עבודה בתכנון ותכנות מערכות

התמחות הגנת הסייבר

5 יחידות לימוד – מספר שאלון 883589

**Social Painter**

לוח ציור שיתופי

מגיש: איתמר קנה

מורה\מנחה: מוטי מתתיהו

בית ספר: אורט אבין רמת גן

שנה: תש"ף 2019

תוכן עניינים

[מסמך ייזום 3](#_Toc23752252)

[רקע 4](#_Toc23752253)

[**מבוא** 4](#_Toc23752254)

[הגדרות 4](#_Toc23752255)

[**משתמשי המערכת** 4](#_Toc23752256)

[**מטרת המערכת ויעדיה** 4](#_Toc23752257)

[**ארכיטקטורת המערכת** 4](#_Toc23752258)

[**הגדרת רכיבי המערכת** 4](#_Toc23752259)

[**אתגרי הפרויקט** 5](#_Toc23752260)

[**תיחום המערכת** 5](#_Toc23752261)

[**סיקור מצב השוק כיום** 5](#_Toc23752262)

# מסמך ייזום

## **רקע**

### **מבוא** למה בחרתי בפרוייקט זה? קראתי בעבר על r/place (צייר שיתופי ענקי שפעל למשך 3 ימים באחד לאפריל 2017). (להוסיף מידע מהמאמר) ורציתי לבנות משהו דומה ובמיוחד שאני אוהב את תחום הציור. מה מעניין ומאתגר אותי בפרוייקט? מאתגר אותי לימוד רשתות ומימוש הידע שלמדתי בכיתה ולבנות מערכת באינטרנט שתאפשר התחברות לכל משתמש מרחוק באמצעות דפדפן. הפרוייקט הוא תוצר של למידה במגמת הנדסת תוכנה. הפרוייקט מאפשר לי ליישם את תחומי הלימוד: תכנות בשפת Python רשתות, אבטחת מידע, וכן לממש כלים שאותם רכשתי, כמו חקירת נושאים חדשים ולמידה עצמית של: redis, sqlite ומודלים שונים בפייתון המתקשרים לרשתות, אני אנסה לבנות תוכנה שאפשר להריץ בעולם הפתוח אך איני מאמין בכך במלואי.

המערכת אמורה בעקרון לתת לאנשים ברחבי העולם קנבס משותף שבו אנשים משאירים את חותמם.

**תיאור המוצר:**

**המטרות המרכזיות של העבודה:**

לוח ציור (קנבס) בגודל 1000x1000 פיקסלים המאפשר למספר אנשים להתחבר ביחד ולצייר ציור שיתופי.

**התרחישים שהמערכת עונה עליהם:**

לאפשר לשחקן להתחבר עם שם משתמש וסיסמה

לאפשר לשחקן לצבוע על לוח הציור פיקסל אחד כל 1 דקות.

לקשר את כל המחשבים יחד.

**אילוצים ודרישות:**

* המערכת תאפשר חיבור מספר רב של משתמשים שיוכלו לצייר על הלוח במקביל.
* המערכת תשלח באופן יעיל את השינויים בלוח הציור.
  + בפעם הראשונה תשלח את כל המידע שיש על לוח ציור.
  + לאחר מכן, ישלחו רק עדכונים של השינויים בלוח ציור.
  + המערכת תתמוך בפעולת refresh ותעדכן את כל לוח הציור בדפדפן הלקוח.
* המערכת תאפשר תקשורת בהודעות טקסטואליות.
* הדפדפן חייב לתמוך בעוגיות

### **אתגרי הפרויקט**

הפצת מידע למספר רב של משתמשים באופן מסונכרן.  
למידת מודולים חדשים Python, למידת javascript, html, css.  
עבודה עם בסיס נתונים.

### **תחום המערכת:**

המערכת תרוץ על פיתון ותתקשר עם בסיס נתונים מסוג Redis ובסיס

נתונים מסוג Cassandra.

המודלים שישמשו אותי יהיו socketio המאפשר נוחות רבה עם תקשורת וגם מאפשר broadcast לכל השחקנים בקלות. מודל tk לגרפיקה. ומודלים המאפשרים קישור עם redis ועם Cassandra.

**סביבה וכלי פיתוח:**

Python 3.7.5 בצד השרת.

Html, css, js בצד הלקוח.

Redis – לשמירת ערכים כללים (בעיקר לוח המשחק) ללא טבלה ובנוסף לכך כתור הודעות.

Sqlite – לשמירת מידע על משתמשי המערכת.

הפיתוח יתבצע בwindows.

הלקוח ישתמש בדפדפן כרום (המערכת תומכת בדפדפנים נוספים).

Visual Studio Code – עורך קוד ה.

Pycharm – עורך קוד הפייתון.

**בדיקות:**

Redis-cli, שירות של רדיס בubunto כדי להסתכל על מסך

DB Browser כדי להסתכל על

**השלבים הנותרים בפיתוח המוצר:**השלבים הנותרים בפיתוח המוצר הינם:

|  |  |
| --- | --- |
|  | תאריך סיום |
| בניית השרת ומסד הנתונים | 12.12.19 |
| כניסה באמצעות שם משתמש וסיסמה. | 20.12.19 |
| האפשרות לבצע צ'אטים \ שיחות קוליות. | 10.2.20 |
| ציור באמצעות | 29.2.20 |
| מקום רב יותר לצייר | 30.3.20 |

**מסמך אפיון:**

**פונקציונליות המערכת:**

**פעולות אצל המשתמש:**

1. הרשמה ראשונית למערכת.
2. אימות באמצעות אימייל.
3. התחברות משתמש קיים למערכת.
4. אפשרות איפוס סיסמה.
5. צביעת פיקסל על הקנבס:
   * אפשרות לחיצה על הקנבס באמצעות העכבר לצביעת פיקסל.
6. ניווט באמצעות זיהוי משאבים בשורת הurl
   * עדכון המסך.
   * התאמת הפרמטרים בfragment כך שיתאים למידע.
   * התאמת פרמטרים כך
7. ניווט בקנבס:
   * הזזת הקנבס באמצעות חצים.
   * הזזת הקנבס באמצעות העכבר.
8. אפשרות לבחור צבע לצבוע:
   * באמצעות לחיצה של העכבר.
   * באמצעות מקשים במקלדת.
   * אפשרות לבחור צבע מועדף שיהיה פתוח בכניסה למשחק.
9. אפשרויות נוספות:
   * מסך מלא.
   * אפשרות להיכנס לקישור חיצוני (צ'אט).
10. צביעה מסונכרנת.

**אילוצים עיקריים:**

* המערכת מחייבת עבודה עם דפדפן chrome וחיבור לאינטרנט (בעתיד אוסיף תמיכה לדפדפנים אחרים).

**הנחות יסוד:**

* המערכת תרוץ בסביבת עבודה MS Windows 7 ומעלה.
* הדפדפן שדרכו השחקן יתחבר יתמוך בcookies.
* מתקיימים שרתי redis בip המתאימים.

**דרישות בסיס נתונים:**

בסיס הנתונים ימומש בשפת SQL וישמרו בו נתוני המשתמש, הפתקים שהמנהלים רושמים על משתמשים.

**סביבת עבודה:**

* השרת נכתב בPython 3.7
* ה-GUI יהיה ברשת.
* בסיס הנתונים ייבנה בעזרת ב-SQLITE

**דרישות המערכת:**

**משתמש:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | דרישה | קלט | תהליך | פלט | טיפול בשגיאות |
| 1 | תהליך רישום משתמש חדש | נתוני משתמש:  שם משתמש  סיסמה  אימות סיסמה | העברת הנתונים לשרת ושמירתם בבסיס הנתונים | הלקוח נרשם בהצלחה – עליו להיכנס דרך קישור האימות. | אם אחד מהבאים קורה:  סיסמה קצרה  אימייל תפוס או לא תקין.  שם משתמש תפוס.  כינוי תפוס, תוחזר הודעת שגיאה.  אם המשתמש חסום תשלח הודעת שגיאה מתאימה הכוללת את הסיבה שבגללה נחסם\עד מתי החסימה תפוג אם בכלל |
| 2 | התחברות של משתמש קיים. | שם משתמש וסיסמה. | העברות הנתונים לשרת ובדיקתם בבסיס הנתונים. | הלקוח התחבר בהצלחה למעכרת – מעבר ללובי. | אם אחד הנתונים לא מתאימים\ שהמשתמש כבר מחובר למערכת\המשתמש חסום תחוזר הודעת שגיאה. |
| 3 | קישור url לאימות | שליחת מייל לאימות השם משתמש והסיסמה. | שליחת הודעת מייל למשתמש עם url לקישור שכניסה דרכו מאפשרת את המשתמש | הלקוח התחבר | N/A |
| 4 | איפוס סיסמה. | שם משתמש ואימייל | בדיקת השם משתמש והאימייל, אם הם תואמים ישלח לאמייל קוד לשחזור סיסמה, לאחר מכן המשתמש יתבקש להכניס קוד זה, אם הקוד נכון תינתן לו אפשרות לשנות את סיסמתו | מתן אפשרות לאיפוס סיסמה. | אם שם משתמש והאמייל לא תואמים (לא שייכים לאף משתמש) או שכתובת האי-מייל לא תקינה\המייל\משתמש חסום תוחזר הודעה מתאימה. |
| 5 | צביעת פיקסל על הקנבס | מיקום עכבר | האדם לוחץ על פיקסל, נשלחת הודעה לשרת עם הפיקסל והשחקן. השרת בודק אם יתכן, | לשרת נשלחת הודעה לצבוע פיקסל | מיקום הפיקסל לא תקין, לא צובע.  השחקן לא יכול לשים צבע – לרשום לו. |
| 6א | הזזת קנבס באמצעות חצים | כפתור | האדם לוחץ על כפתור ומזיז את הקנבס. | הקנבס זז. | הקנבס יוצא מחוץ לטווח. |
| 6ב | הזזת קנבס באמצעות העכבר | מיקום עכבר | האדם לוחץ על כפתור וגורר את הקנבס. | הקנבס זז | הקנבס יוצא מחוץ לטווח. |
| 6ג | הזזת הקנבס באמצעות fragments | Path fragments | המשתמש משנה את כתובת הpath כך שהפרמטרים יתאימו למיקום\גודל שהוא רוצה לראות בקנבס | הצג משתנה כך שהקנבס יראה בהגדלה המתאימה ובמיקום כך שהאמצע יהיה המיקום הנכנס | פרמטר לא תקין - לא ישתנה  פרמטר לא מופיע – יכניס את הרגיל  אם מתרחש תקלה הצג ישנה את הפרמטרים. |
| ג | אפשרות לבחור צבע באמצעות העכבר | לחיצה של עכבר | האדם לוחץ על כפתור המייצג צבע וקובע את ייצוגו | כפתור הצבע מסומן, אם היה צבע לפני הוא מפסיק להיות מסומן. | N/A |
| 8א | אפשרות לבחור צבע באמצעות עכבר. | לחיצה על עכבר. | המשתמש לוחץ על כפתור, "מזיז" את סימון הצבע לצבע מימין או שמאל (מעגלי) | כפתור הצבע מסומן, אם היה צבע לפני הוא מפסיק להיות מסומן. | N/A |
| 8ב | אפשרות לשנות צבע באמצעות המקלדת | לחיצה על מקש | המשתמש לוחץ על כפתור להרמת חלון הצבעים.  כפתור אחר לעבוד לצבע הימיני.  כפתור אחר לעבור לצבע השמאלי. | הצבע שנבחר מקודם מפסיק להיות מסומן והצבע הבא מסומן במקומו\חלון הצבעים מופיע\נסגר. | מנסים לעבור לכפתור הימני מהכפתור הימני ביותר (ומקרה דומה לשמאלי) , פתרון: יעבור לכפתור בקצה השני. |
| 9 | צביעה מסונכרת | הודעה | השרת שולח למשתמש הודעה כי נצבע פיקסל | צביעת הפיקסל המתאים במסכי כל המשתמשים. | אם ההודעה לא תקינה: ההודעה נזרקת |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| דרישה | קלט | תהליך | פלט | טיפול בשגיאות |
| אחראיות |  |  |  |  |

**מסמך עיצוב:**

* חשוב מאוד לציין, כל העבודה מול בסיס הנתונים מסוג sqlite נעשת דרך המודל sql-alchemy המאפשרת לעבוד עם בסיסי נתונים מסוג SQL אל

**דמות:**

|  |  |  |
| --- | --- | --- |
| sqlite | Python | Name |
| int | **int** | **id** |
| varchar(15) | **Int** | **name** |
| varchar(128) | **bytes** | **password** |
| varchar(254) | **String** | **email** |
| float -> datetime | **float** | **next\_time** |
| Boolean | **Role** | **role** |

* **ID – האינדקס של הדמות (יותר מהיר לחפש באמצעות int מאשר string)**
* **Name – השם משתמש של השחקן.**
* **Password –** הסיסמה המוצפנת של השחקן, בעזרתה המשתמש מתחבר לשרת.
* **Email –** המייל של המשתמש בשביל אימותים עם השרת.
* **next\_time –** הזמן הבא שבוא השחקן יוכל לצבוע פיקסל, הזמן נשמר ככמות השניות מאז הראשון הראשון 1970 בחצות.
* **role –** לבדוק את רמת הניהול של המשתמש, מיוצג בידי IntEnum שערכיהם נשמרים באמצעות

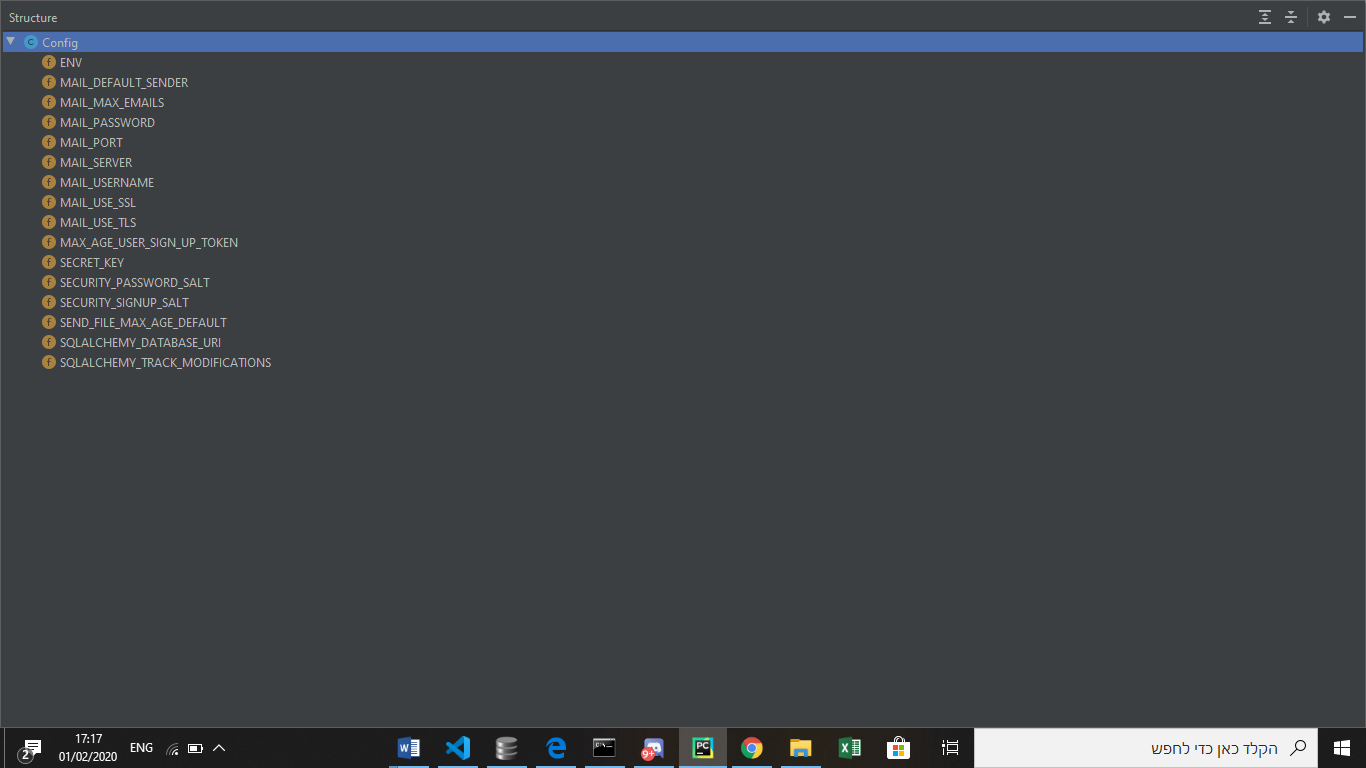
**פיקסל:**

|  |  |  |
| --- | --- | --- |
| sqlite | Python | Name |
| int | **int** | **id** |
| smallint | **Int** | **x** |
| smallint | **int** | **y** |
| smallint | **int** | **color** |
| float | **Float-> datetime** | **drawn** |
| int | **int-> User** | **drawer** |

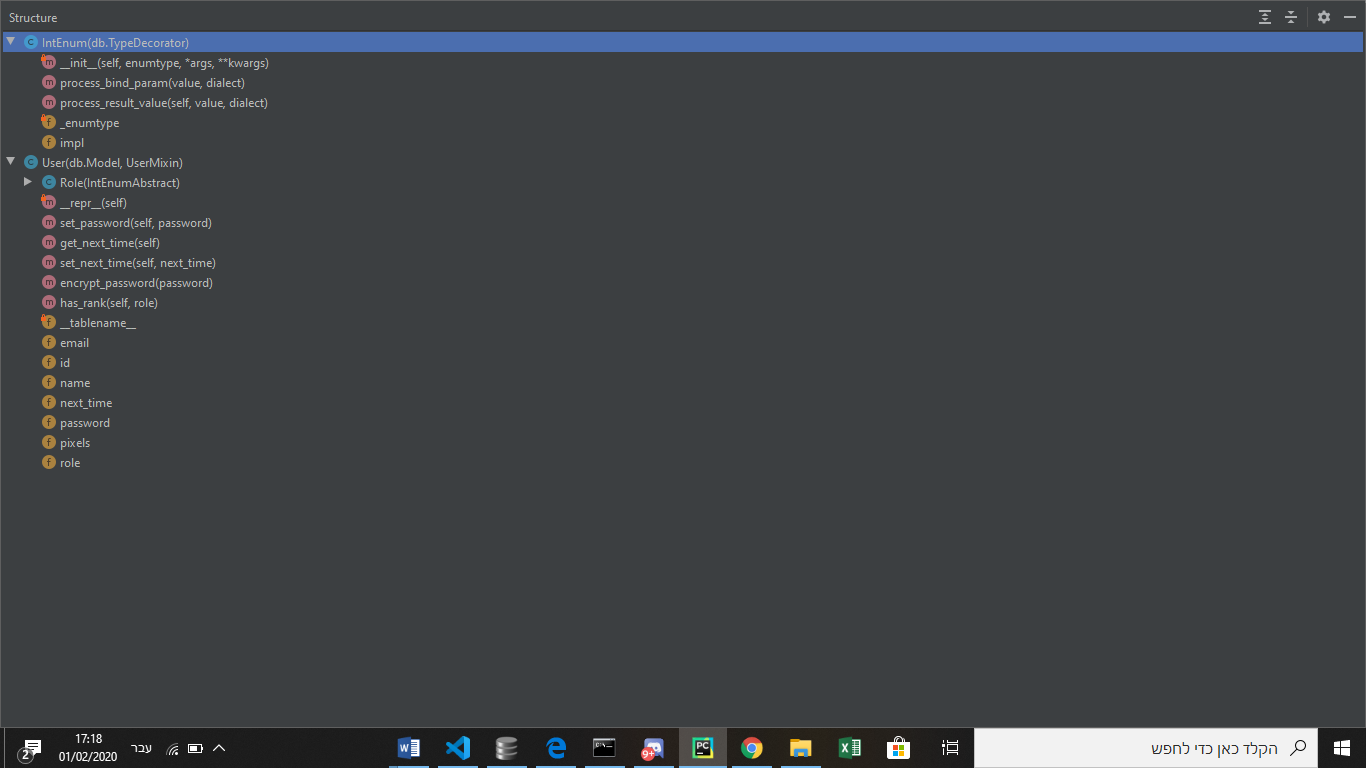
* id - **האינדקס של הפיקסל הנבצע (מהירות חיפוש)**
* x – מיקום הפיקסל בציר ה-x (בין 0 ל-999).
* y – מיקוםהפיקסל הנצבע בציר ה-y (בין 0 ל-999)
* **color – המספר המייצג את הצבע הנצבע (בין 1 ל-16).**
* **drawn –** הזמן הבא שבוא הפיקסל נצבע, הזמן נשמר ככמות השניות מאז הראשון הראשון 1970 בחצות.
* **drawer –** צובע הפיקסל, השחקן שהיה אחראי לצביעת הפיקסל.

**Structure:**

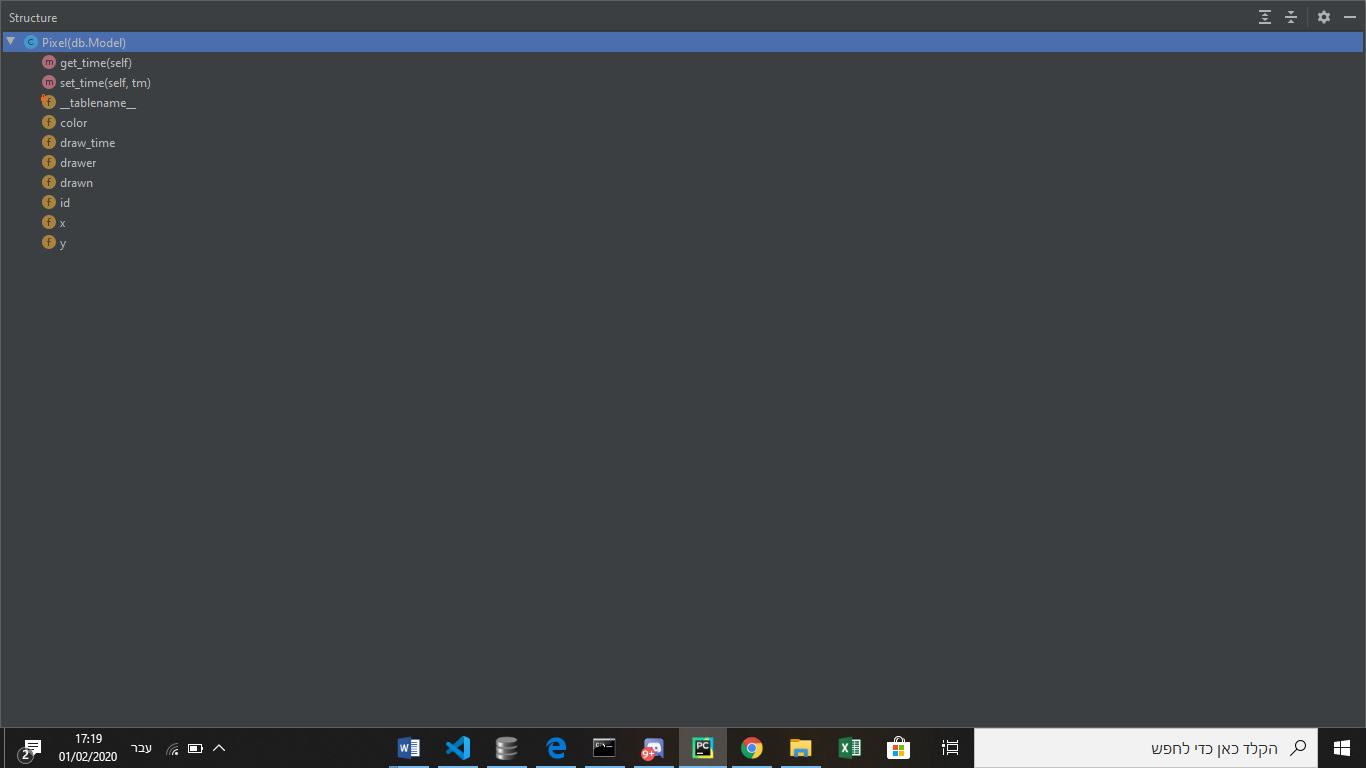
**Config:**



**User.py:**



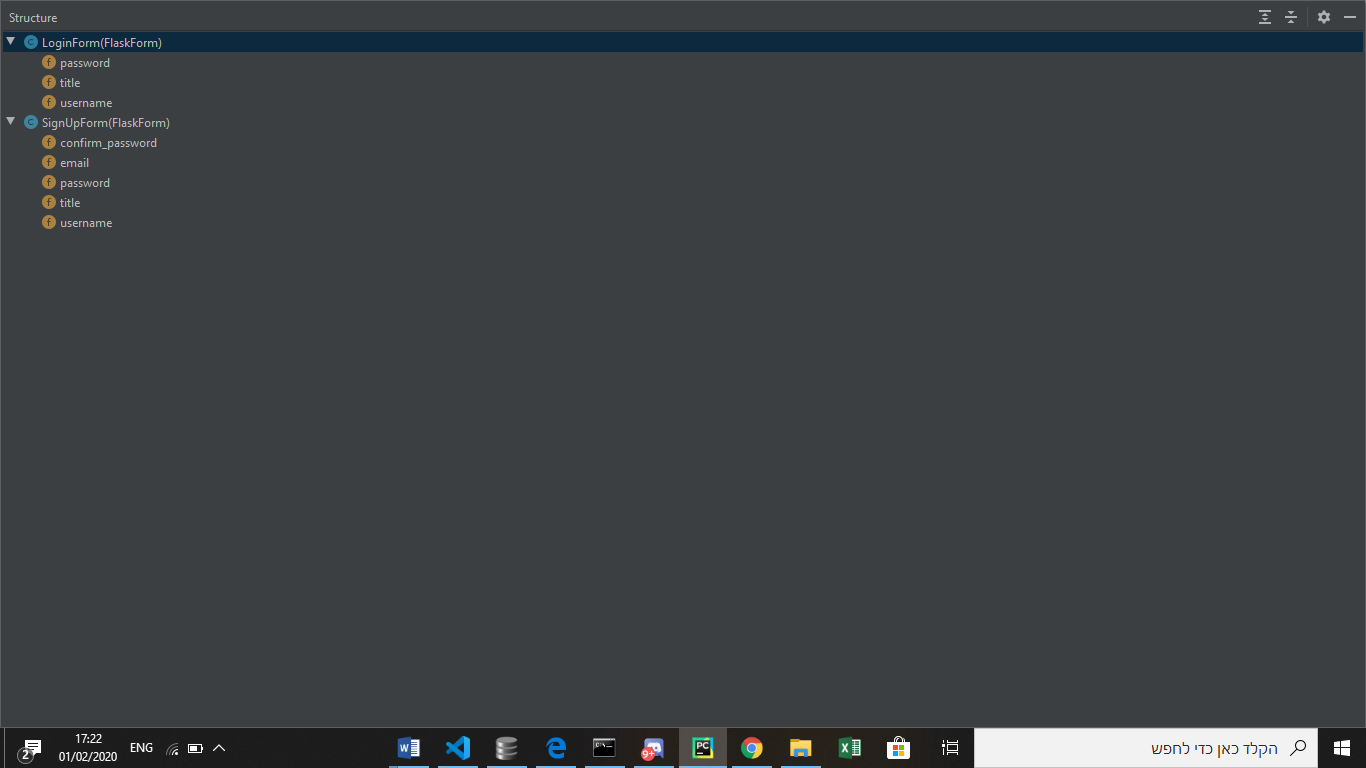
**models\pixel.py:**



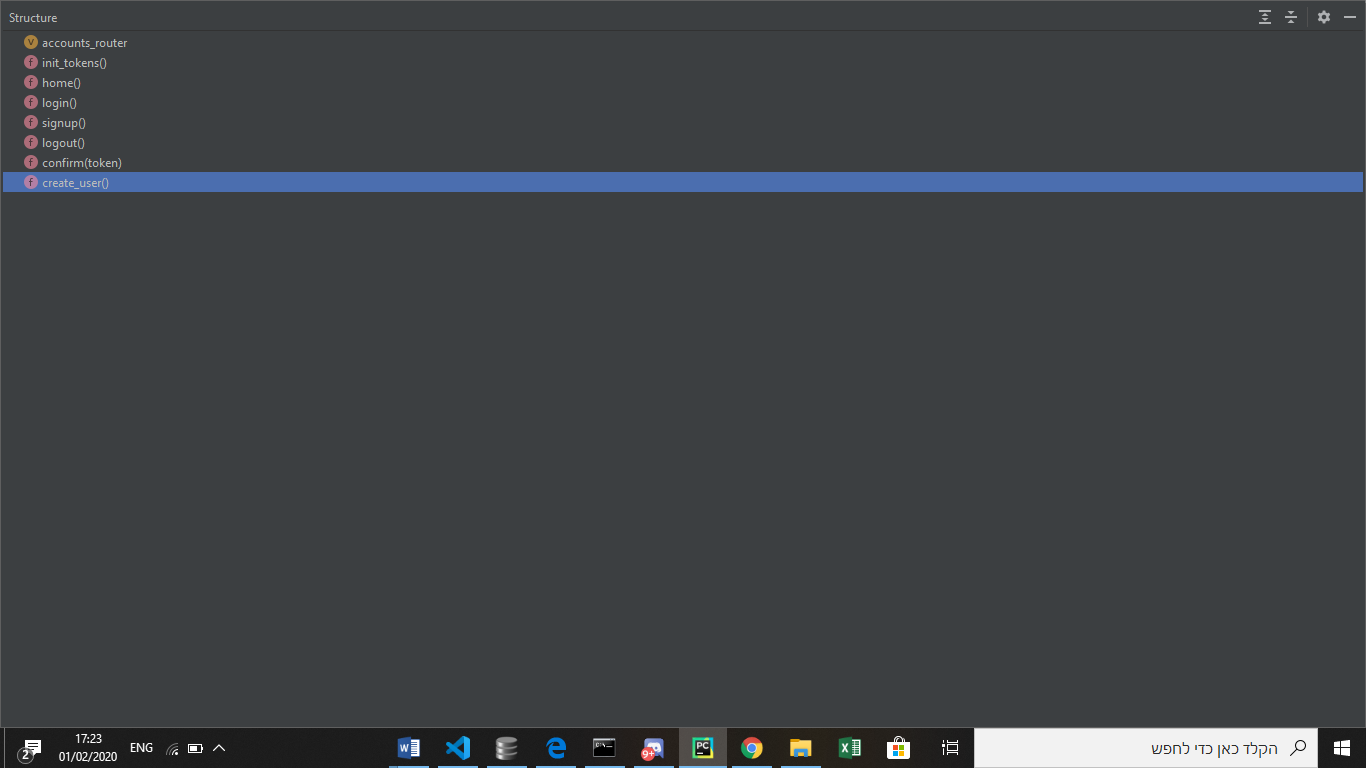
**Apps:**

**\Accounts:**

**\forms.py:**



**\urls.py:**



**תיאור מודלים – פייתון:**

Apps:

* Accounts – חבילה הקשורה לכל הנושא של תקשורת עם חשבונות.
  + urls.py – הקובץ מחזיק את הבלופרינט המתקשר לכל הקישורים המשתייכים לנושא ההתחברות, ושם כל הפונקציות שמקושרים לurls.
  + utils.py – מחזיק את כל הפונקציות והאובייקטים המתקשרים ספציפית אל accounts.py ספציפית.
  + forms.py – משמר אובייקטים המקושרים להתחברות, הרשמה וכד...
  + Mail.py – מחזיק פונקציות היוצרות אובייקטי מייל לשלוח באמצעות accounts.py. כגון מייל הרשמה.
* Place – חבילה הקשורה לקישור האפליקציה.
  + Urls.py – כל הניווט לאפליקציה.
* Other.py – חבילה הקשורה לכל הניווטים שאינם בכיוון ראשי עם האפליקציה כגון הmemes או שליחת קבצים קבועים.
  + urls.py – הקובץ מחזיק את הבלופרינט המתקשר לכל הקישורים המשתייכים לנושא הצייר השיתופי.
  + Helpers.py – מחזיק את כל הפונקציות והאובייקטים המתקשרים לניווט.
* Other – חבילה הקשורה לכל הדברים השונים, כגון אפשרות הmeme ולקישור.
* Models – חבילה המחזיקה את כל האובייקטים הנשמרים בdatabase באמצעות sqlalchemy.
  + User.py – המחלקה של משתמש שעבר את האימות נתונים\שנתוניו נשמרו.
  + Pixel.py – מחלקה של פיקסל שנצבע על המסך.
* Constants.py – קובץ שבו רשומים קבועים המשמשים את כל התוכנה.
* Security.py – קובץ שבו נשמרים הגדרות של האבטחה במסמך כגון האפשרויות של flask-login.
* Extensions.py – קובץ אשר מכיל את התוספים לפלאסק ובונה אותם.
* Functions.py – פונקציות שונות כלליות המשמשות בתוכנה.
* Config.py – קובץ המחזיק מחלקה הנקראת Config, שמגדירה תכונות לאפליקציה
* \_\_init\_\_.py קורא לכל הפעולות המתחילות את השרת, מה שמורץ שם בונה את אפליקצית הפלאסק.
* App.py – קובץ חיצוני המריץ את האפליקציה.

**מבנה הפקודות בשרת:**

הפקודות והפעולות ברוב השרת הם http 1.

אך בחלק מהפעולות (של האפליקציה) משתמשים במבנה פקודה של socket.io, שולחים מחזורת של סוג אירוע ואובייקט מילון המייצג פרמטרים לאירוע, דוגמא לכך:

|  |  |
| --- | --- |
| שם האירוע | פרמטרים |
| Set-board | x, y, color |

השרת והלקוח מבצעים את הפעולות הבסיסיות של socket.io, הרחיב רק על אלה שפועלים באפליקציה.

**פקודות של הלקוח לשרת:**

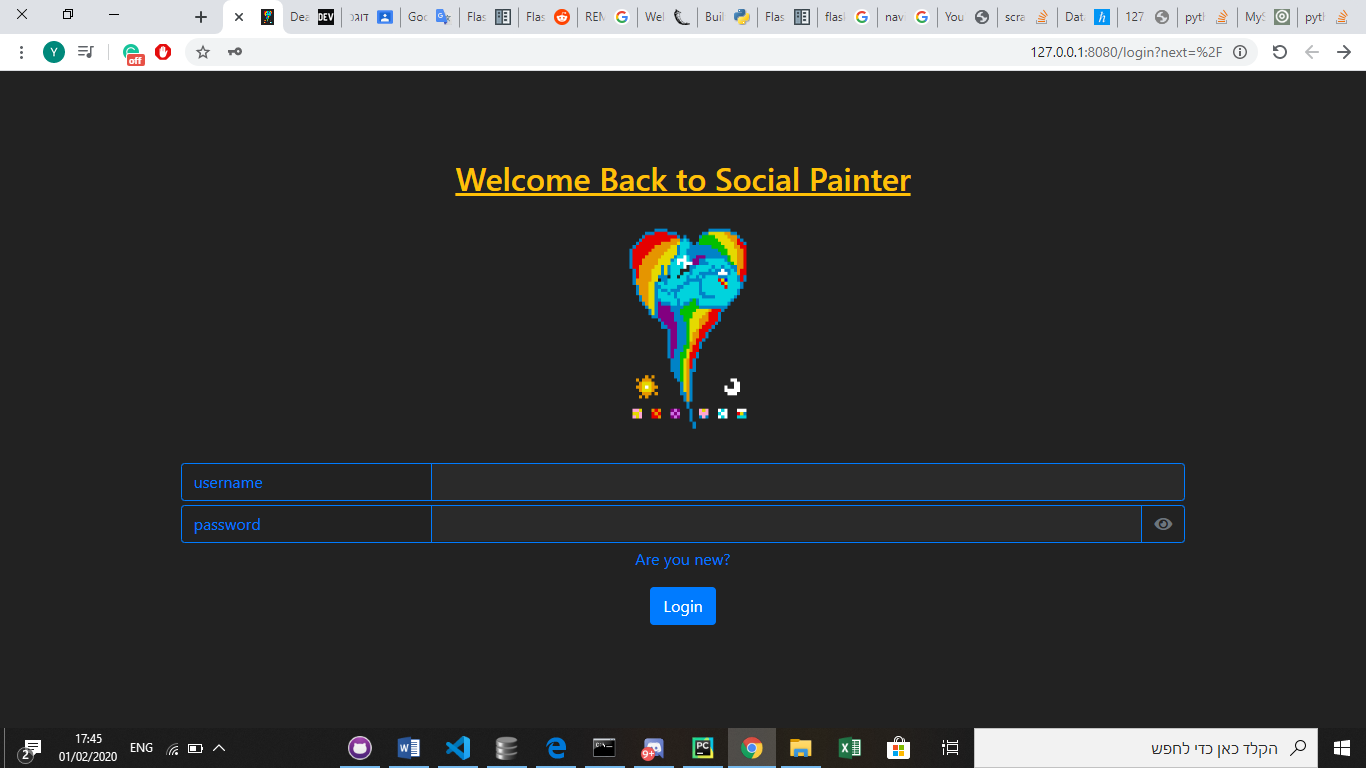
|  |  |  |  |
| --- | --- | --- | --- |
| הודעה | שם האירוע | הסבר | פרמטרים |
| לנסות לקבוע פיקסל על הלוח. | set-board | הלקוח מנסה לצבוע פיקסל על הקנבס | x, y, color |
| התחברות | connect | הלקוח מתחבר באמצעות socket.io אל השרת. השרת מחזיר אל הלקוח את הקנבס ואת הזמן הבא שבו יוכל לעדכן את הלוח. | כלום |

**הודעה של השרת ללקוח:**

|  |  |  |  |
| --- | --- | --- | --- |
| הודעה | שם האירוע | הסבר | פרמטרים |
| התחלת מקום | start-place | השרת משיב ללקוח על החיבור ומודיע לו על פרמטרים שנחוצים לתחילת המסך. | Board  Current\_time |
| קביעת פיקסל | set-board | לאחר בדיקת הנכונות של בקשת הלקוח לצבוע פיקסל השרת שולך הודעה לכל המשתמשים לצבוע את הפיקסל. | X  Y |
|  |  |  |  |

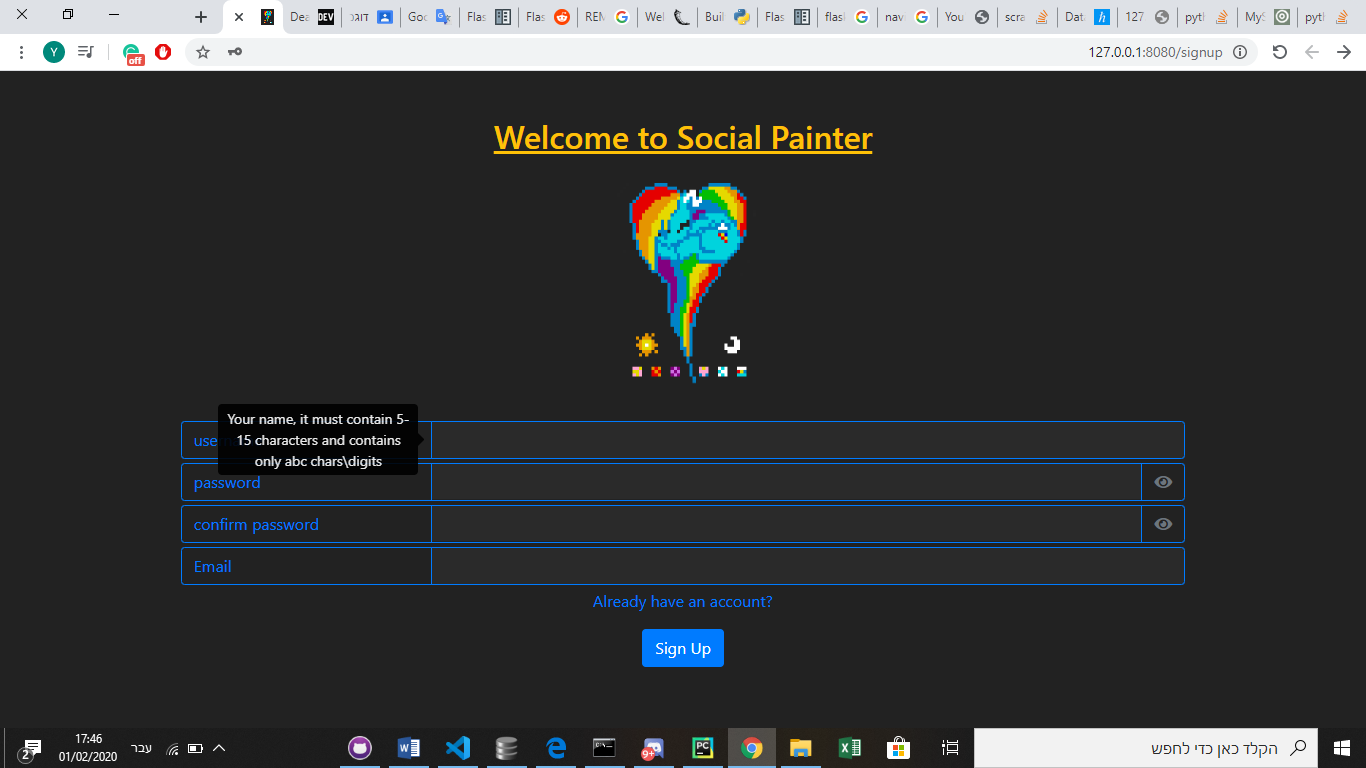
**מסכים:**

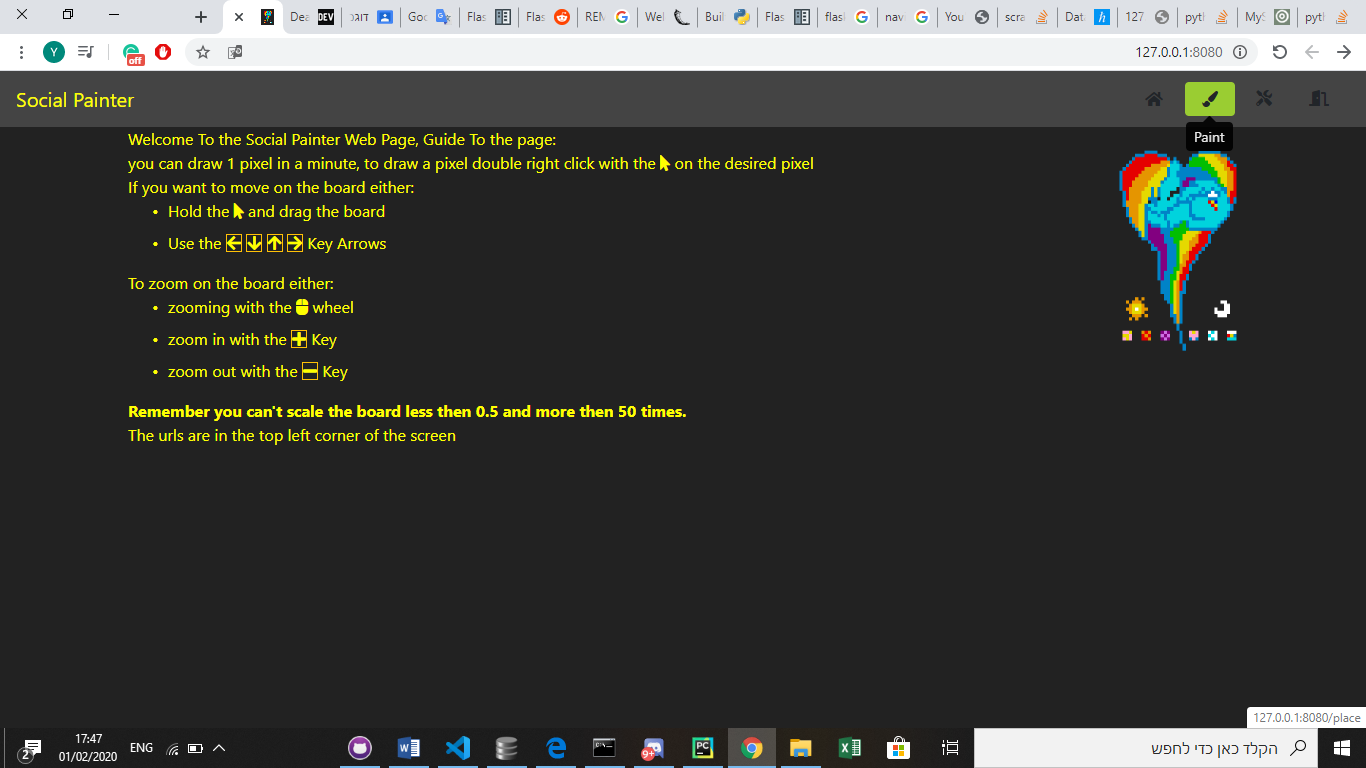
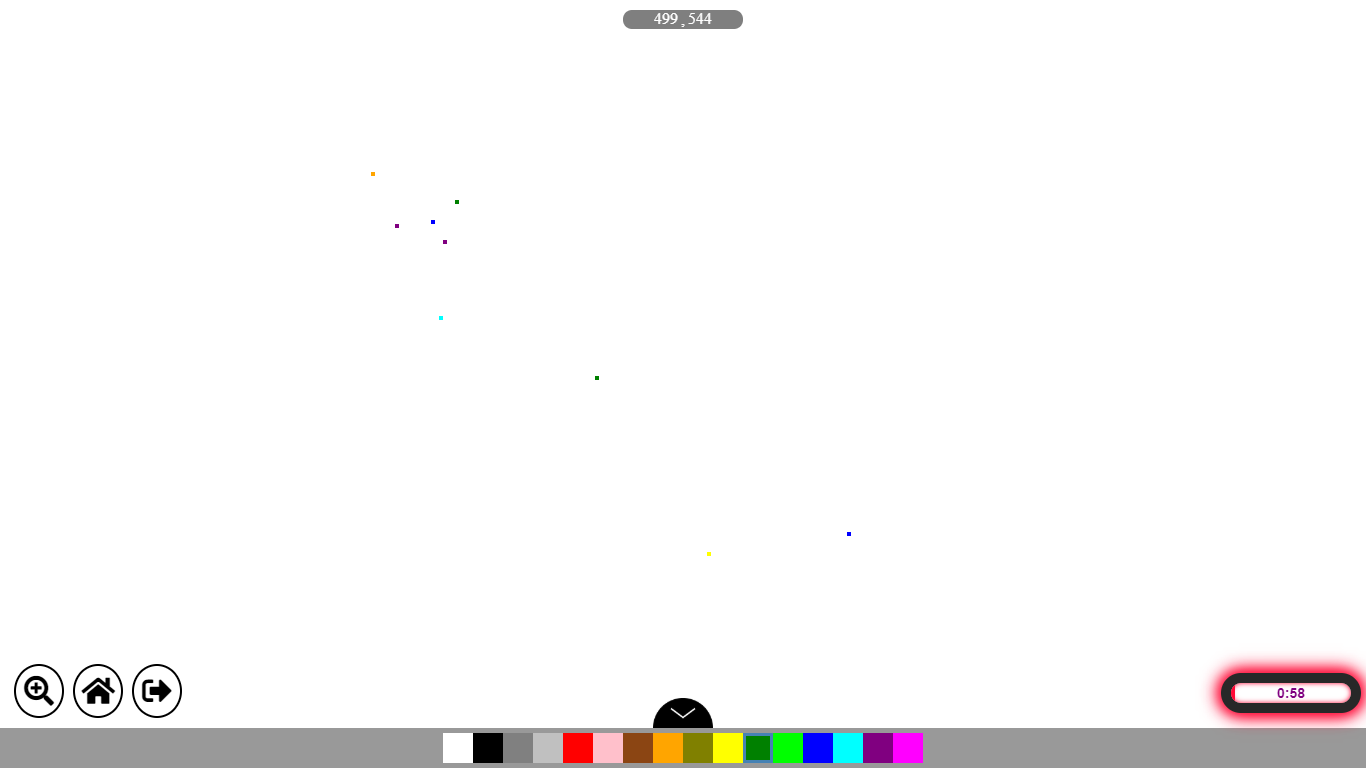
מסך פתיחה:

מציג למשתמשים את האפשרות כניסה שלו למערכת, הרשמה או התחברות רגילה.  


מסך הרשמה:

במקרה שלמשתמש אין עדיין חשבון, הוא יכול לפתוח אחד במסך הזה.

**בית:**

השחקן מועבר למסך הזה לאחר ההתחברות, הוא מסביר לשחקן כיצד לפעולה במשחק.**צייר:**

האפליקציה עצמה, הצייר השיתופי בו ניתן לראות את כל האפשרויות של השחקן, ללכת למסך הבית, לשנות את צבע המסך, הזמן שנותר עד לסיום הזמן שצריך לחכות.