Historias de Usuario

**Nombre:** Alejandro Hidalgo

**Código:** 6823

**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).
* El sistema deberá mostrar al usuario la emoción detectada

**Valor:** 400

**Prioridad:** Alta

**Estimación:** 12H

**Historia de Usuario #2: Registro de Actividad del Modelo Descripción:**

Como administrador, quiero acceder al historial de predicciones realizadas por el modelo, para auditar el funcionamiento y evaluar la frecuencia de uso del sistema.

**Criterios de Aceptación:**

* El sistema debe registrar cada predicción con hora, emoción detectada y usuario (si está autenticado).
* Debe existir una vista de registros con filtros por fecha, emoción y sesión.
* El registro debe mantenerse incluso si se reinicia el sistema.

**Valor:** 300

**Prioridad:** Alta

**Estimación:** 6H

**Historia de Usuario #3: 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 #4: Notificaciones Visuales de Emoción Detectada Descripción:**

Como usuario, quiero recibir notificaciones visuales inmediatas en la interfaz cuando se detecte una emoción, para tener claridad sobre cómo reacciona el sistema ante mi estado.

**Criterios de Aceptación:**

* El sistema debe mostrar una tarjeta visual o grafico cada vez que se detecte una emoción.
* La notificación debe desaparecer automáticamente o permitir su cierre manual.
* Debe cambiar según la emoción (color, ícono, texto).
* Las notificaciones no deben afectar el rendimiento de la detección en tiempo real.

**Valor:** 250

**Prioridad:** Media

**Estimación:** 5H

**Historia de Usuario #5: 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 #6: 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: Visualización de Interacciones**

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

**Criterios de Aceptación:**

* + El sistema debe mostrar una lista o tabla con todas las interacciones del usuario.

**Valor:** 200

* Cada interacción debe incluir:
  + Fecha y hora.
  + 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

**Prioridad:** Media

**Estimación:** 4H

**Historia de Usuario #10: 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 y emoción detectada
* Las características relevantes deben mostrarse gráficamente por ejemplo porcentaje de precisión.
* 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 #11: Generación de Reportes Estadísticos de Emociones**

**Descripción:**  
Como administrador, quiero generar reportes estadísticos de los datos emocionales detectados, para analizar patrones, tomar decisiones informadas y mejorar las sesiones de evaluación emocional.

**Criterios de Aceptación:**

* El sistema debe permitir generar reportes estadísticos a partir de los datos emocionales recolectados en todas las sesiones.
* Los reportes deben incluir métricas clave como:  
  • Frecuencia de emociones por usuario y por sesión.  
  • Distribución porcentual de emociones detectadas.  
  • Evolución temporal de emociones por grupo o individuo.  
  • Comparativas entre sesiones, grupos y objetivos.
* Los reportes deben poder filtrarse por fechas, grupos, sesiones y usuarios específicos.
* El dashboard debe visualizar los reportes mediante gráficos comprensibles (barras, líneas, pastel).
* Los reportes deben poder exportarse en formatos como PDF y Excel.
* El sistema debe permitir guardar reportes generados para futuras consultas.
* La privacidad de los datos debe garantizarse, mostrando solo información permitida según el rol del usuario.

**Valor:** 350

**Prioridad:** Alta

**Estimación:** 10H

**Historia de Usuario #12: Gestión de Usuarios Registrados**

**Descripción:**  
Como administrador, quiero gestionar a los usuarios registrados en el sistema, para asignar roles adecuados cuando sea necesario, asegurando así un control adecuado sobre los accesos y permisos.

**Criterios de Aceptación:**

* El sistema debe permitir visualizar una lista completa de usuarios registrados con sus nombres y rol.
* El administrador podrá cambiar el rol de un usuario (por ejemplo, de “usuario” a “administrador”).
* Se debe poder eliminar usuarios del sistema con confirmación previa.
* El sistema debe contar con filtros y buscador para localizar usuarios fácilmente por nombre, correo o rol.
* El sistema debe validar que un administrador no pueda eliminar cuentas de administrador
* Estas funciones deben estar disponibles únicamente para usuarios con rol de administrador y accesibles desde el menú de administrador

**Valor:** 300

**Prioridad:** Media

**Estimación:** 8H

**Historia de Usuario #13: 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 #14: 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

**Historia de Usuario #15: Configuración de Parámetros del Sistema**

**Descripción:**

Como administrador, quiero poder configurar parámetros del sistema como la cantidad de intentos fallidos antes de bloqueo, el tiempo de duración del bloqueo, la duración de la sesión activa y la duración del token de recuperación de contraseña, para adaptar la seguridad del sistema según las necesidades de la organización.

**Criterios de Aceptación:**

• El sistema debe ofrecer una interfaz accesible solo para administradores donde puedan visualizar y editar los parámetros de configuración del sistema.

• El administrador debe poder establecer:

 - La cantidad máxima de intentos fallidos antes de bloquear una cuenta.

 - El tiempo de duración del bloqueo en minutos.

 - El tiempo de duración de la sesión activa de los usuarios.

 - La duración del token enviado para recuperación de contraseña.

• El sistema debe validar que los valores ingresados por el administrador estén dentro de rangos lógicos y seguros.

• Los cambios deben guardarse de forma persistente y reflejarse de inmediato en el comportamiento del sistema.

• El sistema debe mostrar mensajes claros de éxito o error al modificar los parámetros.

• Toda acción de cambio de configuración debe registrarse en un log de auditoría indicando el administrador que la realizó y la fecha/hora.

**Valor:** 350

**Prioridad:** Alta

**Estimación:** 8H

Historias Técnicas

**Nombre:** Alejandro Hidalgo

**Código:** 6823

# 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.
* 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: Evaluación Automatizada del Modelo CNN Descripción:**

Implementar una función que evalúe automáticamente el desempeño del modelo CNN con métricas estándar (accuracy, precision, recall, F1-score), para validar su rendimiento de forma objetiva.

# Validación:

* Ejecución automática de evaluaciones al finalizar cada ciclo de entrenamiento.
* Visualización clara de métricas en consola o archivo log.

**Valor:** 350

**Prioridad:** Media

# Estimación: 4H

**Historia Técnica #3: 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
  + 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 #4: Implementación de Módulo de Visualización de Emociones en tiempo real**

**Descripción:**

Desarrollar el módulo que visualiza los resultados de las emociones detectadas en tiempo real, utilizando gráficos y notificaciones visuales, para facilitar el análisis por parte del

usuario.

# Validación:

* Gráficos generados en tiempo real usando datos del sistema.
* Uso de librerías como Chart.js, D3.js o Matplotlib.
* Componente React para mostrar estos gráficos en el dashboard.

**Valor:** 300

**Prioridad:** Alta

# Estimación: 6H

**Historia Técnica #5: 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 #6: 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 #7: 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 y devuelve respuestas.
  + Se implementa una ruta /api para procesar las solicitudes .
  + Se implementa /api/graph para generar gráficos basados en interacciones y devolverlos como imágenes.
  + Conexión exitosa con la base de datos PostgreSQL.

**Valor:** 400

**Prioridad:** Alta

# Estimación: 10H

**Historia Técnica #8:** 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 #9: 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 #10: 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:

* + 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 #11: 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 #12: Configuración Avanzada del Frontend en React.js Descripción:

Configurar y estructurar el entorno del frontend utilizando React.js, asegurando una

arquitectura escalable, componentes reutilizables y una base sólida para el desarrollo de la interfaz gráfica del sistema web.

# Criterios de Aceptación:

* Inicialización del proyecto con create-react-app o configuración manual con Webpack.
* Estructura organizada por carpetas: components/, pages/, services/, assets/, utils/.
* Integración de TailwindCSS o Material UI para el diseño visual.
* Implementación de un layout base con barra lateral, cabecera y espacio de contenido.
* Comprobación de compatibilidad con navegadores modernos.
* Verificación de tiempos de carga inicial inferiores a 5 segundos bajo condiciones normales.

**Valor:** 350

**Prioridad:** Alta

# Estimación: 4H

**Historia Técnica #13: Entrenamiento de Modelos de Reconocimiento Emocional**

**Descripción:** Entrenar modelos de reconocimiento de emociones faciales, 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 #14: 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 #15: 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 sistema de reconocimiento, 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 reconocimiento emocional.
    - 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.

1. 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.
2. Proteger las rutas para evitar que usuarios no autenticados accedan a los menús.
3. Validar que las opciones de redirección funcionen correctamente durante pruebas funcionales.

**Valor**: 300

**Prioridad**: Alta

**Estimación**: 4H

# Historia Técnica #16: 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

**Historia Técnica #17: Integración del Modelo con Mayor Precisión a la Aplicación**

**Descripción:**

Como equipo de desarrollo, necesitamos evaluar los modelos entrenados para el

reconocimiento de emociones y seleccionar el que tenga mayor precisión, para

integrarlo en la aplicación de análisis emocional y asegurar resultados confiables.

**Criterios de Aceptación:**

* Se debe realizar una evaluación comparativa entre los modelos entrenados (por ejemplo: VGG16, ResNet50, MobileNet, Xception, etc.).
* La comparación debe utilizar métricas como precisión, recall, F1-score y exactitud global en el dataset de validación.
* El modelo con mejor desempeño debe ser seleccionado e integrado en el sistema de análisis emocional.
* Se debe asegurar la compatibilidad del modelo con el entorno actual de despliegue (por ejemplo, Python backend, API REST o integración con la app móvil).
* El sistema debe seguir funcionando sin errores y con los mismos endpoints de detección emocional una vez integrado el nuevo modelo.
* Se deben registrar los cambios en el repositorio con documentación sobre el modelo seleccionado y la justificación técnica de la elección.
* Se debe incluir una validación manual de resultados tras la integración para verificar su funcionamiento en condiciones reales.

**Valor: 250**

**Prioridad: Alta**

**Estimación: 8H**

**Historia Técnica #18: Script en Python para probar el modelo entrenado en tiempo real**

**Descripción:**  
Como equipo de desarrollo, necesitamos crear un script en Python que utilice un modelo entrenado con Keras para detectar emociones en tiempo real desde la cámara del usuario, mostrando en pantalla las emociones identificadas junto con su nivel de precisión.

**Criterios de Aceptación:**

* El script debe utilizar OpenCV para capturar video en tiempo real desde la cámara del dispositivo.
* Debe cargar un modelo entrenado con Keras (formato .h5) previamente validado para clasificación de emociones faciales.
* En cada frame capturado:
  + Detectar rostros.
  + Realizar el preprocesamiento adecuado (como resize, normalización, conversión a escala de grises si aplica).
  + Predecir la emoción usando el modelo cargado.
* El sistema debe mostrar en pantalla (sobre el rostro detectado) la emoción predicha y la probabilidad asociada (por ejemplo: “Feliz: 87%”).
* La predicción debe actualizarse en tiempo real (fluidez mínima de 5 FPS).
* Debe incluir mensajes de error claros si no se detecta rostro o si hay fallos en la cámara o carga del modelo.
* El script debe estar modularizado y documentado para permitir su futura integración a otras interfaces (web o móvil).
* El script debe cerrarse correctamente con una tecla definida (por ejemplo, 'q' para salir).

**Valor: 300**

**Prioridad: Alta**

**Estimación: 8H**

**Historia Técnica #19: Bloqueo Temporal por Múltiples Intentos Fallidos de Inicio de Sesión**

**Descripción:**

Implementar una funcionalidad de seguridad que bloquee temporalmente la cuenta de

usuario tras varios intentos fallidos de inicio de sesión, para prevenir accesos no autorizados por fuerza bruta.

# Criterios de Aceptación:

* El sistema debe permitir un máximo de 5 intentos fallidos consecutivos.
* Si se superan los intentos, la cuenta se bloquea durante 15 minutos.
* El mensaje debe indicar claramente que la cuenta está temporalmente bloqueada.
* Cada intento debe registrarse en la base de datos con fecha, IP y usuario.
* Un administrador debe poder desbloquear manualmente una cuenta si es necesario.
* El contador debe reiniciarse automáticamente tras un inicio de sesión exitoso.

**Valor:** 300

**Prioridad:** Alta

**Estimación:** 6H

**Historia Técnica #20: Preparación y Preprocesamiento de Datos para el Modelo CNN**

**Descripción:**  
Implementar una función que prepare y preprocese automáticamente los datos antes de entrenar el modelo CNN. Este proceso debe incluir la limpieza, normalización, balanceo y particionamiento de los datos en conjunto de entrenamiento y validacion.

**Validación:**

* Ejecución automática de preprocesamiento al iniciar el ciclo de entrenamiento.
* Implementación de técnicas de balanceo de clases como **Data Augmentation** y **SMOTE** (si es necesario) para evitar el desbalance de clases.
* Normalización de las imágenes y transformación adecuada de las etiquetas.
* División de los datos en conjuntos de entrenamiento, validación y prueba.
* Visualización clara del progreso de los datos procesados en consola o archivo log.

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint** | **Historia** | **Horas Estimadas** | **Talla** | **Prioridad** | **Objetivo Principal** |
| Sprint 1 | HT-01: Configuración de TensorFlow y Keras | 12h | L | Alta | Dejar entorno IA funcionando. |
|  | HT-03: Diseño de Base de Datos PostgreSQL | 4h | M | Alta | Dejar base de datos operativa. |
|  | HT-07: Configuración del Backend en Python | 10h | L | Alta | Backend funcional con Flask. |
|  | HT-08: Configuración inicial del entorno React | 6h | M | Alta | Frontend React base listo. |
| Sprint 2 | HT-20: Preparación y Preprocesamiento de Datos para el Modelo CNN  Descripción: | 8h | M | Alta | Preprocesamiento del dataset para realización del entrenamiento. |
|  | HT-11: Creación del Prototipo de la Interfaz de Usuario | 8h | M | Alta | Diseño de pantallas principales. |
|  | HT-06: Configuración de Matplotlib | 6h | M | Media | Configurar generación de gráficos básicos. |
| Sprint 3 | HT-05: Implementación de Seguridad con Encriptación | 6h | M | Alta | Seguridad de datos en sistema. |
|  | HT-14: Redirección Basada en Roles | 8h | M | Alta | Manejo de roles de usuario. |
|  | HT-10: Conexión del Frontend con el Backend | 8h | M | Alta | Establecer flujo de datos Frontend - Backend. |
|  | HT-19: Bloqueo Temporal por Intentos Fallidos | 6h | M | Alta | Seguridad ante intentos fallidos de acceso. |
| Sprint 4 | HT-15: Desarrollo de Menús Dinámicos | 4h | M | Alta | Menús por rol configurados. |
|  | HU-05: Gestión de Cuenta del Usuario | 4h | M | Media | Actualización de perfiles de usuarios. |
|  | HU-14: Recuperación de Contraseña por Correo Electrónico | 6h | M | Media | Módulo de recuperación de contraseña. |
| Sprint 5 | HT-13: Entrenamiento de Modelos de Reconocimiento Emocional | 12h | L | Alta | Entrenar IA para reconocimiento emocional. |
|  | HT-02: Evaluación del Modelo CNN | 4h | M | Alta | Análisis de desempeño técnico. |
| Sprint 6 | HT-18: Script para probar modelos en tiempo real | 8h | L | Alta | Prueba de modelos entrenados con OpenCV |
|  | HU-01: Detección de Emociones Faciales | 12h | L | Alta | Activar detección emocional en tiempo real. |
|  | HT-17: Integración del Modelo con Mayor Precisión a la Aplicación | 8h | L | Alta | Escoger el mejor modelo e integrarlo a la aplicación |
|  | HU-09: Visualización de Interacciones | 4h | M | Media | Mostrar historial de interacciones de usuario. |
|  | HU-10: Detalles de Emociones Detectadas | 10h | L | Alta | Análisis detallado de emociones detectadas. |
| Sprint 7 | HT-04: Modulo para visualización de Emociones en tiempo real | 6h | M | Alta | Mostrar resultados emocionales. |
|  | HU-06: Dashboard de Administración | 12h | L | Alta | Consolidación de toda la información para administradores. |
|  | HU-04: Notificaciones Visuales de Emoción Detectada | 5h | M | Media | Alertas visuales al detectar emociones. |
|  | HU-11: Generación de Reportes Estadísticos de Emociones | 10h | L | Alta | Generación de reportes estadísticos |
|  | HU-03: Visualización de Patrones Emocionales | 10h | L | Media | Gráficos de tendencias emocionales. |
| Sprint 8 | HU-12: Gestión de usuarios registrados | 8h | M | Media | Reportes generales de uso del sistema. |
|  | HU-15Configuración de Parámetros del Sistema | 8h | M | Media | Configuración de parámetros del sistema |
| Sprint 9 | HU-13: Análisis Comparativo de Usuarios | 12h | L | Alta | Comparar patrones entre usuarios. |
|  | HU-02: Registro de Actividad del Modelo | 6h | M | Alta | Registro de predicciones en el sistema. |
| Sprint 10 | HT-16: Desacoplamiento de la Base de Datos | 10h | L | Alta | Adaptar el sistema para varios motores de BD. |