ספר פרוייקט גמר – י"ד הנדסת תוכנה

שם הסטודנט: ינון-יאיר זוהר
ת.ז הסטודנט 213882442
שם המנחה: אילן פרץ
שם הפרוייקט: MyGamesList
סמל מוסד:
שם המוסד: מכללת כנפי רוח קרית נוער
תאריך הגשה:

<u>הצעת הפרוייקט שאושרה על ידי משרד החינוך</u>

פרטי מגיש ההצעה

סמל מוסד: 140129

שם מכללה: כנפי רוח קריית נוער ירושלים.

שם הסטודנט: ינון-יאיר זוהר

שם הפרויקט: MyGamesList

פרטי הפרויקט

תיאור הפרויקט

כגיימר אני מתקשה לעקוב אחרי איזה משחקי מחשב שיחקתי ומאיזה נהניתי, ולמצוא משחקים חדשים לשחק. בגלל הפירוד בין שירותי מכירת המשחקים השונים, אין מקום יחיד בעל אלגוריתם המלצות שמכיל את כל המידע על כל המשחקים ששיחקתי,
לכן בתור פרויקט בחרתי לפתח אתר WebApp שמאפשר לגיימרים (הלקוח) לעקוב אחרי משחקי הוידאו שהוא שיחק במקום מרוכז שעובד על הפלטפורמות השונות, ולקבל המלצות למשחקים שיתכן שייהנה מהם על פי העדפות הקודמות שלו. הפרויקט יהיה מסוג פתרון אלגוריתמים ויתמקד במערכות המלצה.

הגדרת הבעיה האלגוריתמית

הבעיה האלגוריתמית היא מתן המלצות, האתגר המרכזי היא ביצוע חיזוי של העדפות של המשתמש על בסיס המידע הקיים במערכת בצורה יעילה. הפתרון לבעיה יבוצע באמצעות מערכת המלצות מסוג סינון שיתופי, שתעשה שימוש Matrix factorization, וכן תתחשב בעוד נתונים חיצוניים. על ידי עיבוד המידע של כל המשתמשים ימצאו קשרים בין משתמשים דומים וההבדלים יומלצו למשתמש. המערכת תאפשר למשתמשים לקבל המלצות שאינן תלויות בשירות רכישת המשחקים המועדף עליהם ותעזור להם למקד את ההעדפות שלהם ולנהל אותם באופן נוח ומרוכז מהמכשירים השונים

רקע תיאורטי בתחום הפרויקט

תחום דעת מרכזי: מערכות המלצה
קיימים כמה סוגי מערכות המלצה שניתן לחלק לארבע סוגים

מערכות מבוססות תוכן - מתבססות על תוכן או תיאור הפריטים. המערכות בקטגוריה זו מתחלקות לשלושה סוגים, מבוססות על תוכן פרופיל משתמש ותיאור פריטים, מבוססות על נתוני פרופיל המשתמש (גיל, מיקום גאוגרפי ועוד) ומבוססות על היסטוריית הגלישה של המשתמש
מערכות מבוססות משוב - מערכות אלו תלויות בשיתוף פעולה מצד המשתמש. קטגוריה זו מחולקת לשיטות המבוססות על דירוג פריטים ושיטות המבוססות על דירוג קשרים בין פריטים
מערכות סינון שיתופי- מערכות שעובדות על ההיגיון שמשתמשים שהסכימו בעבר יסכימו בעתיד
מערכות משולבות(HYBRIDS) - מערכות אלו משלבות בין השיטות השונות על מנת לתת המלצות מדויקות יותר

רוב המערכות בשוק כיום הם היברידיות ככה שאין באמת אלגוריתם אחד ספציפי, כל מערכת היא שילוב של מספר מערכות. המערכת ההיברידית נוצרת במיוחד על הצרכים של המערכת הספציפית ולכן ההמלצות שלה מדויקות יותר וכן היא יכולה להתמודד על נתונים מסובכים יותר וקהל רחב יותר
המערכת שלי גם תעבוד גם כן בשיטה היברידית ותתבסס בעיקרה על שילוב של מערכות מבוססות משוב ומערכות סינון שיתופי, ותתחשב בנתונים חיצוניים שונים (כגון דירוג של משחק על ידי מבקרים).
המערכת תקבל את הדירוג של המשתמש על כל משחק, ותבנה ממנו מטריצה של הקשרים בין המשתמשים למשחקים. אותה היא תחלק לשני מטריצות (בשביל חיסכון של מקום ויעילות מוגברת) על פי הכפלה של שני המטריצות המערכת תוכל לבצע השוואה בין המשתמשים ולמצוא משתמשים עם העדפות דומות, על פי הקשרים האלו המערכת תמצא את ה "חוסרים" (הבדלים) בן כל משתמש דומה ותמליץ את ההבדלים ביניהם.
בחירתי במערכת מסוג סינון שיתופי נובעת מתוך הנטייה של קהילת הגיימרים להסכים עם בעלי דעה דומה והחלוקה הטבעית בין מחנות מעריצים של הז'אנרים השונים.

דוגמא והסבר על האלגוריתם
נניח שיש לנו את טבלת הדירוג של 5 משתמשים ו-5 סרטים, והדירוגים הם מספרים שלמים הנעים בין 1 ל-5, המטריצה מסופקת על ידי הטבלה למטה.

	Movie1	Movie2	Movie3	Movie4	Movie5
U1		5	4	2	1
U2	1			5	3
U3	1	4	4	1	
U4			2		2
U5	3	1	1		

מכיוון שלא כל משתמש נותן דירוגים לכל הסרטים, יש הרבה ערכים חסרים במטריצה וזה מביא למטריצה דלילה. לפיכך, ערכי האפס שלא ניתנו על ידי המשתמשים יתמלאו ב-0 כך שהערכים המלאים יסופקו עבור הכפל. לדוגמה, שני משתמשים נותנים דירוג גבוה מסוים הסרט משוחק על ידי השחקן והשחקנית האהובים עליהם או שז'אנר הסרט הוא סרט פעולה וכו'. מהטבלה למעלה, נוכל לגלות שהמשתמש1 והמשתמש3 שניהם נותנים דירוגים גבוהים ל-move2 ולסרט3. לפיכך, מפירוק המטריצה, אנו מסוגלים לגלות את התכונות הסמויות הללו כדי לתת תחזית על דירוג ביחס לדמיון בהעדפות המשתמש ובאינטראקציות.

בהינתן תרחיש, משתמש 4 לא נתן דירוג לסרט 4. ברצוננו לדעת אם משתמש 4 יהיה מרוצה מסרט 4. השיטה היא לגלות משתמשים אחרים עם העדפות דומות של משתמש 4 על ידי לקיחת הדירוגים שניתנו על ידי משתמשים בעלי העדפות דומות לסרט 4 ולחזות אם משתמש 4 ירצה את הסרט 4 או לא.

הגדר קבוצה של משתמשים (U), פריטים (D), גודל R של |U| ו-|D|. המטריצה |D|*|U| כולל את כל הדירוגים שניתנו על ידי המשתמשים. המטרה היא לגלות K תכונות סמויות. בהינתן עם הקלט של שתי מטריצות מטריצות (|U|*k) P ו-(|D|*k) Q, זה ייצור את תוצאת המוצר R.

$$\mathbf{R} \approx \mathbf{P} \times \mathbf{Q}^T = \hat{\mathbf{R}}$$

מטריצה P מייצגת את השיוך בין משתמש לתכונות בעוד שמטריצה Q מייצגת את השיוך בין פריט לתכונות. אנו יכולים לקבל את החיזוי של דירוג של פריט על ידי חישוב מכפלת הנקודה של שני הוקטורים המתאימים ל-u_i ו-d_j.

$$\hat{r}_{ij} = p_i^T q_j = \sum_{k=1}^{k} p_{ik} q_{kj}$$

הליכים עיקריים

המערכת תאפשר לכל משתמש

- לבצע התחברות (LOGIN)

כל משתמש יבצע רישום במערכת, לאחר הרישום המערכת תשמור את המידע שלו בפרופיל אישי

- לחפש משחקים

המשתמש יוכל לחפש משחקים כדי לקבל מידע עליהם. משם המשתמש יוכל להוסיף את המשחק לרשימת המשחקים ,ולאחר מכן יוכל המשתמש לסמן אותו בתגית סטטוס (משחק כרגע, מתכנן לשחק, עזוב סיים) וכן לדרג את שביעות הרצון שלו מהמשחק

- לנהל את האוסף של המשחקים (למחוק, להוסיף) ולדרג כל משחק

המשתמש יוכל להכנס לרשימת המשחקים שלו על מנת לנהל ולעדכן את הנתונים שם. כגון שינוי דירוג של משחק, מחיקה של משחק שהתווסף. וכן שינוי של סטטוס המשחק

- לקבל המלצות על משחקים חדשים לשחק

החלק העיקרי של האתר. פה המשתמש יוכל לקבל המלצות על משחקים חדשים שיעניינו אותו. ההצעות יצאו מתוך האלגוריתם ההמלצות של האתר ויותאמו להעדפות של כל משתמש. משם המשתמש יוכל לסמן כל משחק שהוא מביע בו עניין ולהוסיף אותו לרשימת המשחקים שלו.

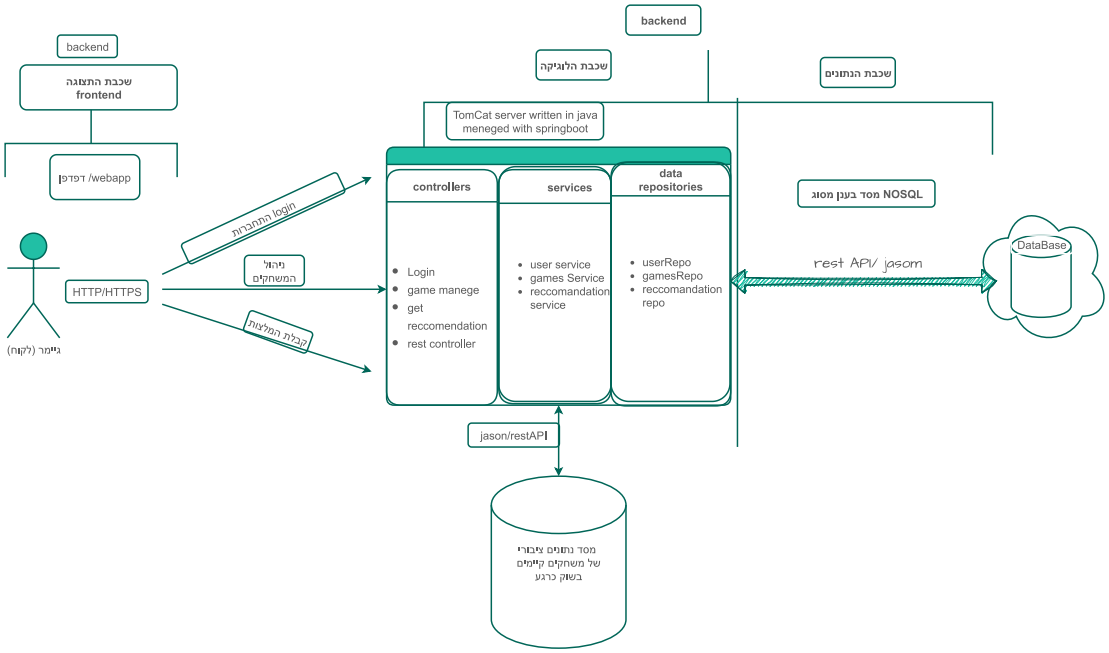
תיאור פרוטוקולי תקשורת
:Web Socket
1. תקשורת בזמן אמת דו-כיונית. באתר שלי, ישמש Web Socket לעדכונים אוטומטיים של פעולות משתמש והתראות חיוניות.

:TCP

2. פרוטוקול העברת נתונים ברשת, מאפשר העברת נתונים רציפה ואמינה. ישמש להעברת דפים מהשרת למשתמש בדפדפן.

HTTP (ובעברית: פרוטוקול העברת טקסט):
3. פרוטוקול תקשורת בין רשתות. ישמש באתר שאני מפתח על פי HTTP לקבלת בקשות מהלקוח בווב, כמו הצגת רשימת המשחקים ופרטי המשתמש.

תיאור טכנולוגית והנדסה



הלקוח של התוכנה יהיה גיימר אשר מעוניין לעקוב אחרי המשחקים שהוא שיחק ולמצוא משחקים חדשים לשחק
המשתמש יוכל לבצע התחברות, דבר שיאפשר למערכת ליצור לו פרופיל אישי ולהשתמש במידע שהוא מכניס כדי לתת לו המלצות רלוונטיות
צד הלקוח יהיה אפליקצית WEB. משם המשתמשים יתממשקו עם השירות.
יעבוד על HTML CSS JAVA SCRIPT ויתקשר בHHTP
השרת שלו יהיה שרת TOMCAT יכתב בשפה JAVA בעזרת ו spring boot Vadin. יכתב בארכיטקטורת dependency injection והוא יתקשר בHTTP.
יורכב מכמה רכיבים מרכזיים
controllers
- Login

- game controller
- get recommendation
- rest controller

services

- **user service**
- **games service**
- **recommendations service**

repositories

- userRepo
- gmaesRepo

מסד הנתונים

יעשה שימוש במסד נתונים MongoDB atlas

Atlas mongoDB הוא מסד נתונים לא טבלאי (noSQL) שיושב בענן

להלן האוספים שיכיל מסד הנתונים (נתון לשינוי)

- אוסף (Users) המשתמשים שמות משתמש, סיסמאות מוצפנות,
- המשחקים של המשתמש, יכיל את סטטוס המשחק (שוחק, משחק כרגע, עזוב, סיים לשחק) וכן שביעות רצון ממנו
- אוסף משחקים קיימים בשוק (כתחלופה יתכן ביצוע השוואה מול מאגרים ציבוריים על ידי שימוש בAPI)

פיתוחים עתידיים

- אפשרות לשליפת מידע באופן אוטומטי מחשבונות המשתמש באתרים אחרים
- תמיכה בפלטפורמות נוספותclient לXBOX למשל)
- מעקב אוטומטי בעזרת שירות Windows אחרי המשחקים שרצים במחשב והוספתם לחשבון המשתמש באופן אוטומטי

פרטים פורמליים

לוח זמנים:

שלבי עבודה	לסיים עד לתאריך
בחירת פרויקט, חקירה ולמידה לעומק של נושאי הפרויקט	**1.12.23**
כתיבה והגשת הצעת הפרויקט לאישור משרד החינוך	**7.12.23**
מימוש הקוד של האלגוריתם המרכזי, ביצוע בדיקות ושיפורים	**15.1.24**

בניית צד שרת	**5.2.24**
בניית מסד הנתונים ושילובו	**19.2.24**
בניית צד לקוח	**4.3.24**
כתיבת ספר הפרויקט	**25.3.24**
הגשת הפרויקט כולו (ספר + קוד) להגנה וקבלת ציון מגן	**1.4.24**

מנחה בפרויקט: **מר אילן פרץ**
חתימת הסטודנט:
Yinon Zohar
חתימת רכז המגמה: _____

תקציר\מבוא

הרקע לפרויקט
אני משחק משחקי מחשב מאז שאני ילד, ואחד הדברים שכל גיימר מתקשה איתו זה להחליט על איזה משחק לשחק. השוק כיום מוצף באלפי משחקים שונים בסגנות וז'אנרים רבים. ומשחקים חדשים יוצאים בקצב שקשה לעקוב אחריו. הרבה גיימרים כיום מתקשים למצוא משחקים חדשים מעבר למה שהמעכל שלהם משחק וכתוצאה מפספסים משחקים שהיו יכולים להנות מהם מאוד. לכן החלטתי לבנות אתר שיאפשר לגיימרים הכניס את הרגלי המשחק שלהם ולקבל המלצות על משחקים שיתכן שיהנו מהם

תהליך המחקר
המחקר שלי עסק בעיקר במערכות ההמלצה השונות היתרונות והחסרונות של כל אחד מהם והשימושים השונים בהם. במהלך המחקר למדי על האלגוריתמים השונים והצורה שבא כל אחד מהם עובד. לאחר שלמדתי על הטכנולוגיות השונות הגעתי למערכות מסוג collaborative filtering שנראו כאילו הם יתאימו למשימה שלי

סקירת ספרות
?????

אתגרים מרכזיים
- בחירת אלגוריתם המלצות מתאים
- מציאת הפרמטר עליו יתבצעו חישובי ההמלצות
- בחירה של כלים לבניית האתר ומאגר המידע(DATABASE)
- בניית המודל וקישורו לאתר

הסיבות לבחירת הפרוייקט

אני משחק משחקי מחשב כבר הרבה זמן ותמיד התקשיתי למצוא מה לשחק, דבר שהחמיר בשנים האחרונות בגלל מיעוט הזמן שיש לי להשקיע בדבר וכן הפיצול של שווקי המשחקים השונים.
לכן רציתי לבנות אתר מרוכז שבו כמה שיותר מהפלטפורמות נמצאות כדי לנסות לגשר ביניהם ולעזור לאנשים שלא יכולים להשאר מעודכנים בכל משחק שיוצא

מוטיבציה לעבודה

רצון לגשר בין הפלטפורמות השונות ולעזור לגיימרים כמוני לבחור משחקים בצורה נוחה וחכמה יוצר

הפתרון שהפרוייקט מעניק

הפרוייקט נותן מענה לאנשים שלא יכולים להשקיע זמן בלחפש ולחקור משחקים שונים כדי לבחור מה לשחק, האתר נותן מענה של מערכת המלצות שחולשת על כל הפלטפורמות המכירה השונות שרבים מהם לא מציעים מערכת המלצות בכלל. ההמלצות מותאמות אישית למשתמש במקום להציע רק את מה שפופולרי עכשיו ככה שהם מגוונות יותר, וכן סיכוי יותר טוב שהלקוח יהיה מרוצה מהם (מאחר והם מורכבות מהעדפות שלו)
בנוסף האתר יהיה נגיש מכל מכשיר בעל דפדפן ככה שהלקוח לא יצטרך להשתמש במחשב\קונסולה כדי לראות את ההמלצות שלו ולעדכן את הנתונים שלו במערכת

פתרונות לבעיה

לבעיה יש כמה פתרונות. אך לרבים מהם יש חסרונות

- לשאול חברים מה הם משחקים
- לקרוא מאמרים באינטרנט (כגון חדשות על משחקים חדשים,רשימות של משחקים מומלצים וכו')
- לקבל המלצות מפלטפורמות הרכישה כגון STEAM,Epic Games ועוד

אחת הבעיות המשותפות לכל הפתרונות הללו שהם צורכים מהמשתמש לבצע חיפוש בעצמו.
בנוסף אף אחד מהם לוקח בחשבון את ההעדפות של הלקוח, כך שהפתרונות האלה אולי יתאימו לחלק מהאנשים זה לא ייתן מענה למי שיש טעם יותר ספציפי\ייחודי.

בנוסף יש חסרונות נוספים כמו אנשים שאין בקרבתם אנשים עם עניין במשחקים, דרישה של זמן מהלקוח בחיפוש וחקר על משחקים שונים, והגבלה של פלטפורמות מסוימות והמשחקים שהם מציעים

מטרות\יעדים

לפתח מערכת שתענה על כל הבעיות בצורה נוחה יעילה. ותייצר המלצות רלוונטיות לכל משתמש בהתאמה להעדפות שלו

המערכת צריכה להיות נגישה וקלה לשימוש עם מינימום דרישות מהמשתמש

המערכת צריכה להיות זמינה ממכשירים רבים

אתגרים

העדפות של בני אדם הוא דבר מסובך שמורכב מהרבה גורמים, לגרום לתוכנת מחשב להבין את הצורה שהמוח עובד הוא דבר לא פשוט

ניהול המידע – באתר יהיה מידע נרחב של הרבה משתמשים, דבר המצריך ניהול חכם של המידע

בניית אתר שתענה על הצרכים של הלקוח בצורה הטובה ביותר

מדדי הצלחה למערכת

- *המערכת צריכה לאפשר למשתמש לנהל את רשימת המשחקים האישית שלו להוסיף למחוק ועוד*
- *על המערכת לספק המלצות למשתמש בצורה נוחה ויעילה מכל הפלטפורמות השונות*
- *המערכת צריכה להיות נגישה מכל מקום*

רקע תיאורטי\ספרות מקצועית

תיאור מצב קיים

על פי סקר שערך yougov.com הרבה שחקנים לא יודעים מהיכן לגלות משחקים חדשים

יותר מ 15%לא יודעים מהיכן הם מגלים משחקים ויותר מ25% לא משתמשים באף אחד המדרכים שהוצעו

תחום רכישת המשחקים עבר שינוי מאוד נרחב בעשור האחרון, המעבר ממשחקים פיזיים (דיסקים, קסטות וכדומה) למשחקים דיגיטליים גרם לכך שהאפשרויות נרחבות מתמיד, הגישה לכל משחק דרך האינטרנט נתן לשחקנים אפשרות לשחק משחקים שלא הייתה להם גישה אליה לפני כן.

כמו כן תעשיית הגיימינג גדלה מאוד בשנים האחרונות ושווה יותר מ 500 ביליון דולר. זה אומר שחברות חדשות מצטרפות כל הזמן והאפשריות רק הולכות וגדלות. אם לפני עשר שנים היו כמה חנויות בודדות שמכרו משחקים כיום יש עשרות פלטפורמות שונות שנועדו למכור אותם, דבר שגורם לקושי משמעותי בבחירה של משחקים

דבר נוסף שגורם לקושי במציאת משחקים היא החלוקה של פלטפורמות המכירה בין החברות והבלעדיות של חלק מהפלטפורמות על משחקים מסוימים.
שתעשיית המשחקים התחילה לעבור לאונליין הייתה פלטפורמה אחת מרכזית שמכרה משחקים, Steam.הדבר הקל מאוד על השחקנים וכן על המפתחים שכן הכל היה מרוכז במקום אחד. אבל החברות האחרות החלו לפתוח פלטפורמות משלהם כדי להימנע מהאחוזים ש STEAM לוקחת מכל מכירה, והחלו למכור את המשחקים שלהם בפלטפורמות ייחודיות משלהם, דבר שאולי גרם לתחרות אבל גם לריבוי פלטפורמות מכירה, דבר שגורם לאי נוחות מיותרת לשחקנים עד כדי כך שהם לא ירכשו משחק אם אינו נמכר בSTEAM
למשל משתמש הגיב בדיון בנושא "האם תסרב לרכוש משחק אם אינו מוצע מכירה בSTEAM

" Yup, I refused to buy starcraft. I definitely would have bought if it were available on Steam."
תרגום: "כן, אני מסרב לרכוש את סטארקראפט. בוודאות הייתי רוכש אותו אם היה מוצע למכירה בSTEAM

משתמש אחר הגיב
" Never; I am, however, sometimes tempted to not to buy a game if I know it has to be registered on steam"
תרגום: "לא. אבל לפעמים אני נמנע מלקנות משחקים אם אני יודע שהוא לא מוצע ב STAEM"

מקורות
https://business.yougov.com/content/47794-how-gamers-across-6-key-markets-discover-new-video-games

https://www.giantbomb.com/forums/general-discussion-30/poll-did-you-not-buy-a-game-because-not-on-steam-w-512176/

ניתוח חלופות מערכתי

?????

תיאור החלופה הנבחרת

הפתרון שאני בחרתי הוא יצירת אתר שיאפשר למשתמשים לקבל המלצות על מה לשחק מחוץ להגבלות של פלטפורמות המכירה השונות. האתר יאפשר למשתמשים להזין את נתוני המשחק שלהם ולקבל על פיהם המלצות מותאמות אישית להעדפות שלהם, בנוסף המערכת תציע מקום מרוכז בשביל המשתמש לעקוב אחרי המחשקים שהוא שיחק מכל מקום מבלי להכנס לכל פלטפורמה בנפרד, הבחירה במערכת זו היא משום שהיא עונה על כל הדרישות לפתרון. ואפיון של מערכות דומות למוצרים שונים כגון https://www.imdb.com/ לסרטים או https://myanimelist.net/ לסדרות וספרים ממוצא יפני

איפיון המערכת שהוגדרה\מוצעת

דרישות המערכת

המערכת צריכה לאפשר למשתמש התחברות (LOGIN) כדי שתוכל לאחסן את המידע שלו

המערכת צריכה לתת המלצות ייחודיות לכל משתמש על פי הנתונים שלו. היא תתחשב בהרגלי המשחק של המשתמש ותיתן המלצות שיוצרו במיוחד בשבילו

המערכת צריכה להתחשב בנתונים חיצוניים כגון דירוג המשחק כדי לאפשר המלצות מדויקות יותר

מודול המערכת

המערכת היא מערכת מסוג אתר אינטרנטי ופועלת על פי תבנית ה MVC (Model, View, Controller)

Model

השכבה בה כל החישובים של המערכת קורים.
המודל הוא בעצם האלגוריתם המרכזי שמייצר את ההמלצות. הוא יכתב בJAVA וירוץ בצד השרת של האתר

View

שכבת התצוגה: האתר עצמו
Vaadin הוא כמו ערכת כלים עבור מפתחי אתרים שמעדיפים לעבוד עם Java או שפות אחרות מבוססות JVM. זה מייעל את תהליך בניית יישומי אינטרנט, וחוסך מהמפתחים לצלול לעומק המורכבות של JavaScript ו-HTML. עם Vaadin, יצירת ממשקי משתמש אלגנטיים ומודרניים הופכת להיות קלה, הודות לאוסף הרכיבים והכלים המובנים מראש לניהול נתונים ותקשורת בין הלקוח לשרת.

Controller

זו השכבה שאחראית לקשר בין כל השכבות השונות.
במערכת שלי יעשה שימוש ב Spring framework וב SPRINGBOOT בפרט. הם יאפשרו בניה קלה של המערכת. היא עובדת על עיקרון

dependency injection דבר שמאפשר בחירה של רכיבים בתוכנה בזמן הידור במקום בזמן ריצה.

אפיון פונקציונלי

ביצועים עיקריים

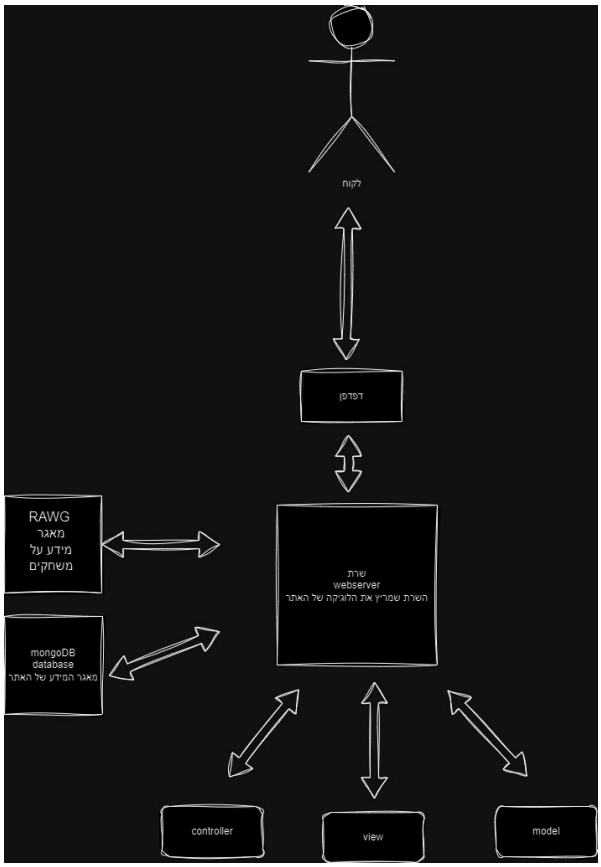
למשתמש יש כמה דברים עיקריים שהוא יכול לבצע במערכת

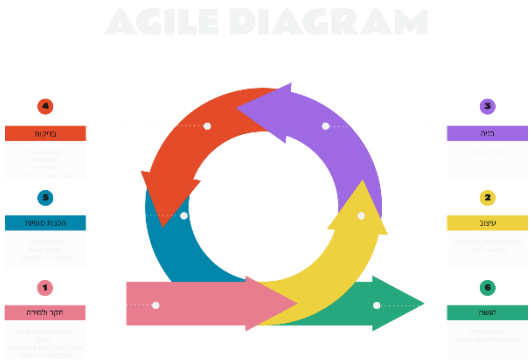
- הרשמה. המשתמש יוכל להרשם בערכת כדי לאפשר לה לאחסן את הנתונים שלו והציע משחקים לפיהם
- התחברות. לאחר ההרשמה המשתמש יוכל להתחבר למערכת כדי לראות את הפרטים שלו, לקבל המלצות ולערוך את רשימת המשחקים שלו
- לקבל המלצות על משחקים חדשים לשחק

אילוצים

- שימוש במאגר מידע על משחקים קיימים. המערכת לא תוכל להחזיק מידע על כל המשחקים הקיימים בשוק כל רגע נתון. לכן היא תשתמש ב API של שרת חיצוני שייתן לה מידע על משחקים קיימים
- התחברות: בשביל שהמערכת תוכל להכיר כל משתמש הוא יהיה חייב לבצע התחברות. לא ניתן להיכנס לאתר כאורח
- אחסון של המידע של כל המשתמשים: המערכת תהיה חייבת לשמור את המידע של כל המשתמשים שלה במאגר מידע בענן לא תאפשר הרצה מקומית

תיאור הארכיטקטורה





<u>תיאור רכיבים</u>

- שרת הנתונים יעשה שימוש ב mongoDB. שירות מאגר מידעלא טבלאי בענן .פה יאוחסנו כל הנתונים של המערכת. נתוני משתמשים מידע טבלת ההמלצות ועוד
- RAGW: מאגר נתונים של משחקים. המערכת תפנה אליו באמצעות rest API שהוא מציע כשירות על מנת לקבל מידע על משחקים
- שרת הWEB SERVER השרת המרכזי של המערכת. השרת יכתב בvaadin ויעשה שימוש בTOMCAT שם האתר ירוץ ושם יקרו כל הפעולות של המערכת
- צד הלקוח יהיה דפדפן לבחירתו

ארכיטקטורת רשת

Vaadin: Vaadin הוא framework מבוסס Java לבניית אפליקציות אינטרנט. הוא מאפשר ליצור רכיבי ממשק משתמש אתר באמצעות קוד Java, שאז מתורגמים ל-HTML, CSS ו-JavaScript עבור הדפדפן. התפקיד שלו בעיקרון הוא לספק ממשק משתמש ידידותי וגמיש עבור האתר.
בנוי על פרוטוקולי האינטרנט התקניים כמו HTTP (הפרוטוקול הידוע להעברת מידע בין השרת לדפדפן), תקשורת אסינכרונית (WebSockets) לתקשורת בזמן אמת בין הלקוח והשרת, ו-AJAX (שיטת פניה אסינכרונית לשרת לעדכון תוכן בדף באופן דינמי בלי לטעון את הדף מחדש).

Spring Boot: Spring Boot הוא framework שנבנה מעל Spring framework ליצירת מהירה של אפליקציות עצמאיות בסביבת Spring. הוא מקל על התקנת והתצורה של אפליקציות Spring, ומספק מגוון של תכונות כגון הגדרה אוטומטית, שרתים מוטמנים וניהול תלות. התפקיד שלו הוא לטפל בלוגיקה בצד השרת ולסייע באינטגרציה של רכיבים שונים באפליקציה.
היא עשויה להשתמש ב- HTTP או ב-HTTPS עבור תקשורת מאובטחת. כאשר Spring Boot משתמש בתקשורת בין שירותים ברקע, זה עשוי להשתמש ב- REST (Representational State Transfer) לשירותי RESTful API,

Tomcat: Apache Tomcat הוא יישום קוד פתוח של טכנולוגיות Java Servlet, JavaServer Pages, Java Expression Language ו-Java WebSocket. הוא פועל כשרת אינטרנט, ואחראי על אירוח והפעלת אפליקציות אינטרנט בנויות ב-Java. התפקיד שלו הוא לטפל בבקשות HTTP, לנהל PORTS ולשרת תוכן סטטי ודינמי למשתמשים.
, Tomcat תומך בתקשורת ב- WebSockets לתקשורת בזמן אמת בין הלקוח והשרת, וב-AJP (Apache JServ Protocol) לתקשורת בין Tomcat לשרת האינטרנט האחורי.

RAGW Game Database (API): RAGW הוא מסד נתונים למשחקים שנגיש באמצעות בקשות API. API (ממשקי תכנות ליישומים) מאפשרים למערכות תוכנה שונות לתקשר ולהתנגש זו עם זו. במקרה זה, ה-API מספק גישה לנתוני משחקים, שיכולים להיות משולבים באפליקציה כדי לשפר את התכונות שלה, כמו תצוגת מידע על משחקים, סטטיסטיקות או להקל על התנהלות המשתמש עם פלטפורמת המשחקים. התפקיד שלו הוא לספק גישה לנתונים ולשירותים חיצוניים, מעשיר את התכונות והתוכן של האתר.
משתמש בפרוטוקולי HTTP או HTTPS לביצוע בקשות ותגובות בין הלקוח (האפליקציה) לשרת. משתמש בשירות RESTful API, בפרוטוקולי HTTP ובפרוטוקולי תקשורת מבוססי JSON (JavaScript Object Notation) להעברת נתונים בין הלקוח והשרת.

תהליכי אבטחת המידע במערכת

בפרוייקט נעשה שימוש בהגנות על פי מתן הרשאות, יש שני תפקידים עיקריים, user וADMIN כאשר למשתמש בעל תפקיד ADMIN יש גישה לפעולות ודפים שלמשתמש רגיל אין

תהליך איסוף הנתונים

על מנת לאסוף את הנתונים פניתי לאתר https://www.kaggle.com/. לאחר חיפוש קצת נתקלתי במאגר מידע של שעות משחק של שחקנים בפלטפורמת מכירת המשחקים STEAM. המאגר הכי מידע של אלפי משמשים הכולל את המשחקים שהם רכשו\שיחקו ואת כמות הזמן שבילו בכל משחק. המידע הזה שימש אותי לבדיקת המערכת ויעזור למערכת כמידע התחלתי לפני שיהיו הרבה משתמשים

ניתוח ותרשים useCases \UML של המערכת המוצעת

האלגוריתם הראשי

האלגוריתם הראשי הוא אלגוריתם סינון שיתופי (collaborative filtering) מסוג matrix factorization

האלגוריתם מבוסס על ההנחה שאם אדם א' אוהב דברים דומים לאם ב' אדם א' יחלוק דעות נוספות עם אדם ב'
האלגורים שאני בחרתי מבוסס על gradient decent

האלגוריתם עובד על ידי פתירת טבלא שמכילה את האינטראקציות של המשתמש(במקרה שלנו הגיימר) עם המוצרים (במקרה שלנו המשחקים)

הטבלא בנויה בצורה שבא כל שורה מייצגת משתמש, כל עמודה מייצגת מוצר, והתאים מייצגים את האינטרקציה (יכול לבוא בצורה של לייק,דירוג, רכישה ועוד)

מטרת האלגוריתם היא למלא את המקומות החסרים (מוצרים ומשתמשים שלא הייתה בניהם אינטרקציה) ובכך להחליט מה ההמלצה של כל משתמש תיהיה

המערכת שאני בחרתי עושה את זה על ידי שימוש באגורים שנקרא gradient decent

מהו gradient decent?
gradient decent הוא אלגורים אופטימיזציה פשוט. הוא באופן איטרטיבי מעדכן משתנים ומשווא את התוצאות למידע האמיתי ולפי זה לומד. הוא ממשיך לעדכן את המשתנים עד אשר הוא מגיע לתוצאה קרובה מספיק לאמת

איך זה נראה בפועל במקרה שלנו?
נגיד ויש לנו את הטבלא הבאה

	מיינקרפט	פורטנייט	אמונג אס	קנדי קראש
דני	5	4	2	4
יואל	1		3	
שירה		2		3
אברהם	5		2	4
דנה	3			

כאשר כל עמודה היא משחק. כל שורה היא שחקן. והתאים הם דירוג שניתן למשחק על ידי המשתמש (תא ריק אומר שלא ניתן דירוג למשחק על ידי המשתמש)
נניח ואנחנו נרצה להמליץ משחק חדש לאברהם, ניתן לראות שהוא ודני נתנו דירוג דומה להרבה משחקים, ולכן נניח שהעדפות שלהם דומות
ומכיוון שדני נתן דירוג גבוהה לפורטנייט, נוכל להמליץ לאברהם לשחק פורטנייט גם

האלגוריתם שלי עובד על אותו עיקרון רק בצורה קצת יותר מתוחכמת
הוא מקבל את הטבלא הזאת ומפרק אותה לשני טבלאות קטנות אחת בגודל משתמשיX פיצרים(ראה בהמשך) ואחד בגודל פיצ'רים X משחקים
לשם ההדגמה מספר הפיצ'רים יהיה 2

לטבלא המקורית נקרא R
ולטבלאות הקטנות נקרא p וq
זה יראה ככה

	מיינקרפט	פורטנייט	אמונג אס	קנדי קראש
דני	5	4	2	4
יואל	1		3	
שירה		2		3
אברהם	5		2	4
דנה	3			

Q

מיינקראפט	מיינקראפט	אמונג אס	קנדי קראש
0.8	0.3	0.3	0.1
0.1	0.4	0.5	0.3

P

דני	0.1	0.3
יואל	0.2	0.2
שירה	0.01	0.4
אברהם	1	0.6
דנה	0.8	0.3

את הטבלאות האלו נמלא בערכים רנדומליים. אלו הם המשקולות. כמו שדויין מקודם הטבלא היא בגודל משתמשיXפיצ'רים. מספר הפיצ'רים הוא פרמטר שנקבע על ידי המתכנת והוא משפיע על כמה האלגוריתם יעמיק בנתונים, ניתן לחשוב עליהם כתכונות שהמשתמשים והמוצרים מחזיקים בהם. למשל האם המשחק הוא משחק מסוג פאזל? מי יצר את המשחק ועוד. הפיצ'רים בצד המשתמש מייצגים את החוזק של ההעדפה של המשתמש כלפי פיצ'ר זה, (חשוב לזכור: המערכת אינה יודעת מה כל פיצ'ר מייצג, הפיצ'ר הוא הבעה של משהו משותף בין משחקים ומשתמשים אבל המערכת אינה יודעת מה הוא מייצג). מספר נמוך מידי של פיצ'רים יכול להביא לפספוס של מידע אבל מספר גדול מידי עולה משאבים רבים ויכול לגרום לאימון יתר.

לאחר מכן נכפיל את הטבלאות
על פי הנוסחא להכפלת
מטריצות

2 × 2 Matrix Multiplication

$$\begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix} \times \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} = \begin{bmatrix} a_1a_2 + b_1c_2 & a_1b_2 + b_1d_2 \\ c_1a_2 + d_1c_2 & c_1b_2 + d_1d_2 \end{bmatrix}$$

ונקבל מטריצה בגודל של הטבלא המקורית עם ערכים חדשים. את הטבלא הזאת נשווא לטבלאה המקורית, על ידי שבעבור כל ערך שאינו חסר (לא אפס) נבצע חישוב על פי נוסחת פונקציית ההפסד שבחרנו
במקרה שלנו היא תיהיה RMSE או Root-mean-square deviation

$$\hat{r}_{mu} = p_m q_u^T = \sum_{n=1}^{n} p_{mk} q_{uk}$$

$$e_{mu} = r_{mu} - \hat{r}_{mu} = r_{mu} - \sum_{n=1}^{n} p_{mk} q_{uk}$$

$$e_{mu}^2 = (r_{mu} - \hat{r}_{mu})^2 = \left(r_{mu} - \sum_{n=1}^{n} p_{mk} q_{uk}\right)^2$$

$$\frac{d}{dp_{mk}} e_{mu}^2 = -2(r_{mu} - \hat{r}_{mu})(q_{uk}) = -2e_{mu} q_{uk}$$

הנוסחא הראשונה היא נוסחא לחישוב דירוג חזוי של המטריצה R כובע
החישוב הוא הכפלה רגילה של מטריצות שנראת ככה

2 × 2 Matrix Multiplication

cuemath
THE MATH EXPERT

$$\begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix} \times \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} = \begin{bmatrix} a_1a_2 + b_1c_2 & a_1b_2 + b_1d_2 \\ c_1a_2 + d_1c_2 & c_1b_2 + d_1d_2 \end{bmatrix}$$

הנוסחא השנייה היא חישוב הטעות (ERROR) על ידי החסרה של כל ערך חזוי בערך המצוי שלו במטריצה המקורית

לאחר מכן נעלה אותה בריבוע כדי להתעלם מערכים שלילים (הנוסחה השלישית)

עכשיו נגזור את הפונקציה emu^2 כדי לגלות כמה ולאיזה כיוון אנחנו צריכים להזיז את המשקולות, כל ערך נצטרך לגזור פעמיים. פעם בשביל Q ופעם בשביל P

זה אומר שעבור כל משקולת הנוסחה תהיה

$$p'_{mk} = p_{mk} + 2e_{mu} q_{uk} = p_{mk} + 2\alpha e_{mu} q_{uk}$$

$$q'_{uk} = q_{uk} + 2e_{mu} p_{mk} = q_{uk} + 2e_{mu} p_{mk} \alpha$$

עבור כל אחת אנחנו נהפוך את הסימן כדי שנתקדם בכיוון הרצוי (אם זה יותר נרצה להחסיר ואם זה פחות נרצה להגדיל)
עכשיו את המספר הזה נכפיל בLERNING RATE
LERNING RATE הוא ערך שניקבע על ידי המתכנת, הוא ערך קטן מאוד (בדרך כלל לא גדול מ 0.2) והוא קובע את גודל הצעדים שהאלגוריתם יקח כל פעם שהוא מכוון את המשקולות, זה נעשה כדי שהצעדים יהיו קטנים ולא נפספס את המינימום

וזהו, עכשיו המערכת תחזור על הצעדים הללו עד שהERROR יגיע לערך מינימאלי
התוצאה הסופית תיהיה טבלא בגודל הטבלא המקורית R שבא כל הערכים מלאים. עכשיו כל מה שנותר זה עבור כל משתמש לקחת את המוצרים עם הערכים שהדירוג שלהם עולה על המינימום שאנחנו הצבנו (או את הגבוהים ביותר) . ולסנן משם את המוצרים שהוא כבר רכש (או במקרה שלנו משחקים שהוא כבר שיחק) ולהציע אותם למשתמש

הקוד

```
// intililze two random matrices in the appropriate size

        double[][] userMatrix = initializeMatrix(matrix[0].length, numFactors, minValue, maxValue);

        double[][] itemMatrix = initializeMatrix(numFactors, matrix.length, minValue, maxValue);

        double prevRMSE = Double.MAX_VALUE;

        int errorCount=0;

        // set error to max value so that it will enter the loop on first go

        double error = Double.MAX_VALUE;

        List<Double> doubleList = new ArrayList<>();

        // loop over the original matrix times that are specfied in the functions

        // parameters(aka steps)

        for (int step = 0; step < steps; step++) {

            // debug: print the number of step being done

loop over each cell in the matrix R (og matrix)

            for (int i = 0; i < matrix.length; i++) {

                for (int j = 0; j < matrix[i].length; j++) {

                    // go over all non zero enteris

                    // if there is an entery in matrix R aka if user has played that game

                    if (matrix[i][j] != 0) {

                        // get the prediction of the currnt cell


                        double dotProduct = dotProduct(getRowOfMtrix(userMatrix, i), getColomnOFMatrix(itemMatrix, j));

                        // calculate the mean squared error of the prediction reletive to the actual

                        // value

                        error = Math.pow(matrix[i][j] - dotProduct, 2);

                        // for each k value in the matrices U and V

                        for (int k = 0; k < numFactors; k++) {

                            // update the k values in the matrices U and V according to the error


                            double Ustep = -2 * ((matrix[i][j] - dotProduct) * (itemMatrix[i][k]));

                            Ustep = flipSign(Ustep);

                            userMatrix[i][k] += Ustep * learningRate;
```

```
                                double Istep = -2 * ((matrix[i][j] - dotProduct) * (userMatrix[k][j]));
                            Istep = flipSign(Istep);
                            itemMatrix[j][k] += Istep * learningRate;
                        //   System.out.println("the steps taken are: " + Istep * learningRate + " and " + learningRate *
Ustep);
                        }

                    }
                }

            }


            double prediction;
            double sumSquaredErrors = 0;

            for (int x = 0; x < matrix.length; x++) {
                for (int y = 0; y < matrix[0].length; y++) {
                    if (matrix[x][y] != 0) {
                        errorCount++;
                        prediction = dotProduct(getColomnOFMatrix(userMatrix, x), getRowOfMtrix(itemMatrix, y));
                        double e = matrix[x][y]-prediction;
                        sumSquaredErrors += e*e;

                    }
                }
            }
            double rmse = Math.sqrt(sumSquaredErrors / errorCount);
           //cheack for convergence of the error
             if (step > 0) {
                rmse = Math.sqrt(sumSquaredErrors / errorCount);
                if (rmse > prevRMSE) {
                    break;
                }
                prevRMSE = rmse;
            }

        }
```

חישוב יעילות

יעילות האלגוריתם היא $O(s((m \times n \times)k))$

כאשר

S -מייצג את כמות הצעדים שהותרו למערכת

m - מייצג את מספר השורות במטריצה המקורית

n – מייצג את מספר העמודות במטריצה המקורית

K – מייצג את מספר הפיצ'רים שנבחר על ידי המתכנת

מבנה נתונים בשימוש

המבנה נתונים העיקרי בפרוייקט הוא מטריצות מסוג double מבנה זה משמש לאחסון טבלאת האינטרקציות של המשתמשים

נעשה שימוש גם ברשימות בעיקר על מנת לאכסן משחקים

הקשרים בין היחידות השונות

תיאור התוכנה

סביבת העבודה ששומש ליציר הפרוייקט היא visual studio code

Visual studio code

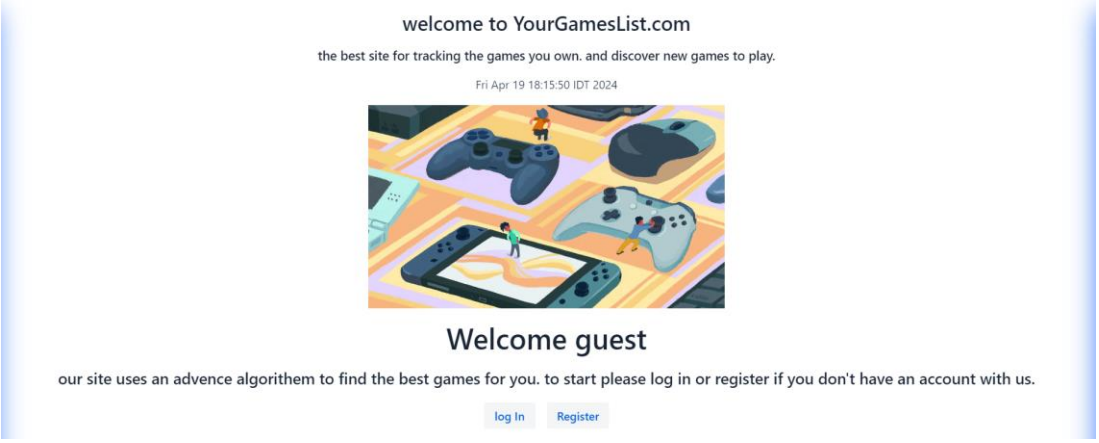
היא סביבת עבודה מודרנית לפיתוח אפליקציות. הוא עובד בעזרת הרחבות (extensions) כך שהשימוש בו מאפשר התאמה של התוכנה לצרכים הספציפיים של המתכנת

שפת התכנות העיקרית של שפרוייקט היא JAVA והיא מרכיבה את רובו. בנוסף נעשה שימוש קל בשפת PYTHON לעיבוד ראשוני של מידע ממאגרי הנתונים ששומש לבדיקת האלגוריתם

תיאור מסכים

עמוד הבית\דף הנחיתה

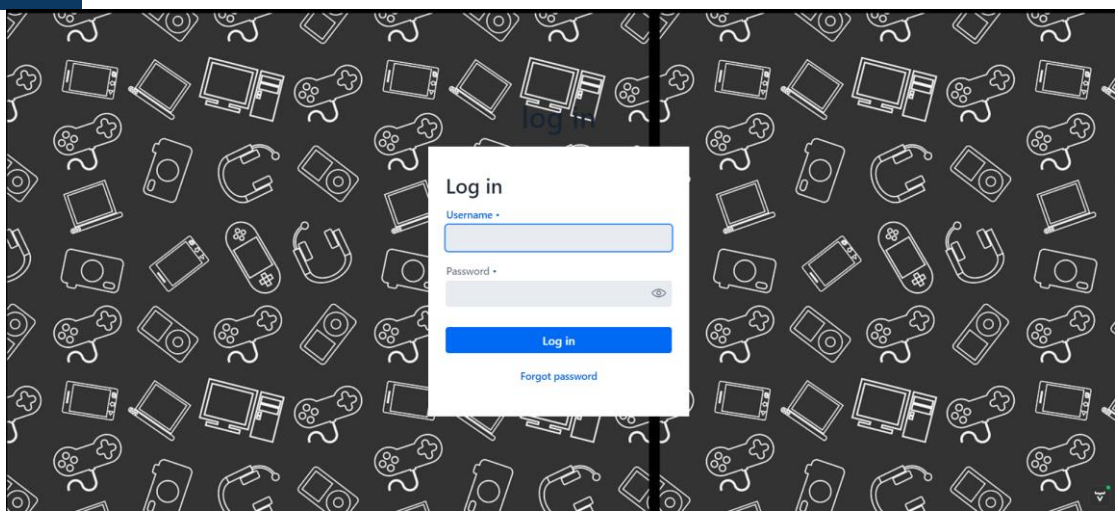
דף זה נמצא ב root של הכתובת והוא כנראה הדף הראשון שבו המשתמש יתקל בדף הסבר קצר על האתר מזמינה את המשתמש להכנס לחשבון קיים או ליצור חשבון חדש



דף ההתחברות\loginPage

Route -/login

דף זה מאפשר למשתמש להתחבר אל החשבון האישי שלו, לאחר ההתחברות המשתמש ינווט אל דף החשבון שלו



עמוד ההירשמות\rgistering page

כאן משתמשים חדשים שאינם חברים באתר יוכלו ליצור חשבון באתר כדי להתחיל להשתמש בו(שימוש באתר מחייב התחברות)