Historias de usuario e Historias Técnicas:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Tipo** | **Descripción** | **Estimación (Horas)** | **Prioridad** | **Criterios de Validación** | **Criterios de Aceptación y Cómo Validarlos** |
| **HT-001** | Técnica | Configurar el entorno de desarrollo para el modelo SVM. | 8 | 1 | - Incluir herramientas necesarias para entrenamiento y validación. - Soporte para GPU si es necesario. | - El entorno está configurado correctamente para entrenar modelos SVM (verificar con un modelo de prueba). - GPU es detectada y utilizada (verificar con comandos de la GPU como nvidia-smi o código de TensorFlow). - Herramientas como Scikit-learn correctamente |
| **HU-001** | Usuario | Entrenar y validar el modelo SVM con datos nutricionales. | 16 | 1 | - Precisión mínima del 85% (RNF-003). - Tiempo de respuesta menor a 2 segundos (RNF-001). | - El modelo alcanza una precisión del 85% o más en los datos de prueba (verificar con métricas como precisión, recall y F1-score). - El tiempo de respuesta del modelo no excede los 2 segundos (medir tiempo de predicción usando el módulo time en Python). |
| **HT-002** | Técnica | Diseñar y normalizar la base de datos nutricional. | 12 | 1 | - Consultas eficientes. - Soporte para escalabilidad horizontal (RNF-014). | - Las consultas a la base de datos deben devolver resultados rápidos con grandes volúmenes de datos (medir tiempos de consulta). - Esta realizada con normalización 2fn |
| **HT-003** | Técnica | Diseñar la interfaz web | 10 | 2 | - La interfaz es intuitiva y accesible para todos los usuarios objetivo. | - Evaluar con pruebas de usuario y herramientas de accesibilidad (como WAVE). |
| **HT-004** | Técnica | Implementar y probar la API que conecta el modelo SVM con la interfaz web. | 16 | 2 | - Tiempo de respuesta menor a 2 segundos (RNF-001). | - La API debe devolver respuestas en menos de 2 segundos (medir tiempo de respuesta usando herramientas como Postman). |
| **HT-005** | Técnica | Realizar programacion modular con el login, usando el patron Factory Method | 6 | 3 | Verificar que la lógica del login sea modular y reutilizable, implementando el patrón Factory Method correctamente. | - Validar que el login sea modular y que el código permita agregar nuevas variantes sin alterar el núcleo.  - Realizar pruebas funcionales que confirmen el correcto funcionamiento del login para diferentes casos de uso. |
| **HT-006** | Técnica | Realizar validaciones en registrar sesión en la parte del front end | 6 | 3 | - Asegurar que los campos del formulario de registro de sesión implementen validaciones de entrada (longitud, formato, campos requeridos). | - Confirmar que el usuario no pueda registrar sesiones con datos inválidos o incompletos.  - Realizar pruebas funcionales que garanticen una experiencia de usuario sin errores, incluso en escenarios adversos (campos vacíos, valores incorrectos). |
| **HU- 002** | Usuario | Recuperar la contraseña del usuario | 6 | 4 | - El sistema debe generar un enlace único y temporal para la recuperación de contraseña. | - Intenta usar contraseñas que no cumplan los requisitos y verifica que el sistema rechace el cambio.  - Usa una contraseña válida y confirma que el cambio se realiza correctamente.  - Asegúrate de que el enlace expire tras un tiempo predefinido (por ejemplo, 24 horas). |
| **HT- 007** | Técnica | Utilizar un api para traducir el idioma de ingles a español, el api a utilizar es de la página rapidapi.com | 6 | 4 | - Valida el idioma configurado antes de enviar la solicitud al API para evitar llamadas innecesarias. | - Uso de console logs para ver la respuesta del api gratuita utilizada. |
| **HT-008** | Técnica | Crear un sistema de autenticación basado en roles. | 10 | 5 | - Implementar roles (estudiante, administrador). - Contraseñas deben estar encriptadas | - Verificar que los roles estén correctamente implementados y que los usuarios solo accedan a lo que les corresponde según su rol (probar con diferentes cuentas). - Asegurar que las contraseñas se gestionen de acuerdo con las políticas de seguridad (probar con contraseñas cifradas y complejas). |
| **HT-009** | Técnica | Crear un sistema CRUD, para el administrador | 5 | 5 | - Implementar las operaciones de Crear, Leer, Actualizar y Eliminar. | - Probar que el administrador puede realizar todas las operaciones CRUD sin restricciones y que los cambios se reflejan correctamente en la base de datos. |
| **HT-010** | Técnica | Crear un botón para que el administrador pueda establecer la hora del token de recuperación de contraseña. | 5 | 5 | - Implementar un botón en la interfaz de administración. | - Verificar que, al hacer clic en el botón, se muestre un formulario para establecer la hora y que esta se guarde correctamente en la base de datos. |
| **HT-011** | Técnica | Mejorar la seguridad de la contraseña al registrarse | 6 | 6 | - Implementar requisitos de complejidad para las contraseñas (mínimo de caracteres, uso de mayúsculas, números, caracteres especiales). | - Probar que las contraseñas que no cumplen con los requisitos sean rechazadas y que las que sí cumplen sean aceptadas y almacenadas de forma segura. |
| **HU-004** | Usuario | Probar las recomendaciones generadas por el modelo SVM. | 8 | 6 | - Módulo de pruebas para validar recomendaciones. | - Probar el módulo de validación para asegurarse de que las recomendaciones generadas son correctas y útiles (realizar pruebas con distintos perfiles de usuario). |
| **HT-012** | Técnica | Configurar sistema de backups automáticos para evitar pérdida de datos. | 8 | 11 | - Backups diarios configurados (RNF-005). - Validación de integridad en datos recuperados. | - Verificar que los backups se realicen automáticamente cada 24 horas (probar con un sistema de simulación de fallos). - Validar que los datos recuperados de los backups sean íntegros y no tengan pérdidas (probar la restauración de datos). |
| **HU-005** | Usuario | Acceder a una interfaz adaptativa para recibir recomendaciones alimenticias. | 10 | 7 | - Interfaz responsive (RNF-007). - Compatible con principales navegadores (RNF-017). | - Verificar que la interfaz se adapte correctamente a diferentes tamaños de pantalla (probar en dispositivos móviles y de escritorio). - Probar la compatibilidad con navegadores populares como Chrome, Firefox y Safari. |
| **HT-16** | Técnica | Crear diagrama de actividades de los procesos realizados hasta el sprint 7 | 8 | 8 | - El diagrama incluye todos los procesos relevantes. | - El diagrama está completo y revisado por los interesados. Validar con retroalimentación de los interesados y compararlo con los entregables del sprint 7. |
| **HT-17** | Técnica | Implementar token para los tipos de usuario que son admin y usuario | 4 | 8 | - El sistema permite autenticación con JWT para ambos roles. | - Probar el inicio de sesión para admin y usuario, verificar que se generan tokens válidos y que las rutas protegidas requieren un token válido, esto se realiza por postman |
| **HT-18** | Técnica | Cambiar las alertas del Login y de restaurar contraseña por notificaciones en la misma pantalla | 4 | 8 | - Las notificaciones se muestran en el área designada de la pantalla de login. | - Realizar pruebas de inicio de sesión con credenciales inválidas y confirmar que las notificaciones aparecen correctamente y son comprensibles para el usuario. |
| **HT-013** | Técnica | Que los usuarios tengan la opción de cambiar su nombre, o contraseña cuando estén logeados | 4 | 9 | - La funcionalidad permite cambiar nombre y contraseña desde un perfil o menú de configuración. | - Probar el cambio de nombre y contraseña confirmando que se actualizan correctamente en la base de datos. Validar errores en inputs inválidos (ej. contraseñas débiles). |
| **HT-19** | Técnica | Realizar programacion modular al componente crud | 10 | 9 | - El componente CRUD está modularizado en funciones reutilizables y estructuradas. | - Revisar que cada operación del CRUD (Crear, Leer, Actualizar, Eliminar) esté encapsulada en módulos independientes. Verificar su funcionalidad con pruebas unitarias. |
| **HU-006** | Usuario | Mantener el chat a pesar de cambiar las pestañas por el menú, además de borrar todo el chat cuando haga click en cerrar sesion | 6 | 10 | - El chat persiste al navegar entre diferentes pestañas del menú.  - El chat se borra al hacer clic en "Cerrar sesión". | - Persistencia del Chat: Navegar por diferentes pestañas del menú; el contenido del chat debe mantenerse sin perder los mensajes.  - Eliminación del Chat: Al hacer clic en "Cerrar sesión", verificar que el chat se vacíe y no conserve mensajes anteriores. |
| **HT-014** | Técnica | Añadir en el menú un botón para guardar como favoritos las recetas que le guste al usuario | 10 | 11 | - El botón "Guardar como favorito" debe estar visible en el menú.  - Las recetas seleccionadas como favoritas se guardan correctamente en la base de datos o almacenamiento correspondiente. | - Visibilidad del Botón: El botón "Guardar como favorito" aparece en el menú y es accesible desde cualquier vista de recetas.  - Funcionalidad: Al hacer clic en el botón, la receta actual se guarda como favorita.  - Persistencia: Verificar que la lista de recetas favoritas se mantenga al actualizar la página. |
| **HU-007** | Usuario | Ver un dashboard de progreso nutricional para monitorear hábitos alimenticios. | 6 | 10 | - Gráficos claros e intuitivos. - Compatible con dispositivos móviles. - Fácil aprendizaje (RNF-008). | - Verificar que los gráficos sean claros y fáciles de interpretar (probar con usuarios no técnicos). - Asegurar que el dashboard sea completamente funcional en dispositivos móviles (probar en diferentes tamaños de pantalla). - Validar que el sistema sea fácil de usar para nuevos usuarios (realizar pruebas de usabilidad). |
| **HT-015** | Técnica | Integrar exportación de datos en formato pdf | 6 | 9 | - Exportación accesible desde interfaz. - Cumplir con estándares (RNF-020). | - Verificar que los datos puedan exportarse correctamente en pdf - Asegurar que la exportación cumpla con los estándares de formato establecidos. |
| **HT - 016** | Técnica | Actualizar los requisitos de todo el proyecto | 6 | 11 | - Los requisitos deben estar documentados y actualizados en un repositorio accesible.  - Incluir versiones actualizadas de requisitos funcionales, no funcionales y técnicos. | - Documentación Completa: Verificar que los requisitos actualizados incluyan aspectos funcionales, no funcionales y técnicos.  - Consistencia: Revisar que no existan contradicciones entre requisitos.  - Accesibilidad: Validar que la documentación esté disponible en el repositorio del proyecto o una herramienta de gestión de documentación. |