GTL Test Plan

**Project Scope**

The project we will be working on is a software development project which will implement a library loaning system meant to replace GTL’s already existing analog loaning system. The system will be used internally, and our testing will be focused on the Business Layer. The main part is focused on the loaning of books and films to its members. The project objectives we are concerned with are keeping track of the loans, the members and the due dates.

**Test Objectives**

1. Verify that a loan is registered in the system
2. Verify that system sends notification on due date depending on the member type
3. Verify that system does not allow a member to borrow more than a maximum of five books simultaneously
4. Verify that a loan’s status is updated when member returns the book
5. Verify that system does not loan out books that can’t be borrowed
6. Verify that system does not allow members that own cards older than four years to borrow books

**Features and Functions to be tested**

ASK

**Testing approach**

TDD

The testing will be the first to start with; the test owners will be the developers; the developers will have full responsibility for the tests; our testing approach will be analytical/consultative (preventive). The testing process will start with unit tests, then integration tests, then go into system test and lastly acceptance tests. The developers will be the testers.

Unit tests because TDD.

Integration test to make sure components work together as they should.

System tests to make sure that system fulfills functional and non-functional requirements.

Acceptance tests: black box to test the system as it should be used.

ASK IF THIS IS OK

**Testing process and procedures**

Start with Unit tests to test the components by themselves, making sure that they are working as intended and to catch errors early on. Integration test is to make sure the components work together and to monitor how they interact with each other. System tests to make sure that system fulfills functional and non-functional requirements. Acceptance tests make sure that the system is up to the user’s requirements and works as intended from the user’s perspective.

**Test compliance**

|  |  |  |
| --- | --- | --- |
| Test objective | **Entry Criteria** | **Exit Criteria** |
| Test objective 1 | Member, valid card, available copy of book | Database is updated and loan is registered with the correct information |
| Test objective 2 | Member, overdue loaned copy of book | Notify librarian book is overdue |
| Test objective 3 | Member, valid card, five loaned copies of books | System denies copy of book loan |
| Test objective 4 | Member, loaned copy of book | Database is updated with the correct information |
| Test objective 5 | Member, valid card, not loanable copy of book | System denies copy of book loan |
| Test objective 6 | Member, invalid card, available copy of book | System denies copy of book loan |

**Testing tools**

To be added later

~~Defect resolution~~

**Roles and responsibilities**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Role | Responsibilities | Time |
| Andreas | Developer/Tester | Creating, Using, Refactoring and Approving Tests | 60% |
| Madalina | Developer/Tester | Creating, Using, Refactoring and Approving Tests | 60% |

**Deliverables**

Tests docs and tests.

**Schedule**

ASK

**Environmental needs**

Laptop, Office Package More to be added soon

**Resource management**

|  |  |
| --- | --- |
| Name | Skills |
| Andreas | Low knowledge of testing |
| Madalina | Low knowledge of testing |

**Risk and contingencies**

Not enough time.

Lack of testing experience.

Lack of knowledge.

**Approvals and workflow**

ASK