Software Testing: Decision Table Testing & Path Testing

COURSE : SIT707

NAME: TOMIN JOSE (224353043)

DATE : 13/03/2024

Decision Table Testing

•A structured way to model complex logic via conditions and actions.
•Also called cause-effect table.
•Common in black box testing.
•Helps handle combinations of inputs and expected outputs.
Use case: Login validation, loan approval, user access systems

Email Login Decision Table

Email	Password	Expected Result
F	F	Error: Enter Email
T	F	Error: Enter Password
F	T	Error: Enter Email
T	T	Login Successful

- 2^n combinations tested clearly.
- Useful when different input combinations affect outcomes

Decision Table – Triangle Classification

Goal: Identify the triangle type based on 3 side lengths (a, b, c)

Conditions to Check

Condition No.	Check	Description
C1	a < b + c	Is side ‘a’ less than the sum of others?
C2	b < a + c	Same for side ‘b’
C3	c < a + b	Same for side ‘c’
C4	a == b	Are sides ‘a’ and ‘b’ equal?
C5	a == c	Are sides ‘a’ and ‘c’ equal?
C6	b == c	Are sides ‘b’ and ‘c’ equal?

Decision Table – Triangle Classification

Resulting Triangle Type

C1	C2	C3	C4	C5	C6	Triangle Type
F	–	–	–	–	–	Not a Triangle
T	T	T	F	F	F	Scalene
T	T	T	T	T	T	Equilateral
T	T	T	T	F	F	Isosceles

Decision Table – Triangle Classification

Examples

a	b	c	Type
1	2	3	Not a Triangle
3	3	3	Equilateral
4	4	5	Isosceles
3	4	5	Scalene

Real-World Use of Decision Table Testing

- Manages complex logic: When systems involve many conditions, a decision table maps them clearly.
- Ensures all combinations are considered – reducing risk of missed scenarios.

Real-World Scenarios:

Scenario	How Decision Table Helps
Login Form Validation	Checks various combos of email/password inputs
Loan Eligibility	Evaluates customer’s income, credit score, and history
E-commerce Checkout	Handles multiple promo codes, payment options, stock
Medical Device Alerts	Ensures correct action for patient vitals thresholds

Path Testing

- A structural (white-box) testing method
- Uses program graphs and DD-paths
- Ensures all logic branches/paths are executed

Real-world use: Loop testing, condition coverage in critical systems (e.g., payment processors)

Program Graphs & DD-Paths

- Nodes: statements or fragments
- Edges: control flow
- DD-Paths = Decision to Decision chains

Useful for measuring test coverage (C0, C1, C2, etc.)

McCabe’s Basis Path Testing

McCabe’s Basis Path Testing is a white-box testing technique that helps you:

- Understand the logic of your code
- Find all the unique paths (like routes through a maze)
- Design tests to cover every path at least once

Limitations:

- Too many paths with loops = infeasible
- Use heuristics or reduce complexity via condensation

Real-World Use of Path Testing

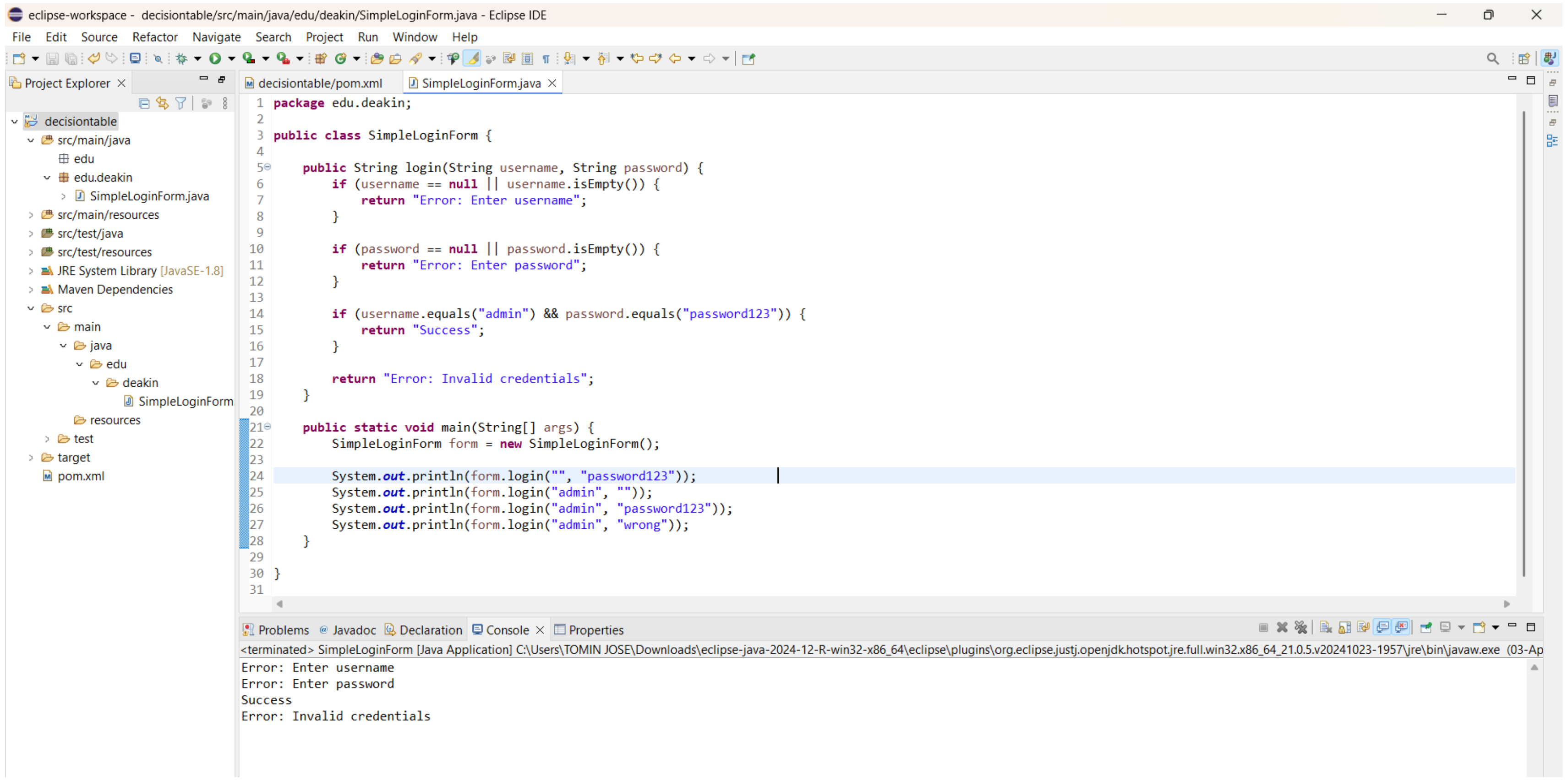
- Covers every logical route the program can take
- Detects logic errors in loops, branches, and decision points
- Improves code quality by exposing dead or unreachable code

Real-World Scenarios:

Scenario	How Path Testing Helps
Vehicle control system	Tests emergency handling logic inside loops
ATM transactions	Ensures each branch (e.g., withdrawal, balance check) is tested
Order fulfillment code	Verifies complex flow of stock, packing, shipping
Medical testing lab	Validates decision-making logic in testing machines

Evidence of active learning session.

Task 1: SimpleLoginForm



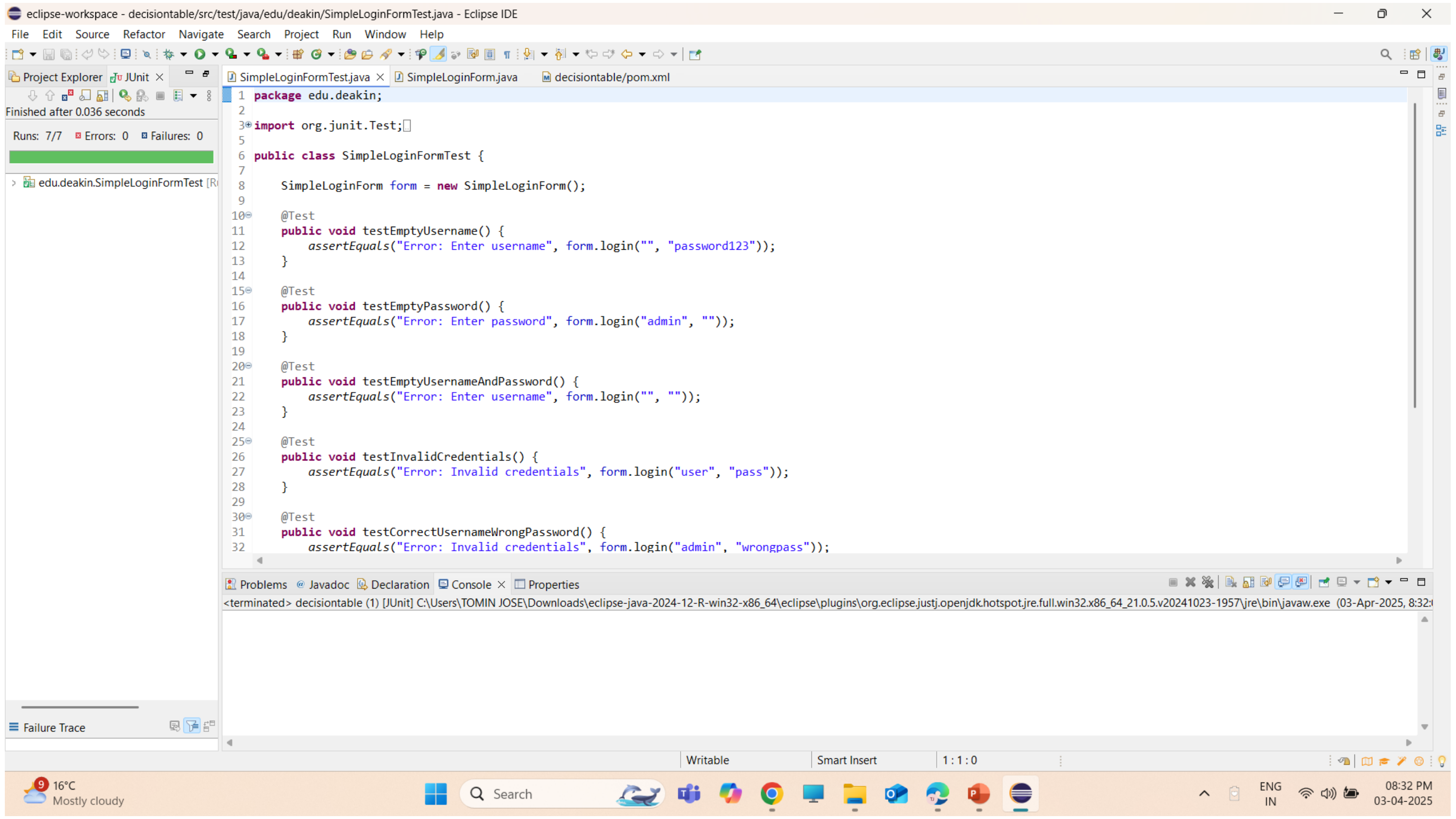
```java
package edu.deakin;

public class SimpleLoginForm {

    public String login(String username, String password) {
        if (username == null || username.isEmpty()) {
            return "Error: Enter username";
        }

        if (password == null || password.isEmpty()) {
            return "Error: Enter password";
        }

        if (username.equals("admin") && password.equals("password123")) {
            return "Success";
        }

        return "Error: Invalid credentials";
    }

    public static void main(String[] args) {
        SimpleLoginForm form = new SimpleLoginForm();

        System.out.println(form.login("", "password123"));
        System.out.println(form.login("admin", ""));
        System.out.println(form.login("admin", "password123"));
        System.out.println(form.login("admin", "wrong"));
    }

}
```

```
<terminated> SimpleLoginForm [Java Application] C:\Users\TOMIN JOSE\Downloads\eclipse-java-2024-12-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.5.v20241023-1957\jre\bin\javaw.exe (03-Ap
Error: Enter username
Error: Enter password
Success
Error: Invalid credentials
```

Decision Table for Test Design

Username	Password	Expected Result
(empty)	(any)	Error: Enter username
(any)	(empty)	Error: Enter password
(empty)	(empty)	Error: Enter username
wrongUsername	wrongPassword	Error: Invalid credentials
admin	wrongPassword	Error: Invalid credentials
wrongUsername	password123	Error: Invalid credentials
admin	password123	Success

Implement test cases



eclipse-workspace - decisiontable/src/test/java/edu/deakin/SimpleLoginFormTest.java - Eclipse IDE

Finished after 0.036 seconds

Runs: 7/7 Errors: 0 Failures: 0

edu.deakin.SimpleLoginFormTest

```java
package edu.deakin;

import org.junit.Test;

public class SimpleLoginFormTest {

    SimpleLoginForm form = new SimpleLoginForm();

    @Test
    public void testEmptyUsername() {
        assertEquals("Error: Enter username", form.login("", "password123"));
    }

    @Test
    public void testEmptyPassword() {
        assertEquals("Error: Enter password", form.login("admin", ""));
    }

    @Test
    public void testEmptyUsernameAndPassword() {
        assertEquals("Error: Enter username", form.login("", ""));
    }

    @Test
    public void testInvalidCredentials() {
        assertEquals("Error: Invalid credentials", form.login("user", "pass"));
    }

    @Test
    public void testCorrectUsernameWrongPassword() {
        assertEquals("Error: Invalid credentials", form.login("admin", "wrongpass"));
```

```
<terminated> decisiontable (1) [JUnit] C:\Users\TOMIN JOSE\Downloads\eclipse-java-2024-12-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.5.v20241023-1957\jre\bin\javaw.exe (03-Apr-2025, 8:32:
```

Generate test cases using ChatGPT

The test cases I wrote for the SimpleLoginForm class cover all the essential scenarios such as empty fields, invalid credentials, and successful login. However, compared to the test cases generated by ChatGPT, mine are slightly less comprehensive. ChatGPT's version includes additional checks for null values in both username and password fields, which adds an extra layer of robustness to the test suite. While both versions effectively validate the core functionality, ChatGPT's tests provide better edge case coverage, making the application more reliable against unexpected input.

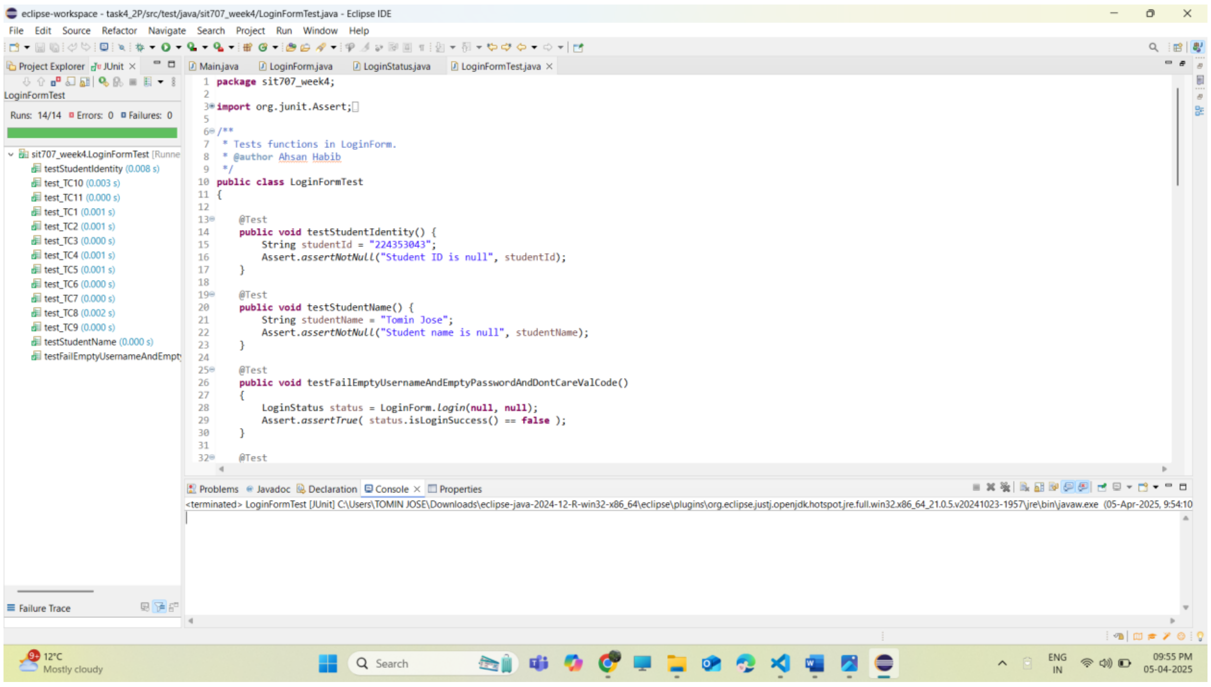
Thank You

Git: https://github.com/TOMINJOSE88/sit707.git

(b) Decision table testing using JUnit

Test	Username	Password	Code	Expected Result
TC1	-	-	X	Login fails: missing username/password
TC2	-	W	X	Login fails: missing username
TC3	-	C	X	Login fails: missing username
TC4	W	-	X	Login fails: missing password
TC5	W	W	X	Login fails: wrong username/password
TC6	W	C	X	Login fails: wrong username/password
TC7	C	-	X	Login fails: missing password
TC8	C	W	X	Login fails: wrong username/password
TC9	C	C	-	Login succeeds; code validation fails
TC10	C	C	W	Login succeeds; code validation fails
TC11	C	C	C	Login succeeds; code validation passes

• A screenshot of your Eclipse IDE's (i) JUnit tab which shows test statistics including Runs, Errors and Failures and (ii) Eclipse console which shows outputs.


eclipse-workspace - task4_2P/src/test/java/sit707_week4/LoginFormTest.java - Eclipse IDE

LoginFormTest

Runs: 14/14 Errors: 0 Failures: 0

```
sit707_week4.LoginFormTest [Runne
    testStudentIdentity (0.008 s)
    test_TC10 (0.003 s)
    test_TC11 (0.000 s)
    test_TC1 (0.001 s)
    test_TC2 (0.001 s)
    test_TC3 (0.000 s)
    test_TC4 (0.001 s)
    test_TC5 (0.001 s)
    test_TC6 (0.000 s)
    test_TC7 (0.000 s)
    test_TC8 (0.002 s)
    test_TC9 (0.000 s)
    testStudentName (0.000 s)
    testFailEmptyUsernameAndEmpt
```

```java
package sit707_week4;

import org.junit.Assert;

/**
 * Tests functions in LoginForm.
 * @author Ahsan Habib
 */
public class LoginFormTest
{

    @Test
    public void testStudentIdentity() {
        String studentId = "224353043";
        Assert.assertNotNull("Student ID is null", studentId);
    }

    @Test
    public void testStudentName() {
        String studentName = "Tomin Jose";
        Assert.assertNotNull("Student name is null", studentName);
    }

    @Test
    public void testFailEmptyUsernameAndEmptyPasswordAndDontCareValCode()
    {
        LoginStatus status = LoginForm.login(null, null);
        Assert.assertTrue( status.isLoginSuccess() == false );
    }

    @Test
```

```
<terminated> LoginFormTest [JUnit] C:\Users\TOMIN JOSE\Downloads\eclipse-java-2024-12-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.5.v20241023-1957\jre\bin\javaw.exe (05-Apr-2025, 9:54:10
```

• **Your program's source code for tests (LoginFormTest.java)**

LoginFormTest.java:

```
package sit707_week4;


import org.junit.Assert;

import org.junit.Test;


/**

 * Tests functions in LoginForm.

 * @author Ahsan Habib

 */

public class LoginFormTest

{


	@Test

	public void testStudentIdentity() {

		String studentId = "224353043";

		Assert.assertNotNull("Student ID is null", studentId);

	}


	@Test

	public void testStudentName() {

		String studentName = "Tomin Jose";

		Assert.assertNotNull("Student name is null", studentName);

	}


	@Test
```

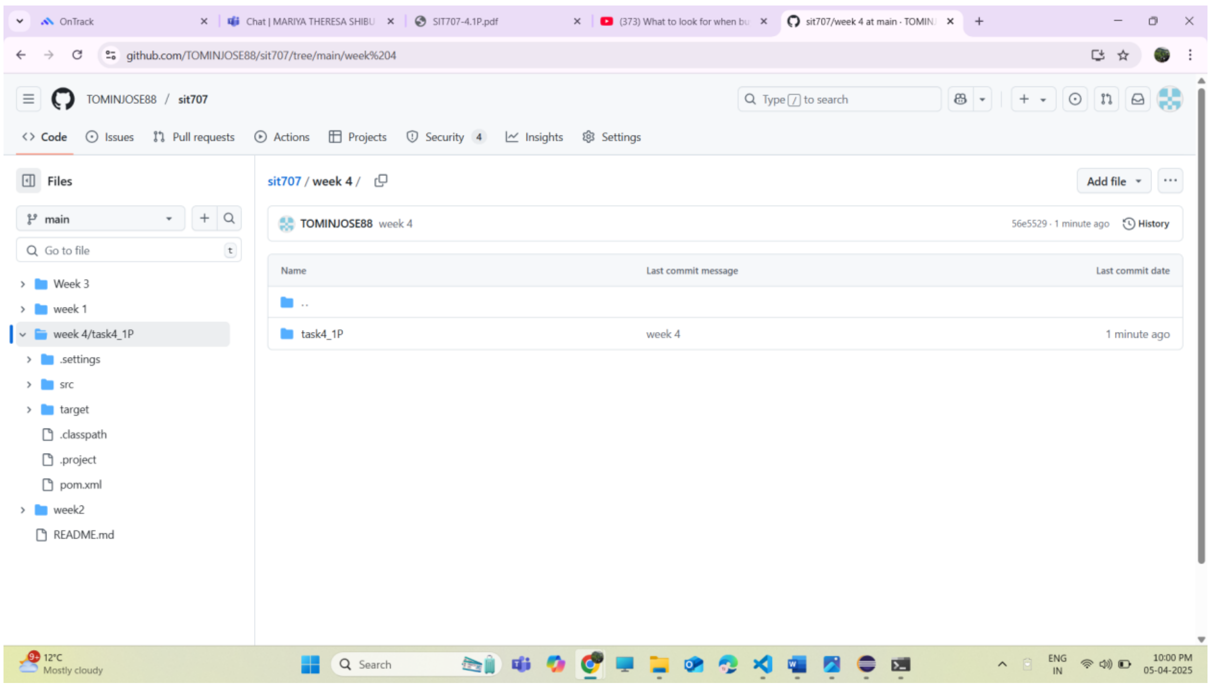
```
public void testFailEmptyUsernameAndEmptyPasswordAndDontCareValCode()
{
            LoginStatus status = LoginForm.login(null, null);
            Assert.assertTrue( status.isLoginSuccess() == false );
}


    @Test
    public void test_TC1() {
      LoginStatus status = LoginForm.login(null, null);
      Assert.assertFalse(status.isLoginSuccess());
      Assert.assertEquals("Empty Username", status.getErrorMsg());
    }

    @Test
    public void test_TC2() {
      LoginStatus status = LoginForm.login(null, "wrong_pass");
      Assert.assertFalse(status.isLoginSuccess());
      Assert.assertEquals("Empty Username", status.getErrorMsg());
    }

    @Test
    public void test_TC3() {
      LoginStatus status = LoginForm.login(null, "tomin_pass");
      Assert.assertFalse(status.isLoginSuccess());
      Assert.assertEquals("Empty Username", status.getErrorMsg());
    }

    @Test
```

```
public void test_TC4() {
  LoginStatus status = LoginForm.login("wrong_user", null);
  Assert.assertFalse(status.isLoginSuccess());
  Assert.assertEquals("Empty Password", status.getErrorMsg());
}

@Test
public void test_TC5() {
  LoginStatus status = LoginForm.login("wrong_user", "wrong_pass");
  Assert.assertFalse(status.isLoginSuccess());
  Assert.assertEquals("Credential mismatch", status.getErrorMsg());
}

@Test
public void test_TC6() {
  LoginStatus status = LoginForm.login("wrong_user", "tomin_pass");
  Assert.assertFalse(status.isLoginSuccess());
  Assert.assertEquals("Credential mismatch", status.getErrorMsg());
}

@Test
public void test_TC7() {
  LoginStatus status = LoginForm.login("tomin", null);
  Assert.assertFalse(status.isLoginSuccess());
  Assert.assertEquals("Empty Password", status.getErrorMsg());
}

@Test
```

```
public void test_TC8() {
  LoginStatus status = LoginForm.login("tomin", "wrong_pass");
  Assert.assertFalse(status.isLoginSuccess());
  Assert.assertEquals("Credential mismatch", status.getErrorMsg());
}

@Test
public void test_TC9() {
  LoginStatus status = LoginForm.login("tomin", "tomin_pass");
  Assert.assertTrue(status.isLoginSuccess());
  boolean codeValidation = LoginForm.validateCode(null);
  Assert.assertFalse(codeValidation);
}

@Test
public void test_TC10() {
  LoginStatus status = LoginForm.login("tomin", "tomin_pass");
  Assert.assertTrue(status.isLoginSuccess());
  boolean codeValidation = LoginForm.validateCode("wrong_code");
  Assert.assertFalse(codeValidation);
}

@Test
public void test_TC11() {
  LoginStatus status = LoginForm.login("tomin", "tomin_pass");
  Assert.assertTrue(status.isLoginSuccess());
  boolean codeValidation = LoginForm.validateCode("123456");
  Assert.assertTrue(codeValidation);
```

```
        }




}
```

- **A screenshot of your GitHub page where your latest project folder is pushed.**



sit707 / week 4 /

TOMINJOSE88 week 4 56e5529 · 1 minute ago

- Week 3
- week 1
- week 4/task4_1P
  - .settings
  - src
  - target
  - .classpath
  - .project
  - pom.xml
- week2
- README.md

Name	Last commit message	Last commit date
..		
task4_1P	week 4	1 minute ago