



# Project: Beetle

13. maj 2015

qzj710 - 121095 - Enes Golic  
rpc308 - 070493 - Yunus Emre Okutan  
cbh239 - 250594 - Casper Lützhøft Christensen  
mhb558 - 250795 - Tor-Salve Dalsgaard

Instructor - Kasper Passov

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Abstract . . . . .	4
<b>2</b>	<b>FACTOR</b>	<b>5</b>
2.1	Functionality . . . . .	5
2.2	Application-domain . . . . .	5
2.3	Conditions . . . . .	5
2.4	Technology . . . . .	5
2.5	Objects . . . . .	5
2.6	Responsibility . . . . .	5
<b>3</b>	<b>Program Specifications</b>	<b>6</b>
3.1	Functional and Non-Functional Requirements . . . . .	6
3.2	Use Case Model . . . . .	7
3.3	Specific Use Cases . . . . .	8
3.4	Class-diagram . . . . .	9
3.5	BCE . . . . .	10
3.6	Sequence-diagram . . . . .	11
<b>4</b>	<b>System Design</b>	<b>13</b>
4.1	Current Progress . . . . .	13
4.2	Summary . . . . .	13
<b>5</b>	<b>Testing</b>	<b>14</b>
5.1	Search-engine test . . . . .	14
<b>6</b>	<b>Interaction and Design</b>	<b>15</b>
6.1	Design . . . . .	15
<b>7</b>	<b>Internal Cooperation</b>	<b>19</b>
7.1	Summary . . . . .	19
<b>8</b>	<b>Litterature Reviews</b>	<b>21</b>
8.1	Designing for usability - J.D. Gould and C. Lewis . . . . .	21
8.2	A Rational Design Process - David Lorge Parnas and Paul C. Clements . . . . .	22

<b>9</b>	<b>Annexes</b>	<b>23</b>
9.1	Testing . . . . .	23
9.2	Github Log . . . . .	23
9.3	Changelog . . . . .	25
9.4	Timeline . . . . .	25

# 1 Introduction

## 1.1 Abstract

Our project is a search-engine attached to a database containing insects as entries, and is to be provided to our client the University of Hamburg (from here on abbreviated "UH"). German law dictates that a university must make all research conducted publicly available to anyone seeking the knowledge – it is in this context our project has been requested. Through research UH's entomology department has established a catalogue of information on different specimen of insect, including their names, species, subspecies, genus and pictures of its anatomy and location. We will be using this information as attributes in our database, which will feature the specimen of insect as the entry. The search-engine we create will then allow a visitor to input a search-term, which will be tried against any of the attributes in the database and return all entries with an attribute matching. In addition, the project will feature an advanced search-engine, which will allow a visitor to input search-terms to be matched against specific attributes, rather than against every attribute. Furthermore, UH must be able to update the database themselves, we will therefore provide an administrator control panel – protected by a login system - where entries can be added and deleted. Because some of the specimen of insects recorded in the database are potentially threatened by extinction, UH has also requested that the insect's location not be available to everybody. We will therefore hide this information from ordinary visitors, but allow the administrator to create temporary users that can view it. The project will be created using PHP, Javascript and MySQL, and will feature the search-engine, the advanced search-engine, the temporary user copies of the 2 former mentioned as well as an administrator control panel.

## **2 FACTOR**

### **2.1 Functionality**

Allow visitors to use the 2 search-functions, basic and advanced, to search through a database of insect-entries with data collected by the University of Hamburg. Allow the administrator with login-credentials to remove or add new entries to the database. Allow the administrator to create temporary users for aspiring scientists or enthusiasts, so that they can view information that cannot be publicly available (location of specimen facing potential extinction). Allow temporary users to use a log into the website and view all information on every insect in the database.

### **2.2 Application-domain**

The product is to be available to everybody seeking to use it in the pursuit of knowledge – this means students, researchers, professors and anonymous visitors alike. Only the administrator however will be able to make changes to the database, and only users who are given login credentials will be able to view an insect's location.

### **2.3 Conditions**

This product is being developed as part of the course PKSU at the University of Copenhagen. It is therefore by us completely voluntarily developed and no gain on our side apart from experience is being received. The product must be freely available, on the University of Hamburg's website, to anyone who wishes access to the information stored within. The product is not to be used to generate any profit whatsoever, its only purpose is the distribution of knowledge.

### **2.4 Technology**

The product is developed through the server-side language PHP, client-side language Javascript and the database-management-language MySQL. As such, nothing but a computer and an internet-connection has been required in order to develop it. Due to this, the product can run in any browser and on any application (computer/phone) with an internet connection.

### **2.5 Objects**

The product consists of a webpage containing the search-engine. From the standard search-engine you can either, as an anonymous viewer, enter the advanced search-engine or, as an employee, login to the editing interface using provided login credentials.

### **2.6 Responsibility**

The responsibility of the product is, through a search-engine, to ensure free and public access to research conducted by the University of Hamburg.

## 3 Program Specifications

### 3.1 Functional and Non-Functional Requirements

If we refer to the abstract outlining the program's specifications, we get a general feel of the architecture of the software and how it is going to work. Drawing on those points we can, for the sake of overview, specify functional and non-functional requirements for the program.

#### Functional Requirements

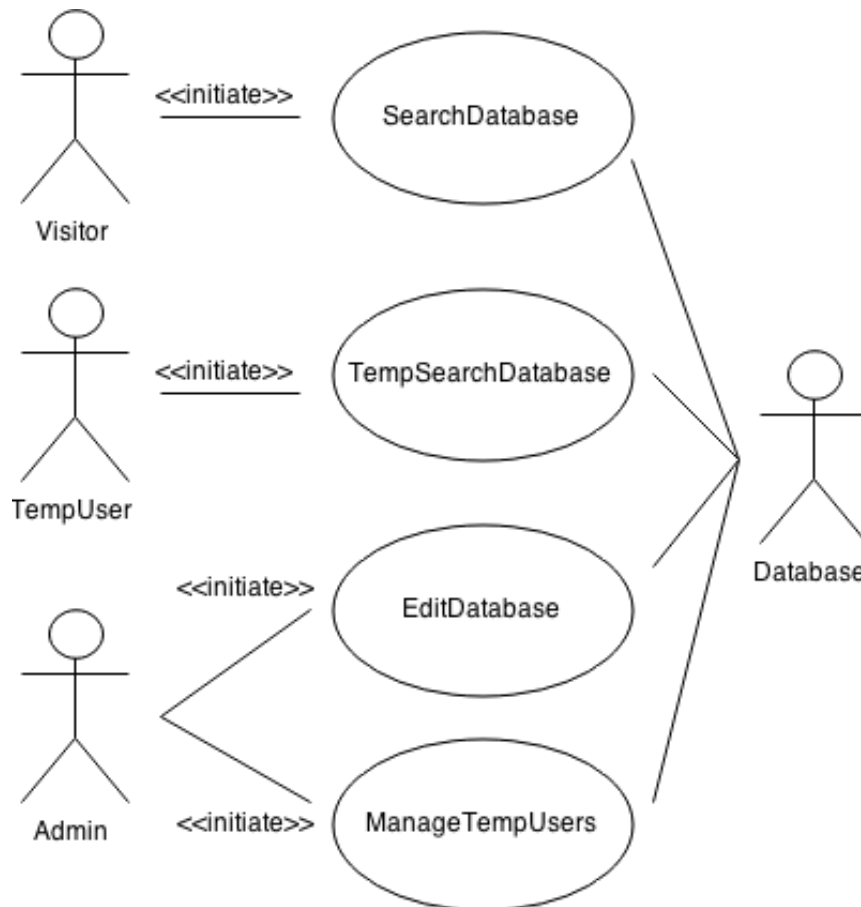
- The product must offer a search-option that returns any and every entry that contains an attribute that matches the search input.
- The product must offer a search-option that allows matching of specific search-inputs to specific attributes and return any and every entry that matches the terms in their specific attributes.
- The database must allow the creation of new entries, editing of existing entries and deletion of existing entries.
- The database must be unable to be edited by anyone save the authorised employees.
- A fitting message must be displayed if no matching entries are found.
- The database must allow the creation of new entries and deletion of existing entries.
- The database must only be editable by the administrator of the site.
- The administrator must be able to create temporary users that can view an insect's location, which is otherwise not displayed.

#### Non-Functional Requirements

- The search-engine must be developed in PHP so it can run on their website in any browser with no additional requirements.
- The method of editing the database must be simple enough for people without pre-existing IT skills to use it.
- The product in its finished state must look nice and fit with the colour-scheme already present on the University of Hamburg's website.

## 3.2 Use Case Model

---



### SearchDatabase

Visitors can use the search-engine by inputting the term(s) they wish to try against the database and the matching results displayed. Searching the database requires no logging in. **TempSearchDatabase**

Temporary users can use the search-engine that will also display the location of the entry. They are required to login to access this functionality.

### EditDatabase

An administrator can edit the database by adding or removing entries from it, as long as he is logged into the control panel.

### ManageTempUsers

An administrator can also create logins for temporary users, so that they can view information that is not supposed to be publicly available.

He can also delete these logins when deemed appropriate.

---

The figure above is a use-case model, depicting the possibilities and permissions of the various different users the site can encounter.

For instance any visitor will be able to use the frontpage search-engine. If granted a login by the administrator, you can log in as a temporary user and use a search-engine that will also display the location.

The administrator can edit the database and also manages the temporary users.

### 3.3 Specific Use Cases

With the use case model created, we have an overview of the functional architecture of the website. With that in mind, we can explore some of the specific cases a bit further:

Use Case Name	SearchDatabase
Participating actors	Initiated by Visitor
Flow of events	1. A visitor enters the website. 2. The visitors enters the search-term into the search-field and presses send. 3. The matching results are displayed.
Entry condition	None
Exit conditions	•Anonymous exits the site. •Anonymous submits a new search term.

The above use case is the ordinary search of the database. It showcases in detail the specifics of the use case by displaying who initiates it, who, if any, it communicates with, the flow of the case itself as well as any conditions necessary. We will now also inspect 2 other use cases:

Use Case Name	ManageTempUsers
Participating actors	Initiated by Administrator Communicates with Database Communicates with TempUser
Flow of events	1. The Administrator enters the website. 2. The Administrator logs into the control-panel using his login credentials. 3. The Administrator creates or removes a temporary user.
Entry condition	Must have Administrator credentials.
Exit conditions	•Administrator exits the site. •Administrator logs out of control-panel.

This is the case of the Administrator either creating or deleting a temporary user. This is, of course, only available to the administrator with credentials.

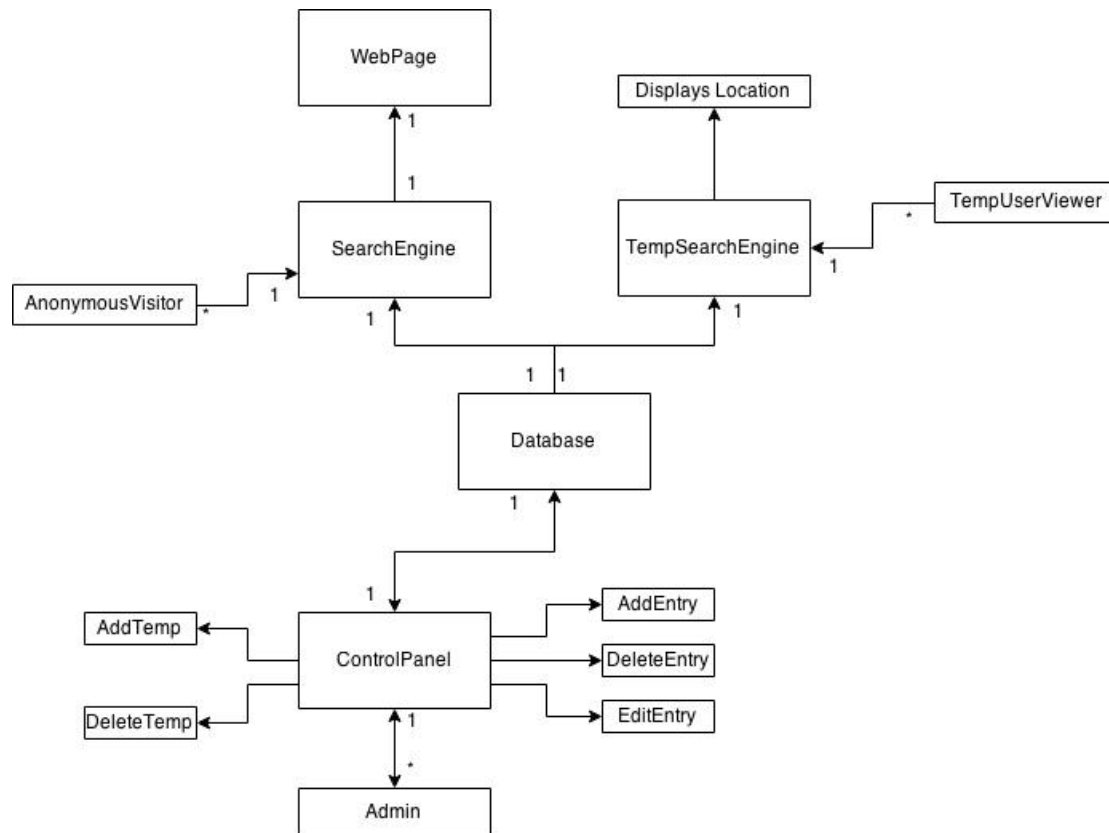
Use Case Name	EditDatabase
Participating actors	Initiated by Administrator Communicates with Visitor Communicates with Database Communicates with TempUser
Flow of events	1. The Administrator enters the website. 2. The Administrator logs into the control-panel using his login credentials. 3. The Administrator makes the desired changes to the database
Entry condition	•Must have Administrator credentials.
Exit conditions	•Administrator exits the site. •Administrator logs out of control-panel.

As with the case preceeding it, administrator credentials are also necessary to edit the database.



### 3.4 Class-diagram

In order to grant an overview of the software-architecture, a class-diagram has been created:



The class diagram shows the elements of our website and how they are connected. Our Webpage has two search engines, one for simple search statements, and for a more advanced approach an advanced search. These two can be used by an anonymous visitor, but in addition the admin is using the same search engine in his control panel. The search engines are connected to a database which holds all the entities, can create, edit or remove entries in the database, hence the pictures that pop up when searched for. The database can't do these things by itself, that is why we have made a control panel in which the admin can change the entries, but the admin can also add guest users to the system. Since our project revolves around special butterflies and insects, our goal is to preserve the nature by not giving out details to anyone. Some of the insects are endangered, meaning that the university now have the opportunity to permit certain users access, by creating a temporary user for them in the control panel. These users can only be added or deleted. The admin is connected to all of this, and as we stated earlier the admin takes use of the search engine to make it easier for himself to check for a specific species, subspecies or genus. Instead of looking through of a list a maybe 100 entries, he can search for the specific genus he wishes to look for. In order to make as many entries available for the public, we have added a box which the owners can tick, and telling the database that it can only be accessed by a temporary user created in the system. In our class diagram we have stated this special incident as being a parametre for the temporary users: "Display Location".

### 3.5 BCE

To provide an easy reference-point for the different objects in our product and their respective responsibilities, a BCE has been created to effectively sort those objects into their representative categories.

<b>B</b>	<b>C</b>	<b>E</b>
<i>Buttons</i>	<i>Search</i>	<i>Entries</i>
<i>Textfields</i>	<i>AdvancedSearch</i>	<i>Images</i>
<i>Headline</i>	<i>TempUserSearch</i>	
<i>Background</i>	<i>TempUserAdvSearch</i>	
	<i>ImageClick</i>	
	<i>Edit</i>	

#### **Boundary:**

Everything a user can see is a boundary object. In our case the buttons, also the text fields, the headline and the grey background.

#### **Entity:**

The entities here are all the information from the database, as well as the images that represent a specific set of information (an entry).

#### **Control:**

The search-button grabs the text input into the search-field and matches the term against the database. The advanced search-button is similar except that it has 4 fields in which a user can match a term against specific attributes.

Clicking on an image will take the user to a full-scale version of the image clicked.

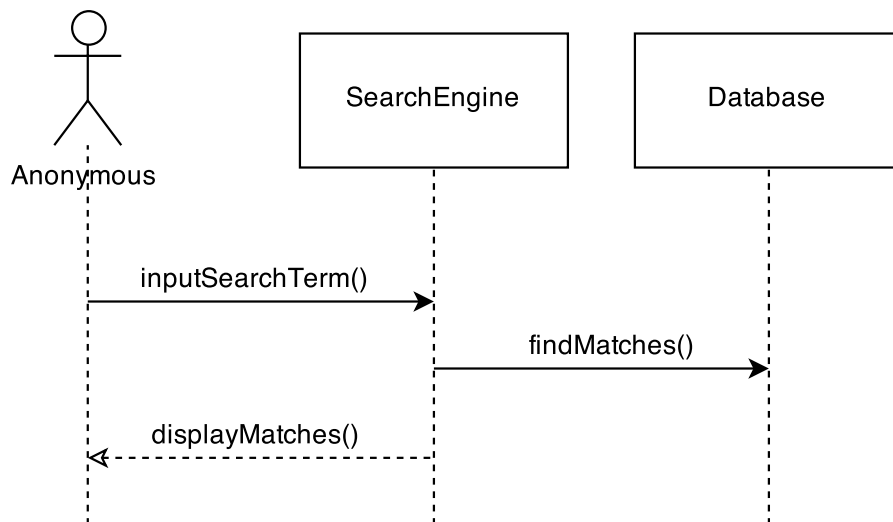
Finally we have the edit button that allows an employee to change the entries in the database, or add/remove entries. These are all control objects as they control one or more parts of the website's functionality.

In some cases when the user is allowed additional access through having a username and password, the user can then view the TempUserSearch and TempUserAdvSearch, which allow them to see the endangered objects.

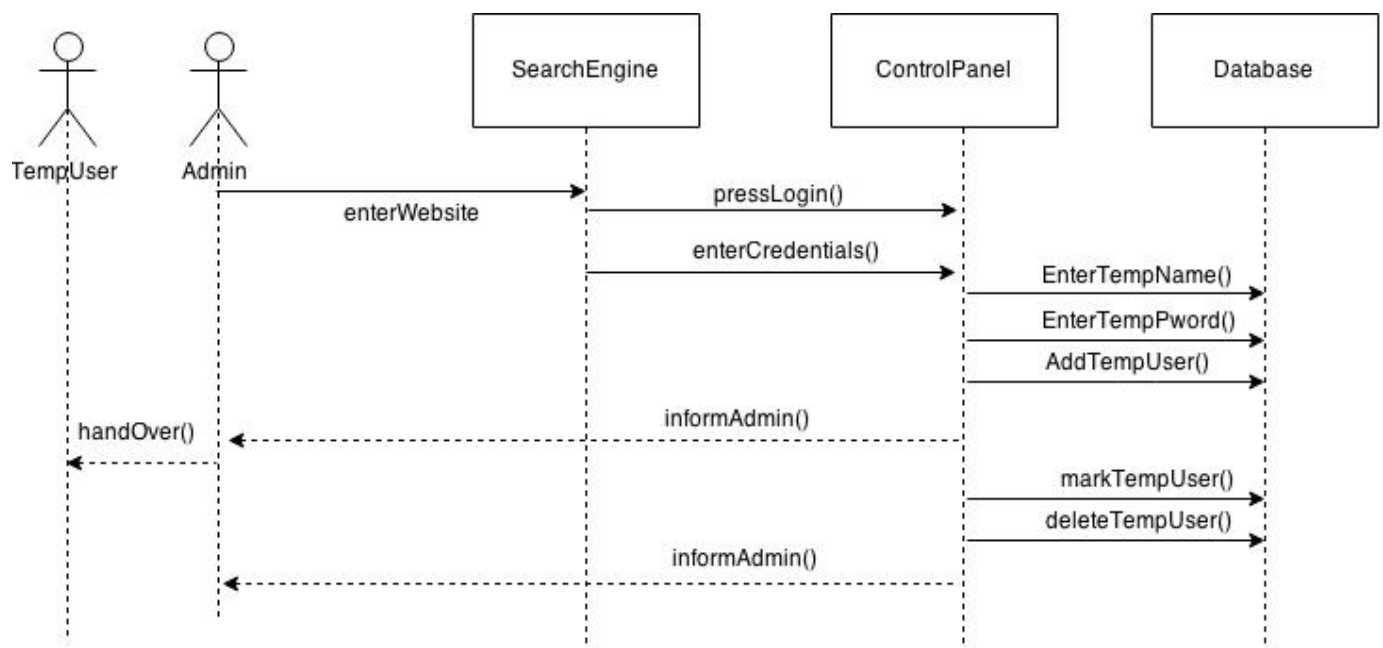
### 3.6 Sequence-diagram

We've already previously established different use cases relating to the project. Through these we can create sequence-diagrams illustrating them:

#### Use Case: SearchDatabase



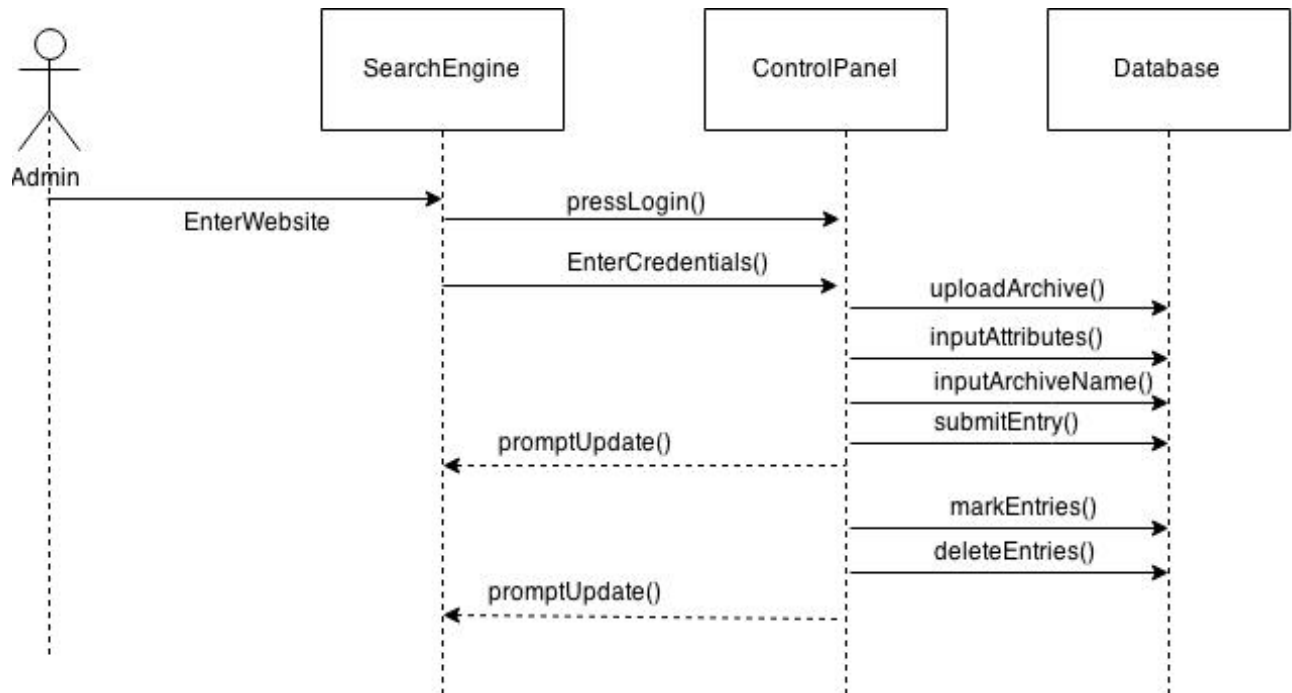
#### Use Case: ManageTempUsers



The idea is that this sequence diagram works with the recently added tempuser. The admin enters the searchengine, where he enters the controlPanel. He can add the name, password and in the end

the temporary user to the database. The admin is then informed of the newly added user, which he then can handover to the temporary user.  
The admin can then later mark the created temporary user, and delete it.

### Use Case: EditDatabase



## 4 System Design

### 4.1 Current Progress

The Prototype of our product can be found and accessed at: <http://echiever.de/ProjectBeetle/Prototype3/prototype3.php>

### 4.2 Summary

At the current-stage of our product everything is fully functional.

The database consists of insect-entries, with order, family, genus, species, subspecies, description and the name of the .zip archive containing the high-resolution zoomify image of the insect as attributes. Both the search-function and the advanced search-function are fully functional, and a method of displaying all existing entries in the database has been added (by simply searching for no term). The database is now editable by either adding or deleting entries.

To add an entry you must now first upload an archive containing the zoomify image of an insect, and thereafter enter the appropriate information into its respective attributes, as well as enter the name of the archive which you uploaded. Deleting an entry is done simply by marking a checkbox and clicking a delete-button. At this point, all functional requirements have been fulfilled, and our work going forward is:

- Improve the design to better match that of the University of Hamburg's existing website
- Implement any further wishes the client may have

## 5 Testing

### 5.1 Search-engine test

Testing the current edition of the product has gone to show that every of our newly implemented changes work as intended. We did however locate one problem, which is that logging out currently does appropriately delete the cookie responsible for checking whether you're logged in or not, and simply pressing 'return' after having logged out, will not re-prompt you for your credentials.

As for the current search, it now works with our updated database that includes the zoomify images. The current search-able words in the normal search-engine are:

- testFamily
- Sterrha
- Eilicrinia

As for testing the advanced search-engine, the following combination of terms and letters allow for full showing of its functionality.

<i>Order</i>	<i>Contains</i>	<i>stOr</i>
<i>Family</i>	<i>Startswith</i>	<i>test</i>
<i>Genus</i>	<i>Endswith</i>	<i>rha</i>
<i>Species</i>	<i>Contains</i>	<i>cord</i>

More in-depth descriptions of the various conducted tests are available as annexes at the end of the report.

## 6 Interaction and Design

### 6.1 Design

In order to showcase the unit-interface of the product, a series of screenshots have been provided.



Above is the frontpage of the website before any interaction has been done.



Blank-search to display all entries.




No matching entries.

## Project: Beetle

Leave field blank to display all entries

Sterrha

Advanced search




Order: testOrder

Family: testFamily

Genus: Sterrha

Species: adherbariata

Subspecies: teuteburgensis



Matching entry (entries).

## Advanced Search

Order	Contains ▼	<input type="text"/>	<input type="checkbox"/>
Family	Contains ▼	<input type="text"/>	<input type="checkbox"/>
Genus	Contains ▼	<input type="text"/>	<input type="checkbox"/>
Species	Contains ▼	<input type="text"/>	<input type="checkbox"/>

The advanced search-engine.

## Advanced Search

Order	Contains ▼	asd <input type="text"/>	<input type="checkbox"/>
Family	Contains ▼	<input type="text"/>	<input type="checkbox"/>
Genus	Contains ▼	<input type="text"/>	<input type="checkbox"/>
Species	Contains ▼	<input type="text"/>	<input type="checkbox"/>

No Entry


No matching entries.



### Advanced Search

Order	Contains ▼	testOrder	<input type="checkbox"/>
Family	Contains ▼		<input type="checkbox"/>
Genus	Contains ▼		<input type="checkbox"/>
Species	Contains ▼		<input type="checkbox"/>




Order: testOrder

Family: testFamily

Genus: Sterrha

Species: adherbariata



Order: testOrder

Family: testFamily

Genus: Eilicrinia


Species: cordiaria

Matching entry (entries).

**LoginBOX**

<b>Username</b>	<b>Password</b>
<input style="width: 90%;" type="text"/>	<input style="width: 90%;" type="password"/>



Close LoginBOX

Login function.


## Project: Beetle

*Leave field blank to display all entries*

Advanced search



*Incorrect Username or Password*

Unsuccesful login.

**Add a new insect to the Website**

Order:   
Family:   
Genus:   
Species:   
Subspecies:   
Description:   
Foldername:

Upload Zoomify-folder as zip-file:  Der er ikke valgt nogen fil



Order: testOrder Family: testFamily Genus: Sterrha Species: adherbariata Subspecies: teuteburgensis ☐



Order: testOrder Family: testFamily Genus: Eilicrinia Species: cordiaria Subspecies: astigmara ☐

[Logout](#)

Editing page displayed after succesful login.

## 7 Internal Cooperation

### 7.1 Summary

#### 22nd of April:

Come thus far, the cooperation of the University of Hamburg is satisfactory.

We have had a total of 2 "official" correspondences with our client in order to formally request server-access and the information we will be needing for the database. Furthermore we have been in contact with the University of Hamburg's Entomology department's representative after every work-session we have had, to present the current state of the product to him. All the correspondences have been fulfilling and we will be gaining access to the server as soon as their IT-department have created a user for us.

Furthermore, we will be receiving the necessary data for creating the proper database the 22th of April, after which we'll have everything we need.

The client also expressed satisfaction regarding the current state of our product, although a desire to have it looking more akin to the current layout of their website (which is only natural given that this is a prototype).

Our current way of work is based on us setting up workdays and meetings depending on necessity. This means that we, rather than using a gridlocked schedule, have been deciding on days to meet when the need was expressed.

For organisation we have been using a group created in Skype for communication as well as a repository in Github for file-access.

We have been internally communicating the progress of the product along the way and have also internally decided which partials have had the most importance at a given point in time, as to ensure we are all on the same page and that deadlines will be met.

So far we are making great progress in the development of the product itself, and we will soon have all the available means to ensure a succesful deliverance of a working product.

Our effort in organising meetings and working days has been less than ideal and together with our non-gridlock scheduling style have caused us to have many subsequent days of work, causing more of a burden than what is needed.

In order to make our developing from here on more efficient, more emphasis will be placed on proper scheduling, in order to spread out the workload and easen the burden on the entire group. This will also allow more time for quality control.

**13th of May:**

At this stage the product is fully functional and everything that the University of Hamburg has desired, has been implemented.

In regards to our work we continued largely with the approach discussed in the summary from the 22nd of April, in the way that we have been setting up days on which we met and attempted to get as much work done as possible.

While we previously discussed the disadvantages to this approach, the progress we had made did not leave a lot of additional work to be completed, and therefore using a predetermined schedule was deemed unnecessary.

Our cooperation internally has been more than satisfactory, as each member of the group has been pulling their own assigned work-load, as well as having been constructive with input in regards to the whole project.

We have been working more closely with the University of Hamburg, showing them the development of the product underway, and have been continuously involving them in the whole process, so that they could provide their own input on the finished product.

The result is, that all their requests have been met satisfactorily, and that we are now almost ready to ship the finished product.

## 8 Literature Reviews

### 8.1 Programming as theory building - Peter Naur

In 1985 Peter Naur gave a brilliant example on how programming and software research should be approached. The general idea is that programming is theory. This theory is the backbone of all future programs. The point is not to code something specific, the code is not the important thing. Code changes because of demands, optimization etc. It might be the best way to code for 10 years, but after that who knows? Peter Naur said in his abstract that *"(...) it is concluded that the proper, primary aim of programming is, not to produce programs, but to have the programmers build theories of the manner in which the problems at hand are solved by program execution."*

The real goal of the software development crew, is not to code something, but to understand the concept behind it. Understanding it rather than just coding something that works makes it easier to solve and develop new solutions. If a theory around your software is created it will make optimization, expansion or modifications easier. Peter Naur even claims that code standing by itself is dead, even though it may be useful for some situations. One quote specifically explains the point of programming:

*"Revival of a program is the rebuilding of its theory by a new programmer team".*

Naur also says that documentation is not an appropriate mechanism to transmit knowledge in software projects. In general the idea is that Peter Naur believes that software development isn't about developing software, but more about developing a shared understanding of the software development.

What we can take with us in our project is that we need to understand what we are doing. We need to think about all the modifications, optimizations and requests that the owner might call for. That is why we simply cannot write some code that works, but we don't understand the principle behind it. We need to have a backbone, something we can lean on when we the need for modifications hits us. Since our project was fairly simple we automatically counted in the need for future development, making it easy to expand. But this is not mostly due to our work, but because the project doesn't have many strings to play on. It's a search engine, that works with a specific algorithm that retrieves elements from a database. The idea here is pretty simple, and so is the theory behind it. One thing we can consider is not analyzing what a program is doing, we should analyze how does it take part and affect in the environment it is executed. How the users feel, how different systems work with it. Programming is about working in teams, and in order for a team to work together, all must understand the same theory behind the project.

## 9 Annexes

### 9.1 Testing

Test plan:

We want to test the general functionality of our product. The searchfunction is the most important part of the product, followed by the option to add an entry to the database. When those two functions work we can add more functions to the product, like the advanced search and the deletion of an entry in the database.

Test specification:

It should not be hard to test these functions, because we easily can test if the search gives the right output. The add and delete functionality should also be easy to test, because of its easy of use.

Testing the search could happen with the mentioned words and letters from section 5. Also we need to test the add entry and delete entry. Here we create a new entry, fitting to a Zoomify-image, and see if it works as intended. After that we could delete that entry again.

Test incident report:

Right now is there one bug in the code: If you login and logout, and go back in the browser you will go back to the edit page. Some could argue, that it is not that bad, because of the user has been logged in before already, but its still a bug that has to be fixed.

Test summary report:

We have tested our product a lot. But there is not much to test; a searchfield, an advanced-search, an add entry function and a delete entry function. Maybe you could count the add Zoomify-image in too, but we count it under the add function.

Until now did all tests work out fine. We could use the searches and get a working result. We could add and delete entries. And we could also add a new Zoomify-image-folder to the server.

### 9.2 Github Log

---

a2f101f (HEAD, origin/master, origin/HEAD, master) Newest version of report 2.

0e78c2f timeline/new changelog

3628967 Reviews

af1ee59 okok

696e499 Added new changelog and code

7757964 Current state of the report.

f7b07df ChangeLog 20-21.04

298c0df cleanUP

5370d72 Mjallo virkelig ny

b0eee35 Mjallo ny

3045fee Mjallo

5726df9 Merge remote-tracking branch 'origin/master'

db797ee Fixed it!

8ee517f try tis

ebe615a Edited the report.  
339fa41 Report + Pictures  
0d21845 Delrapport 1- rettelse 1  
c129a2c Killefilter  
b0d9a30 Scheduleplan  
c477a2f Skillmatrix  
5751436 review of the review  
6d0c506 Echo delrapport  
0df006f Feedback til Project Echo  
f2cdb65 .tex-filer 4 you  
783af0a Redone the enviroment  
16bfea5 English corrections/changes  
5c8451d Changed the name  
dd63eb0 Hermaen  
6b0723b delrapport 1 assignments  
be9faa6 1.6/1.8  
0787272 Clean up!  
00a6714 delete double  
4f6a322 Projekt Etablering  
d20a468 Rewrote the description  
96050db Statements and Project Agreement Definition  
da46259 Hello  
4c5850d Initial Software Architecture (updated)  
fa8824f Initial Software architecture  
906e4ca Initial SPMP  
0789c6f Introductory/Problem statement  
acb4324 7.1 Deployment Diagram  
bfcbd73 sup  
9efec63 Revert "HEJ YUNUS"  
13eef81 HEJ YUNUS  
0851690 Hello Casper

---

The code at this point in time have been developed in cooperation with the entire group through meetings at the school, where we worked on the code on only a single computer. As such, the Github log will not bear trace of editing of the code. The line 696e499 however shows the addition of the beta-version's code to the project's folder.

### 9.3 Changelog

Date	Name	Change
20 – 04 – 15	YO/TD/EG/CL	<i>Added seachfield and -button</i>
20 – 04 – 15	YO/TD/EG/CL	<i>Added database and entries</i>
20 – 04 – 15	YO/TD/EG/CL	<i>Added editlink</i>
21 – 04 – 15	YO/TD	<i>Added Family-, Genus- and Species-names form the database to the webpage</i>
21 – 04 – 15	YO/TD	<i>Added picture of the found insect(s)</i>
21 – 04 – 15	TD	<i>Added CSS-code to shine the webpage a bit up</i>
21 – 04 – 15	TD	<i>Updated the database, so it actually shows an actual insect</i>
21 – 04 – 15	TD	<i>Added Order-name</i>
01 – 05 – 15	TD	<i>Added Zoomify pictures and the description of the insect</i>
01 – 05 – 15	YO	<i>Added login</i>
11 – 05 – 15	TD/CL	<i>Added option, for an employee, to add a new entry into the database</i>
11 – 05 – 15	TD/CL	<i>Added option, for an employee, to delete an old entry from the database</i>
11 – 05 – 15	CL	<i>Added feature, that displays all entries, when the search-field is blank</i>
11 – 05 – 15	YO/EG	<i>Added functionality for the advanced search</i>

### 9.4 Timeline

Initial custommer contact - **12th of March** - The custommer explained the desired functionality for the search-engine and their expectations of the outcome of the project.

Report 1 - **21st of March** - The custommer received a copy of our first report, in order for them to view our progress and method of work. The custommer here expressed their desire for a more advanced option of using the search-engine. The exact specifics of what this should contain have not yet been stated, but they said they'd get back to us. They also expressed their desire for a drop-down menu showcasing all the entries in the database, in order to give the visitors a better overview.

Prototype - **21st of April** - The custommer was officially shown the prototype of our search-engine, and they approved of its design. They pointed out some modifications that had to be made, most notably the correcting the attributes and wanting a new attribute, order, added. The correct order of attribute entries have now been specified as order, family, genus and species.

Finished product - **11th of May** - On this date all functionality missing from the website was implemented and tested, so that we would end up with a product that (apart from a particular bug) is complete.

It was also shown to the University of Hamburg, where they expressed great satisfaction with our product, and also the way we provided the option for displaying all entries, even prefering it to the one they had originally desired.