Woche 10 – Serverseitige Netzwerktechnologien mit Python

# Einleitung

Willkommen zur dritten und letzten Woche unseres Kurses, in der wir uns auf spannende und hochaktuelle Bereiche der Netzwerkprogrammierung konzentrieren: die serverseitige Programmierung mit Technologien wie Flask und FastAPI sowie das Webscraping mit Tools wie BeautifulSoup und Scrapy. Diese Woche ist besonders für diejenigen von Bedeutung, die in die Entwicklung von Webanwendungen und die Automatisierung von Datensammlung einsteigen oder ihre Fähigkeiten in diesen Bereichen vertiefen möchten.

In der heutigen, durch das Internet geprägten Welt ist die Fähigkeit, robuste und effiziente Webserver zu entwickeln, von unschätzbarem Wert. Flask und FastAPI bieten leistungsstarke und zugleich benutzerfreundliche Werkzeuge, um solche Anwendungen zu erstellen. Gleichzeitig wird das Webscraping immer wichtiger, um Informationen aus dem reichhaltigen Angebot des Webs zu extrahieren und nutzbar zu machen. Die Beherrschung dieser Techniken öffnet Türen zu neuen Möglichkeiten in der Webentwicklung, Datenanalyse und darüber hinaus.

Was Sie in dieser Woche lernen:

* Flask-Grundlagen
* Einführung in FastAPI
* Webscraping-Techniken
* Einsatz zur Automatisierung von Netzwerkaufgaben

Diese Woche wird Sie mit den Grundlagen und fortgeschrittenen Techniken der serverseitigen Programmierung sowie mit den Fähigkeiten des Webscrapings ausstatten. Sie erhalten das Rüstzeug, um eigene Webanwendungen zu entwickeln und wertvolle Daten aus dem Internet zu extrahieren und zu nutzen. Machen Sie sich bereit für eine intensive und lohnende Lernerfahrung, die Ihre Fähigkeiten und Ihr Verständnis in diesen Schlüsselbereichen der modernen Programmierung erweitern wird.

# Gliederung

Hier ein Überblick über die Inhalte und Aktivitäten der aktuellen Woche:

* Selbststudium:
  + Flask-Grundlagen
  + Erweiterte Flask-Entwicklung
  + Einführung in FastAPI
  + Webscraping mit BeautifulSoup und Scrapy
  + Automatisierung von Netzwerkaufgaben
* Aufgaben:
  + Erstellen Sie eine Flask-Anwendung mit grundlegenden Funktionen wie Routing und Templating.
  + Implementieren Sie fortgeschrittene Features wie Datenbankintegration und User-Authentifizierung.
  + Entwickeln Sie eine FastAPI-Anwendung, die asynchrone Anfragen effizient verarbeitet.
  + Entwerfen und implementieren Sie ein Webscraping-Skript, das spezifische Daten von einer oder mehreren Webseiten extrahiert.
* Präsenztag:
  + Wiederholung
  + Vertiefung:

# Inhalte und thematische Abgrenzung

Die folgende Auflistung zeigt detailliert, welche Themen Sie in der Woche behandeln und bearbeiten. Sie sind eine Voraussetzung für die folgenden Wochen und sollten gut verstanden worden sein. Wenn es Verständnisprobleme gibt, machen Sie sich Notizen und fragen Sie am Präsenztag nach, so dass wir gemeinsam zu Lösungen kommen können. Und denken Sie bitte immer daran: es gibt keine „dummen“ Fragen!

1. Flask-Grundlagen:

* Einführung in Flask als leichtgewichtiges Web-Framework für Python.
* Grundlagen der Erstellung von Webanwendungen und APIs mit Flask.

2. Erweiterte Webanwendungsentwicklung mit Flask:

* Entwicklung von komplexeren Webanwendungen unter Verwendung von Flask.
* Integration von Datenbanken und Implementierung von Benutzerauthentifizierung.

3. Einführung in FastAPI:

* Kennenlernen von FastAPI als modernem Framework für asynchrone Serverprogrammierung.
* Grundlagen der Erstellung von schnellen und effizienten Web-APIs mit FastAPI.

4. Webscraping-Techniken:

* Erlernen von Methoden zum Extrahieren von Daten aus Webseiten mit BeautifulSoup und Scrapy.
* Praktische Anwendung von Webscraping zur Datensammlung und -verarbeitung.

5. Serverseitige Programmierung:

* Vertiefung in die serverseitige Programmierung und deren Anwendungsfälle.
* Entwicklung und Verwaltung von Serveranwendungen, die robust, sicher und skalierbar sind.

6. Sicherheitsaspekte in der serverseitigen Programmierung:

* Diskussion über wichtige Sicherheitskonzepte für Webanwendungen.
* Implementierung von Sicherheitsmaßnahmen in serverseitigen Anwendungen.

7. Erweiterte Webscraping-Projekte:

* Durchführung komplexerer Webscraping-Aufgaben für spezifische Anforderungen.
* Nutzung fortgeschrittener Funktionen von BeautifulSoup und Scrapy.

# Lernpfad

Der Lernpfad ist ein Vorschlag, in welcher Reihenfolge Sie die Inhalte der Woche angehen können.Betrachten Sie ihn gerne als eine Todo-Liste, die Sie von oben nach unten abhaken. So können Siesicher sein, dass Sie alle wichtigen Themen bearbeitet haben und sind gut vorbereitet für die folgenden Wochen.

1. Erkundung von Flask:

* Beginnen Sie mit einer Einführung in das Flask-Framework. Installieren Sie Flask und erstellen Sie Ihre erste einfache Webanwendung.
* Experimentieren Sie mit Flask-Routings und Templating, um ein grundlegendes Verständnis für Webanwendungen zu entwickeln.

2. Erweiterte Anwendungsentwicklung mit Flask:

* Vertiefen Sie Ihr Wissen, indem Sie komplexere Anwendungen entwickeln. Integrieren Sie Datenbanken und implementieren Sie Benutzerauthentifizierung.
* Erstellen Sie eine kleine Webanwendung, die CRUD-Operationen (Create, Read, Update, Delete) ermöglicht.

3. Einführung in FastAPI:

* Lernen Sie FastAPI kennen, ein modernes Framework für asynchrone Serverprogrammierung.
* Entwickeln Sie eine einfache API mit FastAPI, die grundlegende Anfragen bearbeitet.

4. Grundlagen des Webscrapings:

* Machen Sie sich mit BeautifulSoup und Scrapy vertraut. Beginnen Sie mit einfachen Scraping-Aufgaben, wie das Extrahieren von Text aus HTML.
* Üben Sie das Sammeln von Daten von verschiedenen Webseiten und das Speichern dieser Daten in einem nutzbaren Format.

5. Komplexe Webscraping-Projekte:

* Wenden Sie fortgeschrittene Techniken an, um komplexe Datenextraktionsaufgaben durchzuführen.
* Arbeiten Sie an einem Projekt, das die gesammelten Daten für eine spezifische Analyse oder Anwendung nutzt.

6. Projektarbeit

# Programmieraufgaben

Die folgenden Programmieraufgaben sollen Ihnen eine Anregung geben. Haben Sie eigene Ideen undThemen, die Sie ausprobieren wollen, dann sollten Sie diesen nachgehen. Wichtig ist vor allem, dassSie „Dinge ausprobieren“. Und auch, dass Sie Fehler machen, sowohl syntaktische als auch semantische. Versuchen Sie diese Fehler zu finden und aufzulösen, dann gerade aus den Fehlern lernen Sie am Ende am meisten.

1. Einfache Flask Webanwendung:

Erstellen Sie eine grundlegende Webanwendung mit Flask. Entwickeln Sie eine einfache   
 Anwendung mit mehreren Routen, die verschiedene Arten von Inhalten (z.B. Text, Bilder) anzeigt.

2. Datenbankintegrierte Flask-Anwendung:

Entwickeln Sie eine Flask-Anwendung mit einer integrierten Datenbank. Erstellen Sie eine  
 Anwendung, die Benutzerregistrierung und -authentifizierung ermöglicht und Benutzerdaten in   
 einer Datenbank speichert.

3. FastAPI RESTful API:

Verwenden Sie FastAPI, um eine RESTful API zu entwickeln. Implementieren Sie eine API, die   
 grundlegende CRUD-Operationen (Create, Read, Update, Delete) für ein einfaches Datenmodell   
 (z.B. Blogposts, Kommentare) unterstützt.

4. Webscraping mit BeautifulSoup:

Extrahieren Sie spezifische Daten von einer Webseite. Verwenden Sie BeautifulSoup, um   
 Informationen wie Produktinformationen, Artikel oder Kommentare von einer Webseite zu   
 scrapen.

5. Erweitertes Webscraping-Projekt mit Scrapy:

Erstellen Sie ein umfangreiches Webscraping-Projekt mit Scrapy. Entwickeln Sie ein Scrapy-  
 Projekt, das Daten von mehreren Webseiten sammelt, verarbeitet und in einem geeigneten   
 Format (z.B. CSV, JSON) speichert.

# Abschluss-Quiz

Das Quiz soll Ihnen einen ersten Hinweis auf Ihren Lernfortschritt geben. Nach unserer Einschätzungsollten Sie diese Fragen alle beantworten können, wenn Sie den Stoff der Woche durchgearbeitetund verstanden haben. Natürlich gibt es noch sehr viel mehr mögliche Fragen, dazu wollen wir auf die Literatur und das Internet verweisen. Geben Sie gerne einmal „python quizzes“ bei Google ein.

1. Wie startet man einen Entwicklungsserver in Flask?

a) flask.run()

b) flask.start()

c) app.run()

d) app.start()

2. Welche FastAPI-Funktion wird verwendet, um eine GET-Anfrage zu definieren?

a) FastAPI.get()

b) @app.get()

c) @app.route()

d) @FastAPI.route()

3. Welches Tool wird in Python typischerweise für Webscraping verwendet?

a) Django

b) Flask

c) BeautifulSoup

d) FastAPI

4. Wie verbindet man Flask mit einer SQL-Datenbank?

a) Durch direkte SQL-Befehle.

b) Mit einer ORM wie SQLAlchemy.

c) Mit dem flask-sql Modul.

d) Flask unterstützt keine SQL-Datenbanken.

5. Wie werden Daten in einem FastAPI-Endpunkt validiert?

a) Mit JSON-Schemas.

b) Durch manuelle Überprüfung in der Funktion.

c) Mit Pydantic-Modellen.

d) Validierung ist in FastAPI nicht verfügbar.

6. Was bewirkt der scrapy.Spider in Scrapy?

a) Er dient als Webbrowser zur manuellen Navigation auf Webseiten.

b) Er ist ein Modul zur Datenbankverwaltung.

c) Er ist eine Klasse, die definiert, wie eine Webseite gescraped wird.

d) Er beschleunigt den Scraping-Prozess.

7. Wie kann man in Flask eine neue Route definieren?

a) @app.route('/neue\_route')

b) @flask.route('/neue\_route')

c) new\_route('/neue\_route')

d) Flask.new('/neue\_route')

8. Welches Flask-Modul wird verwendet, um Formulardaten zu verarbeiten?

a) Flask-Forms

b) Flask-WTF

c) Flask-Data

d) Flask-Input

9. Wie extrahiert BeautifulSoup Daten aus einem HTML-Element?

a) element.extract()

b) element.get\_data()

c) element.text

d) element.html()

10. Was ist der Hauptzweck von FastAPI im Vergleich zu Flask?

a) Schnellere Performance und Unterstützung asynchroner Anfragen.

b) Einfachere Template-Engine.

c) Bessere Integration mit Frontend-Frameworks.

d) Eingebauter Support für maschinelles Lernen.

# Ressourcen

Als Ressourcen sind die Online-Dokumentationen der verwendeten Frameworks zu nutzen.