# **Smart Inventory & Delivery System (SIDeS)**

*An advanced ERPNext & Frappe Bootcamp Project*

A lightweight ERPNext extension that enables stock users to request inventory, triggers approval workflows, generates material requests, logs stock movements, and tracks fulfillment using QR codes.

## **💡 Why This Project?**

* **Relatable**: Inventory movement is a core process in many businesses.
* **Modular**: Demonstrates a focused concept through an actual feature.
* **Real-World Ready**: Demonstrates REST APIs, scripting, workflows, automation, and reporting.

## **📆 3-Day Curriculum Overview**

### **🟦 Day 1: Frappe Framework Mastery**

#### **🎯 Goal:**

Lay the foundation for SIDeS by building core Doctypes, workflows, and automation logic that reflect real inventory operations.

#### **✅ What You'll Learn:**

* Create Stock Request (parent) and Requested Items (child) Doctypes
* Validate requested quantities against actual stock
* Build and configure a workflow:
  + **Draft ➝ Pending Approval ➝ Approved ➝ Delivered ➝ Closed**
* Automate backend actions using on\_workflow\_action():
  + Create a **Material Request** (type: Material Issue) when approved
  + Create a **Stock Entry** (type: Material Issue) when delivered
* Add linking fields (stock\_request) in Material Request and Stock Entry
* Auto-fill item\_name on item selection via client script
* Role-based permissions:
  + **Stock User**: Can create and edit requests in Draft
  + **Inventory Manager**: Can approve, mark as delivered, and close requests

### **🟨 Day 2: Live Integrations**

#### **🎯 Goal:**

Enable system-to-system communication and improve delivery tracking using QR codes.

#### **✅ What You'll Learn:**

* How to simulate form submissions from a WordPress site to ERPNext using REST API
* How to handle external POST requests in Frappe to create a Delivery Note
* How to generate and store QR codes for delivery verification

#### **🔧 Features Covered:**

* **WordPress ➝ ERPNext** REST Integration:
  + Simulate a “Request Stock” form on a WordPress site
  + On submission, it posts data to a Frappe API and creates a Stock Request
* **QR Code Integration**:
  + On submit (Delivery Note), generate a QR code with:
    - A verification link or reference number
    - Store it in a custom qr\_code\_image field
* Create a page where users can scan and verify the QR code

## **🟥 Day 3: Reporting, Queues & Capstone Launch**

🎯 **Goal:**Enable insights through reports and dashboards, introduce background jobs, and kick off the capstone project.

✅ **What You'll Learn:**

* How to build a custom script report with filters
* How to create dashboard cards and chart reports
* How to queue a background job using the Frappe queue system
* Capstone project expectations and Q&A

🔧 **Features Covered:**

* **Script Report (Stock Request Status Report)**
  + Filter by workflow state, requested\_by, and date range
  + Show key request data
* **Dashboard (Stock Request Dashboard)**
  + Number cards: This Month's Requests, Delivered, Pending Approval
  + Bar chart: Ordered vs Delivered per Month
* **Frappe Queue (Create Order via API)**
  + Receive external API request
  + Validate and queue a Stock Request creation job
* **Fixtures**
  + How to manage custom customizations with fixtures

## **🏁 Capstone Extension Ideas**

* QR-based check-in for warehouse staff (scan on delivery)
* REST endpoint for external vendors to confirm delivery
* Mobile-friendly PWA for field delivery verification
* Sort Error from Yesterday
* Script report (Stock Request Status Report)
* Dashboard (Stock Request Dashboard)
* Workspace
* Frappe Queue (Create order via API)
* GitHub
* Deploy to DigitalOcean