**Bulgarian Diploma Thesis**

**Endangered Species Website**

**Ralitsa Spasova, ID#100086884**

**Author: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ , Date: \_\_\_\_\_\_\_\_\_\_\_\_\_**

**Supervisor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ , Date: \_\_\_\_\_\_\_\_\_\_\_\_\_**

**Blagoevgrad, 2015**

**Declaration of authorship**

**Title**: Endangered Species Website

**Author**: Ralitsa Spasova

**Abstract:** This project is a website, which lets its users to view instances of endangered animal species on a map. In order to contribute to the map, users can create an account. The goal of the site is to raise awareness of the growing issues with preserving Bulgarian wildlife. Inspired by geocaching, the project relies on user participation and visual appeal. The administrators of the site have access to a dashboard, through which they can manage the account and animal information.

**Declaration of authorship:**

“The Bulgarian Diploma Thesis presented here is the work of the author solely, without any external help, under the supervision of Dimitar Christozov. All sources, used in development, are cited in the text and in the Reference section.”

Author:

**Table of Content**

1. Introduction
   1. Overview of project
   2. Overview of technologies
2. Specification and Analysis of the Software Requirements

2.1 Functional Requirements

2.2 Non-functional Requirements

2.3 Use Cases

2.4 Main UML Class Diagram

1. Design of the software solution

3.1 User Interface  
3.2 MySQL  
3.3 ER Diagram  
3.4 Queries  
3.5 Subsystems  
3.6 UML component diagram  
3.7 UML deployment diagram  
3.8 Security Features

1. Implementation

4.1 Technologies used  
4.2 Installation Requirements  
4.3 Other Features

1. Results and Conclusion

5.1 Screenshots  
5.2 Project Success  
5.3 Schedule  
5.4 Problems  
5.5 Future Development  
5.6 Conclusion

1. References

**Introduction**

Human technological advancement gave way to endless opportunities, which were unheard of just some two or three hundred years ago. Industrialization propelled us into a brand new course of our evolution. It is undoubtedly the fact that we have cemented our superiority over the rest of the animal kingdom. However, as uncle Ben said "With great power comes great responsibility." Being on top of the food chain is only good if there is a chain to begin with. Every day, more and more species become extinct due to their environment being destroyed or changed so drastically that they are unable to adapt fast enough. As human population worldwide boomed, so did our need for resources. Almost all leading scientists are in agreement that global pollution, as well as deforestation, is at a critical level. Whole ecosystems are being demolished and with them, the flora and fauna of that region. WWF attempt to keep an accurate statistic of the endangers animals across the globe, although that appears to be an impossible task, as they remind us that even yet undiscovered species succumb to extinction on a daily basis. As of now, hundreds of documented species have gone extinct and thousands are at serious risk.

The reason I chose to do this project was to raise awareness of the very real ecological issues we face in the modern age. Many organizations are attempting the same and doing a marvelous job at educating people on the impact we have on our planet. Unfortunately, a lot of their efforts go unnoticed by the general population.I chose to develop a website, as the internet, with its ever-growing popularity, seems to be the best medium for transmitting information to the largest amount of people.

Although a website solves the problem of finding an audience, there is another obstacle that needs to be considered. The internet is a massive collection of information, too massive to be explored in its entirety by a single person. Therefore, the information presented needs to be packaged in an attractive manner. In order for a person to be interested in a piece of information, it needs to stand out and capture the user's attention.

A third aspect to take into account is the level of involvement of the user. From personal research I have deducted that people are more interested in something if they are more involved in it. That is why, for example, the popularity of the gaming industry is at an all times high.

With all of these considerations in mind, I decided to develop a website, which would allow users to find information on Bulgaria's endangered species. It is packaged in a visually appealing manner by using Google Maps. Users can add an entry, depicting the place where they have spotted such an animal, and add any extra information, which might be useful or interesting. By doing so, they can inspire others to go on their own search for these specimens. In the long term, this website can provide a good source for any researchers, interested in such fields as preservation of natural wildlife or migration patterns of different species.

My project was partially inspired by the successful geocaching, which has a large dedicated community spanning the entire world. Geocaching also heavily relies on the use of coordinates and mapping, as well as the users’ active participation. The more people are involved in it, the more exciting it becomes. My project functions on the same principal.

Technologies Used

With all of the aforementioned points considered, for my project I decided to develop a website. And as expected, for the purpose I chose to implement the functionalities using the languages PHP and JavaScript. It probably goes without saying that HTML and CSS played an important part as well, as they are really the basis of front-end web design. Another crucial element in the making of this project was the use of a database, where all information about the users and the endangered animal entries is stored. There are many wonderful options for data management. However, I decided to go with MySQL as I have already worked with it before and I am familiar with phpmyadmin, a tool for easy and reliable operation with MySQL. I mainly used PHP for connecting with the database and writing queries, thus connecting the data side of my website with the user interface. In addition to that, it is through PHP that I was able to make forms send and receive information, as well as create and maintain sessions for the logged in users. JavaScript was needed for displaying the maps and the information in them, sending alert messages to the client’s browser, displaying the modals, containing the login/register forms. Moreover, the dynamic validation of those modals was also implemented thanks to this scripting language.

The most important technology in this project would be the Google Maps API, as it si the focus of my website and the idea behind it. I had not utilized it before this semester, so I had to look up tutorial from the developers. The maps on the site display all of the added animal entries and their corresponding information. Users can also search for a map of just a specific class of species or only animals, uploaded by a certain user. I also used google maps to help the user get the coordinates when uploading a new entry, instead of having to manually input them.

For writing and editing the code I used Notepad++. It is incredibly simple to navigate but it has the advantage of recognizing the different types of languages, thus making it harder for me to get lost in the code. For the navigation banners I used Photoshop CS6. The styling f the pages was done with the help of Bootstrap.

For convenience I installed a WAMP server on my laptop, so that I could work on the project anywhere I went.

**Specification and Analysis of the Software Requirements**

Functional Requirements

* Guest users should be able to view the map with locations of all instances of endangered species, spotted by a registered user
* Guest users should be able to view any extra information for a specific chosen entry
* Guest users should be able to search for spotted species, by indicating the animal class
* Guest users should be able to search all animal, which have been spotted by a specific registered user
* Guest users should be able to login into their accounts, if such an account has been already created
* Guest users should be able to register for an account
* Registered users should be able to perform all of the tasks, which a guest user can do
* Registered users should be able to log into their accounts by providing a username and a password
* In case of a forgotten password, users can request it by providing their email address and the security word
* When a user logs out of their account, the site should redirect the user back to the home page
* Registered users should be able to add a new entry to the database through a form, which is located on a separate page
* Admins should be able to perform all of the tasks, which a guest and a registered user can do
* Admins should be able to access a dashboard, which is located on a separate page
* Admins should be able to view the information of all registered users and all animals
* There should exist an option for an admin to change the privilege of a regular account to that of an admin and vise versa
* There should exist an option for admins to delete a specific account or animal entry from the database
* When an admin deletes an account from the database, any entries corresponding to that account should be deleted as well

Non-functional Requirements

**Usability**

The interface of this project is relatively simple to navigate. There is nothing complicated, which might cause a problem for the person. The website, along with the features within it, is incredibly user-friendly. The links are appropriately labelled and any forms have instructions on how to correctly input information.

**Security**

While developing a website, there are a number of security considerations, which should not be overlooked. While the site is completely secure, it has basic functions to keep the data safe from interferences. For example, the login and register forms have dynamic validation. When retrieving a password the user needs to provide us with their security key, which they choose upon registration. The passwords are also encrypted for extra protection.

**Compatibility**

A well-made website should work efficiently on different browsers.

**Availability**

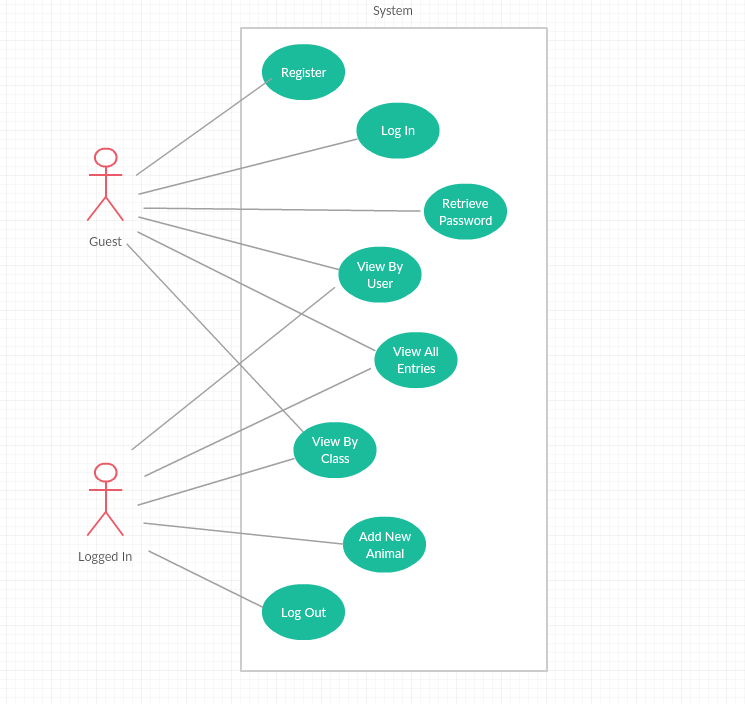
For the purpose of this project, the website is available only on a local server. However, if future development continues, the site will be uploaded to a host, making it available to use by anyone with an internet connection at any time of the day.

Use cases

To better understand the functionalities of the project, it would be easier to separate them according to the type of user. For the website, there are three possible scenarios: a guest user, a logged in user and an admin.

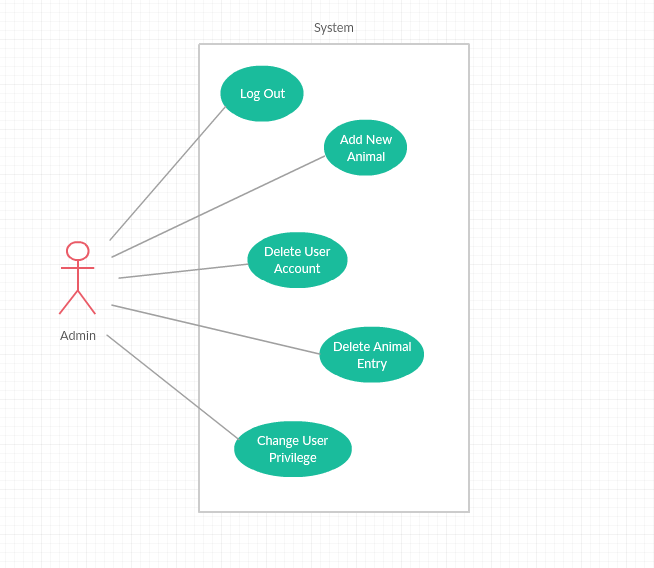
Both guests and registered users can navigate the Home, About and Map pages without any restrictions. Furthermore, they can look through the maps and search for animal entries either by species class or the user who uploaded them. Any information about a specific animal entry is public. The guest user has an option to log into their account, if one exists, or otherwise create a new one. The logged in user has the option to log out of their account instead. The logged in user also has the ability to add their own entry to the animal database by filling out the form on the Add page, which a guest user cannot access.

Below is a use case diagram, depicting this behavior.

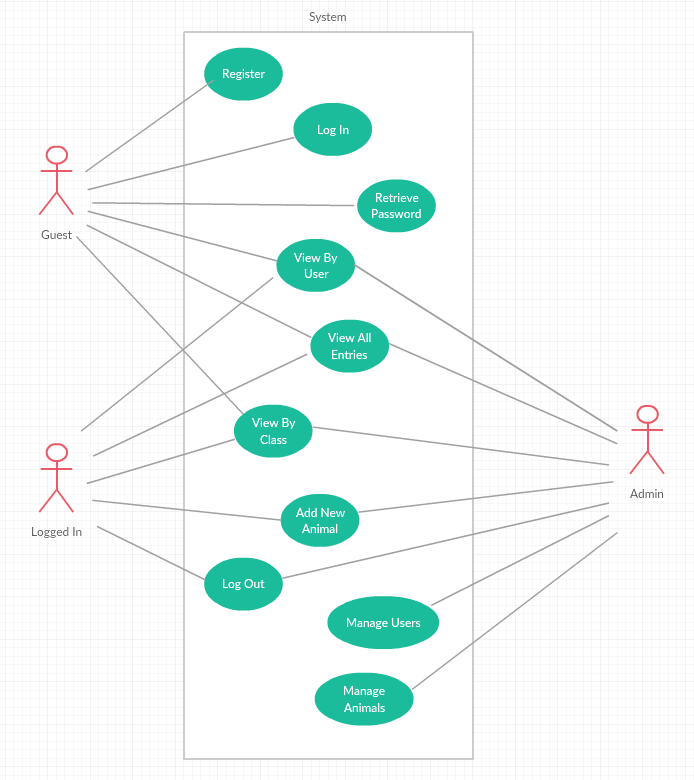


Admins are a special case of a logged in user, therefore they can perform any of the tasks of the previous cases. In addition to that, admins have access to a Dashboard page, where they can view the information of registered accounts and added animal entries. Admins can delete any user or entry, as well as change the privilege of a user to an admin or the privilege of an admin to that of a regular user.

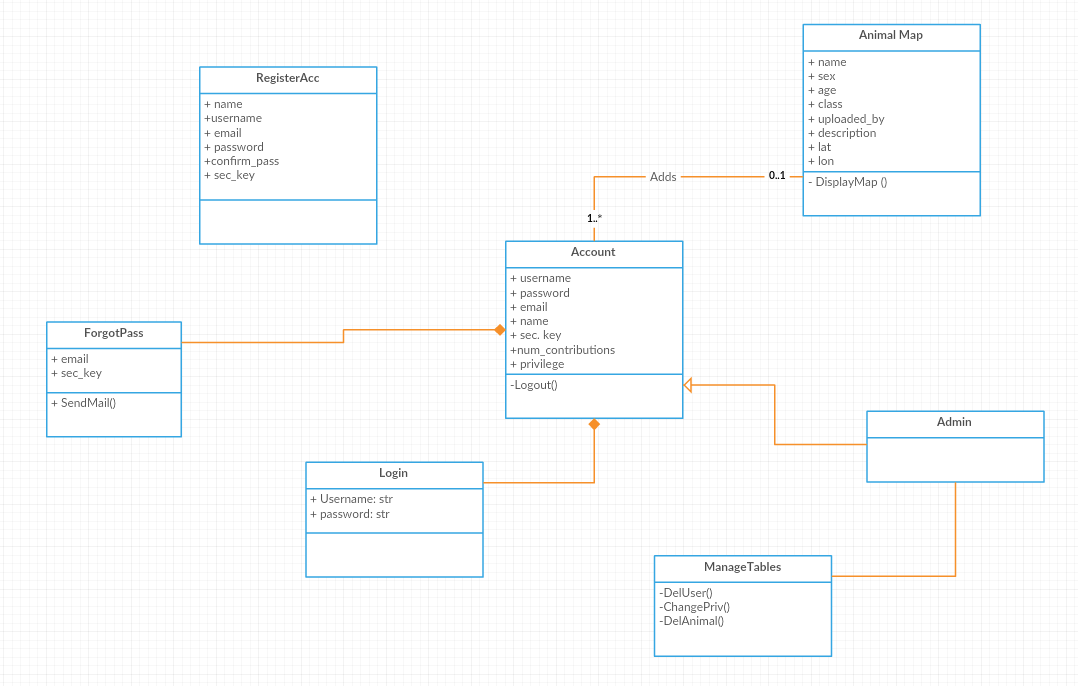
Below is a use case diagram, depicting this behavior.



Complete Use Case Diagram

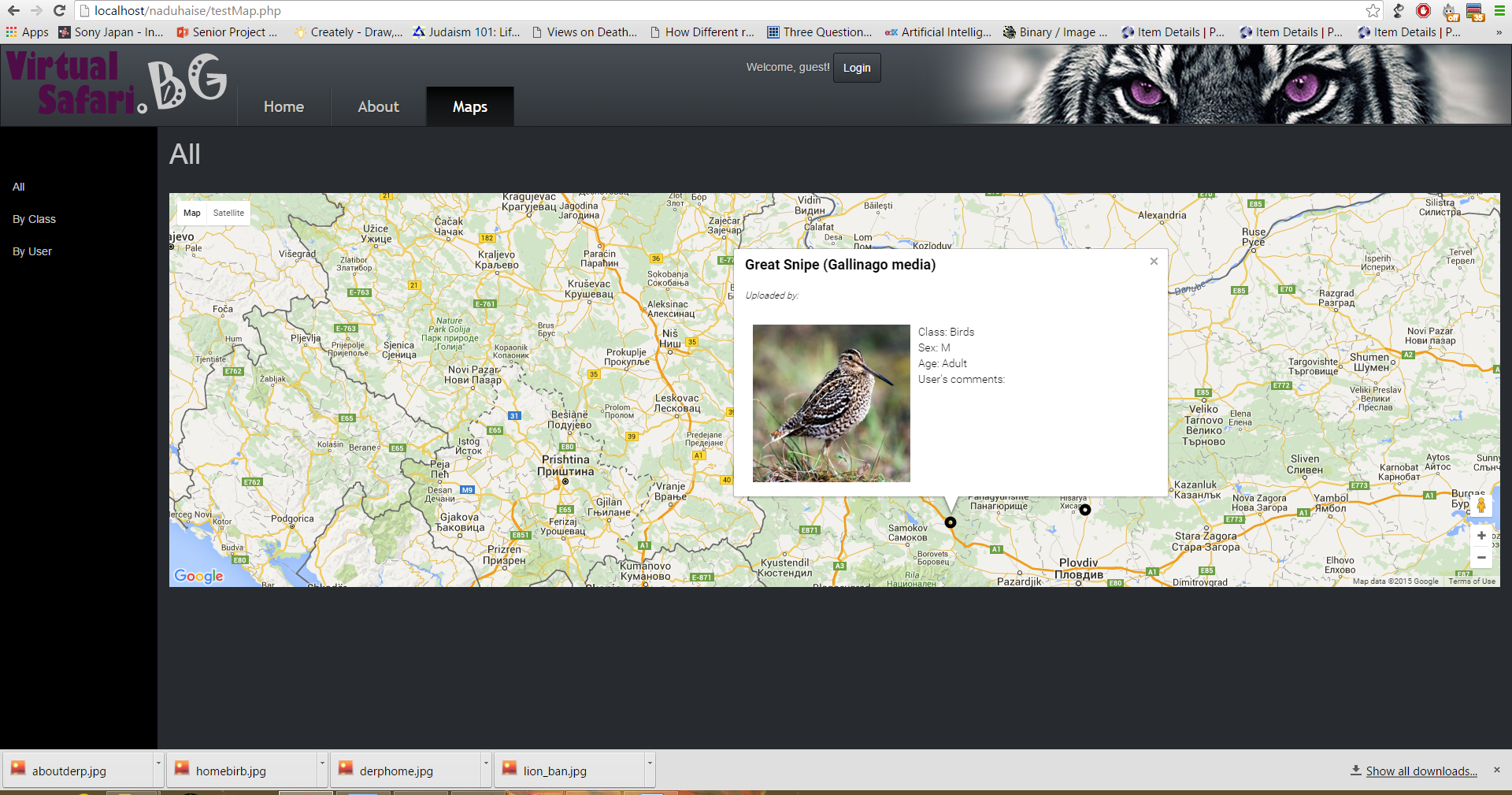


Main UML class Diagram



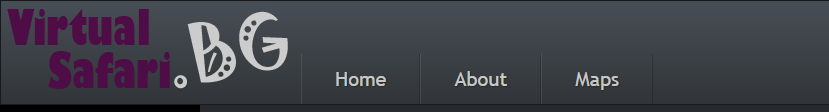
**Design of the software solution**

User Interface



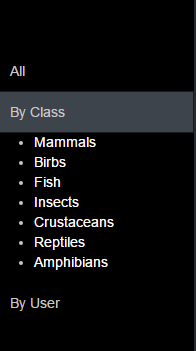
Main Navigation Bar

The global links for the website are located in the top navigation bar. The guest user can go to and from the Home, About and Map pages. When a user is logged in an additional link shows up next to the others, which redirects the user to the form for adding a new animal entry. If the user happens to be an admin, another link is displayed in the navigation bar, which takes the user to the admin dashboard. In addition to those links, the bar also displays a little message for the user, indicating his status as a guest or a user (by displaying the username if logged in). In the case of a guest, the button next to the message opens up the login modal. Otherwise, the button logs them out of their account. Lastly, in the top right corner is an image for the purpose of making the site more visually appealing. Below is a screenshot, illustrating an example of a guest user.





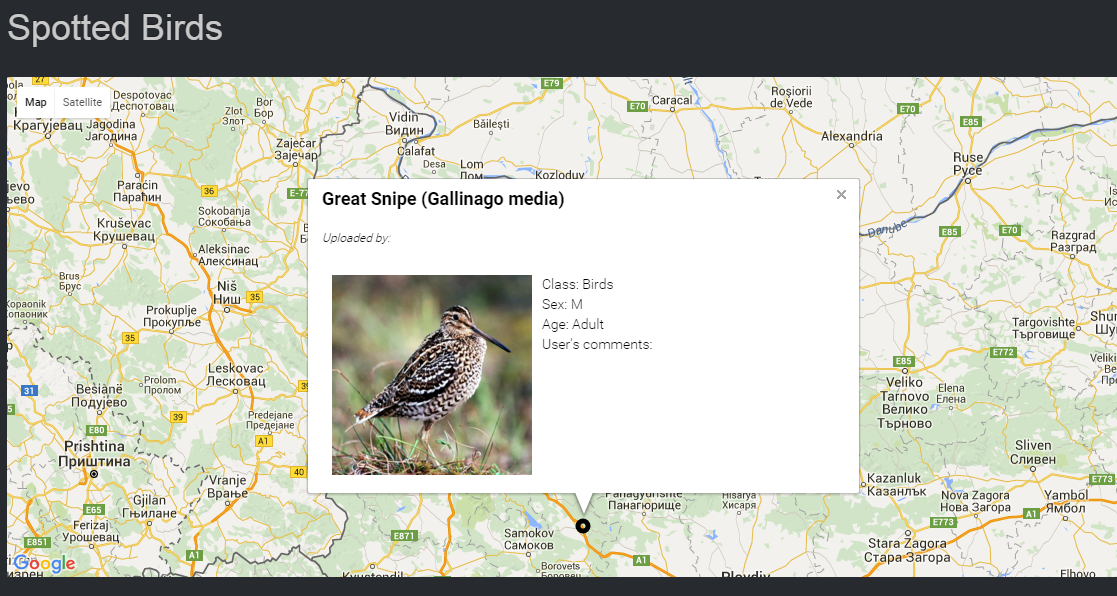
Sidebar



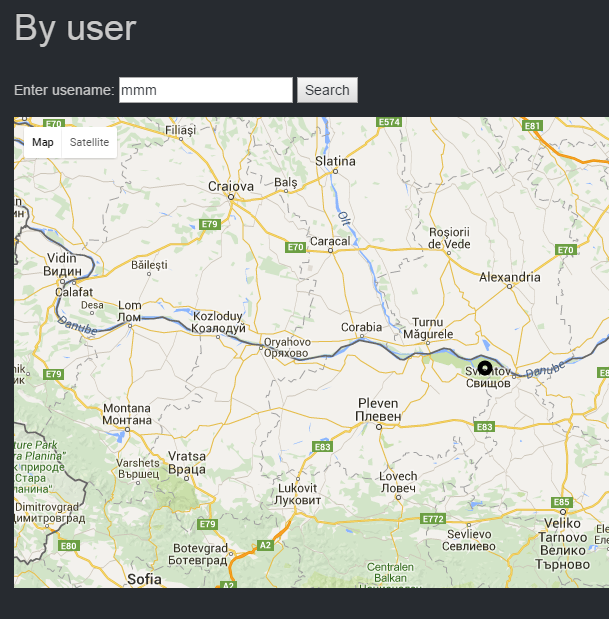
The sidebar on the left of the page contains the secondary links, which are specific to the open web page. They serve to organize the content of the page in a neat and logical fashion. When a user clicks on a link from the sidebar, the corresponding information is shown on the right in the main content area. This is a standard layout for websites, so it is not confusing. In the case of the Map page, when the user chooses to search animals by class, a nested list opens up to allow the person to choose from the available classes.

Google Maps

The geographical representation of the database is arguably the most [] side of the interface. Implementation of the Google maps API is done through a call command in JavaScript. This allows the developer to call functions and methods, specific to the map interface. That way, it is possible to define such features like the default coordinates of the center of the map upon loading the page, as well as the initial level of zoom. In my case, I chose the placement of the focus in such a way that the map will display most of the territory of Bulgaria. Of course, it is not wrong got assume that most users will already be familiar with how to navigating the maps. Clicking an dragging the mouse will move the map in the selected location, while scrolling serves as an easier alternative to the plus and minus buttons, responsible for the zooming in and out. Markers are shown on the map as small black circles. When a user clicks on a marker, an information window opens up above the selected marker. In this information window are displayed the name of the endangered animal, the username of the uploader and a thumbnail picture of the species. Moreover, there is also other useful data, such as the animal class, its sex and age If known and lastly, an optional user commentary or description of the specimen they have observed. The content of the information window and the marker’s styling’s defined in the same function as the map within the script tags. The function executes upon loading the page. In the markup part of the code, the map is called in a div with the corresponding id name.

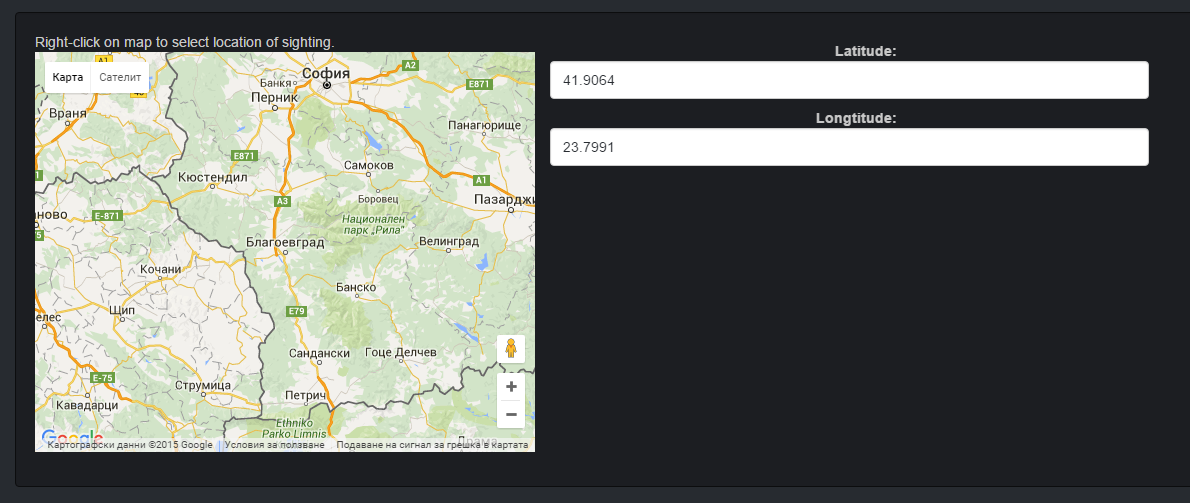


The same principal applies for all of the maps in this project. The only major difference is putting a condition when the user wants to search a specific class of animals or entries by a certain user. IN the former case, the user simply chooses from the options, located in the sidebar menu. In the latter, however, the user is first given just the option of inputting a username. Once they click the search button a map appears below, marking any results, which match that username.

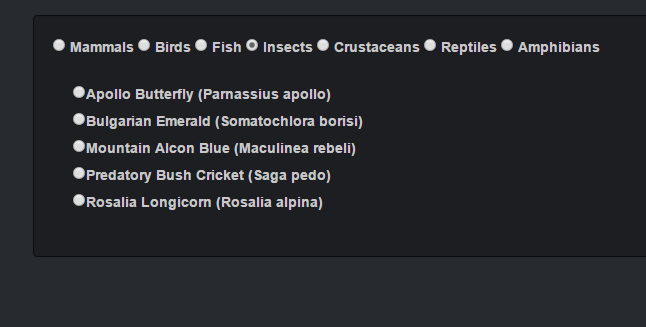


Add entry form

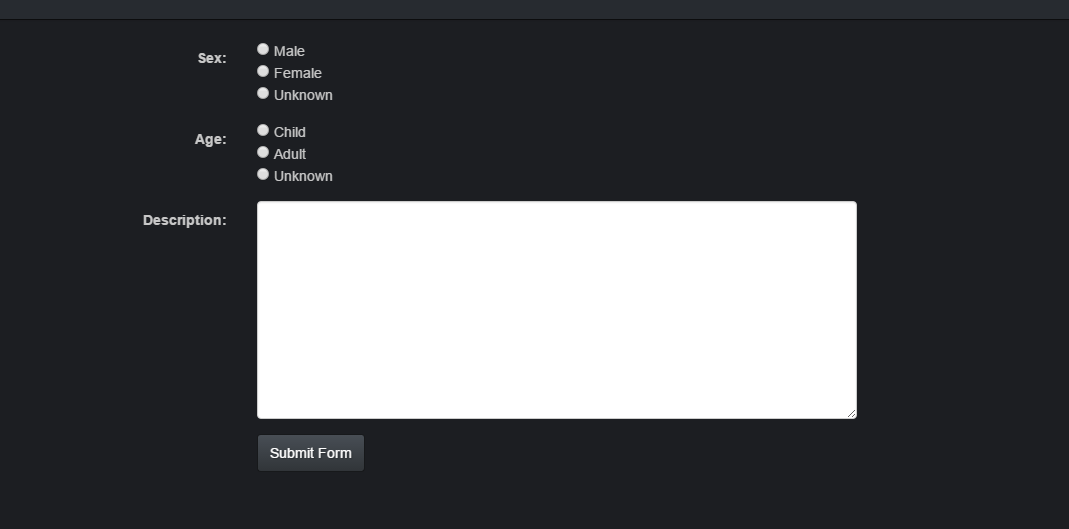
In order to add a new entry, the user must first be logged into their account. That way they have access to the page, in which the form is located. Its design is made in such a way that it provides the user easy to understand steps, accompanied by detailed instructions on what to do for each one. The first thing a person needs to do is choose the location of the animal sighting. Instead of having to manually input the latitude and longitude, the user simply right-clicks on the map and the numbers automatically appear in the input boxes below. The map uses the same interface as the others for the convenience of the user.



The next is choosing the class of the animal. Upon selecting one of the options, another set of radio buttons appears, representing any endangered species of that class within the territories of Bulgaria. The database for all possible endangered species in the country is taken from an external source, the link to which I have provided in the Reference section of the documentation. I decided to use predefined options for the species, as that makes organization relatively easier. If the user were to input their own species names, that could potentially cause problems. For example, the user might add an animal, which is not in danger of extinction, or might type in an incorrect name or might not be familiar with either the common or scientific name. By implementing radio buttons, I make sure that the data is stored in a clean and organized fashion, saving both users and administrators a lot of time and effort.

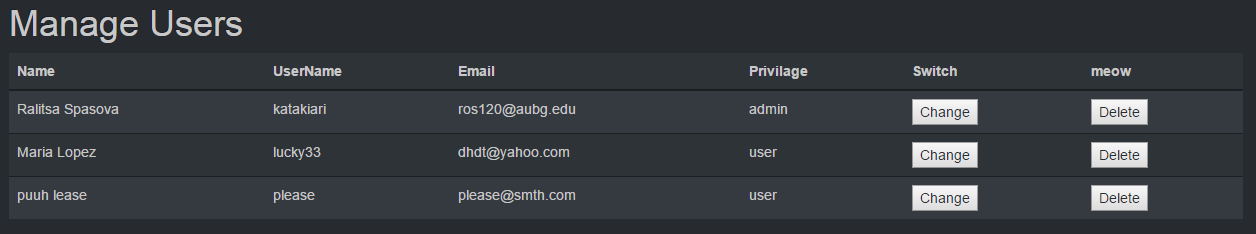


Finally, it is time for the user to choose any extra information, which might be helpful to others, like the sex and age of the species. Since it’s difficult to determine the exact age of an animal just by observation, I have simplified the option by separating it into three possibilities: adult, child and unknown, in cases where it cannot be determined. What is more, there is an optional section, where the user can include his or her own commentary. After all of the fields are filled up, the user clicks on the submit button. An alert will pop up on their browser, telling them if the entry has been successfully added.



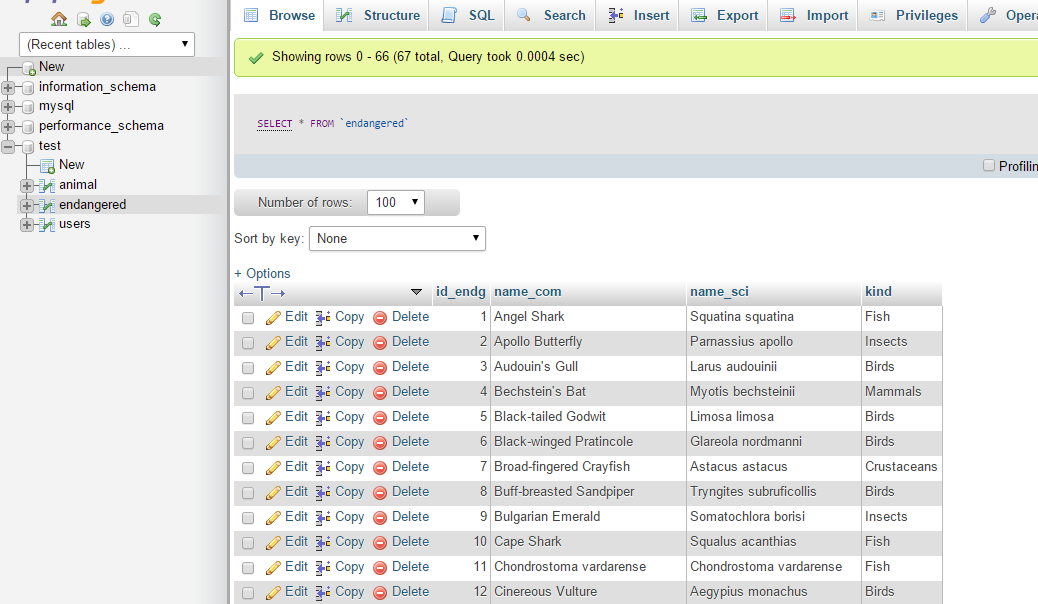
The Admin Dashboard

When a user with administrative privilege logs in, they have access to the dashboard, located on a restricted page. It contains two tables, one with the information of all registered users and one with all user-added animals, currently in the database. In the users’ table there are two columns for managing the accounts. One of them allows the admin to delete any of the accounts, while the other changes their privilege. The animal table has the option for deleting as well. These functionalities are presented in the form of buttons, corresponding to each row. That way the admin can simply click a button on the same row as the entry they want changed without affecting the others. Of course, the admin does not have access to sensitive information, such as the passwords.

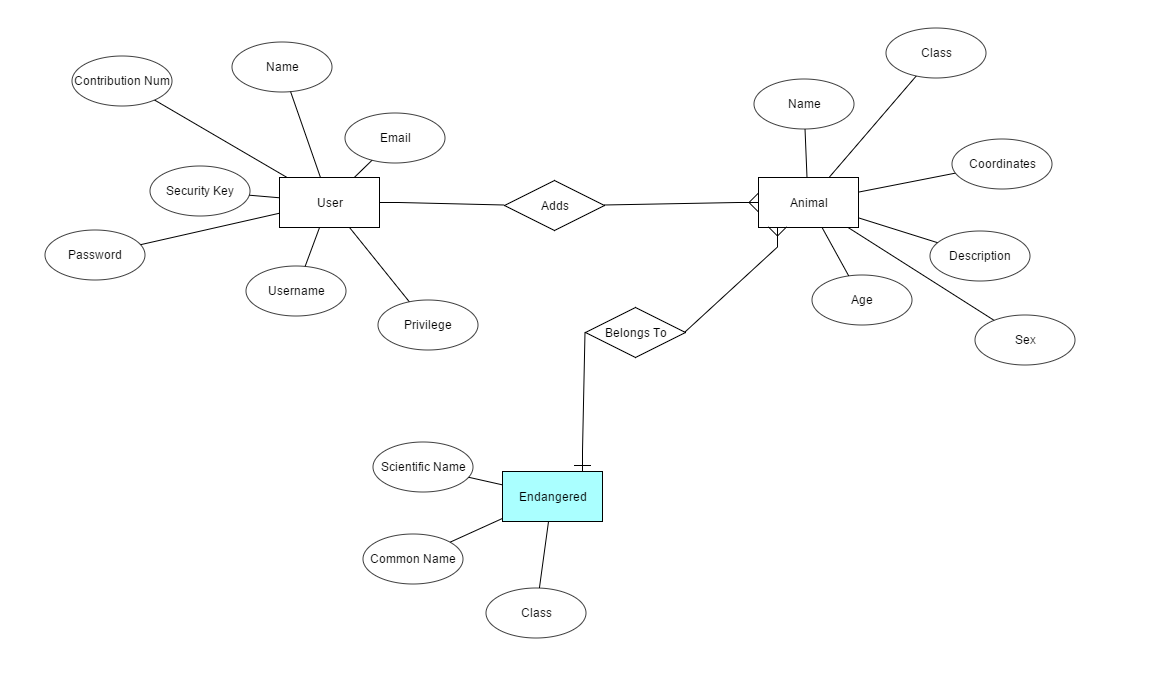


MySQL

Now that I have covered the user interface, it is only appropriate to addressing another crucial layer of the project design. Obviously, in order for the site to work as intended it needs to store the information on accounts and their uploads in a database, which can then be accessed and modified. I have significantly more experience with Mysql than any of the other popular management systems. The advantages of choosing it are that it is free to use. Not only that, but by doing so, I can build my database through phpmyadmin, a handy software, which allows me to manage my tables not only through SQL commands but also a graphical interface. Phpmyadmin has proven to be incredibly convenient. It allowed me to export my tables from the AUBG server and import them on the one I set up on my laptop. All actions, related to the database and its content, are performed through PHP code. My database consists of three tables: one for the data of the user accounts, one for the entries, uploaded by users and a third one, which has all possible species of Bulgarian endangered animals.



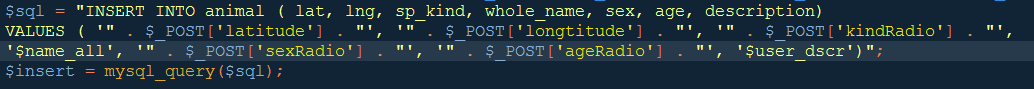
ER Diagram



Queries Used

While the interface and the database are important parts of the website, it is the SQL queries that make the appropriate connection between the two possible. Within my project I have used a number of them, which, according to their function, can be separated into the four types described below. All of the queries are executed once a connection to the database has been established.

**Adding a new row**

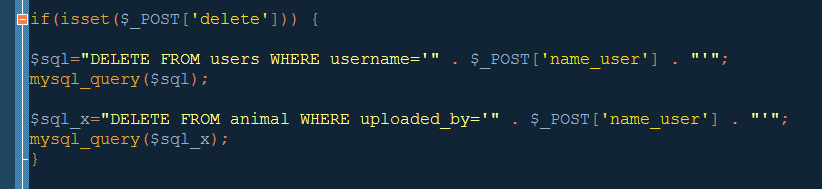


The above code executes when the user has submitted the form for adding a new endangered animal. The data is being read from the inputs and stored in the animal table accordingly. The uploader’s username is taken from the session.

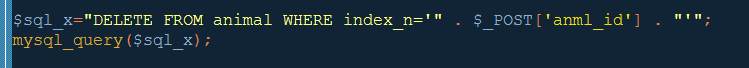


This query adds the newly registered user’s information to the users table, but only after the input values have successfully passed through the validation process.

**Deleting a row**



In this case the admin has pressed the delete button in a certain row of the account’s dash table and the query deletes the row in the database table, which has a matching username. When an account is deleted, any of the entries, associated with that account, are also deleted from the database.

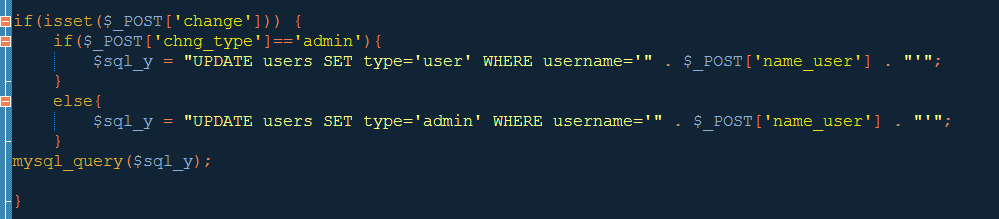


A similar query to the previous one deletes a chosen animal row from the database upon the admin’s command.

**Updating existing information**



When the admin deletes an animal entry, this query is used to change the uploader’s contribution number by decrementing it by 1. I have added an extra condition at the end to make sure that the number cannot fall below zero.

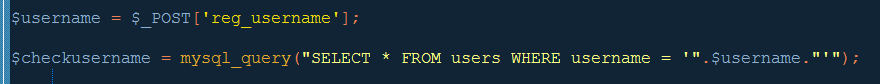


Another query, which deals with already existing data, is the one responsible for changing the account’s privilege. Since there are only two types of privilege, the query basically switches the values around. If the account is an average user, clicking the button will change the value to “admin”. And if the account is an admin, ti will change the privilege to “user”.

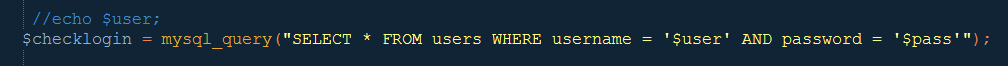
**Displaying information and checking input**

The simplest and most common query, which I have utilized, serves to extract data from a specific table. This is incredibly useful as later I can use the information, for example, display it in the form of a table, as is the case with the admin dashboard.

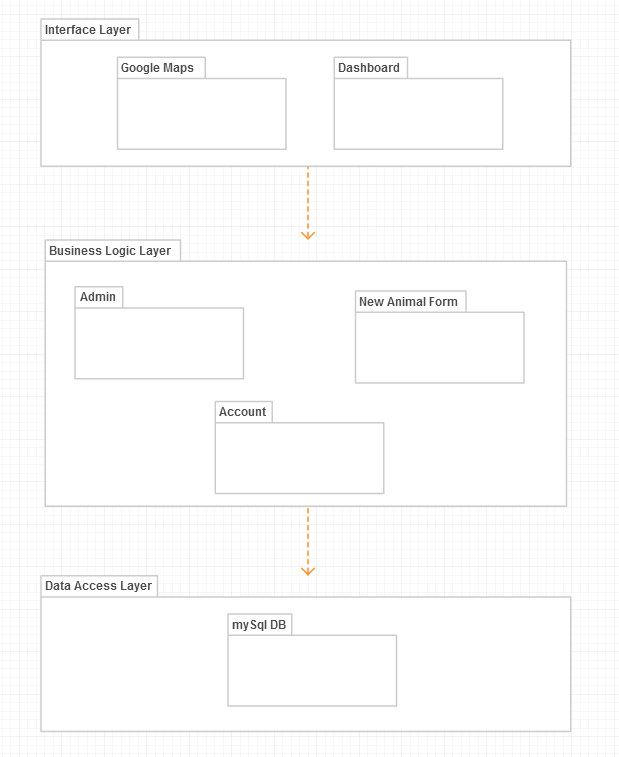
The SELECT statement can also be used along with the WHERE clause, in order to provide some sort of a condition for the data that we need. I have implemented such queries for several reasons. One example is when the user attempts to register their new account. This query extracts data with a matching username. If such a row exists, the user is prompted to change their selected username.



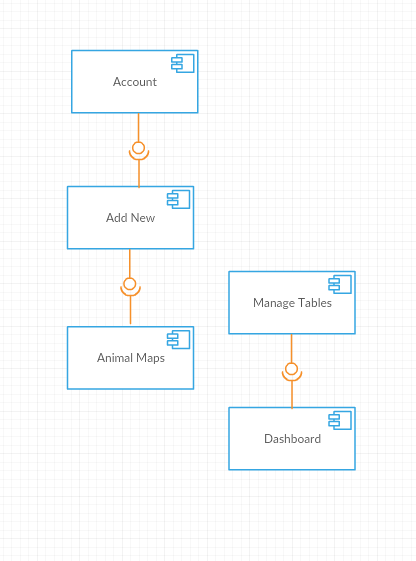
In a similar manner, this query is used to check if there is such a row in the database table, which has the same username and password values as the inputted by the user upon logging in.



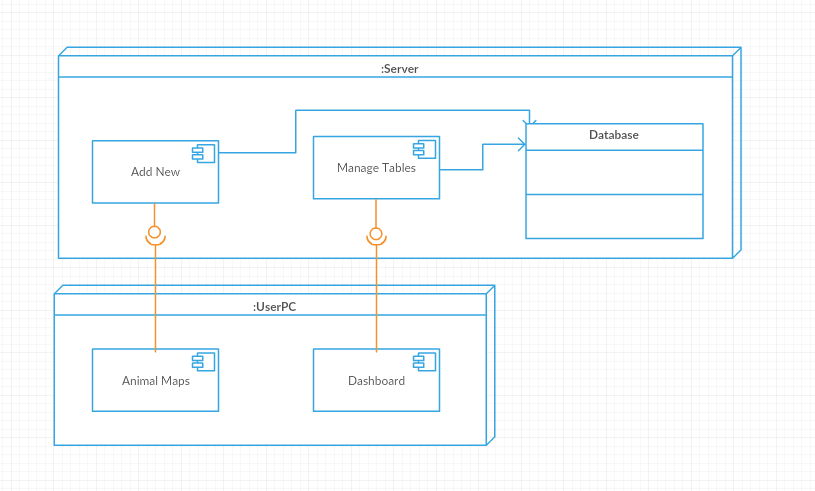
Subsytems



UML component Diagram



UML deployment diagram



Secutiry Features

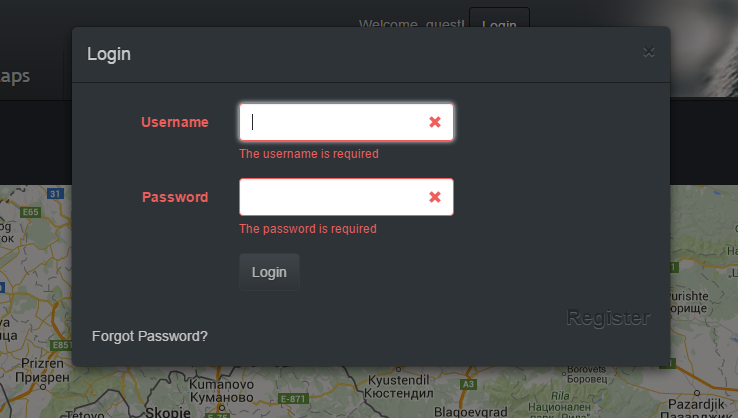
Even though the main concern of this project isn’t related to security, protecting information from intruding third parties or from data corruption is still an indicator for a quality website. There are countless of precautions a developer can add to their work. Proof of this is the high number of articles and pages, found online, dedicated to making a website more invulnerable to attacks. I decided to include some part of these measures and will describe them in detail below.

**Password encrytion**

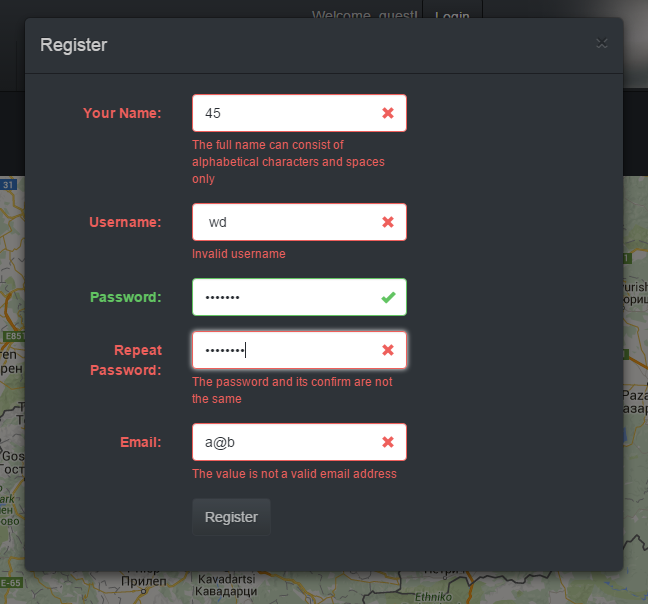
Passwords are encrypted upon registering the new account with md5. The values in the database contain the hash of the pass, so that it cannot be sent or decrypted.

**Dynamic validation**

It is only common sense that we should ensure our data is protected as best as we can. And one of the absolutely necessary ways to do so is to validate the information, which get delivered to the server through a form. Every time we allow the user to send some form of textual or numeric input, we expose the website to any potential hackers. Not only that but setting up some loose guidelines on the type of characters and their order for each text box can in the long run help the database become more maintainable and modifiable. These input “rules” come in the form of conditions, like whether is not the value of a field is null (or, in other words, empty).



The log in form is sufficient enough with only that one rule, however, the one for creating a new account is another matter. For starters, the field, designated for entering the person’s full name, can only accept strings of letters and white spaces. The username, on the other hand, does not accept whitespaces yet allows digits. Passwords share the latter conditions and in addition to them add a restriction on the length of characters. IN my case, I decided that between six and twenty is a reasonable constraint. There’s a confirm password field, as expected. The values of both need to match exactly. Maybe the field with the most complicated regular expression is that for the email. There’s a certain order for the different sets of characters alongside other specifications. I attempted to find the best for my purposes. There are many different expressions, designed by number of developers. Some were too general while others were too long and strict to be considered useful. In the end, I decided on something between the two extremes.



**Forgotten Pasword Security**

If the user has forgotten their password, they can follow the link on the loin modal. They type in their username and an email is sent to the address, corresponding t that username with a reset link. When the user clicks on the link in their received message, they are sent to a page that prompts them to create a new password.

**Restricted pages**

I have already discussed the pages, which a user an open, depending on his status. As far as the interface goes, the links on the top navigation bar change accordingly. A guest user will not see the link to the Add page, for example. However, in case he does have an account and has logged in at least once, he has had the opportunity to see the path to that page. In such a scenario, the guest can theoretically input the URL and reach the page regardless. Or even worse, we can imagine the level of damage a former admin can cause to the data. Without authorization, he could delete all of the entries and accounts. IN order to prevent such a catastrophic event, I have put in some PHP code that checks whether the person is allowed to open those pages or not. IF yes, the page loads normally without the user even being aware of that condition, while in the opposite case, the user is shown an error and redirected to the home page instead.

**Implementation**

Technologies used

**Google Maps API**

The implementation of my project idea would not have been possible without the use of maps. At the very beginning when I was still thinking of ways to go about this issue, I was unsure of the exact method but Google has provided the best solution to my problem. They offer APIs (application programming interface) both for commercial and other uses. The site which introduced me to the basics of Google Maps had a variety of options available, depending on the developer’s needs, as well as the chosen platform. Since I decided on building a website, I needed the appropriate interface, which in this case was Google Maps JavaScript API for Web. Implementation is incredibly simple, done with a single line of script tags, located in the head section of my html.

Now that we have inserted that line, we can initialize and customize a map. Again using JavaScript, the map is declared, along with its default location. In the same way markers and the information windows, which open up when a marker is clicked, are customized to my preference. In the same function any necessary listeners are added, dictating the map’s behavior. Displaying the data from tables in the map is done by first collecting the information into an XML-styled document. That document is then called when initializing the markers. I will go into more detail of this process later.

**MySQL database**

One of the main points of this project is to showcase my knowledge in the field of information systems. I chose to develop a website, which makes use and interacts with a database. In previous classes, like Information Systems and Web Server Technologies, I have had to work on other projects related to MySQL, therefore I it was much easier for me to work with phpmyadmin than any of the other available software for managing a database. This extremely helpful tool is accessed through the local server and not only lets me create, modify and populate my tables but also allows me to export them in the form of sql files.

**PHP**

For the backend part of my code I heavily relied on PHP, as it allowed me to work with the database, as well as keep track of the server sessions. Learning the different command and syntax of the language was essential for the success of the project. Within the designated php tags, I was able to declare the necessary database info, such as its name, along with the name and password for my server’s phpmyadmin. In addition to that, PHP saves me the trouble of having to write out the same snippets of code on every page by simply including a shared file through the appropriate function, as was the case with the navigation bars or the database connection. All of the SQL queries and most of the functions in my project are also written in PHP. Likewise, PHP is needed for maintaining a user’s session. Without it, the user has to log in over and over again. Useful information, like the username and type of privilege of the account, is stored in the session and can be recalled later on a different page.

**JavaScript**

This language was instrumental in developing the site as intended. For starters, it was necessary for displaying all of the maps and their content. Its other function is concerned with the pop-up modals for registration and logging in, as well as any message alerts that are triggered. Overall, JavaScript helped make the design of the website more professional and stylish.

**Bootstrap**

Speaking of style, as I had mentioned before, it is important that the content in the site is presented in a visually appealing manner. And the easiest way to achieve that was through implementing a bootstrap theme, which I then customized to my liking. Bootstrap is a powerful framework, which allows developer to design their pages faster than ever before. Only in some cases did I need to add extra CSS in the page file. The rest was included through a call to the theme file in the project folder.

**Notepad++**

The entirety of my code was written in this text editor. Even though the software does not appear impressive and is not GUI-oriented like some others, it served its purpose well. Once a file is saved with an html or php extension, the program recognizes the different languages and color codes them accordingly. It is free to install and despite its appearance, is a reliable tool for web development.

**Photoshop CS6**

I had to use Photoshop for editing the navigation banners to fit the design of the website. Not necessary for the project but since I use the software regularly in my spare time, I was able to add an extra feature to the project without having to spend time on learning how to fiddle with it first.

**WAMP**

When I first began working on the project I had to regularly go to the dorm’s computer lab, in order to use the server, provided by the university. However, soon I realized that since I cannot install any new software on those computers, it would be highly inconvenient to use them. What’s more, it would have been impossible to work on the project while I was away for the fall break. Therefore, I decide to install a server on my laptop. WAMP is an environment, which allowed me to test my website and manage my database directly on my personal device.

Installation Requirements

Since the website is not uploaded to a host server, it is not yet available only through a browser and requires a local server. Luckily the installation process is relatively simple. One can download for free the needed file for the WAMP server on their official site:[]. Once the correct version is downloaded, the file is double-clicked and the set up begins automatically. The folder, containing all of my files needs to be placed inside the ‘www’ directory. Usually an icon for WAMP should appear in the bottom right corner of the task bar (or wherever the icons on the person’s device are located). When the icon is clicked, the person can choose to go to the directory or open it on their browser. The URL path to the website should look like this:

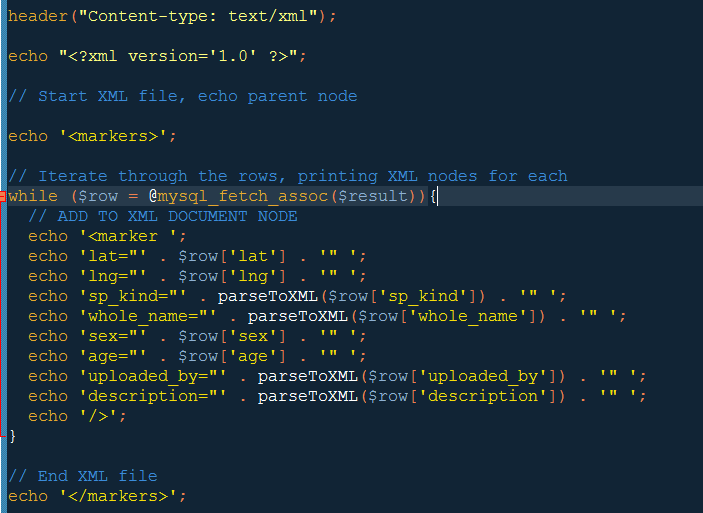


The WAMP icon also gives the user the option to visit phpmyadmin. In the CD I have included the sql files that need to be imported into the database. When phpmyadmin is opened, a database needs to be selected and then on the top menu there should be an option to import files.

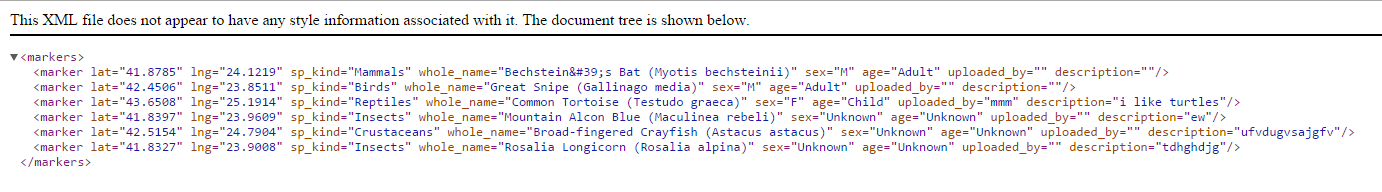
Other Features

**XML conversion**

In a previous section, I discussed the tools, needed to display the information from the database on a google maps. Unfortunately, it is not possible to directly take the data and feed it to the interface. It needs to go through a another process, where the selected table values are converted to XML format. There are a couple fo ways to go about that but I picked an option, which creates a dynamic XML file, which in turn is called in a JavaScript function when creating the map. Te reason I say dynamic is because in other siuations the developer takes the data and stores it in an actual file, located in the same folder as the rest. The map function calls that xml file. The algorithm I have implemented does not make a physical file. Instead the map function uses only the php file, containing the algorithm below.



I can still see how the XML is structured by opening the php file in the browser. The result looks something like this:



**Navigation banners**

Since the top navigation bar changes according to the user, I separated the three cases in different files, which are included in a page after a check is performed to determine if the user is a guest, logged in or an admin. I took advantage of this and created three banners, representing each case. This feature in way influences the functionality of the website, although I believe it is an interesting bonus, which the users can appreciate. Below are screenshots of the three possible navigation bars. The images are of endangered big cats to go along with the theme of the project.

Guest:



Logged in:

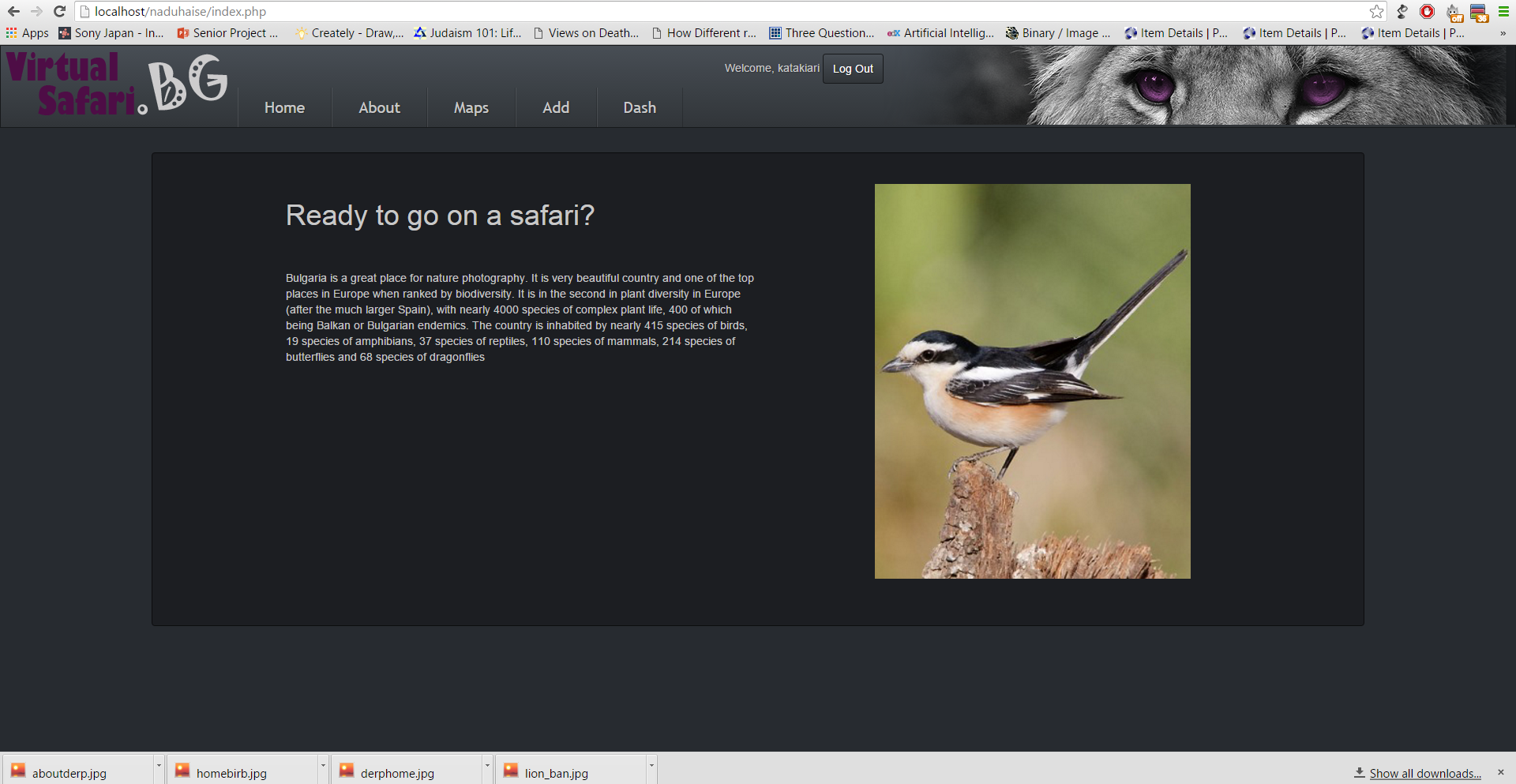


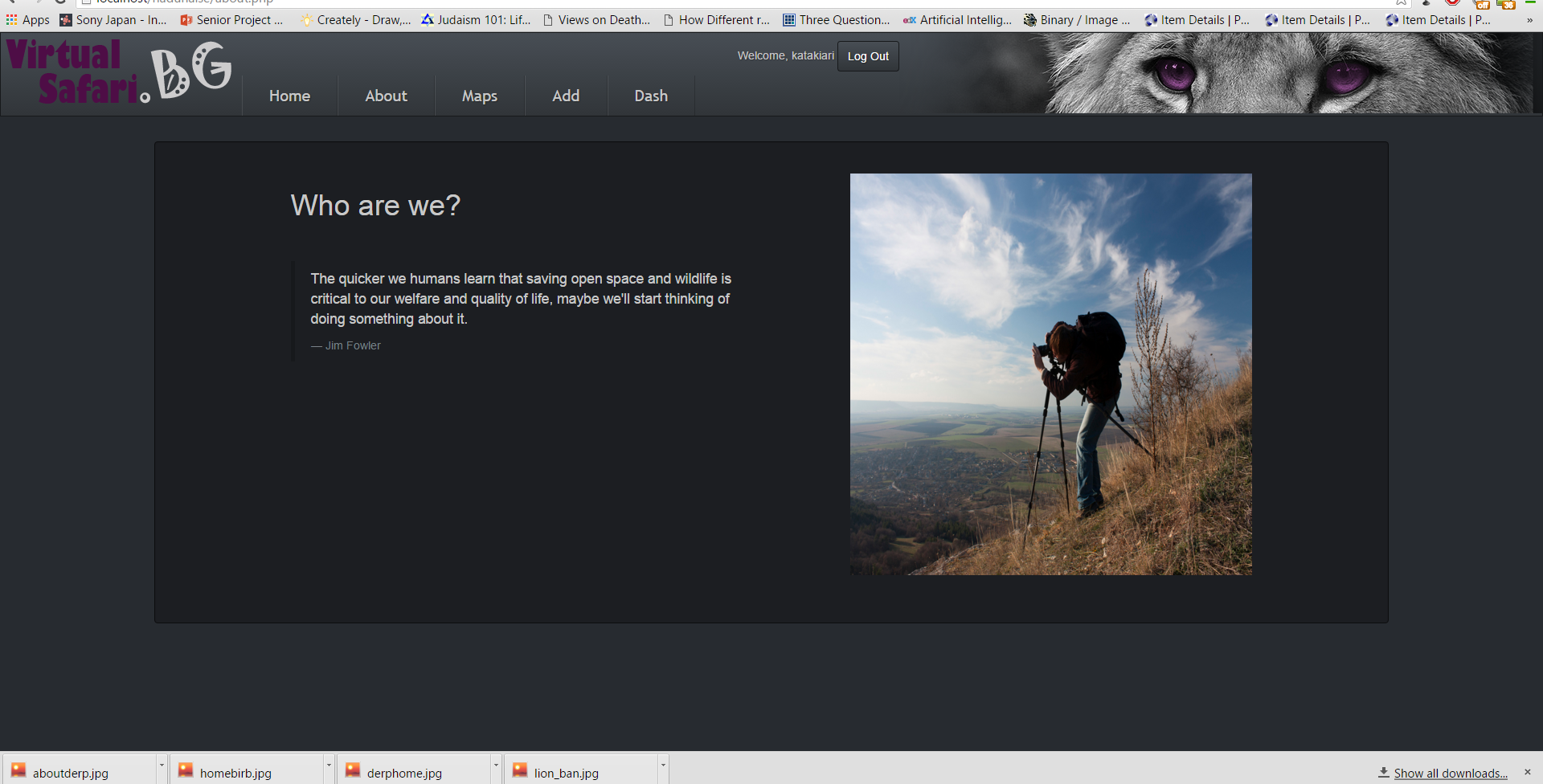
Admin:



**Results and Conclusion**

Screenshots





Project Success

Looking back at the time I spent working on this project, I can say that it has been overall a successful venture. I was able to stick with my initial idea for what to do. All of the functionalities that I set to put in the designing stage were implemented as I had hoped. Not only that, but throughout the course of this semester I was able to include some small extra features to really give the site a better feel. My supervisor and I had scheduled meeting almost every week, where we discussed my progress. Putting tentative deadlines for each stage of the project undoubtedly helped with making sure I produced some results on a regular basis. In the next section I am giving more detailed on the schedule and stages of development.

Schedule

**Week 1-2**

Create database, set up server, basic layout

**Week 3-5**

Add Google maps to site, link database to map

**Week 6-8**

Create add form, create login and register forms

**Week 9-11**

Make admin dashboard, make a forgotten password option, and filter maps

**Week 12-14**

Improve website design, write documentation

Encountered Problems and Solutions

The main problems I faced during this semester mostly had to do with the login and register modals. I thought having a pop-up window for these functions was more professional than directing the user to a separate page. Writing out the code for this part actually took more effort than expected. I had to go back and forth fixing a number of errors. I had to consult online forums to figure out what my mistake was. In the end, I managed to find a workaround that allowed me to do what I had intended. The problem I had was with using the AUBG servers but, as I had explained previously, I installed WAMP on my personal laptop for convenience.

Future Development

For now, the website is only operable through a local server. However, in the future I would like to have it hosted and turn the project into a real site, accessible by everyone. Before doing so, I plan on going back and adding some more functionalities and extra security. Since the site is focused on Bulgarian wild life, I should include a version with Bulgarian translations. Hopefully, I can accumulate a big enough audience to become an educational source for this particular ecological issue.

Conclusion

All in all, I enjoyed working on this project not only because of the initial motivation to raise awareness but also because of the skills I accumulated along the way. Despite its small-scale, this project gave me valuable experience in the field of web development, which will surely help me in future endeavors.

**References**

**W3Schools:**

[**http://www.w3schools.com/**](http://www.w3schools.com/)

**Stack Overflow:**

[**http://stackoverflow.com/**](http://stackoverflow.com/)

**Google Developers:**

[**https://developers.google.com/maps/**](https://developers.google.com/maps/)

**Bootstrap:**

[**http://getbootstrap.com/components/**](http://getbootstrap.com/components/)

**WAMPServer:**

[**http://www.wampserver.com/**](http://www.wampserver.com/)