Kalafong Provincial Tertiary Hospital

Gynaecological Patient Information Management System:

System Tests

Team Pentec:

Ruth Ojo 12042804
Liz Joseph 10075268
Trevor Austin 11310856
Maria Qumayo 29461775
Lindelo Mapumulo 12002862

Pentec

Final Version

August 28, 2015

Contents

1 Introduction

This document documents and taracks the necessary information required to effectively define the approach to be used in the testing and evaluationg of the Patient Information Management System designed by the group Pentect for the Kalafong

The following Patient Infromation Management System Use cases that were thoroughly tested are:

- User Login
- PIMS Artificial Intelligence
- PIMS Statistics
- PIMS Notifications
- PIMS Space

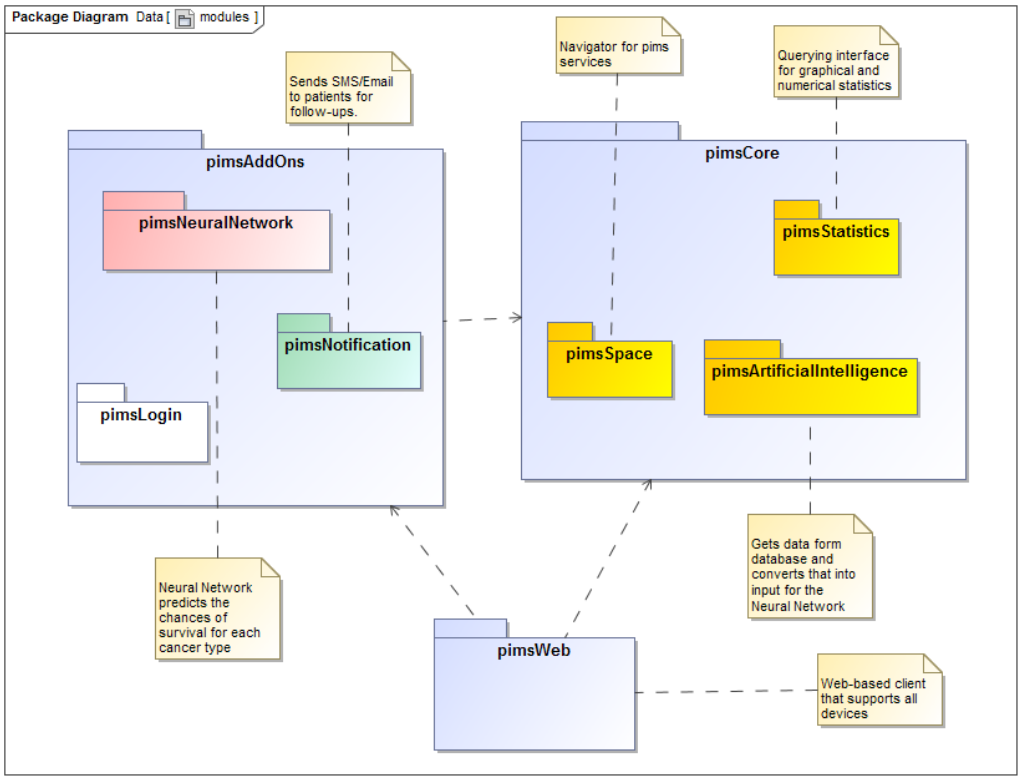
Testing was done on the system mainly for the following 5 reasons:

- To ensure that the system meets both functional and non functional requirements arroding to the given spesifications
- System stress, that is to make sure that the system does not fail with multiple users or any other factors because it is expensive to resolve and fix at a later stage.
- To handle and resolve System failures and bugs appropriatly and in good time.
- To point out the defects and errors that were made during and after the development phases.
- To ensure that the final product is of top, professiona, software engineering standards.

2 Features and Items tested

2.1 Features and Items tested

Our testing involved looking at the core functional requirements as given in our client's spesification document. As well as those we added on as "Nice to have". The following use cases and features as depicted in the PIMS master Node Scope were tested.



Functional Testing - for each use case tested * either success or a list of violations of the contract requirements * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

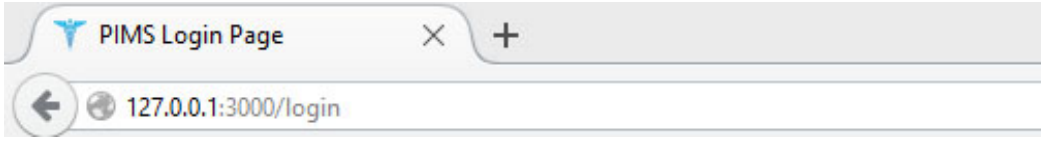
2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability,security, maintainability problems identified with evidence for the identified problem.

3 Functional Testing

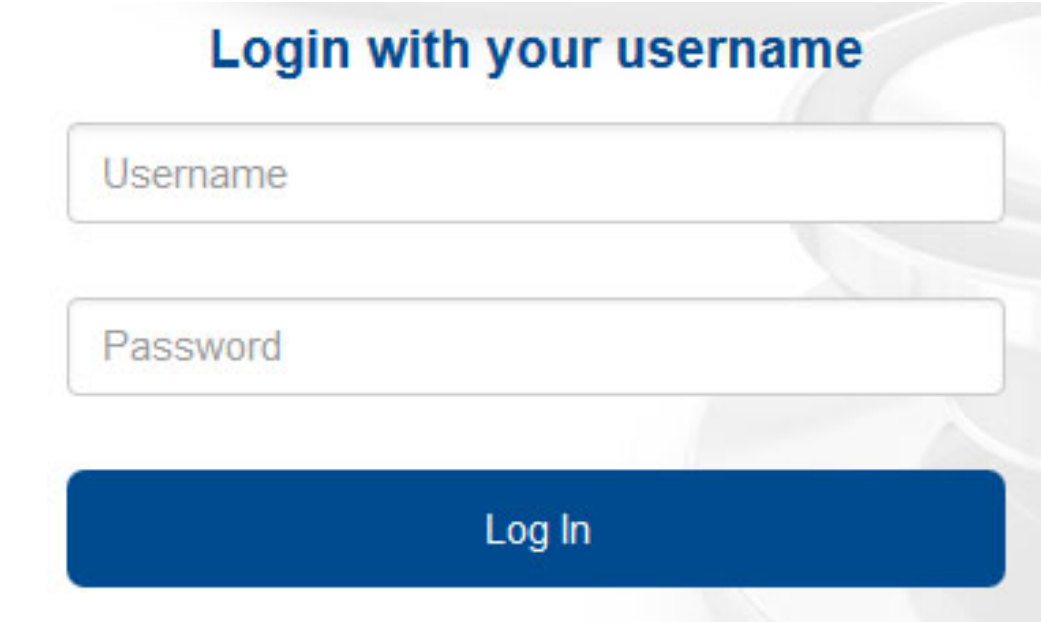
3.1 PIMS Login

3.1.1 Pims Login

Pims User login testes for correct authentication and identification before the user is allowed system access. Further testing was done for user rights and privilages.



Front end representation



User Authentication tested for the following conditions

- Provide user with access
- Retrieve username
- Retrieve user password
- Fail with empty username and/ or empty password
- Return a boolean with regards to user right.

The figure bellow depics the successful testing of the Authentication and checkAdmin functions.

```
login user
  √ authenticate should login user
  √ authenticate should retrieve username
  √ authenticate should retrieve password
  √ authenticate should fail with empty username
  √ authenticate should fail with empty password
  √ authenticate should fail with empty username and empty password
  √ authenticate should return a boolean
  √ checkAdmin should return a boolean
```

3.1.2 Remarks

- Pre-Conditions User does not have access to systme.
- Post-Conditions With correct login details, user successfuly gains access into the system with. User rights are checked upon login authentication

Both Pre and Post conditions are considered in the implimentation of the system login. Unit testing successfuly. tested with no violations to the security of the systm.

3.2 PIMS Notifications

3.2.1 Pims Login

Pims Send Notification is a two part function that we tested. First check if user is found(exists) in the database. Then send email using smtp.The unit test codes bellow demonstrates.

```
it("findPatient should return email address", function(done){
    notification.findPatient("sue", function(found){
        found.should.equal("nodemailingtest@gmail.com");
    });
    done();
});
```

Send Notifications via email

```
describe("send notification to patient", function(){
   it("findPatient should retrieve email address", function(done){
       notification.findPatient("sue", function(err){
           notification.Patient.findOne({patient_name: "sue"},
           function(err, found){
                found.email_address.should.equal("nodemailingtest@gm
                ail.com");
           });
```

The following conditions must be true for the send notification use case to pass.

- Find patient should query the database to see if a user exists by searching for an email adress.
- Oce user is found, send notification must send and email to the found adress.

The figure bellow depics the successful testing of the Send Notification and FindUser functions.

```
send notification to patient
  1) findPatient should retrieve email address
  2) findPatient should return email address
  √ should pass
  3) should pass
```

3.2.2 Remarks

- Pre-Conditions User must exist in database and have and email.
- Post-Conditions User recieves and email from Prof Snyman.

Both Pre and Post conditions are considered in the implimentation of sending notifications. Unit testing successfuly. tested with no violations to the security of the systm.

3.3 PIMS Edit Profile

3.3.1 Pims Login

Pims edit user profile should be able to allow the admin user to update his profile and edit his information accordingly.

The Code bellow demonstrates the update of the user name after retrienving it.

```
describe("update profile", function(){
    it("should retrieve username", function(done){
        User.findOne({username: "Leon"}, function(err, contact) {
            should.not.exist(err);
            contact.username.should.equal("Leon");
        });
        done();
    });

    it("should modify profile details [surname]", function(done){
        User.findOne({username: "Leon"}, function(err, contact) {
            contact.surname = "Snymanss"
            should.not.exist(err);
            contact.surname.should.equal("Snymanss");
```

Edit profile was tested for the following conditions

- Retrieve data
- Modify profile details

The figure bellow depics the successful testing of the UpdateAuthentication and checkAdmin functions.

```
update profile
  √ should retrieve username
  √ should modify profile details [surname]
  √ should modify profile details [password]
  √ should modify profile details [email]
  √ should modify profile details [user_rights]
```

3.4 PIMS Add User

3.5 PIMS Artificial Inteligence

3.6 PIMS Statistics

3.7 PIMS Predictions

4 Non-functional Testing

4.1 Usability

4.2 Scalability

4.3 Performance

4.4 Maintainability

4.5 Reliability

4.6 Secutity

4.7 Monitorability

5 Remarks

5.1 Risks and issues

5.2 Product quality

5.3 Possible improvements

6 Conclusion