# PD 2-3

# FASES DE DESARROLLO Y MODELOS DE CICLOS DE VIDA

## Habilidades Técnicas

- Comunicación eficiente:

Una persona debe tener la habilidad, tanto de transmitir la idea o información, así como de recibirla.

- Buen cuestionamiento y análisis de los requerimientos:

Cuando se analizan los requerimientos y las exigencias del cliente, debemos cuestionar y pensar más allá de la información que recibimos, capaz de encontrar inconsistencias o ambigüedades en los requisitos, expectativas o pedidos a realizar en el proyecto o en los mismos requisitos planteados por el equipo de desarrollo.

- Abstracción visual:

Tenemos la certeza de que poder representar esas ideas en ocasiones abstractas de manera visual es una habilidad indispensable sobre todo en la hora de comunicarse con el cliente, va de la mano con la habilidad de la comunicación.

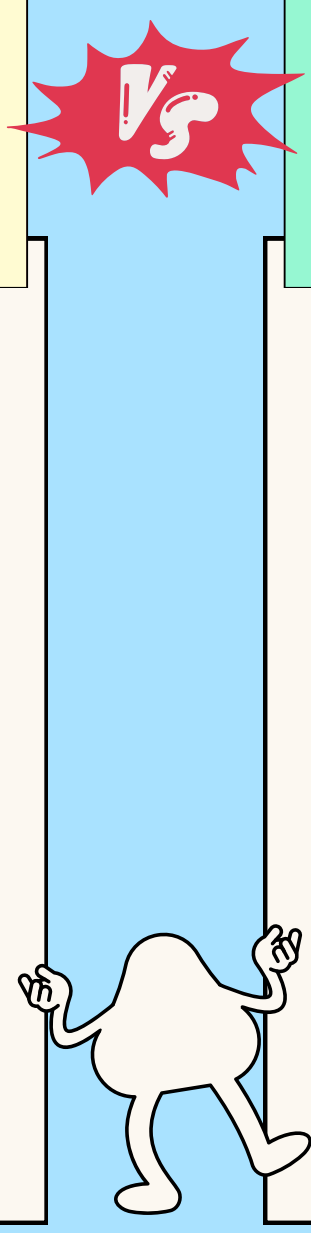
## Especificación de casos de uso

Ventajas:

Proporciona una descripción formal y completa de los requisitos funcionales del sistema.
Es útil para comunicar los requisitos al cliente y a los desarrolladores.

Desventajas:

Puede ser laborioso y complejo de desarrollar.
Puede ser difícil mantener la especificación actualizada a medida que cambian los requisitos.




## Historias de usuario

Ventajas:

Fomentan la colaboración entre el cliente y los desarrolladores.
Son flexibles y pueden adaptarse a los cambios en los requisitos.

Desventajas:

Pueden no proporcionar una descripción completa de los requisitos funcionales del sistema.
Pueden ser difíciles de priorizar y estimar.

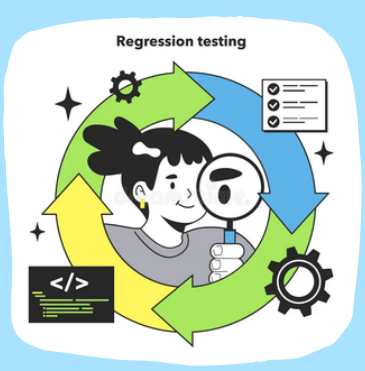
# Producto de la etapa de diseño

Para nuestro proyecto, el diseño de software es una parte fundamental para su éxito, ya que nuestra aplicación está diseñada para un público menos familiarizada con tecnología.

Para nuestro proyecto, utilizamos wireframes en Figma para crear un boceto del diseño de nuestra aplicación con una fidelidad media para lograr observar algún problema o deficiencia en nuestra interfaz de usuario. Durante el mismo, logramos observar que era poco intuitivo el cómo llegar la pantalla de actividades entre tantas pantallas de opciones y otras funciones. Decidimos implementar una barra inferior con pestañas para el inicio y ajustes.

# Técnica para incluir pruebas en el proceso de desarrollo

Las pruebas de regresión se enfocan en asegurar que los cambios no afecten negativamente el funcionamiento existente. Se llevan a cabo después de cambios significativos o al actualizar el software para garantizar su correcto desempeño.




Se inician identificando y documentando los requisitos, definiendo casos de prueba y seleccionándolos según su importancia. Luego, se crea una suite de pruebas considerando las áreas modificadas y las invariables. Posteriormente, se ejecutan los casos de prueba y se analizan los resultados para detectar discrepancias. Finalmente, se genera un informe con los resultados para evaluar la calidad del producto y mejorar en áreas específicas.

Un ejemplo basado en nuestra experiencia fue la comprobación del repositorio cuando se hacían grandes cambios desde nuestras bifurcaciones, así como durante la realización de nuestro prototipo de la interfaz cuando se agregaba una nueva sección en la herramienta figma.

# UN INGENIERO DE SOFTWARE TIENE...

## Habilidades

- Uso de softskills.
- Trabajo en equipo.
- Comunicación constante con el cliente y otros miembros del equipo.

## Conocimientos

- Codificación, lenguajes de programación.
- Comprensión de las pruebas de software y depuración.
- Gestión del ciclo de vida del software.

## Competencias

- Creatividad, liderazgo, iniciativa y respeto.
- Capacidad de análisis, síntesis, organización y planificación.
- Desarrollo de soluciones por medio del análisis de los usuarios y datos.

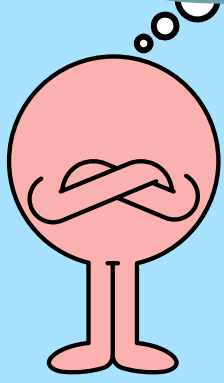
# Valores del manifiesto ágil

Individuos e interacción sobre procesos o herramientas, este valor entra en conflicto cuando no hay una buena comunicación del equipo; sin embargo, algunas soluciones son dar más peso a las interacciones con el equipo de manera presencial (de ser posible) para que la comunicación sea cara a cara y de este modo pueda haber un mayor entendimiento entre todos.

Software funcionando sobre documentación exhaustiva, cuando este valor entra en conflicto es porque no se le está dando la prioridad o importancia al funcionamiento del software, es por eso que las soluciones son enfocarse más en que funcione el software y no tanto en la documentación, sino que solo se documente las cosas verdaderamente relevantes y fuera de eso el equipo verifique constantemente mediante retroalimentaciones que el software esté funcionado correctamente.

Colaboración con el cliente sobre negocios de contratos, si hay conflictos respecto a este valor del manifiesto ágil, las soluciones que se pueden tomar en cuenta son mejorar o mantener constantemente la comunicación desarrolladores-clientes, esto de manera que ambas partes puedan colaborar y generar una relación de confianza, y después de ello, llegar a una negociación o acuerdo.

El cuarto y último valor es Responder al cambio sobre seguimiento a un plan, es muy común que sobre la marcha vayan cambiando las necesidades del cliente u otros aspectos que involucren un desvío del plan inicial, por ello es muy importante poder adaptarse a los cambios y ajustar nuestro proyecto de software a estos.

## Satisfacer a los clientes a través de la entrega temprana y continua (principio 1):

En nuestro proyecto de software integramos este principio al trabajar en equipo de manera eficiente mediante Sprints, cada uno con sus respectivos avances esperados y un periodo de tiempo definido, pero suficiente para completar dichos avances. Asimismo, nos apoyamos de la metodología Kanban por medio de la creación de un tablero visual que nos ayuda a tener total control y conocimiento de las tareas realizadas y por realizar para que podamos cumplir todas en el tiempo establecido y no exista atrasos en las entregas.

## Cambios en los requerimientos (principio 2):

Este principio lo implementamos en la fase de diseño. El uso de una metodología ágil permitió al equipo regresar a la fase de requerimientos para modificar y agregar algunos tras notar que hacían falta para el diseño del programa, posteriormente, seguimos con el diseño y lo adaptamos conforme a los nuevos requerimientos establecidos.

## La simplicidad (principio 10):

Este principio podemos integrarlo en nuestro proyecto en la etapa de diseño. No es necesario un diseño complejo, ya que incluso puede verse saturada la interfaz de tantos elementos o colores, sino que la simplicidad en el diseño supone un ahorro de tiempo, interfaces más ordenadas, cómodas y fáciles de usar para los usuarios, y lo más importante, sigue cumpliendo con los requerimientos y es funcional.