C:\Users\nesia.cohen\Desktop\ספר פרויקט\סמל בית יעקב.wmf

מערכת מעקב אחר דפי אינטרנט ודיוור התוכן ללקוחות

**שרה גורביץ | יהודית כהן |2019**

**שלמי תודה**

בפתח דבר נודה לבורא עולם שהחיינו, קיימנו והביאנו עד הלום וסייע לאורך כל הדרך.

בנוסף נודה לכל אלו שנתנו רבות מכוחם, מרצם ומתבונתם במהלך העבודה.

🙘תודה **-** לגב' רבקה הנאו ולגב' שרה רבקה פוירשטין - מנחות, שסייעו לנו לאורך כל הדרך, החל משלב התכנון ועד לשלב הביצוע. ייעצו, בדקו ונתנו חוות דעת על מנת להגיע להישגים המקסימליים ולפרויקט ראוי ומושלם.

🙘 תודה - למרכזת גב' הדסה גרוס אשר השקיעה וטרחה רבות למעננו.

🙘 תודה - לצוות חדרי מחשבים, וביחוד לגב' תהילה אלקריף על מענה בכל צורך ובכל זמן.

🙘תודה - לצוות המורות המסור על המקצועיות, העזרה הרבה, התמיכה וההשקעה.

🙘 לסמינר מרכז בית יעקב, ולעומדים בראשו הרב ישעיה' ליברמן, הרב ישראל ליברמן גב' מינה זלזניק על החינוך, המחשבה, המסירות והמקצועיות הרבה, וההשקעה לטווח הרחוק .

🙘 תודה - להורינו היקרים ולשאר בני המשפחה וידידים על תמיכה ועידוד לאורך כל הדרך החל מתקופת הלימודים ועד להגשת פרויקט גמר.

**תקציר**

המערכת מתחלקת לתת מערכות המממשות צרכים שונים של המנגנון השלם: -

* אתר המסופק ללקוחות ומאפשר רישום, ניהול תוכן ושירותים נגישים. האתר מבוסס Angularומעדכן את מסד הנתונים של המערכת.
* מערכת .Net להפקת נתונים מאתרים, המתבססת על מסד נתונים sql ומשתמשת בגלישה רובוטית. מערכת זו מכילה אלגוריתמים מורכבים לייעול מהירות ואיכות. מערכת זו מתוזמנת ע"י .Windows service
* extension ל – Google Chrome מיועד לבחירת תוכן האתר ע"י הלקוח, קליטת סיסמאות, רישום פעולות וסינון מידע. התוסף נכתב ב html ו JavaScript..

הפיתוח כולל אפיון, תכנון מסד נתונים, כתיבת שאילתות, הקמת service , בניית אתר לתוכנה, פיתוח תוסף ל Chrome וכן עיצוב גרפי ב CSS.

תוכן עניינים

[**1 תיאור הפרויקט** 6](#_Toc15230611)

[1.1 תיאור כללי 6](#_Toc15230612)

[**2 מפרט טכני** 8](#_Toc15230613)

[2.1 מפרט לעמדת משתמש 8](#_Toc15230614)

[2.2 מפרט לעמדת שרת 8](#_Toc15230615)

[2.3 כלי התוכנה לפיתוח המערכת 8](#_Toc15230616)

[2.4 שפת התכנות 8](#_Toc15230617)

[**3 יעדי המערכת** 9](#_Toc15230618)

[**4 תיחום המערכת** 9](#_Toc15230619)

[**5 אלגוריתם מרכזי** 10](#_Toc15230620)

[**6 המודלים והקשרים ביניהם** 11](#_Toc15230621)

[6.1 Data Diagram 11](#_Toc15230622)

[6.2 Database 12](#_Toc15230623)

[6.3 מחלקות 19](#_Toc15230624)

[**7. מתזמן - Windows service** 36](#_Toc15230625)

[**8 ממשקי המערכת ומדריך למשתמש** 40](#_Toc15230636)

[8.1 מדריך למשתמש באתר 40](#_Toc15230637)

[8.2 extension ל Google Chrome 41](#_Toc15230638)

[8.2.1 מדריך התקנת extension ל Google Chrome 41](#_Toc15230639)

[8.2.2 מדריך למשתמש בתוסף 43](#_Toc15230640)

[**9 מסקנות** 49](#_Toc15230641)

[**PIPE** 49](#_Toc15230645)

[**השימוש בו** 49](#_Toc15230646)

[**10 ביבליוגרפיה** 50](#_Toc15230647)

**1 תיאור הפרויקט**

## 1.1 תיאור כללי

מעקב אחר שינוים ברשת הוא צורך מורכב. על מנת לבנות תוצר יעיל ונוח לשימוש, עלתה הדרישה לחלק את המערכת למספר רכיבים שמתפקדים יחד כמערכת.

* אתר אינטרנט
* מסד נתונים
* מערכת להפקת נתונים מאתרים, איתור שינויים ועדכון המשתמשים.
* extension ל – Google Chrome

משתמש המעוניין להצטרף לשרות נרשם באתר האינטרנט מגדיר פרטים, כתובת מייל ומוריד משם extension ל – Google Chromeהמבצע הקלטה של פעולות הגלישה בדפדפן.

כאשר המשתמש מוצא מידע עליו הוא מעונין לעקוב הוא מעדכן את התוסף על כך וההקלטה נשמרת.  
המשתמש לא יסתפק בכתובת URL בלבד וידרוש הקלטה מכיוון שמעוניין במידע ספציפי שלא מגיע מידית בטעינת האתר אלא לאחר מספר פעולות כמו סיסמאות, הזנת נתונים או מעבר לתתי דפים.

באתר האינטרנט המשתמש מגדיר באילו תדירויות הוא מעוניין להתעדכן, וכן יכול לנהל את הבקשות הקודמות.

נתוני המשתמש והקלטות הגלישה נשמרים במסד נתונים SQL בשרת, הקבצים הן של ההקלטה והן של תוכן הדף נשמרים בטבלה מיוחדת FileTable.

בשרת מתוזמנת בתדירות קבועה מערכת שתפקידה לבצע בפועל מעקב אחר דפים ברשת.  
המערכת שולפת את נתוני הגלישה של הבקשות שמוגדרות לתזמון הנוכחי, מבצעת גלישה רובוטית ומקבלת את דף הHTML המעודכן.  
המערכת משווה בין הגרסאות בודקת אם היו שינויים רלוונטיים ומדגישה אותם בדף הHTML כדי לחסוך קריאה של הדף כולו.  
המערכת שולחת את דפי הHTML הנ"ל לחשבון המייל של המשתמש.

**2 מפרט טכני**

## 2.1 מפרט לעמדת משתמש

* דפדפן Google Chrome
* חיבור לרשת
* חשבון מייל

## 2.2 מפרט לעמדת שרת

* שכבת מסד הנתוניםSQL Server 2014 -
* Imacros
* חיבור לרשת
* Windows service

## 2.3 כלי התוכנה לפיתוח המערכת

Visual Studio 2017

Visual Studio Code

## 2.4 שפת התכנות

* C#
* Angular
* JavaScript

**3 יעדי המערכת**

מטרת המערכת לבצע מעקב אחר דפי אינטרנט ודיוור התוכן ללקוחות באופן יעיל ונגיש. לצורך כך הוגדרו היעדים הבאים:

* ניהול נתוני המשתמשים
* הקלטת פעולות הגלישה בדפדפן
* הפעלת המתזמן – Windows service בשרת.
* גלישה רובוטית לקבלת נתונים מעודכנים
* השוואת גרסאות בצורה יעילה אמינה ואיכותית
* הדגשת שינוים בדפי HTML
* שליחת נתונים עדכניים ללקוח

**4 תיחום המערכת**

* הקלטת פעולות הגלישה תתבצע בדפדפן Google Chrome בלבד.
* המערכת לא מטפלת בדפי HTML שאינם תקינים.
* המערכת אינה מתחיבת ל100 אחוזי הצלחה, בדיוק זיהוי השינויים.

**5 אלגוריתם מרכזי**

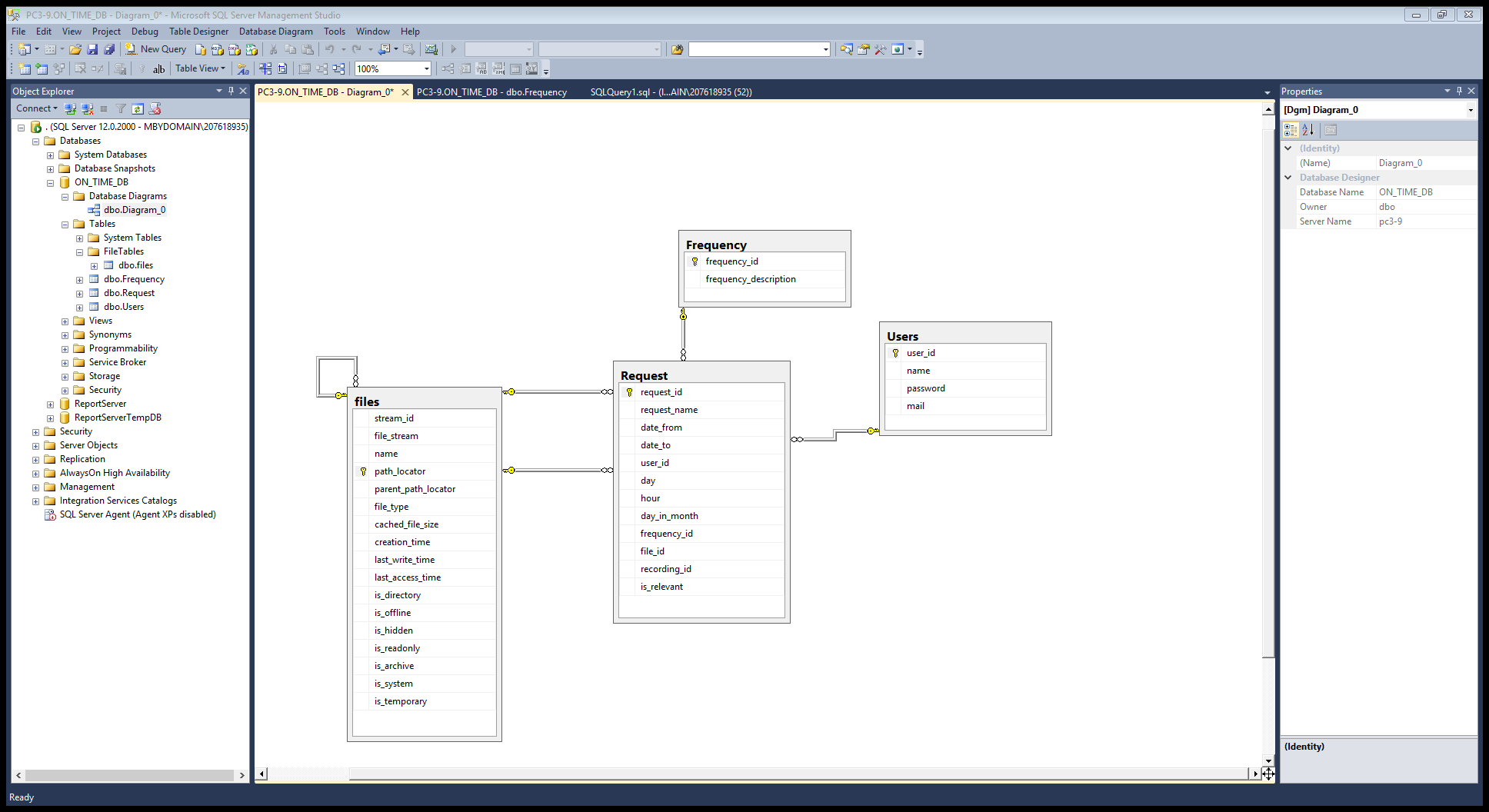
האלגוריתם המרכזי בפרויקט הוא מציאת הבדלים בין הגרסאות השונות של דפי HTML וסימונם.

לאחר ביצוע "גלישה רובוטית" ברשת וקבלת המידע המעודכן ביותר, יש להחליט האם המידע מצדיק שליחה מחדש של הדף ללקוח- האם נערכו בדף הHTML שינוים הרלוונטיים ללקוח.  
ההבחנה האם קיים הבדל בין הגרסאות פשוטה (ניתן לרוץ בצורה סדרתית ולהשוות כל תו). אך אין אפשרות להסתפק בה אלא יש לזהות את השינויים מכיוון שהם נדרשים לתהליכים הבאים:

* החלטה האם השינויים שזוהו מצדיקים שליחה מחדש של הדף – האם הם רלוונטיים ללקוח.
* הדגשה של השינויים כדי לחסוך ללקוח קריאה מחדש של הטקסט כולו העשוי להיות ארוך ומיגע.

**6 המודלים והקשרים ביניהם**

6.1 Data Diagram



6.2 Database

**Users** – טבלת משתמשים  
בטבלה זו שמורים כל המשתמשים

|  |  |  |  |
| --- | --- | --- | --- |
|  | שם | סוג | הסבר |
|  | user\_id | int | קוד משתמש - מספר רץ |
|  | name | string | שם |
|  | password | string | סיסמה |
|  | mail | string | מייל |

**Frequency**– תדירות לבקשה

בטבלה זו שמורים רמות תדירויות בהם יבדקו שינויים באתר.

|  |  |  |  |
| --- | --- | --- | --- |
|  | שם | סוג | הסבר |
|  | frequency\_id | int | קוד תדירות |
|  | Frequency\_description | string | תיאור תדירות |

**Request**– טבלת בקשות שרות  
בטבלה זו שמורים פרטי בקשות השרות של המשתמש.

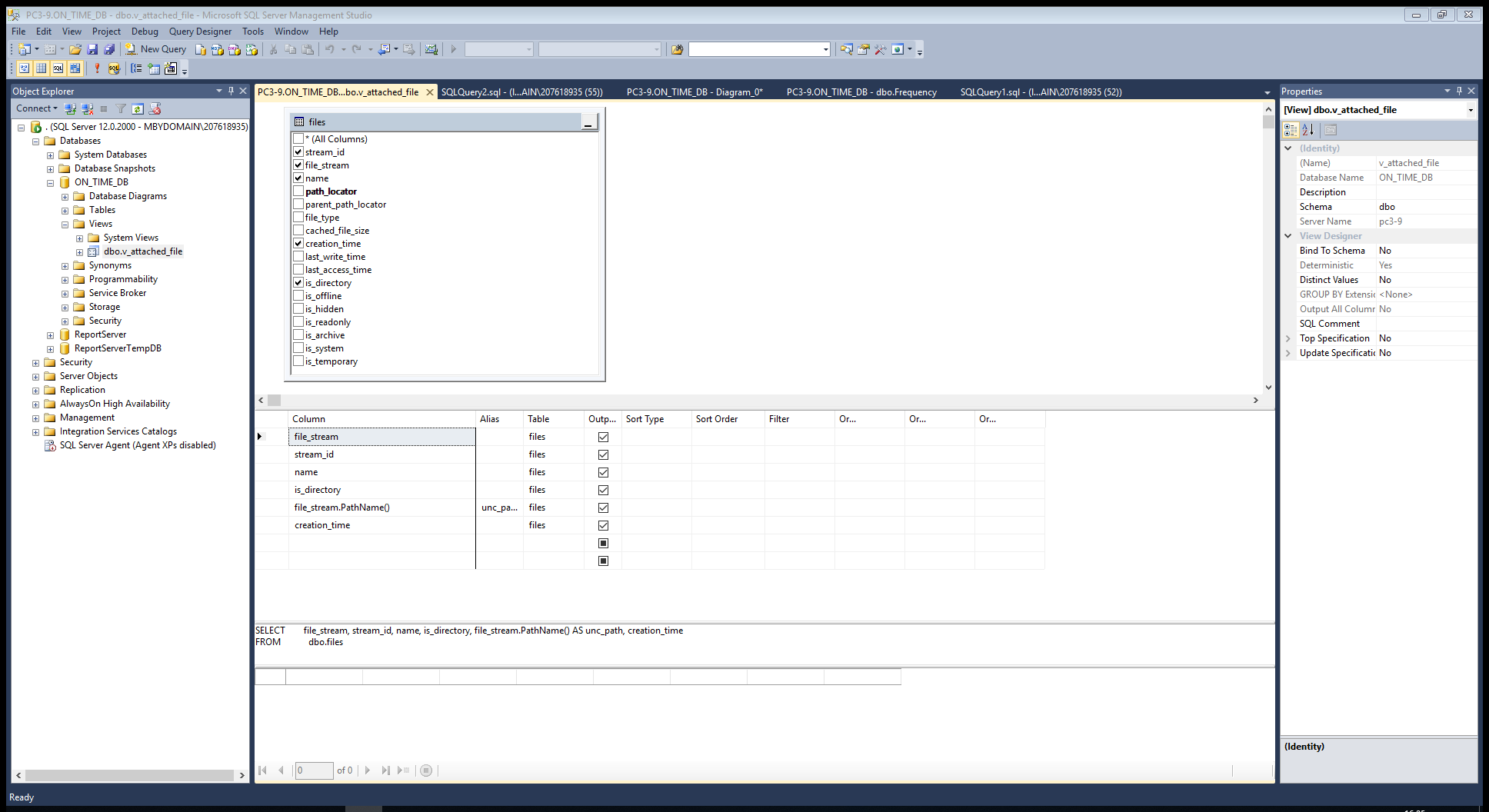
|  |  |  |  |
| --- | --- | --- | --- |
|  | שם | סוג | הסבר |
|  | request\_id | int | קוד בקשת שרות – מספר רץ |
|  | user\_id | int | קןד משתמש |
|  | request\_name | string | תיאור הבקשה |
|  | date\_from | date | תאריך תחילת הבקשה |
|  | date\_to | date | תאריך סיום הבקשה |
|  | frequency\_id | int | קוד תדירות |
|  | hour | int | שעה מסוימת ביום |
|  | day | int | יום מסוים בשבוע |
|  | day\_in\_month | int | יום מסוים בחודש |
|  | file\_id | Guid-uniqueidentifier | קישור לקובץ של תוכן הדף |
|  | recording\_id | guid | קישור לקובץ ההקלטה |
|  | Is\_relevant | Bit-bool | האם הבקשה רלונטית / מחוקה |

**Files**– קבצים  
בטבלה זו שמורים הדפים וההקלטות של המשתמשים.

עבור כל בקשה לניטור יש לשמור שני קבצים: קובץ ההקלטה של הגלישה, וקובץ html של הדף האחרון המעודכן.

הקבצים נשמרים בטבלה מסוג FileTable המיועדת לשמירת קבצים.

|  |  |  |  |
| --- | --- | --- | --- |
|  | שם | סוג | הסבר |
|  | Stream\_id | uniqueidentifier | מזהה ייחודי עבור נתוני FILESTREAM |
|  | File\_stream | string | מכיל את נתוני FILESTREAM |
|  | name | string | שם הקובץ או הספריה. |
|  | Path\_locator | int | המיקום של הצומת הזה בהיררכיה של- FileNamespace |
|  | Parent\_path\_locator | int | היררכיה של הספרייה המכילה |
|  | File\_type | string | מייצג את סוג הקובץ |
|  | Cached\_file\_size | int | הגודל בבתים של נתוני FILESTREAM. |
|  | Creation\_time | date | התאריך והשעה שבהם נוצר הקובץ. |
|  | Last\_write\_time | date | התאריך והשעה שבהם הקובץ עודכן לאחרונה. |
|  | Last\_access\_time | date | התאריך והשעה שבהם הייתה הגישה האחרונה לקובץ. |
|  | Is\_directory | bit | מציין אם השורה מייצגת ספריה. |
|  | Is\_offline | bit | האם הקובץ לא מקוון. |
|  | Is\_hidden | bit | האם הקובץ מוסתר. |
|  | Is\_readonly | bit | האם הקובץ לקריאה בלבד. |
|  | Is\_archive | bit | האם הקובץ ארכיון. |
|  | Is\_system | bit | האם הקובץ מערכת. |
|  | Is\_temporary | bit | האם הקובץ זמני. |

הערה(כאן או בserver)

על מנת להשתמש בטבלה זו ב entity framework יצרנו view שמאפשר לנו להתנהל מול השדות הרלווטים בטבלה.

על מנת להוסיף קבצים לטבלה יצרנו פרוצדורה שמקבלת את שם הקובץ ותוכנו. ומחזירה את הקוד.

USE [ON\_TIME\_DB]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[attached\_file\_add] Script Date:

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[attached\_file\_add](@file\_name nvarchar(255),@file\_data varbinary(max),@file\_id uniqueidentifier OUT)

AS

BEGIN

SET @file\_id=NEWID();

INSERT INTO files(stream\_id,file\_stream,name) VALUES(@file\_id,@file\_data,@file\_name);

SELECT @file\_id

RETURN

END

GO

ארכיטקטורות ושפות הפיתוח

בבואנו לכתוב את התוסף עמדו בפנינו מספר פרמטרים שונים ועל פיהם בחרנו את הארכיטקטורות

המתאימות והיעילות ביותר:

מבחינה מקצועית- בחירת שפה לכתיבת הפרויקט כך שתתאים לצרכי התוסף, תיתן מענה לדרישות

השונות, ותאפשר רמה גבוהה של פונקציות ואפשרויות טכנולוגיות, כמו כן, רצינו שפת תכנות עדכנית

שתאפשר לנו לקנות ניסיון טוב ומבוקש בעולם התכנותי.

ולכן בחרנו לכתוב צד לקוח ב -אנגולר שהינה שפה מתקדמת ועדכנית בעלת מאפייני Angular5 חדשניים

ופונקציונאלית ביותר.

אנגולר הינה )סביבת עבודה( שפותחה על ידי גוגל. מאפשרת לפתח אפליקציות Framework אינטרנט

בקלות ומהירות. במקור היא באה לתת מענה לבנית Applications Page Single בצורה מושלמת

ומהירה. מהיתרונות הבולטים והעיקריים של אנגולר אפשר למנות: חיסכון במשאבים ,מהירות ביצוע

,קוד קצר יותר, רוב העבודה מתבצעת בצד הלקוח ופחות בשרת ויכולת התמודדות טובה )סינון מהיר

ופשוט לביצוע(, של תוכן המתקבל מהשרת לפי מספר רב של פרמטרים.

צד שרת בחרנו לכתוב ב- C# .#C היא שפת תכנות עילית מרובת-פרדיגמות, מונחית עצמים בעיקרה,

המשלבת רעיונות כמו טיפוסיות חזקה, אימפרטיביות, הצהרתיות, פונקציונליות, פרוצדורליות וגנריות.

C #היא שפה מעניינת, נוחה ומלאה פונקציונאליות למתכנת. שימוש בשפה זו נפוץ כיום וכתוצאה מכך,

ניתן היה למצוא בה קודים שונים שנדרשו לפיתוח. כמו כן, לצורך קבלת המידע מתיבת המייל השתמשנו

בספריות השונות המובנות מתוך Apis.Google ובמבחר הפונקציות והיכולות שיש להן להציע.

בנוסף - בחרנו לבנות את הפרויקט במודל שמפריד את ה BL מהחלקים השונים, הפרדה זו מיועדת כדי

לאפשר בעתיד להוסיף ולשכלל את הקוד ביתר קלות ויעילות.

כך שהמשתמש פונה למערכת דרך שכבת הUI והיא פונה לAPI שמפנה אותה לפונקציונאליות

המתאימה בBL.

לצורך ייעול ההפרדה מחד, וטעינת הנתונים מה Base Data מאידך בחרנו להשתמש ב

EntityFramwork- טכנולוגית עבודה מתקדמת של מיקרוסופט.

ה EntityFramwork מאפשר לטעון את הנתונים מה DB ולעשות להם השמה בצורה ישירה ואוטומטית

לתוך אוביקטים בקוד הממפים את מאגר הנתונים בצורה מידית.

השימוש ב EntityFramwork נעשה באמצעות הגדרת קובץ מודל להגדרת המחולל classes לעבודה מול

ה DataBase , כך שנוצרת ישות לכל טבלה.

הישות הנוצרת מכילה אתחולים, מאפיינים )properties ,)וקשרים עבור הטבלה הספציפית וכן שדות

חדשים עבור קשרים )של join . )הודות לישויות הללו שנוצרו על ידי קובץ המודל ניתן להגדיר מופע

)instance )מסוג הטבלה.

מבחינת חווית משתמש- לאפשר את מקסימום הנוחות והבהירות האפשריים בתוסף, כך שיוכל

להתאים לכל משתמש, גם לכזה שאינו בעל ידע תכנותי. וכן ליצור עיצוב קליל ,יפה ומשכנע כדי לייצור

חויית משתמש עשירה ומשכנעת.

לצורך כך עיצבנו את האתר בbootstrap מאחר ואנגולר לא מייצרת UI מספיק עשיר ויפה.

bootstrap משלימה גם את היופי וגם מתאימה לפילוסופיה של אנגולר. bootstrap עוסקת בעיצוב

הפרויקט ובנראותו, ע"י אובייקטים שונים שהיא מציעה

כמו כן נעזרנו בקומפוננטות המובנות של ספרית primeNG.

PrimeNG הוא אוסף של רכיבי ממשק משתמש עשירים עבור אנגולר. כל היישומונים הם קוד פתוח ללא תשלום תחת רישיון PrimeNG. MIT פותחה על ידי Informatics PrimeTek ,ספקית בעלת ניסיון רב שנים בפיתוח פתרונות ממשק פתוח של קוד פתוח.

אבטחה ושרידות

שמירה נתונים מוגנת

עבור כל בקשה לניטור יש לשמור שני קבצים: קובץ ההקלטה של הגלישה, וקובץ html של הדף האחרון המעודכן.

ישנן שתי אפשריות כיצד לשמור קבצים :

* לשמור את הקבצים על המחשב המקומי ובsql לשמור נתיב.
* לשמור את הקבצים עצמם בsql בטבלה מיוחדת שנקראת .file table

הדרך שנבחרה היא הדרך השנייה למרות מורכבותה שאינה מתנהגת כטבלה רגילה, זאת מכיוון שבה הנתונים נשמרים בבטחה הודות לקשרי גומלין של הטבלה עם טבלאות אחרות, לעומת האפשרות של שמירת קבצים בשרת שבה הנתונים אינם מוגנים ממקרה של מחיקה.

שמירת נתונים מוצפנת

כיון שבפרויקט נשמרים נתונים אישים חסויים רבים על מנת לגלוש לתתי דפים כגון: פרטי חשבון בנק, כרטיסי אשראי סיסמאות ועוד. עלה הצורך להתייחס לאבטחת הקבצים ותוכן ההקלטה. ועל כן הנתונים מוצפנים טרם שמירתם במסד הנתונים.

המימוש נעשה בclass

class Encrypt\_Decrypt

בקלאס קיימת פונקציה להצפנה ופונקציה לפענוח.

public static string EncryptStringAES(string plainText, string sharedSecret)

public static string DecryptStringAES(string cipherText, string sharedSecret)

דוגמאות למימוש:

הנתונים המקוריים:

Encoding.UTF8.GetString(requestDto.recording\_stream)=

VERSION BUILD = 1005 RECORDER = CR

URL GOTO = https://stackoverflow.com/questions/43008395/error-encountered-undefined-provider-usually-this-means-you-have-a-circular-d

SAVEAS TYPE = HTML FOLDER =D:\Files FILE =newHtml

לאחר ההצפנה:

Encrypt\_Decrypt.EncryptStringAES(Encoding.UTF8.GetString(requestDto.recording\_stream), "onTime123")=

EAAAAHA2/XRwzvRjwkL2ZaXyjzt1+1nJZBUMktRmuP9g/0+fX76nljoMBQBSsR02o/GBSKENAVJD85xtLJhLSluyqPxF2UalzbaRfXP5JiHSdjDd90/1hkRMa7ZpHY4RGkcI2gKAsNnRJaV4X3jTBcynLElmTdS7u9uMeDusque+0BaH+8rA8UUEDZFf/yyTPmL+cWaNAHMmWpYPAeFo16r2TMJdEm+dU4uUULZJJu0ZV43jzaGWIEBfriuA1eivUSiZquigyOvNPvXZ40mbU6eSLNFFairs53CFVmf4aoHWCiz3y+P8BHv5E8A/H3IA2OfXSw==

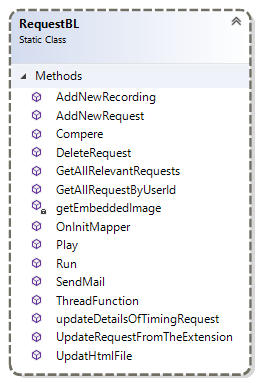
אופן שמירת הנתונים

למסד הנתונים של ה SQL Server -יש כלים נרחבים לגיבוי כל המידע של המערכת, כולל מערכת

ההפעלה, חשבונות המשתמשים והרשאותיהם, הגדרות ההתקנים, תוכניות וכן של שאר הרכיבים

המסופקים עם השרת ואובייקטי המשתמש.

6.3 מחלקות

**BL- Business Logic שכבה לוגית**

**RequestBL**

מחלקה זו אחראית על הטיפול בבקשות:

פונקציות הנקראות מהcontroller:

* הוספת בקשה חדשה
* עדכון פרטי בקשה קיימת
* מחיקת בקשה
* שליפת כל הבקשות הרלוונטיות
* שליפת בקשות של משתמש מסוים
* הוספת הקלטה חדשה

פונקציות הנקראות מהservice:

* השוואת תוכן דפים
* גלישה רובוטית
* שליחת מייל

**פונקציונליות**

**AddNewRequest –** פונקציה המקבלת אוביקט מסוג RequestDto, משנה את תוכן השדה content מURL בלבד להקלטה, שולחת לגלישה רובוטית לצורך שמירת דגימה ראשונית של הדף המבוקש.

**מימוש:**

// add new request

public static RequestDto AddNewRequest(RequestDto requestDto)

{

// converts the URL to a recording format

requestDto.content = "VERSION BUILD = 1005 RECORDER = CR\nURL GOTO = " + requestDto.content + " \nSAVEAS TYPE = HTML FOLDER =D:\\Files FILE =newHtml";

requestDto.recording\_stream = Encoding.ASCII.GetBytes(requestDto.content);

Status status = Run(requestDto.content, "");

// Robotic surfing for keeping the initial state of the page

if (status == Status.sOk)

{

// Read the page content

string newPlay = File.ReadAllText("D:\\Files\\newHtml.htm");

// fill in the data

requestDto.file\_stream = Encoding.ASCII.GetBytes(newPlay);

requestDto.file\_id = RequestDAL.AttachedFile(requestDto.file\_stream, ".html");

// add to database

RequestDto requestsDto = RequestDAL.AddNewRequest(requestDto);

return requestsDto;

}

return null;

}

**updateDetailsOfTimingRequest –** פונקציה המקבלת אוביקט מסוג RequestDto, ומעדכנת את הפריטים שהשתנו, כגון בחירת תזמון שונה, החלפת שם.

**מימוש:**

// update the request's timing details

public static RequestDto updateDetailsOfTimingRequest(RequestDto requestDto)

{

return RequestDAL.updateDetailsOfTimingRequest(requestDto);

}

**UpdateRequestFromTheExtension –** פונקציה המקבלת אוביקט מסוג RequestDto, ומעדכנת את הפריטים שהשתנו, כגון בחירת תזמון שונה, החלפת שם.- לשימוש כאשר תוכן ההקלטה הגיע מהתוסף.

**מימוש:**

// update request details - Recording was done through the extension

public static RequestDto UpdateRequestFromTheExtension(RequestDto requestDto)

{

try

{

if (requestDto.recording\_id == new Guid())

{

RequestDto requestsDto = RequestDAL.FillInDataRequest(requestDto);

Status status = Run(Encoding.UTF8.GetString(requestsDto.recording\_stream), "");

if (status == Status.sOk)

{

string newPlay = File.ReadAllText("D:\\Files\\newHtml.htm");

RequestDAL.UpdateHtmlFile(requestsDto, newPlay);

}

}

}

catch (Exception)

{

throw;

}

return requestDto;

}

**DeleteRequest –** פונקציה המקבלת אוביקט מסוג RequestDto,ומוחקת אותו ממסד הנתונים(מחיקה לוגית).

**מימוש:**

public static RequestDto DeleteRequest(RequestDto requestDto)

{

return RequestDAL.DeleteRequest(requestDto);

}

**GetAllRelevantRequests –** פונקציה השולפת את כל הבקשות הרלוונטיות לרגע זה (פרוט השליפה בשכבת הDAL )

הפונקציה נקראת מהמתזמן- Windows Service וממנו הוזרק אליה הLOG להדפסת אינפורמציה.

**מימוש:**

public static List<RequestDto> GetAllRelevantRequests(EventLog eventLog)

{

try

{

eventLog.WriteEntry("BL:In GetAllRelevantRequests.");

List<RequestDto> requestDtos = RequestDAL.GetAllRelevantRequests(eventLog);

return requestDtos;

}

catch (Exception ex)

{

eventLog.WriteEntry(ex.ToString());

return null;

}

}

**GetAllRequestByUserId –** פונקציה המקבלת קוד משתמש ומחזירה את כל הבקשות שלו.

**מימוש:**

// get all specific user requests

public static List<RequestDto> GetAllRequestByUserId(int userId)

{

List<RequestDto> requestsDto = RequestDAL.GetAllRequestByUserId(userId);

return requestsDto;

}

**AddNewRecording –** פונקציה היוצרת הקלטה חדשה ללא מילוי הפרטים של הבקשה – לשימוש התוסף .

**מימוש:**

// adding a recording - is called from the extension without filling request details yet

public static int AddNewRecording(RequestDto requestDto)

{

//Chaining to Record A command that will save the page as HTML

requestDto.content += " \nSAVEAS TYPE = HTML FOLDER =D:\\Files FILE =newHtml";

requestDto.recording\_stream = Encoding.ASCII.GetBytes(requestDto.content);

// add to database

return RequestDAL.AddNewRecording(requestDto);

}

**Compere –** פונקציה המקבלת תוכן HTML של שני דפים ומשווה בינהם במידה והיה שינוי מזירה את התוכן הדף הנוכחי בתוספת של STYLE שמזריח את השינוי, אחרת מחזירה מחרוזת ריקה. הפונקציה נקראת על ידי המתזמן

**מימוש:**

public static bool Compere(string newPlay, string file\_stream)

{

Algoritem m = new Algoritem();

bool ret = m.MyComparer(file\_stream, newPlay);

return ret;

}

**Run** – פונקציה המקבלת תוכן של ההקלטה וגולשת גלישה רובוטית שוב לאותו דף על מנת לקחת דגימה נוספת.

מימוש:

static public Status Run(string content,string fileId)

{

//string appDirectory = Directory.GetCurrentDirectory();

File.WriteAllText("D:/Files/mac"+ fileId + ".iim", content);

int timeout = 60, errors = 0;

iMacros.Status status;

var app = new iMacros.App();

status = app.iimInit("-V7", true, "", "", "", timeout);

if (status != iMacros.Status.sOk) errors++;

string macro = "D:/Files/mac" + fileId + ".iim";

status = app.iimDisplay("Interface version =\n" + app.iimGetInterfaceVersion().ToString(), timeout);

if (status != Status.sOk) return status;

status = app.iimPlay(macro, timeout);

if (status != Status.sOk) return status;

status = app.iimExit(timeout);

if (status != Status.sOk) return status;

return status;

}

**SendMail** – פונקציה המקבלת כתובת מייל של משתמש, תוכן HTML של הדף שהשתנה ושם הבקשה למעקב, ושולחת מייל למשתמש לידע אותו בשינוי / עדכון.

מימוש:

public static void SendMail(string userMail, string html, string title)

{

try

{

SmtpClient client = new SmtpClient();

client.Port = 587;

client.Host = "smtp.gmail.com";

client.EnableSsl = true;

client.Timeout = 10000;

client.DeliveryMethod = SmtpDeliveryMethod.Network;

client.UseDefaultCredentials = false;

client.Credentials = new System.Net.NetworkCredential("ontimemonitor@gmail.com", "onTime123");

MailMessage mm = new MailMessage("ontimemonitor@gmail.com", userMail, "Your request " + title + " has been updated", html);

mm.IsBodyHtml = true;

mm.BodyEncoding = UTF8Encoding.UTF8;

mm.DeliveryNotificationOptions = DeliveryNotificationOptions.OnFailure;

mm.AlternateViews.Add(getEmbeddedImage("logo.png"));

File.WriteAllText("D:/Files/" + title + ".html", html);

System.Net.Mail.Attachment attachment;

attachment = new System.Net.Mail.Attachment("D:/Files/" + title + ".html");

mm.Attachments.Add(attachment);

client.Send(mm);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

private static AlternateView getEmbeddedImage(String filePath)

{

LinkedResource inline = new LinkedResource(filePath);

inline.ContentId = Guid.NewGuid().ToString();

string htmlBody = @"<img src='cid:" + inline.ContentId + @"'/>";

AlternateView alternateView = AlternateView.CreateAlternateViewFromString(htmlBody, null, MediaTypeNames.Text.Html);

alternateView.LinkedResources.Add(inline);

return alternateView;

}

פונקציות שנקראות מהמתזמן:

public static string Play(byte[] recording,string requestId)

{

string newPlay;

recording = Encoding.ASCII.GetBytes(Encoding.UTF8.GetString(recording).Replace("newHtml", "newHtml" + requestId));

//TODOהפונקציה מקבלת הקלטה שהתוסף הוציא מומרת למערך של בתים

//צריך גלישה רובוטית

// הפונקציה מחזירה את הגלישה החדשה לצורך השוואה

Status status = Run(Encoding.UTF8.GetString(recording), requestId);

// eventLog.WriteEntry("status:" + status);

if (status == Status.sOk)

{

newPlay = File.ReadAllText("D:\\Files\\newHtml" + requestId + ".htm");

return newPlay;

}

return null;

}

public static void UpdatHtmlFile(RequestDto request, string newPlay)

{

RequestDAL.UpdateHtmlFile(request, newPlay);

}

public static void ThreadFunction(object request)

{

RequestDto requestDto = (RequestDto)request;

Console.WriteLine(requestDto.request\_id.ToString());

bool isDiff = false;

string newPlay = Play(requestDto.recording\_stream,requestDto.request\_id.ToString());

if (newPlay != null)

{

isDiff = Compere(newPlay, Encoding.UTF8.GetString(requestDto.file\_stream));

}

if (isDiff)

{

SendMail(requestDto.userMail, newPlay, requestDto.request\_name);

UpdatHtmlFile(requestDto, newPlay);

}

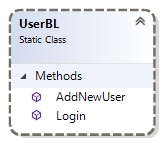
}

public static void OnInitMapper()

{

RequestDAL.OnInitMapper();

}



**UserBL**

מחלקה זו אחראית על התחום הלוגי הקשור במשתמשים:

זיהוי משתמש קיים והוספת משתמש חדש.

**Login** – פונקציה המקבלת כתובת מייל של משתמש וסיסמא ומחפשת אם הוא משתמש קיים במערכת, הפונקציה מחזירה את המשתמש.

מימוש:

//Existing user login

public static UserDto Login(string userMail,string password)

{

var user = UserDAL.Login(userMail, password);

return user;

}

**AddNewUser** – פונקציה המקבלת אובייקט מסוג userDto ומוסיפה אותו למאגר המשתמשים.

מימוש:

// add a new user

public static UserDto AddNewUser(UserDto userDto)

{

var user = UserDAL.AddNewUser(userDto);

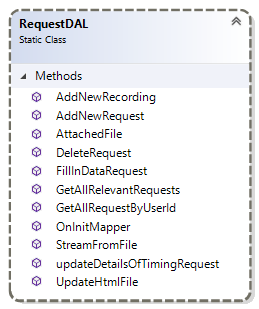
return user;

}

DAL - Data Access Layer

בשכבה זו מתנהלים שליפה, עדכון והוספה מ / אל מסד הנתונים על ידי שימוש בEntityFramework

**RequestDAL**



מחלקה זו אחראית על התנהלות מול המסד נתונים בכל הנוגע לטיפול בבקשות:

הוספת בקשה חדשה, עדכון בקשה קיימת, מחיקת בקשה, שליפת כל הבקשות הרלוונטיות, שליפת בקשות של משתמש מסוים.

וכן טיפול בדברים הנספחים:

הוספת הקלטה חדשה, הוספת קובץ וכו...

public static RequestDto UpdateHtmlFile(RequestDto requestDto, string html)

{

using (var db = new DBContext())

{

try

{

requestDto.file\_stream = Encoding.ASCII.GetBytes(html);

requestDto.file\_id = AttachedFile(requestDto.file\_stream, ".html");

var request = Mapper.Map<RequestDto, Request>(requestDto);

db.Requests.AddOrUpdate(request);

db.SaveChanges();

return requestDto;

}

catch (Exception ex)

{

return null;

};

}

}

public static List<RequestDto> GetAllRequestByUserId(int userId)

{

using (var db = new DBContext())

{

try

{

List<Request> userRequests = db.Requests.Where(request => request.is\_relevant && request.user\_id.Equals(userId)).ToList();

List<RequestDto> userRequestsDto = new List<RequestDto>();

for (int i = 0; i < userRequests.Count; i++)

{

userRequestsDto.Add(Mapper.Map<Request, RequestDto>(userRequests[i]));

}

return userRequestsDto;

}

catch (Exception ex)

{

return null;

}

}

}

public static void OnInitMapper()

{

MapperClass.OnInit();

}

public static List<RequestDto> GetAllRelevantRequests(EventLog eventLog)

{//צריך לבדוק את הפונקציה שוב

eventLog.WriteEntry("DL:In GetAllRelevantRequests.");

using (var db = new DBContext())

{

try

{

DateTime dateTime = DateTime.Now;

var relevantRequests = (from Request request in db.Requests

join v\_attached\_file f in db.v\_attached\_file on request.file\_id equals f.stream\_id

join v\_attached\_file r in db.v\_attached\_file on request.recording\_id equals r.stream\_id

join User user in db.Users on request.user\_id equals user.user\_id

where request.is\_relevant && (request.date\_from.Value.CompareTo(dateTime.Date) <= 0 && request.date\_to.Value.CompareTo(dateTime.Date) >= 0 &&

(request.frequency\_id == (int)FrequencyEnum.TEN\_MINUTE ||

(request.frequency\_id == (int)FrequencyEnum.HALF\_AN\_HOUR && (dateTime.Date.Minute == 0 || dateTime.Date.Minute == 30)) ||

(request.frequency\_id == (int)FrequencyEnum.HOUR && dateTime.Date.Minute == 0) ||

(request.frequency\_id == (int)FrequencyEnum.ONCE\_A\_DAY && request.hour == dateTime.Date.Hour && dateTime.Date.Minute == 0) ||

(request.frequency\_id == (int)FrequencyEnum.ONCE\_A\_WEEK && request.day == (int)dateTime.Date.DayOfWeek + 1 && request.hour == dateTime.Date.Hour && dateTime.Date.Minute == 0)))

select new { record = r.file\_stream, html = f.file\_stream, request, user.mail }).ToList();

List<RequestDto> relevantRequestsDto = new List<RequestDto>();

for (int i = 0; i < relevantRequests.Count; i++)

{

relevantRequestsDto.Add(Mapper.Map<Request, RequestDto>(relevantRequests[i].request));

relevantRequestsDto[i].file\_stream = relevantRequests[i].html;

relevantRequestsDto[i].recording\_stream = relevantRequests[i].record;

relevantRequestsDto[i].recording\_stream = Encoding.ASCII.GetBytes(Encrypt\_Decrypt.DecryptStringAES(Encoding.UTF8.GetString(relevantRequestsDto[i].recording\_stream), "onTime123"));

relevantRequestsDto[i].userMail = relevantRequests[i].mail;

}

return relevantRequestsDto;

}

catch (Exception ex)

{ eventLog.WriteEntry("in catch.");

eventLog.WriteEntry(ex.ToString());

return null;

}

}

}

public static RequestDto updateDetailsOfTimingRequest(RequestDto requestDto)

{

using (var db = new DBContext())

{

try

{

Request request = Mapper.Map<RequestDto, Request>(requestDto);

db.Requests.AddOrUpdate(request);

db.SaveChanges();

return requestDto;

}

catch (Exception ex)

{

return null;

};

}

}

public static Guid AttachedFile(byte[] stream, string end)

{

using (var db = new DBContext())

{

ObjectParameter file\_id = new ObjectParameter("file\_id", typeof(Guid));

try

{

db.attached\_file\_add(String.Format("{0:dMyyyyHHmmss}", DateTime.Now) + end, stream, file\_id);

return (Guid)file\_id.Value;

}

catch

{

return new Guid();

}

}

}

public static byte[] StreamFromFile(Guid fileId)

{

using (var db = new DBContext())

{

try

{

v\_attached\_file vFile = db.v\_attached\_file.Where(v => v.stream\_id == fileId).FirstOrDefault();

return Encoding.ASCII.GetBytes(Encrypt\_Decrypt.DecryptStringAES(Encoding.UTF8.GetString(vFile.file\_stream), "onTime123"));

}

catch (Exception ex)

{

return null;

}

}

}

public static RequestDto FillInDataRequest(RequestDto requestDto)

{

using (var db = new DBContext())

{

try

{

Request requestFromDB = db.Requests.Find(requestDto.request\_id);

requestDto.recording\_id = requestFromDB.recording\_id;

requestDto.recording\_stream= StreamFromFile(requestFromDB.recording\_id);

return requestDto;

}

catch (Exception ex)

{

return null;

}

}

}

public static RequestDto DeleteRequest(RequestDto requestDto)

{

using (var db = new DBContext())

{

try

{

Request request = Mapper.Map<RequestDto, Request>(requestDto);

db.Requests.AddOrUpdate(request);

db.SaveChanges();

return requestDto;

}

catch (Exception ex)

{

return null;

};

}

}

public static RequestDto AddNewRequest(RequestDto requestDto)

{

using (var db = new DBContext())

{

try

{//הצפנה

string a = Encoding.UTF8.GetString(requestDto.recording\_stream);

string b = Encrypt\_Decrypt.EncryptStringAES(Encoding.UTF8.GetString(requestDto.recording\_stream), "onTime123");

requestDto.recording\_stream = Encoding.ASCII.GetBytes(Encrypt\_Decrypt.EncryptStringAES(Encoding.UTF8.GetString(requestDto.recording\_stream), "onTime123"));

requestDto.file\_id= AttachedFile(requestDto.file\_stream, ".html");

requestDto.recording\_id = AttachedFile(requestDto.recording\_stream, ".iim");

var request = Mapper.Map<RequestDto, Request>(requestDto);

//למחיקה

//request.recording\_id = null;

db.Requests.Add(request);

db.SaveChanges();

return requestDto;

}

catch (Exception ex)

{

return null;

};

}

}

public static int AddNewRecording(RequestDto requestDto)

{

using (var db = new DBContext())

{

try

{

requestDto.recording\_stream = Encoding.ASCII.GetBytes(Encrypt\_Decrypt.EncryptStringAES(Encoding.UTF8.GetString(requestDto.recording\_stream), "onTime123"));

requestDto.recording\_id = AttachedFile(requestDto.recording\_stream, ".iim");

requestDto.frequency\_id = 1;

var request = Mapper.Map<RequestDto, Request>(requestDto);

//למחיקה

request.file\_id = null;

var req = db.Requests.Add(request);

db.SaveChanges();

return req.request\_id;

}

catch (Exception ex)

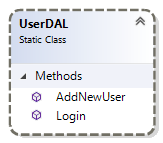
{

return -1;

};

}

}



UserDAL

מחלקה זו אחרית על התנהלות מול המסד נתונים בכל הנוגע לטיפול במשתמשים:

זיהוי משתמש קיים, הוספת משתמש חדש.

**Login** – פונקציה המקבלת כתובת מייל של משתמש וסיסמא ומחפשת אם הוא משתמש קיים במערכת, הפונקציה מחזירה את המשתמש.

מימוש:

public static UserDto Login(string userMail,string password)

{

using (var db = new DBContext())

{

try

{

// user identification by email and password

var user = db.Users.FirstOrDefault(user\_found =>user\_found.mail.Equals(userMail)&& user\_found.password.Equals(password));

if (user != null)

return Mapper.Map<User, UserDto>(user);

else

return null;

}

catch (Exception ex)

{

return null;

}

}

}**AddNewUser** – פונקציה המקבלת אובייקט מסוג userDto ומוסיפה אותו למאגר המשתמשים.

מימוש:

public static UserDto AddNewUser(UserDto userDto)

{

using (var db = new DBContext())

{

try

{

var user = Mapper.Map<UserDto, User>(userDto);

//add a new user to the user table

db.Users.Add(user);

db.SaveChanges();

return userDto;

}

catch (Exception ex)

{

return null;

}

}

}

Common

namespace Common

{

public class MapperClass

{

public static void OnInit()

{

Mapper.Initialize(cfg =>

{

cfg.CreateMap<User, UserDto>();

cfg.CreateMap<UserDto, User>();

cfg.CreateMap<Request, RequestDto>();

cfg.CreateMap<RequestDto, Request>();

});

}

}

public enum FrequencyEnum

{

FIVE\_MINUTE,

HALF\_AN\_HOUR,

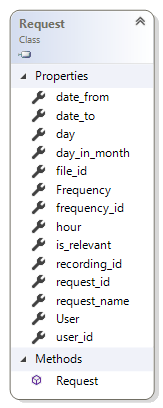
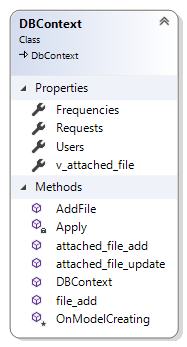
HOUR,

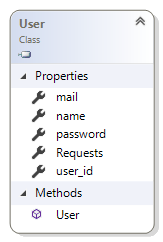
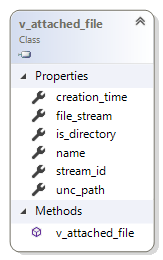
ONCE\_A\_DAY,

ONCE\_A\_WEEK,

}

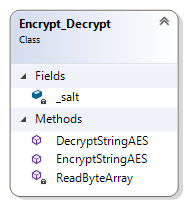
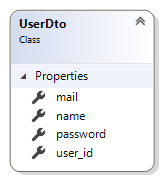
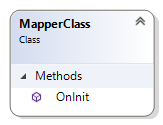
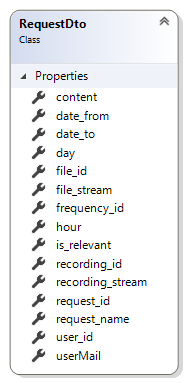
}



בנוסף לכל המחלקות הנ"ל, יש מחלקות המתארות את הטבלאות במסד נתונים. (Entity).

טבלה מסוג FileTable אינה מוצגת בEntity Framework הדרך לתקשר עם טבלת Files היא באמצעות .view

הוספת קובץ לטבלה נעשה באמצעות פרוצדורה.



**7. מתזמן - Windows service**

על מנת לבדוק האם התבצע שינוי בדפים ולבדוק את מצבם יש צורך בשרות שירוץ בתמידות בשרת ויבצע את הפעולות הבאות: ישלוף את הבקשות הצריכות בדיקה, יגלוש רובוטית לדפים, ישווה ובמידה והתרחש שינוי יעדכן את המשתמש.

ישנה אפשרות להשתמש בTask Scheduler של Windows אך אז לא תתאפשר בדיקה בתדירות גבוהה מפעם ביום.

ועל כן החלטנו ליצור Windows service בעצמנו אשר בו ניתן לקבוע את התדירות באופן הנדרש.

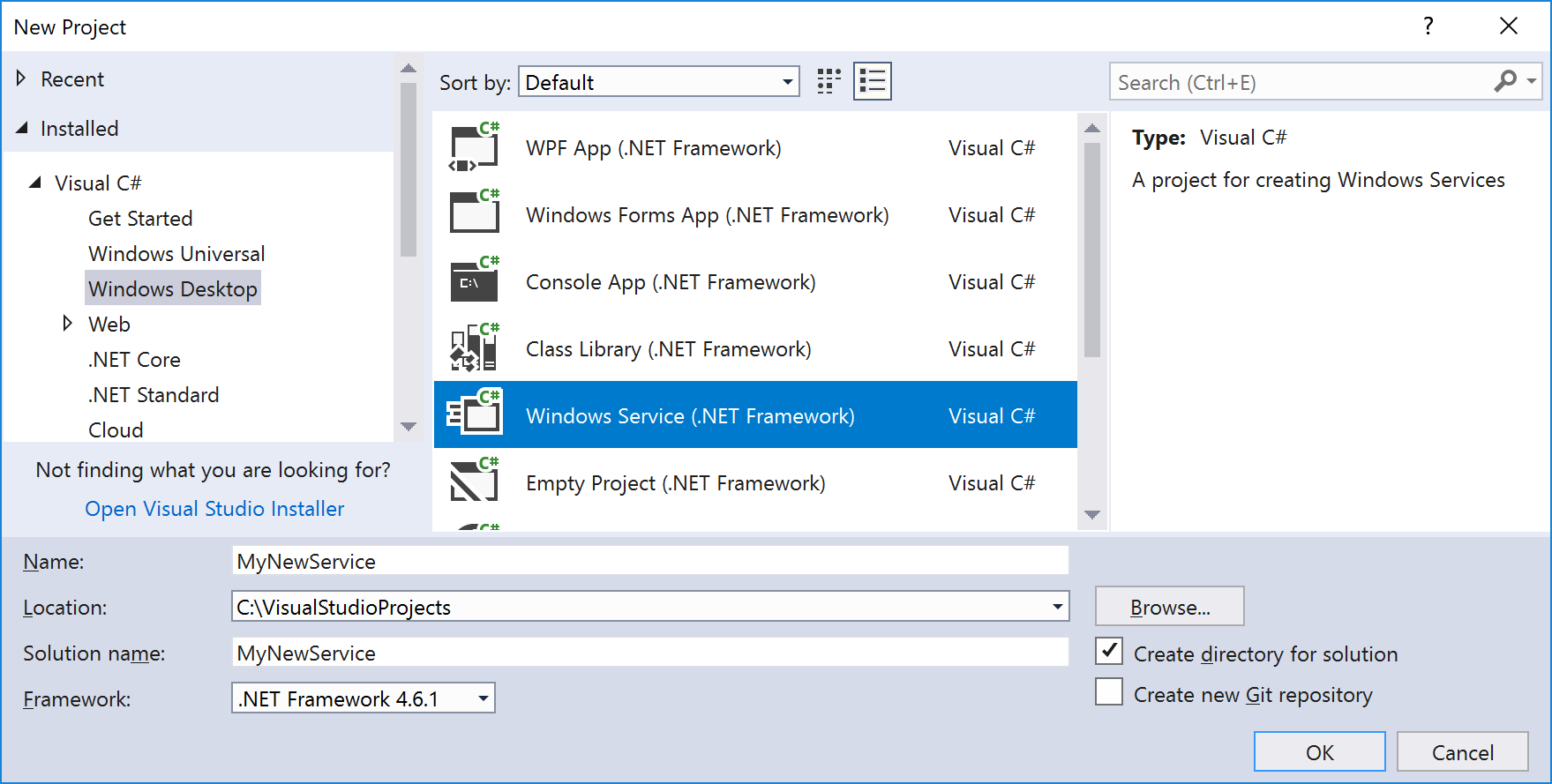
לשם כך נצרכה למידה על אופן כתיבת השרות, ההתקנה והמעקב.

לאחר אריזת השרות והתקנתו במחשב הוא רץ כsystem ומופעל ע"י מעה"פ ועל כן אופן המעקב אחר פעולותיו היא ע"י כתיבה לlog וקריאת המידע מהEvent Viewer.

צורת ההתחברות של השרות למסד הנתונים של sql היא כsystem ועל מנת לאפשר לו גישה ישנו צורך להגדיר בקונפיגורציה של הsql אפשרות גישה עםlogin ייחודי שאת פרטיו מגדירים בconnection string שבapp.config.

התזמון ופעולות המעקב מתבצעים בסבב של פעם עשר דקות.

1. יצירת פרויקט מסוג Windows service.



1. כתיבת הקוד:

הגדרת הclass אתחול הEventLog והגדרת הTimer.

public partial class MonitorService : ServiceBase

{

EventLog eventLog;

public static int eventId = 1;

private System.Threading.Timer timer = null;

public MonitorService()

{

InitializeComponent();

eventLog = new System.Diagnostics.EventLog();

// Create a file to write the log

if (!System.Diagnostics.EventLog.SourceExists("MonitorServiceSource"))

{

System.Diagnostics.EventLog.CreateEventSource(

"MonitorServiceSource", "MonitorServiceLog");

}

eventLog.Source = "MonitorServiceSource";

eventLog.Log = "MonitorServiceLog";

}

OnStart - פונקציה המתרחשת בעת התקנת ה service

protected override void OnStart(string[] args)

{

eventLog.WriteEntry("In OnStart.");

// set the time the service start and setting the time period for each round

TimeSpan scheduledRunTime = new TimeSpan(12, 53, 0), timeBetweenEachRun = new TimeSpan(0, 2, 0);

double current = DateTime.Now.TimeOfDay.TotalMilliseconds;

double scheduledTime = scheduledRunTime.TotalMilliseconds;

double intervalPeriod = timeBetweenEachRun.TotalMilliseconds;

// calculates the first execution of the method, either its today at the scheduled time or tomorrow (if scheduled time has already occurred today)

double firstExecution = current > scheduledTime ? intervalPeriod - (current - scheduledTime) : scheduledTime - current;

// create callback - this is the method that is called on every interval

TimerCallback callback = new TimerCallback(Monitoring);

// create timer

timer = new System.Threading.Timer(callback, null, Convert.ToInt32(firstExecution), Convert.ToInt32(intervalPeriod));

// initialization the mapper

RequestBL.OnInitMapper();

}

פונקציית הניטור המתבצעת בכל איטרציה:

// For each request Robotic browsing, pages comparison and inform on changes.

protected void Monitoring(object sender)

{

eventLog.WriteEntry("MonitorService:In GetAllRelevantRequests.");

// get all relevant requests

List<RequestDto> requestDtos = RequestBL.GetAllRelevantRequests(eventLog);

// for each request

foreach (var request in requestDtos)

{

try

{

// create thread and initialization the function

Thread thread = new Thread(RequestBL.ThreadFunction);

// run the function

thread.Start(request);

}

catch (Exception ex)

{

eventLog.WriteEntry(ex.ToString());

}

}

eventLog.WriteEntry("Monitoring the System", EventLogEntryType.Information, eventId++);

}

OnStop - פונקציה המתרחשת בעת הסרת ה service

protected override void OnStop()

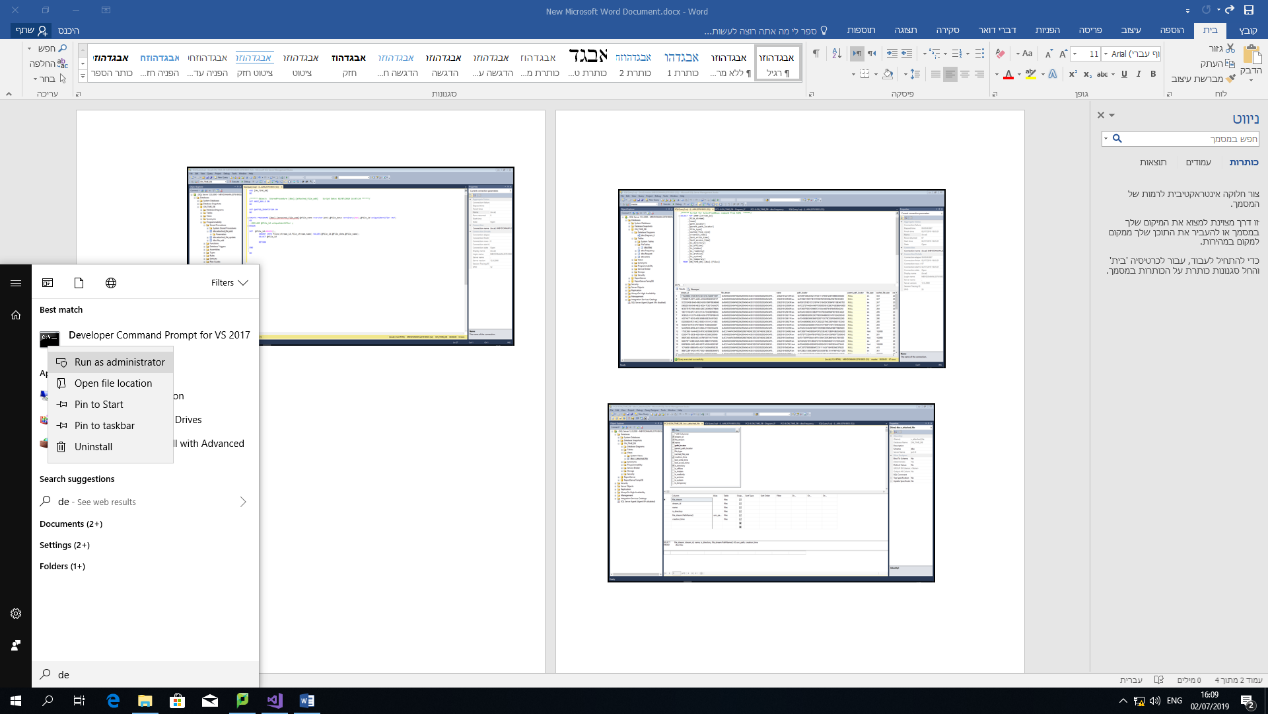
{

eventLog.WriteEntry("In OnStop.");

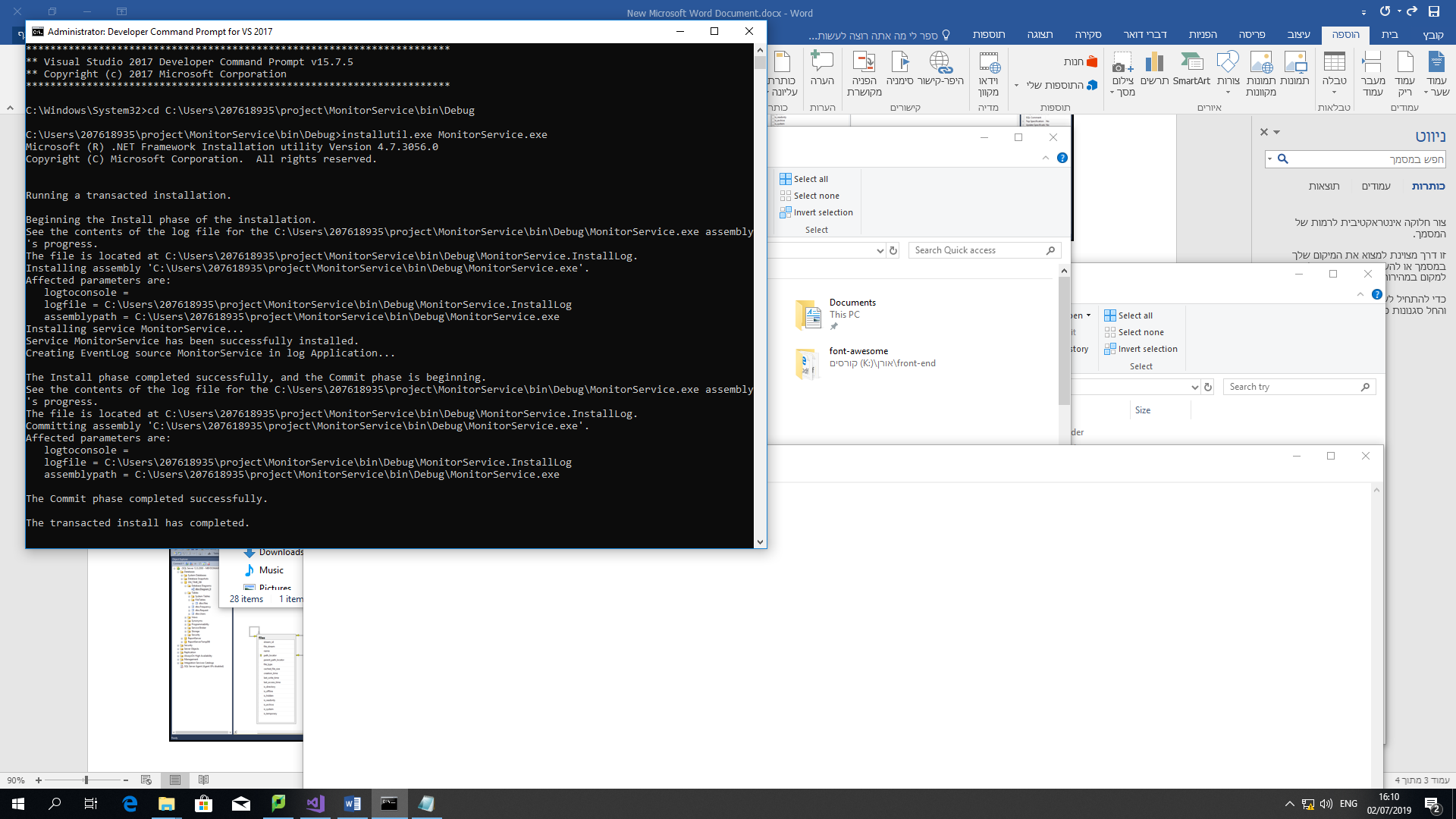
}

1. בניית הפרויקט ויצירת קובץ exe.
2. התקנת הפרויקט:

פתיחת חלונית פקודה בהרשאת Administrator



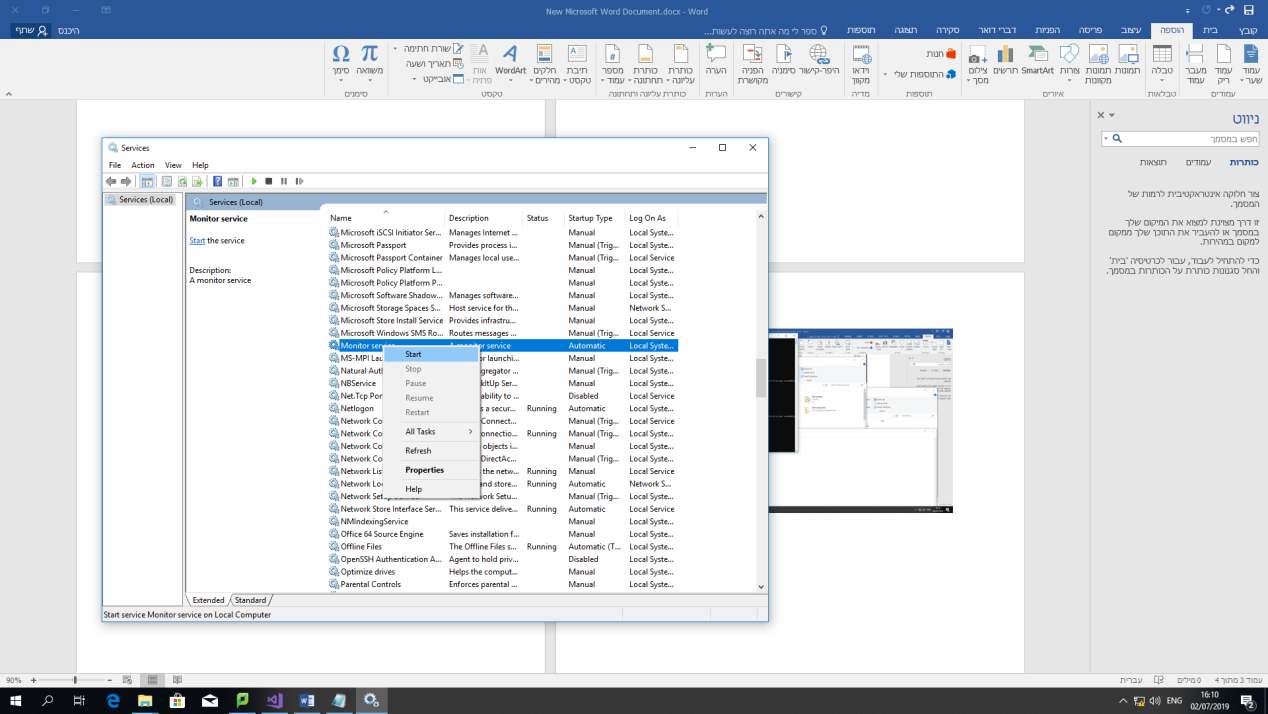
הרצת הפקודות הבאות: ניווט למיקום קובץ ההרצה ולהתקנתו.

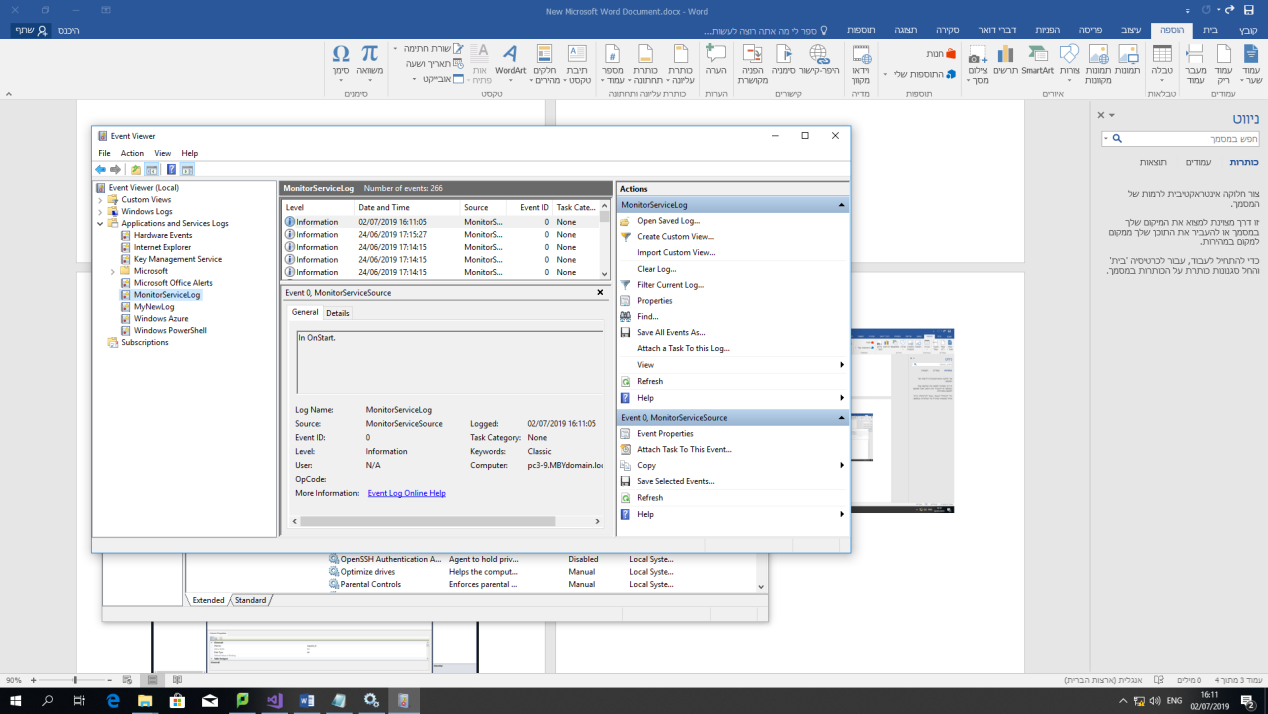


ניווט->

התקנה->

1. הפעלת ה serviceבחלונית ה.services



6. מעקב אחר הדפסות ה service ללוג בחלונית הEvent Viewer

7. גישה למסד הנתונים

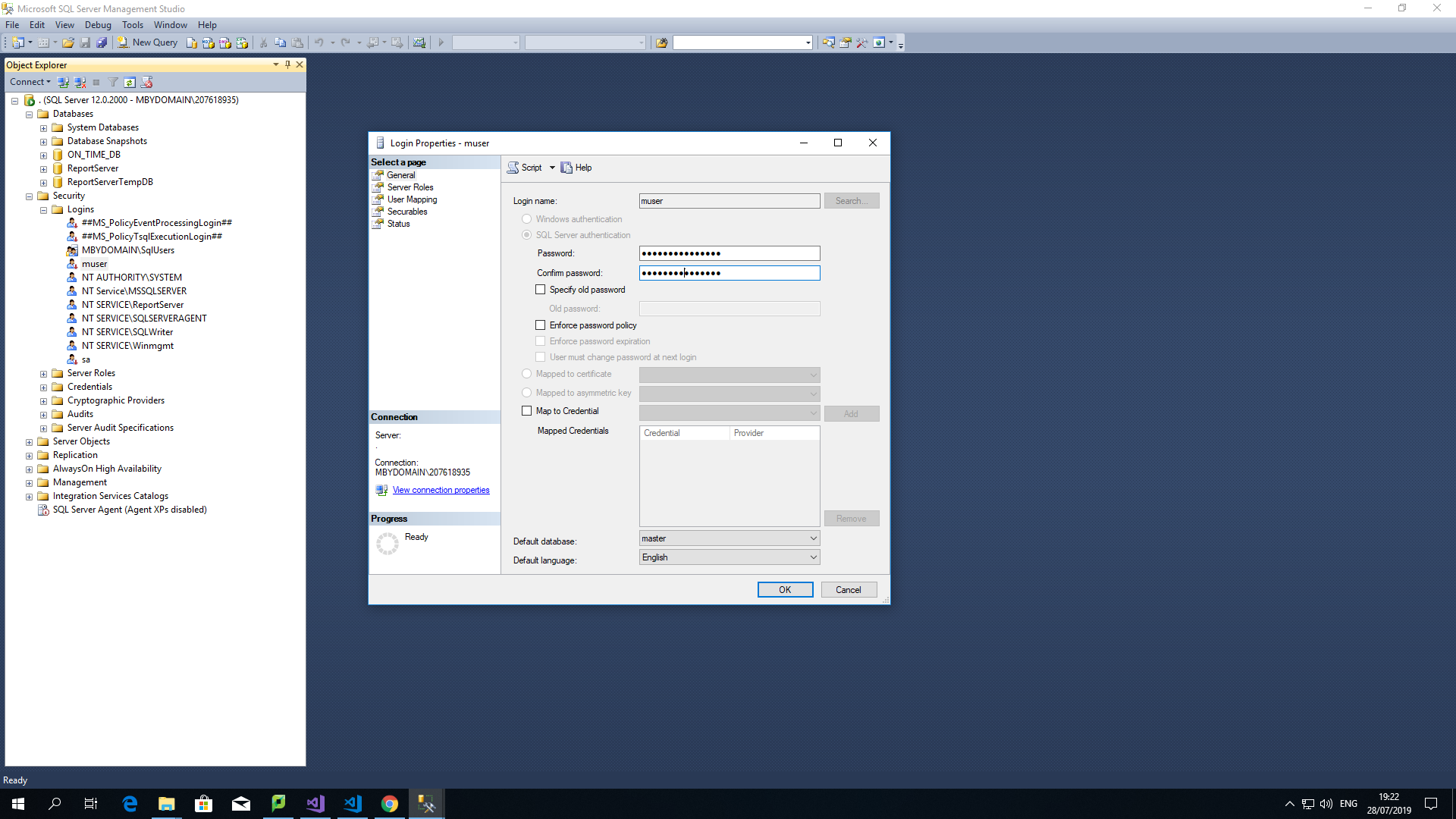
<connectionStrings>

<add name="DBContext" connectionString="metadata=res://\*/Model.csdl|res://\*/Model.ssdl|res://\*/Model.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=.;initial catalog=ON\_TIME\_DB;MultipleActiveResultSets=True; user id=muser;password=mu7414; App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

<add name="DefaultConnection"

connectionString="Data Source = |SQL/CE|" />

</connectionStrings>



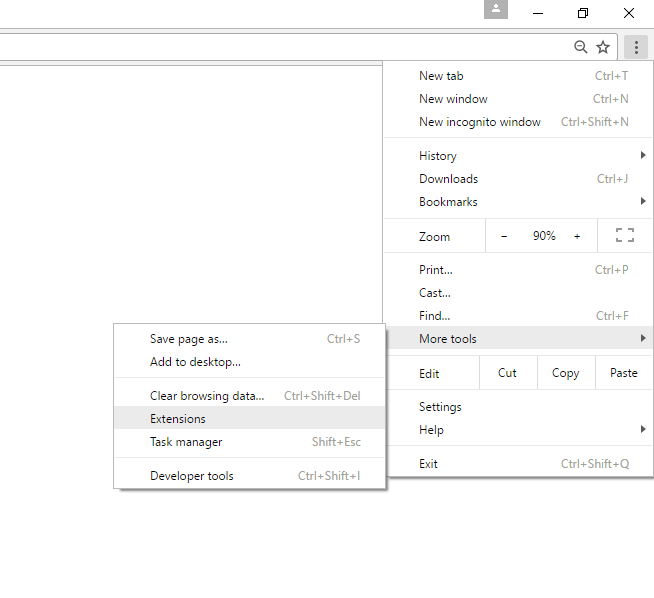
**8 ממשקי המערכת ומדריך למשתמש**

## 8.1 מדריך למשתמש באתר

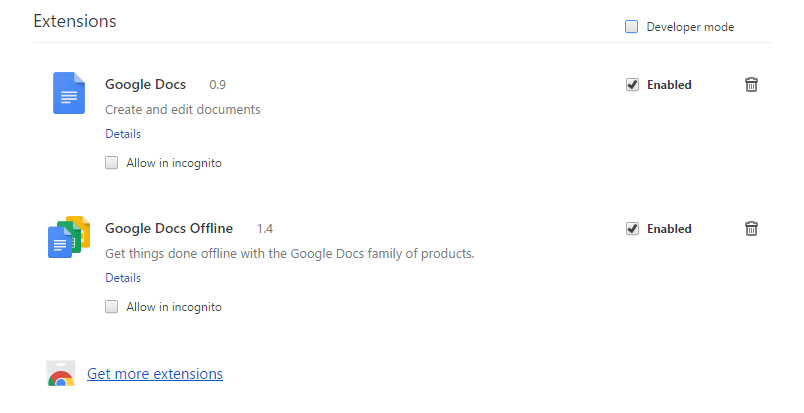
## 8.2 extension ל Google Chrome

### 8.2.1 מדריך התקנת extension ל Google Chrome

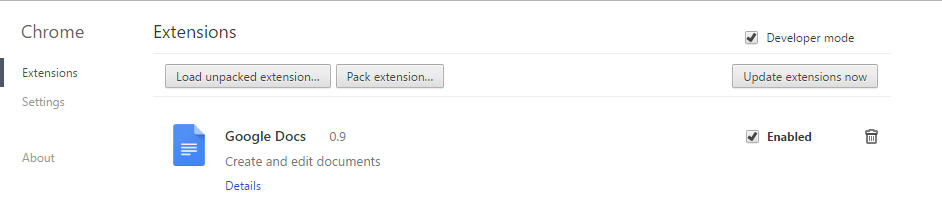
1. הכנס ל **chrome://extensions** בדפדפן שלך או בחר בפינה הימנית של הדפדפן ולבחור **Extensions**  תחת תפריט **Tools** .



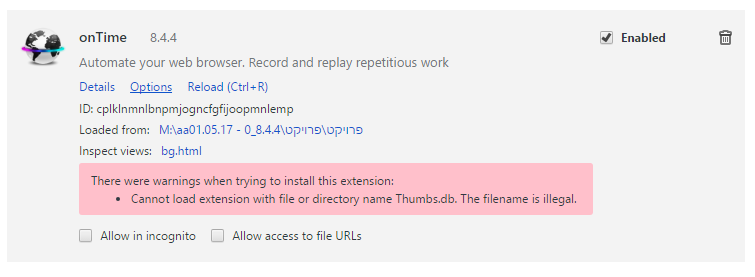
1. וודא כי Developer mode מסומן.



1. לחץ על **Load unpacked extension** ונתב לתיקיה בה התוסף שמור.



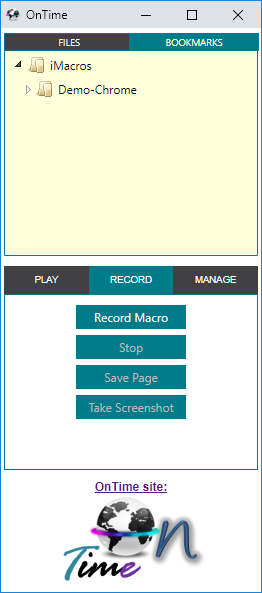
בשלב זה התוסף טעון ואמור להתווסף icon לדפדפן, אם אינו מופיע יש לוודא כי התוסף מאופשר.



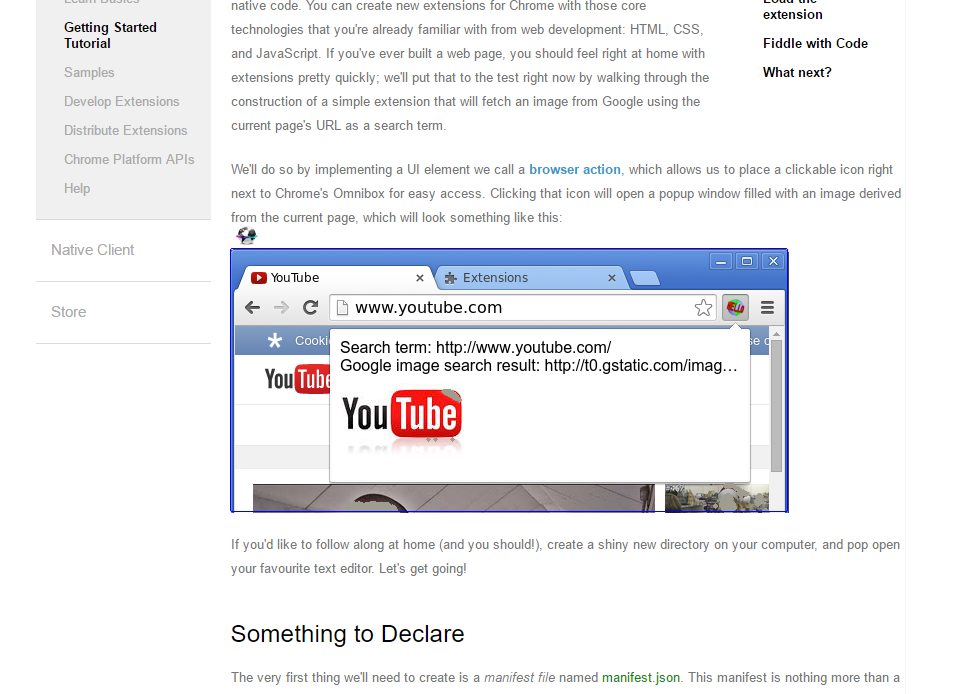
### 8.2.2 מדריך למשתמש בתוסף

הגישה לתוסף היא דרך icon  המופיע בחלק הימני של הדפדפן.

מסך ראשי



במהלך ההקלטה לאחר כל פעולה שמתבצעת בדפדפן מופיע icon המסמל כי הפעולה נקלטה.



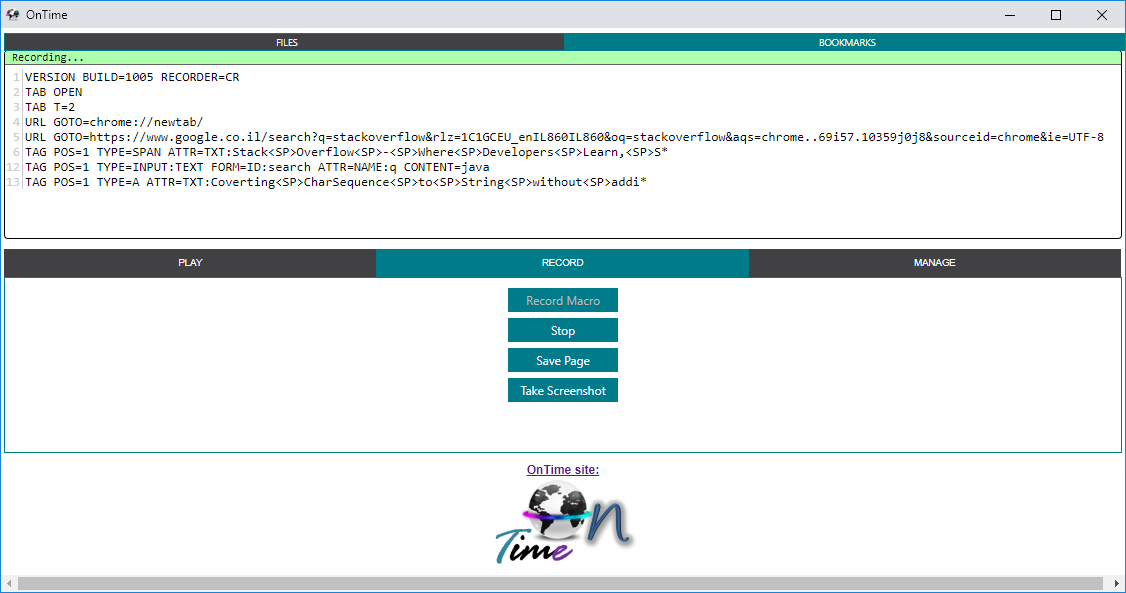
כאשר נמצא הדף המבוקש יש ללחוץ על Stop לסיום ההקלטה.

בסיום ההקלטה מוצגת ההקלטה להתרשמות ועריכה.



שמירה של ההקלטה- ההקלטה נשמרת server.

לאחר שההקלטה נשמרה, נפתח האתר וניתן להגדיר את התזמון שלה כך שיוכל להתבצע מעקב בתדירות נבחרת.



**בדיקות**

**גרסת מוצר: 1.1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | סוג בדיקה | מספר בדיקה | תאור קצר | עבר/נכשל | באגים |
|  | בדיקת שפיות | 1 | בדיקה של תקינות התהליך |  |  |
|  | בדיקת GUI – ממשק משתמש | 2 | בדיקת הזנת נתונים נכונה- ברישום ובבחירת התזמון |  |  |
|  | בדיקות פונקציונליות | 3 | בדיקה שהמערכת פועלת כמצופה ממנה |  |  |
|  | בדיקת שימושיות | 4 | בדיקת נוחות השימוש, חווית המשתמש ומידת הבהירות וההבנה |  |  |
|  | בדיקת ביצועים | 5 | ? |  |  |
|  | בדיקות עומס | 6 | ? |  |  |

**Test Case**- בדיקת שפיות

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | מס' שלב בבדיקה | step | תוצאה צפויה | תוצאה בפועל | עבר/נכשל | באגים |
|  | 1 | כניסה למערכת-משתמש קיים | זיהוי המשתמש ע"י מייל וסיסמא | המשתמש זוהה ונכנס |  |  |
|  | 2 | כניסה למערכת-משתמש חדש | הוספת משתמש ופרטיו: שם, מייל וסיסמא | המשתמש נוסף בהצלחה ונכנס |  |  |
|  | 3 | צפיה/הצגת כל הבקשות של המשתמש הנוכחי | הבקשות יוצגו | הבקשות הוצגו |  |  |
|  | 4 | הוספת בקשה חדשה | הבקשה תוסף | הבקשה נוספה |  |  |
|  | 5 | הקפצת הודעה מתאימה לאופן בו הגיע המשתמש לאתר | בהגעה ע"י התוסף- הודעה מסוימת  בהגעה דרך האתר- הודעה אחרת | ההודעות הוקפצו בהתאמה |  |  |

**Test Case**- בדיקת ממשק משתמש

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | מס' שלב בבדיקה | Step | תוצאה צפויה | תוצאה בפועל | עבר/נכשל | באגים |
|  | 1 | ולידציה על תקינות שדה מייל | יתאפשר הכנסה של מייל תקין בלבד | המייל נבדק ורק מייל תקין עבר |  |  |
|  | 2 | ולידציה על תקינות שדה URL | יתאפשר הכנסה של URL תקין בלבד | הURL נבדק ורק URL תקין עבר |  |  |
|  | 3 | ולידציה על מילוי שדות חובה | המערכת לא תמשיך עד שימולאו שדות החובה | השדות מולאו |  |  |
|  | 4 | חיווי למשתמש על סטטוס הזיהוי/ רישום | המשתמש יראה הודעה מתאימה לסטטוס שלו:  הוסף/זוהה בהצלחה או נכשל |  |  |  |
|  | 5 | הורדת התוסף | בלחיצה על DOWNLOAD ירד התוסף למחשב המשתמש | התוסף הורד |  |  |
|  | 6 | מעבר למסך הבקשות | לאחר ההזדהות המשתמש אוטומטית יועבר למסך הבקשות | המשתמש הועבר למסך הבקשות |  |  |
|  | 7 | שדה URL יוצג /יוסתר | שדה URL יוצג /יוסתר בהתאם לרצון המשתמש | שדה UR הותאם לרצון המשתמש |  |  |
|  | 8 | בחירת תזמון | לפי בחירת סוג התזמון יפתח INPUT מתאים | נפתח INPUT מתאים |  |  |
|  | 9 | אפשרות עריכה לבקשות קודמות | בלחיצה על האייקון פרטי הבקשה יהפכו למאופשרי עריכה | פרטי הבקשה הפכו למאופשרי עריכה |  |  |
|  | 10 | עדכון בקשה | בלחיצה על האייקון פרטי הבקשה יתעדכנו | פרטי הבקשה התעדכנו |  |  |
|  | 11 | מחיקת בקשה | בלחיצה על האייקון הבקשה תמחק | הבקשה נמחקה |  |  |

**Test Case**- בדיקת פונקציונליות

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | מס' שלב בבדיקה | step | תוצאה צפויה | תוצאה בפועל | עבר/נכשל | באגים |
|  | 1 | שליפת הבקשות של משתמש מסוים | שליפת בקשות לפי קוד משתמש | הבקשות נשלפו |  |  |
|  | 2 | שליפת כל הבקשות הרלוונטיות | שליפת כל הבקשות הרלוונטיות ע"פ הקריטריונים שנקבעו |  |  |  |
|  | 3 | גלישה רובוטית | הרצת קטע קוד – גלישה רובוטית | הגלישה בוצעה |  |  |
|  | 4 | השוואת דגימה חדשה לדגימה קודמת | השוואת תוכן HTML של שני הדפים והחזרה אם היה שינוי |  | ההשוואה בוצעה |  |
|  | 5 | שליחת מייל למשתמש | שליחת מייל למשתמש אם הדף השתנה/ התעדכן | נשלח מייל |  |  |

**Test Case**- בדיקת שימושיות

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | מס' שלב בבדיקה | step | תוצאה צפויה | תוצאה בפועל | עבר/נכשל | באגים |
|  | 1 | שליפת הבקשות של משתמש מסוים | שליפת בקשות לפי קוד משתמש | הבקשות נשלפו |  |  |
|  | 2 | שליפת כל הבקשות הרלוונטיות | שליפת כל הבקשות הרלוונטיות ע"פ הקריטריונים שנקבעו |  |  |  |
|  | 3 | השוואת דגימה חדשה לדגימה קודמת | השוואת תוכן HTML של שני הדפים והחזרה אם היה שינוי | ההשוואה בוצעה |  |  |
|  | 4 | גלישה רובוטית | הרצת קטע קוד – גלישה רובוטית | הגלישה בוצעה |  |  |
|  | 5 | שליחת מייל למשתמש | שליחת מייל למשתמש אם הדף השתנה/ התעדכן | נשלח מייל |  |  |

**טבלת באגים**.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | מס' באג | סוג בדיקה | מס' בדיקה | תאור הבאג | האם תוקן? |
|  | 33 | שפיות | 1 | אי אפשר לשנות רמה | כן |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**9 מסקנות**

השלב הבא אם נרצה לפתח זה לאפשר בחירה ומעקב של חלק דף.

לשם כך נצטרך להתגבר על מספר מכשולים (את חלקם כבר פתרנו)

הצגת הדף אליו הגיע המשתמש באתר שלנו על מנת לאפשר בחירת חלק- בעיה של Cross domain יש להקים proxy שיבצע את הגישה

ה SERVER ישמש בPROKXY יגלוש פעם ראשונה כCLIENT יקבל את הדף יחד עם הSOURCE יוסיף את הקוד שלנו וישלח את זה לתצוגה בIFRAME

**PIPE**

import { Pipe, PipeTransform} from '@angular/core'

import { DomSanitizer } from '@angular/platform-browser'

@Pipe({ name: 'safeHtml'})

export class SafeHtmlPipe implements PipeTransform {

constructor(private sanitized: DomSanitizer) {}

transform(value) {

return this.sanitized.bypassSecurityTrustHtml(value);

}

}

**השימוש בו**

זיהוי חלק הדף בפעם הראשונה

זיהוי חלק הדף בכל פעם בגלישה לשם השוואה

היסטוריה- אם נרצה בהמשך להוסיף אפשרות של צפייה בהיסטורית השינויים לשם כך הוספנו קבצים (ולא עדכנו את הקובץ הקיים ) (אולי לשלוח במייל 2 תמונות/דפים להשואה?

**10 ביבליוגרפיה**

[stackoverflow.com](http://stackoverflow.com/)

[github.com](https://github.com)

[www.codeproject.com](http://www.codeproject.com)

[wiki.imacros.net/iMacros\_for\_Chrome](http://wiki.imacros.net/iMacros_for_Chrome)

[www.seleniumhq.org](http://www.seleniumhq.org)

[https://docs.microsoft.com](https://docs.microsoft.com/)

Msdn. Microsoft.Com

[www.tutorialspoint.com](http://www.tutorialspoint.com)

[https://sweetalert2.github.io](https://sweetalert2.github.io/)

[csharp.net-informations.com](http://csharp.net-informations.com/communications/csharp-smtp-mail.htm)

[https://www.w3schools.com](https://www.w3schools.com/)

<https://angular.io/>

[https://www.primefaces.org](https://www.primefaces.org/)

https://auth0.com /

https://chrome.google.com

[https://www.c-sharpcorner.com](https://www.c-sharpcorner.com/)

[https://github.com](https://github.com/)

[https://translate.google.co.il](https://translate.google.co.il/)

[https://www.freecodecamp.org](https://www.freecodecamp.org/news/how-to-create-and-publish-a-chrome-extension-in-20-minutes-6dc8395d7153/)

<https://youtube>

