Fpt University

TRƯỜNG ĐẠI HỌC FPT

CAPSTONE PROJECT REPORT

Report 5 – Software Test Documentation

– Hanoi, September 2024 –

Table of Contents

I. Record of Changes

Date	A* M, D	In charge	Change Description
23/09/2024	A	ThuyDTT	Initiation, add scope of testing
26/09/2024	A	ThuyDTT	Add Test strategy
30/09/2024	A	ThuyDTT	Add test plan, test milestone
30/10/2024	M	ThuyDTT	Update test strategy
13/11/2024	M	ThuyDTT	Update scope of test
03/12/2024	A	ThuyDTT	Add test report
5/12/2024	M	ThuyDTT	Update test report

*A - Added M - Modified D - Deleted

II. Testing Documentation

1. Scope of Testing

1.1.Target of test

1.1.1. Feature, Functional

The test scope of the project includes all features – functions defined in [SEP490_G51_Report 1_Project Introduction]

1.1.2. Non - Functional

A. External Interfaces

1. User Interface

- All texts in the system are displayed in grammatically correct Vietnamese language.
- Whenever the user/admin performs an action that requires connection to the server, a loading indicator should appear so that the user knows what's going on and doesn't misunderstand that the service is down.
- There should be a clear alert when the app encounters a server error (e.g. offline).
- All pages should have a consistent visual theme and typeface.
- Icons and buttons should have tooltips or labels to assist users in understanding their functions.
- Users can copy and paste content using both the mouse and key combinations, enhancing the convenience of operation.
- Notifications displayed on the screen will have an easy-to-see notification frame, with clear content for users to quickly grasp.
- The interface must be compatible with popular browsers such as Chrome, Firefox, and Microsoft Edge.
- The UI must be responsive on different sizes of screen
- The system must ask for confirmation (Y/N) for data deletion operations and bulk operations.
- The entire drop down list must be arranged in A to Z order and ascending numbers, for night shift workers, arranged in order of priority.
- When the user is in a certain function, the position of this function on the navigation bar (sidebar) will be highlighted, making it easy for users to know where they are in the system

2. Software Interfaces

Software interfaces that our framework will associated with amid operation:

- The system uses Gmail Service to send mails for users
- The system also uses SignalR to send notifications to users

B. Quality Attributes

1. Usability

- Implement solid design principles, such as clear icons, button types, tooltips, and logical menu structure. Users should comprehend their functions within 10 seconds.
- Ensure that the app's user interface is intuitive and easy to use, even for non-IT users with less technical expertise. A non-trained new user can understand and use the system proficiently and do the tasks they want with the accuracy of 90% in 2 hours.
- Include in-app guides to assist users in understanding how to use a specific functionality effectively.
- Ensure a consistent UX across.
- The Front-end web application should support Chrome, Edge browsers.

2. Security

To ensure system security, strong security measures such as data encryption, user authentication and access control are required.

- The system shall require users to authenticate using a unique username and a strong password that meets complexity requirements (minimum 8 characters, including uppercase letters, lowercase letters, numbers, and special characters).
- All the password of the users are encrypted
- When a user requests a password reset, the system shall send a verification code to the user's registered email.
- The system shall validate all user inputs to prevent injection attacks.
- Ensure only authorized users can access the corresponding functions and data. The system shall implement token-based authentication (JSON Web Tokens - JWT) to manage user sessions securely and efficiently.

3. Performance

- All web pages shall load completely within 5 seconds over a standard broadband connection (10 Mbps) under normal load conditions.
- The system shall retrieve and display requested data (e.g., course lists, user profiles) within 4 seconds after a query is submitted.

1.2. Testing Levels

We have 4 levels of testing: Unit Test, Integration Test, System Test, Acceptance test

No.	Test Level	In charge	Time	Focuses	Acceptance criteria
1	Unit Test	Developers	Implementing	Checks the functionality of software components that may be tested separately and looks for errors in them.	The entire source code must be entirely covered by branch conditions and correctly executed without unexpected exceptions.
2	Integration Test	Testers and Developer	After accepting UT	Make sure all the parts come together smoothly	Ensure accurate data flow between the controller layer and service layer, the repository, and then the database.
3	System Test	Testers	After accepting IT	Integrate various modules and test the interaction between them to ensure the accuracy of the data.	Each logic flow operates exactly as intended. System interfaces and non-functional requirements operate correctly during production.

4	Acceptance Test	End-users: Parents of student, manager, secretary, staff	Before the product is officially launched	Pure functional testing to check system behaviour with real data.	The system meets customer business needs,product availability.

Table 1. Description of Testing Types

2. Test Strategy

2.1 Testing Types

The test team has to test the following type on Postman, Google Chrome browser, Edge browser:

- Unit Testing
- GUI Testing
- API Testing
- Functional Testing
- Non-Functional Testing

Types	Objective	Technique	Completion Criteria
Unit Test	Verify that all method code in the program functions as intended with no unexpected exceptions for the function for which it was written.	Black box testing: A method for testing the user interface, input, and output. White box testing is performed to test the behaviour of each of the functions.	The entire set of source code's branch conditions must be covered and correctly performed without any unexpected exceptions.

GUI Testing	The primary goal of the GUI testing is to validate the features of the software, or the application performs as per the given requirement / specifications.	A graphical user interface includes all the elements such as menus, checkbox, buttons, colours, fonts, sizes, icons, content, and images. GUI testing is done to check the functionality and usability of design elements as a user for an application under test.	All GUI components render correctly. The intended functionality of the application can be executed using the GUI. Error messages are displayed correctly.
API Testing	Make sure the parts that are connected to one another function effectively. Verify the program logic all the way through from the controller layer to the service layer to the repository layer.	In order to track predicted and actual outcomes, testers develop test cases in accordance with the flow scenario that moves from the controller layer to the service, repository, and finally the database.	Ensure accurate data flow between the controller layer and the service layer, the repository, and the database.
Functional Testing	By engaging with the application through the Graphical User Interface (GUI) and examining the outputs or results, you may confirm the application and its internal workings.	Testing professionals develop test scenarios based on functional requirements. Black-box testing techniques will be used to construct test scenarios. A better checklist will be	To verify adequate data acceptance, processing, retrieval, and the appropriate application of the business rules, all functional test cases have been conducted. Address boundary,

		created by gathering common flaws.	abnormal, and normal cases.
Non-functional Testing	Non-functional testing should improve the product's usability, effectiveness, maintainability, and portability. Reduces the production risk and expense linked to the product's non-functional features. The knowledge of current technology and product behaviour should be improved.	Testers design test scenarios using non-functional requirements as a basis. Testers write reports and run tests according to test scenarios.	The non-functional requirements listed in the software requirement specification must be complied with by all non-functional requirements.

Table 2. Description of testing types

2.2 Test Levels

Type of Tests	Test Level			
	Unit	Integration	System	Acceptance
Unit Test	X			
GUI Test		X	X	X
API Test		X		
Functional Testing	X	X	X	X
Non-functional Testing			X	X

Table 3. Test levels

2.3 Supporting Tools

Purpose	Tool	Vendor/In-house	Version
Create test plan	Google Docs	Google	Online
Create test report	Google Sheet	Google	Online
Manage test case	Google Sheet	Google	Online
Mange test result	Google Sheet	Google	Online
Test APIs	Postman	Postman Inc.	v11
Manage bugs	Gitlab issue	Gitlab	Online

Table 4. Tools test table

3. Test Plan

3.1 Human Resources

Worker/Doer	Role	Specific Responsibilities/Comments
ThuyDTT	Team member	● Create test cases and perform integration tests. ● Create test cases and perform system tests. ● Create test cases, checklists and perform acceptance tests.
ThuyDTT, NinhNT, ManhDD, PhucND	Team member	● Create test cases and perform unit tests. ● Summaries test reports.
NinhNT, HauNX, ManhDD	Team member	● Fix bugs front-end ● Log bugs front-end
NinhNT, ThuyDTT, PhucND, ManhDD	Team member	● Fix bugs back-end ● Log bugs back-end

ThuyDTT, NinhNT, ManhDD, HauNX, PhucND	Team member	• Review Test Plan. • Execute Test (IT, ST) • Review Test Report.

Table 5. Human Resources

3.2 Test Environment

Purpose	Tool	Provider
Write unit test documents	Google sheet	Google
Write test report documents	Google sheet, google doc	Google
Run UT	Visual studio code	Microsoft
Run IT	Postman, Chrome	Postman Inc.
Run ST, AT	Chrome, Microsoft Edge	Google, Microsoft
Tracking defects, issues and Q&A	Gitlab issue	Gitlab

Table 6. Test Environment

3.3 Test Milestones

Milestone Task		Start Date	End Date
Project Initiating	Create test plan	07/09/2024	09/09/2024
Iteration 1	Create test cases	09/09/2024	06/10/2024
	Execute unit test & integration test		
	Create test report		
Iteration 2	Create test cases	07/10/2024	27/10/2024
	Execute unit test & integration test & system test		

	Create test report		
Iteration 3	Create test cases	28/10/2024	17/11/2024
	Execute unit test & integration test & system test		
	Create test report		
Iteration 4	Create test cases	18/11/2024	08/12/2024
	Execute unit test & integration test & system test & acceptance test		
	Create test report		

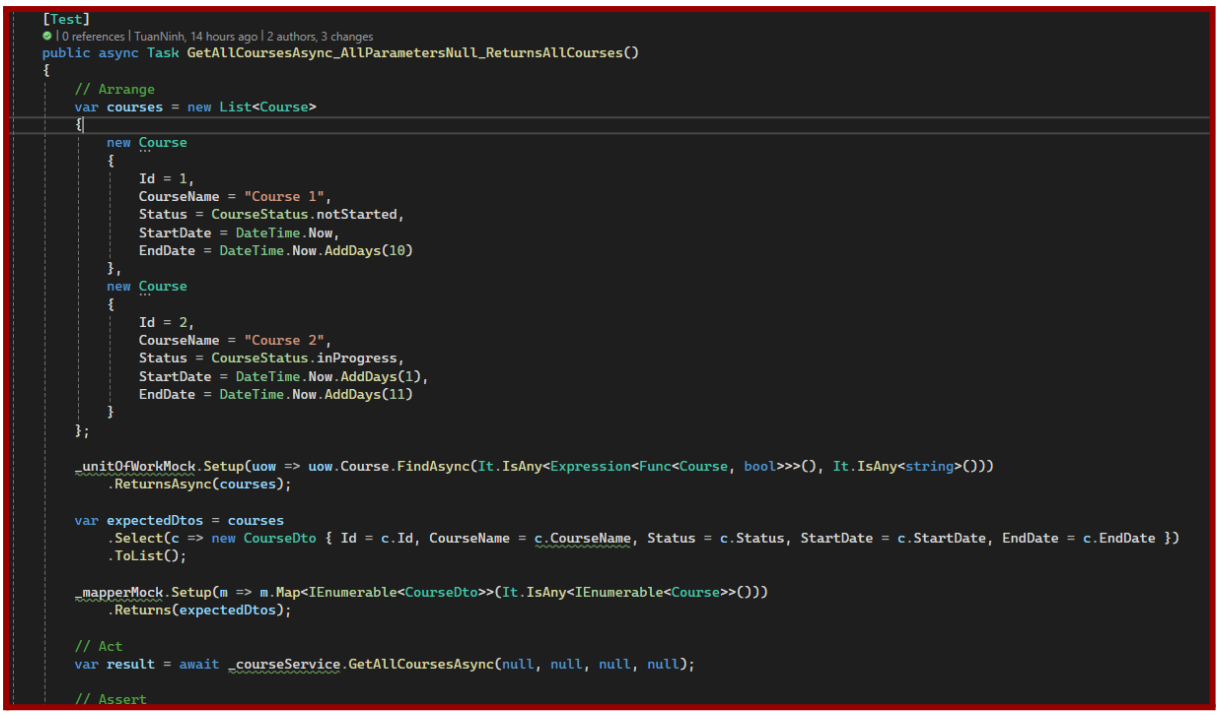
Table 7. Test Milestones

4. Test Cases

4.1. Unit test

Functional testing will be done on the server development side by the backend developer. Development teams use unit testing to achieve the following advantages:

- Detect errors as early as possible, develop and test faster.
- Save development time, detecting errors early means fewer late changes and easier to detect problems than when done at a later stage.
- Reduce the level of errors in production code.
- Documentation can be easily generated from tests.
- Make code changes and refactoring easier by improving code design.



```
[Test]
| 0 references | TuanNinh, 14 hours ago | 2 authors, 3 changes
public async Task GetAllCoursesAsync_AllParametersNull_ReturnsAllCourses()
{
    // Arrange
    var courses = new List<Course>
    {
        new Course
        {
            Id = 1,
            CourseName = "Course 1",
            Status = CourseStatus.notStarted,
            StartDate = DateTime.Now,
            EndDate = DateTime.Now.AddDays(10)
        },
        new Course
        {
            Id = 2,
            CourseName = "Course 2",
            Status = CourseStatus.inProgress,
            StartDate = DateTime.Now.AddDays(1),
            EndDate = DateTime.Now.AddDays(11)
        }
    };

    _unitOfWorkMock.Setup(uow => uow.Course.FindAsync(It.IsAny<Expression<Func<Course, bool>>>(), It.IsAny<string>()))
        .ReturnsAsync(courses);

    var expectedDtos = courses
        .Select(c => new CourseDto { Id = c.Id, CourseName = c.CourseName, Status = c.Status, StartDate = c.StartDate, EndDate = c.EndDate })
        .ToList();

    _mapperMock.Setup(m => m.Map<IEnumerable<CourseDto>>(It.IsAny<IEnumerable<Course>>()))
        .Returns(expectedDtos);

    // Act
    var result = await _courseService.GetAllCoursesAsync(null, null, null, null);

    // Assert
```

Figure 1: Unit Test example

4.2. Integration test.

Integration testing will involve incorporating the API into the application's interface to evaluate the data representation for the user. The tester will select test cases from the Test Report where actions are performed between functions or data is exchanged between modules. Then, Postman will be used to test the input-output data. If any errors occur during testing, especially related to unexpected data, the tester will log the errors or bugs on git issue and track the failed test case until it is resolved by the developers.

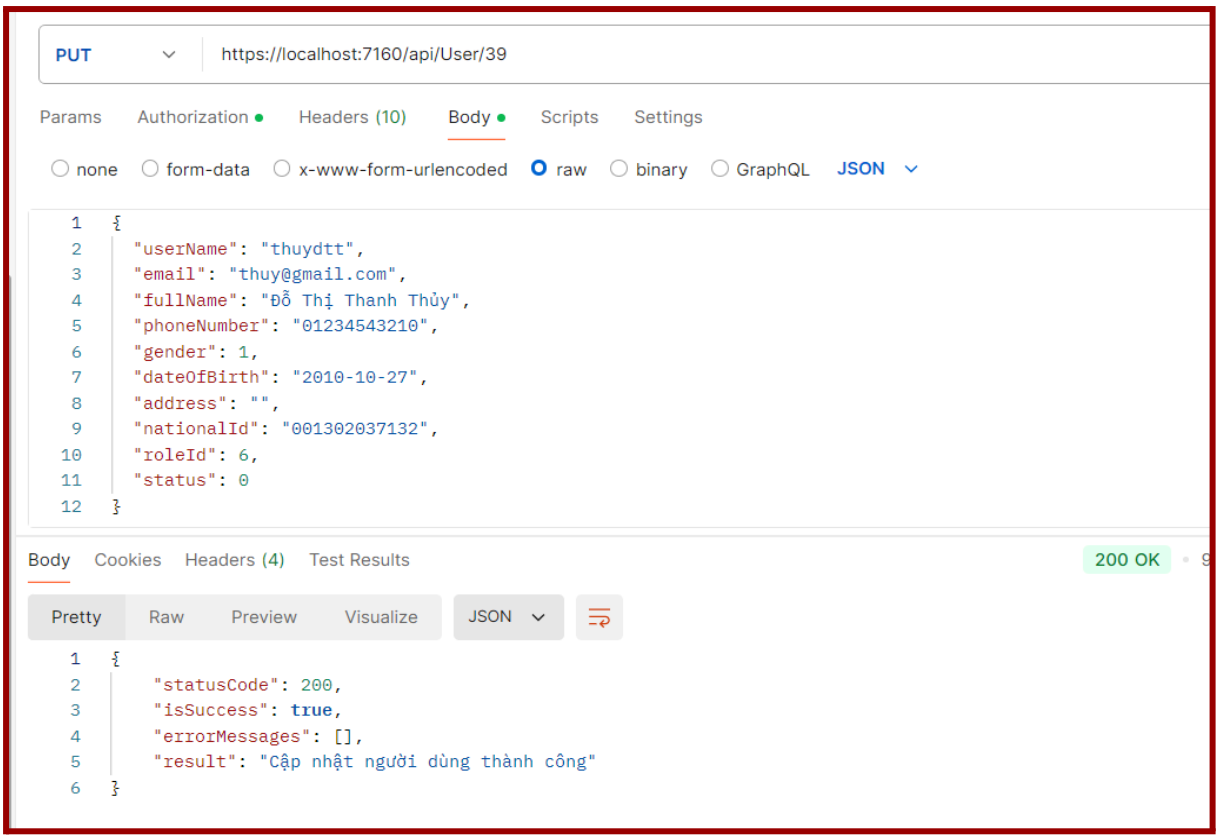
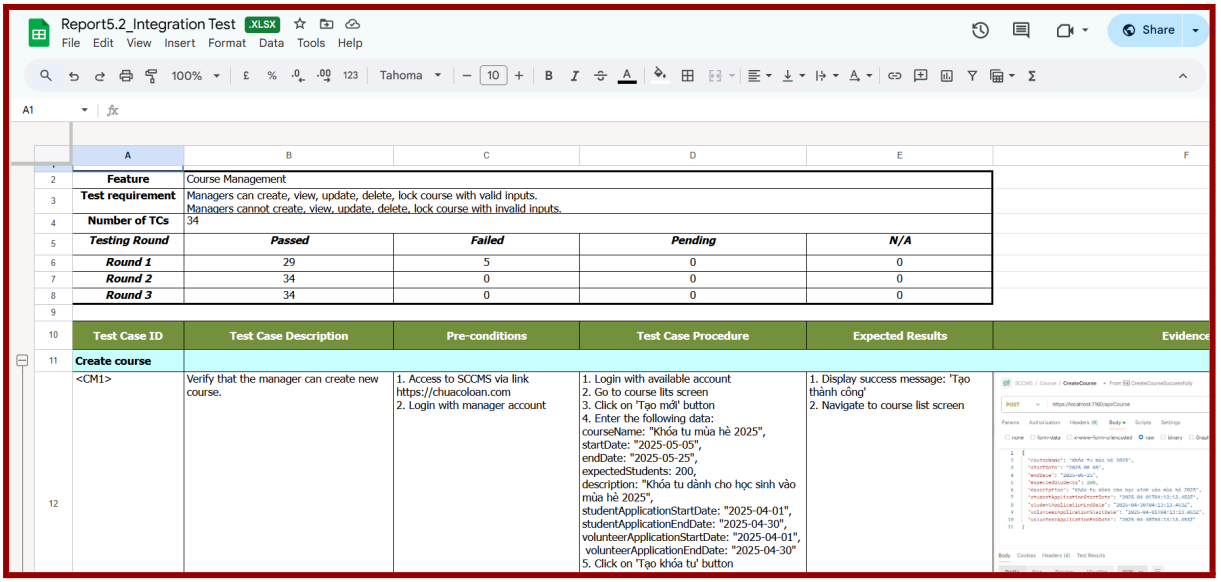


Figure 2: Integration Test example



Feature	Course Management				
Test requirement	Managers can create, view, update, delete, lock course with valid inputs. Managers cannot create, view, update, delete, lock course with invalid inputs.				
Number of TCs	34				
Testing Round	***Passed***	***Failed***	***Pending***	***N/A***	
Round 1	29	5	0	0	
Round 2	34	0	0	0	
Round 3	34	0	0	0	

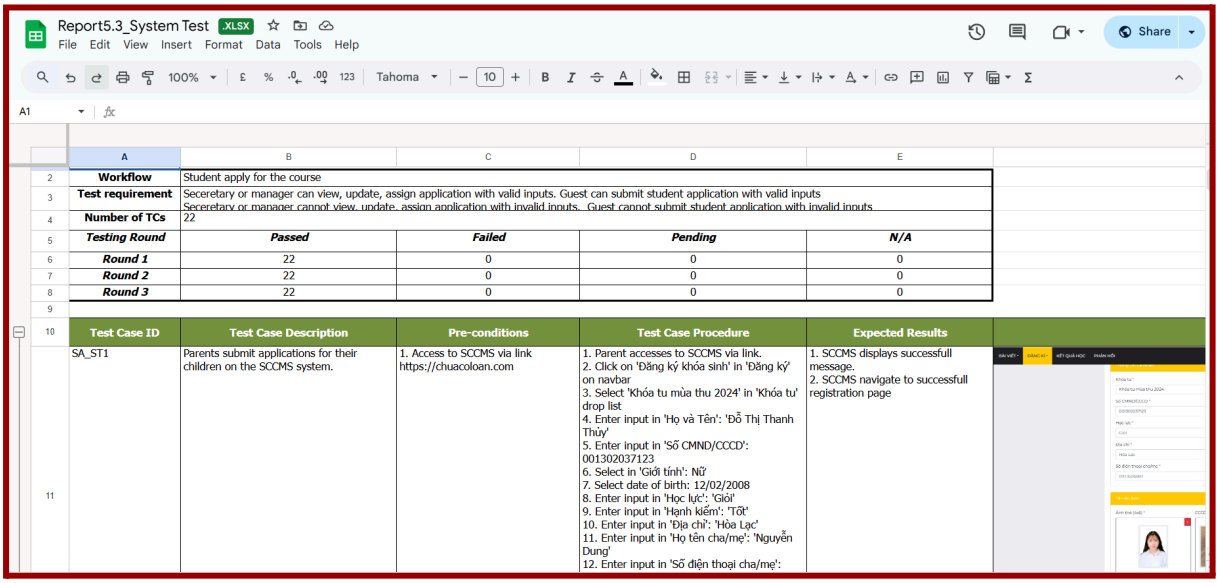
Test Case ID	Test Case Description	Pre-conditions	Test Case Procedure	Expected Results	Evidence
Create course					
<CM1>	Verify that the manager can create new course.	1. Access to SCCMS via link https://chuacoloan.com 2. Login with manager account	1. Login with available account 2. Go to course lits screen 3. Click on 'Tạo mới' button 4. Enter the following data: courseName: "Khóa tu mùa hè 2025", startDate: "2025-05-05", endDate: "2025-05-25", expectedStudents: 200, description: "Khóa tu dành cho học sinh vào mùa hè 2025", studentApplicationStartDate: "2025-04-01", studentApplicationEndDate: "2025-04-30", volunteerApplicationStartDate: "2025-04-01", volunteerApplicationEndDate: "2025-04-30" 5. Click on 'Tạo khóa tu' button	1. Display success message: 'Tạo thành công' 2. Navigate to course list screen	

Figure 3: Integration Test report

4.3. System test

This testing phase takes place after the functionality outlined in the requirements has been fully implemented. We will conduct testing using Edge, Chrome browser. The main goal of

this testing phase is to evaluate the performance of the system when the user operates it on the device, evaluating how the user interacts with the application in different scenarios.



Report5.3_System Test .XLSX

File Edit View Insert Format Data Tools Help

Share

Workflow	Student apply for the course
Test requirement	Secceretary or manager can view, update, assign application with valid inputs. Guest can submit student application with valid inputs Secceretary or manager cannot view, update, assign application with invalid inputs. Guest cannot submit student application with invalid inputs
Number of TCs	22

Testing Round	***Passed***	***Failed***	***Pending***	***N/A***
Round 1	22	0	0	0
Round 2	22	0	0	0
Round 3	22	0	0	0

Test Case ID	Test Case Description	Pre-conditions	Test Case Procedure	Expected Results	
SA_ST1	Parents submit applications for their children on the SCCMS system.	1. Access to SCCMS via link https://chuacoloan.com	1. Parent accesses to SCCMS via link. 2. Click on 'Đăng ký khóa sinh' in 'Đăng ký' on navbar 3. Select 'Khóa tu mùa thu 2024' in 'Khóa tu' drop list 4. Enter input in 'Họ và Tên': 'Đỗ Thị Thanh Thủy' 5. Enter input in 'Số CMND/CCCD': 001302037123 6. Select in 'Giới tính': Nữ' 7. Select date of birth: 12/02/2008 8. Enter input in 'Học lực': 'Giỏi' 9. Enter input in 'Hạnh kiểm': 'Tốt' 10. Enter input in 'Địa chỉ': 'Hòa Lạc' 11. Enter input in 'Họ tên cha/mẹ': 'Nguyễn Dung' 12. Enter input in 'Số điện thoại cha/mẹ':	1. SCCMS displays successfull message. 2. SCCMS navigate to successfull registration page	

Figure 4: System Test report

4.4. Acceptance test

Acceptance Testing is a level of software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.

ID	Checklist	Test Result	
		Pass	Fail
	GUI and Usability		
1	The website design takes inspiration from the following design: Figma Design	☑	☐
2	The screen is well-organized, providing a user-friendly interface.	☑	☐
3	The steps are short and easy to understand so that the user can interact with the application without training	☑	☐
4	The main functions are organized into tabs, facilitating easier access and navigation for users.	☑	☐
5	Links, buttons, and checkboxes are easily clickable.	☑	☐
6	Important commands are displayed as buttons with distinct background colors.	☑	☐
7	The displayed text is in Vietnamese for the convenience of most employees	☑	☐
8	The static text is clear, concise, and meaningful.	☑	☐
9	Pop-up menus are provided for the user to access information about an object's properties or perform specific tasks on the object.	☑	☐
10	System display notification message when meet trouble, error.	☑	☐

	Feature					
	Function	Test case description	Test Case Proceduce	Expected Result	Pass	Fail
13	Login	Users can login with username and password with authorized account	1. Access to SCCMS via link 2. Navigate to Login screen 3. Login with authorized account	1. Display login page 2. Navigate to homepage of authorized account 3. If the password and username is invalid or incorrect, the system will display an error message.	☑	☐
14	Import user list	Users can create multiple accounts by importing a user list file.	1. Login with manager account 2. Click on 'Danh sách người dùng' in 'Người dùng' on navbar 3. Click on 'Nhập từ tệp' button 5. Click on 'Chọn tệp' button 6. Choose file 7. Click on 'Xác nhận' button	1. SCCMS will display success message 2. Navigate to user list screen 3. Send account information via email to users 4. If the input fields are invalid, the system will report an error and cannot create successfully.	☑	☐
15	Create course	Users login with manager account and create course	1. Login with manager account 2. Click on 'Khóa tu' on navbar 3. Click on 'Tạo mới' button 5. Input following data: 'Tên khóa tu': 'Khóa tu mùa hè 2025' 'Thời gian bắt đầu khóa tu' :05/01/2025 'Thời gian kết thúc khóa tu':	1. SCCMS displays successfull message. 2. SCCMS redirects to "Danh sách khóa tu" screen. 3.The course selection dropdowns on other screens will have the newly added course added. 4. If the input fields are invalid, the	☑	☐
	Create student group	Users login with manager account	1. Login with manager account	1. Display success message: 'Tạo thành		

Figure 5: Acceptance Test report

5. Test Reports

5.1 Unit Test

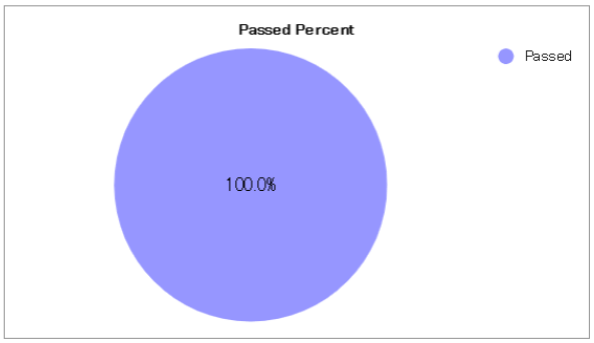
No	Function code	Passed	Failed	Untested	N	A	B	Total Test Cases
1	GetAllStudentApplication	19	0	0	18	0	1	19
2	GetStudentApplicationById	4	0	0	2	1	1	4
3	UpdateStatusStudentApplication	8	0	0	2	5	1	8
4	AutoAssignApplication	3	0	0	1	2	0	3
5	GetAllCourse	11	0	0	9	0	2	11
6	GetCourseById	4	0	0	1	2	1	4
7	CreateCourse	12	0	0	1	10	1	12
8	Updatecourse	12	0	0	1	10	1	12
9	DeleteCourse	4	0	0	1	2	1	4
10	GetAllUsers	21	0	0	13	5	3	21
11	ValidatePassword	9	0	0	1	7	1	9
12	GetAllStudent	24	0	0	10	12	2	24
13	GetStudentById	4	0	0	1	2	1	4
14	CreateStudent	23	0	0	10	11	2	23
15	Login	8	0	0	2	6	0	8
16	VerifyOTP	9	0	0	1	8	0	9
17	ResetPassword	11	0	0	1	9	1	11
18	IsValidEmail	7	0	0	2	5	0	7
19	CreateTeam	14	0	0	5	9	0	14
20	GetAllTeamsByCourseId	4	0	0	1	2	1	4
21	GetTeamById	4	0	0	1	2	1	4
22	UpdateTeam	14	0	0	5	9	0	14
23	DeleteTeam	4	0	0	2	2	0	4
24	CreateFeedback	9	0	0	1	7	1	9
25	GetFeedbackById	4	0	0	2	1	1	4
26	GetAllFeedback	7	0	0	6	0	1	7
27	DeleteFeedback	4	0	0	1	2	1	4

28	CreateRoom	5	0	0	2	2	1	5
29	GetRoomById	3	0	0	1	1	1	3
30	DeleteRoom	3	0	0	1	2	0	3
31	UpdateRoom	7	0	0	2	3	2	7
32	GetAllFreeTime	4	0	0	2	0	2	4
33	GetFreeTimeById	3	0	0	1	1	1	3
34	DeleleFreeTime	3	0	0	1	1	1	3
35	AssignStaffToShift	6	0	0	1	3	2	6
36	UpdateAssignment	7	0	0	1	4	2	7
37	UpdateAssignmentStatus	5	0	0	2	1	2	5
38	CreateUser	19	0	0	1	18	0	19
39	ChangeUserRoleAsync	8	0	0	1	7	0	8
40	MapRoleNameToRoleIdAsync	6	0	0	2	4	0	6
41	ResetPassword	11	0	0	1	9	1	11
42	GeneratePassword	3	0	0	2	1	0	3
43	GetUserById	4	0	0	1	3	0	4
44	ChangePassword	14	0	0	1	12	1	14
45	ChangeUserStatusAsync	5	0	0	1	4	0	5
46	CreatePost	7	0	0	1	6	0	7
47	UpdatePost	8	0	0	1	7	0	8
48	GetPostByIdAsync	3	0	0	3	0	0	3
49	GetAllPostsAsync	12	0	0	8	4	0	12
50	GettAllVolunteerCourse	17	0	0	9	7	1	17
51	UpdateVolunteerApplication	8	0	0	2	5	1	8
52	GetVolunteerApplicationById	4	0	0	1	2	1	4
53	GetVolunteerByCourseId	4	0	0	1	2	1	4
54	GetVolunteerById	4	0	0	1	2	1	4

50	GettAllVolunteerCourse	17	0	0	9	7	1	17
51	UpdateVolunteerApplication	8	0	0	2	5	1	8
52	GetVolunteerApplicationById	4	0	0	1	2	1	4
53	GetVolunteerByCourseId	4	0	0	1	2	1	4
54	GetVolunteerById	4	0	0	1	2	1	4

	Sub total	440	0	0	152	242	46	440

Test coverage	**100.00 %**
Test successful coverage	**100.00 %**
Normal case	**34.55 %**
Abnormal case	**55.00 %**
Boundary case	**10.45 %**



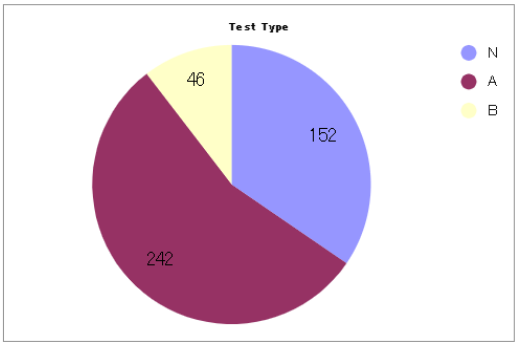


Figure 6. Unit Test Result

The reports of the Unit test procedure are shown details via: Report 5.1

5.2 Integration Test

No	Module code	Passed	Failed	Pending	N/A	Number of test cases
1	Authentication	38	0	0	0	38
2	Student Management	52	0	0	0	52
3	Course Management	34	0	0	0	34
4	Student Application Management	21	0	0	0	21
5	User Management	44	0	0	0	44
6	Student Group Management	24	0	0	0	24
7	Supervisor Management	5	0	0	0	5
8	Volunteer Management	51	0	0	0	51
9	Volunteer Application Management	18	0	0	0	18
10	Team Management	24	0	0	0	24
11	Feedback Management	15	0	0	0	15
12	Night Shift Management	36	0	0	0	36
13	Post Management	19	0	0	0	19
14	Report Management	21	0	0	0	21
	Sub total	402	0	0	0	402

Test coverage 100.00 %
Test successful coverage 100.00 %

Figure 7. Integration Test Result

The reports of the Integration test procedure are shown details via: Report 5.2

5.3 System Test

No	Module code	Passed	Failed	Pending	N/A	Number of test cases
1	User Authorization	33	0	0	0	33
2	Course management	18	0	0	0	18
3	Student group managerment	19	0	0	0	19
4	Team management	20	0	0	0	20
5	Post management	8	0	0	0	8
6	Student apply for the course	22	0	0	0	22
7	Student management	12	0	0	0	12
8	Volunteer apply for the course	22	0	0	0	22
9	Volunteer management	13	0	0	0	13
10	Nightshift management	41	0	0	0	41
11	Report management	7	0	0	0	7
12	Feedback management	12	0	0	0	12
	Sub total	227	0	0	0	227

Test coverage 100.00 %
Test successful coverage 100.00 %

Figure 8. System Test Result

The reports of the System test are shown details via: Report 5.3

5.4 Acceptance test

No	Yes	No	N/A	Number of test cases
1	45	0	0	45
Summary	45	0	0	45

Test coverage 100%
Test successful coverage 100%

Figure 9. Acceptance Test Result

The reports of the System test are shown details via: Report 5.4