עבודת גמר בתכנון ותכנות מערכות  
התמחות בהגנת סייבר  
סמל מקצוע: 883589



**משחק רב משתתפים נחש את הציור**

Multiplayer Drawing And Guessing Game

שם התלמיד: רון אייזן

תעודת זהות: 213412422

בית ספר: ראשון לציון, מקיף ה' אמירים.

שם המנחה: מוטי מתתיהו

מועד הגשה: תשפ"א 2021

**תוכן עניינים**

[מסמך ייזום 4](#_Toc73170990)

[מבוא ורקע כללי לנושא העבודה 5](#_Toc73170991)

[נושא הפרויקט 5](#_Toc73170992)

[תקציר הפרויקט 5](#_Toc73170993)

[הגדרות 5](#_Toc73170994)

[מטרת המערכת ויעדיה 5](#_Toc73170995)

[ארכיטקטורת המערכת 6](#_Toc73170996)

[הגדרת רכיבי המערכת 6](#_Toc73170997)

[תיחום המערכת 7](#_Toc73170998)

[סיקור מצב השוק כיום: 7](#_Toc73170999)

[אילוצים ודרישות: 7](#_Toc73171000)

[פירוט סביבת העבודה והכלים הנדרשים לפיתוח: 9](#_Toc73171001)

[שימוש במודולים: 10](#_Toc73171002)

[תיאור המוצר המוגמר: 11](#_Toc73171003)

[ניהול פרויקט עתידי: 12](#_Toc73171004)

[מסמך אפיון 13](#_Toc73171005)

[פונקציונליות המערכת 14](#_Toc73171006)

[הערכת המוצר לעומת התכנון 17](#_Toc73171007)

[פיתוחים לעתיד 17](#_Toc73171008)

[אילוצים ודרישות המערכת 17](#_Toc73171009)

[סביבת עבודה 17](#_Toc73171010)

[מסמך עיצוב 18](#_Toc73171011)

[תיאור בסיס מערכת הנתונים במערכת 19](#_Toc73171012)

[שדות בסיס הנתונים 19](#_Toc73171013)

[הסבר השדות 19](#_Toc73171014)

[מודולים 20](#_Toc73171015)

[Server 20](#_Toc73171016)

[Surface 21](#_Toc73171017)

[Screen 22](#_Toc73171018)

[22](#_Toc73171019)

[24](#_Toc73171020)

[אלגוריתם המערכת 24](#_Toc73171021)

[הגדרת התקשורת (RFC) 26](#_Toc73171022)

[תיאור ממשק המשתמש 27](#_Toc73171023)

[MVP: 32](#_Toc73171024)

[קטעי קוד מיוחדים 33](#_Toc73171025)

[יומן רפלקציה 34](#_Toc73171026)

[בחירת GUI 35](#_Toc73171027)

[וובליוגרפיה 37](#_Toc73171028)

[נספחים 40](#_Toc73171029)

[קישור לMVP שיצרתי: 41](#_Toc73171030)

[קישור לקבצי הפרויקט בגיט-האב: 41](#_Toc73171031)

[מילון מונחים 42](#_Toc73171032)

­­­­

מסמך ייזום

מבוא ורקע כללי לנושא העבודה

בחרתי לעשות את הפרויקט הזה מפני שהוא מאתגר ומעניין. בנוסף, בסוף הפרויקט יוצא מוצר שמיש וייחודי שגם מעניין מבחינת הנושא שלו. את הרעיון לפרויקט קיבלתי מהאפליקציה skribbl.io שהיא אפליקציה אינטרנטית חינמית בעלת רעיון דומה לרעיון עליו מתבסס הפרויקט שלי. אשתף כי תחילה בחרתי ברעיון "עוזרת אישית קולית", אך לאחר מכן התברר כי רעיון זה נתפס על ידי תלמיד אחר בכיתה, הרעיון החלופי לפרויקט בא מהמורה שלי (מוטי).

נושא הפרויקט

משחק ניחוש באמצעות ציור, רב משתתפים.

תקציר הפרויקט

המערכת תותקן על ארבעה מחשבים המקושרים אחד לשני ברשת המקומית (LAN), מחשב אחד ישמש כשרת והאחרים כלקוחות כאשר אחד ייבחר כצייר והשניים האחרים כמנחשים. יש שני סוגים שונים של לקוח (צייר ומנחש). כל הלקוחות מקושרים ביניהם דרך השרת. המשחק מכיל שלושה סיבובים, בכל סיבוב נבחר צייר אחר מבין שלושת המשתתפים.

הגדרות

משתמשי המערכת

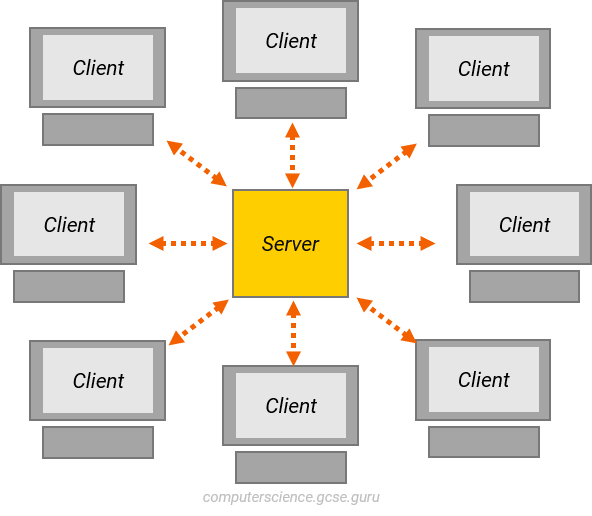
המערכת מיודעת לכל אדם עם מחשב המעוניין לשחק במשחק ניחוש תחרותי ומהנה.

מטרת המערכת ויעדיה

מטרת המערכת היא לתת למשתמש חווית משחק נוחה תחרותית ומהנה, שליטה ובקרה נוחה. במשחק זה רוכז רעיון מהנה וייחודי, ונעטף בממשק גרפי נוח, ברור וקל לתפעול. המשחק הוא משחק רב משתתפים על ניקוד ולכן מעודד תחרותיות בין המשתתפים. מסייע בשיפור יכולת הציור ובוחן אותה.

ארכיטקטורת המערכת

המערכת נבנתה בארכיטקטורת שרת-לקוח. המערכת תומכת במספר רב של לקוחות, תוך תמיכה בריבוי תהליכונים במקביל- המערכת בנויה כך שלכל לקוח נפתח תהליכון (thread) חדש בשרת, ובכל תהליכון נפתח תהליכון נוסף בהתאם למצב הלקוח (צייר/ מנחש) המטפל בשליחות שונות הלקוח לשרת ומהשרת ללקוח. כמו כן, גם בצד הלקוח נעשה שימוש בתהליכונים, בכל לקוח יש שימוש בשני תהליכונים, אחד לשליחת/ קבלת קואורדינטות לציור על מסך והשני לטיימר של שמונים שניות.

**תרשים מספר 1: ארכיטקטורת המערכת – שרת לקוח**

מקור התמונה: <https://www.computerscience.gcse.guru/theory/client-server-networks>

הגדרת רכיבי המערכת

1. רשת מקומית- (Network Area Local- LAN) רשת מקומית. רשת מחשבים המתפרסת על אזור קטן ומוגבל, לרוב בתוך בניין.
2. שרת- (server) משמש לקישור בין כל הלקוחות, מטפל בבקשות ובשליחות של כל הלקוחות.
3. לקוח- (client) מחשב של שחקן המחובר למערכת, משמש להתחברות השחקן והרשמתו במערכת ואחראי על כל הממשק הגרפי.

תיחום המערכת

הפרויקט עוסק ומשלב בתוכו תחומים רבים וביניהם:

1. תקשורת- נעשית תקשורת בין מספר מחשבים (4) שרת ו-3 לקוחות (כמספר המשתתפים).  
2. בסיס נתונים- נעשה שימוש בבסיסי נתונים למאגר שמות השחקנים במשחק.  
3. אבטחת מידע- הצפנת מידע רגיש של השחקן במשחק, לדוגמה סיסמת השחקן.  
4.חלק ויזואלי (GUI)- יצירת ממשק גרפי בעל מקשים נוח וידידותי למשתמש.

המודולים ותתי הנושאים בהם אני משתמש בפרויקט:

1. בתחום התקשורת נעשה שימוש במודולים socket וthreads כשאר שניהם מאפשרים יחד תקשורת בין שרת למספר לקוחות.
2. בתחום בסיס הנתונים נעשה שימוש במסד הנתונים Sqlite3 ו-DBbrowser להצגתו.
3. יצירת הממשק הגרפי תעשה על ידי שימוש במודול tkinter.

סיקור מצב השוק כיום:

כיום ישנם מספר משחקים בעלי רעיון דומה. את ההשראה לפרויקט שאבתי מאפליקציה בשם SKRIBBL.  
ישנן וריאציות שונות למשחק (חלקן ברשת וחלקן מחוצה לה) ובמרבית האפליקציות ניתן לשחק עם אנשים שונים מכל העולם.

אילוצים ודרישות:

1. על המערכת לעבוד על מחשב שמערכת ההפעלה שלו היא WINDOWS.
2. לוח זמנים צפוף, מתכוון לבצע שלושה מועדי פסיכומטרי בעת תקופת ביצוע הפרויקט.
3. קהל היעד הוא עבור ילדים בגילאים 8-18.

פירוט סביבת העבודה והכלים הנדרשים לפיתוח:

שפת התכנות: Python 3.7.3

סביבת העבודה: WINDOWS. ו- PYCHARM

התקנת המערכת:

המערכת תותקן ברשת המקומית (LAN). לקיום המשחק יש צורך בארבעה מחשבים (שרת ושלושה לקוחות). אין צורך בהתקנה של מודולים מיוחדים, יש להתקין את השרת במחשב ובמחשבים האחרים להתקין את הלקוחות, יש להריץ את השרת ולפני למצוא את ה-IP של המחשב ע"י שימוש בפקודה ipconfig ב-cmd, כמו כן יש לבדוק ש-Port 1730 לא חסום על ידי חומת האש של מערכת ההפעלה.

כלים הנדרשים לפיתוח:

1. ארבעה מחשבים.
2. Python 3.7.3.
3. Windows 10.

כלים הנדרשים לעריכה:

1. PyCharm.
2. DBbrowser For Sqlite.

הכלים הנדרשים לבדיקה וסיוע בפיתוח:

1. מספר מחשבים על מנת לבדוק את תחום התקשורת.
2. תיעוד והסבר בקוד מה מקבלת כל פונקציה או כל אובייקט ומתי הלקוח שולח מידע לשרת ומתי השרת אל הלקוח וכו'...
3. שימוש בתכנות מונחה עצמים (OOP) על מנת לרווח את הקוד.
4. שימוש ב-PyCharm למטרת דיבוג.

שימוש במודולים:

1. בתחום התקשורת נעשה שימוש במודולים socket ו- threading כשאר שניהם מאפשרים יחד תקשורת בין שרת למספר לקוחות.
2. בתחום בסיס הנתונים נעשה שימוש במודול sqlite3.
3. אימות דו שלבי, הצפנת המידע תעשה באמצעות פונקציית Hash, על ידי שימוש במודול hashlib.
4. יצירת הממשק הגרפי על ידי שימוש במודול tkinter.

תיאור המוצר המוגמר:

שם המוצר: Draw & Guess.  
מה המוצר המוגמר אמור לבצע: על המוצר המוגמר להיות משחק רב משתתפים של 3 שחקנים, שבעבור כל משתתף מוצג לוח הציור של אותו השחקן שבפניו מוצגת המילה.  
מטרות מרכזיות:

1. חשיפה למודולים חדשים בשפת פייתון.
2. התנסות ראשונית ביצירת פרויקט בהיקף של חמש יח"ל.
3. העמקה והרחבת הידע בתחום הסייבר והצפנת המידע.
4. התנסות בתכנות עם בסיסי נתונים.
5. יצירת ממשק גרפי (GUI).

המערכת עונה על **כל** המטרות המרכזיות שציינתי וניתן לראות זאת בהרחבה בחלק תיחום הפרויקט.

ניהול פרויקט עתידי:

**תרשים מספר 2: טבלת ניהול פרויקט עתידי**

|  |  |
| --- | --- |
| יעד | תאריך יעד |
| קישור בין שרת ולקוח ויצירת בסיס נתונים מלא עם מילים, ובסיס נתונים ריק של שחקנים. | 10.11.20 |
| קישור בין בסיסי הנתונים לשרת.  יצירת פעולה שמקבלת את המילה מהשרת | 30.11.20 |
| פעולה ששולחת את הניחוש לשרת | 15.12.20 |
| הוספת ממשק גרפי פעולה שבאמצעותה מצייר הלקוח את המילה שקיבל. | 15.1.21 |
| שיפורים/ תיקונים אחרונים | 1.4.21 |

מסמך אפיון

פונקציונליות המערכת

פעולות רקע:

1 .חיבור כל המשתמשים לשרת.

2 .שמירת נתוני המשתמשים.

3 .פתיחת מסך הפתיחה.

פעולות השרת:

1. שידור מסכים בזמן אמת.
2. שידור הציור לכלל המשתמשים.
3. שולח לשחקנים אם המשחק הסתיים.
4. חישוב ושליחת הניקוד לכל שחקן ולצייר.
5. הגרלת המילה ממאגר הנתונים ושליחתה.

פעולת הלקוח:

1. התחברות לשרת.
2. שליחה לשרת האם הצליח לנחש את המילה או לא.
3. הרשמה ראשונית למשחק.
4. כניסה למשחק עם משתמש קיים (log in).
5. בדיקה האם הניחוש נכון.
6. טיימר.
7. האם הסיבוב הסתיים או לא.

דרישות פונקציונליות :

**תרשים מספר 3: דרישות פונקציונליות**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | דרישה | תהליך | קלט | פלט | טיפול בשגיאות |
| 1 | התחברות לserver | המשתמש תחבר לserver של המערכת על ידי socket. | Portו IP של ה server | התחברת למערכת | הודעת שגיאה : נסה להתחבר למערכת מחדש. |
| 2 | הרשמה ראשונית למערכת | המשתמש יכניס את פרטיו יבצע בדיקת סיסמא ולאחר מכן פרטיו ישמרו בשדה הנתונים . | פרטי המשתמש (שם משתמש, סיסמא, אמייל) | הלקוח נרשם למערכת בהצלחה. | הודעה שגיאה :  - אמייל או שם משתמש שגוי, סיסמה קצרה מדי. |
| 3 | התחברות למשתמש קיים | המשתמש יכניס את שם המשתמש וסיסמתו אשר יבדקו בשדה הנתונים ולאחר מכן יכנס למערכת . | שם המשתמש והסיסמא של המשתמש. | התחברת למערכת בהצלחה. | במקרה שהמשתמש יכניס פרטים שגויים הוא יתבקש לחזור על הפעולה |
| 4 | טיימר של 80 שניות | יצירת THREAD חדש שבו ירוץ הטיימר במשך 80 שניות, מכיוון שהטיימר בתהליכון משלו גם יהיה מעבר בין מסכים או שליחות בין השרת ללקוח הטיימר ימשיך לעבוד. | זמן | טיימר על מסך המשתמש. | שימוש בTRY וEXCEPT על מנת למנוע קריסות במהלך ההרצה. |
| 5 | שליחת מילה ממאגר הנתונים ללקוח | השרת מגריל מילה ממאגר הנתונים, ושולח אותה ללקוח באמצעות socket | - | המילה המוגרלת | - |
| 6 | בדיקות הניחוש בלקוח | הלקוח בודק האם הלקוח ניחש נכונה. | הניחוש | אמת או שקר. | - |
| 7 | שידור מסכים בזמן אמת. | יצירת תהליכון שמקבל קאורדינטות מהשרת ומצייר בהן על מסך המשתמש. | קאורדינטות מופרדות ב';' מכילות X וY. | ציור על המסך. | שימוש בTRY וEXCEPT על מנת למנוע קריסות במהלך ההרצה. |
| 8 | האם יש צורך לסיים את המשחק. | אני בודק בשרת רם מספר המשתמשים שניחשו כבר או סיימו את ניחושיהם ולא הצליחו לנחש את המילה שווה למספר השחקנים פחות אחד (הצייר) ובמידה וכן השרת שולח ללקוחות 'END' והלקוח מבין שהוא צריך לעצור את הסיבוב ולהתחיל את הסיבוב הבא, או לחליפין במידה שנגמר הזמן. | - | שליחת END. | - |
| 9 | הגרלת המילה ממאגר הנתונים ושליחתה. | בשרת יש רשימת מילים מכל מיני סוגים, באמצעות המודול RANDOM השרת מגריל מילה מן המאגר ושלח אותה לשחקן בצירוף עם תפקידו בסיבוב הנוכחי, נחשן (GUESS) או צייר (DRAW). | - | המילה בסיפוח התפקיד של השחקן | - |
| 10 | 1. חישוב ושליחת הניקוד לכל שחקן ולצייר. | בעת ניחוש המילה הלקוח שולח לשרת כי השחקן ניחש נכונה והשחקן בתגובה מקבל את הניקוד שקיבל, השחקן נרשם ברשימת האנשים שניחשו את המילה, הניקוד המוענק לו מחושב עפ"י מיקומו ברשימה, אם ניחש ראשון יקבל ניקוד גבוה מהאדם שניחש אחרון...  כמו כן גם הצייר מקבל 30 נקודות בעבור כל ניחוש של שחקן. | הודעה מהלקוח כי השחקן ניחש נכונה. | שליחת הודעה המכילה את הניקוד הניתן לו בצירוף עם המילה SCORE. | - |

הערכת המוצר לעומת התכנון

באופן כולל הצלחתי לממש את החזון ההתחלתי שהיה לי, מלבד כמה שינויים ואילוצים שונים למיניהם שהכריחו אותי לשנות את התכנון את ההתחלתי.

פיתוחים לעתיד

ישנם מספר אפיקים בהם הייתי רוצה לעבוד על מנת לפתח את המשחק:

* ממשק גרפי: הייתי רוצה להפוך את הממשק הגרפי ליותר נוח וידידותי בעבור המשתמש, בכמה מובנים, שהציור יהיה יותר חלק וכך גם הצגתו בשאר המחשבים של שאר השחקנים. הוספת כפתורים שונים שיוכלו לסייע במקרים יוצאי דופן (משתמש מתנתק מהמשחק, השרת קורס וכדומה..)
* מאגר נתונים: להצפין את המידע במאגר הנתונים.
* מבחינת הקוד: הייתי רוצה למטב את הקוד אף יותר, להפוך אותו ליעיל יותר, קצר יותר ולארגן אותו טוב יותר למחלקות.

אילוצים ודרישות המערכת

בסיס הנתונים ימומש בשפת – SQL וישמרו בו נתוני המשתמש, ונתוני הקבצים ששותפו .

על המחשבים המחוברים למשחק להיות בעלי חיבור לאינטרנט וצריך שיהיה ביניהם חיבור.

סביבת עבודה

שפת פרויקט – . Python 3.7

מערכת הפעלה – WINDOWS.

ממשק גרפי – Tkinter ליצירת ה-GUI .

בסיס הנתונים – sqlite3 בשביל יצירת מאגר הנתונים. ו- DB Browser בשביל צפייה במאגר.

PyCharm סביבת עבודה - לכתיבת הקוד, Debugging , והרצה.

מסמך עיצוב

תיאור בסיס מערכת הנתונים במערכת

-Users.dbמאגר נתונים שמכיל את פרטי המשתמשים במשחק. השרת מתחבר אל מאגר הנתונים, מוסיף אליו משתתפים, בודק אם משתתפים קיימים, מקבל הרשאה להכנסת משתתפים רשומים כלומר יש לו גישה מלאה בכל הקשור למאגר הנתונים.

שדות בסיס הנתונים

**תרשים מספר 4: שדות בסיס הנתונים**

|  |  |
| --- | --- |
| Type | Field Name |
| TEXT | Username |
| TEXT | Password |
| TEXT | Email |

הסבר השדות

Username – שם המשתמש, מסוג טקסט, מפתח ראשי, חייב להיות ייחודי (כלומר לכל משתמש יש שם ואין כפילויות), לא יכול להיות ריק.

Password – סיסמת המשתמש, מסוג טקסט, לא חייבת להיות ייחודית (כלומר יכול לקרות מצב בו למספר משתמשים ישנה אותה הסיסמה), לא יכולה להיות ריקה.

Email – הדואר האלקטרוני של המשתמש, מסוג טקסט, חייב להיות ייחודי (כלומר לכל משתמש יש דוא"ל ואין כפילויות), לא יכול להיות ריק.

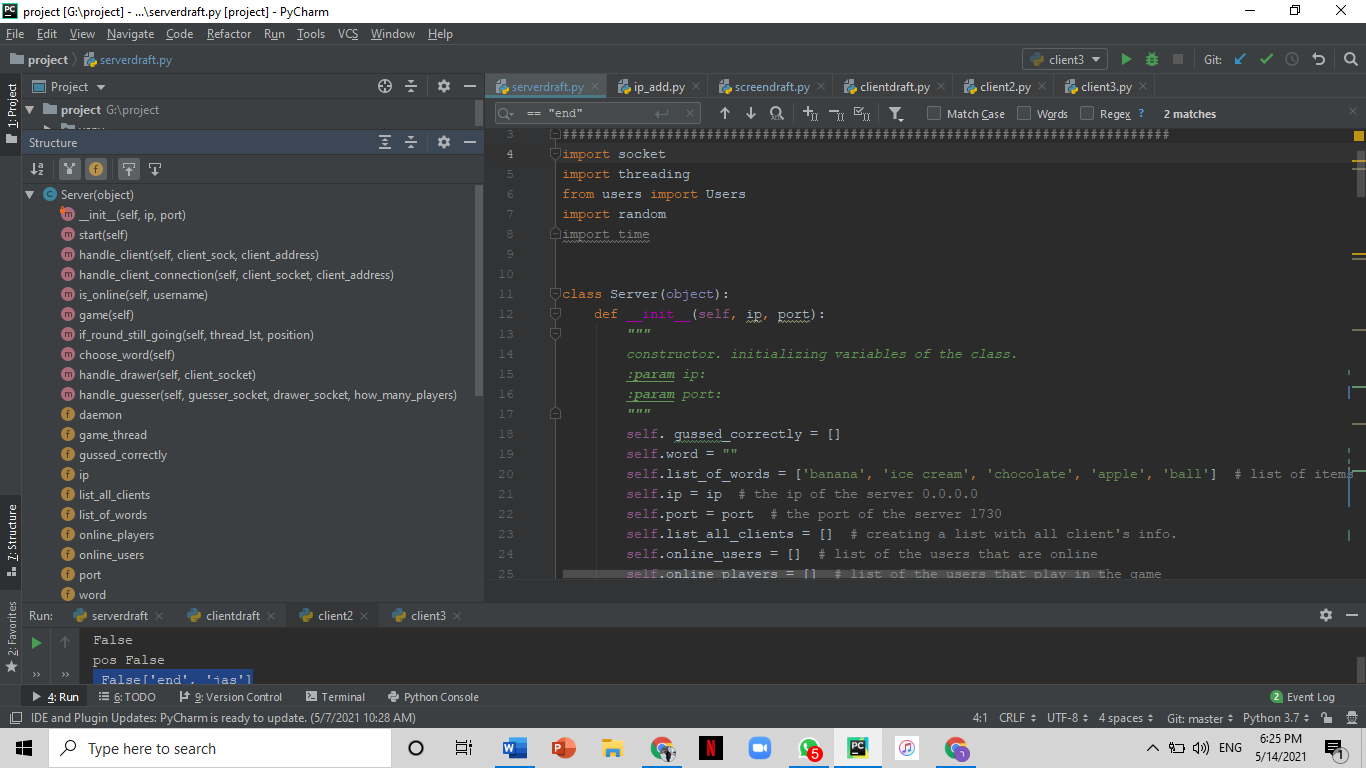
**תרשים מספר 5: דוגמה לבסיס נתונים**

|  |  |  |
| --- | --- | --- |
| Email | Password | username |
| [ronaizen@gmail.com](mailto:ronaizen@gmail.com) | 1234 | ron |
| [jasmin@gamil.com](mailto:jasmin@gamil.com) | 2704 | jas |
| [Moti@outlook.com](mailto:Moti@outlook.com) | Hello123 | Moti |
| [had@walla.com](mailto:had@walla.com) | 1234 | had |
| [hadas@yahoo.com](mailto:hadas@yahoo.com) | H2905 | Hadas |

מודולים

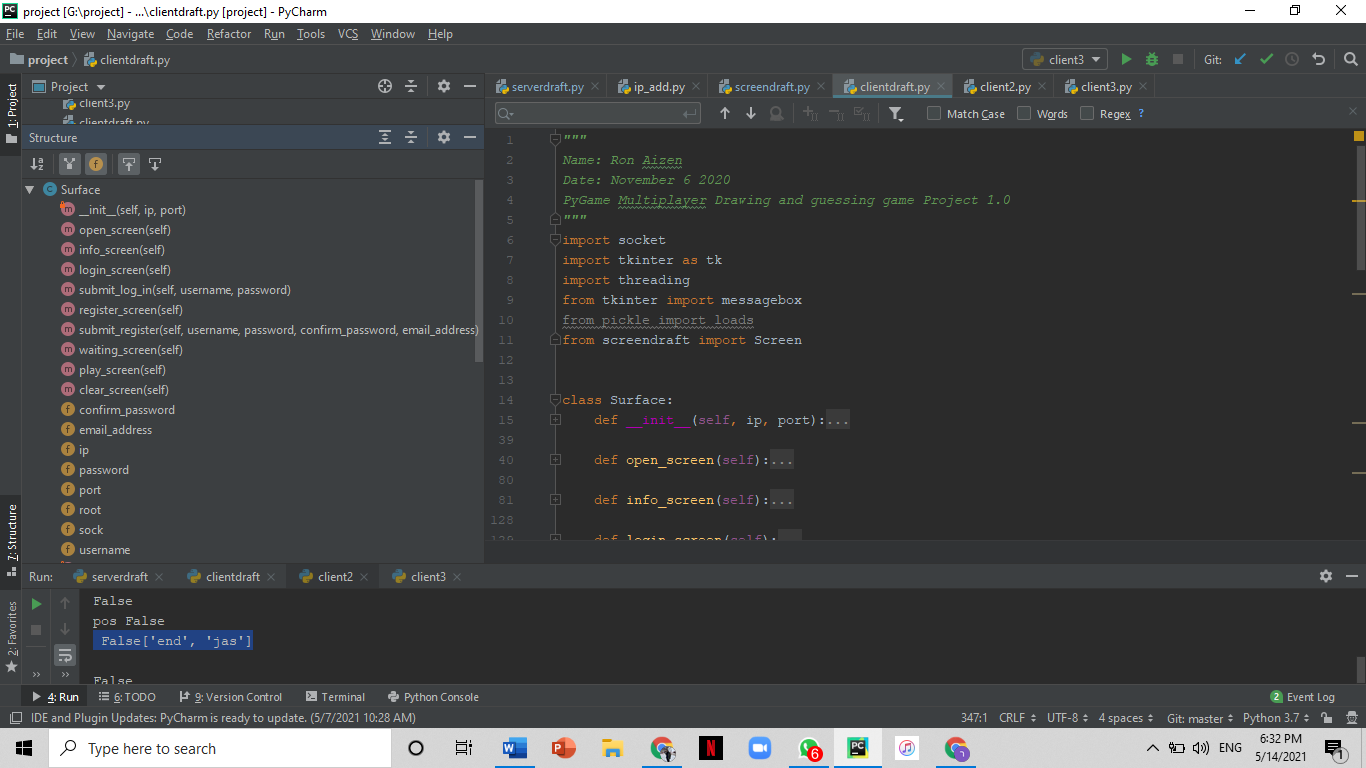
* serverdraft.py- הקובץ הראשי והעיקרי במחשב. כולל את השרת, את פעולותיו, את התקשורת שלו עם הלקוחות וכו'..
* clientdraft.py- הקובץ המכיל בתוכו את הממשק הגרפי ואת הלקוח. הקובץ מתחבר לשרת באמצעות חיבור socket.
* screendraft.py- הקובץ המכיל בתוכו את הקוד שיוצר את המסך עליו מצייר המשתמש.
* users.py- הקובץ שפונה אל מאגר הנתונים, מכניס משתמשים, שולף משתמשים וכו'...
* users.db- הקובץ שמכיל את מאגר המשתמשים.

Server



**תרשים מספר 6: המחלקה Server**

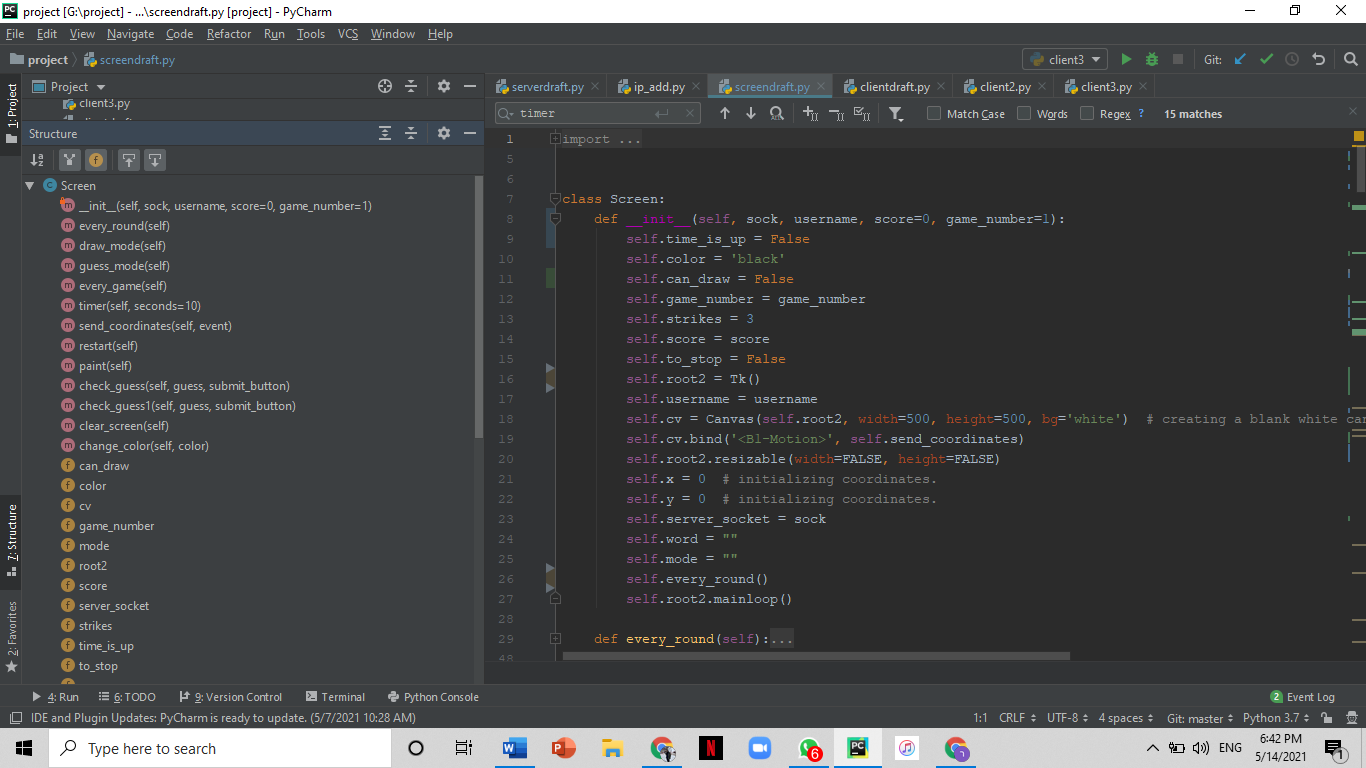
המחלקה server מייצגת, כמובן, את השרת והפעולות השונות שלו. מחלקה זו מכילה את  התכונות השונות של השרת, ואת האופן בו הוא מטפל בלקוחות המתחברים או מתנתקים  ממנו.

Surface

**תרשים מספר 7: המחלקה Surface**

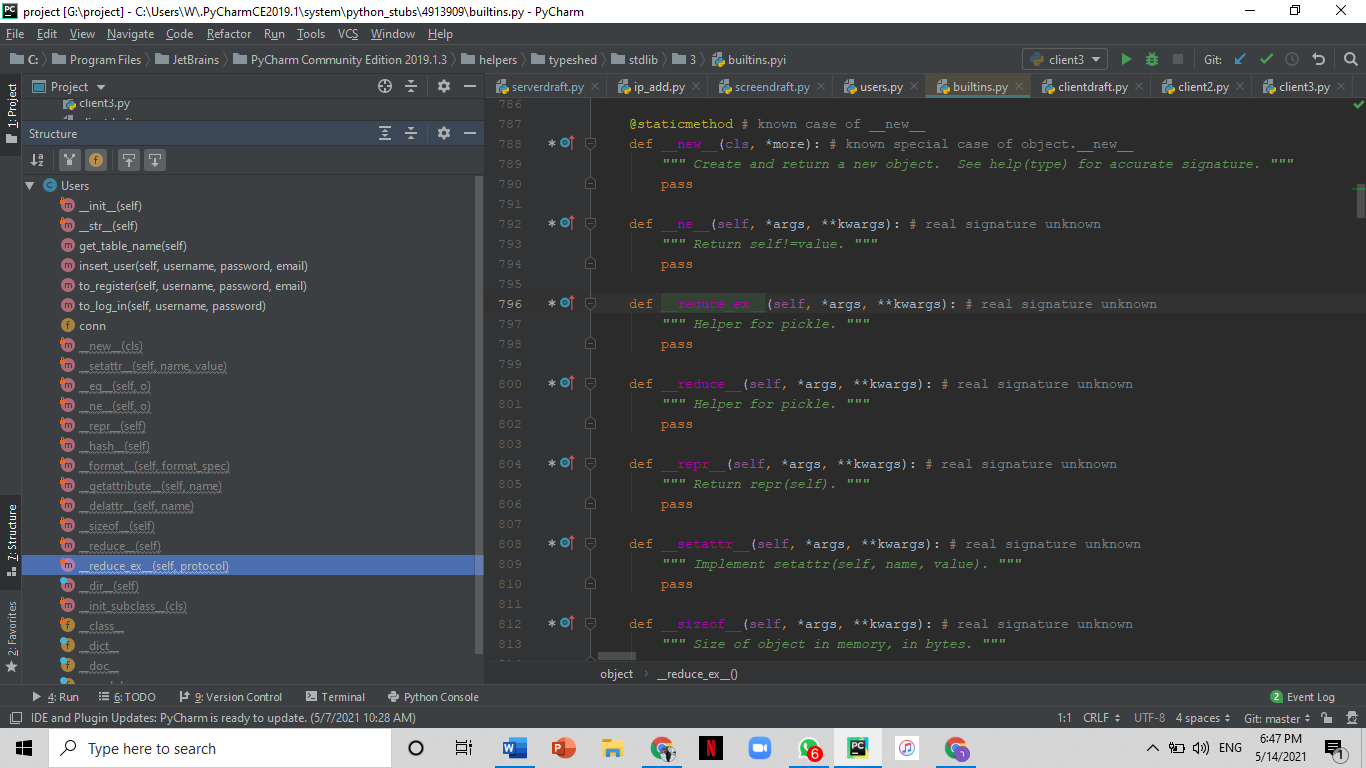
המחלקה Surface מייצגת את הלקוח, הממשק הגרפי והפעולות השונות בהם. מחלקה זו מכילה את התכונות השונות של הלקוח, ואת האופן בו הוא מטופל בעת התחברותו לשרת או בעת התנתקותו.

Screen



**תרשים מספר 8: המחלקה Screen**

המחלקה Screen מייצגת את המשך מחלקת Surface, בעוד שמחלקת Surface אחראית על ההתחברות הראשונית של השחקן לשרת ועל כל השלבים עד לשלב התחלת המשחק (מסך ראשי, התחברות/ הירשמות המשתמש ומסך ההמתנה עד להתחלת המשחק עצמו) מחלקת Screen אחראית על הממשק הגרפי ומסך המשחק עצמו (הקנבס עליו מצייר המשתמש) והפעולות השונות בהם. מחלקה זו מכילה את תכונות שונות של משתמש המשחק, ביניהן המסך עליו מציירים בעת המשחק, המילה שצריך לצייר באותו הסיבוב, הניקוד של אותו משתמש והאם הוא 'צייר' או 'נחשן' .

Users

**תרשים מספר 9: המחלקה Users**

המחלקה אחראית על הכנסת המשתמש למאגר המשתמשים, יצירת משתמש, וידוא הסיסמה המוכנסת בעת התחברותו וכל הקשור במאגר המשתמשים של המשחק.

Timeline

Description automatically generated

אלגוריתם המערכת

אלגוריתם השרת

**תרשים מספר 10: תרשים זרימה של אלגוריתם השרת**

אלגוריתם לקוח

Timeline

Description automatically generated

**תרשים מספר 11: תרשים זרימה של אלגוריתם הלקוח**

הגדרת התקשורת (RFC)

כל הפקודות הן מסוג String.

מבנה הפקודות הנשלחות בין השרת ללקוח

|  |  |
| --- | --- |
| **Parameter1** | **Opcode** |

**תרשים מספר 12: מבנה פקודת שרת-לקוח.**

כל אחד משדות הטבלה מופרד על ידי ; כך שהפקודה הסופית תיראה כך:

**"Opcode; parameter1"…**

הפקודות של שרת-לקוח

|  |  |  |
| --- | --- | --- |
| פקודה | Command | פרמטרים |
| שליחת ניקוד | Score | ניקוד |
| התחלת המשחק | Play | - |
| סיום המשחק | End | - |
| הרשמה | Register | שם משתמש, סיסמה ואימייל |
| התחברות | Login | שם משתמש וסיסמה |
| הקצאת צייר | Draw | - |
| הקצאת מנחשים | Guess | - |
| שליחת קואורדינטות | X; Y | - |

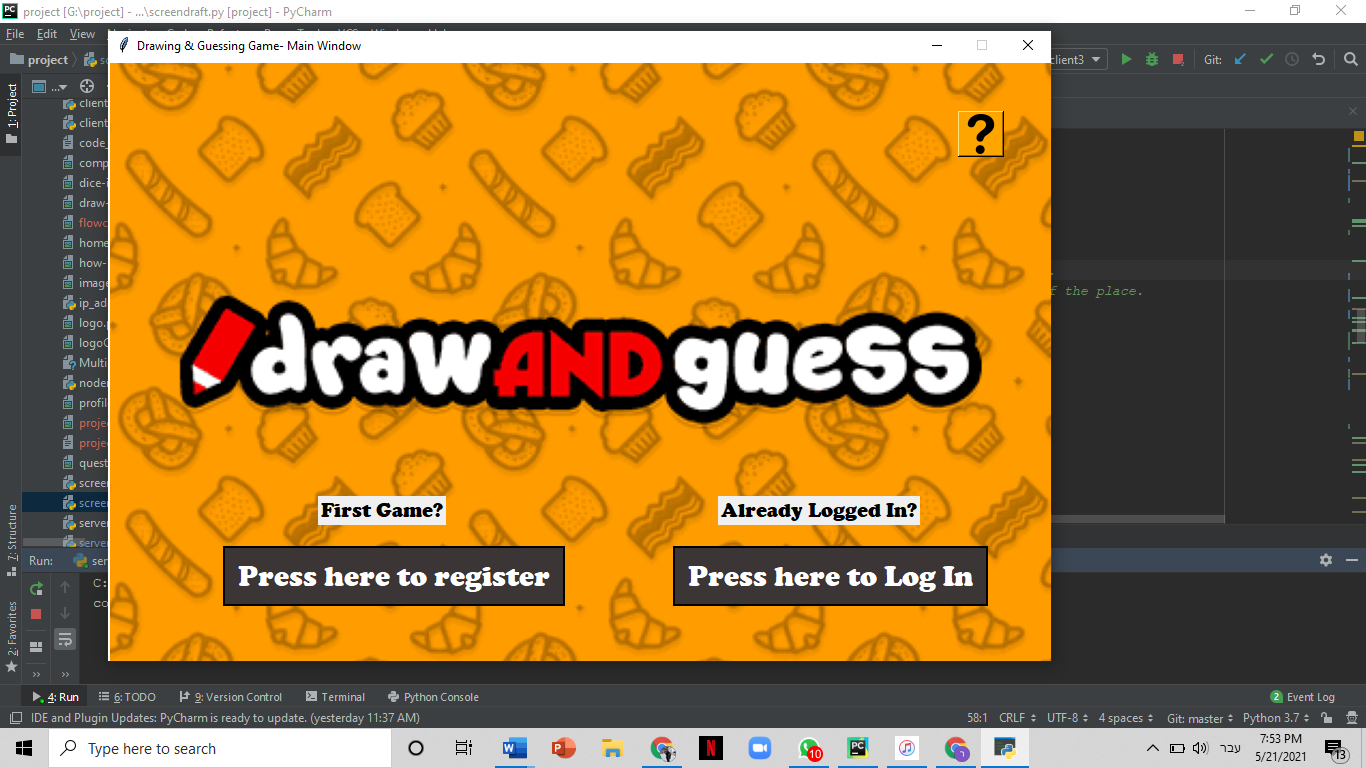
**תרשים מספר 13: רשימת פקודות שרת-לקוח.**

תיאור ממשק המשתמש

התקנה

יש להתקין את הקבצים על המחשב, ולהריץ את קובץ השרת 'Serverdraft.py' במחשב אחד ומשאר המחשבים להריץ את הלקוח 'Clientdraft.py'

ממשק המשתמש



1

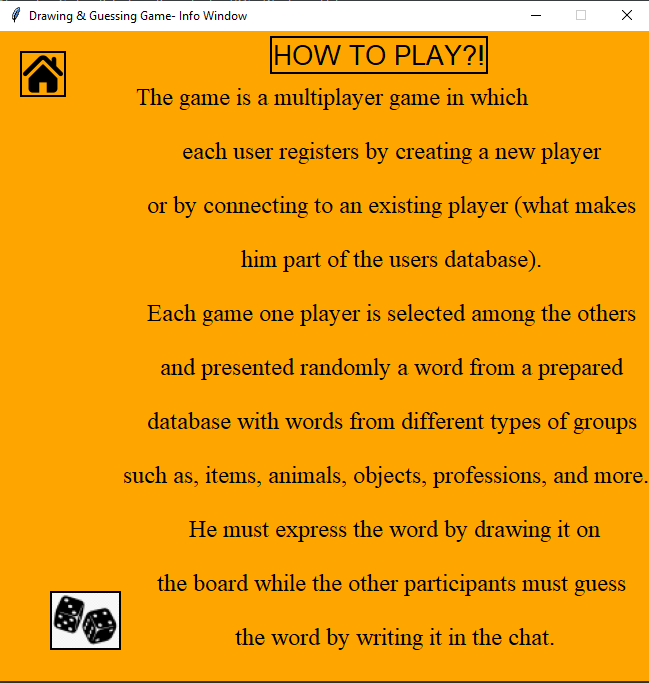
3

2

**תרשים מספר 14: תמונת מסך הפתיחה בממשק המשתמש.**

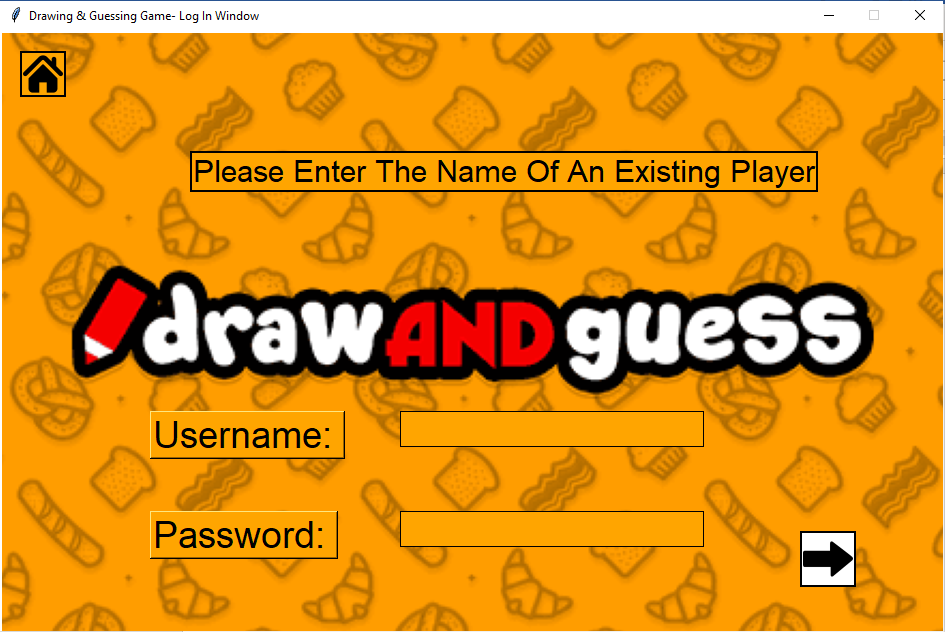
הכפתורים ופעולותיהם

1. כפתור הפותח מסך ובו מידע על המשחק והסבר כיצד פועל המשחק.
2. כפתור המעביר למסך הכניסה, לאנשים בעלי משתמש קיים.
3. כפתור המעביר למסך ההרשמה, לאנשים ללא משתמש.

****

לחצן בית, מחזיר את השחקן למסך הפתיחה

**תרשים מספר 15: תמונת מסך ההוראות בממשק המשתמש.**

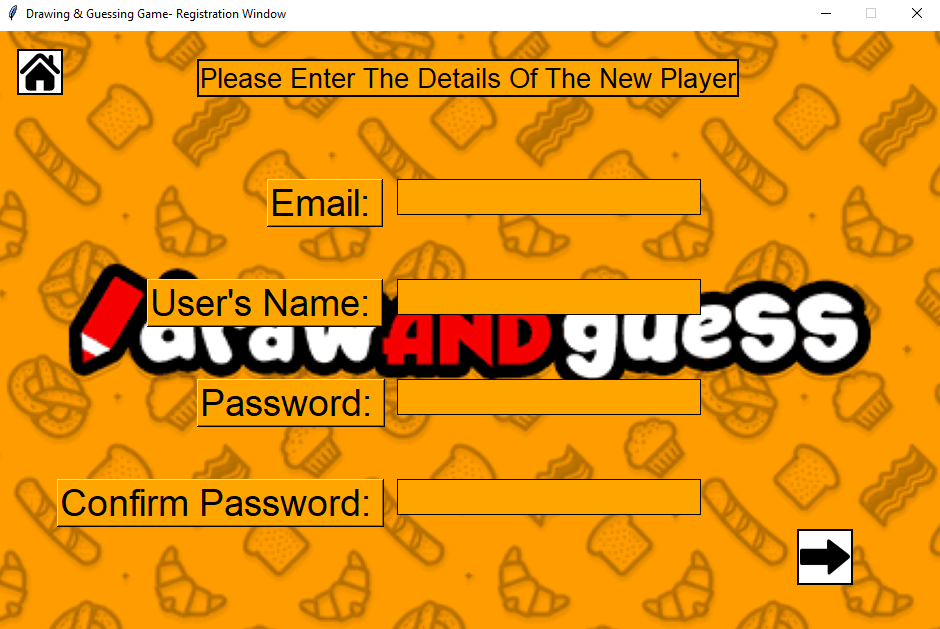


לחצן להכנסת הנתונים ואישורם

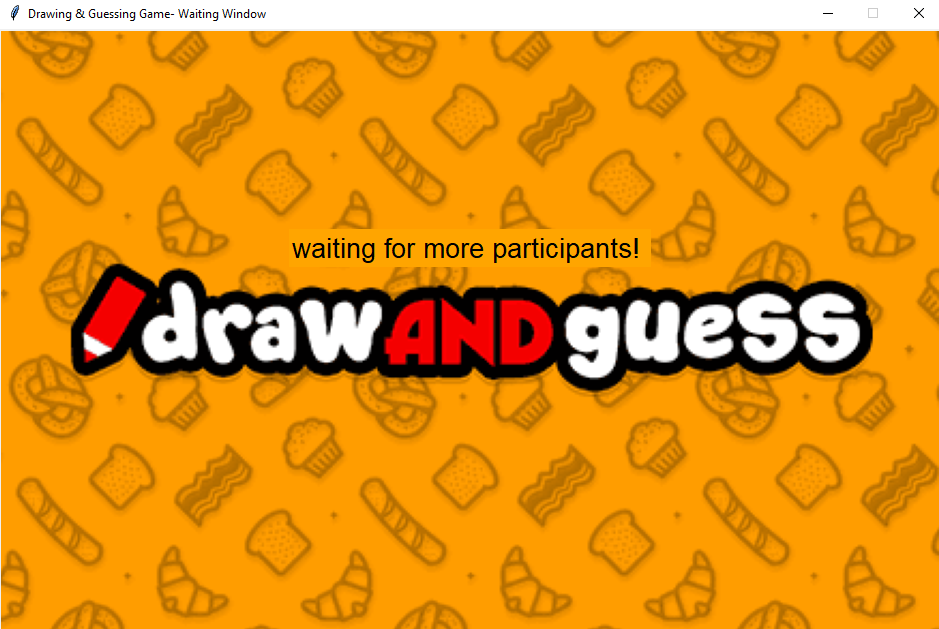
מקום להכנסת שם משתמש של שחקן קיים

**תרשים מספר 16: תמונת מסך ההתחברות בממשק המשתמש.**

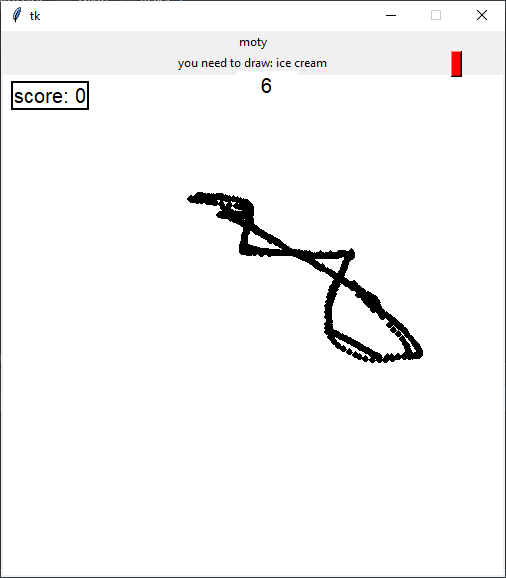
מקום להכנסת סיסמה של שחקן קיים



**תרשים מספר 17: תמונת מסך ההרשמה בממשק המשתמש**

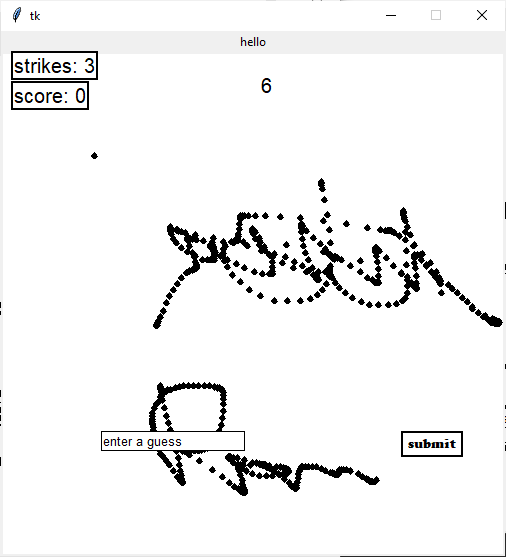


**תרשים מספר 18: תמונת מסך ההמתנה בממשק המשתמש**



**תרשים מספר 19: תמונת מסך הצייר בממשק המשתמש**

ניתן לראות את שם השחקן בראש המסך, את המילה אותה צריך לצייר מתחת (גלידה), שעון טיימר המורה על הזמן הנשאר לסיבוב (6 שניות) ואת הניקד שיש לו.



**תרשים מספר 20: תמונת מסך המנחש בממשק המשתמש**

ניתן לראות בראש המסך את שם השחקן, מתחתיו את הזמן הנותר לסיבוב, מצד שמאל יש את כמות הפסילות הנותרות (3) ומתחת יש את הניקוד. בתחתית המסך יש המקום להקליד את הניחוש ומימין את כפתור שליחת הניחוש במידה והניחוש שגוי תופיע הודעה בהתאם על המסך וכמות הפסילות הנותרות תקטן ב1, במידה ותגיע ל0 תיסגר האופציה של השחקן לנחש ותעלם התיבה להקלדת הניחוש בתחתית המסך. במידה והניחוש נכון יעודכן ניקוד השחקן בהתאם.

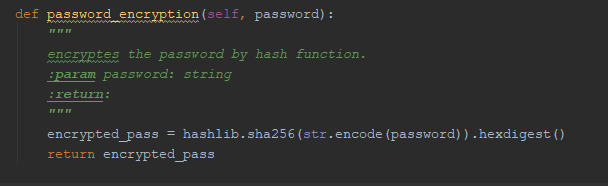
MVP:

הנחיות : לחצו פעמיים עד שהמצגת נפתחת במסך מלא. המצגת מתארת את התכנון הראשוני שלי לפרויקט. לאחר מכן לחצו ESC לחזרה לתיק הפרויקט.



קטעי קוד מיוחדים

שימוש בפונקציית Hash להצפנה:



**תרשים מספר 15: קוד הצפנה באמצעות פונקציית Hash.**

הפעולה מקבלת את הסיסמה של השחקן, בין אם בעת כניסה למשתמש קיים ובין אם בעת ההרשמה כשחקן חדש ומצפינה את הסיסמה לפי פונקציית Hash נעשה שימוש במודול hashlib.  
הסיסמה נשמרת במאגר הנתונים כמוצפנת על מנת למנוע חשיפה של מידע רגיש, לא ניתן לפענח את ההצפנה אלא רק להשוות בין מידע הנכנס לא מוצפן ולהצפינו ולקוות להתאמה.

שימוש בbind על מנת להשיג את הקואורדינטות של העכבר בעת לחיצה



**תרשים מספר 16: שימוש במתודת bind.**

בעת השימוש במתודה bind עם הערך '<B1-Motion>' בכל פעם שתתבצע לחיצה על המקש הימני בעכבר תוך גרירתו במסך תתבצע שליחה של הקואורדינטות (ה-events של אותו ה-bind) לפעולה הפנימית במחלקה send\_coordinates ובאמצעותה יישלח המידע לכלל השחקנים במשחק. קטע קוד זה הוא התשתית למשחק אותו פיתחתי.

יומן רפלקציה

תיעוד מחקר

בחירת GUI

תיעוד הבעיה למחקר:

בפרויקט שביצעתי בכיתה יא' עשיתי שימוש בממשק גרפי באמצעות המודול Pygame, לכן הייתה התלבטות האם ללכת על "הישן והמוכר" או שמא לנסות וללמוד מודול אחד וחדשני יותר, כמו כן היה לי צורך לשימוש ב-tkinter בשביל הצייר ושליחת מיקום העכבר בכל רגע.

חומרי עזר

אתרים רבים באינטרנט יחד עם סרטוני הדרכה, חיפוש שאלות ספציפיות ובעיקר המצגות שהועלו למרחב הכיתתי שלנו בGOOGLE CLASROOM המסייעות בעבודה עם tkinter.

מסקנה

להשתמש במודול, ותמיד לשאוף ללמוד ולהתפתח.

רפלקציה

נהניתי מעשיית הפרויקט, לא אשקר היו עליות ומורדות. זאת הפעם הראשונה בה הכנתי פרויקט רציני ומשמעותי בהיקף כזה רחב המשלב כל כך הרבה מושגים ונושאים שלמדתי בשנים הקודמות. במהלך הפרויקט למדתי כיצד להתמודד עם בעיות לא צפויות, כיצד לחקור נושאים שונים וכיצד להכין תיק פרויקט. בנוסף, למדתי כיצד לתכנת ולתכנן מערכות באופן טוב יותר, המאפשר עבודה יעילה, מהירה, ופשוטה ברגע שבסיס המערכת גמור. יתר על כן, כמובן, גיליתי בעקבות המחקר מודולים ושיטות תכנות, שאני סמוך ובטוח שישמשו אותי גם בעתיד. לסיכום, נהניתי ולמדתי רבות מהמחקר ומהעבודה על הפרויקט, ואני מאוד שמח מהתוצר שהצלחתי להכין.

תודות

ברצוני להודות למספר אנשים שסייעו לי בתהליך ההכנה של הפרויקט. בראש ובראשונה תודה למוטי מתתיהו המורה שלי לסייבר והמנחה שלי בפרויקט שתמיד סיפק אוזן קשבת, עזר בעת הצורך ונתן רעיונות לשיפור והתקדמות בפרויקט.

כמו כן, תודה לתלמידי הכיתה שעזרו בכלל הפרויקט וביניהם תודה מיוחדת ליסמין שלקחה חלק מיוחד בעזרה בפרויקט שלי.

וובליוגרפיה

בתחום התקשורת:

* <https://www.tutorialspoint.com/python-program-to-find-the-ip-address-of-the-client> - קוד למציאת ה-IP של הלקוח.
* <https://stackoverflow.com/questions/25426447/creating-non-blocking-socket-in-python> - מציאת דרך ליצור לקוח שלא נתקע עד לקבלת הודעה מהשרת. טיפול בבלוקינג.

בתחום מסדי הנתונים:

* <https://sqlitebrowser.org/> - הורדת DB Browser For SQLite לצפייה עריכה ויצירה של מאגר הנתונים.
* <https://www.youtube.com/watch?v=Pni6WxHFTUg> – איך להוריד וליצור מסד נתונים.

בתחום הממשק הגרפי:

* <https://stackoverflow.com/questions/15839491/how-to-clear-tkinter-canvas> - איך לנקות את המסך בין סיבוב לסיבוב מכל ציור.
* <https://stackoverflow.com/questions/6433369/deleting-and-changing-a-tkinter-event-binding> - הסבר על שימוש ב- events ב-tkinter.
* <https://www.youtube.com/watch?v=GLnNPjL1U2g&ab_channel=Codemy.comCodemy.com> – הסבר על events ו-bind.

בתחום התהליכונים:

* <https://www.geeksforgeeks.org/python-different-ways-to-kill-a-thread/> - איך לעצור thread.
* <https://stackoverflow.com/questions/27073762/tcl-asyncdelete-error-multithreading-python> - חקירת בעיה של שימוש במספר threads בו זמנית.
* <https://stackoverflow.com/questions/14694408/runtimeerror-main-thread-is-not-in-main-loop> - חקירת בעיה ה-thread הראשי לא נמצא בלולאה הראשית.

נספחים

קישורים של הפרויקט

קישור לMVP שיצרתי:

<https://drive.google.com/file/d/1XJx6nvEZ9YzvY6THOetSCMkC7tsHH2eD/view?usp=sharing>

קישור לקבצי הפרויקט בגיט-האב:

<https://github.com/ron1506/DrawAndGuess>

מילון מונחים

* פונקציית Hash (בעברית פונקציית גיבוב) - היא פונקציה שממירה [קלט](https://he.wikipedia.org/wiki/%D7%A7%D7%9C%D7%98) חופשי באורך משתנה ל[פלט](https://he.wikipedia.org/wiki/%D7%A4%D7%9C%D7%98) באורך קבוע, בדרך כלל קצר בהרבה.
* :Client-Server-model [ארכיטקטורת תוכנה](https://he.wikipedia.org/wiki/%D7%90%D7%A8%D7%9B%D7%99%D7%98%D7%A7%D7%98%D7%95%D7%A8%D7%AA_%D7%AA%D7%95%D7%9B%D7%A0%D7%94) ל[חישוב מבוזר](https://he.wikipedia.org/wiki/%D7%97%D7%99%D7%A9%D7%95%D7%91_%D7%9E%D7%91%D7%95%D7%96%D7%A8), אשר מגדירה את היחס בין [תוכנות](https://he.wikipedia.org/wiki/%D7%AA%D7%95%D7%9B%D7%A0%D7%94) משתפות פעולה. המודל מחלק את המשימות או עומס העבודה בין ספק השירות או המשאבים - ה[שרת](https://he.wikipedia.org/wiki/%D7%A9%D7%A8%D7%AA_(%D7%9E%D7%97%D7%A9%D7%91%D7%99%D7%9D)), לבין מבקש השירות – הלקוח.
* Opcode- קיצור ל- Operation Code מכיל קוד פעולה ובו הוראה למכונה לבצע פעולה כלשהי, כך למשל: הפעולה push במבנה נתונים stack בשפת c#.
* RFC (Request For Comments)- סדרה של מזכרים בתחום טכנולוגיות אינטרנט שונות.
* Threads (תהליכונים)- מערכות הפעלה מודרניות מאפשרות לנהל במסגרת ריצה של [תהליך](https://www.hamichlol.org.il/%D7%AA%D7%94%D7%9C%D7%99%D7%9A_(%D7%9E%D7%93%D7%A2%D7%99_%D7%94%D7%9E%D7%97%D7%A9%D7%91)) (Process) מספר תהליכונים הרצים במקביל במרחב כתובות אחד. במערכות אלו כל תהליך חדש מתחיל את ביצועו באמצעות 'תהליכון ראשי' אשר עשוי בהמשך ליצור תהליכונים נוספים. מנגנון הריצה באמצעות תהליכונים מאפשר לספק למשתמש במערכת ההפעלה מהירות תגובה ורציפות פעולה כאשר התהליך (יישום) מבצע כמה משימות במקביל.
* צד הלקוח- ב[אינטרנט](https://he.wikipedia.org/wiki/%D7%90%D7%99%D7%A0%D7%98%D7%A8%D7%A0%D7%98), ה[מחשב](https://he.wikipedia.org/wiki/%D7%9E%D7%97%D7%A9%D7%91) הביתי מהווה "לקוח" שמקבל את המידע שהגיע מה[שרת](https://he.wikipedia.org/wiki/%D7%A9%D7%A8%D7%AA_(%D7%9E%D7%97%D7%A9%D7%91%D7%99%D7%9D)), ומציג אותו למשתמש באמצעות [תוכנה](https://he.wikipedia.org/wiki/%D7%AA%D7%95%D7%9B%D7%A0%D7%94) ייעודית, לרוב [דפדפן](https://he.wikipedia.org/wiki/%D7%93%D7%A4%D7%93%D7%A4%D7%9F). המושג "תכנות בצד הלקוח" מתאר את הפעולות שמבצעים הדפדפן ושאר חלקי המחשב של הלקוח, כדי להציג על צג הלקוח את הדף שהגיע מהשרת.
* צד השרת- לרוב, שרת היא [תוכנת מחשב](https://he.wikipedia.org/wiki/%D7%AA%D7%95%D7%9B%D7%A0%D7%94) אשר רצה על גבי מכונת שרת מרוחקת, ונגישה מה[מחשב](https://he.wikipedia.org/wiki/%D7%9E%D7%97%D7%A9%D7%91_%D7%90%D7%99%D7%A9%D7%99) או תחנת העבודה המקומית של ה[משתמש](https://he.wikipedia.org/wiki/%D7%9E%D7%A9%D7%AA%D7%9E%D7%A9_(%D7%9E%D7%97%D7%A9%D7%95%D7%91)). יש פעולות המבוצעות בצד השרת מכיוון שהן דורשות גישה ל[נתונים](https://he.wikipedia.org/wiki/%D7%A0%D7%AA%D7%95%D7%A0%D7%99%D7%9D) או לפונקציונליות שאינה זמינה ב[צד הלקוח](https://he.wikipedia.org/wiki/%D7%A6%D7%93_%D7%9C%D7%A7%D7%95%D7%97), או מכיוון שפעולות אלה דורשות התנהגות מסוימת שאינה [בטוחה](https://he.wikipedia.org/wiki/%D7%90%D7%91%D7%98%D7%97%D7%AA_%D7%9E%D7%99%D7%93%D7%A2) כאשר היא מבוצעת בצד הלקוח. פעולות צד-שרת כוללות גם עיבוד ואחסון נתונים המגיעים מצד הלקוח לשרת. דוגמה לעיבוד המתבצע בצד השרת היא יצירה ועדכון של [בסיס הנתונים](https://he.wikipedia.org/wiki/%D7%91%D7%A1%D7%99%D7%A1_%D7%A0%D7%AA%D7%95%D7%A0%D7%99%D7%9D) [MySQL](https://he.wikipedia.org/wiki/MySQL).
* LAN (Local Area Network)- רֶשֶׁת תִּקְשֹׁרֶת מְקוֹמִית היא [רשת מחשבים](https://he.wikipedia.org/wiki/%D7%A8%D7%A9%D7%AA_%D7%9E%D7%97%D7%A9%D7%91%D7%99%D7%9D) המתפרסת, בשונה מ[רשת עירונית](https://he.wikipedia.org/wiki/%D7%A8%D7%A9%D7%AA_%D7%A2%D7%99%D7%A8%D7%95%D7%A0%D7%99%D7%AA) (MAN) או [רשת אזורית](https://he.wikipedia.org/wiki/%D7%A8%D7%A9%D7%AA_%D7%90%D7%96%D7%95%D7%A8%D7%99%D7%AA) (WAN), על אזור גאוגרפי מוגבל (עד אלפים בודדים של מטרים רבועים), בדרך כלל בתוך בניין אחד, או בניינים סמוכים. רשת כזו מתאפיינת ב[רוחב פס](https://he.wikipedia.org/wiki/%D7%A8%D7%95%D7%97%D7%91_%D7%A4%D7%A1) גבוה יחסית וב[זמן השהיה](https://he.wikipedia.org/wiki/%D7%96%D7%9E%D7%9F_%D7%94%D7%A9%D7%94%D7%99%D7%94) נמוך יחסית.
* SQLite- הוא [בסיס נתונים יחסי](https://he.wikipedia.org/wiki/%D7%91%D7%A1%D7%99%D7%A1_%D7%A0%D7%AA%D7%95%D7%A0%D7%99%D7%9D_%D7%99%D7%97%D7%A1%D7%99) [משובץ](https://he.wikipedia.org/w/index.php?title=%D7%91%D7%A1%D7%99%D7%A1_%D7%A0%D7%AA%D7%95%D7%A0%D7%99%D7%9D_%D7%9E%D7%A9%D7%95%D7%91%D7%A5&action=edit&redlink=1). כלומר: להבדיל מרוב [בסיסי הנתונים היחסיים](https://he.wikipedia.org/wiki/%D7%91%D7%A1%D7%99%D7%A1_%D7%A0%D7%AA%D7%95%D7%A0%D7%99%D7%9D_%D7%99%D7%97%D7%A1%D7%99), SQLite אינו [תהליך](https://he.wikipedia.org/wiki/%D7%AA%D7%94%D7%9C%D7%99%D7%9A_(%D7%9E%D7%93%D7%A2%D7%99_%D7%94%D7%9E%D7%97%D7%A9%D7%91)) עצמאי נפרד המקבל קריאות מתהליכים נפרדים או מרוחקים, אלא [ספריה](https://he.wikipedia.org/wiki/%D7%A1%D7%A4%D7%A8%D7%99%D7%99%D7%AA_%D7%AA%D7%95%D7%9B%D7%A0%D7%94) הנקראת או מופעלת מתוך תהליך קיים. הקוד מפורסם כקוד פתוח ללא כל הגנת [זכויות יוצרים](https://he.wikipedia.org/wiki/%D7%96%D7%9B%D7%95%D7%99%D7%95%D7%AA_%D7%99%D7%95%D7%A6%D7%A8%D7%99%D7%9D) כ[רשות הציבור](https://he.wikipedia.org/wiki/%D7%A8%D7%A9%D7%95%D7%AA_%D7%94%D7%A6%D7%99%D7%91%D7%95%D7%A8).
* DB Browser for SQLite (DB4S) is a high quality, visual, open source tool to create, design, and edit database files compatible with SQLite. DB4S is for users and developers who want to create, search, and edit databases. DB4S uses a familiar spreadsheet-like interface, and complicated SQL commands do not have to be learned.