**המחלקה להנדסת תוכנה - קורס מחשוב ענן**

תרגיל בית 3-חלק ב יבש – עבודה בקבוצות וביחידים

**מועד הגשה 25/6/23 , 23:55**

מגישים:

**בר שטיינר 318401957**

**עומרי שלו 315838540**

**רן פולק 318265808**

**עמרי גאוי 313394850**

**עומר סומרשטיין 316439876**

**נועם אופיר 318295888**

***חלק ראשון: עבודה קבוצתית***

*טסטים באיטרציה ראשונה:*

משימת בדיקת פונקציונליות כוללת: פיתוח מקרי בדיקה המכסים מקרי קצה כדי לאמת את הפונקציונליות. לזהות ולתעד את כל הבעיות שהתגלו במהלך הבדיקה.

|  |  |  |  |
| --- | --- | --- | --- |
| Executor | Task | Requirement | Iteration |
|  |  |  | Iteration 1 |
| Backend developr - Omri G | Preparing the database | The application should support a database |  |
| Frontend developer - Omri S | Start the application | The application should contain all the required features |  |
| UI developer - Bar | User interface design | The user interface should be intuitive and user-friendly |  |
| QA - Noam | Functional testing | The application should handle invalid inputs and provide appropriate error messages |  |
| Product manager - Ran | Address feedback and make improvements | The application should be suitable for diverse users |  |

**טסטים באיטרציה שניה:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Requirement** | **Executor** |
| **Task 1** | **Preparing the login database** | backend developer |
|  | Create a login table in the FireBase | backend developer |
| **Task 2** | **Preparing the database to save users session information** | backend developer |
|  | Create a users table in the FireBase | backend developer |
| **Task 3** | **Add leaderboard table** | **Frontend Developer Backend developer  Product Manager  UI  QA** |
|  | Top 3 users will be in different colors | Frontend developer Backend developerUI |
|  | Acceptance Test - The table shows the top10 users where the top 3 are shown with different colors. | QA Product Manager |
| **Task 4** | **Add new screen of the user’s board during the game** | **Scrum Manager Frontend developer Backend developer Product Manager UI** |
|  | Create new button for opening the board | Frontend developer  Scrum Master |
|  | Show the user’s board | Frontend developer  Backend developer UI |
|  | Acceptance Test - check if the user’s ships are the same as in the user bord | QA Product Manager |
| **Task 5** | **Add new Admin option in order to add new question** | **Frontend developer  Backend developer UI  QA** |
|  | Add the new button | UI Frontend |
|  | Send the new question to the DB | Frontend developer  Backend developer |
|  | Acceptance Test - The table shows all the questions from the database, and when the admin clicks on a specific question all the info of the question will be displayed. | QA |
| **Task 6** | **Admin can edit questions from database** | **UI  Backend developer Frontend developer scrum master product manager QA** |
|  | Add new button of edit questions | Frontend developer Backend developer product manager scrum master |
|  | Show the questions from the DB | Frontend developer Backend developer  UI |
|  | Let the user edit the questions | Backend developer  Frontend developer |
|  | Acceptance Test - Check if the information has been updated in the database accordingly to the input | QA |
| **Task 7** | **Add Login and Logout feature** | **UI Backend developer Frontend developer scrum master product manager QA** |
|  | Create login page | UI  Frontend developer  Product Manager  Scrum Master |
|  | Transfer data to and from the DB | Frontend developer Backend developer |
|  | Acceptance Test - Check if the information is existing in the DB | QA |
|  | Create Logout Button on the main screen | Frontend developer  UI Product Manager |
|  | Acceptance Test - Check if the information is existing in the DB | QA |
| **Task 8** | **Add registration option** | **Backend developer Frontend developer Product Manager QA** |
|  | Add register button | Frontend developer Product Manager |
|  | Save the user input into the database | Backend developer |
|  | Acceptance test - Check if the new user added to the database according to the input. | QA |
| **Task 9** | **Improve user interface from the first iteration** | **UI  Frontend developer scrum master Product Manager Backend developer** |
|  | Designing the Screens | UI  Frontend developer scrum master  Product Manager  Backend developer |
| **Task 10** | **Add rules page that describes how to play** | **UI Frontend developer Product Manager  scrum master  Backend developer** |
|  | Rewrite the rules of the game | Product Manager scrum master  Backend developer |
|  | Show the rules to the user | Frontend developer UI |
|  | Design the rules page | UI  Frontend developer |
| **Task 11** | **User can see his stats during the game by graph** | **UI  Backend developer  Product Manager  Frontend developer  scrum master  QA** |
|  | Create a “show me stats button” | Frontend developer |
|  | Take the stats from the DB | Backend developer |
|  | Show a graph by the stats | Frontend developer  Backend developer UI |
|  | Let the user close the stats and back to the game | Frontend developer scrum master |
|  | Acceptance test - Check the information in DB | QA |
| **Task 12** | **User can finish the game whenever he wants** | **Frontend developer Backend developer Product Manager scrum master** |
|  | Add exit button | Frontend developer |
|  | Show the finish game screen and update the DB | **Frontend developer** Backend developer **Product Manager scrum master** |

**בדיקה עבור קבוצת SNAKE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| איזו פונקציונליות הטסט בודק | איך הטסט בודק | ממצאים צפויים | ממצאים בפועל | תוצאות-עבר/לא עבר | במידה ולא עבר מה הייתה הבעיה |
| **הצגת טבלת משתתפים ופירוט ניקוד מתאים** | בדיקה שמספר השחקנים בטבלת המשתתפים תואם את מספר המשחקים (איטרציות שהתבצעו) | הטבלה מציגה את נתוני השחקנים והמשחקים באופן תקין | הטבלה מציגה את נתוני השחקנים והמשחקים באופן תקין | עבר |  |
|  | בדיקה שהניקוד של כל שחקן תואם את התוצאות של המשחקים ששיחק | הטבלה מציגה את ציון השחקנים בהתאם לציון שצברו במהלך המשחק | הטבלה מציגה את ציון השחקנים בהתאם לציון שצברו במהלך המשחק | עבר |  |
| **תצוגת עמוד מנהל כולל אפשרות עריכה והוספת שאלות** | התחברות לעמוד מנהל באמצעות סיסמא תקינה | לאחר הזנת סיסמא תקינה המשתמש יועבר לעמוד מנהל | לאחר הזנת סיסמא תקינה המשתמש עבר לעמוד המנהל | עבר |  |
|  | התחברות לעמוד מנהל באמצעות סיסמא לא תקינה | לאחר הזנת סיסמא שגויה המשתמש לא יועבר לעמוד מנהל | לאחר הזנת סיסמא שגויה המשתמש לא עבר לעמוד מנהל | עבר |  |
|  | בדיקה עבור הוספת שאלה ריקה | לא תתאפשר הוספת שאלה ריקה | לא התאפשרה הוספת שאלה ריקה | עבר |  |
|  | בדיקה עבור הוספת שאלה תקינה | השאלה תתווסף למסד הנתונים בהצלחה ותופיע ברשימת השאלות | השאלה התווספה למסד הנתונים בהצלחה והופיעה ברשימת השאלות | עבר |  |
|  | בדיקה עבור שאלה לא ריקה | הודעת שגיאה מתאימה למשתמש | הודעת שגיאה על שדה ריק | עבר |  |
|  | בדיקה עבור עריכת שאלה לשאלה תקינה | השאלה תתעדכן במסד הנתונים | השאלה התעדכנה בהתאם לעריכה שביצענו | עבר |  |
| **רישום למשחק** | בדיקה עבור רישום פרטים ריקים | הודעת שגיאה מתאימה תוצג למשתמש | הודעה שהשדות ריקים | עבר |  |
|  | בדיקה עבור רישום פרטים בצורה תקינה | המשתמש נוסף בהצלחה למסד נתונים והודעת הצלחה תוצג למשתמש | המשתמש נוסף בהצלחה למסד נתונים והודעת הצלחה מוצגת למשתמש | עבר |  |
|  | בדיקה עבור רישום פרטים בצורה שגויה | הודעת שגיאה מתאימה תוצג למשתמש | הודעה שהפרטים לא נכונים | עבר |  |
|  | בדיקה כי משתמש שנרשם אכן פרטיו מופיעים במסד הנתונים | המשתמש נוסף למסד הנתונים | המשתמש נוסף למסד הנתונים | עבר |  |
|  | בדיקה כי משתמש לא רשום פרטיו לא יופיעו במסד הנתונים | המשתמש לא נוסף למסד הנתונים | המשתמש  לא נוסף למסד הנתונים | עבר |  |
|  | בדיקה עבור משתמש אשר מנסה להירשם עם פרטים של משתמש קיים | לאחר הזנת הפרטים הדומים המערכת תציג הודעת שגיאה מתאימה | הופיעה הודעת שגיאה מתאימה | עבר |  |
| **התחברות למשחק** | בדיקה עבור התחברות עם פרטים ריקים | לאחר הזנת פרטים ריקים תופיע הודעת שגיאה | הופיעה הודעת שגיאה | עבר |  |
|  | בדיקה עבור התחברות עם פרטים תקינים | לאחר הזנת פרטים תקינים המשתמש יועבר לעמוד הבא | המשתמש הזין פרטים תקינים ועבר לעמוד הבא-עמוד ראשי | עבר |  |
|  | בדיקה עבור התחברות עם פרטים שגויים | לאחר הזנת פרטים שגויים המשתמש לא יועבר לעמוד הבא ויקבל התראה שהפרטים שהזין שגויים | המשתמש הזין פרטים שגויים והופיעה לו הודעת שגיאה | עבר |  |
|  | בדיקה עבור התחברות למשחק של משתמש לא רשום | לאחר ניסיון התחברות תופיע הודעת שגיאה | לאחר ניסיון התחברות הופיעה הודעת שגיאה | עבר |  |
| **בדיקת הודעות המערכת** | בדיקה עבור הודעה אשר מופיעה בעת הכנסת פרטים שגויים | הודעות שגיאה יופיעו בהתאם לשגיאות שהתרחשו | הודעות השגיאה לא מוחלפות אלא מתווספות אחת לשנייה בעמוד. | לא עבר | הודעות השגיאה לא מוחלפות אלא מתווספות אחת לשנייה בעמוד. |
|  |  |  |  |  |  |

**חלק שני: עבודה אישית**

**שם הסטודנט: עמרי גאוי 313394850**

|  |  |
| --- | --- |
| **קריטריון** | **הסבר** |
| מימוש | הקוד ברובו מבצע את הנדרש, אך עם זאת ישנו כפתור שמבצע משהו שלא נכון. כלומר כפתור BACK מבצע LOGOUT. |
| יעילות | תמיד ניתן לשפר את יעילות הקוד על ידי שימוש בספריות חיצוניות, תכנון נכון של פונקציות.  המשחק דיי איטי מבחינת כפתורים וכו ונדרש שיפור בקטע הזה. |
| פשטות | ניתן לשפר את פשטות הקוד על ידי שימוש בOOP |
| מודולריות | הקוד מודולרי, ניתן לראות שהקוד מחולק לClasses, פעולות שרק אדמין יכול לבצע, טבלת TOP 10 וכו. כלומר ניתן בכל רגע נתון לדעת איפה כל דבר נמצא בקוד |
| תבניות עיצוב | יש שימוש בSingleton להתחברות לדטאבייס. |
| OOP | אין שימוש כלל בOOP, הקוד בנוי בצורה של סקריפטים שכל פונקציה מבצעת חלק פונקציונלי אחר במשחק. |
| באגים וטעויות | * יש באג כאשר מתחברים למשתמש ואז נכנסים לכפתור HELP. כאשר לוחצים על Back, מבוצע LOGOUT. * כלוחצים LOGIN בפעםם הראשונה עם פרטים נכונים זה לא מתחבר. רק בפעם השנייה. |
| טיפול בשגיאות | * יש התראות על שגיאות משתמש במסך LOGIN/REGISTER. אם היוזר מכניס פרטים לא נכונים אכן מוצגת הודעה מתאימה. עם זאת כאשר מוצגת הודעת שגיאה פעמיים, היא מוצגת מתחת להודעה הראשונה ולא דורסת את הקודמת מה שיוצר עומס לעיניים. * לגבי Exceptions, הם אכן השתמשו בtry catch, מה שמאוד עוזר למנוע טעויות בזמן ריצה. דיבוג הקוד לא כזה קל שכן הקוד יחסית עמוס ודורש מעבר על המון פונקציונליות. * הודעות המשתמש ברורת, אך הייתי רוצה לראות אותם בצורה יותר מודגש ועם צבעים |
| בדיקות | ניתן להוסיף עוד try catch בכל מני קטעי קוד מסויימים כדי למנוע טעויות בזמן ריצה. |
| שימושיות | מאוד, הממשק מאוד נוח ונעים לעין. אין עומס על המשתמש. בנוסף שיחקתי במשך כרבע שעה ולא הרגשתי שיעמום. |
| תיעוד | רוב הפונקציות לא מתועדות, אך התוכן שלהן כן. בנוסף מעל כל חלק קוד משמעותי יש תיעוד שמתאר את החלק הזה, מה שעוזר להבנה כללית של הקוד. |
| אתיקה | * אין נתוני משתמש אישיים בכל המשחק. * מבחינת ניצול יש, ניתן להירשם עם שם משתמש בשם של מישהו וכו ואז להגיע לטבלה בTOP10 ואז כולם יראו תשם הספציפי. |
| אבטחה | אין בעיות אבטחה ידועות. |
| ביצועים | שינוי עתידי יכול לגרום לכך שנהיה צריכים לשנות המון פונקציות אחרות. שינוי בביצועים גם יכול להיות שכן הוספה של פונקציות אחרות יכולות להאט תקוד. |
| ביצועים | ניתן לשפר את הביצועים על ידי צמצום פונקציות, שימוש במבני נתונים, ספריות חיצוניות. |
| readability | הקוד לא מובן בקלות אך ניתן להתמצא בו כי הוא מחולק למקטעים. חלק שלא היה לי מובן כלכך זה התצוגה של הגרף. לאחר קריאה מעמיקה הצלחתי להבין בסופו של דבר. |
| Extensibility | הקוד לא Extensibility בשום צורה, שכן הוספה של כל דבר חדש יכולה לגרום להמון שינויים בכל מני פונקציות אחרות. |
| Portability | המשחק מותאם לריצה רק בgoogle colab. לפלטפורמות אחרות כנראה שיצטרכו לבצע שינוי גדול. |

***1.* כיצד הרגשת במהלך החוויה של מעבר על הקוד?**

המעבר לא היה כזה קל שכן הקוד מאוד עמוס. עם זאת היה לי כיף לראות את הדרך שבה הם מימשו את הדברים, החלוקה למקטעים ושימוש בDesign pattern של Singelton.  
בנוסף ממעבר על הקוד למדתי על ספריות חדשות שלא ידעתי שאפשר להשתמש בהן בCOLAB.  
בנוסף ראיתי דרך של שימוש בJavascript בשילוב PYTHON.

***2.*   האם לדעתך ניתן לשפר תרגיל זה?**ניתן להוסיף בדיקות Unit Tests ובדיקות עומסים דרך הFirebase.

***3.*      הערות נוספות**  
  
היה ניתן לפרסם רשימה של הקבוצות עצמם, היה קשה להשיג את חברה הקבוצה שאותה אנחנו צריכים לבדוק.

**שם הסטודנט: עומרי שלו 315838540**

|  |  |
| --- | --- |
| **קריטריון** | **הסבר** |
| מימוש | המימוש תקין מבחינה פונקציונלית, המשחק עובד באופן מלא. נכנסתי דרך משתמש האדמין וראיתי שכל המסכים אכן עובדים. |
| יעילות | המשחק עובד די לאט, אולי היה אפשר לייעל את המימוש על מנת לגרום למשחק לרוץ מהר יותר. |
| פשטות | ניתן לפשט את הקוד ע"י שימוש ב OOP. |
| מודולריות | כן, הקוד מחולק לפי מחלקות מסודר. ניתן לעבור בין המחלקות ולדעת על מה מדובר בכל רגע נתון. |
| תבניות עיצוב | יש שימוש ב- Singleton עבור התחברות ל-DB. |
| OOP | אין שימוש ב- OOP |
| באגים וטעויות | * כאשר מתחברים למשתמש, נכנסים ל- HELP ואז חוזרים לדף הבית (BACK), מבוצע LOGOUT. |
| טיפול בשגיאות | * יש התראות על שגיאות משתמש במסך LOGIN/REGISTER כאשר היוזר מכניס פרטים לא נכונים.  מוצגת הודעה מתאימה אבל היא מוצגת בכפילות מתחת להודעה הראשונה ולא מחליפה את ההודעה הקודמת מה שמגדיל את המסך. * יש שימוש ב-try and catch, מה שמאוד עוזר למנוע טעויות בזמן ריצה. |
| בדיקות | לא ראיתי עוד בדיקות שחסרות |
| שימושיות | הממשק טיפה כבד בצפייה - מסך שחור קצת מקשה להיות מול זה לזמן רב. חוץ מזה המשחק עובד סבבה ולא מסובכת. |
| תיעוד | יש מעט מאוד תיעוד בקוד, אך ניתן להבין מה מתרחש במחלקה לפי תיעוד הכותרת |
| אתיקה | * אין פרטים אישיים של המשתמשים. * כל משתמש שיגיע לטבלת ה-leaderboard יופיע לשאר המשתמשים. |
| אבטחה | לא ראיתי דרך לפריצת אבטחה כלשהי |
| ביצועים | שינוי עתידי יכול לגרום לשינוי פונקציונליות אחרות והעמסה נוספת על הקוד תאט אותו יותר ממה שכבר כי הוא עמוס מאוד. |
| ביצועים | מכיוון שראיתי שהיה שימוש בספריות רבות אפשר לייעל את הביצועים אולי ע"י כתיבת פו' גנריות לחסכון קוד. כך העומס יקטן. |
| readability | רוב הקוד היה לי מובן, בעיקר מכיוון שמחולק למחלקות. אפשר היה להוסיף קצת יותר רווחים בין קטעי קוד ולהוסיף תיעודים על מנת להקל ההבנה של כל קטע. |
| האם שמות הפונקציות מתארות את פעולתן | כן, נראה ששמות הפו' אכן מסבירות את מה שמבצעות בפועל |
| האם יש כפילויות קוד? | לא זיהיתי כפילויות זהות לגמרי אבל יש קטעי קוד דומים שאפשר היה לחסוך ע"י פו' גנריות. |

**שאלות פתוחות:**

**1.  כיצד הרגשת במהלך החוויה של מעבר על הקוד?**

הקוד מאוד ארוך לכן היה קשה, עקב חוסר בתיעוד היה טיפה ארוך יותר מהמצופה.

**2.      האם לדעתך ניתן לשפר תרגיל זה?**

לדעתי לעבור על קוד של קבוצה אחרת לא הועיל לי בשום דבר, לא למדתי משהו חדש והרגשתי כמו תוכי.

**3.      הערות נוספות**

הייתי מוותר על החלק הזה במטלה, לא הרגשתי למידה כלשהי אלא רק לחץ בזמנים כי לקח הרבה זמן לעבור על הקוד ובסוף הסמסטר שהכל צפוף הרגשתי המון לחץ לסיים את המשימה בזמן.

**שם הסטודנט: בר שטיינר  318401957**

|  |  |
| --- | --- |
| **קריטריון** | **הסבר** |
| מימוש | מימוש המשחק תקין מבחינת פונקציונליות. כל הפונקציות מבצעות את הנדרש, הכפתורים מאפשרים למשתמש לבצע את המצופה, להתקדם בין העמודים השונים ולבצע את מטרת המשחק כראוי. |
| יעילות | המשחק עובד בצורה יחסית איטית |
| פשטות | הממשק פשוט גם מבחינת הפונקציות בקוד.וגם מבחינת הממשק למשתמש אשר הוא נוח ובעל מעט כפתורים יעילים להתקדמות במשחק. לצורך פשטות הקוד ניתן להוסיף ספריות. |
| מודולריות | המשחק בנוי בצורה מודולרית, ישנה חלוקה ברורה למחלקות כך שאפשר לזהות ולהתאים כל חלק בקוד לביצוע שלו במשחק |
| תבניות עיצוב | תבנית עיצוב סינגלטון לצורך התחברות למסד נתונים |
| OOP | אין שימוש במחלקות יורשות ופונקציות הורשה |
| באגים וטעויות | \*התחברות ראשונית למשחק עם פרטי משתמש תקינים לא מצליחה רק לאחר לחיצה שניה על כפתור ההתחברות מצליחים להתחבר כראוי |
| טיפול בשגיאות | \*בניסיון התחברות למשחק עם פרטי משתמש שגויים מופיעה הודעת שגיאה מתאימה  \*בעת רישום למשחק עם סיסמא ואימות סיסמא שונה מופיעה הודעת שגיאה מתאימה |
| בדיקות | \*ניתן להוסיף בדיקה אם המשתמש כבר עשה שגיאה אז לא להוסיף את הודעת השגיאה אלא להציג את ההודעה הרלוונטית בלבד.  \*ניתן להוסיף בדיקה לגבי קלטים ספציפיים בהכנסת הנתונים למשל 7 ספרות לפחות וכו |
| שימושיות | מבחינת השימושיות אני חושבת שהייתי שמחה להשתמש במערכת זאת על מנת ללמוד בצורה יעילה יותר.המערכת אינה מסובכת לשימוש והיא נוחה לשימוש. ישנה עקביות במערכת מבחינת חזרתיות של כפתורים והמערכת נלמדת בקלות ונותנת תחושת ביטחון בעת שימוש בה. |
| תיעוד | המחלקות אינן מתועדות אך חלק מהפונקציות הפנימיות מתועדות בצורה חלקית |
| אתיקה | ישנה שמירה על כללי האתיקה המקובלים:  מבחינת פרטיות ואבטחת מידע- הפרטים האישיים של המשתמשים נשמרים בצורה מאובטחת במסד הנתונים  מבחינת שימור קניין רוחני- המשחק מקורי ולא כולל בתוכו קוד של צד שלישי  מבחינת יחס למשתמש- המערכת שומרת על יחס שווה למשתמשים ללא הבדל או העדפה |
| אבטחה | הפרטים האישיים של המשתמשים נשארים מאובטחים |
| ביצועים | שינוי עתידי בקוד עלול להביא בפגיעה בביצועים אם לא יותאם לעבודה בגוגל COLB |
| ביצועים | ניתן לשפר את הביצועים באמצעות הוספת סיפריות רלוונטיות ובאמצעות שימוש בפונקציות יעילות יותר או שרת נתונים חזק יותר |
| readability | הקוד מובן בקלות באמצעות החלוקה הנכונה למחלקות והגדרה ברורה של הפונקציות לא כל כך הבנתי את החלק של הופעת משתמש אורח בלוח 10 המנצחים במחלקת Leaderboard |
| Maintainability | הקוד יחסית נקי וניתן לתחזוקה בגלל החלוקה הנכונה לפונקציות והתיעוד. רוב הקוד כתוב בצורה מפושטת ויעילה כך שניתן לבצע עליו תחזוקה ובנוסף ישנו שימוש נכון במסד נתונים |
| Correctness | הקוד מבצע את המוטל עליו מבחינה פונקציונלית וישנן בדיקות מתאימות למקרי קצה למשל בדיקה עבור הכנסת קלט שגוי בהתחברות למשתמש |

**שאלות פתוחות:**

**1.  כיצד הרגשת במהלך החוויה של מעבר על הקוד?**

אני חושבת שהמעבר על קוד של קבוצה אחרת לא תרם לי משום שהמעבר לא מספיק מעמיק מול קוד שאנחנו לא מכירים כלל. הייתי מעדיפה שרוב העבודה תהיה על הקוד שלנו כך שתהיה משמעות גדולה יותר עבורנו וגם כך נוכל ללמוד טוב יותר מהטעויות שלנו.

היה מעט קשה להצליח לסקור קוד שלם של קבוצה אחרת ולחשוב על הבדיקות הרלוונטיות עבורם שלא תמיד תאמו את הבדיקות שתכננו עבור הקוד שלנו אבל עם זאת למדתי מהקוד על ספריות שלא הכרתי ומימושים יעילים למספר פונקציות ושילוב של תבנית עיצוב עם שפת פייתון.

**2.      האם לדעתך ניתן לשפר תרגיל זה?**

ניתן לשפר תרגיל זה באמצעות כך שכל קבוצה תעשה את הערכה עבור המטלה שלה ותכין את טבלת הבדיקות עבור הקבוצה הבודקת משום שהקבוצה מכירה הכי טוב את הקוד שלה וכיצד ניתן לבדוק אותו ומקסימום לתת לקבוצה הבודקת להוסיף מספר בדיקות כפי שהיא חושבת

**3.      הערות נוספות**

על מנת לייעל את התקשורת בין הקבוצות ניתן היה לייצר קובץ משותף לסטודנטים כך שכל הקבוצות יוכלו לגשת למחברות של הקבוצות שאותם הם בודקים בצורה קלה ויעילה. היה קושי בהשגת הקבוצות שמולן אנחנו עובדים וזה עיכב את הקבוצה מהתקדמות במטלה.

**שם הסטודנט: רן פולק**

|  |  |
| --- | --- |
| **קריטריון** | **הסבר** |
| מימוש | המשחק מבצע את מה שנדרש ממנו, המשחק עוד באופן מלא , כל כפתור עושה את המצופה ממנו וסה"כ הקוד מבצע ממנו את הנדרש |
| יעילות | המשחק עבד קצת לאט , אבל זה מובן בשל השילוב של כמה שפות תכנות עם סביבת עבודה שלא נחשבת למהירה במיוחד. |
| פשטות | ניתן לפשט את הקוד על ידי שימוש בOOP |
| מודולריות | הקוד אכן עובד בצורה מודולורית , יש חלוקה למחלקות ותאים שונים , כל תא מכיל את קוד שמתאים למצב אחר במשחק. ניתן להתמצא יחסית בקלות ולראות איפה נמצא כל דבר. |
| תבניות עיצוב | היה שימוש בdesign pattern Singelton להתחברות לDB . |
| OOP | אין שימוש |
| באגים וטעויות | כאשר מתחברים למשתמש , נכנסים לעזרה למשתמש ואז חוזרים אחורה למסך הבית, מתבצע התנתקות אוטומטית מהמשתמש. |
| טיפול בשגיאות | יש הודעות מתאימות לשגיאות שעולות, אם הכנסנו פרטי משתמש לא נכונים בהתחברות . אם יש ניסיון להוסיף שאלה בצורה לא תקנית גם כן מוצגת הודעה בהתאם. |
| בדיקות | תמיד ניתן להוסיף עוד בדיקות כדי שהקוד באמת יעבור על כל מקרי הקצה שקיימים, אך סה"כ מרבית השגיאות טופלו בהצלחה טובה |
| שימושיות | המשחק מאוד נעים ונוח למשחק , בהחלט אפשר להעביר בו את הזמן מבלי להשתעממם מהר מדי . |
| תיעוד | אין תיעוד מלא של הקוד |
| אתיקה | יש שמירה על האתיקה במשחק , נתונים אישיים לא מופיעים בשום מקום , המערכת שומרת על מבנה תקין . |
| אבטחה | לא ראיתי בעיה עם האבטחה, הנתונים נשמרו כמו שצריך |
| ביצועים | ניתן לשפר את הביצועים עם שימוש יעיל יותר בפונקציות ושימוש בספריות חיצוניות שיכולות לחסוך זמן |
|  |  |
| readability | ניתן להתמצא בקוד די בקלות מכיוון שהוא כן היה מחולק טוב מבחינת ההפרדה, היה אפשר לשפר זאת עם תיעוד יותר מלא . |
| Portability | אם נרצה להריץ את המשחק הזה על פלטפורמות אחרות כנראה ולא נצליח. המשחק מותאם במיוחד לGoogle Colab . |
| Scalability | בדומה למה שכתבתי מעל , במידה ונרצה להרחיב את המערכת זה יהיה קשה מאוד וכל שינוי כזה ידרוש עבודה כמעט מאפס . |

**שאלות פתוחות:**

1. כיצד הרגשת במהלך החוויה של מעבר על הקוד?

תשובה: נהניתי לראות צורה נוספת של חשיבה , כיצד הם התמודדו עם בעיות שגם עלו אצלנו במהלך כתיבת הקוד , ואיך כל קבוצה לקחה לכיוון שלה את המשחקיות .  
למדתי ספריות ואפשרויות חדשות שלא הכרתי לפני זה עם שימוש בGoogle Colab , התיעוד המלא היה גורם למעבר על הקוד להיות קצת יותר חלק אבל סה"כ נהניתי מאוד .

1. האם לדעתך ניתן לשפר תרגיל זה?

תשובה: אולי שיהיה שיתוף פעולה בלייב בין הקבוצה הבודקת והנבדקת , ככה גם נוכל להבין יותר אם יש לנו שאלות

1. הערות נוספות:

תשובה: אין, תודה רבה

**שם הסטודנט: עומר זומרשטיין 316439876**

|  |  |
| --- | --- |
| **קריטריון** | **הסבר** |
| מימוש | המשחק הוא בעל פונקציונאליות תקינה, הכפתורים עובדים כנדרש ומעבירים למסך חדש שמצופה לעבור אליהם בעת לחיצה עליהם, כלומר הקוד מבצע את הנדרש |
| יעילות | המשחק עובד מעט לאט, אך דבר זה יכול להיווצר בשל העובדה שמשתמשים בפייתון שידוע כשפה איטית ומסורבלת יותר. בנוסף השימוש בגוגל Colab יכול לגרום לקוד להיות איטי יותר בגלל תלות ברוחב פס, אינטרנט, וכמובן העובדה שהקולאב לא באמת עובד כמו שצריך עם HTML. יכול להיות שהיה ניתן לייעל את הקוד, תמיד אפשר אך יש הגבלות מסביב. |
| פשטות | תמיד ניתן לפשט את הקוד יותר אך לא תמיד זה הכרחי, לדעתי הם ביצעו עבודה טובה במימוש הקוד |
| מודולריות | הקוד מחולק כמו שצריך למחלקות ולקטעי קוד שונים, ישנה חלוקה ל- import-ים, פונקציות נפרדות. |
| תבניות עיצוב | ראינו שימוש בתבנית מסוג Singleton על מנת להתחבר למסד נתונים, זאת כדי ליצור מופע אחד ויחיד של המחלקה ולא ליצור כמה מופעים |
| OOP | לא שמתי לב לשימוש בתכנות מונחה עצמים |
| באגים וטעויות | * אני אישית חוויתי באג בהתחברות, יכלתי לרשום שם משתמש וסיסמא תקינים, להתנתק מהמשתמש, ולהיכנס שוב עם שם משתמש תקין וסיסמא לא תקינה וזה היה מתחבר. למעט זה לא חוויית עוד באגים ולא נצפו טעויות.      * בנוסף, לחיצת התחברות ראשונה ב- login לא עובדת ובלחיצה שניה כן עובד למרות שבשתי המקרים הפרטים היו נכונים. |
| טיפול בשגיאות | ישנן התראות על שגיאות משתמש בחלקים שבהם נדרש פעולת משתמש, כמו לדוגמה במסך התחברות - צריך להכניס שם משתמש וסיסמא, אם לא הכנסת יש פלט שנדרש להכניס, אם לא נכונים אז יש פלט כזה. או אם רוצים להוסיף שאלות ב- Admin אז בכל שדה ריק - יופיע x ליד באדום שדואג להדגיש שנדרש למלא את השדות האלו |
| בדיקות | בשל הבאג שתיארתי לעיל כנראה שנדרש לבצע בדיקה אחרת לגבי ההתחברות או התנתקות, כתלות באיפה הבאג נוצר. |
| שימושיות | המשחק בוצע באופן פשוט וקל שמתאים לכלל המשתמשים שיודעים אנגלית. נוח לשימוש, נעים לעין, הכל ברור, עיצוב אחיד. |
| תיעוד | קיים תיעוד במקומות שונים אך לא בהכרח תיעוד מלא, כלומר היה ניתן להוסיף תיעוד. |
| אתיקה | אתיקה תקינה וללא הפרת פרטיות או תיעדוף קבוצות מסויימות. |
| אבטחה | אבטחה תקינה, סיסמאות לא גלויות והן נסתרות, כמו גם המידע לגבי פרטים אישיים של משתמשים אחרים מוסתר. |
| ביצועים | שינוי עתידי בקוד יכול לפגום בבצועים אם פונקציות מסויימות ימחקו. |
| ביצועים | אני לא בטוח שבעבודה עם גוגל קולאב ניתן לשפר ביצועים כי יכול להיות שהקוד עבד לי לאט בגלל האינטרנט שלי ולא בגלל שהוא איטי. |
| readability | הקוד מחולק בצורה יפה לקטעי קוד שונים, כלומר גם אם הוא לא מובן בקריאה ראשונית ניתן עדיין להתמצא בו ולהבין מה קורה |
| מיחזור / כפילות | לא בוצע מיחזור של קוד כלומר הקוד נכתב פעם אחת ולא היה מועתק למקומות אחרים בקוד כמו שהוא. כן היה אפשר להשתמש אולי יותר בספריות מוכנות ולחסוך כתיבה. |
| פיתוח נוסף | פיתוח נוסף של הקוד למימדים יותר גדולים יכול להתרחש בעתיד תוך הוספת קטעי קוד ושיפור פונקציות קיימות לטובת הקמת משחק ברמה יותר גבוהה שיהיה ניתן גם לרכוש עבור סכום מסויים |

**שאלות פתוחות:**

1. כיצד הרגשת במהלך החוויה של מעבר על הקוד?

תשובה:

המעבר על הקוד היה ארוך שכן פרויקט זה לא היה קצר ונדרש היה לכתוב בו לא מעט דברים, אך בגלל העובדה שהחלוקה בוצעה בצורה יפה ובגלל העובדה שהיה ניתן להבין מה פונקציה עושה לפי השמות המשמעותיים שלה, היה יותר קל לעבור על הקוד.

1. האם לדעתך ניתן לשפר תרגיל זה?

תשובה:

לא

1. הערות נוספות:

תשובה:

אין.

**שם הסטודנט: נועם אופיר 318295888**

|  |  |
| --- | --- |
| **קריטריון** | **הסבר** |
| מימוש | מימוש תקין,  הפונקציונליות הדרושה קיימת במשחק והכל עובד כמצופה (מבחינה כללית) |
| יעילות | ממשק המשחק מרגיש איטי,  יכול להיות שזה מצביע על חוסר יעילות באופן רינדור עמודי הממשק או בספריות הגרפיות ששומשו. |
| פשטות | הממשק יחסית פשוט, הפונקציונליות מוצגת בעיקר ככפתורים מסודרים בשורה המפנים לעמ' הרלוונטי, עיצוב אחיד ולא עמוס. |
| מודולריות | הקוד יחסית מודולרי,  ניתן לראות שקיימים קלאסים יחודיים לרוב המרכיבים במערכת המתארים את תהליך המשחק |
| תבניות עיצוב | תבנית עיצוב singleton מומשה בקלאס האחראי לחיבור למסד הנתונים. |
| OOP | לא קיים שימוש בOOP מעבר להגדרת מחלקות ופונקציות בצורה בסיסית |
| באגים וטעויות | הודעות המערכת מתווספות במקום להתחלף,  בעמוד הhelp כפתור ה back מעביר למסך ההתחברות ולא למסך בחירה הראשי,  בהתחברות לפעמים נתונים נכונים מחזירים שגיאה ועובדים רק בניסיון נוסף. |
| טיפול בשגיאות | שגיאות בקלט מייצרות התראות מתאימות במרבית המצבים שדורשים זאת,  לדוגמא במסכי ההתחברות והוספת שאלות. |
| בדיקות | מרבית הפונקציונליות נבדקה וטופלה בהתאם, ישנם כמה מקרי קצה שלא טופלו או נבדקו כמוזכר למעלה בשגיאות |
| שימושיות | אופן המשחק פשוט ושימושי, נגיש ומובן וממלא את מטרותיו.  סך הכל המשחק שימושי לפי דרישותיו |
| תיעוד | קיים תיעוד פנימי מינימלי במחלקות, פונקציות אך חסר תיעוד כללי המתאר את הפונקציה/מחלקה |
| אתיקה | נתונים אישיים לא מופיעים בממשק המשחק |
| אבטחה | אין בעיות אבטחה ברורות, לא קיים טיפול מיוחד בקוד כלפי וריפיקציית נתונים שמועלים לדטבייס (שאלות, הרשמה) כך שאולי ניתן להריץ סקריפטים/קוד זדוני ישירות בדטבייס (לא קיים כלי escaper) |
| ביצועים | שינויים עתידיים ודרישות חדשות עלולות להעמיס על הקוד הכבר עמוס ולהביא את המערכת למצב לא שמיש |
| ביצועים | יתכן שניתן לשפר את ביצועי הקוד ומהירות המערכת ע"י שימוש בספריות, כלים גרפיים רלוונטיים יותר לממשקים כאלה. |
| readability | הקוד יחסית ארוך, דבר המשפיע לרעה על הבנתו.  לעומת זאת, מחולק בצורה ראויה אז ניתן להתמצא. |
| scalability | אין שימוש בתבניות או מתודולוגיות שיתמכו בהרחבה פרקטית ומהירה של המערכת (במידה ורוצים פונקציונליות נוספת), כל דרישה חדשה תצטרך למומש כמעט מאפס |
| **Interoperability** | לא קיימים כלים/חלקים מיוחדים בקוד עבור אינטגרציה עתידית של המערכת עם פלטפורמות/מערכות אחרות |

***1.* כיצד הרגשת במהלך החוויה של מעבר על הקוד?**

המעבר על הקוד היה טיפה קשוח בעקבות אורכו, קיימים חלקים לא מסודרים.

אהבתי לראות שימוש בתבניות עיצוב כמו Singleton במימוש המחלקה שאחראית על החיבור ל Firebase.

סך הכל העומס בקוד באמת הפקטור היחידי שתרם לרעה על חווית ה code review

***2.*   האם לדעתך ניתן לשפר תרגיל זה?**אולי תהליך ה code review יוכל לעבור בצורה יותר מעניינת אם היה מתקיים פרונטלית בקבוצות (זוגות של קבוצות), בצורה פרקטית כמובן (מעבר חלקי/התמקדות בהרצה לעומת בקוד נטו)

***3.*      הערות נוספות**

אין.