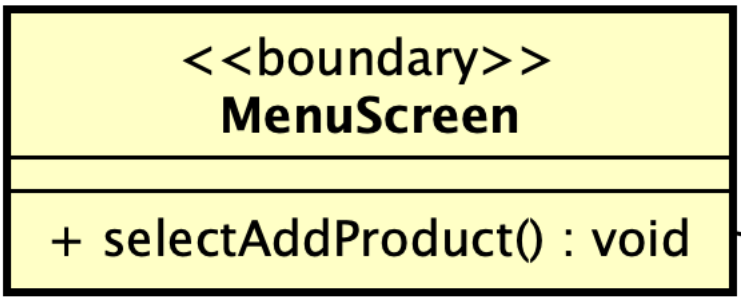
# addProductToStore

1. **Design for class "MenuSceen"**




1.1. Attribute Design:
None

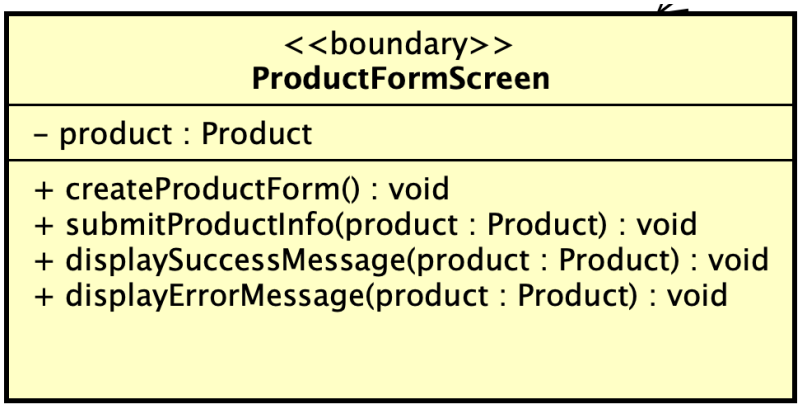
1.2. Operation Design:

| STT | Name | Return Type | Description (Purpose) |
|---|---|---|---|
| 1 | selectAddProduct() | void | Triggers the process of adding a new product |

- **selectAddProduct Method**
  - *Parameter:* None
  - *Exception*: NavigationException if the system fails to navigate to the product addition screen.
  - *How to use parameters/attributes:*

This method is executed when the user selects the option to add a new product from the menu. It interacts with the ProductController to start the product creation process.

2. **Design for class "ProductFormScreen"**




2.1. Attribute Design

| STT | Name | Data type | Default | Description |
|---|---|---|---|---|
| 1 | product | Product | None | Stores the product information being created or edited |

2.2. Operation Design:

| STT | Name | Return Type | Description (Purpose) |
|---|---|---|---|
| 1 | createProductForm() | void | Initializes and displays the product form |
| 2 | submitProductInfo(product: Product) | void | Sends product details to be pro |
| 3 | displaySuccessMessage(product: Product) | void | Shows a success message after product submission |
| 4 | displayErrorMessage(product: Product) | void | Displays an error message if submission fails |

- **createProductForm Method**
  - *Parameter:* None
  - *Exception:* FormLoadException if the product form fails to initialize.
  - *How to use parameters/attributes:*

This method initializes the form for entering product details and prepares the UI components.

- **submitProductInfo Method**
  - *Parameter:*

- product: Instance of Product containing details to be submitted.

  - *Exception:*

- InvalidProductException if required product details are missing or incorrect.
- DatabaseException if an error occurs while saving product information.

  - *How to use parameters/attributes:*

This method sends product data to ProductController for validation and further processing.

- **displaySuccessMessage Method**
  - *Parameter:*

- product: Instance of Product for which the success message is displayed.

  - *Exception:* None
  - *How to use parameters/attributes:*

Displays a confirmation message to the user after successful product submission.

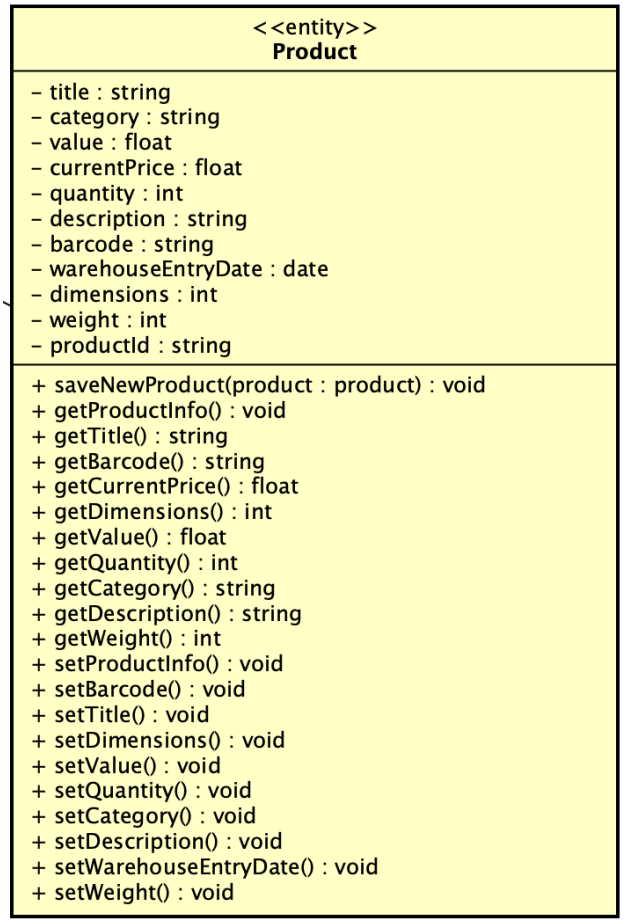
- **displayErrorMessage Method**
  - *Parameter:*

- product: Instance of Product that failed to be processed.

  - *Exception:* None
  - *How to use parameters/attributes:*

This method is responsible for notifying the user of an error during product submission.

3. **Design for class “Product”**




### 3.1. Attribute design

| STT | Name | Date type | Default value | Description |
|---|---|---|---|---|
| 1 | title | string | None | Title of the product |
| 2 | category | string | None | Category of the product |
| 3 | value | float | 0.0 | Value of the product |
| 4 | currentPrice | float | 0.0 | Current price of the product |
| 5 | quantity | int | 0 | Quantity available in stock |
| 6 | description | string | None | Description of the product |
| 7 | barcode | string | None | Barcode number |
| 8 | warehouseEntryDate | date | None | Date when product was added to warehouse |
| 9 | dimensions | int | 0 | Dimensions of the product |
| 10 | weight | Int | 0 | Weight of the product |
| 11 | productId | string | Genersted by system | Unique ID for the product |

### 3.2. Operation Design

| STT | Name | Return type | Description (purpose) |
|---|---|---|---|
| 1 | saveNewProduct(product) | void | Saves the new product |
| 2 | getProductInfo() | void | Retrieves product information |
| 3 | getTitle() | string | Returns the title of the product |
| 4 | getBarcode() | string | Returns the barcode of the product |
| 5 | getCurrentPrice() | float | Returns the current price of the product |
| 6 | getDimensions() | int | Returns the dimensions of the product |
| 7 | getValue() | float | Returns the value of the product |
| 8 | getQuantity() | int | Returns the available quantity of the product |
| 9 | getCategory() | string | Returns the category of the product |
| 10 | getDescription() | string | Returns the description of the product |
| 11 | getWeight() | int | Returns the weight of the product |
| 12 | setProductInfo() | void | Sets product information |
| 13 | setBarcode() | void | Sets barcode |
| 14 | setTitle() | void | Sets title |
| 15 | setDimensions() | void | Sets dimensions |
| 16 | setValue() | void | Sets value |
| 17 | setQuantity() | void | Sets quantity |
| 18 | setCategory() | void | Sets category |
| 19 | setDescription() | void | Sets Description |
| 20 | setWarehouseEntryDate() | void | Sets warehouse entry date |
| 21 | setWeight() | void | Sets weight |

- **saveNewProduct method**
  - *Parameter:*

- product: Instance of Product containing details to be saved.

  - *Exception:*

- InvalidInputException if product details contain invalid data.
- DuplicateProductException if productId already exists in the system.

  - *How to use parameters/attributes:*

This method is called when a new product is added through the system. It ensures that all mandatory fields are provided before saving the product.

- **Getter methods** (getTitle, getBarcode, etc.)
  - *Parameter:*None
  - *Exception:*None
  - *How to use parameters/attributes:*

These getter methods are used by UI screens and controllers to retrieve product details when needed.

- **Setter methods** (setTitle, setBarcode, etc.)
  - *Parameter:*

- Corresponding attribute value to be updated.

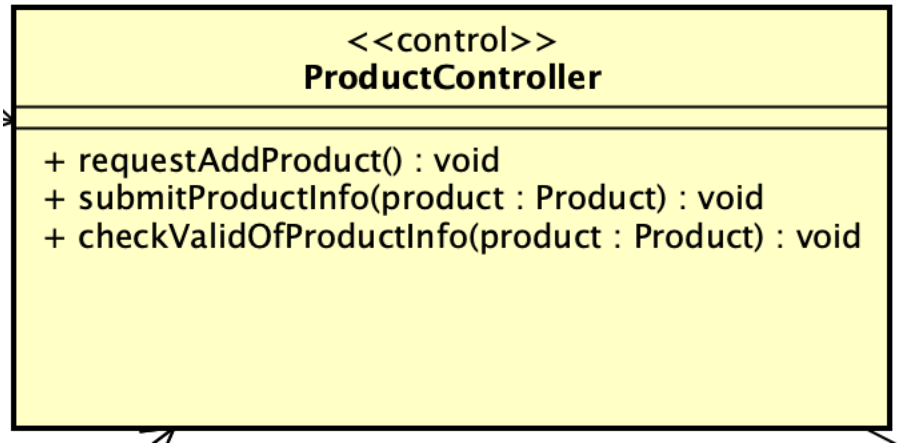
  - *Exception:*

- InvalidInputException if the provided value does not meet constraints.

  - *How to use parameters/attributes:*

These setter methods are used to update product details dynamically in the system.

## 4. Design for class "ProductController"




### 4.1. Attribute design

None

### 4.2. Operation design

| STT | Name | Return type | Description (purpose) |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 1 | requestAddProduct() | void | Initiates the process to add a new product |
| 2 | submitProductInfo(product) | void | Submits product information for processing |
| 3 | checkValidOfProductInfo(product) | void | Checks whether the given product information is valid |

- **requestAddProduct method**
  - *Parameter:* None
  - *Exception:*

- UnauthorizedAccessException if the user does not have permission to add a product.

  - *How to use parameters/attributes:*

This method is triggered when the user selects the option to add a new product. It initializes the product entry process.

- **submitProductInfo method**
  - *Parameter:*

- product: Instance of Product containing details to be submitted.

  - *Exception:*

- InvalidProductException if required product details are missing or incorrect.
- DatabaseException if there is an error saving the product to the database.

  - *How to use parameters/attributes:*

This method is responsible for taking the filled product details from the form and passing them to the system for validation and storage.

- **checkValidOfProductInfo method**
  - *Parameter:*

- product: Instance of Product to be validated.

- *Exception:*

- InvalidProductException if product details are incomplete or incorrect.

  - *How to use parameters/attributes:*

This method verifies whether the provided product details meet the required criteria before submission. It ensures data integrity and correctness before adding the product to the system.