PARTE #1 // PRIORIZACIÓN

- **Clasificación de requerimientos funcionales usando la técnica MoSCoW**

    - **Must Have:** Esenciales para el sistema/Si no lo tiene no funciona
        - Registro e inicio de sesión
        - historial de mensajes
        - Chat individual
    - **Should Have:** Importantes, pero no urgentes.
        - Búsqueda de mensajes y contactos
        - Chat grupal
        - Notificaciones en tiempo real
    - **Could Have:** Deseables, pero no esenciales.
        - integración con calendario académico
    - **Ya Won't Have:** No se incluirán en este ciclo.
        - lista de contactos dinámica
- **Clasificación de requerimientos NO funcionales usando la técnica MoSCoW**
    - **Must Have:** Esenciales para el sistema/Si no lo tiene no funciona.
        - Seguridad
        - Disponibilidad
    - **Should Have:** Importantes, pero no urgentes.
        - Mantenibilidad
    - **Could Have:** Deseables, pero no esenciales.
        - Rendimiento
    - **Won't Have:** No se incluirán en este ciclo.

PARTE #2 // ESTIMACIÓN

Juan David Chacon Munoz

- ***Registro e inicio de sesión (3 días)***
    - Se debe crear formularios para registrar nuevos usuarios y permitirles iniciar sesión. Esto incluye verificar las credenciales consultando una base de datos , también requiere configurar validaciones básicas (como por ejemplo verificar que el correo sea único y manejo de estándares en las contraseñas).
- ***Lista de contactos dinámica (5 días)***
    - Se debe mostrar los contactos de un usuario desde la base de datos y asegurarse de mantener la lista actualizada por si hay cambios (como agregar o eliminar contactos). Esto puede hacerse con llamadas a la base de datos y algunas funciones para actualizar el frontend de forma dinámica, sin embargo se debe tener en cuenta el tiempo de capacitación para dominar las tecnologías necesarias para desarrollar esta funcionalidad.
- ***Chat individual (8 días)***
    - Implementar el envío y recepción de mensajes entre dos usuarios. Requiere diseñar cómo se almacenan los mensajes en la base de datos, crear las interfaces para enviarlos/recibirlos y posiblemente implementar una

comunicación en tiempo real (como WebSockets o algo por el estilo). Esto lleva más tiempo porque hay varios componentes interdependientes.

- ***Chat grupal (13 días)***
    - Se debe seguir la misma lógica del chat individual, pero con la complejidad añadida de gestionar grupos: permitir a los usuarios unirse o salir, manejar listas de participantes y enviar mensajes que lleguen a todos los miembros en tiempo real. La lógica adicional de los grupos eleva el tiempo necesario para desarrollar esta funcionalidad comparada con la del chat individual.
- ***Historial de mensajes (5 días)***
    - Permitir al usuario cargar mensajes anteriores desde la base de datos y mostrarlos de manera ordenada. Esto puede incluir una funcionalidad para "cargar más" mensajes antiguos si el usuario hace scroll. La complejidad depende de optimizar las consultas para manejar grandes volúmenes de datos.
- ***Notificaciones en tiempo real (5 días)***
    - Requiere que la aplicación notifique a los usuarios cuando ocurre un evento, como la llegada de un nuevo mensaje. Puede implementarse con tecnologías como WebSockets o notificaciones push. Si el objetivo es algo funcional y básico, podría lograrse en este tiempo asegurándose de que todo funcione bien.
- ***Búsqueda de mensajes y contactos (5 días)***
    - Se requiere diseñar consultas eficientes a la base de datos y posiblemente usar técnicas de indexación para mejorar el rendimiento. Esto lleva tiempo porque debe equilibrar la funcionalidad con la rapidez de las consultas.
- ***Integración con calendario académico (13 días)***
    - Requiere consumir una API de calendarios (como Google Calendar) o crear un sistema propio para gestionar eventos académicos. La integración incluye conectar eventos a cuentas de usuario, sincronizar datos y presentar la información en el frontend. Este proceso puede tomar tiempo debido a las configuraciones específicas de la API.

Diego Felipe Solorzano Aponte

- ***Registro e inicio de sesión (3 días)***
    - Se necesita conectar el proyecto con una base de datos, para esto hay que identificar los atributos de los usuarios y sus tipos. Además de implementar validaciones y peticiones get y post para manejar todo de manera adecuada.
- ***Lista de contactos dinámica (3 días)***
    - Como ya se cuenta con la base de datos y la conexión adecuada para manejar consultas, hay que dedicarse más al frontend para poder mostrar los contactos de la forma deseada
- ***Chat individual (13 días)***
    - Se necesita más tiempo para este requerimiento porque para que sea posible el conectar a los usuarios en tiempo real hay que tener más conocimiento de interfaces de comunicación y como conectarlas con la base de datos. Son más cosas a tener en cuenta
- ***Chat grupal (5 días)***

    - Se tarda menos que el chat individual porque se trabaja sobre este, sin embargo aún es necesario aumentar considerablemente la complejidad de su funcionamiento, porque hay más cosas a tener en cuenta ahora que existe una dinámica diferente en la comunicación y en las posibilidades de interacción para cada usuario.
- ***Historial de mensajes (5 días)***
    - Guardar los mensajes una vez que ya está la parte de la base de datos y del chat no debería tomar mucho tiempo, pero esto conlleva nuevas funcionalidades como poder buscar un mensaje específico en el historial del chat.
- ***Notificaciones en tiempo real (3 días)***
    - Requiere que la aplicación notifique a los usuarios cuando ocurre un evento, como la llegada de un nuevo mensaje. Puede implementarse con tecnologías como WebSockets o notificaciones push. Si el objetivo es algo funcional y básico, podría lograrse en este tiempo asegurándose de que todo funcione bien.
    - Se necesita un entendimiento amplio sobre WebSockets, pero habiendo trabajado ya con los chats en tiempo real no debería tardar tanto, pues estas notificaciones ya son una especie de chat entre dos partes.
- ***Búsqueda de mensajes y contactos (5 días)***
    - Acá se vuelve muy importante garantizar una búsqueda eficiente, lo cuál puede llevar tiempo implementar si son muchos datos.
- ***Integración con calendario académico (8 días)***
    - El proceso se facilita mucho gracias a las API, aún así esta se necesita saber manejar muy bien y se necesita mucho trabajo tanto en el frontend como en el backend

Daniel Aguilar Castro

- **Registro e inicio de sesión (8 días)**

  Creo que una semana es adecuada porque además del frontend, hay que hacer el backend para que se conecte a una base de datos.

- **Lista de contactos dinámica (3 días)**

  Desde mi inexperiencia en bases de datos me parece que es una tarea fácil. Por esto 3 días es adecuado según mi opinión.

- **Chat individual (21 días)**

  No tengo experiencia haciendo chats en tiempo real y a mi parecer es una tarea muy difícil.

- **Chat grupal (13 días)**

  ya que el chat individual está hecho. El chat grupal debería ser más fácil de implementar

- **Historial de mensajes (2 días)**

  Me parece que se podría implementar solo guardando el mensaje en una base de datos.

- **Notificaciones en tiempo real (1 días)**

  Seria mandar una notificación y no me parece algo muy difícil de implementar

- **Búsqueda de mensajes y contactos (3 días)**

  Sería una búsqueda SQL la cuál es eficiente y sencilla.

- **Integración con calendario académico (34 días)**

  Es difícil dimensionar el tiempo que requeriría implementar este requerimiento, es por esto que debería tener un plazo más amplio que cualquier Otro requerimiento.

Juan David Loaiza Reyes

Suponiendo las funcionalidades en un mínimo estado de funcionalidad, sin test ni nada parecido. Haciendo Backend y FrontEnd

- ***Registro e inicio de sesión (8 días)***
  - Además de crear el CRUD como mencionaron mis compañeros, es necesario tener en cuenta el manejo de las sesiones y la información necesaria a través de cookies de sesión o el uso de JWT. Incluir un manejo de roles y asegurarnos que las sesiones sean seguras para la transmisión de información.
- ***Lista de contactos dinámica (13 días)***
  - La complejidad de la lista de contactos dinámica viene determinada por la definición de dinámica. Suponiendo que estamos desarrollando directamente en python sin el uso de librerías el reto está en la construcción del filtrado y presentación de los contactos a una velocidad alta, es decir con buen rendimiento.
- ***Chat individual (13 días)***
  - La creación de un chat individual en tiempo real y claramente con historial puede parecer una tarea ardua. Pero la implementación de websockets (como lo hace whatsapp) reduce significativamente su complejidad. Además de que el almacenamiento se puede hacer en primera instancia a través de un JSON o un SQLite que se almacene directamente en el dispositivo. Haciendo que lo más complejo sea la implementación y verificación correcta del websocket para dos usuarios a la vez.
- ***Chat grupal (8 días)***
  - Suponiendo que tenemos ya hecho el chat individual a través de un websocket la complejidad de esta tarea reside en aumentar los usuarios recurrentes dentro de la funcionalidad, luego de esto sera necesario poder

ordenar los mensajes para su visualización, lo cual se puede hacer mediante el manejo de la fecha de envío, entre otras técnicas. Resultando en que no será una tarea difícil por la experiencia previa, pero si engorrosa y larga.

- ***Historial de mensajes (5 dia)***
    - Estando implícito en la funcionalidad del chat, realmente mediante el manejo de un SQLite o JSON con los mensajes de manera local (como hace whatsapp) debería ser más que suficiente. Donde el reto está en el previo diseño de esta BD.
- ***Notificaciones en tiempo real (5 días)***
    - Suponiendo que ya hicimos el servicio de mensajería. Este servicio funcionará como una extensión reducida de los mismos mensajes a traves de los websocket. Incluso la tarea se puede reducir a Cron jobs que visiten la API para revisar si hay alguna notificación nueva, haciendo que el trabajo esté en la creación de la API y revisión correcta.
- ***Búsqueda de mensajes y contactos (3 días)***
    - Suponiendo que ya tenemos los contactos dinámicos y que tenemos los mensajes en una BD SQLite, realmente esto funciona como un Query a las funcionalidades ya creadas.
- ***Integración con calendario académico (8 días)***
    - Suponiendo que vamos a usar algún calendario que use el usuario para integrarlo con el calendario académico. La dificultad está en el manejo del mismo tipo de datos y una homogeneización para poder poner estas fechas en los calendarios de los usuarios, haciendo un trabajo de traducción de los datos para ponerlos en una aplicación X de calendario.

CONSENSO

Después de compartir nuestras estimaciones y discutirlas, consideramos los siguientes tiempos para cada requerimiento:

- **Registro e inicio de sesión (5 días)**
- **Lista de contactos dinámica (8 días)**
- **Chat individual (13 días)**
- **Chat grupal (8 días)**
- **Historial de mensajes (5 días)**
- **Notificaciones en tiempo real (3 días)**
- **Búsqueda de mensajes y contactos (3 días)**
- **Integración con calendario académico (13 días)**