Praxisarbeit Modul 322

*CRM App*

Projekt-Bericht 25.06.2024

Alain Bocherens

# Dokumentinformationen

|  |  |
| --- | --- |
| Auftraggeber | Lukas Müller (IBZ Basel) |
| Projektleiter | Alain Bocherens |
| Autor | Alain Bocherens |
| Ausgabedatum | 25 Juni 2024 |
| Status | Abgeschlossen |
| Version | 0.1 |

1. Inhalt

[Dokumentinformationen 1](#_Toc170215459)

[Teil 1: Ablauf und Umfeld 4](#_Toc170215460)

[1. Einleitung 4](#_Toc170215461)

[2. Aufgabenstellung 5](#_Toc170215462)

[2.1 Ausgangslage 5](#_Toc170215463)

[2.2 Detaillierte Aufgabenstellung 5](#_Toc170215464)

[2.3 Mittel und Methoden 7](#_Toc170215465)

[3. Projektmethodik 8](#_Toc170215466)

[3.1 Vorgehensmodell 8](#_Toc170215467)

[3.1.1 IPERKA 8](#_Toc170215468)

[Teil 2: Projektdokumentation 9](#_Toc170215469)

[4. Informieren 9](#_Toc170215470)

[5. Planen 10](#_Toc170215471)

[5.1 Aufwandschätzung 10](#_Toc170215472)

[5.2 ERM 11](#_Toc170215473)

[5.3 UML Use Case 12](#_Toc170215474)

[5.4 Mockups 13](#_Toc170215475)

[5.4.1 MainWindow 13](#_Toc170215476)

[5.4.2 CompanyWindow 15](#_Toc170215477)

[5.4.3 ContactWindow 16](#_Toc170215478)

[6. Entscheiden 17](#_Toc170215479)

[7. Realisieren 18](#_Toc170215480)

[7.1 Datenbank / Models 18](#_Toc170215481)

[7.1.1 Models 18](#_Toc170215482)

[7.1.2 DbContext 21](#_Toc170215483)

[7.1.3 Services / CRUD 22](#_Toc170215484)

[7.2 MainWindow / MainViewModel 25](#_Toc170215485)

[7.2.1 MainWindow 25](#_Toc170215486)

[7.2.2 MainViewModel 27](#_Toc170215487)

[7.2.3 Resultat 29](#_Toc170215488)

[7.3 CompanyWindow / ComanyViewModel 30](#_Toc170215489)

[7.3.1 CompanyWindow 30](#_Toc170215490)

[7.3.2 CompanyViewModel 31](#_Toc170215491)

[7.3.3 Resultat 31](#_Toc170215492)

[7.4 ContactWindow / ContactViewModel 32](#_Toc170215493)

[7.4.1 ContactWindow 32](#_Toc170215494)

[7.4.2 ContactViewModel 33](#_Toc170215495)

[7.4.3 Resultat 33](#_Toc170215496)

[8. Kontrollieren 34](#_Toc170215497)

[8.1 Testkonzept 34](#_Toc170215498)

[8.2 Testprotokoll 34](#_Toc170215499)

[9. Auswerten 35](#_Toc170215500)

[9.1 Reflexion 35](#_Toc170215501)

Teil 1: Ablauf und Umfeld

# Einleitung

Die Praxisarbeit wird im Rahmen des Moduls 322 durchgeführt. Die Durchführung des Projekts beginnt am 18.06.2024 und wird am 25.06.2024 abgeschlossen.

Die Arbeitsbereiche des Projekts sind die folgenden:

* Applikationsentwicklung OO
* MS Windows
* C#

Der Titel des Projekts lautet «CRM App»

# Aufgabenstellung

## Ausgangslage

Kundenbeziehungen sind eine gute Sache. Für die Verwaltung von Kunden und Ansprechpersonen wird oft eine CRM (Customer-Relationship-Management) Anwendung eingesetzt. Eine bereits existierende Softwarelösung soll nun mit einem kleinen CRM-Modul ausgebaut werden. In Vorabgesprächen wurde festgelegt, dass die Benutzeroberfläche strikte nach den Regeln der harmonischen Dialoggestaltung, intuitiv und benutzerfreundlicher aufgebaut werden muss.

## Detaillierte Aufgabenstellung

Sie erhalten den Auftrag für dieses CRM-Modul eine Benutzeroberfläche zu entwerfen, welche die

Grundprinzipien zur Dialoggestaltung nach ISO 9241-110 berücksichtigt, sodass eine optimierte

anwenderfreundliche Bedienung gewährleistet ist.

**In dieser CRM-Anwendung sind folgende Elemente vorzusehen:**

➢ Firma mit Namen, Strasse, PLZ, Ort und Tags (z.B. .NET, Java etc.)

➢ Kontaktperson mit Anrede, Vor-/Nachname, Rolle, E-Mail, Telefon, Handy, Status (Aktiv,

Inaktiv).

➢ Listenanzeige der Firmen mit den Kontaktpersonen und Anzeige der Detailinformationen.

➢ Schaltflächen, um neue Firma oder Ansprechpartner hinzufügen zu können.

➢ Beim Hinzufügen einer Firma oder Ansprechpartners muss eine Speichern- und Abbrechen-

Schaltfläche vorhanden sein.

➢ Ansprechpartner dürfen aufgrund einer langfristigen Datenhaltung nicht gelöscht, sondern

ausschliesslich mit Status Inaktiv markiert werden.

**Aufgabe Design**

Skizzieren Sie für dieses CRM-Modul eine vollständige Benutzeroberfläche mit allen erforderlichen Elementen aus der Auftragsbeschreibung. Erstellen Sie die Skizze mit einem Mockup Tool (Sketch-Flow etc.), welches Sie im Unterricht verwendet haben. Achten Sie, dass die Steuerelemente gut und eindeutig erkennbar sind und der Dialog nach den ergonomischen Richtlinien aufgebaut ist. Prüfen Sie auch, dass die Grundsätze der Dialoggestaltung aus der Norm EN ISO 9241-110 strikte eingehalten sind.

Abzugebende Lösungselemente: A21-Design.png (Designentwurf als JPG oder PNG-Datei.)

**Aufgabe Realisierung**

Erstellen Sie eine WPF-Anwendung und implementieren Sie Ihr Design aus der Entwurfsphase

(Aufgabe Design). Die Benutzeroberfläche muss vollständig inkl. sämtlichen Funktionen realisiert

werden. Zudem soll die Oberfläche auch eine spätere Spracherweiterung ermöglichen und aus

wartungstechnischen Gründen keine hardcodierte Ressourcen enthalten. Um die Darstellung von der Logik der Benutzerschnittstelle zu trennen ist das MVVM-Entwurfsmuster zu verwenden.

## Mittel und Methoden

Entwicklungsumgebung: Microsoft Visual Studio 2022

Programmiersprachen: C#, XAML

Frameworks: .NET 8, WPF

Datenbank: MySQL

Quellcodeverwaltung: GitHub

# Projektmethodik

## Vorgehensmodell

Das Vorgehensmodell beschreibt die Prozesse und Abfolge von Aktivitäten, die zur Durchführung

eines Projektes notwendig sind. Dabei wird das Projekt in verschiedene, strukturierte Abschnitte

unterteilt.

## IPERKA

Das Vorgehensmodell IPERKA besteht aus den sechs Phasen: Infomieren, Planen, Entscheiden,

Realisieren, Kontrollieren und Auswerten. Ich habe mich für dieses Vorgehensmodell entschieden,

da es sich sehr gut für kleine und individuelle Projekte eignet. Es ist universell einsetzbar und kann

leicht an die spezifischen Anforderungen eines Projektes angepasst werden. Dass ich das

IPERKA-Modell bereits aus meiner Ausbildung kenne und bereits mehrere kleinere Projekte damit

absolviert habe, hat bei dieser Entscheidung auch einen Einfluss gehabt.

Teil 2: Projektdokumentation

# Informieren

Die erste Phase von IPERKA ist: Informieren. Diese Phase dient dazu sich über den genauen

Auftrag zu informieren und sich einen Überblick darüber zu verschaffen, um festzustellen, ob man

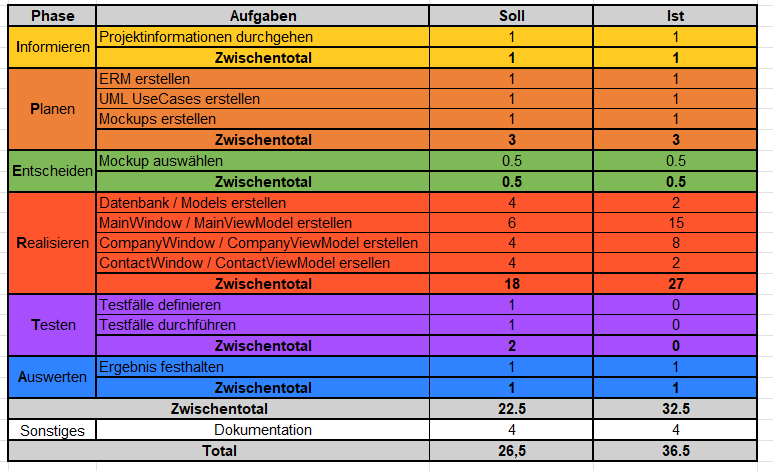
alles Nötige hat, um den Auftrag zu vollenden.

Ich habe mir dafür nochmals die genaue Aufgabenstellung der Praxisarbeit angeschaut und bin dann noch die Bewertungskriterien durchgegangen. Danach habe ich eine Checkliste mit den einzelnen Arbeitsschritten gemäss Aufgabenstellung erstellt.

# Planen

Die zweite Phase von IPERKA ist: Planen. In dieser Phase erstellt man Zeitpläne, Modelle und Diagramme.

## Aufwandschätzung



## ERM

Mit dem ERM (Entity-Relationship-Modell) werden Entitäten, Attribute und Beziehungen zwischen den Entitäten einer Datenbank grafisch dargestellt. Das ERM wurde mit der Software «Dia» erstellt.

Ein Bild, das Text, Diagramm, Muster, Reihe enthält.

Automatisch generierte Beschreibung

## UML Use Case

Ein Bild, das Diagramm, Entwurf, Text, Zeichnung enthält.

Automatisch generierte Beschreibung

|  |  |
| --- | --- |
| Use Case Nummer | Beschreibung |
| Use Case 1 | Der Benutzer kann alle Firmen (Company) und Ansprechpersonen (Contact) ansehen. |
| Use Case 2 | Der Benutzer kann neue Firmen und Ansprechpersonen erstellen. Für Use Case 2 muss man Use Case 1 bereits erledigt haben («include»). Der Use Case 2 kann man optional nach dem Use Case 1 erledigen («extend»). |
| Use Case 3 | Der Benutzer Firmen und Ansprechpersonen bearbeiten. Für Use Case 3 muss man Use Case 1 bereits erledigt haben («include»). Der Use Case 3 kann man optional nach dem Use Case 1 erledigen («extend»). |
| Use Case 4 | Der Benutzer kann eine Ansprechperson löschen. Für Use Case 4 muss man Use Case 1 bereits erledigt haben («include»). Der Use Case 4 kann man optional nach dem Use Case 1 erledigen («extend»). |

## Mockups

Die Mockups dienten mir als Vorlage für die Darstellung der WPF-Anwendung. Ich habe sie mit der Software «Balsamiq Wireframes» erstellt.

## MainWindow

Für das MainWindow habe ich zwei Varianten erstell.

**Variante 1:**

**Ein Bild, das Text, Screenshot, Display, Zahl enthält.

Automatisch generierte Beschreibung**

In der ersten Variante gibt es über der Tabelle einen Button mit dem neuen Firmen erstellt werden. In der Tabelle der Firmen befindet sich rechts eine Spalte mit dem Button zum Erstellen der Ansprechpersonen. Die Tabelle Contact wird als «nested grid» zu jedem Firmeneintrag implementiert. Rechts in der Tabelle Contact befindet sich eine Spalte mit dem Button zum Löschen von Ansprechpersonen. Die Bearbeitung wird in der Zeile stattfinden.

**Variante 2:**

Ein Bild, das Text, Screenshot, Display, parallel enthält.

Automatisch generierte Beschreibung

Bei der zweiten Variante gibt es keine Spalte mit Buttons, sondern diese befinden sich in einer Toolbar über der Tabelle. Die Bearbeitung wird in einem separaten Fenster stattfinden.

## CompanyWindow

**Ein Bild, das Screenshot, Text, parallel, Diagramm enthält.

Automatisch generierte Beschreibung**

Das CompanyWindow beinhaltet ein einfaches Formular mit den Buttons Save und Cancel.

## ContactWindow

Ein Bild, das Screenshot, Text, Diagramm, parallel enthält.

Automatisch generierte Beschreibung

Das ContactWindow beinhaltet ein einfaches Formular mit den Buttons Save und Cancel.

# Entscheiden

Ich habe mich beim MainWindow schlussendlich für die Variante 1 entschieden, da mir das Design mit den Buttons neben den Einträgen besser gefällt und ich somit auch ein XAML-Window weniger brauchen werde.

**Ein Bild, das Text, Screenshot, Display, Zahl enthält.

Automatisch generierte Beschreibung**

# Realisieren

Die vierte Phase der IPERKA ist: Realisieren. In dieser Phase wird das Projekt umgesetzt.

## Datenbank / Models

Die Datenbank wird als «Code First» mit C# und OR Mapper implementiert.

Anmerkung: Der DbContext und die Services wurden zwar implementiert, aber Aufgrund von Problemen und dem Zeitmangel wurden in den ViewModels keine CRUD-Methoden für die Datenbank implementiert.

## Models

Als erstes habe ich die Modelle für die Datenbank erstellt. Diese dienen als Vorlage wie jede Tabelle in der Datenbank erstellt werden soll.

Ich habe die abstrakte Klasse «BaseModel» als Basismodell erstellt, welches an alle anderen Modelle die Eigenschaft «Id» als «Primary Key» übergibt.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

Als nächstes habe ich die Modellklasse «CompanyModel» erstellt, die als Vorlage für die Dantenbanktabelle «Company» dient.

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Für das Feld Tag habe ich das Enum «Tags» erstellt:

Ein Bild, das Text, Screenshot, Schrift, Software enthält.

Automatisch generierte Beschreibung

Danach habe ich die Modelklasse «ContactModel» erstellt, die als Vorlage für die Dantenbanktabelle «Contact» dient.

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Für die Felder Title (Anrede) und Status habe ich jeweils ein Enum erstellt:

Ein Bild, das Text, Screenshot, Schrift, Design enthält.

Automatisch generierte Beschreibung Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

## DbContext

Der «DbContext» oder auch Datenbankkontext ist die Verbindung zwischen der Datenbank und der Anwendung. Es ist verantwortlich für das Mapping der Datenbanktabellen und erleichtert den Zugriff auf die Datenbank. Ausserdem wird es benötigt, um Daten in der Datenbank abzufragen, hinzuzufügen, zu ändern und zu löschen.

Ich habe meinen «DbContext» AppDbContext genannt.



Mit den «DbSet»-Properties ermögliche ich den Zugriff auf die Datenbanktabelle der entsprechenden Entitätsklasse, um Abfragen und Änderungen vorzunehmen.

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

Mit der Methode «OnModelCreating» wird die Datenbank mit den Modellen als Vorlage erstellt.

Ein Bild, das Screenshot, Text, Reihe enthält.

Automatisch generierte Beschreibung

Für den Verbindungsaufbau mit dem MySql-Server habe ich die Methode «OnConfiguring» ersztellt.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

## Services / CRUD

CRUD steht für die Operationen «Create» (Erstellen), «Read» (Lesen), «Update» (Ändern) und «Delete» (Löschen). Mit Create werden Datenbankeinträge erstellt. Mit Read werden Daten aus der Datenbank ausgelesen. Mit Update werden Daten in der Datenbank angepasst oder geändert. Und mit Delete werden Datenbankeinträge gelöscht.

Für jedes Modell wird eine Service-Klasse erstellt, um die CRUD-Methoden zu implementieren.

Ich habe die abstrakte Klasse «BaseService» als Basisklasse für die anderen Services erstellt.

Ein Bild, das Text, Screenshot, Software, Schrift enthält.

Automatisch generierte Beschreibung

**CompanyService:**

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

**ContactService:**

**Ein Bild, das Text, Screenshot, Software, Betriebssystem enthält.

Automatisch generierte Beschreibung**

## MainWindow / MainViewModel

## MainWindow

Für die Tabellen im MainWindow habe ich die GridControl-Komponente von DevExpress verwendet.

**Grid «Company»:**

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

**Nested Grid «Contact»:**

**Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung**

## MainViewModel

Im MainViewModel habe ich dann im Konstruktor Beispiel-Daten erstellt.

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Und ich habe die Methoden für die Buttons implementiert:

**Ein Bild, das Text, Software, Multimedia-Software, Screenshot enthält.

Automatisch generierte Beschreibung**

**Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung**

## Resultat

Das Result sieht dann so aus:

Ein Bild, das Text, Screenshot, Display, Software enthält.

Automatisch generierte Beschreibung

PS: Leider konnte ich auf Grund eines Fehlers, den ich nicht ausfindig machen konnte, die Anwendung nicht starten. Deshalb habe ich nur ein Bild der Vorschau.

## CompanyWindow / ComanyViewModel

## CompanyWindow

Im CompanyWindow habe ich zuerst ein kleines Grid definiert.

Ein Bild, das Text, Screenshot, Schrift, Karte Menü enthält.

Automatisch generierte Beschreibung

Danach habe ich die Formular-Elemente und die Buttons implementiert.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

## CompanyViewModel

Im CompanyViewModel habe ich dann die Methoden für die Buttons Save und Cancel implementiert.

Ein Bild, das Text, Screenshot, Schrift, Software enthält.

Automatisch generierte Beschreibung

## Resultat

Das Result sieht dann so aus:

Ein Bild, das Reihe, Text, Diagramm, Screenshot enthält.

Automatisch generierte Beschreibung

PS: Leider konnte ich auf Grund eines Fehlers, den ich nicht ausfindig machen konnte, die Anwendung nicht starten. Deshalb habe ich nur ein Bild der Vorschau.

## ContactWindow / ContactViewModel

## ContactWindow

Im CompanyWindow habe ich zuerst ein kleines Grid definiert genau wie im CompanyWindow.

Ein Bild, das Text, Screenshot, Schrift, Karte Menü enthält.

Automatisch generierte Beschreibung

Danach habe ich die Formular-Elemente und die Buttons implementiert.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

## ContactViewModel

Im ContactViewModel habe ich dann die Methoden für die Buttons Save und Cancel implementiert.

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

## Resultat

Das Resultat sieht dann so aus:

Ein Bild, das Text, Reihe, Zahl, Screenshot enthält.

Automatisch generierte Beschreibung

PS: Leider konnte ich auf Grund eines Fehlers, den ich nicht ausfindig machen konnte, die Anwendung nicht starten. Deshalb habe ich nur ein Bild der Vorschau.

# Kontrollieren

Die fünfte Phase von IPERKA ist: Kontrollieren. In dieser Phase findet das Testing statt und es wird überprüft, ob das Projekt den Anforderungen entspricht.

## Testkonzept

Bei der Test-Methode habe ich mich für die «Black-Box-Test»-Methode entschieden. In dieser Methode werden die Testfälle anhand des UseCase-Diagramms durchgeführt und dementsprechend nummeriert.

Die Testfälle werden auf der Umgebung, auf der die Anwendung programmiert wurde, getestet.

Die Anwendung wird nur auf der Basis der Benutzbarkeit getestet. Performance-Test werden keine durchgeführt.

Anmerkung: Aufgrund eines Problems, das ich nicht lösen konnte, lässt sich die Anwendung nicht starten. Deshalb ist es auch nicht möglich irgendwelche Tests durchzuführen. Ich habe deswegen und auch auf Grund von Zeitmangel kein Testkonzept erstellt.

## Testprotokoll

Anmerkung: Aufgrund eines Problems, das ich nicht lösen konnte, lässt sich die Anwendung nicht starten. Weil deshalb nicht getestet werden kann, wurde kein Testkonzept erstellt und keine Tests durchgeführt.

# Auswerten

## Reflexion

Es wurde alles gemäss Aufgabenstellung und Planung implementiert. Leider konnte ich die Anwendung nicht starten da folgendes Problem aufgetreten ist. Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Ich konnte deshalb nicht testen, ob alles Funktioniert wie gewünscht. Ich habe auch auf Grund von Zeitmangel schlussendlich kein Exception-Handling mehr machen können und auch für die Speicherung in der Datenbank hat es nicht mehr gereicht. Ich wollte eig. auch noch das Design ausbessern. Zur Implementierung der Sprachauswahl bin ich leider auch nicht mehr gekommen.

Schlussendlich war es zusammen mit dem Stress von der Arbeit im Praktikumsbetrieb und weiteren Praxisarbeiten, die ich parallel zu dieser Arbeit erledigen musste, einfach zu viel und zu wenig Zeit.

Ich hoffe das dies bei der Bewertung berücksichtigt wird.