**Pflichtenheft**

**für**

**das Projekt**

**Helpdesk mit Ticket System**

**Gruppe B**

**Teammitglieder:**

**Matthias Feil**

**Kevin Klose**

**Regina Richter**

Inhaltsverzeichnis

[1. Ausgangssituation 3](#_Toc118905127)

[2. Software-Architektur 3](#_Toc118905128)

[3. UML-Diagramme und Auswahl technischer Komponenten 4](#_Toc118905129)

[4. Wireframes 5](#_Toc118905130)

[5. Konkrete Aufgaben 7](#_Toc118905131)

[5.1 Aus der Sicht des Endnutzers 7](#_Toc118905132)

[5.1.1 Innerhalb von 15 Minuten kann ein Ticket erstellt werden 7](#_Toc118905133)

[5.1.2 Auf das Ticketsystem kann jederzeit innerhalb des Intranets zugegriffen werden 7](#_Toc118905134)

[5.1.3 Es gibt Vorauswahlmöglichkeiten in Form von Drop-Down-Feldern 7](#_Toc118905135)

[5.1.4 Es gibt ein Freitextfeld um das Problem zu schildern 7](#_Toc118905136)

[5.1.5 Es gibt ein Feld zum Hochladen von Screenshots 7](#_Toc118905137)

[5.1.6 Der richtige IT-Mitarbeiter wird mittels der Schlagwörter bestimmt 7](#_Toc118905138)

[5.2 Aus der Sicht des IT-Mitarbeiters 7](#_Toc118905139)

[5.2.1 7](#_Toc118905140)

[5.3 IT-Abteilung **Fehler! Textmarke nicht definiert.**](#_Toc118905141)

[5.3.1 Rollenbeschreibung **Fehler! Textmarke nicht definiert.**](#_Toc118905142)

[5.3.2 User Stories **Fehler! Textmarke nicht definiert.**](#_Toc118905143)

[5.4 Abteilungsleiter **Fehler! Textmarke nicht definiert.**](#_Toc118905144)

[5.4.1 Rollenbeschreibung **Fehler! Textmarke nicht definiert.**](#_Toc118905145)

[5.4.2 User Stories **Fehler! Textmarke nicht definiert.**](#_Toc118905146)

[6. Constraints /Nicht-Funktionale Anforderungen 8](#_Toc118905147)

[7. Abgrenzung, was das Ticketsystem nicht können muss 8](#_Toc118905148)

[8. Ziel **Fehler! Textmarke nicht definiert.**](#_Toc118905149)

1. Ausgangssituation

Viele kleinere Behörden haben kein eigenes Ticketsystem für IT-Angelegenheiten. Wenn bei Mitarbeitern Probleme auftreten oder diese Fragen haben, müssen diese dann versuchen jemanden aus der IT-Abteilung telefonisch oder per E-Mail zu erreichen. Oftmals ist hierbei schon die erste Herausforderung den richtigen Ansprechpartner zu finden. Zudem kann es vorkommen, dass diese Anfragen im normalen Tagesgeschäft untergehen und der Mitarbeiter sich mehrmals melden muss, damit sein Anliegen bearbeitet wird. Dies kann auf beiden Seiten zu Spannungen und einer geringeren Produktivität führen, welche mit einem Ticketsystem für den Erstkontakt vermeidbar wären.

1. Software-Architektur

Es wird eine klassische drei-Schichten-Architektur verwendet. Diese ist den Entwicklern gut bekannt und für unsere Zwecke ausreichend.

Presentation Layer: HTML, CSS, Javascript Webbrowser (z.B. Firefox)

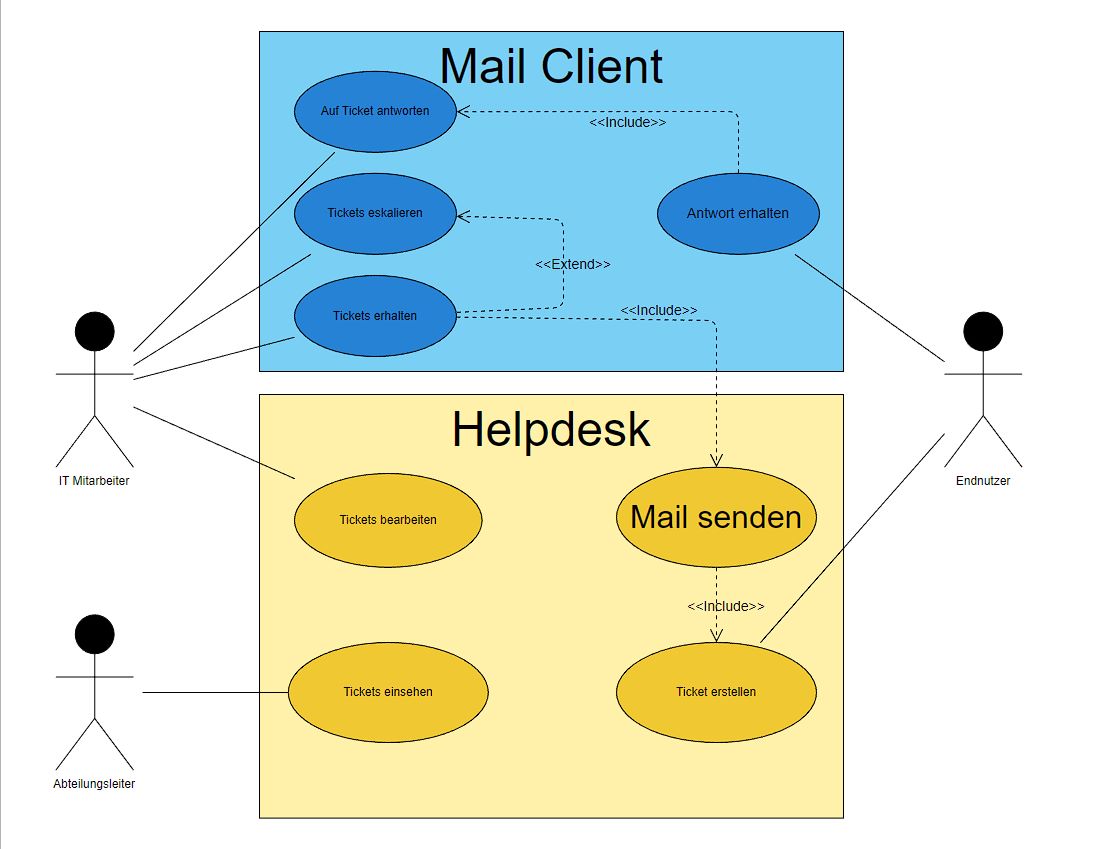
Applikation Layer: PHP Apache Server

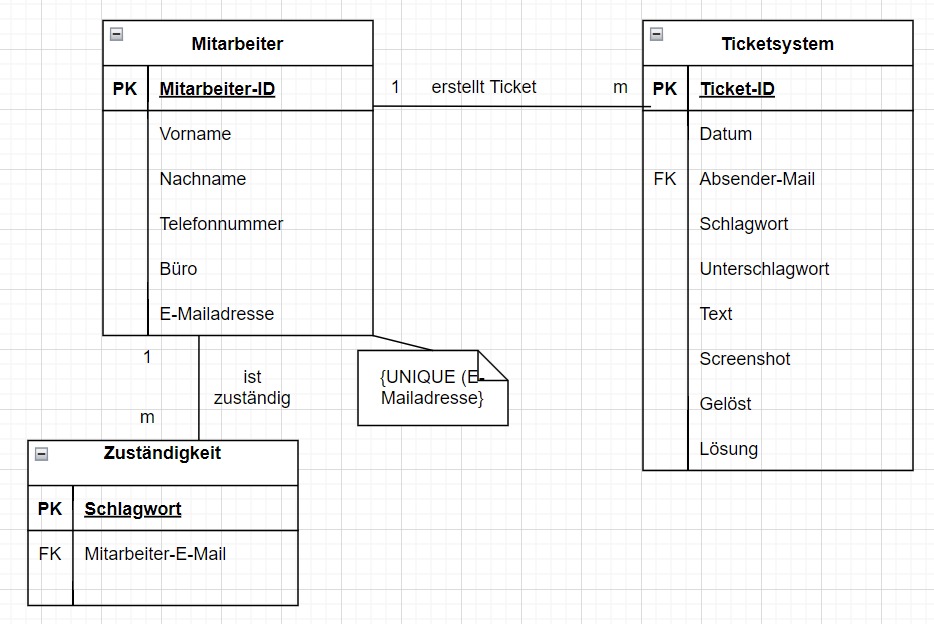
Data Layer: SQL MySQL

Mit diesen Frameworks lässt sich XAMPP verwenden. Damit kann dann lokal auf den eigenen Rechnern entwickelt und getestet werden, wodurch die Beschaffung eines Servers wegfällt.

* Die Daten werden dauerhaft auf der MySQL-Datenbank gespeichert, da sie auch für statistische Zwecke ausgewertet werden sollen.
* Um sämtliche Kontaktinformationen des Anfragestellers zu erhalten, wird das Active-Directory des Unternehmens eingebunden. Hier wird zu Testzwecken eine einfache Datenbank verwendet. Zudem wird ein E-Mail-Client benötigt um die E-Mail-Anfragen zu empfangen.
* Der Benutzer verwendet eine Weboberfläche um seine Anfrage zu stellen.
* Das Projekt ist nur für den deutschsprachigen Raum ausgelegt. Eine Mehrsprachigkeit ist nicht vorgesehen, da sie in der Implementierung zu aufwendig ist und kaum genutzt wird, denn in deutschen Behörden werden Deutschkenntnisse vorausgesetzt und die Amtssprache ist Deutsch.
* Da die Anfragen nur über das Intranet gestellt werden, für dessen Zugang man sich vorher Authentifizieren muss, werden die Daten unverschlüsselt übertragen.
* Eine Protokollierung und Ausnahmebehandlung ist nicht vorgesehen, da sie zu aufwendig sind. Lediglich die E-Mail-Adresse wird überprüft, ob sie in dem Active-Directory vorliegt, damit eine E-Mail versendet werden kann.

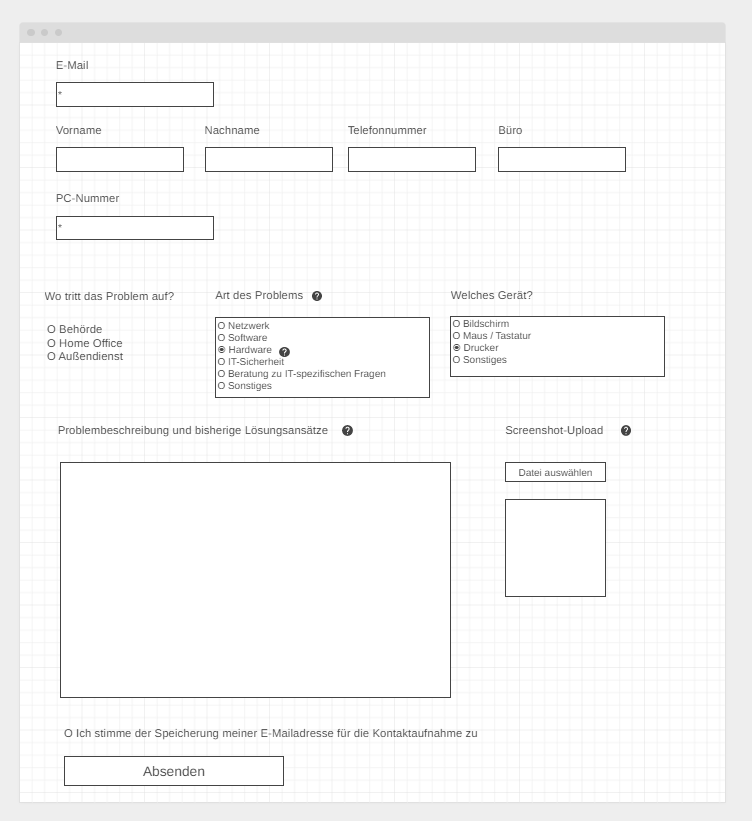
1. Use-Case-Diagramm und Datenbank-Schema





Wireframes

* 1. Ticket erstellen



<https://wireframe.cc/mbCan1>

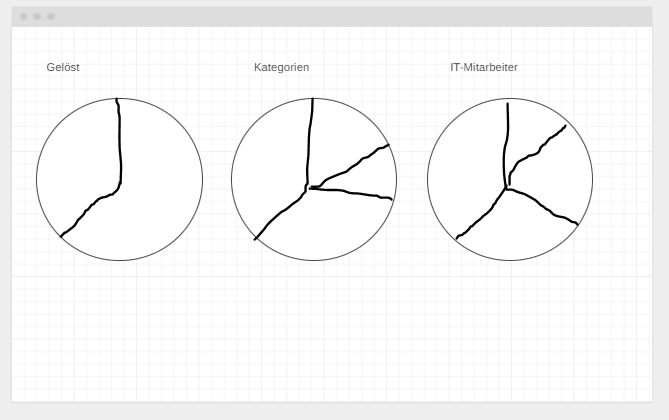
* 1. Ticket lösen

Ein Bild, das Text, Tisch enthält.

Automatisch generierte Beschreibung

<https://wireframe.cc/wmx8q9>

* 1. Statistik einsehen



<https://wireframe.cc/cN9OoE>

1. Konkrete Aufgaben
   1. Aus der Sicht des Endnutzers
      1. Innerhalb von 15 Minuten kann ein Ticket erstellt werden

Die Ticket Erstellung soll mit wenigen Klicks und ohne Vorwissen für den normalen Mitarbeiter möglich sein.

* + 1. Auf das Ticketsystem kann jederzeit innerhalb des Intranets zugegriffen werden

Der Webserver muss 24/7 aus dem Intranet zugänglich sein.

* + 1. Mit Eingabe der E-Mail werden die Kontaktdaten automatisch ergänzt

Dies geschieht mit einer Datenbank-Abfrage nach einem OK-Button.

* + 1. Es gibt Vorauswahlmöglichkeiten in Form von Drop-Down-Feldern

Die Vorauswahlmöglichkeiten sollen alltägliche Schlagwörter sein.

**Netzwerk**:  
**Software**: Office, Adobe, Sonstiges  
**Hardware**: PC, Bildschirm, Drucker, Handy, Sonstiges  
**IT-Sicherheit**:   
**Beratung zu IT-spezifischen Fragen**:  
**Sonstiges**:

* + 1. Es gibt ein Freitextfeld, um das Problem zu schildern

Es gibt ein Hinweis-Button, der eine Formulierungshilfe gibt

* + 1. Es gibt ein Feld zum Hochladen von Screenshots
    2. Der richtige IT-Mitarbeiter wird mittels der Schlagwörter bestimmt

Netzwerk, Software, Adobe, Office, Hardware, PC, Bildschirm, Drucker, Handy, IT-Sicherheit, Beratung zu IT-spezifischen Fragen, Sonstiges

* 1. Aus der Sicht des IT-Mitarbeiters
     1. Die empfangene E-Mail kommt von einer Funktionsmail-Adresse
     2. Die E-Mail enthält alle wichtigen Informationen

Die Felder aus dem Ticket-Formular werden formatiert und in die E-Mail hineingeschrieben.

* + 1. Folgende Felder sollen implementiert sein
* Art des Problems
* Auf welcher Hardware?
* wo tritt der Fehler auf? (Behörde, Homeoffice, Außendienst)
* versuchte Lösungsansätze
  + 1. In der E-Mail befinden sich die Kontaktdaten des Benutzers

Zu den Kontaktdaten zählen: Vorname, Nachname, E-Mailadresse, Telefonnummer, Büroraum  
Diese Daten werden aus einer Datenbank automatisch ergänzt.

* + 1. Über einen generierten Link in der E-Mail wird die Lösung mitgeteilt

Über den Link kommt man auf eine Seite mit der richtigen Ticketnummer. Darüber kann der Lösungsvorschlag eingegeben werden und eine E-Mail wird automatisch an den Ticket-Ersteller geschickt

* 1. Aus der Sicht des IT-Abteilungsleiters
     1. Das Ticketsystem soll jederzeit erreichbar sein
     2. Es gibt eine statistische Auswertung der Ticket-Daten

Alle Tickets werden in einer Datenbank gespeichert mit Datum, wann das Ticket erstellt wurde. Es wird ausgewertet nach den Lösungsstatus, Problembereich, Empfänger.

* + 1. Für das Ticketsystem ist keine Schulung nötig

Orientierung an Kontaktformularen aus dem Internet und Hinweise bei schwierigen Feldern, damit Mitarbeiter ohne IT-Kenntnisse diese ausfüllen können.

* + 1. Es werden gängige Programmiersprachen verwendet

JavaScript, PHP, CSS und HTML. Auf eine gute Dokumentation des Codes wird geachtet.

1. Constraints /Nicht-Funktionale Anforderungen

* Robustheit: Ticketsystem soll bei falschen Eingaben nicht abstürzen
* Verfügbarkeit: Zugriff soll von Überall über das Intranet möglich sein
* Änderbarkeit: Quellcode soll ausführlich kommentiert sein, um nachträgliche Veränderungen vornehmen zu können
* Zuverlässigkeit: Ticketsystem muss den Mitarbeitern zu jeder Uhrzeit (24x7) zur Verfügung stehen
* Benutzbarkeit: Orientierung bei Ticketeingabe an Kontaktformularen
* Sicherheit: Konformität im Umgang mit personenbezogenen Daten

nach Art. 5 Abs. 1 DSGVO

1. Abgrenzung, was das Ticketsystem nicht können muss

Dieses Ticket-System soll nur dabei helfen den Erstkontakt zwischen Benutzern und der IT-Abteilung herzustellen. Es ist nicht für andere Bereiche in einer Behörde z.B. Fuhrpark, Hausmeister, etc. gedacht.