עבודה מסכמת בקורס פיתוח צד לקוח

עבודה בקורס פיתוח מערכות אינטרנטיות 1 סמסטר א' תשפ"ה, מטלה מסכמת JS.

מגישים:

ניב אלכס, מייל: nivalex.01@gmail.com

דן חילקביץ', מייל: danchuk213@gmail.com

<u>תיאור כללי של המערכת (לקורא הרגיל):</u>

מערכת ניהול כרטיס האשראי שלנו מאפשרת למשתמשים לבצע מעקב שוטף על תנועות האשראי שלהם ולנהל את ההוצאות שלהם בצורה יעילה יותר. המערכת שלנו מורכבת מ-8 מסכים.

מסך דף הבית (HomePage) – זהו המסך שמוצג לכל המשתמשים בכניסה הראשונה לאתר (בין אם הם רשומים ובין אם לא). המסך מציג בקצרה תיאור תמציתי של המערכת וכפתור שבלחיצה עליו מוצגים הפיצ'רים שהאתר שלנו מציע למשתמש. לחיצה על כל פיצ'ר תוביל לעמוד הייעודי שלו.

מסך הרשמה (RegisterPage) – זהו מסך ההרשמה למערכת. במסך מוצג טופס הרשמה בו המשתמש נדרש למלא אימייל חוקי, סיסמא תקינה (סיסמא תקינה היא סיסמא הכוללת 8 תווים, אות אחת גדולה, אות אחת קטנה ותו מיוחד אחד, למשל: !12345Aa היא סיסמא תקינה). תאריך לידה (על המשתמש להיות בגיל 16 לפחות), כרטיס אשראי (כרטיס אשראי מכיל 16 ספרות) ותוקף בפורמט mm/yy. לאחר שמשתמש ממלא את הטופס בהצלחה הוא רשום לאתר שלנו.

מסך התחברות (LoginPage) – זהו מסך התחברות בסיסי לאתר, המשתמש נדרש להזין את האימייל והסיסמא איתה ביצע את ההרשמה ובכך להתחבר לאתר.

מסך דשבורד (DashboardPage) – לאחר שהמשתמש מבצע התחברות למערכת הוא פוגש את מסך הדשבורד. במסך זה מוצגת למשתמש הודעת שלום (עם המייל שלו והתאריך + השעה המעודכנים) ובנוסף המשתמש יכול לצפות בחיובים שלו. יוצג למשתמש העסקה הקודמת שביצע (Previous Bill), החיוב הבא שירד מהכרטיס שלו (Upcoming Bill), סכום החיוב של החודש הקודם (Previous month Bill amount) וסכום החיוב הצפוי לחודש הבא

(Upcoming month Bill amount). בנוסף תוצג תמונה של כרטיס אשראי אשר בלחיצה עליה המשתמש יועבר למסך החיובים שלי בו יוכל לצפות בפירוט בכל העסקאות שביצע.

מסך החיובים שלי (MyPaymetsPage) – בעמוד זה יוכל המשתמש לצפות בפירוט בכל החיובים שלו ולקבל סטטיסטיקה עליהם. תחילה תוצג למשתמש רשימה של חודשים (dropdown list מחודש ינואר 2024 עד החודש הנוכחי) ועליו יהיה לבחור את החודש בו הוא רוצה לצפות. אחרי הבחירה תוצג למשתמש טבלה ובה הוא יראה את כל העסקות שהוא ביצע באותו החודש מכל כרטיסי האשראי שנמצאים ברשותו. בנוסף יוצגו עבור המשתמש 3 סוגים של גרפים:

- גרף עמודות: הגרף יתאר את ההוצאות בחודש הנוכחי לעומת 2 החודשים שקדמו לו (למשל אם המשתמש בחר בחודש אפריל, הגרף ישווה בין ההוצאות של חודש אפריל להוצאות של חודשים מרץ ופברואר)
- גרף פאי: הגרף יציג חלוקה של ההוצאות החודשיות עם פילוח לפי קטגוריה (לדוגמה אם המשתמש בחר בחודש ינואר הוא יוכל לראות כמה שילם בחודש זה עבור בגדים, עבור מזון, ועבור תחבורה ציבורית וכו....)
- גרף דונאט (דומה לגרף עוגה) : הגרף יציג עבור השנה הקלנדרית החולפת (2024) את פילוח ההוצאות לפי חודשים. כלומר המשתמש יוכל לקבל מידע פיננסי עבור כל 12 חודשי השנה החולפת ולראות זאת בצורה ויזואלית (למשל כך המשתמש יוכל לנתח באיזה חודש הוציא את הסכום המירבי בשנה החולפת ולהסיק מסקנות)

בנוסף לרשימת החודשים, ישנה רשימה של כרטיסי אשראי (תוספת שלנו) וניתנת למשתמש האופצייה לפלטר את המידע גם לפי כרטיסי האשראי שלו.

מסך טעינת טרנזקציות (TransactionsPage) – מסך פשוט בו המשתמש יכול לטעון את הטרנזקציות של כרטיס האשראי מקובץ עם סיומת csv. יש לשים לב כי בלי טעינת הטרנזקציות מסך הדשבורד ומסך החיובים שלי לא יציגו מידע כלל. לכן קודם עלינו לטעון את קובץ הטרנזקציות של המשתמש. הקובץ הוא קובץ csv שמכיל את העמודות:

Date, BusinessName, Category, Amount, cardNumber

מסך הוספת כרטיס אשראי נוסף (AddNewCardPage) – מסך זה מומש תחת הדרישה למסך שמאפשר פונקציונליות נוספת למערכת. במסך זה המשתמש יוכל להוסיף כרטיס אשראי נוסף לארנק שלו. לאחר הוספת הכרטיס המשתמש יוכל לצפות בעסקאות שבוצעו תחת כרטיס זה בעמוד החיובים שלו, שם הוספנו אופציה לצפייה בעסקאות לפי מספר כרטיס אשראי.

מסך טיפים פיננסים (FinancialTipsPage) – מסך זה מומש תחת הדרישה למסך שיציג תצוגת מידע מעניינת שנותנת ערך למשתמש. הרעיון מאחורי מסך זה הוא לנתח את ההוצאות החודשיות של המשתמש ולתת עבורו טיפים פיננסים. הטיפים הם טיפים דינמיים שמשתנים בהתאם להוצאות החודשיות של המשתמש. מידע זה יכול להועיל למשתמש ולגרום לו להתנהל בצורה נכונה יותר מבחינה כלכלית.

מסך מתכנן פיננסי (SavingPlannerPage) – מסך זה מומש תחת הדרישה לפונקציונליות נוספת של המערכת. מטרתו של מסך זה היא לאפשר למשתמש לקבוע יעד כלכלי מסוים (למשל חופשה או קניית רכב חדש) ולחסוך אליו. בכל כניסה לאתר המשתמש יוכל לרשום את הסכום שחסך וכך יתקדם ליעד.

כפתור התנתקות (Logout) – בלחיצה על הכפתור השתמש מתנתק מהמערכת.

<u>תיאור טכני של המערכת (למתכנת אחר):</u>

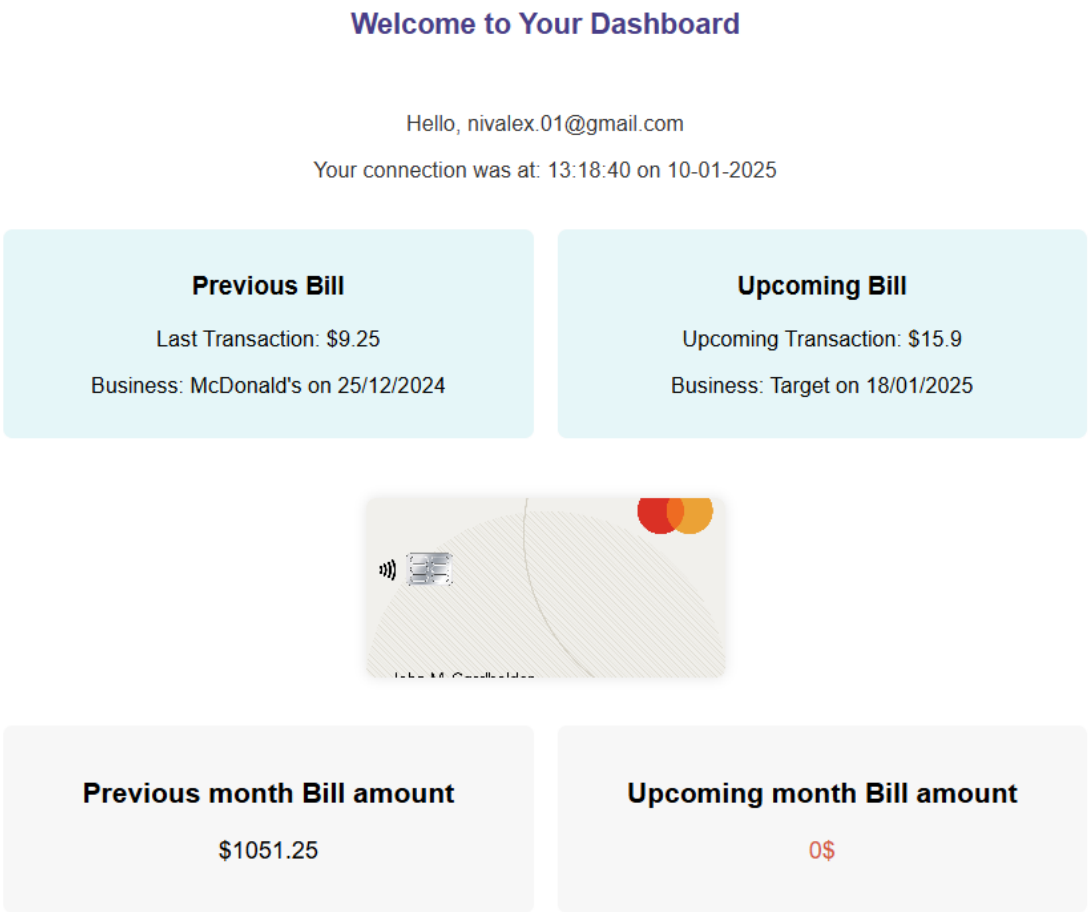
במערכת שלנו אנחנו שומרים 3 מערכים ל- local storage ואובייקט אחד.

- מערך ראשון נקרא listOfUsers והוא מכיל אובייקטים מסוג user. אובייקט מסוג user נוצר לאחר מילוי טופס ההרשמה והוא מכיל את התכונות הבאות: email, password, birthdate, creditCards. התכונות email, password, birthdate, הן מסוג מחרוזת, התכונה creditCards היא בעצם מערך של כרטיסי אשראי מאחר ולכל משתמש יכולים להיות מספר כרטיסי אשראי.

```
▼ [{email: "nivalex.01@gmail.com", password: "12345Aa!", birthDate: "1997-01-04",…},…]
  ▼ 0: {email: "nivalex.01@gmail.com", password: "12345Aa!", birthDate: "1997-01-04",…}
      birthDate: "1997-01-04"
    ▼ creditCards: [{number: "1234123412341234", expirationDate: "02/25"},…]
      ▶ 0: {number: "1234123412341234", expirationDate: "02/25"}
      ▶ 1: {number: "9999999999999999", expirationDate: "02/28"}
      ▶ 2: {number: "1234567812347890", expirationDate: "02/28"}
      email: "nivalex.01@gmail.com"
      password: "12345Aa!"
  ▼ 1: {email: "daniel123@gmail.com", password: "12345Aa!", birthDate: "2006-01-06",…}
      birthDate: "2006-01-06"
    ▼ creditCards: [{number: "1234123412345678", expirationDate: "02/25"}]
      ▼ 0: {number: "1234123412345678", expirationDate: "02/25"}
          expirationDate: "02/25"
          number: "1234123412345678"
      email: "daniel123@gmail.com"
      password: "12345Aa!"
```


בתמונה מוצגת דוגמה של listOfUsers – מערך של אובייקטים מסוג user. אובייקט user מכיל email, password, birthdate, creditCards כאשר creditCards הוא מערך של כרטיסים שכל כרטיס מכיל מספר כרטיס ותאריך תוקף.

- מערך שני שנשמר ב- local storage נקרא loggedInUser – תפקידו של מערך זה הוא לשמור את המיילים של המשתמשים שמחוברים כרגע למערכת. מערך זה מומש בעיקר כדי שנוכל להשתמש בו לצורך הצגת הודעה עם המייל בעמוד הדשבורד.



לדוגמה, בעמוד זה המייל nivalex.01@gmail.com שמוצג בהודעת ה-hello נלקח ממערך ה- loggedInUser. אם יתחבר משתמש אחר עם מייל אחר ההודעה תשתנה בהתאם למייל שיהיה במערך ה-loggedInUser .

- מערך שלישי שנשמר ב- local storage נקרא transactions וזהו בעצם המערך שמכיל אובייקטים של עסקאות. כאמור אנחנו קוראים את העסקאות מקובץ csv, כל עסקא היא אובייקט שנקרא transaction שמכיל את התכונות הבאות: Date, BusinessName, Category, Amount, cardNumber.

המערך transactions שומר בתוכו אובייקטים מסוג transaction. זהו המערך העיקרי בו אנו משתמשים באתר והוא עוזר לנו להציג מידע למשתמש בעמוד החיובים שלי.


```
▼ [{Date: "1/1/2024", BusinessName: "Starbucks", Category: "Food & Beverage", Amount: "5.75",…},…]
  ▼ 0: {Date: "1/1/2024", BusinessName: "Starbucks", Category: "Food & Beverage", Amount: "5.75",…}
      Amount: "5.75"
      BusinessName: "Starbucks"
      Category: "Food & Beverage"
      Date: "1/1/2024"
      cardNumber: "1234123412341230"
  ▼ 1: {Date: "2/1/2024", BusinessName: "Amazon", Category: "Retail & Shopping", Amount: "52.99",…}
      Amount: "52.99"
      BusinessName: "Amazon"
      Category: "Retail & Shopping"
      Date: "2/1/2024"
      cardNumber: "1234123412341230"
  ▶ 2: {Date: "3/2/2024", BusinessName: "Dunkin' Donuts", Category: "Food & Beverage", Amount: "4.5",…}
  ▶ 3: {Date: "5/2/2024", BusinessName: "Walmart", Category: "Retail & Shopping", Amount: "35.2",…}
  ▶ 4: {Date: "6/2/2024", BusinessName: "Netflix", Category: "Entertainment", Amount: "14.99",…}
  ▶ 5: {Date: "7/2/2024", BusinessName: "Chipotle", Category: "Food & Beverage", Amount: "11.5",…}
  ▶ 6: {Date: "10/2/2024", BusinessName: "Apple", Category: "Electronics", Amount: "899.99",…}
  ▶ 7: {Date: "11/2/2024", BusinessName: "Target", Category: "Retail & Shopping", Amount: "62.75",…}
  ▶ 8: {Date: "12/2/2024", BusinessName: "Spotify", Category: "Entertainment", Amount: "9.99",…}
  ▶ 9: {Date: "13/02/2024", BusinessName: "Best Buy", Category: "Electronics", Amount: "120.45",…}
  ▶ 10: {Date: "15/02/2024", BusinessName: "McDonald's", Category: "Food & Beverage", Amount: "8.25",…}
  ▶ 11: {Date: "16/02/2024", BusinessName: "H&M", Category: "Retail & Shopping", Amount: "44.8",…}
  ▶ 12: {Date: "18/02/2024", BusinessName: "Uber", Category: "Transportation", Amount: "22.9",…}
  ▶ 13: {Date: "19/02/2024", BusinessName: "Amazon", Category: "Retail & Shopping", Amount: "13.59",…}
  ▶ 14: {Date: "20/02/2024", BusinessName: "Starbucks", Category: "Food & Beverage", Amount: "6.95",…}
```


- לצורך המימוש של עמוד SavingPlannerPage אנחנו שומרים ב- local storage גם אובייקט שנקרא savingsData שמכיל 3 תכונות:מטרת החיסכון (goalTarget), הסכום הכולל שיש לחסוך (goalAmount) והסכום המצטבר שנחסך עד עכשיו (totalSaved). בעזרת אובייקט זה שנשמר ב- local storage נוכל להציג ליוזר מידע עדכני על התקדמות החיסכון שלו בעמוד SavingPlanner.

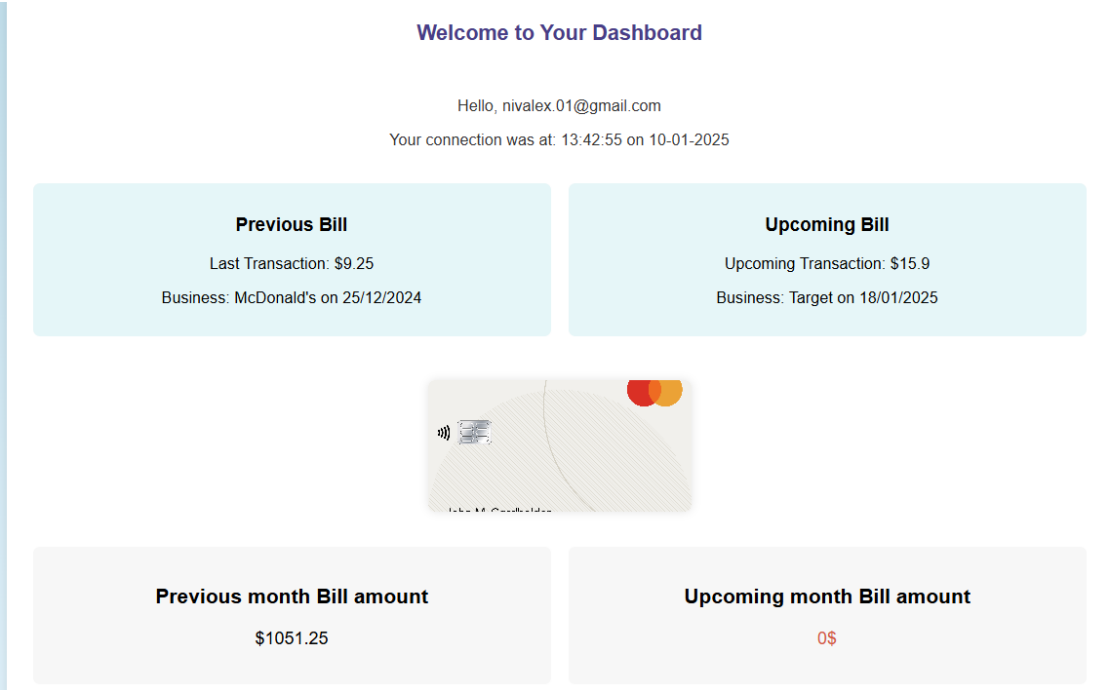
<u>הסבר על הלוגיקה בכל אחד מהעמודים (איך מימשנו ומה מיממשנו):</u>

<u>מסך דף הבית –</u> מכיל כפתור של Explore Features עם אירוע של click, בעת לחיצה על הכפתור אנו עוברים בלולאה על מערך העמודים באתר שלנו ויוצרים div של כרטיס ומוסיפים אותו באופן דינאמי ל-section. כך אנחנו מג'נרטים 6 כרטיסים באופן דינאמי לעמוד בעת לחיצה על הכפתור Explore Features.

<u>מסך הרשמה –</u> מכיל טופס הרשמה סטדרנטי. ה-js מכיל פונקצייה עיקרית שנקראת CheckSubmission ומטרה לבדוק את הווילדציה של כל השדות (כפי שהוגדר). אם הכל עבר בהצלחה, אנחנו יוצרים אובייקט חדש מסוג user, דוחפים אותו למערך ה-listOfUsers ולאחר מכן עושים stringify לערך ה-listOfUsers המעודכן ושולחים אותו בחזרה ל-local storage כסטרינג של json.

<u>מסך התחברות –</u> מסך המכיל טופס התחברות סטדנרטי בו המשתמש נדרש להזין מייל וסיסמא. ה- js של העמוד מכיל פונקצייה עיקרית שנקראת checkLogin בה אנחנו לוקחים את המייל והסיסמא מהשדות שמולאו ע"י היוזר ועוברים על מערך ה-listOfUsers (שאותו אנחנו לוקחים מה- local storage) ובודקים אם קיים יוזר במערך ה-listOfUsers שיש לו את אותו המייל ואותה סיסמא כמו שהתקבלו מהטופס. אם כן, אנחנו מבצעים התחברות ויוצרים מערך חדש של loggedInUser אותו אנחנו גם שולחים ל- local storage כי כעת יש משתמש מחובר למערכת. לאחר מכן אנחנו מפנים את המשתמש שהתחבר לעמוד הדשבורד בעזרת window.location.href.

<u>מסך דשבורד –</u> מסך המכיל קונטיינר בו היוזר יכול לקבל מידע. המידע נוסף באופן דינאמי לקונטיינר:



ה- js של העמוד מכיל מספר פונקציות:

- פונקציית usertimemsg – מטרת פונקצייה זאת היא לקחת את התאריך והזמן הנוכחי (נעשה באמצעות אובייקט Date) ולכתוב אותם עבור המשתמש. הוגדרה ב- html פסקה בעלת id שנקרא current-date-time ובסיום הפונקצייה מתבצע:

```
document.getElementById('current-date-time').textContent = dateTimeString;
```

כאשר dateTimeString הוא התאריך (שעה + תאריך) המודכנים נכון לרגע זה.

- פונקציית displayUserGreeting – מטרת פונקציה זאת היא לקחת את המייל של המשתמש שכעת מחובר (ממערך loggedInUser ששמור ב- local storage) ולהציג אותו בהודעת ה-hello. הוגדרה ב-html פסקה בעלת id שנקראת user-email ובסיום הפונקציה מתבצע:

```
document.getElementById('user-email').textContent = loggedInUser_array[0].email;
```

כאשר loggedInUser במקום ה-0 הוא אובייקט מסוג יוזר שמחובר כעת והתכונה email תביא לנו את המייל שלו.

הפונקציה getPreviousMonthBill, הפונקציה getUpcomingMonthBill, הפונקציה ()getPreviousTransactionBill והפונקציה ()getUpcomingTransactionBill : כולן בעלות לוגיקה דומה מאוד. עוברים על מערך ה-transactions שנמצא ב-local storage ומבצעים פילטור עליו לפי הצורך. למשל הפונקציה ()getPreviousTransactionBill אמורה להחזיר את הפרטים של הטרנזקציה האחרונה שהתבצעה ששמורה במערך ה-transactions.

```javascript
function getPreviousTransactionBill() {
    if (!user_transactions || user_transactions.length === 0) {
        document.getElementById('previous-bill').textContent = "No transactions for the previous month";
        return false;
    }

    let currentDate = new Date();
    let lastMonth = new Date(currentDate.setMonth(currentDate.getMonth() - 1));

    // filter all the transactions of the last month
    let previousMonthTransactions = user_transactions.filter(transaction => {
        let transactionDate = new Date(transaction.Date.split('/').reverse().join('/')); // Reversing for correct Date parsing
        return transactionDate.getMonth() === lastMonth.getMonth() && transactionDate.getFullYear() === lastMonth.getFullYear();
    });

    // check if there are any transactions in the previous month
    if (previousMonthTransactions.length === 0) {
        document.getElementById('previous-bill').textContent = "No transactions for the previous month";
        return;
    }

    // Sort transactions in descending order to get the latest one
    previousMonthTransactions.sort((a, b) => new Date(b.Date.split('/').reverse().join('/')) - new Date(a.Date.split('/').reverse().join('/')));

    // get the most recent transaction (its in the first place in previousMonthTransactions array )
    let lastTransaction = previousMonthTransactions[0];

    // Function to format date to day/month/year
    1 reference
    function formatDate(dateStr) {
        let date = new Date(dateStr.split('/').reverse().join('/'));
        let day = date.getDate().toString().padStart(2, '0');
        let month = (date.getMonth() + 1).toString().padStart(2, '0');
        let year = date.getFullYear();
        return `${day}/${month}/${year}`;
    }

    document.getElementById('previous-bill').textContent = `Last Transaction: $${lastTransaction.Amount}`;
    document.getElementById('previous-bill-details').textContent = `Business: ${lastTransaction.BusinessName} on ${formatDate(lastTransaction.Date)}`;
}
```

שאר הפונקציות נראות באופן דומה, למשל ב- getPreviousMonthBill נפלטר לפי החודש הקודם ואז נבצע סכימה על כל הטרנזקציות שהיו באותו החודש.

<u>מסך החיובים שלי –</u> מכיל קונטיינר עם 2 dropdown list (האחת לבחירת החודש והשנייה לבחירת כרטיס האשראי) – הרשימות נוצרות באופן דינאמי ב- js. כך שאם האתר ירוץ בחודש ינואר 2025, אז רשימת החודשים תהיה מינואר 2024 עד ינואר 2025 ואם האתר ירוץ בחודש מרץ 2025 רשימת החודשים תהיה מינואר 2024 עד מרץ 2025. דבר זה נעשה ב- js בעזרת שימוש באובייקט Date. ברשימת כרטיסי האשראי הדבר נעשה ע"י תפיסת אובייקט היוזר מרשימת ה-loggedInUser ומעבר על רשימת כרטיסי האשראי שלו (כזכור ליוזר יש תכונה שהיא מערך של כרטיסי אשראי). בנוסף מוגדר בקונטיינר הטבלה הבאה:

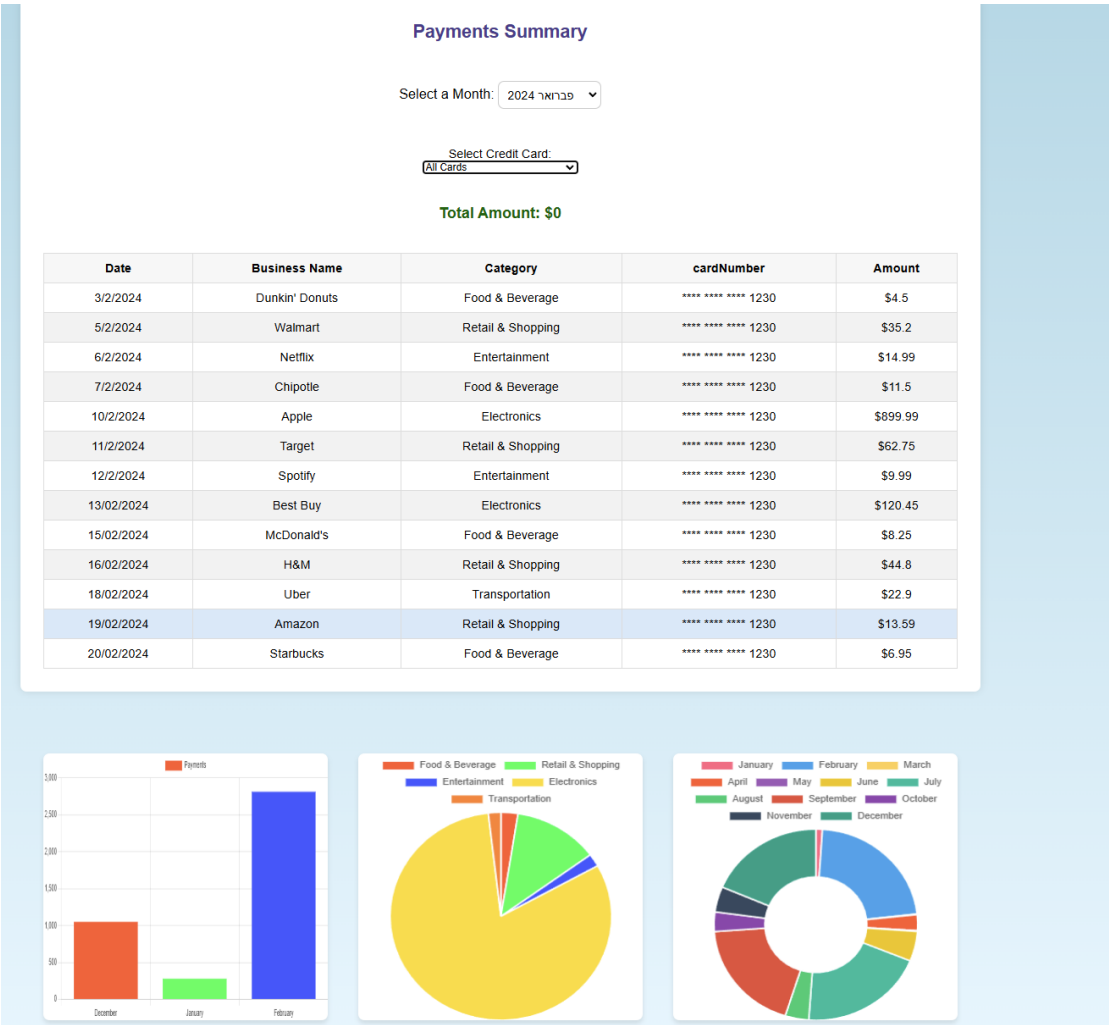
```html
        <!-- Table to display transactions -->
        <table id="transactions-table" style="display: none;">
            <thead>
                <tr>
                    <th>Date</th>
                    <th>Business Name</th>
                    <th>Category</th>
                    <th>cardNumber</th>
                    <th>Amount</th>
                </tr>
            </thead>
            <tbody id="transactions-body">
                <!-- Transaction rows will go here -->
            </tbody>
        </table>

    </div>
```

זו היא טבלה שתתמלא באופן דינאמי ב-js לאחר שהמשתמש בוחר את החודש הרצוי. בנוסף מוגדר עוד קונטיינר שיכיל 3 גרפים שיהיו מיוצרים ע"י הספרייה chart.js

```
    <!-- Charts Section -->
    <div class="charts-container">
        <canvas id="paymentBarChart"></canvas>
        <canvas id="categoryPieChart"></canvas>
        <canvas id="paymentDoughnutChart"></canvas>
    </div>
</div>
```

לסיכום, כך נראה העמוד:



קוד ה-js של עמוד זה הוא יחסית ארוך (האורך נובע בעיקר מעבודה עם ספריית chart js וביצוע הפילטורים לפי חודש בלבד ולפי חודש + מספר כרטיס) והוא החלק העיקרי של ה-js במערכת שלנו.

קובץ ה- js מכיל את הפונקציות הבאות:

1. generateMonthOptions() – מג'נרט רשימה דינמית של חודשים החל מינואר 2024 ועד החודש הנוכחי.
2. generateCreditCardOptions() – מג'נרט רשימה דינמית של כרטיסי אשראי שמבוססת על הכרטיסים של ה-loggedInUser הנוכחי.
3. displayTransactionswithCardNumber() – מציגה (כלומר עושה אפנד ל -transactionsBody) את הטרנזקציות מפולטרות לפי כרטיס אשראי + חודש.
4. displayTransactionsWithoutCardNumber –מציגה (כלומר עושה אפנד ל -transactionsBody) את הטרנזקציות מפולטרות רק לפי החודש הנבחר, פונקציה זאת נכתבת במידה ורוצים לפלטר רק לפי חודש ולא לפי חודש + מספר כרטיס אשראי.
5. יש בעמוד זה 3 פונקציות שנקראות :
   - createPaymentBarChart
   - createCategoryPieChart
   - createDoughnutChart

   כולן נראות פחות או יותר אותו דבר והן יוצרות 3 תרשימים בעזרת הספרייה החיצונית chart.js.
6. filterByCardNumberAndMonth – מחזירה מערך של טרנזקציות מפולטרות לפי מספר כרטיס וחודש.
7. filterByMonth – מחזירה מערך טרנזקציות שמפולטרות רק לפי חודש.
8. filterTransactions – הפונקציה עיקרית לפילטור הצגת הנתונים המפולטרים. נשים לב שיש שני סוגים של פילטורים, אחד לפי חודש וגם מספר כרטיס והשני רק לפי החודש. בהתאם לפילטור שנרצה נקרא לפונקציות העזר filterByCardNumberAndMonth או filterByMonth
9. מוגדרים שני event listeners לשתי הרשימות (חודשים ומספרי כרטיסים) בכל event כזה נשמור את החודש ונשמור את מספר הכרטיס (הערך ה-selected בדרופ דאון) ונשלח נתונים אלה ל-filterTransactions.

מסך טעינת טרנזקציות – מסך יחסית פשוט עם פקד להעלאת קובץ csv. ה-js של עמוד זה יכיל פונקציה עיקרית שעושה המרה של קובץ csv ל-json (ניתן לממש בהרבה דרכים, בקוד שלנו אנחנו בחרנו בדרך הפשוט ביותר לדעתנו)

```
// Function to convert CSV data to JSON
1 reference
function csvToJson(csv) {
    const lines = csv.trim().split("\n"); // split the csv into lines
    const headers = lines[0].split(","); // get tge headers
    const result = []; // array to store the json objects

    for (let i = 1; i < lines.length; i++) {
        const line = lines[i].split(","); // Split the line into values
        const obj = {};

        for (let j = 0; j < headers.length; j++) {
            obj[headers[j].trim()] = line[j] ? line[j].trim() : "";
        }

        result.push(obj);
    }

    return result; // array of objects
}
```

מערך ה- result יהיה מערך של אובייקטים שעליו נעשה stringify ונדחוף אותו כ-json ל transactions שנמצא ב- local storage.

```
document.getElementById('file-input').addEventListener('change', function (event)
{
    const file = event.target.files[0]; // Get the selected file

    if (file) {
        const reader = new FileReader(); // Create a FileReader to read the file

        5 references
        reader.onload = function () {
            const csvData = reader.result; // get the file content as text
            const jsonData = csvToJson(csvData); // convert the CSV to json
            localStorage.setItem('transactions', JSON.stringify(jsonData)); // save the json to localStorage
            console.log('JSON data saved to local storage.');
        };

        5 references
        reader.onerror = function () {
            console.error('Error reading the file.');
        };

        reader.readAsText(file); // Read the file as text
    }
    else {
        console.error('No file selected.');
    }
});
```

מסך הוספת כרטיס חדש – הלוגיקה בעמוד זה היא להוסיף כרטיס אשראי חדש למערך כרטיסי האשראי של השתמש שכעת מחובר. כזכור, אנחנו שומרים ב- local storage מערך של משתמשים מחוברים, אז ניתן בקלות לשלוף את המייל של השתמש המחובר, לחפש אותו ב-listofusers ולהחזיר אובייקט מסוג user. כזכור לאובייקט מסוג user יש תכונה שהיא מערך של כרטיסי אשראי (כל כרטיס מכיל מספר ותאריך תוקף) וכך ניתן להוסיף כרטיס חדש למערך קיים של כרטיסים.

```
function addNewCard() {
    let creditCardNumber = document.getElementById("CreditCardNumber").value;
    let expirationDate = document.getElementById("ExpirationDate").value;
```

```
    let emailArray = JSON.parse(localStorage.getItem("loggedInUser"));
    let email = emailArray[0].email; // Get the first item (the email) in the array
    console.log(email);
    let listOfUsers = getUsersFromLocalStorage(); // Get all users from localStorage
    let currentUser = null;
    for (let i = 0; i < listOfUsers.length; i++) {
        if (listOfUsers[i].email === email) {
            currentUser = listOfUsers[i];
            break;
        }
    }

    if (currentUser != null) {
        // Add new card to user's card list
        currentUser.creditCards.push({
            number: creditCardNumber,
            expirationDate: expirationDate
        });
        localStorage.setItem("listOfUsers", JSON.stringify(listOfUsers));
        alert("Card added successfully!");
        window.location.href = "DashboardPage.html";
    }
    else {
        alert("User not found. Please log in again.");
        window.location.href = "LoginPage.html";
    }
}
```

מסך טיפים פיננסים: הגדרנו מערך סגור של טיפים שנקרא advice שמכיל טיפים שיג'ונרטו לעמוד באופן דינאמי ובהתאם לנתונים של המשתמש. עבור משתמש מחובר תפסנו את רשימת הטרנזקציות שלו מה- local storage ופילטרנו אותן לפי החודש הנוכחי (כך שאם אנחנו בפברואר 2025 מערך הטרנזקציות יכיל רק את הטרנזקציות של החודש הנוכחי) לאחר מכן יצרנו מערך של קטגוריות ועברנו על הטרנזקציות וסכמנו כמה המשתמש הוציא על כל קטגוריה בחודש הנוכחי. כעת אנחנו עם מערך שנקרא categoryExpenses שמכיל קטגוריות וסכום שהיוזר הוציא עליהן. ברגע שהגענו למצב כזה עברנו על כל הקטגוריות וראינו כמה אחוזים היא תופסת מכלל ההוצאות באותו חודש, ובהתאם לכך עשינו push לטיפ הרלוונטי.

מסך מתכנן פיננסי: עוד מסך שמתבסס על עבודה עם local storage ועם chart js. שמרנו ב- local storage אובייקט שנקרא savingsData. לאובייקט הזה יש 3 תכונות: goalTarget, totalSaved,goalAmount

התכונות מייצגות את שם המטרה לשמה היוזר חוסך (goalAmount), את כמות הכסף שחסך עד עכשיו (totalSaved) ואת כמות הכסף שהוא צריך לחסוך עד שיגיע למטרה (goalTarget). בכל פעם שהעמוד עולה הוא לוקח את הדאטה מה- savingsData ששמור כאמור ב-local storage . מטרתו של העמוד היא להציג ליוזר גרף פאי שיראה לו כמה הוא קרוב להשגת היעד שלשמו הוא חוסך (למשל דירה חדשה או חופשה שנתית). היוזר יכול בכל פעם להכניס כמות כסף חדשה שחסך והמידע יתעדכן ב- local storage וגרף העוגה ישתנה בהתאם.

פיצ'רים תחת הקטגוריה של הרחבת דרישות בסיסיות:

1. הוספת כרטיס אשראי חדש למערך כרטיסי האשראי של המשתמש (דרישה פונקציונלית) והצגת מידע על עסקאות של כרטיס זה בעמוד התשלומים שלי (תצוגת מידע מעניינת)

```
// function to add a new card
function addNewCard() {
    let creditCardNumber = document.getElementById("CreditCardNumber").value;
    let expirationDate = document.getElementById("ExpirationDate").value;

    // get the current user email from logged-in array in local storage
    let emailArray = JSON.parse(localStorage.getItem("loggedInUser"));
    let email = emailArray[0].email; // get the first item (the email on the user) in logged-in array
    console.log(email);
    let listOfUsers = getUsersFromLocalStorage();
    let currentUser = null;
    for (let i = 0; i < listOfUsers.length; i++) {
        if (listOfUsers[i].email === email) {
            currentUser = listOfUsers[i];
            break;
        }
    }

    if (currentUser != null) {
        // Add new card to user's card list
        currentUser.creditCards.push({
            number: creditCardNumber,
            expirationDate: expirationDate
        });
        localStorage.setItem("listOfUsers", JSON.stringify(listOfUsers));
        alert("Card added successfully!");
        window.location.href = "DashboardPage.html";
    }
    else {
        alert("User not found. Please log in again.");
        window.location.href = "LoginPage.html";
    }
}
```

לקחנו את המייל של המשתמש שכעת מחובר למערכת (כזכור מייל זה נמצא ב- local storage במערך שנקרא loggedInUser) בנוסף לקחנו את רשימת כל המשתמשים שרשומים לאתר (listOfUsers) וחיפשנו את המשתמש לפי מייל ברשימה זו. אם מצאנו, החזרנו אובייקט מסוג user, לקחנו את תכונת מערך כרטיסי האשראי שלו (creditCards) ודחפנו לשם אובייקט חדש של כרטיס אשראי. לאחר שעדכנו את מערך כרטיסי האשראי של ה- user שלחנו את מערך ה-listOfUsers כ-json מעודכן ל-local storage .

2. טיפים פיננסים חכמים שניתנים למשתמש בהתאם להוצאות החודשיות שלו (תצוגת מידע מעניינת)

```
function getMonthlyExpensesWithAdvice()
{
    const expensesData = localStorage.getItem("transactions");
    if (expensesData == null)
    {
        return { transactions: [], totalExpenses: 0, advice: [] };
    }

    const transactions_array = JSON.parse(expensesData);
    const currentDate = new Date();
    const currentMonth = currentDate.getMonth();
    const currentYear = currentDate.getFullYear();

    const currentMonthExpenses = [];
    for (let i = 0; i < transactions_array.length; i++) {
        const transaction = transactions_array[i];
        const dateParts = transaction.Date.split('/'); // split the date into parts
        const day = parseInt(dateParts[0], 10);
        const month = parseInt(dateParts[1], 10) - 1;
        const year = parseInt(dateParts[2], 10);
        const transactionDate = new Date(year, month, day); // create a valid Date

        if (transactionDate.getMonth() === currentMonth && transactionDate.getFullYear() === currentYear)
        {
            currentMonthExpenses.push(transaction); // Add matching transaction to the array
        }
    }
```

```
    // sum the expenses for the current month
    let totalExpenses = 0;
    const categoryExpenses = {
        "Retail & Shopping": 0,
        "Food & Beverage": 0,
        "Entertainment": 0,
        "Transportation": 0,
        "Bills & Utilities": 0,
        "Travel": 0,
        "Electronics" : 0,
        "Others": 0
    };

    // sum expenses for each category
    currentMonthExpenses.forEach(transaction => {
        totalExpenses += parseFloat(transaction.Amount);

        if (categoryExpenses.hasOwnProperty(transaction.Category)) {
            categoryExpenses[transaction.Category] += parseFloat(transaction.Amount);
        } else {
            categoryExpenses["Others"] += parseFloat(transaction.Amount);
        }
    });

    // generate advice for each category based on its percentage of total expenses
    const advice = [];

    const categories = ["Retail & Shopping", "Food & Beverage", "Entertainment", "Bills & Utilities", "Travel", "Electronics", "Transportation", "Others
```

```
    for (let i = 0; i < categories.length; i++)
    {
        let category = categories[i];
        console.log(category)
        let categoryPercentage = (categoryExpenses[category] / totalExpenses) * 100;
        if (category === "Retail & Shopping" && categoryPercentage > 50) {
            advice.push("You are spending more than half of your monthly expenses on Retail & Shopping. Consider reducing your shopping to save more!");
        } else if (category === "Food & Beverage" && categoryPercentage > 30) {
            advice.push("You are spending a significant portion on Food & Beverage. Try cooking at home more to save!");
        } else if (category === "Entertainment" && categoryPercentage > 20) {
            advice.push("Entertainment expenses are high. You might want to explore cheaper options for entertainment.");
        } else if (category === "Bills & Utilities" && categoryPercentage > 25) {
            advice.push("Your Bills & Utilities are taking up a large part of your budget. Look for ways to cut down on these expenses.");
        } else if (category === "Travel" && categoryPercentage > 10) {
            advice.push("Travel expenses are adding up. You might want to plan your trips more carefully or opt for more affordable options.");
        } else if (category === "Electronics" && categoryPercentage > 15) {
            advice.push("Electronics purchases are significant. Consider delaying unnecessary purchases to save money.");
        } else if (category === "Transportation" && categoryPercentage > 15) {
            advice.push("Transportation costs are high. Look for ways to reduce travel expenses, such as using public transport or carpooling.");
        } else if (category === "Others" && categoryPercentage > 15) {
            advice.push("You may want to track 'Other' expenses more closely to ensure you're not overspending in this undefined category.");
        }
    }

    return { transactions: currentMonthExpenses, totalExpenses, advice };
}
```

פונקציה עיקרית שבעזרתה אנחנו ממשים את הלוגיקה של הטיפים החכמים. יש לנו מערך advice ששומר בתוכו רשימה של טיפים בהתאם לתרחישים שונים שחשבנו עליהם. בנוסף יש לנו מערך categoriesששומר בתוכו מחרוזות שמייצגות קטגוריה. בפונקציה אנחנו תופסים את רשימת הטרנזקציות מה- local storage ותופסים את החודש הנוכחי, יוצרים מערך עזר של currentMonthExpenses בו מאחסנים רק את הטרנזקציות שהתרחשו בחודש הנוכחי, לאחר מכן יוצרים אובייקט מסוג categoryExpenses שפועל בדומה למילון (לכל

key שבמקרה שלנו זה קטגוריה, יש value שבמקרה שלנו זה סכום ההוצאה), אחנו ממלאים את המילון בעזרת מעבר על currentMonthExpenses ואז עוברים על categories ומחשבים כמה אחוז כל קטגוריה תפסה מההוצאה הכוללת, בהתאם למספר אנחנו עושים push למערך advice לטיפ המתאים.

3. מתכנן פיננסי שעוזר למשתמש לקבוע יעד לחיסכון ולהפקיד סכום כסף אותו חסך בכל פעם (פונקציונליות + תצוגת מידע מעניינת)

```
let savingsData = {
    goalAmount: 0,
    totalSaved: 0,
    goalTarget: ''
};
let savingsChart = null;

// load data from localStorage
1 reference
function loadSavingsData() {
    const savedData = localStorage.getItem('savingsData');
    if (savedData) {
        savingsData = JSON.parse(savedData);
    }
}

// save data to localStorage
2 references
function saveSavingsData() {
    localStorage.setItem('savingsData', JSON.stringify(savingsData));
}
```

```
// set the savings goal
0 references
function setSavingsGoal() {
    const targetInput = document.getElementById('goalTarget');
    const amountInput = document.getElementById('goalAmount');

    savingsData.goalTarget = targetInput.value.trim();
    savingsData.goalAmount = parseFloat(amountInput.value);

    if (savingsData.goalTarget && savingsData.goalAmount > 0) {
        saveSavingsData();
        alert(`Savings goal of $${savingsData.goalAmount} for ${savingsData.goalTarget} set!`);
        updateChart();
    } else {
        alert('Please enter a valid target and goal amount.');
    }
}
```

```
// log the savings
0 references
function logSavings() {
    const input = document.getElementById('savingsAmount');
    const amount = parseFloat(input.value);

    if (amount > 0) {
        savingsData.totalSaved += amount;
        saveSavingsData();
        alert(`You saved $${amount}. Total saved: $${savingsData.totalSaved}`);
        updateChart();
    } else {
        alert('Please enter a valid amount.');
    }
}
```

```
function updateChart() {
    const ctx = document.getElementById('progressChart').getContext('2d');
    const percentage = (savingsData.totalSaved / savingsData.goalAmount) * 100;
    if (savingsChart) {
        savingsChart.destroy();
    }

    // create a new chart
    savingsChart = new Chart(ctx, {
        type: 'doughnut',
        data: {
            labels: ['Saved', 'Remaining'],
            datasets: [{
                label: 'Savings Progress',
                data: [savingsData.totalSaved, Math.max(0, savingsData.goalAmount - savingsData.totalSaved)],
                backgroundColor: ['#4caf50', '#f44336'],
            }]
        },
        options: {
            responsive: true,
            plugins: {
                tooltip: {
                    callbacks: {
                        label: (tooltipItem) => {
                            const value = tooltipItem.raw;
                            return `$${value}`;
                        }
                    }
                }
            }
        }
    });
```

```
    const summary = document.getElementById('progressSummary');
    summary.textContent = `You have saved $${savingsData.totalSaved} out of your goal of $${savingsData.goalAmount} (${percentage.toFixed(2)}% complete)
}

4 references
window.onload = function ()
{
    loadSavingsData();
    let targetInput = document.getElementById('goalTarget');
    let amountInput = document.getElementById('goalAmount');

    if (savingsData.goalTarget) {
        targetInput.value = savingsData.goalTarget;
    }

    if (savingsData.goalAmount > 0) {
        amountInput.value = savingsData.goalAmount;
    }

    updateChart();
}
```

חילקנו את העבודה בעמוד זה למספר פונקציות, ראשית יצרנו אובייקט מסוג savingsData שמייצג חיסכון והוא זה שישמר בכל פעם ל- local storage, יצרנו 2 פונקציות אחד לשמירה ואחת לקבלת הנתונים של אובייקט זה מה- local storage. setSavingsGoal() שבאמצעותה אחנו מגדירים את המטרה לשמה ה-user חוסך, פונקציית logSavings שבאמצעותה ה- user מוסיף סכום שהוא חסך, הסכום הכולל ב- local storage יתעדכן והתצוגה בגרף תהיה בהתאם, updateChart() פונקצייה ליצירת chart בעזרת ספריית chart js, הפונקציה דומה לפונקציות שמימשנו בעמוד התשלומים שלי,

מקורות:

מקורות שעבדנו איתם:

ספריית chart js : https://www.chartjs.org/docs/latest/charts/bar.html

ספריית Date ב-js : https://javascript.info/date

עבודה עם פונקציית filter במערכים:
https://www.javascripttutorial.net/javascript-array-filter/

נעזרנו גם ב- stackoverflow בעיקר בנושא של המרת קובץ csv והפיכתו ל- json: https://stackoverflow.com/questions/27979002/convert-csv-data-into-json-format-using-javascript

הערות חשובות:

תיאור קובץ ה-csv:

E	D	C	B	A	
cardNumber	Amount	Category	BusinessName	Date	1
1234123412341234	5.75	Food & Beverage	Starbucks	1/1/2024	2
1234123412341234	52.99	Retail & Shopping	Amazon	2/1/2024	3
1234123412341234	4.5	Food & Beverage	Dunkin' Donuts	3/2/2024	4
1234123412341234	35.2	Retail & Shopping	Walmart	5/2/2024	5
1234123412341234	14.99	Entertainment	Netflix	6/2/2024	6
1234123412341234	11.5	Food & Beverage	Chipotle	7/2/2024	7
1234567812345678	899.99	Electronics	Apple	10/2/2024	8
1234567812345678	62.75	Retail & Shopping	Target	11/2/2024	9
1234567812345678	9.99	Entertainment	Spotify	12/2/2024	10
1234567812345678	120.45	Electronics	Best Buy	13/02/2024	11
1234567812345678	8.25	Food & Beverage	McDonald's	15/02/2024	12
8765432187654321	44.8	Retail & Shopping	H&M	16/02/2024	13
1234567812345678	22.9	Transportation	Uber	18/02/2024	14
8765432187654321	13.59	Retail & Shopping	Amazon	19/02/2024	15
8765432187654321	6.95	Food & Beverage	Starbucks	20/02/2024	16
8765432187654321	7.3	Food & Beverage	Subway	25/3/2024	17
8765432187654321	65	Retail & Shopping	Amazon	5/4/2024	18
9999999999999999	14.99	Entertainment	Netflix	10/4/2024	19
9999999999999999	20.55	Retail & Shopping	Walmart	12/4/2024	20
1234123412341234	45.3	Retail & Shopping	Target	17/4/2024	21
1234123412341234	5.5	Food & Beverage	Starbucks	22/4/2024	22
1234123412341234	8	Food & Beverage	McDonald's	25/5/2024	23
9999999999999990	230.25	Electronics	Best Buy	7/6/2024	24
9999999999999990	12.2	Food & Beverage	Chipotle	15/6/2024	25
9999999999999990	18.7	Retail & Shopping	Amazon	22/6/2024	26
9999999999999990	25	Transportation	Uber	29/6/2024	27
9999999999999990	999.99	Electronics	Apple	2/7/2024	28
9999999999999990	9.99	Entertainment	Spotify	9/7/2024	29
1234123412341234	39.99	Retail & Shopping	H&M	14/7/2024	30
1234123412341234	5	Food & Beverage	Dunkin' Donuts	18/7/2024	31
1234123412341234	61.3	Retail & Shopping	Target	22/7/2024	32
9999999999999990	13.4	Food & Beverage	Chipotle	30/7/2024	33
9999999999999990	48.6	Retail & Shopping	Walmart	3/8/2024	34
9999999999999990	6.5	Food & Beverage	Subway	12/8/2024	35
9999999999999990	7.75	Food & Beverage	Starbucks	16/8/2024	36
9999999999999990	150	Electronics	Best Buy	25/8/2024	37
9999999999999990	950	Electronics	Apple	1/9/2024	38
9999999999999990	22.45	Retail & Shopping	Amazon	5/9/2024	39
9999999999999990	14.99	Entertainment	Netflix	10/9/2024	40
9999999999999990	11.25	Food & Beverage	Chipotle	15/9/2024	41
9999999999999990	57.2	Retail & Shopping	Target	22/9/2024	42
1234123412341234	20.5	Transportation	Uber	30/9/2024	43
1234123412341234	30	Retail & Shopping	Walmart	2/10/2024	44
1234123412341234	49.5	Retail & Shopping	H&M	9/10/2024	45
1234123412341234	6	Food & Beverage	Starbucks	15/10/2024	46
1234123412341234	11.7	Food & Beverage	Chipotle	22/10/2024	47
9999999999999990	89.5	Retail & Shopping	Amazon	31/10/2024	48
1234567812347890	40.3	Retail & Shopping	Target	5/11/2024	49
1234567812347890	7.9	Food & Beverage	McDonald's	10/11/2024	50

E	D	C	B	A
9999999999999990	14.99	Entertainment	Netflix	10/9/2024
9999999999999990	11.25	Food & Beverage	Chipotle	15/9/2024
9999999999999990	57.2	Retail & Shopping	Target	22/9/2024
1234123412341234	20.5	Transportation	Uber	30/9/2024
1234123412341234	30	Retail & Shopping	Walmart	2/10/2024
1234123412341234	49.5	Retail & Shopping	H&M	9/10/2024
1234123412341234	6	Food & Beverage	Starbucks	15/10/2024
1234123412341234	11.7	Food & Beverage	Chipotle	22/10/2024
9999999999999990	89.5	Retail & Shopping	Amazon	31/10/2024
1234567812347890	40.3	Retail & Shopping	Target	5/11/2024
1234567812347890	7.9	Food & Beverage	McDonald's	10/11/2024
1234123412341234	175.65	Electronics	Best Buy	15/11/2024
1234123412341234	14.99	Entertainment	Netflix	20/11/2024
9999999999999990	5	Food & Beverage	Starbucks	25/11/2024
1234567812347890	62.2	Retail & Shopping	Amazon	3/12/2024
1234567812347890	24	Transportation	Uber	10/12/2024
9999999999999990	899	Electronics	Apple	12/12/2024
1234567812347890	56.8	Retail & Shopping	Target	17/12/2024
1234567812347890	9.25	Food & Beverage	McDonald's	25/12/2024
1234567812347890	15.9	Retail & Shopping	Target	18/1/2025
1234567812347890	20.7	Food & Beverage	Starbucks	19/1/2025
1234567812347890	40.9	Food & Beverage	McDonald's	22/1/2025
1234567812347890	105.9	Retail & Shopping	Amazon	25/1/2025
1234567812347890	20.6	Transportation	Uber	27/1/2025
1234567812347890	980.7	studies	ruppin	1/2/2025
1234567812347890	580.99	studies	ruppin dorms	2/2/2025
1234567812347890	20.6	Food & Beverage	yellow	5/2/2025
9999999999999990	100.6	Food & Beverage	sonol	4/2/2025
1234567812347890	70.3	Food & Beverage	karnaf	6/2/2025
1234123412341234	60.7	Retail & Shopping	Amazon	10/2/2025
1234123412341234	40.9	Entertainment	Netflix	19/2/2025
1234123412341234	10.9	Food & Beverage	McDonald's	1/3/2025
9999999999999990	10.9	studies	ruppin	2/3/2025
1234123412341234	107.9	Retail & Shopping	Walmart	10/3/2025

- הערה: כדי לעבוד בצורה מיטבית עם המערכת יש עדיפות להשתמש במספרי כרטיסי האשראי שנמצאים בקובץ ה-csv (ניתן לשים מספר אחר אבל אז הפילטור לפי כ.אשראי בעמוד התשלומים שלי מאחר והכרטיס החדש נשמר רק ב- local storage ולא מוכנס ל-csv) לא יעבוד בצורה מיטבית. לכן יש לעדיף להשתמש במספרי כרטיסי אשראי אלו:
- 1234123412341234
- 1234567812345678
- 8765432187654321
- 9999999999999999
- 9999999999999990

- בעת ההרשמה על המשתמש להיות מעל גיל 16, על כרטיס האשראי להיות בעל 16 ספרות והסיסמא של המשתמש צריכה להכיל 8 תווים כאשר תו אחד הוא אות גדולה, תו אחד אות קטנה ותו אחד סימן מיוחד.

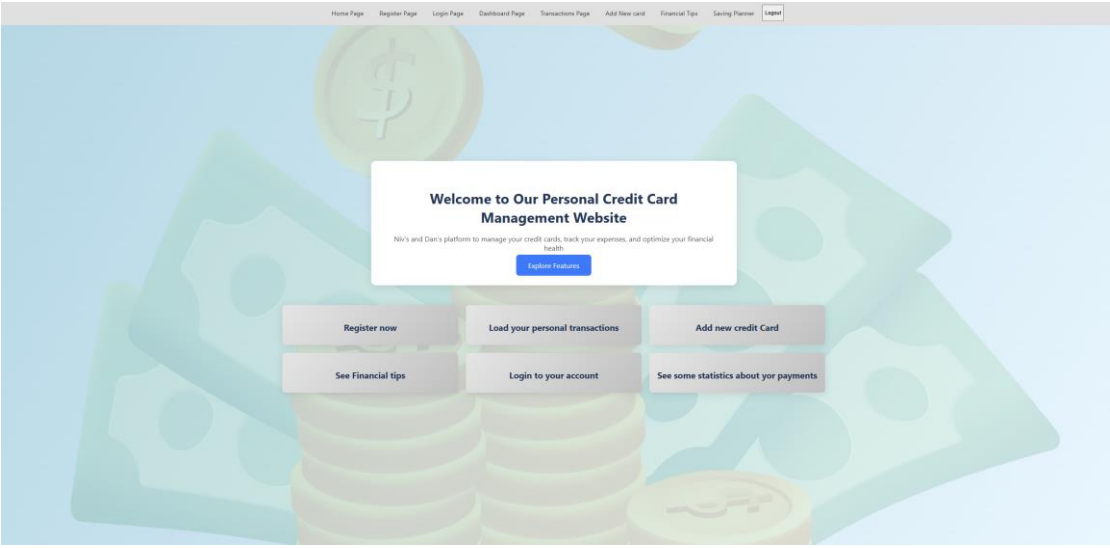
דוגמה להרשמה חוקית:

Email: nivalex.01@gmail.com
Password: 12345Aa!
Bdate: 22/04/2001
Credit card number: 1234123412341234
Exp date: 02/26

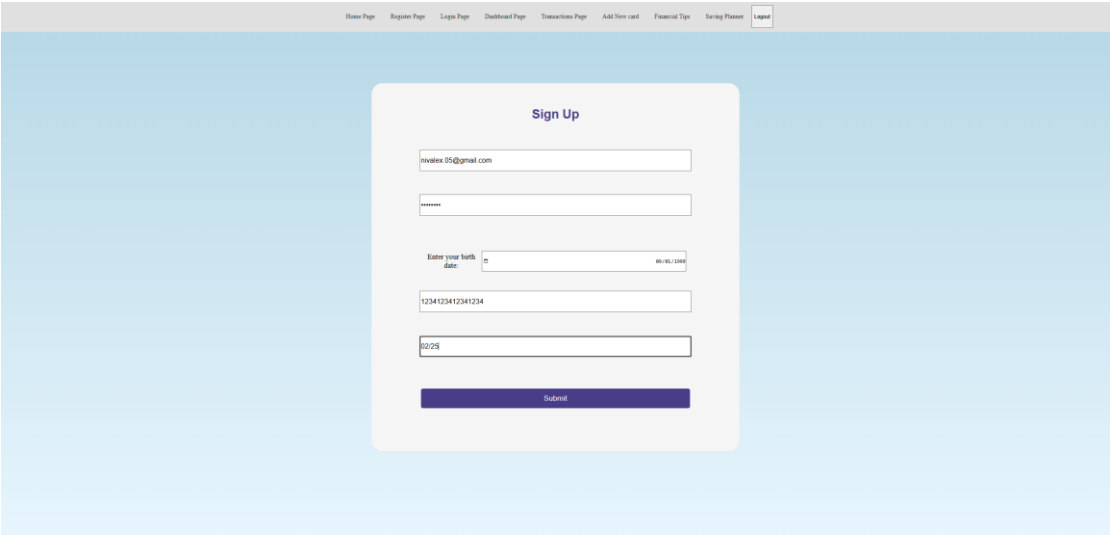
זהו, בעמודים הבאים מצורפים תמונות של מסכים מהאתר שלנו.

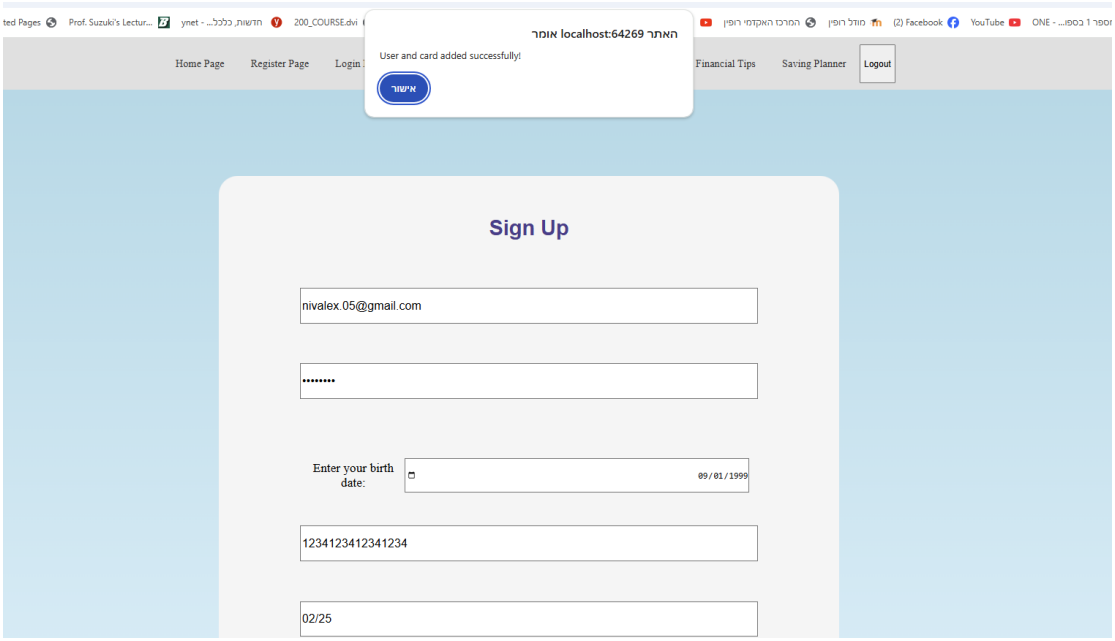
צילומי מסך עבור פלטים במצב (desktop):

עמוד הבית:

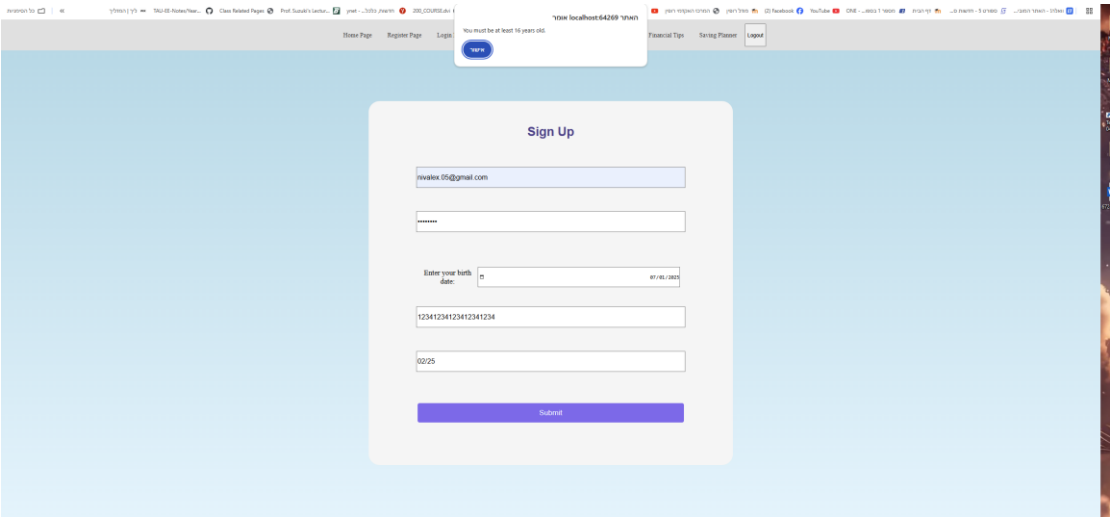


הרשמה מוצלחת של משתמש:

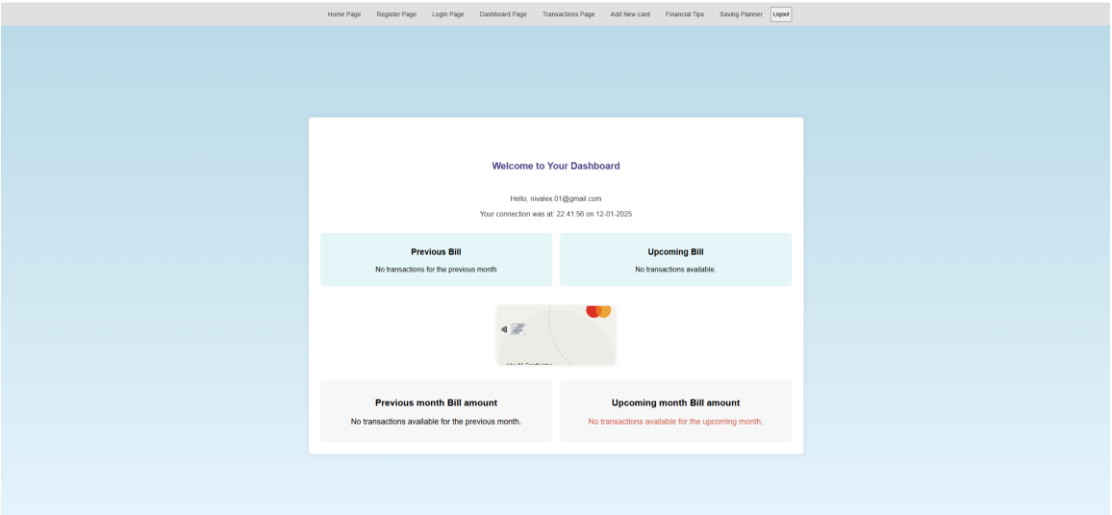




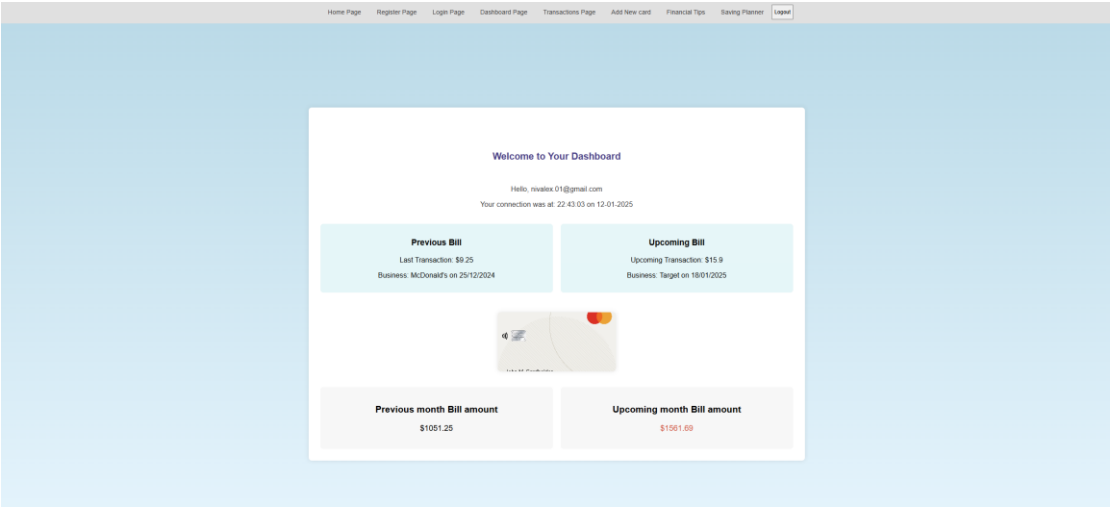
שגיאה באחד מפרטי ההרשמה (במקרה זה מישהו מתחת לגיל 16 מנסה להירשם):



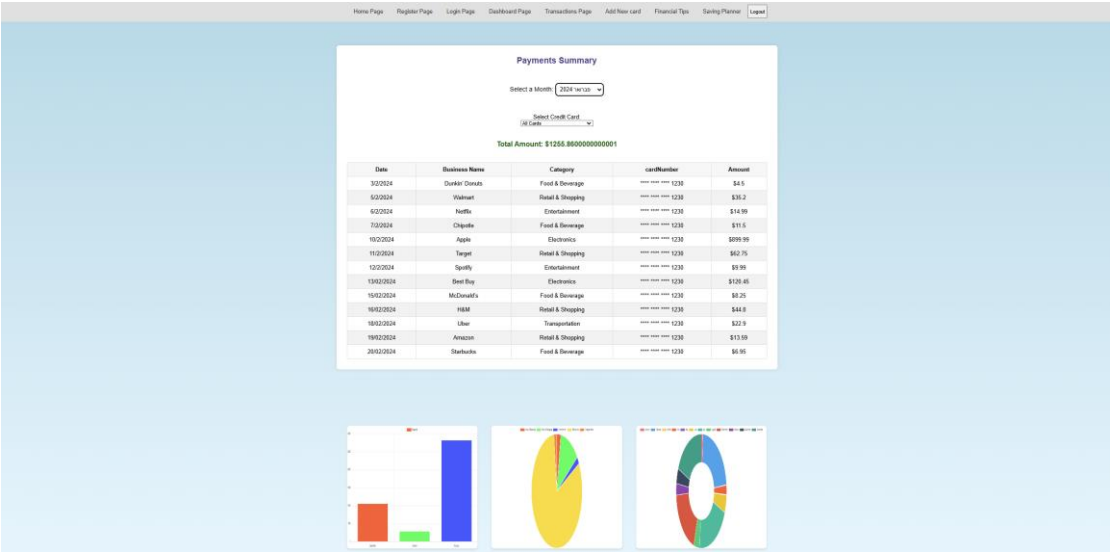
מסך הדשבורד טרם טעינת הטרנזקציות ע"י המשתמש:



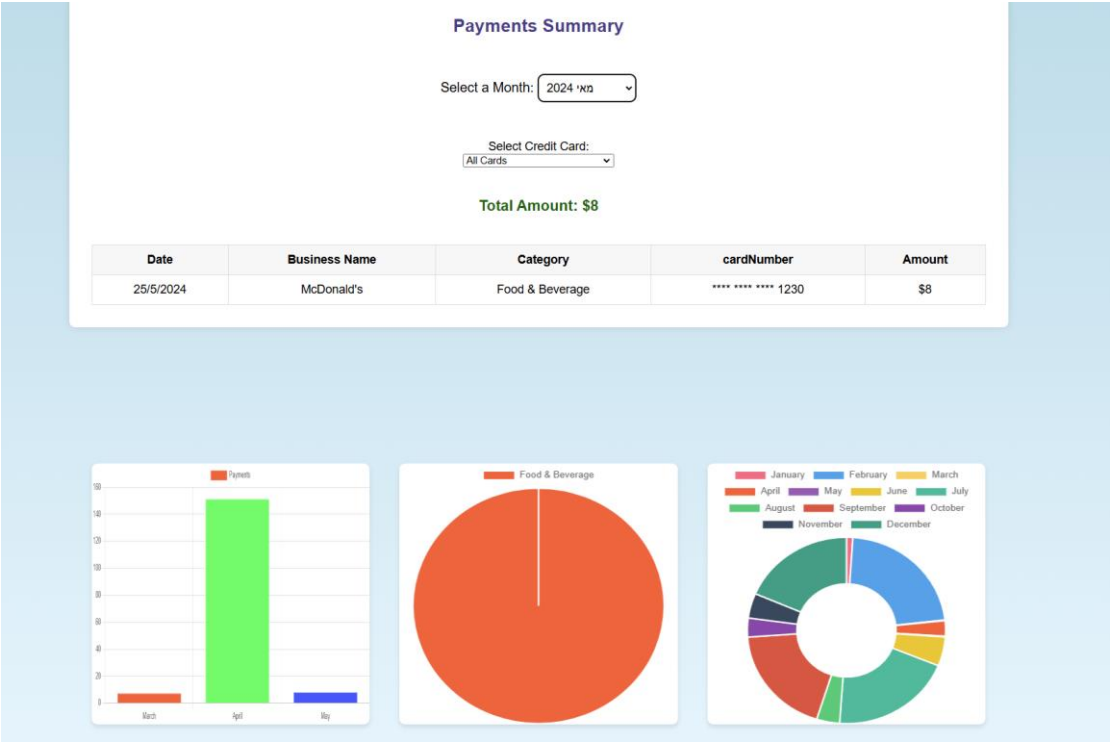
מסך הדשבורד לאחר טעינת הטרנזקציות ע"י המשתמש:



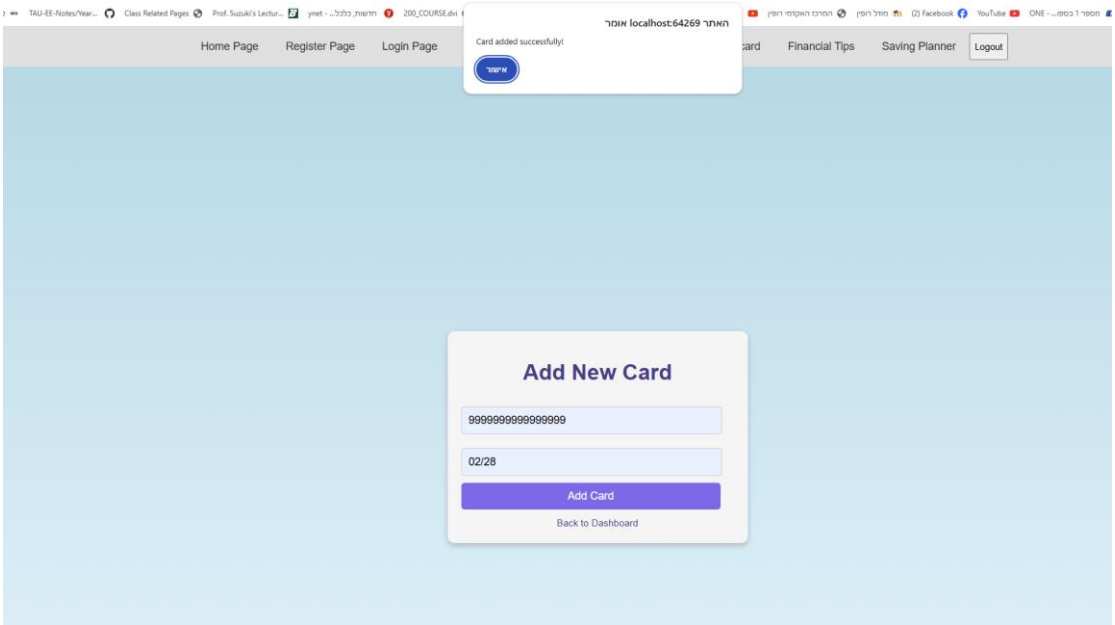
מסך התשלומים שלי:



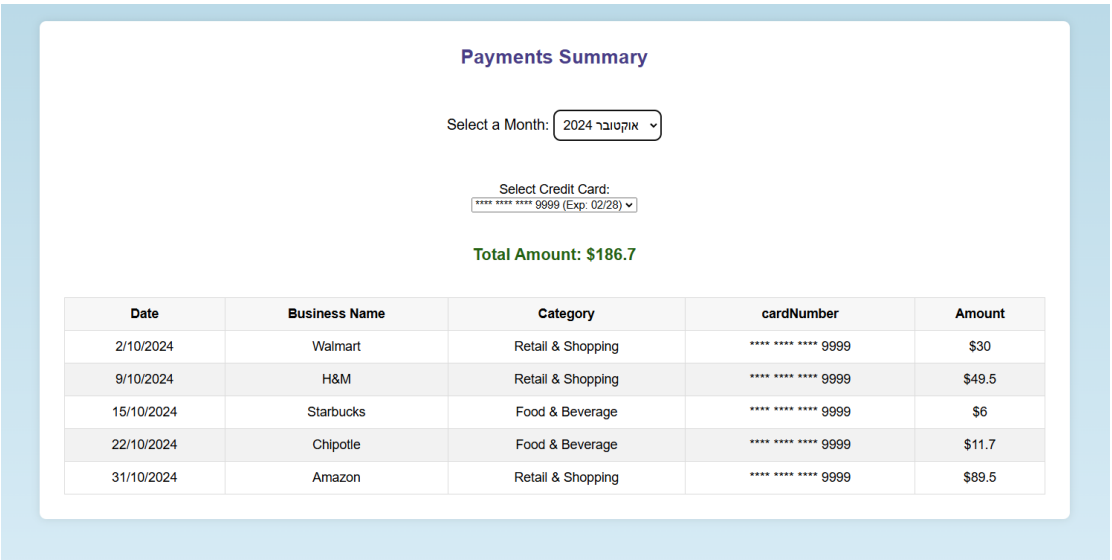
גרפים בעמוד התשלומים שלי:



הוספת כרטיס אשראי חדש:

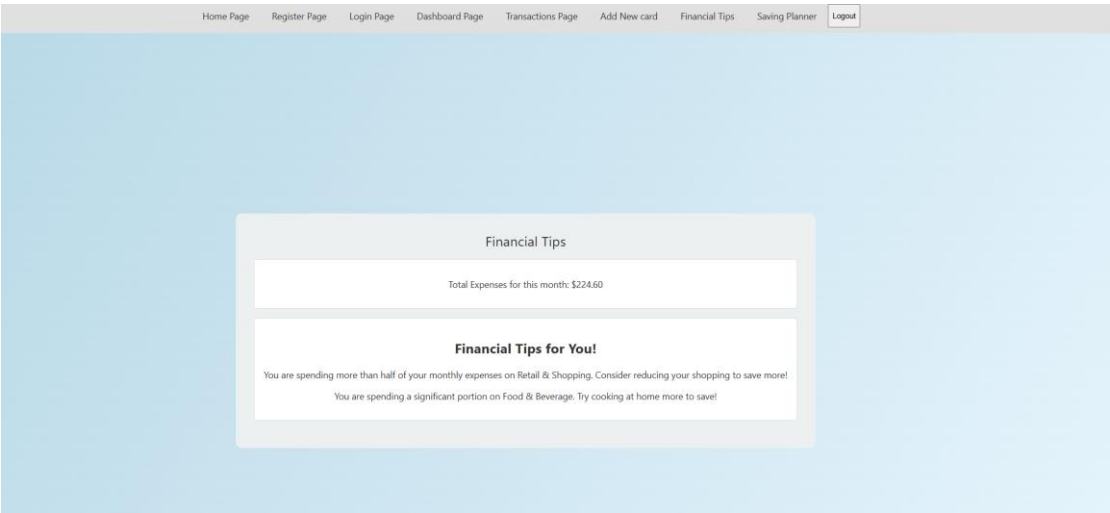


כעת בעמוד התשלומים שלי ניתן לראות לפי 2 הכרטיסים:

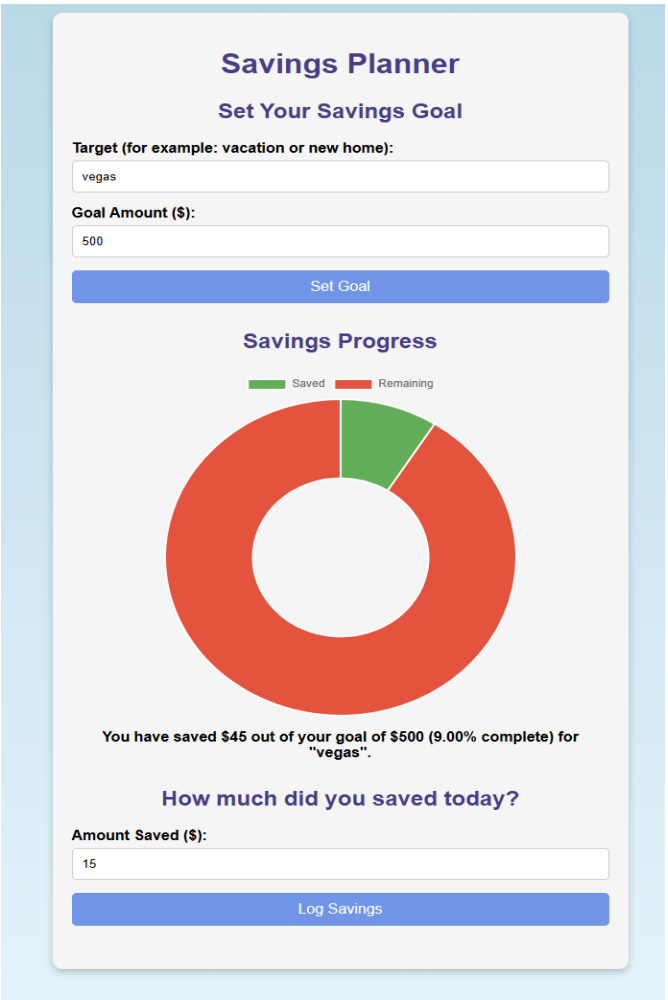


Date	Business Name	Category	cardNumber	Amount
2/10/2024	Walmart	Retail & Shopping	**** **** **** 9999	$30
9/10/2024	H&M	Retail & Shopping	**** **** **** 9999	$49.5
15/10/2024	Starbucks	Food & Beverage	**** **** **** 9999	$6
22/10/2024	Chipotle	Food & Beverage	**** **** **** 9999	$11.7
31/10/2024	Amazon	Retail & Shopping	**** **** **** 9999	$89.5

עמוד טיפים פיננסים (הנתונים דינמיים, יכול להיות שבעת ההרצה החודש יהיה שונה מהחודש שאנחנו הרצנו ולכן יופיעו נתונים אחרים)



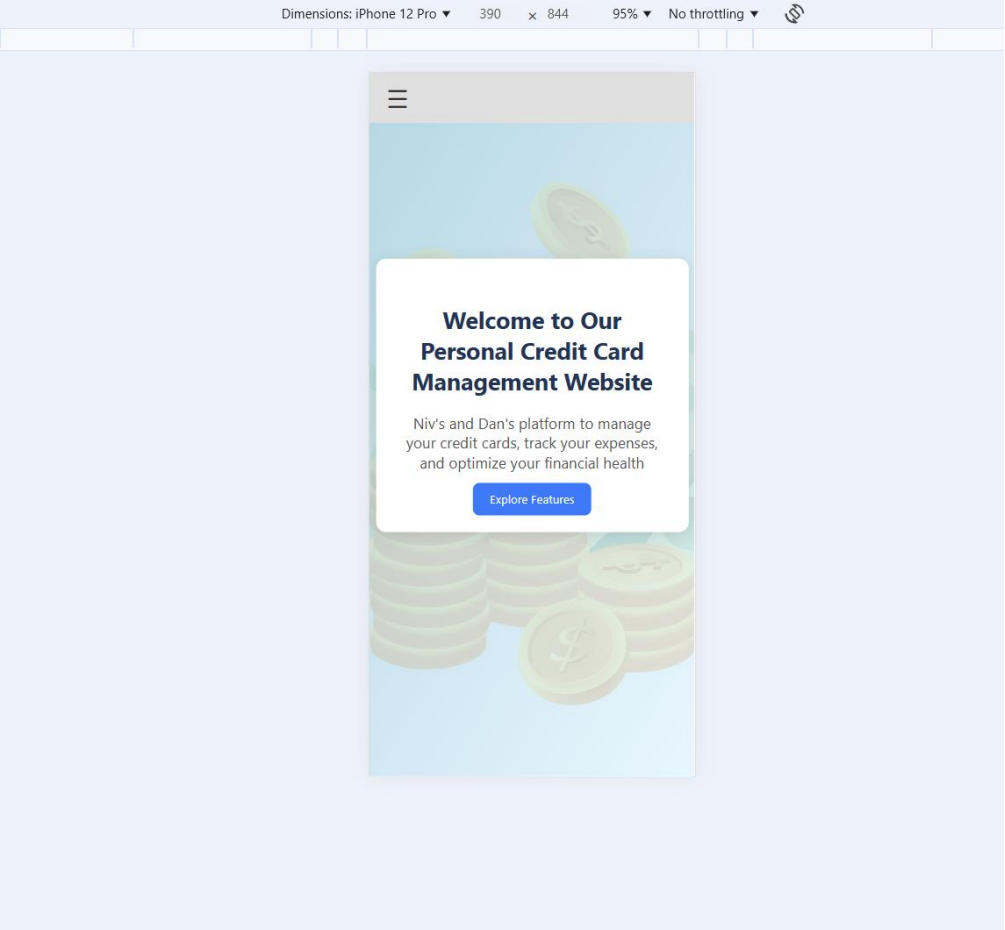
עמוד מתכנן פיננסי:



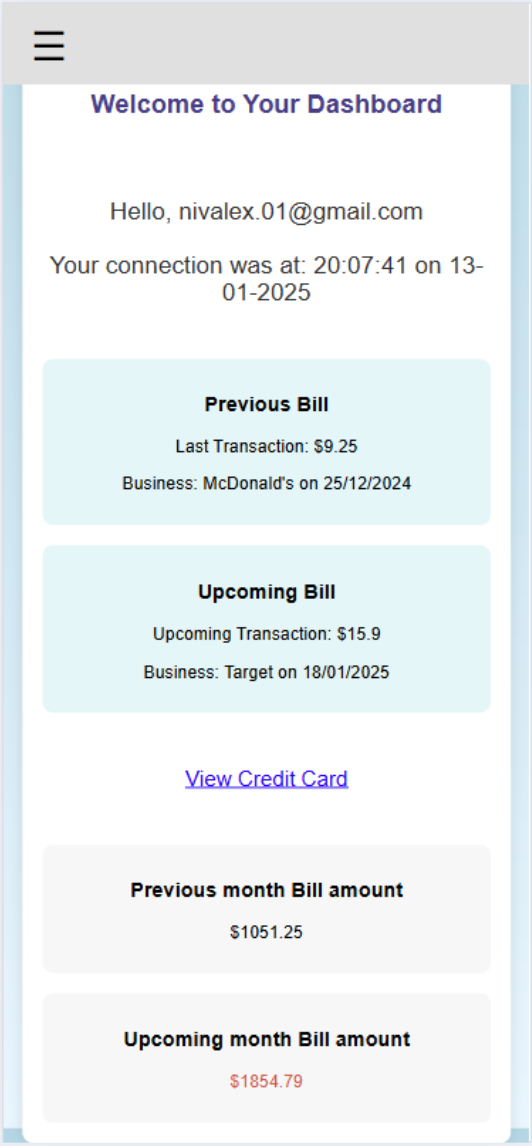
- הוספנו 15 דולר ל-30 דולר שכבר היו שמורים קודם ב- local storage עבור טיסה לוואגס.

צילומי מסך עבור פלטים במצב (phone):

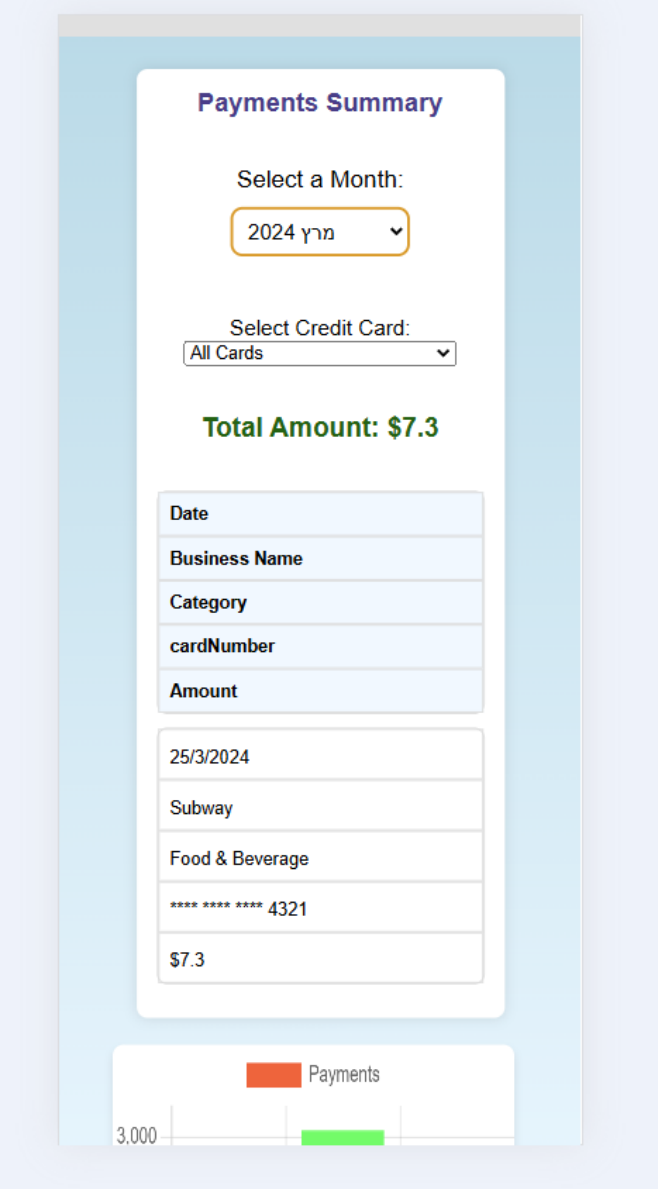
עמוד הבית:



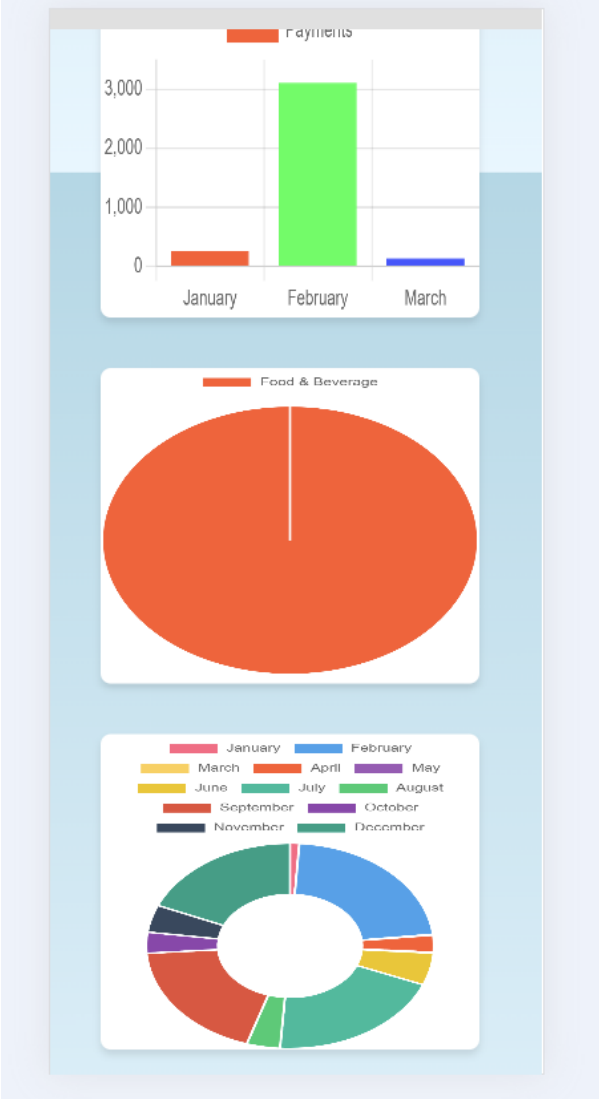
עמוד ה- dashboard במצב phone



עמוד התשלומים שלי במצב phone:



המשך עמוד התשלומים שלי במצב phone:



עמוד מתכנן פיננסי במצב phone: