StockMaster

Release 1.0.0

Gabriel Santiago Robles Robles, Jesús Antonio Flores Briones, Mario Alejandro Castro Lerma, Manuel Iván Melendez Rivera

Nov 09, 2023

Table of Contents

Inventory Views

Inventory.views.**AddLastDate** (*filteredLabels*, *filteredList*, *month*, *year*, *product*, *attribute*)

Add the last date to the lists of labels and filtered data.

Args:

filteredLabels: List of labels. filteredList: List of filtered data. month: Current month. year: Current year. product: Product object. attribute: Attribute to be added.

Returns:

Modified lists of labels and filtered data.

Inventory.views.**AddProduct** (*request*)

Add a new product based on the form data.

Args:

request: HTTP request object.

Returns:

JSON response indicating the success or failure of the product addition.

Inventory.views.**AddProducts** (*request*)

Render the 'add-product' page and process the form data for adding products.

Args:

request: HTTP request object.

Returns:

Rendered 'add-product' page.

Inventory.views.**CreateOrderItem** (*product_data*, *order*)

Create an order item based on the product data.

Args:

product_data: Dictionary containing product data. order: InputOrder object.

Returns:

None

Inventory.views.**DeleteLeftFromSequence** (*text*, *sequence*)

Delete a sequence from the left side of a text.

Args:

text: Text string. sequence: Sequence to be deleted.

Returns:

Modified text string.

Inventory.views.**EditProduct** (*request*, *productid*)

Render the product edit page and process the form data for updating product information.

Args:
request: HTTP request object. productid: ID of the product to be edited.

Returns:
Rendered product edit page.

`Inventory.views.`**`FilterInventory`** (*request*)
Filter the products in the inventory based on different criteria.

Args:
request: HTTP request object.

Returns:
Rendered HTML page with filtered products.

`Inventory.views.`**`FilterSameDates`** (*labels, otherList*)
Filter out the same dates from the lists of labels and other data.

Args:
labels: List of labels. otherList: List of other data.

Returns:
Filtered lists of labels and other data.

`Inventory.views.`**`GetProductPriceData`** (*request*)
Retrieve product price data for a specific month and year.

Args:
request: HTTP request object.

Returns:
JSON response with the product price data.

`Inventory.views.`**`GetProductQuantityData`** (*request*)
Retrieve product quantity data for a specific month and year.

Args:
request: HTTP request object.

Returns:
JSON response with the product quantity data.

`Inventory.views.`**`HandleProductData`** (*request*)
Handle the data for multiple products received from the frontend.

Args:
request: HTTP request object.

Returns:
JSON response indicating the success or failure of handling the product data.

`Inventory.views.`**`Inventory`** (*request*)
Render the inventory page with products filtered by the search query.

Args:
request: HTTP request object.

Returns:
Rendered HTML page with filtered products.

`Inventory.views.`**`MapCategory`** (*value*)
Map a value to a specific category.

Args:
value: Value to be mapped.

Returns:
Mapped category.

`Inventory.views.`**`ProductDetails`** (*request*, *productid*)
Renders the product details page for a specific product.

Args:
request (HttpRequest): The HTTP request. productid (int): The ID of the product.

Returns:
HttpResponse: The rendered product details page with information about the specified product.

`Inventory.views.`**`ProductGraph`** (*request*, *productid*)
Render the product graph page with historical data of the product.

Args:
request: HTTP request object. productid: ID of the product.

Returns:
Rendered product graph page.

`Inventory.views.`**`RemoveAccents`** (*input_str*)
Remove accents from a given string.

Args:
input_str: Input string.

Returns:
String without accents.

`Inventory.views.`**`VerifyAndUpdate`** (*product*)
Verify and update product information.

Args:
product: Dictionary containing product information.

Returns:
None

`Inventory.views.`**`VerifyProductForm`** (*request*)
Verify if the product form contains the required fields.

Args:
request: HTTP request object.

Returns:
Boolean indicating whether the form is valid.

Dashboard Views

Dashboard.views.**Dashboard** (*request*)

Displays the dashboard with various statistics and data about products, orders, and more.

Returns:

A rendered dashboard page with relevant statistics and data.

Dashboard.views.**LoginPage** (*request*)

Displays the login page or redirects to the dashboard if the user is already authenticated.

Returns:

A rendered login page or a redirection to the dashboard if the user is already logged in.

Dashboard.views.**LogoutPage** (*request*)

Logs out the user and redirects to the login page.

Returns:

Redirection to the login page.

Dashboard.views.**RemoveAccents** (*input_str*)

Removes accents from a string.

Args:

input_str (str): The string from which to remove accents.

Returns:

str: The string without accents.

Dashboard.views.**Workers** (*request*)

Displays a list of workers and provides a search functionality.

Returns:

A rendered workers page with a list of workers and search functionality.

Orders Views

OutputHistory.views.**InputHistory** (*request*)

Displays the input history page with search and date range filtering.

Returns:

A rendered input history page with search and date range filtering.

OutputHistory.views.**OutputHistory** (*request*)

Displays the output history page with search and date range filtering.

Returns:

A rendered output history page with search and date range filtering.

OutputHistory.views.**RemoveAccents** (*input_str*)

Removes accents from a string.

Args:

input_str (str): The string from which to remove accents.

Returns:

str: The string without accents.

Graphs Views

`Charts.views.`**`GetStockMonth`**(*request*)

Retrieves the stock data for a specific month and year.

Args:

request (HttpRequest): The HTTP request.

Returns:

JsonResponse: The stock data for the specified month and year.

`Charts.views.`**`ReportCharts`**(*request*)

Renders the report charts page with data visualizations for product categories.

Args:

request (HttpRequest): The HTTP request.

Returns:

HttpResponse: The rendered report charts page with data from the current month and year visualizations for product categories.

Product Model

class Product.models.**HistoricalProduct** (*id*, *name*, *SKU*, *price*, *quantity*, *image*, *threshold*, *category*, *isExternal*, *history_id*, *history_date*, *history_change_reason*, *history_type*, *history_user*)

exception DoesNotExist

exception MultipleObjectsReturned

static get_default_history_user (*instance*)
Returns the user specified by *get_user* method for manually creating historical objects

instance_type
alias of Product.models.Product

property next_record
Get the next history record for the instance. *None* if last.

property prev_record
Get the previous history record for the instance. *None* if first.

revert_url ( )
URL for this change in the default admin site.

class Product.models.**Product** (**args*, ***kwargs*)
Represents a product in the inventory.

exception DoesNotExist

exception MultipleObjectsReturned

SKU
str: The stock keeping unit of the product.

category
str: The category of the product.

history = <django.db.models.manager.HistoryManagerFromHistoricalQuerySet object>
HistoricalRecords: Used to manage historical modifications of the product.

image
str: The image associated with the product.

isExternal
bool: Indicates if the product is acquired from outside the institution.

isLowStock ( )
Check if the product is in low stock.

Returns:
bool: True if the product is in low stock, False otherwise.

name
str: The name of the product.

price
float: The price of the product.

quantity
int: The current stock quantity of the product.

save_without_historical_record (**args*, ***kwargs*)
Save model without saving a historical record
Make sure you know what you're doing before you use this method.

threshold
int: The threshold indicating low stock.

Worker Model

class `Workers.models.`**`Worker`** (**args, **kwargs*)

Represents an employee in the organization.

exception `DoesNotExist`

exception `MultipleObjectsReturned`

`employeeNumber`

int: The unique employee number.

`name`

str: The name of the employee.

`workArea`

str: The work area of the employee.

Input Order Models

class InputHistory.models.**InputOrder** (**args*, ***kwargs*)
Represents an input order in the system.

exception DoesNotExist

GetItems ( )
Retrieve the items associated with the input order.

Returns:
QuerySet: The items associated with the input order.

GetTotal ( )
Calculate the total cost of the input order.

Returns:
float: The total cost of the input order.

exception MultipleObjectsReturned

date_created
DateTimeField: The date and time when the order was created.

specialNotes
str: Special notes related to the input order.

class InputHistory.models.**InputOrderItem** (**args*, ***kwargs*)
Represents an item in an input order.

exception DoesNotExist

exception MultipleObjectsReturned

inputOrder
ForeignKey: The input order associated with the item.

product
ForeignKey: The product associated with the input order item.

quantity
PositiveIntegerField: The quantity of the product in the input order item.

Output Order Models

class OutputHistory.models.**OutputOrder** (**args*, ***kwargs*)

Represents an output order in the system.

Attributes:

worker (ForeignKey): The worker associated with the output order. date_created (DateTimeField): The date and time when the order was created.

exception DoesNotExist

GetItems ( )

Retrieve the items associated with the output order.

Returns:

QuerySet: The items associated with the output order.

GetTotal ( )

Calculate the total cost of the output order.

Returns:

float: The total cost of the output order.

exception MultipleObjectsReturned

date_created

DateTimeField: The date and time when the order was created.

worker

ForeignKey: The worker associated with the output order.

class OutputHistory.models.**OutputOrderItem** (**args*, ***kwargs*)

Represents an item in an output order.

exception DoesNotExist

exception MultipleObjectsReturned

outputOrder

ForeignKey: The output order associated with the item.

product

ForeignKey: The product associated with the output order item.

quantity

PositiveIntegerField: The quantity of the product in the output order item.

- genindex

- modindex
- search

c

d

i

o

p

w

L

M

N

O

P

Q

R

S

T

V

W