Contents

1. 2015

1.1 September

Blog (2015-09-30 12:18)

1.2 October

Our Vision: SAPACA - Face Recognition (2015-10-04 16:56)

Our Vision is to create an application (SAPACA - Face Recognition) which is capable of identifying a person from a digital image or video by comparing the image or video with a facial database. It also will also be possible to train the face recognition process by extending the facial database.

Face Recognition involves two stages:

1. Face Detection: An image is searched to find any face.
2. Face Recognition: The detected faces are compared to a database of known faces and decide which person are shown on the image.

The application will be accessible through a RESTful Web site which provides additional information to a person and therefore uses the Spring Web model-view-controller (MVC) framework. For example it can directly generate XML, JSON and many other types of content and allows the use of validation support with simple Spring annotations. Additional information are:

- (If face is recognized successfully) Name, age, nationality
- When was the first time the person got recognized?
- How many times has the person been recognized?
- Show all images of the person

Stay tuned for more news!

Your SAPACA-Team

Digister (2015-10-05 08:02:30)
Hi, I really like the idea of your project and I am very interested in your next posts. Well done so far :). Kind Regards Fabian - SAS TEAM

theratnar (2015-10-05 08:17:20)
Hello Digister, thank you for your interest. We hope you like our future results ;) Kind Regards Pascal – SAPACA

Pascal Maximilian Bremer (2015-10-05 08:12:42)
I see you are using a MySQL Database for your project. So will there be a public API to use your software? And if that is the case how will you handle the privacy protection of each individual ?

caro340 (2015-10-05 08:18:58)
Hello Pascal, thank you for your interest in our application. About the public API, we are not sure yet. It could be possible. If there is going to be a public API it will come without the facial database, so we're not publishing the API with any personal information. best wishes, caro - sapaca

Nico (2015-10-05 08:21:45)
Hi, I think you are doing a really interesting project. Are your images saved in the database or on an external file server? Kind Regards Nico - Bookster

theratnar (2015-10-05 08:29:00)
Hello Nico, thank you for your interest. We planned to use a Database, but plans can change ;) Kind Regards Pascal – SAPACA

Home (2015-10-05 08:18)

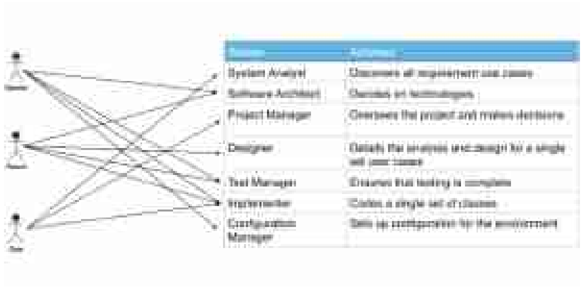
Team roles and specifications (2015-10-05 10:38)

After the vision it's time to define roles for our project.

- System Analyst: Caro works out which functions and use cases are required

- Software Architect: Pascal and Sascha take care that MVC is implemented correctly on decide which technologies will be used
- Project Manager: Caro oversees the project to guarantee the working flow and makes decisions in order to prevent problems
- Designer: Pascal is resonsible for detailing the analysis and design for a single set of user cases
- Test Manager: Pascal and Sascha are responsible for testing our application to ensure our progress
- Implementer: Sascha, Pascal and Caro implement different sets of classes
- Configuration Manager: Sascha sets up configuration for the environment

The defined roles are also shown in the picture below.

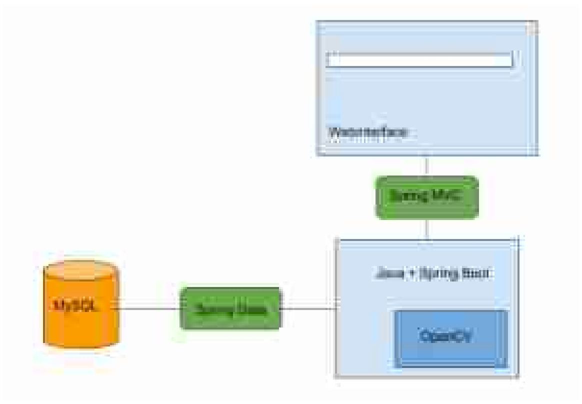


Roles	Activities
System Analyst	Discovers all requirement use cases
Software Architect	Decides on technologies
Project Manager	Oversees the project and makes decisions
Designer	Details the analysis and design for a single set user cases
Test Manager	Ensures that testing is complete
Implementer	Codes a single set of classes
Configuration Manager	Sets up configuration for the environment

The technologies which will be used are:

- Java
- OpenCV (JavaCV)
- MySQL
- Spring Boot, MVC and Data
- Build Tool: Gradle
- (Thymeleaf)

System Overview:



[1]

In this overview you can see our planned structure. We will build a main Java + Spring Boot and OpenCV control unit which has access over Spring Data to a MySQL database and via Spring MVC to a webinterface.

We will use Spring Boot to create production-ready applications. So we don't have to configure an webcontainer like Apache Tomcat. Spring Web MVC will be used to implement easily the MVC pattern and also for the use of annotations. Moreover it will help to simply implementate Thymeleaf, which propably will be the template engine for generating the view. Spring Data will help us to establish the connection to the MySQL database and makes it easy for us to use CRUD operations. The OpenCV library contains efficient algorithms for face detection and recognition - and exactly for that we will use it. So we don't have to programm it ourself.

1. `https://sapacablog.files.wordpress.com/2015/09/screenshot.png`

---

jockelmore (2015-10-11 16:07:07)
Hello sapaca-team, you have clearly defined your team roles and the technology you will be using. For the next pictures you are creating it would be good if the text is bigger to make it easier for reading. Kind regards, Jan-Eric Gaidusch from saphijaga

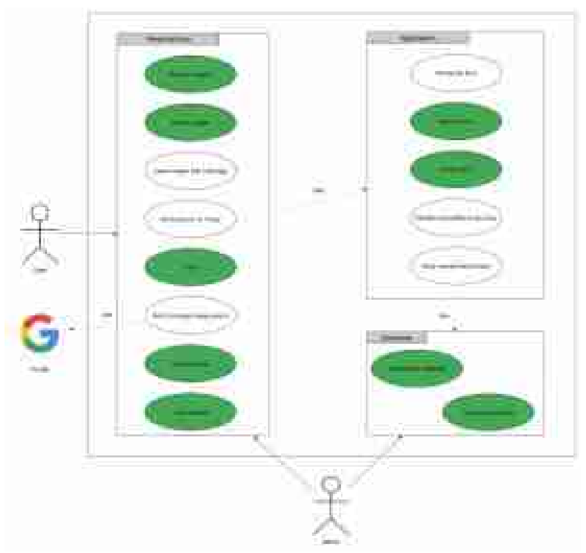
caro340 (2015-10-12 07:21:06)
Hello Jan-Eric Gaidusch, thanks for your reply! We're going to use bigger pictures/text in the future. Best wishes, caro

Digister (2015-10-12 07:37:28)
Hi sapaca-team, you clearly set your responsibilities and also defined team roles using the RUP terminology. The image is great but it would be perfect if there is a link where you can see the picture in a bigger size. The visualization of the using technologies makes it more understandable and is well devided. Well done :). Kind regards, team Unveiled

caro340 (2015-10-12 08:28:07)
Hello Digister, thanks for your reply. We will use bigger images the next time :) best wishes, caro

Software Requirements Specifications (2015-10-16 05:07)

In the following picture you can see our use-case diagram. The yellow colloured circles are the one which will be finished in this semester. The circles which are not coloured will be completed in the next semester.


[1]

We choose GitHub as our Wiki because it has a good version control, everyone can easily access it and our documentation is directly linked to our project.

[2]GitHub Wiki

1. `https://sapacablog.files.wordpress.com/2015/10/ucd.jpg`
2. `https://github.com/Ratnar/FaceRecognition/wiki`

Sebastian Adams (2015-10-18 17:09:03)
Hi SAPACA-Team, I like your Use-Case-Diagram, because it is easy to see, that you have two scopes: basic functionality and face-recognition. Maybe you could differentiate the webinterface-user and the API-user. Your SRC supplement the diagram. How will you implement the "Goolge Search binding"? Are there any interfaces from Google? King regards, Sebastian (Unveiled-Team)

suddenh4x (2015-10-18 20:54:12)
Hi Sebastian, thanks for your reply and your suggestion! I think a distinction between Web interface – user and API – user would not increase the intelligibility. With the “Google Search binding” – feature the user is able to search easily in the Google database for images which are unknown to the application database. It isn’t definite yet, but probably we will use the official Google CustomSearch API to implement this feature. Kind regards, Sascha

Pascal Maximilian Bremer (2015-10-19 08:28:11)
Hi SAPACA-Team, I read your Wiki and I was wondering if you could give a detailed insight into your internal infrastructure. It’s not quit clear to me yet how the recognition is achieved in the backend. What kind of database do you use? You did a good job on clearifying all the requirements and dependencies. Kind regards, Max (MathJunkie)

caro340 (2015-10-19 08:31:59)
Hello Max, thanks for your feedback. We will add a detailed insight of our internal infrastructure. We will also add some information about the used database, yet, it is not quite clear which one we will be using. Best wishes, caro

Use Case **(2015-10-24 12:06)**

Hey!

Today we created 2 Use Cases. Check them out:

- [1]Use Case: Label Image
- [2]Use Case: Detect Face

We also linked them into our [3]SRS.

You can also check out our [4]Mockups.

Best wishes,

caro

1. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Label-Image`
2. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Detect-Face`
3. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Requirements-Specification`
4. `https://github.com/sapacaFaceRecognition/Documentation/tree/master/Mockup/Pictures`

---

streboretkowski (2015-10-26 09:21:23)
Hi sapaca, I really like your Use Cases and especially your Mockups. They are easily understandable. You used verbs for all your activities. You used the template and linked your files in your SRS. But I think in your Use Case diagrams the presentation of parallel activities is a little bit wrong. I think you need to fork and join the activities with a bold stroke. Kind regards, Fabian - strebo team

caro340 (2015-10-26 09:24:23)
Hello Fabian, thanks for your feedback. I didn't notice the error in the diagrams, thanks. We will change the diagrams later today. best wishes, caro

Simon (2015-10-26 09:23:23)
Hey SAPACA, you did a good job with your use-cases. But I have one idea for your use-case „Label Image“. It would be very nice if you could add a nickname to a face this would make your tool even more individual. Best regards, Simon (MathJunkie-Team)

caro340 (2015-10-26 09:30:04)
Hey Simon, thanks for your feedback. We will update the UC and discuss your idea. Best wishes, caro

DaBrauun (2015-11-02 09:19:55)
Hi SaPaCa, nice work with the Use Case definitions. I have one note regarding the Activity Diagram of the Use Case "Label Image": In case of an error, how does the Use Case coninue after the error message has been displayed? I learned, that all paths should eventually end in the black "end point". Kind regards Bookster Daniel

theratnar (2015-11-02 09:29:40)
Hi Daniel from Bookster, you´re right, we must have missed that. I´ll be editing our activity diagram as soon as possible. kinds regards, Pascal – SAPACA-Team

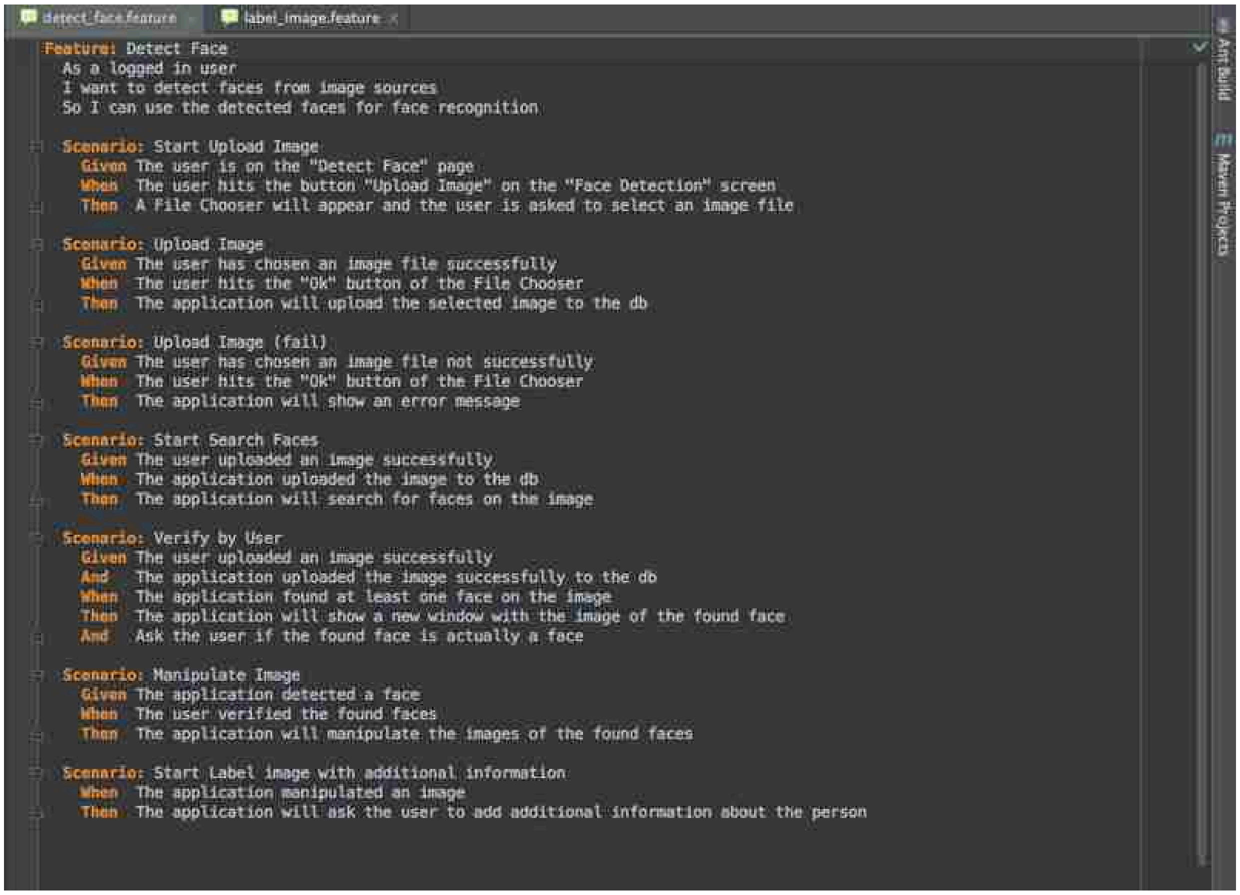
1.3 November

Narratives (2015-11-01 15:48)

Hey followers,

we have some new deatils for you. We made narratives for our Use Cases.

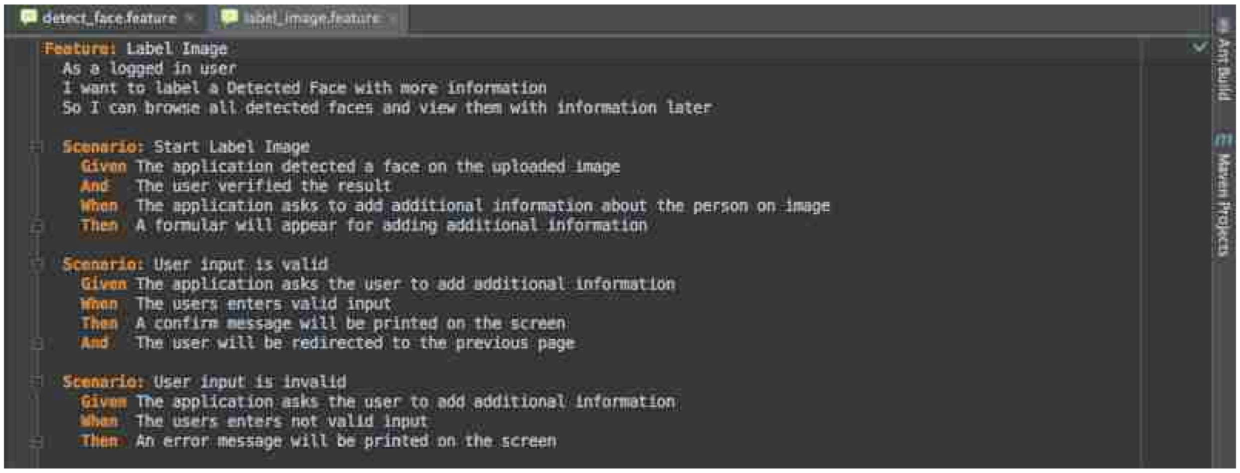
Detect Face:



```
Feature: Detect Face
  As a logged in user
  I want to detect faces from image sources
  So I can use the detected faces for face recognition

  Scenario: Start Upload Image
    Given The user is on the "Detect Face" page
    When  The user hits the button "Upload Image" on the "Face Detection" screen
    Then  A File Chooser will appear and the user is asked to select an image file

  Scenario: Upload Image
    Given The user has chosen an image file successfully
    When  The user hits the "Ok" button of the File Chooser
    Then  The application will upload the selected image to the db

  Scenario: Upload Image (fail)
    Given The user has chosen an image file not successfully
    When  The user hits the "Ok" button of the File Chooser
    Then  The application will show an error message

  Scenario: Start Search Faces
    Given The user uploaded an image successfully
    When  The application uploaded the image to the db
    Then  The application will search for faces on the image

  Scenario: Verify by User
    Given The user uploaded an image successfully
    And   The application uploaded the image successfully to the db
    When  The application found at least one face on the image
    Then  The application will show a new window with the image of the found face
    And   Ask the user if the found face is actually a face

  Scenario: Manipulate Image
    Given The application detected a face
    When  The user verified the found faces
    Then  The application will manipulate the images of the found faces

  Scenario: Start Label image with additional information
    When  The application manipulated an image
    Then  The application will ask the user to add additional information about the person
```

For more details view [1]UC Detect Face

Label Image:



```
Feature: Label Image
  As a logged in user
  I want to label a Detected Face with more information
  So I can browse all detected faces and view them with information later

  Scenario: Start Label Image
    Given The application detected a face on the uploaded image
    And   The user verified the result
    When  The application asks to add additional information about the person on image
    Then  A formular will appear for adding additional information

  Scenario: User input is valid
    Given The application asks the user to add additional information
    When  The users enters valid input
    Then  A confirm message will be printed on the screen
    And   The user will be redirected to the previous page

  Scenario: User input is invalid
    Given The application asks the user to add additional information
    When  The users enters not valid input
    Then  An error message will be printed on the screen
```

More deatils [2]UC Label Image

All together can be viewed on our [3]SRS.

Stay tuned!

Your SAPACA-Team

1. https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Detect-Face
2. https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Label-Image
3. https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Requirements-Specification

---

Alexander Friese (2015-11-02 08:18:30)
Dear SAPACA team, you wrote your narratives for two different use cases, which consists of more than one scenario. You also used the correct syntax for the narrative as well as your IDE identifies your .feature file with color highlighting. You made a good job with this. Do your tests run yet? Kind regards, Alex Team appXpired

theratnar (2015-11-02 08:51:28)
Hello Alex, thank you for your feedback. We are working on our test and they will be comming soon, just stay prepared ;) Kinds regards, Pascal - SAPACA-Team

lolololarry (2015-11-02 08:31:54)
Dear Sapaca Team, I really like your .feature files and the documentation you did. It looks all well-thought-out and everything is well and clear defined. I really like, that you put some screenshots from your feature files on your blog so it is easier to see what you did. You really did a great job with this. Best wishes, Larissa from the MathJunkie Team

theratnar (2015-11-02 08:56:58)
Hello Larissa, thank you for your positiv feedback, we´re working on making this project a good one. kind regards, Pascal - SAPACA-Team

Class Diagram (2015-11-08 22:14)

Hey!

Today we want to show you our first class diagram:

[embed]https://raw.githubusercontent.com/sapacaFaceRecognition/Documentat ion/master/Class %20Diagramm/Class %20Diagramm.PNG[/embed]

[1]https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/ master/Class %20Diagramm/Class %20Diagramm.PNG

It's automatically generated with "The ObjectAid UML Explorer for Eclipse" and if something changes, you will be able to see it here. Maybe you are surprised about the missing relationships. This is a problem of Spring, which uses dependency injection and can't be bypassed without drawing it by hand.

Stay tuned!

Your SAPACA-Team

1. `https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/master/Class%20Diagramm/Class%20Diagramm.PNG`

---

lolololarry (2015-11-09 09:20:06)
Dear Sapaca Team, i really like your Class Diagramm because it is very clear and easy. You can easily see what objects the single classes have. The only thing i wonder about is why there are not dependencies. Do you not have some dependencies between your classes ? To sum up, i really like your class diagrams and will stay tuned. I am really interested in your project and I think it is really cool ! So good job guys ! Best wishes, Larissa from the Math Junkie Team

suddenh4x (2015-11-09 09:47:48)
Hello Larissa, thank you for your comment. The problem with the missing dependencies/relationships is that we use the Spring framework which uses dependency injection. And therefore the tool wasn't able to show up the connections. Kind regards, Sascha from SAPACA

streboretkowski (2015-11-09 09:52:59)
Hey Sapaca Team, I see you have at least 5 classes that contains methods and attributes. Your UML class Diagram is automatically generated and provide a quick and clear overview. But I'm not sure if it provides all functionalities you promised for December. Did you considered the login and the administration you promised in your diagram? And at least I hope you can fix your problem with the missing dependencies. Kind regards Fabian - strebo

suddenh4x (2015-11-09 10:09:08)
Hello Fabian, thank you for your comment. The login and also the administration function is realised within the MainController class. The problem with the missing dependencies/relationships in the class diagram cannot be solved completely because of the dependency injection of the Spring framework. Kind regards Sascha from SAPACA

Software Architecture Document - SAD (2015-11-15 23:11)

Hey,

we created our [1]SAD.

If you're wondered about the missing view: our view contains just HTML-files, which can't be automatically added to the UML diagram. We are going to add the view tomorrow.

Stay tuned.

Best wishes,

caro

1. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Architecture-Documentation`

Simon (2015-11-16 09:45:56)
Hey sapaca, your Software Architecture Document looks really good, but what is with your view? Contains your view HTML-files? best wishes Simon from MathJunkies

caro340 (2015-11-16 10:01:12)
Hey Simon, thanks for your reply. Yes, we are using HTML-files as a view. We will draw them in by hand later that day. Best wishes, caro

streboschreck (2015-11-16 10:23:42)
Hello sapaca, your Software Architecture Document is filled out, your diagrams are very detailed and also auto generated. You also named your framework and explained your architecture. The one thing I have to note is, that the arrows connecting the "FaceDetection" class with the "Face" class are a little close together. Maybe you could rearrange them so that there is a little more space between them. Overall nice work! Greetings, david from strebo

caro340 (2015-11-16 10:27:47)
Hey david, we will change the UML diagram anyway, so we're going to change the arrows :) best wishes, c

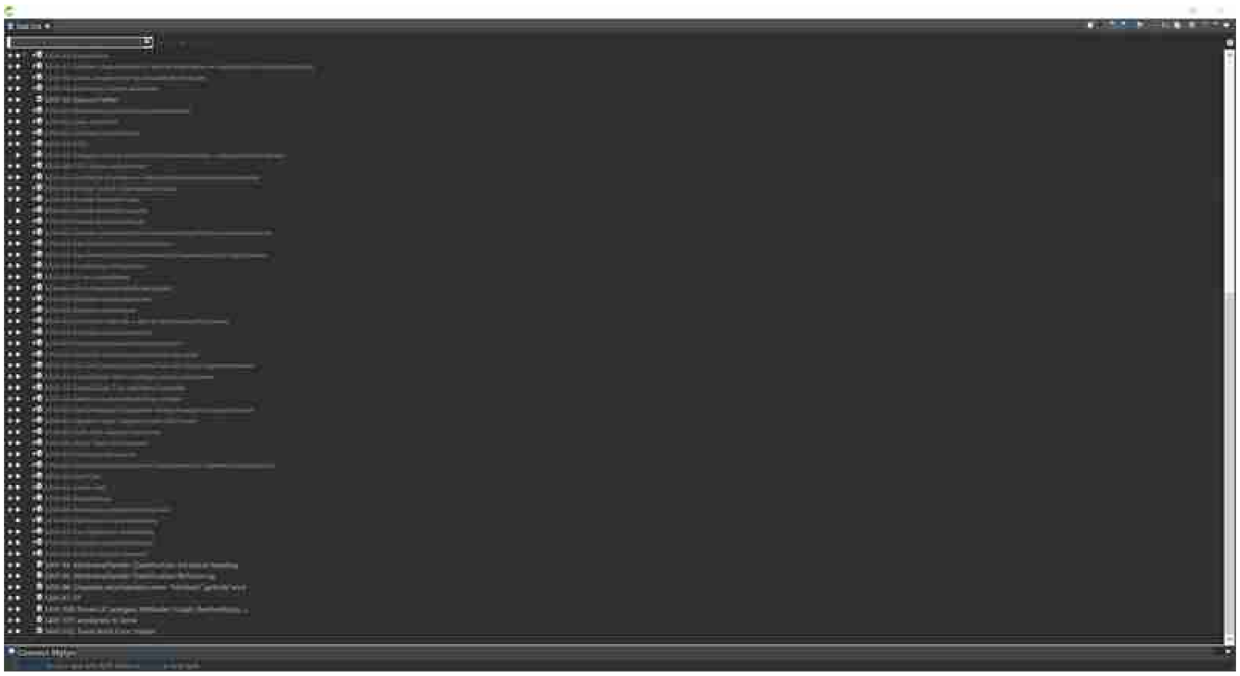
Jira (2015-11-22 14:06)

Hey followers,

we are now on Jira:
http://193.196.7.27:8080/secure/RapidBoard.jspa?rapidView=7 &projectKey=SAVI &selectedIssue=SAVI-10

Our first Sprint which lasts 2 weeks already started.

We also integrated JIRA in our IDE. The following screenshot shows you the JIRA tasks inside the IDE



[1]https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/ b679e848d8c-0769e46947dd65851c85ab1aba0fd/Jira/Jira %20in %20STS.png

stay tuned,

Your SAPACA-Team

1. `https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/b679e848d8c0769e46947dd65851c85ab1aba0fd/Jira/Jira%20in%20STS.png`

Simon (2015-11-23 09:07:47)
Hey sapaca, your jira configuration and your sprint looks really good, but you write, that you could not connect it to your github project. At our project, we have the same problem. best wishes Simon from MathJunkies

theratnar (2015-11-23 09:20:16)
Hey Simon, thank you for you comment. We hope that our Sprint just goes fine. kinds regards, Pascal, SAPACA-Team

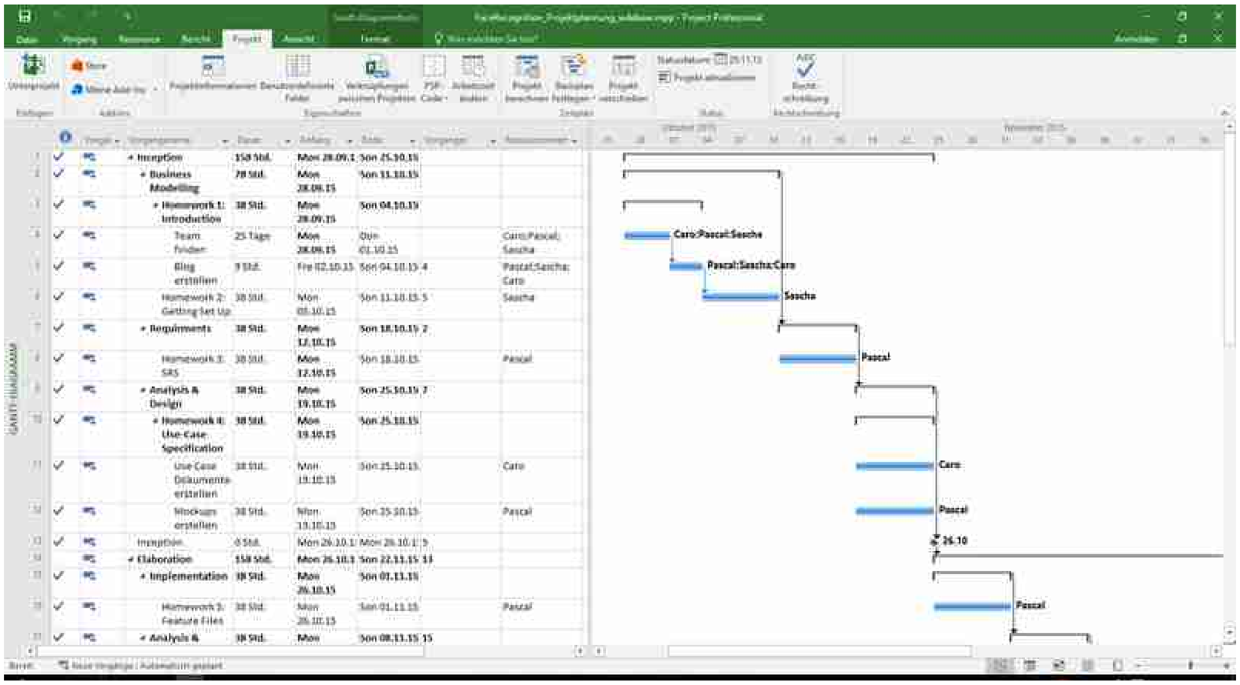
streboretkowski (2015-11-23 09:37:29)
Hi SAPACA, great job you done! You linked to Jira, created your Scrumboard plus your first Sprint with several Stories and Tasks. Only one question left: Do you added somewhere Tags to your Stories according to the RUP workflow (or naming convention) ? This would be nice. Kind regards, Fabian - strebo team

theratnar (2015-11-23 09:41:56)
Hello Fabian, thank you for your comment. We have to add tags to our story, it will be added soon. kinds regards, Pascal, SAPACA-Team

RUP (GANTT Chart) **(2015-11-30 07:50)**

Hey followers,

we uploaded our GANTT Chart:
[1]https://github.com/sapacaFaceRecognition/Documentation/raw/master/MS %20Project/FaceRecognition _Projektplannung _wddkew.pdf



	Vorgangsname	Dauer	Anfang	Ende	Vorgänger	Ressourcennamen
1	Inception	150 Std.	Mon 28.09.1	Son 25.10.15		
2	Business Modelling	78 Std.	Mon 28.09.15	Son 11.10.15		
3	Homework 1: Introduction	38 Std.	Mon 28.09.15	Son 04.10.15		
4	Team finden	25 Tage	Mon 28.09.15	Din 01.10.15		Caro;Pascal;Sascha
5	Blog erstellen	9 Std.	Fre 02.10.15	Son 04.10.15	4	Pascal;Sascha;Caro
6	Homework 2: Getting Set Up	38 Std.	Mon 05.10.15	Son 11.10.15	5	Sascha
7	Requirements	38 Std.	Mon 12.10.15	Son 18.10.15	2	
8	Homework 3: SRS	38 Std.	Mon 12.10.15	Son 18.10.15		Pascal
9	Analysis & Design	38 Std.	Mon 19.10.15	Son 25.10.15	7	
10	Homework 4: Use-Case Specification	38 Std.	Mon 19.10.15	Son 25.10.15		
11	Use Case Dokumente erstellen	38 Std.	Mon 19.10.15	Son 25.10.15		Caro
12	Mockups erstellen	38 Std.	Mon 19.10.15	Son 25.10.15		Pascal
13	Inception	0 Std.	Mon 26.10.1	Mon 26.10.1	9	
14	Elaboration	158 Std.	Mon 26.10.1	Son 22.11.15	13	
15	Implementation	38 Std.	Mon 26.10.15	Son 01.11.15		
16	Homework 5: Feature Files	38 Std.	Mon 26.10.15	Son 01.11.15		Pascal
17	Analysis &	38 Std.	Mon	Son 08.11.15	15	

stay tuned,

Your SAPACA-Team

1. `https://github.com/sapacaFaceRecognition/Documentation/raw/master/MS%20Project/FaceRecognition_Projektplannung_wddkew.pdf`

Simon (2015-11-30 08:34:00)
Hey sapaca, your gantt-chart looks really unclear, it would be better, when you choose a bigger size for the pdf. best wishes Simon from MathJunkies

suddenh4x (2015-11-30 09:32:11)
Hi Simon, thank you for your improvement proposal. Meanwhile we have enlarged the size for a better readability. Kind regards Sascha from SAPACA

Sebastian (2015-11-30 10:17:47)
Hey SAPACA-Team, it's really hard to get the information from your PDF, because you can't see the whole gantt-chart in an overview. In MS-Project you could use an PDF-Printer and print it as A0. Your structure of the tasks is good and you've used milestones. King regards, Sebastian (Unveiled)

suddenh4x (2015-11-30 10:30:36)
Hi Sebastian, thank you for your comment. We already have improved the size and used an PDF Printer for the GANTT Chart. Kind regards Sascha from SAPACA

1.4 December

Midterm (2015-12-22 13:06)

Hello followers,

GitHub-Code: [1]sapacaFaceRecognition/FaceRecognition
GitHub-Doc: [2]sapacaFaceRecognition/Documentation

[3]JIRA

SRS: [4]Software Requirements Specification
SAD: [5]Software Architecture Documentation

UC1: [6]Browse Image
UC2: [7]Delete Image
UC3: [8]Detect Face
UC4: [9]Label Image
UC5: [10]Upload Image

[11]Mockups

Gantt File: [12].mmp
Gantt PDF: [13].pdf

Midterm Präsentation: [14]Präsentation

Feature File Logs:
Wir haben zu jedem Featurefile den letzten Teil des Logs als Screenshot in die einzelnen UseCases eingefügt (2.1.2)

Your SAPACA-Team

1. `https://github.com/sapacaFaceRecognition/FaceRecognition`
2. `https://github.com/sapacaFaceRecognition/Documentation`

3. `http://193.196.7.27:8080/secure/RapidBoard.jspa?rapidView=7&projectKey=SAVI&view=planning.nodetail`
4. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Requirements-Specification`
5. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Architecture-Documentation`
6. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Browse-Image`
7. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Delete-Image`
8. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Detect-Face`
9. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Label-Image`
10. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Upload-Image`
11. `https://github.com/sapacaFaceRecognition/Documentation/tree/master/Mockup/Pictures`
12. `https://github.com/sapacaFaceRecognition/Documentation/blob/master/MS%20Project/FaceRecognition_Projektplanung_wddkew.mpp`
13. `https://github.com/sapacaFaceRecognition/Documentation/blob/master/MS%20Project/FaceRecognition_Projektplanung_wddkew.pdf`
14. `https://github.com/sapacaFaceRecognition/Documentation/blob/master/Midterm/presentationv1.pdf`

2. 2016

2.1 April

Scope + Risk Management (2016-04-06 21:25)

Hello,

Scope 2nd Semester

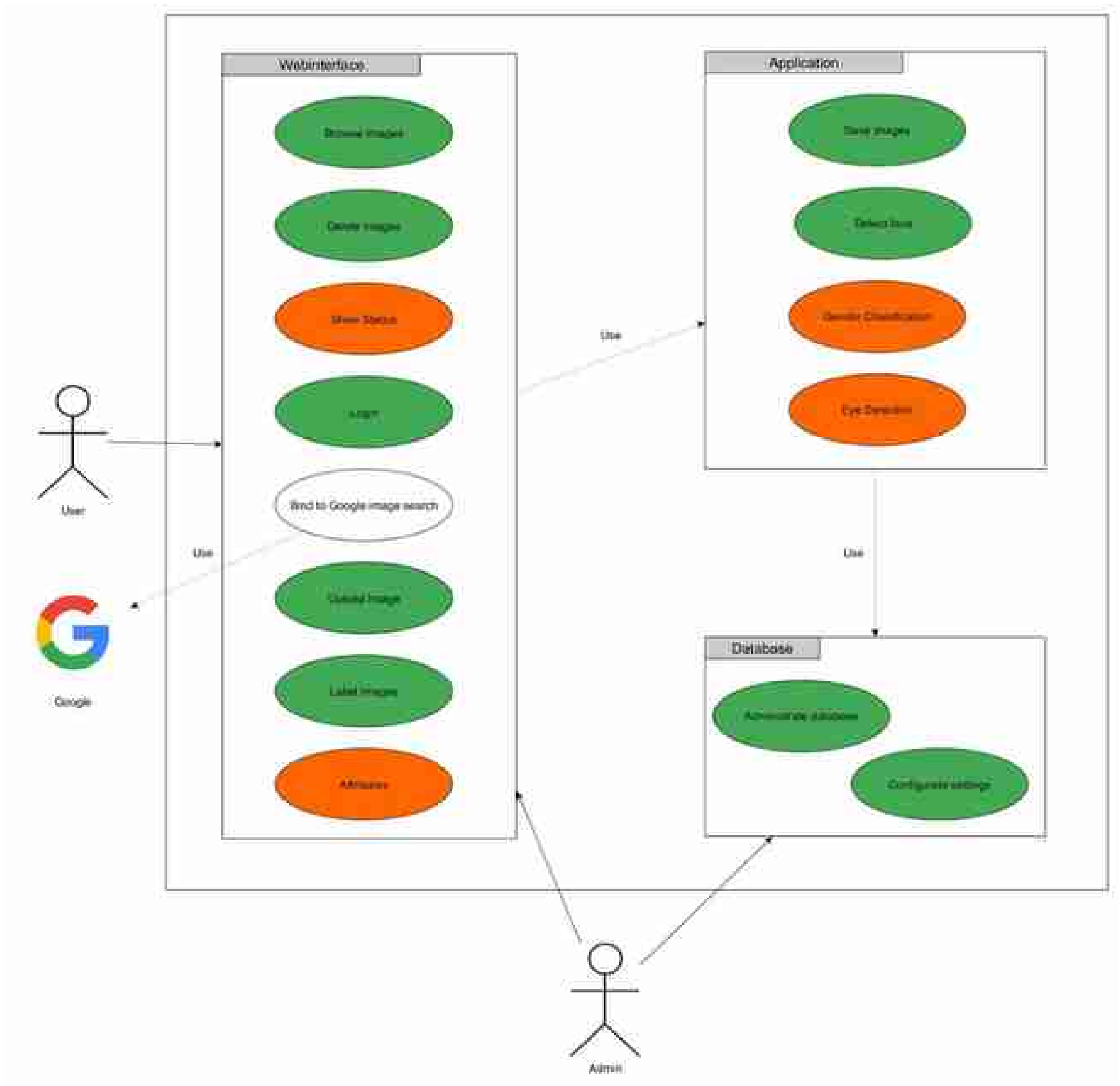
The following image shows our UC diagram. We finished the green ones last semester. The orange ones will be implemented this semester. You can find the Use Cases for this semester under the following links:

[1]Attributes

[2]Gender Classification

[3]Eye Detection

[4]Statistics



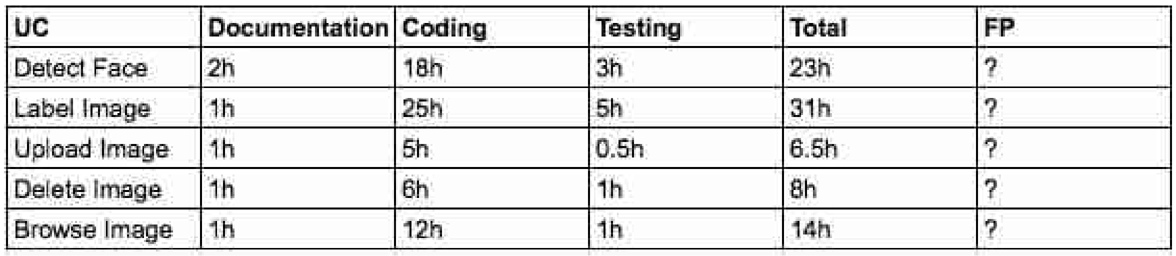
[5]https://raw.githubusercontent.com/sapaca FaceRecognition/Documentation/master/Use %20Cases/UCD.jpg

Risk Management

We used a GitHub wiki page to track our risks. We change the file locally with TextMate, it automatically creates a markdown table and pushes the changes on GH. You can find it here [6]https://github.com/sapacaFaceRecognition /Documentation/wiki/Risk-Management

Use Case Estimation

The following table shows our UC Estimation of the past semester.



UC	Documentation	Coding	Testing	Total	FP
Detect Face	2h	18h	3h	23h	?
Label Image	1h	25h	5h	31h	?
Upload Image	1h	5h	0.5h	6.5h	?
Delete Image	1h	6h	1h	8h	?
Browse Image	1h	12h	1h	14h	?

Best wishes,

caro from sapaca

1. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-%28S2%29:-Attributes`
2. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-(S2):-Gender-Classification`
3. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-(S2):-Eye-Detection`
4. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-(S2):-Statistics`
5. `https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/master/Use%20Cases/UCD.jpg`
6. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Risk-Management`

Simon (2016-04-10 18:59:30)
Hi SAPACA, nice to see something new in your blog ;). Why you using a scale to ten for your risk management, i think this is very undefinied. For example we using a column for the probable damage of the project to calculate them with the time to fix the problems. best wishes Simon from MathJunkies

caro340 (2016-04-10 22:08:22)
Hey Simon, thanks for your reply! We used the scale from 1-10 in our risk management because estimating an actual price seemed even more inaccurate to us. The problem we had was, that we didn't want to use any "fake" numbers in order to calculate the price of the damage. One risk could be that the implementation and documentation of a UC takes - let's say 2 weeks - more than expected. Quantifying a number here is - at least to us - impossible because the delay would perhaps just affect the grade. Best wishes, caro

Sebastian (Unveiled-Team) (2016-04-11 06:54:11)
Hi SAPACA, it's really hard to read your overall UC diagram, because the font size is very small. Your new Use Cases for this semester are looking good. In your Risk management table you have the entry "Balsamiq Lizenz". I would say, that this is not a general Risk for this project, because this is already known problem, you have to handle with, while implementing this project and not a "real Risk" for the hole Project. Best Regards Sebastian from Unveiled

caro340 (2016-04-11 07:09:24)
Hey sebastian, thanks for your reply. I will add a link to the overall UC diagram, wordpress resizes the image. You're right about the license.. I will update the table later. Best wishes, caro

Function Points (2016-04-17 15:00)

Hello follower,

we updated our project and with function points. With this points we can estimate our new use cases. We also updated our use case documentation.

https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/master/ FP %20UC/fp _diagram.PNG

https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/master/ FP %20UC/fp _table.PNG

Use Cases:

- [1]Detect Face
- [2]Label Image
- [3]Browse Image
- [4]Delete Image
- [5]Upload Image

stay tuned,

Your SAPACA-Team

1. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Detect-Face`
2. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Label-Image`
3. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Browse-Image`
4. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Delete-Image`
5. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Upload-Image`

Simon (2016-04-18 06:56:38)
Hey SAPACA, your graph for the fuction points of the use cases looks nice, but can you explain why there is a big difference to one of this use cases? best wishes Simon from MathJunkies

theratnar (2016-04-18 07:09:35)
Hello Simon, nice to see your interest in our project. Do you mean the difference between the use cases on the right side? Well for us it wasn´t such a big difference so we don´t mention it. kinds regards, Pascal - SAPACA

streboparsegyan (2016-04-18 07:26:57)
Hi Team Sapaca, you've done a FP calculation but a link to your UC documents would be nice, so it's easier to verify if you've entered the calculations into your UC documents. You've created a graph showing velocity and explained outliers. Good job. Best regards, Team strebo

theratnar (2016-04-18 07:40:49)
Hi Strebo-Team, thank you for your intrest in our project. The graph and the table are the important thinks, so we only show them in our post. kinds regards, Pascal - SAPACA

Testing (2016-04-22 11:16)

Hey guys,

in this blog entry we are going to show you our toy code (which was implemented with TDD) and our tests for the project.

TDD

I implemented a DFA (deterministic finite automaton (Deterministisch endlicher Automat)) in Java with TDD and recorded the programming process. You can check the video here: [1]https://youtu.be/IjGwMUdeqlg

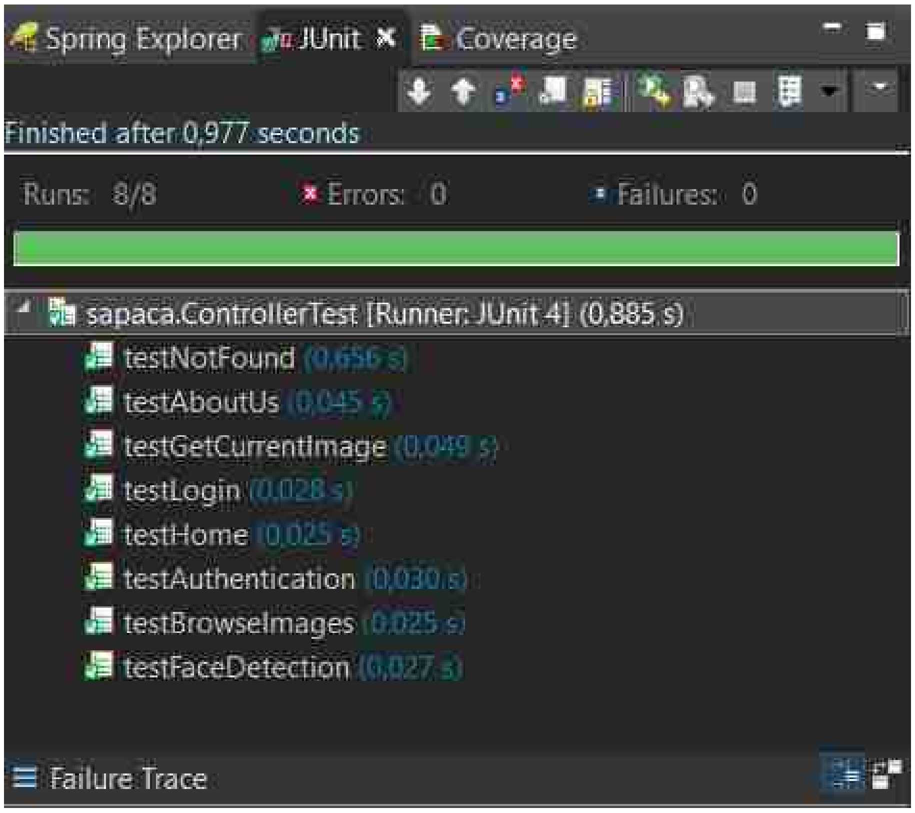
It was my first approach with TDD, I think I missed some "rules". The video is about 40min long. To verify the results I created a DFA which accepts floating point numbers. Sorry for the sloppy programming and some bullshit tests, it was almost 1 AM when I started programming.

Testing

For JUnit Tests in the Spring Tool Suite I used the spring-test and spring-webmvc framework as you can see on github: [2]build.gradle

You can find the test class also there: [3]ControllerTest.java

And last a screenshot of the executed tests:



```
Finished after 0,977 seconds

Runs: 8/8        Errors: 0        Failures: 0

sapaca.ControllerTest [Runner: JUnit 4] (0,885 s)
    testNotFound (0,656 s)
    testAboutUs (0,045 s)
    testGetCurrentImage (0,049 s)
    testLogin (0,028 s)
    testHome (0,025 s)
    testAuthentication (0,030 s)
    testBrowseImages (0,025 s)
    testFaceDetection (0,027 s)

Failure Trace
```

stay tuned,

your SAPACA Team

1. `https://youtu.be/IjGwMUdeqlg`
2. `https://github.com/sapacaFaceRecognition/FaceRecognition/blob/master/complete/build.gradle`
3. `https://github.com/sapacaFaceRecognition/FaceRecognition/blob/master/complete/src/test/java/sapaca/ControllerTest.java`

Larissa (2016-04-24 13:07:33)
Hey Caro, I think you really did a great job. It is clear to see, that you first implemented the tests you want to do and then the code. It helped me for my homework. Good Job. Regards, Larissa from Team MathJunkie

suddenh4x (2016-04-25 07:59:10)
Hey Larissa, thanks for your interest in our project! It's nice to hear that it has helped you :) Kind regards, SuddenH4X from SAPACA

Shark919 (2016-04-25 07:32:54)
Hey Sapaca, I really liked your idea to make a video about your TDD experience. It shows the process in a better way compared to some screenshots. It is also remarkable that you chose to develop a relatively big project compared to a simple calculator or

something similar. Good job! Regards, Christoph from appXpired

caro340 (2016-04-27 18:30:07)
Hey Chris, thanks for your reply and feedback. I couldn't sleep at night and I didn't want to write code for a calculator so I started programming something fun with TDD and recorded it (... guess who was late for class the next morning). Best wishes, caro

2.2 May

Fowler Refactor (2016-05-02 00:26)

Hey there,

here are the GitHub Repos to FowlerRefactor:

Sascha: https://github.com/SuddenH4X/Fowler

Pascal: https://github.com/Ratnar/Refactory

Caro: https://github.com/caroem/Fowler-Refactor

Best wishes,

Caro

Simon (2016-05-02 07:07:48)
Hey SAPACA, you do a nice job at the refactoring and comment clear every step of fowlers code. regards Simon from MathJunkies

caro340 (2016-05-02 07:17:38)
Thanks for the reply!

tom433 (2016-05-02 07:37:35)
Hey, all of you have done the work. Every step is committed and it is clear which step of Fowler you are using. Kind regards, Tom from tohemidevelopment

caro340 (2016-05-02 07:51:25)
Thanks for the reply!

Design pattern (2016-05-06 16:47)

Hey,

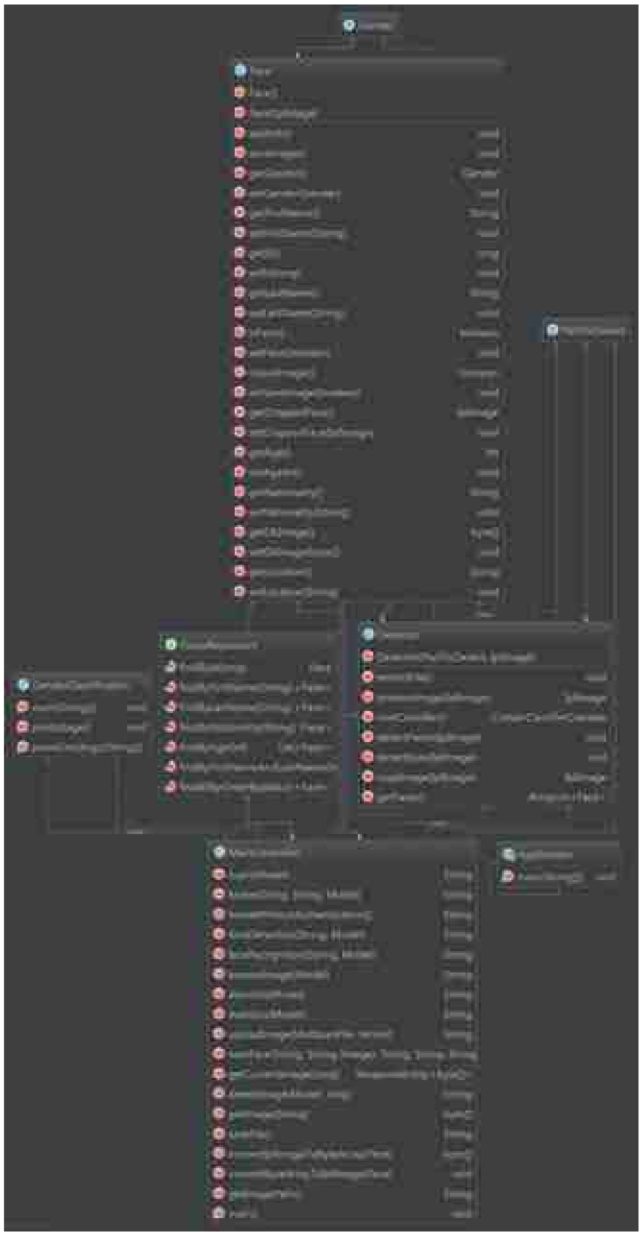
we implemented the factory design pattern to our project. The pattern is described here: [1]https://en.wikipedia.org/wiki/Factory _method _pattern

The factory pattern reduced the dependencies in our project and we now have a single point of control for multiple objects.

In the following we’re going to show you the before and after class diagramm and code.

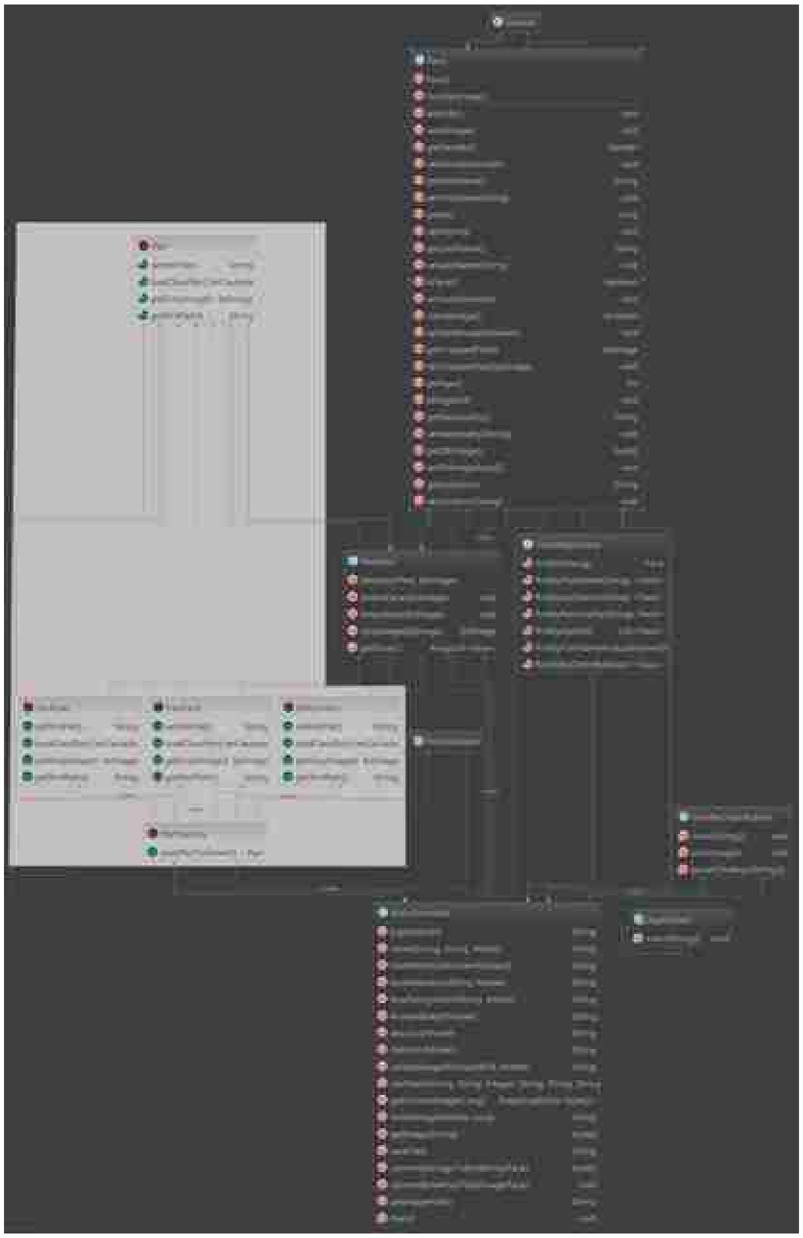
[2]Code before implementation

[3]Class diagramm before implementation



[4]Code after implementation

[5]Class diagramm after implementation (inverted area shows the implementation of the factory pattern):



We also added the pattern to our SAD [6]https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-A rchitecture-Documentation

Best wishes

1. `https://en.wikipedia.org/wiki/Factory_method_pattern`
2. `https://github.com/sapacaFaceRecognition/FaceRecognition/tree/EyeDetection/complete/src/main/java/sapaca`
3. `https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/master/Class%20Diagramm/ClassCiagram m_new.png`
4. `https://github.com/sapacaFaceRecognition/FaceRecognition/tree/master/complete/src/main/java/sapaca`

5. `https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/master/Class%20Diagramm/ClassDiagram_new_withpattern.png`
6. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Architecture-Documentation`

CodeLionX (2016-05-09 07:00:53)
Dear SAPACA-team, you have done a good job. You have chosen the Factory Pattern to apply to your code and described why it improves your design. Your class diagrams a very detailed and you have also marked the new classes. Greetings, CodeLionX from team Unveiled

theratnar (2016-05-09 07:09:52)
Hello CodeLionX from Unveiled, thank you for your comment. We made our documentation detailed make it understandable what we did. kinds regards, Pascal - SAPACA

Simon (2016-05-09 07:09:39)
Hey SAPACA, your Factory and Singleton-Pattern looks very good. Now there is a better overview about the code and and the structure in your project. The diagrams are very detailed and shows a clear overview. Kind regards, Simon from MathJunkies

theratnar (2016-05-09 07:11:16)
Hello Simon from MathJunkies, thank you for your positiv feedback. It was our plan to make it easily understandable ;) kinds regards, Pascal - SAPACA-Team

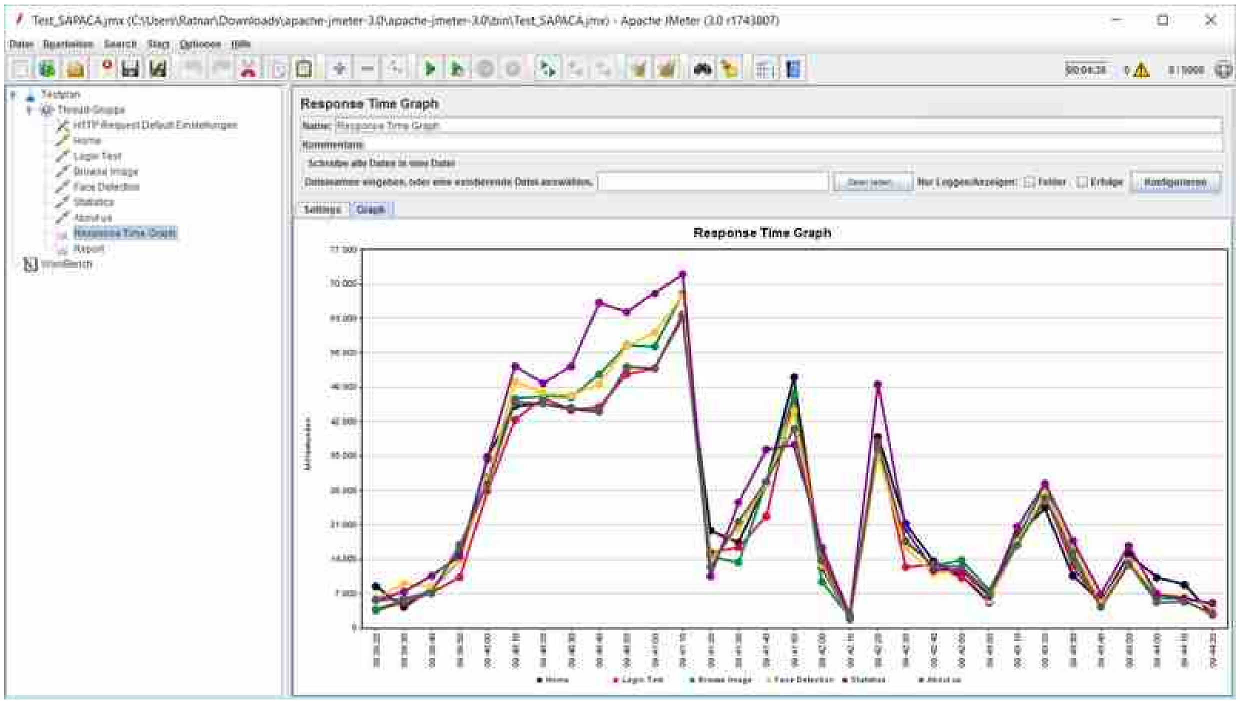
Test Coverage + Automatic Deployment (2016-05-23 06:54)

Hey everyone,

today we show you that our test coverage is over 40 %, just click on Coverall to see our coverage (on sonarqube it´s not accurate):
[1]Travis CI
[2]SonarQube
[3]Coveralls

And our [4]Test Document.
And [5]here can you see our badges.

Load Test:



To test our project under load I used a Tool calles Apache JMeter. With it you can define cases which should be tested. Before you start you decide how many threads you need and the number repeats. When you start the tool it creates the amount of threads you wanted and runs the defined cases. After that you get a graph like you can see in the picture above, where you can look up how much time a response needed.
For example at the time 9:41:10 you can see that the defined case ’Statistics’, the purple graph, needed 72.000 miliseconds to response.

Update:
We use Travis for CI. When we push something to GitHub Travis will automatically build the commit and run the tests. When the build fails, it automatically creates a Jira-Task on our Jira-Board. We created a script which creates the Jira ticket, you can check the script here and updated the travis.yml. You can check the updated files here:
[6]https://github.com/sapacaFaceRecognition/FaceRecognition/commit/b624cc 8f4e9cc565f-9543fc03428292b4791e84c
After the build succeeded on Travis, Travis automatically creates a .jar-file and releases it on GitHub. You can check the releases here: [7]https://github.com/sapacaFaceRecognition/FaceRecognition/releases
You can check our .travis.yml here: [8]https://github.com/sapacaFaceRecognition/FaceRecognition/blob/master/.travis.yml It shows the script for the automatic deployment. The jar is needed for the local installation. The installation guide is here:
[9]https://github.com/sapacaFaceRecognition/Documentation/wiki/Installati on-Guide

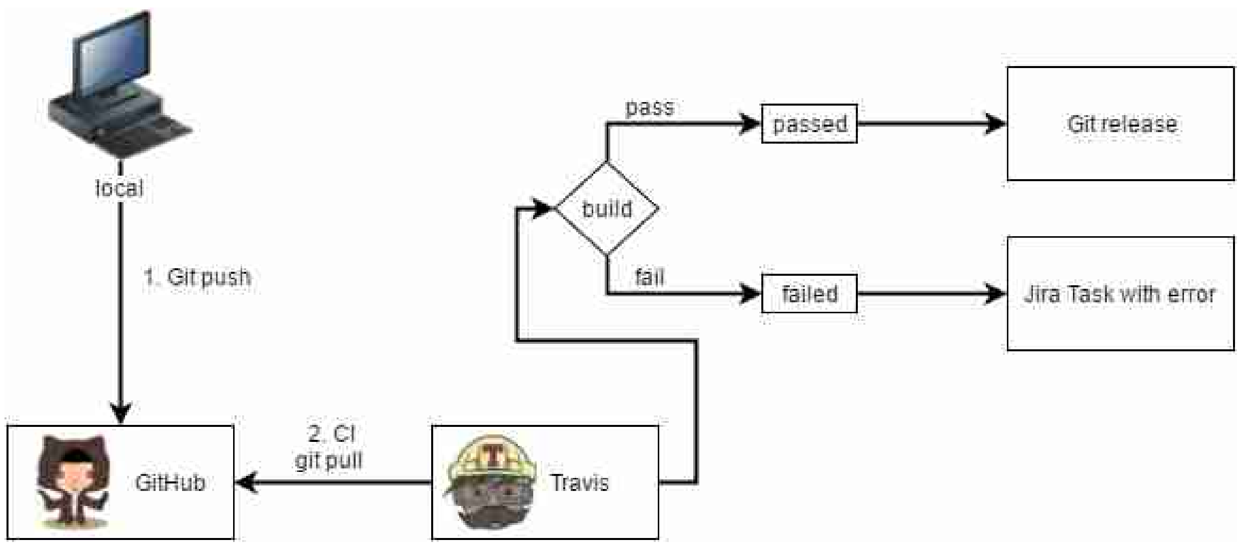


Diagram of our workflow

stay tuned,

Pascal - SAPACA

1. `https://travis-ci.org/sapacaFaceRecognition/FaceRecognition`
2. `http://193.196.7.25/overview?id=sapaca`
3. `https://coveralls.io/github/sapacaFaceRecognition/FaceRecognition?branch=master`
4. `https://github.com/sapacaFaceRecognition/Documentation/raw/master/Test%20Plan/Test%20Plan.docx`
5. `https://github.com/sapacaFaceRecognition/FaceRecognition`
6. `https://github.com/sapacaFaceRecognition/FaceRecognition/commit/b624cc8f4e9cc565f9543fc03428292b4791e84c`
7. `https://github.com/sapacaFaceRecognition/FaceRecognition/releases`
8. `https://github.com/sapacaFaceRecognition/FaceRecognition/blob/master/.travis.yml`
9. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Installation-Guide`

Simon (2016-05-23 08:09:31)
Hi SAPACA, your test coverage is very low, Will you add more test to get a better test coverage? And why your current status is red? kind regards, Simon from MathJunkie

theratnar (2016-05-23 08:22:46)
Hi Simon, this is just our coverage on sonarqube, and it´s not accurate. With sonarqube we have the problem that you don´t have a database there. That´s the reason with the status is red and the coverage low. kinds regards, Pascal - SAPACA

Digister (2016-05-23 08:39:34)
Hi team sapaca, you used coveralls.io as your tool. Also your test coverage is about 42 % which is pretty amazing. It would be nice if you linked your GitHub account directly to see that you got the badge. Furthermore you created and linked your test document. Well done :) Greetings, Team unveiled

theratnar (2016-05-23 08:49:17)
Hello unveiled, thank you for your interest in our project. Oh yeah, I´ll add that link immediately. kinds regards, Pascal - SAPACA

Digister (2016-05-30 11:44:26)
Hi team sapaca, You used travis as atomatically deploy tool. Also you atomatically build and test your code. Furthermore if the build fails a jira task is automatically generated. Well done :) Greetings, Team unveiled

caro340 (2016-05-30 11:46:06)
hey digimon, thanks!

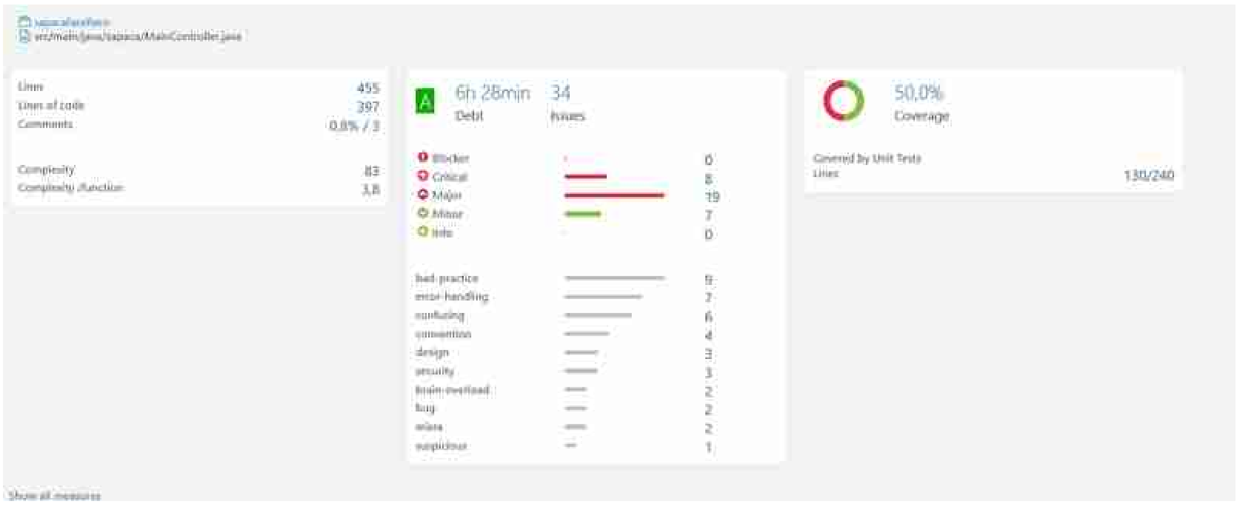
Simon (2016-05-30 11:46:36)
Hey SAPACA, good job with the automatic deployment and your testplan. But I miss the jira ticket, can you update your post with a screenshot. kind regards Simon from MathJunkies

caro340 (2016-05-30 11:50:54)
Hey simon, I'll add a screenshot later, thanks for the reply!

Metrics **(2016-05-29 19:55)**

Today we want to show you our code improvement. We used Codacy and Sonar and MetricsReloaded (IntelliJ Plugin).

As you can see on the screenshots the method statistics() in the class MainController was really big and had a complexity of 14 while the complete class had a complexity of 83:



src/main/java/sapaca/MainController.java

Lines	455
Lines of code	397
Comments	0.8% / 3
Complexity	83
Complexity /function	3,8

A 6h 28min Debt 34 Issues

Blocker	0
Critical	8
Major	19
Minor	7
Info	0

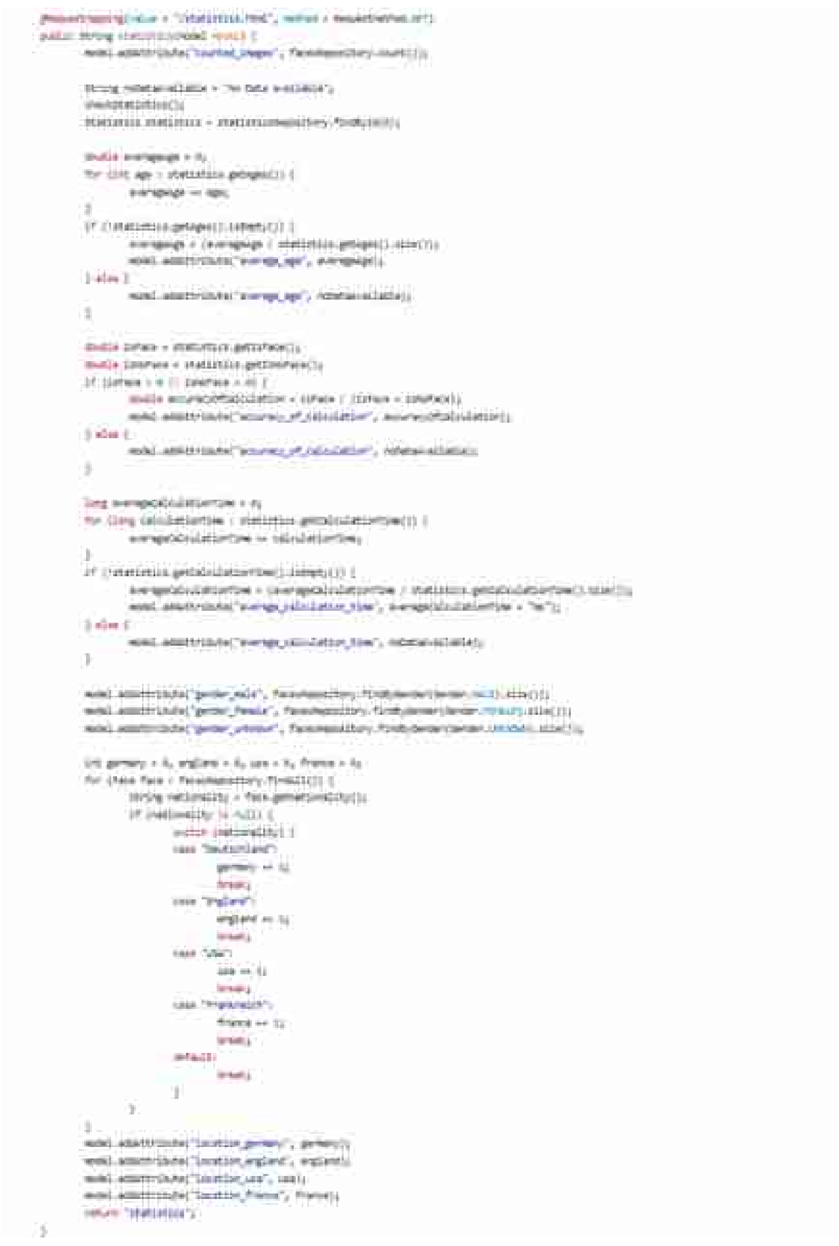
bad-practice	9
error-handling	7
confusing	6
convention	4
design	3
security	3
brain-overload	2
bug	2
misra	2
suspicious	1

50,0% Coverage

Covered by Unit Tests Lines	130/240

Show all measures

SonarQube: detail of MainController.java



```java
@RequestMapping(value = "/statistics.html", method = RequestMethod.GET)
public String statistics(Model model) {
        model.addAttribute("counted_images", faceRepository.count());

        String noDataAvailable = "No Data available";
        checkStatistics();
        Statistics statistics = statisticsRepository.findById(0);

        double averageAge = 0;
        for (int age : statistics.getAges()) {
                averageAge += age;
        }
        if (!statistics.getAges().isEmpty()) {
                averageAge = (averageAge / statistics.getAges().size());
                model.addAttribute("average_age", averageAge);
        } else {
                model.addAttribute("average_age", noDataAvailable);
        }

        double isFace = statistics.getIsFace();
        double isNoFace = statistics.getIsNoFace();
        if (isFace > 0 || isNoFace > 0) {
                double accuracyOfCalculation = isFace / (isFace + isNoFace);
                model.addAttribute("accuracy_of_calculation", accuracyOfCalculation);
        } else {
                model.addAttribute("accuracy_of_calculation", noDataAvailable);
        }

        long averageCalculationTime = 0;
        for (long calculationTime : statistics.getCalculationTime()) {
                averageCalculationTime += calculationTime;
        }
        if (!statistics.getCalculationTime().isEmpty()) {
                averageCalculationTime = (averageCalculationTime / statistics.getCalculationTime().size());
                model.addAttribute("average_calculation_time", averageCalculationTime + "ms");
        } else {
                model.addAttribute("average_calculation_time", noDataAvailable);
        }

        model.addAttribute("gender_male", faceRepository.findByGender(Gender.MALE).size());
        model.addAttribute("gender_female", faceRepository.findByGender(Gender.FEMALE).size());
        model.addAttribute("gender_unknown", faceRepository.findByGender(Gender.UNKNOWN).size());

        int germany = 0, england = 0, usa = 0, france = 0;
        for (Face face : faceRepository.findAll()) {
                String nationality = face.getNationality();
                if (nationality != null) {
                        switch (nationality) {
                        case "Deutschland":
                                germany += 1;
                                break;
                        case "England":
                                england += 1;
                                break;
                        case "USA":
                                usa += 1;
                                break;
                        case "Frankreich":
                                france += 1;
                                break;
                        default:
                                break;
                        }
                }
        }
        model.addAttribute("location_germany", germany);
        model.addAttribute("location_england", england);
        model.addAttribute("location_usa", usa);
        model.addAttribute("location_france", france);
        return "statistics";
}
```

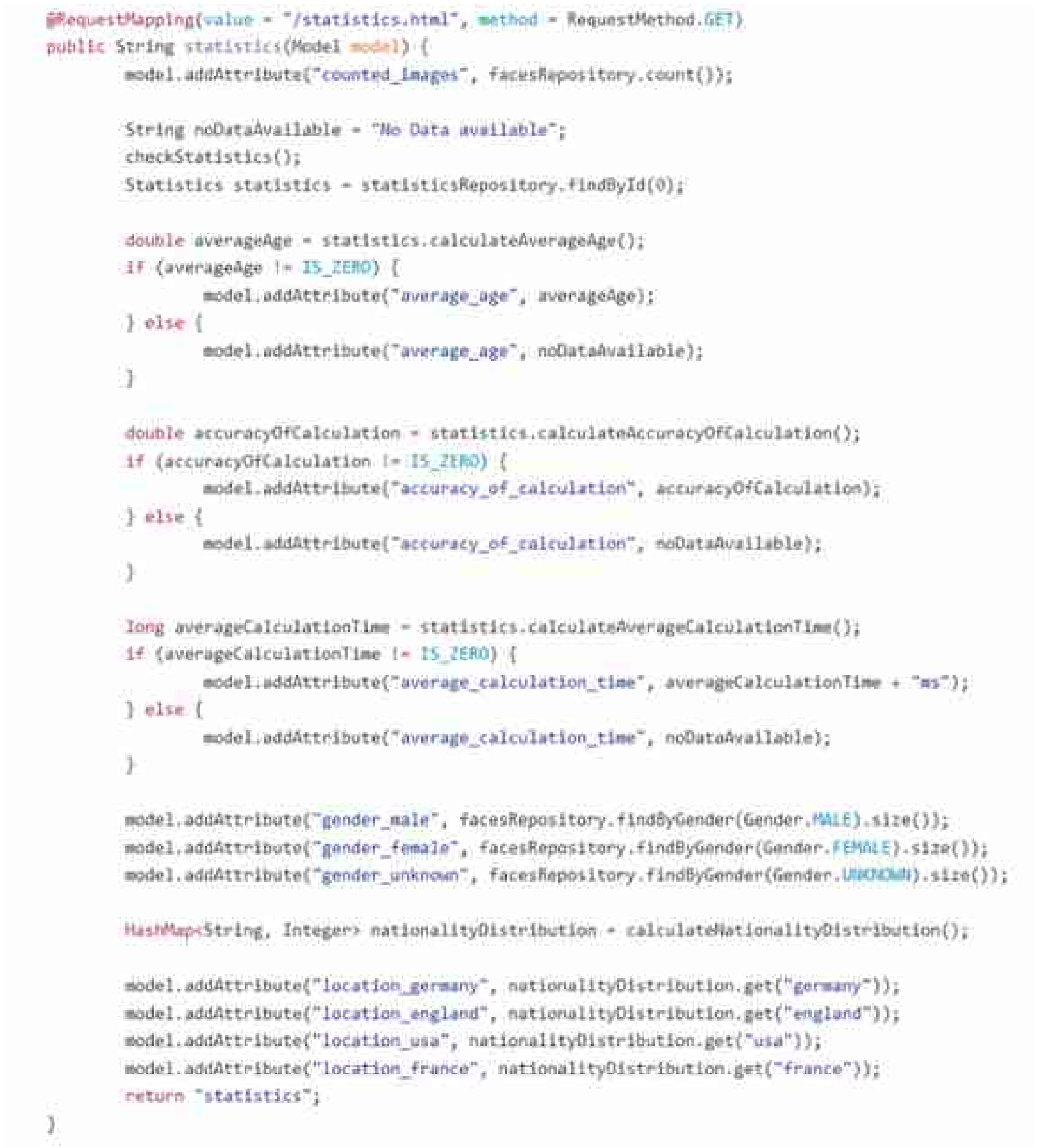
GitHub: method statistics() in MainController.java

For a better understanding of what we did you can see the source code on GitHub:

[1]MainController.java on Github before improvement

[2]Statistics.java on GitHub before improvement

So we decided to reduce the complexity of the method (and the class) and improve the readability at the same time. Therefor we exported the calculation of the statistics into the class Statistics and created one new method in the MainController class. So at the end the statistics method looks so:

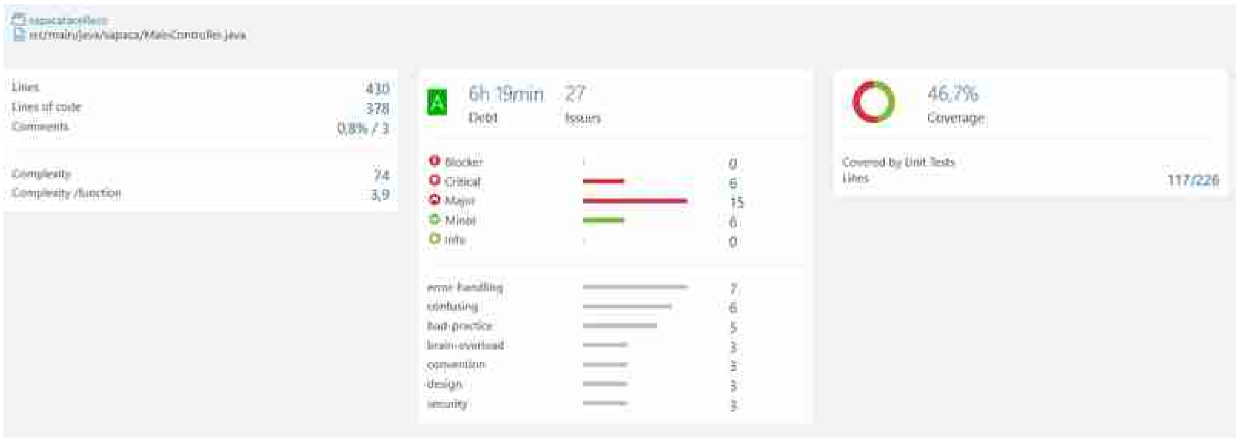


```java
@RequestMapping(value = "/statistics.html", method = RequestMethod.GET)
public String statistics(Model model) {
        model.addAttribute("counted_images", facesRepository.count());

        String noDataAvailable = "No Data available";
        checkStatistics();
        Statistics statistics = statisticsRepository.findById(0);

        double averageAge = statistics.calculateAverageAge();
        if (averageAge != IS_ZERO) {
                model.addAttribute("average_age", averageAge);
        } else {
                model.addAttribute("average_age", noDataAvailable);
        }

        double accuracyOfCalculation = statistics.calculateAccuracyOfCalculation();
        if (accuracyOfCalculation != IS_ZERO) {
                model.addAttribute("accuracy_of_calculation", accuracyOfCalculation);
        } else {
                model.addAttribute("accuracy_of_calculation", noDataAvailable);
        }

        long averageCalculationTime = statistics.calculateAverageCalculationTime();
        if (averageCalculationTime != IS_ZERO) {
                model.addAttribute("average_calculation_time", averageCalculationTime + "ms");
        } else {
                model.addAttribute("average_calculation_time", noDataAvailable);
        }

        model.addAttribute("gender_male", facesRepository.findByGender(Gender.MALE).size());
        model.addAttribute("gender_female", facesRepository.findByGender(Gender.FEMALE).size());
        model.addAttribute("gender_unknown", facesRepository.findByGender(Gender.UNKNOWN).size());

        HashMap<String, Integer> nationalityDistribution = calculateNationalityDistribution();

        model.addAttribute("location_germany", nationalityDistribution.get("germany"));
        model.addAttribute("location_england", nationalityDistribution.get("england"));
        model.addAttribute("location_usa", nationalityDistribution.get("usa"));
        model.addAttribute("location_france", nationalityDistribution.get("france"));
        return "statistics";
}
```

[3]MainController.java on GitHub after improvement

[4]Statistics.java on GitHub after improvement

The improvements leaded to a reduction of complexity in the MainController class from 83 to 74:



[illegible]
src/main/java/sapaca/MainController.java

Lines	430
Lines of code	378
Comments	0,8% / 3
Complexity	74
Complexity /function	3,9

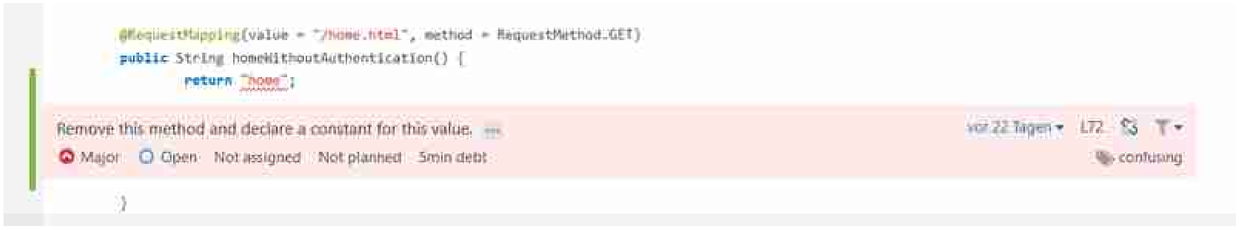
A 6h 19min Debt 27 Issues

Blocker	0
Critical	6
Major	15
Minor	6
Info	0
error-handling	7
confusing	6
bad-practice	5
brain-overload	3
convention	3
design	3
security	3

46,7% Coverage

Covered by Unit Tests Lines 117/226

We have not improved the following because it’s wrong. SonarQube didn’t recognized the Spring annotation @RequestMapping and therefor thinks the method is useless and can be replaced by a constant value:

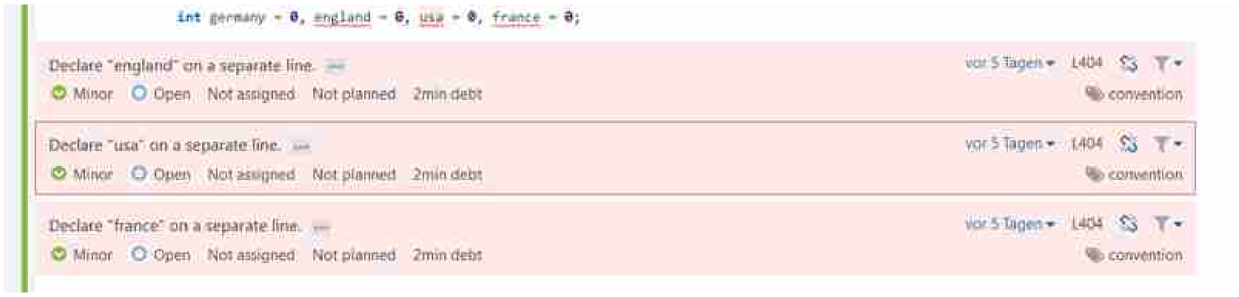


```
@RequestMapping(value = "/home.html", method = RequestMethod.GET)
public String homeWithoutAuthentication() {
        return "home";
```

Remove this method and declare a constant for this value. ... vor 22 Tagen L72
Major Open Not assigned Not planned 5min debt confusing

```
}
```

We have also ignored the warning that I should declare the 4 int variables on separate lines, because we don’t think that it’s usefull. If the variables are completly different and maybe even have different values the warning is okay, but in this case we think my implementation is better:



```
int germany = 0, england = 0, usa = 0, france = 0;
```

Declare "england" on a separate line. ... vor 5 Tagen L404
Minor Open Not assigned Not planned 2min debt convention

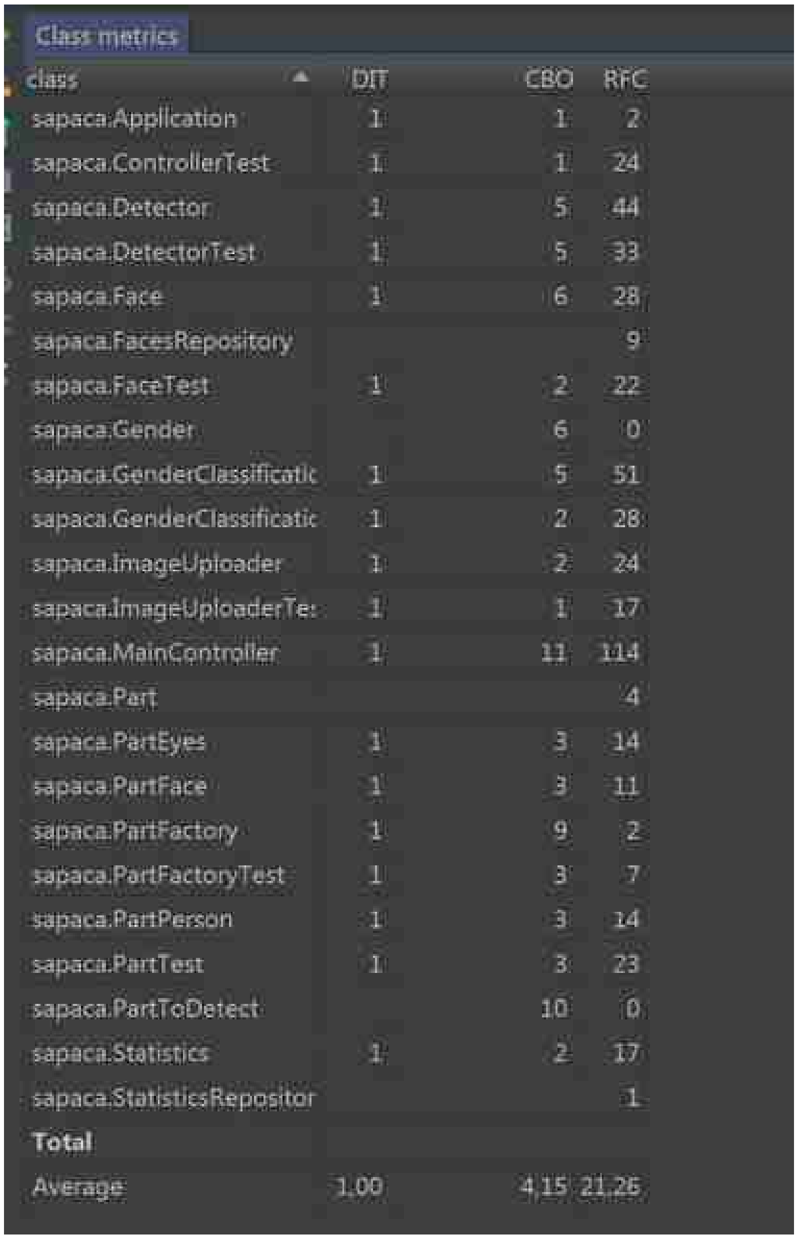
Declare "usa" on a separate line. ... vor 5 Tagen L404
Minor Open Not assigned Not planned 2min debt convention

Declare "france" on a separate line. ... vor 5 Tagen L404
Minor Open Not assigned Not planned 2min debt convention

UPDATE, 2nd metric:

We used the RFC (Response for Class) as the second metric to improve the testability and reduce complexity. A high RFC means high complexity. It can be hard to test the behaviour of the class and to debug problems since comprehending class behaviour requires a deep understanding of the potential interactions that objects of the class can have with the rest of the system.
We are going to improve the GenderClassification class with the RFC metric

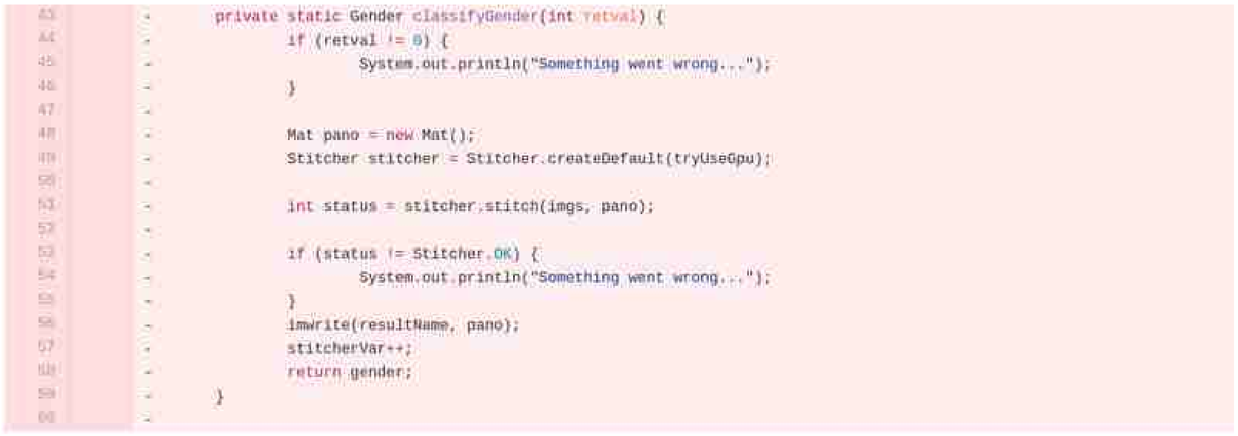
The following screenshot shows the RFC of the classes before the changes. The Genderclassification class has a RFC of 51. The screenshot shows two GenderClassification classes, ignore the second one since it’s just the test class.


Class metrics

class	DIT	CBO	RFC
sapaca.Application	1	1	2
sapaca.ControllerTest	1	1	24
sapaca.Detector	1	5	44
sapaca.DetectorTest	1	5	33
sapaca.Face	1	6	28
sapaca.FacesRepository			9
sapaca.FaceTest	1	2	22
sapaca.Gender		6	0
sapaca.GenderClassificatic	1	5	51
sapaca.GenderClassificatic	1	2	28
sapaca.ImageUploader	1	2	24
sapaca.ImageUploaderTe:	1	1	17
sapaca.MainController	1	11	114
sapaca.Part			4
sapaca.PartEyes	1	3	14
sapaca.PartFace	1	3	11
sapaca.PartFactory	1	9	2
sapaca.PartFactoryTest	1	3	7
sapaca.PartPerson	1	3	14
sapaca.PartTest	1	3	23
sapaca.PartToDetect		10	0
sapaca.Statistics	1	2	17
sapaca.StatisticsRepositor			1
Total			
Average	1.00	4.15	21.26

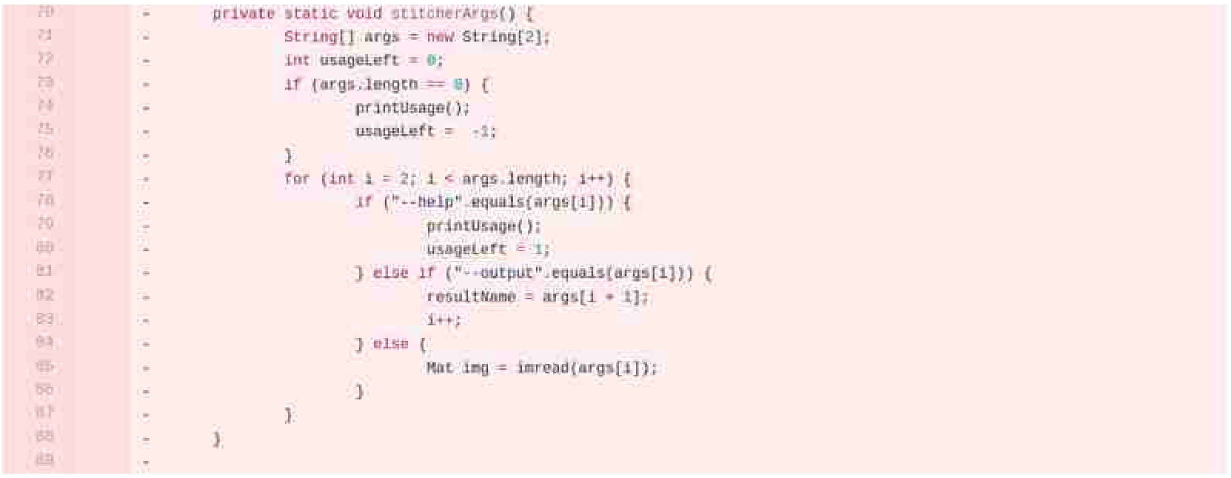
[5]https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/ 150a4a841a9-187898aa0e8ec9fcdc20a04ad58c5/Metrics/rfcBefore.png

To improve the RFC we first removed methods which were used for the gender classification in our first try of implementing the gender classification. The methods are useless because our current implementation takes care of the steps the method executes. The following screenshots show the removed methods:



```
-       private static Gender classifyGender(int retval) {
-               if (retval != 0) {
-                       System.out.println("Something went wrong...");
-               }
-
-               Mat pano = new Mat();
-               Stitcher stitcher = Stitcher.createDefault(tryUseGpu);
-
-               int status = stitcher.stitch(imgs, pano);
-
-               if (status != Stitcher.OK) {
-                       System.out.println("Something went wrong...");
-               }
-               imwrite(resultName, pano);
-               stitcherVar++;
-               return gender;
-       }
-
```

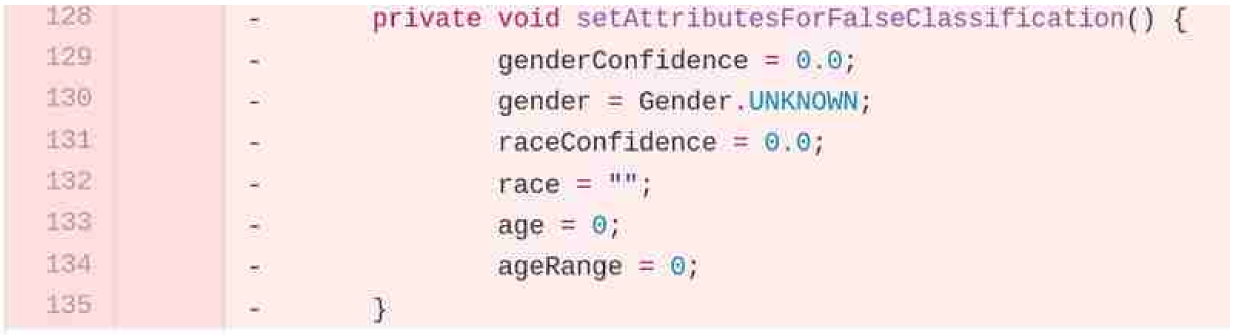
[6]https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/ 6984572fccd-8c6d59ffe66361d8cbea51114909f/Metrics/rfcRemove1.png



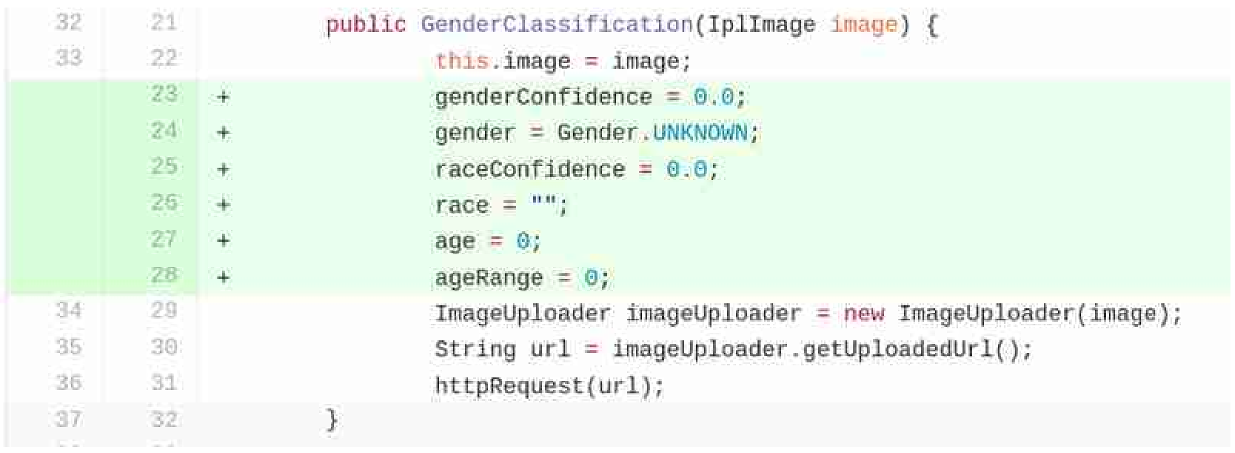
```
-       private static void stitcherArgs() {
-               String[] args = new String[2];
-               int usageLeft = 0;
-               if (args.length == 0) {
-                       printUsage();
-                       usageLeft =  -1;
-               }
-               for (int i = 2; i < args.length; i++) {
-                       if ("--help".equals(args[i])) {
-                               printUsage();
-                               usageLeft = 1;
-                       } else if ("--output".equals(args[i])) {
-                               resultName = args[i + 1];
-                               i++;
-                       } else {
-                               Mat img = imread(args[i]);
-                       }
-               }
-       }
-
```

[7]https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/ 6984572fccd-8c6d59ffe66361d8cbea51114909f/Metrics/rfcRemove2.png

Then we removed the setAttributesForFalseClassification method, which sets all attributes to 0 in case of a failing classification and put the logic into the constructor. The following screenshots show these steps.

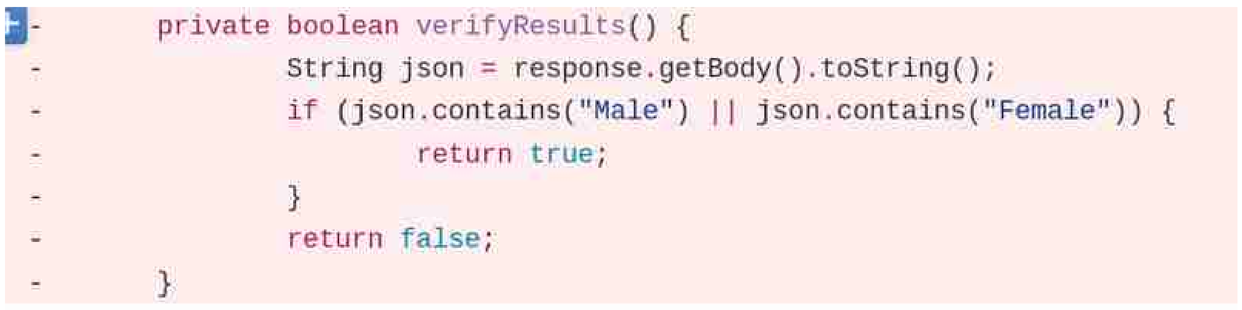


```
-       private void setAttributesForFalseClassification() {
-               genderConfidence = 0.0;
-               gender = Gender.UNKNOWN;
-               raceConfidence = 0.0;
-               race = "";
-               age = 0;
-               ageRange = 0;
-       }
```

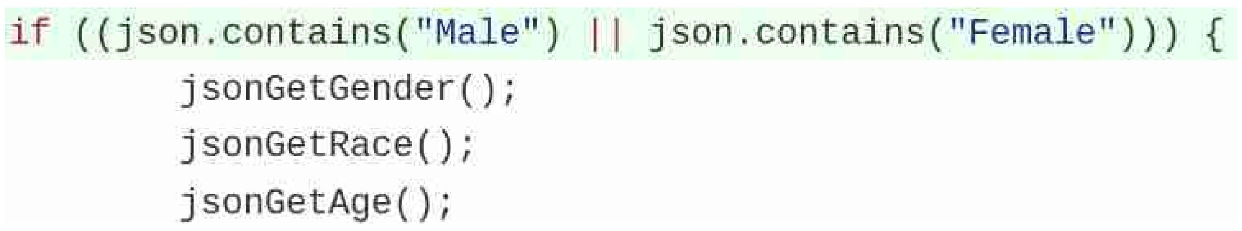


```
        public GenderClassification(IplImage image) {
                this.image = image;
+               genderConfidence = 0.0;
+               gender = Gender.UNKNOWN;
+               raceConfidence = 0.0;
+               race = "";
+               age = 0;
+               ageRange = 0;
                ImageUploader imageUploader = new ImageUploader(image);
                String url = imageUploader.getUploadedUrl();
                httpRequest(url);
        }
```

Then we replaced the verifyResults() method with an if-statement. The whole method is small and just verifies that the classification didn't fail. The following screenshots show these steps.

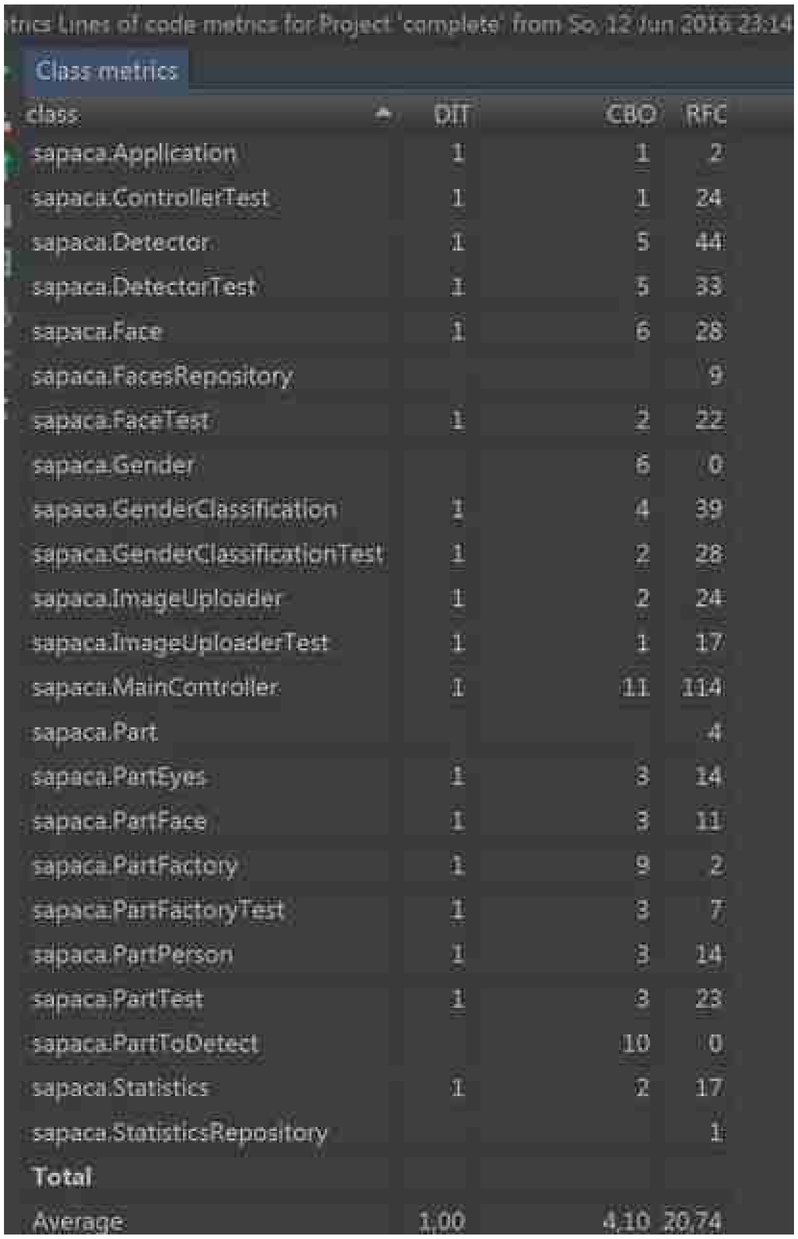


```
-       private boolean verifyResults() {
-               String json = response.getBody().toString();
-               if (json.contains("Male") || json.contains("Female")) {
-                       return true;
-               }
-               return false;
-       }
```



```
if ((json.contains("Male") || json.contains("Female"))) {
        jsonGetGender();
        jsonGetRace();
        jsonGetAge();
```

We reduced the rfc to 39 with these steps. You can see all the changes in the [8]Git Commit



trics Lines of code metrics for Project 'complete' from So, 12 Jun 2016 23:14

Class metrics

class	DIT	CBO	RFC
sapaca.Application	1	1	2
sapaca.ControllerTest	1	1	24
sapaca.Detector	1	5	44
sapaca.DetectorTest	1	5	33
sapaca.Face	1	6	28
sapaca.FacesRepository			9
sapaca.FaceTest	1	2	22
sapaca.Gender		6	0
sapaca.GenderClassification	1	4	39
sapaca.GenderClassificationTest	1	2	28
sapaca.ImageUploader	1	2	24
sapaca.ImageUploaderTest	1	1	17
sapaca.MainController	1	11	114
sapaca.Part			4
sapaca.PartEyes	1	3	14
sapaca.PartFace	1	3	11
sapaca.PartFactory	1	9	2
sapaca.PartFactoryTest	1	3	7
sapaca.PartPerson	1	3	14
sapaca.PartTest	1	3	23
sapaca.PartToDetect		10	0
sapaca.Statistics	1	2	17
sapaca.StatisticsRepository			1
Total			
Average	1,00	4,10	20,74

stay tuned,
Your SAPACA-Team

1. `https://github.com/sapacaFaceRecognition/FaceRecognition/blob/1f3b2420e7f126d715eac1f4dc7d51ace05674dc/complete/src/main/java/sapaca/MainController.java`
2. `https://github.com/sapacaFaceRecognition/FaceRecognition/blob/5500b15184afa84aa98c903e8a638a1540f6670e/complete/src/main/java/sapaca/Statistics.java`
3. `https://github.com/sapacaFaceRecognition/FaceRecognition/blob/master/complete/src/main/java/sapaca/MainController.java`
4. `https://github.com/sapacaFaceRecognition/FaceRecognition/blob/master/complete/src/main/java/sapaca/Statistics.java`
5. `https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/150a4a841a9187898aa0e8ec9fcdc20a04ad58c5/Metrics/rfcBefore.png`
6. `https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/6984572fccd8c6d59ffe66361d8cbea51114`

909f/Metrics/rfcRemove1.png

7. https://raw.githubusercontent.com/sapacaFaceRecognition/Documentation/6984572fccd8c6d59ffe66361d8cbea51114 909f/Metrics/rfcRemove2.png

8. https://github.com/sapacaFaceRecognition/FaceRecognition/commit/fae3518183ccbc0333a032d0f099c2faf3e376a2

Simon (2016-05-30 07:12:28)
Hey SAPACA, your metrics looks very good and the maincontroller is really small in comparison to the last version. It is also better, that you have ignored the warnings, I think with your implementation you have a better overview. Kind regards Simon from MathJunkies

caro340 (2016-05-30 08:57:50)
Hey junkie, thanks for the reply! best wishes, caro

Sebastian (2016-05-30 08:48:15)
Hi SAPACA, your improvements, which are based on the metrics-tool, are looking good. It's hard to read the data from the screenshots, because they don't have a good resolution. Is the run of the metrics-tool part of your CI-process? Best regards, Sebastian from Unveiled

caro340 (2016-05-30 08:57:15)
Hey sebastian, thanks for the reply. I'll add links for the uploaded pictures. Running the metric-tool is part of our CI-process. best wishes, caro

2.3 June

Final Post (2016-06-16 09:28)

Hey followers,

we´re at the end of our project. Thanks for your interesting. You can get our runnable Project [1]here.

- [2]GitHub FaceRecognition
- [3]Documentation
- [4]Final Presentation
- [5]Software Achitecture Documention
- [6]Software Requirments Specification
- [7]Test Plan
  - [8]SonarQube
  - [9]Travis CI
  - [10]Coveralls

- [11]Jira
  - [12]Burndown 1
  - [13]Burndown 2
- Use Cases:
  - [14](S2) Attributes
  - [15](S2) Eye Detection
  - [16](S2) Gender Classification
  - [17](S2) Statistics
  - [18]Browse Image
  - [19]Delete Image
  - [20]Detect Face
  - [21]Label Image
  - [22]Label Image
- [23]Risk Plan
- [24]Gant Chart
- [25]Function Points

your SAPCA-Team!

1. `https://github.com/sapacaFaceRecognition/FaceRecognition/releases/tag/untagged-295f961bf21634239a38`
2. `https://github.com/sapacaFaceRecognition/FaceRecognition`
3. `https://github.com/sapacaFaceRecognition/Documentation`
4. `https://docs.google.com/presentation/d/1UgXAXoivEBnC_S5VHx2nuLb8ThALy2ZA3VZwlXNtXWA/edit?usp=sharing`
5. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Architecture-Documentation`
6. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Software-Requirements-Specification`
7. `https://github.com/sapacaFaceRecognition/Documentation/raw/master/Test%20Plan/Test%20Plan.pdf`
8. `http://193.196.7.25/overview?id=5659`
9. `https://travis-ci.org/sapacaFaceRecognition/FaceRecognition`
10. `https://coveralls.io/github/sapacaFaceRecognition/FaceRecognition?branch=master`
11. `http://193.196.7.27:8080/projects/SAVI/summary`
12. `https://github.com/sapacaFaceRecognition/Documentation/raw/master/Jira/jira_burndown_1.PNG`
13. `https://github.com/sapacaFaceRecognition/Documentation/raw/master/Jira/jira_burndown_2.PNG`
14. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-(S2):-Attributes`
15. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-(S2):-Eye-Detection`
16. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-(S2):-Gender-Classification`
17. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case-(S2):-Statistics`
18. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Browse-Image`
19. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Delete-Image`
20. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Detect-Face`
21. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Label-Image`
22. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Use-Case:-Upload-Image`
23. `https://github.com/sapacaFaceRecognition/Documentation/wiki/Risk-Management`

24. `https://github.com/sapacaFaceRecognition/Documentation/raw/master/MS%20Project/FaceRecognition_Projektplan`
`nung.pdf`
25. `https://github.com/sapacaFaceRecognition/Documentation/raw/master/FP%20UC/fp_all.pdf`

BlogBook v0.9,
LaTeX 2$_\varepsilon$ & GNU/Linux.
`https://www.blogbooker.com`

Edited: June 16, 2016