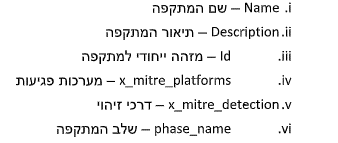
**פרויקט רפאל**

בשביל להתקבל ל"מגשימי נילוס" התבקשתי לכתוב פרויקט שמכיל react וגם node js .

**הכנסת הקבצים לתוך :database**

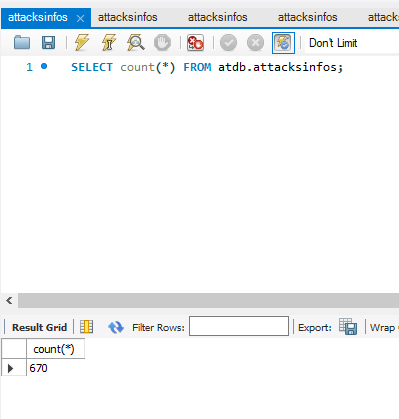
משימתי הראשונה הייתה לכתוב סקריפט שיכנס את כל ה670 מתקפות שהם היו קבצי json database, אני בחרתי להשתמש בdatabase של mySql workbench מכיוון שהוא נוח לשימוש בעל אפליקציה נוחה להבנה של המשתמש וקל להכנסת מידע והוצאתה.

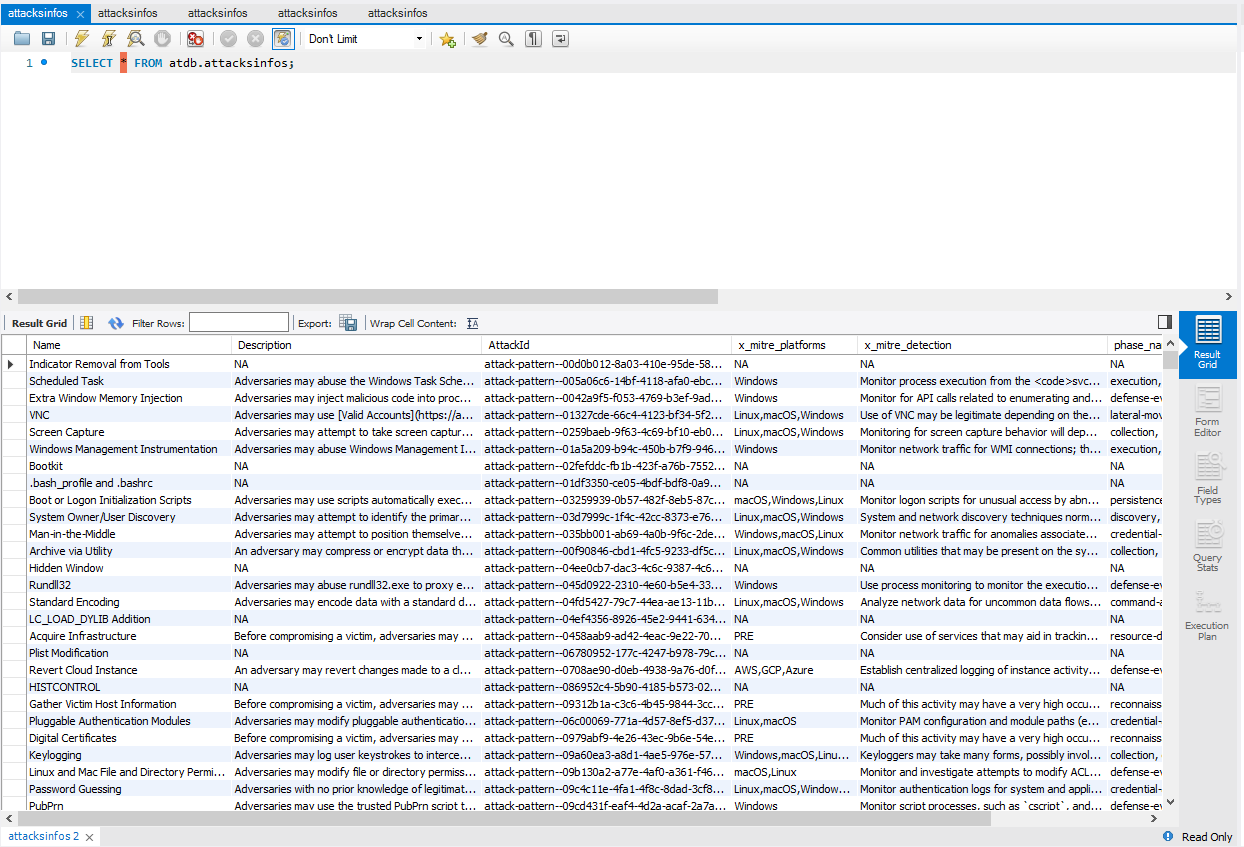
כפי שאפשר ראות הסקריפט נמצא בתוך תיקיית server ונקרא index.js דבר ראשון הייתי צריך לעשות קוד שעובר על כל 670 קבצים בתיקייה לפי הסדר (מלמעלה למטה) אפשר לראות את זה בשורה 22 ושורה 23 אחר כך הייתי צריך לפתוח את אותו קובץ לפי הpath שלו ולעבור על חמישה פרמטרים עיקריים שהייתי צריך להכניס לתוך הdatabase שלי והם:



כפי שאפשר לראות בקוד בין השורות 28-32 אני מאתר כל אחד מן הפרמטרים ואם אין אחד מהם אני שם במקומו את מחרוזת NA אחרי זה התוכנית מבצעת חיבור לdatabase аכפי שאפשר לראות בין השורות 4-9 ושולח את המידע אחד אחד לdatabase שלי.

אחרי זמן קצר שהתוכנית פעלת היא מסיימת את פעולתה ואפשר להיכנס לdatabase ו לראות שכל הקבצים נמצאים שם לפי התמונות מסך אללו:





כפי שאפשר לראות כל ה670 קבצים נמצאים בתוך ה MySQL workbench

הספריות הצד שלישי שהשתמשתי בהן בשביל הסקריפט הזה הם:

**MySql** – השתמשתי בספריה זו בשביל לגשת למוסד הנתונים שלי עם הפקודות שלו ושהוא יכיר אותם והחלק העיקרי בשביל להכניס אליו מידע שאשתמש בו גם לכל שער השלבים

**Fs** – השתמשתי בספרייה זו בשביל לקחת את כל שמות הקבצים הנמצאים בתוך תיקיה ולפתוח אותם ואז ולקחת מן הקבצים את המידע הרצוי עלינו.

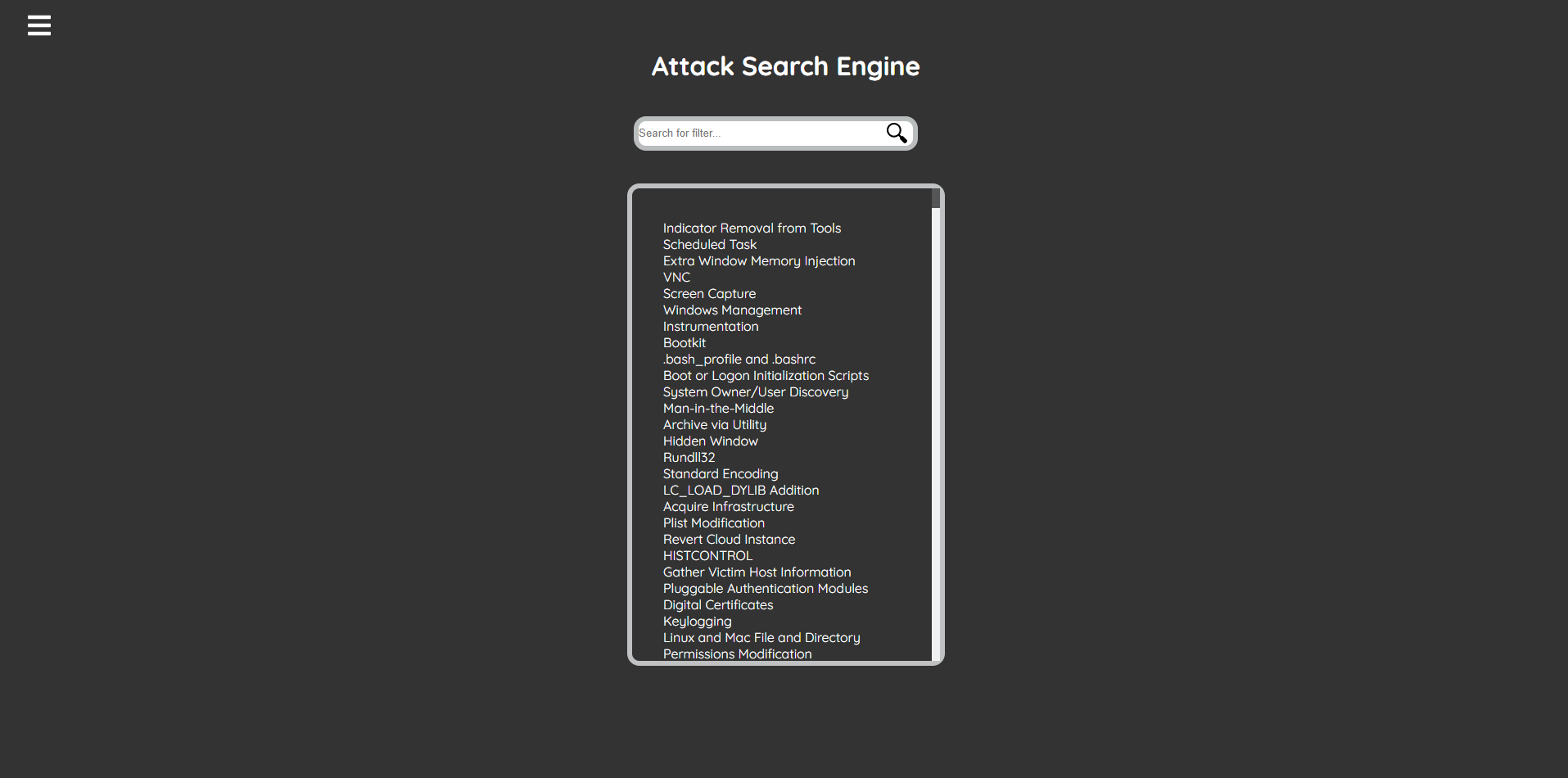
אחרי שעשיתי את כל הפעלות האלה ויש לי מוסד נתונים מאורגן ושלם אפשר להמשיך לחלק א.

**שלב 1:**

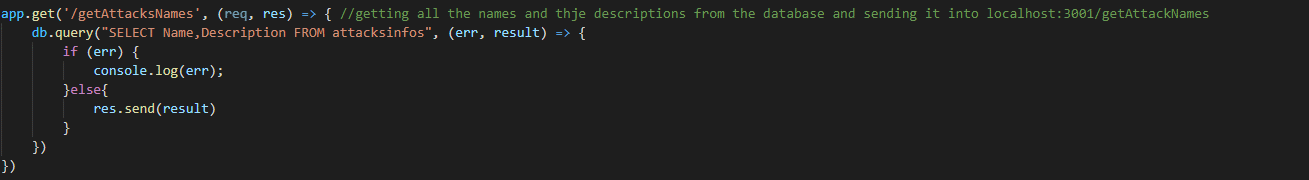
בשלב זה התבקשתי לכתוב אתר בעל צד שרת וצד לקוח שהצד לוקח יכיל גריד של כל המתקפות והמשתמש יכול לגלול אותם ולחפש אותם לפי הdescription שלהם.

אני בחרתי לעשות את החלק הזה לפי תיבת טקסט פשוטה שהמשתמש יכול להגליד שם כל סינון שירצה ומתקפות יראו התאם לסינון במלבן מתחת לתיבת טקסט המשתמש לא צריך ללחוץ שום כפתור בשביל שהסינון יתבצע כל משהוא צריך לעשות זה להגיד וזהו שאני רואה את זה פלוס מכיוון שזה נוח למשתמש ונוח להבנה שלו.

כאשר המשתמש פותח את האתר המנוע חיפוש קופץ לו ישר והוא נראה כך:

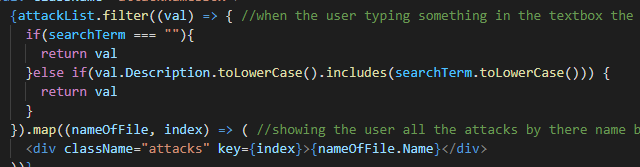


עכשיו בואו נכנס יותר פנימה לחלק הזה וניגש לbackend , הbackend נמצא בתיקיית הserver וככה נקרא backend.js הפונקציה היחידה שאנו משתמשים בה בשביל החלק הזה היא:



פונקציה זו פשוטה למידי תאמת אני משתמש פה בספריה שנקראת express ואני אפרט עלייה בהמשך ההסבר, הפונקציה יוצרת tab בתוך localhost:3001 שהוא שייך לbackend אפשר לראות אני מבצע שאילתה לתוך המסד נתונים ומבקש את כל השמות ואת התיאורים של המתקפות וזאת בשביל להראות למשתמש את השמות שלהם ולסנן לפי התיאור, אחרי שקיבלנו תשובה מין המסד נתונים התוכנית שולחת את המידע לlocalhost:3001/getAttacksName אחרי שהמידע נשלח כאן מסתיימת העבודה של הצד שרת, אחרי זה אני לוקח את כל המידע מאותו localhost:3001/getAttacksName בעזרת ספריית axios שאפרט עלייה בהמשך ושומר אותו בתוך useState שנקראה attackList שהוא בעצם רשימה שמכיל שמכילה את השמות ואת התיאורים של כל המתקפות.

פעולה זה מתבצעת רק פעם אחת והיא בתחילת התוכנית וזאת בשביל לא להעמיס על האתר ולא לתת למשתמש לעשות זאת ולהראות לו ישר פעולה זאת מתבצעת פעם אחת בעזרת useEffect .

עכשיו הגיעה החלק המעניין איך את מסנן ואת אני מדפיס למשתמש אז ישר את הקוד הזה: 

קוד זה לוקח את משהמשתמש מקליד התוך תיבת בטקס ומסנן את כל המתקפות ישר לפי זה אפשר לראות את הפעולה פיטר שבשורה הראשונה אחר כך הפועלת map מדפיסה את כל המתקפות אחרי הסינון אפשר לראות את זה בשורה האחרונה.

ספריות צד שלישי שאני משתמש בהן:

**Express** – ספרייה זאת מאפשרת לתוכנית ליצור tab חדש שיהיה שייך רק לפונקציה הזאת לשלוח לשם את המידע בשביל שהצד לוקח שלנו יכול לקחת אותו ולשמור אותו בהמשך זאת ספרייה נהדרת וקלה וזה למה השתמשי דווקא בה.

**Axios** – ספרייה זאת מאפשרת לקחת את כל המידע מן אתר שאנו בוחרים, בשלב זה השתמשתי בספרייה זו בשביל לקחת את כל המידע מlocalhost:3001/getAttacksNames ולשמור אותם לתוך משתנה ואז לעבוד בהמשך עם אותו משתנה.

**MySQL** – בניגוד לשלב הקודם כאן את צריך את הספרייה הזאת בשביל לא להכניס לתוך המסד הנתונים אלה בשביל לשלוף ממנו את המידע או במקרה שלנו לשלוף את השם ואת התיאור של התקפות.

**Curl** – ספרייה זו עוזרת מכיוון שבלי אותו ספרייה לא היינו יכולים לקשר את הצד שרת וצד לוקח בגלל שצד לקוח לא מזהה את הצד שרת ואומר שהוא לא בטוח ועם ספרייה זו הכל מתוקן והם מזהים אחד את השני זה למה השתמשי בספרייה זו.

אני בחרתי לעשות את הקוד של שלב 1 דווקא ככה בגלל שלדעתי זו היא קלה למשתמש וממש נוחה למשתמש ולמתכנת של התוכנית הזו התוכנית, מובנת ומסודרת וזאת בעזרת הספריות הצד שלישי שלה ושקלות להבנה למימוש.

**שלב 2:**

בשלב זה התבקשתי לכתוב צא'ט בוט ושיהיה לו פונקציה של לחפש בתוך המסד הנתונים בשביל מתקפה שהוא בוחר וגם לעשות שאילתות למאגר חיצוני כגון virusTotal .

אני כתבתי את השלב הזה בתוך tab חדש וקוראים לו Chatbot, לChatbot יש פונקציה ממש נחמדה שהיא מראה אם הוא פעיל או לא כלומר אם הצד שרת פועל או כבוי ביחס לסטטוס של השרת כתוב למעלה באדום Offline או בירוק Online תלוי בתוצאה.

דולק:



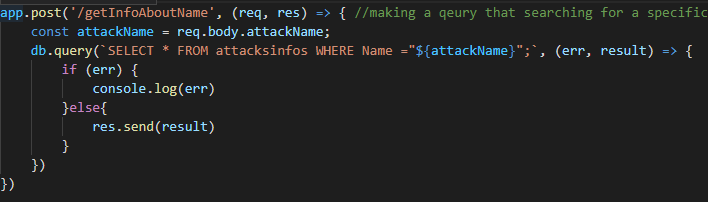
כבוי:



הפונקציות שאפשר להשתמש בהן בדף זה:

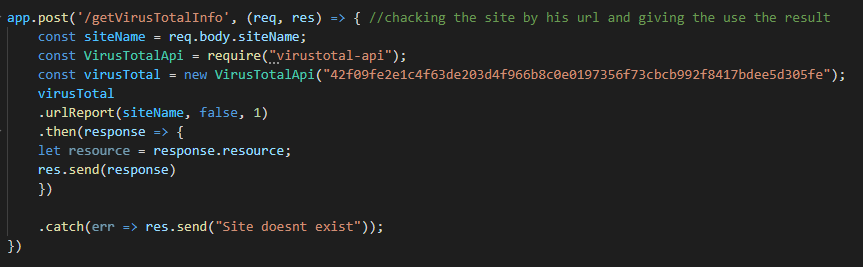
**פונקציית הget info about <attackName> attack**: פונקציה זו מקבלת מן המשתמש את שם המתקפה שהוא רוצה לבדוק ומחזירה לו את כל הפרטים על המתקפה.

הפונקציה הזו נראת בצד שרת כך:

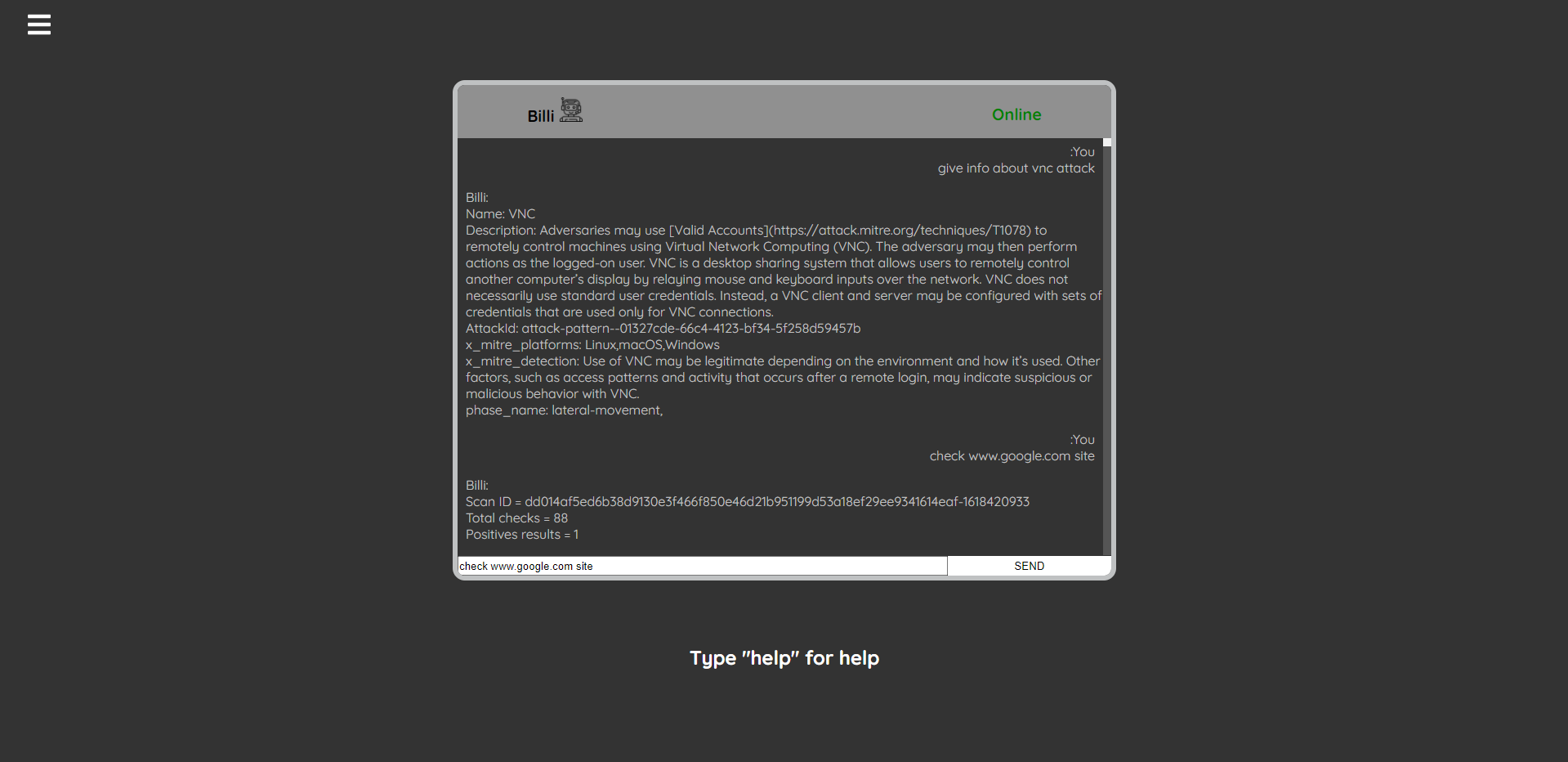


אפשר לראות שהפונקציה מבצעת שאילתות למסד נתונים שלנו ומבקשת ממנו את המתקפה עם שם בול כמו שהמשתמש שלנו רוצה במקרה והמתקפה לא קיימת התוכנית מדפיסה למשתמש שמתקפה זו לא קיימת. אחרי זה הצד שרת שולח ללוקח את המידע בעזרת axios שוב ושומר אותו לתוך משתנה. ושולח למשתנה את המידע בצורה מסודרת וקלה לעיין.

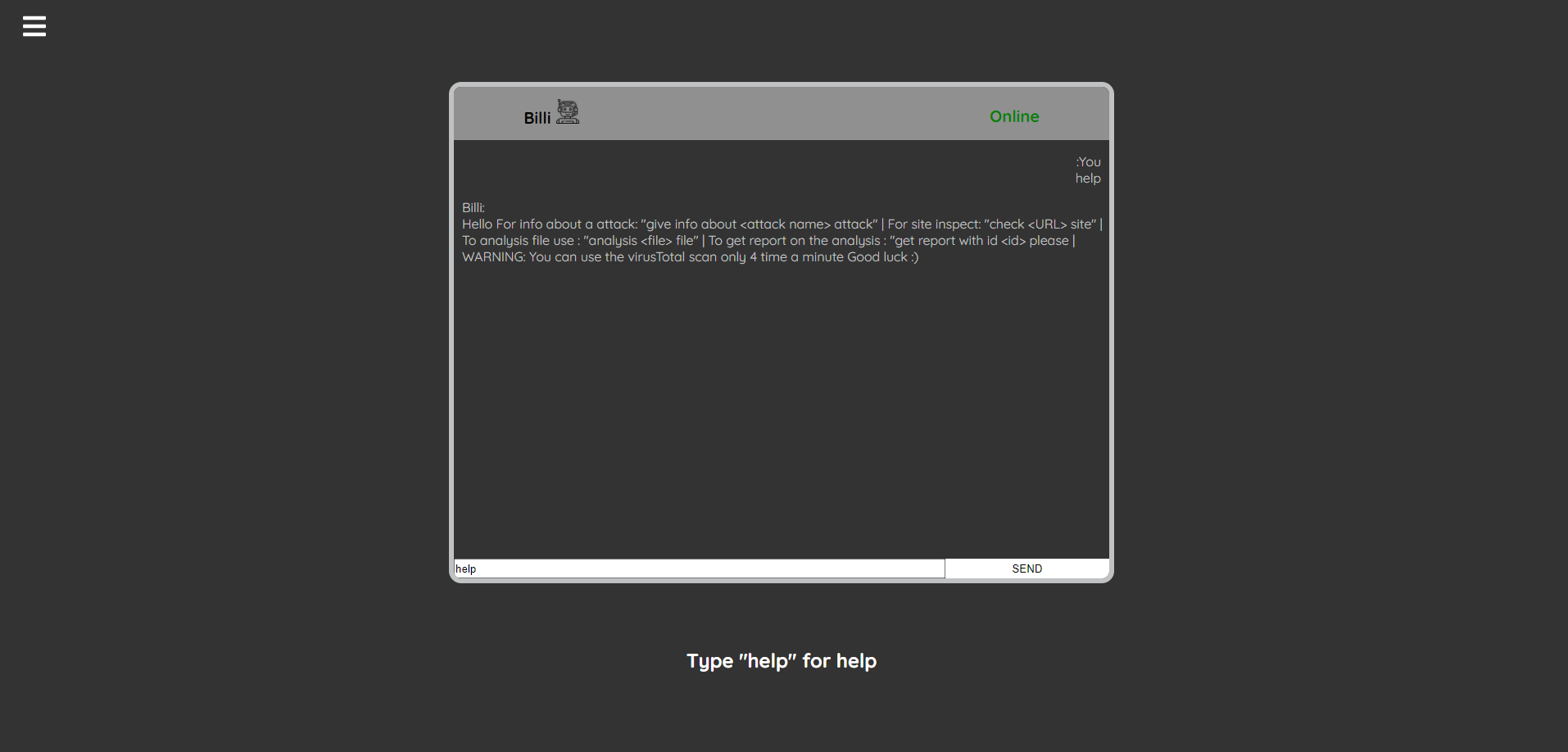
**פונקציית הcheck <url> site**: בהבדל לפעולות הקודמות פונקציה זו לא משתמשת במסד נתונים שלנו אלה משתמשת ב api אחר שאני בחרתי שיהיה virusTotal , virusTotal אתר שבודק וירוסים על אתרים או על קבצים משונים אני בחרתי לעשות בדיקה על אתר, הפונקציה בצד שרת נראת כך:



אפשר לראות פה API KEY שאפשר לקבל אותו אחרי יצרית משתמש באתר המקורי של virusTotal אפשר לראות פה שאני שולח את האתר לבדיקה ושולח tab getVirursTotalInfo את המידע שנשלח לי מין האתר וזאת בשביל שהצד לקוח יוכל לקחת אותו להציג למשתמש אם ישנה בעיה אני שולח למשתמש שאין כזה אתר. דוגמא לפעולה זה וגם לקודמת:

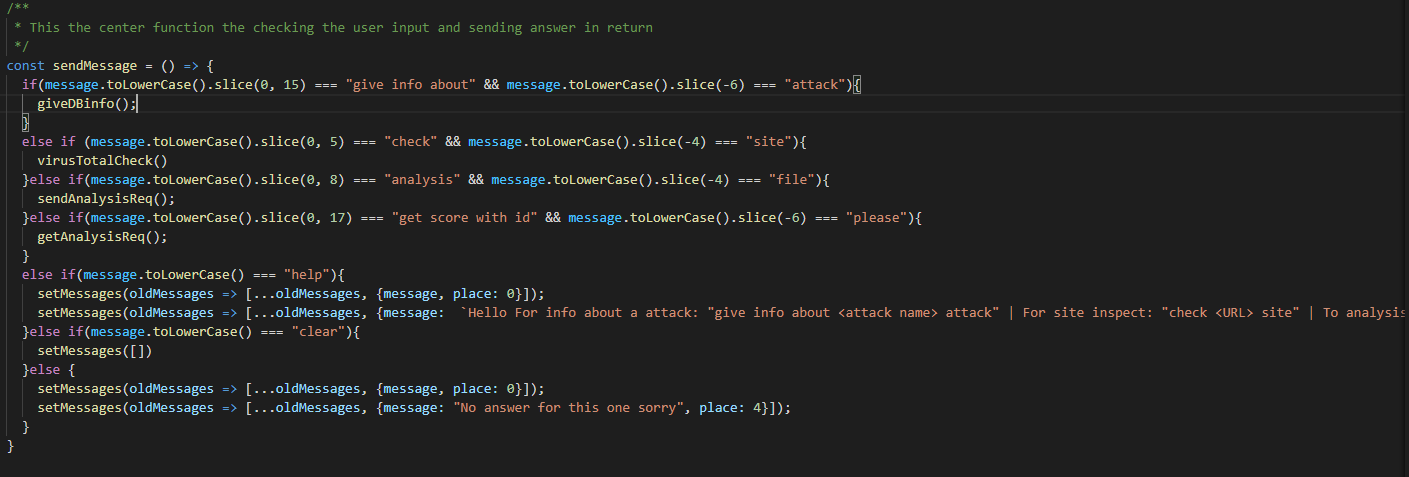


**פונקציית הhelp**: פונקציה זו מציגה למשתמש את כל הפקודות האפשרויות בדף זה. דוגמא:



**פונקציית הclear**: פונקציה זו מוחקת את הכל מהChatbot.

אופן עבודת הchatbot היא שהמשתמש שולח פקודה כלשהיא פקודה זה עוברת לפונקציה sendMessage ושפועלת ישר אחרי שהכפתור נלחץ הפונקציה נראת כך:



הפונקציה בודקת את מה שהמשתמש שלח ונכנסת לif המתאים לה הפונקציה מבינה לאן להיכנס בעזרת זה שהיא בודקת באיזה syntax הפקודה נשלחה, אם הפקודה לא מזוהה לתוכנית היא מדפיסה למשתמש שאין פקודה כזאת.

אופן ההדפסה של כל שאלה ותשובה עובד על ידי שיש רשימה של אובייקטים וכל אובייקט בנוי כך:

**Message** – הדבר שהשרת שלח או הבקשה של המשתמש

**Place** – הקוד של זיהוי ההודעה 0 – 9, קוד 0 מסמן שזאת הודעה על ידי הלוקח וכל השאר הודעות על ידי השרת.

אז יש לנו רשימה של אובייקטים שעכשיו קל לעבוד איתם, פעולת ההדפסה בנוי כך שאני משתמש בmap בשביל להדפיס את ההודעה של המשתמש או של הלוקח, התוכנית יודעת לאן לשים כל הודעה (ימינה או שמאלה) לפי התנאי בclassName שאם הplace שווה ל0 זה בימין וכל שאר הקודים בשמאל.

עכשיו יש לנו גם כמה תנאים בהמשך והם קצת אורכים כל תנאי מוביל לאיך ההודעה אמורה להיכתב ובאיזה פורמט.

הספריות צד שלישי שהשתמשתי בהן:

**Express** – ספרייה זאת מאפשרת לתוכנית ליצור tab חדש שיהיה שייך רק לפונקציה הזאת לשלוח לשם את המידע בשביל שהצד לוקח שלנו יכול לקחת אותו ולשמור אותו בהמשך זאת ספרייה נהדרת וקלה וזה למה השתמשי דווקא בה.

**Axios** – ספרייה זאת מאפשרת לקחת את כל המידע מן אתר שאנו בוחרים, בשלב זה השתמשתי בספרייה זו בשביל לקחת את כל המידע מlocalhost:3001/getAttacksNames ולשמור אותם לתוך משתנה ואז לעבוד בהמשך עם אותו משתנה.

**MySQL** – בניגוד לשלב הקודם כאן את צריך את הספרייה הזאת בשביל לא להכניס לתוך המסד הנתונים אלה בשביל לשלוף ממנו את המידע או במקרה שלנו לשלוף את השם ואת התיאור של התקפות.

**Curl** – ספרייה זו עוזרת מכיוון שבלי אותו ספרייה לא היינו יכולים לקשר את הצד שרת וצד לוקח בגלל שצד לקוח לא מזהה את הצד שרת ואומר שהוא לא בטוח ועם ספרייה זו הכל מתוקן והם מזהים אחד את השני זה למה השתמשי בספרייה זו.

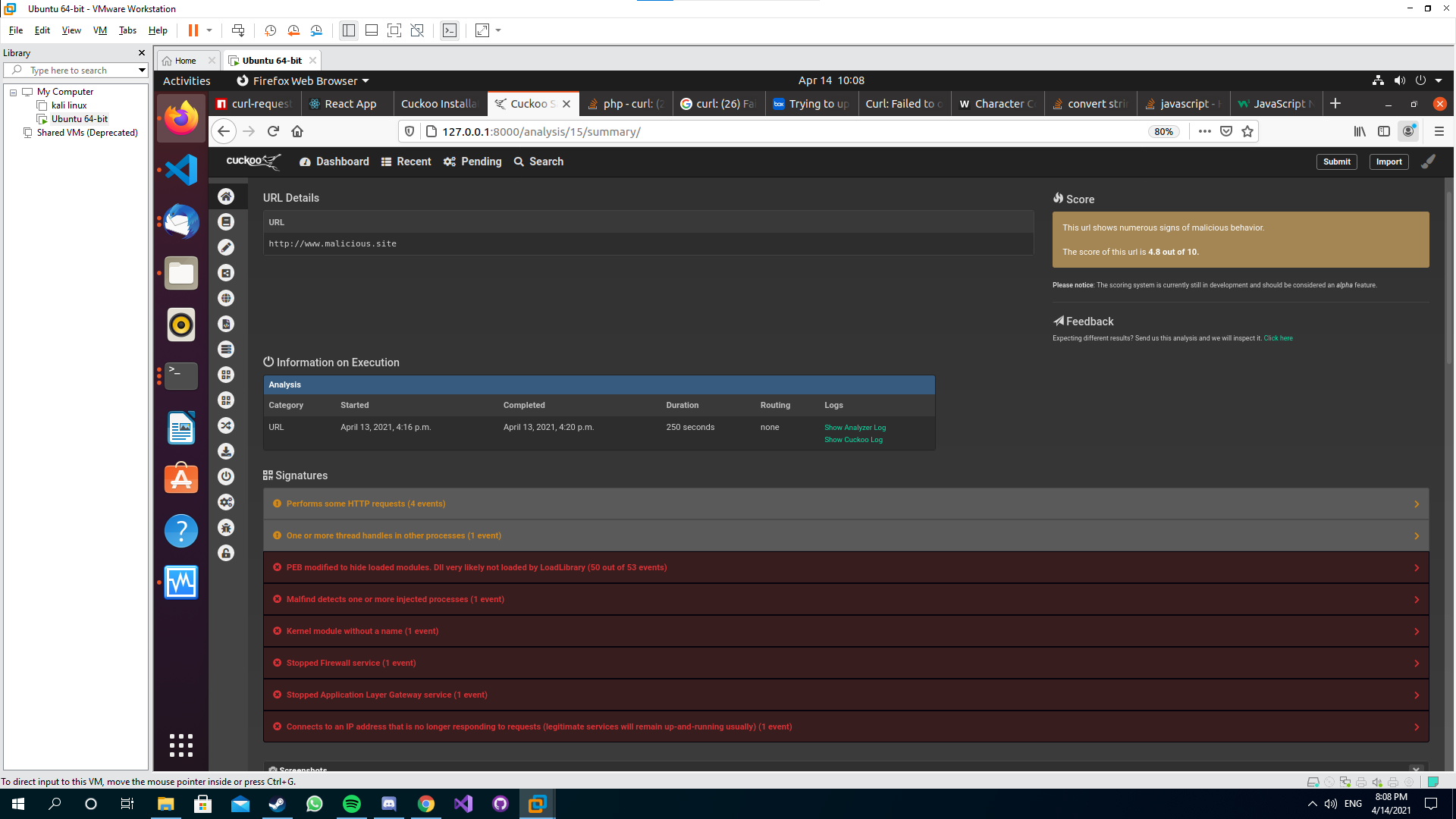
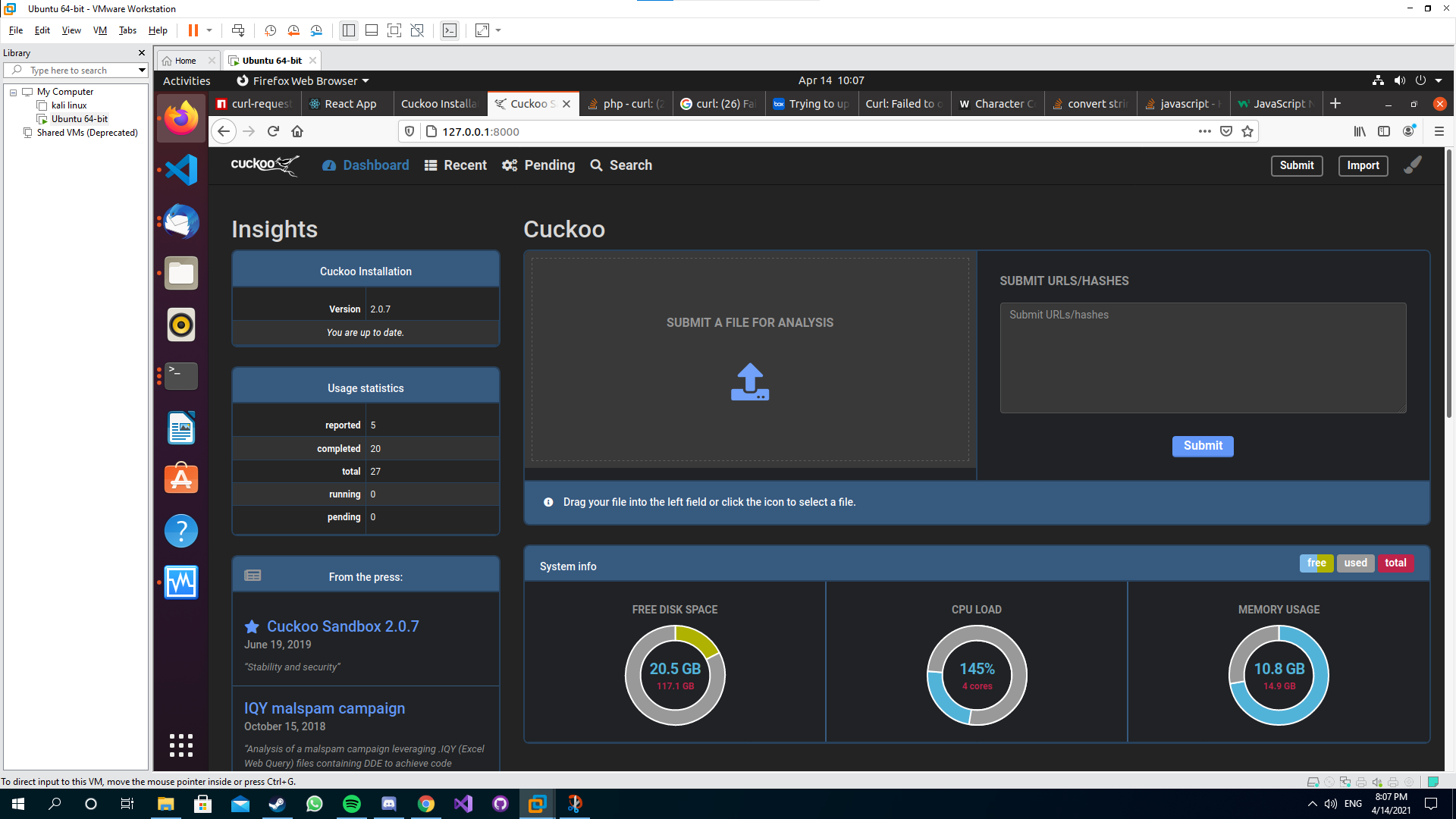
**VirusTotal** – זו היא ספרייה אשר יכולה לבדוק את האתר במקרה שלנו אם יש עליו וירוסים או הוא בטוח, אני בחרתי בספרייה זו מכיוון שיש לה הסבר נוח ברור באינטרנט וגם בגלל שהיא קלה להבנה וממשט את הצרכים שלי לתוכנית הזאת.

בחרתי לבנות את הchatBot ככה מכיוון שלדעתי המשתמש יכול לקרוא את הפוקדת help וישר להבין את הsyntax שהוא לא קשה למדי, וגם כשיש syntax לתוכנית הרבה יותר קל לבדוק את התשובה של המשתמש ולפענח ולהוביל אותו למקום הנכון.

**שלב 3:**

שלב 3 הוא השלב האחרון הפרויקט בו התבקשתי להתקין cuckoo sandbox שזה api המאפשר לחקור קבצים וכתוב קוד הבנוי על הchatbot שיהיה אפשר לשלוח קובץ ולקבל עליו משוב.

דבר ראשון cuckoo sandbox משתמש ב Ubuntu וגם windows 7 בו זמנית אז קודם כל הייתי צריך אותו אחרי חיפוש מעמיק באינטרנט מצאתי מדריך המסביר איך להתקין אותו מהאתחלה עוד הסוף אחרי התקנה ממושכת יחסית קשה הצלחתי לעשות זאת והינה תמונות מסך:

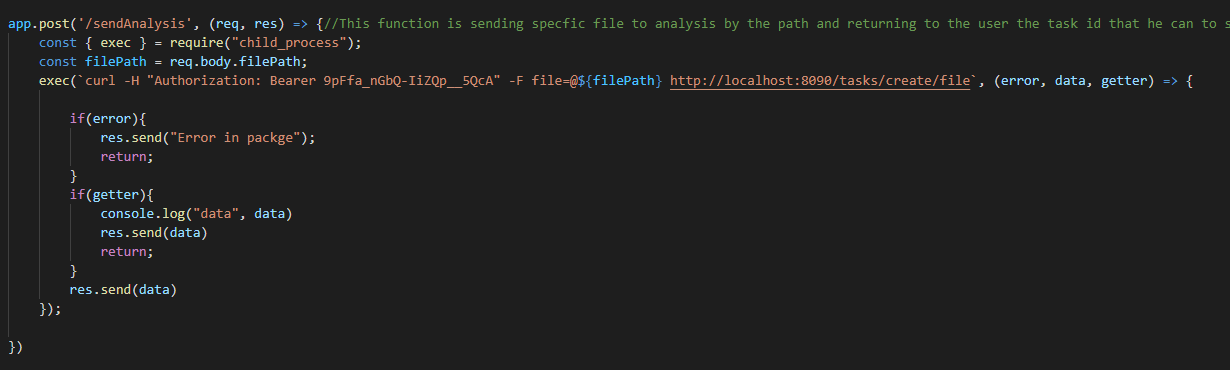


cuckoo sandbox נותן אופציה להשליך עליו כל קובץ או אתר והוא מחזיר עליו משוב כפי שאפשר לראות.

אחרי ההתקנה ישר הלכתי לחפש באינטרנט איך ממשים את כל הפעולת השליחה והקבלה בקוד ומצאתי שאת פעולת השליחה אפשר לעשות עם

Child-process שעליו אני אסביר עליו בהמשך אופן פעולת השליחת קובץ ל cuckoo sandbox היא פשוטה אנו עושים שאילה לתוך הbash עם פקודה מובנת של cuckoo sandbox בשביל לשלוח קובץ וזה מריץ לנו ומחזיר task id שאיתו נעבוד בהמשך. התמנות מסך של הקוד:

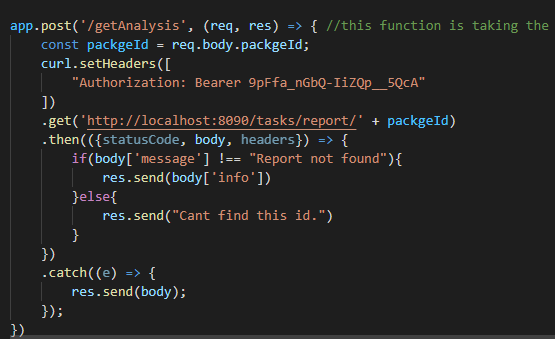
**Syntax: analysis <path> file**



אפשר לראות שאנו עושים שאילה לbash ומקבלים תשובה שולחים אותה loclhost:3001/sendAnalysis ומשם כל הפעולה דומה לפעולת הchatbot מיכון שהוא הבסיס שלו להצגת המידע. המידע שמוצג למשתמש הוא ה task id בשביל לבדוק אותו בהמשך.

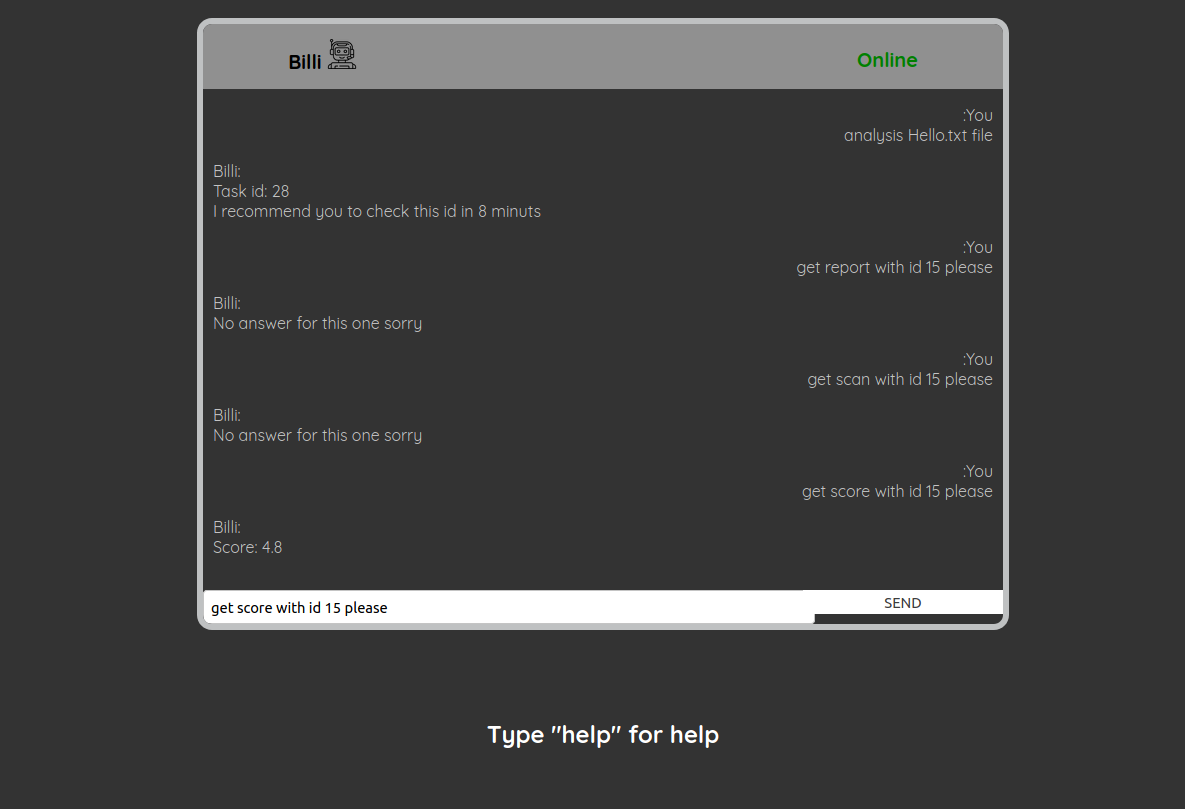
פעולה שנייה שיש לנו היא פעולת על קבלת הסיכום על הקובץ שאנו בדקנו לפני זה, אנו בודקים זאת לפי task Id שהמשתמש שולח, אחרי שהוא שולח אותו cuckoo sandbox מחפש אותו במאגר שלו ומחזיר למשתמש האם הקובץ בטוח או לו כלומר מציג לו נקודות 0 – 10.

**Syntax: get report with <id> please**



התוכנית משתמש בcurl-requset בשביל לעשות שאילתה ל cuckoo sandbox שהיא שאילתת HTTP וה cuckoo sandbox מחזיר לו מידע בהתאם ושולח אותו ל localhost:3001/getAnalysis ומשם הפעולה של להראות למשתמש דומה לכל הפעולת chatbot.

דוגמא לתוצאה של שני הפקודות:



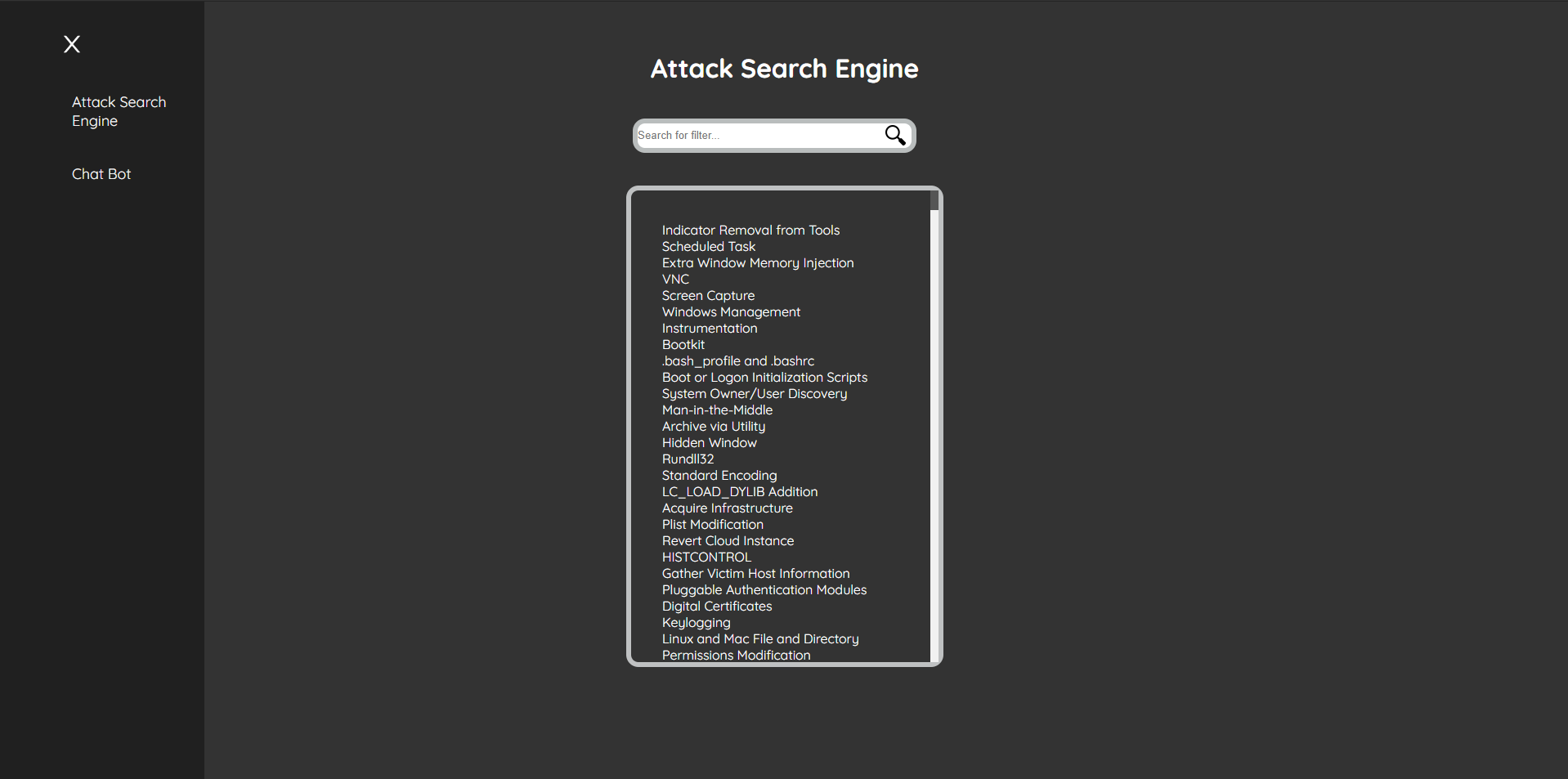
הספריות הצד שלישי שהשתמשתי בהן:

**Child-process** – ספרייה זו נותנת אופציה לתוכנית להריץ פקודות bash עם node js שממש נוח כרגע מכיוון שהוא פשוט להבנה.

**Curl-requset** – ספרייה אשר נותנת לנו אופציה לעשות requests לאתר כלשהו במקרה שלנו היינו צריכים ל cuckoo sandbox כדי לקחת את הסיכום לגבי הקובץ, השתמשי בו בגלל שהוא נוח קל וגמיש באפשרויות וממש מתאים לצרכים של התוכנית הזו.

**משהו נחמד שהוספתי:**

בשביל להתנייד נוח בין שני האתרים יצרתי nav bar מצד שמאל במסך שנפתח ונסגר על ידי כפתור הוא חדשני ומוסיף לעיצוב.



**תודה רבה על הקריאה ☺ קובץ עם התשובות לשאלות הבאות מופיע באותה תיקיה.**