# בס"ד



המכון הממשלתי להכשרה בטכנולוגיה ומדע



# **מוגש ע"י**: אסתר קליין תמר הכהן עדני מגמת הנדסת תוכנה **מנחה**: מ.שמונוביץ



המכון הממשלתי להכשרה בטכנולוגיה ומדע



# **שמות המגישות**: אסתר קליין תמר הכהן עדני **חתימה**:\_\_\_\_\_\_\_\_\_\_ **חתימה**:\_\_\_\_\_\_\_\_\_\_ מגמת הנדסת תוכנה **מנחה**: מ.שמונוביץ **חתימה:**\_\_\_\_\_\_\_\_\_\_ **מרכזת המגמה**: ח.ברגמן **חתימה**:\_\_\_\_\_\_\_\_\_\_

# הבעת תודה

ראשית תודה לבורא העולם.

תודה לרכזת המגמה בסמינר הרב וולף – המו' חנה ברגמן, שפועלת רבות כדי לאפשר לנו תנאים מיטביים ומתאימים.

כמו כן, תודה למנחה על הציוד בעקרונות ובכלים הטובים ביותר לפיתוח תוכנה שלוונו במהלך כתיבת הפרויקט, ועל הסיוע בתכנון הפרויקט בצורה המוצלחת ביותר.

תודה למזכירת המוסד ג. פישר על שיתוף הפעולה, ההכרות עם פעילות המוסד וצרכיו לפרטי פרטים.

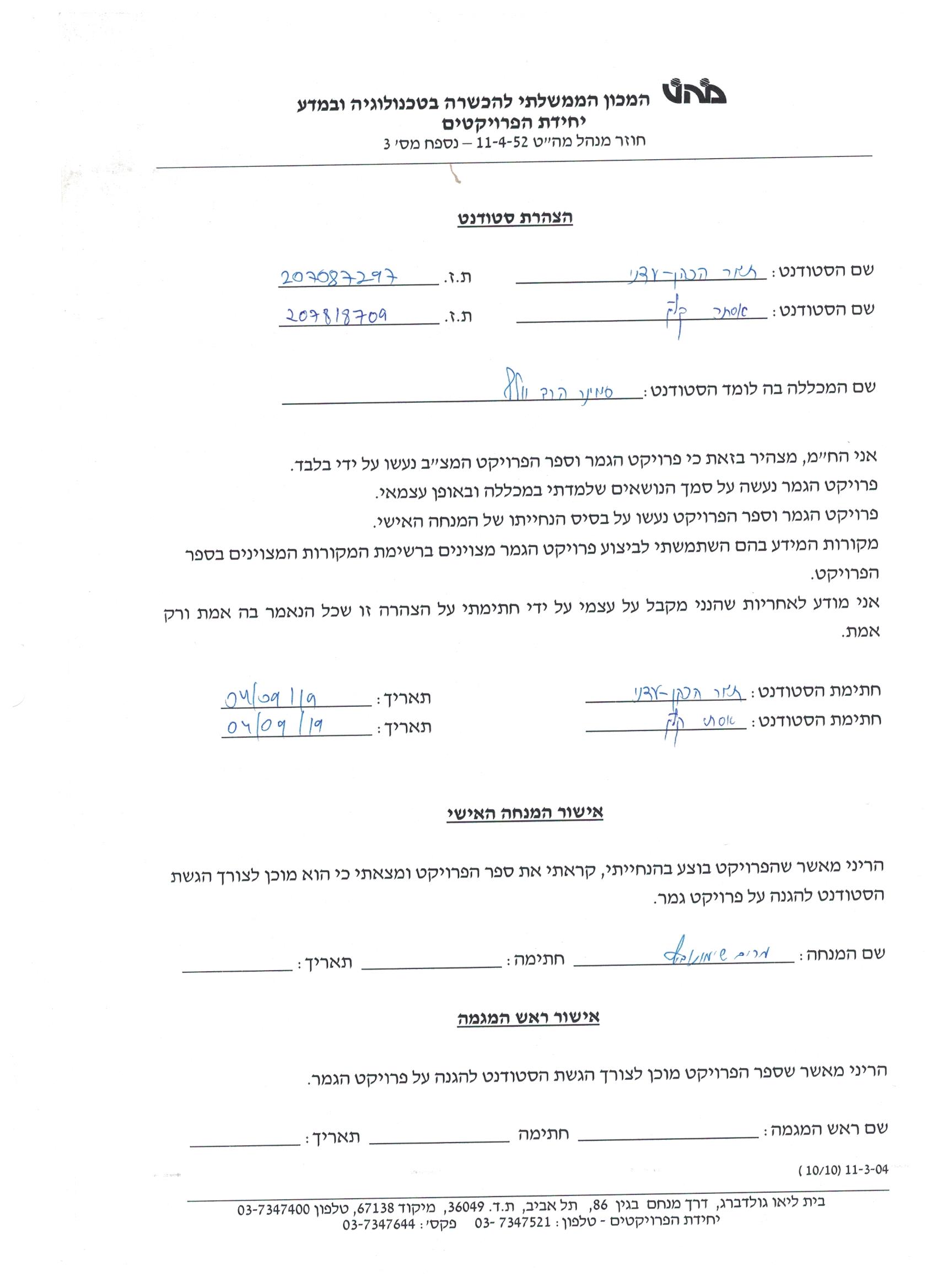
תודה לאחראיות חדרי המחשבים על סיוע טכני בחפץ לב, ושהיו בעת הצורך.

תודה לחברות לספסל הלימודים על העצות הקטנות והרעיונות היעילים שנתנו בין לבין, שתמכו ושתפו.

ולכל מי שליווה תמך ועזר מסביב, אם ישירות ואם בעקיפין.

ולסיום גם אם נמלא כאן דפים מלאי תודות למי שהכל שלו והכל ממנו, אף פעם לא נגיע לקצה גודל התודה האמיתית.

# הצהרה



# תוכן עניינים

[הבעת תודה 3](#_Toc18678806)

[הצהרה 4](#_Toc18678807)

[תוכן עניינים 5](#_Toc18678808)

[תקציר 6](#_Toc18678809)

[מבוא 7](#_Toc18678810)

[תיאור הפרויקט 8](#_Toc18678811)

[אסטרטגיות וטכנולוגיות 8](#_Toc18678812)

[שפות 8](#_Toc18678813)

[טכנולוגיות 9](#_Toc18678814)

[סביבות עבודה 10](#_Toc18678815)

[**תיאור מבנה הפרויקט** 12](#_Toc18678816)

[**עקרונות התכנון הבניה והניתוח** 14](#_Toc18678817)

[הפרדת  שכבות 14](#_Toc18678818)

[OOP 14](#_Toc18678819)

[Binding 14](#_Toc18678820)

[XML DOM 14](#_Toc18678821)

[LINQ to SQL 14](#_Toc18678822)

[Singleton 15](#_Toc18678823)

[תרשימים 16](#_Toc18678824)

[תרשים מראה המחלקות 16](#_Toc18678825)

[מבנה  נתונים 36](#_Toc18678826)

[תיאור מבנה קבצי XML 39](#_Toc18678827)

[תוכן הפרויקט 42](#_Toc18678828)

[תיאור המחלקות 42](#_Toc18678829)

[פונקציות מעניינות 65](#_Toc18678830)

[מדריך למשתמש 71](#_Toc18678831)

[סיכום ומסקנות 82](#_Toc18678832)

[נספחים 83](#_Toc18678833)

[מנגנון אתחול מערכת 83](#_Toc18678834)

[מנגנון טיפול בשגיאות 88](#_Toc18678835)

[XML Classes 88](#_Toc18678836)

[ביבליוגרפיה 93](#_Toc18678837)

# תקציר

המערכת שלנו פותחה ותוכננה עבור מוסד לימודים גדול. היא מאפשרת לנהל היעדרויות של מורות, לשבץ מורות ממלאות מקום. לנהל שעורים- להוסיף שעור במקרה הצורך, להוריד שעור או לעדכן שעור, כלומר לשנות את מקצוע השיעור או את המורה המלמדת. לנהל אירועים-לשבץ אירועים במערכת, כמו מבחן, טיול, הרצאה, מחנה וכדו'.

יש בה 2 הרשאות :  
הרשאת מזכירה- בהכנסת סיסמה בפתיחת המערכת, יכולה לבצע שינויים במערכת לעיל, לבצע חיסורי מורות ולצפות בהם, לערוך אירועים.    
הרשאת אורח- כל משתמש אחר יכול לצפות במערכת ובאירועים השונים.

# מבוא

בניהול מוסד לימודים ישנם תחומים רבים, התחום העיקרי הוא ניהול מערכת השעות.  
כאשר מורה נעדרת יש צורך לטפל בשעורים שבהן היא מלמדת, לקבוע ממלאת/ות מקום או לבצע הזזת שעורים במקרה הצורך, כמו כן להוסיף לה היעדרות.  
כאשר יש צורך לאחד כיתות, לשחרר כיתות וכדו' יש לבצע את הפעולות הנדרשות.  
במקרה של אירוע, ז"א אין שעור/ים סדיר/ים (כמו הרצאה, טיול או מבחן) יש לדאוג לפרטים המתאימים.  
כאשר רוצים לצפות במערכת השעות, או לאפשר למורות/תלמידות צפייה במערכת המתאימה להן, יש לאפשר להן.

בבנייתנו את המערכת חקרנו את המערכות הנמצאות בשימוש כיום, למדנו את הצרכים והדרישות. מערכת זו פותחה ועוצבה כדי לעמוד בדרישות, להיות נוחה, יעילה וזמינה.

כמו שנכתב לעיל המערכת מאפשרת:   
- להוסיף חיסור למורה בתאריך המתאים, וכן לשבץ מורה ממלאת מקום מתוך רשימת המורות.  
- להוסיף שיעור, להוריד שיעור או לעדכן שיעור בפרטים הרלוונטיים.  
- להוסיף אירוע, להוריד אירוע או לעדכן אירוע בפרטים הרלוונטיים.  
- לצפות במערכת המתאימה, לפי כיתות, מורות או אירועים.

ההרשאות הקימות במערכת:  
הרשאת מנהל - המזכירה האחראית יכולה לנהל את השינויים והפרטים שקורים כאשר מורה חסרה. לשנות את המערכת לפי שינויים זמניים, לנהל את השיעורים, לנהל את האירועים.     
הרשאת אורח - כל אחד אחר יכול לצפות במערכת השעות העדכנית, לצפות באירועים שנקבעו וליצור קשר עם הצוות הטכני - צוות התמיכה.

המערכת חיבת כקלט קבצי xml תקניים שנבנו במערכת התואמת -תלמ"ש, המכילים נתונים בסיסיים למערכת השעות. בנתונים אלו כלולים מבנה מוסד הלימודים, הכיתות הקימות בו, המקצועות הנלמדים בו והמורות המלמדות בו.

בתקווה שהבנתם ונהניתם

צוות   School Schedule

# תיאור הפרויקט

## אסטרטגיות וטכנולוגיות

### שפות

בכתיבת הפרויקט השתמשנו בשפות הבאות:

• #C :

שפת #C היא שפת תכנות שפותחה ע"י מיקרוסופט ונחשבת לאחת

משפות התכנות הפופולריות בעולם. התחביר והעקרונות שלה הם עשירים ביכולות ועם זאת פשוטים ונוחים לעבודה.

היא מיועדת לפיתוח כללי של מגוון אפליקציות בכל התחומים כמו משחקים, אתרי Web, אפליקציות למכשירי מובייל וטאבלטים ועוד.

• HTML:

שפת HTML היא כלי פיתוח בסיסי להצגת מידע סטאטי בדפדפני האינטרנט הנפוצים. השיטה בה עובדת שפה זו היא הורדת המידע מהשרת בו מתארח האתר אל מחשב המשתמש.

שפת HTML היא השפה הבסיסית המרכיבה את מבנה ותוכן האתר. כאשר הדפדפן שלנו דוגמת chrome או explorer מעוניין להעלות את האתר הוא למעשה קורא את שפת ה-HTML ועל פיה יודע מה להציג לנו.

שפת ה-HTML היא שפת תגיות, מתקשרת עם שפת ה-CSS שהיא שפת העיצוב.

• : TypeScript

Typescript הינה שפת תכנות חינמית ובקוד פתוח, שפותחה במשך כשנתיים ע"י Microsoft וראתה אור לראשונה בשנת 2012.

היא מאוחסנת ב – GitHub ומטרתה להתמודד עם חסרונות קיימים ב – JavaScript ע"י הוספת סוגים, מחלקות, מודולים ועוד.

TypeScript הינה הרחבה ל – JavaScript והיא מכילה את כל הפקודות והתחביר של שפת המקור.

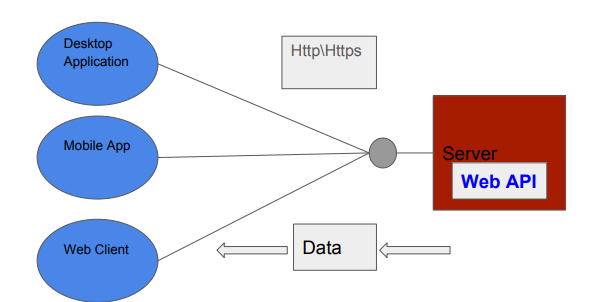
• : CSS

גיליונות סגנון מדורגים הם פורמט לעיצוב דפי אינטרנט. הגיליונות קובעים את עיצובם של תגים ב-HTML, ‏ XHTML וכל שפה דומה ל-XML לבניית אתרי אינטרנט. CSS נוצר במטרה להפריד בין תוכן ומבנה דפי האינטרנט לבין עיצובם.

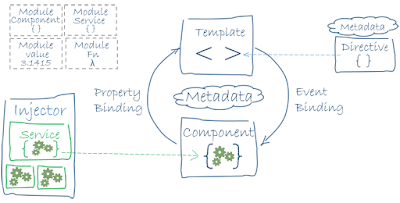
### טכנולוגיות

השתמשנו בטכנולוגיות הבאות לכתיבת הפרויקט:

* Entity Framework-צד שרת:  
   הספרייה Entity Framework דואגת למיפוי של אובייקטי המודל אל תוך מסד הנתונים באופן כמעט אוטומטי שאינו דורש כתיבת קוד. מיקרוסופט פיתחה ORM (Object Relational Mapping) אשר חוסך את כל העבודה המונוטונית הנדרשת לפיתוח שכבת מידע.  
   Entity Framework הוא כלי אשר, אוטומטית, ממפה טבלאות של מסדי נתונים שטוחים למחלקות, חוסך את הצורך בכתיבת קוד רב ומסורבל, ומאפשר למפתחים להתמקד ב-Business Logic של המערכת. Entity Frameworkהיא מערכת טכנולוגיות ב- ADO.NET התומכות בפיתוח יישומי תוכנה מוכווני נתונים. Entity Framework מאפשרת למפתחים לעבוד עם נתונים בצורה של אובייקטים ותכונות ספציפיים לתחום, כגון לקוחות וכתובות לקוחות, מבלי שהם צריכים לדאוג לעצמם לטבלאות ועמודות בסיס הנתונים בהן נשמרים נתונים. באמצעות Entity Framework, מפתחים יכולים לעבוד ברמה גבוהה יותר של הפשטה כאשר הם מתמודדים עם נתונים, ויכולים ליצור ולתחזק יישומים מונחי נתונים עם פחות קוד מאשר ביישומים בעלי טכנולוגיה ישנה יותר.
* Web Api-צד שרת:  
   Web Api הוא API אותו מממש השרת והוא מחזיר לקליינט Data בלבד ולא .View הבקשה והתשובה הן בפרוטוקול Https\Http הקליינטים של Web Api יכולים להיות רבים.



* Angular -צד לקוח:  
   Angular זה Framework לבניית אפליקציות תוך שימוש ב-HTML ו-JavaScript או TypeScript. Angular זה פרויקט קוד פתוח שמפותח ומתוחזק ע"י גוגל וקהילה רחבה של מפתחים. כתיבת אפליקציית Angular מתבצעת ע"י כתיבת HTML שבתוכו משולבים תוספות של Angular, כתיבת components כדי לשלוט על ה-templates, כתיבה של services כדי לשלוט על הלוגיקה, ושילוב של components ו- services ליצירת modules. הארכיטקטורה של Angular:



### סביבות עבודה

* Visual Studio 2017  
  Visual Studio היא סביבת פיתוח מובילה מבית חברת מיקרוסופט, המאפשרת למתכנתים לפתח תוכניות מחשב ואתרי אינטרנט אינטראקטיביים.  
  Visual Studio .NET מאפשרת פיתוח, מתוך תוכנה אחת וסביבה אחת, אפליקציות מסוגים שונים (חלונאיות, WEB, פרויקטי התקנה ועוד) ובשפות שונות הנתמכות ב-NET. ולמעבדים שונים (32 ו-64 סיביות) ואף להתקנים ניידים כמו מחשבי כף יד וטלפונים סלולריים.  
  הרכיב העיקרי של הסביבה הוא עורך טקסט שצובע את המילים לפי ההקשר התכנותי שלהם. לעורך תכונת "השלמה אוטומטית" המציגה חלונית עם שמות העצמים הקשורים לכתיבה ומאפשרת כתיבה מהירה ומניעת טעויות ושגיאות.  
  בין היתר מוטמעים כלי פיתוח חזותיים שונים שמחוללים קוד בצורה אוטומטית, כמו מעצבים חזותיים לממשקי משתמש Winform ו-WPF, וכלים שמאפשרים נגישות והתמצאות במרחבי הפרויקט כמו סייר הפרויקט לעיון בקובצי הפרויקט, סייר בסיסי הנתונים לעבודה עם בסיסי נתונים. כלי ניפוי שגיאות שונים שמאפשרים עצירה בנקודות מסוימות ועיון בערכי המשתנים.
* Visual Code  
  Visual Studio Code הוא שמה של פלטפורמה מבוססת קוד פתוח לעריכת קוד ופיתוח תוכנה הפועלת על מערכות ההפעלה Windows, לינוקס ו-OS X. העורך תומך בניפוי שגיאות, בקרת תצורה של גיט, המחשה סינטקטית של קטעי קוד, השלמת קוד חכמה (intelligent code completion), קטעי קוד אוטומטיים (snippets), ושינויי קוד רוחביים (code refactoring). ויז'ואל סטודיו קוד היא עורך קוד. ויז'ואל סטודיו קוד תומכת במספר שפות תכנות.
* Microsoft SQL Server Management Studio 17  
  Microsoft SQL Server היא מערכת לניהול בסיס נתונים במודל היחסי (RDBMS) של חברת מיקרוסופט. שפת הפיתוח שבאמצעותה מועברות הפקודות למערכת (על ידי כתיבת פקודות או שימוש בממשק גרפי ששולח פקודות באופן סמוי) היא Transact-SQL ‏(TSQL) שהיא מימוש תקן ANSI של שפת SQL משמשת לתשאול וטיפול בנתונים (DML), יצירת טבלאות והיחסים ביניהן (DDL) ותחזוקת המערכת תוך שימוש בתוכניות שירות שונות.

## **תיאור מבנה הפרויקט**

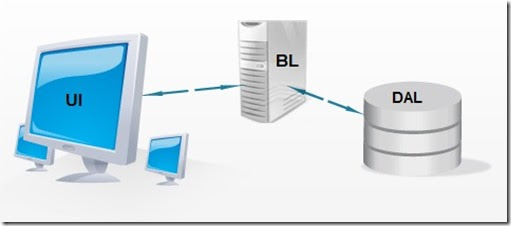
**חלוקה לשכבות** היא תבנית עיצוב בסיסית שמגדירה הפרדת האפליקציה ל שכבת נתונים, שכבת לוגיקה, ושכבת ממשק משתמש.

(כאמור בהגדרה הבסיסית, ככל שהתכנה מורכבת יותר ייתכנו שכבות נוספות)

למה זה טוב?

* תחזוקה: ניתן להחליף או לתקן מימוש פנימי של שכבה אחת בארכיטקטורה בלי לשנות שכבה אחרת.
* נוחות פיתוח: אדם אחד עובד על רכיב בתכנה, אדם אחר עובד על רכיב אחר, כל עוד שהחתימות זהות ניתן לשלב כוחות ולייעל זמני פיתוח.
* בדיקות: תקלה כלשהיא מבודדת בכל שכבה בנפרד, לדוגמא אם לא קיבלנו רשימת נתונים לתצוגה נבדוק קודם את שלב הביניים (שכבת הBL) אם הנתונים שם תקינים נדע בוודאות שגם ברמת שכבת הDAL הנתונים תקינים וכל שנותר הוא לפתור את התקלה ברמת הUI
* שימוש חוזר: נניח שהרכיבים BL + DAL אהובים עלינו ועובדים היטב, ניתן להחליף את פלטפורמת הUI לטכנולוגיה אחרת במינימום מאמץ.
* אבטחה: נוח יותר להגדיר Interface-API לשכבה מסוימת בלי לחשוף מבני נתונים, לוגיקה או נתונים ומידע שלא רלוונטיים למשתמש, כמו גם חסימה בפני האקרים (רלוונטי יותר בטכנולוגיות WEB) ברמות שונות.

ניתן להגדיר רמות שונות של חלוקה לשכבות ולקבל מבנה יעיל יותר; נקבל מבנה שנראה בערך כך:

[](http://blogs.microsoft.co.il/blogs/uriel99/3T_5B0030EA.jpg)

* API  
  עם שכבה זו מתקשר ה Client, כאן קימות מתודות הדואגות לקבל את הבקשות, לתקשר עם השכבות הנוספות ולהשיב תגובות.
* BL  
  השכבה שאמונה על הלוגיקה של המערכת, עוסקת בעיבוד המידע, בחישובים שונים ושליחתו לשכבת הAPI-.  
  בשכבה זו מימשנו את הפונקציונאליות של המערכת.
* DAL  
  שכבה זו מורכבת מקישור למסד הנתונים בטכנולוגית Entity Framework, וממערכת תוכנה אשר תפקידה לקרוא את המידע הנדרש למערכת, לשמור את העדכונים, ולהוסיף מידע חדש או למחוק פרטי מידע קיימים (נקרא גם CRUD- Create, Read, Update, Delete).
* DTO  
  שכבה זו מכילה טיפוסי נתונים (מחלקות) המקבילים לטיפוסים בשכבת ה DAL, ללא שדות הניווט וכו', ובנוסף מתודות הממירות את טיפוסי ה DAL לטיפוסי ה DTO ולהיפך.
* XmlFilesLoadingImplemention  
  שכבה זו קוראת את קבצי הנתונים לפני טעינתם למסד הנתונים, מבצעת את הפעולות המתאימות עד להכנסתם.

## **עקרונות התכנון הבניה והניתוח**

### הפרדת  שכבות

כפי שהרחבנו בתיאור הפרויקט הפרדנו את הפרויקט לשכבות, ובכל שכבה טפלנו בחלקי הקוד המתאימים.

### OOP

היא פיתוח מתודה בעלת שם מתודה ורמת גישה זהה למתודה קיימת אך עם פרמטרים שונים. כך ניתן לכתוב למשל כמה מתודות עם Overloading אותו שם ורמת גישה, אך שמקבלות פרמטרים שונים ומבצעות דברים שונים (ולרוב גם קיימים קשרים בין המתודות השונות).  
ניתן לראות בקלות כי השתמשנו בפונקציות מועמסות בפרויקט, לדוגמה Get(int id) המחזירה אובייקט לפי ID מתאים, וGet() המחזירה את אוסף האובייקטים הקיים.

### Binding

בצד לקוח השתמשנו בAngular , רכיב חשוב בטכנולוגיה זו הואngModel אשר מאפשר Two Way Binding -קישור דו כיווני. קשירה דו-כיוונית נותנת לנו לעדכן את המידע המוצג למשתמש בקובץ התבנית וגם את ערכה של התכונה בקומפוננטה בו-זמנית

### XML DOM

DOM (Document Object Module) מרכיב את קובץ ה-XML בזיכרון המחשב מאובייקטים. בצורה כזו המידע הרבה יותר זמין, דרך הפעולה הזו יותר קלה ואינטואיטיבית והקוד נהיה אף מובן יותר. הנתונים למערכת נטענים באמצעות קבצי xml, בתבנית מסוימת, בטעינת הקבצים אנו אוספים את הנתונים ושופכים לתוך מסד הנתונים.

### LINQ to SQL

**Linq** - ראשי תיבות של Language Integrated Query, מספקת לנו מודל ותחביר אחידים לטיפול במידע ללא קשר למקור הנתונים ממנו הוא מגיע, היא מאפשרת להגדיר שאילתות כחלק אינטגראלי משפת התכנות בה אנו משתמשים. Linq זה כינוי לשפת שאילתות המובנית כחלק אינטגרלי משפת #C. Linq היא שפה שלא רק שהתחביר שלה מאוד דומה לשפת SQL אלא בעיקר המשמעות שלה, היא שפת שאילתות המאפשרת לבצע שאילתות על אוספים. ב-Linq ניתן להשתמש בשתי דרכים: הראשונה בתחביר ייעודי שהפך להיות חלק מתחביר השפה והשניה היא אוסף מתודות ייעודיות העושות שימוש בביטויי למבדה תוך שימוש ב- Func<T> delegate ומסתירות את התחביר הבסיסי ובכך להפחית את כמות הקוד הנדרש למספר שורות בודדות.

Linq מאפשרת ביצוע שאילתות חיפוש, סינון, סידור, צירוף, איחוד לקבוצות של נתונים וכו'.

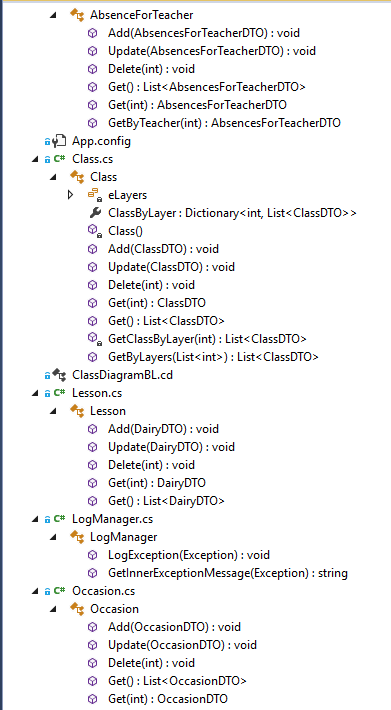
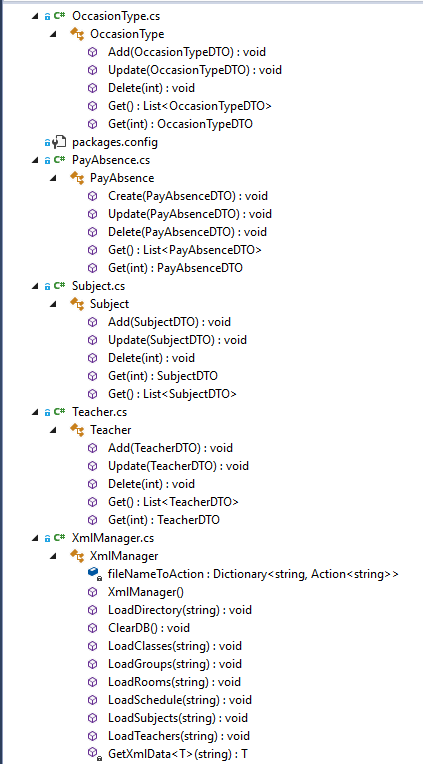
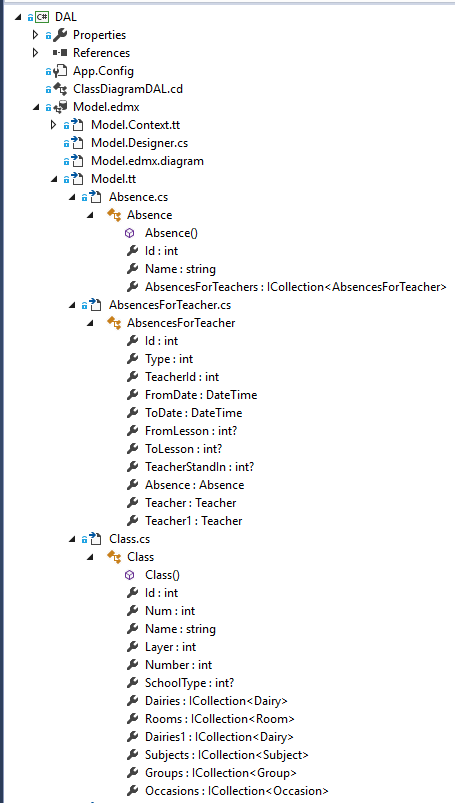
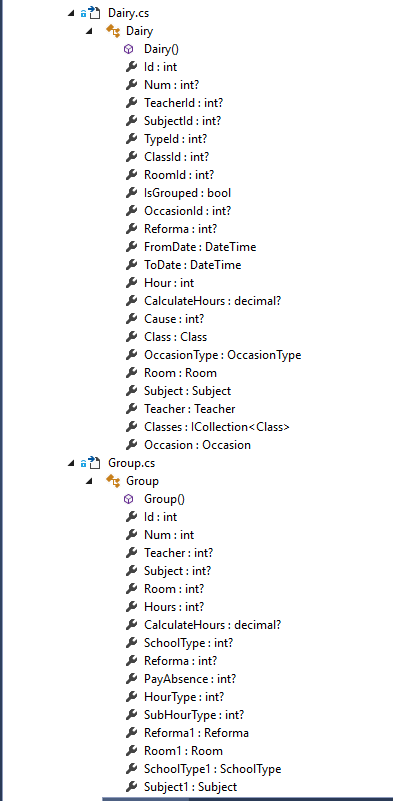
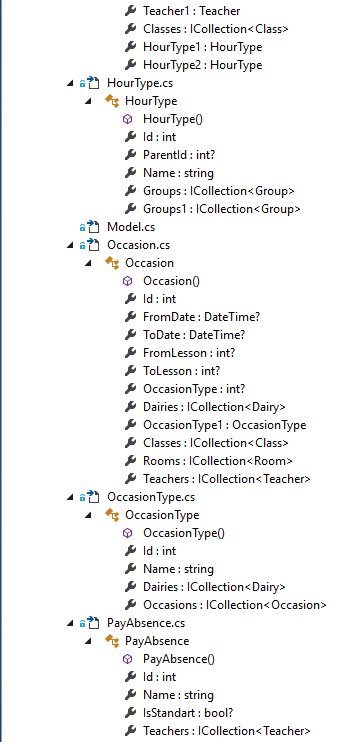
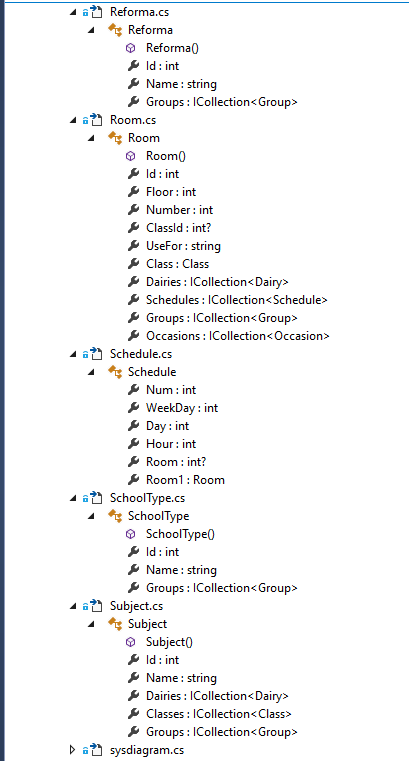
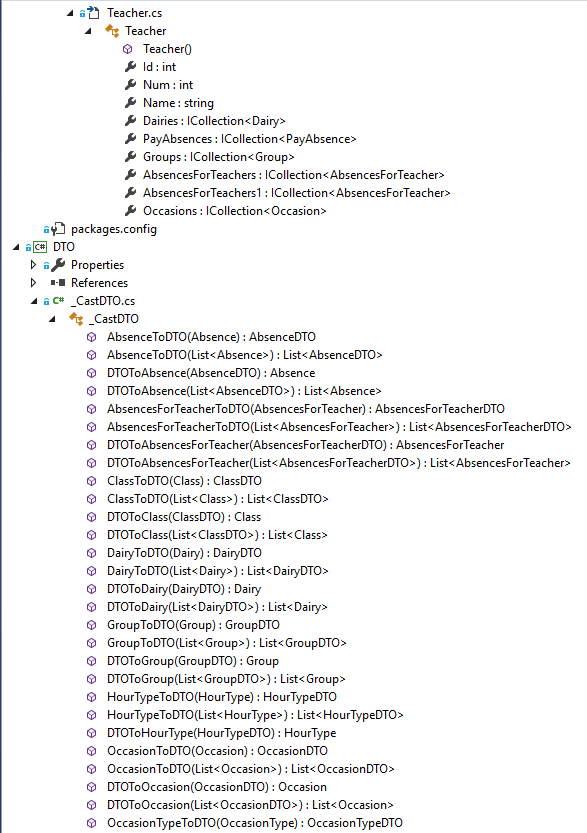
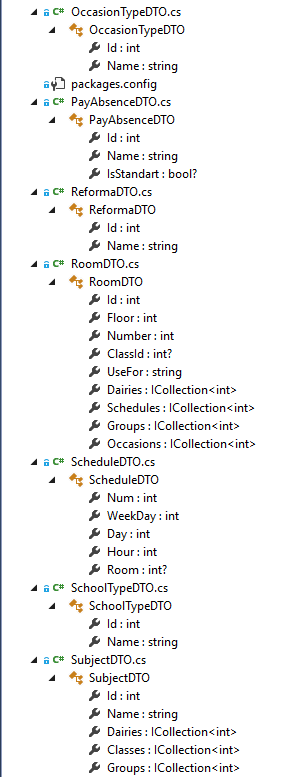
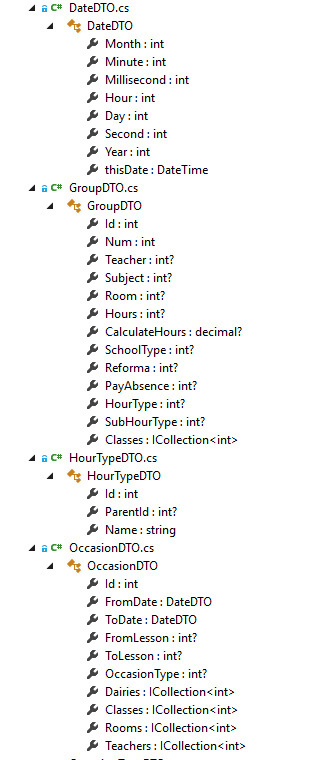
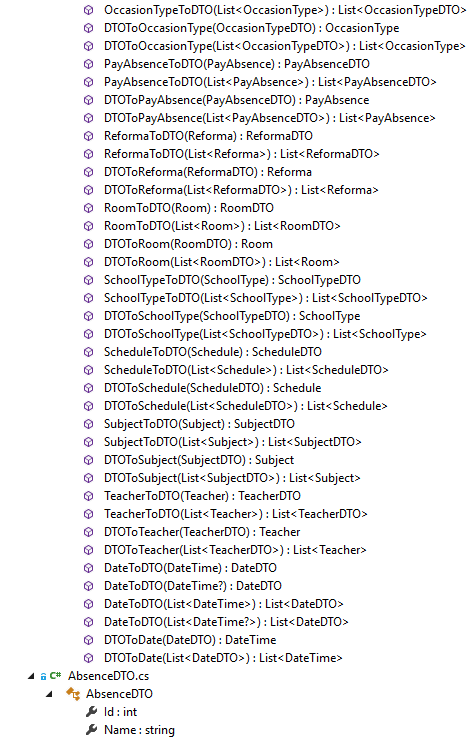
השתמשנו רבות בסיוע לשליפה מהמסד נתונים, לסנן וכדו'.

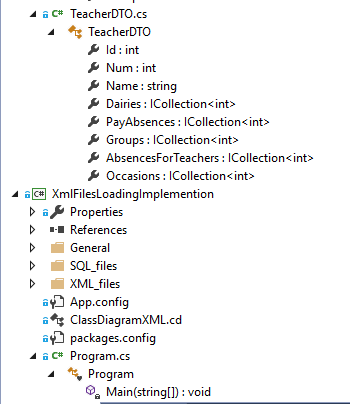
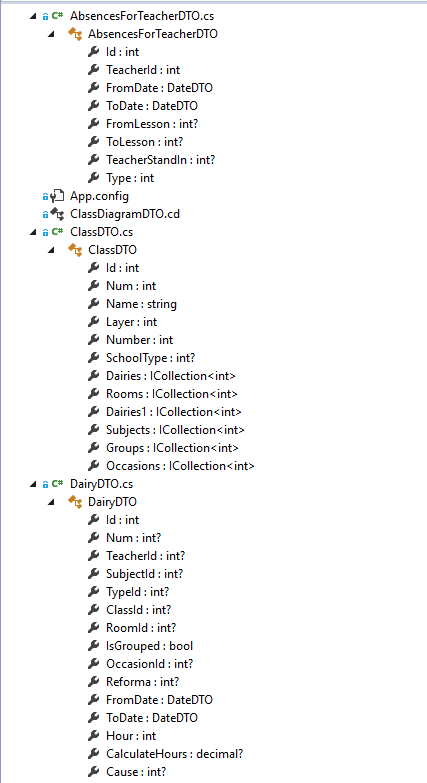
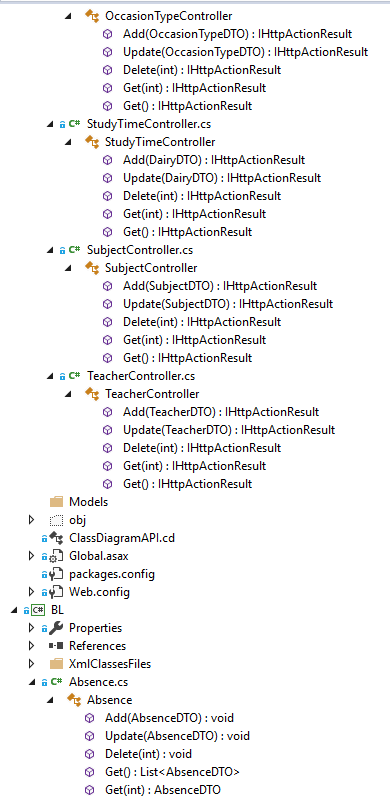
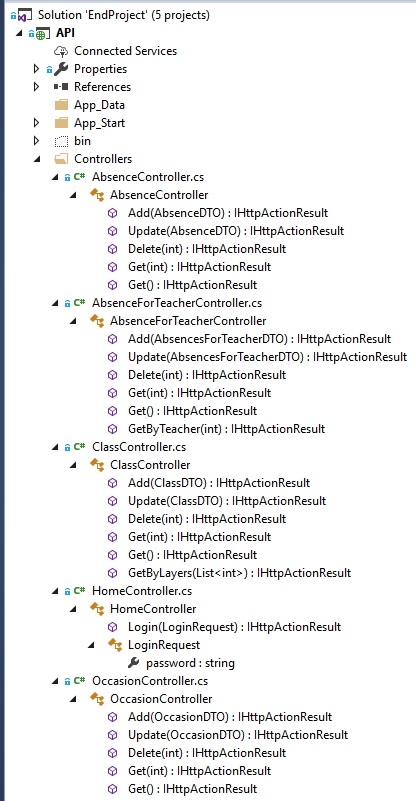
### Singleton

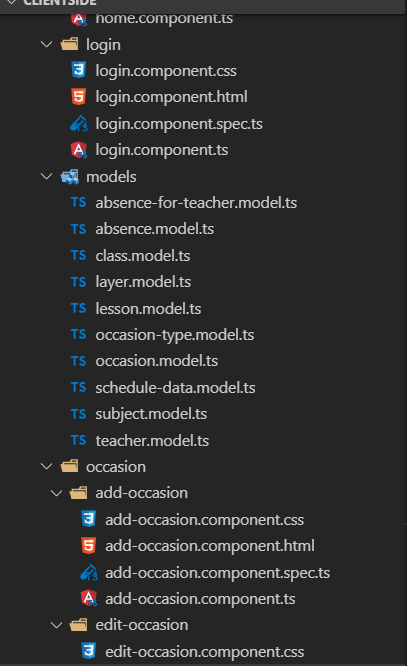
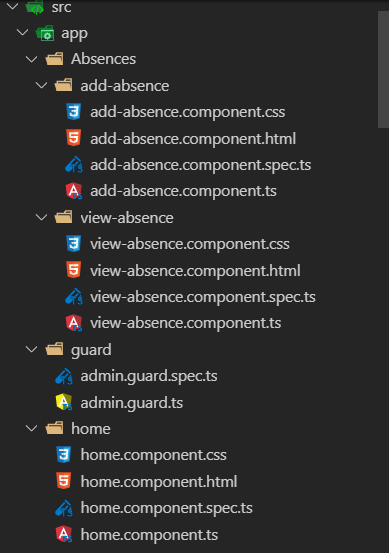
Singleton זו מחלקה שייתכן רק מופע אחד שלה. בצד לקוח השתמשנו בAngular ,השימוש בSingleton נפוץ בAngular לדוגמה בService וכן ב Route.

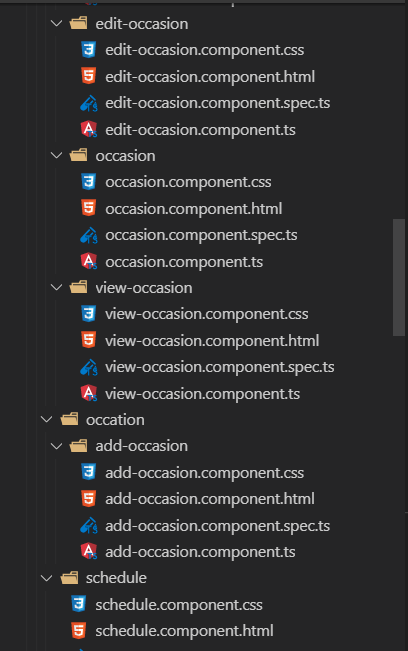
## תרשימים

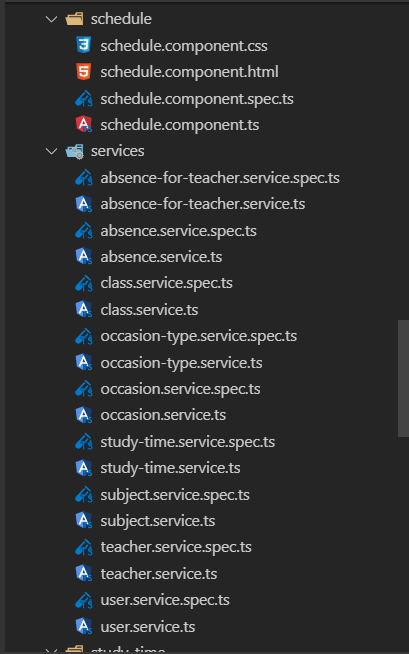
### תרשים מראה המחלקות

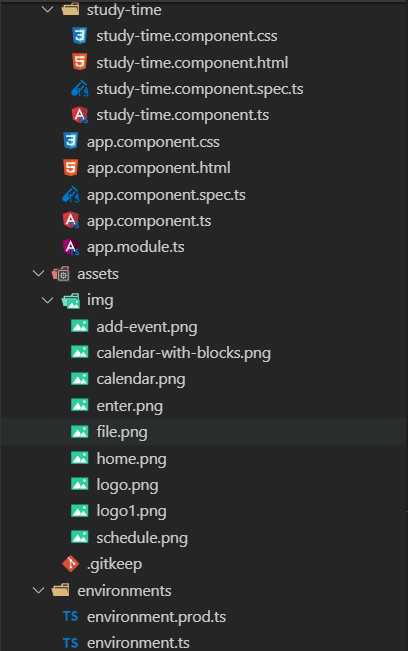
       

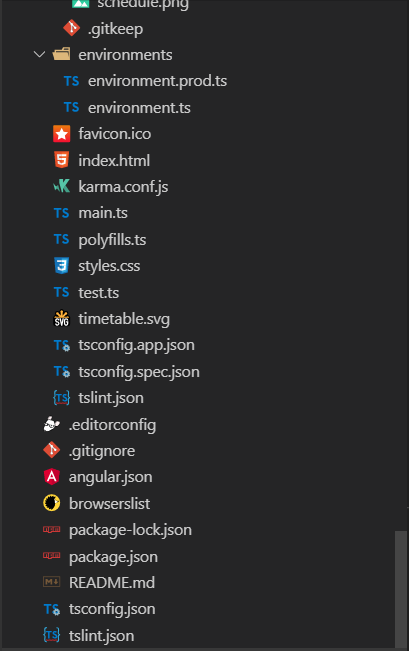




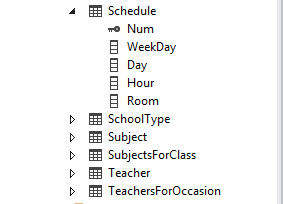
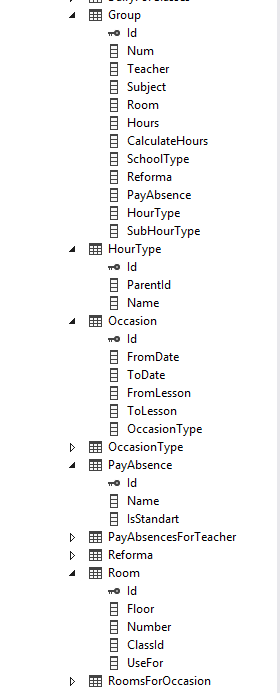
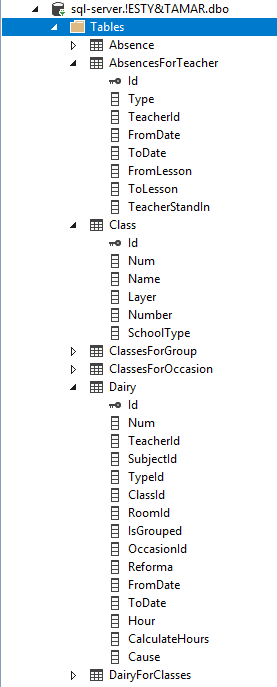








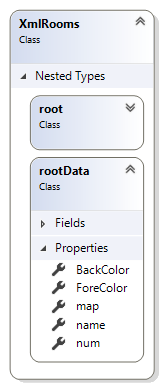
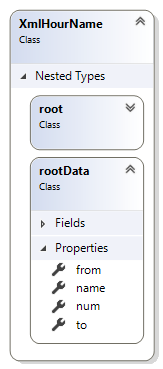
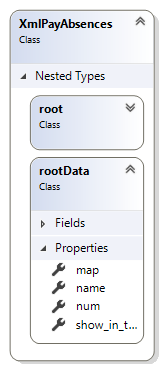
## מבנה  נתונים

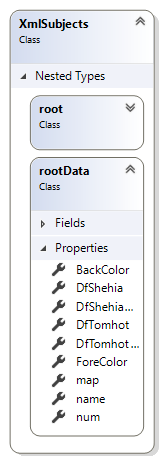
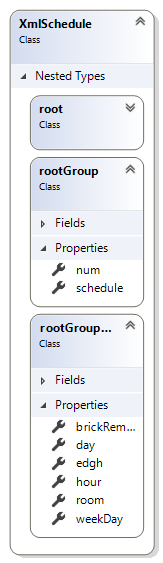
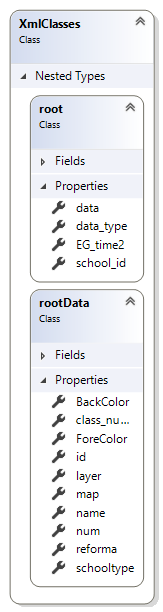
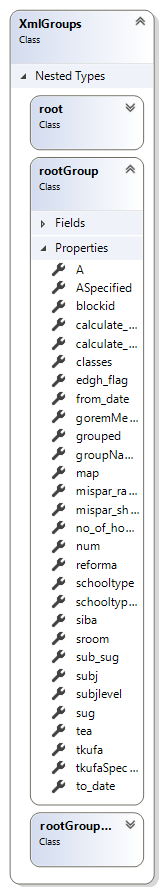
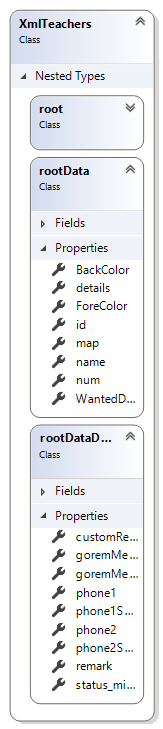






## תיאור מבנה קבצי XML

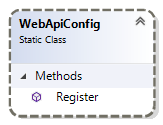
 

## תוכן הפרויקט

### תיאור המחלקות

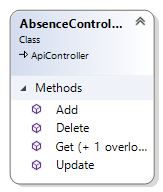
#### API

##### WebApiConfig

מחלקה המטפלת במאפיינים והגדרות של הApplication  


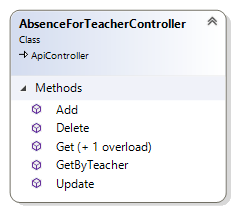
##### AbsenceController

מחלקה המטפלת בסוגי חיסורים (של מורות)



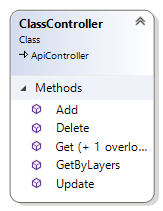
##### AbsenceForTeacherController

מחלקה המטפלת בחיסורים למורות



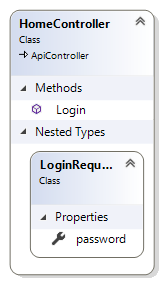
##### ClassController

מחלקה המטפלת בכיתות

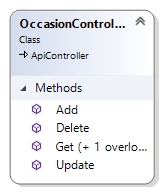


##### HomeController

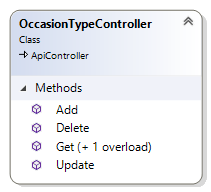
מחלקה המטפלת בהרשאות למערכת



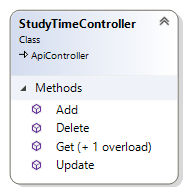
##### OccasionController

מחלקה המטפלת באירועים לכיתות  


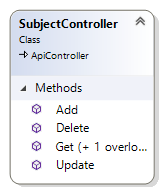
##### OccasionTypeController

מחלקה המטפלת בסוגי אירועים (של כיתות)   


##### StudyTimeController

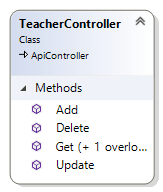
מחלקה המטפלת בשעת לימוד  


##### SubjectController

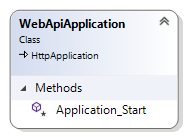
מחלקה המטפלת במקצועות  


##### TeacherController

מחלקה המטפלת במורות

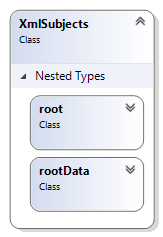
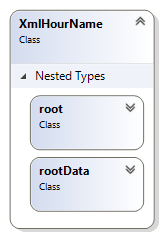
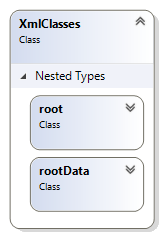
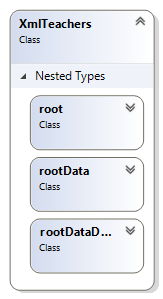
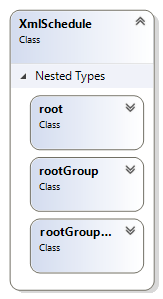
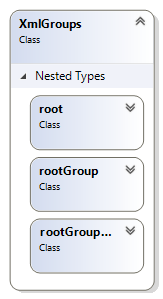
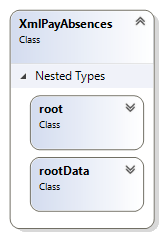
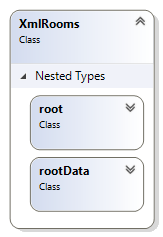


##### WebApiApplication

מחלקה המטפלת בבקשות ה Webבכלליות  


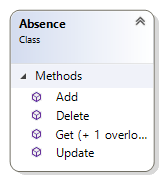
#### BL

##### XmlClassesFiles

המחלקות הבאות מקבילות לקובץ הXml המתאים  
  


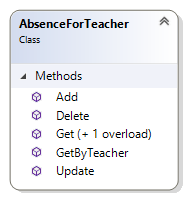
##### Absence

מחלקה המנהלת את סוגי חיסורים



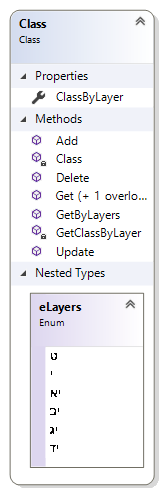
##### AbsenceForTeacher

מחלקה המנהלת את חיסורים למורות



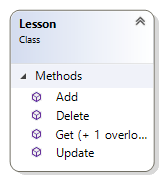
##### Class

מחלקה המנהלת את כיתות



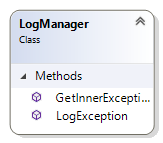
##### Lesson

מחלקה המנהלת את שעות לימוד (משבצת במערכת)



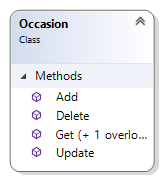
##### LogManager

מחלקה המנהלת את הLog –ים



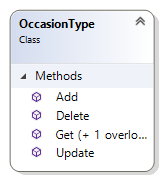
##### Occasion

מחלקה המנהלת את אירועים לכיתות



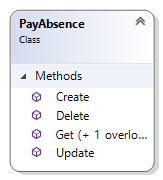
##### OccasionType

מחלקה המנהלת את סוגי אירועים



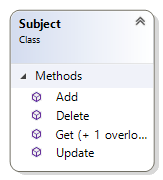
##### PayAbsence

מחלקה המנהלת את גורמים משלמים לחיסורי מורות



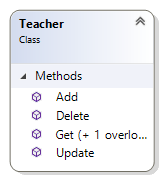
##### Subject

מחלקה המנהלת את מקצועות



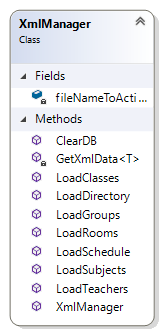
##### Teacher

מחלקה המנהלת את מורות



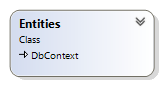
##### XmlManager

מחלקה המנהלת את קבצי הXml

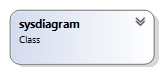


#### DAL

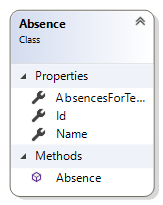
##### Entities

מ*חלקה המכילה אוספים המאפשרים לקבל נתונים מאובייקטים עיקריים במסד הנתונים*  
**

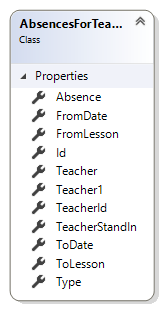
##### Sysdiagram

מחלקה *המכילה את נתוני הדיאגרמות במסד הנתונים*  


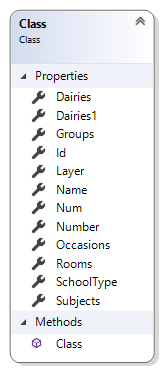
##### Absence

*המחלקה מכילה את צורת אובייקט סוג חיסור של מורה במסד הנתונים*

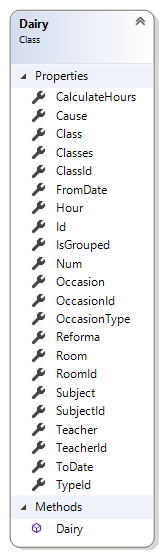
##### AbsencesForTeacher

*המחלקה מכילה את צורת אובייקט חיסור למורה במסד הנתונים*

##### Class

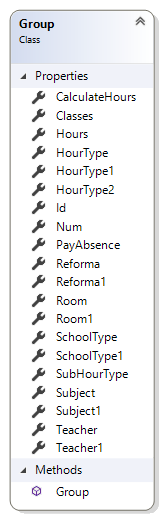
*המחלקה מכילה את צורת אובייקט כיתה במסד הנתונים*

##### Dairy

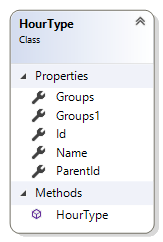
*המחלקה מכילה את צורת אובייקט יומן לימודים במסד הנתונים*

##### Group

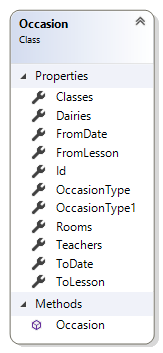
*המחלקה מכילה את צורת אובייקט שיעור הקבצה או איחוד במסד הנתונים*



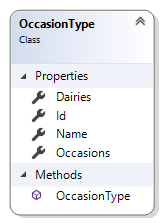
##### HourType

*המחלקה מכילה את צורת אובייקט סוג שעת לימוד במסד הנתונים*

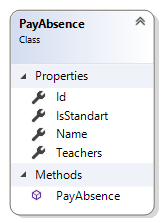
##### Occasion

*המחלקה מכילה את צורת אובייקט אירוע לכיתות במסד הנתונים*  


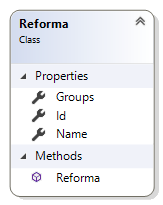
##### OccasionType

*המחלקה מכילה את צורת אובייקט סוג ארוע במסד הנתונים*

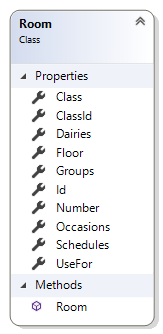
##### PayAbsence

*המחלקה מכילה את צורת אובייקט גורם משלם במסד הנתונים*

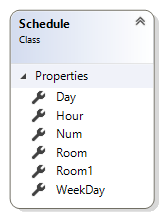
##### Reforma

*המחלקה מכילה את צורת אובייקט רפורמה במסד הנתונים*

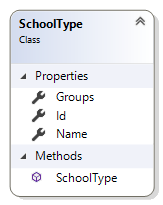
##### Room

*המחלקה מכילה את צורת אובייקט חדר במסד הנתונים*

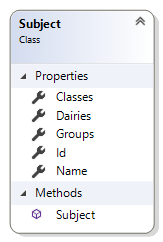
##### Schedule

*המחלקה מכילה את צורת אובייקט מסגרת מערכת השעות הקבועה במסד הנתונים*

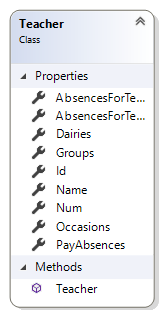
##### SchoolType

*המחלקה מכילה את צורת אובייקט סוג ביה"ס במסד הנתונים*

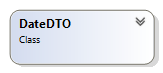
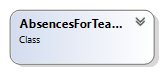
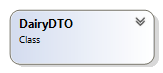
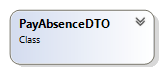
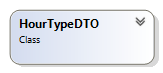
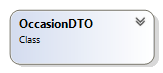
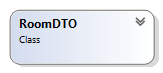
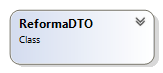
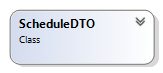
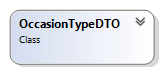
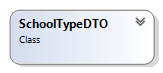
##### Subject

*המחלקה מכילה את צורת אובייקט מקצוע במסד הנתונים*

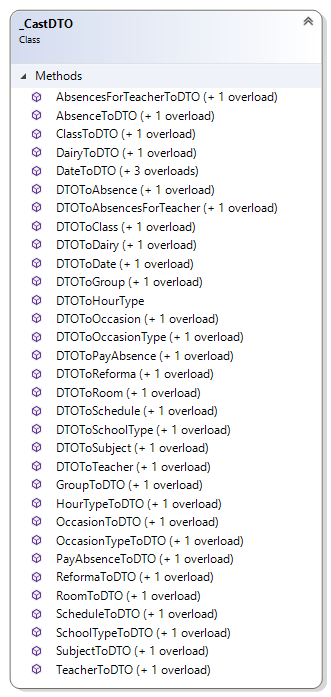
##### Teacher

*המחלקה מכילה את צורת אובייקט מורה במסד הנתונים*

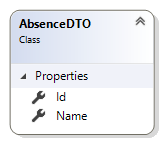
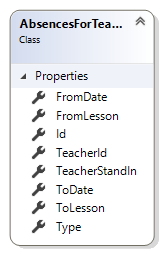
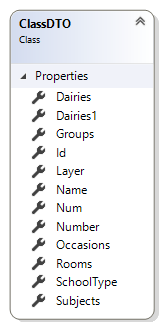
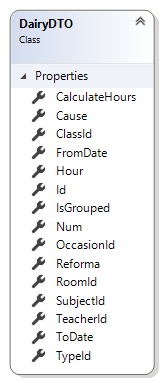
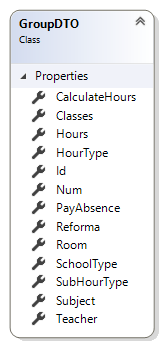
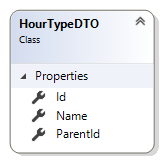
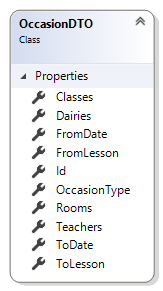
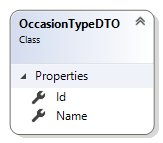
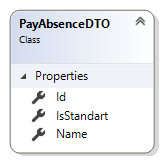
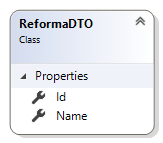
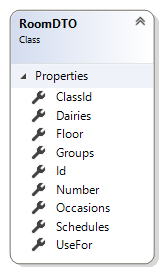
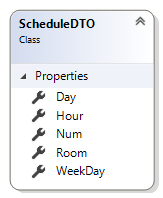
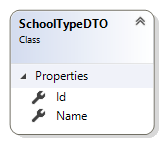
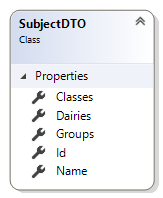
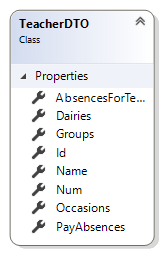
#### DTO

מחלקות המכילות טיפוסי נתונים מקבילים לשכבת הDTO  
               

##### \_CastDTO

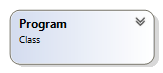
מחלקה המכילה מתודות המרה משכבת הDAL לשכבת הDTO ולהיפך  


##### מחלקות מקבילות למחלקות בשכבת הDal

ללא שדות ניווט  
  
  
  
  
  
  
  
  
  
  
  
  
  
  


#### XmlFilesLoadingImplemention

##### Program

*מחלקה המכילה קוד לטעינת הנתונים מקבצי ה* Xml  


### פונקציות מעניינות

#### API

##### Login

[HttpPost]//define the data transfer method

[Route("login")]//define the url

public IHttpActionResult Login([FromBody]LoginRequest loginRequest)

{

if (loginRequest.password.Equals(WebConfigurationManager.AppSettings["password"])) {

return StatusCode(HttpStatusCode.NoContent);

}

return Unauthorized();

}

public class LoginRequest

{

public string password { get; set; }

}

בודקת האם הסיסמה נכונה, ז"א נגשת לקובץ הקונפיגורציה ובודקת בהגדרות מה הvalue של הסיסמה, אם הסיסמאות שוות- מחזירה תגובה (ריקה אמנם, כי אין מה להחזיר), אחרת מחזירה תגובה מסוג "לא מאופשר".  
לצורך קבלת הסיסמה, לא קבלנו במתודה מחרוזת רגילה, אלא יצרנו מחלקה שעוטפת (Wrapper), מכיוון שAngular יכול רק לקבל או/ ו להחזיר אובייקטים ולא פרימיטיביים (Value Type).

#### BL

##### GetsByLayers

רצינו להחזיר אוסף כיתות לפי מחזור/ים.

public static Dictionary<int, List<ClassDTO>> ClassByLayer { get; } = new Dictionary<int, List<ClassDTO>>();

יצרנו Dictionary שהמפתח הוא מספר מחזור והערך הוא אוסף הכיתות של המחזור.

static Class()

{

for (int i = 9; i < 15; i++)

{

ClassByLayer.Add(i, GetClassByLayer(i));

}

}

בבונה הסטטית של מחלקת כיתה ה Dictionary מתאתחל בנתונים.

private static List<ClassDTO> GetClassByLayer(int layer)

{

using (Entities db = new Entities())

{

return \_CastDTO.ClassToDTO(db.Classes.Where(c => c.Layer == layer).ToList());

}

}

מחזירה אוסף כיתות לפי מספר מחזור. כיון שהעניין דורש גישה למבנה הנתונים, נמצאה דרך יעילה להוריד את הגישות למסד הנתונים כמה שפחות, לכן מתודה זו פרטית.

public static List<ClassDTO> GetByLayers(List<int> layers)

{

List<ClassDTO> classes = new List<ClassDTO>();

if (layers.Contains(0))

{

for (int i = 9; i < 15; i++)

{

classes.AddRange(ClassByLayer[i]);

}

return classes;

}

foreach (int i in layers)

{

classes.AddRange(ClassByLayer[i]);

}

return classes;

}

כאן המתודה מקבלת אוסף מחזורים, ומחזירה כיתות לפי האוסף.  
אם קים מספר 0 באוסף הכונה לכל המחזורים, אחרת עוברים לפי האוסף ומוסיפים את הכיתות המתאימות מה Dictionary.

##### LoadDirectory

##### LoadClasses

##### GetXmlData

#### DTO

##### DateToDTO

public static DateDTO DateToDTO(DateTime date)

{

return new DateDTO()

{

Day = date.Day,

Hour = date.Hour,

Millisecond = date.Millisecond,

Minute = date.Minute,

Month = date.Month,

Second = date.Second,

Year = date.Year

};

}

ממירה מאובייקט DateTime של C#, כחלק מתהליך שליפת נתון, לאובייקט מסוג DateDTO (מחלקה שיצרנו, מכיוון שאובייקט תאריך בAngular הוא Date בלבד, והפרמטרים לא הצליחו להיכנס כראוי)

public static DateDTO DateToDTO(DateTime? date)

{

DateTime thisDate = new DateTime();

return new DateDTO()

{

thisDate = date.GetValueOrDefault(),

Hour = thisDate.Hour,

Millisecond = thisDate.Millisecond,

Minute = thisDate.Minute,

Month = thisDate.Month,

Second = thisDate.Second,

Year = thisDate.Year

};

}

ממירה מאובייקט DateTime של C#, כחלק מתהליך שליפת נתון, לאובייקט מסוג DateDTO.  
פונקציה מועמסת-נוספת כדי להימנע מנפילה במקרה שהתאריך ריק.

public static List<DateDTO> DateToDTO(List<DateTime> dateList)

{

return dateList.Select(d => DateToDTO(d)).ToList();

}

ממירה מאוסף אובייקטים DateTime>> של C#, כחלק מתהליך שליפת נתון, לאוסף אובייקטים מסוג DateDTO>>.

##### DTOToDate

public static DateTime DTOToDate(DateDTO date)

{

return new DateTime(date.Year, date.Month, date.Day, date.Hour, date.Minute, date.Second, date.Millisecond);

}

ממירה מאובייקט DateDTO, כחלק מתהליך שמירת נתון, לאובייקט מסוג DateTime של C#.

public static List<DateTime> DTOToDate(List<DateDTO> dateList)

{

return dateList.Select(d => DTOToDate(d)).ToList();

}

ממירה מאוסף אובייקטים <DateDTO>, כחלק מתהליך שמירת נתון, לאוסף אובייקטים מסוג <DateTime> של C#.

#### מנגנון טיפול בשגיאות-LogManager

פונקציה זו כותבת את הודעת השגיאה (המלאה) ל – Eevent Viewer:

public static void LogException(Exception e)

{

using (EventLog eventLog = new EventLog())

{

eventLog.Source = "SchoolSchedule";

eventLog.WriteEntry(GetInnerExceptionMessage(e), EventLogEntryType.Error);

} /\*\*/

}

פונקציה זו מחזירה את כל הודעות השגיאות הפנימיות משורשרות זו לזו:

public static string GetInnerExceptionMessage(Exception e)

{

if (e.InnerException != null)

{

return e.Message + "\n" + GetInnerExceptionMessage(e.InnerException);

}

return e.Message + "\n" + e.StackTrace + "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

}

#### מנגנון טעינת קבצים ואתחול מערכת-XmlManager

public XmlManager()

{

fileNameToAction = new Dictionary<string, Action<string>>();

fileNameToAction["classes.xml"] = LoadClasses;

fileNameToAction["groups.xml"] = LoadGroups;

fileNameToAction["rooms.xml"] = LoadRooms;

fileNameToAction["schedule.xml"] = LoadSchedule;

fileNameToAction["subjects.xml"] = LoadSubjects;

fileNameToAction["teachers.xml"] = LoadTeachers;

}

פונקציה זו טוענת את תקיית קבצי הXML:

public void LoadDirectory(string path)

{

//reset the DB

ClearDB();

var directoryInfo = new DirectoryInfo(path);

var files = directoryInfo.GetFiles();

foreach (var file in files)

{

Action<string> action;

if (fileNameToAction.TryGetValue(file.Name, out action))

{

action?.Invoke(file.FullName);

Console.WriteLine("{0} Loading....", file.FullName);

}

}

}

פונקציה זו מאתחלת את ה DB מחדש:

public void ClearDB()

{

var conn = new SqlConnection("data source=DESKTOP-7A0S24C; initial catalog=!ESTY&TAMAR; integrated security=True");

SqlCommand cmd;

conn.Open();

cmd = new SqlCommand("sp\_MSforeachtable 'DELETE FROM ?'", conn);

cmd.ExecuteNonQuery();

cmd = new SqlCommand("sp\_MSforeachtable 'truncate table ?'", conn);

cmd.ExecuteNonQuery();

conn.Close();

}

פונקציה זו טוענת קבצי כיתות (:(XML

public void LoadClasses(string path)

פונקציה זו טוענת קבצי קבוצות (:(XML

public void LoadGroups(string path)

פונקציה זו טוענת קבצי חדרים (:(XML

public void LoadRooms(string path)

פונקציה זו טוענת קבצי מערכת קבועה (:(XML

public void LoadSchedule(string path)

פונקציה זו טוענת קבצי מקצועות (:(XML

public void LoadSubjects(string path)

פונקציה זו טוענת קבצי מורות (:(XML

public void LoadTeachers(string path)

פונקציה זו ממירה את הטקסט של הקובץ לאובייקטים:

private static T GetXmlData<T>(string path)

{

XmlSerializer serializer = new XmlSerializer(typeof(T));

StringReader stringReader = new StringReader(File.ReadAllText(path, Encoding.GetEncoding(1255)));

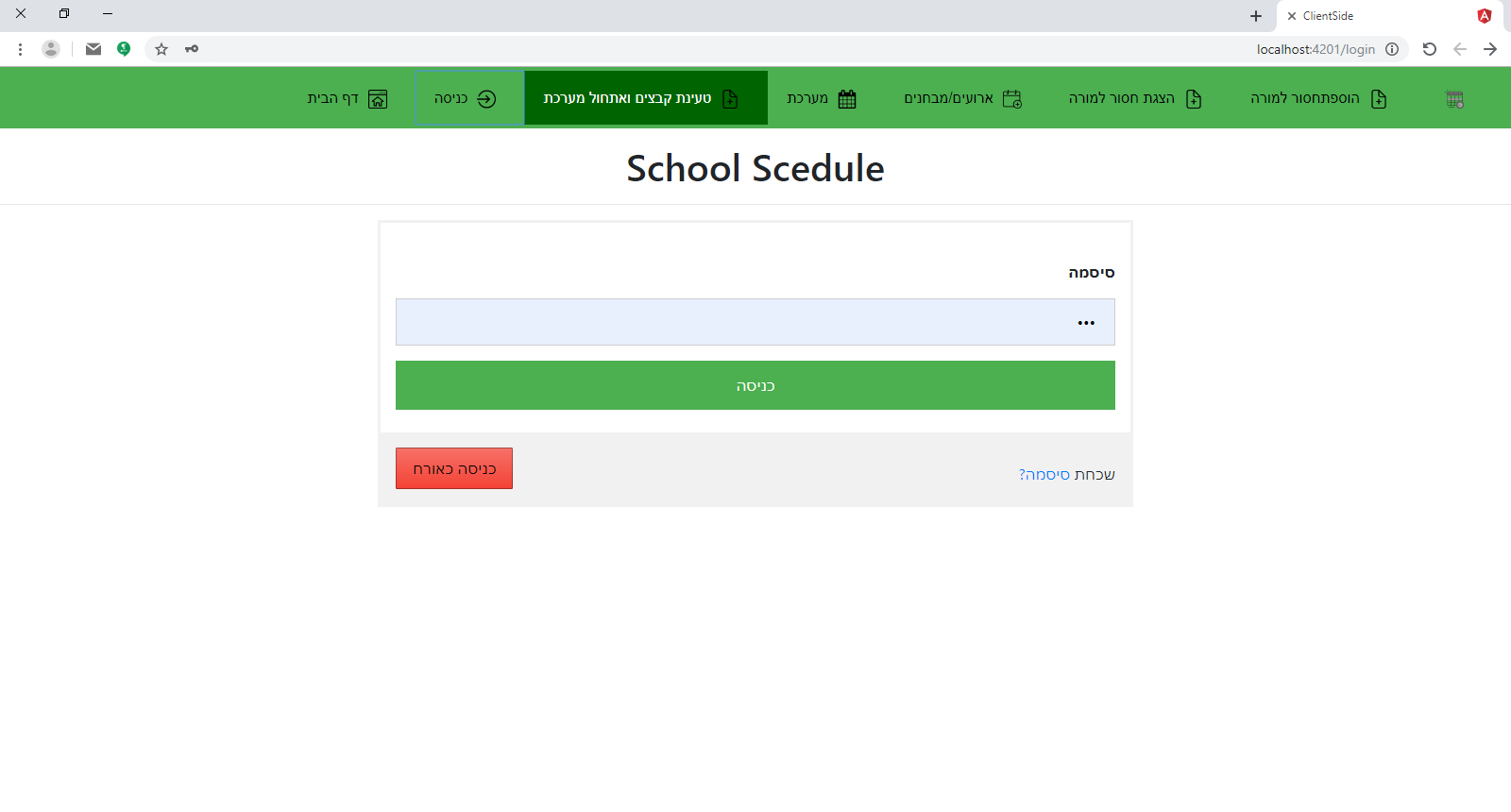
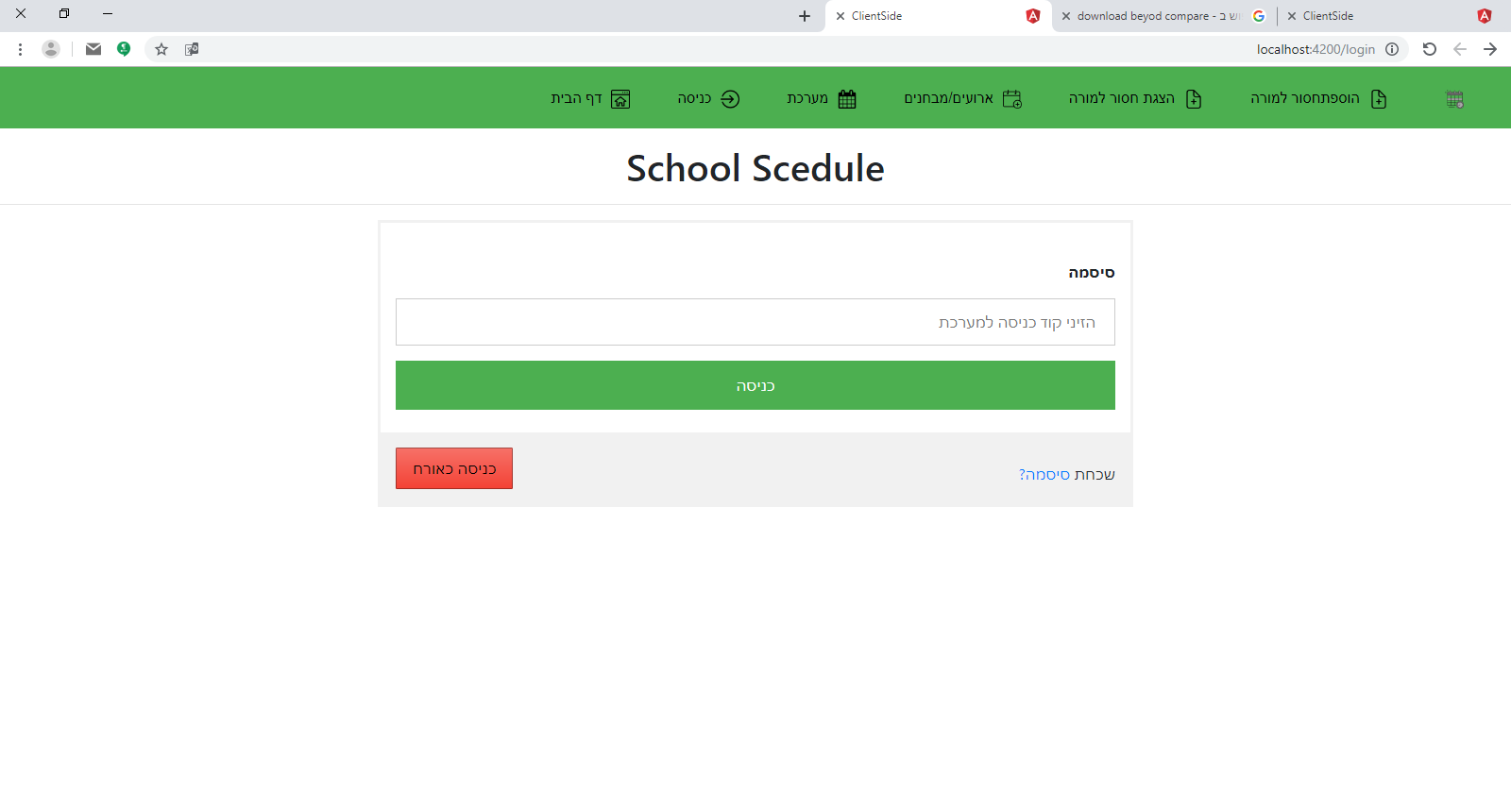
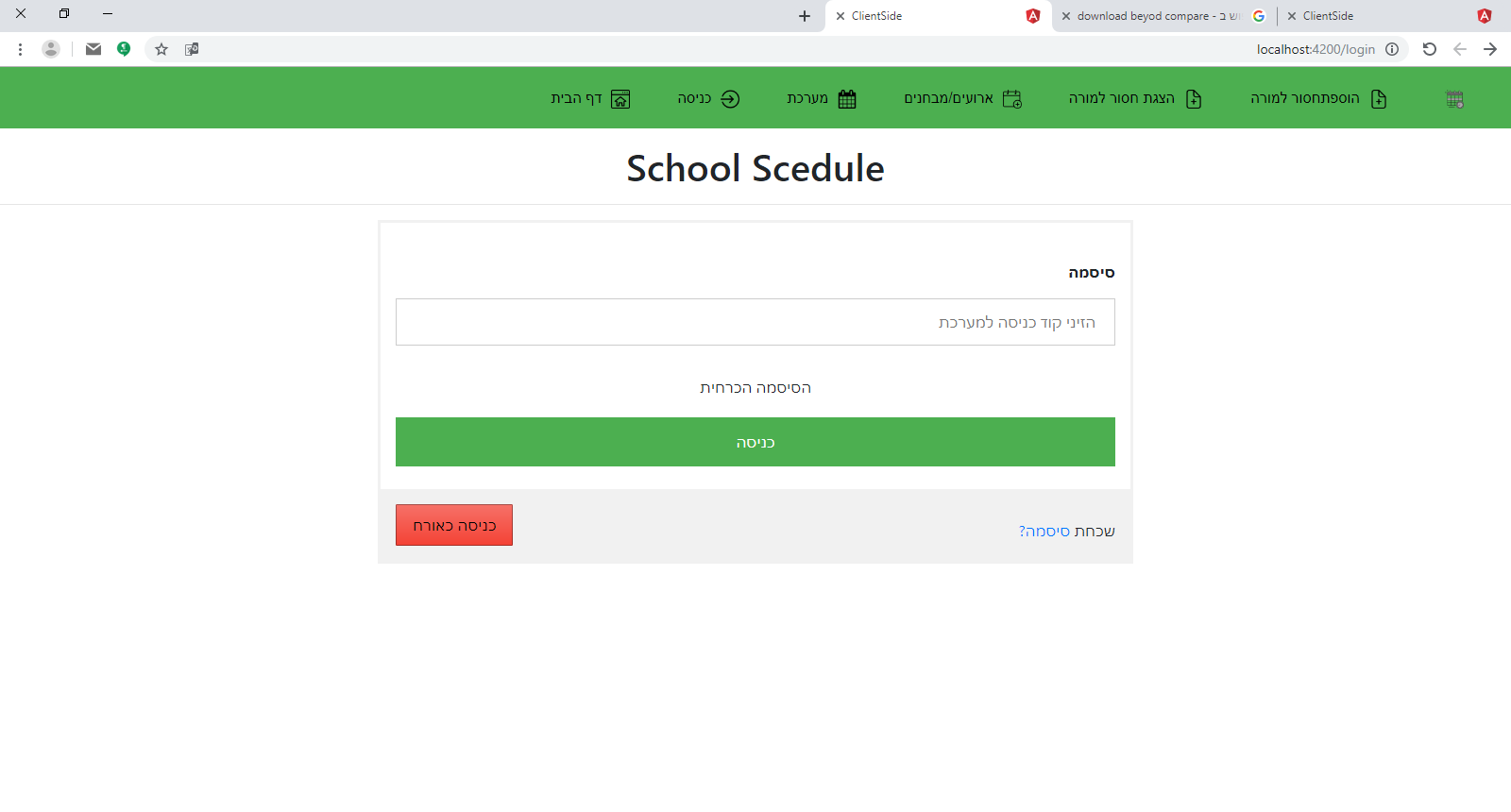
T root = (T)serializer.Deserialize(stringReader);

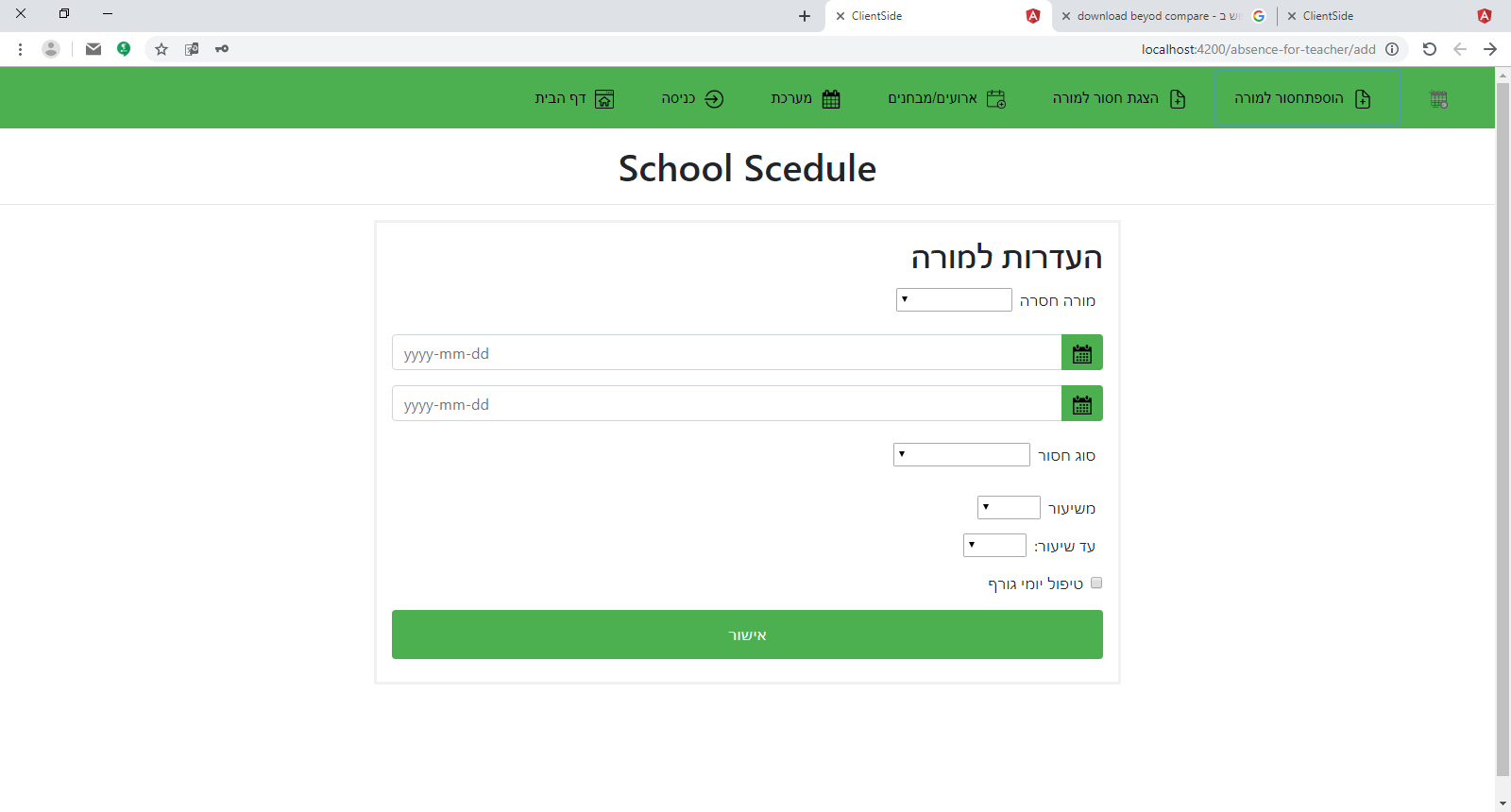
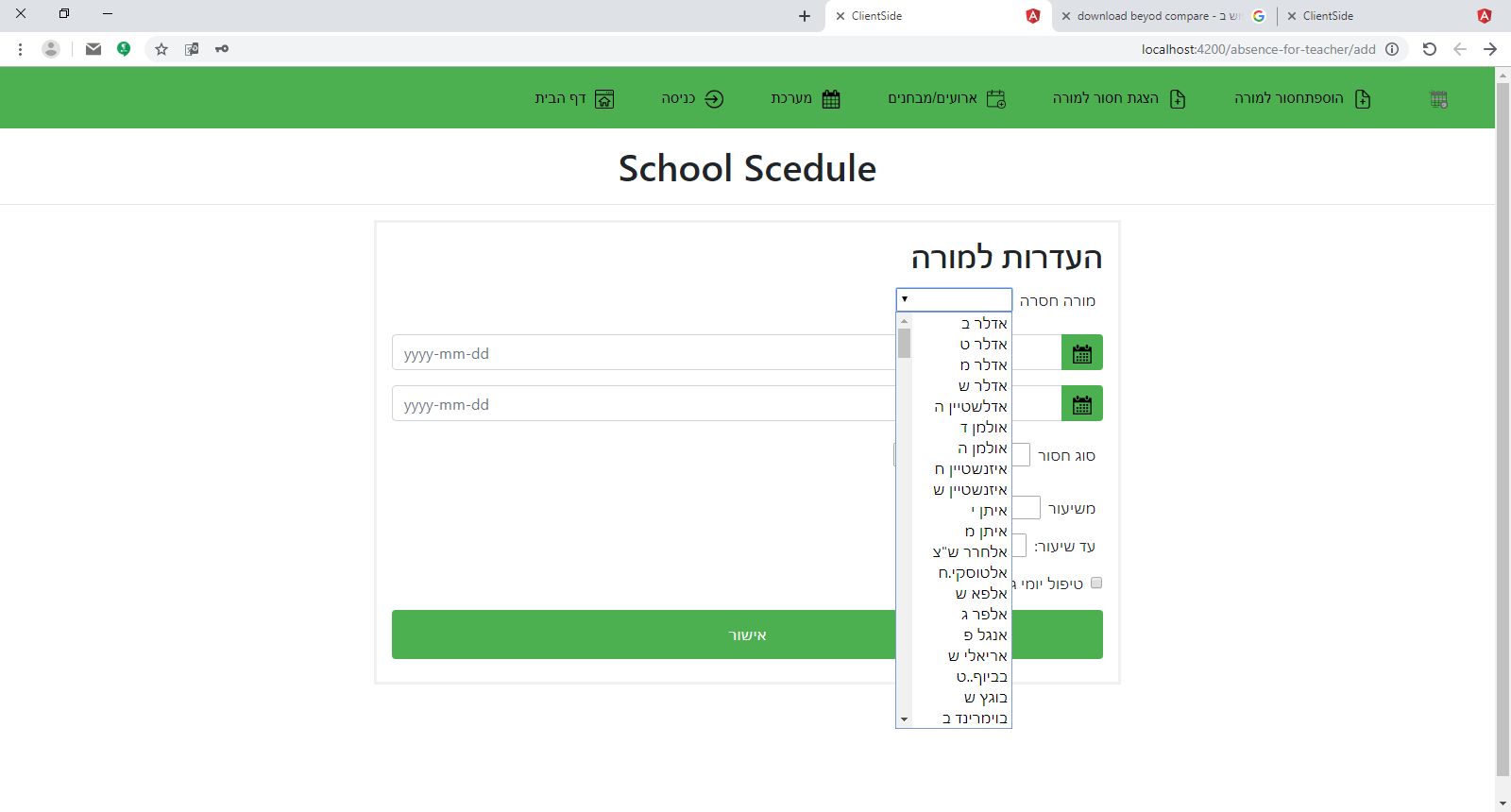
return root;

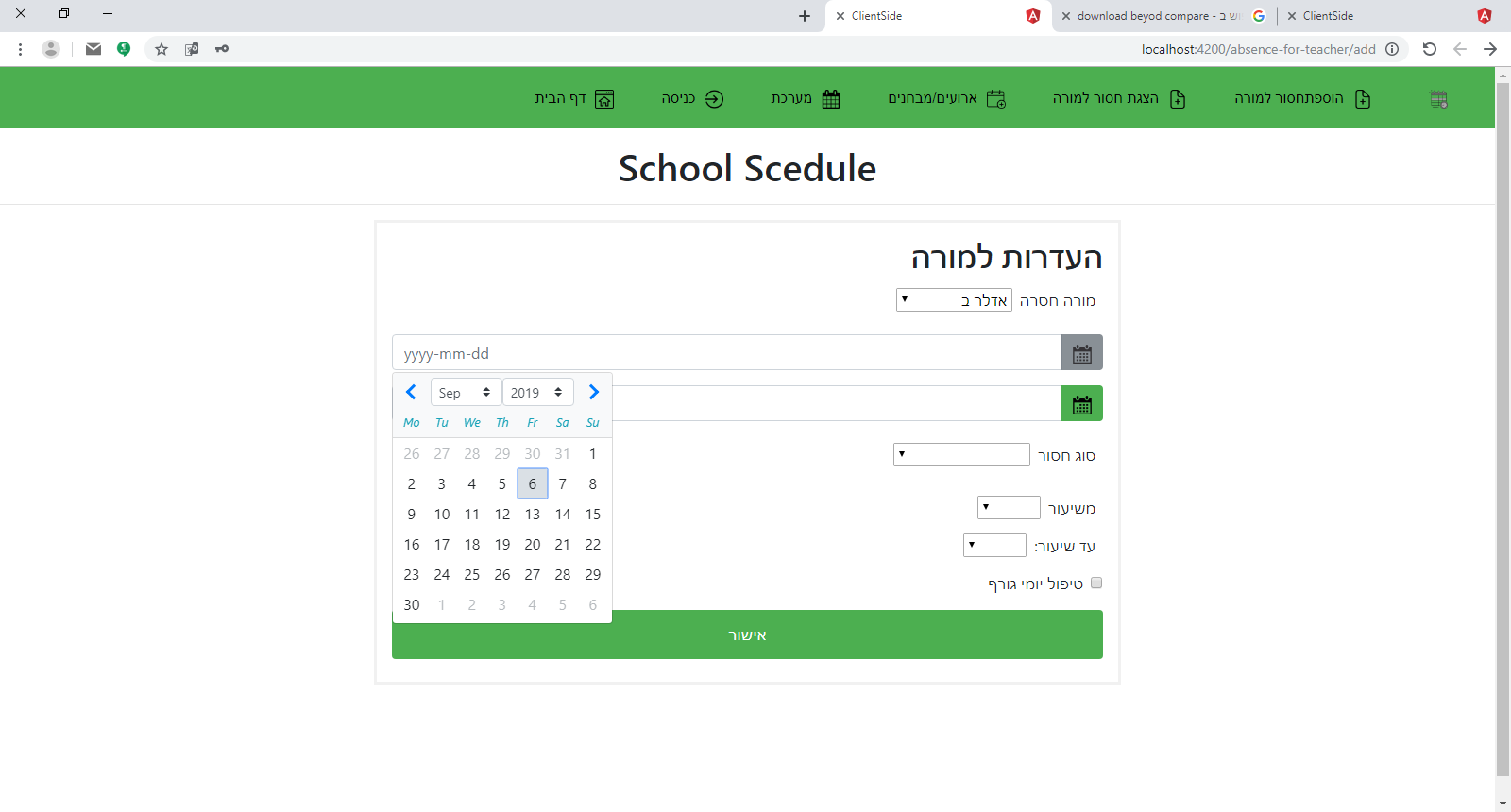
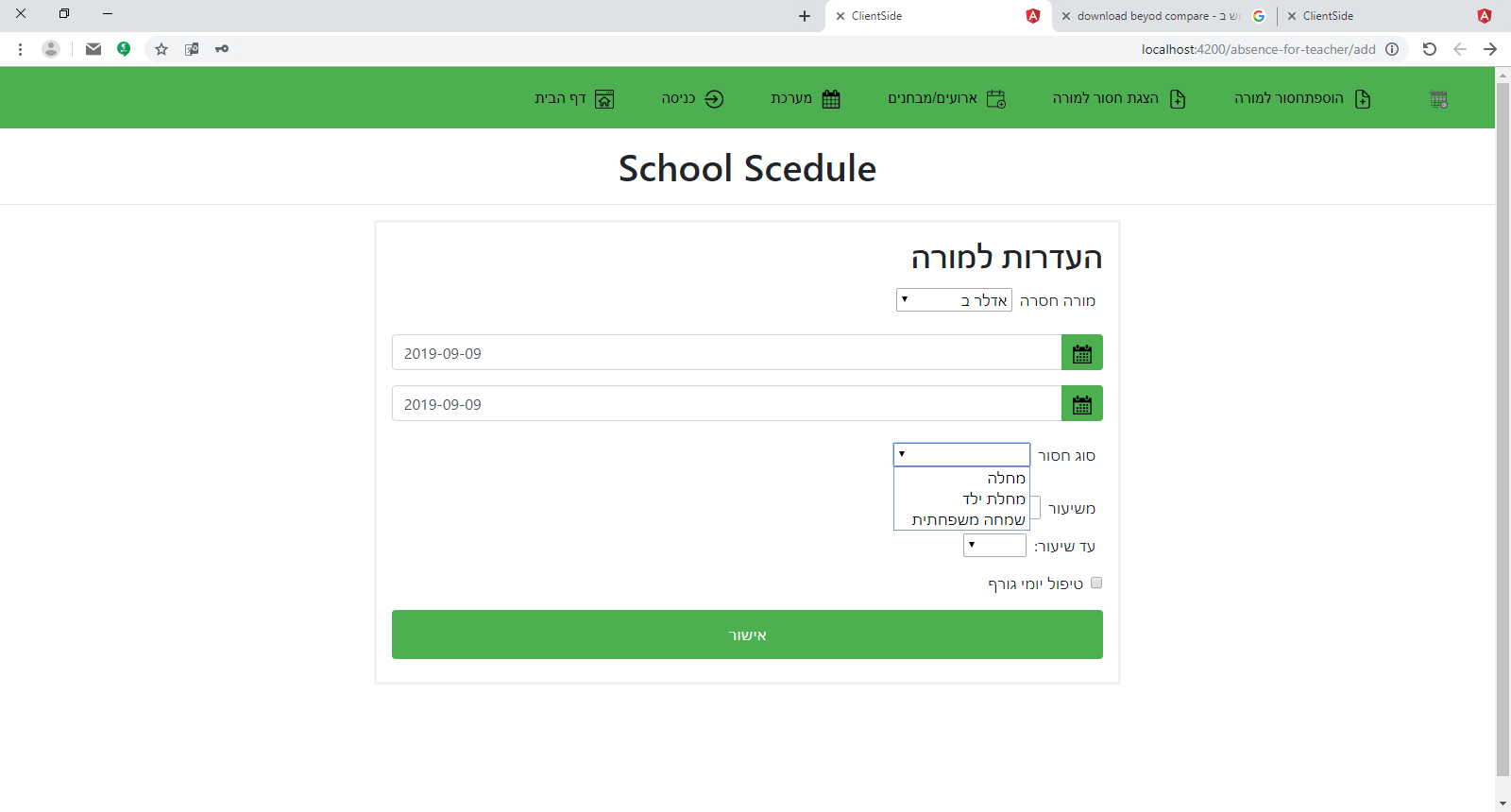
}

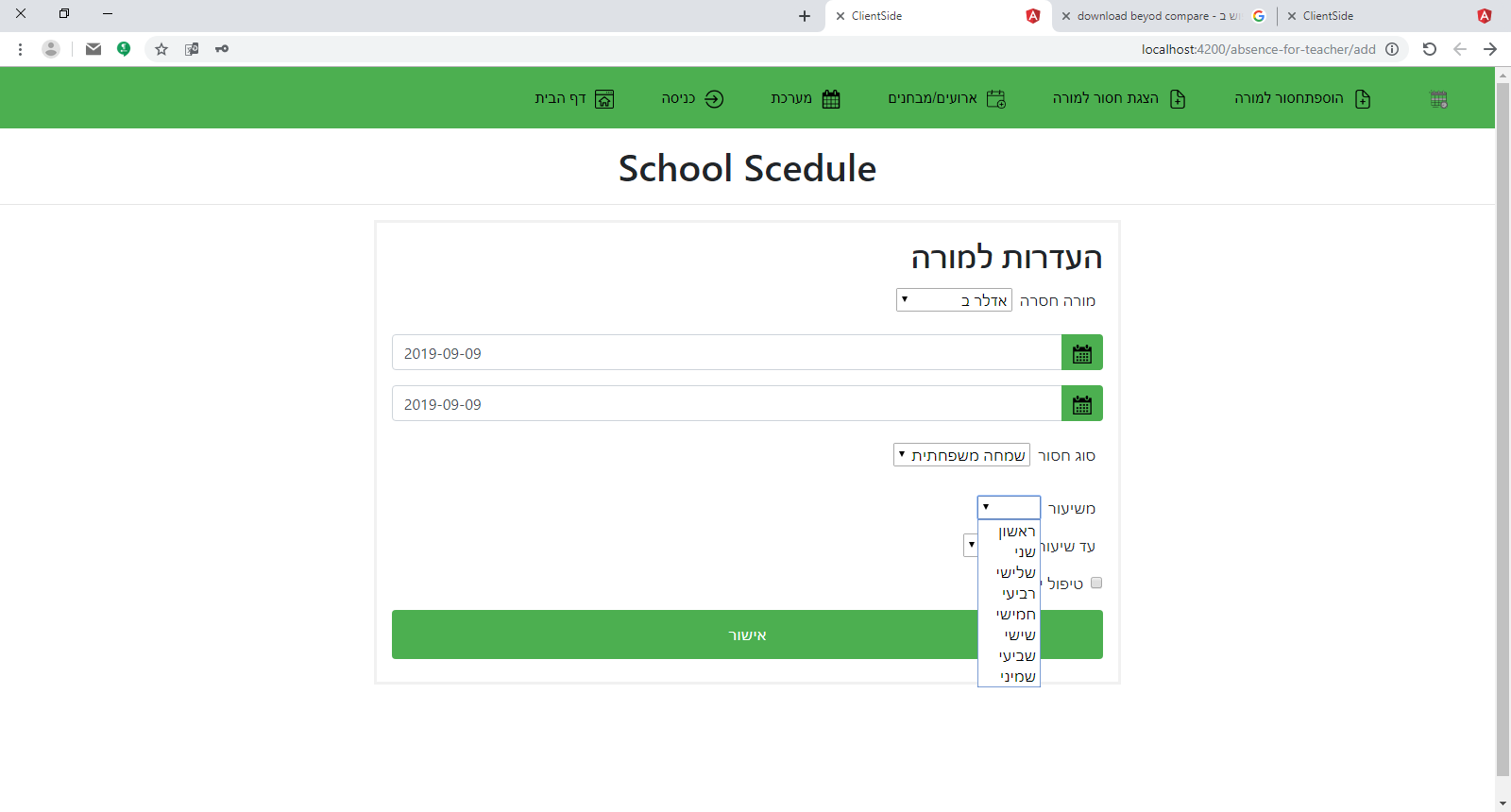
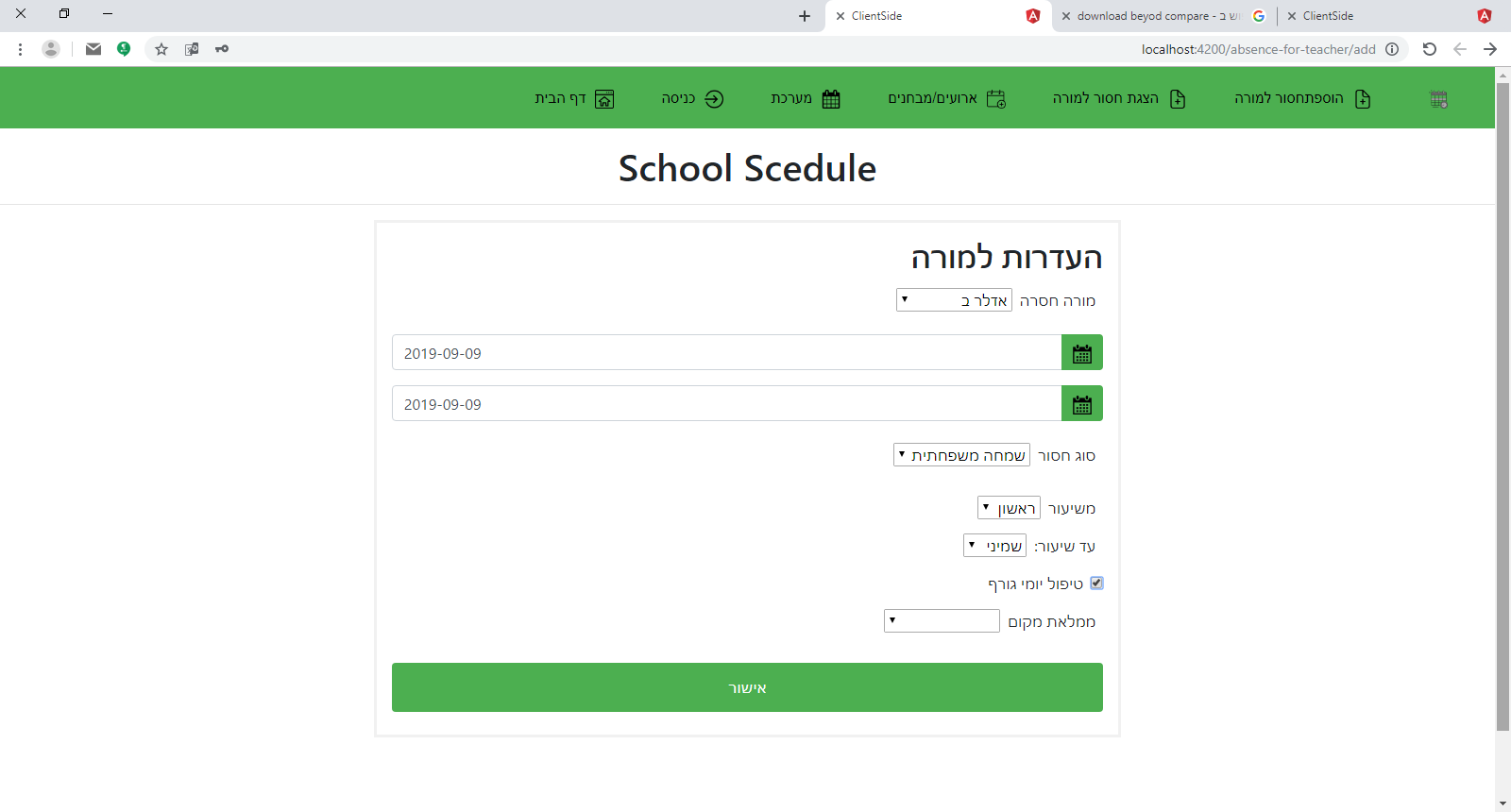
# מדריך למשתמש

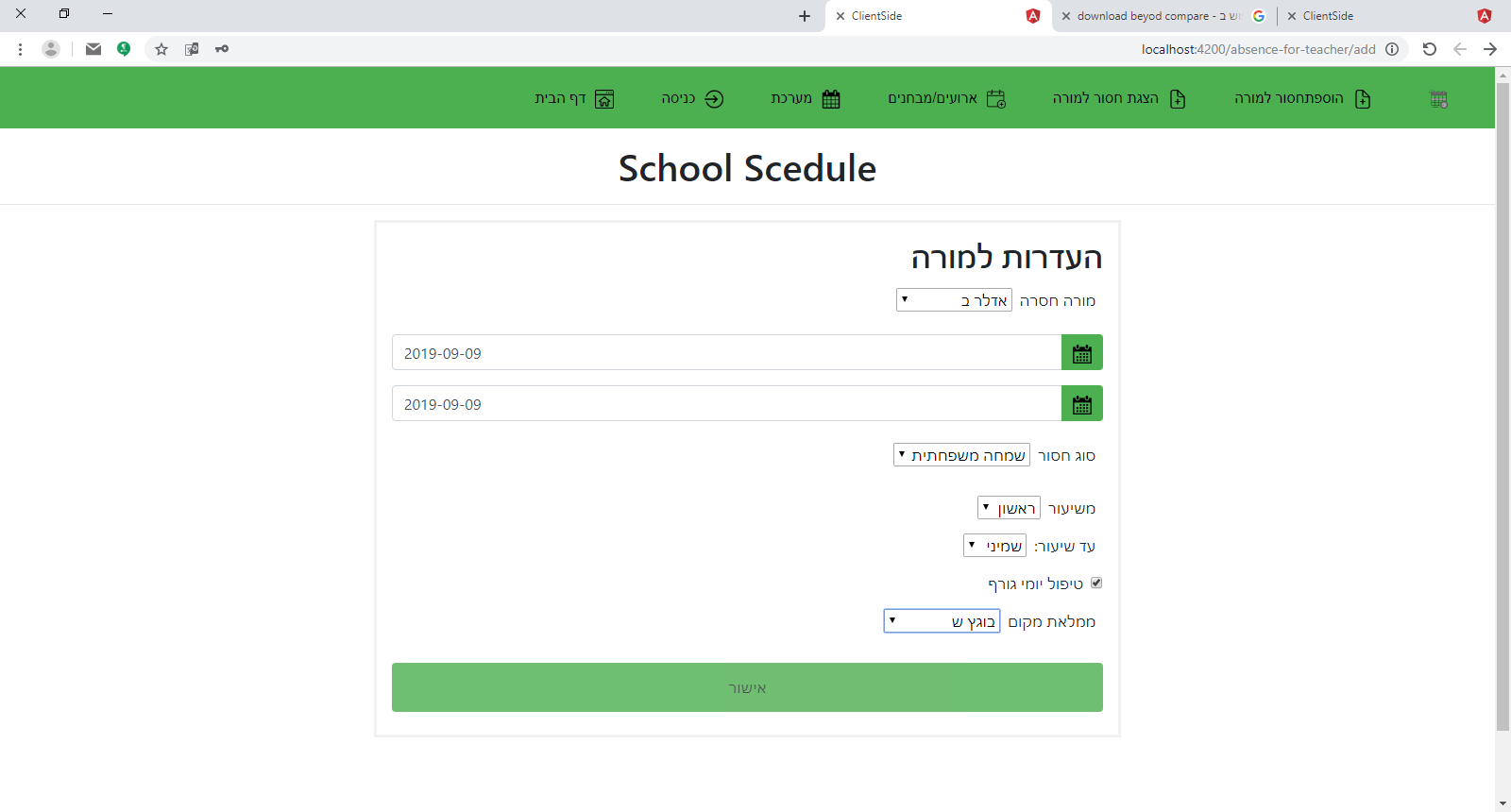
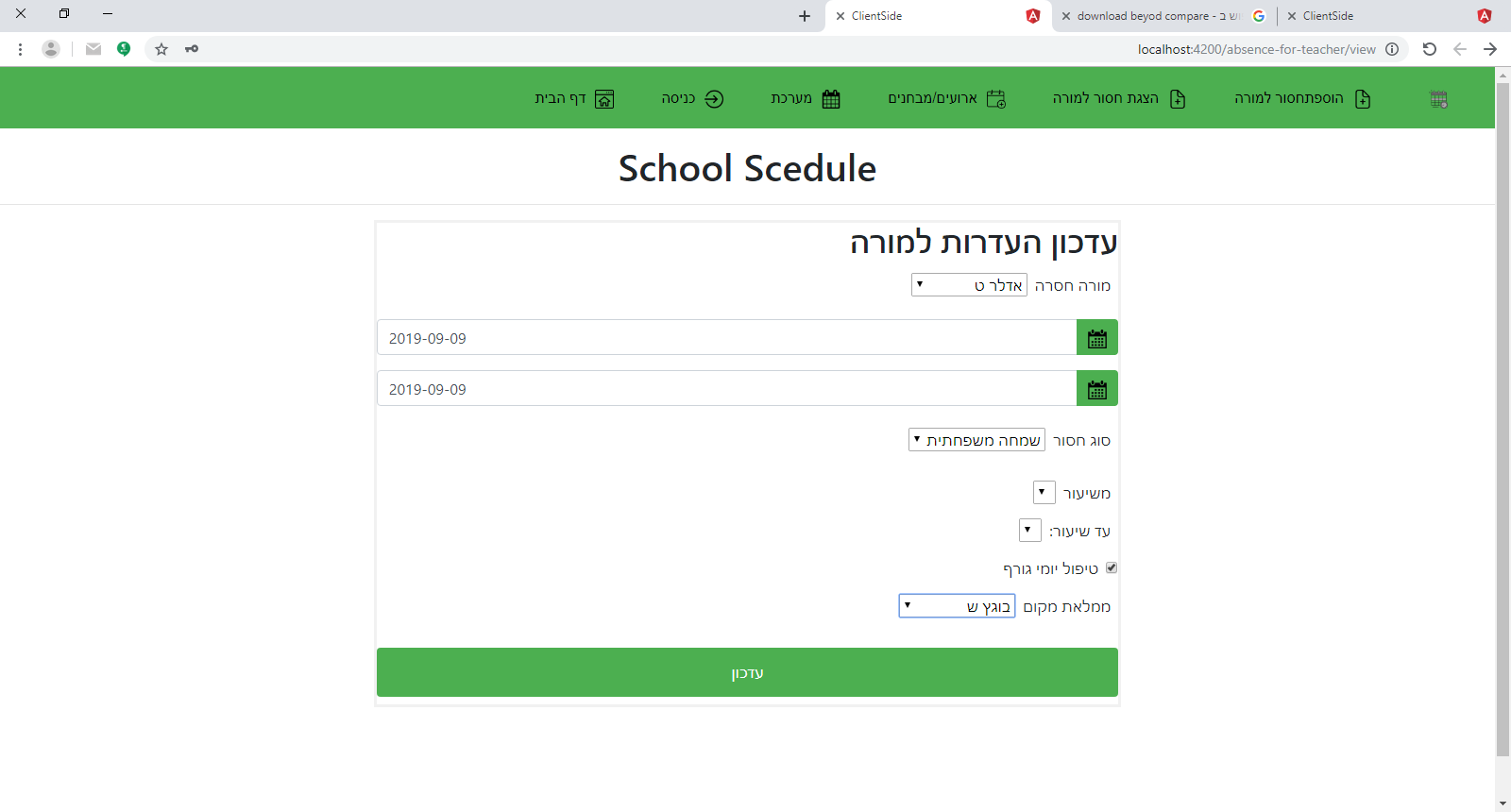
הוראות כלליות לשימוש במערכת:

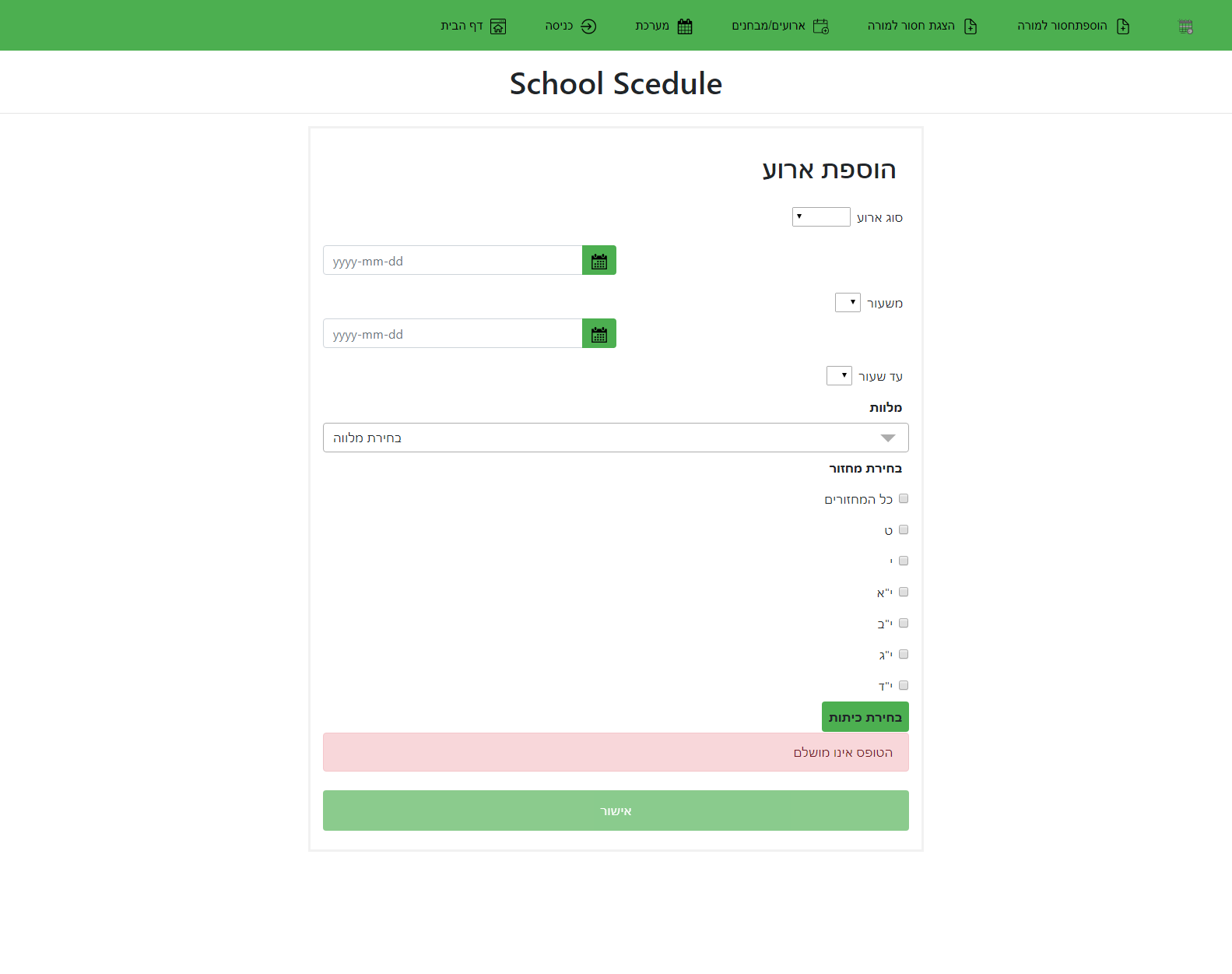
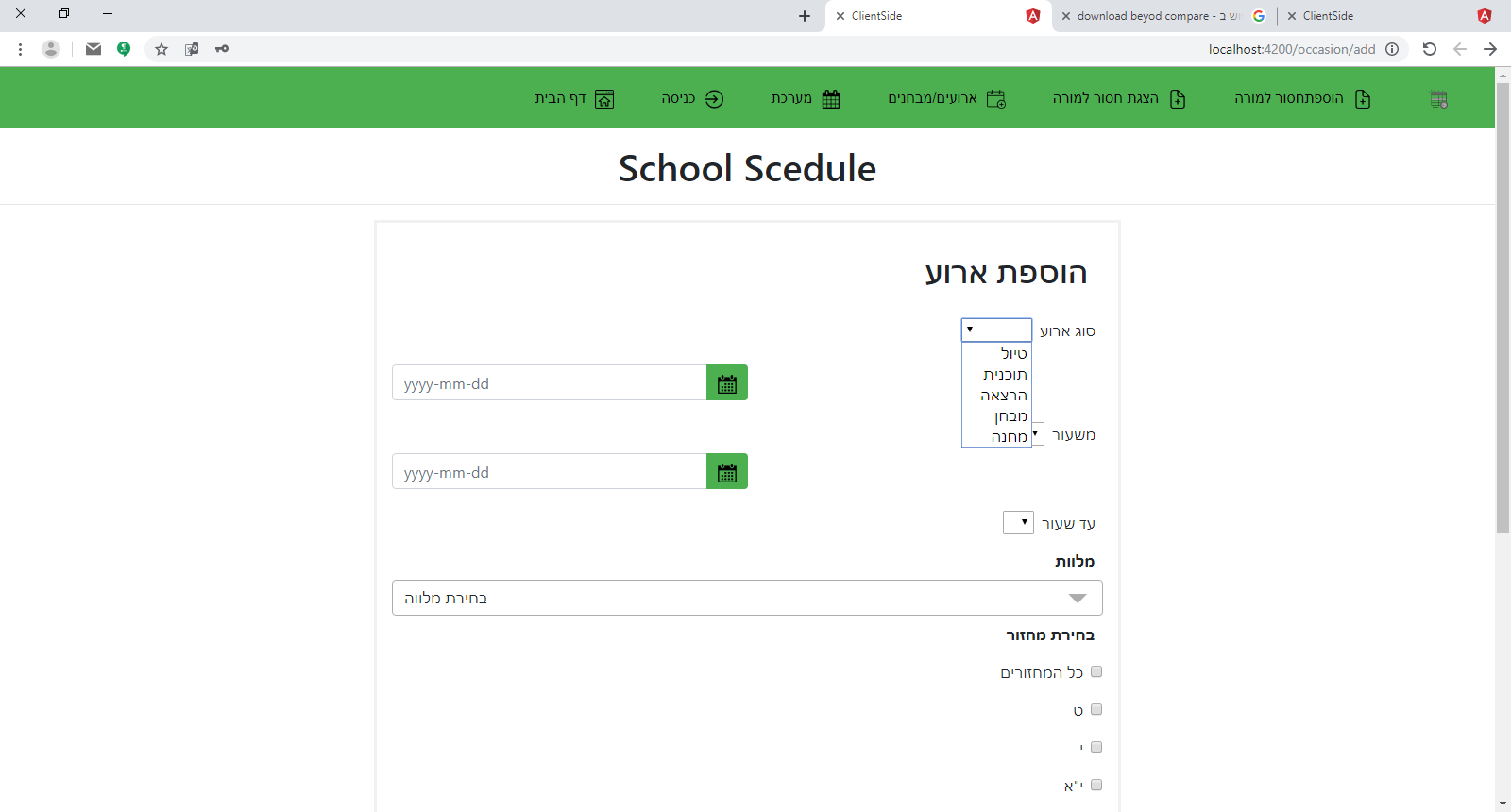
כל חודש יש לטעון את קבצי הנתונים.  
להכניס את הניתוב של הקבצים המתאימים.  
ללחוץ "טעינת קבצים ואתחול מערכת".  
  
לצורך כניסה למערכת נדרשת הכנסת סיסמה, יש אפשרות להיכנס כאורח עם הרשאות מוגבלות. כאשר מנסים ללחוץ על כפתור "כניסה" ללא הכנסת סיסמה, הודעה מתאימה מוצגת.   
אם הסיסמה שגויה הדף מתרענן, והמשתמש יכול להכניס סיסמה שוב.

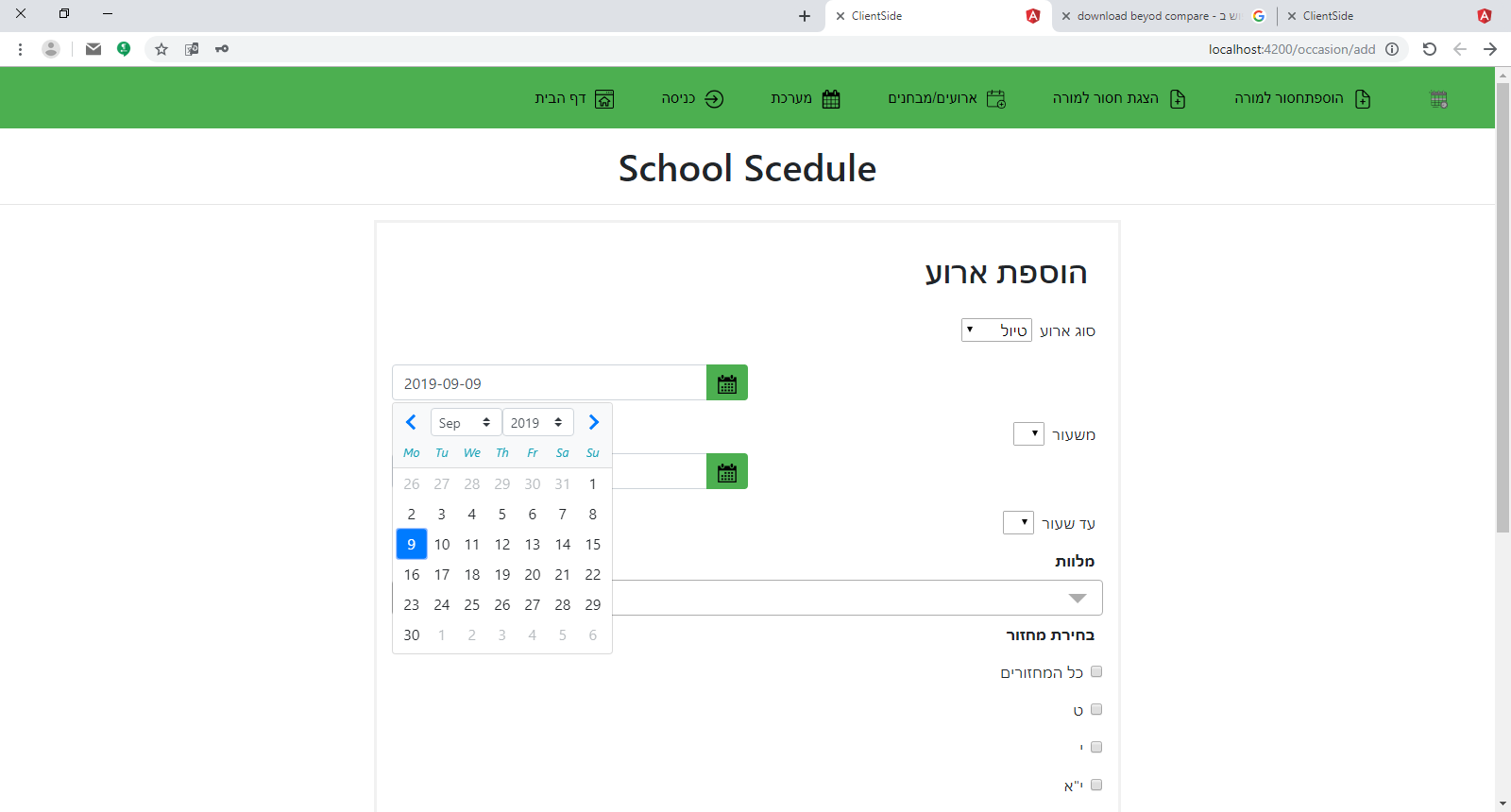
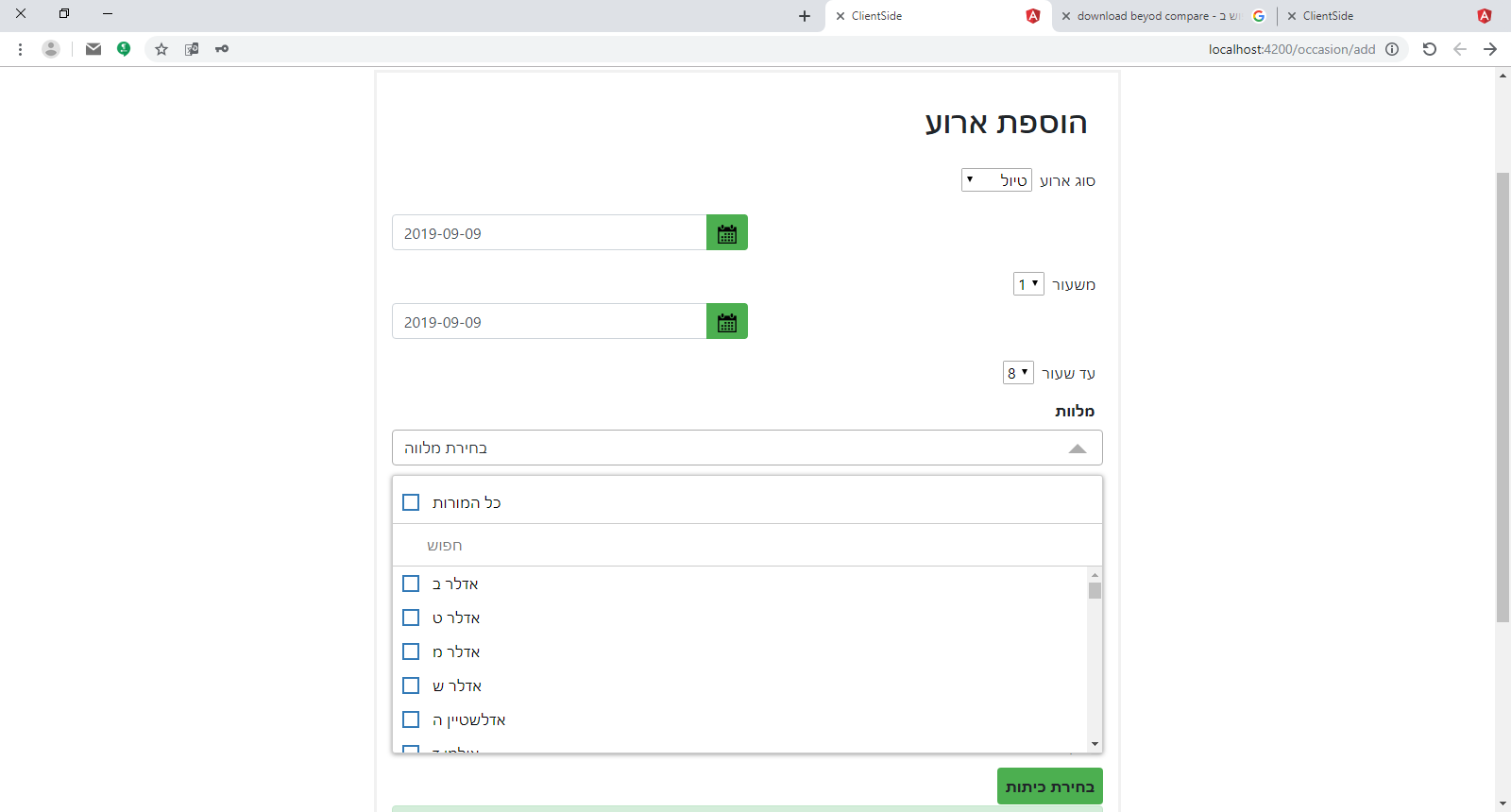
כאשר מורה נעדרת, יש למלא טופס הכולל את פרטי ההיעדרות.לבחור את המורה הנעדרת מרשימת המורות, זהו שדה חובה. 

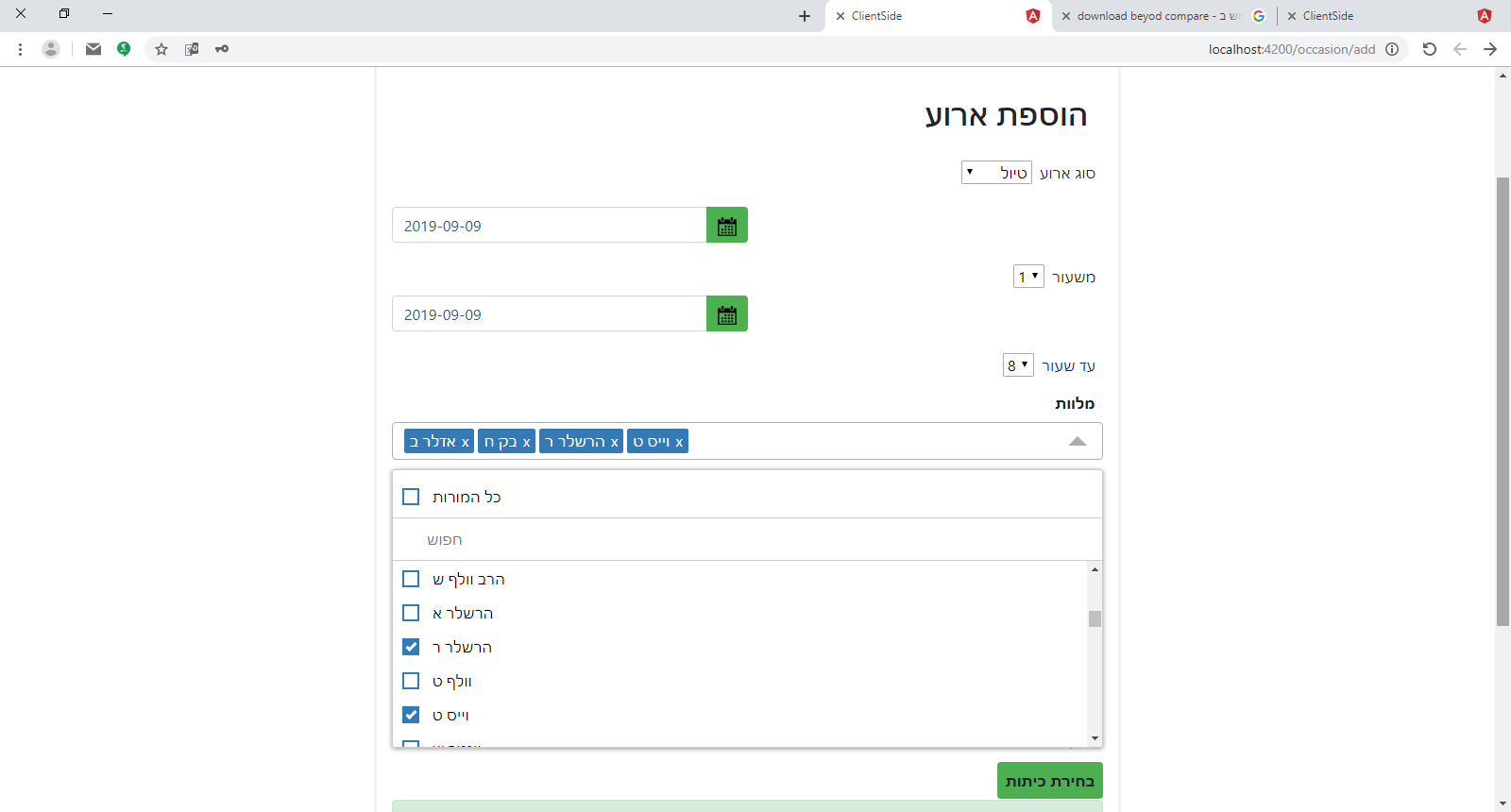
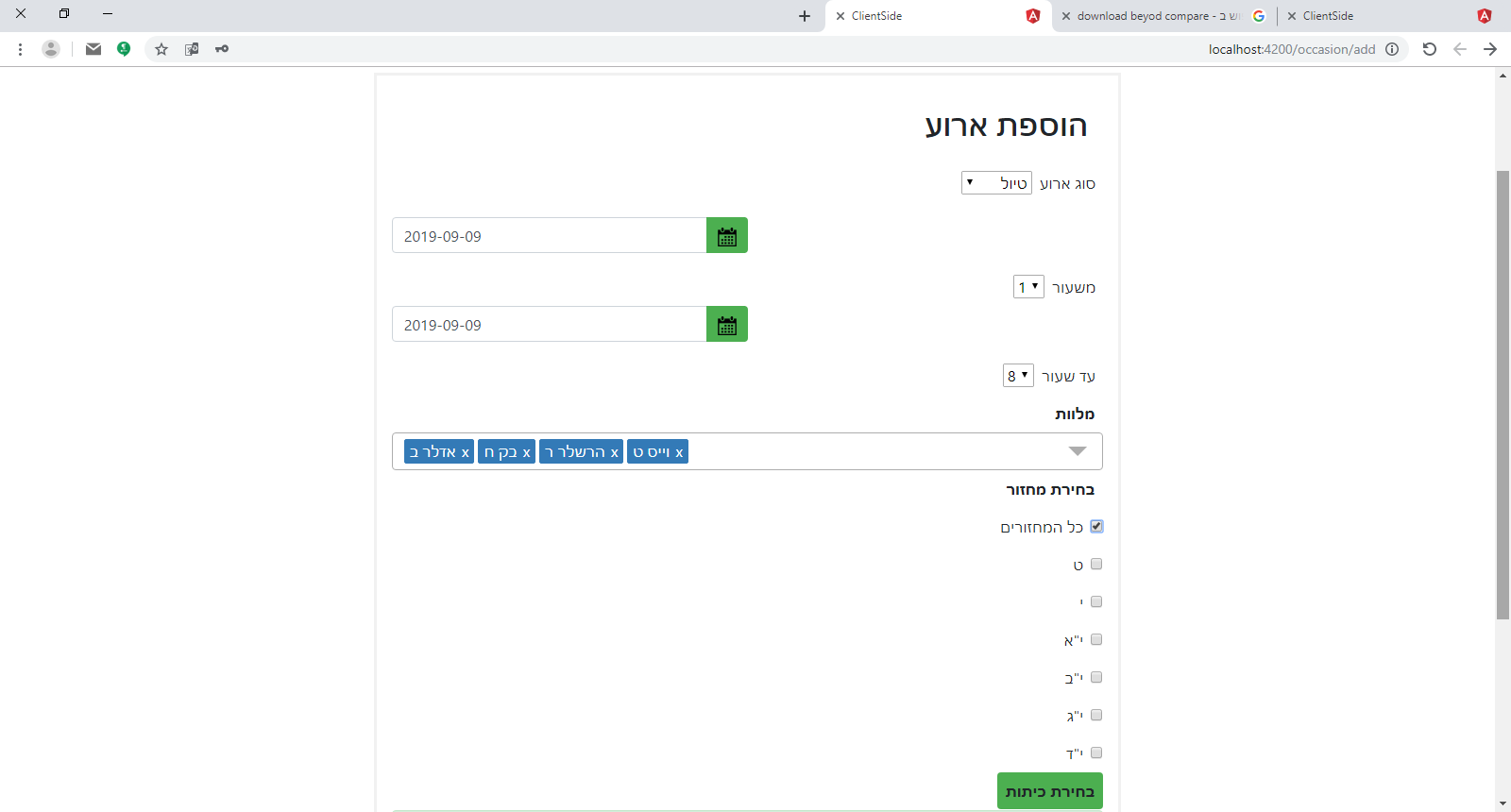
לבחור את תאריך תחילת ההיעדרות מתוך לוח השנה, זהו שדה חובה.  
לבחור את תאריך סיום ההיעדרות כנ"ל, זהו שדה רשות, אם לא ממלאים אותו הוא מתמלא בתאריך תחילת ההיעדרות.לבחור את סוג החיסור המתאים מתוך הרשימה.

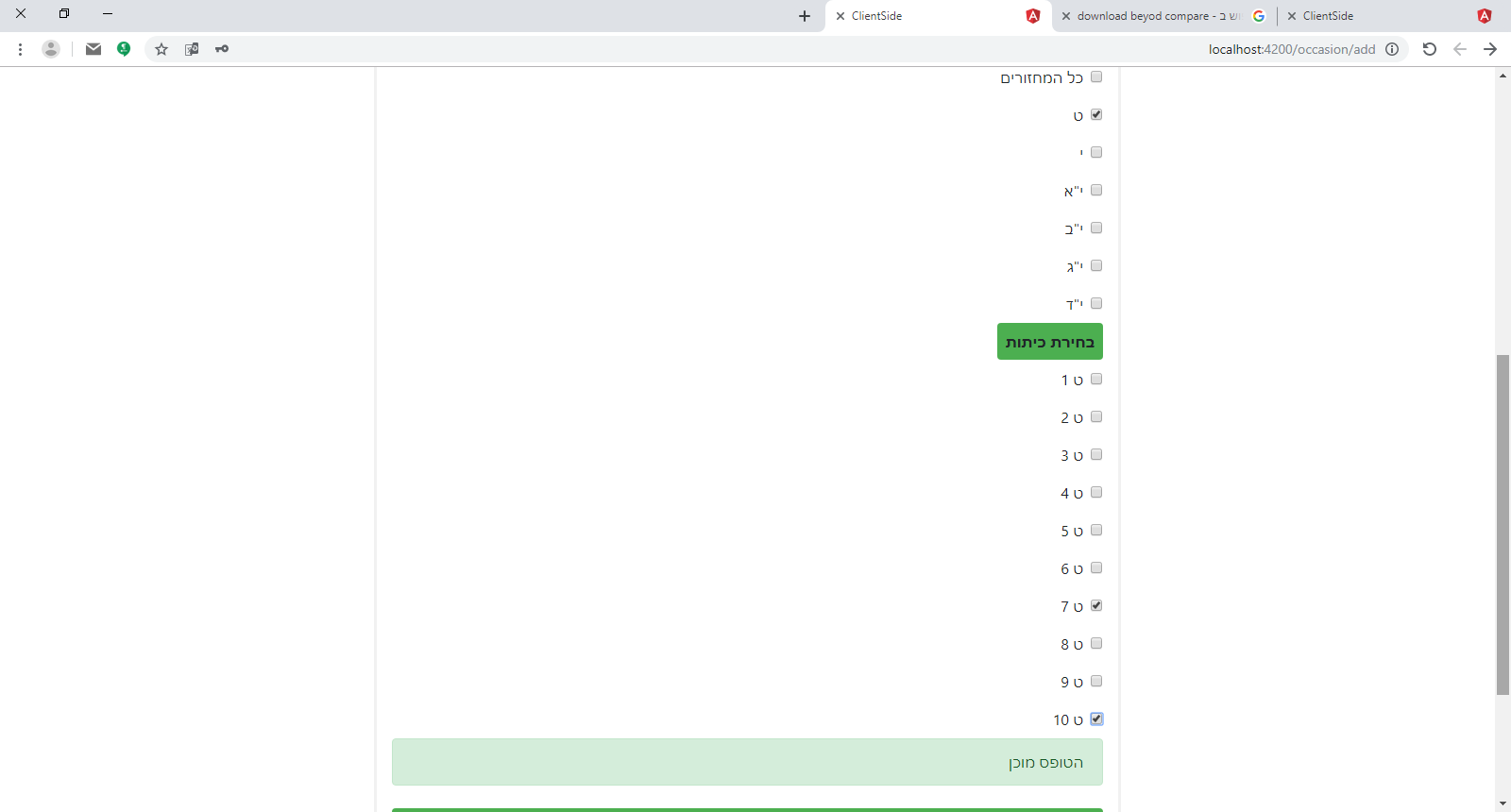
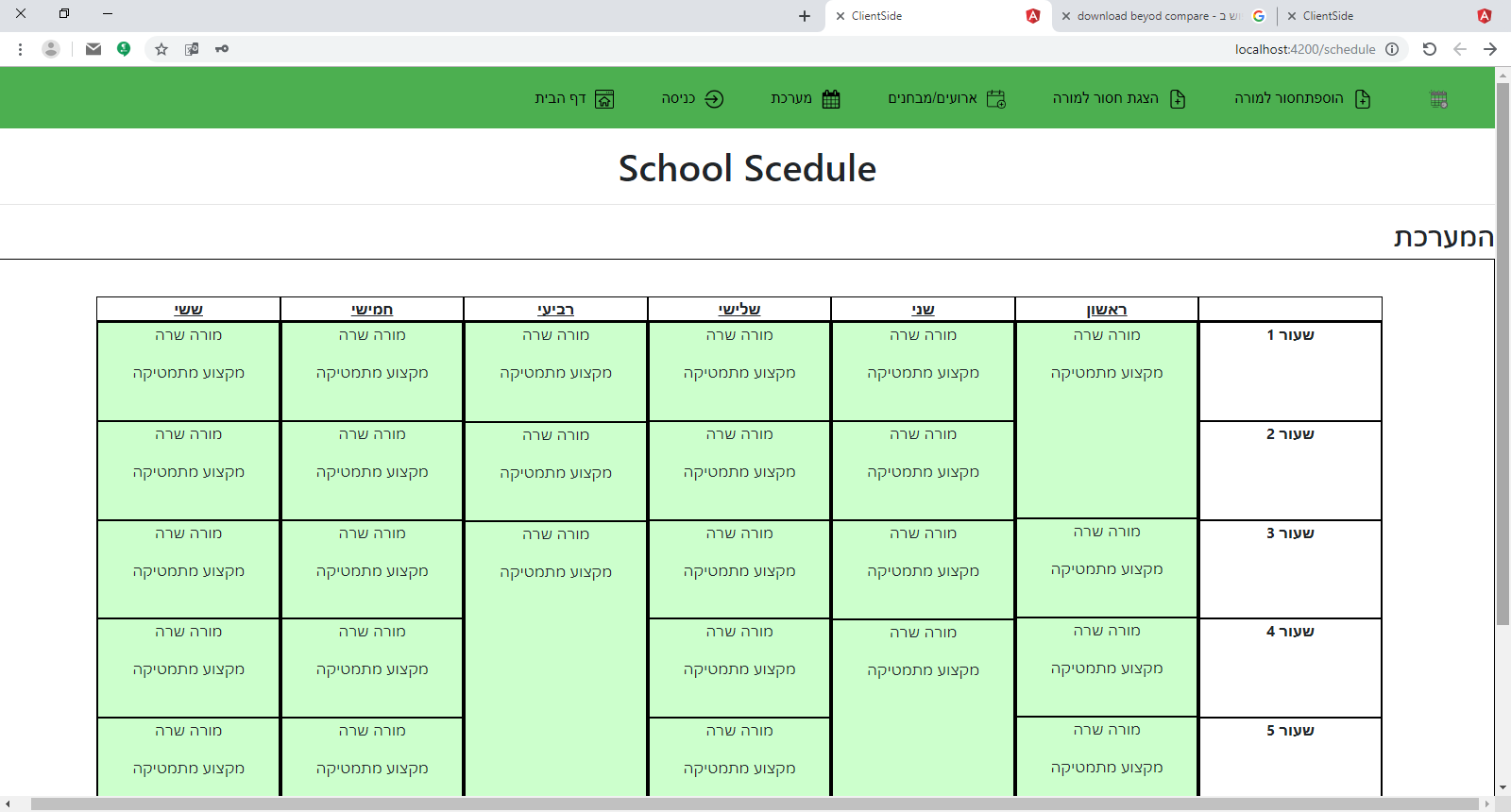
לבחור את שעור תחילת ההיעדרות מתוך הרשימה, זהו שדה חובה.  
לבחור את שעור סיום ההיעדרות כנ"ל, זהו שדה רשות, אם לא ממלאים אותו הוא מתמלא בשעור תחילת ההיעדרות. כאשר מסמנים את תיבת "טיפול יומי גורף", הכוונה שיש ממלאת מקום לכל השעות של המורה הנעדרת. ואז מתוסף שדה "ממלאת מקום" לבחירת מורה מרשימת המורות.

לסיום, ללחוץ על אישור. טופס עדכון היעדרות למורה, יש לעדכן את השדות הרצויים. וללחוץ "עדכון".

כאשר מוסיפים אירוע למערכת, יש למלא טופס הכולל את פרטי האירוע. כל עוד הטופס אינו תקין (שדות חובה לא מולאו וכדו'), תוצג האזהרה הבאה "הטופס אינו מושלם".  
ואזהרות ליד השדות הבעייתיים.  
לבחור את סוג האירוע המתאים מתוך הרשימה. 

לבחור את תאריך תחילת האירוע מתוך לוח השנה, זהו שדה חובה.  
לבחור את תאריך סיום האירוע כנ"ל, זהו שדה רשות, אם לא ממלאים אותו הוא מתמלא בתאריך תחילת האירוע.  לבחור את שעור תחילת ההיעדרות מתוך הרשימה, זהו שדה חובה.  
לבחור את שעור סיום ההיעדרות כנ"ל, זהו שדה רשות, אם לא ממלאים אותו הוא מתמלא בשעור תחילת ההיעדרות.  
  
לסמן את המלווה/ות מתוך הרשימה.  
ניתן לחפש בתיבת החיפוש המתאימה.  
ישנה אפשרות לסמן את כל המורות. 

המורות מתווספות/ מוסרות בהתאם לסימון. לסמן את המחזורים המתאימים.  
ללחוץ על כפתור "בחירת כיתות".

לסמן את הכיתות המתאימות.  
כעת הטופס מוכן לשליחה.תצוגת מערכת השעות.

# סיכום ומסקנות

לאחר תקופה שאנו עובדות ובונות נרצה להצביע על מספר הישגים ותוצאות וגם לחלוק  
במסקנות:  
נרצה לציין שפרויקט בסדר גודל כזה, נתן לנו המון ידע וניסיון בפיתוח הן בצד ה client והן  
בצד ה server. הפרויקט איחד את כל המושגים שלמדנו. וגרם לנו להבין את המושגים  
המופשטים יותר.  
הקמת פרויקט מרעיון כללי ועד לסיומו דורש המון זמן ידע והשקעה. בהחלט השקענו המון,  
ובזמן שניתן לנו הצלחנו לבנות ממשק נוח וברור המקיים את הפונקציונליות המורכבת של  
טיפול יעיל בפניות.  
כפי שניתן לראות, האתר נמצא בשלבי פיתוח מתקדמים, עדיין לא כל הקוד מופיע בספר, ולא  
כל הדפים מוכנים. סביר להניח שעם ההתקדמות והסיום, יעלו עוד שינויים, שיפורים ומסקנות  
חדשות.

# נספחים

## מנגנון אתחול מערכת

/// <summary>

/// this class take care of all the loadings of the XML files

/// </summary>

public class XmlManager

{

private Dictionary<string, Action<string>> fileNameToAction;

/// <summary>

/// each XML file has is his loading function.

/// </summary>

public XmlManager()

{

fileNameToAction = new Dictionary<string, Action<string>>();

fileNameToAction["classes.xml"] = LoadClasses;

fileNameToAction["groups.xml"] = LoadGroups;

fileNameToAction["rooms.xml"] = LoadRooms;

fileNameToAction["schedule.xml"] = LoadSchedule;

fileNameToAction["subjects.xml"] = LoadSubjects;

fileNameToAction["teachers.xml"] = LoadTeachers;

}

/// <summary>

/// this function load the directory of the xml files and for each file,invoke his loading function.

/// </summary>

/// <param name="path">the location of the XML files</param>

public void LoadDirectory(string path)

{

//reset the DB

ClearDB();

var directoryInfo = new DirectoryInfo(path);

var files = directoryInfo.GetFiles();

foreach (var file in files)

{

Action<string> action;

if (fileNameToAction.TryGetValue(file.Name, out action))

{

action?.Invoke(file.FullName);

Console.WriteLine("{0} Loading....", file.FullName);

}

}

}

/// <summary>

/// this function reset the DB

/// </summary>

public void ClearDB()

{

// var conn = new SqlConnection("data source=SQL-SERVER; initial catalog=!ESTY&TAMAR; integrated security=True");

var conn = new SqlConnection("data source=DESKTOP-7A0S24C; initial catalog=!ESTY&TAMAR; integrated security=True");

SqlCommand cmd;

conn.Open();

cmd = new SqlCommand("sp\_MSforeachtable 'DELETE FROM ?'", conn);

cmd.ExecuteNonQuery();

cmd = new SqlCommand("sp\_MSforeachtable 'truncate table ?'", conn);

cmd.ExecuteNonQuery();

conn.Close();

}

/// <summary>

/// classes loading

/// </summary>

/// <param name="path">xml file location</param>

public void LoadClasses(string path)

{

XmlClasses.root root = GetXmlData<XmlClasses.root>(path);

try

{

using (Entities db = new Entities())

{

foreach (XmlClasses.rootData rootData in root.data)

{

db.Classes.Add(\_CastDTO.DTOToClass(new ClassDTO()

{

Layer = rootData.layer,

Name = rootData.name,

Num = rootData.num,

Number = rootData.class\_number,

SchoolType = rootData.schooltype

}));

}

db.SaveChanges();

}

}

catch (Exception e)

{

LogManager.LogException(e);

}

}

/// <summary>

///groups loading

/// </summary>

/// <param name="path">xml file location</param>

public void LoadGroups(string path)

{

XmlGroups.root root = GetXmlData<XmlGroups.root>(path);

try

{

using (Entities db = new Entities())

{

foreach (XmlGroups.rootGroup rootData in root.group)

{

db.Groups.Add(\_CastDTO.DTOToGroup(new GroupDTO()

{

Num = rootData.num,

Teacher = rootData.tea,

Subject = rootData.subj,

Room = rootData.sroom,

Hours = rootData.no\_of\_hours,

CalculateHours = rootData.calculate\_hours,

SchoolType = rootData.schooltype,

Reforma = rootData.reforma,

PayAbsence = rootData.goremMeshalem,

HourType = rootData.sug,

SubHourType = rootData.sub\_sug

}));

}

db.SaveChanges();

}

}

catch (Exception e)

{

LogManager.LogException(e);

}

}

/// <summary>

/// rooms loading

/// </summary>

/// <param name="path">xml file location</param>

public void LoadRooms(string path)

{

XmlRooms.root root = GetXmlData<XmlRooms.root>(path);

try

{

using (Entities db = new Entities())

{

foreach (XmlRooms.rootData rootData in root.data)

{

var roomDataNum = rootData.num.ToString();

db.Rooms.Add(\_CastDTO.DTOToRoom(new RoomDTO()

{

ClassId = db.Classes.Where(c => c.Name == rootData.name).FirstOrDefault()?.Id,

Floor = int.Parse(roomDataNum[2].ToString()),

Number = int.Parse(roomDataNum[2].ToString()) \* 100 + int.Parse(roomDataNum[3].ToString()),

UseFor = rootData.name

}));

}

db.SaveChanges();

}

}

catch (Exception e)

{

LogManager.LogException(e);

}

}

/// <summary>

///schedule loading

/// </summary>

/// <param name="path">xml file location</param>

public void LoadSchedule(string path)

{

XmlSchedule.root root = GetXmlData<XmlSchedule.root>(path);

try

{

using (Entities db = new Entities())

{

foreach (XmlSchedule.rootGroup rootData in root.group)

{

//TODO

foreach (XmlSchedule.rootGroupSchedule rootGroupSchedule in rootData.schedule)

{

db.Schedules.Add(\_CastDTO.DTOToSchedule(new ScheduleDTO()

{

Day = rootGroupSchedule.day,

Hour = rootGroupSchedule.hour,

WeekDay = rootGroupSchedule.weekDay,

Room = rootGroupSchedule.weekDay

}));

}

}

db.SaveChanges();

}

}

catch (Exception e)

{

LogManager.LogException(e);

}

}

/// <summary>

///subjects loading

/// </summary>

/// <param name="path">xml file location</param>

public void LoadSubjects(string path)

{

XmlSubjects.root root = GetXmlData<XmlSubjects.root>(path);

try

{

using (Entities db = new Entities())

{

foreach (XmlSubjects.rootData rootData in root.data)

{

db.Subjects.Add(\_CastDTO.DTOToSubject(new SubjectDTO()

{

Id = rootData.num,

Name = rootData.name

}));

}

db.SaveChanges();

}

}

catch (Exception e)

{

LogManager.LogException(e);

}

}

/// <summary>

///teacher loading

/// </summary>

/// <param name="path">xml file location</param>

public void LoadTeachers(string path)

{

XmlTeachers.root root = GetXmlData<XmlTeachers.root>(path);

try

{

using (Entities db = new Entities())

{

foreach (XmlTeachers.rootData rootData in root.data)

{

db.Teachers.Add(\_CastDTO.DTOToTeacher(new TeacherDTO()

{

Num = rootData.num,

Name = rootData.name,

}));

}

db.SaveChanges();

}

}

catch (Exception e)

{

LogManager.LogException(e);

}

}

/// <summary>

/// read xml file

/// and convert the text to object

/// </summary>

/// <typeparam name="T"></typeparam>

/// <param name="path"></param>

/// <returns>root xml object</returns>

private static T GetXmlData<T>(string path)

{

XmlSerializer serializer = new XmlSerializer(typeof(T));

StringReader stringReader = new StringReader(File.ReadAllText(path, Encoding.GetEncoding(1255)));

T root = (T)serializer.Deserialize(stringReader);

return root;

}

}

## מנגנון טיפול בשגיאות

/// <summary>

/// this class take care of exceptions

/// </summary>

public class LogManager

{

/// <summary>

/// write logs

/// </summary>

/// <param name="e"></param>

public static void LogException(Exception e)

{

// File.AppendAllText("C:\\workspace\\logs.txt", GetInnerExceptionMessage(e));//our file in bin/debug folder

using (EventLog eventLog = new EventLog())

{

eventLog.Source = "SchoolSchedule";

eventLog.WriteEntry(GetInnerExceptionMessage(e), EventLogEntryType.Error);

} /\*\*/

}

/// <summary>

/// find all the inner exception messages

/// </summary>

/// <param name="e"></param>

/// <returns>all the inner exception messages</returns>

public static string GetInnerExceptionMessage(Exception e)

{

if (e.InnerException != null)

{

return e.Message + "\n" + GetInnerExceptionMessage(e.InnerException);

}

return e.Message + "\n" + e.StackTrace + "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

}

}

## XML Classes

public class XmlClasses

{

// NOTE: Generated code may require at least .NET Framework 4.5 or .NET Core/Standard 2.0.

/// <remarks/>

[System.SerializableAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

[System.Xml.Serialization.XmlTypeAttribute(AnonymousType = true)]

[System.Xml.Serialization.XmlRootAttribute(Namespace = "", IsNullable = false)]

public partial class root

{

public rootData[] dataField;

public uint school\_idField;

public string data\_typeField;

public string eG\_time2Field;

/// <remarks/>

[System.Xml.Serialization.XmlElementAttribute("data")]

public rootData[] data

{

get

{

return this.dataField;

}

set

{

this.dataField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public uint school\_id

{

get

{

return this.school\_idField;

}

set

{

this.school\_idField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public string data\_type

{

get

{

return this.data\_typeField;

}

set

{

this.data\_typeField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public string EG\_time2

{

get

{

return this.eG\_time2Field;

}

set

{

this.eG\_time2Field = value;

}

}

}

/// <remarks/>

[System.SerializableAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

[System.Xml.Serialization.XmlTypeAttribute(AnonymousType = true)]

public partial class rootData

{

public string nameField;

public byte numField;

public byte mapField;

public ushort idField;

public byte class\_numberField;

public byte layerField;

public byte reformaField;

public byte schooltypeField;

public ushort backColorField;

public uint foreColorField;

/// <remarks/>

public string name

{

get

{

return this.nameField;

}

set

{

this.nameField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public byte num

{

get

{

return this.numField;

}

set

{

this.numField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public byte map

{

get

{

return this.mapField;

}

set

{

this.mapField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public ushort id

{

get

{

return this.idField;

}

set

{

this.idField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public byte class\_number

{

get

{

return this.class\_numberField;

}

set

{

this.class\_numberField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public byte layer

{

get

{

return this.layerField;

}

set

{

this.layerField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public byte reforma

{

get

{

return this.reformaField;

}

set

{

this.reformaField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public byte schooltype

{

get

{

return this.schooltypeField;

}

set

{

this.schooltypeField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public ushort BackColor

{

get

{

return this.backColorField;

}

set

{

this.backColorField = value;

}

}

/// <remarks/>

[System.Xml.Serialization.XmlAttributeAttribute()]

public uint ForeColor

{

get

{

return this.foreColorField;

}

set

{

this.foreColorField = value;

}

}

}

}

# ביבליוגרפיה

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

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

[https://angular.io](https://angular.io/)

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

[https://www.w3.org](https://www.w3.org/)

[https://scotch.io](https://scotch.io/)

[https://codepen.io](https://codepen.io/)

[https://en.wikipedia.org](https://en.wikipedia.org/)

[https://confluence.atlassian.com](https://confluence.atlassian.com/)

[https://nodejs.org](https://nodejs.org/)

[https://update.angular.io](https://update.angular.io/)

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

[https://ng-bootstrap.github.io](https://ng-bootstrap.github.io/)

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

[https://love2dev.com](https://love2dev.com/)

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

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

[https://regex101.com](https://regex101.com/)

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

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

<https://www.hamichlol.org.il>

http://www.tzurel.co.il/tag

[https://www.geektime.co.il](https://www.geektime.co.il/microsoft-snip-will-upgrade-your-screenshots/)

<https://support.office.com>

[https://www.maketecheasier.com](https://www.maketecheasier.com/capture-scrolling-screenshot-windows/)

<https://blogs.msdn.microsoft.com>

<https://picpick.app/en/>