**Django DRF Ecommerce**

Phase 1

Inventory RESTful API

V1.0.1

**Table of Contents**

[1. Introduction 3](#_Toc122270645)

[1.1. Purpose of Document 3](#_Toc122270646)

[1.2. Project Scope 3](#_Toc122270647)

[2. Functional Objectives 3](#_Toc122270648)

[2.1. High Priority 3](#_Toc122270649)

[3. Supportability 3](#_Toc122270650)

[4. Security 4](#_Toc122270651)

[5. RESTful API documentation 4](#_Toc122270652)

[6. Interfaces 4](#_Toc122270653)

[7. Context Diagram 4](#_Toc122270654)

[Appendix A: Functional Requirements 5](#_Toc122270655)

[Appendix B: Use case specification 6](#_Toc122270656)

# Introduction

## Purpose of Document

This is a Requirements Specification document for a new web-based product inventory RESTful API system.

## Project Scope

The scope of this project is a web-enabled inventory system RESTful API interface that supports client product data requests.

**1.3 Responsibilities**

The primary responsibilities of the new system:

* The new system will provide a client interface returning detailed and accurate update-to-date product information.

# Functional Objectives

## High Priority

1. The system shall provide the following data collection API endpoints:
   1. Return all categories
   2. Return a specified product and associated metadata
   3. Return Product(s), including associated product metadata from a specified category
2. The system shall allow employees to add and administrate product inventory
3. The system shall reflect new and changed products and product data changes x minutes of the database being updated by the product owner.

# Supportability

* The system should be able to accommodate new products and product lines
* The system should support multiple types of product types with varying characteristics. In addition, the system should accommodate physical shippable and downloadable products.

# Security

* The system will provide password-protected access to product data management and administration.

# RESTful API documentation

* The system shall provide web-based documentation detailing all API endpoints and endpoint-specific details.

# Interfaces

The system must interface with

* An SQLite database and be compatible with future migration to other database technologies.

# Context Diagram

# Appendix A: Functional Requirements

User Requirements

* Return a single product and associated product lines
* Return a list of all products
* Return a list of products by category
* Return a list of products by product attribute
* Return a list of new products
* Sort a list of products by price

Low Priority

* Add a review for a product
* Return a list of products on promotion

Business Requirements

* Add, Update or Delete new products
* Add, Return or Update details related to product stock levels
* Report: Return a list of total products sold per product
* Report: Return a list of newly added products for a given timeframe
* Report: Return a list of low or not-in-stock products
* Sort a list of products by date
* Return the name of the user who entered the product into the database
* When was the product added to the database?

System Requirements

* Return product data and images
* Return a single product and associated sub-product

# Appendix B: User Story Analysis

|  |  |
| --- | --- |
| **Overview** | |
| **Title** | Customer Product Browsing Behaviour |
| **Description** | Identifying a basic customer behavioural interaction when browsing products |
| **Actors and Interfaces** | Customer / Web User |
| **Initial Status and Preconditions** | Assumption that customer enters from the root/homepage |
| **Basic Flow** | |
| Step1: Land on the homepage  Step2: Select a product category  Step3: Browse, select, and view individual products related to the selected category  Step4: Select and view individual product-line details | |
| **Alternative Flow(s)** | |
| * Customers may prefer searching for the product using keyword search features * Customers may navigate to a product from an internal promotional panel | |

|  |  |  |
| --- | --- | --- |
| **User Type** | **Activity** | **User Story** |
| Web User | Browse Products | Step1: Land on the homepage |
| Step2: Select a product category |
| Step3: Select and view individual products related to the selected category |
| Step4: Inspect individual product-line details |

|  |  |
| --- | --- |
| **Functional Specifications** | **Status** |
| Return all categories | a |
| Return all products filtered by category | a |
| Return individual product and product-line by (x) | a |

# Appendix C: Database Table Specification

|  |  |  |  |
| --- | --- | --- | --- |
| **Category** | | | |
| **Key** | **Field Name** | **Data Type** | **Field Level Physical/Logical Constraints** |
| PK | id | BigAutoField |  |
|  | name | CharField | max\_length = 235  unique=True |
|  | slug | SlugField | max\_length = 255  unique=True |
|  | is\_active | BooleanField | Default=False |
| FK | parent | TreeForignKey | on\_delete=PROTECT  null=True  Blank=True |
|  |  |  |  |
| **Custom Validation** | | | |
|  |  | | |
| **Methods/Behaviours** | | | |
| 1 | Object string representation = name | | |
| 2 | is\_active() model manager queryset method to filter products by is\_active Boolean field | | |
| **Test Cases / Log** | | | |
| **id** | **Test Description/Expectation** | | |
| 1 | Return error when field name max\_length > 235 | | |
| 2 | Return error when field slug max\_length > 255 | | |
| 3 | Return unique name error when entering a name which already exists in the category table | | |
| 4 | Return unique name error when entering a slug which already exists in the category table | | |
| 5 | Field is\_active on creating a new record = false | | |
| 6 | Deleting a parent category raises ProtectedError | | |
| 7 | On inserting a new record, parent field to remain null | | |
| 8 | is\_active() model manager queryset method to return objects where is\_active field = True | | |
| 9 | Default object manager to return all products when used with all() method | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Product** | | | |
| **Key** | **Field Name** | **Data Type** | **Field Level Physical/Logical Constraints** |
| PK | id | BigAutoField |  |
|  | name | CharField | max\_length = 235  unique=True |
|  | slug | SlugField | max\_length = 255  unique=True |
|  | pid | CharField | max\_length = 10  unique=True |
|  | description |  |  |
|  | is\_digital |  | Default=False |
| FK | category |  | on\_delete=PROTECT  null=True  Blank=True |
|  | is\_active | BooleanField | Default=False |
|  | created\_at | DateTimeField |  |
| FK | product\_type |  |  |
| M2M | attribute\_value |  |  |
| **Custom Validation** | | | |
|  |  | | |
| **Methods/Behaviours** | | | |
| 1 | Object string representation = name | | |
| 2 | is\_active() model manager queryset method to filter products by is\_active Boolean field | | |
| **Test Cases / Log** | | | |
| **id** | **Test Description/Expectation** | | |
| 1 | Return error when field name max\_length > 235 | | |
| 2 | Return error when field slug max\_length > 255 | | |
| 3 | Return error when field pid max\_length > 10 | | |
| 4 | Field is\_digital on creating a new record = false | | |
| 5 | Deleting a category parent raises ProtectedError | | |
| 6 | is\_active() model manager queryset method to return objects where is\_active field = True | | |
| 7 | Default object manager to return all products when used with all() method | | |
| 8 | Object string representation return = {name field} | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Product Line** | | | |
| **Key** | **Field Name** | **Data Type** | **Field Level Physical/Logical Constraints** |
| PK | id | BigAutoField |  |
|  | price | DecimalField | decimal\_places=2  max\_digits=5 |
|  | sku | CharField | max\_length=10 |
|  | stock\_qty | IntegerField |  |
| FK | product |  | on\_delete=PROTECT |
|  | is\_active | Boolean | default=False |
|  | order | CustomField | blank=True |
|  | weight | FloatField |  |
|  | created\_at | DateTimeField | editable=False, |
| FK | product\_type |  |  |
| M2M | attribute\_value |  |  |
| **Custom Validation** | | | |
|  |  | | |
| **Methods/Behaviours** | | | |
| 1 | Object string representation = sku | | |
| 2 | is\_active() model manager queryset method to filter products by is\_active Boolean field | | |
| **Test Cases / Log** | | | |
| **id** | **Test Description/Expectation** | | |
| 1 | Return error when price field decimal places > 2 | | |
| 2 | Return error when price field max digits > 5 | | |
| 3 | Return error when field sku max\_length > 10 | | |
| 4 | Field is\_active on creating a new record = False | | |
| 5 | Deleting a product parent raises ProtectedError | | |
| 6 | is\_active() model manager queryset method to return objects where is\_active field = True | | |
| 7 | Default object manager to return all objects when used with all() method | | |
| 8 | Object string representation return = {sku field} | | |
| 9 | Return error when a new record order field number is not unique | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Product Image** | | | |
| **Key** | **Field Name** | **Data Type** | **Field Level Physical/Logical Constraints** |
| PK | id | BigAutoField |  |
|  | alternative\_text | CharField | max\_length=100 |
|  | url | ImageField |  |
|  | order | CustomField |  |
| FK | productline |  |  |
| **Custom Validation** | | | |
|  |  | | |
| **Methods/Behaviours** | | | |
| 1 | Object string representation = {product\_line sku} | | |
| 2 | is\_active() model manager queryset method to filter products by is\_active Boolean field | | |
| **Test Cases / Log** | | | |
| **id** | **Test Description/Expectation** | | |
| 1 | Return error when field alternative\_text max\_length > 100 | | |
| 2 | Return error when a new record order field number is not unique | | |

The document you've uploaded is a requirements specification for a Django DRF (Django Rest Framework) Ecommerce system, specifically for a RESTful API related to inventory management. It appears to be version 1.0.1 and appears to be in the first phase of the system's development.

Here's a detailed breakdown of the document:

Introduction: This section outlines the purpose of the document and the scope of the project. The document's purpose is to specify the requirements for a new web-based product inventory RESTful API system. The project scope is to build a web-enabled inventory system with a RESTful API interface that can manage and respond to client requests for product data.

Responsibilities: This section details the primary responsibilities of the new system. It is expected to provide a client interface that can return up-to-date and detailed product information.

Functional Objectives: In this section, the high-priority functionalities are listed. The new system must provide API endpoints for returning all categories, a specific product with its metadata, and products from a specific category. It should also allow employees to manage product inventory and reflect any changes in product data within minutes of an update by the product owner.

Supportability: This section discusses the system's adaptability requirements. The system should support new products and product lines, as well as multiple types of products with different characteristics, including both physical products and downloadable ones.

Security: This section outlines the security requirements of the system, stating that access to product data management and administration should be password-protected.

RESTful API documentation: The system should provide web-based documentation that details all API endpoints and their specific characteristics.

Interfaces: The system should interface with an SQLite database and be compatible with future migration to other database technologies.

User Requirements: This section lists the data retrieval requirements from the user perspective. These include retrieving single or multiple products and their related product lines, retrieving products by category or attributes, retrieving new products, and sorting products by price. There are also low-priority requirements like adding product reviews and retrieving promotional products.

Business Requirements: This section outlines the operations related to product and stock management, such as adding, updating, or deleting products and managing stock levels. It also lists several reporting requirements, such as generating reports of total products sold per product, newly added products in a given timeframe, low or out-of-stock products. It also mentions the ability to sort products by date, retrieve the name of the user who entered a product into the database, and when a product was added to the database.

System Requirements: This section details the requirements related to product data and image retrieval, and the retrieval of a single product and its associated sub-product.

The document also mentions appendices with more detailed information, including a functional requirements list (Appendix A), a user story analysis (Appendix B), and a database table specification (Appendix C). However, the content of these appendices was not included in the parts of the document you've uploaded.

Please note that this is a high-level summary and analysis of the document. If you need a more detailed analysis of a specific section or term, please let me know!