**Historias de Usuario**

**Historia de Usuario #1: Detección de Emociones Faciales**

**Descripción:**  
Como usuario, quiero que el sistema detecte mis emociones faciales en tiempo real, **para** mejorar mi experiencia.

**Criterios de Aceptación:**

* Precisión dentro del 70% – 80 %.
* Tiempo de respuesta menor a 1 segundo.
* El sistema debe ser capaz de detectar emociones en diferentes condiciones de iluminación (luz natural, luz tenue, etc.).
* La detección facial debe funcionar correctamente a diferentes ángulos de rostro (30°, 45°, etc.).
* El sistema debe identificar emociones en tiempo real sin que se presenten errores visuales en la interfaz del usuario.
* El sistema debe ser capaz de diferenciar entre emociones similares (por ejemplo, felicidad vs. sorpresa).

**Valor:** 400  
**Prioridad:** Alta  
**Estimación:** 12H

**Historia de Usuario #3: Análisis Emocional en Texto**

**Descripción:**  
Como usuario, quiero que el sistema analice las emociones en los textos de mis conversaciones con el chatbot, **para** registrar y responder adecuadamente.

**Criterios de Aceptación:**

* Análisis basado en palabras clave.
* Datos registrados con timestamps.
* El sistema debe manejar correctamente oraciones complejas y ambigüedades, identificando la emoción predominante.
* Los resultados emocionales deben ser coherentes entre texto y otras entradas (voz y rostro).
* La emoción detectada debe reflejarse en tiempo real en la interfaz, con un cambio inmediato cuando se analice un nuevo mensaje.
* La interfaz debe ser capaz de mostrar un desglose de las emociones detectadas (por ejemplo, "Felicidad (70%) y sorpresa (30%)").
* La calidad de la detección no debe verse afectada por el uso de abreviaturas o jerga.

**Valor:** 350  
**Prioridad:** Alta  
**Estimación:** 10H

**Historia de Usuario #4: Respuestas Dinámicas del Chatbot**

**Descripción:**  
Como usuario, quiero que el chatbot ajuste sus respuestas basadas en emociones detectadas, para mejorar la experiencia del usuario.

**Criterios de Aceptación:**

* Respuestas dinámicas ajustadas a las emociones detectadas.
* Las respuestas deben ser coherentes con la emoción detectada (por ejemplo, respuestas empáticas cuando se detecta tristeza o frustración).
* El chatbot debe ofrecer respuestas alternativas o preguntas de seguimiento si se detecta una emoción negativa.
* El sistema debe ajustar su tono de respuesta (si es posible) para que coincida con la emoción detectada (por ejemplo, tono más amigable para la felicidad).
* Las respuestas deben ser entregadas en menos de 1 segundo desde que se detecta la emoción.

**Valor:** 250  
**Prioridad:** Alta  
**Estimación:** 6H

**Historia de Usuario #5: Visualización de Patrones Emocionales**

**Descripción:**  
Como analista, quiero visualizar patrones de emociones en gráficos, **para** identificar tendencias emocionales.

**Criterios de Aceptación:**

* Visualizaciones disponibles por período y tipo de emoción.
* Los gráficos deben permitir al usuario filtrar emociones por fecha, usuario, o tipo de emoción.
* Las visualizaciones deben ser interactivas y permitir la exportación de los datos (CSV, Excel).
* Los gráficos deben ser dinámicos, con actualizaciones automáticas cuando el sistema detecta nuevas emociones.
* Los patrones deben visualizarse en diferentes escalas (mensual, semanal, diario).

**Valor:** 300  
**Prioridad:** Media  
**Estimación:** 10H

**Historia de Usuario #6: Implementación de Interfaz del Usuario y Chatbot**

**Descripción:**  
Como usuario, quiero una interfaz gráfica completa diseñada con Tailwind CSS y CSS personalizado que permita navegar por las funcionalidades principales del sistema, para interactuar de manera eficiente y agradable.

**Criterios de Aceptación:**

* + Las pantallas de inicio de sesión y registro deben ser claras, amigables y consistentes visualmente.
  + Deben incluir validaciones en el frontend y backend para correos electrónicos válidos y contraseñas con longitud mínima.
  + Los mensajes de error y confirmación deben mostrarse mediante modales.
  + Debe incluir botones para: Gestión de Usuario, Interacciones Previas, Chatbot y Cerrar Sesión.
  + Los botones deben incluir íconos representativos y texto descriptivo.
  + El diseño debe ser responsivo y adaptarse a dispositivos móviles y de escritorio.
  + Los botones deben redirigir correctamente a las pantallas correspondientes.
  + En la pantalla de chatbot se debe mostrar un flujo de mensajes entre el usuario y el chatbot, diferenciando los mensajes por colores o etiquetas.

**Valor:** 300  
**Prioridad:** Alta  
**Estimación:** 6H

**Historia de Usuario #7: Gestión de Cuenta del Usuario**

**Descripción:**  
Como usuario, quiero gestionar mi perfil, actualizando mi nombre, correo y contraseña, o eliminando mi cuenta, para mantener mis datos actualizados y seguros.

**Criterios de Aceptación:**

* + Debe incluir formularios para actualizar nombre, correo y contraseña.
  + Debe haber un botón claro para eliminar la cuenta.
  + Los campos de nombre y correo deben incluir validaciones en el frontend (e.g., formato válido de correo electrónico, si el correo ya existe para otro usuario, etc).
  + El cambio de contraseña debe incluir confirmación de la nueva contraseña y validaciones de longitud mínima.
  + Los cambios exitosos deben mostrar un modal informativo al usuario.
  + La eliminación de cuenta debe requerir confirmación explícita mediante un modal previo.
  + El usuario debe poder cerrar sesión desde el menú principal.
  + Al cerrar sesión, todos los datos temporales relacionados con la sesión deben eliminarse correctamente.

**Valor:** 200  
**Prioridad:** Media  
**Estimación:** 4H

**Historia de Usuario #8: Dashboard de Administración**

**Descripción:**  
Como administrador, quiero un dashboard para visualizar reportes, gráficos y tendencias emocionales, **para** realizar análisis.

**Criterios de Aceptación:**

* Dashboard funcional con acceso a reportes y gráficos.
* Funcionalidades de filtrado y exportación habilitadas.
* El dashboard debe mostrar gráficos de tendencias emocionales
* Los filtros deben ser personalizables (por fecha, por tipo de emoción).
* Los reportes deben ser exportables.

**Valor:** 400  
**Prioridad:** Alta  
**Estimación:** 12H

**Historia de Usuario #9: Entrenamiento Personalizado de Modelos**

**Descripción:**  
Como administrador, quiero tener la opción de cargar nuevos conjuntos de datos emocionales, **para** personalizar el entrenamiento de los modelos y adaptarlos a casos específicos.

**Criterios de Aceptación:**

* Interfaz para cargar nuevos conjuntos de datos.
* Modelos entrenados y ajustados con los datos cargados.
* La interfaz debe permitir cargar diferentes formatos de datos (CSV, JSON).
* El sistema debe ofrecer una retroalimentación clara al usuario sobre el progreso y los resultados del entrenamiento del modelo.
* Los datos cargados deben estar validados antes de ser utilizados para entrenar el modelo (verificación de calidad de datos).
* El modelo entrenado debe ser probado y validado automáticamente antes de ser implementado en el sistema.

**Valor:** 400  
**Prioridad:** Alta  
**Estimación:** 12H

**Historia de Usuario #10: Validación de Resultados**

**Descripción:**  
Como analista, quiero que los resultados emocionales detectados puedan ser validados manualmente, **para** mejorar la precisión del sistema mediante feedback.

**Criterios de Aceptación:**

* Interfaz para revisar y ajustar resultados emocionales.
* Datos validados reflejados en el sistema para futuras iteraciones.
* El sistema debe proporcionar una interfaz donde se puedan ver los resultados emocionales detectados junto con las imágenes (rostro), audios (voz) o textos originales.
* Se debe poder modificar las emociones detectadas manualmente y el sistema debe reflejar estos cambios en tiempo real.
* Cada cambio realizado por el experto debe ser registrado, con fecha, usuario y cambios realizados.
* Después de la validación manual, los resultados ajustados deben ser evaluados en un conjunto de datos para asegurar que el modelo ha mejorado su precisión.
* Los resultados validados deben poder ser exportados en formatos como CSV o PDF para un análisis más profundo.

**Valor:** 350  
**Prioridad:** Media  
**Estimación:** 10H

**Historia de Usuario #11: Visualización de Interacciones**

**Descripción:** Como usuario, quiero visualizar todas las interacciones que he tenido con el chatbot, para tener un registro claro de mis conversaciones previas.

**Criterios de Aceptación:**

* + El sistema debe mostrar una lista o tabla con todas las interacciones del usuario con el chatbot.
  + Cada interacción debe incluir:
    - Fecha y hora.
    - Remitente (Usuario o Chatbot).
    - Contenido del mensaje.
  + Los resultados deben estar ordenados cronológicamente, mostrando las interacciones más recientes primero.
  + Los resultados deben presentarse en una interfaz clara y organizada, permitiendo al usuario navegar cómodamente entre ellos.
  + Las interacciones mostradas deben provenir de la base de datos y actualizarse dinámicamente cuando el usuario acceda a la pantalla.
  + Si el usuario no tiene interacciones previas, se debe mostrar un mensaje claro indicando que no hay interacciones registradas y sugerir que inicie una conversación con el chatbot.

**Valor:** 200 **Prioridad:** Media **Estimación:** 4H

**Historia de Usuario #12: Detalles de las Emociones Detectadas**

**Descripción:**  
Como usuario, quiero ver detalles específicos de cada emoción detectada, **para** entender las razones detrás de las clasificaciones realizadas.

**Criterios de Aceptación:**

* Detalles mostrados incluyen: intensidad emocional, características detectadas (por ejemplo, tono de voz o expresiones faciales).
* Al hacer clic en una emoción específica, el sistema debe mostrar los detalles adicionales, como la intensidad de la emoción y las características relevantes que contribuyeron a la clasificación (tono de voz, patrones faciales, texto).
* Las características relevantes (como la frecuencia o el ritmo en la voz) deben mostrarse gráficamente (por ejemplo, un gráfico de barras o una representación en un gráfico de onda).
* Los detalles de las emociones deben actualizarse en tiempo real conforme el sistema detecta nuevas emociones.
* El sistema debe permitir al usuario comparar los resultados de una emoción detectada con las emociones anteriores de la misma sesión o usuario.
* Debe existir una opción para visualizar la evolución de la emoción a lo largo del tiempo en la misma sesión (por ejemplo, fluctuaciones de la emoción detectada).

**Valor:** 300  
**Prioridad:** Alta  
**Estimación:** 10H

**Historia de Usuario #13: Configuración de Sesiones de Análisis**

**Descripción:**  
Como administrador, quiero configurar sesiones de análisis de emociones para grupos específicos de usuarios, **para** realizar evaluaciones más organizadas.

**Criterios de Aceptación:**

* Sesiones configuradas con parámetros como duración, participantes y objetivos.
* Resultados emocionales registrados por sesión y accesibles desde el dashboard.
* La interfaz debe permitir configurar sesiones fácilmente, seleccionando los parámetros necesarios, como el nombre de la sesión, duración, objetivo de la evaluación, y los participantes.
* Los participantes deben poder acceder a las sesiones configuradas y ser asignados correctamente según los parámetros establecidos.
* Las emociones detectadas deben almacenarse por sesión, y el sistema debe permitir revisar el historial de emociones de sesiones pasadas desde el dashboard.
* Al final de cada sesión, el sistema debe generar un reporte detallado con los resultados emocionales y mostrarlo en el dashboard.
* Las sesiones configuradas deben ser editables, permitiendo modificar los parámetros de la sesión sin perder los resultados anteriores.

**Valor:** 350  
**Prioridad:** Alta  
**Estimación:** 10H

**Historia de Usuario #14: Estadísticas Generales de Uso**

**Descripción:**  
Como administrador, quiero acceder a estadísticas generales de uso del sistema, **para** entender el nivel de interacción de los usuarios y áreas de mejora.

**Criterios de Aceptación:**

* Gráficos y métricas que muestran datos de uso, como cantidad de sesiones analizadas y emociones detectadas.
* Los gráficos deben permitir a los usuarios interactuar con los datos, como hacer clic en un gráfico para obtener más detalles sobre una sesión o emoción detectada.
* Los gráficos deben permitir filtrar los resultados por período (por ejemplo, mensual, semanal), tipo de emoción y otras variables como usuarios o dispositivos.
* El sistema debe generar gráficos que muestren cómo varían las emociones detectadas y el número de sesiones analizadas a lo largo del tiempo.
* Los administradores deben poder acceder a todas las estadísticas a través del dashboard y exportarlas para análisis externos (por ejemplo, en formatos CSV o PDF).
* El sistema debe generar alertas sobre patrones anormales o tendencias inusuales en las interacciones de los usuarios, como picos en emociones negativas.

**Valor:** 300  
**Prioridad:** Media  
**Estimación:** 8H

**Historia de Usuario #15: Análisis Comparativo de Usuarios**

**Descripción:**  
Como administrador, quiero comparar patrones emocionales entre diferentes usuarios, **para** identificar similitudes y diferencias en sus comportamientos.

**Criterios de Aceptación:**

* Comparaciones disponibles en gráficos y tablas.
* Opciones para seleccionar grupos de usuarios específicos.
* Los administradores deben poder seleccionar grupos de usuarios específicos (por ejemplo, todos los usuarios que usan una determinada funcionalidad o grupo de edad) para comparar sus patrones emocionales.
* Los resultados comparativos deben mostrarse tanto en gráficos (barras, líneas, etc.) como en tablas para facilitar la interpretación.
* El sistema debe identificar similitudes y diferencias significativas entre los usuarios en términos de emociones predominantes, frecuencia, y duración de las emociones.
* Los resultados comparativos deben poder ser exportados en un formato adecuado para su análisis posterior, como CSV o PDF.
* El sistema debe permitir filtrar los resultados comparativos por criterios como el tipo de emoción, fecha, o la duración de las sesiones.

**Valor:** 400  
**Prioridad:** Alta  
**Estimación:** 12H

**Historia de Usuario #15: Recuperación de Contraseña por Correo Electrónico**

**Descripción:**  
Como usuario, quiero poder recuperar mi contraseña mediante un correo electrónico de confirmación, para restablecer mi acceso a la plataforma en caso de olvido.

**Criterios de Aceptación:**

* El sistema debe proporcionar un formulario donde el usuario ingrese su correo electrónico registrado.
* Si el correo es válido y está registrado, se enviará un enlace de recuperación con un token único.
* El enlace de recuperación debe ser válido solo por un tiempo determinado (ej. 30 minutos).
* Al hacer clic en el enlace, el usuario podrá ingresar una nueva contraseña y confirmarla.
* La nueva contraseña debe cumplir con los requisitos de seguridad establecidos (ej. mínimo 8 caracteres, al menos una mayúscula, un número y un carácter especial).
* El sistema debe mostrar mensajes de éxito o error en cada paso del proceso.
* Si el usuario intenta usar un enlace vencido, debe recibir un mensaje indicando que debe solicitar uno nuevo.
* La contraseña debe ser encriptada antes de almacenarse en la base de datos.

**Valor:** 250  
**Prioridad:** Media  
**Estimación:** 6H

**Historias Técnicas**

**Historia Técnica #1: Configuración de TensorFlow y Keras**

**Descripción:**  
El sistema requiere TensorFlow y Keras para implementar redes neuronales convolucionales (CNN), **para poder** procesar imágenes y detectar emociones faciales en tiempo real.

**Validación:**

* TensorFlow y Keras configurados correctamente en el entorno.
* Librerías funcionales para entrenamiento y evaluación de modelos de detección emocional.
* TensorFlow y Keras deben ser instalados correctamente en el entorno de desarrollo.
* Comprobar que las versiones de TensorFlow y Keras sean compatibles entre sí y con las bibliotecas de Python utilizadas.
* Ejecutar una prueba de "hello world" (como un modelo simple de regresión o clasificación) para verificar que TensorFlow y Keras están funcionando correctamente en el entorno.
* Configurar el entorno para que pueda ser utilizado para el entrenamiento de modelos de detección facial, sin necesidad de entrenar el modelo en este sprint.
* Incluir una guía de instalación detallada para que otros miembros del equipo puedan configurar fácilmente TensorFlow y Keras en sus entornos locales.

**Valor:** 400  
**Prioridad:** Alta  
**Estimación:** 12H

**Historia Técnica #2: Configuración de Librosa para Análisis de Audio**

**Descripción:**  
El sistema requiere la librería Librosa, **para poder** analizar características de audio y entrenar un modelo de clasificación emocional basado en voz.

**Validación:**

* Librosa instalada y funcional en el entorno.
* Extracción de características de audio probada con muestras iniciales.
* Asegurar que el entorno esté configurado correctamente para procesar archivos de audio, con soporte para formatos comunes como WAV, MP3, y FLAC.

**Valor:** 350  
**Prioridad:** Alta  
**Estimación:** 10H

**Historia Técnica #3: Procesamiento de Lenguaje Natural (PLN)**

**Descripción:**  
El sistema requiere SpaCy o NLTK, **para poder** analizar las emociones en textos generados por el chatbot.

**Validación:**

* Asegurarse de que una de las dos librerías (SpaCy o NLTK) esté instalada correctamente en el entorno.
* Dataset de palabras clave cargado correctamente.
* Pruebas funcionales realizadas con SpaCy o NLTK.
* Configurar el dataset de palabras clave emocionales (por ejemplo, emociones como "felicidad", "tristeza", "enojo", etc.) y asegurarse de que se pueda cargar correctamente en el sistema.
* Realizar pruebas con textos simples para verificar que el sistema pueda identificar y clasificar emociones basadas en el contenido textual.

**Valor:** 350  
**Prioridad:** Alta  
**Estimación:** 10H

**Historia Técnica #4: Diseño de Base de Datos**

**Descripción:**  
El sistema necesita una base de datos PostgreSQL, **para poder** almacenar emociones detectadas, sesiones y registros de interacción.

**Validación:**

* Diseño del esquema relacional de la base de datos.
* Base de datos exportada y funcional en PostgreSQL.
* Verificar que las relaciones entre las tablas estén correctamente implementadas, por ejemplo, claves foráneas entre usuarios y sesiones, o entre sesiones y emociones\_detectadas.
* Configurar la conexión entre el backend y PostgreSQL, asegurando que el sistema pueda almacenar y recuperar datos de manera eficiente.
* Asegurar que la base de datos pueda manejar una cantidad razonable de registros sin experimentar caídas de rendimiento.

**Valor:** 300  
**Prioridad:** Alta  
**Estimación:** 4H

**Historia Técnica #5: Integración de la API del Chatbot con la Base de Datos**

**Descripción**: El sistema requiere integrar la API del chatbot con la base de datos, para poder almacenar información de los usuarios registrados, sus sesiones y mensajes intercambiados con el chatbot.

**Validación:**

* La API debe estar conectada correctamente a la base de datos y permitir registrar nuevos usuarios.
* Las interacciones de los usuarios con el chatbot deben ser registradas correctamente en las tablas correspondientes (usuarios, sesiones, interacciones\_chatbot).
* Las sesiones de usuario deben ser gestionadas de forma eficiente, almacenando los datos ingresados en la api.
* Se deben realizar pruebas de la API para validar que todas las solicitudes principales (por ejemplo, registro de usuario, inicio de sesión, almacenamiento de mensajes) funcionan correctamente.
* Verificar que los datos registrados en la base de datos son consistentes y cumplen con los requisitos definidos (por ejemplo, no se generan duplicados, las sesiones están asociadas al usuario correcto, etc.).
* Proporcionar documentación básica que describa las rutas de la API, los métodos soportados (GET, POST), y los parámetros necesarios.

**Valor:** 300 **Prioridad:** Alta **Estimación:** 8H

**Historia Técnica #6: Implementación de Seguridad con Encriptación**

**Descripción:**  
El sistema necesita implementar encriptación, **para poder** garantizar la privacidad de los datos emocionales capturados.

**Validación:**

* Transferencia de datos segura validada con pruebas.
* Realizar pruebas básicas de seguridad para asegurarse de que no haya vulnerabilidades en la transmisión de datos.
* Verificar que la base de datos esté protegida mediante encriptación (si es necesario) y que los datos sensibles no puedan ser accesibles sin autorización.

**Valor:** 200  
**Prioridad:** Alta  
**Estimación:** 6H

**Historia Técnica #7: Configuración de Matplotlib**

**Descripción:**  
El sistema requiere Matplotlib, **para poder** generar gráficos interactivos que muestren tendencias emocionales.

**Validación:**

* Gráficos generados correctamente con filtros funcionales.
* Validar que se pueden generar gráficos básicos como barras, líneas o dispersión utilizando datos emocionales.
* Verificar que los gráficos de Matplotlib se integren correctamente con el sistema de visualización de emociones y se actualicen en tiempo real.
* Incluir ejemplos básicos de cómo generar gráficos utilizando Matplotlib, y cómo integrar esos gráficos en el frontend.

**Valor:** 200  
**Prioridad:** Media  
**Estimación:** 6H

**Historia Técnica #8: Configuración del Backend en Python  
Descripción:**Configurar el backend en Python para manejar la lógica central del sistema, exponer APIs REST y generar gráficos de interacción.

**Criterios de Aceptación:**

* El backend en Flask recibe solicitudes del chatbot (Node.js) y devuelve respuestas.
* Se implementa una ruta /api/chatbot para procesar las solicitudes del chatbot.
* Se implementa /api/graph para generar gráficos basados en interacciones y devolverlos como imágenes.
* Conexión exitosa con la base de datos PostgreSQL.
* Pruebas unitarias de las rutas con Postman o cURL.

**Valor:** 400 **Prioridad:** Alta **Estimación:** 10H

**Historia Técnica #9:** Configuración Inicial del entorno de React para frontend principal

**Descripción:**Configurar React como framework frontend para establecer el entorno de desarrollo, inicializar el proyecto, definir una estructura organizada y crear componentes reutilizables base que serán usados en todo el sistema.

**Criterios de Aceptación:**

* React debe ser configurado correctamente utilizando herramientas estándar como create-react-app o configuraciones personalizadas.
* Crear carpetas organizadas para componentes, estilos, servicios y utilidades.
* Implementar componentes genéricos como botones, formularios, y contenedores reutilizables.
* Validar que los componentes sean responsivos y estilizados
* Verificar que el proyecto se ejecute correctamente en navegadores modernos como Chrome, Firefox y Edge.
* Asegurar tiempos de carga inicial razonables (menos de 5 segundos bajo condiciones normales).

**Valor:** 300 **Prioridad:** Alta **Estimación:** 6H

**Historia Técnica #10: Implementación de Pantallas de Inicio de Sesión y Registro**

**Descripción:**  
Diseñar e implementar las pantallas de inicio de sesión y registro, asegurando usabilidad y responsividad.

**Validación:**

* Asegurarse de que el framework de diseño (Material-UI o TailwindCSS) esté instalado correctamente a través de npm install o yarn add y que esté correctamente configurado en el proyecto.
* Crear y verificar componentes básicos (por ejemplo, un botón, un campo de texto, etc)
* Campos de formulario validados (email, contraseña, etc.).
* Verificar que los componentes interactivos (como botones, formularios) funcionen correctamente, con respuestas visuales adecuadas
* Integración completa con la API del backend para gestionar usuarios.

**Valor:** 250  
**Prioridad:** Media  
**Estimación:** 8H

**Historia Técnica #11: Conexión del Frontend con el Backend**

**Descripción:**  
El sistema requiere conectar el frontend (React) con el backend python, **para poder** enviar y recibir datos eficientemente.

**Validación:**

* Flujo de inicio de sesión y registro probado y funcional.
* Comunicación con el microservicio establecida y probada para registrar interacciones.
* Pruebas exitosas de intercambio de datos entre frontend y backend.
* Realizar pruebas de integración en las que se envíen y reciban datos entre el frontend y el backend
* Validar que los datos enviados por el frontend se reflejen correctamente en el backend.

**Valor:** 350  
**Prioridad:** Alta  
**Estimación:** 8H

**Historia Técnica #12: Creación del Prototipo de la Interfaz de Usuario**

**Descripción**:

Como equipo de diseño y desarrollo, queremos crear un prototipo de la interfaz de usuario. El prototipo debe reflejar las principales pantallas de la aplicación (dashboard, visualización de emociones, configuración de sesiones, etc.) y ser completamente interactivo para simular las interacciones del usuario. El objetivo es validar los flujos de usuario, la disposición de los elementos y la navegación antes de comenzar el desarrollo real con React.

**Validación**:

* + El prototipo debe incluir las pantallas principales del sistema, como el dashboard, visualización de emociones, y la configuración de sesiones.
  + El prototipo debe ser interactivo, permitiendo simular clics, desplazamientos y cambios de pantalla como si fuera la aplicación real.
  + Los stakeholders y usuarios finales deben poder navegar el prototipo para evaluar la experiencia de usuario (UX) y realizar comentarios.
  + El diseño del prototipo debe estar alineado con los requisitos del sistema, como los gráficos de emociones y las funcionalidades de filtrado.
  + Debe incluir el flujo completo de la interfaz, permitiendo que se puedan realizar pruebas de usabilidad con usuarios reales.
  + El prototipo debe ser entregado en un formato que pueda ser probado y validado
  + Comentarios y ajustes deben ser posibles antes de pasar a la fase de desarrollo de la interfaz.

**Valor**: 300

**Prioridad**: Alta

**Estimación**: 8 horas

**Historia Técnica #14: Configuración del Microservicio Node.js**

**Descripción:**  
Configurar Node.js como microservicio principal del chatbot, gestionando la comunicación entre el frontend en React y el backend en Python.

**Criterios de Aceptación:**

* Node.js recibe y procesa solicitudes del frontend en React.
* Node.js envía solicitudes al backend en Python y recibe respuestas correctamente.
* Implementación de la API /generateText para procesar y devolver respuestas del chatbot.
* Pruebas exitosas de integración con flujo completo React → Node.js → Python.

**Valor:** 350  
**Prioridad:** Alta  
**Estimación:** 8H

**Historia Técnica #15: Personalización del Chatbot**

**Descripción:** Personalizar el chatbot para mostrar mensajes de bienvenida con el nombre del usuario, registrar interacciones en tiempo real y manejar el cierre de sesión.

**Criterios de Aceptación:**

* El chatbot muestra un mensaje de bienvenida que incluye el nombre del usuario autenticado.
* El mensaje debe actualizarse correctamente si otro usuario inicia sesión sin recargar la página.
* Cada mensaje enviado por el usuario se registra con un timestamp en la tabla interacciones\_chatbot de la base de datos.
* El nombre del usuario que envía el mensaje aparece junto a los mensajes en la conversación.
* Al cerrar sesión desde la interfaz del chatbot, el sistema borra el token de autenticación y redirige al usuario a la pantalla de inicio de sesión.
* Todos los datos de la sesión (como mensajes y nombre del usuario) deben reiniciarse al cerrar sesión.

**Valor:** 300

**Prioridad:** Alta

**Estimación:** 8 horas

**Historia Técnica #16: Entrenamiento de Modelos de Reconocimiento Emocional**

**Descripción:** Entrenar modelos de reconocimiento de emociones faciales y vocales, asegurando precisión, y evaluar su desempeño.

**Criterios de Aceptación:**

* Los modelos de reconocimiento facial y de voz son entrenados con datasets específicos que incluyen al menos 6 emociones principales (felicidad, tristeza, enojo, sorpresa, miedo y neutralidad).
* Métricas de desempeño calculadas, incluyendo F1-Score y Accuracy.
* Ajustar hiperparámetros (como tasa de aprendizaje y número de épocas) para mejorar la precisión del modelo.
* Exportar los modelos entrenados en un formato compatible (por ejemplo, .h5 para TensorFlow o .pt para PyTorch) y probar la carga de los modelos en el sistema.
* Validar que los modelos entrenados pueden procesar imágenes y audios en tiempo real.

**Valor:** 400

**Prioridad:** Alta

**Estimación:** 12 horas

**Historia Técnica #17: Implementación de Redirección Basada en Roles**

**Descripción**:  
El sistema necesita implementar la lógica para redirigir automáticamente al menú correspondiente según el rol del usuario (administrador o usuario regular) al iniciar sesión.

**Validación**:

* El sistema debe identificar el rol del usuario al iniciar sesión y almacenarlo en localStorage o token.
* Los usuarios deben ser redirigidos automáticamente a:
  + Administradores: menu\_admin.js.
  + Usuarios: menu\_usuario.js.
* Proteger las rutas para evitar que usuarios no autenticados accedan a menús sin permiso.
* Validar que los usuarios no autenticados sean redirigidos al inicio de sesión.

**Valor**: 300  
**Prioridad**: Alta  
**Estimación**: 8H

**Historia Técnica #18: Desarrollo de Menús Dinámicos para Usuarios y Administradores**

**Descripción**:  
Implementar menús para usuarios y administradores. Cada menú debe ofrecer las opciones correspondientes según el rol del usuario al iniciar sesión. Los usuarios regulares tendrán acceso al chatbot y otras funcionalidades, mientras que los administradores tendrán un menú básico con cierre de sesión y otras funcionalidades que serán agregadas en futuros sprints.

**Validación**:

1. Implementar un componente React que gestione los menús dinámicamente según el rol:
   * **Usuarios**:
     + Botón para redirigir al microservicio del chatbot.
     + Botón para cerrar sesión, eliminando los datos de sesión en localStorage y redirigiendo al inicio de sesión.
   * **Administradores**:
     + Botón para cerrar sesión, eliminando los datos de sesión en localStorage y redirigiendo al inicio de sesión.
     + Mensaje indicando que las funcionalidades administrativas estarán disponibles en futuros sprints.
2. Asegurar que cada menú sea accesible únicamente para usuarios con el rol correspondiente:
   * Usuarios con rol usuario acceden al menú de usuario.
   * Usuarios con rol administrador acceden al menú de administrador.
3. Proteger las rutas para evitar que usuarios no autenticados accedan a los menús.
4. Validar que las opciones de redirección (chatbot y cierre de sesión) funcionen correctamente durante pruebas funcionales.

**Valor**: 300  
**Prioridad**: Alta  
**Estimación**: 4H

**Historia Técnica #19: Desacoplamiento de la Base de Datos**

**Descripción:**  
El sistema debe ser independiente del motor de base de datos, permitiendo la compatibilidad con distintos gestores de bases de datos (ejemplo: MySQL, PostgreSQL, SQL Server, etc.), sin necesidad de realizar cambios significativos en el código.

**Criterios de Aceptación:**

* Utilizar **JPA (Java Persistence API) con Hibernate** o **Spring Data JPA** para la capa de acceso a datos, evitando consultas SQL directas específicas de un motor.
* Configurar **perfiles de conexión** en application.properties o application.yml para que el sistema pueda cambiar de base de datos fácilmente.
* Utilizar un **pool de conexiones (HikariCP, Apache DBCP, etc.)** para manejar eficientemente la comunicación con distintos motores de base de datos.
* Implementar **migraciones con Flyway o Liquibase**, para que la estructura de la base de datos pueda crearse y actualizarse independientemente del motor utilizado.
* Probar el sistema con al menos **dos motores de base de datos diferentes** (ejemplo: MySQL y PostgreSQL) para validar la compatibilidad.
* Evitar consultas SQL nativas específicas de un motor y, en su lugar, utilizar **JPQL (Java Persistence Query Language) o consultas con Criteria API**.
* Documentar el proceso de configuración para cambiar de un motor de base de datos a otro en el entorno de desarrollo y producción.

**Valor:** 350  
**Prioridad:** Alta  
**Estimación:** 10H

**Sprints:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint** | **Historia de Usuario / Técnica** | **Descripción** | **Valor** | **Prioridad** | **Estimación (Horas)** |
| **Sprint 1** | **Historia Técnica #1: Configuración de TensorFlow y Keras** | Configuración de TensorFlow y Keras para procesamiento de emociones faciales en tiempo real. | 400 | Alta | 12H |
|  | **Historia Técnica #2: Configuración de Librosa** | Instalación y configuración de la librería Librosa para análisis de audio en emociones vocales. | 350 | Alta | 10H |
| **Sprint 2** | **Historia Técnica #3: Procesamiento de Lenguaje Natural (PLN)** | Configuración de SpaCy o NLTK para análisis de emociones en texto. | 350 | Alta | 10H |
|  | **Historia Técnica #4: Diseño de Base de Datos** | Creación de una base de datos en PostgreSQL para almacenar emociones y registros. | 300 | Alta | 4H |
| **Sprint 3** | **Historia Técnica #5: Integración de la API del Chatbot con la base de datos** | Integración de la API de chatbot con la base de datos para almacenar interacciones y usuarios del chatbot. | 300 | Alta | 8H |
|  | **Historia Técnica #6: Implementación de Seguridad con Encriptación** | Implementación de encriptación para garantizar la privacidad de los datos. | 200 | Alta | 6H |
| **Sprint 4** | **Historia Técnica #7: Configuración de Matplotlib** | Instalación y configuración de Matplotlib para crear gráficos interactivos de emociones. | 200 | Media | 6H |
|  | **Historia Técnica #8: Configuración del Backend en Python** | Configurar el backend en Python para manejar la lógica central del sistema, exponer APIs REST y generar gráficos de interacción. | 300 | Media | 10H |
|  | **Historia Técnica #12: Creación del Prototipo de la Interfaz de Usuario** | Reflejar las principales pantallas de la aplicación (dashboard, visualización de emociones, configuración de sesiones, etc.) | 300 | Alta | 8H |
|  | **Historia Técnica #14: Configuración del Microservicio Node.js** | Configurar Node.js como microservicio principal del chatbot, gestionando la comunicación entre el frontend en React y el backend en Python. | 350 | Alta | 8H |
| **Sprint 5** | **Historia Técnica #9: Configuración de React para el Frontend** | Configuración Inicial del entorno de React para frontend principal | 300 | Alta | 6H |
|  | **Historia Técnica #10: Implementación de Pantallas de Inicio de Sesión y Registro** | Diseñar e implementar las pantallas de inicio de sesión y registro, asegurando usabilidad y responsividad. | 250 | Media | 8H |
| **Sprint 6** | **Historia Técnica #11: Conexión del Frontend con el Backend** | Conectar el frontend con el backend (React - Node.js) para el intercambio de datos. | 350 | Alta | 8H |
| **Sprint 7** | **Historia Técnica #15: Personalización del Chatbot** | Personalizar el chatbot para mostrar mensajes de bienvenida con el nombre del usuario, registrar interacciones en tiempo real y manejar el cierre de sesión. | 400 | Alta | 8H |
|  | **Historia Técnica #16: Entrenamiento de Modelos de Reconocimiento Emocional** | Entrenar modelos de reconocimiento de emociones faciales y vocales, asegurando precisión mínima del 85%, y evaluar desempeño. | 350 | Alta | 12H |
| **Sprint 8** | **Historia Técnica #17: Implementación de Redirección Basada en Roles** | El sistema necesita implementar la lógica para redirigir automáticamente al menú correspondiente según el rol del usuario (administrador o usuario regular) al iniciar sesión. | 300 | Alta | 8H |
|  | **Historia Técnica #18: Desarrollo de Menús Dinámicos para Usuarios y Administradores** | Implementar menús para usuarios y administradores. Cada menú debe ofrecer las opciones correspondientes según el rol del usuario al iniciar sesión. Los usuarios regulares tendrán acceso al chatbot y otras funcionalidades, mientras que los administradores tendrán un menú básico con cierre de sesión y otras funcionalidades que serán agregadas en futuros sprints. | 300 | Alta | 4H |
| **Sprint 9** | **Historia de Usuario #6: Implementación de Interfaz del Usuario y Chatbot** | Como usuario, quiero una interfaz gráfica que me permita gestionar mi perfil, iniciar sesión, registrarme, navegar por el menú principal y utilizar el chatbot, para interactuar de manera eficiente y mantener mis datos actualizados y seguros. | 300 | Alta | 6H |
|  | **Historia de Usuario #11: Visualización de Interacciones** | Como usuario, quiero visualizar todas las interacciones que he tenido con el chatbot, para tener un registro claro de mis conversaciones previas. | 200 | Media | 4H |
|  | **Historia de Usuario #7: Gestión de Cuenta del Usuario** | Como usuario, quiero gestionar mi perfil, actualizando mi nombre, correo y contraseña, o eliminando mi cuenta, para mantener mis datos actualizados y seguros. | 200 | Media | 4H |
| **Sprint 10** | **Historia de Usuario #1: Detección de Emociones Faciales** | Como usuario, quiero que el sistema detecte mis emociones faciales en tiempo real, para mejorar mi experiencia. | 400 | Alta | 12H |
|  | **Historia de Usuario #8: Dashboard de Administración** | Como administrador, quiero un dashboard para visualizar reportes, gráficos y tendencias emocionales, para realizar análisis efectivos. | 400 | Alta | 10H |
| **Sprint 11** | **Historia de Usuario #3: Análisis Emocional en Texto** | Como usuario, quiero que el sistema analice las emociones en los textos de mis conversaciones con el chatbot, para registrar y responder adecuadamente. | 400 | Alta | 10H |
|  | **Historia de Usuario #4: Respuestas Dinámicas del Chatbot** | Como usuario, quiero que el chatbot ajuste sus respuestas basadas en emociones detectadas, para mejorar la experiencia del usuario. | 250 | Alta | 6H |
|  | **Historia de Usuario #12: Detalles de las Emociones Detectadas** | Como usuario, quiero ver detalles específicos de cada emoción detectada, **para** entender las razones detrás de las clasificaciones realizadas. | 300 | Alta | 10H |
| **Sprint 12** | **Historia de Usuario #5: Visualización de Patrones Emocionales** | Como analista, quiero visualizar patrones de emociones en gráficos, **para** identificar tendencias emocionales. | 300 | Alta | 10H |
|  | **Historia de Usuario #15: Recuperación de Contraseña por Correo Electrónico** | Como usuario, quiero poder recuperar mi contraseña mediante un correo electrónico de confirmación, para restablecer mi acceso a la plataforma en caso de olvido. | 250 | Media | 6H |
|  | **Historia Técnica #19: Desacoplamiento de la Base de Datos** | El sistema debe ser independiente del motor de base de datos, permitiendo la compatibilidad con distintos gestores de bases de datos (ejemplo: MySQL, PostgreSQL, SQL Server, etc.), sin necesidad de realizar cambios significativos en el código. | 350 | Alta | 8H |