Abschlussprüfung Winter 2018

**Fachinformatiker Anwendungsentwicklung**

Entwicklung eines Webtools ­­zur digitalen Erfassung von Rufbereitschaftszeiten für das XIS-System der Deutschen Börse AG

**Prüfungsbewerber:**

Herr Tim Rumrich

E-Mail: [tim.rumrich@gmx.de](mailto:tim.rumrich@gmx.de)

Telefon: +49 6109 62868

**Ausbildungsbetrieb:**

Deutsche Börse AG

Mergenthalerallee 61

65760 Eschborn

**Projektbetreuer:**

Frau Regina Wohak

E-Mail: [regina.wohak@deutsche-boerse.com](mailto:regina.wohak@deutsche-boerse.com)

Telefon: +49 69211 17428

Abgabetermin: 26.11.2018

# Inhaltsverzeichnis

[Inhaltsverzeichnis I](#_Toc531035479)

[Tabellenverzeichnis III](#_Toc531035480)

[Abbildungsverzeichnis IV](#_Toc531035481)

[Glossar V](#_Toc531035482)

[1 Einleitung 1](#_Toc531035483)

[1.1 Projektbeschreibung 1](#_Toc531035484)

[1.2 Projektumfeld 1](#_Toc531035485)

[1.3 Projektziel 2](#_Toc531035486)

[1.4 Projektbegründung 2](#_Toc531035487)

[2 Projektplanung 3](#_Toc531035488)

[2.1 Projektphasen 3](#_Toc531035489)

[2.2 Ressourcenplanung 3](#_Toc531035490)

[2.3 Entwicklungsprozess 3](#_Toc531035491)

[3 Analysephase 4](#_Toc531035492)

[3.1 Ist-Analyse 4](#_Toc531035493)

[3.2 Wirtschaftlichkeitsanalyse 4](#_Toc531035494)

[3.2.1 Make-Or-Buy Entscheidung 4](#_Toc531035495)

[3.2.2 Projektkosten 5](#_Toc531035496)

[3.2.3 Amortisationsdauer 5](#_Toc531035497)

[3.3 Nutzwertanalyse 6](#_Toc531035498)

[3.4 Lastenheft/Fachkonzept 7](#_Toc531035499)

[4 Entwurfsphase 8](#_Toc531035500)

[4.1 Zielplattform 8](#_Toc531035501)

[4.2 Architekturdesign 8](#_Toc531035502)

[4.3 Entwurf der Benutzeroberfläche 9](#_Toc531035503)

[4.4 Geschäftslogik 9](#_Toc531035504)

[4.5 Maßnahmen zur Qualitätssicherung 10](#_Toc531035505)

[4.6 Pflichtenheft/Datenverarbeitungskonzept 11](#_Toc531035506)

[5 Implementierungsphase 11](#_Toc531035507)

[5.1 Implementierung der Benutzeroberfläche 11](#_Toc531035508)

[5.2 Implementierung der Backendlogik 12](#_Toc531035509)

[5.3 Algorithmen zur Verteilung 12](#_Toc531035510)

[5.3.1 Zufallsverteilung 13](#_Toc531035511)

[5.3.2 Lineare Verteilung 13](#_Toc531035512)

[5.3.3 Gemischte Verteilung 13](#_Toc531035513)

[6 Abnahmephase 14](#_Toc531035514)

[7 Fazit 14](#_Toc531035515)

[7.1 Soll-/Ist-Vergleich 14](#_Toc531035516)

[7.2 Lessons Learned 14](#_Toc531035517)

[7.3 Ausblick 15](#_Toc531035518)

[Anhang 16](#_Toc531035519)

# Tabellenverzeichnis

[Table 1: Grobe Zeitplanung 3](#_Toc531031406)

[Table 2: Kostentabelle 5](#_Toc531031407)

[Table 3: Vorher- Nachher Aufwandsvergleich 6](#_Toc531031408)

[Table 4: Nutzwerttabelle Projekt 7](#_Toc531031409)

[Table 5: Algorithmenvergleich 10](#_Toc531031410)

[Table 6: Detaillierte Zeitplanung 18](#_Toc531031411)

[Table 7: Vergleich Zeitplanung 19](#_Toc531031412)

# Abbildungsverzeichnis

[Figure 1: Konzeptionelles Klassendiagramm 17](file:///C:\Users\bj093\Desktop\Doku_3.docx#_Toc531029747)

[Figure 2: Mock-Up Entwurf 21](#_Toc531029748)

[Figure 3: Screenshot fertige Tabelle ohne Styling 22](#_Toc531029749)

[Figure 4: Screenshot Debug Konsole 22](#_Toc531029750)

[Figure 5: Screenshot Download Dialog 22](#_Toc531029751)

[Figure 6: Screenshot Tabelle mit fertigem Styling 23](#_Toc531029752)

[Figure 7: Screenshot Suchfunktion auf Tabelle 23](#_Toc531029753)

[Figure 8: Screenshot Suchergebnis 23](#_Toc531029754)

[Figure 9: Lineare Verteilung Struktogramm 24](#_Toc531029755)

[Figure 10: Zufällige Verteilung Struktogramm 25](#_Toc531029756)

# Glossar

**API** (Application Programming Interface)

Eine Programmierschnittstelle einer Software ist ein Programmteil, der zur Anbindung anderer Programme an das System zur Verfügung gestellt wird.

**DBAG** (Deutsche Börse AG)

Die Deutsche Börse AG ist ein deutsches Finanzunternehmen das Bereitstellung und Vertrieb von Handelsplattformen für den Wertpapierhandel als Kerngeschäft hat.

**Config File** (Configuration File /Konfigurationsdatei)

Eine Textdatei die Konfigurationsdaten für Software bereitstellt.

**CSV** (Comma Separated Values)

Ein Dateiformat für Textdateien zur Speicherung oder zum Austausch einfach strukturierter Daten. In CSV-Dateien können beispielsweise Tabellen abgebildet werden.

**GUI** (Graphical User Interface)

Eine grafische Benutzeroberfläche bildet die Schnittstelle für Benutzer zur Interaktion mit der Anwendungssoftware.

**HTML** (Hypertext Markup Language)

Eine textbasierte Auszeichnungssprache zur Strukturierung digitaler Elemente. HTML-Dokumente sind Grundlage des World-Wide-Web und können von Webbrowsern dargestellt werden.

**MVC** (Model View Controller)

Ein Architekturmuster zur Eingliederung von Software in die Kategorien Model(Daten/Logik), View(Ansicht/Oberfläche) und Controller(Interaktion zwischen Oberfläche und Logik), mit Ziel eines flexiblen Programmentwurfes um spätere Wiederverwendbarkeit oder Änderbarkeit zu erleichtern. Detaillierte Erklärung im Anhang.

**XIS** (Exchange Index System)

Das System der Index-Berechnung der DBAG.

**XML** (Extensible Markup Language)

Eine Auszeichnungssprache für den Austausch und die Darstellung von Daten.

# 1 Einleitung

Die nachfolgende Projektdokumentation beschreibt…

## Projektbeschreibung

## 1.2 Projektumfeld

## 1.3 Projektziel

## 1.4 Projektbegründung

# 2 Projektplanung

## 2.1 Projektphasen

Für die Umsetzung des Projektes standen 70 Stunden zur Verfügung. Zu Beginn des Projektes wurden diese auf verschiedene Phasen aufgeteilt, die während des Projektes durchlaufen werden.

|  |  |
| --- | --- |
| Grobe Zeitplanung der Phasen | |
| Projektphase | Zeitplanung |
| Analysephase | h |
| Entwurfsphase | h |
| Implementierungsphase | h |
| Abnahmephase | h |
| Dokumentation | h |
| **Gesamt** | **70 h** |

Table 1: Grobe Zeitplanung

## 2.2 Ressourcenplanung

In der Ressourcenübersicht sind alle Ressourcen aufgelistet, die für das Projekt verwendet wurden. Damit sind Hard- und Software sowie Personal gemeint. Bei der ausgewählten Software wurde gezielt Software verwendet, die bereits im bestehenden System verwendet wurde um den Gesamtcode homogen zu halten.

Im Anhang findet sich eine entsprechende Auflistung.

## 2.3 Entwicklungsprozess

Noch vor dem Beginn der Umsetzung musste sich für einen geeigneten Entwicklungsprozess entschieden werden, um die Vorgehensweise der Umsetzung vorab zu definieren. Für das Projekt wurde eine agile Entwicklung gewählt, da diese im Index-Services Team Standard ist und sich besonders eignet, um schnell auf Zusatzwünsche oder auftretende Probleme zu reagieren. Das wiederholte Durchlaufen der Projektphasen für einzelne Programmteile und die andauernde Kommunikation mit den Kunden sind einige der Charakteristika für einen agilen Entwicklungsprozess.

Durch kurze Entwicklungszyklen kann man sehr zeitnah erste Ergebnisse präsentieren oder Module testen und auf Probleme bei der Umsetzung schnell reagieren, was zu Zeitersparnis und einem besseren Ergebnis für die Kunden führen kann, da auch auf kurzfristige Änderungswünsche eingegangen werden kann.

# 3 Analysephase

## 3.1 Ist-Analyse

### 3.2.1 Make-Or-Buy Entscheidung

Bei dem Projekt handelt es sich um eine recht grundlegende Problemstellung, für die es bereits Tools gibt. Daher ist es vonnöten, eine gründliche Make-Or-Buy Analyse durchzuführen, da es möglich sein kann, dass es effektiv billiger ist eine Lösung einzukaufen, als selber ein Produkt zu entwickeln.

### 3.2.2 Projektkosten

Hier sollen die anfallenden Projektkosten errechnet werden. Dafür werden die anfallenden Personalkosten sowie die Ressourcenkosten ermittelt, die während der Projektumsetzung anfallen. Die genauen Personalkosten sind als Confidential zu behandeln, weswegen die Berechnung der Personalkosten anhand von Stundensätzen von 35€ für Mitarbeiter und 10€ für Auszubildende erfolgt.

|  |  |  |  |
| --- | --- | --- | --- |
| Kostentabelle | | | |
| Vorgang | Personal | Zeit | Kosten |
| Entwicklung | 1x Auszubildender | 70 h | 700 € |
| Review | 1x Mitarbeiter | 3 h | 105 € |
| Abnahme | 1x Mitarbeiter | 2 h | 70 € |
| **Gesamt** |  |  | **875 €** |

Table 2: Kostentabelle

## 3.4 Lastenheft/Fachkonzept

# 4 Entwurfsphase

## 4.1 Zielplattform

## 4.2 Architekturdesign

## 4.3 Entwurf der Benutzeroberfläche

## 4.4 Geschäftslogik

## 4.5 Maßnahmen zur Qualitätssicherung

## 4.6 Pflichtenheft/Datenverarbeitungskonzept

# 5 Implementierungsphase

## 5.1 Implementierung der Benutzeroberfläche

## 5.2 Implementierung der Backendlogik

# 6 Abnahmephase

Nach Fertigstellung des Projektes konnte die Applikation den Kunden vorgestellt werden. Aufgrund der agilen Entwicklung konnten bereits während der Implementierung Zwischenergebnisse vorgestellt werden, wodurch das Produkt nicht näher erläutert werden musste, da Oberflächen und Funktionsweisen bereits vertraut waren. Dadurch konnte die Einführung reibungslos durchgeführt werden.

# 7 Fazit

## 7.1 Soll-/Ist-Vergleich

## 7.2 Lessons Learned

## 7.3 Ausblick

# Anhang

[A.1 Ressourcenplanung 16](#_Toc531035520)

[A.2 Klassendiagramm (Konzept) 17](#_Toc531035521)

[17](#_Toc531035522)

[A.3 Tabellen zur Zeitplanung 18](#_Toc531035523)

[Detaillierte Zeitplanung 18](#_Toc531035524)

[Zeitvergleich 19](#_Toc531035525)

[A.5 Lastenheft (Auszug) 19](#_Toc531035526)

[A.6 Pflichtenheft (Auszug) 20](#_Toc531035527)

[A.7 Oberflächenentwürfe 21](#_Toc531035528)

[A.8 Screenshots der Anwendung 22](#_Toc531035529)

[A.9 Entwicklerdokumentation (Auszug) 24](#_Toc531035530)

[A.10 Struktogramme der Verteilungsalgorithmen 25](#_Toc531035531)

[A.11 Erklärung MVC 27](#_Toc531035532)

[Eidesstattliche Erklärung/Projektantrag 27](#_Toc531035533)

## A.1 Ressourcenplanung

**Hardware**

* Büroarbeitsplatz mit Developer PC (CentOS)

**Software**

* Vim Texteditor
* C++11 Compiler
* TNTNET
* CXXTOOLS

**Personal**

* Mitarbeiter der Abteilung Index Services

## A.2 Klassendiagramm (Konzept)

## 

Figure : Konzeptionelles Klassendiagramm

## A.3 Tabellen zur Zeitplanung

### Detaillierte Zeitplanung

|  |  |
| --- | --- |
| Detaillierte Zeitplanung | |
| Vorgang | Eingeplante Zeit |
| **Analyse** |  |
| →Durchführung Ist Analyse | 2 h |
| →Wirtschaftlichkeitsanalyse | 1 h |
| →Nutzwertanalyse | 1 h |
| →Erstellung Lastenheft | 3 h |
| **Entwurf** |  |
| →Soll Konzept | 2 h |
| →Klassendiagramme | 6 h |
| →Mock-Ups | 6 h |
| →Algorithmenlogik | 6 h |
| →Pflichtenheft | 3 h |
| **Implementierung** |  |
| →Frontend | 7 h |
| →Backend | 7 h |
| →Download Funktion | 2 h |
| →Aufsetzen von Config Files | 4 h |
| →Testen | 6 h |
| **Abnahme & Einführung** |  |
| **→**Bereitstellen des Tools | 2 h |
| **Dokumentation** |  |
| →Projektdokumentation | 10 h |
| →Benutzerhandbuch | 2 h |
| **Gesamtzeit** | **70 h** |

Table 6: Detaillierte Zeitplanung

### Zeitvergleich

|  |  |  |
| --- | --- | --- |
| Vergleich Zeitplanung / Umsetzung | | |
| Projektphase | Zeitplanung | Umsetzung |
| Analysephase | 6 h | 6 h |
| Entwurfsphase | 24 h | 26 h |
| Implementierungsphase | 26 h | 24 h |
| Abnahmephase | 2 h | 2 h |
| Dokumentation | 12 h | 12 h |
| **Gesamt** | **70 h** | **70 h** |

Table 7: Vergleich Zeitplanung

## A.5 Lastenheft (Auszug)

Im nachfolgenden Lastenheftauszug sind die Anforderungen an die fertige Applikation definiert.

**Die Applikation soll folgende Funktionalitäten bieten:**

* Die Rufbereitschaftsplanung soll vollständig automatisch eine fertige Tabelle ähnlich der bestehenden Tabellen generieren.
* Die Verteilung der Rufbereitschaften soll gleichmäßig auf alle User sein.
* Bei der Verteilung müssen vorgeschriebene Regeln beachtet werden.
* User sollen persönliche Einstellungen vornehmen können, an welchen Tagen sie zu Rufbereitschaften eingeteilt werden können und wann nicht.
* Regeln zur Verteilung sollen veränderbar und erweiterbar sein.
* Die fertige Tabelle soll auf einer Webseite einsehbar sein.
* Eine Download Funktion zum Herunterladen der fertigen Tabelle soll zur Verfügung gestellt werden.
* Gewünscht ist eine Berücksichtigung von Urlauben für die Rufbereitschaftsplanung

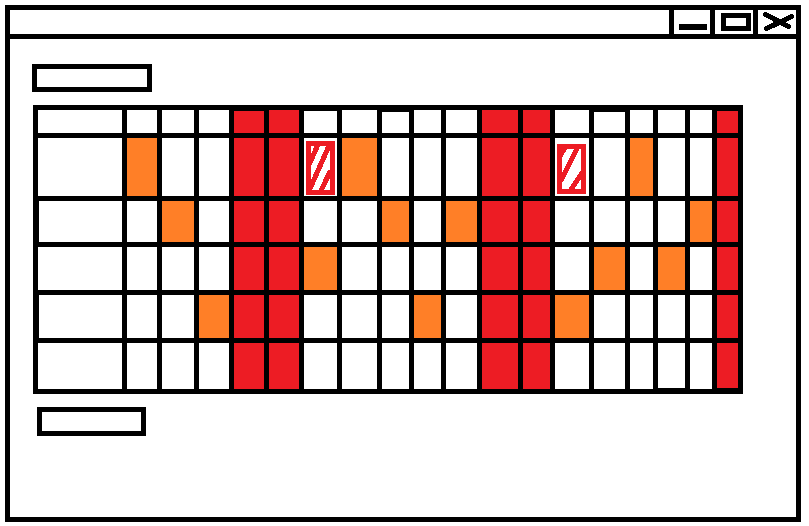
## A.6 Pflichtenheft (Auszug)

Im nachfolgenden Pflichtenheftauszug wird die Umsetzung der Anforderungen aus dem Lastenheft beschrieben.

**Umsetzung der Anforderungen:**

* Die Tabelle soll per Button auf der Oberfläche erstellt werden und dann in die Oberfläche geladen werden.
* Die Verteilung erfolgt über einen Algorithmus der die Häufigkeit der Rufbereitschaften unter den einzelnen Usern angleicht. Hierzu wird entweder eine Zufallsfunktion verwendet.
* Die Verteilung erfolgt unter Berücksichtigung von Regeln. Jede Eintragung im Kalender der Einzelpersonen bekommt dazu eine ID zugewiesen, die ausgelesen werden kann. Abhängig von der ausgelesenen ID wird entschieden, ob ein Mitarbeiter an einem einzelnen Tag Rufbereitschaften oder andere Arbeiten ausführen darf. Falls Regeln eine eindeutige Zuordnung von Rufbereitschaften verhindern, wird ein DummyUser verwendet, von welchem aus Rufbereitschaften manuell verteilt werden müssen.
* User sollen in einem Wochensetup einzelne Tage ausschließen können, an denen sie gesichert keine Rufbereitschaft haben.
* Regeln sind beliebig erweiterbar, es ist darauf zu achten, dass eine laufende Nummer für die einzelnen Regeln als Identifikation verwendet wird.
* Der Download der Tabelle soll als CSV-Datei erfolgen.
* Eine Berücksichtigung von Urlauben, welche im Sharepoint Kalender des Index-Teams eingetragen werden, ist aufgrund der von den Regulatoren vorgeschriebenen Isolation des XIS-Systems nicht möglich.

## A.7 Oberflächenentwürfe



3

8

7

2

1

4

6

5

9

Figure 2: Mock-Up Entwurf

1: Button zum Erstellen der Tabelle

2: Monatstabelle

3: Download Button

4: Geblockter Tag (Wochenende/Feiertag)

5: Geblockter Tag (Persönliches Setup)

6: Freier Tag

7: Username

8: Backup Dummy-User

9: Datumsfeld

## A.8 Screenshots der Anwendung

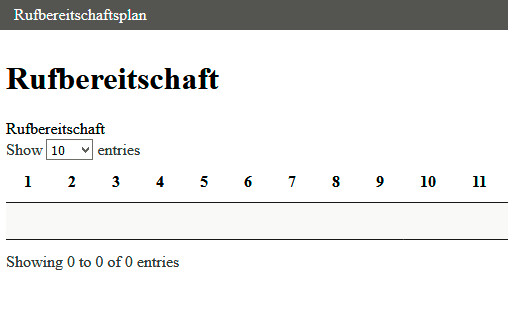


Figure 3: Screenshot Datatable ohne Input

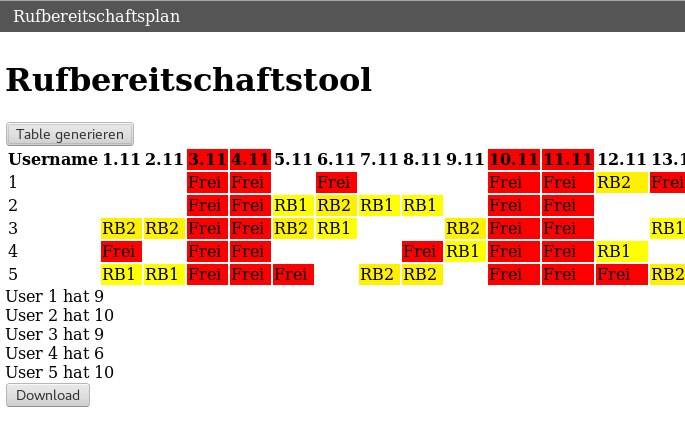


Figure 4: Screenshot fertige Tabelle ohne Styling

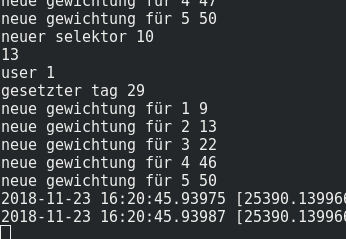


Figure 5: Screenshot Debug Konsole

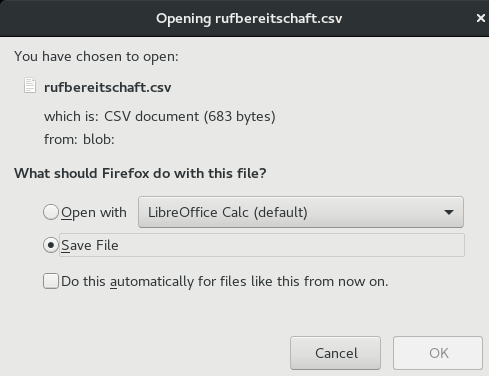


Figure 6: Screenshot Download Dialog

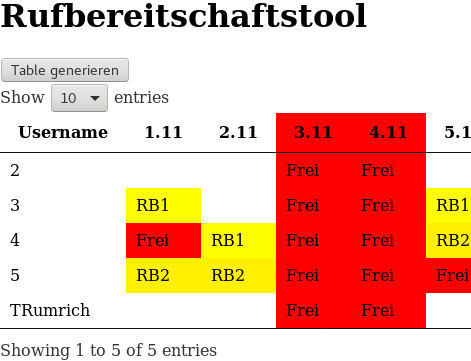


Figure 7: Screenshot Tabelle mit fertigem Styling

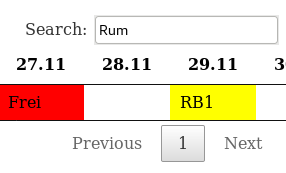


Figure 8: Screenshot Suchfunktion auf Tabelle

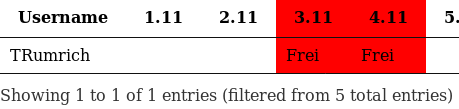


Figure 9: Screenshot Suchergebnis

## A.9 Entwicklerdokumentation (Auszug)

Hier soll eine knappe Entwicklerdokumentation über den Funktionalen Aufbau der Applikation bzw. der komplexen Module erfolgen.

Das gesamte Programm basiert stark auf der Auswertung von Vektoren. Beispielsweise die Monatstage bzw. Arbeitstage werden als Vektor gehalten und dann von verschiedenen Modulen ausgewertet. Es gibt hierzu eine Konvention, nach der verschiedene Einträge in diesen Vektoren unterschiedliche Bedeutungen haben. Diese lassen sich gesammelt in der Erstellung der Tabelle im **rufbereitschaftstable.ecpp** **Modul** einsehen, wo mit einer switch-case Funktion über jeden Eintrag in jedem PersonenObjekt iteriert wird und ein jeweils zugehöriges HTML-Element angefügt wird.

Die Erstellung der Monate bzw. der Struktur der Monate erfolgt über eine komplexe aber nötige Modulo Operation, mit der ermittelt wird welcher **„Shift“** vorliegt, also wie verschoben die Wochen von einer „Idealwoche“ sind, die Sonntags beginnt und Samstags endet. Die Errechnung der Monatstage erfolgt nun über eine Addition von Tag im Monat plus Shift, also z. B. 1+3=4, der erste Tag im Monat erhält den Wert 4 ist also 4 Tage vom letzten Sonntag entfernt, ist also ein Donnerstag.

Die Berechnung der neuen Gewichtung erfolgt aktuell, indem jedem nicht gezogenen Mitarbeiter die Gewichtung um 1 erhöht und die Gewichtung des gezogenen Mitarbeiters um **Anzahl Mitarbeiter -1** verringert.

## A.10 Struktogramme der Verteilungsalgorithmen

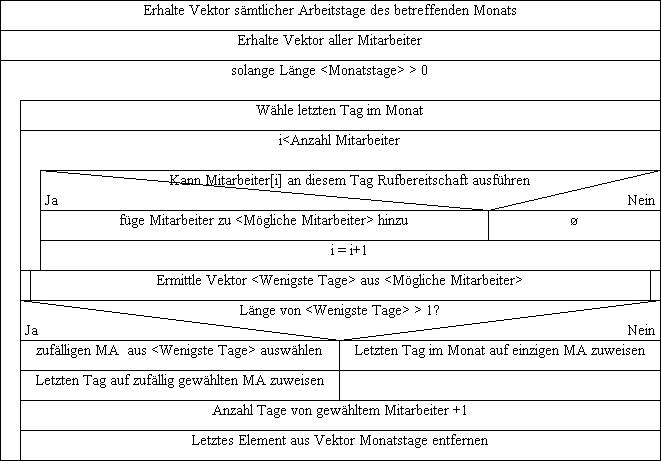


Figure 10: Lineare Verteilung Struktogramm

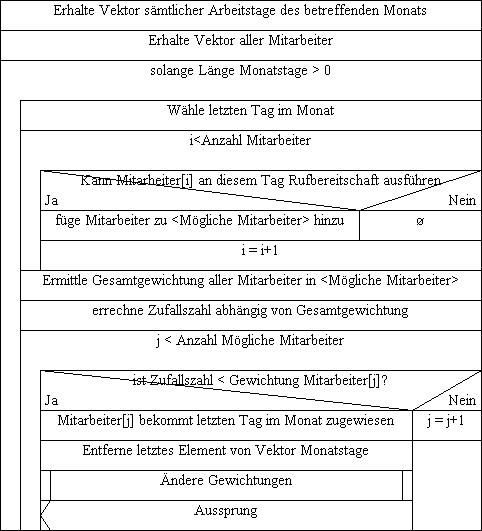


Figure 11: Zufällige Verteilung Struktogramm

## A.11 Erklärung MVC

Das Model-View-Control Modell beschreibt eine Architektur, in der jedes Modul einer der drei Gruppen Model, View und Control eindeutig zuordenbar ist. Die View beinhaltet alle Oberflächen, also z. B. HTML Seiten die keine eigene Logik haben und nur eine Ansicht bzw. Formatierung beschreiben. Der Controller verarbeitet Zugriffe auf die Oberfläche, z. B. Knopfdrücke oder die Einfügung von zusätzlichen Absätzen in die bestehende Struktur der Oberfläche sowie das befüllen der HTML Oberfläche mit Daten. Dafür benötigte Daten werden vom Model bereitgestellt, welches beispielsweise auf Datenbanken zugreift und die eingelesenen Daten verarbeitet und weiterleitet. Interaktionen mit der Oberfläche durch den User werden ebenfalls vom Controller ausgewertet und an das Backend weitergeleitet.

Angefügt ist eine beispielhafte Skizze, die das Zusammenspiel der drei Komponenten bildlich grob erläutern soll.

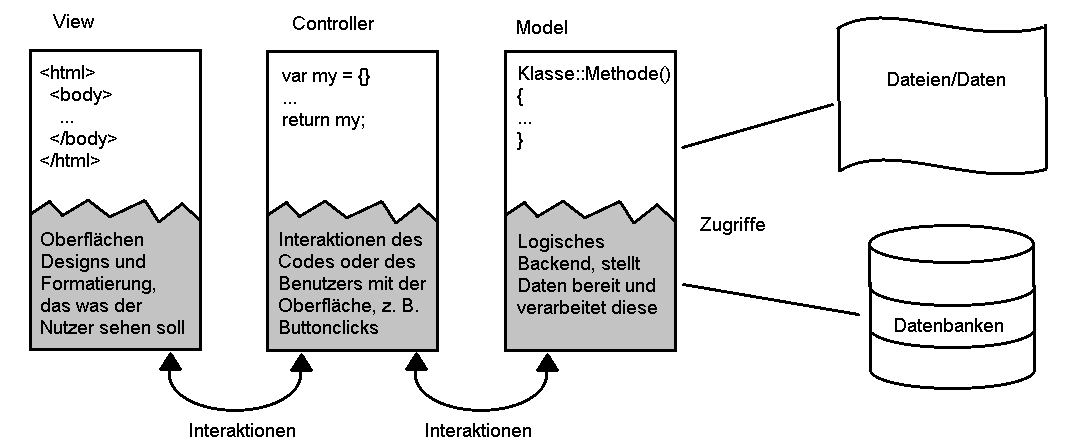


Abbildung 1: MVC Beispielhafte Darstellung

## Eidesstattliche Erklärung/Projektantrag

Es folgen die **Eidesstattliche Erklärung** sowie der **genehmigte Projektantrag**.