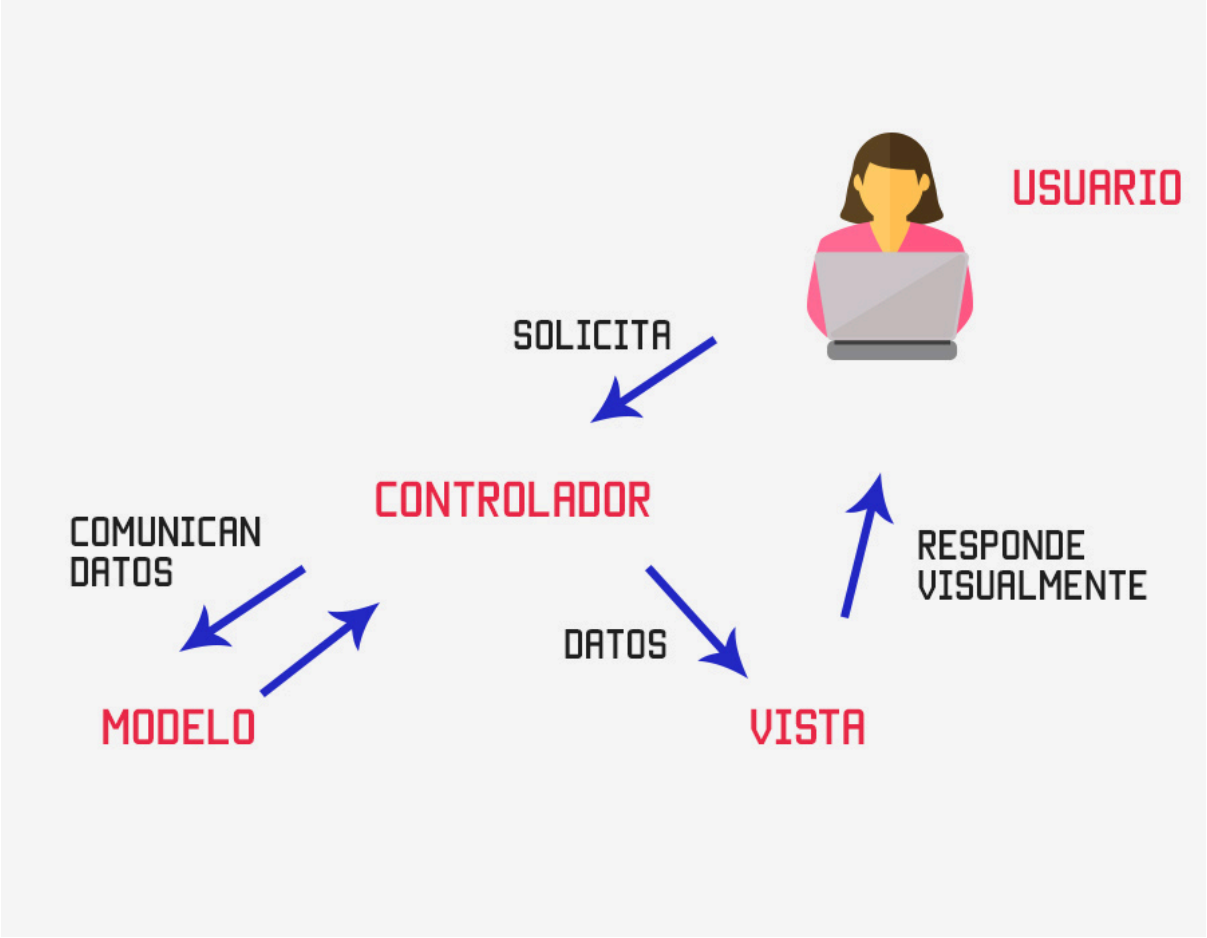
Avance del 90% del análisis de requisitos funcionales:
https://docs.google.com/document/d/124SqTS8QSwmhc2fbGOoD62lblE2Xc29_yhE1fvegmLI/edit?usp=sharing

Descripción inicial de la arquitectura de software:

La estructura de la aplicación se basa en el modelo de arquitectura Modelo-Vista-Controlador, que utilizando 3 componentes permite separar la lógica de la aplicación de la lógica de la vista de la aplicación.



Esta arquitectura permite separar los componentes de nuestra aplicación dependiendo de la responsabilidad que tienen, esto significa que cuando hacemos un cambio en alguna parte de nuestro código, esto no afecta a otra parte del mismo. Es decir, si modificamos nuestra Base de Datos, sólo deberíamos modificar el modelo que es quién se encarga de los datos y el resto de la aplicación debería permanecer intacta. Esto respeta el principio de la responsabilidad única. Es decir, una parte de tu código no debe de saber qué es lo que hace toda la aplicación, sólo debe de tener una responsabilidad.

Modelo: Se encarga de los datos, generalmente consultando la base de datos.

Vista: Representación visual de los datos, todo lo que tenga que ver con la interfaz gráfica va aquí.

Controlador: Recibe las órdenes del usuario y se encarga de solicitar los datos al modelo y de comunicárselos a la vista.

Al tener cada capa responsabilidades únicas, y no verse envuelto en demasiadas funcionalidades, el código se vuelve fácil de manejar, organizado y limpio, lo que lo hace mantenible. Además, al poder crear y modificar cada uno de estos archivos de manera independiente, podemos también agregar la escalabilidad y flexibilidad como otros requisitos no funcionales que la arquitectura permite cumplir.

También, esta modularidad que tiene la arquitectura permite tener un control centralizado, que facilita la implementación de medidas de seguridad, como validación de permisos o autenticación, así como poder optimizar cada parte del código de manera independiente sin que afecte otras partes del código.

> **Avance del 30% del diseño e implementación de requisitos funcionales:** Para la entrega de esta sección se requiere que envíes el código y los scripts de la base de datos. Se sugiere que se implementen los requisitos de mayor valor y riesgo, considerando el dominio técnico que tienes hasta el momento.

Los requisitos funcionales que fueron implementados para este avance incluyen:

1.1 Administrador inicia sesión

2.1 Alumno inicia sesión

1.3 Administrador consulta datos de alumnos

1.5 Administrador consulta profesores

Esto es un avance del 18% del total de nuestros requisitos, por lo que nos faltó un 12% para cumplir con esta entrega. Sin embargo, este porcentaje está en desarrollo a un punto que más del 30% de los requisitos ya están cerca de ser cumplidos en su totalidad y como equipo hemos establecido la meta de completarlos desde el 28 de marzo al 2 de abril. Esto se abordará en detalle en el plan de trabajo.

Scripts de la base de datos:
https://docs.google.com/document/d/1DpTmvLLLYvl5xQsGqE2BJAWa-21R6BWf83rrBMu5-fI/edit?usp=sharing

Código visible en github.

Diseño y ejecución de pruebas: Mismo link que el de el 90% de los requisitos funcionales.

Usabilidad:

Cambios resultantes e implementados por ViaDiseño:

- Medias horas en los horarios.
    - Se requirió que los horarios pudieran implementar las medias horas ya que en la Universidad tienen profesores que dan clases a las medias horas.
- Consistencia, uniformidad y seguimiento de lineamientos de diseño de acuerdo con la institución.
    - Se nos pidió que tuviéramos datos reales además de que siguiera cierto estilo la interfaz, por lo que ya presenta consistencia.
- Cambio del apartado de “Materias” por “Grupos” para el mejor entendimiento.
    - Se solicitó que en lugar de que hubiera un apartado “Materias” hubiera uno que dijera “Grupos” ya que lo que se hace es crear grupos y no materias. En este se incluyeron datos de la base de datos de las materias, profesores y salones.

Identificar el ambiente de producción: Sigue en negociación con el equipo de trabajo.

Plan de trabajo actualizado

:https://docs.google.com/spreadsheets/d/11kTj4KzY7oit2GIdL6TkJZF49sg6Cs6Wyp5jr8rian0/edit?usp=sharing

Aprendizaje adquirido:

- Es importante la arquitectura de software ya que implica muchas ventajas y permite tener un trabajo más limpio y pulido.
- La comunicación en el equipo es efectiva asimismo como el orden de trabajo respecto a la priorización; no obstante la manera en la que se hace el trabajo falla. Todos sabemos lo que tenemos que hacer y qué de eso es más importante, lo que aún nos falta es (que ya nos dimos cuenta y se cambiará para los próximos avances) es que nos falta inicialmente hacer el trabajo todos, todas las partes de un punto juntos para así después cada quien terminar los siguientes puntos. Esto porque todos hacíamos los puntos correspondientes sin embargo al no estar cada uno 100% concientes de todo lo que necesitaba, se alentó mucho el proceso.
