

1. **Project** : Poodle.com
2. **Student Name** : Himarsha Jayanetti
3. **Student UIN** : 01160219

Poodle.com

Home My Profile Add New Contact Us Logout



Explore
Advanced Search

© 2019 Poodle.com | All rights reserved.

4. Milestone 4 Accomplishment

Table 1 Overview of status for Milestone 2 specifications.

Fulfilled	#	Description
Yes	1	Users can delete items from their favorite list
Yes	2	Items in the favorite lists should be descriptive (can't be just a link) and are linked to an external page or a summary page of the item
Yes	3	The search engine implements at least one of the features spell check, autocomplete, Google Map API, Speech-to-text API , or other APIs permitted by the instructor

5. Architecture

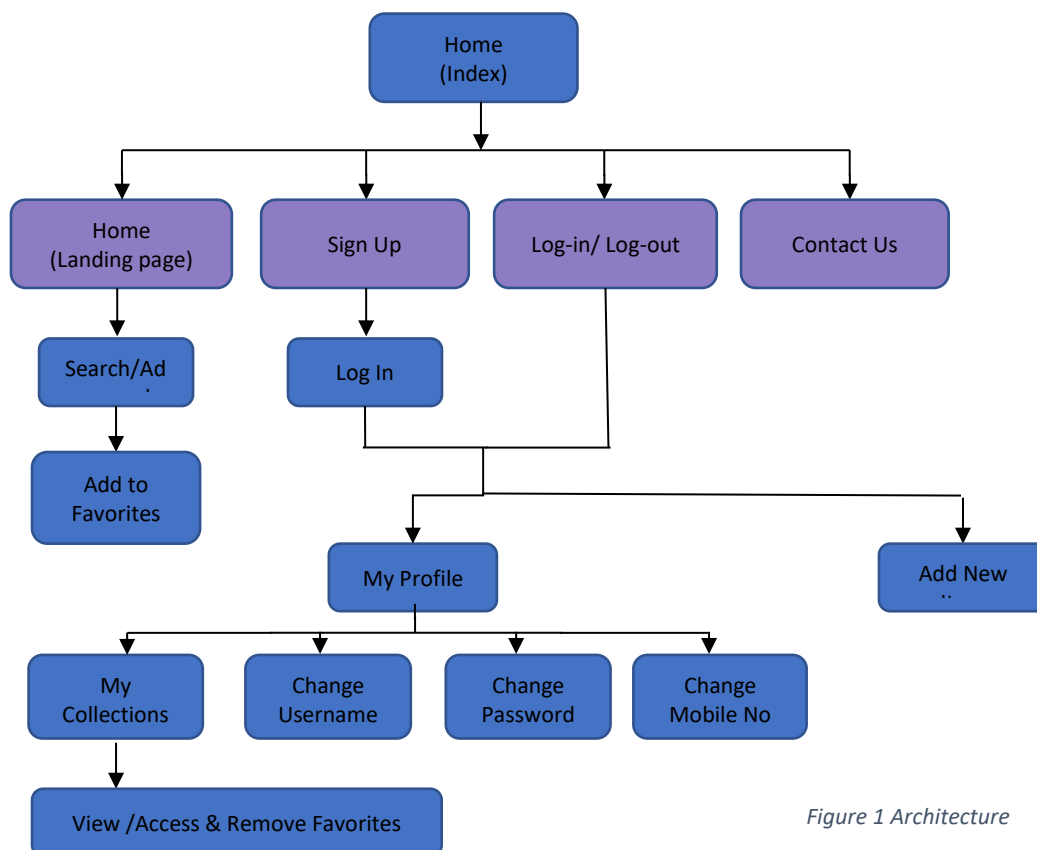


Figure 1 Architecture

The **search function** is implemented by using Elastic search and Php together along with some JavaScript. Home page is the landing page and the navigation bar will be available on all the pages to navigate through to each and every page from wherever you are without any hassle. The goal is to make the number of clicks to get to a page that the user wants as low as possible. User will have access to Home page, Contact page, Login/Logout options from wherever they are on the website. User can log into the account upon signing up on the search engine first. When logged in, he/she will be able to access customized options for the particular user session. 'My Profile' page is having the option to update your user details and maintain a collection of search data according to your preference.

The **front-end** is designed from scratch without the use of any frameworks. The front-end is created using HTML, CSS, and JavaScript to make up the interfaces.

A **database** has been created (namely, userlog) to store the data related to the search engine, handle user records, search, sort, filter and present information based upon web requests from users who access the search engine. The database consists of a table for user information handling (namely, users) which stores information related to a user. The information is collected through the signup form at the initial stage when a user signs up with the search engine. The information that were collected are Name, Email, Mobile, Preferred Username & Password.

6. Data

I have been able to locate the required dataset through Kaggle.com (https://www.kaggle.com/rturley/pet-breed-characteristics#dog_breed_characteristics.csv). Made necessary changes to the fields as required.

Breed Name	Name of the dog breed
Group	Group based on different purposes, uses and characteristics of the dogs. These different groups are used as a way to identify dogs
Weight	Average pup weight
Temperament	Dog's personality, which is also defined as a combination of instinctive and learned behaviors
Price	Average price level (High/Medium/Low)
Popularity	Popularity of the dog breed (High/Medium/Low)
URL	External URL to the AKC site. (https://www.akc.org/)

7. Implementation

4.1 Account registration (including reCAPTCHA and confirmation email, if applicable)

The sign-up page contains the below required fields.

Name: Name of the User.

Email: Email address of the user. Sign up feature is designed in a way that the user will not be able to sign up if the email address is being used already. A simple query will test if the email address is already present in the database during the sign-up process.

Mobile: Users mobile number. Sign-up feature is designed in a way that the user will not be able to sign up if the mobile number is being used already. User can update this information through his/her 'My Profile' page anytime in the future if logged in.

Username: User should setup a username when signing up. Sign up feature is designed in a way that the user will not be able to sign up if the mobile number is being used already This username should be used to login to the site.

Password: User can setup a preferred password (Password must contain digits and/or numbers of minimum 8 characters long)

Once the sign-up form is completed, the user will be sent a confirmation email to the email address used during the sign-up process for account activation. The user will have to click on the link provided in the email to get directed to the login page.

4.2 Account login (including reCAPTCHA)


User can log into the account upon signing up on the search engine first. The username should be a valid username available on the database and the password should match. When logged in, he/she will be able to access customized options for the particular user session. 'My Profile' page is having the option to update your user details, add new record to the system and maintain a collection of search data according to your preference.

The Login and Sign up process is both requiring re-captcha validations. The user will be able to complete the signup process only if the re-captcha is validated. Also, the user is able to login to the account when the re-captcha verification is completed only. I am using Google re-captcha v2 to fulfill this requirement.

Username:

Password:

Forgot your password ? [Click here](#)

☒ I'm not a robot 

Select all images with bridges

Name: Himans

Email: jayash@

Mobile: 75782

Username: marsh

Password: *****

Re-Type Password: *****

Sign Up

Already

In order to implement the **ReCAPTCHA**, I have included the necessary JavaScript resource and a g-recaptcha tag for rendering the reCAPTCHA widget on my page. The g-recaptcha tag is a DIV element with class name g-recaptcha and my site key obtained when registered for the service in the data-sitekey attribute. The script is loaded using the HTTPS protocol and included in the beginning of the code.

```
<script src='https://www.google.com/recaptcha/api.js' async defer></script>
</head>
dy>

<rm action="authenticate_user.php" method="post">
<table cellpadding="6" border="0" align="center">
  <br><br><br>
  <tr><td></td><td><?php if(isset($_GET['error']))
    echo "<p id='error'>".$_GET['error']. "</p>"; ?></td></tr>
  <tr><th>Username:</th><td><input type="text" name="uid"/></td></tr>
  <tr><th>Password:</th><td><input type="password" name="pwd"/></td></tr>
  <tr><td></td><td><input type="submit" value="Login" /></td></tr>
  <tr><td></td><td>Forgot your password ? <a href="resetpassword.php">Click here</a></td></tr>
  <tr><td></td><td><div class="g-recaptcha" data-sitekey="6LeNucEUAAAAAD-AUGZdmaEpdCDdOUuMnCDWNrww"></div></td></tr>
  <tr><td><br><br></td></tr>
```

```
<?php
require 'vendor/autoload.php';
use Elasticsearch\ClientBuilder;
```

```
$hosts = [
    'localhost:9200'
];
```

```
$client = ClientBuilder::create()->setHosts($hosts)->build();
```

4.3 Password reset (including email, if applicable)

The user has the ability to reset the password in case they forget it. User can enter the email address used for sign up process. User will receive

a new link on the email and get prompted to a page where he/she can set a new password. The user will also have the ability to change the password at any time once logged into the account through “My Profile” page.

Reset_token: This is used to store the token created randomly for the forgot password/reset password option. This hashing is done using md5 message digest algorithm which produces a 128-bit hash value.

4.4 Users’ homepage

User has the ability to access his collections, change username, change password and update their mobile number through the home page.

POODLE.COM

SN	Saved Time	Dog Breed	Edit
1	2019-11-21 13:52:04	German Pinscher	Remove
2	2019-11-21 13:53:19	Alaskan Malamute	Remove

4.5 Main search function

The search function is implemented by using Elastic search and Php together along with some JavaScript. The image shows the important line of code used to connect the frontend pages to the Elasticsearch. We should make sure to define the host along with the port number (9200 by default). We should use the composer to connect the php to the Elastic search. We have to include elasticsearch-php in our composer json file. Composer will also create the vendor/autoload.php file that is a requirement for the php to run with Elastic search without any issues. The matching of attributes was done by using the Boolean function and should match option. Once the matching has been completed the search results were saved into a variable called \$response which is of type array. Unless otherwise the search results are converted to a more user-friendly way, the results will be displayed as an array. I have taken several steps to make sure the search results will be displayed in a separate page (SERP) in a more user-readable way than just an array.

4.6 Advanced search function

German Shepherd Explore

Select Group ▼

Select Temperament ▼

Select Intelligence ▼

Select Popularity ▼

Select Price ▼

1.782 seconds to display 251 matches

German Shepherd

Name: German Shepherd Dog
 Group: Herding
 Temperament: Alert
 Popularity: Low , Intelligence: Low
 Additional: can be also grouped as a Herding dog , With an average pup weight of: 35 and a price level: Medium

```

$match_string = ("{$searchterm} {$selected1} {$selected2} {$selected3} {$selected4} {$selected5}");

$params = [
    'index' => 'finaldata',
    'size' => 1000,
    'body' => [
        'query' => [
            'bool' => [
                'should' => [
                    ['match' => ['BreedName' => "{$match_string}"]],
                    ['match' => ['Group' => "{$match_string}"]],
                    ['match' => ['Intelligence' => "{$match_string}"]],
                    ['match' => ['Popularity' => "{$match_string}"]],
                    ['match' => ['Temperament' => "{$match_string}"]],
                    ['match' => ['Group1' => "{$match_string}"]],
                    ['match' => ['Weight' => "{$match_string}"]],
                    ['match' => ['Price' => "{$match_string}"]],
                ],
            ],
        ],
    ],
];

$response = $client->search($params);

```

The advanced search is implemented in a manner where a new match string is created with the combination of all the selections in the advanced search options provided. The user is able to choose one or more specifications in the advanced search options provided and the search results will provide the best match first which fulfills all the criteria.

4.7 SERP

Poodle.com

[Home](#) [My Profile](#) [Add New](#) [Contact Us](#) [Logout](#)

You are looking for: German Shepherd German Shepherd Explore
 1.278 seconds to display 52 matches

[Save Record](#)

Name: German Shepherd Dog
 Group: Herding
 Temperament: Alert
 Popularity: Low , Intelligence: Low
 Can be also grouped as: Herding , With an average pup weight of: 35 and a price level: Medium

[View more info](#)

[Save Record](#)

Name: White German Shepherd
 Group: Herding
 Temperament: Alert
 Popularity: Low , Intelligence: Low
 Can be also grouped as: Herding , With an average pup weight of: 35 and a price level: Medium

The SERP (Search result display page) is used to display all the matching results. The SERP is also having a search box where the user can search again. The results will be displayed on the same page.

The SERP will also display the actual search term user used on top followed by “You are looking for:”, number of matching results and the time taken to display the results.

The SERP will also have a record for each result, and it will contain details about the search results including an external link for more information and a “Save Record” option to save a record to their collections as a favourite.

4.8 XSS vulnerability filtering

There was a requirement for our search engine to prevent XSS vulnerability by removing tags existing in the query. This requirement is achieved by using the simple `strip_tags()` function to eliminate the tags existing while the user input any keyword for searching. I have implemented an additional step to avoid cross-site scripting attacks by using the `trim()` function to validate the input to be a non zero length string. Then the search term is sanitized by removing any html tags it may contain. Also, while displaying the results I have made sure to filter the output by using `htmlspecialchars()` function. These simple steps that were taken will avoid any XSS vulnerability attacks by any user.

```

$searchterm = "" ;

if(isset($_GET["keyword"]))
    #validate searchterm
    $searchterm = trim($_GET['keyword']);
    #sanitize searchterm
    $searchterm = strip_tags($searchterm);

```

4.9 Insert a new entry

[Contact Us](#)
[Logout](#)

Breed:

Main Group: Working/Sporting/Herding/Terrier/Hound

Other Groups:

Temperament: Affectionate/Alert/Active/Friendly/Adaptable

Intelligence: High/Medium/Low

Popularity: High/Medium/Low

Weight: Kg

Price: High/Medium/Low

Add new item

```

if (empty($BreedName)){
    header("Location:../newitem.php?error=First name can't be empty");
    exit();
}

else{
    $params = [
        'index' => 'dogs',
        'id' => $id,
        'body' => [
            'BreedName' => $BreedName,
            'Group' => $Group,
            'Group1' => $Group1,
            'Temperment' => $Temperment,
            'Intelligence' => $Intelligence,
            'Popularity' => $Popularity,
            'Weight' => $Weight,
            'Price' => $Price ]
    ];
}

```

The user is able to add a new item to the search engine. User will have to specify the Breed, Group, Temperament, Intelligence, Popularity, Weight and a Price level for a dog. The user is unable to enter a blank record with no Breed Name. However, the other fields are made optional.

4.10 Pagination

The pagination is implemented using a JavaScript plugin made available via List.js. The script can be embedded anywhere in the code and there are several parameters that we have control over to customize the pagination as we require. The control module fetches ALL records, sort them, and display them all together and the script will be used to organize them across pages.

```

<script src = "https://cdnjs.cloudflare.com/ajax/libs/list.js/1.5.0/list.min.js"></script>
<script>
var options = {
    valueNames: [ 'name', 'category' ],
    page: 10,
    pagination: true,
    innerWindow: 10,
    outerWindow: 10
};

var listObj = new List('listId', options);
</script>

```

(<https://listjs.com/docs/pagination/>)

- **Page** : Number of records per page
- **innerWindow** : How many pages should be visible on each side of the current page.
innerWindow: 2 (Example: ... 3 4 **5** 6 7)
- **outerWindow** : How many pages should be visible on from the beginning and from the end of the pagination.
outerWindow: 2 (Example: 1 2 ... 4 5 **6** 7 8 ... 11 12)

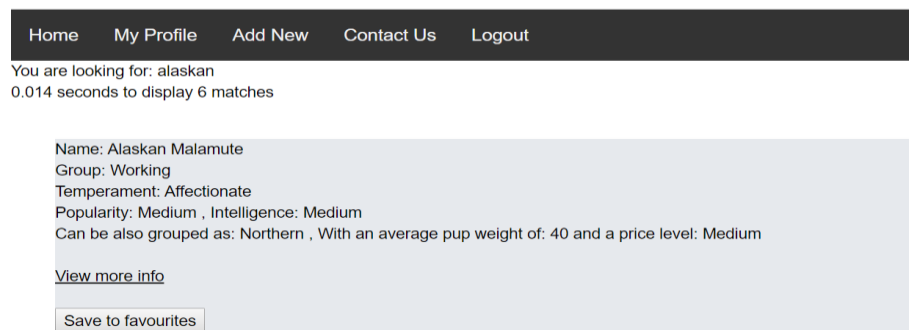
4.11 Highlighting search terms

Describe how the highlight is implemented. The highlight only works in the first page due to the fact that the highlighting function is called when the search term / keyword is initially fetched. But in pagination from the way I have implemented, it does not re-search but re-arrange the already fetched data.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script src="https://cdn.jsdelivr.net/mark.js/8.6.0/jquery.mark.min.js"></script>
<script>
$(document).ready(function() {
$(function() {
var mark = function()
{
var keyword = $("input[name='keyword']").val();
$(".snippet").unmark({
done: function() {
$(".snippet").mark(keyword);
}
});
});
$("input[name='keyword']").on("blur", mark);
});
</script>
```

4.12 Save items to user's profiles

The user has to be logged in to save the documents as favorites. If the user is not logged in the save favorites option will not be showing up. I have made it in such a way that the show result is different for a user in session and a general user.



Log in to your account to save search results to your profile! Don't have an account yet? Sign up here first..

Name: German Pinscher
Group: Working
Temperament: Even-tempered
Popularity: High, Intelligence: Low
Can be also grouped as: Terrier, With an average pup weight of: 14 and a price level: Medium
[View more info](#)

If the user is not logged in the user interface will be like how it shows in the image.

It will prompt a message requesting to login to the account if the user needs to save items as favorites.

The schema of the favorites database table is as the diagram. There is an "id" which is auto incrementing and is the primary key. The uname field is the username which distinguishes my session. I will be saving the document ID and the time when the user save the document as a favorite into the database.

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(100)			No	None		AUTO_INCREMENT	Change Drop More
2	uname	varchar(100)	utf8mb4_0900_ai_ci		No	None			Change Drop More
3	doc_id	varchar(100)	utf8mb4_0900_ai_ci		No	None			Change Drop More
4	item	varchar(100)	utf8mb4_0900_ai_ci		No	None			Change Drop More
5	savetime	varchar(100)	utf8mb4_0900_ai_ci		No	None			Change Drop More

☐ Check all With selected: Browse Change Drop Primary Unique Index Fulltext

4.13 Speech-to-text API

Explore

Advanced Search

```

echo '<table align="right"><br>
<tr><td align="right"><form id="labnol" action="search.php" method="get" >
<input id= "transcript" type="text" value="'. $searchterm. '" size="40" name="keyword" >

<input type="submit" name="sub" value="Explore" />
</form>
</td></tr></table>
';
  
```

```

<script>
function startDictation()
{
  if (window.hasOwnProperty('webkitSpeechRecognition'))
  {
    var recognition = new webkitSpeechRecognition();
    recognition.continuous = false;
    recognition.interimResults = false;
    recognition.lang = "en-US";
    recognition.start();
    recognition.onresult = function(e)
    {
      document.getElementById('transcript').value
      = e.results[0][0].transcript;
      recognition.stop();
      document.getElementById('labnol').submit();
    };
    recognition.onerror = function(e)
    {
      recognition.stop();
    }
  }
}
</script>
  
```

The users are able to use the speech-to-text feature. This enables the recognition and translation of spoken language into text. User can click on the microphone icon on the side of the search box. Once the browser shows up as listening/recording, they can speak to the computer and the spoken language will be translated to text and displayed in the search box. Once completed it will automatically do the search and provide the search results. User is not required click on anything upon completion. The user can type in and use the Explore button to search the normal way.

8. Challenges and Lessons [extra credit 10 points]

- Initially, the most challenging part was to find an appropriate dataset which satisfy all of my requirements. The datasets about dogs were available only towards a more specific area (intelligence, registered names, etc.) It was not possible to find a dataset with all the characteristics of a dog. I used an alternate dataset (accounts.json) to work out the features which was provided as an example in class. Finally, decided to use available datasets along with some fake data added manually to make sure the dataset is complete. Later, I was able to locate the dataset through Kagel and adjusted the features implemented so far with the final dataset.
- It was also quite challenging to figure out how to connect php with Elasticsearch. By revisiting the lecture slides and checking online sources it was easier to overcome that challenge. Also, it was very challenging to get the steps in order to display the search results in a more user-friendly way after you get the search results as an array. No clear reference sources were available on it either. Upon trying different approaches, I managed to overcome that challenge as well.
- I also learned that the display of data is not an easy task as it may look like. It is required to spend more time on improving the design to make sure the search results page will resemble most popular search engine result pages (Google, Bing, etc.)
- Another challenge I faced was during the stage where we had to implement pagination to display the search results. Initially, I have been trying to implement the pagination using the traditional way by defining the parameters and function by myself which was quite challenging. In order to overcome this challenge, I have discovered a JavaScript resource built for pagination and started using it which was less challenging to implement but with the same amount of control over the necessary attributes.
- Another challenge that I faced during this Milestone was the save favorites requirement. The idea was clear and I made sure the save option will be provided to the users only if they are logged in. But the way of actually saving the document and retrieving it was not clearly understood. Upon speaking with the professor during the work-time lectures, the idea of saving the documents using the unique doc id and retrieving them using the same helped a lot in implementing this feature. The database schema was also clearly explained by the professor during the same session.
- The main lesson I learned during the project is that, it is important to have a clear understanding of the requirements as early as possible so that you can clear all your doubts at an earlier stage to be able to meet your deadlines. If I were to do anything differently about what I have done so far, it would be to have the MVC design pattern in mind from the initial stage and apply that to the design of my search engine. Also, I would use bootstrap or any other framework for user interface improvements from day one.

9. Additional comments [extra credit 10 points]

- Have the project tested over docker in future so that the student will be able to learn a new platform and the instructor will be able to test the student implementations more strictly.
- Make the Milestone Requirements and Report Templates available to students through blackboard at the initial stage of the course itself.
- Have the work time sessions before midway presentations, so that the students will have enough time to get help from the instructor and implement their project by the respective milestones.
- Have the grades published in a timely manner so the students will have an idea on where they stand.