**-**

**HTL Saalfelden**

**Systemplanung und Projektentwicklung**

****

**Projektdokumentation**

**2024 / 2025**

|  |  |
| --- | --- |
| **Projektbezeichnung** | MBot - Backend |
| **Projektteam** | Aberger Jonas, Haslinger Fabian, Hechenberger Tim |
| **Erstellt am** | 06.02.2025 |
| **Letzte Änderung am** | 27.03.2025 |
| **Status** | In Bearbeitung |
| **Aktuelle Version** | 3.0 |

**Änderungsverlauf**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nr.** | **Datum** | **Version** | **Geänderte Kapitel** | **Art der Änderung** | **Autor** |
| <1> | <06.02.2025> | <1.0> | <Alle> | <Erstellung> | Tim Hechenberger |
| <2> | <26.02.2025> | <2.0> |  | <Hinzufügen> | Fabian Haslinger |
| <3> | <27.03.2025> | <3.0> |  | <Komplette Überarbeitung + Sprint 3> | Jonas Aberger |

**Inhalt**

[1. Allgemeines / Projektübersicht 4](#_Toc965522712)

[1.1 Projektbeschreibung 4](#_Toc1040632592)

[1.2 Projektteam und Schnittstellen 4](#_Toc284604490)

[2. Funktionale Anforderungen 4](#_Toc521691458)

[2.1 Use Cases 4](#_Toc1840059643)

[2.1.1 <Name Use Case 1> 4](#_Toc1384425644)

[2.1.2 <Name Use Case 2> 4](#_Toc2131939576)

[2.1.3 <Name Use Case n> 4](#_Toc1332338971)

[3. Nichtfunktionale Anforderungen 4](#_Toc542409565)

[4. Projektplanung 5](#_Toc1181161558)

[4.1 Variantenbildung 5](#_Toc191553135)

[4.2 Machbarkeitsstudie 5](#_Toc729802831)

[4.3 Allgemeine Planungsinformationen 5](#_Toc2024584587)

[4.4 Projektumfeldanalyse 5](#_Toc502666407)

[5. Softwarearchitektur 5](#_Toc2079537197)

[5.1 Aktivitätsdiagramme 6](#_Toc659892040)

[5.1.1 Aktivitätsdiagramm 1 Name 6](#_Toc1701078734)

[5.1.2 Aktivitätsdiagramm n Name 6](#_Toc1952219604)

[5.2 Sequenzdiagramme 6](#_Toc452400050)

[5.2.1 Sequenzdiagramm 1 Name 6](#_Toc539048854)

[5.2.2 Sequenzdiagramm n Name 6](#_Toc1656117885)

[5.3 Komponentendiagramme 6](#_Toc1549670347)

[5.4 Verteilungsdiagramme 6](#_Toc1782933099)

[5.5 Softwarekomponenten / Programme 6](#_Toc863584749)

[5.5.1 SW Programme 6](#_Toc2059131748)

[5.5.2 SW Komponenten 6](#_Toc2020768912)

[6. Projektdurchführung 7](#_Toc1459545787)

[6.1 Sprint 1 7](#_Toc965912477)

[6.1.1 Sprintplanung 7](#_Toc1238068464)

[6.1.2 Sprint Demo 7](#_Toc1608356264)

[6.1.3 Sprint Retrospektive 7](#_Toc1207657317)

[6.1.4 Sprint Zusammenfassung 7](#_Toc56107529)

[6.2 Sprint 2 8](#_Toc501505208)

[6.2.1 Sprintplanung 8](#_Toc1492261807)

[6.2.2 Sprint Demo 8](#_Toc577798987)

[6.2.3 Sprint Retrospektive 8](#_Toc194081203)

[6.2.4 Sprint Zusammenfassung 8](#_Toc729203803)

[6.3 Sprint n 8](#_Toc562254915)

[7. Installation / Software deployment 8](#_Toc1526420175)

[8. Projektabschluß 8](#_Toc1271613368)

[8.1 Projektzusammenfassung 8](#_Toc1743416473)

[8.2 Attachments 8](#_Toc1717973310)

# Allgemeines / Projektübersicht

## Projektbeschreibung

Das Projekt umfasst die Entwicklung einer Applikation zur ferngesteuerten Bedienung des mBot2-Roboters. Die Konfiguration des Roboters unterstützt eine zweiseitige Kommunikation zwischen dem Klienten und dem Endgerät, dabei wird neben dem primären Steuerungsmodul eine Vielzahl weiterer Schnittstellen implementiert.

Die Anwendung soll sowohl auf PCs als auch auf mobilen Geräten nutzbar sein.

**Projektteam**

|  |  |  |  |
| --- | --- | --- | --- |
| **Rolle(n)** | **Name** | **E-Mail** | **Team** |
| Entwickler | Jonas Aberger | jonas.aberger@htl-saalfelden.at | Backend |
| Entwickler | Fabian Haslinger | fabian.haslinger@htl-saalfelden.at | Backend |
| Entwickler | Tim Hechenberger | tim.hechenberger@htl-saalfelden.at | Backend |

# Funktionale Anforderungen

Ein Bild, das Diagramm, Entwurf, Text, Zeichnung enthält.

KI-generierte Inhalte können fehlerhaft sein.

## Use Cases

### <Steuerung des Mbots>

Der Benutzer steuert den mBot über das Frontend. Während der Steuerung gibt es die Möglichkeit das Programm mit einem Exit-Command zu beenden. Falls das Frontend die Fahrstrecke speichern will, wird der Use Case „Fahrstrecke speichern“ automatisch ausgeführt.

### <Fahrstrecke manuell erstellen>

Der Benutzer kann über das Frontend eine Fahrstrecke manuell erstellen. Dies bedeutet, dass er eine Route ohne direkte Steuerung des mBots vorgibt. Dies erfolgt durch eine grafische Benutzeroberfläche, bei der vordefinierte Wegpunkte abgefahren werden. Diese Wegpunkte bestehen aus den Attributen Richtung, Geschwindigkeit und Zeit.

### <Fahrstrecke speichern>

Dieser Use Case speichert die Fahrstrecke des mBots in einer **MongoDB-Datenbank**. Er kann durch zwei verschiedene Aktionen ausgeführt werden.

* Automatisch nach der Steuerung des mBots (wenn {fahrstreckenSpeichern == true}.
* Manuell durch den Use Case **„Fahrstrecke manuell erstellen“,** der das Speichern direkt einbindet´.

### <Fahrstrecke abrufen>

Der Benutzer kann eine bereits gespeicherte Fahrstrecke aus der Datenbank auswählen und abfahren. Damit eine Fahrstrecke jedoch abgerufen werden kann, muss zuvor mindestens eine Strecke in der Datenbank gespeichert worden sein.

### <LED & Geschwindigkeit anpassen>

Der Benutzer hat die Möglichkeit, die LEDs des mBots ein- oder auszuschalten, indem er entweder einen physischen Knopf betätigt oder eine digitale Schaltfläche in der Benutzeroberfläche auswählt. Wenn der Benutzer den Knopf drückt, werden die LEDs des mBots entweder aktiviert oder deaktiviert, abhängig von der gewählten Einstellung.

### <Statusinformationen senden>

Das System tauscht kontinuierlich Daten aus, um dem Benutzer aktuelle Statusinformationen zur Hardware anzuzeigen. Diese Informationen geben einen klaren Überblick über den Zustand der Hardware, wie beispielsweise Batteriestatus oder weitere Sensordaten. Dadurch behält der Benutzer stets den Überblick und kann bei Bedarf reagieren.

### <Kartografie des Raumes>

Der Benutzer möchte den umliegenden Raum erkunden und automatisch eine übersichtliche Karte erstellen lassen. Dieser Kartografie-Modus läuft eigenständig und unabhängig von anderen Funktionen. Der Benutzer kann ihn aktivieren, und der Mbot beginnt, den Raum systematisch zu erfassen und auszumappen. Die gesammelten Daten werden in Echtzeit verarbeitet und als detaillierte Karte dargestellt, die dem Benutzer zur Verfügung steht.

# Nichtfunktionale Anforderungen

Hier werden alle nichtfunktionalen Anforderungen der Backend-Gruppe-1 beschrieben:

* **Systemanforderungen:**
  + Das Backend wird als Webservice entwickelt und läuft auf einem lokalen Rechner
  + Die Kommunikation erfolgt über UDP
* **Leistung und Skalierbarkeit:**
  + Das System soll in der Lage sein, mehrere Roboter gleichzeitig zu verwalten
  + Keine zu hohen Latenzzeiten bei Steuerbefehlen und Synchronisation
* **Speicher und Ressourcennutzung:**
  + Kein zu hohem Speicherbedarf bei speichern der Daten, …

# Projektplanung

\*In diesem Kapitel sollen grundlegende Fragen geklärt werden bevor mit der Projektdurchführung begonnen wird. Obwohl es klar ist, dass dieses Projekt mit Hilfe von agilen PM Methoden durchgeführt wird, sollen vorab grundsätzliche Entscheidungen geklärt werden wie z.B. verwendete Programmiersprache, Variantenbildung, Betriebssystem, ...

Manche dieser Punkte können auch auf Grund der gegebenen Anforderungen als „gesetzt“ hingenommen werden.

(…)

## Variantenbildung

\*Auf Basis der Projektanforderungen, welche Varianten für die Umsetzung wurden, ausgearbeitet und welche wurde schließlich gewählt und warum. Welche Varianten wurden verworfen und aus welchem Grund.

* Basierend auf den Projektanforderungen wurden **verschiedene Varianten** analysiert:
* Programmiersprache: Python (mit MicroPython), Java, C#
  + Gewählte Variante: Python mit MicroPython
* Datenbank: FireBase, MongoDB, SQL
  + Gewählte Variante: MongoDB
* (…)

## Machbarkeitsstudie

\*Gab es einzelne Punkte, die vorher grob getestet wurden, um zu zeigen das die Umsetzung grundsätzlich möglich ist.

z.B. Kommunikation mit einem MQTT-Server mit Hilfe der Programmiersprache Python, ...

* **Nein**, es gab keine Punkte, die getestet wurden

## Allgemeine Planungsinformationen

\*Andere Planungsinformationen, welche nicht durch andere Kapitel abgedeckt sind, werden hier eingetragen.

* **Allgemeine Planungsinformationen:**
  + Die Entwicklung erfolgt in SCRUM mit einem Sprint 0, in dem das Grundkonzept, die Architektur und erste Tests durchgeführt wurden
  + Der Code und die Dokumentation werden in einem GitHub-Repository verwaltet
  + Die API-Spezifikation zwischen Frontend und Backend wurde bereits im 1. Sprint festgelegt, um parallele Entwicklung und testen zu ermöglichen

## Projektumfeldanalyse

\*Führen Sie eine Analyse des Projektumfeldes durch. Welche vergleichbaren Produkte gibt es bereits am Markt. Wie erfolgt die Abgrenzung zu diesen bereits bestehenden Produkten? Wer sind die relevanten Stakeholder des Projektes.

* **Vergleichbare Produkte:**
* Es gibt verschiedene mBot2-Steuerungs-Apps von mBlock, die aber keine individuelle Steuerung und Speicherung von Fahrstrecken bieten
* Bestehende Lösungen basieren meist auf Scratch, während unser System eine direkte Steuerung über Python und eine API ermöglicht
* **Abgrenzung:**
* Unser Projekt ermöglicht eine plattformunabhängige Steuerung (PC & Mobil)
* Der mBot2 kann nicht nur manuell gesteuert werden, sondern auch automatisierte Fahrstrecken abfahren
* Erweiterbare Backend-Architektur, die zukünftige Anpassungen erlaubt

# Softwarearchitektur

\*In diesem Kapitel soll der Aufbau der Software/Hardware beschrieben werden.

aus welchen Komponenten besteht das SW-Produkt.

Wie interagieren die einzelnen Komponenten miteinander.

Auf welcher Hardware läuft das System bzw. handelt es sich vielleicht sogar um ein verteiltes System. Wie kommunizieren diese Komponenten miteinander.

Dieses Kapitel beschreibt die Software- und Hardware-Architektur des Projekts. Es werden die einzelnen Komponenten erläutert, deren Interaktionen beschrieben und die technische Umsetzung visualisiert.

## Aktivitätsdiagramme

### Aktivitätsdiagramm 1 Name

Beschreibung der entsprechenden Aktivität und einfügen des Aktivitätsdiagramm

### Aktivitätsdiagramm n Name

Beschreibung der entsprechenden Aktivität und einfügen des Aktivitätsdiagramm

## Sequenzdiagramme

### Sequenzdiagramm 1 Name

Beschreibung der entsprechenden Sequenz und Einfügen des Sequenzdiagramm

### Sequenzdiagramm n Name

Beschreibung der entsprechenden Sequenz und Einfügen des Sequenzdiagramm

## Komponentendiagramme

Aus welchen Komponenten besteht die Software und welche Schnittstellen bieten diese an.

## Verteilungsdiagramme

Zeigt an wie die einzelnen Teile der Software auf die Hardwarekomponenten verteilt sind und wie die Hardwarekomponenten miteinander verbunden sind.

Auf welchem Rechner läuft welcher Software. Wie sind diese über ein Netzwerk miteinander verbunden.

## Softwarekomponenten / Programme

### SW-Programme

\*Auflistung aller verwendeten SW-Programme die bei der Umsetzung des Projektes verwendet worden sind. inkl. Angabe der Versionsnummer

z.B.: Visual Studio 2022,...

|  |  |  |
| --- | --- | --- |
| **Name** | **Version** | **Beschreibung** |
| Visual Studio Code | 1.97 | IDE |
| Python / Micro Python | 3.0 / akt. Ver. | Hauptprogrammiersprache für das Backend |
| Flask | 2.0 | API-Framework für das Backend |
| MongoDB | 8.0 | Speichern von Daten |
| GitHub | 3.15.2 |  |

### 5.5.2 SW Komponenten

Auflistung aller verwendeten SW Komponenten, welche für den Betrieb der SW benötigt werden. z.B.: Java Version, Apache Webserver, DotNet Framework, SW Library XY

inkl. Versionsnummer, Hersteller, Bezugsquelle (Downloadlink, ...) und SW-Lizenz (GPL, LGPL, Apache License, ...

# Projektdurchführung

## Sprint 1

### Sprintplanung

Dauer: <29.01.2025 – 18.02.2925>

**Ausgewählte User Stories:**

|  |  |
| --- | --- |
| **User-Story** | **Beschreibung** |
| Netzwerkanbindung | Automatische Verbindung Schul-WLAN und IP anzeigen |
| Verbindung zum Mbot herstellen | UDP-Verbindung einrichten und testen |

**Anzahl Story-Points:** 105 Story-Points

**Ausgewählte Punkte aus der Impediment Liste:**

* Netzwerkanbindung:
  + Statusinformation mittels LED ausgeben
  + IP-Adresse anzeigen
  + Eindeutige Identifikation
* Verbindung zum Mbot herstellen:
  + Kommunikation mit dem Mbot über das Netzwerk
  + Unterstützung von Steuerbefehlen an den Mbot
  + Antwort des Mbot auf Befehle verarbeiten und ans Frontend zurücksenden

### Sprint Demo

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

KI-generierte Inhalte können fehlerhaft sein.

### Sprint Retrospektive

Während des Sprints lief die Zusammenarbeit im Team sehr gut, und wir konnten die geplanten Aufgaben erfolgreich umsetzen. Anfangs brauchten wir jedoch etwas Zeit, um uns in das Projekt einzuarbeiten und die technischen Anforderungen zu verstehen.

### Sprint Zusammenfassung

Der Sprint verlief sehr gut. Basierend auf unserer Sprint-Velocity können wir davon ausgehen, dass wir den Endtermin voraussichtlich etwas schneller als geplant erreichen werden.

Ein Bild, das Text, Reihe, Screenshot, Diagramm enthält.

KI-generierte Inhalte können fehlerhaft sein.

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

KI-generierte Inhalte können fehlerhaft sein.

## Sprint 2

### Sprintplanung

Dauer: <19.02.2025 – 12.03.2025>

|  |  |
| --- | --- |
| **User-Story** | **Beschreibung** |
| Kommunikation / Anbindung zum Frontend | Informationsaustausch zwischen Frontend & Backend mittels Schnittstellen; Entgegennahme von Befehlen |
| Physisches Steuerungsmodul | Ansteuerung des MBOTS; Verbindung multipler Geräte |

**Anzahl Story-Points**: 185 Story-Points

**Ausgewählte Punkte aus der Impediment Liste:**

**Kommunikation / Anbindung zum Frontend:**

* Bereitstellung von Schnittstellen
* Genereller Informationsaustausch mit dem Frontend
* Erfolgreiche Entgegennahme von Inputs vom Frontend
* Abfertigung der Frontend-Backend Kommunikation

**Physisches Steuerungsmodul:**

* Erfolgreiche Anbindung an das DB-System
* Lokale Speicherung der Fahrstrecken
* Uploaden der Fahrstrecken zur DB
* Abrufen der gespeicherten Fahrstrecken

### Sprint Demo

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

KI-generierte Inhalte können fehlerhaft sein.

**Zusätzlich angefangene / erledigte User Stories:**

Ein Bild, das Text, Screenshot, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

### Sprint Retrospektive

Während des Sprints lief die Zusammenarbeit im Team sehr gut, und wir konnten die geplanten Aufgaben erfolgreich umsetzen. Da wir sehr schnell fertig wurden, konnten wir auch noch zusätzliche User Stories beginnen und finalisieren. Allerdings wurde der Fortschritt bei der Fahrstreckenlogik durch Komplikationen mit dem Frontend stark verzögert.

### Sprint Zusammenfassung

Der Sprint verlief sehr gut. Basierend auf unserer Sprint-Velocity können wir davon ausgehen, dass wir den Endtermin voraussichtlich etwas schneller als geplant erreichen werden.

Ein Bild, das Text, Quittung, Screenshot, Reihe enthält.

KI-generierte Inhalte können fehlerhaft sein.

## Sprint 3

### Sprintplanung

**Dauer: <13.03.2025 – 09.04.2025>**

|  |  |
| --- | --- |
| **User Story** | **Beschreibung** |
| DB | Kommunikation und Speicherung/Laden von Fahrstrecken |

**Anzahl Story-Points:** 65 Story-Points

**Ausgewählte Punkte aus der Impediment Liste:**

**DB:**

* Form der Datenspeicherung bestimmen
* Konfiguration & Setup der Datenbank
* Hosting der Datenbank
* Kommunikation mit der Datenbank

### Sprint Demo

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

KI-generierte Inhalte können fehlerhaft sein.

**Zusätzlich angefangene / erledigte User Stories:**

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

KI-generierte Inhalte können fehlerhaft sein.

### Spike Story:

* **Ziel des Spikes:** Erforschung und Entwicklung eines Prototyps für den Discovery Mode des Mbot, um Hindernisse zu erkennen und zu umgehen.
* **Timeboxing:** Ein fester Zeitraum wurde gesetzt, um den Prototypen zu entwickeln und Lösungen für die Hindernisvermeidung zu testen.
* **Prototypen / Variantenbildung:**
  + Verschiedene Ansätze wurden getestet, darunter:
    - BFS (Breitensuche)
    - A\* (A\*-Algorithmus)
    - Neander Förmig (ein weniger strukturierter Ansatz)
* **Lösungsansatz:** Alte Ansätze wurden verworfen, um auf einen neuen, von Herrn Eigner vorgeschlagenen Ansatz zurückzugreifen.
* **Ergebnisse:** Der Discovery Mode konnte erfolgreich implementiert werden, Hindernisse wurden korrekt erkannt und umfahren.

### Sprint Retrospektive

Die Arbeitsatmosphäre während dieses Sprints war insgesamt positiv und von einem ruhigen, konzentrierten Miteinander geprägt. Herausforderungen traten insbesondere in der Abstimmung mit dem Frontend-Team sowie im Bereich des Discovery-Mode-Moduls auf (vgl. Spike-Story). Die Kommunikation zwischen den beteiligten Teams war teilweise unklar, was zu Missverständnissen und Verzögerungen führte.

Für den nächsten Sprint wurde beschlossen, den sprachlichen Austausch gezielter zu verbessern – etwa durch regelmäßige kurze Sync-Meetings oder klar definierte Kommunikationskanäle. Zudem soll durch die stärkere Abgrenzung und Isolation einzelner Arbeitsprozesse die Eigenverantwortung innerhalb der Teams gestärkt und Schnittstellenprobleme reduziert werden.

### Sprint Zusammenfassung

Der Sprint verlief insgesamt erfolgreich. Es gab einige Herausforderungen im Zusammenhang mit dem Discovery Mode, die jedoch erfolgreich gelöst werden konnten. Basierend auf unserer Sprint-Velocity gehen wir davon aus, dass wir den Endtermin voraussichtlich etwas früher als ursprünglich geplant erreichen werden.

## Sprint 4

### Sprintdauer

**Dauer: <10.04.2025 – 30.04.2025>**

|  |  |
| --- | --- |
| **User Story** | **Beschreibung** |
| Fahrstreckenlogik | Korrektes Laden vorherig gespeicherter Fahrstrecken & Möglichkeit zum Empfangen manuell erstellter Routen |
| Kartographie-Modul | Sensordaten sammeln und verarbeiten. Aufbauend auf der Spike-Story Breakthrough & Testing implementieren. |

**Anzahl Story-Points:** 170 Story-Points

**Ausgewählte Punkte aus der Impediment Liste:**

**Fahrstreckenlogik:**

* Verifizieren der Anbindung an ein DB-System
* Speichern der Fahrstrecken
* Abrufen von den gespeicherten Fahrstrecken
* Speicherung & Verarbeitung manueller Routen
* Switch Option zwischen Free-Roam + Kartographie und Steuerung

**Kartographie-Module / Discovery-Mode:**

* Sammeln der Daten (Sensor)
* Aufarbeitung der empfangenen Daten
* Bereitstellung & Speicherung der Daten

### Sprint Demo



XXX KARTOGRAPHIE NOCH EINFÜGEN XXX

### Sprint Retrospektive

Die Arbeitsatmosphäre während dieses Sprints war insgesamt positiv und von einem ruhigen, konzentrierten Miteinander geprägt. Herausforderungen traten insbesondere in der Abstimmung mit dem Frontend-Team sowie im Bereich des Discovery-Mode-Moduls auf (vgl. Spike-Story). Die Kommunikation zwischen den beteiligten Teams war teilweise unklar, was zu Missverständnissen und Verzögerungen führte.

Für den nächsten Sprint wurde beschlossen, den sprachlichen Austausch gezielter zu verbessern – etwa durch regelmäßige kurze Sync-Meetings oder klar definierte Kommunikationskanäle. Zudem soll durch die stärkere Abgrenzung und Isolation einzelner Arbeitsprozesse die Eigenverantwortung innerhalb der Teams gestärkt und Schnittstellenprobleme reduziert werden.

### Sprint Zusammenfassung

Der Sprint verlief insgesamt erfolgreich. Es gab einige Herausforderungen im Zusammenhang mit dem Discovery Mode, die jedoch erfolgreich gelöst werden konnten. Basierend auf unserer Sprint-Velocity gehen wir davon aus, dass wir den Endtermin voraussichtlich etwas früher als ursprünglich geplant erreichen werden.

# Installation / Software Deployment

Anleitung welche Schritte notwendig sind um das fertige SW Produkt zu installieren und in Betrieb zu nehmen.

# Projektabschluß

## Projektzusammenfassung

Zusammenfassung der Projektdurchführung. Was lief gut/schlecht. Welche Erkenntnisse wurden während der Durchführung des Projektes gewonnen. Was würde man, nun anders machen bzw. wieder gleich machen?

## Attachments

Tabellarische Auflistung der Projektdateien.

z.B.: ZIP-File mit dem Quellcode, Projektpräsentationen, ...