

**שם התלמיד:** רז שניידר

**ת"ז:** 212287510

**שם בית ספר:** אורט שפירא כפר סבא

**שם המורה:** צביקה שטרקמן

**תאריך הגשה:** 29/4/2021



**פרויקט גמר בהגנת סייבר**

**Space Racer**

תוכן עניינים

[מבוא 3](#_Toc70350517)

[תקציר כללי – תיאור הפרויקט: 3](#_Toc70350518)

[מה היו המניעים לפיתוח הפרויקט: 3](#_Toc70350519)

[דרישות פרויקט 4](#_Toc70350520)

[תיאור דרישות פונקציונליות: 4](#_Toc70350521)

[מה קיים בשוק 5](#_Toc70350522)

[טכנולוגיות בהן השתמשתי בפרויקט 6](#_Toc70350523)

[תיאור מבנה הפרויקט 7](#_Toc70350524)

[מדריך למשתמש ותיאור ה-UX/UI למשתמשים 11](#_Toc70350525)

[הדרכה למשתמש 11](#_Toc70350526)

[תמונות: 12](#_Toc70350527)

[מדריך למפתח 14](#_Toc70350528)

[קבצים ומיקומם 14](#_Toc70350529)

[משתנים חשובים 14](#_Toc70350530)

[היבטי אבטחת מידע 15](#_Toc70350531)

[מחקר משמעותי/נושאים שלמדתי 16](#_Toc70350532)

[בעיות שנתקלתי במהלך פיתוח הפרויקט 17](#_Toc70350533)

[תיאור שלבי העבודה שעשיתי במהלך הפרויקט 18](#_Toc70350534)

[תכנון מקורי: 18](#_Toc70350535)

[מה השתנה במהלך הפיתוח: 18](#_Toc70350536)

[בדיקות שעשיתי: 18](#_Toc70350537)

[באגים פתוחים / בעיות קיימות + מגבלות שקיימות: 18](#_Toc70350538)

[רפלקציה – מבט אישי על העבודה ועל תהליך פיתוח הפרויקט 19](#_Toc70350539)

# מבוא

## תקציר כללי – תיאור הפרויקט:

הפרויקט שלי הוא מערכת לתחרויות הקלדה online. המערכת כוללת שרת שיש בו אתר שאפשר להירשם אליו ולהיכנס עם חברים לדו קרב בתחרות כתיבה מהירה. התחרות מתחילה באותו הזמן לשני המשתתפים ומופיע להם על המסך את המשפט שהם צריכים לכתוב, קופסת כתיבה לכתוב את המשפט, ואיפה הם נמצא ביחס ליריב. בסוף התחרות מוצג למשתמשים מי מקום ראשון, שני ושלישי עם מהירות הכתיבה של כל המשתתפים.

## מה היו המניעים לפיתוח הפרויקט:

המניע שלי לכתיבת הפרויקט הזה הוא שאני מאוד אוהב תחרויות כתיבה. בכיתה יא' עשיתי פרויקט באסמבלי של תחרות כתיבה. המניע השני הוא שאני משתמש הרבה באתרים אחרים כמו type racer וכשאני משתמש בהם יותר ויותר אני רואה את כל הבעיות שיש שם וזה גרם לי לרצות להכין אתר מהסוג הזה ולנסות לפתור את הבעיות שמצאתי.

# דרישות פרויקט

## תיאור דרישות פונקציונליות:

המשתמשים באתר:

* שחקנים שרוצים לשפר את מהירות או איכות הכתיבה בשילוב של כיף ותחרות.
* המשתמשים יכול להירשם לאתר ולהיכנס לתחרויות כתיבה ולאחר מכן לראות בעמוד הפרופיל שלו אם הוא השתפר או לא עם הזמן.
* פרטי המשתמשים (שם וסיסמה) נשמרים ב- database על השרת.

התחרויות:

* למשתמשים מוצג עמוד של בחירת סוג התחרויות עם אופציות של תחרות ארוכה, קצרה וכמות משתתפים.
* יש גם את האופציה לשחק לבד.

דוחות שמוצגים למשתמשים:

* בעמוד פרופיל יש גרף שמראה את היסטורית התחרויות של המשתמש וממוצע המהירויות.

עיבוד מידע:

* המידע שמועבד בפרויקט הוא המידע שהמשתמש מקליד במשחק עצמו.
* המידע שהוא כותב נשלח לשרת ושם השרת שולח את המידע לכל שאר המשתמשים בחדר.
* שהשרת מזהה שאחד השחקנים סיים לכתוב את המשפט, הוא שומר את כמות הזמן שלקח למשתמש הזה לסיים את התחרות ובעזרת מספר המילים במשפט הוא יכול לחשב את מהירות הכתיבה ולשמור אותה בבסיס נתונים על השרת.
* השרת גם מחשב מחדש את המהירות הממוצעת של המשתמש ושומר אותה בשביל להציג אותה כל פעם שהמשתמש נכנס לעמוד הפרופיל שלו.

דרישות מערכת:

הפרויקט הוא אתר web שנמצא בhost של שרת בחול שהוא מקבל את כל הבקשות ומנהל את המשתמשים באתר.

# מה קיים בשוק

בשוק קיימים היום המון סוגים של אתרים שמלמדים את המשתמש איך לכתוב נכון ומהר ואתרים שמאפשרים לשחקנים להיכנס לתחרויות כתיבה שונות.

<https://play.typeracer.com/>

<https://typing.works/>

<https://www.typing.com/>

אחרי הרבה שעות ומאות תחרויות שהשתמשתי באתרים האלה אני הכנתי רשימה של כמה בעיות גדולות שיש באתרים האלה ובעזרתם הצלחתי ליצור אתר תחרויות כתיבה חדש.

* בחלק מהאתרים יש את האפשרות להתחרות מול אנשים אבל אין את האפשרות לראות את ההתקדמות של המשתמש. האתר שלי מראה למשתמש בעמוד הפרופיל שלו בדיוק כמה הוא טוב עכשיו ביחס לכמה הוא היה טוב פעם בעזרת גרף שמראה את היסטורית התחרויות של המשתמש.
* קרה לי המון פעמים באתרים הללו ובtype racer לרוב כאשר התחלתי תחרות כתיבה במטרה לכתוב משהו מהיר וללכת אבל כשלחצתי על הכפתור להתחיל תחרות, האתר הכניס אותי לתחרות כתיבה של 6 עד 8 שורות ארוכות.
* הבדל אחד גדול בין האתרים האלה לבין הפרויקט שלי הוא שבאתר שלי שילבתי את הבחירה של גודל התחרות. זאת אומרת שבאתר החדש שלי יש את האפשרות בעמוד הראשי לבחור את כמות המילים שיהיו בתחרות, וככה האתר יודע לאיזה לובי להכניס את המשתמש.

# טכנולוגיות בהן השתמשתי בפרויקט

הפרויקט שלי מחולק לשניים, החלק של השרת והחלק של הלקוח (האתר).

החלק של האתר בנוי בעזרת: html, css, javascript. השתמשתי בgoogle charts בשביל הגרף בעמוד הפרופיל של המשתמש ו bootstrap בשביל לאפשר למשתמש בטלפון להיכנס לאתר.

החלק של השרת בנוי בpython בעזרת flask וsqlite בשביל לנהל את הבסיס נתונים.

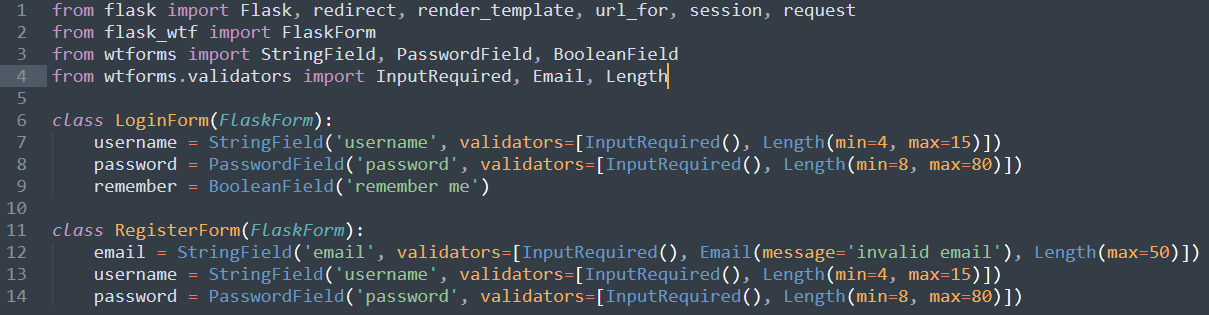
חשבתי להשתמש בdjango אבל ראיתי שהוא בנוי בשביל פרויקטים ענקים ומסובכים בזמן שflask היה מהיר פשוט והוא ענה על כל הדרישות של הפרויקט שלי כמו: שליחת עמודים, ניהול בסיס נתונים, ניהול עמוד רשימה וכניסה של משתמשים ושליחת הודעות מהירות בין השרת ללקוח בשביל לנהל הרבה משחקים שונים באותו הזמן. בשרת השתמשתי גם בwerkzeug בשביל להצפין את הסיסמאות של המשתמשים לפני שהשרת שומר אותם בבסיס נתונים.

# תיאור מבנה הפרויקט

לקוח

שרת

בסיס נתונים

Forms.py - 

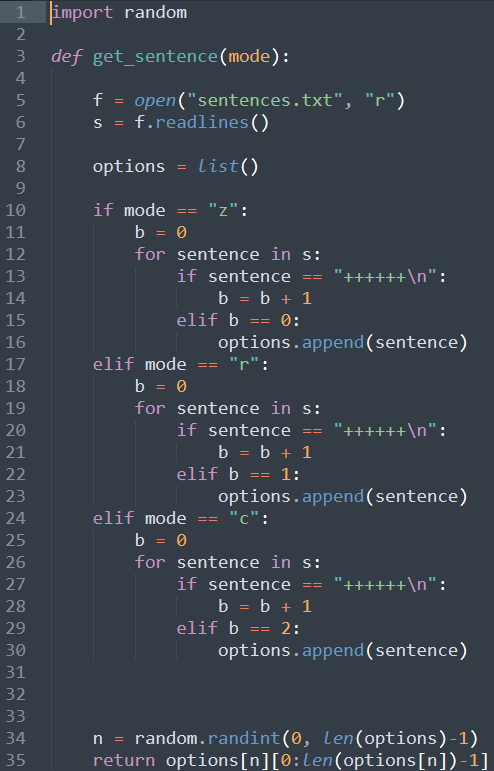
הקובץ הזה מקבץ את השורות קוד שאחראיות על הפורום באתר שבו הלקוח נרשם למשתמש קיים או יוצר משתמש חדש. הקובץ הראשי משתמש במודול הזה כאשר הוא שולח ללקוח את העמוד ההירשמות, השרת לוקח את המחלקות הללו, יוצר מהן אובייקט ושם נשמרות התשובות מהלקוח.

User.py –



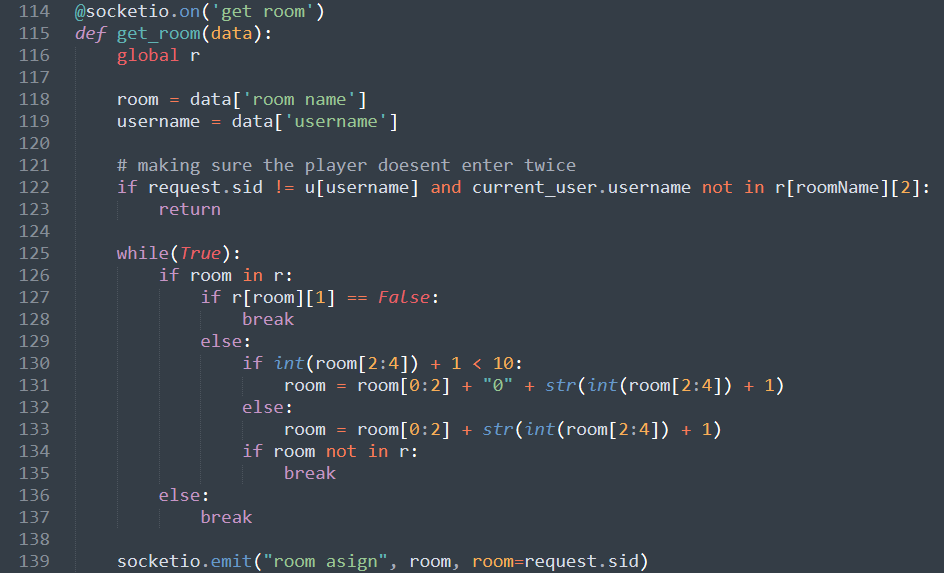
הקובץ הזה מאתחל את השרת הראשי, פה יש את הדברים החשובים שקשורים לתחילת השרת. יש את ההגדרות של השרת ויש גם את המחלקה של הuser, עם הרבה הגבלות על המשתמש, יש הגבלות כמו גודל השם משתמש וסיסמה ויש שמירת מקום למידע בעתיד למשתמש הזה כמו הציונים של המשחקים שלו.

Sentences.py –



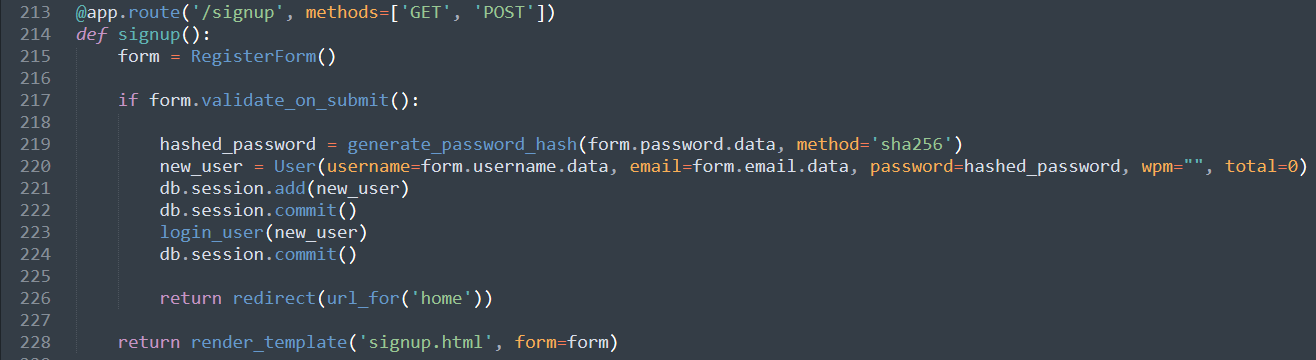
הקובץ הזה מנהל את המשפטים בתחרויות. הוא שולף את כל המשפטים מהקובץ sentences.txt, מחלק אותם לקבוצות גודל שלהם (קטן [{blitz}mode=”z”], בינוני [{rapid}mode=”r”] וגדול [{classical}mode=”c”]) ואחרי זה שולף משפט רנדומלי מהאורך שהשרת ביקש. השרת לוקח את המשפט שהוא קיבל ושולח אותו לכל המשתתפים בלובי ביחד עם השמות של כל המשתתפים.

Get\_room() –



הפונקציה הזאת נקראת כשהשרת מקבל בקשה מלקוח להיכנס לחדר ולהתחיל תחרות. הפונקציה מתחילה בלבדוק שהלקוח הזה כבר לא נמצא בחדר הזה למקרה שלקוח אחד מנסה להתחבר פעמיים לאותו חדר. אם הפונקציה לא הייתה בודקת את זה לקוח אחד היה יכול להיכנס לחדר עם עצמו. אחרי זה הפונקציה מחפשת חדר פתוח ופנוי בmode שהלקוח ביקש. השרת עובר על כל החדרים הפתוחים שלו ואם הוא לא מוצא אז הוא מכין חדר חדש ומגדיר אותו כפתוח ופנוי לשחקנים.

Signup() –



הפונקציה הזאת נקראת שהמשתמש מבקש להיכנס לעמוד שבו נרשמים. הפונקציה יוצרת פורום ושולחת אותו, לאחר מכן שהלקוח שולח את הפורום ונרשם הפונקציה הזאת נקראת שוב ונכנסת לif. שהוא נכנס לif הוא מצפין את הסיסמה של המשתמש בעזרת sha256. הסיבה שהשתמשתי באחד הזה זה כי לא משנה מה תהיה אורך הסיסמה שהלקוח מכניס, הסיסמה המוצפנת תמיד תהיה באורך 64 תווים וככה יותר קל לשמור אותם בבסיס נתונים. לאחר ההצפנה הפונקציה מכניסה את המשתמש החדש לבסיס נתונים, עושה login למשתמש ושולח את המשתמש לעמוד הראשי.

# מדריך למשתמש ותיאור ה-UX/UI למשתמשים

להיכנס לתחרות

להירשם כמשתמש חדש

עמוד ראשי

להיכנס לתרגול אישי

להיכנס לעמוד פרופיל

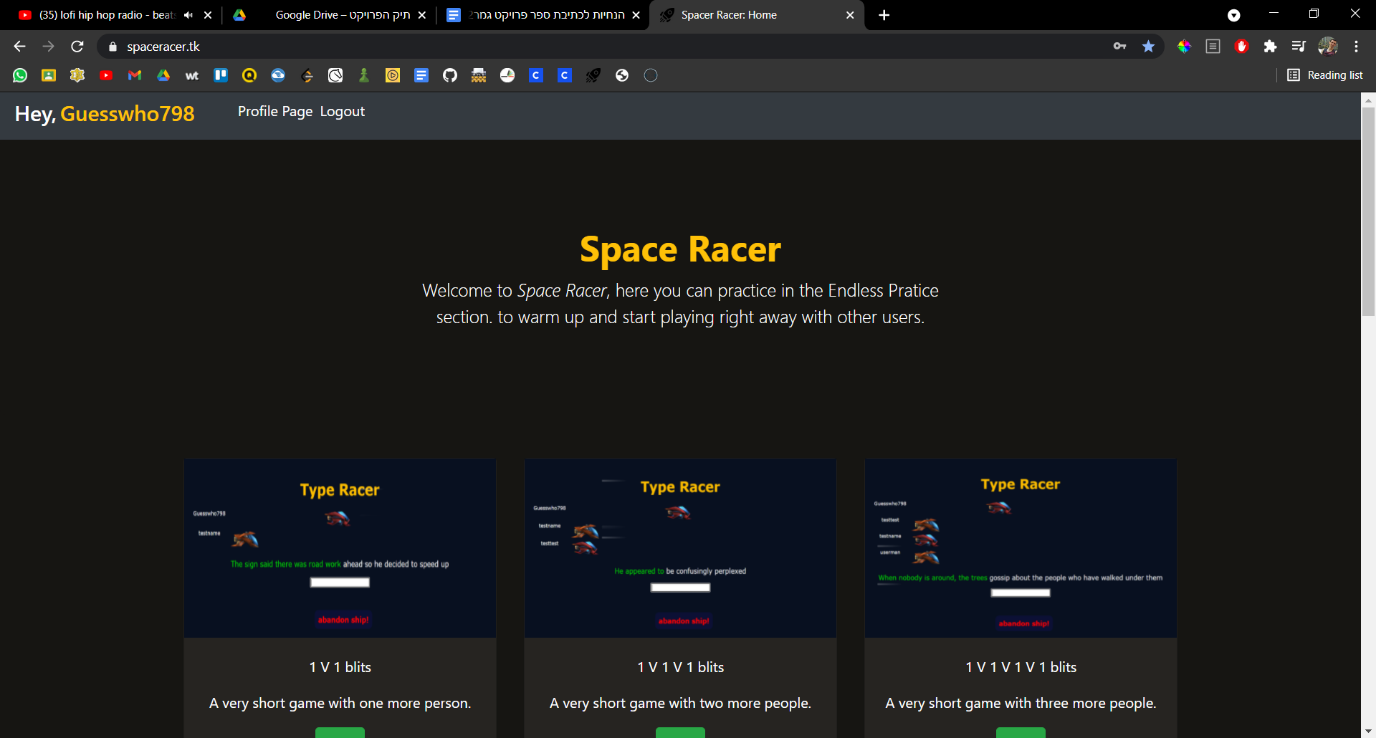
להיכנס כמשתמש רשום

## הדרכה למשתמש

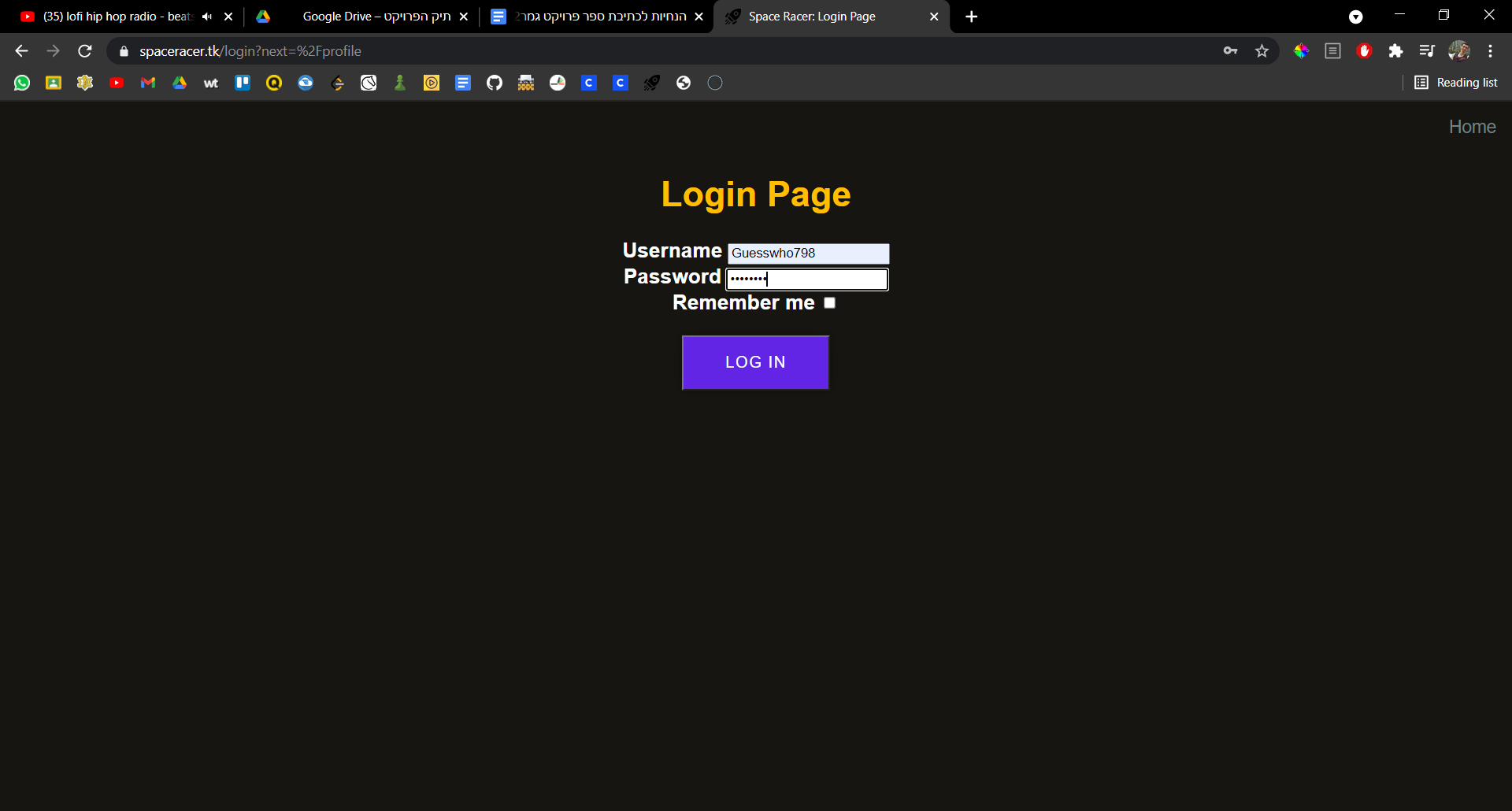
* בכניסה לאתר בפעם הראשונה יעלה העמוד הראשי שבו אפשר להיכנס לתחרויות שונות כמשתמש רשום (עד שהלקוח לא משתמש אי אפשר להיכנס לתחרות).
* בצד שמאל למעלה מוצג הכפתור signup, שלוחצים אליו הלקוח יועבר לעמוד בו הוא יוכל להירשם כמשתמש.
* אחרי שהלקוח נרשם הוא יכול לחזור לעמוד הראשי שבו הוא יוכל להיכנס לעמוד הפרופיל שלו בצד שמאל למעלה.
* בעמוד הפרופיל יש את השם של המשתמש, מהירות הכתיבה בממוצע, מספר התחרויות שהמשתמש שיחק, האם המשתמש רוצה לשמוע קולות שהוא עושה טעות כתיבה וגרף שמראה את מהירות הכתיבה לאורך התחרויות.
* אם חוזרים לעמוד הראשי וגוללים למטה אפשר לראות את לוח השחקנים המובילים עם התוצאות הכי גבוהות באתר ובאמצע המסך אפשר לראות המון כפתורים.
* על כל כפתור יש שני מושגים שמסבירים איזה תחרות זאת, אם רשום “1V1” אז מדובר בדו קרב של שחקן אחד נגד שחקן אחד, אם רשום “1v1v1” מדובר בשלושה שחקנים שמשחקים כולם נגד כולם ומנסים להגיע להיות מקום ראשון.
* הדבר השני שרשום על הכפתור זה “blitz”, “rapid” ו “classical”. זה האורך של המשפט שהלקוח יצטרך לכתוב, blitz זה תחרות מהירה עם משפט קצר, rapid זה תחרות יותר ארוכה עם 2 עד 3 שורות כתיבה וclassical זאת תחרות מאוד ארוכה עם 6 עד 9 שורות.

## תמונות:

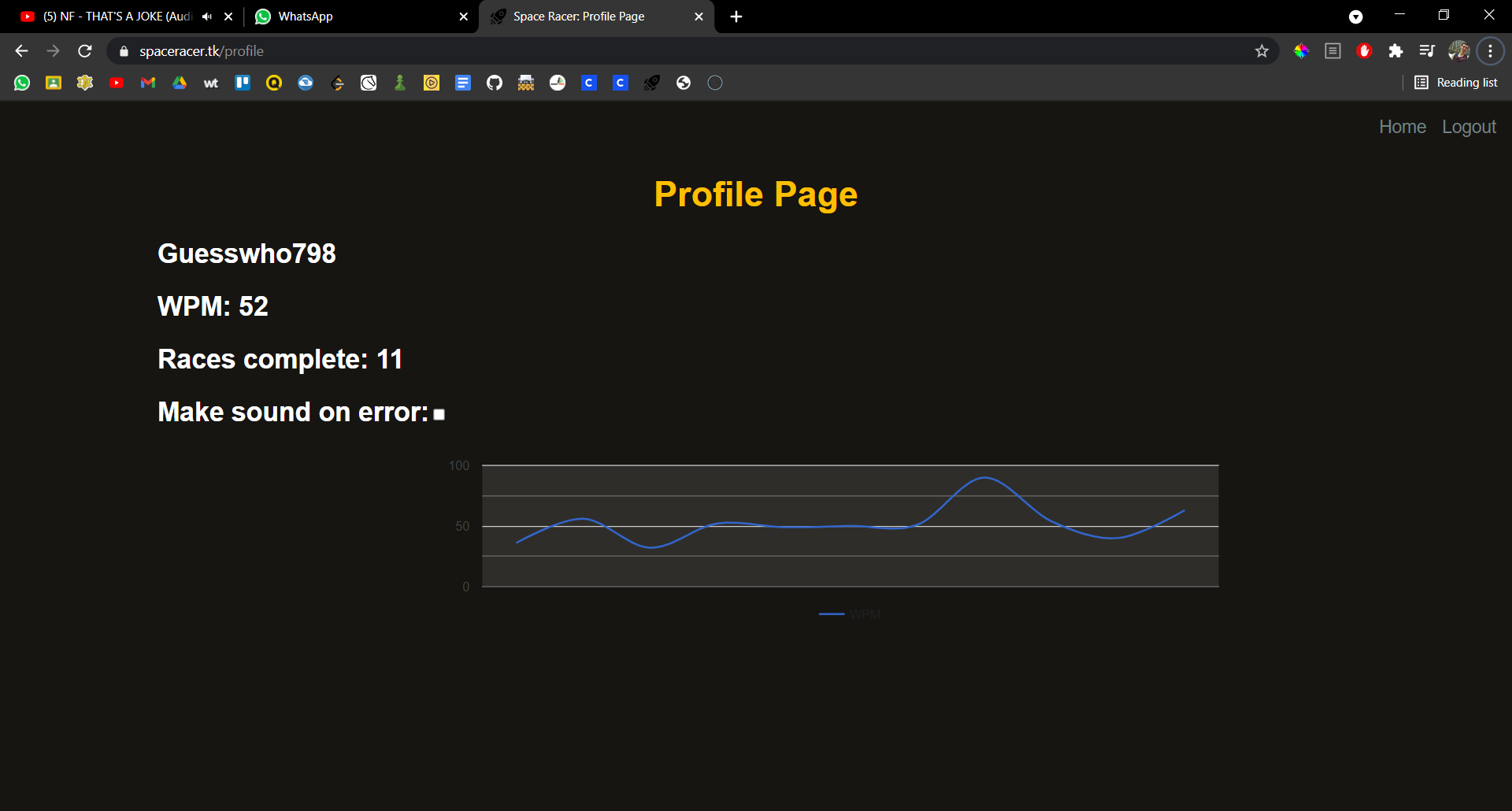
עמוד ראשי



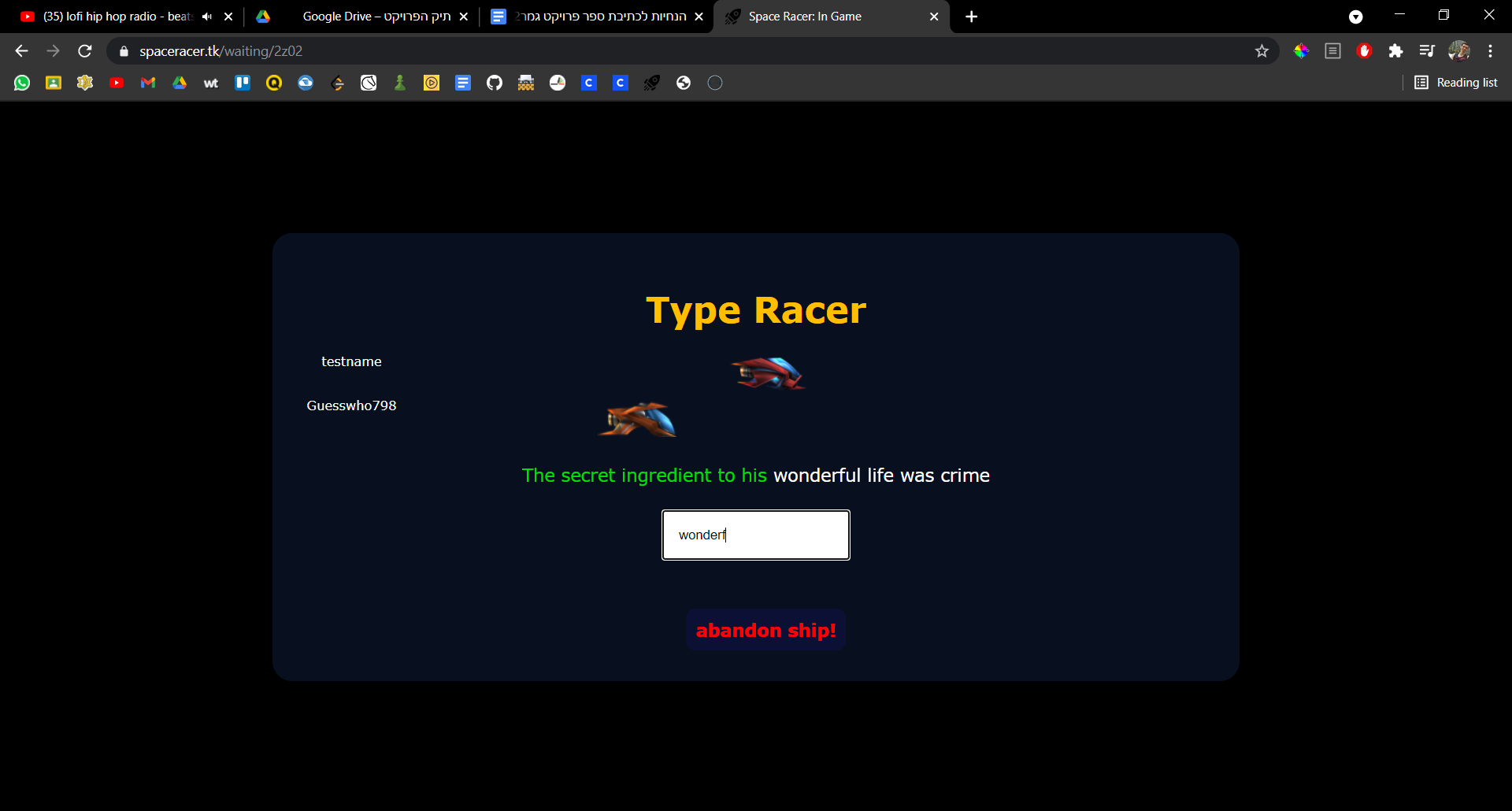
עמוד הירשמות



עמוד פרופיל



עמוד התחרות



# מדריך למפתח

## קבצים ומיקומם

הפרויקט עובד לפי מסגרת רשת של flask. בתיקיה של הפרויקט יש את הבסיס נתונים של השרת ביחד עם הקוד python. App.py זה הקובץ הראשי של השרת ובו נמצא הקוד שמנהל את הבקשות של הלקוח לדוגמה, signup, join lobby ובקשת עמודים שונים.

אחרי זה יש את התיקיה בשם static, היא שומרת את כל התמונות של השרת כמו האייקון של הפרויקט, התמונות של החלליות וכמה צילומי מסך בשביל הread me file ב Github. חוץ מהתמונות, התיקיה static שומרת גם את הקבצי css של האתר.

בתיקיה הראשית יש גם את התיקיה בשם template שהיא שומרת את כל הדפי html ומשם הקוד בשרת שולף את הדפים ויודע לשלוח אותם ללקוח.

## משתנים חשובים

משתנה חשוב שיש לי בשרת הוא מילון של מחרוזות במפתח (השם של החדר) לרשימה בערך (מספר האנשים בחדר, האם החדר מחפש שחקנים ושמות השחקנים). המילון הזה שומר את כל החדרים הפתוחים והמידע אליהם, הסיבה שהשתמשתי במילון היא כי מילון עושה הרבה תהליכים בשרת יותר פשוטים, לדוגמה, כשהלקוח מחפש חדר פנוי שהוא יכול להצטרף אליו, בשרת אפשר לבדוק אם החדר הזה קיים בשורה אחת פשוטה.

# היבטי אבטחת מידע

* ברישום ובכניסה למערכת המשתמש שולח את שם המשתמש והסיסמה לשרת, השרת לוקח את הסיסמה שהכניס המשתמש, מצפין אותה בעזרת הצפנת sh256, וככה מכניס את הנתונים לבסיס נתונים.
* כאשר המשתמש מנסה להיכנס למשתמש קיים הוא שולח את השם משתמש וסיסמה לשרת, שם הוא מחפש משתמש עם השם משתמש שהוא קיבל ומשווה בין הסיסמה שהלקוח הכניס אחרי הצפנה לבין הסיסמה שקיימת בבסיס נתונים תחת השם משתמש שהלקוח ביקש.
* הצפנת sha256 היא הצפנה מהירה חד כיוונית. הצפנה חד כיוונית אומרת שאפשר לקחת טקסט ולהצפין אותו אבל אי אפשר לקחת את הסיסמה המוצפנת ולפענח אותה.

# מחקר משמעותי/נושאים שלמדתי

Flask:

* שהתחלתי לעבוד על הפרויקט לא יודעתי מה זה flask ולכן חיפשתי בתיעוד את כל המידע שהייתי צריך.
* אחרי כמה זמן שעלו לי הרבה שאלות שהתיעוד לא היה יכול לענות אליהם עברתי ליוטיוב ושם היו כל התשובות לשאלות שלי ומשם הצלחתי לבנות את השרת מאפס.

Javascript:

* לפני שהתחלתי את הפרויקט ידעתי מה זה javascript וכמה דברים קטנים בו אבל בפרויקט הזה השתמשתי בו בפעם הראשונה.
* גיליתי איך לעשות דברים כמו לשנות ערכים של אלמנטים בזמן אמת, פונקציות שרצות כל זמן קבוע ועוד הרבה דברים.

Sqlite:

* אף פעם לא השתמשתי בבסיסי נתונים לפני הפרויקט הזה ובשביל לשמור את הנתונים שלי ולנהל אותם השתמשתי בsqlite3.
* למדתי איך ליצור בסיס נתונים, להכניס מידע ולשלוף אותו לפי סדר מסוים.

# בעיות שנתקלתי במהלך פיתוח הפרויקט

בעיית החדרים הפתוחים:

* בתחילת הדרך, כל פעם שמשתמש רצה להיכנס לתחרות, השרת הכניס את המשתמש הזה ל"חדר".
* חדר זה דרך לשלוח הודעות broadcast לאנשים ספציפיים.
* הבעיה שלי הייתה שאני אף פעם לא סגרתי אותם ולכן משתמשים שונים היו נמשכים לחדרים שהם פעם ביקרו בהם כשחדר חדש היה צריך להיפתח.
* הפתרון של הבעיה הייתה לא רק לגלות מתי אפשר לסגור את החדר אלה גם להכין מנגנון חדש שידע לשייך משתמשים לחדרים חדשים כאשר הלובי שהמשתמשים רוצים להתחבר אליו נמצא בשימוש.
* המנגנון החדש יודע לעבור על החדרים, לזהות אם יש חדר פנוי ואם אין אז לפתוח אחד חדש ולתת לו index חדש שיבדיל אותו מהחדרים האחרים.

שמירת המשפטים:

* האתר צריך לשמור הרבה משפטים כדי שהמשתמשים לא יכתבו את אותם משפטים שוב ושוב.
* הבעיה שלי הייתה ששמרתי את כל המשפטים hard coded בתוך הקובץ של השרת עם מספר רנדומלי שהייתי צריך לשנות את המקסימום שלו כל פעם שהכנסתי משפט חדש.
* עוד בעיה היא שבעתיד אם מישהו אחרי ירצה להוסיף עוד משפטים, הוא יצטרך לדעת לתכנת והוא יצטרך לשנות כמה דברים רק בשביל להוסיף משפט אחד.
* הפתרון שלי היה קודם כל להפריד את החלק הזה מהקובץ הראשי לתוך module של עצמו, וגם לשנות את הקוד שהוא רק יקרא קובץ ויוכל לבחור משם לבד משפט אקראי מהקטגוריה שהמשתמש בחר.
* הפתרון הזה טוב בגלל שאם מישהו רוצה להכניס עוד משפטים עכשיו הוא רק צריך להכניס אותו לקובץ, בלי להיכנס לקוד ולשנות כמה משתנים ולהוסיף תנאים.

# תיאור שלבי העבודה שעשיתי במהלך הפרויקט

## תכנון מקורי:

בתכנון הראשוני שלי האתר היה רק אתר כתיבה רגיל, בלי משתמשים ובלי אופציות למשחקים שונים.

## מה השתנה במהלך הפיתוח:

אחרי שהתקדמתי בהכנת הפרויקט התחלתי להוסיף עוד ועוד אופציות של משחקים בין אנשים. לאחר מכן הוספתי משתמשים ובסיס נתונים לשמור את המידע של כל המשתמשים. בשלב הזה גם החלפתי את הנושא של הפרויקט לתחרויות של חלליות ושיניתי את המכוניות לחלליות.

היו עוד הרבה בעיות קטנות באתר ששמתי לב אליהם רק שנתתי לאנשים אחרים להשתמש באתר שלי בלי להסביר להם איך דברים עובדים, לראות אם האתר שלי מצליח להעביר את המידע ללקוח בצורה נוחה ואינטואיטיבית.

לדוגמה:

* ראיתי שאנשים לא שמים לב שהם עושים טעויות בכתיבה אז הוספתי את האפשרות לשמוע אם הלקוח עשה טעות כתיבה, להשמיע ביפ שיראה שצריך לחזור אחורה ולתקן את הטעות.
* ראיתי שחלק מהאנשים לא אהבו את הרעיון של הביפ כשהם עושים טעות אז הוספתי בעמוד פרופיל מקום לסגור או לפתוח את האפשרות הזאת, הוספתי עמודה בבסיס נתונים ששומרת לכל לקוח את הבחירה שלו.
* הוספתי שבסוף כל משחק רואים מה הייתה המהירות של היריבים לאחר התחרות.

## בדיקות שעשיתי:

* נכנסתי לעמוד פרופיל עם משתמש חדש לפני שעשיתי תחרות אחת. הבדיקה הזאת הקריסה את התוכנה בגלל שבעמוד פרופיל יש את הגרף שמראה את ההיסטוריות כתיבה של המשתמש וכאשר המשתמש פותח את העמוד וטוען את הגרף, הוא שולח לגרף 0 נקודות והגרף מחזיר שגיאה.
* כשמשתמש נכנס למשחק והוא מועבר ללובי. שם הוא יכול לטעון מחדש את העמוד שלו ובכך להתחבר שוב לאותו לובי. פתרתי את הבעיה הזאת בעזרת שמירת השמות של האנשים בלובי. ככה כל פעם שמישהו רוצה להתחבר ללובי מסוים הלובי משווה את השם של המשתמש שרוצה להיכנס ללובי לרשימה שיש לו.

## באגים פתוחים / בעיות קיימות + מגבלות שקיימות:

האתר לא בנוי לטלפון - האתר עצמו בנוי על bootstrap שעובד עם טלפונים ויכול לשנות את איך שהעמוד נראה לפי הגודל של המסך, אפשר אפילו להירשם לאתר או להיכנס עם משתמש רשום ולראות את עמוד הפרופיל או לוח השחקנים הטובים ביותר. הבעיה היא ששחקן נכנס למשחק הוא צריך לשלוח לשרת הודעות מהירות בעזרת socket-io שלא תומך בטלפונים ולכן השחקן לא יכול לשחק בטלפון.

# רפלקציה – מבט אישי על העבודה ועל תהליך פיתוח הפרויקט

איך הייתה העבודה עבורי:

העבודה הייתה עבורי חוויה לימודים ארוכה ומשמעותית. הפרויקט הזה נתן לי את האפשרות ללמוד הרבה דברים חדשים ביחד עם אנשים אחרים. \*צריך להוסיף עוד\*