**HISTORIAS TÉCNICAS**

|  |  |
| --- | --- |
| **Id HT001 Prioridad: Alta** | |
| **Título:** | Configuración del Entorno de Desarrollo |
| **Descripción:** | Como desarrollador, necesito configurar un entorno de desarrollo con Node.js, SQLite, para comenzar a implementar las funcionalidades del sistema. Esto incluye instalar dependencias, configurar un servidor Express, y crear un repositorio Git para control de versiones. |
| **Criterios de Aceptación:** | - Node.js y npm están instalados y funcionales.  - SQLite está instalado y se puede crear una base de datos de prueba.  - Un servidor Express básico responde con un mensaje "Hola Mundo" en el puerto 3000.  - Un proyecto React básico se ejecuta en el navegador.  - El proyecto está inicializado en un repositorio Git. |
| **Pruebas de Aceptación:** | - Ejecutar node --version y npm --version para verificar la instalación.  - Crear una base de datos SQLite y ejecutar un comando para crear una tabla de prueba.  - Iniciar el servidor Express y acceder a http://localhost:3000 para ver el mensaje "Hola Mundo".  - Iniciar el proyecto React y verificar que se muestra una página básica en el navegador.  - Verificar que el repositorio Git tiene un commit inicial. |
| **Estimación:** | 10.5 horas |
| **Dependencias:** | Ninguna |

|  |  |
| --- | --- |
| **Id HT002 Prioridad: Alta** | |
| **Título:** | Diseño e Implementación de la Base de Datos |
| **Descripción:** | Como desarrollador, necesito diseñar y crear la base de datos SQLite con las tablas necesarias para almacenar los datos del sistema y soportar las funcionalidades de las historias de usuario HU001-HU013. |
| **Criterios de Aceptación:** | - Las tablas están creadas con los campos definidos (por ejemplo, usuario: id, nombre\_usuario, contraseña, rol).  - Se insertan datos de prueba en cada tabla.  - Se pueden realizar consultas SELECT para verificar los datos. |
| **Pruebas de Aceptación:** | - Verificar que las tablas (usuario, producto, etc.) existen en la base de datos SQLite.  - Insertar un usuario de prueba y ejecutar un SELECT para confirmar que los datos están almacenados.  - Insertar datos de prueba en las demás tablas y confirmar que las consultas devuelven los datos esperados. |
| **Estimación:** | 10.5 horas |
| **Dependencias:** | Depende de HT001 |

|  |  |
| --- | --- |
| **Id HT003 Prioridad: Alta** | |
| **Título:** | Implementación de la API de Autenticación |
| **Descripción:** | Como desarrollador, necesito implementar una API REST en Express para manejar la autenticación de usuarios, para soportar la historia de usuario HU001 (Inicio de Sesión). La API debe validar las credenciales contra la tabla usuario y devolver un mensaje de éxito o error. |
| **Criterios de Aceptación:** | - Existe un endpoint POST /api/login que recibe nombre de usuario y contraseña.  - Si las credenciales son correctas, devuelve un código 200 y un mensaje de éxito.  - Si las credenciales son incorrectas, devuelve un código 401 y un mensaje de error. |
| **Pruebas de Aceptación:** | - Enviar una solicitud POST a /api/login con credenciales correctas y verificar que se recibe un código 200 y un mensaje de éxito.  - Enviar una solicitud con credenciales incorrectas y verificar que se recibe un código 401 y el mensaje "Usuario o contraseña incorrectos".  - Enviar una solicitud sin datos y verificar que se recibe un código 400 y un mensaje de error. |
| **Estimación:** | 3 horas |
| **Dependencias:** | Depende de HT002 |

|  |  |
| --- | --- |
| **Id HT004 Prioridad: Alta** | |
| **Título:** | Implementación de la API de Gestión de Usuarios |
| **Descripción:** | Como desarrollador, necesito implementar APIs REST en Flask para gestionar usuarios, soportando HU002 (Gestión de Usuarios). Incluye endpoints para listar, actualizar y eliminar (lógico) usuarios, accesibles solo por admin. |
| **Criterios de Aceptación:** | * Endpoint GET /usuarios lista usuarios. * Endpoint PUT /usuarios/ actualiza username, rol, estado. * Endpoint DELETE /usuarios/ marca estado como inactivo. * Solo admin accede (JWT con rol=admin). * Códigos 200, 400, 401, 403, 404. |
| **Pruebas de Aceptación:** | * Probar GET /usuarios como admin (200) y no admin (403). * Probar PUT con datos válidos (200) e inexistentes (404). * Probar DELETE con usuario existente (200, estado=inactivo) e inexistente (404). |
| **Estimación:** | 10.5 horas |
| **Dependencias:** | Depende de HT002 |

|  |  |
| --- | --- |
| **Id HT004 Prioridad: Alta** | |
| **Título:** | Implementación de la API de Gestión de Inventario |
| **Descripción:** | Como desarrollador, necesito implementar APIs REST en Express para manejar entradas y salidas de inventario, para soportar las historias de usuario HU002 (Registro de Entrada de Inventario) y HU005 (Registro de Salida de Inventario). Esto incluye endpoints para registrar entradas y salidas, y actualizar las existencias en la tabla producto. |
| **Criterios de Aceptación:** | - Existe un endpoint POST /api/inventory/entry para registrar entradas.  - Existe un endpoint POST /api/inventory/exit para registrar salidas.  - Las existencias en la tabla producto se actualizan correctamente.  - Si no hay existencias suficientes para una salida, se devuelve un error. |
| **Pruebas de Aceptación:** | - Enviar una solicitud POST a /api/inventory/entry con datos válidos y verificar que se registra en facturac y las existencias aumentan.  - Enviar una solicitud POST a /api/inventory/exit con cantidad válida y verificar que se registra en egresobodega y las existencias disminuyen.  - Enviar una solicitud de salida con cantidad mayor a las existencias y verificar que se recibe un código 400 y el mensaje "Existencias insuficientes". |
| **Estimación:** | 10.5 horas |
| **Dependencias:** | Depende de HT002 |

|  |  |
| --- | --- |
| **Id HT005 Prioridad: Media** | |
| **Título:** | Implementación de la API de Reportes Operativos |
| **Descripción:** | Como desarrollador, necesito implementar una API REST en Express para generar reportes operativos, para soportar las historias de usuario HU004 (Generación de Reporte Operativo) y HU013 (Exportación de Reporte Operativo). El endpoint debe devolver datos filtrados para mostrar en tabla y gráfico, y permitir exportación en CSV/PDF. |
| **Criterios de Aceptación:** | - Existe un endpoint GET /api/reports que recibe filtros (fecha, producto, colaborador) y devuelve datos.  - Los datos devueltos son compatibles con tablas y gráficos.  - Existe un endpoint GET /api/reports/export que genera un archivo CSV o PDF. |
| **Pruebas de Aceptación:** | - Enviar una solicitud GET a /api/reports con filtros válidos y verificar que devuelve los datos esperados.  - Enviar una solicitud con filtros sin datos y verificar que devuelve un mensaje "No hay datos".  - Enviar una solicitud a /api/reports/export con formato CSV y verificar que se genera un archivo descargable. |
| **Estimación:** | 10.5 horas |
| **Dependencias:** | Depende de HT002 |

|  |  |
| --- | --- |
| **Id HT006 Prioridad: Alta** | |
| **Título:** | Desarrollo de la Interfaz de Inicio de Sesión |
| **Descripción:** | Como desarrollador, necesito crear una interfaz en React para el inicio de sesión, para soportar la historia de usuario HU001 (Inicio de Sesión). La interfaz debe incluir un formulario con campos para nombre de usuario y contraseña, y mostrar mensajes de éxito o error tras la autenticación. |
| **Criterios de Aceptación:** | - La interfaz tiene un formulario con campos para nombre de usuario y contraseña.  - Al enviar credenciales correctas, redirige al dashboard.  - Al enviar credenciales incorrectas, muestra un mensaje de error. |
| **Pruebas de Aceptación:** | - Cargar la página y verificar que el formulario tiene los campos esperados.  - Ingresar credenciales correctas y verificar que redirige al dashboard.  - Ingresar credenciales incorrectas y verificar que se muestra el mensaje "Usuario o contraseña incorrectos". |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT003 |

|  |  |
| --- | --- |
| **Id HT007 Prioridad: Alta** | |
| **Título:** | Desarrollo de la Interfaz de Gestión de Usuarios |
| **Descripción:** | Como desarrollador, necesito crear una interfaz en HTML/CSS/JS/jQuery para gestionar usuarios, soportando HU002 (Gestión de Usuarios). Incluye tabla y botones para editar/eliminar, accesible solo por admin. |
| **Criterios de Aceptación:** | * Tabla muestra usuarios (id, username, rol, estado). * Botones para editar/eliminar conectan con API /usuarios. * Solo admin accede (verificar rol en localStorage). * Diseño responsivo. |
| **Pruebas de Aceptación:** | * Cargar interfaz como admin y verificar tabla. * Probar edición/eliminación y verificar cambios en SSMS. * Intentar acceso como no admin (redirige a login) |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT004 |

|  |  |
| --- | --- |
| **Id HT008 Prioridad: Media** | |
| **Título:** | Configuración de Notificaciones en la Interfaz |
| **Descripción:** | Como desarrollador, necesito implementar un sistema de notificaciones en la interfaz React, para soportar las historias de usuario HU007 (Notificación de Stock Bajo) y HU010 (Notificación de Orden Próxima a Vencer). Las notificaciones deben mostrarse en el dashboard cuando el stock está bajo o una orden está próxima a vencer. |
| **Criterios de Aceptación:** | - Las notificaciones se muestran en el dashboard.  - Las notificaciones de stock bajo incluyen el producto y cantidad.  - Las notificaciones de órdenes próximas a vencer incluyen el ID y fecha. |
| **Pruebas de Aceptación:** | - Registrar una salida que deje el stock bajo y verificar que aparece una notificación en el dashboard.  - Crear una orden con fecha de vencimiento cercana y verificar que aparece una notificación con el ID y fecha.  - Verificar que sin condiciones de notificación no se muestra nada. |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT004 y HT007 |

|  |  |
| --- | --- |
| **Id HT009 Prioridad: Media** | |
| **Título:** | Implementación de Seguridad Básica |
| **Descripción:** | Como desarrollador, necesito implementar medidas de seguridad básicas, como encriptación de contraseñas y validación de roles, para proteger el sistema y soportar las historias de usuario que requieren autenticación (HU001-HU013). Esto incluye usar bcrypt para encriptar contraseñas y verificar roles al acceder a endpoints. |
| **Criterios de Aceptación:** | - Las contraseñas en la tabla usuario están encriptadas con bcrypt.  - Los endpoints verifican el rol del usuario antes de permitir acceso.  - Un usuario sin el rol adecuado recibe un error 403. |
| **Pruebas de Aceptación:** | - Insertar un usuario y verificar que su contraseña está encriptada en la base de datos.  - Intentar acceder a un endpoint de administrador con un usuario de rol Encargado y verificar que se recibe un código 403.  - Acceder a un endpoint permitido para el rol y verificar que se recibe un código 200. |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT003 |

|  |  |
| --- | --- |
| **Id HT010 Prioridad: Media** | |
| **Título:** | Configuración de Pruebas Unitarias |
| **Descripción:** | Como desarrollador, necesito configurar y escribir pruebas unitarias para los endpoints principales del backend, para garantizar la calidad del código y facilitar la depuración. Esto incluye pruebas para los endpoints de autenticación y gestión de inventario. |
| **Criterios de Aceptación:** | - Se configuran pruebas unitarias con Jest.  - Existen pruebas para los endpoints /api/login, /api/inventory/entry, y /api/inventory/exit.  - Todas las pruebas pasan correctamente. |
| **Pruebas de Aceptación:** | - Ejecutar las pruebas unitarias y verificar que todas pasan.  - Modificar un endpoint para que falle (por ejemplo, devolver un código incorrecto) y verificar que la prueba detecta el error.  - Verificar que las pruebas cubren los flujos principal y alterno de los endpoints. |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT003 y HT004 |

|  |  |
| --- | --- |
| **Id HT011 Prioridad: Media** | |
| **Título:** | Documentación Técnica del Sistema |
| **Descripción:** | Como desarrollador, necesito documentar la arquitectura, los endpoints de la API, y las instrucciones de instalación, para cumplir con los requisitos de mi tesis y facilitar el uso del sistema por parte de evaluadores. |
| **Criterios de Aceptación:** | - Existe un documento con la arquitectura del sistema (frontend, backend, base de datos).  - Los endpoints de la API están documentados con su método, URL, parámetros y respuestas.  - Se incluyen instrucciones de instalación y ejecución. |
| **Pruebas de Aceptación:** | - Verificar que el documento incluye un diagrama de arquitectura.  - Verificar que los endpoints /api/login, /api/inventory/entry, y /api/reports están documentados.  - Seguir las instrucciones de instalación en un entorno limpio y confirmar que el sistema se ejecuta correctamente. |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT001-HT009 (todas las tareas técnicas previas). |

|  |  |
| --- | --- |
| **Id HT012 Prioridad: Alta** | |
| **Título:** | Implementación de la API de Registro y restablecimiento de contraseña |
| **Descripción:** | Como desarrollador, necesito implementar APIs REST en Flask para registro de usuarios y restablecimiento de contraseña por correo, soportando HU014 (Registro de Nuevo Usuario) y HU015 (restablecimiento de Contraseña). |
| **Criterios de Aceptación:** | * Endpoint POST /register crea usuario (username, password, rol). * Endpoint POST /reset-password envía correo con enlace de restablecimiento. * Endpoint POST /reset-password/ actualiza contraseña. * Códigos 200, 400, 404. |
| **Pruebas de Aceptación:** | * Probar POST /register con datos válidos (200) e inválidos (400). * Probar POST /reset-password con email válido (200) e inválido (404). * Probar POST /reset-password/ y verificar cambio en SSMS. * Verificar recepción de correo. |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT002-HT003. |

|  |  |
| --- | --- |
| **Id HT013 Prioridad: Alta** | |
| **Título:** | Desarrollo de la Interfaz de Registro y restablecimiento de Contraseña |
| **Descripción:** | Como desarrollador, necesito crear interfaces en HTML/CSS/JS/jQuery para registro y reestablecimiento de contraseña, soportando HU014 y HU015. |
| **Criterios de Aceptación:** | * Formularios para registro (username, password, rol) y reestablecimiento (email, nueva contraseña). * Conexión con API /register y /reset-password. * Diseño responsivo, mensajes de éxito/error claros |
| **Pruebas de Aceptación:** | * Cargar formularios y verificar campos. * Probar registro con datos válidos (redirige a login) e inválidos (error). * Probar reestablecimiento con email válido (correo enviado) e inválido (error). * Verificar diseño en móvil y escritorio. |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT012. |

|  |  |
| --- | --- |
| **Id HT014 Prioridad: Alta** | |
| **Título:** | Implementación de la API de Gestión de Inventario |
| **Descripción:** | Como desarrollador, necesito implementar APIs REST en Flask para gestionar entradas/salidas de inventario, soportando HU003 (Registro de Entrada) y HU004 (Registro de Salida). |
| **Criterios de Aceptación:** | * Endpoint POST /inventory/entry registra entrada. * Endpoint POST /inventory/exit registra salida. * Existencias en tabla Materiales actualizadas. * Error 400 si existencias insuficientes. |
| **Pruebas de Aceptación:** | * Probar POST /inventory/entry y verificar aumento de existencias en SSMS. * Probar POST /inventory/exit con cantidad válida (200) e insuficiente (400). |
| **Estimación:** | 10.5 horas |
| **Dependencias:** | Depende de HT002. |

|  |  |
| --- | --- |
| **Id HT015 Prioridad: Alta** | |
| **Título:** | Desarrollo de la Interfaz de Gestión de Inventario |
| **Descripción:** | omo desarrollador, necesito crear una interfaz en HTML/CSS/JS/jQuery para registrar entradas/salidas y monitorear existencias, soportando HU003, HU004, y HU005. |
| **Criterios de Aceptación:** | * Interfaz con formularios para entradas/salidas y tabla de existencias. * Conexión con API /inventory. * Stock bajo resaltado en rojo. * Diseño responsivo. |
| **Pruebas de Aceptación:** | * Cargar interfaz y verificar formularios/tabla. * Probar entrada/salida y confirmar actualización en tabla. * Registrar salida con stock bajo y verificar resaltado. * Verificar diseño en móvil y escritorio. |
| **Estimación:** | 7 horas |
| **Dependencias:** | Depende de HT014. |