

**המחלקה להנדסת תוכנה  
פרויקט גמר – תשע"ז**

**מערכת ניהול משמרות  
Shifts Management System**

**מאת:  
אופיר נגדי  
גל זקן**

**מנחה אקדמי: דר' נצר יעל אישור: תאריך:**

**רכז הפרויקטים: דר' יגל ראובן אישור: תאריך:**

**אחראי תעשייתי: יוני דואיק**

|  |  |  |
| --- | --- | --- |
| # | מערכת | מיקום |
| 1 | מאגר קוד | [GitHub](https://github.com/GalZaken/Shifts-Management-System) |
| 2 | יומן | [Calendar](https://teamup.com/ksaaf83e445dd25044) |
| 3 | ניהול פרויקט | [GitHub Issue](https://github.com/GalZaken/Shifts-Management-System/issues) |
| 4 | הפצה |  |

# מבוא:

עסקים רבים מתנהלים על בסיס משמרות עובדים, ניהול המשמרות וסידור העבודה הוא חלק בלתי נמנע מהניהול השוטף של העסק.  
ברוב העסקים ניהול וקביעת סידור העבודה נעשה ע"י מנהל העבודה או עובד שהוסמך לכך ע"י המעסיק וכדומה.

הפרויקט שלנו יכלול מערכת לניהול משמרות עבור חברת גושן שירותי אבטחה, המספקת שירותי אבטחה לחברת טבע ( הר – חוצבים).

הפרויקט יהיה בנוי כאפליקציה אינטרנטית על שרת פרטי ומותאמת לתצוגה בסלולר, הכולל מערכת שלמה לניהול השותף של משמרות החברה הן בצד הניהולי והן בצד העובד.

בצד העובד – אפשרות לצפות בסידור העבודה הנוכחי והגשת משמרות לשבוע הבא עפ"י קריטריונים הנקבעים מראש ע"י מנהל החברה, הקריטריונים עלולים להשתנות מעובד לעובד לפי החלטתו של המנהל (האחראי על הגשת סידור העבודה).

בצד המנהל – המערכת מקבלת את סידור העבודה של כל העובדים, ומבצעת יצירה של סידור עבודה ראשוני (עפ"י אלגוריתם קבוע מראש הנקבע על פי בקשות ואילוצי החברה).  
המנהל מקבל את הסקיצה הראשונית של הסידור, מבצע בו שינויים כרצונו ומאשר את הסידור.

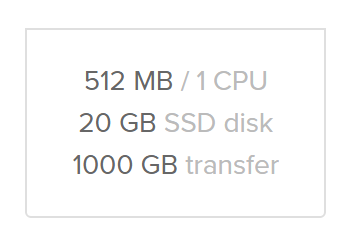
האלגוריתם – ייבנה לפי בקשת החברה.

השרת – השרת צריך לספק גישה למערכת וצריך לספק שירותי אבטחה ותהליכי רשת בהתאם.

# סביבת העבודה:

סביבת העבודה שנבחרה היא **שרת לינוקס** (Centos 7) שיכיל את קבצי האתר ויספק תהליכי רשת, תהליכי חומת אש, ותהליכים נוספים המריצים את המערכת (יפורט בהמשך).

מפרט השרת הוא:



**אף על פי** העבודה המרובה שתמונה בעבודה על שרת ריק מתוכן ובנייתו מאפס, השיקול העיקרי היה לא להיות מוגבלים מבחינת הפעולות שברצוננו לבצע (הגבלות כמו פורטים חסומים לעבודה, סביבת שרת האינטרנט, מספר מסדי נתונים וכו') – וכמו כן העדפנו שהפרויקט יהיה תחת מסגרת אחת ולא מפוצלת (שרת אינטרנט ומסד נתונים מפוצלים).

**סביבת האינטרנט** שנבחרה היא שרת NODE-JS – הסיבה לבחירת סביבה זו היא בגלל ביצועי השרת שמאוד קרוב למכונה (נכתב ב-C++ ומבוסס על מנוע V8 - JS) כמו כן הבחירה בסביבה זו היא בגלל מגוון האפשרויות הרחב הקל והיעיל שבעבודה בפרוטוקול HTTP ומול מסד נתונים .

**מסד נתונים** סביבת מסד הנתונים היא MONGO DB – סביבה זו נבחרה בגלל שניתן ליצור הכנסה מגוונת ולא מוגבלת ע"י עמודות טבלה -לעמת MYSQL וכדומה, וכן בגלל האפשרות להמיר אובייקטים למחרוזות הנתמכות ע"י מסד הנתונים.  
את העבודה מול מסד הנתונים אנו מבצעים באמצאות מודול של NODE הנקרא MONGOOSE.

**שפת התכנות** שפות התכנות שנבחרו הן: לממשק המשתמש אנו עובדים ב CSS\HTML ונעזרים בספריית העיצוב BOOTSRAP.  
פונקציונאליות המערכת נכתבת בשפת JS ובאמצאות הספריה ANGULAR-JS.

# מטרה:

מטרת הפרויקט היא הקמת מערכת WEB אשר מספקת שירות של ניהול וקביעת סידור העבודה עבור חברת גושן שירותי אבטחה, המספקת אבטחה בחברת טבע בירושלים – הר חוצבים.

השירות יענה על הדרישה של קביעת סידור העבודה ע"י אלגוריתם חכם שיתחשב בהזנת המשמרות של העובדים ובדרישות וההגבלות שנקבעו מראש ע"י אחראי האבטחה ובכך ניהול סידורי המשמרות יתבצעו באופן שוטף ע"י המערכת.  
המערכת תספק עבור העובד אפשרות להזנת המשמרות המועדפות עליו לסידור השבועי ע"י מתן פרמטרים הנקבעים מראש וגמישים לשינויים והמערכת תבנה את הסידור בהתאם למשמרות שהוזנו ובהתאם לדרישות ואילוצי המנהל.

סידור העבודה נקבע בלחיצת כפתור ובכך התעסקות של בעל התפקיד המנהל לביצוע הסידור מתקצר ומתייעל משמעותית.  
כמו כן, המערכת מספק ממשק לניהול העובדים, כמו: הוספה \ מחיקה פרופיל עובד, מתן הרשאות: אחמ"ש \ מאבטח,  
עריכת משמרת וכו'.

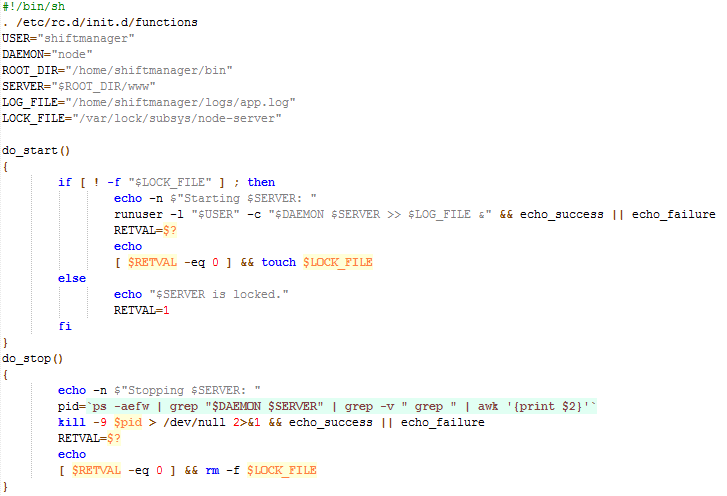
המטרה העיקרית היא בנייה של מערכת המשלבת בצורה היעילה ביותר את כל אילוצי המערכת, הן בצד הפיתוח והן מצד דרישות החברה.

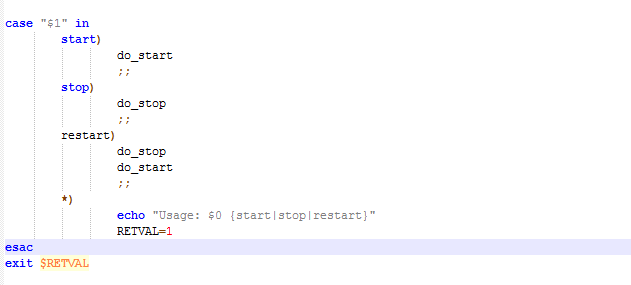
# תיאור הבעיות:

* המערכת בראש ובראשונה צריכה להיות מאובטחת ככל האפשר וצריכה לכלול הגנה מגורמים חיצוניים,  
  המערכת גם צריכה לכלול הגנה מבפנים – עובדים לא אמורים להיות מסוגלים לבצע שינויים בסידור.
* השרת צריך לדעת להריץ בכל זמן נתון את האפליקציה ברשת, ולתמוך בהפעלה כיבוי, ואתחול במידת הצורך של התהליכים השונים.
* בניית ממשק גרפי נוח לשימוש הן של העובד והן של מנהל המערכת, הממשק צריך להיות ריספונסיבי ולתמוך במגוון רזולוציות שונות, וכמו כן הממשק צריך להיות מותאם לתצוגה בטלפונים חכמים.
* העובד צריך להיות מודע לכל האילוצים שנקבעו עבורו (כמו מספר מינימלי של משמרות, מספר מינימום של משמרות לילה, מינימום משמרות סופ'ש וכו' ).  
  ויחד עם זאת הממשק צריך להיות קל ונוח לשימוש ולא לסבך את תהליך הגשת הסידור.  
  הממשק גם צריך לכלול תצוגה גרפית וזמינה לסידור העבודה הנוכחי.
* ממשק נוח וקל לתפעול של מנהל המערכת הכולל:  
  - צפייה בכל עובדי המערכת כולל הוספה, הסרה ועריכת פרטי עובד.  
  - אפשרות לקבוע קריטריונים להגשת סידור עבודה של כל עובד בנפרד על פי שיקול דעתו (תיעדוף עובד, קביעת מינימום משמרות וכו').  
  - אפשרות לקביעת סידור עבודה בצורה פשוטה ויעילה, ע"י ביצוע שינויים ידניים לפני הפצת סידור העבודה לאחר סקיצה ראשונית של הסידור שבוצע ע"י המערכת.  
  - הוספה והסרת של עמדות תוך כדי מתן עדיפות לעמדה מסוימת לשיקולי האלגוריתם.
* אלגוריתם לקביעה ויצירת סידור עבודה - האלגוריתם צריך להיות חכם ככל האפשר וצריך לדעת להיות גמיש לשינויים.  
  האלגוריתם צריך לדעת לשבץ את העובדים לפי עדיפויות שונות הנקבעות ע"י מנהל המערכת (יפורט בתיאור הפתרון בהרחבה).

# תיאור הפתרון:

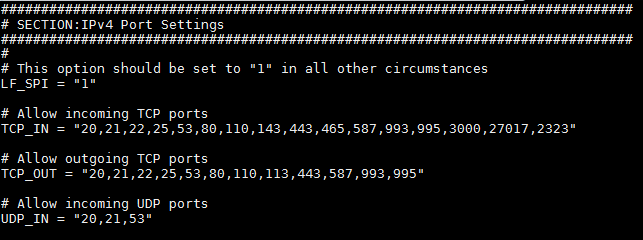
* על מנת לאבטח את השרת הותקנה מערכת חומת אש (CSF) שחוסמת גישה לפרוטוקולי תקשורת השונים (FTP, SSH וכו') מחוץ לישראל.  
  מערכת חומת האש מכילה אפשרויות קונפיגורציה רבות ולכן גמישה לשינויים על פי שיקול מפתחי המערכת.  
  נספח קונפיגורציה להמחשה:





* כדי שהשרת יוכל להריץ בכל זמן נתון את אפליקציית הרשת וכדי שיתמוך בכיבוי, הפעלה אתחול כתבנו סקריפט ב BASH אשר רץ יחד עם הפעלת המערכת והיודע לטפל בנ"ל.

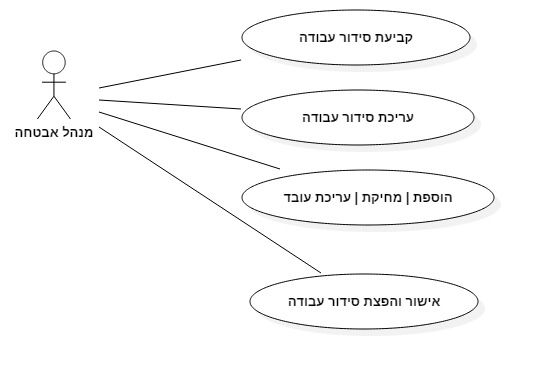
נספחים:

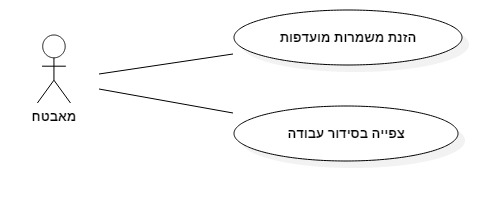


**הפתרון מבחינה הנדסית הוא מימוש ע"פ המודל של שרת-לקוח:**

* שרת NodeJS – שרת אשר יעבוד באופן יעיל בזמן אמת עבור הפעולות הנדרשות ע"י המערכת.  
  שפת השרת היא JavaScript ולכן יהיה קל יותר להתנהל עם הלוגיקה של האפליקציה אותה בחרנו לכתוב גם ב-JavaScript.
* מסד נתונים MongoDB – מסד נתונים בעל מבני נתונים גמיש הבנוי על מבנה של מסמך הממומש מעל JSON,  
  פורמט טקסטואלי נוח להעברת נתונים ומידע בין שרת ללקוח, מתאים לעבודה עם אפליקציות WEB, כמו כן,  
  שפת בסיס הנתונים היא JavaScript ושימוש ב-JSON מתבצע באופן טבעי בשפה זו.

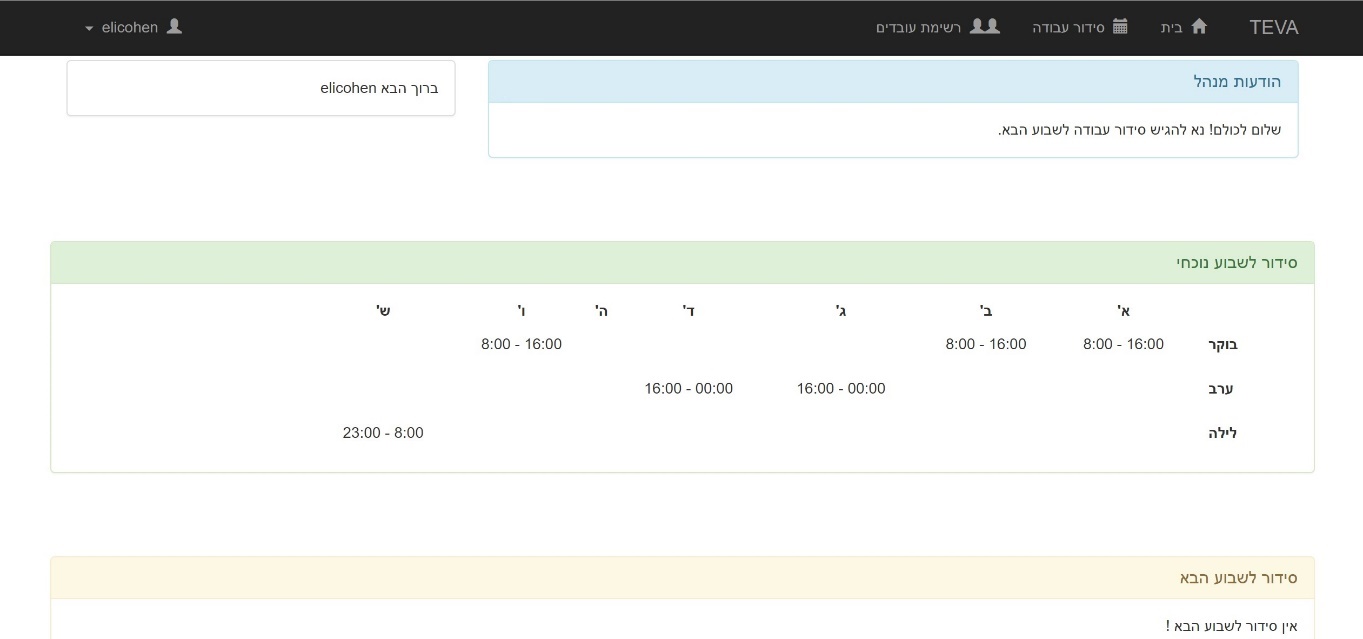
**תרשימי שימוש:**

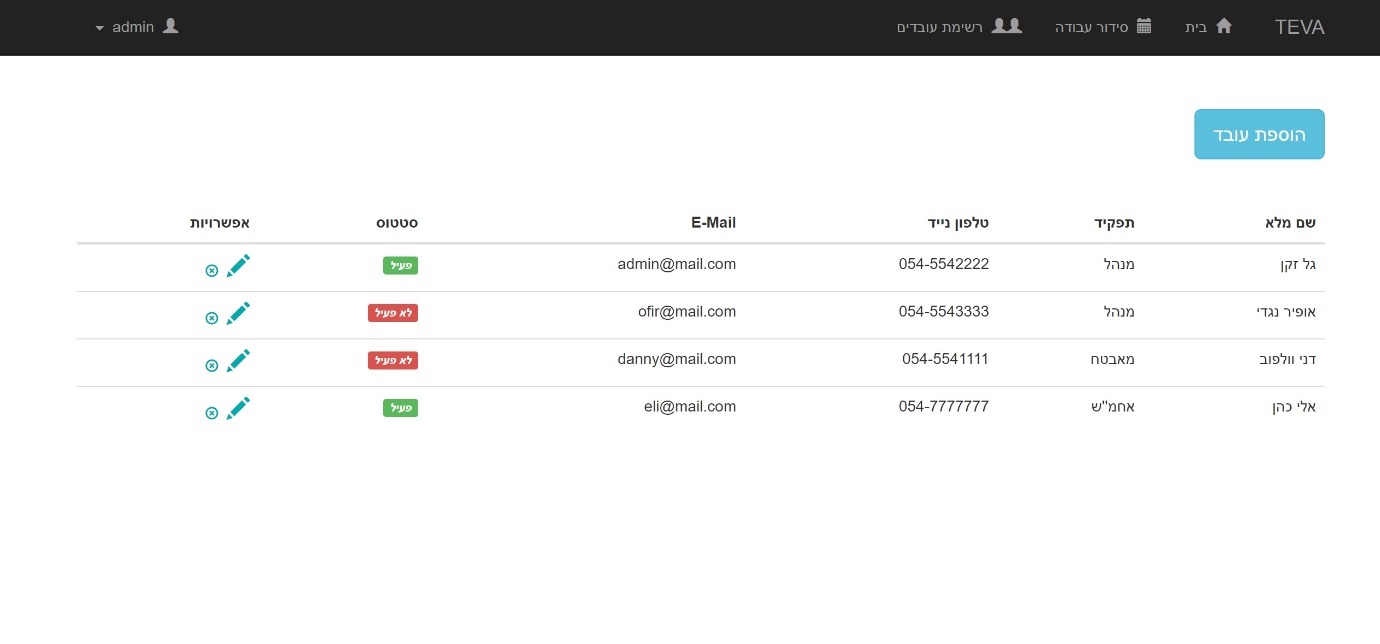
****



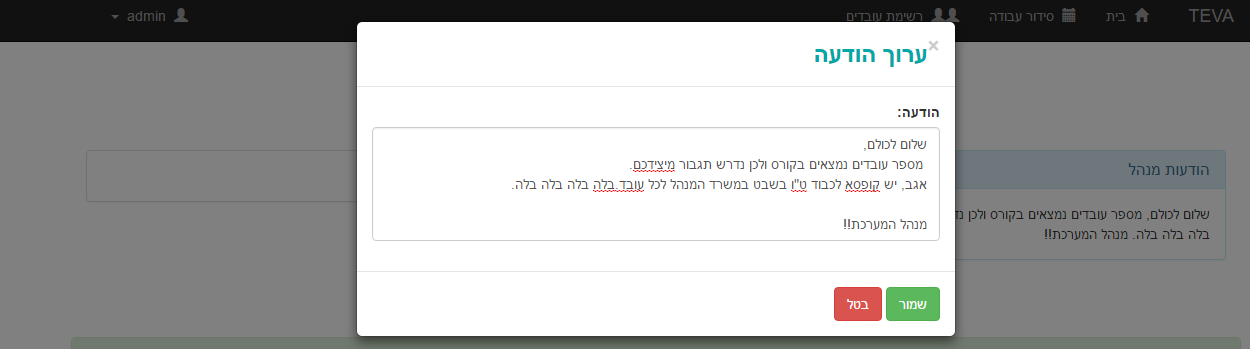
* **הממשק הגרפי** – כדי לספק חווית משתמש בצורה הטובה ביותר בחרנו לכתוב את ה- UI באמצאות שימוש בשפות CSS\HTML תוך כדי שימוש בסיפריה BOOTSTRAP - המאפשרת תצוגה ריספונסיבית ונוחה לממשקים שונים, כולל תמיכה בטלפונים חכמים.

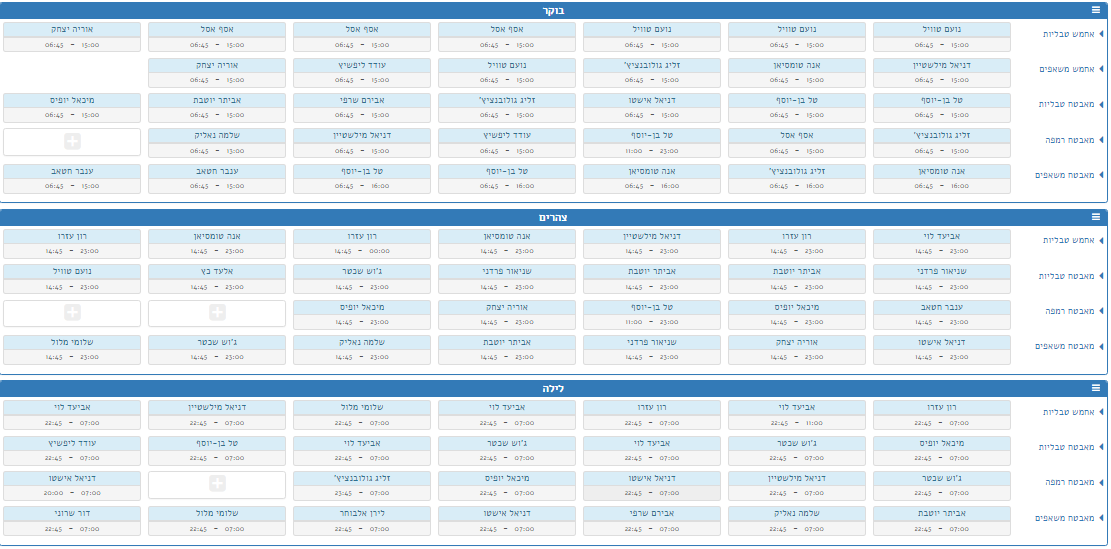




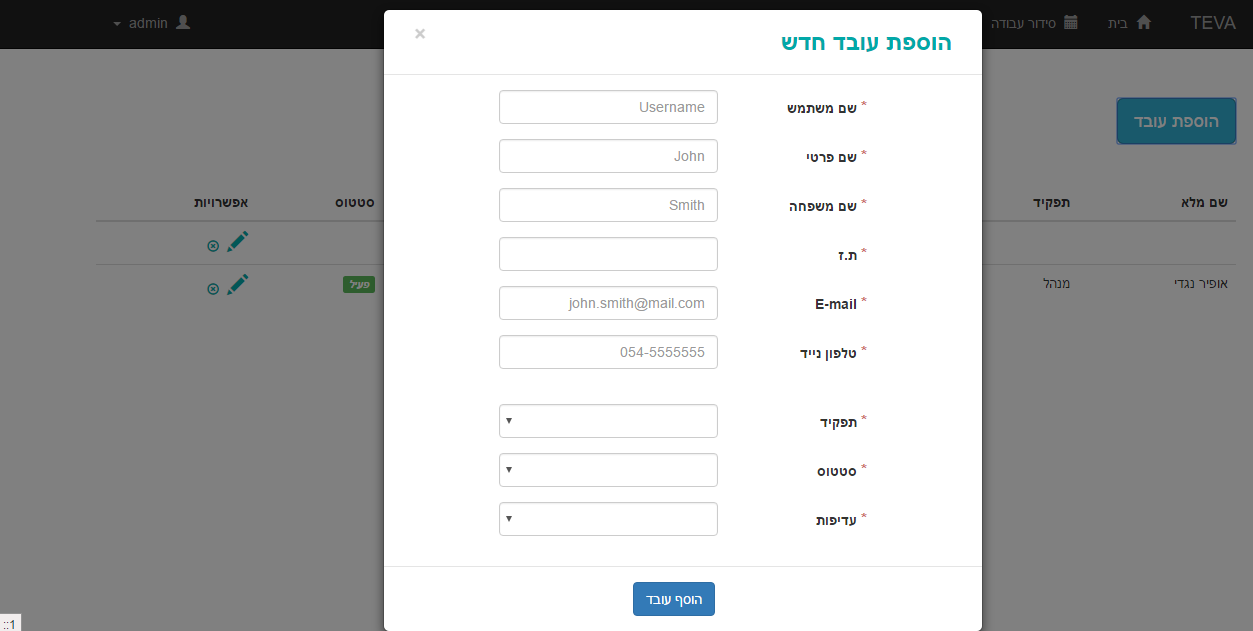


* **הגשת סידור העובד –** כאשר מנהל המערכת ייצור עובד חדש, יצירת העובד מותנית בקביעת אילוצים,  
  האילוצים על פי דרישת הלקוח הם:  
  -מספר מינימלי של משמרות להגשה.  
  -מספר מינימלי של משמרות לילה.  
  -מספר מינימלי של משמרות בסוף שבוע.  
  -תיעדוף העובד לקבלת משמרת – המשמעות היא שאם ישנם מספר העומדים בקריטריונים של האלגוריתם לקבלת משמרת מסויימת, העובד בעל התיעדוף הגבוהה יותר יקבל את המשמרת.  
  המערכת החכמה תבצע אותנטיקציה של הדרישות הנ"ל ועובד חדש יתווסף למערכת רק אם כל השדות הנ"ל מולאו באופן תקין.  
  לאחר מכן, יוכל העובד לגשת לדף הגשת סידור העבודה כאשר הוא יודע מהם האילוצים לגביו (המערכת תציג את האילוצים בממשק הגרפי), ובמידה וההגשה תקינה המערכת תאפשר לעובד להגיש את הסידור.  
  כמו כן המערכת תאפשר ביצוע שינויים בהגשת הסידור עד למועד האחרון שנקבע.
* **ניהול העובדים - המטרה העיקרית של הפרויקט היא להקל על מנהל המערכת ביצירת סידור העבודה,**ולכן המערכת צריכה להיות גם חכמה וגם נוחה ליצירת סידור העבודה השבועי.  
  ממשקי המערכת:  
  - למנהל ישנה אפשרות לכתוב הודעות גלובליות לכל העובדים במערכת.



- ממשק נוח לצפייה בסידור העבודה המחולק לפי משמרות (בוקר, צהריים\ערב ולילה) ולפי המחלקות השונות

* ממשק נוח להוספת עובדים וצפייה בהם ועריכת פרטים אישיים



* המערכת צריכה להיות גמישה להוספת עמדות במידת הצורך – גם לעמדות יינתנו תיעדופים ( אופן התיעדופים ושילובם באלגוריתם ייקבע בשלב מאוחר יותר).
* לאחר ביצוע סידור עבודה ע"י המערכת (חישוב האלגוריתם ע" התחשבות באילוצים והפצת סקיצת סידור ראשונית) על המנהל לבצע שינויים בסידור עבודה במידת הצורך ולאשר אותו.  
  לאחר האישור הסופי המערכת תפיץ את הסידור והעובדים יוכלו לצפות בו.
* **האלגוריתם** – תפקיד המערכת לאסוף את כל הנתונים למספר Collection’s בצורה חכמה ככל האפשר.  
  לאחר איסוף כל הנתונים יחל תהליך חישוב סידור העבודה.  
  בגלל שמספר העובדים ידוע מראש (ובד"כ מספר דיי קטן לחישובים מתמטיים, כ 20-30 עובדים) קל יותר לבצע חישוב של סידור העבודה.  
  - האלגוריתם ראשית יצור מערך\רשימה של מטריצות המכילה את כל ימי השבוע כאשר כל יום מחולק ל-3 משמרות (בוקר, צהריים וערב).  
  - כל משמרת תכיל את כל העמדות האפשריות לאותה משמרת.  
  - כל משמרת תמולא בכל העובדים שהגישו את אותה המשמרת ותספור את כמות העובדים בכל משמרת.  
  - לאחר מכן האלגוריתם יעבור על המשמרות לפי התיעדוף שלהם ( משמרות לילה וסופש יקבלו כנראה תיעדוף גבוה יותר).

- במידה ויש מספר משמרות בעלי אותו תיעדוף, המשמרת בעלת "מספר ההגשות" הגבוה יותר תקבל עדיפות גבוה יותר.  
- העובד בעל העדיפות הגבוהה יותר יקבל קודם את המשמרת ( תיעדוף עובד נקבע ביצירת העובד).  
- במידה ויש 2 עובדים בעלי תיעדוף זהה "נקודות זכות" יינתנו לבעל תפקיד בכיר יותר (אחמ"ש) ולאחר מכן למי שהגיש כמות גדולה יותר של משמרות.  
- כל עובד שנבחר יסומן ב"דגל" עבור אותו יום שנבחר ובכך האלגוריתם יוכל לטפל בכפילויות (במידה ועובד יכול לעבוד בכל המשמרות באותו יום) ולשמור על השמת המשמרות במסגרת החוק.  
- לאחר יצירת הסקיצה הראשונית הסידור מובא לאישור המנהל.  
- במידה והמנהל מבצע שינויים חריגים המערכת תתקיע – כמו משמרת כפולה, חריגה מהשעות שמאפשר החוק וכו'.  
- לאחר אישור המנהל סידור העבודה יופץ.  
- סך הכל האלגוריתם ירוץ 3 פעמים על ימי השבוע ומספר העובדים, במקרה הכי גרוע (כל העובדים הגישו את כל המשמרות) כיוון שסך העובדים ידוע מראש לכל היותר המערכת תבצע 3\*3\*7 (3 ריצות כפול 3 משמרות כפול 7 ימים) כפול מספר העובדים כפול מספר העמדות, כאשר לכל משמרת, מקס' 50 עובדים, וכאשר המשמרת העמוסה ביותר מאיישת 5 עמדות (כ 600 אלף פעולות), מעבד פשוט של 512MB\s יבצע זאת תוך 2-3 שניות.  
כיום יש בחברה כ – 24 עובדים פעילים, המגישים בממוצע כ-10 משמרות לפי חישוב זה זמן ריצת האלגוריתם הוא נמוך בהרבה.

# תכנית בדיקות:

**מדדים להצלחה:**

1. מהירות.

2. תאימות קלט – אותנטיקציה תקינה.

4. יציבות.

**טסטים חיובים:**

* ניסיונות התחברות תקינות למערכת ממשתמשים שונים.

בצד המנהל:

* בדיקות הוספת עובדים עם פרמטרים שונים ובדיקת העדכון במסד הנתונים.
* בדיקות עריכת עובדים ובדיקת העדכון במסד הנתונים.
* מחיקות עובדים ובדיקת העדכון במסד הנתונים.
* בדיקת תרחישים תקינים ליצירת והפצת סידור העבודה ושמירתו במערכת.
* בדיקת חוקיות הסידור לפי קריטריוני האלגוריתם.
* בדיקת חוקיות הסידור לפי קריטריוני משרד העבודה.
* בדיקות יציבות עבור סידורי עבודה גדולים מאוד.
* ביצוע הוספת עמדות וקביעת תיעדופים.

בצד הלקוח:

* הגשות תקינות של סידור העבודה.
* בדיקת מעבר תקין ורציף בין העמודים השונים.

**טסטים שלילים:**

* ניסיונות התחברות עם פרטים שגויים, וביצוע בדיקות אבטחה.
* ניסיונות להגיע לדפים פנימיים ללא חיבור למערכת.
* הוספת \ עריכת עובד עם פרטים שגויים.
* הגשות סידור עבודה החורגות מהקריטריונים.

**בדיקות יחידה:**

נחלק את הבדיקות ל – 2 חלקים עיקריים: בדיקות בצד העובד, ובדיקות בצד המנהל.  
ואת הבדיקות האלה נפרק לתתי בדיקות:

בצד המנהל נפרק ל:

* בדיקות אבטחה.
* בדיקות סכמת העובדים.
* בדיקות סכמת הסידור.
* בדיקות אלגוריתם.

בצד העובד:

* בדיקות הגשת סידור.
* בדיקות בממשק הגרפי.

**כלים לביצוע בדיקות:**

MOCHA - A **test** framework running on node.js and the browser which runs **tests** serially, mapping uncaught exceptions to the correct **test** cases.

בדיקות ידניות שונות במערכת הלינוקס.

# לוחות זמנים וניהול סיכונים:

* איפיון, תיכון ועיצוב מבנה הפרויקט – עד 2 שבועות: 11.12.16  
  **סיכונים:** איפיון לא מתאים או לוקה בחסר, תיכון ומבנה לא טוב.  
  **סבירות:** נמוכה.
* בניית שרת ותחברות למסד נתונים – עד 2 שבועות: 25.12.16  
  **סיכונים:** זמן רב ללימוד וישום בנית השרת בצורה תקינה.  
  **סבירות:** בינונית.
* בניית אב טיפוס הכולל ממשק משתמש ראשוני עבור המנהל (אחראי) – עד 3 שבועות: 15.1.17  
  **סיכונים:** אי עמידה בזמנים של המרכיבים, שרת, ממשק המנהל והאינטגרציה ביניהם.  
  **סבירות:** גבוהה.
* תכנון ובניית האלגוריתם לקביעת סידור העבודה – עד 3 שבועות: 5.2.17  
  **סיכונים:** זמן רב למציאת אלגוריתם \ בניית אלגוריתם מתאים שיענה על הדרישות, הבנת האלגוריתם והתאמתו למע'.  
  **סבירות:** גבוהה.
* בניית ממשק משתמש עבור עובד (מאבטח) הכולל פיצ'ר להזנת המשמרות – עד 3 שבועות: 26.2.17  
  **סיכונים:** קושי בבנית הפיצ'ר כך שיתאים לאלגוריתם סידור העבודה ובנוסף יהיה ידידותי למשתמש.  
  **סבירות:** בינוני-גבוהה.
* אינטגרציה בין ממשק המשתמש לבין האלגוריתם לקביעת סידור העבודה - עד 3 שבועות: 19.3.17  
  **סיכונים:** באגים בלתי צפויים, תכנון לא נכון או לוקה בחסר של הלוגיקה אשר יוביל לבעיה בשילוב בין השניים.  
  **סבירות:** בינונית.
* בדיקות קצה והתנהלות שותפת של המערכת לאחר האינטגרציה – עד 2 שבועות: 2.4.17  
  **סיכונים:** קושי בשילוב המרכיבים יחד או אי עמידה בזמנים של כל המרכיבים הנדרשים.  
  **סבירות:** גבוהה.
* בניית האפליקציה ופיצ'ר להזנת משמרות ע"י העובד (מאבטח) – עד 3 שבועות: 23.4.17  
  **סיכונים:** בעיה באינטגרציה בין הפלטפורמות.  
  **סבירות:** גבוהה.
* אינטגרציה סופית של המערכת – עד 4 שבועות: 21.5.17  
  **סיכונים:** אי עמידה בזמנים, קושי בשילוב הסופי.  
  **סבירות:** בינונית.
* בדיקות עבור התנהלות שוטפת של המערכת ותיקונים במידת הצורך – עד 4 שבועות: 18.6.17  
  **סיכונים:** באגים שלא צפינו אשר ישפיעו על תפקודה השוטף של המערכת.  
  **סבירות:** בינונית.

# אפליקציות ושירותים דומים:

* EasyShift – תוכנה לניהול וסידור עבודה – אתר ישראלי המספק שירות לניהול וסידור משמרות.
* ShiftOraganizer - תוכנה לניהול וסידור עבודה – אתר ישראלי המספק שירות לניהול וסידור משמרות.

# סיכום:

עד כה הצלחנו להקים שרת מאובטח מבוסס מערכת לינוקס, בעל חומת אש חזקה ו – NETWORK SERVICE.  
יישמנו פלטפורמתNODE-JS המתממשק עם מסד נתונים MONGO DB.

הצלחנו ליצור ממשק נוח וידידותי הן עבור העובד והן עבור המנהל.  
הממשק מכיל ממשק התחברות מאובטח, המזהה את המתחבר (מנהל או עובד) ומציג ממשק התחברות בהתאם.  
הממשק מכיל דף בית נוח וידידותי הן למנהל והן לעובד.  
הממשק ריספונסיבי ומותאם למסכים שונים כולל טלפונים חכמים וטאבלטים.  
ממשק המנהל מאפשר הוספה, עריכה ומחיקת עובדים ומסנכרן בצורה תקינה מול מסד הנתונים.

להמשך הפרוייקט:

* יש להוסיף פרמטרים נוספים ליצירת העובד אשר יסופקו ע"י דרישות הלקוח.
* יש ליצור ממשק נוח לתצוגת סידור העבודה בצד המנהל הכולל הוספת עמדות נוספות ותיעדופם.
* יש לממש את אלגוריתם סידור העבודה.
* לבצע בדיקות ותיקונים במידת הצורך.

# ספרות:

https://nodejs.org/en/

<https://en.wikipedia.org/wiki/Nurse_scheduling_problem>

<http://docs.jboss.org/optaplanner/release/latest/optaplanner-docs/html_single/index.html#tabuSearch>

https://www.youtube.com/watch?v=7nPagqJK3bs

<https://semaphoreci.com/community/tutorials/getting-started-with-node-js-and-mocha>

<https://mochajs.org/>

# תרשימים:

