**Ticket 1: Backend - Implementar Registro con Cuentas Externas (Google, Facebook)**

**Descripción:** Implementar el sistema de autenticación en el backend para que los usuarios puedan registrarse y autenticarse usando sus cuentas externas de Google o Facebook. La funcionalidad debe integrarse con los servicios de OAuth 2.0 de estas plataformas y manejar la verificación y creación de usuarios en la base de datos.

**Pasos para Desarrollo:**

1. **Integración con Proveedores de OAuth 2.0:**
   * Configurar credenciales de desarrollador en Google y Facebook para obtener los Client ID y Client Secret.
   * Implementar en FastAPI los endpoints necesarios para iniciar y manejar el flujo de OAuth 2.0.
   * Implementar la lógica para redirigir a los usuarios a la página de autorización de Google o Facebook.
2. **Intercambio de Tokens:**
   * Implementar la lógica para intercambiar el authorization code por un access token y refresh token.
   * Validar el access token y obtener información del usuario desde los endpoints de Google y Facebook.
3. **Registro o Inicio de Sesión:**
   * Verificar si el usuario ya existe en la base de datos utilizando el correo electrónico obtenido del proveedor de OAuth.
   * Si no existe, crear un nuevo usuario en la base de datos.
   * Si ya existe, autenticar al usuario y generar un token JWT para el manejo de sesiones.
4. **Seguridad y Manejo de Errores:**
   * Implementar validaciones para asegurarse de que los tokens sean válidos y no hayan sido comprometidos.
   * Manejar errores comunes como tokens expirados, acceso denegado, y problemas de red.
   * Implementar logs para capturar todos los eventos importantes y posibles errores.
5. **Pruebas:**
   * Crear pruebas unitarias y de integración para asegurar que todas las rutas y funcionalidades de OAuth 2.0 funcionen correctamente.
   * Validar que los tokens JWT generados sean válidos y seguros.
6. **Documentación:**
   * Actualizar la documentación de la API para incluir la información sobre los nuevos endpoints de autenticación.
   * Proveer ejemplos de cómo los usuarios pueden registrarse y autenticarse usando cuentas de Google y Facebook.

**Criterios de Aceptación:**

* Los usuarios deben poder registrarse e iniciar sesión usando sus cuentas de Google o Facebook.
* La cuenta debe ser creada automáticamente si es la primera vez que el usuario utiliza el sistema.
* El sistema debe manejar los errores comunes de OAuth (e.g., token expirado) y proporcionar mensajes claros al usuario.
* El token JWT debe ser generado y ser válido para futuras solicitudes.

**Ticket 2: Frontend - Implementar Selección de Servicio Específico**

**Descripción:** Desarrollar la funcionalidad en la interfaz de usuario que permita a los clientes seleccionar un servicio específico de un proveedor. Esta funcionalidad debe ser intuitiva y permitir la selección de fecha y hora de forma dinámica y visual.

**Pasos para Desarrollo:**

1. **Diseño de Interfaz:**
   * Crear un diseño UI/UX que permita la selección de servicios de forma clara y fácil de usar.
   * Implementar un calendario interactivo para la selección de fecha y hora.
2. **Integración con Backend:**
   * Conectar el frontend con los endpoints del backend que proporcionan la información sobre la disponibilidad de servicios.
   * Asegurarse de que las opciones de fecha y hora estén sincronizadas con la disponibilidad actualizada desde el backend.
3. **Validaciones:**
   * Implementar validaciones en el frontend para asegurar que la fecha y hora seleccionadas sean válidas y disponibles.
   * Desactivar automáticamente fechas y horas no disponibles para la selección del usuario.
4. **Manejo de Errores:**
   * Implementar mensajes de error claros y concisos para el usuario en caso de problemas de disponibilidad o fallos en la conexión con el backend.
5. **Pruebas:**
   * Realizar pruebas de usuario para asegurar la usabilidad y claridad de la interfaz.
   * Validar que la información presentada en la interfaz sea siempre la más actualizada posible.
6. **Optimización:**
   * Asegurar que la interfaz sea responsiva y funcione bien en dispositivos móviles.
   * Optimizar el rendimiento para minimizar el tiempo de carga y mejorar la experiencia del usuario.

**Criterios de Aceptación:**

* Los clientes deben poder seleccionar un servicio específico, así como la fecha y hora deseadas.
* La interfaz debe desactivar automáticamente las fechas y horas que no estén disponibles.
* El sistema debe proporcionar mensajes de error claros en caso de problemas de disponibilidad o conexión.
* La interfaz debe ser responsiva y funcionar sin problemas en dispositivos móviles.

**Ticket 3: Base de Datos - Desarrollar Sistema de Registro Manual con Correo y Contraseña**

**Descripción:** Crear la estructura de base de datos y lógica necesaria para permitir a los usuarios registrarse manualmente en la aplicación utilizando su correo electrónico y una contraseña segura. La funcionalidad también debe incluir la verificación por correo electrónico.

**Pasos para Desarrollo:**

1. **Diseño de la Base de Datos:**
   * Crear las tablas necesarias en PostgreSQL para almacenar la información de usuarios, incluyendo email, password\_hash, is\_verified, created\_at, updated\_at.
   * Implementar índices en los campos de correo electrónico para optimizar la búsqueda y garantizar la unicidad.
2. **Lógica de Registro:**
   * Implementar la lógica para registrar un nuevo usuario, incluyendo el hash de la contraseña utilizando un algoritmo seguro (e.g., bcrypt).
   * Crear un sistema para generar un verification\_token y almacenar la información en la base de datos.
3. **Manejo de Confirmación de Correo:**
   * Implementar la lógica para enviar un correo electrónico de confirmación con un enlace único para verificar la cuenta del usuario.
   * Crear un endpoint en el backend para recibir la solicitud de verificación y actualizar el campo is\_verified en la base de datos.
4. **Seguridad:**
   * Asegurar que las contraseñas estén siempre almacenadas en formato cifrado.
   * Implementar validaciones de seguridad como la longitud mínima de la contraseña, mezcla de caracteres, y protección contra ataques de fuerza bruta.
5. **Pruebas:**
   * Realizar pruebas unitarias para asegurar que el registro, hash de contraseñas, y verificación de correos funcionen correctamente.
   * Validar que el flujo de trabajo de registro y verificación sea robusto y seguro.
6. **Documentación:**
   * Documentar la estructura de la base de datos y los procedimientos de registro y verificación.
   * Proveer ejemplos de cómo deben interactuar las APIs con la base de datos para el registro de usuarios.

**Criterios de Aceptación:**

* Los usuarios deben poder registrarse usando un correo y una contraseña.
* Las contraseñas deben ser almacenadas de manera segura en la base de datos.
* El sistema debe enviar un correo electrónico de confirmación con un enlace de verificación.
* El enlace de verificación debe activar correctamente la cuenta del usuario.
* La base de datos debe reflejar el estado de verificación del usuario.