



UNIVERSIDAD TECNOLÓGICA
NACIONAL
FACULTAD REGIONAL DE VILLA
MARÍA

Entrega N°3

ASIGNATURA: Programación Avanzada
DOCENTE: Vanzetti, Juan José

INTEGRANTES:
Botta, Francisco
Bortolin, Stéfano
Falco, Augusto
Frighetto, Franco
Lopez Bruno, Valentino

Plan de Proyecto	3
Introducción	3
Visión General del Proyecto	3
Entregable del Proyecto	3
Material de Referencia	4
Organización del Proyecto	4
Responsabilidades	4
Proceso de Gestión	5
Objetivos y Prioridades de Gestión	5
Supuestos, dependencias y restricciones	5
Gestión de Riesgos	5
Proceso Técnico	5
Métodos, herramientas y técnicas	5
Documentación de Software	6
Calidad	12
Plan de pruebas	13
Monitoreo	14
Ramas GitHub	15
Video del funcionamiento del sistema	16
Capturas del interfaz del sistema	17

Plan de Proyecto

Introducción

Visión General del Proyecto

El proyecto tiene como objetivo principal el desarrollo y la implementación de un Sistema de Gestión (integral) para el taller de autos “Autos S.A”. El mismo permitirá llevar a cabo la gestión de clientes, gestión de órdenes de trabajo, gestión de stock del taller y la generación de informes y estadísticas para la toma de decisiones basadas en datos dentro de la empresa. Todos estos aspectos serán implementados para su utilización en un software informático que se le entregará al cliente en cuestión.

El proyecto se desarrollará a través de una serie de actividades en las que se incluyen:

- Análisis de Requerimientos
- Diseño del sistema
- Implementación del sistema
- Generación de Plan de Proyecto
- Estimación de requerimientos.
- Pruebas de software.
- Revisiones y control de calidad del software.

Entregable del Proyecto

Versión	Descripción	Entregable	Fecha Entrega	Lugar Entrega
1.0	Funcionalidad inicial que permite la gestión de vehículos, marca, modelo, clientes y técnico.	Gestión de vehículos, marca, modelo, clientes y técnicos.	11/09/23	A definir.
1.1	Funcionalidad que permita registrar las órdenes de trabajo	Gestión orden de trabajo	09/10/23	A definir.
1.1.2	Funcionalidad que permite asignarle un estado a las órdenes de trabajo.	Gestión orden de trabajo	21/12/23	A definir.
1.2	Funcionalidad que permite crear un documento con datos, a modo de ‘factura’.	Gestión factura	21/12/23	A definir.

Material de Referencia

Título	Informe N°	Autor	Organización Publicada	Referencia
Norme IEEE 1058.1	1	Cobos Lomeli Manuel Alejandro Lopez Rivera Jose Miguel Hernande Hernandez Aaron Prof. Margarita Maria de Lourdes Sanchez	Universidad Autónoma Metropolitana	http://aniei.org.mx/paginas/uam/CursoAI/IEEE_rep.pdf

Organización del Proyecto

La organización encargada del proyecto se limitará a las actividades relacionadas con la comunicación constante con el cliente, la captación de requerimientos y el desarrollo e implementación de los mismos. Por otro lado, llevará a cabo la autoevaluación utilizando como base las retrospectivas dadas por el cliente y de esta forma desempeñará una serie de actividades para la reorientación del proyecto, en caso de que fuese necesario.

Por otra parte, el cliente, limita su participación en el proyecto a las actividades de informe de requerimientos que desea, así como también a la mantención y/o actualización de los mismos en caso de que fuese necesario. A su vez y siempre que lo considere pertinente podrá estar presente en las reuniones llevadas a cabo diariamente por la organización del proyecto para conocer distintos aspectos, como por ejemplo el avance del mismo. Finalmente será el encargado de evaluar los posibles incrementos de producto que la parte desarrolladora del software le presente y realizar una retroalimentación de la misma para poder realizar las mejoras que considere necesarias, en caso de que lo fuese.

Responsabilidades

Actividad	Responsable	Consultado	Informado
Comunicación con el Cliente	Equipo de Analistas	Equipo de Desarrolladores, QA	Equipo de Ingenieros
Captación de Requerimientos	Equipo de Analistas	Equipo de Ingenieros	Equipo de Ingenieros
Desarrollo e Implementación	Equipo de Desarrolladores	Equipo de QA	Equipo de Analistas, Ingenieros
Autoevaluación y Retrospectivas	Equipo de Desarrolladores	Equipo de QA, Analistas	Equipo de Ingenieros
Mantenimiento y actualización	Cliente	Equipo de Analistas	Equipo de QA, Desarrolladores, Ingenieros
Evaluación de Crecimientos	Cliente	Equipo de Desarrolladores	Equipo de QA, Analistas, Ingenieros

Proceso de Gestión

Objetivos y Prioridades de Gestión

Durante el desarrollo del proyecto se tendrán en cuenta los siguientes objetivos y prioridades:

1. **Entregar un sistema de gestión funcional:** Lo primordial es desarrollar un sistema que cumpla con todos los requisitos especificados.
2. **Satisfacción del cliente:** se debe asegurar que el software cumpla con las expectativas y necesidades del cliente.
3. **Calidad del software:** Es muy importante para la gerencia la fiabilidad, el rendimiento, la seguridad y la capacidad de mantenimiento del software.
4. **Cumplimiento de plazos:** Mantenerse dentro del cronograma planificado es crucial para la entrega puntual y para evitar retrasos costosos.
5. **Colaboración del equipo:** Fomentar la colaboración y la comunicación efectiva dentro del equipo de desarrollo para mantener la eficiencia y la moral del equipo.

A medida que transcurra el proyecto y nuestros objetivos y prioridades cambien/evolucionen los iremos identificando en el presente apartado.

Supuestos, dependencias y restricciones

Supuestos:

- El cliente está predispuesto para la comunicación continua.
- Disponibilidad de recursos para iniciar el proyecto.
- Aceptación de cambios.

Dependencias:

- Adquisición de licencias.
- Capacitación de personal.

Restricciones:

- Tiempo y recursos.
- Normativas del país.

Gestión de Riesgos

Este apartado no aplica.

Proceso Técnico

Métodos, herramientas y técnicas

Para el análisis y desarrollo de este proyecto hemos decidido aplicar métodos basados en ideologías ágiles, lo cual nos permitirá llevar a cabo el mismo, haciendo utilización del marco de trabajo definido por "SCRUM". Por otra parte haremos utilización de la notación UML para la representación y comprensión del dominio, específicamente los "Diagramas de Clase" que provee dicha notación, así como cualquier otro que nos sea de ayuda durante el transcurso del proyecto.

Para la implementación haremos utilización del lenguaje de programación “Java” version 20.x, acompañado de distintas tecnologías como:

- **Framework Angular** (frontend)
- **Framework Spring Boot** (back end)
- **PostgreSQL** (base de datos)

Por otra parte, utilizaremos distintas técnicas que nos permita identificar los requerimientos del cliente y que podamos reconocer la complejidad de los mismos para así lograr una correcta organización en las actividades a desarrollar. Las técnicas en cuestión son:

- **Historia de Usuario:** Técnica utilizada para la captura y representación de los requerimientos.
- **Poker Planning:** Técnica para llevar a cabo la estimación de los requerimientos y conocer la complejidad de los mismos.

Documentación de Software

- **Product Backlog:** es un artefacto para la visualización de las necesidades individuales del product owner y donde el equipo Scrum puede saber el alcance del proyecto.

Gestión de cliente

- Registrar cliente
- Consultar cliente
- Programar cita con cliente
- Visualizar calendario
- Enviar confirmaciones de citas

Gestión de órdenes de trabajo

- Crear orden de trabajo
- Asignar técnico
- Consultar órdenes de trabajo
- Consultar estado de orden de trabajo

Gestión de Stock

- Registrar entrada de repuesto
- Registrar salidas del repuesto

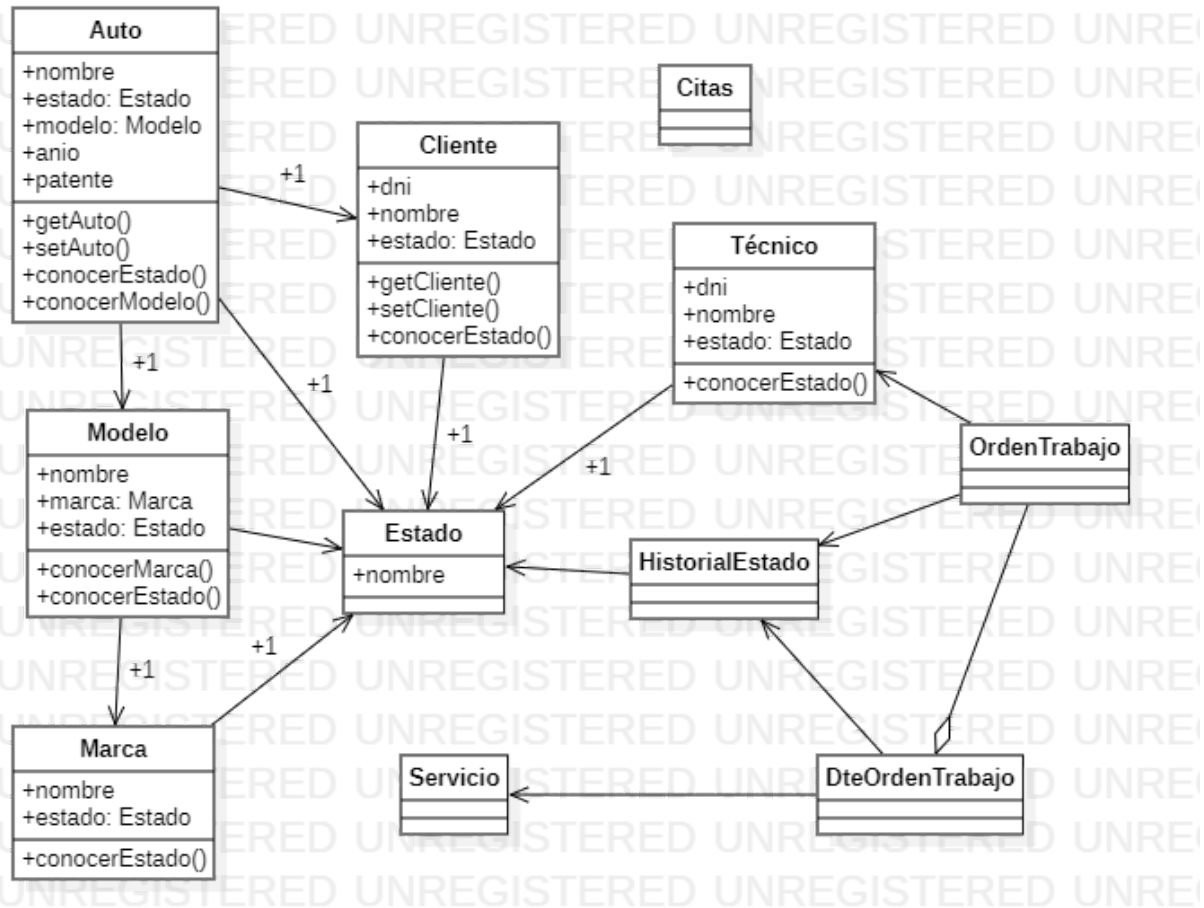
Informes y Estadísticas

- Generar informe de ventas periódicas
- Generar servicios más solicitados

Gestión de vehículos

- Registrar vehículo
- Consultar vehículos
- Registrar marca

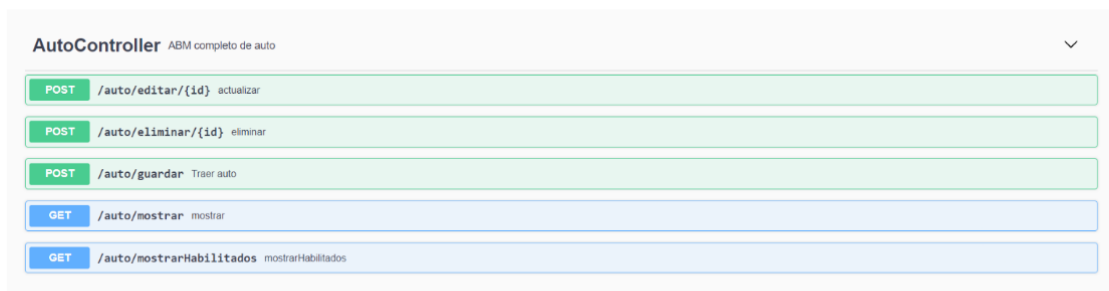
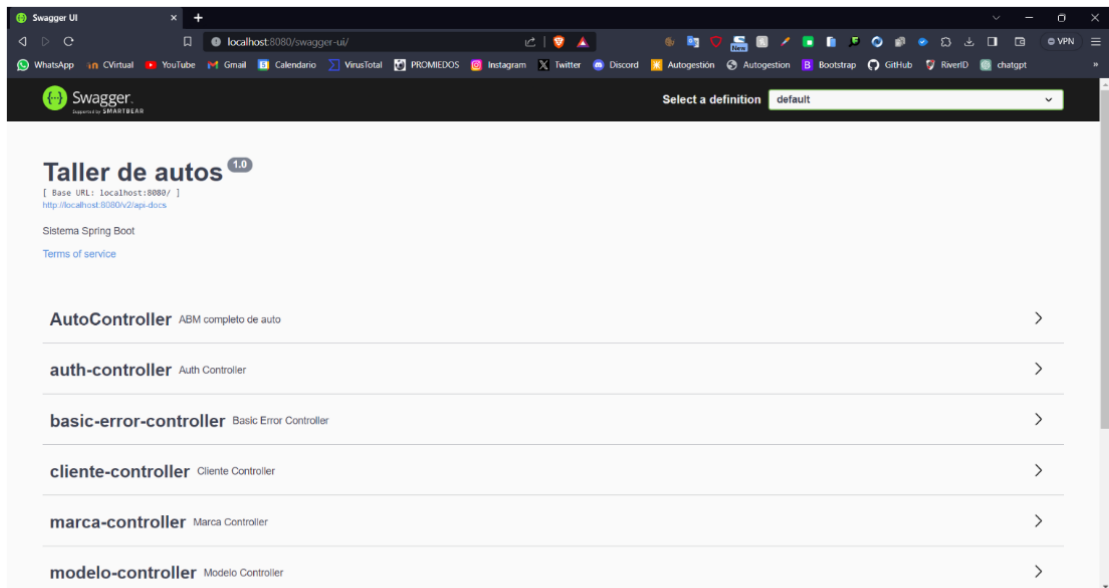
- **Diagrama Entidad-Relación:** es un tipo de diagrama de flujo que ilustra entidades, personas, objetos o conceptos que intervienen en el sistema y se relacionan entre sí. Su uso principal será para el diseño y depuración de bases de datos.



- **Diagrama de Paquetes:** Cada paquete representa una unidad de gestión del product backlog, con las historias de usuario que se tienen en cuenta.



- **Documentación de APIs:** Habíamos implementado [Swagger](#) para realizar la documentación, pero teníamos que bajar la versión del proyecto para que sea compatible, por lo que decidimos no dejarlo, pero sacamos captura de como eran nuestros endpoints.



POST /auto/editar/{id} actualizar Try it out

Name	Description
id required integer(\$int32) (path)	id <input type="text" value="id - id"/>
model required object (body)	model Example Value Model <pre>{ "anio": "string", "cliente": { "dni": 0, "estado": true, "id": 0, "nombre": "string" }, "estado": true, "id": 0, "modelo": { "estado": true, "id": 0, "marca": { "estado": true, "id": 0, "nombre": "string" }, "nombre": "string" }, "patente": "string" }</pre>

Parameter content type:

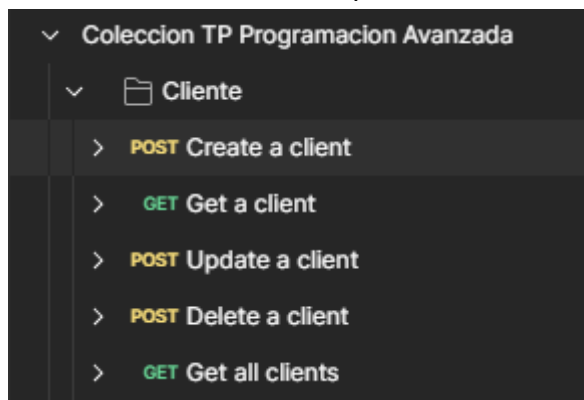
Responses Response content type: "JSON"

Code	Description
200	OK Example Value Model <pre>{ "body": {}, "statusCode": "ACCEPTED", "statusCodeValue": 0 }</pre>
201	Created
401	Unauthorized
403	Forbidden
404	Not Found

Ahora usamos [Postman](#), y en la documentación guardamos una colección con algunos [endpoints](#).

Coleccion Postman

Se encuentra subida en la parte de “Documentación” dentro del repositorio de GitHub.



Create a client:

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/cliente/guardar`. The request body is a JSON object representing a client. The response is a JSON object with a success message.

Request:

```
1 {
2   "estado": true,
3   "nombre": "Stefano Bortolin",
4   "dni": 43609975,
5   "telefono": 3634811755,
6   "email": "stefbortolin@gmail.comk",
7   "direccion": "Florida 868",
8   "observaciones": "Recomendado por Franco"
9 }
10
```

Response:

```
1 {
2   "message": "success"
3 }
```

Status: 200 OK Time: 112 ms Size: 444 B Save as example

Get a client:

The screenshot shows a REST client interface with a GET request to `http://localhost:8080/cliente/mostrar/Stefano Bortolin`. The response is a JSON object representing a client.

Request:

This request does not have a body

Response:

```
1 [
2   {
3     "id": 4,
4     "dni": 43609975,
5     "nombre": "Stefano Bortolin",
6     "direccion": "Florida 868",
7     "telefono": "3634811755",
8     "email": "stefbortolin@gmail.comk",
9     "estado": true,
10    "fecha_ultima_actualizacion": null,
11    "observaciones": "Recomendado por Franco"
12  }
13 ]
```

Status: 200 OK Time: 72 ms Size: 649 B Save as example

Update a client:

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/cliente/editar/4`. The request body is a JSON object with the following fields:

```
1 {
2   "id": 4,
3   "estado": true,
4   "nombre": "Stefano Bortolin",
5   "dni": 43609975,
6   "telefono": 3534811755,
7   "email": "stefbortolin@gmail.com",
8   "direccion": "Victoria Ocampo 1875",
9   "observaciones": "Recomendado por Francisco Botta"
10 }
```

The response is a JSON object with a success message:

```
1 {
2   "message": "success"
3 }
```

The status bar indicates a 200 OK response with a time of 32 ms and a size of 444 B.

Delete a client:

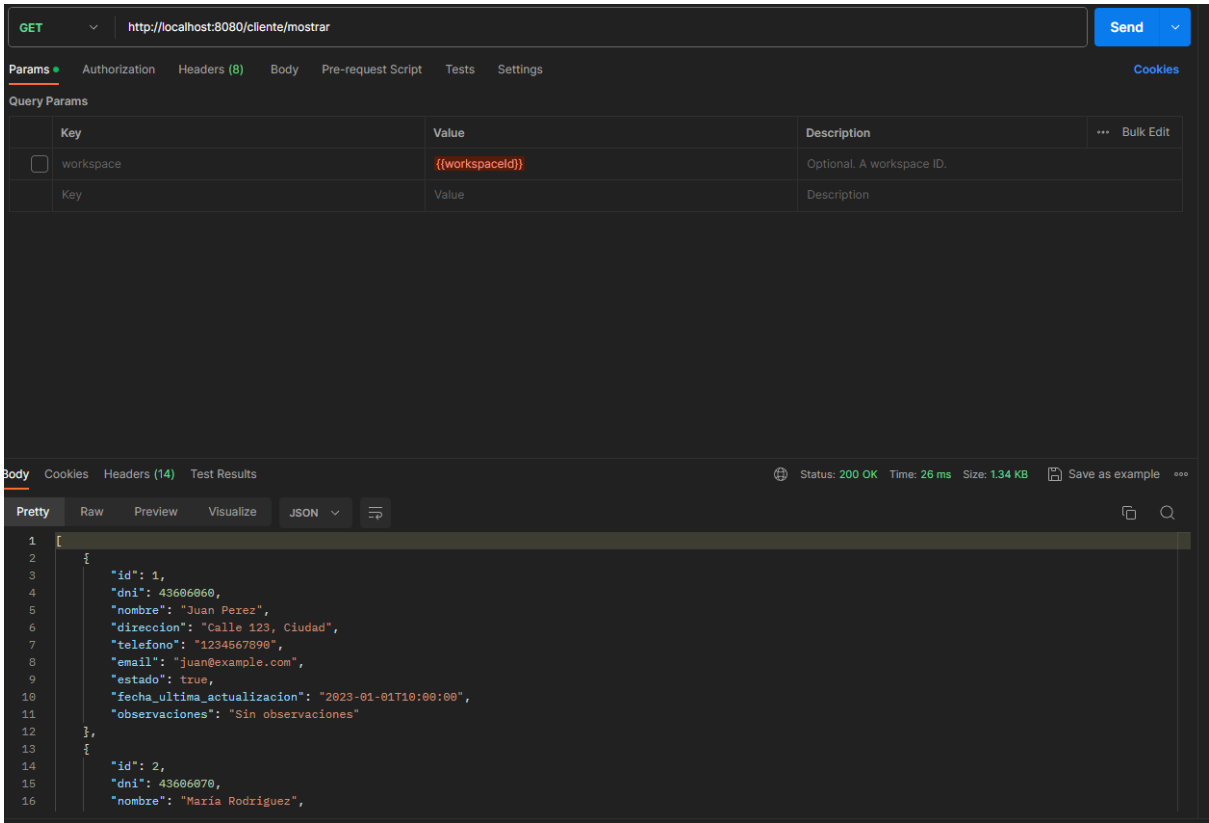
The screenshot shows a REST client interface with a POST request to `http://localhost:8080/cliente/eliminar/4`. The request body is empty, with the message "This request does not have a body".

The response is a JSON object with a success message:

```
1 {
2   "message": "success"
3 }
```

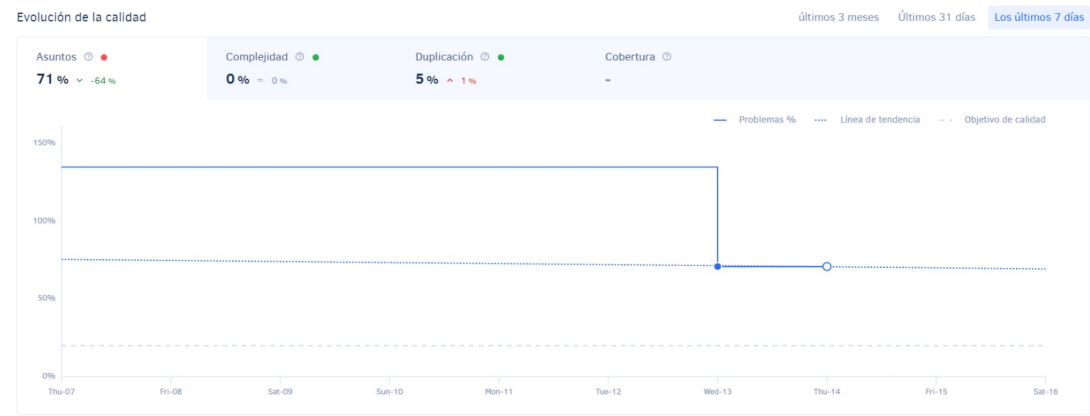
The status bar indicates a 200 OK response with a time of 41 ms and a size of 444 B.

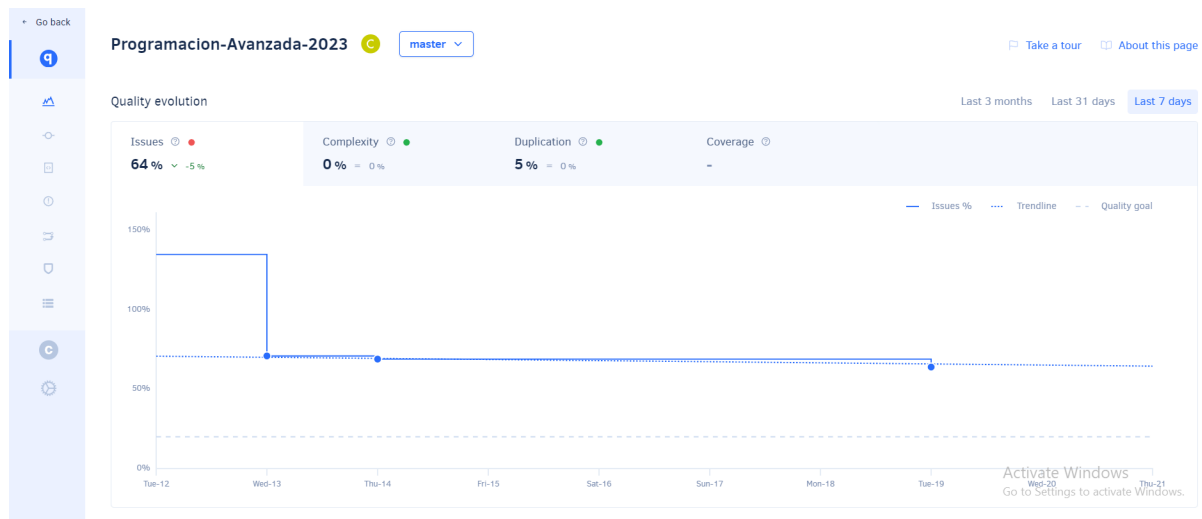
Get all clients:



Calidad

Para apoyarnos en este proceso utilizamos la herramienta [Codacy](#). A través de los issues realizamos algunas correcciones:





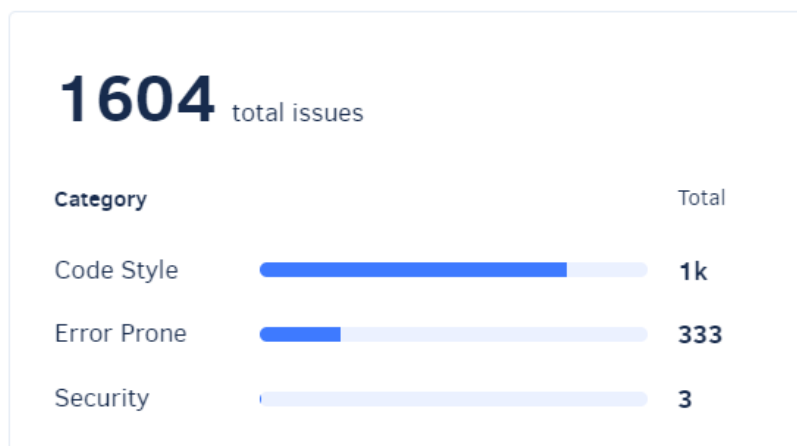
Issues ?

64% -5%

Obteniendo una mejora del 5%, ya que consideramos que las otras correcciones no eran justificadas.

[Commit con las mejoras.](#)

Issues breakdown



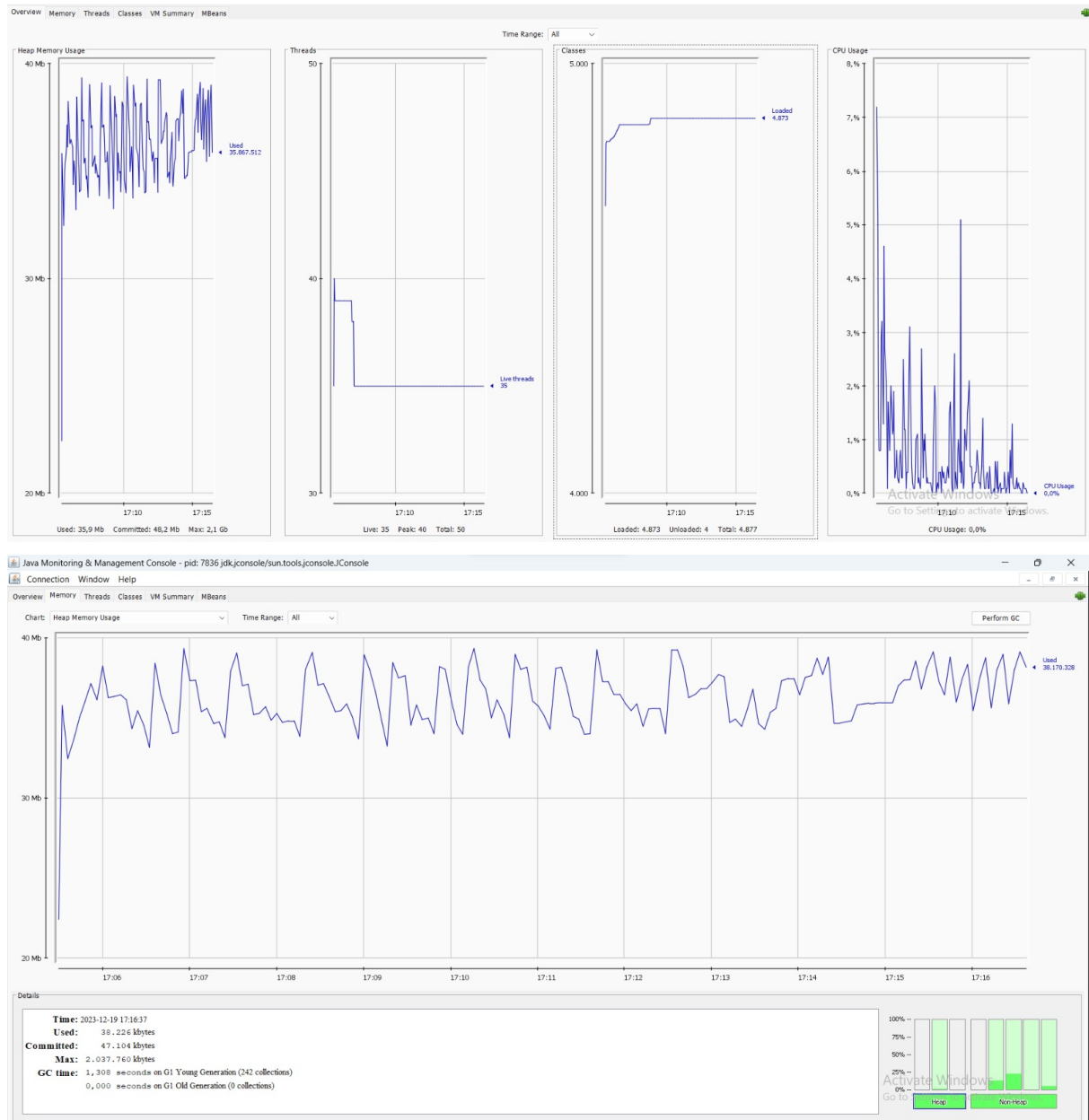
Plan de pruebas

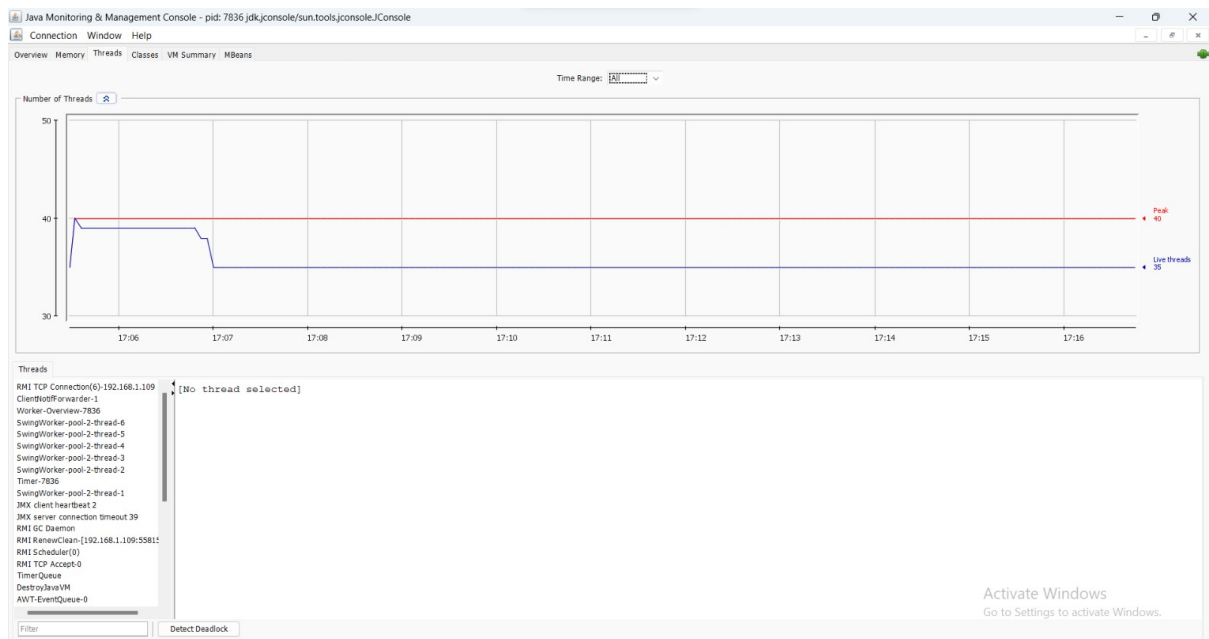
Realizamos un plan detallado de pruebas para el proyecto, que se va a basar en pruebas de Gestión de Órdenes de Trabajo y Gestión de Autos.

[Plan de pruebas](#)

Monitoreo

Para monitorear los recursos consumidos de la computadora, utilizamos una herramienta propia de Java, que viene con JDK llamada JConsole, obteniendo los siguientes gráficos:





Ramas GitHub

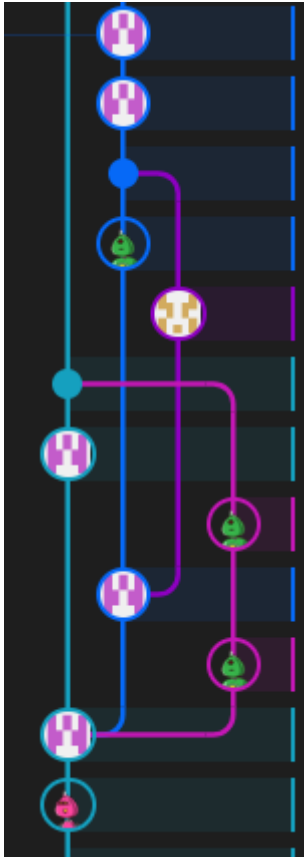
Para trabajar con ramas usábamos los siguientes criterios:

- La funcionalidad a desarrollar como nombre.
- Si hay más de una rama creada que estuvieran activas en desarrollo, agregamos el nombre de quien la creó para más transparencia o comunicación.
- Una vez terminemos la funcionalidad o creamos que no es más necesaria, la borramos a través de consola.
- Hacemos merge a la rama master.

Algunos ejemplos de las últimas ramas que usamos.

```
🔗 master 3812b897
🔗 franco-filtro-informe 3499cd44
🔗 editar-orden 649585b2
🔗 pagination-front d1d2af5e
🔗 botta-OrdenTrabajoFront ca30597b
```

Ramificación real:



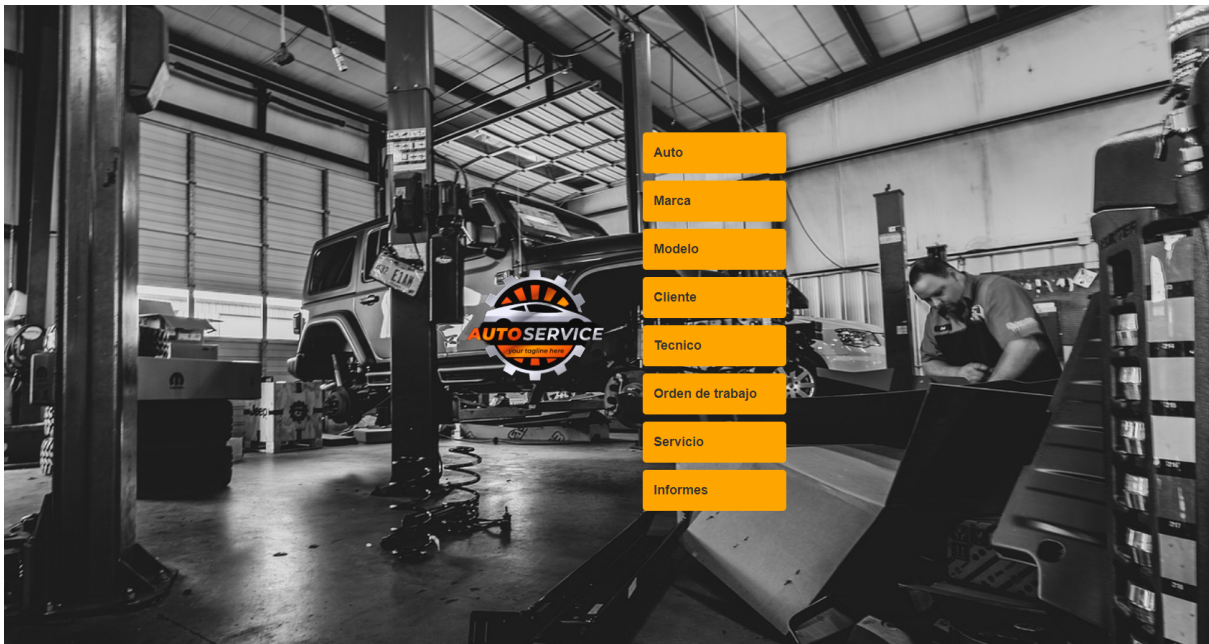
Video del funcionamiento del sistema

El video en el repositorio web no se puede visualizar por el tamaño, si se descarga el repositorio, en la computadora si se puede visualizar.


[Link hacia video](#)

Capturas del interfaz del sistema

Home:



Auto:



AutoMarcaModeloClienteTecnicoOrdenServicioInforme

Auto

Cliente

Patente

Año

Marca

Modelo

Crear


Cancelar

Buscar

ID	Patente	Año	Modelo	Marca	Cliente	Acciones
1	ABC123	2020	Corolla	Toyota	Juan Perez	<div><div></div><div></div><div>VER</div><div>+</div></div>
2	XYZ987	2018	Fiesta	Ford	María Rodriguez	<div><div></div><div></div><div>VER</div><div>+</div></div>
3	DEF456	2022	Cruze	Chevrolet	Carlos Gutierrez	<div><div></div><div></div><div>VER</div><div>+</div></div>

Items per page: 100 of 0<>

Marca:



Auto

Marca

Modelo

Cliente

Tecnico

Orden

Servicio

Informe

Nombre


Crear

Cancelar

ID	Nombre	Acciones
1	Toyota	<div><div></div><div></div></div>
2	Ford	<div><div></div><div></div></div>
3	Chevrolet	<div><div></div><div></div></div>

Items per page: 101 ~ 3 of 3<>

Modelo:



Auto

Marca

Modelo

Cliente

Tecnico

Orden

Servicio

Informe

Nombre

Crear

Cancelar


Marca

Ford

ID	Nombre	Marca	Acciones
1	Corolla	Toyota	<div><div></div><div></div></div>
2	Fiesta	Ford	<div><div></div><div></div></div>
3	Cruze	Chevrolet	<div><div></div><div></div></div>

Items per page: 100 of 0<>

Cliente:



AutoMarcaModelo**Cliente**TecnicoOrdenServicioInforme

Cliente

Nombre

DNI

Teléfono (caract sin 0 y sin 15)

Email

Dirección

Observaciones

Crear

Cancelar


Buscar

ID	Nombre	DNI	Telefono	Email	Dirección	Última orden	Acciones
1	Juan Perez	43606060	1234567890	juan@example.com	Calle 123, Ciudad	2023-01-01	<div><div></div><div></div><div>VER</div></div>
2	María Rodriguez	43606070	9876543210	maria@example.com	Av. Principal, etc.	2023-01-02	<div><div></div><div></div><div>VER</div></div>
3	Carlos Gutierrez	43606080	5555555555	carlos@example.com	Otra dirección	2023-01-03	<div><div></div><div></div><div>VER</div></div>

Items per page: 100 of 0<>

localhost:4200/home/modelo

Tecnico:



AutoMarcaModelo**Cliente**Tecnico**Orden**ServicioInforme

Tecnico

Nombre

DNI

Crear

Cancelar

ID

DNI

Nombre

Acciones

1	12345	Juan Perez	<div><div></div><div></div></div>
2	54321	María Gonzalez	<div><div></div><div></div></div>
3	67890	Carlos Gutierrez	<div><div></div><div></div></div>

Items per page: 100 of 0<>

localhost:4200/home/auto

En el video no se mostró la funcionalidad, pero en la pantalla “Orden de trabajo”, cuando la orden está finalizada, podemos emitir la factura en formato PDF, que se visualiza de la siguiente manera:

