Test planning

Version: 1.4
Date: 2012-05-17
Authors: Oscar Brodefors, Filip Askviken, Rikard Andersson, Emil Nyström

This version overrides all previous versions.

1. User tests

In order to test all requirements for the application, it is suitable to formulate two test cases for each requirement, one positive test and one negative test.

Below is a list of user tests. A P in the test ID means that it is a positive test and a N is connected to a negative test. Test ID are connected to Requirements ID. The tests are written with the following syntax:

Description:
Precondition:
Test steps:
Postcondition:
Related requirements:

Test ID 0.1-P **Startable application**
Description: The application is startable on the phone.
Precondition: The tester have the application installed.
Test steps:
1. The tester presses the application in the phone menu
2. The tester verifies that the application starts without any exception messages.
Postcondition: The application is runnable
Related requirements: NF0

Test ID 1.1-P **Book registration**
Description: A seller can register a book for sale
Precondition: Test ID 0.1. The seller have a book to sell, have the application and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title: Robinson Crusoe
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price: 300 SEK
Comments: excellent quality

3. The tester presses “add” and the book is added to the market.

Postcondition: A book is uploaded to the market.
Related requirements: 1.1

Test ID 1.1-N **Book registration without required information**
Description: A seller can register a book for sale
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title:
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price:
Comments: excellent quality

3. The tester presses “add”.

Postcondition: The tester gets a error message saying “missing title and price”.
Related requirements: 1.1

Test ID 1.1.1.1-P **Pricing a book - 99 SEK**
Description: The seller can set a price for the book
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text box “price”: 99 SEK
3. The tester presses “add”.

Postcondition: A book is uploaded to the market with the correct price.
Related requirements: 1.1.1

Test ID 1.1.1.2-P **Pricing a book - 300 SEK**
Description: The seller can set a price for the book
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text box "price": 300 SEK
3. The tester presses "add".

Postcondition: A book is uploaded to the market with the correct price.
Related requirements: 1.1.1

Test ID 1.1.1.3-P **Pricing a book - 1000 SEK**
Description: The seller can set a price for the book
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text box "price": 1000 SEK
3. The tester presses "add".

Postcondition: A book is uploaded to the market with the correct price.
Related requirements: 1.1.1

Test ID 1.1.1.4-P **Pricing a book - 0 SEK**
Description: The seller can set a price for the book
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text box "price": 0 SEK
3. The tester presses "add".

Postcondition: A book is uploaded to the market with the correct price.
Related requirements: 1.1.1

Test ID 1.1.1.1-N **Pricing a book - hundra kronor**
Description: The seller can set a price for the book

Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the textbox “price”: hundra kronor
3. The tester presses “add”.

Postcondition: The tester should get a error message stating: “Invalid price, please write a number between 0-10000 SEK.
Related requirements: 1.1.1

Test ID 1.1.1.2-N **Pricing a book - 133700 SEK**
Description: The seller can set a price for the book
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the textbox “price”: 133700 SEK
3. The tester presses “add”.

Postcondition: The tester should get a error message stating: “Invalid price, please write a number between 0-10000 SEK.
Related requirements: 1.1.1

Test ID 1.1.1.3-N **Pricing a book - no input**
Description: The seller can set a price for the book
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the textbox “price”: no input
3. The tester presses “add”.

Postcondition: The tester should get a error message stating: “Invalid price, please write a number between 0-10000 SEK.
Related requirements: 1.1.1

Test ID 1.1.2-P **Picture uploading**

Description: The seller can upload a picture of the book for sale

Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details, and wants to show a picture of the book.

Test steps:

1. The tester presses the button “add book”
2. The tester fills in the information details:

Author: Daniel Defoe
Book title: Robinson Crusoe
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price: 300 SEK
Comments: excellent quality

3. The user then presses the button “Photo” and takes a photo of a book.
4. The tester presses the button “add” and the book is added to the market.

Postcondition: A book in the market has a picture connected to it.

Related requirements: 1.1.2

Test ID 1.1.3-P **Title is required, >= four characters**

Description: The seller has to set a title for the book

Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.

Test steps:

1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:

Author: Daniel Defoe
Book title: Robinson Crusoe
Price: 300

3. The tester presses “add”.

Postcondition: Every added book have a title.

Related requirements: 1.1.3

Test ID 1.1.3-N **Title is required, < 4 characters**

Description: The seller has to set a title for the book
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title: xxyy
Price: 300

3. The tester presses "add".

Postcondition: A book cannot be added without a title and the tester gets a exception message "Title must be greater than 10 characters".
Related requirements: 1.1.3

Test ID 1.1.4-P **ISBN number**
Description: The seller can set an ISBN for the book for sale
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Jan Skansholm
Book title: Java direkt med Swing
ISBN: 9789144060743
Version:
Publishing year:
Price: 300

3. The tester presses "add".

Postcondition: A added book have a ISBN number
Related requirements: 1.1.4

Test ID 1.1.4-N **ISBN number**
Description: The seller can set an ISBN for the book for sale
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:

1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Jan Skansholm
Book title: Java direkt med Swing
ISBN: 123456
Version:
Price: 300

3. The tester presses “add”.

Postcondition: The tester gets a exception message: “The ISBN-number must be 10 or 13 digits long.”
Related requirements: 1.1.4

Test ID 1.1.5-P **Publishing year**
Description: The seller can set a publishing year for the book for sale.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title: testing the title
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price: 300

3. The tester presses “add”.

Postcondition: A added book have a publishing year
Related requirements: 1.1.5

Test ID 1.1.5-N **Publishing year**
Description: The seller can set a publishing year for the book for sale.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.

2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title: testing the title
ISBN:
Version: 2nd edition
Publishing year: 2015
Course: literature
Price: 300

3. The tester presses “add”.

Postcondition: The tester gets a exception message: “this is not a valid publishing year”.
Related requirements: 1.1.5

Test ID 1.1.6.1-P **Book connected to a course - course name**
Description: The seller can set a course for which he/she used the book.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
The tester presses the button for adding a new book.
The tester then fills in the following information details in the text boxes:
Author: Emil Nyström
Book title: Databases rule
Course: Databases
Price: 300

3. The tester presses “add”.

Postcondition: A book is added with a course connected to it.
Related requirements: 1.1.6

Test ID 1.1.6.2-P **Book connected to a course - course code**
Description: The seller can set a course for which he/she used the book.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.

2. The tester then fills in the following information details in the text boxes:
Author: Emil Nyström
Book title: Databases rule
Course: TDA357
Price: 300

3. The tester presses “add”.

Postcondition: A book is added with a course connected to it.
Related requirements: 1.1.6

Test ID 1.1.7-P **Comments**
Description: The seller can add a comment about the book.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Emil Nyström
Book title: Databases rule
Price: 300
Comments: excellent quality
3. The tester presses “add”.

Postcondition: A book is added with a comment connected to it.
Related requirements: 1.1.7

Test ID 1.1.8-N **Author**
Description: The seller can add a author to the book.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: My
Book title: Databases rule
Price: 300
Comments: excellent quality
3. The tester presses “add”.

Postcondition: The tester should get a exception message saying “a author must be five characters or more”

Related requirements: 1.1.8

Test ID 1.2-P **Taking a book off market**
Description: The seller can take a book off the market.
Precondition: Test ID 0.1, 1.1. The tester have a book advertisement.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title: Robinson Crusoe
Edition:
Publishing year:
Course: literature
Price: 300
Comments: excellent quality

3. The tester presses "add".
The book should then be uploaded to the market.
4. The tester tries to delete the ad by pressing the edit-button at the market and use his/hers password. In the ad-view, the user presses "delete" and deletes the ad.

Postcondition: The book should be deleted from the market.
Related requirements: 1.2

Test ID 1.2-N **Taking a book off market**
Description: The seller can take a book off the market.
Precondition: Test ID 0.1, 1.1. The tester have a book advertisement.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title: Robinson Crusoe
Edition:
Publishing year:
Course: literature
Price: 300
Comments: excellent quality

3. The tester presses "add".
The book should then be uploaded to the market.

4. The tester tries to delete the ad by pressing the edit-button at the market and use his/hers password. The tester fills in a wrong password.

Postcondition: The book is not editable and should be deleted from the market within two months.
Related requirements: 1.2

Test ID 1.3-P **Updating information about a book**
Description: The seller can update information about a book.
Precondition: Test ID 0.1, 1.1. The tester have a book advertisement.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title: Robinson Crusoe
Edition:
Publishing year:
Course: literature
Price: 300
Comments: excellent quality
3. The tester presses "add".
The book should then be uploaded to the market.
4. The tester then tries to edit the ad by pressing the edit-button at the market and use his/hers password. The tester changes the author to Astrid Lindgren and then presses the button "save".

Postcondition: The book is editable and the author is changed to Astrid Lindgren
Related requirements: 1.2

Test ID 2.1.1-P **Search by title**
Description: In a search, a buyer can specify a book's title.
Precondition: Test ID 0.1. The tester are looking for a specific book at the market and the book "Robinson Crusoe" by Daniel Defoe is added to the market.
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title: Robinson Crusoe

Course:
3. The tester presses the button “search”.

Postcondition: The tester should get a match and find the book.
Related requirements: 2.1.1

Test ID 2.1.1-N **Search by title**
Description: In a search, a buyer can specify a book’s title.
Precondition: Test ID 0.1. The tester are looking for a specific book at the market
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title: Petter och hans fyra getter
Course:

3. The tester presses the button “search”.

Postcondition: The tester will not get a match in the market with the following message: “your search didn't match any books at the market”.
Related requirements: 2.1.1

Test ID 2.1.2-P **Search by author**
Description: In a search, a buyer can specify a book’s author.
Precondition: Test ID 0.1. The tester are looking for a specific book at the market. The Book “Robinson Crusoe” with the author “Daniel Defoe” is in the market.
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title:
Course:

3. The tester presses the button “search”.

Postcondition: The tester gets a match and finds the book “Robinson Crusoe”
Related requirements: 2.1.2

Test ID 2.1.2-N **Search by author**

Description: In a search, a buyer can specify a book's author.
Precondition: Test ID 0.1. The tester are looking for a specific book at the market
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author: Emil Nyström
Book title:
Course:

3. The tester presses the button "search".

Postcondition: The tester will not get a match in the market, and get the following exception message: "your search didn't match any books at the market".
Related requirements: 2.1.2

Test ID 2.1.3.1-P **Search by course - course name**

Description: In a search, a buyer can specify which course the book should be used in
Precondition: Test ID 0.1. The tester are looking for a specific book at the market. The book "Java direkt med Swing" for the course "objektorienterad programmering" is in the market. The book "Linear algebra and its applications" for the course TMV215 is in the market.
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title:
Course: objektorienterad programmering

3. The tester presses the button "search".
Postcondition: The tester will get a match on the search: "Java direkt med Swing".

Related requirements: 2.1.3

Test ID 2.1.3.2-P **Search by course - course code**

Description: In a search, a buyer can specify which course the book should be used in

Precondition: Test ID 0.1. The tester are looking for a specific book at the market. The book “Java direkt med Swing” for the course “objektorienterad programmering” is in the market. The book “Linear algebra and its applications” for the course TMV215 is in the market.
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title:
Course: TMV215

3. The tester presses the button “search”.

Postcondition: The tester will get a match on the search: “Linear algebra and its applications”.
Related requirements: 2.1.3

Test ID 2.1.3-N **Search by course**
Description: In a search, a buyer can specify which course the book should be used in
Precondition: Test ID 0.1. The tester are looking for a specific book at the market. The book “Java direkt med Swing” for the course “objektorienterad programmering” is in the market. The book “Linear algebra and its applications” for the course TMV215 is in the market.
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title:
Course: Harry Potter Knowledge

3. The tester presses the button “search”.

Postcondition: The tester will not get a match in the market with the following message: “your search didn't match any books at the market”.
Related requirements: 2.1.3

Test ID 2.1.4-P **Search sorting by price**

Description: A search can be sorted on any of the available attributes that a book can have. Originally it is sorted by price.
Precondition: Test ID 0.1. The tester are looking for a specific book at the market. Three ads of the book "Robinson Crusoe" is in the market with the prices: 300, 100, 500.
Test steps:
1. The tester presses the button for searching a book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title: Robinson Crusoe
Course:

3. The tester presses the button "search".

Postcondition: The tester gets three books as result with the cheapest first (100, 300, 500).
Related requirements: 2.1.4

Test ID 2.2-P **Buying requests**
Description: A buyer can upload a buy request
Precondition: Test ID 0.1, installed application.
Test steps:
1. The tester opens the application
2. The tester presses the button for "add a request"
3. The tester fills in the information regarding the request:
Author: Daniel Defoe
Book title: Robinson Crusoe
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price: 300 SEK
Comments: excellent quality

4. The tester presses the button "add request"
Postcondition: A buying request is added to the market.
Related requirements: 2.2

Test ID 3.1-P **Book added by ISBN-number**
Description: A book can be uploaded by ISBN-number.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.

Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title:
ISBN: 9789144060743
Version:
Publishing year:
Course: Objektorienterad programmering
Price: 300 SEK
Comments:

3. The tester presses "add".

Postcondition: The tester gets the title: "Java direkt med Swing", the author "Jan Skansholm", and the edition "6" automatically.
Related requirements: 3.1, 3.2

Test ID 3.1-N **Book added by ISBN-number**
Description: A book can be uploaded by ISBN-number.
Precondition: Test ID 0.1. The tester have a book to sell, have the application, and have filled in user details.
Test steps:
1. The tester presses the button for adding a new book.
2. The tester then fills in the following information details in the text boxes:
Author:
Book title:
ISBN: 11111111111
Version:
Publishing year:
Course: Objektorienterad programmering
Price: 300 SEK
Comments:

3. The tester presses "add".

Postcondition: The tester gets a error message in the text box for author: "No title found".
Related requirements: 3.1, 3.2

Test ID 6.1-P **Book sold by auction**
Description: A seller can sell his/hers book on auction
Precondition: Test ID 0.1, 1.1

Test steps:
1. The tester presses the button for adding a new book.
2. The tester presses the button “sell in auction”
3. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title:
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price:
Comments: excellent quality

3. The tester presses “fill in auction-details”.
4. The tester fills in auction details: startdate: 2012-06-01, enddate: 2012-06-15 and start amount: 50 SEK.
5. Tester presses “add book”

Postcondition: A book is uploaded at the market and sold by auction.
Related requirements: 6.1

Test ID 6.1-N **Auction, invalid date**
Description: A seller can sell his/hers book on auction
Precondition: Test ID 0.1, 1.1
Test steps:
1. The tester presses the button for adding a new book.
2. The tester presses the button “sell in auction”
3. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title:
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price:
Comments: excellent quality

3. The tester presses “fill in auction-details”.
4. The tester fills in auction details: startdate: 2012-01-01, enddate: 2012-06-15, start amount: 50 SEK, reservation price:
5. Tester presses “add book”

Postcondition: The tester gets a exception message: “Invalid date, a auction must start after the present date”.
Related requirements: 6.1

Test ID 6.2-P **Book sold by auction - reservation price**
Description: A seller can sell his/hers book on auction
Precondition: Test ID 0.1, 1.1
Test steps:
1. The tester presses the button for adding a new book.
2. The tester presses the button “sell in auction”
3. The tester then fills in the following information details in the text boxes:
Author: Daniel Defoe
Book title:
ISBN:
Version: 2nd edition
Publishing year: 1999
Course: literature
Price:
Comments: excellent quality

3. The tester presses “fill in auction-details”.
4. The tester fills in auction details: startdate: 2012-06-01, enddate: 2012-06-15, start amount: 50 SEK, reservation price: 300 SEK
5. Tester presses “add book”

Postcondition: A book is uploaded at the market and sold by auction with a reservation price.
Related requirements: 6.2

1.1 Non-functional tests

Test ID NF1 **Usability**
Description: The tester is checking the usability of the application.
Test steps: The tester should ask someone who hasn't used the application before and ask them to add a book. and take time of the process and ask questions afterwards, in order to understand if the application is considered easy to understand. Read more under headline 7 regarding black box testing.
Postcondition: The application is easy to understand and use.
Related requirements: NF1

Test ID NF2 **Usability**
Description: The tester is checking the usability of the application.
Test steps: The tester identifies one person who does not understand a part of the application, and then asks the person to read in the help-view. The person does this and then understands how to use the application. The tester also reads the information in the help-view and make sure it's understandable and correct.
Postcondition: The application is easy to understand and use.
Related requirements: NF2

Test ID NF3 **Reliability**
Description: The tester is checking the reliability of the application.
Test steps: The tester will add books at the same time as three other testers. This small test will indicate if the server can handle four uploading books at the same time.The testers will use the android emulator in Eclipse for these tests.
Postcondition: The application is reliable.
Related requirements: NF3

Test ID NF4 **Performance**
Description: The tester is checking the performance of the application.
Test steps: The tester should view information about a book and then turn off the internet connection the the phone. The tester should then open the application and still see the information on the book viewed last time.
Postcondition: The application har good performance.
Related requirements: NF4

Test ID NF6 **Implementation**
Description: The tester is checking the implementation and size of the application.
Test steps: After every release, the tester should control that the client-side application cannot exceed 10 MB.
Postcondition: The application is slim.
Related requirements: NF6

2. Black box testing

A black box test is conducted in order to test and verify some of the non-functional requirements stated above (NF1, NF2, NF4).

Black box testing is a method of software testing that tests functionality of an application without considering the internal structures or workings. The testers in the black box test will only consider the functionality and ease of use of the application, and nothing regarding the actual code. A black box test will be done together with three potential users of the application who never have seen the application before. The testers will be adding a book and do a search of a book and give comments on the application. Information from this black box test will be used in the non-functional tests listed above.

3 Automatic tests (UNIT)

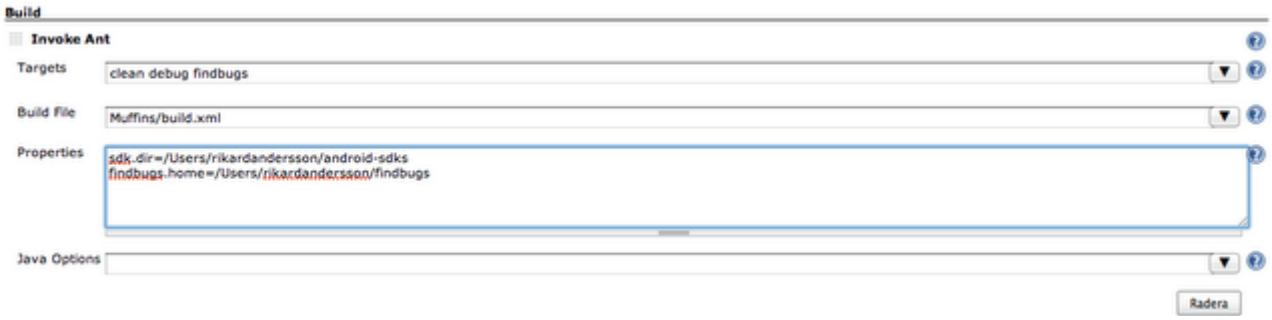
Automatic and UNIT testing refers to tests which verify the functionality of a specific section of code. Automatic tests have been conducted in this project in order to test the functionality of the software. An android test project has been created in Muffins/tests. From the start, only regular JUnit tests were run for testing the various regular java methods in DataBookFactory.java, MuffinsUtility.java, and ServerCommunicator.java. Later, Activity tests were also implemented. Problems faced during testing were, among other things, the implementation of the equals method of DataBook and JSONObject. Since time is short, the project team chose to test the attributes as found by for example getAuthor(), instead of comparing entire DataBooks. This will be corrected in future releases.

3.1 Nightly builds

Nightly builds have been conducted with the software Jenkins CI. Jenkins CI is the leading open-source continuous integration server and provides over 400 plugins to support building and testing of software projects.

Jenkins is installed locally on one computer where all automatic builds are made. A plugin for github is used to pull the latest version from the branch WorkingBranch.

Also findbugs is used to analyze the results from the build. On activation an ant-line is invoked: "clean debug findbugs" which uses Muffins/build.xml to build the project. For this Jenkins also need the paths for both the Android SDK and findbugs.



Worth noting is that ant requires the external libraries used in the project to be located in the folder /{path_to_project_folder}/libs/ and not /{path_to_project_folder}/lib/. This caused us a few hours of bashing our heads against the keyboard (not really since we all use laptop and that would break the computer).

The test is run every night at 10 pm and requires the computer to be turned on at this time. During this time all test are run and a report is generated. More about this in the coming sections. Builds is also run before every release.

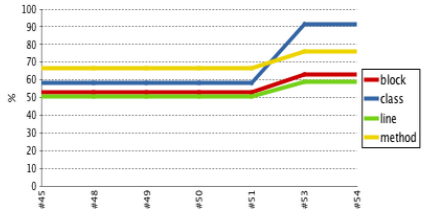
3.2 Code coverage

Code coverage is a useful tool within software testing in order to see how much of the code being tested. Although, it is important to remember that it only tells you how much of the code being tested, and not whether your code is tested enough. Neither if the tests are good and appropriate. Code coverage can also be different depending on how and what you measure. The percentage of classes may be 100% but actual methods being

tested can be very low. Code coverage of 100% is of course good, but does not automatically mean you have good testing strategy. For this project, a specific objective regarding the amount of code coverage have been of minor importance. Higher priority have been given to developing user and UNIT-tests suitable for testing the different functionality of the application. However, a goal minimum of 30% code coverage (lines) have been set for the most important classes. The code coverage have been conducted with the use of EMMA and Jenkins.

All automatic unit-tests are run during nightly builds by Jenkins and a coverage report is generated by the EMMA-plugin for Jenkins. Also a coverage trend is recorded and presented by Jenkins. This can look something like this:

Package: itBrainiacs.muffins



Coverage Summary

name	class	method	block	line
itBrainiacs.muffins	91,7% 11/12	76,0% 57/75	62,9% 1204/1915	58,7% 310/528

Coverage Breakdown by Source File

name	class	method	block	line
AddBookActivity.java	100,0% 2/2	60,0% 6/10	57,4% 221/385	56,2% 50/88
DataBook.java	100,0% 1/1	89,2% 33/37	86,1% 173/201	86,3% 63/73
DataBookFactory.java	100,0% 1/1	80,0% 4/5	67,9% 391/576	53,3% 106/199
MuffinsActivity.java	100,0% 1/1	100,0% 2/2	100,0% 103/103	100,0% 19/19
MuffinsUtility.java	100,0% 1/1	50,0% 1/2	58,7% 37/63	42,7% 6/15
SearchActivity.java	100,0% 1/1	100,0% 4/4	100,0% 130/130	100,0% 25/25
SearchItemActivity.java	0,0% 0/1	0,0% 0/2	0,0% 0/4	0,0% 0/2
SearchResultActivity.java	100,0% 2/2	75,0% 3/4	59,8% 55/92	73,7% 14/19
ServerCommunicator.java	100,0% 1/1	50,0% 2/4	73,0% 65/89	60,0% 17/28
SettingsActivity.java	100,0% 1/1	40,0% 2/5	10,7% 29/272	16,7% 10/60

From this build we can conclude that more tests needs to be written for SettingsActivity and SearchItemActivity in order to achieve our goal of 30% code coverage (lines).
The rate of 30 percent can be seen as low and the goal needs to be revised progressively. Already now, we can see that a higher rate of code coverage is reached and therefore we can increase the goal in order to ensure testing quality for the project.
When we discussed the level of a new percentage goal in our project team, we disagreed and had problems of finding arguments for a specific level. After some discussion however, we could agree on a new goal:

We want to aim for an increasing trend in code coverage

Junior programmers like us should not focus on a certain level of code coverage. It it hard and demands a lot of experience. It is better if we focus on writing good tests and use code coverage as a complementary tool for ensuring that a major part of the code is being tested; aiming for a increasing trend in coverage.

4. Other related documents

Test_report_release_0.2.pdf	Test report for the latest release
Developer_and_new_user_guidelines.pdf	Information for new users and testers
Requirements.pdf	Information regarding requirements