se producen en tiempos elevados, los costes de desarrollo se incrementa o la calidad es inaceptable, cuando se necesita baja capacidad de reacción por parte de la competencia, entre otros motivos.

### Como comenzó Scrum

**Inicios**

Scrum surge en un contexto en el que los proyectos software que se desarrollaban hasta entonces tenían una serie de características comunes:

* Requisitos fuera de control.
* No se cumplían los tiempos planificados.
* Las estimaciones muchas veces eran deficientes.
* Había que repetir trabajo de manera excesiva.
* Proyectos de mala calidad con costes excedidos.
* Se producía insatisfacción por parte del cliente y de los profesionales participantes.

Todas estas situaciones, producían en numerosas ocasiones una situación de malestar en los clientes, debido al constante aumento de costes, tiempos de entrega y mala elicitación de requisitos por parte de los profesionales.

**Historia**

La historia de Scrum se remonta a 1986 [39] en un artículo de la Harvard Business Review, “El nuevo juego para el desarrollo de productos” (“The New ProductDevelopmentGame”). Este artículo habla de cómo empresas como Canon, Honda y Fuji-Xerox desarrolan nuevos productos a nivel mundial usando una perspectiva escalable la cual se basa en equipos integrales. Este enfoque prioriza la importancia de delegar poder a los equipos auto-organizados. Scrum [9] no es un acrónimo, sino un término extraído del rugby y significa melé; es decir, que los compañeros de un mismo equipo se amontonan, formando una piña para empujar todos con una misma dirección.

Los impulsores del manifiesto ágil promovieron modelos de organización basados en las personas y la colaboración.

**Principios del manifiesto ágil**

En el año 2001, Ken Schwaber escribe el primer libro que habla sobre desarrollo ágil de software con Scrum [9], y junto a Sutherland y otros 15 agilistas firman el manifiesto Ágil de desarrollo de software [12], que sienta los principios de la agilidad[13]. Estos principios guían la forma en que los equipos ágiles trabajan y toman decisiones. Son una representación del compromiso de estos equipos con prácticas de trabajo flexibles, comunicación efectiva y entrega continua de software de alta calidad. A continuación se detallan (tomados de [13]):

1. “Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.”

2. “Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.”

3. “Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.”

4. “Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.”

5. “Los proyectos se desarrollan en torno a individuos motivados.

Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.”

6. “El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.”

7. “El software funcionando es la medida principal de progreso.”

8. “Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.”

9. “La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.”

10. “La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.”

11. “Las mejores arquitecturas, surgen de equipos auto-organizados.”

12. “A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación, ajustar y perfeccionar su comportamiento en consecuencia.”

Actualmente, se usa Scrum en diferentes tipos de negocio y, sobre todo, en lo relacionado con el desarrollo de software. La Scrum Alliance es una organización sin ánimo de lucro que se encarga de promover Scrum en este ámbito.

### Principios de Scrum

A partir del manifiesto ágil de desarrollo de software nace Scrum. Con una serie de principios [16] que se basan en defienden los axiomas que se defienden en este. Los más importantes son:

* Control empírico del proceso: Este principio enfatiza la filosofía de Scrum basada en las tres ideas principales de:

1. Transparencia: Asegura que toda la organización esté consciente de los desarrollos y progresos en el proyecto.

2. Revisión: Se efectúa de manera frecuente para identificar y corregir posibles desviaciones.

3. Adaptación: Incentiva la realización de ajustes necesarios para adaptarse a las mejores formas de trabajo para el proyecto.

* Auto-organización: Este principio pone el foco en la autoorganización de los miembros del equipo, promoviendo así una mayor entrega de valor y fomentando un entorno innovador y creativo propicio para el crecimiento.
* Colaboración: Este principio destaca las tres dimensiones fundamentales relacionadas con el trabajo colaborativo: la conciencia, la articulación y la apropiación.
* Valor basado en la priorización: Este principio tiene como objetivo ofrecer el máximo valor comercial al negocio del cliente durante todo el desarrollo del proyecto.
* Time-boxing: Este principio subraya que el tiempo es un recurso invaluable y finito, y que su utilización adecuada contribuye significativamente al cumplimiento efectivo de la planificación del proyecto. Dentro de Scrum, los elementos que siguen este principio incluyen:

1. Sprints

2. Reuniones de equipo diarias

3. Reuniones de Planificación de Sprint

4. Reuniones de revisión de Sprint

* Desarrollo iterativo: Este principio se enfoca en mejorar la gestión de cambios, permitiendo así construir productos que satisfagan las necesidades del cliente. Además, establece las responsabilidades del Propietario del Producto, el equipo de desarrollo y otros roles involucrados en el proyecto.

### Roles

Dentro del marco de Scrum, cada participante en el proyecto mantiene una conexión ya sea directa o indirecta con el mismo, y se pueden categorizar en dos grupos distintos: los comprometidos, que abarcan al Product Owner, el Scrum Master y el Equipo de Desarrollo, y los implicados, que incluyen roles como dirección, gerencia, ventas, marketing, entre otros. En este documento, nos enfocaremos en aquellos que tienen una conexión directa con el proyecto.

* Product Owner (Dueño del Producto): Esta figura es crucial, ya que es el encargado de maximizar el valor del producto para clientes, usuarios y otros participantes. Actúa como un enlace entre el cliente y el equipo de desarrollo. Una de sus principales tareas es determinar la visión final del producto, guiando la secuencia de crecimiento mediante la selección y priorización de elementos en el backlog del producto. Está bien informado sobre el plan de producto, sus capacidades y las expectativas de retorno de la inversión, asumiendo la responsabilidad de definir fechas de lanzamiento y características de las distintas versiones. Es imperativo que clarifique todas las especificaciones al equipo, incluyendo al Scrum Master y al equipo de desarrollo.

Para asumir este rol eficazmente, se requiere que:

* Comprenda profundamente las necesidades y metas asociadas con el sistema en desarrollo, con un conocimiento profundo del contexto empresarial del cliente.
* Posea una clara visión del producto y comprenda las demandas específicas del proyecto para priorizar las tareas de manera efectiva.
* Tenga un profundo entendimiento y autoridad sobre el plan del producto para tomar decisiones críticas durante el desarrollo, cubriendo así las expectativas de retorno de inversión.
* Sea proactivo en recibir y evaluar feedback continuo del ambiente empresarial y del proyecto, incluyendo recomendaciones del equipo, alternativas técnicas y evaluación de cada incremento.
* Finalmente, es vital que tenga una comprensión sólida de Scrum para abordar las tareas de manera efectiva y tenga una familiaridad previa o experiencia trabajando con el equipo.
* Scrum Master: Este individual supervisa la adherencia al marco de Scrum por parte del equipo de desarrollo. Asegura que el equipo se alinee con la teoría, las prácticas y las normas de Scrum. Ofrece varios servicios tanto al Product Owner como al equipo, como:
  + Orientación y capacitación al equipo para fomentar la autoorganización y la responsabilidad colectiva.
  + Revisión y validación del backlog del producto.
  + Facilitación de las reuniones.
  + Solución de problemas que puedan obstruir el progreso durante un sprint.

Cabe destacar que, aunque un Scrum Master puede servir a múltiples equipos, su disponibilidad puede influir en el éxito final del proceso Scrum.

* Equipo de Desarrollo (Developer Team): Consiste en un conjunto de expertos que llevan a cabo el desarrollo en cada Sprint. Idealmente, este equipo debería tener entre 3 y 9 miembros para facilitar una comunicación fluida entre todos. Es importante mencionar que esta cuenta no incluye al Product Owner ni al Scrum Master. Debería ser un equipo autoorganizado, con la autonomía de participar directamente en la construcción del producto.

El equipo de desarrollo adopta una estructura multifuncional, lo que significa que no asigna roles específicos como Diseñador, Analista, Programador, etc. En lugar de eso, todos los miembros asumen una responsabilidad colectiva. Aunque cada uno puede tener una especialización, la meta común es avanzar con cada sprint, una responsabilidad que recae en el equipo en su conjunto.

### Artefactos

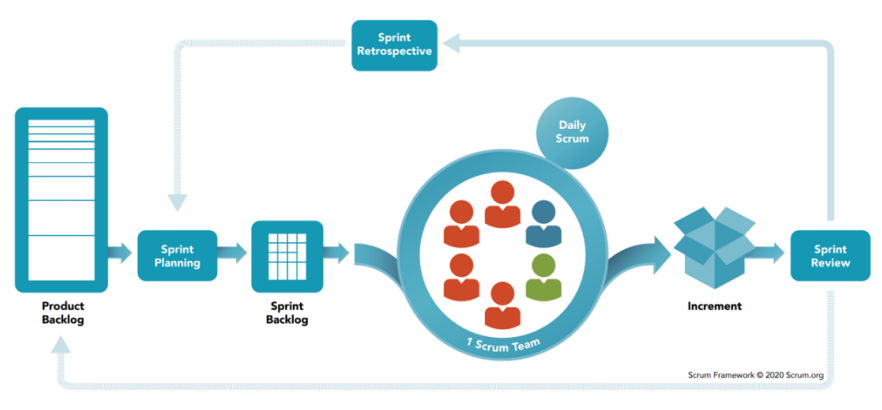
Se denomina artefacto, todo aquel elemento físico que es producido al usar la metodología Scrum. Los tres principales son: el Product Backlog, Sprint Backlog, el Incremento y el Gráfico Burndown Chart .

* Pila del Producto (Product Backlog): Registro evolutivo de requerimientos del usuario, que se desarrolla progresivamente desde la concepción inicial del producto.
* Pila del Sprint (Sprint Backlog): Conjunto de tareas seleccionadas por el equipo para ser completadas durante el sprint actual, contribuyendo al incremento planificado.
* Incremento: Producto tangible y funcional obtenido al final de cada sprint.
* Gráfico Burndown Chart: Representación gráfica del progreso del equipo, mostrando la cantidad de trabajo pendiente a lo largo del tiempo y reflejando la rapidez con la que se están alcanzando los objetivos.

### Reuniones

Scrum establece cuatro eventos centrales para asegurar la gestión eficaz de sus procesos, cada uno con una característica única que los convierte en esenciales para este enfoque. La justificación para tener estos cuatro componente es facilitar que la supervisión eficaz de los procesos.

* La Organización del Sprint al principio del Sprint.
* Reuniones Scrum diarias.
* Una Evaluación del Sprint al terminar el sprint para examinar el avance logrado.
* Una Retrospectiva para evaluar al equipo y establecer mejoras a implementar en el próximo sprint.



**¿Qué es un Sprint?**

Un Sprint representa cada ciclo de desarrollo. Al iniciar con Scrum, se sugiere visualizar el sprint como el acontecimiento que engloba todas las ceremonias, desde la junta de planificación del sprint, pasando por el Scrum diario, hasta la evaluación del sprint y la sesión retrospectiva.

El propósito de la retrospectiva es reflexionar acerca de la iteración finalizada e identificar áreas de mejora para la siguiente. Se efectúa una introspección de los métodos de trabajo actuales, resaltando tanto virtudes como áreas de oportunidad. La meta es fortalecer las primeras y establecer estrategias de mejora para las segundas. Todo lo expuesto anteriormente, hacen de Scrum una metodología ideal, con muchas fortalezas, y la cual usan la mayoría de empresas de desarrollo de software actualmente.

# Diseño inicial

AQUÍ VA LA CAPTURA DEL DIAGRAMA DE LUCICHART

# Requisitos del sistema

## 1. Gestión de Usuarios y Roles

### Autenticación y Control de Acceso

* **Inicio de Sesión:** La aplicación contará con una pestaña de login, donde los usuarios deberán autenticarse mediante un sistema seguro basado en email/contraseña.
* **Roles de Usuario:** Se definirán dos roles fundamentales:
  + **Administrador (admin):** Tiene privilegios avanzados. Entre ellos, la capacidad de crear nuevos usuarios y gestionar aspectos generales del sistema.
  + **Usuario (user):** Dispone de acceso restringido a determinadas funciones del dashboard, sin la capacidad de crear nuevos usuarios.

### Gestión de Usuarios

* **Creación de Usuarios:** Exclusiva para usuarios administradores. Esta funcionalidad permite al admin incorporar nuevos perfiles al sistema, asegurando así el control centralizado de los accesos.

## 2. Funcionalidades del Dashboard

El dashboard será la interfaz central de la aplicación, compartida en apariencia entre administradores y usuarios, salvo la funcionalidad exclusiva del administrador para la creación de usuarios.

### Botón de "Añadir Depósito"

* **Funcionalidad:** Permite al usuario introducir los datos del cuestionario relativo a un depósito. Este depósito representa el registro de un artículo que se entrega a la tienda para su reparación o para solucionar el problema presentado.
* **Proceso:** Al pulsar este botón, se despliega un formulario donde se recogen los datos necesarios para generar un nuevo registro de depósito, asumiendo que cada registro incluye información clave sobre el artículo y el problema identificado.

### Secciones Principales del Dashboard

1. **Ver Depósito Finalizado**
   * **Descripción:** Sección en la que se muestran los artículos que ya han sido reparados y se encuentran listos para ser entregados nuevamente a su propietario. Esta visualización incluye aquellos depósitos cuyo proceso de reparación ha concluido con éxito.
2. **Ver Depósito en Curso**
   * **Descripción:** Área dedicada a los productos que, en el momento, se encuentran en proceso de reparación. Permite tanto a usuarios como a administradores visualizar el estado actual de los depósitos en curso, facilitando la gestión y el seguimiento de las reparaciones.
3. **Ver/Gestionar Clientes**
   * **Descripción:** Espacio para gestionar la información de los clientes. En esta sección se podrán añadir nuevos clientes, así como visualizar y administrar la lista de clientes existentes. Esto posibilita un control integral sobre quiénes depositan sus artículos para reparación y permite una comunicación efectiva sobre el estado de sus productos.

### Funcionalidades Adicionales en el Dashboard

* **Histórico de Actividades:**  
  Se dispone de una pestaña específica donde se registra un historial detallado de las acciones realizadas, como reparaciones concluidas, compras o cualquier transacción que involucre a un cliente. Este registro cronológico permite tener un seguimiento completo y transparente de las operaciones efectuadas en la aplicación.
* **Notificaciones vía WhatsApp:**
  + **Acción Automática:** Cuando se mueve un producto de la sección “Depósito en Curso” a “Depósito Finalizado”, se activa un sistema de notificación que envía un mensaje por WhatsApp al cliente.
  + **Propósito:** El mensaje notifica al cliente que su artículo está listo para ser recogido, mejorando la comunicación y la eficiencia en la entrega.

## 3. Flujo de Trabajo Global de la Aplicación

1. **Ingreso y Autenticación:**  
   El usuario (administrador o usuario común) inicia sesión desde la pestaña de login.
2. **Acceso al Dashboard:**  
   Una vez autenticados, acceden a un dashboard unificado. Aunque la visualización es similar para ambos roles, el administrador dispone de la función adicional para crear nuevos usuarios.
3. **Añadir Depósito:**  
   Mediante el botón “Añadir Depósito”, el usuario completa un cuestionario para registrar un nuevo artículo destinado a reparación. Esta acción crea un nuevo registro en el sistema.
4. **Gestión de Depósitos:**
   * Se pueden visualizar tres recuadros o secciones que facilitan el seguimiento:
     + **Depósitos Finalizados:** Artículos reparados y listos para ser entregados.
     + **Depósitos en Curso:** Artículos actualmente en proceso de reparación.
     + **Clientes:** Registro y gestión de clientes, permitiendo añadir nuevos o revisar los existentes.
5. **Histórico de Actividades:**  
   Se mantiene un registro histórico de todas las transacciones y reparaciones, permitiendo consultar la información en cualquier momento para verificar fechas, procesos y detalles de cada reparación o compra.
6. **Notificaciones Proactivas:**  
   Una vez que un artículo se marca como finalizado, se envía automáticamente una notificación por WhatsApp al cliente para avisarle que puede pasar a recoger su producto en la tienda.

# Prototipos

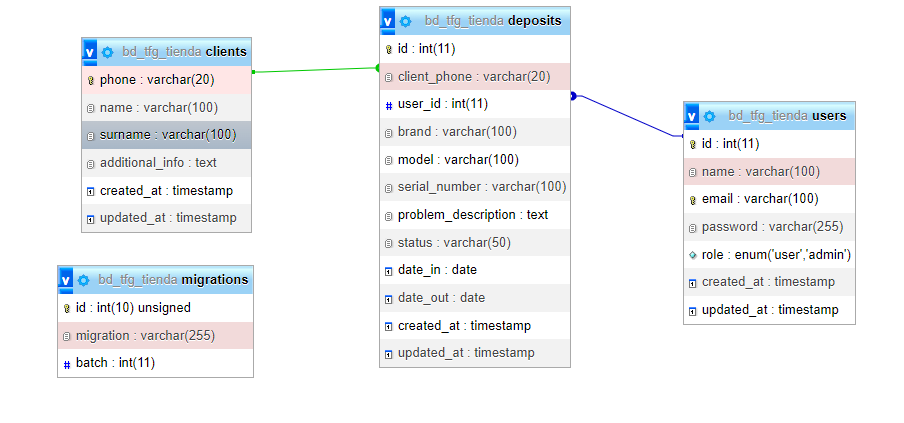
DESARROLLO EN FIGMA

# Arquitectura

# Modelo de datos

En la siguiente imagen se muestra el modelo de datos que tendrá nuestra aplicación. En el cual distinguimos las siguientes tablas o entidades:

* **Clientes:** Aquí se guarda la información de cada cliente, como su nombre, teléfono y otros datos básicos.
* **Depósitos:** Esta parte es donde se registran los artículos que los clientes dejan para su reparación. Cada depósito está vinculado a un cliente y también a un usuario (la persona encargada) que gestiona ese depósito.
* **Usuarios:** En esta sección se almacenan los datos de las personas que usan la aplicación (tanto los administradores como los usuarios normales), incluyendo su nombre, correo y otros detalles importantes.
* **Migraciones:** Esta parte sirve para mantener un registro de los cambios que se han hecho en la estructura de la base de datos a lo largo del tiempo; es una herramienta común en Laravel para organizar y controlar esos cambios.



# Desarrollo del sistema

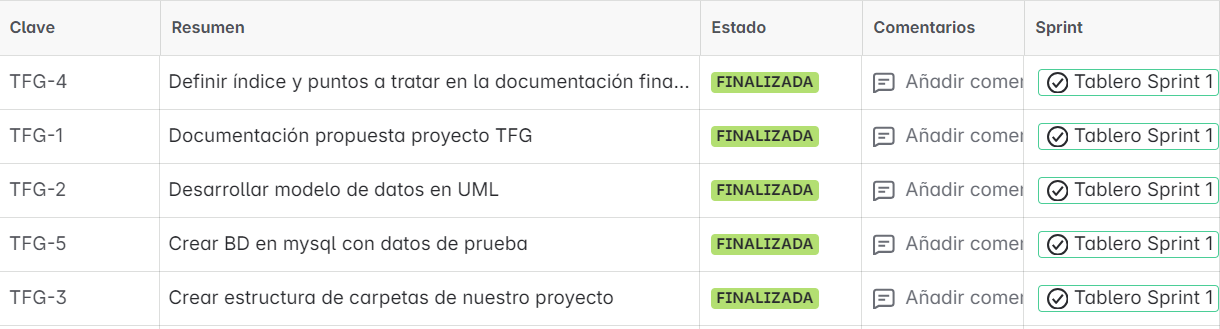
Para el desarrollo de la aplicación, se dividió el esfuerzo en Sprints usando la metodología Scrum que previamente fue en apartados anteriores fue mencionado y explicado.

Para cada Sprint se creó una serie de tareas las cuales se añadieron a un Tablero Kanban, el cual permitió durante la etapa de desarrollo organizar y agrupar el estado de consecución de dichas tareas.

Para el seguimiento de estos Sprints se usó la herramienta Jira de Atalassian.

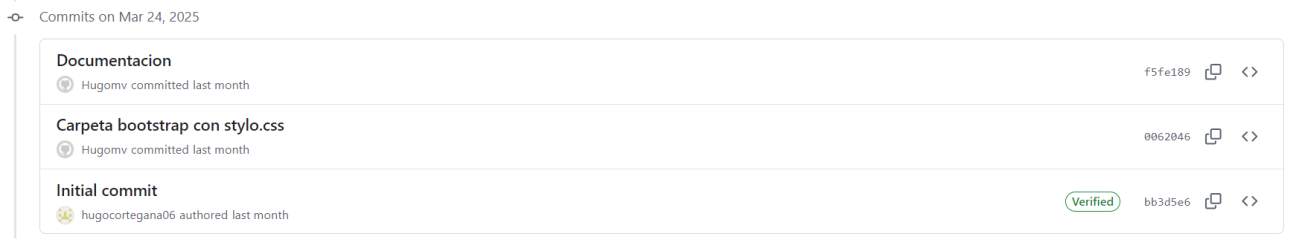
### Sprint 1

Para este Sprint se crearon las tareas relacionadas con la creación de la base del proyecto. Abarcando tanto los aspectos de documentación , donde se creó el esqueleto de nuestro documento final como la parte de desarrollo del modelo de datos y base de datos en MySql. Que posteriormente nos permitirá almacenar en nuestro sistema la información.

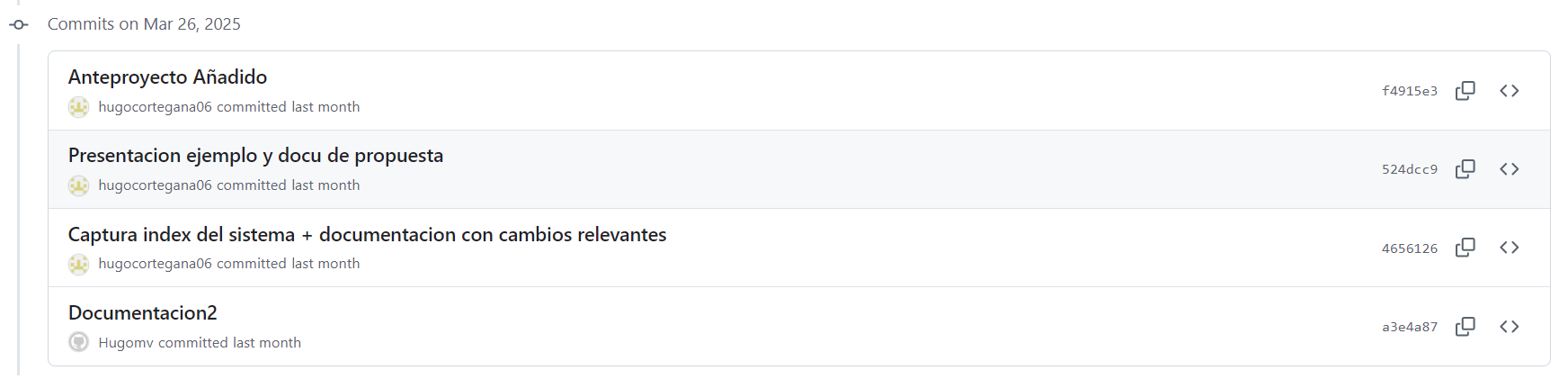


En cuanto a los commits relacionados que se realizaron en nuestro repositorio de Github, se encuentran los que se detallan a continuación:

1. Primer commit



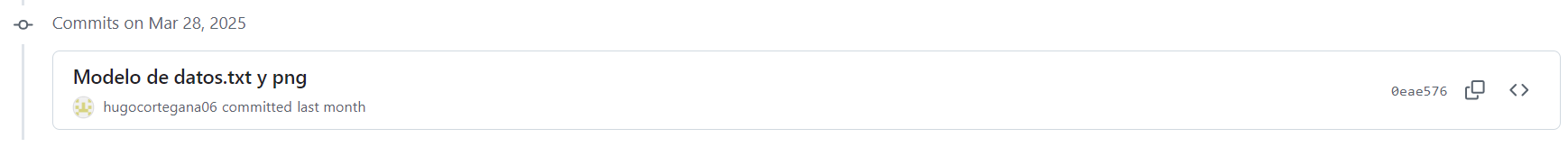
1. Segundo commit



1. Tercer commit



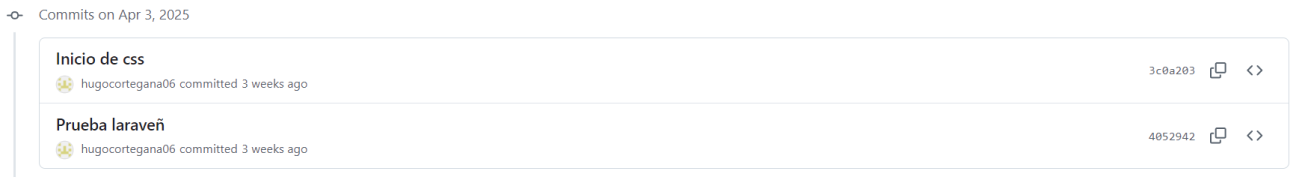
1. Cuarto commit



1. Quinto commit

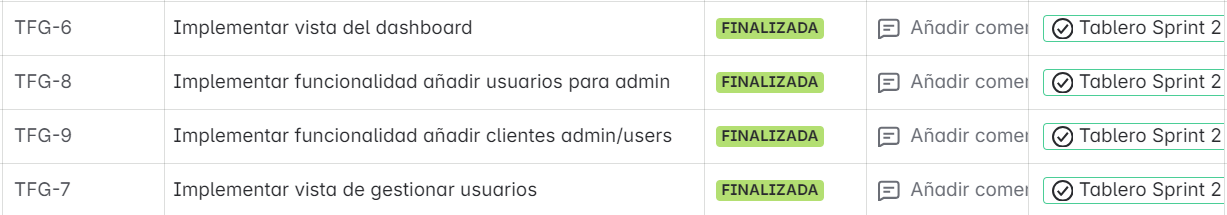


1. Sexto commit

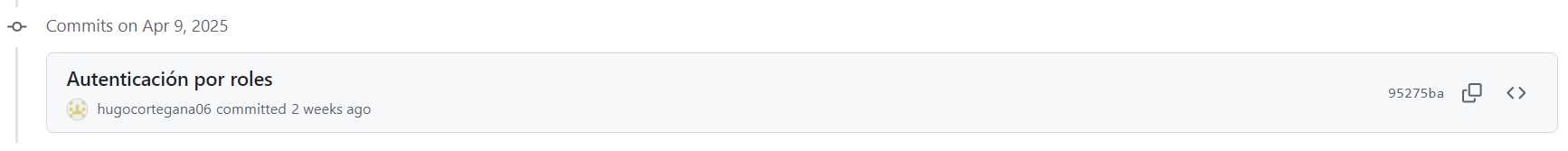


### Sprint 2

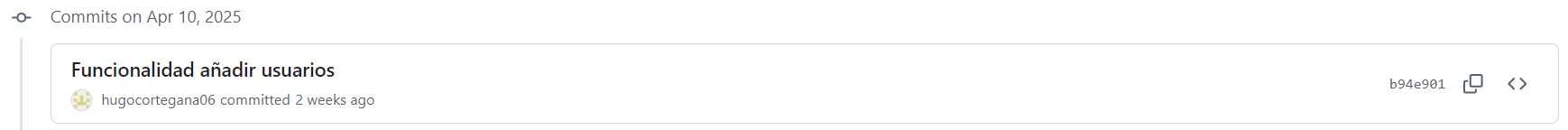
EXPLICACION DE LO QUE HICISTE EN ESTE SPRINT



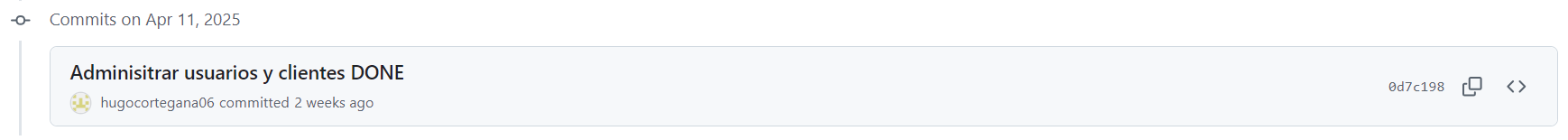
1. Séptimo commit



1. Octavo commit

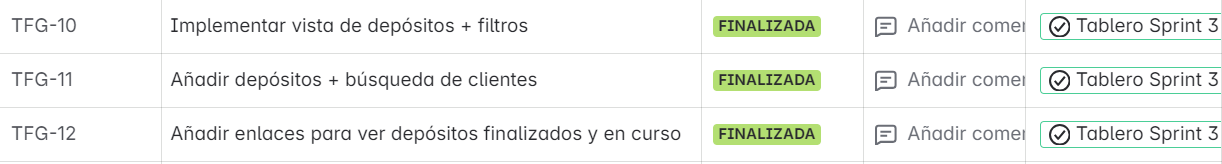


1. Noveno commit



### Sprint 3

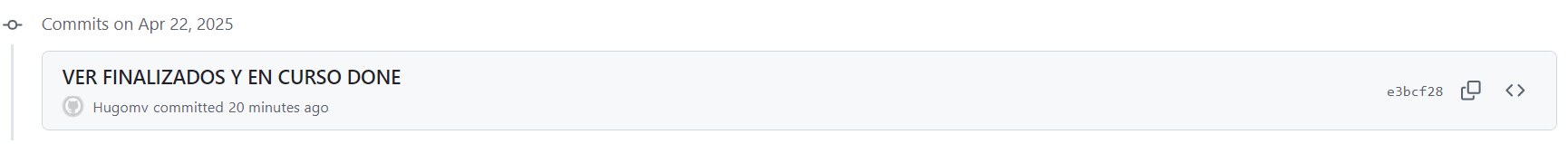
EXPLICACION DE LO QUE HICISTE EN ESTE SPRINT



1. Decimo commit



1. Undécimo commit

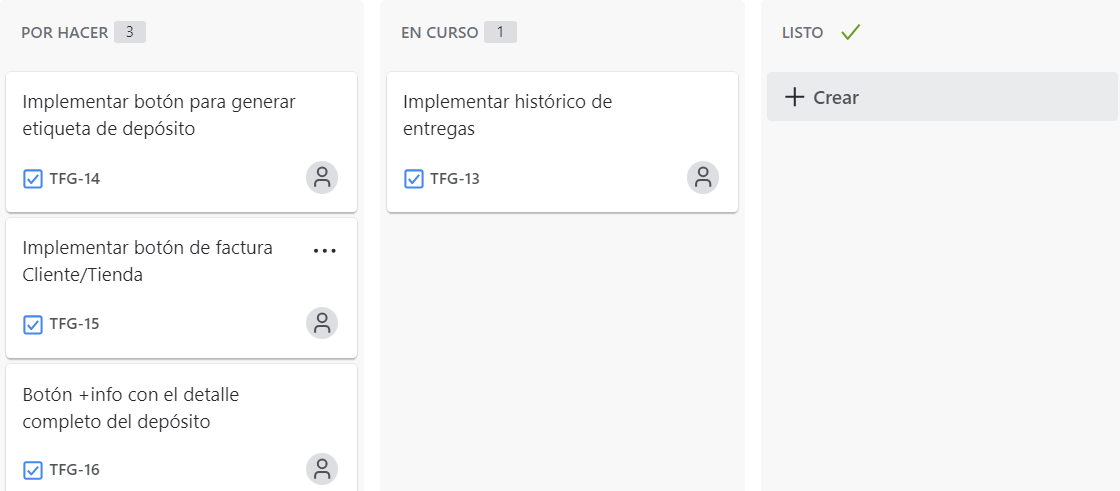


### Sprint 4

EXPLICACION DE LO QUE HICISTE EN ESTE SPRINT



Para este Sprint se adjunta una captura del Tablero Kanban donde se muestran las tareas creadas para completar el mismo.



### Sprint 5

EXPLICACION DE LO QUE HICISTE EN ESTE SPRINT

TAREAS DE SPRINT

COMMITS ASOCIADOS.

# Pruebas realizadas y validaciones

# Resumen

# Cierre del proyecto

# Manual de uso

# Framework y herramientas utilizadas

Framework -> Laravel para el backend

Frontend ->Javascript (Alpine JS) + HTML + Boostrap