TAU INF202 Software Engineering

Individuelles Projekt

Pflichtenheft

Projektdokumentation

Version: 2023.04.16

Status: freigegeben

Public Transport Application

Verantwortliche/r:

Ramadan SALMAN: 190503007

Barkan ÇEK: 190503034

e190503007@stud.tau.edu.tr | rmdnslmn39@gmail.com
e190503034@stud.tau.edu.tr | barkancek2000@gmail.com

Stakeholder: DI. Ömer Karacan, omer.karacan@tau.edu.tr

Dokumentenverwaltung

Dokument-Historie

Version	Status *)	Datum	Verantwortlicher	Änderungsgrund
v1.0	freigegeben	25.02.2023	Ö. Karacan	Vorlage wurde für die Studentenprojekte freigegeben

**) Sofern im Projekt nicht anders vereinbart, sind folgende Statusbezeichnungen zu verwenden (in obiger Tabelle und am Deckblatt):*
Dokument-Status: Entwurf / in Review / freigegeben (abgegeben)

Dokument wurde mit folgenden Tools erstellt:

Microsoft Office Word

Inhaltsverzeichnis

1. Einleitung 4
2. Ausgangssituation und Ziele 5
3. Gesamtarchitektur 6
4. Funktionale Anforderungen 7
5. Nichtfunktionale Anforderungen 9
6. Abnahmekriterien 10
7. Projekt Meilensteine 11
8. Referenzen 12

1. Einleitung

Dieses Dokument dient dazu, die Anforderungen zu definieren, die verbindlich für die Prototypenentwicklung einer öffentlichen Verkehrs-App sind, und sie vollständig und konsistent zu beschreiben.

Die Use Cases und Anforderungen sind aus der Sicht des Stakeholders beschrieben.

Die grafische Benutzeroberfläche zur Überwachung der Stadtkarte, der Routen und der Bushaltestellen wird aus der Perspektive eines Fahrgastes beschrieben.

Im Kapitel 2 „ Ausgangssituation und Ziele “ sind die Ausgangssituation und der Grund zur Wahl von öffentliche Verkehrs-App anschaulich dargestellt.

Im Kapitel 3 „ Gesamtarchitektur “ sind die konzeptionelle Architektur des Systems, und die wichtigsten Subsysteme (Komponenten), die Anwender und die notwendigen Kommunikationsschnittstellen dargestellt. Hier sind auch zusätzliche Anforderungen an die Architektur oder Komponenten zu finden.

Im Kapitel 4 „ Funktionale Anforderungen “ beinhaltet die Beschreibung der funktionalen Anforderungen durch die Ablaufbeschreibungen (User Stories), und die Anwendungsfällen (Use Cases). Alle betriebsrelevanten
Daten werden durch die Datenmodellen definiert.

Im Kapitel 5 „ Nichtfunktionale Anforderungen “ sind die funktionalen Anforderungen durch diejenigen Anforderungen erweitert, die keine funktionalen Anforderungen sind.

Im Kapitel 6 „ Abnahmekriterien “ sind die Abgabeartefakten festgelegt, die ohne Abstimmung des Stakeholders nicht zu manipulieren sind.

Im Kapitel 7 „ Projekt Meilensteine “ sind die wichtigsten Termine aufgelistet, die den Fortschritt die Teilergebnisse des Projektes definieren.

Im Kapitel 8 „ Referenzen “ sind die wichtigsten Referenzen aufgelistet.

2. Ausgangssituation und Ziele

Einleitung

Menschen brauchten eine Anwendung, die ihnen dabei helfen würde, die Busse, die sie auf ihrem Weg von einem Ort zum anderen inmitten des Stadtdschungels nutzen werden, optimal auszuwählen. Sie benötigten eine Anwendung, die ihnen bei der Suche nach Antworten auf Fragen wie "Welchen Bus kann ich von welcher Haltestelle aus nehmen?", "Wann wird der Bus kommen?" oder "An welcher Haltestelle steige ich aus?" und "Gibt es alternative Routen?" hilft.

Problemstellung (Funktionalität)

In einer Stadt so groß wie Istanbul verbringen die Bürger einen Teil ihrer Zeit damit, von einem Ort zum anderen zu pendeln. Öffentliche Verkehrsmittel spielen in einer solch großen Stadt eine unvermeidliche Rolle. Es gibt Zeiten, in denen Menschen Hilfe benötigen, um zu entscheiden, welchen Bus sie nehmen sollen oder um die Busse an den Haltestellen zu sehen. Wenn sie aufgrund fehlender Direktbusse zu ihrem Ziel umsteigen müssen, kann es noch komplizierter werden. Es ist nicht so einfach, herauszufinden, welchen Bus man von welcher Haltestelle nehmen soll und um welche Uhrzeit man ihn erwischen muss. Die Schätzung, ob sie den Bus rechtzeitig erreichen werden, und die Vorhersage, wann der Bus an der Haltestelle ankommt, können schwierig sein. In einigen Fällen können sich die Routenpräferenzen der Benutzer unterscheiden. Zum Beispiel kann es Situationen geben, in denen sie für die schnellste Route viel laufen müssen. Dies kann dazu führen, dass die Route nicht bevorzugt wird. Unsere Anwendung zielt darauf ab, Benutzern in solchen Situationen zu helfen.

Stakeholder (Anwender):

Jeder, der Stadtbusse als Verkehrsmittel nutzen möchte und sich im Leistungsbereich der Anwendung befindet, kann diese Anwendung nutzen. Unsere Anwendung bietet einen Vorteil, da sie verschiedene Routenoptionen gemäß den Vorlieben des Benutzers erstellt.

Systemumfeld (Einsatzumgebung)

Das System wird in einer Computerumgebung funktionieren.

Rahmenbedingung (Einschränkungen)

Als Software-Entwicklungstool soll entweder Visual Studio Code verwendet werden,
die Backend Applikationen sollen mit Java Framework und Spring Framework,
die Frontend Applikationen mit Javascript realisiert werden, und
die persistenten Daten sollen in einem SQL-Datenbank abgespeichert werden.

Ziele (Lösung)

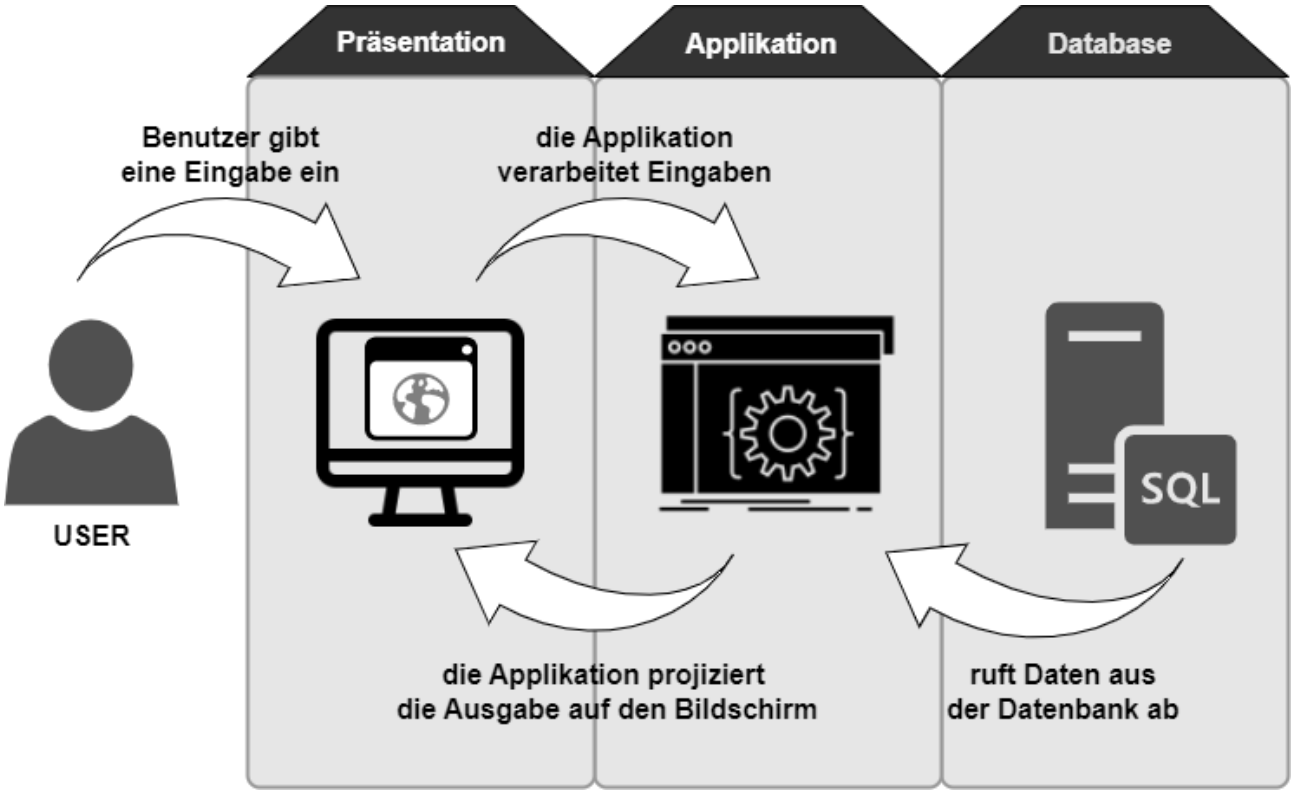
In der Anwendung werden dem Benutzer Felder zur Eingabe von Eingaben gemäß seinem Ziel bereitgestellt. Die ausgewählten Haltestellen werden auf der Karte markiert. Die Position der Busse und ihre Routen werden visualisiert. Die Anwendung ermöglicht es dem Benutzer, durch Auswahl der Elemente zu interagieren.
Die Persistenz soll mit einem Datenbanksystem erfolgen.

3. Gesamtarchitektur

Einleitung
In diesem Abschnitt werden Systemgrenzen definiert und modelliert.
Dieses Kapitel führt zu dem besseren Verständnis des angeforderten Systems und dadurch genaue Definition der funktionalen und nicht-funktionalen Anforderungen.

Gesamtarchitektur
Die Gesamtarchitektur sind in den folgenden Aspekten betrachtet:

Public Transport Application Infrastructure als Kontext,
3 Schichten für Komponentenarchitektur: Präsentation
Applikation
Datenbank
Ein UML-Diagramm, das die Klassen und Attributen beschreibt.



Komponente <x>
Komponente Presentation Layer: In dieser Schicht erfolgt die Kommunikation zwischen dem Benutzer und der Anwendung über eine visuelle Schnittstelle.

Komponente Application Layer: In der Anwendungsschicht wird der Datenaustausch zwischen dem Benutzer und der Datenbank bereitgestellt.

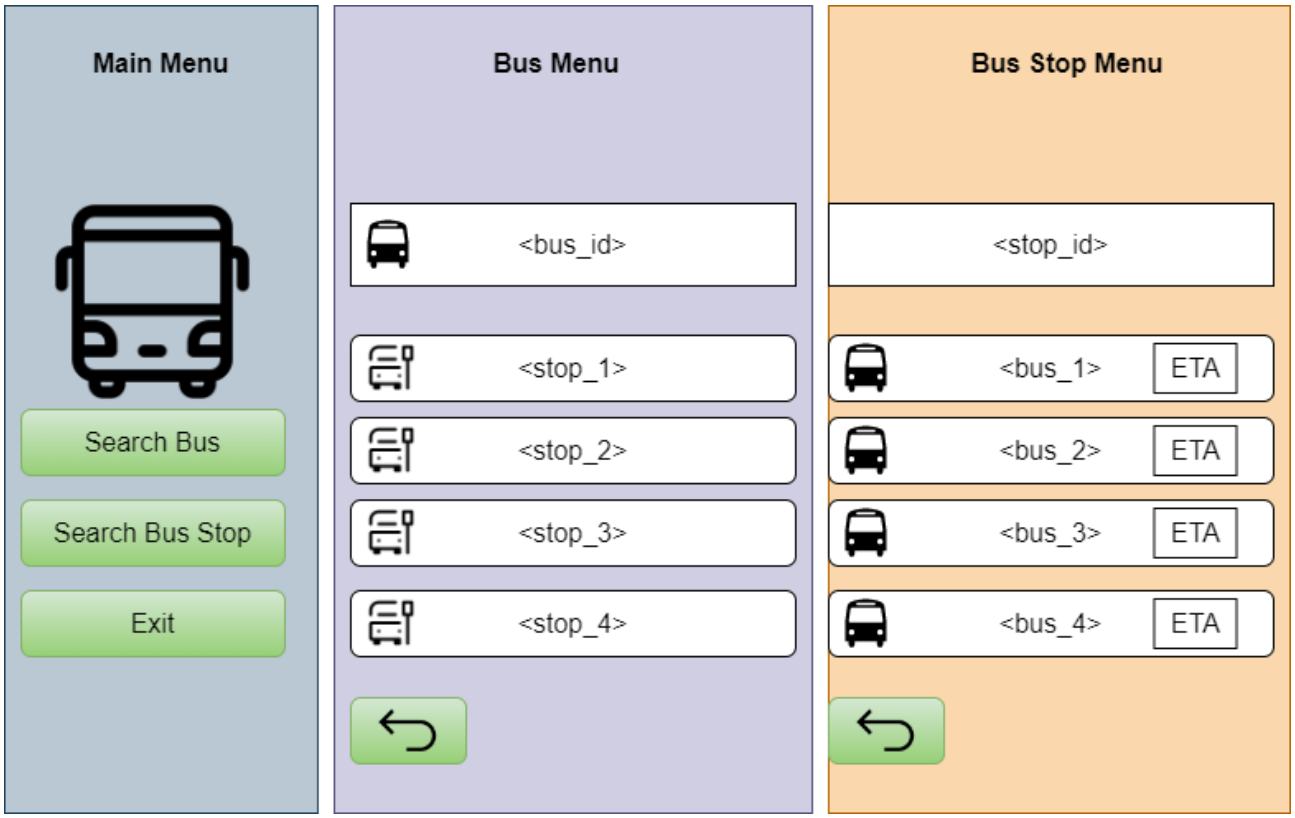
Komponente Database Layer: In dieser Schicht werden die Daten aller Busse und Haltestellen gespeichert. Wenn der Benutzer es braucht, greift die Anwendung auf diese Schicht zu und überträgt die Daten.

4. Funktionale Anforderungen

Einleitung

In diesem Kapitel sind Anforderungen (inklusive Use Cases) an Gesamtsystem
aber auch an die einzelnen Systemkomponenten definiert. Darüber hinaus ist die PTA
prototypisch visualisiert.

UI Use Cases

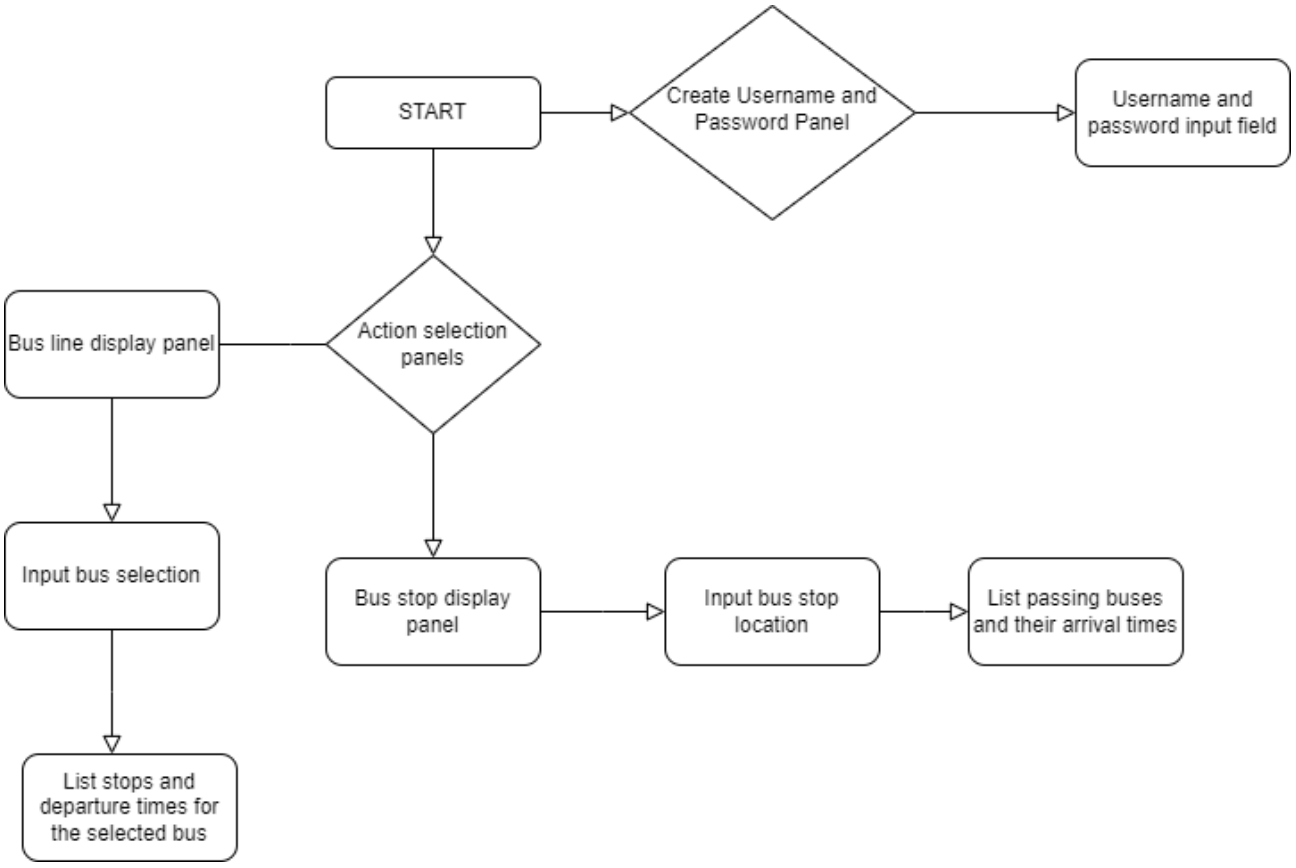


PTA - GUI Prototype

Desktop App Use Cases/User Stories

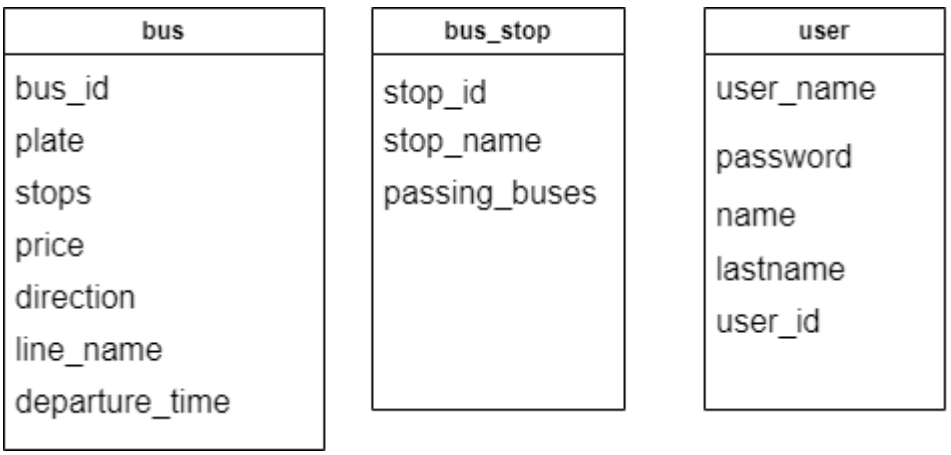
/PTA-1/ Bei der ersten Verwendung öffnet sich auf dem Anmeldebildschirm der Anwendung ein Panel zur Erstellung des Benutzernamens und des Passworts.
/PTA-2/ Im täglichen Gebrauch der Anwendung gibt es Panels, in denen der Benutzer auswählen kann, welche Aktionen er ausführen möchte.
/PTA-3/ Im Routenerstellungspanel gibt es Felder, in denen der Benutzer den Abfahrts- und Ankunftsort auswählt. Nach Eingabe der Benutzereingaben werden die Routen aufgelistet.
/PTA-4/ Auf dem Bildschirm, auf dem der Benutzer die Routendetails anzeigt, werden die Haltestellen, Abfahrtszeiten und die durchfahrenen Haltestellen angezeigt.
(Wir mussten eine Distanz zwischen den Haltestellen anhand eines Graphenkonzepts zuweisen, um diesen Use Case zu realisieren. Da es schwierig war, diese Haltestellen in der Tabelle zu verknüpfen, haben wir uns entschieden, diese beiden Use Cases zu entfernen.)
/PTA-5/ Im Haltestellendarstellungspanel gibt es Felder, in denen der Benutzer die Haltestelle auswählt. Nach der Eingabe der Benutzereingaben werden die Busse, die die Haltestelle passieren, und ihre Ankunftszeiten aufgelistet.

/PTA-6/ Im Busliniendarstellungspanel gibt es ein Feld, in dem der Benutzer den Bus auswählt. Nach der Eingabe der Benutzereingaben werden die Haltestellen und Abfahrtszeiten des ausgewählten Busses aufgelistet.



Datenmodel

- **/DB-1/** Die Daten werden in einer externen SQL-Datenbank gespeichert.
- **/DB-2/** Daten von Halteplätze, Busse und Benutzern werden in zwei getrennten Tabellen gehalten.



5. Nichtfunktionale Anforderungen

Einleitung
In diesem Kapitel sind die nicht-funktionalen Anforderungen ans Gesamtsystem aber auch an die einzelnen Systemkomponenten definiert. Es wird besonders auf die Software Qualität Wert gelegt (Testing).

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

/1*/ Beim Auflisten der Busse, die die Haltestelle passieren, werden alle Busse aufgelistet, die innerhalb der nächsten halben Stunde nach der Zeit, zu der die Daten angefordert wurden, passieren, beginnend mit den letzten drei Bussen, die vor dieser Zeit durchgefahren sind.

Nicht-funktionale Anforderungen an die Entwicklungsumgebung

/DEV-1/ Die Entwicklungsumgebung ist frei wählbar!

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

/TOL-1/ Die Backend Applikationen sollen mit Java Framework implementiert werden.
/TOL-2/ Die Frontend Applikation soll mit Javascript realisiert werden.
/TOL-3/ Die persistenten Daten sollen in einem SQL-Datenbank abgespeichert werden.

Nicht-funktionale Anforderungen an die Teststrategie (Qualitätssicherung)

/TEST-1/ Zuerst DB wird getestet, ob Daten sich stündlich ändern.
/TEST-2/ Alle Use Cases, User Stories und Anforderungen sollen getestet und berichtet werden.
/TEST-3/ Letztlich , Desktop-App wird bezüglich der Genauigkeit der Ergebnisse getestet.

6. Abnahmekriterien

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 Projektartefakts werden durch den Stakeholder festgelegt!

7. Projekt Meilensteine

Meilenstein M#1:

- Das Lastenheft ist fertiggestellt und mit dem Stakeholder abgestimmt.

Meilenstein e#2 (Zwischenabgabe):

- Das Pflichtenheft-Entwurf ist fertiggestellt um mit dem Stakeholder abzustimmen.

Meilenstein M#2:

- Das Pflichtenheft ist fertiggestellt und mit dem Stakeholder abgestimmt.

Meilenstein M#3:

- In diesem Meilenstein ist die Architektur im Vordergrund.
- Komponenten-basierte Architektur ist entworfen und komponentenweise implementiert.
- Die externen Schnittstellen (Webservices) wurden entworfen/implementiert.
- Die GUI Komponente ist ansatzweise fertig.

Meilenstein M#4:

- In diesem Meilenstein ist das Testen im Vordergrund.
- Die Test-Cases wurden aus den Use Cases abgeleitet und ansatzweise beschrieben.
- Die Testumgebung ist vorbereitet und ein Smoke Test ist durchgeführt.

Meilenstein M#5:

Das Projekt ist per Vereinbarung abgegeben.

8. Referenzen