README

Kurzbeschreibung
Die Webanwendung „Marmeladenladen“ soll dem Anwender die Möglichkeit bieten, basierend auf seinen favorisierten Zutaten ein Rezept zur Herstellung von Marmelade zu erhalten.
Hierzu wird eine Weboberfläche bereitgestellt, von deren Startseite aus der Nutzer mittels eines „Create your own recipe“-Buttons zur Rezeptzusammenstellungsseite navigieren kann. Auf dieser kann er anschließend aus einer großen Auswahl an Früchten und Gewürzen jeweils bis zu drei Fruchtsorten und ein Gewürz für seine gewünschte Marmelade wählen. Daraufhin wird dem Anwender ein Rezept mit den gewählten Fruchtsorten und dem Gewürz zur Verfügung gestellt, welches er nun nutzen kann, um seine eigene Marmelade herzustellen.

Als Datenbank wurde MySQL gewählt. Die Anwendung läuft über einen Apache http Server via XAMPP. Auf Backendseite wurde mit Python 3.7 gearbeitet, auf Frontendseite mit HTML5 und CSS3. Übergreifend kam das Webframework Django in der Version 2.2.1 zum Einsatz. Die genutzten APIs sind durch Django bedingt die sogenannte Form API, Model API und QuerySet API. Die Form API dient dabei dem Sammeln von Nutzerdaten, die daraufhin vom Server verarbeitet werden. Die Form API und Django stellen einen automatischen Validierungstest zur Verfügung, sodass das Form nur bei korrekten Input funktioniert und im Fehlerfall eine Exception wirft. Außerdem kann die Erstellung von Formularen durch die dynamische Erstellung. Die anderen beiden APIs dienen zum Einen der Nutzbarkeit und Verbindung zu einer Datenbank und zum Anderen dem Durchsuchen der Datenbank. Die bereitgestellten Methoden erleichtern die Erstellung einer DB-Anfrage, sodass anders als in PHP oder blanken Python kein String mit der SQL-Anweisung geschrieben werden muss. Die Daten der Anfrage werden in dem Datentyp „QuerySet“ gespeichert, welcher dann spezielle Methoden zur Bearbeitung und Extraktion von Daten zulässt.
Weitere Informationen bezüglich des Codes sind den Kommentaren innerhalb des tatsächlichen Codes aus dem Git-Repository zu entnehmen.

Auf Frontendseite gestaltet sich die Ausarbeitung wie folgt:
Frontpage.html ist die Startseite, welche einen „Create your own recipe“ Button besitzt. Beim Klicken des Buttons wird man zur Creatyourownrecipe.html Datei bzw. Seite weitergeleitet.
Createyourownrecipe.html zeigt daraufhin mehrere Boxen zum Auswählen an, welche Früchte oder Gewürze man in seinem Marmeladenrezept beinhaltet haben will. Durch Klicken auf das Bild oder auf den Namen der einzelnen Früchte bzw. Gewürze erscheint ein Häkchen in der Checkbox, was dem Nutzer klarmachen soll, dass seine getroffene Wahl im Rezept beinhaltet sein wird bzw. im Fruchtkorb gelandet ist.
Am Ende des Auswahlprozesses gelangt der Nutzer durch Klicken des „Submit“-Buttons zu seinem Rezept. Dieses beinhaltet die Zutaten basierend auf der Auswahl des Nutzers und eine dazugehörige Zubereitungsanweisung.
Auch hier sind alle Klassen und Funktionen innerhalb des Codes mit einem Kommentar bezüglich ihrer Funktionsweise versehen. Fur den dynamischen Aufbau der Seiten wurde das Templating von Django genutzt.

Testing:

Zum Testen des Pythoncodes wurde ein Unit Test aus der Standard-Test-Bibliothek von Python verwendet. Um die Tests laufen zu lassen, muss „python manage.py test „im Projektverzeichnis >marmelade eingegeben werden. Dadurch werden die Methoden und die Logik aus der view.py Klasse getestet.
Ende zu Ende Testing wurde durch die Verwendung von Cypress automatisiert. Um Cypress zu starten, muss „npx cypress open“ im Projektordner eingegeben werden. Die Datei marmeladeE2ETest.js ist nun unter >cypress>integration im Projektverzeichnis zu finden. Diese Tests stellen sicher, dass jede Frucht und jedes Gewürz in der Datenbank nach ihrer Auswahl auf der Website auch tatsächlich im Rezept erscheinen und dass dabei auch stets die richtigen Kombinationen von Früchten und Gewürzen dargestellt werden (dies gilt bis zu einem Limit von den auf der Website festgelegten drei Fruchtsorten und einem Gewürz).

Zuweisung der Aufgabenbereiche zu Studierenden

Datenbank	Rebekka Günther, Danielle McKenney
Backend	Jessica Rother
Frontend	Antonia Böhm, Jördis Fritzsche
Testing	Danielle McKenney
Dokumentation	Isabelle Meichsner

Lizenzierung

Installationsanweisung

Um die Webanwendung nutzen zu können, müssen Python 3, pip (in der Version für Python 3), das Framework Django und ein Webserver, beispielsweise via XAMPP, zur Verfügung stehen.
Zur Installation von Python und insbesondere Django wird auf folgendes Tutorial verwiesen: https://docs.djangoproject.com/en/2.2/topics/install/
Für die Entwicklung der Webanwendung wurde Python 3.7 verwendet, als Datenbank wurde MySQL gewählt. Zur Verwendung von MySQL muss über die Konsole noch der DB API driver „mysqlclient“ installiert werden (siehe auch: https://docs.djangoproject.com/en/2.2/ref/databases/#mysql-notes).

Anschließend muss die SQL-Tabelle „marmeladenladen_ingredients.sql“ aus dem Github-Repository über localhost/phpmyadmin in die MySQL-Datenbank eingepflegt werden. Hierfür wird erste eine neue Datenbank names „marmelade“ erstelt. Anschließend kann die sql-Datei importiert werden, sodass eine neue Tabelle mit allen Inhalten eingefügt.

Ebenfalls über die phpmyadmin-Oberfläche gilt es noch den Punkt über den Punkt „Rechte“ einen neuen Nutzer anzulegen. Der Name muss dabei „marmelade“ und das Passwort „Gelierzucker“ lauten. Der Host muss mit 127.0.0.1 festgelegt werden. Der Nutzer erhält alle Berechtigungen außer „GRANT“.

Zur Projekterstellung wurde bei der Entwicklung folgendes Tutorial zu Rate gezogen:
https://docs.djangoproject.com/en/2.2/intro/tutorial01/
Hierbei wurde jeweils der Projektname „mysite“ durch „marmelade“ und der Name der App „polls“ durch „marmeladenladen“ ersetzt.

Die automatisch generierte Settingsdatei marmelade/settings.py sollte unter dem Punkt Database folgendes enthalten:

```
77  DATABASES = {
78      'default': {
79          'ENGINE': 'django.db.backends.mysql',
80          'NAME': 'marmelade',
81          'USER': 'marmelade',
82          'PASSWORD': 'Gelierzucker',
83          'HOST': '127.0.0.1',
84          'PORT': '3306',
85      }
86  }
```

Passwort und Username der Datenbank müssen mit denen aus Django übereinstimmen.

Abschließend müssen noch Anpassungen in der setting.py durchgeführt werden, um statische Dateien wie CSS/JS/Bilder laden zu können:

```
125  STATIC_URL = '/static/'
126  STATIC_PATH = os.path.join('/Users/jessi/mysite/marmelade/marmeladenladen/static',)
127  STATICFILES_DIRS = [
128      os.path.join(BASE_DIR, 'marmeladenladen/static'),
129  ]
```

➔ Das Verzeichnis gilt es in den nutzerspezifischen Pfad, in dem der Ordner "static" liegt, umzuändern.

Nun sollte die Anwendung ausführbar sein.

Shortcut:
Weiterhin kann der Projektordner auch einfach kopiert werden. Anschließend muss „settings.py“ an den entsprechenden Stellen angepasst werden. → Static-Settings
Nach Anlegung der DB und Einpflege der bereits erklärten Pflichtinformation ist die Webseite funktionstüchtig.