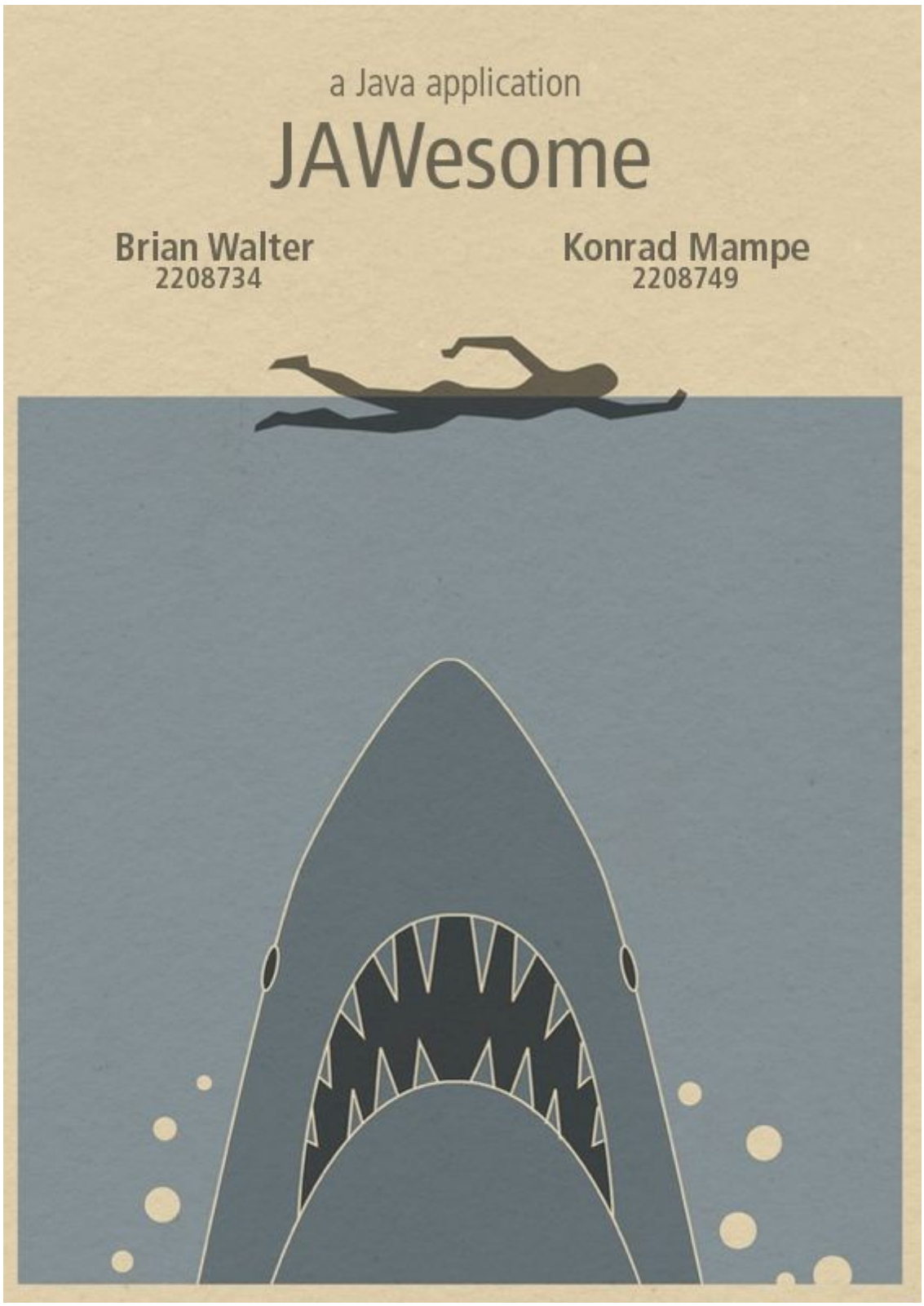
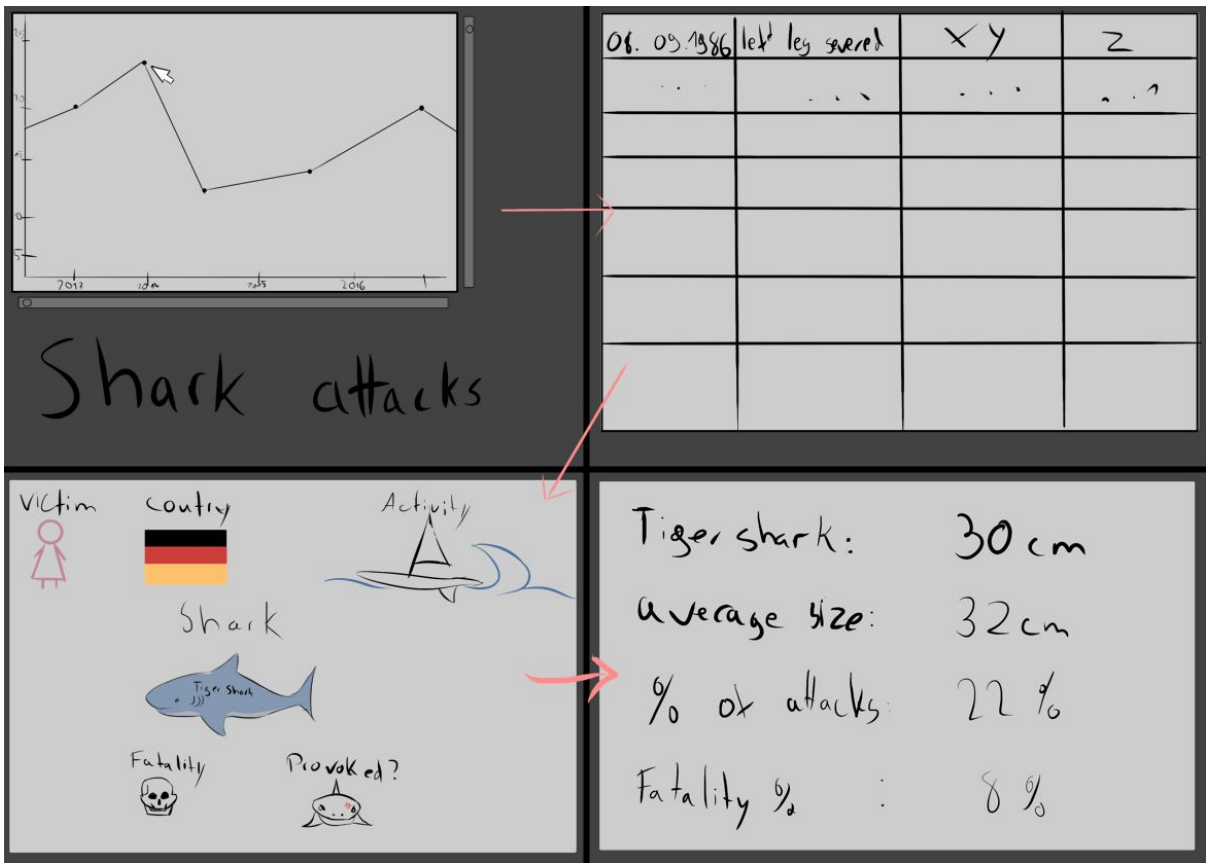


(Abb. A: Original by Brit Morin (https://www.pinterest.com/pin/152066924887678487/))

1. Konzept

Für das Projekt wollten wir uns mit Daten beschäftigen, die uns interessant vorkamen. Bei unserer Suche im Internet sind wir auf eine Website zu Haiangriffen gestoßen, die jedoch nicht besonders übersichtlich gestaltet ist. Für unser Projekt wollten wir also ein Programm entwickeln, dass einem genau sagt wo und unter welchen Umständen, wie viele Haiangriffe vorgefallen sind. Das Programm soll sich an Urlauber richten, die nachschauen möchten, wie gefährlich Ihr Reiseziel ist.

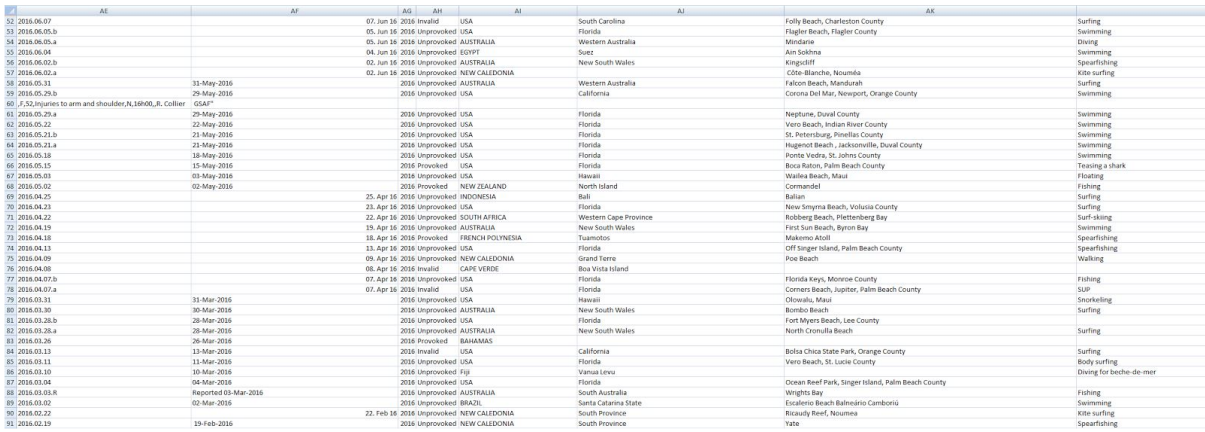
2. Frontend Mockup



(Abb. 1: Unser erster Entwurf)

3. Verwendete Daten

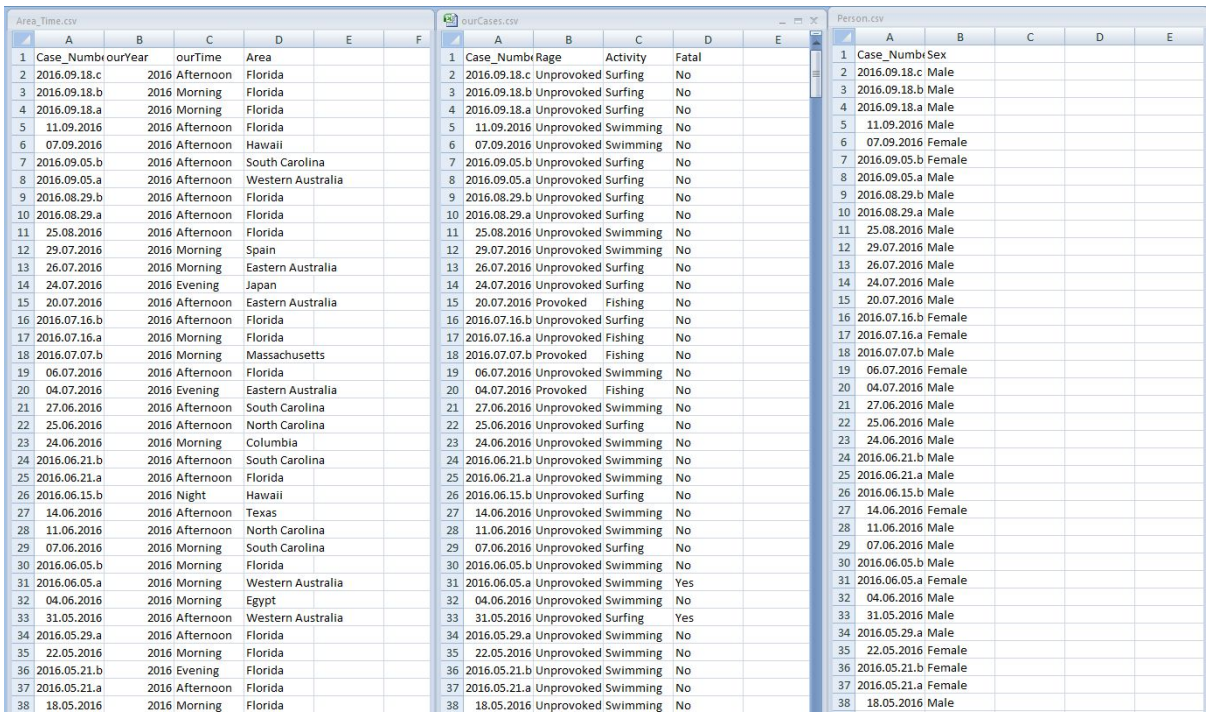
Unsere Daten stammen von der Website www.sharkattackfile.net. Diese Seite gehört dem “Shark Research Institute” und bietet eine Sammelstelle für Organisationen, die irgendwelche Informationen zu Haiangriffen besitzen. Diese Informationen werden alle in einer Exceltabelle gesammelt. Diese Tabelle haben wir uns für unser Projekt zu nutzen gemacht.



	AE	AF	AG	AH	AI	AJ	AK	
52	2016.06.07		07. Jun 16	2016	Invalid	USA	South Carolina	Folly Beach, Charleston County	Surfing
53	2016.06.05.b		05. Jun 16	2016	Unprovoked	USA	Florida	Flagler Beach, Flagler County	Swimming
54	2016.06.05.a		05. Jun 16	2016	Unprovoked	AUSTRALIA	Western Australia	Mindarie	Diving
55	2016.06.04		04. Jun 16	2016	Unprovoked	EGYPT	Suez	Ain Sokhna	Swimming
56	2016.06.02.b		02. Jun 16	2016	Unprovoked	AUSTRALIA	New South Wales	Kingscliff	Spearfishing
57	2016.06.02.a		02. Jun 16	2016	Unprovoked	NEW CALEDONIA		Côte-Blanche, Nouméa	Kite-surfing
58	2016.05.31	31-May-2016		2016	Unprovoked	AUSTRALIA	Western Australia	Falcon Beach, Mandurah	Surfing
59	2016.05.29.b	29-May-2016		2016	Unprovoked	USA	California	Corona Del Mar, Newport, Orange County	Swimming
60	,F,52,Injuries to arm and shoulder,N,16h00,,R. Collier	GSAF"							
61	2016.05.29.a	29-May-2016		2016	Unprovoked	USA	Florida	Neptune, Duval County	Swimming
62	2016.05.22	22-May-2016		2016	Unprovoked	USA	Florida	Vero Beach, Indian River County	Swimming
63	2016.05.21.b	21-May-2016		2016	Unprovoked	USA	Florida	St. Petersburg, Pinellas County	Swimming
64	2016.05.21.a	21-May-2016		2016	Unprovoked	USA	Florida	Hugenot Beach , Jacksonville, Duval County	Swimming
65	2016.05.18	18-May-2016		2016	Unprovoked	USA	Florida	Ponte Vedra, St. Johns County	Swimming
66	2016.05.15	15-May-2016		2016	Provoked	USA	Florida	Boca Raton, Palm Beach County	Teasing a shark
67	2016.05.03	03-May-2016		2016	Unprovoked	USA	Hawaii	Wailea Beach, Maui	Floating
68	2016.05.02	02-May-2016		2016	Provoked	NEW ZEALAND	North Island	Cormandel	Fishing
69	2016.04.25		25. Apr 16	2016	Unprovoked	INDONESIA	Bali	Balian	Surfing
70	2016.04.23		23. Apr 16	2016	Unprovoked	USA	Florida	New Smyrna Beach, Volusia County	Surfing
71	2016.04.22		22. Apr 16	2016	Unprovoked	SOUTH AFRICA	Western Cape Province	Robberg Beach, Plettenberg Bay	Surf-skiing
72	2016.04.19		19. Apr 16	2016	Unprovoked	AUSTRALIA	New South Wales	First Sun Beach, Byron Bay	Swimming
73	2016.04.18		18. Apr 16	2016	Provoked	FRENCH POLYNESIA	Tuamotos	Makemo Atoll	Spearfishing
74	2016.04.13		13. Apr 16	2016	Unprovoked	USA	Florida	Off Singer Island, Palm Beach County	Spearfishing
75	2016.04.09		09. Apr 16	2016	Unprovoked	NEW CALEDONIA	Grand Terre	Poe Beach	Walking
76	2016.04.08		08. Apr 16	2016	Invalid	CAPE VERDE	Boa Vista Island		
77	2016.04.07.b		07. Apr 16	2016	Unprovoked	USA	Florida	Florida Keys, Monroe County	Fishing
78	2016.04.07.a		07. Apr 16	2016	Invalid	USA	Florida	Corners Beach, Jupiter, Palm Beach County	SUP
79	2016.03.31	31-Mar-2016		2016	Unprovoked	USA	Hawaii	Olowalu, Maui	Snorkeling
80	2016.03.30	30-Mar-2016		2016	Unprovoked	AUSTRALIA	New South Wales	Bombo Beach	Surfing
81	2016.03.28.b	28-Mar-2016		2016	Unprovoked	USA	Florida	Fort Myers Beach, Lee County	
82	2016.03.28.a	28-Mar-2016		2016	Unprovoked	AUSTRALIA	New South Wales	North Cronulla Beach	Surfing
83	2016.03.26	26-Mar-2016		2016	Provoked	BAHAMAS			
84	2016.03.13	13-Mar-2016		2016	Invalid	USA	California	Bolsa Chica State Park, Orange County	Surfing
85	2016.03.11	11-Mar-2016		2016	Unprovoked	USA	Florida	Vero Beach, St. Lucie County	Body surfing
86	2016.03.10	10-Mar-2016		2016	Unprovoked	Fiji	Vanua Levu		Diving for beche-de-mer
87	2016.03.04	04-Mar-2016		2016	Unprovoked	USA	Florida	Ocean Reef Park, Singer Island, Palm Beach County	
88	2016.03.03.R	Reported 03-Mar-2016		2016	Unprovoked	AUSTRALIA	South Australia	Wrights Bay	Fishing
89	2016.03.02	02-Mar-2016		2016	Unprovoked	BRAZIL	Santa Catarina State	Escalerio Beach Balneário Camboriú	Swimming
90	2016.02.22		22. Feb 16	2016	Unprovoked	NEW CALEDONIA	South Province	Ricaudy Reef, Noumea	Kite-surfing
91	2016.02.19	19-Feb-2016		2016	Unprovoked	NEW CALEDONIA	South Province	Yate	Spearfishing

(Abb. 2: So sahen die Daten in der erwähnten Tabelle aus)

Anhand dem oberen Bild kann man erkennen, dass die Daten nicht sehr gut gepflegt sind. In manchen Spalten stehen Informationen, die eigentlich woanders hineingehören und manchmal machen bestimmte Informationen einfach keinen Sinn. Deshalb mussten wir diese Tabelle erst mal etwas säubern. Da die Datei aber über 10.000 Einträge hat, haben wir uns auf die Einträge der Jahre 2010 bis 2016 konzentriert. Diese Tabelle musste auch erst mal durch den Normalisierungsprozess durch, aber dazu später mehr.
Was wir dann mit den Einträgen getan haben, ist ungültige Eingaben zu entfernen, alles möglichst einheitlich zu haben und Dinge zusammenzufassen, z. B. in der Spalte Aktivitäten haben wir alle Arten von Angeln zu einem Oberbegriff zusammengefasst und in der Zeiten-Spalte alle Einträge in die vier Tageszeiten verallgemeinert. Dadurch wurde es auch einfacher für uns mit den Daten zu arbeiten und die technischen Hintergründe zu programmieren.



Area_Time.csv

	A	B	C	D	E	F
1	Case_Numbe	ourYear	ourTime	Area		
2	2016.09.18.c	2016	Afternoon	Florida		
3	2016.09.18.b	2016	Morning	Florida		
4	2016.09.18.a	2016	Morning	Florida		
5	11.09.2016	2016	Afternoon	Florida		
6	07.09.2016	2016	Afternoon	Hawaii		
7	2016.09.05.b	2016	Afternoon	South Carolina		
8	2016.09.05.a	2016	Afternoon	Western Australia		
9	2016.08.29.b	2016	Afternoon	Florida		
10	2016.08.29.a	2016	Afternoon	Florida		
11	25.08.2016	2016	Afternoon	Florida		
12	29.07.2016	2016	Morning	Spain		
13	26.07.2016	2016	Morning	Eastern Australia		
14	24.07.2016	2016	Evening	Japan		
15	20.07.2016	2016	Afternoon	Eastern Australia		
16	2016.07.16.b	2016	Afternoon	Florida		
17	2016.07.16.a	2016	Morning	Florida		
18	2016.07.07.b	2016	Morning	Massachusetts		
19	06.07.2016	2016	Afternoon	Florida		
20	04.07.2016	2016	Evening	Eastern Australia		
21	27.06.2016	2016	Afternoon	South Carolina		
22	25.06.2016	2016	Afternoon	North Carolina		
23	24.06.2016	2016	Morning	Columbia		
24	2016.06.21.b	2016	Afternoon	South Carolina		
25	2016.06.21.a	2016	Afternoon	Florida		
26	2016.06.15.b	2016	Night	Hawaii		
27	14.06.2016	2016	Afternoon	Texas		
28	11.06.2016	2016	Afternoon	North Carolina		
29	07.06.2016	2016	Morning	South Carolina		
30	2016.06.05.b	2016	Morning	Florida		
31	2016.06.05.a	2016	Morning	Western Australia		
32	04.06.2016	2016	Morning	Egypt		
33	31.05.2016	2016	Afternoon	Western Australia		
34	2016.05.29.a	2016	Afternoon	Florida		
35	22.05.2016	2016	Morning	Florida		
36	2016.05.21.b	2016	Evening	Florida		
37	2016.05.21.a	2016	Afternoon	Florida		
38	18.05.2016	2016	Morning	Florida		

ourCases.csv

	A	B	C	D	E
1	Case_Numbe	Rage	Activity	Fatal	
2	2016.09.18.c	Unprovoked	Surfing	No	
3	2016.09.18.b	Unprovoked	Surfing	No	
4	2016.09.18.a	Unprovoked	Surfing	No	
5	11.09.2016	Unprovoked	Swimming	No	
6	07.09.2016	Unprovoked	Swimming	No	
7	2016.09.05.b	Unprovoked	Surfing	No	
8	2016.09.05.a	Unprovoked	Surfing	No	
9	2016.08.29.b	Unprovoked	Surfing	No	
10	2016.08.29.a	Unprovoked	Surfing	No	
11	25.08.2016	Unprovoked	Swimming	No	
12	29.07.2016	Unprovoked	Swimming	No	
13	26.07.2016	Unprovoked	Surfing	No	
14	24.07.2016	Unprovoked	Surfing	No	
15	20.07.2016	Provoked	Fishing	No	
16	2016.07.16.b	Unprovoked	Surfing	No	
17	2016.07.16.a	Unprovoked	Fishing	No	
18	2016.07.07.b	Provoked	Fishing	No	
19	06.07.2016	Unprovoked	Swimming	No	
20	04.07.2016	Provoked	Fishing	No	
21	27.06.2016	Unprovoked	Swimming	No	
22	25.06.2016	Unprovoked	Surfing	No	
23	24.06.2016	Unprovoked	Swimming	No	
24	2016.06.21.b	Unprovoked	Swimming	No	
25	2016.06.21.a	Unprovoked	Swimming	No	
26	2016.06.15.b	Unprovoked	Surfing	No	
27	14.06.2016	Unprovoked	Swimming	No	
28	11.06.2016	Unprovoked	Swimming	No	
29	07.06.2016	Unprovoked	Surfing	No	
30	2016.06.05.b	Unprovoked	Swimming	No	
31	2016.06.05.a	Unprovoked	Swimming	Yes	
32	04.06.2016	Unprovoked	Swimming	No	
33	31.05.2016	Unprovoked	Surfing	Yes	
34	2016.05.29.a	Unprovoked	Swimming	No	
35	22.05.2016	Unprovoked	Swimming	No	
36	2016.05.21.b	Unprovoked	Swimming	No	
37	2016.05.21.a	Unprovoked	Swimming	No	
38	18.05.2016	Unprovoked	Swimming	No	

Person.csv

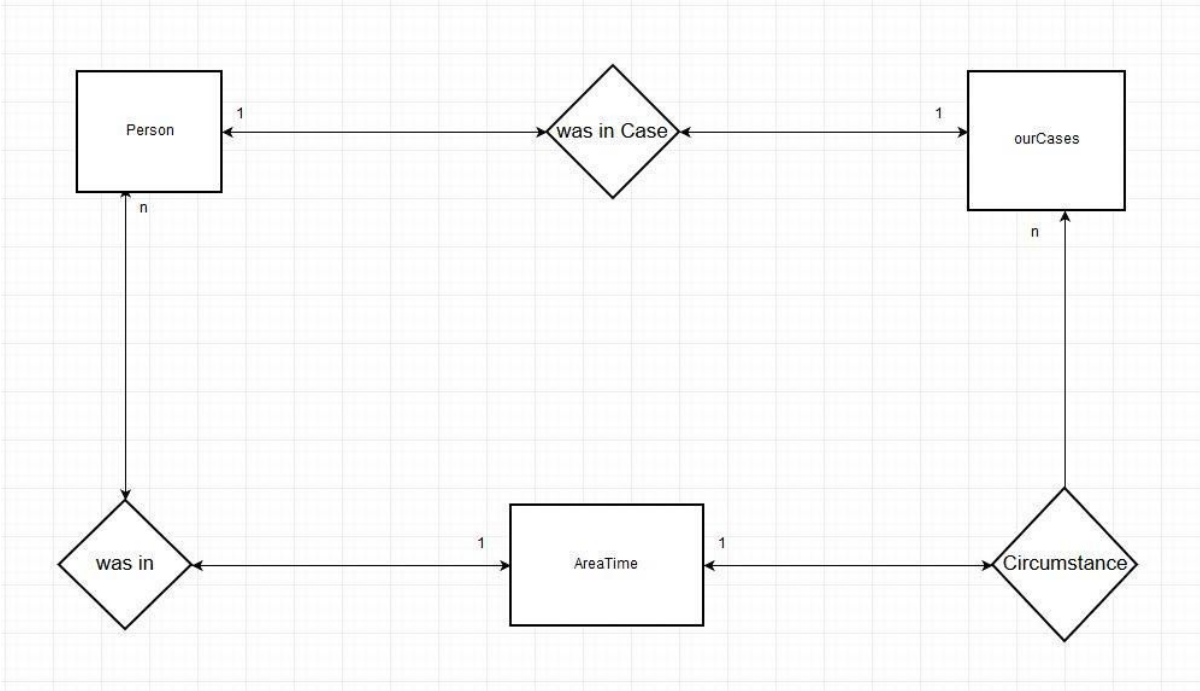
	A	B	C	D	E
1	Case_Numbe	Sex			
2	2016.09.18.c	Male			
3	2016.09.18.b	Male			
4	2016.09.18.a	Male			
5	11.09.2016	Male			
6	07.09.2016	Female			
7	2016.09.05.b	Female			
8	2016.09.05.a	Male			
9	2016.08.29.b	Male			
10	2016.08.29.a	Male			
11	25.08.2016	Male			
12	29.07.2016	Male			
13	26.07.2016	Male			
14	24.07.2016	Male			
15	20.07.2016	Male			
16	2016.07.16.b	Female			
17	2016.07.16.a	Female			
18	2016.07.07.b	Male			
19	06.07.2016	Female			
20	04.07.2016	Male			
21	27.06.2016	Male			
22	25.06.2016	Male			
23	24.06.2016	Male			
24	2016.06.21.b	Male			
25	2016.06.21.a	Male			
26	2016.06.15.b	Male			
27	14.06.2016	Female			
28	11.06.2016	Male			
29	07.06.2016	Male			
30	2016.06.05.b	Male			
31	2016.06.05.a	Female			
32	04.06.2016	Male			
33	31.05.2016	Male			
34	2016.05.29.a	Male			
35	22.05.2016	Female			
36	2016.05.21.b	Female			
37	2016.05.21.a	Female			
38	18.05.2016	Male			

(Abb. 3: Die .csv nachdem wir diese überarbeitet haben)

4. ER-Modelle

Entity-Relationship-Model

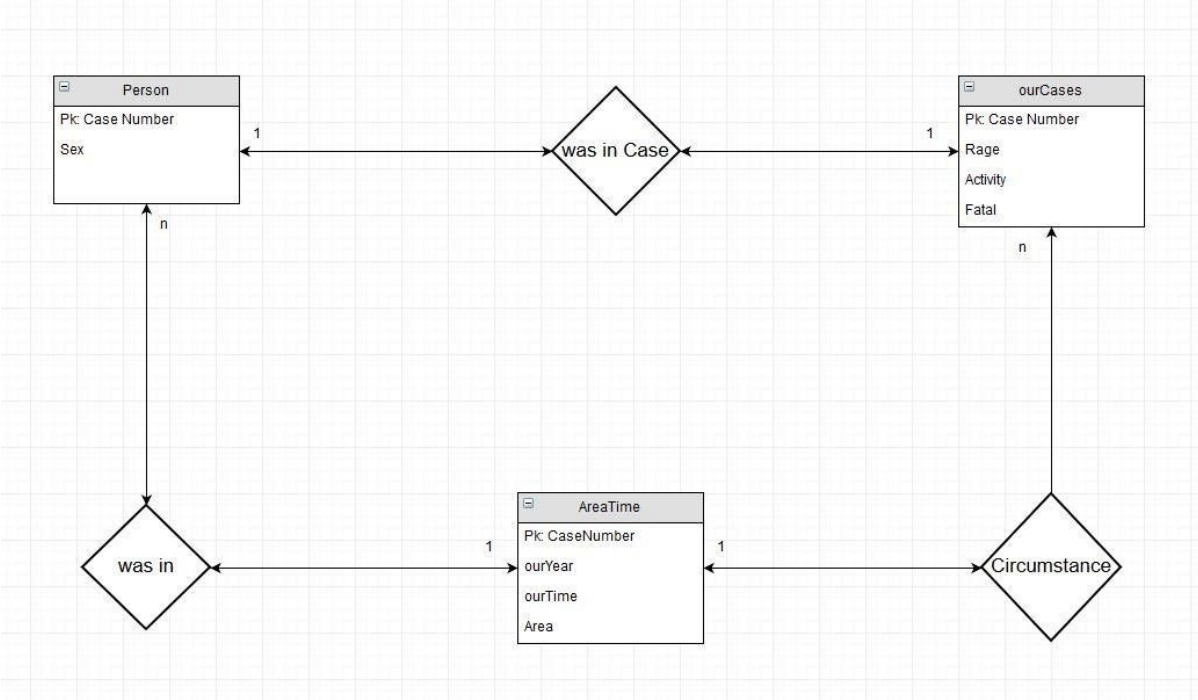
Die Entwicklung unserer Entitäten war bei uns größtenteils trivial. Schon in der Originaltabelle konnte man drei verschiedene Unterpunkte ausmachen: Angaben zur angegriffenen Person, Angaben über das Umfeld und Angaben über die Art des Angriffs.
In unseren ersten Entwürfen war die Zeit noch nicht mit in das Umfeld inbegriffen, uns ist aber aufgefallen, dass es Sinn macht eine Umwelt-entity anstatt eine reine Location-entity zu haben. Diese wurde dann zur Entity AreaTime.
Bei uns waren das einfache und das komplexe konzeptionelle Entity-Relationship-Modell identisch, da wir keine n-zu-m Beziehungen haben.



(Abb. 4: Unser Entity-Relationship-Modell)

Crow's-Foot-Diagram

In unseren Entitäten hatten wir anfangs noch weitere Punkte, wie den Namen und das Alter des Opfers, den genauen Strand des Angriffs oder Daten zum Hai.
Diese Punkte waren aber zu spezifisch oder zu oft nicht bekannt, sodass wir sie gestrichen haben.
Dadurch ist unser Crow's-Foot-Model immer weiter auf die wichtigsten Punkte geschrumpft.



(Abb. 5: Unser Crow's-Foot-Diagram)

5. Normalisierung

Zum Veranschaulichen des hier geschrieben guckt man sich am besten die Grafiken aus den Punkten "Verwendete Daten" & "ER-Modelle" an.

Unsere Daten waren zwar etwas ungepflegt aber recht simpel zu normalisieren. Natürlich waren nicht alle Werte von Anfang an wirklich atomar, w.z.B. hätte man die Namen noch in Vor- und Nachname einteilen können, oder das Datum in Tag, Monat und Jahr auseinander nehmen können. Das war allerdings für uns nicht nötig, da unsere Anwendung diese Struktur gar nicht nutzen würde. Dementsprechend waren die meisten Einträge bereits atomar, alles was unpassend war hatten wir auch schon bei der Säuberung der Daten berücksichtigt.
Wiederholungsgruppen gab es keine, deshalb musste zu dem Aspekt auch nichts getan werden.
Das merkwürdigste und einfachste war es aber für all unsere Tabellen einen geeigneten Primärschlüssel zu finden. Direkt am Anfang ist uns aufgefallen, dass wir einfach die Fallnummer (Case Number) als Primärschlüssel für all unsere Tabellen verwenden können, da diese eine 100% eindeutige Identifikation der jeweiligen Zeilen ist. Da das uns aber etwas zu einfach erschien haben wir uns noch Gedanken darüber gemacht, was wir vielleicht sonst als Primärschlüssel verwenden könnten. Aber auch hier sieht man direkt, dass alle Einträge, bis auf die Fallnummer nicht eindeutig sind und somit nicht als Primärschlüssel geeignet sein können. So haben wir einfach für jede Tabelle die Fallnummer als Primärschlüssel genommen. Das beendet die Bildung der ersten Normalform.

Bei der Bildung der zweiten Normalform gab es auch nicht viel zu tun, da alle Einträge voneinander unabhängig waren und auch nur unseren Standard Primärschlüssel benötigten. An diesem Punkt haben wir dann aber aus der einen großen Entity drei kleine herausgezogen, und zwar die Entities AreaTime, ourCases und Person.

Nachdem das getan wurde lag uns die Dritte Normalform bereits vor, da neben dem Primärschlüssel bei Relationen nicht mehr als ein Nichtschlüsselattribut vorhanden war.

6. Implementierung der Datenbank

Hier am besten unsere SQL Abfragen aus unserem Projektordner nebenbei offen haben.

Die Datenbank wurde mittels der mySQL-Workbench erstellt. Dabei handelt es sich um eine lokale Datenbank. Da die Datenbank eben nur lokal war mussten wir auf jedem Rechner auf dem wir unsere Software nutzen wollten die Datenbank wieder einrichten. Deshalb haben wir uns einfach alle unsere SQL Abfragen als Dateien gespeichert um uns einiges an Arbeit abzunehmen.

Als erstes musste natürlich die Datenbank an sich erstellt werden. Das geschah dann mit der .sql Datei “createMySharkAttacksDB” in dieser wurden lediglich die Tabellen und die dazugehörigen Spalten erstellt. Als nächstes musste man die Tabellen natürlich auch befüllen. Das geschah dann durch den Import der jeweiligen .csv Dateien, in dem Fall “importPerson”, “importCases” und importAreaTime. In diesen Abfragen musste man dann aber immer noch den Pfad anpassen, damit die Daten auch von der Abfrage gefunden werden. Und zu guter letzt haben wir uns noch Abfragen erstellt mit denen wir überprüfen konnten, ob alles richtig erstellt worden ist. Dazu nutzten wird die Abfragen “checkPerson”, “checkCases” und “checkAreaTime”, die uns einfach nur alle Einträge ausgegeben haben und die Abfragen “countPerson”, “countCases” und “countAreaTime” die uns lediglich die Anzahl an Einträgen in der jeweiligen Tabelle wiedergegeben haben.

Nachdem all diese Befehle ausgeführt und die Einträge überprüft worden sind war die Datenbank einsatzbereit.

7. Implementierung des Frontends

Es stand von Anfang an fest, dass wir unser Programm in Java schreiben wollten. Java ist die Programmiersprache, in der wir beide am meisten Erfahrung haben.

Dadurch mussten wir uns außer SQL nicht noch eine zweite Sprache für das Projekt aneignen.
Das Frontend ist mit Java Swing gemacht. Die Hauptkomponente ist ein JFrame, auf den alle anderen Elemente angeordnet werden.
Die verschiedenen, auswählbaren Optionen, die einem beim Öffnen des Programms sofort ins Auge stechen, sind Comboboxen. Diese existieren in Java-Swing so erstmal nicht. Wir wollten aber unbedingt ein Dropdown-Menü, dass anstatt Schrift Bilder anzeigt. Falls einige der Bilder in den Boxen, z. B einige Flaggen von recht kleinen Ländern, dem User nicht bekannt sein sollten, wollten wir auch den Namen anzeigen lassen, wenn man mit der Maus darüber schwebt. Unsere Lösung dafür waren dann die oben genannten Comboboxen.

```
public Component getListCellRendererComponent(
                                    JList list,
                                    Object value,
                                    int index,
                                    boolean isSelected,
                                    boolean cellHasFocus)
{

    //Get the selected index. (The index param isn't
    //always valid, so just use the value.)
    int selectedIndex = ((Integer)value).intValue();

    if (isSelected) {
        setBackground(list.getSelectionBackground());
        setForeground(list.getSelectionForeground());

        if (-1 < index) {
            list.setToolTipText(localCurrentString[index]);
          }

    } else {
        setBackground(list.getBackground());
        setForeground(list.getForeground());
    }

    //Set the icon and text.  If icon was null, say so.
    ImageIcon icon = localCurrentImage[selectedIndex];
    String pet = localCurrentString[selectedIndex];
    setIcon(icon);
    if (icon != null) {
        setText(pet);
        setFont(list.getFont());
    } else {
        setNormalFont(pet + " (no image available)",
                      list.getFont());
    }
```

(Abb. 6: Ausschnitt aus der ComboBoxRenderer Klasse)

Wir haben aus unseren Daten die sechs Optionen: Land/Gebiet, Aktivität, Tödlichkeit, Tageszeit, Provokation und Geschlecht des Opfers ausgewählt.

Für jede dieser Optionen sind die verschiedenen Unterpunkte in einem Array unter dem gleichen Namen wie in der Datenbank gespeichert. Würde die Datenbank also zu einem späteren Zeitpunkt um weitere Unterpunkte, z.B. ein neues Land, erweitert, müsste man diese nur ins Array schreiben und die Comboboxen würden automatisch größer werden.
Der Slider am unteren Ende des Programms ist ein Standard JSlider aus Java-Swing, genau, wie der Standard Button “Check” ein JButton ist.
Beim Bewegen des Sliders wird das Jahr ausgewählt. Wenn man alle Optionen so eingestellt hat, wie man möchte, kann man auf den “Check”-Button drücken, der dann den die Verbindung zur Datenbank auslöst. Die zusammengerechneten Haiangriffe werden dann in der Mitte auf einem JPanel angezeigt.

8. Anbindung der Datenbank

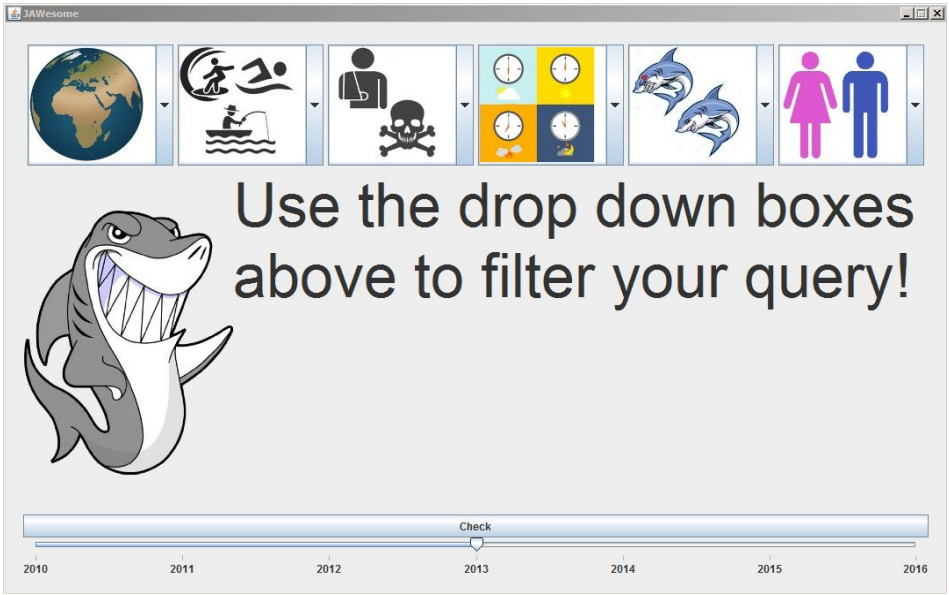
Die Anbindung der Datenbank war einfacher als gedacht. Dadurch, dass wir unser Programm in Java geschrieben haben, was zu Oracle gehört und mySQL benutzt haben, was auch zu Oracle gehört, hat die genannte Firma natürlich eine Schnittstelle zwischen den beiden zur Verfügung gestellt, nämlich JDBC.
JDBC steht für “Java Database Connector” und ist eine Library die eben genau das tut, was der Name impliziert. Durch die Library ist es möglich in seinen Java Programmen Verbindungen zu Datenbanken zu erstellen.

```
//Initialize necessary DB connection components
Connection myConn;
PreparedStatement myStmt;
ResultSet myRs;
//Storage for the result of query
String result;

public Driver() {
    //Try & Catch because of connection type of thing
    try {
        //Connect to Database
        myConn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysharkattacksdb", "root", "test");
    }
    catch (Exception exc) {
        exc.printStackTrace();
    }
}

//Query method
public String filterResult(String Region, String Activity, String Fatal, String Time, String Rage, String Gender, int Year) {
    try {
        //Creating the Query as a PreparedStatement
        myStmt = myConn.prepareStatement(


            "SELECT COUNT(*) "
            + "FROM Area_Time, ourCases, Person "
            + "WHERE "
            + "area_time.Case_Number = ourcases.Case_Number AND "
            + "area_time.Case_Number = person.Case_Number AND "
            + "person.Case_Number = ourcases.Case_Number AND "
            + "ourYear = ? AND "
            + "ourTime = ? AND "
            + "Area = ? AND "
            + "Rage = ? AND "
            + "Activity = ? AND "
            + "Fatal = ? AND "
            + "Sex = ? "
```

(Abb. 7: Ein Ausschnitt der Driver Klasse, die für die Datenbankverbindung und Abfragen zuständig ist.)

Um nun technisch eine Verbindung herzustellen musste man ein Objekt der Klasse Connection erstellen und dieser über die Methode .getConnection(...) die Adresse der Datenbank, einen Benutzernamen und das entsprechende Passwort übergeben. In unserem Fall ist es die standard Lokalhostadresse gefolgt von dem Datenbanknamen. Somit besteht auch schon die Verbindung zwischen Software und Datenbank.
Nun mussten wir für unser Programm in der Klasse noch eine eigene Methode anlegen, die die Usereingabe aus dem Frontende in eine Abfrage überführt und dessen Ergebnis ausgibt.
Um mehr Code dieser Klasse zu sehen einfach die Datei Driver.java angucken.
Das eben erwähnte geschieht durch die Methode .filterResult(...) in der sich unsere Abfrage befindet, aber noch ohne Parameter. Die Parameter wurden zuerst als Fragezeichen dargestellt, diese Fragezeichen dienen als Platzhalter für die Variablen aus der Benutzereingabe. Der folgende Code ist dann nur noch für das ausführen, speichern und wiedergeben der Abfrage zuständig.

9. Das Ergebnis



(Abb. 8: Die Applikation nach dem Start.)

Wenn man das Programm gestartet hat, kann man auf die verschiedenen Dropdownboxen drücken. Man kann ein Land oder im Fall von den USA oder Australien ein Bundesstaat auswählen, in der nächsten Box die Aktivität beim Angriff, ob der Angriff tödlich war oder nicht, die Tageszeit, ob der Hai provoziert war und schließlich das Geschlecht des Opfers. Als letztes wählt man mit dem Slider noch das gewünschte Jahr.
Wenn man seine Filter gesetzt hat drückt man auf den Check-Button und erhält das Ergebnis in Form eines Satzes.

(Abb. 9: Die Applikation nach der Filterung.)

10. Aufgabenverteilung

(Abb. B: Original (http://www.iblogontheside.com/what-is-the-difference-web-designer-vs-web-professional/))

Beide Studenten haben sich um die Implementierung der Datenbank gekümmert.

11. Orientierungshilfe

- JAWesome/Shark-Attack-DB/Misc
  Hier finden Sie zwei Ordner. Einen für unsere SQL-Abfragen und einen für die verwendeten Tabellen.
- JAWesome/Shark-Attack-DB/mySharkAttacks/src
  Hier befindet sich der Sourcecode der Software.