בית חינוך ע"ש הרצוג כפר-סבא

שרת משחקים

שם: עידן ינאי – 214293300

שם המנחה – יוסי זהבי

תאריך הגשה -

תוכן עניינים

מבוא

תיאור תכולת הספר:

- מבוא
- ארכיטקטורה
- מדריך למשתמש
- מדריך למפתח
- רפלקציה
- ביבליוגרפיה

הרקע לפרויקט

עולם המחשבים התפתח מאוד בשנים האחרונות, דבר שגרם להתפתחותם של החנויות הדיגיטליות המוכרות לנו כיום. בעבר היינו יוצאים מהבית כדי לרכוש משהו בחנות, אמנם כיום ניתן לרכוש כמעט כל דבר מהאינטרנט. לכל חנות/חברה המוכרת מוצר כלשהו, יש חנות דיגיטלית שממנה ניתן להזמין הביתה מוצרים. היתרון של חנויות דיגיטליות הוא שהם הרבה יותר נוחות למשתמש, בקושי צריך לעשות משהו והמוצר כבר אצלך בבית. דוגמאות לחנויות דיגיטליות: Amazon, Ebay, AliExpress, שופרסל-Online, ועוד... חנויות דיגיטליות גם השפיעו על עולם המשחקים, וכיום הדרך המקובלת לרכוש משחק היא דרך האינטרנט.

תהליך המחקר

באמצעות החנויות הדיגיטליות ניתן לשחק Online באמצעות שרת משחקים המקושר לחנות. בגדול, קיימות המון חנויות משחקים דיגיטליות כיום אבל, ההשראה המרכזית של פרויקט זה היא חנות הנקראת Steam הנוצרה על ידי Valve Corporation. Steam היא חנות המשחקים הדיגיטלית הכי פופולארית ומצליחה שקיימת, וכל מי שהוריד משחק אי פעם כנראה שמע/השתמש בה. נכון ל-2019 ל-Steam יש יותר מ-34,000 משחקים להורדה, ויותר מ-95 מיליון משתמשים חודשיים אקטיבים. בנוסף להורדת משחקים, ניתן גם לראות סטטיסטיקות על הרגלי המשחק שלך. זאת ועוד, קיים גם אזור קהילתי בו ניתן לפרסם מדריכים, תמונות, סיפורים בנושא משחקים שונים. בסך הכל Steam מספקת כל מה שגיימר צריך, משחקים וקהילה, והכל בממשק נוח שקל מאוד להבנה.

סקירת המצב הקיים בשוק

בנוסף ל-Steam קיימות גם חנויות דיגיטליות נוספות כגון: Origin, Epic Games, Uplay, AppStore, Google Play. כל אחת מהם עם החידושים והשינויים שלה. למשל: Google Play, AppStore הן חנויות משחקים לטלפון, GooglePlay בשביל Android, ו-AppStore בשביל IPhone. ההבדל העיקרי בין Origin, Epic Games, Uplay, Steam הוא המשחקים שהם מציעים למשתמש והקהילה שלהם.

אתגרים מרכזיים

אתגרים מרכזיים איתם אאלץ להתמודד הם: למידה של P2P והגנה עליו. הצפנת RSA. הורדות בשלבים. אבטחת מידע. וכמובן, עמידה בזמנים.

מוטיבציה לעבודה

הסיבה שבחרתי בנושא זה היא כי אני משתמש הרבה ב-Steam והנושאים שאני צריך ללמוד בשביל ליישם את הפרויקט, מעניינים אותי ונשמעים מאתגרים מספיק בשביל שאני באמת אוכל להגיד שזה הפרויקט שלי.

מבנה/ארכיטקטורה של הפרויקט

ארכיטקטורת רשת - פרוטוקול

מטרת השירות: מתן אפשרות להורדת משחקים ולשחק/להתכתב עם חברים.

סוגי הודעות:

לקוח לשרת

סוג הודעה	פרוט	פרמטרים
LOGIN	בקשת התחברות	שם משתמש, סיסמא
REGIS	בקשת הרשמה	שם משתמש, סיסמא, מייל, כינוי
DOWNL	בקשת הורדת משחק	ID של המשחק
MSGFR	בקשת שליחת הודעה לחבר	ID של החבר, מה לשלוח
INVIT	הזמנת חבר למשחק	ID של החבר, ID של המשחק
FREQU	שליחת בקשת חברות	ID של המשתמש
PURCH	בקשת רכישת משחק	ID של המשחק
MONEY	הוספת כסף לחשבון	פרטי תשלום, סכום כסף
DISCO	בקשת התנתקות מהשרת	-

שרת ללקוח

סוג הודעה	פרוט	פרמטרים
LOGRE	שאילת הרשמה או התחברות	-
ACKNW	Acknowledge	-
FILES	שליחת קבצים בזמן הורדה	Path
ERROR	שגיאה	מספר שגיאה + פירוט
PINVI	התראת הזמנה למשחק מחבר אחר	IP של החבר שהזמין, פורט
START	שליחה של הטוקן לשני השחקנים	Token, Expire Time
UINFO	שליחת פרטי המשתמש ללקוח שהתחבר	רשימת משחקים, רשימת חברים, כסף נוכחי.
FRMSG	שליחה של הודעה לחבר	מה שהוא שלח

מבנה הודעות:

5 תווים ראשונים אורך ההודעה. אחרי זה תו מפריד | ,אחר כך סוג ההודעה, רווח, ובסוף פרמטרים אם צריך (רווח בין כל פרמטר). ייצוג מחרוזתי.

דוגמא להודעה – 00018|LOGIN IdanY 123456 (הודעת התחברות מלקוח לשרת)

<u>צורת מענה - סינכרונית:</u> לאחר כל בקשה של הלקוח תישלח תשובה של השרת.

<u>שגיאות:</u>

מספר שגיאה	פרוט
1	לקוח התנתק בצורה לא צפויה
2	פרמטרים של ההודעה לא מולאו
3	פרמטרים של ההודעה לא תקינים
4	לקוח כבר מחובר
5	משתמש זה כבר קיים

מודולים

Pickle – בשביל להעביר מידע בצורה נוחה יותר לאחר שאילתה של ה-DB.

Socket – בשביל תקשורת נוחה.

Hashlib – בשביל בדיקה פשוט של Checksum במקרה הצורך.

Crypto – בשביל הצפנת ה-RSA, הפענוח שלה וייצור מפתחות.

מחלקות

<u>צד לקוח:</u>

Screen: אחראית על ה-User Interface.

פעולות המחלקה:

update_screen – מעדכנת את המסך.

get_command – מקבלת מיקום של לחיצת העכבר של המשתמש וממירה אותה לפקודה ומחזירה.

Chat: אחראית על ההודעות המתקבלות על החזרתן, ועל משתנים חשובים.

תכונות המחלקה:

user - שומר את הנתונים של המשתמש.

friend_list – רשימת החברים של המשתמש.

game_list – רשימת המשחקים הקיימים.

games_owned – רשימת המשחקים שהמשתמש רכש.

chats – מילון ששומר את הצ'אטים של המשתמש בתור מפתח שהוא ID של המשתמשים וערך שהוא רשימה עם ההודעות והכיוונים שלהם (to – T ,from – F).

{ FriendID : [F:waddup?,T:im fine man] }

פעולות המחלקה:

encrypt_password – מצפינה את הסיסמא של המשתמש בזמן הרשמה או התחברות. מקבל הודעה לשרת עם סיסמא לא מוצפנת ומחזירה את אותה הודעה עם הסיסמא המוצפנת.

handle_message: קוראת לפעולות/מחלקות מתאימות לפי ההודעות המתקבלות. מקבלת הודעה. מחזירה הודעה בחזרה לשליחה במידת הצורך.

P2P: אחראית על תקשורת P2P במידת הצורך.

תכונות המחלקה:

Port – פורט התקשורת.

Token – הטוקן שנשלח מהשרת.

Expire – זמן תפוגת השרת במידה ולא נוצר חיבור.

פעולות המחלקה:

create_server – יוצרת שרת UDP עם המשחק הרצוי ומנהלת אותו. מקבלת Game ID.

create_client – יוצרת לקוח UDP עם המשחק הרצוי ומנהלת אותו. מקבלת Game ID.

<u>צד שרת:</u>

DataBase: אחראית על הוצאה והכנסה של נתונים מבסיס הנתונים.

פעולות המחלקה:

FriendList(UserID) – מקבלת ID של משתמש ומחזירה את רשימת החברים שלו.

SELECT FriendID FROM Friends WHERE UserID = UserID;

GamesList(UserID) – מקבלת ID של משתמש ומחזירה את רשימת המשחקים שלו.

SELECT GameID FROM GamesOwned WHERE UserID = UserID;

GetPassword(UserID) – מקבל ID של משתמש ומחזירה את הסיסמא שלו(ערך גיבוב) למען בדיקה.

SELECT Password FROM Users WHERE ID = UserID;

EmailExist(Email) – מקבל מייל ובודק האם הוא קיים במערכת. אם הוא קיים מחזיר True ולא יחזיר False. *SELECT Email FROM Users WHERE Email = Email (אם נמצא מייל אז True)*

AcceptFriendRQ(Sender, Receiver) – מקבל ID של השולח והמקבל, מוסיף אותם לרשימת החברים, ומוחק את הבקשה.

INSERT INTO Friends (UserID, FriendID) VALUES (Sender, Receiver);

INSERT INTO Friends (UserID, FriendID) VALUES (Receiver, Sender);

DELETE FROM FriendRequests WHERE Sender = Sender AND Receiver = Receiver;

DenyFriendRQ(Sender, Receiver) – מקבל ID של השולח והמקבל ומוחק את הבקשה.

DELETE FROM FriendRequests WHERE Sender = Sender AND Receiver = Receiver;

NickToID(Nick) – מקבל כינוי של משתמש ומחזיר את ה-ID של המשתמש.

SELECT ID FROM Users WHERE Nickname = Nick;

AddCash(UserID, Amount) – מקבל ID של המשתמש וסכום כסף להוסיף לחשבון שלו.

UPDATE Users SET Money = (Money + Amount) WHERE ID = UserID

Purchase(UserID, GameID) – מקבל ID של המשתמש ושל והמשחק הרצוי. בודק האם למשתמש יש מספיק כסף לרכוש את המשחק. אם כן, מוריד את הכסף של המשתמש לפי המחיר, ומוסיף את המשחק לבעלות המשתמש.

SELECT Price FROM Games WHERE ID = GameID;

SELECT Money FROM Users WHERE ID = UserID;

UPDATE Users SET Money = Money – Price WHERE ID = UserID;

INSERT INTO GamesOwned (UserID, GameID) VALUES (UserID, GameID);

Manager: אחראית על הודעות המתקבלות ועל שליחה חזרה.

תכונות המחלקה:

db – עצם מסוג DataBase.

Enc – עצם מסוג RSA.

User – עצם מסוג User.

פעולות המחלקה:

handle_message: קוראת לפעולות/מחלקות מתאימות לפי ההודעות המתקבלות. מקבלת הודעה אחרי הפירוק שלה. מחזירה הודעה בחזרה לשליחה במידת הצורך.

שני הצדדים:

פרוטוקול: אחראית על מבנה ההודעות שנשלחות ועל פענוח הודעות מתקבלות.

פעולות המחלקה:

send_with_size – הפעולה מקבלת הודעה לשליחה, ומוסיפה בתחילתה 6 תווים. 5 תווים ראשונים הם אורך ההודעה, והתו השישי הוא תו מפריד |. לאחר מכן היא שולחת את ההודעה.

recv_by_size – הפעולה מקבלת הודעה עד לסופה (לפי אורך ההודעה שכתוב בהתחלה) ומחזירה את ההודעה ללא האורך.

check_message – הפעולה מקבלת את ההודעה לאחר הפעולה recv_by_size בודק את סוג ההודעה והאם הפרמטרים מולאו/תקינים. מחזיר מספר שגיאה ופירוט במידת הצורך.

MsgRSA: אחראית על שליחה וקבלה של סיסמאות בצורה בטוחה.

תכונות המחלקה:

PKey – מפתח ציבורי.

SKey – מפתח פרטי/סודי.

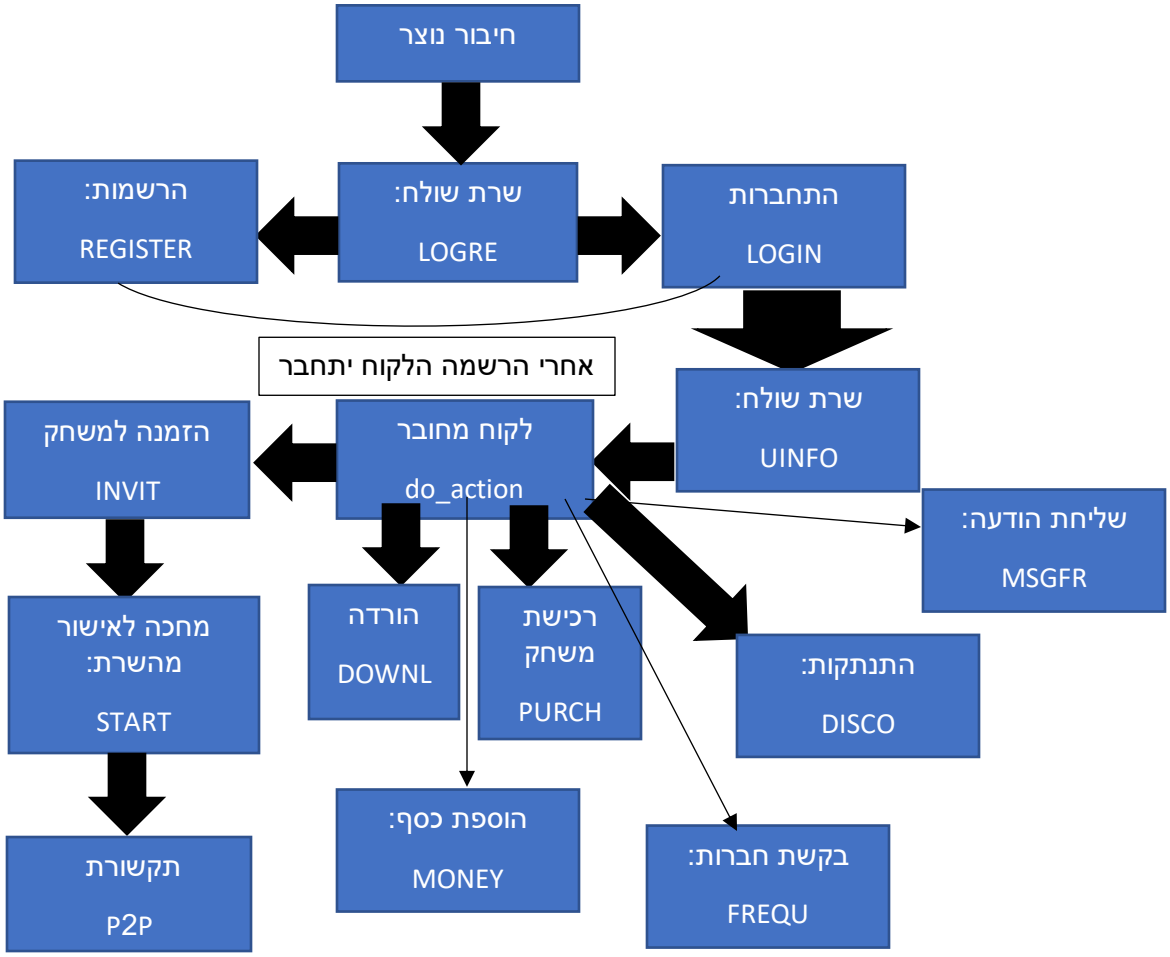
פעולות המחלקה:

rsa_encrypt – הפעולה מקבלת הודעת התחברות/הרשמה לשליחה, ואת המפתח הציבורי של השרת. הפעולה מחזירה את ההודעה לאחר הצפנתה באמצעות RSA.

rsa_decrypt – הפעולה מקבלת הודעה שהצפינו אותה באמצעות RSA ומקבלת מפתח פרטי הקשור למפתח הציבורי איתו הצפינו את ההודעה. הפעולה מחזירה את ההודעה לאחר הפענוח שלה.

לולאות

לולאה ראשית – לקוח



הצפנות

הצפנת RSA

RSA היא מערכת הצפנת מפתח ציבורי הנעשית בשימוש נרחב במערכות אבטחת מידע מודרניות, תקשורת מחשבים ומסחר אלקטרוני. הצפנת RSA היא הצפנה אסימטרית, כלומר מפתח הפענוח שונה ממפתח ההצפנה. מפתח ההצפנה הוא מפתח ציבורי ומפתח הפענוח הוא מפתח סודי. את המפתח הציבורי ניתן לבחור והוא גלוי לכולם, בעוד שהמפתח הסודי תלוי במפתח הציבורי ורק הנמען יודע אותו. ב-RSA המוען משתמש במפתח ההצפנה הציבורי של הנמען כדי להצפין עבורו מסר כך שרק הנמען מסוגל לפענחו באמצעות המפתח הפרטי המתאים שברשותו.

הכנה

תחילה הנמען בוחר שני מספרים ראשוניים איתם הוא מחשב את המספר N שהוא המכפלה שלהם. בנוסף, הוא בוחר מספר E שהוא מספר אי זוגי והוא זר ל-N (כלומר אין מספר המתחלק בשניהם הגדול מאחד). המספרים N ו-E הם המפתח הציבורי. איתם מחשבים את D שהוא המפתח הסודי. המוען משתמש במפתחות הציבוריים N ו -E ואיתם מצפין את ההודעה שהוא רוצה להעביר. הנמען משתמש ב-D בשביל לפענח את ההודעה.

החוזק של RSA

מה שהופך את RSA להצפנה חזקה הוא השימוש שלה במספרים ראשוניים. בשביל להגיע למפתח הסודי D משתמשים בפונקציית אוילר על המספר N, פונקציה זאתי קלה לחישוב אם יודעים את שני המספרים הראשוניים המרכיבים אותו. מצד שני, מאוד קשה ואף בלתי אפשרי, להגיע לשני המספרים הראשוניים המרכיבים את N אם בוחרים שני מספרים גדולים במיוחד. לכן, אי אפשר להגיע לחשב את פונקציית אוילר של N, ומכאן לא ניתן להגיע ל-D שהוא המפתח הסודי.

השימוש בפרויקט

בפרויקט אעשה שימוש בהצפנת RSA על מנת שמשתמשים יוכלו להירשם ולהתחבר בצורה בטוחה. ובכך לשמור על הסיסמאות והפרטים האישיים של המשתמשים מידי אנשים אחרים.

פונקציית גיבוב - Hash

פונקציית Hash או פונקציית גיבוב היא קריפטוגרפית פונקציה חד כיוונית שממירה קלט באורך משתנה לפלט באורך קבוע, ובדרך כלל קצר בהרבה הנקרא ערך גיבוב. פונקציה חד כיוונית אומרת כי בהינתן פלט מסוים לא ניתן למצוא את הקלט המקורי שלו. בפונקציית גיבוב קריפטוגרפית לכל קלט אפשרי יש פלט ייחודי רק לו, ובזה הפונקציה נמדדת. יש כמה סוגים של פונקציות Hash – MD5,SHA,Skein ועוד... בפרויקט אעשה שימוש ב-SHA3.

השימוש בפרויקט

לפונקציית הגיבוב יש כמה שימושים, כגון: הבטחת שלמות של הודעה, חתימה דיגיטלית על קבצים, והשימוש שאני אעשה בפרויקט – הגנה על סיסמאות. בפרויקט, הפרטים האישיים של המשתמשים, כולל הסיסמאות נמצאים ב-DataBase. לכן כדי למנוע גניבה של סיסמאות במקרה של פריצה ל-DataBase אני אשמור את ערך הגיבוב של הסיסמאות. בצורה זו לא ניתן להגיע לסיסמאות המקוריות של המשתמשים, ועדיין ניתן לבדוק אם משתמש המתחבר לשרת הכניס את הסיסמא הנכונה.

בסיס הנתונים

בשביל בסיס הנתונים בשרת אני משתמש בכמה טבלאות SQL:

Users

Column	Type	Other
ID	Integer	Primary Key
Username	Text	Unique
Password	Text	
Nickname	Text	Unique
Email	Text	Unique
Money	Real	Default 0

טבלה זו אחראית על הפרטים האישיים של המשתמשים.

Games

Column	Type	Other
ID	Integer	Primary Key
Name	Text	Unique
Path	Text	Unique
Price	Real	
Description	Text	
Release Date	Text	Date

טבלה זו אחראית על המשחקים המוצעים למכירה.

GamesOwned

Column	Type	Other
UserID	Integer	Foreign Key
GameID	Integer	Foreign Key

טבלה זו אחראית על המשחקים של כל אחד מהמשתמשים.

Friends

Column	Type	Other
UserID	Integer	Foreign Key
FriendID	Integer	Foreign Key

טבלה זו אחראית על החברים של כל אחד מהמשתמשים.

FriendRequests

Column	Type	Other
Sender	Integer	Foreign Key (UserID)
Receiver	Integer	Foreign Key (UserID)

טבלה זו אחראית על בקשות החברות הנשלחות ממשתמש למשתמש.

מדריך למשתמש

הוראות התקנה

ראשית, יש להתקין Python 3 ו-Python 2. שנית, יש להתקין את המודולים הבאים לכל גרסה בהתאם:

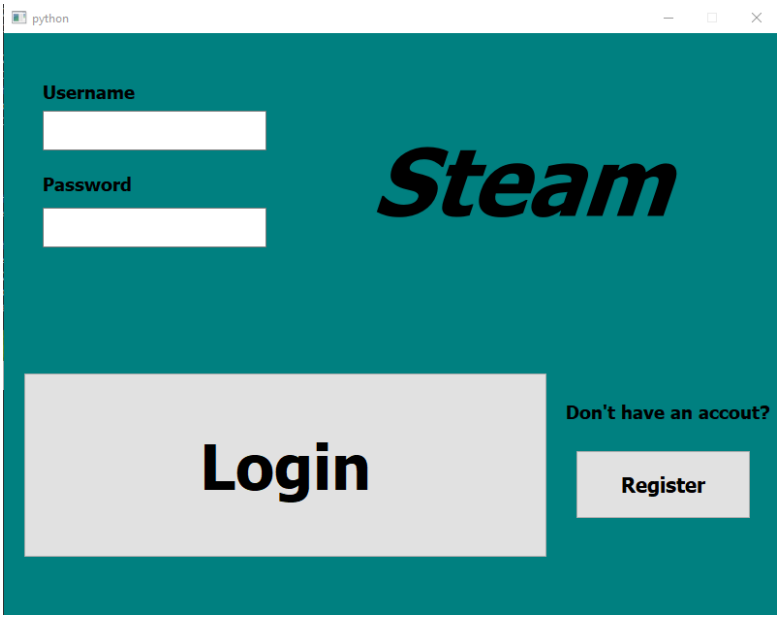
Python3	Python2
pycryptodome	pygame
PyQt5	
bcrypt	

יש לוודא שהקבצים הבאים קיימים:

קבצי לקוח	קבצי שרת
Client.py	Server.py
Chat.py	Manager.py
Screen.py	Database.py
RSA.py	RSA.py
Protocol.py	Protocol.py
uis (תיקיה עם ארבעה קבצים: chat/login/main/register.ui)	Server Games (תיקיה המכילה תיקיות של משחקים)
Games (תיקיה שנוצרת אוטומטית)	Database.db

סוגי מסכים

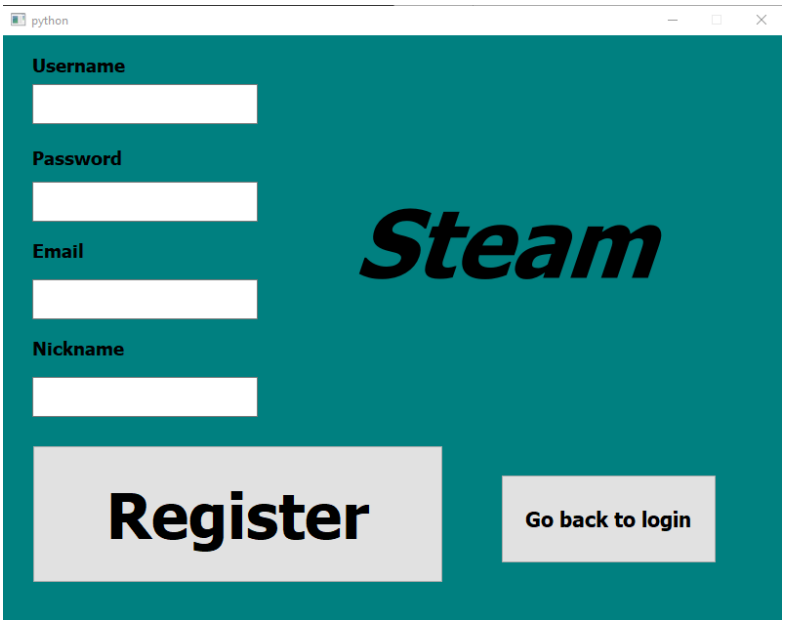
מסך Login – התחברות (מסך פתיחה)



במידה וללקוח יש משתמש קיים הוא יקליד את שם המשתמש והסיסמא שלו וילחץ על Login כדי להתחבר. אם ללקוח אין משתמש הוא ילחץ על כפתור Register כדי לעבור למסך הרשמה.

הודעות: שם משתמש או סיסמא לא נכונים.

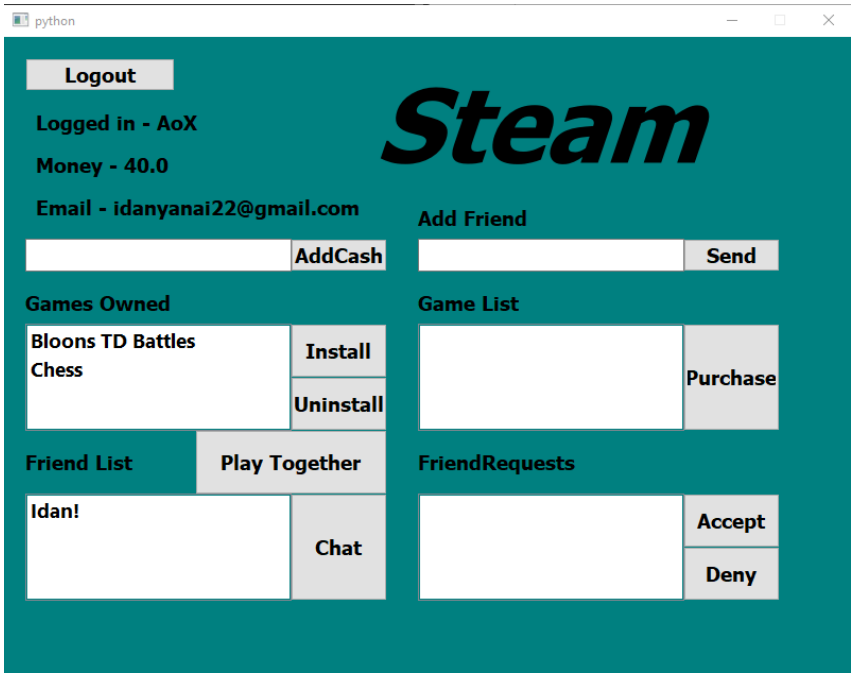
מסך Register – הרשמה



במידה וללקוח אין משתמש הוא ירשם במסך זה, באמצעות שם משתמש, סיסמא, מייל, כינוי ולחיצה על כפתור Register. לאחר ההרשמה הוא יעבור מיד למסך הראשי. הכפתור האחר מחזיר למסך Login.

הודעות: שם משתמש\מייל\כינוי תפוס.

מסך ראשי



המסך הראשי של הפרויקט לאחר ההתחברות. כפתורים:

AddCash – באזור משמאל מקלידים סכום כסף להוסיף לחשבון ולוחצים על הכפתור.

Send – באזור משמאל מקלידים כינוי של משתמש לצרף לחברים ולוחצים על הכפתור.

Purchase – לוחצים על המשחק הרצוי מרשימת המשחקים ולוחצים ועל הכפתור.

Accept/Deny – לוחצים על משתמש שרוצים לאשר\לדחות את הצעת החברות שלו מהרשימה משמאל ולוחצים על אחד הכפתור הרצוי.

Install/Uninstall – לוחצים על המשחק הרצוי מרשימת המשחקים בבעלותך שרוצים להוריד\למחוק ולוחצים על אחד הכפתורים.

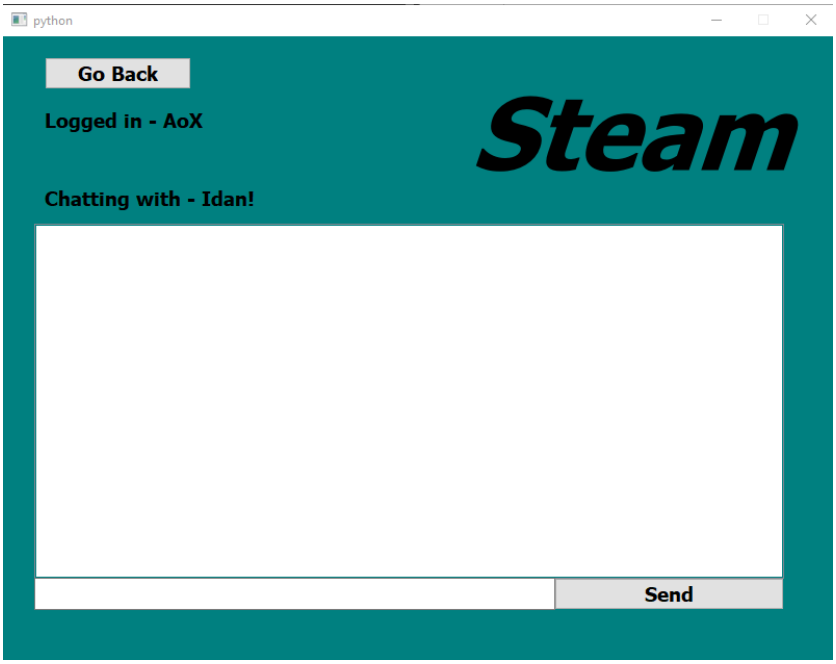
Chat – לוחצים על החבר הרצוי מרשימת החברים ולוחצים על הכפתור כדי לעבור למסך צ'אט איתו.

Play Together – בוחרים משחק בבעלותך וחבר מרשימת החברים ולוחצים על הכפתור כדי להזמין אותו למשחק.

Logout – מתנתק מהמשתמש וחוזר למסך התחברות.

הודעות: משחק זה לא בבעלות החבר שהזמנת, משחק לא מותקן, משחק כבר מותקן, כינוי לא קיים.

מסך צ'אט



מסך זה הוא צ'אט עם חבר. כדי לשלוח הודעה מקלידים אותה בתיבת טקסט למטה ולוחצים על כפתור Send. כדי לחזור למסך הראשי לוחצים על כפתור Go Back.

הודעות: חבר לא מחובר.

מדריך למפתח

קבצי השרת

Server.py – קובץ ה-main של השרת. מקבל ארגומנט אחד שהוא IP. מקבל לקוחות ויוצר thread לכל אחד מהם. מדפיס את ההודעות הנשלחות והמתקבלות, וגם את רשימת הלקוחות המחוברים בעת התנתקות\התחברות לקוח.

משתנים חשובים:

users – מילון של כל משתמש שמתחבר לפי מפתח שהוא הכינוי ומידע שהוא ה-socket. (גלובלי)

lock – מנעול כדי למנוע התנגשויות ב-threads במשאב המשותף users. (גלובלי)

threads – רשימה של כל ה-threads בשרת.

MG – משתנה מסוג Manager (מחלקה), כל thread יוצר אחד.

פונקציות חשובות:

handle_client – פעולת ה-thread כדי לדבר עם הלקוח.

update_users – מעדכנת את המשתנה הגלובלי users כשלקוח חדש מתחבר.

user_logout - מעדכנת את המשתנה הגלובלי users כשלקוח מתנתק.

Manager.py – מחלקת Manager אחראית על ההודעות המתקבלות והנשלחות.

משתנים חשובים:

self.db – משתנה מסוג DatabaseORM (מחלקה), באמצעותו מנהלים את ה-Database.

self.Enc - משתנה מסוג Encryptor (מחלקה), באמצעותו מצפינים\מפענחים צופן RSA.

self.user – משתנה מסוג user (מחלקה), שומר את הפרטים של הלקוח המחובר.

self.sock – ה-socket של הלקוח.

self.address – (ip,port) של הלקוח.

פונקציות חשובות:

handle_message – מקבלת הודעה מהלקוח מטפלת בבקשה שלו ומחזירה הודעה לשליחה.

Database.py – בקובץ זה מחלקה לכל טבלת SQL בצד השרת כפי שפירטתי בבסיס הנתונים. בנוסף לכך, המחלקה DatabaseORM אחראית על ניהול ה-Database (הוצאה והכנסה של מידע).

קבצי הלקוח

Client.py – קובץ ה-main של הלקוח. מקבל ארגומנט אחד שהוא IP. יוצר socket ומתחבר לשרת. יוצר שני threads, אחד שאחראי על הגרפיקה והקלט של הלקוח (מחלקת Screen) ואחד שמקבל הודעות בצורה אסינכורנית.

משתנים חשובים:

input – משתנה שתוכנו הוא קלט מהלקוח. (גלובלי)

login – משתנה בוליאני שמראה האם המשתמש מחובר. (גלובלי)

chat – משתנה מסוג Chat (מחלקה) אחראי על אחזקת המידע שנשלח מהשרת (כמו רשימת חברים) ואחראי על ההודעות המתקבלות והנשלחות.

פונקציות חשובות:

async_recv – לולאה שמקבלת הודעות מהשרת. אחראי על קבלת הודעות אסינכרונית של הלקוח.

wait_for_input – לולאה שמחכה לקלט מהלקוח.

user_input – פעולה שמשנה את המשתנה הגלובלי input. המחלקה Screen משתמשת בפעולה זו כדי להעביר את הקלט ל-main.

Chat.py – מחלקת Chat אחראי על אחזקת המידע שנשלח מהשרת ועל ההודעות המתקבלות והנשלחות.

משתנים חשובים:

self.friend_list – שומר את רשימת החברים הנשלחת מהשרת.

self.game_list – שומר את רשימת המשחקים הנשלחת מהשרת.

self.games_owned – שומר את רשימת המשחקים בבעלות המשתמש הנשלחת מהשרת.

self.friend_request – שומר את רשימת בקשות החברות הנשלחת מהשרת.

self.P2P – רשימה של משתנים נחוצים לתקשורת P2P: החבר, טוקן, כתובת, המשחק.

self.processes – רשימת המשחקים שרצים כרגע אצל המשתמש.

פונקציות חשובים:

encrypt_password – מקבל את ההודעה לשליחה של המשתמש ומצפין את הסיסמא. (רק בהודעות התחברות והרשמה)

handle_message – מקבל הודעה מהשרת ומטפל במידע.

start_game – מקבל משתנה בוליאני האם ליצור שרת או לקוח. יודע מה המשחק לפי המשתנה self.P2P.

Screen.py – מחלקת Screen אחראית על ה-GUI והקלט של הלקוח. לכל מסך בלקוח יש מחלקה משלו (למשל Login זה מחלקה), ובמחלקה לכל כפתור מחלקים פונקציה שמעבירה קלט ל-main. בנוסף, יש מחלקה Poper שאחראית על סיגנלים כשמתקבל מידע מהשרת כדי שהמסך יתעדכן.

<u>משתנים חשובים:</u>

chats – מילון שמחזיק את הצ'אטים של המשתמש עם החברים שלו לפי מפתח שם החבר וערך שהוא רשימה של הודעות. (גלובלי)

input_function – משתנה שהוא בעצם פונקציה ב-main שהיא user_input ככה שיהיה אפשר להעביר את קלט המשתמש מה-thread של Screen ל-main. (גלובלי)

Popup – משתנה שמשנים אותו דרך ה-main כשרוצים להעלות popup למשתמש (כשיש error למשל). (גלובלי)

info - משתנה שמשנים אותו דרך ה-main כשמתקבל מידע על המשתמש מהשרת (לאחר התחברות או הרשמה). (גלובלי)

got_msg - משתנה שמשנים אותו דרך ה-main כשמתקבלת הודעה ממשתמש אחר. (גלובלי)

<u>פונקציות חשובות:</u>

update_chat – מעדכן את הצ'אט כשמתקבלת הודעה מחבר. מקבל כלום.

load_user – מעדכן את המידע של המשתמש לאחר הרשמה\התחברות. מקבל כלום.

show_popup – מקפיץ popup למשתמש. מקבל מידע לרשום.

<u>קבצים בשרת ובלקוח:</u>

RSA.py – מחלקת Encryptor אחראית על פענוח והצפנה של הודעות בצופן RSA. הלקוח מקבל את המפתח הציבורי מהשרת בעת ההתחברות (accept) כדי שהוא יוכל לשלוח את הסיסמא מוצפנת.

<u>משתנים חשובים:</u>

self.PKey – המפתח הציבורי.

self.SKey – המפתח הסודי.

<u>פונקציות חשובות:</u>

rsa_encrypt – מקבל הודעה ומחזיר את ההודעה מוצפנת.

rsa_decrypt – מקבל הודעה מוצפנת, מפענח ומצפין אותה.

Protocol.py – בקובץ זה שתי פעולות, אחת לשליחת מידע ואחת לקבלת מידע לפי הפרוטוקול שבתחילת כל הודעה האורך שלה.

<u>פונקציות</u>

recv_by_size – מקבל סוקט ומחזיר הודעה שהתקבלה אחרי הבדיקה שהגיע בשלמותה.

send_with_size – מקבל סוקט והודעה לשליחה ושולח את ההודעה עם האורך שלה בהתחלה.

רפלקציה

העבודה על הפרויקט הייתה קשה אבל מהנה באיזשהו מקום, כנראה סיפוק עצמי. הספקתי לסיים את מה שרציתי, בסך הכל אני מאוד מרוצה מהפרויקט שיצא לי. אני למדתי לתכנן ולהכין פרויקט בסדר גודל אחר ממה שעשיתי עד עכשיו. למדתי גם לדבג בצורה יעילה יותר. מה שהיה קשה לי בפרויקט היה בעיקר התכנון שלו, תכננתי לעשות דברים שמעולם לא התנסיתי בהם בעבר. וכתוצאה מכך היה לי קשה לחשוב איך הפרויקט יהיה בנוי. למשל מעולם לא יצרתי GUI וגם לא הצפנת RSA בשרת לקוח.
המסקנה שלי מהפרויקט היא שאני יכול לעבוד על פרויקט גדול אם יש לי זמן. אני חושב שחילקתי את הזמן בצורה יעילה מאוד, ולכן סיימתי את הפרויקט ללא הרבה לחץ.

אם הייתי מתחיל היום כנראה שהייתי מתכנן טוב יותר את הפרויקט. לחקור לעומק בעיקר באילו packages אני משתמש ואיך הם עובדים כדי שאני אדע למה הם מסוגלים. למשל בתכנון רציתי להשתמש ב-GUI של kivy אבל כשהתחלתי את הפרויקט וחקרתי לעומק הבנתי שזה מאוד לא נוח (לי), לכן עברתי ל-PyQt5. אם הייתי חוקר מוקדם יותר זה כנראה היה חוסך לי כמה פעמים שהתחלתי מחדש.

אני חושב שהעבודה שלי הייתה יעילה מאוד, אבל עדיין היו כמה מקומות לשיפור. למרות שעמדתי בזמנים והתמדתי להגיש כל חלק בזמן. עדיין, יכלתי להתקדם אף יותר. בעיקר העדפתי שיהיה לי יותר זמן פנוי.

בסופו של דבר, אני מרגיש שהפרויקט שיפר אותו בתכנות, בתכנון, ואת הידע שלי בתחום בכללי. אני יודע לעבוד על פרויקט גדול יחסית, ויש לי עוד המון מקומות לשיפור. כלומר, יש לי פוטנציאל לעבוד על פרויקט גדול אף יותר בעתיד, ולשם אני מכוון.

ביבליוגרפיה

:RSA

סרטון המסביר את ההצפנה לעומק - https://www.youtube.com/watch?v=wXB-V_Keiu8&t

ערך בוקיפידיה - https://he.wikipedia.org/wiki/RSA