

Aufgabenstellung Refrigerator(Flask)

Erstellung der Klassen

a) Erstellen Sie Klasse `Consumable`, die die Attribute `name: str`, `expiry_date: str`, `expired: bool`, `kind: str` besitzt. Dabei stellt `expiry_date` das Ablaufdatum im Format "01.01.2020" dar, `name` den Namen und `expired`, ob das Lebensmittel abgelaufen ist oder nicht. Dies soll standardmäßig auf `False` gesetzt werden. Das Attribut `kind` kann die Werte „Frucht“, „Fleisch“ oder „Gemuese“ annehmen.

Erstellen Sie ebenfalls in der Klasse eine Methode `check_expired(self, current_date: str)`, die das Ablaufdatum mit dem Datum `current_date` vergleicht und eventuell das Attribut `expired` auf `True` setzt.

b) Erstellen Sie die Klasse `Refrigeator` mit dem Attribut `consumables: List[Consumable]` und eine Instanz dieser Klasse mit beliebigem Namen (es sollen sowohl Fleisch, Gemüse und Früchte enthalten sein).

Flask-Server

a) Erstellen Sie ein Basis-Template in HTML

b) Erstellen Sie die Templates `content.html`, `kind.html`, welche von `base.html` erben und eine Funktion, die bei `@app.route("/consumables")` eine Aufzählung der 3 in Ihrem Kühlschrank gelagerten Arten von Lebensmittel gibt (`content.html`). Die 3 Elemente sollen dynamische URL's besitzen, die auf eine Seite(`kind.html`) verweisen, auf welcher jeweils alle Attribute aller Lebensmittel der bestimmten Art in Tabellenform dargestellt werden.

c) Beim Aufrufen dieser Seiten soll die Funktion `check_expired` mit einem beliebigen, vorher erstellten, konstanten Datum ausgeführt werden.

Die dynamischen URL's sollen auf den Routen

```
@app.route("/consumables/meat/<string:date>")
```

```
@app.route("/consumables/vegetables/<string:date>")
```

```
@app.route("/consumables/fruits/<string:date>")
```

die Funktionen

```
show_meat(date):, show_vegetables(date):, show_fruits(date):
```

ausführen, die `kind.html` zurückgeben (es sollen ausschließlich Lebensmittel des gewählten Typs angezeigt werden).