|  |
| --- |
| Kreisverwaltung Mettmann |
| Entwickluung einer To-Do-Anwendung |
| [Untertitel des Dokuments] |

|  |
| --- |
| Dirk  [Datum] |

Inhaltsverzeichnis

[1 Einleitende Bemerkungen 2](#_Toc409942326)

[2 Projektauftrag 3](#_Toc409942327)

[2.1 Projektumfeld 3](#_Toc409942328)

[2.1.1 Firmenumfeld 3](#_Toc409942329)

[2.1.2 Abteilungsumfeld 3](#_Toc409942330)

[2.2 Ist-Zustand 3](#_Toc409942331)

[2.2.1 Zielsetzung 3](#_Toc409942332)

[2.3 Aufgabenstellung 3](#_Toc409942333)

[2.3.1 Abgrenzung zu Fremdleistungen 3](#_Toc409942334)

[2.4 Prozessschnittstellen 3](#_Toc409942335)

[2.4.1 Personelle Schnittstellen 3](#_Toc409942336)

[2.4.2 Technische Schnittstellen 3](#_Toc409942337)

[2.5 Technisches Umfeld 3](#_Toc409942338)

[2.6 Einzuhaltende Standards 3](#_Toc409942339)

[2.6.1 Qualitätsmanagement-Standards 3](#_Toc409942340)

[2.6.2 Entwicklungsstandards 3](#_Toc409942341)

[3 Projektplanung 4](#_Toc409942342)

[3.1 Vorgehensweise 4](#_Toc409942343)

[3.1.1 Mögliche Lösungsalternativen 4](#_Toc409942344)

[3.2 Projekt- und Zeitplanung 4](#_Toc409942345)

[3.3 Änderungen gegenüber dem Projektauftrag 4](#_Toc409942346)

[3.4 Geplante Wirtschaftlichkeit 4](#_Toc409942347)

[3.5 Qualitätsmanagementmaßnahmen 4](#_Toc409942348)

[3.5.1 Zieldefinition 4](#_Toc409942349)

[3.5.2 Planung und Kontrolle 4](#_Toc409942350)

[3.5.3 Testplanung 4](#_Toc409942351)

[4 Projektdurchführung 4](#_Toc409942352)

[4.1 Phasenbezogener Projektablauf 4](#_Toc409942353)

[4.2 Gewählte Prinzipien, Methoden, Techniken und Werkzeuge 4](#_Toc409942354)

[4.3 Phasenbezogene Ergebnisse 4](#_Toc409942355)

[4.4 Zeitaufwand 4](#_Toc409942356)

[4.5 Kontrolle und Steuerung 4](#_Toc409942357)

[5 Projektabschluss 5](#_Toc409942358)

[5.1 Fachlicher Soll- / Ist-Vergleich 5](#_Toc409942359)

[5.2 Zeitlicher Soll- / Ist-Vergleich 5](#_Toc409942360)

[5.2.1 Abweichungen 5](#_Toc409942361)

[5.3 Wirtschaftlichkeitsbetrachtung 5](#_Toc409942362)

[5.3.1 Break-even-point 5](#_Toc409942363)

[5.4 Reflexion 5](#_Toc409942364)

[5.4.1 Fachliche Reflexion 5](#_Toc409942365)

[5.4.2 Persönliche Reflexion / lessons learned 5](#_Toc409942366)

[6 Anhang 6](#_Toc409942367)

[Anhang A: Glossar 6](#_Toc409942368)

[Anhang B: Abkürzungsverzeichnis 6](#_Toc409942369)

[Anhang C: Quellenverzeichnis 6](#_Toc409942370)

[Anhang D: Lasten- und Pflichtenheft 6](#_Toc409942371)

[Anhang E: Testprotokoll 6](#_Toc409942372)

[Anhang F: Projektantrag 6](#_Toc409942373)

# 1 Einleitende Bemerkungen

Die Folgende Projektdokumentation beschreibt die Planung und den Ablauf eines Projekts, wie es auch als Abschlussprojekt im Rahmen der Ausbildung zum Fachinformatiker durchgeführt wird. Das Projekt wird eigenständig in Partnerarbeit durchgeführt.

Begriffe, die im Glossar erklärt werden, werden beim ersten Auftreten im Text durch ein \* gekennzeichnet.

# 2 Projektauftrag

## 2.1 Projektumfeld

Das Berufskolleg Hilden ist eine Bildungseinrichtung im Kreis Mettmann zur teil- oder vollzeitschulischen Ausbildung in verschiedenen wirtschaftlichen und technischen Bereichen. In den ITB-Klassen zur teilschulischen Ausbildung zum Fachinformatiker, wird in den Klassen ein Projekt zur Vorbereitung auf das Abschlussprojekt der IHK durchgeführt.

## 2.2 Ist-Zustand

Innerhalb des Berufskollegs Hilden werden zu erledigende Aufgaben bisher manuell erstellt und verwaltet in Form von offline Checklisten oder als Listen on Word- oder Excel-Dateien gespeichert. Diese verwendeten Zettel können leicht verloren gehen oder unübersichtlich werden.

## 2.3 Zielsetzung

Ziel des Projekts ist es eine Anwendung in C# zu entwickeln, die es ermöglicht beliebig viele To-Do-Listen mit dazugehörigen Aufgaben in einem System anzulegen und zu verwalten. Die einzelnen Aufgaben sollen jeweils ein Start- und Enddatum enthalten sowie die Möglichkeit der Aufgabe einen Kontakt hinzuzufügen. Die Listen können durch das Programm angelegt, bearbeitet und gelöscht werden. Dies trifft auch auf die dazugehörigen Aufgaben und Kontakte zu. Des Weiteren sollen die Aufgaben einen Status besitzen, der anzeigt, ob eine Aufgabe erledigt wurde oder noch zu bearbeiten ist. Alle anfallenden Daten werden in einer dafür angelegten Datenbank abgespeichert. Das Programm kann durch die Eingabe eines Benutzernamens und dem dazugehörigen Passwort, von berechtigten Benutzern verwendet werden.

Damit soll die Anwendung dazu beitragen, zu erledigende Aufgaben übersichtlicher und strukturiert darzustellen sowie den Verwaltungsaufwand beim Planen von Tätigkeiten zu verringern.

## 2.5 Prozessschnittstellen

Für die Ausführung des Projekts werden keine weitere Schnittstellen zu anderen externen System benötigt.

## 2.5 Technisches Umfeld

Die zu erstellende Anwendung wird nach Beendigung auf den Windows 7 Rechnern, die in dem großen Schulnetzwerk verwaltet werden, zur Verfügung gestellt.

## 2.6 Einzuhaltende Standards

Bei der Umsetzung des Projekts werden zur Sicherstellung der Softwarequalität und Lesbarkeit unterschiedliche Standards zur Anwendung gebracht. Mit der ISO 9126 soll gewährleistet werden, dass die Anwendung zuverlässig und ordnungsgemäß arbeitet. Des Weiteren soll sie dazu beitragen, die Bedienbarkeit des Programms benutzerfreundlich zu halten und durch eine einfache Bedienung eine zeitsparende Benutzung zu ermöglichen.

Darüber hinaus werden auch betriebsinterne Standards angewendet, sie sich auf den Aufbau, Benennungen und die Strukturierung der Anwendung beziehen. Diese sollen ebenfalls die Lesbarkeit und Übersichtlichkeit des Programms steigern.

# 3 Projektplanung

## 3.1 Verwendetes Vorgehensmodell

Für die Umsetzung des Projekts wird als Vorgehensmodell das erweiterte Wasserfallmodell von Winsten W. Royce verwendet. Im Gegensatz zum klassischen Wasserfallmodell ermöglicht es die Erweiterung bei ungeplanten Komplikationen wieder zur vorherigen Phase zurückzukehren und diese mit den veränderten Bedingungen erneut zu durchlaufen. Dies ermöglicht einen agilen Umgang mit z.B veränderten Anforderungen. Unter Anhang G befindet sich eine graphische Darstellung des verwendeten Wasserfallmodells.

## 3.2 Projekt- und Zeitplanung

Für die Umsetzung des Projekts wurde eine Gesamtzeit von 70 Stunden eingeplant, die in mehrere Phasen eingeteilt wird. Die jeweiligen Phasen können aus der folgenden Tabelle entnommen werden. Unter Anhang D befindet sich eine genauere Aufteilung der Tätigkeiten der einzelnen Phasen.

|  |  |
| --- | --- |
| Planungsphase | Geplante Zeit |
| Planungsphase | 10,5 Std. |
| Durchführungsphase | 44 Std. |
| Testphase | 3 Std. |
| Abnahme und Deployment | 2,5 Std. |
| Dokumentation | 10 Std. |
| Gesamt | 70 Std. |

## 3.3 Änderungen gegenüber dem Projektauftrag

Anders als im Projektantrag angegeben werden die Datenbankabfragen nicht über das Entity Framework getätigt. Hierzu wird eine Extension für C# namens Dapper verwendet. Es ermöglicht eine übersichtliche Strukturierung der verwendeten SQLite-Befehle und das Anbinden von Variablen an die SQL-Query zur Laufzeit.

## 3.4 Geplante Wirtschaftlichkeit

### 3.4.1 „Make or Buy“- Entscheidung

Auf dem Markt gibt es viele gibt es viele Anwendungen, die eine Verwaltung von Aufgaben mit Hilfe von ToDo-Listen ermöglichen. Es wurde sich für eine Eigene Lösung entscheiden, da bei den meisten der bereits vorhandenen Software Lösungen Zum Beispiel monatliche Gebühren für die Verwendung anfallen, die den Haushalt regemäßig stark belasten. Des Weiteren kann die eigenentwickelte Anwendung jeder Zeit auf die internen Anforderungen angepasst werden.

### 3.4.2 Kostenplanung

Die Kosten des Projektes ergeben sich aus den für die Umsetzung benötigten Arbeitsstunden und Bruttostundensatz der am Projekt beteiligten Mitarbeiter. Da aus Datenschutzgründen keine genauen Gehaltskosten herausgegeben werden dürfen, werden für das Projekt exemplarisch folgende Gehaltssätze für die beteiligten Mitarbeiter festgelegt:

* Auszubildender 20€
* Mitarbeiter des BK Hilden (BKHM) 45€

Die anfallenden Kosten für Ressourcen wie Strom, Internet und die verwendete Hardware am Arbeitsplatz sind in den Stundensätzen bereits enthalten. Dadurch ergibt sich die folgende Kostenverteilung:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fachgespräch** | 1x Azubi | 2 Std. | 20 € | 22 € | **102** € |
| 1 | 1x BKHM | 2 Std. | 40 € | 80 € |  |
| **Entwicklung** | 2x Azubi | 27,75 Std. | 20 € | 1.110 € | **1.110** € |
| **Deploymnet** | 1x Azubi | 2,5 Std. | 20 € | 50 € | **50** € |
| **Gesamt** |  |  |  |  | **1.262** € |

Damit ergeben sich für die Eigenentwicklung Kosten in Höhe von **1.262 €.**

### 3.4.3 Amortisationsdauer

Eine wichtige Frage bei jedem Projekt ist es, wie lange es dauert bis die resultierenden Einsparungen den nötigen Aufwand auf gewiegt haben. Dies wird durch die Amortisationszeit beschrieben, die sich für gewöhnlich aus der Division der Anschaffungsausgaben und des durchschnittlichen Rückflusses per Jahr berechnet.

Einsparte Zeit pro Jahr:

Eingespartes Geld pro Jahr:

Amortisationsdauer:

Damit wird der Break-Even-Point nach etwa 17 Tagen erreicht.

# 4 Projektdurchführung

## 4.1 Aufbau der Datenbank

Zur Speicherung und Verwaltung der anfallenden Daten der Anwendung wie die Listen und deren dazugehörigen Aufgaben, wird eine SQLite Datenbank aufgesetzt. Innerhalb der Anwendung wird die Datenbank neu erstellet soweit sie nicht bereits vorhanden ist. Danach werden die benötigten Tabellen über SQL-befehle erstellt. Ein systematischer Aufbau der verwendeten Datenbank findet sich im Anhang E Datenbank ER-Modell.

## 4.2 Aufbau des Userinterfaces

Um das Programm verwenden zu können benötigt dieses eine Grafische Benutzer Oberfläche. Da es sich um eine WPF-Anwendung handelt, besteht die Benutzeroberfläche aus mehreren Fenstern, die vorher durch XML strukturiert wurden. Buttons auf den einzelnen Fenstern verwiesen auf weitere, wobei jedes Fenster eine andere Aufgabe hat.

## 4.3 Implementierung der Programmlogik

Damit den einzelnen Feldern und Buttons innerhalb der 20Fenster Funktionen und Aktionen zugeteilt werden können, werden sogenannte Code-behind Dateien verwendet. Sie beschreiben, was beim Aufruf eines Fensters oder beim Klick auf einen Button geschehen soll geschehen soll. Innerhalb der XML-Datei eines Fensters werden den Feldern Namen zugeordnet, damit diese von dem Code angesprochen werden können. Des Weiteren werden von dort die Funktionen zum Lesen oder Speichern der Daten aus bzw. in die Datenbank aufgerufen. Die Funktionalitäten der Datenbank wie das Schreiben oder Lesen werden durch die Klasse ApplicationService bereitgestellt. Sie enthält alle Aktionen, die innerhalb des Programms auf die Datenbank ausgeführt werden können.

## 4.4 Ablauf der Anwendung

Bevor ein Benutzer die Anwendung verwenden kann, muss dieser sich mit einem Benutzernamen und einem Passwort in der Anwendung registrieren. Nach einer erfolgreichen Registrierung kann der User sich über den Login anmelden. Danach bieten sich die Möglichkeiten Kontakte anzulegen, die später Aufgaben zugeteilt werden können, oder ToDo- Listen anzulegen. Die angelegten Kontakte und ToDo-Listen sind User abhängig und könne auch nur von diesem eingesehen werden. Es können beliebig viele ToDo Listen mit beliebig Aufgaben angelegt werden. Wird eine Aufgabe erledigt, wird dies in der Anwendung durch ein Häkchen gekennzeichnet. Werden alle Aufgaben einer ToDo-Liste erledigt, wird auch diese als erledigt markiert. Alle angelegten Elemente können durch den Klick auf den Löschen Button auch wieder aus der Anwendung entfernt werden.

Ein beispielhafter Ablauf der Anwendung wird unter Anhang F noch mal grafisch in Form eines Sequenzdiagrams dargestellt.

# 5 Testphase

Bevor die Anwendung den Kunden übergeben werden kann, muss geprüft werden, dass alle Funktionen wie geplant ausgeführt werden und Fehler beim Auftreten richtig behandelt werden. Dafür werden einige Testszenarien aufgesetzt, die Funktionalität ausgewählter Funktionen überprüft. Die restlichen Funktionen werden per Hand auf Fehler geprüft. Eine Auswahl der Testszenarien findet sich unter Anhang H.

# 6 Projektabschluss

## 6.1 Fachlicher Soll- / Ist-Vergleich

Alle im Pflichtenheft vermerkten erforderlichen Kriterien konnten während der Durchführung in der Anwendung realisiert werden. Die angeführten Wunschkriterien konnten nicht in die Anwendung implementiert werden, da die Zeit mit der Umsetzung der Musskriterien bereits ausgereizt war.

## 6.2 Zeitlicher Soll- / Ist-Vergleich

Wie aus der folgenden Tabelle abgelesen werden kann, wurde für die Planung des Projekts weniger Zeit benötigt als eingeplant wurde. Da die Durchführung des Projekt jedoch etwas mehr Zeit in Anspruch genommen hat, gleichen sich die beiden zeitlichen Abweichungen aus und haben somit keinen Einfluss auf die Gesamtzeit.

|  |  |  |  |
| --- | --- | --- | --- |
| **Planungsphase** | **Geplante Zeit** | **Tatsächliche** | **Differenz** |
| Planungsphase | 10.5 Std. | 10 Std. | + 0.5 |
| Durchführungsphase | 44 Std. | 44.5 Std. | - 0.5 |
| Testphase | 3 Std. | 3 Std. | 0 Std. |
| Abnahme und Deployment | 2.5 Std. | 2.5 Std. | 0 Std. |
| Dokumentation | 10 Std. | 10 Std. | 0 Std. |
| **Gesamt** | **70 Std.** | **70 Std.** | **0 Std.** |

## 6.4 Reflexion

Mit dem Durchführen des Projekts wurden Erfahrungen im Bereich der Planung und Konzeption eines Projektes gesammelt. Durch das Arbeiten mit WPF wurde das Verständnis von der Funktionsweise und dessen Aufbau verbessert. Auch das Verfassen von Dokumenten wie Pflichtenhefte konnte an einem realen Beispiel geübt und umgesetzt werden. Ebenso konnte Erfahrung in bei der Organisation der Mitarbeiter und der Wichtigkeit einer geordneten Kommunikation gemacht werden. Mit der Verwendung einer SQLite Datenbank konnte dadurch das Wissen zu SQL und dessen Verwendung vertieft werden.

# 6 Anhang

## Anhang A: Glossar

|  |  |
| --- | --- |
| Dapper | Eine NuGet Bibliothek für C# zum Verwalten von SQL-Datenbank aubfragen |
| NuGet | Ein System zur freien Verteilung von Software-Komponenten |

## Anhang B: Abkürzungsverzeichnis

|  |  |
| --- | --- |
| IDE | Integrated Development Environment |
| SQL | Structured Query Language |
| WPF | Windows Presentation Foundation |
| XML | Markup Language |

## Anhang C: Auszug Pflichtenheft

1. Einleitung

Durch einen eingehenden Kundenauftrag soll eine WPF-Anwendung erstellt werden. Diese soll ein Verwaltungstool für Kontakten und To-Do-Listen sein. Hierbei ist es wichtig, dass die To-Do-Listen mehrere Aufgaben mit jeweils mehreren Kontakten beinhalten können.

2. Zielbestimmungen

2.1 Musskriterien

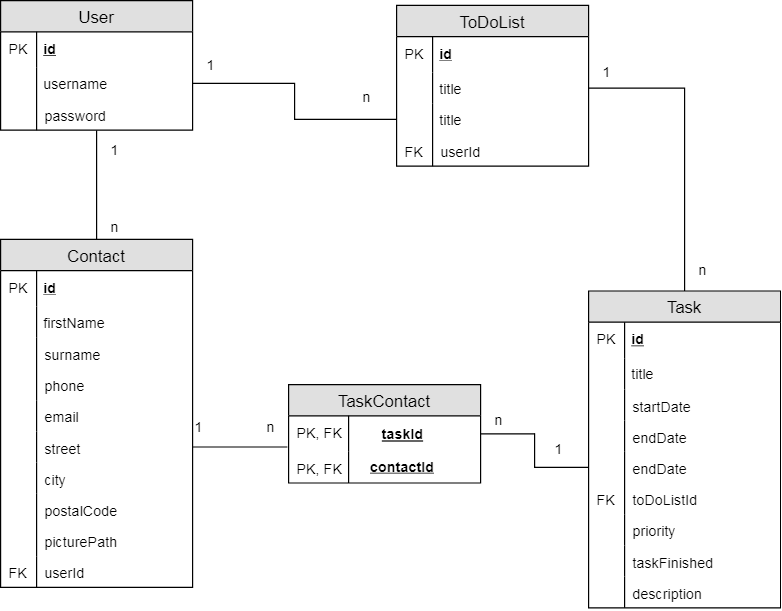
Die folgend genannten Kriterien müssen in der Anwendung enthalten sein:

* Eine übersichtliche visuelle Bedienoberfläche
  + Anlegen von To-Do-Listen mit folgenden Angaben:
* Titel
  + Anlegen von To-Do-Listen-Aufgaben mit folgenden Angaben:
* Titel
* Start- und Enddatum
* Priorität
* Beschreibung
  + Anlegen von Kontakten mit folgenden Angaben:
* Vor- und Nachname
* Adresse
* Email
  + Zuweisung von Kontakten zu den Aufgaben
  + Auflistung aller angelegten Listen
  + Löschen der Listen und der dazugehörigen Aufgaben
  + Löschen von einzelnen Aufgaben
  + Löschung von Kontakten
  + Editieren von Kontakten, Listen und Aufgaben
  + Kennzeichnung der Aufgaben als erledigt oder offen
  + Speicherung und Abruf der Daten über eine Datenbank

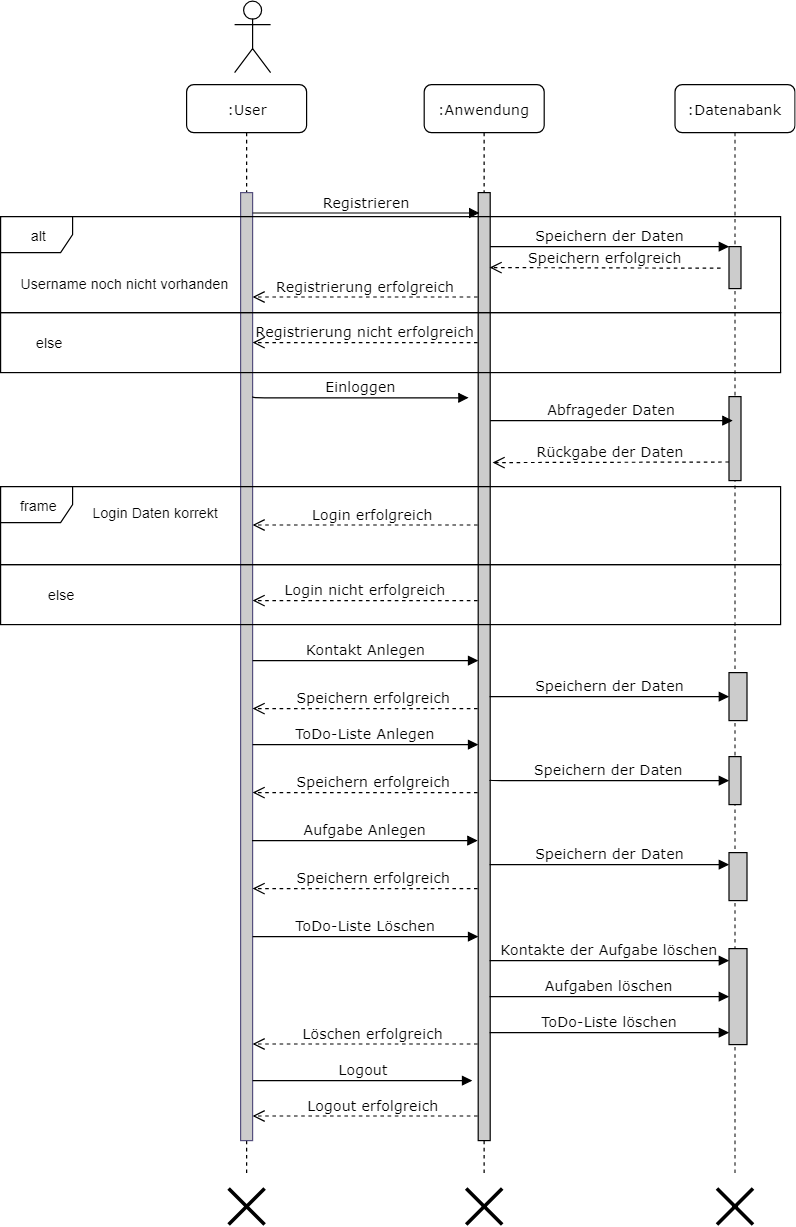
## Anhang D: Detaillierte Zeitplanung

|  |  |  |
| --- | --- | --- |
| **Planungsphase** |  | **10,5** |
| 1. Einteilung der Phasen |  | 1,5 |
| 2. Durchführung ist Analyse |  | 1,0 |
| 3. Erstellung Soll Konzept |  | 2 |
| 4. Erstellung des Pflichtenhefts |  | 2 |
| 5. Durchführung der Wirtschaftlichkeitsanalyse | | 2 |
| 6. Erstellung des ER Models |  | 1 |
| 7. Erstellung des Sequenzdiagramms |  | 1 |
|  |  |  |
| **Durchführungsphase** |  | **44** |
| 1. Aufbau und Anbindung der Datenbank |  | 2 |
| 2. Erstellen des User Interfaces |  | 5 |
| 3. Implementierung Programlogik |  | 33 |
| 3.1 Login |  | 4 |
| 3.2 Anlegen von Todo-Listen |  | 7 |
| 3.3. Anlegen von Aufgaben |  | 12 |
| 3.4 Erstellen von Kontakten |  | 10 |
| 4. Erstellung des Benutzerhandbuchs |  | 4 |
|  |  |  |
| **Testphase** |  | **3** |
| 1. Test der Funktionalität |  | 3 |
|  |  |  |
| **Abnahme und Depolyment** |  | **2,5** |
| 1. Soll-Ist Vergleich |  | 1,5 |
| 2. Abnahme durch Kunden |  | 1 |
|  |  |  |
| **Dokumentation** |  | **10** |
|  |  |  |
| **Summe** |  | **70** |

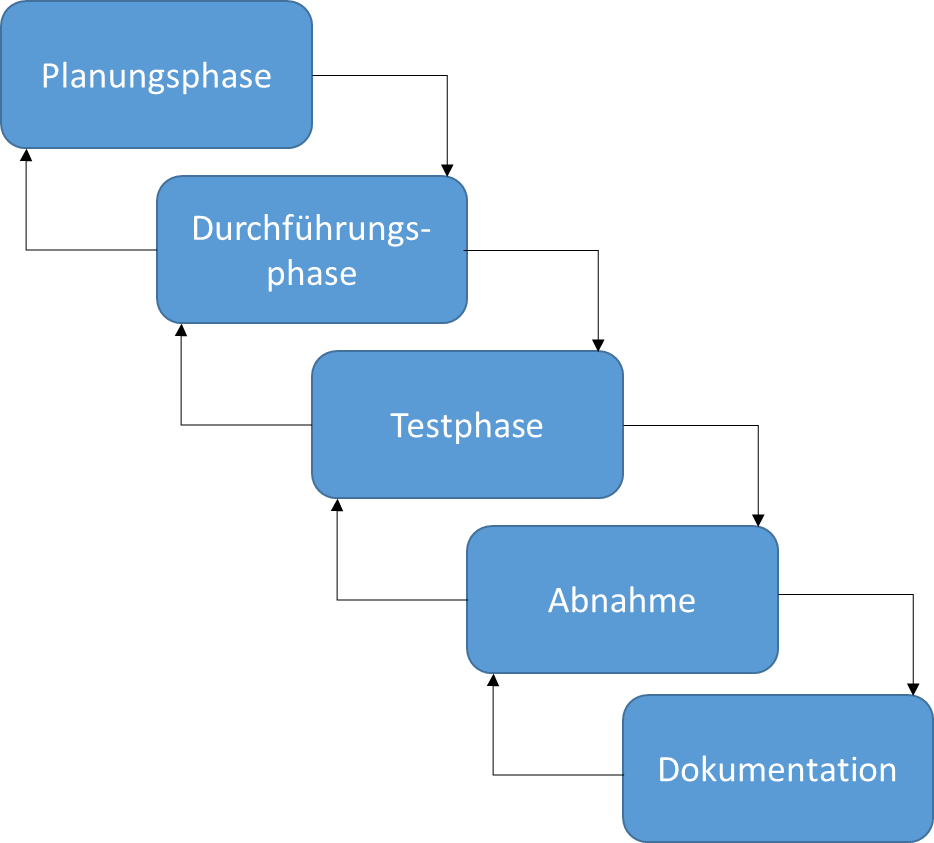
## Anhang E: Datenbank ER-Modell



## Anhang F: Sequenzdiagramm



## Anhang G: Wasserfallmodell



## Anhang H: Testprotokoll

Legende:

|  |  |
| --- | --- |
| OK | Test erfolgreich durgeführt. |
| NOK | Test nicht erfolgreich durchgeführt. |
| OK mit Bewertung | Test erfolgreich mit kleinen Abweichungen. |
| NT | Nicht testbar. |
| Feedback | Absprache notwendig. |

Erfolgreiches Login:

|  |  |  |
| --- | --- | --- |
| Aktion | Erwartetes Ergebnis | Beobachtets Ergebnis |
| User startet das Programm. | Das Login Fenster wird geöffnet. |  |
| Der User loggt sich mit seinen vorhandenen Daten ein. | Das Willkommen-Fenster öffnet sich. |  |
| Testresultat |  | OK |

Login und Kontakt hinzufügen ohne Bild:

|  |  |  |
| --- | --- | --- |
| Aktion | Erwartetes Ergebnis | Beobachtets Ergebnis |
| User startet das Programm. | Das Login Fenster wird geöffnet. |  |
| Der User loggt sich mit seinen vorhandenen Daten ein. | Das Willkommen-Fenster öffnet sich. |  |
| Der User klickt auf den Button Kontakte. | Das Fenster für die Kontakte öffnet sich. |  |
| Der User klickt auf den Button Kontakt hinzufügen. | Das Fenster Kontakt hinzufügen öffnet sich ohne Daten. |  |
| Der User gibt die Daten ein. | Die Daten werden übernommen. |  |
| User klickt auf Speichern. | Daten werden in die Datenbank übernommen. Das Fenster schließt sich und das Fenster Kontaktliste erscheint. In der Liste steht der Kontakt mit drin. |  |
| Testresultat |  | OK |

Task bearbeiten:

|  |  |  |
| --- | --- | --- |
| Aktion | Erwartetes Ergebnis | Beobachtets Ergebnis |
| User startet das Programm. | Das Login Fenster wird geöffnet. |  |
| Der User loggt sich mit seinen vorhandenen Daten ein. | Das Willkommen-Fenster öffnet sich. |  |
| Der User klickt auf den Button ToDoListen. | Das Fenster für die ToDoListen öffnet sich. |  |
| Der User klickt auf eine Aufgabe. | Aufgabe lässt sich selektieren. |  |
| Der User klickt auf den Button Task bearbeiten. | Ein neues Fenster öffnet sich. Die Felder sind mit den Daten des Tasks befüllt. |  |
| Der User ändert die Daten. | Daten werden übernommen. |  |
| Der User klickt auf Speichern. | Die Daten werden in die Datenbank gespeichert und das Fenster schließt sich. Das Fenster für die Anzeige der ToDoListen erscheint. |  |
| Testresultat |  | OK |