|  |
| --- |
|  |
|  |

|  |  |
| --- | --- |
| THEIA Baustellen App | |
| Eine Lösungsarchitektur für eine mobile Applikation | |
| **Masterthesis** | |
| Studiengang: | MAS-IT Software Architecture |
| Autor\*in: | Zürcher Amos | Software Architect |
| Betreuer\*in: | Staub Bill | Business Unit Lead “Business Solutions” |
| Auftraggeber\*in: | Hellion AG |
| Expert\*in: | Kleiner Helmut Max, Schmidhauser Arno |
| Datum: | [Datum einfügen] |

Management Summary

**Ausgangslage**

**Ziel**

**Vorgehen**

**Ergebnisse**

Inhaltsverzeichnis

[Eine Lösungsarchitektur für eine mobile Applikation 1](#_Toc188879106)

[Management Summary 2](#_Toc188879107)

[Inhaltsverzeichnis 3](#_Toc188879108)

[1 Einleitung 4](#_Toc188879109)

[1.1 Ausganslage und Motivation 4](#_Toc188879110)

[1.2 Problemstellung 4](#_Toc188879111)

[1.3 Ziel der Arbeit 4](#_Toc188879112)

[1.4 Abgrenzung 5](#_Toc188879113)

[1.5 Aufbau der Arbeit 5](#_Toc188879114)

[2 Literaturreview 5](#_Toc188879115)

[2.1 Grundlagen 5](#_Toc188879116)

[2.2 CAP Theorem 6](#_Toc188879117)

[2.2.1 Konsistenz 6](#_Toc188879118)

[2.2.2 Verfügbarkeit 6](#_Toc188879119)

[2.2.3 Partitionstoleranz 6](#_Toc188879120)

[2.3 Entwurfsmuster für die Datensynchronisation 7](#_Toc188879121)

[2.3.1 Datensynchronisation 7](#_Toc188879122)

[2.3.2 Datenspeicher und Verfügbarkeit 8](#_Toc188879123)

[2.3.3 Datenübertragung 8](#_Toc188879124)

[3 Methodik 8](#_Toc188879125)

[3.1 Überschrift 9](#_Toc188879126)

[3.1.1 Überschrift 9](#_Toc188879127)

[3.2 Überschrift 9](#_Toc188879128)

[4 Ergebnisse 9](#_Toc188879129)

[4.1 Pflichtenheft 9](#_Toc188879130)

[4.1.1 Authentifizierung 9](#_Toc188879131)

[4.1.2 Projektübersicht 10](#_Toc188879132)

[4.1.3 Projektdetails 10](#_Toc188879133)

[4.1.4 Notizen zum Auftrag 10](#_Toc188879134)

[4.1.5 Mockups zur Benutzeroberfläche der mobilen Applikation 11](#_Toc188879135)

[4.2 Anwendungsfälle Baustellen App 15](#_Toc188879136)

[4.3 Lösungsarchitektur 16](#_Toc188879137)

[5 Diskussion / Zusammenfassung 16](#_Toc188879138)

[5.1 Überschrift 16](#_Toc188879139)

[5.2 Überschrift 16](#_Toc188879140)

[Abbildungsverzeichnis 17](#_Toc188879141)

[Tabellenverzeichnis 17](#_Toc188879142)

[Glossar 17](#_Toc188879143)

[Literaturverzeichnis 18](#_Toc188879144)

[Anhang 19](#_Toc188879145)

[Selbständigkeitserklärung 20](#_Toc188879146)

# Einleitung

## Ausganslage und Motivation

Für die Versorgung ihrer Mitarbeitenden mit projektrelevanten Informationen, hat die Helion AG die Isolutions AG angefragt, eine Offerte für eine Mobile Applikation namens «THEIA Baustellen App» zu erstellen. Diese Applikation sollte mit einer API eines bestehende Systems kommunizieren. Eine erwähnenswerte Anforderung an die Applikation ist die erwartete offline Fähigkeit.

Die Helion AG ist eine Expertin für Energielösungen in der Schweiz und hat sich zum Ziel gesetzt die Energiewelt voranzutreiben. Helion AG bietet Lösungen im Bereich Photovoltaik, Stromspeicher, Smart Energy, Wärmepumpen und Ladesationen für Elektrofahrzeuge. Die Dienstleistungen umfassen die Beratung, Verkauf, Montage und Wartung und wird von einem Team von über 500 Mitarbeitenden ausgeführt.

Heute ist es besonders für die Mitarbeitenden auf der Baustelle schwierig auf die benötigten Informationen zum Projekt zu zugreifen. Ausserdem gibt es im Projektablauf Prozesse, welche die Interaktion der Mitarbeitenden erfordern. Zum Beispiel, wird für den Abschluss des Projekts ein Inbetribenahmeprotokoll mit einer vorgegebenen Checkliste verlangt. Weitere Prozesse sind Materialbestellungen, Informationsaustausch mit dem Projektmanagement oder Backoffice, usw. Heute werden diese Prozesse zum grossen Teil auf Papier durchgeführt.

Da die Anzahl der Mitarbeitenden und das Auftragsvolumen in den letzten Jahren gewachsen ist, stellt sich der Helion AG auch die Herausforderung ihre Prozesse in der Projektumsetzung zu überarbeiten. Deshalb hat sich die Helion AG dazu entschieden einen mobile Applikation für Ihre Mitarbeitenden umzusetzen.

Im Verlauf der Antragstellung dieser Master Thesis, hat sich die Helion AG dazu entschieden , die «THEIA Baustellen App» nicht in zusammenarbeit mit Isolutions zu erstellen. In Absprache mit den Experten und der Studienleitenden der Master Thesis, wurde entschieden, die Arbeit trotzdem auf Basis der Anforderungen der Helion AG durchzuführen.

## Problemstellung

Die bestehende Lösung THEIA besteht aus einem Desktop Client und einem Backendsystem. Der Kontext in dem dieses Client/Server System arbeitet, zeichnet sich durch eine stabile Netzwerkverbindung aus und der Speicher im Client ist keine limitierende Ressource. Ausserdem werden Daten höchstens lokal zwischengespeichert aber nicht persistiert, da von einem permanent verfügbaren Backen ausgegangen werden kann.

Mit der Implementierung einer mobilen Applikation werden diese zwei Vorteile der Desktop Lösung negiert. In einer mobilen Applikation kann nicht von einer durchgehenenden Netzwerkkonnektivität ausgegangen werden und der Speicher auf dem Gerät ist limitiert und meistens nicht erweiterbar. Tatsächlich ist eine spezifische Anforderung an die mobile App, die offline fähigkeit. Da gemäss Helion AG die Verbindung in das Internet auf der Baustelle oft nicht gewährleistet ist.

Aufgrund dieser Ausgangslage ist die klar, dass auf die Verfügbarkeit der Projektdaten, Funktionalität der App ohne Internet, und Konsistenz der Daten ein spezielles Augenmerk gelegt werden muss. Es gibt unterschiedliche Ansatze, Entwurfsmuster und Mehtoden für die Synchronisierung von Daten, Sicherstellung von Verfügbarkeit und Konsistenz. Es stellt sich die Frage, welches Entwurfsmuster ist am besten geeignet, um die Anforderungen der Helion AG an diese Applikation zu erfüllen.

## Ziel der Arbeit

Das Ziel dieser Arbeit ist es in einem Softwarearchitekturdokument (SAD) eine mögliche Lösungsarchitektur aufzuzeigen. Folgende Lieferobjekte werden erarbeitet, um die Fragen der Problemstellung zu beantworten.

* Lösungsarchitektur für die «Baustellen App» nach Arc42.
* Architekturentscheid für die Umsetzung der Datensynchronisation.
* Minimum Viable Product (MVP) der Baustellen App umsetzen.

## Abgrenzung

Folgende Punkte sind nicht Teil der Masterthesis:

* Die Lösungsarchitektur konzentriert sich auf die Frage der Datensychnronisation und offline Modus. Eine vollständige Implementierung der Funktionalen Anforderungen ist nicht Teil dieser Arbeit.
* Auf Grund der vorhandenen Infrastruktur beschränkt sich der PoC auf die Plattform Apple iOS.
* Eine Integration in die IT-Landschaft der Helion AG ist nicht teil dieser Masterthesis.

Wie bereits in der Ausgangslage erwähnt, hat sich die Helion AG aus dem Projekt zurückgezogen. Das führt zu einigen Einschränkungen:

* Es steht kein Backend mehr zur Verfügung. Deshalb muss für den MVP ein eigenes Backend erstellt werden.
* Auf Grund der zeitlichen Einschränkung ist Sicherheit kein Bestandteil dieser Arbeit
* Die Lösungsarchitektur beschreibt den MVP entsprechend der Anforderungen der Helion AG.

## Aufbau der Arbeit

Die Masterarbeit ist folgendermassen aufgebaut. Zuerst wird in einer Einführung die Ausgangslage, Motivation, Problemstellung und die Zielsetzung erläutert.

Das zweite Kapitel beschäftigt sich mit dem Literaturreview rund um das Thema Datensynchronisation. Der Autor setzt sich mit der Beziehung und Problematik von Partitionstoleranz, Konsistenz und Verfügbarkeit in einem Verteilten System ausseinander und beleuchtet Datenübertragung, -synchronisation und -speicherung für mobile Applikationen.

Im dritten Kapitel geht der Autor auf die Methodik ein. Durch Transparenz in der Methodik soll die Reproduzierbarkeit gewährleistet werden und die Grundlage für eine kritische Bewertung legen.

Im vierten Kapitel ist das Pflichtenheft aus der Anfrage der Helion AG an die Isolutions AG aufgeführt. Dieses Pflichtenheft ist die Grundlage für die Lösungsarchitektur.

Im fünften Kapitel werden die Lieferobjekte beschrieben. Im ersten Teil wird die Lösungsarchitektur aufgezeigt. Der zweite Teil befasst sich bessonders mit der Datensynchronisation und des gewählten Entwurfsmusters. Die Validierung der Umsetzung des MVPs schliesst dieses Kapitel ab.

# Literaturreview

Dieses Kapitel beinhaltet die Grundlagen zur Datensynchronisation und Datenkonsistenz.

## Grundlagen

In einem Multiusersystem mit verteilten Komponenten, wie beispielsweise einer Webapplikation mit einer Frontendkomponente als Client und dem Backend als Server, gibt es zum einen den Aspekt der Synchronisation von Daten zwischen den Komponenten und zum anderen den simultanen Zugriff von meheren Clients auf die möglicherweise gleichen Datensätze. Der Aspekt der Datensynchronisation hat vor allem Einfluss auf die Verfügbarkeit der Anwendung und damit auf das Benutzererlebnis. Der zweite Aspekt beeinflusst primär die Konsistenz der Daten, welche für eine Geschäftsapplikation wie in diesem Fall, von grosser Bedeutung ist.

## CAP Theorem

Gemäss [1] spielt in einem verteilten System das CAP Theorem eine Rolle. Das Theorem beschreibt frei Einflussbereiche denen ein verteiltes System unterliegt.

### Konsistenz

Consistency bzw. Konsistenz beschreibt die Eigenschaft, dass der Informationsgehalt von Daten für alle Komponenten oder Teilnehmer in einem System zu jedem Zeitpunkt gleich ist [3].

A diagram of a server

Description automatically generated

Abbildung 7 Konsistentes System

A diagram of a server

Description automatically generated

Abbildung 8 Inkonsistentes System

Abbildung 7 zeigt ein verteiltes System von drei Clients, welche den Datensatz {p} vom Server abfragen. {p} ist in diesem Fall konsistent, da alle Komponenten im System mit {p1} den gleichen Informationsgehalt von {p} haben.

Abbildung 8 zeigt ein inkonsistentes System. Dies ist beispielsweise in dem Moment der Fall, wenn der Client A den Datensatz {p} bearbeitet und an den Server sendet, mit der Anweisung den Datensatz {p} anzupassen zu {p2}. Der Datensatz {p} ist nicht mehr Konsistent da der Client B und C den Datensatz {p1} haben, wärend Client A und der Server den Datensatz {p2} haben.

Um die Konsistenz wieder herzustellen, muss Client B und C den Datensatz {p} abfragen, um die Version {p2} zu erhalten.

### Verfügbarkeit

Als Verfügbar bzw. Available gilt ein Service, wenn der Service auf eine Anfrage eine Antwort zurücksendet. Je nach Art des Systems ist die Zeitspanne von der Anfrage bis zu der Antwort auch von Bedeutung. In einem Echtzeit-System oder einer Benutzeranwendung ist eine Antwort welche zu lange braucht genau so schlecht wie keine Antwort [1].

### Partitionstoleranz

Die Partionstoleranz bzw. Partition Tolerance beschreibt die Fähigkeit eines verteilten Systems, mit Kommunikationsausfällen oder -verzögerungen umzugehen. Ein System ist partitionstolerant, wenn es trotz Kommunikationsausfall zwischen Knoten im System weiter arbeiten kann [2].

## Entwurfsmuster für die Datensynchronisation

Gemäss [3] gibt es einige Design Pattern für die synchronisation von Daten im mobilen Applikationsdesign. Diese Entwurfsmuster werden in diesem Abschnitt beschrieben. Sie lassen sich grob in 3 Bereiche aufteilen, „Datensynchronisation“, „Datenspeicher und Verfügbarkeit“ und „Datenübertragung“ [4].

### Datensynchronisation

Dieser Bereich lässt sich in zwei Entwurfsmuster einteilen. Synchrone und Asynchrone Datensynchronisation. Beide Varianten haben Vor- und Nachteile und eignen sich für spezifische Einsatzgebiete.

A screenshot of a computer

Description automatically generated

Abbildung 9 Datensynchronisation als synchroner Fluss

In Abbildung 9 wird ein synchroner Datenfluss in bzw. einer mobilen App dargestellt. Die Abbildung stellt dar, dass wenn die Applikation Daten von einem Backendserver synchronisiert, die Benutzeroberfläche für den Benutzer gesperrt wird. Erst wenn die Synchronisation der Daten abgeschlossen ist, kehrt die Applikation wieder in einen benutzbaren Zustand zurück. Der Vorteil dieser Variante liegt in einem einfacheren State Management in der Applikation. Als negativer Punkt steht dem gegenüber die Einschränkung des Benutzererlebnisses durch das Blockieren der Applikationsoberfläche [3].

A screenshot of a phone

Description automatically generated

Abbildung 10 Darstellung einer asynchronen Datensynchronisation

In der Abbildung 10 ist der Abblauf einer asynchronen Datensynchronisation zu sehen. Der Unterschied zur synchronen Datensynchronisation ist, dass sich die Applikation bei der Initiierung der Synchronisation in zwei States begibt. Die Benutzeroberfläche kehrt in den benutzbaren Stand zurück, wärend gleichzeitig die Datensynchronisation im Hintergrund läuft [3]. Das ergibt den Vorteil, dass die Applikation für den Benutzer benutzbar bleibt. Ausserdem erlaubt eine asynchrone Datensynchronisation den Austausch von Daten im Hintergrund. Der Nachteil ist die erhöhte Komplexität von Datenabfragen. Mehrere konkurrierende Anfragen an ein Datenset können auftreten und müssen behandelt werden [3]. Beispielsweise durch den Einsatz von Transaktionen [3] und/oder von Sperrmechanismen für den Datensatz.

### Datenspeicher und Verfügbarkeit

In diesem Bereich gibt es ebenfalls zwei Entwurfsmuster, welche sich auf die lokale Verfügbarkeit auf dem Gerät auswirken. Bei der Wahl dieser Entwurfsmuster sind bessonders der verfügbare Speicherplatz auf dem Gerät und die Netzwerkbandbreite von Bedeutung [3]. Das erste Entwurfsmuster ist „Partial Storage“ bzw Teilspeicher. Dieses Entwurfsmuster reduziert den Speicherbedarf der Daten auf dem Gerät, da mit diesem Muster nur die gerade benötigten Daten vom Backend geladen werden. Ein „Data Access Object“ reagiert auf eine Datenanfrage in dem es die angefragten Daten aus dem lokalen Speicher zurückgibt oder vom Backend abfragt, wenn sie lokal nicht vorhanden sind. Auf der negativen Seite führt dieses Muster zu erhöhtem Netzwerkverkehr [3]. Ein weiterer Nachteil dieses Musters ist die Abhängigkeit zur Verfügbarkeit des Netzwerkes [4]. Fällt die Konnektivität im Netzwerk aus, kann die mobile Applikation keine Daten mehr nachladen.

Das zweite Entwurfsmuster ist der Complete Storage bzw. Komplettspeicher. Im Gegensatz zum Partial Storage wird in diesem Muster der komplete Datensatz auf ein mobiles Gerät geladen. Dieser Prozess wird üblicherweise beim Start der Applikation ausgeführt, damit subsequente Datenanfragen immer aus dem lokalen Speicher beantwortet werden können. Der Vorteil ist, dass die Abhängigkeit zur Netzwerkverfügbarkeit geringer ist und damit die Verfügbarkeit der Daten erhöht wird. Wärend der Datensynchronisation aus dem Backend ist dagegen die Netzwerkbandbreite gross, da die Menge an Daten bei der Datensynchronisation grösser ist. Ausserdem benötigt diese Variante mehr speicher auf dem mobilen Gerät [3].

### Datenübertragung

Dieser Abschnitt behandelt das Thema Datenübertragung. Das Ziel ist die optimierte Übertragung von Daten aus dem Backend zu der mobilen Applikation. Bei der Wahl des Entwurfmuster zur Datenübertragung, muss primär der Typ der zu übertragenden Daten in Betracht gezogen werden [3]. Für kleine Datensätze oder Daten, welche jeweils als ganzes verändert werden, beispielsweise ganze Dateien, eignet sich das Entwurfsmuster „Full Transfer“. Das Entwurfsmuster zeichnet sich dadurch aus, dass der komplete Datensatz übertragen wird und den aktuellen Datensatz auf dem mobilen Gerät ersetzt. „Full Transfer“ hat den Vorteil, dass es ein einfach zu implementierendes Entwurfsmuster ist. Der Nachteil ist, dass die Daten redundant gesendet werden und dadurch wird Bandbreite verschwendet [3].

Das Muster „Timestamp Transfer“ macht sich ein Zeitstempel auf dem Datensatz zu nutze, um zu determinieren ob der Datensatz synchronisiert werden muss. Die mobile Applikation fragt den Zeitstempel der letzten Mutation vom Backend ab und vergleicht diesen Zeitstempel mit dem Zeitstempel im lokalen Datensatz. Ist der Zeitstempel ungleich, wird eine Synchronisation der Daten initiiert [3]. Das hat den Vorteil, dass gezielt Daten synchronisiert werden und damit die Menge an Daten, welche über das Netzwerk übertragen werden minimiert werden. Bei der Implementierung ergeben sich hingegen zwei Nachteile. Die Quelle des Timestamps muss konsitent bleiben, um keine inkonsistente Synchronisation zu riskieren und die Löschung von Datensätzen muss unter umständen separt behandelt werden, da auf einem gelöschten Datensatz kein Zeitstempel der letzten Mutation vorhanden ist [3].

Das dritte Entwurfsmuster zur Datenübertragung verwendet, für die Bestimmung der zu übertragenden Daten, mathematische Algorithmen. Beispielsweise der Vergleich von Checksummen oder komplexere Algorithmen [3]. Dieses Entwurfsmuster hat das Potenzial die beste Ausnutzung von Bandbreite zu bieten und ist deshalb interessant, wenn die Bandbreite sehr limitiert ist. Der Nachteil ist die komplexe Implementierung, welche entsprechend mehr Zeit für die Implementierung benötigt [3].

# Methodik

In diesem Kapitel wird die verwendete Methodik beschrieben Durch die Transparenz in der Methodik wird nicht nur die Reproduzierbarkeit gewährleistet, sondern auch die Grundlage für eine kritische Bewertung der Arbeit gelegt.

## Projektmanagement

Für die Umsetzung wurde eine Agile Vorgehensweise gewählt. In einem Projektboard in Github werden Tasks erfasst und iterativ abgearbeitet. Die Tasks sind primär auf die Lieferobjekte zugeschnitten. Das Projektboard orientiert sich dabei an der Kanban Methode [5]. Das Projektboard befindet sich auf Github:

<https://github.com/users/zuercheram/projects/2/views/3>

Der Gesamtvortschritt der Masterthesis ist in einem übergeordneten Projektplan dargestellt. Er zeigt die Termine, Meilensteine und geplante Umsetzungsabschnitte für die Masterthesis.

## Masterthesis Zeitplan



## Entwurf der Lösungsarchitektur

## Analyse der Datensynchronisation

## Umsetzung der Baustellen App als MVP

# Ergebnisse

In diesem Kapitel werden die Anforderungen der Helion AG an die Baustellen App aufgezeigt. Dies dient dem Leser zur Orientierung da die Lösungsarchitektur auf diesen Anforderungen beruht.

## Pflichtenheft

Im Pflichtenheft sind die buisnessrelevanten Anforderungen an die Applikation aufgeführt. Zusätzlich zu den Anforderungen sind Mockups im Pflichtenheft enthalten. Die Mockups visualisieren wie Anforderungen in die Benutzeroberfläche realisiert werden sollen. Die Mockups sind Illustrationen und fokusieren sich auf das Layout und positionierung von Informationen auf dem Bildschirm. Je nach Plattform werden sich die Elemente visuell von den Mockups unterscheiden.

Die Anforderungen und Mockups sind aus dem Angebot «Konzeption THEIA für die Baustelle» für die Helion AG abgeleitet. Das Angebotsdokument wurde von der Isolutions AG verfasst. Da das Angebotsdokument interne Informationen enthält, wurden die Anforderungen generalisiert.

### Authentifizierung

Tabelle 1 Anforderungen an die Authentifizierung

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-A1 | Authentisierung | Die Benutzer müssen sich beim Einstieg in die App als erstes authentifizieren. Erst nach Authentisierung und Autorisierung können die Benutzer auf die Inhalte zugreifen. |
| REQ-A2 | Speicherung der Session | Nach dem Einloggen bleibt die Session des Benutzers in der App gespeichert, damit sich der Benutzer nicht immer wieder neu anmelden muss. Nach einer zu definierenden Zeitdauer wird die Session ungültig und der Benutzer muss sich neu anmelden. |
| REQ-A3 | Logout | Der Benutzer muss die Möglichkeit haben, sich wieder abzumelden. |

### Projektübersicht

Tabelle 2 Anforderungen an die Projektübersicht

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-B1 | Projektliste | Nach dem Start der Applikation soll als erste Seite eine Liste mit den Projekten dargestellt werden. Vorausgesetzt der Benutzer ist eingeloggt. |
| REQ-B2 | Personalisierte Projekte | In der Projektliste werden nur diejenigen Projekte angezeigt, auf die der Benutzer die Berechtigung zum Zugriff hat. |
| REQ-B3 | Projektname | Jedes Projekt ist mit dem Namen und der Ortschaft der Baustelle aufgelistet. |
| REQ-B4 | Projektsuche | In einem Suchfeld kann ein Projekt nach dem Namen und der Ortschaft gesucht werden. Die Liste passt sich entsprechend dem Suchbegriff laufend an. Der Suchbegriff muss wieder gelöscht werden können. |
| REQ-B5 | Projektauswahl | Mittels Touch auf eines der Projekte können die Projektdetails aufgerufen werden. |
| REQ-B6 | Offlinefähigkeit | Projekte sollen auch Offline verfügbar sein. Der Benutzer kann in der Projektliste Projekte markieren welche offline verfügbar sein sollen. Sämtliche Projektdaten zu einem markierten Projekt sollen offline verfügbar sein. |

### Projektdetails

Tabelle 3 Anforderungen an die Projektdetails

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-C1 | Projektdetails | Es werden die wichtigsten Infos eines Projektes gemäss obigen Mockups angezeigt. Ist eine Internetverbindung vorhanden, werden die Daten vom Backend geladen. Ohne Internetverbindung werden die Daten aus dem lokal Speicher geladen. |
| REQ-C2 | Adressen in Maps öffnen | Kontakt- und Objektadresse können per Touch in Karten (ios) oder Maps (android) geöffnet werden. |
| REQ-C3 | Absprung auf externe Ablagen | Wird auf dem Projekt ein Link zu einer externen Ablage mitgeliefert, wird in der Detailsansicht ein Link Button auf die externe Ablage angezeigt. |
| REQ-C4 | Notizen zum Auftrag | Über einen weiteren Button kann die Projektnotizen-Funktion aufgerufen werden. |
| REQ-C5 | Inbetriebnahme | Über einen weiteren Button kann die Inbetriebnahme-Funktion aufgerufen werden. |
| REQ-C6 | Zurück zur Übersicht | Mittels eines Zurück-Buttons muss wieder in die Projektübersicht gewechselt werden können. |
| REQ-C7 | Offlinefähigkeit | Sämtliche Daten dieses Projekts müssen auch ohne aktive Internetverbindung auf den Smartphones abrufbar sein. Online Links auf externe Ablagen funktionieren im Offline-Modus nicht. |

### Notizen zum Auftrag

Tabelle 4 Anforderung zu den Notizen

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-19 | Notizen-Übersicht | In einer Liste werden sämtliche bisher erfassten Notizen zum ausgewählten Projekt angezeigt. |
| REQ-20 | Filter nach Kategorie | In einem Dropdown Feld kann die jeweilige Notizen-Kategorie gefiltert werden. Standardmässig ist kein Filter gesetzt. |
| REQ-21 | Detailansicht Notizen | In einer Detailansicht können die Notizen angezeigt werden. Externe Links (z.B. auf Word-Dokumente welche in SharePoint abgelegt sind) können angeklickt und im externen Browser (Safari oder Chrome) angezeigt werden, falls die entsprechenden Berechtigungen vorhanden sind. Eingebettete Bilder werden geladen und angezeigt. |
| REQ-22 | Notiz editieren und erstellen. | Notizen können editiert und erstellt werden. |
| REQ-23 | Zurück zur Übersicht | Mittels eines Zurück-Buttons muss wieder in die Notizen-Übersicht gewechselt werden können. |
| REQ-24 | Offlinefähigkeit | Sämtliche Notizen der geladenen Projekte müssen auch ohne aktive Internetverbindung auf den Smartphones abrufbar sein. Externe Links funktionieren offline nicht. |

### Mockups zur Benutzeroberfläche der mobilen Applikation

Mit den Mockups soll das Konzept der Benutzeroberfläche visualisiert werden. Die Mockups wurden mit dem UI/UX Design Tool Figma[[1]](#footnote-1) erstellt. Benutzt wurden standart Asset Bibliotheken. Das Look and Feel der visuellen Elementen entspricht nicht dem finalen Aussehen in der Applikation, sondern dient nur zur Veranschaulichung des Benutzeroberfläche und der User Experience

#### Benutzeroberfläche für die Authentifizierung

A screenshot of a login screen

Description automatically generated

Abbildung 1 Loginmaske

#### Projektbezogene Ansichten

A screenshot of a phone

Description automatically generated

Abbildung 2 Projektübersicht

A screenshot of a phone

Description automatically generated

Abbildung 3 Projektdetailansicht

#### Notizenansichten

A screenshot of a phone

Description automatically generated

Abbildung 4 Notizenübersicht

A screenshot of a cell phone

Description automatically generated

Abbildung 5 Notizdetailansicht

A screenshot of a phone

Description automatically generated

Abbildung 6 Neue Notiz erfassen

## Anwendungsfälle Baustellen App

Basierend auf den Anforderungen aus dem Pflichtenheft, wurden für den MVP der Baustellen App Anwendungsfälle abgeleitet. Die Anwendungsfälle dienen zur Visualisierung welche Aktionen das System asuführen kann, welche Aktoren mit dem System interagieren und in welche Beziehung die Aktoren und Aktionen stehen. [6]

A screenshot of a diagram

Description automatically generated

Abbildung 11 Anwendungsfalldiagramm für die Applikation Baustellen App

|  |  |  |
| --- | --- | --- |
| Name | **UC-1 Neues Projekt anlegen** | |
| Ziel im Kontext | Der Sachbearbeiter oder Projektleiter, nachfolgend nur noch Benutzer genannt, sollen in der Applikation neue Projekte erstellen können. | |
| Akteure | Sachbearbeiter, Projektleiter | |
| Trigger | Wird vom Benutzer ausgelöst | |
| Vorbedingungen | Keine | |
| Essenzielle Schritte | **Intention der Systemumgebung** | **Systemreaktion** |
| Benutzer klickt auf Schaltfläche „Projekt eröffnen“ | Applikation öffnet eine View mit der Eingabemaske für die Erfassung von Projektdaten |
| Benutzer gibt Projektdaten ein |  |
|  | Benutzer klickt auf die Schaltfläche «Projekt speichern» | Die Applikation überprüft und speichert die Eingabe. |
| Synchronisations-verhalten |  | |

## Lösungsarchitektur

In diesem Kapitel wird das Konzept und für die Entwicklung relevanten Querschnittsthemen betrachtet. Ein ergänzendes Dokument zu diesem Kapitel ist das Software Architektur Dokument (SAD). Das SAD ist gemäss der arc42 Vorlage von Dr. Peter Hruschka und Dr. Gernot Starke in der Version 8.2 von Januar 2023 aufgebaut.[[2]](#footnote-2)

# Diskussion / Zusammenfassung

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit.

## Überschrift

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit.

Quatur ad quibusamus, et exerionem eostis peror sedipis aut int la peris eatibusam is aut autem imporum soluptatium coritas perepratem doluptas sitatur atium, ilitat velenihictem eaquas molor serit doloratiis abo.

## Überschrift

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit.

# Abbildungsverzeichnis

[Abbildung 7 Konsistentes System 6](#_Toc188879473)

[Abbildung 8 Inkonsistentes System 6](#_Toc188879474)

[Abbildung 9 Datensynchronisation als synchroner Fluss 7](#_Toc188879475)

[Abbildung 10 Darstellung einer asynchronen Datensynchronisation 7](#_Toc188879476)

[Abbildung 1 Loginmaske 11](#_Toc188879477)

[Abbildung 2 Projektübersicht 12](#_Toc188879478)

[Abbildung 3 Projektdetailansicht 12](#_Toc188879479)

[Abbildung 4 Notizenübersicht 13](#_Toc188879480)

[Abbildung 5 Notizdetailansicht 13](#_Toc188879481)

[Abbildung 6 Neue Notiz erfassen 14](#_Toc188879482)

[Abbildung 11 Anwendungsfalldiagramm für die Applikation Baustellen App 15](#_Toc188879483)

# Tabellenverzeichnis

[Tabelle 1 Anforderungen an die Authentifizierung 9](#_Toc188879484)

[Tabelle 2 Anforderungen an die Projektübersicht 10](#_Toc188879485)

[Tabelle 3 Anforderungen an die Projektdetails 10](#_Toc188879486)

[Tabelle 4 Anforderung zu den Notizen 10](#_Toc188879487)

# Glossar

**Backend**

Als Backend wird die Softwarekomponente eines verteilten Systems bezeichnet, welche auf einem Server betrieben wird und über ein privates Netzwerk oder das Internet erreichbar ist. Üblicherweise ist das Backend für die Persistierung von Daten zuständig und enthält unter Umständen Geschäftslogik, welche die ankommenden oder ausgehenden Daten entsprechend verarbeitet.

**Desing Patter**  
Auf Deutsch Entwurfsmuster, ist ein Begriff aus der Softwareentwicklung. Ein Entwurfsmuster ist wie der Name sagt ein Muster um ein bestimmtes Problem zu lösen. Das Design Pattern beschreibt die Lösung ohne konkrete Implementierungen. Dadurch kann das Entwurfsmuster unabhängig von der Plattform eingesetzt werden.

**Frontend**

Das Frontend ist der Teil eines Softwaresystems, welches für die Interaktion mit den Anwendern zuständig ist. Es stellt eine Benutzeroberfläche zur Verfügung.

**MVP**

Minimum Viable Product ist die englische Beschreibung eines Produkts mit dem minimal nötigen Funktionsumfang, um von Kunden oder Benutzern verwendet zu werden und Rückmeldung für die zukünftige Funktionsentwicklung zu gewinnen. Dieser Term wird üblicherweise in der Software Entwicklung verwendet.

**SAD**

Das Software Architektur Dokument ist das Ergebnis der Arbeit des Software Architekten. Das Dokument gibt darüber Auskunft, wie die Qualitätsanforderungen an die Applikation umgesetzt werden sollen. Es dokumentiert die relevanten Architekturentscheidungen, analysiert Risiken, definiert Massnahmen zur Risikominderung und enthält weitere Querschnittsthemen, welche aus Sicht des Architekten für die Applikation von Bedeutung sind.

**UI/UX**

UI ist die Abkürzung für User Interface. Zu Deutsch Benutzeroberfläche. UX steht für User Experience und beschreibt die Benutzererfahrung. Beide Begriffe zusammen behandeln das Aussehen und Verhalten von grafischen Oberflächen von Softwaresystemen, welche von menschlichen Anwendern verwendet werden.

**Use Cases**

Zu Deutsch Anwendungsfälle. Ein Anwendungsfall beschreibt eine Funktionalität eines Systems, welcher Aktor oder welches System diese Funktionalität verwendet

# Literaturverzeichnis

[1] S. Gilbert und N. Lynch, «Perspectives on the CAP Theorem», *Computer*, Bd. 45, Nr. 2, S. 30–36, Feb. 2012, doi: 10.1109/MC.2011.389.

[2] «Was ist das CAP-Theorem? | IBM». Zugegriffen: 25. Januar 2025. [Online]. Verfügbar unter: https://www.ibm.com/de-de/topics/cap-theorem

[3] Z. McCormick, «Data Synchronization Patterns in Mobile Application Design».

[4] D. Hasanenko, «Data Synchronization Patterns», Medium. Zugegriffen: 1. Juli 2024. [Online]. Verfügbar unter: https://hasanenko.medium.com/data-synchronization-patterns-c222bd749f99

[5] «Kanban», *Wikipedia*. 12. Januar 2025. Zugegriffen: 27. Januar 2025. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Kanban&oldid=252173638

[6] «Use case diagrams are UML diagrams describing units of useful functionality (use cases) performed by a system in collaboration with external users (actors).» Zugegriffen: 31. Dezember 2024. [Online]. Verfügbar unter: https://www.uml-diagrams.org/use-case-diagrams.html

# Anhang

## Sourcecode

Der Sourcecode zum PoC der Baustellen App befindet sich in folgendem Git Repository

<https://github.com/zuercheram/bauestellen-app>

# Verwendung von KI

In dieser Arbeit wurde auf die Verwendung von KI verzichtet.

# Selbständigkeitserklärung

Ich bestätige, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der im Literaturverzeichnis angegebenen Quellen und Hilfsmittel angefertigt habe. Sämtliche Textstellen, die nicht von mir stammen, sind als Zitate gekennzeichnet und mit dem genauen Hinweis auf ihre Herkunft versehen.

Ich bestätige weiterhin, dass ich bei der Erstellung dieser Studienarbeit durchgehend steuernd gearbeitet habe und von einer KI erzeugte Texte bzw. Textfragmente nicht unreflektiert übernommen habe.

Ort, Datum:

1. https://www.figma.com/ [↑](#footnote-ref-1)
2. https://arc42.org/overview [↑](#footnote-ref-2)