Disease diagnosis system

Abstract

A simple website was created for diagnosis of various diseases based on symptoms present with the patient. The website was done using PHP and MySQL and uses fuzzy logic for reasoning in determining the likelihood of each known disease.

Introduction

Let’s say we have a patient with a shoulder problem. The patient complains of pain, stiffness, inability to lift weight, etc. There are many possible diseases, but we want to find out which are the most likely ones based on his symptoms. So we use our website where we input the symptoms and as output we get a rank list of possible diseases, ordered from the most likely one to the least likely one.

Here’s an example of a few of the symptoms that we need to answer about the patient:

1. Ability to lift arm above head
2. Deformation of back
3. Stiffness of joint
4. … etc

And here’s list of a few possible diseases:

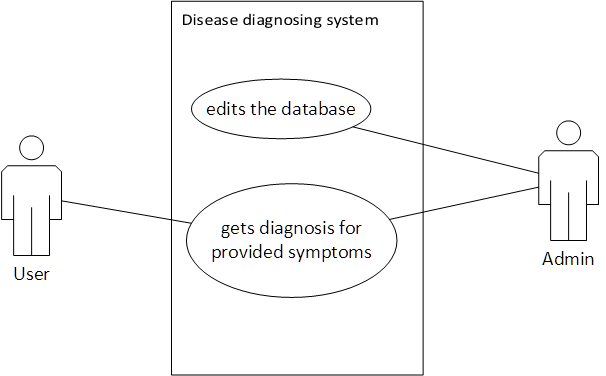
1. Frozen shoulder
2. Profile for winged scapula
3. Shoulder impingement
4. … etc

An important feature of our system is that each symptom needs to be answered with a “fuzzy” value, that means not an objective number like a result from a lab test, but with terms like “Yes”, “No”, “Very much”, “Little”, etc. For example for a symptom like “Stiffness of joint” we might ask the patient “How stiff do you feel your shoulder is?” and the answer might be something like “Little”. So we ask the patient all questions for all symptoms for the sick body part, we input his answers in the system, and the system outputs the possible diseases.

System Design

Our system supports two types of users:

1. Ordinary users – they can input symptom values and obtain a rank list of possible diseases.
2. Administrator – the administrator can edit the database, meaning he can add/edit/delete symptoms, diseases and rules for each disease.



For each body part the system has a set of symptoms and a set of diseases. Then for each disease only a subset of the symptoms are present in the form of rules. Here’s one example:

# Profile for Winged Scapula

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attributes**  **Features** | Very Low | Low | Moderate | High | Very High |
| Deformation of back | No | No | May be | Yes | Yes |
| Ability to lift arm above head | No | May be | Yes | Yes | Yes |
| Pain | No | May be | Yes | Yes | Yes |
| Difficulty lifting heavy objects | No | May be | Yes | Yes | Yes |
| Stiffness of joint | No | No | Maybe | Yes | Yes |
| Pressure when sitting | No | No | Maybe | Yes | Yes |
| Bump in the back of the shoulder | No | No | No | Yes | Yes |

In the example above the disease “Profile for Winged Scapula” has 7 rules, each represented with a symptom. The symptoms are measured using 5 distinct values: “Very Low”, “Low”, “Moderate”, “High”, and “Very High”. The resulting answer for each of the disease based on a given symptom can take 3 values: “No”, “Maybe” and “Yes”. For example if we take the second symptom (Ability to lift arm above head) and we assign it the value “Moderate”, then the resulting answer for the disease will be “Yes”. However in reality not all symptoms will give the same answer, so in order to obtain an answer for the presence of this disease we need to take all answers into account and calculate some type of average. In the end we obtain a result in the following format:

|  |  |
| --- | --- |
| **Disease** | **Score** |
| Frozen Shoulder | 0.933 |
| Profile for Winged Scapula | 0.75 |
| Shoulder impingement | 0.666 |

So the most likely disease is “Frozen Shoulder”, then comes second “Profile for Winged Scapula”, and last comes “Shoulder impingement”. It should be noted that the score which is obtained is not probability, it’s just a score obtained from a fuzzy logic calculation and its only use in our case is to compare the score of one disease to another so that we can determine which one is more likely.

Calculation

The underlying method of calculation of the diagnosis in our system is fuzzy logic computing. The hallmark of fuzzy logic computing is that it’s based on linguistic rules. Example rules in some control system might be:

1. If the speed is high, then decrease the fuel valve.
2. If the speed is very high, then hit the brake.

The first part of each rule (the IF part) is called **antecedent**, and the second part (the THEN part) is called **consequent**.

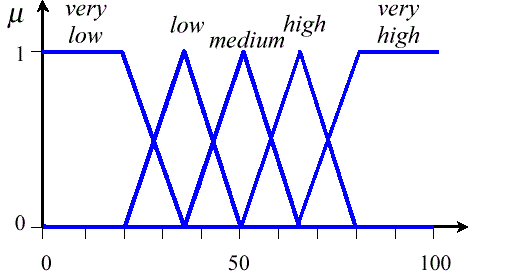
Let’s go through all the steps of the fuzzy computing procedure and see how it’s implemented in our system.

Fuzzification

This is the first step of the computation. Usually some numeric variable is read from a sensor and this variable needs to activate some relevant rules. For example in some system we might have the rules:

1. If the temperature is **very low** then increase the heating **very much**.
2. If the temperature is **low** then increase the heating **moderately**.

Then some value for the temperature is input in the controller, let’s say it’s 30 degrees and this is somewhere between “very low” and “low”, hence it will activate both rules above and possibly others as well. So in order to determine to which degree is each rule applicable we use a fuzzification membership function, let’s say in our example we have these fuzzification membership functions:



The membership functions in our example are of triangular shapes, but they don’t always have to be as such. They can also be slightly curved and take many other forms as well.

In this case it can be seen that the first rule will apply with a degree of 0.34 and the second rule with a degree of 0.66. Rules where the antecedent doesn’t contain the values “very low” and “low” will apply with a degree of 0.0 (meaning they won’t apply).

As a result of this step in a fuzzy logic control system multiple rules will be activated and each will have a certain relevance degree attached to it.

Now let’s turn attention to our musculoskeletal diagnosis system. In our system this step **is not needed and it’s skipped**. In our system there’s no need to fuzzify the input because the user already inputs fuzzy values. For example the symptom “Stiffness of joint” in the table above represents 5 rules and they are:

1. If the stiffness of joint is **very low**, then the winged scapula disease is **not** present.
2. If the stiffness of joint is **low**, then the winged scapula disease is **not** present.
3. If the stiffness of joint is **moderate**, then the winged scapula disease is **maybe** present.
4. If the stiffness of joint is **high**, then the winged scapula disease **is (yes)** present.
5. If the stiffness of joint is **very high**, then the winged scapula disease **is (yes)** present.

When the user selects a value “Very high” the 5th rule will be activated with a degree 1.0 and the other 4 rules with degree 0.0 (that is they will not be activated).

Inference

Here we have multiple rules activated and their results might not all be the same. For example some rule can have a consequent “Increase the heating **much**” and another rule can have “Increase the heating **moderately**”. So we need to find some sort of average of all the rules (not necessarily an arithmetic average) to output as a crisp value to the actuator organ. The average can be calculated in many ways and in fuzzy logic computing the output values are represented again with membership functions. Let’s say the membership functions for the output values “Little”, “Moderately” and “Much” in some control system have the following forms:

Little Moderately Much

0

0.2

0.4

0.6

0.8

1

0

20

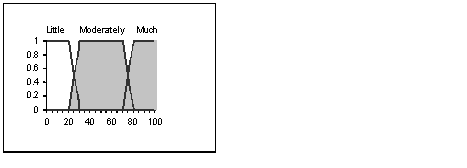
40

60

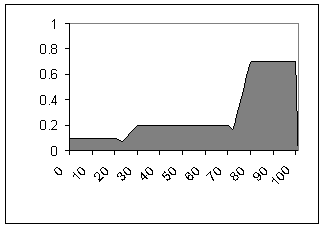
80

100

In order to calculate the average first we need to find some type of union of all the output values. If for example we had one or more “Moderately” values outputted by some of the rules and one or more “Much” values outputted by other rules as well, a plain union would be the grey area:



Another type of union would be where the degree of each variable is taken into account. Let’s say we got “Little” with a degree 0.1, “Moderately” with 0.2 and “Much” with degrees 0.1, 0.2 and 0.4. We can add the 3 degrees for “Much” to get a 0.7, and then use the degree of each value as its multiplier, then the union of the 3 values will have the following form:



Let’s get back to our disease diagnosis system. The output values in our system are “No”, “Maybe” and “Yes”, and we chose their membership functions to be very simple, they look like vertical lines:

No Maybe Yes

0

0.2

0.4

0.6

0.8

1

0.0

0.2

0.4

0.6

0.8

1.0

and they have the following values:

|  |  |
| --- | --- |
| **linguistic output value** | **numeric output value** |
| No | 0.0 |
| Maybe | 0.5 |
| Yes | 1.0 |

When we calculate the union we also take into account the number of occurrences of each value and use it as its weight. Here is an example: let’s say one rule outputted the value “No”, four rules outputted the value “Maybe” and two rules outputted the value “Yes”. Then the union of the 7 output values in our system will be:

|  |  |  |
| --- | --- | --- |
| **linguistic output value** | **numeric output value** | **number of occurences** |
| No | 0.0 | 1 |
| Maybe | 0.5 | 4 |
| Yes | 1.0 | 2 |

Or graphically:

No Maybe Yes

0

1.0

2.0

3.0

4.0

5.0

0.0

0.2

0.4

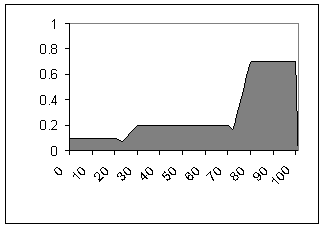
0.6

0.8

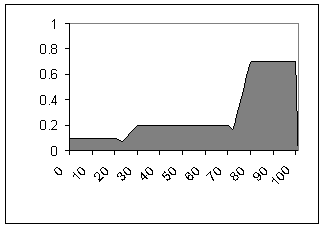
1.0

Defuzzification

This is the process of obtaining a crisp numeric output value from a fuzzy membership function which we obtained as a union of the outputs of multiple rules from the previous step. Multiple methods exist for obtaining this crisp numeric value. Probably the most common one is the **centroid** method and this method finds for crisp output value the center of gravity of the shaded area. That means it finds a value on the horizontal axis such that if we split the shaded surface with a vertical line passing through this value, the surface on the left side will have an equal area to the surface on the right side of the line. In the previous example this value would be around 79:



Another defuzzification method is the **mean-of-maximum** method, and this is the highest value of the shaded area, or if there’s a range of maximum values then it’s the middle of that range. In the last example this would be the middle of the range 80-100 and that is the value 90:



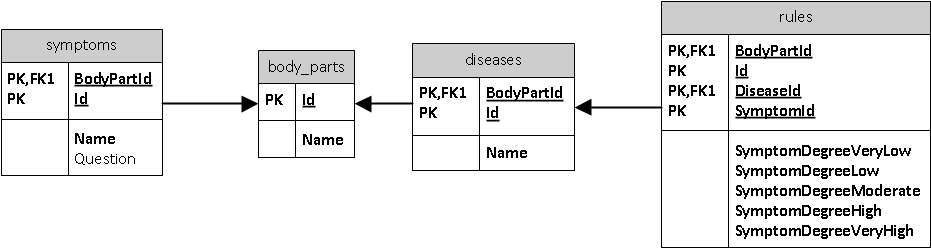
Let’s go back to our musculoskeletal diagnosis system. In our system we used the centroid method to obtain a crisp output value. Since we used vertical lines to represent the membership functions of each output value, the centroid method is very easy to calculate: we just find the arithmetic average of all values (taking into account their relevance degrees as weights). Let’s see that on the example above:

|  |  |  |
| --- | --- | --- |
| **linguistic output value** | **numeric output value** | **number of occurences as a relevance degree** |
| No | 0.0 | 1 |
| Maybe | 0.5 | 4 |
| Yes | 1.0 | 2 |

output = (1 ∙ 0.0 + 4 ∙ 0.5 + 2 ∙ 1.0) / (1 + 4 + 2) = 0.571

Database

The database for this project is relatively simple; it’s got 4 tables only:



The highest level table in the database is body\_parts. An example data of the body\_parts would be:

|  |  |
| --- | --- |
| **Id** | **Name** |
| 1 | Shoulder |
| 2 | Elbow |
| 3 | Hand |
| … | … |

Next is the symptoms table:

|  |  |  |  |
| --- | --- | --- | --- |
| **BodyPartId** | **Id** | **Name** | **Question** |
| 1 | 1 | Deformation of back | Is the patient’s back deformed? |
| 1 | 2 | Loss of motion | Has motion been reduced? |
| 1 | 3 | Pain | Is there any pain on the shoulder? |
| … | … |  |  |
| 2 | 1 | Stiffness of joint | Is the joint stiff? |
| 2 | 2 | Pain | Is there any pain on the elbow? |
| … | … |  |  |

It can be seen that each symptom refers to exactly one body part. This is implemented using foreign keys (also visible on the diagram above). The BodyPartId field of the symptoms table is a foreign key to the Id field of the body\_parts table.

The diseases table is very similar to the symptoms table:

|  |  |  |
| --- | --- | --- |
| **BodyPartId** | **Id** | **Name** |
| 1 | 1 | Profile for Winged Scapula |
| 1 | 2 | Frozen Shoulder |
| 1 | 3 | Shoulder impingement |
| … | … |  |
| 2 | 1 | Cubitus valgus |
| … | … |  |

Last is the rules table:

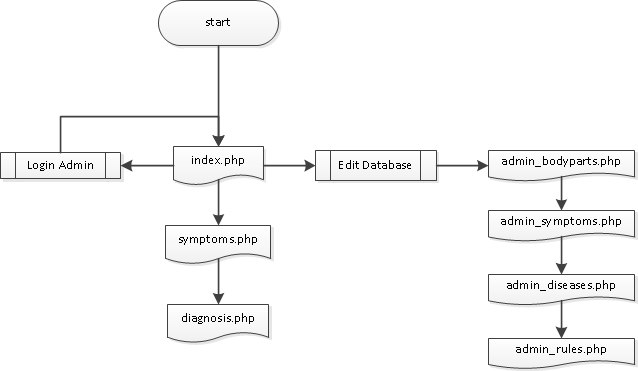
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| BodyPartId | DiseaseId | SymptomId | VeryLow | Low | Moderate | High | VeryHigh | Weight |
| 1 | 1 | 1 | 0.0 | 0.0 | 0.5 | 1.0 | 1.0 | 1.0 |
| 1 | 1 | 2 | 0.0 | 0.5 | 1.0 | 1.0 | 1.0 | 1.0 |
| … |  |  |  |  |  |  |  |  |
| 1 | 2 | 1 | 0.0 | 0.5 | 1.0 | 1.0 | 1.0 | 1.0 |
| … |  |  |  |  |  |  |  |  |

This table uses three foreign keys (BodyPartId, DiseaseId and SymptomId), so each row represents the presence of a certain symptom for determining a certain disease, and all this pertains to one body part only. It can also be seen that each row represents 5 rules (i.e. first rule is: if symptom presence is “VeryLow”, then “No” disease, second rule is: if symptom is “Low”, then “Maybe” disease, etc).

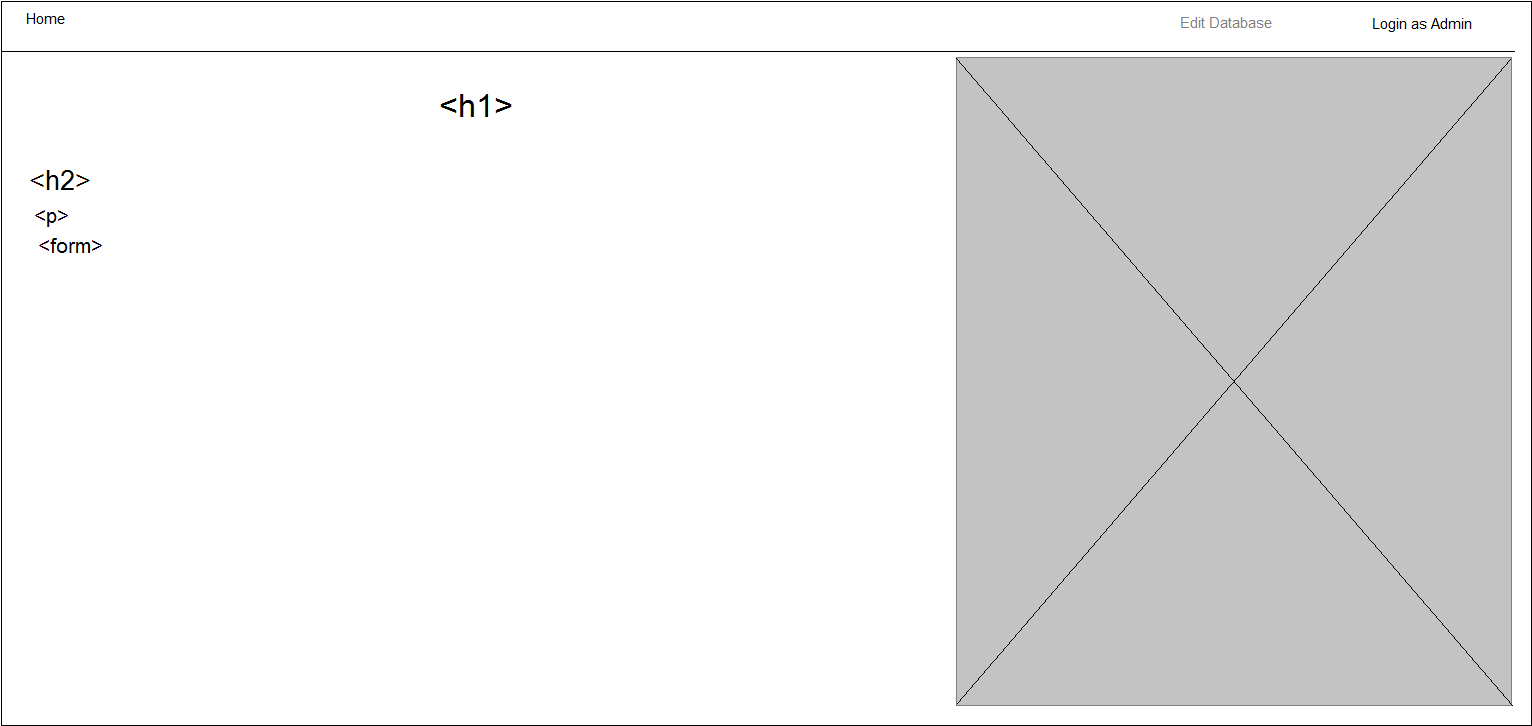
It should be noted that the Weight field of the rules table is not the same as the “degree of relevance” argument used in fuzzy logic computing (although in some cases by coincidence it can have the same effect). Here it merely means that some rules are more important than others so they’re given a higher weight.

Website

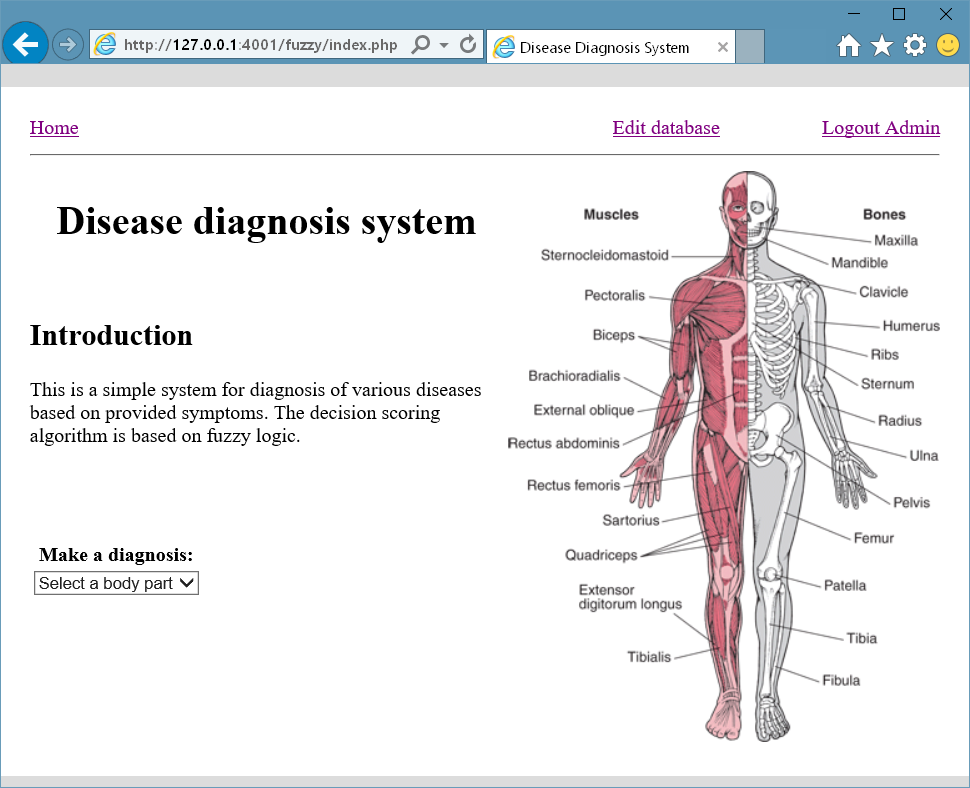
The website is relatively simple, it uses HTML, Javascript and PHP to achieve interactivity, it consists of 7 webpages and on the following dataflow diagram can be seen the data flow of the website:



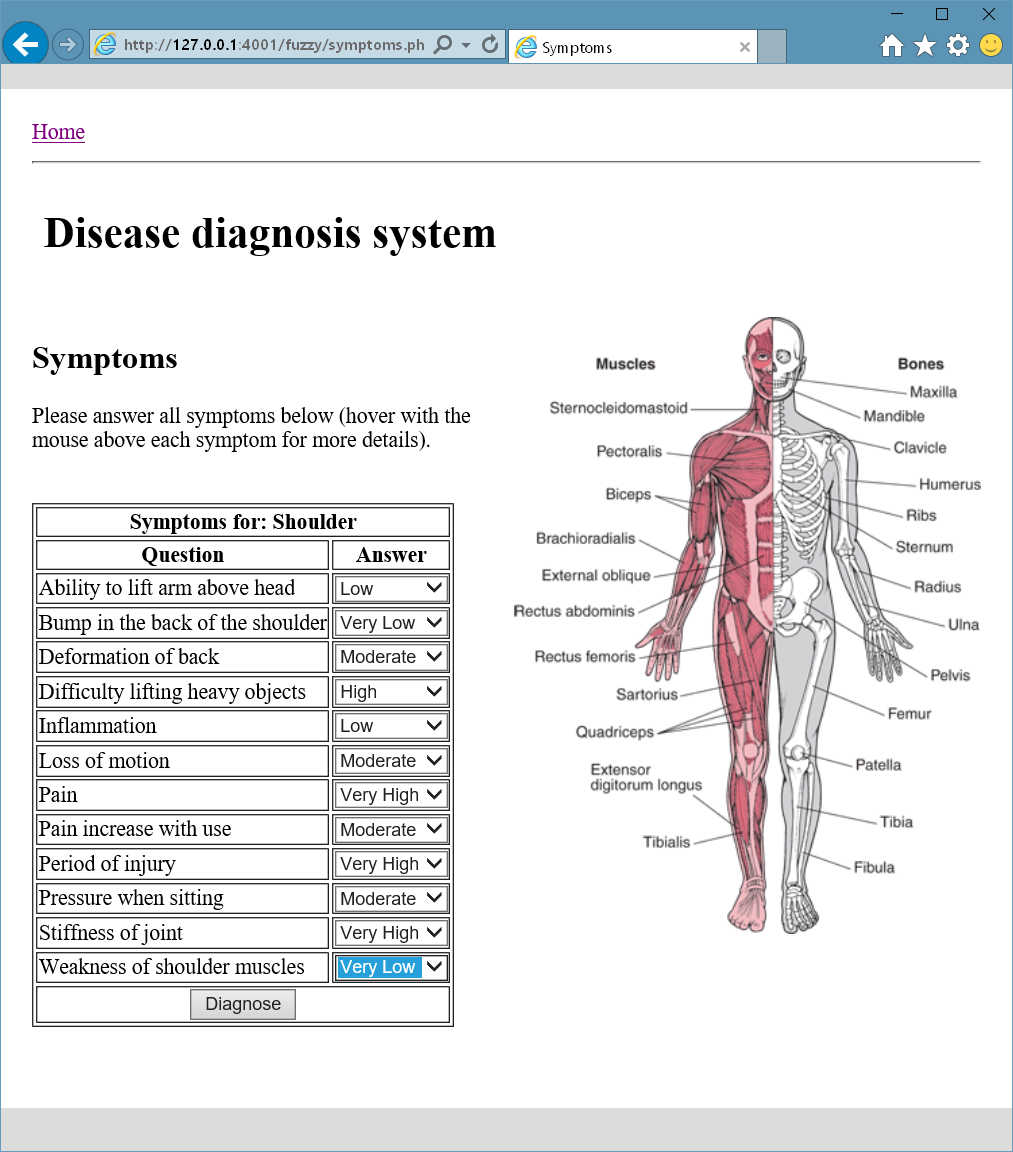
All webpages use a single wireframe for arrangement of the elements:



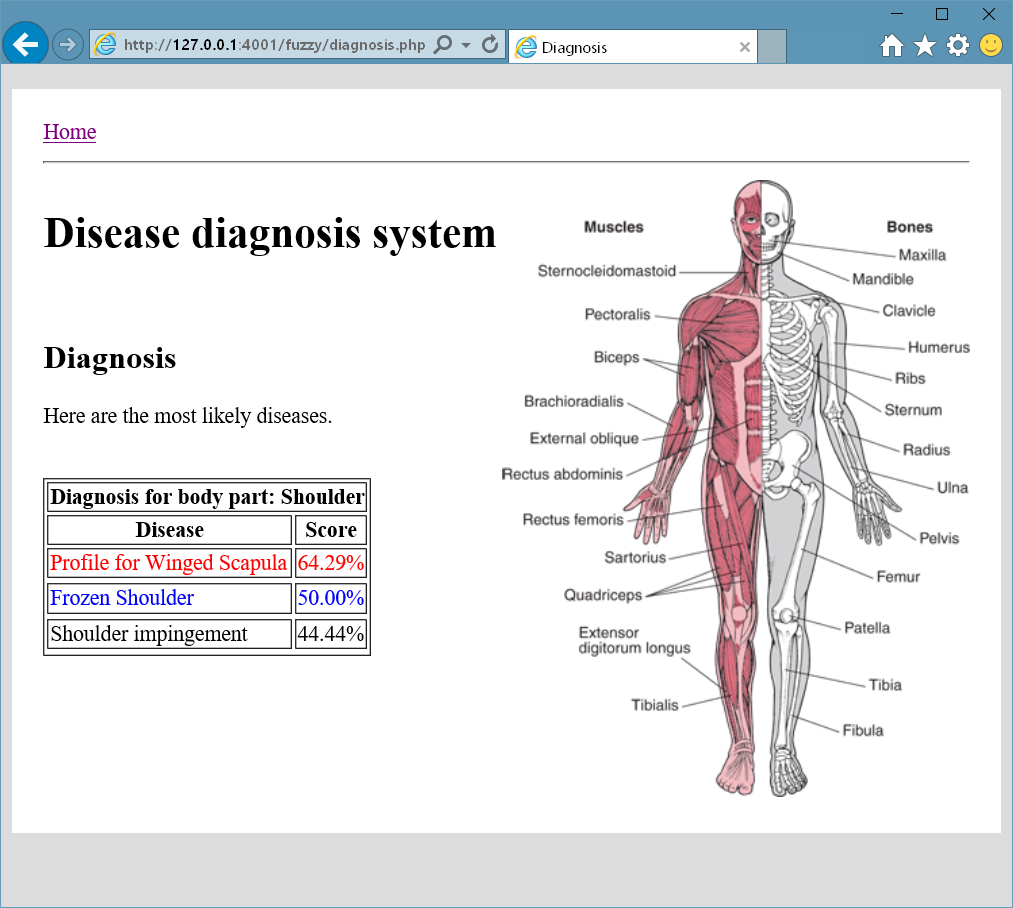
And here are its pages in more detail:



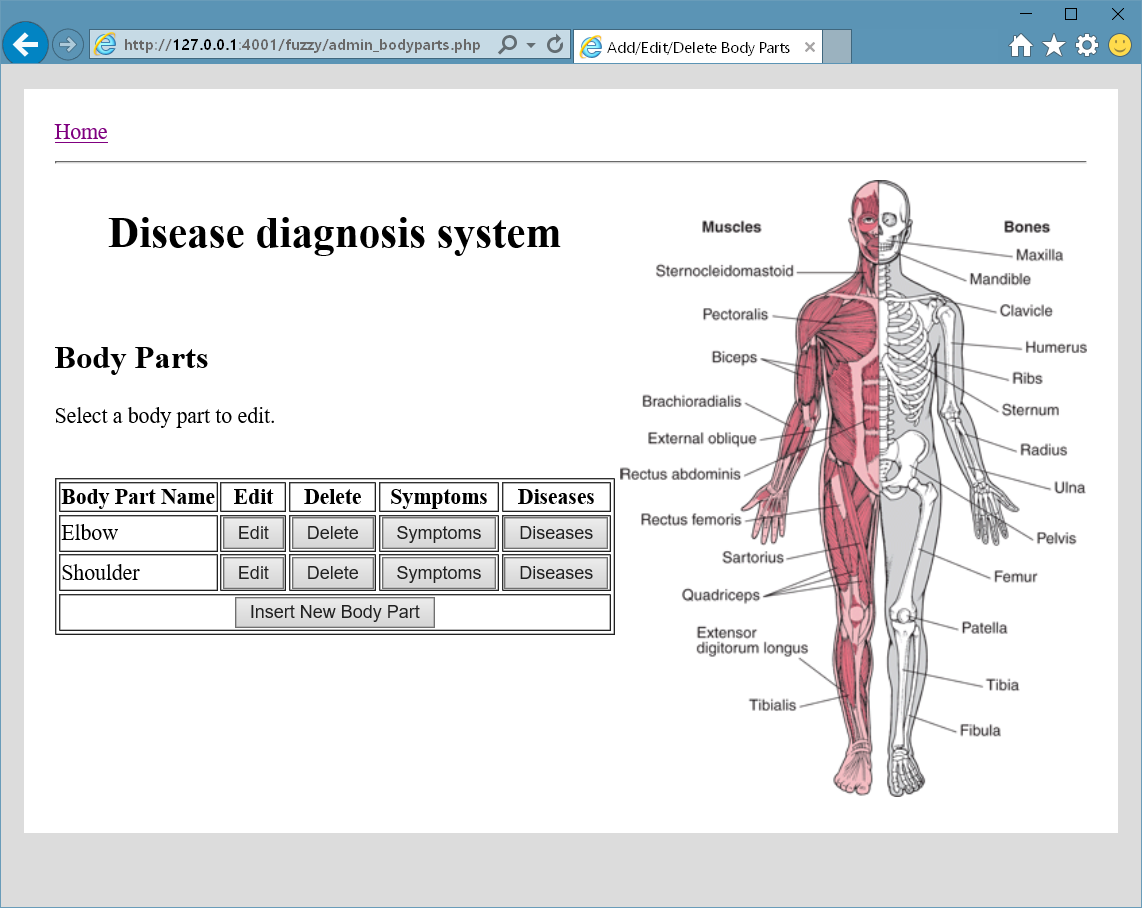
1. index.php – home page and introduction. Here the user can select a body part to proceed with the symptoms and diagnosis, or can login as administrator to edit the database, which is to add/edit/delete bodypart, symptoms, diseases and rules.



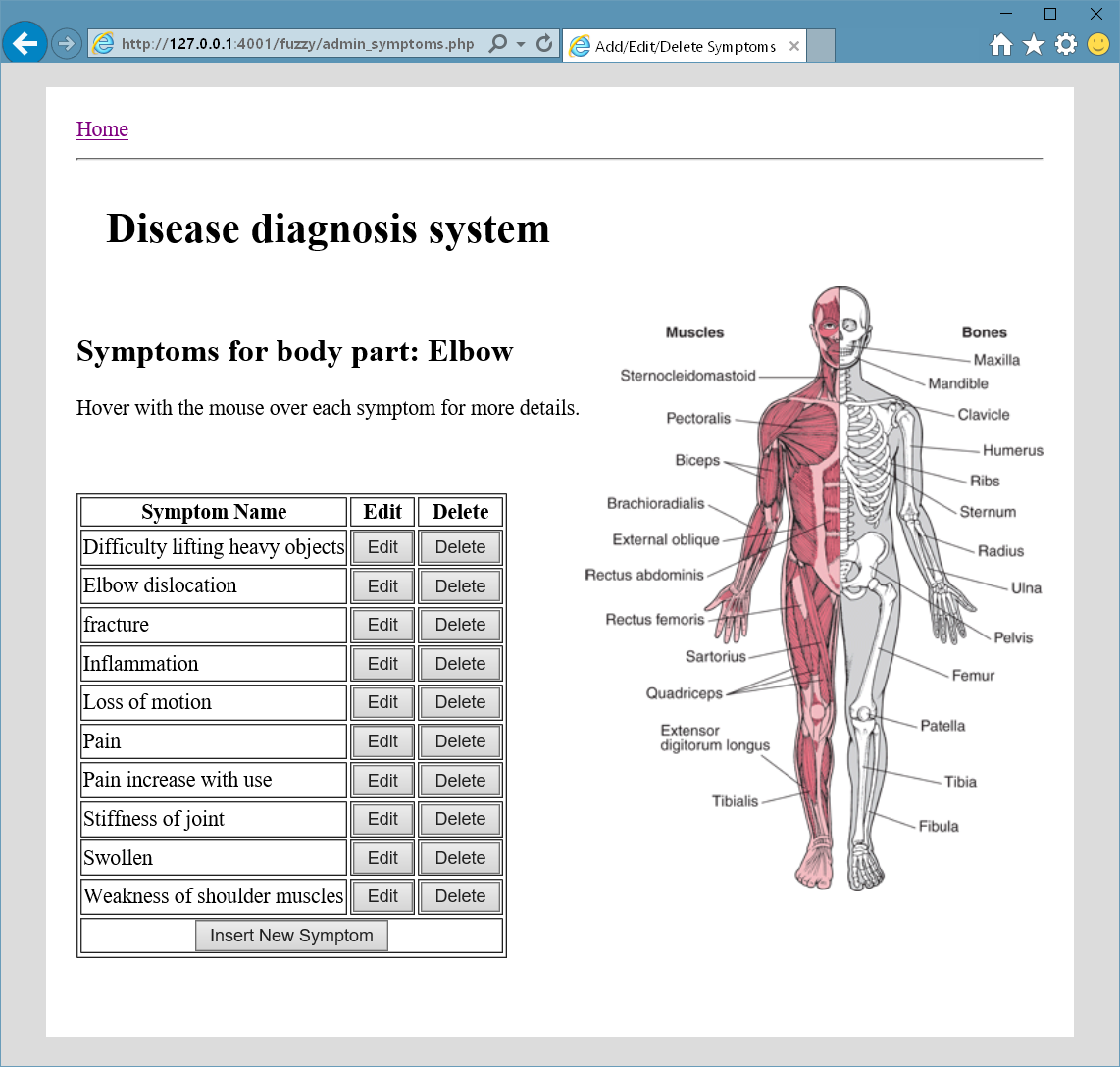
1. symptoms.php – this is the next page where the user can enter the symptoms for the chosen body part.



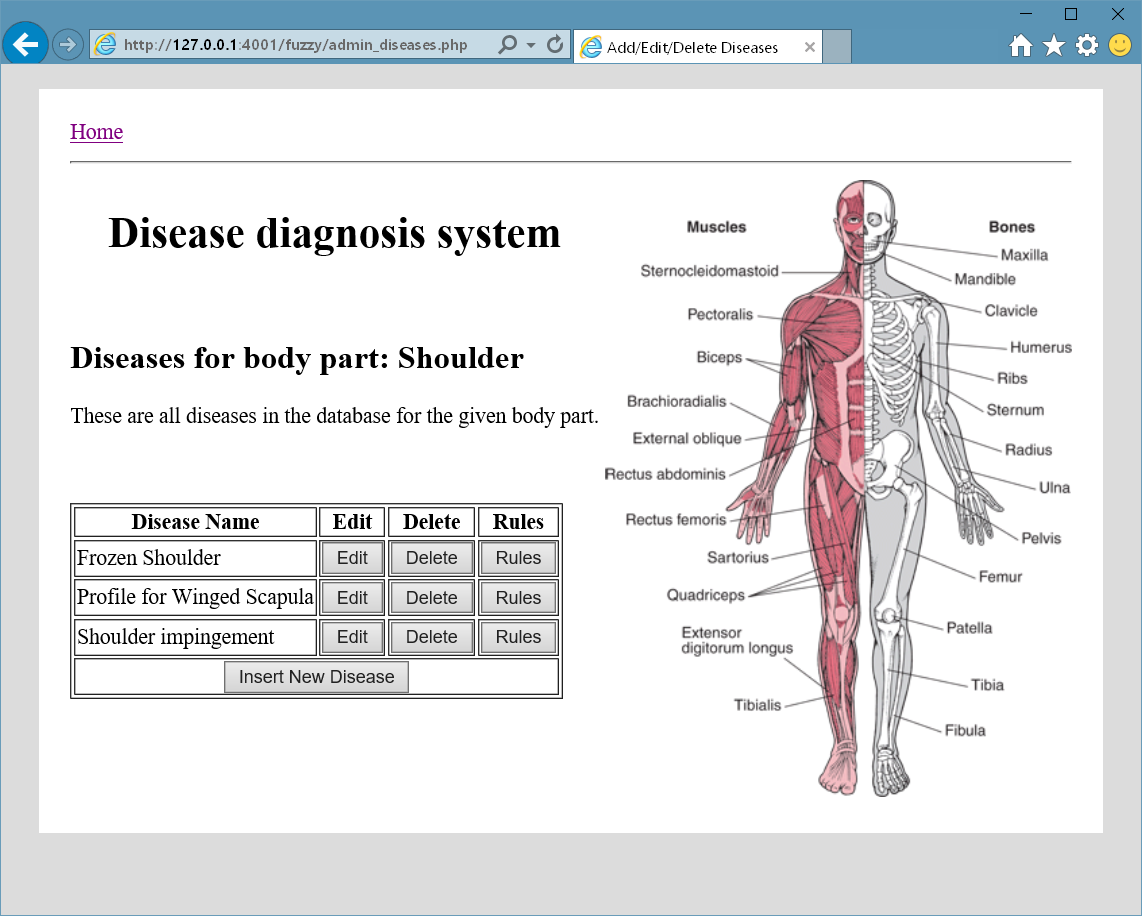
1. diagnosis.php – this is the last page accessible to the ordinary user where he can see the diagnosis output by the website. The diseases are printed in a table in descending order according to their likelihood scores.



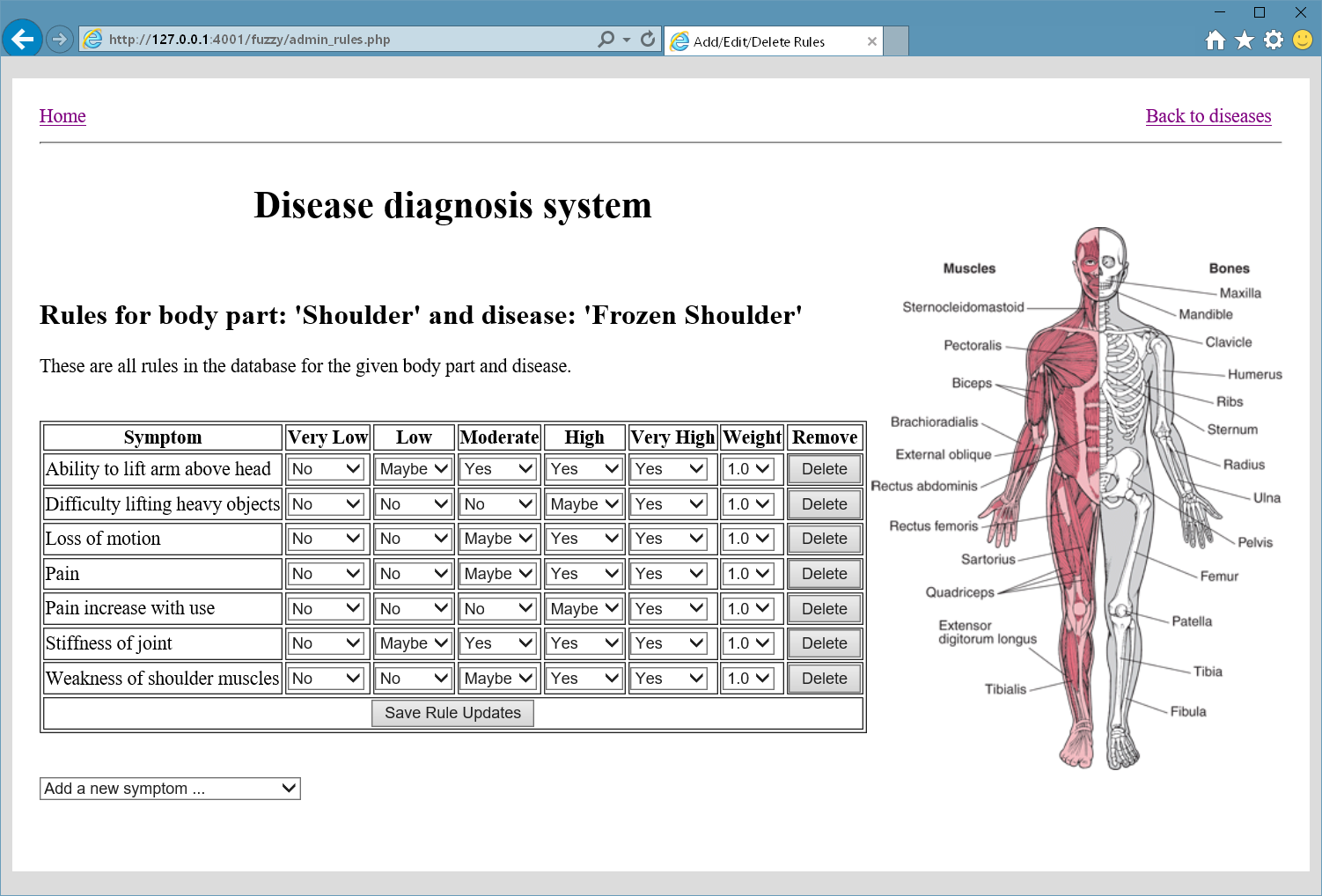
1. admin\_bodyparts.php – this is where the administrator can edit the body parts after he has logged in.



1. admin\_symptoms.php – here the symptoms for the selected body part can be changed. It can be seen that for each symptom there’s a tooltip question displayed when the mouse hovers over the symptom (unfortunately the screenshot doesn’t capture the mouse cursor so the cursor is not shown but it’s over the first symptom in the table above).



1. admin\_diseases.php – here the diseases are entered and this webpage is very similar to the symptoms webpage.



1. admin\_rules.php – here the rules can be changed for a given body part and disease.

Testing

The data input functionality for the administrator works. Whatever data is modified it can be immediately seen on the screen and everything looks good in that part. The only functionality that needs to be checked in detail is the calculation of the fuzzy score. We can easily do this with a specific example:

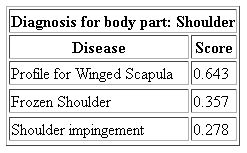
# Profile for Winged Scapula

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attributes**  **Features** | Very Low | Low | Moderate | High | Very High |
| Deformation of back | No | No | May be | Yes | Yes |
| Ability to lift arm above head | No | May be | Yes | Yes | Yes |
| Pain | No | May be | Yes | Yes | Yes |
| Difficulty lifting heavy objects | No | May be | Yes | Yes | Yes |
| Stiffness of joint | No | No | Maybe | Yes | Yes |
| Pressure when sitting | No | No | Maybe | Yes | Yes |
| Bump in the back of the shoulder | No | No | No | Yes | Yes |

The values in green are the values which we have selected as symptoms of some hypothetical patient. The score that we should get is:

(0.5 + 1.0 + 1.0 + 0.5 + 1.0 + 0.5 + 0.0) / 7 = 0.643

When trying it out on the website the result we get is:



The result for “Profile for Winged Scapula” is just as expected.