

## פרויקט גמר לתואר הנדסאי מחשבים



מגישות: דבורה קיסטר ודינה יעקובוביץ  
מגמת הנדסאת תוכנה  
שם המנחה: גב' מ. שמעוןוביץ  
תאריך ההגשה : נובמבר 2019

**מהט**  
המכון הממשלתי להכשרה  
בטכנולוגיה ובמדע



שמות המגישות: דינה יעקובוביץ ודבורה קיסטר

חתימה: דינה יעקובוביץ  
חתימה: דבורה קיסטר

שם המנחה: גב' מ. שמעוןוביץ  
חתימה: \_\_\_\_\_

שם רכזת המגמה: גב' ח. ברגמן  
חתימה: \_\_\_\_\_

נובמבר 2019

## המכון הממשלתי להכשרה בטכנולוגיה ובמדע **מחשבים** הפרויקטים

תאריך: 29/05/2019

לכבוד: יחידת הפרויקטים מהיט

### הצעה לפרויקט גמר

#### א. פרטי הסטודנטים

שם הסטודנט	ת.ז. 9 ספרות	כתובת	טלפון נייד	שנת סיום הלימודים
דבורה קיסטר	207085580	נויפלד 8	0583235540	2019
דינה יעקובוביץ	318993532	חזון איש 58 א	0548597054	2019

שם המכללה: סמינר וולף – שלוחת המכללה למנהל ראש"צ

סמל המכללה: 72395

מסלול ההכשרה: הנדסאים

מגמת לימוד: תכנות מחשבים

מקום ביצוע הפרויקט: בסמינר

#### ב. פרטי המנחה האישי

שם המנחה	כתובת	טלפון נייד	תואר	מקום עבודה/תפקיד
מרים שמעוןוביץ	חזון"א 11 ב"ב	052-7171295	DB. E הנדסאי מחשבים	המכללה למנהל

חתימת הגורם המקצועי מטעם מהיט

חתימת המנחה האישי

חתימת הסטודנט

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

## 1. שם הפרויקט:

Ideal

## 2. רקע

### 2.1. תיאור כללי

הפרויקט נכתב לארגון 'אידיאל' העוסק בהשמת עובדות חרדיות במקומות עבודה המתאימים לאישה החרדית. הפרויקט הינו אתר דרושים המאפשר פרסום וחיפוש משרות בהתאמה אישית לפי קריטריונים לבחירה. ובנוסף המערכת כוללת טכנולוגית "סוכן חכם" השולחת למיל האישי של מחפש המשרה משרות חדשות המתאימות לו. כמו כן האתר כולל פורום שאלות ותשובות בהלכה בנושא מקומות עבודה, ועוד.

### 2.2. מטרת המערכת

- המערכת תספק מידע על מקומות העבודה הן מהבחינה הטכנית והן מבחינה רוחנית על סביבת העבודה, סוג חסימת רשת ועוד.
- האתר יאפשר למחפש עבודה להתעדכן במשרות חדשות במשק.
- לאפשר לעובדת להמליץ למעסיקה על מועמדות מתאימות למשרה שברצונם לאישי
- לאפשר לעובדים לשאול ולהתייעץ בענייני הלכה בפורום פתוח לציבור.
- לתת גב ותמיכה לעובדות חרדיות שחפצות לעבוד במקומות שמורים בלבד.

## 3. סקירת מצב קיים בשוק

קיימים אתרי דרושים אך הם לא מספקים מידע מקיף מבחינה רוחנית על העבודה. (פעילות זו מתבצעת כיום ע"י הארגון בצורה ידנית על ידי קבצי WORD, XSL ועוד).

## 4. מה הפרויקט אמור לחדש או לשפר

הפרויקט יציג על המשרות המתפרסמות את סביבת העבודה וסוג חסימה של האינטרנט, ובכך יקל על ברור אודות מקום העבודה המוצע.

## 5. דרישות מערכת ופונקציונאליות

### 5.1. דרישות מערכת, סביבת הטמעה ושימוש.

- המערכת תעבור קומפילציה והפצה בסביבת Visual Studio
  - בצד השרת המערכת אמורה לרוץ בסביבת שרת אשר מריץ IIS לקבלת בקשות לנתונים.
  - לקוח יכול להכנס למערכת מכל התקן המריץ דפדפן אינטרנט.
- ### 5.2. שרידות, ביצועים, התמודדות עם עומסים
- צד השרת מריץ את IIS Express המסוגל להתמודד עם מספר קריאות בו זמנית.

- גם עומס על שרת ה-SQL אינו צפוי בסדר גודל כזה של אתר.
- מסד הנתונים יותקן בשרת אשר מכיל גיבוי אוטומטי לנתונים בכל פרק זמן קצר.
- למורך ביצועים מיטביים נדאג לרכז קריאות לשרת האפליקציה ולשרת הנתונים, בקריאות בודדות ובמינימום תעבורת נתונים.

### 5.3. דרישות פונקציונאליות

5.3.1. בדף ההרשמה תהיה למשתמש אפשרות להיכנס בתור מפרסם משרה או מחפש משרה.

5.3.2. הנרשם למערכת נכנס להגרלה חודשית.

#### רשימת דרישות משתמש מחפש משרה למערכת:

- 5.3.3. משתמש יוכל לצפות ברשימת דרושים הכוללת את כל הצעות העבודה במאגר.
- 5.3.4. מאפשר למשתמש חדש להירשם למערכת, הוא נדרש להכניס את פרטיו האישיים ולבחור שם משתמש וסיסמא ולהזין קובץ קו"ח
- 5.3.5. המערכת מאפשרת למשתמש לעדכן פרטי חיפוש הכוללים תחום, אזור וחלקי משרה, ומציגה בפניו טבלת המשרות הרלוונטיות לגביו.
- 5.3.6. המשתמש רשום יכול להעלות קובץ קו"ח חדש.
- 5.3.7. המשתמש רשום יכול לשלוח שאלה לרב ולהגדיר את אופן קבלת התשובה.
- 5.3.8. משתמש רשום יכול להירשם למנוי 'סוכן חכם' שבו מקבל אליו עדכונים של משרות רלוונטיות לגביו ע"פ הנתונים שבמערכת.
- 5.3.9. המערכת מאפשרת שינוי פרטים אישיים ומעדכנת על פיהם את הסוכן החכם.
- 5.3.10. משתמש רשום יוכל לחוות לדעתו על מקומות עבודה וטבען, ע"י בחירת מקום העבודה מרשימה
- 5.3.11. משתמש יוכל להירשם למשרה.
- 5.3.12. משתמש יוכל לצפות במאמרי חיזוק דרך האתר.
- 5.3.13. משתמש יוכל לצפות בפרסומות דרך האתר.
- 5.3.14. משתמש יוכל לשלוח שאלה הלכתית לרב, הוא יצרך לשאלה פרטי התקשרות לקבלת מענה.
- 5.3.15. משתמשים יוכלו לצפות בפורום שאלות ותשובות הלכתיות באתר.
- 5.3.16. משתמש יוכל לשלוח תגובה למערכת.

#### דרישות משתמש מציע משרה למערכת:

- 5.3.17. מאפשר למשתמש חדש להירשם למערכת, הוא נדרש להכניס את פרטי העסק ולבחור שם משתמש וסיסמא.
- 5.3.18. מאפשר למפרסם לפרסם משרה חדשה במערכת.
- 5.3.19. מאפשר למפרסם משרה לעדכן את פרטי המשרה שלו (מידע, סוג חסימה, דרישות...).

5.3.20. המפרסם מקבל קו"ח של הנרשמים למשרה ששיתף.

#### דרישות מנהלי מערכת:

- 5.3.21. מנהלי המערכת יוכלו להפעיל פקודה המייבאת לאתר פרסומות, מידע ומאמרים.
- 5.3.22. מנהלי המערכת יוכלו לערוך את כל המשרות (סטטוס משרה).
- 5.3.23. מנהלי המערכת יצפו בכל המשתמשים הרשומים באתר.
- 5.3.24. מנהל יוכל לצפות בשליחת שאלה לרב ולעדכן שאלה ותשובה בפורום.
- 5.3.25. מנהל יוכל לצפות ברשימת נרשמים למרות ולשלוח קו"ח למציע המשרה, ע"פ בחירתו המערכת תשנה את סטטוס הנרשם לעבודה ל"הוגשו קו"ח".

## 6. בעיות צפויות במהלך הפיתוח

- 6.1. הבעיות- הללו כפועל יוצא של דרישות המשתמש מהתוכנה.
  - בעיה 1: משתמש המחפש משרה העלה קורות חיים וברצונו לעדכןם.
  - בעיה 2: משתמש שיתף משרה והיא כבר לא רלוונטית.
- 6.2. פתרונות אפשריים:
  - 6.2.1. לבעיה 1:
    - פתרון 1:** לאפשר למשתמש להעלות קורות חיים חדשים והמערכת תמחק את הקבצים בעלי התאריך הישן יותר.
    - פתרון 2:** המערכת תאפשר לערוך את המסמך באתר.
  - 6.2.2. לבעיה 2:
    - פתרון 1: רק למנהל המערכת יהיה הרשאות מחיקה ועל כן המפרסם יצטרך לידע אותו על כך.
    - פתרון 2: לאפשר למפרסם למחוק את המשרה כשהוא גולש תחת שם המשתמש שבו פרסם אותה.
- 6.3. הפתרון הנבחר עבור כל אחת מהבעיות:
  - 6.3.1. לבעיה 1: פתרון 1. מכיוון שברצונו לאפשר למשתמש לערוך את הקו"ח בסביבה בה ערך זאת לראשונה.
  - 6.3.2. לבעיה 2: פתרון 2, ע"מ שהמערכת לא תהיה תלויה בגורמים חיצוניים.

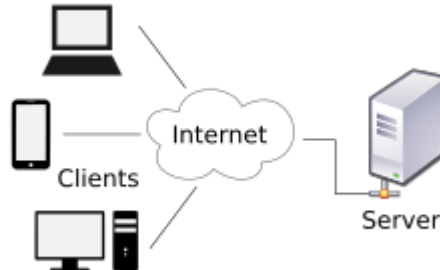
## 7. פתרון טכנולוגי נבחר

### 7.1. טופולוגית הפתרון

- 7.1.1. טופולוגית הפתרון: המערכת מורכבת משרת IIS המריץ את API-WEB בסביבת ה-server,
- 7.1.2. אתר בסביבת אנגולר

7.1.3. ממשק משתמש בצד הלקוח: דפדפן אינטרנט כלשהו: chrome, firefox, internet explorer

7.1.4. דיאגרמה



7.1.5. טכנולוגיות בשימוש:

כתיבת צד הלקוח: Angular6 + TypeScript  
בשיתוף bootstrap4 ליצירת אתר רספונסיבי, מעוצב ונעים לעין.  
כתיבת צד השרת: שירות אינטרנט באמצעות WEB API  
מסד נתונים באמצעות Sql-Server

7.1.6. שפות הפיתוח:

7.1.7. בצד השרת:

#C

7.1.8. בצד הלקוח:

TYPE SCRIPT

JAVA SCRIPT

CSS

HTML5

7.2. תיאור הארכיטקטורה הנבחרת

צד לקוח: טכנולוגית ה angular מאורגנת בצורה מסודרת ומונחת עצמים. היא עדכנית, מתקדמת ומבוקשת מאד בתחום ההייטק.  
צד שרת: פיתוח ב#C שהיא שפה נוחה וידידותית לשימוש.

7.3. חלוקה לתכניות ומודולים:

7.3.1. בצד השרת:

BL – לוגיקה

DAL – פונקציות הגישה למסדי הנתונים

Entities - ישויות המוגדרות מול מסד הנתונים.

WEB-API -, תקשורת בין השכבות לאנגולר.

מסד נתונים-פרוצדורות.

7.3.2. בצד הלקוח:

Component - Login-register, jobTable, layout, numbers, smart-agent, adv, connect, information, about etc.  
 Service - מחלקה שמספקת מידע לשאר המחלקות.  
 Model - ישויות.  
 Module - ייבוא קבצים והגדרת המערכת.

## 8. סביבת השרת

8.1. ממשק המשתמש/לקוח – **GUI** ממשק המשתמש יתקבל כ- **HTML** ויהיה מוצג באמצעות דפדפן אינטרנט.

8.2. ממשקים למערכות אחרות / **API**

התממשקות לשרת ה-gmail לשליחת מיילים של google.

8.3. שימוש בחבילות תוכנה

- angularJS
- angularCore
- Bootstrap
- Entity framework

## 9. מבני נתונים וארגון קבצים

9.1. שיטת האחסון

הנתונים ישמרו במסד נתונים של sql-server.

9.2. מבני הנתונים

9.2.1. אזורים קוד, שם

9.2.2. משתפי משרה קוד, שם, סיסמא, טלפון, קוד עיר, מייל, רחוב, מספר כוכבים, קוד תחום-משרה

9.2.3. ערים קוד, שם, קוד אזור

9.2.4. קורות חיים קוד, קישור, קוד משתמש

9.2.5. משרה קוד, תאריך הצעה, קוד אזור, קוד עיר, קוד תחום-משרה, תפקיד, קוד כמות שעות,



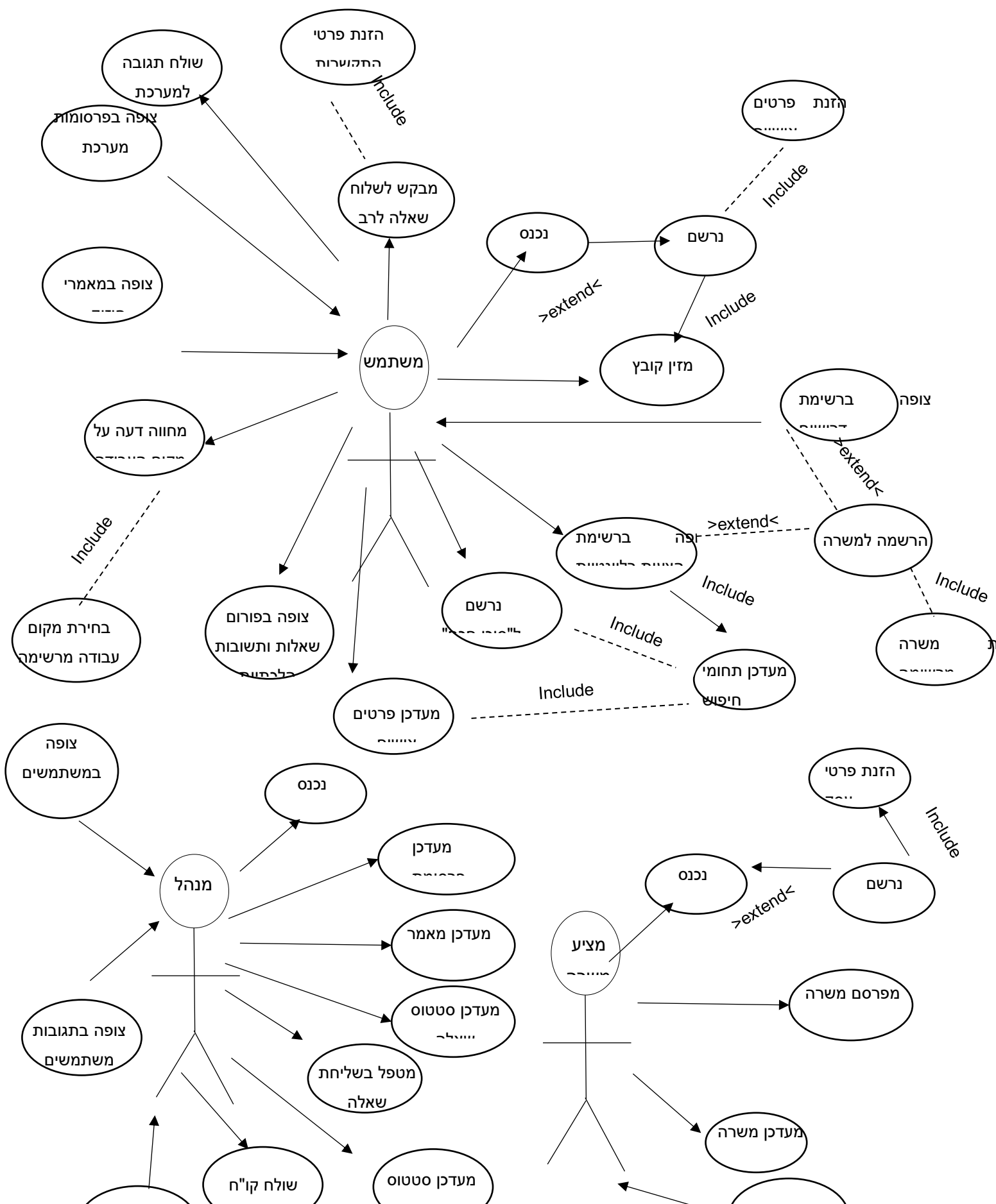
קוד סוג- חסימה, קוד סביבת-עבודה, שנות ניסיון, קוד קו"ח, קוד מציע, מספר כוכבים, קוד משתף-

משרה, סטטוס

- 9.2.6. חסימה קוד, שם
  - 9.2.7. מספר שעות-עבודה קוד, שם
  - 9.2.8. השמות קוד, קוד משרה, קוד משתמש, תאריך השמה
  - 9.2.9. שאלות לרב קוד, קוד משתמש, שאלה, תשובה, קוד רב, קוד תחום-שאלה
  - 9.2.10. רבנים קוד, שם
  - 9.2.11. המלצות קוד, קוד משתמש, קוד משרה, מידע
  - 9.2.12. תגובה מהמערכת קוד, קוד משתמש, תוכן התגובה, אמצעות הקשר
  - 9.2.13. תחומי משרות קוד, שם
  - 9.2.14. נושא השאלות-לרב קוד, שם
  - 9.2.15. אמצעות תקשורת קוד, שם
  - 9.2.16. משתמש קוד, שם, טלפון, מייל, קוד עיר, קוד תחום משרה, קוד מספר שעות-עבודה, סיסמא
  - 9.2.17. סביבת עבודה קוד, שם
  - 9.2.18. נרשם לעבודה קוד, קוד משרה, קוד משתמש, תאריך, סטטוס
- 9.3. מנגנוני התאוששות מנפילה/ קריסה/ תמיכה בטראנזקציות.
- צד השרת מריץ את IIS Express המסוגל להתמודד עם מספר קריאות בו זמנית.
  - גם עומס על שרת ה SQL אינו צפוי בסדר גודל כזה של אתר.

## 10. תרשימי מערכת מרכזיים

## Use Case .10.1



## 1.1. תיאור המרכיב האלגוריתמי – חישובי

11.1. איזה בעיה בא לפתור, איך יפתור?

11.2. איסוף מידע וניתוחים סטטיסטיים (אנליטיקות)

ניתן לבצע סטטיסטיקה על כמות האנשים שפנו לחברות דרך האתר והושמו בעבודה, הסטטיסטיקה תוכל גם להציג את הנתונים ע"פ האזורים, תחומים.

## 1.2. תיאור/התייחסות לנושאי אבטחת מידע

יש לדאוג לאבטחת השרת מעומסים מופרזים הנגרמים יל ידי גורמים זדוניים, ואת שרת

. sql injection מ - server sql-

משתמש הגולש באתר מריץ אותו באופן אבסולוטי. קוד המשתמש ישלח לשרת ויצטרף לכל רשומה הנשמרת עבורו. לא יתכן מצב בו משתמש מריץ אפליקציה בתור משתמש אחר.

**דוגמאות:**

- במקרה שמשתמש חדש מנסה להתחבר כמשתמש רשום, המערכת תפנה אותו לדף ההרשמה באתר.
- במקרה שבעת כניסת משתמש הסיסמא אינה תואמת לשם המשתמש שהקיש, המערכת תציג הודעת שגיאה ולא תאפשר כניסה.
- הסיסמא תהיה מוסתרת.

## 1.3. משאבים הנדרשים לפרויקט:

13.1. מספר שעות המוקדש לפרויקט: 720

13.2. חלוקת עבודה בין חברי הצוות:

עבודת צוות משותפת, בשעת הצורך - חלוקת המשימות בצורה שווה והוגנת .

13.3. ציוד נדרש

מחשב הכולל חיבור לאינטרנט CPU i5, RAM 8GB, SSD HD, WIN10.

13.4. תוכנות נדרשות

Visual Studio, Visual Studio Code, Sql Server Ssms

13.5. ידע חדש שנדרש ללמוד לצורך ביצוע הפרויקט

לימוד angular, gmail api ועוד.

13.6. ספרות ומקורות מידע

<https://getbootstrap.com>

<https://reshetech.co.il/javascript>

<https://github.com/>

<https://stackoverflow.com/>

<https://w3schools.com/>

<https://www.flaticon.com/categories>

<https://developers.google.com/gmail/api/quickstart/dotnet>

## 14. תכנית עבודה ושלבים למימוש הפרויקט:

שלב	משך זמן משוער	תאריכים משוערים
ייזום הרעיון	שבועיים	1/3/2019
ניתוח מערכת	שבועיים	15/3/2019
ניתוח מבנה נתונים	שבועיים	1/4/2019
איפיון UX - UI	שבועיים	15/4/2019
כתיבת הלוגיקה העסקית	חודשיים	7/2019 – 5/2019
כתיבת ממשק המשתמש	חודש וחצי	8/2019 – 7/2019
עיצוב	חודש	9/2019 – 8/2019

## 15. תכנון הבדיקות שיבוצעו

מספר בדיקה	מס' דרישה במסמך אפיון	מקרי הבדיקה	ידינית/אוטומטית	חשיבות
1	5.3.1	בדיקת כניסה למערכת באמצעות סיסמה קימת	ידינית	גבוהה
2	5.3.1	בדיקת כניסה למערכת באמצעות סיסמה שגויה	ידינית	גבוהה
3	5.3.4	בעת יצירת משתמש חדש המערכת בודקת שהסיסמא לא קימת	ידינית	גבוהה
4	5.3.5	בעת עדכון תחומי חיפשו המערכת מציגה משרות רלוונטיות	ידינית	בינונית
5	5.3.7	בעת הכנסת שאלה המערכת מאפשרת למשתמש להגדיר את אמצעות הקשר	ידינית	בינונית
6	5.3.8	בעת רישום המשתמש לסוכן החכם המערכת בודקת שיש לה את הפרטים העדכניים שלו	ידינית	בינונית
7	5.3.9	בעת עדכון פרטים אישים המערכת מעדכנת את מערכת הסוכן החכם אם המשתמש רשום	ידינית	בינונית


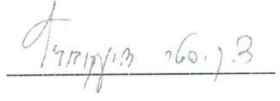
8	5.3.19	בעת פרסום משרה חדשה המערכת מדרגת את המשרה בהתאם לנתונים	ידנית	בינונית
9	5.3.19	בעת עדכון משרה קימת המערכת בודקת שהנתונים מלאים	ידנית	בינונית
10	5.3.11	בעת רישום משתמש למשרה המערכת בודקת מי פרסם לשליחת קו"ח	ידנית	בינונית
11	5.3.2	המערכת בודקת משתמשים שנכנסו בחודש האחרון ומכניסה אותם לרשימה להגרלה	ידנית	נמוכה

## 16. בדיקות יחידה (test unit)

לא רלוונטי.

## 17. בקרת גרסאות (version control)

כרגע לא רלוונטי - אם יהיה צורך הבקרה תתבצע על ידי git.

חתימת המנחה האישי	חתימת הסטודנט
	

הערות ראש המגמה במכללה:

---



---



---

אישור ראש המגמה במכללה:

שם: איתן גל חתימה: איתן תאריך: 13/8/19

הערות הגורם המקצועי מטעם מה"ט:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

צביקה שטרקמן

אישור הגורם המקצועי מטעם מה"ט:

שם: \_\_\_\_\_ חתימה: \_\_\_\_\_ תאריך: 25/2/19

## אישור הצעת הפרויקט ממה"ט

## תודות

תודה לבורא העולם.

צרור ברכות להוריני היקרים על התמיכה, סיוע  
והאוזן הקשבת במהלך השעות האינסופיות ( גם על חשבון הבית 😞 )

וכן לצוות הסמינר



## הצהרה

הצהרת התלמידים/ות

אנו דבורה קיסטר ת.ז. 207085580

ודינה יעקובוביץ ת.ז. 318993532

החתומים/ות מטה, מצהירים/ות בזאת כי כל הפרויקט המוגש בספר זה, הינו פרי עבודתנו בלבד על בסיס הנחייתה של המנחה ונתוך הסתמכות על לימודינו במכללה והמידע אשר רכשנו תוך כדי העבודה.

חתימת התלמידים/ות: דבורה קיסטר דינה יעקובוביץ

חתימת המנחה:

---

## תקציר

הפרויקט שלנו הינו אתר דרושים המאפשר פרסום וחיפוש משרות בהתאמה אישית לפי קריטריונים לבחירה. נותן דגש יחודי על מידע אודות סביבת העבודה והתאמתה לעובדת החרדית. המשתמש בוחר את המשרות הרלוונטיות וקורות החיים שלו נשלחים למעסיקים. ובנוסף המערכת כוללת טכנולוגית "סוכן חכם" השולחת למיל האישי של מחפש המשרה משרות חדשות המתאימות לו, ובכך. תדירות המייל תלויה בבחירת המשתמש. כמו כן האתר כולל פורום שאלות ותשובות בהלכה בנושא מקומות עבודה, ועוד.

## תוכן העניינים

I.....	דף כריכה
II.....	דף שער
.....	הצעה לפרויקט גמר..... שגיאה! הסימניה אינה מוגדרת.
XV.....	אישור הצעת הפרויקט ממה"ט
1.....	תודות
2.....	הצהרה
3.....	תקציר
4.....	תוכן העניינים
5.....	מבוא
7.....	מדריך למתכנת:
7.....	2.1. אסטרטגיות טכנולוגיות:
8.....	2.2. תיאור מבנה הפרויקט:
9.....	2.3. עקרונות התכנון/ הבניה/ הניתוח:
12.....	2.4. תרשימים:
15.....	2.5. מבנה נתונים מאוכסנים:
19.....	2.6. תוכן הפרויקט:
38.....	3. מדריך למשתמש: 3.1 הוראות כלליות לשימוש באתר:
41.....	3.2. מסכים:
45.....	4. סיכום ומסקנות:
48.....	5. נספחים:
48.....	6. ביבליוגרפיה:

## 1. מבוא

כאשר אישה חרדית מחפשת לעצמה עבודה, מלבד הפרמטרים המקובלים כמו תחום, שעות, מקום העבודה וכו' יש לה עקרון נוסף: היא רוצה לעבוד בסביבת עבודה המתאימה להשקפת עולמה. לצורך זה היא צריכה לעשות עבודת תחקיר מקיפה לגבי סביבת העבודה, חסימת האינטרנט וכו'. מצד שני ישנן עובדות רבות שבמשרדן מחפשים עובדות נוספות והן רוצות לחפש פרטנריות לעניין. קשה עד בלתי אפשרי להצליח במשימה המורכבת מכיוון שבדרך"כ הן לא מכירות קהל גדול רלוונטי. ארגון 'אידיאל' העוסק בהשמת עובדות חרדיות במקומות עבודה המתאימים לאישה החרדית, החליט לסייע לעניין. והדרך - פרמטיבית וקשה... עד היום, הם תפעלו את המיזם בעזרת מייל וקבצי אקסל... השילוב המורכב הנ"ל הוליד את הצורך בהקמת אתר מסודר ובנוי היטב, שיקל משמעותית הן על עובדות הארגון והן על הציבור בכללותו. וכאן אנחנו נכנסו לתמונה : ) קדימה, נפתח אתר!

- האתר יספק מידע על מקומות העבודה הן מהבחינה הטכנית והן מבחינה רוחנית על סביבת העבודה, סוג חסימת רשת ועוד.
- האתר יאפשר למחפש עבודה להתעדכן במשרות חדשות במשק.
- לאפשר לעובדת להמליץ למעסיקה על מועמדות מתאימות למשרה שברצונם לאייש.
- לאפשר לעובדים לשאול ולהתייעץ בענייני הלכה בפורום כתוח לציבור.
- לתת גב ותמיכה לעובדות חרדיות שחפצות לעבוד במקומות שמורים בלבד.

על מנת להקל בשימוש פתחנו אתר חברתי ידידותי למשתמש ונעים לעין.

### ממשק לאורח מזדמן-

האורח יכול לצפות במשרות ובפרטים שלהם, לערוך חיפוש מתקדם, לראות המלצות על המשרות וכן לעיין בפורם ההלכתי ליצור קשר, לענות על סקר וכו'

### ממשקים לחברי האתר-

#### ממשק למשתמש מחפש משרה

משתמש הרשום לאתר יכול מלבד האופציות של האורח המזדמן ליהנות משליחת קורות חיים שלו למשתפים, לראות לאיזה משרות שלח קו"ח, להתחבר למערכת "סוכן חכם" שבו הוא מקבל את המשרות החדשות למייל ועדכונים למייל.

#### ממשק למשתמש מציע משרה

מלבד האופציות של האורח המזדמן הוא יכול להציע משרות לחברה שלו ולעדכןם, להוסיף חברות חדשות, לקבל אליו למייל קורות חיים של מועמדים.

#### ממשק מנהל

הוספה עדכון ומחיקה של כל הנתונים, לראות איסה משתמשים נרשמו לכל משרה ולהעביר את הקורות חיים, לראות את המשרות החדשות שנוספו ולאשרם.

כמו כן יש לו אפשרות לראות סטטיסטיקות לגבי האתר כמו מספר המשרות שאוישו, מספר הנכנסים לאתר וכו'.

מכיוון שהאתר עתיד לעלות לענן, השתדלנו לפתח אותו רספונסיבי ובצורה שתוכל לקדמו באינטרנט.

## 2. מדריך למתכנת:

### אסטרטגיות טכנולוגיות:

בכתיבת פרויקט זה שמנו דגש רב על התאמה טכנולוגית מדויקת ושאיפה לטכנולוגיות חדשניות:  
צד שרת – **server side** כתבנו בשפת **C#** בפלטפורמת **Asp.net Web API**, תוך שימוש בטכנולוגיות  
נוספות כמו **Entity Framework** בסביבת עבודה של **Visual Studio 2017**.  
צד לקוח – **client side** כתבנו בשפות **Typescript, Html5, CSS** בפלטפורמת **Angular7** בסביבת  
**Visual Code**.  
בסיס הנתונים נכתב בעזרת **SQL Server**.  
את הממשק עצמו עצבנו בספריות **Angular Material, Material Icon Design** בשילוב  
**Bootstrap4**.  
העברת הנתונים בין ה- **server** וה- **client** היא ע"י אובייקטים בשפת **JSON**.

תיאור מבנה הפרויקט:

מבנה ה- Server side:

ה Solution שלנו כולל 4 פרויקטים בהתאם למודל השכבות:

### • DAL

פרויקט זה מורכב ממקור נתונים-מסד הנתונים שלנו, ומערכת תוכנה Entity Framework אשר תפקידה לקרוא את המידע הנדרש למערכת, לשמור את העדכונים, להוסיף מידע חדש או למחוק פרטי מידע קיימים. הפרויקט מובנה בשיטת DataBase First Entity Framework ולכן ה DB נבנה ראשון ועל סמך זה נסנו המחלקות והמאפיינים.

### • Entities

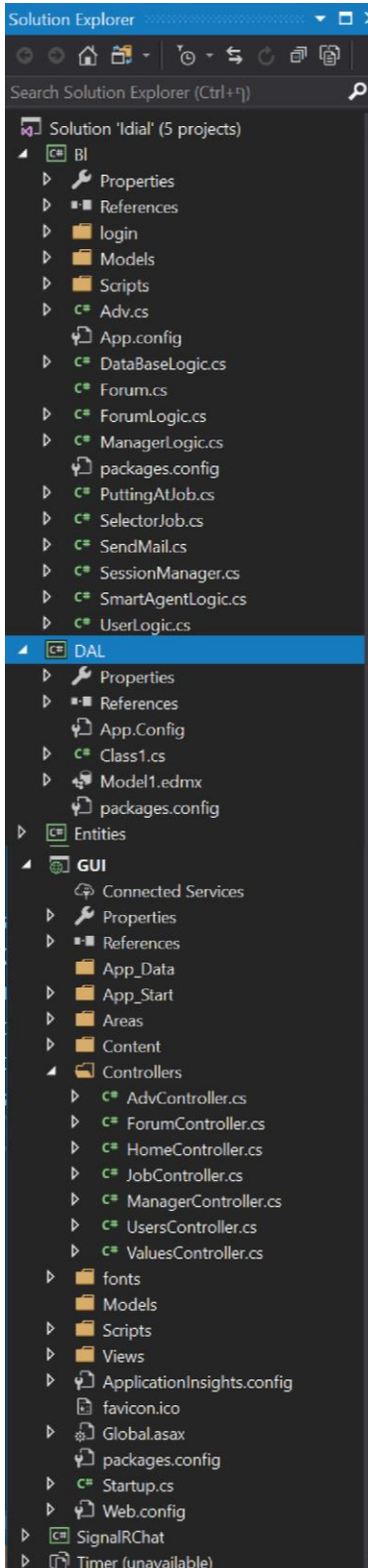
שכבה זו אחראית על המרת אובייקטים ממחלקת מיקרוסופט למחלקות שלנו על מנת שנוכל להעביר אותם בין הצד שרת לצד הקלינט. לשם כך יצרנו מחלקות (Class) עבור כל מחלקה בשכבת ה DAL. ופונקציות המרה זו כיוניות.

### • BL

הפרויקט שאחראי על הלוגיקה העסקית של המערכת, עוסקת בעיבוד המידע, חישובים שונים ושליחת המידע לשכבת התצוגה. בפרויקט זה נממש את הפונקציונליות של המערכת. מסד הנתונים והממשק משתמש מתקשרים דרך השכבה הזו.

### • GUI

פרויקט Web-API עימו מתממשק האתר. הוא מכיל Controllers: אלו מחלקות המכילות פונקציות המקבלות קריאות Http מפעילות פונקציות בשכבת ה BL ומחזירות את המידע הרצוי לצד הקלינט.



## מבנה ה Client side:

מבנה ה-client מורכב משלושה אלמנטים: דפי SCSS דפי HTML ודפי TS.  
דפי ה-TS כוללים בתוכם פונקציות לוגיות שונות, התחברות לשרתים ולספריות Angular Material. bootstrap וכו'  
דפי ה-HTML מהווים את התצוגה.  
דפי ה-CSS אחראיים על העיצוב והמראה הכללי.  
ההתחברות לצד שרת ע"י קריאות Http הנמצאות בדפים המכונים Services.

## עקרונות התכנון/ הבניה/ הניתוח:

### עקרונות תיאורטיים:

### **Server**

בצד השרת (המכונה גם ss) מתייחסים לפעולות המבוצעות על ידי השרת במערכת יחסי לקוח-שרת ברשת מחשבים.  
בדרך כלל, שרת היא תכנית מחשב, כגון שרת אינטרנט, שפועל על שרת מרוחק, נגיש מהחשב המקומי של משתמש או תחנת העבודה. פעילות יכולה להתבצע בצד השרת כי הם דורשים גישה למידע או פונקציונלי שאינה זמין בלקוח, או לדרוש התנהגות אופיינית שאינה אמין כאשר הוא נעשה בצד לקוח.

### **C#**

אלגוריתמיקה בסביבת visual studio ושפת c#  
C# היא שפת תכנות עילית מרובת-פרדיגמות, מונחית עצמים.  
בעיקרה המשלבת רעיונות כמו טיפוסיות חזקה, אימפרטיביות, הצהרתיות, פונקציונליות, פרוצדורליות וגנריות. השפה פותחה על ידי מיקרוסופט בשנת 2000 כחלק מפרויקט דוט נט ותוקננה בשנים 2005-2006 על ידי ארגון התקינה Ecma כתקן 334-ECMA וארגון התקינה הבינלאומי (איזו) כתקן ISO/IEC 23270:2006.

### **Entity Framework:**

Entity Framework Model הוא כלי חדשני ויעיל של Microsoft על מנת ליצור חיבור למסד נתונים. Entity Framework Model הוא ממפה אובייקטים מקושרים-קר, בסיס נתונים- המאפשר למפתחי .NET לעבוד עם בסיסי נתונים טבלאיים באמצעות אובייקטים ייעודיים.



## ASP.net Web API

מה זה API?

API הוא קיצור של Interface Program Application. במונחים פשוטים, זהו ממשק של פיסת תוכנה עם העולם החיצוני, הקובע איך ישתמשו בה. אלה החוקים שנקבעו עבור האינטראקציה שלה עם העולם הרחב, אלה יקבעו אילו חלקים של התוכנה יכולות לדבר עם תוכנות אחרות, ואיך היא תגיב.

Web API מה זה?

ASP.NET Web API הוא מסגרת המקלה על בניית שירותי HTTP המגיעים למגוון רחב של לקוחות, כולל דפדפנים והתקנים ניידים. API Web ASP.NET הוא פלטפורמה אידיאלית לבניית יישומים ב .NET Framework. בפריקט שלנו זו שכבת GUI.

### הפרדת שכבות:

כל תוכנה/ אתר שנפגוש בעולם מבוססת על ארכיטקטורת שכבות הנקראת בשם Tier- Three - שלש שכבות במבנה של BL-UI - Application DAL. זוהי תבנית עיצוב בסיסית שמגדירה הפרדת האפליקציה ל: שכבת נתונים, שכבת לוגיקה, ושכבת ממשק משתמש. לתבנית עיצוב זו יתרונות רבים:

1. תחזוקה:

ניתן להחליף או לתקן מימוש פנימי של שכבה אחת בארכיטקטורה בלי לשנות שכבה אחרת.

2. נוחות פיתוח:

אדם אחד עובד על רכיב בתכנה, אדם אחר עובד על רכיב אחר, כל עוד שהחתימות זהות ניתן לשלב כוחות ולייעל זמני פיתוח.

3. בדיקות:

תקלה כלשהיא מבודדת בכל שכבה בנפרד, לדוגמה אם לא קיבלנו רשימת נתונים לתצוגה

נבדוק קודם את שלב הביניים (שכבת ה BL) אם הנתונים שם תקינים נדע בוודאות שגם ברמת שכבת ה DAL הנתונים תקינים וכל שנותר הוא לפתור את התקלה ברמת ה UI.

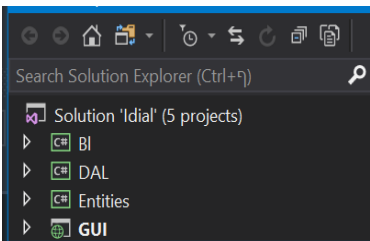
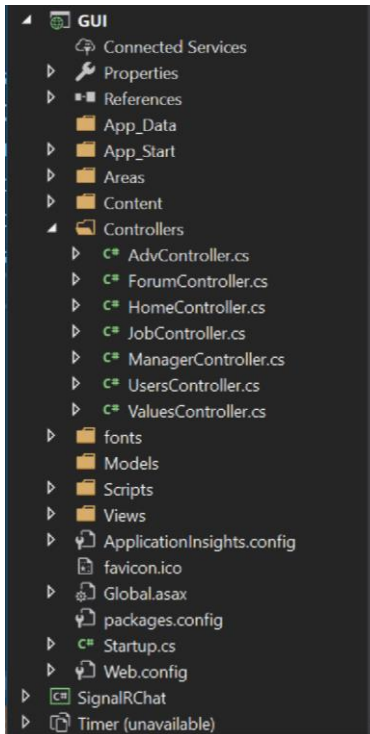
4. שימוש חוזר:

5. נניח שהרכיבים DAL + BL אהובים עלינו ועובדים היטב, ניתן להחליף את פלטפורמת UI לטכנולוגיה אחרת במינימום מאמץ.

6. אבטחה:

נוח יותר להגדיר Interface API לשכבה מסוימת בלי לחשוף מבני נתונים Logic או Data שלא רלוונטי למשתמש, כמו גם חסימה בפני האקרים (רלוונטי יותר בטכנולוגיות Web) ברמות שונות.

הפריקט של צד שרת-IdealServer מכיל את שלושת השכבות הנ"ל.



## Client

בצד הלקוח מתייחסים לפעולות המבוצעות על ידי הלקוח ביחסי לקוח-שרת ברשת מחשבים. בדרך כלל, הלקוח הוא יישום מחשב, כגון דפדפן אינטרנט, שפועל על המחשב המקומי של משתמש או תחנת עבודה ומתחבר לשרת כנדרש. פעילות יכולה להתבצע בצד הלקוח כי הם דורשים גישה למידע או פונקציונלי שנגיש בלקוח, אך לא בשרת, כי המשתמש צריך לצפות בהם או לספק קלט, או מכיוון שהשרת חסר כוח העיבוד כדי לבצע את הפעולות במועד לכל הלקוחות שהוא משרת. בנוסף, אם ניתן לבצע פעולות על ידי הלקוח, בלי לשלוח נתונים בשרת, הם יכולים לקחת פחות זמן, להשתמש בפחות רוחב פס, וכרוך בסיכון ביטחוני פחותה.

כאשר השרת משמש נתונים באופן נפוץ, למשל פי פרוטוקולי FTP או HTTP, משתמשים יכולים להיות בחירתם של מספר תוכניות לקוח (רוב הדפדפנים המודרניים יכולים לבקש ולקבל נתונים באמצעות שני פרוטוקולים אלה). במקרה של יישומים מיוחדים נוסף, מתכנתים יכולים לכתוב פרוטוקול שלהם שרת, הלקוח, והתקשורת, שניתן להשתמש בהם רק אחד עם השני. תוכניות הפועלות על המחשב המקומי של משתמש מבלי שולח או מקבלים נתונים דרך רשת אינן לקוחות נחשבים, ופעולות בצד הלקוח כך הפעולות של תוכניות כאלה לא תיחשבנה.

## Angular

Angular היא תשתית תוכנה בקוד פתוח ליישומי רשת המתוחזקת על ידי גוגל ועל ידי קהילה של מפתחים רבים וחברות, לצורך פתירת אתגרים רבים איתם נתקלים בפיתוח יישומי דף-יחיד. מטרתה היא פישוט הפיתוח והבדיקות של יישומים כאלו באמצעות תשתית תוכנה לארכיטקטורות צד לקוח כמו MVC או MVVM, יחד עם רכיבים בהם משתמשים בדרך כלל ביישומי אינטרנט עשירים.

ספריית Angular עובדת על ידי קריאת דף ה-HTML, שאל התגיות שבו נוספו תכונות נוספות. Angular יפרש את התכונות הללו כהנחיות לקשר את אזורי הקלט או הפלט בדף למודל שמיוצג על ידי משתני Javascript. האלו ניתנים לשינוי על ידי קוד, וניתן לגשת אליהם בצורה סטטית, או בצורה דינמית בעזרת JSON.

## Typescript

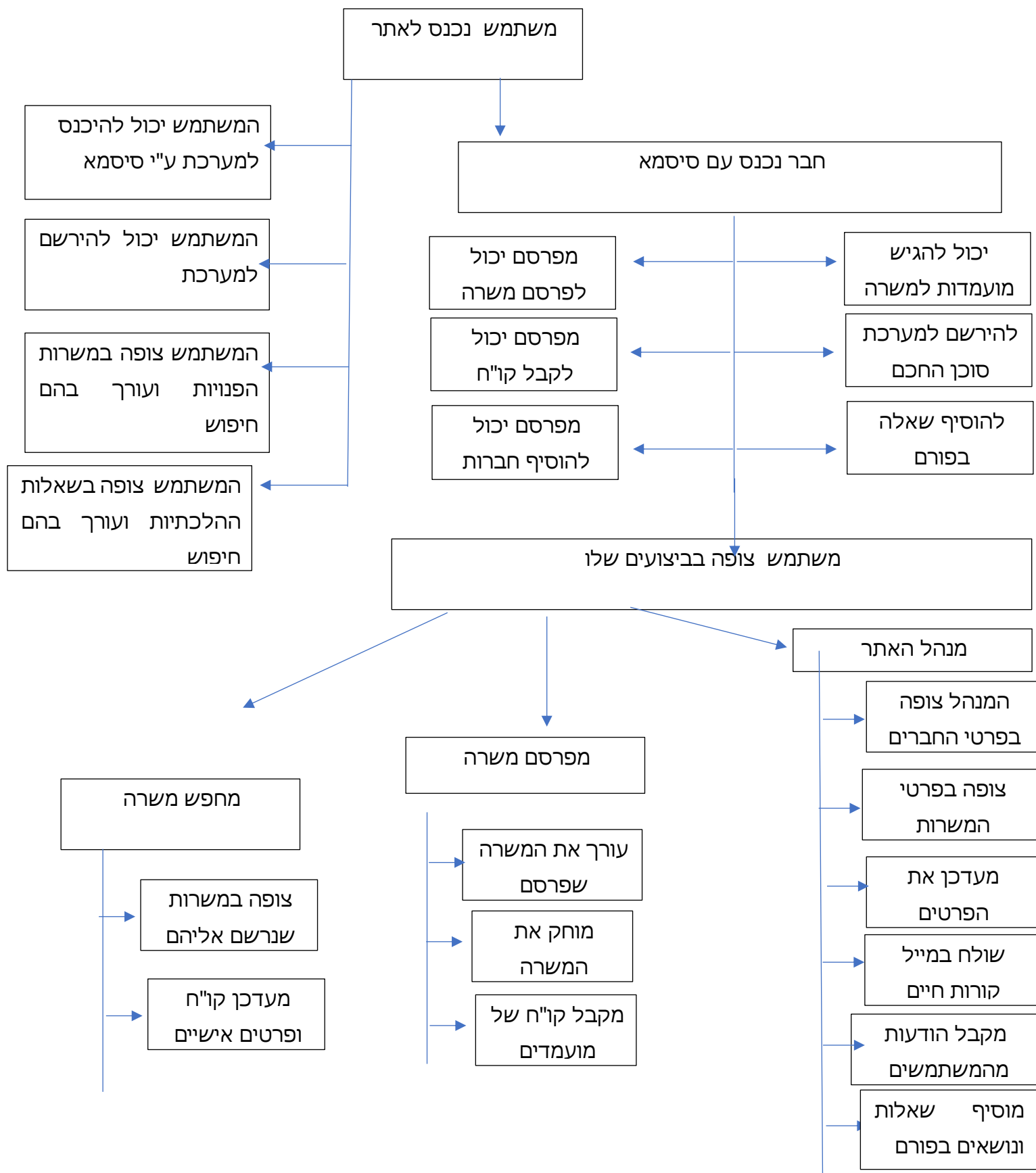
היא שפת תכנות חינוכית ומבוססת קוד פתוח המפותחת ומתוחזקת על ידי מיקרוסופט. היא מכילה את קבוצת כל פקודות ותחביר JavaScript הפופולרית, כלומר כל קוד JavaScript הוא גם קוד Typescript תקין, ומוסיפה עליה טיפוסים סטטיים ותכונות מונחה עצמים מבוסס מחלקות. בין התכונות שהיא מוסיפה: static typing, תמיכה במחלקות ותמיכה במודולים ובדקורטורים.

Typescript היא סופר-סט (superset) של Javascript כלומר, הקוד מבוסס על Javascript וחייב לעבור קומפילציה (תרגום) ל- Javascript כדי שהדפדפנים יבינו אותו מפני שהדפדפנים מבינים Javascript בלבד. בסופו של דבר בסופו של דבר קוד Typescript מעובד לכדי קוד JavaScript ולכן ניתן לומר "באחריות" שהשפה נתמכת בכל דפדפן שתומך ב-JavaScript.

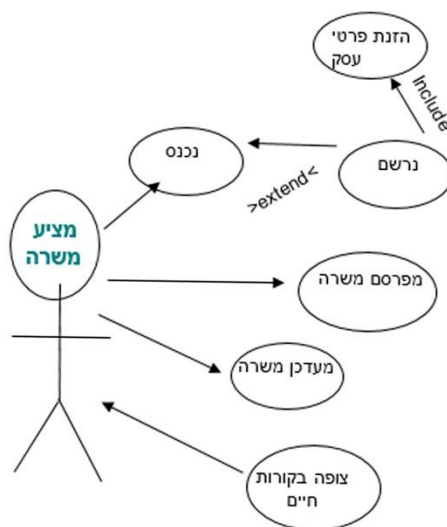
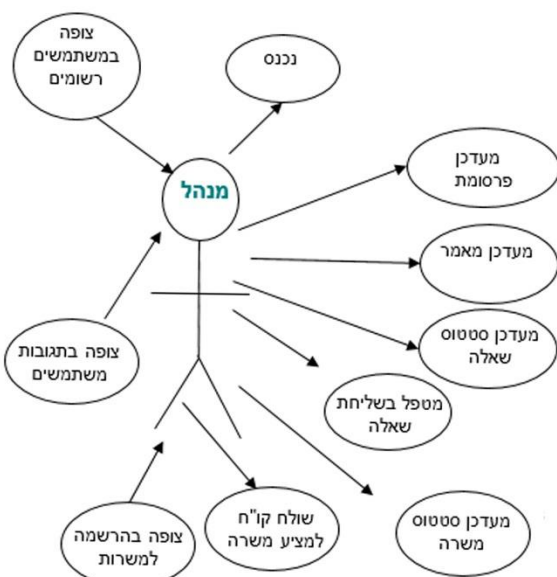
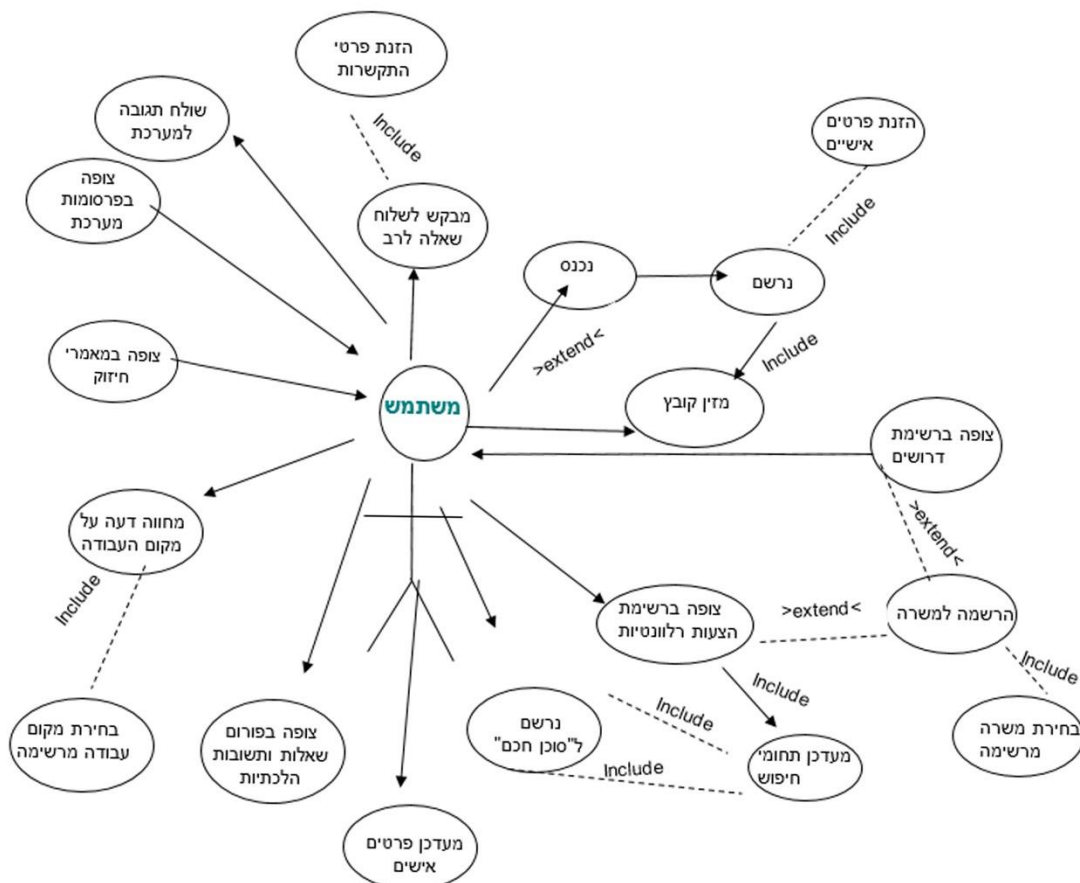
קבצי Typescript יישמרו בסיומת ts.

## תרשימים:

### עץ תהליכים:

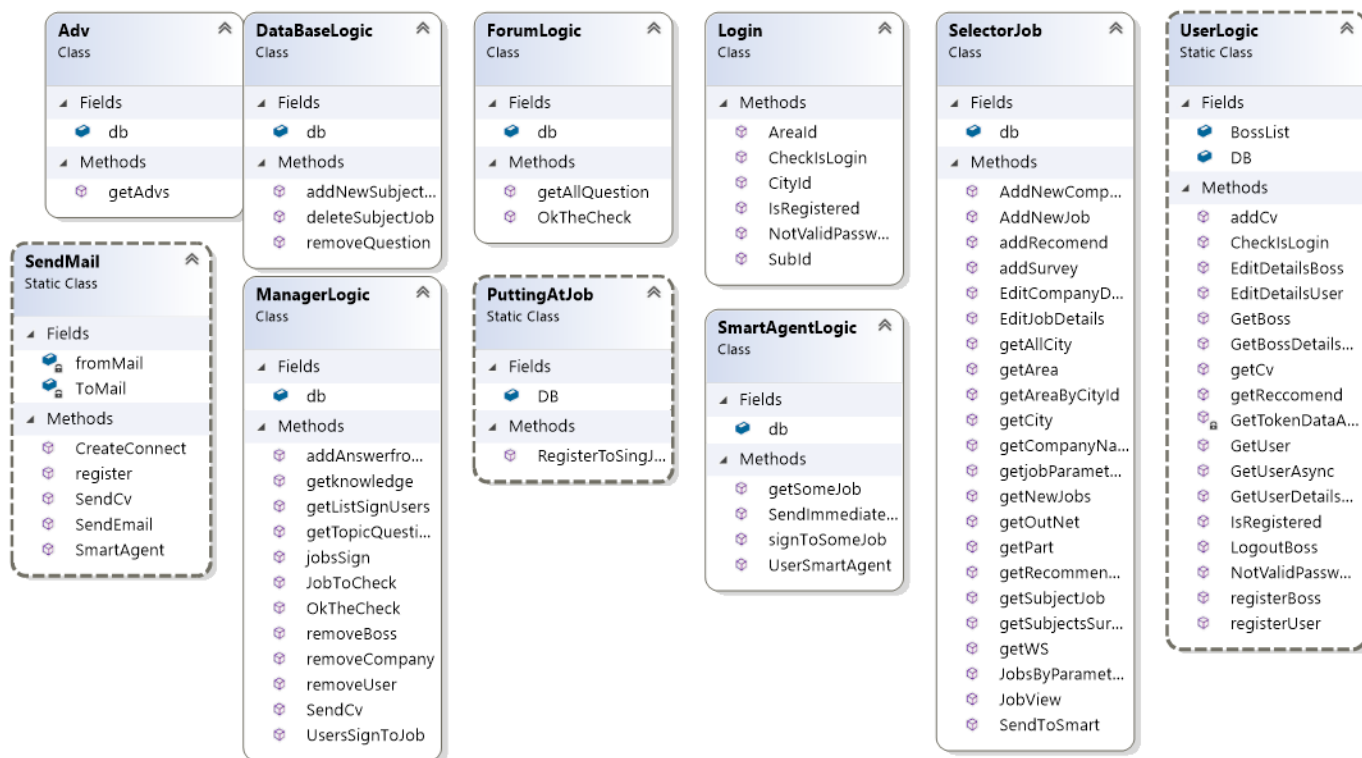


תרשים Uml:



## תרשים מראה המחלקות:

BL



API

**ManagerControl...**  
Class  
→ ApiController
 

**Methods**

- addAnswerfro...
- addNewSubject...
- deleteSubjectJob
- EditBoss
- EditUser
- getknowledge
- getListSignUsers
- getTopicQuesti...
- jobsSign
- JobsToCheck
- OkTheCheck
- removeBoss
- removeCompany
- removeQuestion
- removeUser
- SendCv

**UsersController**  
Class  
→ ApiController
 

**Methods**

- connect
- EditBoss
- EditUser
- GetBossById
- GetBossDetails
- getCv
- GetUserById
- GetUserDetails
- Login (+ 1 over...
- loginBoss
- LoginBoss
- register
- registerBoss
- registerToJob
- UploadJsonFile
- UserSmartAgent

**JobController**  
Class  
→ ApiController
 

**Methods**

- AddCompany
- AddJob
- addNewRecom...
- addSurvey
- EditCompany
- EditJob
- GetAllPart
- getArea
- getCity
- getCompany
- getRecommends
- getSomeJob
- getSubjectJob
- getSubjectsSur...
- jobParameters
- JobsByParamet...
- NewJobs
- signToSomeJob

**ForumController**  
Class  
→ ApiController
 

**Methods**

- getAllQuestion

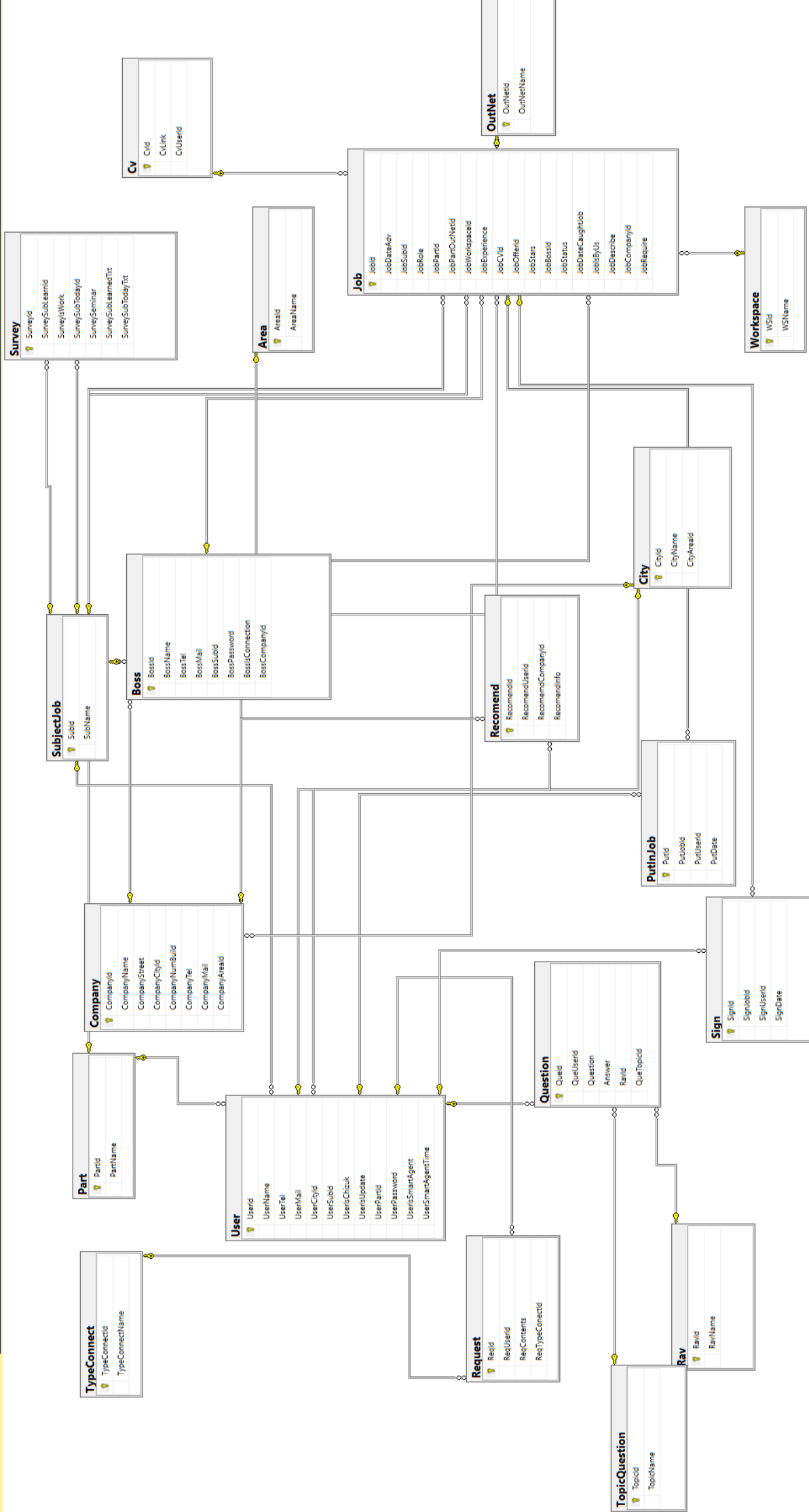
**AdvController**  
Class  
→ ApiController
 

**Methods**

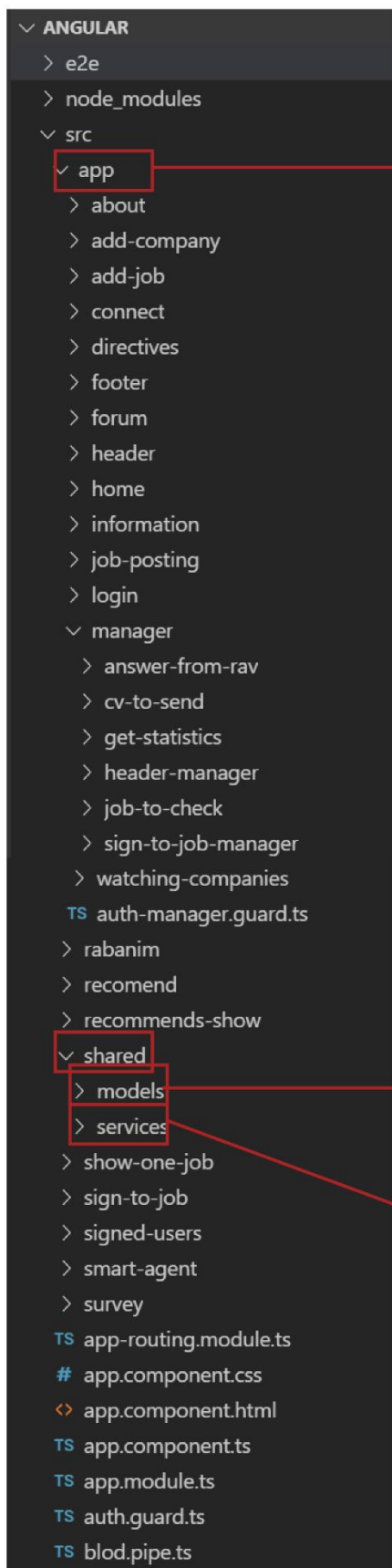
- getAdvs

מבנה נתונים מאוכסנים:

תיאור מבנה קבצי SQL:



## מבנה קבצים ותיקיות:



בתוך תיקיה זו נמצאים הקבצים שמגדירים דברים לכל האתר, והוא הקומפוננטה הראשית שמכילה בתוכה את כל הדפים וכן הקומפוננטות המסודרות בתיקיות ותת תיקיות הבנויות משלושה קבצים

1. html  
2. css  
3. .ts

תיקיה זו מכילה את כל המחלקות והאובייקטים

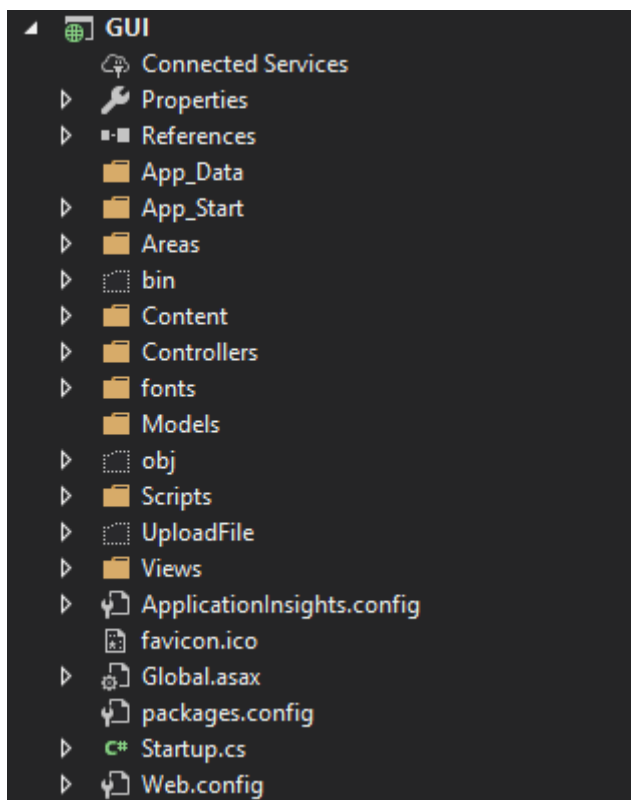
תיקיה זו מכילה קבצים שבהם פונקציות השולחות ומקבלות קריאות http תפקידן לקשר בין קליינט לסרבר



-

## תיקיית קורות חיים-

בעת רישום לאתר, המשתמש יכול להעלות את הקו"ח שלו.  
 התמונות נשמרות בתיקית UploadFile בשרת.



תוכן הפרויקט:

תאור המחלקות:

## מחלקות ב-DAL:

**User** - (משתמש)

מחלקה זו נותנת אפשרות לניהול משתמש מחפש משרה באתר.

```
public partial class User
{
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors",
    1 reference | 0 exceptions
    public User()
    {
        this.PutInJob = new HashSet<PutInJob>();
        this.Question = new HashSet<Question>();
        this.Recomend = new HashSet<Recomend>();
        this.Request = new HashSet<Request>();
        this.Sign = new HashSet<Sign>();
    }
    7 references | 0 exceptions
    public int UserId { get; set; }
    4 references | 0 exceptions
    public string UserName { get; set; }
    3 references | 0 exceptions
    public string UserTel { get; set; }
    8 references | 0 exceptions
    public string UserMail { get; set; }
    6 references | 0 exceptions
    public Nullable<int> UserCityId { get; set; }
    6 references | 0 exceptions
    public Nullable<int> UserSubId { get; set; }
    3 references | 0 exceptions
    public Nullable<bool> UserIsChizuk { get; set; }
    3 references | 0 exceptions
    public Nullable<bool> UserIsUpdate { get; set; }
    5 references | 0 exceptions
    public Nullable<int> UserPartId { get; set; }
    9 references | 0 exceptions
    public string UserPassword { get; set; }
    5 references | 0 exceptions
    public string UserPassword { get; set; }
    5 references | 0 exceptions
    public Nullable<bool> UserIsSmartAgent { get; set; }
    5 references | 0 exceptions
    public Nullable<int> UserSmartAgentTime { get; set; }
    0 references | 0 exceptions
    public virtual City City { get; set; }
    0 references | 0 exceptions
    public virtual Part Part { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<PutInJob> PutInJob { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Question> Question { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Recomend> Recomend { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Request> Request { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    3 references | 0 exceptions
    public virtual ICollection<Sign> Sign { get; set; }
    0 references | 0 exceptions
    public virtual SubjectJob SubjectJob { get; set; }
}
```

## **Boss** - (משתמש)

מחלקה זו נותנת אפשרות לניהול מפרסם משרה באתר.

```
public partial class Boss
{
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
    1 reference | 0 exceptions
    public Boss()
    {
        this.Job = new HashSet<Job>();
    }
    7 references | 0 exceptions
    public int BossId { get; set; }
    3 references | 0 exceptions
    public string BossName { get; set; }
    3 references | 0 exceptions
    public string BossTel { get; set; }
    6 references | 0 exceptions
    public string BossMail { get; set; }
    3 references | 0 exceptions
    public Nullable<int> BossSubId { get; set; }
    7 references | 0 exceptions
    public string BossPassword { get; set; }
    4 references | 0 exceptions
    public Nullable<bool> BossIsConnection { get; set; }
    3 references | 0 exceptions
    public Nullable<int> BossCompanyId { get; set; }
    0 references | 0 exceptions
    public virtual Company Company { get; set; }
    0 references | 0 exceptions
    public virtual SubjectJob SubjectJob { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Job> Job { get; set; }
}
```

## **Recommend** - (המלצה)

מחלקה זו נותנת אפשרות להוסיף המלצה למשרה

```
public partial class Recomend
{
    2 references | 0 exceptions
    public int RecomendId { get; set; }
    2 references | 0 exceptions
    public Nullable<int> RecomendUserId { get; set; }
    4 references | 0 exceptions
    public Nullable<int> RecomendCompanyId { get; set; }
    2 references | 0 exceptions
    public string RecomendInfo { get; set; }

    0 references | 0 exceptions
    public virtual Company Company { get; set; }
    0 references | 0 exceptions
    public virtual User User { get; set; }
}
```

## Company - (חברה)

מחלקה זו נותנת אפשרות להוסיף חברה

```
public partial class Company
{
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
    1 reference | 0 exceptions
    public Company()
    {
        this.Boss = new HashSet<Boss>();
        this.Job = new HashSet<Job>();
        this.Job1 = new HashSet<Job>();
        this.Recomend = new HashSet<Recomend>();
    }
    9 references | 0 exceptions
    public int CompanyId { get; set; }
    4 references | 0 exceptions
    public string CompanyName { get; set; }
    3 references | 0 exceptions
    public string CompanyStreet { get; set; }
    5 references | 0 exceptions
    public Nullable<int> CompanyCityId { get; set; }
    3 references | 0 exceptions
    public Nullable<int> CompanyNumBuild { get; set; }
    3 references | 0 exceptions
    public string CompanyTel { get; set; }
    4 references | 0 exceptions
    public string CompanyMail { get; set; }
    6 references | 0 exceptions
    public Nullable<int> CompanyAreaId { get; set; }
    0 references | 0 exceptions
    public virtual Area Area { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Boss> Boss { get; set; }
    0 references | 0 exceptions
    public virtual City City { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Job> Job { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Job> Job1 { get; set; }
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
    1 reference | 0 exceptions
    public virtual ICollection<Recomend> Recomend { get; set; }
}
```

## Survey - (סקר)

מחלקה זו נותנת אפשרות להוסיף תשובות לסקר

```
public partial class Survey
{
    2 references | 0 exceptions
    public int SurveyId { get; set; }
    3 references | 0 exceptions
    public Nullable<int> SurveySubLearnId { get; set; }
    2 references | 0 exceptions
    public Nullable<bool> SurveyIsWork { get; set; }
    3 references | 0 exceptions
    public Nullable<int> SurveySubTodayId { get; set; }
    2 references | 0 exceptions
    public string SurveySeminar { get; set; }
    2 references | 0 exceptions
    public string SurveySubLearnedTxt { get; set; }
    2 references | 0 exceptions
    public string SurveySubTodayTxt { get; set; }

    0 references | 0 exceptions
    public virtual SubjectJob SubjectJob { get; set; }
    0 references | 0 exceptions
    public virtual SubjectJob SubjectJob1 { get; set; }
}
```

### Question - (שאלה)

מחלקה זו מאפשרת להוסיף שו"ת לפורום הרבני.

```
public partial class Question
{
    3 references | 0 exceptions
    public int QueId { get; set; }
    2 references | 0 exceptions
    public Nullable<int> QueUserId { get; set; }
    2 references | 0 exceptions
    public string Question1 { get; set; }
    2 references | 0 exceptions
    public string Answer { get; set; }
    0 references | 0 exceptions
    public Nullable<int> RavId { get; set; }
    2 references | 0 exceptions
    public Nullable<int> QueTopicId { get; set; }

    0 references | 0 exceptions
    public virtual Rav Rav { get; set; }
    0 references | 0 exceptions
    public virtual TopicQuestion TopicQuestion { get; set; }
    0 references | 0 exceptions
    public virtual User User { get; set; }
}
```

## מחלקות ב API

### UserController (משתמש)

```
public class UsersController : ApiController
{
    // [Route("login")] ...
    [Route("login")]//מקבלת אובייקט משתמש ובודקת אם קיים
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult Login([FromBody] Entities.User user)...
    [Route("GetUser/{id}")]//user ומחזירה id
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult Login(int id)...
    // [HttpGet] ...
    [Route("GetBoss/{id}")]//Boss ומחזירה id
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public Entities.Boss LoginBoss(int id)...
    [Route("loginBoss")]//בודקת אם קיים Boss מקבלת אובייקט
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public Entities.Boss loginBoss([FromBody] Entities.Boss boss)...
    [Route("registerBoss")]//מקבלת Boss ורושמת אותו
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public Entities.Boss registerBoss([FromBody] Entities.Boss boss)...
    [Route("registerUser")]//מקבלת User ורושמת אותו
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public Entities.User register([FromBody] Entities.User user)...
    [Route("GetUserById/{id}")]//id לפי user מחזירה אובייקט
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public Entities.User GetUserById(int id)...
    [Route("GetBossById/{id}")]//id לפי Boss מחזירה אובייקט
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public Entities.Boss GetBossById(int id)...
    [Route("EditUser")]//עריכת פרטי משתמש
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult EditUser([FromBody]Entities.User editUser)...
    [Route("EditBoss")]//עריכת פרטי מפרסם
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult EditBoss([FromBody]Entities.Boss editBoss)...
    [Route("file")]//מקבלת קובץ (קורות חיים) ושומר בתקית
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public HttpResponseMessage UploadJsonFile()...
    [Route("registerToJob/{idJob}/{idUser}")]//משתמש נרשם למשרה ספציפית
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult registerToJob(int idJob, int idUser)...
    [Route("connect/{mail}/{phone}/{name}/{details}/{subject}")]//יצירת קשר
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult connect(string mail, string phone, string name, string details)
    [Route("getCv/{idUser}")]//מחזיר קורות חיים של משתמש מסוים
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public string getCv(int idUser)...
```



## JobController - (נהול משרות)

```
public class JobController : ApiController
{
    [Route("getParts")]//מחזיר חלקי משרה
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult GetAllPart()...

    [Route("getArea")]//מחזיר אזורים
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getArea()
    {
        return Ok(BL.SelectorJob.getArea());
    }

    [Route("getCity/{AreaId}")]
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getCity(int AreaId)
    {
        return Ok(BL.SelectorJob.getCity(AreaId));
    }

    [Route("getSubjectJob")]//מחזיר מקצועות
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getSubjectJob()...

    [Route("removeWorkspace/{id}")]//הסרת סביבת עבודה מסוימת לפי id מסוים
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult removeWorkspace(int id)...

    [Route("removeJob/{id}")]//הסרת עבודה מסוימת לפי Id
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult removeJob(int id)...

    [Route("addPart/{partName}")]//הוספה של חלקי משרה
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult addPart(string partName)...

    [Route("removePart/{id}")]//הסרה של חלקי משרה לפי id
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult removePart(int id)...

    [Route("addSubjectQues/{question}")]//הוספת נושא לפורום הרבני
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult addSubjectQues(string question)
    {
        return Ok(BL.ManagerLogic.addTopicQuestion(question));
    }
}
```

## ManagerController - (ביצועים של מנהל האתר)

```
public class ManagerController : ApiController
{
    [Route("JobsToCheck")]//מקבל רשימת משרות ע"מ לאשר אותם
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult JobsToCheck()...
    [Route("OkTheCheck/{idJob}")]//מאשר משרה ספציפית
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult OkTheCheck(int idJob)...
    [Route("removeCompany/{companyId}")]//הסרה חברה לפי id שהיא מקבלת
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult removeCompany(int companyId)...
    [Route("removeBoss/{bossId}")]//הסרת מפרטס משרה לפי id מסוים
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult removeBoss(int bossId)...
    [Route("removeUser/{userId}")]//הסרת משתמש משרה לפי id מסוים
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult removeUser(int userId)...
    [Route("addAnswerfromRav")]//הוספת שאלה לפורום הרבני
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult addAnswerfromRav([FromBody]Entities.Question question)...
    [Route("getTopicQuestion")]//מחזיר את נושאי השאלות בפורום הרבני
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getTopicQuestion()...
    [Route("jobsSign")]
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult jobsSign()//רשימת העבודות שיש להן אנשים שרשומים
    [Route("userToSpecificJob/{jobId}")]//רשימת האנשים שרשומים למשרה מסוימת
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getListSignUsers(int jobId)...
    [Route("EditUser")]//לעדכן נתוני משתמש
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult EditUser([FromBody]Entities.User editUser)...
    [Route("EditBoss")]//לעדכן נתוני בוס
    [HttpPost]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult EditBoss([FromBody]Entities.Boss editBoss)...
    [Route("getknowledge/")]//לקבל את תוצאות הסקר
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getknowledge()...
    [Route("addNewSubjectJob/{subjectName}")]
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult addNewSubjectJob(string subjectName)//הוספת מקצוע חדש
    ...
}
```



## ForumController - (נהול הפורום)

```
[EnableCors(origins: "*", headers: "*", methods: "*")]
[System.Web.Http.RoutePrefix("api/forum")]
0 references
public class ForumController : ApiController
{
    [Route("getAllQuestion")]//מחזיר את כל השאלות שבפורום
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getAllQuestion()...

    [Route("AddQuestion")]//הוספת שאלה
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult AddQuestion()...

    // [Route("deleteJob{jobId}")] ...
}
```

## AdvController - (ניהול הפרסומות באתר)

```
[RoutePrefix("Api/Adv")]
[EnableCors(origins: "*", headers: "*", methods: "*")]
0 references
public class AdvController : ApiController
{
    [Route("getAdvs")]//מחזיר פרסומות
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult getAdvs()...

    [Route("editAdvs")]//מעדכן פרסומות
    [HttpGet]
    0 references | 0 requests | 0 exceptions
    public IHttpActionResult editAdvs()...
}
```

## תאור פונקציות

### פונקציות בצד שרת

1. פונקציית `JobByParameters` במחלקת `SelectorJob` מקבלת מספר פרמטרים ועל פיהם מתבצע הסינון בתוך מאגר המשרות. במידה ועבור קריטריון מסוים לא נבחר פרמטר ברירת המחדל תהיה "הכל".  
הפונקציה מחזירה את המשרות התואמות לפרמטרים שנשלחו בחיפוש.

```
public static List<JobView> JobsByParameters(int? city1 = 0, int? area1 = 0, int? part1 = 0, int? sub1 = 0)
{
    var com = db.Company.ToList();
    List<Entities.Job> Job1 = new List<Entities.Job>();
    List<Entities.Job> Job2 = new List<Entities.Job>();
    var y = from u in db.Job
            where u.JobStatus == true
            orderby u.JobDateAdv descending
            select u;
    foreach (var item in y)
    {
        Job1.Add(Entities.Job.JobEntities(item));
    }
    if (sub1 != 0)
        Job1 = Job1.Where(j => j.JobSubId == sub1).ToList();
    if (part1 != 0)
        Job1 = Job1.Where(j => j.JobPartId == part1).ToList();
    if (area1 == 0 && city1 == 0)
    {
        return JobView(Job1);
    }
    if (area1 != 1)
    {
        if (area1 != 0)
            com = com.Where(p => p.CompanyAreaId == area1).ToList();
        if (city1 != 0 && city1 != 1)
            com = com.Where(p => p.CompanyCityId == city1).ToList();
    }
    foreach (var item in com)
    {
        Job2.AddRange(Job1.Where(c => c.JobCompanyId == item.CompanyId && c.JobStatus == true).ToList());
    }
    return JobView(Job2);
}
```

1 reference | 0 exceptions

```
public static List<JobView> getNewJobs()
{
    List<Entities.Job> Job2 = new List<Entities.Job>();
    var y = from u in db.Job
            where u.JobStatus == true
            orderby u.JobDateAdv descending
            select u;
    foreach (var item in y)
    {
        Job2.Add(Entities.Job.JobEntities(item));
    }

    return JobView(Job2);
}
```

## 2. מערכת סוכן חכם

מערכת דיוורים השולחת למייל המשתמש משרות חדשות. לכל משתמש יתכנו תדירויות דיוור של פעם ביום, שבוע וכו'. היא מורכבת משלושה חלקים עיקריים:

א- במחלקת LoadProject: בונה סטטית (מופעלת בקריאה חיצונית בעת טעינת האתר) המפעילה פעם ביום (Timer) את CheckTimer - פונקציה הבודקת את היום והתאריך ולפיהם מתזמנת את UserSmartUser.

```
public static class LoadProjectBL
{
    private readonly static System.Timers.Timer _checkTimer = new System.Timers.Timer(); // הקצאת הטימר
    public static readonly int CheckTimerInterval = 1000 * 60 * 60 * 24; // הטימר מופעל פעם ביום
    /// <summary> להפעיל פונקציה לאיזה פונקציה להפעיל
    0 references | 0 exceptions
    static LoadProjectBL() // בונה סטטית
    {
        _checkTimer.Elapsed += CheckTimerElapsed;
        _checkTimer.Interval = CheckTimerInterval;
        _checkTimer.Enabled = true;
    }
    1 reference | 0 exceptions
    static void CheckTimerElapsed(object source, ElapsedEventArgs e)
    {
        BL.SmartAgentLogic.UserSmartAgent(1); // פעם ביום
        if ((DateTime.Today.Date.DayOfWeek).ToString() == "Thursday") // פעם בשבוע (יום חמישי)
            BL.SmartAgentLogic.UserSmartAgent(2);
        if (DateTime.Today.Day == 28) // פעם בחודש (ב-28)
            BL.SmartAgentLogic.UserSmartAgent(3);
    }
    1 reference | 0 exceptions
    public static void UseFuncToLoadThisClass()
    { int x = 1; }
}
```

## ב- פונקציית UserSmartAgent שבמחלקת SmartAgent

פונקציה אוטומטית המקבלת תדירות, ושולחת מיילים למשתמשים שנרשמו למערכת "סוכן חכם", ובהם פירוט המשרות הרלוונטיות לקריטריונים של המשתמש, שנוספו בפרק זמן התדירות הנוכחית (באותו יום/שבוע וכדו').

```
public static bool UserSmartAgent(int timeNumber)
{
    List<Entities.User> users = new List<Entities.User>();
    var h = db.User.ToList();
    foreach (var item in h)
    {
        if (item.UserIsSmartAgent == true)//if the user login to the smartAgent
        {
            if (item.UserSmartAgentTime == timeNumber)
            {
                users.Add(Entities.User.UserEntities(item));
                int areaId = BL.SelectorJob.getAreaByCityId(Convert.ToInt32(item.UserCityId));
                //a list of jobs that can be ok to the user
                List<JobView> jobsView = SelectorJob.JobsByParameters(item.UserCityId, areaId, item.UserPartId, item.UserSubId);
                List<Job> jobs = new List<Job>();
                foreach (var j in jobsView)
                {
                    var currentJob = db.Job.FirstOrDefault(p => p.JobId == j.JobId);
                    if (currentJob!=null)
                    {
                        jobs.Add(Entities.Job.JobEntities( currentJob));
                    }
                }
                List<Job> jobToSend = new List<Job>();
                foreach (var i in jobs)
                {
                    if (i.JobStatus!=null&&i.JobStatus == true)
                    {
                        int timePastFromAdv;
                        DateTime temp;
                        DateTime.TryParse(i.JobDateAdv.ToString(), out temp);
                        if (temp != null)
                        {
                            double diff = (DateTime.Today - temp).TotalDays;

                            switch (timeNumber)
                            {
                                case 1:
                                    timePastFromAdv = 1;
                                    break;
                                case 2:
                                    timePastFromAdv = 7;
                                    break;
                                case 3:
                                    timePastFromAdv = 30;
                                    break;
                                default:
                                    timePastFromAdv = 30;
                                    break;
                            }
                            if (Convert.ToInt32(diff) <= timePastFromAdv)
                            {
                                jobToSend.Add(i);
                            }
                        }
                    }
                }
                if (jobToSend.Count > 0)
                {
                    BL.SendMail.SmartAgent(Entities.User.UserEntities(item), BL.SelectorJob.JobView(jobToSend));
                }
            }
        }
    }
    return true;
}
```

## ג- פונקציית SmartAgent במחלקת SendMail

מקבלת מ UserSmartAgent משתמש ורשימת משרות, ושולחת לו אותם במייל בצורה מסודרת, ומקשרת אותו חזרה לאתר- בעזרת פונקציית SendMail.

```
public static bool SmartAgent(Entities.User user, List<Entities.JobView> jobs)
{
    string htmlText = @"
    string str = "";
    foreach (var job in jobs)
    {
        htmlText += @"<table style='z-index: 1000;
        padding: 39px 40px 40px 40px;
        padding-top: 35px;
        width: 518px;
        border: none;
        direction: rtl;
        font-family: Arial;
        font-size: 13px;
        background-color: white;
        margin-right: 25px;
        float: right;
        margin-bottom: 70px;
        border-radius: 6px;'>
        <tr><b>תפקיד</b> <td>" + job.JobRole + "</td> </tr>" +
        "<tr><b>תאור</b> <td>" + job.JobDescribe + "</td> </tr>" +
        "<tr><b>דרישות</b> <td>" + job.JobRequire + "</td> </tr>" +
        "<tr><b>מקום</b> <td>" + job.AreaName + "</td> </tr>" +
        "<tr><b>סוג משרה</b> <td>" + job.PartName + "</td> </tr>" +
        "<tr><b>סוג חסימת</b> <td>" + job.OutNetName + "</td> </tr>" +
        "<tr><b>סביבת עבודה</b> <td>" + job.WSName + "</td> </tr>" +
        "<a href='\"http://localhost:4200/show-jobs/\"' + job.JobId.ToString() + \"\" style='padding: 12px 45px;font-size: 14px;color: #ffffff;' +
        \"line-height: 22px;background-color: #003399;font-weight: bold;border-radius: 2px;'>\" +
        str += job.JobId.ToString() + 'A';
    }
    htmlText += "<a href='\"http://localhost:4200/show-jobs/\"' + str + \"\" style='padding: 12px 45px;font-size: 14px;color: #ffffff;line-height: 22px;' +
    \"background-color: #003399;font-weight: bold;border-radius: 2px;'>לכל המשרות</a></div></body>";
    if (jobs.Count == 1) SendEmail(htmlText, "משרה לוחטת: " + jobs[0].JobRole, user.UserMail);
    else SendEmail(htmlText, "נמצאו " + jobs.Count.ToString() + " משרות המתאימות לך", user.UserMail);
    return true;
}
```

```
public static MailMessage SendEmail(string htmlText, string subject, string toMail)
{
    try
    {
        SmtpClient smtp = new SmtpClient();
        smtp.Host = "smtp.gmail.com";
        smtp.Port = 587;
        smtp.UseDefaultCredentials = true;
        smtp.Credentials = new System.Net.NetworkCredential("idealtojob@gmail.com", "963852741+");//
        AlternateView plainView = AlternateView
        .CreateAlternateViewFromString("Some plaintext", Encoding.UTF8, "text/plain");
        smtp.EnableSsl = true;
        if (toMail == "")
            toMail = ToMail;
        MailMessage mail = new MailMessage(fromMail, toMail, subject, htmlText);
        mail.AlternateViews.Add(plainView);
        System.Net.Mail.Attachment attachment;
        AlternateView htmlView =
            AlternateView.CreateAlternateViewFromString(htmlText, Encoding.UTF8, "text/html");
        mail.AlternateViews.Add(htmlView); // And a html attachment to make sure.
        mail.IsBodyHtml = true;
        mail.BodyEncoding = UTF8Encoding.UTF8;
        smtp.Send(mail);
        return new MailMessage();
    }
    catch (Exception ex)
    {
        var x = ex.Message;
        return new MailMessage();
    }
}
```

### 3. פונקציה **SendImmediatelySmart**

ברגע שמנהל האתר מאשר משרה חדשה- הפונקציה מתוזמנת ובודקת איזה מבין המשתמשים שהיא יכולה להתאים להם מבחינת הקטריונים, רשומים בסוכן חכם לקבל משרה מיד בעת פרסומה (ולא רק בתדירות מסוימת), ומזמנת להם את פונקציית שליחת המייל.

```
public static bool SendImmediatelySmartAgent(Entities.Job newJob)
{
    var company = db.Company.FirstOrDefault(p => p.CompanyId == newJob.JobCompanyId);
    List<Entities.User> userEnt = new List<User>();
    List<DAL.User> users =
        db.User.Where(p => p.UserIsSmartAgent == true && p.UserSmartAgentTime == 4
        && p.UserPartId == newJob.JobPartId && p.UserSubId == newJob.JobSubId).ToList();

    List<Entities.City> cities = SelectorJob.getAllCity();
    foreach (var item in users)
    {
        var cityid = cities.FirstOrDefault(c => c.CityId == item.UserCityId);
        if (cityid.CityAreaId == company.CompanyAreaId)
            userEnt.Add(Entities.User.UserEntities(item));
    }
    List<Entities.Job> jobs = new List<Job>();
    jobs.Add(newJob);
    var jobView = SelectorJob.JobView(jobs);
    foreach (var item in userEnt)
    {
        SendMail.SmartAgent(item, jobView);
    }
    return true;
}
```

### 4. פונקציה **jobSign**

פונקציה שמחזירה את כל העבודות שיש אנשים שנרשמו אליהם.

```
public static List<Entities.JobView> jobsSign()
{
    var jobs = db.Job.Where(job => job.Sign.Count > 0).ToList();
    var temp = jobs?.Select(job => Entities.Job.JobEntities(job)).ToList();
    return BL.SelectorJob.JobView(temp);
}
```

## פונקציות בצד לקוח

### • מחשב מסתמש

בדף JobTable – חיפוש משרות

1. פונקציית המקבלת מהשרת את כל המשרות הפנויות ובודקת אם המשתמש לא נרשם אליהם עדין על פי חילוץ המערך "Myjobs" השמור בlocalStorage.

```
ngOnInit() {
  this.load = true;
  this.subscriber = this.jobService.getJobParameters().subscribe(state =>
{
  this.jobParameters = state;
  this.jobService.getNewJobs().subscribe(s => {
    this.load = false;
    this.jobs = s;
    this.allJobs = s;
    this.length = s.length;
    this.fillJobs(0);

  });
});
if(localStorage.getItem("myJobs"))
  this.dataSource = JSON.parse(localStorage.getItem("myJobs"));
}

fillJobs(pageIndex) {
  this.jobs = this.allJobs.slice(pageIndex * this.pageSize, (pageIndex +
1) * this.pageSize);
  if (localStorage.getItem("myJobs"))
    for (let i = 0; i < this.jobs.length; i++) {
      this.j = 0;
      for (this.j = 0; this.j < this.dataSource.length; this.j++) {
        if (this.jobs[i].JobId == this.dataSource[this.j]) {
          this.jobs[i].JobSigned = true;
          break;
        }
      }
    }
}
}
```



2. פונקציות רישום למשרה :

(אם המשתמש עדין לא נרשם לאתר- פותח דיאלוג הרשמה).

רושם משתמש למשרה, שומר את המשרה ב `localStorage` בתוך `"myJobs"`.

```
registerToJob(idJob: number) {
  if (localStorage.getItem("token") == null) {
    const dialogRef = this.dialog.open(LoginUserComponent, {
      width: '250px',
      height: '80vh',
    });
    dialogRef.afterClosed().subscribe(result => { });
  }
  else {
    this.userService.registerToJob(idJob,
    this.userService.user.UserId).subscribe(res => {
      if (res) {
        this._snackBar.open('נרשמת בהצלחה', 'X', { duration: 3000 });
        // הוספה למשרות שנרשם
        this.allJobs.find(p => p.JobId == idJob).JobSigned = true;
        if (!localStorage.getItem("myJobs")) { // אם עדיין אין משרות שנרשם אליהם
          this.dataSource = [];
          this.dataSource.push(idJob);
        }
        else {
          this.dataSource =
JSON.parse(localStorage.getItem("myJobs")); // שליפת המשרות
          this.dataSource.push(idJob);
        }
        localStorage.setItem("myJobs", JSON.stringify(this.dataSource));
        this.subjectBasket.next(this.dataSource.length); // העברה של המידע
        // שהיה שינוי
      }
      else {
        this._snackBar.open('תקלה במערכת. נסי שוב מאוחר יותר', 'X', { duration:
6000 });
      }
    })
  }
}
```

3. עדכון פרטים והעלאת קו"ח חדשים לפני ההרשמה למשרה

```
UpdateDetails(idJob: number) {
  if (localStorage.getItem("token") == null) {
    const dialogRef = this.dialog.open(LoginUserComponent, {
      // width: '250px',
```

```

        height: '80vh',
    });
    dialogRef.afterClosed().subscribe(result => { });
}
else {
    const dialogRef = this.dialog.open(SignToJobComponent, {
        height: '80vh',
        data: { id: idJob },
    });
    dialogRef.afterClosed().subscribe(result => {
    });
}
}
}

```

4. Recommend מאפשרת לראות ולהוסיף המלצות על המשרה.

```

ngOnInit() {

    this.subscriber = this.jobService.getCompanies().subscribe(state => {
        this.Companies = state;
    });
}
company(id: number) {
    this.currentRecomend.RecomemdCompanyId=id;

    this.currentRecomend.RecomendUserId=parseInt(localStorage.getItem("UserId"));
}
addRecomend(){
    this.subscriber =
    this.jobService.addRecomend(this.currentRecomend).subscribe(state => {

    });
}
}
}

```

4. בדף Add-Job שיתוף משרה

בדיקה אם נרשם כמשתף משרה. אם לא- מנתב לכניסה כמשתף.

```

isBoss() {
    if (localStorage.getItem("isBoss"))
        this.router.navigate(['add-job']);
    else {
        const dialogRef = this.dialog.open(LoginBossComponent, {
            height: '65vh',

```

```
width: '40vw'
});
dialogRef.afterClosed().subscribe(result => {
});
}
}
```

5. פונקציות שונות – הוספת והסרת קריטריונים ומקבלת תשובה האם הוספת המסרה הצליחה

```
addParts(workspace, outnet, parts, subjectjob, role, describe, require) {
  debugger;
  this.currentJob.WSName = workspace.value;
  this.currentJob.OutNetName = outnet.value;
  this.currentJob.PartName = parts.value;
  this.currentJob.SubjectName = subjectjob.value;
  this.currentJob.JobRole = role.value;
  this.currentJob.JobDescribe = describe.value;
  this.currentJob.JobRequire = require.value;

  this.currentJob.CompanyId =
  parseInt(localStorage.getItem("BossCompanyId"));
  //
  this.currentJob.CompanyId = parseInt(localStorage.getItem("BossCompanyId"));
  this.currentJob.BossId = parseInt(localStorage.getItem("UserId"));
  debugger
  this.add();
}

add() {
  this.jobService.addJob(this.currentJob).subscribe(res => {
    Swal.fire({
      title: 'success!',
      text: 'נרשמת בהצלחה!!!',
      type: 'success',
      confirmButtonText: 'המשך'
    })
    this.router.navigate(['home']);
  },
  err => { })
}
```

6. פונקציית הוספת חברה – שולחת את החברה ומקבלת תשובה האם ההוספה הצליחה

```
addCompanyParameters(area1) {
  this.currentCompany.CompanyAreaId = area1.value;
  this.jobService.addCompany(this.currentCompany).subscribe(res => {
```

```

this.bossRe.company=res;
Swal.fire({
  title: 'success!',
  text: 'נרשמת בהצלחה!!!',
  type: 'success',
  confirmButtonText: 'המשך'
})
this.router.navigate(['home']);
},
err => {
},);
if (localStorage.getItem("bossAddCompany")) {
  this.bossRe.notRegistered = false;
  this.router.navigate(['register/register-boss']);
  this.jobService.getCompanies().subscribe(state => {
    this.bossRe.company = state;
  } );
}
}
}

```

7. בדף SmartAgent ניהול הסוכן חכם

פונקציה בעת טעינה - בדיקה האם נכנס כמחפש משורה. אם לא, פתיחת דיאלוג התחברות

```

ngOnInit() {
  this.details = true;
  this.getUser(this.userService.user);
  this.jobService.getJobParameters().subscribe(state => {
    this.jobParameters = state;
  });
  if(!localStorage.getItem("token")){
    const dialogRef = this.dialog.open(LoginUserComponent, {
      height: '65vh',
      width:'40vw'
    });
    dialogRef.afterClosed().subscribe(result => {
    });
  }
}

```

עדכון פרטי משתמש ותדירות דיוורים של הסוכן חכם

```

updateDetails() {
  this.currentUser.UserIsSmartAgent = true;
  this.userService.updateUser(this.currentUser).subscribe(res => {
    this.userService.user = res;
    Swal.fire({
      title: 'הפרטים עודכנו בהצלחה!',
      // text: this.name,
      type: 'success',
      confirmButtonText: 'המשך'
    })
  })
}

```

```
    })
  })
  this.details = false;
}
}
```

8. בדף **ForumRabanim** פורום שו"ת הלכתי קבלת השאלות הקיימות והספת שאלה חדשה.

```
ngOnInit() {
  this.forumService.getForum().subscribe(res=>{
    this.questionsList=res;
  })
}
askQuation(){
this.forumService.askQuetion(this.currentQuestion).subscribe(res=>{
})
}
```

## • ממשק מנהל

9. קבלת המשרות ואישורן

```
ngOnInit() {
  this.managerService.getJobToCheck().subscribe(res => {
    this.allJobs = res;
    this.length = res.length;
    this.fillJobs(0);
  })
}
OKTheJob(JobId: number) {
  this.managerService.okTheCheck(JobId).subscribe(res => {
    this._snackBar.open("המשרה אושרה", 'X', { duration: 6000 });
    if (res) {
      this.managerService.getJobToCheck().subscribe(res => {
        this.allJobs = res;
        this.length = res.length;
        this.fillJobs(0);
      })
    }
  })
}
```

10. הצגת הרשומים לכל משרה, לשליחת הקו"ח שלהם

```
getSignedUsers(idJob: number) {
  this.managerService.userSignedToSpecificJob(idJob).subscribe(res => {
    this.managerService.signedUser = res;
    this.managerService.jobId = idJob;
    this.managerService.companyId = this.jobSign.find(p => p.JobId ==
idJob).CompanyId;
    const dialogRef = this.dialog.open(CvToSendComponent, {
      width: '70vw',
      height: '70vh',
    });
  });
}
```

שליחה נפרדת של כל קו"ח

```
sendCv(userId) {
  this.managerService.sendCv(userId).subscribe(res => {
    if (res) {
      this._snackBar.open('קורות החיים נשלחו בהצלחה', 'X', { duration: 6000 });
      this.managerService.signedUser =
this.managerService.signedUser.filter(p => p.UserId != userId);
    }
    else
      this._snackBar.open('תקלה במערכת. נסי שוב מאוחר יותר', 'X', { duration: 6000
});
  })
  if (this.managerService.signedUser.length == 0) this.dialog.closeAll();
}
```

שליחת כל הקו"ח ביחד

```
sendAllCv() {
  debugger;
  for (let index = 0; index < this.managerService.signedUser.length;
index++) {
    this.managerService.sendCv(this.managerService.signedUser[index].UserId).su
bscribe(res => {
      if (res) {
        this.managerService.signedUser =
this.managerService.signedUser.filter(p => p.UserId !=
this.managerService.signedUser[index].UserId);
      }
    });
  }
  this.dialog.closeAll();
}
```

### 3. מדריך למשתמש:

#### 3.1 הוראות כלליות לשימוש באתר:

##### מדריך לאורח:

בעת כניסת אורח לאתר יש לו מספר אפשרויות מוגבלות:  
הוא יכול להכנס לדף הבית, לצפות במשרות הפנויות ולסנן את המתאים לו ע"פ קטריונים. להיכנס לפרטים של כל משרה, לראות את ההמלצות עליה.  
כמו כן, יכול לצפות בפורום ההלכתי ולחפש שאלות בתחומים שונים.  
וכמובן- ליצור קשר עם המערכת...

##### מדריך למשתמש:

כאשר משתמש נכנס לאתר הוא יכול לבצע הרבה פעולות מלבד אפשרויות האורח:

##### **משתמש מחפש משרה**

בפעם הראשונה שנכנס, מזין מלבד פרטי זהות גם את תחומי החיפוש שלו ומעלה קובץ קו"ח. הפרטים נשמרים ומכאן ואילך יכול:

- לצפות במשרות ולשלוח קורות חיים למעסיקים.
- לעקוב אחרי משרות שנרשם אליהם.
- להוסיף המלצות חדשות על משרות.
- להירשם למערכת "סוכן חכם" השולחת לו למייל משרות חדשות שרלוונטיות לגביו, בתדירות שהוא בוחר.
- להוסיף שאלות לפורום ההלכתי.
- משתמש רשום מקבל עדכונים למייל

##### **משתמש מפרסם משרה**

בפעם הראשונה שנכנס מזין מלבד פרטי זהות גם את פרטי מקום העבודה שלו (יכול לבחור גם מרשימה), וכן אם ברצונו להיות איש קשר. הפרטים נשמרים, ומכאן ואילך בכל פעם שנכנס תחת שמו, יכול:

- לשתף משרות חדשות לחברה שלו ולעדכן.
- להוסיף חברות חדשות.
- לקבל אליו למייל קורות חיים של המועמדים.

## מנהל האתר

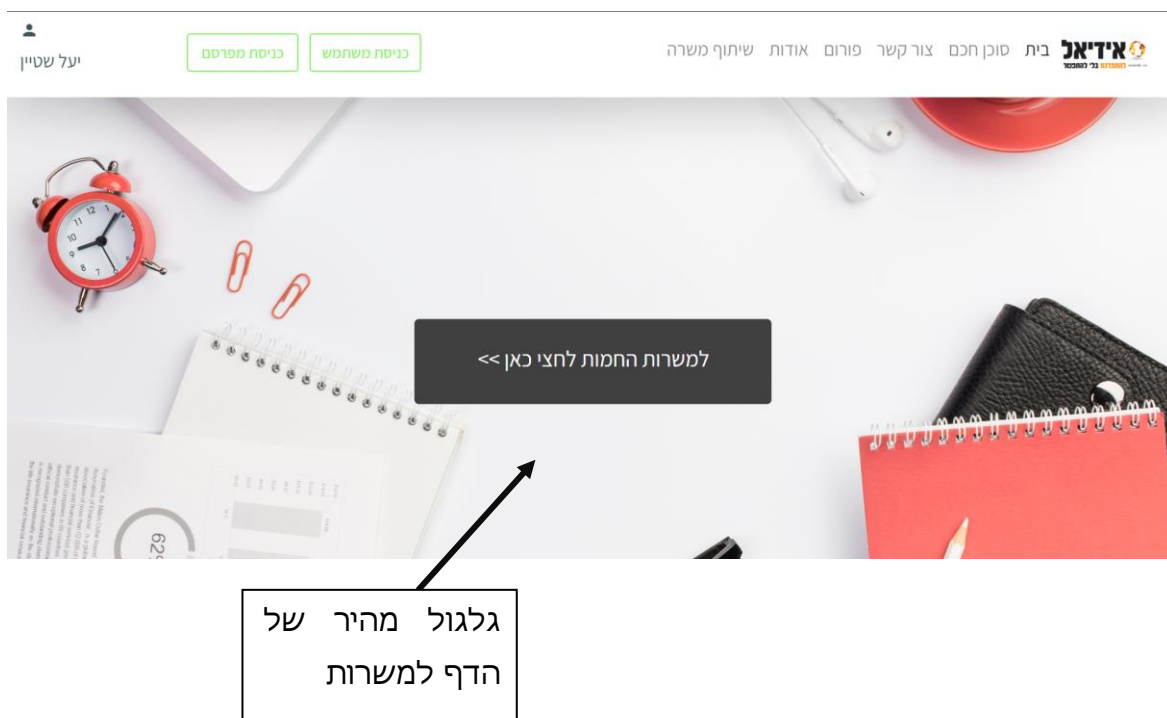
המנהל נכנס דרך הקישור במייל, מזין סיסמא, והרי האופציות שלפניו... :

- ניהול מסד הנתונים- הוספה עדכון ומחיקה של הנתונים שבאתר.
- יכול לראות איזה משתמשים נרשמו לכל משרה ולהעביר את הקורות חיים שלהם למשתתפים או למנהלי החברות.
- לראות את המשרות החדשות שנוספו ולאשרם.
- לענות תשובות על השאלות שבפורום ההלכתי.
- לצפות בנתונים סטטיסטיים על האתר.

## מסכים:

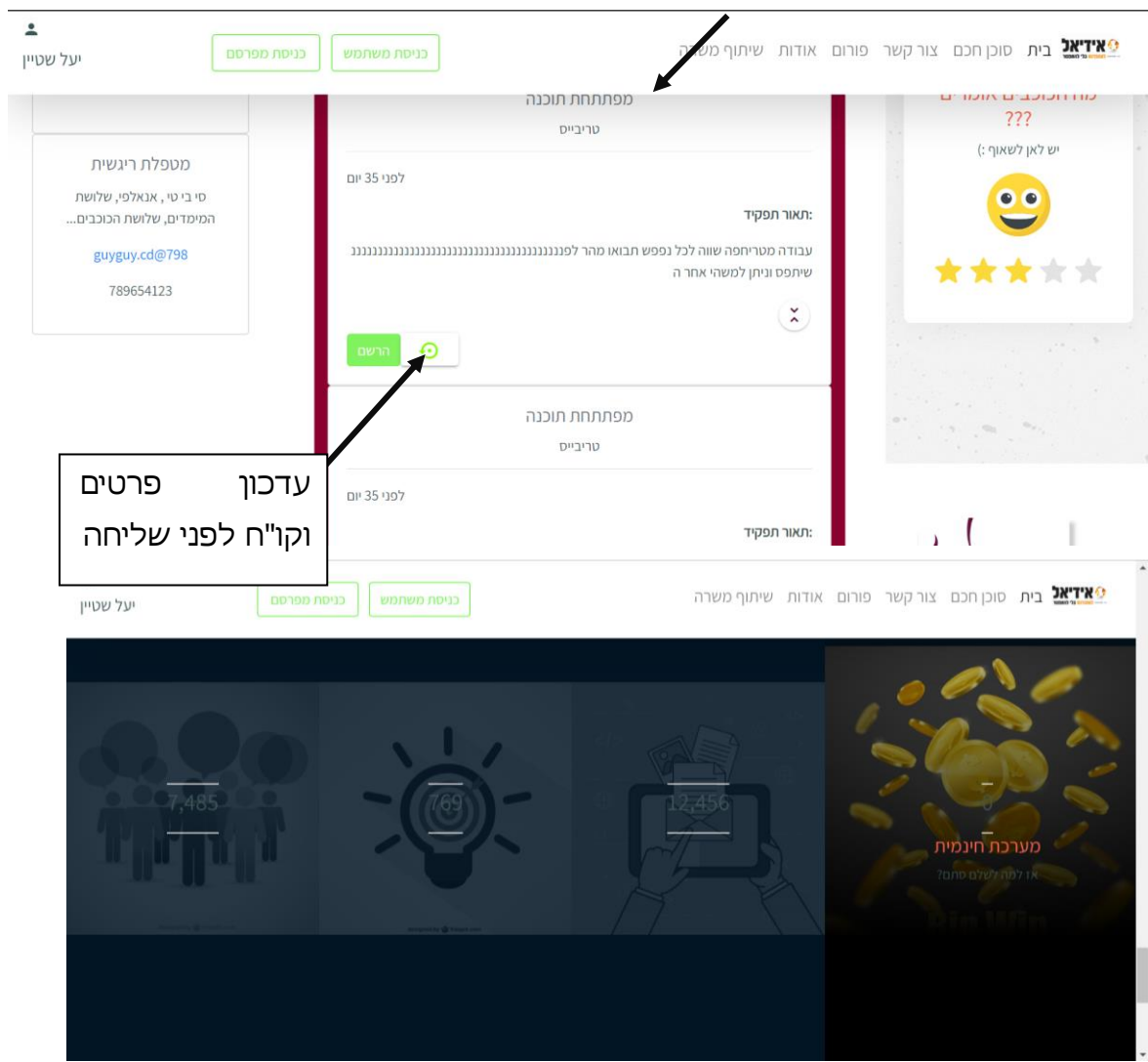
### ממשק רגיל

### מסך ראשי:



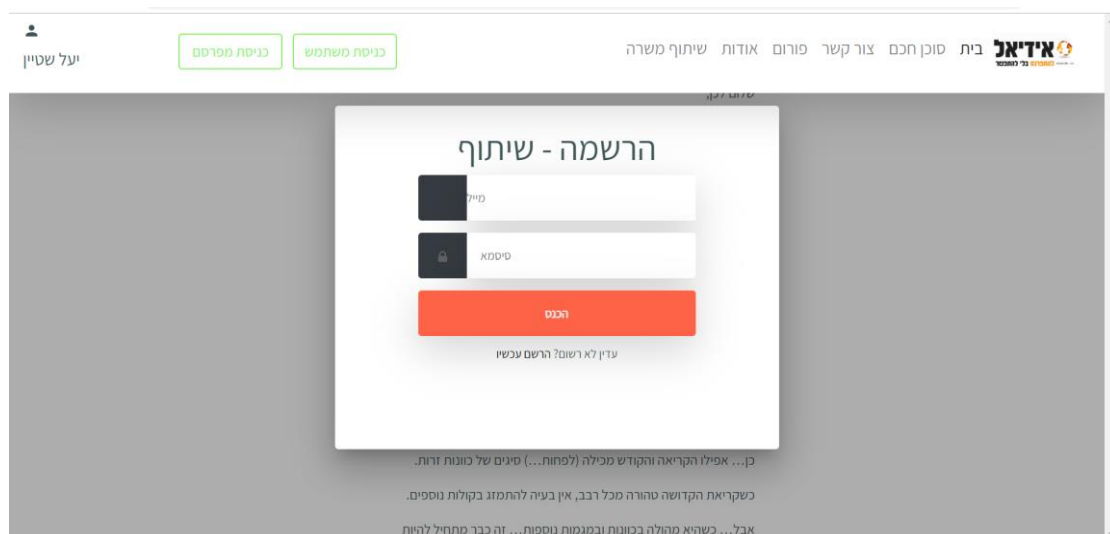



קישור  
להמלצות



הקף  
הפעילות


### 3.1.1. רישום וכניסה לאתר- מפרסם ומחפש





כניסת משתמש

כניסת מפרסם



בית | סוכן חכם | צור קשר | פורום | אודות | שיתוף משרה

הכנסת פרטי משתמש:

שם

סיסמא

טלפון


כתובת מייל

\* בחר אזור

בחר עיר


\* חשבונית חשבונית

## שיתוף משרה חדשה



כניסת משתמש

כניסת מפרסם



בית | סוכן חכם | צור קשר | פורום | אודות | שיתוף משרה

הייטק

כותרת התפקיד

תאור המשרה

פרטי משרה. ככל שתפרטי יותר יבינו אותך יותר.

כמות שנים נדרשות:

הדרישות

חידית נשית-דירוג גבוה ביותר

סוג החסימה

ללא אינטרנט

זמן ביממה שאת מעוניית לעבוד

חלקית

הרשם

המשרה בחברת:

הוספה

## פורום- צפיה בשאלות פתוחות

חיפוש נושא שאלה

הוספת שאלה חדשה למשתמש רשום

יצירת קשר

אדיאנל

המרכז הלאומי למידע

בית סוכן חכם צור קשר פורום אודות שיתוף משרה

יעל שטיין

כניסת מפרסם

כניסת משתמש

נושא הפניה

פרטי פניה

שלח

## ממשק מנהל

## צפיה במשרות המחכות לאישור

[illegible]

### 3.1.1. שליחת קו"ח

הוספת תשובות לפורום ההלכתי

3.1.1.

הוספת שאלות ותשובות מרבנים

השאלה

התשובה

נושא

▼

הוסף שאלה

## 4. סיכום ומסקנות:

### ”יגעת ומצאת תאמין”.

הפרויקט הקנה לנו מרחב הסתכלות ופרספקטיבה רחבה על פרויקט מושלם, משלב ההצעה ועד לתוצר המוגמר, בס”ד; עמלנו רבות בבניה נכונה ומסודרת של המערכת ועל תקשורת נכונה עם השרת. השקענו זמן ומאמץ מרובים בלמידת טכנולוגיות ונושאים חדשים. גילינו שמהנה (גם אם לא כ”כ קל ...) ללמוד ולהכיר תחומים חדשים, ולנסות דרכי חשיבה מגוונות ומפתיעות שלא חשבנו עליהם ממבט ראשון. למדנו לחשוב בהגיון וביעילות על מנת להשיג תוצאות במהירות. נחשפנו לתחומים שהכרנו באופן שטחי, אם בכלל, למדנו לעבוד עם Angular ועם Angular Material, bootstarp. הבנו שהחוכמה זה לא להמציא את הגלגל, אלא לקחת עליו טרמפ.

וכשבאגים צצו ללא הזמנה מוקדמת- למדנו להתגבר עליהם בחיוך. לפתור מהשורש. ולהמשיך הלאה. ומעל הכול הפנמנו שסבלנות והתמדה-אין כמותם בכל משימה הנראית קשה עד בלתי אפשרית...

ארוכה היתה הדרך, ומכשולים בה הרבה.  
אך מה שלא הרג, חישל.

## 5. נוספים:

קובץ Package.json:

```
{
  "name": "angular-idial",
  "version": "0.0.0",
  "scripts": {
    "ng": "ng",
    "start": "ng serve",
    "start_dev": "ng serve --proxy-config proxy.conf.json",
    "build": "ng build",
    "test": "ng test",
    "lint": "ng lint",
    "e2e": "ng e2e"
  },
  "private": true,
  "dependencies": {
    "@angular-devkit/build-angular": "^0.13.0",
    "@angular/animations": "^6.1.10",
    "@angular/cdk": "^7.3.7",
    "@angular/common": "^6.1.0",
    "@angular/compiler": "^6.1.0",
    "@angular/core": "^6.1.0",
    "@angular/forms": "^6.1.0",
    "@angular/http": "^6.1.0",
    "@angular/material": "^7.3.7",
    "@angular/platform-browser": "^6.1.0",
    "@angular/platform-browser-dynamic": "^6.1.0",
    "@angular/router": "^6.1.0",
    "async-sha256": "^1.0.3",
    "bootstrap": "^4.3.1",
    "core-js": "^2.5.4",
    "ng2-pdf-viewer": "^5.3.2",
    "ngx-bootstrap": "^4.1.1",
    "ngx-doc-viewer": "^0.1.20",
    "rxjs": "~6.2.0",
    "sweetalert2": "^8.12.1",
    "tingle.js": "^0.15.1",
    "zone.js": "~0.8.26"
  },
  "devDependencies": {
    "@angular/cli": "~6.2.5",
    "@angular/compiler-cli": "^8.0.0",
    "@angular/language-service": "^6.1.0",
    "@types/jasmine": "~2.8.8",
    "@types/jasminewd2": "~2.0.3",
  }
}
```

```
"@types/node": "~8.9.4",  
"codalyzer": "~4.3.0",  
"jasmine-core": "~2.99.1",  
"jasmine-spec-reporter": "~4.2.1",  
"karma": "^4.1.0",  
"karma-chrome-launcher": "~2.2.0",  
"karma-coverage-istanbul-reporter": "~2.0.1",  
"karma-jasmine": "~1.1.2",  
"karma-jasmine-html-reporter": "^0.2.2",  
"protractor": "~5.4.0",  
"ts-node": "~7.0.0",  
"tslint": "~5.11.0",  
"typescript": "^3.4.5"  
}  
}
```



## 6. ביבליוגרפיה:

אתרים בנושא תכנות:

- <https://stackoverflow.com/>
- <https://webmaster.org.il/>
- <https://material.io/>
- <https://github.com/>
- <https://yoast.com/>
- <https://codepen.io/>
-

