INF 370 PROJECT

IMOS

PRESENTED BY INFAS

Contents

Material Request Report

Generate material request report

IMOS system		
DATE:	**AUTHOR:**Lehlogonolo Lekgera	**VERSION:1.0**
USE CASE NAME:	**Generate material request report**	**USE CASE TYPE**
USE CASE ID:	**6.1**	**Business Requirements:** ☐
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** ☐
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM ACTOR:	None	
OTHER PARTICIPATING ACTORS:	None	

OTHER INTERESTED STAKEHOLDERS:	None	
DESCRIPTION:	This use case describes when an admin wants generate a material request report. The admin will navigate to the report screen and select to generate a material request report. The admin will filter the report using the material request status.	
PRE-CONDITION:	Admin must be logged in	
TRIGGER:	The admin wants to generate a material request report	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
	Step 1: The admin wants to generate a material request report	Step 2: The system display the report screen. The screen will contain a list of card with the different reports the admin can generate.
	Step 3: The Admin select the status and selects to generate the report.	Step 4: The system will retrieve material request data using the status as a parameter.
		Step 5: The system display the material request report and an option to download the report. **[ALT]**
ALTERNATE COURSES:	Alt Step 5: The system does not find data to generate the report. • Displays a message to the admin. • System returns to step 2.	
CONCLUSION:	The system generates the material request report	
POST-CONDITION:	The admin is able to download the report.	
BUSINESS RULES:	• Only the admin can generate a material request report.	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	• The system is offline	
ASSUMPTIONS:	None	
OPEN ISSUES:	None	

Incident Report

IMOS system		
DATE:	**AUTHOR:** Chase Nel	**VERSION:1.0**
USE CASE DETAILS:	**Incident Report**	**USE CASE TYPE**
USE CASE ID:	**13.4**	**Business Requirements:** □
PRIORITY:	Medium	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** □
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM ACTOR:	None	

Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This use case describes the event where a The Admin would want to cire the number of incidents that has occurred per a project. This would be used for clients to understand why the project took longer than expected.	
Pre-condition:	The Admin has been logged onto the system. The Admin must be in the Report card screen	
Trigger:	The Admin wishes to view the incident report	
Typical Course	**Actor Action:**	**system Response:**
	Step 1: The Admin will request to view the report by clicking on the incident report card	**Step 2:** The system will generate the incident report. The app header with a side menu and a personal details button • An H1 tag with the heading: Incident Report • A paragraph tag : "Developer by:" and the name of the admin who developed it • A paragraph tag: "Date Developed:" with the current date at which it is was developed • The TWS Logo • A bar chart using Chart.js for the incidents to projects. which is retrieved by the system using a GET function from the SQL Database • A pie chart using CHart.JS for the Incident to Project. which is retrieved by the system using a GET function from the SQL

		DATABASE • A BUTTON CALLED DOWNLOAD WHICH USES H2MLCANVAS AND JSPDF TO DOWNLOAD THE REPORT SCREEN INTO A PDF AND INTO THE DOWNLOADS OF A LAPTOP
	Step 3: The admin will be able to click the download button which will download the Report into a PDF onto his/her device	
ALTERNATE COURSES:	None	
CONCLUSION:	The system Downloads the Incident Report onto their laptop for further viewing in the future	
POST-CONDITION:	The Incident Report will continue to gather the information for future uses as new projects are started.	
BUSINESS RULES:	Only Admins will be able to view a Incident Report	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None	
ASSUMPTIONS:	None	
OPEN ISSUES:	None	

Invoice Subsystem

Add Invoice

IMOS system		
DATE:	**AUTHOR:** Sydney Zinyama	**VERSION:1.0**
USE CASE NAME:	**Add Invoice**	**USE CASE TYPE**
USE CASE ID:	**13.1**	**Business Requirements:** □
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** □
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM ACTOR:	None	

OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	None	
DESCRIPTION:	This use case describes the process of adding Invoices on the system. The use case begins when the admin decides on adding new Invoice details. The system verifies the input data type and ensures correct format of Invoice details entered by the Admin. The use case ends when the system displays a success notification that Invoice details were Submitted.	
PRE-CONDITION:	The Invoice details do not yet exist on the system. The Admin is logged onto the system.	
TRIGGER:	The admin would like to Submit new Invoice details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
		Step1: The system invokes Use Case 12.3 Search/View Invoice displays the Invoice screen.
	Step2: The Admin selects the option to add a new Invoice.	**Step3:** The system displays the Submit Invoice form and prompts the Admin to input Invoice details. ⌦ Project ⌦ Task ⌦ Amount
	Step4: The Admin enters the Invoice details indicated in **Step3.**	
	Step5: The Admin selects the option to Save new Invoice details onto the system.	**Step6:**The system validates and verifies that the input entered matches data type and have correct format.[ALT]
		Step7: The system prompts the Admin to select confirmation to **Save** Invoice details.
	Step8: The admin selects the option to save details [ALT].	**Step9:** The system generates a unique Invoiced, incremented from the last retrieved Invoice_Id then Submits the details in the Invoice Entity.
		Step10: The system displays a success notification that Invoice

		details were Submitted.
ALTERNATE COURSES:	Alt Step6: The system displays validation errors. ⌦ Invalid data type or incorrect input format.	
	Alt Step8: The Admin cancels request to Submit Invoice. ⌦ Return to View Invoice screen.	
CONCLUSION:	The system displays a success notification that Invoice details were Submitted.	
POST-CONDITION:	The Invoice Entity has a new Submitted Invoice record.	
BUSINESS RULES:	Only authorised Users can Submit new Invoice details onto the system.	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None	
ASSUMPTIONS:	None	
OPEN ISSUES:	None	

Update Invoice

IMOS system		
DATE:	**AUTHOR:** Sydney Zinyama	**VERSION:1.0**
USE CASE NAME:	**Update Invoice**	**USE CASE TYPE**
USE CASE ID:	**13.2**	**Business Requirements:** □
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** □
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM	None	

ACTOR:		
OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	None	
DESCRIPTION:	This use case describes the event where the Admin updates existing Invoice details. The use case starts when the Admin requests to update specific Invoice details. The Admin searches and provides updated input for the specific Invoice they would like to update. The system verifies the data type and ensures correct format of updated details provided The use case ends when the system displays a success notification that Invoice details were updated.	
PRE-CONDITION:	The Admin is logged onto the system. The Client details exist on the system.	
TRIGGER:	The Admin wishes to update Invoice details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
		Step1: The system invokes Use Case 12.3 Search/View Invoice displays the Invoice screen.
	Step2: The Admin selects the option to update Invoice details.	**Step3:** The system displays the updated Invoice form and prompts the Admin to input updated Invoice details. ⌦ Project ⌦ Task ⌦ Amount
	Step4: The Admin enters the Invoice details indicated in **Step3.**	
	Step5: The Admin selects the update option to **Save** updated Invoice details onto the system.	**Step6:**The system validates and verifies that the input entered matches data type and have correct format.[ALT]
		Step7: The system prompts the Admin to select confirmation to **Save updated** Invoice details.
	Step8: The admin selects the option to save details [ALT].	**Step9:** The system displays a success notification that Invoice details were updated.

ALTERNATE COURSES:	**Alt Step6:** The system displays validation errors. ⌦ Invalid data type or incorrect input format.
	Alt Step8: The Admin cancels request to update Invoice. ⌦ Return to View Invoice screen.
CONCLUSION:	The system displays a success notification that Invoice details were updated.
POST-CONDITION:	The Invoice Entity has updated the Invoice record.
BUSINESS RULES:	Only authorised Users can add new Invoice details onto the system.
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None
ASSUMPTIONS:	None
OPEN ISSUES:	None

Search Invoice

IMOS system		
DATE:	**AUTHOR:** Sydney Zinyama	**VERSION:1.0**
USE CASE NAME:	**Search Invoice**	**USE CASE TYPE**
USE CASE ID:	**13.2**	**Business Requirements:** □
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** □
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY SYSTEM ACTOR:	None	

OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	None	
DESCRIPTION:	This use case describes the process when an Admin searches for Invoice details on the system. The use case begins when the Admin requests to search for specific Invoice details. The use case concludes when the displays the results of Invoice search for the Admin to View.	
PRE-CONDITION:	The Invoice details to be searched for exist on the system. The Admin is logged onto the system.	
TRIGGER:	The Admin wishes to search for specific Invoice details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
	Step1: The Admin wishes to search for specific Invoice details on the system.	**Step2:**The system displays in table a list of Invoice details, retrieved from the **Invoice Entity** with the following attributes: ⌦ Project ⌦ Task ⌦ Amount
	Step3: The Admin clicks "Search Table" to input the search keyword or criteria [ALT].	**Step4:** The system filters and displays specific Invoice details matching search parameters.
ALTERNATE COURSES:	Alt Step1: The Admin selects to update Invoice details. ⌦ Redirects to the Update Invoice screen.	
	Alt Step3: The Admin selects to delete Invoice details. ⌦ Displays delete confirmation prompt.	
	Alt Step3: The Admin selects to Add New Invoice details. ⌦ Redirects to Add Invoice screen.	
CONCLUSION:	The system displays the results of Invoice search for the Admin to View.	
POST-CONDITION:	Specific Invoice details are displayed for Admin to view.	
BUSINESS RULES:	Only authorised Users will be allowed to search and view Invoice details.	
IMPLEMENTATION CONSTRAINTS AND	None	

SPECIFICATIONS:	
ASSUMPTIONS:	None
OPEN ISSUES:	None

Delete Invoice

IMOS system		
DATE:	**AUTHOR:** Sydney Zinyama	**VERSION:1.0**
USE CASE DETAILS:	**Delete Invoice**	**USE CASE TYPE**
USE CASE ID:	**13.4**	**Business Requirements:** □
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** □
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM ACTOR:	None	

OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	None	
DESCRIPTION:	This use case describes the event where an Invoice is removed from the system. The use case starts when the Admin wishes to remove Invoice details. The Admin searches for specific Invoice details then requests to delete Invoices. The use case concludes when the system display a success notification that the Invoice has been deleted from the system.	
PRE-CONDITION:	The Admin has been logged onto the system. The Invoice details exist on the system.	
TRIGGER:	The Admin wishes to remove Invoice details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
		Step1: The system invokes Use Case 12.3 "Search Invoice"
	Step2: The Admin requests to remove Invoice details from the system.	**Step3:** The system prompts the Admin to confirm to remove details from the system.
	Step4: The Admin selects the option to proceed with removing the Invoice permanently from the system [ALT].	**Step5:** The system displays a successful snack bar message that the Invoice was deleted, successfully.
ALTERNATE COURSES:	Alt Step4: The Admin selects the option to cancel Invoice removal from the system.	
CONCLUSION:	The system displays a snack bar with success removing Invoice details. "Invoice details deleted"	
POST-CONDITION:	An Invoice has been deleted and the Invoice table updated.	
BUSINESS RULES:	Only authorised Users will be allowed access to Invoice management within the system.	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None	
ASSUMPTIONS:	None	
OPEN ISSUES:	None	

Stock Take Subsystem

Add Stock Take

IMOS system		
DATE:	**AUTHOR:** ARMIN WESSELS	**VERSION:1.0**
USE CASE DETAILS:	**Add Stock Take**	**USE CASE TYPE**
USE CASE ID:	**13.1**	**Business Requirements:** □
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** □

PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM ACTOR:	None	
OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	Owner	
DESCRIPTION:	This use case describes the event where Stock Take record is added to the system. The use case starts when the admin wishes to add new Stock Take details. The admin searches for specific Stock Take details then requests to add new Stock Takes. The use case concludes when the system displays a success notification that the Stock Take has been successfully added to the system.	
PRE-CONDITION:	The admin has been logged onto the system. The Stock Take details do not exist on the system.	
TRIGGER:	The admin wishes to add new Stock Take details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
		Step 1: The system invokes Use Case "Search Stock Take"
	Step 2: The Admin requests to add Stock Take details to the system.	**Step 3:** The system prompts the admin to confirm adding the details to the system.
	Step 4: The Admin selects the option to proceed with adding the Stock Take to the system. [ALT].	**Step 5:** The system displays a successful snack bar message that the Stock Take was successfully added.
ALTERNATE COURSES:	ALT Step 4: The Admin selects the option to cancel adding Stock Take to the system.	
CONCLUSION:	The system displays a snack bar with success adding Stock Take details. "Stock Take details successfully added"	
POST-CONDITION:	A Stock Take record has been added and the Stock Take table updated.	
BUSINESS RULES:	Only authorised Users will be allowed access to Stock Take management within the system.	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None	

ASSUMPTIONS:	None
OPEN ISSUES:	None

Update Stock Take

IMOS system		
DATE:	**AUTHOR:** ARMIN WESSELS	**VERSION:1.0**
USE CASE DETAILS:	**Update Stock Take**	**USE CASE TYPE**
USE CASE ID:	**13.2**	**Business Requirements:** □
PRIORITY:	High	**System Analysis:** ☑

SOURCE:	IMOS Functional Requirements	**System Design:** □
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM ACTOR:	None	
OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	Owner	
DESCRIPTION:	This use case describes the event where Stock Take record is updated on the system. The use case starts when the admin wants to update Stock Take details. The Admin searches for specific Stock Take details then requests to update Stock Takes. The use case concludes when the system displays a success notification that the Stock Take has been successfully updated to the system.	
PRE-CONDITION:	The admin has been logged onto the system. The Stock Take details exist on the system.	
TRIGGER:	The admin wishes to update Stock Take details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
		Step 1: The system invokes Use Case "Search Stock Take"
	Step 2: The Admin requests to update Stock Take details from the system.	**Step 3:** The system prompts the admin to confirm details update to the system.
	Step 4: The Admin selects the option to proceed with updating the Stock Take to the system. [ALT].	**Step 5:** The system displays a successful snack bar message that the Stock Take was successfully updated.
ALTERNATE COURSES:	ALT Step 4: The Admin selects the option to cancel Stock Take update from the system.	
CONCLUSION:	The system displays a snack bar with success updating Stock Take details. "Stock Take details successfully updated"	
POST-CONDITION:	A Stock Take has been updated and the Stock Take table updated.	
BUSINESS RULES:	Only authorised Users will be allowed access to Stock Take management within the system.	

IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None
ASSUMPTIONS:	None
OPEN ISSUES:	None

Delete Stock Take

IMOS system		
DATE:	**AUTHOR:** Sydney Zinyama	**VERSION:1.0**
USE CASE DETAILS:	**Delete Stock Take**	**USE CASE TYPE**
USE CASE ID:	**13.4**	**Business Requirements:** □

PRIORITY:	High	**System Analysis:**	☑
SOURCE:	IMOS Functional Requirements	**System Design:**	□
PRIMARY BUSINESS ACTOR:	Admin		
PRIMARY THE SYSTEM ACTOR:	None		
OTHER PARTICIPATING ACTORS:	None		
OTHER INTERESTED STAKEHOLDERS:	None		
DESCRIPTION:	This use case describes the event where a Stock Take is removed from the system. The use case starts when the Admin wishes to remove Stock Take details. The Admin searches for specific Stock Take details then requests to delete Stock Takes. The use case concludes when the system display a success notification that the Stock Take has been deleted from the system.		
PRE-CONDITION:	The Admin has been logged onto the system. The Stock Take details exist on the system.		
TRIGGER:	The Admin wishes to remove Stock Take details on the system.		
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**	
		Step1: The system invokes Use Case 12.3 "Search Stock Take"	
	Step2: The Admin requests to remove Stock Take details from the system.	**Step3:** The system prompts the Admin to confirm to remove details from the system.	
	Step4: The Admin selects the option to proceed with removing the Stock Take permanently from the system [ALT].	**Step5:** The system displays a successful snack bar message that the Stock Take was deleted, successfully.	
ALTERNATE COURSES:	Alt Step4: The Admin selects the option to cancel Stock Take removal from the system.		
CONCLUSION:	The system displays a snack bar with success removing Stock Take details. "Stock Take details deleted"		
POST-CONDITION:	A Stock Take has been deleted and the Stock Take table updated.		
BUSINESS RULES:	Only authorised Users will be allowed access to Stock Take management		

	within the system.
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None
ASSUMPTIONS:	None
OPEN ISSUES:	None

Employee Attendance

Complete Employee Attendance

IMOS system		
DATE:	**AUTHOR:** ARMIN WESSELS	**VERSION:1.0**
USE CASE NAME:	**Complete Employee Attendance**	**USE CASE TYPE**
USE CASE ID:	**16.1**	**Business Requirements:** □

PRIORITY:	High	**System Analysis:**	☑
SOURCE:	IMOS Functional Requirements	**System Design:**	□
PRIMARY BUSINESS ACTOR:	Admin		
PRIMARY THE SYSTEM ACTOR:	None		
OTHER PARTICIPATING ACTORS:	None		
OTHER INTERESTED STAKEHOLDERS:	Owner, Manager		
DESCRIPTION:	This use case describes the process of completing the Employee Attendance on the system. The use case begins when the admin decides on adding new Employee Attendance details. The system verifies the input data type and ensures correct format of Employee Attendance details entered by the Admin. The use case ends when the system displays a success notification that Employee Attendance details were completed/submitted.		
PRE-CONDITION:	The Employee Attendance details do not yet exist on the system. The Admin is logged onto the system.		
TRIGGER:	The Admin would like to Submit new Employee Attendance details on the system.		
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**	
		Step 1: The system invokes Use Case "Search/View Employee Attendance" and displays the screen.	
	Step 2: The Admin selects the option to submit new Employee Attendance.	**Step 3:** The system displays the Submit Employee Attendance form and prompts the Admin to input the details. ⌦ Project ⌦ Task ⌦ Hours	
	Step 4: The Admin enters the Employee Attendance details indicated in **Step 3.**		
	Step 5: The Admin selects the option to Save new Employee Attendance details onto the system.	**Step 6:** The system validates and verifies that the input entered matches the data type and have correct format.[ALT]	
		Step 7: The system prompts the Admin to select confirmation to **Save** Employee Attendance details.	
	Step 8: The admin selects the option to save Employee Attendance details. [ALT].	**Step 9:** The system generates a unique EP_ID, incremented from the last retrieved EP_ID then Submits the details in the Employee Entity.	

		Step 10: The system displays a success notification that Employee Attendance details were Submitted.
ALTERNATE COURSES:	ALT Step 6: The system displays validation errors. ⌦ Invalid data type or incorrect input format.	
	ALT Step 8: The Admin cancels request to Complete Employee Attendance. ⌦ Return to View Employee Attendance screen.	
CONCLUSION:	The system displays a success notification that Employee Attendance details were Completed.	
POST-CONDITION:	The Employee Entity has a new Submitted Employee Attendance record.	
BUSINESS RULES:	Only authorised Users can Submit new Employee Attendance details onto the system.	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None	
ASSUMPTIONS:	None	
OPEN ISSUES:	None	

Delete Employee Attendance

IMOS system		
DATE:	**AUTHOR:** ARMIN WESSELS	**VERSION:1.0**
USE CASE DETAILS:	**Delete Employee Attendance**	**USE CASE TYPE**

USE CASE ID:	**16.2**	**Business Requirements:**	☐
PRIORITY:	High	**System Analysis:**	☑
SOURCE:	IMOS Functional Requirements	**System Design:**	☐
PRIMARY BUSINESS ACTOR:	Admin		
PRIMARY THE SYSTEM ACTOR:	None		
OTHER PARTICIPATING ACTORS:	None		
OTHER INTERESTED STAKEHOLDERS:	Owner, Manager		
DESCRIPTION:	This use case describes the event where Employee Attendance is removed from the system. The use case starts when the Admin wishes to remove Employee Attendance details. The Admin searches for specific Employee Attendance details then requests to delete. The use case concludes when the system displays a success notification that the Employee Attendance has been deleted from the system.		
PRE-CONDITION:	The Admin has been logged onto the system. The Employee exists on the system.		
TRIGGER:	The Admin wishes to remove Employee Attendance details on the system.		
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**	
		Step 1: The system invokes Use Case "Search Employee Attendance"	
	Step 2: The Admin requests to remove Employee Attendance details from the system.	**Step 3:** The system prompts the Admin to confirm to remove Employee Attendance details from the system.	
	Step 4: The Admin selects the option to proceed with removing the Employee Attendance permanently from the system [ALT].	**Step 5:** The system displays a successful snack bar message that the Employee Attendance was deleted, successfully.	
ALTERNATE COURSES:	ALT Step 4: The Admin selects the option to cancel Employee Attendance removal from the system.		
CONCLUSION:	The system displays a snack bar with success removing Employee Attendance details. "Employee Attendance details deleted"		
POST-CONDITION:	Employee Attendance has been deleted and the Employee table updated.		
BUSINESS RULES:	Only authorised Users will be allowed access to Employee Attendance management within the system.		
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None		
ASSUMPTIONS:	None		
OPEN ISSUES:	None		

Search Employee Attendance

IMOS system		
DATE:	**AUTHOR:** ARMIN WESSELS	**VERSION:1.0**
USE CASE NAME:	**Search Employee Attendance**	**USE CASE TYPE**
USE CASE ID:	**16.3**	**Business Requirements:** ☐
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** ☐
PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY SYSTEM ACTOR:	None	

OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	Owner, Manager	
DESCRIPTION:	This use case describes the process when an Admin searches for Employee Attendance details on the system. The use case begins when the Admin requests to search for specific Employee Attendance details. The use case concludes when the system displays the results of Employee Attendance search for the Admin to View.	
PRE-CONDITION:	The Employee Attendance details to be searched for exist on the system. The Admin is logged onto the system.	
TRIGGER:	The Admin wishes to search for specific Employee Attendance details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
	Step 1: The Admin wishes to search for specific Employee Attendance details on the system.	**Step 2:** The system displays in table a list of Employee Attendance details, retrieved from the **Employee Entity** with the following attributes: ⌦ Project ⌦ Task ⌦ Hours
	Step 3: The Admin clicks "Search Table" to input the search keyword or criteria. [ALT].	**Step 4:** The system filters and displays specific Employee Attendance details matching search parameters.
ALTERNATE COURSES:	ALT Step 1: The Admin selects to update Employee Attendance details. ⌦ Redirects to the Update Employee Attendance screen.	
	ALT Step 3: The Admin selects to delete Employee Attendance details. ⌦ Displays delete confirmation prompt.	
	ALT Step 3: The Admin selects to Add New Employee Attendance details. ⌦ Redirects to Add Employee Attendance screen.	
CONCLUSION:	The system displays the results of Employee Attendance search for the Admin to View.	
POST-CONDITION:	Specific Employee Attendance details are displayed for Admin to view.	
BUSINESS RULES:	Only authorised Users will be allowed to search and view Employee Attendance details.	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None	
ASSUMPTIONS:	None	
OPEN ISSUES:	None	

Update Employee Attendance

IMOS system		
DATE:	**AUTHOR:** ARMIN WESSELS	**VERSION:1.0**
USE CASE NAME:	**Update Employee Attendance**	**USE CASE TYPE**
USE CASE ID:	**16.4**	**Business Requirements:** ☐
PRIORITY:	High	**System Analysis:** ☑
SOURCE:	IMOS Functional Requirements	**System Design:** ☐

PRIMARY BUSINESS ACTOR:	Admin	
PRIMARY THE SYSTEM ACTOR:	None	
OTHER PARTICIPATING ACTORS:	None	
OTHER INTERESTED STAKEHOLDERS:	Owner, Manager	
DESCRIPTION:	This use case describes the event where the Admin updates existing Employee Attendance details. The use case starts when the Admin requests to update specific Employee Attendance details. The Admin searches and provides updated input for the specific Employee Attendance they would like to update. The system verifies the data type and ensures correct format of updated details provided. The use case ends when the system displays a success notification that Employee Attendance details were updated.	
PRE-CONDITION:	The Admin is logged onto the system. The Employee Attendance details exist on the system.	
TRIGGER:	The Admin wishes to update Employee Attendance details on the system.	
TYPICAL COURSE	**ACTOR ACTION:**	**SYSTEM RESPONSE:**
		Step 1: The system invokes Use Case "Search/View Employee Attendance" displays the Employee screen.
	Step 2: The Admin selects the option to update Employee Attendance details.	**Step 3:** The system displays the updated Employee Attendance form and prompts the Admin to input updated Attendance details: ⌦ Project ⌦ Task ⌦ Hours
	Step 4: The Admin enters the Employee Attendance details indicated in **Step 3.**	
	Step 5: The Admin selects the update Employee Attendance option to **Save** updated Employee Attendance details onto the system.	**Step 6:** The system validates and verifies that the input entered matches data type and have correct format.[ALT]
		Step 7: The system prompts the Admin to select confirmation to **Save updated** Employee Attendance details.
	Step 8: The admin selects the option to save Employee Attendance details. [ALT].	**Step 9:** The system displays a success notification that Employee Attendance details were updated.
ALTERNATE COURSES:	**ALT Step 6:** The system displays validation errors. ⌦ Invalid data type or incorrect input format.	

	ALT Step 8: The Admin cancels request to update Employee Attendance. ⌦ Return to View Employee screen.
CONCLUSION:	The system displays a success notification that Employee Attendance details were updated.
POST-CONDITION:	The Employee Entity has updated the Attendance record.
BUSINESS RULES:	Only authorised Users can add new Employee Attendance details onto the system.
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:	None
ASSUMPTIONS:	None
OPEN ISSUES:	None