**תרגיל בית מס'1**

**323419416 | 211849823 | 208169805 | 316568856 | 207533621 | 206705816**

מי לא הלך אי פעם לבית הקולנוע וגילה שיש לו אולם ריק להנאתו?

זה אולי נחמד ללקוח שמקבל אולם ריק אבל העסק לא מרוויח מכך.

החלטנו שניתן לגשת אל הבעיה הזו או אל חלקה בעזרת בניית אתר של "מכירות חיסול" לכרטיסי

סרטים ביום ההקרנה.

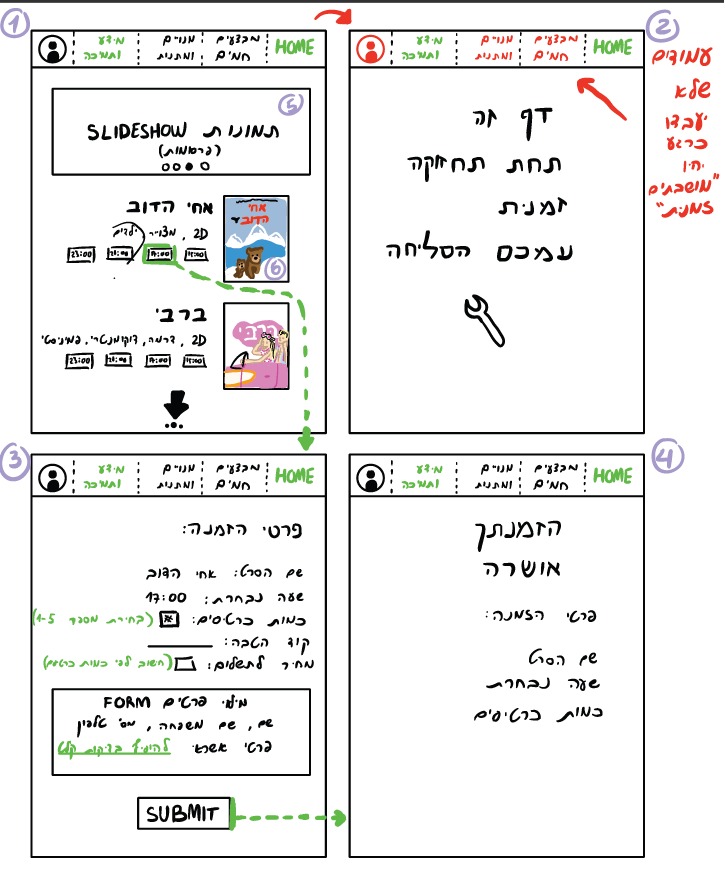
כרגע האתר שלנו יכול לקבל מידע מהלקוח, לבדוק אותו, ולהשתמש בו במידת הצורך.

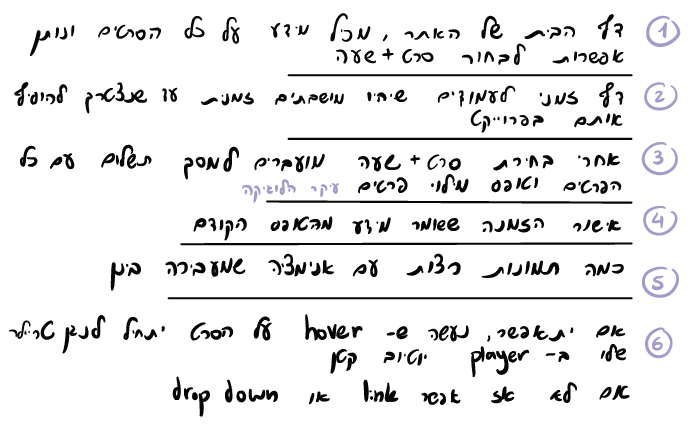
בעתיד אנחנו מתכננים להשתמש ב-Database שישמור מידע פנימי על הקולנוע ושכבה לוגית שתוכל להצליב את המידע הנקלט ביחד איתו.

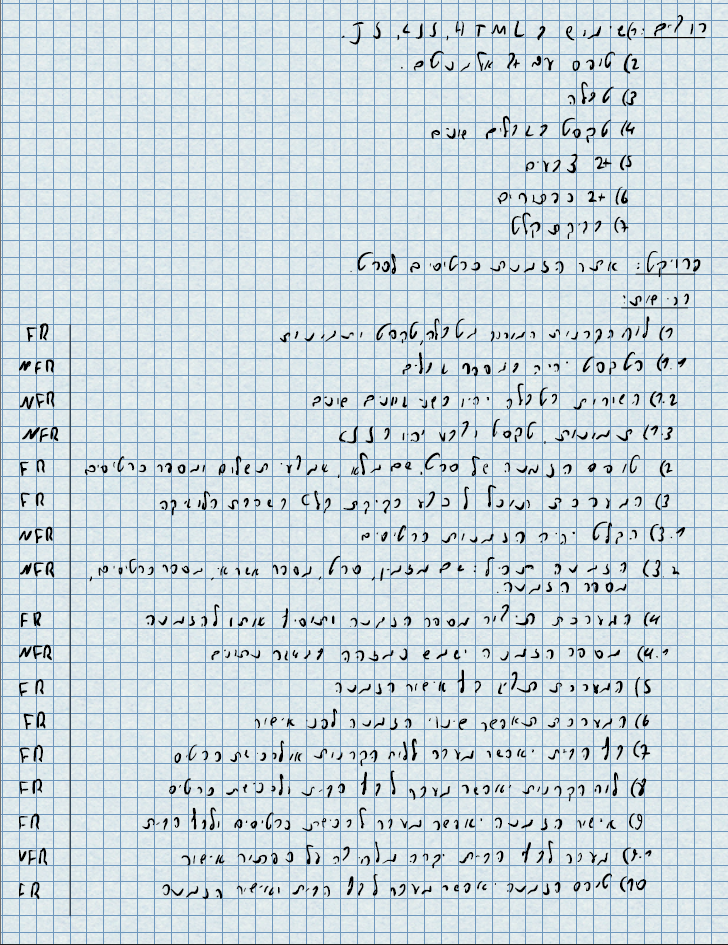
HTML : אלכסנדרה + דניאל

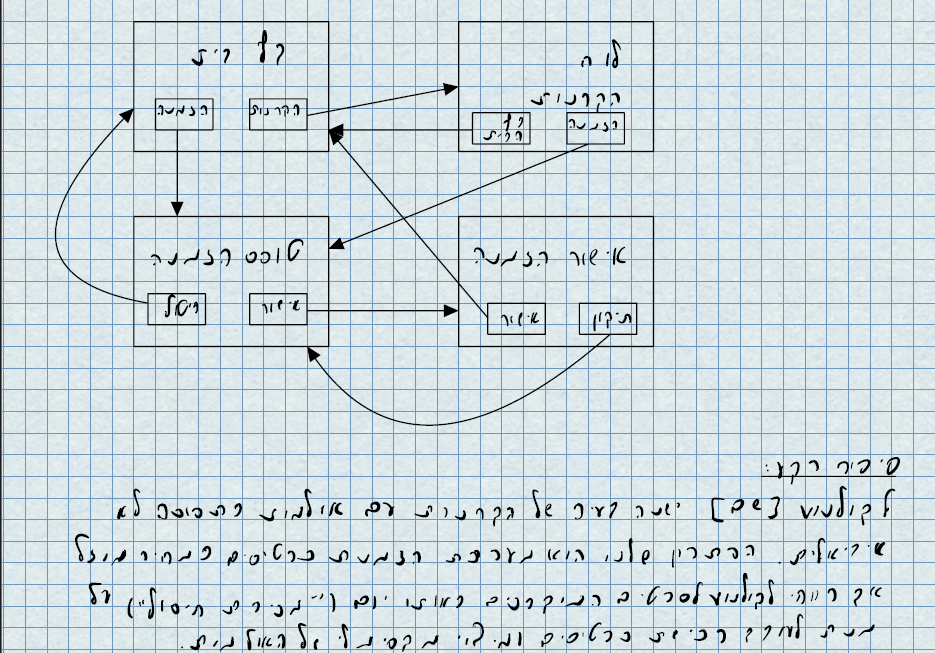
CSS: גבריאל + אמיר

JS : ישי + גלעד

**סקיצה ראשונית של תכנון האתר:**



****

****

**שכבת HTML:**

<link rel="icon" type="image/x-icon" href="..\media assets\website-logo.ico">

שימוש באלמנט אייקון כדי להגדיר את האייקון של האתר בראש העמוד בדפדפן



Figure הוא אלמנט אשר מכיל קבוצה של תמונות שמקושרות תחת קטגוריה משותפת אחת

<figure>

<img src="..\media assets\ad1.png">

<img src="..\media assets\ad2.png">

<img src="..\media assets\ad3.png">

<img src="..\media assets\ad4.png">

<img src="..\media assets\ad1.png">

</figure>

בכל דך HTML אנו טוענים את דף ה- Javascript הרלוונטי שלו כפי שאנו טוענים גם דפי CSS

<script src="..\logic and data layers\TheatrixLogic.js"></script>

בעת טעינה של הדף - מתבצעת הפונקציה. הפונקציה מגדירה את פרטי הסרט הנבחר ע''י שליפת נתונים מהזיכרון הזמני של הדפדפן (session storage)

<body onload="setTimeAndMovie()">

מתבצעת הדפסה של הפרטים של הסרט הנבחר בעזרת span כאשר במקום הריק אנו מחליפים את המחרוזת הריקה במחרוזת בעלת אותו id אשר שמורה ב session storage

<p>שם הסרט: <span id="movie-name"></span></p>

<p>זמן: <span id="time"></span></p>

בעת לחיצה על תפריט בחירת כמות הכרטיסים מתבצעת פנייה לפונקציה אשר מחשבת ומעדכנת את המחיר הכולל של הכרטיסים

<select class = "ticket-count-selection" id= "numTickets" onclick="ticketSum()">

באלמנט זה אנחנו נותנים למשתמש אפשרות להזין קוד הנחה. לאחר הזנה ולחיצה על הכפתור, מתבצעת בדיקה בשכבת הלוגיקה אם ההנחה תקינה ומתבצעת הדפסה אם הקוד תקין או לא + עדכון המחיר.

<p>קוד הנחה: <input type="text" id="discount-code">

<button class = "discount-button" type="button" onclick="checkDiscount()">הזן קוד</button>

<p id="discount-code-applied"><span class = "discount-msg" id="discount-code-applied-msg"></span> </p>

</p>

<p id="cost">מחיר סופי: <span id="totalPrice"></span> 35₪</p>

Fieldset הוא אובייקט שמקבץ אלמנטים תחת form יחיד

<fieldset>

<legend>פרטי המשתמש</legend><br>

<form id="user-info-form">

<label for="fname">שם פרטי:</label>

<input type="text" id="fname" name="fname"><br><br>

. . . .

</fieldset>

בעת טעינת העמוד, האתר טוען את המידע שהלקוח הזין בעת קניית הכרטיסים ושמור ב session storage

<body onload="loadOrderData()">

לאחר שכל המידע נטען בדף, הדף יציג את הנתונים הרלוונטיים בכל איזור רלוונטי עפ''י id  
CSS עובד בעזרת שימוש בclasses, ו- HTML עובד עם Javascript בעזרת id's

<fieldset>

<legend>פרטי הזמנה:</legend>

<div id="orderDetails">

<p>שם הסרט: <span id="movieName"></span></p>

<p>זמן: <span id="time"></span></p>

<p>מספר הכרטיסים: <span id="numTickets"></span></p>

<p>מחיר: <span id="price"></span>₪</p>

</div>

</fieldset>

Window location מעביר את המשתמש לעמוד אחר, פועל כמו ahref

<button class = "back-homepage" type="button" onclick = "window.location = 'homepage.html'">חזרה לדף הבית</button>

**שכבת CSS:**

בשכבה זו אנו ממשים את הנראות של כל מה שמוצג באתר - שינויים אלו נעים בין גדלי וצבע פונט ועד להתנהגות של אובייקטים שזזים בעזרת הנפשות (יפורט בהמשך).  
בשכבה זו אנו משתמשים ב-3 קבצים, כאשר כל אחד שייך לדף אחר באתר. בחרנו לפצל את קבצי ה-CSS ל-3 קבצים כדי להימנע מכפילויות שעלולות לקרות וגם כדי לעשות שינויים קטנים היכן שנרצה במקום להוסיף עוד הגדרות.

כחלק מתהליך העיצוב למדנו מספר הגדרות שונות שמשפיעות על ההתנהגות והנראות של שלל האלמנטים:

background-position: top;

background-repeat: no-repeat;

background-size: cover;

background-attachment: fixed;

כאן השתמשנו בארבעת ההגדרות האלו על מנת לקבע את הרקע על העמוד כך שכאשר מתבצע זום אין/אאוט הרקע ישאר סטטי ולא יתמרכז או יתרחק גם יחד עם העמוד וגם יהיה בגודל מלא.

div.menu {

height: 5rem;

direction: rtl;

text-align: center;

position: sticky;

top: 0;

z-index: 1000;

background-color: rgb(0, 0, 0);

}

כאן יצרנו תפריט בראש העמוד אשר "מודבק" לאזור העליון של העמוד בעזרת התכונה sticky ובעזרת z index  
ויורד תמיד למטה ביחד עם גלילה באתר, כלומר הוא תמיד יהיה ממוקם למעלה. לאחר מכן יצרנו בתוכו את הכפתורים אשר מובילים לאזורים שונים באתר (חלקם מבוטלים כרגע).

בחלקים רבים של הפרויקט השתמשו בתכונות border וב- padding/margin המאפשרים ליצור מסגרת סביב האלמנטים וגם לשנות כיצד היא נראית. כמו כן השתמשנו ביחידות מידה שונות חוץ מ-px.  
השתמשנו ביחידות מידה em, vh, ו-rem.

יחידות המידה האלה הן יחידות מידה אוניברסליות אשר מגדילות אלמנטים באופן יחסי לפי גודל הפונט של האתר לפי הזום של האתר ונוח להשתמש בהם כדי ליצור מראה אחיד יותר באתר.

border-radius: 15px;

border-style: groove;

padding-right: 2em;

padding-left: 2em;

border-right: 2px solid rgb(255, 255, 255);

border-left: 2px solid rgb(255, 255, 255);

border-top: 2px solid rgb(255, 255, 255);

border-bottom: 2px solid rgb(255, 255, 255);

על הכפתורים באתר השתמשנו בתכונה hover כדי לשנות את הנראות שלהם כאשר המשתמש מזיז עליהם את העכבר. השתמשנו גם בתכונה cursor כדי להראות אזורים שהמשתמש לא יכול לגשת אליהם כרגע על מנת לא ליצור בלבול מדוע כפתורים מסוימים אינם לחיצים. כמו כן הדגשנו כל כפתור שעובד כדי שהמשתמשים ידעו שניתן ללחוץ עליהם ומה הבחירה שלהם.

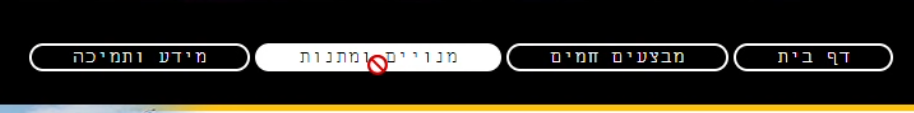
div.menu a.restrictAcess:hover {

color: rgb(0, 0, 0);

background-color: rgb(255, 255, 255);

cursor: not-allowed;

}





את הלוגו של האתר, הפרסומות והאייקון יצרנו בפוטושופ (כמה תמונות להמחשה) והלכנו על color-scheme של צבעים סגולים, שחור, אפור ולבן:



למדנו גם להשתמש בתכונות כמו flexbox, box, ו- inlineblock כדי להגדיר אלמנטים מסוימים בתור "קופסאות" כדי שנוכל להזיז את המיקום שלהם בעמוד ולהגדיר להם את הגובה והרוחב יותר בקלות.  
למשל התצוגה של התמונה של כל סרט מיוצגת כך:

div.menuphoto{

position: sticky;

top: 0;

z-index: 2000;

display:block;

max-block-size: 0px;

width: 100px;

}

בראש העמוד עשינו מצגת של פרסומות שזזות באופן אוטומטי בעזרת הנפשה:  
המצגת מוגדרת כתמונה גדולה בגודל 500% אשר רק חמישית ממנה מוצגת בכל רגע נתון. המצגת פיזית מורכבת מ-4 קבצי תמונה (+חזרה על התמונה הראשונה) וכל תמונה תופסת 100% וכך בעצם מוצגת רק תמונה אחת בכל רגע, כאשר בתצוגה אנו מזיזים את המיקום שמוצג ע''י תכונת keyframe.

אנו קובעים מה האזור המוצג במצגת ע''י שימוש בהפרשי זמן קבועים אשר "מקדמים" את האיזור המוצג במצגת ויוצרים מעבר חלק בין השקופיות.

#slider {

overflow: hidden;

position: relative;

width: 63.3rem;

top:0;

bottom: 0;

left: 0;

right: 0;

margin: auto;

}

#slider figure {

position: relative;

width: 500%;

margin: 0;

left: 0;

animation: 20s slider infinite;

}

#slider figure img {

width: 20%;

float: left;

}

התכונות החשובות כאן הן overflow - אשר גורמת לכך שכל מה שחורג מגבולות האלמנט לא יהיה מוצג ולא יהיה ניתן לגלול בתוך האלמנט עצמו כדי לראות את המידע החבוי (במקרה זה - שאר הפרסומות).

מה שגורם לתזוזה של המצגת היא התכונה animation אשר פועלת באורך 20 שניות סה''כ ומבצעת את האנימציה בשם slider באופן אינסופי.

@keyframes slider {

0% {

left: 0;

}

20% {

left: 0;

}

25% {

left: -100%;

}

45% {

left: -100%;

}

50% {

left: -200%;

}

70% {

left: -200%;

}

75% {

left: -300%;

}

95% {

left: -300%;

}

100% {

left: -400%;

}

}

האנימציה עצמה מוגדרת בעזרת keyframes אשר מגדירים את התנהגות האנימציה לפי רגעים/פריימים:  
כאשר האנימציה ב-0% כלומר 0 שניות המיקום של המצגת יהיה 0 מתוך 500 (כאשר כל 100 אחוזים מהווים תמונה אחרת). לאחר מכן לאחר 5 שניות (25%) מיקום המצגת יהיה מינוס 100% כלומר תהיה תזוזה שמאלה ותוצג תמונה אחרת במצגת וכך הלאה עד אשר נסיים את כל ההנפשות ונחזור להתחלה.



Drop shadow מאפשר לנו ליצור אפקט של צל/זוהר סביב אובייקט. השתמשנו בו לתמונות של הסרטים ולחלק מהכפתורים. הערכים בתכונה מהווים את המרחק בציר x, מרחק בציר y, את הפיזור של הצל ולבסוף את הצבע.

את הטבלה עיצבנו בצבעים שונים לכל שורה ע''י שימוש בתכונה even/odd  
משום שהשתמשנו בטבלאות בתוך טבלה ראשית כל שורה נספרה בין אם היא הייתה בתוך טבלה פנימית או בטבלה החיצונית פתרנו את בעיה זו ע''י הפיכת כל הטבלה לשקופה.

table {

background-color: transparent;

border-collapse: collapse;

width: 50rem;

margin: auto;

}

table tr.movie-listing:nth-child(even) {

background-color: #f2f2f2;

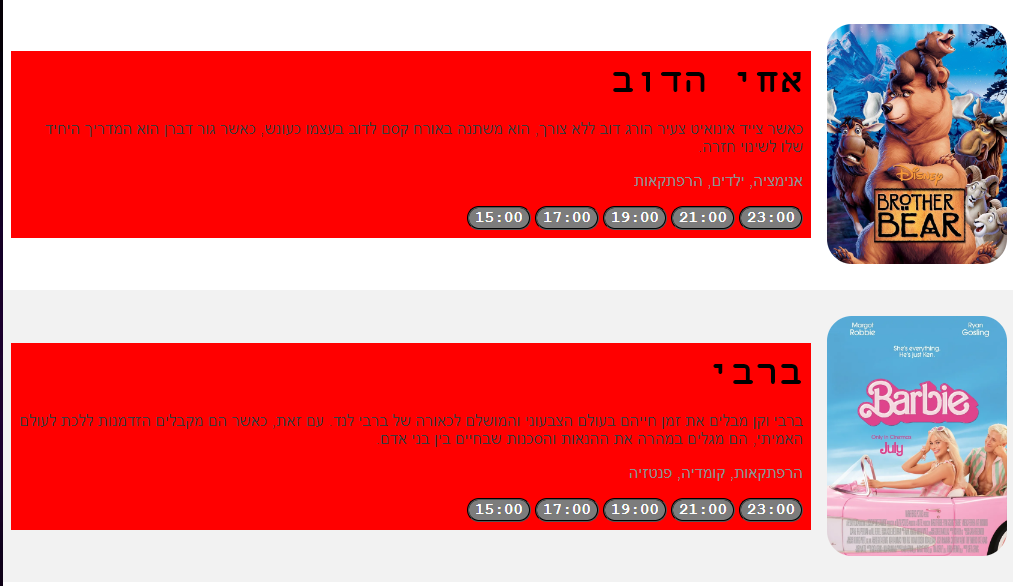
}

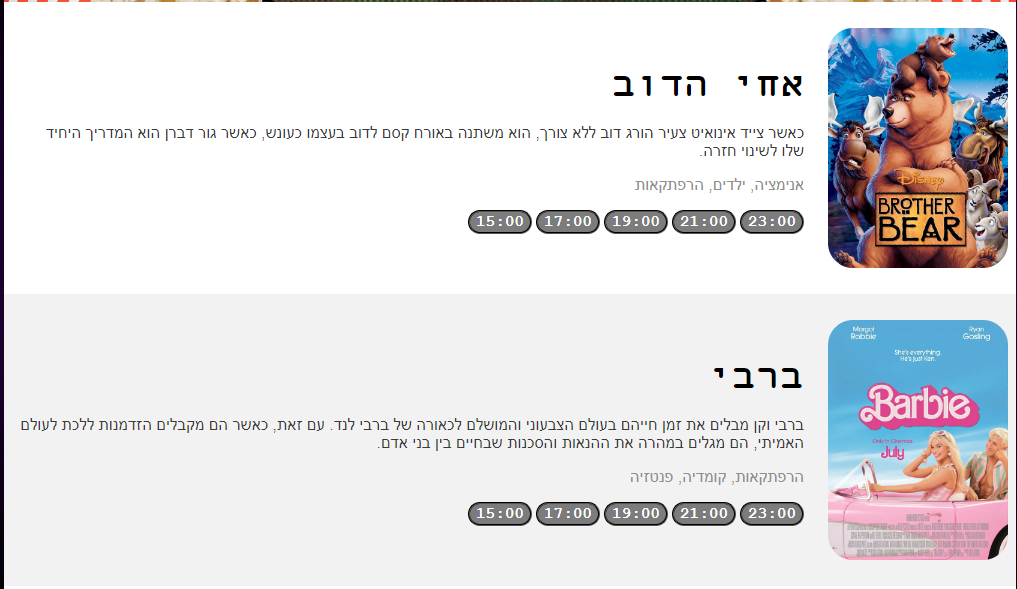
table tr.movie-listing:nth-child(odd) {

background-color: #ffffff;

}

**השוואה בין 2 הצורות (עם רקע שקוף ובלי)**





האלמנט legend יוצר איזור תחום עם כותרת שמוצגת בתוך איזור תחום גם כן שניתן לערוך עם סגנונות שונים כמו טבלה.

legend {

display:table;

text-align: center;

margin-left: auto;

margin-right: auto;

width: 200px;

font-family: 'Courier New', Courier, monospace;

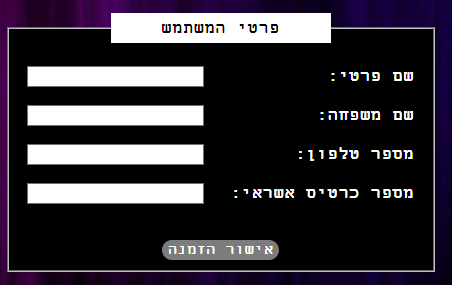
font-weight: bolder;

font-size: large;

background-color: rgb(255, 255, 255);

color: rgb(0, 0, 0);

padding: 5px 10px;

}

האלמנט fieldset מקבץ מספר אלמנטים קשורים בתוך איזור form יחד עם שימוש באלמנט legend

fieldset {

direction: rtl;

background-color: #000000;

color: white;

...}

**שכבת ניהול הנתונים:**

בשכבה זו מימשנו פונקציה לשמירת נתונים בעזרת - session storage באותו האופן כמו שהכנסנו מידע

ל - local storage בהרצאה, השתמשנו ב - ™ תו שכמעט לא משומש בכדי להפריד את הנתונים ב - stringify אשר נוכל להיות בטוחים שלא יופיע במידע שנכנס. וget-רים מסויימים על מנת להוציא את המידע החוצה גם כן כמו שעשינו בכיתה.

בחרנו להשתמש ב - session storage מאחר שאין לנו צורך לשמור את המידע ליותר מדי זמן. למידע שנשמר יש תוקף מוגבל שלאחריו או לאחר סגירת הטאב. ה - session storage הוא פתרון זמני, בעתיד אנחנו מתכננים לעשות אינטגרציה ל - Database כלשהו שיחליף את השימוש בכלי זה.

function processInfo(first\_name, last\_name, card\_number, ticket\_num, hour, movie\_name, phone\_number)

{

var orderStr = stringify(first\_name, last\_name, card\_number, ticket\_num, hour, movie\_name, phone\_number);

sessionStorage.setItem('order', orderStr);

}

function stringify(first\_name, last\_name, card\_number, ticket\_num, hour, movie\_name, phone\_number) {

var firstnameStr = first\_name;

var lastNameStr = '™' + last\_name;

var cardStr = '™' + card\_number;

var ticketStr = '™' + ticket\_num;

var hourStr = '™' + hour;

var movieStr = '™' + movie\_name;

var phoneStr = '™' + phone\_number;

var ordStr = firstnameStr + lastNameStr + cardStr + ticketStr + hourStr + movieStr + phoneStr;

return ordStr;

}

function getMovieName(id)

{

var name =sessionStorage.getItem(id)

return name.split('™')[5];

}

function getMovieHour(id)

{

var hour =sessionStorage.getItem(id)

return hour.split('™')[4];

}

function getTicketNum(id)

{

return sessionStorage.getItem(id).split('™')[3];

}

**שכבת לוגיקה:**

בשכבה זו מימשנו מספר פונקציות על מנת לבדוק את קלט שנשלח על ידי המשתמש ובכדי לבצע מעבר של הקלט התקין אל ה - session storage.

פונקציות בדיקת הקלט שלנו:

בדיקת שם/שם משפחה:

נעזרנו ביצירת קבוצת regex שמאפשרת לנו להחזיר False לכל מחרוזת שיש בה תו לא מהקבוצה

function nameCheck(name)

{

var name\_pattern = /[^a-zA-Z\u0590-\u05fe\s]/;

return name\_pattern.test(name);

}

בדיקת כרטיס אשראי:

עיבוד הנתונים קורה בפונקציה מסודרת עם פונקציות עזר לניקוי ובדיקה של המידע הנקלט.  
למשל, נעזרנו באלגוריתם לון (Luhn) על מנת לוודא את תקינות מספר האשראי שהוכנס. בדיקת התקינות של מספר הכרטיס לפני הרצת האלגוריתם נעזרת בקבוצת regex על מנת למצוא כל תו שאינו מספר.  
(מספרי אשראי מזויפים מאתר PayPal לדוגמה: 4032031398926708, 4032037073483576, 4032035017017690)

function cardChecker(card\_num)

{

// Checks for the content and length of card\_num

if (/[^0-9-\s]+/.test(card\_num) || card\_num.length != 16)

{

return false; // usage of invalid characters or invalid length

}

//n\_check is the sum that needs to be checked by the algorithm

var num\_check = 0;

var is\_even = false;

for (var i = card\_num.length - 1; i >= 0; i--)

{

var char\_digit = card\_num.charAt(i);

var int\_digit = parseInt(char\_digit, 10);

if (is\_even && (int\_digit \*= 2) > 9)

{

int\_digit -= 9;

}

num\_check += int\_digit;

is\_even = !is\_even;

}

return (num\_check % 10) == 0;

}

בדיקת מספר הפאלפון:

נעזרנו ביצירת קבוצת regex שמאפשרת שמאפשרת לקבל איזשהו פאטרן אפשרי למספרי פלאפון המוכרים לנו בארץ.

function CheckPhoneNumber(phone\_number)

{

var israeli\_phone\_pattern = /^0(50|52|53|54|58)\d{7}$/; //creates a pattern to check (05XYYYYYYY)

if (israeli\_phone\_pattern.test(phone\_number)) // crosses pattern against the number that was sent

{

return true;

}

return false;

}

בנוסף, מימשנו בשכבת הלוגיקה את מעבר העמודים בעזרת window.location על ידי השמה של URL חדש לחלון:

window.location = 'orderpage.html';

בשכבת הלוגיקה ישנן גם פונקציות הנועדו לעדכן את המידע המוצג באתר באופן דינמי כמו הפונקציה ticketSum אשר מעדכנת את המחיר של הכרטיסים שרוצים להזמין לפי קוד הנחה שהוזן ומספר הכרטיסים.

בנוסף פונקציה אשר תבדוק קודי הנחה מסוימים checkDiscount:

function ticketSum()

{

var base\_cost = 40;

var ticket\_sum = base\_cost \* document.getElementById('numTickets').value;

document.getElementById('cost').innerText ='מחיר סופי:' + ticket\_sum + '₪';

}

function checkDiscount()

{

var dis\_code = document.getElementById('discount-code').value;

if(dis\_code == "IDF4U")

{

current\_ticket\_price = 28;

DiscountSuccessful();

}

else

{

current\_ticket\_price = 35;

DiscountUnsuccessful();

}

}

function DiscountSuccessful(){

ticketSum();

document.getElementById('discount-code').value = '';

document.getElementById('discount-code-applied-msg').innerText = "קוד הוזן בהצלחה";

document.getElementById('discount-code-applied-msg').style.color = 'green';

document.getElementById('discount-code-applied-msg').style.fontWeight = bold;

}

function DiscountUnsuccessful(){

ticketSum();

document.getElementById('discount-code').value = '';

document.getElementById('discount-code-applied-msg').innerText = "קוד שגוי";

document.getElementById('discount-code-applied-msg').style.color = 'red';

document.getElementById('discount-code-applied-msg').style.fontWeight = bold;

}

רצינו להשתמש בשם הסרט ובזמן ההקרה בעמוד קליטת הנתונים העיקרי שלנו, הפתרון שלנו לכך היה לשלוח בזמן לחיצה את המידע הנחוץ ל - session storage. לאחר סיום הקליטה אנו עוברים לדף האחרון שלנו שבו מדפיסים את סיכום ההזמנה שלנו ומוחקים את ההזמנה מה - session storage שלנו כי אין לנו יותר צורך בה.

function setTimeAndMovie()

{

document.getElementById('movie-name').innerText =sessionStorage.getItem('movie\_name');

document.getElementById('time').innerText = sessionStorage.getItem('time');

}

function loadOrderData()

{

var name = getMovieName('order');

document.getElementById('movieName').innerText = name;

document.getElementById('time').innerText = getMovieHour('order');

document.getElementById('numTickets').innerText = getTicketNum('order');

document.getElementById('price').innerText = getTicketNum('order')\*40;

sessionStorage.removeItem('order');

}