**Logo_Uni_1Universität Bremen**

FB4 Produktionstechnik

Software-Projekt

**Entwicklung einer Web-Applikation zur Erkennung von Zusammenhängen in den Daten aus Einzelblattmontagen**

Von

Nepomuk Bense

Sascha Bollmann

Mehmet Erdede

Zelgai Nemati

Alexander Siewert

Studiengang: Systems Engineering

Betreuender Professor: Prof. Dr.-Ing Klaus-Dieter Thoben

Betreuer: M. Sc. Maria Teresa Alvela Nieto

M. Sc. Aljoscha Sander

Abgabedatum: 31. August 2021

**BIK Institut für integrierte Produktentwicklung**

Prof. Dr.-Ing. Klaus-Dieter Thoben

**Erklärung**

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht.

Bremen, den 31. August 2021 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name, Vorname

Bremen, den 31. August 2021 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name, Vorname

Bremen, den 31. August 2021 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name, Vorname

Bremen, den 31. August 2021 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name, Vorname

Bremen, den 31. August 2021 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name, Vorname

**Inhaltsverzeichnis**

[1 Einleitung 5](#_Toc78390706)

[2 Web-Applikation 6](#_Toc78390707)

[2.1 Technologieauswahl 6](#_Toc78390708)

[2.2 Architektur 11](#_Toc78390709)

[2.3 Versions-Management 12](#_Toc78390710)

[2.4 Integrationsautomatisierung (CI/CD) 14](#_Toc78390711)

[2.5 Webseite Funktionalität 16](#_Toc78390712)

[3 Maschinelles Lernen 17](#_Toc78390713)

[4 Fazit 18](#_Toc78390714)

[5 Literaturverzeichnis 19](#_Toc78390715)

**Symbole und Abkürzungen**

**Abkürzungen**

OSBI Offshore single blade installation

UML Unified Modelling Language

WEB Webseite bzw. Internetseite

**Formelzeichen**

**Zeichen Einheit Erklärung**

mm² Probenquerschnitt

# Einleitung

Offshore Wind liefert bereits heute einen substanziellen Teil des Energiemixes. Heutzutage werden die meisten Offshore-Windenergieanlagen komponentenweise installiert und um die Kosten weiter zu senken, ist es insbesondere notwendig, den Installationprozess zu verbessern. Die Montage der Blätter stellt dabei die größte Herausforderung dar, denn hier ist hohe Präzision und Sorgfalt erforderlich, um die Blattschraube in den Nabenflansch einzusetzen. Wind und Wellen üben Last auf die Strukturen aus und die daraus resultierenden Relativbewegungen zwischen Turm und den Rotorblättern erschwert die Blattmontage. Überschreitet die Relativbewegung einen bestimmten Schwellenwert, kann die Installation nicht mehr durchgeführt werden und es kommt zu einer kostspieligen Verzögerung. Basierend auf den Messdaten, die während der Installation des Windparks in der Nordsee aufgezeichnet wurden, werden Methoden des maschinellen Lernens verwendet, um den Zusammenhang zwischen Umgebungsvariablen wie Windgeschwindigkeit und der Beschleunigung von Turm, Gondel und Rotorblättern zu identifizieren und prognostizieren.

Ziel des Softwaretechnikprojektes ist die Erkennung von Zusammenhängen in Messdaten aus der Einzelblattmontage. Hierzu soll eine WEB-Applikation entwickelt werden, die basierend auf aufgezeichneten Daten, Zusammenhänge berechnet, damit Ingenieure diese Ergebnisse nutzen können, um ein besseres Verständnis der Rotorblattmontage und die gegebenen Umweltbedingungen zu erhalten.

# Web-Applikation

## Technologieauswahl

Die Technologieauswahl zu Beginn eines Software-Projektes bestimmt den weiteren Projektverlauf [Sch09]. Um dieser Entscheidung gerecht zu werden, bedarf es dem Einsatz einer systematischen Technologieanalyse. Im Rahmen dieser Projektarbeit wurde der folgende Ansatz gewählt, der sich in der Softwaretechnik Literatur etabliert hat [Sam08] [Das05] [Ahu05]:

1. Erstellen eines Kriterienkatalogs
2. Anwenden der Kriterien auf die zu untersuchenden Technologien
3. Bewertung und Gewichtung der einzelnen Ergebnisse
4. Entscheidung auf dieser Grundlage treffen

Diese Systematik zur Entscheidungsfindung wird folgend mehrmals praktisch angewandt, da moderne Webanwendungen aus einem Kanon verschiedener Technologien bestehen. Um diesen Sachverhalt zu verdeutlichen, kann das Beispiel herangezogen werden, dass eine abstrakte Datenbanksprache wie MySQL nicht dafür verwendet werden kann benutzerfreundliche Oberflächen zu designen, da diese Sprache für einen anderen Zweck programmiert wurde und somit nicht mit einer Stylesheet-Sprache wie CSS verglichen werden kann.

Tabelle 2‑1 Entscheidungsmatrix Frontend

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Frontend** | | | | | | | |
| **Technologie**  **Optionen** | | **Beliebtheit** | **Konnektivität** | **Teaminterne Erfahrung** | **Funktions-umfang** | **Usability** | **Geschwindigkeit** |
| **Markup Language** | **HTML** |  |  |  |  |  |  |
| **HAML** |  |  |  |  |  |  |
| **restructuredText** |  |  |  |  |  |  |
| **Stylesheet Language** | **CSS** |  |  |  |  |  |  |
| **SASS** |  |  |  |  |  |  |
| **Tailwind CSS** |  |  |  |  |  |  |
| **Template Library** | **Bootstrap** |  |  |  |  |  |  |
| **Material UI** |  |  |  |  |  |  |
| **Frontend Framework** | **React.JS** |  |  |  |  |  |  |
| **Angular** |  |  |  |  |  |  |
| **Vue.JS** |  |  |  |  |  |  |

Der Tabelle 2-1 ist die Entscheidungsgrundlage für die benötigten Frontend Technologien zu entnehmen. Es wird eine Markup-Sprache benötigt, welche das Grundgerüst der Homepage bilden soll, eine Stylesheet Sprache, welche das Grundgerüst in ein benutzerfreundliches UI wandelt, eine Template Bibliothek welche die benötigten Icons, Schriftarten und vordefinierte Komponenten zur Verfügung stellt und ein Frontend Framework, welches die statische Homepage in eine dynamische wandelt. Diesen Untergruppen sind verschiedene lila markierte Optionen zugeordnet. Es soll eine Option pro Untergruppe für den weiteren Projektverlauf gewählt werden. Die gewählten Technologien haben wir in der Tabelle 2-3 zusammengefasst.

Tabelle 2‑2 Entscheidungsmatrix Backend

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Backend** | | | | | | | |
| **Technologie**  **Optionen** | | **Beliebtheit** | **Konnektivität** | **Teaminterne Erfahrung** | **Funktions-umfang** | **Usability** | **Geschwindigkeit** |
| **Authentifizierung** | **Firebase Authentification** |  |  |  |  |  |  |
| **JWT** |  |  |  |  |  |  |
| **Node.JS Passport** |  |  |  |  |  |  |
| **Datenbank** | **MySQL** |  |  |  |  |  |  |
| **MongoDB** |  |  |  |  |  |  |
| **API** | **GraphQL** |  |  |  |  |  |  |
| **REST** |  |  |  |  |  |  |
| **Neuronales Netzwerk** |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Die Entscheidungsgrundlage für die Backend Technologien ist der Tabelle 2-2 zu entnehmen. Es wird eine Authentifizierungsschnittstelle benötigt, welche die Benutzerdaten verwalten soll, eine Datenbank, welche die hochgeladenen Datensätze speichert, eine API-Technologie, um die vom Benutzer getätigten Anfragen z.B. an die Datenbank zu überprüfen und ein Framework zum Programmieren des neuronalen Netzwerks. Auch hier wurden die ausgewählten Technologien in der Tabelle 2-4 zusammengefasst.

Tabelle 2‑3 Ausgewählte Technologien Frontend

|  |  |  |
| --- | --- | --- |
| **Technologie** | **Funktion** | **Kurzbeschreibung** |
| **HTML5**  Other html 5 Icon | Plex Iconset | Cornmanthe3rd | Homepage Grundgerüst | HTML5 ist die aktuellste Version die von jedem Internetbrowser unterstützt wird und dient zur Strukturierung von Internetseiten. Nicht nur die Konnektivität bzw. Kompatibilität, sondern auch der Funktionsumfang dieser Markup Sprache sind dafür verantwortlich, dass Sie sich hoher Beliebtheit erfreut. |
| **SASS** | Styling Benutzeroberfläche | SASS steht für „Syntactically Awesome Stylesheets“, ist ein Präprozessor für die Stylesheet-Sprache CSS und erweitert diese um z.B. Funktionen, Variablen und Schleifen. Da alle etablierten Internetbrowser lediglich CSS verarbeiten können, wird der geschrieben SASS Code vom mitinstallierten Compiler nach jeder Codeänderung in CSS-Code übersetzt. |
| **Material UI**  Material Ui Icon - Lade PNG und Vektor kostenlos herunter | Icons und Schriftarten | Material UI ist eine vom Unternehmen Google entwickelte Bibliothek. Sie enthält diverse Komponenten, Schriftarten und Icons, die einer bestimmten Designkonvention entsprechen. Somit wird sichergestellt, dass alle Komponenten der Webseite einheitlich designed sind. |
| **React.JS** | Dynamische Funktionen | React ist die weltweit meistbenutzte Bibliothek zum Erstellen von Benutzeroberflächen und ist in JavaScript programmiert. Dem Programmierer ist es mithilfe dieser Bibliothek möglich einzelne wiederverwendbare Komponenten zu programmieren, um somit redundantem Code vorzubeugen. |

|  |  |  |
| --- | --- | --- |
| **Technologie** | **Funktion** | **Kurzbeschreibung** |
| **Firebase Authentification**  download logo firebase icon svg eps png psd ai vector color free - el fonts  vectors | Authentifizierungs  Datenbank | Um hohe Sicherheitsstandards einzuhalten, verwendet die Webseite ein tokenbasiertes Authentifizierungssystem. Firebase Auth. ist eine vom Unternehmen Google zur Verfügung gestellte Authentifizierungsschnittstelle, die jedem Benutzer einen individuellen Token zuweist. Diese Tokens werden dann im lokalen Browserspeicher des jeweiligen Benutzers (mithilfe vom einem SSL Skript) abgelegt. |
| **MongoDB**  How to Install MongoDB on CentOS and cPanel - ServerNoobs | Datenverwaltung | MongoDB ist eine sogenannte NoSQL Datenbank. Die interne Struktur ist somit nicht durch statische Schlüsselwerte bestimmt und die hinterlegten Objekte sind in Ihrer Wertzuweisung flexibel. Dadurch kann die Datenverarbeitung der Webseite stetig angepasst werden, ohne den internen Datenbankaufbau ändern zu müssen. |
| **GraphQL**  Exploring GraphQL – Rohan Varma – Software Engineer @ Facebook | Schnittstellen-Programmierung  (API) | GraphQL ist eine neue Technologie zum Programmieren von Systemschnittstellen (APIs) und stellt eine Alternative zu REST dar. Im Rahmen dieses Softwareprojekts wurden alle APIs, die der Abbildung 2-1 zu entnehmen sind, mit GraphQL programmiert. |
|  | Neuronales Netzwerk |  |

Tabelle 2‑4 Ausgewählte Technologien Backend

## Architektur

Die UML (Unified Modelling Language) ist eine Modellierungssprache, bzw. eine Familie von Notationen und Diagrammtypen, die dazu verwendet wird, die interne Struktur von Systemen abzubilden [Fow04]. Zu dieser Familie von Diagrammtypen gehört auch das Anwendungsfalldiagramm (auf Englisch Use-Case-Diagram). Die im Rahmen einer Architekturanalyse erstellten Anwendungsfalldiagramme, zeigen das nach außen sichtbare Verhalten des Systems bzw. der einzelnen Systemkomponenten. Dadurch können die Kommunikationswege zwischen den einzelnen Subsystemen besser nachvollzogen werden. [Rup12]

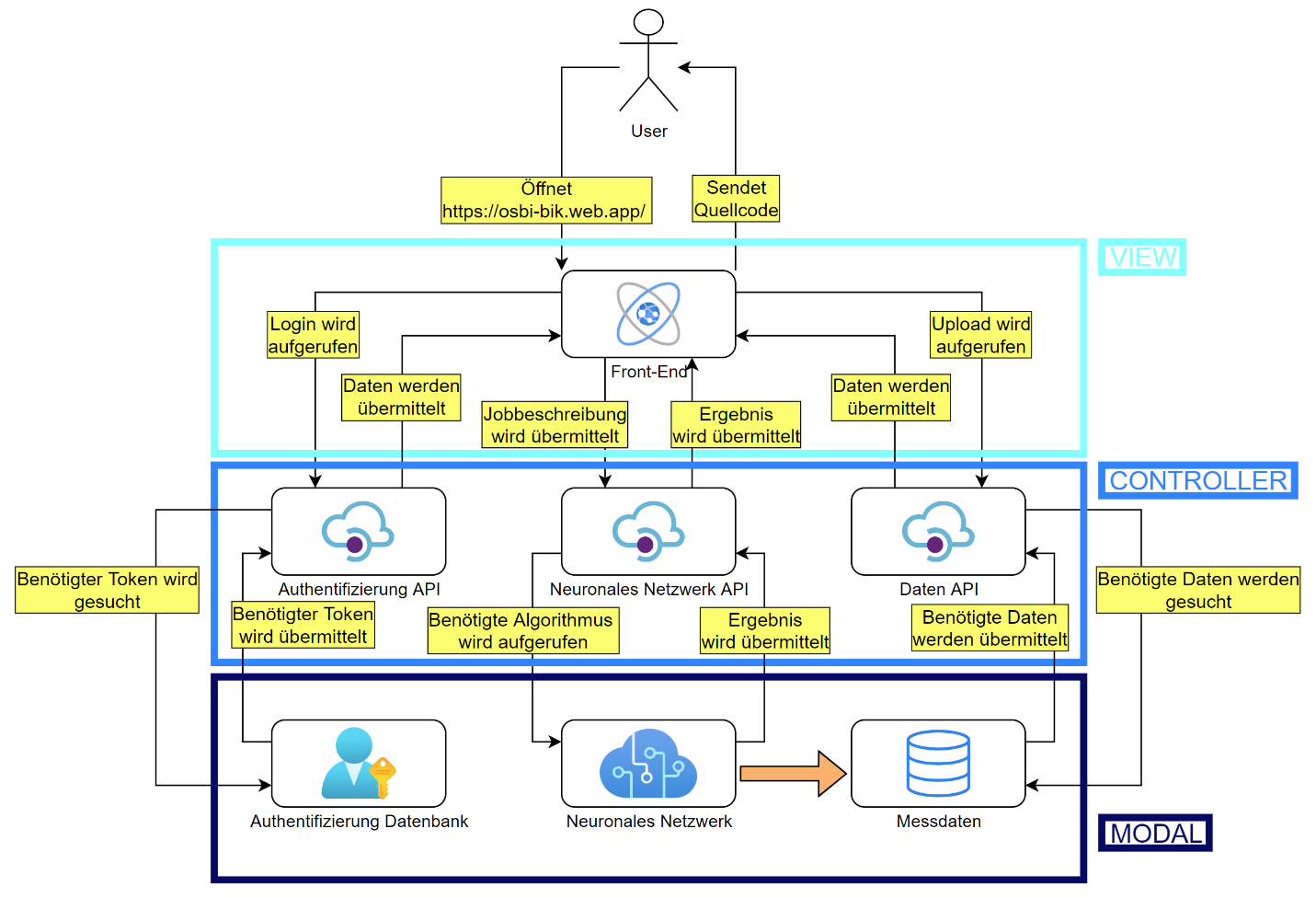
Der Abbildung 2-1 ist die Architektur der programmierten Webseite als Anwendungsfalldiagramm zu entnehmen. MVC (Modal-View-Controller) ist eine Dreischichtenarchitektur im Bereich der Webprogrammierung, bei der die Daten (Modal) durch gesicherte Schnittstellen (Controller) verwaltet und dem Benutzer gesondert präsentiert werden (View). [Kal14]

Abbildung 2‑1 Webseite Architektur

## Versions-Management

Unter dem Begriff Versions-Management, versteht man die Verwaltung von Quellcodeänderungen, bzw. den Vergleich zwischen unterschiedlichen Quellcodezuständen. Dadurch ist es möglich frühere Stände des Quellcodes wieder herzustellen, wenn z.B. ein Softwarefehler in der neusten Version entdeckt wurde. Außerdem wird die Transparenz des Entwicklungsprozesses erhöht, indem zusätzlich festgehalten wird, wann und von wem die Änderungen gemacht wurden. Softwaretechnikern ist es dann auf Grundlage dieser Daten möglich verschiedenste Metriken zu erheben, um somit den Entwicklungsprozess optimieren zu können. Eine weitere Grundfunktion des Versions-Managements ist das Branch-and-Merge-System. Durch die Möglichkeit Quellcodezustände zu vergleichen können beliebig viele Programmierer an unterschiedlichen Zuständen (Branches) bzw. Dateien arbeiten und dann Ihre Änderungen zusammenfügen (Mergen). Dieses Vorgehen ist der Abbildung 2-2 zu entnehmen, bei der 2 Programmierer an unterschiedlichen Zuständen arbeiten und Ihre Änderungen dann zusammenfügen. [Aug18]

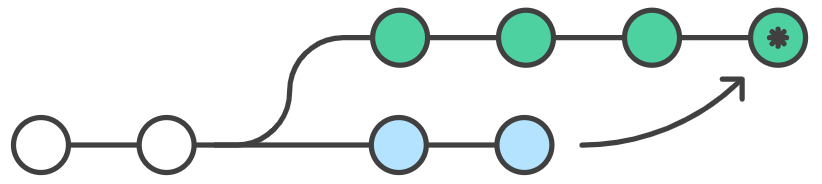


Abbildung 2‑2 Branch-and-Merge-System

Quelle: https://cdn-media-1.freecodecamp.org/images/VonhijTBQgjwtRXz31wLzF7iWDnDFk2o8EWi

Um Versions-Management zu betreiben, benötigt man ein Versionskontrollsystem (auf Englisch Version-Control-System). Im Rahmen dieser Projektarbeit wird Git in Kombination mit Github als solches System verwendet. Dabei ist Git das eigentliche Versionskontrollsystem und Github ein Cloud Anbieter zur Verwaltung eines solchen Datenbestandes. Dieser Datenbestand ist öffentlich zugänglich (Open Source) und kann unter folgender URL aufgerufen werden:

**https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation**

Um diese grundlegend qualitätssichernden Aspekte des Versions-Managements zu erweitern, kann die Branchstruktur optimiert werden. Die Branchstruktur, in diesem Softwareprojekt, besteht aus einem Hauptbranch (Master) und zwei hierarchisch angeordneten Testumgebungen (Staging 1 und 2), welches der Abbildung 2-3 zu entnehmen ist. Im Kapitel 2.4 wird dann die Interaktion bzw. Kommunikation zwischen diesen Branches, im Kontext der Integrationsautomatisierung, thematisiert.

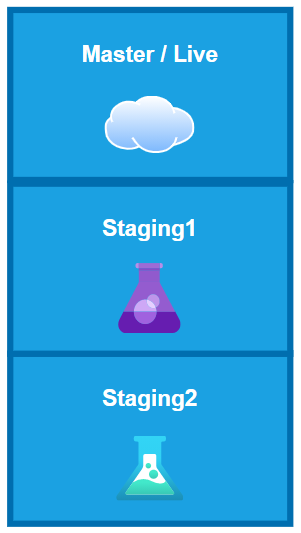


Abbildung 2‑3 Branchstruktur

Die hier erwähnte Branchstruktur (des Softwareprojekts), kann unter folgendem Link aufgerufen und konfiguriert werden:

**https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation/branches**

## Integrationsautomatisierung (CI/CD)

Unter Integrationsautomatisierung versteht man die Überwachung des kompletten Software-Lebenszyklus. Diese besteht aus zwei Kernkonzepten, nämlich der kontinuierlichen Integration (Continuous Integration) und der kontinuierlichen Serveraktualisierung (Continuous Deployment). Bei der kontinuierlichen Integration (CI) wird nach jeder bestätigten Codeänderung und jedem Merge ein branchabhängiges Skript ausgeführt, welche unter dem folgenden Link aufgerufen werden können:

**https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation/tree/main/.github/workflows**

Die Skripte bestehen aus mehreren Arbeitspaketen (Tasks) die in chronologischer Reihenfolge abgearbeitet werden. Der Abbildung 2-4 sind die Tasks der drei Skripte zu entnehmen:

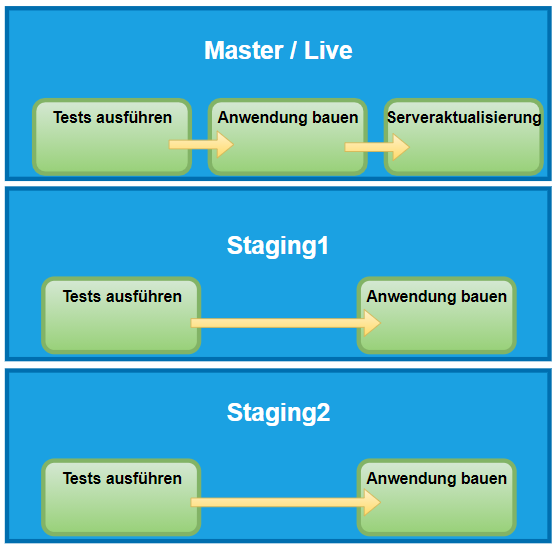


Abbildung 2‑4 CI/CD Taskübersicht

Dieses Konzept stellt sicher, dass die Korrektheit jeder isolierten Softwarekomponente gegeben ist, da alle programmierten Tests (bei der Skriptausführung) ausgeführt werden und die Anwendung gebaut wird, um syntaktische Fehler auszuschließen. Die korrekte Interaktion zwischen einzelnen Komponenten wird dadurch aber nicht gewährleistet, weshalb Integrationstests notwendig sind. Diese Integrationstests werden auf den beiden Testumgebungen in explorativer Form durchgeführt und ggf. freigegeben oder abgelehnt. Wenn durch die Integrationstests keine Fehler gefunden wurden und die Codeänderung freigegeben wurde, wird sie auf den darüberliegenden Branch gemerged. Falls bei den Integrationstests Fehler gefunden wurden, wird die Codeänderung auf den darunterliegenden Branch gemerged und der Fehler kann vom Programmierer behoben werden. Dieser Prozess kann der Abbildung 2-5 entnommen werden.

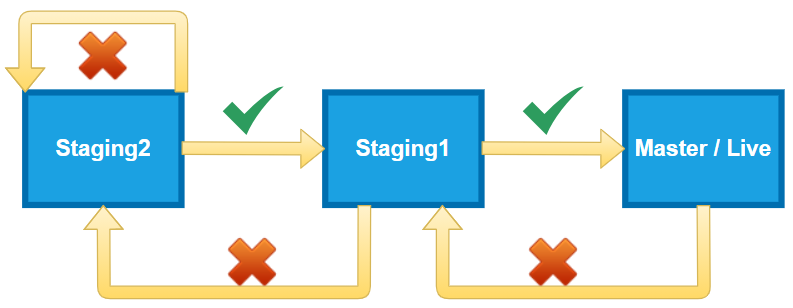


Abbildung 2‑5 Software-Lebenszyklus

Die kontinuierliche Serveraktualisierung (CD) soll sicherstellen, dass die öffentlich erreichbare Webseite aktuell ist und dem letzten Stand des Master-Branches entspricht, weshalb Sie nur als Task im Master-Skript zu finden ist.

## Webseite Funktionalität

Die im Rahmen dieser Projektarbeit programmierte Webseite ist unter folgender URL erreichbar:

**https://osbi-bik.web.app/**

Initial wird einem die Landing-Page angezeigt, welche die Funktionalität der Webseite zusammenfasst und einige Features (wie z.B. das SSL Zertifikat) hervorhebt. Nachdem der Login Prozess erfolgreich durchgeführt wurde, wird das Dashboard angezeigt. Der Aufbau des Dashboards ist der Abbildung 2-6 zu entnehmen.

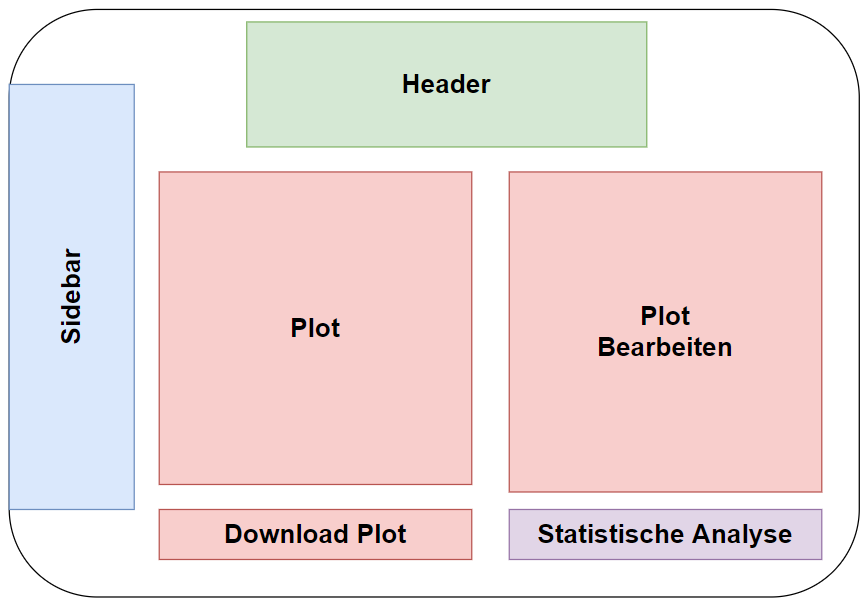


Abbildung 2‑6 Webseite Aufbau

Im Header kann man die zu analysierenden Daten hochladen oder Testdaten herunterladen, um die Funktionalität des Dashboards zu testen. Auf der linken Seite befindet sich die Sidebar, welche die Hilfesektion enthält und unter anderem zur öffentlich zugänglichen Dokumentation weiterleitet.

Im Zentrum den Dashboards befindet sich der aus den hochgeladenen Daten erzeugte Plot, der individuell bearbeitet und heruntergeladen werden kann. Außerdem befindet sich in der unteren rechten Ecke des Dashboards eine Sektion, in der die wichtigsten statistischen Daten zusammengefasst sind.

# Maschinelles Lernen

# Fazit

Die im Rahmen dieser Projektarbeit geforderten Funktionalitäten (der WEB-Applikation) und die Realisierung dieser sind der Tabelle 4-1 zu entnehmen.

Nicht nur wurde die Mindestanforderung für das Bestehen des Projektes erfüllt, sondern darüber hinaus weitere Funktionen implementiert. Da unsere Gruppe insgesamt wenig Vorwissen im Bereich des maschinellen Lernens bzw. der Datenanalyse hatte und trotz der, während des kompletten Projektverlaufs, geltenden Covid19 Kontakteinschränkungen, konnten wir uns gemeinsam als Gruppe in diese thematische Herausforderung einarbeiten.

Ein enger teaminterner Austausch, ein strukturiertes Zeitmanagement, eine klare Rolleneinteilung und Agilität waren dabei ausschlaggebend für den erfolgreichen Projektverlauf.

Tabelle 4‑1 Projektanforderungen

|  |  |  |  |
| --- | --- | --- | --- |
| Anforderungen / Features | Nicht erfüllt | Erfüllt | Übertroffen |
| Lokal Lauffähige Homepage |  |  | Häkchen mit einfarbiger Füllung  Homepage ist öffentlich erreichbar,  SSL Zertifikat |
| Code ist Open Source  auf GitHub |  |  | Häkchen mit einfarbiger Füllung  Vollständige Dokumentation,  Integrationsmanagement |
| 2 NN Methoden implementieren |  |  |  |
| Ansprechendes User Interface |  |  | Häkchen mit einfarbiger Füllung  Homepage ist Responsive |
| Authentifizierungssystem  Registrieren / Einloggen |  |  | Häkchen mit einfarbiger Füllung  Tokenbasiertes System,  SSL Zertifikat |
| Datenupload |  |  | Häkchen mit einfarbiger Füllung  Testdaten verfügbar,  Upload in Echtzeit |
| Grafische Darstellung und interaktive Bearbeitung hochgeladener Daten |  |  | Häkchen mit einfarbiger Füllung  Verschiedene Plottypen,  Bearbeitung in Echtezeit, |
| Speichern von Plots |  | Häkchen mit einfarbiger Füllung |  |

# Literaturverzeichnis

**[Sch09] Schmidt, M. 2009,** Software-Metriken als Mittel zur Technologieauswahl am Beispiel zweier Implementierungen einer Web-Applikation in Rails und JEE. (Seite 53)

**[Sam08] Samkari, K. 2008,** Comparison matrix for web hci. Information and Communications Technologies: From Theory to Applications. (Seite 1-5)

**[Das05] Daschiel, S. 2005,** Evaluierung von Web-frameworks, Erstellung eines Prototyps und Konzeption der multimedialen Wissensdokumentation für die Kunstuniversität linz. Master's thesis

**[Ahu05] Ahuja, S. 2005,** Comparison of web services technologies from a developer's perspective. Proceedings of the International Conference on Information Technologie: Coding and Computing. (Seite 791-792)

**[Fow04] Fowler, M. 2004,** UML konzentriert, 3. Auflage (Seite 21)

**[Rup12] Rupp, C. 2012,** UML2 glasklar, 4. Auflage (Seite 242)

**[Kal14] Kalelkar, M. 2014,** Implementation of Modal-View-Controller Architecture Pattern for Business Intelligence Architecture (Seite 1)

**[Aug18] Augsten, S. 2018,** Was ist Versionskontrolle, dev-insider.de/was-ist-versionskontrolle-a-701050/, aufgerufen am 23.07.2021