9 Testing

9.1 Functional Testing

Functional testing involves testing the application against the business requirements. It incorporates all test types designed to guarantee each part of a piece of software behaves as expected by using uses cases provided by the design team or business analyst. These testing methods are usually conducted in order and include:

- Unit testing
- Integration testing
- System testing
- Acceptance testing

9.2 Non-functional Testing

Non-functional testing methods incorporate all test types focused on the operational aspects of a piece of software. These include:

- Performance testing
- Security testing
- Usability testing
- Compatibility testing

Reference:

1. https://smartbear.com/learn/automated-testing/software-testing-methodologies/

9.1 Functional Testing Structure

9.1.1 Unit Test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
Frontend	A single function/class can work correctly	• All unit tests are completed and passing • Archive at least 80% statement coverage for code tests	• Jest: Jest is a delightful JavaScript Testing Framework with a focus on simplicity
Backend	A single function/class can work correctly	• All unit tests are completed and passing • Archive at least 80% statement coverage for code tests	• Unittest: Unittest supports test automation, sharing test setup, aggregation of tests into collections, and the independence of tests and reporting frameworks • HTML TestRunner: build test • Django's Coverage: build coverage report

9.1.2 Integration Test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
API Test	Expose faults in the interface between frontend and backend, integrated links, data transfer between modules.	• All the API test cases have been executed	• Postman

9.1.3 System Test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
Web Test	Check if jumps between frontend pages are correct.	• Cover all use cases • Archive at least 80% statement coverage for code tests	• Jest: Jest is a delightful JavaScript Testing Framework with a focus on simplicity

9.1.4 Acceptance Test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
Acceptance Test	Confirm the system works as expected by end-users	• Pass all UAT cases	Not Applicable

9.2 Non-Functional Testing Structure

9.1.1 Performance Test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
Frontend	Functions can finish within reasonable time frame and with adequate resource usage	• Functions will not hang the system (parallelism considered) • All functions executes without using excessive resources and time	Not Applicable
Backend	Functions can finish within reasonable time frame and with adequate resource usage	• Functions will not hang the system (parallelism considered) • All functions executes without using excessive resources and time	Not Applicable

9.1.2 Security Test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
Database Test	System is not vulnerable to common attacks. e.g. SQL injection	• System is immune to common attacks • Sensitive information are kept encrypted	Not Applicable
Web Test	System is not vulnerable to common attacks. e.g. DDoS	• System is immune to common attacks	

9.1.3 Usability Test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
UX Test	Ensure ease-of-use and an intuitive interface	• Clear navigation • Reasonable Aesthetic	Not Applicable

9.1.4 Compatibility test

Part	Objectives	Exit Criteria	Tools (& Reasoning)
Compatibility Test	Confirm the system works on specified deploying environments	• Pass deployment tests	Not Applicable

9.3 Acceptance Testing

1. What is Acceptance Testing?

In engineering and its various subdisciplines, Acceptance Testing is a test conducted to determine if a specification or contract requirements are met.

User Acceptance Testing (UAT) is a type of testing performed by the end-user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration, and system testing is done.

To be noticed, this is a black-box testing technique where only the functionality is verified to ensure that the product meets the specified acceptance criteria (no need for design/implementation knowledge). Therefore, the client can escape those terrible technique details and focus on the functionality of the software project.

The formal process of successful test execution:

- using predetermined data, test cases are executed
- actual results are recorded
- actual and expected results are compared
- test results can be determined as 'Acceptance' if actual and expected results are the same

2.API-Level Acceptance Testing

In Sprint 1, the Backend Team has completed the development of some APIs with Acceptance Testing: Backend-Spint1-Acceptance Test

3. Requirement-Level Acceptance Testing Table

In this part, an Acceptance Testing Table is generated from 1.5 User stories. Our client can use this table to do Acceptance Testing conveniently.

Acceptance Criteria ID	Acceptance Test	Expected Results	Actual Results	Critical	Actor	Comment
US_1	Jimmy assesses students' work to import student projects from the confluence				Jimmy	
US_2	Jimmy views each project to see those imported projects				Jimmy	
US_3	Jimmy only keeps the projects from my subject(s) to delete a project that does not belong to my subject				Jimmy	
US_4	Jimmy assesses the product quality of each team based on the summary to see some metrics from a third-party code analysis tool				Jimmy	
US_5	Jimmy views the progress they made each week to view up to date statistical data				Jimmy	
US_6	Jimmy assesses the process quality of each team based on the summary to see the process quality summary of my project teams				Jimmy	
US_7	Jimmy measures and assesses a project team's process quality in terms of task management to view graphical data on a project team's Confluence activities				Jimmy	
US_8	Jimmy measures and assesses a project team's process quality in terms of task management to view graphical visualizations on a project team's JIRA task activities (ie, total vs completed vs remaining tasks)				Jimmy	
US_9	Jimmy measures and assesses a project team's process quality in terms of codebase management to view graphical visualizations on the number of code commits with proper code reviews and Continuous Integrations tests in GitHub/Bitbucket/GitLab				Jimmy	
US_10	Lucy assesses the communication of each team based on the summary to see the communication summary of my project teams				Lucy	
US_11	Lucy measures and assesses a project team's communication quality in Slack accurately by excluding irrelevant messages to let the system filter out irrelevant messages (emoji only messages) when gathering assessable communication data for each project team				Lucy	
US_12	Lucy measures and assesses a project team's communication quality in terms of chat frequency in Slack to view statistical data on the total number of messages sent over assessable Slack channels for each project team				Lucy	

US_13	Lucy measures and assesses a project team's meeting frequency and meeting quality to view each meeting information including meeting minutes for each project team				Lucy	
US_14	Lucy assesses the individual contribution of each team member based on the summary to see the individual contribution summary of each team member in my project teams				Lucy	
US_15	Lucy assesses the individual contribution of each team member based on their pull requests to view statistical data on the number of pull requests involving peer reviews made per team member in GitHub/Bitbucket/GitLab				Lucy	
US_16	Lucy assesses the individual contribution of each team member based on the number of lines of code they have written to view statistical data on the number of lines of executable code made per team member				Lucy	
US_17	Lucy assesses the individual contribution of each team member based on their task management to view statistical data on the number of completed tasks/tickets on JIRA by each team member				Lucy	
US_18	Lucy assesses the individual contribution of each team member based on their individual contribution to view statistical data on the number of complete tasks from JIRA by each team member				Lucy	
	Lucy assesses the individual contribution of each team member based on their documentation work to view statistical data on publishing and editing activity made by each team member on Confluence				Lucy	
US_19	Lucy assesses the individual contribution of each team member based on their attendance in meetings to view statistical data on meeting attendance of each team member				Lucy	
US_20	Lucy makes any changes to the links of JIRA, Confluence, and Git in case some project teams might want to change their JIRA/Confluence/Git destination to change the URLs of JIRA, Confluence, and Git				Lucy	

Reference:

1. Wikipedia: https://en.wikipedia.org/wiki/Acceptance_testing
2. Example: Acceptance Testing
3. What is User Acceptance Testing (UAT)? with Examples:https://www.guru99.com/user-acceptance-testing.html
4. What Is Acceptance Testing (A Complete Guide):https://www.softwaretestinghelp.com/what-is-acceptance-testing/#Why_Acceptance_Tests

9.4 Testing Progress

Overall Progress: 15/15, 100%

Checklist

Web Page	Functionality	Front-End Owner	Front-End Implementation	Back-End Owner	Back-End Implementation	Joint Testing Passed
Login	Login	Fengrui Zhang		Boyang Sun		
Coordinator Home	Search projects	Sai Zhang		Pin Wang		
	Import projects	Ruofan Zhang		Boyang Sun		
	Show imported projects	Sai Zhang		Jingdan Cui		
	Delete imported projects	Ruofan Zhang		Pin Wang		
Project Home	Show team members	Yuhang Xie		Pin Wang		
Product Quality	Show code metrics	Fengrui Zhang		Jinzhe Shan		
Process Quality	Show the number of Confluence pages per day	Ruofan Zhang		Pin Wang		
	Show the number of Github commits per day	Ruofan Zhang		Chongjing ZHANG		
	Show the number of Jira tickets per day	Ruofan Zhang		Fu Xie Haoyu Qin		
Communication Quality	Show the meeting minutes	Zixin Ye		Jingdan Cui		
Individual Contribution	Show the number of Confluence pages edited by each team member	Zixin Ye		Boyang Sun		
	Show the number of Github commits made by each team member	Zixin Ye		ZISHENG CHENG		
	Show the number of Jira tickets done by each team member	Zixin Ye		Haoyu Qin F u Xie		
Configuration	Submit the configuration for a team	Jingyu LI		Fu Xie		

9.5 Super User Account

As the client requires a super user, we created a super user account.

Username: admin

Password:123