Universidad Nacional de Colombia - Sede Bogotá

Facultad de Ingeniería

Departamento de Sistemas e Industrial

Curso: Ingeniería de Software 1 (2016701)

Daniel Alejandro Duitama Correa
Edwin Felipe Pinilla Peralta
Juan Sebastian Umaña Camacho
Miguel Angel Martinez Fernandez

Planning poker individual

Registro e inicio de sesión:

- Daniel: 5 días, porque se va a trabajar con MongoDB que es algo con lo que yo y creo que mis compañeros no estamos muy familiarizados, el tema de la teoría, modelo e implementación.

- Edwin: 3 días, porque esto implica también la inicialización del proyecto, refiriéndose a inicializar esquemas o tablas en bases de datos, etc.

- Juan Sebastian: 5 días, ya que se tendría que crear la base de datos desde 0 e implementar la autenticación del programa, esto incluyendo metodologías de cifrado de contraseñas, tokens , etc.

- Miguel: Estimo que se demora 5 días porque se debe realizar una integración entre Firebase (si se va a usar) y distintos tipos de inicio de sesión.

- Consenso: 5 días

Búsqueda de contactos:

- Daniel: 1 día, ya teniendo la base de datos “montada”, seria con filtros o simples consultas.

- Edwin: 1 día, es un filtro sencillo y se puede solucionar con una lista dinámica.

- Juan Sebastian:1 día, ya que simplemente se debe hacer la respectiva query a la base de datos y manejar una lista en el frontend.

- Miguel:Estimo que se puede demorar 3 días, generando algoritmos eficientes y confiables.

- Consenso: 1 día

Chat individual:

- Daniel: 3 días, solo hacer la página web con algún framework y conectarlo con la base de datos, y probar una interfaz que sea cómoda.

- Edwin: 3 días, es una funcionalidad fácil de implementar al comienzo, tengo experiencia trabajando con ello.

- Juan Sebastian:5 días, ya que es una funcionalidad que debe ser implementada desde 0, usando métodos y protocolos específicos.

- Miguel:Estimo que se demora 8 días, ya que se debe utilizar un tipo de comunicación diferente, además de realizar las pruebas pertinentes.

- Consenso: 5 días

Historial de mensajes:

- Daniel: 3 días, por el mismo tema de cómo manejar la base de datos, cómo obtener los registros y ya no más sería mostrarlos.

- Edwin: 3 días, se debería solucionar al mismo tiempo que el chat individual, pues la existencia del chat implica la existencia del historial.

- Juan Sebastián:3 días, ya que solamente se debe implementar una manera de sincronizar el cliente con el servidor que almacena las conversaciones y renderizarlas en el frontend.

- Miguel:Estimo que se demora 5 días, considerando la selección de la base de datos a usar (relacional o no relacional) y la implementación de algoritmos eficientes para realizar búsquedas.

- Consenso:3 días

Seguridad

- Daniel: 2 días, librerías de encriptación RSA y usarlo en la base de datos.

- Edwin: 2 días, si utilizamos un contenedor de docker con una instancia ec2, nginx nos puede dar certificados ssl gratuitos junto a un control de peticiones sencillo, aunque es seguro, no asegura que no obtendremos peticiones maliciosas.

- Juan Sebastian: 3 días, ya que además de realizar el encriptado de extremo a extremo habría que realizarle tests a este encriptado para asegurar su fiabilidad.

- Miguel:Estimo que se puede demorar 3 días implementando algoritmos de cifrado y descifrado para los chats.

- Consenso:3 días

Rendimiento

- Daniel: 34 días, que serían los que estimo que dure el proyecto, en mi punto de vista hacer unos días de prueba de rendimiento, pero serian pocos, porque no se necesitan algoritmos complejos, solo la encriptación, tener una buena estructura de datos, y guardar los datos en la base. Ya sería tema de tener espacio suficiente en el servidor donde se guardaran los datos.

- Edwin: 24 días, es algo que se debe trabajar a lo largo del tiempo de desarrollo, esto implica buenas prácticas desde el principio

- Juan Sebastian:34 días, creo que el rendimiento de la aplicación debe ser un tema a tratar desde el día 1 hasta el final del proyecto, que estimo que estaria completo en 34 días de desarrollo, esto porque a medida que se implementan nuevas características estas pueden traer nuevos problemas de rendimiento y hacer los tests al final implicaría cambiar código antes escrito, lo cual es más difícil que cambiar código a medida que se realiza.

- Miguel:Estimo que se puede demorar 3 días, analizando los módulos críticos en términos de rendimiento y optimizándolos.

- Consenso:34 días

Mantenimiento:

- Daniel: 34 días, los que dura el proyecto.

- Edwin: 24 días, es algo que se debe trabajar a lo largo del tiempo de desarrollo, esto implica buenas prácticas desde el principio.

- Juan Sebastian:34 días, así como el rendimiento, el mantenimiento debe ser tratado desde el día 1 hasta finalizar el proyecto, para así, asegurar un código limpio y escalable sin necesidad de cambiar código antes escrito.

- Miguel:Para mí no aplica (N/A), no se puede estimar, usualmente la carga de trabajo para un sprint se basa en los puntos de historia y no tendría sentido estimar carga de trabajo por hacer código con buenas prácticas.

- Consenso:34 días

Chat grupal:

- Daniel: 3 días, parecido a el chat individual pero con varios usuarios.

- Edwin: 9 días, puede ser complejo evitar el cruce entre chats existentes, aunque lo ideal sería encapsular sockets en instancias grupales para evitar los cruces.

- Juan Sebastian: 3 días, ya que teniendo la implementación del chat individual, no debería ser demasiado complejo de escalar para añadir más personas al chat.

- Miguel:Estimo que se demora 3 días, teniendo en cuenta que ya se cuenta con un desarrollo previo para el chat individual.

- Consenso:5 días

Disponibilidad:

- Daniel: 3 días.

- Edwin: 2 días, es algo que implica cambios en infraestructura de nube dependiendo la cantidad de usuarios, pero es fácilmente trabajable preparando una escalabilidad, por ahora vertical, elástica.

- Juan Sebastian: 2 días, ya que el trabajo de despliegue no creo que sea tan complejo, sobre todo usando servicios web modernos como Vercel.

- Miguel:Estimo que se demora 3 días desplegando el aplicativo, ya que, si se utiliza un servicio web como AWS u OCI, cuentan con una infraestructura subyacente que asegura la disponibilidad.

- Consenso:3 días

Integración con calendario académico:

- Daniel: 3 días. Con una API.

- Edwin: 2 días, el API de google es bastante intuitivo y solo se necesita un token, pero no estoy seguro de que pasaría para integrarlo con varias personas al tiempo, en este caso con la entidad de la universidad

- Juan Sebastian: 3 días, ya que no tengo experiencia usando la API de google para la integración con Google Calendar.

- Miguel:Estimo que se puede demorar 3 días, dependiendo de la complejidad de utilizar el API de Google.

- Consenso: 3 días