

# SDV701

Assignment Two

Stage 1

Oleg Sivers

2020

## Contents

Business scenario .....	3
Database Schema.....	4
Applications.....	4
Windows WPF Client.....	4
Navigation .....	4
Main window .....	5
Orders window.....	5
Category selection window.....	6
Category details window.....	6
Parts base window (edit item window) .....	7
Web Client.....	8
Navigation .....	8
Index page.....	8
Category detail page .....	9
Order placement page .....	10
UML.....	11
Domain model classes.....	11
Server .....	11
Web client .....	12
Windows WPF client .....	13

## Business scenario

Our company is going to sale network equipment and PC parts related to networking. According to requirements we should provide for the company a management tool to modify what's in stock and check orders and a frontend application for clients, where order placement will occur.

Sample list of items:

- Network interface card
- Network switch
- Wireless router
- Wired router

All devices or parts within the system will have this common fields: PartsID, Name, CategoryID, Currency, Price, QtyInStock, LastModified.

Parts are going to be categorized on three categories:

- Wired
- Wireless
- Wired and Wireless

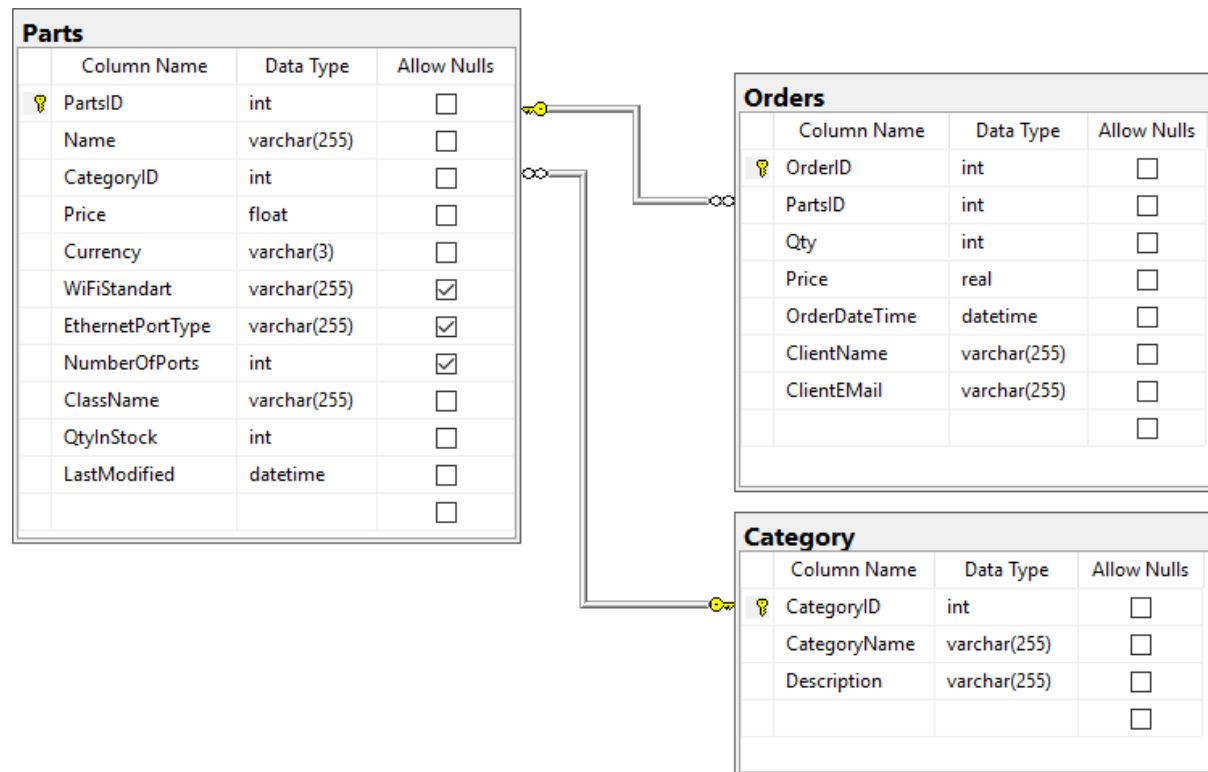
Each category will include category name and description fields.

There are extra parameters extends common parameters of parts, depending of category:

- Wired devices have extra parameters: Number of ports and port speed (10/100/1000)
- Wireless devices extra parameter is wireless standard (a\b\g\n\ac\6)
- Combined devices are having all extra parameters of wired and wireless devices

## Database Schema

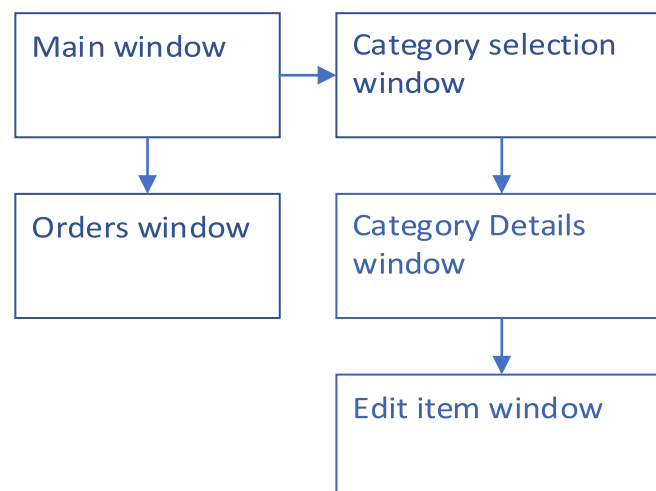
I'm going to use single table inheritance in my project so Parts table fields will include all the data for all classes in hierarchy (NPart, NWiredPart, NWirelessPart and NWiredWirelessPart) there would be an extra field with a class name (ClassName) to restore table data according to inheritance.



## Applications

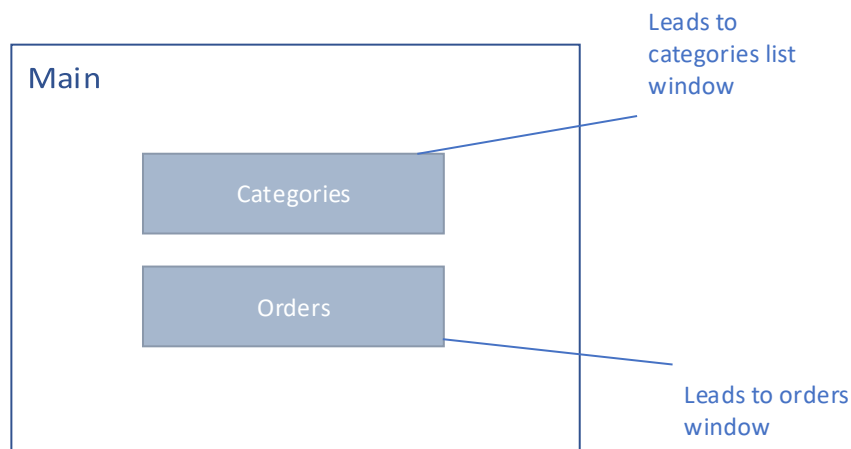
Windows WPF Client

Navigation



## Main window

You can navigate to categories window or to orders window from main form. Clicking on both buttons will open a new window.



## Orders window

From orders form you can use context menu to delete current selected order. A confirmation message will appear before an actual delete process. Upon receiving the confirmation application will call API to delete the order.

Orders						
OrderID	Order Date	Client name	Client e-mail	Item name	Qty	Price
9	11/11/2020	Alex Coal	ac@gmail.com	Sample Rourer	2	140NZD
10	11/07/2020	Mike Smith	ms@gmail.com	Sample NIC	1	30NZD

Total value of orders: 170NZD

Close

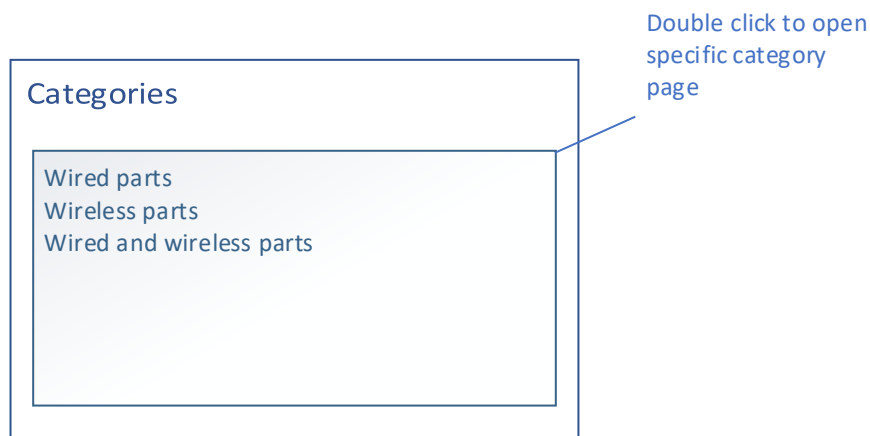
Right click to call menu:

- Delete Item

On delete click: Confirmation message box "Are you sure you want to delete the order #[OrderID] ?"

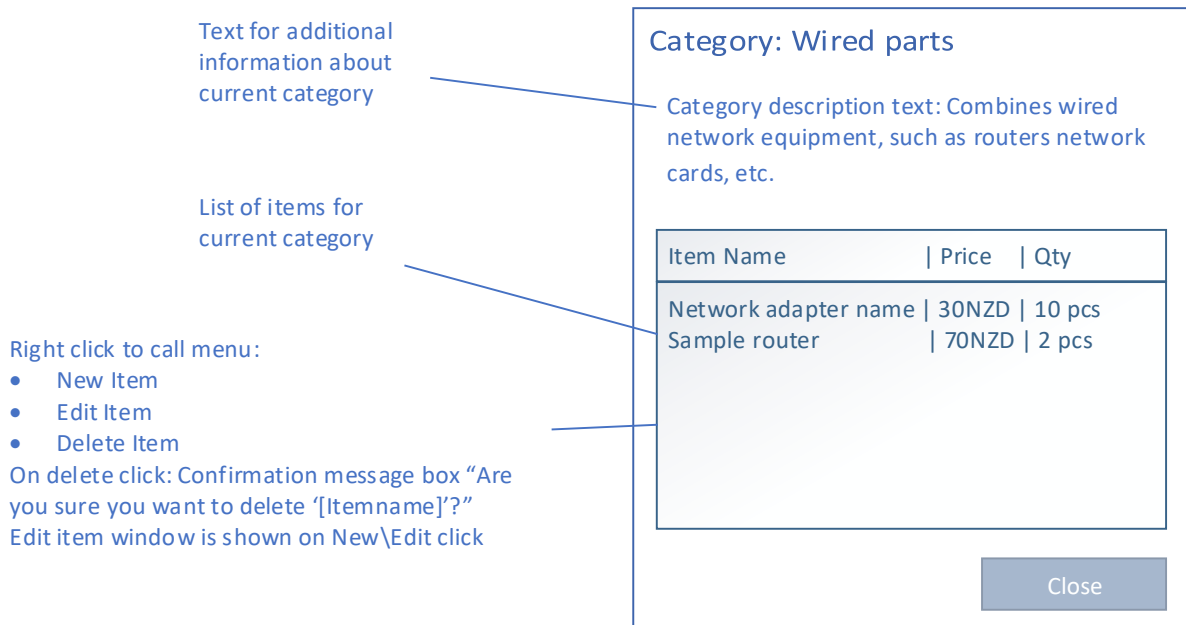
### Category selection window

Shows an ordered list of categories. Clicking on category will lead to open of category details window.



### Category details window

Shows a sorted list of items, belong to current category. The list is having a context menu to create\edit\delete an item. A confirmation is requested on delete menu item click. Clicking on edit\new menu item will lead to open of Parts base window (edit item window).



## Parts base window (edit item window)

Allow to edit information about an item, window layout depends on item type so some extra fields can appear and some fields may be missing, current layout corresponds to an item from “wired parts” category. Save button click will lead to validation process, then to the API call for create\update the item. Cancel button click will lead to confirmation message shows up, either user wants to quit without saving or not.

Edit item: Sample router

Last modified: 11/11/2020 13:04

Category: Wired parts

Item name:  
Sample router

Wired standard:  
10\100\1000

Currency name:  
NZD

Number of ports:  
5

In stock qty:  
30

Price:  
30,95

Save

Cancel

This content part of the window can vary, depending on actual class type we currently editing

### Validate data:

- Qty should be int
- Price should be float
- Number of ports should be int

If validation failed: Show message “Wrong field format in [Fieldname]”

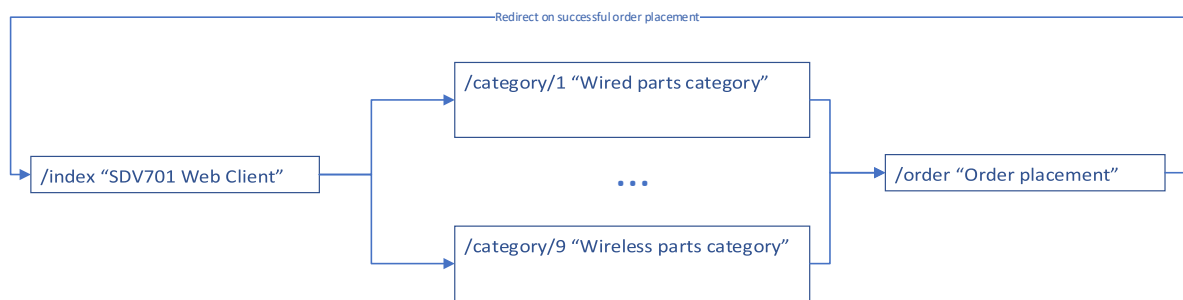
If data is validated: Perform API update\create request

If API request successful: Update item list for category

Ask for confirmation before exit editing.

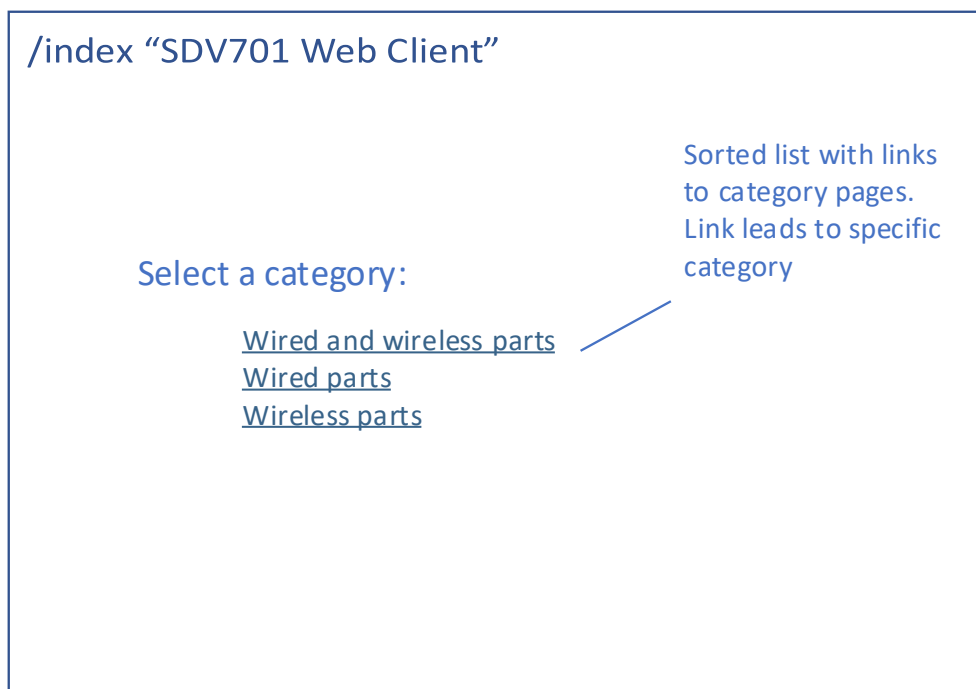
## Web Client

### Navigation



### Index page

Displays an ordered list of categories. Clicking on category will lead to open of a category details page.





## Category detail page

Displays a sorted list of items, which belongs to current category with additional info like price, in stock quantity color, item specific fields. Following “Order now” link will lead user to an “Order placement” page

/category/1 “Wired parts category”

Wired parts

[Category description text]

Text for additional information about current category

Extra parameter for current category

Item Name	Price	In Stock	Number of ports	Wired standard	Order
Router sample	130NZD		6	10/100/1000	<a href="#">Order now</a>
Sample NIC	30NZD		1	10/100	<a href="#">Order now</a>

Sorted by name

Color will depend on amount left in stock

Link to order item page for an item in current line

## Order placement page

Will allow user to input contact details, specify quantity if required and check item name and final price. Place order button will lead to data validation process followed by checking in stock quantity. On successful completion user will see an "Order confirmed message". If the process fails user will be notified.

/order "Order placement"

Order

Item Name	Price	Number of ports	Wired standard	Quantity
Router sample	130NZD	6	10/100/1000	[EDITABLE]

User input name

Your name:  
[Name input field]

Your e-mail:  
[E-mail input field]

User input mail

Total order value:130NZD

Place the order

User input quantity, 1 by default

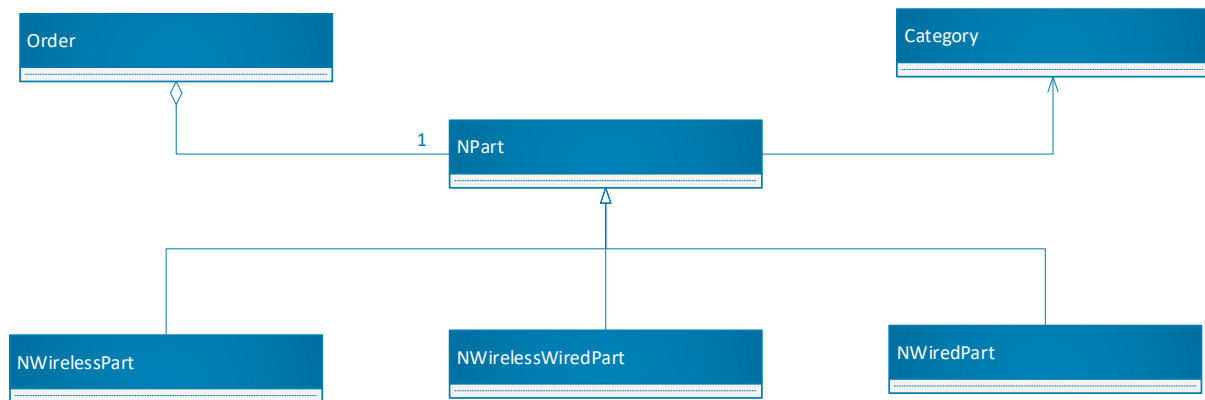
Should be recalculated upon quantity changes

- Validate quantity and email. Highlight field red in case validation not passed.
- Check available quantity in stock(according last obtained info from DB), inform user with popup if there is not enough items in stock.
- Call stored procedure to place order (check in stock quantity again just before inserting a record)
- Display a result of stored procedure, either order is placed or not with popup.
- If order is placed redirect to index page.

## UML

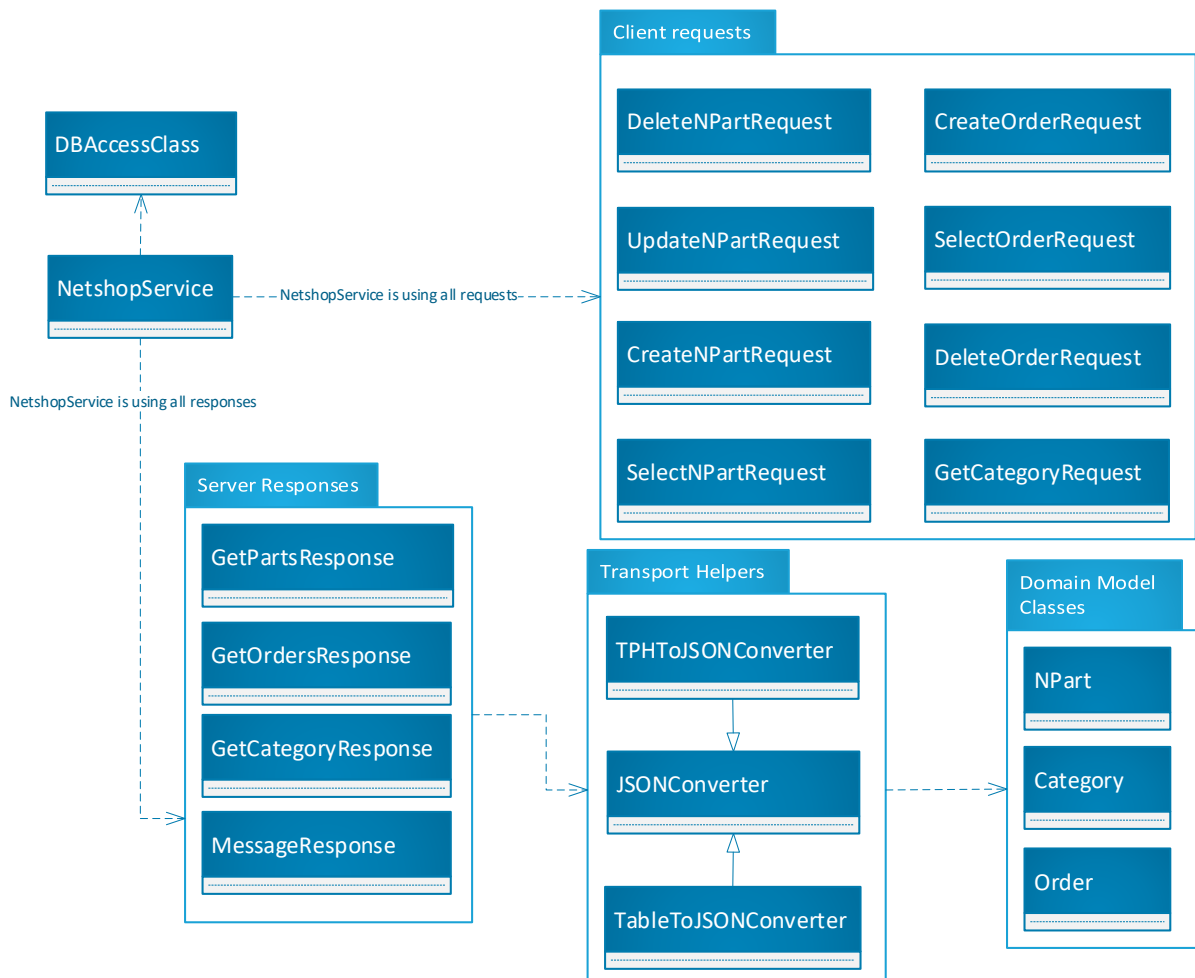
### Domain model classes

Classes are placed in shared assembly. So, all applications can access it.



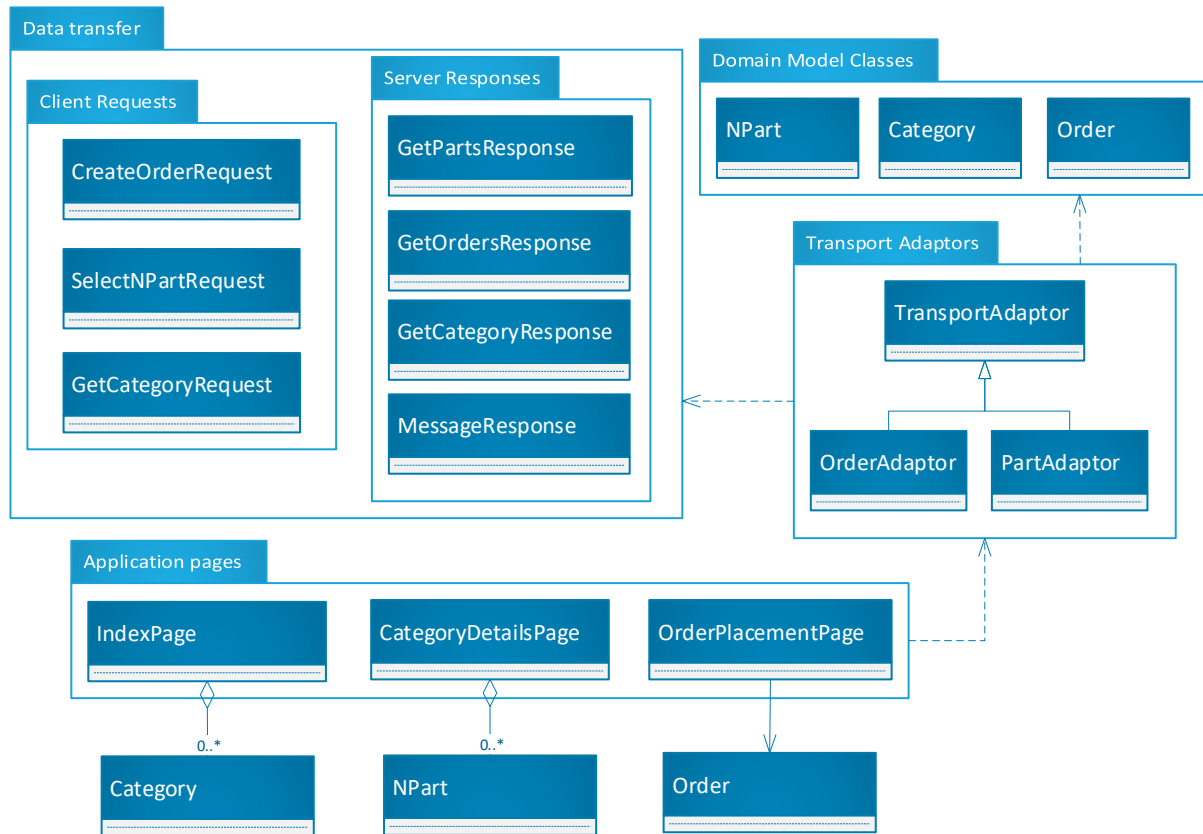
### Server

For each situation server is having gRPC response, request and NetshopService method to handle request and responses transport. Each request leads to generating response either containing a serialized set of domain model classes or a message response. Requests and response are serving as data transfer objects, domain model classes are used to unify packing and unpacking process.



## Web client

Client requests and server responses are DTOs, which carrying requested and responded data, if system requires converting of domain model class into DTO Transport adaptors are used in that case. Adaptors are also used to generate requests for create, update, delete operations.



## Windows WPF client

Is using the same data transfer technic as web client, but set of requests is different.

As client is based on WPF we are using strategy pattern to replace parts of window to cover inheritance representation while displaying items in “PartsBaseWindow”, as visual inheritance cannot be done in WPF the same way as in Windows forms applications.

