FRM

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 25/04/2017 | 1.0 | Creation and filling with basic information | Karl Spickermann |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 5

1.1 Purpose 5

1.2 Scope 5

1.3 Intended Audience 5

1.4 Document Terminology and Acronyms 5

1.5 References 5

2. Evaluation Mission and Test Motivation 5

2.1 Background 5

2.2 Evaluation Mission 6

2.3 Test Motivators 6

3. Target Test Items 6

4. Outline of Planned Tests 6

4.1 Outline of Test Inclusions 6

4.2 Outline of other candidates for potential inclusion 6

4.3 Outline of Test Exclusions 6

5. Test Approach 7

5.1 Initial Test-Idea Catalogs and other reference sources 7

5.2 Testing Techniques and Types 7

5.2.1 Data and Database Integrity Testing 7

5.2.2 Function Testing 8

5.2.3 Business Cycle Testing 10

5.2.4 User Interface Testing 11

5.2.5 Performance Profiling 11

5.2.6 Load Testing 13

5.2.7 Stress Testing 14

5.2.8 Volume Testing 16

5.2.9 Security and Access Control Testing 17

5.2.10 Failover and Recovery Testing 18

5.2.11 Configuration Testing 20

5.2.12 Installation Testing 21

6. Entry and Exit Criteria 22

6.1 Test Plan 22

6.1.1 Test Plan Entry Criteria 22

6.1.2 Test Plan Exit Criteria 22

6.1.3 Suspension and resumption criteria 22

6.2 Test Cycles 22

6.2.1 Test Cycle Entry Criteria 22

6.2.2 Test Cycle Exit Criteria 22

6.2.3 Test Cycle abnormal termination 22

7. Deliverables 22

7.1 Test Evaluation Summaries 22

7.2 Reporting on Test Coverage 22

7.3 Perceived Quality Reports 22

7.4 Incident Logs and Change Requests 23

7.5 Smoke Test Suite and supporting Test Scripts 23

7.6 Additional work products 23

7.6.1 Detailed Test Results 23

7.6.2 Additional automated functional Test Scripts 23

7.6.3 Test Guidelines 23

7.6.4 Traceability Matrices 23

8. Testing Workflow 23

9. Environmental Needs 24

9.1 Base System Hardware 24

9.2 Base Software Elements in the Test Environment 24

9.3 Productivity and Support Tools 25

9.4 Test Environment Configurations 25

10. Responsibilities, Staffing and Training Needs 25

10.1 People and Roles 25

10.2 Staffing and Training Needs 27

11. Iteration Milestones 28

12. Risks, Dependencies, Assumptions and Constraints 28

13. Management Process and Procedures 29

13.1 Measuring and Assessing the Extent of Testing 29

13.2 Assessing the deliverables of this Test Plan 30

13.3 Problem Reporting, Escalation and Issue Resolution 30

13.4 Managing Test Cycles 30

13.5 Traceability Strategies 30

13.6 Approval and Signoff 30

# Introduction

## Purpose

The purpose of the Iteration Test Plan is to gather all of the information necessary to plan and control the test effort for a given iteration. It describes the approach to testing the software, and is the top-level plan generated and used by managers to direct the test effort.

This *Test Plan* for the supports the following objectives:

* Controller
* Model
* View

## Scope

Integration Testing with Travis CI and PHPUnit

* Travis CI for managing the testing by triggering builds and tests
* PHPUnit for coding the actual tests

Unit tests

* Will test the internal application logic.

Testing with end user

* Will test the user interface if it is easy to learn.

## Intended Audience

● Students

● Professors

● Programmer

## Document Terminology and Acronyms

n/a

## References

[This subsection provides a list of the documents referenced elsewhere within the **Test Plan**. Identify each document by title, version (or report number if applicable), date, and publishing organization or original author. Avoid listing documents that are influential but not directly referenced. Specify the sources from which the “official versions” of the references can be obtained, such as intranet UNC names or document reference codes. This information may be provided by reference to an appendix or to another document.]

# Evaluation Mission and Test Motivation

Testing is done to guarantee that the software is stable and furthermore stays stable over the development of new features and bug fixes.

[Provide an overview of the mission and motivation for the testing that will be conducted in this iteration.]

## Background

By testing our project, we can monitor the effects that changes to the source code and user interactions cause to the functionality and performance of the software.

As a result we can:

1. Ensure that what we create does what it’s supposed to do.

Testing guarantees that new functionalities work as intended and detects possible conflicts between the new and old functionalities. As an example a new feature could break an old legacy feature by testing we can prevent this from happening and safe our users from the trouble of dysfunctioning core services.

1. Catch all possible edge cases.

“No user would ever do that.” This sentence creates edge cases. No developer can ever think of all possible combination of user interactions possible in his system to still catch all possible bugs ,hidden in bizarre action combination, excessive testing is needed.

## Evaluation Mission

Testing is done to provide a stable software. And we will fulfill the goal by the following points.

* find as many bugs as possible
* find important problems
* certify to a standard
* verify a specification (requirements, design or claims)

## Test Motivators

* Reduce quality and technical risks.
* Functional and no-functional requirements
* Design elements
* realize use cases faster by providing stability

# Target Test Items

The listing below identifies those test items⎯software, hardware, and supporting product elements ⎯that have been identified as targets for testing. This list represents what items will be tested.

* Controller (Logic)
* View (Design)
* Model (Database)
* Routing (Interaction of parts above)

# Outline of Planned Tests

## Outline of Test Inclusions

* Integration Testing with Travis CI
* Unit Testing with PHPUnit
* Testing with the end user

## Outline of Other Candidates for Potential Inclusion

Stress testing the application

## Outline of Test Exclusions

n/a

# Test Approach

* Testing with end user
* Integration Test
* Unit Test

## Initial Test-Idea Catalogs and Other Reference Sources

n/a

## Testing Techniques and Types

### Testing with end user

|  |  |
| --- | --- |
| Technique Objective: | Testing the simplicity of the app |
| Technique: | ● Testing the menu for simplicity  ● Testing the app for easy understanding  ● Users fill out a survey |
| Oracles: | The test users are happy with the app. The whole app is easy to understand,  it is self-explaining. The menu navigation is simple. |
| Required Tools: | A Device capable of navigating and interacting with the website. (Preferable a Laptop or Desktop Computer) |
| Success Criteria: | The user is happy. |
| Special Considerations: | - |

Integration Test

|  |  |
| --- | --- |
| Technique Objective: | Testing if the combination of units work well together. |
| Technique: | * Whenever a feature branch merges with the master branch Travis CI automatically triggers a build and runs Unit Tests |
| Oracles: | We assume, that all tests pass. |
| Required Tools: | CI tool – Travis CI |
| Success Criteria: | * 40% test coverage * all tests pass in deployment process |
| Special Considerations: | - |

Unit Test

|  |  |
| --- | --- |
| Technique Objective: | Testing the functionality of the code |
| Technique: | Testing the code of the testable classes. |
| Oracles: | We assume, that all tests pass. |
| Required Tools: | PHPUnit |
| Success Criteria: | All test pass |
| Special Considerations: | - |

# Entry and Exit Criteria

## Test Plan

### Test Plan Entry Criteria

This Test Plan can begin as soon as the development and build environment is set, all Use Cases are defined and the development has begun.

### Test Plan Exit Criteria

With the successful deployment of a FRM-System on another server including a test of all its features the product can be labelled as fully functional and testing can end.

### Suspension and Resumption Criteria

If the project gets cancelled testing would stop prematurely.

## Test Cycles

### Test Cycle Entry Criteria

n/a

### Test Cycle Exit Criteria

n/a

### Test Cycle Abnormal Termination

n/a

# Deliverables

## Test Evaluation Summaries

Travis runs the functional and unit tests on each push. It can either fail, pass or error.

## Reporting on Test Coverage

We use Coverall.io to publish our code coverage information: https://coveralls.io/github/d-wagner/frmsystem

## Perceived Quality Reports

With the help of phpmetrics we are able to present an overview of the quality of our codebase.

## Incident Logs and Change Requests

N/A

## Smoke Test Suite and Supporting Test Scripts

n/a

## Additional Work Products

n/a

### Detailed Test Results

n/a

### Additional Automated Functional Test Scripts

n/a

### Test Guidelines

n/a

### Traceability Matrices

n/a

# Testing Workflow

We use PHPUnit for testing as our build process supports testing, every push to the master branch causes the functional and unit tests to be run by Travis CI. All the test results are reported to Coveralls.io.  
If the build and all quality reports were successful the project is deployed to our server. If a build failed the creator of the build is notified to ensure that the issues are addressed immediately.  
We r perform end user tests when we think it is necessary.

# Environmental Needs

[This section presents the non-human resources required for the **Test Plan**.]

## Base System Hardware

The following table sets forth the system resources for the test effort presented in this *Test Plan*.

| **System Resources** | | |
| --- | --- | --- |
| **Resource** | **Quantity** | **Name and Type** |
| Database Server | 1 | SERVER4YOU, vServer SSD S8 |
| Test Environment | 1 | localhost, xampp |

## Base Software Elements in the Test Environment

The following base software elements are required in the test environment for this *Test Plan*.

| **Software Element Name** | **Version** | **Type and Other Notes** |
| --- | --- | --- |
| Ubuntu | 16.04 | Operating System |
| Apache | newest | Web Server |
| Chrome | newest | Internet Browser |
| PHP Storm | 7.0.6 | IDE |
| Node.js | 6.2.0 | Serversided Script |
| Travis CI |  | CI Enviroment |

## Productivity and Support Tools

The following tools will be employed to support the test process for this *Test Plan*.

| **Tool Category or Type** | **Tool Brand Name** | **Vendor or In-house** | **Version** |
| --- | --- | --- | --- |
| Metrics | phpmetrics | Open Source | newest |
| Testing | PHPUnit | Open Source | 5.3 |
| Test Coverage Monitor or Profiler | Coveralls.io | Coveralls | newest |
| Project Management | Youtrack | Jetbrains | newest |
| DBMS tools | MySQL | Open Source | newest |

## Test Environment Configurations

The following Test Environment Configurations needs to be provided and supported for this project.

| **Configuration Name** | **Description** | **Implemented in Physical Configuration** |
| --- | --- | --- |
| Average user configuration | Number of users accessing the application at the same time | 1 |
| Minimal configuration supported | Speed and power of the internet connection provided by the server host. | 100Mbit/s |
| Network installation (not client) | Performance of the application server and database server. | WebServer with 2GB Ram and Ubuntu 16.04 |

# Responsibilities, Staffing, and Training Needs

## People and Roles

This table shows the staffing assumptions for the test effort.

| **Human Resources** | | |
| --- | --- | --- |
| **Role** | **Minimum Resources Recommended**  **(number of full-time roles allocated)** | **Specific Responsibilities or Comments** |
| Test Manager | 1 | Provides management oversight.  Responsibilities include:   * planning and logistics * agree mission * identify motivators * acquire appropriate resources * present management reporting * advocate the interests of test * evaluate effectiveness of test effort |
| Test Analyst | 2 | Identifies and defines the specific tests to be conducted.  Responsibilities include:   * identify test ideas * define test details * determine test results * document change requests * evaluate product quality |
| Test Designer | 2 | Defines the technical approach to the implementation of the test effort.  Responsibilities include:   * define test approach * define test automation architecture * verify test techniques * define testability elements * structure test implementation |
| Tester | 3 | Implements and executes the tests.  Responsibilities include:   * implement tests and test suites * execute test suites * log results * analyze and recover from test failures * document incidents |
| Test System Administrator | 1 | Ensures test environment and assets are managed and maintained.  Responsibilities include:   * administer test management system * install and support access to, and recovery of, test environment configurations and test labs |
| Database Administrator, Database Manager | 1 | Ensures test data (database) environment and assets are managed and maintained.  Responsibilities include:   * support the administration of test data and test beds (database). |
| Designer | 1 | Identifies and defines the operations, attributes, and associations of the test classes.  Responsibilities include:   * defines the test classes required to support testability requirements as defined by the test team |
| Implementer | 3 | Implements and unit tests the test classes and test packages.  Responsibilities include:   * creates the test components required to support testability requirements as defined by the designer |

## Staffing and Training Needs

This section outlines how to approach staffing and training the test roles for the project.

n/a

# Iteration Milestones

| **Milestone** | **Planned Start Date** | **Actual Start Date** | **Planned End Date** | **Actual End Date** |
| --- | --- | --- | --- | --- |
| At least 20% Test Coverage | Project Start | Project Start | 30.6.2017 |  |
| Release Build Passes | Project Start | Project Start | 30.6.2017 |  |

# Risks, Dependencies, Assumptions, and Constraints

| **Risk** | **Mitigation Strategy** | **Contingency (Risk is realized)** |
| --- | --- | --- |
| Technical Problems | <Tester> needs to be informed and has to fix problems. | * Fix the problem |
| Test data proves to be inadequate. | <Customer> will ensure a full set of suitable and protected test data is available.  <Tester> will indicate what is required and will verify the suitability of test data. | * Redefine test data * Review Test Plan and modify * components (that is, scripts) * Consider Load Test Failure |
| Database requires refresh. | <System Admin> will endeavor to ensure the Database is regularly refreshed as required by <Tester>. | * Restore data and restart * Clear Database |

| **Dependency between** | **Potential Impact of Dependency** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

| **Assumption to be proven** | **Impact of Assumption being incorrect** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

| **Constraint on** | **Impact Constraint has on test effort** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

# Management Process and Procedures

n/a

## Measuring and Assessing the Extent of Testing

n/a

## Assessing the Deliverables of this Test Plan

n/a

## Problem Reporting, Escalation, and Issue Resolution

n/a

## Managing Test Cycles

n/a

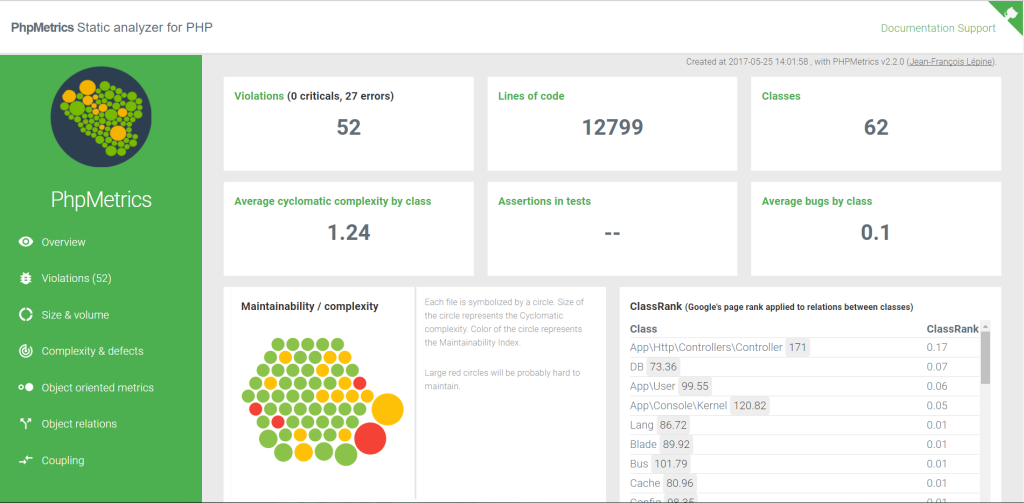
## Traceability Strategies

n/a

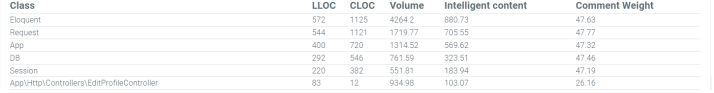
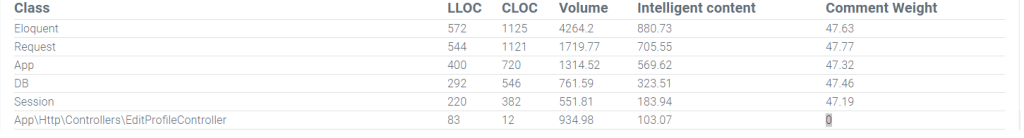
## Approval and Signoff

n/a

**Appendix – Metrics Reports**



**Appendix – Comment Weight**



**Appendix – Cyclomatic Complexity**

