

Sample Test Project

Mobile Applications Development

Module D Development & Testing

Day 4

Contents

Contents	2
Introduction	3
Description of project and tasks	3
Instructions to the Competitor	11
Automated Test Guide	12

Introduction

As the organizer of the WorldSkills Competition, WorldSkills International is looking forward to the grand event being held in France. As the capital of gastronomy, France boasts a rich and diverse culinary culture, whose unique flavors attract food enthusiasts from all over the world. Therefore, we have decided to develop an application related to French cuisine, aiming at providing the comprehensive culinary experience for both competitors and visitors.

In this module, you need to develop the basic App functional logic, and write automated test scripts to run the App.

Description of project and tasks

You should push a full project and an installable apk (Android) or app (iOS) package to Git Server.

Competition time:

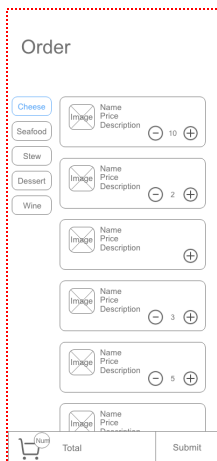
- 3 hours

Marking Emulator:

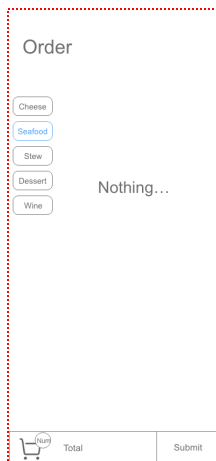
- iOS: iPhone 8 Plus
- Android: Pixel 2

Application wireframes

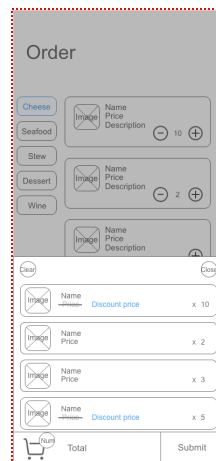
The App wireframes for reference only. You do not need to copy the same appearance, colors, decorative icons/images, or elements' positions/style.



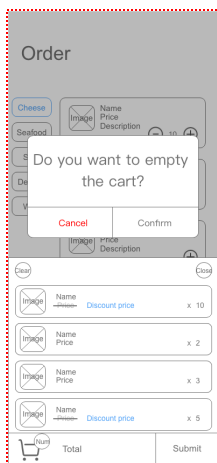
Order page



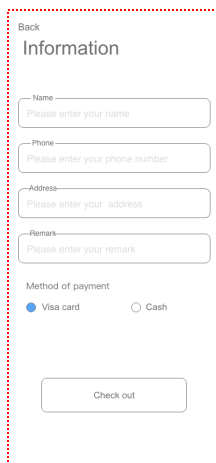
Order(no data) page



Cart page



Cart(dialog box) page



Information page



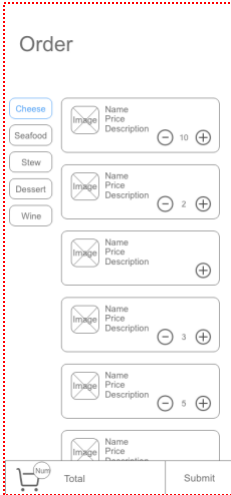
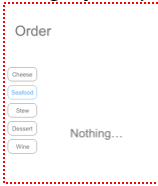
Payment page

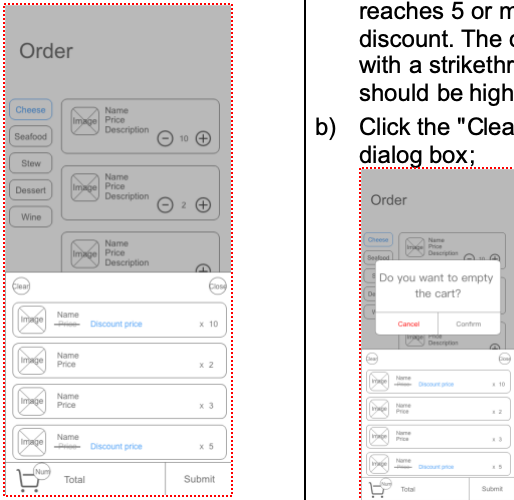
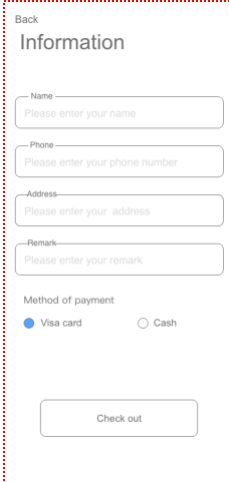
General Demands


1. All data which you need are in **media-files/json**.
2. Your application functions should develop by following all test cases (Form 1.1/1.2) and automated function test case (Form 2.1).

1. Develop by Test Case

Develop the application by understanding the Test case.

Form 1.1: Application UI Test Case		
No.	Area	Demands
1	The Application	The application should always be portrait mode.
2	Order page 	<p>a) When the quantity of items is less than 1, hide the decrease button and the quantity;</p> <p>b) Click the shopping cart button, if the total quantity of items is greater than 0, the shopping cart will be presented;</p> <p>c) Dynamically calculate the total price of items at the bottom and the total quantity of items. If the total quantity of items is 0, hide the total quantity icon;</p> <p>d) Click the "Submit" button to check if there are any items in the shopping cart. If there are items, navigate to the Information page;</p> <p>e) If there is no product data under a specific category, display "Nothing...";</p>  <p>f) Highlight the selected category.</p>

3	<p>Cart page</p> 	<ol style="list-style-type: none"> When the quantity of the same item added reaches 5 or more, the price is eligible for a 10% discount. The original price should be displayed with a strikethrough, and the discounted price should be highlighted; Click the "Clear" button to pop up a confirmation dialog box; Click "Confirm" in the dialog box to empty the shopping cart, close the shopping cart and clear all selected items in Order page; Click the "Close" button to close the shopping cart.
4	<p>Information page</p> 	<ol style="list-style-type: none"> Either method of payment can be selected; When the conditions(refer to Form 1.2: Application Test Case) are met, click the "Check out" button to navigate to Payment page; Click the back button to back to Order page.

5	<p>Payment page</p> 	<ul style="list-style-type: none"> a) The image shows the icon from the file "icon_finish.png"; b) Click the "Confirm" button to back to Order page and clear all selected items in Order page; c) Click the back button to back to Information page.
---	--	--

Form 1.2: Application Test Case					
No.	Area	Demands			
1	<div><div>Back</div><div>Information</div><div><div><div>Name</div><div>Please enter your name</div></div><div><div>Phone</div><div>Please enter your phone number</div></div><div><div>Address</div><div>Please enter your address</div></div><div><div>Remark</div><div>Please enter your remark</div></div><div><div>Method of payment</div><div><div><div>Visa card</div></div><div><div>Cash</div></div></div><div><div>Check out</div></div></div></div></div>		Trigger Timing	Constraints Type	Constraints Description
		Name input box	Click the Check out Button	length	[5, 10]
				type	string
				validity	required, not blank ^{*1}
		Phone input box	Click the Check out Button	length	10
				type	number
				validity	required, not blank ^{*1}
		Address input box	Click the Check out Button	length	[10, 50]
				type	string
				validity	required, not blank ^{*1}
Click the Check out button to show a notice when the input does not satisfy the conditions;					

^{*1}: Not blank means a char sequence is not empty and contains some characters except of whitespace characters.

2. Automated Test

Finish the automated test script according to the test project description.

It is noted that the test script files must be stored in the location according to the **Automated Test Guide**.

All steps need to be held on for at least 5 seconds, and the "Step No: \$Steps_No" should be logged on the "output" area, such as terminal, debug area or console.

Form 2.1: Application Function Test Case			
Test Type		Function Test	
Test method		Black Box Test, Automated Test	
Test demand		The workflow of the application functions runs normally.	
Test Execution Steps			
Steps_No.	Description of the Action	Input Data	Expected Results
1	Application Startup		Application starts normally
2	Click the third item in the category		When the category is selected, the product list displays the correct items
3	Slide the product list and find the tenth item		The product list is scrolled to the tenth item
4	Click the increase button for the tenth item three times		The quantity of the items is displayed as 3, and a decrease button presents
5	Click the fourth item in the category		When the category is selected, the product list displays the correct items
6	Click the increase button for the first item six times		The quantity of the items is displayed as 6, and a decrease button presents
7	Click the decrease button for the first item once		The quantity of the items is displayed as 5
8	Click the shopping cart button		The shopping cart appears, and the item with a quantity of 5 displays the discounted price
9	Click the "Submit" button		Navigate to Information page

10	Enter text in the Name input box	"Mary"	"Mary" is entered into the input box
11	Enter text into the Phone input box	"0612345678"	"0612345678" is entered into the input box
12	Enter text into the Address input box	"128 Rue Franklin, Lyon"	"128 Rue Franklin, Lyon" is entered into the input box
13	Click the "Cash" radio button		The cash radio button is selected
14	Click the "Check out" button		Navigate to Payment page
15	Click the "Confirm" button		Back to Order page and clear all selected items

Instructions to the Competitor

1. Please create the project with correct package name (Android)/ Organization Identifier (iOS). Naming convention: edu.ws2024.dXX;
2. You should save the project in the folder: XX_Module_D;
3. You should rename the generated apk (Android)/ app (iOS) file as XX_Module_D.apk (Android)/ XX_Module_D.app (iOS), and save it in the root of the XX_Module_D folder;
4. The whole XX_Module_D folder should be pushed to the provided remote Git repository;
5. Note: XX is your workstation code.

Automated Test Guide

Automated test dependencies configuration guide and scripts storage demands:

- If you use native Android platform, you should use "UI Automator" to finish the automated test scripts. You must write your UI test scripts entrance method "startTesting" in ApplicationUITesting class in androidTest directory. You must make sure that UI test in Android Studio can be started by clicking the start run button of the method: "startTesting".

Before building your UI test with UI Automator, make sure to configure your test source code location and project dependencies. In the build.gradle file of your Android app module, you must set a dependency reference to the UI Automator library:

```
dependencies {
    ...
    androidTestImplementation 'androidx.test.uiautomator:uiautomator:2.2.0'
}
```

- If you use Flutter platforms, you should use 'flutter_test' and 'integration_test' packages to finish the automated test scripts. You must write your UI test scripts in integration_test/app_test.dart file. You must make sure that the UI test can be started by running the console command:

- 1."flutter pub get --offline";
- 2."flutter test --no-pub integration_test/app_test.dart".

Before building your UI test with flutter_test and integration_test packages, make sure to configure your project dependencies. In the pubspec.yaml file of your flutter module, you must set dependencies reference to the flutter_test and integration_test packages:

```
dev_dependencies:
  flutter_test:
    sdk: flutter
  integration_test:
    sdk: flutter
```

- If you use native iOS platforms, you should finish the automated test scripts. During the marking process, the Experts will mark your UI test with the Xcode. You must make sure that the UI test in Xcode can be started by clicking the start run button of the method: "testStart" in "[Project Name]UITests.swift" or "[Project Name]UITests.m".

When you create a project, select the Include Unit Tests and Include UI Tests checkboxes that appear under the Language pop-up menu on the first sheet. These options preconfigure your project with [Project Name]Tests and [Project Name]UITests targets. In the Test navigator, you can view and edit the code for these tests, and add additional tests. Then run the UI tests and unit tests on simulated and connected devices.

