**IBMS**

**(Inventory and Billing Management System)**

The Inventory and Billing Management System is a web-based application designed to help small and medium-sized businesses manage their inventory, billing, and reporting processes efficiently. The system automates tasks such as tracking product stock, creating invoices, updating inventory, and generating analytical reports.

**PROJECT OVERVIEW**

This project is a full-stack web application named **IBMS**, organized into two major parts:

* **Backend:** Django (Python)
* **Frontend:** Next.js (React)

It provides structured management for inventory items, billing processes, and customer information. It also includes provisions for reporting and application settings.

**BACKEND TECHNOLOGIES**

**1. Django**

* **What:** A high-level Python web framework.
* **Why Used:** Serves as the backend framework for handling models, business logic, views, and RESTful APIs.
* **Importance:** Provides robust ORM, admin interface, and user management.
* **Suitability:** Well-suited for rapid development and complex backend workflows.

**Structure:**

* ibms\_backend/ibms\_backend/: Core project settings (ASGI/WSGI, URLs, settings).
* ibms\_backend/inventory/: Django app managing inventory, billing, and customers.
  + models.py: Database models
  + views.py: Logic for endpoints
  + urls.py: App-level routing
  + migrations/: Tracks database schema changes

**Other Backend Components:**

* manage.py: CLI for backend tasks
* venv/: Virtual environment for Python dependencies

**FRONTEND TECHNOLOGIES**

**1. Next.js**

* **What:** A React-based frontend framework with SSR and SSG support.
* **Why Used:** Enables fast, SEO-friendly, modular front-end development.
* **Importance:** Supports modern design structure with flexibility in routing and page rendering.
* **Suitability:** Excellent for scalable frontend development with React.

**Structure:**

* app/: Contains route-based folders like billing/, customers/, inventory/ with respective page.js files
* reports/, settings/: Future features placeholders
* layout.js: Root layout of the application
* page.js: Landing page
* globals.css: Global styling
* public/: Static assets (icons, images)
* Config Files: next.config.mjs, jsconfig.json, package.json

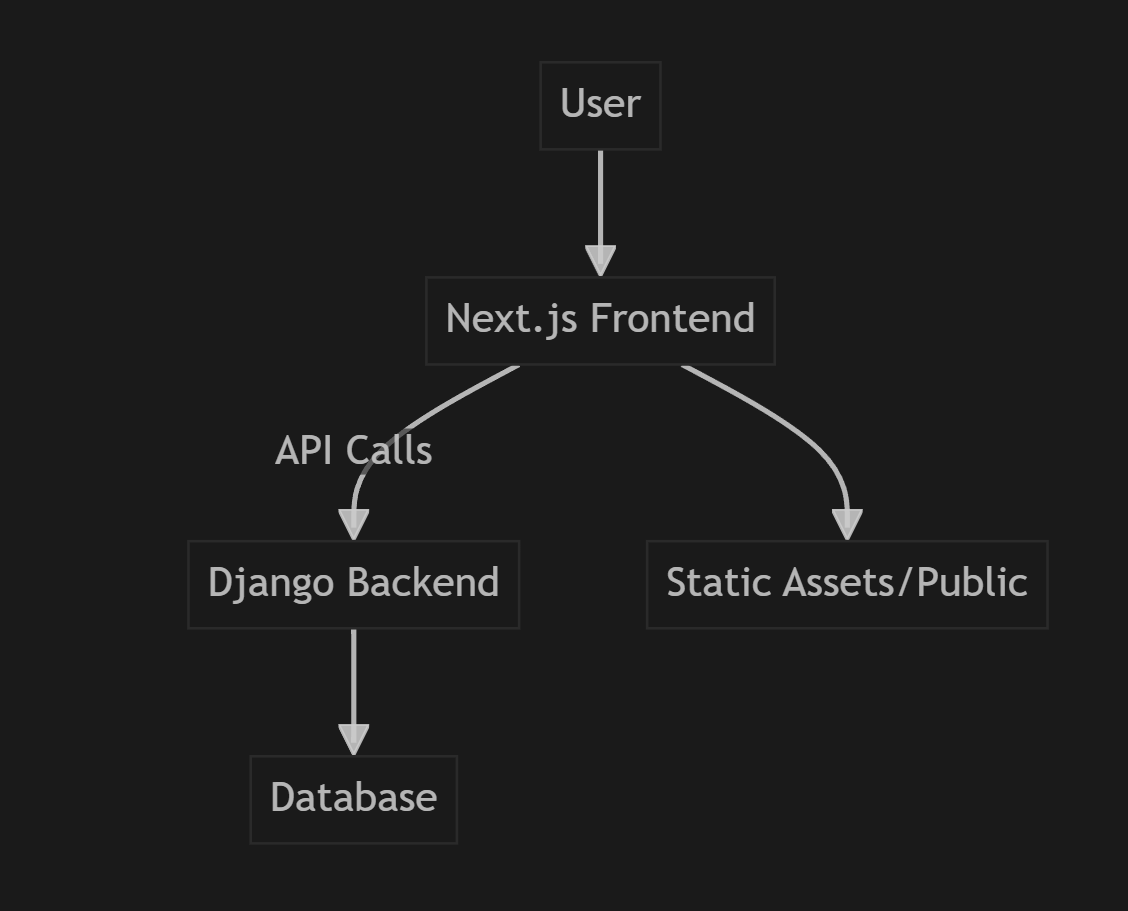
**KEY FEATURES (INFERRED)**

* **Inventory Management:** End-to-end flow for adding, updating, and tracking inventory items.
* **Billing System:** Handle invoice creation, billing records, and itemized billing details.
* **Customer Management:** Store and manage customer information.
* **Reports & Settings:** Placeholders for administrative tools and configurations.
* **Modular Development:** Clear separation between frontend and backend logic.

**DEVELOPMENT & USAGE**

* **Backend:** Uses Django ORM and views. Likely uses PostgreSQL, MySQL, or SQLite.
* **Frontend:** Built using Next.js App Router; communicates via REST API.
* **Virtual Environment:** Ensures isolated Python dependencies.
* **Node.js + NPM:** For managing frontend libraries and scripts.

**SUMMARY DIAGRAM**



This project demonstrates solid full-stack design principles with clear modularity and maintainability using Django and Next.js.