סביבות פיתוח באינטרנט – עבודה 3 חלק ב'

# API

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Method Name | HTTP Method | Parameters | Returns | Explanation |
| 1 | Users/Login | POST | UserName, Password | Bool IsAdmin,  Bool IsActive | התחברות לא יכולה להתבצע בGET. הפונקציה האם המשתמש מנהל או לא והאם הוא פעיל או לא. |
| 2 | Products/ProductCategoriesById | GET | productId | CategoryId[] | מוצרים וקטגוריות מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 3 | Products/ListProducts | GET | - | returns all products: Products[] | מוצרים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 4 | Products/HotProducts | GET | numberOfProducts | returns hot products: Products[numberOfProducts] (מחזיר את רשימת המוצרים החמים בגודל המבוקש) | מוצרים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת (פרט לכמות המוצרים הרצוייה) או בקשה לעדכון מידע. |
| 5 | Users/Register | POST | UserName, Password, CountryId, ForgetPassQuastion, ForgetPassAns, CategoriesId |  | הוספת לקוח חדש תבוצע בצד השרת. בחרנו בPOST כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת. |
| 6 | Users/ForgetPassword | GET | UserName | PasswordQuastion | כיון שמדובר בפונקציה שצריכה להחזיר מידע השמור בצד השרת היא תתבצע בצד השרת. אין מידע לעדכן ולכן נשתמש בGET. הפונקציה תחזיר את השאלה לשחזור סיסמא של המשתמש. |
| 7 | Users/ForgetPassAnswer | POST | UserName, ForgetPassAns | Password | כיון שמדובר בפונקציה שצריכה להחזיר מידע השמור בצד השרת היא תתבצע בצד השרת. כיוון שזו פונקציה הדומה להתחברות, אותם התנאים חלים גם כאן ונשתמש בPOST. הפונקציה תחזיר את הסיסמא של המשתמש. |
| 8 | Products/LastestProducts | GET | - | returns all last month new products: Products[] | מוצרים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 9 | Products/LastestProducts | GET | numberOfProducts | returns all last month new products: Products[numberOfProducts] (מחזיר את רשימת המוצרים החדשים מהחודש האחרון בגודל המבוקש) | מוצרים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת (פרט לכמות המוצרים הרצוייה) או בקשה לעדכון מידע |
| 10 | Users/CategoriesList | GET | - | return all categories: Categories[] | קטגוריות מקושרות לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 11 | Products/RecomandedProducts | GET | userName, numberOfProducts | returns list of recommended products for the user: Products[] | מוצרים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 12 | Products/ProductById | GET | productId | return specific product | מוצרים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת (פרט למזהה מוצר) או בקשה לעדכון מידע. |
| 13 | Orders/AddToBasket | POST | UserId, ProductId, Quantity |  | הוספת מוצר חדש לעגלה תבוצע בצד השרת. בחרנו בPOST כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת (מוסיפה את המוצר לעגלת הקניות הפעילה של המשתמש). |
| 14 | Orders/ActiveBaket | GET | userId | returns the activ basket of the user includes list of products and their quantity : Basket | עגלת הקניות מקושרת לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 15 | Orders/RemoveFromBasket | PUT | UserId, ProductId |  | הסרת מוצר מעגלה תבוצע בצד השרת. בחרנו בPUT כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת (מסירה את המוצר מעגלת הקניות הפעילה של המשתמש). |
| 16 | Orders/PreviousOrders | GET | userId | returns all the previous orders of the user includes list of products and their quantity : Order [] | הזמנות קודמות מקושרת לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 17 | Orders/PreviousOrderById | GET | userId, orderId | return specific order of the user includes list of products and their quantity : Order | הזמנות מקושרות לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת (פרט למזהה משתמש והזמנה) או בקשה לעדכון מידע. |
| 18 | Orders/buyBasket | POST | UserId, BasketId, ShippingDate | Order | הוספת הזמנה חדשה למשתמש תבוצע בצד השרת. בחרנו בPOST כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת (ממירה את עגלת הקניות הפעילה של המשתמש להזמנה סגורה). הפונקציה מחזירה את מספר ההזמנה ופרטיה במידה והצליחה ומספר הזמנה 0 ופרטים חסרים במידה ולא. |
| 19 | Admins/AdminListUsers | POST | AdminUserName, AdminPassword | returns all users: Users[] | משתמשים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 20 | Admins/AdminListOrders | POST | AdminUserName, AdminPassword | returns all baskets: Basket[] | הזמנות מקושרות לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 21 | Admins/AdminListProducts | POST | AdminUserName, AdminPassword | returns all products: Products[] | מוצרים מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |
| 22 | Admins/AdminAddProduct | POST | AdminUserName, AdminPassword, Product + CategoriesId[] | - | הוספת מוצר חדש תבוצע בצד השרת. בחרנו בPOST כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת. |
| 23 | Admins/AdminUpdateProduct | PUT | AdminUserName, AdminPassword, Product+ CategoriesId[] | - | עדכון מוצר יבוצע בצד השרת. בחרנו בPUT כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת. |
| 24 | Admins/AdminAddUser | POST | AdminUserName, AdminPassword, User+ CategoriesId[] | - | הוספת משתמש חדש תבוצע בצד השרת. בחרנו בPOST כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת. |
| 25 | Admins/AdminUpdateUser | PUT | AdminUserName, AdminPassword, User+ CategoriesId[] | - | עדכון פרטי משתמש יבוצע בצד השרת. בחרנו בPUT כיוון שהפונקציה מקבלת מידע ואמורה לעדכן אותו בצד השרת. |
| 26 | Products/CategoryProductsById | GET | categoryId | Products[]+CategoryId | מוצרים וקטגוריות מקושרים לצד שרת ולא לצד לקוח. נשתמש בGET כי אין מידע שמועבר לשרת או בקשה לעדכון מידע. |

\*הוספנו 26 בכדי לסנן מוצרים לפי קטגוריות.

**כל אחת מהפונקציות מחזירה JSON במבנה הבא:**  
 status- OK במידה והצליח, ERROR במידה ונכשל.  
 message- הודעה המסבירה את ההצלחה או הכשלון  
resObject – אובייקט המכיל את הResponse לפי הטבלה.

Catrgory: { Id, Name, Description }

Product : { Id, Name, Weight, Color, Price, Description, QuantityInStock, Active, SoldInPastWeek, ImgUrl, CreateDate }

User: { Id, UserName, Password, Country, PasswordQuastion, PasswordAnswer, LastLogin, Active, IsAdmin }

Basket: { Id, Submitted, ShippingDate, UserId }

Order: { Id , ProductId , Quantity , ProductName , ProductWeight , ProductColor , ProductPrice , Description , ProductQuantityInStock , ProductIsActive }

Country: { ID, Name }

* מחיקה של משתמש או מוצר זהה ל Active=false.
* בכל פונקצייה של הוספת מוצר חדש או משתמש חדש שדה הId אינו רלוונטי כי המבנה נתונים נותן לכל רשומה חדשה Id חדש בסדר עולה.
* מאזין לפורט – 4000.
* חילקנו את הפונקציות ל5 קטגוריות:
  + Users – פונקציות הקשורות לניהול משתמשים.
  + Products – פונקציות הקשורות לניהול מוצרים.
  + Orders – פנקציות הקשורות לניהול הזמנות ועגלות.
  + Admins – פונקציות הקשורות לממשק ניהול האתר למשתמשים שהם מנהלים.
  + Countries- פונקציות הקשורות לניהול מדינות (כרגע רק פונקציה אחת אשר קוראת רשימת מדינות מקובץ XML נתון).
* בDatabase 4 טבלאות ראשיות:
  + Products : { Id, Name, Weight, Color, Price, Description, QuantityInStock, Active, SoldInPastWeek, ImgUrl, CreateDate }
  + Users: { Id, UserName, Password, Country, PasswordQuastion, PasswordAnswer, LastLogin, Active, IsAdmin }
  + Categories : { Id, Name, Description }
  + Basket: { Id, Submitted, ShippingDate, UserId }
* כמו כן 3 טבלאות מקשרות של קשרי רבים לרבים:
  + ProductCategories : { ProductId, CategoryId } - עבור כל מוצר לאילו קטגוריות שייך
  + UserCategories : { UserId, CategoryId } עבור כל משתמש אילו קטגוריות מעניינות אותו-
  + BasketProducts : { BasketId, ProductId, Quantity } - עבור כל עגלת קניות אילו מוצרים נמצאים בה ומה הכמות של כל מוצר בעגלה

משתמשים:

