**PDL Inventory Management System**

**Development Environment Setup Guide**

**Target OS:** Windows 10/11  
**Estimated Time:** 45-60 minutes  
**Last Updated:** October 20, 2025

**Table of Contents**

1. [System Requirements](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#1-system-requirements)
2. [Install Python](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#2-install-python)
3. [Install PostgreSQL](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#3-install-postgresql)
4. [Install Git](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#4-install-git)
5. [Install Visual Studio Code](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#5-install-visual-studio-code)
6. [Create Project Structure](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#6-create-project-structure)
7. [Setup Python Virtual Environment](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#7-setup-python-virtual-environment)
8. [Install Django and Dependencies](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#8-install-django-and-dependencies)
9. [Configure PostgreSQL Database](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#9-configure-postgresql-database)
10. [Initialize Django Project](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#10-initialize-django-project)
11. [Verify Installation](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#11-verify-installation)
12. [Troubleshooting](https://claude.ai/chat/98ead4d5-fa8b-412c-ab36-6781ce8c3db1#12-troubleshooting)

**1. System Requirements**

**Minimum Hardware**

* **CPU:** Intel Core i3 or equivalent (4 cores recommended)
* **RAM:** 8 GB (16 GB recommended)
* **Storage:** 50 GB free space (SSD recommended)
* **Network:** Internet connection for initial setup

**Software Requirements**

* Windows 10/11 (64-bit)
* Administrator access
* Antivirus temporarily disabled during installation (can re-enable after)

**2. Install Python**

**Step 2.1: Download Python**

1. Open your web browser
2. Go to: https://www.python.org/downloads/
3. Click **"Download Python 3.12.x"** (latest stable version)
4. Save the installer (e.g., python-3.12.0-amd64.exe)

**Step 2.2: Install Python**

1. **Locate and run** the downloaded installer
2. ⚠️ **IMPORTANT:** Check **"Add Python to PATH"** checkbox at the bottom
3. Click **"Install Now"**
4. Wait for installation to complete
5. Click **"Close"**

**Step 2.3: Verify Python Installation**

1. Press Win + R to open Run dialog
2. Type cmd and press Enter to open Command Prompt
3. Type the following command and press Enter:
4. python --version

**Expected Output:** Python 3.12.x

1. Check pip (Python package manager):
2. pip --version

**Expected Output:** pip 24.x.x from C:\Users\YourName\AppData\Local\Programs\Python\...

**✅ If you see version numbers, Python is installed correctly!**

**3. Install PostgreSQL**

**Step 3.1: Download PostgreSQL**

1. Go to: https://www.postgresql.org/download/windows/
2. Click **"Download the installer"**
3. You'll be redirected to EDB website
4. Download **PostgreSQL 16.x** (Windows x86-64)
5. Save the installer (e.g., postgresql-16.x-windows-x64.exe)

**Step 3.2: Install PostgreSQL**

1. **Run the installer** as Administrator
2. Click **"Next"** on the Setup Wizard
3. **Installation Directory:** Keep default C:\Program Files\PostgreSQL\16
4. **Select Components:** Check all boxes:
   * ✅ PostgreSQL Server
   * ✅ pgAdmin 4
   * ✅ Stack Builder
   * ✅ Command Line Tools
5. Click **"Next"**
6. **Data Directory:** Keep default C:\Program Files\PostgreSQL\16\data
7. Click **"Next"**
8. **Password:** Enter a strong password (e.g., postgres@123)
   * ⚠️ **REMEMBER THIS PASSWORD** – PDL#@1423
9. Confirm password
10. **Port:** Keep default 5432
11. **Locale:** Keep default (English, United States)
12. Click **"Next"** through remaining screens
13. Click **"Finish"**

**Step 3.3: Verify PostgreSQL Installation**

1. Open **Start Menu**
2. Search for **"pgAdmin 4"**
3. Open pgAdmin 4
4. It will open in your browser
5. Click on **"Servers"** in left sidebar
6. Click on **"PostgreSQL 16"**
7. Enter the password you set during installation
8. If you see "Dashboard" page, PostgreSQL is working!

**Step 3.4: Add PostgreSQL to PATH**

1. Press Win + X, select **"System"**
2. Click **"Advanced system settings"** (right sidebar)
3. Click **"Environment Variables"**
4. Under **"System variables"**, find **"Path"**
5. Click **"Edit"**
6. Click **"New"**
7. Add: C:\Program Files\PostgreSQL\16\bin
8. Click **"OK"** on all windows

**Verify:**

psql --version

**Expected Output:** psql (PostgreSQL) 16.x

**4. Install Git**

**Step 4.1: Download Git**

1. Go to: https://git-scm.com/download/win
2. Download will start automatically
3. Save the installer (e.g., Git-2.43.0-64-bit.exe)

**Step 4.2: Install Git**

1. **Run the installer**
2. Click **"Next"** through the welcome screens
3. **Select Components:** Keep defaults, ensure these are checked:
   * ✅ Git Bash Here
   * ✅ Git GUI Here
4. **Default Editor:** Select "Use Visual Studio Code as Git's default editor" (if available, else keep Vim)
5. **PATH Environment:** Select "Git from the command line and also from 3rd-party software"
6. **HTTPS Backend:** Keep default "Use the OpenSSL library"
7. **Line Ending Conversion:** Select "Checkout Windows-style, commit Unix-style"
8. **Terminal Emulator:** Select "Use Windows' default console window"
9. Click **"Next"** through remaining options (keep defaults)
10. Click **"Install"**
11. Click **"Finish"**

**Step 4.3: Verify Git Installation**

Open Command Prompt and type:

git --version

**Expected Output:** git version 2.43.x.windows.x

**5. Install Visual Studio Code**

**Step 5.1: Download VS Code**

1. Go to: https://code.visualstudio.com/
2. Click **"Download for Windows"**
3. Save the installer (e.g., VSCodeUserSetup-x64-1.x.x.exe)

**Step 5.2: Install VS Code**

1. **Run the installer**
2. Accept the agreement
3. **Select Destination Location:** Keep default
4. **Select Additional Tasks:** Check ALL boxes:
   * ✅ Create a desktop icon
   * ✅ Add "Open with Code" action to Windows Explorer file context menu
   * ✅ Add "Open with Code" action to Windows Explorer directory context menu
   * ✅ Register Code as an editor for supported file types
   * ✅ Add to PATH
5. Click **"Next"**, then **"Install"**
6. Click **"Finish"** (can check "Launch Visual Studio Code")

**Step 5.3: Install VS Code Extensions**

1. Open **VS Code**
2. Click **Extensions** icon (left sidebar) or press Ctrl + Shift + X
3. Search and install these extensions:
   * **Python** (by Microsoft) - Click "Install"
   * **Pylance** (by Microsoft) - Click "Install"
   * **Django** (by Baptiste Darthenay) - Click "Install"
   * **PostgreSQL** (by Chris Kolkman) - Click "Install"
   * **GitLens** (by GitKraken) - Click "Install"
   * **Thunder Client** (by Thunder Client) - Click "Install" (for API testing)

**6. Create Project Structure**

**Step 6.1: Create Project Directory**

1. Open **File Explorer**
2. Navigate to C:\
3. Create a new folder: C:\Projects
4. Inside Projects, create: C:\Projects\PDL\_IMS **(E:\Projects\PDL\_IMS)**

**Alternative using Command Prompt:**

cd C:\

mkdir Projects

cd Projects

mkdir PDL\_IMS

cd PDL\_IMS

**Step 6.2: Open Project in VS Code**

1. Open **VS Code**
2. Click **File** > **Open Folder**
3. Navigate to C:\Projects\PDL\_IMS **(E:\Projects\PDL\_IMS)**
4. Click **"Select Folder"**

**Step 6.3: Open Integrated Terminal**

1. In VS Code, press Ctrl + (backtick)` or
2. Click **Terminal** > **New Terminal**

You should see terminal at the bottom of VS Code showing:

PS C:\Projects\PDL\_IMS>

**7. Setup Python Virtual Environment**

A virtual environment keeps your project dependencies isolated from other Python projects.

**Step 7.1: Create Virtual Environment**

In VS Code terminal, type:

python -m venv venv

Wait 10-15 seconds. This creates a venv folder in your project.

**Step 7.2: Activate Virtual Environment**

**For PowerShell (default in VS Code):**

.\venv\Scripts\Activate.ps1

**If you get an error about execution policy:**

Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser

Then try activating again.

**For Command Prompt (cmd):**

venv\Scripts\activate.bat

**Success Indicator:** Your prompt should now show (venv) at the beginning:

(venv) PS C:\Projects\PDL\_IMS>

**Step 7.3: Upgrade pip**

python -m pip install --upgrade pip

**8. Install Django and Dependencies**

**Step 8.1: Create Requirements File**

1. In VS Code, create a new file: requirements.txt
2. Add the following content:

Django==5.0.1

djangorestframework==3.14.0

psycopg2==2.9.9

python-decouple==3.8

Pillow==10.2.0

django-cors-headers==4.3.1

django-filter==23.5

reportlab==4.0.9

openpyxl==3.1.2

celery==5.3.6

redis==5.0.1

django-celery-beat==2.5.0

gunicorn==21.2.0

whitenoise==6.6.0

1. Save the file (Ctrl + S)

**Step 8.2: Install All Dependencies**

In terminal (make sure virtual environment is activated):

pip install -r requirements.txt

This will take 3-5 minutes. You'll see progress as packages install.

**Step 8.3: Verify Django Installation**

django-admin --version

**Expected Output:** 5.0.1

**9. Configure PostgreSQL Database**

**Step 9.1: Create Database for IMS**

1. Open **pgAdmin 4** from Start Menu
2. In left sidebar, expand **"Servers"** > **"PostgreSQL 16"**
3. Enter your PostgreSQL password
4. Right-click on **"Databases"**
5. Select **"Create"** > **"Database..."**
6. **Database Name:** pdl\_ims\_db
7. **Owner:** Select postgres
8. Click **"Save"**

**Alternative using Command Line:**

Open Command Prompt (new window, not VS Code terminal):

psql -U postgres

Enter your PostgreSQL password, then:

CREATE DATABASE pdl\_ims\_db;

\l

You should see pdl\_ims\_db in the list. Type \q to exit.

**Step 9.2: Create Database User (Recommended)**

In pgAdmin or psql:

CREATE USER pdl\_ims\_user WITH PASSWORD 'pdl\_secure\_password\_123';

ALTER ROLE pdl\_ims\_user SET client\_encoding TO 'utf8';

ALTER ROLE pdl\_ims\_user SET default\_transaction\_isolation TO 'read committed';

ALTER ROLE pdl\_ims\_user SET timezone TO 'Asia/Dhaka';

GRANT ALL PRIVILEGES ON DATABASE pdl\_ims\_db TO pdl\_ims\_user;

**10. Initialize Django Project**

**Step 10.1: Create Django Project**

In VS Code terminal (venv activated):

django-admin startproject pdl\_ims\_backend .

**Note the dot (.) at the end** - this creates the project in the current directory.

Your folder structure should now look like:

C:\Projects\PDL\_IMS\

├── venv\

├── pdl\_ims\_backend\

│ ├── \_\_init\_\_.py

│ ├── settings.py

│ ├── urls.py

│ ├── asgi.py

│ └── wsgi.py

├── manage.py

└── requirements.txt

**Step 10.2: Create Environment Variables File**

1. Create a new file: .env
2. Add the following:

# Database Configuration

DB\_NAME=pdl\_ims\_db

DB\_USER=pdl\_ims\_user

DB\_PASSWORD=pdl\_secure\_password\_123

DB\_HOST=localhost

DB\_PORT=5432

# Django Secret Key (generate a new one later)

SECRET\_KEY=django-insecure-your-secret-key-here-change-in-production

# Debug Mode

DEBUG=True

# Allowed Hosts

ALLOWED\_HOSTS=localhost,127.0.0.1

1. Save the file

**Step 10.3: Create .gitignore File**

1. Create a new file: .gitignore
2. Add the following:

# Python

\*.py[cod]

\*$py.class

\_\_pycache\_\_/

\*.so

\*.egg

\*.egg-info/

dist/

build/

venv/

env/

# Django

\*.log

local\_settings.py

db.sqlite3

db.sqlite3-journal

media/

staticfiles/

# Environment Variables

.env

.env.local

# IDE

.vscode/

.idea/

\*.swp

\*.swo

# OS

.DS\_Store

Thumbs.db

# Database

\*.sql

\*.sqlite

1. Save the file

**Step 10.4: Configure Django Settings**

Open pdl\_ims\_backend/settings.py and make the following changes:

**1. At the top of the file, add:**

from decouple import config

from pathlib import Path

import os

**2. Replace the SECRET\_KEY line with:**

SECRET\_KEY = config('SECRET\_KEY')

**3. Replace the DEBUG line with:**

DEBUG = config('DEBUG', default=False, cast=bool)

**4. Replace ALLOWED\_HOSTS with:**

ALLOWED\_HOSTS = config('ALLOWED\_HOSTS', default='localhost,127.0.0.1').split(',')

**5. Find INSTALLED\_APPS and add to the list:**

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

# Third-party apps

'rest\_framework',

'corsheaders',

'django\_filters',

]

**6. Find MIDDLEWARE and add after SecurityMiddleware:**

MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware',

'corsheaders.middleware.CorsMiddleware', # Add this

'django.contrib.sessions.middleware.SessionMiddleware',

# ... rest of middleware

]

**7. Replace the DATABASES section completely:**

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.postgresql',

'NAME': config('DB\_NAME'),

'USER': config('DB\_USER'),

'PASSWORD': config('DB\_PASSWORD'),

'HOST': config('DB\_HOST', default='localhost'),

'PORT': config('DB\_PORT', default='5432'),

}

}

**8. Find LANGUAGE\_CODE and update timezone:**

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'Asia/Dhaka'

USE\_TZ = True

**9. At the end of the file, add:**

# Static files (CSS, JavaScript, Images)

STATIC\_URL = '/static/'

STATIC\_ROOT = os.path.join(BASE\_DIR, 'staticfiles')

STATICFILES\_DIRS = [os.path.join(BASE\_DIR, 'static')]

# Media files (User uploads)

MEDIA\_URL = '/media/'

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media')

# REST Framework Configuration

REST\_FRAMEWORK = {

'DEFAULT\_AUTHENTICATION\_CLASSES': [

'rest\_framework.authentication.SessionAuthentication',

],

'DEFAULT\_PERMISSION\_CLASSES': [

'rest\_framework.permissions.IsAuthenticated',

],

'DEFAULT\_PAGINATION\_CLASS': 'rest\_framework.pagination.PageNumberPagination',

'PAGE\_SIZE': 50,

'DEFAULT\_FILTER\_BACKENDS': [

'django\_filters.rest\_framework.DjangoFilterBackend',

'rest\_framework.filters.SearchFilter',

'rest\_framework.filters.OrderingFilter',

],

}

# CORS Settings (for future frontend)

CORS\_ALLOWED\_ORIGINS = [

"http://localhost:3000",

"http://127.0.0.1:3000",

]

1. Save the file (Ctrl + S)

**Step 10.5: Create Required Directories**

In VS Code terminal:

mkdir static

mkdir media

mkdir templates

**11. Verify Installation**

**Step 11.1: Test Database Connection**

python manage.py check --database default

**Expected Output:** System check identified no issues (0 silenced).

**Step 11.2: Run Initial Migrations**

python manage.py migrate

You should see output like:

Operations to perform:

Apply all migrations: admin, auth, contenttypes, sessions

Running migrations:

Applying contenttypes.0001\_initial... OK

Applying auth.0001\_initial... OK

...

**Step 11.3: Create Superuser**

python manage.py createsuperuser

Enter the following when prompted:

* **Username:** admin (hasib)
* **Email:** [admin@pdlgroup.com](mailto:admin@pdlgroup.com) (hasib@pantexbd.com)
* **Password:** (enter a secure password) (PDL#@1423)
* **Password (again):** (confirm password) (PDL#@1423)

**Step 11.4: Start Development Server**

python manage.py runserver

You should see:

Watching for file changes with StatReloader

Performing system checks...

System check identified no issues (0 silenced).

October 20, 2025 - 10:30:00

Django version 5.0.1, using settings 'pdl\_ims\_backend.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CTRL-BREAK.

**Step 11.5: Test Django Admin**

1. Open your web browser
2. Go to: http://127.0.0.1:8000/admin/
3. Login with:
   * **Username:** admin (hasib)
   * **Password:** (the password you created) (PDL#@1423)
4. You should see the Django Administration page!

**🎉 Congratulations! Your development environment is ready!**

Press Ctrl + C in terminal to stop the server.

**12. Troubleshooting**

**Issue: "python is not recognized"**

**Solution:**

* Reinstall Python and ensure "Add Python to PATH" is checked
* Or manually add Python to PATH (see Section 2.3)

**Issue: Cannot activate virtual environment**

**Solution (PowerShell):**

Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser

**Issue: "psycopg2" installation fails**

**Solution:**

pip install psycopg2-binary

Then update requirements.txt to use psycopg2-binary instead of psycopg2

**Issue: Cannot connect to PostgreSQL**

**Solution:**

1. Check if PostgreSQL service is running:
   * Press Win + R, type services.msc
   * Find "postgresql-x64-16"
   * Right-click > Start (if not running)
2. Verify database credentials in .env file
3. Test connection in pgAdmin

**Issue: Port 8000 already in use**

**Solution:**

python manage.py runserver 8001

(Use a different port)

**Issue: Migration errors**

**Solution:**

python manage.py migrate --run-syncdb

**Next Steps**

Now that your development environment is ready:

1. ✅ Python installed
2. ✅ PostgreSQL installed and configured
3. ✅ Django project initialized
4. ✅ Database connected
5. ✅ Admin panel accessible

**You're ready to proceed with:**

* Creating Django models from the database schema
* Implementing core modules
* Building the inventory management system

**Quick Reference Commands**

**Virtual Environment**

# Activate (PowerShell)

.\venv\Scripts\Activate.ps1

# Activate (CMD)

venv\Scripts\activate.bat

# Deactivate

deactivate

**Django Management**

# Run development server

python manage.py runserver

# Create migrations

python manage.py makemigrations

# Apply migrations

python manage.py migrate

# Create superuser

python manage.py createsuperuser

# Django shell

python manage.py shell

**Database**

# Access PostgreSQL

psql -U postgres

# Access specific database

psql -U pdl\_ims\_user -d pdl\_ims\_db

**Document Version:** 1.0  
**Last Updated:** October 20, 2025  
**Author:** Mahmudur Rahman