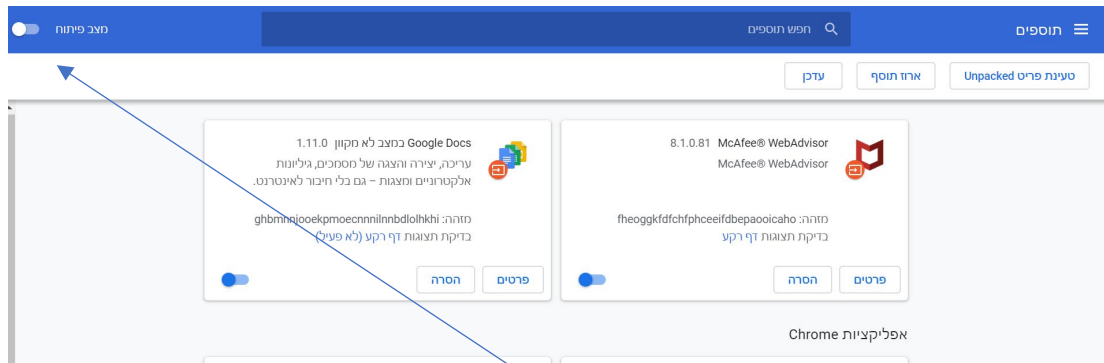


מיני פרויקט באבטחת מחשבים מגשים: הדר הררי, כפיר רימברג

כיצד להתקין את הפרויקט?

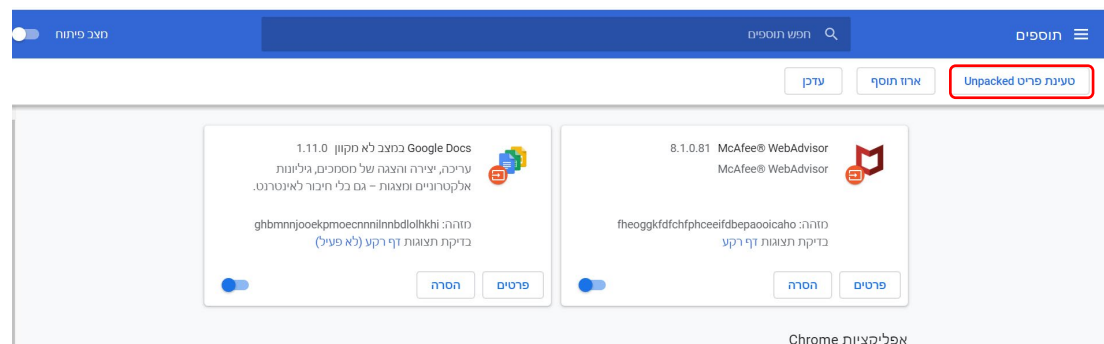
הפרויקט מחולק ל-2 צדדים מרכזיים: צד לקוח וצד שרת. על כן, בקבצי הגשה שהגשנו ישנם 2 תיקיות מרכזיות: Extension-Server. שרת, Extension – לקוח. תחילה נציג כיצד יש להתקין את התוסף ל-Chrome:

ב-Chrome, יש להיכנס ללינק - <chrome://extensions/>:



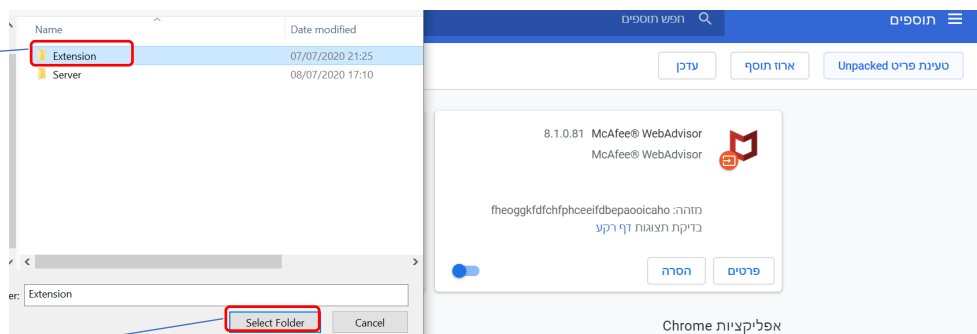
יש לוודא מצד פיתוח במצב מאופשר

לאחר מכן יש להוסיף את תיקיית Extension מקבצי ההגשה בצורה הבאה:

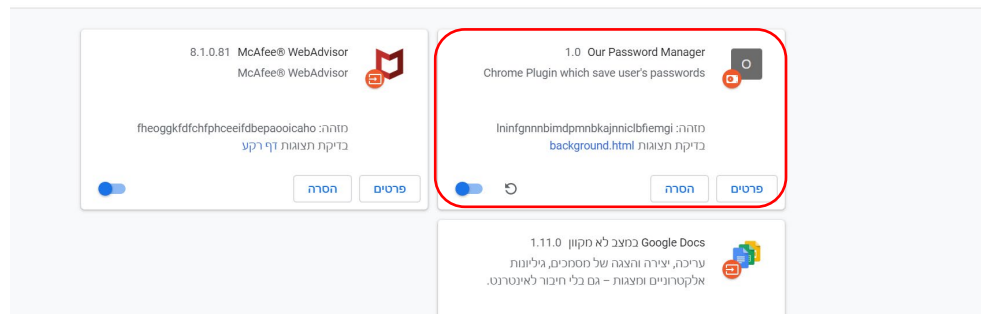


יפתח חלון ויש לבצע את הפעולות הבאות:

סימון
התיקייה
Extension
שנמצאת
בתיקייה
הגשה

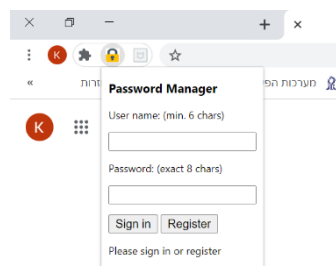
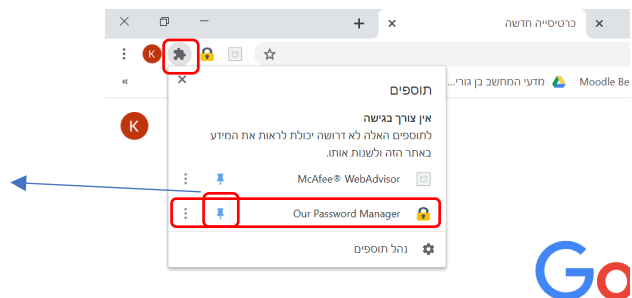


בחירת התיקייה
הרצויה



מצויין! כעת, התוסף זמין לשירותך:

ניתן ללחוץ על
ה'סיכה' אם ברצונך
שהתוסף יופיע בבר
באופן קבוע



צד הלקוח מוכן:

כעת נעבור לצד השרת:

יש להיכנס ל-cmd של windows ולהגיע לתיקיית ההגשה לאחר הורדתה למחשבך האישי. כמו שצויין לעיל, בתיקיית ההגשה 2 תיקיות, במקרה זה אנו ניכנס לתיקיית Server דרך ה-cmd ונבצע את ההתקנות הבאות (גם הן דרך ה-cmd):

יש להוריד Node.js ו-npm למחשב – מצ"ב לינק להורדה גרסה עדכנית ומדריך –

<https://nodejs.org/en/download/> - הורדה

<https://phoenixnap.com/kb/install-node-js-npm-on-windows> - מדריך

Next install Mongoose from the command line using `npm`:

```
$ npm install mongoose
```

crypto-js build passing

JavaScript library of crypto standards.

Node.js (Install)

Requirements:

- Node.js
- npm (Node.js package manager)

```
npm install crypto-js
```

Installation

```
npm install socket.io
```

כעת, לאחר שביצענו את כלל ההתקנות הנדרשות, אנחנו מוכנים להריץ! נעשה זאת בצורה הבאה:

```
Microsoft Windows [Version 10.0.18362.900]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\kfirr>cd Desktop
C:\Users\kfirr\Desktop>cd proj12
C:\Users\kfirr\Desktop\proj12>cd HADARnKFIR
C:\Users\kfirr\Desktop\proj12\HADARnKFIR>cd Server
C:\Users\kfirr\Desktop\proj12\HADARnKFIR\Server>node server.js
```

```
C:\Users\kfirr\Desktop\mini_project_computer_security\READYPRO\HADARnKFIR\Server>node server.js
starting server...
server is running!
```

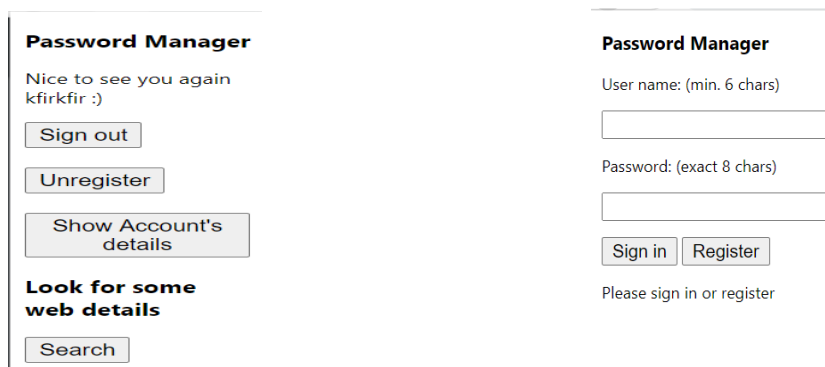
אם מופיעה ההודעה הבאה, השרת רץ, ומוכן להמשיך עבודתך!
עד כאן הפרטים הטכניים, בדף הבא נתחיל בהצגת פונקציונאליות ותוכן הפרויקט.

מהות הפרויקט

הפרויקט מממש מנהל סיסמאות ללקוחות דרך תוסף ל-Chrome, בעזרת צד שרת וצד לקוח (כפי שהוצג לעיל), בצורה בטוחה ומוצפנת מבלי שהשרת 'מודע' לתוכן המידע שנשמר כיוון שהמידע נשמר בצורה מוצפנת על פי הנחיות הפרויקט שניתנו (יפורט בהמשך). בנוסף, התוסף יכול לתמוך במספר לקוחות וניתן להורידו מכל מכשיר.

פונקציונאליות משתמש

התוסף תומך במגוון אפשרויות ללקוחות:

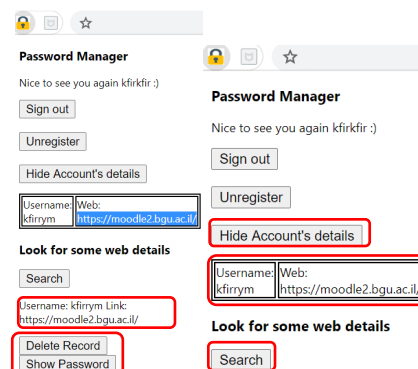


The image shows two side-by-side screenshots of a web application titled "Password Manager".

The left screenshot shows a logged-in user interface with the text "Nice to see you again kfirkfir :)". It contains three buttons: "Sign out", "Unregister", and "Show Account's details". Below these is a section titled "Look for some web details" with a "Search" button.

The right screenshot shows a login/register interface. It has input fields for "User name: (min. 6 chars)" and "Password: (exact 8 chars)". Below these are "Sign in" and "Register" buttons, followed by the text "Please sign in or register".

1. Register – הרשמה לשירות מנהל הסיסמאות. מבוצעת בדיקה אם שם המשתמש אינו תפוס ונתוני ההרשמה תקינים (שם משתמש וסיסמא). אם הבדיקות צלחו, המשתמש נוסף למאגר הלקוחות, אם לא צלח, הלקוח יקבל הודעה מתאימה ויפעל על פיה.
2. Sign in – לאחר הרשמה לשירות, בכדי להנות ממנהל הסיסמאות על המשתמש להתחבר עם כניסתו לדפדפן. אם הנתונים שהוקלדו אכן נכונים וקיים משתמש במאגר הלקוחות, הלקוח יקבל הודעת התחברות וכן השרת מחזיק גישה לקובץ נתונים של המשתמש על מנת לספק שירות לפעולות עתידיות של הלקוח.
3. Sign out – אפשרות לצאת מהשירות זמנית בכל שלב בעת גלישה בדפדפן. בעת לחיצה על כפתור זה, השרת מעדכן את קובץ הנתונים של הלקוח שנמצא במאגר ומוחק אותו משירות האחסון המקומי. לאחר ביצוע sign out דף התוסף חוזר לתצוגתו המקורית.
4. Unregister – אי הרשמה לשירות, כלומר מחיקת הלקוח ממאגר הלקוחות וכן מחיקת כלל הנתונים של הלקוח שנשמרו אי פעם.
5. Show Account's details – טבלה הנפתחת בתוסף ובה מוצגים כלל הנתונים שנשמרו עבור הלקוח. לדוגמה: לקוח מבצע כניסה אתר facebook, התוסף יבצע שמירה אוטומטית של הנתונים (שם משתמש וסיסמא) ובטבלה שלו יופיעו האתר אליו נרשמו הנתונים וכן שם המשתמש לאתר זה (הסיסמא לא תופיע בטבלה משיקולי אבטחה).



The image shows two side-by-side screenshots of the "Look for some web details" section of the Password Manager interface.

The left screenshot shows a search bar with the text "Search". Below it, there are two input fields: "Username: kfirkfir" and "Web: https://moodle2.bgu.ac.il/". Below these fields are two buttons: "Delete Record" and "Show Password".

The right screenshot shows the same search bar and input fields. Below the input fields, there are two buttons: "Delete Record" and "Show Password".

6. Look for some web details -> Search – פיצ'ר בו למשתמש יש אפשרות לבצע פעולות בעזרת הקלדה של האתר הספציפי עליו הוא רוצה לבצע את הפעולות (בעת לחיצה

על כפתור Search על הלקוח להקליד את URL של האתר הספציפי עליו נדרש לבצע פעולות. אם אין מידע שמור על אתר זה, הלקוח מקבל הודעה מתאימה, אחרת נפתח ללקוח בדף התוסף אופציות נוספות לשימוש):

- a. Delete Record – המשתמש יכול למחוק את הנתונים שנשמרו עבור אתר מסויים בתוסף. אם וכאשר המחיקה צלחה, נתוני המשתמש מתעדכנים בשירות האחסון המקומי וכן במאגר הנתונים של הלקוח.
- b. Show Password – כפתור זה נועד ללקוח המעוניין לראות סיסמא לאתר מסויים. מנגנון זה מאובטח ב-2 בקשות אישור מהמשתמש כולל הקלדת סיסמא לתוסף על מנת להיות בטוחים כי הלקוח מודע לכך שהוא חושף את סיסמתו בדפדפן ואכן מעוניין לראות את הסיסמאות.

לאורך ביצוע כלל הפעולות ניתן לראות 'הודעות' מהשרת אודות הפעולות ואופן טיפול השרת בבקשות המתאימות (מצ"ב דוגמאות לחלק מהפיצ'רים):

```
starting server
server is on
(node:22368) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discovery and Monitoring engine, pass option { useUnifiedTopology: true } to MongoClient constructor.
kfirkfir register request
succed new user content
kfirkfir registered successfully
kfirkfir login request
user kfirkfir exists in data base
password correct
```

צד שרת

על מנת לממש את צד השרת בחרנו להשתמש ב-mongoDB על מנת לאחסן את קהל הלקוחות וכן את נתוני הלקוחות (לכל לקוח רשומה משלו כמובן). בנוסף, השתמשנו ב-Google Local Storage על מנת לשמור באופן זמני את הנתונים של כל לקוח כאשר הוא נרשם ומתחבר לשירות. כאשר הלקוח מתחבר לשירות, השרת מייבא את רשומת הנתונים שלו אל שירות האחסון המקומי וכלל הפעולות אשר מתבצעות ע"י הלקוח פועלות על האחסון המקומי של Google ורק כאשר הלקוח מבצע יציאה מהתוסף, השרת מייבא את הנתונים האחרונים שעודכנו באחסון המקומי ושומר אותם במאגר הנתונים 'המרוחק' (MongoDB) ומנקה את האחסון המקומי (איננו מעוניינים שמידע זה יישמר לאורך זמן באחסון המקומי – שיקולי אבטחה). בנוסף, השתמשנו בפלטפורמה Socket.io המהירה בין צד השרת ללקוח בצורה מיטבית לאחר שחקרנו והעמקנו בנושא זה (מעייין 'צינור' המאפשר תקשורת דו-כיוונית וטיפול בקריאות הן מצד הלקוח והן מצד השרת) –

"Socket.IO enables real-time, bidirectional and event-based communication. It works on every platform, browser or device, focusing equally on reliability and speed"
(<https://socket.io/>)

הרצת השרת מתבצעת על PORT 3000 (מצ"ב תמונות לחיבור התקשורת מצד הלקוח ומצד השרת):

```
var io = require('socket.io').listen(3000);
var socket = io('http://localhost:3000');
```

```
"permissions": [
  "storage",
  "activeTab",
  "tabs",
  "<call_urls>",
  "background",
  "http://localhost:3000/*",
  "webRequest",
  "webRequestBlocking"
]
```

הרשאות גישה בתוסף:

אבטחת הנתונים והצפנות

כאשר מתבצעת כניסה לתוסף ע"י לקוח מתבצעת גזירה של ה-MasterPassword ל-3:

1. PasswordForServer
2. EncryptionKey
3. AuthenticationKey

החלק הראשון של הסיסמא, אנו משרשרים לו את הספרה "1" ומבצעים הצפנה ע"י פונקציית HmacSHA256 וזוהי הסיסמא שנשלחת לשרת בנוסף לשם משתמש. בשרת מתבצעת המלחה של 16 מספרים/אותיות רנדומליות ושרשור הסיסמא שהתקבלה ע"י השרת לערך salted שהמלחנו. לאחר מכן מתבצעת פונקציית Hash על המפתח שכולל את הסיסמא לשרת + ההמלחה ונשמרת רשומה במאגר הלקוחות בצורה הבאה:

<username, salted, Hashed(salted, PasswordForServer)>

בעזרת שמירה נתונים אלו, בכל פעם שנשלחת בקשת התחברות של לקוח מסוים, מתבצעת הגזירה שהוזכרה לעיל וכך נבחנת בקשת ההתחברות אל מול הנתונים הקיימים על הלקוח. במידה ו Hashed השמור ללקוח הספציפי אינו תואם ל Hashed החדש שנוצר בעת בקשת ההתחברות, התוסף (דרך השרת) מעדכן את הלקוח כי הנתונים שהוקלדו אינם נכונים.

החלק השני של הסיסמא, אנו משרשרים לו את הספרה "2" ומבצעים הצפנה ע"י פונקציית HmacSHA256 וזוהי סיסמא המייצגת את מפתח ההצפנה. בעזרת הצפנה זו אנו נעזרים על מנת להצפין / לפענח נתונים של לקוח. כהוזכר לעיל, בכל התחברות של לקוח לתוסף, השרת מייבא את נתוני הלקוח שנשמרו עד כה (במידה וקיימים), מאחסן אותם זמנית באחסון המקומי ושולח בקשה לקליינט לבצע עליהם פיענוח (decrypt) על מנת לאפשר ללקוח לעדכן נתונים במידת הנדרש. כאשר מסיים הלקוח לבצע את הפעולות בתוסף, מבוצע שוב הצפנה (encrypt) לנתונים שלו, נשלחת בקשה מהקליינט לשרת לשמור את הנתונים החדשים (לאחר שכבר הוצפנו) והנתונים נשמרים במאגר הלקוחות 'המרוחק' ונמחקים מהאחסון המקומי של Google.

החלק השלישי של הסיסמא, אנו משרשרים לו את הספרה "3" ומבצעים הצפנה ע"י פונקציית HmacSHA256 וזוהי סיסמא לבדיקת אימות (authentication). כלומר, סיסמא זו נועדה על מנת לאמת כי הנתונים של הלקוח שנמצאים כרגע במאגר הלקוחות 'המרוחק' הינם באמת הנתונים שאותם שמר הלקוח. בכל עידכון ושמירה של מידע נוסף ברשומת הנתונים של הלקוח אני משרשרים את ה MACkey בסוף המידע המוצפן, וכך בכל פעם שהלוקח מתחבר ומתבצע בקשה מהשרת לקבל את הקובץ סיסמאות של הלקוח מה DB מתבצעת בדיקה אם ה MACkey שנשמר אצל הלקוח אכן תואם ל MACkey ששורשר לסוף המידע שהלקוח הצפין בכניסתו הקודמת, ובכך או מוודאים כי לא היה שינוי תוכן המשתמש ע"י צד עוין / חשיפת פרטי המשתמש. כל שינוי של לקוח, כלומר מחיקה או הוספה של נתונים חדשים, משנה את ה MACKey (שכן הוא מתבצע על כל המידע של הלקוח).