* DB
* 2 טבלאות (ממתינים/סוגי סטטוס)
* 3 פרוצדורות
* קבלת כל הממתינים בתנאי שהסטטוס שלהם לא שווה ל-2 מסודר לפי הראשון בתור
* הוספה/עדכון ממתין חדש (מחליט על הוספה או עדכון לפי בדיקה האם נשלח לו ID)
* בדיקת פרוצדורות - בודק האם הפרוצדורה ששלחו למערכת להריץ ב-DB אינה פרוצדורת מערכת.
* DAL
* 2 פונקציות קריאה ל-DB (עם/ובלי פרמטרים)
* Controllers
* קונטרולר אחד לשם קריאה ל-DB (RequestController)
* Models/BL
* Class של RequestBL בו מוטמע המנגנון של הקריאה ל-DB
* Views (אנגולר)
* Index.html
* App.js
* getData – מביא את כל הממתינים ומחלק אותם לשני מערכים (ממתינים/בטיפול)
* addCustomer – קורא ל- Api עם אובייקט של לקוח חדש ומוסיף ל-DB
* customerNext – קורא ל- Api עם אובייקט של הלקוח הראשון בתור מהמערך של הממתינים ומשנה לו את הסטטוס ושולח ל- DB.
* לאחר מכן קורא שוב ל- Api עם אובייקט של הלקוח הראשון בתור מהמערך של ה-בטיפול ומשנה לו את הסטטוס ושולח ל- DB.
* customerFinis (יופעל במידה ואין עוד לקוחות ממתינים ויש רק לקוח אחרון בטיפול)- קורא ל- Api עם אובייקט של הלקוח הראשון בתור מהמערך של ה-בטיפול ומשנה לו את הסטטוס ושולח ל- DB.
* server Api - פניות לשרת
* פונקציה להמרת ה-type של html ל-type של sql
* פונקצייה $http לקריאה לשרת מקבל: שם הפרוצדורה, ומערך של פרמטרים שנוצרים מהאובייקט ששלחתי לפונקצייה הקודמת