



UNITED INTERNATIONAL UNIVERSITY

Software Testing And Quality Assurance

Course Code: CSE 4495

Section: A

Trimester: Summer 2024

Name	ID	Work Distribution
Al-Momen Reyad	011203011	PHPUnit
Sonjoy Dey	011202074	Jmeter
Md. Kabirul Hossain	011202026	Selenium
Md. Abdur Rahman	011202260	ZAP + Ettercap

Unit Testing using PHPUnit

#TC	Function	Description	Input Data	Expected Output	Actual Output	Status
TC1	testAssignLeaveProperties()	Validate leave assignment	Employee: Asif Iqbal, Leave Type: Medical, Leave Date: 2021-10-01	Date: 2021-10-01, Length: 8 hours, 1 day, Status: 2, Correct Employee, Leave Type	Leave date: 2021-10-01, length: 8 hours, 1 day, status 2, correct employee, leave type	Pass
TC2	testSetAndGetName()	Test setting and getting education name	Name: Bachelor of Science	Bachelor of Science	Bachelor of Science	Pass
TC3	testAttendanceRecordEntityPunchIn()	Verifies punch in functionality	PunchInUtcTime: +6:00, PunchInUserTime: 11:00	State: PUNCH IN	State: PUNCH IN	Pass
TC4	testLocationEntity()	Validating the Location	Country: BD, Location: Name: Dhaka, Phone: 01521765655	Country: BD, Location: Name: Dhaka	Country: BD, Location: Name: Dhaka	Pass
TC5	testAttendanceRecordEntityPunchOut()	Verifies punch out functionality	PunchInUtcTime: +6:00, PunchOutUserTime: 18:00	State: PUNCH OUT	State: PUNCH OUT	Pass
TC6	testSetAndGetCountryName()	Verifies that the country name	Name: "Bangladesh"	Bangladesh	Bangladesh	Pass
TC7	testGetCountryList()	Verifies that a list of countries is retrieved correctly	N/A	Bangladesh, India...	Bangladesh, India...	Pass
TC8	testLoginLogEntity()	Verifies that the LoginLog entity stores correct user data and login time	DateTime: 2022-07-04 10:56:56, userId: 1, userName: 'username', userRole: 'Admin'	userName: 'username', UserRole: 'Admin' Login Time: 2022-07-04 10:56	userName: 'username', UserRole: 'Admin' Login Time: 2022-07-04 10:56	Pass

#TC	Function	Description	Input Data	Expected Output	Actual Output	Status
TC9	testGetTotalActiveEmployeeCount()	The total active employee count	N/A	4	4	Pass
TC10	testGetLatestAttendanceRecordByEmpNumber()	Fetching the latest attendance record for an employee	Employee number: 2	State: PUNCHED IN - PunchIn UTC: '2024-09-06 09:15:00'	State: PUNCHED IN - PunchIn UTC: '2024-09-06 09:15:00'	Pass
TC11	testEmailConfigurationEntity()	Validating the Email Configuration	MailType: smtp, SentAs: test@orangehrm.com, Smtphost: smtp.gmail.com, Smtport: 587	MailType: smtp	MailType: smtp	Pass
TC12	emailSubscriberTest()	Validating the Email Subscriber and Notification	mailSubscriber: Name: Subs, Email: Subs@subs.com EmailNotification: Name: Leave Assignments, Enabled: true	EmailNotification ID:1	EmailNotification ID:1	Pass
TC13	testNationalityEntity()	Validating the Nationality entity	Nationality: Name: Afghan	Nationality: Afghan and Id: 1	Nationality: Afghan and Id: 1	Pass
TC14	testOrganizationEntity()	Validating the Organization entity	Organization Id: 1, Name: OHRM, TaxId: 12345, RegistrationNumber: 45678, Phone: +880..., Fax: +880..., Email: admin@orangehrm.com, Country: BD, City: Dhaka	Organization Id: 1, Name: OHRM	Organization Id: 1, Name: OHRM	Pass
TC15	testWorkShiftEntity()	Validating the WorkShift entity	StartTime: 08:00:00, EndTime: 17:00:00	HoursPerDay: 8.00	HoursPerDay: 8.00	Pass
TC16	testGetSkillById()	Test retrieving a skill by ID	Skill ID: 1	Skill name: Driving	Skill name: Driving	Pass

#TC	Function	Description	Input Data	Expected Output	Actual Output	Status
TC17	testDeleteSkill() ()	Deleting multiple skills	Skill IDs to delete: [3, 2]	Number of deleted skills: 2	Number of deleted skills: 2	Pass
TC18	testSaveSkill()	Test saving a new skill	Skill name: MS Word	Saved	Saved	Pass
TC19	testEditSkill()	Test editing an existing skill	Skill ID: 1, New name: MS Excel	Updated	Updated	Pass
TC20	testGetSkillsCount()	Counting total skills	N/A	Skills count: 3	Skills count: 3	Pass
TC21	testGetPayGradeList()	Retrieving the list of pay grades	N/A	Number of pay grades: 3	Number of pay grades: 3	Pass
TC22	testGetPayGradeById()	Retrieving a pay grade by ID	Pay Grade ID: 1	Pay Grade Found	Pay Grade Found	Pass
TC23	testGetPayPeriods()	Retrieving list of pay periods	N/A	Number of pay periods: 6	Number of pay periods: 6	Pass
TC24	testGetCurrencies()	Retrieving the list of currencies	N/A	Number of currencies: 2	Number of currencies: 2	Pass
TC25	testDeletePayGrades()	Deleting multiple pay grades	Pay Grade IDs: [1, 2]	Deleted pay grades: 2	Deleted pay grades: 2	Pass
TC26	testGetJobTitleList()	Retrieving the list of job titles	N/A	Number of job titles: 3	Number of job titles: 3	Pass
TC27	testGetJobTitlesForEmployee() ()	Retrieving job titles for a specific employee	Employee ID: 1	Employee ID: 1	Job title names: Quality Assurance	Pass
TC28	testDeleteJobTitle() itle()	Test deleting job titles	Job Title IDs: [3, 2]	Deleted job titles: 2	Deleted job titles: 2	Pass
TC29	testGetJobTitleById()	Retrieving a job title by ID	Job Title ID: 1	Job title name: Software Architect	Job title name: Software Architect	Pass
TC30	testGetJobTitlesCount()	Test counting job titles	Active Only: true	Count of active job titles: 3	Count of active job titles: 3	Pass

Case Study: **testAssignLeaveProperties (TC1)**

Objective: The objective of this test case is to validate that the properties of the Leave entity are correctly assigned and associated with the relevant entities, including the employee's details (such as name and ID), the type of leave, the duration of leave, and the date applied.

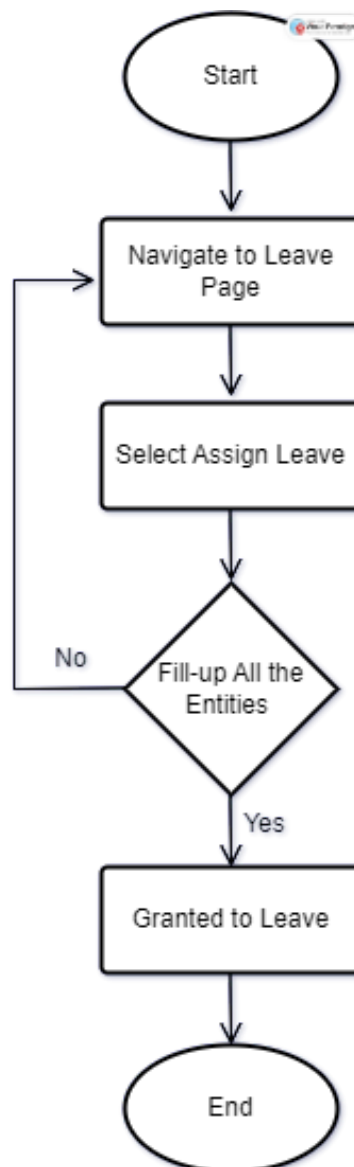
Preconditions:

1. A user must be logged in to the OrangeHRM system.
2. A leave type (Medical, Parental) must be configured in the system.

Test Process:

1. Setup: Initializes employee, leave type, and leave request entities.
2. Mocking Leave: A mock Leave entity is used to avoid external dependencies.
3. Assertions: The test checks that leave properties like date, duration (hours/days), status, and associated entities (employee, leave type, leave request) are correctly assigned.

Flowchart:



Case Study: **testSetAndGetCountryName (TC6)**

Objective: The objective of the testSetAndGetName method in the CountryTest class is to verify that the setName and getName methods in the Country entity correctly set and retrieve the name of a country.

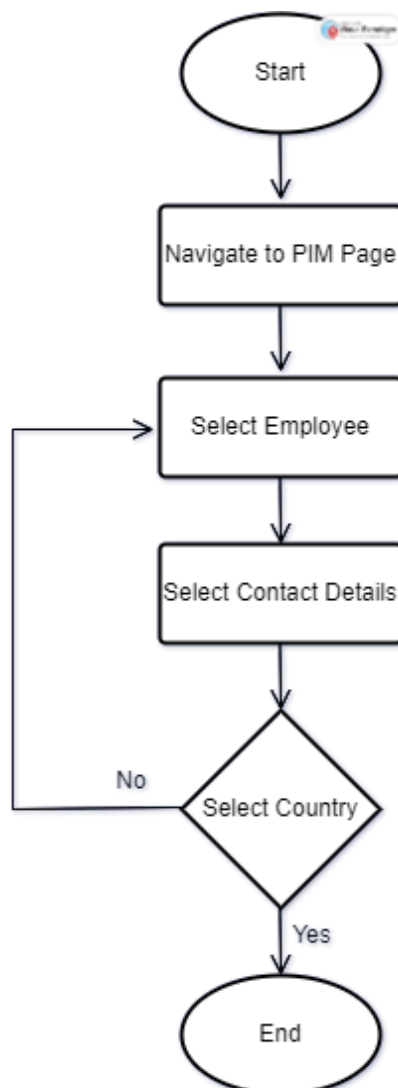
Preconditions:

1. A user must be logged in to the OrangeHRM system.

Test Process:

1. Setup: A new instance of the Country entity is initialized in the setUp method.
2. Set Name: The setName method is used to assign the name "Bangladesh" to the Country object.
3. Get Name: The getName method retrieves the country name previously set.
4. Assertion: The test asserts that the retrieved name matches the expected value "Bangladesh" using assertEquals.

Flowchart:



Case Study: **testLoginLogEntity (TC8)**

Objective: The purpose of this test is to verify that the LoginLog entity in the OrangeHRM system correctly records user login information.

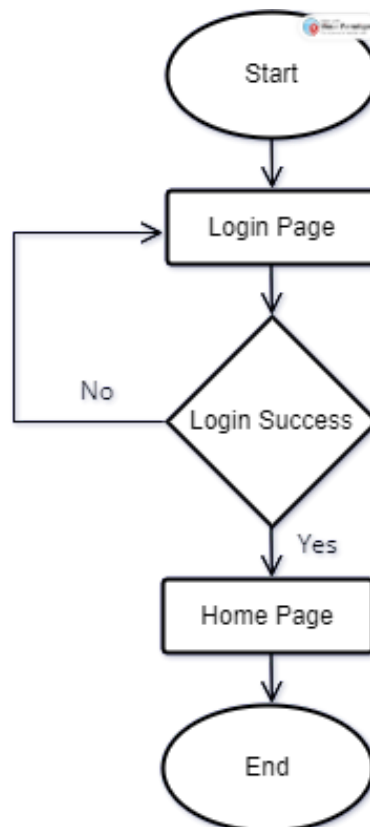
Preconditions:

1. The LoginLog table must exist in the database.

Test Process:

1. Setup: Clears the LoginLog table for a clean test environment.
2. Entity Creation: Creates a LoginLog entity and sets properties like userId, userName, userRoleName, and userRolePredefined.
3. Persistence: Saves the entity to the database.
4. Assertions: Validates that the stored properties match the expected values, including the login time.

Flowchart:



Case Study: **testEmailConfigurationEntity (TC11)**

Objective: The objective is to validate the EmailConfiguration entity in OrangeHRM by ensuring the accurate assignment and persistence of email configuration properties, such as mail type, SMTP host, and security settings.

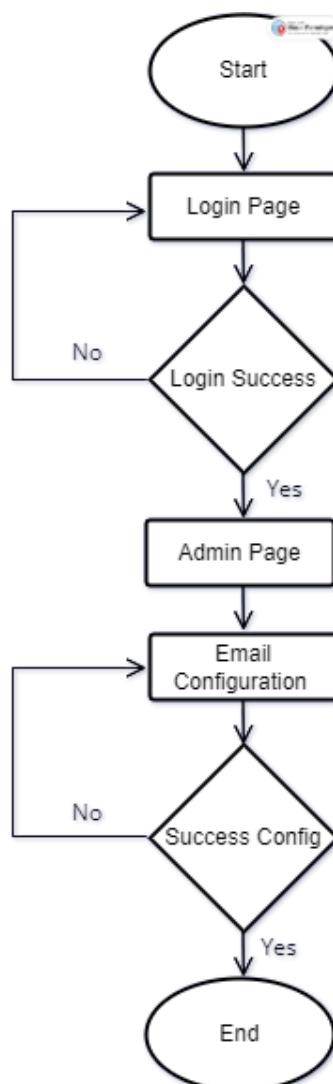
Preconditions:

1. The EmailConfiguration table must be truncated to ensure a clean state.
2. Proper access to the database to persist and retrieve the entity.
3. A user must be logged in to the OrangeHRM system.

Test Process:

1. Setup: Truncate the EmailConfiguration table for a clean environment.
2. Entity Creation: Create and assign email configuration properties (e.g., mail type, SMTP host).
3. Persistence: Save the entity to the database.
4. Validation: Retrieve the saved entity and verify its properties match the expected values.

Flowchart:



Case Study: **testWorkShiftEntity (TC15)**

Objective: Validate that the WorkShift entity's properties, such as name, hours per day, and shift times, are correctly assigned and persisted in the database.

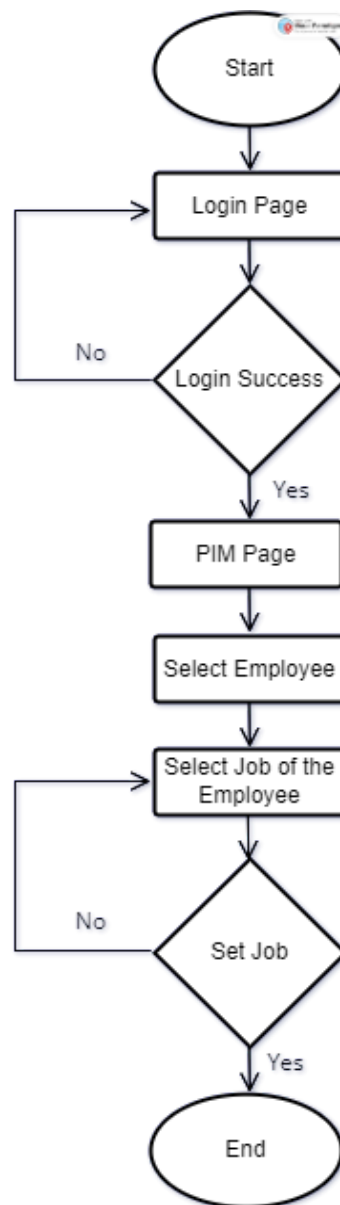
Preconditions:

1. A user must be logged in to the OrangeHRM system.
2. The WorkShift table is truncated before the test to ensure a clean environment.

Test Process:

1. Create: Initialize WorkShift with name, hours per day, start, and end time.
2. Persist: Save the WorkShift entity to the database.
3. Validate: Retrieve and verify that all properties (name, hours, times) are correctly stored.

Flowchart:





Performance testing

Features Selected for Testing:

1. Login
2. Dashboard
3. Admin Control panel
4. Add employee
5. Directory
6. Notification
7. Recruitment
8. Claim
9. Profile setting
10. Employee claim
11. View Candidates
12. Time Module
13. Buzz

Load Testing

Load testing examines how the system behaves during normal and high loads and determines whether a system, piece of software, or computing device can handle high loads given the high demand from end-users.

Test Scenario-1: Performing login admin and viewing system user

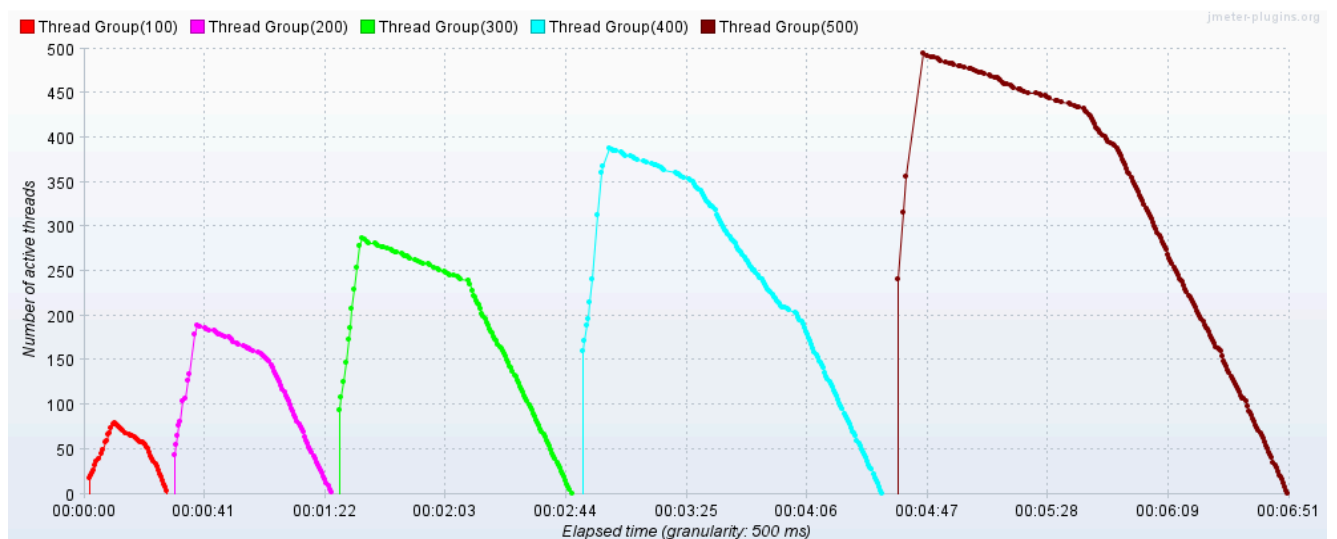
Thread Group: 100-500

Ramp Up Time:10s

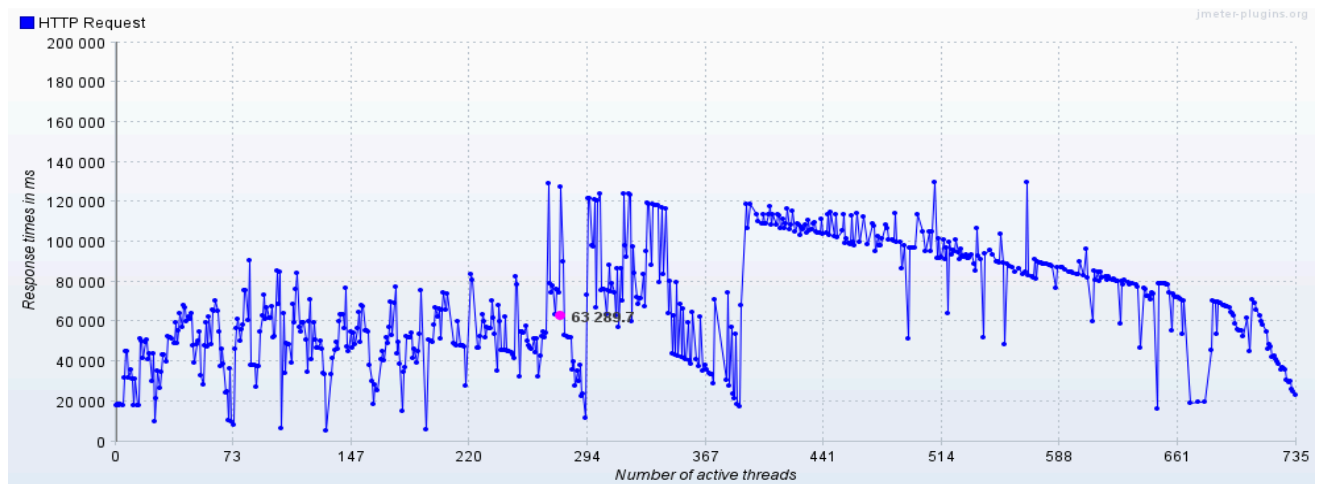
Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
100	13542	0.00	3.5/s	17.15	1.04
200	33773	0.00	3.6/s	17.45	1.06
300	51229	0.00	3.7/s	17.56	1.08
400	64939	0.00	3.8/s	18.31	1.11
500	90962	0.00	3.6//s	17.3	1.06

Active Thread Over Time:



Response Time Vs Active Thread:



Observation :

This test simulates login admin and viewing system users with thread groups ranging from 100 to 500 and a ramp-up time of 10 seconds. The results show that the system handles up to 500 users without any errors. Throughput remained relatively stable at around 3.5 to 3.8 requests per second, and the data transfer rate (both received and sent) also remained constant.

Test Scenario-2: Performing View Pay Grades

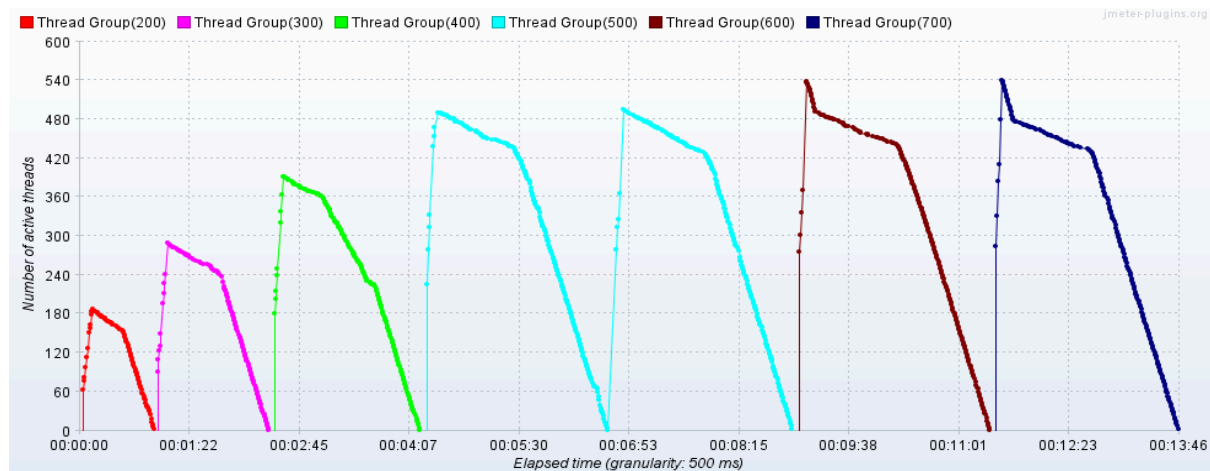
Thread Group: 200-700

Ramp Up Time:10s

Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
200	33628	0.00	3.5/s	17.22	1.04
300	54700	0.00	3.5/s	16.89	1.02
400	70938	0.00	3.5/s	17.09	1.03
500	92881	0.00	3.6/s	17.18	1.03
600	85006	16.50	4.0/s	18.52	1.03
700	68254	30.0	4.9/s	21.10	1.07

Active Thread Over Time:



Observation :

The system performed well up to 500 users but started to show a rising error rate from 600 users onward, reaching 30% at 700 users. Although throughput increased, the error rate suggests that the system struggled with higher user loads beyond 500 users, indicating potential performance bottlenecks.

Stress Testing

Stress testing is a software testing activity that determines the robustness of software by testing beyond the limits of normal operation. Stress testing is particularly important for "mission critical" software, but is used for all types of software.

Test Scenario-1: Performing login admin and viewing system user

Thread Group: 500-1000

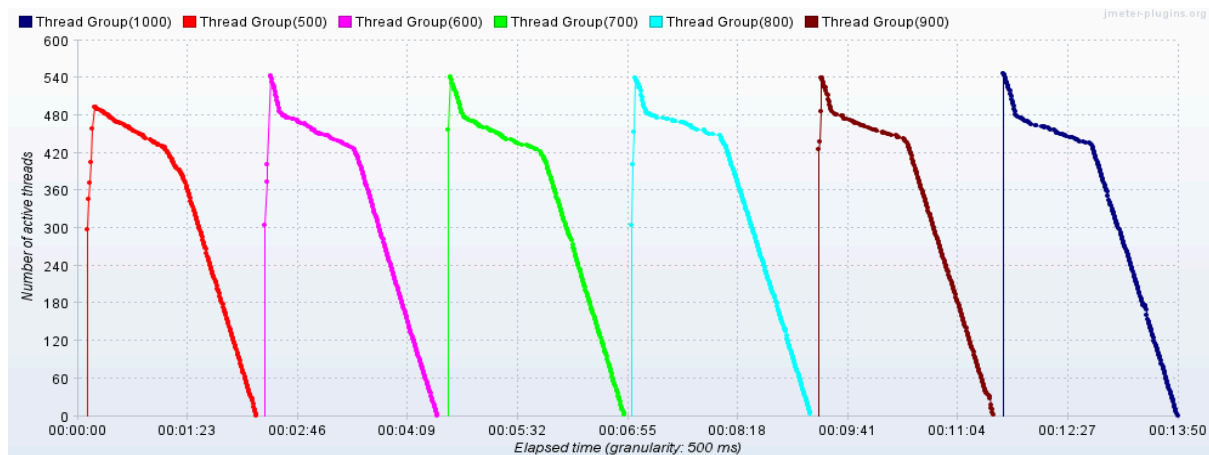
Ramp Up Time: 10s

Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
500	88249	0.00	3.7/s	17.97	1.04
600	75923	17.6	4.4/s	20.11	1.12
700	68112	30.4	5.0/s	21.18	1.07
800	60296	38.0	5.7/s	23.16	1.09

900	51924	45.5	6.5/s	25.41	1.10
1000	47134	51.6	7.2//s	27.13	1.10

Active Thread Over Time:



Observation :

The system began to fail after 500 users, with error rates increasing to 51.6% at 1000 users. The performance declined as throughput increased, indicating that the system's resources (CPU/memory) were insufficient to handle this stress level. The test led to a device crash due to resource exhaustion.

Test Scenario-2: Performing Buzz

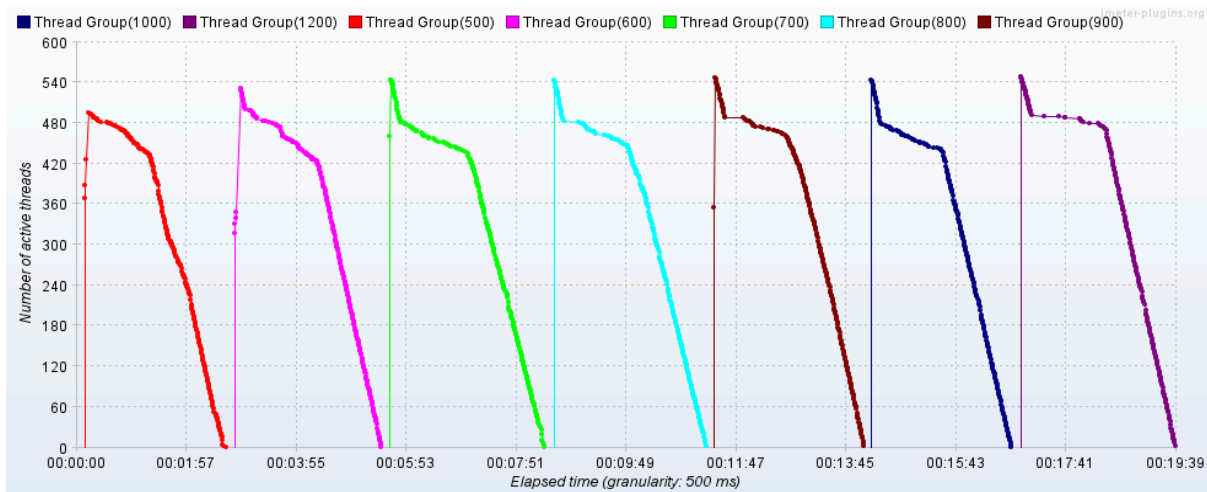
Thread Group: 500-1200

Ramp Up Time:10s

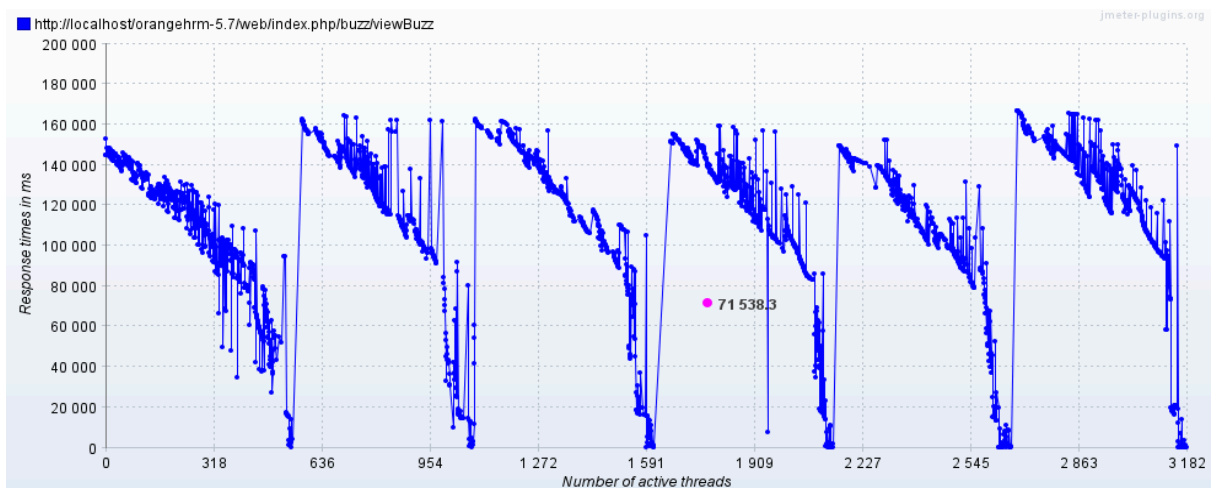
Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
500	103562	0.00	3.1/s	15.02	0.89
600	95610	15.17	3.6/s	16.56	0.96
700	84951	29.71	4.0/s	17.19	0.85
800	75782	39.0	4.6/s	18.69	0.85
900	66939	43.44	5.3/s	20.82	0.89
1000	53789	51.90	6.3/s	23.81	0.93
1200	53516	56.17	6.8/s	24.56	0.88

Active Thread Over Time:



Response Times vs Threads:



Observation :

Errors started to appear at 600 users, reaching a significant 56.17% at 1200 users. Although throughput improved as more users were added, the rising error rates indicated that the system was overwhelmed by the load, suggesting the need for system optimization under stress.

Spike Testing

Spike testing is a type of performance testing in which an application receives a sudden and extreme increase or decrease in load. The goal of spike testing is to determine the behavior of a software application when it receives extreme variations in traffic.

Test Scenario-1: Performing recruitment and viewing candidates

Thread Group: 500,200,150,100,50

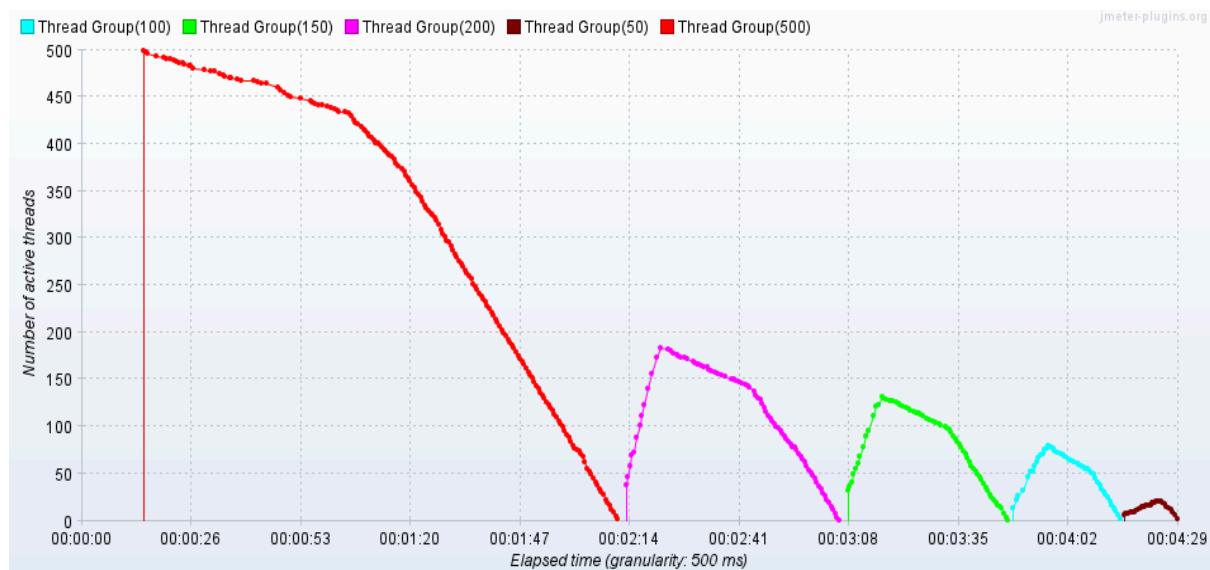
Ramp Up Time:10s

Time: 2min After the First load, then maintained 4s for each thread group

Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
50	3506	0.00	3.6/s	17.36	1.07
100	13151	0.00	3.6/s	17.55	1.08
150	22250	0.00	3.6/s	17.55	1.08
200	31167	0.00	3.7/s	17.84	1.10
500	88030	18.3	1.6//s	7.25	0.41

Active Thread Over Time:



Observation :

The system is tested with sudden increases in users (50, 100, 150, 200, 500 users) over short periods. The results show that as the user load spikes, the system maintains stable performance up to 200 users. However, at **500 users**, the **error rate jumps to 18.3%**, throughput drops to **1.6 requests/sec**, and the data transfer rate (KB/s) declines, indicating the system struggles to handle sudden high spikes in load.

Test Scenario-2: Performing recruitment and viewing Candidates

Thread Group: 500,200,150,100,50

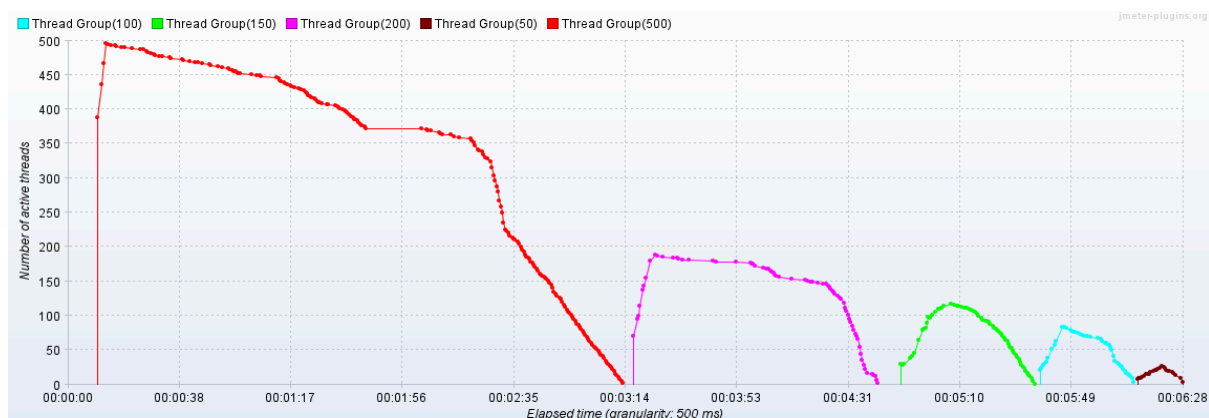
Ramp Up Time:10s

Time: 2.30 min After the First load, then maintained 3s for each thread group

Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
50	5287	0.00	3.0/s	14.41	0.88
100	17950	0.00	2.9/s	14.00	0.85
150	24612	0.00.	2.8/s	13.41	0.82
200	63831	0.00	2.3/s	10.93	0.67
500	132183	2.4	2.6//s	12.54	0.76

Active Thread Over Time:



Observation :

As user load spiked, the system maintained steady performance up to 200 users, after which errors began to appear. At 500 users, the system's error rate increased significantly, highlighting its inability to handle large sudden spikes efficiently.

Concurrency Testing

Concurrency testing is software testing that aims to identify and resolve issues that may arise when multiple users or processes access a system simultaneously. It helps ensure the system can handle concurrent requests without crashing or causing data corruption.

Test Scenario-1: Performing Claim and employee claims

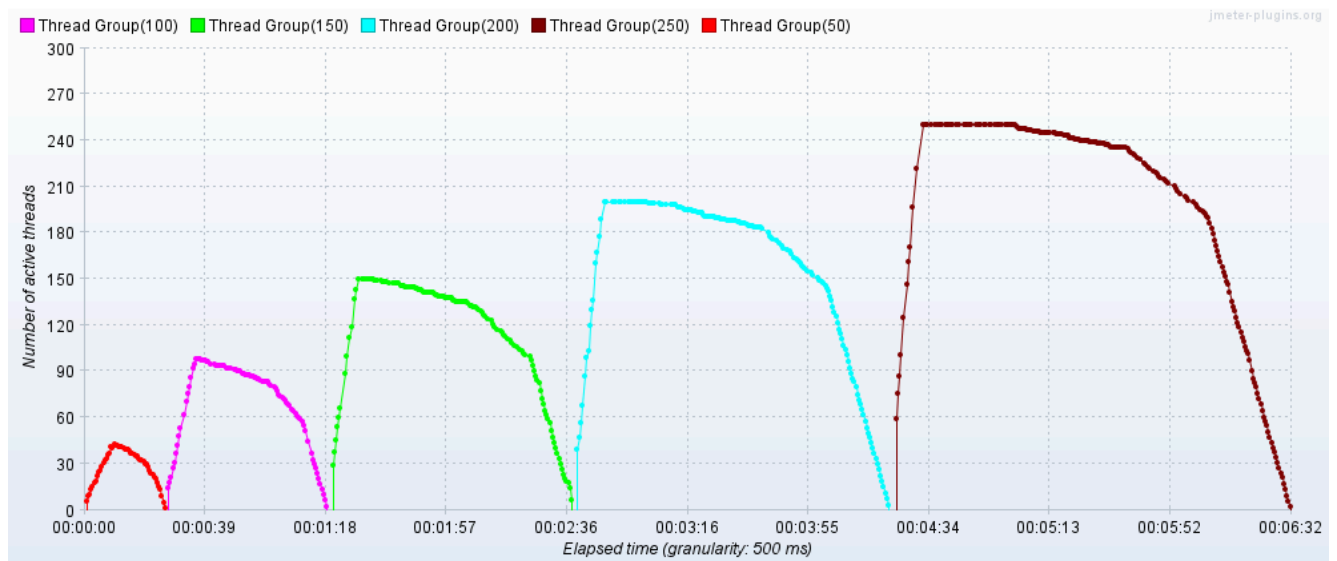
Thread Group: 50,100,150,200,250

Ramp Up Time:10s

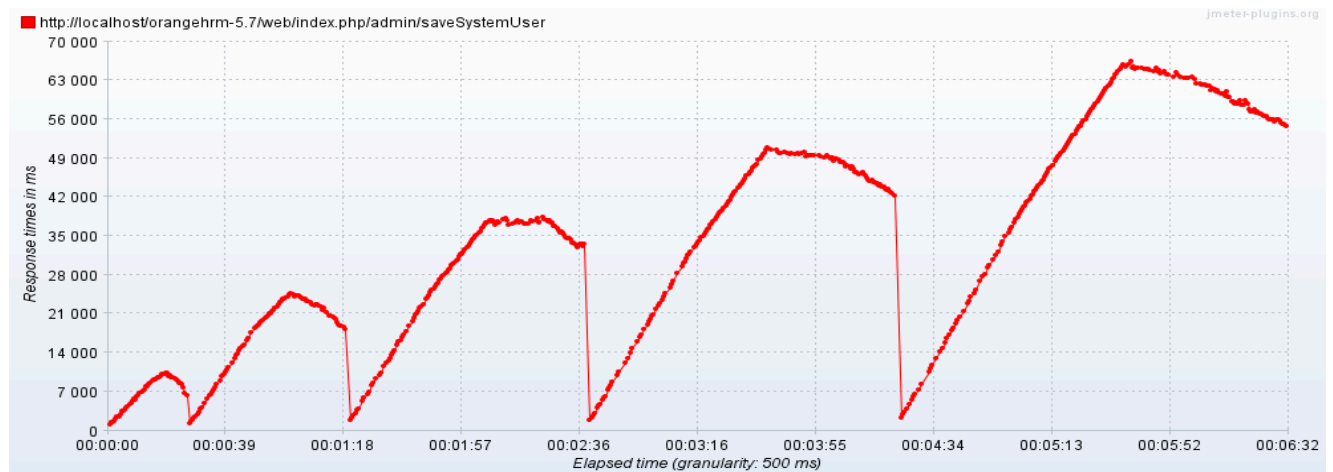
Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
50	7293	0.00	3.8/s	18.33	1.11
100	18325	0.00	3.8/s	18.44	1.11
150	29862	0.00.	3.8/s	18.29	1.10
200	39969	0.00	3.9/s	18.86	1.14
250	51904	0.00	3.8//s	18.61	1.12

Active Thread Over Time:



Response Time Over Time:



Observation :

The system successfully handled concurrent users with no errors up to 250 users. Throughput remained stable, and there were no signs of performance bottlenecks. The test showed that the system could efficiently handle concurrent requests without failure.

Test Scenario-2: Performing Time Module

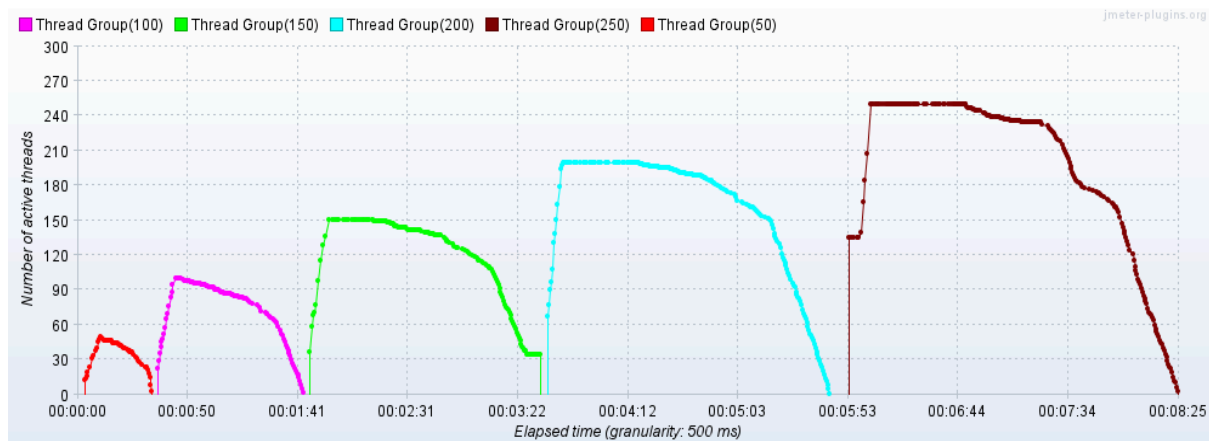
Thread Group: 50,100,150,200,250

Ramp Up Time:10s

Aggregate Report:

Thread	Avg	Error %	Throughput	Receive KB/s	Sent KB/s
50	10907	0.00	3.0/s	14.64	0.88
100	24834	0.00	2.9/s	14.01	0.84
150	41827	0.00	2.8/s	13.44	0.81
200	52718	0.00	3.0/s	14.61	0.88
250	60489	0.00	3.1/s	15.20	0.91

Active Thread Over Time:



Observation :

The system handled concurrent user access with minimal issues. Performance remained stable with no errors up to 250 users, though there was a slight decline in throughput. This indicates the system's good concurrency handling for time-based operations.

Functional Testing Tools - Selenium

Selenium testing involves the use of the Selenium framework to automate web applications in order to test them. It is an open-source tool, and its usage is quite common these days, allowing testers and developers to write tests in a number of different programming languages. For this project I write a test case using Java as a programming language.

Scenario Formulation

1. Login → Navigate to PIM → Add Employee → Fill require details → Click save button → Logout.
2. Login → Navigate to PIM → Select Employee list → Search by exist name or ID → click search button → Verify → Logout.
3. Login → Navigate to PIM → Select Employee List → Search employee by name → Click search button → delete employee → Logout
4. Login → Navigate to Leave → apply for leave → logout.
5. Login → navigate to Time → Define Time → Click to save → logout.
6. Login → navigate to Recruitment → create new job vacancy → logout.
7. Login → navigate to Recruitment → Vacancy → Edit Job Vacancy Details → Logout

8. Login → Navigate to PIM → Select Employee List → Sort Employees Ascending order using by Name → Logout
9. Login → navigate to performance → update performance evaluation form → logout.
10. Login → navigate leave → view leave transactions history → logout.
11. Login → navigate to employee → view employee details → logout.
12. Login →navigate to time →Attendance Summary→ view attendance summary for specific time → logout.
13. Login → Recruitment → delete job vacancy→ logout.
14. Login → navigate to leave → Terminate all leave requests →logout.
15. Login → navigate to PIM → employee list → delete an employee → logout.
16. Login → Navigate to time → go to attendance module → view attendance calendar → logout.
17. Login → navigate to performance → generate performance evaluation report → logout.
18. Login → navigate to employee → edit employees personal information → logout.
19. Login → go to the training module → Enrol a group of employees in a training program → logout.
20. Login → navigate to time → Create an attendance report → logout.
21. Login → navigate to recruitment → candidate interview schedule→logout
22. Login → navigate to recruitment → search job vacancies using title → logout.
23. Login → navigate to recruitment → add job vacancy→ logout.
24. Login → navigate to time → view overtime hours → logout.
25. Login → navigate to training → add employee in training program → remove employee from training program → logout.
26. Login → navigate to leave → view transactions history → logout.
27. Login → navigate to the admin module → update user role permissions → deactivate a user account → logout.
28. Login → navigate to the time module → record attendance for an employee → approve attendance → logout.
29. Login →navigate to leave → approves leave request → logout.
30. Login → navigate to leave → view upcoming holidays → logout.

Result Summary:

#TC	Scenario	TC Steps	Expected Outcome	Observed Result	Verdict
TC01	To check the Add Employee feature in the PIM module works correctly.	Login → Navigate to PIM → Add Employee → Fill require details → Click save button → Logout.	Employee data is added to the employee list	Successfully add Employees data and store in the system.	Pass
TC02	To check employee search function by name or ID operates smoothly and retrieves the correct data.	Login → Navigate to PIM → Select Employee list → Search by exist name or ID → click search button → Verify → Logout.	Validate Data Retrieval.	Record Found	Pass
TC03	To ensure the functionality of the employee deletion process	Login → Navigate to PIM → Select Employee List → Search employee by name → Click search button → delete employee → Logout	Employee should be deleted from the employee list	Employee data deleted successfully.	Pass
TC04	To verify that users can successfully apply for leave through the Leave module.	Login → Navigate to Leave → apply for leave → logout.	Test apply for leave	Leave application is successfully submitted	Pass
TC05	To check if the process of defining and saving time	Login → navigate to Time → Define Time → Click to save	Validate the clock-in and clock-out feature	Clock in and Clock out is successful	Pass

		→ logout.			
TC06	To verify that a new job vacancy can be successfully created through the Recruitment section.	Login → navigate to Recruitment → create new job vacancy → logout.	Create job vacancy	New job vacancy is successfully created	Pass
TC07	To check if users can smoothly navigate to the Vacancy section and update job vacancy details	Login → navigate to Recruitment → Vacancy → Edit Job Vacancy Details → Logout	Edit details job vacancy	Job vacancy details are edited successfully	Pass
TC08	To validate that the system allows accurate sorting of employees in ascending order based on their names.	Login → Navigate to PIM → Select Employee List → Sort Employees Ascending order using by Name → Logout	Sort Employees Name ascending order.	Sort Employees name successfully	Pass
TC09	To ensure the functionality of updating the performance evaluation form works as expected.	Login → navigate to performance → update performance evaluation form → logout.	Assess the updating procedure for the performance evaluation form	The performance evaluation form update was applied without any issues	Pass
TC10	To confirm that admin can create job vacancies in the Recruitment	Login → navigate leave → view leave transactions history → logout.	Validate the leave transaction history	The leave transactions history is	Pass

	module without issues.		viewing feature	displayed correctly	
TC11	To verify that users can view employee details after navigating to the employee section.	Login → navigate to employee → view employee details → logout.	View employee details	Employee details are viewed successfully	Pass
TC12	To check that users can view the attendance summary for a specific time.	Login →navigate to time →Attendance Summary→ view attendance summary for specific time → logout.	View attendance summary	Attendance summary is viewed successfully	Pass
TC13	To ensure that users can delete a job vacancy through the Recruitment section.	Login → Recruitment → delete job vacancy→ logout.	Delete job vacancy	Job vacancy is successfully deleted	Pass
TC14	To verify that users can terminate all leave requests through the Leave section.	Login → navigate to leave → Terminate all leave requests →logout.	Terminate leave requests	All leave requests are successfully terminated	Pass
TC15	To check that users can delete an employee from the employee list	Login → navigate to PIM → employee list → delete an	Delete employee	Employee is successfully deleted	Pass

	through the PIM section.	employee → logout.			
TC16	To ensure that users can view the attendance calendar through the Attendance module in the Time section.	Login → Navigate to time → go to attendance module → view attendance calendar → logout.	View attendance calendar	Attendance calendar is successfully viewed	Pass
TC17	To check that users can generate a performance evaluation report through the Performance section.	Login → navigate to performance → generate performance evaluation report → logout.	Generate performance evaluation report	Performance evaluation report is successfully generated	Pass
TC18	To verify that users can edit an employee's personal information through the Employee section.	Login → navigate to employee → edit employees personal information → logout.	Edit employee's personal information	Employee's personal information is successfully edited	Pass
TC19	To ensure that users can enroll a group of employees in a training program through the Training module.	Login → go to training module → Enroll a group of employees in a training program → logout.	Enroll employees in training program	Group of employees is successfully enrolled in the training program	Pass

TC20	To verify that users can create an attendance report through the Time section.	Login → navigate to time → Create an attendance report → logout.	Create attendance report	Attendance report is successfully created	Pass
TC21	To verify that users can access the candidate interview schedule through the Recruitment section.	Login → navigate to recruitment → candidate interview schedule → logout	Access candidate interview schedule	Candidate interview schedule is successfully accessed	Pass
TC22	To ensure that users can search for job vacancies using the title through the Recruitment section.	Login → navigate to recruitment → search job vacancies using title → logout.	Search job vacancies	Job vacancies are successfully searched by title	Pass
TC23	To ensure that users can add a job vacancy through the Recruitment section.	Login → navigate to recruitment → add job vacancy → logout.	Add job vacancy	Job vacancy is successfully added	Pass
TC24	To verify that users can view overtime hours through the Time section.	Login → navigate to time → view overtime hours → logout.	View overtime hours	Overtime hours are successfully viewed	Pass
TC25	To check that users can add and remove an employee from a	Login → navigate to training → add employee in training program	Add and remove employee from	Employee is successfully added and then removed	Pass

	training program through the Training section.	→ remove employee from training program → logout.	training program	from the training program	
TC26	To ensure that users can view transaction history through the Leave section.	Login → navigate to leave → view transactions history → logout.	View transaction history	Transaction history is successfully viewed	Pass
TC27	To ensure that users can update user role permissions and deactivate a user account through the Admin module.	Login → navigate to the admin module → update user role permissions → deactivate a user account → logout.	Update user role permissions and deactivate user account	User role permissions are updated, and the user account is successfully deactivated	Pass
TC28	To verify that users can record attendance for an employee and approve it through the Time module.	Login → navigate to the time module → record attendance for an employee → approve attendance → logout.	Record and approve attendance	Attendance for the employee is successfully recorded and approved	Pass
TC29	To check that users can approve leave requests through the Leave section.	Login → navigate to leave → approves leave request → logout.	Approve leave request	Leave request is successfully approved	Pass

TC30	To check that users can view upcoming holidays through the Leave section.	Login → navigate to leave → view upcoming holidays → logout.	View upcoming holidays	Upcoming holidays are successfully viewed	Pass
------	---	--	------------------------	---	------

Case Study:

Case Study 01: Employee Deletion Process in PIM Module

Introduction: This case study assesses the deletion of employee records in the PIM module, ensuring HR can search, delete, and confirm the removal smoothly and without errors.

Challenges:

User Errors: Mistakenly deleting an employee record is a major concern. To prevent this, the system should provide clear confirmation prompts and double-checks before finalizing the deletion.

Authorization: Only authorized personnel should delete employee records. The system must enforce access controls to prevent unauthorized users from initiating the deletion process.

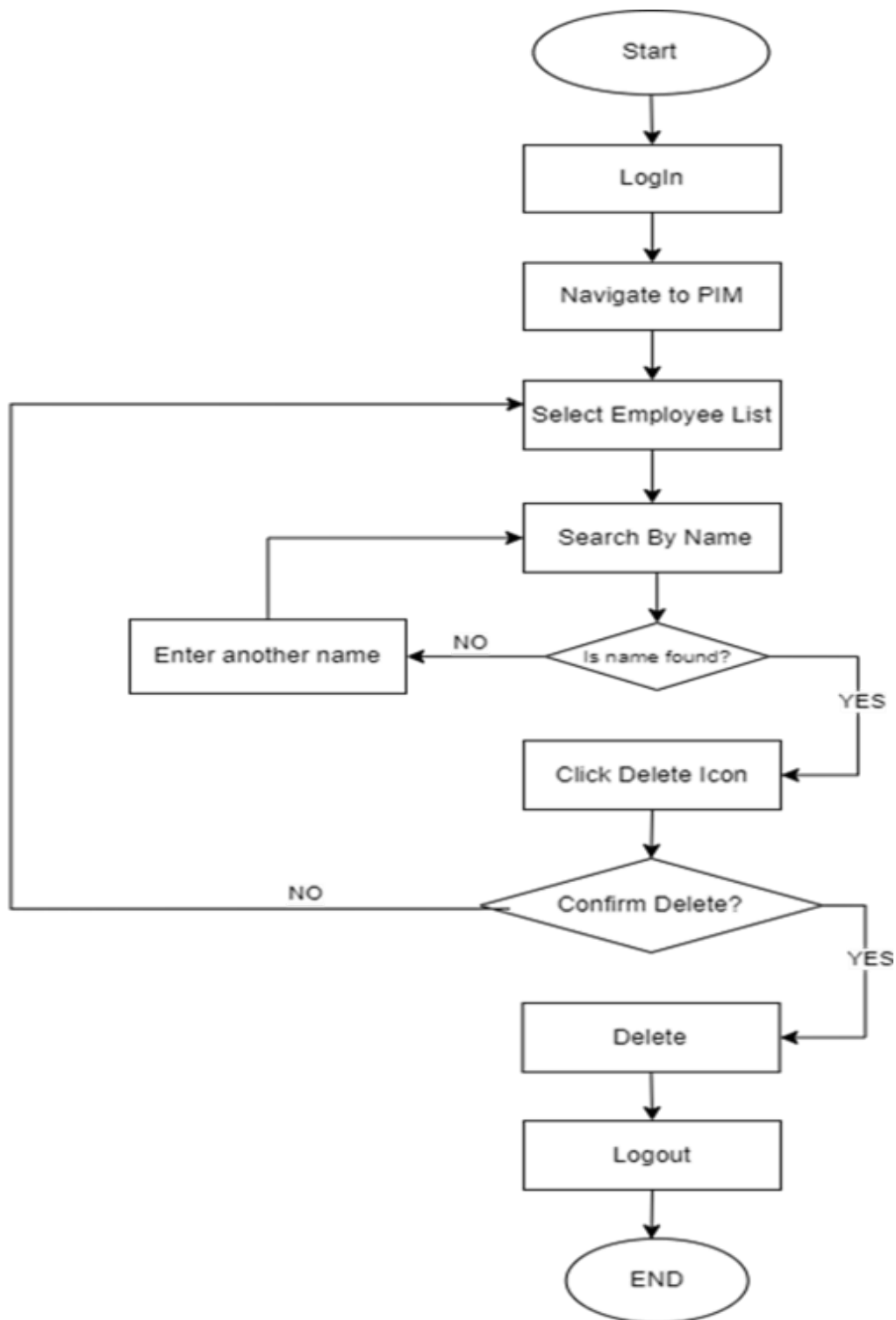
System Feedback: Users need confirmation of successful deletion; without it, they may think the action failed, causing confusion or repeated attempts.

Risks and Mitigations:

Accidental Deletions: To mitigate this risk, the system must incorporate robust confirmation mechanisms

Unauthorized Access: Strict role-based access control policies. ensuring that only authorized users can delete employee records.

FlowChart:



Case Study 02: Admin Module User Role Management

Introduction: In modern organizations, effective user management is crucial for maintaining security and operational efficiency. The Admin module serves as a pivotal component of user access control, enabling administrators to update

user role permissions and deactivate user accounts. This case study examines the process of managing user roles through the admin module, focusing on the steps involved, expected outcomes, challenges faced, and potential risks associated with these actions.

Objective: The objective of this case study is to assess the effectiveness of the admin module in updating user role permissions and deactivating user accounts while ensuring compliance with organizational policies and maintaining operational integrity.

Challenges:

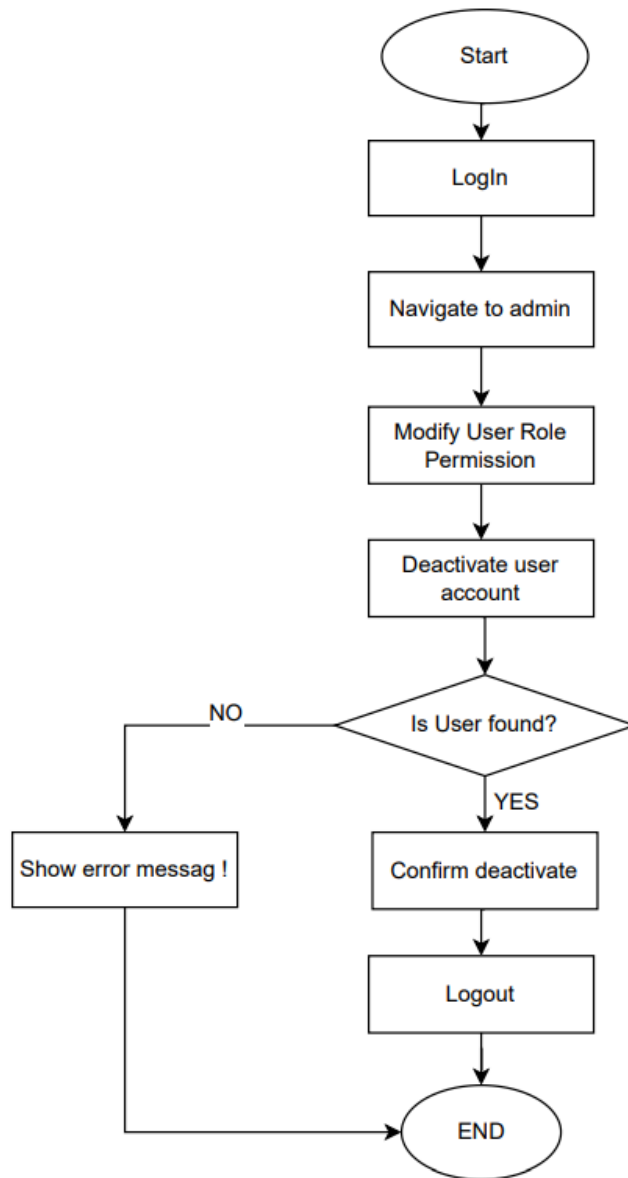
Complex Role Structures: Organizations often have complex role hierarchies, making it challenging to determine appropriate permissions for different users. Misconfigurations can lead to either excessive access or unnecessary restrictions.

User Resistance: Employees may resist changes to their roles, especially if they perceive that their access has been unfairly restricted. This resistance can lead to dissatisfaction and reduced productivity.

Risks:

Security Vulnerabilities: If user roles are not managed properly, unauthorized access may occur, potentially exposing sensitive information.

FlowChart:



Case Study 03: Employee Enrollment in Training Program

Introduction: In a rapidly evolving business environment, continuous training and development of employees are essential for organizational growth. The training module in an HR system enables organizations to streamline the process of enrolling employees in training programs. This case study explores the process of enrolling a group of employees into a training program using the training module, focusing on the steps involved, potential challenges, and associated risks.

Objective: The objective of this case study is to evaluate the effectiveness of using the training module to enroll multiple employees in a program and analyze the challenges and risks associated with this process.

Challenges :

Employee Availability: One of the common challenges is ensuring the availability of all selected employees for the training. Scheduling conflicts may arise if employees are already committed to other tasks, meetings, or training sessions.

Group Management: Enrolling a large group of employees can be complex, especially when dealing with multiple departments or job roles. Ensuring that all participants meet the program's requirements or prerequisites may take time.

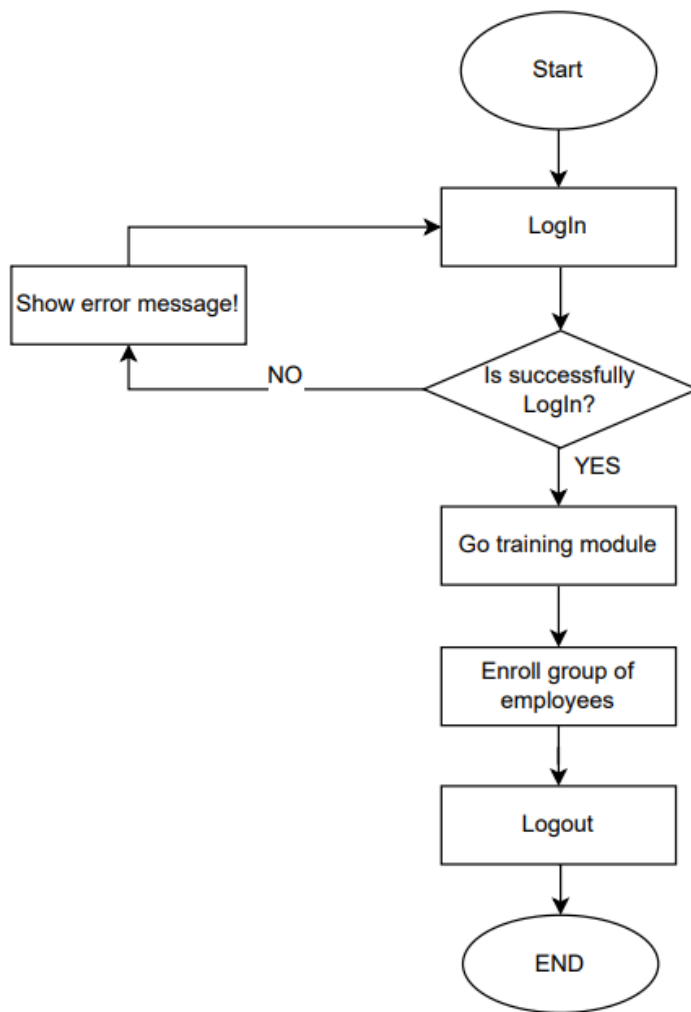
Tracking Progress: After enrollment, tracking the progress of a large group can be cumbersome. The system should have adequate reporting features to monitor who attended the training and completed the program.

Risks:

Overloading Training Resources: If too many employees are enrolled at once, the training program might become oversubscribed, leading to insufficient resources (e.g., trainers, space, or materials), which could hinder training quality.

Data Security: There is always a risk of unauthorized access or exposure of sensitive employee data during enrollment. Administrators must ensure data privacy protocols are followed throughout the process.

FlowChart:



FINAL REPORT

Software Testing and Quality Assurance

by
Md. Abdur Rahman
ID: 011202260
Section: A

1. Introduction

This report documents the security testing conducted on the orangehrm software with a focus on identifying vulnerabilities through various test cases. The goal of this testing is to identify different weaknesses of the software against various security attacks.

2. Tools Overview

Initially, the *OWASP ZAP* (Zed Attack Proxy) tool was selected to conduct the security testing. ZAP is a widely recognized open-source application that is used as a security scanner. It is a highly extensible tool that supports a range of plugins and scripts. Primarily it includes features like:

- *Automated scanning*
- *Interception proxy*
- *Web crawler*
- *Fuzzer*
- *Token extractor*
- *Query decoder*
- *Script engines, and many more.*

However, it still lacks features to generate attacks like *MITM*, Address spoofing, *ARP poisoning*, etc. Thus for testing, another tool named *Ettercap* was selected which includes attacks such as:

- *ARP poisoning*
- *NDP poisoning*
- *DHCP spoofing*
- *Port stealing*
- *SSL intercepting, etc.*

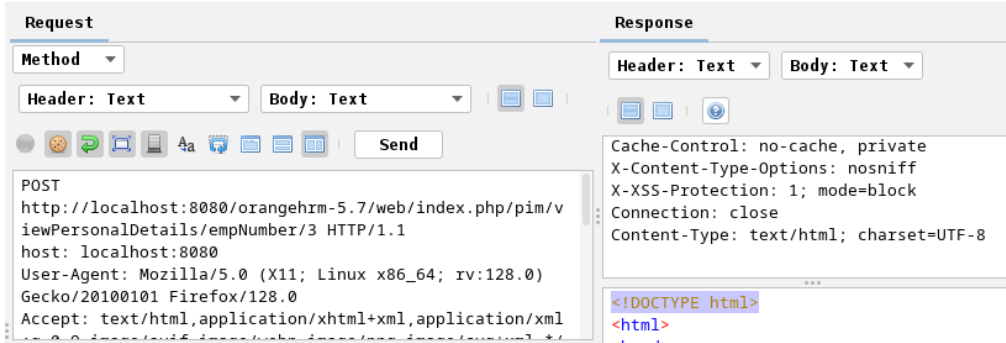
3. Test Cases Overview

A list of conducted security test cases are as listed below:

- TC01.** To check unauthorized update of users' info (ZAP)
- TC02.** To view other users' personal info as non-admin user (ZAP)
- TC03.** To check add employee feature with malicious entries (Manual)
- TC04.** To insert with improper file name to do Path Traversal Attacks (Manual)
- TC05.** To upload vulnerable XSS payloads in the file attachment (ZAP)
- TC06.** To check SQL injection vulnerability in the login functionalities (ZAP)
- TC07.** To check for bruteforce attempts vulnerability in the login functionalities (ZAP)
- TC08.** To check the access control to the users for *candidates'* application (ZAP)
- TC09.** To perform the purge action to all users and vacancy posts (Ettercap)
- TC10.** To perform the purge action to all users and vacancy posts (ZAP)
- TC11.** To validate input sanitization in user registration forms (Manual)
- TC12.** To test for session expiration after logout (ZAP)
- TC13.** To verify that file uploads are restricted to allowed types only (Manual)
- TC14.** To scan for vulnerabilities in the Buzz Module (ZAP)
- TC15.** To test SQL injection in the *search* functionalities (ZAP)
- TC16.** To check for CSRF protection on sensitive form submissions (ZAP)
- TC17.** To check DNS Spoofing for Redirecting Traffic (Ettercap)

4. Details on Each Test Cases

TC01: To check unauthorized update of users' info:

Goal	In PIM module, under Employee Information tab, all of the info about an employee can be seen (and as admin: can be modified). Goal is to check that by intercepting ongoing request-response action, and then modifying the request (as a non-admin user) if anyone can update other user's information.
Steps	<ul style="list-style-type: none">● Start ZAP as a proxy.● Log in as a non-admin user.● Go to the PIM section in <i>orangehrm</i> and access the employee list.● Capture the request to access user information in ZAP.● Modify the request (header) to attempt an unauthorized update of another user's info.● Make a <i>POST</i> request● Resend the modified request.● Observe the response in ZAP 
Expectation	Non-admin users should not be able to update any information of other users.
Observation	Non-admin users cannot update any information of other users. It is only possible by the user himself and the admin.
Risk	None

TC02. To view other users' personal info as non-admin user

Goal	This is somewhat similar to TC01, but here instead of trying to modify information (which failed), it was checked if the (private) information such as joining dates, addresses, contact numbers, or user names can be seen as a non-admin user.
------	--

- Start ZAP as a proxy.
- Log in as a non-admin user.
- Go to the PIM section in *orangehrm* and access the employee list.
- Capture the request to access user information in ZAP.
- Modify the request and make a *GET* request
- Resend the modified request.
- Observe the response in ZAP

Steps

The screenshot shows the ZAP interface with a 'Request' tab selected. The request is a GET method to the URL `http://localhost:8080/orangehrm-5.7/web/index.php/pim/viewPersonalDetails/empNumber/3`. The response is an HTTP 1.0 200 OK status with an HTML body. The response body contains the following HTML code:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
</head>
```

Expectation

Non-admin users should not be able to view any private information of other users.

Observation

Although the response returned 200 OK, The information was not visible while logged in as a non-admin user.

Risk

None

TC03. To check add employee feature with malicious entries

Goal

Here, it was check whether the system supports malicious entries as First/Middle/Last name in the *Add Employee* option. This action (adding employee) can only be done by Admin, so the goal was to check if malicious entries (or inappropriate ones) can be put in the these fields.

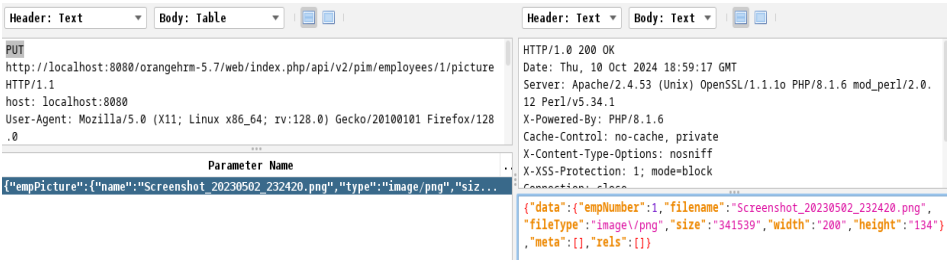
- Login as Admin
- Go to PIM
- Select Add Employee option
- Put malicious (or odd) code as name entries in the fields
- Try to create employee
- Check if system throws any warning.

Steps

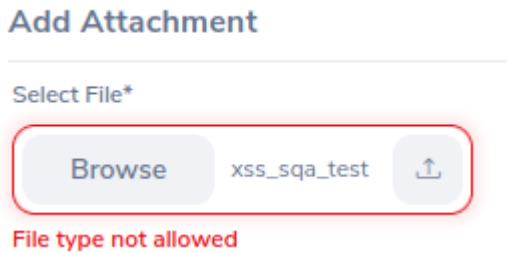
The screenshot shows the 'Add Employee' form. It has a profile picture field with a placeholder icon and an orange '+' button. Below it are three input fields for 'Employee Full Name*'. The first field contains the malicious code `<script>alert('XSS')</script>`, the second field contains `.../etc/something`, and the third field contains `.../etc/something`. Below these is an 'Employee Id' field containing the value '0008'.

	<div> <input type="checkbox"/> </div> <div> <div>Id</div> <div>↕</div> </div> <div> <div>First (& Middle) Name</div> <div>↕</div> </div> <div> <div>Last Name</div> <div>↕</div> </div>
	<div> <input type="checkbox"/> </div> <div> <div>0008</div> <div><script>alert("XSS")</div> <div></script> ../../etc/ something</div> </div> <div> <div>0001</div> <div>Abdur</div> <div>Rahman</div> </div>
Expectation	The application should have rejected the user creation attempt with inappropriate entries such as scripts, code, or any link.
Observation	It is observed that the system allows such entries in this module (though it was not allowed in the registration option).
Risk	Yes

TC04. To insert with improper file name to do Path Traversal Attacks

Goal	Target for this test case is to make a Path Traversal Attack by modifying uploaded file's name. If it is successful, then whenever a file is uploaded, it should redirect to another path (in this case to the profile menu, see 5 th step) immediately.
Steps	<ul style="list-style-type: none"> ● Start ZAP as a proxy. ● Go to My Info → Personal Details ● Click on profile icon ● Upload a valid image (jpg/png) ● Intercept the request in ZAP and modify the filename to “../../empNumber/1” ● Check if it is updated
	
Expectation	File name should not be changed as file path fashion, rather should have sanitation mechanism so that it prevents from renaming them to inappropriate names.
Observation	File Name is not changed. Put request was unmodified
Risk	None

TC05. To upload vulnerable XSS payloads in the file attachment

Goal	Similar to TC04, this test was also designed for testing file upload criteria in the My Info module. Target for this test case is to upload a XSS script as executable file.
Steps	<ul style="list-style-type: none">● Start ZAP as a proxy.● Go to My Info → Personal Details● Select Add Attachment option● Upload an XSS file● Check if it is uploaded 
Expectation	Executable files should not be allowed to be uploaded
Observation	Executable files are not allowed.
Risk	None

TC06. To check SQL injection vulnerability in the login functionalities

Goal	Goal for this TC was to verify if the application is secure against SQL injection attacks during login.
Steps	<ul style="list-style-type: none">● Start ZAP proxy browser (top right)● Go to the login page● Try a login with any username/password (as we need the request header)● Intercept request, and get the parameter name● Modify the request in <i>Repeater</i>● Put different SQL injection queries● Tested ones: <code>admin' --;</code> <code>admin' OR '1'=';</code> <code>SELECT * FROM user WHERE name = 'Admin' OR 1=1;</code>● Keep the other entries empty in the request Repeater



Expectation SQL injections should not be allowed

Observation SQL injections are not allowed in the login module.

Risk None

TC07. To check for bruteforce attempts vulnerability in the login functionalities

Goal In this test case, vulnerability against bruteforce attack was evaluated. It was also done on login module. The plan was that: first a known username has to be present, then using that name, multiple attempts on password will be carried out. Although it is obvious that it will be nearly impossible to hit a correct pswd (as pswd is 8=< chars long with upper/lower case, and special character constrains), so what actually was check is the response from the site after sending multiple attempts from the same username.

Steps

- Start ZAP proxy browser (top right)
- Go to the login page
- Put valid username in the username field (but keep password black)
- Get a login request in ZAP
- Open Intruder → Positions → Payloads
- Add 8-digit passwords in the Payloads tab (or import in a txt file)
- Select attack type as Sniper (to avoid ram issue)

Expectation Multiple attempts from same username should provide different response (like “try again”, or “account locked”, or “contact admin”, or something of that sorts.

Observation No response from the site was noted even after multiple (10 actually) attempts. The site seemed to allow these attempts without flagging any warning.

Risk Yes

TC08. To check the access control to the users for *candidates*' application

Goal	For this test case, it was check whether access control in in the <i>Recruitment</i> module can be altered. The Recruitment module contains Candidates and Vacancies options where hiring manager (could be any user) makes post mentioning the job title, description, etc as vacancy posts. On the other hand, the candidates makes application to those posts. When done, the hiring manager can short-list/reject the applicant. Now target for this test case is to see if it is possible to non-hiring managers (i.e., normal user/candidates) can alter this application system (e.g. reject it, or short-list it) themselves.
Steps	<ul style="list-style-type: none">● Login as hiring manager● Make vacancy post● Login as normal user (i.e., as a candidate)● Apply for the post● Log out● Get the url of hiring manager ('s Recruitment module)● Try to short-list/reject the applications
Expectation	Applications should be short-listed/rejected only by the respective hiring managers
Observation	Only the hiring manager and the admin can short-list/reject the applicants.
Risk	None (however the admin should not be allowed to short-list/reject the applicants); Besides no log file was generated so it cannot be said if it is the admin or the hiring manager who short-list/reject the applicants

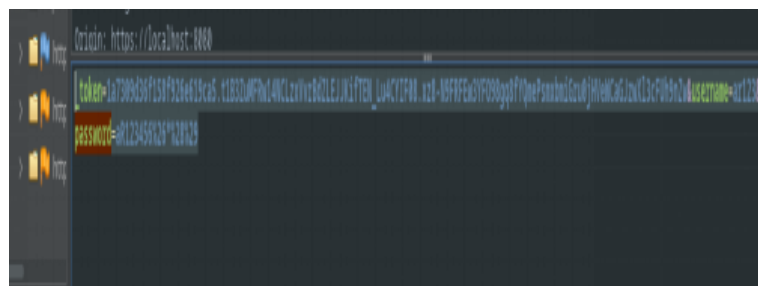
TC09. To perform the purge action to all users and vacancy posts

Goal	In the Maintenance module, there is a 2 nd login form available for the admin access. In this module, the admin can see all the purged user's details, as well as purge users and vacancy posts. So, for this test case, it was tested whether a non-admin can do it using MITM attack. Ettercap was used to carry out this attack using its ARP poisoning module. In the console, it should show the input text fields' (username, and password) entries. So the target is to get the entries and try to login as admin to the Maintenance module and try to purge all the information of users/vacancy posts.
Steps	<ul style="list-style-type: none">● Login as admin (must)● Start Ettercap as root user● Scan for the hosts● Select router's IP (gateway) as target 1● Select admin's IP as target 2● Select ARP poisoning option (top-right)

	<ul style="list-style-type: none"> ● Start MITM ● Look for input fields in the Ettercap's console
Expectation	The credential information should not be visible in the Ettercap's console
Observation	No info was noted
Risk	None

TC10. To perform the purge action to all users and vacancy posts

Goal	The goal for this TC is similar to TC09. However, in this TC, instead of Ettercap, ZAP was used.
Steps	<ul style="list-style-type: none"> ● Login as admin (must) ● Start ZAP's Spider ● Scan for all the available ports (and respective requests) ● Find "auth" request ● Look for responses in the field. ● Get the token, uname, and pswd ● Make PUT request and add them in it.



Expectation	The credential information should not be visible in the response; even if it should, then it it should have been encrypted.
Observation	All username, token, and password were found in plain text (w/o any encryption or compression).
Risk	Yes

TC11. To validate input sanitization in user registration forms











Goal	The goal for this TC is to ensure whether the user registration forms correctly sanitize inputs to prevent injection attacks and ensure that only valid data is accepted in the registry form.		
Steps	<ul style="list-style-type: none">● Go to installation step of orangehrm (as the site doesn't offer new account creation in the login page)● Locate fields like username, password, email, and full name● Test a list of various malicious entries for checking sanitization● Tested ones (in each field (as entries):		
	Category	Attempts	Result
	SQL	'; DROP TABLE users; --	×
	XSS	<script>alert('SQA Test')</script>	×
	HTML	link test	×
	Characters	!@#\$\$%^&*()	×
Expectation	Malicious entries should not be allowed as executing (or reading) them will cause security issue.		
Observation	Malicious entries in the registration are not allowed. In fact, the system required a username of minimum 5 chars, and password of 8<= chars with strict conditions.		
Risk	None		

TC12. To test for session expiration after logout

Goal	The goal for this TC is to check if the user sessions are properly invalidated upon logout preventing unauthorized access to sensitive areas of the application after a user has logged out. If the sessions can be accessed again after expiration using that session, unauthorized access to the site can be made.
Steps	<ul style="list-style-type: none">● Login with the proxy browser of ZAP● Monitor the requests (ensure login action is captured)● Go to (any) other section (for instance, viewEmployeeList), and establish an active session● Capture that request in ZAP as well● Logout● Inspect the session tokens used before and after the logout request● Use ZAP's "Request Editor" to manually enter URLs for restricted areas (e.g., http://localhost:8080/orangehrm-5.7/web/index.php/pim/viewEmployeeList).● Send the request and monitor the response.

Expectation	Once the session is expired, it should redirect to login page (not allow to use that same session)
Observation	It redirects to the login page rather than allowing to access the PIM module w/o new session.
Risk	None

TC13. To verify that file uploads are restricted to allowed types only

Goal	<p>This is somewhat similar to TC04 and TC05. However, instead of trying to upload XSS payload or File Traversal Attack, in this TC the file-upload criteria is validated. It is tested whether unacceptable files can be uploaded (encoding it to acceptable types). For instance, if the site supports jpg/png, and in that attachment other files (scripts/pdf) can be uploaded by changing their extensions.</p>																	
Steps	<div><ul style="list-style-type: none">● Login → My Info → Personal Details● Select profile icon● Upload a file (which is actually pdf/txt, but renamed to .jpg extension)● Click upload● Select “Add” in the Attachment tab● Upload a script● Click “Save”<p>Both of the files below were of txt type, but the 2nd one was <i>renamed</i> to png extension.</p><div><table><thead><tr><th>Name</th><th>Size</th><th>Type</th></tr></thead><tbody><tr><td> text_file.txt</td><td>2 B</td><td>plain text document</td></tr><tr><td> text_file.png</td><td>2 B</td><td>PNG image</td></tr></tbody></table><div><div><div>Change Profile Picture</div><div><div>Accepted</div></div><div><div>Avatar (text_file.png, gif up to 1MB, Recommended dimensions: 200px X 200px)</div><div><div>profile picture</div><div>+</div></div><div>Accepts jpg, .png, .gif up to 1MB. Recommended dimensions: 200px X 200px</div></div></div><div><div>localhost:8080 wants to open</div><table><thead><tr><th>Name</th><th>Location</th><th>Size</th><th>Type</th></tr></thead><tbody><tr><td> text_file.png</td><td>.../SQA/Assignment/Report</td><td>2 bytes</td><td>Image</td></tr></tbody></table><div>Abdur Rahman</div><div><div>profile picture</div><div>+</div></div></div></div></div></div>	Name	Size	Type	 text_file.txt	2 B	plain text document	 text_file.png	2 B	PNG image	Name	Location	Size	Type	 text_file.png	.../SQA/Assignment/Report	2 bytes	Image
Name	Size	Type																
 text_file.txt	2 B	plain text document																
 text_file.png	2 B	PNG image																
Name	Location	Size	Type															
 text_file.png	.../SQA/Assignment/Report	2 bytes	Image															
Expectation	<p>Along with file extension, the content should also be validated. If the file type if not of jpg/png (for images), then it should not be allowed to be uploaded regardless of it “renamed” extension</p>																	

Observation	For the profile picture, it was noticed that file (of not picture type) can be uploaded in the picture section if it has jpg/png extension. For the attachment part (below), it accepted pdf files (within 1MB), but any executable was denied uploading.
Risk	Yes (along with extension, file content should've been verified)

TC14. To scan for vulnerabilities in the Buzz Module

Goal	The goal for this TC is to automatically detect vulnerabilities in the Buzz Newsfeed posting feature using OWASP ZAP's active scanner. The main focus is to check if any inappropriate content (e.g., scripts, sql injection, command queries, direct link, etc.) can be posted in the Buzz Newsfeed feature.
Steps	<ul style="list-style-type: none"> ● Login is with the proxy browser of ZAP ● Go to the Buzz module ● Write post (include scripts, sql injection, command queries, direct link, etc.) ● Click post ● In ZAP, locate the request for posting content under the "Sites" tab. ● Select "Attack" → "Active Scan". ● When done, navigate to the "Alerts" tab ● Check the Alerts (Low/Medium/High)
Expectation	The content of the post should be sanitized to filter vulnerable contents
Observation	No filtering observed; all posts are as they are made. Besides ZAP's scanner did not flag anything as "risky"
Risk	Undecided

TC15. To test SQL injection in the *search* functionalities

Goal	The goal for this TC is to identify potential SQL injection vulnerabilities in the search functionalities of the application.
Steps	<ul style="list-style-type: none"> ● Login with the proxy browser of ZAP. ● Go to the Search bar. ● Perform a search with normal (w/o sql queries) entries. ● Locate the search request under the "Sites" tab. ● Right-click on the request and select "Send to Repeater." ● Open the Repeater tab and modify the search input with common SQL

	injection payloads, such as:
	<ul style="list-style-type: none"> ➤ ' OR '1'='1 ' ➤ UNION SELECT NULL, username, password FROM users -- ➤ '; DROP TABLE users; --
	<ul style="list-style-type: none"> ● Return to the “Sites” tab → select “Attack” > “Active Scan.” ● When done, navigate to the “Alerts” tab. ● Look for any alerts related to SQL injection
Expectation	Such search entries should not be able to modify any data inside the site.
Observation	No alteration in the data is noted
Risk	None

TC16. To check for CSRF protection on form submission

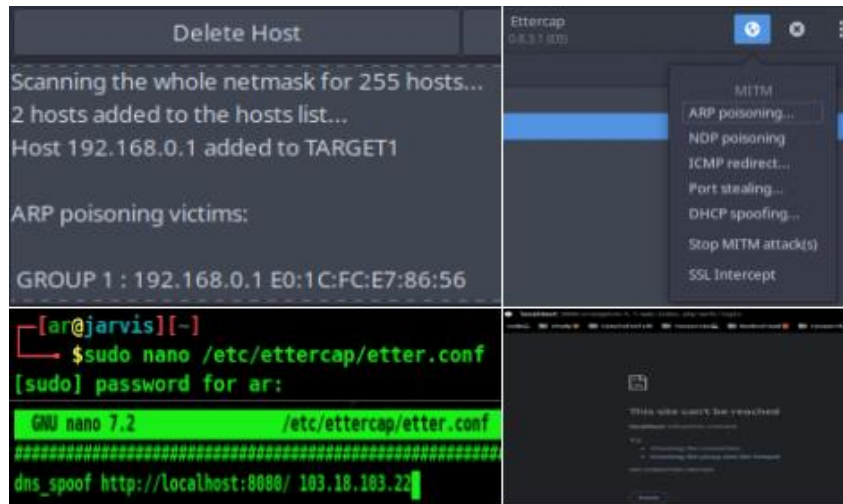
Goal	The goal for this TC is to verify that the application implements Cross-Site Request Forgery (CSRF) protection on sensitive form submissions.
Steps	<ul style="list-style-type: none"> ● Login with the proxy browser of ZAP. ● Access any form (say PIM’s Employee List) ● Perform a submission with normal entries. ● Capture the request in ZAP → Go to “Sites” tab ● Manually check for any CSRF tokens present in the request (look for hidden fields or headers) ● Go to the “Tools” → select “Anti-CSRF Tokens.” → Enable ● Return to the “Sites” tab → select “Attack” > “Active Scan.” ● When done, navigate to the “Alerts” tab. ● Look for any identified CSRF vulnerabilities.
Expectation	If the CSRF token is invalid or missing, the application should reject the request or provide alert or notification of that sorts.
Observation	No notification was seen
Risk	None

TC17. To check DNS spoofing for redirecting traffics

Goal	Goal for this TC is to test whether DNS spoofing can redirect traffic within the site to a malicious site.
------	--

- Launch Ettercap (as root)
- Go to “Sniff” → Unified Sniffing → select “network interface”
- Scan for hosts → Host → Host List
- Set router’s gateway (192.168.0.1) as Target 1
- Set <http://localhost:8080/> as Target 2
- Go to Plugins → Manage Plugins → dns_spoof
- Go to the MITM tab→ ARP poisoning.
- Start → Start Sniffing
- Check if the request is redirected to another IP (gateway).

Steps



(malicious IP list: https://www.projecthoneypot.org/list_of_ips.php)

Expectation	If attack such as this is properly addressed, the site still can be visited in the local host without any redirection.
Observation	The site was not not visited, neither was the redirected site (IP).
Risk	Undecided (this could be an issue due to the fact that the site was running on the local host (which do not have secure transfer protocol (i.e, https)

5. Conclusion

Throughout the tests, it was checked whether orangehrm was secure against some of the common attacks. The test cases included: Unauthorized Access and Privilege Escalation (TC01, TC02, TC08), Input Validation and Injection Attacks (TC03, TC06, TC11, TC15), File Upload Vulnerabilities (TC04, TC05, TC13), Brute Force and Session Management (TC07, TC12), Cross-Site Request Forgery (TC16). Data Purging (TC09, TC10), General Vulnerability Scanning (TC14), and Man-in-the-Middle attacks (TC17). it is worth noting that all the test cases were conducted on the local host, thus the tests on live server might provide different results.