

### I. Introduction

ERPNext is a comprehensive business management solution that enables small and medium-sized enterprises to manage all their business transactions and operations in one system. Its primary modules include Accounting, Human Resources, and Customer Relationship Management (CRM).

The Accounting module allows users to manage accounts, transactions, and taxes, while the Human Resources module offers features such as attendance tracking, expense claims, employee payroll management, and KPI tracking. The CRM module helps users manage presales processes, including lead capture, opportunity tracking, email and call management, quotation preparation, and pricing.

The software is built on the Frappe Framework, a Python and JavaScript-based full-stack web application development framework. The software is open-source and can be accessed on the public repository at <https://github.com/frappe/erpnext>. This report outlines the plan that will be used to test the software.

#### I.1 PURPOSE

This document provides an overview of the testing plan for the ERPNext software system, which is available on GitHub. ERPNext is an all-in-one software system that covers various functions, including accounting, warehouse management, and customer relations management. The system is built on the Frappe Framework, which uses Python and JavaScript. To ensure the testing project's effectiveness, it is important to limit its scope, especially when dealing with complex software like ERPNext. This document will explain the project's structure, assumptions, limitations, and schedule.

#### I.2 CONSTRAINTS

To ensure a comprehensive testing of software, several factors including available resources, intended purpose of the software, and development practices are taken into consideration. However, the process is often constrained by various assumptions and limitations that restrict the extent to which it can be conducted.

In this project, it is assumed that the team has knowledge of software development and is familiar with software testing tools and concepts. The testing will be conducted on an 8GB laptop which may restrict the type of testing tools that can be used.

Additionally, the project is limited to using open-source testing tools, which may restrict the level of testing that can be achieved.

## **II.SCOPE MANAGEMENT**

The objective of this project is to assess the software for any errors, deficiencies, or unfulfilled requirements by conducting a comprehensive testing process. The scope of the testing includes various aspects of the software, such as ease of installation and usability, graphical interfaces, and each of the ERPNext services (excluding customized services) to ensure that all requirements are met.

Additionally, the testing will evaluate the software's adherence to Object Oriented Programming concepts and software principles to identify any programming gaps. Furthermore, the security of the software will be assessed by testing APIs and verifying its resilience against threats such as Cross-Site Scripting.

## **III. WORK BREAKDOWN STRUCTURE**

The project will be conducted in the following phases:

A. Testing requirements specification

B. Planning

In this test functions to test, test design methods and test schedules will be planned. The requirements will also be redefined according to the plan designed.

C. Selection of testing tools and methods

D. Test case specification and testing.

Testing will be done via the black box methodology, which will involve behavioral testing. Therefore, the test cases will be defined based on user experience and the general principles of software.

E. Reporting

## **IV. DEPENDENCIES**

- The development of a test plan requires the specification of requirements. It will be impossible to create a productive plan without clearly stated needs and objectives.
- Resource planning and test environment setup need the development of a test plan. It is impossible to locate the required resources and build up a suitable testing environment without a clearly defined test plan.
- Test Execution requires Resource Planning as a prerequisite. Effective testing cannot be done without sufficient resources.
- Test Execution requires the setup of the test environment. For accurate and trustworthy test results, a testing environment that has been correctly designed is necessary.
- Reporting is a requirement for test execution. Analysis and documentation of test results are required for stakeholders.

## **V. Objectives**

The testing plan will include the following objectives:

- Ensure ERPNext functions as intended and meets SMEs' business requirements.
- Uncover defects, bugs, and other issues in the software.
- Ensure compatibility with major platforms, operating systems, and browsers.
- Validate the security of ERPNext, including user authentication, data encryption, and protection against vulnerabilities.
- Test the system's performance, speed, scalability, and response time under different load conditions.
- Evaluate the software's usability, interface, navigation, and user experience to ensure it is user-friendly.
- Test compliance with relevant industry standards and regulations such as accounting regulations and data privacy laws.
- The objectives will guide the testing effort and ensure the software meets business requirements and quality standards.

Any anticipated limitations during the testing process for each objective will be described in the document.

## **VI. FEATURES TO BE TESTED**

The testing process will cover all features within the core modules of ERPNext software, including:

- Accounting: sales and purchase management, inventory tracking, attendance tracking, expense claims, employee payroll management, and KPI tracking.
- CRM: lead capture, opportunity tracking, email and call management, and quotation preparation for effective selling prices.
- The Human Resource module includes features such as attendance tracking, expense claims, and employee payroll management.

ERPNext has also customized its features for different industries, including manufacturing, healthcare, and agriculture, with functionalities such as stock replenishment, managing sales orders, customers, suppliers, shipments, deliverables, order fulfillment, creating purchase invoices, and journal entries for accounting entries like payments and credits.

### **VI.A TESTING STRATEGY**

1. The testing approach will involve analysing the business requirements for each module and designing test scenarios and cases accordingly.
2. Prioritization of testing will be based on the importance of features and modules, determined via the gathered requirements.
  - a. Prioritization criteria will include criticality to businesses, customer impact, and complexity.
  - b. Criticality will evaluate how essential the feature is to SMEs' day-to-day operations.

- c. Customer impact will determine which feature or module has a direct impact on the customer experience.
- d. Complexity will prioritize features or modules that are more complex and have a higher risk of defects or issues.
3. The testing approach will include system testing to ensure that the software meets the performance, compatibility, compliance, and usability requirements. This will be done using black box testing.
4. Unit testing will be conducted on individual functions within the core modules using pytest and examining code files for bugs and issues.
5. Functional testing will be performed to verify that the system satisfies SMEs' business requirements and specifications.
6. Integration testing will be carried out to ensure that the core modules work well together and with third-party integrations.
7. Automated testing tools such as Selenium and Robot Framework will be utilized to reduce testing time and increase test coverage.
8. Jira will be used to manage test cases and track bugs and issues.

## **VI.2. Unit and Functional Testing**

Unit and functionality testing will be performed on the UI and code level for each core module of ERPNext, including security testing. The testing will ensure that all functions within the core modules are tested, cover all possible input combinations and edge cases, and execute all code paths at least once.

Code coverage analysis will be used to measure the comprehensiveness of the testing effort, with a minimum requirement of 60% code coverage. The requirement traceability matrix (RTM) technique will be used to trace requirements and ensure that all requirements are covered by the tests.

Functionality testing will be conducted using the BlackBox approach, while unit testing will use the WhiteBox approach.

- The testing process will start by reviewing the functional requirements and specifications for the feature or system being tested.
- Test scenarios will be created based on the functional requirements and specifications. Each team member will create scenarios that cover all possible inputs, workflows, and expected results.
- Test cases will be designed to outline the specific steps to execute each scenario.
- The test cases will be executed.
- Results of each test case, including pass/fail status, any defects or issues discovered, and additional notes or comments, will be recorded.

### **CREATING CHART OF ACCOUNTS:**

Test Case Identifier	TC 001
Test Items/feature	Navigate Accounting module of ERPNext
Preconditions	ERPNext application should be available.
Test Steps	1. Open ERPNext application 2. Navigate accounting module.
Output Specifications	Accounting module should open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 002
Test Items/feature	Chart of accounts
Preconditions	Accounting module of the application should be open.
Test Steps	1. Open ERPNext application 2. Navigate accounting Module 3. Click on Chart of accounts icon.
Output Specifications	Chart of accounts should display.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 003
Test Items/feature	New Chart of accounts
Preconditions	Chart of accounts should be open.
Test Steps	1. Open Charts of account. 2. Click “New”
Output Specifications	Creation space of new charts of account should open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 004
Test Items/feature	<u>U</u> nique name for the chart of accounts
Preconditions	New Chart of accounts should work.
Test Steps	1. Name the Chart of accounts a unique name.
Output Specifications	The given name should be saved
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 005
Test Items/feature	Account details for chart.
Preconditions	The chart should be saved.
Test Steps	1) For the chart of accounts enter the following data: a) Account number b) Account type c) Currency
Output Specifications	The entered details must be saved
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 006
Test Items/feature	Save Chart of accounts
Preconditions	The Above test cases should be successful
Test Steps	1. Save the chart of accounts.
Output Specifications	Saved new chart of accounts.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 007
Test Items/feature	Newly saved chart of accounts should appear on the list.
Preconditions	A new chart of accounts should be created and saved.
Test Steps	<ol style="list-style-type: none"> <li>1. Verify the chart of accounts with the give name has been saved.</li> <li>2. Navigate Through Chart of accounts to test if the new one has been added to the list.</li> </ol>
Output Specifications	The new one should appear on the list of Chart of accounts.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 008
Test Items/feature	Edit and Delete Chart of accounts.
Preconditions	List of Chart of accounts should be saved and should be accessible
Test Steps	<ol style="list-style-type: none"> <li>1. Open chart of accounts.</li> <li>2. Attempt to edit and delete to verify the actions are possible.</li> </ol>
Output Specifications	Specific chart of accounts should be Edited or deleted.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 009
Test Items/feature	Search and sort Chart of accounts.
Preconditions	List of Charts should be open.
Test Steps	<ol style="list-style-type: none"> <li>1. Open chart of accounts.</li> <li>2. Search a new chart in chart of accounts.</li> <li>3. Sort for the chart of accounts in specific order.</li> </ol>

Output Specifications	<ol style="list-style-type: none"> <li>1. The chart searched should be opened.</li> <li>2. The list Should be sorted accordingly.</li> </ol>
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 0010
Test Items/feature	Export and import Chart of accounts.
Preconditions	Appropriate features should be used for import and export.
Test Steps	<ol style="list-style-type: none"> <li>1. Export the chart of accounts of the file.</li> <li>2. Then Import the file back into the application.</li> </ol>
Output Specifications	Charts should import and export successfully.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

### **TAX CONFIGURATION:**

Test Case Identifier	TC 011
Test Items/feature	Navigate Accounting module.
Preconditions	ERPNext application should be available.
Test Steps	<ol style="list-style-type: none"> <li>1. Open the ERPNext application.</li> <li>2. Navigate the accounting module.</li> </ol>
Output Specifications	The Accounting module should open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 012
Test Items/feature	Check if the tax template opens.
Preconditions	Accounting module should open.



Test Steps	<ol style="list-style-type: none"> <li>1. Open tax module.</li> <li>2. Click on “item tax template” label</li> </ol>
Output Specifications	Item tax template should open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 013
Test Items/feature	Add Item Tax template.
Preconditions	Item tax template should open.
Test Steps	<ol style="list-style-type: none"> <li>1. Open Item tax template.</li> <li>2. Click on “Add Item tax template”.</li> </ol>
Output Specifications	New item tax template should open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 014
Test Items/feature	Tax and template details should be saved.
Preconditions	Item tax template should open.
Test Steps	<ol style="list-style-type: none"> <li>1. Open New item tax template.</li> <li>2. Enter title, company, tax and tax rate.</li> <li>3. Save the entered details.</li> </ol>
Output Specifications	The entered details like company, tile, tax should be saved.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 015
Test Items/feature	Save the item tax template.
Preconditions	The entered details should be saved.

Test Steps	1. Save the item tax template.
Output Specifications	The item tax template should be saved.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 016
Test Items/feature	Verify the tax template enabled works automatically.
Preconditions	The item tax template should be saved.
Test Steps	1. Search for the tax template saved. 2. Verify if the tax template works automatically.
Output Specifications	The tax should be working based on the type of tax template.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

### **CREATE COST CENTERS AND COST TREES:**

Test Case Identifier	TC 017
Test Items/feature	Navigate Accounting module of ERPNext application.
Preconditions	ERPNext application should be available.
Test Steps	1. Open ERPNext application 2. Navigate the accounting module and open.
Output Specifications	The accounting module should open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 018
Test Items/feature	Open Cost Center page.
Preconditions	Accounting module should open.

Test Steps	<ol style="list-style-type: none"> <li>1. Open Accounting module</li> <li>2. Click on “cost center” icon</li> </ol>
Output Specifications	Cost Center page should open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 019
Test Items/feature	Create new cost center page.
Preconditions	Cost center page should be open.
Test Steps	<ol style="list-style-type: none"> <li>1. Click on “new” button.</li> </ol>
Output Specifications	A new cost center page will open.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 020
Test Items/feature	Unique name and description for cost center page.
Preconditions	A new Cost center page should be open.
Test Steps	<ol style="list-style-type: none"> <li>1. Enter unique name and description if applicable for the cost center page.</li> <li>2. Save the details for the cost center page.</li> </ol>
Output Specifications	New cost center should have the given name and description if available.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 021
Test Items/feature	Assign the new cost center to a parent cost center.
Preconditions	New cost center should be saved.

Test Steps	<ol style="list-style-type: none"> <li>1. Assign the new cost center to a parent cost center if applicable.</li> <li>2. To create cost center tree.</li> </ol>
Output Specifications	The cost center should be added to present parent.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 022
Test Items/feature	Save the cost center.
Preconditions	New Cost center should be created.
Test Steps	<ol style="list-style-type: none"> <li>1. Click on “Save”</li> <li>2. Then save the cost center.</li> </ol>
Output Specifications	The cost center should have been saved.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 023
Test Items/feature	Verify if the cost center saved is available to navigate.
Preconditions	Cost center should be saved.
Test Steps	<ol style="list-style-type: none"> <li>1. Verify if the cost center is saved successfully.</li> <li>2. Verify if it appears in the cost center list.</li> </ol>
Output Specifications	The cost center should be accessible on the cost center.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 024
Test Items/feature	Edit or delete cost center.
Preconditions	Cost center should be saved.

Test Steps	<ol style="list-style-type: none"> <li>1. Edit the cost center.</li> <li>2. Delete the cost center.</li> </ol>
Output Specifications	The Cost center should be edited and deleted accordingly.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 025
Test Items/feature	Search and sort in the list of cost centers
Preconditions	Cost center should be saved.
Test Steps	<ol style="list-style-type: none"> <li>1. Search for the cost center.</li> <li>2. Sort the cost center in a specific way.</li> </ol>
Output Specifications	The searched cost center should be visible in list of cost center and they should be sorted accordingly.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 026
Test Items/feature	Export and import cost centers.
Preconditions	Cost center list should be open.
Test Steps	<ol style="list-style-type: none"> <li>1. Export a cost center file using appropriate features.</li> <li>2. Import the cost center to the application.</li> </ol>
Output Specifications	The cost centers should be exported and imported successfully.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

### **CREATE EMPLOYEE ACCOUNT:**

Test Case Identifier	TC 027
----------------------	--------

Test Items/feature	Search “employee”..
Preconditions	ERPNext application should be open.
Test Steps	1. Search for “employee” on top search bar.
Output Specifications	The search bar should work and the keyword should give a list of results.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 027
Test Items/feature	“New Employee”
Preconditions	Search tab for employee should be displayed.
Test Steps	1. Click on “New Employee” result from the result.
Output Specifications	The new employee selection should be on the search tab.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 028
Test Items/feature	Enter Employee details.
Preconditions	New Employee TABe should be open.
Test Steps	1) In the New Employee tab enter the following employee details: Name, gender, Date of Birth, Address, Emails, And contact details.
Output Specifications	The employee details should be saved.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 029
----------------------	--------

Test Items/feature	Save Employee joining details.
Preconditions	Employee details should be saved under new employee.
Test Steps	1. Enter the joining details, offer date, confirmation date, contract end date, notice days and date of retirement.
Output Specifications	The joining details should be saved.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

Test Case Identifier	TC 030
Test Items/feature	Save the employee account.
Preconditions	The employee details should be saved.
Test Steps	1. Save the employee account.
Output Specifications	The account for new employee with the specified details should be created and saved.
Inter-case Dependencies	N/A
<b>Test Result</b>	<b>Pass</b>

### VI.3. System and Integration Testing

System testing will be conducted in a BlackBox approach to validate that the software meets all specified requirements and functions as intended in a real-world environment, from a user's perspective.

- Integration testing will involve combining the three core modules in different ways to test compliance, performance, and compatibility.
- System testing will be performed before integration testing, both based on the BlackBox approach, to ensure compliance, compatibility, and usability requirements are met.
- The testing efforts for both will ensure that all requirements and business processes are tested to meet the necessary performance, compatibility, usability, and compliance requirements.
- Test coverage analysis and traceability matrices will be used to ensure comprehensive testing, with error frequency used as a completion criterion.

The following steps are followed to achieve system and integration testing:

- a) Review the requirements for the feature or system being tested.
- b) Create test scenarios that cover all possible combinations of inputs, workflows, and expected results.

- c) Design test cases that outline the specific steps to execute each scenario.
- d) Execute the test cases.
- e) Record the results of each test case, including pass/fail status, any defects or issues discovered, and additional notes or comments.

Additionally, create their test scripts for their specific type of testing.

## A. SYSTEM TESTING

**System testing involved testing the entire system as a whole to ensure that it is functioning as intended and that all the components are working together correctly.**

In this case, the following tests were conducted.

### Integration testing.

This involved testing how well different modules or components of the system work together. The following tests were conducted:

- A. Create a purchase order and verify that it updates supplier data, and item data and can create a purchase invoice from the order.

**Test Date:** 04/06/2023 - 04/11/2023

### Test Results:

Here is the interface for creating the purchase order.

The screenshot displays the 'New Purchase Order' form in the ERPNext application. The form is organized into several sections: 'Details', 'Address & Contact', 'Terms', and 'More Info'. The 'Details' section is currently active and contains the following fields:

- Series \***: A dropdown menu showing 'PUR-ORD-.YYYY.-'.
- Date \***: A date field set to '05-03-2023'.
- Supplier \***: A text input field.
- Required By**: A date field set to '05-24-2023'.
- Accounting Dimensions**: A section with two sub-fields: 'Cost Center' and 'Project', both with text input fields.
- Currency and Price List**: A section with a dropdown menu.

On the right side of the form, there are two checkboxes: 'Apply Tax Withholding Amount' (checked) and 'Is Subcontracted' (unchecked). Below these checkboxes is a 'Tax Withholding Category' field. At the top right of the form, there are buttons for 'Get Items From', 'Tools', and 'Save'. A search bar is located at the top of the page, and a 'Show all' button is at the bottom right.

To create the purchase order you add the details, Supplier, Date of order, Date of order fulfillment (Required), order items, taxes, and charge.



**Additional Comments:** This test proved the integration of the order creation module with supplier creation, item creation, and stock ledger module.

B. Create a journal entry and verify that it updates the general ledger correctly

**Test Date:** 04/06/2023 - 04/11/2023

### Test Results:

Adding a journal entry to the system followed the following steps.

- Accessing the accounting module.
- Clicking on 'New Journal Entry.'
- Add the transactions under accounting entries
- Click on save at the top right of the page.
- After saving, click on submit to submit the entries. Then go back to accounting then click on the general ledger to see the transactions.

- The journal entry will look like this,

ERPNext | erpnext | Explorer | ERPNext | frappe | ERPNext | ERPNext | ERPNext | gmail | ERPNext | Acc X | [GitHub] | (2) Wi-Fi | + | - | X

anandhi.erpnext.com/app/journal-entry/new-journal-entry-1

Search or type a command (Ctrl + G) | Help | AN

**Accumulated Depreciation - AT** | Journal Entry | View | Actions | < | > | Print | ... | Cancel

Assigned To: +

Attachments: Attach File +

Shared With: +

Tags: Add a tag ...

0 | 0 | FOLLOW

Entry Type: Journal Entry

Company: AR Tech

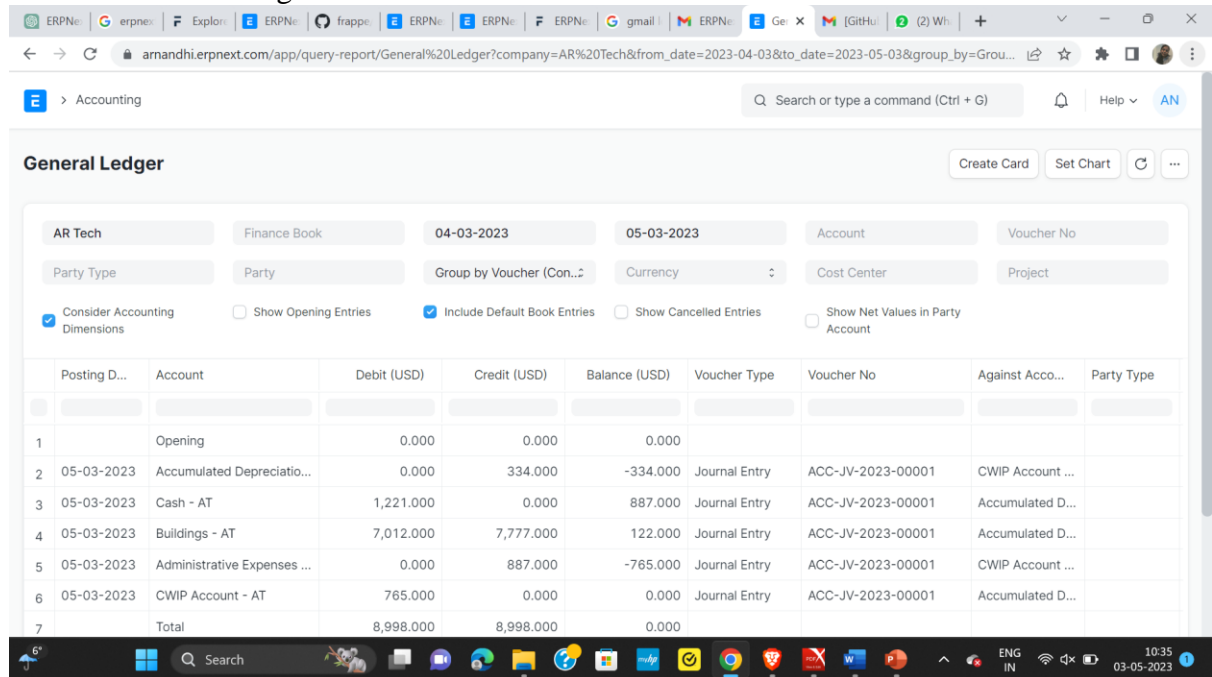
Posting Date: 05-03-2023

No.	Account *	Party Type	Party	Debit	Credit	
1	Accumulated Deprecia...			\$ 0.00	\$ 334.00	Edit
2	Cash - AT			\$ 1,221.00	\$ 0.00	Edit
3	Buildings - AT			\$ 7,012.00	\$ 0.00	Edit
4	Buildings - AT			\$ 0.00	\$ 7,777.00	Edit
5	Administrative Expens...			\$ 0.00	\$ 887.00	Edit

WhatsApp Image...jpeg | Show all | X

6° | Search | ENG IN | 10:34 | 03-05-2023

- While the ledger looks like this.



The screenshot displays the ERPNext General Ledger report for the company 'AR Tech'. The report is filtered for the period from 04-03-2023 to 05-03-2023. The table shows the following entries:

Posting Date	Account	Debit (USD)	Credit (USD)	Balance (USD)	Voucher Type	Voucher No	Against Account	Party Type
1	Opening	0.000	0.000	0.000				
2	05-03-2023	Accumulated Depreciation	0.000	334.000	-334.000	Journal Entry	ACC-JV-2023-00001	CWIP Account ...
3	05-03-2023	Cash - AT	1,221.000	0.000	887.000	Journal Entry	ACC-JV-2023-00001	Accumulated D...
4	05-03-2023	Buildings - AT	7,012.000	7,777.000	122.000	Journal Entry	ACC-JV-2023-00001	Accumulated D...
5	05-03-2023	Administrative Expenses ...	0.000	887.000	-765.000	Journal Entry	ACC-JV-2023-00001	CWIP Account ...
6	05-03-2023	CWIP Account - AT	765.000	0.000	0.000	Journal Entry	ACC-JV-2023-00001	Accumulated D...
7	Total	8,998.000	8,998.000	0.000				

This shows integration between the journal and ledger modules.

## **A. Performance Testing**

Performance testing of ERPNext will include testing its stability, speed, and responsiveness under different workloads and stress conditions. The testing will aim to ensure that all functions and features meet the necessary performance requirements and goals. The comprehensiveness of the testing effort will be judged based on the following criteria:

- a) Response time, which is the time taken for the system to respond to user actions, queries, and commands.
- b) Throughput, which is the number of transactions or operations the system can handle within a given period.
- c) Load capacity, which is the maximum load or number of concurrent users the system can handle without crashing or slowing down.
- d) Resource utilization, which is the usage of system resources such as CPU, memory, and disk space during performance testing.
- e) Error rate, which is the frequency and severity of errors and exceptions that occur during performance testing.
- f) Each criterion will be measured and compared against the desired levels specified in the requirements.

The following steps will be done for performance testing:

- Review the performance requirements specified in the requirements documentation.
- Create test scenarios that cover different workloads and stress conditions.
- Design test cases that outline the specific steps to execute each scenario.
- Execute the test cases.
- Record the results of each test case, including the measured response time, throughput, load capacity, resource utilization, and error rate.

## **B. Requirements and tools.**

- The testing effort will require several tools including Jira for test management, Selenium and Robot Framework for test automation, Apache JMeter for load testing, OWASP ZAP for security testing, and SonarQube and Pytest for code analysis.
- Jenkins and GitLab CI will be used as continuous integration and delivery tools.
- The installation of these tools will require specific hardware capabilities including a RAM capacity of at least 4GB, 250GB or more of disk capacity, a processor speed of 2.0GHz or higher, and either Windows or Linux operating systems.
- The testing efforts will require the following automation testing tools.
  - A test management tool, Jira.
  - Test automation tools. Selenium (Selenium 2023) and Robot Framework (Robot Framework 2023)
  - Load testing tool, Apache JMeter.

- Security testing tool, OWASP ZAP
- Code analysis tools SonarQube and Pytest
- Continuous integration and delivery tools, Jenkins and GitLab CI (Jenkins 2023).

## **VII. Test Record Keeping**

- Test record keeping for ERPNext involves documenting and organizing all testing-related information for future reference.
- Test cases and scenarios should be documented with details of the inputs, expected results, and actual results.
- Any defects or issues discovered during testing should be recorded in a defect tracking tool such as Jira, along with details such as severity, steps to reproduce, and other relevant information.
- Test results and progress should be documented and reported regularly to stakeholders, including any issues or risks that may impact the testing effort.
- Documentation should be organized and accessible, with clear and consistent naming conventions for files and folders.
- A version control system such as Git should be used to track changes to test cases and other testing-related documentation.
- Test summary reports should be generated at the end of each testing phase, highlighting the test results, defects found, and overall test coverage.

## **VII. TEST SUMMARY**

- **Project Name:** Software Testing
- **System Name:** ERPNext
- **Version Number:** 14

## 1. Security testing.

Tested how well the system handles user authentication and access control. The following tests were conducted in this section.

1. Tested user authentication and authorization to ensure that users can only access the data and functionality that they are authorized to use

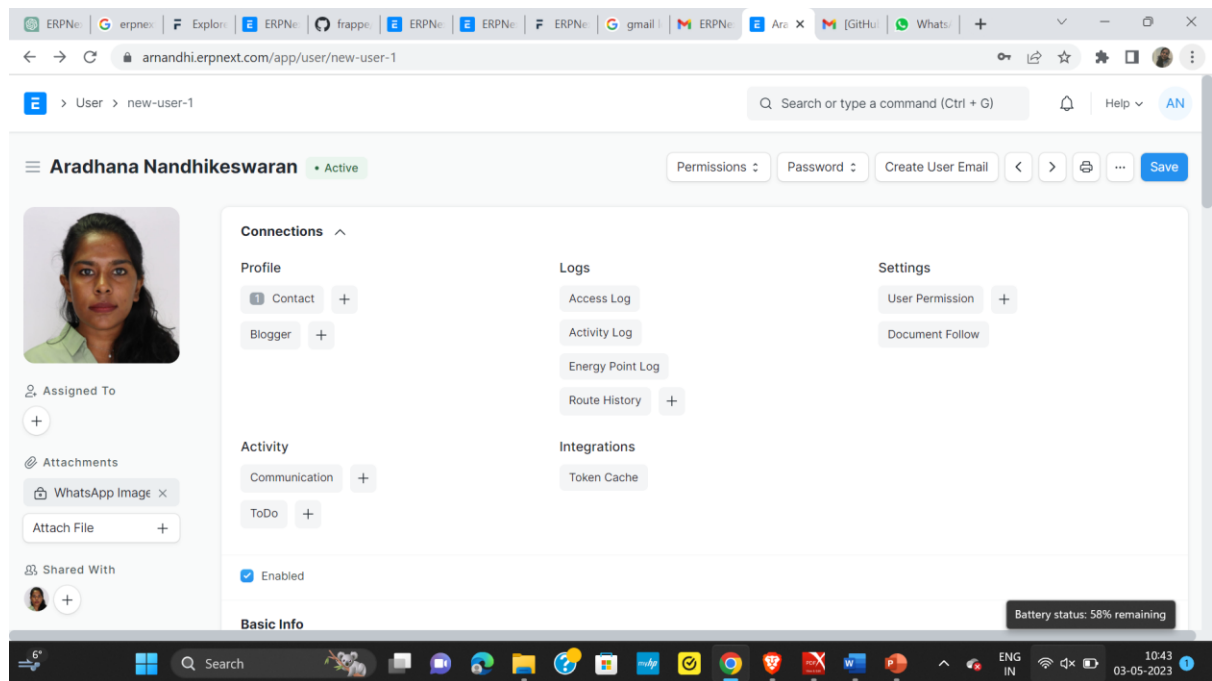
**Test Date:** 04/06/2023 - 04/11/2023

## 2. Test Results:

To perform this test, the following steps were followed.

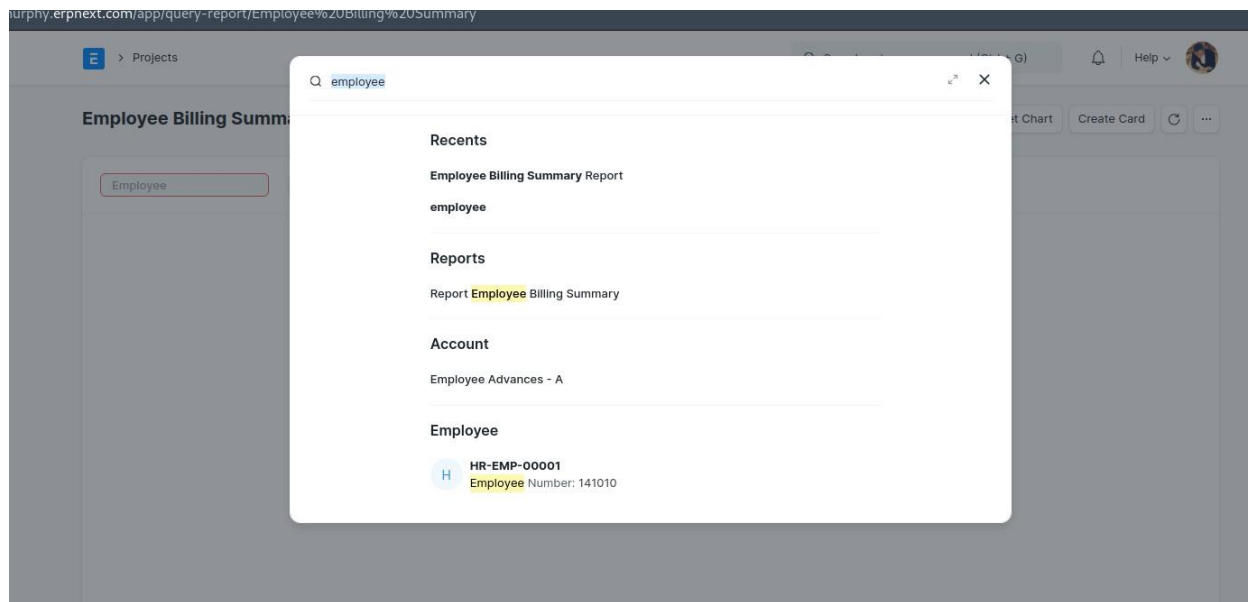
1. Click on the user icon to access the user creation page.
2. Click on the 'add user' button to initiate creating a user.
3. A widget popup is used to add the details of the user. After creating the user, click on the profile to open the characteristics assigned to the user. Including roles.
4. Assign a few roles then test the user against those roles.

Here is a user name Aradhana Nandhikeswaran



Here are the roles assigned.

The test involved testing if the user can create employees, or manage them. The following image shows the user searching for an employee



As viewed the user could not view the “new employee label” therefore they could not create the user. They could not also access the users labeled.

**Additional Comments:** ERPNext does not have a label/button that a user can use to access the employee creation interface. To access the interface, you have to use the search bar at the Top.

### 3. FUNCTIONAL TESTING

Functional testing involves testing specific features or functionalities of the system. The following features were tested.

- a. Creating a Chart of accounts
- b. Tax Configuration
- c. Creating Cost Centers and cost center trees
- d. Creating Employee Account

#### 4. Creating Chart of Accounts:

**Test Date:** 04/08/2023 - 04/09/2023

## 5. Test Results:

In creating a chart of accounts, ERPNext automates this process while creating the initial company account. It creates a standard chart of accounts with standard accounts.

Once the chart of accounts is created, it is then possible to create a child account for one of the standard accounts. The child account contains the following details:

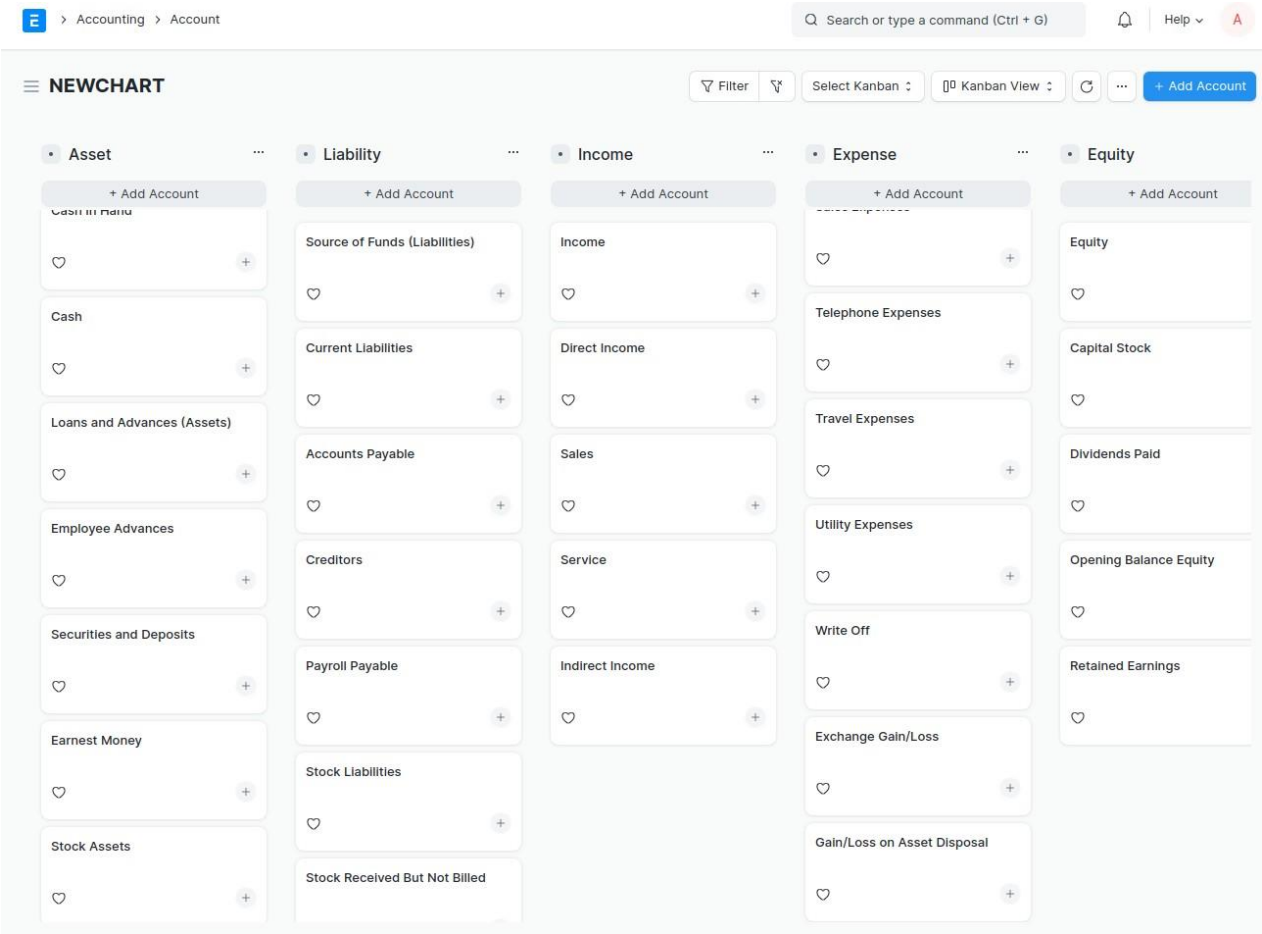
- Account Name
- Is Group
- Company
- Currency
- Begin typing for results.
- Parent Account
- Account Type (Setting Account Type helps in selecting this Account in transactions.)
- Rate (Rate at which this tax is applied).

This is shown in the screenshot below.

The screenshot shows the 'New Account' form in ERPNext. The breadcrumb navigation at the top reads 'Accounting > Account > new-account-2'. A search bar with the placeholder 'Search or type a command (Ctrl + G)' and a 'Help' dropdown are also visible. The form title is 'New Account' with a 'Not Saved' indicator and a 'Save' button. The form contains the following fields and options:

- ☐ Disable
- Account Name \* (Text input with value 'Assigned')
- ☐ Is Group
- Company \* (Text input with value 'Alex254')
- Currency (Text input with value 'KES')
- Parent Account \* (Text input)
- Account Type (Dropdown menu)
- Setting Account Type helps in selecting this Account in transactions. (Help text)
- Rate (Text input)
- Rate at which this tax is applied (Text input)
- Balance must be (Text input)

The account is then listed in the chart of accounts as shown below.



**Additional Comments:** The software was still restricted by subscription charges, therefore features such as account group creation were not available. Therefore, the full extent of this feature could not be tested.



## 6. Tax Configuration

**Test Date:** 04/08/2023 - 04/09/2023

## 7. Test Results:

In ERPNext, creating and configuring taxes is eased by the availability of prepared templates. The templates consist of these details.

- Title
- Company
- Tax Account Name
- Tax Rate

As shown

below.

The screenshot shows the 'New Item Tax Template' form in ERPNext. The form is titled 'New Item Tax Template' and has a 'Not Saved' status. It includes a 'Save' button in the top right corner. The form fields are as follows:

- Title \***: P.A.Y.E
- Company \***: Alex254
- Tax Rates**: A table with columns for 'No.', 'Tax', and 'Tax Rate'. The table contains one row with '1' in the 'No.' column, 'Expenses Included In Valuation - A' in the 'Tax' column, and '10' in the 'Tax Rate' column. There is an 'Add Row' button at the bottom of the table.

Filling these details configures a tax that can be connected to various accounts.

The template names were Sales Taxes and Charges template.z

## Creating Cost Centers and cost center trees

**Test Date:** 04/08/2023 - 04/09/2023

## Test Results:

In ERPNext, Cost centers are organizational units within a company that is responsible for managing and controlling costs associated with specific business activities or functions. In other words, a cost center is a department, team, or individual that is responsible for controlling the costs associated with a particular aspect of a company's operations.

To create these centers the following steps were followed.

- Open the ERPNext application and navigate to the Accounting module.

- Click on the "Cost Centers" icon.
- Click on the "New" button to create a new cost center.
- Enter a unique name for the cost center and a description, if applicable.
- Assign the cost center to a parent cost center, if applicable, to create a cost center tree.
- Save and verify that the center can be edited, searched, and deleted.

New Cost Center

• Not Saved

Convert to Group

...

Save

Cost Center Name \*

Parent Cost Center \*

Company \*

Alex254

☐ Is Group

☐ Disabled

Successfully added test centers appear as follows.

Cost Center

Iter By

Assigned To

Created By

Alt Filters

Tags

Low Tags

ID

Cost Center Num

Company

Filter

Last Updated On

+ Add Cost Center

<input type="checkbox"/>		ID	Status	Cost Center Name	Cost Center Num...	Parent Cost Center	2 of 2
<input type="checkbox"/>		Alex254 - A	Enabled	Alex254			- 23 h 0
<input type="checkbox"/>		Main - A	Enabled	Main		Alex254 - A	- 23 h 0

## 8. Creating Employee Account

**Test Date:** 04/08/2023- 04/09/2023

## 9. Test Results:

To add an employee account the following steps were followed.

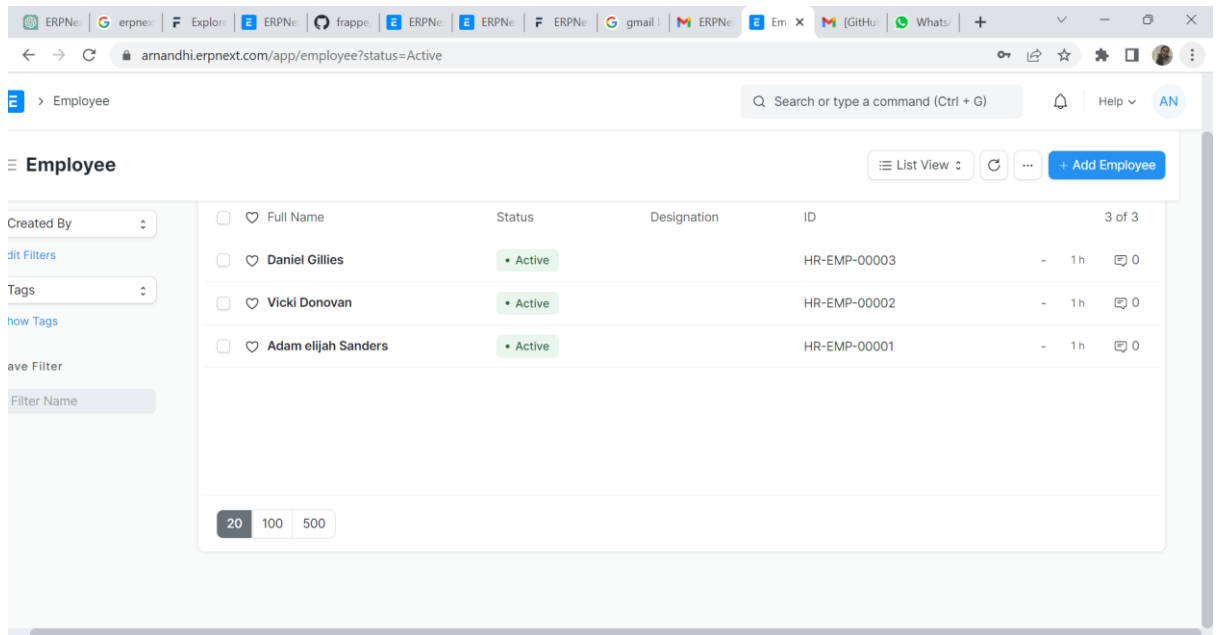
- Search 'Employee' on the top search bar.
- Click on the 'New Employee' result.
- Enter employee details, Names, gender, date of birth, address, emails, and contact details.
- Enter Joining details, offer date, confirmation date, contract end date, notice days, and date of retirement
- Save the employee account.

A successfully added employee account resembled the following.

The screenshot shows the 'New Employee' form in the ERPNext application. The browser address bar indicates the URL is `amandhi.erpnext.com/app/employee/new-employee-1`. The form is for an employee named 'Adam elijah Sanders' with a status of 'Active'. The 'Overview' tab is selected, showing fields for First Name, Middle Name, Last Name, Full Name, Gender, Date of Birth, Date of Joining, Status, and Salutation. The 'User Details' and 'Company Details' sections are also visible. The left sidebar contains options for 'Assigned To', 'Attachments', 'Shared With', and 'Tags'.

Field	Value
First Name *	Adam
Middle Name	elijah
Last Name	Sanders
Full Name	Adam elijah Sanders
Gender *	Male
Date of Birth *	08-21-1996
Date of Joining *	05-03-2023
Status *	Active
Salutation	

**Additional Comments:** Additional details of the system such as Attendance and leaves, Salary, Address, and Contacts were also captured.



## 10. TEST ASSESSMENT

- The testing process was differentiated into three types of tests system testing, functional tests, and unit testing. System testing is then divided into two types integration and security testing.
- The approach used in the system and functional testing is black box mainly. Where we focus on testing the system without any knowledge of the inner workings of the system. The black box testing was a useful technique when testing a complex system such as ERPNext as it allowed us to test the system from the user's perspective without requiring in-depth knowledge of the system's internal workings.
- We test how modules performing different functions integrated into system testing without examining the code. In functional testing, we test various features via the system's Graphical interface. Steps implemented in all the test cases follow this concept where they are simple and reliant on just the graphical interface.
- Compatibility testing was also done when running the system in the Linux operating system, the recommended testing, and on Chrome and Firefox browsers

## **TEST RESULTS**

In the original test plan, the objectives were as follows, the testing process deviated from some of these objectives as shown below:

- **Functionality assessment:** The testing process ensured that ERPNext software functions as intended, but additional test cases were added to test essential features of the system.
- **Defect Testing:** The test cases were designed to detect any defects in the system, ranging from simple UI glitches to major functionality issues.
- **Compatibility assessment:** The testing process ensured ERPNext is compatible with major platforms, operating systems, and browsers by testing on a Linux machine and Chrome and Firefox browsers.
- **Security assessment:** User authentication and authorization mechanisms were tested, but data encryption tests were not performed.
- **Performance Testing:** The testing process did not involve stress testing, but the system worked as expected under optimum conditions.
- **Usability validation:** Usability validation was performed via black box testing, but the testing process was not extensive enough to involve recruiting test users.
- **Compliance testing:** The testing process did not cover compliance testing to ensure ERPNext complies with all relevant industry standards.

No software issues were discovered in the testing process. Further testing of the software might be required.

## **TEST INCIDENTS**

During the testing process, the following testing incident was experienced.

The absence of clear labels to access employee information or the employee creation interface on the left side of the page resulted in the team having to look for an alternative solution. They eventually resorted to using the custom search bar on the homepage.

## RISK MANAGEMENT AND MITIGATION

- Inadequate resources.
- Constant changing of requirements.
- Unfamiliarity with technologies.

The following steps are taken to overcome the risks:

To ensure that the available resources are completely optimized, popular tools and techniques will be favored above other ones. This will also guarantee that technological unfamiliarity is reduced by their documentation.

To guarantee that the criteria stay flexible but fall inside a given scope, test cases will be designed with restricted scopes.

## References

*ERPNext manual: Table of contents*. Home. (n.d.). Retrieved March 6, 2023, from <https://docs.erpnext.com/docs/v13/user/manual/en>  
*Jenkins*. GitLab. (n.d.). Retrieved March 6, 2023, from <https://docs.gitlab.com/ee/integration/jenkins.html>  
Selenium. (n.d.). Retrieved March 6, 2023, from <https://www.selenium.dev/>  
*Robot framework*. Robot Framework. (n.d.). Retrieved March 6, 2023, from <https://robotframework.org/>  
Docs ERPNext. (n.d.). Retrieved March 6, 2023, from <https://docs.erpnext.com/>