# **הקדמה: Wix CMS REST API**

Wix מספקת REST API המאפשר גישה ליכולות ה-CMS (מערכת ניהול תוכן) של האתר באופן חיצוני. באמצעות ה-API ניתן לבצע פעולות קריאה וכתיבה של נתונים המאוחסנים באוספי ה-CMS (Database Collections) של Wix, בדומה לשימוש ב-Wix Data API הפנימי, אך בהתקשרות HTTPS סטנדרטית​

[dev.wix.com](https://dev.wix.com/docs/rest#:~:text=REST%20Docs%20,Stores%20product%20catalog%20and)

. לדוגמה, מפתח יכול להוסיף רשומה חדשה לאוסף, לעדכן רשומה קיימת או למחוק רשומה – והכול מתוך יישום צד-שרת (Node.js וכד') או אפליקציה חיצונית. כדי להשתמש ב-API, יש תחילה **להגדיר הרשאות וגישה**: ניתן לעשות זאת על-ידי רישום אפליקציה ב-Wix Developers וקבלת אסימוני OAuth מתאימים, או לחלופין (אם אתה בעל האתר) באמצעות יצירת מפתח API ייעודי.

**הגדרת גישה (בקצרה):** אם בוחרים בגישת OAuth, יש ליצור אפליקציה בממשק המפתחים של Wix, להגדיר לה את ההרשאות הדרושות (Scopes) עבור ה-CMS, ולצרף כתובת redirect לשרת שלכם. לאחר מכן מתקינים את האפליקציה באתר ה-Wix הרצוי (באמצעות קישור הזמנה/התקנה), מה שמפעיל את תהליך ההרשאה (OAuth consent flow). בסיום התהליך, Wix תספק לשרת שלכם **קוד autoriztion code** חד-פעמי ו-**instanceId** (מזהה מופע האפליקציה באתר) כחלק מקריאת ה-redirect​

[stackoverflow.com](https://stackoverflow.com/questions/68065780/wix-api-how-to-get-access-token-for-using-rest-api-for-my-app#:~:text=In%20order%20to%20use%20the,your%20app%20in%20his%20site)

​

[stackoverflow.com](https://stackoverflow.com/questions/68065780/wix-api-how-to-get-access-token-for-using-rest-api-for-my-app#:~:text=On%20your%20app%20Redirect%20URL,save%20some%20extra%20data%20there)

. את ה-auth code הזה מחליפים בקריאה ל-EndPoint המתאים של Wix (כמו /oauth/access) תמורת **Access Token** ו-**Refresh Token**​

[beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=%7B%20,%3CAUTH_CODE%3E)

​

[beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=Response%3A)

. לאחר שיש לכם Access Token תקף, תוכלו להתחיל לקרוא ל-REST API של Wix.

**מפתח API (חלופה):** בעלי אתר יכולים גם להשתמש במפתח API במקום ב-OAuth. מפתח API מונפק דרך ממשק Wix (ניהול אתר ב-Wix Studio) עבור חשבון המשתמש, ומשמש כאסימון גישה קבוע. בגישה זו אין צורך בתהליך OAuth, אך השימוש מוגבל לבעל האתר עצמו. בהמשך נפרט על אופן השימוש בכל אחת מהשיטות.

# **אימות והרשאות (Authentication) ב-Wix API**

**Access Token ו-Refresh Token:** ה-Access Token הוא אסימון ההרשאה המשמש לביצוע קריאות API בשם המשתמש/אתר. הוא מקודד את ההרשאות (Scopes) שהוקצו לאפליקציה. חשוב לציין שאסימון זה תקף לזמן קצר יחסית (בסביבות **5 דקות** בלבד במנגנון OAuth הישן​

[beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=,credentials%20every%20time%20one%20expires)

), ולאחר פקיעתו יש להשתמש ב-Refresh Token כדי לבקש Access Token חדש מבלי שהמשתמש יצטרך להתחבר מחדש​

[beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=access%20specific%20parts%20of%20a,credentials%20every%20time%20one%20expires)

. במודל OAuth 2.0 של Wix, לאחר קבלת ה-authorization code, שולחים אותו יחד עם מזהה האפליקציה (App ID) והסוד (App Secret) אל Wix ומקבלים חזרה access\_token ו-refresh\_token​

[beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=%7B%20,%3CAUTH_CODE%3E)

​

[beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=The%20API%20will%20return%20the,token%20and%20the%20refresh%20token)

. יש לשמור את ה-Refresh Token במקום בטוח – הוא ארוך-טווח ומשמש לקבלת אסימונים חדשים לאורך חיי ההרשאה.

**Instance ID (מזהה מופע אפליקציה):** זהו מזהה ייחודי המייצג את התקנת האפליקציה שלכם באתר Wix מסוים​

[dev.wix.com](https://dev.wix.com/docs/build-apps/develop-your-app/access/app-instances/about-app-instances#:~:text=An%20app%20instance%20is%20an,app%20is%20uninstalled%20and%20reinstalled)

. ה-instanceId נשאר קבוע עבור אותו אתר גם אם האפליקציה מוסרת ומותקנת מחדש​

[dev.wix.com](https://dev.wix.com/docs/build-apps/develop-your-app/access/app-instances/about-app-instances#:~:text=An%20app%20instance%20is%20an,app%20is%20uninstalled%20and%20reinstalled)

. Wix מספקת את ה-instanceId בעת התקנת האפליקציה (כפי שתואר בכתובת ה-redirect) או דרך webhook של התקנה. כדאי לשמור מזהה זה לשימוש עתידי. למשל, בגישת OAuth החדשה של Wix, ניתן להשתמש ב-instanceId ישירות כדי לבקש Access Token בכל עת (במקום לעבוד עם Refresh Token): קריאה ל-API של Wix עם ה-instanceId, App Secret ו-App ID תיצור Access Token חדש התקף לכ-4 שעות​

[dev.wix.com](https://dev.wix.com/docs/build-apps/develop-your-app/access/authentication/authenticate-using-oauth#:~:text=1,in%20the%20HTTP%20body)

. גישה זו מפשטת את הצורך בניהול Refresh Token, אך שתי הגישות מקובלות.

**מנגנון ההרשאה בבקשות API:** בכל קריאת API אל Wix יש לשלוח Header מתאים לצורך אימות. עבור OAuth, יש לכלול בכותרת Authorization את הערך Bearer <Access\_Token>​

[dev.wix.com](https://dev.wix.com/docs/build-apps/develop-your-app/access/authentication/authenticate-using-oauth#:~:text=2.%20Store%20the%20,is%20valid%20for%204%20hours)

. עבור מפתח API, שולחים את המפתח עצמו ב-Authorization (ללא המילה "Bearer"). בנוסף, **יש לכלול תמיד את מזהה החשבון (Account ID) ומזהה האתר (Site ID) בכותרות** כדי לזהות לאיזה אתר מתייחסת הבקשה​

[reddit.com](https://www.reddit.com/r/WixHelp/comments/1e99yb3/using_apis_to_manage_collections/#:~:text=Solved,others)

. מזהה החשבון הוא ה-UUID של חשבון בעל האתר (או מי שהנפיק את המפתח), ומזהה האתר הוא ה-UUID הפנימי של אתר ה-Wix. למעשה, Wix דורשת את שני הערכים הללו בכל בקשה ברמת אתר (Site-level API calls) כאשר משתמשים במפתח API, וכדאי לצרף אותם גם בעת שימוש ב-Access Token כדי למנוע דו-משמעות. אם משתמשים במפתח API של בעל האתר, יש לוודא שמזהה החשבון שמועבר הוא של **בעל האתר עצמו** – קריאות עם מפתח או חשבון של משתמש שאינו הבעלים יידחו​

[reddit.com](https://www.reddit.com/r/WixHelp/comments/1e99yb3/using_apis_to_manage_collections/#:~:text=https%3A%2F%2Fdev.wix.com%2Fdocs%2Frest%2Fbusiness,key%20created%20by%20the%20owner)

. לאחר אימות ה-Bearer token או ה-API key וולידציה של ה-siteId/accountId, Wix תאפשר גישה לפעולות שההרשאות (Scopes) מתירות. למשל, כדי לעבוד עם ה-CMS עליכם לכלול בהרשאות האפליקציה את הזכויות לנהל תוכן באתר (CRUD על אוספי נתונים). כמו כן, יש לוודא שהאוסף (Data Collection) מוגדר ב-Wix כ-"Published" ושהרשאות הגישה לאוסף אינן חוסמות את הפעולה. אם למשל האוסף מוגבל לקריאה/כתיבה רק לאדמין, יש לוודא שהקריאה מתבצעת עם אסימון של אדמין (למשל בעל האתר או אפליקציה מותקנת עם הרשאות מתאימות), אחרת תתקבל שגיאה כדוגמת WDE0027: the current user does not have permissions המצביעה על חסימת הרשאה.

# **הוספת, עדכון ומחיקת רשומות – דוגמאות ב-Node.js**

לאחר שהגדרנו את מנגנון האימות, נוכל לגשת לקריאות ה-API של Wix CMS. כתובת הבסיס לפניות ל-Wix Data (ה-CMS) היא https://www.wixapis.com/wix-data/v2/. בפניות אלו נשלח את ה-Access Token ב-Authorization (עם Bearer ), את ה-accountId וה-siteId בכותרות, וכן גוף הבקשה בפורמט JSON לפי הצורך. להלן דוגמאות ליישום פעולות CRUD (Create, Read, Update, Delete) בסיסיות על אוסף נתונים, באמצעות ספריית **axios** ב-Node.js:

**➡️ הוספת רשומה חדשה (Create):** כדי להוסיף פריט חדש לאוסף, נבצע קריאת **POST** לנתיב /items. בגוף הבקשה שולחים את ה-dataCollectionId (מזהה או שם האוסף) ואת האובייקט dataItem המכיל את הנתונים. בתוך dataItem אפשר (לא חובה) לציין שדה id כדי לקבוע מזהה ידני לפריט; אם לא נציין, Wix ייצור ID ייחודי אוטומטית. תחת dataItem.data יופיעו כל שדות הנתונים של הרשומה. למשל, עבור אוסף בשם "Affiliates" עם שדות **contactName** ו-**contactEmail**, גוף הבקשה ייראה כך:

json

Copy

{

"dataCollectionId": "Affiliates",

"dataItem": {

"id": "nywh46x",

"data": {

"contactName": "Rodney Allen",

"contactEmail": "rakess70@gmail.com",

"referralCode": "nywh46x"

}

}

}

בדוגמה זו הוספנו באופן ידני את השדה id עם ערך "nywh46x" (נניח שהוא קוד ייחוס שאנחנו רוצים כ-ID)​

[forum.wixstudio.com](https://forum.wixstudio.com/t/wde0027-the-current-user-does-not-have-permissions-to-insert-on-the-collection/64028#:~:text=Data%20to%20be%20sent%20to,com%E2%80%9D%2C%20%E2%80%9CreferralCode%E2%80%9D%3A%20%E2%80%9Cnywh46x%E2%80%9D)

. ניתן להשמיט את ה-id והמערכת תשמור את הפריט עם מזהה שהיא תייצר. יש לוודא שכותרות הבקשה כוללות את ה-Access Token ואת מזהי החשבון והאתר כפי שהוסבר לעיל. למשל, בקוד Node.js:

js

Copy

const axios = require('axios');

const headers = {

Authorization: `Bearer ${ACCESS\_TOKEN}`,

'wix-account-id': WIX\_ACCOUNT\_ID,

'wix-site-id': WIX\_SITE\_ID,

'Content-Type': 'application/json'

};

const newItem = {

dataCollectionId: 'Affiliates',

dataItem: {

data: {

contactName: 'Rodney Allen',

contactEmail: 'rakess70@gmail.com',

referralCode: 'nywh46x'

}

}

};

const response = await axios.post(

'https://www.wixapis.com/wix-data/v2/items',

newItem,

{ headers }

);

console.log("New item ID:", response.data.item.id);

לאחר הקריאה, במידה והפעולה הצליחה, Wix תחזיר אובייקט עם המידע שנוסף, כולל ה-ID (אם לא סופק על ידיכם).

**➡️ עדכון רשומה קיימת (Update):** כדי לעדכן פריט קיים באוסף, נשתמש בקריאת **PUT** לנתיב /items/{itemId} (כאשר {itemId} הוא מזהה הרשומה לעדכון). בגוף הבקשה נשלח מבנה דומה להוספה: dataCollectionId ואת אובייקט dataItem עם הנתונים החדשים. חשוב לכלול בתוך dataItem את ה-id של הפריט (שאמור להתאים ל-{itemId} בנתיב), לצד data עם השדות שאנו מעדכנים. ה-API של Wix מבצע החלפה מלאה של תוכן הרשומה בערכים המסופקים – כלומר, הוא יחליף את הנתונים הקיימים בערכים שב-dataItem.data בבקשה​

[dev.wix.com](https://dev.wix.com/docs/rest/business-solutions/cms/data-items/update-data-item#:~:text=Updates%20an%20item%20in%20a,data%20in%20the%20request)

. לפיכך, אם לא שולחים שדה שקיים בפריט, ייתכן שהערך הישן שלו יימחק או יישאר ללא שינוי (בהתאם לאופן המימוש – בויקס, אי-שליחת שדה כנראה תשאיר אותו ללא שינוי **אך** כדי להיות בטוחים, כדאי לכלול את כל השדות שרוצים שיישארו).

דוגמה בקוד Node.js לעדכון שדה מסוים ברשומה: נניח שיש לנו אוסף "Projects" ובו פריט עם מזהה מסוים, ואנו רוצים לעדכן את השדה **projectStatus** שלו לערך חדש:

js

Copy

const itemId = '<ITEM\_ID\_TO\_UPDATE>';

const updatePayload = {

dataCollectionId: 'Projects',

dataItem: {

id: itemId,

data: {

projectStatus: 'Completed' // עדכון שדה projectStatus לערך "Completed"

}

}

};

await axios.put(

`https://www.wixapis.com/wix-data/v2/items/${itemId}`,

updatePayload,

{ headers }

);

בדוגמה זו headers מוגדר כפי שהודגם קודם (כולל Authorization וכו'). גוף הבקשה שולח את מזהה האוסף "Projects", ואת הנתונים החדשים של הפריט. כאשר Wix מקבלת קריאת עדכון, היא תחפש את הפריט עם המזהה הנתון באוסף, ואם הוא קיים – תחליף את נתוניו בנתונים שב-dataItem.data שסופקו​

[dev.wix.com](https://dev.wix.com/docs/rest/business-solutions/cms/data-items/update-data-item#:~:text=Updates%20an%20item%20in%20a,data%20in%20the%20request)

. (אם הפריט לא קיים, סביר להניח שתוחזר שגיאה HTTP 404). שימו לב: במקרים מסוימים, ייתכן שיהיה שדה מערכת של "Revision" שיש להעביר כדי למנוע התנגשויות עריכה, אך לפי תיעוד Wix, פעולת ה-Update מחליפה את הנתונים ללא דרישה מפורשת לשדה revision (מערכת Wix מנהלת זאת מאחורי הקלעים). אם מתקבלת שגיאת 400, יש לוודא שמבנה ה-JSON תואם בדיוק לצפוי – למשל, לכלול את ה-id ברמת dataItem (ולא כשדה בתוך data).

**➡️ מחיקת רשומה (Delete):** למחיקת פריט מאוסף, נבצע קריאת **DELETE** לנתיב /items/{itemId} עם ציון האוסף. גם כאן, הפורמט ב-Wix דורש לצרף את dataCollectionId כדי לזהות מאיזה אוסף למחוק את הפריט. מכיוון שקריאות DELETE לא תמיד תומכות בגוף בקשה בכל ספריות HTTP, יש לוודא שספריית HTTP שבה אתם משתמשים מאפשרת שליחת גוף בבקשת DELETE (ב-axios ניתן לשלוח באמצעות { data: { ... } } בפרמטרים הנוספים). דוגמה למחיקת פריט:

js

Copy

const itemToDelete = '<ITEM\_ID\_TO\_DELETE>';

await axios.delete(

`https://www.wixapis.com/wix-data/v2/items/${itemToDelete}`,

{

headers,

data: { dataCollectionId: 'Projects' }

}

);

הקריאה תסיר את הפריט מהאוסף "Projects". אם הפריט נמחק בהצלחה, תקבלו תגובה ללא תוכן מיוחד (לרוב סטטוס 204 No Content). במידה והפריט היה מקושר (referenced) מפריטים אחרים, לפי תיעוד Wix ייתכן שהקישורים הללו יוסרו אוטומטית בעת המחיקה​

[dev.wix.com](https://dev.wix.com/docs/rest/business-solutions/cms/data-items/remove-data-item#:~:text=Removes%20an%20item%20from%20a,fields%2C%20those%20fields%20are)

.

**הערה:** קריאות **קריאה (Read)** גם קיימות (כגון GET לפריט בודד, או POST לנתיב /items/query לביצוע שאילתה מרובת תוצאות), אך בהתמקדות על עדכון רשומות – הנ"ל מכסה את עיקר העבודה. ניתן תמיד לעיין בתיעוד הרשמי של Wix לקבלת מבנה מלא של כל Endpoint (למשל “Insert Data Item”, “Update Data Item”, “Remove Data Item” וכדומה במסגרת ה-Wix REST API).

# **מגבלות ודגשים חשובים (Limitations & Considerations)**

1. **תוקף אסימוני הגישה (Tokens):** כפי שצוין, Access Token הניתן מ-Wix תקף לזמן קצר מאוד (בדגש על OAuth legacy – כ-5 דקות)​  
   [beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=,credentials%20every%20time%20one%20expires). לכן, אם אתם משתמשים במנגנון OAuth ו-Refresh Token, יש לממש לוגיקה במערכת שלכם לרענון האסימון באופן שוטף – למשל, לבדוק לפני כל קריאה אם האסימון פג (או עומד לפוג) ובמידת הצורך לקרוא ל-Refresh (עם ה-Refresh Token) כדי לקבל Access Token חדש​  
   [beehexa.com](https://www.beehexa.com/devdocs/wix-api-how-to-get-an-access-token-and-refresh-the-access-token/#:~:text=access%20specific%20parts%20of%20a,credentials%20every%20time%20one%20expires). לחלופין, במודל OAuth העדכני יותר של Wix, ניתן פשוט לקרוא ליצירת אסימון חדש באמצעות ה-instanceId בכל פעם שהקודם פג (מה שמייתר את הצורך בניהול מחזור חיים ארוך של Refresh Token). שימו לב גם שה-Refresh Token עצמו עשוי לפוג אם לא נעשה בו שימוש לאורך זמן רב או אם המשתמש הסיר את ההרשאה לאפליקציה.
2. **הגבלות קצב (Rate Limiting):** Wix מטילה מגבלות על קצב וכמות הקריאות ל-API. נכון להיום, מגבלת הקצב היא בסביבות **200 קריאות בדקה לכל instance (התקנת אפליקציה באתר)**​  
   [instapage.com](https://instapage.com/en/template/api-documentation-page-template-compatible-with-wix#:~:text=API%20documentation%20page%20template%20compatible,requests%20per%20minute%2C%20per%20instance). המשמעות היא שאם חורגים ממספר זה, הקריאות הנוספות עלולות להידחות (Rate limit exceeded). בעת תכנון המערכת, ודאו שאתם מקבצים או מפזרים קריאות מסיביות כדי להישאר בגבולות אלו.
3. **הרשאות וגישה לנתונים:** רק משתמש בעל הרשאות מתאימות יכול לבצע פעולות כתיבה/עדכון/מחיקה. במקרה של שימוש במפתח API – אך ורק בעל האתר יכול להשתמש במפתח כדי לגשת ל-CMS​  
   [reddit.com](https://www.reddit.com/r/WixHelp/comments/1e99yb3/using_apis_to_manage_collections/#:~:text=https%3A%2F%2Fdev.wix.com%2Fdocs%2Frest%2Fbusiness,key%20created%20by%20the%20owner) (מפתחות של תורמים או אדמין-משנה לא יעבדו ישירות מול ה-API). במקרה של אפליקציית OAuth – ודאו שהאפליקציה הותקנה על-ידי בעל האתר או שיש לכם את ה-instanceId והאישורים המתאימים. בנוסף, ודאו שאוסף ה-CMS מוגדר בצורה שמאפשרת את הפעולה: אוסף לא **פורסם** (Unpublished) או עם הגבלות מחמירות יכול למנוע גישה. בדרך כלל, אוספים צריכים להיות בסטטוס "Live" (פורסמו מה-Sandbox) כדי שה-API ייגש אליהם. אם אתם רואים שגיאות Authorization (כגון 403 Forbidden או הודעת שגיאה מותאמת של Wix), בדקו את הגדרות ההרשאה של האוסף ב-Wix Content Manager, וודאו שהאפליקציה קיבלה את כל ההרשאות הדרושות (Scope של Data Collections וכדומה) בעת ההתקנה.
4. **מגבלות API נוספות:** כאשר מבצעים שאילתות (Query) דרך ה-API, יש גבול לכמות הפריטים המוחזרת כברירת מחדל (למשל 50 או 100 רשומות בעמוד, עם אפשרות להשתמש במנגנון paging לקבלת המשך התוצאות). עבור פעולות עדכון/הוספה, שימו לב להגבלות כמו גודל מקסימלי לבקשה (אם מעדכנים שדות טקסט גדולים במיוחד) או סוגי שדות מסוימים הדורשים שימוש ב-API יהודי – למשל, שדה **Reference** (ייחוס לפריט אחר) לא מתעדכן באמצעות אותו Endpoint של עדכון פריט רגיל, וישנו Endpoint נפרד להוספת ייחוסים (insert-reference וכד'). אם האוסף שלכם מכיל שדות כאלה, קראו בתיעוד Wix כיצד לטפל בהם.
5. **בדיקות ו-Workflow:** מומלץ מאוד לבדוק את קריאות ה-API תחילה באמצעות כלי כמו **Postman** או cURL, עם ה-Access Token/API Key שלכם, כדי לוודא שהשאילתות והעדכונים עובדים כמצופה לפני שמממשים אותם בקוד. כמו כן, עקבו אחר קודי התגובה (HTTP response codes) מכל קריאה: קוד 200/201 מעיד על הצלחה (201 Created במקרה של הוספה), 204 ללא תוכן למחיקה מוצלחת, 400 לסינטקס שגוי או נתונים חסרים, 401/403 לבעיות אימות/הרשאה, וכו'. טיפול נכון בשגיאות בלוגיקה שלכם יסייע באיתור מהיר של בעיות (כגון Token שפג תוקף או הרשאות חסרות).

לסיכום, Wix CMS REST API מאפשר שליטה מרחוק בתוכן האתר – הוספה, עדכון ומחיקה של רשומות – בתנאי שמגדירים נכון את האפליקציה/גישה ומקפידים על כללי האבטחה וההרשאות. עם ההכוונה הנכונה והדוגמאות שלעיל, מפתח יכול להבין את הלוגיקה הנדרשת וליישם אותה בהצלחה בצד השרת שלו לטובת אינטגרציה חלקה עם Wix.

**מקורות:** מדריך Wix Headless API ותיעוד REST הרשמי​

[dev.wix.com](https://dev.wix.com/docs/rest#:~:text=REST%20Docs%20,Stores%20product%20catalog%20and)

​

[dev.wix.com](https://dev.wix.com/docs/go-headless/coding/rest-api/admin/make-api-calls-with-an-api-key#:~:text=To%20make%20API%20call%20with,Key%2C%20use%20the%20following%20headers)

, תשובות מפתחים בקהילה (פורום Wix, Reddit, StackOverflow) לגבי OAuth ודרישות כותרות​

[stackoverflow.com](https://stackoverflow.com/questions/68065780/wix-api-how-to-get-access-token-for-using-rest-api-for-my-app#:~:text=In%20order%20to%20use%20the,your%20app%20in%20his%20site)

​

[reddit.com](https://www.reddit.com/r/WixHelp/comments/1e99yb3/using_apis_to_manage_collections/#:~:text=Solved,others)

, וכן דוגמאות קוד ממפתחים אחרים לשימוש ב-axios מול Wix Data API​

[forum.wixstudio.com](https://forum.wixstudio.com/t/wde0027-the-current-user-does-not-have-permissions-to-insert-on-the-collection/64028#:~:text=Data%20to%20be%20sent%20to,com%E2%80%9D%2C%20%E2%80%9CreferralCode%E2%80%9D%3A%20%E2%80%9Cnywh46x%E2%80%9D)

. כל אלו שולבו במדריך הנ"ל על מנת לספק הבנה מעמיקה וכלים פרקטיים להתממשקות עם Wix CMS.