**Fallstudie**

**IPWA01-01**

**Aufgabenstellung 1**

**co2-Footprint**

von

**Kevin Faller**

Klosterstr. 30

77723 Gengenbach

Bearbeitungszeitraum: 1. November 2022 – 22. Juni 2023

GitHub <https://github.com/KevFaller>

Wissenschaftliche Kursleitung Prof. Dr. Marian Benner-Wickner

Tutor Paul Libbrecht

**Vorwort**

Dies ist eine Fallstudie zum Nachweis der Prüfungsleistung für das Modul IPWA02-01. Recht herzlich bedanken möchte ich mich bei Prof. Dr. Marian Banner-Wickner, sowie bei meinem Tutor Paul Libbrecht, die die Betreuung meiner Fallstudie übernommen haben.

Vielen Dank!

**Eidesstattliche Erklärung**

Hiermit versichere ich eidesstattlich, dass die vorliegende Fallstudie von mir selbstständig und ohne unerlaubte fremde Hilfe angefertigt worden ist, insbesondere, dass ich alle Stellen, die wörtlich oder annähernd wörtlich oder dem Gedanken nach aus Veröffentlichungen, unveröffentlichten Unterlagen und Gesprächen entnommen worden sind, als solche an den entsprechenden Stellen innerhalb der Arbeit durch Zitate kenntlich gemacht habe, wobei in den Zitaten jeweils der Umfang der entnommenen Originalzitate kenntlich gemacht wurde. Ich bin mir bewusst, dass eine falsche Versicherung rechtliche Folgen haben wird.



(Datum, Unterschrift)

20.07.2023

Diese Fallstudie ist urheberrechtlich geschützt, unbeschadet dessen wird folgenden Rechtsübertragungen zugestimmt:

• der Übertragung des Rechts zur Vervielfältigung der Bachelor-Thesis für Lehrzwecke an der IU-Internationale Hochschule (§ 16 UrhG),

• der Übertragung des Vortrags-, Aufführungs- und Vorführungs-rechts für Lehrzwecke durch Professoren der IU-Internationale Hochschule (§ 19 UrhG),

• der Übertragung des Rechts auf Wiedergabe durch Bild- oder Ton-

träger an die IU-Internationale Hochschule (§21 UrhG).

**Zusammenfassung**

Die Webanwendung "Ghost Net Fishing" ermöglicht die Erfassung, Bergung und Meldung von Geisternetzen. Die Implementierung erfolgt durch die Verwendung von SQL-Statements und die Integration einer Datenbank.

Die MoSCoW-Methode wird verwendet, um die Anforderungen zu priorisieren. Must-have-Anforderungen werden als kritisch betrachtet und müssen erfüllt werden, während die Soll-, Kann- und Werden-Nicht-Anforderungen entsprechend ihrer Wichtigkeit bewertet werden. Dadurch wird sichergestellt, dass die Schlüsselkomponenten des Projekts fokussiert und die entscheidenden Anforderungen erfüllt werden.

Eine wichtige Funktion ist das Erfassen von Geisternetzen. Jeder kann anonym Geisternetze erfassen, ohne sich registrieren oder anmelden zu müssen. Dabei werden GPS-Daten und die Größe des Netzes angegeben, um die Lokalisierung und Bergung zu ermöglichen.

Bergende Personen können sich registrieren und anschließend Netze zur Bergung auswählen. Nach dem Einloggen wird durch die Sitzungsvariable "username" die Zuordnung der Verantwortlichkeiten ermöglicht. Dadurch kann nachvollzogen werden, wer sich für die Bergung eines bestimmten Netzes eingetragen hat.

Beliebige Personen können auch Netze als verschollen melden, ohne registriert oder eingeloggt zu sein. Hierfür werden ebenfalls SQL-Statements verwendet, um den Status der Netze entsprechend zu aktualisieren.

Die Klasse "ConnectionDB" stellt die Datenbankverbindung her und gewährleistet eine stabile Verbindung zur Datenbank.

Insgesamt trägt die Implementierung der Funktionalitäten in "Ghost Net Fishing" zur effektiven Erfassung, Bergung und Meldung von Geisternetzen bei. Die Verwendung von SQL-Statements, die Integration der Datenbank und die Umsetzung der MoSCoW-Methode ermöglichen einen reibungslosen Ablauf und eine zuverlässige Verwaltung der Daten. Das Projekt "Ghost Net Fishing" trägt somit zur Sicherung der Meeresumwelt vor den Gefahren durch Geisternetze bei.

Inhaltsverzeichnis

[1 Einleitung 6](#_Toc140153217)

[1.1 Motivation 6](#_Toc140153218)

[1.2 Zielsetzung 6](#_Toc140153219)

[1.3 Vorgaben 6](#_Toc140153220)

[2 Fallvorstellung 6](#_Toc140153221)

[3 Verwendete Technologien 7](#_Toc140153222)

[3.1 HTML und HTTP 7](#_Toc140153223)

[3.2 JavaServerPages 8](#_Toc140153224)

[3.3 Java Enterprise Edition 8](#_Toc140153225)

[3.4 CSS 8](#_Toc140153226)

[3.5 Maven 9](#_Toc140153227)

[3.6 Docker 9](#_Toc140153228)

[3.6.1 Docker Desktop 9](#_Toc140153229)

[3.6.2 Docker-Compose 9](#_Toc140153230)

[3.7 JavaScript 10](#_Toc140153231)

[3.8 GitHub 10](#_Toc140153232)

[3.9 IntelliJ IDEA 10](#_Toc140153233)

[4 Implementierung 10](#_Toc140153234)

[4.1 Geisternetze (anonym) erfassen 11](#_Toc140153235)

[4.2 Bergung eines Geisternetzes eintragen 12](#_Toc140153236)

[4.2.1 Registrieren 12](#_Toc140153237)

[4.2.2 Login 15](#_Toc140153238)

[4.2.3 Bergung bevorstehend 15](#_Toc140153239)

[4.3 Einsehen, welche Geisternetze noch zu bergen sind. 16](#_Toc140153240)

[4.4 Geisternetz als geborgen melden 17](#_Toc140153241)

[4.5 Netz als Verschollen Melden 18](#_Toc140153242)

[4.6 Datenbank Connection 18](#_Toc140153243)

[5 Fazit 19](#_Toc140153244)

[6 Literaturverzeichnis 22](#_Toc140153245)

# 1 Einleitung

## Motivation

Diese Fallstudie wurde im Rahmen meines Studiums zum Softwareentwicklers bei der IU – Internationale Hochschule als Prüfungsleistung für das Modul IPWA02-01 erstellt. Zur Überprüfung meiner Leistung habe ich mir die Aufgabe 3 „Ghost Net Fishing“ ausgesucht.

## Zielsetzung

Diese wissenschaftliche Arbeit hat das Ziel, eine Webanwendung zur Meldung und Bergung von Geisternetzen zu entwickeln. Das Projekt wird im Web-Development-Team der Non-Profit-Organisation "Shea Shepherd" durchgeführt. Die Anwendung soll es meldenden Personen ermöglichen, anonym Geisternetze zu erfassen, während bergende Personen sich für die Bergung eines Netzes eintragen und den Status der noch zu bergenden Geisternetze einsehen können. Zudem sollen bergende Personen die Möglichkeit haben, erfolgreich geborgene Netze zu melden.

Die Arbeit wird gemäß den Anforderungen des Product Backlogs und der MoSCoW-Methode durchgeführt. Im ersten Schritt wird ein öffentliches Code-Repository, wie beispielsweise GitHub, für das Projekt eingerichtet. Anschließend wurden diese 5 aufgaben aus dem Product Backlog ausgewaehlt und Implementiert

1. MUST: Als meldende Person möchte ich Geisternetze (anonym) erfassen können.

2. MUST: Als bergende Person will ich mich für die Bergung eines Geisternetzes eintragen können.

3. MUST: Als bergende Person möchte ich sehen, welche Geisternetze noch zu bergen sind.

4. MUST: Als bergende Person möchte ich Geisternetze als geborgen melden können.

5. COULD: Als beliebige Person möchte ich Geisternetze als verschollen melden können.

## Vorgaben

Zu Beginn der Fallstudie werden momentan verfügbare Technogien untersucht, welche sich am besten zur Umsetzung eignen. Vorgegeben ist die Verwendung von HTML, JSP, CSS, JavaEE und JavaScript. Das Verwenden von weiteren JavaScript-Programmbibliotheken wie React.js oder Angula.js wurde freigestellt, ebenso die zu verwendende Entwicklungsumgebung.

# Fallvorstellung

Das Web-Development-Team der Non-Profit-Organisation "Shea Sepherd" hat sich zum Ziel gesetzt, eine Web-App zur Meldung und Bergung von Geisternetzen zu entwickeln. Geisternetze sind herrenlose Fischernetze, die im Meer treiben und eine ernsthafte Bedrohung für die marine Umwelt darstellen. Die Organisation möchte diese Gefahr mindern und das Bewusstsein für das Problem erhöhen.

Der Requirements Engineer hat bereits vor Beginn des Projekts intensive Gespräche mit verschiedenen Stakeholdern, sowohl innerhalb der Organisation als auch in der Fischereibranche, geführt, um die Anforderungen und Bedürfnisse zu verstehen. Das Fachkonzept wurde entwickelt und enthält wichtige Informationen über Geisternetze. Ein Geisternetz hat Eigenschaften wie den Standort (GPS-Koordinaten), eine geschätzte Größe und einen Status.

Der Status eines Geisternetzes kann verschiedene Ausprägungen haben: Gemeldet, Bergung bevorstehend, Geborgen oder Verschollen. Meldende Personen können Geisternetze anonym erfassen, während bergende Personen sich für die Bergung eines Netzes eintragen und den Status der noch zu bergenden Geisternetze einsehen können. Eine wichtige Einschränkung besteht darin, dass Geisternetze immer nur einer bergenden Person zugeordnet werden können, um unnötige Bergungsfahrten und Missverständnisse zu vermeiden. Bergende Personen können jedoch mehrere Geisternetze gleichzeitig bergen.

Das Projektteam hat die Anforderungen gemäß der MoSCoW-Methode priorisiert. Es gibt Must-have-Anforderungen, die zwingend umgesetzt werden müssen, wie das Erfassen von Geisternetzen, das Eintragen zur Bergung, das Anzeigen verfügbarer Geisternetze und das Melden von geborgenen Netzen. Zudem gibt es eine Could-have-Anforderung, die es beliebigen Personen ermöglicht, Geisternetze als verschollen zu melden.

Das Projekt ist agil konfiguriert, und du bist Teil eines interdisziplinären Teams von Entwickler:innen. Die Aufgabenstellung umfasst die Einrichtung eines öffentlichen Code-Repositories, die Auswahl von fünf Anforderungen aus dem Product Backlog für die prototypische Umsetzung im ersten Sprint, die Entwicklung eines Oberflächenprototypen mit Schwerpunkt auf persistenter Datenhaltung sowie die Dokumentation der Entwicklung der Webanwendung, einschließlich Design, Umsetzung, Datenbankstruktur und softwaretechnischer Architektur.

Der Product Owner erwartet einen funktionalen Prototypen mit einer pragmatischen Gestaltung der Nutzeroberfläche. Das Ergebnis muss keinen Design Award gewinnen, sondern vor allem die Anforderungen erfüllen und einen effektiven Beitrag zur Meldung und Bergung von Geisternetzen leisten.

# Verwendete Technologien

## HTML und HTTP

Die Auszeichnungssprache HTML ist neben der Stilsprache CSS und der Programmiersprache JavaScript eine der drei groben Sprachen, die für die Frontend-Entwicklung wichtig sind. (Ackermann, 2021, p. 47) *Das HyperText Transfer Protocoll* (HTTP), das ein Anwendungsprotokoll ist, definiert, wie Webserver und Webbrowser unterereinander kommunizieren. Das Protokoll selbst ist relativ einfach gehalten, um den Webserver möglichst wenig zu belasten. Die Kommunikation selbst setzt ein Transportprotokoll wie TCP/IP voraus. Eine Kommunikation über HTTP ist immer gleich aufgebaut (Herold, 2017, p. 478)

## JavaServerPages

JavaServer Pages (JSP) ist eine Java-Technologie, die zur Erstellung dynamischer Webseiten verwendet wird. Es ermöglicht die Integration von Java-Code direkt in den HTML-Code, wodurch eine nahtlose Verbindung zwischen dem Frontend und dem Backend hergestellt wird. JSP wird oft in Kombination mit JavaEE eingesetzt, da es die Erstellung komplexer, serverseitiger Anwendungen erleichtert.

Durch die Verwendung von JSP in JavaEE können Entwickler:innen die Leistungsfähigkeit der Java-Plattform voll ausschöpfen. Es bietet eine robuste Umgebung für die Entwicklung von unternehmenskritischen Anwendungen und unterstützt die Skalierbarkeit und Wiederverwendbarkeit von Code. Darüber hinaus erleichtert JSP die Trennung von Präsentationslogik und Geschäftslogik, was zu einer verbesserten Wartbarkeit und Erweiterbarkeit der Anwendung führt. Die Kombination von JSP und JavaEE bietet Entwickler:innen ein starkes Werkzeugset, um effiziente, sichere und zuverlässige Webanwendungen zu erstellen.

## Java Enterprise Edition

JavaEE (Java Enterprise Edition) ist eine umfangreiche Sammlung von Java-Technologien und APIs, die für die Entwicklung von skalierbaren, verteilten und unternehmenskritischen Anwendungen konzipiert sind. Es bietet eine robuste Plattform mit Funktionen wie Servlets, JavaServer Pages (JSP), Enterprise JavaBeans (EJB), Java Persistence API (JPA) und Java Message Service (JMS), die Entwickler:innen bei der Erstellung leistungsstarker und zuverlässiger Anwendungen unterstützen. JavaEE bietet eine standardisierte und stabile Entwicklungsumgebung, die bewährte Muster und bewährte Praktiken für die Unternehmensentwicklung nutzt. Es erleichtert die Entwicklung von Anwendungen, die hohe Anforderungen an Sicherheit, Skalierbarkeit und Zuverlässigkeit erfüllen müssen. JavaEE wird häufig in Unternehmen eingesetzt, um Enterprise-Anwendungen zu entwickeln, die die Anforderungen an komplexe Geschäftsprozesse und Datenmanagement erfüllen und gleichzeitig die Flexibilität und Wiederverwendbarkeit von Java nutzen.

## CSS

HTML beschreibt, was dargestellt werden soll, […]. Aber es beschreibt nicht, wie die Information dem Benutzer präsentiert werden soll. Über so genannte Cascading Style Sheets (CSS) kann dem Browser mitgeteilt werden, in welchen Schriftstil, -größe, -farbe, Hintergrund, Ausrichtung und Abstand der jeweilige Text darstellen soll. Teilweise kann auch noch das Verhalten bei Benutzerinteraktion modifiziert werden. (Herold, 2017, p. 479)

## Maven

Maven ist ein leistungsstarkes Build-Management-Tool für die Java-Entwicklung, das die Automatisierung des Build-Prozesses und die Verwaltung von Abhängigkeiten ermöglicht. Es bietet eine standardisierte Projektstruktur, in der Entwickler:innen ihre Projekte organisieren können, sowie eine zentrale Konfigurationsdatei (pom.xml), die Informationen über das Projekt, seine Abhängigkeiten und Build-Einstellungen enthält. Durch die Verwendung von Maven können Entwickler:innen effizientere Build-Prozesse erstellen, Abhängigkeiten automatisch verwalten und die Wiederverwendbarkeit von Code fördern.

## Docker

Docker ist eine Open-Source-Plattform, die es Entwickler:innen ermöglicht, Anwendungen in Containern zu verpacken und bereitzustellen. Container bieten eine konsistente und isolierte Umgebung, in der Anwendungen unabhängig von der zugrunde liegenden Infrastruktur ausgeführt werden können, was die Portabilität und Skalierbarkeit von Anwendungen erleichtert.

### Docker Desktop

Docker Desktop ist eine benutzerfreundliche Desktop-Anwendung, die es Entwickler:innen ermöglicht, Docker-Container auf ihren lokalen Systemen auszuführen und zu verwalten. Es bietet eine intuitive Benutzeroberfläche, um Container zu erstellen, zu starten, zu stoppen und zu überwachen, und ermöglicht somit eine einfache Entwicklung und Testung von Anwendungen in einer Containerumgebung.

### Docker-Compose

Docker Compose ist ein Tool, das es ermöglicht, mehrere Docker-Container als Teil einer Anwendung zu definieren und zu orchestrieren. Es bietet eine einfache Möglichkeit, komplexe Anwendungen mit verschiedenen Komponenten wie Datenbanken, DB-Management-Tools und Webservern zu konfigurieren und zu verwalten.

Docker Compose ermöglicht es, die gesamte Infrastruktur einer Anwendung in einer einzigen Konfigurationsdatei zu definieren, was die Wartung und Bereitstellung erleichtert. Durch die Verwendung von Docker Compose können Entwickler:innen die Container für die einzelnen Komponenten einer Anwendung mit nur einem Befehl erstellen, starten und stoppen.

Die Verwendung von Docker Compose ist besonders vorteilhaft in wissenschaftlichen Arbeiten, in denen komplexe Anwendungen mit mehreren Komponenten benötigt werden. Es vereinfacht die Einrichtung und den Betrieb von Entwicklungsumgebungen, indem es ermöglicht, alle erforderlichen Dienste und Tools, wie Datenbanken und Webserver, mit wenigen Schritten zu starten und zu verbinden. Dadurch können Entwickler:innen effizienter arbeiten und sich auf ihre eigentlichen Forschungs- und Entwicklungsaktivitäten konzentrieren, anstatt sich mit der manuellen Einrichtung und Konfiguration der einzelnen Komponenten auseinandersetzen zu müssen.

## JavaScript

Reine HTML-Dokumente erlauben nur sehr begrenzte Interaktion mit dem Benutzer, über Hyperlinks zu andere Dokumenten sowie über Formulare. Dabei steckt die ganze Logik hinter der Interaktion in der Implementierung des Servers. Schon Ende 1995 wurde erkannt, dass für eine bessere Bedienung eine echte Programmiersprache nötig ist, und mit JavaScript eine solche entwickelt. Trotz der Namensähnlichkeit hat JavaScript nichts mit der Sprache Java zu tun. (Herold, 2017, p. 480)

## GitHub

GitHub setzt sich aus zwei Worten zusammen, Git und Hub. Git ist eine Versionsverwaltungssoftware, entwickelt von dem Linuxerfinder Linus Torvalds. Hub wird ein Ort oder eine Logik bezeichnet Beispiele dafür sind Verteilzentren, Verkehrsknotenpunkte oder Güterverteilzentren. Das Besondere daran ist, dass man seinen Quellcode nicht nur verwalten, sondern auch anderen Menschen zur Verfügung stellen kann. GitHub steht so für Verwalten und Teilen.

## IntelliJ IDEA

IntelliJ IDE ist eine integrierte Entwicklungsumgebung (IDE), die von JetBrains entwickelt wurde und hauptsächlich für die Java-Programmierung verwendet wird. Sie bietet eine umfangreiche Palette an Funktionen und Werkzeugen zur effizienten Entwicklung, Fehlerbehebung und Codeverwaltung. Mit seiner intuitiven Benutzeroberfläche und Unterstützung für verschiedene Technologien erleichtert IntelliJ IDEA Entwickler:innen die Erstellung qualitativ hochwertiger Software.

# Implementierung

Beim Vorgehen nach der MoSCoW-Methode beginnt die Priorisierung mit der Identifizierung und Gewichtung der Must-have-Anforderungen. Diese Anforderungen sind von kritischer Bedeutung für den Erfolg des Projekts und müssen unbedingt erfüllt werden, um das definierte Ziel zu erreichen. Anschließend werden die Soll-, Kann- und Werden-Nicht-Anforderungen entsprechend ihrer relativen Wichtigkeit und Relevanz für das Projekt bewertet und priorisiert. Diese systematische Priorisierung ermöglicht es, Ressourcen und Aufmerksamkeit auf die Schlüsselkomponenten des Projekts zu konzentrieren und sicherzustellen, dass die entscheidenden Anforderungen erfüllt werden.

1. MUST: Als meldende Person möchte ich Geisternetze (anonym) erfassen können.

2. MUST: Als bergende Person will ich mich für die Bergung eines Geisternetzes eintragen können.

3. MUST: Als bergende Person möchte ich sehen, welche Geisternetze noch zu bergen sind.

4. MUST: Als bergende Person möchte ich Geisternetze als geborgen melden können.

5. COULD: Als beliebige Person möchte ich Geisternetze als verschollen melden können.

## Geisternetze (anonym) erfassen

Unter der URL http://localhost:8080/netz/netzBearbeiten besteht die Möglichkeit, Geisternetze zu melden oder den Status eines Geisternetzes zu ändern. Um ein Geisternetz zu melden, wählt die Person zunächst im Top-Down-Menü den Punkt "Melden" aus und gibt anschließend die GPS-Daten sowie die geschätzte Größe des Netzes an. Diese Informationen ermöglichen es, das Geisternetz zu lokalisieren und entsprechende Maßnahmen einzuleiten.



Abbildung 1

Die Methode "doPost" implementiert die Funktionalität des HTTP POST-Requests im Kontext der Klasse "netzBearbeiten". Dabei werden die übergebenen Parameter aus dem Formular, insbesondere der Status, abgerufen und protokolliert. Wenn der Status den Wert "melden" hat, werden zusätzlich die GPS-Daten und die Größe des Geisternetzes aus dem Formular extrahiert und ein SQL-Statement zum Einfügen der Daten erstellt. Eine Verbindung zur Datenbank wird hergestellt, ein vorbereitetes Statement mit dem SQL-Statement wird erstellt und schließlich das Statement ausgeführt, um die Daten in die Datenbank einzufügen.

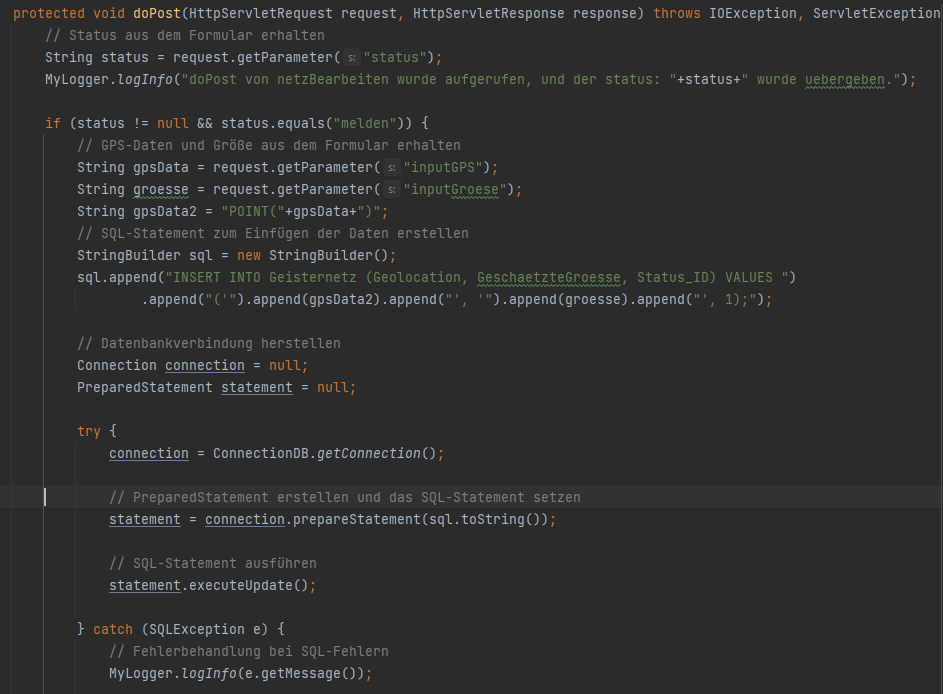


Abbildung 2

Diese Implementierung zeigt, wie die Daten aus dem Formular verarbeitet und in die Datenbank eingefügt werden. Sie folgt dabei bewährten Praktiken, wie dem Einsatz von vorbereiteten Statements zur Vermeidung von SQL-Injection und der Nutzung einer Verbindungspooling-Methode zur effizienten Verwaltung der Datenbankverbindungen.

Gemäß der Anforderung des Product Owners im ersten Muss-Kriterium, "Als meldende Person möchte ich Geisternetze (anonym) erfassen können", wurde festgelegt, dass es nicht erforderlich ist, sich anzumelden oder einzuloggen, um ein Netz zu melden. Dies ermöglicht es den meldenden Personen, ihre Informationen anonym zu halten und den Meldungsprozess unkompliziert und zugänglich zu gestalten. Durch diese Entscheidung wird die Benutzerfreundlichkeit der Webanwendung verbessert und Hürden für die Erfassung von Geisternetzen abgebaut.

## Bergung eines Geisternetzes eintragen

### Registrieren

Im Rahmen der Funktionalität, sich als bergende Person einzutragen, ist es erforderlich, sich zuerst zu registrieren. Die Registrierung erfolgt über die Seite http://localhost:8080/netz/login, wo Benutzer auf den "Create Account" Button klicken können, um zur Registrierungsseite zu gelangen. Dort können sie ihre erforderlichen Informationen eingeben und einen Account erstellen, um anschließend als bergende Person aktiv am Projekt teilnehmen zu können. Diese Vorgehensweise gewährleistet die Identifikation und Authentifizierung bergender Personen, um eine geordnete und sichere Bergung der Geisternetze zu ermöglichen.

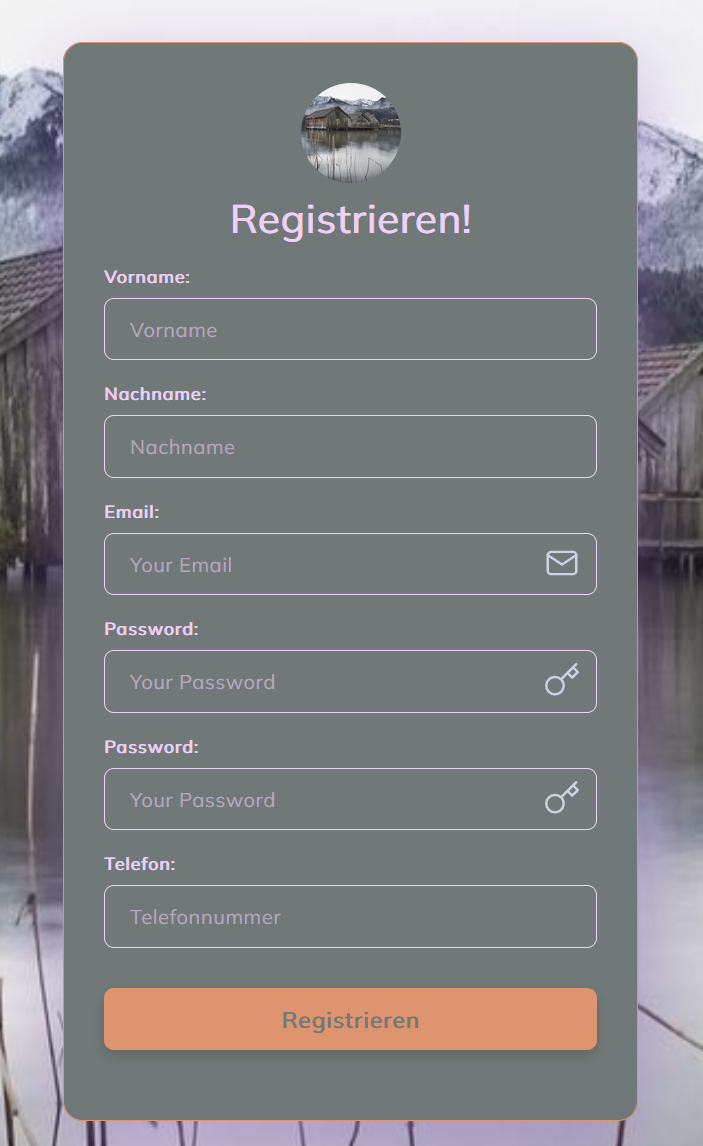


Abbildung 3

Im Formular zur Registrierung sind alle erforderlichen Daten einzufüllen, um die Person zu erfassen. Dabei wird bereits während der Eingabe überprüft, ob das eingegebene Passwort mit der Bestätigung übereinstimmt. Sollte dies nicht der Fall sein, erhält der Benutzer eine Benachrichtigung, um auf den Fehler hingewiesen zu werden und das Passwort korrekt einzugeben. Durch diese Validierung wird sichergestellt, dass die Daten korrekt und vollständig eingegeben werden, um eine erfolgreiche Registrierung der Person zu gewährleisten.

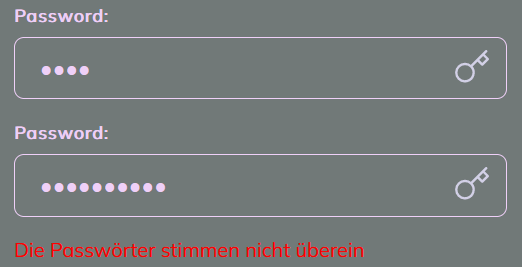


Abbildung 4

In der Funktion mit dem Namen "checkPasswordMatch()" enthält. Diese Funktion dient dazu, die Übereinstimmung der eingegebenen Passwörter in den Feldern "rPassword" und "rPassword2" zu überprüfen. Falls die Passwörter nicht übereinstimmen, wird ein Fehlerhinweis mit der ID "passwordMatchError" angezeigt, andernfalls bleibt der Fehlerhinweis ausgeblendet. Diese Funktion trägt dazu bei, sicherzustellen, dass die Passwörter korrekt eingegeben und bestätigt werden, um eine fehlerfreie Registrierung zu gewährleisten.



Abbildung 5

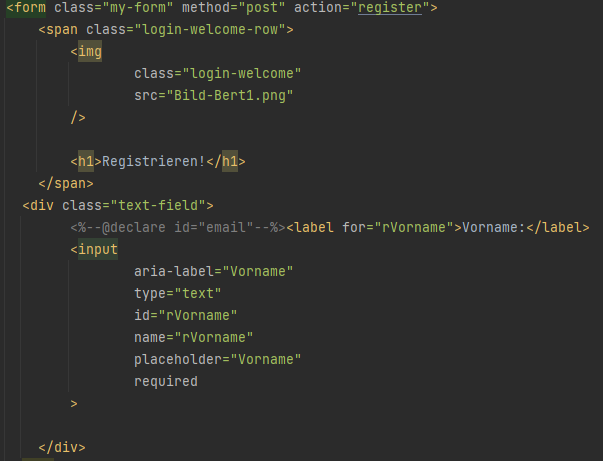


Abbildung 6

Der HTML-Code (Abbildung 6) repräsentiert ein Formular mit der Klasse "my-form" und der Methode "post". Durch die Angabe der Aktion "register" wird festgelegt, wohin die Formulardaten gesendet werden sollen. Das Formular umfasst eine Willkommenszeile, die ein Bild und die Überschrift "Registrieren!" enthält. Darunter befindet sich ein Textfeld für den Vornamen, das durch das Label "Vorname" gekennzeichnet ist. Es ist erforderlich, dass das Feld ausgefüllt wird, da das "required"-Attribut angegeben ist. Diese Struktur des Codes ermöglicht es Benutzer:innen, ihre Vorname-Informationen bequem und intuitiv im Formular einzutragen.

Nachdem der Benutzer auf die Schaltfläche "Registrieren" geklickt hat, werden die eingegebenen Daten persistierend in die Datenbank in der Tabelle "Person" geschrieben. Dieser Vorgang gewährleistet, dass die registrierten Informationen langfristig gespeichert werden und für zukünftige Verwendungszwecke abrufbar sind. Durch die Datenpersistenz wird die Integrität der Benutzerdaten gewährleistet und ermöglicht eine zuverlässige Verwaltung

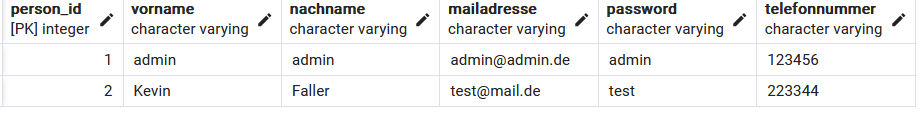


Abbildung 7

### Login

Unter der URL http://localhost:8080/netz/login gelangt der Benutzer zum Login-Bereich der Webanwendung. Nachdem er sich erfolgreich mit seiner E-Mail-Adresse und seinem Passwort authentifiziert hat, wird eine Sitzungsvariable namens "username" erstellt und gesetzt. Diese Sitzungsvariable ermöglicht es während der laufenden Sitzung, den Benutzer zu identifizieren und relevante Informationen beizubehalten. Durch den Einsatz von Sitzungsvariablen wird eine effektive Verwaltung des Sitzungszustands und der Daten innerhalb der Webanwendung ermöglicht. Die erfolgreiche Setzung der Sitzungsvariable wird durch die Protokollierung mittels "MyLogger.logInfo" dokumentiert. Abbildung 8 zeigt den erfolgreichen Prozess der Sitzungsvariablen-Setzung.

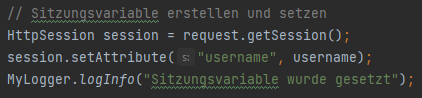


Abbildung 8

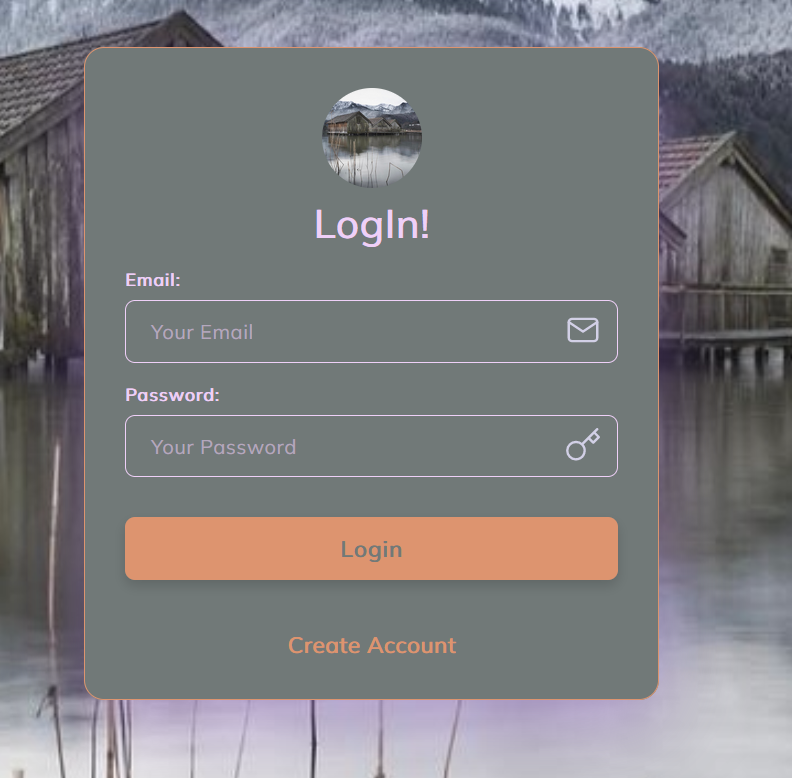


Abbildung 9

### Bergung bevorstehend

Nachdem sich der Benutzer erfolgreich eingeloggt hat, wird er zur Startseite (index.jsp) weitergeleitet. Auf dieser Seite kann der Benutzer über den Button "Geisternetze-Melden/Bergen" die Funktion "Bergung bevorstehend" im Top-Down-Menü (Abbildung 1) auswählen und anschließend auf "Bearbeiten" klicken (Abbildung 10), um entsprechende Aktionen durchzuführen. Dies ermöglicht dem Benutzer die gezielte Bearbeitung von bevorstehenden Bergungen von Geisternetzen und trägt zur effektiven Nutzung der Anwendung bei.

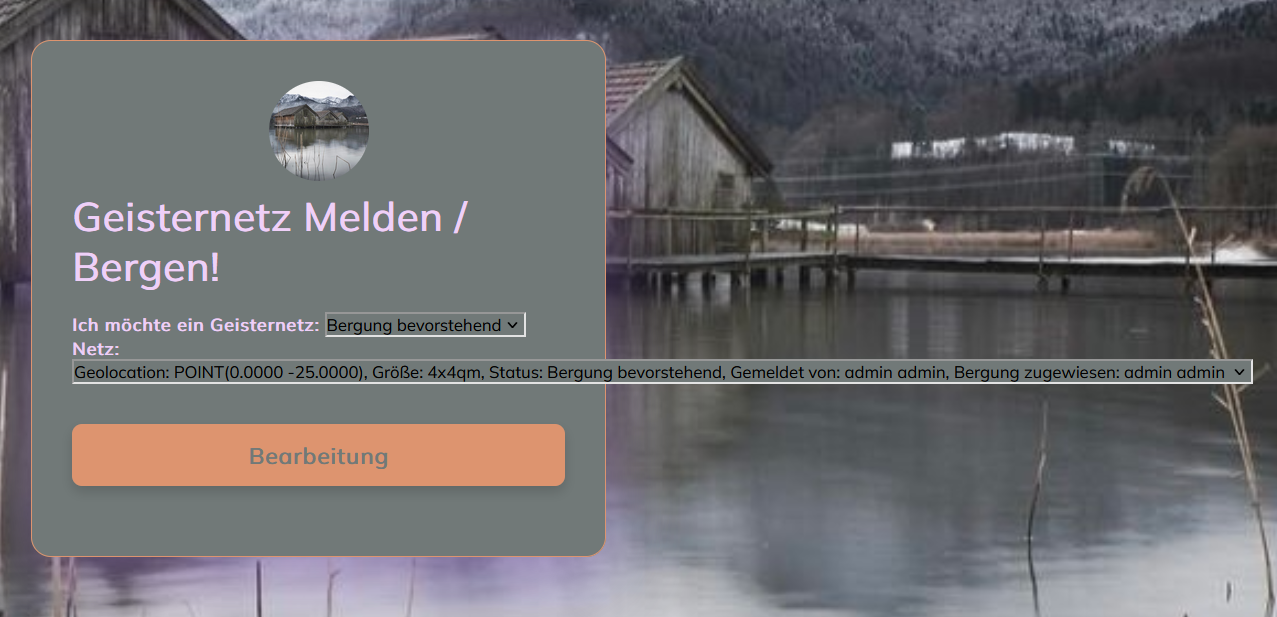


Abbildung 10

Die effektive Nutzung besteht darin, dass in der Tabelle (Abbildung 11), in der die Geisternetze aufgelistet sind, deutlich erkennbar ist, wer sich zur Bergung eines bestimmten Netzes eingetragen hat. Dies wird ermöglicht, indem die Sitzungsvariable nach dem Einloggen gesetzt wird und anhand dieser nachvollzogen werden kann, welche Person den Status auf "Bergung bevorstehend" gesetzt hat. Dadurch wird eine eindeutige Zuordnung der Verantwortlichkeiten ermöglicht und eine effiziente Organisation der Bergungsaktivitäten gewährleistet.

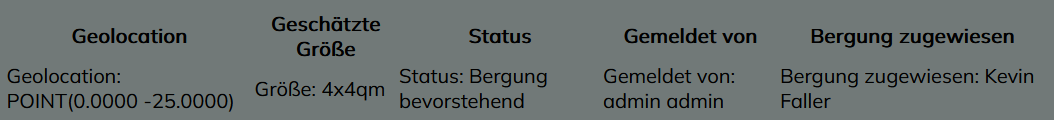


Abbildung 11

## Einsehen, welche Geisternetze noch zu bergen sind.

Unter der Adresse http://localhost:8080/netz/netz können Informationen eingesehen werden, welche Netze noch zu bergen sind, welche als verschollen gemeldet wurden und welche Benutzer sich zum Bergen eines Netzes bereiterklärt haben. Diese Übersicht ermöglicht einen reibungslosen Ablauf und eine effiziente Koordination der Bergungsaktivitäten, indem alle relevanten Informationen zentralisiert und zugänglich gemacht werden. Durch die Bereitstellung dieser Informationen wird die Transparenz und Zusammenarbeit innerhalb des Projekts "Ghost Net Fishing" gefördert.

Die Umsetzung erfolgt durch den Aufruf der Seite http://localhost:8080/netz/netz, wodurch das doGet-Ereignis der Netz.jsp ausgelöst wird (Abbildung 12). Dieses Ereignis ruft wiederum die Funktion "getGeisternetzArray()" der Klasse "netzArrayCreator" auf. Dadurch wird eine Verbindung zur Datenbank hergestellt und das Geisternetz-Array erstellt, das die erforderlichen Informationen über die Netze enthält. Dieser Ablauf ermöglicht es, die Daten dynamisch zu laden und auf der Netzseite anzuzeigen, um einen aktuellen Überblick über die vorhandenen Netze zu erhalten.

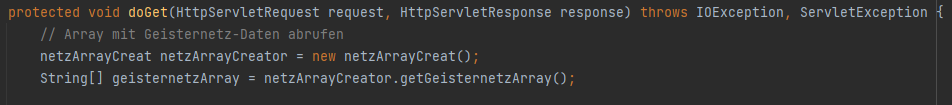


Abbildung 12

Innerhalb der Funktion "getGeisternetzArray()" wird eine SQL-Abfrage erstellt (Abbildung 13), um die relevanten Daten aus der Datenbank abzurufen. Die zurückgegebenen Daten werden in einem Array gespeichert und anschließend an die Klasse "Netz.java" zurückgegeben. Dieser Datenfluss ermöglicht es, die abgerufenen Informationen effizient zu verarbeiten und in der Netzseite darzustellen.

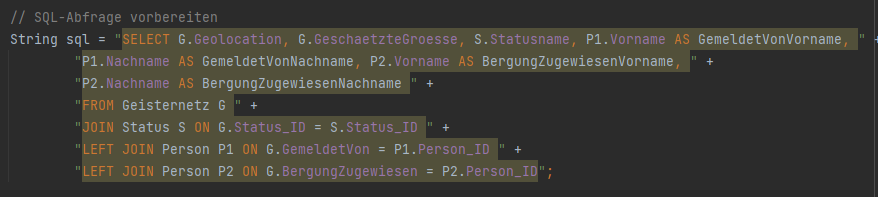


Abbildung 13

## Geisternetz als geborgen melden

Nachdem der Benutzer sich erfolgreich eingeloggt hat (Abschnitt 4.2.2) und zur Seite "Geisternetz-Melden/Bergen" navigiert hat, hat er die Möglichkeit, im Top-Down-Menü den Status "Bergen" auszuwählen. Dadurch wird ein weiteres Top-Down-Menü angezeigt, in dem die verfügbaren Netze aufgelistet sind. Der Benutzer kann sein geborgenes Netz auswählen und auf "Bearbeiten" klicken. Dadurch wird das ausgewählte geborgene Netz aus der Geisternetz-Datenbank gelöscht, um die erfolgreiche Bergung zu markieren und eine Aktualisierung der Datenbank vorzunehmen. Diese Funktionalität ermöglicht eine effektive Verwaltung und Aktualisierung der Geisternetz-Datenbank basierend auf den durchgeführten Bergungsaktionen.

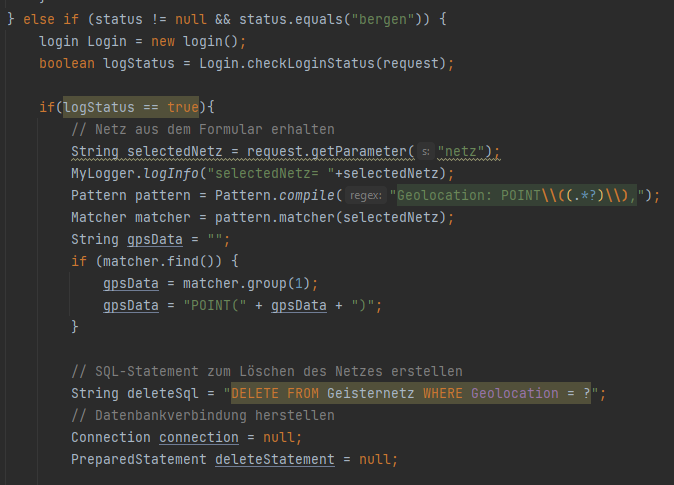


Abbildung 14

Das Löschen des Netzes wird durch ein SQL-Statement ermöglicht (Abbildung 14), das in der Klasse "netzBearbeiten.java" erstellt wird. Wenn der Status "bergen" ist, wird überprüft, ob der Benutzer eingeloggt ist. Wenn dies der Fall ist, wird das ausgewählte Netz aus dem Formular extrahiert und ein SQL-Statement zum Löschen des Netzes erstellt. Die entsprechende Geolocation des Netzes wird aus dem ausgewählten Netz extrahiert und als Parameter im SQL-Statement verwendet. Anschließend wird das SQL-Statement ausgeführt, um das ausgewählte Netz aus der Datenbank zu entfernen.

## Netz als Verschollen Melden

Als beliebiger Benutzer oder Besucher der Web-App habe ich die Möglichkeit, im Bereich "Geisternetze Melden/Bergen" auch Netze als verschollen zu melden, ohne registriert oder eingeloggt zu sein. Diese Funktion wird ähnlich wie in Abschnitt 4.4 implementiert, indem SQL-Statements erstellt werden. Im Code-Abschnitt "else if (status != null && status.equals("verschollen"))" wird das ausgewählte Netz aus dem Formular extrahiert und die entsprechende Geolocation ermittelt. Anschließend wird ein SQL-Statement erstellt, um den Status des Netzes auf "verschollen" zu aktualisieren. Die Datenbankverbindung wird hergestellt und das SQL-Statement wird ausgeführt, um die Aktualisierung durchzuführen. Dadurch wird eine benutzerfreundliche Möglichkeit geboten, verschollene Geisternetze zu melden und den Status entsprechend zu aktualisieren.

## Datenbank Connection

Die Klasse „ConnectionDB“ stellt die Datenbankverbindung für die Anwendung her. Dabei werden die notwendigen Datenbankinformationen wie die URL, der Benutzername und das Passwort festgelegt. Die statische Methode "getConnection()" liefert eine Verbindung zur Datenbank zurück, während die private Methode "createConnection()" die eigentliche Verbindung erstellt. Hierbei wird der PostgreSQL-Treiber geladen und die Verbindung mit den angegebenen Daten hergestellt. Im Falle von Fehlern bei der Verbindung werden entsprechende Fehlermeldungen in den Protokollen ("MyLogger.logInfo") ausgegeben. Diese Klasse ermöglicht es, eine stabile und zuverlässige Verbindung zur Datenbank herzustellen und ist entscheidend für den Zugriff auf die erforderlichen Daten.

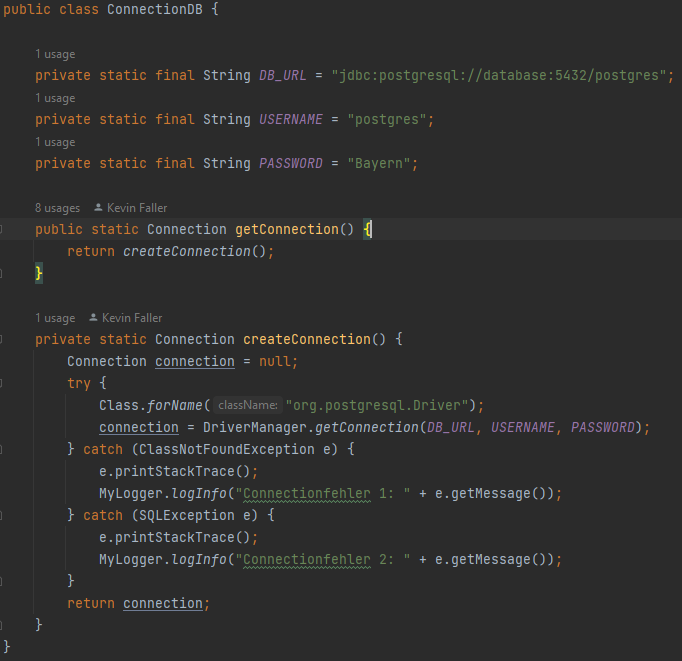


Abbildung 15

# Fazit

Die implementierte Funktionalität in der Webanwendung "Ghost Net Fishing" ermöglicht es den Benutzern, Geisternetze zu erfassen, zu bergen und als verschollen zu melden. Diese Funktionen werden durch das Erstellen und Ausführen von SQL-Statements sowie die Verwendung von Sitzungsvariablen zur Identifikation und Verwaltung der Benutzer realisiert.

Bei der Entwicklung der Webanwendung wurde die MoSCoW-Methode verwendet, um die Anforderungen zu priorisieren. Die Must-have-Anforderungen, die von kritischer Bedeutung für den Erfolg des Projekts sind, werden dabei besonders hervorgehoben und müssen unbedingt erfüllt werden, um das definierte Ziel zu erreichen. Die Soll-, Kann- und Werden-Nicht-Anforderungen werden entsprechend ihrer relativen Wichtigkeit und Relevanz bewertet und priorisiert. Dieser Ansatz ermöglicht es, Ressourcen und Aufmerksamkeit auf die Schlüsselkomponenten des Projekts zu konzentrieren und sicherzustellen, dass die entscheidenden Anforderungen erfüllt werden.

Eine der Must-have-Funktionen ist das Erfassen von Geisternetzen. Hierfür kann eine beliebige Person, auch ohne Registrierung oder Anmeldung, die Seite "http://localhost:8080/netz/netzBearbeiten" aufrufen. Dort kann die Person im Top-Down-Menü den Punkt "Melden" auswählen und die GPS-Daten sowie die geschätzte Größe des Netzes eingeben. Diese Informationen ermöglichen es, das Geisternetz zu lokalisieren und entsprechende Maßnahmen zur Bergung einzuleiten. Die eingegebenen Daten werden anschließend in der Datenbank persistiert.

Die Bergung eines Geisternetzes kann von bergenden Personen durchgeführt werden. Hierfür müssen sie sich zuerst registrieren, was über die Seite "http://localhost:8080/netz/login" möglich ist. Nach erfolgreicher Registrierung und Authentifizierung können die bergenden Personen die Funktion "Bergung bevorstehend" im Top-Down-Menü auswählen und auf "Bearbeiten" klicken. Dadurch gelangen sie zur Seite "index.jsp", auf der sie Netze auswählen können, die noch zu bergen sind.

Die effektive Nutzung der Anwendung liegt darin, dass in der Tabelle, in der die Geisternetze aufgelistet sind, deutlich erkennbar ist, wer sich zur Bergung eines bestimmten Netzes eingetragen hat. Dies wird ermöglicht, indem nach dem Einloggen eine Sitzungsvariable gesetzt wird und anhand dieser nachvollzogen werden kann, welche Person den Status auf "Bergung bevorstehend" gesetzt hat. Dadurch wird eine eindeutige Zuordnung der Verantwortlichkeiten ermöglicht und eine effiziente Organisation der Bergungsaktivitäten gewährleistet.

Des Weiteren können beliebige Personen auch Netze als verschollen melden, ohne registriert oder eingeloggt zu sein. Dies wird ähnlich wie die Bergungsfunktion umgesetzt, indem SQL-Statements erstellt werden. Durch Auswahl des Status "verschollen" im Top-Down-Menü und Klicken auf "Bearbeiten" können Personen ein Netz als verschollen markieren.

Für die Kommunikation mit der Datenbank wurde die Klasse "ConnectionDB" implementiert. Sie stellt die notwendige Datenbankverbindung her, indem die entsprechenden Informationen wie die URL, der Benutzername und das Passwort festgelegt werden. Die statische Methode "getConnection()" liefert eine Verbindung zur Datenbank zurück, während die private Methode "createConnection()" die eigentliche Verbindung herstellt. Dabei wird der PostgreSQL-Treiber geladen und die Verbindung mit den angegebenen Daten aufgebaut. Fehler bei der Verbindung werden protokolliert, um bei auftretenden Problemen entsprechend reagieren zu können.

Die Implementierung der Funktionalitäten in der Webanwendung "Ghost Net Fishing" ermöglicht einen reibungslosen Ablauf der Geisternetz-Erfassung, Bergung und Meldung von verschollenen Netzen. Durch die Verwendung von SQL-Statements und die Integration der Datenbank werden die Daten zuverlässig und effizient verwaltet. Die Umsetzung der Priorisierung nach der MoSCoW-Methode gewährleistet, dass die entscheidenden Anforderungen erfüllt werden, um das definierte Ziel des Projekts zu erreichen. Die erfolgreiche Implementierung dieser Funktionen trägt zur erfolgreichen Durchführung des Projekts "Ghost Net Fishing" bei und unterstützt den Schutz der Meeresumwelt vor den Gefahren durch Geisternetze.

# Abbildungsverzeichnis

[Abbildung 1 11](#_Toc140153389)

[Abbildung 2 12](#_Toc140153390)

[Abbildung 3 13](#_Toc140153391)

[Abbildung 4 13](#_Toc140153392)

[Abbildung 5 14](#_Toc140153393)

[Abbildung 6 14](#_Toc140153394)

[Abbildung 7 15](#_Toc140153395)

[Abbildung 8 15](#_Toc140153396)

[Abbildung 9 15](#_Toc140153397)

[Abbildung 10 16](#_Toc140153398)

[Abbildung 11 16](#_Toc140153399)

[Abbildung 12 17](#_Toc140153400)

[Abbildung 13 17](#_Toc140153401)

[Abbildung 14 18](#_Toc140153402)

[Abbildung 15 19](#_Toc140153403)

# Literaturverzeichnis

Öggl, B., & Kofler, M. (2022). *Git - Projektverwaltung für Entwickler und DevOps-Teams.* Rheinwerk.

Ackermann, P. (2021). *Webentwicklung.* Rheinwerk Compting.

Herold, L. W. (2017). *Grundlagen der Informatik.* Pearson.

**8 Anhangsverzeichnis**

1. Inbetriebnahmeanleitung