SC2006 – Software Engineering

Lab 4 Deliverables

Group 26

Content

I. Black Box Testing

1. AuthController

Control class to test - **AuthController**

An AuthController (short for Authentication Controller) is a crucial component of the application's controller layer responsible for handling authentication-related tasks. The AuthController typically manages customer login, registration, logout, password reset, and customer verification processes.

- **Customer Registration:** Handles customer sign-up by validating and saving new customer information to the database, ensuring security practices like password hashing are followed.
- **Customer Login:** Manages customer login by verifying the customer credentials (email and password) and issuing a session token or JSON Web Token (JWT) upon successful authentication.
- **Logout:** Logs the customer out by invalidating the customer session or token, typically by deleting or marking it as expired.
- **Password Reset:** Initiates password reset procedures, often involving sending a reset link or token to the user’s registered email address and allowing them to create a new password.
- **Email Verification:** Confirms the user's email address by sending a verification link, which the customer must click to activate their account.

This unique identifier becomes a key element in distinguishing and managing customer data. Additionally, if the role of the customer selected is Hawker, the system will require additional inputs such as Operating License, Operating Hours, Food Type.

2. Equivalence Class and Boundary Value Testing

Equivalence Class Testing

Equivalence Class Testing (ECT), also known as Equivalence Partitioning, is a Black Box Testing technique used in software testing to reduce the number of test cases. The goal of ECT is to divide input data into partitions, or equivalence classes, such that any test case within an equivalence class is expected to yield the same results. This method assumes that if one test case in a partition passes, then all other cases in that partition will also pass; similarly, if one fails, others in that partition are likely to fail.

- **Identify Input Data:** Determine the input domain for the software or system under test.
- **Partition the Input:** Divide the input data into equivalence classes. Each partition should represent a set of **valid** or **invalid** inputs.

- **Select Representatives:** Choose one representative from each equivalence class as the test case. This saves time and effort, as it is assumed that all values within a partition behave similarly.
- **Design Tests:** Create test cases using these representatives to cover all identified equivalence classes.

Valid equivalence classes describe valid situations, and the system should handle them normally.

Invalid equivalence classes describe invalid situations, and the system should reject them.

Boundary Value Testing

Boundary Value Testing (BVT), or Boundary Value Analysis, is a software testing technique focused on testing the boundaries of input ranges. The idea behind BVT is that errors often occur at the "edges" of input ranges, so testing the boundaries provides a higher chance of finding defects.

1. Login functions:
   Valid Equivalence Class: Emails and passwords with correct formats.
   Invalid equivalence Class: Emails and passwords with incorrect formats or missing information.
2. Sign- up functions:
   Valid Equivalence Class: Email, Password, and Confirm Password with correct formats
   Invalid Equivalence Class: Email, Password, and Confirm Password with incorrect formats or missing information
   If domain selected is a Hawker:
   Valid Equivalence Class: Email, Password, Confirm Password, and Operating Licences with correct formats
   Invalid Equivalence Class: Email, Password, Confirm Password, and Operating Licences with incorrect formats or missing information

3. Test Cases and Results

a. Login

Input parameters: Domain, Email and Password

No.	Test Input	Expected Output	Actual Output	Pass? (Y/N)
1	**(Valid) Domain:** "Customer" **(Valid)** Email: "halo@gmail.com" **(Valid)** Password: "111111aA"	"Log in successfully"	"Log in successfully"	Y
2	**(Invalid) Domain:** "" **(Valid)** Email: "halo@gmail.com" **(Valid)** Password: "111111aA"	"Invalid domain"	"Invalid domain"	Y
3	**(Valid) Domain:** "Customer" **(Invalid)** Email: "" **(Valid)** Password: "111111aA	"Please fill out this field"	"Please fill out this field"	Y
4	**(Valid) Domain:** "Customer" **(Valid)** Email: "halo@gmail.com" **(Invalid)** Password: ""	"Please fill out this field"	"Please fill out this field"	Y
5	**(Valid) Domain:** "Customer" **(Invalid)** Email: "halo" **(Valid)** Password: "111111aA	"Invalid email address"	"Invalid email address"	Y
6	**(Valid) Domain:** "Customer" **(Valid)** Email: "halo@gmail.com" **(Invalid)** Password: "1111aA"	"The password must be at least 8 characters long"	"The password must be at least 8 character long"	Y
7	**(Valid) Domain:** "Customer" **(Valid)** Email: "halo@gmail.com" **(Invalid)** Password: "111111aa"	"The password must contain at least one uppercase character, one lowercase character and one number"	"The password must contain at least one uppercase character, one lowercase character and one number"	Y
7	**(Valid) Domain:** "Customer"	"The password must contain at	"The password must contain at	Y

	(Valid) Email: “halo@gmail.com” **(Invalid)** Password: “111111AA”	least one uppercase character, one lowercase character and one number”	least one uppercase character, one lowercase character and one number”	
8	**(Valid) Domain:** “Customer” **(Valid)** Email: “halo@gmail.com” **(Invalid)** Password: “aaaaAAAA”	“The password must contain at least one uppercase character, one lowercase character and one number”	“The password must contain at least one uppercase character, one lowercase character and one number”	Y

b. Sign up

Input parameters: Domain, Email, Password, and Confirm Password

No.	Test Input	Expected Output	Actual Output	Pass? (Y/N)
1	**(All Valid Inputs)** Domain: “Customer” Email: halo@gmail.com Password: “111111aA” Confirm Password: “111111aA”	“Sign up successfully”	“Sign up successfully”	Y
2	**(All Valid Inputs except Domain)** Domain: “”	"Please select domain”	"Please select domain”	Y
3	**(All Valid Inputs except Email)** Email: “”	“Please fill out this field”	“Please fill out this field”	Y
4	**(All Valid Inputs except Email)** Email: “halo”	“Invalid email address”	“Invalid email address”	Y
5	**(All Valid Inputs except Password)** Password: “”	“Please fill out this field”	“Please fill out this field”	Y
6	**(All Valid Inputs except Password)** Password: “1111aA”	“The password must be at least 8 characters long”	“The password must be at least 8 character long”	Y
7	**(All Valid Inputs except Password)** Password: “111111aa”	“The password must contain at least one uppercase character, one lowercase character and one number”	“The password must contain at least one uppercase character, one lowercase character and one number”	Y
8	**(All Valid Inputs except Password)** Password: “111111AA”	“The password must contain at least one uppercase character, one lowercase character and one number”	“The password must contain at least one uppercase character, one lowercase character and one number”	

9	**(All Valid Inputs except Password)** Password: “aaaaAAAA”	“The password must contain at least one uppercase character, one lowercase character and one number”	“The password must contain at least one uppercase character, one lowercase character and one number”	Y
10	**(All Valid Inputs except Confirm Password)** Confirm Password: “”	“Please confirm password”	“Please confirm password”	Y
11	**(All Valid Inputs except Confirm Password)** Password: “111111aA” Confirm Password: “222222aA”	“Password mismatch!”	“Password mismatch!”	Y

c. Sign up – If domain selected is Hawker

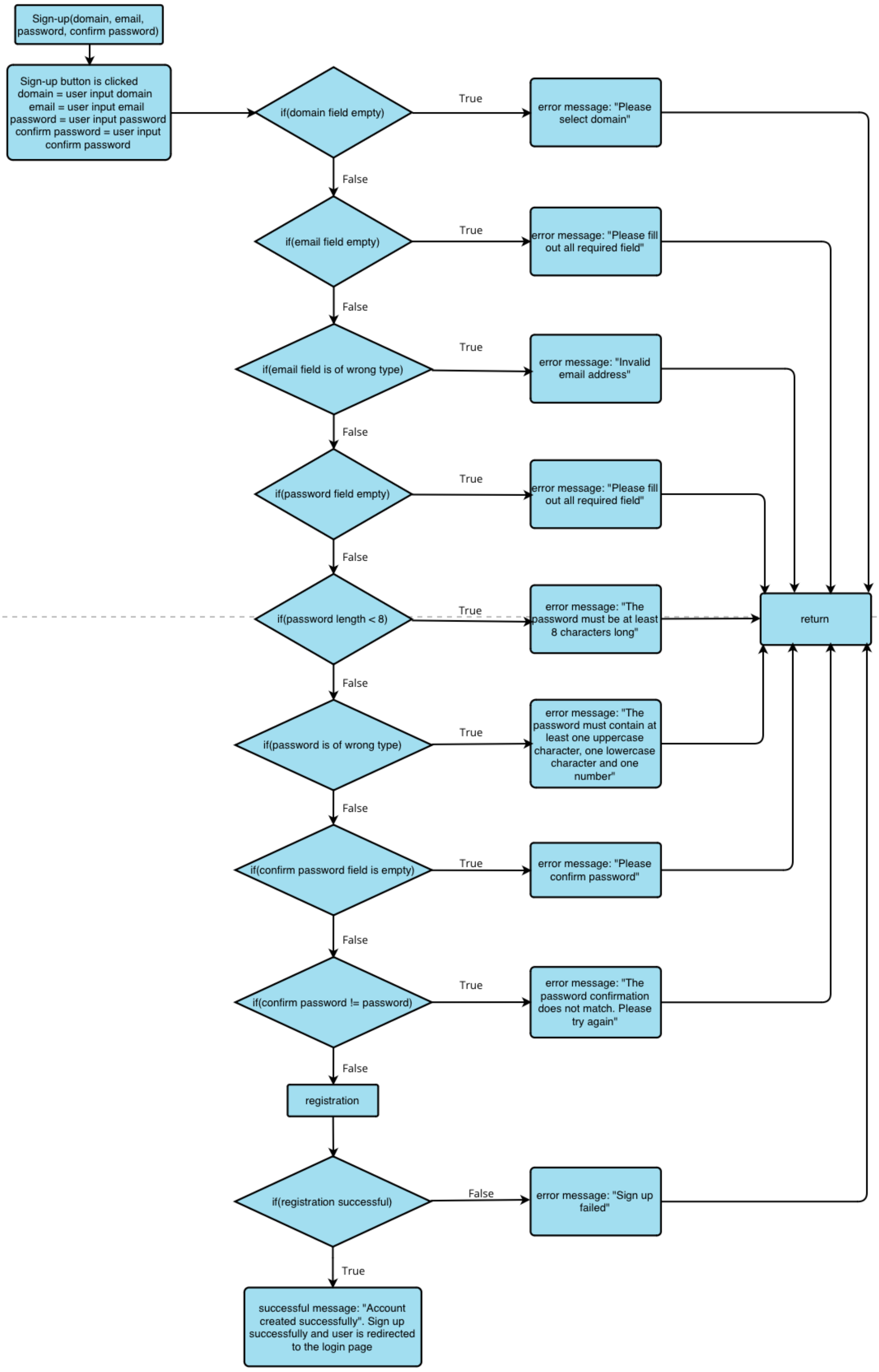
Input parameters: Domain, Email, Password, Confirm Password, and Licences

No.	Test Input	Expected Output	Actual Output	Pass? (Y/N)
1	**(All Valid Inputs)** Domain: “Hawker” Email: abc123@gmail.com Password: “Password123” Confirm Password: “Password123” Licence: “Submitted”	“Account created successfully”	“Account created successfully”	Y
2	**(All Valid Inputs except Licence)** Licence: “”	“Please upload a valid operating licence”	“Please upload a valid operating licence”	Y

II. White Box Testing

1. Create New Account (Customer)

a. Control Flow Path


Sign-up(domain, email, password, confirm password)
Sign-up button is clicked
domain = user input domain
email = user input email
password = user input password
confirm password = user input confirm password
if(domain field empty)
True
error message: "Please select domain"
False
if(email field empty)
True
error message: "Please fill out all required field"
False
if(email field is of wrong type)
True
error message: "Invalid email address"
False
if(password field empty)
True
error message: "Please fill out all required field"
False
if(password length < 8)
True
error message: "The password must be at least 8 characters long"
return
False
if(password is of wrong type)
True
error message: "The password must contain at least one uppercase character, one lowercase character and one number"
False
if(confirm password field is empty)
True
error message: "Please confirm password"
False
if(confirm password != password)
True
error message: "The password confirmation does not match. Please try again"
False
registration
if(registration successful)
False
error message: "Sign up failed"
True
successful message: "Account created successfully". Sign up successfully and user is redirected to the login page

b. Test Cases and Results

SignUp(email, password, confirmpwd)

No.	Test Input	Expected Output	Actual Output	Pass?

1	Email = “halo@gmail.com”
	Password = “111111aA”
	Confirmpwd = “111111aA”
		“Sign up successfully”	“Sign up successfully”	Y

2	Email = “halo”
	Password = “111111aA”
	Confirmpwd = “111111aA"
		“Please include an ‘@’ in the email address. ‘halo’ is missing an ‘@’”	“Please include an ‘@’ in the email address. ‘halo’ is missing an ‘@’”	Y

3	Email = halo@gmail.com
	Password = “111111aA”
	Confirmpwd = “222222aA”
		“Password mismatch!”	“Password mismatch!”	Y

4	Email = “”
	Password = “111111aA”
	Confirmpwd = “111111aA”
		“Please fill out this field”	“Please fill out this field”	Y

5	Email = “halo@gmail.com”
	Password = “”
	Confirmpwd = “111111aA"
		“Please fill out this field”	“Please fill out this field”	Y

6	Email = “halo@gmail.com”
	Password = “111111aA”
	Confirmpwd = “"
		“Please fill out this field”	“Please fill out this field”	Y

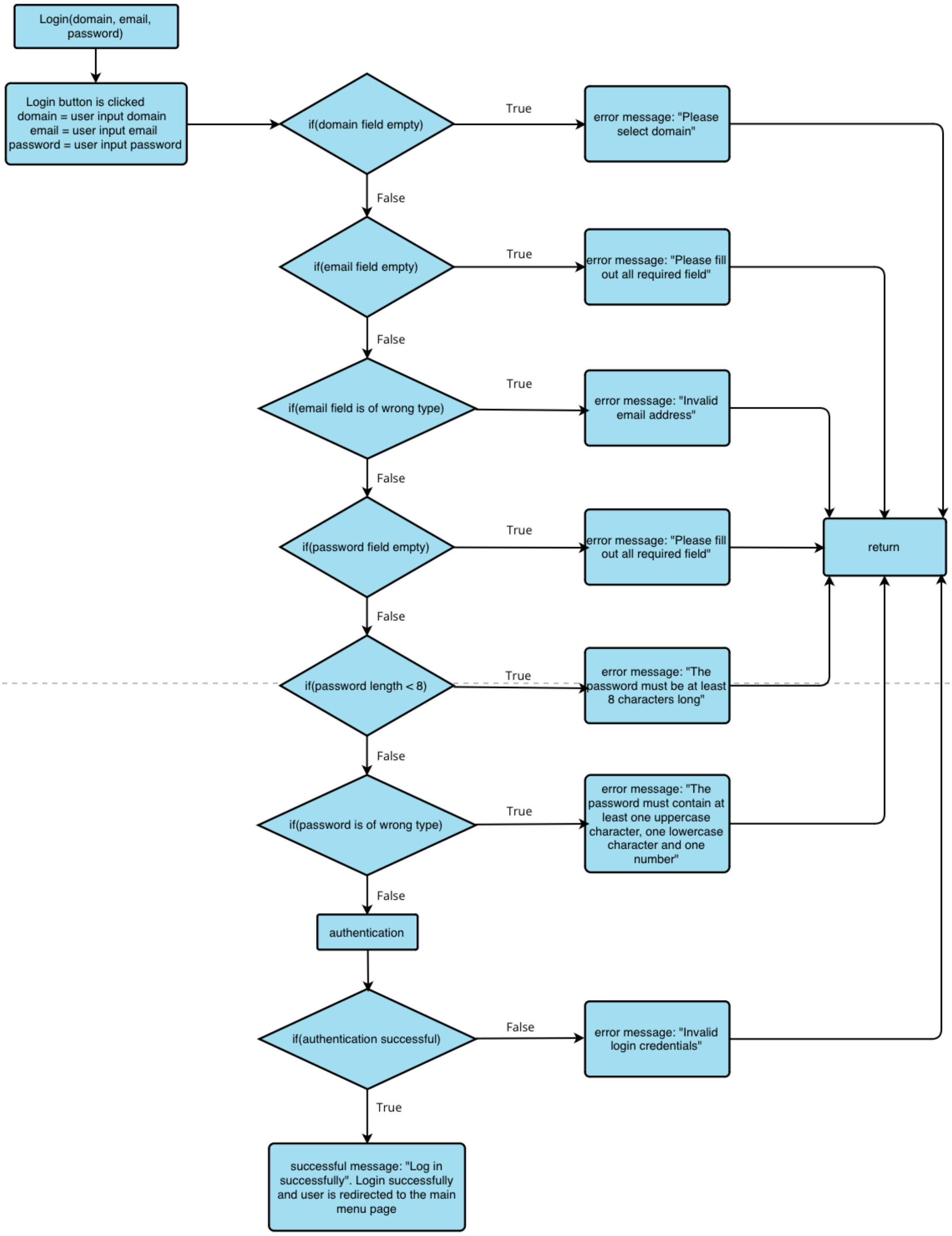
7	Email = “halo@gmail.com”
	Password = “1111aA”
	Confirmpwd = “1111aA"
		“Password must be at least 8 characters long”	“Password must be at least 8 characters long”	Y

8	Email = “halo@gmail.com”
	Password = “111111aa”
	Confirmpwd = “111111aa"
		“Password must contain both uppercase and lowercase letters”	“Password must contain both uppercase and lowercase letters”	Y

9	Email = *exsisting email*
	Password = “111111aA”
	Comfirmpwd = “111111aA”
		“Email exists, do you want to login?”	“Email exists, do you want to login?”	Y

2. Login

a. Control Flow Path


Login(domain, email, password)
Login button is clicked domain = user input domain email = user input email password = user input password
if(domain field empty)
True
error message: "Please select domain"
False
if(email field empty)
True
error message: "Please fill out all required field"
False
if(email field is of wrong type)
True
error message: "Invalid email address"
False
if(password field empty)
True
error message: "Please fill out all required field"
return
False
if(password length < 8)
True
error message: "The password must be at least 8 characters long"
False
if(password is of wrong type)
True
error message: "The password must contain at least one uppercase character, one lowercase character and one number"
False
authentication
if(authentication successful)
False
error message: "Invalid login credentials"
True
successful message: "Log in successfully". Login successfully and user is redirected to the main menu page

b. Test Cases and Results

login(domain, email, password)

No.	Test Input	Expected Output	Actual Output	Pass?

1	domain = “Customer”
	email = "halo@gmail.com”
	password = “111111aA”
		“Login successfully”	“Login successfully”	Y

2	domain = “Hawker”
	email = “hawker123@test.com”
	password = “Hawker123#”
		“Login successfully”	“Login successfully”	Y

3	domain = “”
	email = “halo@gmail.com”
	password = “111111aA”
		“Invalid domain”	“Invalid domain”	Y

4	domain = “Customer”
	email = “”
	password = “111111aA”
		“Please fill out this field”	“Please fill out this field”	Y

5	domain = “Customer”
	email = “halo@gmail.com”
	password = “”
		“Please fill out this field”	“Please fill out this field”	Y

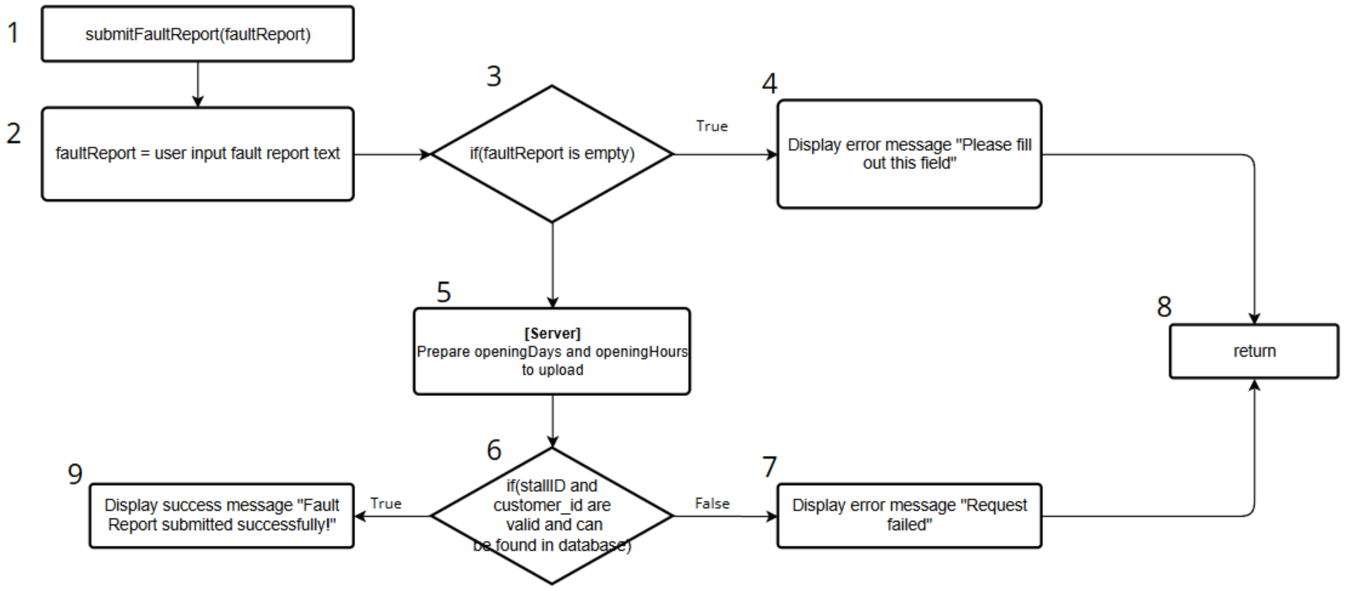
6	domain = “Customer”
	email = “halo”
	password = “111111aA”
		“Please include an ‘@’ in the email address. ‘halo’ is missing an ‘@’”	“Please include an ‘@’ in the email address. ‘halo’ is missing an ‘@’”	Y

7	domain = “Customer”
	email = “hawker123@test.com”
	password = “Hawker123#”
		“Invalid email or password”	“Invalid email or password”	Y

8	domain = “Hawker”
	email = “hawker123@test.com”
	password = “Hawker456#”
		“Invalid email or password”	“Invalid email or password”	Y

3. Customer Submit Fault Report

a. Control Flow Path


1
submitFaultReport(faultReport)
2
faultReport = user input fault report text
3
if(faultReport is empty)
True
4
Display error message "Please fill out this field"
5
[Server]
Prepare openingDays and openingHours to upload
8
return
6
if(stallID and customer_id are valid and can be found in database)
True
9
Display success message "Fault Report submitted successfully!"
False
7
Display error message "Request failed"

b. Test Cases and Results

submitFaultReport(faultReport)

- Input: faultReport
- Implied input: stallID (passed automatically by inferring from the current stall selected by customer), user_id (passed automatically by inferring from the current current logged in customer)

No.	Test Input	Expected Output	Actual Output	Pass?

1	faultReport = “Wrong opening hours”
	
	stallID = 4
	user_id = 1
		“Fault Report submitted successfully”	“Fault Report submitted successfully!”	Y

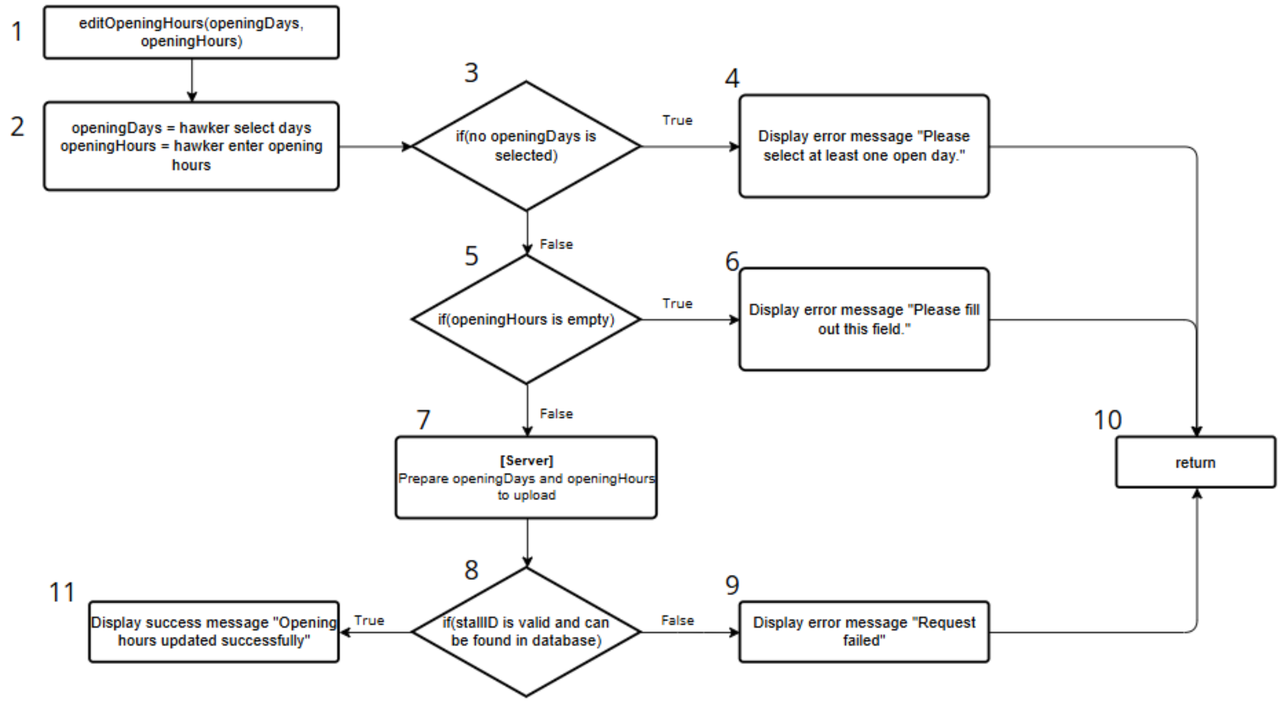
2	faultReport = “”
	
	stallID = 4
	user_id = 1
		“Please fill out this field”	“Please fill out this field”	Y

3	faultReport = “Wrong opening hours”
	
	stallID = -1 (invalid and not found in database)
	user_id = 1
		“Request Failed”	“Request failed”	Y

4	faultReport = “Wrong opening hours”
	
	stallID = 4
	user_id = -1 (invalid and not found in database)
		“Request Failed”	“Request Failed”	Y

4. Hawker Edit Opening Hours

a. Control Flow Path


1
editOpeningHours(openingDays, openingHours)
2
openingDays = hawker select days
openingHours = hawker enter opening hours
3
if(no openingDays is selected)
True
4
Display error message "Please select at least one open day."
False
5
if(openingHours is empty)
True
6
Display error message "Please fill out this field."
False
7
[Server]
Prepare openingDays and openingHours to upload
10
return
8
if(stallID is valid and can be found in database)
True
11
Display success message "Opening hours updated successfully"
False
9
Display error message "Request failed"

b. Test Cases and Results

editOpeningHours(openingDays, openingHours)

- Inputs: openingDays, openingHours
- Implied input: StallID (passed automatically by inferring from the current current logged in hawker)

No.	Test Input	Expected Output	Actual Output	Pass?
1	openingDays != “0000000” openingHours = “9AM - 9PM” StallID = 4	“Opening hours updated successfully.”	“Opening hours updated successfully.”	Y

2	openingDays = “0000000”
	openingHours = “9AM - 9PM”
	
	StallID = 4
		“Please select at least one open day.”	“Please select at least one open day."	Y

3	openingDays != “0000000”
	openingHours = “”
	
	StallID = 4
		“Please fill out this field”	“Please fill out this field”	Y

4	openingDays != “0000000” openingHours = “9AM - 9PM” StallID = -1(invalid and not found in database)	“Request Failed”	“Request Failed”	Y