ITSS SOFTWARE DEVELOPMENT
Lab 05-Interface Design

Lecturer: NGUYEN Thi Thu Trang, trangntt@soict.hust.edu.vn

1. SUBMISSION GUIDELINE

You are required to push all your work to the valid GitHub repository complying with the naming convention:

"<MSTeamName>-<StudentID>.<StudentName>".

For this lab, you have to turn in your work twice before the following deadlines:

- **Right after class**: Push all the work you have done during class time to Github.
- **10 PM the day after the class**: Create a branch named "***release/lab05***" in your GitHub repository and push the full submission for this lab, including in-class tasks and homework assignments, to this branch.

2. IN-CLASS ASSIGNMENT

In this section, we will get familiar with the software detailed design process and start with interface design for the Case Study.

You are asked to work individually for this section, and then put all your file(s) and directories to a directory, namely "DetailedDesign/InterfaceDesign". After that, push your commit to your individual repository before the announced deadline.

2.1. USER INTERFACE

Boundary classes are used to model the interaction between a system and its surroundings, i.e., its actors. Hence, they can be utilized to capture the requirements on a user interface. The interactions between people and systems might be through various kinds of user interfaces (UI) such as batch interface, command-line interface (CLI), and graphical user interface (GUI).

In this subsection, we use GUI to illustrate how to design UIs step by step.

2.1.1. Standardizing the screen configuration

Display

Number of colors supported: 16,777,216 colors

Resolution: 1366×768 $pixels$

Screen

Location of standard buttons: At the bottom (vertically) and in the middle (horizontally) of the frame

Location of the messages: Starting from the top vertically and in the middle horizontally of the frame down to the bottom.

Display of the screen title: The title is located at the top of the frame in the middle.

Consistency in expression of alphanumeric numbers: comma for separator of thousand while strings only consist of characters, digits, commas, dots, spaces, underscores, and hyphen symbol.

Control

Size of the text: medium size (mostly 24px). Font: Segoe UI. Color: #000000

Input check process: Should check if it is empty or not. Next, check if the input is in the correct format or not

Sequence of moving the focus: There will be no stack frames. Each screen will be separated. However, the manual is considered a popup message, as the main screen cannot be operated while the manual screen is shown. After the opening screen, the app will start with splash screen, and then the first screen (home screen) will appear.

Sequences of the system screens:

1. Splash screen (first screen)
2. Home screen
3. View cart screen – view items in the cart
4. Delivery form – fill delivery information
5. Invoice screen – view order details
6. Payment form – fill payment information
7. Result screen

Direct input from the keyboard

There will be no shortcuts. There are back buttons to move back to the previous screen. Also, there is the close button "X" located at the title bar to the right to close the screen.

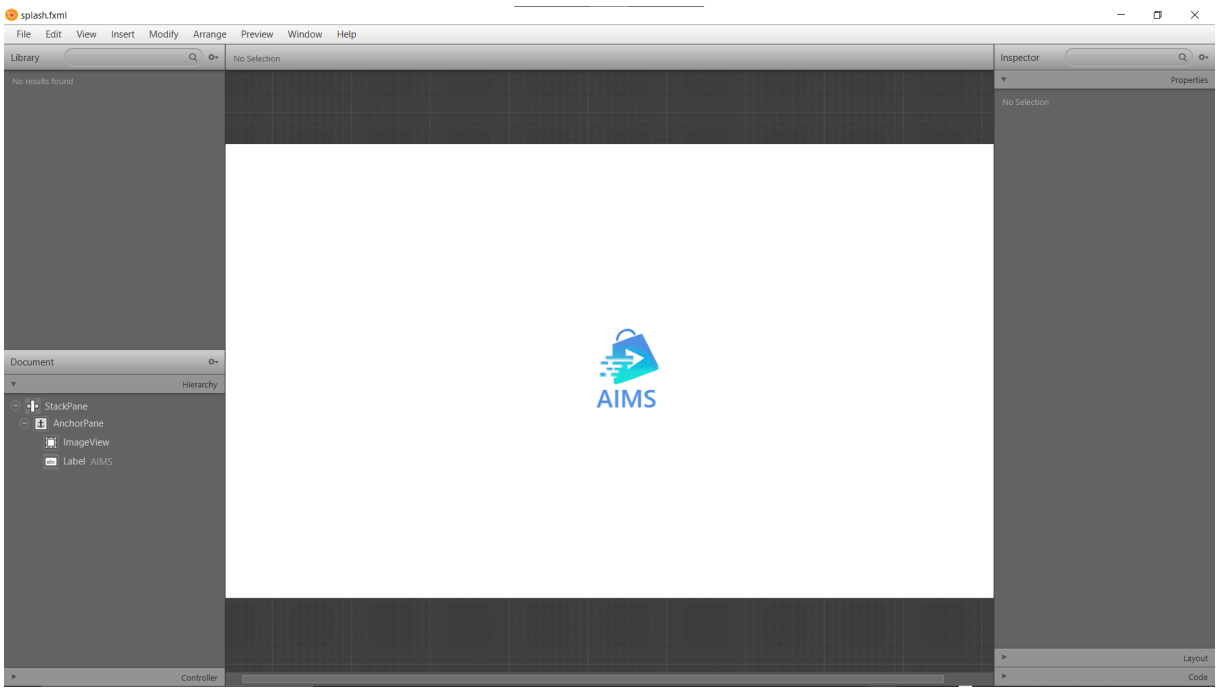
Error

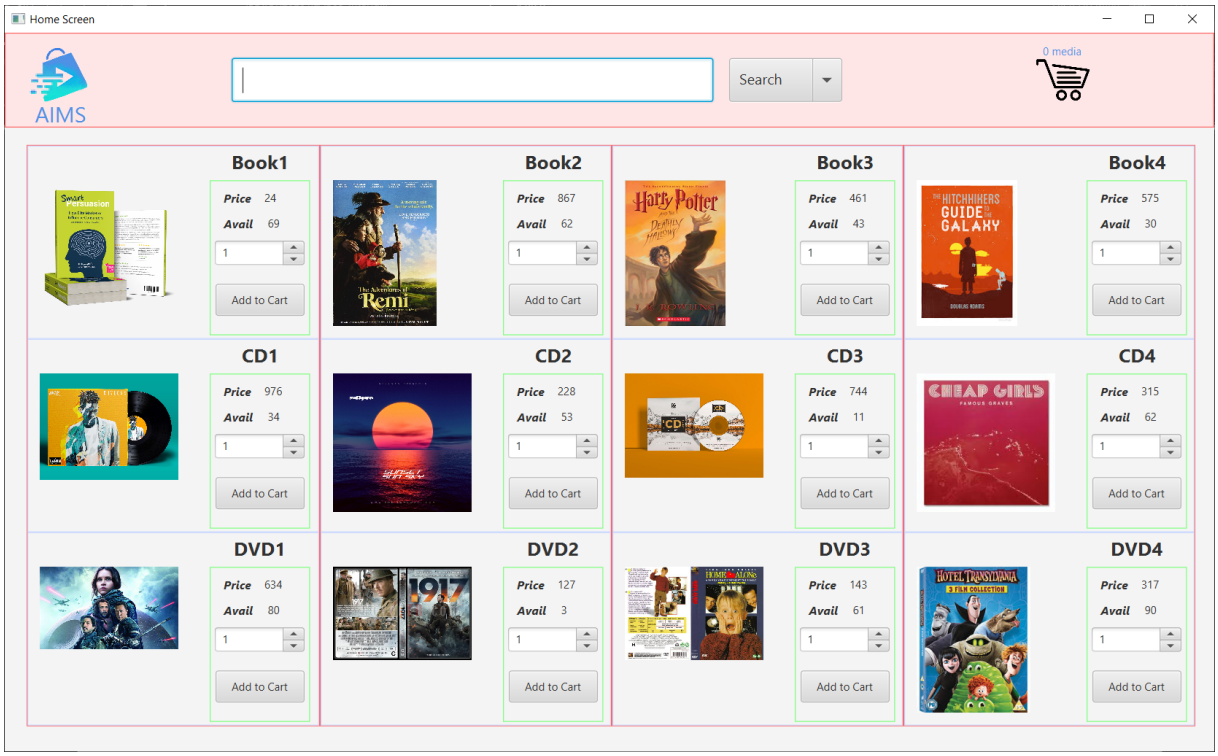
A message will be given to notify the users what is the problem.

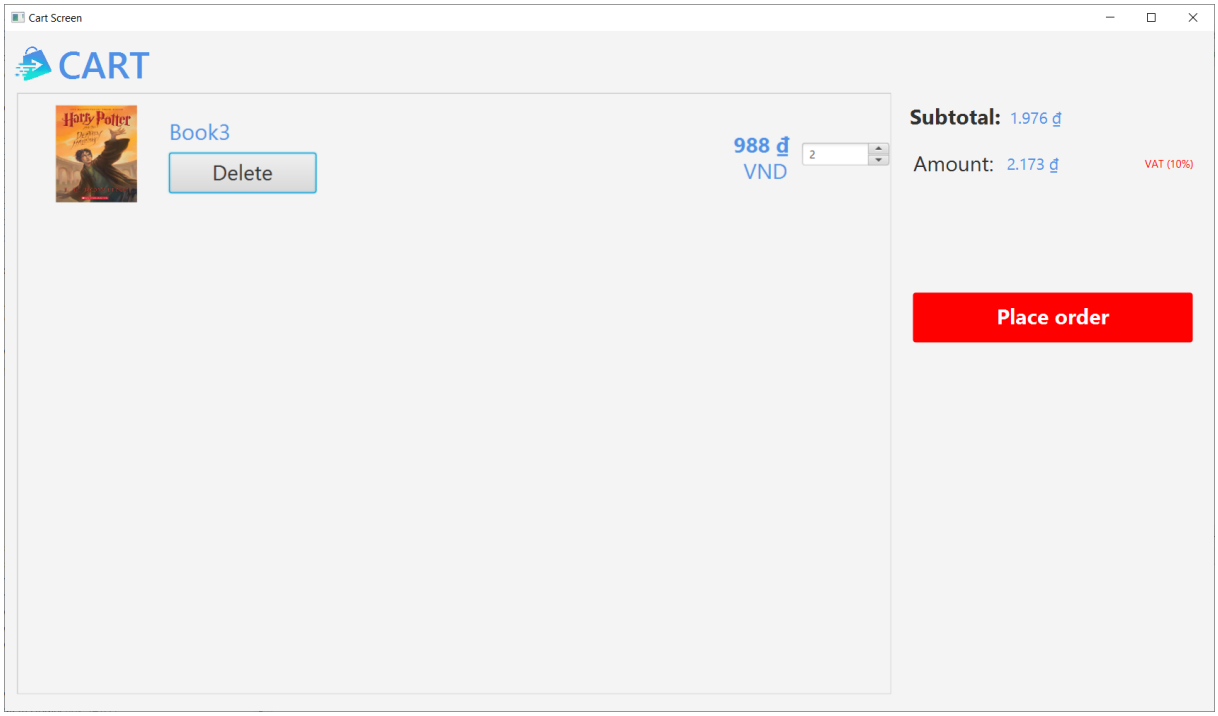
2.1.2. Creating screen images

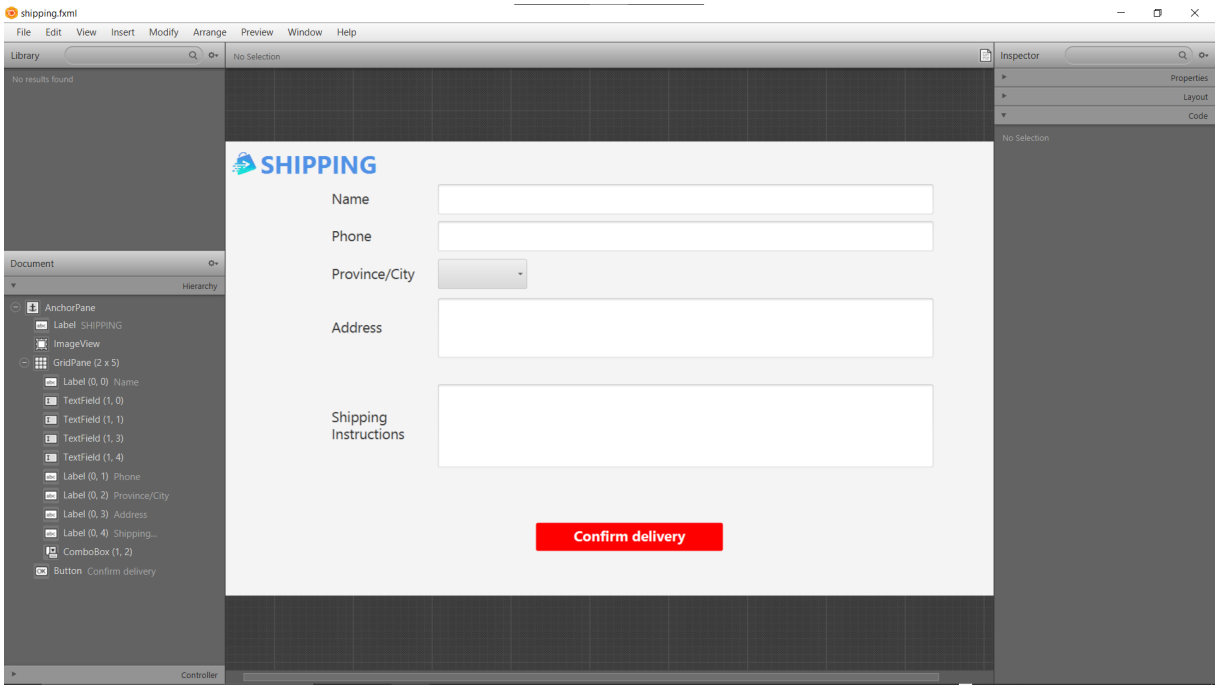
Creating screen images or mockup designs can be done by using applications such as https://moqups.com/, http://draw.io/, Figma, InVision Studio, Paint, Adobe Graphic design software, Adobe XD, or Scene Builder.

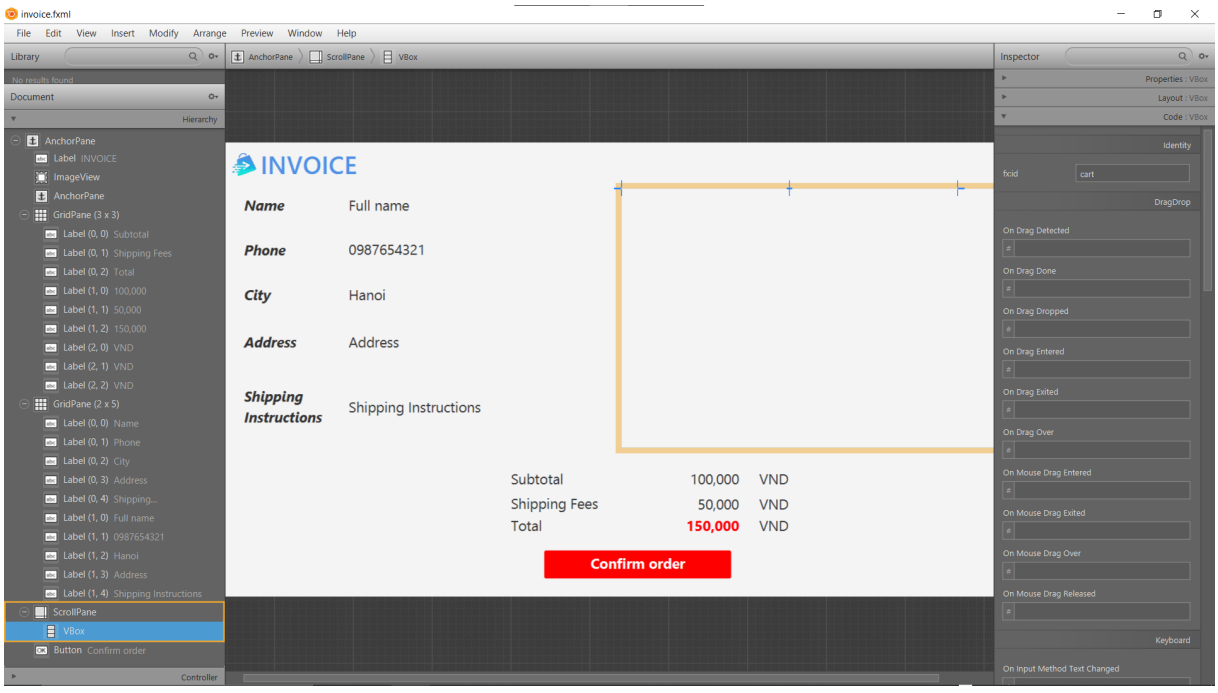
The following figures are made with Scene Builder.

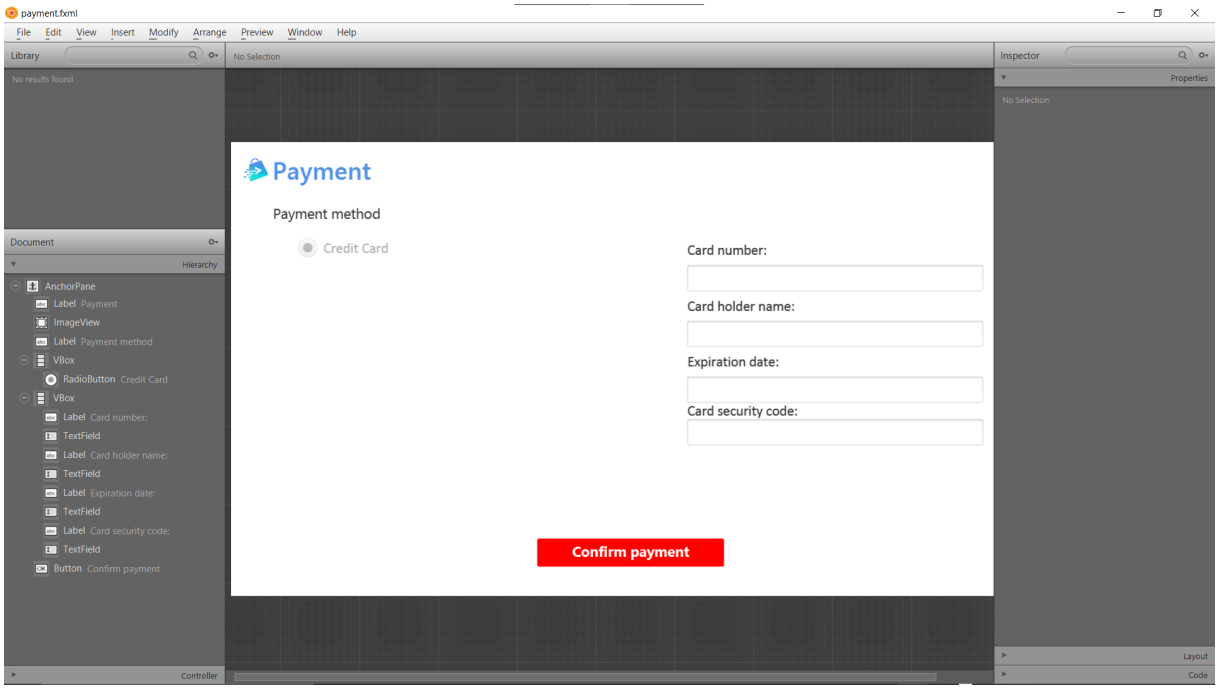


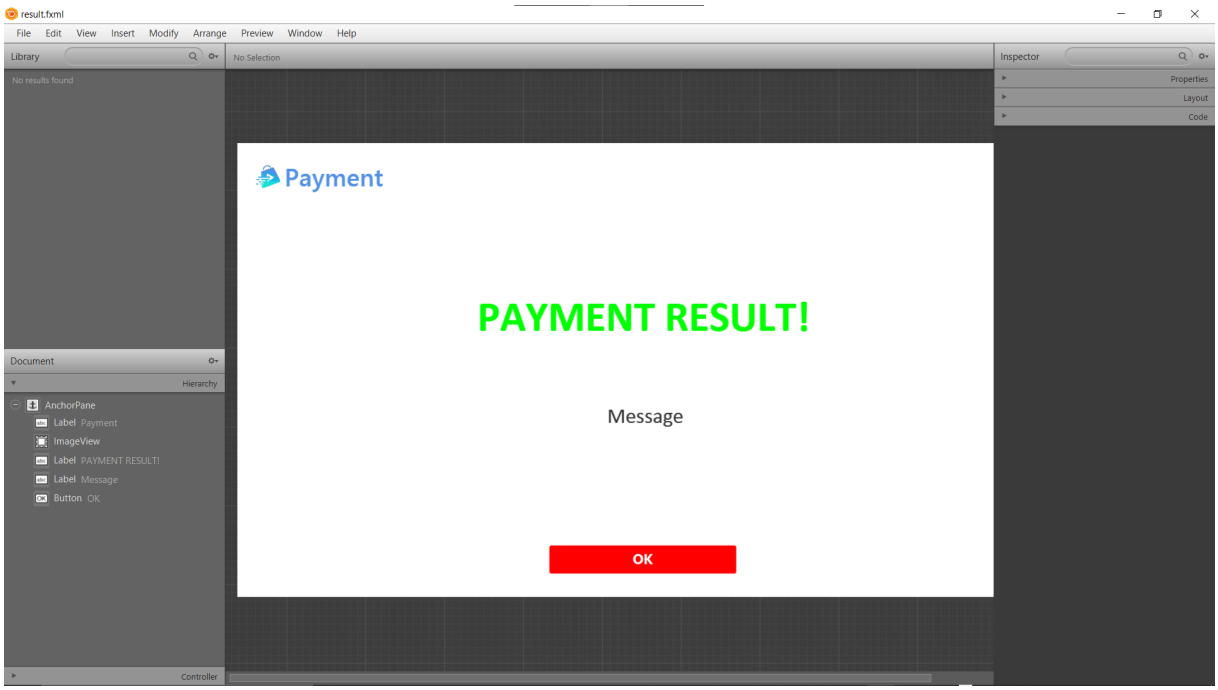

Home Screen
AIMS
Search
0 media
Book1
Price 24
Avail 69
Book2
Price 867
Avail 62
Book3
Price 461
Avail 43
Book4
Price 575
Avail 30
CD1
Price 976
Avail 34
CD2
Price 228
Avail 53
CD3
Price 744
Avail 11
CD4
Price 315
Avail 62
DVD1
Price 634
Avail 80
DVD2
Price 127
Avail 3
DVD3
Price 143
Avail 61
DVD4
Price 317
Avail 90
Add to Cart


Cart Screen
CART
Book3
Delete
988 ₫
VND
Subtotal: 1.976 ₫
Amount: 2.173 ₫
VAT (10%)
Place order


shipping.fxml
File Edit View Insert Modify Arrange Preview Window Help
SHIPPING
Name
Phone
Province/City
Address
Shipping Instructions
Confirm delivery


invoice.fxml
File Edit View Insert Modify Arrange Preview Window Help
INVOICE
Name Full name
Phone 0987654321
City Hanoi
Address Address
Shipping Instructions Shipping Instructions
Subtotal 100,000 VND
Shipping Fees 50,000 VND
Total 150,000 VND
Confirm order

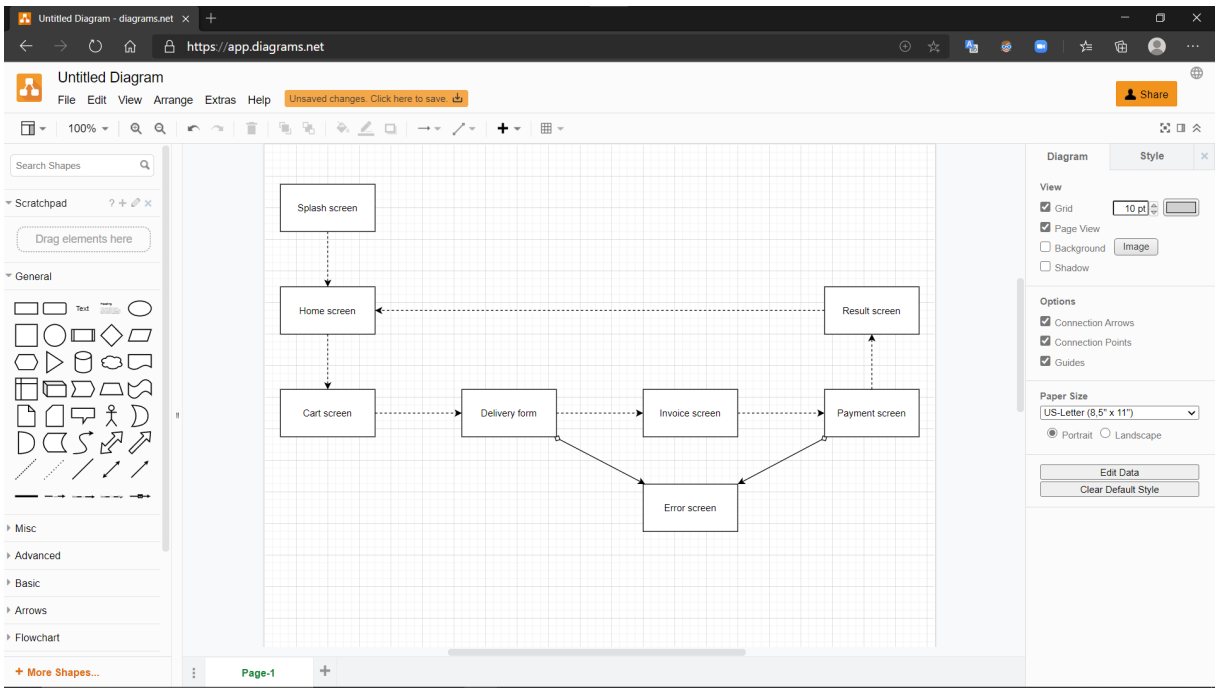




2.1.3. Creating a screen transition diagram

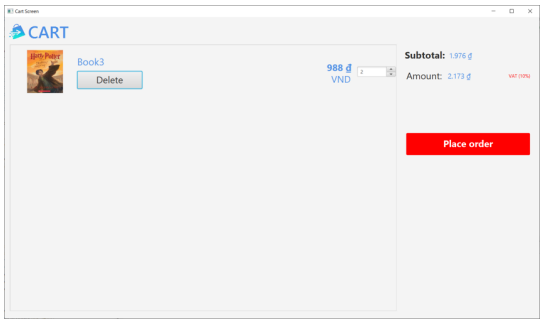
Creating screen transition diagram can be done by using draw.io or any drawing applications as specified in the previous section. In case you choose to use Moqups, you need to use the UML Class Diagram template of Moqups.

To illustrate, an example of the screen transition diagram for AIMS Software, made with http://draw.io/, is shown as follows .



2.1.4. Creating screen specification

Screen specification

AIMS Software		Date of creation	Approved by	Reviewed by	Persion in charge
Screen specification	View cart screen	30/10/2020			Đỗ Minh Hiếu
		Control	Operation	Function	
		Area for displaying the subtotal	Initial	Display the subtotal	
		Area for display items in the cart	Initial	Display the media with the corresponding information	
		Place order button	Click	Display the Delivery form	
		Delete button	Click	Remove the item from the cart	

Defining the field attributes

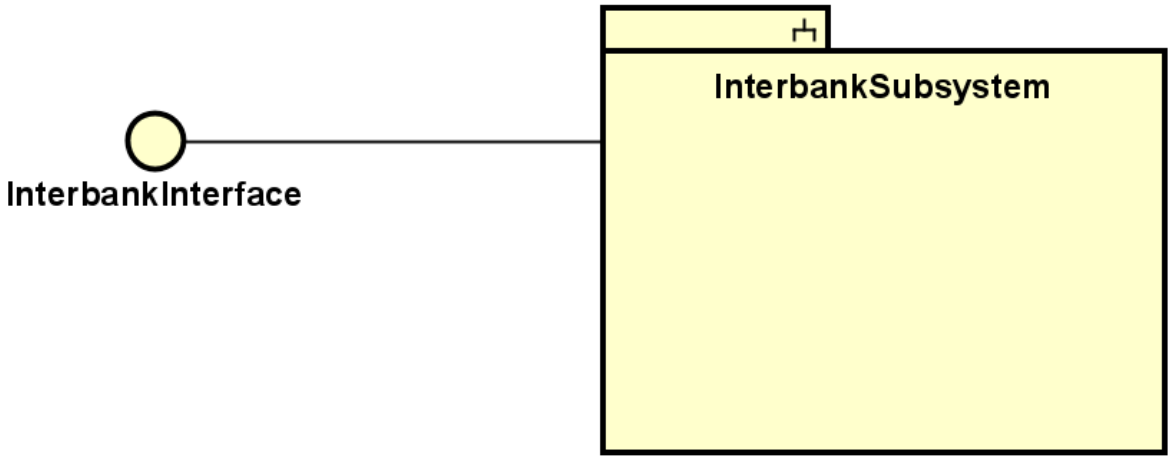
Screen name	View cart			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Media title	50	Numeral	Blue	Left-justified
Price	20	Numeral	Blue	Right justified
Subtotal	20	Numeral	Blue	Left-justified

You are asked to design GUIs step by step for UC "Place Rush Order."

2.2. INTERFACE DESIGN

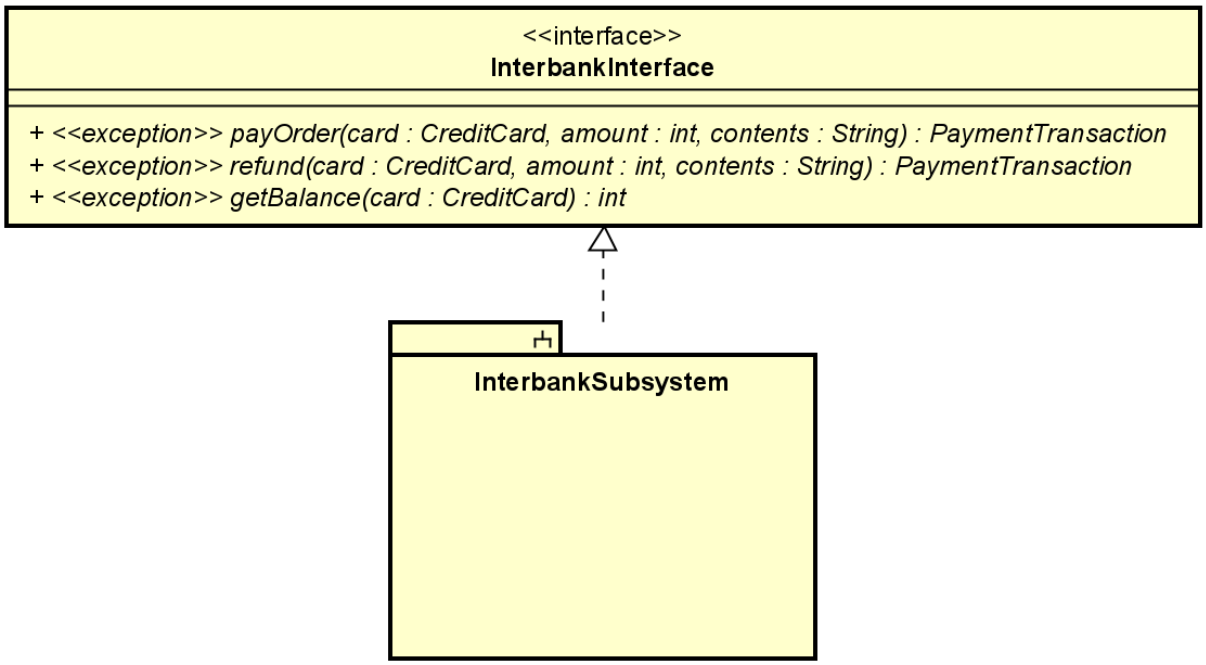
2.2.1. Identify subsystems

As can be seen, the InterbankBoundary class in the analysis class diagram provides complex services related to the communication between AIMS Software and the Interbank while representing an independent capability with clear interface. Hence, we evolve the Interbank Boundary from an analysis class into a subsystem.



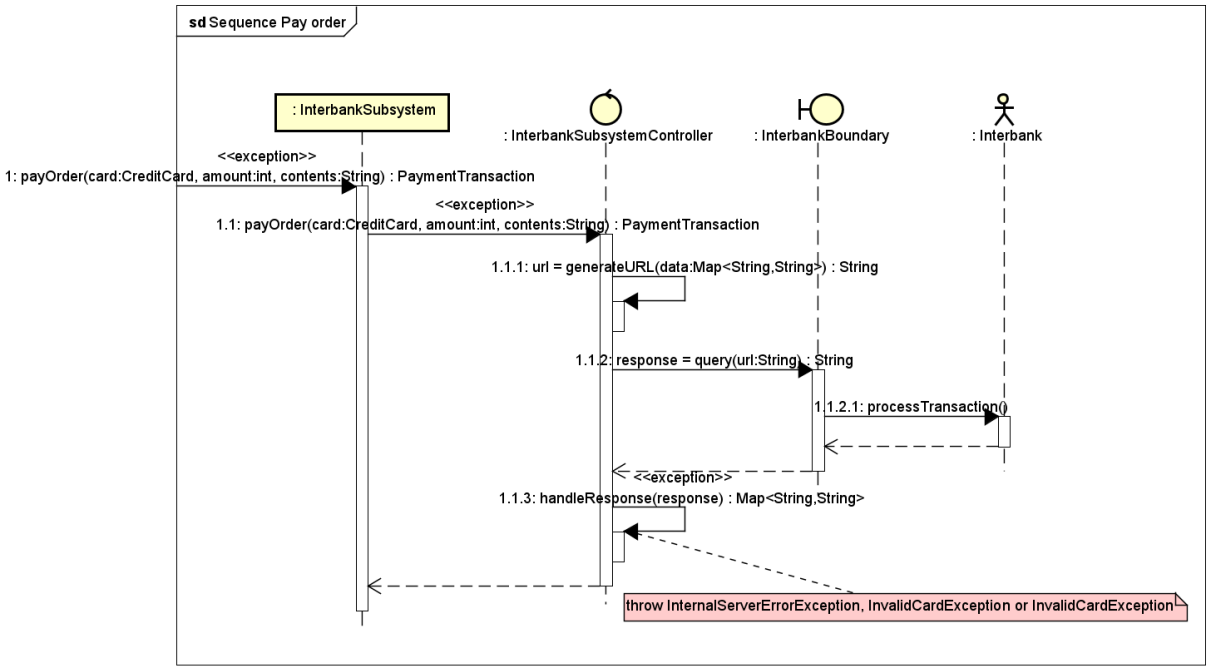
2.2.2. Identify subsystem interface

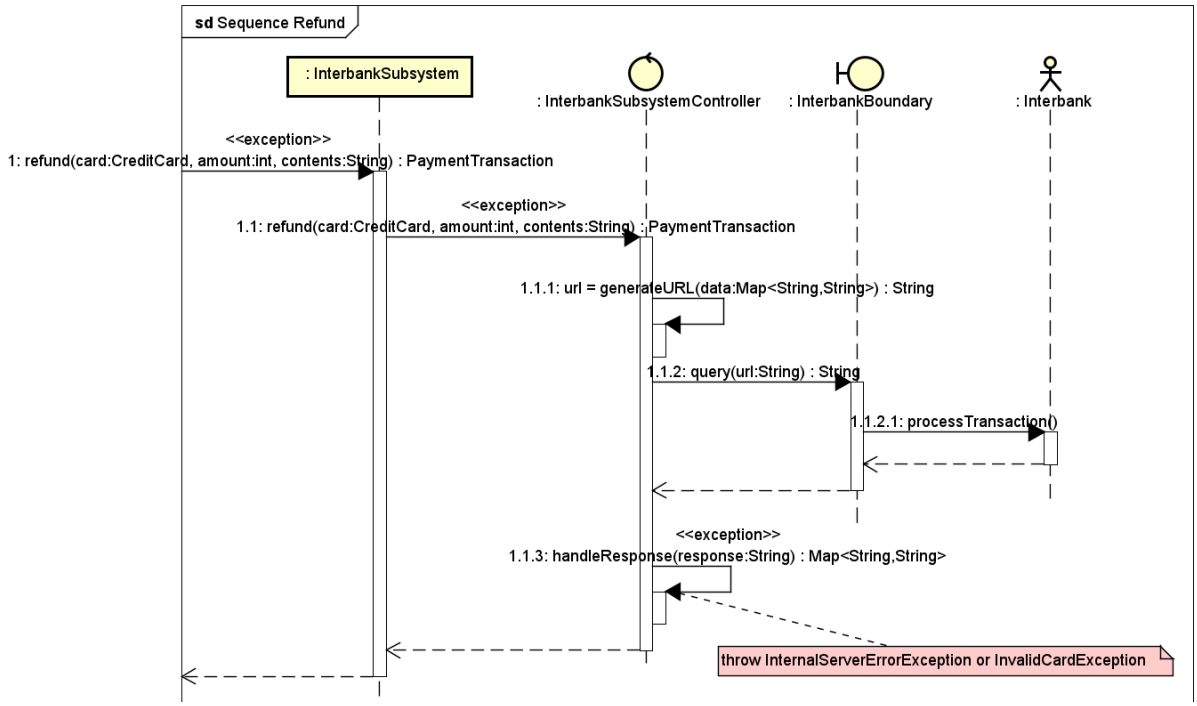
Based on the responsibilities of a payment system, we can identify the interface for our subsystem as follows.

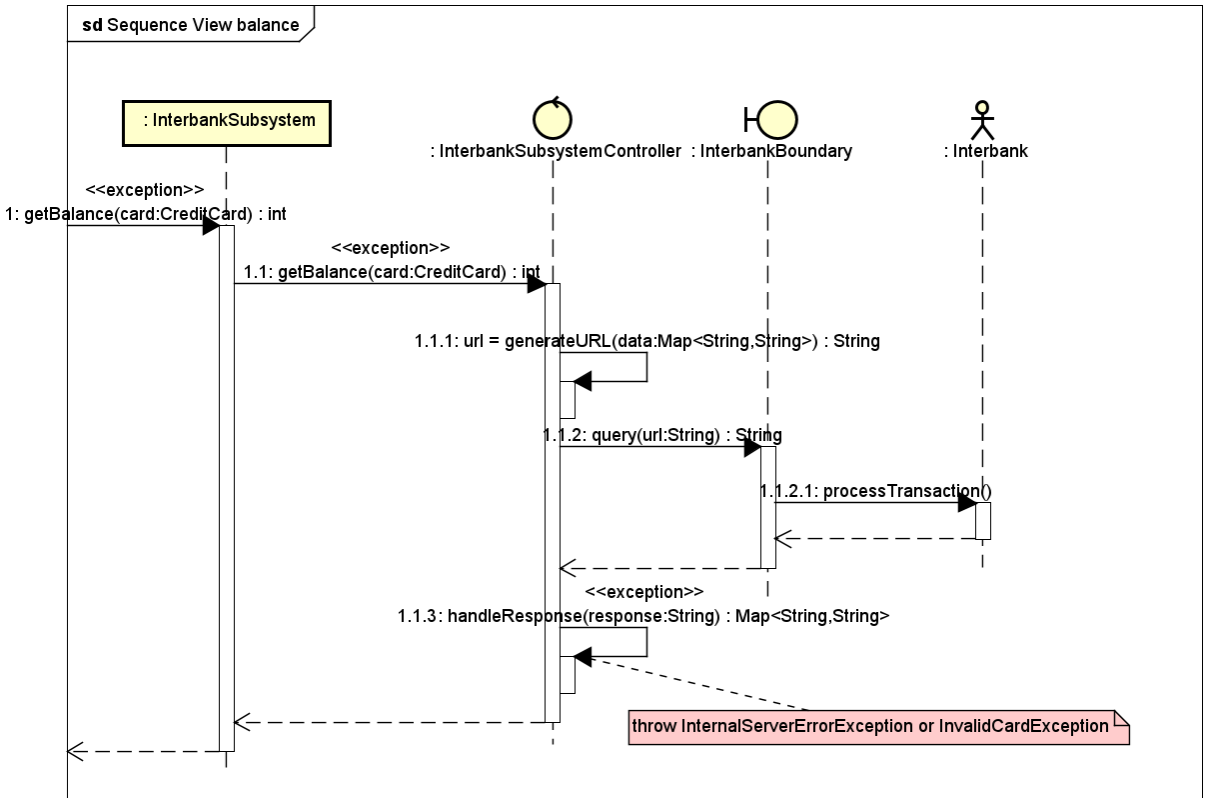


2.2.3. Subsystem design

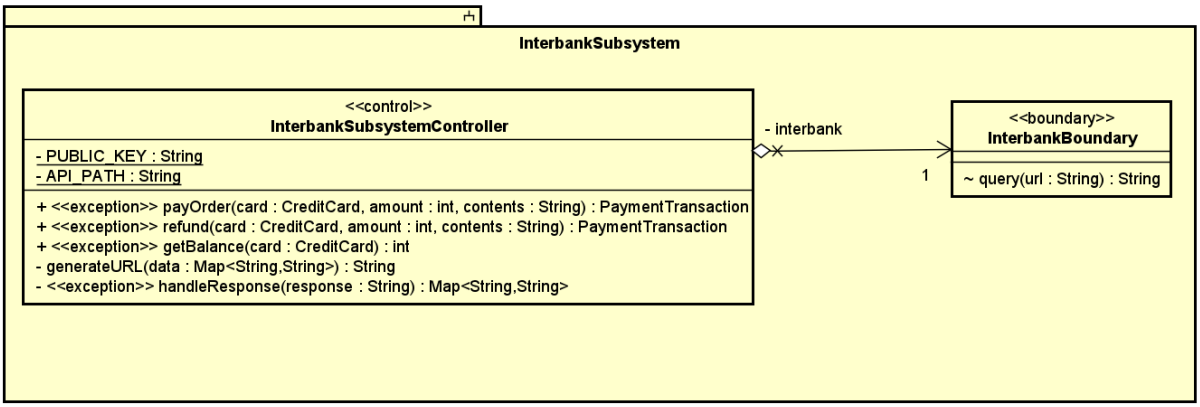
Distribute subsystem behavior to subsystem elements



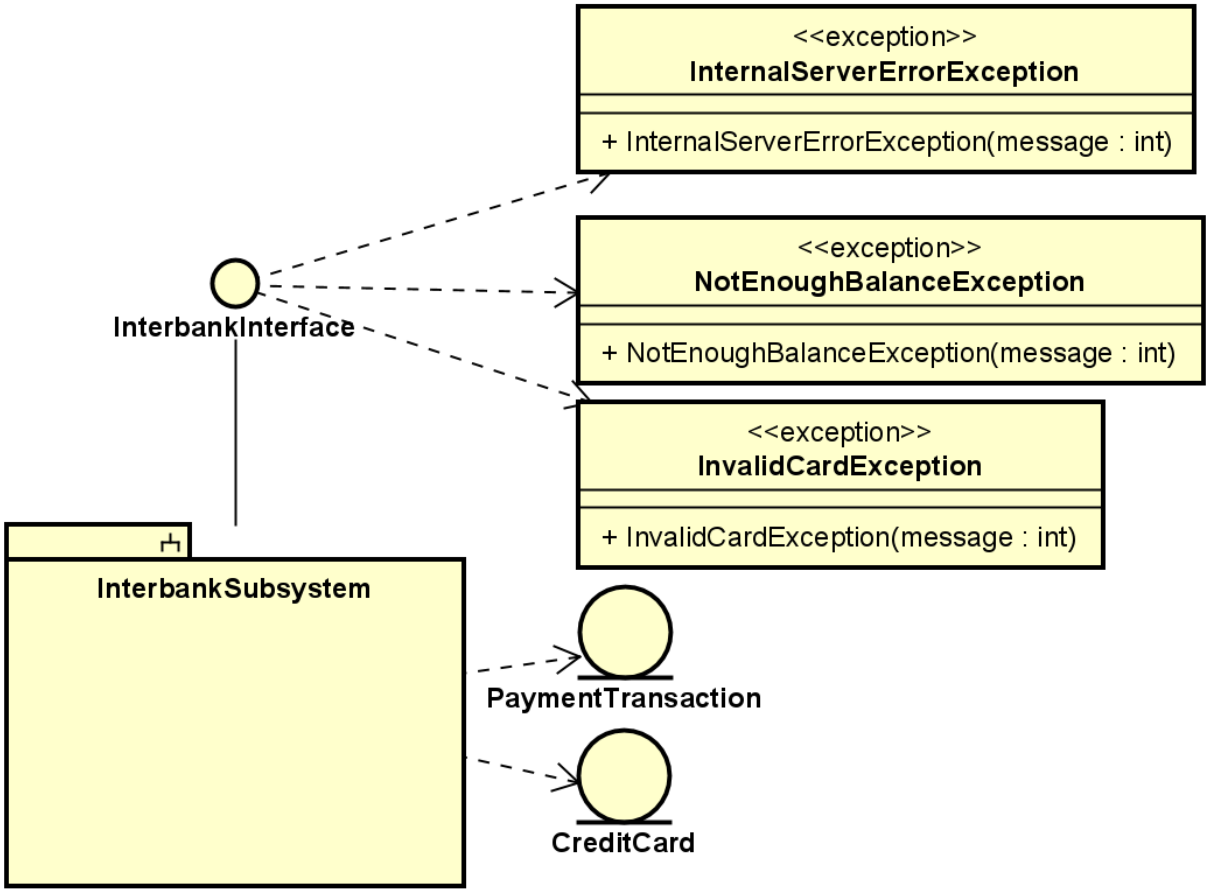

sd Sequence Refund
: InterbankSubsystem
: InterbankSubsystemController
: InterbankBoundary
: Interbank
<<exception>>
1: refund(card:CreditCard, amount:int, contents:String) : PaymentTransaction
<<exception>>
1.1: refund(card:CreditCard, amount:int, contents:String) : PaymentTransaction
1.1.1: url = generateURL(data:Map<String,String>) : String
1.1.2: query(url:String) : String
1.1.2.1: processTransaction()
<<exception>>
1.1.3: handleResponse(response:String) : Map<String,String>
throw InternalServerErrorException or InvalidCardException


sd Sequence View balance
: InterbankSubsystem
: InterbankSubsystemController
: InterbankBoundary
: Interbank
<<exception>>
1: getBalance(card:CreditCard) : int
<<exception>>
1.1: getBalance(card:CreditCard) : int
1.1.1: url = generateURL(data:Map<String,String>) : String
1.1.2: query(url:String) : String
1.1.2.1: processTransaction()
<<exception>>
1.1.3: handleResponse(response:String) : Map<String,String>
throw InternalServerErrorException or InvalidCardException

Document subsystem elements



Describe subsystem dependencies



Checkpoints