

Systemdokumentation

PR Software Engineering

Gruppe 3

259035 (2018S)

 Template

2017-05-03

**Über arc42**

arc42, das Template zur Dokumentation von Software- und Systemarchitekturen.

Erstellt von Dr. Gernot Starke, Dr. Peter Hruschka und Mitwirkenden.

Template Revision: 7.0 DE (asciidoc-based), January 2017

© We acknowledge that this document uses material from the arc 42 architecture template, <http://www.arc42.de>. Created by Dr. Peter Hruschka & Dr. Gernot Starke.

Inhaltsverzeichnis

[1. Einleitung 4](#_Toc514874033)

[1.1. Aufgabenstellung 4](#_Toc514874034)

[1.1.1. Inhalt 4](#_Toc514874035)

[1.1.2. Motivation 4](#_Toc514874036)

[1.2. Systemanforderungen 4](#_Toc514874037)

[1.3. Qualitätsziele 5](#_Toc514874038)

[1.4. Stakeholder 5](#_Toc514874039)

[2. Randbedingungen 6](#_Toc514874040)

[3. Kontextabgrenzung 7](#_Toc514874041)

[3.1. Fachlicher Kontext 7](#_Toc514874042)

[3.2. Technischer Kontext 7](#_Toc514874043)

[4. Lösungsstrategie 7](#_Toc514874044)

[5. Bausteinsicht 7](#_Toc514874045)

[5.1. Whitebox Gesamtsystem 7](#_Toc514874046)

[6. Laufzeitsicht 7](#_Toc514874047)

[7. Verteilungssicht 7](#_Toc514874048)

[8. Querschnittliche Konzepte 7](#_Toc514874049)

[9. Entwurfsentscheidungen 7](#_Toc514874050)

[10. Qualitätsanforderungen 7](#_Toc514874051)

[11. Risiken und technische Schulden 7](#_Toc514874052)

[12. Glossar 8](#_Toc514874053)

# Einleitung

Diese, in deutscher Sprache verfasste, Systemdokumentation basiert auf jener von Gruppe 3, im weiteren Verlauf der Dokumentation als „Team“ bezeichnet, für „PR Software Engineering“, erstellten App. Dabei handelt es sich um ein „Task Management und Time Tracker“ Tool, im weiteren Verlauf der Dokumentation als „TimeTrackerApp“ bezeichnet.

## Aufgabenstellung

### Inhalt

Folgende Funktionalitäten (durch die LVA Leitung vorgegeben) sollen zur Verfügung gestellt werden:

* Erfassen von Projekten, Aufgabenbereichen und Aufgaben
* Aufgabenbereiche als Swimlanes mit Aufgaben darstellen
* Erfassen von Sollzeiten mit Fortschrittsanzeige
* Zuordnung von Aufgaben zu Personen
* Erfassen der Zeit zu bestimmten Aktivitäten, Zuordnung zu Aufgaben
* Definition eines Rundungsintervalls
* Erfassen strukturierter Notizen zu jedem erfassten Eintrag
* Nachträgliches Ändern von Datum, Zeit und Dauer
* Status zu Aufgaben anzeigen
* Reporting für ein gesamtes Projekt, für unterschiedliche Aufgabenbereiche, Personen, etc.
* Synchronisation über mehrere Rechner
* Export (.csv, JSON)

### Motivation

Konkret wird also eine App erzeugt, welche das Management von Projekten, speziell Aufgaben, erleichtern soll. Dazu stellt diese Funktionen zum Erstellen von Projekten, Aufgabenbereichen und Aufgaben bereit. Außerdem ist es möglich, Zeitmanagement für die einzelnen Aufgaben, unter Berücksichtigung beteiligter Personen, zu betreiben. Die App vereint all diese Funktionen in einer Software, welche einfach zu bedienen sein soll, und ein effizientes Arbeiten damit ermöglicht.

Wir entwickeln sie mit dem Hintergedanken, sie selbst alsbald für unser Projekt nutzen zu können, und somit die Zeiterfassung als Excel Dokument ablösen zu können.

## Systemanforderungen

Da das Programm mithilfe von Java entwickelt wurde, lässt sich diese auf einer Vielzahl an System ausführen, sofern Java von jenem unterstützt wird. Die minimalen Systemanforderungen setzen sich somit wie folgt zusammen:

Betriebssystem: Microsoft Windows 7 oder höher

Java 8 oder höher

Prozessor: Intel Core 2 Duo, 1GHz (oder vergleichbar)

Arbeitsspeicher: 1GB oder mehr

Festplatte: 500MB freier Festplattenspeicher

Internetverbindung: 5Mbit/s oder schneller

## Qualitätsziele

Nachfolgend werden die Qualitätsziele in tabellarischer Ansicht angeführt. Diese wurden zum Teil von der LVA Leitung, durch die Beurteilungskriterien, indirekt vorgegeben. Die Reihenfolge spiegelt die Relevanz der Ziele, welche durch das Team festgelegt wurde, wieder.

|  |  |
| --- | --- |
| **Qualitätsziel** | **Beschreibung** |
| Effizienz | Die App arbeitet effizient, in erster Linien betrifft dies den Programmcode. Zusätzlich soll eine effiziente Benutzung der App selbst möglich sein. |
| Stabilität | Die App arbeitet stabil, sprich sie liefert die Ergebnisse, welche der Benutzer, in Form von Erwartungen an sie stellt. Fehlerhafte Ausgaben sollten soweit eliminiert werden, sodass die Benutzung durch den Anwender reibungslos erfolgen kann. |
| Programmcode | Ist Teil des Qualitätsziels „Effizienz“. Der Code sollte schlank sein, lösungsorientiert und effizient arbeiten. Redundanzen im Code sollen weitestgehend, soweit möglich, entfernt werden. |
| Benutzeroberfläche | Die Benutzeroberfläche sollte selbsterklärend, und ohne Bedienungsanleitung möglich sein. Ebenso soll diese schlank, nicht mit Menüs überladen, und einfach strukturiert sein. |
| Sicherheit | Die App lässt sich ohne Sicherheitsrisiken verwenden. Verbindungen, welche zu Datenbanken hergestellt werden, basieren auf einer SSL-Verbindung, welche eine sichere Datenübertragung gewährleisten soll. |

## Stakeholder

Nachfolgend werden die Stakeholder, welche Einfluss auf das Projekt und die Entwicklung dessen haben, angeführt.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Kontakt** | **Funktion** | **Erwartungshaltung** |
|  |  | LVA Leiter |  |
|  |  | LVA Leiter |  |
| Aistleithner Andrea | aistleithner.andrea97@gmx.at | Projektmitarbeiter | * sinnvolle Testung der App, um Fehler frühzeitig zu erkennen und um gegensteuern zu können * Erzeugung einer fehlerfreien App |
| Dusanic  Maja | majadusanic@gmail.com | Projektmitarbeiter | * sinnvolle Nutzung des SCR für die Entwicklung der Applikation und zum Management des Programmcodes |
| Huber  Christoph | christoph.huber91@gmail.com | Projektmitarbeiter | * effiziente Planung des Projekts, sowie der Entwicklung der App * effizientes Issuemanagement, welches die Fehler aus den Tests aufschlüsselt |
| Teuchtmann Alexander | a.teuchtmann@alteutech.at | Projektmitarbeiter | * sinnvolle Dokumentation des Entwicklungsfortschritts und der Anforderungen * verständliche Dokumentation des Systems (Systemdokumentation) |
| Tomic  Milos | tomicmilos94@yahoo.at | Projektmitarbeiter | * die Erzeugung von qualitativ hochwertigem Programmcode, welcher den Coderichtlinien entspricht |

# Randbedingungen

Grundsätzlich sind keine Randbedingungen, bis auf die geforderten Funktionalitäten, gegeben. Durch die Zusammensetzung des Teams ergeben sich jedoch bestimmte Einschränkungen. Ebenso hat das Team selbst Bedingungen festgelegt.

Organisatorische Einschränkung

|  |  |  |
| --- | --- | --- |
| **Einschränkung** | **Erklärung** | **Quelle** |
| Namenskonventionen | Die Namenskonventionen schränken den Gebrauch von Namen bei der Entwicklung ein. Sie stellen sicher, dass einheitliche Namen verwendet werden. | Code Richtlinien  Seite 15-16 |
| Coderichtlinien | Die Coderichtlinien schränken generell den Programmierstil ein. Es werden Richtlinien aufgestellt, welche die Qualität des Codes sicherstellen sollen. | Code Richtlinien |
| zeitliche Frist | Durch die LVA Leitung wurden zeitliche Fristen festgelegt (Meilensteine), welche einzuhalten sind. | Vorbesprechung |

Personelle Einschränkungen

|  |  |
| --- | --- |
| **Einschränkung** | **Erklärung** |
| Programmiersprache | Durch die Einschränkung, dass der Großteil des Teams nur die Programmiersprache Java beherrscht, wurde diese auch für das Projekt als Grundlage gewählt. Gleiches gilt für die Benutzung einer (my)SQL Datenbank für die Datenspeicherung. |

# Kontextabgrenzung

## Fachlicher Kontext

Nachfolgend sind alle Kommunikationspartner bezüglich der Benutzung der TimeTrackerApp aufgelistet. Ebenso finden sich hier die Ein- und Ausgabedaten, sowie die genutzten Schnittstellen und Kommunikationskanäle, wieder.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Partner** | **Eingabe** | **Ausgabe** | **Schnittstellen** | **Kanal** |
| Benutzer1 |  |  |  |  |
| Arbeitsplatz2 |  |  |  |  |
| Instanz3 |  |  |  |  |
| Datenbank4 |  |  |  |  |

1. Benutzer:

2. Arbeitsplatz:

3. Instanz:

4. Datenbank:

## Technischer Kontext

// technische Schnittstellten

Abbildung 1 Technische Schnittstellen

|  |  |  |
| --- | --- | --- |
| Kanal | Eingabe | Ausgabe |
|  |  |  |
|  |  |  |

# Lösungsstrategie

Dieses Kapitel bietet Einsicht in die grundlegenden Entscheidungen und Lösungsansätze, welche die TimeTrackerApp prägen. Nachfolgend werden diese kurz erklärt, genauere Ausführungen sind in den Folgekapiteln zu finden.

**Technologieentscheidungen**

**Entwurfs- und Architekturmuster**

**Erreichung der Qualitätsanforderungen**

**organisatorische Entscheidungen**

# Bausteinsicht

// Grundriss der Software

Abbildung 2 Systemzerlegung in Bausteine

## Whitebox Gesamtsystem

// Zerlegung des Gesamtsystems in Bausteine (z.B.: Daten, Views, …)

# Laufzeitsicht

// Arbeit der Bausteine zur Laufzeit (Szenarien)

Abbildung 3 Szenario 1

Abbildung 4 Szenario 2

# Verteilungssicht

Die Verteilung der einzelnen Bausteine ist im Falle der TimeTrackerApp auf zwei unterschiedliche Systeme beschränkt. Die Ausführungseinheit der TimeTrackerApp befindet sich auf einem oder mehreren Arbeitsplätzen, welche unterschiedliche Endgeräte darstellen können (Laptop, Desktop). Das zweite System stellt die konsistente Datenspeicherung dar, sie erfolgt in einer extern gehosteten Datenbank auf einem Drittsystem, auf welches von den Arbeitsplätzen zugegriffen wird.

Abbildung 5 Verteilung der Infrastruktur

## Konkrete Verteilung

**Infrastruktur 1**

Im Verständnis dieser Arbeit haben wir die Infrastruktur 1 wie folgt definiert. Sie besteht im Grunde genommen aus der Ausführungseinheit der TimeTrackerApp. Dies kann ein mobiles Endgerät (zum Beispiel ein Laptop) oder auch ein Desktopcomputer sein. Da die Infrastruktur dieser Geräte im weitesten Sinne gleich oder ähnlich sind (logischer Aufbau), wurden diese in einer Infrastruktur zusammengefasst.

**Infrastruktur 2**

Diese stellt das System, auf welchem die Datenbank gespeichert ist, dar. Ausgehend von der Infrastruktur 1, werden hier Daten konsistent gespeichert, und bei Bedarf abgerufen. Die Datenbank wird auf einem externen Server gehostet, welcher administrativ vom Dienstleistungsanbieter zur Gänze verwaltet wird. Die Installation, Verwaltung und Pflege der Datenbank selbst wird vom Team ausgeführt.

## Begründung der Verteilung

**Infrastruktur 1**

Um unabhängig von jeweiligen Endgerät zu sein, wurde auf ein universell einsetzbares Programmsystem (Java) gesetzt, welches ohne aufwändige Vorkehrungen auf fast jedem zuvor beschriebenen Benutzergerät zum Einsatz kommen kann. Somit ist die Benutzung der TimeTrackerApp nahezu ortsungebunden.

**Infrastruktur 2**

Um eine dauerhafte, und ebenso ortsunabhängige Datenspeicherung und -abrufung möglich zu machen, wurde dafür eine extern gehostete Datenbank verwendet. Diese ist mit 99%iger (laut Dienstleister) Wahrscheinlichkeit immer verfügbar und im Falle des Datenverlusts können Backups problemlos eingespielt werden.

# Querschnittliche Konzepte

// übergreifende Regelungen, Prinzipien (Richtlinien)

# Entwurfsentscheidungen

// siehe #4, Beschreibung der Auswahlkriterien für eine Strategie

# Qualitätsanforderungen

// Erweiterung von #1.3

# Risiken und technische Schulden

// Liste von Risiken und Bewertung derer

# Glossar

Dieses Glossar umfasst alle Begriffe und Abkürzungen, welche, im Zusammenhang mit in diesem Praktikum entstandener Systemdokumentation, verwendet wurden. Es soll ein einheitliches Verständnis für alle Begrifflichkeiten schaffen, und Redundanzen, in der Verwendung von mehreren Begriffen für ein Element, eliminieren.

Projektinterne Bezeichnungen

|  |  |
| --- | --- |
| **Begriff** | **Definition** |
| Projekt | Name für die in diesem Praktikum durchgeführten Aktionen |
| Team | Name für die „Gruppe 3“ des Praktikums |
| TimeTrackerApp | Name für die in diesem Praktikum erstellten Applikation |

Abkürzungsverzeichnis

|  |  |
| --- | --- |
| **Abkürzung** | **Definition** |
| .csv | Comma-separated values (Dateiformat) |
| App | Applikation |
| SCR | Source Code Repository |

Begriffserklärungen

|  |  |
| --- | --- |
| **Begriff** | **Definition** |
| Java | objektorientierte Programmiersprache |
| mySQL | Datenbankverwaltungssystem |
| SQL | Datenbanksprache für relationale Datenbanken |
| JSON | JavaScript Object Notation (Dateiformat) |