Project: Beetle

22. april 2015

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

Instructor - Kasper Passov

Contents

1 Introduction

1.1 Abstract

Our project is, in broad terms, a search-engine for insects, which is to be provided to our client, the University of Hamburg.
Through German law and regulation, it is specified that in order for a university to receive governmental funding, it must allow public access to the research done by the specific university, and it is this context our project is necessary.
The entomology department of the University of Hamburg has a catalogue of information on different insects, their names, species, genus and a picture of the insect's anatomy.
This information needs to be available to the public, and the University of Hamburg wishes this done through the usage of a search-engine connected to a database.
The database must contain the insects as entries, with their names, species, genus and anatomy-pictures as attributes. The search-engine must then allow a visitor to input a search-term, which will be tried against any of the database's attributes, and return any entry that matches the search-term. It must also contain an advanced search-function that allows you to match a specific term to a specific attribute.
Furthermore, the University of Hamburg requires a method through which they can update the database themselves. Both to allow them to edit entries that may contain errors and to add/remove entries to/from the database. This method must only be available to authorised employees and not at all available to visitors on the website.
We are creating the search-engine in PHP, and it is to be implemented onto their website. The database is being created in MySQL and it will be editable through a method created either in PHP and implemented onto the website itself, or a stand-alone program, created in Java, given to the authorised people.

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 employees with the necessary login credentials to edit the database to either remove entries, change existing entries or add new entries.

2.2 Application-domain

Allow free, public access of data collected through research done by the University of Hamburg, so that everyone may learn from the conducted research.

2.3 Conditions

The product is being developed as part of the course PKSU at the University of Copenhagen. It is done entirely voluntarily and is non-profit. The client's wishes to the product are all taken into account and will be attempted fulfilled, although no complete product can be expected.

2.4 Technology

The product is developed on completely normal personal computers - no special tools are necessary in development apart from a PC. The product is designed to run in any browser without any necessary plugins, meaning that nothing but a PC and a working internet connection is required to use the product.

2.5 Objects

Anonymous visitors and employees can respectively view and edit the database entries.

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.

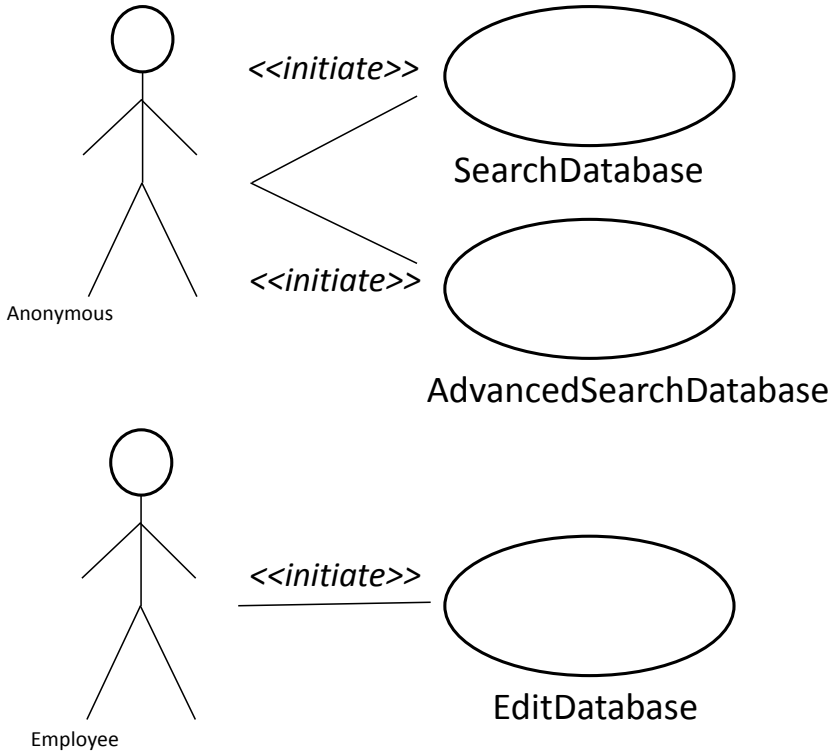
Functional Requirements

- 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.

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
Anonymous users use the search-engine and input a term in other to search the database and have the results they require displayed. Searching the database does not require any special permissions.

AdvancedSearchDatabase
Anonymous users use the advanced search-engine and input their search-terms into the fields representing the attributes they wish to match them against, in order to have the results they require displayed. Searching the database does not require any special permissions.

EditDatabase
An employee edits the database by accessing the edit-page through entering the necessary login credentials, modifies the database in which way desired and updates the database. EditDatabase cannot be used without valid login credentials.

The figure above is a use case model, which showcases the different possibilties of the product and to which kind of users they are available.

It is for instance evident through the model and the description of the functions, that any user will be able to search the database both in a basic and advanced way, where as only an employee with the necessary login credentials will be able to edit the entries in the database.

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 Anonymous
Flow of events	1. An anonymous user enters the website. 2. The anonymous user 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 the remaining 2 use cases:

Use Case Name	AdvancedSearchDatabase
Participating actors	Initiated by Anonymous
Flow of events	1. An anonymous user enters the website. 2. The anonymous user enters the search-terms into the fields of the matching attributes and presses send. 3. The matching results are displayed.
Entry condition	None
Exit conditions	•Anonymous exits the site. •Anonymous submits a new search term.

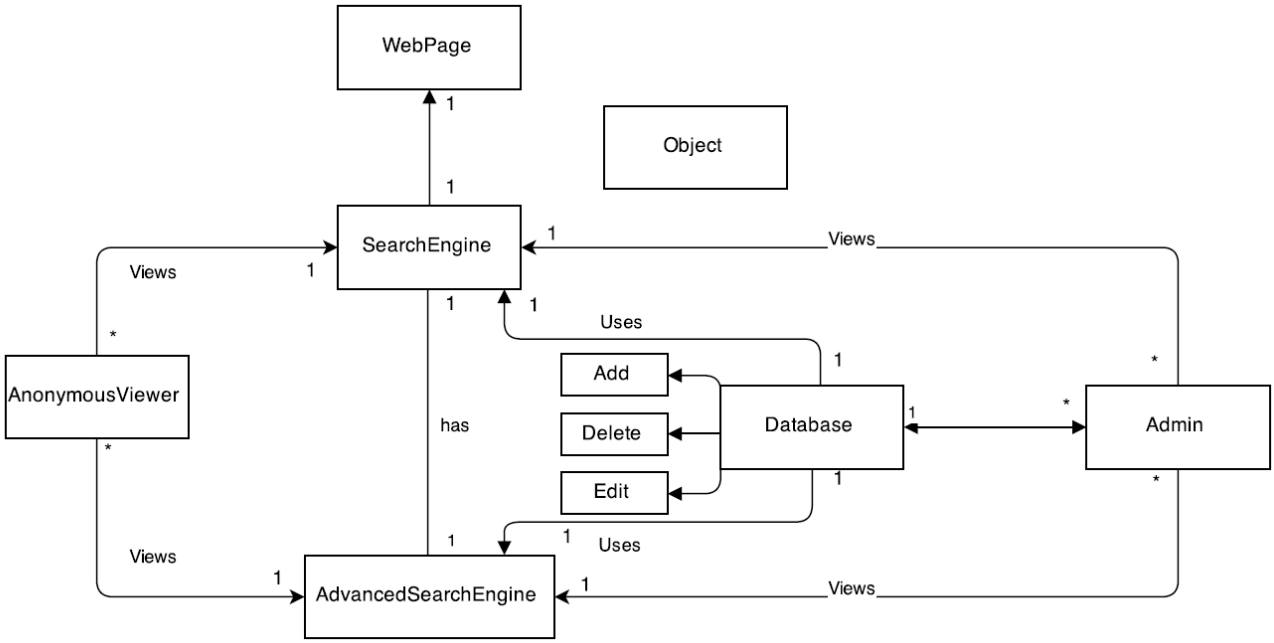
As we can see in the above case, no additional permissions/credentials are necessary to use the advanced search function.

Use Case Name	UpdateDatabase
Participating actors	Initiated by Employee Communicates with Anonymous
Flow of events	1. An employee user enters the website. 2. The employee clicks on the edit hyperlink. 3. The employee enters his/her login credentials. 4. The employee makes the desired changes to the database and presses submit.
Entry condition	•Must have login credentials
Exit conditions	•Employee exits the site. •Employee cancels the editing.

In this use case however, it's evident that in order to update the database, login credentials are needed. These are only to be provided to employees.

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 final product as a webpage offering 2 search-engines; one for basic and one for advanced searching. Furthemore, the engines are linked to a database on the back-end, which is editable by administrators (employees), as they have the necessary login credentials.

3.5 BCE-model

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

Boundary	**Control**	**Entity**
Buttons	*Search*	*Database*
Textfields	*AdvancedSearch*	*Images*
Headline	*ImageClick*	
Background	*Edit*	

Boundary:

Everything a user can see is a boundary object. In our case the buttons, search, advanced-search and edit, although 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).

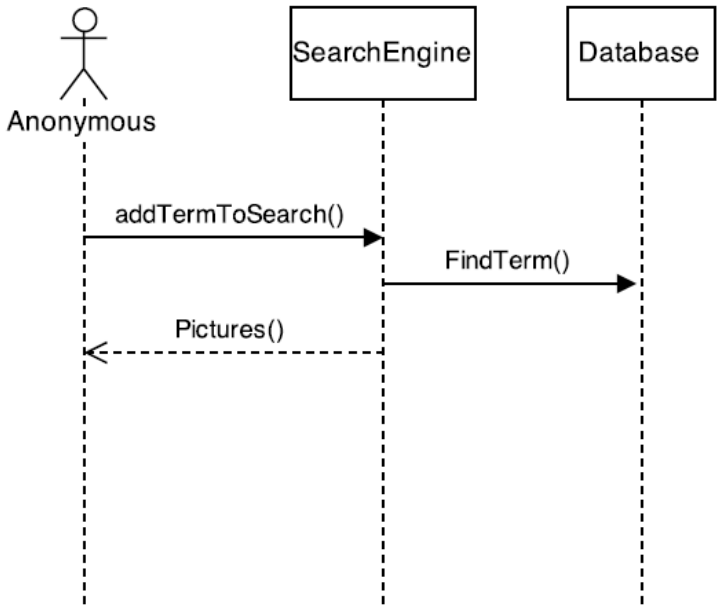
Control:

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.

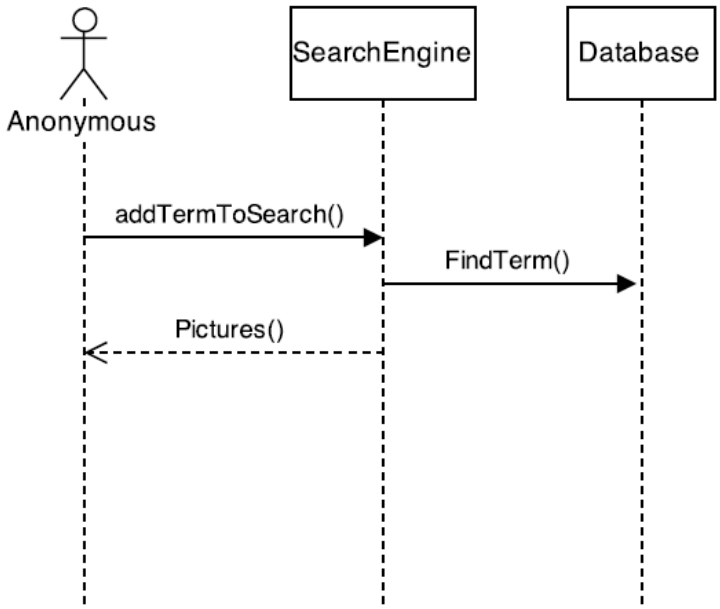
3.6 Sequence-diagram

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

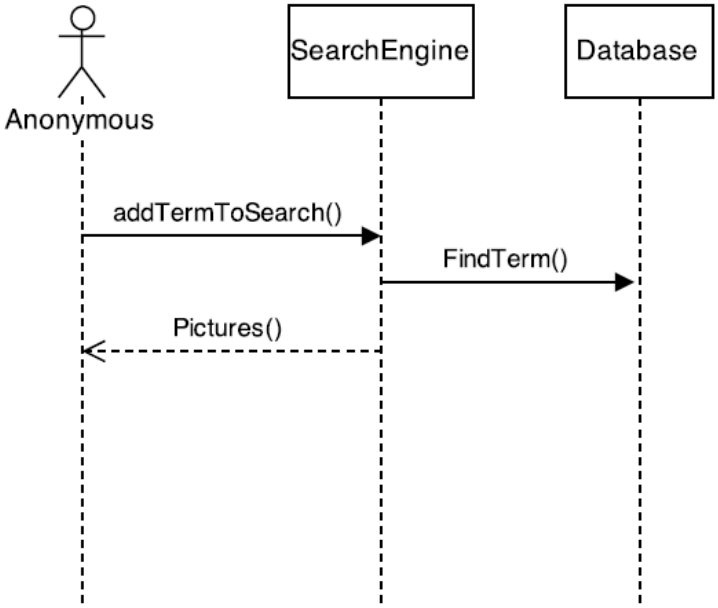
Use Case: SearchDatabase



Use Case: AdvancedSearchDatabase



Use Case: UpdateDatabase



4 System Design

4.1 Current Progress

The Prototype of our product can be found and accessed at: `http://echiever.de/ProjectBeetle/prototype.php#`

4.2 Summary

As of now, we're in the beta stages of the product. We've established a database containing 2 entries through MySQL using test-entries defined by ourselves. The basic search-engine is perfectly functional and returns matching results or an error message if none are found. The advanced search-engine has been designed and contains the 4 different search-fields, it is however not yet operational. A hyperlink leading to the editing of the database has been created, but the method of editing itself is still yet to be implemented. Finally, the design of the product so far is very basic as we're still in testing/implementing phase. Going forward, our tasks of importance are:

- Make the advanced search-engine operational
- Implement the editing portion of the product
- Establish the "correct" database using data provided by the University of Hamburg
- Improve the design to better match that of the University of Hamburg's existing website

5 Testing

5.1 Search-engine test

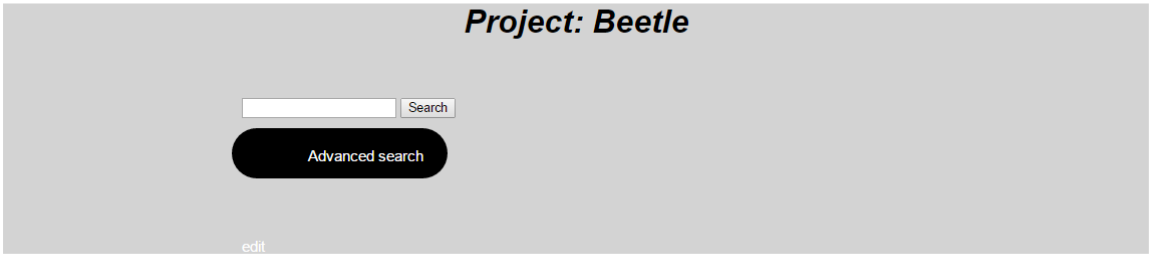
Through testing the basic search-engine, we've seen that sending in a search-input with no match will cause the error message "No Entry". We've also seen that inputting any term matching one of the 4 attributes of our own 2 entries will yield the appropriate entries being displayed. The current "search-able" terms are:

- Lepidoptera
- Nymphalidae
- Cethosia
- Cyane
- Heliconius
- Ismenius

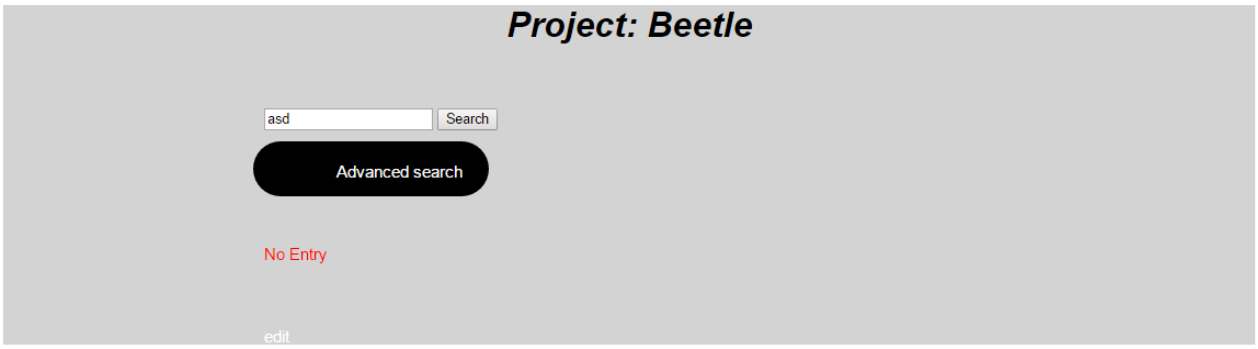
6 Interaction and Design

6.1 Design

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

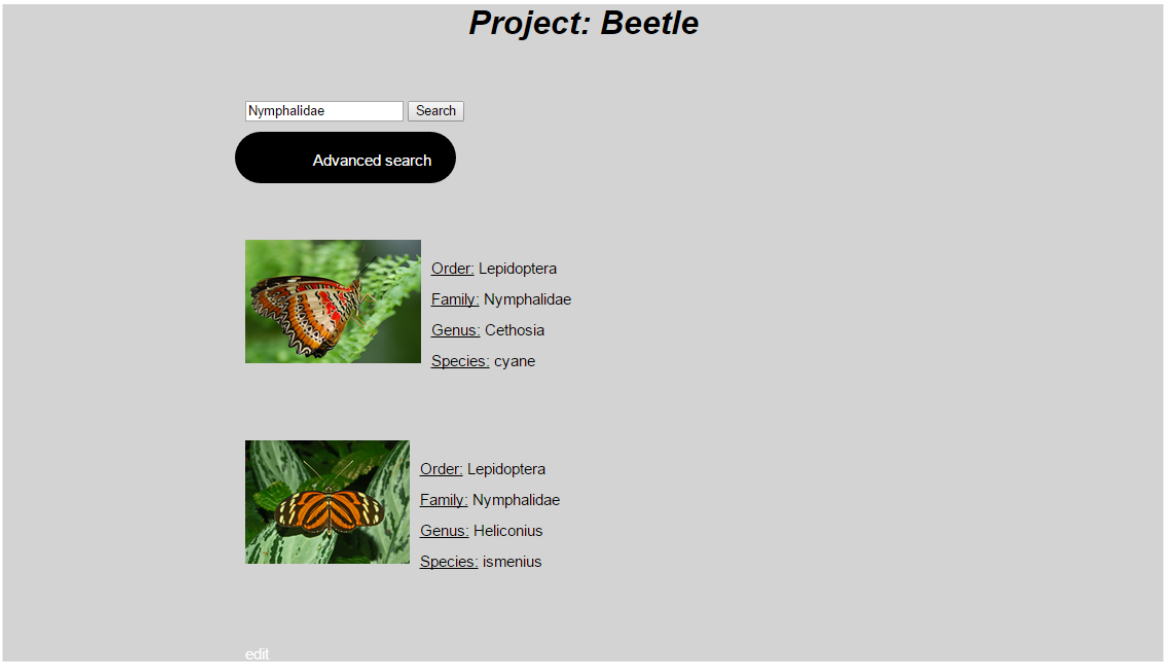


Above is the frontpage of the website before any interaction has been done.

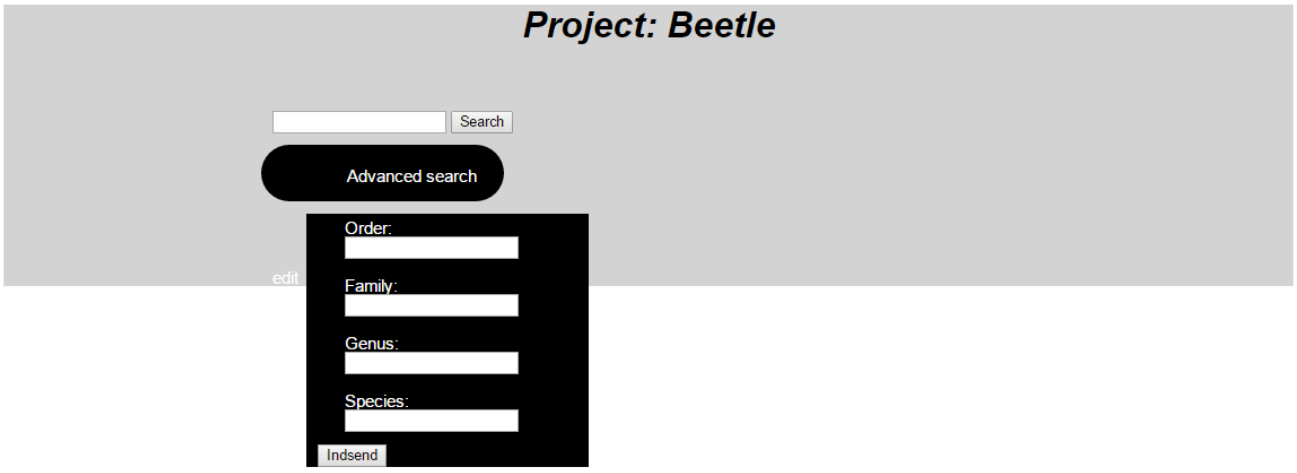


Here is what occurs if no match is found for the input.

The return of one matching entry.



The return of two matching entries.



The advanced search-engine, yet to implemented, shown by hovering your mouse over the button.

7 Internal Cooperation

7.1 Summary

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 correspondonces 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.

8 Litterature Reviews

8.1 Designing for usability - J.D. Gould and C. Lewis

The text is based on the premise of three simple principles: Early focus on users, emphirical measurement and iterative design. The text is about system designs and how in the modern day everyone finds change very drastic. The designers and developers' jobs are to make everything easily accessible, and to create the system as something which can turn into a successor to old habits.
The fact that the article is from 1985 would in some people's minds be a signal of distress as the paper is 30 years old. What could designers 30 years ago possibly know about designing systems that we don't today? Especially considering the technology we work with? Well to our surprise the paper was very valid, even today. The principles which the paper state are things people may feel are intuitive and obvious. Even we thought so. But either way it is not to be forgotten that when you begin designing, you need to familiarise yourself with the potential users. The developer needs to think as a user and not as a developer. Creating something that seems easy for you might not seem as easy for others. The paper also emphasises the use of emphirical measurements. Developers tend to forget the purpose of testing publicly. Every type of person should be involved in the testing because of the fact that not all people may find using the given system easy. Presenting the system for new users/testers should also be done in such a manner that normal citizens can relate to why the system provides improvements. It is naive to only look at it in a rational matter, because doing so doesn't involve users at all. It is after all the users who are going to be using it. One cannot say that the users don't know what they want, they just need help understanding it. Why should the school system in Denmark start using programs like TI-Nspire and Maple, when a calculator and paper has worked for centuries? The argumentation and advantages need to be pointed out. Another good point the paper mentions is promises. Promising voice recognition and touch screen doesn't exactly mean that the device/system will be good quality. The design must be iterative, there is no advantage in locking the design at one state - it is only stubborn. Everything can be improved. The paper, in short terms, focuses on the involvement of users and how the developers can involve more people. Changing habits is hard, but that doesn't mean it is impossible.

This paper is relatable to our own project involving the University of Hamburg and their desire for a search-engine. Our approach so far hasn't really followed the principles to the fullest, but since we are so early in the developing process it doesn't matter that much as of yet. What we have done, is kept a constant contact with our client. After every session we ask the owner how he feels about our work. Our intended way of working is using the AGIL project management, where we make sure to be open to ideas, and try to implement the ideas the client asks for. What we maybe should consider is asking other users, since the owner might feel like it's a good idea, but his customers/users might not feel the same way. Even though we're only making a search-engine, which should be straight forward, we can't work with the premise that people can't have issues at this early stage. We certainly need to remember testing it with users of all sorts, going forward. User involvement is something we need to try and implement future work.

8.2 A Rational Design Process - David Lorge Parnas and Paul C. Clements

This text is about a rational approach to designing a system. The idea is that you can't achieve one perfect result, but that you can fake it. Faking isn't seen as a bad thing in this situation. The idea is to do a lot of background work before the project coding commences. Today many coders usually utilise a "stream of consciousness" in which they realise an idea. The idea is then executed in a "stream of execution". In general this is a bad approach because the amount of time it would take to reach a result would be too long. That is why the paper recommends first making a requirements page in which you state what you, as a developer, can do and what the software has to include according to the customer. For instance which operating system the software has to work. Timing and accuracy constraints also help the developer set realistic goals and meet expectations. A probably worthwhile addition to a requirements document would also be changes likely to come up. This would not only make it easier for the developer long term, but also for newcomers who wish to improve the software. Another thing this paper urges new developers to do is to make sure documentation is proper and every step is documented. The paper urges developers to not have poor organisation, avoid "stream of consciousness" and not write excessive amounts of text when a shorter or more efficient paragraph could be used in place. Using confusing terminology should also be avoided - it is important to remember that even though you have lived inside this software for months, newcomers will be face-to-face with the software on a first time basis and not possess the knowledge you do. The small details are not as important as one would think - the bigger picture is considered more important. The general idea is to use terms that are clear and easy to understand, using figures and expressions to make it all easier for newcomers to take in. The final document is not meant to be relaxing to read, but interesting. It should reward the reader with precise and detailed information.

The key element in our project is the documentation. There is a reason that we need to use more time on the document than on the code itself. The backbone to the whole project is the documentation. We try to implement as many figures as we possibly can. One thing we could work on is not using long phrases and sentences which seems to be a recurring element in our documents. Something we should consider implementing in our work in the future is requirements that are more precise. As of now, we usually decide where we are going whilst working. If we had something to work up to it would make the process much easier, and easier to see how much we actually are missing to reach the requirements. Besides, if the project is to be given to Hamburg, they would surely love to know the requirements for future developers to work and adjust to. We should try doing more rational design, as it would make our work easier in the future.

9 Annexes

9.1 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 Killefilder
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.

9.2 Changelog

Date	Name	Change
$20-04-15$	$YO/TD/EG/CL$	*Added searchfield and -button*
$20-04-15$	$YO/TD/EG/CL$	*Added database and entries*
$20-04-15$	$YO/TD/EG/CL$	*Added edit link (it is still incomplete, but it's meant to be the employee-access to add new entries in the database)*
$21-04-15$	YO/TD	*Added Family-, Genus- and Species-names form the database to the webpage*
$21-04-15$	YO/TD	*Added picture of the found insect(s)*
$21-04-15$	TD	*Added CSS-code to improve look of webpage*
$21-04-15$	TD	*Updated database to contain actual insect entries*
$21-04-15$	TD	*Added Order-name*

9.3 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.