Programmierung eines Discord Chatbots

**Maturarbeit**

**Datum**: 14.10.2021

**Klasse**: G2018E

**Schule**: Neue Kantonsschule Aarau

Dieses Dokument © 2021 ist lizenziert unter Attribution-NonCommercial-ShareAlike 4.0 International. Eine Kopie dieser Lizenz finden Sie unter <http://creativecommons.org/licenses/by-nc-sa/4.0/>

### Abstract

Die Maturarbeit legt die Vorgehensweise bei der Programmierung eines eigenen Chatbots dar. Anfänglich werden die theoretischen Grundlagen zu dem Begriff «Bot» an sich erklärt. Zudem wird aufgezeigt, wie man Bots heutzutage einsetzt und wie die Zukunftsaussichten sind. Danach werden die allgemeinen Arbeitsschritte zur Erstellung eines eigenen Discord Chatbots dargelegt. Im darauffolgenden Teil der Arbeit werden die allgemeinen Arbeitsschritte präzisiert und implementiert. Das Endergebnis der Arbeit ist ein eigener funktionsfähiger Chatbot für die Applikation Discord. Der Bot reagiert auf grundlegende Befehle und kann wahlweise Datum, Uhrzeit oder das Wetter ausgeben.

Diese Arbeit, in Kombination mit der Entwicklung des Bots, soll Einblicke in den Bereich Programmierung und die Welt der Bots gewähren.

### Vorwort

Mir war von Anfang an klar, dass sich meine Maturarbeit um das Thema Programmieren drehen sollte. Im Verlauf der I-Week im Januar 2021 wurde diese Idee dann konkret. Dort beschäftigte ich mich erstmals mit dem Thema «Bots». Schnell kam mir die Idee, einen Bot für meinen eigenen Discord Server zu entwickeln.

Discord ist eine Plattform, um zu chatten und Video- und Sprachanrufe zu tätigen. Ursprünglich wurde sie für die Gaming Community entwickelt, wird inzwischen aber auch für andere Zwecke verwendet. Discord ist sowohl als Mobile-, wie auch Webanwendung, oder auch als Desktop Applikation verfügbar. (Wikipedia – Discord, 2021)

Das Thema «Bots» ist sehr interessant für mich, denn es gibt unendlich viele Möglichkeiten, was die Funktionalität betrifft. Man kann seiner Kreativität freien Lauf lassen und immer neue Features hinzufügen. Somit konnte ich das Thema Programmieren mit dem Interesse an Bots verknüpfen und in dieser Arbeit zusammenführen.

### Inhaltsverzeichnis

[1 Einleitung 6](#_Toc83147851)

[2 Theoretische Grundlagen 7](#_Toc83147852)

[2.1 Begriffserklärung Chatbot 7](#_Toc83147853)

[2.2 Geschichte der Chatbots 7](#_Toc83147854)

[2.3 Verschiedene Arten von Bots 9](#_Toc83147855)

[2.4 Einsatzbereiche 10](#_Toc83147856)

[2.5 Vor- und Nachteile 11](#_Toc83147857)

[2.6 Zukunft 12](#_Toc83147858)

[3 Grundlagen für die Entwicklung eines eigenen Bots 13](#_Toc83147859)

[3.1 Anwendungsplattform 13](#_Toc83147860)

[3.2 Funktionalität 13](#_Toc83147861)

[3.3 Programmiersprachen / Code-Editoren 13](#_Toc83147862)

[3.4 Hosting Plattform 14](#_Toc83147863)

[4 Entwicklung eines eigenen Discord Chatbots 14](#_Toc83147864)

[4.1 Funktionalitätsentscheidungen 14](#_Toc83147865)

[4.2 Discord Developer Portal 14](#_Toc83147866)

[4.3 Einladen auf einen Server 17](#_Toc83147867)

[4.4 Entwicklungsprozess 20](#_Toc83147868)

[4.5 Hosting 24](#_Toc83147869)

[5 Diskussion & Reflexion 26](#_Toc83147870)

[6 Literaturverzeichnis 27](#_Toc83147871)

[7 Abbildungsverzeichnis 29](#_Toc83147872)

# Einleitung

Chatbots, oder auch allgemein Bots, gibt es bereits seit längerer Zeit, erst in den letzten Jahren jedoch haben sie bei Unternehmen sowie auch Privatbenutzern an Popularität gewonnen. Bots werden heutzutage vielseitig und in allen Lebensbereichen eingesetzt. Für die zunehmende Aufmerksamkeit in diesem Bereich haben vor allem Fortschritte in den Bereichen Künstliche Intelligenz (KI) und Machine Learning (ML) gesorgt. (Onlim – Die Geschichte und Entwicklung von Chatbots, 2017)

Ein weiterer Faktor ist auch das seit Jahren zunehmende Interesse an Messaging-Services und Apps, wo Bots heute eine wichtige Rolle spielen. Bots sind allerdings nicht nur auf Sozialen Medien präsent, sondern auch im Internet, ohne dass Nutzer davon etwas mitbekommen, dies am häufigsten in Form von Webcrawlern, auf die ich später noch zurückkommen werde. (Degruyter- «Good Bot, Bad Bot», 2018)

Ich habe mich bei dieser Arbeit hauptsächlich im Gebiet der Chatbots bewegt, andere Arten von Bots werden jedoch kurz angesprochen. Im Folgenden wird mit dem Begriff «Bot» jedoch meist «Chatbot» gemeint.

# Theoretische Grundlagen

## Begriffserklärung Chatbot

«Unter einem Bot versteht man ein Computerprogramm, das weitgehend automatisch sich wiederholende Aufgaben abarbeitet, ohne dabei auf eine Interaktion mit einem menschlichen Benutzer angewiesen zu sein.» (Wikipedia – Bot, 2021)

Der allgemeine Begriff «Bot» leitet sich von dem englischen Wort «robot» ab. Mehr als die Hälfte des heutigen Internetverkehrs findet durch Bots statt. Die genaue Anzahl an Bots ist jedoch unbekannt. (Degruyter- «Good Bot, Bad Bot», 2018)

Ein Chatbot-System, oder generell ein Bot, ist ein Computer-Programm. Man kann es als Kommunikationsschnittstelle ansehen, durch welche die Nutzer mit Systemen und Unternehmen, in beide Richtungen einfach, schnell und sicher interagieren können. (Cloudflare – what is a bot, 2021)

Bots unterscheidet man in zwei Kategorien: Sogenannte «gutartige Bots» und «bösartige Bots». «Gute Bots» sind Bots, die dem Benutzer das Leben vereinfachen, indem sie z.B. Inhalte für Suchen indizieren oder Kundendienste anbieten. «Bösartige» Bots hingegen sind darauf ausgerichtet, z.B. Benutzerkonten zu hacken, Kontaktinformationen zu sammeln, um sie danach für Spam zu benutzen. (Cloudflare – what is a bot, 2021)

Es gibt einfach gestaltete Chatbots, welche nur auf vorgegebene Befehle hin reagieren, aber auch interaktive, komplexere Bots. Solche Bots greifen häufig auf Künstliche Intelligenz (KI) zurück, dies meist mit dem Ziel, einen menschlichen Chatteilnehmer zu imitieren und ein angenehmes Gesprächsklima für Kunden und Nutzer zu bieten. (Cloudflare – How is an internet bot constructed, 2021)

## Geschichte der Chatbots

**ELIZA**

Der erste Chatbot, der auf den Namen «ELIZA» hörte, wurde 1966 von MIT Professor Joseph Weizenbaum entwickelt. Sie stellte die Simulation eines Psychotherapeuten dar.

Weizenbaums Modell «ELIZA» haben seitdem viele Entwickler übernommen und angepasst, um die menschenähnlichen Interaktionen weiterzuentwickeln. (Onlim – Die Geschichte und Entwicklung von Chatbots, 2017)

Weitere bekannte Beispiele für frühe Chatbots sind unter anderem:

|  |  |
| --- | --- |
| **Name, Jahr** | **Funktionalität** |
| PARRY, 1971 | Imitierte Patienten mit Schizophrenie, Ziel war es die Krankheit besser zu verstehen. |
| Jabberwacky, 1988 | War der erste Chatbot mit dem ausschliesslichen Zweck, Unterhaltungen zu führen. Er verwendete Mustererkennung, um passende Antworten zu berechnen. |
| Dr. Sbaitso, 1992 | Erste Versuche Künstliche Intelligenz (KI) in einen Chatbot zu integrieren. Er verfügte über ein sprachgesteuertes Chatprogramm. |

**A.L.I.C.E.**

«A.L.I.C.E.» ist die Abkürzung für «Artificial Linguistic Internet Computer Entity». Ursprünglich entwickelt wurde es von Richard Wallace. Als Open-Source-Projekt arbeiteten aber auch zahlreiche andere Programmierer an der Software mit. Das Programm imitierte die Persönlichkeit einer jung aussehenden Frau und konnte Nutzer über ihre Persönlichkeit informieren. (Onlim – Die Geschichte und Entwicklung von Chatbots, 2017)

**Siri & Google Assistant**

Moderne Chatbots sind wesentlich umfangreicher und verfügen über verschiedenste Funktionen. Im Folgenden werden 2 der bekanntesten Chatbots besprochen.

Siri ist der persönliche Assistent von Apple. Er wurde ausschliesslich für das Apple-eigene Betriebssystem IOS entwickelt und existiert seit 2010.

Siri war anfangs ausschliesslich ein sprachgesteuertes System, wurde aber auch schnell um ein «klassisches» Texteingabesystem erweitert. (Onlim – Die Geschichte und Entwicklung von Chatbots, 2017)

Google Assistant ist ein weiteres Beispiel für einen Chatbot. Er ist ein persönlicher Assistent, der auf Fragen antwortet, Tätigkeiten auf Wunsch ausführt und Empfehlungen abgibt.

Eine der Hauptfunktionen des Google Assistant ist, dass er Fragen in Bezug auf den aktuellen Bildschirm-inhalt in Echtzeit beantworten kann.

Ansonsten deckt sich die Funktionalität mit der Konkurrenz, wie Siri, Amazon Alexa oder Microsoft Cortana. (Onlim – Die Geschichte und Entwicklung von Chatbots, 2017)

Beide, Siri und Google Assistant, gehören zu den fortschrittlichsten Chatbots in der Branche.

## Verschiedene Arten von Bots

Es gibt diverse Arten von Bots. Am häufigsten kommen jedoch die nachfolgend beschriebenen Arten vor.

**Webcrawler**

Webcrawler sind Bots, die von Suchmaschinen-Betreibern entwickelt wurden (Google, Bing, Yahoo etc.). Sie besuchen selbstständig Webseiten und werten deren Inhalt aus. Webcrawler halten sich an das «Robots-Exclusion-Standard-Protokoll». Bei diesem Protokoll wird festgelegt, dass ein Webcrawler beim Besuchen einer Webseite zuerst eine Datei mit dem Namen «robots.txt» lesen muss.

In dieser Datei kann der Betreiber einer Webseite festlegen, welche Bereiche einer Website der Webcrawler besuchen darf. Dadurch lassen sich ausgesuchte Bereiche der Webseite für Suchmaschinen sperren. (Wikipedia – Robots Exclusion Standard, 2021)

**Chatbots**

Ein Chatterbot, Chatbot oder auch einfach nur Bot genannt, ist eine der häufigsten Arten von Bots. Ein Chatbot ist ein textbasiertes Dialogsystem, welches es dem Nutzer ermöglicht, mit einem technischen System zu kommunizieren. Die bekanntesten Beispiele dafür sind digitale Assistenten wie z.B. Google Assistant, Amazon Alexa oder Apple Siri. Ein Chatbot hat üblicherweise zwei grundlegende Bereiche, und zwar Textein- und -ausgabe. Komplexere Chatbot-Systeme können sogar auf Künstliche Intelligenz (KI) zurückgreifen, um einen noch authentischeren Dialog zu bieten. (Wikipedia – Chatbot, 2021)

Solche Systeme werden heutzutage auch als virtuelle persönliche Assistenten bezeichnet und werden häufig von Unternehmen verwendet, welche darüber Dienstleistungen anbieten. (Wikipedia – Chatbot, 2021)

**Social Bots**

Bots, die in Sozialen Medien eingesetzt werden, nennt man Social Bots. Grundsätzlich imitieren sie einen Nutzeraccount und kommentieren, liken oder verbreiten Werbung.

Viele dieser Bots greifen auf einfache Algorithmen zurück und verwenden Keyword-Suchen. Sie analysieren Posts auf der Suche nach bestimmten Hashtags oder Stichwörtern. Sie können aber auch Nutzerprofile, Verläufe und Suchhistorien analysieren, um so dem Nutzer massgeschneiderte Nachrichten oder Werbung für ein Produkt schicken. (Wikipedia – Bot, 2021)

Eines der grössten Probleme mit Social Bots ist die Manipulationsfähigkeit. Mit einem Netzwerk von Bots kann man politische Debatten oder auch direkt Nutzer relativ stark beeinflussen und Einfluss auf die Meinung Einzelner und/oder Gruppen nehmen. Ein Beispiel dafür ist der US-Wahlkampf von 2016. Laut einer Analyse des Tools «Twitter Audit» stellten sich zeitweise je 30% der Follower beider Kandidaten als Bots heraus.

**Spambots**

Ein Spambot ist ein Beispiel für einen sogenannten «bösartigen» Bot. Sie können von Betreibern dazu genutzt werden, um z.B. E-Mail-Adressen zu sammeln, um dann unaufgefordert Werbemails an die Empfänger zu schicken. (Wikipedia – Bot, 2021)

**Spider-Bots**

Ein weiteres Beispiel für «bösartige» Bots sind die Spider-Bots. So werden Bots genannt, welche massenhaft unautorisiert Kopien von Webinhalten erstellen oder systematisch Sofwarelücken von Servern ausspionieren mit dem Ziel, den Server zum Absturz zu zwingen.

Besonders grosse Schäden können viele zusammengeschlossene «bösartige» Bots anrichten. Ein Netzwerk von Bots, die untereinander kommunizieren, nennt man ein Bot-Netz oder Botnet. (Wikipedia – Bot, 2021)

Dies ist ein hohes Sicherheitsrisiko für Webseitenbetreiber, gegen das man Massnahmen ergreifen muss.

Eine der meist verbreitetsten Möglichkeiten gegen das «Spidern» besteht darin, das Verhalten des Bots zu analysieren und mit z.B. einer Sperrung der dazugehörigen IP-Adresse darauf zu reagieren. Das Verhalten des Bots lässt sich innerhalb einer Falle, auch «Honeypot» genannt, analysieren. «Honeypots» sind separat aufgesetzte Server, welche die Netzwerkdienste eines Computers oder gar eines ganzen Rechnernetzes simulieren. Erfolgt dann ein Zugriff auf den virtuellen Dienst oder Anbieter, werden sämtliche Aktionen protokolliert. Durch deren Analyse kann man dann auf dem Hauptnetzwerk entsprechende Gegenmassnahmen ergreifen. (Wikipedia – Honeypot, 2021)

## Einsatzbereiche

Chatbots werden heutzutage in den verschiedensten Arbeitsbereichen und Industrien eingesetzt. Durch Fortschritte in der Technologie, wie Künstliche Intelligenz (KI) oder Machine Learning (ML) haben viele Unternehmen den Nutzen von Bots für sich entdeckt.

Chatbots können Standardinteraktionen mit den Benutzern und Kunden automatisieren, somit die Geschäftsprozesse unterstützen und beschleunigen, da Computer die Arbeitsabläufe effizienter und fehlerfreier ausführen können als z.B. ein Supportangestellter. Zudem kann ein Unternehmen durch die Automatisierung von Vorgängen Kosten einsparen und auf andere Bereiche mehr Fokus legen. (Onlim – Die Geschichte und Entwicklung von Chatbots, 2017)

So findet man Bots, zum Beispiel, in Bereichen wie Internet-Banking, Versicherungen, Reiseindustrie oder der Gastronomie.

Im Internet-Banking werden mithilfe von Chatbots Vorgänge wie Überprüfen des Guthabens, Transaktionen, Anfordern einer neuen Karte, Sperren einer verlorenen Karte oder ähnliche Standard-Supportfunktionen automatisiert. Eine weitere Branche, in der Chatbots auf gleiche Art & Weise eingesetzt werden, um den Kontakt mit Kunden zu automatisieren, ist der Versicherungssektor. (E-Business – Wo werden Chatbots derzeit eingesetzt, 2018)

In der Reisebranche werden Umsätze über die gezielte Preispolitik generiert. Kunden wollen immer das beste Preis-Leistungs-Verhältnis. Chatbots können hier den Prozess erheblich vereinfachen und beschleunigen. Sie kommunizieren mit Datenbanken, um Preise für Flüge oder Hotels herauszufinden und vergleichen diese. Dem Nutzer werden so verschiedene Optionen dargelegt und der Bot kann bei allfälligen Preisveränderungen den Kunden über diese informieren. (E-Business – Wo werden Chatbots derzeit eingesetzt, 2018)

## Vor- und Nachteile

Chatbots haben sehr viele Vorteile, sowohl aus der Sicht von Unternehmen als auch der Kunden.

Ein Vorteil von Chatbots ist beispielsweise, dass sie Prozesse rationalisieren können. Sie können Kunden bei der Suche nach geeigneten Produkten oder Dienstleistungen helfen und die gewünschten Informationen bereitstellen.

Chatbots vermögen aber auch komplexere Vorgänge in klar definierte Schritte aufzuteilen. Damit ermöglichen sie eine «persönliche» Beratung des Kunden. (ECN – Chatbots: Vor- und Nachteile dieser Tools, 2018)

Der grösste Vorteil bei der Rationalisierung ist jedoch die Schnelligkeit, in der Bots Tätigkeiten ausführen können. Auch hervorzuheben ist, dass Chatbots, im Vergleich zum menschlichen Personal, rund um die Uhr zur Verfügung stehen. Zudem beherrscht ein professioneller Chatbot mehr als eine Sprache und können Anfragen gleichzeitig bearbeiten, somit mehrere Kunden parallel bedienen. (ECN – Chatbots: Vor- und Nachteile dieser Tools, 2018)

Chatbots sind jedoch nicht nur von Vorteil. Ein Nachteil z.B. ist, dass ein Chatbot nicht in jeder Situation optimal einsetzbar ist. Probleme, die zu komplex oder zu fallspezifisch sind, erfordern immer noch menschliches Personal.

Technologien wie Künstliche Intelligenz (KI) sind noch lange nicht Industriestandard bei Chatbots. Das heisst, dass viele Systeme auf vorprogrammierte Befehle zurückgreifen oder nur über ein begrenztes Verständnis von Kundenanfragen verfügen. Ferner finden meist ältere Menschen keinen Zugang zu einem virtuellen Assistenten. (ScienceSoft – Chatbot-Entwicklung: Was zu beachten ist, um einen erfolgreichen Chatbot zu erstellen, 2020)

Ferner folgen Chatbots bei Anfragen meist einem Muster. Wenn dieses Muster durch Kleinigkeiten, seien es nur Tippfehler, unspezifische Fragen oder Sonstiges, unterbrochen wird, kann das dazu führen, dass der Chatbot das Anliegen nicht bearbeiten kann.

## Zukunft

Die Zukunft der Chatbots ist sehr vielversprechend. Durch externe Faktoren, wie die weiterhin andauernde Covid-19 Pandemie, wird die Digitalisierung verstärkt vorangetrieben. Im Zusammenhang mit Home-Office haben Unternehmen Interesse an Chatbots gewonnen und das Interesse an Bot weist einen steigenden Trend auf (Abbildung 5).

Dadurch, dass es heutzutage für den Nutzer einfach ist mit dem Bot zu interagieren, kann man davon ausgehen, dass Bots immer häufiger eingesetzt werden, um den Massengeschäft zu bedienen.

Chatbots wie Siri oder Google Assistant sind bereits sehr fortgeschritten, jedoch noch lange nicht perfekt. Der heutige Zustand der Chatbots lässt sich laut Experten mit dem Zustand des «World Wide Web» im Jahr 1995 vergleichen. Man kann sie als «Teilabschnitt auf dem Weg in die Zukunft» ansehen. (E-Business – Wo werden Chatbots derzeit eingesetzt, 2018)

Laut einer Untersuchung von Grand View Research soll bis 2025 der Markt für Chatbots global um 25% wachsen. Adobe gab bekannt, dass bereits jetzt knapp 1/3 aller führenden Unternehmen Chatbots und Künstliche Intelligenz (KI) für Marketingzwecke nutzen und diesen Bereich zukünftig stark ausbauen wollen. (freshservice Blog – Die besten Chatbots für Ihr Unternehmen: Aktueller Stand und Zukunftsblick, 2021)

Die Zukunft der Chatbots bleibt also spannend, insbesondere mit Blick auf Technologien wie KI oder ML, welche in ein paar Jahren höchst wahrscheinlich zum Industriestandard werden.

Ebenfalls kann man erwarten, dass sich mit höherer Rechenleistung der Computer auch die Effizienz und Schnelligkeit der Chatbots verbessern wird. (freshservice Blog – Die besten Chatbots für Ihr Unternehmen: Aktueller Stand und Zukunftsblick, 2021)

# Grundlagen für die Entwicklung eines eigenen Bots

Grundlegend für die Programmierung ist die Überlegung, für welche Applikation man einen Bot schreiben will und welche Funktionalität der Bot haben soll.

## Anwendungsplattform

Es gibt mittlerweile viele Applikationen, die es einem ermöglichen, einen eigenen Bot zu erstellen. Beispiele dafür wären Discord, Instagram, WhatsApp, Telegram, Skype etc. Jede dieser Applikation hat jedoch spezielle Anforderungen, weshalb man sich in einem ersten Schritt für eine Applikation entscheiden muss.

## Funktionalität

Hat man die Applikation gefunden, für die man einen Bot erstellen möchte, muss man sich Gedanken über die Funktionalität machen. Grenzen dabei werden einem nur durch technische Limitationen gesetzt. Wissenslücken kann man leicht schliessen, denn es gibt heutzutage eine Vielzahl von Online-Kursen, YouTube Tutorials oder Artikeln zu dem Thema.

## Programmiersprachen / Code-Editoren

Sobald man eine konkrete Idee im Sinn hat, steht der Entwicklung nichts mehr im Wege. Spätestens jetzt aber stellt sich noch die Frage, welche Programmiersprache man benutzen möchte. Da es nicht «die eine» Sprache zur Bot-Programmierung gibt, sollte man einfach diejenige verwenden, die einem am besten liegt und welche von der Ziel-Applikation unterstützt wird.

Hat man seine Programmiersprache gefunden, stellt sich die Frage nach der Entwicklungsumgebung, die verwendet werden soll. Grundsätzlich reicht es, einen einfachen Texteditor wie z.B. Notepad++ zur Erstellung der Quellcode-Dateien zu verwenden.

Komfortabler präsentieren sich hingegen integrierte Entwicklungsumgebungen (IDEs) wie Visual Studio (für .Net Sprachen) oder Eclipse (für Java) und andere, wie z.B. PyCharm für Python.

Diese IDEs stellen dem Entwickler diverse Werkzeuge zur Verfügung, welche die Produktivität erhöhen sollen, kosten aber vielfach auch Geld für die Lizenzgebühren.

Einen guten Kompromiss stellt Visual Studio Code dar. Dieses Werkzeug ist Open-Source, ist für diverse Betriebssysteme und Programmiersprachen verwendbar und erleichtert durch verschiedenste Erweiterungen die Erstellung von Quellcode.

## Hosting Plattform

Spätestens wenn der Bot einsatzbereit ist, muss man die Entscheidung treffen, wie man ihn zur Verfügung stellt. In der Regel möchte man, dass er rund um die Uhr verfügbar ist, somit stellt sich die Frage, wo man ihn hosten kann. Es gibt diverse Anbieter, die solch ein Hosting anbieten. Bekannte Anbieter, wie z.B. Amazon Cloud Services oder Google Cloud sind eher auf professionelle Anliegen ausgelegt. Sie müssen in der Regel nach einer kostenlosen Testphase bezahlt werden und können schnell kostspielig werden.

Es gibt jedoch auch einige Anbieter, die das Hosting bis zu einem gewissen Masse gratis ermöglichen. Beispiele dafür wären Firebase von Google, Back4App oder Heroku. (Back4App – Top 5 Alternativen zu Heroku, 2021)

# Entwicklung eines eigenen Discord Chatbots

Die Idee für diese Arbeit entstand während der I-Week im Januar 2021. Die Entscheidung, einen Bot für die Applikation Discord zu programmieren lag nahe, da ich dort einen eigenen Server betreibe.

Um die Funktionalität meines Bots aufzuzeigen, habe ich ein kurzes Video erstellt: <https://www.youtube.com/watch?v=QGXj_cqukZM>

Der Code des Bots ist in einem Repository auf GitHub abgelegt: <https://github.com/floriankalis/Programmierung-eines-Discord-Chatbots>

## Funktionalitätsentscheidungen

Der nächste Schritt war dann, sich Gedanken über die Funktionalität meines Bots zu machen. Meine Absicht war es zunächst, dass der Bot auf simple Befehle hin antworten kann. Ich wollte allerdings keinen Chatbot programmieren, den man einfach anschreiben kann, sondern ihn so gestalten, dass er nur auf vordefinierte Nachrichten reagiert.

Dabei habe ich mich an Konzepten anderer Discord Bots orientiert und mich dafür entschieden, dass alle Nachrichten, auf die der Bot reagieren soll, mit einem «!» Zeichen anfangen müssen.

Die Hauptfunktion des Bots sollte jedoch eine andere sein. Der Nutzer sollte Spieler-Statistiken eines bestimmten Videospiels mithilfe des Bots abfragen können. Schickt der Benutzer «!stats» mit anschliessendem Spielernamen an den Bot, soll dieser von mir vordefinierte Statistiken des Nutzerprofils von einer Website holen und sie in Discord ausgeben. Auf diese Weise braucht der Spieler nicht mehr separat die externe Statistik-Website zu besuchen.

## Discord Developer Portal

Um einen Bot für Discord zu erstellen, bedarf es einiger vorbereitender Schritte. Discord selbst stellt für die Entwicklung eigener Bots bereits ein Tool zur Verfügung: das Discord Developer Portal. Um einen eigenen Bot entwickeln zu können, muss man sich dort registrieren, oder mit seinem existierenden Discord Account einloggen und eine API-Applikation erstellen. Hierfür klickt man auf den entsprechenden Button neben seinem Profil, wie in der unteren Abbildung zu sehen ist.

Die Applikation wird anschliessend unter «My Applications» aufgelistet. Klickt man die Applikation an, gelangt man auf die Übersichtseite der Anwendung. Auf der API-Applikation kann man nun einen Bot erstellen, dargestellt in der folgenden Abbildung.

Nun kann man die Einstellungen für den Bot anpassen, zum Bespiel Name, Profilbild etc. Gleichzeitig mit dem Bot hat Discord einen Token generiert. Dieser Token ist essenziell dafür, dass der Bot funktioniert. Er identifiziert den Bot eindeutig in Discord und sollte unter keinen Umständen an Dritte weitergegeben werden. Wer über den Token verfügt, kontrolliert die Anwendung. Discord zeigt ihn deshalb auf der Verwaltungsseite des Bots standardmässig nicht an, bietet aber die Möglichkeit, ihn bei Bedarf anzeigen zu lassen, ebenso wie einen neuen Token zu generieren (siehe Abbildung unten). (Wikipedia – Security-Token, 2021)

## Einladen auf einen Server

Um den Bot nun auch verwenden zu können, muss er noch auf einen Discord Server eingeladen werden. Dies passiert über einen Einladungslink. Diesen kann man spezifisch für seine Applikation auf dem Discord Developer Portal erstellen lassen. Man navigiert auf der linken Seite zu OAuth2. Dort wählt man unter «Scopes» Bot aus. Daraufhin wird bereits ein Einladungslink generiert, wie in der unteren Abbildung gezeigt.

Bevor man den Link jedoch verwenden kann, sollten noch die Rechte des Bots festgelegt werden (Abbildung 10). Je nach Berechtigungen verändert sich dadurch nämlich der oben generierte Link. Sorgfalt sollte man walten lassen, wenn der Bot mit Administrator-Rechten ausgestattet werden soll, da dies ein Sicherheitsrisiko darstellen kann.

Sobald der Bot über die gewünschten Berechtigungen verfügt, muss man nur noch den generierten Link kopieren und im Browser eingeben. Anschliessend wird man automatisch zu Discord weitergeleitet und aufgefordert, sich einzuloggen.

Nach erfolgreichem Log-In muss festgelegt werden, auf welchem Server der Bot verfügbar sein soll (Abbildung 11). Des Weiteren wird man an dieser Stelle aufgefordert, die Berechtigungseinstellungen erneut zu überprüfen (Abbildung 12).

Nach der erfolgreichen Autorisierung erscheint der Bot nun auf dem Server als «Mitglied». Darüber hinaus erhalten Bots in Discord zur besseren Erkennung neben dem Namen noch einen Bot-Tag.

## Entwicklungsprozess

Hat man alle diese vorbereitenden Schritte erledigt, kann mit der Programmierung beginnen. Discord unterstützt 2 Programmiersprachen: Python und JavaScript. Ich habe mich für Python entschieden, da ich bereits über entsprechende Kenntnisse verfügte. Als Code-Editor benutze ich Visual Studio Code.

Als Erstes muss die Discord-Library eingebunden werden. Diese Library stellt das Client-Objekt zur Verfügung, welches die Basis für sämtliche Interaktionen mit Discord ist. Des Weiteren werden zwei Funktionen erstellt. Die erste Funktion («on\_ready») ist ein Verbindungstest. Es wird überprüft, ob sich der Bot erfolgreich mit dem Server verbunden hat. Ist dies der Fall wird eine entsprechende Statusmeldung ausgegeben. (Discord.py – Quickstart, 2021)

In der Funktion «on\_message» wird definiert, wie der Bot reagieren soll, wenn er eine Nachricht bekommt. Zuerst wird sichergestellt, dass der Bot nur auf Nachrichten anderer Nutzer reagiert, ansonsten könnte er auf seine eigenen Nachrichten antworten, was zu einer Endlos-Schleife führt. Danach teilt man dem Bot mit, auf welche Nachrichten er reagieren soll. Da ich einen Bot erstellen wollte, der ausschliesslich auf Nachrichten, die mit einem «!» beginnen, antworten soll, habe ich als Beispiel die Nachricht «!hallo» genommen. Wenn der Bot nun eine Nachricht mit «!hallo» bekommt, dann soll er mit «Hallo!» antworten. Durch den «==» Operator ist festgelegt, dass der Text der Nachricht exakt der Zeichenfolge entsprechen muss, die im Code festgelegt ist.

Als letzter Befehl führt «client.run()» zusammen mit der Angabe des Tokens die Anwendung auf dem Server aus. (Discord.py – Quickstart, 2021)

Während der Entwicklungsphase habe ich den Bot lokal auf meinem Computer gehostet und mittels «python bot.py» jeweils manuell gestartet, um die Funktionalität zu testen. Im späteren Echtbetrieb wurde der Bot dann auf eine Online-Plattform gebracht, um rund um die Uhr verfügbar zu sein. Die entsprechenden Schritte werden in Kapitel 4.5 noch ausführlicher erläutert.

Um die limitierte Funktionalität etwas zu erweitern, entschied ich mich dann dazu, den Bot das aktuelle Datum ausgeben zu lassen. Python stellt dafür mit «datetime» eine eigene Library zur Verfügung. Ich erstellte dafür eine Variable und wies ihr das aktuelle Systemdatum zu. Der Bot musste dann lediglich die Variable in einen String konvertieren und konnte diesen ausgeben.

Ich probierte anfangs, Informationen über das Wetter von der Seite «Weather.com» zu bekommen. Dafür verwendete ich das Modul «lxml». Es ermöglicht einem, schnell und einfach HTML-Dateien zu handhaben. Um meinen Code direkt testen zu können, verwendete ich für alles, was mit Webscraping zu tun hatte, die Applikation Jupyter Lab, welche im Anaconda Navigator zur Verfügung steht. Anaconda ist eine Plattform für Data-Science. (Stack Abuse – Introduction to the Python lxml Library, 2019)

Um «lxml» verwenden zu können, muss man das Modul erst herunterladen, da es standardmässig nicht in Python vorhanden ist. Hierzu verwendet man den Befehl «pip install lxml». Danach muss man noch das Modul «requests» und von «lxml» das Element «html» importieren. Das weitere Vorgehen wird durch «lxml» stark vereinfacht. Man erstellt eine Variable, in der die URL gespeichert wird. Danach wird die zur URL dazugehörige Seite in einer weiteren Variablen gespeichert. Diese Seite wird schlussendlich noch zu HTML-Code umgewandelt. Mithilfe der Funktion «xpath», die von «lxml» bereitgestellt wird, kann man nun den HTML-Code nach den Infos durchsuchen, die man benötigt.

Jetzt musste ich diese Funktion lediglich noch vom Bot auf Befehl ausführen lassen. Ich erstellte innerhalb des Projektordners ein neues Skript für die Webscraping Funktion. Ich konnte aus dem neu erstellten Skript anschliessend die Funktion importieren und sie bei einer Anfrage ausführen lassen.

Diese Struktur konnte ich anschliessend übernehmen, um die Spielerstatistiken des Videospiels zu bekommen. Ich musste lediglich die URL entsprechend anpassen, und die Namen der Objekte, die ich im HTML-Code der Seite suchte. Zusätzlich erstellte ich zwei weitere Funktionen, die die gewünschten Daten ausgaben. Analog zur Wetter-Funktion erstellte ich einzelne Skripts dafür im Projektordner und importierte die Funktionen ins Hauptskript.

Der Unterschied zur Wetter-Funktion war, dass der Nutzer dem Bot sagen sollte, von welchem Spieler er die Statistiken wissen wollte. Deshalb konnte der Bot die Funktionen nicht 1:1 so verwenden. Dafür erstellte ich zwei leere Strings, die je nach Nachricht an den Bot initiiert werden. Anschliessend musste der Bot noch die Nachricht splitten, um den Nutzernamen zu bekommen und diesen in die URL eingeben zu können. Die ganze Nachricht an den Bot bestand nämlich nicht nur aus dem Nutzernamen, sondern «!stats + Nutzername» oder «!operators + Nutzername».

Der letzte Schritt, den ich machen musste, damit die Funktionen korrekt ausgeführt wurden, war, die zwei Strings dann noch den jeweiligen Funktionen zu übergeben.

## Hosting

Damit man den Bot nicht immer manuell starten muss, gibt es die Möglichkeit, ihn online zu hosten. Ich habe mich dabei für die Plattform Heroku entschieden. Nachdem man einen Account eingerichtet hat, kann man eine neue App erstellen. Neben der Vergabe eines Namens kann man auch die geographische Region festlegen, in der Heroku die App hosten soll. Zur Auswahl stehen dabei Europa und die USA.

Nachdem die App erstellt wurde, wird man zur «Deploy» Seite weitergeleitet. Heroku bietet einem dort die Möglichkeit, die App mit einem GitHub Repository zu verknüpfen (siehe Abbildung 23).

Hat man das Repository und die App auf Heroku miteinander verknüpft, kann man direkt von Github aus den Bot deployen. Man kann Heroku so einstellen, dass es auf Änderungen im Repository automatisch reagiert und, ebenfalls automatisch, die aktuellste Version des Bot deployed. Damit ein Deployment fehlerfrei funktioniert, muss man einige Dateien erstellen, die von Heroku vorausgesetzt werden. Unter anderem sind dies die Dateien «requirements.txt» und «runtime.txt» sowie eine Procfile Datei. Im «requirements.txt» müssen alle Module enthalten sein, die der Bot benötigt. In «runtime.txt» wird die erwartete Version von Python festgelegt, in diesem Falle «python-3.8.7».

Im Procfile wird der «worker» für Heroku festgelegt, hier «worker: python bot.py». Der «worker» sagt Heroku, welcher Befehl ausgeführt werden soll. Dadurch wird in meinem Fall der Bot durch den Befehl «python bot.py» gestartet. Diese Dateien sind essentiell für das initiale Deployment, in der folgenden Abbildung aufgezeigt. Dabei werden die Module und die Python Version, die in den beiden «.txt» Dateien festgelegt sind, installiert.

Nach der Installation wird der Code des GitHub Repositories released und anschliessend auf Heroku deployed, d.h. der Bot, bzw. die neueste Version des Bots ist nun funktionsfähig.

Auf der Übersichtsseite von Heroku sieht man auf der rechten Seite auch die letzten Aktivitäten. Dort kann man unter anderem entnehmen, wer wann was deployed hat und wie oft deployed wurde. Zudem ist der «worker» ebenfalls ersichtlich.

# Diskussion & Reflexion

Bevor ich mich im Laufe dieser Arbeit intensiv mit dem Themenbereich der Bots auseinandersetzte, wusste ich nicht viel darüber. Ich wusste zwar, was ein Bot war, da ich einige Bots bereits verwendete, nicht jedoch, wie sie genau funktionieren oder wie vielseitig sie eingesetzt werden können.

Die Recherche für den theoretischen Teil dieser Arbeit fiel wesentlich umfangreicher aus, als ich gedacht hatte, sowohl zeitlich als auch inhaltlich gesehen. Es gibt zahllose Onlineartikel zum Thema Chatbots, weshalb es schwierig war, gute, zuverlässige Quellen zu finden.

Im Anschluss an den theoretischen Teil folgte dann der Methodenteil. Diesen habe ich in 2 Abschnitte eingeteilt. Der Erste behandelt die allgemein notwendigen Schritte und vermittelt einen groben Überblick. Der nachfolgende zweite Teil erklärt dann meinen Arbeitsprozess im Detail.

Die Arbeit mit dem Discord Developer Portal war sehr angenehm. Die einzelnen Schritte, die man ausführen musste, waren nicht komplex. Somit konnte man innerhalb kürzester Zeit eine API-Anwendung mit Bot erstellen.

Der gesamte Arbeitsprozess verlief im Gegensatz zum theoretischen Teil schneller als erwartet. Ich hatte mit grösseren Schwierigkeiten beim Webscraping gerechnet, da ich bisher noch nie Erfahrung damit gemacht hatte. Dann habe ich mich jedoch mit meinem Bruder darüber ausgetauscht und er empfahl mir das Modul «Beautiful Soup», welches das bekannteste Python Modul für Webscraping ist. Durch Tutorials wurde ich aber auf eine Alternative aufmerksam, das Modul «lxml». Nach kurzer Probezeit entschied ich mich dann für «lxml». Was den Funktionsumfang betrifft, gibt es keine signifikanten Unterschiede zwischen beiden Libraries. Ich persönlich finde es einfacher und intuitiver mit «lxml» zu arbeiten im Vergleich zu «Beautiful Soup».

Gehostet habe ich den Bot dann auf Heroku. Heroku ist sehr intuitiv, man muss nicht viel Aufwand betreiben, um einen Bot zu hosten. Die Plattform erfüllt vollkommen meine Ansprüche, weshalb ich mich nie ernsthaft mit Alternativen wie z.B. Firebase von Google beschäftigt habe. Was ich anfangs verwirrend fand, war die Heroku spezifische Datei «Procfile». Dateien wie z.B. «requirements.txt» haben für mich Sinn ergeben, da diese bei ziemlich allen Projekten zum Einsatz kommen.

Abschliessend bin ich sehr zufrieden mit meiner Arbeit. Beim Programmieren des Bots bin ich um Einiges weitergekommen, als ich erwartet hatte, jedoch konnte ich nicht all meine Ziele erreichen. Ein weiterer Punkt, den ich gerne noch ausprobiert hätte, der jedoch aus zeitlichen Gründen nicht mehr umsetzbar war, ist, dass der Bot Sprachkanälen auf Anweisung beitritt und Audiodateien abspielt. Ebenfalls möchte ich eine «Zensurfunktion» hinzufügen, die den gesamten Chatverlauf nach Nachrichten, die anstössige Wörter enthalten, durchsucht und diese löscht. Diese Funktionalitäten nehme ich mir für die Zukunft vor. Ich werde den Bot auch nach der Maturarbeit eigenständig weiterentwickeln, da Discord eine App ist, die ich praktisch jeden Tag benutze und ich ebenfalls meine Programmierfähigkeiten erweitern möchte.

# Literaturverzeichnis

Apple – Siri (2021) aufgerufen am 29.07.2021 von

<https://www.apple.com/de/siri/>

Back4App – Top 5 Alternativen zu Heroku (2021) aufgerufen am 18.08.2021 von <https://blog.back4app.com/top-5-alternativen-zu-heroku/>

Boxcryptor – Welchen Einfluss haben sie auf Gesellschaft und Politik (31.03.2020) aufgerufen am 02.08.2021 von <https://www.boxcryptor.com/de/blog/post/social-bots-detection-examples-of-political-impact/>

Cloudflare – What is a bot (2021) aufgerufen am 31.07.2021 von <https://www.cloudflare.com/de-de/learning/bots/what-is-a-bot/>

Cloudflare – How is an internet bot constructed (2021) aufgerufen am 31.07.2021 von <https://www.cloudflare.com/de-de/learning/bots/how-is-an-internet-bot-constructed/>

Degruyter – «Good Bot, Bad Bot» (15.11.2021) aufgerufen am 15.09.2021 von <https://www.degruyter.com/document/doi/10.1515/iwp-2018-0040/html>

Discord.py – Quickstart (2021) aufgerufen am 04.09.2021 von <https://discordpy.readthedocs.io/en/stable/quickstart.html>

E-Business – Wo werden Chatbots derzeit eingesetzt (12.06.2018) aufgerufen am 02.08.2021 von <https://ebusiness2020.wordpress.com/2018/06/12/wo-werden-chatbots-derzeit-eingesetzt/>

ECN – Chatbots: Vor- und Nachteile dieser Tools (06.12.2018) aufgerufen am 04.08.2021 von <https://www.ecommerce-nation.de/chatbots-vor-und-nachteile-dieser-tools/>

freshservice Blog – Die besten Chatbots für Ihr Unternehmen: Aktueller Stand und Zukunftsblick (01.02.2021) aufgerufen am 04.08.2021 von <https://freshservice.com/de/die-besten-chatbots-2021-blog/>

Geekflare – 7 zuverlässige Discord Bot-Hosting-Plattformen (30.06.2021) aufgerufen am 18.08.2021 von <https://geekflare.com/de/discord-bot-hosting/>

Geekflare – 10 besten Code- und Texteditoren für Webentwickler (04.05.2021) aufgerufen am 18.08.2021 von <https://geekflare.com/de/code-text-editors/>

Geekflare – Discord Bot hosting (22.04.2021) aufgerufen am 27.07.2021 von <https://geekflare.com/de/discord-bot-hosting/>

Onlim – Die Geschichte und Entwicklung von Chatbots (12.10.2017) aufgerufen am 27.07.2021 von <https://onlim.com/die-geschichte-und-entwicklung-von-chatbots/>

PCWELT – Die Zukunft von Chatbots: Was kommt nach dem Hype (30.04.2020) aufgerufen am 04.08.2021 von <https://www.pcwelt.de/ratgeber/Die-Zukunft-von-Chatbots-Was-kommt-nach-dem-Hype-10721738.html>

ScienceSoft – Chatbot-Entwicklung: was zu beachten ist, um einen erfolgreichen Chatbot zu erstellen (27.05.2020) aufgerufen am 04.08.2021 von <https://www.scnsoft.de/blog/einen-erfolgreichen-chatbot-erstellen>

Stack Abuse - Introduction to the Python lxml Library (10.04.2021) aufgerufen am 06.09.2021 von <https://stackabuse.com/introduction-to-the-python-lxml-library/>

Wikipedia – Bot (2021) aufgerufen am 27.07. 2021 von

<https://de.wikipedia.com/wiki/Bot>

Wikipedia – Chatbot (2021) aufgerufen am 27.07. 2021 von <https://de.wikipedia.com/wiki/Chatbot>

Wikipedia - Discord (2021) aufgerufen am 26.08.2021 von <https://de.wikipedia.org/wiki/Discord_(Software)>

Wikipedia – ELIZA (2021) aufgerufen am 29.07.2021 von

<https://de.wikipedia.org/wiki/ELIZA>

Wikipedia – Google Assistant (2021) aufgerufen am 30.07.2021 von <https://de.wikipedia.org/wiki/Google_Assistant>

Wikipedia - Honeypot (2021) aufgerufen am 27.07. 2021 von <https://de.wikipedia.org/wiki/Honeypot>

Wikipedia – Robots Exclusion Standard (2021) aufgerufen am 27.07. 2021 von <https://de.wikipedia.org/wiki/Robots_Exclusion_Standard>

Wikipedia – Security-Token (2021) aufgerufen am 29.08.2021 von <https://de.wikipedia.org/wiki/Security-Token>

# Abbildungsverzeichnis

[Abb. 1 Titelbild (Geekflare - Discord Bot hosting, 2021) 1](#_Toc83120672)

[Abb. 2 ELIZA (Wikipedia – ELIZA, 2021) 7](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120673)

[Abb. 3 Siri (Apple – Siri, 2021) 8](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120674)

[Abb. 4 Google Assistant (Wikipedia - Google Assistant, 2021) 9](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120675)

[Abb. 5 Chatbot Hype (PCWELT - Die Zukunft von Chatbots: Was kommt nach dem Hype, 2020) 12](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120676)

[Abb. 6 Discord Developer Portal - Create Application (Eigene Abbildung, 2021) 15](#_Toc83120677)

[Abb. 7 Discord Developer Portal - Create Bot (Eigene Abbildung, 2021) 16](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120678)

[Abb. 8 Discord Developer Portal - Bot Token (Eigene Abbildung, 2021) 16](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120679)

[Abb. 9 Discord Developer Portal - Create Invitelink (Eigene Abbildung, 2021) 17](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120680)

[Abb. 10 Discord Developer Portal - Bot Permissions (Eigene Abbildung, 2021) 18](#_Toc83120681)

[Abb. 11 Discord Developer Portal - Select Server (Eigene Abbildung, 2021) 19](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120682)

[Abb. 12 Discord Developer Portal - Select Permissions (Eigene Abbildung, 2021) 19](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120683)

[Abb. 13 Discord - Memberlist (Eigene Abbildung, 2021) 19](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120684)

[Abb. 14 VS-Code - Default Template (Eigene Abbildung, 2021) 20](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120685)

[Abb. 15 VS-Code - !datum Function (Eigene Abbildung, 2021) 21](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120686)

[Abb. 16 Jupyter Lab - Webscraping weather (Eigene Abbildung, 2021) 21](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120687)

[Abb. 17 VS-Code - !wetter Function (Eigene Abbildung, 2021) 22](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120688)

[Abb. 18 Jupyter Lab - Webscraping stats (Eigene Abbildung, 2021) 22](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120689)

[Abb. 19 Jupyter Lab - Webscraping operators (Eigene Abbildung, 2021) 23](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120690)

[Abb. 20 VS-Code - Username Input (Eigene Abbildung, 2021) 23](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120691)

[Abb. 21 VS-Code - Assign String to Function (Eigene Abbildung, 2021) 24](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120692)

[Abb. 22 Heroku - Create new Application (Eigene Abbildung, 2021) 24](#_Toc83120693)

[Abb. 23 Heroku - Connect GitHub Repository (Eigene Abbildung, 2021) 24](#_Toc83120694)

[Abb. 24 Heroku - First Deploy (Eigene Abbildung, 2021) 25](#_Toc83120695)

[Abb. 25 Heroku - Application Overview (Eigene Abbildung, 2021) 26](https://nksa.sharepoint.com/sites/Projektarbeiten-Chat-Bot/Freigegebene%20Dokumente/Chat-Bot/Programmierung%20eines%20Discord%20Chatbots.docx#_Toc83120696)