Hey I'm in

autor(en) : Pascal Honegger

version : 0.1

status : Draft

quelle : Atos

dokumentendatum : 06 März 2018

anzahl der seiten : 22

Änderungshistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Beschreibung | Autor(en) |
| 0.1 | 06.03.2018 | Initiale Version erstellt | Pascal Honegger |
| 1.0 | 21.03.2018 | Erster Release & Abgabe | Pascal Honegger |

# Inhalt

**Teil 1**

1 Aufgabenstellung 4

2 Projektorganisation 5

2.1 Beteiligte Personen 5

2.2 Projektmanagementmethode 5

2.3 Backup-Konzept 5

3 Ausgangslage 6

3.1 Vorkenntnisse 6

3.2 Vorarbeit 6

3.3 Firmenstandards 6

4 Zeitplanung 7

4.1 Meilensteine 7

4.2 Verhalten im Falle von Verzug 7

4.3 Gantt-Diagramm 8

5 Arbeitsprotokoll 9

5.1 Tag 1 (06.03.2018) 9

6 Kurzfassung 10

6.1 Kurze Ausgangssituation 10

6.2 Umsetzung 10

6.3 Ergebnis 10

**Teil 2**

7 Informieren 11

7.1 Ist-Analyse 11

7.2 Soll-Analyse 11

7.3 Systemübersicht 11

8 Planen 12

8.1 Use-Cases 12

8.2 Testkonzept 13

8.3 Software-Architektur 14

8.4 Datenbankdesign 15

9 Entscheiden 16

10 Realisieren 17

10.1 Backend-Klassenübersicht 17

10.2 Klassenname 17

10.3 Frontend-Klassenübersicht 17

11 Kontrollieren 18

11.1 Zeitmanagement 18

11.2 Tests 18

11.3 Testresultate: Browserkompatibilität 18

11.4 Testfazit 18

12 Auswerten 19

12.1 Reflexion 19

12.2 Erfahrungen 19

12.3 Einsetzbarkeit 19

12.4 Erweiterungen 19

12.5 Schlusswort 19

**Anhang**

13 Glossar 20

14 Verzeichnisse 21

14.1 Abbildverzeichnis 21

14.2 Tabellenverzeichnis 21

15 Quellcode 22

15.1 Frontend 22

15.2 Backend 22

# Aufgabenstellung

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

## Ausgangslage

Es gibt bereits zahlreiche grosse Anbieter für die Planung der Teilnehmer von Einzelevents. Einer der bekanntesten ist sicherlich doodle.com. Wiederkehrende Ereignisse werden hingegen nur sehr rudimentär unterstützt. Im Arbeitsumfeld von der Atos gibt es zahlreiche Ereignisse, welche sich regelmässig wiederholen und viele Rahmenbedingungen (z.B. der potentielle Teilnehmerkreis) unverändert bleiben.

## Detaillierte Aufgabenstellung

Ziel dieser IPA Arbeit ist die Erstellung einer responsive Web Applikation mit welcher die Benutzer der Rolle Organisator ihre Eventserien planen und publizieren können. Dabei können sie die Metadaten (Titel, Beschreibung, Ort, Einzeltermine, Kosten, Antwortmöglichkeiten für die Teilnehmer) erfassen, die Notifikationen an die potentiellen Teilnehmer festlegen und die Antworten und Kosten der Teilnehmer verwalten. Termine eines Events können vom Organisator abgesagt, verschoben oder neu erstellt werden.

Benutzer mit der Rolle Teilnehmer können sich für eine Eventserie registrieren und zu jedem Termin ihre Teilnahme planen. Über eine Kommentarfunktion können die Teilnehmer zusätzliche Informationen zu ihrer Teilnahme bereitstellen. Sie können die Notifikationen der Organisatoren nutzen, um sich an einen bevorstehenden Termin erinnern zu lassen und sich eine Zusammenfassung (Metadaten + Teilnehmer) des bevorstehendes Termins zusenden zu lassen. Die Notifikationen werden per Mail versendet und sollen einen temporär gültigen Zugangslink (inkl. Authentisierung) zur Web Applikation enthalten.

Für Benutzer ist ein Self-Service zum Anlegen, Löschen und Passwort zurücksetzen erforderlich.

In den Vorarbeiten werden zusammen mit dem Auftragsgeber GUI Mockups erstellt. Der Auftraggeber soll sich dabei auf die Anforderungen konzentrieren, während es die Aufgabe des Kandidaten sein soll, diese Anforderungen benutzerfreundlich umzusetzen. Die Mockups sind nicht fix und können während der Arbeit durch den Kandidaten verbessert werden.

Folgende Features sind optional und werden in einem späteren Release nachgeliefert (grau hinterlegt in den Mockups):

* Kommentarfunktion
* Verwaltung der Kosten

### Fehlerbehandlung

* In keinem Fall darf das Backend abstürzen
* Alle Fehler müssen im Minimum geloggt werden
* Exceptions müssen mit den "Inner Exceptions" und allen Callstacks geloggt werden
* Fehler sollen dem Benutzer in einem für ihn verständlichen Format angezeigt werden
* Alle Eingabefelder sollen validiert werden. Validierungsfehler werden dem Benutzer angezeigt.

### Ereignisprotokoll / Logging

* Wichtige Ereignisse sollen in eine eigene Textdatei geschrieben werden und mindestens 6 Montate erhalten bleiben (Audit-Log). Diese sind:
* Benutzer erstellen / löschen
* Benutzer loggt sich ein / loggt sich aus
* Änderungen an Terminen und Zusagen
* Normale Logs sollen:
* begrenzt auf maximal 5 Dateien à 20MB sein
* sollten für die Auswertung möglichst maschinenlesbar sein (z.B. per logstash). Mögliche Formate: JSON, XML oder Text per Regex parsbar
* Wichtige Eigenschaften sollten zur besseren Auswertung als "Log Context Properties" geloggt werden, z.B. Benutzername, Client Hostname, Dauer einer Anfrage
* Folgende Eigenschaften müssen in der Logdatei vorhanden sein: Zeit des Ereignisses, Severity (TRACE ... FATAL), Namespace mit Klasse, Methodenname, Threadname, Meldungstext und evt. Kontextvariablen
* Die Konfiguration des Loggers soll per Datei editierbar sein. Änderungen sollten ohne Neustart erkannt und angewendet werden. Die Datei muss nicht per GUI veränderbar sein.

### Weitere Anforderungen

* Browser: >= IE11, Chrome >= V63

### Erwartete Lieferartefakte

* Code abgelegt in GIT Projekt (z.B. privates GitHub Projekt)
* funktionsfähiges Kompilat in GIT abgelegt
* Applikation in Azure deployed
* Einfaches Deployment von Updates in die Azure Cloud
* Technische Dokumentation, Architekturbeschreibung

## Mittel und Methoden

Betriebsystem: Azure Windows Web-App (lokal Windows)

Versionsverwaltung: GIT

GUI: Farben und Logos gemäss Atos Firmenstandard

### Backend:

* C# / .NET 4.7.x
* ASP.NET Web API 2.0
* SQL Server 2017
* Entity Framework (OR Mapper)
* Visual Studio 2017
* Resharper
* DI mit Autofac

### Frontend:

* HTML 5
* Angular 5
* Typescript
* Webpack
* Material Design

# Projektorganisation

## Beteiligte Personen

|  |  |  |  |
| --- | --- | --- | --- |
| Position | Name | E-Mail | Telefonnummer |
| Auftraggeber | Christoph Fauti | [christoph.fauti@atos.net](mailto:christoph.fauti@atos.net) | - |
| Verantwortlicher Fachkraft | 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 | Guido Jansen | [guidojansen@gmx.ch](mailto:guidojansen@gmx.ch) | - |
| Zweit-Experte | Philip Karrer | [pipo.karrer+pkorg@gmail.com](mailto:pipo.karrer+pkorg@gmail.com) | - |
| Validierungsexperte | Tilo Steiger | [steiger@id.ethz.ch](mailto:steiger@id.ethz.ch) | - |

Tabelle 1: Projektteilnehmer

**UNTERKAPITEL PRO PERSON?**

## Projektmanagementmethode

## Backup-Konzept

# Ausgangslage

## Vorkenntnisse

## Vorarbeit

## Firmenstandards

# Zeitplanung

## Meilensteine

|  |  |  |
| --- | --- | --- |
| Meilenstein | Beschreibung | Datum |
| Projektstart | Start des Projektes | 06.03.2018 09:00 |
| Projektabschluss | Erfolgreicher Projektabschluss | 21.03.2018 17:00 |

Tabelle 2: Meilensteine

## Verhalten im Falle von Verzug

## Gantt-Diagramm



Abbildung 1: Zeitplan

# Arbeitsprotokoll

## Tag 1 (06.03.2018)

### Planung

Heute möchte ich mit dem Projekt beginnen, die Zeitplanung abschliessen und die erste IPERKA-Phase (Informieren) abschliessen. Die restliche Zeit investiere ich mit den Use-Cases.

|  |  |  |  |
| --- | --- | --- | --- |
| Was | Soll-Dauer | Ist-Dauer | Fertig |
| Projektbeginn | 15min |  | ✔ |
| Zeitplanung & Meilensteine | 2h 15min |  |  |
| Projektorganisation dokumentieren | 30min |  |  |
| Ausgangslage dokumentieren | 30min |  |  |
| Ist- und Soll-Analyse | 1h |  |  |
| Systemübersicht | 30min |  |  |
| Use-Cases | 2h 30min |  |  |
| Arbeitsprotokoll | 30min |  |  |

Tabelle 3: Konkrete Planung Tag 1

### Erledigte Arbeit

Zusammenfassung der Tabelle

Bsp. GitHub aufgesetzt, Komponenten realisiert etc.

### Probleme

Bsp. Zeitplant zu knapp, unerwartete Fehler…

### Reflexion

Fazit / Gelerntes

# Kurzfassung

## Kurze Ausgangssituation

## Umsetzung

## Ergebnis

# Informieren

## Ist-Analyse

Eine Analyse des (noch nicht) vorhandenen Systems.

## Soll-Analyse

Eine genauere Analyse der Anforderungen und des Soll-Zustands.

### Frontend

### Backend

### Datenbank

## Systemübersicht

# Planen

## Use-Cases

### Use-Case Diagramm

Beschreibung zur Aufteilung des Diagrammes.

### Use-Cases 1: Gruppe 1

Alle Use-Cases in diesem Kapitel werden von Testbenutzern durchgeführt.

|  |  |
| --- | --- |
| Use-Case 1.1: Tagesabhängige Gefühlslage sehen | |
| Beschreibung | Eine aussagekräftige Beschreibung |
| Ergebnis / Nachbedingung | Detaillierte Beschreibung des erwarteten Ergebnisses |
| Vorbedingung | Vorbedingungen für diesen Use-Case |
| Ablauf | |  |  | | --- | --- | | Schritt | Aktion | | 1 | Erster Schritt | | 2 | Weiterer Schritt | |

Tabelle 8: Use-Case 1.1

## 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. Die Testfallnummer besitzt als Präfix die Nummer des dazugehörigen Use-Cases.

### Eingesetzte Testmittel und -Methoden

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

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

### Testfälle 1.1: Gruppe 1

Diese Tests können von Benutzern und Administratoren durchgeführt werden.

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Vorbedingung | Testschritte | Erwartetes Ergebnis |
| 1.1.1 | Der Online-Rechner ist konfiguriert | |  |  | | --- | --- | | 1 | Startseite öffnen | | 2 | «1 + 1» eingeben | | 3 | «Rechnen» drücken | | Die Zahl 2 wird dargestellt |

Tabelle 10: Testfälle 1.1 – Gruppe 1

## 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.

### Übersicht

\*Komponenten- / Klassendiagramm\*

Das Backend ist als REST Service umgesetzt und bietet eine REST-Schnittstelle an, welche über HTTP(S) und JSON kommuniziert. Alle im Controller aufgeführten Methoden können von aussen angesprochen werden und bilden die Schnittstelle zum Frontend.

## Datenbankdesign

In diesem Kapitel wird die Struktur der Datenbank festgelegt. Die Struktur der MSSQL-Datenbank wird direkt im Code erstellt und danach durch das Entity Framework in Tabellen umgewandelt. Hierbei werden auch Migrationen beim Umbau des Models und das korrekte Mapping von Klassen- zu Datenbanktypen übernommen.

### ERD

\*Diagramm\*

### Entitätsbeschreibungen

|  |  |
| --- | --- |
| TableName (Gefühlslage) | |
| Beschreibung | Was wird in dieser Tabelle gespeichert? |
| Verbindungen | Welche Relationen bestehen weshalb? |
| Primärschlüssel | id |
| Felder | |  |  |  |  | | --- | --- | --- | --- | | Feld | Typ | Beschreibung | Sonstiges | | id | INT | Die eindeutige Identifikation | Pflichtfeld  Generiert | | created\_date | DATE | Erstelldatum | Pflichtfeld | | Example | VARCHAR(250) | Dummy Textfeld |  | |

Tabelle 12: Entitätsbeschreibung TableName

# Entscheiden

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

# Realisieren

In diesem Kapitel werden auf die Details der Implementierung eingegangen.

## Backend-Klassenübersicht

Kurze Beschreibung zu den Kernkonzepten im Backend.

## Klassenname

Beschreibung

## Frontend-Klassenübersicht

Kurze Beschreibung zu den Kernkonzepten im Frontend.

### Klassenname

Beschreibung

# Kontrollieren

In diesem Kapitel wird die Zielerreichung kontrolliert und entschieden, ob das Projekt abgeschlossen ist.

## Zeitmanagement

|  |  |  |
| --- | --- | --- |
| Meilenstein | Gesetzes Datum | Effektives Datum |
| Projektstart | 01.11.2017 09:00 | 01.11.2017 09:00 |
| Projektabschluss | 10.11.2017 17:00 | 10.11.2017 17:30 |

## Tests

Die Anforderungen wurden alle umgesetzt. Zur Überprüfung der umgesetzten Anforderungen werden die geplanten Tests durchgeführt.

✔ = Bestanden

✔ (**~**) = Bestanden mit Abweichung

✘ = Fehlgeschlagen

### Testresultate 1: Startseite

|  |  |  |
| --- | --- | --- |
| Nr. | Bestanden | Abweichungen |
| 1.1 | ✔ |  |
| 1.2 | ✔ (**~**) | Was war nicht wie erwartet? |
| 1.3 | ✘ | Was lief schief? |

Tabelle 15: Testresultate 1

## Testresultate: Browserkompatibilität

|  |  |  |
| --- | --- | --- |
| Browser | Kommentar | Unterstützt |
| Mozilla Firefox 58 |  | ✔ |
| Google Chrome 64 |  | ✔ (**~**) |
| Internet Explorer 11 |  | ✘ |

Tabelle 17: Testresultate Browserkompatibilität

Der minimale Browsersupport ist gewährleistet.

## Testfazit

Anzahl bestandener Testfälle

Details zu fehlgeschlagenen Testfällen

Fazit: Ist Programm verwendbar?

Rechtfertigung Fazit

# Auswerten

Das komplette Projekt wird nochmals reflektiert und ausgewertet.

## Reflexion

Rückblick

## Erfahrungen

Gelerntes

## Einsetzbarkeit

Kann die Software verwendet werden? Wie wird sie verwendet werden?

## Erweiterungen

Ist die Erweiterbarkeit gewährleistet?

## Schlusswort

# Glossar

|  |  |
| --- | --- |
| Begriff | Erklärung |
| Projektspezifischer Begriff | Beschreibung |

Tabelle 18: Glossar

# Verzeichnisse

## Abbildverzeichnis

Die Abbildungen in diesem Dokument sind alle selbst erstellt.

[Abbildung 1: Zeitplan 8](file:///C:\Users\Pascal\Desktop\HeyImIn\Dokumentation\IPA-Dokumentation_Vorlage.docx#_Toc508050375)

## Tabellenverzeichnis

[Tabelle 1: Projektteilnehmer 5](#_Toc508050366)

[Tabelle 2: Meilensteine 7](#_Toc508050367)

[Tabelle 3: Konkrete Planung Tag 1 9](#_Toc508050368)

[Tabelle 8: Use-Case 1.1 13](#_Toc508050369)

[Tabelle 10: Testfälle 1.1 – Gruppe 1 13](#_Toc508050370)

[Tabelle 12: Entitätsbeschreibung TableName 15](#_Toc508050371)

[Tabelle 15: Testresultate 1 18](#_Toc508050372)

[Tabelle 17: Testresultate Browserkompatibilität 18](#_Toc508050373)

[Tabelle 18: Glossar 20](#_Toc508050374)

# Quellcode

## Frontend

### File.html

## Backend

### File.cs