Proposal:

The purpose of this document is to define and present the test cases for Eratos website, covering the test cases for the system use cases.

Target users:

This document is mainly designed for those responsible for executing the test cases for back-end in this project.

Functional Requirements Summary:

User Story	User Story Title	Test ID	Test Case Title
1	get all active modules	1	User front-end is able to get all active modules
2	Create tasks	2	user front-end is able to create a task
3	Get task information	3.1	User/Admin front-end is able to sync the task information
		3.2	User/admin front-end is able to get the task information
4	Get order history and status	4.1	User/admin front-end is able to sync the order information
		4.2	User/admin front-end is able to get the order history
5	Create new users	5	User/admin front-end is able to register as a new user

6	Get user information and identity	6.1	User front-end is able to the user information
		6.2	Admin front-end is able to all user information
7	Update user information and identity	7	User/admin front-end is able to update their user information
8	Create and modify modules	8.1	Admin front-end is able to create a resource/modules
		8.2	Admin front-end is able to modify a resource/modules
9	Get all modules	9	Admin front-end is able to get all available modules

Test case for Story 1:

This test case is only for the user front-end, before creating a task, users need to check all active modules which are the acceptable modules or not.

Test ID		1
Test Case Title		User front-end is able to get all active modules
Test Type		Functional
Objective		Verify that user front-end is able to get all active modules
Pre-conditions		1. The admin front-end had created modules 2. The admin front-end should make these modules active.
Steps		1. send the request to the server
Expected Outcome		User front-end can get the list of all active modules
Acceptance Criteria	Given	Make the request
	When	I parse the request to the Eratos server
	Then	I should get the list of all active modules.
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test case for Story 2:

This test case is for the user front-end to create a task for the back-end to process their requests.

Test ID		2
Test Case Title		user front-end is able to create a task
Test Type		Functional
Objective		Verify that the user front-end is able to create a task after finishing the payment.
Precondition		1. Login to get their user uri 2. Get the resource uri 3. Get the resource name 4. Get the payment ID & price from front-end 5. Get the geometry info from front-end
Steps		1. Take priority to this task 2. Send the request with these parameters
Expected Outcome		Output success message with TaskID
Acceptance	Given	Received user's data

Criteria	When	I create a task based on provided data
	Then	I should get the success message.
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test case for Story 3:

These test cases are for the user & admin front-end to sync/get all tasks status back to users. There are two test cases, one is for the user front-end to sync the task information from the eratos server and the other is for the user front-end to be able to get their task information.

Test ID		3.1
Test Case Title		User/admin front-end is able to sync the task information
Test Type		Functional
Objective		Verify that the user front-end syncs the task information in our database with eratos.
Precondition		1. the task status is incomplete (Queued or Processing)
Steps		1. Connect to back-end database 2. check the status is “Queued” or “Processing” 3. send the request to Eratos server.
Expected Outcome		Output the success message “Database is up to date”
Acceptance Criteria	Given	check the task status and connect to the database
	When	I send the request to Eratos server
	Then	I can get the success message.
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

story ID	3.2
Test title	User/admin front-end is able to get the task information
Test Type	Functional

Objective		Verify that the users can get the latest task status.
Precondition		1. Sync the task info 2. Get the user uri
Steps		1. Find the task information by user uri
Expected Outcome		Users can get the task information including task id, priority, state, type, resource order id and so on.
Acceptance Criteria	Given	User uri
	When	I use the user uri as parameter to find task information
	Then	I can get the string of all tasks information.
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test case for Story 4:

These test cases are for the user & admin front-end to sync/get the order history back to users. There are two test cases, one is for the user front-end to sync the order information from the eratos server and the other is for the user front-end to be able to get their order history.

Test ID		4.1
Test Case Title		User/admin front-end is able to sync the order information
Test Type		Functional
Objective		Verify that the users sync the order information in our database with eratos.
Precondition		2. the order status is incomplete (Queued or Processing)
Steps		4. Connect to back-end database 5. check the order status is “Queued” or “Processing” 6. send the request to Eratos server.
Expected Outcome		Output the success message “Database is up to date”
Acceptance Criteria	Given	check the order status and connect to the database
	When	I send the request to Eratos server
	Then	I can get the success message.
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test ID		4.2
Test Case Title		User/admin front-end is able to get the order history
Test Type		Functional
Objective		Verify that the users can get the latest order history.
Precondition		1. Sync the order info 2. Get the user uri
Steps		1. Find the order information by user uri
Expected Outcome		Users can get the order information including order id, price, status, order time, user id and payment id.
Acceptance Criteria	Given	User uri
	When	I use the user uri as parameter to find order information in the database
	Then	I can get the string of all order information.
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test case for Story 5:

This case is for the user & admin front-end, back-end needs to store their user data for the further operation because we cannot get the data from Eratos directly.

Test ID		5
Test Case Title		User/admin front-end is able to register as a new user
Test Type		Functional
Objective		Register new users for our back-end database
precondition		1. get user data from front-end
steps		1. insert these data to the back-end database.
Expected Outcome		output the success message with username.
Acceptance Criteria	Given	user data from front end
	When	I get the user info from the front-end and then insert it to our database.
	Then	I should get the successful message.
Test result		PASS
Notes		

Tester	En-Wen TSAI
Date	28/05/2021

Test case for Story 6:

These cases are for the user & admin front-end, user front-end can get their user information and admin front-end can access all user information. There are two sub-cases, one is for users to be able to get their user information and the other is admin can get all user information.

Test ID		6.1
Test Case Title		User front-end is able to the user information
Test Type		Functional
Objective		Verify that user front-end can get their user details from the database
Precondition		1. Get the user uri
Steps		1. Query with user uri to their user information.
Expected Outcome		Output the success message with a string including EratosUserID, Email, Auth0ID, UserID, and name.
Acceptance Criteria	Given	Login to get user uri
	When	I query with valid user uri to the database
	Then	I should get the message about the user details.
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test ID	6.2
Test Case Title	Admin front-end is able to all user information
Test Type	Functional
Objective	Admin front-end can get all user information
Precondition	1. Start index of user table 2. End index of user table 3. The range between start and end must be less than 100.
Steps	1. Query with start point and end point of the user ids.
Expected Outcome	output the success message with a string including eratosUserId, email, Auth0ID, UserID, and name. Each user is separated by “,”.

Acceptance Criteria	Given	Put the given start and end point
	When	I send the query to the database
	Then	I should get the message about specific user details.
Test result		PASS
Notes		[1] If an admin wants to get more than a hundred user information, he/she should query more than one time.
Tester		En-Wen TSAI
Date		28/05/2021

Test case for Story 7:

This case is for the user & admin front-end, both can update their own information in the database. However, only the admin can edit the isAdmin value.

Test ID		7
Test Case Title		User/admin front-end is able to update their user information
Test Type		Functional
Objective		Users and admins can update user information
Precondition		1. Had already registered as an user/admin.
Steps		1. Make some changes in their personal information. ex: name, email, info or isAdmin.
Expected Outcome		Output the success message “The user info is up to date.”.
Acceptance Criteria	Given	Make some changes
	When	I update these changes to the database
	Then	I should get the success message.
Test result		PASS
Notes		[1] Only admins can change the isAdmin parameter to true or false.
Tester		En-Wen TSAI
Date		28/05/2021

Test case for Story 8:

These cases are only for the admin front-end, admin is able to add new modules, modify modules or toggle them on and off based on their needs. There are two cases, one is for admins to create new modules and the other is for admins to modify the modules including activating or deactivating them.

Test ID		8.1
Test Case Title		Admin front-end is able to create a resource/modules
Test Type		Functional
Objective		Verify that admins are able to create a new resource for users.
precondition		
Steps		1. Create a module name 2. Create a module schema 3. Make this module is active or not
Expected Outcome		Get the success message
Acceptance Criteria	Given	Put needed parameter
	When	I send these parameters to the server
	Then	I should get the success message from the server
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test ID		8.2
Test Case Title		Admin front-end is able to modify a resource/modules
Test Type		Functional
Objective		Verify that admins are able to modify the existing resource.
precondition		1. admin had already created a resource
Steps		1. Change the module name 2. Change the module schema 3. Change the module status to inactive.
Expected Outcome		Get the success message
Acceptance Criteria	Given	Put changed parameters
	When	I send these parameters to the server
	Then	I should get the success message from the server
Test result		PASS
Notes		
Tester		En-Wen TSAI
Date		28/05/2021

Test case for Story 9:

This case is only for the admin front-end, admin front-end is able to get available modules for a range between one to hundred.

Test ID		9
Test Case Title		Admin front-end is able to get all available modules
Test Type		Functional
Objective		Verify that admin front-end is able to get the all available modules
Precondition		1. Start index of module id 2. End index of module id 3. The range between start and end must be less than 100.
Steps		1. Query with start point and end point of module id
Expected Outcome		Output the success message with a string including Moduleid, Module name, Module schema, and IsActive or not. Each module is separated by “,”
Acceptance Criteria	Given	Put the given start and end point
	When	I send the query to the database
	Then	I should get the message about available modules’ details.
Test result		PASS
Notes		[1] If an admin wants to get more than a hundred user information, he/she should query more than one time.
Tester		En-Wen TSAI
Date		28/05/2021