המחלקה להנדסת תוכנה

פרויקט גמר –

סביבת אוטומציה לבדיקת אפליקציית Keepers

Automation Environment for testing Keepers application

מאת:

חני בלאט - 208876193

ימי צחנוביץ - 209349679

מנחה אקדמי: גב' מיכל גולדשטיין	אישור:	תאריך:
אחראי תעשייתי: מר איליה סינטסין	אישור:	תאריך:
רכז הפרויקטים: ד"ר אסף שפיינר	אישור:	תאריך:

תוכן עניינים

מערכות ניהול הפרויקט

1	מאגר קוד	https://github.com/Yami12/Final-Project
2	יומן	https://trello.com/b/6tI7GSLl/automation-environment-for-keepe
5	סרטון גרסת אלפא	https://drive.google.com/file/d/1YllWVTWzyP84eSLZKNl1Ep4V9ZQ-JHXx/view

מילון מונחים, סימנים וקיצורים

לקוח - חברת Keepers המפתחת את אפליקצית Keepers.

Keepers - אפליקציה המגנה על ילדים ברשת, פותחה על ידי חברת Keepers.

אוטומציה (Automation) - שימוש באביזרים מכניים או אלקטרוניים, על-מנת לבצע סדרת פעולות, ברצף מתוכנן, ללא מגע יד אדם. אצלינו הפעולות הינן בדיקות.

Component behavior test (בדיקת התנהגות רכיב) - בדיקה האם פעולה מסוימת על רכיב סוים באפליקציה נותנת את התוצאה הצפויה על פי הגדרת הלקוח.

Feature test (בדיקת תכונה) - בדיקה על תכונה/ כלי מסוים של האפליקציה.

test flow- בדיקה של מספר טסטים בצורה סדרתית אחד אחרי השני כדי ליצור תרחיש שימוש מלא, כפי הגדרת הלקוח.

קובץ xml - קובץ השומר מידע (בפרויקט הנוכחי - טסטים) בצורה מקודדת באופן טקסטואלי.

app activity - מחרוזת של תווים המזהה מסך באפליקציה.

app package - מחרוזת של תווים המזהה את האפליקציה עצמה.

resource id - מחרוזת של תווים המזהה רכיב באופן חד חד ערכי.

driver (דרייבר) - תוכנה המאפשרת תקשורת בין 2 פלטפורמות שונות.

unit test (בדיקת יחידה) - בדיקה ברמת יחידות המערכת הקטנות שמאמתות את פעילותן התקינה של היחידות.

GUI (ממשק משתמש) - חלקה של מערכת החשוף למשתמש בה, כך שדרכו מתקיים הקשר בין המשתמש לבין תוכנה.

port (פתחה) - תהליך ספציפי שדרכו יכולות תוכנות להעביר נתונים באופן ישיר, במקום אמצעים אחרים

נאום המעלית

הפרויקט כולל מימוש לסביבת אוטומציה המאפשרת ניהול טסטים בצורה דינאמית עבור אפליקצית keepers, אפליקציה להגנה על ילדים ברשת.
הסביבה תומכת בשני סוגי טסטים - טסטים על תכונות - נכתבו עבור תכונות האפליקציה המדוברת, וטסטים על רכיבים - טסטים כלליים שבודקים התנהגות של רכיב וניתן להוסיף/ למחוק/ לעדכן אותם לפי הצורך.
תוצאות הטסטים ישלחו בקובץ html לכתובות המייל הנדרשות מהלקוח.

הסביבה מפותחת בשפת פייתון ומשתמשת בפלטפורמת appium.

מבוא

תופעת השימוש באינטרנט היא תופעה הולכת וגדלה בכל אוכלוסיות העולם ובפרט בקרב אוכלוסיית הילדים.

לפי דוח השנתון הסטטיסטי של המועצה לשלום הילד שפורסם בשנת 2017

- 87% מהילדים בגיל 7-17 גולשים באינטרנט שעתיים ומעלה ביום. כ-60% מהם גולשים 4 שעות ומעלה: שליש מהם גולשים 4-6 שעות ורבע ויותר גולשים 6 שעות ומעלה.
- 90% מבני הנוער בגיל 13-17 פעילים בווטסאפ, 75% פעילים בפייסבוק, 61% פעילים באינסטגרם וכמחצית פעילים בסנאפצ'ט

כמובן שבכל שנה האחוזים רק הולכים וגדלים.

תופעות בריונות ברשת ושיימינג גם כן הולכות ומתעצמות בעולם. כיום אין להורים גישה על דפוסי ההתנהגות של ילדיהם ברשתות החברתיות.
על פי הסקר של האו"ם, בכל יום, כ 160,000 ילדים בארה"ב לא הולכים לבית הספר מכיוון שהם חוששים מהטרדות ברשת.
אחד מתוך שלושה ילדים נפגע מאלימות ברשת על בסיס יומי. ילדים סופגים פגיעות נפשיות שילוו אותם כל החיים ובמקרים קיצוניים, המתרחשים לא מעט, אף ישימו קץ לחייהם.

מאות הודעות נשלחות ממכשיר הילד והמעקב הוא בלתי אפשרי.
חברת keepers פיתחה אפליקציה בשם keepers הזמינה לאנדרואיד ו iOS .
האפליקציה מותקנת על מכשירי הילדים וההורה, מנטרת את כל התקשורת הנכנסת והיוצאת של הילדים ברשתות החברתיות, מאתרת את כל ההודעות הנכנסות והיוצאות במכשיר הילד בכל הרשתות החברתיות הקיימות (פייסבוק, וואטסאפ ועוד...), לאחר מכן משתמשת באלגוריתם מתקדם של זיהוי שפה המבוסס על שימוש בבינה מלאכותית, וכך היא מזהה טקסטים שליליים במכשיר הילד כגון בריונות ברשת, מקרי חרם, הוצאה מקבוצות צ'אט, מצבים נפשיים, והתנהגות מינית לא הולמת.
האפליקציה מאתרת גם הטרדות מיניות, פדופיליה, סמים, סיגריות, אלכוהול ומקרי פשיעה.
כך המערכת מצליחה לאתר מכלל ההודעות אך ורק את אלו הרלוונטיות להורה ויכולה לדווח לו בזמן אמת

(תוך 20 דקות) על כל מקרה חריג שמתרחש, ובכך להחזיר את תחושת השליטה לידיו, ובו זמנית לשמור על פרטיות הילד.

האפליקציה מציעה כלים נוספים המסיעים להורה לוודא את ביטחון הילד, כגון:
התראה כשהבטרייה בטלפון הילד מתרוקנת מתחת ל-10%.
זיהוי מיקום הילד בזמן אמת ויידוע של ההורה כשהילד במקום מסוכן.
ניתוח זמני ואופן השימוש של הילד באפליקציות השונות וחסימתם, עפ"י הגדרות ההורה, בזמנים מסוימים.

קישור לאתר החברה: https://www.keeperschildsafety.net/

הפרויקט יכלול פיתוח כלי אוטומציה, אשר יאפשר לחברה לבדוק את האפליקציה שהיא מפתחת ביעילות ובאמינות.

תיאור הבעיה

דרישות ואפיון הבעיה

כיום בחברת keepers לא קיימת בכלל סביבת אוטומציה.
מה שכן קיים הן בדיקות JUnit שבודקות כל יחידה בנפרד ולא כמערכת שלמה.
הבעיה היא, שבדיקות יחידה לא מבטיחות מערכת תקינה. הן אינן עוסקות בבעיות שילוב, בעיות ביצועים ובעיות מערכתיות נוספות. כמו כן, בדיקות היחידה לא ישימו בהכרח דגש על קלטים סבירים מצד משתמש הקצה, משום שהן מתוכננות על ידי המפתח.
בנוסף לכך, ב Keepers בדיקות אלו מקיפות רק כ- 10% מהאפליקציה .

בנוסף, קיימים כמה טסטים בודדים שכתובים על פלטפורמת espresso אך היא פלטפורמה המיועדת בעיקר ל web , והאפליקציה פועלת על אנדרואיד ו- IOS.

באופן כללי, מכיוון שהבדיקות נעשות בצורה ידנית הן צורכות משאבים רבים, בעיקר של כוח אדם, וחשופות לטעויות אנוש רבות העלולות לפגום בתפקוד ואמינות האפליקציה.

הבעיה מבחינת הנדסת תוכנה

בשלבי העבודה השונים של הפרויקט נתקלנו במספר בעיות מעשיות, ביניהם:

1. איך ניתן לתקשר בין התוכנה שעל המחשב לאפליקציה שרצה על מכשיר האנדרואיד.
2. איך ניתן לדעת מה ה- app package וה- app activity בכדי להתחבר למכשיר הטלפון עליו מריצים את הטסטים.
3. איך ניתן לדעת את resourceId של כל רכיב על מנת לאתר אותו, לבצע עליו פעולות ולקבל ממנו מידע.

4. מה הדרך הטובה ביותר להכליל את הטסטים בכדי למנוע כפל קוד ולהפוך את הסביבה לדינאמית וניתנת להרחבה (Expandable).

תיאור הפתרון

תיאור הפתרון המוצע

הפרויקט כולל סביבת אוטומציה לאפליקציה באנדרואיד בלבד.

ממשק המשתמש מאפשר לבצע את הפעולות הבאות:

1. טסטים שבודקים את התכונות של האפליקציה - **Feature tests**
   1.1. הרצת flow של תכונה מסוימת
   1.2. הרצת flows של כל התכונות
   1.3. עדכון flow קיים
   1.4. מחיקת flow קיים

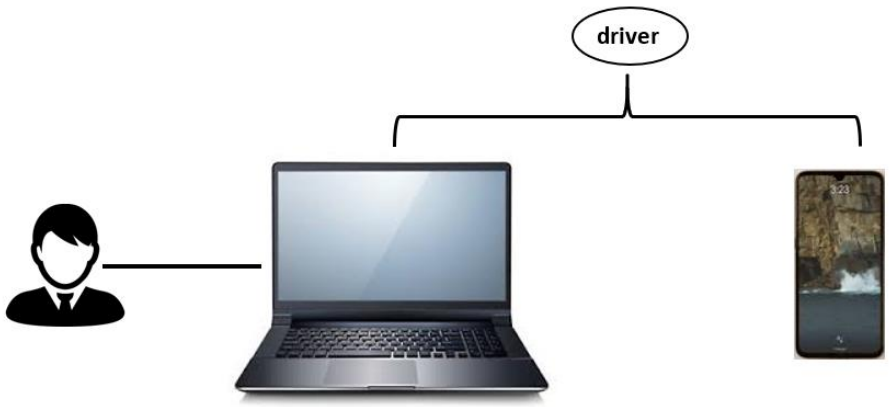
2. טסטים שבודקים את התנהגות הרכיבים של האפליקציה- **Component behavior tests**
   2.1. הרצת flow מסוים
   2.2. הרצת טסט ספציפי מתוך flow
   2.3. הוספת טסט חדש
   2.4. מחיקת טסט קיים
   2.5. עדכון טסט קיים

3. שליחת תוצאות הבדיקות לכתובות מיילים שהזין המשתמש.

תיאור המערכת

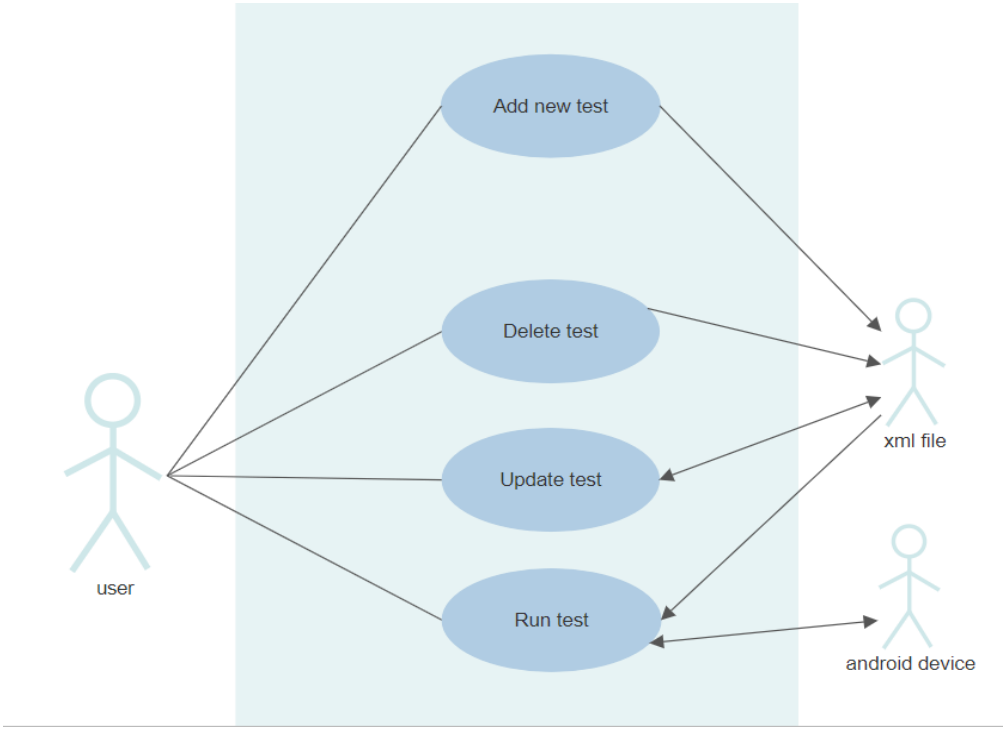
כדי לאפשר הרצת בדיקות על מכשיר האנדרואיד, צריך לקשר בין התוכנה למכשיר.

תקשורת זו מתבצעת באמצעות driver של Appium, כפי שמתואר בתרשים הבא:



כאשר המשתמש רוצה להשתמש במערכת, הוא צריך למלא בממשק המשתמש פרמטרים שונים, כגון:
שם מכשיר האנדרואיד, הגרסה שלו, port בו רוצים שה driver יאזין וכו'.
בעזרת פרמטרים אלו, Appium מקשרת בין פלטפורמת הבדיקות למכשיר האנדרואיד.

תרשים תרחישי השימוש של המערכת:



פירוט הפתרון

הפרויקט נכתב על סמך הגרסה הנוכחית של האפליקציה ורשימת בדיקות נדרשות, כפי שהתקבלו מהלקוח.
הטסטים מחולקים ל- flows. **דוגמה מתוך קובץ הבדיקות מצורפת בנספחים**.
מכאן ואילך, כאשר החברה מעלה גרסה חדשה של האפליקציה, הממשק מאפשר לה להשתמש בבדיקות הקיימות מהגרסה הקודמת, לעדכן אותן, להוסיף ולמחוק, מה שמאפשר לבדוק את הגרסה החדשה ביעילות ובקלות.

המערכת תומכת בשני סוגי בדיקות: בדיקות התנהגות רכיב, ובדיקות תכונות.

כאשר התוכנה מריצה בדיקה מכל סוג שהוא, היא פותחת את האפליקציה ומדמה עליה פעולות אנושיות, כגון לחיצה על כפתור, הכנסת טקסט, וכו'.

לאחר הרצת הבדיקה, תוצאותיה נשמרות במערכת.
התוצאות נשלחות ללקוח בפורמט הבא: flow name , test name , result
במידה והתוצאה שהתקבלה מהבדיקה תואמת את התוצאה המצופה, הטסט עבר בהצלחה ונרשם ב- result שלו: "OK"
במקרה והטסט נכשל, נרשם ב- result שלו: "ERROR" ואת פרטי השגיאה.

בדיקות התנהגות רכיב:

טסטים מסוג זה נשמרים בקובץ xml, בפורמט הבא:

```
- <flowes>
  - <flow>
      <name/>
    - <tests>
      - <test>
          <name/>
          <resourceId/>
          <appActivity/>
          <type/>
          <actionType/>
          <expectedResult/>
        </test>
      - <test>
          <name/>
          <resourceId/>
          <appActivity/>
          <type/>
          <content/>
          <expectedResult/>
        </test>
      </tests>
    </flow>
  + <flow>
  </flowes>
```

כל flow מורכב מהשדות הבאים:

1. שם
2. רשימה של טסטים.

כל טסט מורכב מהשדות הבאים:

1. שם
2. מזהה הרכיב
3. המסך בו ממוקם הרכיב
4. סוג הרכיב. בשלב זה מומשה תמיכה ב 3 סוגי רכיבים :

   כפתור - Button, תיבת טקסט - Entry, תיבת סימון - CheckBox
5. פרטים נוספים בהתאמה לסוג הרכיב. בשלב זה המערכת תומכת בפרטים הבאים:

   כפתור, תיבת סימון - סוג הפעולה (actionType)

   תיבת טקסט - תוכן (content)
6. תוצאה רצויה (במקרה ומתבצעת הפעולה הנוכחית על הרכיב הנוכחי)

בשלב זה, מומשה תמיכה בפורמט הבא:

סוג התוצאה: תוכן. לדוגמא: "toastMessage:Please accept the terms".

- כאשר המשתמש בוחר להוסיף/ לעדכן/ למחוק טסט, הטסט מתווסף/ מתעדכן/ נמחק מקובץ הטסטים הקיים.
- כאשר המשתמש בוחר להריץ flow מסויים, לפי שם הflow, התוכנה מריצה את כל הטסטים שבו אחד אחרי השני.

הרצת ה flow מתבצעת כך:

1. אתחול ה driver והתחברות לפלטפורמת Appium

2. העברת האפליקציה ל app Activity של הטסט הראשון ב- flow
3. עבור כל טסט:
   3.1. הפעלת הפעולה המבוקשת על הרכיב המבוקש, לפי סוג הרכיב.
   3.2. בדיקה האם התוצאה שהתקבלה תואמת לתוצאה המבוקשת.

- כאשר המשתמש בוחר להריץ טסט מסוים, לפי שם הflow ושם ה test, התוכנה מריצה את הטסט בצורה הבאה:
  1. העברת האפליקציה ל app Activity של הטסט
  2. הפעלת הפעולה המבוקשת על הרכיב המבוקש
  3. בדיקה האם התוצאה שהתקבלה תואמת לתוצאה המבוקשת

בדיקות תכונות:

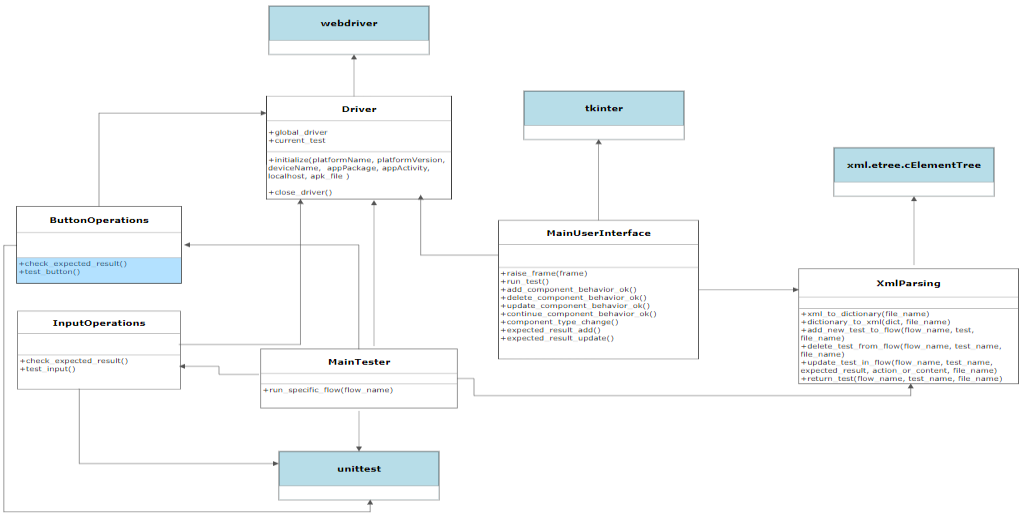
בדיקות אלו כוללות תהליכים חיצוניים, שלא מתרחשים באפליקציה עצמה. לדוגמה: הודעה שנשלחה ב whatsapp.

כרגע, מטרת הפרויקט היא לממש את הבדיקות עבור התכונה שמזהה הודעות פוגעניות שנשלחות ברשתות החברתיות.

כשנשלחת הודעה ממכשיר הילד/ למכשיר הילד, האפליקציה מנתחת את ההודעה, במידה וזוהו מילים פוגעניות היא שולחת את כל תוכן ההודעה, יחד עם אינפורמציה נוספת, למכשיר ההורה.

לצורך מימוש בדיקות אלו, המערכת צריכה להתממשק עם הרשתות החברתיות ולבדוק מקצה לקצה האם ההודעה שהועברה להורה, כוללת את כל האינפורמציה בצורה נכונה ומדויקת.

להלן דיאגרמת המחלקות:



תיאור המסכים מצורף בנספחים.

כלים בהם השתמשנו לפתרון

Appium - כלי אוטומציה המהווה קוד פתוח להפעלת טסטים באנדרואיד או iOS באמצעות דרייבר אינטרנטי.

uiautomatorviewer - כלי GUI לסריקה וניתוח של רכיבי ממשק המשתמש של יישום אנדרואיד.
בכדי להקים אוטומציה של כל יישום אנדרואיד באמצעות Appium, המפתח צריך לזהות את האובייקטים ב- AUT (יישום שנבדק).
באמצעות כלי זה ניתן לבדוק את ממשק המשתמש של יישום אנדרואיד בכדי לקבל את התכונות השונות של האלמנט (resource id, טקסט וכד').

PyCharm - סביבת עבודה המיועדת לפיתוח בשפת python.

Python - שפת תכנות דינאמית

tkinter - ספריית פיתון המספקת ערכות כלים לעבודה עם ממשק משתמש (GUI).

Unittest - ספריית פיתון המספקת פלטפורמה לכתיבת אוטומציה.

xml.etree.cElementTree- ספריית פיתון המספקת כלים לעבודה עם קבצי xml.

webdriver- ספריית פיתון המספקת תשתית לתקשורת עם Appium

תכנית בדיקות

בדיקה	הסבר
בדיקת תקינות של קובץ האקסמל	נוודא שאין 2 flows עם שם זהה ואין 2 טסטים באותו flow עם שם זהה. נוודא שטסט שהמשתמש מחק אכן נמחק מהקובץ, טסט שהוא הוסיף אכן נוסף וטסט שהוא עדכן אכן עודכן.
בדיקת שהרצת flow נעשית ע"פ הסדר	נריץ flow, נצפה במכשיר ונוודא שהפעולות קורות לפי הסדר.
בדיקות קוד	קוד גנרי ויעיל
בדיקת התקשורת בין התוכנה למכשיר	לפעמים לא ניתן להריץ את הטסטים בגלל בעיה שהתרחשה ב driver של Appium
בדיקת הרשאות גישה למכשיר האנדרואיד	במכשיר האנדרואיד צריך לאשר גישה חיצונית אליו והפעלת תוכניות מרחוק
בדיקה שכל פרטי הטסט הוכנסו ושכל הפרטים תקינים	אם לא הוכנסו כל הפרטים כנדרש, הבדיקה עלולה להכשל

סקירת עבודות דומות בספרות והשוואה

לגבי החלק הראשון של הפרויקט, בדיקות התנהגות רכיב:
מסקר שוק ערכנו, גילנו שלא קיימת בשוק אפליקציה/ מערכת דומה.
מה שכן קיימות הן אפליקציות שמבצעות פעולות אוטומטיות על המכשיר לפי הגדרת המשתמש.

- **IFTTT:** האפליקציה מאפשרת ליצור "מתכונים" קבועים מראש אשר באמצעותם יכול המשתמש להתנות פעולה אחת באחרת. למשל, ניתן להגדיר כי ברגע שנצא משטח העבודה, יבוטל הפרופיל השקט.

-**Tasker:** האפליקציה מאפשרת למשתמש להגדיר למכשיר משימות בתנאים מסוימים. כמו למשל: כאשר מגיע בדואר האלקטרוני טופס הזמנה דיגיטלי, שלוף את המספר הסלולרי של הלקוח מהטופס ושלח לו הודעת וואטסאפ עם הכיתוב: "תודה, טופס ההזמנה שלך התקבל בהצלחה ומטופל על ידינו".

מטרת אפליקציות אלו אינה בדיקת תפקוד האפליקציה, אלא רק ביצוע פעולות אוטומטיות בתנאים מסוימים.
מטרה שונה לגמרי ממטרת הפרויקט שלנו.

לגבי החלק השני, בדיקות תכונות האפליקציה:
כלי האוטומציה שנפתח הם אינדיבידואליים לאפליקציית keepers ועונים רק על הדרישות שלה.
לכן, אין בשוק כלי שעונה על צרכי האפליקציה הספציפית הזו ומה שנפתח ייחודי ולא קיים כלי דומה לו.

סיכום

לאחר לימוד מעמיק של עולם האוטומציה על מובייל והכרות עם הכלים הנדרשים, התקנו סביבת Appium במחשב וקישרנו בין פייתון ל Appium באמצעות ספריות של פיתון.

עד כה התעסקנו במימוש טסטים מסוג התנהגות רכיב.
ישנן 4 אופציות המוצגות למשתמש בסוג טסטים זה: הרצת טסט, הוספת טסט, עדכון טסט ומחיקת טסט.
בשלב זה ניתן להריץ flow ולא טסטים בודדים.

המערכת תומכת כעת ב 3 סוגי רכיבים ובפעולות ספציפיות עליהם בצורה הבאה:

- רכיב כפתור: פעולות - לחיצה/ מעבר על הכפתור, תוצאה רצויה - הודעת שגיאה למשתמש.
- רכיב תיבת טקסט: פעולות - הכנסת טקסט, תוצאה רצויה - הודעת שגיאה למשתמש.
- רכיב תיבת בחירה: פעולות: בחירה, תוצאה רצויה - הודעת שגיאה למשתמש.

בשלב הבא:

1. נוסיף מימוש לטסטים של תכונות
2. נממש יצירת דף html המכיל את תוצאות הבדיקות, ושליחתו לכתובות המייל המבוקשות
3. ניתן אופציה להרצת טסט בודד מתוך flow בטסטים של התנהגות רכיב
4. נוסיף אפשרות לבדוק רכיבים נוספים
5. נוסיף אופציות נוספות לפעולות על רכיב
6. נטפל במקרי קצה וחריגים

נספחים

תיאור המסכים:

דוגמא מתוך קובץ הבדיקות של הלקוח:

flow

test

	Error handling: Parent's registration page errors		
	Parent's name field - empty		
13.1	Open the Keepers app	The parent\ child page appeared	PASS
13.2	Click the checkbox -I accept the terms	V appeared in the check box	PASS
13.3.2	Click the parent button	A personal information page will open	PASS
13.4.1	Enter shira14@gmail.com In the email field	shira14@gmail.com will appear In the email field	PASS
13.4.2	Enter 123456 into the password box	******will appear in the Password field	PASS
13.4.3	The name field leave empty		
13.4.4	Press "Create a new account	An orange error message "Add your name" will appear under the name field	PASS
	Error handling: Invalid email address		
14.1	Open the Keepers app	The parent\ child page appeared	PASS
14.2	Click the checkbox -I accept the terms	V appeared in the check box	PASS
14.3.1	Click the parent button	A personal information page will open	PASS
14.4.1	Enter shira in the Name field	shira will appear in the Name field	PASS
14.4.2	Enter shira14@.com In the email field (incorrect email template)	shira14@.com will appear In the email field	PASS
14.4.3	Enter 123456 into the password box	******will appear in the Password field	PASS
14.4.4	Press "Create a new account	An orange error message "Invalid email address" will appear under the email field	PASS
	Error handling: 5-character password		
15.1	Open the Keepers app	The parent\ child page appeared	PASS
15.2	Click the checkbox -I accept the terms	V appeared in the check box	PASS
15.3.1	Click the parent button	A personal information page will open	PASS
15.3.2	Enter shira14@gmail.com In the email field	shira14@gmail.com will appear In the email field	PASS
15.4.1	Enter shira in the Name field	shira will appear in the Name field	PASS
15.4.3	Enter 12345 into the password box	*****will appear in the Password field	PASS

תרשימים וטבלאות

א. טבלת סיכונים

הסיכון	חומרה	מענה אפשרי
קיימים באפליקציה רכיבים בלי מזהה	3	בקשה מהלקוח להוסיף מזהה
טסטים שלא ניתנים להכללה	1	כתיבת קוד נפרד לטסטים אלו

חוסר התמצאות בממשקים של האפליקציות אליהן מתממשקת האפליקציה	2	למידה של כל ה API באמצעות האינטרנט
אי נגישות לשרת האמיתי	1	דימוי גישות לשרת דרך האפליקציה postman

ב. טבלת דרישות

דרישות פונקציונליות

מס' דרישה	תיאור
1	הרצת טסטים לבדיקת התנהגות רכיבים
2	הוספת טסט לטסטים הקיימים
3	עדכון טסט קיים
4	מחיקת טסט קיים
5	הרצת טסטים לבדיקת תקינות תכונת שליחת וקבלת הודעות
6	קבלת תוצאות אמיתיות של הטסטים לכתובות מייל מבוקשות
7	שיתוף פעולה עם אפליקציות נוספות כמו whatsapp

דרישות נוספות לא פונקציונאליות:

- **ניידות:** ניתן לבצע את הרצת הטסטים על כל מכשיר אנדרואיד
- **שימושיות:** תהליך הרצת טסטים פשוט ומהיר.
- **תחזוקה:** הכללת הטסטים כמה שיותר כך שניתן לשנותם בלי שינוי בקוד או בשינוי מינימלי. מה שמפחית את התלות במפתחים.

תכנון הפרויקט

קישור ליומן: https://trello.com/b/6tI7GSLl/automation-environment-for-keepe