IPA Formatvorlage

autor : Pascal Honegger

version : 0.1

status : Draft

quelle : Atos

dokumentendatum : 31 Oktober 2017

anzahl der seiten : 16

Änderungshistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Beschreibung | Autor |
| 0.1 | 31.10.2017 | Initiale Version erstellt | Pascal Honegger |
| 0.2 | 01.10.2017 | Zeitplan & Informieren abgeschlossen | Pascal Honegger |
| 0.3 | 02.11.2017 | Planen & Entscheiden abgeschlossen | Pascal Honegger |
| 0.4 |  |  | Pascal Honegger |
| 0.5 |  |  | Pascal Honegger |
| 0.6 |  |  | Pascal Honegger |
| 1.0 | xx.yy.2017 | Erster Release | Pascal Honegger |

# Inhaltsverzeichnis

**Teil 1**

1 Aufgabenstellung 4

1.1 Hintergrund 4

1.2 Funktionale Anforderungen 4

1.3 Technische Anforderungen 4

1.4 Erweiterungen 5

1.5 Wahlkriterien 5

2 Projektorganisation 6

2.1 Beteiligte Personen 6

2.2 Projektmanagementmethode 6

2.3 Backup-Konzept 6

3 Ausgangslage 7

3.1 Vorkenntnisse 7

3.2 Vorarbeit 7

3.3 Firmenstandards 7

4 Zeitplanung 8

4.1 Meilensteine 8

4.2 Verhalten im Falle von Verzug 8

4.3 Gantt-Diagramm 9

5 Arbeitsprotokoll 10

5.1 Tag 1 (01.11.2017) 10

5.2 Tag 2 (02.11.2017) 11

5.3 Tag 3 (03.11.2017) 13

5.4 Tag 4 (08.11.2017) 14

5.5 Tag 5 (10.11.2017) 15

**Teil 2**

6 Management Summary 16

6.1 Projektbeschreibung 16

6.2 System-Beschreibung 16

7 Informieren 17

7.1 Ist-Analyse 17

7.2 Soll-Analyse 17

8 Planen 18

8.1 Use-Cases 18

8.2 Testkonzept 24

8.3 Software-Architektur 29

8.4 Datenbankdesign 31

9 Entscheiden 33

9.1 Versionskontrolle 33

9.2 Backend-Framework 33

9.3 Frontend-Framework 33

9.4 Smiley-Umsetzung 33

9.5 Diagramm-Visualisierung 33

10 Realisieren 34

10.1 Backend 34

10.2 Frontend 34

11 Kontrollieren 35

11.1 Zeitmanagement 35

11.2 Funktionalität 35

11.3 Codequalität 35

11.4 Fertigstellung 35

12 Auswerten 36

12.1 Reflexion 36

12.2 Verbesserungen / Erfahrungen 36

12.3 Einsetzbarkeit 36

12.4 Fazit 36

**Anhang**

13 Verzeichnisse 36

13.1 Quellenverzeichnis 36

13.2 Abbildverzeichnis 36

13.3 Tabellenverzeichnis 36

14 Quellcode 37

# Aufgabenstellung

Hier ist die detaillierte Aufgabenstellung abgebildet, welche vom Auftraggeber vorgegeben ist.

## Hintergrund

Der Projektleiter und C#-Teamleiter Christoph Fauti bemüht sich stets um die Zufriedenheit seiner Arbeitskollegen. Deshalb wird das AHM (Atos Happy Meter - Gefühlslage-Messer) als Website realisiert. Die Idee hierbei ist, dass ein Mitarbeiter schnell seine jetzige Gefühlslage festhalten kann und sieht, wie es den Arbeitskollegen geht.

## Funktionale Anforderungen

### Gefühlslage Erfassen

Das Ziel der Website liegt darin, dass ein Benutzer schnell und anonym seine Gefühlslage eintragen kann. Auf der Startseite sieht man eine Auswahl von Smileys (von sehr glücklich bis zu sehr unglücklich). Hier kann man basierend auf seiner jetzigen Gefühlslage einen Smiley drücken. Zusätzlich kann optional ein Kommentar verfasst werden, welcher die eigene Gefühlslage beschreibt. Nach dem Absenden seiner Gefühlslage wird der ausgewählte Smiley und optionale Kommentar zurückgesetzt. Dies erlaubt beispielsweise das Aufsetzen eines dedizierten Laptops im Pausenbereich, auf welchem jeder seine Gefühlslage eintragen kann, ohne die vorherige Auswahl zu sehen. Als ein simpler Spam-Schutz wird das erneute Absenden einer Gefühlslage für eine Minute blockiert.

### Gefühlslage auswerten

Durch herunterscrollen findet man unterhalb der Erfassung eine Übersicht der allgemeinen Gefühlslage. Man sieht die gleichen Smileys, welche man beim Erfassen auswählen kann. Die Grösse der Smileys variiert je nachdem, wie oft dieser an diesem Tag bereits ausgewählt wurde. Diese Daten werden automatisch aktualisiert, ohne dass ein Benutzer die Seite neu laden muss. Zusätzlich zu jedem Smiley wird ein zufälliger Kommentar angezeigt, falls einer für diesen Smiley an diesem Tag existiert.

### Administration

Zusätzlich zu dieser Funktionalität gibt es einen Administrator-Modus. Dieser ist durch ein Login geschützt und für die normalen Benutzer nicht sichtbar. Der Administrator bleibt während der Browser-Session angemeldet und kann sich bei Bedarf abmelden.

Der Administrator kann einen Smiley-Verlauf ansehen. Er kann einen Datumsbereich auswählen und sieht in Form eines Liniendiagramms an welchem Tag welche Smileys wie oft gewählt wurden.

Der Administrator kann die verfügbaren Smileys verwalten. Man kann neue hinzufügen, bestehende bearbeiten und ungebrauchte deaktivieren. Durch das deaktivieren können keine neuen Gefühlslagen mit diesem Smiley erstellt werden, die Auswertungen bleiben trotzdem in der Administration erhalten.

Der Administrator kann seine eigenen Anmeldedaten (Benutzername & Passwort) bearbeiten.

## Technische Anforderungen

* Frontend als Website
  + Unterstützt werden folgende Browser
    - Muss: Google Chrome
    - Muss: Mozilla Firefox
    - Optional: Microsoft Internet Explorer 11
  + Läuft unter Windows 7
* Backend-Server
  + Windows-basierter Server
  + Datenspeicherung in einer relationalen Datenbank
* Admin Login
  + Passwort sicher gespeichert (nicht Klartext)
  + Standard Benutzer ‘admin’ mit dem Passwort ‘admin’

## Erweiterungen

Die Installation auf dem Testsystem ist nicht teil der IPA. Falls die Rückmeldungen nach den ersten Tests positive ausfallen wird das System um eine geografische Gruppierung (Bsp. Stockwerke, Gebäude) erweitert. Diese Funktionalität sollte beim Erstellen der Datenstruktur berücksichtigt werden. Während der Testphase kann dieser Ort für alle Einträge auf einen Test-Wert gesetzt werden.

## Wahlkriterien

* 125 Redundanzfreie Gliederung des Programms
* 130 Vollständiges ERM
* 164 Error-Handling
* 166 Lesbarer Code
* 225 Versionsverwaltung
* 235 Use Cases & Activity Diagram korrektes UML
* 250 Schichtentrennung (MVC Backend | MVVM Frontend)

# Projektorganisation

## Beteiligte Personen

|  |  |  |  |
| --- | --- | --- | --- |
| Position | Name | E-Mail | Telefonnummer |
| Auftraggeber | Christoph Fauti | [christoph.fauto@atos.net](mailto:christoph.fauto@atos.net) | - |
| Fachexperte | Rolf Fux | [rolf.fux@atos.net](mailto:rolf.fux@atos.net) | - |
| Kandidat | Pascal Honegger | [pascal.honegger@siemens.com](mailto:pascal.honegger@siemens.com) | 077 421 51 28 |
| Hauptexperte | Remo Steinmann | [remo.steinmann@siemens.com](mailto:remo.steinmann@siemens.com) | - |
| Zweit-Experte | - | - | - |

Tabelle : Projektteilnehmer

## Projektmanagementmethode

Ich verwende die Projektmanagementmethode IPERKA. Diese Wasserfallmethode hat den Vorteil, dass die Struktur gut mit den Vorgaben der IPA übereinstimmen. Als Alternative wurde Scrum verworfen, da der Overhead mit den formalen Scrum-Meetings zu gross wäre.

## Backup-Konzept

Um die Chance eines Datenverlusts zu minimieren sind regelmässige Backups essenziell. Es werden regelmässig Commits erstellt und online auf einem privaten GitHub-Repository gesichert. Im Falle eines Laptopfehlers kann eine relative aktuelle Version geladen werden. Dies wird sowohl für Word-Dokumente wie Quellcode gemacht. Zusätzlich wird täglich ein Backup auf einen USB-Stick geladen, für den Fall, dass GitHub und der Laptop zur gleichen Zeit nicht mehr erreichbar sind.

# Ausgangslage

Beschreibung Umfeld / Projekt.

## Vorkenntnisse

* Vertieftes C#-Wissen dank zwei Jahren Projekterfahrung
* Verständnis von Angular (2+) dank einem Jahr Projekterfahrung
* Erfahrungen in der Webentwicklung mit HTML, CSS, JavaScript unter Verwendung von Bootstrap und dem Google Material Design
* Erfahrungen mit Java und Python

## Vorarbeit

Als Vorarbeit wurde lediglich eine simple Word-Vorlage für die IPA aufbereitet, welche auch für sonstige Dokumentationen verwendet wird.

## Firmenstandards

### OR-Mapper

Ich setze das Entity Framework 6 von Microsoft ein. Dieser direkt in die DIE integrierte OR-Mapper ist Atos-Firmenstandard für C#-basierte Projekte.

### Frontend

Für das Frontend wird das gleiche Framework (Angular 4.4.6) wie im Produktivprojekt verwendet. Ausserdem werden die selben DIE-Konfigurationen verwendet, damit der korrekte Coding-Style eingehalten wird.

# Zeitplanung

## Meilensteine

|  |  |  |
| --- | --- | --- |
| Meilenstein | Beschreibung | Datum |
| Projektstart | Start des Projektes | 01.11.2017 09:00 |
| Ende Informieren | Informieren Phase abgeschlossen | 01.11.2017 11:00 |
| Ende Planen | Planung abgeschlossen | 02.11.2017 16:00 |
| Ende Entscheiden | Entscheiden Phase abgeschlossen | 02.11.2017 17:00 |
| Ende Realisieren | Realisierung abgeschlossen | 08.11.2017 17:00 |
| Ende Kontrollieren | Kontrolle durch Tests abgeschlossen | 10.11.2017 11:00 |
| Projektabschluss | Erfolgreicher Projektabschluss | 10.11.2017 17:00 |

Tabelle : Meilensteine

## Verhalten im Falle von Verzug

Falls Meilensteine nicht erreicht werden können müssen Aktionen getroffen werden. Primär werden nicht kritische Teilaufgaben vernachlässigt. Die Teilaufgaben, welche vernachlässigt werden sind das Administrator-Design, Emojis und Ort-Erweiterungen in dieser Reihenfolge.

## Gantt-Diagramm



Abbildung : Zeitplan

# Arbeitsprotokoll

## Tag 1 (01.11.2017)

### Planung

Heute möchte ich mit dem Projekt beginnen und die erste IPERKA-Phase abschliessen. Zusätzlich wird mit der Planung begonnen.

|  |  |  |  |
| --- | --- | --- | --- |
| Was | Soll-Dauer | Ist-Dauer | Fertig |
| Projektbeginn | 30min | 15min | ✔ |
| Zeitplanung | 2h | 2h | ✔ |
| Projektorganisation dokumentieren | 30min | 5min | ✔ |
| Ausgangslage dokumentieren | 30min | 10min | ✔ |
| Ist- und Soll-Analyse | 30min | 20min | ✔ |
| Use-Cases | 2h | 2h 30min | ✔ |
| Software-Architektur | 30min | 10min | ✘ |
| Arbeitsprotokoll | 45min | 20min | ✔ |

Tabelle : Konkrete Planung Tag 1

### Getane Arbeit

Heute war der erste Tag des Projektes. Nach dem Ankommen begann die Arbeit flott und ohne Probleme. Beim Erstellen des Zeitplans habe ich festgestellt, dass die total fünf Tage à sechs Stunden knapp bemessen sind. Der Grossteil der Dokumentation muss morgen fertig gestellt werden, damit ich am Freitag und nächsten Mittwoch die Implementation abschliessen kann. Danach habe ich lediglich einen Tag für das Testen und Finalisieren des Projekts, solange nichts schiefläuft. Im Falle unerwarteter Probleme habe ich keinen wirklichen Platz für einen Puffer und werden wohl an anderen Funktionen kürzen müssen, damit das Projekt abgeschlossen wird. Also hoffe ich, dass es dazu nicht kommen wird.

### Probleme

Nach der ersten Schnell-Analyse des Auftrages bin ich davon ausgegangen, dass ich lediglich ~5 Use-Cases haben werden. Nach der Umsetzung hat sich jedoch gezeigt, dass ich ganze 11 Use-Cases zu beschreiben habe. Deshalb habe ich auch etwas länger als die initial geplanten 2 Stunden benötigt, um diese auszuformulieren.

### Reflexion

Meine Stimmung wurde direkt zu Beginn zerschmettert. Beim Erstellen des Zeitplans wurde mir bewusst, wie viel Arbeit noch vor mir liegt. Zum Glück ist diese Angst mit jeder Stunde geschrumpft, da ich gute Fortschritte machte. Allgemein bin ich zufrieden mit diesem Projektstart und habe zum ersten Mal in einem Projekt nicht das Gefühl, die Planung zu vernachlässigen.

## Tag 2 (02.11.2017)

### Planung

Heute möchte ich die zwei IPERKA-Phasen Planen und Entscheiden abschliessen.

|  |  |  |  |
| --- | --- | --- | --- |
| Was | Soll-Dauer | Ist-Dauer | Fertig |
| Software-Architektur – Komponentendiagramm | 30min | 40min | ✔ |
| Software-Architektur – Aktivitätsdiagramm | 30min | 40min | ✔ |
| Software-Architektur – Beschreibung | 15min | 20min | ✔ |
| ERM – Schema in Workbench erstellen | 45min | 1h | ✔ |
| ERM – Sinnvolle Feld-Typen definieren und dokumentieren | 45min | 40min | ✔ |
| Testfälle erstellen | 2h 30min | 2h 40min | ✔ |
| Entscheidungen dokumentieren | 30min | 30min | ✔ |
| Expertenbesuch | - | 15min | ✔ |
| Emoji-Recherche | 0min | 10min | **~** |
| Arbeitsprotokoll | 30min | 30min | ✔ |

Tabelle : Konkrete Planung Tag 2

### Getane Arbeit

Heute habe ich die beiden IPERKA-Phasen Planen und Entscheiden abgeschlossen. Als Teil der Planung habe ich zwei UML-Diagramme mit PlantUML erstellt. Danach habe ich das ERM realisiert und dokumentiert und die Testfälle beschrieben. Der Zeitaufwand beim ERM und den Testfällen war grösser als erwartet. Da wir heute anstatt der geplanten sechs Stunden mehr Zeit zum Arbeitete hatten, bin ich immer noch im Zeitplan.

### Probleme

Wie erwartet gab es Probleme mit den Smileys. Diese wollte ich als Charakter in der Datenbank speichern, wobei hier die Kollation geändert werden muss. Nach weiteren Fehlern habe ich versucht via C# Emojis in die Datenbank zu speichern, in der Hoffnung, dass der bisherige Fehler in der MySQL-Workbench liegt. Dabei stiess ich auf einen weiteren Fehler, welcher nach Recherche auf einen MySQL-Fehler zurückzuführen ist. Als Lösung konnte ich schlussendlich zwei Befehle ausführen, welche ich in folgendem Stack Overflow gefunden haben: <https://stackoverflow.com/questions/39071021/vs-error-the-value-for-column-isprimarykey-in-table-tabledetails-is-dbnull>. Leider hat das Speichern von Emojis wie C# auch nicht funktioniert. Schlussendlich habe ich mich dazu entschieden einen Workaround zu benutzen und den HEX-Code der Emoji-Zeichen in der Datenbank zu speichern. Diese sind nach Unicode bis zu sechs Zeichen lange, weshalb ich das Datenbankfeld 6 Zeichen langgemacht habe. Die Limitierung dieser Lösung ist jedoch, dass gewisse «komplexe» Emojis nicht verwendbar sind. Komplexe Emojis sind hier zusammengebaute Emojis. Beispielsweise kann ich einen gelben (farbneutralen) Emoji mit einer Hautfarbe kombinieren. Diese komplexen Fälle habe ich aus Zeitgründen ignoriert, da diese Spezialfälle nicht verlangt sind.

Im Verlaufe vom Tag sind immer mehr Use-Cases und Testfälle aufgefallen, welche ich nicht bedacht habe. So ist die ohnehin grosse Zahl an Use-Cases noch grösser geworden.

Wir mussten unseren Zeitplan als PDF abgeben, was mir Schwierigkeiten bereitet hat. So habe ich es nicht geschafft den gesamten Zeitplan auf eine Seite zu laden. Meine Abgabe beinhaltet nun leider ein PDF mit zwei Seiten, welches nicht sehr anschaulich ist.

### Reflexion

Heute habe ich viel Zeit mit vergolden verschwendet, was ich auf jeden Fall bereue. Ich habe naiv versucht die Emojis als einen Charakter in der Datenbank abzuspeichern. Dazu kamen zuerst Probleme mit der Kollation, gefolgt von zusammengesetzten Emojis und der Zeichenlänge, welche variieren kann. Ich habe erst verschwenden von Zeit und Nerven meine Idee an den Nagel gehängt und einfach die Hexadezimale Darstellung eines Emojis in die Datenbank gespeichert. Zusammengesetzte / Komplexe Emojis funktionieren so zwar nicht, sind aber auch keine Anforderung. Im Allgemeinen hätte ich vor dem Beginn der Recherche meiner ersten Idee weitere Ansätze studieren müssen. Stattdessen bin ich davon ausgegangen, dass meine Lösung die korrekte ist, und habe unnötig Zeit verloren.

Noch gestern hatte ich das Gefühl, nichts vergessen zu haben und genügend Zeit in die Planung zu investieren. Heute ist mir aufgefallen, dass ich, trotz der vielen Überlegungen während der Planung, Anwendungsfälle und Testfälle vergessen habe. Ich hoffe, dass keine weiteren Überraschungen in der Implementation auftauchen. Hätten wir heute nur die geplanten sechs Stunden arbeiten können, wäre ich bereits hinter dem Zeitplan. Falls ich also bei der Implementation auf weitere Probleme stosse, wird der erfolgreiche Abschluss des Projekts gefährdet.

## Tag 3 (03.11.2017)

### Planung

Heute möchte ich mit der Realisierung beginnen, die Projekte aufsetzen und das Backend fertigstellen.

|  |  |  |  |
| --- | --- | --- | --- |
| Was | Soll-Dauer | Ist-Dauer | Fertig |
| Backend-Projekt aufsetzen | 15min | 15min |  |
| Frontend-Projekt aufsetzen | 15minc | 15min |  |
| Datenbankanbindung | 45min | 1h 15min |  |
| Backend Schnittstellen | 1h 30min | 2h |  |
| Backend Autorisierung | 30min | 30min |  |
| Frontend Serveranbindung | 45min | 1h 15min |  |
| Frontend Emoji-Handling | 45min |  |  |
| Arbeitsprotokoll | 45min |  |  |
| MySQL-Probleme beheben | 0min | 1h |  |

Tabelle : Konkrete Planung Tag 3

### Probleme

Heute begann endlich die Implementierungs- / Realisierungs-Phase. Direkt zu streikte jedoch Visual Studio 2017, meine C#-IDE. Hierbei stürzte die Benutzeroberfläche des OR-Mappers ohne Kommentar ab, sobald man Code generieren wollte. Aus diesem Grund musste ich den generierten Code von Hand schreiben, was mich unnötig Zeit gekostet hat. Zusätzlich tauchten noch MySQL-Verbindungsfehler auf. Nach Internet-Recherche habe ich herausgefunden, dass dies auf die aktuelle Version von .Net/MySQL-Connector zurückzuführen ist. Nach der Installation einer älteren Version gab es bei der Interaktion zwischen C# und MySQL keine weiteren Probleme.

### Reflexion

Nach den initialen Startschwierigkeiten ging die Arbeit schnell voran. Bei der Implementation des Servers hatte ich das Gefühl, ich würde einen menschlichen Generator nachahmen. Auch beim Schreiben der Frontend-Backend-Kommunikation mussten viele DTOs einmal in C# und einmal in TypeScript definieren. Der sichtbare Fortschritt meiner Implementation ist leider noch nicht sichtbar, was frustrierend ist. Dafür sollten jetzt auf Server-Seite alle Spezialitäten wie Autorisierung und Validierung abgeschlossen sein. Auch im Frontend sind alle Daten bereit geladen zu werden, lediglich die Benutzerdarstellung fehlt. Ich hoffe stark, dass die Umsetzung der Darstellung schnell von Statten geht, da ich sonst garantiert Funktionalität streichen muss.

## Tag 4 (08.11.2017)

### Planung

## Tag 5 (10.11.2017)

### Planung

# Management Summary

In diesem Kapitel folgt eine Kurzzusammenfassung über das Projekt.

## Projektbeschreibung

Teamleiter C# Christoph Fauti möchte Gefühlslage der Mitarbeiter besser verstehen.

Einfache und anonyme Erfassung seiner Gefühlslage anhand von Smileys

## System-Beschreibung

# Informieren

## Ist-Analyse

Eine Analyse des (noch nicht) vorhandenen Systems.

Momentan bietet die Atos kein direktes Tool an, mit welchem die Mitarbeiter ihre Gefühlslage veröffentlichen können. Möchte man eine Beschwerde oder einen Kommentar zu etwas abgeben, muss man eine betroffene Person direkt kontaktieren. Gefällt einem jedoch das warme Wetter nicht, gibt es keine direkte Ansprechperson. Auch fällt es einem Mitarbeiter schwer Kritik an seinem Chef oder Projekt auszuüben, da dies nicht anonym ist.

## Soll-Analyse

Eine genauere Analyse der Anforderungen und des Soll-Zustands

### Frontend

Das Web-Frontend wird mit Angular (2+) umgesetzt und funktioniert mindestens im Firefox und Chome unter Windows 7. Auf der Startseite sieht man eine Auswahl der verschiedenen Emotionen in Form von Smileys. Je öfter ein solcher Smiley an diesem Tag bereits gedrückt wurde, desto Grösser erscheint dieser. Zufälligerweise werden zu den Smileys Kommentare angezeigt, falls es welche der letzten 24 Stunden gibt. Der Benutzer, normalerweise ein Atos-Angestellter, wählt einen zu seiner Gefühlslage passenden Smiley aus und kann diese Emotion absenden. Optional gibt es unter den Smileys ein Textfeld, in welchem ein bis zu 250 Zeichen langer Kommentar verfasst werden kann. Nach dem Absenden wird die Seite direkt aktualisiert.

Zusätzlich kann der Administrator über die URL auf eine Administrator-Ansicht wechseln. Diese ist mit einem Passwort geschützt und erlaubt das verwalten der verfügbaren Smileys (CRUD-Operationen) und einer grafischen Historie der Gefühlslagen.

### Backend

Das Backend ist in C# geschriebener REST-Service. Dieser Service leitet grösstenteils gefilterte Daten aus der Datenbank an das Frontend. Zusätzlich gibt es die Authentifizierung, welche sicherstellt, dass nur Administratoren auf den gesicherten Bereich zugreifen können.

### Datenbank

Die Datenbank ist in MySQL erstellt. Der Zugriff auf diese Erfolgt über das Microsoft Entity Framework 6, der offizielle OR-Mapper von Microsoft. Alle Aufrufe an die Datenbank über diesen OR-Mapper sind transaktionell gesichert.

# Planen

## Use-Cases

### Use-Case Diagramm

Alle Use-Cases des Atos-Happy-Meters. Der Administrator teilt alle Use-Cases mit einem normalen Benutzer und erweitert diesen. Die Use-Cases sind in die drei Hauptteile der Applikation «Administration», «Anmeldung» und «Startseite» getrennt.

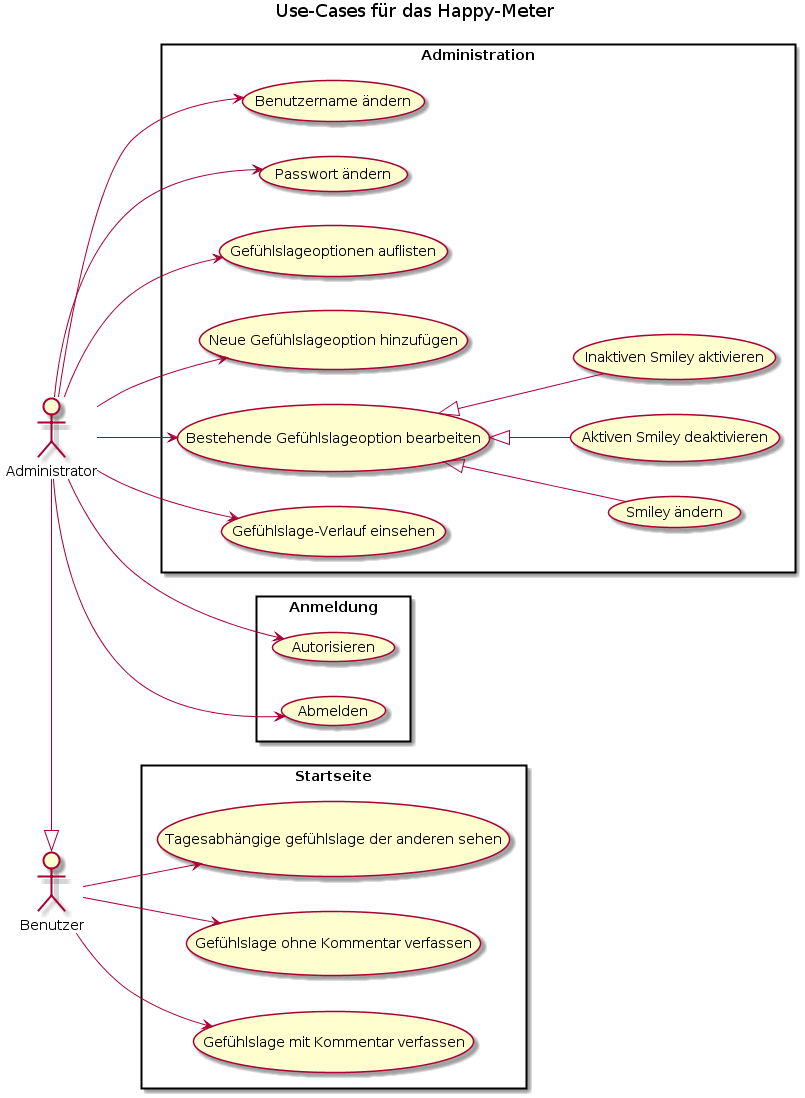


Abbildung : Use-Case Diagramm

### Use-Cases 1: Startseite

Alle Use-Cases in diesem Kapitel werden von einem Benutzer oder autorisierten Administrator durchgeführt.

|  |  |
| --- | --- |
| Use-Case 1.1: Tagesabhängige Gefühlslage sehen | |
| Beschreibung | Ein Benutzer kann die allgemeine Gefühlslage sehen. Dadurch kann er einschätze, ob die Stimmung positiv oder negativ ist. Zusätzlich werden die Kommentare anderer Benutzer dargestellt. |
| Ergebnis / Nachbedingung | Die allgemeine Gefühlslage für diesen Tag wird dargestellt |
| Vorbedingung | Mehrere Gefühlslagen sind erfasst, darunter mindestens eine mit einem Kommentar |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Startseite öffnen | |

Tabelle : Use-Case 1.1

|  |  |
| --- | --- |
| Use-Case 1.2: Gefühlslage ohne Kommentar erfassen | |
| Beschreibung | Ein Benutzer kann seine jetzige Gefühlslage durch das Auswählen eines Smileys erfassen |
| Ergebnis / Nachbedingung | Die eingetragene Gefühlslage verändert die allgemeine Gefühlslage |
| Vorbedingung | Keine Gefühlslagen sind erfasst, mindestens zwei Gefühlslageoptionen existieren |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Startseite öffnen | | 2 | Einen Smiley auswählen | | 3 | Auf Senden drücken | | 4 | Der ausgewählte Smiley wird in der Übersicht grösser dargestellt | |

Tabelle : Use-Case 1.2

|  |  |
| --- | --- |
| Use-Case 1.3: Gefühlslage mit Kommentar erfassen | |
| Beschreibung | Ein Benutzer kann seine jetzige Gefühlslage durch das Auswählen eines Smileys erfassen und zusätzlich einen beschreibenden Kommentar verfassen |
| Ergebnis / Nachbedingung | Die eingetragene Gefühlslage verändert die allgemeine Gefühlslage und der eingetragene Kommentar wird angezeigt |
| Vorbedingung | Keine Gefühlslagen sind erfasst, mindestens zwei Gefühlslageoptionen existieren |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Startseite öffnen | | 2 | Einen Smiley auswählen | | 3 | In das Kommentar-Feld «Mir gefallen diese Smileys» eingeben | | 4 | Auf Senden drücken | | 5 | Der ausgewählte Smiley wird in der Übersicht grösser dargestellt | | 6 | Zusätzlich zum ausgewählten Smiley wird der eingegebene Kommentar «Mir gefallen diese Smileys» angezeigt | |

Tabelle : Use-Case 1.3

### Use-Cases 2: Anmeldung

Alle Use-Cases in diesem Kapitel werden von einem noch nicht autorisierten Administrator durchgeführt.

|  |  |
| --- | --- |
| Use-Case 2.1: Anmelden (Direkt) | |
| Beschreibung | Ein Administrator muss sich zuerst als solcher autorisieren, bevor er auf die Administration zugreifen darf. |
| Trigger | Ein noch nicht autorisierter Administrator möchte auf die Administration zugreifen |
| Ergebnis / Nachbedingung | Der Administrator wird auf die Administration weitergeleitet |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Login-Seite öffnen | | 2 | Benutzername «admin» und Passwort «admin» eingeben | | 3 | Knopf «Anmelden» drücken | | 4 | Weiterleitung auf Administrations-Seite | |

Tabelle : Use-Case 2.1

|  |  |
| --- | --- |
| Use-Case 2.2: Anmelden (Indirekt) | |
| Beschreibung | Ein Administrator muss sich zuerst als solcher autorisieren, bevor er auf die Administration zugreifen darf. |
| Trigger | Ein noch nicht autorisierter Administrator möchte auf die Administration zugreifen |
| Ergebnis / Nachbedingung | Der Administrator wird auf die Administration weitergeleitet |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | | 2 | Weiterleitung auf Login-Seite | | 3 | Benutzername «admin» und Passwort «admin» eingeben | | 4 | Knopf Anmelden drücken | | 5 | Weiterleitung auf Administrations-Seite | |

Tabelle : Use-Case 2.2

|  |  |
| --- | --- |
| Use-Case 2.3: Abmelden | |
| Beschreibung | Ein Administrator kann sich nach dem Anmelden wieder abmelden |
| Trigger | Ein autorisierter Administrator möchte seine Sitzung beenden |
| Ergebnis / Nachbedingung | Der Administrator wird auf die Startseite weitergeleitet |
| Vorbedingung | Use-Case 2.1 oder 2.2 |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Auf den Knopf «Abmelden» drücken | |

Tabelle : Use-Case 2.3

### Use-Cases 3: Administration

Alle Use-Cases in diesem Kapitel werden von einem autorisierten Administrator durchgeführt.

|  |  |
| --- | --- |
| Use-Case 3.1: Gefühlslageoptionen auflisten | |
| Beschreibung | Ein Administrator kann alle (auch deaktivierte) Gefühlslageoptionen einsehen |
| Ergebnis / Nachbedingung | Der Administrator sieht eine Auflistung aller Gefühlslageoptionen |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | |

Tabelle : Use-Case 3.1

|  |  |
| --- | --- |
| Use-Case 3.2: Neue Gefühlslageoption hinzufügen | |
| Beschreibung | Ein Administrator kann eine neue Gefühlslageoption hinzufügen |
| Ergebnis / Nachbedingung | Die hinzugefügte Gefühlslageoption ist auf der Startseite und in der Administration sichtbar |
| Vorbedingung | Eine Gefühlslage mit dem Smiley «🎃» existiert noch nicht |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | | 2 | Auf «Neue Gefühlslageoption hinzufügen» drücken | | 3 | Im Dialog den Smiley «🎃» eingeben | | 4 | Dialog bestätigen | |

Tabelle : Use-Case 3.2

|  |  |
| --- | --- |
| Use-Case 3.3: Deaktivierte Gefühlslageoption aktivieren | |
| Beschreibung | Eine deaktivierte Gefühlslageoption wird den Benutzern nicht angezeigt. Der Administrator kann diese Gefühlslage wieder verfügbar machen, indem er diese aktiviert |
| Ergebnis / Nachbedingung | Die neu deaktivierte Gefühlslageoption ist auf der Startseite nicht mehr sichtbar |
| Vorbedingung | Die Gefühlslage mit dem Smiley «🎃» existiert und ist aktiviert |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | | 2 | In der Übersicht die Gefühlslageoption «🎃» suchen | | 3 | Bei dieser Gefühlslageoption den Knopf «Aktivieren» drücken | |

Tabelle : Use-Case 3.3

|  |  |
| --- | --- |
| Use-Case 3.4: Aktivierte Gefühlslageoption deaktivieren | |
| Beschreibung | Eine aktivierte Gefühlslageoption wird den Benutzern angezeigt. Der Administrator kann diese Gefühlslage nicht mehr verfügbar machen, indem er diese deaktiviert |
| Ergebnis / Nachbedingung | Die neu aktivierte Gefühlslageoption ist auf der Startseite sichtbar |
| Vorbedingung | Die Gefühlslage mit dem Smiley «🎃» existiert und ist deaktiviert |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | | 2 | In der Übersicht die Gefühlslageoption «🎃» suchen | | 3 | Bei dieser Gefühlslageoption den Knopf «Deaktivieren» drücken | |

Tabelle : Use-Case 3.4

|  |  |
| --- | --- |
| Use-Case 3.5: Gefühlslage-Verlauf einsehen | |
| Beschreibung | Als Administrator kann ich die Gefühlslage zwischen zwei Daten sehen |
| Ergebnis / Nachbedingung | In Form eines Liniendiagramms wird die tägliche Gefühlslage dargestellt. |
| Vorbedingung | Die Gefühlslage mit dem Smiley «🎃» existiert und hat Gefühlslagen für den jetzigen und vorherigen Tag |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | |

Tabelle : Use-Case 3.5

|  |  |
| --- | --- |
| Use-Case 3.6: Benutzername ändern | |
| Beschreibung | Als Administrator kann man seinen Benutzernamen ändern |
| Ergebnis / Nachbedingung | Anmeldung mit dem neuen Benutzernamen ist möglich |
| Vorbedingung | Benutzer «admin2» existiert nicht |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | | 2 | Benutzername ändern drücken | | 3 | Im Dialog den neuen Benutzernamen «admin» eingeben | | 4 | Benutzername ändern anwählen | |

Tabelle : Use-Case 3.6

|  |  |
| --- | --- |
| Use-Case 3.7: Passwort ändern | |
| Beschreibung | Als Administrator kann man sein Passwort ändern |
| Ergebnis / Nachbedingung | Anmeldung mit dem neuen Passwort ist möglich |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | | 2 | Passwort ändern drücken | | 3 | Im Dialog für die Felder Passwort und Passwort wiederholen das gleiche neue Passwort eingeben | | 4 | Passwort ändern anwählen | |

Tabelle : Use-Case 3.7

|  |  |
| --- | --- |
| Use-Case 3.8: Gefühlslage-Smiley ändern | |
| Beschreibung | Als Administrator kann man im Nachhinein den Smiley einer Gefühlslage ändern |
| Ergebnis / Nachbedingung | Der neue Smiley wird für die Gefühlslage angezeigt |
| Vorbedingung | Die Gefühlslage mit dem Smiley «🎃» existiert, die Gefühlslage mit dem Smiley «🎅» existiert nicht |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Administrations-Seite öffnen | | 2 | In der Übersicht die Gefühlslageoption «🎃» suchen | | 3 | Bei dieser Gefühlslageoption den Knopf «Smiley ändern» drücken | | 4 | Im Dialog den neuen Smiley «🎅» eingeben | | 5 | Dialog bestätigen | |

Tabelle : Use-Case 3.8

## Testkonzept

Das Testkonzept beschreibt wie nach der Realisierung sichergestellt wird, dass die Funktionalität funktioniert. Dazu werden Testfälle definiert und später in der Kontrollieren-Phase durchgeführt.

### Eingesetzte Testmittel und -Methoden

Alle Tests werden, ausser explizit angegeben, mit folgender Hardware & Software durchgeführt:

* Dell-Laptop mit 16Gb Arbeitsspeicher, einem Intel i7 Prozessor und fungierender Internetverbindung
* Google Chrome 62.0.3202.75
* Windows 7 (Systemsprache: Deutsch)

### Testfälle 1: Startseite

Diese Tests können von Benutzern und Administratoren durchgeführt werden.

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Vorbedingung | Testschritte | Erwartetes Ergebnis |
| 1.1 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃  🎅 x 2 Gefühlslagen  🎃 x 1 Gefühlslagen | |  |  | | --- | --- | | 1 | Startseite öffnen | | Die Gefühlslage 🎅 ist doppelt so gross wie 🎃 |
| 1.2 | Es existieren keine aktiven Gefühlslageoptionen | |  |  | | --- | --- | | 1 | Startseite öffnen | | Eine leere Seite wird angezeigt |
| 1.3 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃 und die deaktivierte Gefühlslageoption 🍄 | |  |  | | --- | --- | | 1 | Startseite öffnen | | 🍄 wird nicht angezeigt |
| 1.4 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃  🎅 x 1 Gefühlslagen  🎃 x 1 Gefühlslagen | |  |  | | --- | --- | | 1 | Startseite öffnen | | 2 | Die Gefühlslageoption 🎅 auswählen | | 3 | Senden drücken | | Die Gefühlslage 🎅 ist doppelt so gross wie 🎃 |
| 1.5 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃  🎅 x 1 Gefühlslagen  🎃 x 1 Gefühlslagen | |  |  | | --- | --- | | 1 | Startseite öffnen | | 2 | Die Gefühlslageoption 🎅 auswählen | | 3 | Als Kommentar «Weihnachten ist cool» eingeben | | 4 | Senden drücken | | Die Gefühlslage 🎅 ist doppelt so gross wie 🎃  Bei der Gefühlslage 🎅 wird der Kommentar «Weihnachten ist cool» angezeigt |
| 1.6 | Es existieren die aktive Gefühlslageoptionen 🎅 | |  |  | | --- | --- | | 1 | Startseite öffnen | | 2 | Die Gefühlslageoption 🎅 auswählen | | 3 | Als Kommentar mindestens 251-mal das Zeichen «a» eingeben | | 4 | Senden drücken | | Beim Senden wird ein Fehler angezeigt, dass der Kommentar zu lang ist |
| 1.7 | Es existieren die aktive Gefühlslageoptionen 🎅 | |  |  | | --- | --- | | 1 | Startseite öffnen | | 2 | Die Gefühlslageoption 🎅 auswählen | | 3 | Als Kommentar mindestens 251-mal das Zeichen «a» eingeben | | 4 | Senden drücken | | Beim Senden wird ein Fehler angezeigt, dass der Kommentar zu lang ist |

Tabelle : Testfälle 1 - Startseite

### Testfälle 2: Anmeldung

Diese Tests können ausschliesslich von noch nicht autorisierten Benutzern durchgeführt werden.

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Vorbedingung | Testschritte | Erwartetes Ergebnis |
| 2.1 | Der Administrator «admin» mit dem Passwort «admin» existiert | |  |  | | --- | --- | | 1 | Login öffnen | | 2 | Benutzername «admin» eingeben | | 3 | Passwort «admin» eingeben | | 4 | Anmelden drücken | | Man wird auf die Administration weitergeleitet |
| 2.2 | Der Administrator «admin» mit dem Passwort «admin» existiert | |  |  | | --- | --- | | 1 | Login öffnen | | 2 | Benutzername «admin» eingeben | | 3 | Passwort «admin» eingeben | | 4 | Anmelden drücken | | Man wird auf die Administration weitergeleitet |
| 2.3 | Der Administrator «qwer» existiert nicht | |  |  | | --- | --- | | 1 | Login öffnen | | 2 | Benutzername «qwer» eingeben | | 3 | Passwort «123» eingeben | | 4 | Anmelden drücken | | Ein Fehler wird angezeigt, dass der Benutzername oder das Passwort falsch ist |
| 2.4 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | Man wird auf die Login-Seite weitergeleitet |
| 2.5 | Der Administrator «admin» mit dem Passwort «admin» existiert | |  |  | | --- | --- | | 1 | Login öffnen | | 2 | Benutzername «admin» eingeben | | 3 | Passwort «admin» eingeben | | 4 | Anmelden drücken | | 5 | Knopf Abmelden drücken | | Man wird auf die Startseite weitergeleitet |

Tabelle : Testfälle 2 - Anmeldung

### Testfälle 3: Administration

Diese Tests können ausschliesslich von autorisierten Administratoren durchgeführt werden

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Vorbedingung | Testschritte | Erwartetes Ergebnis |
| 3.1 | Der Benutzer «admin2» existiert nicht | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Benutzername ändern klicken | | 3 | Als neuen Benutzername «admin2» eingeben | | 4 | Dialog bestätigen | | Login mit dem Benutzernamen «admin2» möglich |
| 3.2 | Der Benutzer «admin2» existiert | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Benutzername ändern klicken | | 3 | Als neuen Benutzername «admin2» eingeben | | 4 | Dialog bestätigen | | Ein Fehler wird angezeigt, der neue Benutzername existiert bereits |
| 3.3 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Benutzername ändern klicken | | 3 | Als neuen Benutzername 101-mal das Zeichen «a» eingeben | | 4 | Dialog bestätigen | | Ein Fehler wird angezeigt, der neue Benutzername ist zu lange |
| 3.4 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Benutzername ändern klicken | | 3 | Dialog bestätigen | | Ein Fehler wird angezeigt, der neue Benutzername kann nicht leer sein |
| 3.5 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Passwort ändern auswählen | | 3 | Als neues Passwort «123» eingeben | | 4 | Als Passwort wiederholen «123» eingeben | | 5 | Dialog bestätigen | | Login mit dem Passwort «123» möglich |
| 3.6 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Passwort ändern auswählen | | 3 | Als neues Passwort «123» eingeben | | 4 | Als Passwort wiederholen «qwer» eingeben | | 5 | Dialog bestätigen | | Fehler wird angezeigt, Passwörter stimmen nicht überein |
| 3.7 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Passwort ändern auswählen | | 3 | Dialog bestätigen | | Fehler wird angezeigt, Passwort kann nicht leer sein |
| 3.8 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃 und die deaktivierte Gefühlslageoption 🍄 | |  |  | | --- | --- | | 1 | Administration öffnen | | Die Gefühlslageoptionen 🎅, 🎃 und 🍄 werden aufgelistet. Die ersten beiden werden aktiviert angezeigt, die letzte deaktiviert |
| 3.9 | Es existieren keine Gefühlslageoptionen | |  |  | | --- | --- | | 1 | Administration öffnen | | Eine leere Liste wird angezeigt |
| 3.10 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃 | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Gefühlslageoption hinzufügen drücken | | 3 | Im Dialog 🍄 eingeben | | 4 | Dialog bestätigen | | Die Gefühlslageoptionen 🎅, 🎃 und 🍄 werden aufgelistet. Alle werden aktiviert angezeigt. Auf der Startseite werden alle drei angezeigt |
| 3.11 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Gefühlslageoption hinzufügen drücken | | 3 | Dialog bestätigen | | Ein Fehler wird angezeigt, der Smiley kann nicht leer sein |
| 3.12 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Gefühlslageoption hinzufügen drücken | | 3 | Als Smiley «qwer» eingeben | | 4 | Dialog bestätigen | | Ein Fehler wird angezeigt, der Smiley ist nicht valide |
| 3.13 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Gefühlslageoption hinzufügen drücken | | 3 | Als Smiley «🥄» eingeben | | 4 | Dialog bestätigen | | Ein Fehler wird angezeigt, der Smiley ist nicht valide |
| 3.14 | Es existiert die Gefühlslageoption 🎅 | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Gefühlslageoption hinzufügen drücken | | 3 | Als Smiley 🎅 eingeben | | 4 | Dialog bestätigen | | Ein Fehler wird angezeigt, der Smiley existiert bereits |
| 3.15 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃 | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Die Gefühlslage 🎅 suchen | | 3 | Smiley ändern anwählen | | 4 | Im Dialog den Smiley 🎃 eingeben | | 5 | Dialog bestätigen | | Ein Fehler wird angezeigt, der Smiley existiert bereits |
| 3.16 | Es existiert die Gefühlslageoption 🎅 | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Die Gefühlslage 🎅 suchen | | 3 | Smiley ändern anwählen | | 4 | Im Dialog den Smiley 🎃 eingeben | | 5 | Dialog bestätigen | | Die Gefühlslage besitzt nun einen neuen Smiley |
| 3.17 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃 und die deaktivierte Gefühlslageoption 🍄 | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Die Gefühlslage 🎅 suchen | | 3 | Gefühlslage deaktivieren drücken | | Die Gefühlslageoption 🎃 wird aktiv angezeigt, die anderen zwei 🎅 und 🍄 werden deaktiviert angezeigt. Auf der Startseite sieht man nur 🎃 |
| 3.18 | Es existieren die zwei aktiven Gefühlslageoptionen 🎅 & 🎃 und die deaktivierte Gefühlslageoption 🍄 | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Die Gefühlslage 🍄 suchen | | 3 | Gefühlslage aktivieren drücken | | Die Gefühlslageoptionen 🎅, 🎃 und 🍄 werden aufgelistet. Alle werden aktiviert angezeigt. Auf der Startseite werden alle drei angezeigt |
| 3.19 | Die Gefühlslagen 🎃 & 🎅 existieren und haben Gefühlslagen für den jetzigen und vorherigen Tag | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Als Von-Datum das Datum des vorherigen Tages eintragen | | 3 | Als Bis-Datum das jetzige Datum eintragen | | Ein Liniendiagramm mit zwei unterschiedlichen Linien für die beiden Smileys wird angezeigt. Auf der X-Achse sind die zwei Tage zu sehen, auf der Y-Achse die Anzahl Gefühlslagen |
| 3.20 | Es sind keine Gefühlslagen vorhanden | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Als Von-Datum das Datum des vorherigen Tages eintragen | | 3 | Als Bis-Datum das jetzige Datum eintragen | | Leeres Liniendiagramm wird dargestellt |
| 3.21 |  | |  |  | | --- | --- | | 1 | Administration öffnen | | 2 | Als Von-Datum das jetzige Datum eintragen | | 3 | Als Bis-Datum das Datum gestrige Datum eintragen | | Ein Fehler wird angezeigt, das Bis-Datum muss grösser oder gleich dem Von-Datum sein |

Tabelle : Testfälle 3 - Administration

## Software-Architektur

In diesem Kapitel wird die allgemeine Architektur festgelegt. Zwar werden noch keine Klassendiagramme festgelegt, doch der allgemeine Aufbau der einzelnen Komponenten und ihr Zusammenspiel wird festgehalten.

### C:\IPA\HappyMeter\UML\ZLFBKiGW4BpFLr2vxozO-r1FUafvilP00cOf2UWmsIgrykz24f5ioEeHfhjdkWUsZZXIrse0BL-8aMMecXCneL3MfAm1C5P8LjoZDIIDADWNCFPyoucN4i-gbiTmVh2D.pngÜbersicht

Abbildung : Komponentenübersicht

Das Backend ist als RESTful Service umgesetzt und bietet eine REST-Schnittstelle an, welche über HTTP & JSON kommuniziert. Alle im Controller aufgeführten Komponenten können von aussen angesprochen werden und bilden die Schnittstelle zum Frontend. Das Frontend spricht diese Schnittstellen via HTTP-Anfragen an und stellt die geladenen Daten im Browser dar.

### Autorisierung

Die Autorisierung ist selbst implementiert und möglichst simpel gehalten. Da die Website ohne HTTPS und firmenintern betrieben wird, ist die Sicherheit der Übermittlung kein kritischer Faktor. Bei der Anmeldung speichert das Frontend die Anmeldedaten im Browser-Sessionstorage, welcher für solch sensitive Daten ausreichend ist. Bei jeder Abfrage werden am Server Benutzername und Passwort mitgeteilt, damit dieser sicherstellen kann, dass man autorisiert ist.

### Frontend

Das Frontend verfügt Angular (2+) über Hilfsmittel, welche das Trennen von Verantwortlichkeiten erleichtern. Das Frontend verfügt neben der Darstellung über drei Dienste, welche mit dem Server kommunizieren. Bei einer Serveranfrage wird ein HTTP-Anfrage erstellt und an den Server übermittelt. Falls der Benutzer angemeldet ist sind in dieser Anfrage auch die Login-Daten zur Autorisierung vorhanden. Sobald der Server antwortet wird je nach Fall anders mit den Daten umgegangen. Einheitlich werden die HTTP-Codes 500 (Internal Server Error) und 401 (Unauthorized) behandelt. Im Falle eines 500 wird ein Dialog angezeigt, dass etwas schiefgelaufen ist. Im Falle einer 401-Antwort wird der Benutzer auf die Login-Seite weitergeleitet, falls er nicht bereits auf dieser ist.

### Backend

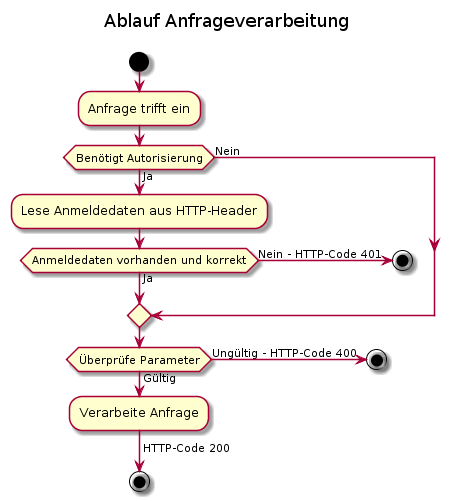
Da REST per Definition Stateless ist (jede Anfrage wird isoliert für sich bearbeitet), werden Autorisierungen über das «AuthorizationAttribute» gelöst. Dieses Attribut wird selbst implementiert und kann an jede Server-Schnittstelle angebracht werden. Sobald das Attribut aktiv ist werden alle Anfragen zuerst auf Autorisierung überprüft und sonst zurückgewiesen.

Abbildung : Ablauf Anfrageverarbeitung

## Datenbankdesign

In diesem Kapitel wird die Struktur der Datenbank festgelegt. Die Struktur der MySQL Datenbank wird mit der MySQL-Workbench erstellt. Eine Erweiterung der emotional\_state Tabelle zum Ergänzen eines Ortes sollte keine Schwierigkeiten darstellen.

### ERD

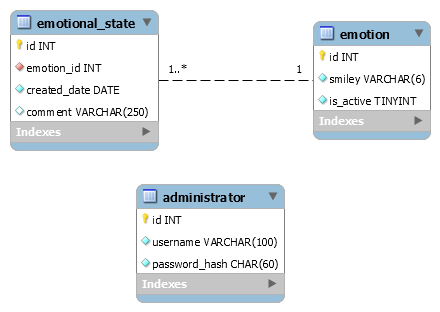


Abbildung : ERD

### Entitätsbeschreibungen

|  |  |
| --- | --- |
| emotional\_state (Gefühlslage) | |
| Beschreibung | Eine erfasste Gefühlslage mit einem optionalen Kommentar |
| Verbindungen | Eine Gefühlslage hat immer exakt eine zugewiesene Gefühlslageoption |
| Primärschlüssel | id |
| Felder | |  |  |  |  | | --- | --- | --- | --- | | Feld | Typ | Beschreibung | Sonstiges | | id | INT | Die eindeutige Identifikation | Pflichtfeld  Generiert | | emotion\_id | INT | Die dazugehörige Emotion | Pflichtfeld | | created\_date | DATE | Das Datum, an welchem die Gefühlslage erfasst wurde | Pflichtfeld | | comment | VARCHAR(250) | Ein optionaler Kommentar mit maximal 250 Zeichen |  | |

Tabelle : Entitätsbeschreibung emotional\_state

|  |  |
| --- | --- |
| emotion (Gefühlslageoption) | |
| Beschreibung | Eine Gefühlslageoption mit einem Smiley und Aktiv-Zustand |
| Verbindungen | Eine Gefühlslageoption hat mehrere Gefühlslagen |
| Primärschlüssel | id |
| Felder | |  |  |  |  | | --- | --- | --- | --- | | Feld | Typ | Beschreibung | Sonstiges | | id | INT | Die eindeutige Identifikation | Pflichtfeld  Generiert | | smiley | VARCHAR(6) | Der Unicode-Charaktercode als hexadezimaler string  Bsp. Für 1F383 für 🎃 | Pflichtfeld | | is\_active | TINYINT(1)  (BOOLEAN) | Ob diese Gefühlslageoption ausgewählt werden kann | Pflichtfeld  Default: 1 | |

Tabelle : Entitätsbeschreibung emotion

|  |  |
| --- | --- |
| administrator | |
| Beschreibung | Ein Administratorbenutzer |
| Primärschlüssel | id |
| Felder | |  |  |  |  | | --- | --- | --- | --- | | Feld | Typ | Beschreibung | Sonstiges | | id | INT | Die eindeutige Identifikation | Pflichtfeld  Generiert | | username | VARCHAR(100) | Der Unicode-Charaktercode als hexadezimaler string  Bsp. Für 1F383 für 🎃. | Pflichtfeld | | password\_hash | CHAR(60) | Das Datum, an welchem die Gefühlslage erfasst wurde | Pflichtfeld | |

Tabelle : Entitätsbeschreibung administrator

# Entscheiden

In diesem Kapitel geht um die getroffenen Entscheidungen zu verwendeten Frameworks und dessen Versionen.

## Versionskontrolle

Für regelmässige Backups und risikofreies Programmieren ist eine gute Versionsverwaltung essenziell. Aus diesem Grund habe ich mich für Git entschieden, der De-Facto-Standard in moderner Codeverwaltung. Die grossen Vorteile von Git sind, dass es keinen Server benötigt und man ohne Probleme zu einem funktionierenden Stand zurückgehen kann. Die Arbeit kann also auch ohne Internetverbindung problemlos fortgesetzt werden, falls es einmal Netzwerkschwankungen gibt. Falls man dann mit dem Internet verbunden ist kann man aus diversen Online-Git-Anbietern auswählen oder einen firmeninternen Git-Server benutzen. Da ich gute Erfahrungen mit GitHub gemacht habe und ein privates Repository erstellen kann, werde ich diesen Online-Dienst für meine Backups benutzen.

## Backend-Framework

Microsoft arbeitet seit geraumer Zeit an der plattformunabhängigen C#-Implementation .NET Core. Diese Core Implementation sind übersichtlicher, schneller und moderner. Microsoft hat eine [Offizielle Seite](https://docs.microsoft.com/en-us/dotnet/standard/choosing-core-framework-server), auf welcher alle Vergleiche aufgelistet werden. Die Hauptunterschiede belassen sich darauf, dass .NET Core auch unter macOS und Linux funktioniert.

Schlussendlich habe ich mich für das herkömmliche ASP.NET mit Web API 2 entschieden (basierend auf dem Windows .NET Framework). ASP.NET Core wäre zwar sicher die bessere Variante für ein Schulprojekt, jedoch besitze ich damit keine Erfahrungen. Ausserdem arbeitet meine Abteilung exklusiv mit ASP.NET und Web API 2, der Hauptgrund für die Entscheidung dieses Frameworks.

## Frontend-Framework

Als Frontend- Framework wird Angular4 eingesetzt. Sowohl in der Abteilung als in Schulprojekten habe ich Erfahrungen mit diesem von Google entwickelte Single-Page-Application-Framework gesammelt, weshalb ich es weiterhin verwenden möchte. Erst kürzlich ist Angular5 erschienen, eine neuere Version mit diversen Verbesserungen. Diese sind jedoch für mein Projekt nicht relevant, weshalb ich mit der stabilen Angular4 Version arbeiten werde.

## Smiley-Umsetzung

Als Vorbereitung für dieses Projekt habe ich mich grob mit dem Thema Emoji im Web auseinandergesetzt. Dabei auffallend ist vor allem, dass die Darstellung der Emojis stark vom Browser und Betriebssystem abhängig ist, wie man am [Beispiel des Kürbis-Emojis online sehen kann](https://emojipedia.org/jack-o-lantern/). So unterstützen alle Browser unter Windows 10 Emojis, unter Windows 7 hingegen nur der Firefox. Die anderen Browser stellen Emojis zwar dar, jedoch als Schwarz-Weiss-Skizze und nicht als farbenfrohes Emoji. Deshalb habe ich mich dazu entschieden alle Emojis plattformunabhängig darzustellen. Ein Open-Source-Standard von Twitter namens [Twemoji](https://twitter.github.io/twemoji/preview.html) hat schnell meine Aufmerksamkeit auf sich gezogen. Die verschiedenen Emojis können über eine URL angesprochen werden, welches einfach aus dem Unicode-Zeichen zusammengesetzt wird. Ich habe mich daher dafür entschieden die Eingabe über das Emoji-Zeichen zu gestalten (aus einer Auswahl) und diese danach als Twemoji auf allen Plattformen gleich darzustellen.

## Diagramm-Visualisierung

Wie in allen Schulprojekten werde ich [PlantUML](http://plantuml.com/) beziehungsweise die Web-Version [PlantText](https://www.planttext.com/) für die Darstellung meiner Diagramme verwenden. Im Unterschied zu Visio und ähnlichen Programmen erstellt man die Diagramme nicht in einem GUI-Editor, sondern in einem Text-Editor. Man programmiert sozusagen Diagramme und muss sich nicht um die korrekte UML-Darstellung kümmern. Da es gratis zur Verfügung steht und auch in meiner Abteilung oft verwendet wird werde ich all meine Architektur-Diagramme mithilfe dieses Tools erstellen.

# Realisieren

## Backend

### Klassendiagramm

PlantUML erstelltes Diagramm

### Klassenübersicht

## Frontend

# Kontrollieren

## Zeitmanagement

Wurden Meilensteine erfüllt?

## Funktionalität

Sind Anforderungen vollständig erfüllt? Wurden Tests bestanden?

## Codequalität

Erfüllt die Codequalität die Anforderungen? Bin ich mit der Codequalität zufrieden? (Kommentare, Erweiterbarkeit, Clean-Code)

## Fertigstellung

Projekt finalisieren und abgeben.

# Auswerten

## Reflexion

Wurden alle Ziele erreicht?

## Verbesserungen / Erfahrungen

Welche Verbesserungen am Projekt gibt es? Was würde ich anders machen? Was habe ich gelernt?

## Einsetzbarkeit

Kann es Produktiv eingesetzt werden?

## Fazit

Wie verlief das Projekt?

# Verzeichnisse

## Quellenverzeichnis

**Im aktuellen Dokument sind keine Quellen vorhanden.**

## Abbildverzeichnis

[Abbildung 1: Zeitplan 9](file:///C:\Users\Pascal\Desktop\IPA-Dokumentation.docx#_Toc497423219)

[Abbildung 2: Use-Case Diagramm 18](file:///C:\Users\Pascal\Desktop\IPA-Dokumentation.docx#_Toc497423220)

[Abbildung 3: Komponentenübersicht 28](file:///C:\Users\Pascal\Desktop\IPA-Dokumentation.docx#_Toc497423221)

[Abbildung 4: Ablauf Anfrageverarbeitung 29](file:///C:\Users\Pascal\Desktop\IPA-Dokumentation.docx#_Toc497423222)

[Abbildung 5: ERD 30](#_Toc497423223)

## Tabellenverzeichnis

[Tabelle 1: Projektteilnehmer 6](#_Toc497423194)

[Tabelle 2: Meilensteine 8](#_Toc497423195)

[Tabelle 3: Konkrete Planung Tag 1 10](#_Toc497423196)

[Tabelle 4: Konkrete Planung Tag 2 11](#_Toc497423197)

[Tabelle 5: Konkrete Planung Tag 3 13](#_Toc497423198)

[Tabelle 6: Use-Case 1.1 19](#_Toc497423199)

[Tabelle 7: Use-Case 1.2 19](#_Toc497423200)

[Tabelle 8: Use-Case 1.3 20](#_Toc497423201)

[Tabelle 9: Use-Case 2.1 20](#_Toc497423202)

[Tabelle 10: Use-Case 2.2 21](#_Toc497423203)

[Tabelle 11: Use-Case 2.3 21](#_Toc497423204)

[Tabelle 12: Use-Case 3.1 21](#_Toc497423205)

[Tabelle 13: Use-Case 3.2 22](#_Toc497423206)

[Tabelle 14: Use-Case 3.3 22](#_Toc497423207)

[Tabelle 15: Use-Case 3.4 22](#_Toc497423208)

[Tabelle 16: Use-Case 3.5 23](#_Toc497423209)

[Tabelle 17: Use-Case 3.6 23](#_Toc497423210)

[Tabelle 18: Use-Case 3.7 23](#_Toc497423211)

[Tabelle 19: Use-Case 3.8 24](#_Toc497423212)

[Tabelle 20: Testfälle 1 - Startseite 25](#_Toc497423213)

[Tabelle 21: Testfälle 2 - Anmeldung 25](#_Toc497423214)

[Tabelle 22: Testfälle 3 - Administration 28](#_Toc497423215)

[Tabelle 23: Entitätsbeschreibung emotional\_state 31](#_Toc497423216)

[Tabelle 24: Entitätsbeschreibung emotion 31](#_Toc497423217)

[Tabelle 25: Entitätsbeschreibung administrator 32](#_Toc497423218)

# Quellcode