|  |  |
| --- | --- |
| Dokumentart | **Dokumentation** |
|
| Titel | **IPA Bericht** |
| Produkt | **Webbasierter Signalzeitenplan** |
| Projekt | **Individuelle Praktische Arbeit (IPA)** |

|  |  |  |  |
| --- | --- | --- | --- |
| Erstellt | Alain Keller | MO CDC VT | 05.04.2017 |
|  |  | Org.-Einheit | Datum |

Inhaltsverzeichnis

[Teil 1: Umfeld & Ablauf 4](#_Toc497384728)

[1 Aufgabenstellung 4](#_Toc497384729)

[1.1 Funktionen nicht angemeldeter Nutzer 4](#_Toc497384730)

[1.2 Funktionen angemeldeter Nutzer 4](#_Toc497384731)

[1.3 Funktionen Admin 4](#_Toc497384732)

[1.4 Technologien 4](#_Toc497384733)

[1.5 Wahlkriterien 5](#_Toc497384734)

[2 Projektorganisation 5](#_Toc497384735)

[2.1 Beteiligte Personen 5](#_Toc497384736)

[3 Ausgangslage 5](#_Toc497384737)

[3.1 Vorkenntnisse 5](#_Toc497384738)

[4 Mittel & Methoden 5](#_Toc497384739)

[4.1 Software 5](#_Toc497384740)

[4.2 Style Guide and Coding Conventions 5](#_Toc497384741)

[5 Firmenstandards 6](#_Toc497384742)

[5.1 Dokumentationsvorlage 6](#_Toc497384743)

[6 Zeitplan 7](#_Toc497384744)

[6.1 Meilensteine 10](#_Toc497384745)

[7 Arbeitsprotokoll 11](#_Toc497384746)

[7.1 Mittwoch, 01.11.2017 11](#_Toc497384747)

[7.2 Donnerstag, 02.11.2017 12](#_Toc497384748)

[7.3 Freitag, 03.11.2017 13](#_Toc497384749)

[7.4 Mittwoch, 08.11.2017 14](#_Toc497384750)

[7.5 Freitag, 10.11.2017 15](#_Toc497384751)

[Teil 2: Projekt 16](#_Toc497384752)

[1 Kurzfassung IPA Bericht 16](#_Toc497384753)

[1.1 Ausgangssituation 16](#_Toc497384754)

[1.2 Umsetzung 16](#_Toc497384755)

[1.3 Ergebnis 16](#_Toc497384756)

[2 Projektmanagement 17](#_Toc497384757)

[2.1 Versionierung 17](#_Toc497384758)

[2.2 Backupsystem 17](#_Toc497384759)

[2.3 Entwicklungsumgebungen 17](#_Toc497384760)

[3 Informieren 17](#_Toc497384761)

[3.1 Ist-Analyse 17](#_Toc497384762)

[3.2 Soll-Analyse 17](#_Toc497384763)

[4 Planen 19](#_Toc497384764)

[4.1 Funktionale Anforderungen 19](#_Toc497384765)

[4.2 Nicht funktionale Anforderungen 20](#_Toc497384766)

[4.3 Priorisieren 20](#_Toc497384767)

[4.4 Architektur 20](#_Toc497384768)

[4.5 Testkonzept 25](#_Toc497384769)

[4.6 Testfälle 27](#_Toc497384770)

[5 Entscheiden 28](#_Toc497384771)

[6 Realisieren 28](#_Toc497384772)

[6.1 Verzeichnisstruktur 28](#_Toc497384773)

[6.2 Source Code 28](#_Toc497384774)

[6.3 HTML Filet 28](#_Toc497384775)

[6.4 CSS Filet 30](#_Toc497384776)

[6.5 JavaScript Filet 31](#_Toc497384777)

[6.6 Installationsanweisung 33](#_Toc497384778)

[7 Kontrollieren 33](#_Toc497384779)

[7.1 Testübersicht 33](#_Toc497384780)

[7.2 Testfazit 33](#_Toc497384781)

[7.3 Testprotokoll 34](#_Toc497384782)

[8 Auswerten 35](#_Toc497384783)

[8.1 Erfüllung Anforderungen 35](#_Toc497384784)

[8.2 Stand der Applikation 36](#_Toc497384785)

[8.3 Reflexion 36](#_Toc497384786)

[8.4 Fazit 36](#_Toc497384787)

[9 Abbildungsverzeichnis 37](#_Toc497384788)

[10 Tabellenverzeichnis 38](#_Toc497384789)

[11 Literaturverzeichnis 39](#_Toc497384790)

[12 Quellenverzeichnis 39](#_Toc497384791)

[13 Glossar 39](#_Toc497384792)

Teil 1: Umfeld & Ablauf

# Aufgabenstellung

Es soll eine Webseite erstellt werden, über welche man Inserate erstellen kann. Auf der Webseite können Benutzer Inserate für Produkte und Dienstleistungen erstellen und ansehen. Die Webseite richtet sich an Desktop und Laptop-User. Ein Responsive-Design ist nicht Pflicht.

Die Benutzer der Webseite können sich über ein Login mit Benutzernamen und Passwort anmelden. Hat ein User noch kein Login, kann er sich registrieren. Es werden folgende Daten für die Registration benötigt: Name, Vorname, Benutzername, E-Mail, Telefonnummer und das Passwort. Die Registration muss nicht über E-Mail bestätigt werden. Nur angemeldete Benutzer können Inserate erstellen. Es können auch die eigenen Inserate angesehen werden, und bei Bedarf deaktiviert werden, falls diese nicht mehr aktuell sind. Deaktivierte Inserate werden für die anderen Benutzer nicht mehr angezeigt. Das Login wird nicht gespeichert und der Benutzer muss sich bei Neustart des Browsers neu anmelden. Der Benutzer kann sich auch während dem Benutzen der Website abmelden. Zusätzlich gibt es ein Admin Account. Dieser kann alle Inserate ansehen und deaktivieren. Jedoch darf er keine Inserate erfassen.   
Auf der Mainpage sieht der Benutzer die Inserate der anderen Personen. Es werden pro Seite 10 Inserate angezeigt und über eine Seitennavigation am unteren Ende der Seite kann man zwischen den Seiten navigieren. Die Inserate werden nach dem Erstelldatum Sortiert. Weitere Filter, und Sortiermöglichkeiten sind nicht vorhanden. Öffnet man ein Inserat wird man auf eine detailliertere Seite weitergeleitet, welche weitere Informationen enthält. Ein Inserat enthält folgende Informationen: Titel, Beschreibung, Vorname Ersteller, E-Mail-Adresse Ersteller, Telefonnummer Ersteller, Ort und Art des Angebotes. Auf der Hauptübersicht sieht man den Titel, die Art und der Ort.

## Funktionen nicht angemeldeter Nutzer

Nicht anmeldete Nutzer können die Inserate der anderen Nutzer ansehen. Sie dürfen keine Inserate erfasse oder Deaktivieren.

## Funktionen angemeldeter Nutzer

Angemeldete Nutzer dürfen Inserate erfassen, ihre eigenen deaktivieren und die Inserate anderer Nutzer einsehen. Zusätzlich hat der angemeldete Nutzer die gleichen Funktionen wie der nicht angemeldete Nutzer.

## Funktionen Admin

Der Admin darf Inserate von anderen Nutzern deaktivieren und somit aus der öffentlichen Anzeige entfernen. Er darf keine Inserate erstellen.

## Technologien

Zur Entwicklung der Seite wird HTML, CSS, Javascript, JQuery & PHP verwendet. Der Server wird in PHP 7 objektorientiert implementiert. Für das Frontend wird HTML und CSS verwendet. Zur Steigerung der Benutzerfreundlichkeit werden zusätzliche Funktionen mit Javascript und JQuery verwendet. Die Datenbank wird mit MySQL entwickelt. Da die Webseite von mehreren Nutzern gleichzeitig benutzt werden kann, muss die Transaktionssicherheit gewährleistet sein.

## Wahlkriterien

225 – Versionsverwaltung

166 – Codingstyle, Lesbarer Code

167 – Coding Dokumentiert

123 – Kommentare im Quellcode

164 – Codierung: Fehlerbehandlung

125 – Gliederung des Programms

130 – Vollständiges ERM

# Projektorganisation

## Beteiligte Personen

|  |  |  |
| --- | --- | --- |
| Name | Funktion |  |
| Alain Keller | IPA-Kandidat |  |
| Remo Steinmann | Experte |  |

# Ausgangslage

## Vorkenntnisse

* HTML
* CSS
* Javascript
* PHP 7
* C#

# Mittel & Methoden

## Software

* Windows 10
* Word 2016
* Excel 2016
* PHP Storm 2017.1.3
* Google Chrome V 61.0.3163.100
* Mozilla Firefox

## Style Guide and Coding Conventions

# Firmenstandards

## Dokumentationsvorlage

Die verwendete Dokumentationsvorlage entspricht dem Siemens Firmenstandard und wurde mithilfe des Tools «Docugate» generiert.

# Zeitplan









## Meilensteine

|  |  |  |
| --- | --- | --- |
| **Meilenstein** | **Beschreibung** | **Datum** |
| Meilenstein 1: Start | Start des Projektes | 01.11.2017 |
| Meilenstein 2: Planung | Planung vollständig beendet, Start der Realisierung | 03.11.2017 |
| Meilenstein 3: Realisierung | Realisierung vollständig beendet, Start der Tests | 08.11.2017 |
| Meilenstein 4: Abgabe | Abgabe der IPA | 10.11.2017 |

Tabelle : Projektmeilensteine

# Arbeitsprotokoll

## Mittwoch, 01.11.2017

|  |  |
| --- | --- |
| **Tätigkeitsliste** | |
| **Nr./Thema** | **Beschreibung ausgeführte Tätigkeit** |
| **Teil 1 IPA Bericht** | Ich habe den ersten Teil des IPA-Berichters erstellt. Dazu gehörte der Projektauftrag, die Projektorganisation und meine Vorkenntnisse |
| **Teil 2 VCS** | Damit ich meine Dateien von Beginn an Versionisiert habe, erstellte ich heute das GIT-Repository. Dieses ist in GutHub unter folgendem Link erreichbar: <https://github.com/LordMcFungus/Probe-IPA>. |
| **Teil 2: Informieren** | Ich habe heute den Informieren-Teil von IPERKA abgeschlossen. Zum Informieren gehörte die Analyse der Ausgangslage und die Soll-Situation. |
| **Teil 2: Use Cases** | Anhand der Soll-Analyse erstellte ich die UseCase-diagramme für das Projekt. Diese erstellte ich mit Hilfe von PlantUml, einem Tool, um Textbasierte UMLS zu erstellen. Ein Vorteil dieses Tools ist es, dass man sich nicht mit dem Design auseinandersetzten muss. Dafür sollte man den Syntax kennen um Diagramme zu schreiben |
| **Teil 2: Komponenten** | Ich startete mit der Architektur. Dafür machte ich mir Gedanken welche Komponenten im Programm existieren werden, und was für Verantwortlichkeiten diese haben sollen. Ich habe schon damit begonnen |
| **Journal** | |
| **Erreichte & unerreichte Ziele** | * Informieren abgeschlossen ✔ * UseCase abgeschlossen ✔ * Architektur angefangen ✔ |
| **Aufgetretene Probleme & erarbeitete Lösungen** | Mir ist noch nicht ganz klar wie das Komponentendiagramm aussehen soll. Im Internet werden verschiedene Meinungen vertreten |
| **Reflexion** | Der Heutige Tag verlief recht gut. Ich konnte mich gut an den Zeitplan halten und konnte ohne grössere Probleme das Projekt starten. Es ist teils schwer in einem Raum mit 6 anderen Lernenden zu arbeite, ohne dass man teils abgelenkt ist. Jedoch war es oft still und ich konnte mich gut konzentrieren. |

Tabelle : Arbeitsprotokoll 01.11.2017

## Donnerstag, 02.11.2017

|  |  |
| --- | --- |
| **Tätigkeitsliste** | |
| **Nr./Thema** | **Beschreibung ausgeführte Tätigkeit** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **Journal** | |
|  |  |
|  |  |
|  |  |
|  |  |

Tabelle : Arbeitsprotokoll 22.03.2017

## Freitag, 03.11.2017

|  |  |
| --- | --- |
| **Tätigkeitsliste** | |
| **Nr./Thema** | **Beschreibung ausgeführte Tätigkeit** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **Journal** | |
|  |  |
|  |  |
|  |  |

Tabelle : Arbeitsprotokoll 23.03.2017

## Mittwoch, 08.11.2017

|  |  |
| --- | --- |
| **Tätigkeitsliste** | |
| **Nr./Thema** | **Beschreibung ausgeführte Tätigkeit** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **Journal** | |
| **Erreichte & unerreichte Ziele** |  |
| **Reflexion** |  |

Tabelle : Arbeitsprotokoll 24.03.2017

## Freitag, 10.11.2017

|  |  |  |
| --- | --- | --- |
| **Tätigkeitsliste** | | |
| **Nr./Thema** | | **Beschreibung ausgeführte Tätigkeit** |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
| **Journal** | | |
|  |  | |
|  |  | |
|  |  | |

Tabelle : Arbeitsprotokoll 28.03.2017

Teil 2: Projekt

# Kurzfassung IPA Bericht

## Ausgangssituation

## Umsetzung

## Ergebnis

# Projektmanagement

Als Projektmanagementmethode verwende ich in diesem Projekt IPERKA. Es ist einfach aufgebaut und eignet sich gut für Einzel Projekte über eine kurze Laufzeit

## Versionierung

Als VCS verwende ich Git über die Server von Github. Ich habe mich für Git entschieden, da ich in vergangenen Schulprojekten schon Git über GitHub verwendet habe.

## Backupsystem

Neben dem regelmässigen speichern auf GitHub wird am Ende eines jeden Tages eine Kopie auf Dropbox gespeichert. Dies stellt die Verfügbarkeit der Arbeit sicher, falls der Laptop und GitHub gleichzeitig nicht erreicht werden können.

## Entwicklungsumgebungen

Den Code entwickle ich in PHP-Storm. Es ist eine IntelliJ basierte IDE, welche ich auch schon von vergangenen Projekten kenne. Den Apache und MySql-Server verwalte ich über Xampp, da sich dieses gut für solche Projekte eignet und gratis verfügbar ist.

# Informieren

In diesem Teil trage ich die Informationen zusammen und beschreibe diese.

## Ist-Analyse

Zurzeit besteht noch nichts von der gewünschten Applikation.

## Soll-Analyse

Es soll eine WebApp erstellt werden, über welche Benutzer Inserate erstellen und ansehen können.

### Frontend

Das Frontend wird mit Hilfe von HTML, CSS und Javascript erstellt.

Der Benutzer sieht auf der Startseite die schon erstellten Inserate der anderen Benutzer. Diese werden in mehrere Seiten aufgeteilt. Je Seite werden 10 Inserate angezeigt. Auf der Inserat-Liste werden nicht alle Informationen angezeigt. Wählt der Nutzer ein Inserat aus, wird dieses Geöffnet und man sieht mehr Details. Der Benutzer kann sich auf der Webseite regsitrieren und Anmelden. Als angemeldeter Nutzer kann er ein eigenes Inserat erstellen, und die erstellten Inserate deaktivieren.

### Backend

Das Backend ist vor allem dazu da, die Schnitstelle zwischen Frontend und Datenbank zu machen. Die Logindaten werden übers Backend validiert, die Inserate werden ausgelesen und gespeichert. Das Backend soll objektorientiert mit Hilfe von PHP 7.1 geschrieben werden.

### Datenbank

Die Datenbank wird mit MySql gemacht. Die Transaktionssicherheit muss jederzeit gewährleistet sein. In der Datenbank werden die Benutzerdaten und die Inserate gespeichert.

### Design und Layout

Es stehen keine konkreten Anforderungen zum Design. Die Webseite muss jedoch nicht Responsive implementiert sein.

### Errorhandling

Fehler im Programm sollten erkannt werden und entsprechend damit umgegangen werden.

### Source Code

Der Source-Code soll verständlich sein und kommentiert. Es sollen die Clean-Code Guidelines eingehalten werden.

### Tests

Die Applikation wird nach der Implementation ausführlich gegen die Anforderungen getestet. Dabei wird auf Blackboxtesting gesetzt.

# Planen

## Funktionale Anforderungen

Die Soll-Analyse wird nun nochmals verfeinert und in funktionale- und nichtfunktionale Anforderungen gegliedert.

|  |  |
| --- | --- |
| Name | Beschreibung |
| Registrieren | Der Benutzer kann sich auf der Seite Registrieren. Für die Registration muss der Benutzer folgende Daten angeben:   * Name * Vorname * Benutzername * E-Mail * Telefonnummer * Passwort   Es gibt keine Bestätigungsemail bei der Registration |
| Login | Ein registrierter Nutzer kann sich mit seinen Daten einloggen. Für das Login braucht er seinen Benutzernamen und das Passwort. |
| Logout | Ein eingeloggter Nutzer kann sich ausloggen. Seine aktuelle Sitzung wird somit beendet und er hat nur noch Zugriff auf die Funktionen eines nicht angemeldeten Nutzer. |
| Inserat erstellen | Ein angemeldeter Nutzer kann ein Inserat erstellen. Dazu muss in einem Inserat folgende Daten angeben:   * Titel des Inserates * Beschreibung * Vorname des Erstellers * Die E-Mail des Erstellers * Die Telefonnummer des Erstellers * Ort des Angebotes * Art des Angebotes   Hat ein Nutzer ein Inserat erstellt, wird dieses in der Übersicht auf der Mainpage angezeigt. |
| Inserat Löschen | Ein angemeldeter Nutzer kann seine Inserate Löschen, falls diese nicht mehr aktuell sind. Hat ein Nutzer ein Inserat gelöscht, kann es von anderen Nutzern nicht mehr gesehen werden. |
| Inserat Übersicht | Auf der Startseite werden die Inserate in einer Liste angezeigt. Die Liste enthält nicht alle Informationen eines Inserates, sondern nur den Titel, die Art des Inserates und der Ort. Nach 10 Inseraten muss man über eine Navigation auf die nächste Seite gehen, um die folgenden Inserate zu sehen. Die Inserate werden nach der Erfassungszeit absteigend sortiert; die neusten sind zuoberst. |
| Inserat öffnen | Auf der Übersichtsseite kann man die Einzelnen Inserat öffnen um weitere Details zu erfahren. Die angezeigten Details entsprechen den selben Daten wie beim Inserat erstellen. |

### Berechtigungen

|  |  |  |  |
| --- | --- | --- | --- |
| Funktion | Nicht angemeldeter Nutzer | Angemeldeter Nutzer | Admin |
| Inserate Ansehen | ✔ | ✔ | ✔ |
| Inserate Öffnen | ✔ | ✔ | ✔ |
| Inserat erstellen | ✖ | ✔ | ✖ |
| Eigene Inserate Löschen | ✖ | ✔ | ✖ |
| Beliebiges Inserat Löschen | ✖ | ✖ | ✔ |

## Nicht funktionale Anforderungen

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

## Priorisieren

## Architektur

### Use Cases

|  |  |
| --- | --- |
| Use Case Nr. | 1 |
| Use Case Bezeichnung | User Registrierung |
| Akteur | Benutzer |
| Vorbedingung | Benutzer hat die Seite geöffnet und ist nicht eingeloggt |
| Use Case Beschreibung | Nutzer öffnet Registrationsseite  Nutzer gibt valide Daten ein  Nutzer wird Registriert |
| Resultat | Nutzer ist Eingeloggt auf der Startseite |

Tabelle : Use Case #1

|  |  |
| --- | --- |
| Use Case Nr. | 2 |
| Use Case Bezeichnung | User Login |
| Akteur | Benutzer |
| Vorbedingung | Benutzer hat die Seite geöffnet und hat schon einen Account |
| Use Case Beschreibung | Nutzer öffnet das Loginfenster  Nutzer gibt seine Daten ein  Nutzer wird Eingeloggt |
| Resultat | Nutzer ist Eingeloggt auf der Startseite |

Tabelle : Use Case #2

|  |  |
| --- | --- |
| Use Case Nr. | 3 |
| Use Case Bezeichnung | Logout |
| Akteur | Benutzer |
| Vorbedingung | Benutzer hat sich eingeloggt |
| Use Case Beschreibung | Nutzer loggt sich aus |
| Resultat | Der Nutzer ist abgemeldet. Er kann nicht mehr auf seine Daten zugreifen. |

Tabelle : Use Case #3

|  |  |
| --- | --- |
| Use Case Nr. | 4 |
| Use Case Bezeichnung | Inserat ansehen |
| Akteur | Benutzer |
| Vorbedingung | Benutzer befindet sich auf der Startseite |
| Use Case Beschreibung | Nutzer öffnet ein Inserat |
| Resultat | Es wird die Detailseite des Inserates angezeigt. |

Tabelle : Use Case #4

|  |  |
| --- | --- |
| Use Case Nr. | 5 |
| Use Case Bezeichnung | Inserat erstellen |
| Akteur | Benutzer |
| Vorbedingung | Benutzer hat sich eingeloggt |
| Use Case Beschreibung | Nutzer öffnet Seite zum neues Inserat erstellen  Nutzer gibt valide Daten ein  Inserat wird gespeichert |
| Resultat | Inserat ist auf der Datenbank gespeichert und kann von anderen Nutzern auf der Startseite angesehen werden. |

Tabelle : Use Case #5

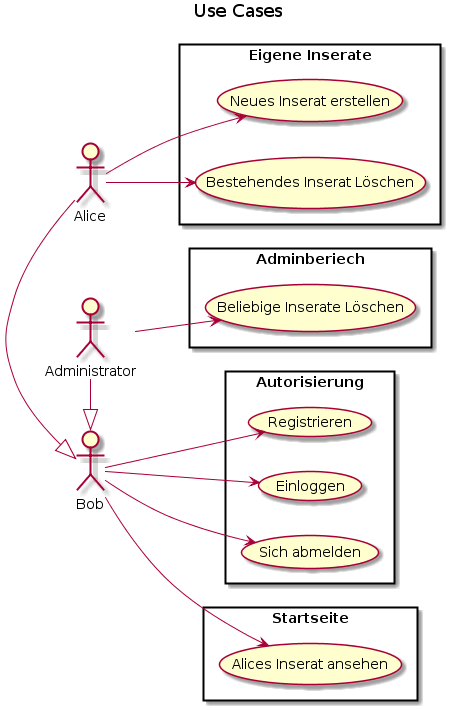
|  |  |
| --- | --- |
| Use Case Nr. | 6 |
| Use Case Bezeichnung | Inserat Löschen |
| Akteur | Benutzer |
| Vorbedingung | Benutzer hat sich eingeloggt, Nutzer hat schon ein Inserat erstellt |
| Use Case Beschreibung | Nutzer öffnet Seite seiner Inserate  Nutzer löscht ein Inserat |
| Resultat | Das Inserat wird nicht mehr angezeigt. |

Tabelle : Use Case #6

|  |  |
| --- | --- |
| Use Case Nr. | 7 |
| Use Case Bezeichnung | Inserat erstellen |
| Akteur | Administrator |
| Vorbedingung | Administrator hat sich eingeloggt, Es wurden schon Inserate erstellt |
| Use Case Beschreibung | Administrator öffnet Inserateseite  Administrator Löscht beliebiges Inserat |
| Resultat | Das Inserat wird nicht mehr angezeigt. |

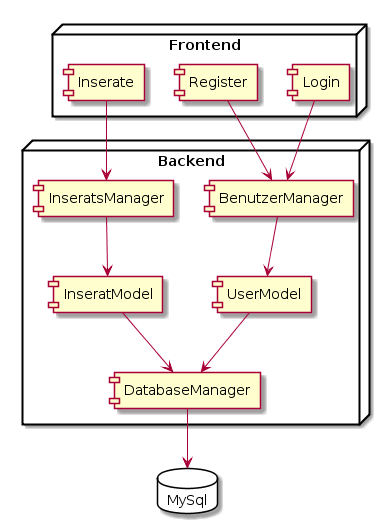
Tabelle : Use Case #7

### Use Case Diagramm



### Komponenten

Das Programm ist in drei Systemen aufgeteilt, welche miteinander kommunizieren müssen. Diese drei Systeme umfassen das Frontend, das Backend und die Datenbank. Um den Datenaustausch zwischen den einzelnen Dokumenten genauer zu erläutern habe ich folgendes Komponentendiagramm erstellt.



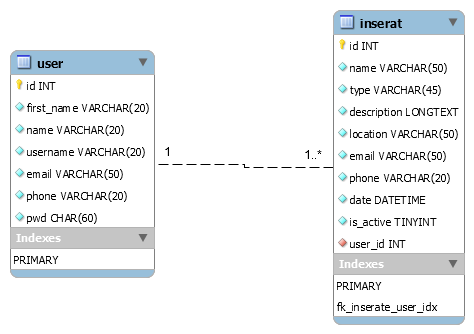
### Frontend

Das Frontend besitzt über mehrere Javascriptkomponenten, welche die Daten an den Server weiterleiten. Die Kommunikation zum Server erfolgt über das http und wird mit Ajax implementiert. Dies ermöglicht es, dass auf einer Seite verschiedene Controller auf dem Server unabhängig voneinander aufgerufen werden können. Zusätzlich kann das Frontend Serveranfragen machen, ohne dass die Seite danach neu geladen werden muss.

### Backend

Im Backend werden die Daten aufbereitet um sie entweder in der Datenbank zu speichern, oder um die Daten von der Datenbank auszulesen, um diese dann auf der Webseite anzeigen zu können. Das Backend wird in Controller und Models aufgeteilt, damit eine klare Trennung der Funktionen vorhanden ist. Die Datenbank wird über einen Datenbank Manager angesprochen. Dieser enthält die Instanz, um die Datenbank ansprechen zu können. Somit soll sichergestellt sein, dass die Datenbankanbindung nur dann steht, wenn sie auch wirklich gebraucht wird.

### Datenbank



|  |  |
| --- | --- |
| User |  |
| Beschreibung | Diese Tabelle speichert die Benutzerdaten ab. Diese entsprechen denjenigen, welche in den Anforderungen gewünscht sind. |
| Relationen | Ein Benutzer kann keine oder mehrere Inserate haben |
| Felder | |  |  |  |  | | --- | --- | --- | --- | | Feldname | Datentyp | Beschreibung |  | | Id | INT | Dieses Feld ist der Primärschlüssel der Tabelle. Es ist ein Integer, welcher automatisch Inkrementiert. | Not Null  Auto Increment  PK | | first\_name | Varchar(20) | Der Vorname des Benutzers | Not Null | | name | Varchar(20) | Der Nachname des Benutzers | Not Null | | username | Varchar(20) | Der Benutzername des Benutzers | Not Null | | email | Varchar(50) | Die E-Mail des Benutzers | Not Null | | phone | Varchar(20) | Die Telefonnummer des Benutzers | Not Null | | pwd | Char(60) | Das Passwort des Benutzers. Dieses wird von Bycrypt gehasht und abgespeichert. Daher wird auch kein zusätzliches Feld für einen Salt benötigt. | Not Null | |

|  |  |
| --- | --- |
| Inserat |  |
| Beschreibung |  |
| Relationen |  |
| Felder | |  |  |  |  | | --- | --- | --- | --- | | Feldname | Datentyp | Beschreibung |  | | Id | INT | Dieses Feld ist der Primärschlüssel der Tabelle. Es ist ein Integer, welcher automatisch Inkrementiert. | Not Null  Auto Increment  PK | | name | Varchar(50) | Dieses Feld enthält die Überschrift des Inserates | Not Null | | type | Varchar(45) | Diese Feild enthält den Inserattyp | Not Null | | description | Longtext | Diese Feld enthält die Inseratbeschreibung | Not Null | | location | Varchar(50) | Dieses Feld enthält den Ort, wo das im Inserat beworbene Produkt verfügbar ist | Not Null | | email | Varchar(50) | Dieses Fled enthält die Kontaktemail für das Inserat | Not Null | | phone | Varchar(20) | Dieses Feld enthält die Kontakttelefonnummer des Inserates | Not Null | | date | Datetime | Diese Feld enthält das Datum an dem das Inserat erstellt wurde | Not Null | | is\_active | Tinyint | Dieses Feld enthält den Wert, ob das Inserat noch verfügbar ist oder nicht. | Not Null | | user\_id | Int | Fremdschlüssel für den User, der dieses Inserat erstellt hat. | Not Null | |

## Testkonzept

### Testobjekt

### Testziele

### Teststrategie

### Testmethoden/Testmittel

### Organisation und Durchführung

### Testumgebung

Die Testumgebung setzt sich folgendermassen zusammen:

* Software:
* Windows 10 Professional
* XAMPP
* Hardware:
* Dell XPS 15

### Testdaten

### Vorbedingungen

### Testabdeckung

## Testfälle

### Registrierung

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Vorbedingungen | Testschritte | Erwartetes Ergebnis |
| 1.1 | Startseite ist Geöffnet und der Benutzer ist noch nicht eingeloggt | |  |  | | --- | --- | | 1. | Registrierungsknopf drücken | | Oben rechts erscheint das Registrierungsformular. Der Rest der Seite beliebt gleich |
| 1.2 | Registrierungsformular ist geöffnet, User ist noch nicht Angemeldet oder Registriert. | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1. | Folgende Daten eingeben:   |  |  | | --- | --- | | **Name:** | Hans | | **Vornamen:** | Zimmer | | **Benutzernamen:** | Hansli | | **E-Mail:** | haz@hans.ch | | **Telefonnummer:** | 076 666 66 66 | | **Passwort:** | Schlachtlauch | | **Passwort**  **Wiederholen:** | Schlachtsellerie | | | 2. |  | | Registrierung Fehlgeschlagen. Passwörter werden gelöscht und rot umrahmt |
| 1.3 | Registrierungsformular ist geöffnet, User ist noch nicht Angemeldet oder Registriert. | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1. | Folgende Daten eingeben:   |  |  | | --- | --- | | **Name:** | Hans | | **Vornamen:** | Zimmer | | **Benutzernamen:** | Hansli | | **E-Mail:** | haz@hans.ch | | **Telefonnummer:** | 076 666 66 66 | | **Passwort:** | Schlachtlauch | | **Passwort**  **Wiederholen:** | Schlachtlauch | | | 2. |  | | Registrierung ist erfolgreich Hans ist eingeloggt |
| 1.4 | Registrierungsformular ist geöffnet, User ist noch nicht Angemeldet oder Registriert. Hansli hat sich schon registriert | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1. | Folgende Daten eingeben:   |  |  | | --- | --- | | **Name:** | Hans | | **Vornamen:** | Zimmer | | **Benutzernamen:** | Hansli | | **E-Mail:** | haz@hans.ch | | **Telefonnummer:** | 076 666 66 66 | | **Passwort:** | Schlachtzwiebel | | **Passwort**  **Wiederholen:** | Schlachtzwiebel | | | 2. |  | | Registrierung ist nicht erfolgreich. Es besteht schon ein User mit demselben Namen. Das Feld Username wird rot umrahmt |

### Login

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Vorbedingungen | Testschritte | Erwartetes Ergebnis |
| 2.1 | Startseite ist Geöffnet | |  |  | | --- | --- | | 1. | Loginknopf drücken | | Oben rechts erscheint das Loginsformular. Der Rest der Seite beliebt gleich |
| 2.2 | Registrierungsformular ist geöffnet, Hansli ist registriert, Hansli ist noch nicht angemeldet | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 1. | Folgende Daten eingeben:   |  |  | | --- | --- | | **Benutzername:** | Hansli | | **Passwort:** | Schlachtlauch | | | 2. |  | | Login erfolgreich, Hansli ist eingelogt |
| 2.3 | Registrierungsformular ist geöffnet, Hansli ist registriert, Hansli ist noch nicht angemeldet | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 1. | Folgende Daten eingeben:   |  |  | | --- | --- | | **Benutzername:** | Peterli | | **Passwort:** | Schlachtlauch | | | 2. |  | | Login nicht erfolgreich, Passwort wird aus dem Passwortfeld gelöscht |
| 2.4 | Registrierungsformular ist geöffnet, User ist noch nicht Angemeldet oder Registriert. Hansli hat sich schon registriert | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1. | Folgende Daten eingeben:   |  |  | | --- | --- | | **Name:** | Hans | | **Vornamen:** | Zimmer | | **Benutzernamen:** | Hansli | | **E-Mail:** | haz@hans.ch | | **Telefonnummer:** | 076 666 66 66 | | **Passwort:** | Schlachtzwiebel | | **Passwort**  **Wiederholen:** | Schlachtzwiebel | | | 2. |  | | Registrierung ist nicht erfolgreich. Es besteht schon ein User mit demselben Namen. Das Feld Username wird rot umrahmt |

# Entscheiden

# Realisieren

## Verzeichnisstruktur

## Source Code

## HTML Filet

|  |  |
| --- | --- |
| Dateiname |  |
| Verzeichnis |  |
| Funktion |  |
| GUI Screenshot |  |
| Detaillierte Beschrei-bung |  |

Tabelle : HTML Template für SZP

## CSS Filet

|  |  |
| --- | --- |
| Dateiname |  |
| Verzeichnis |  |
| Funktion |  |
| Source Code Ausschnitt |  |
| Detaillierte Beschreibung |  |

Tabelle : CSS Design SZP

## JavaScript Filet

|  |  |
| --- | --- |
| Dateiname |  |
| Verzeichnis |  |
| Funktion |  |
| Globale Variablen/  Begründung |  |
| Detaillierte Beschreibung |  |

Tabelle : JavaScript Datei für SZP

### Funktion: Objekte aus Persistenz auslesen & abfüllen

|  |  |
| --- | --- |
| Datei |  |
| Funktionsname |  |
| Parameter |  |
| Rückgabewerte |  |
| Source Code Ausschnitt (Schlüsselstelle) |  |
| Beschreibung |  |

Tabelle : Funktion setArrays

### Errorhandling

|  |  |  |  |
| --- | --- | --- | --- |
| Fehlerverursacher | Fehler | Abgefangen durch | Implementierungsort |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Tabelle : Errorhandling

## Installationsanweisung

# Kontrollieren

## Testübersicht

|  |  |
| --- | --- |
| Kategorie | Ergebnis Testfälle |
| Erfolgreich | 35 |
| Teilweise Erfolgreich | 0 |
| Nicht erfolgreich | 0 |

Tabelle : Ergebnis Testfälle

## Testfazit

## Testprotokoll

### Testgruppe: Aufbau

| **Testfall ID** | **Testname** | **Aktionen/Parameter** | **Effektives Ergebnis** | **Status Ergebnis** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Tabelle : Testprotokoll Testgruppe Aufbau

# Auswerten

## Erfüllung Anforderungen

|  |  |
| --- | --- |
| **Funktionale Anforderungen** | |
| Anforderung | Status |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **Nicht funktionale Anforderungen** | |
| Anforderung | Status |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Tabelle : Erfüllung Anforderungen

Es wurden alle gestellten funktionalen als auch nichtfunktionalen Anforderungen lückenlos umgesetzt und erfüllt.

## Stand der Applikation

## Reflexion

## Fazit

.

# Abbildungsverzeichnis

[Abbildung 17: Ablaufdiagramm SZP starten 23](file:///C:\Users\Alain\Dropbox\Dokumente\ÜK\IPA-Doku_vorlage.docx#_Toc497231154)

[Abbildung 18: Architekturdiagramm 23](file:///C:\Users\Alain\Dropbox\Dokumente\ÜK\IPA-Doku_vorlage.docx#_Toc497231155)

# Tabellenverzeichnis

[Tabelle 1: Rollenbeschrieb 5](#_Toc497231131)

[Tabelle 3: Projektmeilensteine 9](#_Toc497231132)

[Tabelle 4: Arbeitsprotokoll 21.03.2017 10](#_Toc497231133)

[Tabelle 5: Arbeitsprotokoll 22.03.2017 11](#_Toc497231134)

[Tabelle 6: Arbeitsprotokoll 23.03.2017 12](#_Toc497231135)

[Tabelle 7: Arbeitsprotokoll 24.03.2017 13](#_Toc497231136)

[Tabelle 8: Arbeitsprotokoll 28.03.2017 14](#_Toc497231137)

[Tabelle 22: Umsetzung funktionale Anforderungen 21](#_Toc497231138)

[Tabelle 23: Umsetzung nichtfunktionale Anforderungen 21](#_Toc497231139)

[Tabelle 24: Legende Prioritätenliste 21](#_Toc497231140)

[Tabelle 25: Prioritätenliste 22](#_Toc497231141)

[Tabelle 26: Use Case #1 23](#_Toc497231142)

[Tabelle 44: Legende Punktzahl 27](#_Toc497231143)

[Tabelle 45: Auswahlkriterien Raster (grafisch) 27](#_Toc497231144)

[Tabelle 46: Nutzwertanalyse Raster (grafisch) 28](#_Toc497231145)

[Tabelle 61: HTML Template für SZP 29](#_Toc497231146)

[Tabelle 62: CSS Design SZP 30](#_Toc497231147)

[Tabelle 63: JavaScript Datei für SZP 31](#_Toc497231148)

[Tabelle 64: Funktion setArrays 31](#_Toc497231149)

[Tabelle 79: Errorhandling 32](#_Toc497231150)

[Tabelle 82: Ergebnis Testfälle 33](#_Toc497231151)

[Tabelle 83: Testprotokoll Testgruppe Aufbau 34](#_Toc497231152)

[Tabelle 90: Erfüllung Anforderungen 35](#_Toc497231153)

# Literaturverzeichnis

Das Literaturverzeichnis beinhaltet alle Webseiten, welche während der Realisierung zur Hilfe verwendet wurden.

|  |  |
| --- | --- |
| Was | Quelle |
|  |  |
|  |  |

# Quellenverzeichnis

Das Quellenverzeichnis beinhaltet alle Dokumente & Webseiten, von welchen Texte/Bilder/Vorlagen/Programmcode übernommen und/oder angepasst wurden.

|  |  |
| --- | --- |
| Was | Quelle |
|  |  |
|  |  |
|  |  |
|  |  |

# Glossar

|  |  |
| --- | --- |
| Abkürzung | Definition |
|  |  |
|  |  |
|  |  |
|  |  |