Testing - Alpha Integration

Members

14011396 Grobler A (Arno)
14103207 Mogase L (Lethabo)
12017800 Maree AA (Armand)
12059138 Saaiman C (Christiaan)
13027205 Smallwood DW (Duncan)
13040686 Nyuswa ML (Maluleki)
14035538 Heins D (Dillon)
14395283 Alberts JD (Josef)
12031748 Khumalo S (Sandile)
13278012 Muranga KJ (Kudzai)
14222583 Molefe KP (Keletso)

[8cm]

Lorem
Ipsum
20 April 2016

Contents

1 Introduction

This project is a Research Support System. Its main purpose is to keep track of many aspects that are regularly needed by researchers and a easy way for research heads or HOD of the respective facility to be able to see progress that has been made by the researchers in their team.

Scope for this project is only the Computer Science department and its respective lecturers and leaders. General uses of this system should include keeping track of:

- Research Papers
- Reports, of which has different types
- Research Groups
- Running Costs
- Historical publications
- List of Authors
- List of Users
- Units

1.1 Testing

Software testing plays an important part of the software development pipeline and is necessary for creating maintainable and scalable programmes. It has many different objectives and goals but ultimately it comes down to:

- Finding defects in the program that was developed and possibly finding solutions to those problems
- To ensure the level of quality has been maintained throughout the program
- Find potential security flaws and vulnerabilities
- Ensuring usability
- Ensuring the program has lived up to the system requirements set out by the clients.

Testing ensures that the project, which should be in its final stages of development, is ready to be rolled out to the clients and that it meets the requests made by the clients. Thus certain fields of the program could be addressed. These fields are:

- Flexibility
- Maintainability
- Scalability
- Performance requirements
- Reliability
- Security
- Auditability
- Testability
- Usability
- Integrability
- Deployability

2 Functional Testing

2.1 Reporting

2.1.1 Implementation

2.1.2 Short-comings

2.1.3 Missing

2.2 Notifications

2.2.1 Implementation

Unit tests were performed for all the functionality within the notification module.Dependency injection was used through the spring framework and everything worked.J unit was used accordingly and all the test cases that were written passed.

2.2.2 Short-comings

There are no short comings, all functionality specified by the specification were implemented.

2.2.3 Missing

2.3 People

2.3.1 Implementation

2.3.2 Short-comings

2.3.3 Missing

2.4 Publications

2.4.1 Implementation

The integration functionality to get a publication works, it takes in a json object request and returns a response with is a json object.if the publication does not exist the is a catch an exception or if the request in not valid it will catch an exception. for that they use a rest layer. This is the only functionality implemented by the integration team for publication.

2.4.2 Short-comings

There were no short commings in the get pulbication function

2.4.3 Missing

There is no other functionality to modify a publication such as to change publication state ,add publication type, modify publication type and also to getPublicationForPerson,getPublicationForGroup and calculate Accreditation points for both group and person.

2.5 Integration/Interface

2.5.1 Implementation

2.5.2 Short-comings

2.5.3 Missing

3 Architectural Compliance

3.1 Reporting

3.1.1 Software Architecture Adhered

3.1.2 Software Architecture Partially Adhered

3.1.3 Software Architecture Not Adhered

3.2 Notifications

3.2.1 Software Architecture Adhered

3.2.2 Software Architecture Partially Adhered

3.2.3 Software Architecture Not Adhered

3.3 People

3.3.1 Software Architecture Adhered

3.3.2 Software Architecture Partially Adhered

3.3.3 Software Architecture Not Adhered

3.4 Publications

3.4.1 Software Architecture Adhered

3.4.2 Software Architecture Partially Adhered

3.4.3 Software Architecture Not Adhered

3.5 Integration/Interface

3.5.1 Software Architecture Adhered

3.5.2 Software Architecture Partially Adhered

3.5.3 Software Architecture Not Adhered

4 Publication Architecture Compliance

4.1 Flexibility

The integration code complies to this requirement because it has the ability to deploy different versions of the system with as little down-time as possible. This is partly due to the decoupled nature of the integration and publication sections. The code also makes use of dependency injection for contract based software development.

4.2 Maintainability

The code has very little to no documentation, which makes it very difficult for developers that did not on the system initially to be able to understand the code.

4.3 Scalability

The system is able to store a large amount of the department's publication meta data on the database without any difficulty.

4.4 Performance Requirements

All the the services of Publication operate within the required time of 0.2 seconds.

4.5 Auditability

The system does not log any of the requests, responses and exceptions of the publication services.

4.6 Reliability

The service requests are fully implemented and can be fully executed for each service contract for the publication section. It also supports commit and rollback functionality for the database.

4.7 Security

The integration code does not check the authorisation of the user for the services that only should be used by specific users. For example, the addPublicationType service should only allow administrators to use it. This is not implemented.

4.8 Testability

Junit is used for out-of-container testing to test the publication section. Embedded container unit testing is not used, however, this is listed as a nice-to-have.

5 Front-end Interfaces: Web and Android Development

Front end developers were tasked to use information from the back-end, such as database objects and requests and display it in a way that is user friendly to the user and intuitive for the designer. The interface also needs to provide the user easy tools to make requests to the back-end without the user having to worry about actual functionality of the system.

5.1 Web Development

5.1.1 Flexibility

5.1.2 Maintainability

5.1.3 Scalability

5.1.4 Performance requirements

5.1.5 Reliability

5.1.6 Security

5.1.7 Auditability

5.1.8 Testability

5.1.9 Usability

5.1.10 Integrability

5.1.11 Deployability

5.1.12 Correctly implemented functionality

5.1.13 Short-comings of the implemented functionality

5.1.14 Missing functionality

5.2 Android

5.2.1 Flexibility

5.2.2 Maintainability

5.2.3 Scalability

5.2.4 Performance requirements

5.2.5 Reliability

5.2.6 Security

5.2.7 Auditability

5.2.8 Testability

5.2.9 Usability

5.2.10 Integrability

5.2.11 Deployability

5.2.12 Correctly implemented functionality

5.2.13 Short-comings of the implemented functionality

5.2.14 Missing functionality

5.3 Alpha People

5.3.1 Functional Testing

Functional testing was partially implemented on the functionalities in the Person module. The test cases where coded and documented but did not cover all cases.

5.3.1.1 Functionality Implementation

5.3.1.2 Functionality Short-comings

5.3.1.3 Functionality Missing

5.3.2 Architecture Compliance

5.3.2.1 Software Architecture Adhered

5.3.2.2 Software Architecture Partially Adhered

5.3.2.3 Software Architecture Not Adhered

6 Body

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.