Problembeschreibung

Jeder kennt sie. Offene, angefangene, angeschnittene oder angerissene Lebensmittel im Kühlschrank, mit denen man nichts anzufangen weiß. Der Feind eines jeden Umweltschützers. Manchmal hat man sie nur für ein besonderes Gericht gekauft, oder man hat einfach schon alle seine Standardrezepte satt und sieht nur noch lustlos auf die übrigen Lebensmittel.

Damit man guten Lebensmitteln den letzten Weg vom Kühlschrank in den Müll zu ersparen, soll eine Anwendung entwickelt werden, mit der man ganz einfach neue Rezepte basierend auf den vorhandenen Lebensmitteln und weiteren Vorlieben, wie Ernährungsweise oder der Schwierigkeit, finden kann. Um eine komfortable Eingabe der Zutaten zu ermöglichen, soll der Benutzer nur die Lebensmittel fotografieren und hochladen müssen. Die auf den Fotos befindlichen Zutaten werden selbstständig erkannt.

Als Unterstützung bei der Vor- und Zubereitung soll es zudem eine Möglichkeit geben, die dem Nutzer hilft, einfach Informationen zu den verwendeten Zutaten abzufragen. Diese Informationen können sich von der Zubereitung über den Einkauf und Verwahrung mit allem befassen.

Lösungsansatz

Im folgenden Abschnitt werden die Lösungskonzepte für die oben erläuterte Problemstellung beschrieben.

Das System soll aus den folgenden Komponenten bestehen:

- Zutatenerkennung
- Chatbot
- Web-Scraper für Rezepte und Zutaten
- API
- Web-App

Zutatenerkennung

Damit die Zutaten auf den Fotos erkannt werden können, wird die **CustomVision API** der Cognitive Services von Microsoft verwendet. Mit Custom Vision kann man selbst eine künstliche Intelligenz trainieren, welche für den eigenen Anwendungsfall zugeschnitten Objekte in Bilder identifizieren kann.

Chatbot

Für das Abfragen von Informationen zu den Zutaten wird eine Knowledge Datenbank des QnA Maker Services von Microsoft verwendet. Der QnA Maker Service ist ein cloud-basierter Natural Language Processing Service. Er verwendet die erstellte Knowledge Datenbank, um Antworten für gestellte Fragen nachzuschlagen.

Web-Scraper

Web-Scraper werden verwendet um die Datengrundlage für das Projekt zur Verfügung zu stellen.

Um passende Rezepte vorschlagen zu können, werden diese von verschiedenen Websites abgefragt und in einer Datenbank gespeichert. Als Datengrundlage für den Chatbot zur Information zu Zutaten, wird ebenfalls eine Webseite gescraped.

API

Die API soll folgende Funktionen bieten:

- Anhand von hochgeladenen Bildern soll eine Liste von darin enthaltenen Zutaten erstellt werden.
- Basierend auf Zutaten und weiteren Tags sollen Rezeptvorschläge geladen werden.

Web-App

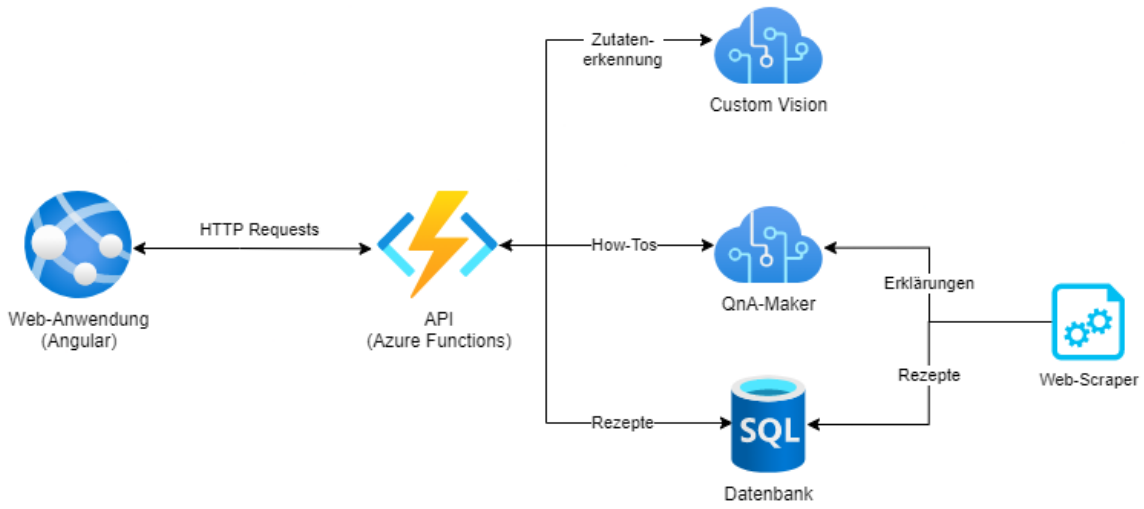
Die Web-App gibt dem Benutzer die Möglichkeit Fotos hochzuladen, weitere Zutaten sowie andere Präferenzen auszuwählen. Basierend auf diesen Angaben werden passende Rezepte angezeigt.

Implementierung

Implementationsdetails können in unserem GitHub Repository eingesehen werden:

https://github.com/gratzerl/Perry

In der folgenden Abbildung wird der Aufbau der Anwendung dargestellt. Die Web-Scraper werden, wie zuvor erwähnt, verwendet um Daten für den Chatbot (QnA-Maker) bereitzustellen und Rezepte in die Datenbank zu speichern. Als API für die Kommunikation zwischen der Web-Anwendung und den Features werden Azure Functions verwendet.



Das Backend wurde in .NET Core 3 umgesetzt, da Azure Functions bisher kein .NET 5 unterstützen und es ansonsten zu Kompatibilitätsproblemen der verschiedenen Versionen gekommen wäre.

Zutatenerkennung

Die Zutatenerkennung wird mithilfe der Custom-Vision-API der Cognitive-Services umgesetzt. Hierfür wurde im Azure-Portal eine Ressourcen-Gruppe angelegt, welche unter anderem die Cognitive-Services enthält.

Über das dazugehörige Portal (https://www.customvision.ai/) wurden Fotos von insgesamt 46 Lebensmitteln hochgeladen und die Lebensmittel in den Bildern entsprechend getaggt. Für ein initiales Training werden pro zu erkennendes Objekt mindestens 15 Tags benötigt, für optimale Ergebnisse werden 50 Tags pro Objekt empfohlen[1].

Da die Erkennung von Obst und Gemüse am zuverlässigsten funktioniert, wurde der Fokus auf diese Art der Lebensmittel gelegt. Die Erkennung von Verpackungen und Dosen wurde außen vorgelassen.

Web-Scraper

Da sowohl die Erklärungen für den Qna-Maker, als auch Rezepte von Websites gescraped werden, können die Scraper grob in diese zwei Kategorien unterteilt werden.

Für jede diese Kategorien gibt es innerhalb eines Generic-Hosts eigene BackgroundTasks. Diese BackgroundTasks bekommen über DI eine Liste mit den zu scrapenden URLs und den verfügbaren Scrapern. Innerhalb der Scraper wird für jede URL ein Scraper gestartet.

Die Funktionalität der Scraper wurde in einer abstrakten Basisklasse definiert und jeder der Scraping-Services leitet von dieser Basisklasse ab.

Rezept-Scraper

Die Scraping Services unterscheiden sich zum Teil nicht sehr, daher konnte einer der Scraper für eine zweite Seite wiederverwendet. Es wird mit viel mit Interfaces und Dependency Injection gearbeitet.

Für jede zu scrapende Website wurde ein eigener Service implementiert, welcher mithilfe des Nuget-Packages „HtmlAgilityPack“ die benötigten Daten (Titel, Beschreibung, Methode, Zutaten, Url) aus der Website ausliest.

Sobald alle Scraper fertig sind werden zunächst die neuen Tags gespeichert und anschließend die neuen Rezepte. Über die URL eines Rezepts wird sichergestellt, dass sich Rezepte, zumindest von einer Website, nicht mehrfach in der Datenbank befinden. Die Tags werden dafür verwendet um Präferenzen und Kategorien zu filtern.

Folgende Seiten werden für das Scrapen der Rezepte verwendet:

- https://www.bbcgoodfood.com/
- https://www.allrecipes.com/
- https://www.eatingwell.com/

How-To-Scraper

Der Scraper ist ähnlich wie die Rezept-Scraper aufgebaut. Der Hauptunterschied besteht hierbei darin, dass die Daten in Form eines Excel Sheets lokal abgespeichert werden. Dies ist der Fall, da man die Knowledge Datenbank des Azure QnA Makers mittels Excel Sheet füllen kann.

Folgende Seite wird für das Scrapen der How-To Daten verwendet:

- https://www.bbcgoodfood.com/glossary

[1] https://docs.microsoft.com/en-us/azure/cognitive-services/custom-vision-service/getting-started-improving-your-classifier

API und Services

Die API wurde mithilfe von Azure-Functions v3 umgesetzt. Es gibt für jede Funktion der API eine eigene Function, welche die entsprechenden Services mit der nötigen Logik über DI verwendet. Die Konfiguration der Services, wie die nötigen Verbindungsinfos für den PredictionClient, werden über die local.settings.json Datei verwaltet, wodurch diese Daten später über Umgebungsvariablen zur Verfügung stehen.

Es wurde versucht, möglichst wenig Anwendungslogik in den Functions zu implementieren und diese in eigene Services auszulagern. Bei den Functions handelt es sich um http-triggered Functions, welche folgende Routen zur Verfügung stellen:

HTTP Call	Funktion
[POST] /ingredient-identification	CustomVision Abfrage zur Zutatenerkennung
[GET] /suggestions	Datenbank Abfrage für Rezeptvorschläge
[GET]/query-answer	QnA Maker Abfrage für Zutateninformationen

Zutatenerkennung

Die Function für die Erkennung der Zutaten liest aus dem HttpRequest die hochgeladenen Bilder als Stream aus und verarbeitet diese mithilfe des entsprechenden Service aus. Dieser Service verwendet das „CustomVision“ Nuget-Package von Microsoft. Der entsprechende Prediction-Client wird dem Service über DI zur Verfügung gestellt. Die Streams der Fotos werden von dem PredictionClient verarbeitet. Als Request-Antwort werden nur jene Zutaten zurückgegeben, bei denen die Wahrscheinlichkeit größer als 83% ist.

Rezeptvorschläge

Um eine einfachere Suche zu ermöglichen, werden die Zutaten denormalisiert in einer Spalte des jeweiligen Rezepts gespeichert. Auf dieser Spalte liegt außerdem ein Fulltext-Index, welcher eine schnellere Suche nach den Zutaten ermöglichen soll. Für die Abfrage der Rezepte wurde ein Service implementiert, welcher alle Rezepte zurückliefert, die die vorgegebenen Zutaten und Tags enthalten. Die zurückgegebenen Rezepte werden anhand des Übereinstimmungsgrades der Zutaten sortiert. Da EF.Functions keine Methode für die Verwendung der ContainsTable Funktionalität bietet, musste auf die Abfrage mithilfe der Methode „FromSqlInterpolated“ umgesetzt werden.

Web-App

Die Web-App wurde mithilfe von Angular v11 und der NG-Zorro Komponentenbibliothek umgesetzt. Der Benutzer wird hier von einem Stepper durch den Abfrageprozess geführt und die gefundenen Rezepte werden anschließend in einer Liste angezeigt. Mit einem Klick auf ein Rezept wird man zur entsprechenden Website weitergeleitet.

Der eingebundene Chatbot verwendet einen selbstgeschriebenen Service, welcher die gesendeten Nachrichten und empfangenen Antworten in den SessionStorage speichert.

QnA-Maker

Der QnA Maker verwendet das „Knowledge.QnAMaker” Nuget-Package von Microsoft um einen Prediction-Client für die Abfrage auf die erstellte Knowledge Database zu starten. Es können mittels eines Prediction Clients Antworten auf übergebene Fragen abgefragt werden. Als Threshold für valide Antworten haben wir einen Score (Sicherheit der Antwort) von 83% gewählt.

Für den QnA-Maker Service wurde eine eigene Ressourcen Gruppe angelegt. Diese enhält neben dem QnA-Maker Service noch einen Search Service. Auf der zugehörigen Webseite

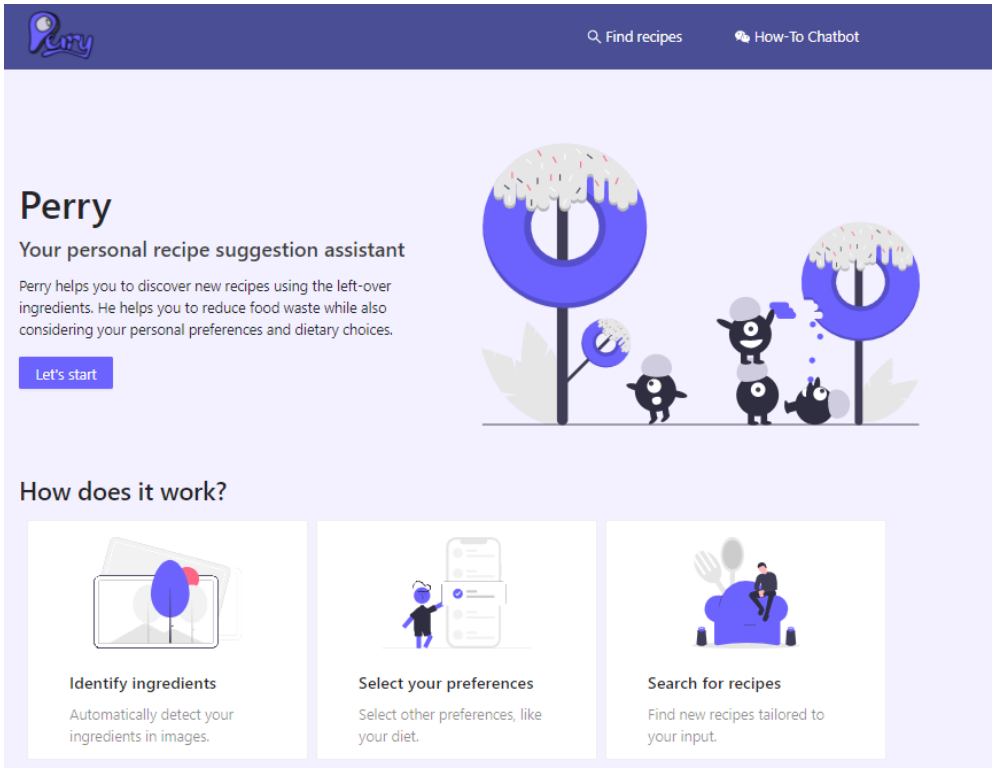
(https://www.qnamaker.ai/) kann eine Knowledge Datenbank angelegt werden. Diese kann mithilfe von Dateien oder Links zu FAQ Webseiten trainiert werden.

Azure bietet einen Azure Bot an welcher in Form eines Chatbots als Verbindung zwischen Applikation und Service dient. Diesen konnten wir aufgrund der Einschränkungen unseres Studentenkontos bei Azure leider nicht verwenden.

Ergebnisse

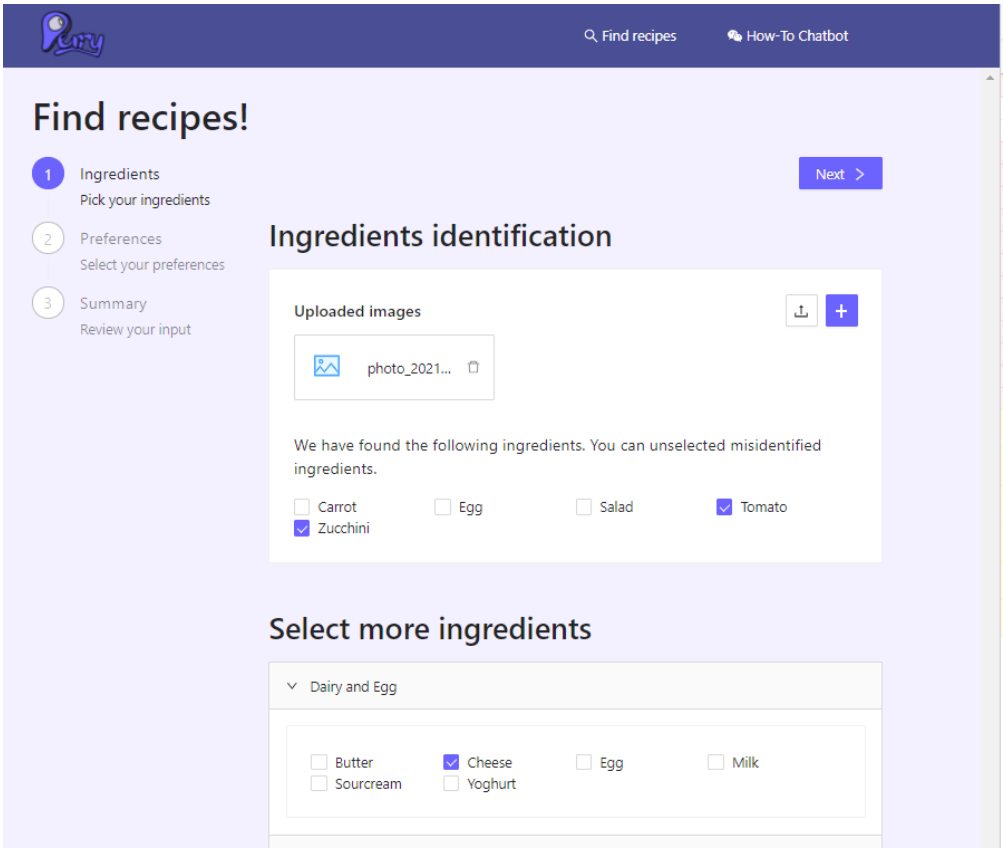
Das Ergebnis des Projektes ist eine responsive Webapplikation, welche den Nutzer bei der alltäglichen Entscheidung, welches Gericht gekocht werden kann, unter die Arme greift.

Die Landing Page gibt dem Benutzer noch einmal einen kurzen Überblick über die Nutzung der Seite.

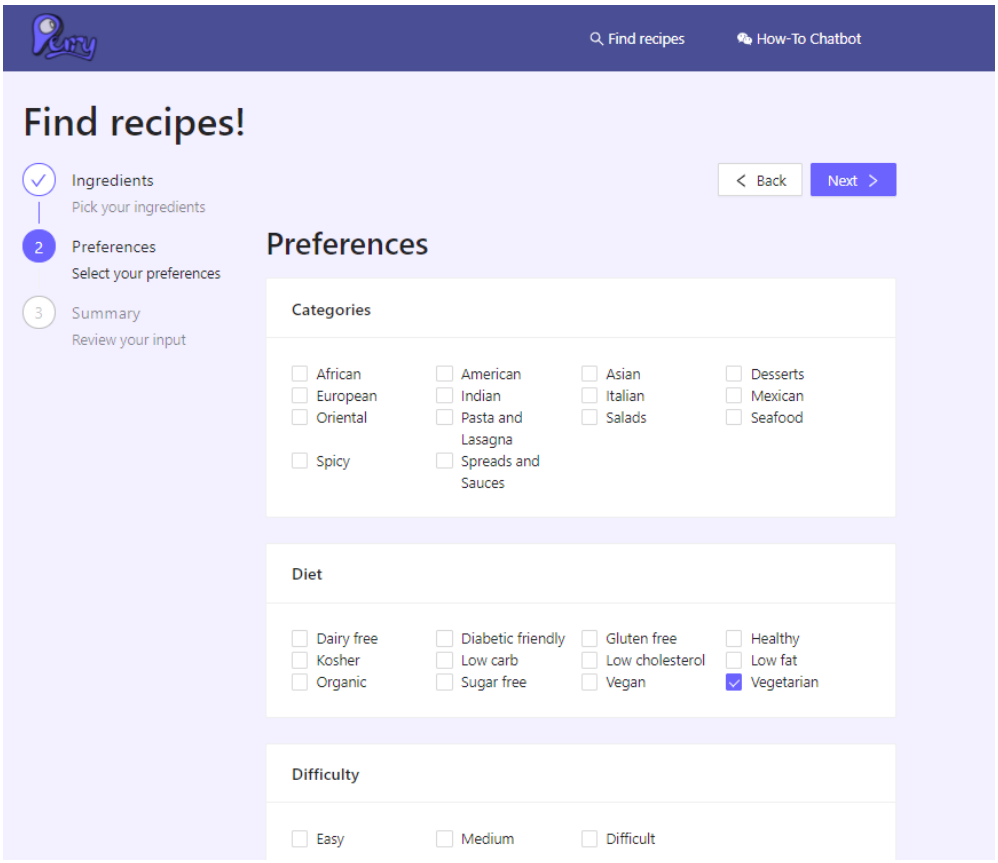


Die Rezeptsuche ist in Form eines Steppers arrangiert. Der Nutzer hat die Möglichkeit Bilder von Zutaten hinaufzuladen, welche dann erkannt werden können. Die erkannten Zutaten können nach Lust und Laune abgewählt werden.

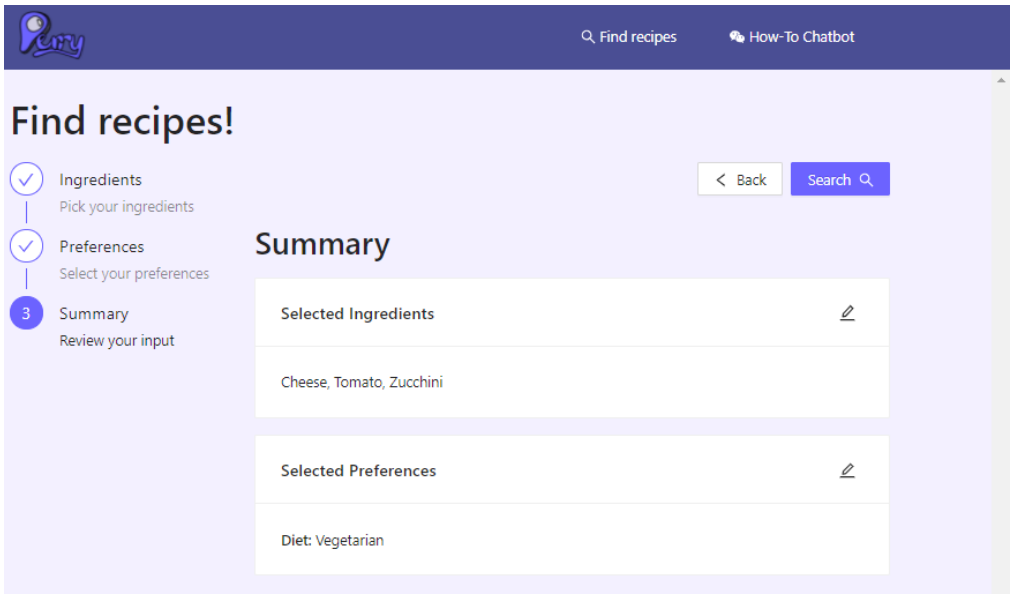
Darüberhinaus gibt es noch die Möglichkeit zusätzliche Zutaten zu wählen, welche nicht erkannt werden können. Für die angegebenen Optionen sind Werte hinterlegt, nach welchen gesucht wird. Beispielsweise wird bei der Option “Cheese” in den Rezepten zusätzlich nach “Parmesan”, “Gouda” und “Edamer” gesucht.



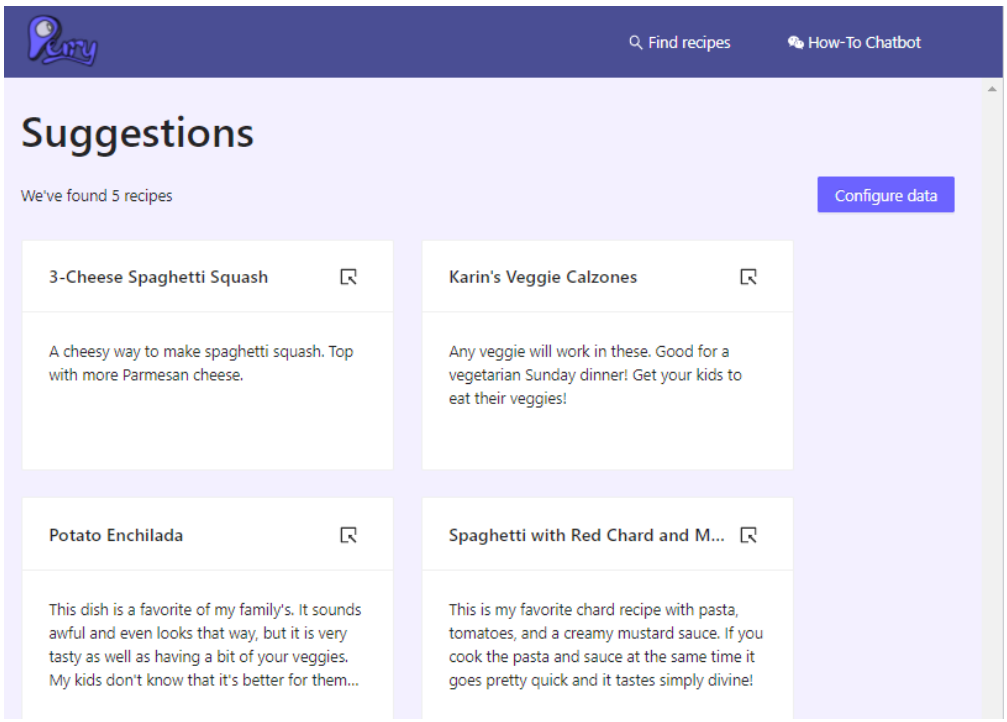
Der Preferences Step gibt dem Nutzer die Möglichkeit zusätzliche Vorlieben auszuwählen.



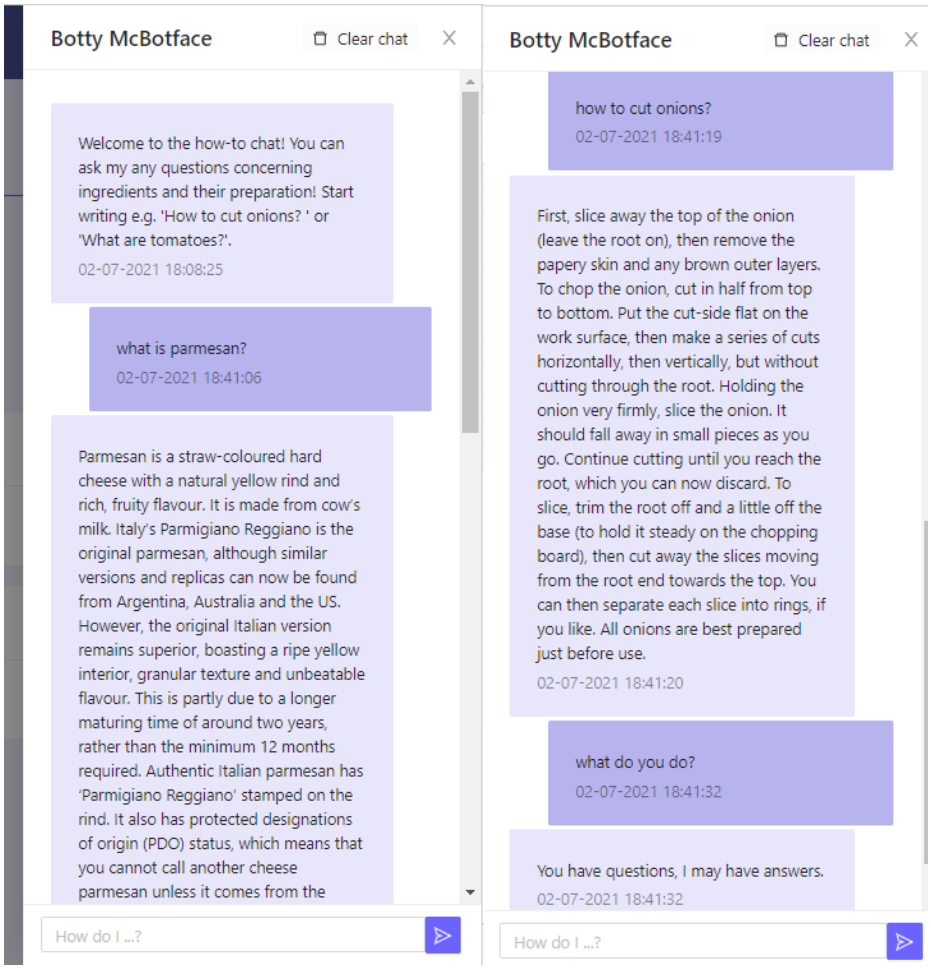
Im Summary Step bekommt der Nutzer noch einmal die Gelegenheit seine Auswahl zu kontrollieren und falls nötig anzupassen.



Auf der Vorschlagsseite werden alle gefundenen Rezepte angezeigt. Ein Klick auf ein Rezept öffnet einen neuen Tab mit dem Rezept auf der originalen Webseite.



Der QnA-Maker Chatbot bietet dem Nutzer die Gelegenheit genauere Informationen bezüglich der Zutaten, ihrer Zubereitung und Aufbewahrung abzufragen. Weiters bietet die Knowledge Datenbank eine ChitChat Funktion, welche sich um Allgemeine Fragen zum Bot selbst kümmert.



Fazit

Das Projekt ist sehr umfangreich geworden und hat viele Möglichkeiten geboten sich in verschiedene Richtungen weiterzubilden. Das Grundthema “Sustainability” wurde in Form von Vermeidung von Essensverschwendung gut aufgegriffen.

Natürlich hat es hier und da gehakt. Es gab einige Einschränkungen durch den Student Account auf Azure welche die Entwicklung etwas einschränkten.

Unser Fazit ist, dass die Arbeit an “Perry” sehr viel Spaß gemacht hat und uns die Möglichkeit gegeben hat unsere Fähigkeiten auszubauen.

Ausblick

Das entwickelte Projekt kann zurzeit lokal gestartet und einwandfrei eingesetzt werden. Durch die Umfassende Einsetzbarkeit kann man an verschiedenen Punkten ansetzen und das Projekt erweitern.

- Deployment der Datenbank, Azure Functions und Web-Applikation auf Azure
- Erkennung von Dosen mittels Custom Vision oder ähnlichem Service. Zurzeit werden nur Obst, Gemüse und andere offene Lebensmittel erkannt.

- Eigene Rezepte hinterlegen und bearbeiten. Hierfür kann die Read API verwendet werden um handschriftliche Rezepte (z.B. von Oma) einzuscannen
- Fremdsprachen Erweiterung. Die Web-App ist zurzeit schon mehrsprachig aufgebaut, allerdings sind die Rezepte zurzeit nur auf Englisch hinterlegt. Hier bietet sich eine einfache Erweiterung an.