ממשיכים full stack

FORMS

המטרה: לקבל קלט מהמשתמש ולהחזיר פלט.

Form הדרך הכי פשוטה:

# HTML

* טאג של form
  + 2 attributes:
  + **Action** 
    - ברירת מחדל זה ה URL שממנו קיבלנו את ה form.
    - זה ה endpoint על השרת שהולך לקבל את ה inputs של ה form כששולחים את ה form כלומר כשעושים submit
  + **Method**
    - שיטה שבה הדפדפן הולך לשלוח את ה INPUTS לשרת:
    - GET
      * ברירת מחדל אם לא מגדירים action
      * כשלוחצים על submit, הדפדפן אוטומטית שולח את הinputs כזוגות של name ו value על ה URL ב query string:

http://tal.com/register?username=something&email=tal@tal.com

* + - POST
      * כשלוחצים על submit, הדפדפן אוטומטית שולח בקשה ל URL לפי ה action אבל שם את ה inputs בתוך הבקשה, כלומר

[username=something&email=tal@tal.com](mailto:username%3Dsomething%26email%3Dtal@tal.com)

לא יהיה חשוף למשתמש.

בתוך ה form צריך תגיות inputs ב HTML, לכל input חייב להיות attribute בשם name, ואז הוא יישלח עם ה form.

וגם תגית input type="submit" כדי לשלוח את ה form.

# FLASK

Endpoint אחד או שניים (אחד לקבל והשני לשלוח):

Endpoint שיחזיר את את ה form, תמיד יהיה ב GET.

בפונקציה של ה route מחלצים את ה inputs לפי ה method:

GET:

* Request.args
* סוג של מילון, כלומר כדי לחלץ את ה username מהדוגמא הנ"ל, עושים request.args["username"]

POST:

* Request.form
* סוג של מילון, כלומר כדי לחלץ את ה username מהדוגמא הנ"ל, עושים request.form["username"]

תרגיל:

נכתוב form שמחזיר את ה method שלו.

תרגיל:

**לאחד את /login ו /login\_form לאותו endpoint. אותה פונקציה תטפל גם ב**

**GET וגם ב POST**

# Templates

המטרה:

להפריד את ה HTML מהפייתון, לכתוב HTML כמו front.

לשלב HTML עם לוגיקה בפייתון (דרך jinja).