בס"ד

שליחת API Request

**API (Application Programming Interface):**

**API הוא בעצם סט של כלים ופונקציות שמאפשרים לשתי תוכנות שונות לדבר אחת עם השנייה.**

במקום שתוכנה אחת תצטרך לדעת איך התוכנה השנייה בנויה מפנים, הAPI מספק לה מעין "תפריט" של פעולות שהיא יכולה לבצע. הAPI מגדיר בדיוק אילו בקשות ניתן לשלוח ואיזה סוג של תשובה יש לקבל.

יהיה לנו קל יותר להבין את זה באמצעות הדוגמא הבאה: את רוצה להזמין אוכל במסעדה, אין צורך שתיכנסי למטבח בעצמך להכין עבורך את האוכל כי אם ככה אין מטרה לזה שאת מזמינה במסעדה אלא את רוצה לקבל את השירות הרצוי בדיוק בשבילך, ובדיוק בשביל זה נועד התפריט, את מסתכלת בתפריט, בוחרת את המנה הרצויה לך, ואומרת למלצר מה בחרת. המלצר מעביר את ההזמנה של למטבח, במטבח מבשלים ומכינים עבורך את האוכל שהזמנה, והמנה חוזרת אליך ע"י המלצר.

אם נפרק את הסיפור הבא וננסה להקביל אותו לעולם הAPI.

נוכל לראות שיש לנו כמה כלים ראשיים שסייעו לנו בעבודה.

תפריט – API

מנה – פונקציה

מטבח – השרת

ואילו העברת ההזמנה ממך למלצר נקראת הבקשה, והחזרת המנה מהמטבח אליך נקראת התשובה.

כדי להבין את הענין בקטע התכנותי נקח לדוגמא את שירות גימייל.

גוגל מספקים לנו ממשק גרפי שדרכו אנחנו יכולים לבצע כל מיני פעולות בחשבון המייל שלנו,

כמו: שליחת מייל, סינון רשימת המיילים, מחיקת מייל וכדומה.  
למעשה, הלחיצה על הכפתור באפליקצית הג'ימייל שולחת קריאה לשרת של גוגל ומפעילה את הפונקציה הרלוונטית ב.API –

גם את הדוגמא הזאת נוכל לפרק כמו הדוגמא הראשונה.

אין צורך שכל אחת מאיתנו תבנה פונקציה שיודעת לשלוח מייל. אלא אנחנו רק צריכות ללחוץ על הכפתור של השליחה.  
פעמים רבות הAPI- חשוף לשימוש לא רק דרך האתר הרשמי של החברה, אלא גם להפעלה דרך קוד אחר.  
אם נמשיך עם הדוגמא של ג'ימייל, גוגל פתחו את ה API- שלהם לשימוש דרך קוד חיצוני שלא שייך להם.  
כעת, נוכל לכתוב אפליקציה משלנו ששולחת מייל מחשבון הג'ימייל שלנו באמצעות ה ,API - בלי שנצטרך לממש בעצמנו את הקוד של שליחת מייל.

נוכל למצוא דוגמאות נוספות לזה כמו:

Google maps, pay pal ועוד (ההודעה על מספר ימים המלחמה).

התצורה של API הפכה לפופולרית יותר ויותר כיון שהיא מעניקה הרבה כוח ויכולות למפתחים.  
למעשה כיום זו הדרך הסטנדרטית והמקובלת לכתוב את צד השרת.

ז"א כשאני באה לבנות צד שרת מלכתחילה אני בונה אותה בצורה שנותנת אופציה לגישה לקודים אחרים.

כמובן שבבניית השרת נוכל תמיד לחסום את מי שאיננו מעוניינות שיגש לקוד אבל על זה עוד נדבר...

נעבור למושג הבא,

**Web API:**

Web API הוא סוג ספציפי של API שמשתמש בתקשורת על גבי האינטרנט, בעיקר באמצעות פרוטוקול .HTTP זהו הכלי המרכזי שמאפשר לאתרים, יישומים ניידים ושירותים שונים באינטרנט לדבר זה עם זה.

ז"א זהו שרת שנועד בשביל אתרים ומערכות שחיות על האינטרנט בניגוד למערכות API שאינם נמצאות על גבי הרשת כמו תוכנות שנמצאות על windows וכו'.

Web API הוא בעצם צד שרת שחשוף לקריות מצד הלקוח.

בסוג תקשורת הזאת שאנחנו מדברות עליה תמיד הקליינט (צד הלקוח) הוא זה שיזום את הבקשה ואילו הסרבר (צד השרת) יתפוס את הבקשה יפענח ויחזיר את התשובה.

אז כבר הבנו מי הוא הצד שרת שלנו ועכשיו נשאר לנו להבין מיהו הלקוח ועבור מי אנחנו בונות את האפליקציה.

יש לנו 3 סוגי קליינטים (הם אותו הדבר, רק משמשים כל פעם עבור משהו אחר):

* צד לקוח רגיל ומוכר שהתעסקתם איתו כבר עד היום אתרים שהכנתם (בשפת הjs).
* יש מצב שגם הסרבר הופך להיות צד לקוח הכל כמובן בצורה יחסית נחשוב על זה שאני רוצה להוסיף למערכת שלי מפות אין צורך שאני ארשום את הקוד הזה בעצמי אלא אני אוכל להתחבר לAPI של גוגל מאפס ולהציג את המפות הרצויות לי וכך נהפכתי להיות הצרכנית של אפליקציית גוגל מאפס וממילא המערכת שלי מוגדרת כקליינטית של שרת אחר.
* והדבר השלישי הוא כלי לבדיקת מערכת API: (אולי כבר יצא לכם לשמוע על חלק מערכות כאלו) יש לנו את swagger, postman כדי לבחון האם האפליקציה שלי עובדת בצורה הרצויה והטובה אין צורך שאני אבנה עבורה אתר שלם אלא יש כבר מערכת מוכנה שפשוט באמצעותה נוכל לבדוק הכל.

ועכשיו לשלבים המעשיים:

נכנס לדוגמא לקישור הבא:

<https://eventsapi.onrender.com/swagger/index.html>

תמונה שמכילה טקסט, צילום מסך, תוכנה, מערכת הפעלה

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

ונקבל את התמונה הזאת.

מה שהקישור פתח לנו זה בעצם התוכנה שבודקת מערכות API כמו שהזכרנו למעלה (תוכנת ה swagger) האפליקציה שאותה בודקים זוהי אפליקציה לניהול יומן אירועים (נוצר בשביל למידה ולכן יכול בהחלט להיות טעויות ובאגים 😊).

טוב, בואו נתחיל לשחק במסך, יש לנו פה 4 פונקציות:  
get – משמש לקבל את האירועים.

Post – משמש להוספת אירוע חדש.

Put – לעדכן אירוע קיים.

Delete – למחיקת אירוע קיים.

(להסביר כל דבר שרואים במסך)

## **מטלה - שליחת קריאות API**

עלייך לפתח אפליקציה לניהול יומן אירועים.  
האפליקציה תדע להציג לוח שנה שבו משובצים אירועים שונים.  
בנוסף היא תאפשר להוסיף אירוע, לעדכן ולמחוק.  
האפליקציה מורכבת מצד קליינט וצד סרבר.  
בצד הסרבר תשתמשי בשלב זה ב-API מוכן שמספק את הפונקציות הנדרשות לניהול נתוני האירועים.  
(אל דאגה, בהמשך תפתחי בעצמך גם את ה-API)  
בצד הקליינט תשתמשי ברכיב לוח שנה מוכן על מנת להציג את האירועים.

#### תהליך הפיתוח

בצד הקליינט עלייך לבנות דף HTML בסיסי שמציג לוח שנה.  
עם עליית הדף, תישלח בקשה לסרבר, הוא ה-API, לקבלת רשימת האירועים, והם ישובצו בלוח השנה בהתאם.  
השתמשי ברכיב לוח שנה הזה:

[https://fullcalendar.io](https://fullcalendar.io/)

צרי תיקיה חדשה לפרויקט הקליינט ופתחי אותה ב-Visual Studio Code.  
בקישור הזה תמצאי דוגמת קוד בסיסית לשימוש בלוח שנה:

<https://fullcalendar.io/docs/initialize-globals>

(כדי להגיע לדף הנ"ל לחצי בדף הבית על GetStarted ואז על Initializing with script tags)  
הוסיפי קובץ HTML והעתיקי אליו את דוגמת הקוד הראשונה שבדף הנ"ל.  
פתחי את הקובץ בדפדפן ובדקי שאכן מוצג רכיב של לוח שנה.  
🎉 👏

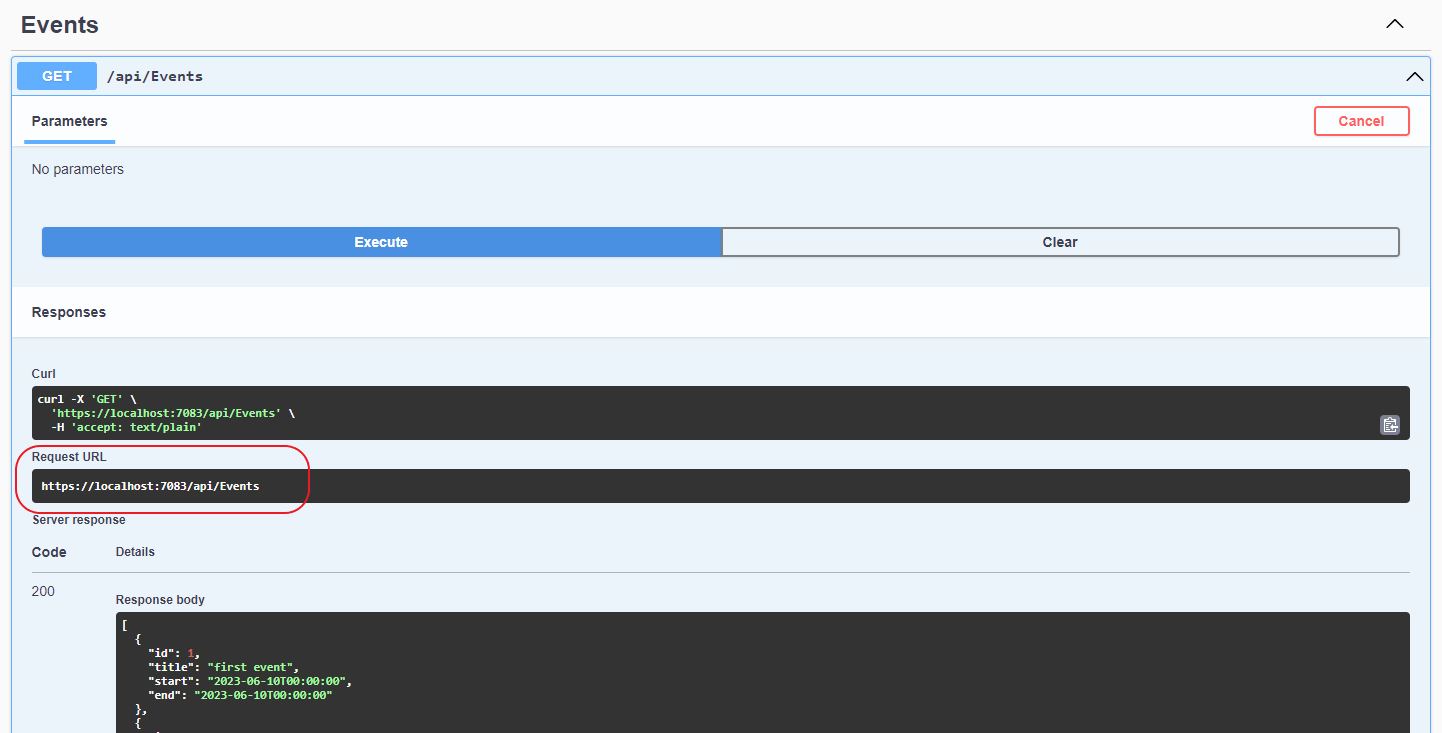
כעת, עלייך לדאוג שתישלח בקשה ל-API לקבלת רשימת האירועים בעת טעינת הדף.  
השתמשי בספריית axios כדי לשלוח בקשות HTTP מהקליינט לסרבר.

[https://axios-http.com](https://axios-http.com/)

כדי לראות את הפונקציות שה-API מספק, היכנסי לממשק ה-Swagger בכתובת הזו:

<https://eventsapi.onrender.com/swagger>

דרך ה-Swagger תוכלי לראות את מבנה הנתונים שה-API מחזיר ומצפה לקבל וכן את הקישור הספציפי לכל פונקציה ב-API. ראי בצילום המסך שלהלן את הקישור המסומן (לאחר הרצה בלחיצה על Try it out)

[](https://learn.malkabruk.co.il/assets/images/send-api-request-swagger.png)

**שימי לב**

ה-API הנ"ל נועד לצורך למידה ובתור שכזה הוא סובל מבעיות התפתחות קלות...  
יתכן שיקח זמן עד שהדף של Swagger יעלה, אנא היאזרי בסבלנות. 🤌  
בנוסף, במקרים מסוימים יתכנו שיבושים בנתונים. בדרך כלל פתיחת הקישור בדפדפן אחר (Edge לדוגמא) תפתור את הבעיה.

דוגמת קוד לשליפת רשימת האירועים ושיבוץ בלוח:

|  |  |
| --- | --- |
| **script.js** | |
| [1](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-1)  [2](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-2)  [3](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-3)  [4](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-4)  [5](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-5)  [6](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-6)  [7](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-7)  [8](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-8)  [9](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-9)  [10](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-10)  [11](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-11)  [12](https://learn.malkabruk.co.il/courses/net/basic/net-01-send-api-request/#__codelineno-0-12) | <script>  document.addEventListener("DOMContentLoaded", function () {  axios.get("https://myserver.com/events").then((res) => {  var calendarEl = document.getElementById("calendar");  var calendar = new FullCalendar.Calendar(calendarEl, {  initialView: "dayGridMonth",  events: res.data,  });  calendar.render();  });  });  </script> |

#### ממשק פעולות

הוסיפי בדף ה-HTML ממשק לפעולות: הוספה, עדכון ומחיקה של אירועים.

* בהוספה יופיעו שני שדות להזנה: כותרת ותאריך האירוע.  
  בלחיצה על כפתור הוספה תשלח קריאה ל-API להוספת אירוע חדש עם הנתונים שהוזנו.
* במחיקה יופיע שדה רשימה שיציג את רשימת כל האירועים הקימים.  
  המשתמש יבחר אירוע מהרשימה ובלחיצה על כפתור מחיקה תשלח קריאה ל-API למחיקת האירוע הנבחר.
* בעדכון יופיע שדה רשימה שיציג את רשימת כל האירועים הקימים ושני שדות לכותרת ותאריך. כאשר המשתמש יבחר אירוע מהרשימה, יתמלאו נתוני הכותרת והתאריך בשדות המתאימים והמשתמש יוכל לעדכן אותם. בלחיצה על כפתור עדכון תשלח קריאה ל-API לעדכון האירוע שנבחר עם הנתונים שהוזנו.

**שימי לב**

אין צורך להשקיע בעיצוב מיוחד, העיקר זה ההתממשקות לסרבר.

**שדרוג**

לאחר כל פעולה של הוספה/עדכון/מחיקה, בצעי רענון של הדף כדי שיופיעו הנתונים המעודכנים.  
חפשי בגוגל איך לעשות זאת.