

Report Final Project

Nguyen Phu Truong 20184319

ITSSS | 25/12/2021

Appendix

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

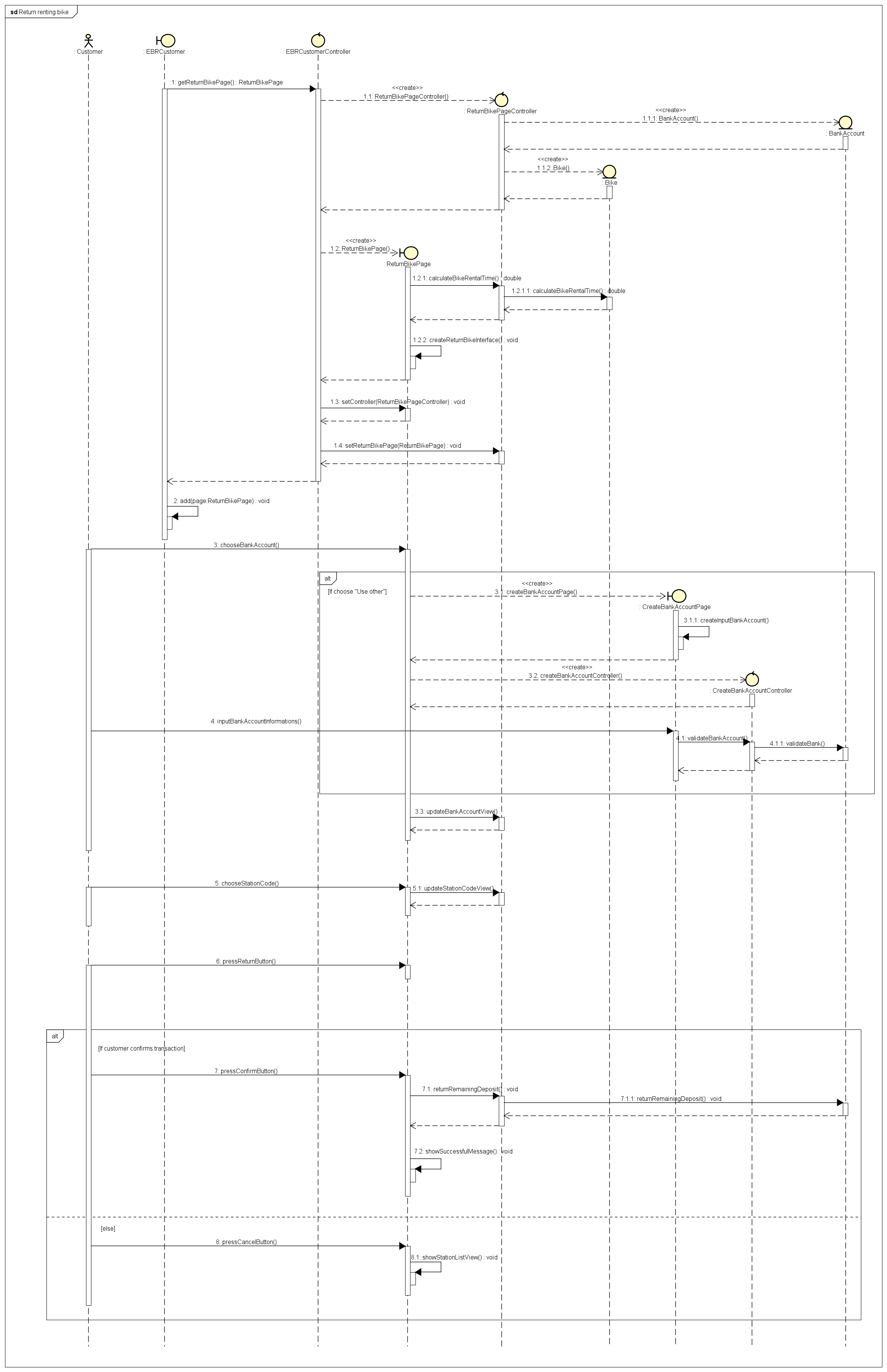
Use case “Return a renting bike”

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | | |

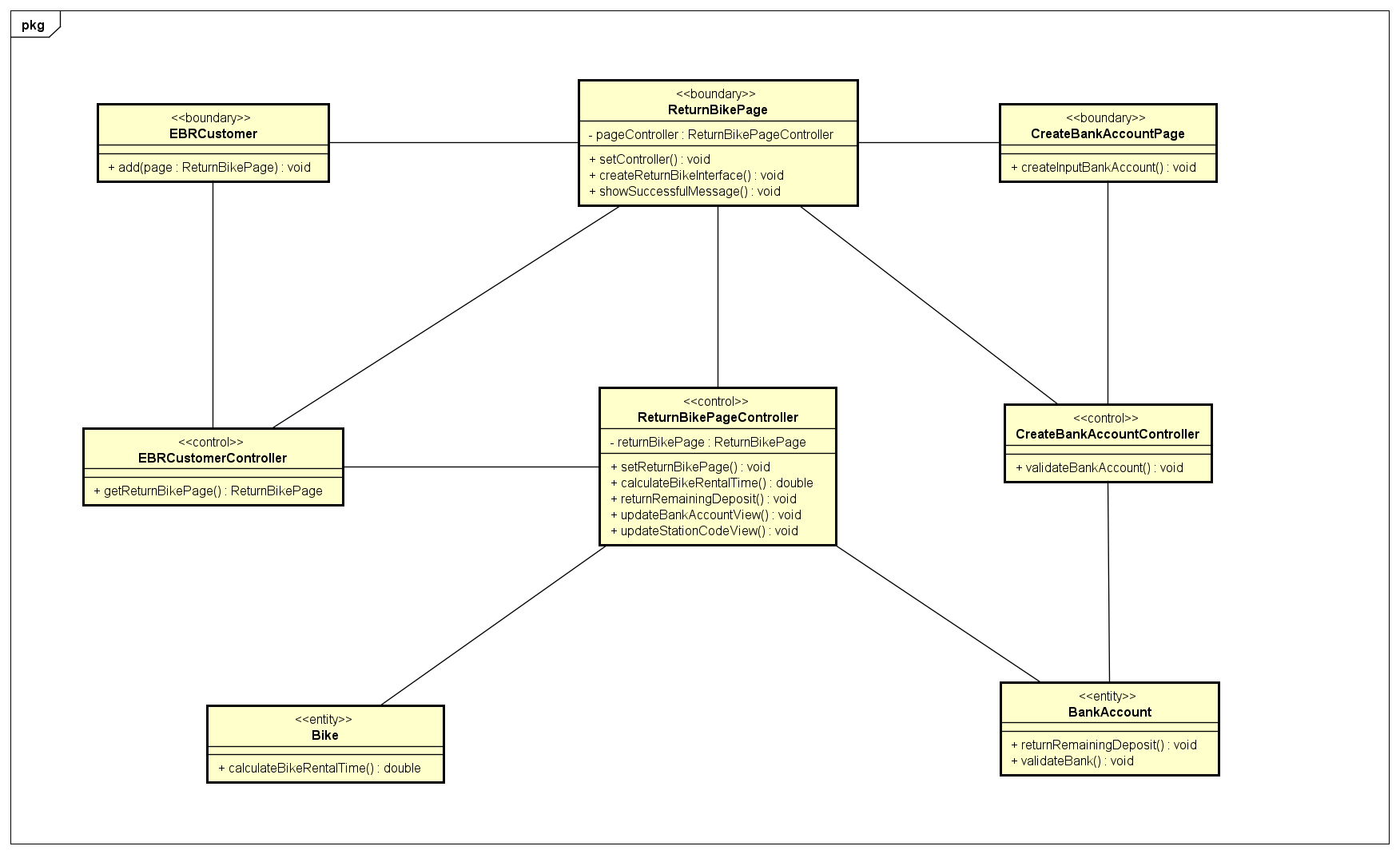
\* Input data:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Field** | **Description** | **Required?** | **Valid condition** | **Example** |
|  | Card number |  | yes | - Not null or empty  - Contain 9 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 |

1. Use case analysis
   1. Sequence analysis diagram



* 1. Class analysis diagram



1. GUI design
   1. Standardizing the screen configuration
2. Display

Resolution: 900 x 600 px

1. 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.

1. Control

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

Check input: check empty and format.

1. Enter input from keyboard

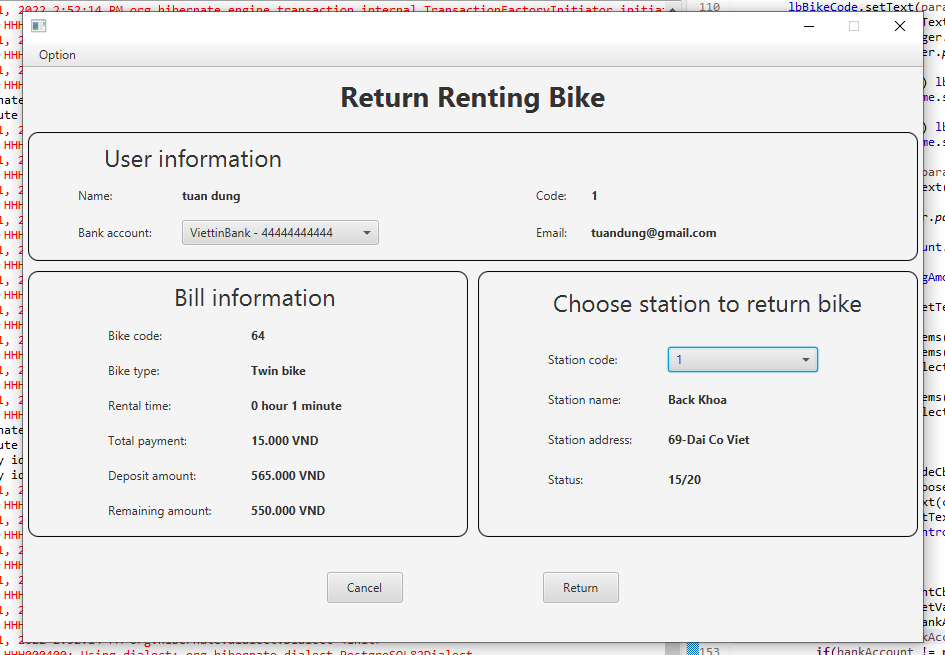
No keyboard shortcuts. Using button to return previous screen. Otherwise, button “X” in top-right of window to close screen.

1. Error

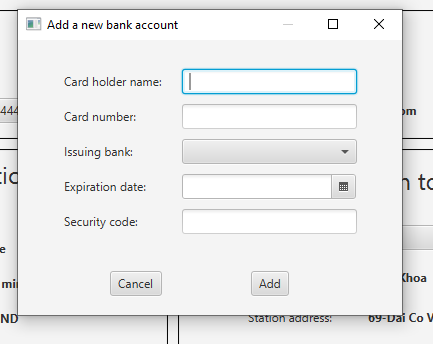
Show message by label warning.

* 1. Create screen images

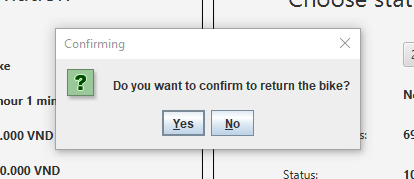
1. Return Bike Screen



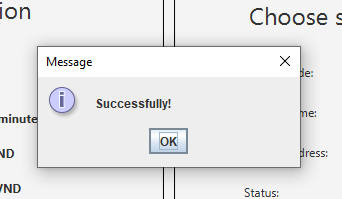
1. Create a New Bank Account Pop-up



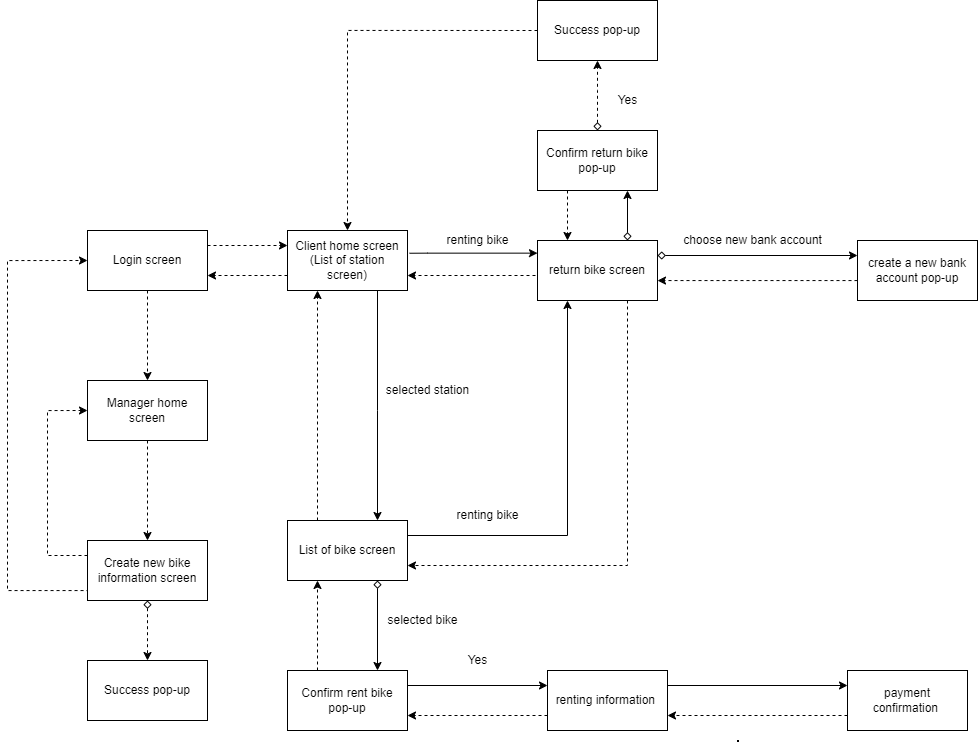
1. Confirm Return Bike Pop-up



1. Success Pop-up



* 1. Create a screen transition diagram



* 1. Creating screen specification   
     \* screen specification table

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

|  |  |  |  |
| --- | --- | --- | --- |
| 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 of the bank account |
| “Cancel” button | Click | Cancel create a new bank account and close the pop-up |
| “Add” button | Click | Create a new bank account and close the pop-up |

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

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

* 1. Define field

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Screen name | Add a new bank account pop-up | | | |
| Item name | Number of digits (bytes) | Type | Required | Remarks |
| Cardholder name | 100 | Char | true | Left-justified |
| Card number | 20 | Digit | true | Left- justified |
| Issue Bank |  | Char | true | Left-justified |
| Expiration Date |  | Date | true |  |
| Security Code |  | Char | true | Left-justified |

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
* Interface segregation principle
* Dependency Inversion Principle

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

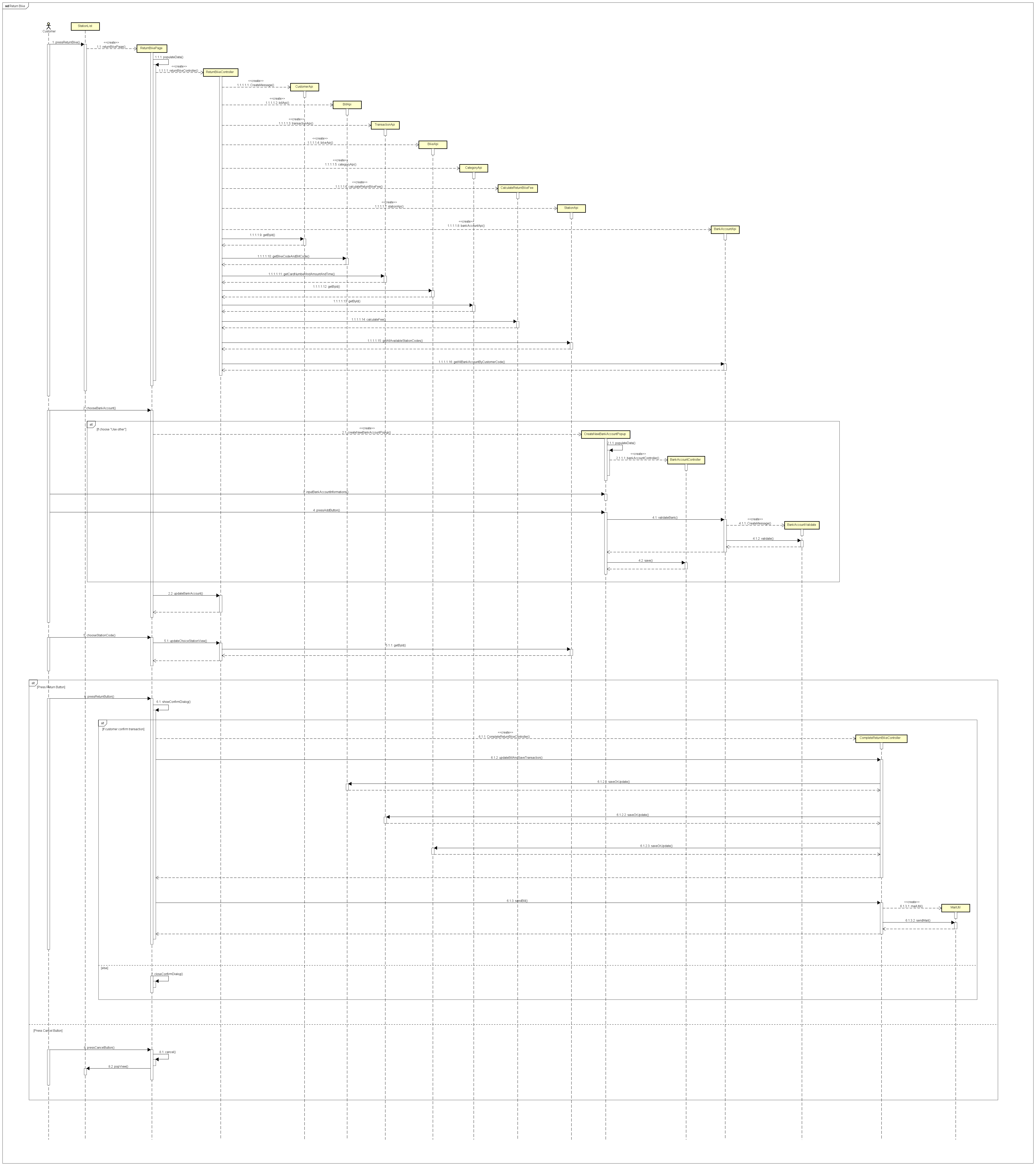
1. Pattern:

* MVC

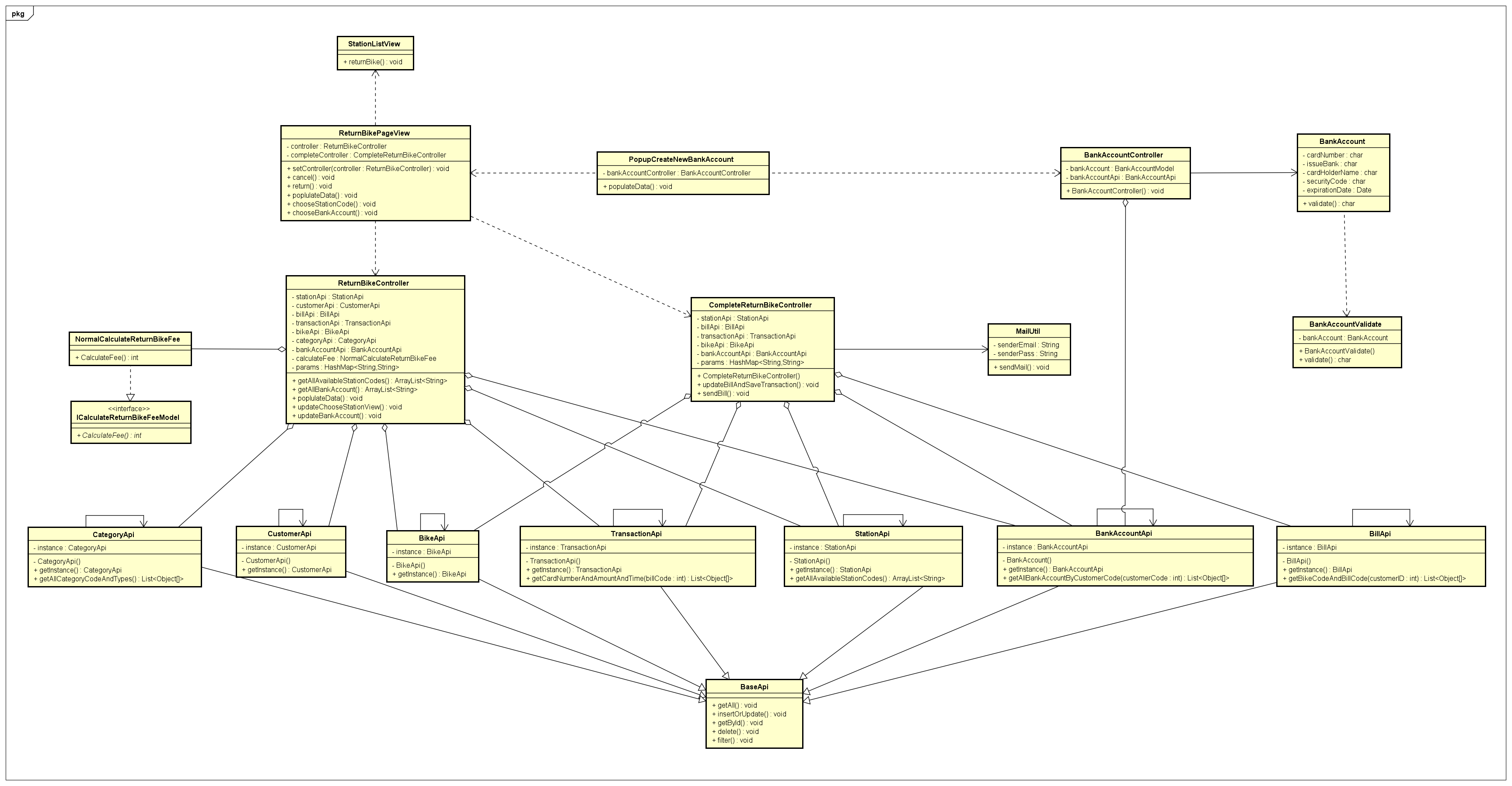
For example: we have return bike screen, return bike controller, and corresponding models.

* Singleton pattern
* Strategy pattern

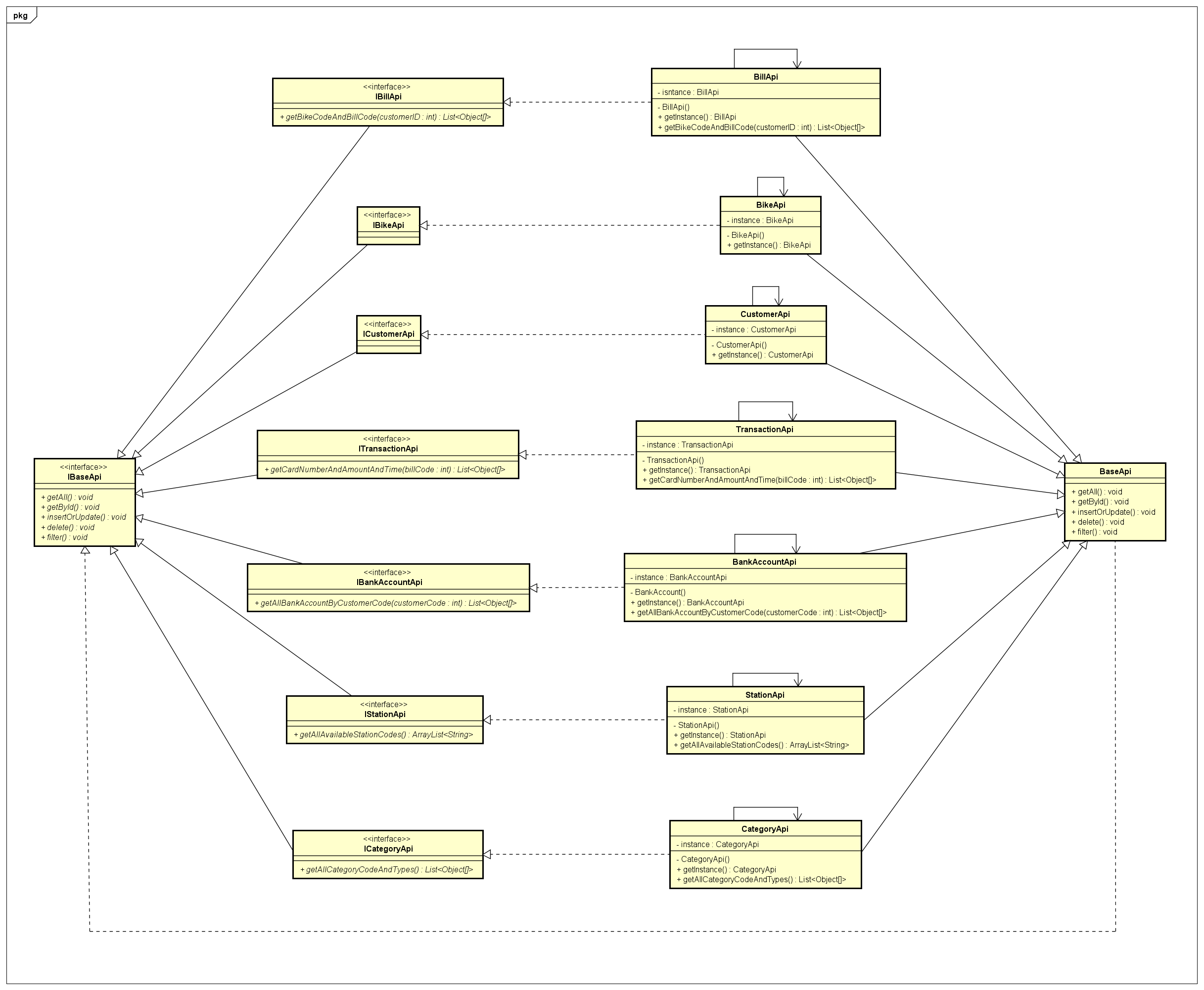
1. Detail design
   1. Sequence diagram: Return Bike Screen



* 1. Class diagram:



* 1. Detail design subsystem



1. Unit test

- Test\_Usecase\_ReturnBike.xlsx

- TestCase\_CalculateReturnBikeFee.doc