



**Handelskammer Bremen**  
für Bremen und Bremerhaven

Dokumentation zur betrieblichen Projektarbeit  
— Mittelstufenprojekt —

## **BotHQ – ein Discord-Bot-Framework**

Untertitel

Prüfungsausschuss: Bernd Heidemann  
Katrín Deeken

Abgabedatum: 29. Mai 2024

Prüfungsbewerber: Philipp Batelka  
Jan Mahnken  
Daniel Quellenberg  
Fabian Reichwald  
Justus Sieweke  
Christopher Spencer

Ausbildungsbetrieb: Europaschule Schulzentrum SII Utbremen  
Meta-Sattler-Str. 33  
28217 Bremen

## Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>III</b>
<b>Tabellenverzeichnis</b>	<b>IV</b>
<b>Quelltextverzeichnis</b>	<b>V</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Discord und Discord Bots . . . . .	1
1.2 Projektbeschreibung . . . . .	2
1.3 Projektziel . . . . .	2
<b>2 Projektumfeld</b>	<b>2</b>
2.1 Produktportfolio der HiTec GmbH . . . . .	2
2.2 Entstehung der Projektidee . . . . .	3
<b>3 Projektvorbereitung</b>	<b>3</b>
3.1 Ist-Analyse . . . . .	3
3.2 Soll-Analyse . . . . .	3
3.3 Projektziel detailliert . . . . .	3
3.4 Anforderungen und Technologien . . . . .	3
<b>4 Projektdurchführung</b>	<b>4</b>
4.1 Vorgehensmodell . . . . .	4
4.2 Umsetzung . . . . .	4
4.2.1 Klassenmodell . . . . .	4
4.2.2 Datenhaltung . . . . .	5
4.2.3 Design Patterns . . . . .	5
4.3 Qualitätssicherung . . . . .	5
4.3.1 Teststrategien . . . . .	5
4.3.2 Testfälle . . . . .	5
<b>5 Wirtschaftliche Betrachtung</b>	<b>5</b>
5.1 Marktuntersuchung . . . . .	5
5.1.1 Zielgruppe . . . . .	5
5.1.2 Marktvolumen und Marktpotential . . . . .	6
5.1.3 Konkurrenzanalyse . . . . .	6
5.2 Marketingmix (4P) . . . . .	6
5.2.1 Produktpolitik . . . . .	6
5.2.2 Preispolitik . . . . .	6
5.2.3 Kommunikationspolitik . . . . .	6
5.2.4 Distributionspolitik . . . . .	6

5.3	Kostenplanung . . . . .	7
5.3.1	Personalkosten . . . . .	7
5.3.2	Sachmittelkosten . . . . .	7
5.4	Wirtschaftlichkeitsberechnung . . . . .	7
5.4.1	Gewinnschwellenberechnung . . . . .	7
5.4.2	Amortisationsrechnung . . . . .	7
<b>6</b>	<b>Projektabschluss</b>	<b>7</b>
6.1	Erreichung des Projektziels . . . . .	7
6.2	Änderungen zur anfänglichen Planung . . . . .	8
6.3	Fazit . . . . .	8
<b>A</b>	<b>Anhang</b>	<b>VI</b>
A.1	Glossar . . . . .	VI
A.2	Literaturverzeichnis . . . . .	VIII
A.3	Quellcode . . . . .	IX

## Abbildungsverzeichnis

## Tabellenverzeichnis

## Quelltextverzeichnis

1	Beispielcode für einen ReadyListener in JDA . . . . .	IX
---	---	----

# 1 Einleitung

## 1.1 Discord und Discord Bots

Discord ist eine kostenlose Kommunikationsplattform, die ursprünglich für die Gaming-Community entwickelt wurde, mittlerweile jedoch von einer Vielzahl von Communities und Organisationen genutzt wird. Die Plattform bietet umfassende Funktionen für Text-, Sprach- und Video-Kommunikation sowie für die Organisation von Gruppen.

Ein grundlegendes Konzept von Discord ist der *Server*. Ein Server ist eine dedizierte Instanz, die von einer Gruppe von Benutzern genutzt wird, um zu kommunizieren und Inhalte zu teilen. Jeder Server kann mehrere *Kanäle* enthalten, die weiter in Text- und Sprachkanäle unterteilt sind. Textkanäle dienen der schriftlichen Kommunikation und dem Austausch von Dateien, während Sprachkanäle für Echtzeit-Audio-Gespräche verwendet werden.<sup>1</sup> Discord nutzt eine *Client-Server-Architektur* ▶<sup>2</sup>, bei der alle Daten über zentrale Server verarbeitet und gespeichert werden. Diese Server sind in verschiedenen Rechenzentren weltweit verteilt, um niedrige Latenzzeiten und hohe Verfügbarkeit zu gewährleisten. Die Kommunikation zwischen dem Discord-Client (verfügbar für Desktop, Web und Mobilgeräte) und den Servern erfolgt über das *HTTPS-Protokoll* ▶ und *WebSocket-Verbindungen* ▶.<sup>3</sup>

Ein besonders mächtiges Feature von Discord ist die Unterstützung von *Bots*. Bots sind automatisierte Programme, die über die *Discord API* ▶ interagieren können. Die API bietet eine Vielzahl von *Endpunkt* ▶, die es Entwicklern ermöglichen, Nachrichten zu senden, Benutzerinformationen abzurufen, Kanäle zu verwalten und auf Ereignisse zu reagieren. Bots werden häufig genutzt, um Server zu moderieren, Spiele zu integrieren, Musik abzuspielen und vieles mehr.

Die *Java Discord API (JDA)* ist eine beliebte Java-Bibliothek, die eine einfache Nutzung der Discord API ermöglicht. Sie abstrahiert viele der komplexen Aspekte der API und bietet eine benutzerfreundliche Schnittstelle für die Bot-Entwicklung.

Discord legt großen Wert auf Sicherheit und Datenschutz. Alle Datenübertragungen sind mit TLS (Transport Layer Security) verschlüsselt, um die Vertraulichkeit und Integrität der Daten zu gewährleisten. Benutzer haben die Kontrolle über ihre Privatsphäre-Einstellungen und können festlegen, wer sie kontaktieren kann und welche Informationen öffentlich sichtbar sind. Discord hat auch Richtlinien und Maßnahmen zum Schutz vor Spam, Missbrauch und Belästigung, um eine positive und sichere Umgebung für alle Benutzer zu gewährleisten.<sup>4</sup>

---

1 Discord 2024a.

2 Alle Begriffe, die im Glossar erklärt werden, sind mit diesem Zeichen ▶ gekennzeichnet und führen per Klick direkt zur Erklärung.

3 Discord 2024b.

4 Discord 2024c.

## 1.2 Projektbeschreibung

In dieser Arbeit wird die Entwicklung eines modularen Discord-Bot-Frameworks als Cloud-Service beschrieben. Dieses Projekt zielt darauf ab, ein innovatives und modulares Framework für Discord-Bots zu entwickeln, das auf der *Java Discord API* <sup>5</sup> basiert. Das Framework wird es den Nutzer:innen ermöglichen, benutzerdefinierte Plugins einfach zu erstellen, zu laden und über eine benutzerfreundliche Weboberfläche zu verwalten. Um den Betrieb und die Wartung zu vereinfachen, wird das Framework als Cloud-Service angeboten, sodass die Nutzer keine eigene Hosting-Infrastruktur bereitstellen müssen. Das Projekt beinhaltet die Entwicklung einer *REST-API* für die Kommunikation zwischen der Weboberfläche und dem Backend, sowie die Implementierung von vorgefertigten Plugins, die grundlegende Bot-Funktionalitäten abdecken. Durch diese Lösung wird eine hohe Flexibilität und Erweiterbarkeit gewährleistet, um den unterschiedlichen Anforderungen der Nutzer gerecht zu werden.

## 1.3 Projektziel

Das Hauptziel dieses Projekts ist es, ein leistungsfähiges und flexibles Discord-Bot-Framework zu entwickeln, das sowohl für Anfänger als auch für fortgeschrittene Nutzer:innen zugänglich ist. Das Framework soll es ermöglichen, verschiedene Plugins nahtlos zu integrieren und zu verwalten, ohne dass tiefgehende technische Kenntnisse erforderlich sind. Ein weiteres Ziel ist die Bereitstellung als Cloud-Service, um den Nutzern die Komplexität des eigenen Hostings abzunehmen und gleichzeitig eine hohe Verfügbarkeit und Skalierbarkeit sicherzustellen. Dank der REST-API können Benutzeränderungen und Konfigurationen in Echtzeit verarbeitet werden. Die benutzerfreundliche Oberfläche soll Nutzer in die Lage versetzen, ihre Bots einfach zu konfigurieren und anzupassen, was die allgemeine Benutzererfahrung erheblich verbessert.

## 2 Projektumfeld

### 2.1 Produktportfolio der HiTec GmbH

- Beschreibung des Produktportfolios der HiTec GmbH:
- Welche Produkte und Dienstleistungen bietet die HiTec GmbH an?

### 2.2 Entstehung der Projektidee

- Erklärung, wie es zur Projektidee kam:
- Was waren die Auslöser und Bedürfnisse, die zur Entwicklung dieses Projekts führten?

---

<sup>5</sup> Austin Keener 2024a,b.



### 3 Projektvorbereitung

#### 3.1 Ist-Analyse

- Beschreibung der gegenwärtigen Situation:
- Welche Probleme oder Herausforderungen bestehen derzeit?
- Was ist der aktuelle Stand der Technik bzw. des Systems?

#### 3.2 Soll-Analyse

- Detaillierte Beschreibung der Projektziele:
- Was soll am Ende des Projektes erreicht sein?
- Welche Verbesserungen und Neuerungen sind geplant?

#### 3.3 Projektziel detailliert

- Ausführliche Darstellung der Projektziele:
- Beschreibung der funktionalen und nicht-funktionalen Anforderungen.

#### 3.4 Anforderungen und Technologien

- Auflistung und Beschreibung der Anforderungen:
- Technologische Anforderungen (z.B. REST-API, Datenbanken)
- Zielplattformen und benötigte Technologien (z.B. Cloud-Services)

Die Java Discord API (JDA) ist eine umfassende Programmbibliothek, die Entwicklern ermöglicht, Discord-Bots in der Programmiersprache Java zu erstellen und zu verwalten. JDA bietet eine hohe Abstraktionsebene, um die Kommunikation zwischen einem Java-Programm und der Discord-API zu erleichtern. Mit JDA können Entwickler auf verschiedene Funktionen und Ereignisse in Discord zugreifen, wie das Empfangen und Senden von Nachrichten, das Verwalten von Servern, Kanälen und Benutzern sowie das Reagieren auf verschiedene Interaktionen innerhalb von Discord.

Zu den Hauptfunktionen der JDA gehören:

- **Nachrichtenverwaltung:** Senden, Bearbeiten und Löschen von Nachrichten in Textkanälen.
- **Benutzerverwaltung:** Abrufen von Benutzerinformationen, Verwalten von Rollen und Berechtigungen.
- **Ereignisbehandlung:** Reagieren auf Ereignisse wie Nachrichtenempfang, Benutzerbeitritt und -austritt, Reaktionen und vieles mehr.
- **Sprachunterstützung:** Unterstützung für die Audioübertragung in Sprachkanälen.

JDA ist besonders für seine einfache Handhabung und umfangreiche Dokumentation bekannt, die es sowohl Anfängern als auch erfahrenen Entwicklern ermöglicht, leistungsstarke und funktionsreiche Discord-Bots zu erstellen. Die Bibliothek wird aktiv gepflegt und weiterentwickelt, um stets mit den neuesten Änderungen und Funktionen der Discord-API kompatibel zu bleiben.

## **4 Projektdurchführung**

### **4.1 Vorgehensmodell**

- Wahl des Vorgehensmodells (z.B. Scrum, Kanban):
- Begründung für die Wahl des Vorgehensmodells
- Auswirkungen des Vorgehensmodells auf die Projektdurchführung

### **4.2 Umsetzung**

#### **4.2.1 Klassenmodell**

- Darstellung des Klassenmodells:
- Diagramme und Beschreibungen der Klassen und deren Beziehungen

#### **4.2.2 Datenhaltung**

- Wahl der Datenhaltung:
- Beschreibung der Datenbankstruktur und des Datenbankmodells

#### **4.2.3 Design Patterns**

- Einsatz von Design Patterns:
- Welche Design Patterns werden verwendet und warum?

### **4.3 Qualitätssicherung**

#### **4.3.1 Teststrategien**

- Beschreibung der Teststrategien:
- Welche Tests werden durchgeführt (Unit Tests, Integrationstests, Systemtests)?

#### **4.3.2 Testfälle**

- Konkrete Testfälle:
  - Beschreibung der durchgeführten Testfälle und deren Ergebnisse

### **5 Wirtschaftliche Betrachtung**

#### **5.1 Marktuntersuchung**

##### **5.1.1 Zielgruppe**

- Beschreibung der Zielgruppe:
  - Welche Erwartungen hat die Zielgruppe an das Produkt?

##### **5.1.2 Marktvolumen und Marktpotential**

- Analyse des Marktvolumens und -potentials:
  - Wie groß ist der Markt für das Produkt?
  - Welche Wachstumsmöglichkeiten gibt es?

##### **5.1.3 Konkurrenzanalyse**

- Analyse der Wettbewerbssituation:
  - Welche Konkurrenzprodukte gibt es und wie unterscheiden sie sich?

#### **5.2 Marketingmix (4P)**

##### **5.2.1 Produktpolitik**

- Beschreibung der Produktpolitik:
  - Welche Produktvarianten werden angeboten?

##### **5.2.2 Preispolitik**

- Festlegung der Preispolitik:
  - Welche Preisstrategie wird verfolgt?

### **5.2.3 Kommunikationspolitik**

- Beschreibung der Kommunikationsstrategie:
  - Wie wird das Produkt beworben?

### **5.2.4 Distributionspolitik**

- Beschreibung der Distributionspolitik:
  - Wie gelangt das Produkt zum Endkunden?

## **5.3 Kostenplanung**

### **5.3.1 Personalkosten**

- Aufschlüsselung der Personalkosten:
  - Kosten für die beteiligten Mitarbeiter und deren Einsatzzeiten

### **5.3.2 Sachmittelkosten**

- Aufschlüsselung der Sachmittelkosten:
  - Kosten für benötigte Hardware und Software

## **5.4 Wirtschaftlichkeitsberechnung**

### **5.4.1 Gewinnschwellenberechnung**

- Berechnung der Gewinnschwelle:
  - Ab welcher Menge erzielt das Projekt Gewinn?

### **5.4.2 Amortisationsrechnung**

- Berechnung der Amortisationszeit:
  - Wie lange dauert es, bis sich das Projekt amortisiert hat?

## **6 Projektabschluss**

### **6.1 Erreichung des Projektziels**

- Zusammenfassung, ob das Projektziel erreicht wurde:
  - Welche Ziele wurden erreicht und welche nicht?

### **6.2 Änderungen zur anfänglichen Planung**

- Darstellung der Änderungen zur ursprünglichen Planung:
  - Welche Änderungen wurden während des Projekts vorgenommen und warum?

### **6.3 Fazit**

- Persönliches Fazit:
  - Wie hat die Teamarbeit funktioniert?
  - Was hat das Projekt persönlich gebracht?

## A Anhang

### A.1 Glossar

#### C

**Client-Server-Architektur** Die Client-Server-Architektur ist ein Netzwerkmodell, bei dem Aufgaben und Dienste zwischen zwei Hauptkomponenten aufgeteilt werden: dem Client und dem Server. Der Client stellt Anfragen, um auf Ressourcen oder Dienste zuzugreifen, die vom Server bereitgestellt werden. Der Server verarbeitet diese Anfragen und liefert die entsprechenden Antworten oder Ressourcen zurück. Dieses Modell ist weit verbreitet in der Entwicklung von Webanwendungen, bei denen der Client in der Regel ein Webbrowser ist, der Anfragen an einen Webserver stellt. 1

#### D

**Discord API** Die Discord API ist eine Programmierschnittstelle, die Entwicklern Zugriff auf die Funktionen und Daten von Discord bietet. Mit der API können Entwickler Bots und Anwendungen erstellen, die mit Discord-Servern und -Benutzern interagieren. Die API ermöglicht das Senden und Empfangen von Nachrichten, das Verwalten von Servern und Kanälen, das Abrufen von Benutzerinformationen und vieles mehr. Die Discord API nutzt REST für die meisten ihrer Funktionen und bietet Echtzeit-Updates über WebSockets. 1

#### E

**Endpunkt** Ein Endpunkt ist eine spezifische URL innerhalb einer API, die eine bestimmte Funktion oder Ressource repräsentiert. Endpunkte werden verwendet, um verschiedene Operationen wie das Abrufen von Daten, das Senden von Informationen oder das Ausführen von Aktionen zu ermöglichen. Jeder Endpunkt wird durch eine HTTP-Methode (GET, POST, PUT, DELETE usw.) und eine URL definiert. Zum Beispiel könnte ein Endpunkt für das Abrufen von Benutzerinformationen in einer API die URL `/users/{id}` haben und mit der HTTP-Methode GET aufgerufen werden. 1

#### H

**HTTPS-Protokoll** Das HTTPS-Protokoll (Hypertext Transfer Protocol Secure) ist eine Erweiterung des HTTP-Protokolls, die eine sichere Kommunikation über ein Computernetzwerk ermöglicht. HTTPS verwendet das Transport Layer Security (TLS) Protokoll, um die Daten zwischen dem Webbrowser und dem Webserver zu verschlüsseln. Dies stellt sicher, dass die übertragenen Daten vertraulich und vor Manipulationen geschützt sind. HTTPS ist besonders wichtig für Webseiten, die sensible Daten wie Passwörter, Kreditkarteninformationen und persönliche Informationen verarbeiten. 1

#### J

**Java Discord API** Die Java Discord API (JDA) ist eine umfassende Programmbibliothek, die Entwicklern ermöglicht, Discord-Bots in der Programmiersprache Java zu erstellen und zu verwalten. JDA bietet eine hohe Abstraktionsebene, um die Kommunikation zwischen einem Java-Programm und der Discord-API zu erleichtern. Mit JDA können Entwickler auf verschiedene Funktionen und Ereignisse in Discord zugreifen, wie das Empfangen und Senden von Nachrichten, das Verwalten von Servern, Kanälen und Benutzern sowie das Reagieren auf verschiedene Interaktionen innerhalb von Discord. 2

## R

**REST-API** Eine REST-API (Representational State Transfer Application Programming Interface) ist ein Architekturstil für APIs, der auf den Prinzipien von REST basiert. REST-APIs verwenden HTTP-Anfragen, um auf Ressourcen zuzugreifen und diese zu manipulieren. Diese Ressourcen werden durch URLs identifiziert, und die gängigen HTTP-Methoden (GET, POST, PUT, DELETE) werden verwendet, um verschiedene Operationen auszuführen. REST-APIs sind stateless, was bedeutet, dass jede Anfrage alle Informationen enthält, die der Server benötigt, um sie zu verarbeiten, ohne den Zustand zwischen den Anfragen zu speichern. REST-APIs sind bekannt für ihre Einfachheit, Flexibilität und Skalierbarkeit. 2

## W

**WebSocket-Verbindungen** WebSocket-Verbindungen sind ein Kommunikationsprotokoll, das eine bidirektionale, Echtzeit-Kommunikation zwischen einem Client (z.B. einem Webbrowser) und einem Server ermöglicht. Im Gegensatz zu traditionellen HTTP-Verbindungen, die unidirektional sind und für jede Datenübertragung eine neue Verbindung aufbauen, bleibt eine WebSocket-Verbindung während der gesamten Kommunikationssitzung offen. Dies ermöglicht eine effizientere und schnellere Datenübertragung, die besonders für Anwendungen wie Chat-Dienste, Online-Spiele und Echtzeit-Datenfeeds nützlich ist. WebSockets sind in der WebSocket-Spezifikation des IETF (Internet Engineering Task Force) definiert und nutzen das ws:// oder wss:// Schema für unverschlüsselte bzw. verschlüsselte Verbindungen. 1

## A.2 Literaturverzeichnis

Austin Keener, Florian Spieß (2024a). *JDA (Java Discord API) Documentation*. Version v5.0.0-beta.24.

URL: <https://jda.wiki/introduction/jda/> (besucht am 25. 05. 2024).

- (2024b). *JDA: Java Discord API*. Version v5.0.0-beta.24. URL: <https://github.com/discord-jda/JDA> (besucht am 20. 05. 2024).

Discord (2024a). *Beginner's Guide to Discord*. URL: <https://support.discord.com/hc/en-us/articles/360045138571-Beginner-s-Guide-to-Discord> (besucht am 22. 05. 2024).

- (2024b). *Discord Developer Portal*. URL: <https://discord.com/developers/docs/intro> (besucht am 25. 05. 2024).

- (2024c). *Discord Privacy Policy*. URL: <https://discord.com/privacy#2> (besucht am 25. 05. 2024).



### A.3 Quellcode

```
1  public class ReadyListener implements EventListener
2  {
3      public static void main(String[] args)
4          throws InterruptedException
5      {
6          // Note: It is important to register your ReadyListener before building
7          JDA jda = JDABuilder.createDefault("token")
8              .addEventListeners(new ReadyListener())
9              .build();
10
11          // optionally block until JDA is ready
12          jda.awaitReady();
13      }
14
15      @Override
16      public void onEvent(GenericEvent event)
17      {
18          if (event instanceof ReadyEvent)
19              System.out.println("API is ready!");
20      }
21 }
```

Quelltext 1: Beispielcode für einen ReadyListener in JDA