05.02.2024

שיעור 6

HTTP

פרוטוקול תקשורת שמגדיר שפה בין 2 דברים – בין הלקוח והשרת.   
הדפדפן עובד עם הפרוטוקול. המידע שעובר הוא טקסטואלי. המטרה שלו לגרום לנוחות, כך שאפשר יהיה לקרוא את הכתובת. לעשות חיבור בין הלקוח לשרת.  
  
בקשת HTTP  
אם הלקוח פונה לשרת:

בשורת הפתיחה: יש מתודות: get, post, put, delete

למשל אנחנו פונים לכתובת, iCode – ומבקשים באמצעות get את דף index.html

Get – לבקש דף אינטרנט, שיחזיר לי מידע. לבקש להעביר פרמטרים (להביא סטודנט מהמערכת – באמצעות ID למשל, ואני מקבלת פרטי סטודנט)  
Post – כשרוצים לשלוח מידע לשרת. למשל אני רוצה להכניס פרטי ילד לתוך מאגר של משרד הפנים.

תגובת HTTP: response, כשאני שולחת בקשה, אני מקבלת תגובה. יש לי את הסטטוס של הבקשה – 200 ok – כלומר הבקשה ששלחתי התקבלה בהצלחה.

סטטוס קוד של HTTP:

משהו שמתחיל ב-2 – למשל 200 זה הצלחה

404 – או משהו שמתחיל ב4 – זה שגיאה

A white background with black text

Description automatically generated

כשיש לי סימן שאלה '?' – כשאני רוצה להפריד URL מהפרמטרים. כלומר החל מ? אני מכניסה פרמטר נגיד q=computer – הQ זה כאילו המילה שאני מחפשת בגוגל

הפרדה עושים עם & - נגיד הפרדה בין פרמטרים

אם הולכים לInspect ולnetwork – headers – אני רואה את פרטי הבקשה

A screenshot of a computer

Description automatically generated

נגיד אני רואה 200 ok – על כל מיני דברים – על תמונה ועל דברים כלליים

HTTP הוא stateless – הוא לא שומר מידע בין קריאה לקריאה, כל פעם צריך להביא לו את כל המידע. אם אני רוצה לשמור מידע אז שומרים בעוגיות או בlocal storage.

HTTPS – פרוטוקול מאובטח, משהו מוצפן כזה.

2 מתודות נוספות, חוץ מget וpost:  
put – כמו post אבל יותר מיועד לעדכן נתונים, למשל התחתנתי ואני רוצה לעדכן את השם משפחה שלי. Post יותר מיועד ליצירת דברים.

Delete – למחוק מידע מהשרת.

Crud – create – read – update - delete

**מודל שכבות**

מערכת WEB מחולקת לשכבות. צריך שכבות כדי שיהיה קל יותר ומסודר לעבוד ולשנות – אם מחליטים לשנות למשל את הUI אז מחליפים רק את שכבת הUI.

1. שכבת התצוגה UI – הריאקט למשל.
2. שכבת האפליקציה – צד שרת. הכנסה ועבודה עם הDB
3. שכבת הדאטה – הDB – אחסון.

**API** – סט של כללים שמאפשר לתוכנות לדבר אחת עם השנייה.   
Application programming interface

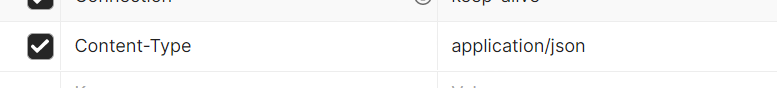
Endpoints: הAPI צריך שייתן לי להיחשף אליו. כמו גוגל שנותנים לנו google maps שאפשר להשתמש בו – דוגמה לאנדפוינט – google.com/maps

Methods – הדרכים לעבוד עם זה CRUD

Request – יחזור בJSON – ואני משתמשת בJSON הזה אח"כ כדי להציג אותו בדף.

שאר הדברים בשקופית – פחות ניגע.  
   
API STYLE – כן נלמד.  
  
REST API – עושה סדר לשימוש בAPI.

postman – מאפשר לעשות קריאות ולקבל JSON.



הוספנו את השורה הזאת בheaders

<https://developer.themoviedb.org/reference/intro/getting-started>

* לעבודה עם IMDB ל

**WEB application**

A screenshot of a computer

Description automatically generated

מורידים את הוי של HTTPS

כל הקריאות מגיעות לcontrollers

מוסיפים קונטרולר:

A screenshot of a computer

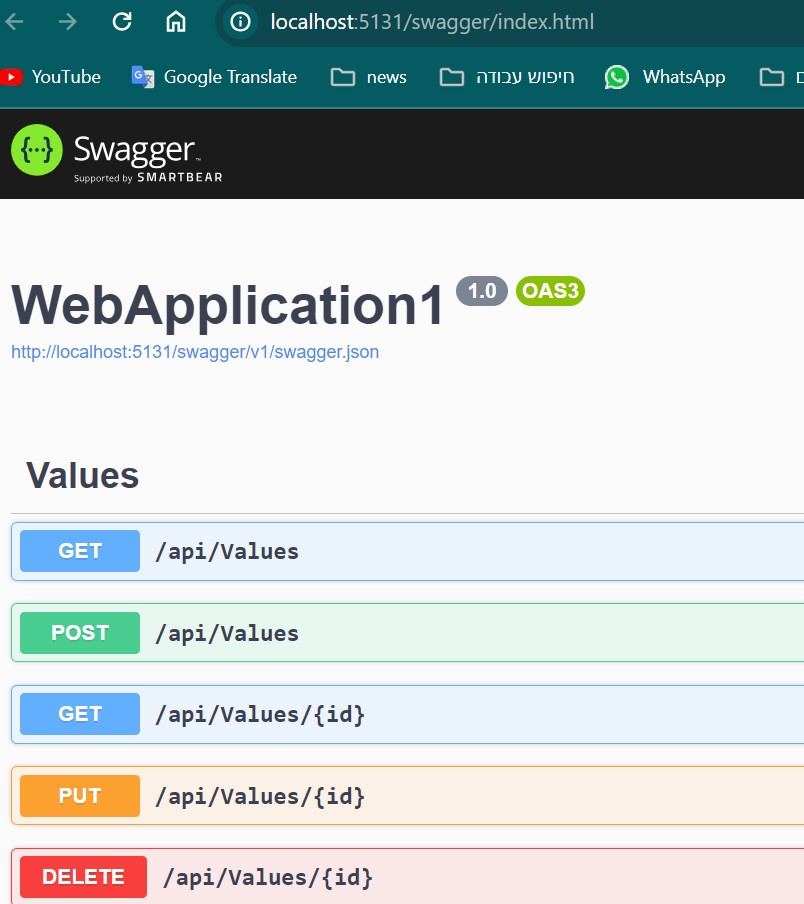
Description automatically generated

A screenshot of a computer

Description automatically generated

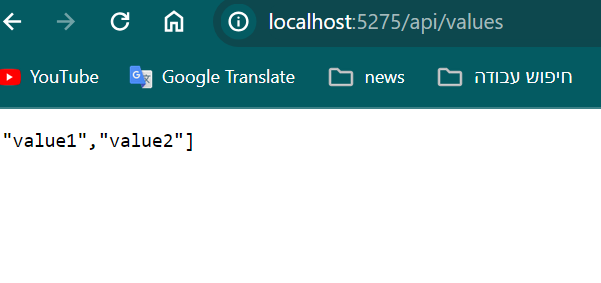
Read write – יעשה טמפלט של get put delete

אני מקבלת טמפלטים כלליים שאפשר להשתמש



כדי להפעיל עושים קונטרול F5 או פשוט להריץ. נפתח דף אינטרנט.

כדי לגשת עושים סלש, api, סלש שם הקונטרולר (values)

זה מייצר בקשת GET 

\*\*הקונטרולר חייב להסתיים עם המילה controller

שמים ברייקפוינט בPOST בסטודיו ועושים דיבאג כדי להריץ דברים.

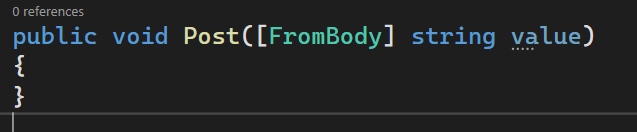
יצירת בקשה GET לפי ID, שמקבל פרמטר:  
אחרי הvalues לעשות / ולהכניס מספר, כמו 169  
A black background with white text

Description automatically generated וזה מפעיל את הפונקציה הזאתA screenshot of a computer

Description automatically generated

כדי לעשות put, delete, post אי אפשר לעשות דרך הbrowser וצריך את הpostman.

אם אני רוצה להגיע לpost: צריך להשתמש בPostman

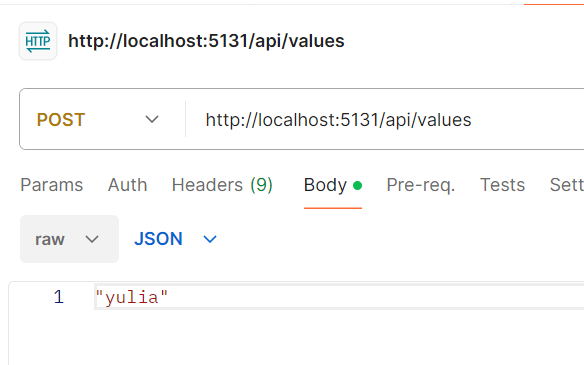


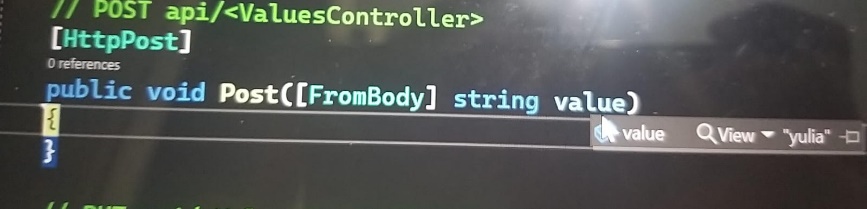
A screenshot of a computer

Description automatically generated

להכניס בצורה הזאת – שם שרת, סלש api, סלש values ולבחור בpost

זה לא יעשה כלום בעיקרון, זה סתם דוגמה מפגרת



 רואים בדיבאג שזה קיבל את הפרמטר Yulia

**חיבור השרת לבסיס הנתונים:**

יצירת קונקשן סטרינג: מוסיפים אותו לappsetting.json שיש לנו בweb application

ומעתיקים אליו את הקונקשן סטרינג מהמצגת (שקופית 56) (לעשות פסיק אחרי האובייקט שקודם לו – אחרי הallowed host)

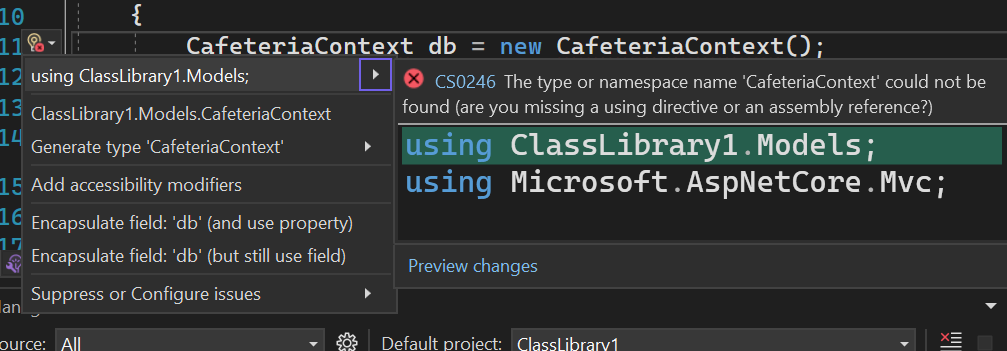
A screenshot of a computer

Description automatically generated

יוצרים קונטרולר חדש: בשם studentController

A screenshot of a computer

Description automatically generated



כותבים פה את השורה של קפיטריה קונטקסט. בעיקרון אמור להיות אופציה של add reference ל class library – כמו האופציה שלמעלה - using

A screen shot of a computer program

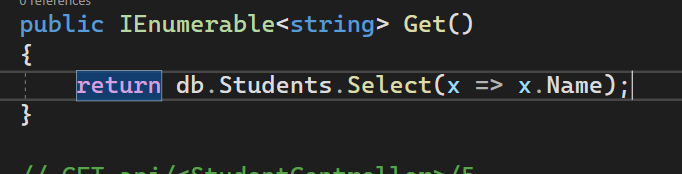
Description automatically generated

החלפנו מזה

A computer code on a black background

Description automatically generated

לזה



ואז מריצים, עושים api/student – כי זה השם של הקונטרולר ומקבלים את הרשימה של כל הסטודנטים.

A screenshot of a computer

Description automatically generated