*TAU INF202 Software Engineering*

*Individuelles Projekt*

***Pflichtenheft***

Projektdokumentation

Version: 2023.04.07

Status: Entwurf

Projektname: Autovermietung Application

Verantwortliche/r:

Buğra BALCI, e190503026@stud.tau.edu.tr

Fırat Ulaş GÜNEŞ, e220503070@stud.tau.edu.tr

Stakeholder: Ömer Karacan, [omer.karacan@tau.edu.tr](mailto:omer.karacan@tau.edu.tr)

Fulya Yenilmez, fulya.yenilmez@tau.edu.tr

**Dokumentenverwaltung**

**Dokument-Historie**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Status \*)** | **Datum** | **Verantwortlicher** | **Änderungsgrund** |
| v1.0 | Entwurf | 04.04.2023 | Buğra BALCI  Fırat Ulaş GÜNEŞ |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Inhaltsverzeichnis**

[**1. Einleitung 4**](#_1fob9te)

[**2. Ausgangssituation und Ziele 4**](#_2et92p0)

[**3. Gesamtarchitektur 6**](#_3dy6vkm)

[**4. Funktionale Anforderungen 7**](#_1t3h5sf)

[**5. Nichtfunktionale Anforderungen 10**](#_17dp8vu)

[**6. Abnahmekriterien 11**](#_26in1rg)

[**7. Projekt Meilensteine 11**](#_lnxbz9)

[**8. Referenzen 11**](#_35nkun2)

# Einleitung

Dieses Dokument dient zur Erläuterung der bereitgestellten Anwendung. Alle notwendigen Informationen und Funktionen werden erklärt (Käufer - Händlerinformationen, Fahrzeuginformationen, Serviceinformationen). Der Ausgangszustand des Projekts und die Gründe für die Auswahl dieses Projekts, die Architektur des Systems und seiner Subsysteme, Benutzer und notwendige Kommunikationsschnittstellen werden erläutert. Daneben werden auch User Stories, Use Cases und technische Anforderungen genannt.

# Ausgangssituation und Ziele

Mit den steigenden Autopreisen ist der Autokauf für viele zu einer schwierigen Situation geworden, aber die Menschen haben möglicherweise das Bedürfnis, aus dringenden oder willkürlichen Gründen mit dem Auto von Ort zu Ort zu fahren, weshalb beschlossen wurde, eine Autovermietungsanwendung für alle zu erstellen benutzen.

**Einleitung**

In diesem Abschnitt wird die Problemlösung unseres Projekts beschrieben, dh die Funktionalität welcher Altersgruppen im Allgemeinen oder welche Art von Personen die Anwendung anspricht, die Infrastruktur, auf der dieses System funktioniert, und die Bedingungen für die Verwendung der Anwendung besprochen.

**Problemstellung (Funktionalität)**

Durch die Verwendung dieses Systems kann jede Person ein Auto auf die am besten geeignete Weise mieten oder vermieten, und die negativen Auswirkungen, die das Fehlen eines eigenen Fahrzeugs auf die Menschen hat, können zu einem erschwinglichen Preis gelöst werden. Außerdem ist es für unser Fahrzeug besser, eine Einnahmequelle zu sein, als bei Nichtgebrauch still zu stehen.

**Stakeholder (Anwender):**

Mit diesem System wird es viele Anwendungsbenutzer geben, von Angestellten bis hin zu Mietern und Mietern. Beispielsweise sind die Personen, die die Anwendung verwenden werden, unten aufgeführt.

* Kunden: Privatpersonen oder Unternehmen, die ein Fahrzeug für den persönlichen oder beruflichen Gebrauch mieten möchten.
* Autovermieter: Unternehmen, die Autos an ihre Kunden vermieten.
* Mitarbeiter von Mietwagenunternehmen: Mitarbeiter von Mietwagenunternehmen, die den Anmietvorgang verwalten, einschließlich Reservierungen, Check-in und Check-out.
* Wartungs- und Reparaturtechniker: Personal, das für die Wartung und Reparatur von Mietfahrzeugen verantwortlich ist.
* Administratoren: Personen, die für die Verwaltung der Anwendung verantwortlich sind, z. B. das Hinzufügen neuer Autos zum System, die Verwaltung von Benutzerkonten und die Abwicklung von Zahlungsvorgängen.
* Geschäftspartner: Unternehmen, die mit Mietwagenfirmen zusammenarbeiten, um zusätzliche Dienstleistungen wie Versicherungen oder Pannenhilfe anzubieten.

**Systemumfeld (Einsatzumgebung)**

Die für diese Anwendung bereitzustellende Infrastrukturumgebung erfordert einen leistungsstarken und redundanten Server, damit die Anwendung funktioniert. Die Anwendung benötigt auch eine Datenbank, um Daten wie Kundeninformationen, Fahrzeuginformationen, Reservierungen und Zahlungen zu speichern. Es ist ein sicheres System erforderlich, da die Zahlungen über die Anwendung erfolgen. Abgesehen davon sollte die Anwendung im Falle eines Datenverlusts oder Systemausfalls für Sicherungs- und Wiederherstellungsvorgänge bereit sein.

**Rahmenbedingung (Einschränkungen)**

Die einzige Voraussetzung für die Nutzung unserer Anwendung ist, über 20 Jahre alt zu sein und einen Führerschein zu besitzen. Abgesehen von diesen Einschränkungen sind einige andere Bedingungen wie folgt:

Projektdurchführung:

* Benutzerfreundliche Oberfläche: Die Anwendung muss über eine benutzerfreundliche Oberfläche verfügen, damit Kunden Fahrzeuge einfach finden, Reservierungen vornehmen und bezahlen können.
* Autovermietung: Die App muss bestimmte Schritte ausführen, um den Autovermietungsprozess der Kunden zu automatisieren. Diese Schritte umfassen das Erstellen einer Reservierung, die Kundenauthentifizierung, die Abholung des Fahrzeugs und die Rückgabe.
* 3.Mobile Kompatibilität: Die App muss mobilkompatibel sein, um reibungslos auf den mobilen Geräten der Kunden zu laufen.

Systemanwendung:

* 1.Sicherheit: Die App muss Sicherheitsmaßnahmen auf hohem Niveau ergreifen, um die persönlichen Daten und Finanzinformationen der Kunden zu schützen.
* 2. Reservierungsverwaltung: Die Anwendung muss bestimmten Prozessen folgen, um Autoreservierungen zu verwalten. Diese Prozesse umfassen das Finden eines Fahrzeugs, das Bestätigen einer Reservierung, das Informieren von Kunden und das Zurückgeben eines Fahrzeugs.
* 3. Fahrzeugverfolgung: Die App muss GPS-Technologie verwenden, um den Standort von Fahrzeugen zu verfolgen und Fahrzeuge zu finden, die den Kunden am nächsten sind.

Betriebsumgebung:

* 1. Hohe Leistung: Die Anwendung benötigt eine leistungsstarke Betriebsumgebung, um die Anforderungen mehrerer Benutzer gleichzeitig zu erfüllen.
* 2. Datenspeicherung: Die Anwendung benötigt ein Datenbanksystem, um Kunden- und Fahrzeugdaten sicher zu speichern.
* 3. Skalierbarkeit: Die Anwendung sollte eine Skalierbarkeitsfunktion bieten, damit die Systemleistung nicht abnimmt, wenn die Anzahl der Benutzer zunimmt.
* 4.Backup: Die Anwendung muss eine Backup-Funktion im Falle eines Datenverlusts oder Systemabsturzes bieten.
* 5. Integration: Die Anwendung muss in der Lage sein, sich mit anderen Betriebsumgebungskomponenten wie Zahlungssystemen, GPS-Systemen, Fahrzeugverfolgungssystemen zu integrieren.

**Ziele (Lösung)**

Mit unserer Anwendung werden wir in der Lage sein, eine qualitativ breite Palette von Tools, eine hohe Benutzerfreundlichkeit, Sicherheit, Kundenunterstützung und eine benutzerfreundliche Oberfläche zu erreichen. Wir werden in der Lage sein, den quantitativen Anstieg der Autovermietungen, die steigende Zahl der Kunden und die steigende Zahl der Mieter zu erreichen, indem wir den Kunden ein besseres Benutzererlebnis bieten.

# Gesamtarchitektur

Jede Benutzeranforderung hat nur eine Funktion. Benutzeranforderungen implementieren nur Schnittstellen, die die von ihnen verwendeten Funktionen enthalten. Dies vermeidet unnötige Codeduplizierung und Abhängigkeiten. Erstellte Unterklassen können Oberklassen ersetzen und das gleiche Verhalten zeigen. Eine nach diesen Prinzipien gestaltete Anwendungsarchitektur kann die Zukunftsfähigkeit der Anwendung erhöhen und die Anpassung an zukünftige Änderungen erleichtern.

**Einleitung**

Mietwagenanwendungen sind eine Branche, die in den letzten Jahren zunehmend an Popularität gewonnen hat. Diese Anwendungen bieten eine Plattform, auf der Benutzer ein Auto nach ihren Bedürfnissen mieten und ihre Anmietungen online verwalten können.

Die Gesamtarchitektur einer Autovermietungsanwendung besteht aus drei Hauptkomponenten: einer Benutzeroberfläche, einer Datenbank und einem Backend-Server. Die Benutzeroberfläche ist die Web- oder mobile Anwendung, in der Benutzer ihre Mietwagentransaktionen durchführen. Eine Datenbank ist ein System, das Daten von Benutzern und Fahrzeugen speichert. Ein Backend-Server ist eine Softwarekomponente, die Anfragen von der Benutzeroberfläche verarbeitet und die erforderlichen Informationen von der Datenbank erhält und auf die Benutzeroberfläche antwortet.

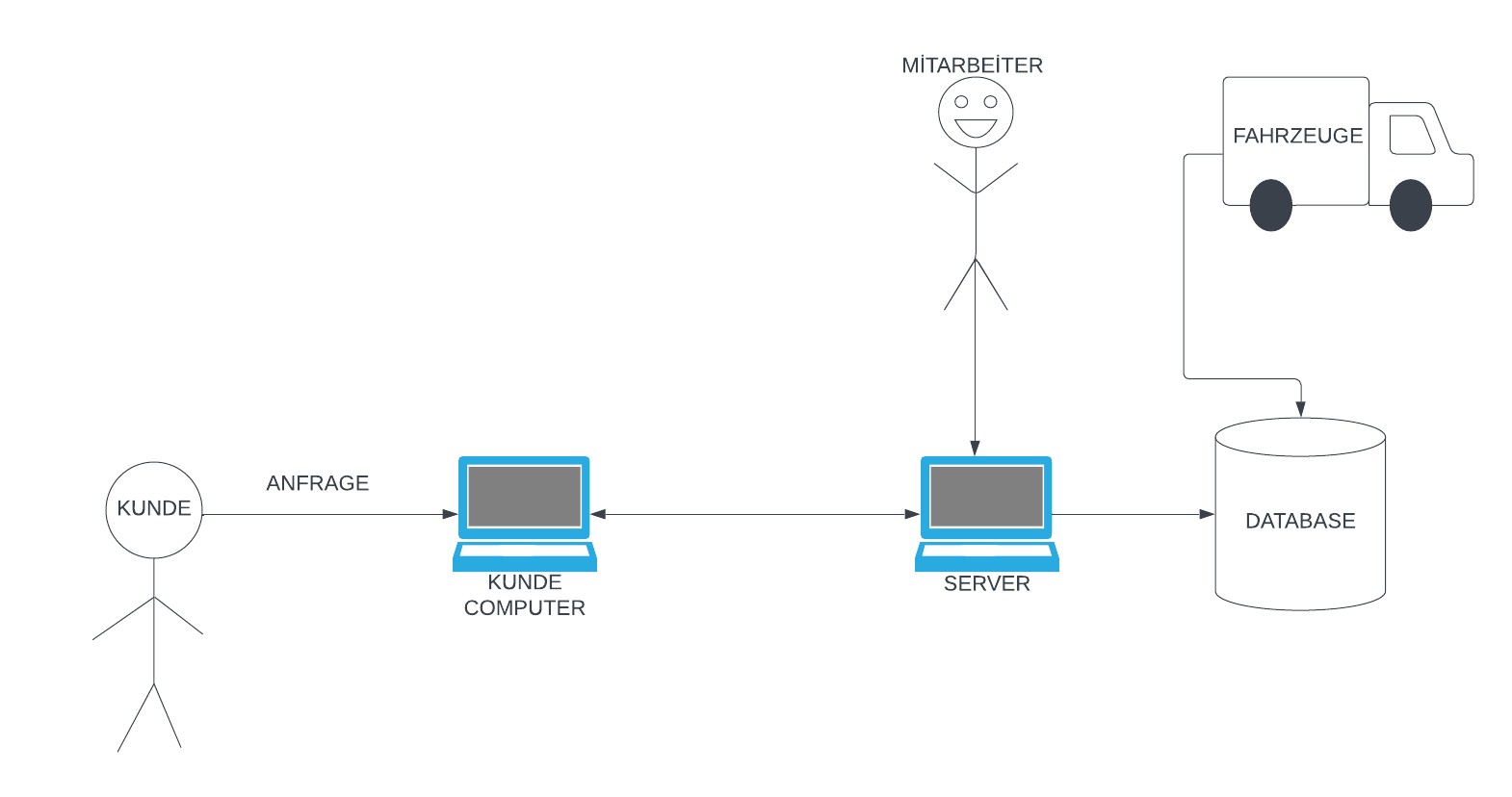
Die Benutzeroberfläche stellt eine Plattform bereit, auf der Benutzer Vorgänge wie das Suchen von Fahrzeugen, Vergleichen von Preisen, Buchen von Mieten und Ausführen von Zahlungen ausführen können. Die mobile App ist eine beliebte Option, da sie den Benutzern mehr Flexibilität beim Zugriff auf und bei der Anmietung von Fahrzeugen bietet.

Eine Datenbank ist ein System, das Daten von Benutzern und Fahrzeugen speichert. Diese Daten können Informationen wie Registrierungsinformationen der Benutzer, Zahlungsinformationen, Anmietungen und Fahrzeugmerkmale enthalten. Die Sicherheit und Vertraulichkeit dieser Daten ist wichtig für den Erfolg der Anwendung.

Ein Backend-Server ist eine Softwarekomponente, die die Anfragen von Benutzern verarbeitet. Dieser Server ruft nach Erhalt der Benutzeranfrage die erforderlichen Informationen aus der Datenbank ab und antwortet der Benutzerschnittstelle. Über diesen Server wird auch der Zahlungsverkehr abgewickelt.

Die Gesamtarchitektur einer Autovermietungsanwendung ist ein System, in dem diese drei Komponenten miteinander interagieren. Während Benutzer die Lease-Operationen von der Benutzeroberfläche aus durchführen, führen der Back-End-Server und die Datenbank die notwendigen Operationen für den erfolgreichen Abschluss dieser Operationen durch. Auf diese Weise können Nutzer einfacher, schneller und sicherer ein Auto mieten.

***Gesamtarchitektur***

**

Figure

***Komponente***

Kunden können ihre Informationen über den Server von ihren eigenen Computern in der Datenbank speichern.

Kunden können ihre Fahrzeuge von ihren eigenen Computern über den Server in der Datenbank speichern.

Kunden können die aus der Datenbank über den Server registrierten Fahrzeuge von ihren eigenen Computern aus einsehen.

Mitarbeiter lösen Kundenfragen über den Server des Kunden.

Mitarbeiter können Anwendungsupdates auf dem Server des Kunden vornehmen.

# Funktionale Anforderungen

Zu den funktionalen Anforderungen einer Autovermietungsanwendung gehören:

1- Benutzerregistrierung: Benutzern wird ein Registrierungsformular zur Verfügung gestellt, um sich für die Anwendung zu registrieren.

2- Autosuche: Benutzern steht eine Suchfunktion zur Verfügung, mit der sie nach Mietdaten, Standorten und Merkmalen des zu mietenden Fahrzeugs suchen können.

3- Reservierungsverwaltung: Ein Reservierungsverwaltungssystem wird den Benutzern angeboten, um die von ihnen ausgewählten Fahrzeuge für ein bestimmtes Datum und einen bestimmten Zeitraum zu reservieren.

4- Preisgestaltung: Preisoptionen werden für von Benutzern ausgewählte Fahrzeuge angeboten. Diese Preise können auf Faktoren wie Mietdauer, Standort, Fahrzeugmerkmalen und ausgewählten Versicherungsoptionen basieren.

5- Zahlungsmanagement: Benutzern wird ein sicheres Zahlungssystem zur Verfügung gestellt, um ihre Zahlungen zu leisten.

6- Fahrzeuglieferung und -rückgabe: Benutzern wird ein Verwaltungssystem zur Verfügung gestellt, um Fahrzeuge an einem bestimmten Ort zu erhalten und zurückzugeben.

7- Benutzerverwaltung: Es wird ein Benutzerverwaltungssystem angeboten, in dem Benutzer ihre Profilinformationen und ihren Reservierungsverlauf verwalten können.

8- Fahrzeugstatus: Ein Managementsystem wird bereitgestellt, um den Status des Fahrzeugs vor und nach der Anmietung zu melden, um die Prozesse im Zusammenhang mit der Reinigung und Kontrolle der Fehlfunktionen des Fahrzeugs zu verfolgen.

9- Kundendienst: Ein Kundendienst wird angeboten, um die Fragen der Benutzer zu beantworten und Lösungen für ihre Probleme zu finden.

Diese Anforderungen sind notwendig, um die Kernfunktionalität der Anwendung bereitzustellen. Der App-Eigentümer kann weitere Funktionen hinzufügen oder vorhandene Funktionen anpassen.

**Einleitung**

In diesem Abschnitt wird erklärt, welche Funktionen in der entwickelten Application enthalten sein sollten.

**UI Use Cases**

Die Benutzeroberfläche sollte leicht verständlich, einfach und nützlich sein.

**/Use Case-1/** Benutzern wird die Möglichkeit geboten, die Website zu durchsuchen, ohne sich auf der Website zu registrieren, um eine schnellere Nutzung zu ermöglichen.

**/Use Case-2/** Benutzer können einfach den Status ihrer Verbindung sehen.

**/Use Case-3/** Ein schneller Mietwagen-Button für diejenigen, die nur ihr Auto vermieten möchten.

**/Use Case-4/** Übersichtliche Anwendung. Es werden nur Autos und ihre Merkmale angezeigt. Möchte man sich das Fahrzeugdetail ansehen, wird der Nutzer auf eine andere Seite geleitet.



Figure : Startseite

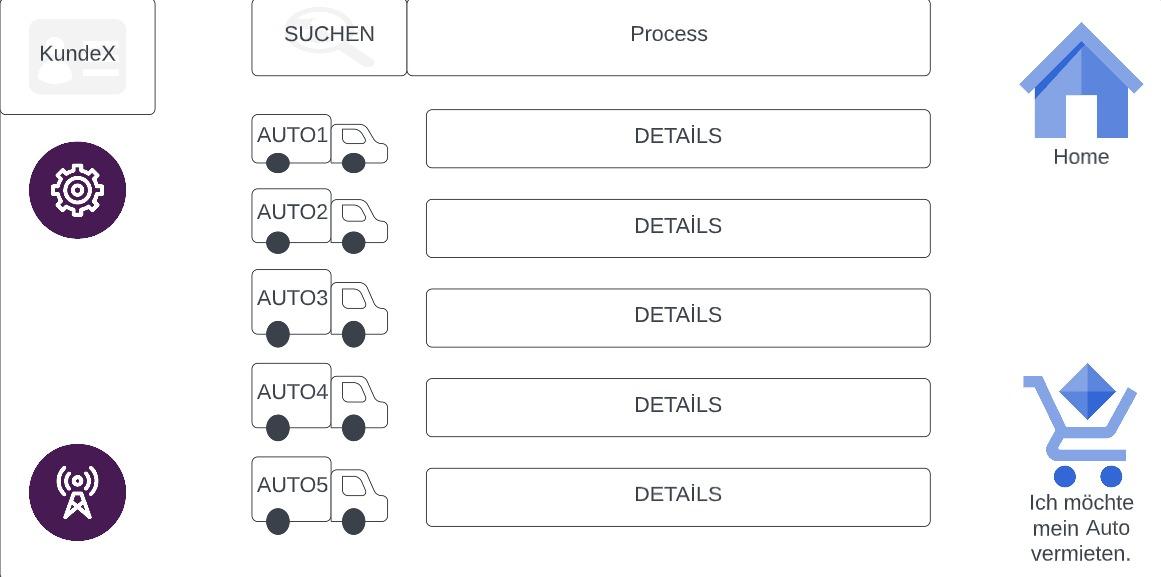


Figure : Suchseite

**API Use Cases**

Eine API ist eine Schnittstelle, die es einer Anwendung ermöglicht, mit anderen Anwendungen oder Plattformen zu interagieren. Eine gute API sollte so gestaltet sein, dass Entwickler sie leicht verstehen und verwenden können.

**Technischen und fachliche Anforderungen**

Die fachlichen und fachlichen Anforderungen einer Bewerbung beziehen sich auf die fachlichen und fachlichen Fähigkeiten und Ressourcen, die für die Bewerbung erforderlich sind. Einige grundlegende technische und fachliche Anforderungen sind:

1- Programmiersprachen: C#, C, Java

2- Datenbank: SQL

3- Entwicklungsumgebung: Visual Studio Code

4- UI/UX-Design: Für die Benutzeroberfläche und Benutzerfreundlichkeit der Anwendung sollte ein geeignetes UI/UX-Design erstellt werden.

**Datenmodel**

Das Datenmodell definiert die Daten der Anwendung und die Beziehungen zwischen diesen Daten. In einer Autovermietungsanwendung werden die Beziehungen zwischen Daten wie Benutzern, Fahrzeugen, Fahrten und Zahlungen definiert. Dieses Datenmodell stellt sicher, dass Daten korrekt gespeichert, verwaltet und abgerufen werden.

# Nichtfunktionale Anforderungen

Nicht-funktionale Anforderungen beinhalten oft Faktoren wie eine benutzerfreundliche Oberfläche, Leistung, Sicherheit, Kompatibilität. Die Anwendung verfügt über eine benutzerfreundliche Oberfläche, die den Benutzern ein besseres Erlebnis bietet. Bei Benutzern beliebte Funktionen sind anpassbare Themen, einfache Navigation, intuitive Benutzeroberfläche und reaktive Designs.

Die schnelle und reibungslose Ausführung der Anwendung ermöglicht Benutzern eine bessere Erfahrung bei der Verwendung der Anwendung. Zu den Leistungsanforderungen gehören schnelle Startzeiten, Seitenladezeiten, reduzierte Latenz und schnelle Datenverarbeitungsgeschwindigkeit.

Die Sicherheit der Benutzerdaten ist eine der wichtigsten Anforderungen an eine Anwendung. Zu den Sicherheitsanforderungen können Verschlüsselung, Benutzerauthentifizierung, Datenschutz, Sitzungsverwaltung und Maßnahmen zur Verhinderung von Eindringlingen gehören.

Kompatibilitätsanforderungen für verschiedene Plattformen und Geräte ermöglichen es der App, mehr Benutzer zu erreichen. Kompatibilitätsanforderungen können eine App umfassen, die für verschiedene Browser, Betriebssysteme und Geräte optimiert ist.

**Einleitung**

Dieser Abschnitt beschreibt nichtfunktionale Anforderungen und deren Inhalte.

**Nicht-funktionale Anforderungen an die Systemarchitektur (Architekturmuster, Deployment)**

**/SYS-1/** Die Anwendung sollte in der Lage sein, den Anforderungen einer wachsenden Anzahl von Benutzern und Transaktionen gerecht zu werden. Skalierbarkeitsanforderungen können Load Balancing, horizontale Skalierung und vertikale Skalierung beinhalten.

**/SYS-2/** Die Anwendung sollte immer verfügbar sein, um den Benutzern ein reibungsloses Erlebnis zu bieten. Verfügbarkeitsanforderungen können hochverfügbare Architekturen wie Failover-Cluster, Lastverteilung und geografisch verteilte Datenzentren beinhalten.

**/SYS-3/** Die Anwendung sollte sicher sein, um Benutzerdaten und Transaktionsdaten vor unbefugtem Zugriff zu schützen. Sicherheitsanforderungen können Verschlüsselung, Zwei-Faktor-Authentifizierung, Berechtigungs- und Zugriffskontrollen und regelmäßige Sicherheitsaudits umfassen.

**Nicht-funktionale Anforderungen an die Entwicklungsumgebung**

**/DEV-1/** Visual Studio Code wird als Entwicklungsumgebung verwendet.

**Nicht-funktionale Anforderungen an die Entwicklungswerkzeuge (Sprache, IDE, Frameworks)**

**/TOL-1/** Die Daten müssen in einer SQL-Datenbank gespeichert werden.

**/TOL-2/** Backend-Anwendungen werden mit .NET Framework implementiert.

**/TOL-3/** Frontend-Anwendungen werden mit JavaFX implementiert.

**Nicht-funktionale Anforderungen an die Teststrategie (Qualitätssicherung)**

**/TEST-1/** Die Teststrategie sollte eine ausreichende Abdeckung aller Funktionsbereiche und Anwendungsfälle sicherstellen. Dies kann durch die Verwendung von verschiedenen Testmethoden wie Einheitstests, Integrationstests, Systemtests und Akzeptanztests erreicht werden.

**/TEST-2/** Die Teststrategie sollte eine klare Rückverfolgbarkeit von Testfällen zu Anforderungen und Code ermöglichen, um eine effektive Fehlerbehebung und Problemlösung zu ermöglichen.

**/TEST-3/** Jede Komponente des Systems kann einzeln getestet werden.

# Abnahmekriterien

*Die Abnahmekriterien sind durch den Stakeholder definiert und sie dürfen nur mit Zustimmung des Stakeholders neu definiert, geändert oder erweitert werden.*

Das Projekt wird mit den folgenden Artefakten abgegeben:

* Dokumentation:
  + Pflichtenheft: *file-name.docx*
* Software
  + Link zu GitHub Projekt: *git-hub-link*
* Evidenz:
  + System/Software-Demo via Videoclip: *videoclip-link*

Anm.: Die Abgabetermine der Projetartefakts werden durch den Stakeholder festgelegt!

# Projekt Meilensteine

* *Meilenstein #1: Das Lastenheft ist fertiggestellt und mit dem Stakeholder abgestimmt.*
* *Meilenstein #2: Das Pflichtenheft Entwurf fertiggestellt und mit dem Stakeholder geteilt.*

# Referenzen

*Diese Kapitel beinhaltet die wichtigsten Literaturquellen aufgelistet.*