HTL Dornbirn

Höhere Technische Bundeslehr- und Versuchsanstalt für Wirtschaftsingenieurwesen

Ausbildungsschwerpunkt: Betriebsinformatik, Produktmanagement und Futuretecs

Diplomarbeit

Konzeption eines intelligenten, automatischen Fahrradparksystems

Ausgeführt im Schuljahr 2022/23 von:

Paul Hartmann	5aWI
Joshua Lung	5aWP

Betreuer:

Prof. MMag. Günter Hämmerle
Mag. Andreas Riedmann
LTW Intralogistics GmbH

Dornbirn, am 24. April 2023

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich den vorliegenden Diplomarbeitsteil/die vorliegende Diplomarbeit selbsständig und ohne fremde Hilfe verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die den benutzten Quellen wörtlich und inhaltich entnommenen Stellen als solche erkenntlich gemacht habe.

Dornbirn Höchsterstraße 73, 6850 Dornbirn am 24. April 2023

Paul Hartmann

Paul Hartmann

Joshua Lung

Versionshistorie

Version	**Datum**	**Änderungen**
0.1	06.03.2023	Vorschauversion
0.2	15.03.2023	Version für Matze
0.3	16.03.2023	Korrekturversion Riedmann I
0.4	16.03.2023	Korrekturversion Hämmerle
0.5	24.03.2023	Version für LTW
0.6	26.03.2023	Korrekturversion Riedmann II
1.0	31.03.2023	Einreichungsversion
1.1	21.04.2023	HTL V Version

Tabelle 1: Versionshistorie

Abstract

In recent years, cycling has become an increasingly popular and environmentally friendly alternative to cars. E-bikes in particular have gained popularity in the latest years, as they offer a fast and efficient way to get around. However, theft and vandalism of expensive bikes are a major problem and there is a shortage of secure parking and charging options for bikes.

Our goal is to design an automated bicycle tower, which is a vertical storage system designed to securely store bicycles in a limited space. By developing this technology, e-bikes can be made more attractive. The goal is to encourage more people to use a bicycle and reduce carbon dioxide emissions.

The project idea came from our project supervisor Günter Hämmerle, who put us in contact with our partner company LTW Intralogistics. The topic sparked our interest immediately and we quickly decided to work out the task as our diploma thesis. With the thesis we had both a challenging job and a broad subject area.

This thesis shows a market analysis, the design of two new automated bicycle towers and the development of a prototype, to show how an automated bicycle tower works.

Vorwort

In den letzten Jahren hat sich das Fahrradfahren zu einer immer beliebteren und umweltfreundlicheren Alternative zum Autofahren entwickelt. Besonders E-Bikes haben in den letzten Jahren an Beliebtheit gewonnen, da sie eine schnelle und effiziente Fortbewegungsmöglichkeit bieten. Jedoch sind Diebstahl und Vandalismus von teuren Fahrrädern ein großes Problem und es gibt zu wenig sichere und praktische Abstell- und Auflademöglichkeiten für Fahrräder. Hier kommt das automatische Fahrradparkhaus ins Spiel. Ein automatisches Fahrradparkhaus ist eine turmförmige Konstruktion, in der möglichst viele Fahrräder auf kleinstem Raum gelagert werden können. Die Vision sieht vor, die Fahrräder mittels automatischer Lagerroboter auf mehreren vertikalen Ebenen aufzuteilen, um die Platzeffizienz weiter zu steigern. Durch die Förderung dieser Technologie können E-Bikes attraktiver gemacht werden und dazu beitragen, die Nutzung des umweltfreundlichen Verkehrsmittels zu erhöhen.

Die Projektidee stammt von unserem Projektbetreuer Günter Hämmerle, der den Kontakt zu unserem Partnerunternehmen LTW Intralogistics herstellte. Das Thema interessierte uns sofort und wir entschieden uns rasch, die Aufgabenstellung im Zuge unserer Diplomarbeit auszuarbeiten.

Mit dem Thema „Konzeption eines intelligenten, automatischen Fahrradparksystems“ hatten wir sowohl eine herausfordernde Arbeit als auch ein breites Themengebiet. In den letzten Wochen und Monaten arbeiteten wir fleißig an der Durchführung und Dokumentierung der Arbeit und sind stolz, die Ergebnisse nun präsentieren zu können.

Danksagung

An dieser Stelle möchten wir uns bei all denjenigen bedanken, die uns während dem Verlauf der Diplomarbeit unterstützt und motiviert haben:

Prof. Mag. Günter Hämmerle Besonders danken wir unserem Projektbetreuer Günter Hämmerle. Ihm verdanken wir die Projektidee und den Kontakt zu unserem Partnerunternehmen. Er unterstütze uns in allen Projektphasen, war bei jedem Treffen bei dem Partnerunternehmen dabei, motivierte uns und brachte neue Ideen ein.

Mag. Andreas Riedmann Ein weiterer Dank geht an Andreas Riedmann, der uns bei Fragen zur Verfügung stand, bei der Entscheidung für die Technologien maßgeblich beteiligt war und uns bei der Dokumentation der Diplomarbeit unterstützte.

LTW Intralogistics Vielen Dank an das Unternehmen LTW Intralogistics für die Ermöglichung der Diplomarbeit und die gute Zusammenarbeit. Besonders danken wir Johannes Schwartze für die Betreuung des Projektes. Außerdem möchten wir uns bei Rainer Sohm, Lisa-Maria Burtscher, Thomas Fitz, Chiara De Sousa, Benedikt Sabato und allen anderen beteiligten Mitarbeiter:innen von LTW für die interessanten und aufschlussreichen Meetings bedanken.

Bernd Frankenhauser Des Weiteren danken wir Bernd Frankenhauser, ohne den das klassenübergreifende Team nicht bestehen würde.

Thomas Schmid Thomas Schmid spielte eine wichtige Rolle beim Bereitstellen der Komponenten. Er half uns bei der Auswahl der Hardware und stellte uns diese zur Verfügung.

Eltern, Freunde und Bekannte Ein herzliches Dankeschön geht an alle Eltern, Freunde und Bekannte, die uns bei der Durchführung und Dokumentierung der Diplomarbeit motiviert und unterstützt haben.

Benutzer von Stackoverflow Ein besonderer Dank geht an die Benutzer von Stackoverflow, die uns bei der Lösung von Problemen geholfen haben. Stackoverflow ist eine online Plattform auf der Entwickler Fragen stellen und beantworten können. Alle Fragen und Antworten sind gratis, frei zugänglich und werden von freiwilligen Helfern beantwortet.

Abdulla Abdurakhmanov Abdulla Abdurakhmanov (`@abdolence`) ist der Entwickler und Betreuer der firestore_rs Softwarebibliothek. Bei Bugs und Fragen zu dieser Softwarebibliothek war er stets hilfsbereit und hat uns schnell geholfen.

Inhaltsverzeichnis

1 Impressum **9**

1.1 Projektteam . . . 9

1.2 Projektbetreuer der Schule . . . 10

1.3 Projektbetreuer vom Partnerunternehmen . . . 10

2 LTW Intralogistics GmbH **11**

3 Einleitung **12**

3.1 Entstehung der Idee . . . 13

3.2 Problemstellung . . . 14

3.3 Zielsetzung . . . 14

4 Projektmanagement **15**

4.1 Meilensteine . . . 15

4.2 Projektauftrag . . . 16

4.3 Projektorganigramm . . . 18

4.4 Projektstrukturplan . . . 18

4.5 Gantt Diagramm . . . 20

4.6 Objektstrukturplan . . . 21

4.7 Risikoanalyse . . . 21

4.8 Projektumweltanalyse . . . 23

4.9 Kanban . . . 24

4.10 Projektfunktionendiagramm . . . 25

4.11 Zeiterfassung . . . 26

5 Marktanalyse **29**

5.1 Existierende Modelle . . . 29

5.1.1 Fahrradständer . . . 29

5.1.2 Unterirdisches Fahrradparkhaus . . . 29

5.1.3 Fahrradbox . . . 30

5.1.4 Wöhr Bikesafe . . . 30

5.1.5 Eco Cycle . . . 31

5.1.6 V-Locker . . . 32

5.2 Nutzwertanalyse . . . 33

5.2.1 Gegenüberstellung . . . 36

5.3 Erweiterung der Nutzungstiefe . . . 37
5.3.1 Vermieten von Fahrrädern . . . 37
5.3.2 Vermieten von E-Scootern . . . 37
5.3.3 Abstellen von Gegenständen . . . 37
5.3.4 Abgabe und Abholung von Paketen . . . 37
5.3.5 Abgabe zur Fahrradreparatur . . . 37
5.4 Kundensegmente . . . 38
5.4.1 Großunternehmen . . . 38
5.4.2 Wohnbau-Unternehmen . . . 38
5.4.3 Öffentliche Einrichtungen . . . 38
5.5 Öffentliches Verkehrsnetz und -verbünde . . . 38
5.6 Verkehrspolitische und kommerzielle Rahmenbedingungen . . . 38
5.7 Umfrage . . . 39
5.8 Schlussfolgerung . . . 40

6 Produktkonzeption 41
6.1 Varianten . . . 41
6.1.1 Rondell System . . . 41
6.1.2 Hochregallager System . . . 45
6.2 Innovation . . . 47

7 Konzept des Prototypen 48
7.1 Turm . . . 48
7.2 App . . . 49
7.2.1 Funktionen . . . 49
7.2.2 Technologie . . . 49
7.2.3 Mockups . . . 51
7.3 Datenbank . . . 55
7.3.1 Arten von Datenverknüpfung . . . 55

8 Umsetzung des Prototypen 56
8.1 App . . . 56
8.1.1 Verwendete Pakete . . . 56
8.1.2 Navigation . . . 61
8.1.3 Komponenten . . . 63
8.1.4 Screens . . . 66
8.2 Turm . . . 79
8.2.1 Software . . . 79
8.2.2 Hardware . . . 89
8.3 Datenbank . . . 93

8.3.1 Serialisierung / Deserialisierung . . . 93
8.3.2 Datenbankschema . . . 93
8.4 Kommunikation . . . 96
8.4.1 Realtime Updates . . . 96
8.4.2 Fahrradturm Protokoll . . . 96

9 Marktskalierung **99**
9.1 Jugend Innovativ . . . 99
9.2 Einnahmeströme und Austauschprozesse . . . 101
9.3 Absatzplan . . . 102
9.3.1 Stellplätze Zuwachs . . . 102
9.3.2 Investitionssummen . . . 103
9.4 Kostenrechnung . . . 104
9.5 Finanzplan . . . 105
9.5.1 LTW Finanzplan . . . 105
9.5.2 BikeTower GmbH Finanzplan . . . 106

10 Verwendete Technologien **107**
10.1 Firebase . . . 107
10.2 Raspberry PI . . . 108
10.3 Neopixel . . . 108
10.4 Rust . . . 109
10.4.1 `firestore_rs` Softwarebibliothek . . . 109
10.5 React Native . . . 110
10.6 Javascript . . . 110
10.7 Expo . . . 111
10.8 Autodesk Inventor . . . 111
10.9 3D Druck . . . 112
10.9.1 Stereolithographie (SLA) . . . 112
10.9.2 Selektives Laser Sintern (SLS) . . . 113
10.10 Cinema 4D . . . 114
10.11 Visual Studio Code . . . 114
10.12 GitHub . . . 115
10.13 LaTeX . . . 115

11 Zusammenfassung **116**

12 Literatur- und Quellenverzeichnis **117**

13 Abbildungsverzeichnis **123**

14 Codeverzeichnis **126**

15 Tabellenverzeichnis **127**

16 Abkürzungsverzeichnis **128**

17 Glossar **130**

18 Autorenverzeichnis **133**

19 Anhang **134**

1 Impressum

Das Impressum beinhaltet die Schüler, die die Diplomarbeit ausgearbeitet haben, und deren Betreuer.

1.1 Projektteam

Das Projektteam besteht aus folgenden Schülern:

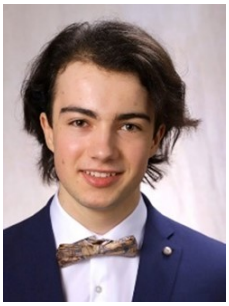
Projektteam		
Paul Hartmann	Paul übernahm die Projektleitung und war für die Kommunikation mit dem Partnerunternehmen zuständig. Er recherchierte zu Varianten zur Lagerung von Fahrrädern und führte die Nutzwertanalyse und Umfrage aus. Paul entwickelte die Mobile-App.	
Joshua Lung	Joshua recherchierte zu Varianten zur Lagerung von Fahrrädern, führte die Nutzwertanalyse aus, erstellte Fragen der Umfrage und konzeptionierte das Rondell-Modell. Er baute den Prototyp des Turmes und übernahm die Softwareentwicklung der Steuerung.	

Tabelle 2: Projektteam

1.2 Projektbetreuer der Schule

Projektbetreuer		
Prof. MMag. Hämmerle Günter	Günter Hämmerle initiierte das Projekt mit dem Partnerunternehmen LTW Intralogistics und war bei allen Besprechungen mit dem Partnerunternehmen dabei. Er unterstützte das Team in allen Projektphasen, besonders bei der Analyse der Kosten und der Dokumentation der Diplomarbeit.	
Prof. Mag. Riedmann Andreas	Andreas Riedmann unterstützte das Team bei der Softwareentwicklung des Prototypen und bei der Dokumentation der Diplomarbeit.	

Tabelle 3: Projektbetreuer

1.3 Projektbetreuer vom Partnerunternehmen

Johannes Schwartze war der Betreuer der Diplomarbeit vonseiten des Partnerunternehmens LTW Intralogistics. Er war der Ansprechpartner bei Fragen und gab neue Aufgabenstellungen und Feedback. Mit dem Partnerunternehmen gab es mehrere Besprechungen, an denen die Fortschritte präsentiert und diskutiert wurden.

2 LTW Intralogistics GmbH

LTW Intralogistics ist ein Tochterunternehmen der Doppelmayr-Gruppe. LTW Intralogistics wurde 1981 gegründet und betreibt mittlerweile 4 Standorte. Es baut erfolgreich Intralogistik-Systeme, wie zum Beispiel Regalbediengeräte, automatische Hochregallager, automatische Kleinteillager und anderes. Das Unternehmen hat dabei schon 950 Projekte in 35 Ländern umgesetzt. Zu den Kunden von LTW Intralogistics gehören unter anderem Continental, IKEA, die Jungfraubahnen und Haberkorn. (vgl. *LTW Unternehmenswebsite* 2023; *Lösungen* 2023)

Die Technologien und das Knowhow von LTW eigenen sich perfekt für das Bauen eines automatischen Fahrradparkhauses und somit als Partnerunternehmen für die Diplomarbeit.

Abbildung 1: LTW Logo (vgl. *LTW Unternehmenswebsite* 2023)

3 Einleitung

Vor allem im 21. Jahrhundert wird Umweltfreundlichkeit immer wichtiger. Immer mehr Menschen fahren deshalb und aus vielen anderen Gründen (Gesundheit, Zeitersparnis, etc. . .) mit dem Fahrrad (vgl. Statista 2023). In vielen engen (und immer enger werdenden (vgl. Ritchie und Roser 2018)) Großstädten der Welt ist das sichere Lagern von Fahrrädern jedoch schwierig bis unmöglich. Herkömmliche Fahrradständer sind zwar platzeffizient und billig, jedoch nicht sicher (vgl. Vorarlberger Landesregierung 2020). Besonders bei teuren E-Bikes trauen sich viele nicht, dieses einfach an einen Pfosten zu sperren. Als Alternative dazu gibt es Rad-Boxen oder unterirdische Garagen, beide Optionen sind aber teurer, eingeschränkt nutzerfreundlich und oft zu weit weg von dem Ort, an den man will.

Abbildung 2: Randvoller Fahrradabstellplatz in Amsterdam (vgl. Hałun 2005)

In Zukunft wird es sogar noch mehr E-Bikes geben, denn fast die Hälfte aller verkauften Fahrräder sind mit Elektroantrieb ausgestattet (Tendenz stark steigend) (vgl. Zsifkovits 2022). Vor allem der urbane Raum benötigt derzeit dringend Abstellmöglichkeiten, an denen Fahrräder schnell, sicher und möglichst billig in der Nähe geparkt werden können.

3.1 Entstehung der Idee

In Städten wie London oder Wien ist dem Projektbetreuer Günter Hämmerle aufgefallen, dass immer mehr Stadtbewohner:innen Fahrräder nutzen, es aber große Defizite bei sicheren Abstellplätzen gibt. Zuhause werden teure Fahrräder nicht selten auf die kleinen, wertvollen Balkone der Privatwohnungen (sicher) abgestellt. Einmal mit dem Fahrrad im Stadtzentrum, wird es noch schwieriger. Das Projektteam vertritt die Hypothese, dass viele Stadtbewohner:innen auf das Fahrrad umsteigen werden, sobald dieses Problem zufriedenstellend gelöst ist. Daraus formte sich die Idee eines „Fahrradturms“. Wie der Name schon verrät, ist ein Fahrradturm eine turmförmige Konstruktion, in der möglichst viele Fahrräder auf kleinstem Raum gelagert werden können. Die Vision sieht vor, die Fahrräder mittels automatischer Lagerroboter auf mehrere vertikale Ebenen aufzuteilen, um die Platzeffizienz weiter zu steigern.

Abbildung 3: Fahrrad auf Balkon (vgl. Muller 2017)

3.2 Problemstellung

Problem aus der Sicht von Fahrradfahrer:innen in Ballungsräumen:

> „*Es besteht keine Möglichkeit ihr hochwertiges Fahrrad (Trend zu E-Bikes und hochpreisige Rennfahrrädern) sicher zu verwahren. Viele Fahrradbesitzer:innen weichen auf den eigenen Balkon oder sogar ihren Wohn- oder Schlafzimmer als Abstellplatz aus, weil das Risiko von Totalverlust durch Vandalismus oder Diebstahl im öffentlichen Raum zu hoch ist. Sobald sie ihr Fahrrad benutzen, d. h. damit in die Stadt oder zum nächsten Bahnhof (bzw. U-Bahn-Station) fahren, stehen sie vor demselben Problem.*“ (vgl. Schwartze 2022)

Problem aus der Sicht von Bewohner:innen von Ballungsräumen:

> „*Die Städteplanung nach dem 2. Weltkrieg stand unter dem Primat des PKWs als dem vorrangigen Verkehrsmittel. Dementsprechend wurden breite Straßen mit hohen Kapazitäten (und dementsprechendem Platzverbrauch) und vor allem auch der notwendige Parkraum geschaffen. Man kann davon ausgehen, dass ein PKW im Stadtverkehr max. 2 Stunden/Tag fährt, d. h. Straßenkapazität beansprucht, aber dementsprechend mindestens das 11-fache an Parkraumkapazität belegt. (2 Stunden Fahrzeit = 22 Stunden Standzeit pro Tag). Kurz- und mittelfristig wurde versucht, das Problem mit dem Bau von Tiefgaragen und Parkhäusern zu lösen, aber mit steigenden Grundstückspreisen wird dieser Lösung zunehmend Kostengrenzen (aus der Sicht der Anbieter) bzw. Parkticketpreisgrenzen (aus Sicht der PKW-Nutzer) gesetzt. Das Verhältnis zwischen Platzverbrauch eines PKWs (2,5 m x 4 m = 10 qm?) und beförderter Person (meistens nur 1 Person / PkW) ist schlicht und einfach unwirtschaftlich, d. h. für moderne Städte untragbar.*“ (vgl. Schwartze 2022)

3.3 Zielsetzung

Das Ziel ist die Entwicklung eines automatischen Fahrradparksystems zur effizienten, platzsparenden und sicheren Lagerung von Fahrrädern oder anderen Gegenständen an der Schnittstelle zum öffentlichen Verkehrsnetz. Es muss analysiert werden, welches System am Ökonomischsten ist, unter anderem bei den Punkten Kosten, Flächenverbrauch und Nutzerfreundlichkeit, um anschließend ein Modell davon zu erstellen. Eine detaillierte technische Ausarbeitung ist nicht vorgesehen. Schlussendlich ist geplant eine Softwarelösung für Benutzer (App) und einen Prototyp zur Veranschaulichung zu entwickeln.

4 Projektmanagement

Um größere Projekte erfolgreich zu meistern, ist es notwendig, Projektmanagement zu etablieren. Dieses Projektmanagement ist in der Lage, Ziele zu definieren, Projekte zu organisieren, Ressourcen zu managen, Qualität zu gewährleisten, Kommunikation zu etablieren und Risiken zu identifizieren. Um verschiedene Teile eines Projekts zu planen werden verschiedene Projektmanagementpläne verwendet. Das Projektteam hat sich entschieden, die auf den nächsten Seiten folgenden Projektmanagementpläne zu erstellen. (vgl. Frankenhauser 2019c)

Die Projektmanagementpläne wurden in Drawio (vgl. *Diagram Software and Flowchart Maker* 2023) und XMind 8 (vgl. *Xmind - Mind Mapping App* 2023) erstellt.

4.1 Meilensteine

Meilensteine sind wichtige Ziele im Projekt deren Erreichnung zeitlich festgelegt sind. Alle Meilensteine wurden zeitgereicht erreicht. Die Meilensteine sind in der Tabelle 4 aufgeführt. Bei jedem Meilenstein sind alle Projektmitglieder beteiligt, jedoch wurden einzelne Teilaufgaben gerecht verteilt.

Meilensteine

PSP-Code	Meilenstein	Plantermin
1.1.1	Projektstart	2022-06-28
1.2.7	Analysephase Abgeschlossen	2022-12-31
1.3.9	Fertigstellung Prototyp	2023-02-15
1.4.4	Testphase abgeschlossen	2023-02-20
1.5.8	Fertigstellung schrifliche Arbeit	2023-03-30
1.5.9	Präsentation Diplomarbeit	2023-06-14
1.1.4	Projektabschluss	2023-06-14

Tabelle 4: Meilensteine

4.2 Projektauftrag

Ein Projektauftrag legt wichtige Eckdaten des Projekts fest und wird von Projektleitern und Auftraggebern unterschrieben (vgl. Frankenhauser 2019b). Auf den folgenden Projektauftrag hat sich das Projektteam geeinigt (Tabelle 5).

Projektauftrag

Projektname	Konzeption eines intelligenten, automatischen Fahrradparksystems
Projektbeschreibung	Fahrräder sollen platzeffizienter und sicherer als bei herkömmlichen Systemen gelagert werden.
Projektstartergebnis	Kickoff meeting mit LTW Intralogistics GmbH
Projektstarttermin	2022-10-17
Projektendergebnis	Präsentation Diplomarbeit
Projektendtermin	2023-06-14
Projektziele	• Platzeffiziente, sichere und kostengünstige Fahrradlagerung • Analyse verschiedener bestehender und neuer Systeme • Konstruktion eines Prototypen • Entwicklung einer prototyp App • Businessplan & Kostenrechnung • Ausarbeitung der Ergebnisse als Diplomarbeit
Nicht-Ziele	• Konstruieren eines fertigen Fahrradparkhauses • Entwickeln einer fertigen App • Gründen einer Firma
Meilensteine	Siehe Meilensteine (Tabelle 4)

Hauptaufgaben	• Projektmanagement • Analyse • App und Protoypnentwicklung • Testphase • Dokumentation
Projektressourcen	• PCs für Teammitglieder • Benötigte Software (z.B. Konstruktionsprogram (CAD), Entwicklungsumgebung (IDE)) • Benötigte Hardware (z.B. Raspberry PI, LEDs)
Projektbudget	150 € (Kosten für die Herstellung des Prototypen)
Projektauftraggeber	LTW Intralogistics GmbH
Projektbetreuer	Prof. Mag. Günter Hämmerle, Mag. Andreas Riedmann, LTW Intralogistics GmbH
Projektleiter	Paul Hartmann
Projektteammitglieder	Paul Hartmann, Joshua Lung
Unterschrift Projektleiter	Paul Hartmann

Tabelle 5: Projektauftrag

4.3 Projektorganigramm

Das Projektorganigramm gibt Aufschluss über die genaue Projekthierachie. Es zeigt die einzelnen Projektmitglieder und deren Rollen innerhalb des Projektes (vgl. Frankenhauser 2019e).

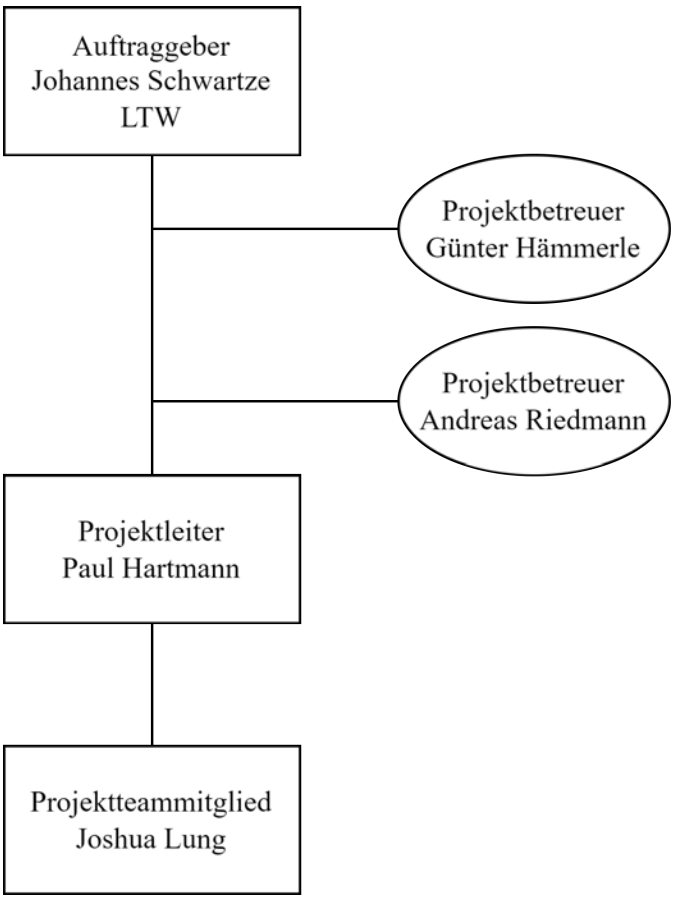


Abbildung 4: Projektorganigramm

4.4 Projektstrukturplan

Ein größeres Projekt umzusetzen erfordert viel Arbeit. Manchmal erscheinen die Aufgaben so komplex, das man nicht weiß wo man zuerst anfangen soll. Um Arbeit und konkrete Arbeitsschritte besser darstellen zu können, greifen Projektmanager:innen oftmals auf den Projektstrukturplan zurück. Der Projektstrukturplan teilt größere und komplexere Arbeiten in kleine und konkrete Arbeitspakete auf. Die konkreten Arbeitspakete werden anschließend gruppiert, sequenziell nummeriert und den verschiedenen Abteilungen bzw. Mitarbeiter:innen zugeteilt. Anhand eines Projektstrukturplans kann ein Gantt Plan erstellt werden. Der Gantt Plan zeigt die zeitliche Abfolge der einzelnen Arbeitspakete. Der Projektstrukturplan ist ein wichtiges Werkzeug für die Planung und Steuerung von Projekten. In Abbildung 5 wird der Projektstrukturplan des

Prototypen dargestellt.

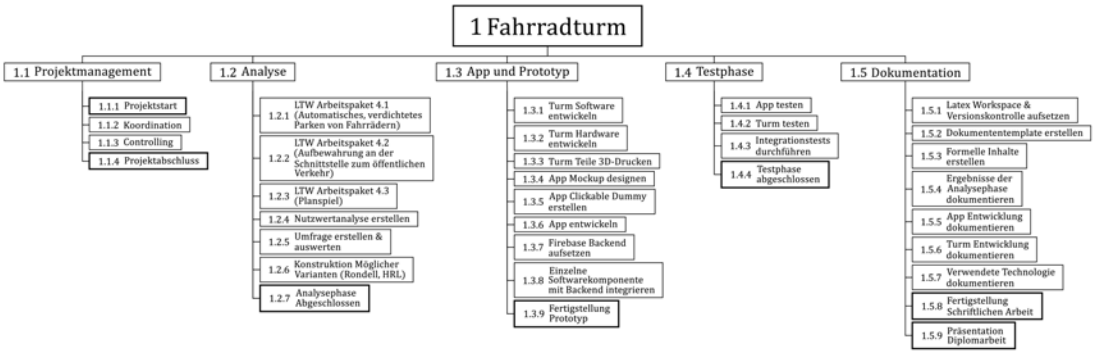


Abbildung 5: Projektstrukturplan

4.5 Gantt Diagramm

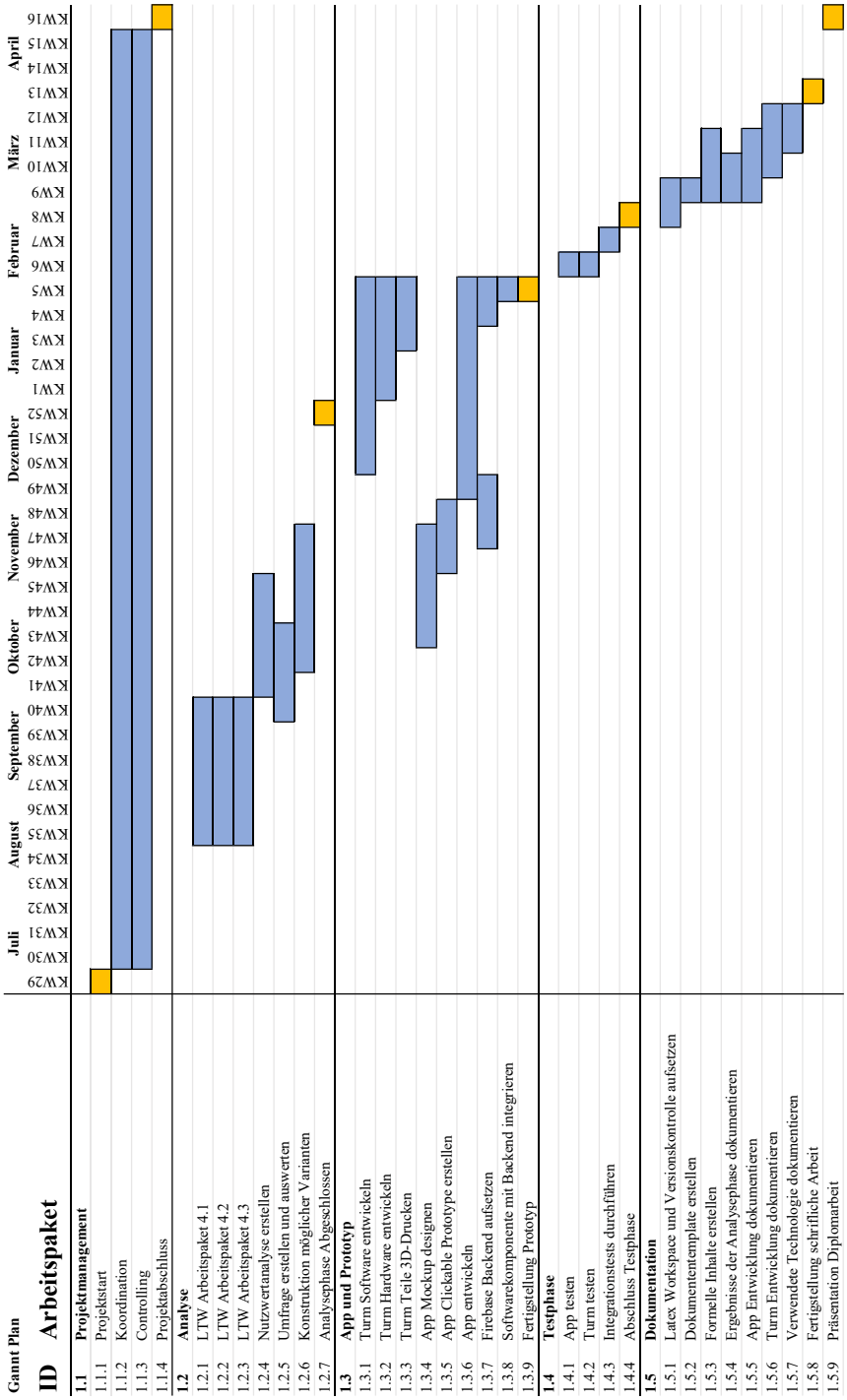


Tabelle 6: Gantt Diagramm

4.6 Objektstrukturplan

Der Objektstrukturplan ist visuell sehr änhlich zum Projektstrukturplan, jedoch werden hier nicht Arbeitspakete dargestellt, sondern die einzelnen Teilkomponenten des Produktes. Der Objektstrukturplan schafft einen guten Überblick über das Produkt. In Abbildung 6 wird der Objektstrukturplan des Prototypen dargestellt. (vgl. *Projektstrukturplan und Objektstrukturplan* 2015)

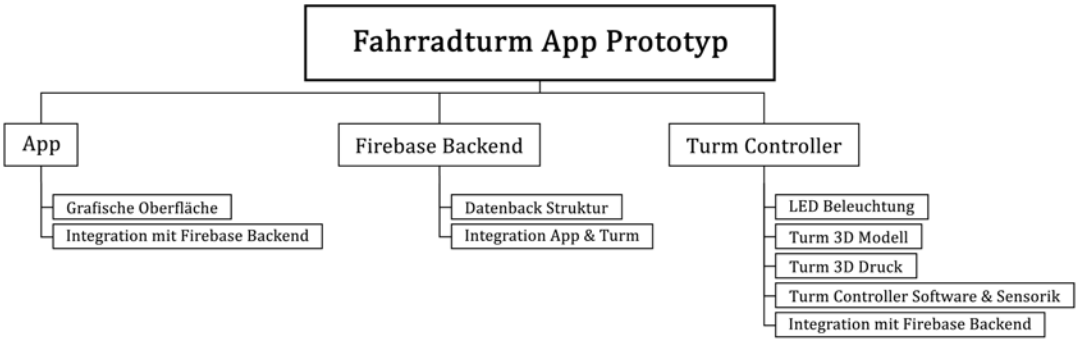


Abbildung 6: Objektstrukturplan des Prototypen

4.7 Risikoanalyse

Jedes Projekt birgt Risiken. Diese Risiken können sich auf die Projektdauer, die Kosten oder die Qualität auswirken. Um diese Risiken zu minimieren, wird eine Risikoanalyse erstellt. Zuerst werden die Risiken identifiziert und wie diese Risiken eintreten können. Anschließend werden Maßnahmen erarbeitet, die gegen diese Risiken getroffen werden sollen. Die Wahrscheinlichkeit, dass diese Risiken eintreten, wird ebenfalls ermittelt. Unabhängig davon, wie detailliert eine Risikoanalyse ausgearbeitet wird, ist es dennoch immer wichtig mit unidentifizierten Risiken zu rechnen. Auch während des Projektverlaufs können neue Risiken auftreten und müssen dann ebenfalls berücksichtigt und laufend zur Risikoanalyse hinzugefügt werden (vgl. *Risikomanagement im Projekt* 2023). In der Tabelle 7 werden Risiken, die das Projektteam identifiziert hat aufgelistet.

Kosten des endgültigen Produktes haben für die Diplomarbeit keine Relevanz. Trotzdem werden die Kosten und die damit verbundenen Risiken miteinbezogen. Falls das Projekt weiter verfolgt wird, werden Kosten relevanter und haben dadurch auch ein höheres Risiko.

Risikoanalyse

ID	Risiko	Mögliche Risikoursachen	Maßnahmen	%
R1	Projekt wird nicht rechtzeitig fertig	Produktmanagement- bzw. Planungsfehler, interne und externe Kommunikation funktioniert nicht, Projektumfang nicht klar	Pufferzeiten einplanen, Projekt detailliert planen	10%
R2	Umsetzung zu kostspielig	Varianten nicht ausreichend analysiert, Fehlplanung, Fehlkonstruktion	Genaue Kostenrechnung durchführen, Businessplan erstellen, innovativere Varianten konzeptionieren	50%
R3	„Projektmitarbeiter“ erledigt Arbeit nicht	„Projektmitarbeiter“ ist krank oder anderweitig beschäftigt	„Projektmitarbeiter“ übernimmt keine kritischen Aufgaben, geringeres Arbeitspensum, Pufferzeiten einplanen	45%

Tabelle 7: Risikoanalyse

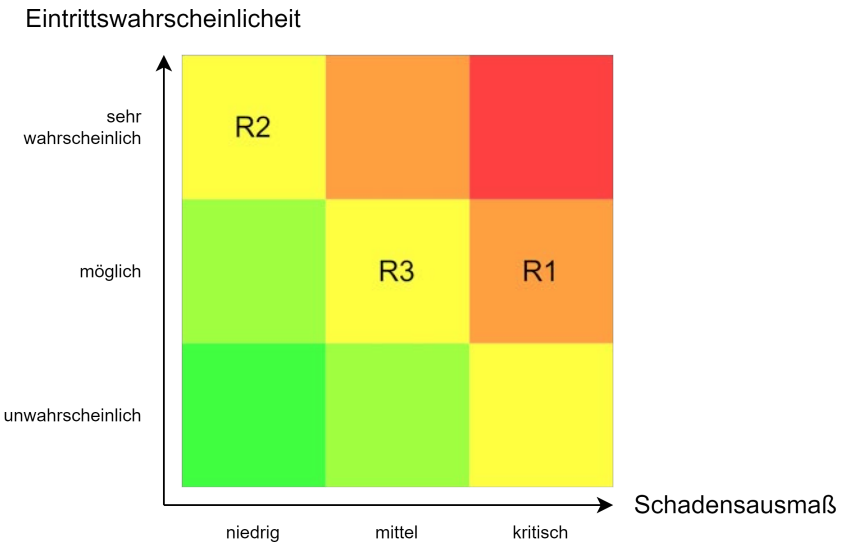


Abbildung 7: Risikoanalyse Matrix

Projektumweltanalyse (Tabellarische Form)		
Umwelt	**Risiko**	**Maßnahme Strategie**
Lieferant (Amazon) [0]	Lieferzeit zu lang oder verpätet	• Pufferzeiten einplanen • Frühzeitig bestellen

Tabelle 8: Projektumweltanalyse (Tabellarische Form)

4.8 Projektumweltanalyse

Die Projektumweltanalyse visualisiert interne und externe Beziehungen zu einem Projekt. Sie dient der Identifikation von Einflussfaktoren auf das Projekt (sowohl positive als auch negative). Anhand der Projektumweltanalyse können die Projektmitglieder die Auswirkungen der Einflussfaktoren auf das Projekt abschätzen und entsprechende Maßnahmen ergreifen. (vgl. Frankenhauser 2019f)

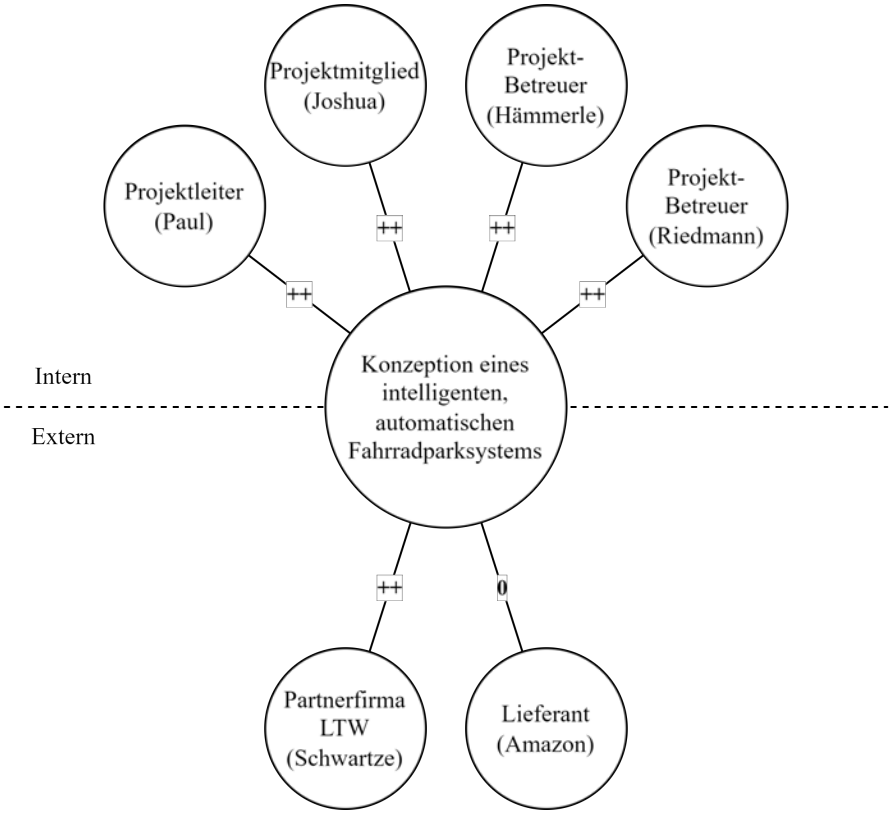


Abbildung 8: Projektumweltanalyse

4.9 Kanban

Kanban ist eine agile Methode zur Steuerung von Softwareentwicklungsprozessen. Bei der Kanban Methode wird ein Kanban-Board verwendet, dies kann entweder ein Whiteboard sein oder digital abgebildet werden (z.B. Trello, Microsoft Teams To-Do). Bei Kanban werden die einzelnen Arbeitspakete als Karten dargestellt und in verschiedene Kategorieren eingeteilt. Im Bereich der Softwareentwicklung könnten zum Beispiel die Kategorien Anforderungen, Programmierung, Dokumentation, Test und Inbetriebnahme verwendet werden. Einzelne Arbeitspakete können in vielen Fällen direkt vom Projektstrukturplan übernommen werden, werden in manchen Fällen aber noch genauer spezifiziert und aufgeteilt. (vgl. Frankenhauser 2019d)

Das Projektteam hat ursprünglich das Kanban System von Microsoft Teams verwendet, jedoch hat sich zeitnah herausgestellt, dass dieses nicht so benutzerfreundlich und flexibel ist wie Trello. Aus diesem Grund hat sich das Projektteam entschieden, auf Trello zu wechseln. Trello wurde verwendet um die schriftliche Arbeit einzuteilen und zu organisieren. Für diesen Zweck hat sich das Projektteam auf folgende Kategorien geeinigt:

- **To-Do:** Hier werden die Aufgaben aufgelistet, die noch erledigt werden müssen.
- **In Progress:** Hier werden die Aufgaben aufgelistet, die gerade bearbeitet werden.
- **Done:** Hier werden die Aufgaben aufgelistet, die bereits erledigt wurden.

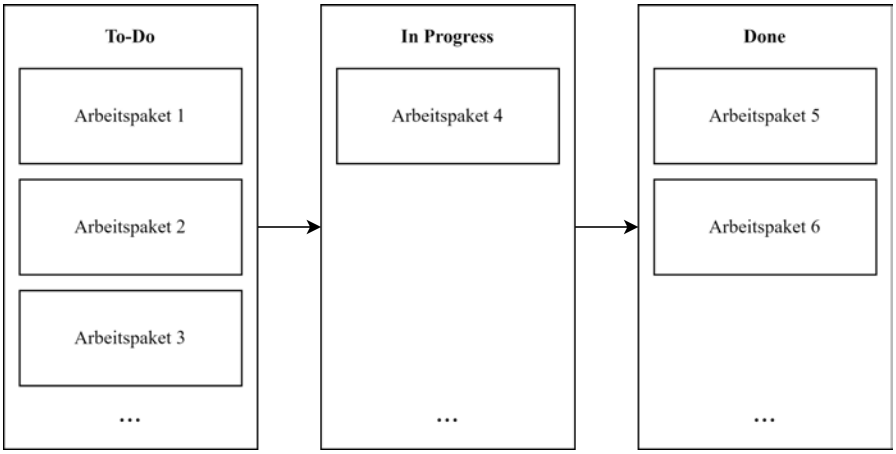


Abbildung 9: Schematische Darstellung eines Kanban-Boards

4.10 Projektfunktionendiagramm

Das Projektfunktionendiagramm zeigt die Aufgabenverteilung innerhalb des Projektes. Es wird in der Projektplanung erstellt und dient als Grundlage für die Projektorganisation. Es zeigt die Aufgabenverteilung auf die einzelnen Projektmitglieder. Die Arbeitspakete werden dem Projektstrukturplan entnommen. Diese werden dann den einzelnen Projektmitgliedern zugeordnet. Im Projektfunktionendiagramm werden sowohl direkte Projektmitglieder als auch Projektbetreuer und Partner dargestellt. (vgl. Frankenhauser 2019a)

Funktionendiagramm

Phasen / AP	Paul (PL)	Joshua	LTW	Hämmerle	Riedmann
1.1 Projektmanagement					
1.1.1 Projektstart	D	M	E	E	
1.1.2 Koordination	D	M			
1.1.3 Controlling	D	M	I		
1.1.4 Projektabschluss	D	M	M		
1.2 Analyse					
1.2.1 LTW Arbeitspaket 4.1	D				
1.2.2 LTW Arbeitspaket 4.2		D			
1.2.3 LTW Arbeitspaket 4.3	D				
1.2.4 Nutzwertanalyse erstellen	D	M		E	
1.2.5 Umfrage erstellen und auswerten	D	M	E	E	
1.2.6 Konstruktion möglicher Varianten		D	E		
1.2.7 Abschluss Analysephase	D	M	E	E	
1.3 App und Prototyp					
1.3.1 Turm Software entwickeln		D			I
1.3.2 Turm Hardware entwickeln		D			I
1.3.3 Turm Teile 3D-Drucken		D			
1.3.4 App Mockup designen	D	M			
1.3.5 App Clickable Prototype erstellen	D	M			
1.3.6 App entwickeln	D				I
1.3.7 Firebase Backend aufsetzen	D	M			I
1.3.8 Softwarekomponente mit Backend integrieren	D	M			I
1.3.9 Prototyp fertigstellen	D	M			

Funktionendiagramm

Phasen / AP	Paul (PL)	Joshua	LTW	Hämmerle	Riedmann
1.4 Testphase					
1.4.1 App testen	D				
1.4.2 Turm testen		D			
1.4.3 Integrationstests durchführen	D	M			
1.4.4 Abschluss Testphase	D	M			
1.5 Dokumentation und Schrifliche Arbeit					
1.5.1 Latex Workspace und Versionskontrolle aufsetzen		D			
1.5.2 Dokumententemplate erstellen		D			
1.5.3 Formelle Inhalte erstellen	D	M			
1.5.4 Ergebnisse der Analysephase dokumentieren	D	M	I	E	
1.5.5 App Entwicklung dokumentieren	D				
1.5.6 Turm Entwicklung dokumentieren		D			
1.5.7 Verwendete Technologie dokumentieren	D	M			
1.5.8 Fertigstellen der schriflichen Arbeit	D	M	I	E	E
1.5.9 Präsentation der Diplomarbeit	D	M			

D: Durchführungsverantwortung

M: Mitarbeit

E: Entscheidung

I: Information

Tabelle 9: Projektfunktionendiagramm

4.11 Zeiterfassung

Für die Diplomarbeit sind 180 Arbeitsstuden pro Teammitglied vorgeschrieben. Um den Arbeitsaufwand jedes Teammitglieds genau festzustellen wird das Programm Clockify verwendet. Clockify ist ein kostenfreies Tool zur Zeiterfassung. Es gibt sowohl eine Desktop Applikation als auch eine Webanwendung. Clockify vereinfacht es, die Arbeitszeit zu erfassen und zu dokumentieren. Die Zeiterfassung erfolgt über die Desktop Applikation. Die Webanwendung dient zur Einsicht und Auswertung der Daten. Die Daten können in verschiedenen Formaten exportiert werden (csv, xlsx, json). Zur graphischen Darstellung der Daten wird Excel verwendet.

Stand 31.03.2023

Arbeitsaufwand der Teammitglieder Die folgende Abbildung und Tabelle zeigt den Arbeitsaufwand der einzelnen Teammitglieder.

Teammitglied	Arbeitsaufwand
Joshua Lung	241 Stunden
Paul Hartmann	229 Stunden
Summe	470 Stunden

Tabelle 10: Arbeitsaufwand der Teammitglieder

Abbildung 10: Zeitaufwand der Teammitglieder

Arbeitsaufwand der einzelnen Projekte Die folgende Abbildung und Tabelle zeigen den Arbeitsaufwand der einzelnen Projekte.

Stand 12.03.2023

Projekt	**Arbeitsaufwand**	**Paul Hartmann**	**Joshua Lung**
Schriftliche Arbeit	108 Stunden	56 Stunden	51 Stunden
Turm Controller	93 Stunden		93 Stunden
Mobile App	83 Stunden	77 Stunden	6 Stunden
Sonstiges	43 Stunden	25 Stunden	16 Stunden
Recherche	30 Stunden	13 Stunden	13 Stunden
Konstruktion	18 Stunden	7 Stunden	7 Stunden
Turm Physical Prototype	16 Stunden		16 Stunden
Backend (Firebase)	3 Stunden	3 Stunden	
Admin Panel	3 Stunden	3 Stunden	
Summe	387 Stunden	184 Stunden	203 Stunden

Tabelle 11: Arbeitsaufwand der einzelnen Projekte

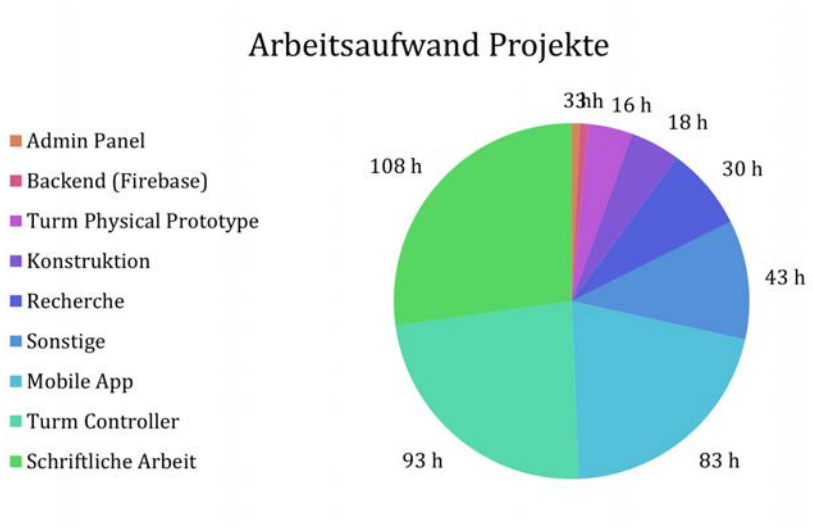


Abbildung 11: Zeitaufwand der einzelnen Projekte

5 Marktanalyse

Bei der Marktanalyse werden wichtige Merkmale des potenziellen Marktes beobachtet. Sie soll Aussicht geben, welche Modelle zum Abstellen von Fahrrädern es bereits gibt, wieso momentan zu wenig sichere Abstellmöglichkeiten vorhanden sind und welche Chancen ein vom Projektteam entwickeltes automatisches Fahrradhaus auf dem Markt hat.

5.1 Existierende Modelle

Es gibt viele unterschiedliche Möglichkeiten zum sicheren Abstellen eines Fahrrades. Das Team befasste sich bei der Recherche mit den gängigsten Varianten und modernen, automatischen Systemen.

5.1.1 Fahrradständer

Fahrradständer sind die am meisten verbreitete und kostengünstigste Variante zum Abstellen von Fahrrädern. Die Vorteile liegen bei den niedrigen Kosten und der Skalierbarkeit. Jedoch überwiegen die Nachteile, denn die Fahrräder sind äußeren Umständen wie Wetter und Vandalismus ausgesetzt. Zudem können bei gewissen Fahrradständern Felge, Rahmen und Lenkerkabel beschädigt werden. An vielen urbanen Orten sind verwahrloste Fahrradstellplätze ein großes Problem.

5.1.2 Unterirdisches Fahrradparkhaus

Unterirdische Fahrradparkhäuser sind ein aktueller Trend. Besonders verbreitet sind sie in den Niederlanden, wo sehr großzügig in die Fahrradinfrastruktur investiert wird. Die Vorteile von einem Fahrradparkhaus sind die multiplen Verwendungszwecke, zum Beispiel kann eine Reparaturstation integriert werden.

Der Nachteil ist, dass sie keinen großen Mehrwert gegenüber einem klassischen Fahrradständer oder einer Fahrradbox bieten, da zwar ein Schutz gegenüber den Umwelteinflüssen wie Regen und Schnee besteht, jedoch Vandalismus trotzdem möglich ist. Ein wesentlicher Nachteil sind die hohen Baukosten für ein entsprechendes Gebäude.

Abbildung 12: Fahrrad-Tiefgarage in Amsterdam (vgl. Weichert 2023)

Die Abbildung 12 zeigt ein unterirdisches Parkhaus in Amsterdam. Die Baukosten des Gebäudes betragen 60 Millionen Euro. Es bietet Platz für 11.000 Fahrräder.

5.1.3 Fahrradbox

Fahrradboxen bieten im Gegensatz zu einem Fahrradständer eine sichere Verwahrung des Fahrrades. Zusätzlich können andere Gegenstände wie Jacke oder Helm verstaut werden. Falls es sich um eine zweistöckige Fahrradbox handelt, führt dies zu einer geringen Nutzerfreundlichkeit, denn das Heben des Fahrrades ist nicht für alle Personen möglich. Zusätzlich benötigt eine Fahrradbox viel Platz.

Abbildung 13: Fahrradboxen in Vorarlberg (vgl. Mair 2022)

5.1.4 Wöhr Bikesafe

Die deutsche Firma Wöhr nahm 2016 den ersten sogenannten „Bikesafe“ in Betrieb (vgl. *Historie: Entwicklungen der - WÖHR Autoparksysteme GmbH* 2023). Mittlerweile wurden bereits 3 Projekte umgesetzt (vgl. *WÖHR BIKESAFE - WÖHR Autoparksysteme GmbH - WÖHR Autoparksysteme GmbH* 2023). Das System ist sowohl oberirdisch als auch unterirdisch verfügbar. Insgesamt können je nach Ausführung zwischen 58 und 192 Fahrräder gelagert werden. Die Kosten für die Variante mit 122 Stellplätzen beträgt 825.000 €, was eine Investitionssumme von ca.

6.760 € pro Stellplatz ergibt. (vgl. W. A. GmbH 2022)

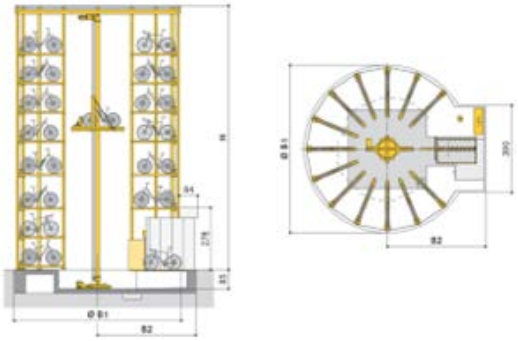


Abbildung 14: Plan vom Wöhr Bikesafe (vgl. *WÖHR Bikesafe Broschüre* 2022)

Die Abbildung 14 zeigt die Funktionsweise vom Wöhr Bikesafe. Das System hat einen runden Aufbau, die Benutzerin oder der Benutzer gibt das Fahrrad an der Vorderseite des Turmes ab. Im Inneren des Turmes befindet sich ein Bediengerät, welches das Fahrrad zu seinem Platz bringt. Der Nachteil des „Bikesafe“ sind die Schienen zum Einsetzen der Fahrräder. Damit ist das System nicht für alle Fahrradtypen geeignet. Außerdem ist die Lagerung von weiteren Gegenständen wie einer Jacke oder dem Helm, nicht möglich.

5.1.5 Eco Cycle

In Japan baut das Unternehmen „Giken LTD.“ erfolgreich das automatische Fahrradparkhaus „Eco Cycle“. Es ist das erfolgreichste automatische Parksystem und wurde bereits über 50-mal gebaut. (vgl. Ginsberg 2020) Das System kann als Turm oder unterirdisch gebaut werden. Die Funktionsweise ist vergleichbar mit dem „Wöhr Bikesafe“: Das Fahrrad wird in eine Schiene gestellt und mit einem Lagerbediengerät zu einem freien Platz befördert. Es bietet Platz für bis zu 204 Fahrräder.2023a Zudem gibt es eine mobile Version des Systems, die leicht auf- und abgebaut werden kann.2023b

Abbildung 15: Eco Cycle (vgl. CNN 2016)

Die Nachteile des Systems sind ähnlich wie beim „Wöhr Bikesafe“, denn es ist nicht für alle Fahrradtypen oder E-Scooter geeignet und bietet keine alternativen Verwendungszwecke. Die unterirdische Variante ist außerdem nicht überall baulich möglich.

5.1.6 V-Locker

Die 2019 gegründete Schweizer Firma V-Locker betreibt bisher 4 automatische Systeme zur Lagerung von Fahrrädern. (vgl. Borkert 2020) Weitere Projekte sind bereits in Planung. (vgl. AG 2023a)
Das Prinzip des Mechanismus ähnelt einem Paternoster: An der Vorderseite können Fahrräder eingelagert werden, innen fahren die Boxen in einem Kreis.
Die Investitionssumme des Systems beträgt je nach Ausführung zwischen 5.000 € bis 6.000 € pro Stellplatz. (vgl. Peter 2020a) Der Nachteil des Systems liegt an der geringen Größe, es bietet nur Platz für maximal 20 Fahrräder.

Abbildung 16: Funktionsweise vom V-Locker (vgl. AG 2023b)

5.2 Nutzwertanalyse

Mithilfe einer Nutzwertanalyse wurden elf neue und vorhandene Systeme zur Lagerung von Fahrrädern bewertet. Drei der bewerteten Systeme wurden von Grund auf selbst konzipiert. Anschließend wurden die Systeme anhand 24 Kriterien mit 1-5 Punkten bewertet.
Schwer quantifizierbare Aspekte, wie z.B. die Nutzerfreundlichkeit oder Ausfallsicherheit wurden basierend auf den Ergebnissen der Umfrage und durch Abstimmungen innerhalb der Gruppe bewertet. Einfacher quantifizierbare Kategorien wie Flächenverbrauch, Baukosten und die durchschnittliche Einlagerungszeit wurden anhand des Logarithmus der Merkmalsausprägung mit 1-5 Punkten versehen.

Quantitative Kriterien Die quantitativen Kriterien wurden aus verschiedenen Quellen (vorallem Herstellerseiten und Angebote) zusammengetragen und in Tabelle 12 dargestellt.

Qualitative Kriterien Die qualitativen Kriterien wurden innerhalb des Teams festgelegt. Die qualitative Bewertung der einzelnen Kriterien kann in Tabelle 13 eingesehen werden.

Quantitative Kriterien

Variante	Investitionssumme (€)	Platzverbrauch (m^2)	Kapazität (Stk.)	Investitionssumme pro Fahrrad (€/Stk.)	Platzverbrauch pro Fahrrad (m^2/Stk.)	Fahrraddurchsatz (Stk./h)	Durchschnittliche Einlagerungszeit (s)
Bestehende Systeme							
Paternoster 20 Räder (Vlocker)	100.000	[1] 5,57	[2] 20	[3] 5.000	0.28	?	?
Rondell (Wöhr Bikesafe)	[4] 825.000	[5] 50,00	[6] 122	6.800	0.41	225	[7] 16
Rondell Unterirdisch (Eco Cycle)	[8] 1.400.000	[9] 8,56	[10] 204	6.900	0.04	280	[11] 13
Fahrradbox	[12] 32.500	[13] 24,10	[14] 30	1.100	0.80	-	-
Fahrradständer	-	-	-	[15] 53	[16] 2.1	-	-
Unterirdischer Fahrradständer (Amsterdam Centraal)	[17] 60.000.000	-	[18] 11.000	5.500	-	-	-
Neue Systeme							
Rondell Auslagerung von Außen	[19] 1.461.000	[20] 21,52	[21] 104	14.000	0.21	[22] 70	51
Hochregallager	[23] 1.005.000	[24] 115,96	[25] 212	4.700	0.55	[26] 70	51

Tabelle 12: Quantitative Kriterien

Qualitative Kriterien

Variante	Skalierbarkeit	Diebstahlsicherheit	Brandsicherheit	Stärungssicherheit	Nutzerfreundlichkeit	Sichtbarkeit	Gestaltbarkeit	Zusatznutzen
Bestehende Systeme								
Paternoster 20 Räder (Vlocker)	2	5	2	2	4	4	4	3
Rondell (Wöhr Bikesafe)	4	5	4	3	4	5	4	4
Rondell Unterirdisch (Eco Cycle)	1	5	3	3	4	2	2	3
Fahrradbox	4	4	3	4	4	2	3	3
Fahrradständer	5	1	4	5	4	2	1	1
Unterirdischer Fahrradständer (Amsterdam Centraal)	2	2	2	5	3	1	1	1
Neue Systeme								
Rondell Auslagerung von Außen	4	5	4	4	4	5	3	4
Hochregallager	3	5	3	3	4	4	4	4

Tabelle 13: Qualitative Kriterien

5.2.1 Gegenüberstellung

Anhand der ermittelten Parameter können die einzelnen Modelle miteinander verglichen werden. Zum Beispiel kann eine Gegenüberstellungsmatrix mit zwei oder mehrere Parameter wie in der folgenden Abbildung 17 darstellen.

Kategorien: Investitionskosten / Fahrrad, Platzverbrauch / Fahrrad

Modelle: Rondell, Paternoster, Hochregallager, Fahrradbox, Fahrradständer, Unterirdisches Fahrradparkhaus

Anhand dieser Kategorien und Modelle wird ein Diagramm erstellt, in der die einzelnen nach den Kriterien bewerteten Modelle gegenübergestellt werden. Anhand dieser Matrix können Nutzbarkeit und Einsatzgebiete der einzelnen Modelle verglichen werden. Die Matrix erleichtert es Entscheidungen über das geeignete System zu treffen und potenzielle Marktlücken zu identifizieren.

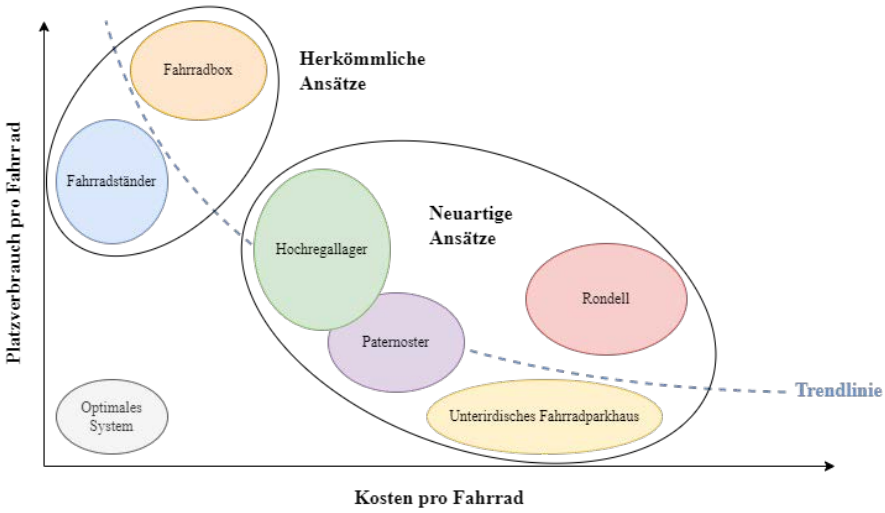


Abbildung 17: Gegenüberstellung der Modelle

Anhand der Ergebnisse dieser Nutzwertanalyse, wurden zwei vielversprechende Modelle zur weiteren Bearbeitung gewählt. Diese Modelle werden in den nächsten Kapiteln näher betrachtet und beschrieben. Die Ausgewählten Modelle sind: **Rondell** und **Hochregallager**. Das Rondell ist eine Adaptierung bestehender Systeme, das Hochregallager ist ein neues System, welches auf herkömmlichen Hochregallagern aufbaut.

5.3 Erweiterung der Nutzungstiefe

Im Laufe der Arbeit stellt sich heraus, dass das alleinige Lagern von Fahrrädern das Potential des Systems zu stark limitiert. Wenn eine Vielzahl an Gegenständen gelagert werden kann, erhöht dies die Nutzungstiefe, da mehr Personen angesprochen und verschiedene Verwendungszwecke möglich sind.

Es ist möglich andere Unternehmen wie zum Beispiel Paketdienste oder Leihanbieter in das automatische Fahrradparkhaus einzubinden. Das Projektteam kann sich folgende Verwendungszwecke vorstellen, die keine technischen Änderungen des Fahrradparkhauses erfordern:

5.3.1 Vermieten von Fahrrädern

Die Vermietung von Fahrrädern kann über das Fahrradparkhaus erfolgen. Die Vorteile sind Schutz vor äußeren Einflüssen, gute Sichtbarkeit und unabhängige Buchung des Fahrrades.

5.3.2 Vermieten von E-Scootern

Immer mehr Städte verbieten das Abstellen von E-Scooter auf Gehsteigen (vgl. Krutzler 2022). Durch eine Kooperation mit einem Anbieter von E-Scootern, kann das Ausleihen über das automatische Fahrradparkhaus erfolgen. Es bietet Vorteile für die Anbieterfirma, Nutzende der E-Scooter und der Allgemeinheit, da die E-Scooter niemanden auf dem Gehsteig behindern und sicher verwahrt sind.

5.3.3 Abstellen von Gegenständen

Ein weiterer Verwendungszweck ist die Abgabe von Gegenständen. Ein geeigneter Standort ist zum Beispiel ein Bahnhof in einer touristischen Stadt. Dort können Touristen ihren Rucksack oder Koffer einlagern, wärend sie die Stadt erkunden.

5.3.4 Abgabe und Abholung von Paketen

Das automatische Fahrradparkhaus kann außerdem als Paketshop fungieren. Somit wird eine zeitunabhängige Abgabe bzw. Abholung von Paketen auf dem Arbeitsweg ermöglicht. Grundlage dafür ist eine Kooperation mit einem Paketdienst wie zum Beispiel DPD oder DHL.

5.3.5 Abgabe zur Fahrradreparatur

Ein weiterer Verwendungszweck ist, dass man ein Fahrrad zur Reparatur abgeben kann. Dabei kann man das Fahrrad am Morgen in Fahrradparkhaus lagern und am Abend wieder abholen. Für Benutzer:innen des Fahrradparkhauses, gibt es den Vorteil, dass sie keinen Umweg zur Fahrradwerkstatt machen müssen. Die Fahrradwerkstatt profitiert durch neue Kund:innen. Grundlegend dafür ist, dass der Werkstatt Zugang zum Fahrrad gewährt wird.

5.4 Kundensegmente

Als Kundschaft des Systems kommen sowohl öffentliche als auch private Einrichtungen in Frage.

5.4.1 Großunternehmen

Häufig besteht ein beträchtlicher Abteil des CO^2-Fußabrucks eines Unternehmens aus dem Arbeitsweg der Mitarbeitenden, die derzeit noch oft mit dem PKW zur Arbeit fahren. Viele Unternehmen bauen aufgrund der steigenden Anzahl der Mitarbeiter:innen, teure Parkplätze (vgl. parkhere 2020). Ein durchschnittlicher Parkplatz benötigt wertvolle Flächen und der Bau eines Auto-Parkhauses kostet viel Geld. Ein automatisches Fahrradparkhaus bietet sicheres Verstauen der Fahrräder der Mitarbeiter:innen mit einem geringen Flächenverbrauch und Anwohner:innen profitieren von reduzierter Verkehrsbelastung.

5.4.2 Wohnbau-Unternehmen

Beim Bau von großen Wohnanlagen kann ein automatisches Fahrradparkhaus einen Fahrradraum ersetzen. Je größer das Projekt ist, desto rentabler ist der Bau eines Fahrradparkhauses. Dieses kann entweder freistehend oder an der Außenseite eines Wohngebäudes platziert werden.

5.4.3 Öffentliche Einrichtungen

Öffentliche Einrichtungen wie das Land Vorarlberg sind bereit, hohe Summen in den Ausbau der Radinfrastruktur zu investieren, um den motorisierten Verkehr zu reduzieren (vgl. v. O. a. r. v. 2023). Ein automatisches Fahrradparkhaus macht die Verwendung von Fahrrädern attraktiver und kann auch für Einnahmen sorgen.

5.5 Öffentliches Verkehrsnetz und -verbünde

Ein automatisches Fahrradparkhaus kann bei einem größeren Bahnhof installiert werden. Es benötigt wenig Platz und bietet eine sichere, attraktive Aufbewahrung für Fahrräder. Damit werden mehr Menschen zur Nutzung von öffentlichen Verkehrsmitteln bewegt. Zusätzlich kann das System für weitere Zwecke, wie dem Ausleihen von Fahrrädern oder dem Verstauen von Koffern, verwendet werden.

5.6 Verkehrspolitische und kommerzielle Rahmenbedingungen

Fahrradfahren hat zahlreiche Vorteile: Es ist wesentlich umweltfreundlicher und günstiger als die Benutzung eines PKWs und fördert die eigene Gesundheit. Die EU hat sich zum Ziel gesetzt, die Zahl der per Rad zurückgelegten Kilometer bis 2030 auf 312 Milliarden zu verdoppeln, um den Klimawandel zu bekämpfen und die Verkehrsbelastung in Städten zu reduzieren (vgl. Lüdecke 2023). Dabei spielen intelligente Fahrradparksysteme eine wichtige Rolle bei der Förderung des

Fahrradverkehrs. Kommunen und Städte können die Schaffung von Fahrradparkhäusern unterstützen, indem sie Anreize für Unternehmen und Verkehrsverbünde bieten oder selbst in den Bau von Fahrradparkhäusern investieren.

Aktuelle Gesetze und Verordnungen in Österreich bevorzugen das Auto gegenüber dem Fahrrad. Besonders beim Bau von Wohnungen oder Betrieben ist die verordnete Mindestanzahl von Fahrradstellplätzen geringer als die Mindestanzahl von Stellplätzen für Kraftfahrzeuge (vgl. Vorarlberger Landesregierung 2020). So ist beim Bau von einem Supermarkt 1 Parkplatz pro $30m^2$ Geschäftsfläche und nur ein Fahrradstellplatz pro $50m^2$ erforderlich (vgl. *RIS - Stellplatzverordnung - Landesrecht konsolidiert Vorarlberg, Fassung vom 09.03.2023* 2023).

5.7 Umfrage

Um besser nachzuvollziehen, was die Meinung von potenziellen Kunden zu solch einem System ist, wurde eine Online-Umfrage erstellt, die auf Internet-Foren zum Thema Fahrradfahren veröffentlicht wurde. Dabei geben die Teilnehmer:innen Antworten über die persönliche Nutzung von Fahrrädern, bewerten verschiedene Modelle und geben Auskunft darüber, ob sie ein solches automatisches Fahrradparkhaus verwenden würden.
Insgesamt verzeichnet die Umfrage 259 Teilnehmer aus dem deutschsprachigen Raum.
Die Umfrage zeigt klar, dass die meisten Fahrradfahrerinnen und Fahrradfahrer mit klassischen Methoden zum Abstellen von Fahrräder, wie einem Fahrradständer, unzufrieden sind. Außerdem können sich der Großteil der Befragten die Nutzung eines automatischen Fahrradparkhaus vorstellen und wäre auch bereit einen gewissen Betrag dafür zu bezahlen.

5.8 Schlussfolgerung

Die Umfrage zeigt, dass viele der Radfahrer:innen unzufrieden mit den klassischen Varianten zum Abstellen von Fahrrädern sind und es Bereitschaft gibt, ein automatisches Fahrradparkhaus zu verwenden und dafür zu zahlen.

In Europa ist der Markt für ein automatisches Fahrradparkhaus noch offen. Die Mitbewerber:innen haben sich noch nicht etabliert, jedoch zeigt die steigende Anzahl umgesetzter Projekte und der Erfolg der Firma „Giken LTD.“ in Japan, dass es sich um einen Trend handelt und noch viel Potential offen ist.

Durch die steigende Investitionsbereitschaft durch den Staat, sieht das Projektteam, dass ein vom Partnerunternehmen entwickeltes Fahrradparkhaus Potential auf dem europäischen Markt hat und es eine Vielzahl potenzieller Käufer:innen gibt.

6 Produktkonzeption

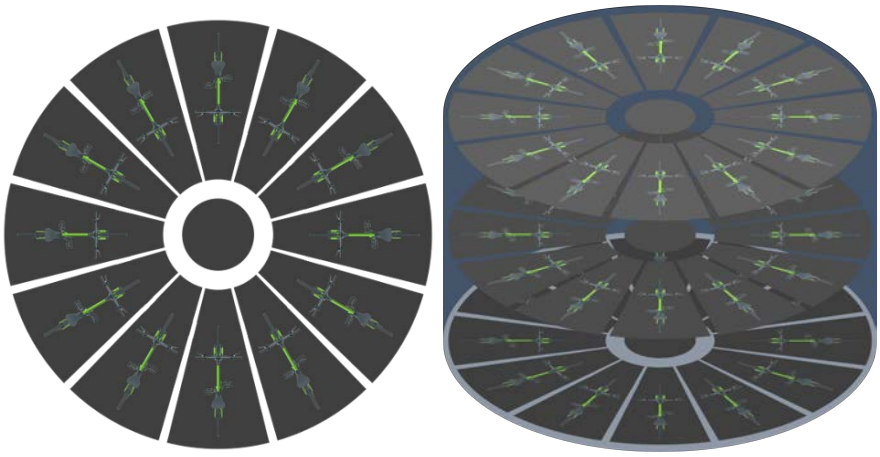
Bei der Produktkonzeption werden zwei neue Systeme zur Lagerung von Fahrrädern entwickelt. Es wird gezeigt wie sich die Systeme von der Konkurrenz abheben und welche Funktionen integriert werden können.

6.1 Varianten

Anhand der Nutzwertanalyse werden die zwei vielversprechendsten Systeme weiter untersucht. Dabei werden die Varianten „Rondell“ und „Hochregallager“ untersucht.

6.1.1 Rondell System

Eine der zwei vorgestellten Varianten basiert teilweise auf dem Rondell-Modell der Konkurrenz, jedoch mit dem Unterschied, dass die Fahrräder von außen, anstatt von innen ausgelagert werden. Zudem werden die Fahrräder nicht auf Schienen, sondern in Boxen gelagert, welche dann zusammen ein und ausgelagert werden. Die abgeschlossenen Boxen verringern zwar die Platzeffizienz, jedoch sind sie sicherer und erlauben höheren Zusatznutzen. Das System ist hinsichtlich der ringförmigen Anordnung der Fahrräder sehr ähnlich wie bestehende Varianten. Je nach Bedarf können unterschiedlich viele Ebenen übereinandergestapelt werden.



(a) Ringförmige Anordnung der Fahrräder

(b) Ringförmige Anordnung der Fahrräder auf mehreren Ebenen

Auslagerung Bei einer ringförmigen Formation an Lagerplätzen bieten sich zwei Möglichkeiten an, Fahrräder zu entnehmen:

Auslagerung von innen Bei der Auslagerung von innen werden die Fahrräder mittels eines, in der Mitte gelegenen, Lagerroboters aus- und eingelagert. Herkömmliche Fahrradlagersysteme verwenden diese Variante. Dieses System hat den Vorteil, dass mit einem einzelnen Lagerroboter alle Fahrräder bewegt werden können. Gleichzeitig bedeutet dies aber auch, dass dieser Roboter alle Fahrräder bewegen muss, was zu Staus führen kann. Ein weiterer Nachteil ist, dass durch die vorliegende Geometrie eines solchen Systems viel Platz verschwendet wird. Dies ist vor allem der Fall, wenn die Fahrräder in Boxen gelagert werden.

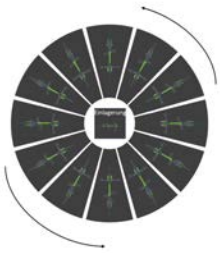


Abbildung 19: Rondell System mit Auslagerung von innen

Auslagerung von außen Bei der Auslagerung von außen werden die Fahrräder mittels eines Lifts von außen manipuliert. Im Gegensatz zur Auslagerung von innen drehen sich hier die einzelnen Ebenen und nicht der Lagerroboter selbst. Je nach Bedarf kann die Anzahl der Lagerroboter angepasst werden und optimal auf die Anforderungen des Standorts zugeschnitten werden.

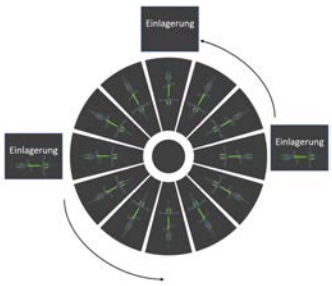
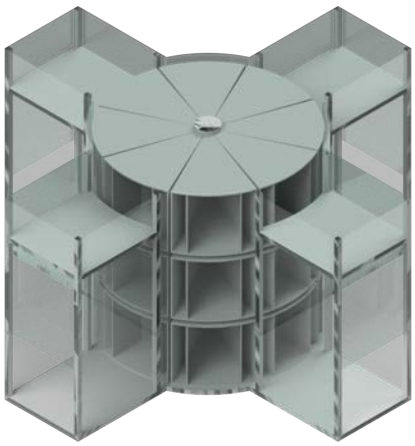
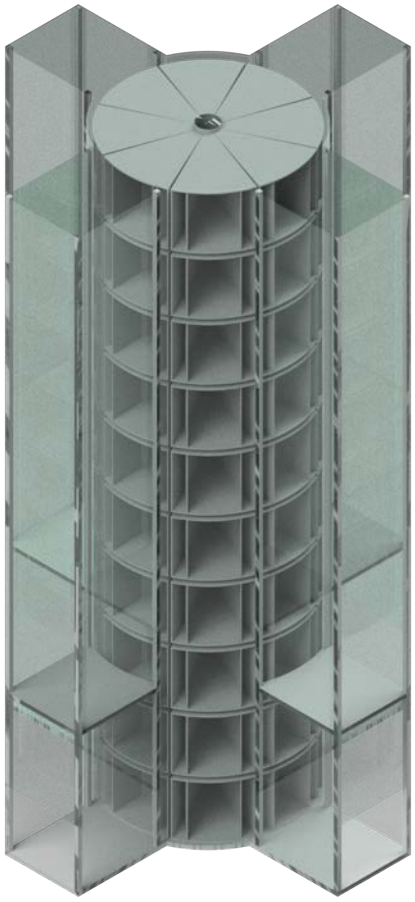


Abbildung 20: Rondell System mit Auslagerung von außen

Mögliche Umsetzung Das System erlaubt große Flexibilität hinsichtlich Umsetzung und kann, falls benötigt, auch modular erweitert oder verkleinert werden. Parameter, welche je nach Anwendung verändebar sind, beinhalten Anzahl an Fahrrädern pro Ebene, Anzahl an Ebenen und Anzahl an Einlagerungsliften. Ein Beispiel für eine solche Variante ist auf der folgenden Abbildung zu sehen.



(a) Rondell System mit 3 Ebenen



(b) Rondell System mit 10 Ebenen

Konstruktionsgetriebene Analysen Um den optimalen Platzverbrauch zu garantieren, wurde die technische Zeichnung mit verschiedenen Parametergrößen untersucht. Eine dieser Größen ist der Durchmesser des „Rückgrats“ des Turms, in dem die ganze Technik verbaut ist und an der die Ebenen gelagert werden.

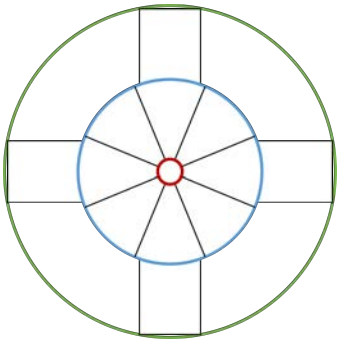


Abbildung 22: Konstruktionsgrößenanalyse

Umso kleiner dieser Durchmesser, desto weniger Fahrräder können gelagert werden. Um den optimalen Wert zu finden, wurden Werte zwischen $0.1m - 2.1m$ im Abstand von $0.1m$ getestet. Laut den Ergebnissen des Tests gibt es einen optimalen Wert, um die „Minimum“ Fläche (Blau) zu minimieren $d = 1.4m$, um die „Maximum“ Fläche (Grün) zu minimieren, muss ein Durchmesser von über $2.2m$ gewählt werden.

Vorteile des Rondell Systems

- Gute Sichtbarkeit
- Hoher Wiedererkennungswert
- Optimales Einsatzgebiet in Parks und anderen nicht-rechteckigen Grundstücken.
- Einfache Erweiterbarkeit
- Fahrraddurchsatz je nach Anzahl Lagerroboter (Erweiterbar)
- Einfache Wartung, da Lagerroboter auf der Außenseite angebracht sind

Nachteile des Rondell Systems

- Platzverschwendung durch pizzastückförmige Boxen. (Rechteckige Boxen passen besser auf die Form der Fahrräder)
- Es werden mehrere Lagerroboter benötigt (Je nach Konfiguration)
- Kompliziertes Lagersystem (Ein- und Aushängen der „Pizzastücke“, Rotieren der gesamten Ebene)

- Neuartige Lagertechnologie, kann sich nicht auf bestehende Systeme stützen

6.1.2 Hochregallager System

Die Partnerfirma des Projekts entwickelt und produziert klassische automatische Hochregallager. *Warum also nicht deren Produkte verwenden?*

Ein Fahrradturm mit LTW Produkten wird vom Projektteam folgendermaßen skizziert: Im Erdgeschoss gibt es die Einlagerungsplätze. Jeder dieser Plätze hat von vorne mit Rolltoren verschließbare Öffnungen, an welche die Boxen rückseitig geschoben werden. Das System erkennt nach ein paar Tagen automatisch die Hauptnutzungszeiten, und weiß, wann aus oder eingelagert wird. Es besteht die Möglichkeit, dass die App dem Turm meldet, wenn sich jemand dem Turm nähert und dann dessen Box schon im vorherein in einen freien Platz gestellt wird. Kommt es dann zur Auslagerung muss nur noch das Rolltor und die Boxentüre geöffnet werden und das Regalbediengerät (RBG) kann im Hintergrund andere Aufgaben erledigen.
Um nach einem Vorgang prüfen zu können, ob die Box wieder leer ist, kommen Höhensensoren, Wärmebildkameras und Bewegungsmelder zum Einsatz. Ist der Platz wirklich menschenleer, wird die Box von hinten vom Regalbediengerät aufgenommen. Dahinter ist der Schacht, durch den die Boxen verfahren werden. In den in beliebiger Anzahl vorhandenen Ebenen befinden sich die Lagerplätze. Je nach Ausführung wird hier beidseitig oder einseitig gelagert.
Das RBG verfährt nur in horizontaler und gleichzeitig in vertikaler Richtung zu einem freien Platz. Dort angekommen wird die Box in das Regal übergeschoben.

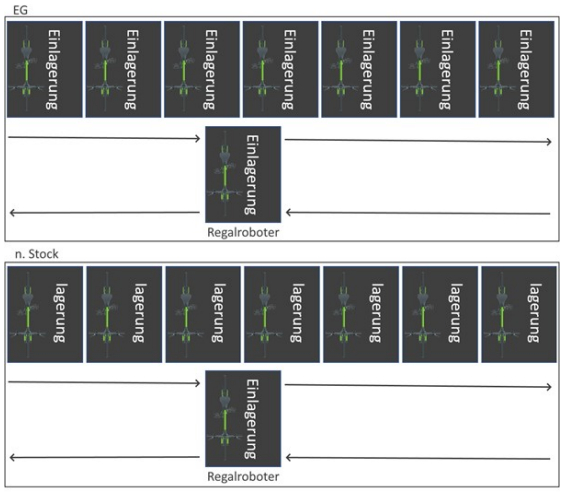


Abbildung 23: Hochregallager 2D Ansicht

Je nach den lokalen Anforderungen kann das Konzept variieren. Mehrere Einlagerungsplätze einzubauen ist dabei am billigsten. Reicht auch das nicht aus, kann die Anzahl der RBG erhöht werden, es gibt Systeme, allerdings in Lagern, bei denen mehrere Roboter gleichzeitig im Gebäude in Betrieb sind.
Reicht die Lageranzahl nicht aus, besteht im Gegensatz zum Rondell, neben der Möglichkeit höher zu bauen, auch die Variante die Grundfläche zu erhöhen. Zieht man die Gebäudegeometrie so in die Länge, erhöht sich die Anzahl an Stellplätzen proportional um den doppelten Faktor der übereinander angeordneten Boxen.

Abbildung 24: Hochregallager mit doppelten Plätzen, 2D Ansicht

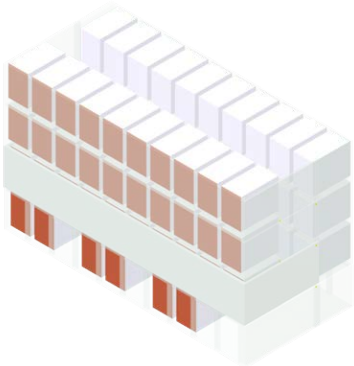


Abbildung 25: Hochregallager 3D Ansicht

6.2 Innovation

Grundsätzlich bietet das System kurz- und mittelfristige Lagerung von Fahrrädern zu einem geringen Nutzpreis an. Dies passiert auf geringstem Raum, gewährleistet jedoch Sicherheit vor Diebstahl und Umwelteinflüssen, im Gegensatz zu herkömmlichen Fahrradständern. Nutzer können ihr Fahrrad mit einer App ganz einfach lagern und abholen, zudem kann man in der App Infos zu nahegelegenen Türmen erhalten.

Es gibt schon einige andere Firmen, die automatische Fahrradparksysteme anbieten, bis jetzt hat sich jedoch kein Konkurrent als Markführer etabliert. Das System lernt aus den Fehlern bestehender Fahrradlagertürme und hebt sich mit einigen grundlegenden Features von der Konkurrenz ab.

Die vom Projektteam entworfenen Fahrradtürme sind mobil und modular, das bedeutet, dass die Türme aus einzelnen Segmenten bestehen, die je nach Bedarf an verschiedenen Orten auf- und abgebaut werden können. Zum Beispiel kann bei einem großen Event, ein Fahrradturm temporär aufgebaut werden, um der erhöhten Nachfrage nachzukommen.

Im Gegensatz zu Konkurrenten limitiert sich das System nicht ausschließlich auf Fahrräder, denn andere muskel- und E-motorbetriebene Kleinfahrzeuge werden immer häufiger. Der Turm ist so konzipiert, dass alle möglichen Fahrzeuge und Gegenstände gelagert werden können. So verbreitert das Projektteam das Einsatzspektrum und erhöht zugleich die Anzahl potenzieller Kunden:innen. Auch Pakete können in vorgesehenen Plätzen kurzfristig gelagert werden. Das ermöglicht Kooperationen mit etwaigen Versandunternehmen, sodass ein Paket einfach vom Lieferanten zu deinem Fahrrad gelegt werden kann.

(a) Lagerung eines Fahrrads

(b) Lagerung eines Pakets

7 Konzept des Prototypen

Um das Benutzen des Fahrradturms besser darstellen zu können, hat das Projektteam einen Prototypen entwickelt. Der Prototyp umfasst eine App und eine Miniaturversion des Fahrradturms.

7.1 Turm

Um den Aus- und Einlagerungsvorgang besser darzustellenm, hat das Projektteam ein Modell des Turms gebaut. Dieser zeigt den aktuellen Status (belegte und ausgelagerte Stellplätze) des Turms und reagiert auf Anfragen des Clients. Es ist ein Prototyp des Turms, welcher die Funktionalität des Turmes simuliert. Mittels RGB LEDs werden die Stellplätze des Turmes dargestellt.

Neben der Hardware, beinhaltet der Turm auch Software um als Backend für die App zu dienen. Jeder Turm dient als eigenständige Einheit und kann Anfragen selbstständig beantworten.

Da das Projektteam sowohl die 3D-Drucker der HTL, als auch einen eigenen 3D-Drucker zur Verfügung hat, besteht der Turm aus 3D-Druckteilen. 3D-Druck erlaubt dem Projektteam, den Turm sehr schnell und ohne viel Aufwand zu bauen. Zudem erlaubt es einen sehr detailierteren Turm zu fertigen.

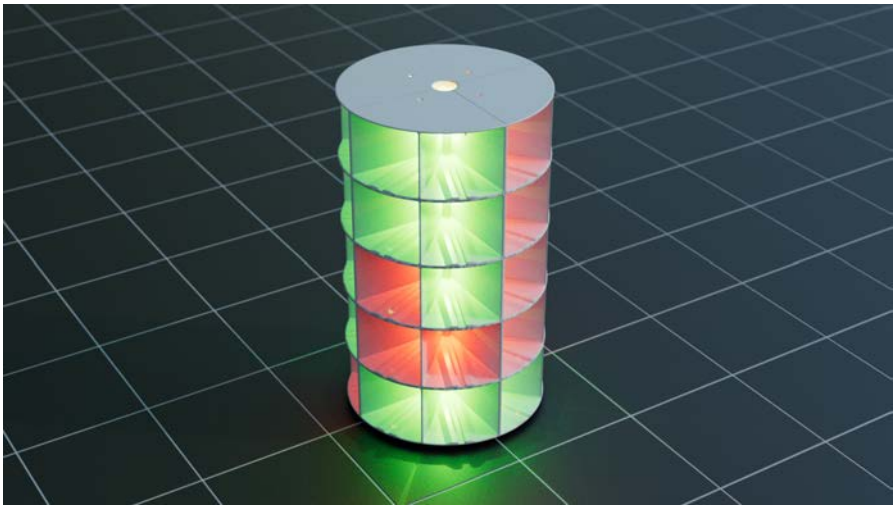


Abbildung 27: Visualisierung des Modell-Turms

7.2 App

Bei der Konzeptionierung der App werden die Funktionsweise, das Design und die Technologien zur Entwicklung festgelegt. Die App soll dazu dienen, dass anwendende Personen Möglichkeiten zum Lagern von Fahrrädern finden und benutzen können. Sowohl die Einlagerung als auch die Auslagerung des Fahrrades oder Gegenstands wird über die App gesteuert werden.

7.2.1 Funktionen

Die App wird folgende Funktionen haben:

- Karte zum Suchen von Abstellmöglichkeiten für Fahrräder
- Möglichkeit zum Erstellen von Account
- Anzeige der Buchungen von Fahrradabstellplätzen
- Ein- und Auslagerungsprozess kann von der anwendenden Person aktiviert werden

7.2.2 Technologie

Um die App zu entwickeln, wird ein Framework benötigt, das eine gemeinsame Codebasis bietet, um die App für iOS und Android zugänglich zu machen. Es wird eine zeitsparende und effiziente Entwicklung gewährt. Dafür bieten sich u.a. folgende Technologien an:

React Native Dabei handelt es sich um eine Open-Source-Bibliothek, die von Facebook entwickelt wurde und auf JavaScript basiert. React Native ermöglicht die Entwicklung von mobilen Apps für iOS und Android mit einer Codebasis.(siehe 10.5)

Xamarin Xamarin ist eine plattformübergreifende Entwicklungsumgebung, die von Microsoft entwickelt wurde. Xamarin ermöglicht die Entwicklung von Apps für iOS, Android und Windows mit einer Codebasis und unterstützt die Programmiersprachen C# und F#. (vgl. anna.cherniavska 2021)

Flutter Flutter ist eine von Google entwickelte Open-Source-Bibliothek, die auf der Programmiersprache Dart basiert. Flutter ermöglicht die Entwicklung von mobilen Apps für iOS, Android und das Web mit einer Codebasis. (vgl. *Flutter - Build apps for any screen* 2023)

Entscheidung für React Native Für die Entwicklung der App verwendet das Projektteam das Framework React Native. Mit React Native benötigt man nur einen Code für die Entwicklung der App, die auf Android und iOS funktioniert. Es ermöglicht eine schnelle und unkomplizierte Entwicklung der App.(vgl. *Vor- und Nachteile von React Native für die Entwicklung mobiler Apps* 2023) Bei der Entscheidung für React Native, ist es außerdem bedeutend, dass das Projektteam bereits Erfahrungen mit React, dessen Funktionsweise ähnlich ist, gesammelt hat. React Native eignet sich zudem gut für die Integration von Firebase, der Entwicklungs-Plattform, die für die Entwicklung verwendet wird (siehe 10.1), welche sehr gut dokumentiert ist. (vgl. *React Native · Learn once, write anywhere* 2022)

Entscheidung für Expo Das Projektteam hat sich bei der App für das Framework Expo (10.7), welches auf React Native basiert, entschieden. Damit kann man die App auf iOS und Android testen und vorführen. Man muss dabei keine Entwicklungsumgebung einrichten und spart wiederum Zeit. Das Framework ist sehr gut dokumentiert und die verschiedenen Bibliotheken eignen sich für die benötigten Funktionen der App.

7.2.3 Mockups

Für die Erstellung der Mockups wird das Programm Figma verwendet. Innerhalb eines geteilten Workspaces werden erste Entwürfe für die fünf verschiedenen Menüs der Applikation erstellt. Damit wird gezeigt, welche Funktionen die App haben wird und wie das Design aussehen wird.

Map Der Tab „Map“ dient zur Darstellung der verfügbaren Möglichkeiten zur Lagerung von Fahrrädern in der Umgebung auf einer Karte. Mit einem Marker wird ein Fahrradturm dargestellt. Außerdem soll die Verfügbarkeit von freien Plätzen dargestellt werden.

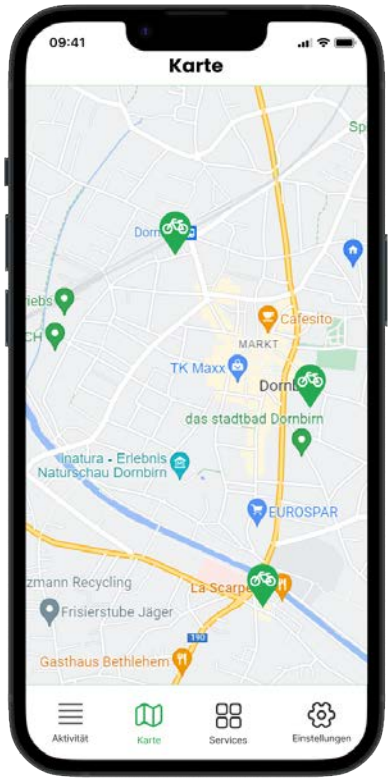


Abbildung 28: Mockup vom Screen Map

Tower Der Tab „Tower“ dient zur Anzeige folgender Informationen:

- Karte mit Standort des Turmes
- Anzahl verfügbarer Lagerungsmöglichkeiten
- Bereits von der anwendenden Person bei diesem Turm gelagerte Gegenstände
- Möglichkeit zum Einlagern von einem weiteren Rad oder Gegenstand

Abbildung 29: Mockup vom Screen Tower

Aktivität Im oberen Teil vom Tab „Aktivität“ werden die von der anwendenden Person gelagerten Gegenstände angezeigt. Auf der unteren Hälfte werden vergangene Buchungen mit weiteren Informationen aufgelistet. Falls man auf eine Buchung klickt, sollen weitere Infos erscheinen.

Abbildung 30: Mockup vom Screen Aktivität

Einstellungen Der Tab „Einstellungen“ bietet die Möglichkeit, zu verschiedenen Unterseiten zu gelangen, wo man wichtige Einstellungen treffen und Informationen abrufen kann.

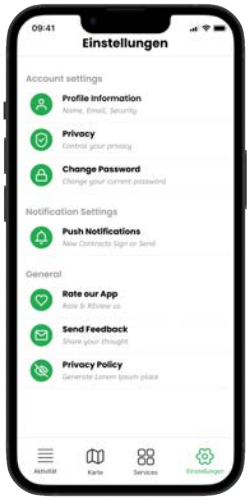


Abbildung 31: Mockup vom Screen Einstellungen

Services Der Tab „Services“ dient zum Darstellen verschiedener Dienste. Es gibt die Möglichkeit zum Teilen eines Entsperrungscodes für ein gelagertes Fahrrad und die Eingabe eines solchen Codes. Außerdem werden mehrere externe Dienstleistungsmöglichkeiten zum Beispiel für das Ausleihen eines E-Scooters oder zum Abgeben von Paketen aufgelistet.

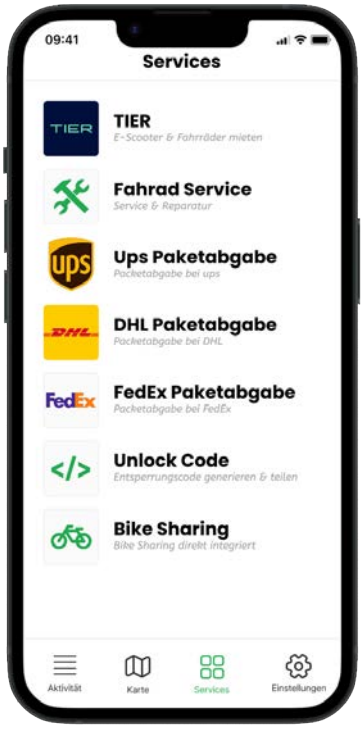


Abbildung 32: Mockup vom Screen Services

7.3 Datenbank

> „*Cloud Firestore is optimized for storing large collections of small documents.*“ (vgl. *Cloud Firestore-Datenmodell | Firebase* 2023)

Die Grundidee ist, die Daten schon so weit vorbereitet zu speichern, dass sie mit so wenig Aufwand wie möglich abgerufen und ausgegeben werden können. Um hingegen möglichst effizient zu sein, sollen die Collections möglichst groß, die Dokumente hingegen möglichst klein sein.

7.3.1 Arten von Datenverknüpfung

Um Daten zu verknüpfen gibt es für das Projektteam zwei Varianten:

Tiefe Ebene Die Verknüpfung wird als Spalte mit dem eindeutigen Namen des zu Verknüpfenden, als Booleanwert true gespeichert. Dadurch kann die Datenbank einfach auf andere Datenbanken verschoben werden, es wird aber auch bei der Abfrage mehr Rechenpower benötigt.

Höhere Ebene In Firestore gibt es die Möglichkeit Referenzen zu erstellen. Dies stellt die beste Art dar, da diese speziell auf Firestore optimiert wurde und das Fehlerpotential durch eine automatische Überprüfung der Referenz minimiert werden kann. Die Verknüpfung erfolgt dabei auf den Dokumentennamen.

8 Umsetzung des Prototypen

Für die Umsetzung des Prototypen muss die App entwickelt, der physiarchitektursche Prototyp gebaut und dessen Steuerung programmiert werden.

8.1 App

Die App wird aufgrund der Entscheidung bei 7.2.2 mit dem Framework Expo entwickelt. Zuerst werden die benötigten Pakete eingebunden. Anschließend wird die Navigation realisiert und das Design der einzelnen Screens und die Komponenten erstellt. Schlussendlich wird Firebase eingebunden und die einzelnen Funktionen entwickelt.

8.1.1 Verwendete Pakete

> „*Softwarepakete sind Sätze von Programmdateien, Bibliotheken und anderen Ressourcen, die Softwareentwicklern helfen sollen, Anwendungen zu erstellen. Sie bieten eine Grundlage von vorgefertigtem Code und Komponenten, die Entwickler verwenden können, um schnell Anwendungen zu erstellen, ohne den Code von Grund auf neu schreiben zu müssen.*“ (vgl. *Eine Einführung in Software-Pakete – Definitionen* 2023)

Beim Erstellen der App mithilfe von Expo sind folgende Pakete bereits vorinstalliert:

- expo
- expo-status-bar
- react
- react-native
- expo-updates

Jedoch werden für die Entwicklung der App viele weitere Pakete benötigt. Expo bietet zwar viele integrierte Funktionen und Bibliotheken, aber es gibt immer noch einige Szenarien, in denen zusätzliche Bibliotheken erforderlich sind, um bestimmte Funktionen zu implementieren. So ist zum Beispiel für die Verwendung von Firebase und der Navigation, die Einbindung von externen Paketen notwendig. In anderen Fällen werden Pakete verwendet, da es die Entwicklung vereinfacht. Ein Beispiel dafür ist das Paket Moment, welches zum Umwandeln von Zeitstempeln verwendet wird.
Folgende Pakete werden zusätzlich für die Entwicklung der App verwendet:

react-native-switch-selector
Version: 2.3.0
Quelle: https://www.npmjs.com/package/react-native-switch-selector
Dieses Paket wird für den Switch-Selector auf dem Screen Tower verwendet.

Abbildung 33: Switch-Selector in der App

Moment
Version: 2.29.4
Quelle: https://momentjs.com/
In Firebase werden die Zeitpunkte der Einlagerung als Unix-Timestamp gespeichert. Moment erleichtert die Umwandlung vom Unix-Timestamp zu einem Datum mit Uhrzeit. Es wird bei der Anzeige der Einlagerungs- und Auslagerungszeiten verwendet.

Geolib
Version: 3.3.3
Quelle: https://github.com/manuelbieh/geolib/blob/master/CHANGELOG.md
Bei diesem Paket wird die Methode GetPreciseDistance verwendet, um die Distanz zwischen der anwendenden Person und dem Fahrradturm zu berechnen.

```
let distance = getPreciseDistance(
{
latitude: userlocation.coords.latitude,
longitude: userlocation.coords.longitude,},
{
latitude: tower.location.latitude,
longitude: tower.location.longitude }
);
```

Code-Auflistung 1: Verwendung der Methode getPreciseDistance

Der Code zeigt die Anwendung der Methode GetPreciseDistance. Dabei werden die Koordinaten der anwendenden Person und des Turmes übergeben. Die Methode liefert die Distanz zwischen den zwei Punkten in Metern.

expo-location
Version: 15.0.1
Quelle: https://docs.expo.dev/versions/latest/sdk/location/
Expo-location ermöglicht es, auf die GPS-Koordinaten des verwendeten Gerätes zuzugreifen. Dies wird verwendet, um sicherzustellen, dass sich die anwendende Person in der Nähe des Fahrradturms befindet.

Firebase
Version: 9.15.0
Quelle: https://docs.expo.dev/guides/using-firebase/
Von dieser Bibliothek werden die Methoden auth und firestore verwendet.

Auth steht für Authentifizierung. Es wird bei der Entwicklung der App für folgende Zwecke verwendet:

- Erstellen von Accounts
- Anmelden mit Accounts
- Abmelden
- Zugriff auf ID der anwendenden Person

Firestore wird für das Schreiben und Lesen von Daten aus der Datenbank verwendet. Dies wird für folgende Zwecke benötigt:

- Ausgabe der gelagerten Gegenstände
- Anzeige der Fahrradtürme
- Einlagerungs- und Auslagerungsprozess

react-native-maps
Version: 1.3.2
Quelle: https://github.com/react-native-maps/react-native-maps
Mit react-native-maps kann man eine Karte auf dem jeweiligen Gerät erzeugen und Marker setzen. Es werden die Komponenten <Mapview> für die Karte und <Marker> für den Location-Pin verwendet.

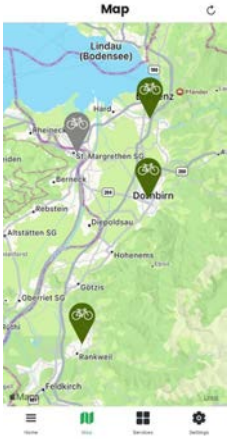


Abbildung 34: Karte im Tab „Map“

Wie auf der Abbildung 34 ersichtlich, wird bei der App die Bibliothek react-native-maps für die Karte mit den Möglichkeiten zum Lagern eines Fahrrades verwendet.

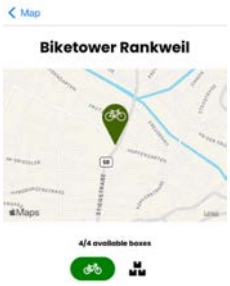


Abbildung 35: Karte im Tab „TowerInfo“

Die Abbildung 35 zeigt die zweite Karte, die im Screen „TowerInfo“ verwendet wird, wo der Standort des Turmes angezeigt wird.

react-navigation/native
Version: 6.1.4
Quelle: https://reactnavigation.org
React-navigation/navtive ist ein Paket, das Teil des React Navigation-Frameworks ist. Das Paket wird verwendet, um die Navigation zu initialisieren. Zusätzlich beinhaltet es die Methode getFocusedRouteNameFromRoute. Damit bekommt man den Namen des Screens, auf dem sich die anwendende Person befindet.

```
const routeName = getFocusedRouteNameFromRoute(route);
if (routeName === "assignment"){
    navigation.setOptions({tabBarStyle: {display: 'none'}});

}else {
    navigation.setOptions({tabBarStyle: {display: 'flex'}});
}
```

Code-Auflistung 2: Verwendung der Methode getFocusedRouteNameFromRoute

Der Code zeigt die Verwendung der Methode getFocusedRouteNameFromRoute. Falls sich die anwendende Person auf dem Screen Assignment befindet, wird die untere TabBar nicht angezeigt.

react-navigation/native-stack

Version: 6.9.7

Quelle: https://github.com/software-mansion/react-native-screens#readme

Dieses Paket ist eine Erweiterung von react-navigation/native und wird für den StackNavigator verwendet. Es ermöglicht eine einfache und unkomplizierte Lösung zum Navigieren.

```
<Stack.Navigator
  screenOptions={({ route }) => ({
    headerTitleStyle: {
      fontFamily: "Poppins",
      fontWeight: "400",
    fontSize: 21
  }})}>
  <Stack.Screen name="map" options={{ title: 'Map' }} component={Map} />
  <Stack.Screen name="tower" options={{ title: '' }} component={TowerInfo} />
  <Stack.Screen name="boxinfo" options={{ title: '' }} component={BoxInfo} />
  <Stack.Screen name="assignment" options={{headerShown: false, title: '' }}
  component={Assignment} />
</Stack.Navigator>
```

Code-Auflistung 3: Verwendung vom StackNavigator

Der abgebildete Code zeigt den StackNavigator. Er beinhaltet Optionen für den Header. Jeder einzelne Screen hat einen Namen, einen Titel und eine Komponente.

react-navigation/bottom-tabs

Version: 6.5.2

Quelle: https://github.com/react-navigation/react-navigation#readme

Mit diesem Paket lässt sich einfach ein TabNavigator erstellen. Das Paket bietet eine Vielzahl

von Konfigurationsoptionen für Tabs, damit zum Beispiel der Tab auf dem man sich befindet, hervorgehoben wird.

expo-splash-screen
Version: 0.17.5
Quelle: https://docs.expo.dev/versions/latest/sdk/splash-screen/
expo-splash-screen ist ein Paket, das Teil der Expo Software-Entwicklungskit (SDK) ist und zur Verwendung eines Splash-Screens in React Native-Anwendungen verwendet wird. Ein Splash-Screen ist ein Startbildschirm, der beim Starten der Anwendung angezeigt wird, um dem Benutzer visuelles Feedback zu geben, dass die App geladen wird. Das Paket wird beim Screen „Map“ verwendet, wo beim Laden der Türme aus firestore eine leere Karte angezeigt wird.

8.1.2 Navigation

TabNavigator Der StackNavigator ist die Navigationsleiste unten am Bildschirm. Damit kann die anwendende Person zwischen verschiedenen Ansichten wechseln. Die Ansicht, auf der man sich gerade befindet, wird mit einem Farbton hervorgehoben. Die Abbildung 36 zeigt die BottomTabBar der entwickelten App. (vgl. *Tab navigation | React Navigation* 2022)

Abbildung 36: BottomTab der App

StackNavigator Beim StackNavigator wird eine Seite auf die aktuelle Seite gelegt. Dadurch ist es möglich, mehrere Seiten aufeinander zu stapeln. Mit dem Klicken auf die Rücktaste (im Beispiel auf der Abbildung 37 ist es der Schriftzug „Map“) gelangt man auf den vorherigen Screen zurück. (vgl. *Stack Navigator | React Navigation* 2022)

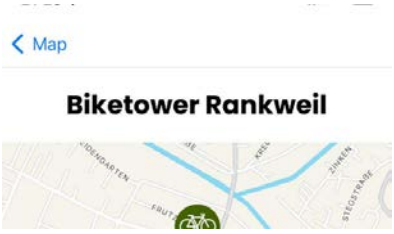


Abbildung 37: Header der App mit Rücktaste

Die Navigation der App besteht aus einem BottomStackNavigator. Dieser führt zu vier verschiedenen StackNavigator:

- HomeStack
- MapStack
- ServiceStack
- SettingsStack

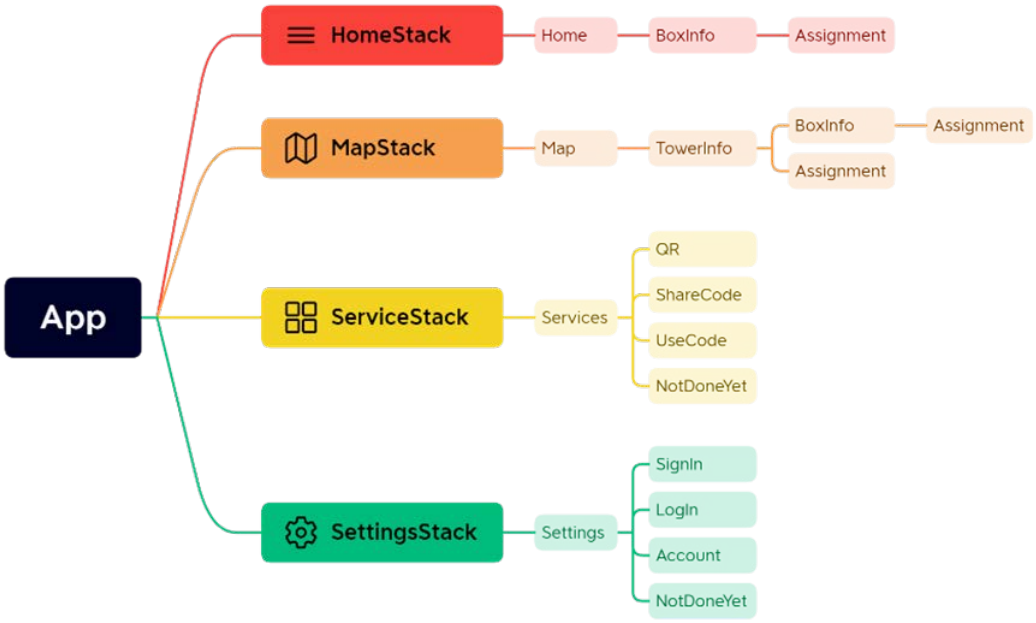


Abbildung 38: Navigations-Struktur der App

Die Abbildung 38 zeigt die Navigations-Struktur der App. Grundsätzlich kann man über die BottomTabBar von einem StackNavigator zu einem anderen navigieren. Bei jedem Stack wird der erste Screen angezeigt. Von dort aus kann man auf eine oder mehrere Unterseiten gelangen.

8.1.3 Komponenten

Komponenten erlauben es, in der Benutzeroberfläche wiederverwendbare Teile zu verwenden. Komponenten akzeptieren Eingaben, die „props“ genannt werden, und geben React-Elemente zurück. (vgl. *Komponenten und Props – React* 2023)

In der App werden folgende Komponenten verwendet:

Box Hiermit können die aktuell gelagerten Gegenstände oder vergangene Buchungen angezeigt werden. Angezeigt wird der Name des Einlagerungsortes, ein Icon (entweder ein Fahrrad oder Boxen) und das Datum der Einlagerung. Falls die Buchung zu Ende ist, wird außerdem das Enddatum angezeigt.

Abbildung 39: Komponente Box mit Fahrrad

Abbildung 40: Komponente Box mit Gegenstand

Die Komponente wird auf den Seiten „Aktivität“ und „Fahrradturm“ verwendet.

```
<Box navigation={navigation} boxinfo={item} />
```

Code-Auflistung 4: Ausdruck für die Komponente Box

Verwendete Props:

- Element für Navigation (navigation)
- Array der Box (boxinfo)

SettingLink Mit der Komponente SettingLink erstellt man einen Link, mit dem man zu einer Unterseite mit Einstellungen kommt. Die Komponente wird auf der Seite „Einstellungen“ verwendet.

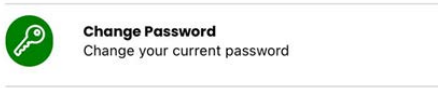


Abbildung 41: Komponente SettingLink

```
<SettingLink
    navigation={navigation}
    navigate={"notdone"}
    picturename="key-outline"
    heading="Change Password"
    info="Change your current password"
  />
```

Code-Auflistung 5: Ausdruck für die Komponente SettingLink

Verwendete Props:

- Element für Navigation (navigation)
- Screen auf den navigiert wird (navigate)
- Name des Bildes (picturename)
- Text der Überschrift (heading)
- Text der Info (info)

OtherLink Die Komponente OtherLink funktioniert ähnlich wie die Komponente SettingLink. Sie wird auf dem Screen Services verwendet. Die Komponente dient als Link zu Unterseiten.

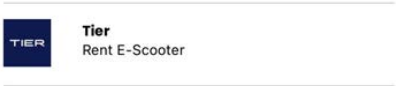


Abbildung 42: Komponente OtherLink

```
<OtherLink
    navigation={navigation}
    navigate={"qr"}
    picturename={require("./../assets/images/tierlogo.jpg")}
    heading="Tier"
    info="Rent E-Scooter"
  />
```

Code-Auflistung 6: Ausdruck für die Komponente OtherLink

Verwendete Props:

- Element für Navigation (navigation)
- Screen auf den navigiert wird (navigate)
- Link zum Bild (picturename)
- Text der Überschrift (heading)
- Text der Info (info)

BoxImage Die Komponente BoxImage wird auf dem Screen „Assignment“ und „BoxInfo“ verwendet. Die Komponente rendert ein Bild der Box entweder mit einem Fahrrad oder mit Boxen, je nachdem ob die Variable type „bike“ oder „item“ lautet.

Abbildung 43: Komponente BoxImage als Fahrradbox

Abbildung 44: Komponente BoxImage als Box für Gegenstände

```
<BoxImage type={box.boxref.type}/>
```

Code-Auflistung 7: Ausdruck für die Komponente BoxImage

Verwendete Props:

- Art der Box (type)

8.1.4 Screens

Zur besseren Übersichtlichkeit sind die Texte in der App im Gegensatz zu den Mockups in Englisch.

Home Die Seite „Home“ ist der erste Screen auf den die anwendende Person kommt. Die Funktion UseEffect, die beim Rendern des Screens ausgeführt wird, führt die Funktion UserStatus aus.
Die Funktion UserStatus überprüft, ob die anwendende Person angemeldet ist. Falls die anwendende Person nicht angemeldet ist, wird eine Aufforderung zur Anmeldung angezeigt (siehe Abbildung x). Falls die anwendende Person angemeldet ist, wird die Funktion getActivity() ausgeführt.
Die Funktion getActivity() wird verwendet, um alle Buchungen der anwendenden Person abzufragen. Falls die Buchungen in der Vergangenheit liegen, werden sie im Array FormerList gespeichert, wenn das Fahrrad noch gelagert ist, wird die Buchung im Array ActivityList gespeichert.
Die Buchungen werden auf zwei Listen anzeigt. Falls keine Buchung gefunden wurde, kommt der Text „no data“.
Mit einem Klick auf den „Aktualisieren Button“ in der rechten oberen Ecke, werden die Buchungen der anwendenden Person neu aus der Datenbank abgerufen.

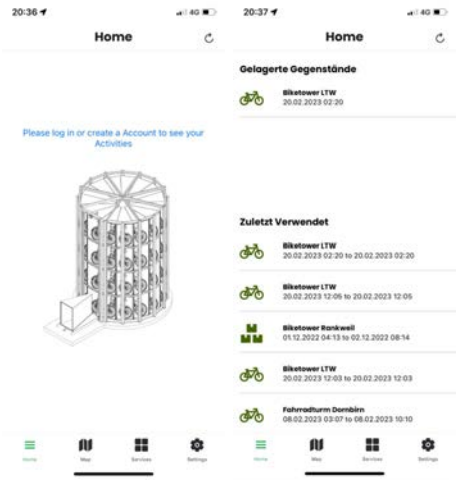


Abbildung 45: Screen Home

Die Abbildung 45 zeigt links die Ansicht ohne Anmeldung und rechts die Ansicht mit Anmeldung.

BoxInfo Auf dem Screen „BoxInfo“ werden folgende Daten angezeigt:

- Name des Fahrradparkhauses, in dem sich die Box befindet

- Beginn der Buchung und falls die Buchung in der Vergangenheit liegt, zusätzlich die Zeit der Abholung
- Komponente BoxImage für Bild der Fahrradbox

Diese Infos werden über die Navigation übergeben.

```
navigation.navigate("boxinfo", { box: this.props.boxinfo })
```

Code-Auflistung 8: Methode zum Navigieren

Bei der Code-Auflistung 8 wird gezeigt, wie man von der Komponente Box zum aktuellen Screen navigiert wird. Dort wird das Array boxinfo übergeben.

```
export default function BoxInfo({ route, navigation }) {
const { box } = route.params;
...}
```

Code-Auflistung 9: Zugriff auf Variable Box

Wie bei der Code-Auflistung 9, wird beim aktuellen Screen nun die Variable Box mithilfe von route.params abgerufen.
Falls man das Fahrrad noch nicht abgeholt hat, wird außerdem ein Button zum Auslagern angezeigt. Durch das Drücken des Buttons kommt man zum Screen „Assignment“.

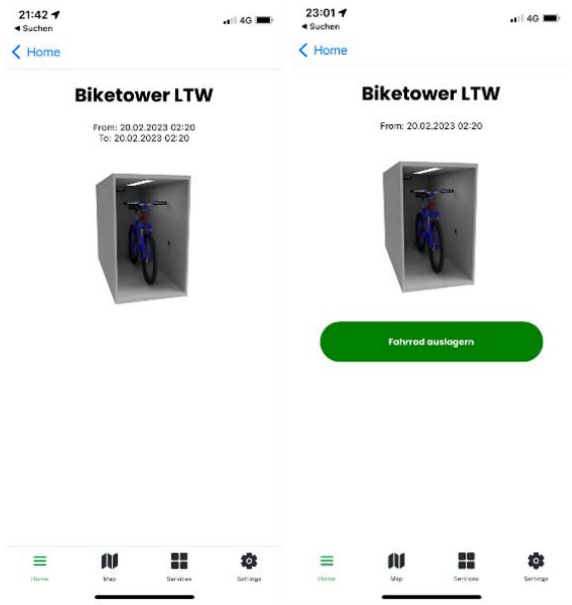


Abbildung 46: Screen BoxInfo

Die Abbildung 46 zeigt links die Ansicht bei einer bereits vergangenen Buchung und rechts die Ansicht bei einer aktiven Buchung.

Services Beim Screen „Services“ werden verschiedene Dienstleistungen angezeigt. Dafür wird die Komponente OtherLink verwendet, mit der man zu einer Unterseite gelangt.

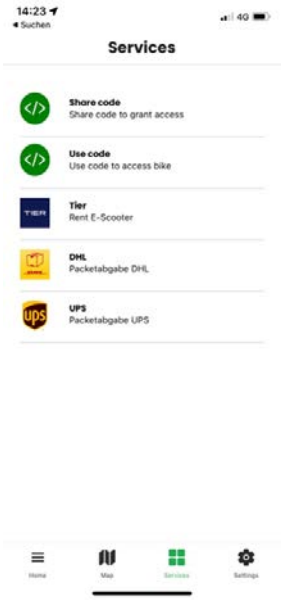


Abbildung 47: Screen Services

Settings Der Screen „Settings“ wird zur Verlinkung auf diverse Unterseiten verwendet. Dafür wird die Komponente SettingLink mehrmals verwendet.
Mit der Funktion UserStatus wird überprüft, ob die anwendende Person angemeldet ist und dementsprechend andere Einstellungen angezeigt.

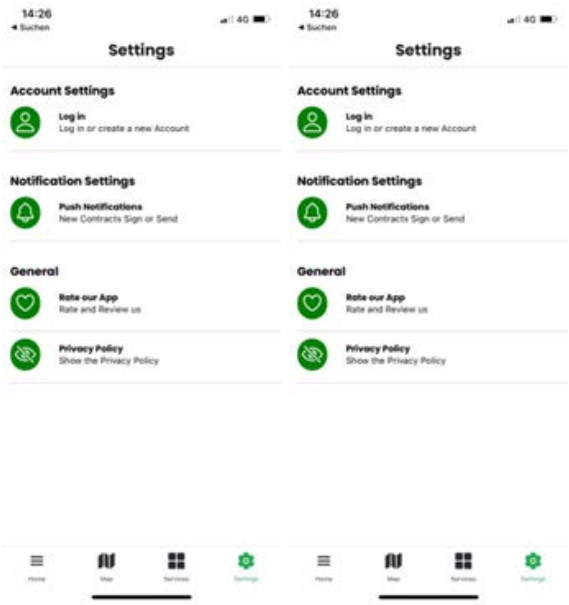


Abbildung 48: Screen Settings

Die Abbildung links zeigt die Ansicht, wenn die anwendende Person nicht angemeldet ist. Bei der Abbildung rechts sieht man, dass mehrere Einstellungen zum Account angezeigt werden, falls die anwendende Person eingeloggt ist.

LogIn Der Screen „LogIn“ dient zum Anmelden mit einem bestehenden Account. Im Screen „LogIn“ gibt es zwei Textfelder zum Eingeben der E-Mail und des Passworts. Mit einem Klick auf den Button „Sign in“ wird die Funktion SignIn ausgeführt.

```
  function SignIn() {
  auth
    .signInWithEmailAndPassword(email, password)
    .then((userCredential) => {
      setError("")
      const user = userCredential.user;
      Alert.alert("signed in");
      navigation.navigate("settings", {created: false});
    })
    .catch((error) => {
      const errorCode = error.code;
      const errorMessage = error.message;
      setError(errorMessage)
    });
};
```

Code-Auflistung 10: Funktion SignIn

Dabei wird die Methode signInWithEmailAndPasswort des Authentizierungs-Service auth von Firebase verwendet. Es werden die eingegebene E-Mail und Passwort übergeben. Bei einer erfolgreichen Anmeldung bekommt die anwendende Person die Meldung „signed in“ und wird zum Screen „Settings“ zurückgeleitet. Beim Klicken auf den Button „Register“ wird man zur Seite „SignIn“ geführt.

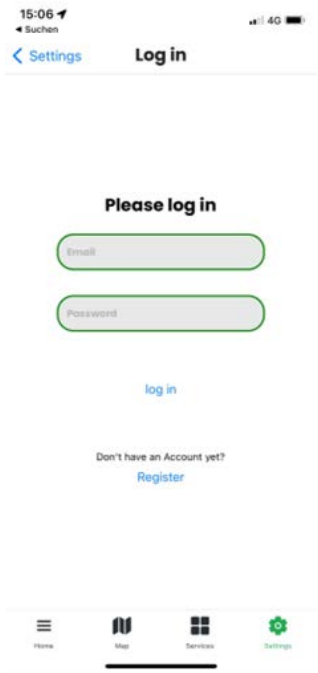


Abbildung 49: Screen LogIn

SignIn Der Screen „SignIn“ dient zum Erstellen eines Accounts. In den Textfeldern kann man den eigenen Namen, die E-Mail und ein Passwort eingeben. Wenn man auf den button „create Account“ klickt, wird die Funktion createAccount ausgeführt.

```
function createAccount() {
 auth
   .createUserWithEmailAndPassword(email, password)
   .then(() => {
     setError("")
     addUser(auth.currentUser.uid,email,name)
     Alert.alert("User signed in and logged in");
     navigation.navigate("settings", { created: true, name: name});
   })
   .catch((error) => {
     if (error.code === "auth/email-already-in-use") {
      setError("That email address is already in use!");
     }

     else if (error.code === "auth/invalid-email") {
       setError("That email address is invalid!");
     }
     else {
     setError(error.message);
     }
   });
};
```

Code-Auflistung 11: Funktion createAccount

Die Methode createUserWithEmailAndPasswort, die zum Erstellen eines Users verwendet wird, ist vergleichbar mit signInWithEmailAndPasswort. Sie erstellt beim Authentifizierungs-Service auth einen neuen User und meldet diesen an. Falls die Erstellung des Accounts erfolgreich ist, werden folgende Schritte ausgeführt:

- Die Funktion addUser wird ausgeführt. Damit wird für den User in Firestore ein Dokument mit dem Namen und der userid erstellt.
- Ausgabe der Meldung „User signed in and logged in“.
- Die anwendende Person wird zurück zu Settings navigiert.

Falls die Erstellung des Accounts nicht möglich ist, wird eine entsprechende Fehlermeldung ausgegeben.

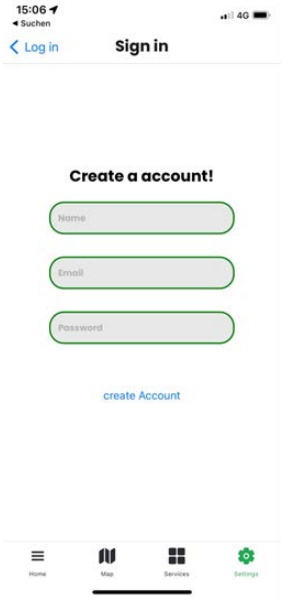


Abbildung 50: Screen SignIn

Map Auf dem Screen „Map“ werden die Standorte von Fahrradparkhäusern angezeigt. Mithilfe vom Expo SplashScreen wird zuerst eine leere Karte angezeigt, bis die Funktion getTowers fertig ist. Die Funktion getTowers liefert die Daten zu allen Fahrradtürmen.
Anschließend werden alle Marker auf der Karte angezeigt. Die Funktion MarkerImage liefert entweder einen grünen Marker, falls noch Plätze frei sind, oder einen grauen Marker, wenn es keine freien Plätze für Fahrräder gibt. Beim Klicken auf einen Marker, kommt man zum Screen „TowerInfo“.

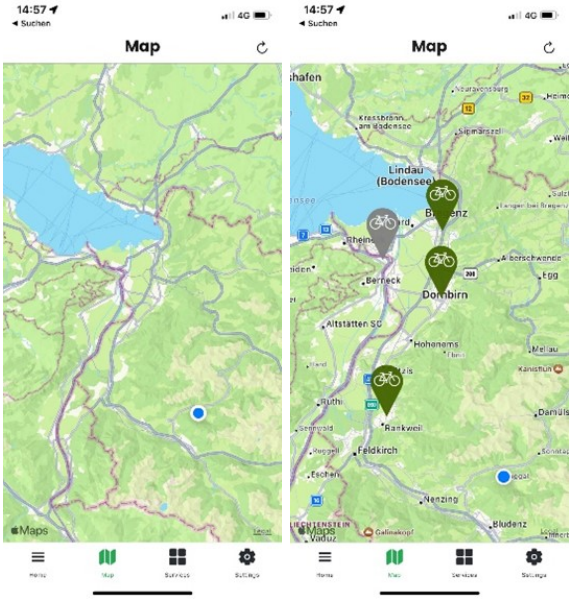


Abbildung 51: Screen Map

Die Abbildung links zeigt die Karte, wenn die Türme noch nicht aus der Datenbank geladen sind. Auf der Abbildung rechts sieht man eine Karte mit vier Beispiel-Türmen. Der Turm links oben ist grau, da dort kein Platz frei ist.

TowerInfo Beim Screen „Towerinfo“ werden Daten zum Fahrradturm angezeigt. Sie werden über das Objekt Route übergeben.

Es werden folgende Daten angezeigt:

- Name des Turmes
- Karte mit Standort vom Turm
- Anzahl verfügbarer Boxen und aller Boxen

Falls man nicht angemeldet ist:

- Button mit Aufforderung zum Anmelden

Falls man angemeldet ist:

- Möglichkeit zum Lagern von Fahrrad oder Gegenstand
- Gegenstände, die im Turm gelagert sind (Komponente Box)

Mit einem Klick auf das Aktualisieren-Symbol auf der rechten oberen Ecke, werden die aktuellen Daten des Turmes von der Datenbank abgerufen.

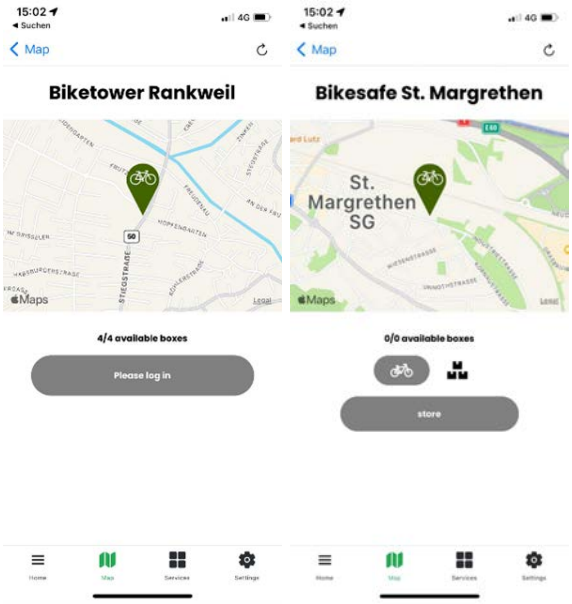


Abbildung 52: Screen TowerInfo

Abbildung 52 links: Falls die anwendende Person noch nicht angemeldet ist, erscheint ein grauer Button mit Aufforderung zum Anmelden. Beim Klicken auf den Button wird man zu den Einstellungen navigiert.
Abbildung 52 rechts: Falls die anwendende Person angemeldet ist, aber keine freie Fahrradbox verfügbar ist, wird der Button mit einer grauen Farbe angezeigt. Beim Klicken auf den Button erscheint die Meldung „no available space“.

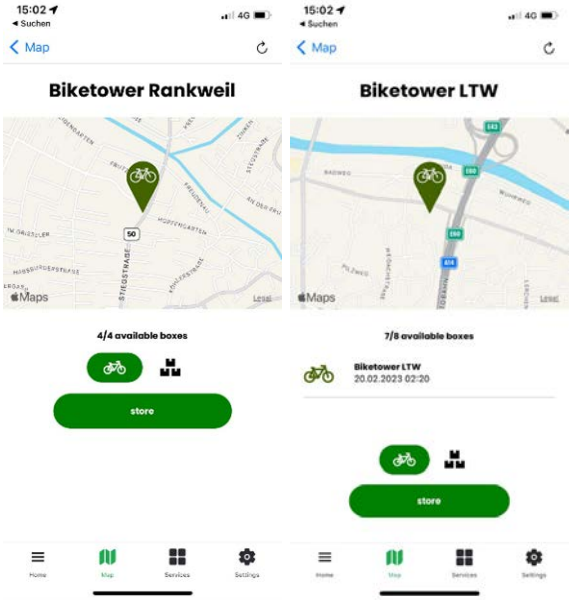


Abbildung 53: Screen TowerInfo

Abbildung 53 links: Falls die anwendende Person angemeldet ist und es einen freien Platz gibt, erscheint der Button in grüner Farbe. Mit dem SwitchSelector kann man die Art der Box auswählen. Beim Klicken auf den Button wird man zum Screen „Assignment“ navigiert.
Abbildung 53 rechts: Falls die anwendende Person angemeldet ist und eine Box beim Turm gelagert hat, erscheint diese als Komponente Box in der Mitte des Bildschirms. Mit einem Klick auf die Box kommt man zum Screen „BoxInfo“.

Assignment Der Screen „Assignment“ wird zum Aus- und Einlagern von Gegenständen verwendet.
Mit der Funktion UseEffect, die beim Rendern des Screens ausführt, wird die Funktion storeBike durchgeführt. Die Funktion überprüft zuerst mit dem Standort der anwendenden Person, ob er sich tatsächlich in der Nähe des Fahrradturms befindet. Das Projektteam hat sich für eine maximale Entfernung von 250m entschieden, damit es bei Testzwecken zu keinen Problemen kommt.
Falls sich die anwendende Person innerhalb des definierten Umkreises befindet, wird die Anfrage in Firestore erstellt. Diese befindet sich im Pfad tower/towerid/jobs. Dabei wird ein Listener erstellt, der bei einer Änderung der Anfrage durch den Tower-Controller, die Anfrage erneut aus der Datenbank abruft.
Anschließend wird auf die Antwort vom Tower-Controller gewartet. Dabei gibt es verschiedene Szenarien:

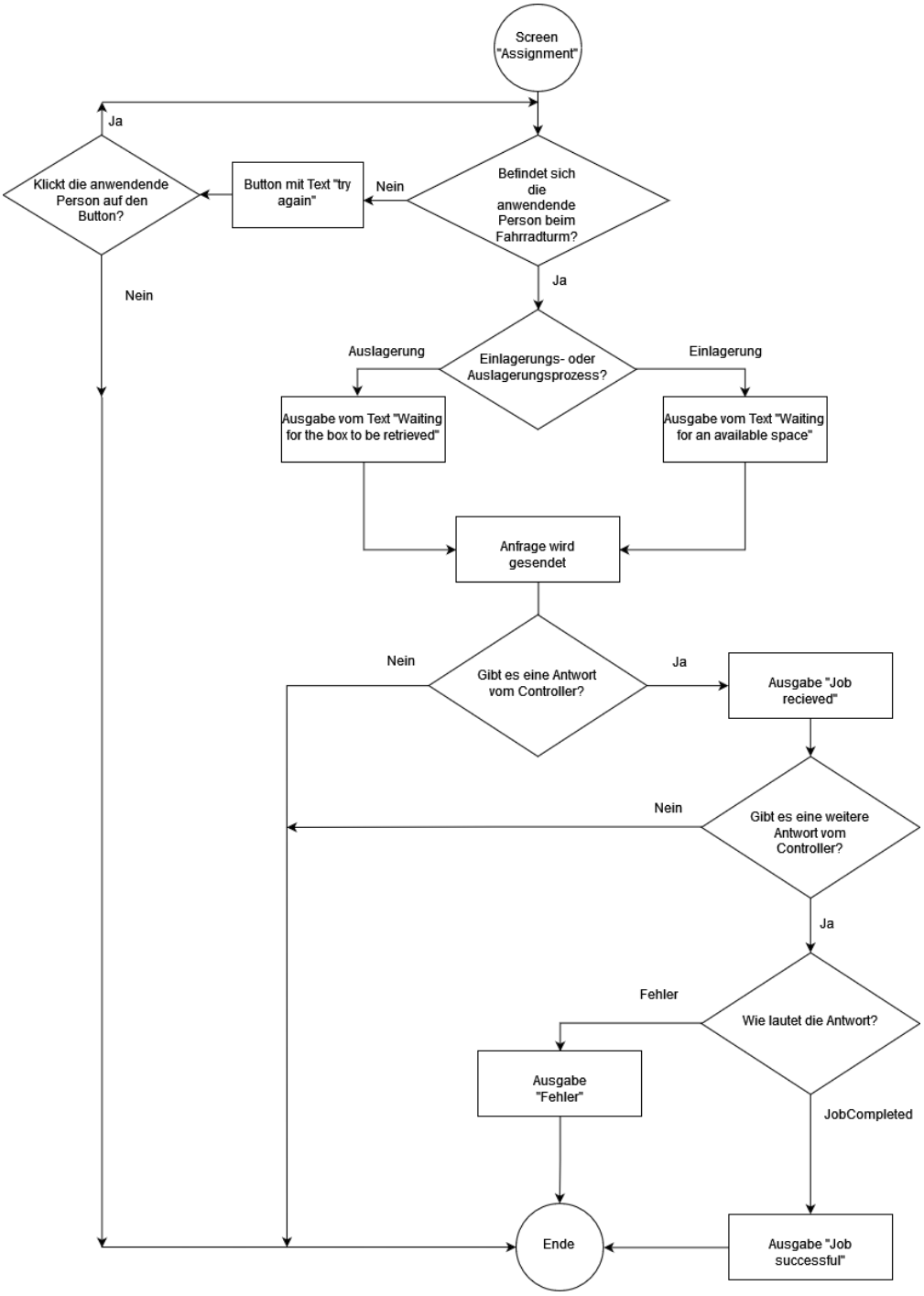


Abbildung 54: Szenarien beim Screen Assignment

Wie bei der Abbildung 54 ersichtlich, gibt es nach dem Überprüfen des Standorts mehrere Szenarien. Je nach Art der Anfrage und Antwort des Controllers wird dem User etwas anderes angezeigt.

Abbildung 55: Senden der Anfrage zum Aus-/Einlagern

Abbildung 55: Beim Senden der Anfrage kommt die Nachricht „Waiting for an available space“ beim Einlagern und beim Auslagern die Nachricht „Waiting for bike to store out“.

Abbildung 56: JobRecieved

Abbildung 56 links: Wenn der Tower-Controller die Anfrage bekommen hat, wird die Meldung „Job recieved“ angezeigt. Abbildung 56 rechts: Falls der Tower-Controller eine Fehlermeldung sendet, wird diese mit roter Schrift angezeigt.

Abbildung 57: JobSuccessful

Abbildung 57: Falls der Job erfolgreich abgeschlossen wurde, wird entweder die Box mit dem Fahrrad oder den Paketboxen angezeigt.

8.2 Turm

Um den geplanten Miniaturturm in die Tat umzusetzen, wurde eine Vielzahl an Komponenten benötigt. Die einzelnen Komponenten werden in Abildung 58 dargestellt.

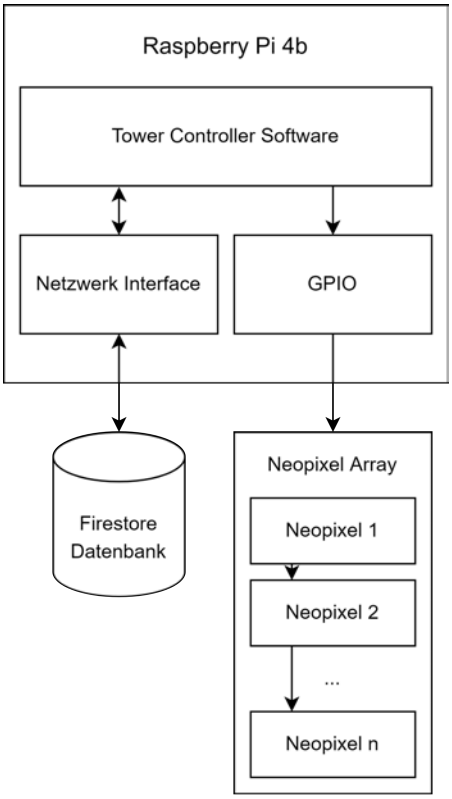


Abbildung 58: Tower Controller Komponenten

8.2.1 Software

Für die Stuerung des Turmes wird ein eigenes Softwaremodul entwickelt. Dieses kontrollieren Hardware und Datenbank und reagiert auf Anfragen. Ursprünglich war geplant Python als Programmiersprache zu verwenden, da es einfach und simpel ist und man schnell Prototypen entwickeln kann. Nach den ersten Versuchen wechselt das Projektteam zur Programmiersprache Rust.

V1 - Python Die Pythonversion ist basierend auf einem Queue System, bei dem verschiedene Komponenten in verschiedenen Threads selbstständig Jobs und Events hinzufügen. Ein oder mehrere Job-Handler entnehmen sich anschließend Jobs aus der Queue und führen ausführen. Ein Job-Handler stellt jeweils eine Einlagerungsstation dar. Damit werdern mehrere verschiedene Auslagerungsstationen abgebildet.

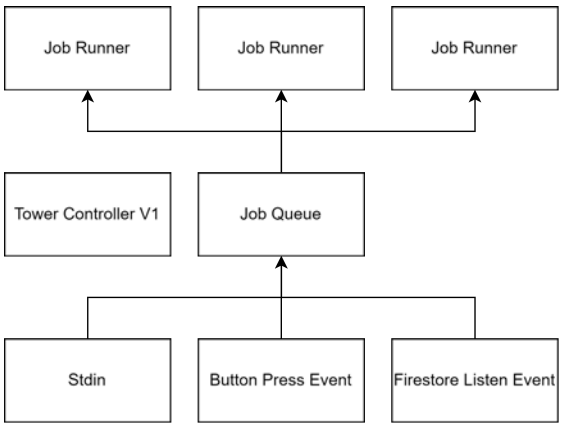


Abbildung 59: Datenströme des Tower Controller V1

Standardeingabe (Stdin) und Standardausgabe (Stdout) werden verwendet, um mit dem Benutzer zu kommunizieren. Stdin wird verwendet, damit ein Benutzer manuell Jobs zur Queue hinzufügen kann. Dies ist vorallem beim Debuggen nützlich. Stdout wird verwendet um dem Benutzer Informationen über den aktuellen Status des Turms zu geben. Eingaben sind eine blockierende Operation, deshalb wurde `aioconsole` verwendet um Stdin asynchron abzurufen.

In einem weiteren Thread kommuniziert ein General Purpose Input Output (GPIO) Listener mit einem physikalischen Button, welcher zur Bestätigung des Einlagerungsvorgangs verwendet wird. Der Listener wartet auf einen Tastendruck und fügt anschließend einen Job (Event) zur Queue hinzu.

Der wichtigste Listener ist der Datenbanklistener, dieser wartet auf Änderungen in der Datenbank und fügt entsprechende Jobs zur Queue hinzu. Dieser Listener ist essenziell für die Kommunikation zwischen der App und dem Turm. Der Listener wird von der offiziellen Python Bibliothek für Firebase bereitgestellt.

Zusätzlich wird eine grafische Benutzeroberfläche (GUI) entwickelt, welche die aktuellen Zustände der einzelnen Radboxen darstellt. Diese GUI wird ebenfalls von einem Thread ausgeführt und aktualisiert sich automatisch, wenn sich der Zustand der Radboxen ändert. Die GUI ist in `tkinter` implementiert und soll bei der Entwicklung helfen, falls das Programm nicht auf dem Raspberri Pi ausgeführt wird oder die Elektronik noch nicht fertig ist.

Die Python Version war zu Anfang vielversprechend, jedoch gibt es einige Probleme mit der Architektur, die zu einem komplizierten und unübersichtlichen Code führten. Die Architektur wird in der V2 komplett überarbeitet.

V2 - Python Die Python Version 2 wird komplett überarbeitet und viel einfacher und übersichtlicher gestaltet. Das Konzept der mehreren Auslagerungsplätze wird und nur ein Auslagerungsplatz ist implementiert.
V2 ist basierend auf einer State machine. Die verschiedenen States sind in Abbildung 60 zu sehen.

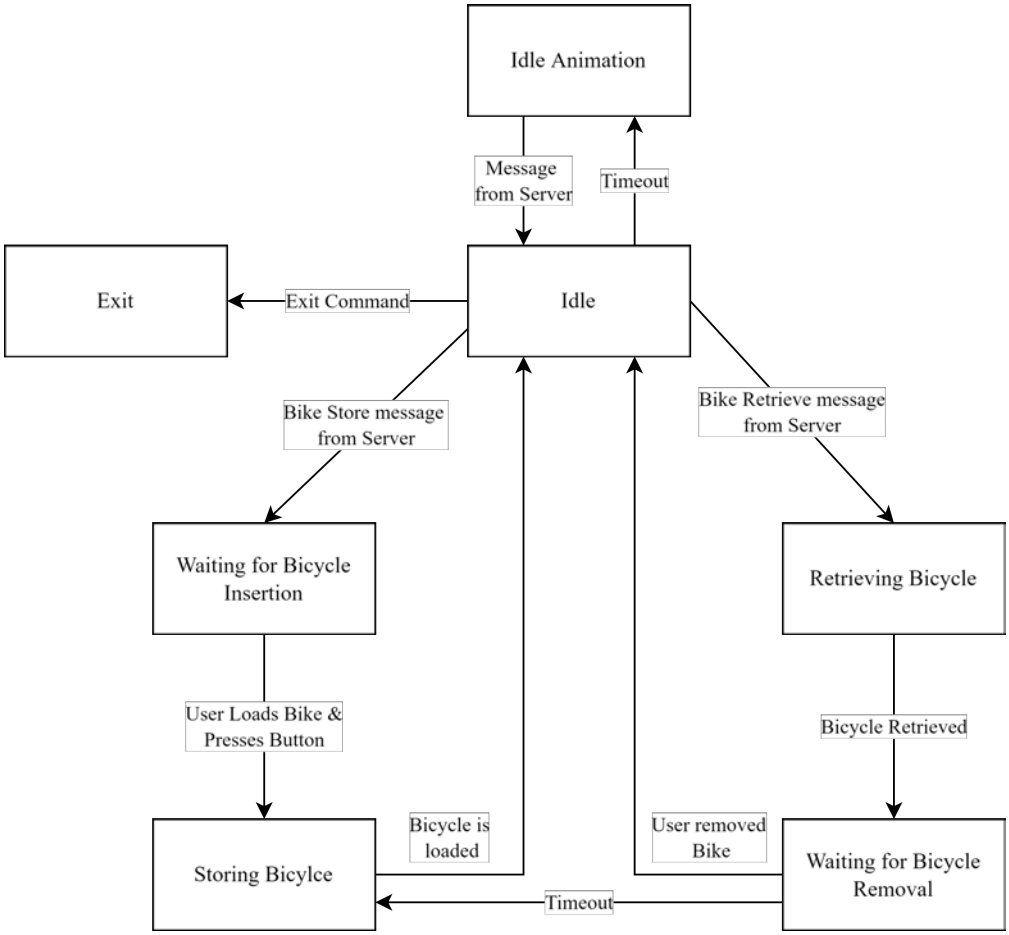


Abbildung 60: State Machine des Tower Controller V2

States:

- **Idle** - Der Turm ist im Idle State, wenn er nicht gerade eine Aktion ausführt. In diesem Zustand wird auf Events gewartet. Wenn ein Event eintritt, wird der Zustand entsprechend angepasst.
- **Waiting for Bicycle Insertion** - Ein Nutzer hat den Einlagerungsprozess eines Fahrrads gestartet, der Turm wartet nun auf das Einlegen des Fahrrads in die Radbox und die Einlagerungsbestätigung (Drücken des Buttons) des Nutzers.
- **Storing Bicycle** - Das Fahrrad wurde in die Radbox eingelagert und der Nutzer hat die Einlagerungsbestätigung erteilt. Der Turm wartet nun auf das Schließen der Radbox und das Abschließen des Einlagerungsprozesses. Anschließend wird der Zustand wieder auf Idle gesetzt.
- **Retrieving Bicycle** - Ein Nutzer hat den Auslagerungsprozess eines Fahrrads gestartet, der Turm wartet nun auf das Abschließen des Auslagerungsprozesses und das Öffnen der Radbox.
- **Waiting for Bicycle Removal** - Der Turm wartet auf das Entfernen des Fahrrads aus der Radbox und die Auslagerungsbestätigung des Nutzers. Anschließend wird der Zustand wieder auf Idle gesetzt. Falls der Nutzer die Auslagerungsbestätigung innerhalb eines gewissen Zeitraums nicht erteilt, wird der Zustand auf Storing Bicycle gesetzt und das Fahrrad wird eingelagert.
- **Idle Animation** - Falls der Turm für eine gewisse Zeit im Idle Zustand ist, wird eine Animation ausgeführt. Diese Animation soll den Turm visuell ansprechender machen.
- **Exit** - Der Turm Controller wird beendet.

Änhlich wie V1 besitzt V2 auch eine GUI, die den aktuellen Zustand des Turms darstellt. Diese wurde einfach von V1 übernommen und angepasst.

Die V2 Variante funktioniert gut und hätte womöglich auch architekturell funktioniert, jedoch gibt es Probleme mit Python selbst. Die Probleme stammen von zyklischen Importen und erlauben es nicht die Entwicklung weiterzuführen. Womöglich kann das Problem gelöst werden indem auf Type Hints verzichtet wird, jedoch beeinträchtigt dies die Developer Experience und Lesbarkeit des Codes stark. Aus diesem Grund wurde entschieden die Entwicklung von V2 einzustellen und auf eine andere Programmiersprache zu wechseln.

V3 - Rust Die V3 Variante ist in Rust (10.4) implementiert. Rust ist eine moderne Programmiersprache, die sich durch eine gute Performance und eine gute Developer Experience auszeichnet. Rust unterscheidet sich stark von Python, da es eine Systemsprache ist.

Gegenüberstellung Rust und Python:

Rust vs. Python	**Rust**	**Python**
Typisierung	Statisch	Dynamisch
Garbage Collection	Manuell	Automatisch
Performance (100mio Stellen Pi Benchmark (vgl. *Speed Comparison* 2023))	69.31s	5851.53s (84x langsamer)
Kompiliert/Interpretiert	Kompiliert	Interpretiert
Multi-Threaded	Ja	Nein

Tabelle 14: Gegenüberstellung Rust und Python

Rust wurde für V3 gewählt, da das zuständige Projektmitglied mehr Erfahrung mit Rust hat als mit Python. Obwohl Rust komplizierter ist als Python, ist die Entwicklungsgeschwindigkeit durch die bessere Developer Experience deutlich höher.
V3 besitzt auch eine GUI, die den aktuellen Zustand des Turms darstellt. Diese ist in `fltk_rs` implementiert.
Architekturtechnisch ist V3 sehr ähnlich zu V2. Dennoch ist die Architektur immer noch zu kompliziert und somit die Entwicklungsgeschwindigkeit stark beeinträchtigt.

V4 - Rust Die V4 Variante wird ebenfalls in Rust entwickelt, aber von Grund auf neu aufgebaut. Die Architektur wird komplett überarbeitet und stark vereinfacht. Die State machine wird verworfen und die ganze Applikation basiert auf dem Datenbank Listener. Der Listener wird mithilfe Funktionalität der `firestore_rs` (10.4.1) Softwarebilbiothek implementiert.

Projektstruktur Das Projekt wird mit dem command `cargo new` erstellt, deshalb ist die Projektstruktur wie folgt:

```
tower_controller_v4
└─ src
   ├─ bin
   └─ entities
```

- `tower_controller_v4` Root Verzeichnis des Projekts, hier befinden sich alle Konfigurationsdateien (Cargo.toml, Cargo.lock, .gitignore, etc. . .).
- `src` Hier befinden sich alle Quellcode Dateien.
- `bin` Hier befinden sich alle Quellcode Dateien, die ausgeführt werden können.
- `entities` Hier befinden sich die serialisierbaren Datenstrukturen (Entities), die den Dokumenten in der Datenbank entsprechen.

Programmablauf

1. Der Turm Controller wird mittels der `main` Funktion gestartet.
2. Umgebungsvariablen werden geladen.
3. Datenbank Interface wird erstellt (`TowerDatabase::new`).
4. Turm Datenstruktur wird erstellt und mit den Daten aus der Datenbank initialisiert (`Tower::new`).
5. Assignment Scheduler wird erstellt (`AssignmentScheduler::new`).
6. Der Scheduler wird gestartet (`AssignmentScheduler::start`) und der main Thread wird in den Sleep Modus versetzt.
7. Das Programm wartet nun auf Änderungen in der Datenbank.

Abbildung 61: Intialisierung des Turm Controllers

Event Handling Pipeline Wenn der Turm Controller ein Event empfängt, wird dieses in der Event Handling Pipeline verarbeitet. Die Event Handling Pipeline besteht aus Validierungs- und Ausführungsstufen. Die Validierungsstufen prüfen, ob das Event gültig ist und ob es überhaupt verarbeitet werden muss. Die Ausführungsstufen führen die eigentliche Verarbeitung des Events aus. Die Funktionsweise der Event Handling Pipeline ist in Abbildung 62 dargestellt.

Abbildung 62: Event Handling Pipeline

In Abbildung 62 werden nicht alle möglichen Fehler dargestellt sondern nur welche, für die der Benutzer der Programmierschnittstelle (API) verantwortlich ist. Fehler, welche aufgrund eines internen Fehlers auftreten, werden nicht dargestellt (Fehler bei Datenbankverbindung, Programmlogikfehler, etc...).

TowerDatabase `TowerDatabase` kümmert sich um die Kommunikation mit der Datenbank. Es ist ein Wrapper für die Datenbankverbindung der `firestore_rs` Softwarebilbiothek. Es vereinfacht die Interaktion mit der Datenbank und minimiert die möglichekeit von Fehlern.

```
struct TowerDatabase {
    db: FirestoreDb,
    tower_id: String,
    project_id: String,
}
```

Code-Auflistung 12: TowerDatabase Struktur

Felder:

- `db`: Datenbankverbindung.
- `tower_id`: ID des Turms.
- `project_id`: ID des Projekts.

Methoden:

- `new`: Erstellt eine neue Instanz von `TowerDatabase`.
- `has_subscription`: Prüft, ob ein nutzer eine Abonnement hat.
- `set_error`: Setzt einen Fehlerstatus eines Assignments.
- `set_confirm`: Setzt einen Bestätigungsstatus eines Assignments.
- `set_slot`: Setzt einen Slot eines Assignments.
- `create_listener`: Erstellt einen Listener, welcher auf Änderungen in der `jobs` Collection wartet.
- `get_tower`: Lädt die Daten des Turms aus der Datenbank (Wichtig für die Initialisierung des Turms).
- `create_boxes`: Erstellt fehlende Boxen in der Datenbank anhand des Layout Feldes im Turm Dokument.
- `new_rental`: Erstellt einen neuen Mietvorgang in der Datenbank (users > rentals).
- `finish_rental`: Beendet einen Mietvorgang in der Datenbank.

Tower `Tower` ist die Datenstruktur des Turms. Sie enthält alle Informationen über den Turm und die Boxen. Sie wird vom `AssignmentScheduler` verwendet, um die Boxen zu verwalten. Tower enthält zusätzlich Methoden um die Datenstruktur zu verwalten.

```
struct Tower {
    id: String,
    slots: HashMap<Vec<u32>, Slot>,
    layout: Vec<u32>,
    db: Arc<TowerDatabase>,
}
```

Code-Auflistung 13: Tower Struktur

Felder:

- `id`: ID des Turms.
- `slots`: Alle Slots des Turms. Der Schlüssel ist ein Vektor (Dynamischer Array) an `u32` Werten, welche die Position der Boxen im Turm beschreiben.
- `layout`: Layout des Turms.
- `db`: Datenbankverbindung (`Arc<T>` als Wrapper um Referenzen in mehreren Threads zu erlauben).

Methoden:

- `new`: Erstellt eine neue Instanz von `Tower`.
- `find_free_slot`: Sucht einen freien Slot für ein Assignment.
- `store_object`: Speichert ein Objekt (Fahrrad) in einer Box.
- `retrieve_object`: Entfernt ein Objekt (Fahrrad) aus einer Box.
- `slot_exitst`: Prüft ob ein Slot existiert.
- `slot_rented_by_user`: Prüft ob ein Slot von einem bestimmten Nutzer gemietet ist.
- `rent_box`: Mietet eine Box.
- `unrent_box`: Beendet die Miete einer Box.

AssignmentScheduler Der AssignmentScheduler hört auf Datenbankevents und verarbeitet diese.

```rust
pub struct AssignmentScheduler {
    db: Arc<TowerDatabase>,
    tower: Arc<Mutex<Tower>>,
    listener: FirestoreListener<FirestoreDb, HashMapTokenStorage>,
}
```

Code-Auflistung 14: AssignmentScheduler Struktur

Felder:

- `db`: Datenbankverbindung.
- `tower`: Turm (Der `Mutex<T>` erlaubt das paralelle Ändern von Daten ohne race conditions zu erzeugen).
- `listener`: Listener für die `jobs` Collection.

Methoden:

- `new`: Erstellt eine neue Instanz des `AssignmentScheduler`.
- `listen`: Startet den Listener.
- `stop`: Stoppt den Listener.

TowerDisplay Der TowerDisplay ist für die Anzeige mittels der LEDs zuständig. Er wird von der `Tower` Klasse verwendet, um den momentanen Zustand des Turms anzuzeigen.

Als erstes wird der Zustand des Turms zu einem langen hexadezimal String umgewandelt. Dieser String wird dann an ein kleines Python Script übergeben, welches die LEDs ansteuert. Das Python Script wird verwendet da es einfacher ist, die LEDs mit Python anzusteuern als mit Rust.

8.2.2 Hardware

Ursprünglich war geplant eine Varaiante des Turms mit 40 Stellplätzen zu bauen. Wegen Zeitmangels wird die Anzahl der Stellplätze auf 8 reduziert. Zudem wird nur die vordere Hälfte des Turms gebaut.

Stückliste (Reduzierte Version):

- 1x **Raspberry Pi 4b**: Hauptplatine des Turms.
- 8x **5mm Neopixel RGB Throughhole LEDs**: LEDs für die Anzeige der Boxen.
- 4x **3D-Druck Turmsegment**: Einzelne Segmente des Turm Modells.

Stückliste (Vollständige Version):

- 1x **Raspberry Pi 4b**
- 20x **5mm Neopixel RGB Throughhole LEDs**
- 20x **3D-Druck Turmsegment**

Elektronik Das folgende Ersatzschaltbild zeigt die Verkabelung des Turms. Die LEDs werden über den GPIO Port des Raspberry Pi angesprochen. Die Ansteuerung der LEDs erfolgt über die `Neopixel` Bibliothek. Die Datenleitungen der LEDs sind in Serie geschaltet. Die Stromversorgung der LEDs erfolgt über den 5V Port des Raspberry Pi. Die Stromversorgung des Raspberry Pi erfolgt über das mitgelieferte Raspberry Pi Netzteil.

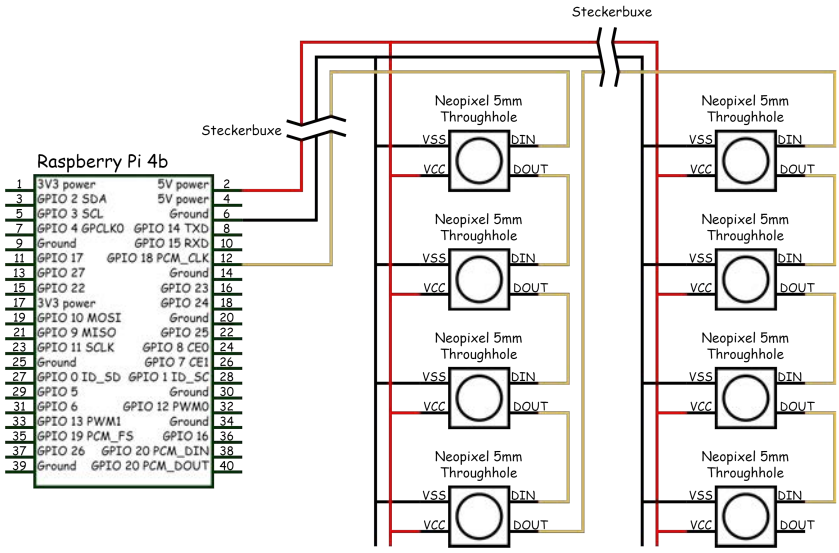


Abbildung 63: Ersatzschaltbild des Turms

Die einzelnen Komponenten werden verlötet. Jede Ebene des Turmes ist einzeln verlötet, sodass die einzelnen Ebenen des Turmes einfach zusammengesteckt werden können.

3D-Druck Die eigentliche Form des Turm Models wird mit dem 3D-Drucker (10.9) gedruckt. Dazu wird das Miniaturmodell zuerst in Autodesk Inventor (10.8) konstruiert (Abbildung 64). Anschließend wird dies als STL-Datei exportiert und in SuperSlicer für den 3D-Druck vorbereitet. Aufgrund der knappen Zeit, werden die Druckparameter so gewählt, dass der Druck möglichst schnell fertig ist. Der G-Code wird exportiert und mittels Octoprint auf einem Wanhao i3 Drucker ausgeführt. Die finale Druckzeit pro Teil beträgt ca. 10 Stunden.

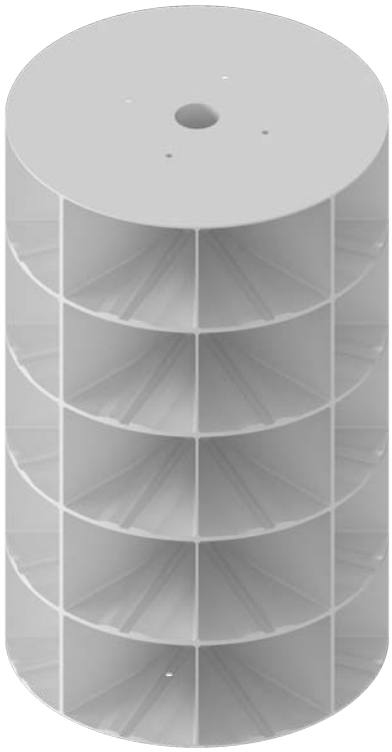
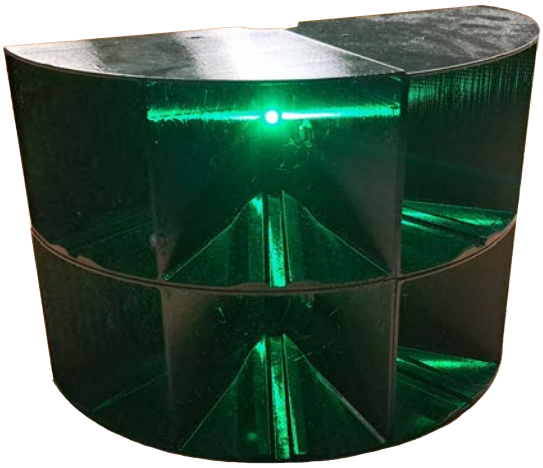
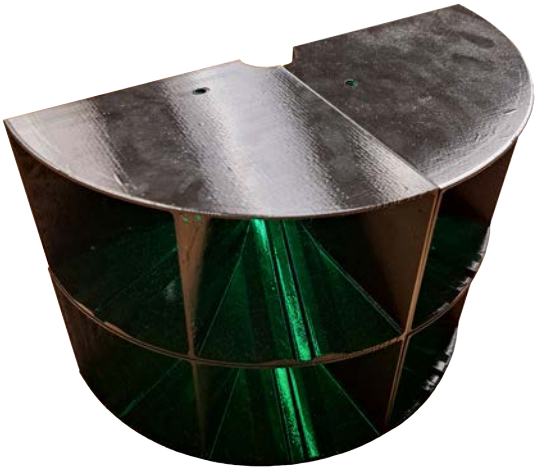


Abbildung 64: Konstruktion des Turmsegmentes

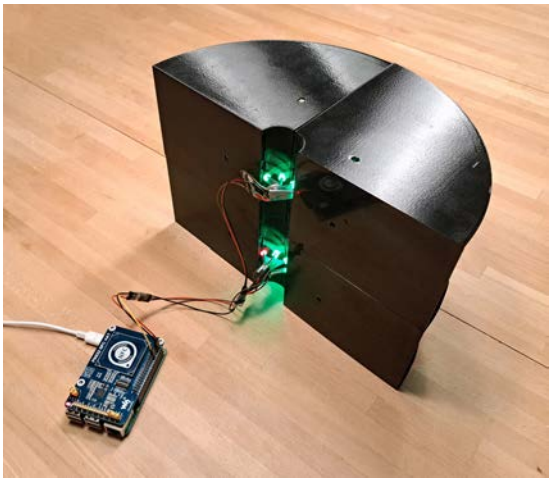
Fertiggestellter Turm Auf den folgenden Abbildungen (65a,65b, 66a, 66b) ist der fertige Turm zu sehen.



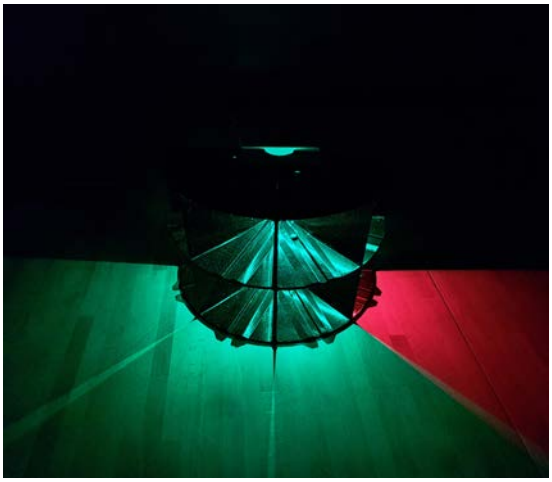
(a) Modell des Turmsegmentes



(b) Druck des Turmsegmentes



(a) Modell des Turmsegmentes



(b) Druck des Turmsegmentes

8.3 Datenbank

Als Datenbank wurde Firestore verwendet, ein Untermodul von Firebase (10.1), welches Nutzern bis zu einem gewissen Umfang kostenlos zur Verfügung steht. Firestore ist eine NoSQL-Datenbank, welche auf Dokumenten basiert. Ähnlich wie bei MongoDB werden Daten als kleine Dokumente gespeichert, die wiederum in Kollektionen gespeichert werden. Firestore erlaubt nesting, damit komplexere Datenstrukturen abgebildet werden können ohne, dass einzelnen Dokumente zu groß werden.

8.3.1 Serialisierung / Deserialisierung

Um die Daten der Datenbank auf der Seite des Tower Controllers zu verarbeiten, müssen diese zunächst in ein strukturiertes Format umgewandelt werden. Dies geschieht durch die Verwendung von `serde` und `serde_json`. `Serde` ist eine Bibliothek, die es ermöglicht, beliebige Datenstrukturen zu serialisieren und deserialisieren. Es ist sehr einfach Datenstrukturen mittels `serde` serialiserbar zu machen.

8.3.2 Datenbankschema

Das Datenbankschema ist, trotz der Verwendung einer Dokumenten-Datenbank, ähnlich zu relationalen Datenbanken (SQL). Aus diesem Grund kann das Schema der Datenbank auch als Entity Relation (ER)-Diagramm dargestellt werden.

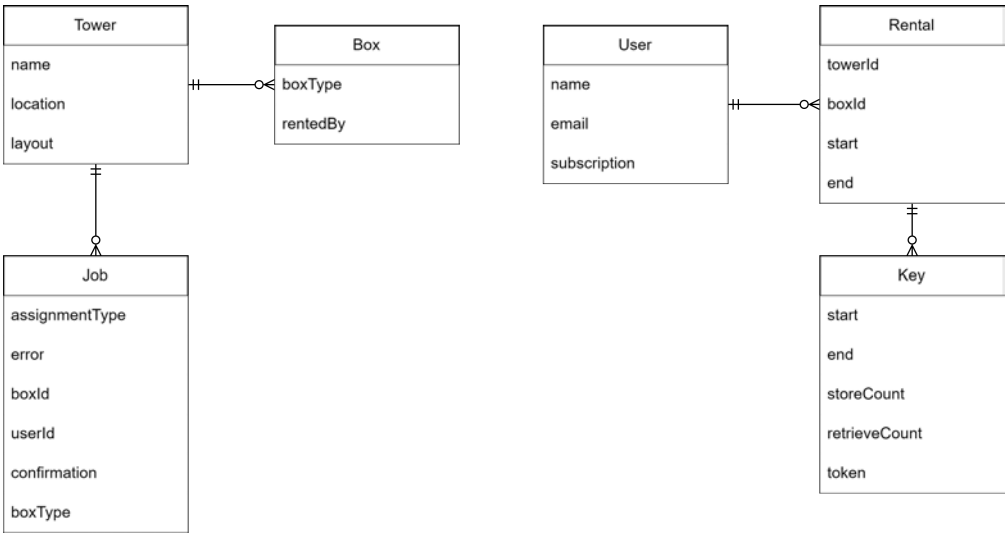


Abbildung 67: ER-Diagramm der Datenbank

Tower Tower repräsentieren die einzelnen Standorte der Fahrradtürme. Sie enthalten die Koordinaten des Standortes, den Namen des Standortes und die Kapazität des Standortes.

```
Tower {
    layout: [number],
    location: LatLng,
    name: String,
}
```

Code-Auflistung 15: Schema für Tower Dokumente

Box Eine Box repräsentiert einen einzelnen Lagerplatz bzw. Stellplatz im Fahrradturm. Sie enthalten die Art des Lagerplatzes und die ID des Nutzers, der den Platz aktuell nutzt (sofern der Stellplatz belegt ist).

```
Box {
    boxType: "bike" | "item",
    rentedBy?: String,
}
```

Code-Auflistung 16: Schema für Box Dokumente

Job Jobs sind für die Kommunikation zwischen Turm und App verantwortlich. Sie enthalten Informationen über den Typ des Jobs, den Nutzer, der den Job anfrägt, die Box, die betroffen ist und den Status des Jobs. Je nach Typ des Jobs und Job-Fortschritt sind manche Felder erforderlich oder leer. Die Kommunikation wird im Abschnitt Kommunikation (8.4) erläutert.

```
Job {
    assignmentType: "store" | "retrieve",
    error?: "noFreeSlots" | "invalidMessage" | "invalidPermissions" | ...,
    boxId?: String,
    userId: String,
    confirmation?: "jobRecieved" | "jobCompleted",
    boxType?: "bike" | "item",
}
```

Code-Auflistung 17: Schema für Job Dokumente

User User enthalten die Informationen über einen einzelnen Nutzer. Sie enthalten den Namen, die E-Mail-Adresse und den Status des Nutzers (ob er ein kostenpflichtiges Abo hat oder nicht).

```
User {
    name: String,
    email: String,
    subscription: "free" | "premium"
}
```

Code-Auflistung 18: Schema für User Dokumente

Rental Ein Rental repräsentiert einen einzelnen Verleihvorgang. Sie enthalten die ID des Turms, bei dem die Box gemietet wurde, die ID der Box, den Startzeitpunkt der Miete und den Endzeitpunkt der Miete (sofern die Miete bereits beendet ist).

```
Rental {
    towerId: String,
    boxId: String,
    start: Timestamp,
    end?: Timestamp,
}
```

Code-Auflistung 19: Schema für Rental Dokumente

Key Keys sind dafür da, um Dritten Zugriff auf vermietete Boxen eines Nutzers zu gewähren. Je nach Verwendungszweck kann ein Ablaufdatum für den Key gesetzt werden. Zudem kann auch ein Limit für die Anzahl der Verwendungen gesetzt werden (separat für Ein- und Auslagerung). Keys werden zum Beispiel dazu verwendet um Paketlieferanten Zugriff auf die Boxen zu gewähren, damit diese ein Paket abliefern können.

```
Key {
    start: Timestamp,
    end?: Timestamp,
    retrieveCount?: number,
    storeCount?: number,
    token: String,
}
```

Code-Auflistung 20: Schema für Key Dokumente

8.4 Kommunikation

App und Turm müssen miteinander kommunizieren. Es gibt mehrere verschiedene Möglichkeiten, jedoch hat sich das Team darauf geeinigt, die bestehende Firebase Datenbank (10.1) und deren „Realtime Updates“ Funktion zu verwenden.

Austauschmöglichkeiten:

- Websocket
- Firebase Cloud Messaging
- Firestore Datenbank Realtime Updates

Abbildung 68: Kommunikation zwischen App und Turm

8.4.1 Realtime Updates

Realtime Updates ist ein Feature von Firestore, welches es ermöglicht, dass sich Client Daten automatisch aktualisieren, sobald sich die Daten in der Datenbank ändern. Ein Listener kann für ein oder mehrere Dokumente/Collections registriert werden und sobald sich die Daten in der Datenbank ändern, wird der Listener automatisch über die Änderungen informiert.

8.4.2 Fahrradturm Protokoll

Das Fahrradturm Protokoll basiert auf Dokumenten in der Firestore Datenbank. Jedes Dokument repräsentiert einen einzelnen Job, der vom Fahrradturm ausgeführt werden soll. Die Dokumente werden in einer Subcollection des Turms namens „jobs“ gespeichert. Um Jobs zu starten muss der Client ein Dokument im Turm erstellen.

Gemeinsame Felder Es gibt einige Felder, welche bei jedem Job vorhanden sind bzw. immer die gleiche Bedeutung haben. Diese Felder sind wie folgt:

- `error` ist ein optionales Feld, welches vom Fahrradturm gesetzt wird, wenn ein Fehler auftritt. Es gibt verschiedene, mögliche Werte, welche die Art des Fehlers beschreiben. Anhand dieser Fehlermeldungen kann der Client die Fehlermeldung dem Nutzer anzeigen.
- `userId` ist die ID des Nutzers, der den Job ausführen möchte. Sie wird verwendet um festzustellen, ob der Nutzer berechtigt ist den Job auszuführen.

- `confirmation` ist ein optionales Feld, welches vom Fahrradturm gesetzt wird, um dem Client mitzuteilen, dass der Job erfolgreich empfangen bzw. ausgeführt wurde. Es gibt zwei mögliche Werte: `jobRecieved` und `jobCompleted`. `jobRecieved` wird gesetzt, wenn der Fahrradturm den Job empfangen hat. `jobCompleted` wird gesetzt, sobald der Fahrradturm den Job erfolgreich ausgeführt hat.
- `assignmentType` gibt an, welche Art von Job ausgeführt werden soll. Es gibt zwei mögliche Werte: `store` und `retrieve`.

Einlagern Einlager Jobs müssen folgende Felder enthalten:

```
Job {
    assignmentType: "store",
    userId: String,
    boxType: "bike" | "item",
}
```

Code-Auflistung 21: Einlager Anfrage Assignment Struktur

- `assignmentType` gibt an, dass es sich um einen Einlagerjob handelt.
- `boxType` gibt an, ob es sich um ein Fahrrad oder ein anderes Objekt handelt.

Falls der Fahrradturm den Job erfolgreich empfangen hat, wird das Feld `confirmation` auf `jobRecieved` gesetzt. Nun wird der Job auf der Seite des Fahrradturms ausgeführt. Sobald der Job abgeschlossen ist, wird das Feld `confirmation` auf `jobCompleted` gesetzt. Das Feld `boxId` wird ebenfalls gesetzt, welches die ID der Box ist, in der das Objekt/Fahrrad abgelegt wurde. Ein abgeschlossener Einlagerjob sieht wie folgt aus:

```
Job {
    assignmentType: "store",
    userId: String,
    boxType: "bike" | "item",
    confirmation: "jobCompleted",
    boxId: String,
}
```

Code-Auflistung 22: Einlagerbestätigung Assignment Struktur

Auslagern Auslager Jobs müssen folgende Felder enthalten:

```
Job {
    assignmentType: "retrieve",
    userId: String,
    boxId: String,
}
```

Code-Auflistung 23: Auslager Anfrage Assignment Struktur

- `assignmentType` gibt an, dass es sich um einen Auslagerjob handelt.
- `boxId` ist die ID der Box, aus der das Item/Fahrrad entnommen werden soll.

Ein abgeschlossener Auslagerjob sieht wie folgt aus:

```
Job {
    assignmentType: "retrieve",
    userId: String,
    boxId: String,
    confirmation: "jobCompleted",
  }
```

Code-Auflistung 24: Auslagerbestätigung Assignment Struktur

9 Marktskalierung

Die Marktskalierung beschäftigt sich damit, wie ein automatischen Fahrradparkhaus finanziert und womit Einnahmen erzielt werden können.

9.1 Jugend Innovativ

> „*Jugend Innovativ ist Österreichs smartester Schulwettbewerb für innovative Ideen und fördert die besten Talente seit 1987.*“ (vgl. *Über Jugend Innovativ- Jugend Innovativ* 2023)

Jugend Innovativ ist ein Wettbewerb, bei dem Schüler:innen ihre Ideen einreichen können. Die besten Ideen kommen dann in die nächste Runde und können dort weitere Preise gewinnen. Die Projekte werden den Kategorien ICT und Digital, Engineering, Science, Design, Entrepreneurship und Sustainability zugeordnet (vgl. *Jugend Innovativ* 2023).

Das Projektteam hat sich für die Kategorie Engineering entschieden, da diese am besten zum Projekt passt. Das Projektteam hat sich bis zum jetztigen Zeitpunkt für das Halbfinale qualifiziert und wartet momentan darauf, ob das Projekt in die nächste Runde kommt. Um sich für das Finale zu qualifizieren, musste das Projektteam einen etwa 20 seitigen Projektbericht einreichen. Dieser Projektbericht wurde von den Projektmitgliedern selbst erstellt und beinhaltet alle Informationen über das Projekt. Um den Projektbericht anschaulicher zu gestalten, wurde mittels CINEMA 4D ein Render des Turms erstellt. Dieser Render wurde als Deckblatt verwendet und ist in Abbildung 69 zu sehen.

Abbildung 69: Visualisierung des Turms im Projektbericht

9.2 Einnahmeströme und Austauschprozesse

Das Projektteam hat die folgenden Einnahmeströme und Austauschprozesse identifiziert:

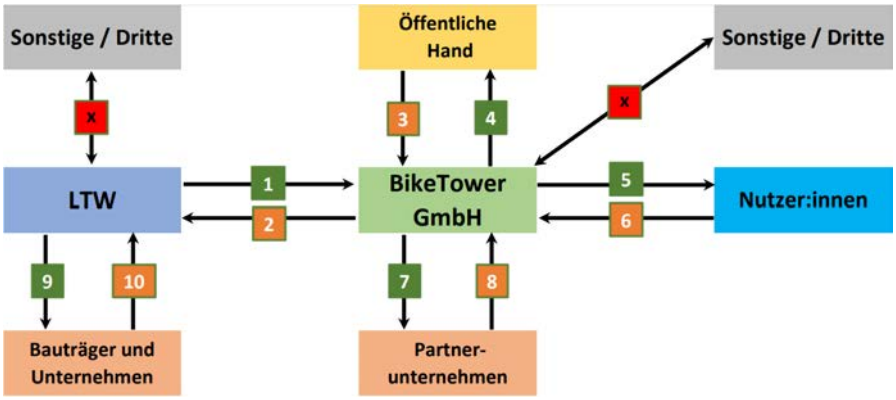


Abbildung 70: Einnahmeströme

Zahlungsströme / Austauschprozesse			
X	**LTW**	Auszahlungen	„Kosten“
2		Einzahlungen	Verkaufserlöse - BikeTower
10			Verkaufserlöse - Andere
3	**BikeTower GmbH**	Einzahlungen	Subventionen
6			Nutzerendgelte
8			Sonstige (Daten, Miete, etc...)
X		Auszahlungen	„Kosten“
1	BikeTower GmbH		
4	Öffentliche Hand		
5	Nutzer:innen	Kundennutzen	
7	Partner-Unternehmen		
9	Bauträger u. Unternehmen		

Tabelle 15: Einnahmeströme

9.3 Absatzplan

Übersicht Modelle

Modell	KAPAZITÄT	Werteinsatz / Box	VPr. / Box	Subventionen
LTW 15	212 Boxen	2 845 €	3 770 €	3 205 €
LTW 12	104 Boxen	4 396 €	5 770 €	4 905 €
Rondel 20	104 Boxen	9 298 €	12 020 €	10 217 €

Tabelle 16: Übersicht Modelle

Absatzzahlen - BikeTower GmbH (2)

Modell	2024	2025	2026	2027	2028
LTW 15	0 Stk.	0 Stk.	2 Stk.	2 Stk.	3 Stk.
LTW 12	3 Stk.	3 Stk.	6 Stk.	6 Stk.	8 Stk.
Rondel 20	4 Stk.	7 Stk.	10 Stk.	14 Stk.	18 Stk.

Tabelle 17: Absatzzahlen - BikeTower GmbH

Absatzzahlen - Andere (10)

Modell	2024	2025	2026	2027	2028
LTW 15	3 Stk.	5 Stk.	8 Stk.	10 Stk.	12 Stk.
LTW 12	2 Stk.	5 Stk.	8 Stk.	12 Stk.	25 Stk.
Rondel 20	1 Stk.	2 Stk.	4 Stk.	6 Stk.	16 Stk.

Tabelle 18: Absatzzahlen - Andere

9.3.1 Stellplätze Zuwachs

Stellplätze Zuwachs - BikeTower GmbH (2)

Modell	2024	2025	2026	2027	2028
LTW 15	0 Boxen	0 Boxen	424 Boxen	424 Boxen	636 Boxen
LTW 12	312 Boxen	312 Boxen	624 Boxen	624 Boxen	832 Boxen
Rondel 20	416 Boxen	728 Boxen	1 040 Boxen	1 456 Boxen	1 872 Boxen
Gesamt	728 Boxen	1 040 Boxen	2 088 Boxen	2 504 Boxen	3 340 Boxen

Tabelle 19: Stellplätze Zuwachs - BikeTower GmbH

Stellplätze Zuwachs - Andere (10)

Modell	2024	2025	2026	2027	2028
LTW 15	636 Boxen	1 060 Boxen	1 696 Boxen	2 120 Boxen	2 544 Boxen
LTW 12	208 Boxen	520 Boxen	832 Boxen	1 248 Boxen	2 600 Boxen
Rondel 20	104 Boxen	208 Boxen	416 Boxen	624 Boxen	1 664 Boxen
Gesamt	948 Boxen	1 788 Boxen	2 944 Boxen	3 992 Boxen	6 808 Boxen

Tabelle 20: Stellplätze Zuwachs - Andere

Stellplätze akkumuliert - Jahresbestand

Modell	2024	2025	2026	2027	2028
LTW 15	636 Boxen	1 060 Boxen	1 484 Boxen	1 908 Boxen	2 544 Boxen
LTW 12	520 Boxen	832 Boxen	1 456 Boxen	2 080 Boxen	2 912 Boxen
Rondel 20	520 Boxen	936 Boxen	1 976 Boxen	3 432 Boxen	5 304 Boxen
Gesamt	1 676 Boxen	2 828 Boxen	4 916 Boxen	7 420 Boxen	10 760 Boxen

Tabelle 21: Stellplätze akkumuliert - Jahresbestand

9.3.2 Investitionssummen

Investitionssummen - BikeTower GmbH (tsd. €)

AW	Modell	2024	2025	2026	2027	2028
799	LTW 15	-	-	1 598	1 598	2 398
600	LTW 12	1 800	1 800	3 600	3 600	4 801
1 250	Rondel 20	5 000	8 751	12 501	17 501	22 501
	Gesamt	6 800	10 551	17 699	22 699	29 700

Tabelle 22: Investitionssummen - BikeTower GmbH

9.4 Kostenrechnung

Es wurde eine Nutzungsdauer von 25 Jahren angenommen, zusätzlich wurde mit einem Zinssatz von 6% gerechnet.

Kosten (tsd. €)	**2024**	**2025**	**2026**	**2027**	**2028**
Personalkosten	250	284	346	420	519
Management - Verwaltung	120	120	120	120	120
Vertrieb	80	80	80	80	80
Instandhaltung - Wartung	50	84	146	220	319
Sonstige Kosten	135	145	155	165	175
Marketing	80	80	80	80	80
IT	30	30	30	30	30
Strom (Energie)	25	35	45	5	6
Kalkulatorische Abschreibung	40	69	116	153	201
Kalkulatorische Zinsen	30	52	87	115	150
KOSTEN - GESAMT	456	550	705	853	1 046
ERTRÄGE (ohne Subventionen)	348	588	1 022	1 543	2 238
Betriebsergebnis	-107	37	317	689	1 191

Tabelle 23: Kostenrechnung

9.5 Finanzplan

9.5.1 LTW Finanzplan

Um die Liquidität und den Cash-Flow der BikeTower GmbH zu sichern, wird ein Finanzplan erstellt. Die Planung wurde bis 2028 durchgeführt.

Zahlungsströme / Austauschprozesse (tsd. €)				**2024**	**2025**	**2026**	**2027**	**2028**
X	**LTW**	Auszahlungen	„Kosten“	1.809	3.016	6.031	7.237	9.047
2		Einzahlungen	Verkaufserlöse - BikeTower	-	-	1.598	1.598	2.397
10			Verkaufserlöse - Andere	2.397	3.996	6.393	7.992	9.590
			Cash-Flow	588	980	1.961	2.353	2.941

Tabelle 24: Finanzplan

9.5.2 BikeTower GmbH Finanzplan

Annahmen:

- Subventionen: 85%
- Durchschnittliche Auslastung: 80%
- Jahresbeitrag / Nutzer:in: 240€
- Entgelt für Datensätze/Nutzer/Jahr: 20€

Finanzplan (tsd. €)

Zahlungsströme / Austauschprozesse (tsd. €)		**2024**	**2025**	**2026**	**2027**	**2028**
Einzahlungen (betrieblich)		6 128	9 556	16 066	20 837	27 483
Einnahmeströme	Subventionen (3)	5 780	8 968	15 044	19 294	25 245
	Nutzerentgelte (6)	321	543	943	1 424	2 065
	Vermietung von Boxen (8_A)					
	Werbeeinnahmen (8_B)					
	Daten (8_C)	26	45	78	118	172
	Sonstige (8_D)					
Auszahlungen		7 365	11 160	18 380	23 464	30 574
Cash-Flow (betrieblich)		-1 236	-1 603	-2 313	-2 626	-3 090
Gesellschaftereinlagen		1 000				
Überdeckung/ Unterdeckung		-236	-1 603	-2 313	-2 626	-3 090
Personal	Personalausgaben	430	464	526	600	699
	Management - Verwaltung	180	180	180	180	180
	Vertrieb	200	200	200	200	200
	Instandhaltung - Wartung	50	84	146	220	319
Investitionen	alle Modelle	6 800	10 551	17 699	22 699	29 700
	LTW 15	-	-	1 598	1 598	2 398
	LTW 12	1 800	1 800	3 600	3 600	4 801
	Rondel 20	5 000	8 751	12 501	17 501	22 501
	Sonstige Ausgaben	135	145	155	165	175
	Marketing	80	80	80	80	80
	IT	30	30	30	30	30
	Strom (Energie)	25	35	45	55	65

Anhand der Ergebnisse des Finanzplans ist das Projektteam zu der Einsicht gekommen, dass die Anschaffungskosten der Türme zu fast 100% subventioniert werden müssen. Anhand der Annahmen, dass die Auslastung der Türme bei 80% liegt, ist die Überdeckung der Kosten durch die Einnahmen nicht gegeben. Die Einnahmen aus den Nutzerentgelten sind zu gering, um die Kosten zu decken. Jedoch reichen die Nutzerentgelte aus, um die laufenden Kosten (Betriebskosten) abzudecken. Dies beinhaltet Wartung, Personalkosten und Stromkosten. Die Summe der Investitionssummen für die Türme liegt bei 74,3 Millionen € und werden als etwaige Infrastrukturinvestitionen vom Staat finanziert.

10 Verwendete Technologien

Bei der Durchführung der praktischen und schriftlichten Arbeit werden die folgenden Technologien und Programme vom Projektteam verwendet.

10.1 Firebase

Firebase ist eine Plattform von Google, welche es Entwickler:innen ermöglicht, eine Vielzahl von Funktionen in ihre Applikationen zu integrieren. Diese Funktionen sind zum Beispiel Authentifizierung, Datenbanken, Cloud Messaging und Hosting. Die Plattform ist für kleine Projekte kostenlos und bietet eine kostenlose Testversion mit 1 GB Speicherplatz und 50.000 Anfragen pro Tag. Firebase vereinfacht das Entwickeln von Mobil- und Webapplikationen, da die Entwickler:innen sich nicht um die Infrastruktur kümmern müssen. Die Plattform ist in verschiedene Komponenten aufgeteilt, die einzeln oder zusammen verwendet werden können. Firebase verwendet ein „pay as you go“ Modell bei dem Nutzer:innen pro Anfrage einen bestimmten Betrag bezahlen. Firebase ist Teil von Google Cloud Platform.

Abbildung 71: Firebase Logo (vgl. G. Inc. 2020)

Um die Entwicklung weiter zu vereinfachen, bietet Google offizielle Firebase SDKs für viele gängige Programmiersprachen an. Diese SDKs bieten eine einfache API, welche die Kommunikation mit der Plattform vereinfacht. Die API ist sehr detailliert dokumentiert, weshalb es auch SDKs für andere Programmiersprachen gibt.

Firebase vereinfacht zwar die urprüngliche Entwicklung, jedoch gibt es das Problem von „Vendor Lock-in“. Dies bedeutet, dass die Entwickler:innen auf die Plattform von Firebase angewiesen sind, da diese sehr eng mit Applikationen integriert sind.

10.2 Raspberry PI

Raspberry PIs sind eine Familie an Einplatinencomputer (SBCs), die von der Raspberry PI Foundation entwickelt und verkauft werden. Die Acorn RISC Machine (Arm) basierten Computer haben etwa die Grundfläche einer Kreditkarte und sind etwa 2 cm hoch. Trotz ihrer kleinen Größe haben sie eine achtbare Rechenkapazität (neuere Modelle sogar bis zu 8 GB Arbeitsspeicher (RAM) (vgl. *Raspberry Pi 4 Model B* 2023)). Dank der vielen Peripheriegeräte, wie GPIO, Serial Peripheral Interface (SPI) und Inter-Integrated Circuit (I2C) sind diese kleinen Computer ideal für das Steuern von Hardware und Internet of Things (IoT) Anwendungen.

Abbildung 72: Raspberry PI 4b (vgl. *Raspberry Pi 4 Model B* 2023)

Anders als bei Mikrocontrollern muss sich der Entwickelnde bei einem Raspberry PI nicht um alles kümmern. Zwischen Hard- und Software gibt es einen Kernel und das Linux Betriebssystem. Diese übernehmen viele der Aufgaben und erleichtern die Entwicklung. Somit ist es zum Beispiel möglich eine Python Applikation auf dem Raspberry PI auszuführen.

10.3 Neopixel

Neopixel ist eine Serie an LED Produkten vom Hersteller Adafruit. Neopixel vereinfachen die Verwendung von LEDs, da sie bereits mit einem Mikrocontroller ausgestattet sind. Die LEDs sind in verschiedenen Größen und Formen erhältlich und können über eine einzige Datenleitung angesteuert werden. Dieses Feature erlaubt unter anderem Daisy-Chaining, was es stark erleichtert, viele LEDs mit wenigen Pins anzusteuern. Softwarebibliothenken für die Neopixel sind in vielen gängigen Programmiersprachen verfügbar (vgl. *NeoPixels* 2023).

Adafruit wurde 2005 von Limor Fried gegründet. Sie ist eine amerikanische Ingenieurin und Informatikerin, die sich das Ziel gesetzt hat Elektronik einfacher und zugänglicher zu gestalten (vgl. *Adafruit About Us* 2023).

10.4 Rust

Rust ist eine kompilierte System-Programiersprache, die trotz high-level Features hohe Ausführungsgeschwindigkeiten verspricht. Anders als herkömmliche Programiersprachen verwendet Rust keinen Garbage Compiler, jedoch garantiert sie mit Ressourcenverwaltung durch Initialisierung (RAII) und einem unkonventionellen *Borrow Checker* trotzdem Memory-Safety.

Rust wurde 2006 von Softwareentwickler Graydon Hoare erfunden und wird seit 2009 von Mozilla gesponsert (vgl. *Rust (programming language)* 2023). In den letzten Jahren verwenden immer mehr Entwickler, auch große Firmen wie Google, Rust für ihre Projekte. Rust ist eine der am schnellsten wachsenden Programmiersprachen und hat sich in den letzten Jahren zu einer der beliebtesten System-Programmiersprachen entwickelt (vgl. *Stack Overflow Developer Survey* 2023). Rust wurde vom Projektteam für den Turm-Prototypen verwendet.

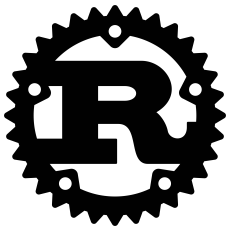


Abbildung 73: Rust Logo

10.4.1 `firestore_rs` Softwarebibliothek

Die `firestore_rs` Softwarebibliothek ist eine Rust Softwarebibliothek, welche die Kommunikation mit der Google Firestore Datenbank vereinfacht. Die Softwarebibliothek ist Open-Source auf GitHub und kann sehr einfach mittels crates.io zu einem Projekt hinzugefügt werden. Die Softwarebibliothek ist sehr einfach zu bedienen und bietet eine große Anzahl an Funktionen, welche die Kommunikation mit der Firestore Datenbank vereinfachen.

Die Bibliothek umfasst unter anderem folgende Funktionen:

- Erstellen, Lesen, Aktualisieren, Löschen (CRUD) Operationen
- Querys erstellen
- Datenbank Listener

Vorallem die Listener Funktion ist für das Projekt sehr wichtig, da dieses Feature zu kompliziert ist, um es selbst zu implementieren. Die Listener Funktion ermöglicht es, dass sich das System automatisch aktualisiert, wenn sich Daten in der Datenbank ändern. Dieses Feature ist für das Projekt essenziell, da es die Kommunikation zwischen den einzelnen Komponenten ermöglicht.

Es wurden kleinere Änderungen an der Softwarebibliothek vorgenommen, um die Kommunikation mit der Datenbank zu vereinfachen. Die Änderungen wurden in einem Fork der Softwarebibliothek vorgenommen und können im GitHub Repository[27] eingesehen werden. Manche dieser Änderungen wurden mittels einer Pull-request bereits in den Master-branch der Softwarebibliothek übernommen und sind ab Version `v0.29.0` verfügbar.

10.5 React Native

React Native ist ein Open-Source-Framework, das von Meta erstellt wurde. Es ermöglicht die Entwicklung von nativen, mobilen Anwendungen für verschiedene Betriebssysteme (Android, iOS, Windows, macOS, ...) mit einer gemeinsamen Codebasis. React Native wird von vielen bekannten Unternehmen, wie zum Beispiel Meta oder Microsoft zur Entwicklung von Apps verwendet.
React Native verwendet Elemente wie zum Beispiel <Text> um Text darzustellen oder <Image> für ein Bild. Diese Elemente werden in den nativen Code der Android- oder iOS-App umgewandelt. (vgl. *React Native · Learn once, write anywhere* 2022)

Abbildung 74: React Naive Logo (vgl. *React Native · Learn once, write anywhere* 2022)

10.6 Javascript

JavaScript ist eine Programmiersprache, die bei der Entwicklung von interaktiven Webanwendungen verwendet wird.(vgl. *Grundlagen* 2022) Da React-Native auf Javascript basiert, verwendet das Projektteam, die Programmiersprache für die Entwicklung der App. (vgl. *React Native im Überblick* 2021)

Abbildung 75: Javascript Logo (vgl. Williams 2011)

[27] `https://github.com/jokil123/firestore-rs`

10.7 Expo

Expo ist ein Open-Source-Framework, das auf React-Native aufgebaut ist. Es ermöglicht eine schnelle und einfache Entwicklung einer App mit React Native, ohne dass man sich um die Konfiguration und den Build-Prozess kümmern muss. (vgl. *Expo* 2023)
Mithilfe der Expo Go App kann man die App außerdem jederzeit problemlos testen, auch für iOS, ohne dass man ein Gerät mit MacOS benötigt. Expo Go führt die App auf dem verwendeten Gerät aus und stellt wichtige Schnittstellen zur Verfügung. (vgl. Ravichandran 2022)

Abbildung 76: Expo Logo (vgl. *Expo* 2023)

10.8 Autodesk Inventor

Inventor ist eine von Autodesk entwickelte CAD/CAM Software. Mittels Autodesk Inventor können Bauteile digital konstruiert werden, Baugruppen zusammengebaut werden und Bauteil Simulationen durchgeführt werden. Fertige Bauteile können direkt zur additiven Herstellung exportiert werden. Ansonsten können mittels der computergestützten Fertigung (CAM) Suite Fräsanleitungen für automatisierte Fräs- und Drehanlagen erstellt werden.

Inventor unterstützt viele gängige 3D Konstruktionsmethoden, unter anderem Subdivision Modeling, Spline Patch Modeling und 2D- und 3D-skizzenbasierte Konstruktion.

Inventor hat exzellenten Cloud support mithilfe von Autodesk Vault einer in-house Komplettlösung für kooperatives Konstruieren und Versionskontrolle in Teams. Die Autodesk Vault Applikation befindet sich auf einem zentralen Server und arrangiert laufend die Änderungen verschiedener Benutzer. Aber auch ohne Autodesk Vault ist das Zusammenarbeiten recht einfach. Wenn Projekte richtig angelegt werden, können diese problemlos in einer Cloud wie Microsoft SharePoint gespeichert werden und Änderungen werden allen Team-Mitgliedern immer in Echtzeit übertragen.

10.9 3D Druck

Fused Deposition Modeling (FDM) umgangssprachlich auch 3D Druck genannt ist eine Art der additiven Fertigung. Bei FDM werden CAD Modelle in viele einzelne Schichten aufgeteilt und diese dann Schicht für Schicht aufgebaut. Dabei wird ein Kunststoff in einem Druckkopf (Extruder) erwärmt und dann auf die vorherige Schicht aufgetragen. Dieser Vorgang wird solange wiederholt bis das CAD Modell fertig gedruckt ist. FDM ist eine der am meisten verbreiteten Methoden um 3D Objekte zu drucken. FDM ist auch die Methode, die vom Projektteam gewählt wurde. Andere Formen des 3D-Drucks wie SLA oder SLS funktionieren auf einem ähnlichen Prinzip, jedoch sind diese Methoden teurer, komplizierter und oft langsamer. Eine Darstellung des FDM Verfahrens ist in Abbildung 77 zu sehen.

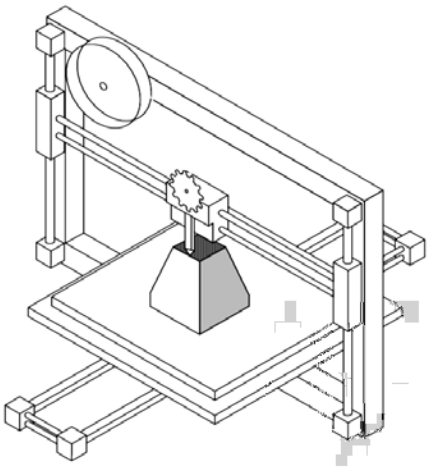


Abbildung 77: Schematische Darstellung des FDM Verfahrens (vgl. C. F. Inc. 2019)

10.9.1 SLA

SLA ist ein Verfahren bei dem ein flüssiges Harz mit einem UV-Laser selektiv, Schicht für Schicht, ausgehärtet wird. Dieser Vorgang wird solange wiederholt bis das CAD Modell fertig gedruckt ist. SLA ist eine der teuersten Methoden um 3D Objekte zu drucken (vgl. *Stereolithografie (SLA) 3D-Druck* 2023). Eine Darstellung des SLA Verfahrens ist in Abbildung 78 zu sehen.

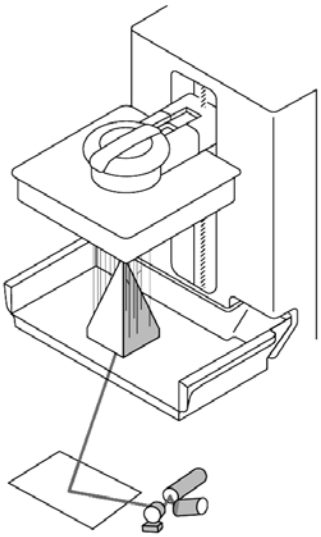


Abbildung 78: Schematische Darstellung des SLA Verfahrens (vgl. C. F. Inc. 2019)

10.9.2 SLS

Bei SLS wird Kunststoffpulver von einem Laser geschmolzen. Sobald eine Ebene vollendet ist wird neues Pulver aufgetragen und der Vorgang wird wiederholt (vgl. *Leitfaden zum 3D-Druck mit selektivem Lasersintern (SLS)* 2023). SLS hat unter anderem den Vorteil, dass Bauteile keine Stüztstrukturen benötigen. Eine Darstellung des SLS Verfahrens ist in Abbildung 79 zu sehen.

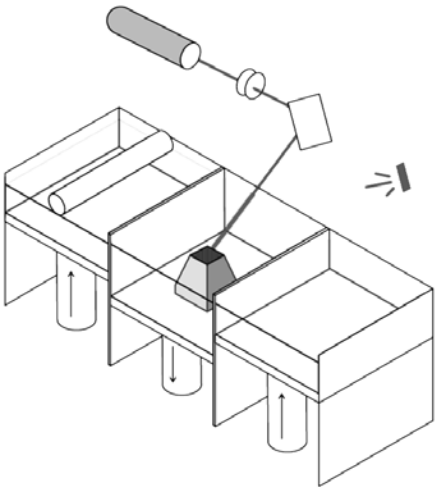


Abbildung 79: Schematische Darstellung des SLS Verfahrens (vgl. C. F. Inc. 2019)

10.10 Cinema 4D

Cinema 4D ist eine von Maxon entwickelte 3D Suite, welche Nutzern eine Vielzahl an Tools zur Verfügung stellt um 3D Objekte zu erstellen, animieren, visualisieren und exportieren (vgl. *3D Modeling, Texturing, Lighting, Animation and Simulation Software |...* 2023). Maxon ist eine deutsche Softwarefirma, welche im Jahr 1986 gegründet wurde. Die erste Version von Cinema 4D wurde 1993 veröffentlicht, und hat seitdem regelmäßig Updates und neue Features erhalten.

Dank Cinema 4Ds einfacher und benutzerfreundlicher Bedienoberfläche und des simplen Programaufbaus, ist Cinema 4D vor allem bei Anfängern sehr beliebt. Besonders Tutorials auf YouTube erlaubt es vielen Benutzern die Software selbständig zu lernen.

Abbildung 80: Cinema 4D Logo (vgl. M. C. GmbH 2011)

10.11 Visual Studio Code

Visual Studio Code ist Code-Editor, der von Microsoft entwickelt wurde. Ein großer Vorteil von Visual Studio Code ist die kostenfreie Nutzung und dass es sehr viele Erweiterungen und Plugins gibt (vgl. *Visual Studio Code* 2023). Das Programm wurde im Laufe der Diplomarbeit bei der Softwareentwicklung der App und des Controllers verwendet.

Abbildung 81: Visual Studio Code Logo (vgl. *Datei:Visual Studio Code 1.35 icon.svg – Wikipedia* 2019)

10.12 GitHub

GitHub ist eine Online-Plattform für die Versionsverwaltung von Softwareprojekten. Es bietet eine Möglichkeit für Entwickler:innen, den Code online zu teilen und zu speichern, um effektiv mit anderen zusammenzuarbeiten und Änderungen an Projekten zu verwalten (vgl. *Was ist GitHub? Eine Einführung in GitHub für Einsteiger* 2020).
GitHub wird bei der Diplomarbeit verwendet, um den Code der App und des Controllers sowie die Dokumentation der Arbeit zu teilen und die Zusammenarbeit zu ermöglichen.

Abbildung 82: GitHub Logo (vgl. *GitHub Logos and Usage* 2023)

10.13 LaTeX

LaTeX ist ein Schriftsetzungs-System, ausgelegt für technische und wissenschaftliche Dokumente. LaTeX ist eine Weiterentwicklung von TeX und wurde 1984 zum ersten Mal veröffentlicht. Es ist der de facto Standard für wissenschaftliche Dokumente und wird, da es Open-Source ist, kontinuierlich weiterentwickelt. (vgl. *LaTeX - A document preparation system* 2023; *LaTeX* 2023)

Latex wurde vom Projektteam verwendet, um die schrifliche Arbeit zu erstellen. Es bietet viele quality-of-life Features wie automatische Bild- und Tabellennummerierung, Referenzierung und Inhaltsverzeichnisse. Da ein LaTeX Projekt in viele kleine Dateien aufgeteilt werden kann, ist es einfach ein Versionskontrollsystem wie Git zu verwenden. Dies erleichterte die Zusammenarbeit und die Verwaltung von Änderungen.

11 Zusammenfassung

In einem ersten Gespräch beim Partnerunternehmen hat das Projektteam (Paul Hartmann und Joshua Lung) schnell festgestellt, dass das Thema „automatisches Fahrradparkhaus“ interressant ist und hat die ersten Ziele gemeinsam formuliert.
Die Ziele sind die Durchführung einer Marktanalyse, Konzeptionierung eines Fahrradparkhauses, Entwicklung einer App zum Buchen von Abstellplätzen und das Erstellen eines Modells des Fahrradturms. Für die Ziele sind entsprechende Meilensteine und Termine definiert, damit die Projektarbeit strukturiert voran geht. Zusätzlich wurden die aufgewendete Arbeitszeit pro Person dokumentiert.

Ausgangspunkt der Arbeit ist die Marktanalyse. Dabei werden bereits bestehende Systeme zur Lagerung von Fahrrädern analysiert, Recherchen und Anfragen bei Unternehmen gemacht und Informationen über gesetzliche Richtlinien gesammelt. Außerdem wird eine Online-Umfrage durchgeführt, um die Anforderungen und Interessen von Radfahrer:innen zu erfassen.

Anschließend werden zwei neue automatische Fahrradparksysteme konzeptioniert. Bei dem Hochregallager handelt es sich um ein neuartiges System, dass bisher noch nicht für die Lagerung von Fahrrädern verwendet wird. Das Rondell mit Auslagerung von außen ist eine verbesserte Version von bereits verwendeten Modellen.

Als Hauptteil der praktischen Ausführung wird ein Prototyp des Turmes gebaut und eine App entwickelt. Mit der App kann man Fahrradparkhäuser in der Umgebung finden, einen Account erstellen und ein Fahrrad oder einen Gegenstand bei einem Fahrradparkhaus abgeben oder abholen. Dabei handelt es sich um ein Modell des konstruierten Fahrradparkhauses, welches mithilfe von LEDs die verfügbaren Plätze darstellt.

Alle gesetzten Meilensteine wurden zeitgerecht erreicht. Da die Abgabe der Diplomarbeit bis zum 31.03 erfolgen muss, wurde die praktische Ausarbeitung des Prototyps am 20.02 beendet. Einzelne Funktionen wie die Möglichkeit, den Zugriff auf ein gelagertes Fahrrad zu gewähren, wurden aufgrund des strikten Zeitplanes nicht umgesetzt.

Mit der Arbeit ist ein Grundstein für den Bau eines Fahrradparkhauses gelegt. Die App kann mit gewissen Anpassungen später veröffentlicht und verwendet werden. Mit dem Prototypen kann man eine Buchung darstellen und vorzeigen.
Zusammengefasst ist das Projektteam sehr zufrieden mit dem Ergebnis der Arbeit. Die Arbeit besteht aus einem breiten Themengebiet und es wurden viele nützliche Ergebnisse erzielt und Erfahrungen gesammelt.

12 Literatur- und Quellenverzeichnis

3D Modeling, Texturing, Lighting, Animation and Simulation Software |... (2023). Maxon. URL: https://www.maxon.net/en/cinema-4d/features (besucht am 20. 03. 2023).

Adafruit About Us (2023). URL: https://www.adafruit.com/about (besucht am 14. 03. 2023).

AG, V.-Locker (2023a). *V-Locker I Fahrradparkturm I Intelligente Fahrrad Abstellanlagen Türme.* URL: https://v-locker.ch/standorte-und-preise/ (besucht am 07. 03. 2023).

— (2023b). *V-Locker I Fahrradparkturm I Intelligente Fahrrad Abstellanlagen Türme.* URL: https://v-locker.ch/bike-parking-system/ (besucht am 06. 03. 2023).

Amsterdam eröffnet Unterwasser-Parkgaragen für Fahrräder (o. D.). URL: https://www.derstandard.at/story/2000143170008/amsterdam-eroeffnet-unterwasser-parkgaragen-fuer-fahrraeder. (Zugriff: 14.03.2023).

anna.cherniavska (29. Apr. 2021). *Xamarin vs React Native. Plattformübergreifende App Entwicklung.* Chudovo. URL: https://chudovo.de/xamarin-vs-react-native-welche-software-soll-man-2021-waehlen/ (besucht am 11. 08. 2022).

Borkert, Interview: Stefan (2020). *Interview - «Neue Velowege allein genügen nicht» – Jetzt kommen die smarten Veloparktürme für den Thurgau.* URL: https://www.tagblatt.ch/wirtschaft/neue-velowege-allein-genugen-nicht-ld.1244373 (besucht am 06. 03. 2023).

Cloud Firestore-Datenmodell | Firebase (2023). URL: https://firebase.google.com/docs/firestore/data-model (besucht am 24. 01. 2023).

CNN Sol Han, by Kate Springer (2016). *Japan's crazy underground bike vaults.* URL: https://www.cnn.com/style/article/japan-underground-bike-vaults/index.html (besucht am 10. 03. 2023).

Datei:Visual Studio Code 1.35 icon.svg – Wikipedia (2019). URL: https://commons.wikimedia.org/wiki/File:Visual_Studio_Code_1.35_icon.svg (besucht am 13. 03. 2023).

Diagram Software and Flowchart Maker (2023). URL: https://www.diagrams.net/ (besucht am 08. 03. 2023).

E-Bikes in Türmen geparkt - V-Locker gewinnt "Bundespreis Ecodesign 2022" (o. D.). URL: https://innovators-guide.ch/2022/12/e-bikes-in-tuermen-geparkt-v-locker-gewinnt-bundespreis-ecodesign-2022. (Zugriff: 13.03.2023).

ECO Cycle™ (2023). GIKEN LTD. URL: https://www.giken.com/en/products/automated-parking-facilities/eco-cycle/ (besucht am 14. 03. 2023).

Eine Einführung in Software-Pakete – Definitionen (27. Jan. 2023). URL: https://datei.wiki/definition/eine-einfuehrung-in-software-pakete/ (besucht am 09. 03. 2023).

Expo (2023). Expo. URL: https://expo.dev/ (besucht am 26. 02. 2023).

Fahrradständer für 5 Fahrräder Rostfreier Stahl und Verzinkter Stahl (o. D.). URL: https://www.verkehrsshop.de/fahrradstaender-5-fahrraeder-l-foermiger-boden.html. (Zugriff: 14.03.2023).

Flutter - Build apps for any screen (2023). URL: //flutter.dev/ (besucht am 20. 03. 2023).

Frameworks für App und Website Entwicklung | itPortal24 (2023). URL: https://www.itportal24.de/ratgeber/framework-fuer-app-und-website-entwicklung (besucht am 20. 03. 2023).

Frankenhauser, Diplomingenieur (FH) Bernd (2019a). „Projekt-Funktionendiagramm PM-Prozesse". Unveröffentlichte Unterrichtsunterlagen.

— (2019b). „Projektbeauftragung PM-Prozesse". Unveröffentlichte Unterrichtsunterlagen.

— (2019c). „Projekte und Projektmanagement". Unveröffentlichte Unterrichtsunterlagen.

— (2019d). „Projekte und Projektmanagement PPM1 - Kanban Vorgehensmodell". Unveröffentlichte Unterrichtsunterlagen.

— (2019e). „Projektorganisation und Rollen". Unveröffentlichte Unterrichtsunterlagen.

— (2019f). „Projektumweltanalyse PUA". Unveröffentlichte Unterrichtsunterlagen.

Ginsberg, Nina (2020). *ECO-Cycle: Automated bike parking in Japan.* URL: https://www.bicyclescreatechange.com/eco-cycle-automated-bike-parking-in-japan/ (besucht am 09. 03. 2023).

GitHub Logos and Usage (2023). URL: https://github.com/logos (besucht am 13. 03. 2023).

GmbH, LTW Intralogistics (2023a). „Spielzeiten Fahrradturm 15m Höhe". Unveröffentliche Simulationsergebnisse.

— (2023b). „Spielzeiten Fahrradturm 20m Rondell". Unveröffentliche Simulationsergebnisse.

GmbH, Maxon Computer (2011). *C4D Logo.* URL: https://en.wikipedia.org/wiki/File:C4D_Logo.png (besucht am 04. 03. 2023).

GmbH, Orion Bausysteme (Sep. 2022). „Angebot". Unveröffentlichtes Angebot.

GmbH, WÖHR Autoparksysteme (2022). „RICHTPREISANGEBOT Wöhr Bikesfae". Unveröffentlichtes Angebot.

Grundlagen (18. Juni 2022). *Grundlagen: Javascript programmieren.* URL: https://www.mediaevent.de/javascript/ (besucht am 29. 03. 2023).

Hałun, CC Jakub (2005). *Bicycles parked near the Amsterdam Centraal railway station.* URL: https://commons.wikimedia.org/wiki/File:20050402_Amsterdam_bicycles_0563.jpg (besucht am 02. 03. 2023).

Historie: Entwicklungen der - WÖHR Autoparksysteme GmbH (2023). URL: https://woehr.de/de/historie.html (besucht am 07. 03. 2023).

Hoch hinaus und sicher verstaut | Der V-Locker Veloturm und seine Steuerung (o. D.). URL: https://www.technik-und-wissen.ch/der-v-locker-veloturm-und-seine-steuerung.html. (Zugriff: 13.03.2023).

How Large is a Parking Space? (o. D.). URL: https://franklinst.com/how-large-is-a-parking-space/. (Zugriff: 13.03.2023).

Inc., CC Formlabs (2019). *3D Printing Technologies for Plastic.* URL: https://www.flickr.com/photos/161389331@N04/31895725397 (besucht am 04. 03. 2023).

Inc., Google (2020). *Firebase Logo.* URL: https://de.m.wikipedia.org/wiki/Datei:Firebase_Logo.svg (besucht am 01. 03. 2023).

Jugend Innovativ (2023). URL: https://www.jugendinnovativ.at/wettbewerb/kategorien/kategorie-uebersicht (besucht am 14. 03. 2023).

Komponenten und Props – React (2023). URL: https://de.reactjs.org/docs/components-and-props.html (besucht am 06. 03. 2023).

Krutzler, David (2022). *Wien verbietet Abstellen von E-Scootern auf dem Gehsteig.* URL: https://www.derstandard.at/story/2000141204966/wien-verbietet-abstellen-von-e-scootern-auf-dem-gehsteig (besucht am 29. 01. 2023).

LaTeX (2023). URL: https://en.wikipedia.org/wiki/LaTeX (besucht am 14. 03. 2023).

LaTeX - A document preparation system (2023). URL: https://www.latex-project.org/ (besucht am 14. 03. 2023).

Leitfaden zum 3D-Druck mit selektivem Lasersintern (SLS) (2023). Formlabs. URL: https://formlabs.com/de/blog/einfuehrung-sls-3d-druck/ (besucht am 20. 03. 2023).

Lösungen (2023). URL: https://www.ltw.at/loesungen (besucht am 20. 03. 2023).

LTD, GIKEN (2023a). *ECO Cycle™.* URL: https://www.giken.com/en/products/automated-parking-facilities/eco-cycle/ (besucht am 09. 03. 2023).

— (2023b). *MOBILE ECO Cycle™.* URL: https://www.giken.com/en/products/automated-parking-facilities/mobile_eco_cycle/ (besucht am 09. 03. 2023).

LTW Intralogistics GmbH, Fitz Thomas (2023). „Fahrrad Turm Kalkulation“. Unveröffentliche Kalkulationen.

LTW Unternehmenswebsite (2023). URL: https://www.ltw.at/unternehmen (besucht am 14. 03. 2023).

Lüdecke, Karl (2023). *EU beschließt wichtigen Fahrradplan: Das soll sich auf den Straßen ändern.* URL: https://efahrer.chip.de/news/eu-beschliesst-wichtigen-fahrradplan-das-soll-sich-auf-den-strassen-aendern_1011605 (besucht am 08. 03. 2023).

mail@selfphp.de, Damir Enseleit (4. Sep. 2007). *Was sind Arrays?* URL: https://www.selfphp.de/praxisbuch/praxisbuchseite.php?site=155&group=28 (besucht am 25. 03. 2023).

Mair, Thomas (2022). *Land Vorarlberg - Presse - VMOBIL Radboxen ab jetzt am Bahnhof Altach.* URL: https://presse.vorarlberg.at/land/dist/vlk-65352.html (besucht am 10. 03. 2023).

Muller, CC Cliff (2017). *Bikes on a Balcony.* URL: https://www.flickr.com/photos/cliff_robin/32320042743/ (besucht am 02. 03. 2023).

NeoPixels (2023). URL: https://www.adafruit.com/category/168 (besucht am 14. 03. 2023).

New urban trend (2023). *New urban trend: underground cycle parking - news construction and development pose structures.* URL: https://undergroundexpert.info/issledovaniya-i-tehnologii/analitika/podzemnye-veloparkovki-novyj-trend/ (besucht am 14. 03. 2023).

Open Source Software einfach erklärt (2023). URL: https://www.plusserver.com/blog/open-source-software (besucht am 20. 03. 2023).

parkhere (26. Aug. 2020). *Versteckte Kosten für Mitarbeiterparkplätze*. ParkHere. URL: https://park-here.eu/ratgeber/kosten-mitarbeiterparkplaetze/ (besucht am 29. 03. 2023).

Peter, Fabio (2020a). *Der Veloparkplatz im Turm*. URL: https://v-locker.ch/wp-content/uploads/2020/10/181020-Berner-Zeitung_V-Locker.pdf (besucht am 06. 03. 2023).

— (Okt. 2020b). „Der Veloparkplatz im Turm". In: *Berner Zeitung*. (Zugriff: 13.03.2023). URL: https://v-locker.ch/wp-content/uploads/2020/10/181020-Berner-Zeitung_V-Locker.pdf.

Projektstrukturplan und Objektstrukturplan (11. Jan. 2015). Projektnachwuchs.de. URL: https://projektnachwuchs.de/projektstrukturplan-und-objektstrukturplan/ (besucht am 20. 03. 2023).

Raspberry Pi 4 Model B (2023). URL: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ (besucht am 28. 02. 2023).

Ravichandran, Adhithi (17. Juni 2022). *Building React Native Apps — Expo or not?* Medium. URL: https://adhithiravi.medium.com/building-react-native-apps-expo-or-not-d49770d1f5b8 (besucht am 26. 02. 2023).

React Native · Learn once, write anywhere (2022). URL: https://reactnative.dev/ (besucht am 05. 03. 2022).

React Native im Überblick (7. Sep. 2021). entwickler.de Deine Wissensplattform. Section: Artikel. URL: https://entwickler.de/react/react-native-im-uberblick (besucht am 29. 03. 2023).

RIS - Stellplatzverordnung - Landesrecht konsolidiert Vorarlberg, Fassung vom 09.03.2023 (2023). URL: https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=LrVbg&Gesetzesnummer=20000739 (besucht am 09. 03. 2023).

Risikomanagement im Projekt (2023). URL: https://www.gpm-blog.de/risikomanagement-im-projekt/ (besucht am 07. 03. 2023).

Ritchie, Hannah und Max Roser (13. Juni 2018). „Urbanization". In: *Our World in Data*. URL: https://ourworldindata.org/urbanization (besucht am 20. 03. 2023).

Rust (programming language) (2023). URL: https://en.wikipedia.org/wiki/Rust_(programming_language) (besucht am 13. 03. 2023).

Schwartze, Johannes (2022). „Automatisches Fahrradparkhaus | Technische Studie". Unveröffentlichtes Diplomarbeitsbriefing.

Software Development Kit (29. Dez. 2022). In: URL: https://de.wikipedia.org/w/index.php?title=Software_Development_Kit&oldid=229284894 (besucht am 25. 03. 2023).

Specifications of Eco Cycle (2023). GIKEN LTD. URL: https://www.giken.com/en/products/automated-parking-facilities/eco-cycle/specifications-of-eco-cycle/ (besucht am 14. 03. 2023).

Speed Comparison (2023). URL: https://niklas-heer.github.io/speed-comparison/ (besucht am 09. 03. 2023).

Stack Navigator | React Navigation (2022). URL: https://reactnavigation.org/docs/stack-navigator/ (besucht am 07. 03. 2022).

Stack Overflow Developer Survey (2023). URL: https://survey.stackoverflow.co/2022 (besucht am 28. 02. 2023).

Statista (2023). *Topic: Bicycle industry in Europe*. Statista. URL: https://www.statista.com/topics/3883/electric-bicycle-usage-in-europe/ (besucht am 20. 03. 2023).

Stereolithografie (SLA) 3D-Druck (2023). *Stereolithografie (SLA) 3D-Druck: Der ultimative Leitfaden*. Formlabs. URL: https://formlabs.com/de/blog/leitfaden-stereolithografie-sla-3d-druck/ (besucht am 20. 03. 2023).

Tab navigation | React Navigation (2022). URL: https://reactnavigation.org/docs/tab-based-navigation/ (besucht am 07. 03. 2022).

Über Jugend Innovativ- Jugend Innovativ (2023). URL: https://www.jugendinnovativ.at/wettbewerb/ueber-jugend-innovativ (besucht am 14. 03. 2023).

Unix Time Stamp - Epoch Converter (2023). URL: https://www.unixtimestamp.com/ (besucht am 25. 03. 2023).

Visual Studio Code (2023). URL: https://de.wikipedia.org/w/index.php?title=Visual_Studio_Code&oldid=230642960 (besucht am 13. 03. 2023).

Vor- und Nachteile von React Native für die Entwicklung mobiler Apps (2023). URL: https://www.cisin.com/coffee-break/de/technology/pros-and-cons-of-react-native-for-mobile-app-development.html (besucht am 30. 03. 2023).

, vorarlberg ORF at red vorarlberg (10. Feb. 2023). *Zug, Rad und Carsharing: Angebot soll wachsen*. vorarlberg.ORF.at. URL: https://vorarlberg.orf.at/stories/3194202/ (besucht am 29. 03. 2023).

Vorarlberger Landesregierung, Amt der (2020). *Fahrradparkierung im verdichteten Wohnbau - Leitfaden für Planungsbüros, Bauträger und Gemeinden*. URL: https://vorarlberg.at/-/leitfaden-fahrradparken (besucht am 05. 03. 2023).

Was ist GitHub? Eine Einführung in GitHub für Einsteiger (2020). URL: https://kinsta.com/de/wissensdatenbank/was-ist-github/ (besucht am 13. 03. 2023).

Weichert, Rune (2023). *Platz für 11.000 Fahrräder: Amsterdam eröffnet riesige Parkhäuser unter Wasser*. URL: https://www.stern.de/panorama/amsterdam-eroeffnet-riesige-fahrrad-parkhaeuser-unter-wasser-33138624.html (besucht am 10. 03. 2023).

Williams, Ramaksoud2000 via Chris (2011). *English: Unofficial JavaScript logo by Chris Williams, from GitHub logo.js, under very permissive licensing (WTFPL)*. URL: https://commons.wikimedia.org/wiki/File:JavaScript-logo.png (besucht am 29. 03. 2023).

WÖHR BIKESAFE (2023). WÖHR Autoparksysteme GmbH. URL: https://woehr.de/de/produkt/woehr-bikesafe.html (besucht am 14. 03. 2023).

WÖHR BIKESAFE (2023). WÖHR Autoparksysteme GmbH. URL: https://woehr.de/de/produkt/woehr-bikesafe.html (besucht am 14. 03. 2023).

WÖHR BIKESAFE - WÖHR Autoparksysteme GmbH - WÖHR Autoparksysteme GmbH (2023). URL: https://woehr.de/de/produkt/woehr-bikesafe.html (besucht am 07. 03. 2023).

WÖHR Bikesafe Broschüre (2022). URL: https://woehr.de/files/uploads/woehr/downloads/datasheets/cycle/de/W%C3%96HR2021_Bikesafe_Brosch%C3%BCre_08.pdf (besucht am 20. 09. 2022).

Xmind - Mind Mapping App (2023). URL: https://xmind.app/xmind8-pro/ (besucht am 08. 03. 2023).

Zsifkovits, Stephanie (3. Mai 2022). *Fast jedes zweite verkaufte Fahrrad 2021 war ein E-Bike.* UFH. URL: https://ufh.at/fast-jedes-zweite-verkaufte-fahrrad-2021-war-ein-e-bike/ (besucht am 20. 03. 2023).

13 Abbildungsverzeichnis

1 LTW Logo (vgl. *LTW Unternehmenswebsite* 2023) 11
2 Randvoller Fahrradabstellplatz in Amsterdam (vgl. Hałun 2005) 12
3 Fahrrad auf Balkon (vgl. Muller 2017) 13
4 Projektorganigramm 18
5 Projektstrukturplan 19
6 Objektstrukturplan des Prototypen 21
7 Risikoanalyse Matrix 22
8 Projektumweltanalyse 23
9 Schematische Darstellung eines Kanban-Boards 24
10 Zeitaufwand der Teammitglieder 27
11 Zeitaufwand der einzelnen Projekte 28
12 Fahrrad-Tiefgarage in Amsterdam (vgl. Weichert 2023) 30
13 Fahrradboxen in Vorarlberg (vgl. Mair 2022) 30
14 Plan vom Wöhr Bikesafe (vgl. *WÖHR Bikesafe Broschüre* 2022) 31
15 Eco Cycle (vgl. CNN 2016) 32
16 Funktionsweise vom V-Locker (vgl. AG 2023b) 32
17 Gegenüberstellung der Modelle 36
19 Rondell System mit Auslagerung von innen 42
20 Rondell System mit Auslagerung von außen 42
22 Konstruktionsgrößenanalyse 44
23 Hochregallager 2D Ansicht 45
24 Hochregallager mit doppelten Plätzen, 2D Ansicht 46
25 Hochregallager 3D Ansicht 46
27 Visualisierung des Modell-Turms 48
28 Mockup vom Screen Map 51
29 Mockup vom Screen Tower 52
30 Mockup vom Screen Aktivität 53
31 Mockup vom Screen Einstellungen 53
32 Mockup vom Screen Services 54
33 Switch-Selector in der App 57
34 Karte im Tab „Map“ 59
35 Karte im Tab „TowerInfo“ 59
36 BottomTab der App 61
37 Header der App mit Rücktaste 61

38 Navigations-Struktur der App 62
39 Komponente Box mit Fahrrad 63
40 Komponente Box mit Gegenstand 63
41 Komponente SettingLink 63
42 Komponente OtherLink 64
43 Komponente BoxImage als Fahrradbox 65
44 Komponente BoxImage als Box für Gegenstände 65
45 Screen Home 66
46 Screen BoxInfo 67
47 Screen Services 68
48 Screen Settings 69
49 Screen LogIn 70
50 Screen SignIn 72
51 Screen Map 73
52 Screen TowerInfo 74
53 Screen TowerInfo 75
54 Szenarien beim Screen Assignment 76
55 Senden der Anfrage zum Aus-/Einlagern 77
56 JobRecieved 77
57 JobSuccessful 78
58 Tower Controller Komponenten 79
59 Datenströme des Tower Controller V1 80
60 State Machine des Tower Controller V2 81
61 Intialisierung des Turm Controllers 84
62 Event Handling Pipeline 85
63 Ersatzschaltbild des Turms 90
64 Konstruktion des Turmsegmentes 91
67 ER-Diagramm der Datenbank 93
68 Kommunikation zwischen App und Turm 96
69 Visualisierung des Turms im Projektbericht 100
70 Einnahmeströme 101
71 Firebase Logo (vgl. G. Inc. 2020) 107
72 Raspberry PI 4b (vgl. *Raspberry Pi 4 Model B* 2023) 108
73 Rust Logo 109
74 React Naive Logo (vgl. *React Native · Learn once, write anywhere* 2022) 110
75 Javascript Logo (vgl. Williams 2011) 110
76 Expo Logo (vgl. *Expo* 2023) 111
77 Schematische Darstellung des FDM Verfahrens (vgl. C. F. Inc. 2019) 112
78 Schematische Darstellung des SLA Verfahrens (vgl. C. F. Inc. 2019) 113

79 Schematische Darstellung des SLS Verfahrens (vgl. C. F. Inc. 2019) 113
80 Cinema 4D Logo (vgl. M. C. GmbH 2011) 114
81 Visual Studio Code Logo (vgl. *Datei:Visual Studio Code 1.35 icon.svg – Wikipedia* 2019) 114
82 GitHub Logo (vgl. *GitHub Logos and Usage* 2023) 115

14 Codeverzeichnis

1 Verwendung der Methode getPreciseDistance 57
2 Verwendung der Methode getFocusedRouteNameFromRoute 60
3 Verwendung vom StackNavigator 60
4 Ausdruck für die Komponente Box 63
5 Ausdruck für die Komponente SettingLink 64
6 Ausdruck für die Komponente OtherLink 64
7 Ausdruck für die Komponente BoxImage 65
8 Methode zum Navigieren 67
9 Zugriff auf Variable Box 67
10 Funktion SignIn 70
11 Funktion createAccount 71
12 TowerDatabase Struktur 86
13 Tower Struktur 87
14 AssignmentScheduler Struktur 88
15 Schema für Tower Dokumente 94
16 Schema für Box Dokumente 94
17 Schema für Job Dokumente 94
18 Schema für User Dokumente 94
19 Schema für Rental Dokumente 95
20 Schema für Key Dokumente 95
21 Einlager Anfrage Assignment Struktur 97
22 Einlagerbestätigung Assignment Struktur 97
23 Auslager Anfrage Assignment Struktur 98
24 Auslagerbestätigung Assignment Struktur 98

15 Tabellenverzeichnis

1 Versionshistorie 2
2 Projektteam 9
3 Projektbetreuer 10
4 Meilensteine 15
5 Projektauftrag 17
6 Gantt Diagramm 20
7 Risikoanalyse 22
8 Projektumweltanalyse (Tabellarische Form) 23
9 Projektfunktionendiagramm 26
10 Arbeitsaufwand der Teammitglieder 27
11 Arbeitsaufwand der einzelnen Projekte 28
12 Quantitative Kriterien 34
13 Qualitative Kriterien 35
14 Gegenüberstellung Rust und Python 83
15 Einnahmeströme 101
16 Übersicht Modelle 102
17 Absatzzahlen - BikeTower GmbH 102
18 Absatzzahlen - Andere 102
19 Stellplätze Zuwachs - BikeTower GmbH 102
20 Stellplätze Zuwachs - Andere 103
21 Stellplätze akkumuliert - Jahresbestand 103
22 Investitionssummen - BikeTower GmbH 103
23 Kostenrechnung 104
24 Finanzplan 105
25 Autorenverzeichnis 133

16 Abkürzungsverzeichnis

RBG Regalbediengerät 45

API Programmierschnittstelle 85
Application Programming Interface

SBC Einplatinencomputer 108
Single Board Computer. Ein Computer auf einer einzelnen Platine.

GPIO General Purpose Input Output 80

SPI Serial Peripheral Interface 108

I2C Inter-Integrated Circuit 108
On-Board-Kommunikationsprotokoll für Peripheriegeräte. Gut für kurze Distanzen und niedrige Übertragungsraten geeignet

RAM Arbeitsspeicher 108
Random Access Memory

Arm Acorn RISC Machine 108
CPU Architektur

IoT Internet of Things 108

SDK Software-Entwicklungskit 61
Ein Software Entwicklungskit(Software Development Kit) ist eine Sammlung von Programmierwerkzeugen und Programmbibliotheken, die zur Entwicklung von Software dient (vgl. *Software Development Kit* 2022)

CAD Konstruktionsprogram 17
Computer Aided Design

CAM computergestützte Fertigung 111
Computer Aided Manufacturing

IDE Entwicklungsumgebung 17
Integrated Development Environment

FDM Fused Deposition Modeling 112

SLA Stereolithographie 7
Stereolithography

SLS Selektives Laser Sintern 7
Selective Laser Sintering Interprocess Communication

Stdin Standardeingabe 80
Standard Input. Die Standardeingabe ist die Eingabe, die ein Programm von der Tastatur oder einer Datei erhält.

Stdout Standardausgabe 80
Standard Output. Die Standardausgabe ist die Ausgabe, die ein Programm auf dem Bildschirm oder in einer Datei ausgibt. Meist mit dem Monitor bzw. Terminal verbunden.

FIFO First In First Out 132

GUI grafische Benutzeroberfläche 80
Graphical User Interface Command Line Interface

RAII Ressourcenverwaltung durch Initialisierung 109
Resource Acquisition Is Initialization. Ein Konzept, welches es ermöglicht, Ressourcen automatisch zu verwalten.

CRUD Erstellen, Lesen, Aktualisieren, Löschen 109
Create, Read, Update, Delete. CRUD ist ein Akronym für die vier grundlegenden Funktionen, die in relationalen Datenbanken verwendet werden.

ER Entity Relation 93

17 Glossar

Array Ein Array ist ein Datentyp, in dem beliebig viele Werte abgespeichert werden können. (vgl. mail@selfphp.de 2007). 63, 66, 67

BottomTabBar Die BottomTabBar ist eine Navigationsleiste, die sich am unteren Ende des Bildschirms befindet. 61, 62

crates.io crates.io ist eine Plattform, auf der Rust-Pakete veröffentlicht werden können.. 109

Daisy-Chaining Daisy-Chaining bedeutet, dass mehrere Geräte aneinander geschaltet werden.. 108

debuggen Debuggen ist das Finden von Fehlern in einem Programm wärend es ausgeführt wird. 80

Deserialisierung Deserialisierung ist ein Prozess, bei dem eine Zeichenfolge in ein Objekt umgewandelt wird.. 7, 93

event Ein Event ist ein Ereignis, welches in einem Programm auftritt. 80, 82

Fahrraddurchsatz Der Fahrraddurchsatz ist die Anzahl der Fahrräder, die ein Lagersystem pro Stunde Ein- und Auslagert.. 34, 44

Fork Ein Fork ist eine Kopie eines Git-Repositories, die auf dem eigenen GitHub-Account erstellt wird.. 110

Framework „*Ein Framework ist eine Ansammlung von Werkzeugen, vorgefertigten Features in Form von Code-Schnipseln und Regeln, welche dabei helfen sollen, Softwareprojekte schneller umzusetzen.*“ (vgl. *Frameworks für App und Website Entwicklung | itPortal24* 2023). 49, 50, 56, 59, 110, 111

high-level High-level ist ein Begriff, der angibt, dass ein Programm in einer Programmiersprache geschrieben, die für Menschen leicht verständlich ist und stark abstrahiert.. 109

interpretiert Interpretiert ist ein Begriff, der angibt, dass ein Programm nicht in eine Maschinensprache übersetzt wurde, sondern von einer Interpreter-Software ausgeführt wird (Bsp. Java).. 83

kernel Ein Kernel ist ein Programm, das die Hardware eines Computers verwaltet.. 108

kompiliert Kompiliert ist ein Begriff, der angibt, dass ein Programm in eine Maschinensprache übersetzt wurde (Bsp. C).. 83, 109

listener Ein Listener ist ein Programm, welches auf Ereignisse wartet und anhand dieser Ereignisse eine Aktion ausführt. 80, 84, 96, 109

Location-Pin Ein Location-Pin ist ein Symbol, das den Standort eines Ortes oder Objekts auf einer Karte markiert.. 58

master-branch Der Master-Branch ist der Haupt-Zweig eines Git-Repositories, also der Zweig, welcher normalerweise von den Entwicklern verwendet wird. 110

Memory-Safety Memory-Safety ist ein Begriff, der angibt, dass ein Programm keine Speicherzugriffsfehler hat (SEGFAULT).. 109

Mockup Ein Mockup ist ein digital gestalteter Entwurf von einer App. 51–54, 123

MongoDB MongoDB ist eine NoSQL-Datenbank, die Dokumentenorientiert ist.. 93

nesting Nesting ist ein Begriff, der angibt, dass etwas in etwas anderes eingebettet ist.. 93

NoSQL NoSQL ist ein Datenbankmodell, das es ermöglicht, Daten in einer flexiblen Struktur zu speichern.. 93

Open-Source Open-Source ist ein Software-Entwicklungsmodell, bei dem der Quellcode öffentlich ist und von jeder Person gratis weiterentwickelt und genutzt werden kann.*Open Source Software einfach erklärt* 2023. 49, 109–111

Pin Ein Pin ist ein elektrischer Kontakt, der mit einem Draht verbunden ist. Ein Raspberry Pi hat z.B. 40 Pins.. 108

pull-request Ein Pull-Request ist eine Änderungsvorschlag, welcher von einem Entwickler an einen anderen Entwickler gestellt wird, um Änderungen an einem Git-Repository zu übernehmen. 110

quality-of-life Quality-of-life ist ein Begriff, der angibt, dass ein Programm die Arbeit erleichtert.. 115

Query Eine Query ist ein Befehl, der es ermöglicht, Daten aus einer Datenbank zu lesen. So können beispielsweise alle Dokumente mit einem bestimmten Wert in einem bestimmten Feld ausgelesen werden.. 109

queue Eine Queue ist eine Datenstruktur, welche Daten in einer bestimmten Reihenfolge speichert. Eine queue fuktioniert nach einem First In First Out (FIFO) Prinzip. 80

race condition Eine Race Condition ist ein Fehler, welcher auftritt, wenn zwei oder mehrere Threads auf eine Ressource zugreifen und diese Ressource nicht synchronisiert ist. 88

relational In einer relationalen Datenbank werden die Daten in Tabellen gespeichert. Und deren Beziehungen werden definiert.. 93

Serialisierung Serialisierung ist ein Prozess, bei dem ein Objekt in eine Zeichenfolge umgewandelt wird, die es ermöglicht, das Objekt zu speichern oder zu übertragen.. 7, 93

sleep Sleep ist eine Funktion, welche das Programm für eine bestimmte oder unbestimmte Zeit anhält. 84

StackNavigator Ein StackNavigator ist eine Art der Navigation in einer App. Dabei werden die verschieden Ansichten aufeinander „gestapelt und die anwendende Person kann vor und zurück navigieren.“ (vgl. *Stack Navigator | React Navigation* 2022). 61, 62

StackNavigator Ein TabNavigator ist eine Art der Navigation in einer App. Dadurch kann die anwendende Person zwischen verschiedenen Abschnitten in der App wechseln (vgl. *Tab navigation | React Navigation* 2022). 61, 62

state machine Eine State Machine ist ein Programm, welches immer in einem von mehreren bestimmten Zuständen ist. Jeder Zustand kann auf ein Ereignis reagieren und einen anderen Zustand auslösen. 81, 84

Switch-Selector Ein Switch-Selector ist ein interaktives Element in einer App, das es der anwendenden Person ermöglicht, zwischen verschiedenen Optionen oder Zuständen zu wechseln. 57

Tab Ansicht in der App. 51–54

thread Ein Thread ist ein Teil eines Programms, welches unabhängig von anderen Teilen des Programms ausgeführt wird. 80, 84, 87

Unix-Timestamp Ein Unix-Timestamp ist eine Art der Zeitdarstellung, die die Anzahl der Sekunden seit dem 1. Januar 1970 um 00:00:00 Uhr UTC angibt und häufig in Datenbanken verwendet wird.(vgl. *Unix Time Stamp - Epoch Converter* 2023). 57

wrapper Ein Wrapper ist ein Programm, welches eine andere Software in eine andere Programmiersprache übersetzt. 86, 87

18 Autorenverzeichnis

Stand 21.03.2023

Kapitel	**Autor**
Impressum	Paul Hartmann
LTW	Paul Hartmann
Einleitung	Joshua Lung
Projektmanagement	Joshua Lung
Marktanalyse	
Existierende Modelle	Paul Hartmann
Nutzeranalyse	Joshua Lung
Nutzungstiefe	Paul Hartmann
Kundensegmente	Paul Hartmann
Rahmenbedingungen	Paul Hartmann
Umfrage	Paul Hartmann
Produktkonzeption	Paul Hartmann
	Joshua Lung
Konzept des Prototypen	
Turm	Joshua Lung
App	Paul Hartmann
Datenbank	Paul Hartmann
Umsetzung des Prototypen	
Turm	Joshua Lung
App	Paul Hartmann
Kommunikation	Joshua Lung
Datenbank	Joshua Lung
Marktskalierung	Paul Hartmann
	Joshua Lung
Verwendete Technologien	Paul Hartmann
	Joshua Lung

Tabelle 25: Autorenverzeichnis

19 Anhang

- Richtwerte für Betriebskosten von RBG - gültig ab 2018
- Kostenaufstellung
- Konstruktionszeichnung Fahrradlager 12m
- Konstruktionszeichnung Fahrradlager 15m
- Konstruktionszeichnung Fahrradlager 20m
- Spielzeiten Fahrradturm 12m
- Spielzeiten Fahrradturm 15m
- Spielzeiten Fahrradturm 20m
- Angebot Fahrradbox von Orion Bausysteme GmbH
- Amgebot Rondell von Wöhr Autoparksysteme GmbH

Richtwerte für Betriebskosten von RBG - gültig ab 2018

Die Betriebskosten setzen sich zusammen aus

1 Wartungskosten

Kosten pro Wartung pro RBG:

Automatisches RBG mit einer (1) Teleskopgabel (TG)	EUR	[redacted]
Automatisches RBG mit zwei (2) TG oder einem (1) Shuttle	EUR	[redacted]

Die Kosten verstehen sich exkl. Überstundenzuschlägen, Reisekosten und Nächtigungskosten.

Wartungsintervalle:

1-Schicht-Betrieb	ca. 1 bis 2 x pro Jahr
2-Schicht-Betrieb	ca. 2 bis 3 x pro Jahr
3-Schicht-Betrieb	ca. 3 bis 4 x pro Jahr

2 Kosten für Ersatz- und Verschleißteile

Es werden grobe Erfahrungswerte angeführt. Die Kosten für Ersatz- und Verschleißteile steigen mit zunehmenden Betriebsstunden der RBG an.

Richtwert pro RBG pro Jahr:

1-Schicht-Betrieb	ca. [redacted]% des RBG-Verkaufspreises
2-Schicht-Betrieb	ca. [redacted]% des RBG-Verkaufspreises
3-Schicht-Betrieb	ca. [redacted]% des RBG-Verkaufspreises

3 Energiekosten

Der Energieverbrauch bzw. die Energiekosten hängen stark von den Leistungsdaten und der Einschaltdauer ab. Es stehen diverse ECO-Funktionen zur Auswahl - siehe separate Dokumente.

Kostenaufstellung

LTW Lösung 15m Höhe

Länge	16	m
Breite	7.32	m
Höhe	14.86	m

LTW-System						
Benennung	**STk.**	**m²**	**Stückpreis**	**€/m²**		
Transportbox	**212**		[redacted] €			[redacted] €
Regal						
Regalbediengerät	1		[redacted] €			[redacted] €
Regalanlage	212		[redacted] €			[redacted] €
Verkleidung						
Stahlkonstruktion für Fassade und Dach, 9 Steher 15m Hoch	9		[redacted] €			[redacted] €
Wandpaneel		453		[redacted] €		[redacted] €
Dachpaneel		116		[redacted] €		[redacted] €
Bodenplatte		116		[redacted] €		[redacted] €
Software						
Programmierung ca. 300 Std.	1		[redacted] €			[redacted] €
E-Material und Installation	1		[redacted] €			[redacted] €
App Programmierung	1		[redacted] €			[redacted] €
Gesamt Herstellkosten						[redacted]
Verkaufspreis (+Deckungsbeitrag [redacted] %)						**1 004 611.89 €**
Kosten/Stellplatz						**4 738.74 €**

LTW Lösung 12m Höhe

Länge	11	m
Breite	7.538	m
Höhe	12.1	m

LTW-System					
Benennung	**STk.**	**m²**	**Stückpreis**	**€/m²**	
Transportbox	**104**		[redacted] €		[redacted] €
Regal					
Regalbediengerät	1		[redacted] €		[redacted] €
Regalanlage	104		[redacted] €		[redacted] €
Verkleidung					
Stahlkonstruktion für Fassade und Dach, 8 Steher 15m Hoch	8		[redacted] €		[redacted] €
Wandpaneel		321		[redacted] €	[redacted] €
Dachpaneel		86		[redacted] €	[redacted] €
Bodenplatte		86		[redacted] €	[redacted] €
Software					
Programmierung ca. 300 Std.	1		[redacted] €		[redacted] €
E-Material und Installation	1		[redacted] €		[redacted] €
App Programmierung	1		[redacted] €		[redacted] €
Gesamt Herstellkosten					[redacted] €
Verkaufspreis (+Deckungsbeitrag [redacted] %)					**825 855.45 €**
Kosten/Stellplatz					**7 940.92 €**

LTW-Lösung 500 Plätze

Länge	23.541	m
Breite	7.538	m
Höhe	20.38	m

LTW-System					
Benennung	**STk.**	**m^2**	**Stückpreis**	**€/m^2**	
Transportbox	**494**		[redacted] €		[redacted] €
Regal					
Regalbediengerät	1		[redacted] €		[redacted] €
Regalanlage	494		[redacted] €		[redacted] €
Verkleidung					
Stahlkonstruktion für Fassade und Dach, 11 Steher 20m Hoch	11		[redacted] €		[redacted] €
Wandpaneel		787		[redacted] €	[redacted] €
Dachpaneel		177		[redacted] €	[redacted] €
Bodenplatte		177		[redacted] €	[redacted] €
Software					
Programmierung ca. 300 Std.	1		[redacted] €		[redacted] €
E-Material und Installation	1		[redacted] €		[redacted] €
App Programmierung	1		[redacted] €		[redacted] €
Gesamt Herstellkosten					[redacted] €
Verkaufspreis (+Deckungsbeitrag [redacted] %)					**1 426 804.60 €**
Kosten/Stellplatz					**2 888.27 €**

Rondel 40m Höhe

Länge Lift	2.2894	m
Breite Lift	1.89739	m
Durchmesser Rondel	4.04252	m
Höhe	40	m

Rondel

Benennung	STk.	m^2	Stückpreis	$€/m^2$	
Transportbox	**208**		[redacted] €		[redacted] €
Regal					- €
Vertikalförderer mechanisch	4		[redacted] €		[redacted] €
5 Rondelsegment mit Drehkranz, 5 Ebenen, Antrieb	5		[redacted] €		[redacted] €
Verkleidung					- €
Stahlkonstruktion für Fassade und Dach, 16 Steher, 40 m Hoch	16		[redacted] €		[redacted] €
Wandpaneel		1290		[redacted] €	[redacted] €
Dachpaneel		43		[redacted] €	[redacted] €
Bodenplatte		43		[redacted] €	[redacted] €
Software					
Programmierung, ca. 380 Std.	1		[redacted] €		[redacted] €
E-Material und Installation	1		[redacted] €		[redacted] €
App Programmierung	1		[redacted] €		[redacted] €
Gesamt Herstellkosten					[redacted] €

Verkaufspreis (+Deckungsbeitrag [redacted] %)	**2 076 958.54 €**
Kosten/Stellplatz	**9 985.38 €**

Rondel 20m Höhe

Länge Lift	2.2894	m
Breite Lift	1.89739	m
Durchmesser Rondel	4.04252	m
Höhe	20	m

Rondel

Benennung	STk.	m^2	Stückpreis	$€/m^2$	
Transportbox	**104**		[redacted] €		[redacted] €
Regal					- €
Vertikalförderer mechanisch	4		[redacted] €		[redacted] €
2 Rondelsegment mit Drehkranz, 6/7 Ebenen, Antrieb	2		[redacted] €		[redacted] €
Verkleidung					- €
Stahlkonstruktion für Fassade und Dach, 16 Steher, 20 m Hoch	16		[redacted] €		[redacted] €
Wandpaneel		645		[redacted] €	[redacted] €
Dachpaneel		43		[redacted] €	[redacted] €
Bodenplatte		43		[redacted] €	[redacted] €
Software					
Programmierung, ca. 380 Std.	1		[redacted] €		[redacted] €
E-Material und Installation	1		[redacted] €		[redacted] €
App Programmierung	1		[redacted] €		[redacted] €
Gesamt Herstellkosten					[redacted] €

Verkaufspreis (+Deckungsbeitrag [redacted] %)	**1 460 700.02 €**
Kosten/Stellplatz	**14 045.19 €**

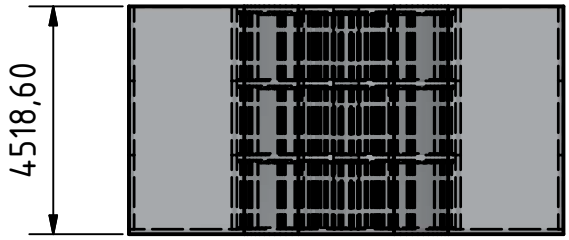
Rondel 3 Etagen

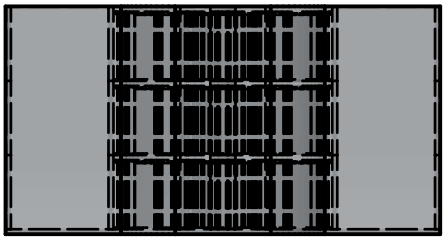
Länge Lift	2.2894	m
Breite Lift	1.89739	m
Durchmesser Rondel	4.04252	m
Höhe	4.5	m

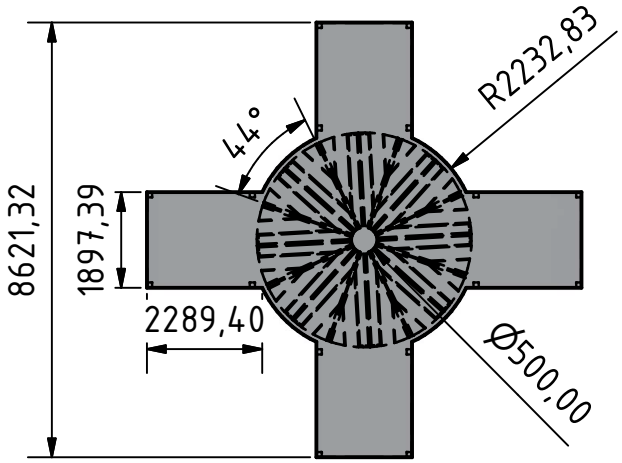
Rondel

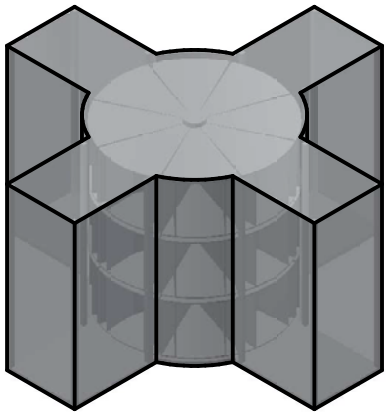
Benennung	STk.	m^2	Stückpreis	$€/m^2$	
Transportbox	**24**		[redacted] €		[redacted] €
Regal					- €
Vertikalförderer mechanisch	4		[redacted] €		[redacted] €
1 Rondelsegment mit Drehkranz, 3 Ebenen, Antrieb	1		[redacted] €		[redacted] €
Verkleidung					- €
Stahlkonstruktion für Fassade und Dach, 16 Steher, 4,5 m Hoch	16		[redacted] €		[redacted] €
Wandpaneel		145		[redacted] €	[redacted] €
Dachpaneel		43		[redacted] €	[redacted] €
Bodenplatte		43		[redacted] €	[redacted] €
Software					
Programmierung, ca. 380 Std.	1		[redacted] €		[redacted] €
E-Material und Installation	1		[redacted] €		[redacted] €
App Programmierung	1		[redacted] €		[redacted] €
Gesamt Herstellkosten					[redacted] €

Verkaufspreis (+Deckungsbeitrag [redacted] %)	**1 031 467.60 €**
Kosten/Stellplatz	**42 977.82 €**









Allgemeintoleranz nach	Maßstab	Werkstoff	Oberfläche	Werkstückkanten nach
Länge/Winkel DIN ISO 2768-1 — Form/Lage DIN ISO 2768-2	1 : 150	Rohteil/Halbzeug	nach ISO 1302	DIN ISO 13715

	Datum	Name
Gezeichnet	03.12.2022	jo_ki
Kontrolliert		
Freigegeben		

Projekt

Bezeichnung

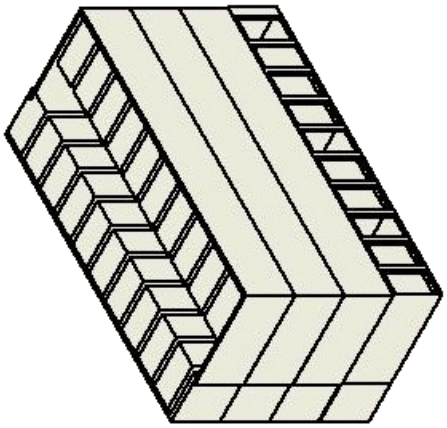
HTL DORNBIRN

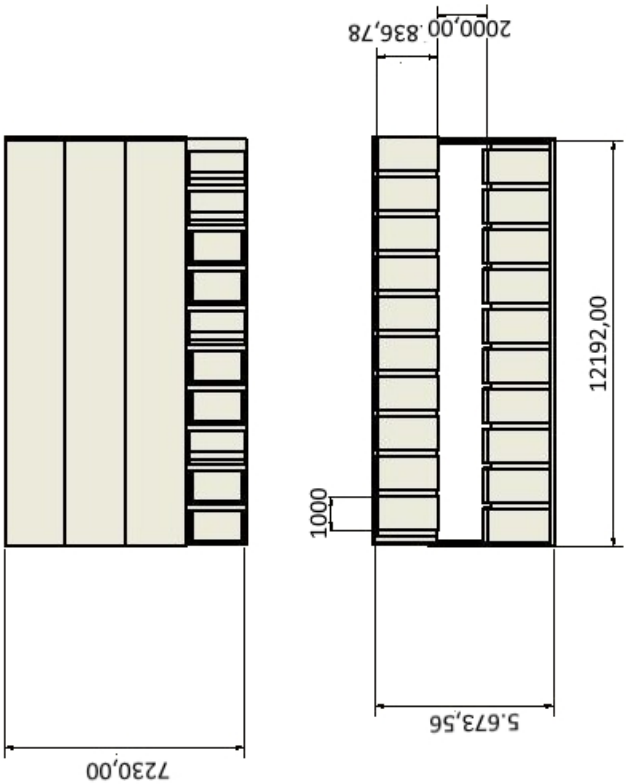
Bauteilnummer: rondel_bemaßung

Blatt-Nr.: 1

Projektionsmethode

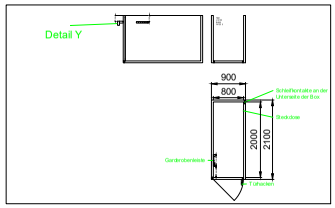
Änderungen	Datum	Name




7230,00
5.673,56
1000
12192,00
2000,00
836,78

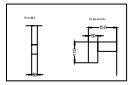
Transport Einhet

Maßstab 1:50



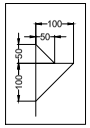
Verbindung der Türen Detail Y

Maßstab 1:10



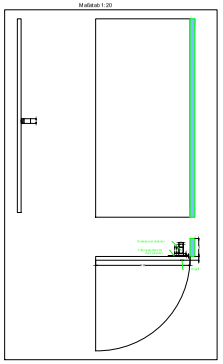
Palettenablage im Lager Detail X

Maßstab 1:5



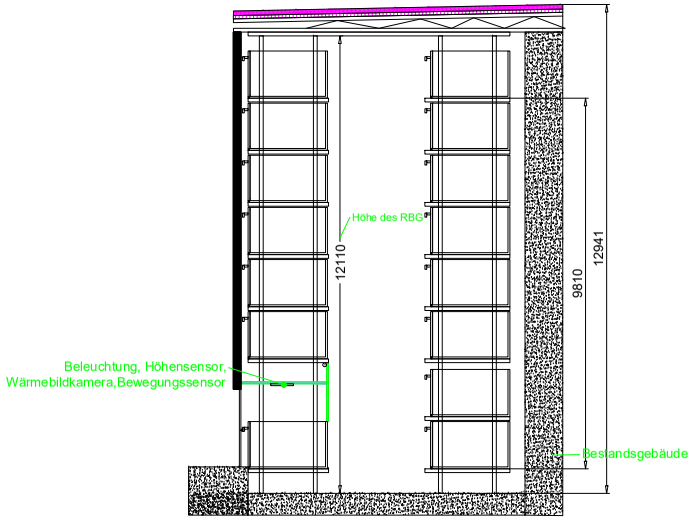
Außentür

Maßstab 1:20



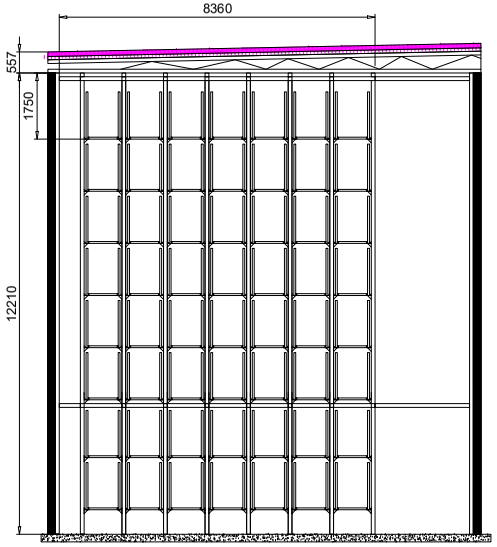
Schnitt B-B

Maßstab 1:50



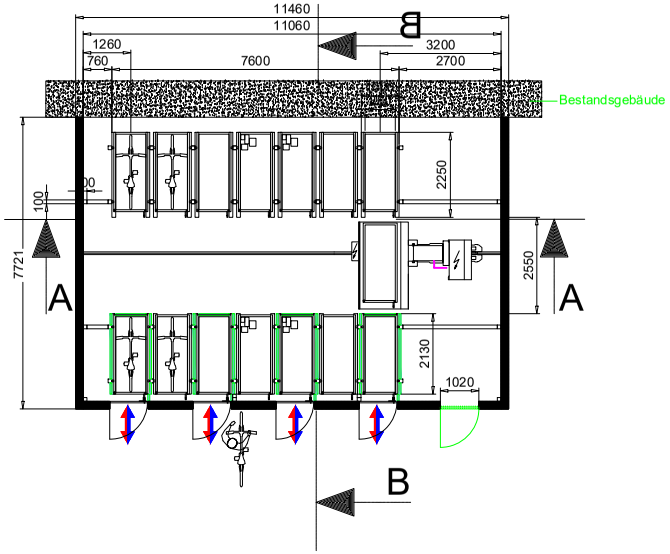
Schnitt A-A

Maßstab 1:50



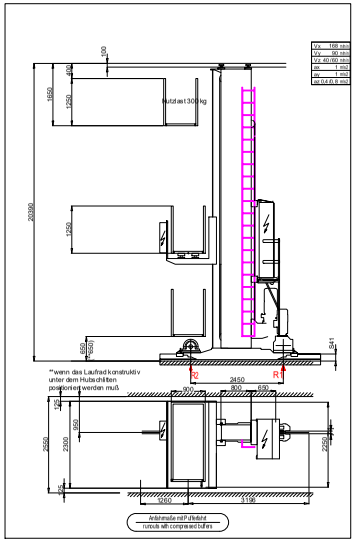
Grundriss

Maßstab 1:50



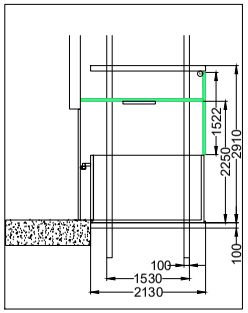
Regalbediengerät

Maßstab 1:50



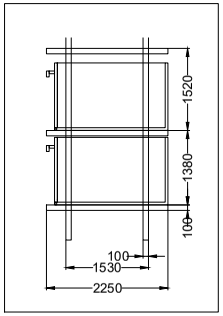
Boxen-Umschlagsplatz-Kreuzriss

Maßstab 1:36



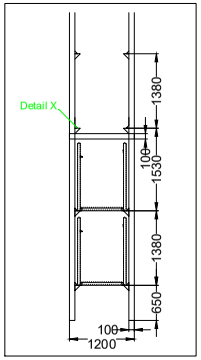
Lagerplatz-Rückseite-Kreuzriss

Maßstab 1:36



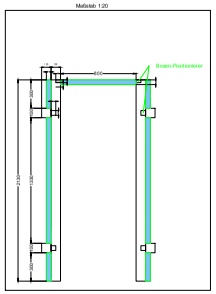
Lagerplatz-Aufriss

Maßstab 1:36



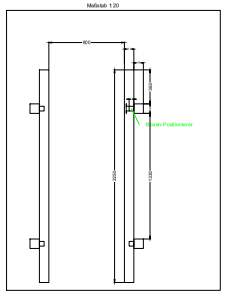
Boxen-Umschlagsplatz-Grundriss

Maßstab 1:20



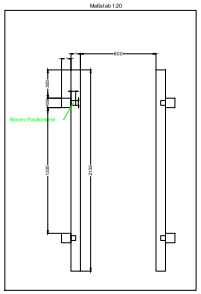
Lagerplatz-Rückseite-Grundriss

Maßstab 1:20



Lagerplatz-Vorderseite-Grundriss

Maßstab 1:20



Lagerkapazität brutto:
Transporteinheit: Metallbox: 900x2100
Gasse 1

Anzahl Palettenplätze in X-Richtung:	7	PP
Anzahl Palettenplätze in Y-Richtung:	8	PP
Anzahl Palettenplätze in Z-Richtung:	2	PP
Zwischensumme	112	PP
Verluste durch Einlagerungsplätze:	8	PP
Gesamt	ca. 104	PP

VORABZUG - UNVERBINDLICH !
PRELIMINARY - NOT BINDING !

Index	Änderung	Datum	Name / Schema	Ausführung	gezeichnet / Datum / Name
			0	1/50 (Faktor 0,707)	20.03.2023 SONJ

Auftrag: Fahrradlager
Aufbau: Fahrradlager
Block: - X
Layout Fahrradlager

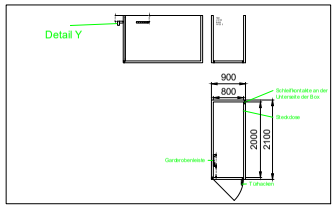
Maßstab: 1/50
Gewicht: kg
Freiformat:
Dateiangabe Nr.:
Blatt Nr.:
Zeichnungsnr.: Fahrradlager_v01
Index:

This drawing is intellectual property of LTW Intralogistics GmbH, Achdorf, Austria. This drawing is confidential and may not be copied, nor shown to third parties without permission.

A0

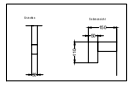
Transport Einhet

Maßstab 1:50



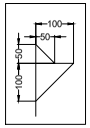
Verbindung der Türen Detail Y

Maßstab 1:10



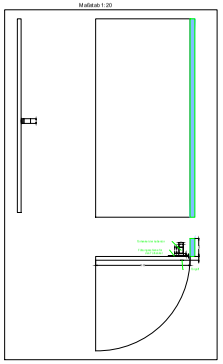
Palettenablage im Lager Detail X

Maßstab 1:5



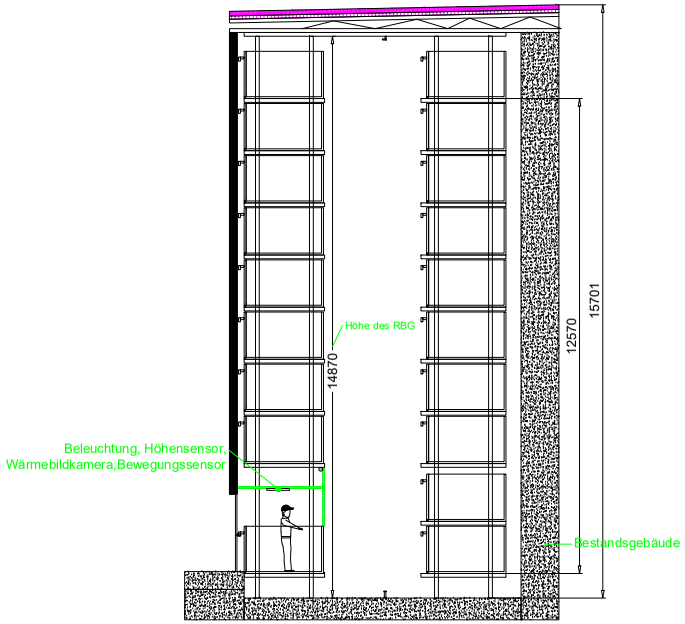
Außentür

Maßstab 1:20



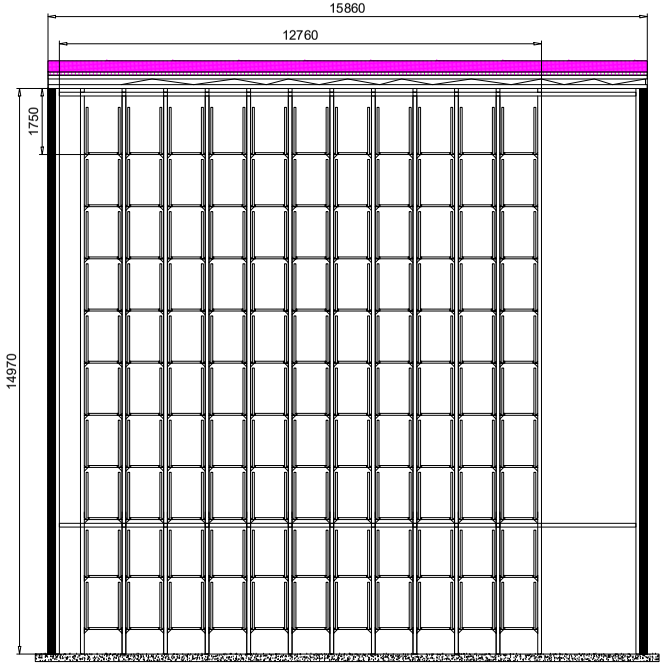
Schnitt B-B

Maßstab 1:50



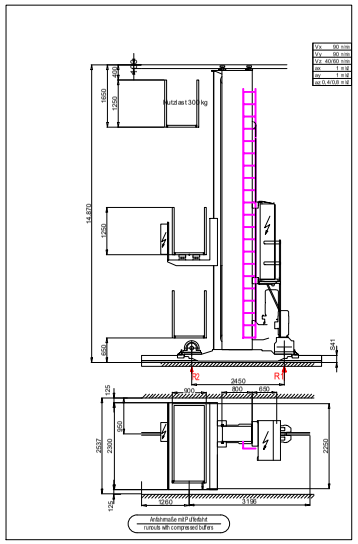
Schnitt A-A

Maßstab 1:50



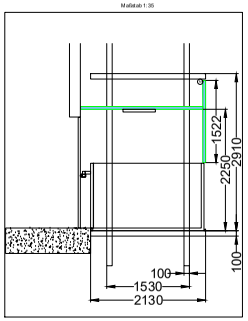
Regalbediengerät

Maßstab 1:50



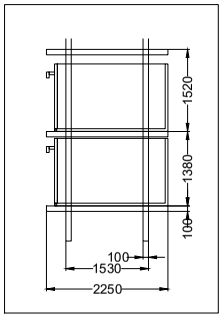
Boxen-Umschlagsplatz-Kreuzriss

Maßstab 1:36



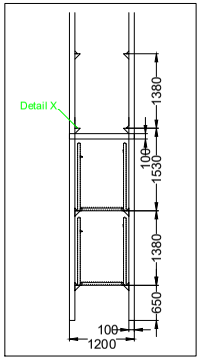
Lagerplatz-Rückseite-Kreuzriss

Maßstab 1:36



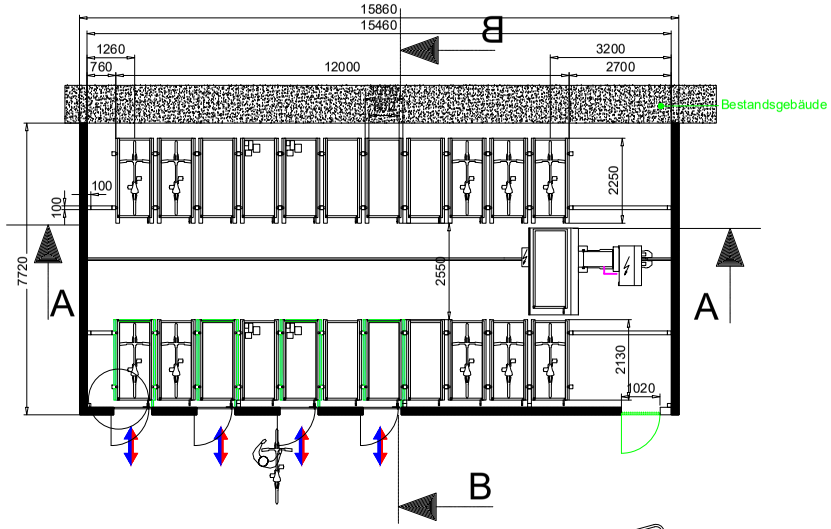
Lagerplatz-Aufriss

Maßstab 1:36



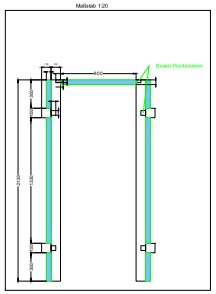
Grundriss

Maßstab 1:50



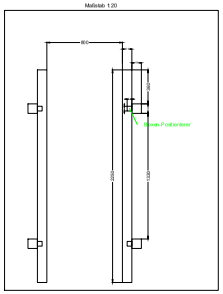
Boxen-Umschlagsplatz-Grundriss

Maßstab 1:20



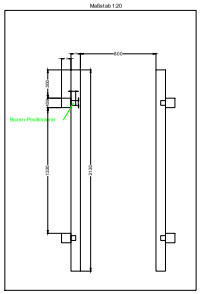
Lagerplatz-Rückseite-Grundriss

Maßstab 1:20



Lagerplatz-Vorderseite-Grundriss

Maßstab 1:20



Lagerkapazität brutto:
Transporteinheit: Metallbox: 900x2100
Gasse 1

Anzahl Palettenplätze in X-Richtung:	11	PP
Anzahl Palettenplätze in Y-Richtung:	10	PP
Anzahl Palettenplätze in Z-Richtung:	2	PP
Zwischensumme	220	PP
Verluste durch Einlagerungsplätze:	8	PP
Gesamt	ca. 212	PP

VORABZUG - UNVERBINDLICH !
PRELIMINARY - NOT BINDING !

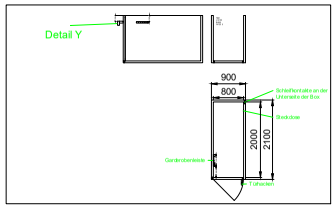
Index	Änderung	Datum	Name / Schreiber	Ausführung	gezeichnet / Stadt No.
			0	1/50	Datum 20.02.2023 Name SONU
Auftrag	Fahrradlager				
Auftrag	Fahrradlager	Block - X			
Layout Fahrradlager					
				Zeichnungs-Nr.	Stadt No.
				Zeichnungsdatei	Index
				Fahrradlager_v01	

This drawing is intellectual property of LTW Intralogistics GmbH, Achtal-strasse 33, E 8322 Gnaftach / Austria
This drawing or / refer material is our prop erty and must not be copied nor et << ed or shown to p ad without permission.

A0

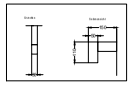
Transport Einhet

Maßstab 1:50



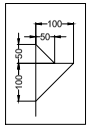
Verbindung der Türen Detail Y

Maßstab 1:10



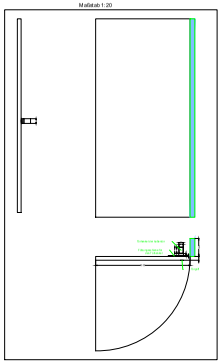
Palettenablage im Lager Detail X

Maßstab 1:5



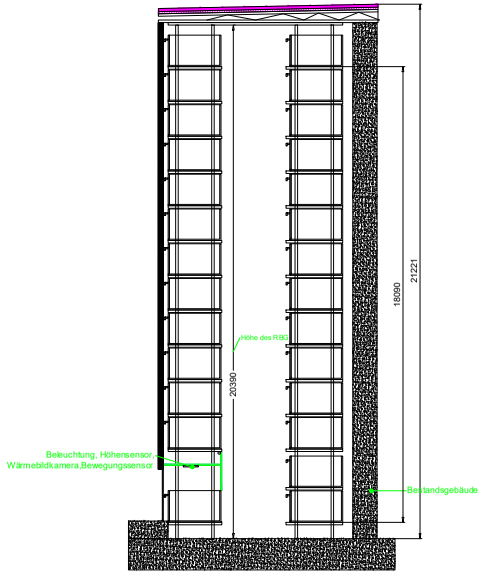
Außentür

Maßstab 1:20



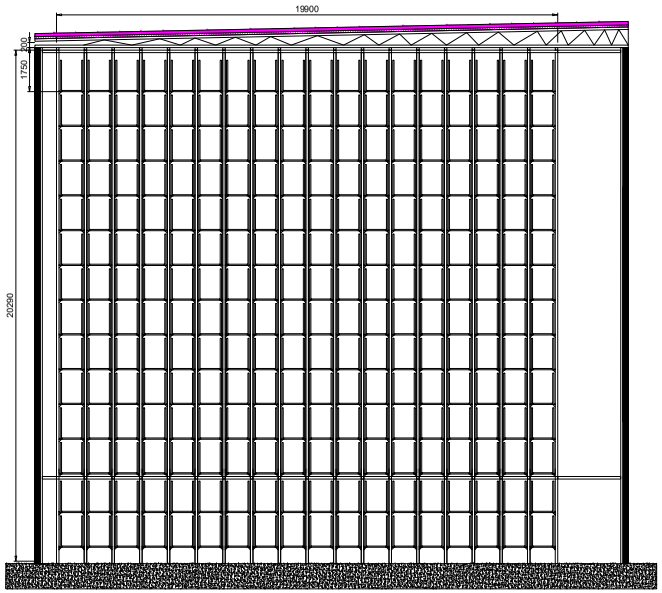
Schnitt B-B

Maßstab 1:75



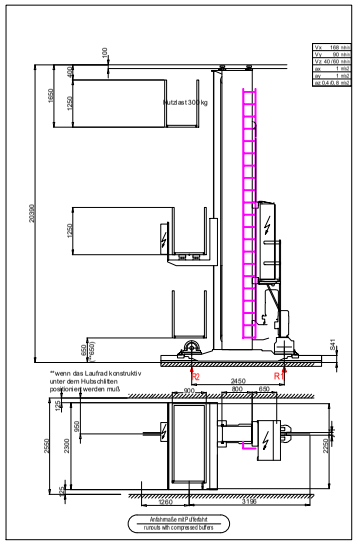
Schnitt A-A

Maßstab 1:75



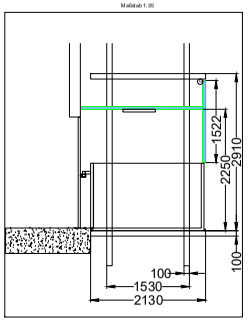
Regalbediengerät

Maßstab 1:50



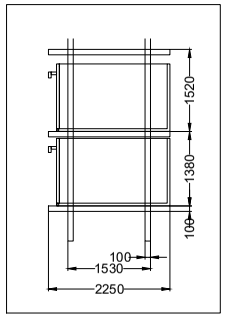
Boxen-Umschlagsplatz-Kreuzriss

Maßstab 1:36



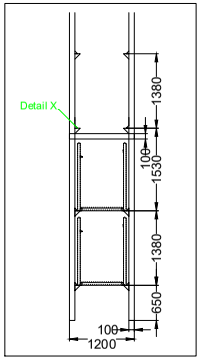
Lagerplatz-Rückseite-Kreuzriss

Maßstab 1:36



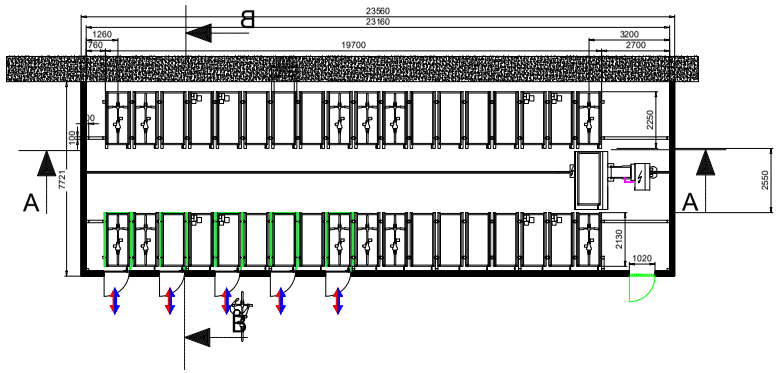
Lagerplatz-Aufriss

Maßstab 1:36



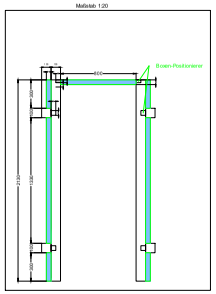
Grundriss

Maßstab 1:75



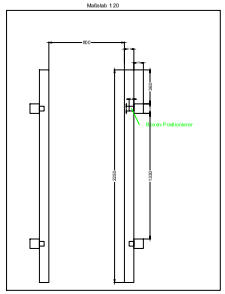
Boxen-Umschlagsplatz-Grundriss

Maßstab 1:20



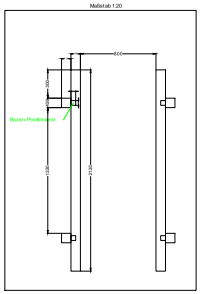
Lagerplatz-Rückseite-Grundriss

Maßstab 1:20



Lagerplatz-Vorderseite-Grundriss

Maßstab 1:20



Lagerkapazität brutto:

Transporteinheit: Metallbox: 900x2100

Gasse 1

Anzahl Palettenplätze in X-Richtung:	18	PP
Anzahl Palettenplätze in Y-Richtung:	14	PP
Anzahl Palettenplätze in Z-Richtung:	2	PP
Zwischensumme	504	PP
Verluste durch Einlagerungsplätze:	10	PP
Gesamt	ca. 494	PP

VORABZUG - UNVERBINDLICH !
PRELIMINARY - NOT BINDING !

Index	Änderung	Datum	Name	Ausführung	gezeichnet / checked No.
			Schlosser	0	1/50
					Datum 20.02.2023 Name SONJ
					gepr.
					genehm.
					Maßstab 1/50
					Blatt-Nr.

GTW Gesellschaft für Technische Logistik mbH

Anlage: Fahrradlager

Auftrag: Fahrradlager Block - X

Layout Fahrradlager

Maßeinh. kg

Zeichnungs-Nr.

Zeichnungsdatei: Fahrradlager_v01

This drawing is intellectual property of GTW Gesellschaft mbH, Achtal-Lana 33, D 922 Windhaf / Austria
This drawing or / or information is our prop erty and must not be copied, not distributed or made available to a ll without permission.

A0

Zusammenfassung Spielzeitenberechnung

Projekt	**Fahrrad-Turm**
Layout Nr.	**Fahrradlager**
Datum	

Geforderte Gesamtleistung

Einlagerung	40	Pal/h
Auslagerung	40	Pal/h
Prozent kombinierte Spiele	50%	
ergibt Einzelspiele Einlagerung	20,0	/h
ergibt Einzelspiele Auslagerung	20,0	/h
ergibt kombinierte Spiele	20,0	/h
Anzahl RBG	1	-

Berechnete Einzelzeiten je Fall
kein Sonderfall
ein RBG in einem Gang

Fall 4 Ein- und Auslagerung in x-Richtung verschoben, E=A

Dauer Einzelspiel Einlagerung	40,7	s
Dauer Einzelspiel Auslagerung	40,7	s
Dauer kombiniertes Spiel	74,2	s

Berechnete Gesamtzeit

20 x Einzelspiel Einlagerung	814	s
20 x Einzelspiel Auslagerung	814	s
20 x kombiniertes Spiel	1483	s
Berechnete Gesamtzeit	**3112**	**s**
Brechnete Gesamtzeit pro Gerät	**3112**	**s**
Gesamtauslastung	**86**	**%**

Zusammenfassung Spielzeitenberechnung

Projekt	**Fahrrad-Turm**
Layout Nr.	**Fahrradlager**
Datum	

Geforderte Gesamtleistung

Einlagerung	40	Pal/h
Auslagerung	40	Pal/h
Prozent kombinierte Spiele	80%	
ergibt Einzelspiele Einlagerung	8,0	/h
ergibt Einzelspiele Auslagerung	8,0	/h
ergibt kombinierte Spiele	32,0	/h
Anzahl RBG	1	-

Berechnete Einzelzeiten je Fall
kein Sonderfall
ein RBG in einem Gang

Fall 4 Ein- und Auslagerung in x-Richtung verschoben, E=A

Dauer Einzelspiel Einlagerung	41,9	s
Dauer Einzelspiel Auslagerung	41,9	s
Dauer kombiniertes Spiel	76,3	s

Berechnete Gesamtzeit

8 x Einzelspiel Einlagerung	335	s
8 x Einzelspiel Auslagerung	335	s
32 x kombiniertes Spiel	2440	s
Berechnete Gesamtzeit	**3111**	**s**
Brechnete Gesamtzeit pro Gerät	**3111**	**s**
Gesamtauslastung	**86**	**%**

Zusammenfassung Spielzeitenberechnung

Projekt	**Fahrrad-Turm**
Layout Nr.	**Fahrradlager**
Datum	

Geforderte Gesamtleistung

Einlagerung	40	Pal/h
Auslagerung	40	Pal/h
Prozent kombinierte Spiele	50%	
ergibt Einzelspiele Einlagerung	20,0	/h
ergibt Einzelspiele Auslagerung	20,0	/h
ergibt kombinierte Spiele	20,0	/h
Anzahl RBG	1	-

Berechnete Einzelzeiten je Fall
kein Sonderfall
ein RBG in einem Gang

Fall 4 Ein- und Auslagerung in x-Richtung verschoben, E=A

Dauer Einzelspiel Einlagerung	45,6	s
Dauer Einzelspiel Auslagerung	45,6	s
Dauer kombiniertes Spiel	81,7	s

Berechnete Gesamtzeit

20 x Einzelspiel Einlagerung	913	s
20 x Einzelspiel Auslagerung	913	s
20 x kombiniertes Spiel	1634	s
Berechnete Gesamtzeit	**3459**	**s**
Brechnete Gesamtzeit pro Gerät	**3459**	**s**
Gesamtauslastung	**96**	**%**

per E-Mail

Orion Bausysteme GmbH Postfach 1105 64580 Biebesheim am Rhein

Wolfgang Huber
A-6800 Feldkirch

Vorgangsart / Vorgangsnummer	Aktenzeichen	Datum	Zeichen	Kunden-Nr.	Seite
ANGEBOT 107539	**10.9648/22**	**29.09.2022**	**ib**	**1**	**1 / 4**

Bestelldaten:	per E-Mail am 29.09.2022 durch Herrn Huber	Bearbeiter:	Ivan Bykov
		Tel.-Dw.:	5552-0

Diesem Vorgang liegen unsere Allgemeinen Verkaufs- und Lieferbedingungen zugrunde, die wir Ihnen auf Wunsch gerne kostenlos zur Verfügung stellen. Ebenso besteht die Möglichkeit, vorgenannte Bedingungen auf unserer Internetseite www.orion-bausysteme.de einzusehen.

ORION BAUSYSTEME GMBH

Pos	Kennziffer	Menge	ME	Bezeichnung	Einzelpreis	Gesamtpreis

BV: Kärnten, Projekt: "Unterstützung emissionsfreier Mobilität"

499997
FK

Multifunktionsbox **ARETUS-Doppelstock**
Modularer Aufbau, bestehend aus einem Grundelement und einer beliebigen Anzahl von Anbauelementen.

Stabiles Stahlgerüst über spezielle Verbindungselemente zu einem tragenden Gehäuse konfektioniert. Wandverkleidung aus gekanteten Stahlblechkassetten. Dachelement aus stabilem Stahlblech. Massive Türelemente mit verstärkten Rahmen. Alle Stahlteile sind feuerverzinkt. Die Sichtflächen der Stahlbleche werden zusätzlich pulverbeschichtet.

Nummerierung der Boxen: 001 - 030

Gesamtbreite : ca. 12,05 m

1 Reihenanlage mit 15 Doppelboxen (= 30 Radeinstellungen)

ARETUS-Reihenanlage, bestehend aus:

Vorgangsart / Vorgangsnummer	Aktenzeichen	Datum	Zeichen	Kunden-Nr.	Seite
ANGEBOT 107539	**10.9648/22**	**29.09.2022**	**ib**	**1**	**2 / 4**

Pos	Kennziffer	Menge	ME	Bezeichnung	Einzelpreis	Gesamtpreis
				--- Übertrag ---		
1	402200 FK	1	St	**Fahrradbox ARETUS Doppelstock - Grundelement** Gesamthöhe : ca. 2.550 mm Gesamtbreite : ca. 850 mm Gesamttiefe : ca. 2.000 mm Türmaß (Breite x Höhe) : ca. 750 x 1.190 mm Anzahl Radeinstellungen : 2 Verriegelung : Einsteckschloss Oberfläche Stahlrahmen : feuerverzinkt Oberfläche Stahlbleche : RAL		
2	403200 FK	14	St	**Fahrradbox ARETUS Doppelstock - Anbauelement** Gesamthöhe : ca. 2.550 mm Gesamtbreite : ca. 800 mm Gesamttiefe : ca. 2.000 mm Türmaß (Breite x Höhe) : ca. 750 x 1.190 mm Anzahl Radeinstellungen : 2 Verriegelung : Einsteckschloss Oberfläche Stahlrahmen : feuerverzinkt Oberfläche Stahlbleche : RAL		
		1	St	Grundpreis für 1 Reihenanlage mit 15 Doppelstockboxen (=30 Radeinstellungen)	32.495,00	32.495,00

Pos	Kennziffer	Menge	ME	Bezeichnung	Einzelpreis	Gesamtpreis
				--- Übertrag ---		32.495,00
3	499996 FK	30	St	Aufpreis für eine Ausführung der v. g. Fahrradbox ARETUS-Doppelstock mit Energiesäule Ausstattung Energiesäule: - 230V Steckdose innenliegend - Einsteckschloss mit Notfallöffnung - Unterverteiler mit FI Schalter und 10A Sicherung - Öffnung Erdanschlusskabel - Potentialausgleich - CE-Gütezeichen <u>Bauseitige Leistungen</u> Herstellung einer ebenen, befestigten Grundfläche und der Anschluss der ARETUS Boxen an das örtliche Stromnetz **(PREIS JE FAHRRADBOX)** Die Gesamtbreite erhöht sich um ca. 10 cm je Box! Selbstverständlich können Sie auch lediglich die untere Box mit einer E-Ladesäule ausstatten	700,00	21.000,00
4	8894 M	1	St	Montage der Fahrradboxen durch Aufschrauben auf befestigtem Untergrund, ohne Rücknahme und Entsorgung der Transportverpackung. Bei der Kalkulation der Montage sind wir davon ausgegangen, dass diese in einem Zuge erfolgen kann. Sollte dies nicht möglich sein, werden wir zusätzliches An- und Abfahren gesondert in Rechnung stellen. Sie erhalten im Auftragsfall unsere Dispositionszeichnung zur eigenverantwortlichen Prüfung und genehmigten Rücksendung. Danach wird die Produktion aufgenommen. Wir sind bei unserer Kalkulation davon ausgegangen, dass die Fahrradboxen ebenerdig,d.h. ohne Geländegefälle ausgeführt werden.	5.000,00	5.000,00

Vorgangsart / Vorgangsnummer	Aktenzeichen	Datum	Zeichen	Kunden-Nr.	Seite
ANGEBOT 107539	**10.9648/22**	**29.09.2022**	**ib**	**1**	**4 / 4**

Pos	Kennziffer	Menge	ME	Bezeichnung	Einzelpreis	Gesamtpreis
				--- Übertrag ---		58.495,00

Preise:
netto, ab Werk, anteilige Transportkosten siehe unten, Spalte "Fracht", Ware nicht abgeladen, ohne / mit Montage, zzgl. MwSt., gemäß unseren Allgemeinen Geschäftsbedingungen.

Lieferzeit:
ca. 16 Wochen nach Vorlage aller zur Produktion erforderlichen Daten, Zahlungseingang und Planfreigabe.

==========
Qualitätsmanagement nach DIN EN ISO 9001:2015
Zertifiziert als Schweißfachbetrieb nach
DIN EN ISO 3834 - 2 : 2006
Zertifizierung der werkseigenen
Produktionskontrolle nach EN 1090 - 1
in Verbindung mit der EN 1090 - 2, EXC 3
DB Zertifizierungsstufe CL 2 nach DIN EN 15085 -2
Leistungserklärungen zu den Produkten finden Sie
auf unserer Homepage www.orion-bausysteme.de
==========

Warenwert	Verpackung	Fracht	Steuerpfl. Betrag	StSchl.	Mwst-Satz	MwSt-Betrag	Endbetrag
58.495,00		**1.500,00**	**59.995,00**	**2**	**19,00**	**11.399,05**	**71.394,05**

EURO

Zahlung: 1/3 bei Erhalt der Auftragsbestätigung
1/3 vor Versand, 1/3 nach Rechnung

Termin ca.: siehe Text
Versandanschrift:

Versandart:
Frachtvermerk:

Adresse
Waldstraße 2, 64584 Biebesheim am Rhein
Postfach 11 05, 64580 Biebesheim am Rhein
Amtsgericht Darmstadt,
Registergericht Darmstadt, HRB 51755
Geschäftsführer: Jörg Suckow, Frank Suckow

Telefon (0 62 58) 5552-0
Telefax (0 62 58) 5552-36
E-Mail info@orion-bausysteme.de

Banken
Kreissparkasse Groß-Gerau (BLZ 508 525 53) 14 000 012
IBAN DE14 5085 2553 0014 0000 12 • SWIFT-BIC=HELADEF1GRG
Volksbank Darmstadt-Südhessen eG (BLZ 508 900 00) 3 203 956
IBAN DE68 5089 0000 0003 2039 56 • SWIFT-BIC=GENODEF1VBD
Steuer-Nr. 007 240 70330 • Umsatzsteuer-Ident. Nr.: DE 111 652 494

X

Datum 29.09.2022
Fon 02262/22 111 / 0
Fax 02262/22 111 / 99
Betreuer /ph
E-Mail wl@woehrparking.at
Aktenzeichen **(Angebots-Nr) 22/xxxx**

WÖHR Autoparksysteme GmbH | 2100 Korneuburg | Industriestrasse 1/A OG.03

Herrn Wolfgang Huber
Gemeinde Velden am Wörthersee

RICHTPREISANGEBOT
Lieferung, Montage und Inbetriebnahme von einem automatischen Fahrradparksystem WÖHR Bikesafe Turm 885 / 16 / 8 / 122 Stellplätzen, Turm 885 / 16 / 6 / 90 Stellplätzen und 885 / 16 / 4 / 58 Stellplätzen

Sehr geehrter Herr Huber,

vielen Dank für Ihr Interesse an unserem automatischen Fahrradparksystem.
Gerne erhalten Sie nachfolgend unsere Richtpreise für den Bikesafe 885 für 58, 90 und 122 Fahrradstellplätze in der Standardkonfiguration.

Bitte beachten Sie, dass die genannten Preise für unsere Standardkonfiguration als Turmvariante gelten. Unterschiedliche Konfigurationen der Fassade, Anzahl der Übergabebereiche und Kombination Turm- / Schachtvariante sind möglich. Bitte teilen Sie uns Ihre Anforderungen mit, damit wir Ihnen ein projektbezogenes Angebot unterbreiten können.

WÖHR entwickelt, produziert und montiert hauseigene Auto- und Fahrradparksysteme, die wir nach Übergabe auch servicetechnisch betreuen.

Unsere deutsche Muttergesellschaft ist ein Familienunternehmen, wurde bereits 1902 gegründet und feiert nunmehr sein 120-jähriges Bestehen. Bereits 1959 wurde von WÖHR das erste Parksystem für zwei Autos übereinander entwickelt und montiert. Seit 60 Jahren haben wir uns auf innovative Parkraumlösungen spezialisiert: vom einfachen Doppelparker für zwei Autos übereinander bis hin zu vollautomatischen Hochregallagern mit mehreren hundert Stellplätzen.

Unsere Stärken sind die individuelle und effiziente Erfüllung von Kundenwünschen und die Lösung von Parkproblemen. Qualifizierte Mitarbeiter und modernster Maschinenpark unterstützen die Innovationsstärke des Unternehmens.

Sie bekommen bei uns alles aus einer Hand. Daher sichern wir Ihnen eine seriöse Beratung und einen kompetenten After-Sales-Service zu. Vertrauen Sie auf die Erfahrung von WÖHR.

Bestandteile unseres Richtpreisangebots:

1x Fahrradparksystem Bikesafe mit 8 Parkebenen und 122 Stellplätzen als Turmvariante
Standard-Bedienkonzept mit RFID Chips für Dauermieter
Übergabekabine in Aluminium Verkleidung, Farbton RAL 9006 Weißaluminium und RAL 1003 Signalgelb (alternative Farben aus dem RAL Classic möglich)
Inkl. Schnittstelle für Ferndiagnosesystem
inkl. Transport innerhalb Deutschland
inkl. Montage und Elektroarbeiten am Schaltschrank WÖHR
inkl. Kran, Arbeitsbühne, und Gerüst für die Montage
inkl. Inbetriebnahme und Dokumentation

Richtpreis ***netto EURO 825.000,-***

1x Fahrradparksystem Bikesafe mit 6 Parkebenen und 90 Stellplätzen als Turmvariante
Standard-Bedienkonzept mit RFID Chips für Dauermieter
Übergabekabine in Aluminium Verkleidung, Farbton RAL 9006 Weißaluminium und RAL 1003 Signalgelb (alternative Farben aus dem RAL Classic möglich)
Inkl. Schnittstelle für Ferndiagnosesystem
inkl. Transport innerhalb Deutschland
inkl. Montage und Elektroarbeiten am Schaltschrank WÖHR
inkl. Kran, Arbeitsbühne und Gerüst für die Montage
inkl. Inbetriebnahme und Dokumentation

Richtpreis ***netto EURO 750.000,-***

1x Fahrradparksystem Bikesafe mit 4 Parkebenen und 58 Stellplätzen als Turmvariante
Standard-Bedienkonzept mit RFID Chips für Dauermieter
Übergabekabine in Aluminium Verkleidung, Farbton RAL 9006 Weißaluminium und RAL 1003 Signalgelb (alternative Farben aus dem RAL Classic möglich)
Inkl. Schnittstelle für Ferndiagnosesystem
inkl. Transport innerhalb Deutschland
inkl. Montage und Elektroarbeiten am Schaltschrank WÖHR
inkl. Kran, Arbeitsbühne und Gerüst für die Montage
inkl. Inbetriebnahme und Dokumentation

Richtpreis ***netto EURO 690.000,-***

Alle Preise sind netto, die aktuell gültige gesetzliche Mehrwertsteuer ist hinzuzurechnen.

Bestandteil Mehrpreis für Schließfachanlage mit Ladeeinheit:

1x Schließfacheinheit S20
Tragende Konstruktion: Feuerverzinktes Portalrahmen aus Stahl-Profilen
Ausführung als 4-Feld Schließfacheinheit
Tragende Konstruktion kpl. in RAL Ton nach Serienfarben WÖHR oder Absprache beschichtet
Abmessungen: HxBxT ca. 2085x1630x533,5 mm

Abmessungen Schließfach: HxBxT ca. 400x400x500 mm

- 19 von 20 Fächer frei nutzbar, mit Lademöglichkeit (1 Ladesteckdose 230 V pro Fach)
- 1 von 20 Fächer wird für Steuerung benötigt
- Pro Schrankzeile werden die Ladesteckdosen über FI abgesichert
- Absicherung pro Schrankzeile mit 16A
- Aufgedübelt auf festem Untergrund
- Inkl. Montage

Richtpreis ***netto EURO 59.000,-***

Bild 1:

Beispielhafte Darstellung einer 3-Feld Schließfachanlage. Überdachung ist optional und in unserem Preis Pos. 2 nicht enthalten.
Farbton ist individuell wählbar.

3. Technische Daten

Stromanschluss:	3 Phasen + N + PE, (TN-S Netz), 400 V, 50 Hz
Elektrische Absicherung:	Betriebsklasse gL/gG (3x50 A) -K träge

4. Richtwerte Energieaufwand für den Bikesafe (ohne Schließfachanlage)

Stromverbrauch pro Ein- oder Auslagerungsvorgang	ca. 0,04 kWh

5. Zugriffszeiten

Bikesafe 885 - 8 Parkebenen und 122 Stellplätze

Die Zugriffzeit ist die Zeit von der Anforderung des Fahrrades am Bedienmonitor bis das Fahrrad abholbereit in der Radschiene des Übergabebereiches steht. Die Zugriffzeit Ausparken bezieht sich jeweils auf eine Einzelanforderung. Mehrfachanforderungen sind nicht berücksichtigt.

Minimale Zugriffzeit des nächsten Rades: ca. 20 Sekunden
Maximale Zugriffzeit des entferntesten Rades: ca. 30 Sekunden

Die Einparkzeit ist die Anlagenzeit gemessen von der Bestätigung des Nutzers am Bedienmonitor bis die Anlagentür nach Einzug des Fahrrades wieder geschlossen ist.

Einparkzeit: ca. 10 Sekunden

Die Nutzerzeit ist die angenommene Zeit die ein durchschnittlicher Nutzer benötigt sein Fahrrad beim Einparken abzustellen bis zur Anforderung am Bedienmonitor bzw. beim Ausparken sein Fahrrad nach Bereitstellung wieder zu entnehmen und den Übergabebereich freizumachen.

Nutzerzeit: ca. 20 Sekunden

Die Parkkapazität ergibt sich durch die Einparkzeit (Anlagenzeit) plus die Nutzerzeit bezogen auf eine Stunde. Sie gilt für kontinuierliches Ein- bzw. Ausparken. Ein Vorzug der Technik ist hierbei, dass der Lift unabhängig vom Übergabebereich arbeiten kann, sodass sich die abgeleisteten Parkvorgänge pro Stunde erhöhen.

Parkkapazität ca. 120 Fahrräder pro Stunde

6. Max. Fahrradabmessungen

Bei einer lichten Einbauhöhe von 11,80 m (entspricht 8 Parkebenen) und einem Durchmesser von 7,50 m, können Fahrräder mit den aufgeführten max. Abmessungen eingelagert werden.

Länge:	min.	1.500	mm
	max.	2.000	mm
Lenker-Breite:	max.	760	mm
Höhe:	max.	1.200	mm (90 Stellplätze) und
	max.	1.400	mm (32 Stellplätze)
Gewicht:	max.	30	kg
Raddurchmesser:	min.	24	Zoll
	max.	29	Zoll
Felgenbreite:	min.	22	mm
	max.	80	mm

Optional steht der Bikesafe für Lenkerbreiten von max. 83 cm ebenfalls zur Verfügung. Bitte fragen Sie da den entsprechenden Mehrpreis an.

7. ACHTUNG:

1. Helme, Taschen sowie feste Anbauteile (z.B. Fahrradkörbe, Einkaufskörbe, Satteltaschen, Fahrradtaschen, Kindersitze oder sonstiges) können im Bikesafe eingelagert werden, sofern sie den maximalen Abmessungen und Anforderungen des Bikesafes entsprechen.

2. 20“ Klappfahrräder können nicht eingelagert werden.

3. Der **Bieter** rät ab, Fahrräder mit Spikes einzulagern. Spikes führen auf Dauer zu einem erhöhten Verschleiß der Beschichtung auf den Radschienen.

4. Im eingeklappten Zustand darf der Fahrradständer nicht über das Hinterrad hinausragen.

5. Das Hinterrad-Schutzblech darf max. 30 cm über dem Boden enden.

6. Die Länge des Gepäcks auf dem Gepäckträger darf max. 6 cm länger als das Fahrrad sein, gemessen am hinteren Reifen.

7. Der Abstand zwischen Gepäcktaschen im Front und der vorderen Reifenflanke muss mind. 18 cm betragen.

8. Bedienkonzept

Der Übergabebereich ist mit einem Bedientableau ausgestattet. Dieser befindet sich in Sichtweite des Einfahrtstors. Der Nutzer betritt die Trittmatte und stellt sein Fahrrad in die dafür vorgesehene Radschiene ab. Das Einfahrtstor öffnet analog der Radbreite, das Vorderrad wird bis zum Radanschlag eingeschoben.

a) Bedienung über RFID Chip für Dauermieter bzw. gleichbleibenden Nutzerkreis (Standardkonfiguration):

Das Schiebetor wird geschlossen und das Rad fixiert. Der Nutzer verlässt die Trittmatte und bestätigt am Bedienterminal mit dem RFID Chip die Einlagerung. Das Vorderrad wird vom Vertikalförderer gefasst und in das Regalsystem eingezogen. Gleichzeitig prüfen Lichtschranken die Höhe, Breite und Länge des Fahrrads.
Falls das einzulagernde Fahrrad die genannten max. Abmessungen überschreitet, wird die Einlagerung abgebrochen und das Fahrrad im Schleichgang wieder in den Übergabebereich herausgeschoben. Wenn das Fahrrad korrekt eingelagert werden kann, erhält der Nutzer eine Bestätigung auf der Textanzeige am Bedientableau.

Das Einfahrtstor wird automatisch geschlossen, der Nutzer verlässt den Übergabebereich. Hinter verschlossenen Toren wird das Fahrrad automatisch, sicher und schnell im Fahrradparksystem eingelagert.

Beim Auslagern erfolgt die Anforderung des jeweiligen Fahrrads wieder mittels RFID Chip. Dieser wird an das Bedientableau gehalten, das Fahrrad wird automatisch in den Übergabebereich transportiert.
Der Nutzer wartet im Durchschnitt ca. 16 Sekunden bis sein Fahrrad im Übergabebereich bereit steht. Sobald der Nutzer die Trittmatte betritt, wird das Vorderrad vom Vertikalförderer freigegeben und das Einfahrtstor wird geöffnet. Der Nutzer muss sein Fahrrad rückwärts herausschieben. Das Einfahrtstor wird automatisch geschlossen.

Mehrere Ausparkvorgänge können gleichzeitig angefordert werden. Die Systemsteuerung speichert die Reihenfolge der Anforderungen und lagert nacheinander aus.
Die Einlagerung der Fahrräder erfolgt dynamisch. Das einzulagernde Fahrrad hat keinen festen Lagerplatz, sondern wird auf ein beliebiges, gerade nicht belegtes und mit geringster Entfernung zum Übergabebereich verfügbaren Lagerplatz befördert.

b) Bedienung über einen QR Code für einen wechselnden Nutzerkreis (gegen Mehrpreis erhältlich):

Der **Bieter** stellt dem **Auftraggeber** ein Online Buchungsportal zur Verfügung. Der Kurzzeit Parker muss eine einmalige Registrierung inklusive Hinterlegung der Bankdaten durchführen.
Dann bucht der Nutzer einen Stellplatz für die Dauer x und erhält eine automatische Benachrichtigung auf seinem Smartphone mit dem Zugangscode. Sobald der Zugangscode am Bedienterminal aktiviert wird, kann der Nutzer sein Fahrrad wie oben beschrieben einlagern. Nach der Auslagerung erlischt die Gültigkeit des QR Codes.
Aufgrund der geringen Parkgebühr für das Parken von Fahrrädern, wird die Gebühr – falls vom Auftraggeber so gewünscht – direkt beim Buchen eines Stellplatzes vom Bankkonto des Nutzers abgebucht. Die Höhe der Parkgebühr obliegt dem Auftraggeber bzw. Betreiber.

9. Ausschluss

Folgende Punkte sind nicht im Lieferumfang enthalten und vom **Auftraggeber** zu erbringen:

a. Einholung der Baugenehmigung
b. Abstimmung Brandschutz und alle damit verbundenen und eventuell zu erbringenden Maßnahmen (ist bauseits mit dem jeweiligen Bauamt abzustimmen)
c. Alle Kosten und Gebühren für behördliche oder sonstige Abnahmen und Prüfungen
d. Alle Kosten und Gebühren für Umweltschutzauflagen, Vogelschutz und dergleichen
e. Blitzschutz
f. Erschließung des Grundstücks
g. Vermessen der zu bebauenden Fläche
h. Alle Arbeiten und Teile unter Punkt 10 und 11
i. Erforderliche Drainage außerhalb und innerhalb des Fahrradparksystems
j. Fundament- und Betonarbeiten für die Kraftaufnehmer und Verankerungen der Parksystemteile oder Komponenten
k. Kabelkanäle und Leitungen die bis zum Schaltschrank des **Bieters** gelegt werden müssen

l. Erdung und Potentialausgleich
m. Stromanschluss gemäß Punkt 4
n. Verkehrssicherungsmaßnahmen und Baustellensicherung während der Montage
o. Alle Leistungen die in unserem Angebot nicht ausdrücklich erwähnt wurden

10. Bauseitige Montagevoraussetzungen

1. Fertige Fundament- und Betonarbeiten gemäß Vorgaben des **Bieters**
2. Bereitstellung eines ausreichenden und geeigneten Lagerplatzes für das angelieferte Material (mindestens 60 m²)
3. Der Montagestandort muss per Lkw, Kran, Gabelstapler zugänglich sein
4. Baustrom und – Verteiler, Beleuchtung, Steckdosen, usw. müssen fertiggestellt sein. Der Elektrofestanschluss für das Fahrradparksystem muss spätestens 6 Wochen vor Übergabetermin installiert sein.
5. Fertige Hauptzuleitung zum Schaltschrank, Absicherung gemäß Punkt 4
6. Sicherung der Montagestelle

11. Zusätzliche Hinweise

Bauseitig ist insbesondere folgendes zu beachten:

1. **Behördliche Abstimmung**
   Evtl. behördliche Auflagen, insbesondere zum Brandschutz sind bauseitig mit dem Bauamt abzustimmen. Falls durch solche Auflagen Änderungen an der vertraglichen Systemkonzeption erforderlich werden, wird eine Angebotsanpassung vorgenommen.

2. **Elektroverdrahtung:**
   Die Elektroverdrahtung bis zum Schaltschrank des Bieters mit 3 Phasen + N + PE (TN-S Netz), 400 Volt, 50 Hz, TN-S Netz erfolgt bauseitig.
   Die max. Netzspannungstoleranz darf nicht mehr als ± 10%, die max. Frequenztoleranz nicht mehr als ± 2% betragen.
   Die Zuleitung muss bauseitig vorgesehen werden und nach den Angaben des Bieters gesichert werden. Die Elektroversorgung muss abgesichert werden.

3. **Erdung und Potentialausgleich**
   Im Aufstellungsort des Schaltschranks ist vom Auftraggeber eine Anschlussmöglichkeit zum Fundamenterder vorzusehen, da die Potentialausgleichsschiene im Schaltschrank über eine möglichst kurze Leitung mit dem Fundamenterder verbunden werden muss.
   Im Aufstellungsbereich des Stahlbaus sind vom **Auftraggeber** Fundamenterder vorzusehen, da der gesamte Stahlbau über möglichst kurze Leitungen durch den **Bieter** mit den Erden verbunden werden muss.

4. **Maße**
   Alle Maße sind Mindestfertigmaße. Toleranzen sind zusätzlich zu berücksichtigen.

5. **Straßensperrung**
   Falls aus Gründen außerhalb des Einflusses vom Anbieter ein Abladen nur mittels Straßensperrung oder außerhalb der betriebsüblichen Arbeitszeit möglich ist, werden solche Zusatzaufwendungen gesondert in Rechnung gestellt.

Die angegebenen Preise sind nur als Richtpreis anzusehen. Ein Fixpreisangebot können wir erst nach Klärung der Projektanforderungen abgeben.

Mit freundlichen Grüßen

WÖHR Autoparksysteme GmbH

Wolfgang Lenke