Detail Test

Our group chooses **Use Information Management** module to test.

Introduction

Black-box testing

1. Draw state transform diagram and get the scenario;
2. Use EP and BVA to furtherly refine [valid] and [invalid].
3. Use decision tables to determine the combination of [valid] and [invalid].

White-box testing

1. Static Codes testing (SonarQube)
2. Junit for backend part

Test Tool Implementation

- Junit
- SonarQube
- Postman

Junit

JUnit is a Java testing framework that makes it easy to write reliable and efficient tests. It can be used for applications made in most languages, but is particularly well suited for testing Java applications.JUnit can also be used to create automated tests.

The JUnit framework is one of the most popular Java testing frameworks. It provides several features that make writing tests easy, including support for multiple test cases, assertions, and reports. JUnit is also versatile, allowing tests to be written in a variety of languages.

It allows you to create and run tests efficiently and has become one of the most popular Java testing frameworks. xUnit framework inspired JUnit for Smalltalk and C++. Since JUnit is a member of the xUnit family of testing frameworks, it is designed to support different tests, including unit, functional and integration tests.

JUnit is primarily used for unit testing, but it can also be used for other tests such as functional and integration tests. Functional tests test the functionality of a system. They are different from unit tests because they test the entire system rather than individual units. Integration tests test the integration of two or more systems. They are different from unit tests because they test how the components of the system work together, rather than individually.

SonarQube

SonarQube is an open-source platform that provides continuous code quality management and code analysis tools. It is designed to help developers and teams assess and improve the quality of their codebase. SonarQube supports various programming languages and provides a comprehensive set of features to detect bugs, vulnerabilities, code smells, and other code quality issues. It analyzes code metrics, enforces coding standards, and generates detailed reports and visualizations to track the quality and maintainability of software projects. SonarQube integrates with popular build systems and CI/CD pipelines, enabling developers to automate code analysis and ensure that code quality is monitored continuously throughout the development process.

Postman

Postman is a popular collaboration platform for API development. It provides a user-friendly interface for sending HTTP requests, testing APIs, and building API workflows. Postman allows developers to create and organize collections of requests, define request headers and parameters, and inspect API responses. It supports various request types such as GET, POST, PUT, DELETE, and more. Postman also includes features for authentication, handling cookies, and working with environments to facilitate testing across different configurations. Additionally, Postman offers capabilities for writing automated tests and generating API documentation. It is widely used by developers and teams to streamline API development, testing, and documentation processes, ensuring the reliability and correctness of APIs.

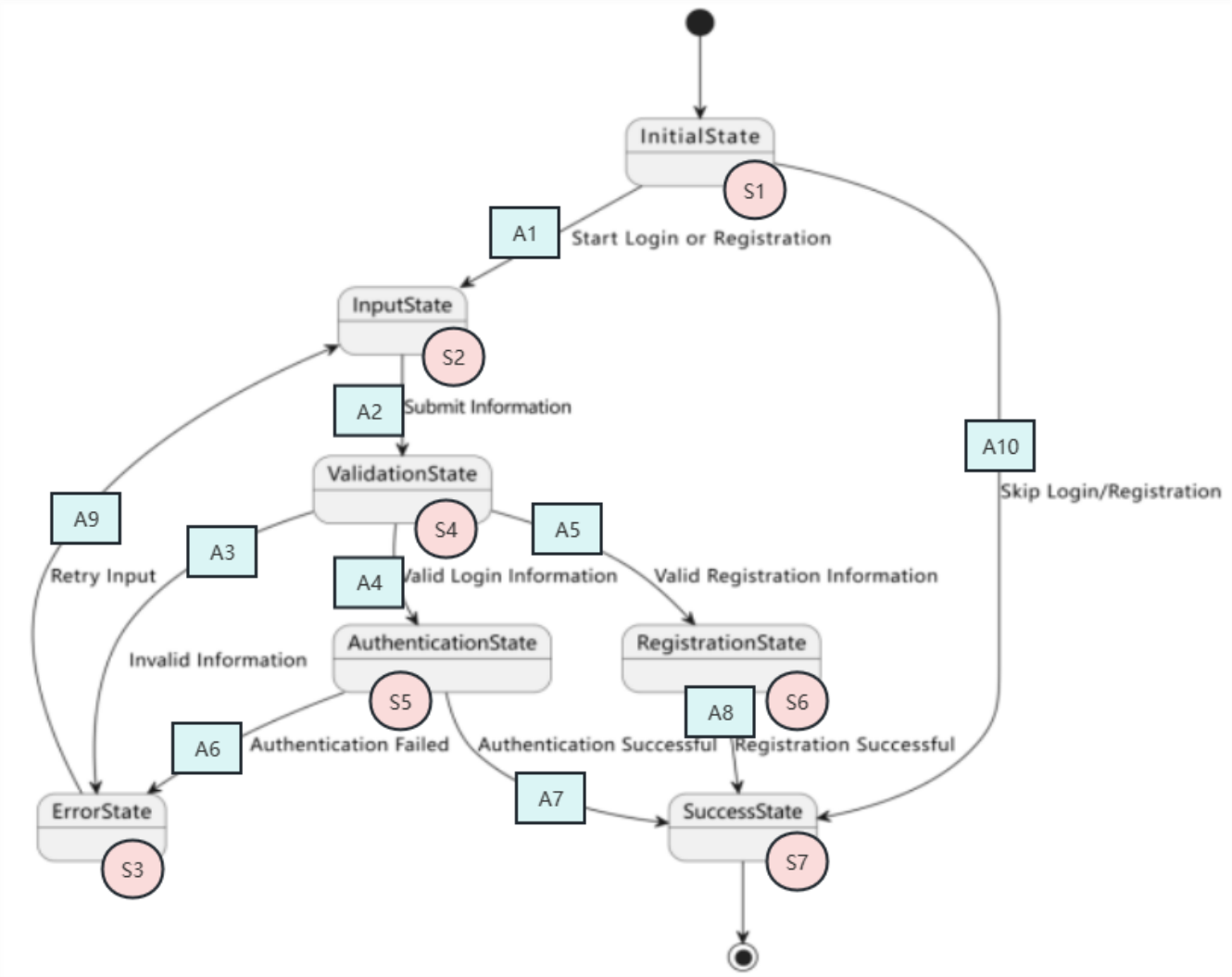
Black-box Testing

Unit_1 Login & Register

State Diagram

Draw the state diagram and determine scenario.

State Diagram of Login&Register:



State Diagram of Login&Register

Scenario Analysis

According to the given state diagram, we consider the following test scenarios for the Login&Register feature:

No.	Test Scenario	Description	Inputs	Expected Outcome
#1	Registratio n	Verify that a user can successfully register with valid	Valid and unique username,	User is successfully registered and

		and unique credentials.	email, password, etc.	redirected to the login page.
#2		Verify that registration fails if the username or email is already taken.	Usename or email that already exists in the system.	Registration fails, and an error message is displayed indicating that the username or email is already taken.
#3		Verify that registration fails if the password does not meet the minimum requirements (e.g., length, complexity) or invalid format of other attributes.	Weak or invalid password that does not meet the minimum requirements.	Registration fails, and an error message is displayed indicating the password requirements (e.g., minimum length, complexity).
#4		Verify that registration fails if any required fields are left blank.	Leaving any required fields blank.	Registration fails, and error messages are displayed for the missing fields.
#5	Login	Verify that a registered user can successfully log in with valid credentials.	Valid username/email and correct password.	User is successfully logged in and redirected to the homepage or a designated landing page.
#6		Verify that login fails if the username or email is incorrect.	Incorrect username or email.	Login fails, and an error message is displayed indicating that the username or email is incorrect.
#7		Verify that login fails if the password is incorrect.	Incorrect password.	Login fails, and an error message is displayed indicating that the password is incorrect.
#8				

		Verify that login fails if any required fields are left blank.	Leaving the username/email or password field blank.	Login fails, and error messages are displayed for the missing fields.
#9		Verify that the user is redirected to the correct page after successful login.	-	After successful login, the user is redirected to the homepage or a designated landing page.
#10		Undo/cancel logging in after valid login information (e.g. Close the browser.)	-	Login fails.
#11	Forgot Password	Verify that a user can request a password reset email if they have forgotten their password.	User requests a password reset email.	User receives a password reset email with a unique reset link.
#12		Verify that the password reset email is sent to the correct email address.	-	The password reset email is sent to the correct email address associated with the user's account.
#13		Verify that the password reset link in the email is valid and not expired.	-	The password reset link in the email is valid and not expired.
#14		Verify that the user can reset their password using the password reset link.	User resets their password using the provided link.	User is able to reset their password successfully and redirected to the login page.
#15		Verify that the user is redirected to the correct page after resetting the password.	-	After resetting the password, the user is redirected to the login page or a designated page

				indicating the successful password reset.
#16	Account Lockout	Verify that the account gets locked after a certain number of consecutive failed login attempts.	Entering incorrect credentials multiple times.	After a certain number of consecutive failed login attempts, the user's account gets locked.
#17		Verify that the account is not locked after a successful login.	Successful login after failed attempts.	The account is not locked, and the user is able to log in successfully.
#18		Verify that the user receives appropriate notification when their account is locked.	-	User receives appropriate notification (e.g., error message) when their account is locked.
#19		Verify that the user can unlock their account using a provided mechanism (e.g., email verification).	User unlocks their account using a provided mechanism (e.g., email verification).	User's account gets unlocked, and they can proceed with logging in.
#20	Security	Verify that password fields are masked (hidden) while entering the password.	-	Password fields are masked (hidden) while entering the password for security reasons.
#21		Verify that password fields are case-sensitive.	-	Password fields are case-sensitive, so "Password" and "password" are considered different.
#22		Verify that the system enforces session timeouts to prevent unauthorized access.	-	The system enforces session timeouts to prevent unauthorized access,

				automatically logging out the user after a specified period of inactivity.
#23		Verify that the system logs failed login attempts for security auditing purposes.	-	Failed login attempts are logged for security auditing purposes.
#24	User Experience	Verify that appropriate error messages are displayed for various failure scenarios.	-	Appropriate error messages are displayed for various failure scenarios, guiding the user on how to resolve the issues.
#25		Verify that the user can easily switch between the login and registration forms.	-	The user can easily switch between the login and registration forms, either through tabs or separate pages.
#26		Verify that the user can access the login and registration pages from different entry points (e.g., homepage, header).	-	The user can access the login and registration pages from different entry points (e.g., homepage, header) without any issues.
#27	Skip Login/Regis tration	Verify that the user can browser some specific pages without logging in.	-	The user can browser some specific pages without logging in.

0 – switch coverage: The test case covers all direct state transitions (through only one edge). It is able to find problems in all systems in all state transitions, but not in successive state transitions where

problems may occur.

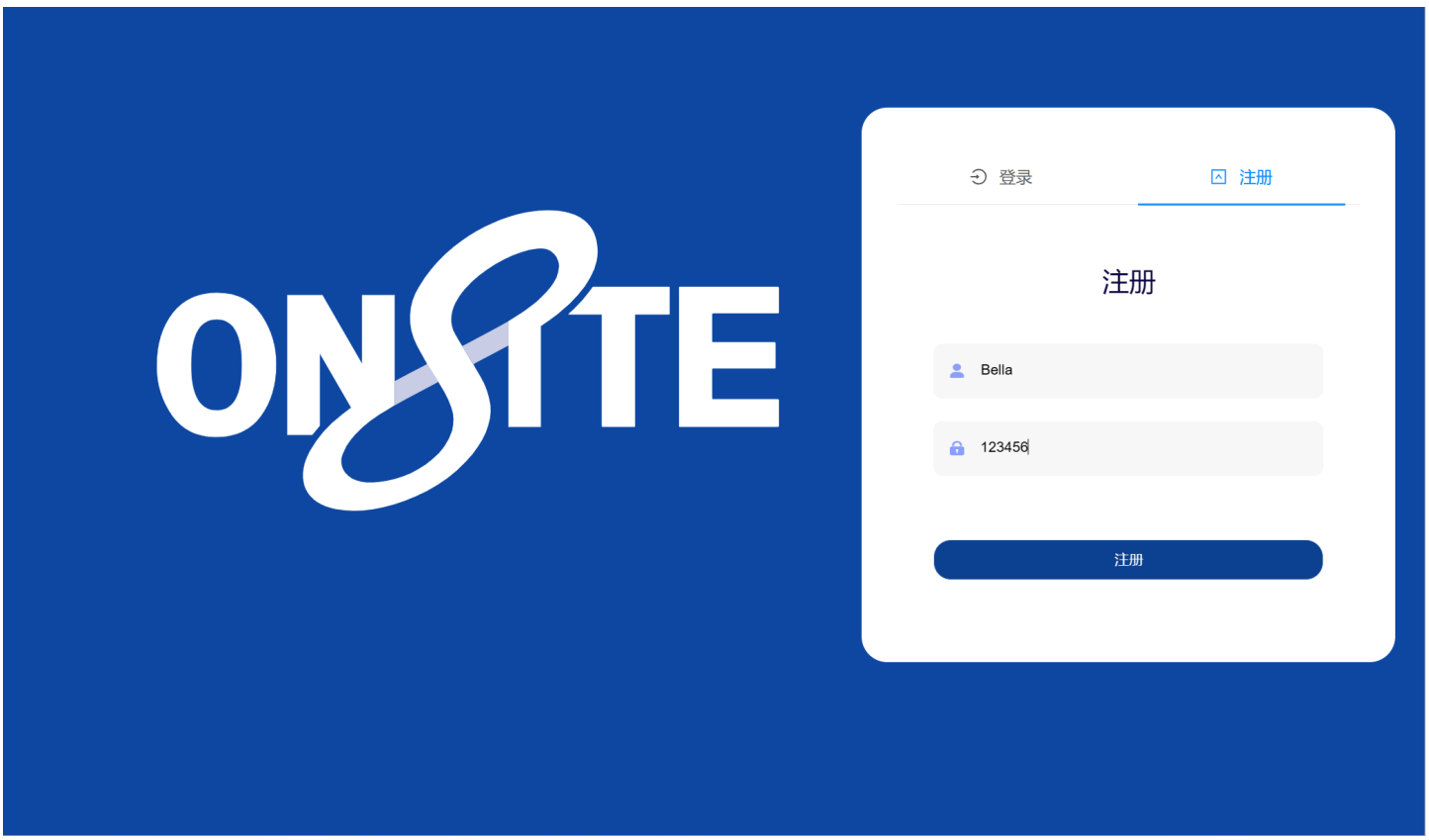
Determine whether the scenarios satisfy 0-switch coverage:

No.	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
Start State	S1	S1	S2	S3	S4	S4	S4	S5	S5	S6
End State	A1	A10	A2	A9	A3	A4	A5	A6	A7	A8
Action#1	S2	S7	S3	S2	S3	S5	S6	S3	S7	S7
Contain Scenario	1,2,3,4,5,etc.	27	1,2,3,4,5,etc.	15,16	2,3,4,6,7,8,etc.	5	1	10	5	1

Conclusion: our test scenario has reached 100% 0-switch coverage.

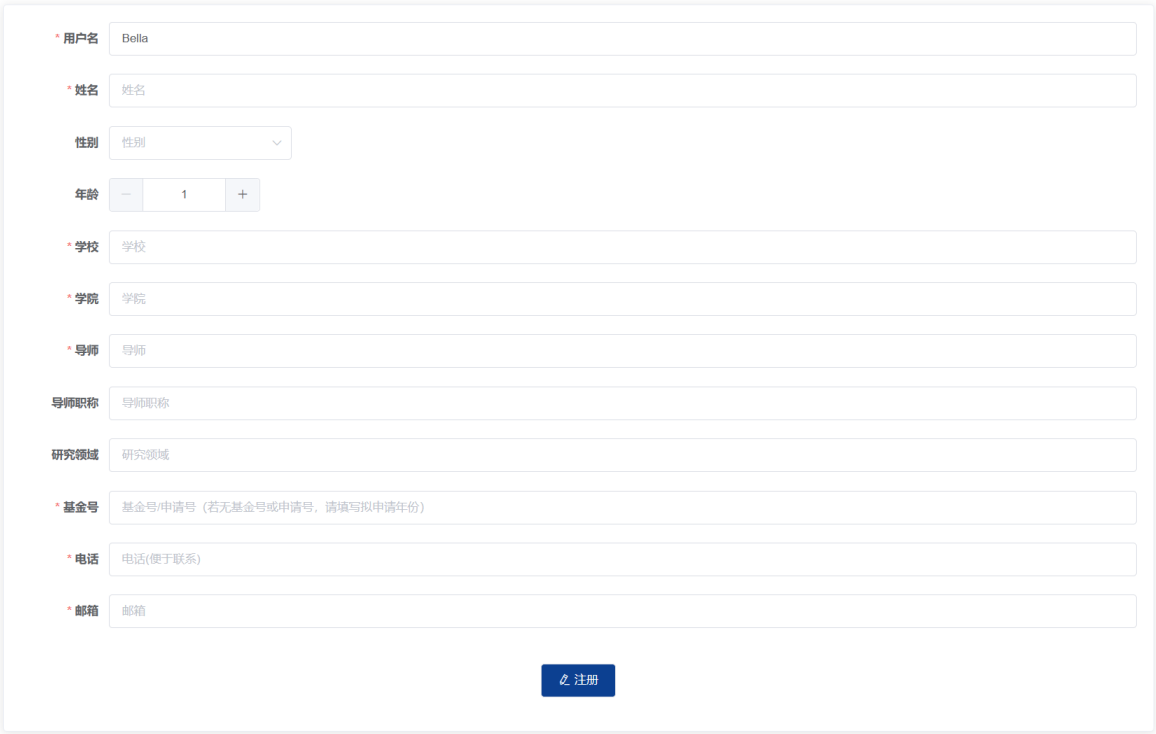
Register Testing

When we drew the state diagram, we only considered [valid] or [invalid]. We use the EP and BVA to further determine which kind of input is considered valid or invalid.



Register Page

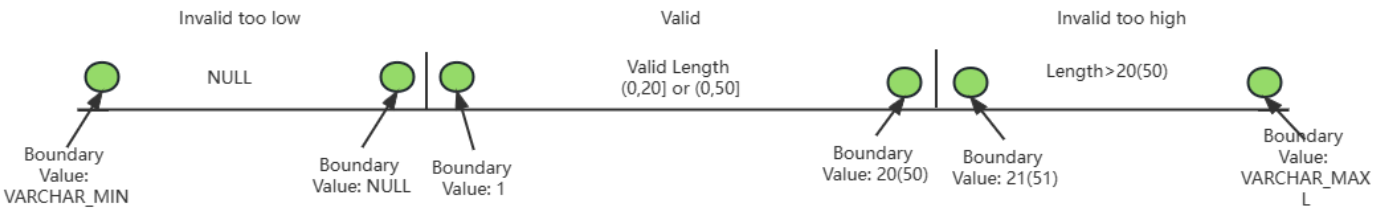
补充更多个人信息



Further detail Page

Property	Effective Equivalence Class	No.	Invalid Equivalence Class	No.
Username	Strings of length greater than 0 and less than or equal to 20	1	Strings of length greater than 20	12
			Null	13
Passowrd	Strings of length greater than 6 and less than or equal to 50 without any Chinese character and blank.	2	Null	14
			Strings of length less than 6	15
			Strings of length greater than 6 and less than or equal to 50 with some Chinese characters or blanks.	16
			Strings of length greater than 50	17
Name	Strings of length greater than 0 and less than or equal to 50	3	Strings of length greater than 50	18
			Null	19

School	Strings of length greater than 0 and less than or equal to 50	4	Strings of length greater than 50	20
			Null	21
Faculty	Strings of length greater than 0 and less than or equal to 50	5	Strings of length greater than 50	22
			Null	23
Tutor	Strings of length greater than 0 and less than or equal to 20	6	Strings of length greater than 20	24
			Null	25
Title of Supervisor	Strings of length greater than 0 and less than or equal to 50	7	Strings of length greater than 50	26
Field of Research	Strings of length greater than 0 and less than or equal to 50	8	Strings of length greater than 50	27
Fund Number	Strings of length greater than 0 and less than or equal to 255	9	Strings of length greater than 255	28
			Null	29
Phone	Strings of length greater than 0 and less than or equal to 20	10	Strings of length greater than 20	30
			Null	31
Email	Strings of length greater than 0 and less than or equal to 50	11	Strings of length greater than 50	32
			Null	33



BVA

According scenario and coverage analysis above, we list our detailed test cases in the following table:

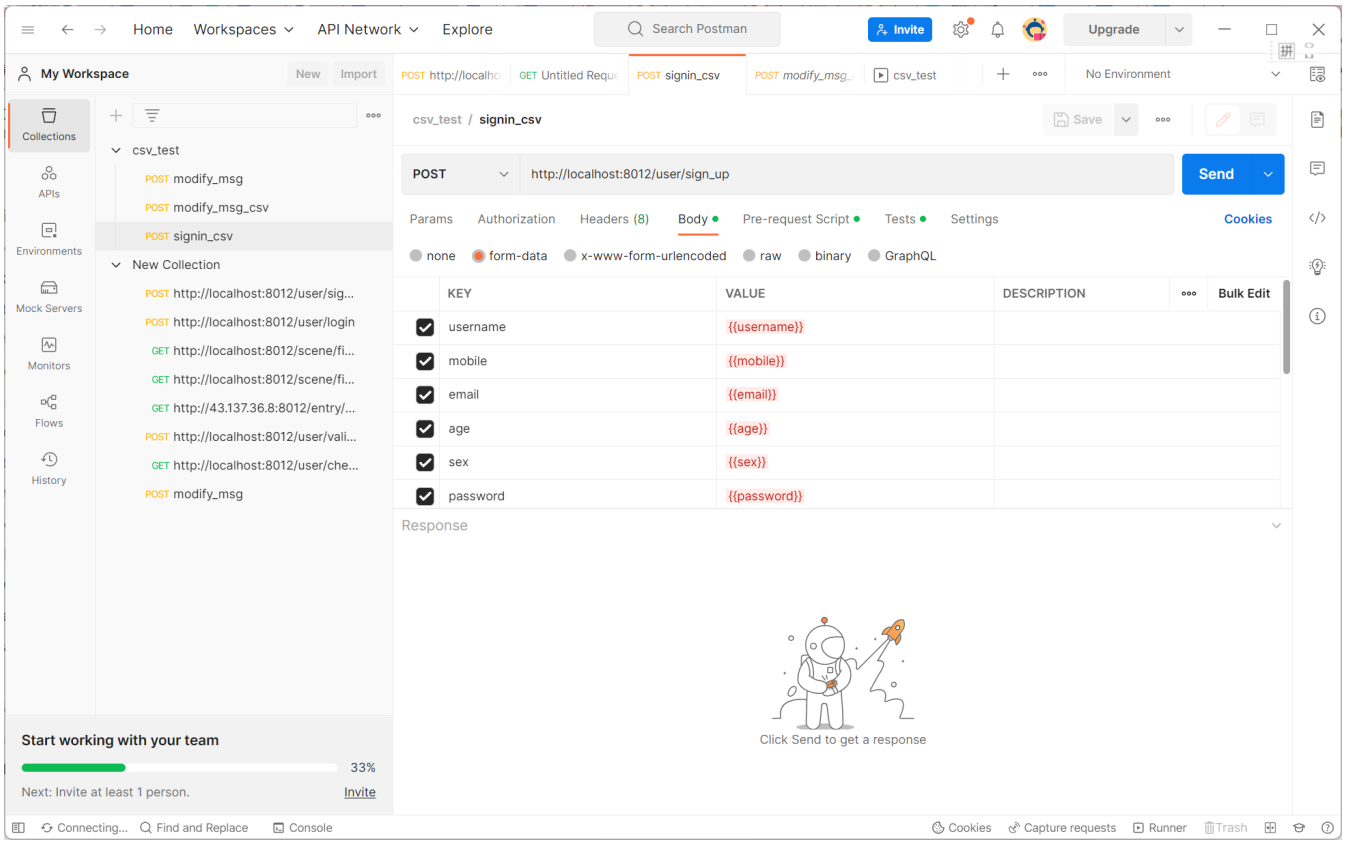
No.	EP&BVA	username	password	mobile	email	age	sex	school	departmei	supervisornam	supervisor	researchfi	works	name
Test Scenario #1														
1	1,2,3,4,5,6,7,8,9,10,11	Bella	123456	17758689102	2051849@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2023	王二虎
Test Scenario #2														
2		River	123456	17758689103	1306715855@qq.com	21	女	同济大学	交通学院	史杨	副教授	自动驾驶	2023	霍林飞
Test Scenario #3														
3	BVA 12	01234567	123456	17758689103	1306715855@qq.com	21	女	同济大学	交通学院	史杨	副教授	自动驾驶	2023	霍林飞
4	BVA 15	Eve_11	12345	15877160554	2051849@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
5	BVA 17	Eve_12	123456789012	15877160554	2051849@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
6	16	Eve_13	123456你好	15877160554	2051849@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
7	16	Eve_14	123456 你好	15877160554	2051849@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
8	16	Eve_15	123 456	15877160554	2051849@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
9	BVA 18	Eve_16	123456	15877160554	2051849@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	123456789
10	BVA 20	Eve_17	123457	15877160554	2051849@tongji.edu.cn	21	女	12345678901234567	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
11	BVA 22	Eve_18	123458	15877160554	2051849@tongji.edu.cn	21	女	同济大学	12345678	黄杰	副教授	自动驾驶	2022	王二虎
12	BVA 24	Eve_19	123459	15877160554	2051849@tongji.edu.cn	21	女	同济大学	交通学院	123456789012	副教授	自动驾驶	2022	王二虎
13	BVA 26	Eve_20	123460	15877160554	2051850@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	12345678	自动驾驶	2022	王二虎
14	BVA 27	Eve_21	123461	15877160554	2051851@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	12345678	2022	王二虎
15	BVA 28	Eve_22	123462	15877160554	2051852@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	12345678	王二虎
16	BVA 30	Eve_23	123463	123456789012	2051853@tongji.edu.cn	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
17	BVA 32	Eve_23	123463	15877160554	123456789012345678901	21	女	同济大学	交通学院	黄杰	副教授	自动驾驶	2022	王二虎
Test Scenario #4														
18	BVA 13		123456	15877160550	2051001@tongji.edu.cn	21	男	同济大学	交通学院	史杨	副教授	自动驾驶	2022	王二虎
19	14	Eve_03		15877160550	2051001@tongji.edu.cn	21	男	同济大学	交通学院	史杨	副教授	自动驾驶	2022	王二虎
20	BVA 31	Eve_04	123456		2051001@tongji.edu.cn	21	男	同济大学	交通学院	史杨	副教授	自动驾驶	2022	王二虎
21	BVA 33	Eve_05	123456	15877160550		21	男	同济大学	交通学院	史杨	副教授	自动驾驶	2022	王二虎
22	BVA 21	Eve_08	123456	15877160550	2051001@tongji.edu.cn	21	男		交通学院	史杨	副教授	自动驾驶	2022	王二虎
23	BVA 23	Eve_09	123456	15877160550	2051001@tongji.edu.cn	21	男	同济大学		史杨	副教授	自动驾驶	2022	王二虎
24	BVA 25	Eve_10	123456	15877160550	2051001@tongji.edu.cn	21	男	同济大学	交通学院		副教授	自动驾驶	2022	王二虎
25	BVA 29	Eve_13	123456	15877160550	2051001@tongji.edu.cn	21	男	同济大学	交通学院	史杨	副教授	自动驾驶		王二虎
26	BVA 19	Eve_14	123456	15877160550	2051001@tongji.edu.cn	21	男	同济大学	交通学院	史杨	副教授	自动驾驶	2022	

Then, we use Postman to test our cases:

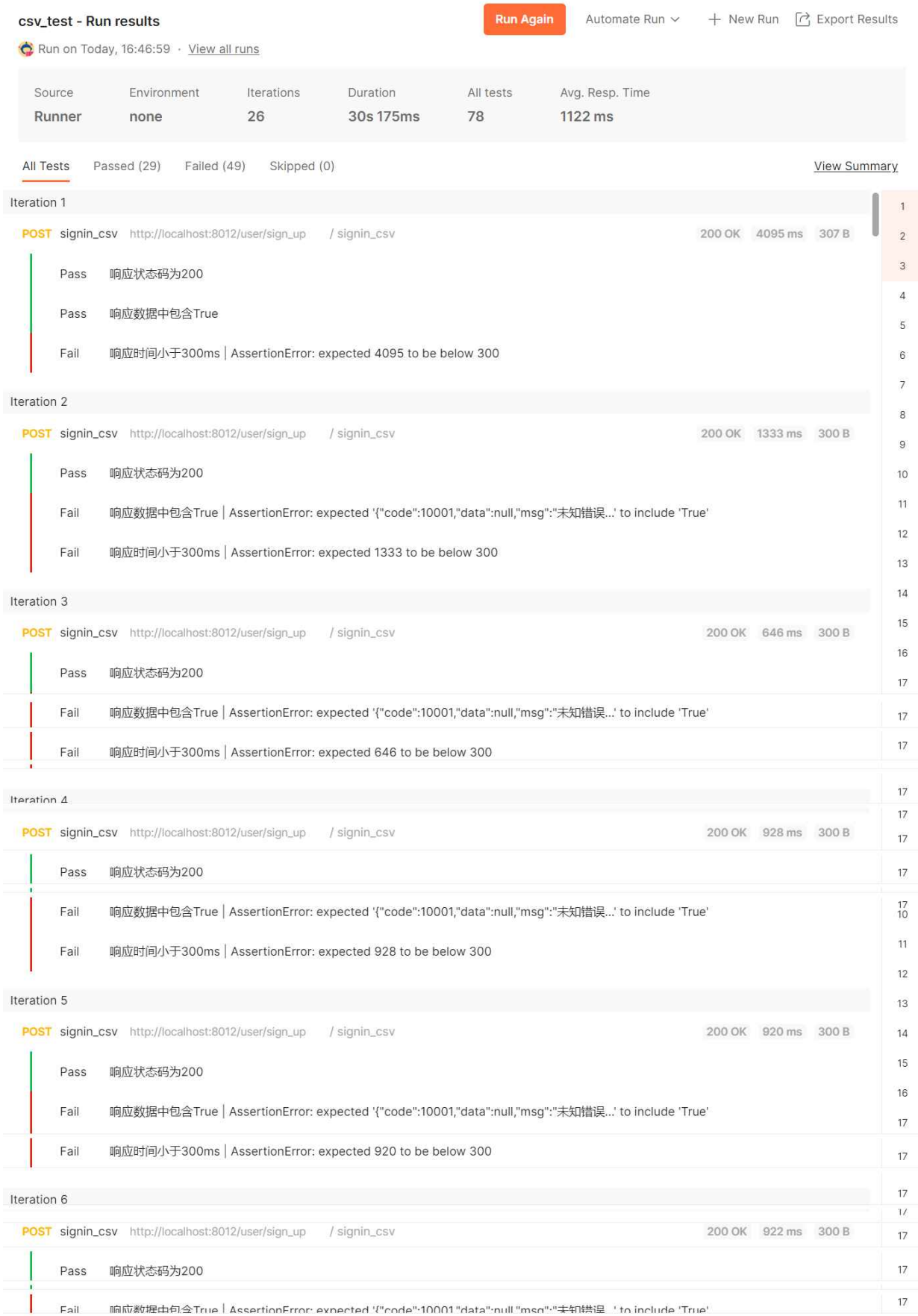
Test Requirements are as follows:

```
pm.test("响应状态码为200", function () {
    pm.response.to.have.status(200);
});

pm.test("响应数据中包含True", function () {
    pm.expect(pm.response.text()).to.include("True");
});

pm.test("响应时间小于300ms", function () {
    pm.expect(pm.response.responseTime).to.be.below(300);
});
```

API in Postman:

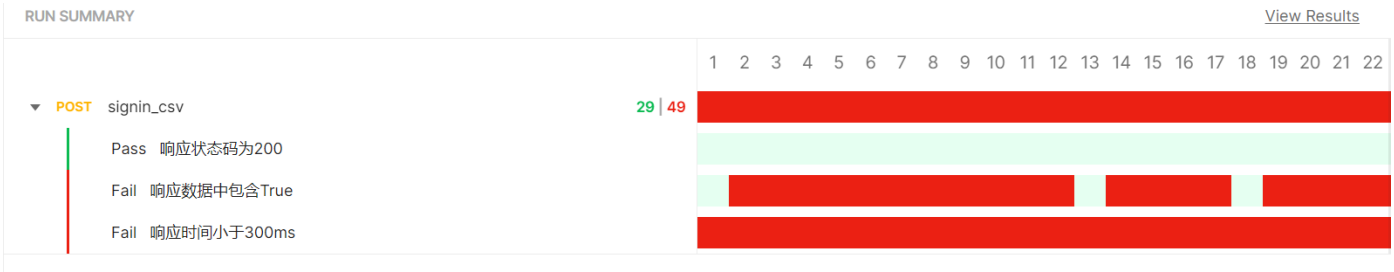


Overall Result:



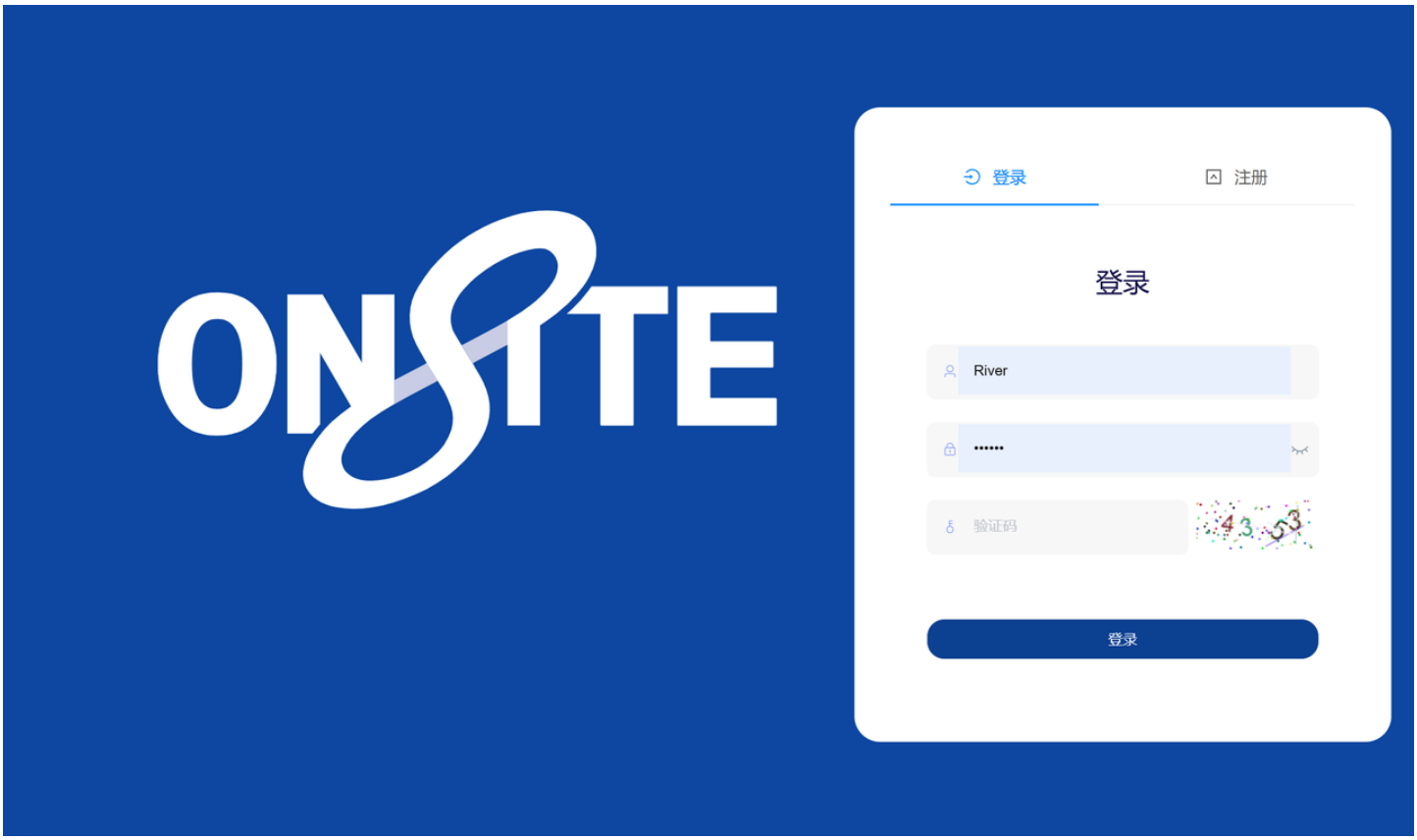
Test Result in Postman

Result Summary:



Login Testing

When we drew the state diagram, we considered [valid] or [invalid]. Since whether [valid] or [invalid] is the sole consideration when logging in (username/password/validcode), it is unnecessary to perform an extra EP&BVA here.



According scenario and coverage analysis above, we list our detailed test cases as well as the testing results in the following table:

No.	Scenario No.	username	password	Validcode	Expect Result	Result	Test Pass ?
1	5,9	River	123456	5713 [valid]	PASS	PASS	Y
2	8	-	123456	3616 [valid]	FAIL	FAIL	Y
3	6	@#$%	123456	4521 [valid]	FAIL	FAIL	Y
4	7	River	123457	4353 [valid]	FAIL	FAIL	Y

5	8	River	-	7143 [valid]	FAIL	FAIL	Y
6	-	River	123456	6666 [invalid]	FAIL	FAIL	Y

Other Scenario Testing

Due to our system, there is no mechanism for "Forgot Password" and "Account Lockout" handling, the scenario#11-19 has not been tested yet.

Other test cases and test results are as follows:

No.	Scenario No.	Describe	Expect Result	Test Pass ?
1	20	Verify that password fields are masked (hidden) while entering the password.	Password fields are masked (hidden) while entering the password for security reasons.	PASS
2	21	Verify that password fields are case-sensitive.	Password fields are case-sensitive, so "Password" and "password" are considered different.	PASS
3	22	Verify that the system enforces session timeouts to prevent unauthorized access.	The system enforces session timeouts to prevent unauthorized access, automatically logging out the user after a specified period of inactivity.	PASS
4	23	Verify that the system logs failed login attempts for security auditing purposes.	Failed login attempts are logged for security auditing purposes.	FAIL
5	24	Verify that appropriate error messages are displayed for various failure scenarios.	Appropriate error messages are displayed for various failure scenarios, guiding the user on how to resolve the issues.	PASS

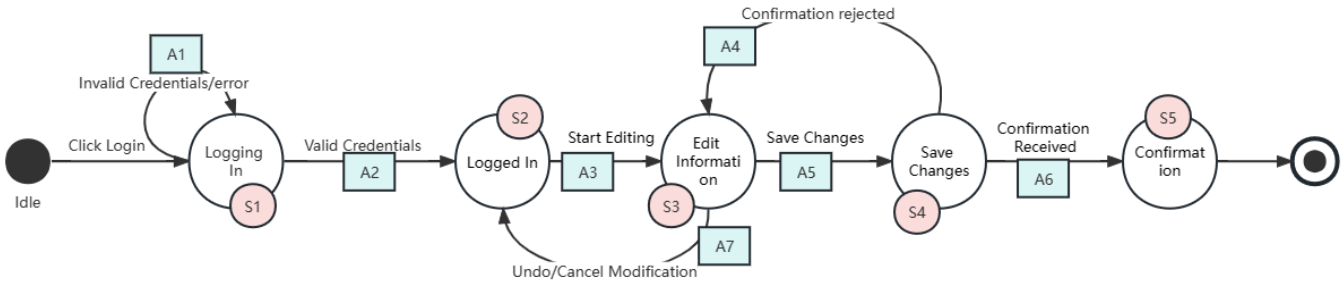
6	25	Verify that the user can easily switch between the login and registration forms.	The user can easily switch between the login and registration forms, either through tabs or separate pages.	PASS
7	26	Verify that the user can access the login and registration pages from different entry points (e.g., homepage, header).	The user can access the login and registration pages from different entry points (e.g., homepage, header) without any issues.	FAIL
8	27	Verify that the user can browser some specific pages without logging in.	The user can browser some specific pages without logging in.	PASS

Unit_2 Modify Personal Information

State Diagram

Draw the state diagram and determine scenario.

State Diagram of Modifying Personal Information



State Diagram of Modifying Personal Information

Scenario Analysis

According to the given state diagram, we consider the following test scenarios for the modifying personal information feature:

No.	Test Scenario	Description	Inputs	Expected Outcome
#1	Successful Modificatio	Users successfully modify their	• Valid userid • Valid password (token)	Personal information is updated

	n	personal information.	• Updated personal information (e.g., name, email, phone number)	successfully, and the changes are reflected in the user's profile.
#2	Invalid Credentials	User attempts to modify personal information with invalid login credentials.	• Invalid userid • Valid password (token) • Updated personal information	The system should reject the modification request and display an error message indicating invalid credentials.
#3	Empty Fields	User tries to modify personal information with empty fields.	• Valid userid • Valid password (token) • Empty fields for the updated personal information	The system should reject the modification request and display an error message indicating that all required fields must be filled.
#4	Restricted Field Modificatio n	User attempts to modify a restricted field (e.g., userid) in personal information.	• Valid userid • Valid password (token) • Updated username	The system should reject the modification request and display an error message indicating that the field cannot be modified.
#5	Invalid Input Format	User provides personal information in an invalid format.	• Valid userid • Valid password (token) • Updated personal information in an invalid format (e.g., invalid email format)	The system should reject the modification request and display an error message indicating the invalid input format.
#6	Maximum Field Length	User attempts to modify personal information with fields exceeding the maximum allowed length.	• Valid userid • Valid password (token) • Updated personal information with fields exceeding maximum length.	The system should reject the modification request and display an error message indicating the maximum allowed length has been exceeded.

#7	Concurrency Handling	Two users simultaneously attempt to modify personal information for the same account.	• Valid userid • Valid password (token) • Updated personal information (by two different users).	The system should handle the concurrency scenario gracefully, ensuring that only one user's changes are applied while providing appropriate feedback to the other user.
#8	Undo/Cancel Modification	User initiates the modification process but decides to cancel or undo the changes.	• Valid userid • valid password (token) • Updated personal information, cancel/undo action.	The system should discard the changes made by the user and revert back to the original personal information.

0 – switch coverage: The test case covers all direct state transitions (through only one edge). It is able to find problems in all systems in all state transitions, but not in successive state transitions where

problems may occur.

Determine whether the scenarios satisfy 0-switch coverage:

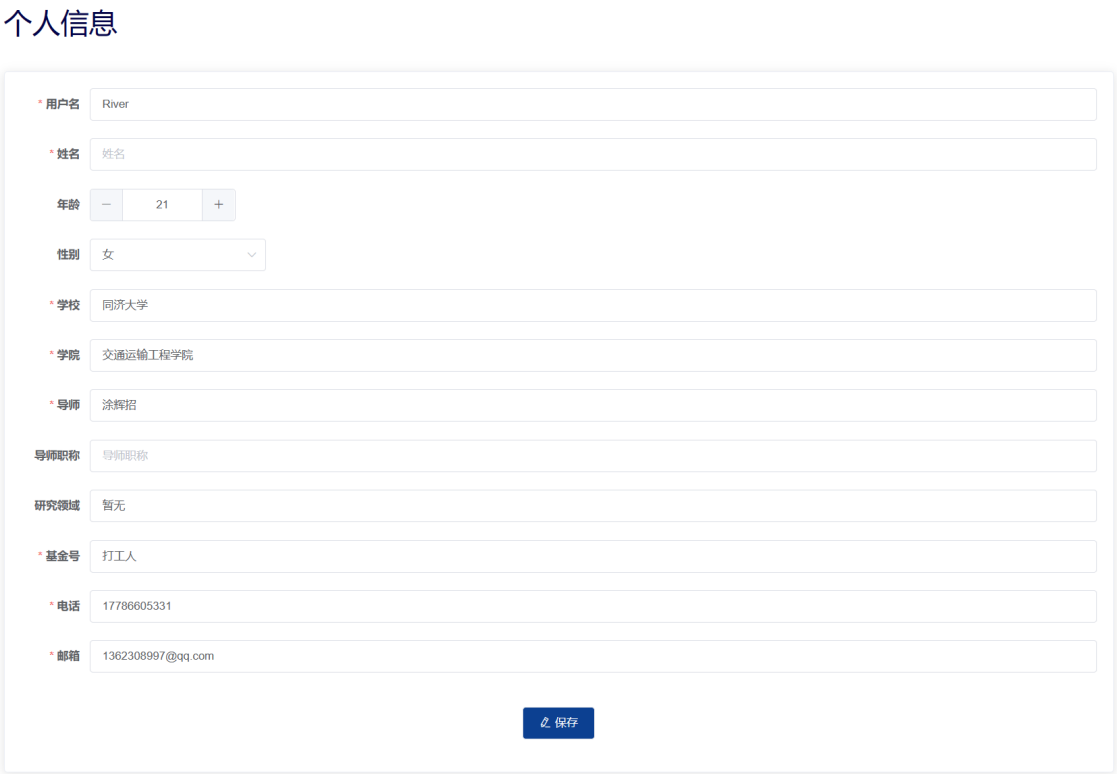
No.	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**
Start State	S1	S1	S2	S3	S3	S4	S4
End State	S1	S2	S3	S2	S4	S3	S5
Action#1	A1	A2	A3	A7	A5	A4	A6
Contain Scenario	2	1	1	8	1	3,4,5,6	1

Conclusion: our test scenario has reached 100% 0-switch coverage.

EP & BVA

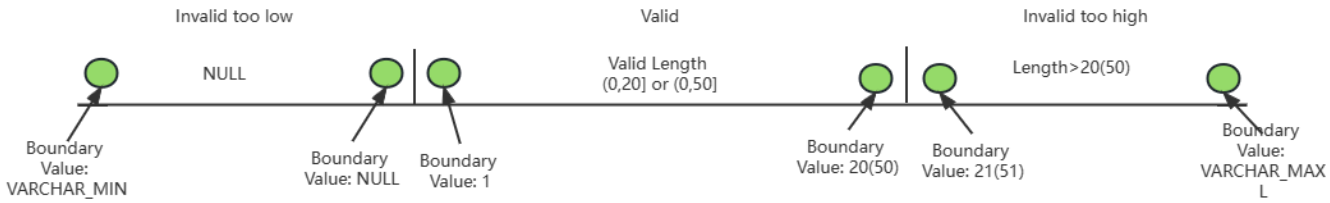
When we drew the state diagram, we only considered [valid] or [invalid]. We use the EP and BVA

to further determine which kind of input is considered valid or invalid.



Property	Effective Equivalence Class	No.	Invalid Equivalence Class	No.
Username	Strings of length greater than 0 and less than or equal to 20	1	Strings of length greater than 20	11
			Null	12
Name	Strings of length greater than 0 and less than or equal to 50	2	Strings of length greater than 50	13
			Null	14
School	Strings of length greater than 0 and less than or equal to 50	3	Strings of length greater than 50	15
			Null	16
Faculty	Strings of length greater than 0 and less than or equal to 50	4	Strings of length greater than 50	17
			Null	18
Tutor	Strings of length greater than 0 and less than or equal to 20	5	Strings of length greater than 20	19

			Null	20
Title of Supervisor	Strings of length greater than 0 and less than or equal to 50	6	Strings of length greater than 50	21
Field of Research	Strings of length greater than 0 and less than or equal to 50	7	Strings of length greater than 50	22
Fund Number	Strings of length greater than 0 and less than or equal to 255	8	Strings of length greater than 255	23
			Null	24
Phone	Strings of length greater than 0 and less than or equal to 20	9	Strings of length greater than 20	25
			Null	26
Email	Strings of length greater than 0 and less than or equal to 50	10	Strings of length greater than 50	27
			Null	28



BVA

Decision Table

The EP and BVA methods help us to determine the legal and illegal values of each input. However, our input will include a combination of 10 values. So, we also need to consider whether the combination of 10 values valid/invalid will have an impact on our results. We use a decision table approach to merge different combinations of the same action to determine how we should finally combine our 10 input values.

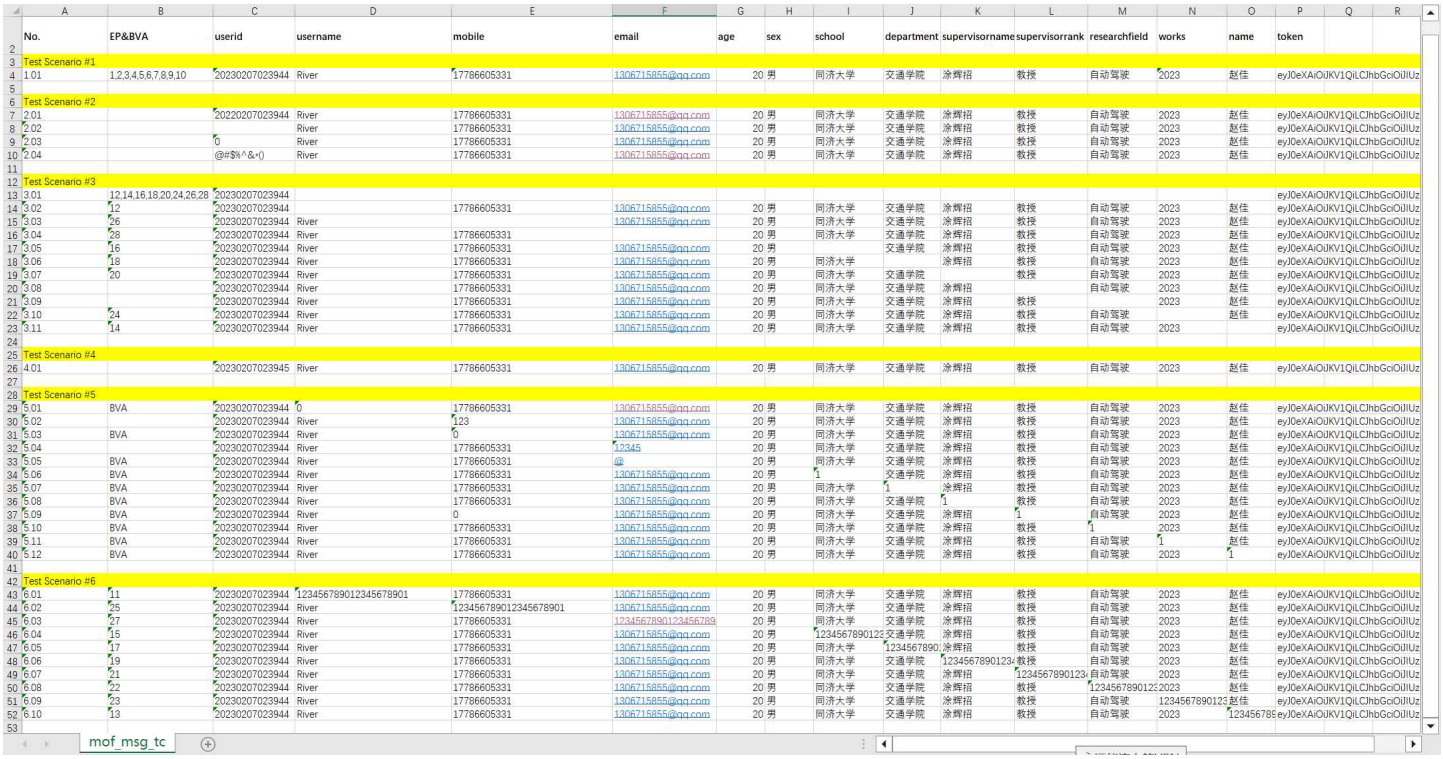
Decision Table		TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10	TC1
Cond.		No	-	-	-	-	-	-	-	-	-	Yes

	Username valid?										
	Name valid?	-	No	-	-	-	-	-	-	-	Yes
	School valid?	-	-	No	-	-	-	-	-	-	Yes
	Faculty valid?	-	-	-	No	-	-	-	-	-	Yes
	Tutor valid?	-	-	-	-	No	-	-	-	-	Yes
	Title of Supervisor valid?	-	-	-	-	-	No	-	-	-	Yes
	Field of Research valid?	-	-	-	-	-	-	No	-	-	Yes
	Fund Number valid?	-	-	-	-	-	-	-	No	-	Yes
	Phone valid?	-	-	-	-	-	-	-	-	No	Yes
	Email valid?	-	-	-	-	-	-	-	-	No	Yes
Actions	Reject operation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	-
	Save changes	-	-	-	-	-	-	-	-	-	Yes

In accordance with the decision table, since the 8 conditions are independent, there are total 2^8 test cases that need to be valid. We consider it unnecessary to test all the combinations and choose several to test.

On the other hand, since age and sex input can not be invalid value in the page, we do not consider these two inputs in our detailed test cases.

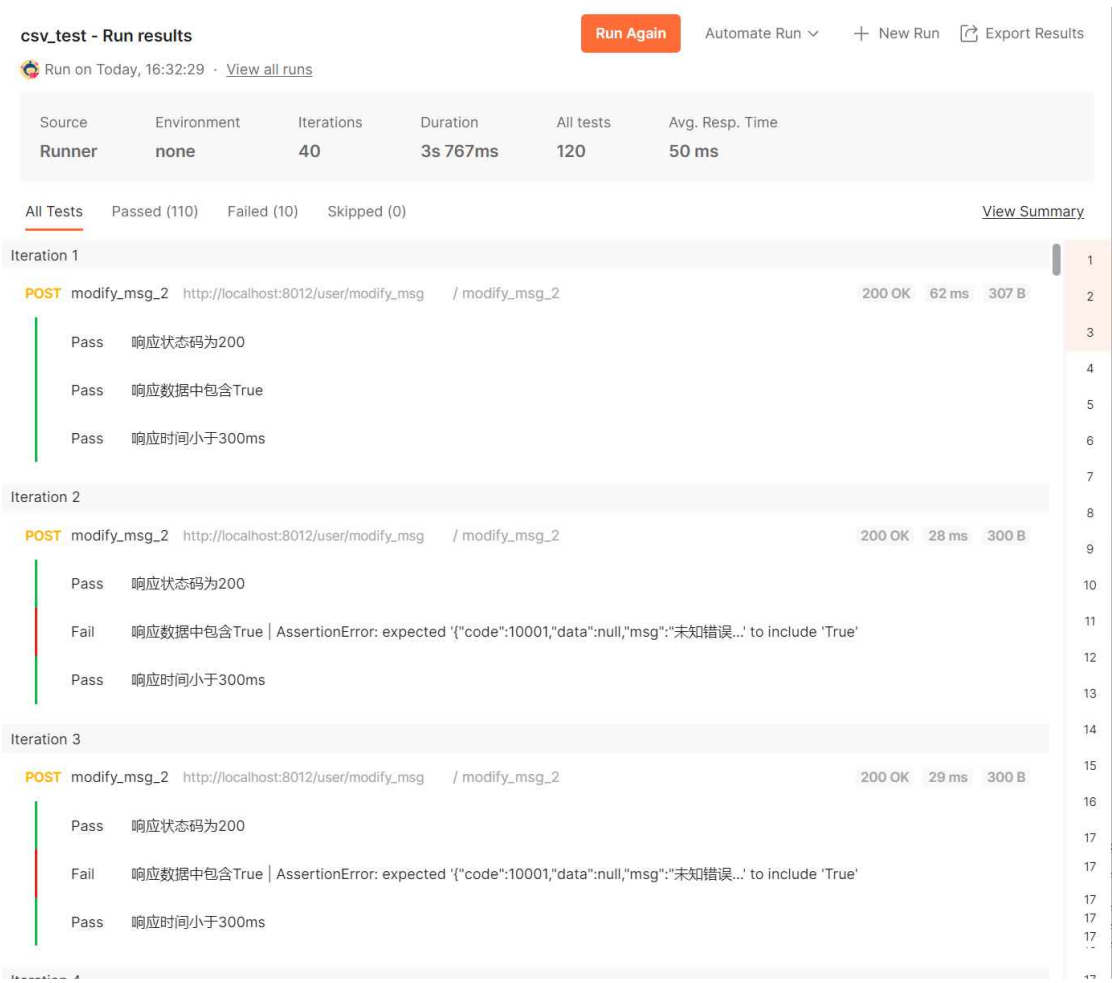
According scenario and coverage analysis above, we list our detailed test cases in the following table:



No.	EP&BVA	userid	username	mobile	email	age	sex	school	department	supervisorname	supervisorrank	researchfield	works	name	token
Test Scenario #1															
1.01	1,2,3,4,5,6,7,8,9,10	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
Test Scenario #2															
2.01		20220207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
2.02			River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
2.03		0	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
2.04		@#$%^&*()	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
Test Scenario #3															
3.01	12,14,16,18,20,24,26,28	20230207023944													eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.02	12	20230207023944		17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.03	26	20230207023944	River		1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.04	28	20230207023944	River	17786605331		20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.05	16	20230207023944	River	17786605331	1306715855@qq.com	20	男		交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.06	18	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学		涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.07	20	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院		教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.08		20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招		自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.09		20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授		2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.10	24	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶		赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
3.11	14	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023		eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
Test Scenario #4															
4.01		20230207023945	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
Test Scenario #5															
5.01	BVA	20230207023944	0	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.02		20230207023944	River	123	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.03	BVA	20230207023944	River	0	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.04		20230207023944	River	17786605331	12345	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.05	BVA	20230207023944	River	17786605331	@	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.06	BVA	20230207023944	River	17786605331	1306715855@qq.com	20	男	1	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.07	BVA	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	1	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.08	BVA	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	1	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.09	BVA	20230207023944	River	0	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	1	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.10	BVA	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	1	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.11	BVA	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	1	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
5.12	BVA	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	1	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
Test Scenario #6															
6.01	11	20230207023944	123456789012345678901	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.02	25	20230207023944	River	123456789012345678901	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.03	27	20230207023944	River	17786605331	12345678901234567890	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.04	15	20230207023944	River	17786605331	1306715855@qq.com	20	男	12345678901234	交通学院	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.05	17	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	12345678901	涂辉招	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.06	19	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	12345678901234	教授	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.07	21	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	12345678901234	自动驾驶	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.08	22	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	12345678901234	2023	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.09	23	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	12345678901234	赵佳	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz
6.10	13	20230207023944	River	17786605331	1306715855@qq.com	20	男	同济大学	交通学院	涂辉招	教授	自动驾驶	2023	12345678	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz

Modify Message detailed test case

Then we run all test cases in Postman.



Iteration 4

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 30 ms 300 B

Pass 响应状态码为200

Fail 响应数据中包含True | AssertionError: expected '{"code":10001,"data":null,"msg":"未知错误..' to include 'True'

Pass 响应时间小于300ms

Iteration 5

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 32 ms 300 B

Pass 响应状态码为200

Fail 响应数据中包含True | AssertionError: expected '{"code":10001,"data":null,"msg":"未知错误..' to include 'True'

Pass 响应时间小于300ms

Iteration 6

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 37 ms 307 B

Pass 响应状态码为200

Pass 响应数据中包含True

Pass 响应时间小于300ms

Iteration 7

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 43 ms 307 B

Pass 响应状态码为200

Pass 响应数据中包含True

Pass 响应时间小于300ms

Iteration 8

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 56 ms 307 B

Pass 响应状态码为200

Pass 响应数据中包含True

Pass 响应时间小于300ms

Iteration 9

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 49 ms 307 B

Pass 响应状态码为200

Pass 响应数据中包含True

Pass 响应时间小于300ms

Iteration 10

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 43 ms 307 B

Pass 响应状态码为200

Pass 响应数据中包含True

Pass 响应时间小于300ms

Iteration 11

POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 57 ms 307 B

Pass 响应状态码为200

Pass 响应数据中包含True

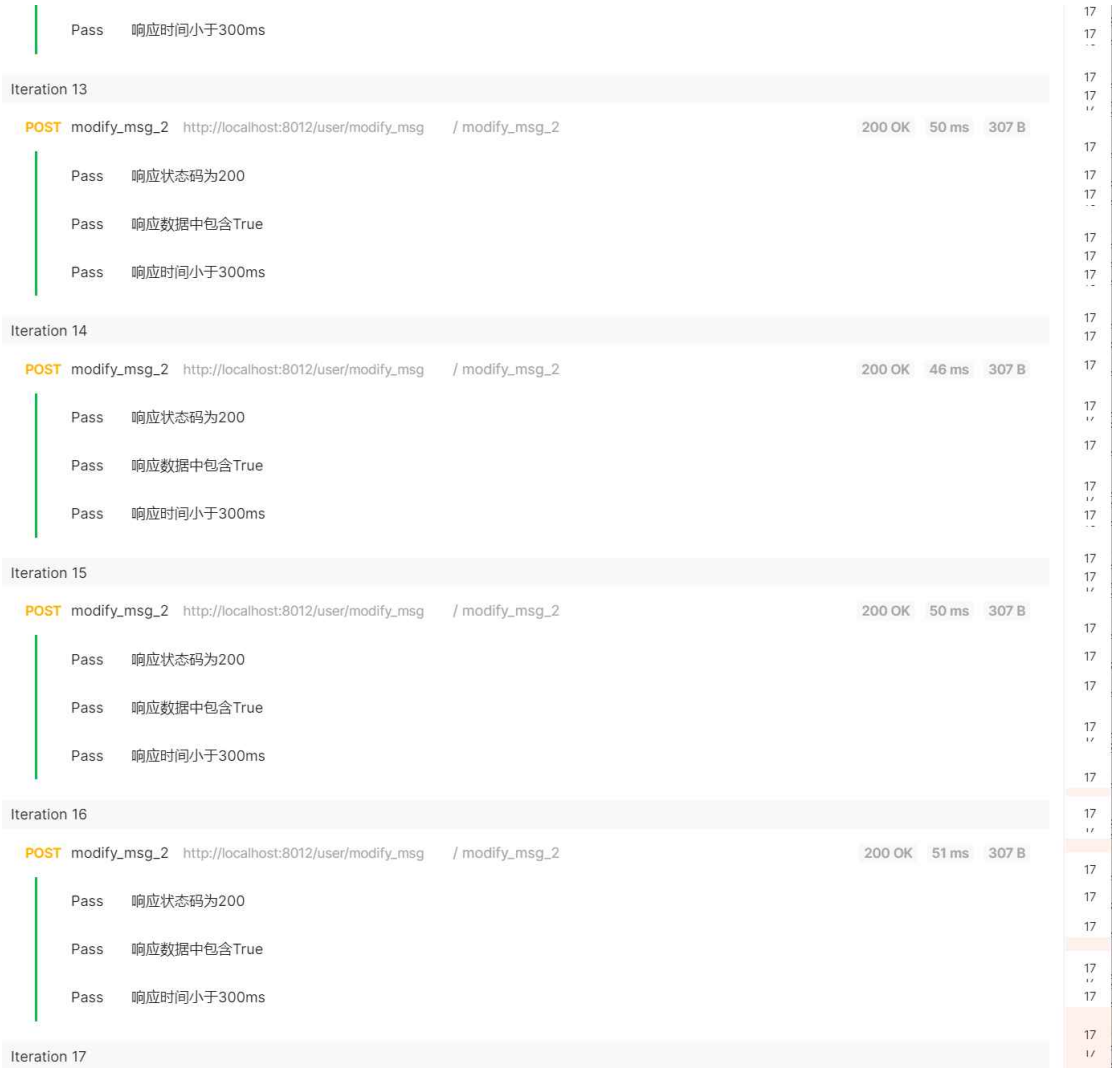
Pass 响应时间小于300ms

Iteration 12

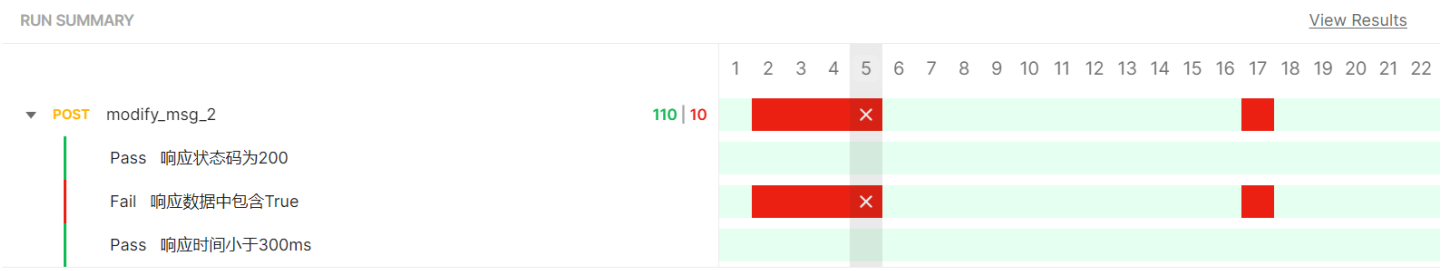
POST modify_msg_2 http://localhost:8012/user/modify_msg / modify_msg_2 200 OK 60 ms 307 B

Pass 响应状态码为200

Pass 响应数据中包含True



Test Result in Postman



Running Result Summary

Since scenario#7 and scenario#8 are not in above testing, we conduct these two test cases in addition as follows:

No.	Scenario No.	Test Case	Expect Result	PASS ?
41	#7 Concurrency Handling	Two users simultaneously attempt to modify personal information for the same account.	The system should handle the concurrency scenario gracefully, ensuring that only one user's changes are	Y

			applied while providing appropriate feedback to the other user.	
42	#8 Undo/Cancel Modification	User initiates the modification process but decides to cancel or undo the changes.	The system should discard the changes made by the user and revert back to the original personal information.	Y

White-box Testing

Test Target

The objectives of white-box testing for all the classes can include:

1. Testing the business logic: Verify that the business rules and algorithms implemented in the classes are working correctly. This includes checking the accuracy of calculations, proper handling of edge cases, and adherence to the specified requirements.
2. Ensuring data integrity: Validate that the data manipulation operations performed by the classes are correct, including data retrieval, modification, and storage. This includes verifying the consistency and accuracy of the data stored in the system.
3. Checking error handling and exception scenarios: Validate that the classes handle error conditions gracefully and appropriately. This involves testing error handling, exception handling, and proper error messages or notifications to the users.
4. Testing integration with external dependencies: Validate the integration of the classes with external components, such as databases, APIs, or third-party services. This ensures that the classes interact correctly with these dependencies and handle the expected responses and errors.
5. Assessing code coverage: Ensure that the test cases cover a significant portion of the codebase, aiming for high code coverage. This helps identify any untested or unreachable code segments, reducing the risk of undetected bugs.
6. Performance and scalability testing: Evaluate the performance of the classes under different scenarios, such as high load or concurrent requests, to ensure that they can handle the expected workload and meet performance requirements.

7. Security testing: Validate the security measures implemented in the classes, such as input validation, authentication, and authorization. This includes testing for potential vulnerabilities, such as SQL injection or cross-site scripting.

By performing white-box testing on all the classes, we can gain confidence in the reliability, functionality, and quality of the entire system, identifying and addressing any issues or bugs in the codebase.

For example:

One purpose of the white-box testing using the provided code is to test the internal implementation and logic of the `UserController` class. By directly accessing the controller's methods through `MockMvc`, we can verify if the controller functions correctly based on different scenarios and inputs.

Specifically, the tests aim to accomplish the following objectives:

1. `loginValidate()`: Verify that the login functionality works correctly by sending a POST request with valid username and password, and asserting that the expected response status is "OK".
2. `signUp()`: Test the user registration process by sending a POST request with the necessary user information, such as username, password, and email. The objective is to ensure that the registration endpoint returns an "OK" status, indicating that the user has been successfully registered.
3. `checkMsg()`: Validate the functionality of checking user messages by sending a GET request with a specific username and asserting that the response status is "OK". This test verifies if the endpoint correctly handles the request and returns the appropriate response.
4. `modifyMsg()`: Test the modification of user information by sending a PUT request with updated user data and asserting that the response status is "OK". This verifies if the endpoint properly handles the update request and returns the expected result.
5. `modifyPassword()`: Verify the functionality of modifying the user's password by sending a PUT request with the user's ID, old password, and new password. The objective is to ensure that the endpoint processes the password change request correctly and returns the expected response status.
6. `validateUsername()`: Test the validation of a username by sending a GET request with a specific username and asserting that the response status is "OK". This verifies if the endpoint correctly validates the username and returns the appropriate response.

7. `validateMobile()` : Validate the mobile number by sending a GET request with a specific mobile number and asserting that the response status is "OK". This test ensures that the endpoint properly handles the validation request and returns the expected response.
8. `validateEmail()` : Test the validation of an email address by sending a GET request with a specific email and asserting that the response status is "OK". This verifies if the endpoint correctly validates the email address and returns the appropriate response.

By conducting these white-box tests, we can verify the correctness of the internal implementation of the `UserController` class and ensure that it behaves as expected for various input scenarios.

Test Criteria

1. Functional Test Criteria:
   Functional test criteria are based on the desired functionality or behavior of the software system. They assess whether the system functions correctly and meets the specified requirements. Examples of functional test criteria include:
   - Validating that all user interface elements and features work as expected.
   - Verifying that input validation and data processing functions correctly.
   - Testing different scenarios and conditions to ensure the system handles them appropriately.
   - Checking that the system produces the expected outputs and results.
   - Verifying that the system integrates correctly with external components, such as databases or APIs.
2. Performance Test Criteria:
   Performance test criteria assess the system's performance and its ability to handle different workloads and stress conditions. They focus on factors such as speed, scalability, and resource usage. Examples of performance test criteria include:
   - Measuring the response time of critical operations and ensuring they meet acceptable performance thresholds.
   - Testing the system under various load levels to assess its scalability and resource consumption.
   - Verifying that the system can handle a large number of concurrent users or requests without performance degradation.

- Assessing the system's stability and resource utilization over an extended period of operation.
- Monitoring and analyzing system metrics, such as CPU and memory usage, to identify any bottlenecks or performance issues.

Test Design

The **object of this white box test** is a Java class named UserController.java, which contains nine

methods as follows:

	A	B	C	D
1	**Function Name**	**Description**	**Included in Test Cases?**	**Branches within Function?**
2	loginValidate()	Validates user login credentials and performs authentication.	Yes	No
3	signUp()	Registers a new user and adds them to the system.	Yes	No
4	checkMsg()	Checks for new messages for the logged-in user.	Yes	No
5	modifyMsg()	Modifies user messages, such as marking them as read.	Yes	No
	modifyPass	Allows users to change		

6	word()	their account password.	Yes	No
7	validateUsername()	Validates the uniqueness and format of a username during registration.	Yes	No
8	validateMobile()	Validates the uniqueness and format of a mobile number during registration.	Yes	No
9	validateEmail()	Validates the uniqueness and format of an email address during registration.	Yes	No

Database connection test

For all Class test

Code example:

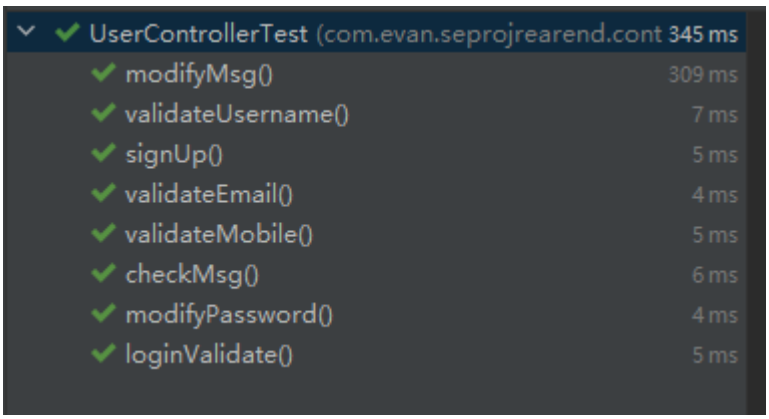
```
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMock
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.http.MediaType;
```

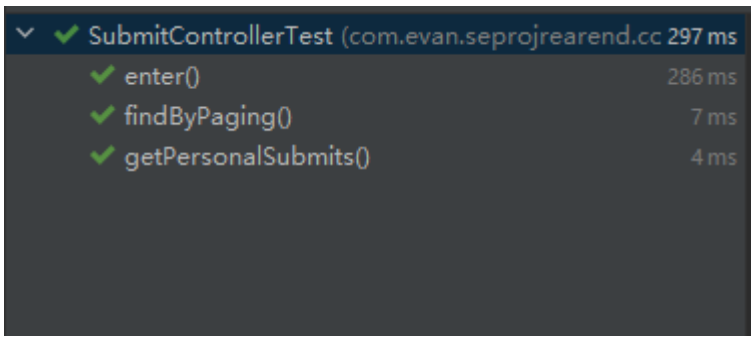
```java
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;
import org.springframework.test.web.servlet.result.MockMvcResultMatchers;

import static org.springframework.test.web.servlet.result.MockMvcResultHandlers.

@SpringBootTest
@AutoConfigureMockMvc
class SceneControllerTest {

    @Autowired
    private MockMvc mockMvc;

    @Test
    void findByPaging() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.get("/scenes")
                        .param("page", "1")
                        .param("size", "10")
                        .contentType(MediaType.APPLICATION_JSON))
                .andExpect(MockMvcResultMatchers.status().isOk())
                .andDo(print());
    }

    @Test
    void getSceneMsg() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.get("/scenes/1"))
                .andExpect(MockMvcResultMatchers.status().isOk())
                .andExpect(MockMvcResultMatchers.jsonPath("$.id").value(1))
                .andExpect(MockMvcResultMatchers.jsonPath("$.name").value("Test
                .andDo(print());
    }

    @Test
    void getSceneUser() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.get("/scenes/1/users"))
                .andExpect(MockMvcResultMatchers.status().isOk())
                .andExpect(MockMvcResultMatchers.jsonPath("$[0].id").value(1))
                .andExpect(MockMvcResultMatchers.jsonPath("$[0].name").value("Us
                .andExpect(MockMvcResultMatchers.jsonPath("$[1].id").value(2))
                .andExpect(MockMvcResultMatchers.jsonPath("$[1].name").value("Us
                .andDo(print());
    }
}
```

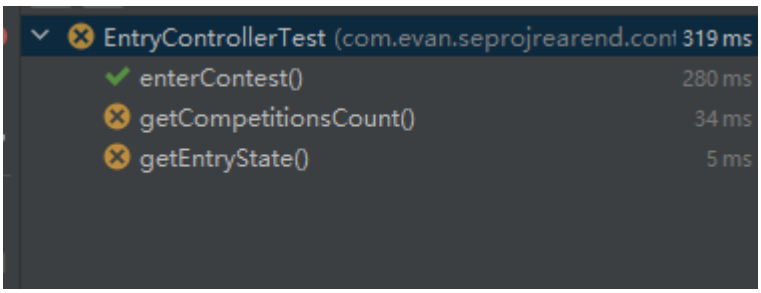
The purpose of the `MockMvc` in the provided code is to simulate HTTP requests and test the functionality of the `UserController` class without actually making real network connections

or accessing the database. It allows for isolated testing of the controller's endpoints by providing a controlled environment where the controller can be tested independently of other components, such as the web server or the database. By mocking the HTTP requests and responses, it enables the execution of specific test scenarios and assertions on the expected behavior of the controller methods, ensuring that they handle the requests correctly and return the expected responses.

Result







All the back-end code was tested, and the part of the code where the interceptor existed failed to pass the test indicating that the interceptor was working

Test Result Analysis

1. The functionality of the end-user system is basically met, and the associated risks and concerns are necessary.
2. There are critical problems discovered in black-box testing with the testing module:
   a. Register:
      - In almost all 26 test cases in register, the response time suspended 300ms, it may cause lower usability.
      - There are no mechanisms to check whether email or phone are in the correct format. It may cause some problems when fake or fraudulent registrations occur (risk 1.05) or inadequate validation of user input during the registration process(risk 1.06), which means that in this feature, potential risks are not handled properly.
   b. Login
      - It will be a serious problem that there is no mechanism for "Forgot Password" and "Account Lockout" handling. Users can not find their password in an effective way which leads to low usability. On the other hand, no mechanism for "Account Lockout" means the system can not deal with brute-force attacks (malicious actors can attempt to guess or crack user passwords), so that the user's personal information can be exposed.
   c. Modifying personal information
      - The modifying personal information page can not handle wrong input (empty field, invalid format, etc) properly.
      - It is not reasonable to allow users to change their username, which is defined when registering. In other words, strict areas, such as username or email can not be modified easily.

3. During the white box testing process for the User management module, we have reached the following conclusions:
    a. Functional tests: All functional test cases for the UserController.java file have passed. This indicates that the functions and logic in the file are functioning correctly under different input conditions. The requests are processed accurately, and the expected results are returned.
    b. Boundary tests: By testing the boundary cases of input, we have confirmed that the UserController.java file handles boundary conditions correctly for different ranges of input values. This demonstrates that the file is robust and can handle all possible input values effectively.
    c. Exception handling: The test results indicate that the UserController.java file adequately captures and handles various exception situations. Whether incorrect parameters are provided or other contingencies occur, the file handles them in a reasonable manner to avoid application crashes or unexpected errors.
    d. Code coverage: In the white box test, our test cases have achieved 100% coverage of the code in the UserController.java file. This means that we have tested every statement and branch in the file, ensuring code integrity and reliability.
    e. However, one area for improvement is the usage of int data for the annotation's result instead of following HTTP standard codes. This can pose challenges in test design and may lead to inconsistencies. It would be beneficial to align the result annotations with the HTTP status codes to enhance clarity and conformity with industry standards.

In conclusion, the system (OnSite) basically attains its initial goals, however it is far more enough to guarantee its security and usability.