ינאי אלבג – תיק פרויקט מדעי המחשב

1. אפיון ותכנון הפרויקט

* שם הפרויקט: TaskMasterIL
* תיאור כללי: "TaskMasterIL" הוא כלי ניהול משימות ומטלות המאפשר למשתמשים ליצור רשימת משימות. הכלי עוזר למשתמשים לנהל את זמנם ביעילות ולא לשכוח לגבי המטרות שלהם.
* בעיה שהפרויקט פותר: אנשים רבים מתקשים לארגן את המשימות היומיומיות שלהם ולעיתים שוכחים לגבי מטלות חשובות. ניהול זמן לקוי עלול להוביל ללחץ ותחושת חוסר יכולת לעמוד בלוחות זמנים. בנוסף, בלי מערכת לעקוב אחרי התקדמות המשימות, קשה לדעת אילו משימות הושלמו ואילו עדיין נותרו לביצוע.
* קהל יעד: כלי זה נועד לעזור לאנשים עסוקים, עובדים וסטודנטים שמחפשים לשפר את ניהול הזמן והיעילות האישית.

דרישות פונקציונליות:

* המשתמש יכול להירשם ולהתחבר למערכת.
* המשתמש יכול להוסיף משימות לפי סדר עדיפות, למחוק אותן או לסמן אותן כמבוצעות.

דרישות טכניות:

* Frontend: HTML, CSS, JavaScript
* Backend: Node.js, Express,
* Database: sqlite3

2. תכנון מבנה האפליקציה

מבנה תיקיות:

/TaskMasterIL

| -- /node\_modules #קבצי node.js

--/public |#קבצי HTML

| --/server #קבצים הקשורים ל-database

| --/server/db#טבלאות database

| --/style#קבצי עיצוב של CSS

|--LICENSE#רישיון תוכנה

|--package-lock.json#חבילות הקוד

|--package.json#הגדרות הפרויקט

|--server.js#מסד נתונים

הסבר על כל תיקייה:

/node\_modules – מכיל את הספריות שהפרויקט צריך כדי לפעול.

/public – מכיל את קבצי הHTML של דפי האתר.

/server – מכיל את קבצי הdatabase הנדרשים.

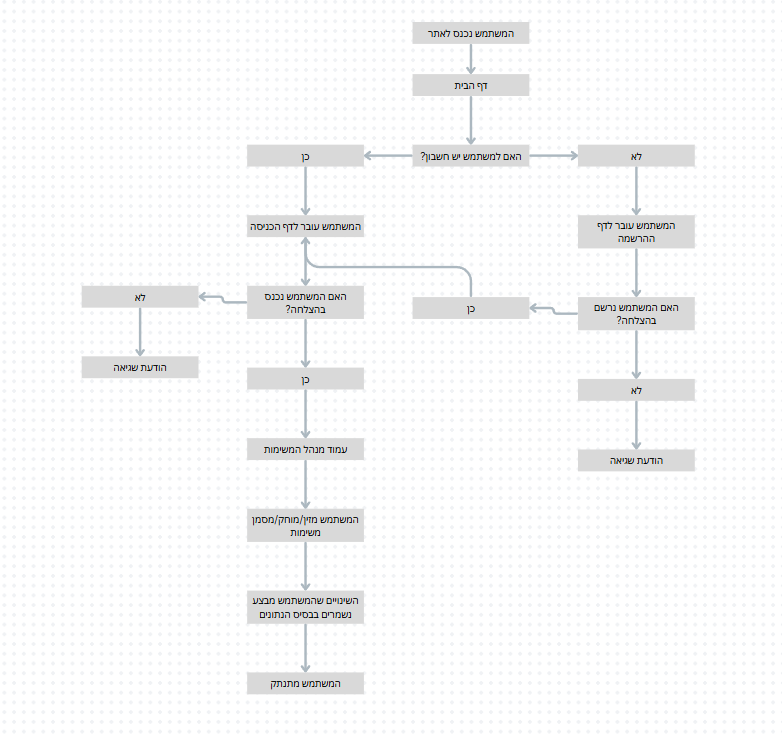
/server/db – מכיל את טבלאות הdatabase.

/style – מכיל את קבצי הCSS שאחראיים על עיצובו של האתר.

מסכים עיקריים:

* עמוד הבית – מכיל הסבר קצר על השירות ועל יתרונותיו.
* עמוד התחברות – מאפשר כניסה למערכת.
* עמוד מנהל המשימות – השירות עצמו; מציג משימות שהמשתמש הזין.

תרשים זרימה של הניווט:

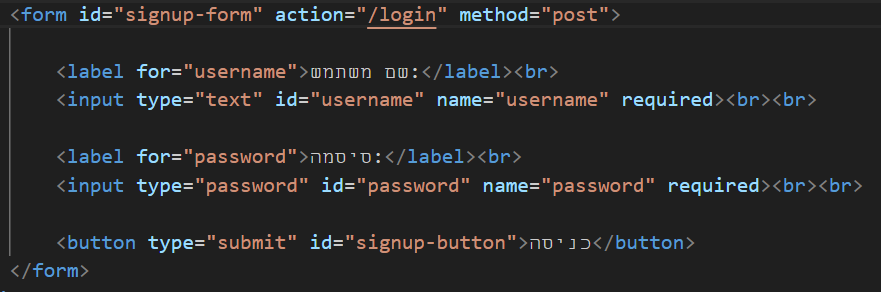


3. פיתוח צד לקוח

הדפים העיקריים באתר:

* Index.html – עמוד הבית
* Login.html – עמוד המאפשר ההתחברות
* TaskMaster.html – עמוד מנהל המשימות של המשתמש

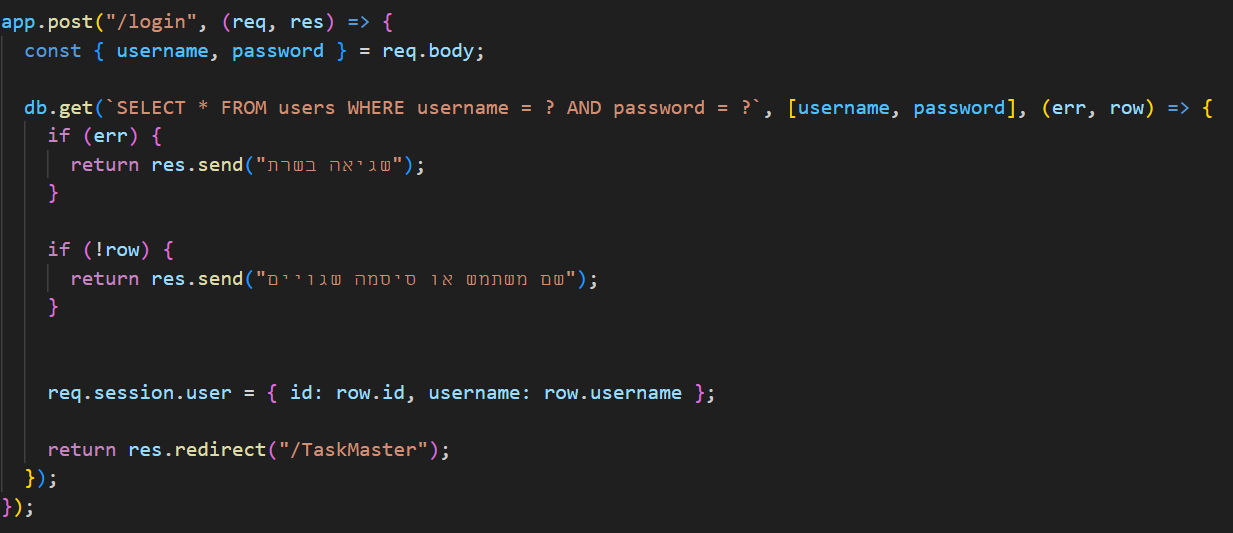
3.1 קטע קוד לדף התחברות ((login.html:



הסבר מפורט על כל רכיב בקוד:

* <Form> – תגית זו מגדירה תופס HTML לאיסוף הנתונים מהמשתמש.
* <Label> - מספקת תווית לשדה קלט, כך שהמשתמש יבין מה צריך להזין.
* <input type = "username"> - שדה קלט ייחודי לשם משתמש.
* <input type = "password"> - שדה קלט לסיסמה, מסתיר את מה שהמשתמש כותב בשדה דרך נקודות.
* Required – מבטיח שהמשתמש לא ישאיר את השדה ריק לפני שליחת הטופס.
* <button type = "submit"> - כפתור לשליחת הנתונים לרשת.

3.2 קטע קוד JavaScript (frontend):

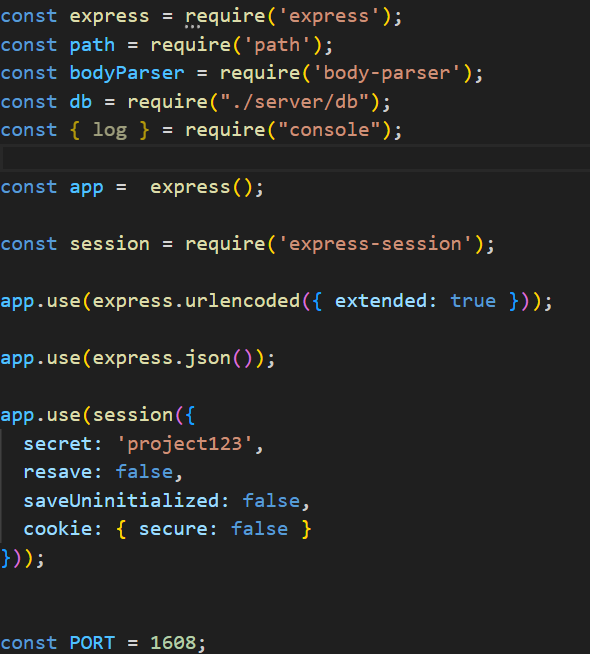


הסבר מפורט על כל רכיב בקוד:

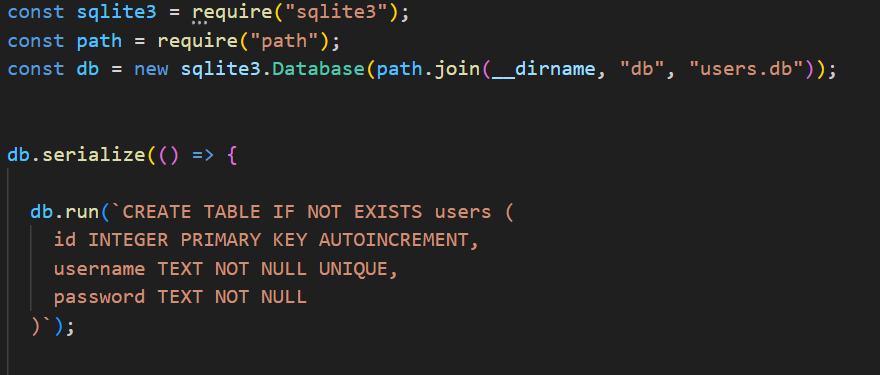
* app.post(...) - מגדיר מסלול לטיפול בבקשה מסוג POST לכתובת /login.
* const { username, password } = req.body - שליפה של המשתנים username ו־password מתוך גוף הבקשה (req.body), הערכים הללו הינם הערכים שהמשתמש הזין בטופס ההתחברות.
* If(err){…} – אם הייתה שגיאה, כנראה במסד הנתונים, למשתמש תועבר הודעה אשר מודיעה לו שהתרחשה שגיאה בשרת.
* If(!row){…} – אם שם המשתמש לא קיים, או שהסיסמה שגויה, למשתמש תועבר הודעה אשר מודיעה לו על כך.
* return res.redirect("/TaskMaster") – אילו כל התנאים נפגשו, המשתמש מועבר לדף של השירות עצמו, מנהל המשימות.

4. פיתוח צד שרת (backend)

4.1 יצירת שרת (Node.js):

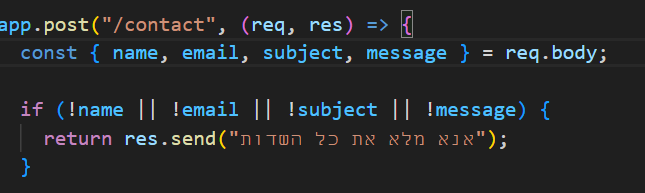


5. תכנון מסד נתונים

5.1 מבנה מסד נתונים:  


6. בדיקות ואבטחת מידע

דוגמא לקוד בדיקה:



אם המשתמש השאיר שדה ריק בפנייתו, תועבר לו הודעת שגיאה.