

Report Final Project

Group 02

Nguyễn Mạnh Tiến\_20184312

Nguyễn Phú Trường\_20184319

Nguyễn Tuấn Dũng\_20184246

Chu Mạnh Hải\_20184253

ITSSS | 25/12/2021

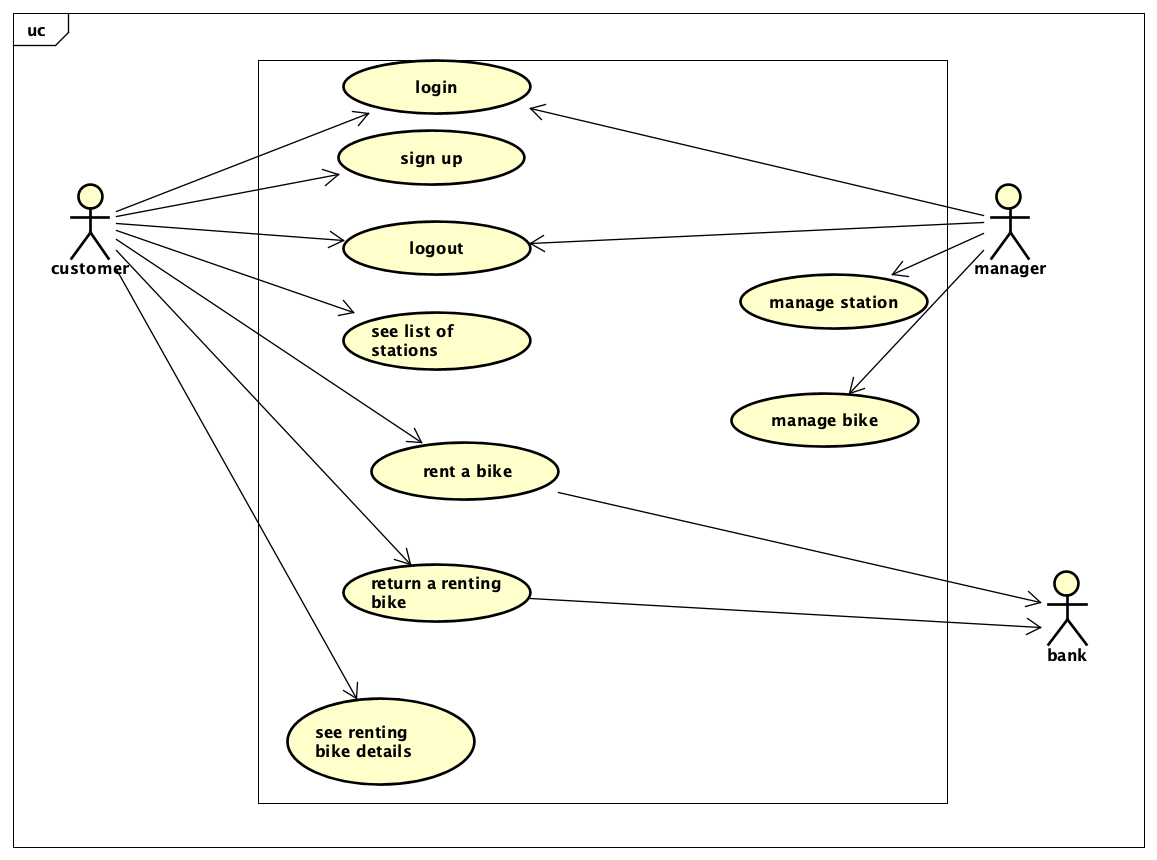
**Appendix**

1. Use case diagram
2. Use case specification
3. Use case analysis
4. GUI design
5. Apply principal SOLID and pattern
6. Detail design
7. Package organization
8. Database design
9. Glossary
10. Supplementary specification
11. **Use case diagram**

**1. General use case diagram**

Description of actors:

* Customer: interact with system, can rent bike, return rented bike, ...
* Manager: manage system (manage station, manage bike, ...)
* Bank: manipulate card (call API increase, decrease, check balance in bank account, ...)



Explanation: To rent a bike, first, customer must login application (if customer don’t have account, they must sign up). Then, they can see their current location on the map with location of bike station. They can directly choose the station on the map or search station to see station in detail. When see detail station, they can choose bike to see bike in detail and rent the bike. After using the bike, they can return bike in any station.

Besides customer, there are also application manager and bank to manage station, bike and manipulate card corresponding.

Composite use case:

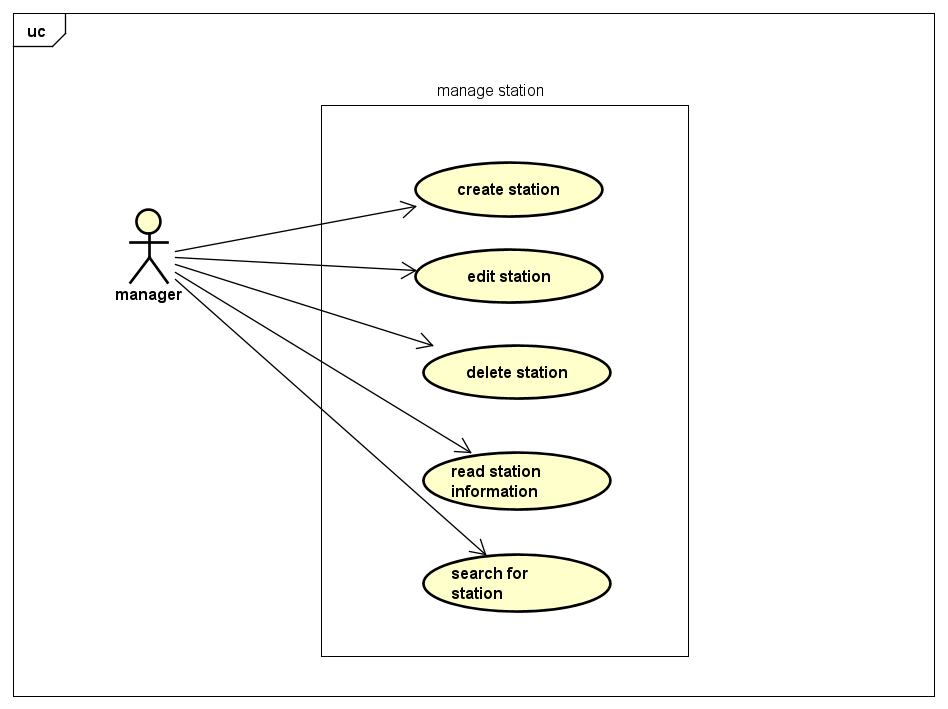
1. See list of stations
2. Manage station
3. Manage bike

**2. Use case diagram for “See list of stations”**

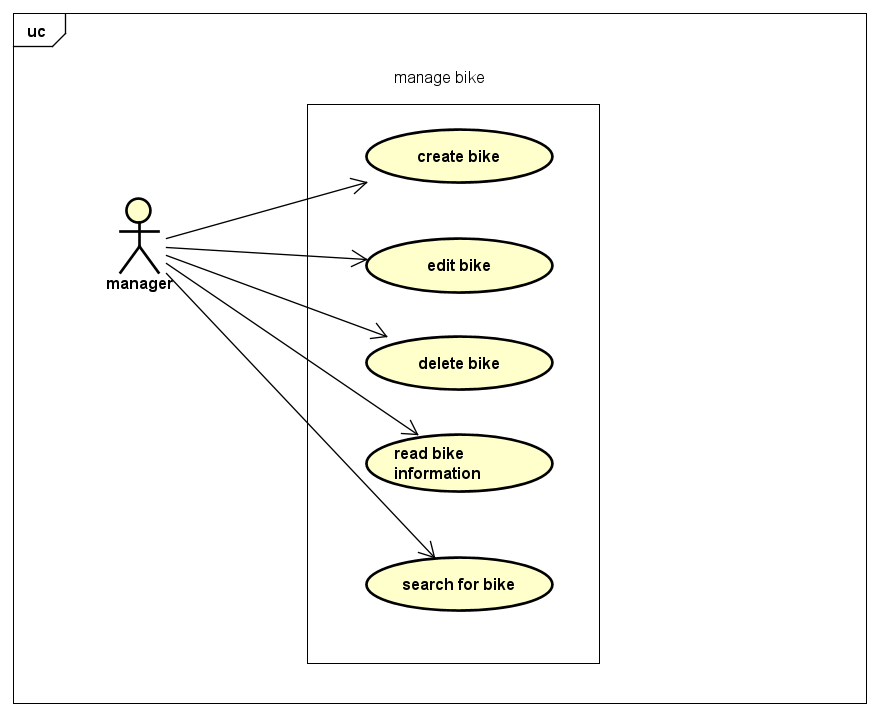
Diagram

Description automatically generated

**3.** **Use case diagram for “Manage station”**



**4. Use case diagram for “Manage bike”**



1. **Use case specification**
   1. **Use case “Rent bike” (Nguyễn Mạnh Tiến)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case Code** | UC001 | **Use case name** | rent bike |
| **Actor** | customer | | |
| **Precondition** | Login success | | |
| **Main flow of event**  **(success)** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | |  | Customer | Choose rent bike function | |  | System | Display rent bike function interface | |  | Customer | Input Bike Code | |  | Customer | Choose bike request | |  | System | Check bike code valid or not? | |  | System | Display the bike information | |  | Customer | Confirm to choose the bike or not? | |  | System | Display bill in detail | |  | Customer | Confirm to rent bike or not? | |  | System | Display chose bank account interface | |  | Customer | Input bank account or choose existed account in the system | |  | System | Validate bank account | |  | Customer | Confirm pay or not? | |  | System | Call use case “pay for service” | | | |
| **Alternative flow of event** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | | 5a | System | Error: Invalid bike code | | 7a | System | Display rent bike screen | | 9a | System | Display rent bike screen | | 12a | System | Error: Invalid bank account | | 13a | System | Display rent bike screen | |  | | | | | |
| **Post condition** | Save transaction, send email to customer | | |

\* Input data:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Field** | **Description** | **Required?** | **Valid condition** | **Example** |
|  | Bike code |  | Yes | Not null or empty | B12345 |
|  | Card number |  | yes | * Not null or empty * Contain <=20 digit | 123456789 |
|  | Card holder name |  | yes | Not null or empty | Nguyen Van A |
|  | Issues Bank |  | yes | Not null or empty | Viettin Bank |
|  | Expiration date |  | yes | Not null or empty, date format | 20/02/2030 |
|  | Security code |  | yes | Not null or empty | 123456 |
|  | Transaction description |  | no |  |  |

* 1. **Use case “create bike” (Chu Mạnh Hải)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case Code** | UC002 | **Use case name** | Create bike’s information |
| **Actor** | manager | | |
| **Precondition** | Already login as manager role | | |
| **Main flow of event**  **(success)** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | |  | manager | Login successfully | |  | system | Show manager home screen | |  | manager | Click add “create new bike” button | |  | system | Show a form to add a new bike’s information | |  | manager | Fill in bike’s information | |  | manager | Click on submit button | |  | system | Validate information | |  | system | Notify a success message | |  | system | Redirect to list of bikes screen | | | |
| **Alternative flow of event** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | | 1a. | system | Ask user to login if he/she has not already login as manager role | | 6a. | system | If user click on “back” button, redirect to manager home screen | | 6b. | system | If user click on “exit” button, redirect to login screen | | 6c. | system | If form is not filled in completely, submit button will be disable | | 8a. | system | If information is invalid, show error message and require to re-enter information of invalid fields | | | |
| **Post condition** | No | | |

\* Input data:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Field** | **Description** | **Required?** | **Valid condition** | **Example** |
| 1. | name | Name of bike | yes | Not null/empty | Mount-S |
| 2. | manufacturer | Name of manufacturer | yes | Not null/empty | Kona Bikes |
| 3. | category | Type of bike | yes | Not null/empty | E-bike |
| 4. | weight | Weight of bike (kg) | yes | Positive Number | 34 |
| 5. | cost | Cost of bike ($) | yes | Positive number | 200 |
| 6. | station code | Station code | yes | Positive integer | 2 |
| 7. | Manufacturer date | Date of creation | yes | Date | 12-1-2021 |

* 1. **Use case “See list station details” (Nguyễn Tuấn Dũng)**

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 “Return a renting bike” (Nguyễn Phú Trường)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case Code** | UC004 | **Use case name** | Return a renting bike |
| **Actor** | Customer | | |
| **Precondition** | Login success | | |
| **Main flow of event**  **(success)** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | |  | Customer | Choose Return a renting bike function | |  | System | Calculate bike rental time | |  | System | Display Return a renting bike function interface | |  | Customer | Choose bank account | |  | Customer | Choose bike station code | |  | Customer | Click on “Return” button | |  | Customer | Click on “Confirm” button to finish renting transaction | |  | System | Return the remaining deposit to the bank account | |  | System | Notify a success message | | | |
| **Alternative flow of event** | |  |  |  | | --- | --- | --- | | **#** | **Doer** | **Action** | | 4a | System | If customer choose “Use other”, open a popup to add a new bank account | | 6a | System | If customer click on “Cancel” button, redirect to list of station screen | | 8a | System | If the total payment amount is greater than the deposit amount, pay the extra amount from the bank account | | | |
| **Post condition** | Save transaction, send email to customer | | |

\*Input data:

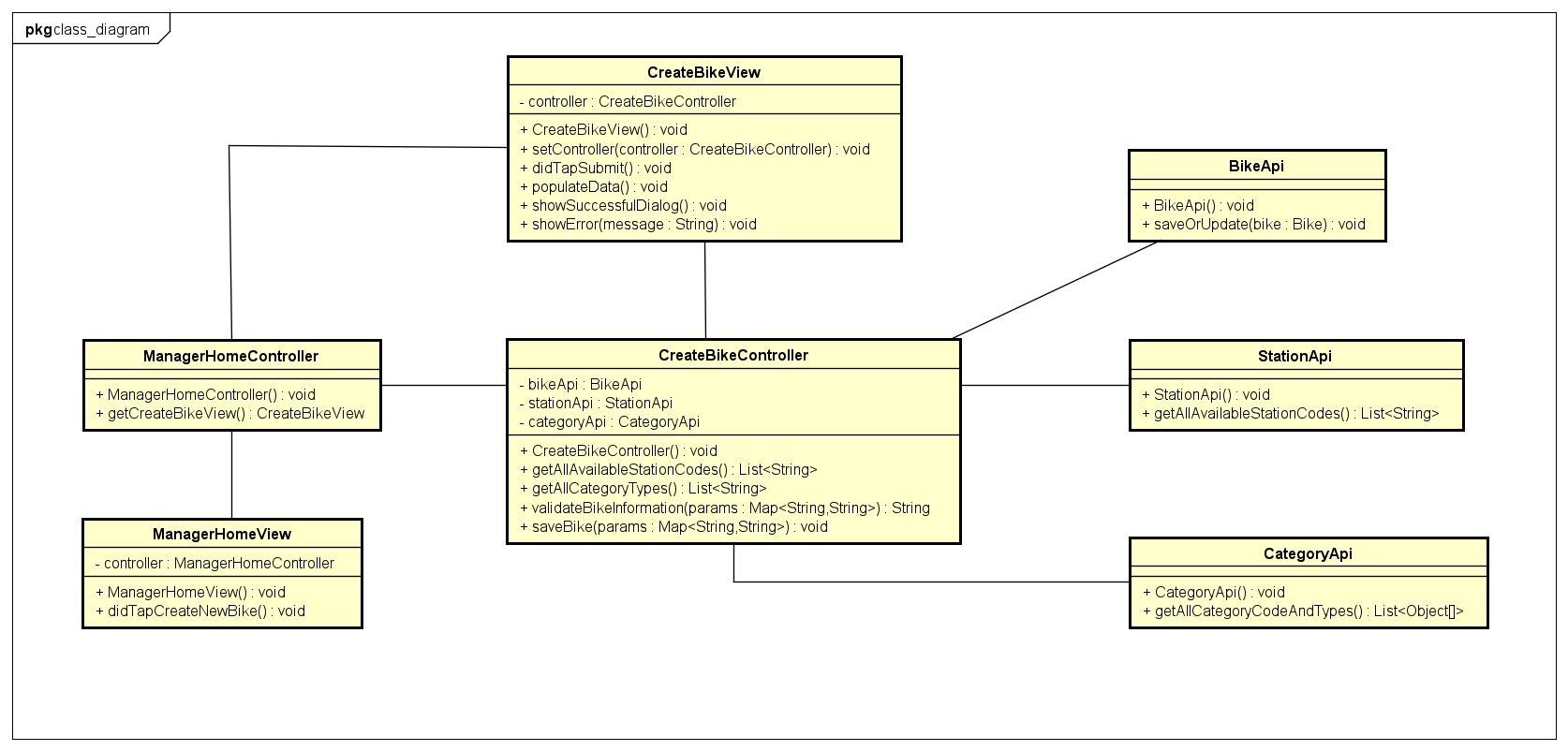
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Field** | **Description** | **Required?** | **Valid condition** | **Example** |
|  | Card number |  | yes | * Not null or empty * Contain <=20 digit | 123456789 |
|  | Card holder name |  | yes | Not null or empty | Nguyen Van A |
|  | Issues Bank |  | yes | Not null or empty | ViettinBank |
|  | Expiration date |  | yes | Not null or empty, date format | 20/02/2030 |
|  | Security code |  | yes | Not null or empty | 123456 |

1. **Use case analysis**
   1. **Create a new bike**
2. Sequence analysis diagram

A picture containing table

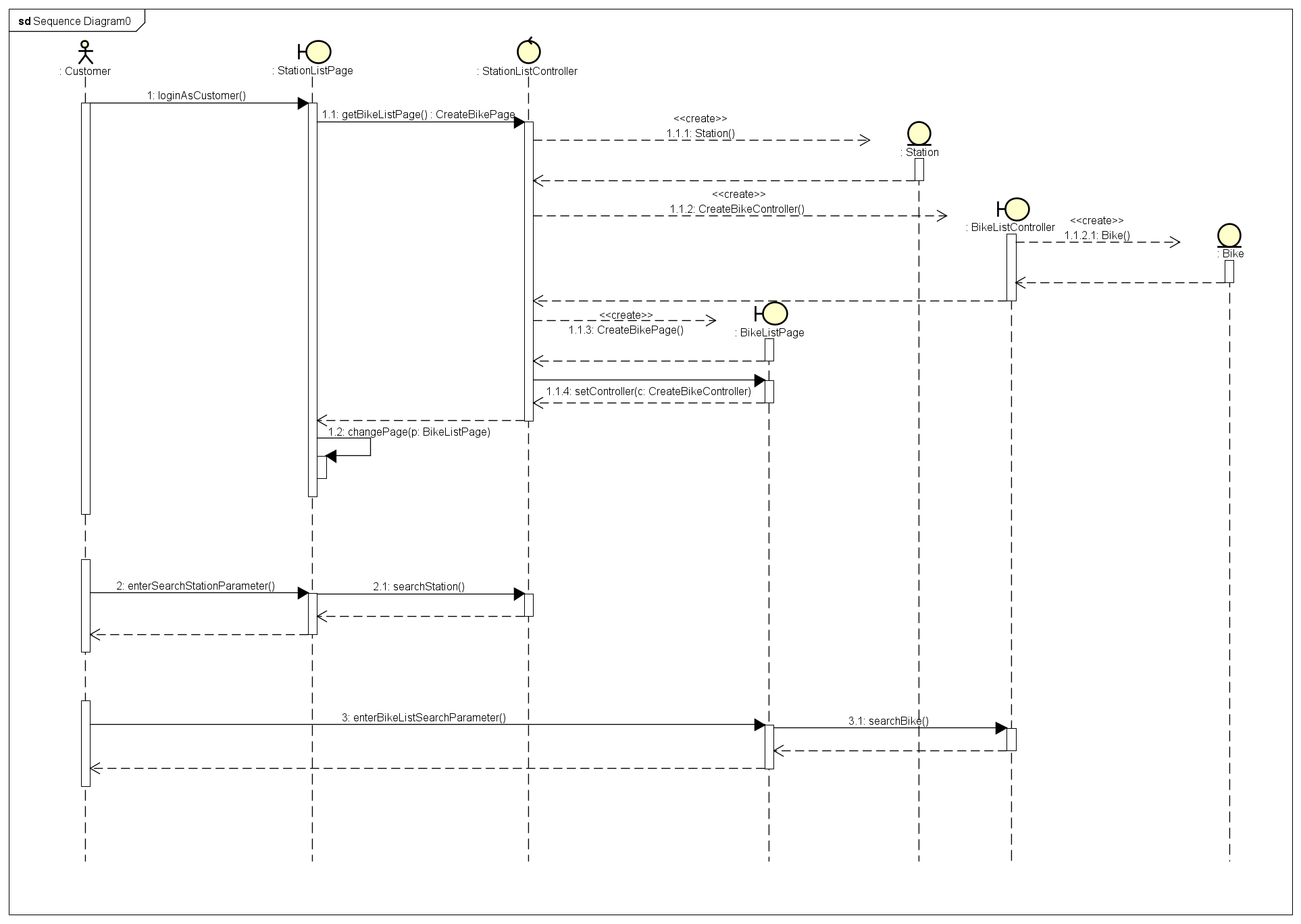
Description automatically generated

1. Class analysis diagram

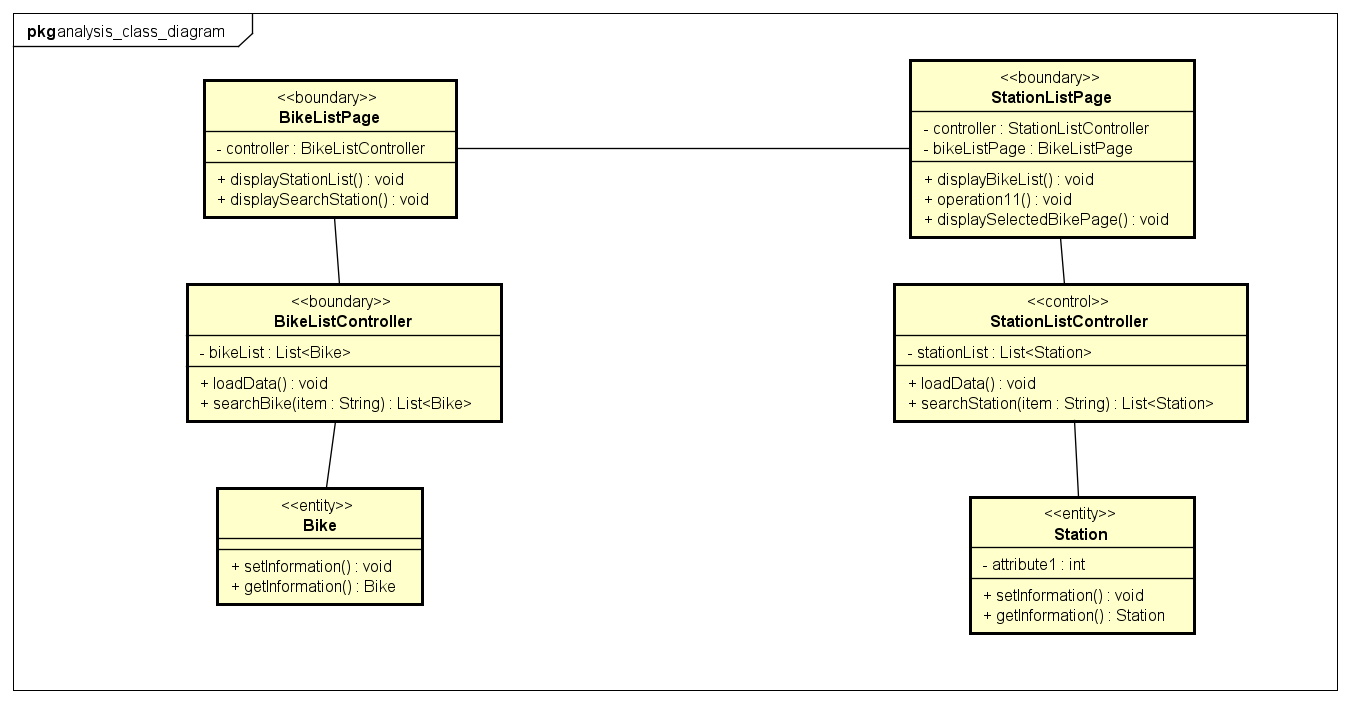


* 1. **See station list**

1. Sequence analysis diagram

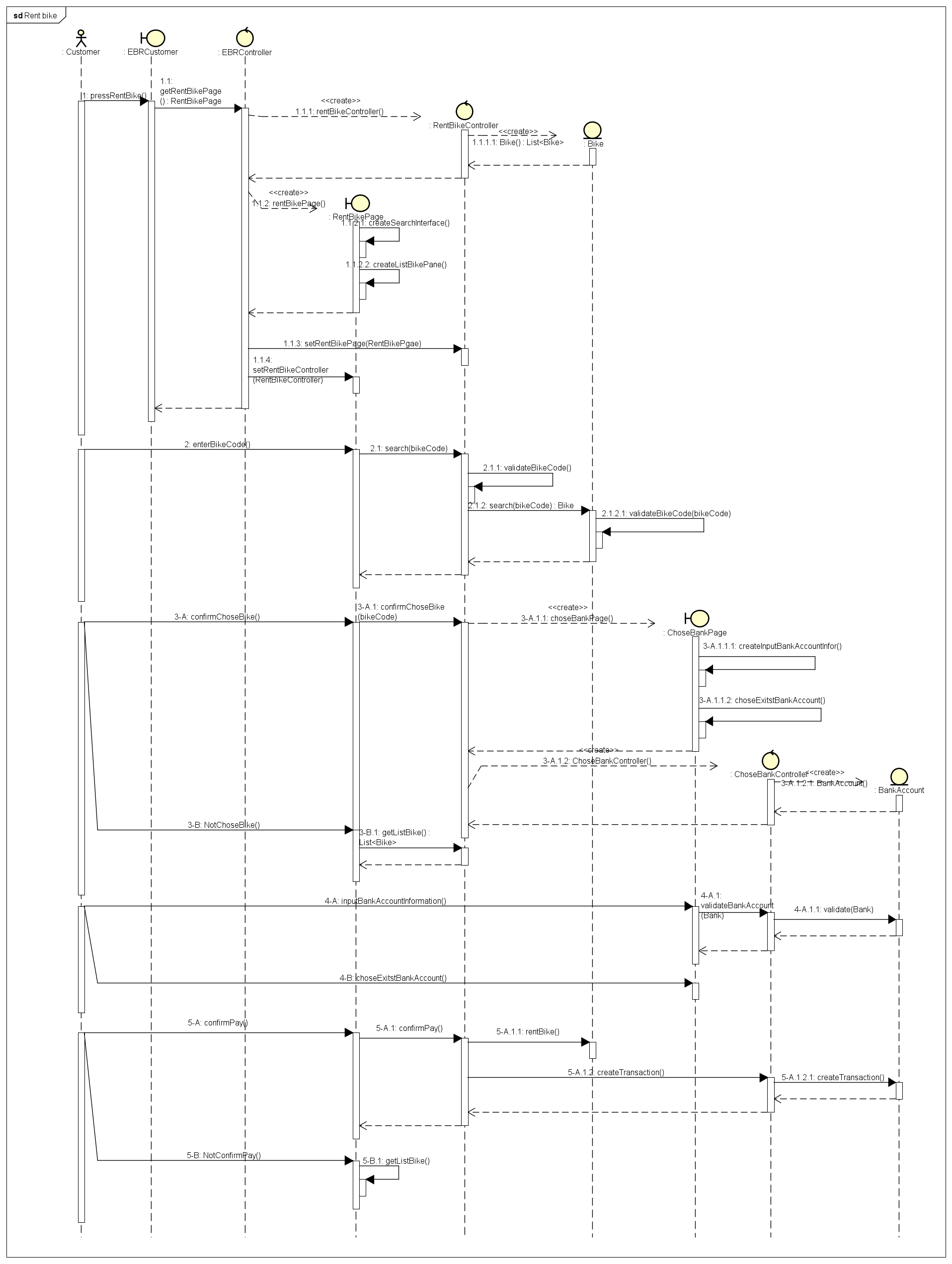


1. Class analysis diagram

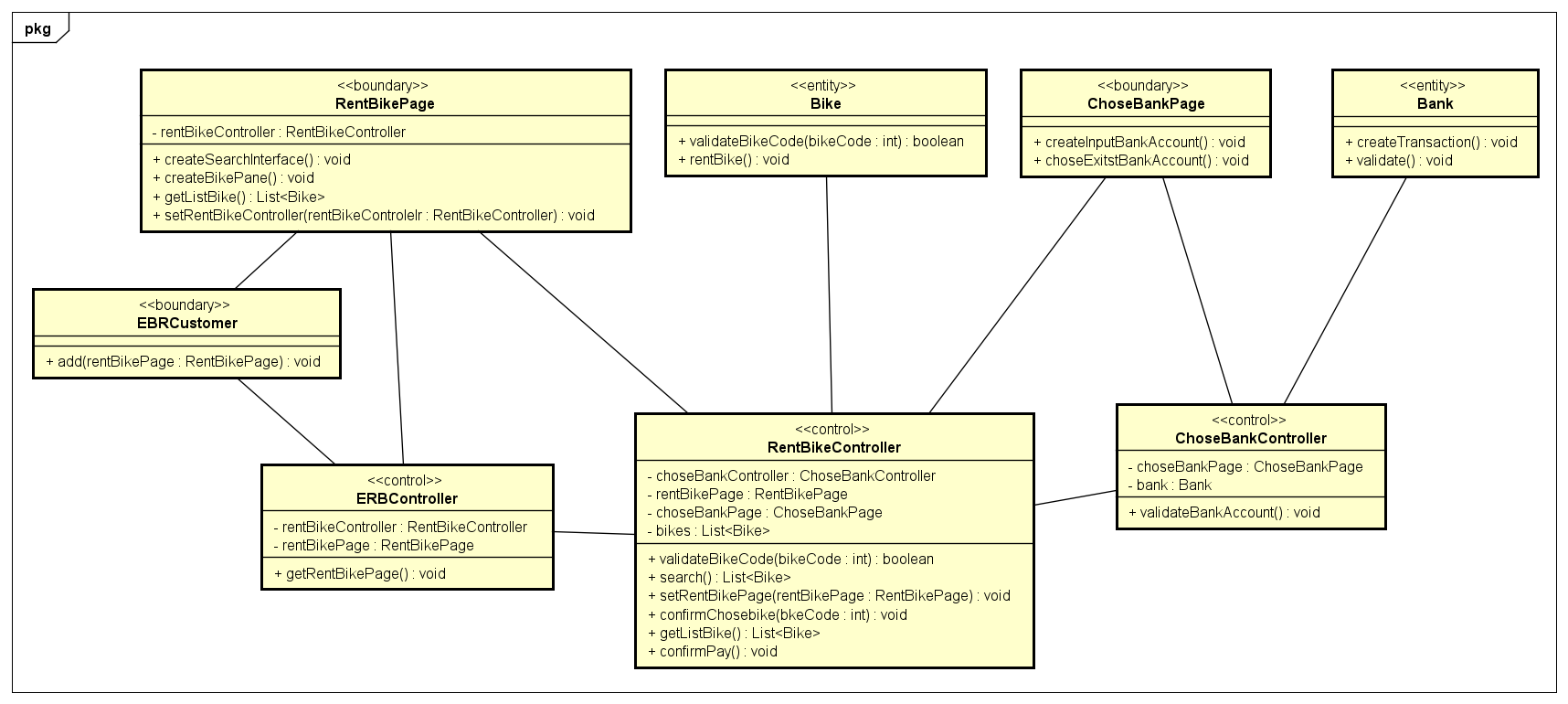


* 1. **Rent bike**

1. Sequence analysis diagram

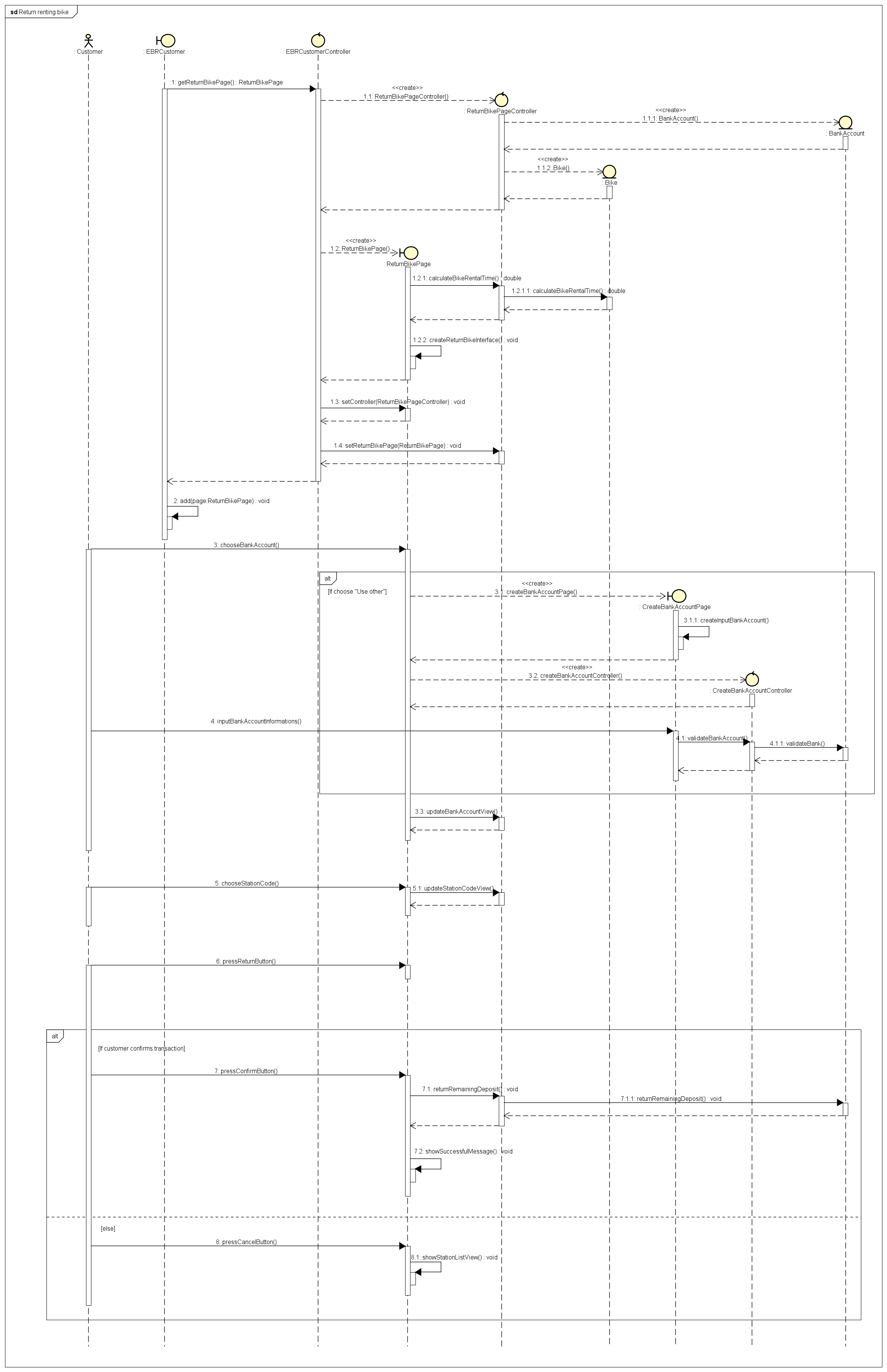


1. Class analysis diagram

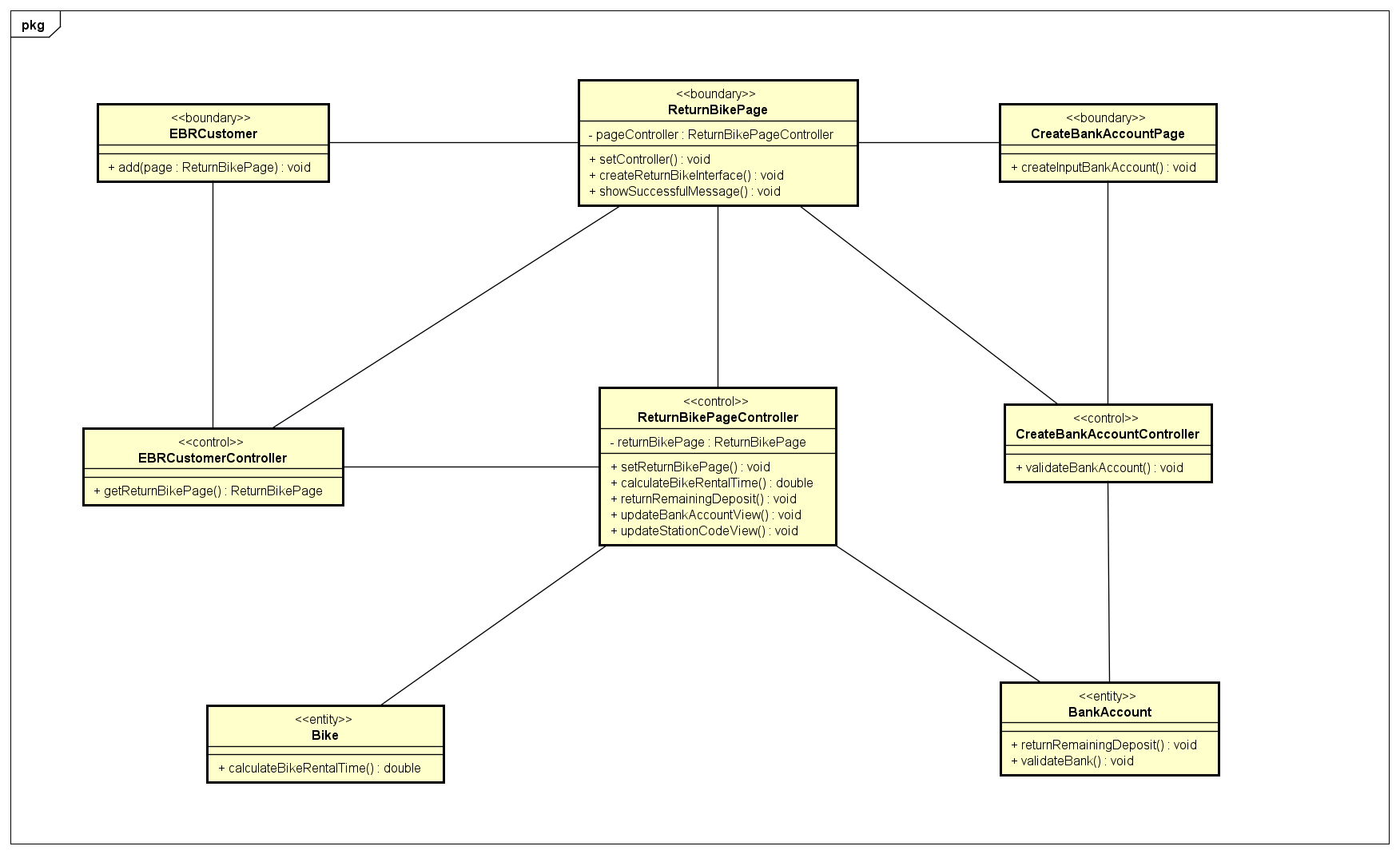


* 1. **Return renting bike**

1. Sequence analysis diagram



1. Class analysis diagram



1. **GUI design**

**1. 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.

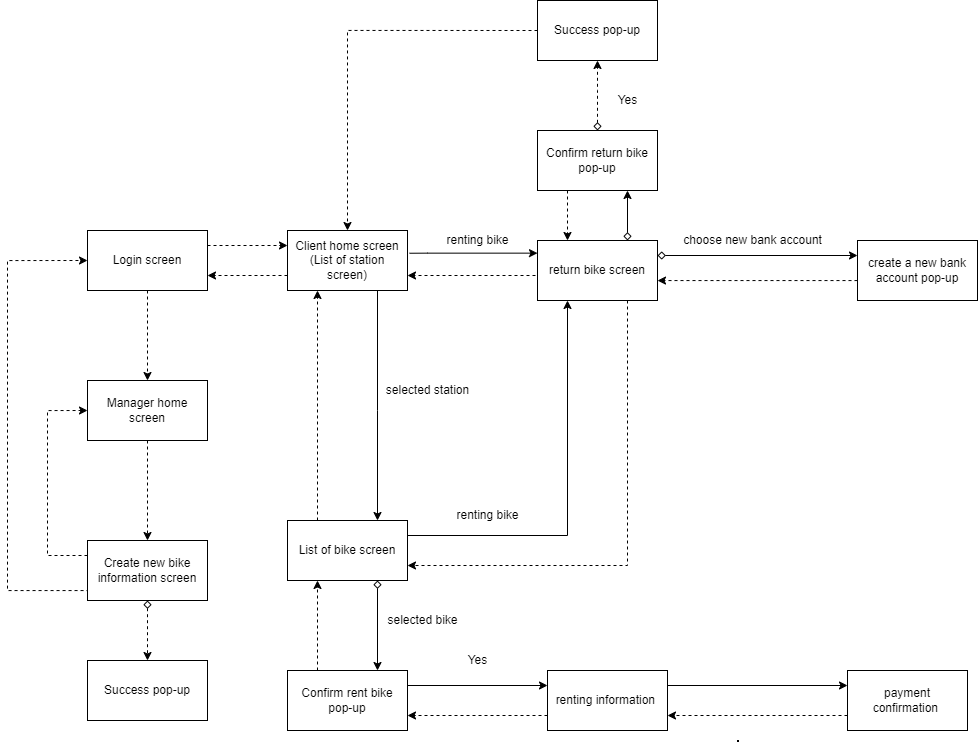
**Enter input from keyboard**

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 warning.

**2. Create a screen transition diagram**



**3. Creating screen specification**

1. Login screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
| Ảnh có chứa văn bản  Mô tả được tạo tự động | “Login As Customer” button | Click | Display client home screen |
| “Login As Manager” button | Click | Display manager home screen |

1. Manager home screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
| Graphical user interface, application  Description automatically generated | “Create new station” button | Click | Display a form to create new station |
| “Create new bike” button | Click | Display a form to create new bike |
| “exit” button | Click | Navigate back to login screen |

1. Create new bike information screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
| Graphical user interface  Description automatically generated | Area to display form | Initial | Display a form for user to create new bike information |
| “Submit” button | Click | Submit a form to handle new bike creation |
| “back” button | Click | Return to manager home screen |
| “exit” button | Click | Return to login screen |

1. List of station screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
|  | 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 |
|  | 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. Confirm rent bike pop-up

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
|  | Confirm Button | Click | Display renting information screen |
| Cancel Button | Click | Close the pop-up |

1. Renting information screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
| BillScreen | Label | Initial | Display the bill information |
| Cancel button | Click | Cancel rent bike and return list of bike screen |
| Confirm button | Click | Confirm the bill and transition to next screen |

1. Payment confirmation screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
|  | Label | Initial | Display the account information |
| Area text | Initial | Display the transaction description |
| Cancel button | Click | Cancel rent bike and return list of bike screen |
| Confirm button | Click | Confirm the account and pay |
| Radio Button | Click | Chose current bank account or new ones |

1. Return bike screen

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
|  | Labels | Initial | Display user and bill information |
| Bank account choice box | Choose options | Select existing bank accounts or create a new bank account |
| Station code choice box | Choose options | Select the station to return the bike |
| “Return” button | Click | Confirm the bill and pay |
| “Cancel” button | Click | Cancel return renting bike and return list of bike station screen |

1. Create a new bank account pop-up

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
|  | Text field | Enter information | Enter new bank account data |
| Issuing bank combo box | Choose options | Choose the issuing bank |
| Date picker | Choose options | Choose the expiration date |
| “Cancel” button | Click | Close the pop-up |
| “Add” button | Click | Create a new bank account and close the pop-up |

1. Confirm return bike pop-up

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
|  | “Yes” button | Click | Enter new bank account data |
| “No” button | Click | Choose the issuing bank |

1. Success Pop-up

|  |  |  |  |
| --- | --- | --- | --- |
| Screen image | Control | Operation | Function |
|  | “OK” button | Click | Return to list of bike station screen |

1. **Apply SOLID principle and pattern**
2. **SOLID Principles**

* Single-responsibility principle: each class has only one responsibility, such as: billApi to connect, interact with database (insert, update, get data, delete), Bill 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. Therefore, it will be open to add new feature in new class and close for modify BaseApi.  
  
* Liskov substitution principle: For example, every time we use BaseApi to get bike’s information, it could be substituted by BikeApi, which does the same functionality
* Interface segregation principle: We divide subsystems into many smaller interfaces, each one will be responsible for each purpose. Such as IStationApi interface is responsible for station’s information manipulation  
  Text

  Description automatically generated with medium confidence
* Dependency Inversion Principle

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




1. **Pattern:**

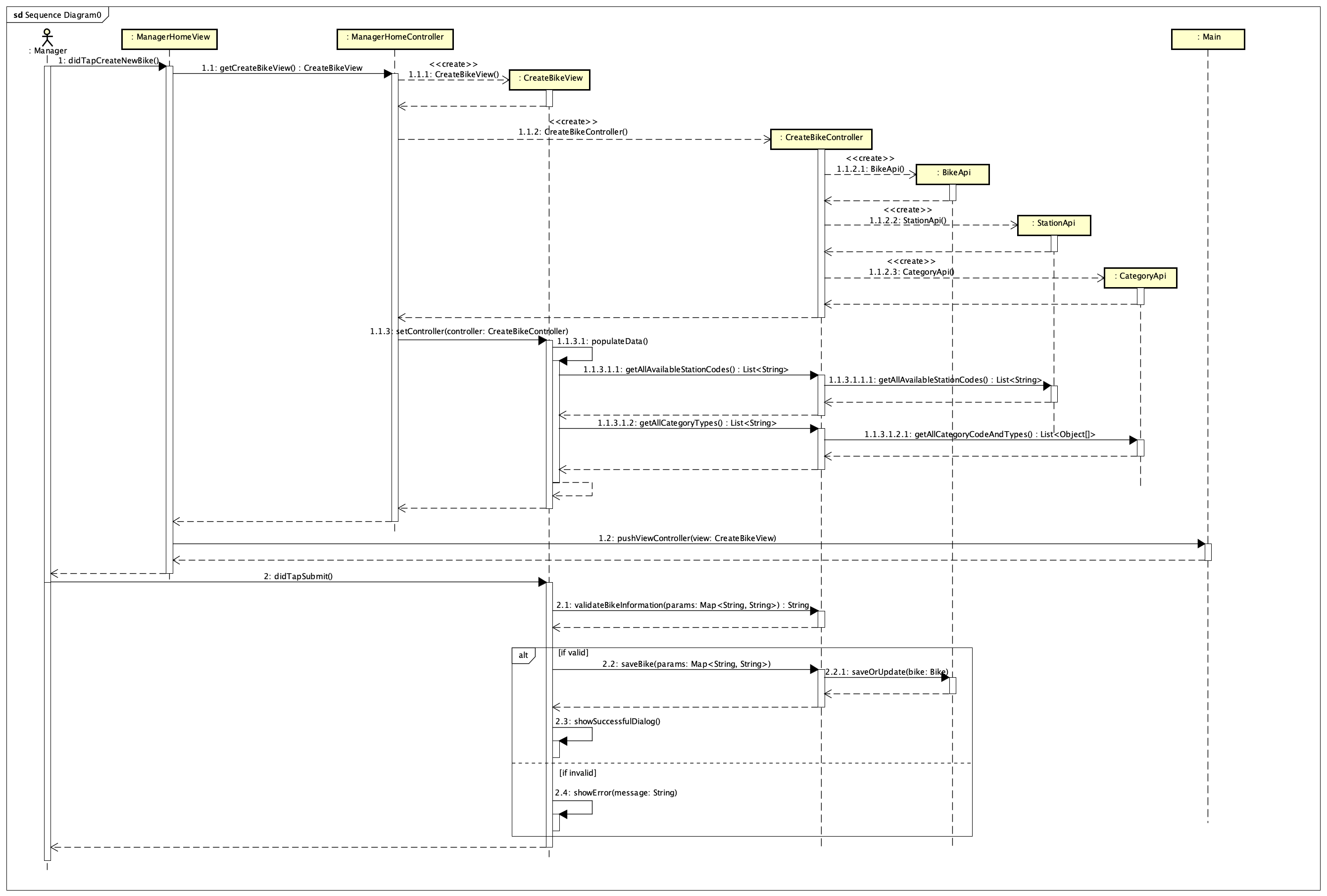
* MVC: For example: we have return bike screen, return bike controller, and corresponding models.
* Singleton pattern: We also apply singleton for all subsystem classes. It helps us not to initialize subsystem and connect/close to database so many times 🡺 save time, memory  
  Text

  Description automatically generated
* Strategy pattern

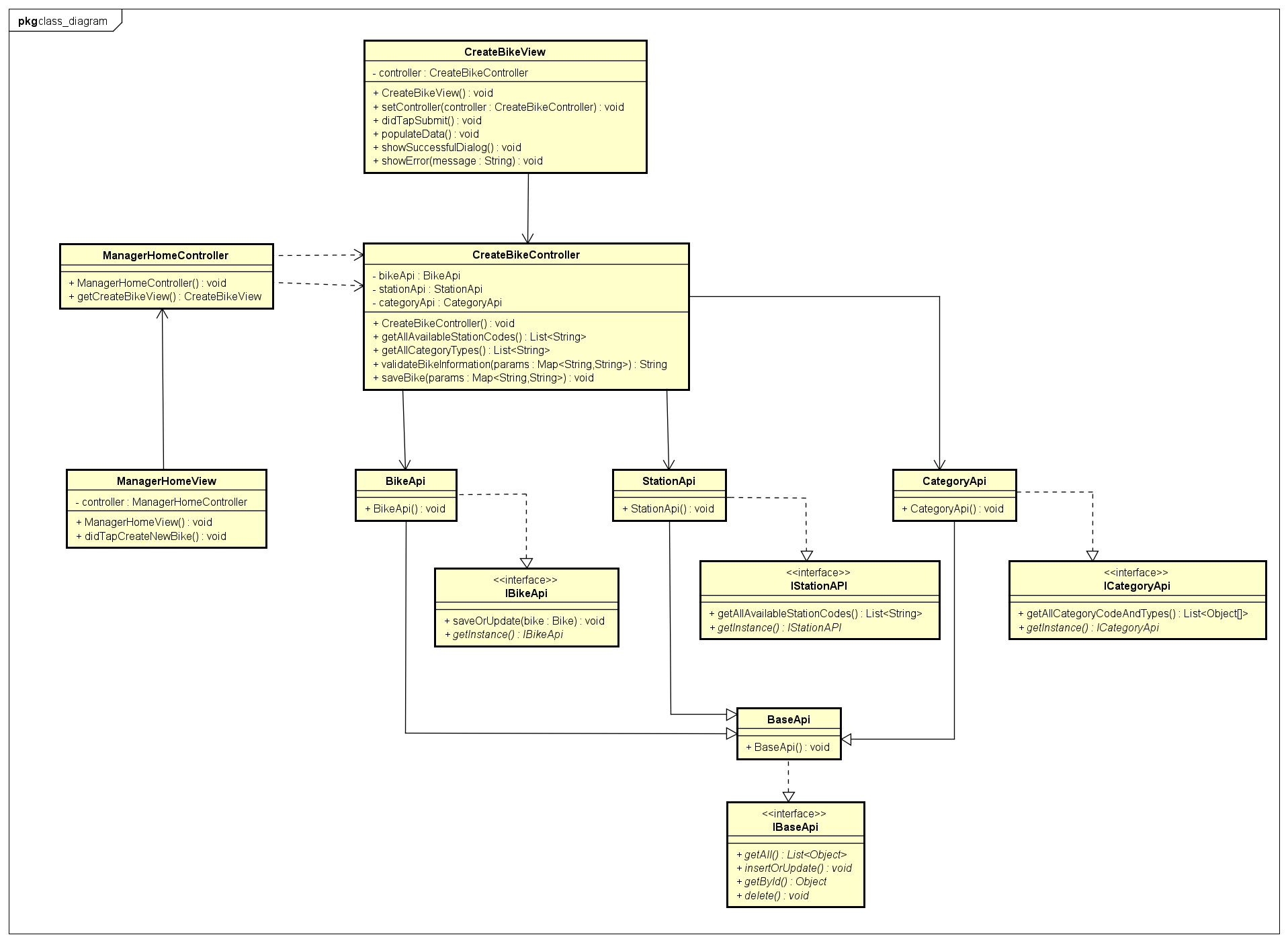
1. **Detail design**

1. Create a new bike

* 1. Sequence diagram

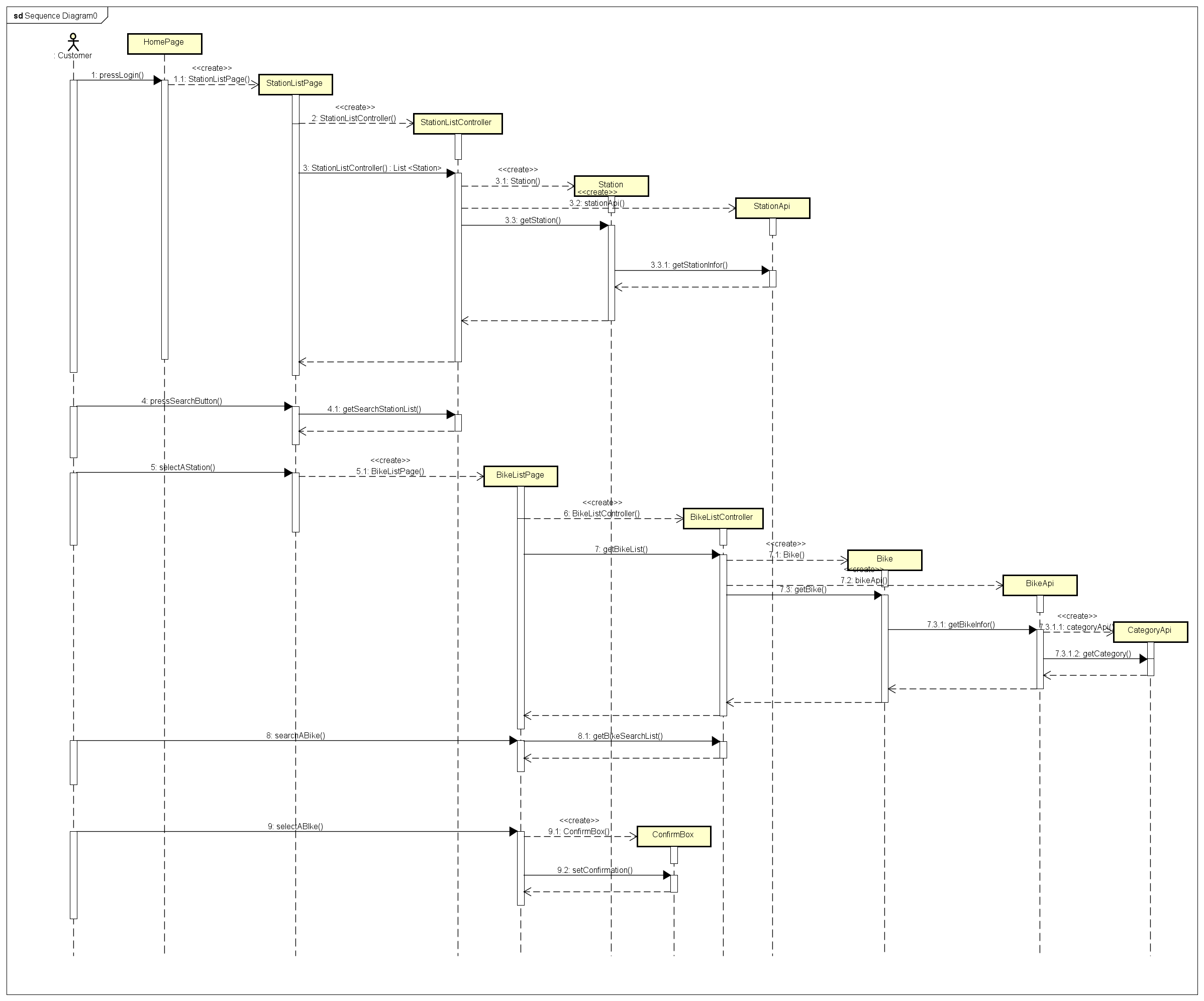


* 1. Sequence diagram

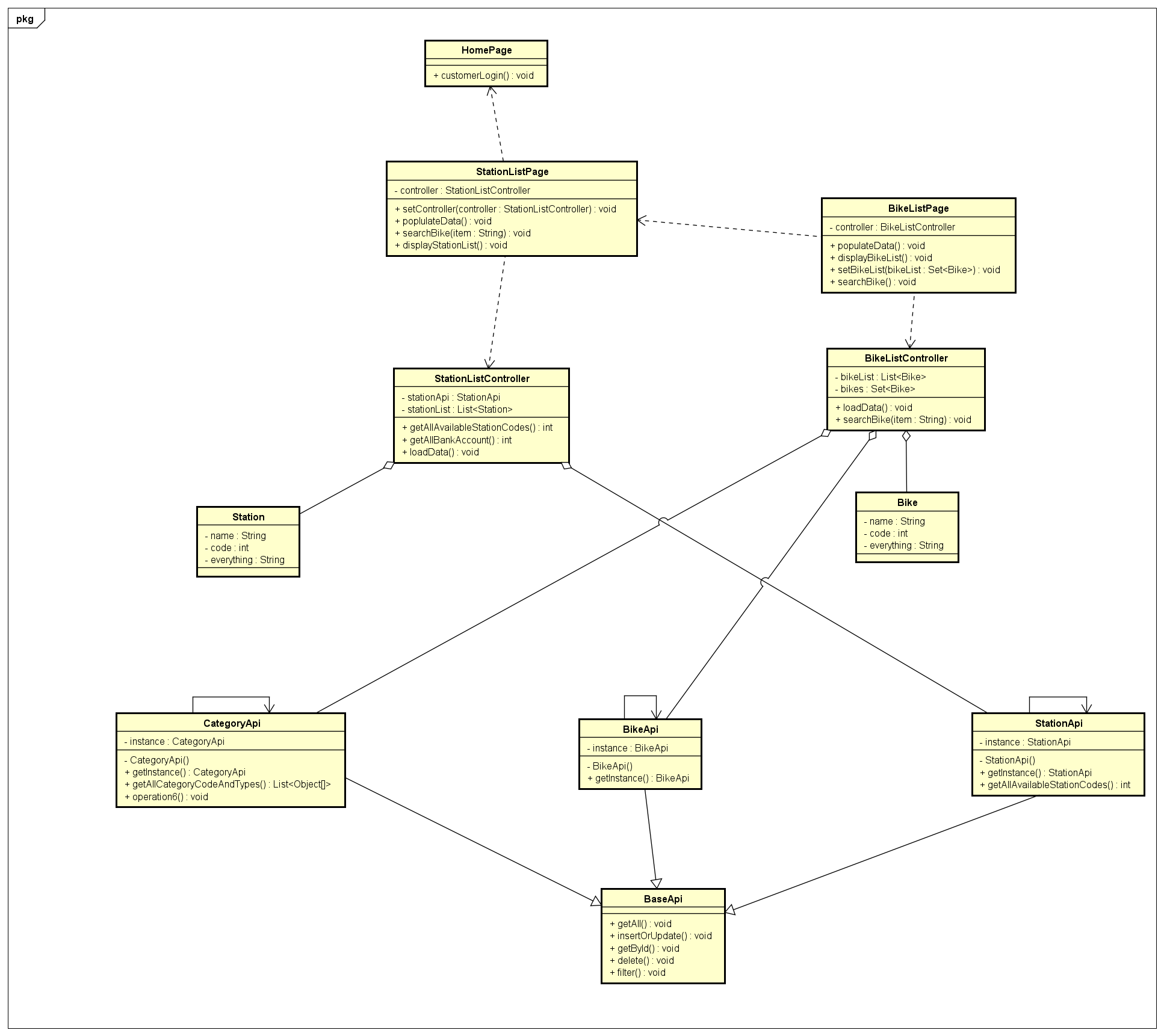


2. See station list

1. Sequence diagram



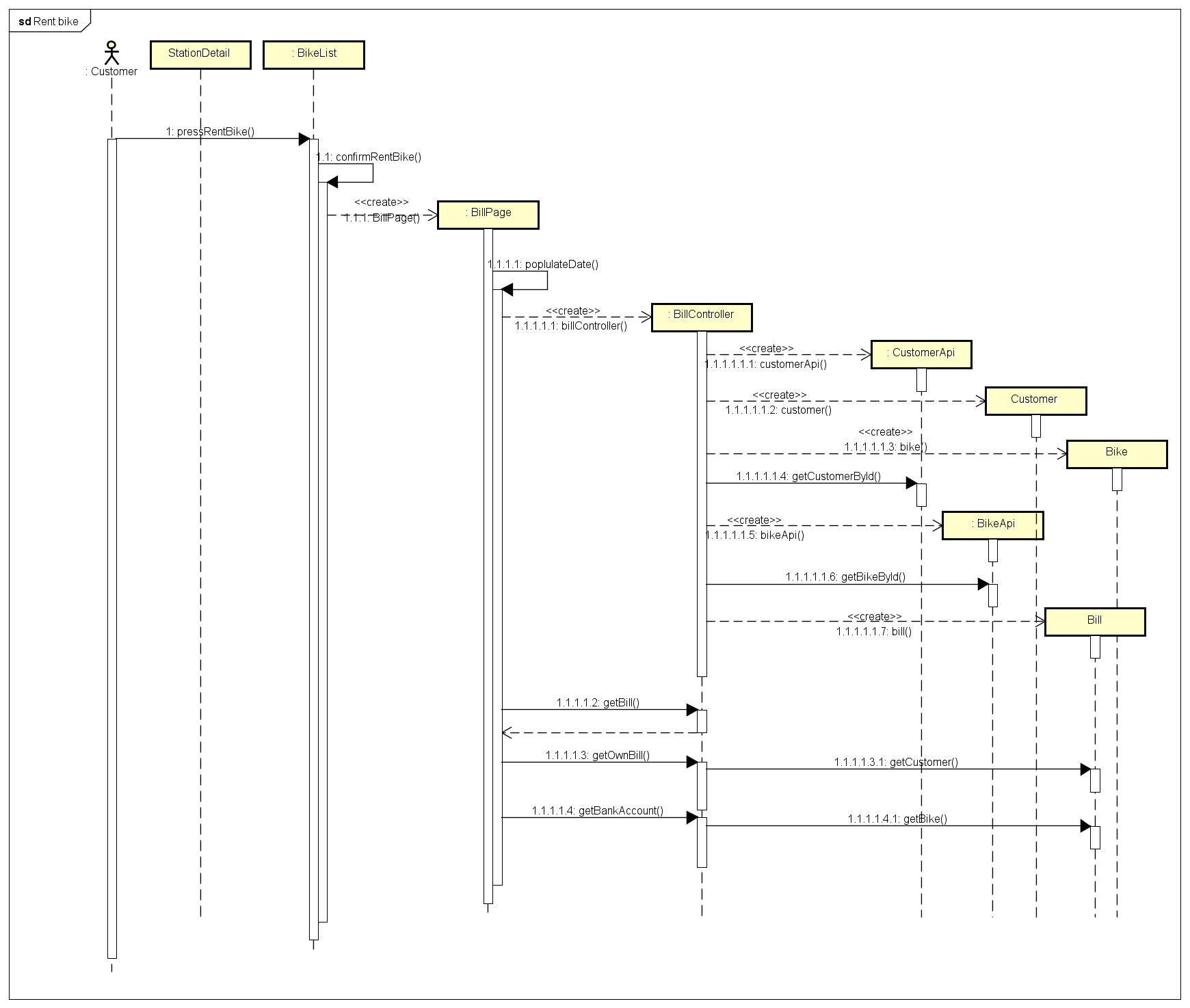
1. Class diagram



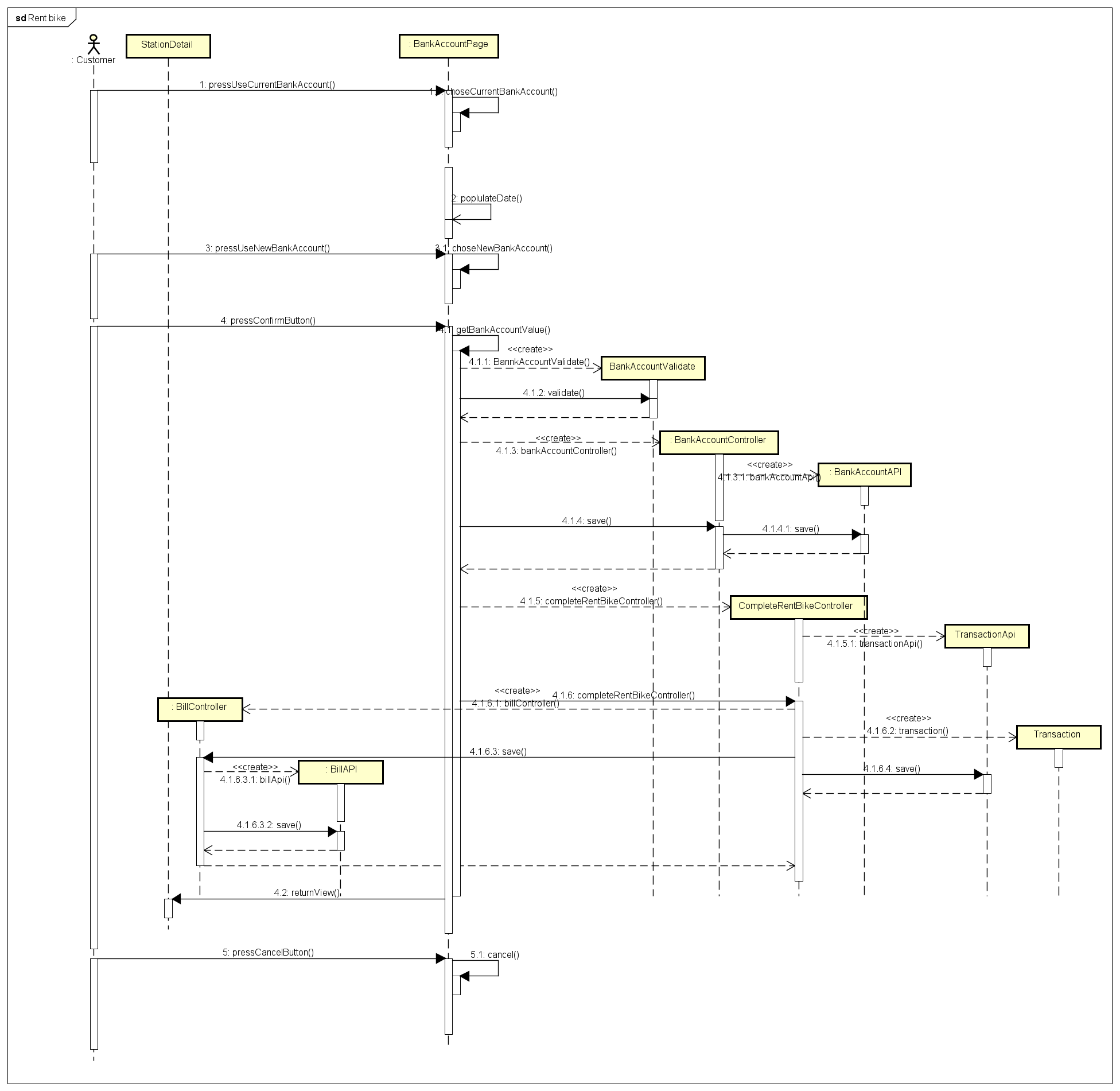
3. Rent bike

1. Sequence diagram

Sequence diagram: billScreen



Sequence diagram: bankAccountScreen & complete rent bike

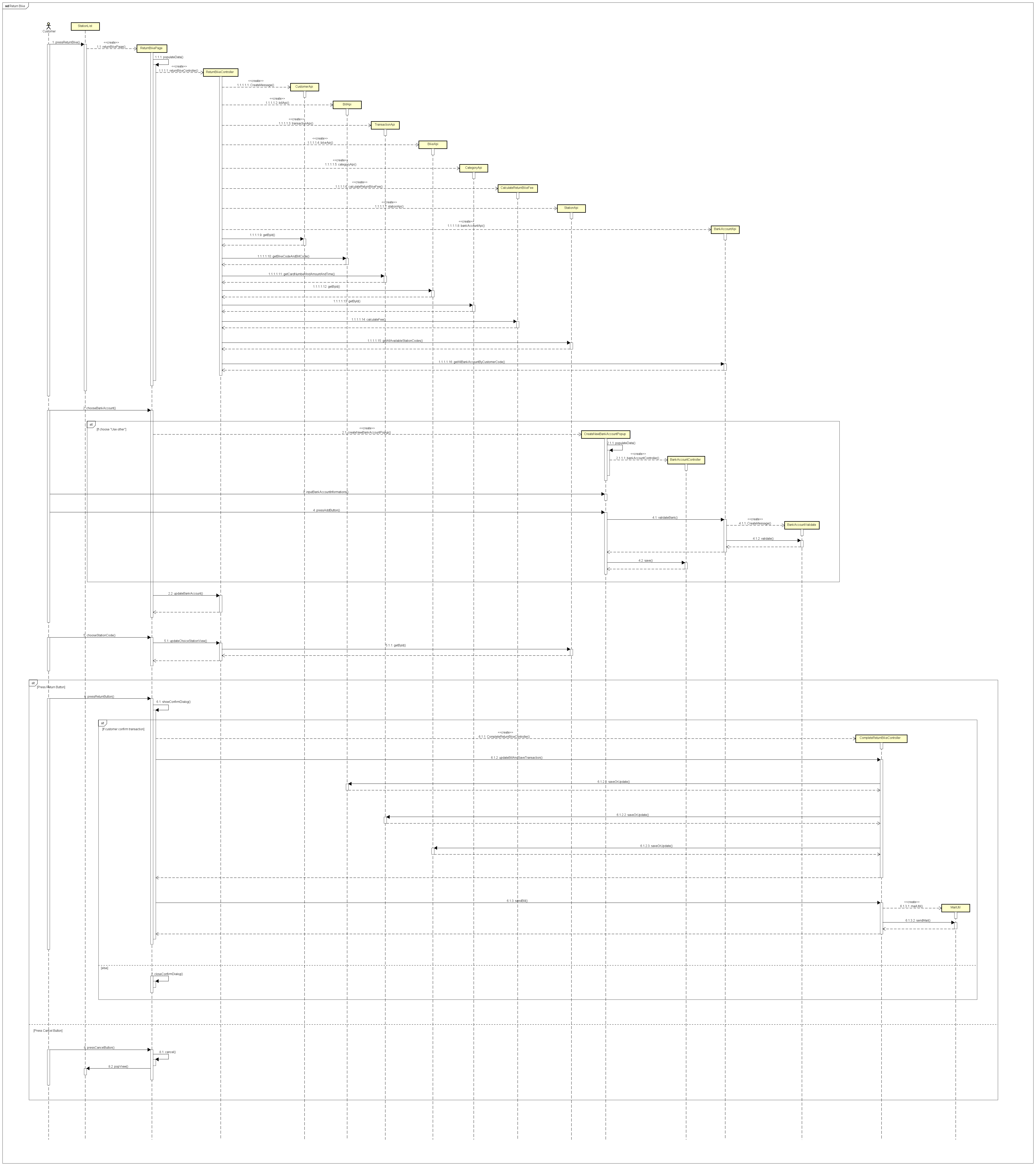


1. Sequence diagram

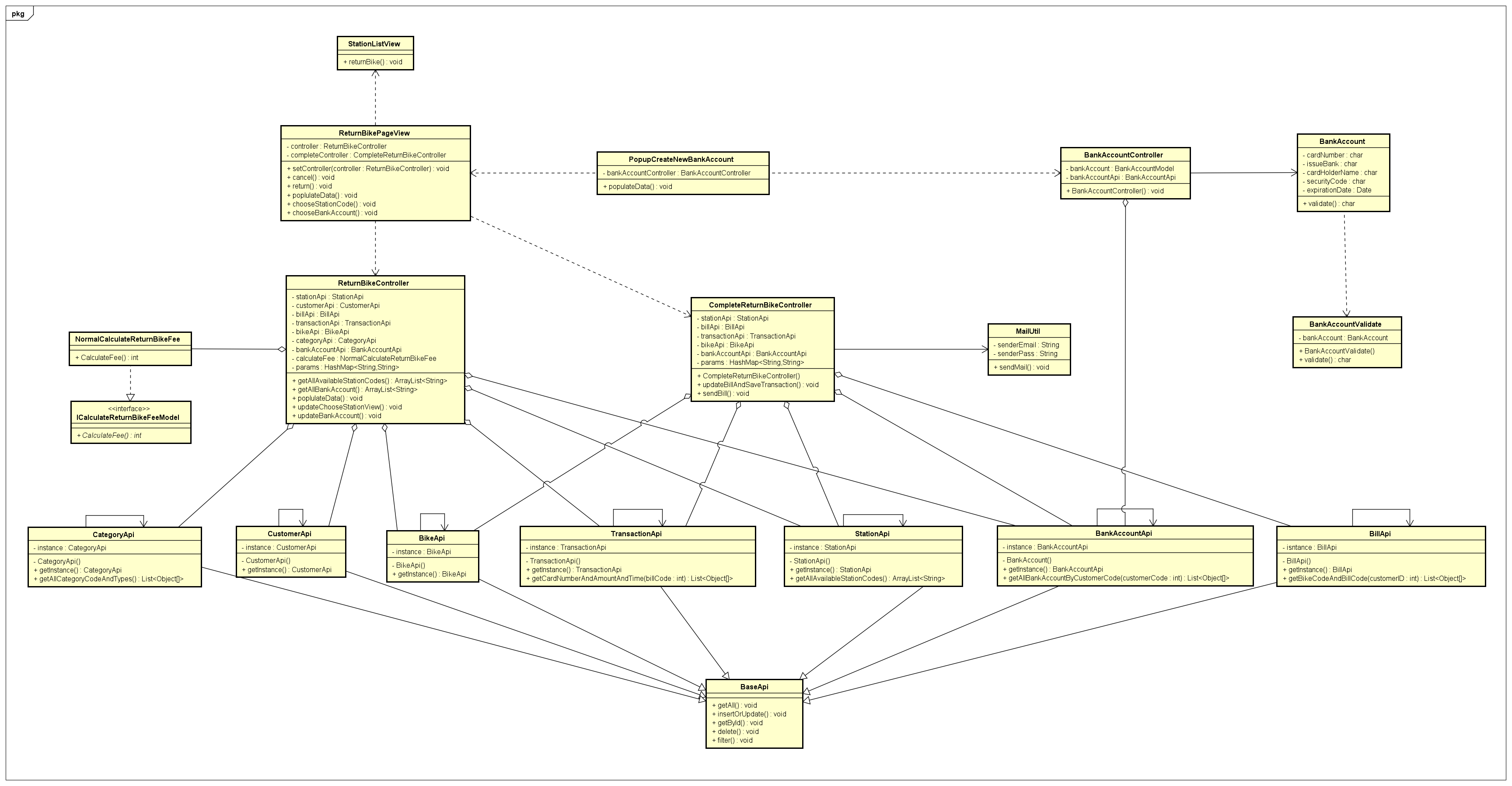


4. Return renting bike

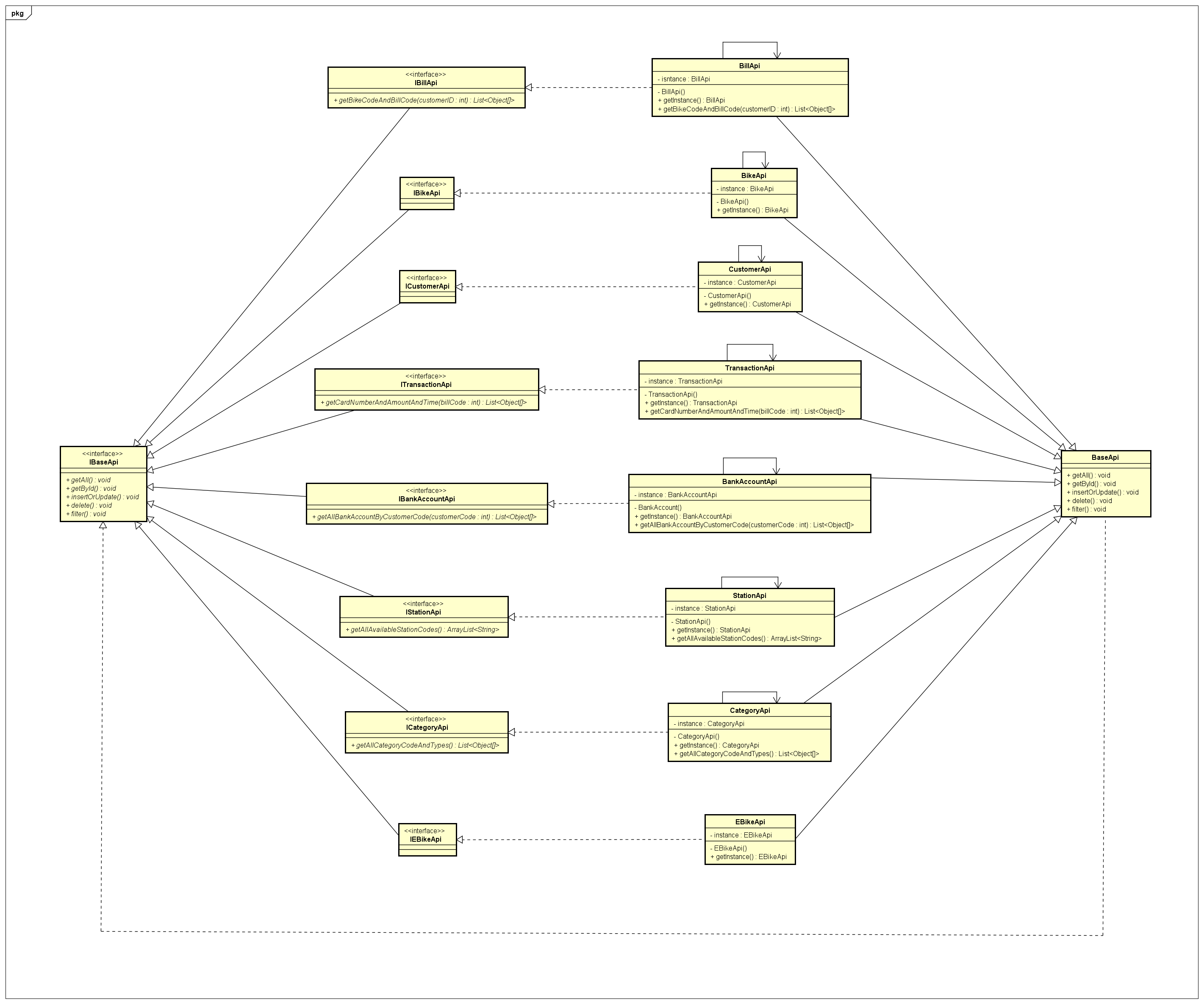
1. Sequence diagram



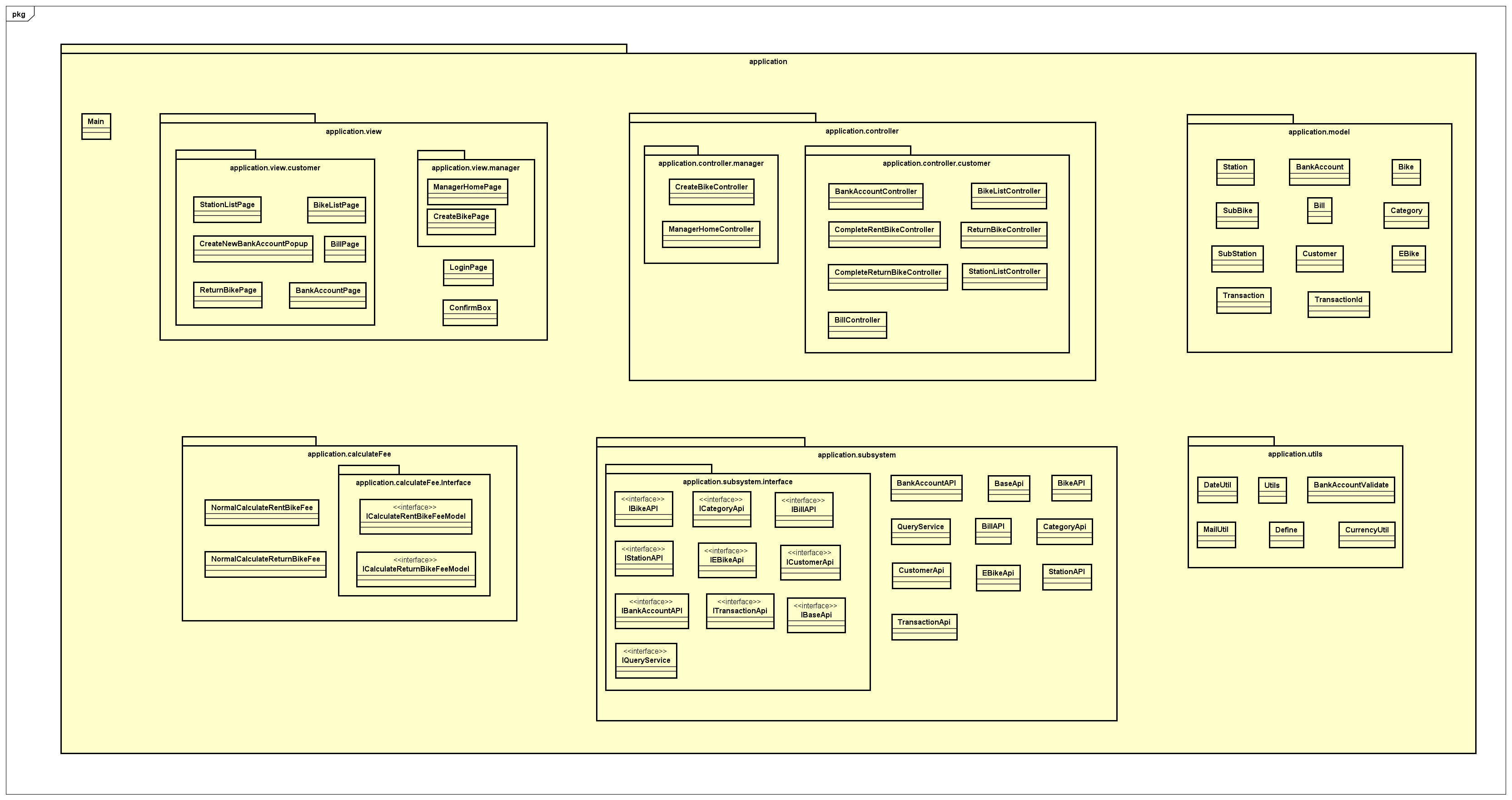
1. Class diagram:



**5. Detail design subsystem**



1. **Package organization**

****

1. **Package organization model:** MVC model
2. **Package meaning explanation:**

- application: contain the entire the program

- application.model: contains all the objects in the program such as bike, station and customer

- application.view: contains the user interfaces (theme). Divided into 2 sub-packages application.view.customer and application.view.manager are for customers and managers.

- application.controller: responsible for handling user requests made through the view. From there, the controller gives the right data to the user. Besides, the Controller also has the function of connecting to the model

- application.utils: contains utilities such as sending email, validating bank account, normalizing dates and currencies, etc.

- application.calculateFee: contains classes used to charge when starting to rent a car, charging when returning the car. It can be easily upgraded and maintained or modified charging manner.

- application.subSystem: Contains APIs that help the model connect to the database for manipulation such as viewing, retrieving, processing data, etc.

1. **Communication and navigation screen methods**

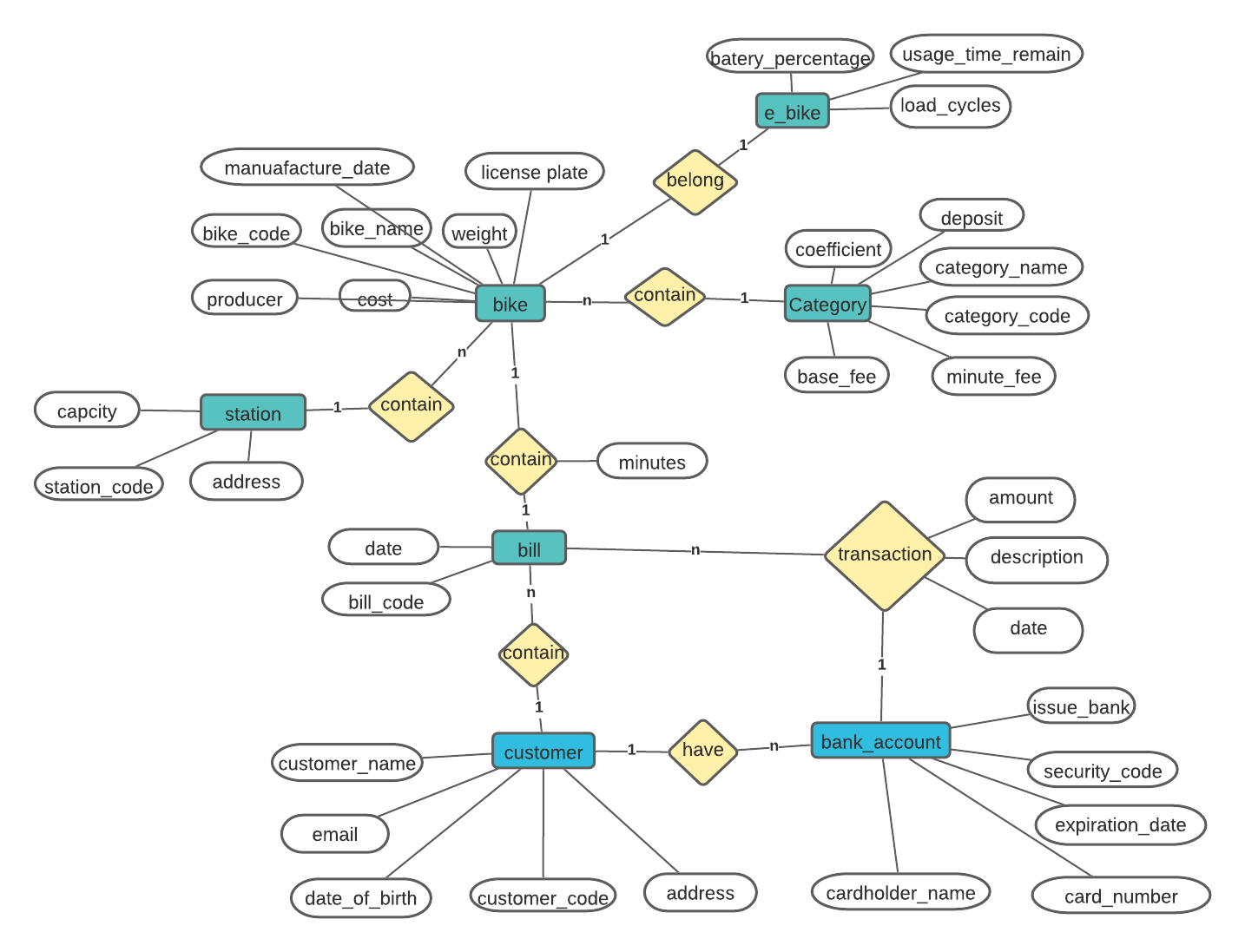
- Navigating to the next screen: pushViewController(ViewController vc), where the function changeScreen(ViewController vc) is used to change the screen.

- Back to previous screen: popViewController()

- Back to the previous n screens: popViewControllers(int n)

- Back to the root screen: popToRootViewController()

1. **Database design**

****

**Ảnh có chứa văn bản, đen, ảnh chụp màn hình, kim loại

Mô tả được tạo tự động**

1. **Glossary**

This document is used to define terminology specific to the problem domain, explaining terms, which may be unfamiliar to the reader of the use-case descriptions or other project documents. Often, this document can be used as an informal data dictionary, capturing data definitions so that use-case descriptions and other project documents can focus on what the system must do with the information.

**1. Station**

A place where bikes are available for customer to take or return

**2. Bike**

A bicycle

**3. E-bike**

An electronic bike

**4. Twin Bike**

A bike which is designed to be ridden by 2 people

**5. Rent**

An amount of money that you regularly pay so that you can use a bike

**6. Deposit**

A sum of money that is given as the first part of a larger payment

**7. GPS**

A system by which signals are sent from satellites to a special device, used to show the position of a person or thing on the surface of the earth very accurately

**8. Barcode**

A pattern of thick and thin lines that is printed on things you buy. It contains information that a computer can read

**9. Transaction**

A piece of business that is done between people, especially an act of buying or selling

**10. Credit card**

A small plastic card that you can use to buy goods and services and pay for them later

**11. Pin**

Source energy to enable electric device operating

**12. Balance**

Available remaining money in bank account

1. **Supplementary specification**

**1. Functionality**

1. Enable customer to rent a bike quickly and easily
2. Accept online payments (credit card/e-wallet)
3. Notify transaction details in email of customer

**2. Performance**

1. The system shall support up to 100 users at the same time
2. The responding time of system must be less than 2 seconds (1s at normal time and 2s at picky time)

**3. Reliability**

1. The system shall be available 24 hours a day 7 days a week
2. Be able to work properly in 200 hours continuously without errors, and back to normal working state within 2 hours after errors occurs

**4. Usability**

Available for all mobile operating system

**5. Supportability**

None