Lenjeriidelux-pucioasa Website Project

— Test Plan —

Gosav Irina

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 18.09.2024 | v1.0 | Irina Gosav | draft plan |

**Table of content:**

1. **Introduction**
   1. Test Objectives
   2. Functionalities in scope
   3. Functionalities out of scope
2. **Test process**
   1. Test planning
   2. Test analysis
   3. Test design
   4. Test implementation
   5. Test execution
   6. Test closure
   7. Test monitoring and control
3. **Test deliverables**
   1. Test plan
   2. Test conditions
   3. Test cases
   4. Daily test summary reports
   5. Traceability matrix
   6. Test case results

3.7 Bugs report

# Introduction

The purpose of this test plan is to have a better view on how the [lenjeriidelux-pucioasa.ro](https://lenjeriidelux-pucioasa.ro/) website -“Huse” category was tested. In order to test the website we will include GUI testing and functional testing techniques to guarantee an end-to-end flow for processes such as purchasing the selected product from “Huse” category.

## Test Objectives

* The purpose of this project is to simulate a real experience by applying all the acknowledgment gained during the IT Factory journey.
* Confirming the functionality and usability of the lenjeriidelux-pucioasa.ro, specifically the “Huse” category.
* The objective is to ensure the client that they can choose the desired product by selecting the desired category, filtering it, choosing the desired number of products, adding them to the cart and adding vouchers if the case.
* Another goal is to identify and report any difficulties encountered during testing.

**1.2 Functionalities in scope**

* All the feature of Huse module which were defined in software requirement specifications need to be tested: functional testing and GUI testing. Successfully obtaining a product purchase by completing the product selection and to add them to the cart and add vouchers if the case.

**1.3 Functionalities out of scope**

* Non-functional testing encompasses performance, security, and compatibility testing.
* Payment steps are not tested.
* Testing on specific hardware configurations or devices.
* Testing functionalities that refer to other types of account options or products (except: “Huse”)
* Automated testing is not covered.

# Test process

**2.1 Test planning:**

**Roles and responsibilities:**

|  |  |
| --- | --- |
| **Software Developer** | Nicu Apetrei |
| **Product Owner** | Sandra Alexei |
| **Project Manager** | Delia Pruteanu |
| **QA Engineer** | Irina Gosav |
| **Senior QA Engineer** | Radu Molocea |

**Entry Criteria:**

* Business documentation should be available.
* Functional specifications are defined
* Roles and responsibilities have been established.
* The test plan has been created.
* Initial project risks were identified and defined.

**Exit Criteria:**

* All tests from the plan have been completed.
* Number of unresolved bugs is insignificant or they have low priority
* All bugs were identified, documented, and resolved if was possible.
* Created and reviewed test reports and documentation.
* Business documentation was covered and met.
* The established deadline was reached.

**Risks:**

**Project risks:**

* **Resource Constraints**: Insufficient skilled personnel, such as developers, testers, or subject matter experts, could hinder project progress and quality.
* **Budget Overruns**: Exceeding the allocated budget due to incorrect cost estimations, unforeseen expenses, or insufficient resources allocation.
* **Communication Challenges**: Inadequate communication and coordination among project stakeholders, including testers, the development team, and company representatives, may result in misunderstandings, misalignment, and decision-making delays.
* **Scope limitation**: Frequent changes in project scope without thorough assessment and impact analysis might disrupt project schedules, resource allocation, and deliverables.

**Product risks:**

* **Incomplete or Inaccurate Content**: Presence of inaccuracies, incomplete details, or outdated content on the website could adversely affect the overall user experience.
* **Security Vulnerability**: Existence of vulnerabilities within the application might expose user data, compromising the platform’s security and trust.
* **Performance Challenges**: Possibility of delays in processing user actions, slow page loading times, or inadequate performance during periods of high traffic.
* **Compatibility Issues**: Potential compatibility conflicts with major web browsers, operating systems, or devices may cause usability challenges for certain users.
* **Functional Deficiencies**: Risk that implemented functionalities may not fully meet the expected requirements, potentially resulting in user dissatisfaction or negative experiences.

**2.2 Test Analysis:**

* Evaluating the business requirements and acceptance standards relevant to the functionality under consideration.
* Identifying the testing method as functional testing of the GUI, which evaluates the functionality and behavior of the application.
* Test categories include functional testing: positive testing and negative testing.
* Techniques for GUI testing encompass: verifies that all the functionalities of the software work according to the specified requirements, preventing incorrect or invalid data entry, and validating acceptance of minimum and maximum values.
* Test Conditions for "Huse" Category:

1. Ensuring correct main interface display for a smooth user experience.
2. Validation of “Huse Menu” button - It is essential to ensure that the required category can be selected and the user can view all the available products from the selected category.
3. Functionality of the “Filtreaza” button from Huse module - It is essential to ensure that the desired product can be selected by the client by choosing the price.
4. Functionality of the “Sorteaza dupa ” button from Huse module - It is essential to ensure that the desired product can be selected by the client by choosing the price, the newest articles, the best sellers’ articles or the average product sales.
5. Functionality of the “Adauga in cos” button” from Huse module products - It is essential to ensure that the client can add the desired product to the cart.
6. Functionality of the “Continua cumparaturile” button from cart - It is essential to ensure that the client can add other products to the cart.
7. Validation of Required Field Marking of the quantity selection from cart - It is essential to ensure that the client can add as more products of the same type to the cart but not more than 100 pieces.
8. Functionality of the “Actualizeaza cosul” button from cart - It is essential to ensure that the button works correctly and performs the intended functionality and the total cost will be updated accordingly.
9. Recording of ”Aplica cupon” button from cart - This condition checks if the code was successfully registered when the user enters a correct code in the field. The code should not be accepted if is not one of the codes: “Husa2024” ( discount 20%), “HusaIepuras” ( discount 10%); “Husa5%”( discount 5%).
10. Non-recording of ”Aplica cupon” button from cart- When the user enters incorrect or invalid data in the field, we will check that an error message will be received. The code should not be accepted if is not one of the codes: “Husa2024” ( discount 20%), “HusaIepuras” ( discount 10%); “Husa5%”( discount 5%). The codes from the previous year such as “Husa2023” and “HusaCraciun” should not be accepted.
11. Functionality of the “Plaseaza comanda” button from cart - It is essential to ensure that the button works correctly and performs the intended functionality and the client can start placing the order.

**2.3. Test Design**

* The Zephyr Squad tool aims to streamline the creation of functional test cases. Techniques such as positive testing, negative testing, and use case testing will be utilized for crafting test cases through specification analysis.
* Test scenarios and cases will align with identified acceptance criteria.
* Each test case will establish the necessary data and expected outcomes.
* Prioritization of test cases will be based on risk and significance.

**2.4. Test Implementation**

* Ensure the test environment is operational at  [lenjeriidelux-pucioasa.ro](https://teodent.ro/lenjeriidelux-pucioasa.ro).
* Create a cycle summary titled: GUI Testing” containing the designed test cases.
* Ensure the application is running and ready for testing.
* Configure required test data and environment.
* Prepare testing tools and resources for utilization.
* Organize the necessary test data required for executing test cases.

**2.5. Test Execution**

* Execute test cases according to the designated test cycle summary.
* Identify errors based on performed tests and compile comprehensive bug reports.
* Execute test cases as per the test plan schedule.
* Compare actual and expected results.
* Document any encountered problems or issues during testing.
* Perform additional tests to validate defect resolutions.

**2.6. Test Closure**

* Test closure reports will encompass test activities, outcomes and recommendations.
* Evaluate exit criteria.
* Generate test completion reports and distribute them to stakeholders.
* Address any remaining defect and archive testing materials.

**2.7. Test Monitoring and Control**

* Regular tests status reports will be generated weekly to reflect testing progress and identify new project risk for mitigation.

# Test Deliverables

|  |  |  |  |
| --- | --- | --- | --- |
| **Crt.** | **Deliverable Name** | **Author** | **Reviewer** |
| 1 | Test Plan | Test Lead – Radu Molocea | Project Manager / Business Analyst – Ioana Dulgheru/ Ion Craiu |
| 2 | Functional Test Cases | Test Team – Irina Gosav | Business Analyst – Ion Craiu |
| 3 | Daily/weekly status report | Test Team/Test Lead – Irina Gosav / Radu Molocea | Project Manager – Ioana Dulgheru |
| 4 | Test Closure report | Test Lead -– Radu Molocea | Project Manager – Ioana Dulgheru |
| 5 | Traceability matrix | Test Team – Irina Gosav | Test Lead - Radu Molocea |

**Fail Criteria**

Defects found during the Testing will be categorized according to the bug-reporting tool “JIRA” and the categories are:

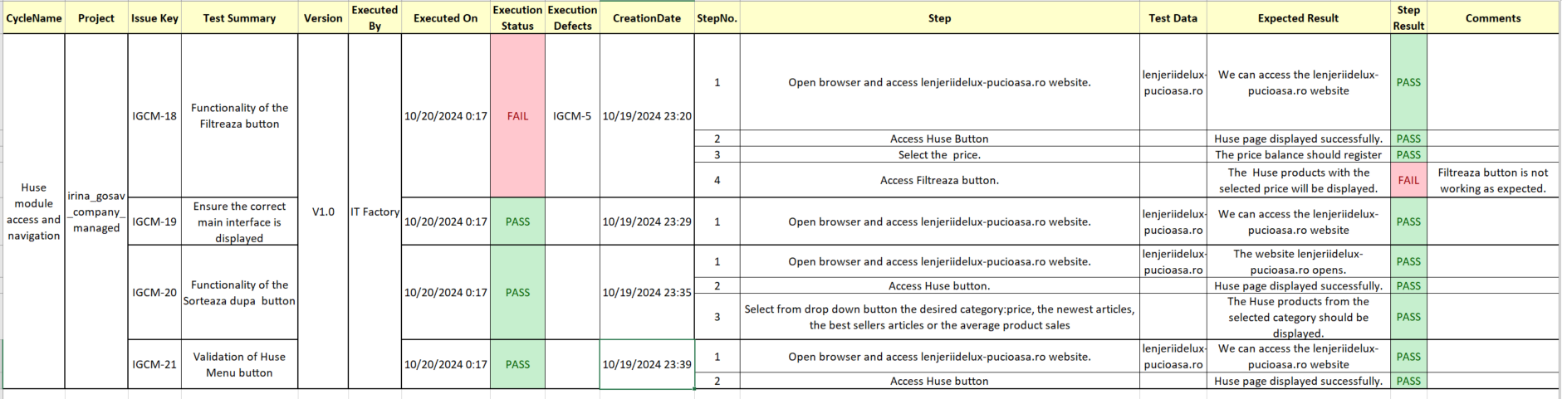
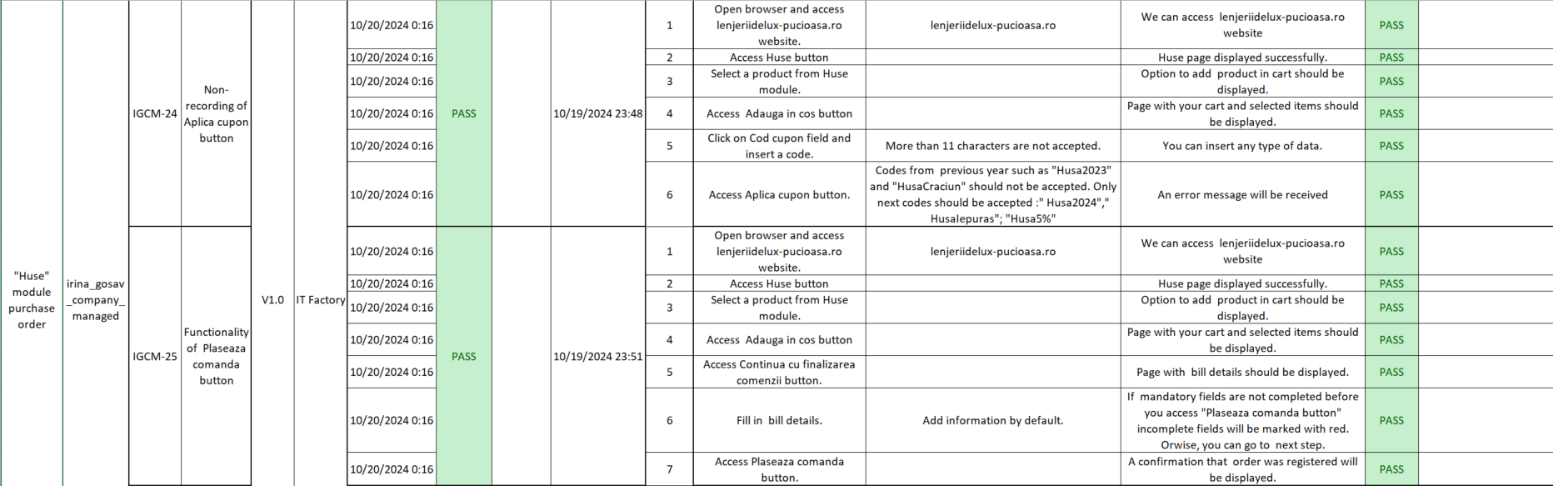
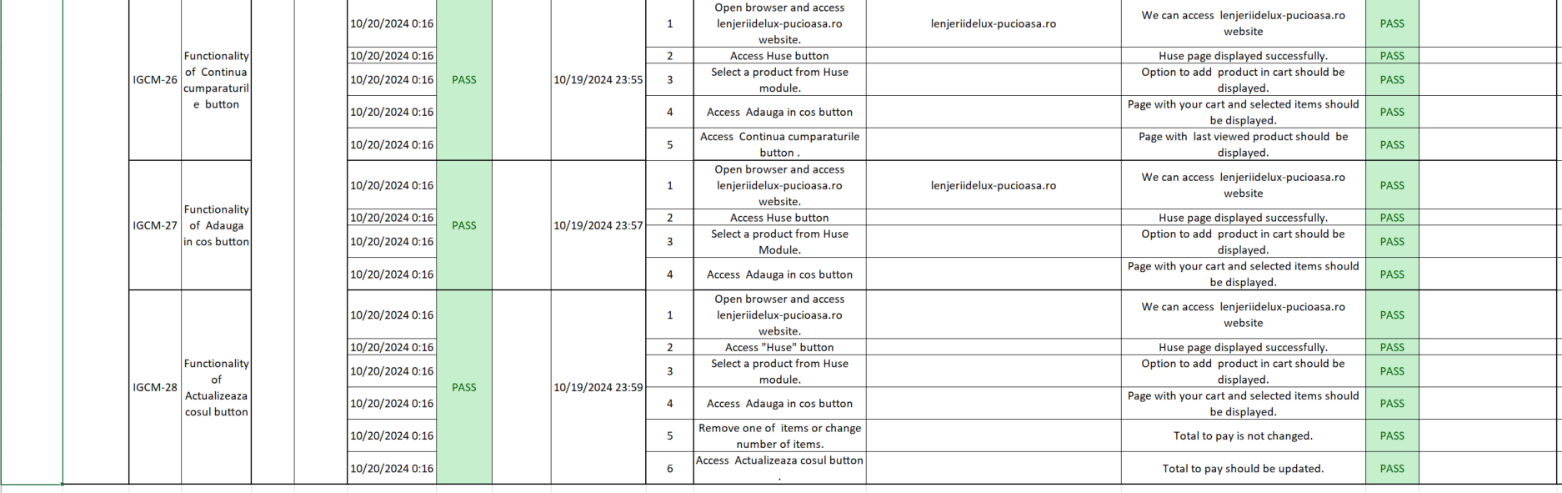
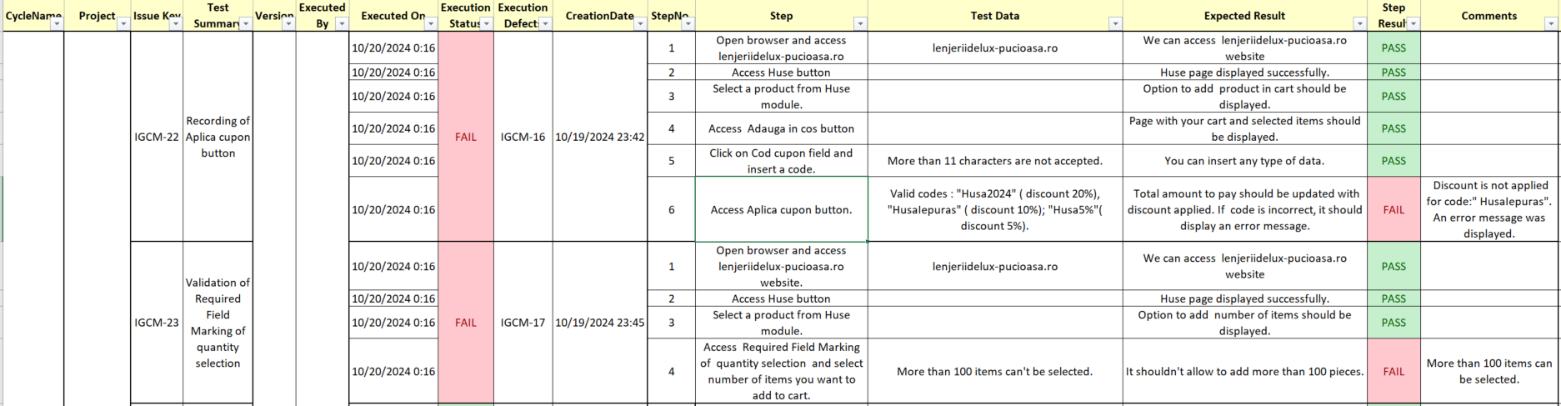
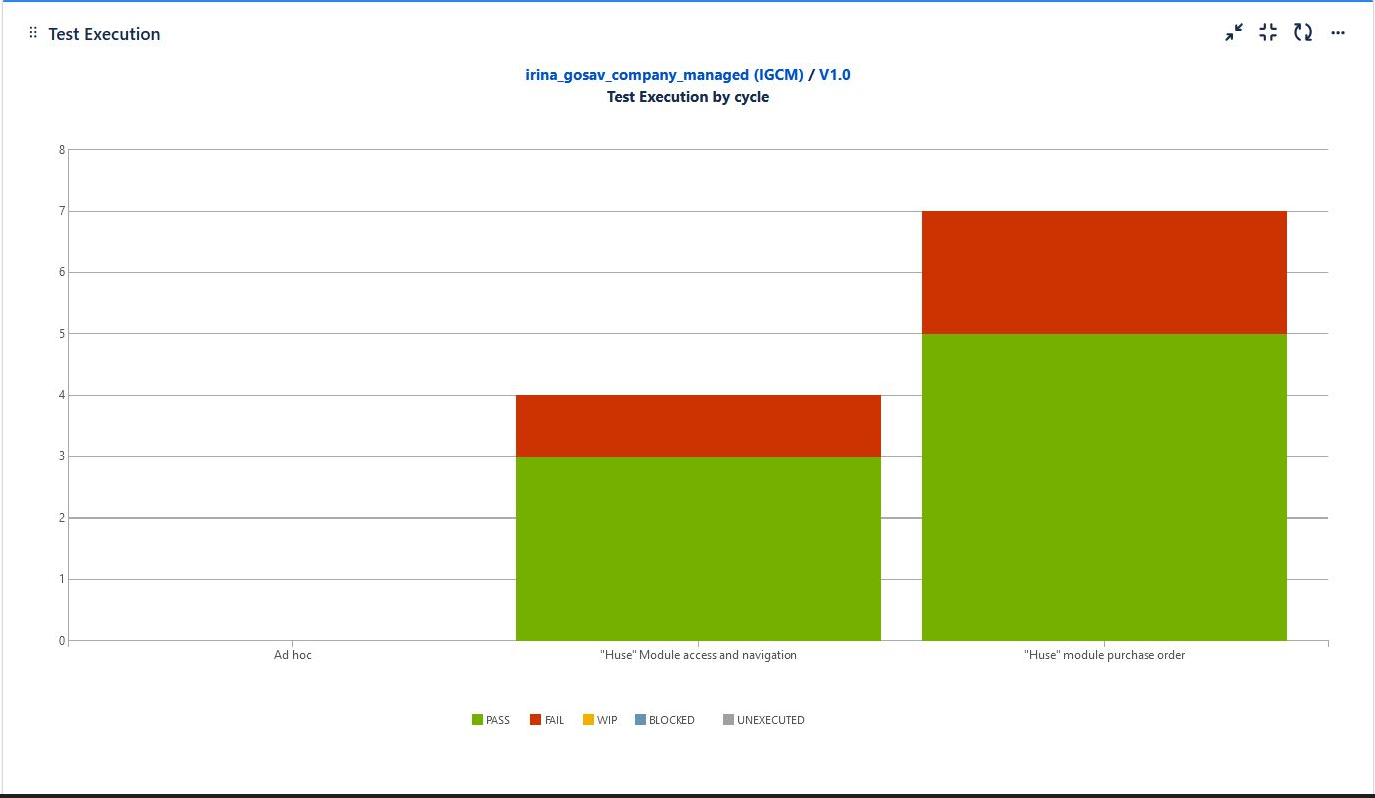
|  |  |
| --- | --- |
| **Severity** | **Impact** |
| 1 (Highest) | * This problem will block progress |
| 2 (High) | * Serious problem that could block progress. |
| 3 (Medium) | * Has the potential to affect progress. |
| 4 (Low) | * Minor problem or easily worked around. |
| 5 (Lowest) | * Trivial problem with little or no impact on progress. |
|

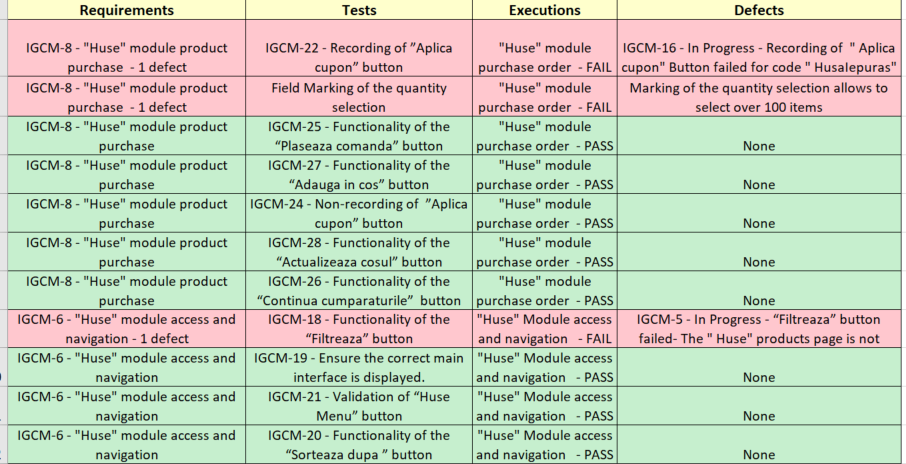
* 1. [**Test Plan**](https://github.com/Andriesei-Diana/Practical-Manual-Testing-Project/blob/main/TestingDocumentation/TestPlan)

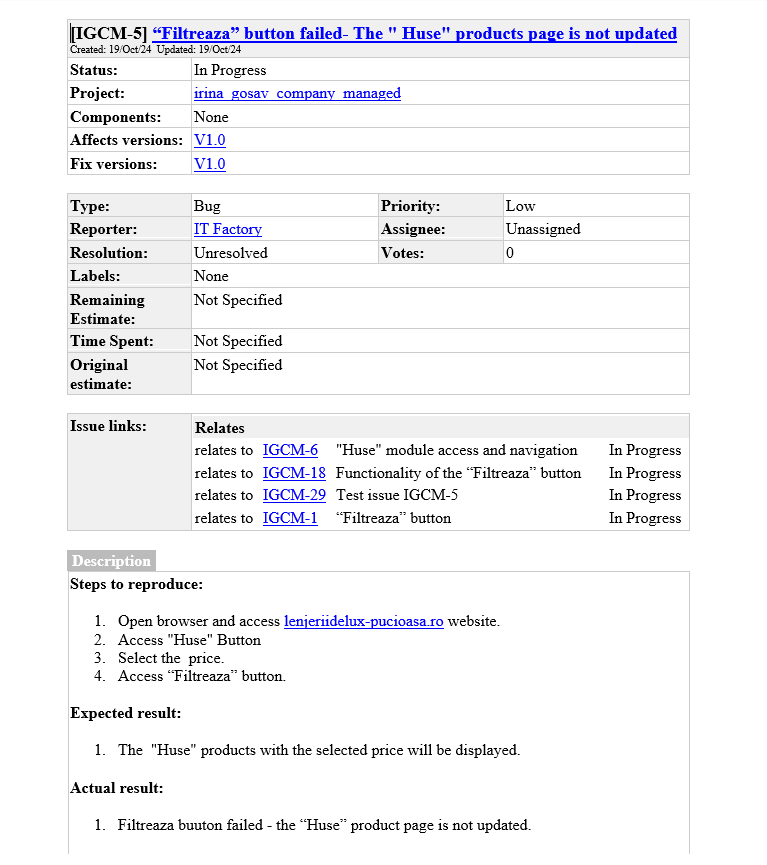
Based on the above requirements, for the Huse modu le, the testing process will be executed based on the following test conditions:

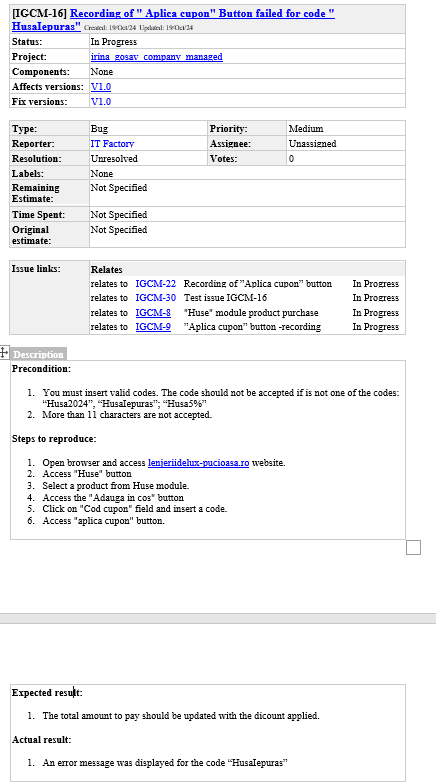
* Ensure the correct main interface is displayed.
* Validate the “Huse Menu” button and ensure the Huse module is created/updated.
* Make sure that the next buttons from Huse module are functional: “Filtreaza”, “Sorteaza dupa” and “Adauga in cos”.
* Enter data for the required field to choose the desired quantity of the product and validate if it is correctly registered.
* Make sure that the next buttons from Huse module are functional: “Continua cumparaturile”, “Actualizeaza cosul”, “Cod cupon”, “Aplica cupon” and “Plaseaza comanda”.
  1. **Test Conditions**

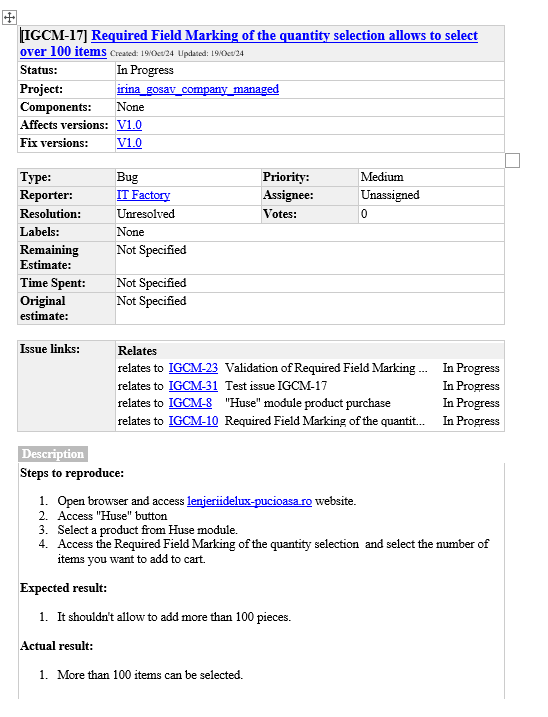


* 1. [**Test Cases**](https://github.com/Andriesei-Diana/Practical-Manual-Testing-Project/blob/main/TestingDocumentation/CycleSummaryTeodent)
  2. **Daily Test Summary Reports**

**3.5** [**Traceability Matrix**](https://github.com/Andriesei-Diana/Practical-Manual-Testing-Project/blob/main/TestingDocumentation/MatriceaTrasabilitatii)

**3.7** [**Bug Reports**](https://github.com/Andriesei-Diana/Practical-Manual-Testing-Project/blob/main/TestingDocumentation/RaportDeBug-uri)





Based on Jira tool, a test execution report focused on the "GUI Testing" cycle was generated. From 2 stories covered with 11 performed tests, 3 tests failed. For the failed tests we identified 1 test with low severity and 2 tests with medium severity. We recommend a careful and focused approach to these failures, including the following aspects: identifying the underlying causes for each failure, determining the impact of each failure on the user experience, and taking the necessary measures for the prompt correction of the identified problems. Our goal is to provide an improved experience in order to avoid users from abandoning the order.

**3.8 Test completion report**

As the Exit criteria were met and satisfied as mentioned in the appropriate section, this feature is suggested to ‘Go Live’ by the Testing team.

The traceability matrix was generated and can be found here: Traceability\_matrix.csv

Test execution chart was generated, the final report shows that a number 3 tests have failed of a total of 11.

A number of 11 test cases were planned for execution and all of them were executed.

A number of 3 total bugs were found, from which the priority is: 2 are medium and 1 is low.