# ‏תרגיל 3

Ajax, Promises, REST services

‏

‏מטרת התרגיל היא לבנות אתר SPA שמציג מידע ‏על משתמשים מתוך ‏האתר github, ‏וגם מאפשר שמירת מידע על השרת שלנו.

## ‏חלק א׳ Search and display (Ajax – client side)

‏אתם תבנו דף שמכיל input ‏ המאפשר להקליד שם משתמש ‏ולבצע חיפוש באתר github.

‏החיפוש מתבצע לפי login ‏של המשתמש .שימו לב שאין חשיבות לאותיות גדולות או קטנות. המטרה היא להציג מידע על עוקבים ו- repositoriesשל המשתמש.

‏החיפוש יציג את:

1. ‏ ה-login של המשתמש
2. ‏רשימת ה- public repositoriesשל המשתמש.   
   לכל אחד יהיה קישור לכתובת שלו (‏כלומר המשתמש יוכל להקליק על כל אחד ‏ויפתח אתר בהתאם).
3. ‏רשימת העוקבים של המשתמש (‏ ‏אותו עיקרון, ‏עם קישור לאתר של כל אחד)

‏ במידה והמשתמש לא נמצא, או אם ואחת הרשימות ריקה, יש להציג הודעה אינפורמטיבית (״משתמש לא קיים״, ״ ‏ ללא עוקבים״, ״ ‏ללא repositories״). ‏

‏על מנת לחלץ את המידע אתם תשתמשו במימשק (REST API) של github.

‏ניתן לחפש משתמש ב-Ajax על ידי:

**https://api.github.com/users/**${username}  
‏לדוגמא <https://api.github.com/users/nickwalton> ‏יחזיר את ה-JSON הבא :

{

"**login**": "nickwalton",

"id": 22206272,

"node\_id": "MDQ6VXNlcjIyMjA2Mjcy",

"avatar\_url": "https://avatars1.githubusercontent.com/u/22206272?v=4",

"gravatar\_id": "",

"url": "https://api.github.com/users/nickwalton",

"html\_url": "https://github.com/nickwalton",

"**followers\_url**": "https://api.github.com/users/nickwalton/followers",

"following\_url": "https://api.github.com/users/nickwalton/following{/other\_user}",

"gists\_url": "https://api.github.com/users/nickwalton/gists{/gist\_id}",

"starred\_url": "https://api.github.com/users/nickwalton/starred{/owner}{/repo}",

"subscriptions\_url": "https://api.github.com/users/nickwalton/subscriptions",

"organizations\_url": "https://api.github.com/users/nickwalton/orgs",

"**repos\_url**": "https://api.github.com/users/nickwalton/repos",

"events\_url": "https://api.github.com/users/nickwalton/events{/privacy}",

"received\_events\_url": "https://api.github.com/users/nickwalton/received\_events",

"type": "User",

"site\_admin": false,

"name": "Nick Walton",

"company": null,

"blog": "",

"location": null,

"email": null,

"hireable": null,

"bio": "Robotics, Simulation, Self Driving Vehicles.",

"**public\_repos**": 36,

"public\_gists": 0,

"**followers**": 37,

"following": 3,

"created\_at": "2016-09-14T23:27:36Z",

"updated\_at": "2019-12-07T22:26:00Z"

}

‏ קישור github: <https://developer.github.com/v3/users/#get-a-single-user>

‏‏כפי שאתם רואים יוחזר JSON data הכולל ‏מספר רב של תכונות. ‏מומלץ מאוד לחקור  
 (chrome inspector – network tab) את התקשורת בין הדף הלבין github על מנת להבין את מבנה התשובה.

אנחנו מעוניינים ב:

1. : **login** ‏זהו השם המשתמש
2. **public\_repos**: ‏מספר ה- repos ציבורים של המשתמש
3. **repos\_url**: ‏כתובת URL שתאפשר לחלץ את רשימת ה- repos . ‏זאת אומרת שתצטרכו לבצע קריאה נוספת על מנת לחלץ את הרשימה.  
   קישור github: <https://developer.github.com/v3/repos/#list-all-public-repositories>  
   1. ‏‏עבור כל repo ‏נקבל JSON data המכיל את התכונות שלו:
      1. **full\_name:** שם
      2. **html\_url:** קישור ל-repo
4. **followers**: ‏מספר העוקבים
5. **followers\_url**: כתובת URL שתאפשר לחלץ את רשימת ה- עוקבים . ‏זאת אומרת שתצטרכו שוב לבצע קריאה נוספת על מנת לחלץ את הרשימה.
   1. עבור כל עוקב ‏נקבל JSON data המכיל את התכונות שלו:
      1. **login** ‏זהו השם משתמש
      2. **html\_url:** קישור ל-עוקב

קישור: <https://developer.github.com/v3/users/followers/#list-followers-of-a-user>

‏מטרת התרגיל לי להשתמש ב-Ajax ‏על מנת לחלץ את המידע ולהציג אותו בדף.

ראו דוגמא של עיצוב אפשרי‏קישור לסרטון בסוף המסמך.

‏דרישות:

1. ‏יש להשתמש ב Promises ‏ולנצל בצורה יעילה את הקוד ה-asynchronous ‏
2. ‏יש לבצע קריאות ajax בצורה יעילה:
   1. ‏העברת נתונים ב-JSON ובצורה מינימלית (אין להעביר מידע לא שימושי(
   2. ‏אין לבצע קריאות מיותרות
3. ‏כמו תמיד האתר רספונסיבי
4. יש לנקות רווחים בקלט ‏ולעבוד ב -case insensitive
5. אין לאחסן מידע בצד הלקוח : כל המידע מגיע מ-github ומהשרת שלנו (חלק ב׳) ‏

## ‏אזהרה

‏יש להימנע מלהריץ את התרגיל מספר רב מאוד של פעמים מתוך אותה כתובת IP. יש חשש שהשרת github יחסום אתכם. ‏אם זה קורה ‏תצטרכו להירשם כמפתח באתר github.

## ‏חלק ב׳ Saving users (REST services – server side)

‏בחלק הזה אתם תבנו בצד השרת ממשק REST API ‏במטרה לשמור רשימה מועדפת של משתמשים. ‏אין להשתמש במסד נתונים ולכן הרשימה ‏תשב בזיכרון השרת (‏משתנה גלובאלי) ותתאפס בכל ההרצה מחדש.

‏תוסיפו לחלק א׳.

1. ‏ ‏אפשרות שמירה על מנת ‏להוסיף משתמש ‏ברשימת המועדפים. ‏ניתן לדוגמא ‏להוסיף כפתור שמירה של תוצאת החיפוש.
2. ‏ אפשרות מחיקה מרשימת המועדפים. ‏ניתן לדוגמא להוסיף כפתור ‏מחיקה על מנת למחוק את המשתמש ‏ששמו מופיע ב-input.

‏רשימת המועדפים תופיע בדף ותכלול רשימה של שמות משתמשים עם קישור לאתר github של כל אחד (‏ראו את ה סרטון בסוף המסמך).

‏

‏הערה כללית:

‏אין להשאיר על הדף אלמנטים שלא רלוונטיים לתוכן הדף (‏לדוגמא הודעת שגיאה כבר לא ‏תקפה או כפתור ‏פעיל - ‏ שכן ניתן ללחוץ עליו - ‏אבל ללא ‏אפקט).

באופן כללי כאשר בונים אתר יש לדאוג לכול קלט לא חוקי או פעולה לא צפויה של המשתמש. המטרה היא לשמור על התוכן של האתר ‏ותמיד להציג מצב עדכני והודעות אינפורמטיבית. ניתן לדוגמא לנטרל רכיבים ‏כאשר שימוש בהם לא רלוונטי. ‏דוגמה נוספת: ‏הרשימה שמופיע על ‏הדף צריכה לשקף בדיוק את האמת בצד השרת. ‏אם נמחק משתמש מהרשימה זה חייב ‏להתעדכן מיד על הדף ‏ללא צורך בריענון ‏ ‏ידני בדפדפן.

## הגשה:

‏יש להגיש יחד עם הפרויקט קובץ readme.html ( ‏תקראו שוב את ההנחיות באתר של הקורס). ‏הקובץ יכלול ‏בנוסף 3 סעיפים עם המספור:

1. ‏תיאור קצר של כל קריאת Ajax
   1. ‏URL
   2. פרמטרים (‏שליחה)
   3. תוכן התשובה (‏קבלה): ‏התכונות הרלוונטיות לפתרון שלכם
2. ‏ ‏הסבר קצר איך דאגתם לזרימת ‏אופטימאלית של התכנית ע״י קוד asynchronous - בעזרת Promises -
3. ‏הסבר קצר על צד השרת: תיאור הממשק REST ‏שלכם (‏סוג הבקשה http, URL והפרמטרים שלה)

## ‏סרטון הדגמה

<https://youtu.be/ctKqU0GksSM>

A screenshot of a cell phone

Description automatically generated