



Universidad Tecnológica de Aguascalientes

Ingeniería en Desarrollo y Gestión de Software

Desarrollo Web Integral

Proyecto Parcial 2

Alumno:

Rodriguez Fragoso Omar

GRADO Y GRUPO:

IDGS 9-A-11

CUATRIMESTRE MAYO - AGOSTO 2025

4 de Julio del 2025

1. Introducción

Nombre del Proyecto: Inventrack

Descripción General: Sistema web de gestión de inventario dirigido a usuarios individuales y pequeños negocios.

Tecnologías empleadas:

- **Frontend:** Angular
- **Backend:** Flask (Python)
- **Base de datos:** MongoDB (Atlas)
- **Control de versiones:** Git + GitHub
- **Plataforma de despliegue:** Render

2. Descripción del proyecto

El presente proyecto tiene como finalidad desarrollar un sistema de software para la gestión de inventarios personales o de pequeños negocios. Muchas personas requieren un medio para registrar, monitorear y controlar los artículos que poseen, ya sea en su hogar, taller o comercio, sin necesidad de utilizar herramientas complejas o costosas. Este sistema permitirá a los usuarios administrar productos, registrar entradas y salidas de stock, y visualizar el estado general del inventario de forma fácil e intuitiva.

3. Objetivo

Desarrollar un sistema web que permita a los usuarios registrar y gestionar artículos de inventario, llevar el control de cantidades disponibles, categorías y actualizaciones de stock, usando una interfaz sencilla basada en Angular, un backend en Flask, y una base de datos MongoDB.

4. Alcance

Incluye:

- Registro, edición y eliminación de productos.
- Consulta del listado de inventario.
- Clasificación por categorías.
- Historial básico de cambios de stock.

No incluye:

- Múltiples roles de usuario.
- Módulos de compra/venta o facturación.
- Integración con dispositivos externos (por ejemplo, escáneres).

5. Requerimientos funcionales

1. **RF01:** El sistema deberá permitir registrar un nuevo producto con nombre, descripción, cantidad y categoría.
2. **RF02:** El sistema deberá permitir consultar la lista completa de productos registrados.
3. **RF03:** El sistema deberá permitir editar o eliminar productos existentes.
4. **RF04:** El sistema deberá permitir ajustar el stock (entrada/salida) de un producto.
5. **RF05:** El sistema deberá mostrar alertas cuando un producto tenga un stock por debajo de un mínimo definido por el usuario.

6. Requerimientos no funcionales

- **RNF01:** El sistema debe ser accesible desde navegadores modernos (Chrome, Firefox, Edge).
- **RNF02:** El sistema debe responder en menos de 2 segundos por acción.
- **RNF03:** El sistema debe permitir un manejo fluido hasta con 500 productos registrados.
- **RNF04:** El sistema debe tener una interfaz intuitiva y responsive.
- **RNF05:** La información debe persistir en una base de datos MongoDB segura.

7. Justificación de la arquitectura seleccionada

Se ha seleccionado una arquitectura cliente-servidor, donde el frontend se desarrolla con Angular y se comunica mediante peticiones HTTP con una API REST desarrollada en Flask. Esta separación facilita el mantenimiento, la escalabilidad y permite una distribución clara de responsabilidades.

Además, el backend sigue el patrón de diseño Modelo-Vista-Controlador (MVC):

- **Modelo:** gestionará el acceso y persistencia de los datos en MongoDB.
- **Vista:** corresponde a las respuestas JSON que el servidor envía al cliente.
- **Controlador:** gestiona la lógica de negocio y la conexión entre los modelos y las vistas.

Esta combinación permite que el sistema sea modular, fácil de probar y de extender en el futuro.

8. Modelado

Diagrama de Casos de Uso:

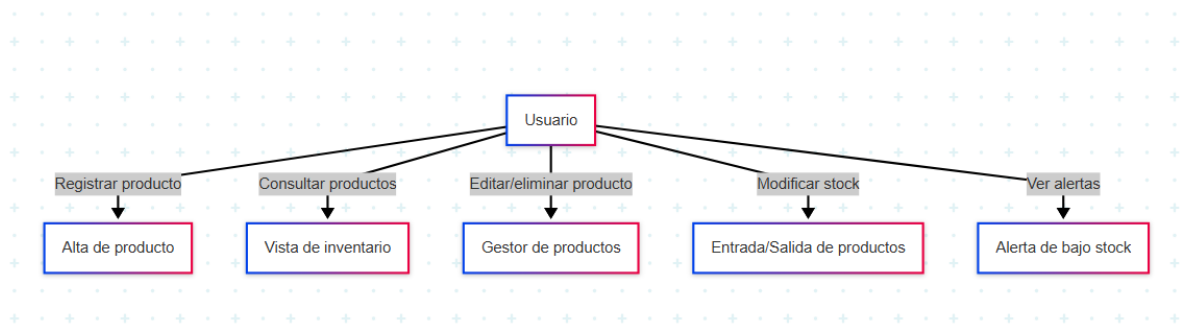


Diagrama de Clases:

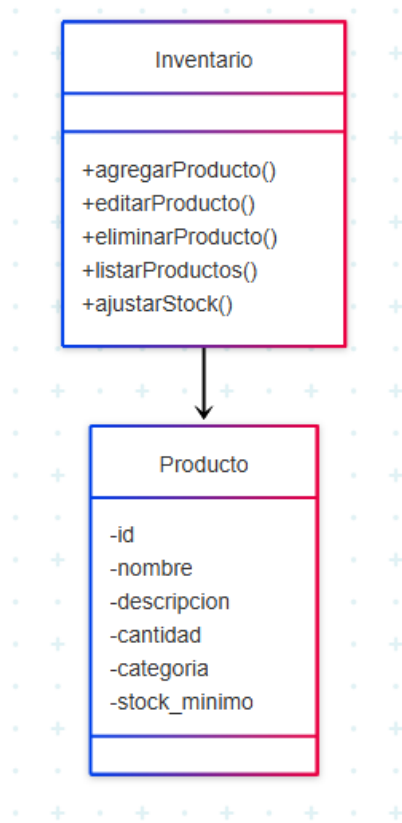
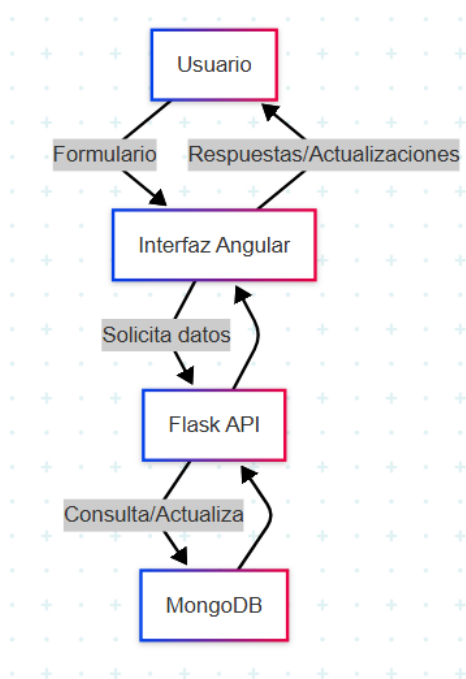








Diagrama de Flujo de Datos (DFD) Nivel 1:



9. Prototipo

Inventario

Buscar producto

	Producto 1	Cantidad	Status <input checked="" type="checkbox"/>	Eliminar	Editar >
	Producto 2	Cantidad	Status <input type="checkbox"/>	Eliminar	Editar >
	Producto 3	Cantidad	Status <input checked="" type="checkbox"/>	Eliminar	Editar >
	Producto 4	Cantidad	Status <input checked="" type="checkbox"/>	Eliminar	Editar >
	Producto 5	Cantidad	Status <input type="checkbox"/>	Eliminar	Editar >
	Producto 6	Cantidad	Status <input checked="" type="checkbox"/>	Eliminar	Editar >

[< Regresar](#)[Agregar Producto](#)

Agregar/Editar Producto

Nombre del Producto

Descripción

Categoría

Categoría

[< Regresar](#)[Agregar Producto](#)

10. Seguridad Implementada

- Autenticación con JWT
- Verificación en dos pasos mediante código enviado al correo (MFA)
- Protección de rutas del backend mediante decoradores
- CORS habilitado correctamente para comunicación entre frontend y backend
- Validación de datos en backend

11. Servicios Web

- **Servicio propio:**
API RESTful en Flask con rutas para autenticación, verificación, y CRUD de productos.
- **Servicio de terceros:**
Gmail API (a través de Flask-Mail) para el envío de códigos de verificación por correo electrónico como parte del proceso de autenticación multifactor.

12. Conclusiones

Este proyecto permitió aplicar conocimientos sobre desarrollo web fullstack, metodologías ágiles, arquitectura cliente-servidor, seguridad y pruebas. El desarrollo con Angular y Flask demostró ser una combinación eficiente para aplicaciones ligeras. Además, el uso de MongoDB como base de datos facilitó la gestión flexible de documentos. La correcta planificación, diseño modular y despliegue en plataformas como Render, permitieron obtener un sistema funcional, seguro y accesible desde cualquier navegador.

13. Repositorio y Enlace de Producción

GitHub: <https://github.com/omarRodFrag/inventario-web.git>

Página: <https://inventario-web-iozf.onrender.com/>