**Logotipo, nombre de la empresa

Descripción generada automáticamenteForma

Descripción generada automáticamenteDesarrollo del proyecto**

*Procesos de Ingeniería del Software*

Curso 2023-2024

**Autor:** Alejandro Navarro de la Cruz

Contenido

[Introducción y objetivos del Proyecto 4](#_Toc147507943)

[Sprint 1: Arquitectura base y puesta en marcha 4](#_Toc147507944)

[Reunión de preparación del Sprint 4](#_Toc147507945)

[Desarrollo del Sprint 4](#_Toc147507946)

[Reunión de revisión 5](#_Toc147507947)

[Anexos 6](#_Toc147507948)

[Diagrama de secuencia: Agregar Usuario 6](#_Toc147507949)

Índice de figuras

[Ilustración 1: Modelo de dominio inicial 4](#_Toc147507926)

[Ilustración 2: Diagrama de arquitectura inicial 4](#_Toc147507927)

[Ilustración 3: Gráfica de Burndown del sprint 1 5](#_Toc147507928)

[Ilustración 4: Diagrama de secuencia AgregarUsuario 5](#_Toc147507929)

# Introducción y objetivos del Proyecto

En este documento se presenta el desarrollo de un proyecto para la asignatura de Procesos de Ingeniería del Software cuyo objetivo principal es desarrollar una aplicación basada en servicios con registro e inicio de sesión de usuarios.

En las siguientes secciones, se detallarán los aspectos más esenciales del proyecto separados por Sprint ya que el proyecto tiene un enfoque ágil.

# Sprint 1: Arquitectura base y puesta en marcha

## Reunión de preparación del Sprint

En este sprint, se comenzó a poner en marcha el proyecto, escogimos como herramienta de modelado StarUML y en ella creamos dos diagramas que comentaré más adelante. Y se empezó a diseñar e implementar la arquitectura base del proyecto.

Creamos un repositorio en el conocido software de gestión de proyectos, Jira y definimos dos épicas, desarrollo y documentación, de las cuales colgarán las historias asociadas a cada épica respectivamente.

Detallamos las tareas relacionadas con las historias y estimamos las horas para intentar llevar un control unificado.

## Desarrollo del Sprint

Durante el desarrollo del Sprint hemos ido cambiando la solución para ir puliendo la solución inicial. Primero implementamos la lógica, las pruebas y el modelo, luego el backend correspondiente a los artefactos que integraremos (nuestro modelo y las pruebas) en el servidor. Expusimos la lógica en la capa Rest e implementamos un cliente de comunicación con dicha capa.

Finalmente implementamos el componente de visualización para mostrar los formularios y desplegamos nuestra aplicación en Cloud Run (<https://procesos2324-5rfb3lj4yq-ew.a.run.app/>).

Modelamos el diagrama inicial de dominio que tendrá nuestra aplicación, el cual tendrá un Sistema que tendrá 1 o muchos Usuarios. Inicialmente Usuario solo tiene un Nick, pero posteriormente en siguientes Sprints lo modificaremos según nuestras necesidades.

Texto

Descripción generada automáticamente con confianza media

Ilustración : Modelo de dominio inicial

También modelamos el diagrama de la arquitectura de nuestra aplicación que contará con una arquitectura MVC por parte del cliente con esa capa de comunicación con el API Rest, para mantener desacoplados los datos y las funcionalidades de nuestro sistema. Y un servidor NodeJS clásico de tres capas, aunque la conexión con MongoDB se ha quedado fuera del alcance de este Sprint.

Interfaz de usuario gráfica

Descripción generada automáticamente

Ilustración : Diagrama de arquitectura inicial

## Reunión de revisión

Mostrar incremento al PO  
Reunión de retrospectiva

Como ya he mencionado anteriormente, hemos utilizado Jira como herramienta de gestión de proyectos, pero cada semana dentro del sprint, hemos modificado la estimación de los PBI en el Sprint Backlog y casi siempre dicha reestimación ha sido al alza.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Ilustración : Gráfica de Burndown del sprint 1

# Anexos

## Diagrama de secuencia: Agregar Usuario

En este diagrama de secuencia se muestran los pasos que ocurren en el sistema cuando le llega una petición para agregar un Usuario al sistema. Como vemos, cuando llega al sistema una petición, controlamos el evento de click en el elemento de visualización desde controlWeb.js, y creamos una instancia del clienteRest que se comunica con la capa de presentación de nuestro servidor y ésta con nuestra lógica, ubicada en modelo.js. Actualmente, como no es una solución definitiva y el registro de usuarios cambie, no he visto oportuno representar todas las alternativas de la lógica.

Texto

Descripción generada automáticamente con confianza media

Ilustración : Diagrama de secuencia AgregarUsuario