# Machbarkeitsstudie

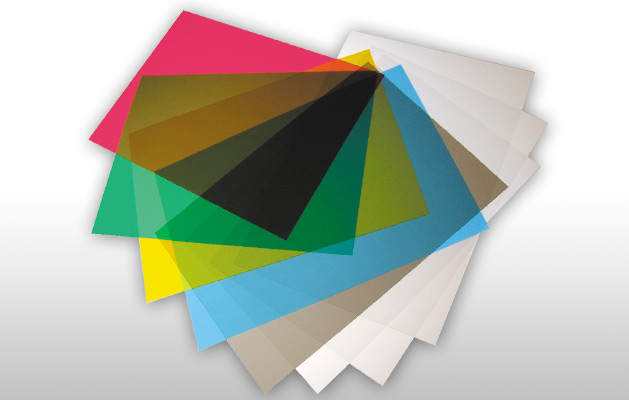
**Flitzer Ges.m.b.H**

# Datenbank für Flitzer Ges.m.b.H

**Simon Appel**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Datum | Unterschrift |
| *erstellt* | Simon Appel | 23.11.2016 |  |
| *geprüft* | Simon Appel | 11.12.2016 |  |
| *abgenommen* |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Autor | Datum | Status | Kommentar |
| 1 | Appel | 23.11.2016 | Draft | Erstellt |
| 2 | Appel | 11.12.2016 | Überarbeitet | Verbessert |



11.12.2016

**Inhaltsverzeichnis**

Inhalt

[Machbarkeitsstudie 1](#_Toc475979380)

[Datenbank für Flitzer Ges.m.b.H 1](#_Toc475979381)

[Name 1](#_Toc475979382)

[Datum 1](#_Toc475979383)

[Unterschrift 1](#_Toc475979384)

[erstellt 1](#_Toc475979385)

[Simon Appel 1](#_Toc475979386)

[23.11.2016 1](#_Toc475979387)

[geprüft 1](#_Toc475979388)

[Simon Appel 1](#_Toc475979389)

[11.12.2016 1](#_Toc475979390)

[abgenommen 1](#_Toc475979391)

[1)    Einleitung 2](#_Toc475979392)

[3)    Produktauswahl 2](#_Toc475979393)

[3.1 Trendanalyse 2](#_Toc475979394)

[3.2 Marktanalyse 3](#_Toc475979395)

[4)    Soll-Zustand 3](#_Toc475979396)

[4.1 Muss-Ziele 3](#_Toc475979397)

[4.2 Kann-Ziele 3](#_Toc475979398)

[4.3 Nicht-Ziele 4](#_Toc475979399)

[5) Produktfunktionen 4](#_Toc475979400)

[5.1 Benutzerverwaltung 4](#_Toc475979401)

[5.1.1 Benutzer registrieren (/LF0010/) 4](#_Toc475979402)

[5.1.1 Benutzer anmelden (/LF0020/) 5](#_Toc475979403)

[5.1.2 Benutzer anmelden (/LF0030/) 6](#_Toc475979404)

[5.2 Döner suchen 8](#_Toc475979405)

[5.2.1 Döner per Adresse finden (/LF0110/) 8](#_Toc475979406)

[5.2.2 Döner in der Umgebung finden (/LF0120/) 9](#_Toc475979407)

[5.2.3 Döner-Karte filtern (/LF0130/) 10](#_Toc475979408)

[5.3 Freunde verwalten 11](#_Toc475979409)

[5.3.1 Freunde hinzufügen (/LF0210/) 11](#_Toc475979410)

[5.3.2 Freundesliste anzeigen (/LF0220/) 12](#_Toc475979411)

[5.3.3 Freunde entfernen (/LF0230/) 13](#_Toc475979412)

[5.4 Freunde verwalten 14](#_Toc475979413)

[5.4.1 Freunde per App einladen (/LF0310/) 14](#_Toc475979414)

[5.4.2 Einladungen empfangen und bestätigen (/LF0320/) 15](#_Toc475979415)

[5.4 Aktivitätsdiagramm 17](#_Toc475979416)

[6) Machbarkeit 18](#_Toc475979417)

[6.1) TECHNISCHE MACHBARKEIT 18](#_Toc475979418)

[6.1.1 Technologie 18](#_Toc475979419)

[6.1.2 Umsetzung 18](#_Toc475979420)

[6.2.1 Personalaufwand 18](#_Toc475979421)

[6.2.3 Investitionsaufwand 19](#_Toc475979422)

[6.2.4 Nutzen 19](#_Toc475979423)

[6.2.6 Projektumfeldanalyse 19](#_Toc475979424)

[6.3) Persönliche Machbarkeit 20](#_Toc475979425)

[6.3.1 Nutzwertanalyse 20](#_Toc475979426)

[7)    Projektorganisation 21](#_Toc475979427)

[8)    Projektplanung 22](#_Toc475979428)

[8.1) Projektstrukturplan 22](#_Toc475979429)

[8.2) Meilensteinplan 22](#_Toc475979430)

[9)    Management Summary: 23](#_Toc475979431)

# 1)    Einleitung

Bei der Firma Flitzer sollen die Kundenanfragen, die Vermietung, die Rückgabe und die Wartung der Autoflotte soll informationsunterstützt umgesetzt werden.

Dazu soll eine Software und eine Datenbank zur Vermietung von Autos entwickelt werden. Weiters wird eine Webseite erstellt, mit der auch Autos reserviert und online vermietet werden können. Die Infrastruktur muss auch erneuert werden.

Die alte Datenbank muss mitübernommen werden damit bestehende Kunden und Autos schon in der neuen Datenbank vorhanden sind.  
2)    Ist-Zustand

Der gesamt Businessprozess wird per Hand erledigt. Um sich ein Auto zu mieten muss man derzeit noch zu den Vermietungsstellen gehen und dort persönlich das Auto mieten. Man kann sich auch noch nicht anschauen, was es für Autos gibt, da es noch keine Webseite gibt.

3)    Produktauswahl

3.1 Trendanalyse

Die Kunden verlangen, dass immer neuere Technologien verwendet werden. Man kann davon ausgehen, dass in den nächsten Jahren immer mehr Autos gemietet werden, da Geschäftsreisen oder ähnliches immer mehr vorkommen. Dabei braucht man öfters ein Mietauto.

3.2 Marktanalyse

Die Firma Flitzer muss sich von den anderen Firmen abheben, damit unsere Autos gemietet werden. Dies passiert dadurch, dass man sein Auto gezielt aussuchen kann und es welche von den verschiedensten Marken gibt. Dies ist bei anderen Firmen nicht der Fall. Dadurch wird sich unser Produkt im Markt behaupten.

4)    Soll-Zustand

4.1 Muss-Ziele

1. **Hohe Kompatibilität**

Doener-Spotted soll mit den Android-Versionen 5.X kompatibel sein. Da es in Android keine Bildschirmauflösung vorgeschrieben wird, muss das Programm mit unterschiedlichen Auflösungen und Bildschirmformaten kompatibel sein. Dies wird mittels einigen verschiedenen Geräten getestet, um eventuelle Änderungen vollziehen zu können.

1. **Benutzerfreundliche Oberfläche**

Die App muss einfach zu bedienen sein, damit jeder Benutzer schnell den nächsten Döner-Stand finden kann. Testpersonen werden mit dem zukünftigen Layout vertraut gemacht, um zu schauen ob es intuitiv bedienbar ist.

1. **Großer Datenbestand**

Die App soll einen großen Datenbestand aufweisen, damit sich die Wahrscheinlichkeit erhöht, einen Dönerstand in der Nähe zu finden. Deshalb werden alle Daten, von den Dönerständen, von [www.yelp.at](http://www.yelp.at) implementiert.

4.2 Kann-Ziele

1. **Implementierung der App auf einer Website**

Falls es sich zeitlich ausgeht, soll die App auch online als Website abrufbar sein. Dadurch kann man auch mit anderen Geräten, als mit einem Smartphone oder Tablette, die Funktionen der App benützen.

1. **Sicherheit**

Alle abgespeicherten Daten, sollen sicher auf einen Server abgespeichert werden. Diese sollen verschlüsselt übertragen werden.

4.3 Nicht-Ziele

1. **Bestellen per App**

Unsere App soll sich ausschließlich mit dem Finden von Döner-läden und ausmachen von Essens-Treffen beschäftigen. Für das Bestellen über das Internet, gibt es schon genug Vertreter die dieses Thema umgesetzt haben.

5) Produktfunktionen

* 1. Benutzerverwaltung
     1. Benutzer registrieren (/LF0010/)

Bei der Erstbenützung der App, soll sich der Benutzer registrieren. Dafür wird ein einmaliger Benutzername, eine E-Mail-Adresse und ein Passwort mit mindestens 8 Zeichen (min. 1 Großbuchstabe, min. 1 Zahl) als Anmeldedaten verwendet.

Nach erfolgreicher Registration sollen alle Dienste der App für den Benutzer erreichbar sein.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Benutzer registrieren (/LF0010/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Bei der Erstbenützung der App, soll sich der Benutzer registrieren. Dafür wird ein einmaliger Benutzername, eine E-Mail-Adresse und ein Passwort mit mindestens 8 Zeichen (min. 1 Großbuchstabe, min. 1 Zahl) als Anmeldedaten verwendet.  Nach erfolgreicher Registration sollen alle Dienste der App für den Benutzer erreich-bar sein. |
| **Auslöser** | Benutzer möchte sich bei der App registrieren |
| **Ergebnis** | Benutzer hat ein eigenes Konto |
| **Akteure** | Neuer Benutzer |
| **Eingehende Informationen** | Benutzername, Passwort und E-Mail |
| **Vorbedingungen** | Benutzer darf noch nicht am System registriert sein |
| **Nachbedingung** | Benutzer kann sich bei der Software anmelden |

* + 1. Benutzer anmelden (/LF0020/)

Nach erfolgreicher Registration (/LF0010/) kann sich der Benutzer anmelden, um auf die App zugreifen zu können. Dafür wird die Eingabe des Benutzernamens und des Passwortes benötigt. Wenn die eingegebenen Daten mit denen die in der Datenbank gespeichert sind übereinstimmen, soll der Benutzer alle Tasks der App ausführen können.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Benutzer anmelden (/LF0020/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Nach erfolgreicher Registration (/LF0010/) kann sich der Benutzer anmelden, um auf die App zugreifen zu können. Dafür wird die Eingabe des Benutzernamens und des Passwortes benötigt. Wenn die eingegebenen Daten mit denen die in der Datenbank gespeichert sind übereinstimmen, soll der Benutzer alle Tasks der App ausführen können. |
| **Auslöser** | Benutzer möchte sich bei der App anmelden |
| **Ergebnis** | User kann Funktionen der App benützen |
| **Akteure** | registrierter Benutzer |
| **Eingehende Informationen** | Benutzername, Passwort |
| **Vorbedingungen** | Benutzer muss registriert sein |
| **Nachbedingung** | Benutzer kann die Funktionen der App benutzen |

* + 1. Benutzer anmelden (/LF0030/)

Nach der Anmeldung (/LF0020/), kann sich der Benutzer wieder abmelden.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Benutzer abmelden (/LF0030/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Nach der Anmeldung (/LF0020/), kann sich der Benutzer wieder abmelden. |
| **Auslöser** | Benutzer möchte sich bei der App abmelden |
| **Ergebnis** | Benutzer kann nicht mehr auf die Funktionen der App zugreifen |
| **Akteure** | angemeldeter Benutzer |
| **Eingehende Informationen** | / |
| **Vorbedingungen** | Benutzer muss angemeldet sein |
| **Nachbedingung** | Benutzer kann sich jederzeit wieder an der App anmelden |

* 1. Döner suchen
     1. Döner per Adresse finden (/LF0110/)

Wenn eine neue Adresse eingetragen wird, wird der Weg-Punkt der Karte zur eingegebenen Adresse gesetzt und alle Läden in der Nähe anzeigt.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Döner per Adresse finden (/LF0110/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Wenn eine neue Adresse eingetragen wird, wird der Weg-Punkt der Karte zur eingegebenen Adresse gesetzt und alle Läden in der Nähe anzeigt. |
| **Auslöser** | Benutzer möchte nach einen Döner Laden suchen |
| **Ergebnis** | Benutzer finden einen Döner-Laden in der Nähe der eingegebenen Adresse |
| **Akteure** | angemeldeter Benutzer |
| **Eingehende Informationen** | Adresse |
| **Vorbedingungen** | Benutzer muss angemeldet sein und eine Adresse eingeben, außerdem muss eine Internetverbindung bestehen |
| **Nachbedingung** | Es kann die Karte mit einer anderen Adresse angezeigt werden |

#### Döner in der Umgebung finden (/LF0120/)

Sobald die App, nach der Anmeldung, geöffnet wird, öffnet sich eine Karte die deinen Standort und alle Döner-Läden in der Nähe auf der Karte anzeigt.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Döner per Adresse finden (/LF0120/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Sobald die App, nach der Anmeldung, geöffnet wird, öffnet sich eine Karte die deinen Standort und alle Döner-Läden in der Nähe auf der Karte anzeigt. |
| **Auslöser** | Benutzer möchte nach einen Döner Laden suchen |
| **Ergebnis** | Benutzer finden einen Döner-Laden in der Nähe der eingegebenen Adresse |
| **Akteure** | angemeldeter Benutzer |
| **Eingehende Informationen** | Benutzer Position |
| **Vorbedingungen** | Benutzer muss angemeldet sein, außerdem muss eine Internetverbindung bestehen |
| **Nachbedingung** | Benutzer kann einen Döner-Laden in seiner Umgebung finden |

#### Döner-Karte filtern (/LF0130/)

Man soll die Möglichkeit haben, die Karte nach bestimmten Kriterien zu filtern, um eine genauere Suche zu gestalten.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Mittel | Mittel | SH |
| **Name** | Döner-Karte filtern (/LF0130/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Man soll die Möglichkeit haben, die Karte nach bestimmten Kriterien zu filtern, um eine genauere Suche zu gestalten. |
| **Auslöser** | Benutzer möchte nach einen Döner Laden in seiner Umgebung suchen |
| **Ergebnis** | Benutzer finden einen Döner-Laden in seiner Nähe |
| **Akteure** | angemeldeter Benutzer |
| **Eingehende Informationen** | Filter-Informationen |
| **Vorbedingungen** | Benutzer muss angemeldet sein und einen Filter auswählen |
| **Nachbedingung** | Benutzer kann nach einem bestimmten Muster nach einen Döner-Laden suchen |

* 1. Freunde verwalten
     1. Freunde hinzufügen (/LF0210/)

Mann soll über eine Suchleiste Freunde finden können. Diese werden durch den   
Benutzernamen gefiltert.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Freunde hinzufügen (/LF0210/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Mann soll über eine Suchleiste Freunde finden können. Diese werden durch den  Benutzernamen gefiltert. |
| **Auslöser** | Benützer möchte einen Freund hinzufügen |
| **Ergebnis** | Benutzer fügt einen Freund zu seiner Freundesliste hinzu |
| **Akteure** | angemeldete Benutzer |
| **Eingehende Informationen** | Benutzername |
| **Vorbedingungen** | Benutzer sollte den Namen des Freundes kennen, damit er danach suchen kann |
| **Nachbedingung** | Ein Freund wird gefunden |

#### Freundesliste anzeigen (/LF0220/)

Man soll sich eine Liste anzeigen lassen können, die alle Freunde anzeigt, die Online und Offline sind.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Mittel | Gering | MH |
| **Name** | Freundesliste anzeigen (/LF0220/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Man soll sich eine Liste anzeigen lassen können, die alle Freunde anzeigt, die Online und Offline sind. |
| **Auslöser** | Benutzer möchte sich seine Freunde anzeigen lassen |
| **Ergebnis** | Benutzer sieht alle Freunde, die On-, und Offline sind |
| **Akteure** | angemeldeter Benutzer |
| **Eingehende Informationen** | / |
| **Vorbedingungen** | Benutzer muss Freunde haben und sich diese anzeigen lassen wollen |
| **Nachbedingung** | Benutzer hat einen Überblick über all seine Freunde |

#### Freunde entfernen (/LF0230/)

Man soll aus der Freundesliste Freunde entfernen können.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Mittel | Gering | MH |
| **Name** | Freunde entfernen (/LF0230/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Man soll aus der Freundesliste Freunde entfernen können. |
| **Auslöser** | Benutzer möchte einen Freund entfernen |
| **Ergebnis** | Benutzer entfernt einen Freund |
| **Akteure** | angemeldeter Benutzer |
| **Eingehende Informationen** | Gelöschter Freund |
| **Vorbedingungen** | Benutzer muss Freunde haben, damit er einen entfernen kann |
| **Nachbedingung** | Benutzer kann seine Freundesliste verwalten |

* 1. Freunde verwalten
     1. Freunde per App einladen (/LF0310/)

Man soll nach erfolgreicher Suche, Freunde auf einen Döner einladen können. Man soll den Döner-Stand, Nachrichtentext, Uhrzeit und Datum für das Treffen angeben. Die eingeladenen Freunde bekommen eine Nachricht zugeschickt die alle wichtigen Daten enthält.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Freunde per App einladen (/LF0310/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Man soll nach erfolgreicher Suche, Freunde auf einen Döner einladen können. Man soll den Döner-Stand, Nachrichtentext, Uhrzeit und Datum für das Treffen angeben. Die eingeladenen Freunde bekommen eine Nachricht zugeschickt die alle wichtigen Daten enthält. |
| **Auslöser** | Benutzer möchte einen Freund auf einen Döner einladen |
| **Ergebnis** | Freund erhält eine Einladung für einen Döner |
| **Akteure** | angemeldete Benutzer |
| **Eingehende Informationen** | Adresse, Nachrichtentext, Uhrzeit und Datum |
| **Vorbedingungen** | Man hat sich einen Döner-Laden ausgesuch, und man hat Freunde die man einladen kann |
| **Nachbedingung** | Ein Freund wird eingeladen |

#### Einladungen empfangen und bestätigen (/LF0320/)

Angemeldet: Nachdem dich ein Freund auf einen Döner eingeladen hat, soll sich ein kleines Nachrichten Fenster öffnen und die Einladung anzeigen. Man soll die Anfrage mit einem Button annehmen, oder ablehnen können.  
Abgemeldet:  
Wenn man die App nicht geöffnet hat, soll eine Nachricht am Handy angezeigt wer-den, dass man eine Einladung erhalten hat. Es soll außerdem ein Icon als Eyecat-cher angezeigt werden.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funktion | | Nutzen | Aufwand | Must  Should  Nice |
| UseCase | | Hoch | Mittel | MH |
| **Name** | Einladungen empfangen und bestätigen (/LF0320/) |
| **Art** | Anwendungsfall |
| **Kurzbeschreibung** | Angemeldet: Nachdem dich ein Freund auf einen Döner eingeladen hat, soll sich ein kleines Nachrichten Fenster öffnen und die Einladung anzeigen. Man soll die Anfrage mit einem Button annehmen, oder ablehnen können. Abgemeldet: Wenn man die App nicht geöffnet hat, soll eine Nachricht am Handy angezeigt wer-den, dass man eine Einladung erhalten hat. Es soll außerdem ein Icon als Eyecat-cher angezeigt werden. |
| **Auslöser** | Freund möchte dich auf einen Döner einladen |
| **Ergebnis** | Man erhält eine Einladung |
| **Akteure** | Freund und man selber |
| **Eingehende Informationen** | / |
| **Vorbedingungen** | Benutzer muss dich in seiner Freundesliste haben, und alle wichtigen Informationen in die Einladung eingetragen haben |
| **Nachbedingung** | Man erhält eine Einladung, die man Annehmen oder ablehnen kann |

5.4 Aktivitätsdiagramm



6) Machbarkeit

6.1) TECHNISCHE MACHBARKEIT

6.1.1 Technologie  
Derzeit sind und einige technische Risiken bekannt, die wir folgendermaßen lösen können. Die Daten die die Mitarbeiter an die Datenbank schicken um sie zu ändern, werden sofort aufgenommen sofern der Server online ist. Der Server läuft von 22:00 bis 6:00 in der Früh nicht, um Stromkosten zu sparen. Falls ein Mitarbeiter zu dieser Zeit etwas ändern will, muss er das offline abspeichern und eine Offline-Notiz machen, damit dies auch andere Mitarbeiter sehen können. Falls ein Kunde zu dieser Zeit (22:00-6:00) ein Auto mieten will, kann er eine Offlineanfrage schicken die ab 6:00 bearbeitet wird. Die Webseite wird immer online sein.

6.1.2 Umsetzung

Da es in der Marktwirtschaft keine gleiche Lösungsvariante gibt, wird dieses Produkt neu entwickelt. Da wir aber schon viel Erfahrung mit SQL haben, werden wir damit dieses Produkt gut umsetzen können. Wir könnten auch Microsoft Access verwenden, aber das würde einige große Probleme machen, da sich kaum jemand damit auskennt. OpenOffice könnte auch verwendet werden, aber damit kann man die Datenbank nicht mehr rasch ändern.  
  
6.2)   WIRTSCHAFTLICHE MACHBARKEIT

6.2.1 Personalaufwand

Es muss sehr viel Software implementiert und ein Server konfiguriert werden. Dies braucht einiges an Zeit, aber da wir mit 4 Arbeitern agieren, brauchen wir nur 3 Monate. Die Personalkosten belaufen sich auf 22504 € da sich 2 meiner Mitarbeiter danach mit den Mitarbeitern von der Firma Flitzer zusammensetzten und die Funktionen besprechen. (4\*90\*58€+2\*14\*58€)  
6.2.2 Materialaufwand

Alle nötigen Materialien sind vorhanden und können verwendet werden. Dadurch entstehen nur Stromkosten die sich auf 140 € für 3 Monate, bei durchschnittlichem Verbrauch, belaufen.

6.2.3 Investitionsaufwand

Der Server muss gekauft werden, aber die restliche Software und Hardware ist vorhanden und kann verwendet werden. So kommt man auf 200 € Serverkosten und 10€ pro Monat für die Wartung.

6.2.4 Nutzen

Das System wird von den Mitarbeitern der Firma Flitzer verwaltet werden und die Webseite wird für alle Personen offen sein. Durch diese Aufwände erwartet man sich 10000€ Gewinn.  
 6.2.5 Risikoanalyse

Bei diesem Projekt sind Risiken, wie bei jedem anderen Projekt auch, nicht auszuschließen. Einer dieser Faktoren könnte eventuell eine längere, gesundheitlich bedingte Abwesenheit vom Arbeitsplatz sein. Für diesen Fall werden die übrigen Personen den zu bewältigenden Teil übernehmen. Dieser Fall wird hoffentlich relativ selten eintreten.

Ein großer Faktor bei der Umsetzung der Projektziele kann die Teamarbeit sein. Es ist wichtig, dass die Mitarbeiter zusammenarbeiten, da sonst nur alles verzögert wird.

Da Viren und Malware nie auszuschließen sind müssen wir mehreren Arbeitsgeräten arbeiten regelmäßige Backups gemäß Backup-Policy durchführen. Dieses wird nicht bei uns gespeichert, sondern in einer externen Zentrale.

Falls wir schneller Vorankommen als erwartet, kann schon früher mit der Erklärung der Funktionen begonnen werden.

6.2.6 Projektumfeldanalyse

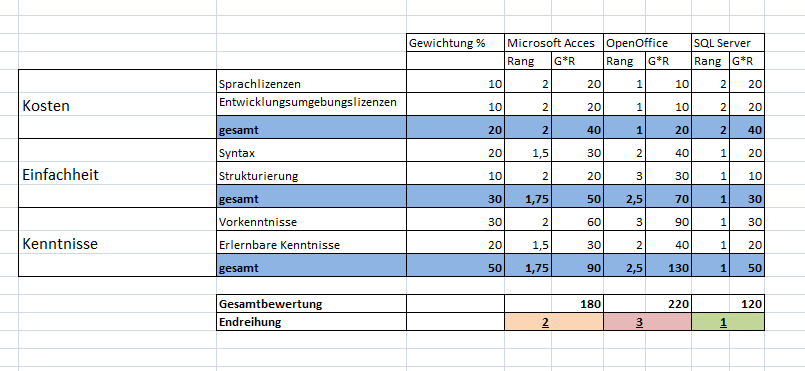
Einbeziehung der zukünftigen User in die Gestaltung der Webseite.

Einbeziehung der Mitarbeiter der Firma Flitzer in den Aufbau der Datenbank.

6.3) Persönliche Machbarkeit

Aufgrund der Nutzwertanalyse ist ersichtlich, ob das Know-how des Teams für die bevorzugte Lösungsvariante ausreichend ist. Weiteres werden andere Möglichkeiten damit gegenübergestellt.

6.3.1 Nutzwertanalyse



Hier werden 3 verschiedene Varianten mit 3 großen Kriterien dargestellt. Wir haben uns für die Kriterien Kosten, Einfachheit und Kenntnisse entschieden, da diese sehr wichtige Punkte für die Fertigstellung einer Datenbank sind. Die Kosten sind bei OpenOffice am geringsten. SQL Server überzeugt bei der Einfachheit und bei den Kenntnissen, da unsere Mitarbeiter jahrelange Erfahrung damit haben. MicrosoftAccess ist bei allen Punkten im Durchschnitt. Letztlich kommt bei der Gesamtbewertung SQL Server auf den ersten Platz und damit haben wir das geeignetste Werkzeug für unser Projekt gefunden.

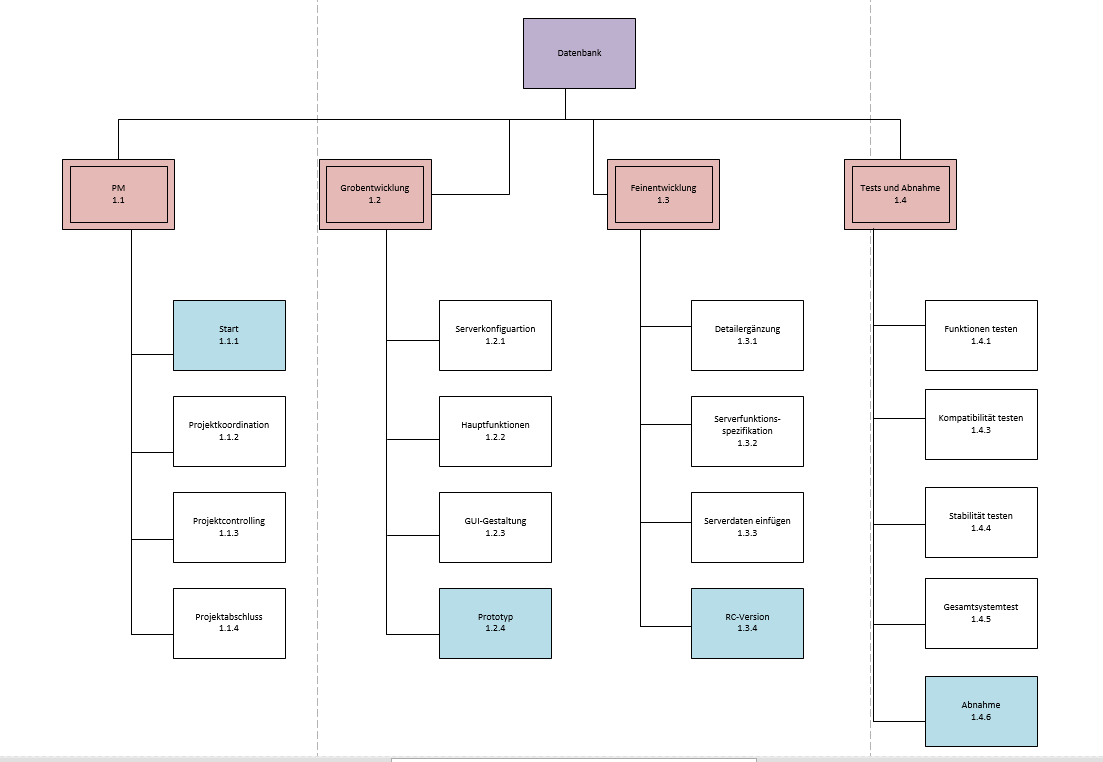
7)    Projektorganisation

Der Grund für dieses Projekt, ist der Auftrag von Herr prof. Rafeiner-Magor.  
Der Projektleiter ist Matthias Tieber und das Team besteht noch aus 3 weiteren Programmierern. Die Programmierer sind Armin Freudenthaler, Jan Langela und Simon Appel.



8)    Projektplanung

8.1) Projektstrukturplan



8.2) Meilensteinplan

|  |  |  |
| --- | --- | --- |
| Meilenstein | Deliverable | Datum |
| Start | Dokumentationen mit Informationen | 01.12.2016 |
| Prototyp | Prototyp der Datenbank mit wichtigen Funktionen | Voraussichtlich  Am 10.02.2017 |
| RC-Version | Visualisierung mit richtiger GUI und allen Funktionen | Voraussichtlich  Am 22.04.2017 |
| Abnahme | Fertiges Produkt einsatzbereit | Am 31.05.2017 |

9)    Management Summary:

Die Firma Flitzer braucht eine neue Datenbank und da sie größere Anforderungen haben müssen wir vieles neu implementieren.

Die Kosten belaufen sich auf 22874€. (22504€ Personalkosten + 140€ Stromkosten + 230€ Serverkosten)

Wie bei der Nutzwertanalyse gezeigt, ist SQL Server die vorgeschlagene Technologie.

Das komplette Projekt wird ca. ein halbes Jahr benötigen. Die genauen Termine sind in der Meilensteinplanung ersichtlich.

Alles in Allem kann man sagen, dass das Projekt wirklich gut durchführbar ist, da unsere Mitarbeiter darauf geschult sind solche Projekte zu meistern. Dies wäre nicht unser erstes Projekt mit dieser Größe und wir wissen, dass wir es schaffen.