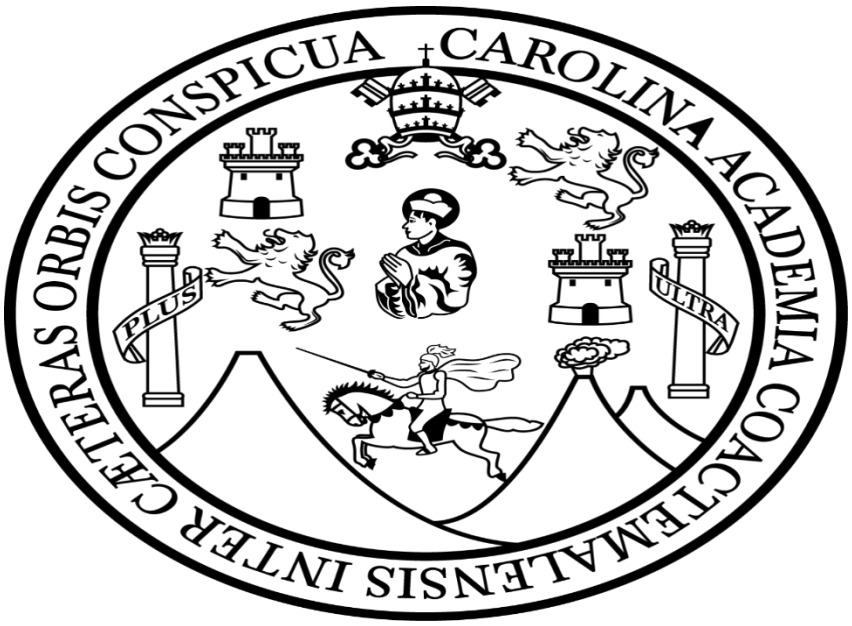
Universidad de San Carlos de Guatemala
Facultad de ingeniería
Escuela de ciencias y sistemas
Análisis y diseño de sistemas 2

Documentación AYDrive Fase3

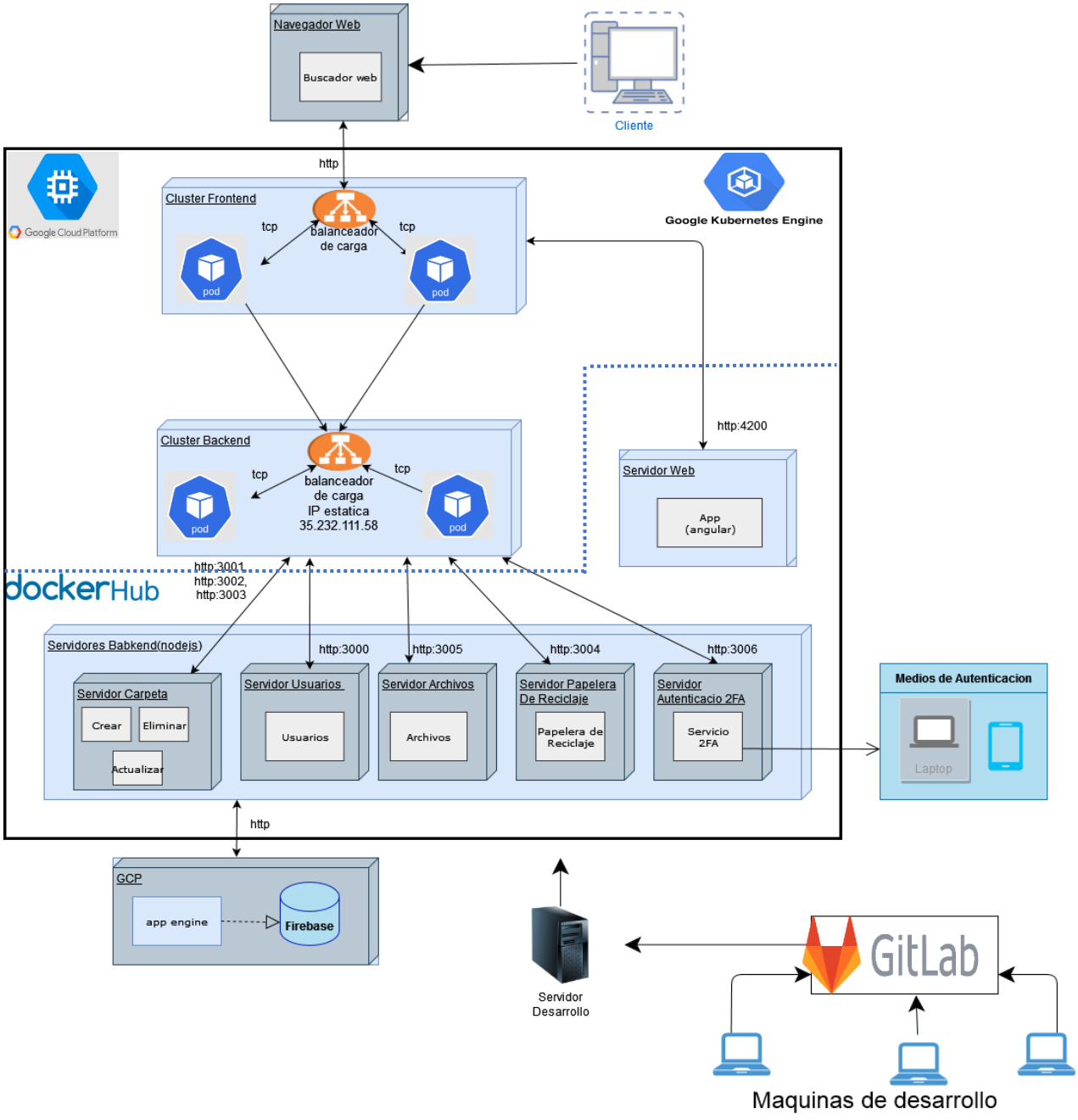


Juan Carlos I Alonzo Colocho
Davis Francisco Edward Enríquez
Selvin Lisandro Aragón Pérez
Edson Armando Guix Manuel
Kelvin Manfredy Vasquez Gomez

29 de octubre de 2021, Guatemala.

Diagrama de diseño arquitectónico:

El diseño arquitectónico se interesa como debe organizarse un sistema y como debe diseñarse la estructura global de ese sistema. Este diagrama muestra la forma en que se organiza el sistema como un conjunto de componentes en comunicación.

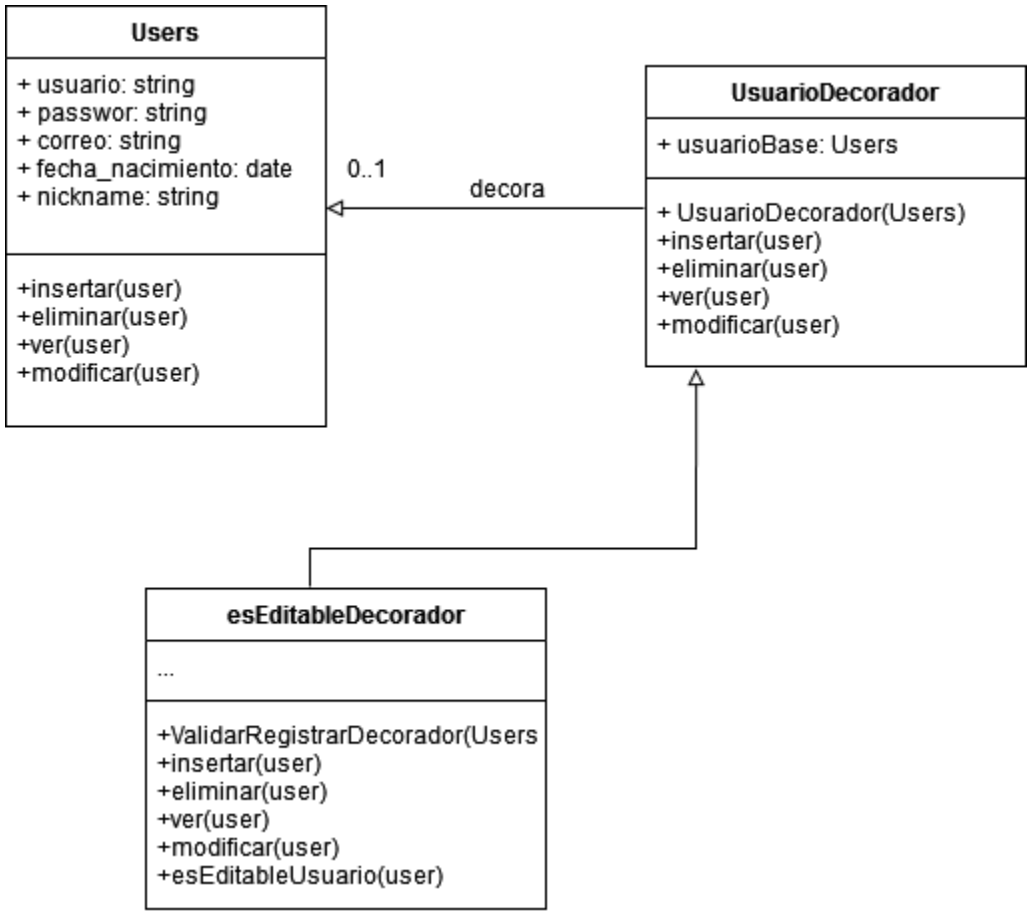


Diagramas de Clases:

Patrón Decorador:

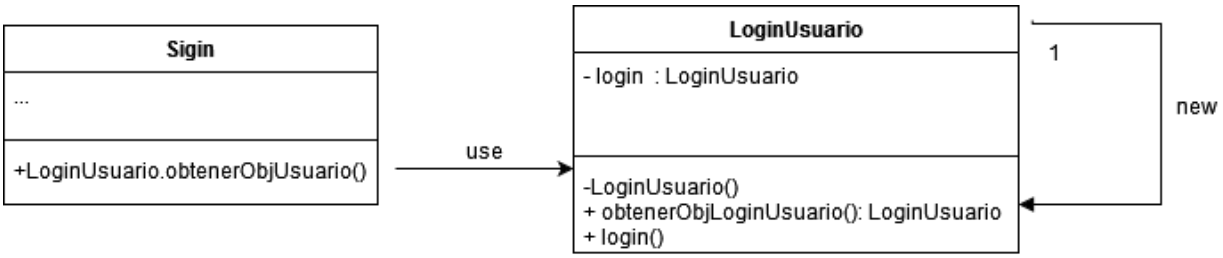
Utilizamos el patrón decorador para extender la funcionalidad de los usuarios, el usuario tendrá una nueva opción donde puede activar o desactivar la función que le permite editar o no la información del perfil.

***El proyecto está en JavaScript**



Patrón Singleton:

Utilizamos el patrón singleton en la clase login, donde se agrega un método que devuelve el objeto login, con esto aseguramos que la clase tiene una sola instancia y que solo se va a inicializar cuando sea requerido por primera vez.



Patrón Factory:

Utilizamos el patrón factory, para poder autenticarnos de 2 formas diferentes, una forma es activando el 2FA, donde el método enviara información para poder loguearse al correo y teléfono registrado por el usuario.

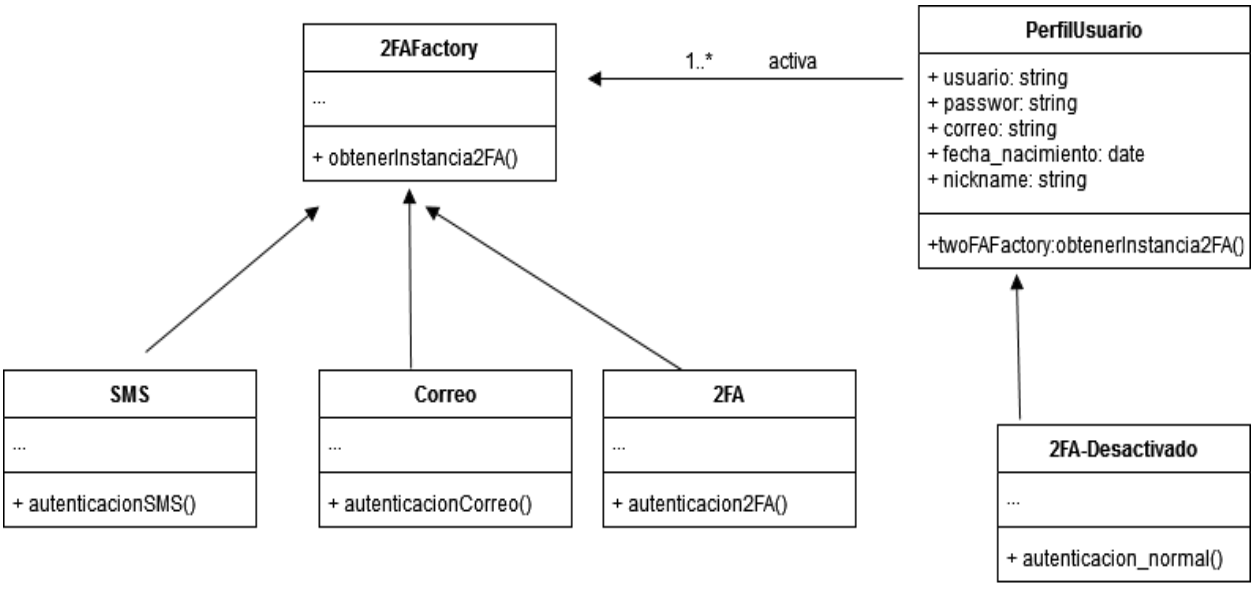


Diagrama Devops:

A continuación, se detallan los pasos realizados por todo el proceso de integración y despliegue continuos, en el diagrama los pasos serán representados con un valor numérico que indica el número de paso para una mayor comprensión.

Nota: todos los pasos que se detallaran se realizan automáticamente por medio de un script, los únicos pasos que se realizan manualmente es la fusión de las ramas.

Integración continua:

1. En la rama feature se crea una nueva funcionalidad, después de completarla se desea agregar el incremento a la rama de desarrollo(develop).
2. Con la ayuda de la herramienta de automatización gitlab-runner, cuando se detecta una petición de fusión(merge) a la rama de desarrollo(develop), se contruyen las imágenes de los contenedores donde se realizarán las pruebas.
3. Se ejecutan las pruebas unitarias y funcionales consecutivamente sobre los contenedores, si todas las pruebas pasan satisfactoriamente el flujo devops sigue su curso, caso contrario, el flujo se detiene.
4. Los contenedores se suben a un repositorio en la nube(docker hub).
5. Después de haber pasado las pruebas satisfactoriamente y los cambios subidos al repositorio de imágenes de docker, se procede a realizar la fusión de la rama de desarrollo a la rama de producción(main).

Despliegue continuo:

6. El despliegue inicia con el merge de la rama de desarrollo a la rama de producción, se descargan las imágenes de los contenedores del repositorio de imágenes.
7. Con la ayuda de un orquestador de contenedores como kuberntes y el pipeline de gitlab, ejecutara un script que aprovisionara la aplicación en dos 2 pods para backend y 2 para frontend, de esa forme se logra desplegar a producción las nuevas versiones en el sistema de forma automatica
8. Kubernetes mantiene un monitoreo sobre algún cambio que suceda en los contenedores para automatizar el despliegue.

El diagrama de devops se muestra a continuación, en la siguiente página.

Integración continua: