Universität Potsdam

Department Linguistik

Dokumentation des Skills “Mensa-Auskunft”

Projektbericht über eine Funktionserweiterung von Amazons digitalem Assistenten *Alexa*

Modul: ANW A / AM 4 – Anwendungen der CL
Seminar: Praktische Dialogmodellierung: Ein Dialogsystem erstellen
Semester: Sommersemester 2019
Dozent: Prof. Dr. David Schlangen

Autoren: Bogdan Kostić, MATRIKELNUMMER
Maria Lomaeva, MATRIKELNUMMER
Monique Noss, MATRIKELNUMMER
Olha Zolotarenko, MATRIKELNUMMER

24. Oktober 2019

Inhaltsverzeichnis

1 Zusammenfassung **2**

2 Projektziele und Funktionen **2**

2.1 Anwendungsbereich und Zielgruppe 2

2.1.1 Essensplan 3

2.1.2 Preis der Gerichte 4

2.1.3 Adresse der Mensa 4

2.1.4 Mensas in einer Stadt 4

2.1.5 Mensa in der Nähe 5

2.2 Szenarien und Beispieldialoge 5

2.2.1 Beispieldialoge: Essensplan und Preise erfahren 5

2.2.2 Beispieldialoge: Die nächste Mensa finden 7

2.2.3 Beispieldialoge: Die Adresse einer Mensa erfahren . . . 8

3 Projektorganisation **9**

4 Entwurf des Systems, Dokumentation **9**

4.1 Entwicklungsumgebung 9

4.2 Tests 10

5 Projektabschluss, Evaluation **10**

1 Zusammenfassung

...

2 Projektziele und Funktionen

Im Nachfolgenden werden die erreichten Projektziele und Anforderungen erläutert. Darüber hinaus werden die Funktionen des Skills anhand von Beispieldialogen gezeigt.

2.1 Anwendungsbereich und Zielgruppe

Bei einem Mensa-Skill, dessen Hauptaufgabe es ist, über den Essensplan einer Mensa zu informieren, sind vor allem Studierende, aber auch Dozierende die Zielgruppe. Eine kleine, aber keinesfalls unwichtige Teilgruppe dieser doch recht großen Zielgruppe sind blinde Studierende und Beschäftigte der Universitäten. Dieser Skill ermöglicht es Menschen mit Sehbehinderung leichter an Informationen über das Essensangebot an deutschen Hochschulen zu kommen und macht damit die deutsche Hochschullandschaft noch ein wenig barriefreier. Eine weitere kleine Zielgruppe dieses Skills könnten Beschäftigte eines Studienakkreditierungsinstituts sein, die oft durch ganz Deutschland reisen und viel Zeit an den unterschiedlichsten Hochschulen verbringen. Dementsprechend könnten diese Beschäftigte an den Speiseplänen vieler verschiedener Universitäten und Fachhochschulen interessiert sein. Da wir versucht haben, die ganze Bandbreite der OpenMensa API zu nutzen und somit Informationen für 490 verschiedene Mensen bereitstellen können, ist dieser Skill dazu in der Lage, diesem Interesse entgegenzukommen.
Das Informationsbedürfnis dieser Zielgruppen ist es einerseits, eine grobe Übersicht über das Angebot zu erhalten, andererseits aber auch nach bestimmten Zutaten gefilterte Ergebnisse zu erhalten, falls Allergien oder besondere Ernährungsweisen vorliegen (z.B. Veganismus). Auch der Preis eines bestimmten Gerichtes könnte für eine finale Auswahl entscheidend sein, da das Budget eines Studierenden selten großzügig ausfällt. Diese und weitere verschiedene Funktionen wurden eingebaut, um die in der OpenMensa API angebotenen Daten ausgiebig zu nutzen.
Dazu gehören auch die folgenden Lokationsfunktionen:

- Adresse von Mensen
- Koordinaten, um die nächste Mensa zu finden
- Mensen nach Stadt finden

Neben den oben genannten Funktionen, die durch selbsterstellte Intents des Skills umgesetzt wurden, bietet Amazon den Skill-Entwicklern an, dynamische, auf den Benutzer zugeschnittene Funktionen einzubauen. Ein Beispiel hierfür sind die Persist Attributes"(= "bleibende Attribute"). Persist Attributes dienen dazu, bestimmte Daten in einer Session zu speichern, damit sie in der nächsten wieder abgerufen werden können. So müsste ein Benutzer nicht bei jedem Launch mit erwähnen, für welche Mensa ein Essensplan ausgegeben werden soll. Während die Implementierung und Einbettung über AWS DynamoDB einfach vorgenommen werden kann, könnten diese beim Entwickler Kosten verursachen, weshalb wir in diesem Skill auf die Benutzung dieser Attribute verzichtet haben. Auch könnten User Accounts angelegt werden, um persönliche Vorlieben des Users speichern zu können.
Eine weitere mögliche Verbesserung haben wir während der zweiten Projektphase, also nach der ersten Vorstellung der Projekte, umsetzen können. Es handelt sich hierbei um syntaktisches Parsing der Gerichte, um nur den Kopf der "Gerichtsphraseäuszugeben und somit die Antworten des Skills zu kürzen, um den Nutzer nicht mit Informationen zu überhäufen. Aus demselben Grund werden nun immer nur vier Gerichte in einem Prompt vorgelesen. Möchte der Nutzer weitere Gerichte erfahren, kann einfach "weiter"gescrollt werden. Mit einem Intent für Details können dagegen Details zu einem Gericht erfragt werden.
Für uns war es wichtig, die gesamte Breite der API zu nutzen, um dem Nutzer so viele Funktionen wie möglich zu bieten. Nichtsdestotrotz findet die In-App-Kommunikation schnell ein Ende, wenn der Benutzer ein mögliches Ziel erreicht hat oder ein Fehler aufgetreten ist. Der Skill kann dann in einem One-Shot erneut gelaunched werden.

2.1.1 Essensplan

Gibt den Essensplan einer bestimmten Mensa für ein bestimmtes Datum aus. Optional kann eine Zutat angegeben werden, die enthalten sein soll.
Beispieläußerungen:

> "lies mir den plan für {date} vor"
> "gibt es {date} {ingredient} gerichte in der {mensa_name}"
> "was gibt's in der mensa"

2.1.1.1 Gerichte ohne Zutat

Sucht Gerichte in einer bestimmten Mensa für ein bestimmtes Datum ohne bis zu zwei Zutaten.

Beispieläußerungen:

> “suche für {date} {synonyms_gericht} ohne {ingredient} in {mensa_name}”
> “gibt es {date} {synonyms_gericht} ohne {ingredient}”
> “nach {synonyms_gericht} ohne {ingredient} bitte”

2.1.1.2 Gerichte mit Zutat

Sucht Gerichte in einer bestimmten Mensa für ein bestimmtes Datum mit bis zu zwei Zutaten.
Beispieläußerungen:

> “suche für {date} {synonyms_gericht} mit {ingredient} in {mensa_name}”
> “gibt es {date} {synonyms_gericht} mit {ingredient}”
> “nach {synonyms_gericht} mit {ingredient} bitte”

2.1.2 Preis der Gerichte

Der Skill gibt den Preis für ein Gericht zurück.
Beispieläußerungen:

> “preis für nummer {number}”
> “wie viel kostet das {number} für {user_group}”
> “wie teuer ist das {number} gericht”

2.1.3 Adresse der Mensa

Die Adresse einer Mensa wird vorgelesen.
Beispieläußerungen:

> “zeige mir die adresse der {mensa_name}”
> “standort {mensa_name}”
> “adresse {mensa_name}”

2.1.4 Mensas in einer Stadt

Listet die Mensas in einer genannten Stadt auf.
Beispieläußerungen:

"welche mensas gibt es in {city}"
"suche mensas in {city}"
"gibt es mensas in {city}"

2.1.5 Mensa in der Nähe

Findet die Mensa, die dem Benutzerstandort am nächsten ist.
Beispieläußerungen:

"wo ist die nächste mensa"
"ich bin hungrig"
"welche mensa ist in der nähe"

2.2 Szenarien und Beispieldialoge

Im nachfolgenden Absatz werden Beispieldialoge für jeden Intent aufgelistet, die Skillerfolge und -misserfolge darstellen sollen, um die Funktionalität des Skills zu verdeutlichen. Diese werden jeweils anhand von spezifischen Szenarien veranschaulicht und erklärt.

2.2.1 Beispieldialoge: Essensplan und Preise erfahren

Die am häufigsten genutzte Funktion eines Mensa-Skills wird voraussichtlich das Abfragen des Essensplans für einen bestimmten Tag in der eigenen Mensa sein. Hierbei kann der Nutzer Zutaten inkludieren, um die Auswahl von Anfang an einzuschränken. Wenn eine Auswahl getroffen wurde, wird der Benutzer gefragt, ob er Details oder den Preis eines der Gerichte erfahren möchte. Der Benutzer kann an dieser Stelle auch verneinen, woraufhin der Skill geschlossen wird.
Szenario 1: Benutzer möchte den Essensplan für eine bestimmte Mensa, für einen bestimmten Tag erfragen.
Wann? Jederzeit, wann immer der User in der Mensa essen möchte.

Situation 1.1: Morgen ist ein Werktag.

U: Alexa, öffne Mensaauskunft.

A: Willkommen bei der Mensaauskunft! Wenn du Hilfe bei der Bedienung brauchst, sag bitte HILFE. Was möchtest du wissen?

U: Was gibt es morgen in der Mensa?

A: In welcher Mensa möchtest du essen?

U: Mensa Golm.

A: Es gibt 5 Gerichte zur Auswahl: 1. Eintopf aus Kichererbsen, Linsen, Gemüse,Kartoffeln, 2. Eintopf aus Kichererbsen, Linsen, Gemüse,Kartoffeln, 3. Gulasch und 4. Hähnchenbrust „Piccata Milanese“. Möchtest du mehr Gerichte hören oder Details?

U: Mehr!

A: 5. Griechisches Kartoffelgericht. Möchtest du Details zu einem dieser Gerichte erfahren? Sag zum Beispiel: Details. oder: Wie viel kostet Gericht Nummer 2 für Studenten.

U: Wie viel kostet Gericht Nummer 2 für Studenten?

A: Das Gericht Eintopf aus Kichererbsen, Linsen, Gemüse,Kartoffeln und Rindfleisch kostet 1,6 Euro für Studenten.

— SKILL SCHLIEßT SICH —

Situation 1.2: Morgen ist Wochenende oder Feiertag.

U: Alexa, frag Mensaauskunft, was gibt es morgen in der Mensa Golm?

A: Sorry, für den ausgewählten Tag 2019-10-20 gibt es leider keinen Essensplan für mensa golm.

— SKILL SCHLIEßT SICH —

Situation 1.3: Die Mensa wurde nicht richtig verstanden (ASR Error) oder ist nicht in der Datenbank.

U: Alexa, frag Mensaauskunft, was gibt es morgen in der Mensa Golm?

A: Sorry, Essenspläne für mensa golum habe ich leider nicht im Angebot.

— SKILL SCHLIEßT SICH —

Situation 1.4: Der Benutzer sucht nach Gerichten mit einer bestimmten Zutat.

Dialog 1.4.1

U: Alexa, frag Mensaauskunft nach veganen Gerichten.
A: Für wann?
U: Morgen.
A: In welcher Mensa möchtest du essen?
U: Mensa Golm.
A: Es gibt 2 Vegan Gerichte zur Auswahl: 1. Vegane Knusperkroketten und 2. Glasnudel-Bowl. Möchtest du Details zu einem dieser Gerichte erfahren? Sag zum Beispiel: Details. oder: Wie viel kostet Gericht Nummer 2 für Studenten.
U: Stopp.
A: Guten Hunger!
— SKILL SCHLIEßT SICH —

Dialog 1.4.2

U: Alexa, frag Mensaauskunft nach Gerichten mit Sellerie.
A: Für wann?
U: Morgen.
A: Leider gibt es keine passenden Gerichte mit sellerie. Kann ich sonst noch helfen?
U: Nein.
A: Okay, tschüss!
— SKILL SCHLIEßT SICH —

2.2.2 Beispieldialoge: Die nächste Mensa finden

Ein wenig komplexer ist die Funktionalität, eine Mensa in der Nähe des Standortes des Benutzers zu finden. Aus dem Intent können daher daher unterschiedliche Reaktionen hervorgehen, je nach Status der Standortfreigabe für das benutzte Gerät und den Skill selbst.

Szenario 2: Benutzer möchte eine Mensa in seiner Nähe finden und befindet sich im Ortsteil Golm in Potsdam.
Wann? Höchstwahrscheinlich nur ein Mal, wenn der User wissen möchte, welche Mensa die nächste in der Umgebung ist.

Situation 2.1: Der User hat den Gerätestandort nicht freigegeben.

U: Alexa, frag Mensaauskunft, wo die nächste Mensa ist.

A: Um die nächste Mensa zu finden, benötige ich Deinen Standort. Bitte öffne die Alexa-App, um deinen Standort mit mir zu teilen.

— SKILL SCHLIEßT SICH —

Situation 2.2: Der User hat den Gerätestandort freigegeben.

U: Alexa, frag Mensaauskunft, wo die nächste Mensa ist.

A: Die nächste Mensa ist Mensa Golm in Karl-Liebknecht-Str. 24/25, 14476 Potsdam OT Golm.

— SKILL SCHLIEßT SICH —

2.2.3 Beispieldialoge: Die Adresse einer Mensa erfahren

Die Adresse der Mensa kann erneut erfragt werden oder es kann auch nach Adressen anderer Mensen gesucht werden.

Szenario 3: Benutzer möchte die Adresse einer Mensa erfahren.
Situation 3.1: Die Adresse der Mensa kann gefunden werden, sie ist also in der Datenbank vorhanden.

U: Alexa, frag Mensaauskunft nach der Adresse der Mensa Golm.

A: Die Adresse der mensa golm lautet Karl-Liebknecht-Str. 24/25, 14476 Potsdam OT Golm.

— SKILL SCHLIEßT SICH —

Situation 3.2: Die Adresse der Mensa kann nicht gefunden werden, sie ist also nicht in der Datenbank vorhanden. Möglich ist auch, dass ASR den User falsch verstanden hat.

U: Alexa, frag Mensaauskunft nach der Adresse der Mensa Golm.

A: Die Adresse der angefragten Mensa mensa ball um konnte leider nicht gefunden werden.

— SKILL SCHLIEßT SICH —

Szenario 4: Benutzer möchte erfahren, welche Mensen es in einer bestimmten Stadt gibt.
Situation 4.1: Der Nutzer fragt nach Mensen in einer deutschen Stadt, zum Beispiel Potsdam.

U: Alexa, frag Mensaauskunft, welche Mensas gibt es in Potsdam?

A: Es gibt die folgenden Mensas in potsdam: Mensa Am Neuen Palais, Mensa Kiepenheuerallee, Mensa Golm, Mensa Griebnitzsee, Bistro Tasty Studio Babelsberg, Ulf's Café (HPI Cafeteria), Erfrischungshalle.

— SKILL SCHLIEßT SICH —

Situation 4.2: Der Nutzer fragt nach Mensen in einer Stadt, die nicht in Deutschland liegt, zum Beispiel Tokyo.

U: Alexa, frag Mensaauskunft, welche Mensas gibt es in Tokyo?

A: Leider keine Mensas in Tokyo gefunden. Du kannst eine andere Stadt in Deutschland auswählen.

— SKILL SCHLIEßT SICH —

3 Projektorganisation

- TeilnehmerInnen, Aufgabenverteilung
- Planungsdokumente, Milestones (und ihre Dynamik über die Entwicklungsphase: Wie wurde der Plan angepasst über die Entwicklungszeit?)

4 Entwurf des Systems, Dokumentation

4.1 Entwicklungsumgebung

Das Diagram zeigt, mit welcher Software der Skill erstellt wurde. Der Skill läuft unter der Pythonversion 3.6 und importiert einige Bibliotheken, die einerseits native Python-Bibliotheken sind und andererseits von dem Alexa Skills Kit zur Verfügung gestellt wurden oder installiert werden müssen. Diese Bibliotheken müssen im selben Ordner der `lambda_function.py` liegen, damit diese auch von AWS Lambda eingelesen werden können. Für die Installation wurde `pip` und das Kommando `pip3 install -r requirements.txt -t .` benutzt. Zu den Requirements gehören:

- ask-sdk
- flask
- flask-ask-sdk

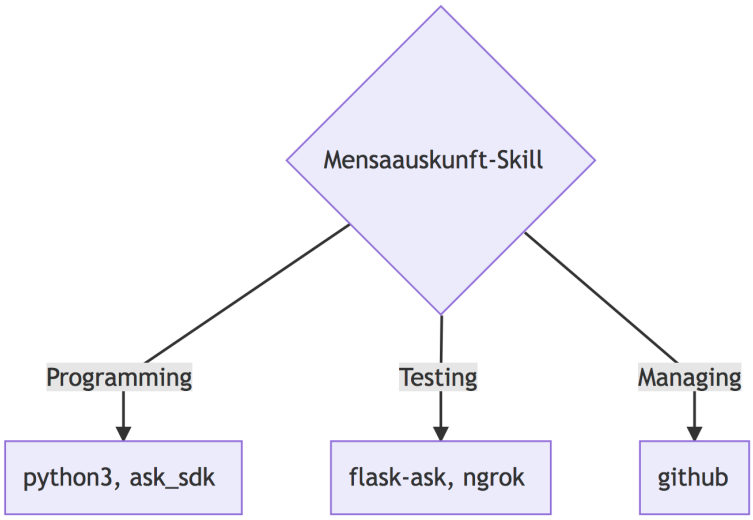

Abbildung 1: Benutzte Software während der Entwicklung des Skills

- haversine

`ask-sdk` wird benötigt, um die Kommunikation mit Alexa herzustellen. Das ganze Skill-Building basiert auf den Bibliotheken dieser Software, dazu gehören die Intent-Klassen und Funktionen sowie das Verarbeiten des Sprach-Outputs (Prompts).
`flask` und `flask-ask-sdk` wird zum Aufsetzen eines lokalen Servers verwendet. Wenn der Server gestartet ist, kann das Alexa Skills Kit mit dem auf dem Computer gespeicherten Code kommunizieren, wenn dieser lokal ausgeführt wird. Dazu wird außerdem das Tool `ngrok` benötigt, um den Port des Computers nach außen zu leiten. Diese Tools waren essenziell für das Testen unseres Skills.
`haversine` ist eine Bibliothek, die die Distanz zwischen zwei Punkten auf der Erde mithilfe des Breiten- und Längengrades kalkuliert. Diese wird benötigt, um die nächste Mensa in der Nähe des Benutzerstandortes zu finden.

4.2 Tests

Getestet wurde der Skill mithilfe von *flask* und *ngrok* lokal auf unseren eigenen Rechnern, um Status- und Fehlermeldungen angezeigt bekommen zu können, sodass sie direkt nachvollzogen werden konnten.
Dokumentation Intents => mit Sphinx

5 Projektabschluss, Evaluation

- Versuchsanordnung: VPs, Material, Methode (Fragebogen, Erfolg)

- Auswertung
- Ausblick, Verbesserungsmöglichkeiten