### **📌 Capstone Part 1: Idea & Planning Phase**

#### **✅ The Idea:**

The project is borne out of a personal pain point at work where I manually manage IT Assets with a Google Spreadsheet.

An **Asset Management API** that helps organizations efficiently track physical and digital assets across their lifecycle. It will cover registration, allocation, depreciation, maintenance, transfer of ownership, and disposal of assets.

#### **🎯 Main Features:**

1. **Asset Lifecycle Tracking** – from acquisition to disposal.
2. **Depreciation Calculations** – support for straight-line method.
3. **Ownership History** – logs transfers between users/departments.
4. **Maintenance Logs** – track repair history, upcoming servicing.
5. **Status Updates** – assign statuses such as active, in repair, retired.
6. **CRUD Operations** – for assets, departments, owners, and maintenance records.

#### **🌐 API Usage:**

No external API is required. This project focuses on designing and building a RESTful API from scratch using Django Rest Framework.

### **🧱 Project Structure & Planning**

#### **🗃️ Django Apps:**

1. assets – Manage asset registration, status, depreciation.
2. owners – Handle users/departments that own assets.
3. maintenance – Log repairs, servicing, and downtime.
4. history – Track ownership changes and status updates.

#### **🔄 API Endpoints:**

* **/api/assets/** GET, POST – list or register assets
* **/api/assets//** GET, PUT, DELETE – single asset operations
* **/api/assets//depreciation/** – fetch asset’s current value
* **/api/owners/** – manage owners or departments
* **/api/maintenance/** – manage maintenance logs
* **/api/history/** – view asset history

#### **🗃️ Core Models:**

* **Asset** – name, serial number, category, purchase\_date, cost, status, current\_owner, useful\_life
* **Owner** – name, type (department, individual), location
* **MaintenanceLog** – asset, date, description, cost
* **OwnershipHistory** – asset, from\_owner, to\_owner, transfer\_date

### **📅 5-Week Execution Plan**

#### **🔹 Week 1 – Idea & Planning Phase (✅ This Week)**

* Define project scope and core features
* Select models and design rough structure
* Draft week-by-week project plan (this document)

#### **🔹 Week 2 – Design Phase**

* Draw ERD Diagram (Entity Relationship Diagram)
* Finalize database schema
* Plan serializers and model relationships

#### **🔹 Week 3 – Start Building**

* Initialize Django project and apps
* Build and test models, admin config
* Set up API routes for assets and owners

#### **🔹 Week 4 – Continue Building**

* Add depreciation logic, ownership history, and maintenance logs
* Implement status transitions
* Write tests for major endpoints

#### **🔹 Week 5 – Finalize & Submit**

* Clean up code, add documentation (README + Swagger)
* Test thoroughly for edge cases and validation
* Package and submit final version

### **📝 Additional Notes**

* **Authentication (Optional)**: Token-based or simple API key access for different users
* **Tech Stack**: Python, Django, Django REST Framework, SQLite (or PostgreSQL)
* **Testing**: Use DRF’s APITestCase for unit/integration tests
* **Documentation**: Swagger/OpenAPI or DRF’s built-in API docs