Backend

איפה אנחנו?

### Frontend

HTML, CSS, JS

רצים בתוך הדפדפן

שאלה: איך הם מגיעים בכלל לדפדפן?

תשובה: מה backend!

### מה זה backend?

יש מספר חלקים ל-backend: שרת ווב, API, DB.

## שרת ווב (סטטי)

בעצם כמו שרת קבצים. הדפדפן מבקש מהשרת את ה URL, ב URL יש את שם הקובץ.

השרת מסתכל בתיקיית ה HTML שלו (!) ואם הקובץ שם הוא מחזיר אותו לדפדפן.

URL

http://127.0.0.1:3000/index.html

**http**: הפרוטוקול

**127.0.0.1**: כתובת IP מיוחדת של המחשב המקומי (שלי). נועד לבדיקות, יש שם שצמוד לכל IP, במקרה הזה localhost.

**3000**: פורט, כמו כתובת נוספת שמאפשרת להריץ כמה שרתים בו זמנית אותו IP.

הפורט הסטנדרטי של http הוא 80 או 443 במקרה של https

(לנו הפורט רלוונטי בעיקר לבדיקות)

מאחורי הקלעים: פרוטוקול TCP/IP

### שרתי ווב (או שרתי http) נפוצים:

Nginx

Apache

IIS

Express

בדרך כלל רצים בסביבת linux. חוץ מ IIS.

בשביל לראות מה קורה בין הדפדפן לשרת, נכנסים ל developer tools, network

### תרגילון:

ב VSCODE, הוסיפו מה שצריך כדי לקשר ל JS ו CSS

העתיקו את הקובץ המעודכן לתיקית html של nginx (לדרוס את הקובץ הקודם)

רפרשו את הדף בכתובת localhost בזמן שאתם עוקבים ב developer tools

אם לא עובד localhost תנסו 127.0.0.1

אמורים לראות:

Index.html

Style.css

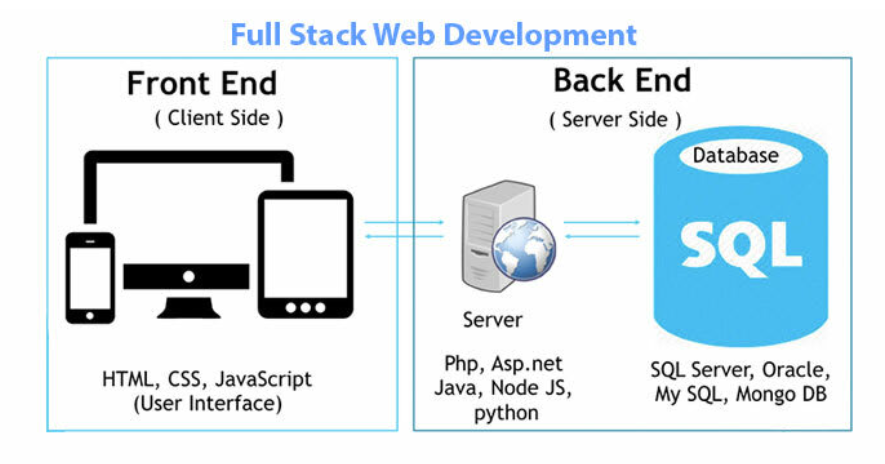
Index.js

**דיאגרמת ארכיטקטורה**

ארכיטקטורה: הרכיבים השונים והקשרים בינהם

רכיב (במבט על): דפדפן, שרת, אפשר גם HTML, JS

זה מאד מופשט לכן צריך ציור (דיאגרמה)



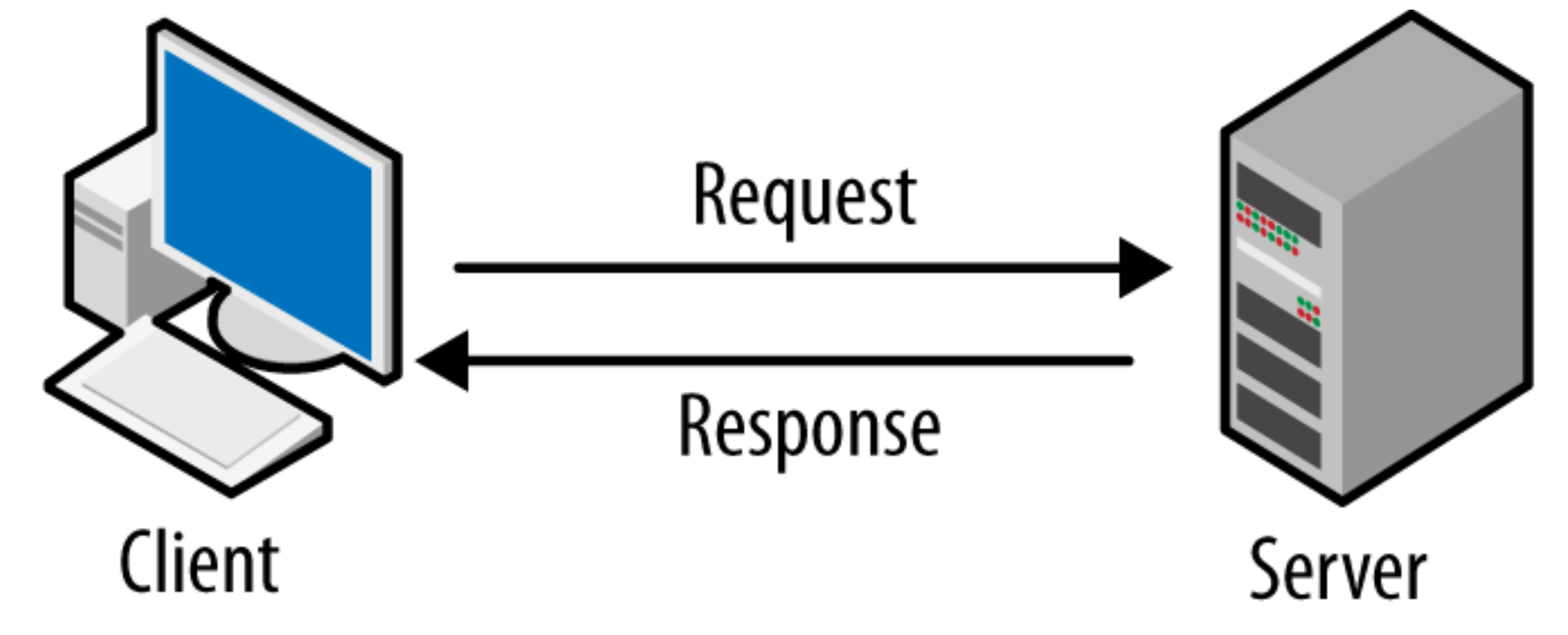
## HTTP

זה פרוטוקול תקשורת בין לקוח (דפדפן) ושרת (nginx)

התקשורת מתבצעת ע"י בקשות ותשובות

HTTP Request

HTTP Response



### HTTP REQUEST / RESPONSE

מה יש בו?

1. URL, למשל /js/index.js, הנתיב לקובץ או למשאב resource
2. HTTP Method, יש הרבה, רק כמה נפוצים
3. Headers
4. BODY (אם שולחים נתונים לשרת)

ב response יש מבנה דומה, כולל את התוכן שביקשנו ב REQUEST

### HTTP METHODS

אפשר לבקש מהשרת בקשות עם מתודות שונות.

GET: כשרוצים לקבל או להוריד משהו מהשרת

POST: כשרוצים להעלות משהו לשרת

DELETE: למחוק משהו

PUT: לעדכן משהו

יש עוד... <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>

### מה עוד יש ב backend?

## Data

מידע שמור ב database

## API

Application Programming Interface

סדרה של URLS (routes) שנותנת לקליינט (frontend) גישה ל DATABASE.

למשל:

/students

/courses

/courses/python

/courses/add

## DATABASE

יש שני סוגים עיקריים של DB

1. SQL
2. noSQL

**SQL: המידע נשמר בטבלאות, ואנחנו (המתכנתים) מתשאלים (query) את ה DB בעזרת שפת SQL.**

**Structured Query Language SQL**

כלומר צריך לדעת SQL כדי לתשאל את ה DB. נשתמש בפייתון כדי להתחבר ל DB ולכתוב SQL.

ORM: ממשק לתשאל DB ע"י קוד בפייתון (או שפת תכנות אחרת) בלי SQL, שמייצר עבורנו את ה SQL אוטומטית.

**יש 4 פעולות ב DB שחייבים לדעת (CRUD):**

**CREATE**

**READ**

**UPDATE**

**DELETE**

#### הדגמה

1. להתחבר ל DB
   1. נתחבר ל DB שנקרא SQLITE
      1. יש עוד הרבה,
         1. MySQL
         2. Oracle
         3. SQL Server
   2. ניצור טבלה לשמירה של בלוגים, נקרא לה blogs
   3. נשמור שם שני פוסטים (אפשר גם 1000 או יותר)
   4. נקרא את הפוסטים מהטבלה.

### סיכום שאילתות SQL:

CREATE TABLE

INSERT INTO

SELECT FROM

https://www.w3schools.com/sql/sql\_create\_table.asp

<https://www.w3schools.com/sql/sql_insert.asp>

https://www.w3schools.com/sql/sql\_select.asp