

Report Final Project

Nguyen Tuan Dung 20184246

ITSS | 4/1/2021

Appendix

1. Use case specification
2. Use case analysis
3. GUI design
4. Apply principle SOLID and pattern
5. Detail design
6. Use case specification

Use case specification

Use case “See list station details”

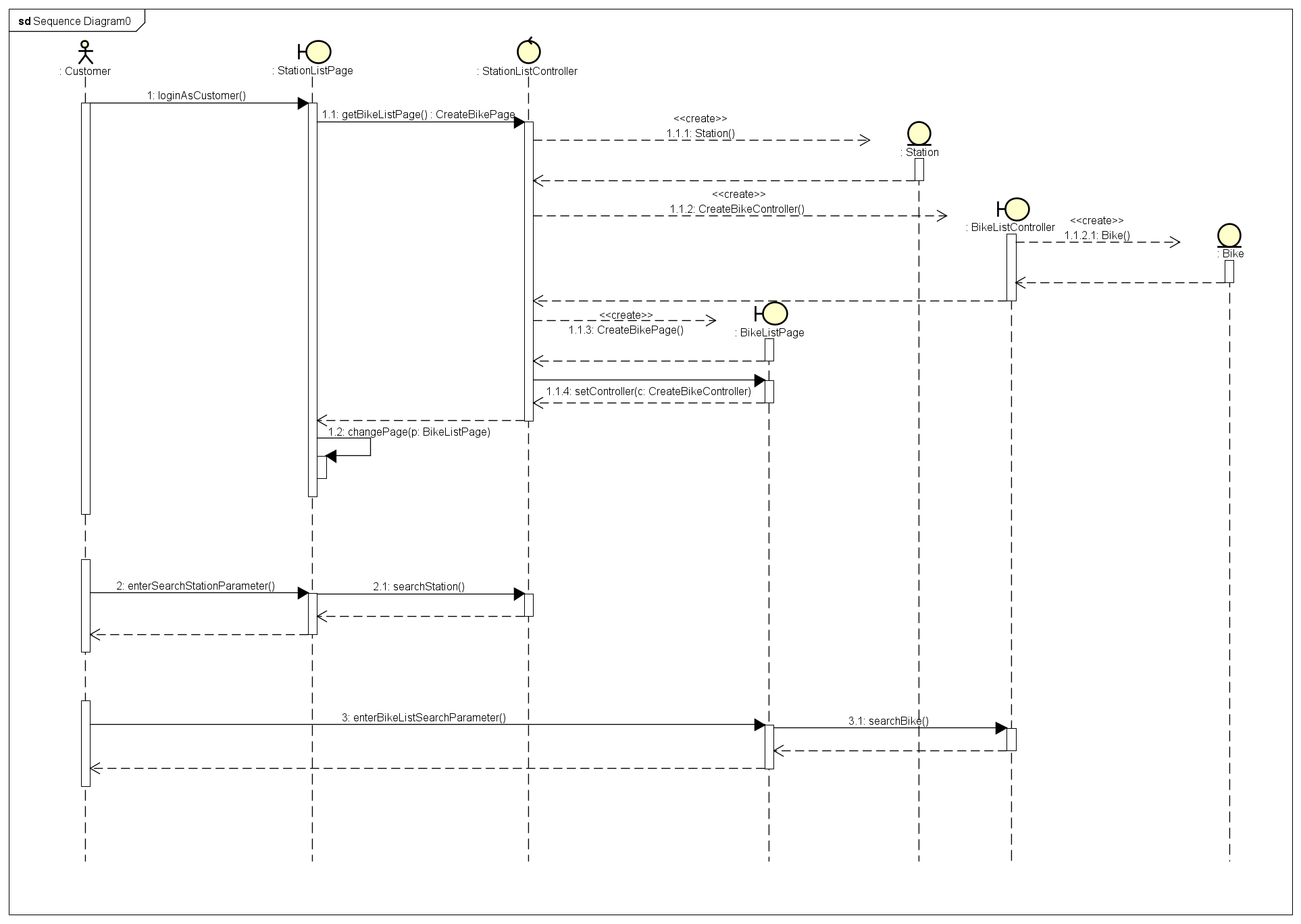
|  |  |  |  |
| --- | --- | --- | --- |
| **Use case Code** | UC003 | **Use case name** | See list station details |
| **Actor** | customer | | |
| **Precondition** | Login success as customer | | |
| **Main flow of event**  **(success)** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | |  | System | Display a list of stations: name, address, code, the number of bikes available | |  | System | Display list station page. Giving a text field for typing in | |  | Customer | Select a station | |  | System | Display the bike’s information: id, name, type, license plate | |  | Customer | Select a bike | |  | System | Display: Station name, list of bikes with several details | |  | System | Display the confirm box | |  | Customer | Press confirms | | | |
| **Alternative flow of event** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | | 3a | Customer | Type in search field | | 3b | Customer | Press “go” | | 3c | System | Back to #2 | | 5a | Customer | Type in search field | | 5b | Customer | Press “go” | | 5c | System | Back to #4 | |  | | | | | |
| **Post condition** | No | | |

\* Input data:

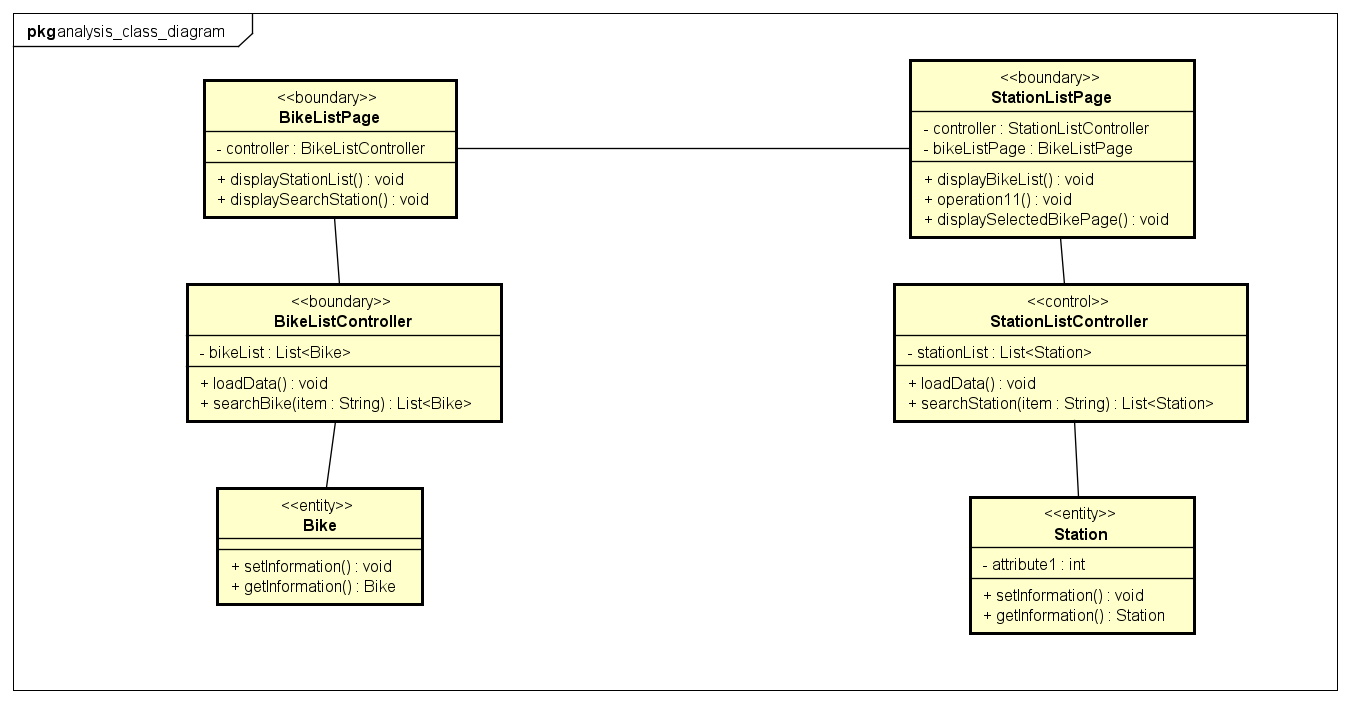
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Field** | **Description** | **Required?** | **Valid condition** | **Example** |
|  | Station search field | No | No | No | Abcd1234 |
|  | Bike search field | No | No | No | Abcd1234 |

1. Use case analysis

Sequence analysis diagram



Class analysis diagram



1. GUI design

# Standardizing the screen configuration

## Display

Resolution: 900 x 600 px

## Screen

Position of button: bottom (vertical) and center (horizontal) of frame.

Position of message: center of frame

Position of screen title: Title top-left of frame.

Numeric display consistency: commas to separate thousands, and strings consisting of only characters, digits, commas, periods, spaces, underscores, and hyphen symbols.

## Control

Size text: medium size (15px). Font: system UI. Color: #000000

Check input: check empty and format.

## Enter input from keyboard

No keyboard shortcuts. Using button to return previous screen. Otherwise button “X” in top-right of window to close screen.

## Error

Show message by label warnning.

# Create screen images

Ảnh có chứa bàn

Mô tả được tạo tự độngStation List

Ảnh có chứa bàn

Mô tả được tạo tự động

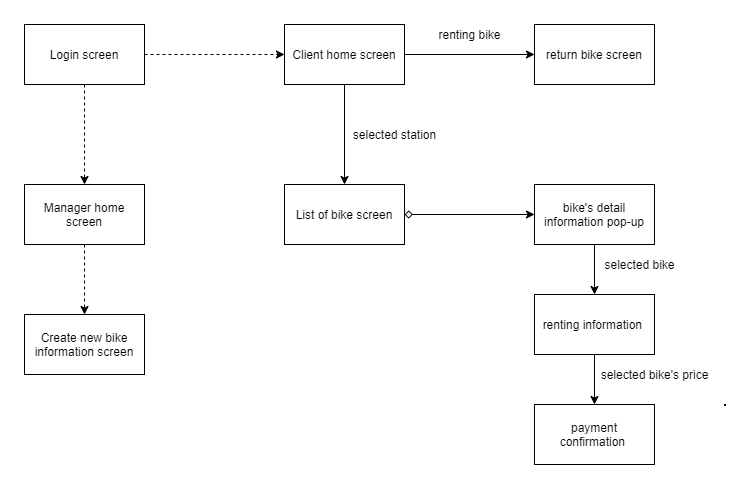
Bike list

Ảnh có chứa văn bản

Mô tả được tạo tự động

Renting bike confirmation

# Create a screen transition diagram



# Creating screen specification

1. List of station screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
| Ảnh có chứa bàn  Mô tả được tạo tự động | Labels and text field | Initial | Display stations information |
| Back | Click | Go for more details |
| Go | Click | Search for typed information |
| Reset | Click | Clear Discover text field |

1. List of bike screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
| Ảnh có chứa bàn  Mô tả được tạo tự động | Labels and text field | Initial | Display bikes information |
| Back | Click | Pop-up to bike confirmation tab |
| Go | Click | Search for typed information |
| Reset | Click | Clear Discover text field |

1. Bike’s information pop-up

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
| Ảnh có chứa văn bản  Mô tả được tạo tự động | Confirm Button | Click | Display renting information screen |
| Cancel Button | Click | Close the pop-up |

1. Apply principle SOLID and pattern
2. Principle SOLID

* Single-responsiblity principle: each class has only one responsibility, such as: stationApi to connect, interact with database (insert, update, get data, delete), Station model for hold data.
* Open-closed principle: we have class BaseApi, if in the future, there is new object want to connect database, it will create a new class inherit BaseApi. Therefor, it will be open to add new feature in new class and close for modify BaseApi.
* Liskov substitution principle
* Interface segregation principle
* Dependency Inversion Principle

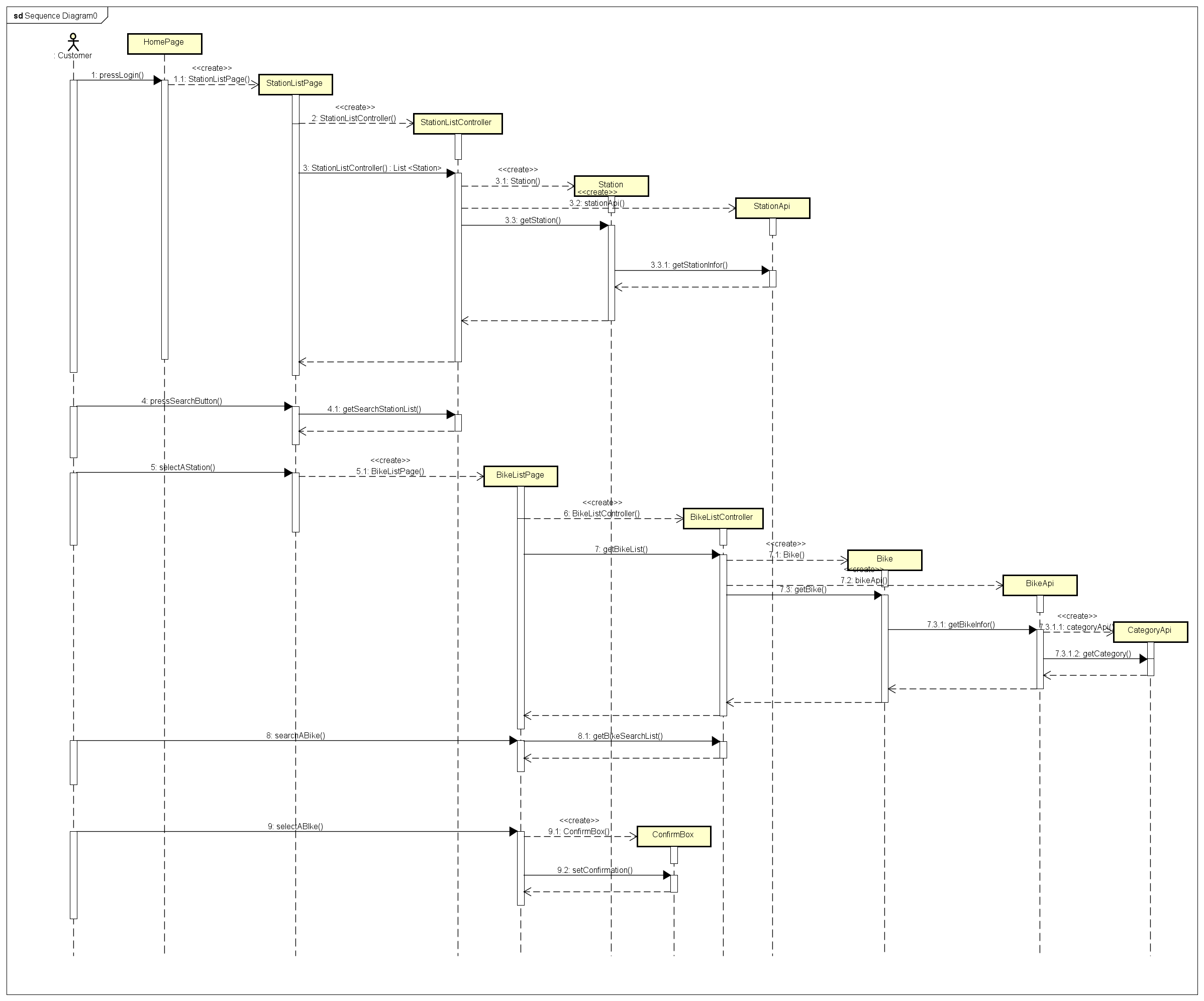
We using interface between subsystem and application. Each subsystem has own interface corresponding.

1. Pattern:

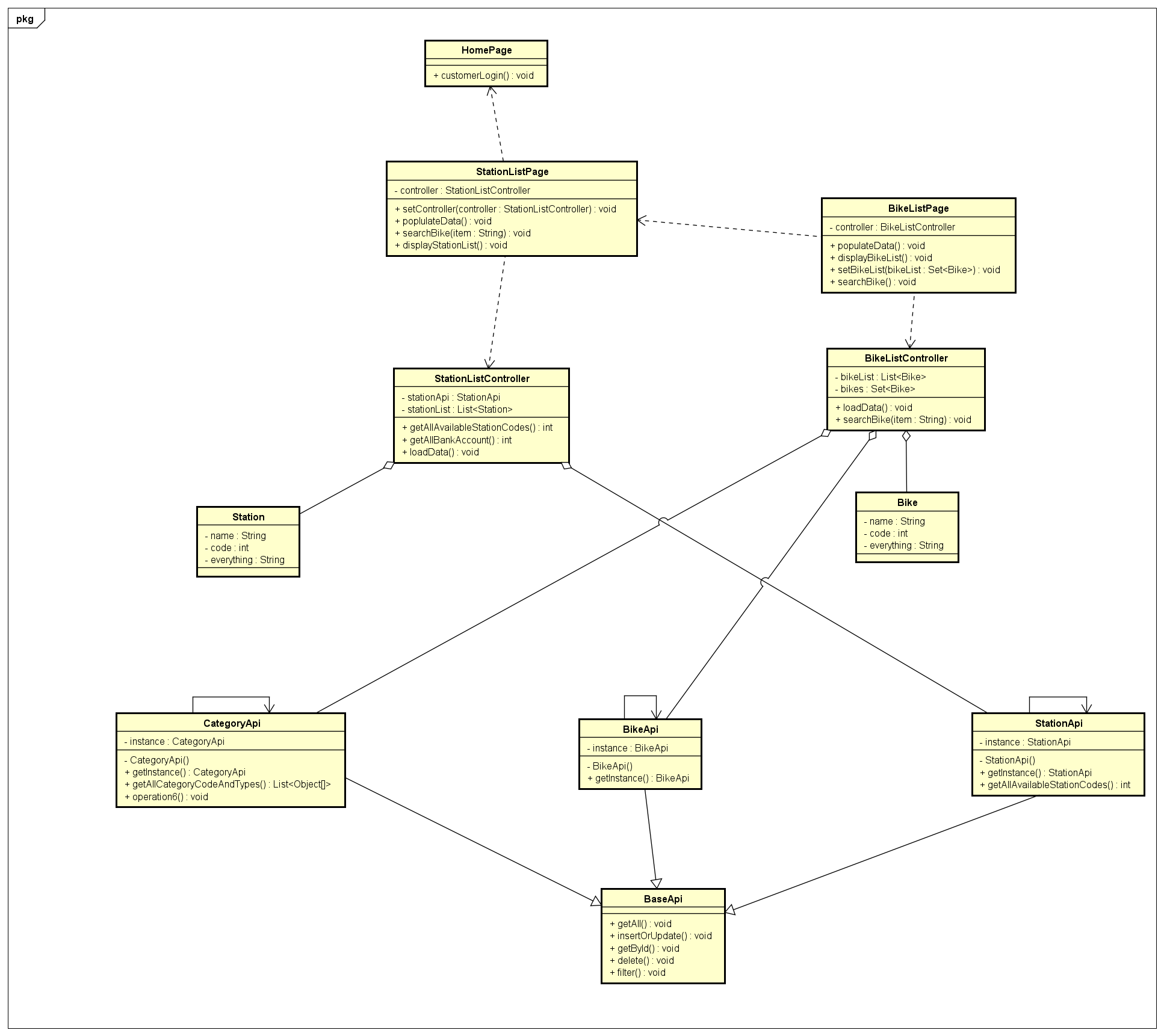
* MVC: For example: we have bill screen, billController, and billModel.
* Singleton pattern: init api
* Strategy pattern

1. Detail design

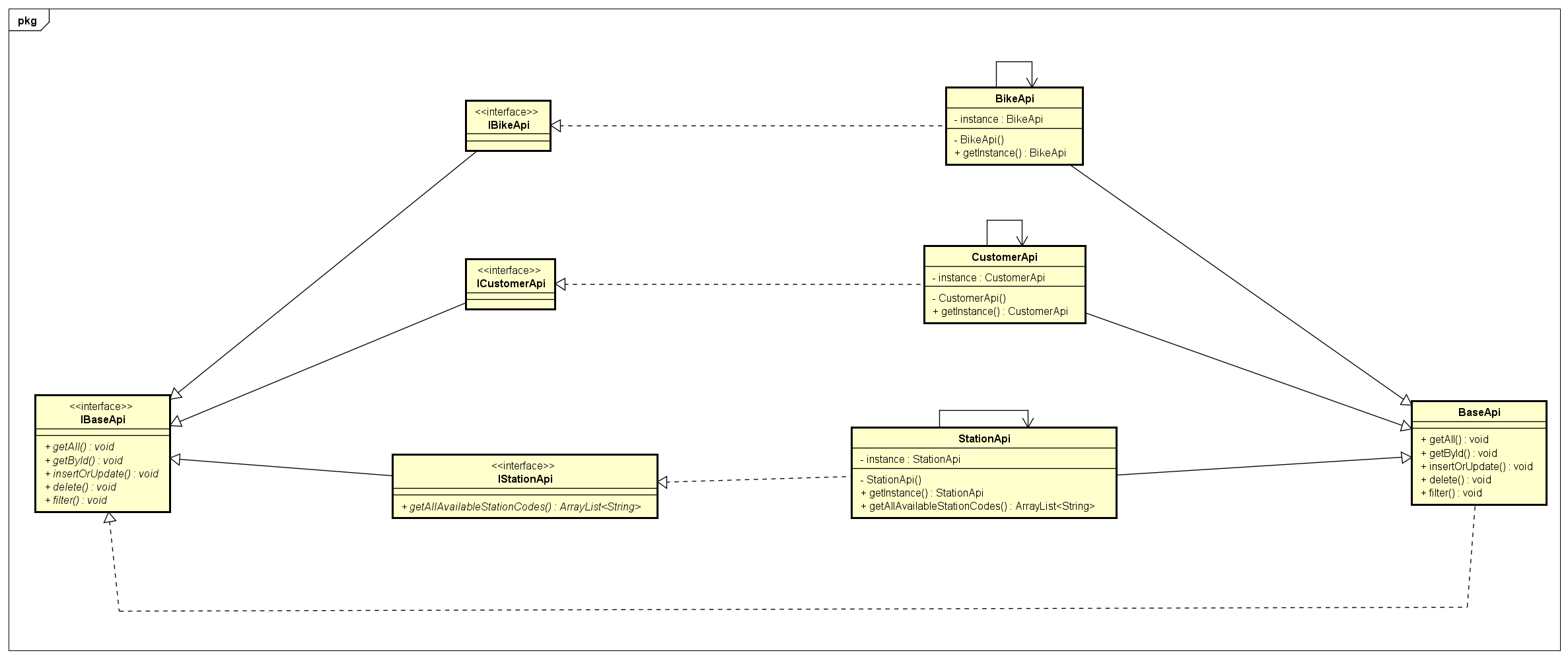
Sequence diagram: display station list and bike list



Class diagram:



Detail design subsystem



1. Unit test

In Test\_Usecase\_SeeStationlist.doc file