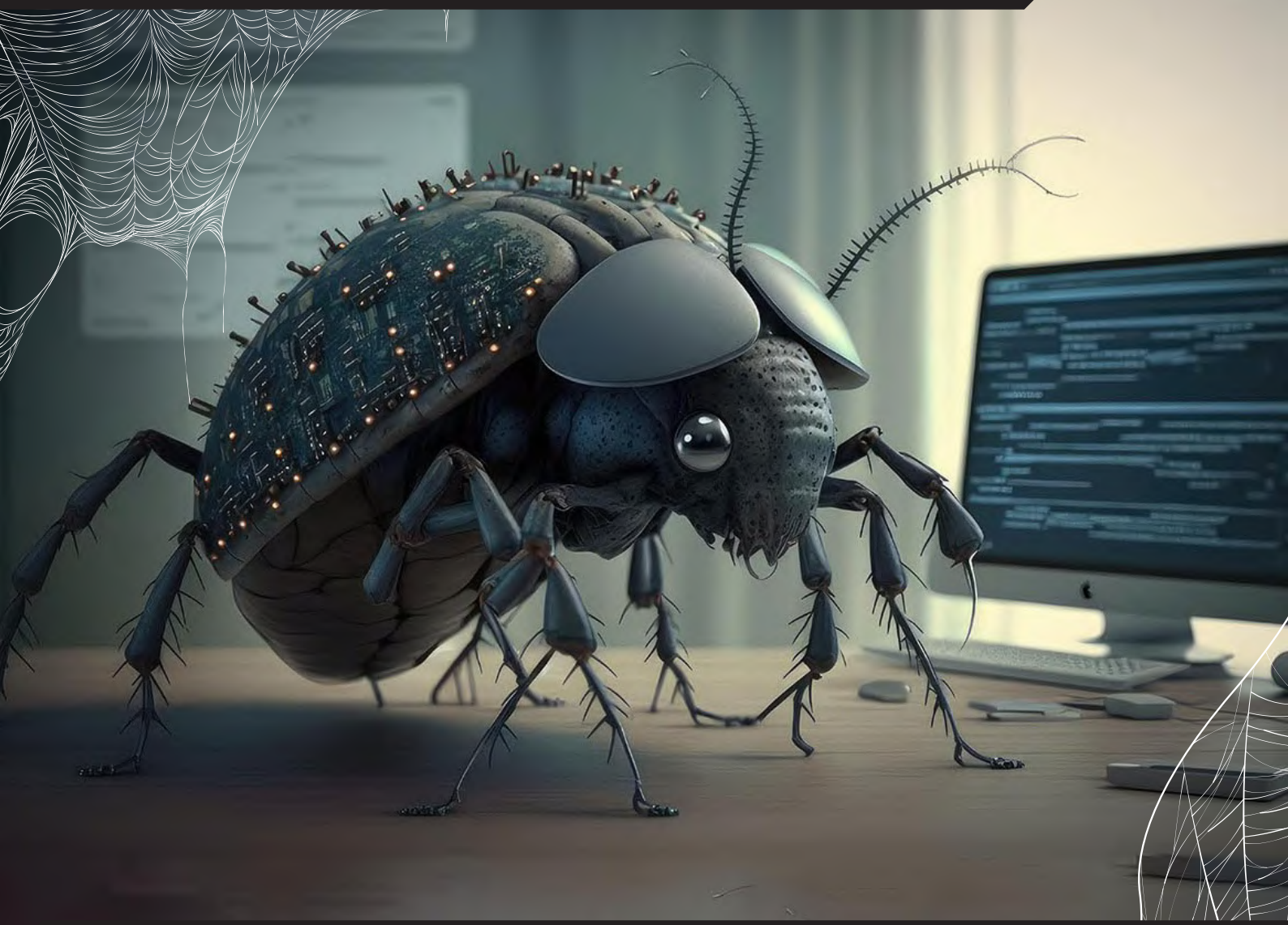
מגזין
עולם הבדיקות
רבעון שני 2023 | גיליון מס' 33
www.testingworld.co.il

איזה סוג של אנשי QA אנחנו
ניסים סער אריאל

ראיון עם מנהלת בדיקות – ג'ני רויטמן
ניצן גולדנברג



אוטומציה מורכבת פחות עם פריימוורק הבדיקות Gauge
לירן יושינסקי

תשתית אוטומציה יעילה בעבודה נכונה עם XML
נתי צדוק

דבר העורך
ניצן גולדנברג

ניצן גולדנברג

מזה 7 שנים בתחום בדיקות התוכנה, תפקיד נוכחי מהנדס בדיקות בכיר בחברת SeatGeek, מנהל קבוצת ה-AB של ארגון ITCB® המוביל הראשי של קבוצת המיטאפ TestIL, מרצה בקורסים לבודקי תוכנה

קוראים יקרים,

"What you get by achieving your goals is not as important as what you become by achieving your goals – Achievement."

Henry David Thoreau

הפעם אני רוצה להביא לכם קוראים וקוראות יקרים משפט מאוד חשוב ובעל השפעה שלאחרונה הרגשתי מאוד מחובר אליו, "מה שאתה מקבל על ידי השגת היעדים שלך אינו חשוב כמו מה שאתה הופך להיות על ידי השגת היעדים שלך".

יצא לי לאחרונה להשתתף בכנס GA -General Assembly של ארגון הבודקים העולמי ISTQB® ולהשתתף בפעם הראשונה בקבוצת עבודה של צוות מרקטינג של הארגון. מיותר לי לציין שזו הייתה חוויה לא מהעולם הזה שכן ייצגתי את הבורד הישראלי ITCB® בישיבות עם כל נציגי הבורדים מכל העולם.

יצא לי לשוחח ולהציג את הפעילויות שיש לנו בבורד הישראלי להציע לקהילת הבודקים כגון המגזין, המיטאפים והסדנאות וכן תחרות הבדיקות הישראלית ISTC. הנציגים מכמה מדינות כל כך התלהבו מהרעיון של התחרות שהם ביקשו שאעזור להם לארגן במדינות שלהם תחרות דומה.

אני חייב לומר שלהיות במעמד שמדינות אחרות מעוניינות לקדם מה שאני מקדם גורם לי להבין שהשגת היעדים שלי הופכים אותי למה שאני כיום.

מונח לפניכם גליון מס 33 של מגזין "עולם הבדיקות".

בגליון זה תוכלו למצוא מגוון רחב של מאמרים חדשים, בנוסך לטורים המעולים והקבועים שלנו:

"אוטומציה מורכבת פחות עם פריימוורק הבדיקות Gauge" - לירן יושינסקי

ניסים סער אריאל מביא לנו עוד מאמר מעניין והפעם "איזה סוג של אנשי QA אנחנו"

"תשתית אוטומציה יעילה בעבודה נכונה עם XML" - נתי צדוק

יש לנו גם פינה חדשה לתקופת ניסיון של "בודקים בכיף" - חידות ותשבצים, נשמח לשמוע את דעתכם בנוגע לטור

בנוסף, יש לנו את הטורים הקבועים: מחפש צרות, מה הסקרים אומרים, ראיון עם מנהל/ת בדיקות, האנציקלופדיה לבדיקות, עושים QA לקריירה, סקירת כלים ומה הסקרים אומרים.

אנו נשמח לקבל בקשות לנושאים חדשים ומעניינים למאמרים, צרו עימנו קשר במייל

magazine@testingworld.co.il

קריאה מהנה,
ניצן גולדנברג

עדכון מוסמכים בישראל

סה"כ מוסמכים בבורד הישראלי ITCB® - 9,504

מתוכם 728 "מצטיינים" אשר ענו מעל 90% תשובות נכונות.

28 מצטיינים ברבעון הראשון של שנת 2023

הודעות ועדכונים:

הסילבוס החדש של הסמכת ה-CTFL 4.0, מוזמנים לעיין בסילבוס באתר של הארגון העולמי ISTQB® בקישור

הפודקסט החדש TestIL Podcasts מבית ITCB® יצא לדרך, מוזמנים לעקוב אחרינו בקישור

אני שמח לעדכן כי קבוצת המיטאפ TestIL עברה את סף ה-4000 חברים

קבוצת הדיונים בבדיקות בפייסבוק TestIL עברה את סף ה-11,000 חברים

מו"ל
Israeli Testing Certification Board
ITCB®

ניהול המגזין
יאן ברון, *iMDsoft*

ניהול התוכן
קובי הלפרין, *Nokia*

עורך ראשי
ניצן גולדנברג, SeatGeek

עיצוב גרפי
בית נלי מדיה
סטניסלב קולנקו
www.beitnelly.com

יצירת קשר
אימייל:
magazine@testingworld.co.il

הרשמה
http://bit.ly/TW-Reg
פקס:03-6176605
כתובת: ברוך הירש 14 בני ברק 51202

www.testingworld.co.il

www.itcb.org.il

תוכן העניינים

דבר העורך 2

עושים QA לקריירה | איילת מלמד כהן 4

אוטומציה מורכבת פחות עם פריימוורק הבדיקות Gauge
לירן יושינסקי 6

סקירת כלים Xray Exploratory | ניצן גולדנברג 10

האנציקלופדיה לבדיקות | קובי יונסי 12

איזה סוג של אנשי QA אנחנו | ניסים אריאל 15

מחפש צרות | מיכאל שטאל 17

ראיון עם מנהלת בדיקות ג'ני רויטמן, igentify 19

תשתית אוטומציה יעילה בעבודה נכונה עם XML | נתי צדוק .. 21

מה הסקרים אומרים? | ניצן גולדנברג 24

משולחנו של שביט –
אל תפתח מטרייה לפני שהתחיל לרדת גשם | שביט ג'רסי 25

תחרות הבדיקות הישראלית ISTC 2023 26

בודקים בכיף 28

דף העורכים 29

עולם הבדיקות נכתב ע"י בודקים עבור בודקים

ITCB® מקדמים את קהילת הבודקים בישראל

מגזין עולם הבדיקות

עולם הבדיקות הינו מגזין רבעוני.
כל הזכויות שמורות. זכויות היוצרים על חומר שפורסם על ידי המפרסם הינן רכושו של המחבר. הדעות המובאות במאמרים והתוכן לא בהכרח משקפים את דעת המפרסם. המחברים הינם האחראים הבלעדיים על תוכן מאמרם. מובהר כי העתקה ו/או נטילה שיטתית של מידע מהמגזין לצורך פעילות מסחרית ו/או עסקית, או לצורך כל פעילות אחרת שיש בה כדי לפגוע בפעילות העמותה, אסורה בהחלט.
לקבלת אישור לשימוש בתוכן צור קשר בדוא"ל **magazine@testingworld.co.il**.

עושים QA לקריירה | איילת מלמד כהן

איך לטפח Quality Mindset מחוץ לארגון ה-QA

איכות היא לא רק טסטים, באגים או בקרה לפני שחרור קוד לפרודקשן, אך ברוב הארגונים, כשמדברים על איכות מתכוונים לזה.

Quality Mindset היא גישה חוצת צוותים שמתפתחת כשההסתכלות על איכות היא מתוך פרספקטיבת הלקוח והגדרתו למוצר או שירות איכותיים.

כשזה מתממש, בכל צוות ומחלקה מרגישים שאיכות היא ערך שמקבל ביטוי משמעותי **בשיח הארגוני ובפוקוס,** מרמת ההנהלה הבכירה ומטה. כשאיכות היא ערך, מדדי איכות הם חלק מהיעדים של כל הצוותים המשתתפים בתהליך הדליברי ובתקשורת עם הלקוחות, ותהליכי שיפור מתמיד גם הם חלק מהתרבות וההתנהלות.

מחקרים הוכיחו שמיינדסט כזה הוא אחד ממנועי הצמיחה של הארגון - בהכנסות, בנתח שוק, בשימור לקוחות ובניית מונטין.

זאת גישה שגם אם לא קיימת בארגון שלכם, תוכלו להשפיע על הדרך לייצר כזאת. בכך שהפוקוס שלכם הוא לא רק על ה-Execution במיקרו, אלא גם על איך זה מתחבר לארגון ולביזנס במאקרו, תפתחו את הקריירה שלכם, תייצרו לעצמכם עניין, אתגר וחשיפה גם מחוץ לארגון הפיתוח.

לא בכל ארגון זה פשוט לביצוע, ולהשפיע מחוץ לגבולות הגדרת התפקיד שרגילים אליו זה אתגר, אך זה יהיה לחלוטין שווה את זה אם אתם שואפים להתפתחות מקצועית, ארגונית או ניהולית בתחום שלכם, בארגון שלכם או מחוצה לו.

הנה כמה רעיונות לאיך תוכלו לדחוף לפיתוח מיינדסט כזה מחוץ לארגון ה-QA. **כולם** נכתבו מניסיוני בפועל כחלק מאסטרטגית האיכות לארגון:

1. **תמחישו איך איכות נתפסת בעיניי הלקוח נכון להיום**

תוודאו שיש מנגנון שמאפשר לאסוף ולנהל פידבקים מלקוחות ושהפידבקים מגיעים לכל המשתתפים בתהליך הפיתוח והדליברי - מפתחים, בודקים, מנהלי פרוייקטים, פרודקט.

הפידבקים יכולים להגיע דרך מחלקת ה-Customer Success, דרך ה-Support, המוצר עצמו או דרך צוות ייעודי שזה תפקידו. כשהפידבק מהלקוחות מגיע לכל השותפים בתהליך, נראה יותר שאלות לפני ביצוע ושמשתמשים בדאטה הקיים או שמשכללים אותו כדי להבין את **ההשפעה של כל שלב בפיתוח על חווית הלקוח.**

זה מחבר את כולם למחיר האיכות עד ללקוח ומחדד את השאלה אצלם - למה לקוחות שמים לב?.

כשיש לארגון לקוחות, יש דרך לאסוף את המידע - לפנות ללקוחות בשאלות שביעות רצון ספציפיות למוצר ו/או לנתח את התלונות והדיווחים מכל המקורות. אלו למעשה הפידבקים. אם אין מספיק לקוחות, תוכלו לייצר משתמשי ביתא או כמו בדוגמא הבאה - להקים צוות:

לפני מס' שנים עבדתי בסטאראפ שפיתח מערכת חדשה להיתוך מידע עבור רשויות אכיפה כמו משטרה. כיוון שאף אחד בארגון בו עבדתי לא הגיע מרקע של חוקר במשטרה או שב"כ שהיו לקוחות המטרה שלנו, אז לא יכולנו לוודא שמה שפיתחנו אכן עומד בציפיות הלקוח בהיבט איכות וערך.

כמובילת איכות זה היה קריטי לדעת מה אנחנו בעצם בודקים, לכן בניתי צוות של חוקרים לשעבר שעבדו על המערכת לאחר שמידלנו לתוכה דאטה של פרשיות פשע אמיתיות. התובנות שצוות הפרודקט והנהלת הארגון הפיקו בזכות הפידבקים האלו שינו את פני המוצר.

אם אנחנו רוצים להשפיע על המיינדסט נתחיל מפידבקים שממחישים את רמת האיכות והחוויה הנתפסת היום ובאותם כלים, נמשיך ונמדוד את השיפור.

2. **כל מה שקורה בפרודקשן לא נשאר בפרודקשן**

תמשיכו עם מה שמעניין את כולם ברמת הארגון - אירועי פרודקשן ואסקלציות מלקוחות.

יש ארגונים שבהם עושים הפקת לקחים, אך לא אוכפים את המסקנות. תוודאו שלכל באג משמעותי שקורה בפרודקשן עושים RCA (Root Cause) Analysis ושמפרסמים אותו לרשימת תפוצה הרלוונטית. אם אין מי שיזם RCA, תובילו אתם. אם אף אחד לא מוודא שהמסקנות אכן יושמו, כדאי שתהיו אתם אלו שמוודאים, ויחד איתכם, לכל אלו, תרתמו את מנהלי הפיתוח.

כדי להפנות את תשומת הלב של הפרודקט והפיתוח לאיכות מעבר לתכולה בכל ספרינט ועמידה בלוחות זמנים, תוכלו לעשות Zoom-Out דרך אנליזה של הבאגים שהגיעו מפרודקשן לפי פילוחים שרלוונטיים למוצר שלכם כמו לקוחות, איזורים במוצר, גיאוגרפיות וכו'. קחו איתכם מישהו מצוות התמיכה, או ה-Customer success ואם אין שותף, רוצו על זה עם המנהלים שלכם ותגדירו איך תראה התוצאה ומה מחפשים.

אם כמות הלקוחות והטיקטים גבוהה, בחרו יחד כמה לקוחות אסטרטגיים, ותעשו את זה רק עבורם.

> **"אם אנחנו רוצים להשפיע על המיינדסט נתחיל מפידבקים שממחישים את רמת האיכות והחוויה הנתפסת היום ובאותם כלים, נמשיך ונמדוד את השיפור."**

התוצאות יעזרו להשפיע על סדרי העדיפויות של הפרודקט והפיתוח ויתרמו לניהול סיכוני גרסא ותכנון בדיקות יעיל. אם תחזקו את הממצאים דרך הפידבקים מהלקוחות, זה יתקף את המסקנות משמעותית.

יש ארגונים שבהם המפתחים והמנהלים שמשחררים קוד לפרודקשן, הם לא אלו שמטפלים באסקלציות מהשטח באזור הקוד שלהם, אלא יש צוות אחר שמנטר את הפרודקשן החל מהרגע שהקוד שוחרר. רמת החיבור והפוקוס של המפתחים למשמעות של איכות כפי שבאה לידי ביטוי אצל הלקוח, לא תהיה גבוהה בהגדרה.

לכם יש מנדט בתחום ה-QA, השפעה על מה קורה בפיתוח אך לא שליטה על מי יפתור איזו תקלה. התפקיד שלכם הוא לעורר למודעות, להציג את ההשפעות, ליזום פתרונות לשיפור והתייעלות גם אם קבוצה אחרת תממש אותם ובכל שלב - לשתף את המנהלים

איילת מלמד כהן

בעלת ניסיון של 19 שנה בעולם ה-QA, רוב השנים כמנהלת בכירה בסטארטאפים וחברות גדולות. בשנים האחרונות מאמנת לפיתוח מנהיגות עצמית וניהולית, שיפור מיומנויות ניהול, כניסה לתפקיד חדש או ההתמודדות עם אתגרי התפקיד הקיים. בנוסף לאימון, איילת מלווה סטארטאפים בכל הקשור לניהול איכות, בונה צוותים ואסטרטגיות QA מותאמות לארגון. מאמנת מוסמכת בינ"ל, בעלת תואר ראשון ושני במנהל עסקים וכלכלה מאוניברסיטת בר-אילן. אמא לשלושה וזוקפת לזכותם חלק גדול ממיומנויות הניהול שלה.

עושים QA לקריירה | איילת מלמד כהן

הישירים/עקיפים ולהראות למה כדאי לעשות את השינוי, ואיך זה יועיל להם. דרך נוספת ויעילה להשפיע היא להגדיר מדד איכות - לכמות אירועי הפרודקשן שמשפיעים על לקוחות.

כשרוצים לייצר מיינדסט חדש צריך לקנפג את הקיים.

3. **תרחיבו את הגדרת האיכות מהמוצר ל- Quality of Service, Quality of Data ותמדדו אותם מנקודת מבטו של הלקוח**

Quality Mindset מחוץ לארגון ה-QA קורה כשאנו עוזרים לממשקים שלנו להתמקד באיכות התהליך או השירות, הנה כמה דוגמאות:

לפני שחרור גרסא: תגדירו אילו בדיקות הם חלק מה-Definition of Done של כל משימת פיתוח עבור המוצר ואת סט הבדיקות שירוץ לפני שהגרסה מגיעה ל-QA. כדאי להגדיר את שני הנושאים הללו, ביחד עם החברים בצוות פיתוח ולשדרג את התהליך תוך כדי תנועה, בכך גם משתפרת האיכות וגם מצטמצם הפינג-פונג בתקשורת עד לתחילת בדיקות. אם איכות הדאטה (שלמות, נכונות, סנכרון, פרטיות וכו') היא קריטית, תוודאו שיש בדיקות שמוודאות את זה כחלק מה-Pipeline. בשחרור: אם רלוונטי עושים Gradual rollout לגרסאות חדשות, ובכל מקרה מי שפיתח הוא זה שמנטר את ההשפעה על הלקוחות ותומך בתקלות. נוודא כי יש תהליכי התאוששות מהירים במקרה של נפילות, מדידות של Down-Time, כמות Errors וסוגיהם.

לאחר שחרור הגרסא: נוודא שיש SLA לתיקוני באגים קריטיים, הגדרה ברורה של מי מתקן ומתי פורסים תיקון ושיש מי שמודד/אוכף את ה-SLA הזה כדי לצמצם את הפגיעה באיכות השירות ללקוחות.

"כשרוצים לייצר מיינדסט חדש צריך לקנפג את הקיים"

"Quality is not an act, it is a habit" (אריסטו)

במשך שנים המשפט הזה היה בחתימת המייל האוטומטית שלי ובכל מצגת שהעברתי, וכך היה בכמה ארגונים. זה היה המשך ישיר של המסרים והפעולות שצלחו כי **קרו בהדרגה**, ובכך גם האסטרטגיה התממשה.

אז איך מתחילים? תבחרו משהו אחד, נקודה אחת מאלו שכתובות כאן בכל רבעון ורק בה תתמקדו. בסיום, תלמדו ממנה על עצמכם ועל מה שהצליח ותתקדמו לבאה.

הצטרפו לצוות המוביל את מגזין עולם הבדיקות

מעוניינים להצטרף לעשייה? התפנה מקום בצוות המגזין!

הפעילים במגזין הינם אנשי מקצוע בתחום הבדיקות שפועלים בהתנדבות למען קהילת הבודקים בארץ.

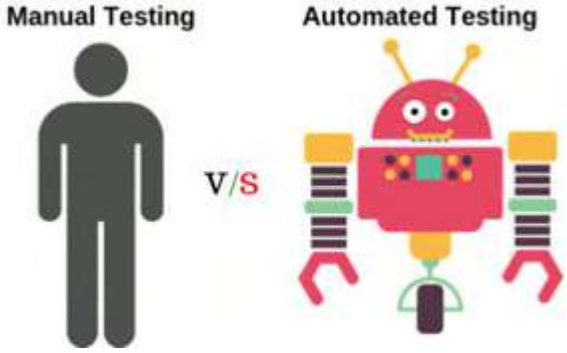
לקבלת פרטים נוספים פנו לניצן גולדנברג:
magazine@testingworld.co.il

אוטומציה מורכבת פחות עם פריימוורק הבדיקות Gauge | לירן יושינסקי

נתחיל ב"למה בכלל צריך אוטומציה?"

אענה על כך במשפט:

כדי להפחית את מאמץ הבדיקה הידנית ב-cycle הרגרסיה ובכך להגדיל את החזר ההשקעה (ROI).



קיימות מספר סיבות משמעותיות להשתמש בבדיקות אוטומציה בתהליך הפיתוח והבדיקה של תוכנה. הנה כמה מהסיבות העיקריות:

1. **חסכון בזמן ומשאבים:** בדיקות אוטומציה מאפשרות לבצע בדיקות באופן אוטומטי, מה שמפחית את הצורך בבדיקות ידניות. השימוש בבדיקות אוטומטיות מפחית את הזמן והמאמץ שנדרשים מהצוות לביצוע בדיקות חוזרות ומורכבות.
2. **דיוק ואמינות:** בדיקות אוטומציה יכולות להיות מדויקות ואמינות ביותר. כאשר הם מוגדרים מבוצעים כהלכה, ניתן להיות בטוחים שהתהליך יבוצע בדיוק ולא יתרחשו שגיאות אנושיות. זה יכול לגרום להפחתת באגים ולשיפור איכות התוכנה.
3. **ניתוח יעיל:** בדיקות אוטומציה מאפשרות לבצע ניתוח מקיף ופרקטי של התוכנה. ניתוח זה מאפשר לזהות באגים, למדוד ביצועים, לבדוק פונקציות מסוימות ולבצע בדיקות שונות בצורה מהירה ויעילה.
4. **חסכון בעלויות:** אף על פי שהשקעת המשאבים בהתחלה לפיתוח בדיקות אוטומציה יכולה להיות גבוהה ניתן לזהות באופן מוקדם יותר בעיות ותקלות בתוכנה. זה מאפשר תיקון מהיר ויעיל עוד בשלבים מוקדמים של תהליך הפיתוח, כאשר העלות של תיקון השגיאות עוד נמוכה. בכך ניתן לחסוך בעלויות עתידיות שעשויות להתרחש עקב פעולות תיקון מאוחרות.
5. **חסכון בטעויות אנוש:** בעת ביצוע בדיקות ידניות, ייתכן שיתרחשו טעויות אנוש כתוצאה מטיפול שגוי או שכחת מקרי קצה. הבדיקות האוטומטיות מבצעות את הבדיקות בדיוק לפי ההוראות שהוגדרו, מכך נובעת הפחתת הסיכוי לשגיאות בעת הבדיקה.
6. **הפחתת סיכון:** על יד הפעלה תכופה של הבדיקות. דבר זה מאפשר לאתר באגים ובעיות בתוכנה בשלב מוקדם יותר ולתקן אותם לפני שהם מתרחשים ביישום הפועל.
7. **תמיכה בריבוי פלטפורמות ומכשירים:** בעת פיתוח תוכנה, ייתכן שיש צורך לבדוק את היישום על מגוון רחב של פלטפורמות ומכשירים שונים. בדיקות אוטומציה מאפשרות ליצור בדיקות חוזרות שניתן להריץ במקביל על מספר פלטפורמות ומכשירים, מה שמקל על הבדיקה והפעלת היישום במגוון תנאים.
8. **יכולת לבצע בדיקות מרובות:** בדיקות אוטומציה מאפשרות לבצע מגוון של בדיקות חוזרות בצורה מהירה ויעילה זה מאפשר לזהות בעיות שוב ושוב, לבדוק את הפיתוח החדש בכל ריצה ולוודא שאין השפעה של שינויים קודמים על התוכנה. בנוסף להתאים את הבדיקות לתסריטים שונים ולתנאים שונים.

ולמה להשתמש ב-Automation Framework?

כי אחרת היינו מיישמים הכל מאפס...

הנה כמה מהיתרונות המרכזיים:

- **ניהול וניתוח מרוכז:** ה-Automation Framework מספק מבנה ותשתית לניהול מרוכז של הבדיקות האוטומטיות. זה כולל את הכלים, התהליכים, מנגנוני הדיווח והמתודולוגיות הנדרשות לביצוע הבדיקות בצורה יעילה ומסודרת.
- **חסכון בזמן ומשאבים:** ה-automation framework מאפשר כתיבת קוד אוטומציה מבוסס תבניות וגם שימוש בכלים וספריות קיימות. זה מפחית את הזמן והמאמץ הדרוש לכתיבת קוד חדש עבור כל בדיקה ומאפשר שימוש חוזר והפחתת הדרישה למשאבים.
- **יכולת רחבה:** ה-automation framework מספק פונקציות ותכונות מובנות שניתן להשתמש בהן כדי לממש ולבצע בדיקות מורכבות ומגוונות. זה כולל דרייברים, אסטרטגיות איתור (locators), פעולות דפדפן (actions), אירועי משתמש (events) ועוד.
- **יעילות ודיוק:** ה-automation framework מבצע אוטומציה תחת תהליכים חוזרים ומנוהלים, מה שמקנה יעילות ודיוק גבוהים בבדיקות.

BDD – Behavior-Driven-development, מה?!?

מתמקד בעיקר בנקודת המבט העסקית ובדרישות התוכנה ופחות ביישום הטכני.
ובנוסף מעודד שיתוף פעולה בין מפתחים, QA ומשתתפים עסקיים, במאמץ לעזור לגשר על הפער העסקי/טכנולוגי.

מעולה, אז מה זה Gauge?

כלי חינמי open source לכתיבה והרצת בדיקות קבלה(acceptance tests) שמאפשר למהנדסי האוטומציה להאיץ את תהליך פיתוח התוכנה.

פותח על ידי חברת **thoughtworks**, הידועה בעיקרה כחברת יישומי תוכנה בינלאומית ומחלוצי ה-Selenium.

הרעיון היה לאפשר לאנשי QA ומפתחים ליצור בדיקות בצורה פשוטה ואינטואיטיבית, בלי להיות תלויים בידע עמוק בתכנות. הקונספט המוצלח מאחורי Gauge הוא "תיעוד חי וביצוע", כלומר מסמך שניתן לערוך ולעדכן כל הזמן.

יש לו מספר רב של פיצ'רים ייחודיים אל מול פריימוורקים אחרים, נראה למשל בטבלה את ההשוואה לפריימוורק Cucumber

לירן יושינסקי

בעל ניסיון של כ-20 שנה בעולם ההיי טק ובפרט באוטומציה

בארבע וחצי השנים האחרונות עובד בחברת הסייבר Tufin

בתור Automation Tech Lead בקבוצת ה-DevOps ומוביל Automation Guild בחברה.

חובב טכנולוגיות ובפרט מכור לכלי Open source

בזמני הפנוי עוסק בספורט ימי בעיקר חתירת קיאקים בארץ ובעולם.

אוטומציה מורכבת פחות עם פריימוורק הבדיקות Gauge | לירן יושינסקי

CUCUMBER	GAUGE
מבוסס על קבצי טקסט עם Given, When Then	תחביר פשוט, גמיש, קריא ועשיר המבוסס על Markdown
תומך בכמה סוגים בסיסיים של פרמטרים (int, float, word, string), טבלאות נתונים וסוגי פרמטרים מותאמים אישית שאפשר להצהיר באופן ידני.	תומך ב-data driven execution ו external data sources. מאפשר להחדיר טקסט שלם של קובץ או CSV (עם ניתוח אוטומטי לטבלה) ללא תוספת קוד. נפרט עליהן בהמשך.
אין תכונה מובנית כזו זמינה	הרצה בקלות של חבילות בדיקה (test suites) על מספר סביבות במקביל.
אין תכונה מובנית כזו זמינה	בעל יכולת לשלב קבוצות לוגיות של שלבים לשימוש חוזר לתוך יחידה אחת של scenarios הידועים בשם concepts
רק ה-Background steps זמינים ברמת ה-feature	שלבי ההקמה (Background steps) וגם שלבי tear down זמינים ברמת ה-specification/feature
דורש תוספים נוספים כדי להשיג תכונה זו.	ביצוע מקביל של scenarios הוא פשוט מאוד ללא תוסף נוסף
אין אובייקט מובנה לאחסון run time data.	תכונה מובנית לאחסון run time context data.
אין תכונה מובנית לקריאת נתונים מקובץ חיצוני.	תכונה מובנית לקריאת נתונים מקובץ חיצוני.
אין תכונה מובנית ללכידת צילומי מסך	תכונה מובנית ללכידת צילומי מסך.
Hooks מוגבלים: Before/After/BeforeStep/ AfterStep.	הרבה יותר Hooks כמו: BeforeSuite/AfterSuite/ BeforeSpec/AfterSpec/ BeforeScenario/AfterScenario/ BeforeStep/AfterStep נרחיב עליהם בהמשך
אין תכונה מובנית להרצה חוזרת של טסטים שנכשלו	תכונה מובנית להרצה חוזרת של טסטים שנכשלו
אין תכונה כזו זמינה	תכונה מובנת להמשך הריצה גם לאחר כישלון טסט.

עוד מקבץ פיצ'רים:

- **ארכיטקטורה מודולרית**

היא גישה בה המערכת מורכבת ממודולים נפרדים שמכילים פונקציות ותכונות ספציפיות. כשמדברים על ארכיטקטורה מודולרית לכיסוי כל קונספט ה-(Behavior Driven Development) BDD, מתכוונים לבניית מערכת בדיקות המכילה מודולים שונים שתומכים בתהליכי ה-BDD והפיתוח המבוסס על התנהגות (behavior) של התוכנה. המודולים בארכיטקטורה המודולרית ל-BDD יכולים להיות בנויים בצורה פלגינית, כלומר, ניתן להוסיף ולהשתמש בפלאגינים שונים במערכת הבדיקות. בדרך כלל, הפלאגינים המשתמשים ב-BDD תומכים בשפות תכנות ספציפיות כמו Gauge, שמאפשרת למפתחים כתיבת סקריפטים בצורה קרובה לשפה האנגלית.

בנוסף, עם ארכיטקטורה מודולרית ל-BDD, ניתן לשלב כלים וטכנולוגיות נוספות בצורה קלה וגמישה. לדוגמה, ניתן לשלב את Selenium ככלי אוטומציה לבדיקות ממשק משתמש וניווט בדפים, את Rest Assured לבדיקות API, את Appium לבדיקות יישומים ניידים ואת AutoIt לבדיקות על ממשקי משתמש בווינדוס.

- **תמיכה עקבית בין פלטפורמות**

הגישה המודולרית מאפשרת לתוכנית הבדיקות להיות ניידת בפלטפורמות שונות כמו לינוקס, מקינטוש ווינדוס. זאת מאפשרת לצוות הבדיקות להריץ את אותן הבדיקות במגוון סביבות עבודה ולוודא שהתוכנה עובדת כמו שצריך בכל אחת מהן. בנוסף, התמיכה בפלטפורמות שונות מאפשרת לצוות להתמודד עם האתגרים הספציפיים של כל פלטפורמה ולבדוק את התוכנה בסביבות מגוונות.

תמיכה בשפות תכנות מרובות

Gauge מאפשר לכתוב קוד בדיקה במגוון שפות תכנות כמו Java, Ruby, C#, Go, JavaScript ו-Python. זה מאפשר לצוות להשתמש בשפה המועדפת עליו ושימוש בכלים וספריות ידועות בשפה הספציפית. זה מבטיח גמישות ואפשרויות מרובות לצוות הבדיקות לכתוב ולתחזק את קוד הבדיקה.

- **יצירת חבילות בדיקה (test suites) ניתנות לתחזוקה**

ניתן לארוז את הבדיקות לחבילות שונות על פי נושאים, תכליתם או חלקי המערכת השונים שצריך לבדוק. היתרון המרכזי של יצירת חבילות בדיקה הוא היכולת לתחזק ולנהל אותן בצורה יעילה. במקום לנהל מאות או אלפי בדיקות בודדות, ניתן לארוז אותן לחבילות המותאמות לצרכים שונים. כל חבילה יכולה להיות מתוכנתת ונבדקת בנפרד, וכל שינוי או תיקון שיעשו בחבילה לא ישפיע על שאר החבילות. בנוסף, יכולת התחזוקה הנוחה מאפשרת עדכון והוספת בדיקות בקלות. כאשר ישנן שינויים במערכת או נדרש לבדוק תכונה חדשה, ניתן להוסיף בדיקות לחבילה הרלוונטית בלבד ולהריץ אותן כחלק מהתהליך האוטומטי. זה מבטיח תחזוקה קלה ונוחה של הבדיקות, מבלי להשפיע על הבדיקות האחרות בפרויקט.

- **Gauge משתלב באינטגרציה רציפה (Continuous Integration)**

יכול להיות משולב כחלק מפייפליין ב-Jenkins או כל כלי אחר לניהול אינטגרציה רציפה. דבר נפוץ מאוד להכניס את בדיקות האוטומציה כחלק מתהליך CI/CD, כך שבכל פעם שקוד חדש מתווסף או מתעדכן יוכל להיבדק באופן אוטומטי. בעזרת הפיתוח הרציף והאינטגרציה עם Jenkins, אפשר להפעיל את בדיקות האוטומציה באופן אוטומטי בתהליך הבניה והפרסום של הקוד. כאשר יש שינוי חדש בקוד המקור, Jenkins מפעיל את הבדיקות האוטומטיות באופן אוטומטי ומספק מידע על התוצאות לצוות הפיתוח. זה מאפשר לזהות במהירות בעיות אפשריות ולהבטיח שהשינויים לא פוגעים בתפקוד התוכנה. לפיכך בדיקות אוטומטיות כחלק מתהליך CI/CD מביאות יתרונות רבים, כמו למשל זיהוי ותיקון באופן מוקדם של בעיות.

- **Data stores**

זהו אחד הפיצ'רים המגניבים ביותר ב-Gauge שבהן ניתן להשתמש כדי לשמור את הנתונים בין ה-steps ב-Scenario או לשמור נתונים בתוך Scenario בזמן הריצה.

- **מנגנון דיווח מותאם ויעיל עם מספר סוגי דוחות**

Gauge מספק דוח ביצוע בדיקה live שהוא מאוד ייחודי ל-Gauge ושיעזור לנו לזהות את התקלות מבלי להמשיך לחכות עד להשלמת הביצוע. בנוסף לכך, Gauge מספק את דוח ביצוע הבדיקה בפורמטים HTML, XML ו-JSON.

- **Parallel Test Execution**

על ידי הגדרת משתנה סביבה כדי להגדיר את מספר ה-streams. דבר שיאפשר לנו multi-threading בהרצת בדיקות במקביל כך ניתן לסיים את הבדיקה במהירות כמובן בהנחה שאפשר להריץ את הבדיקות במקביל.

אוטומציה מורכבת פחות עם פריימוורק הבדיקות Gauge | לירן יושינסקי

ל- Gauge יש פתרון וקבצים משלו לניהול סביבה

אפשר לשים קבצי "properties" של Gauge במבנה הפרויקט המוגדר שלו ולאחר מכן לכוון לשם את הסביבה בזמן הפעלת הבדיקות. אפשרי לייצר מספר פרופילים שונים כמספר הסביבות שנצטרך.

- **Hooks**

כמו ל-Selenium גם Gauge מציע hooks רבים כך שנוכל פשוט להשתמש ב-annotation מתאים כאשר נרצה שקטע קוד מסוים יופעל. כמו @BeforeSpec, @AfterSpec, @BeforeScenario, @AfterScenario ועוד...

- **Rerun Failed Tests**

מאפשר לאלץ את Gauge להפעיל מחדש את המבחנים שנכשלו.

- **Continue on Failures**

בעוד שהתנהגות ברירת המחדל היא שבירת ביצוע בכישלון הראשון בstep, אפשר פשוט לסמן את השלבים כ-"ContinueOnFailure".

צילום מסך של חלק מהפלאגינים שאנו משתמשים בהם:

```
11:42:25  + gauge -v
11:42:25  Gauge version: 1.4.3
11:42:25  Commit Hash: f98dd40
11:42:25
11:42:25  Plugins
11:42:25  -------
11:42:25  html-report (4.1.4)
11:42:25  java (0.9.1)
11:42:25  screenshot (0.1.0)
```

מקבץ מושגים חשובים ב-Gauge

- **Specification**

או בקצרה spec
הוא בעצם business test case המתאר תכונה מסוימת של האפליקציה הדורשת בדיקה. כל spec יכול להכיל scenario אחד או יותר, כל spec יכול להיות מסומן עם Labels שמשתמש ב-tags

- **Scenarios**

כל scenario מייצג single workflow ב-spec מסוים. spec חייב להכיל לפחות scenario אחד. scenario מתחיל אחרי scenario heading או scenario name.

- **Steps**

הם רכיבי ההפעלה של spec שנכתבים באמצעות תחביר רשימת Markdown.
ב-spec, steps יכולים להתקיים בתוך scenario או מחוץ לו.
Steps קיימים גם בתוך Concept.
לכל implementation של step יש קוד שווה ערך לפי הפלאגין של השפה המשמשת בעת התקנת Gauge.

- **Concepts**

מספקים את היכולת לשלב קבוצות לוגיות הניתנות לשימוש חוזר של steps ליחידה אחת.

כאשר ה-spec מופעל, כל ה-concepts וה-steps שלהם מבוצעים בסדר המוגדר.

- **Tags**

תגיות משמשות לשיוך Labels ל-spec או ל-scenarios. עוזרים בחיפוש או סינון specs או scenarios. תג שהוחל על spec חל אוטומטית גם על scenario.

- **Parameters**

steps מוגדרים לקחת ערכים כפרמטרים כך שניתן לעשות בהם שימוש חוזר עם ערכי פרמטר שונים. בקובץ ה-implementation יהיה לנו אותו מספר פרמטרים כפי שהוזכר ב-step.

קיימים מספר סוגי פרמטרים:

1. Simple parameters - הם ערכי מחרוזת, המשמשים ב-step בתוך מרכאות כפולות.
2. Table parameters - משמשים כאשר מבוצע step עבור מספר ערכים בטבלה.
3. Dynamic parameters - משמשים כ-placeholders במקום ערכים בפועל. אלה משמשים כאשר מתייחסים לערך עמודת טבלה של טבלת נתונים.
4. Special parameters - סוגי הפרמטרים המיוחדים הם כדלקמן: TXT,*CSV,*XML,*JSON*
   העברת תוכן הקובץ כפרמטר מחרוזת.

דוגמא לקובץ Spec:

דוגמא לקובץ Concept

חשוב:

הקישור בין שני הקבצים נעשה על ידי ה-step ב-scenario בקובץ ה-spec
עם ה-scenario heading בקובץ ה-concept
למשל:

```
Check access for new Super admin to Monitoring group and verify they can
                                                    access to Grafana dashboard
```

החיבור של step מקובץ ה-concept לקובץ ה-implement יעשה על ידי שימוש ב-hook

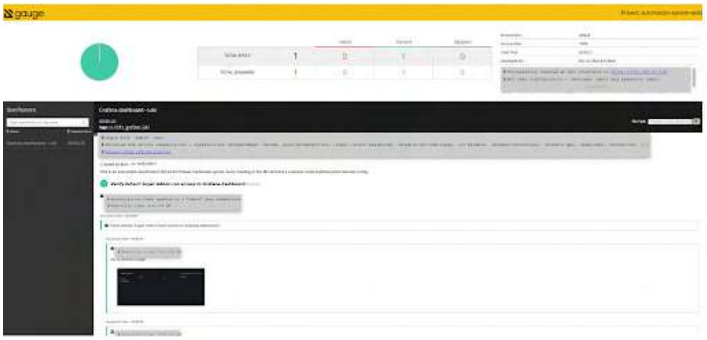
אוטומציה מורכבת פחות עם פריימוורק הבדיקות Gauge | לירן יושינסקי

@step כפי שמופיע בדוגמא הבאה:

```
@Step("Verify access to Grafana dashboard")
fun verifyAccessToGrafana() {
    when (checkAccessToGrafana()){
        Status.SUCCESS -> return
        Status.UNAUTHORIZED -> {
            Gauge.writeMessage("Received unauthorized instead")
            fail(Status.UNAUTHORIZED.message)
        }
        Status.PAGE_NOT_LOADED -> {
            Gauge.writeMessage("Didn't found any element of Grafana - authorized or not, probably the page didn't reload yet")
            fail(Status.PAGE_NOT_LOADED.message)
        }
    }
}
```

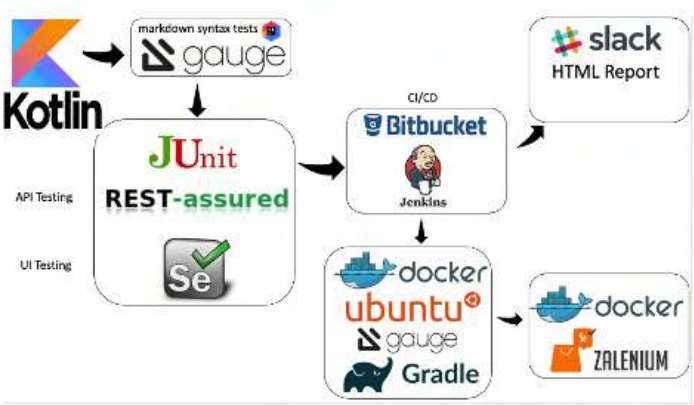
דו"ח HTML של Gauge ל-scenario:

שנמצא בדוגמא של קובץ ה-spec



Verify default Super Admin can access to Grafana dashboard

תיאור התממשקות הטכנולוגיות שאנו משתמשים כדי לייצר את מערך האוטומציה שלנו:



לסיכום

Gauge הוא כלי מתקדם לבדיקות אוטומטיות שמספק פתרונות לבעיות תחזוקת הבדיקות של ממשק ה-Web שלכם. בעזרת Gauge, אתם יכולים לבצע אלפי בדיקות על ממשק ה-UI וה-API, באופן אוטומטי ובקלות.

הוא מאפשר לכם להריץ את הבדיקות על multiple browser, מה שמבטיח את תקינות הממשק בסביבות שונות.

למעשה, Gauge מאפשר לבדוק את הפלטפורמה שלכם באמצעות system tests.

בעזרת הכלים שלו, ניתן לבדוק את תהליכי המערכת ולוודא שהם פועלים כצפוי.

הפריימוורק של Gauge מיועד להיות פשוט לשימוש, גם עבור משתמשים שאינם טכניים. בזכות זה, גם חברים בצוות שאינם מפתחים יכולים להצטרף לתהליך הבדיקה ולתרום בצורה פעילה.

בנוסף, Gauge מתמודד במקצועיות עם בעיות תחזוקת הבדיקות שלכם.

הוא מספק פתרונות לבעיות המתקיימות במערכת הבדיקות, בזכות תורמים מרחבי העולם שמשפרים את הכלים ומקנים להם יכולות חדשות.

כמו כן, ישנה קהילה גדולה שמתפתחת ומגדילה את הכלים ומספקת תמיכה והדרכה למשתמשים. לקבלת מידע נוסף והוראות נוספות, ניתן לבקר בלינק הבא:

https://docs.gauge.org

סקירת כלים Xray Exploratory

ניצן גולדנברג

בגיליון זה אני רוצה להביא לכם כלי שהוא תוספת ישירה של ג'ירה בשם Xray Exploratory. את הכלי יצא לי להכיר במהלך הטמעת XRAY בארגון שלי וברגע שהכרתי אותו, חיי השתנו בכל מה שקשור לבדיקות חקרניות.

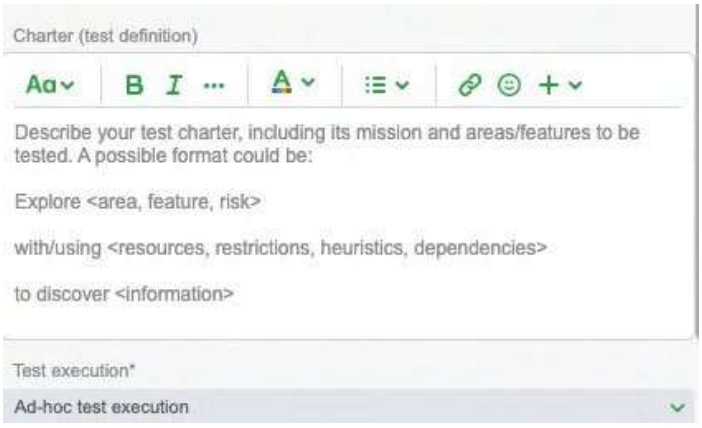
- הקלטת וידאו וצילומי מסך במהלך הרצת הבדיקות
- התממשקות ישירה לג'ירה
- יכולת עבודה על סביבת חלונות, מקינטוש או לינוקס
- יצוא דוחות בפורמט PDF
- יכולת בדיקות על פלטפורמת דפדפן, שולחן עבודה
- שימוש במסמך צ'רטר

קצת רקע על הטכניקה

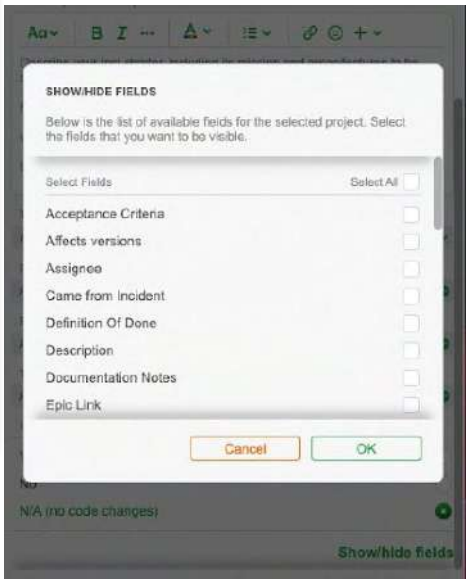
בדיקות חקרניות הינה טכניקה אשר משלבת את "עיצוב "הבדיקות ו"ביצוע "הבדיקות. הגישה מאפשרת לבודק להריץ את הבדיקות בצורה חופשית ובו זמנית לתעד את הצעדים של הבדיקה. הכנת הבדיקות כוללת שימוש במסמך Test Charter אשר כולל מיקוד לבדיקה, אסטרטגיה ותכנון זמן לבדיקה.

Xray Exploratory מאפשר לנו לאפיין כמה זמן נרצה להשקיע בבדיקה וכמה זמן בפועל אנו מריצים את הבדיקה. אנו יכולים לרשום את תוכנית הבדיקה

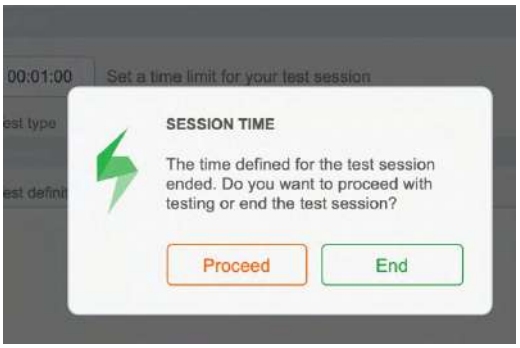
יש לנו מקום למלא את המסמך תכנון הבדיקה (Test Charter)



ברגע שהכלי מקונפג עם חיבור ישיר לפרויקט ג'ירה אז ניתן לקנפג אילו שדות מתוך הפרויקט אנו נציג בהרצה כגון: Definition of Done, קישור לאפיק, תוצאה צפויה, שיוך לגרסה מסוימת או כל שדה שמאופיין בפרויקט שלנו.



כאשר מסתיים הזמן שהגדרנו לבדיקה, אנו מקבלים הודעה מסודרת אם אנו מעוניינים לסיים את הבדיקה או להמשיך



במהלך הרצת הבדיקות נפתחים לנו אפשרויות נוספות: צילום מסך, הקלטת וידאו של המסך, הקלטת אודיו, רישום הערות במהלך ההרצה, עצירה זמנית של ההרצאה וסיום ההרצה.

ניצן גולדנברג

מזה 7 שנים בתחום בדיקות התוכנה, תפקיד נוכחי מהנדס בדיקות בכיר בחברת SeatGeek, מנהל קבוצת ה-AB של ארגון ITCB® המוביל הראשי של קבוצת המיטאפ TestIL, מרצה בקורסים לבודקי תוכנה

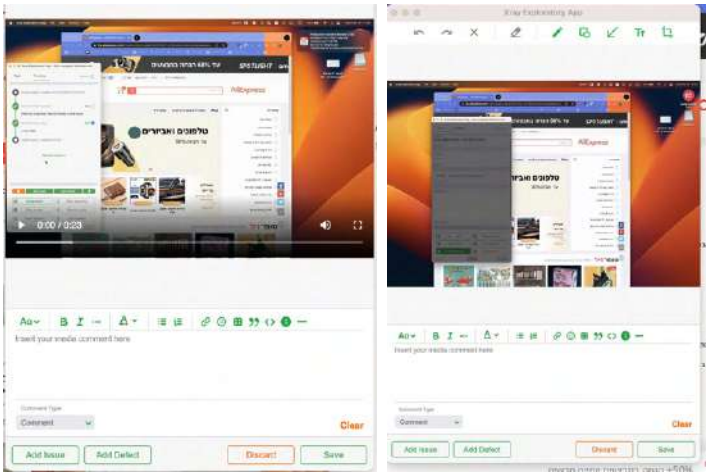
סקירת כלים Xray Exploratory

ניצן גולדנברג

צילום מסך יחיד או סרטון

בעת צילום המסך בין אם זה צילום יחיד או סרטון, ניתן להוסיף הערות, לייצר ישירות מתוך הצילום אייטם בג'ירה (כל סוג אייטם מאופיין בפרויקט שבחרנו) או תקלה.

ניתן לכתוב או לסמן על גבי התמונה כל מה שאנו רוצים

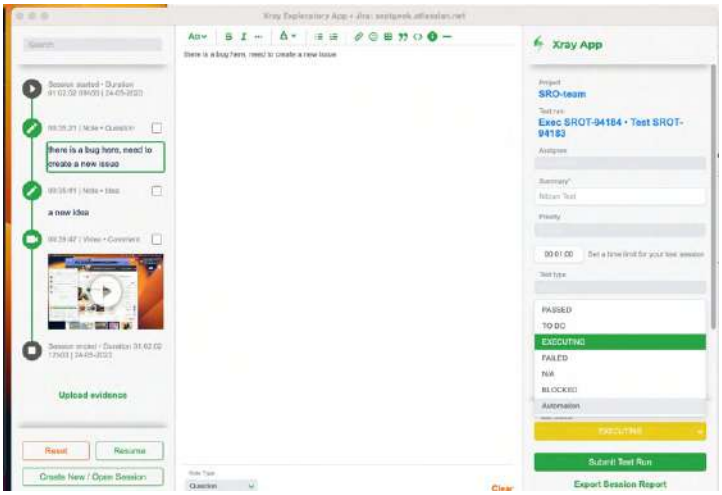


הוספת הערה במהלך הרצה אפשרית בכל שלב של העבודה, בין אם זו הערה כללית, שאלה, הצעה או כל דבר אחר שעולה לנו באותו הרגע ובכל רגע נתון ניתן לראות את כל ההערות בלשונית "ציר הזמן" ולראות את ההודעות או לייצר מהן אייטמים בג'ירה.

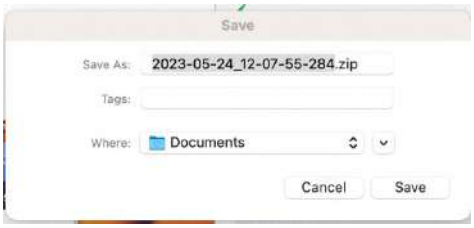
כאשר מסיימים את ההרצה ולוחצים על סיום הסשן,

מקבלים חלון המאפשר לבחור אילו הערות או צילומי מסך שיצרנו במהלך הריצה אנו מעוניינים לצרף לאייטם.

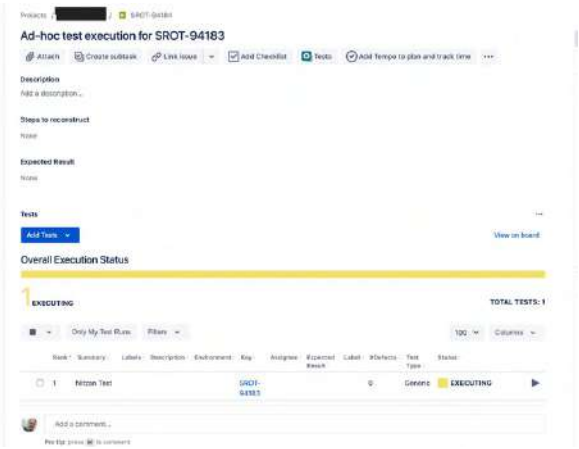
ניתן להגדיר את הסטטוס של מנת ההרצה לפי הסטטוסים המאופיינים לנו בפרויקט



ניתן לייצא את הריצה עם כל הקבצים שבחרנו (הערות, סרטונים, צילומי מסך וכו') לקובץ מקובץ מסוג zip אשר מכיל קובץ PDF וקבצי TXT ו-MP4



לאחר שמייצרים את מנת ההרצה, היא מיוצרת ישירות בג'ירה תחת Testing Board



יתרונות:

- תוסף ישיר של חברת Atlassian אשר מתממשק עם ג'ירה
- ניתן להשתמש בכל השדות המאופיינים לנו בפרויקט
- ניתן להשתמש בכלי גם ללא חיבור ישיר לג'ירה
- שימוש בכלי ככלי ייעודי לצילומי מסך וידאו חינמי
- מתאים לפלטפורמות חלונות, מקינטוש ולינוקס
- יש מדריכי שימוש באתר שנותנים מענה מספק על אופן השימוש של הכלי
- הממשק גרפי מזכיר מאוד את מערכת ה-Xray למרות שמדובר על תוכנת שולחן עבודה

חסרונות:

- כאשר מייצרים את מנת ההרצה בסוף הריצה, המסך של ג'ירה לא נפתח בצורה אוטומטית, דבר אשר מקשה על המשך העבודה שכן אנו צריכים לחפש את מנת ההרצה במערכת שלנו.
- התמיכה הינה דרך אתר Xray, מה שאומר שצריך לפתוח טיקט רשמי
- ביצירת מנות ההרצה, למרות שסימנתי את הקבצים שאני רוצה לצרף, לא ראיתי את הקבצים במקרי בדיקה שנוצרו בג'ירה (רק ביצירת מקרי בדיקה בודדים)
- אין קהילת משתמשים גדולה

מענה לצרכים של החברה 9/10
נותן למשתמש חווית שימוש 7/10
תמיכה וקהילה 4/10
סה"כ 6.6

קריטריונים של כניסה ויציאה
Entry & Exit criteria

במאמר זה נסביר מה הוא ההבדל בין קריטריון כניסה ליציאה.
כיצד מגדירים קריטריונים אלו ואיפה אנו פוגשים אותם במהלך ההתהוות של פרויקט הבדיקות. בין היתר נחשף לדוגמאות שונות של קריטריוני כניסה ויציאה מפרויקטים אמיתיים.

קריטריוני כניסה ויציאה הם בעצם פרמטרים חשובים לצוות הבדיקות בכדי שנבין מתי פעילות הבדיקות אמורה להתחיל ומתי להסתיים. הקריטריונים הם חלק בלתי נפרד מתהליך תכנון ואומדן הבדיקות ומופיעים כחלק ממסמך תכנון הבדיקות הרשמי (STP).

ההבדלים בין הקריטריונים

קריטריון כניסה:

קריטריון כניסה מגדיר את תנאי הקדם להתחלה של פעילות בדיקות נתונה.

במילים אחרות מדובר על הגדרה של נקודת ההתחלה, כך גם נקרא הקריטריון כשאנו עובדים בשיטה האג'ילית.

הנחת המוצא היא שלא נתחיל בבדיקות לתוכנה שאינה יציבה או שלא הגיעה אלינו ברמה מספקת בכדי להתחיל להריץ בדיקות שהכנו מראש.

פעילות הבדיקות היא בד"כ תהליך עתיר משאבים שמצריך תיאום ותכנון ואין הגיון להתחיל בבדיקות כאשר המערכת לא עוברת למשל בדיקות שפיות ראשוניות, כלומר איננה מבצעת את הפונקציות הבסיסיות אותן היא נדרשת.

מטרת הקריטריונים היא להבטיח שהמערכת אכן מוכנה לבדיקות ובכך למנוע מצבים של אי יציבות ועבודה כפולה.

קריטריונים אפשריים שיכולים לשמש אותנו להחלטה אם להתחיל או לא:

- זמינות של סביבת הבדיקות
- זמינות של הכלים איתם נבצע בדיקות

(למשל אם הושגו הרישיונות הנדרשים)

- זמינות של נתוני הבדיקות
- זמינות של פריטי מידע שעמדו בקריטריון היציאה
- זמינות של דרישות אשר ניתנות לבדיקה

קריטריוני כניסה יכולים גם להיות מחוץ לעולמות התוכנה כמו למשל:

- היערכות של הצוות מבחינת זמינות כוח אדם
- תנאי כניסה המוגדרים על ידי רגולציה
- תנאי כניסה המושפעים מתוקף חוזה מול הלקוח
- תנאי כניסה המותנים בפירעון תשלום של הלקוח

כל אלו יכולים להוות סוגים שונים של קריטריוני כניסה שאם לא יתקיימו, פעילות הבדיקה תימשך כנראה זמן רב יותר מהצפוי, תעלה יותר כסף ותסכן את תהליך הבדיקות בהיבטים של עלויות, זמן וחריגויות אחרות שאינן רצויות.

דוגמא ממשית לקריטריון כניסה:

שלב	קריטריון
בדיקות שפיות	בוצעו כל הבדיקות שתוכננו
בדיקות שפיות	כל הבדיקות שבוצעו, עברו בהצלחה

בדוגמא הנ"ל מבקש מתכנן הבדיקות להצהיר שמבחינתו כניסה לתהליך הבדיקות מותנית בכך שכל הבדיקות שתוכננו מבעוד מועד כבדיקות שפיות ירוצו ומבחינתו חשוב שכל הבדיקות שהורצו עברו בהצלחה.

ראוי למשל במקרה כזה, לשאול את המחבר – בכמה בדיקות מדובר שכן אם מדובר באתר תדמית למשל אפשר להסתפק במס' מועט של בדיקות שפיות כיוון שהמערכת אינה קריטית.

זאת בניגוד למערכת מעולמות המדיקל, הצבא או התעופה ששם ראוי שהקריטריונים יהיו מהודקים יותר ולכן נצפה לכמות גבוהה מאוד של בדיקות שפיות וכך נקבע רף גבוה יותר לתחילת העבודה.

> **"פעילות הבדיקות היא בד"כ תהליך עתיר משאבים ומצריך תיאום ותכנון ואין הגיון להתחיל בבדיקות כאשר המערכת לא עוברת למשל בדיקות שפיות ראשוניות, כלומר איננה מבצעת את הפונקציות הבסיסיות אותן היא נדרשת"**

קריטריון יציאה:

קריטריון זה מגדיר אילו תנאים יש להשיג בכדי להצהיר על השלמה של רמת בדיקה או סבב בדיקות.

אנו נפגוש את הקריטריון במסמך תכנון הבדיקות (STP) וגם במסמך סיכום הבדיקות (STR) שבו נוכל לגלות אם עמדנו או לא עמדנו בקריטריון המבוקש.

עמידה או אי עמידה בקריטריון היציאה תוכל לאפשר לנו להמליץ אם לעבור לשלב הבא או לא. המלצות ומסקנות אלו יועברו לדרג בעלי העניין שממתינים לקבל את התוצאות. (מנהל הפרויקט, הלקוח, אנשי הפיתוח וכו').

קובי יונסי

בעל תואר ראשון בלוגיסטיקה וכלכלה מאונ' בר אילן.
מייסד את המרכז המוביל לבדיקות תוכנה בישראל.
תכניו נלמדים באקדמיה בגופים כמו: הטכניון, מכללה להנדסה עזריאלי י-ם, אקדמית ר"ג, אקדמית פרס ברחובות, מכללת תל חי, מכללת איקום ועוד.
בתעשייה הוא מרצה הבית של חברות הבדיקה הגדולות במשק כמו: SQLINK, Ness, Qualitest.
מלמד אנשים כיצד לחשוב יצירתי ולהיכנס להיי טק דרך מקצוע בדיקות התוכנה אותו הוא חי ונושם 24/7

כללים לקביעת קריטריון יציאה:

- הבטחה שהעיקרון ישים וברור לכולם.
- יש להגדיר את הקריטריונים לפני תחילת הפרויקט.
- לכל הפריטים יש קריטריוני יציאה דומים.
- אין "הנחות" או הקלות על אי עמידה בקריטריונים לאחר שהוגדרו.

קריטריונים אפשריים שיכולים לשמש אותנו להחלטה מתי ניתן להפסיק:

- כל הבדיקות המתוכננות בוצעו.
- הושגה רמת הכיסוי המבוקשת (כיסוי של קוד, סיפורי משתמש וכו')
- מספר התקלות הלא פתורות נמצא בתחום ידוע מראש.
- רמות מאפייני האיכות מספקים (היבטים כמו: אמינות, ביצועים, שימושיות, אבטחה ונושאים אחרים)
- מספר התקלות שנשארו בתוכנה נמוך.
- רמת חומרת התקלות שנותרה מאחור היא נמוכה ולא מהווה סיכון.

חשוב לציין שגם בלי שקריטריון היציאה מושג, במקרים מסוימים הבדיקות מסתיימות לפני שהגיעו לסיום מאחר והסתיים התקציב, אזל הזמן שהוקצה לסבב הבדיקות או שקיים לחץ להוציא את המוצר לשוק מצידם של אנשי המוצר. הפסקת הבדיקות בתנאים שכאלו יכולה להילקח בחשבון בתנאי שהסיכון בשחרור המוצר ללא בדיקות נוספות מוצג לבעלי העניין וקיימת הסכמה מצידם להמשיך.

"עמידה או אי עמידה בקריטריון היציאה תוכל לאפשר לנו להמליץ אם לעבור לשלב הבא או לא. המלצות ומסקנות אלו יועברו לדרג בעלי העניין שממתינים לקבל את התוצאות"

בסופו של יום תמיד יתקיים המתח בין אנשי המוצר שמבקשים להרוויח זמן כאשר מה שמניע אותם הוא: Time to the market מול אנשי הבדיקות שמבקשים יותר זמן בכדי להבטיח יותר איכות.

דוגמא ממשית לקריטריון יציאה:

להלן הגדרת הקריטריונים לאישור העברת המערכת לייצור, ברמת אחוז התקלות

מסך הבדיקות שבוצעו:

קריטריונים	קריטי	חמור	בינוני	מינורי
תקלות פתוחות	0%	2%	12%>	20%>

להלן הגדרת הקריטריונים לאישור העברת המערכת לייצור, ברמת הבדיקות:

קריטריונים	%
% בדיקות שבוצעו מתוך הבדיקות שתוכננו	85%
% בדיקות שעברו מתוך הבדיקות שבוצעו	90%

מחוץ להגדרה של קריטריוני יציאה עבור הפרויקט, ניתן להגדיר גם קריטריוני יציאה לכל פעילות שמתבצעת כחלק מתהליך הבדיקות, להלן סקירה של כמה דוגמאות:

קריטריוני יציאה אפשריים עבור משימת פיתוח שמטרתה לכתוב קוד עבור התכונה או סיפור משתמש שיש לפתח יכולים לכלול:

- יש לכתוב קוד שמכסה את כל דרישות הלקוח
- הקוד חייב לעמוד בסטנדרטים מקצועיים (הערות מצורפות למשל)
- כל יחידת קוד עברה בדיקות קופסה לבנה (הצהרה או החלטה למשל)
- כל קוד שנכתב נבדק על יד חבר צוות נוסף בסקירת עמיתים (Peer Review)

קריטריוני יציאה אפשריים עבור פגישות עבודה או סקירות:

- הפגישה הסתיימה לאחר שלא נותרו נושאים פתוחים מאחור.
- הפגישה מתועדת בפרוטוקול מסודר על ידי אחד מהמשתתפים (סוקרים).
- ההחלטות שהתקבלו בפגישה עובדו מחדש (Rework) והתקיים מעקב (Follow Up) שנועד לוודא שהשינויים מומשו.

קריטריוני יציאה אפשריים עבור סבב בדיקות או ספרינט:

- כל סיפורי המשתמש בבקלוג נכתבו והושלמו.
- לא קיימות תקלות קריטיות לאחר הרצת הבדיקות.
- בדיקות רגרסיה של פיתוחים מסבבים קודמים בוצעו ועברו בהצלחה.
- אוטומציה של רגרסיה מפיתוחים קודמים הורצה ללא תקלות מיוחדות.

קריטריוני יציאה אפשריים עבור סיפורי משתמש:

- כל הבדיקות נכתבו ותועדו במערכת ניהול הבדיקות.
- כל הבדיקות עברו סקירה על ידי חברי הצוות שמסרו משוב לגביהם.
- הבדיקות הורצו ותקלות שנפתחו מקושרים לכל סיפור משתמש בנפרד.

קריטריוני יציאה אפשריים עבור בניית תשתית אוטומציה:

- אובחנו ונבחרו מקרי בדיקה שיהפכו להיות אוטומטיים.
- קוד האוטומציה נכתב עבור כל בדיקה והושלם.
- סקריפטים שנוצרו מתווספים לחבילת הבדיקות המלאה עם תגים רלוונטיים.
- הבדיקות הורצו לפחות פעם אחת במהלך הפרויקט.

חשוב לציין שאין רשימה אידיאלית של קריטריוני יציאה שמתאימה לכל צוות, כלומר כל אחד שאחראי על הבדיקות יכול להחליט על

האנציקלופדיה לבדיקות | קובי יונסי

קריטריוני יציאה בהתבסס על הצוות והקשר הבדיקות, ניתן כמובן להקל או להחמיר עימם בהתאם למהות המוצר ולפי הסכמות עם הלקוח ומנהל המוצר.

עובדה זאת עולה בקנה אחד עם אחד מעקרונות הבדיקה שמדבר על כך שבדיקות הן תמיד תלויות הקשר. כלומר קריטריוני יציאה של אתר תדמית לא יהיו זהים לאפליקציית ניווט או תוכנת ניהול מערכת מדיקל וכיוצא בזאת.

נקודה נוספת שמבליטה את חשיבותו של קריטריון היציאה היא שהוא משפיע גם על בקרת הבדיקות שכוללת פעולה של תיקון כתוצאה ממידע ומדדים שנאספים ומדווחים תוך כדי הפרויקט והם עלולות להשפיע על כלל פעילויות הבדיקה ואף על פעילויות הפיתוח.

מדדים שנוגעים בהתקדמות העבודה מול לוחות הזמנים כמו:

- אחוז הבדיקות שהורצו מול הבדיקות שנכתבו
- אחוז העבודה שבוצעה ביחס להכנת סביבת הבדיקות
- מקרי הבדיקה שעברו בהצלחה מול אלו שנכשלו
- עלות הבדיקות ביחס להשוואה של עלות מול תועלת עלולים שנמצאו

כל אלו הם פרמטרים ואינדיקציות שנועדו לאפשר לנו לזהות איפה אנו עומדים ביחס לקריטריוני היציאה שנקבעו מראש וכך להבין אם אנחנו עומדים ביעדי הבדיקות או שלא.

לסיכום

קריטריוני כניסה ויציאה הם אמות מידה חשובות בניהול פרויקט איכותי.

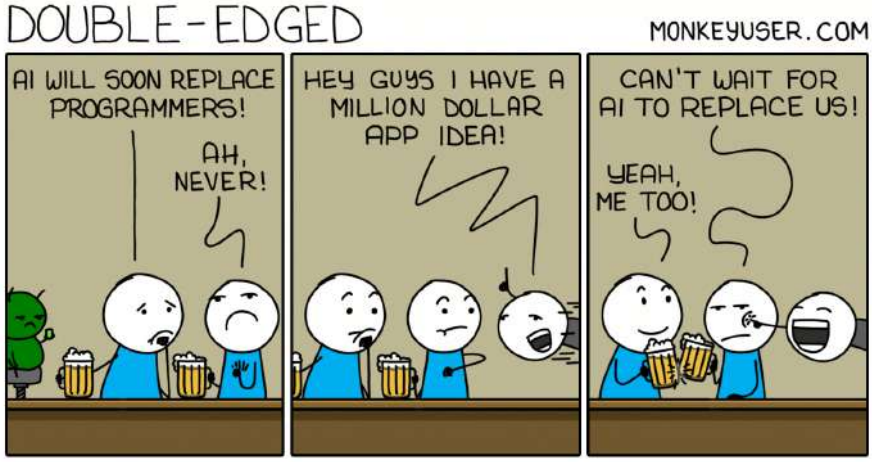
ללא הבנה מה מבחינתנו ייחשב לתחילת ההנעה של הפרויקט ומה הם התנאים שיוכלו להוכיח לנו שאנו עומדים בהצלחה ויכולים להכריז על המשך לשלב הבא לא נוכל לצאת לדרך.

הגדרת הקו האדום בעיקר סביב הנושא של קריטריון היציאה הוא הכרחי על מנת להבטיח זרימה חלקה של עבודה באיכות גבוהה.

בדרך לשם נזכור שכללי הברזל יהיו:

- הגדרה מראש של הקריטריונים.
- מעקב ובקרה צמודים לאורך הפרויקט אם אנו עומדים בקריטריונים שנקבעו מראש.
- אימוץ פעולות, שינויים והתאמות לאורך התהליך בכדי לעמוד בקריטריונים.
- חזרה אל אותם קריטריונים בשלב סיכום הפעילות ופעולות סיום הפרויקט כמו דוח סיכום הבדיקות (STR).

יש נושאים ומושגים נוספים שהייתם רוצים לקרוא עליהם? תשלחו מייל ואשמח לכתוב kobi757@gmail.com



איזה סוג של אנשי QA אנחנו | ניסים אריאל

יש הרבה גישות בתעשייה ביחס לתפקידו ומקומו של ה-QA

אין הדבר סוד, יש חברות שאין להן QA, ואין הדבר סתמי או מקרי אלא מבטא גישה, כמו שאמר לי חבר שהוא בעלים של חברה: "אני לא צריך QA בכלל". הוא לא היחיד, יש לא מעט חברות שסבורות כך, בין אם הן חושבות שאצלן אין בעיות, בין אם הן סבורות שהן שולטות במצב בצורה הדוקה כי המפתחים שלהם בודקים את עצמם כמו שצריך, בין אם הן שבויות באשליה שאצלן אין בעיות, בכל אופן זו גישה קיצונית, שרואה בבדיקות דבר מיותר שאינו מצדיק הוצאה כלכלית, ואינו חלק חיוני מתהליך הפיתוח, אלא משהו נספח שאפשר להסתדר בלעדיו Add-on. תופעה זו נפוצה יותר בחברות קטנות בתחילת דרכן.

יש עוד גישה קיצונית נוספת מן העבר השני. חברות שיש להן המון בודקים עם איזה חוסר קשר ופרופורציה לגודל של המוצר. גישה זו היא פחות גרועה לעניות דעתי, אך גם היא מבטאת בעייתיות משמעותית. גישה זו מגלמת בתוכה תפיסה, שהבודק הוא אדם מאוד מוגבל ביכולותיו, שמסוגל לכל היותר ללחוץ על כפתורים ולספק לנו למעשה איזה כלי סקריפטינג אנושי. אז צריך הרבה כאלה כלים - כדי שיהיה כיסוי יותר רחב. תופעה זו נפוצה יותר בחברות גדולות עם מוצרים גדולים.

שני המקרים המתוארים לעיל בעיניי הם סוג של אנומליה שנובעת מאותו ליקוי - חוסר הבנה יסודי ובסיסי של המושג Quality.

במקרה הראשון לא צריך Quality בכלל כאיבר באורגניות של הפיתוח, מפני שאפשר למצוא פתרון או תחליף. במקרה השני Quality הוא מושג **טכני**, כמעט ריק ממשמעות. תפיסה זו של המקרה השני של ריבוי בודקים טכני הייתה נפוצה יותר בשני העשורים הקודמים, בהם בודקים זכו לכינויים לא סמפטיים, או מזכיר לי מכר שאמר לי כמה ימים לפני שהתחלתי את התפקיד הראשון: "תנסה למצוא טעם בעולמות היובש האלה". העולם הולך ומתעורר מהגישה הזו, בעיקר על ידי למידה דרך הכיס בצורה פחות סימפטית.

האמירה שלי אינה קלה כלל : אחיי ואחיותיי הבודקים, אם אתם נמצאים במקום בו אתם לא נתפסים כמשאב חיוני לחברה, לתהליך ולמוצר – תעשו מאמץ להגיע למקום שכן תופס אתכם כחיוניים ואיכותיים בתהליך. והכל כמובן במסגרת היכולת והמקרה הפרטי של כל אחד ואחת.

> "אם אתם נמצאים במקום בו אתם לא נתפסים כמשאב חיוני לחברה, לתהליך ולמוצר – תעשו מאמץ להגיע למקום שכן תופס אתכם כחיוניים ואיכותיים בתהליך"

ישנם סוגים רבים של בני אדם, כשם שפרצופיהם שונים...

כל בר דעת מודע היטב לסגנונות השונים של בני אדם. בין אם זה באופן חשיבותם, בין אם זה בהתנהלותם וכמובן באישיותו המיוחדת של כל אחד מאתנו. עובדת הבסיס הזו שמרכיבה את המין האנושי על שלל גווניו לא פוסחת על אנשי QA - ומתוך כך ישנו מנעד רחב ביותר של סגנונות עבודה בתחום. יש אנשים שמאוד ממוקדים בפונקציונאליות של המוצר, מאוד מחוברים לכל פיצ'ר ברמה של שדה וכפתור. לעומתם, יש אנשים שלא נחה דעתם עד שהם מבינים מה שורש התקלה ואיך היא הולכת להיפתר ברמת הארכיטקטורה. לעניות דעתי וייתכן שיש רבים שיחלקו עלי בדבר, רובנו נמצאים איפשהו באמצע וברבות השניים נעים מכיוון הפונקציונאליות על עבר הבנה עמוקה יותר של המערכת ומעורבות גדולה יותר באופן הפיתוח שייתקן בעיות שנתגלו. וישנם כמובן "צלפים", שאמנם מעט חסרי סבלנות יחסית, אבל יש להם עין חודרת שקולטת די מהר היכן יש לחפש באג, לעומתם יש כאלה בעלי סבלנות אין סופית שעושים חריש רחב לאורך ולרוחב. כל הטיפוסים שהזכרתי כאן חשובים אי אפשר בלעדיהם. ויש עוד רבים וטובים שתקצר היד למנות את כולם.

זה לא קשור לרצינות

הטיפוסים הבסיסיים המוזכרים כאן אינם עומדים בסולם של מדרג רצינות, כולם יכולים והינם אנשים שעובדים בחריצות ובהשקעה של זמן ומשאבים. אדרבה, יכולה להיווצר סיטואציה שבה מגיע בודק עם הרבה שנות ניסיון ועם הכרות רבה של הרבה מערכות - ודווקא הוא לא יתאים לצרכים של החברה, כי בנקודה הספציפית של אותה חברה יש לה צורך אחר, וכן כמובן להיפך, בודק טוב שיודע להבין מערכת אבל מבטו מאוד ממוקד רק במה שמול פניו - ואת זה הוא עושה מעולה, אבל לא מעבר, עלול להיות לא מתאים לאותה חברה, וכן כמובן להיפך. אם כן הכל תלוי נסיבות, פוזיציות וצרכים של מקומות שונים ואינו מסמן מישהו כטוב יותר או פחות.

קטנות וגדולות בבדיקות

אפשר וחייבים ללמוד כלים חדשים, אופני בדיקות, מתודולוגיות בדיקה חדשות, פלטפורמות, כלי אוטומציה והמון המון דברים שמעשירים מאוד את תיק כלי העבודה של הבודק. אי אפשר להפריז בחשיבות העניין. דרכי פעולה רבות וטובות יש והן הכרח.

אך הנני מעוניין לגעת בנקודה קצת אחרת, עמוקה יותר לעניות דעתי. אשתמש ברשותכם במשל כדי לסבר את האוזן. כשילד נולד, הוא פתאום מוצא עצמו מול כל המציאות, מהי כבר יכולתו להכיל ממנה? מעט מאוד, בעיקר את אימו, את צרכיו הבסיסיים ביותר כמזונו, שעות השינה שלו וכו'. אבל הוא הולך ומתפתח, זאת אומרת, שהיכולת שלו להכיל ולהיפגש עם יותר מהמציאות גדלה יחד איתו – הוא מתחיל להכיר בעובדה שהוא חלק ממשפחה ומפתח איתה מערכת יחסים, הוא נהיה מודע יותר לגופו, הוא מתחיל לנסות לנוע, למצב את עצמו לבד בתוך המציאות, מעגליו הולכים ומתרחבים. כמובן שזהו תהליך ארוך עד שהאדם עומד בקומתו השלמה כאיש בוגר. אם תשאלו אותי מהי בגרות אמיתית – אומר, שלדעתי היא מתבטאת בעיקר בזה שהאדם נהיה שותף בבניית המציאות, משפיע. זה אדם בוגר. הוא לא רק עסוק באופן שהוא מסתדר במציאות ברמה כזו או אחרת, אלא הוא משנה עמדה להיות פועל על המציאות.

כך בדיוק במקצוע שלנו. החוש, ההבנה, הסגנון וכיוון החשיבה של הבודק הולך ומתפתח. הוא מתחיל ממקום מאוד גולמי – בדיקת פונקציונאליות, בודקים שדברים מתממשים כראוי, מתקדם להבנה יותר מעמיקה של המערכת ויודע לגעת בנקודות יותר רגישות – כתוצאה מראיה יותר היקפית וכוללת, וכמובן שריבוי של כלי בדיקות רק מעשיר ומוסיף בזה.

אבל יש שלב נוסף בתהליך ההתפתחות של בודק.
הוא לא רק בודק...הוא לא רק עוקב אחרי המוצר בצורה מתקדמת ככל הניתן, אלא הוא פועל עליו – הוא רואה את כל התהליך שהוא נמצא בתוכו ויש לו אמירה משמעותית איך לשפר את כל המהלך של החברה שהוא נמצא בה.

ניסים אריאל

ראש צוות בדיקות בחברת X-trodes, מספר שנים בתחום הבדיקות, נשוי+6 וחובב היסטוריה ופילוסופיה

איזה סוג של אנשי QA אנחנו | ניסים אריאל

"אפשר וחייבים ללמוד כלים חדשים, אופני בדיקות, מתודולוגיות בדיקה חדשות, פלטפורמות, כלי אוטומציה והמון המון דברים שמעשירים מאוד את תיק כלי העבודה של הבודק"

זהו כבר מצב שונה. זהו כבר מצב שבו הבודק נהיה יותר מבקר של החברה. זה לא קורה ביום ולא ביומיים, וברור לי לחלוטין שזה לא אפשרי בכל מקום ולא כל מקום רוצה בדבר הזה. אבל בעיניי זהו שלב אידאלי שכל איש QA (שזה ביטוי שאני הרבה יותר מזדהה איתו מאשר בודק) צריך לשאוף להגיע אליו.

איך כל החלקים מתחברים בחברה? האם כל חלק פועל באופן תקין? **איך החברה מנגנת והאם יש לי מה לתת בנקודה הזו כדי שדברים ייעשו על הצד היותר טוב?...** אלה הן שאלות מפתח שאיש QA שמגיע לרמה בוגרת יותר מתחיל לשאול את עצמו. וזה כמובן לא סותר את הדאגה המתמדת לכיסוי מלא של המוצר בבדיקות ולעושר הכלים והאפשרויות שאני יכול לתת לחברה, אדרבה – זה בא על גבי זה. **הרי אנחנו אנשי איכות, והאיכות אינה מצטמצמת רק לכדי פיצ'ר אחד בודד, כי כשאתה חושב ככזה זה כבר נהיה אופן חשיבה שבו אתה תופס את הכל, לא רק את המיקרו אלא גם את המאקרו.** וזה יכול להתבטא ביחסי הפעולה בין הפיתוח ל-QA, בצורת עבודה יותר נכונה, בהכנסה של נהלים וסדרים שיועילו לחברה, בשינוי תפיסה לגבי דברים ישנים שכדאי לחשוב עליהם מחדש, כמו איזו אוטומציה יש לנו, עד כמה היא יעילה, איך בנויה חלוקת העבודה אצלינו ביחס לבדיקות ואיך אנחנו בודקים בכלל – ומה אני חושב על זה ואיך זה ייקדם את החברה. **וזהו משפט המפתח – איך אני מקדם את החברה!**

המציאות

אין ספק, שיש לנהוג במתינות ואי אפשר לתת לשאיפות דרור באופן שאינו מתחשב במציאות, הרי זהו מתכון לנזק שיש מאוד להיזהר ממנו. כל אדם צריך לדעת את מקומו בעבודה ולדעת מה נדרש ממנו, מה ביכולתו לשנות ומה לא ביכולתו לשנות. לא בוגר יהיה ללכת ראש בראש עם גורם ניהולי שדעתו מוצקה ואינו מעוניין לקבל עוד נקודת מבט, או שסבור שזה לא מתפקידו של מאן דהוא – ובמקרה זה על אנשי ה-QA לחוות דעתם רק על דברים שלתפיסתו נמצאים בתחומם. צריך לחוש מה ניתן, מה אפשר ולצמוח בהתאם.

צריכים להיות חכמים, וזה אומר לא לוותר על הגדילה הפנימית שלנו כאנשי איכות, אנשי QA, ויחד עם זה להביא ברכה ולהועיל ולשפר ולהעניק במקומות שאנו יכולים ויש בידינו. תמיד בחיים החשבון הוא כפול – יש שיקול תיאורטי, מקצועי, איכותי של איך נכון להתבונן ואיך נכון לרצות שיהיה, ומאידך ישנה המציאות שלכל אחד מאיתנו מאפשרת כפי התנאים והסגנון הנהוג בה. והחוכמה היא לבנות את הגשר בין שני הרבדים.

להרגיש שמחה

הכיוון הזה שאני מנסה להתוות לעצמי ובכלל, נראה לי שהוא פתח גדול לצמיחה בתחום שלנו. מחד להתמקצעות משמעותית בכל המרחבים, ויש המון – ידני, API, אוטומציה, Back end, DATA, צד לקוח וצד שרת, הבנה של פלטפורמות, קונטיינרים, **פשוט אין סוף למה ללמוד ובמה להתמקצע. ויחד עם זה היכולת לגדול בחשיבה, בהיקף התפיסה, באמירה.** המוצקות הזו – במקצועיות מחד וההוספה של עוד רבדים של חשיבה מתרחבת, הכח הגדול הזה של היכולת בתוך המערכת מעניק שמחה, הרגשה טובה. אין פה עולמות של יובש, יש פה עולם שלם של אחריות, אחריות על המוצר מתוך הבנה מעמיקה של התפקיד כתפקיד מפתח בחברה. גם אם אחרים ואפילו ממונים עלינו לא תמיד מבינים עד כמה – אנו אלה שמחוייבים לדבר!

הכלי החביב עלי

לפני כשנה, מישהו שלח לי לינק לבלוג שעסק בשאלה מעניינת: "מה הכלי החביב עליך?" ("what is your favorite tool"). קראתי, וחשבתי: השאלה מעניינת. התשובה לעומת זאת, היתה לדעתי לא מועילה ובעיקר קיבלתי רושם שהכותב מנסה להציג את עצמו כאיש "עמוק". בקיצור, התאכזבתי. הרי מי ששאלה את השאלה לא רצתה לשמוע התחכמויות כמו "המוח שלי", "האנשים" או "כלים שמאפשרים לבנות כלים". היא רצתה לשמוע שם של כלי שתוכל להוריד מהרשת ולהנות ממנו.

האם הייתי עונה אחרת לשאלה? כן. ולהלן תשובתי. מטבע הדברים, טור כזה יהיה מלא בלינקים והפניות. על מנת שלא להכביד על הקוראים, ריכזתי את הכל בסוף.

קריטריונים

הקריטריון הראשון הוא לתת כלים שקשורים לבדיקות. אבל כמו שתראו מיד, רק חלק מהכלים שאציין כאן הם ממש כלי בדיקה. האחרים הם כלים כלליים, והסיבה שרשמתי אותם היא הקריטריון השני: כלים שבהם אני משתמש שוב ושוב לאורך הרבה שנים. כלומר, כלים שהיו רלוונטים ליותר מצורך חד-פעמי או פרוייקט אחד.

מהסתכלות על הרשימה, מסתבר לי שאני מעדיף כלים שניתן להפעיל דרך ה-Command Line. תגידו ...Old School אבל הרבה פעמים יותר יעיל לעבוד ככה, בלי התעסקות יתר עם העכבר ועם האפשרות הנוחה להרצת פקודות מתוך קבצי batch.

"מה הכלי החביב עליך?"

PICT (Pairwise Independent Combinatorial Testing)

יש לא מעט כלים ליצירת בדיקות לכיסוי קומבינציוני (all-pairs testing; combinatorial testing). את רובם לא ניסיתי, פשוט כי PICT הוא הכלי שאני מעדיף. זה כלי שנכתב על ידי יאצק צ'רוונקה (Jacek Czerwonka) ממיקרוסופט ופורסם כקוד פתוח. מעבר לייצור קלט לבדיקה ברמות שונות של קומבינטוריקה, הכלי מאפשר הגדרת תנאים לוגיים שבעזרתם ניתן לפלטר מראש צירופים מיותרים, לתת משקל לצרופים מועדפים, להגדיר צירופים הכרחיים ועוד. ממשק המשתמש הוא CLI, וכל ההגדרות (ערכים אפשריים של קלט, לוגיקה) נמצאות בקובץ טקסט יחיד. גם אם בסוף מחליטים לא להשתמש בצירופים שהכלי מייצר, עצם הגדרת הלוגיקה של הצירופים עוזרת להבין הרבה יותר טוב את התוכנה שאנו צריכים לבדוק.

PerlScript

כלי CLI ממש ישן שפותח על ידי דני פוט (Danny Faught) וג'יימס באך (James Bach), הגורו של בדיקות חוקרות (exploratory testing). הכלי בן למעלה מ-18 שנה, אבל עדיין עובד אחלה. הכלי מייצר קלט על פי כללים שונים, שמועתק אוטומטית ל-clipboard, לשימוש מיידי. זה יעיל מאוד בבדיקות חוקרות. למשל: יצירת קלט ארוך מאוד (נגיד, מיליון תווים) בלי צורך לפתוח מעבד מילים ולהתחיל לספור. או מחרוזת שמכילה את כל 255 התווים שמוגדרים ב-ASCII. הכלי מאפשר ייצור "מחרוזת סופרת" Counter String – מחרוזת שהתוכן שלה מתאר את אורכה. אפשר לייצר שתי מחרוזות כאלה, אחת שהיא קלט תקין (הבדיקה עוברת) ואחת ארוכה מידי ולכן הבדיקה נכשלת. מנקודה זו, הכלי מאפשר להמשיך ולייצר מחרוזות באורכים שממשמשים חיפוש בינארי אחרי האורך המקסימלי שעדיין עובר בבדיקה.

Rapid Reporter

כלי שפותח על ידי חברי הטוב שמואל גרשון. לכלי יש ממשק משתמש מינימלי של שורה אחת, שתמיד נמצא בקדמת המסך ללא קשר איזו אפליקציה נמצאת בפוקוס.

בשורה הזו אפשר לרשום הערות, הארות וכל מה שעולה בדעתכם תוך כדי הרצת בדיקות – בין אם אלה בדיקות חוקרות או הרצה של צעדים מוגדרים היטב. הכלי מאפשר גם לצלם את המסך בלחיצת כפתור. כל מה שרשמתם נשמר בקובץ csv, וצילומי המסך בקבצי jpg. בסוף אפשר לייצר מכל מזה דו"ח html יפה שמכיל גם את הטקסט וגם את צילומי המסך. מעבר למטרת השימוש המקורית (רישום הערות בזמן ביצוע בדיקות), אני משתמש בו, בין השאר, במקרים שאני בהרצאה וירטואלית ורוצה לשמור עותקים של חלק מהשקפים. בלחיצת כפתור אחת אני יכול לשמור את המסך, להוסיף הערות שעולות בדעתי תוך כדי – וכל זאת תוך הסחת דעת מינימלית מההרצאה.

"יש סיכון מובנה בשימוש בכלים מועדפים"

File Checksum Integrity Verifier (fciv.exe)

כלי פשוט ביותר שמחשב חתימה (hash) של קובץ או משווה חתימה של קובץ לחתימה צפויה. הכלי רץ במערכת Windows – אבל אני מנחש שיש משהו מקביל ב-Linux. בדיקת חתימה מאפשרת לבדוק אם קובץ נתון לא השתנה או אם התוכן של הקובץ תואם למה שציפינו. איך זה עוזר בבדיקות? לפעמים צריך לבדוק שפלט של בדיקה תואם לציפיות שלנו. אפשר כמובן לכתוב סקריפט בפייתון או שפה אחרת שיריץ את האפליקציה, יאסוף את הפלט לתוך משתנה, וישווה אותו. הפתרון הזה מסתבך לעיתים קרובות כשהפלט מכיל מספר שורות, משתנה בתלות בקלט, ויש בו גם טקסט שלא ממש קשור לבדיקה הספציפית (כגון כותרות, הודעות תוך כדי עיבוד וכו'). לפעמים אנחנו רוצים פתרון מהיר ופשוט בלי להתחיל לפתוח סביבת פיתוח. בעזרת fciv.exe אפשר ללכת על פתרון שהוא במידה רבה brute force:

- מריצים את האפליקציה הנבדקת בעזרת קובץ פקודות של מערכת ההפעלה (למשל קובץ bat. ב-windows). אפשר להריץ את האפליקציה מספר פעמים, עם קלטים שונים, על ידי הוספת שורות לקובץ הפקודות.
- מכוונים שהפלט יכתב לקובץ ולא למסך (בעזרת '>' או '>>').
- עוברים בשבע עיניים על התוצאות שהודפסו. אם הם נראות נכונות, שומרים את הקובץ בתור "תוצאות מצופות"
- מריצים את fciv על הקובץ ושומרים את החתימה בקובץ XML (fciv --help מסביר את זה)

מעכשיו, אפשר להריץ את הבדיקות שוב על וורסיה חדשה של היישום, לשמור את הפלט לקובץ, ולהריץ השוואה של החתימה שלו

מיכאל שטאל

ארכיטקט בדיקות תוכנה באינטל, ישראל
עוסק בעיקר בבדיקות מערכות משובצות מחשב.
במסגרת תפקידו, מיכאל מגדיר שיטות בדיקה ומתודולוגיות עבודה, עוסק בהדרכה ולפעמים אפילו מרשים לו לבדוק משהו (שזה הכי כייף).
מיכאל מציג תכופות בכנסים בארץ ובחו"ל ומלמד בדיקות תוכנה בפקולטה למדעי המחשב באוניברסיטה העברית.
ניתן לראות חלק מהמצגות והמאמרים שלו באתר www.testprincipia.com.

לעומת זו שציפינו. אם רוצים עוד קצת אוטומציה, אפשר לבחון את ערך משתנה המערכת ErrorLevel: אם ההשוואה הצליחה, ערכו הוא 0, ואם נכשלה, ערכו הוא 1.

Windows Subsystem for Linux (WSL)

גם משתמשים קבועים ב-Windows צריכים לעיתים גישה ל-Linux. זה יכול להיות כדי להריץ כלי שקיים בלינוקס ופותר בעייה מיידית; או צורך לפתח תוכנה שתרוץ על מכונת לינוקס, שאין אליה גישה נוחה (מישהו אמר עבודה מהבית ולא קיבל?), וכו'. אם אתם לא צריכים מכונת לינוקס "טהורה", רוב הסיכויים ש-WSL יהיה מספיק טוב בשבילכם. זו התקנה של לינוקס על Windows, ללא תסבוכות של dual boot או האיטיות המובנית בהתקנה על Virtual Machine. אחרי ההתקנה, WSL מאפשר להשתמש בכלים של windows לצורך גישה וניהול קבצים, להתממשק עם Visual Code, להתקין כמה ורסיות שונות של לינוקס ולעבור בקלות ביניהן, וכללית מקל על החיים למי שצריך לינוקס רק לפעמים.

"גם משתמשים קבועים ב-Windows צריכים לעיתים גישה ל-Linux"

ולסיום, עוד שני כלים שלא ממש קשורים לבדיקות אבל חביבים עלי, ועוזרים מאוד מידי פעם.

Bulk Rename Utility

כלי נהדר עם ממשק משתמש מזעזע, שמאפשר לשנות שמות של הרבה קבצים בבת אחת, על פי כל מיני כללים. למשל: תוספת של קידומת; או סיומת, או סתם שינוי השמות של כל הקבצים בספריה למספרים עוקבים. או קביעת השם על פי תאריך היצירה של הקובץ או תאריך השינוי שלו, או שינוי תאריך של הקובץ, או... או... המון אפשרויות, שכולן מוצגות על מסך אחד עם מלאנת'אלפים כפתורים קטנים וחלונות לקביעת ערכים. אחרי שתתגברו על ההלם של חוויית המשתמש הגרועה להפליא, אני מאמין שגם אתם תתלהבו.

MergeCells() Excel macro

אחד הדברים המעצבנים בהעתקה מטבלה ב-Word ל-Excel זה שנתונים שבטבלת וורד המקורית נמצאים בתא אחד, מופיעים באקסל בתאים שונים. משום מה אקסל חושב שכל שורה צריכה לקבל תא פרטי משלה, למרות שבוורד זה לא היה ככה. ועוד יותר מעצבן זה שהמפתחים של אקסל לא נתנו כלי או פונקציה שתטפל בעניין בצורה סבירה. זה כבר הפריע לי מזמן, אז כתבתי מקרו שמטפל בזה. אני עדיין משתמש בו היום מדי פעם. אפשר למצוא אותו באחת התשובות לשאלה ב-StackExchange (ראו לינק למטה).

אחלה, אבל...

יש סיכון מובנה בשימוש בכלים מועדפים: בגלל שהם מועדפים עלי, אני לא הולך לחפש אם יש משהו חדש ויותר טוב. אשמח לשמוע על כלים אחרים שעולים על אלה שרשמתי כאן, או כלים ש"עושים לכם את זה".

קישורים

PICT:

הורדה:
https://github.com/microsoft/pict/releases

קוד:
https://github.com/microsoft/pict

אתר הבית של המפתח:
https://www.jacekcz.com

כלים נוספים ל-combinatorial testing:

https://www.pairwise.org

PerlString

הורדה:

https://www.satisfice.com/blog/archives/22

אתר הבית של ג'יימס באך:

https://www.satisfice.com

Rapid Reporter

הורדה:

http://testing.gershon.info/reporter

אתר הבית של שמואל גרשון:

http://testing.gershon.info

File Checksum Integrity Verifier

הורדה:

https://archive.org/download/FCIV_v2_05/Windows-KB841290-x86-ENU.exe

WSL

https://learn.microsoft.com/en-us/windows/wsl/about

Bulk Rename Utility

https://download.cnet.com/Bulk-Rename-Utility/3000-2248_4-10174242.html

MergeCells() Excel macro

https://stackoverflow.com/questions/2089756/combine-multiple-cells-into-one-in-excel-with-macro

ראיון עם מנהלת בדיקות
ג'ני רויטמן, igentify

ג'ני רויטמן

גרה בשלומי, עליתי לארץ מאוקראינה ב-1994.
מאז התחתנתי, קנינו בית קרקע עם גינה ואנחנו מגדלים עץ לימון ותבלינים. הלימונים משמשים להכנת לימונצ'לו לשמחת חבריי בבית ובעבודה.
למדתי בביה"ס האזורי, אבל לא סיימתי ואין לי תעודת בגרות מלאה, בגלל יחידה אחת בהיסטוריה.
אחר כך, התחלתי ללמוד תואר של הנדסאי תוכנה אבל שוב - לא סיימתי.
הגעתי למסקנה שאני אוטודידקטית, והדרך הטובה עבורי ללמוד זה לבד.
במקרה לגמרי הגעתי לעבוד בחברה קטנה בקיבוץ ,להחלפה לחופשת לידה של אשת QA בחברה ו...
התאהבתי במקצוע. מצאתי קורס מקצועי עם השמה לעבודה ו... צללתי כולי לתוכו ל-4 חודשים של נסיעות לת"א עם יציאה מהבית ב-5 בבוקר וחזרה בלילה. מאז עבדתי בשתי חברות, אחת מתעסקת בפלטפורמות להימורים אונליין (ספירל סולושנס) ואחת במכשירים ותוכנות לשוק הרפואי (פיליפס).
הרגשתי שגדלתי מספיק כדי לקבל יותר אחריות ומצאתי את העבודה הנוכחית בסטארטאפ.
אחרי כמה שנים בחברה התקדמתי לראש צוות בדיקות. עברה לה עוד שנה כמעט וקודמתי למנהלת בדיקות. כיום מנהלת שתי קבוצות - בדיקות ידניות ובדיקות אוטומטיות.
למרות כל האתגרים, ויש הרבה, אני מאוד אוהבת את העבודה שלי ואת האנשים בחברה. מגיעה עם חיוך.

במה עוסקת הקבוצה וכיצד היא בנויה?

חברי הקבוצה מתחלקים לבודקים ידניים ובודקי אוטומציה. הבודקים הידניים יושבים בתוך צוותי סקראם יחד עם מפתחים ומנהלי המוצר (כל צוות פר מודול במערכת) ועובדים על הכנסה שוטפת של פיצ'רים למודול הרלוונטי. מתכננים בדיקות פר סטורי כאג'ייל ובדיקות רגרסיה פר אפיק. עושים מעבר אחד עם השני ועם פרודאקט ובסוף גם איתי. יש גם טק ליד בקבוצה שהוא יד ימיני ועזרה גדולה מאוד.

בודקי אוטומציה מתנהלים כצוות נפרד, עובדים על מימוש הטסטים הידניים מסט ריגרסיה שלנו.

אחת לרבעון, בשחרור גרסה, הבודקים מתאחדים לטובת הרצת סט הרגרסיה (Regression tests). יש בסביבות ה-1000 טסטים בסט בדיקות הזה, אנחנו מעריכים בשיחות עם מנהלי המוצר והפיתוח, מה להריץ הפעם ועל מה אפשר לוותר, בהתאם לתכולת הגרסה.

בדרך כלל מבקשים גם עזרה מעוד אנשים בחברה להריץ את הבדיקות, מה שאומר שכתיבת התסריטים צריכה להיות פשוטה ומדויקת, כדי שכל בן אדם יבין למה התכוון המשורר ולא רק המשורר עצמו.

איך נראה יום העבודה שלך?

בין אם אני עובדת מהבית או מהמשרד (אנחנו עובדים במתכונת היברידית) אני מתחילה את היום שלי במעבר על ריצת האוטומציה הלילית. מגיעה לישיבת צוות האוטומציה עם משימות לתיקון ואחזקה של הטסטים האוטומטיים.

עונה על השאלות של הבודקים הידניים, עושה מעבר על המשימות השוטפות שלהם, מזיזה כוחות לעזרתם אחד לשני, במידה ונדרש.
אחת לשבוע יש לנו ישיבת בודקים שבועית, שבה אנחנו עוברים על הנושאים השוטפים, פורקים תסכולים ומחפשים פתרונות ביחד.
מדי פעם מעבירים הרצאה על נושא נבחר או ריענון על המוצר שלנו והפיצ'רים השונים בו.

בין היתר, היום שלי מלא בישיבות עם מנהלי קבוצות, מנהל פיתוח ומנהל המוצר.

אנחנו דנים בנושאים שונים, מנסים לשפר תהליכים ולעזור לאנשים, לפתור בעיות של תמיכה שמגיעות מלקוחות.

מה האתגרים שלכם בתחום הבדיקות וכיצד אתם מתגברים עליהם?, האם נראה לך כי אלו ישתנו בעתיד?

כיוון שאנחנו חברת סטארטאפ, הדברים קורים כאן מהר, הדרישות יכולות להשתנות בין לילה וצריך להתאים את עצמך ואת הטסטים, לבדוק מחדש, לג'נגל כמה משימות במקביל וכו'.

אחד לרבעון מוציאים גרסה ומריצים סט של בדיקות ריגרסיה, שגודל כל הזמן. עבודה סזיפית ולפעמים אפילו מתסכלת.

כרגע אנחנו עושים מאמצים להעביר חלק מהטסטים לאוטומציה, לפתח יותר טסטים בצד הפיתוח – יחידה ואינטגרציה.

הדרך עוד ארוכה, לדעתי אנחנו בכיוון הנכון.

אני רואה את היכולת להגן על הצוות שלי מהפרעות ולחץ כאחד האתגרים הכי גדולים שלי.

מה האתגרים הייחודיים של קבוצת הבדיקות שלכם – וכיצד אתם מתמודדים איתם?

כיוון שאנחנו מפתחים מוצר רפואי יש לנו את נושא הרגולציה והאיכות – אנחנו חייבים להיות מדויקים בקישור הטסטים לדרישות, בכיסוי טוב של הדרישות, בציון סיבות נפילה של טסטים ובאופן כללי בכל הפראקטיקות של דוקומנטציה טובה. זה עובד גם לטובתנו, מכיוון שזה גורם לנו להיות יותר מסודרים. יש חשיבות מאוד גדולה לבאלאנס בין תיעוד בשביל תיעוד ותיעוד בשביל סדר.

לאחרונה הגיע אתגר חדש לכוחותינו, חלק מהאנשים בחברה גרים באוקראינה ועובדים משם בתנאי מלחמה – הפסקות חשמל, הפצצות וסירנות, חוסר שינה ומצב כללי מהורהר. אני מורידה את הכובע בפני האנשים האלה, שמצליחים על אף כל זאת לעבוד, להחזיק מעמד, לחנך ילדים ובכלל לחיות ולחייך בישיבות.

מאז תחילת המלחמה באוקראינה אני מתחילה כל בוקר בבדיקה שכולם בסדר.

מה היית ממליצה לבודקי תוכנה הנמצאים בתחילת הקריירה שלהם?

הייתי ממליצה לבחון טוב טוב האם אתה באמת אוהב ומתאים להיות בודק. לא להתחיל קורס בודקים, כי הבטיחו לך שזאת הדרך הקלה להיכנס להייטק ועוד כל מיני סיבות מסוג זה.

לפי דעתי, בודק זה סוג של בן אדם יותר מאשר לימוד, כמה אינטנסיבי ומדויק שלא יהיה.

אם אתם בחיים הרגילים שלכם רואים את כל הטעויות ואי הדיוקים מסביב, כמו תריסי מזגן שבורים במקום מסוים באוטובוס, מכונת קפה, שאם לוחצים על אותו כפתור פעמיים היא נתקעת, יש לכם בחילה מאתרים לא מגיבים, לא מסודרים, אפליקציות נתקעות רק אצלכם, כי לחצתם על משהו לא טריוויאלי – אתם בודקים בנשמה.

ראיון עם מנהלת בדיקות
ג'ני רויטמן, igentify

באילו אתגרים ניהוליים הנך נתקלת?

האתגר וההשקעה העיקרית שלי - באנשים. העשרת ידע שלהם, תמיכה בבעיות שצצות להם, איזון בית-עבודה עבורם. להגיד להם בוקר טוב ולילה טוב, להודות להם על העבודה מאומצת, לארגן איתם ימי כיף של הקבוצה, להכיל אותם עם כל האתגרים האישיים והמקצועיים שלהם.

עוד היבט חשוב בניהול לטעמי - זה שיקוף של האתגרים והצלחות מעלה למנהליי ומטה לבודקים והעברת פידבק לשני הצדדים.

יש גם את האתגר של העובדים בחו"ל, אומנם כולנו עובדים לרוב מהבית, אבל הייתי מאוד שמחה לראות ולחבק אותם לעיתים יותר תקופות.

המראיין: ניצן גולדנברג
מזה 7 שנים בתחום בדיקות התוכנה, בתפקיד
נוכחי מהנדס בדיקות בכיר בחברת Seatgeek.
מנהל קבוצת ה-AB של ארגון ITCB®
המוביל הראשי של קבוצת המיטאפ TestIL ומרצה בקורסים לבודקי תוכנה

פספורט קבוצתי
igentify

סוג המוצר הנבדק: מוצר וב, יועץ גנטי דיגיטלי, המסייע ליועצים גנטיים לזרז ולייעל תהליכים

גודל קבוצת הבדיקות: קבוצת הבדיקות כוללת 5 בודקים ו-2 אנשי אוטומציה

וותק הבודקים: טק ליד 3 שנים בחברה, הבודקים בסביבות שנה - שנתיים

מבנה הקבוצה: הבודקים הידניים נמצאים בקבוצות סקראם ועובדים כחלק מהם. הבודקי אוטומציה הם צוות נפרד

סוגי בדיקות: כרגע רק בדיקות פונקציונליות, לוקאליזציה ותוכן. מתכננים בדיקות עומסים בעתיד

שיטת עבודה: עובדים באג'ייל, ספרינטים של שבועיים

כמות המוצרים הנבדקים וקצב שחרור

גרסאות: 3-4 גרסאות לשנה, מוצר כולל 3 מודולים ואינטגרציה ביניהם

- ארגון ISTQB אישר ופרסם לאחרונה גרסה חדשה של תכני הלימוד להסמכה הבסיסית (סילבוס CTFL - רמה בסיסית - גרסה 4.0). המטרה הייתה להפוך את הסילבוס לפחות תלוי במסלול פיתוח (SDLC) ספציפי, תוך מיזוג מושגים, והתייחסות לטכניקות וגישות המזוהות עם מסלולי פיתוח איטרטיביים כגון SCRUM, Agile ו-XP ובנוסף DevOps (CI/CD). הדבר ניכר בכך שחלק מהחומר החדש בגרסה 4.0 מקורו בתכני הלימוד של ההסמכה ל-Agile (CTFL-AT).
- בהתאם לכך ארגון ITCB מכין סילבוס ובחינות מותאמים בעברית, ועל המכללות השולחות את תלמידיהן לבחינת ההסמכה להתאים את חומרי הלימוד שלהן בכדי שיהיו מיושרים עם תכני ורמת בחינת ההסמכה לכל המאוחר עד תחילת מאי שנת 2024.
- בודקים שברשותם הסמכת CTFL קודמת לגרסה 4.0 לא יידרשו לעבור בחינה מחודשת, והסמכתם נשארת בתוקף.

תשתית אוטומציה יעילה בעבודה נכונה עם XML

נתי צדוק

מפתחת תשתיות אוטומציה, עובדת כ-8 שנים בחברת Bmc Software

בעלת ידע נרחב בבדיקות תוכנה וכתיבת אוטומציה ,UI Backend ו-API

בעלת תואר ראשון במדעי המחשב ומתימטיקה

בעבר מרצה לשפות תכנות במכללה הטכנולוגית "תל-חי"

מנהלת סניף בארגון she codes

בעבודה עם פרויקטים של Maven, חשוב לארגן טסטים בצורה מובנית ויעילה. גישה אחת היא לחלק בדיקות למספר קבצי XML במקום לתחזק קובץ אחד גדול המכיל את כל הבדיקות, חלוקת בדיקות למספר קבצי XML מקלה על ניהולם. כאשר פרויקט גדל בגודלו ובמורכבותו, מספר הטסטים גדל גם הוא, ונרצה יכולת שליטה על כל הבדיקות שנעשות באוטומציה, נראה את היתרונות בכך.

חלוקה לוגית נכונה

ניהול מספר רב של בדיקות בקובץ XML בודד עלול להפוך למסורבל, כאשר בדיקות מפוזרות על פני חלקים שונים של הקובץ, מה שמקשה על איתור בדיקות ספציפיות. עם קבצי XML מרובים, ניתן לארגן בדיקות באופן לוגי תקין לקבצים נפרדים על סמך סוגם או הפונקציונליות שלהם, מה שמקל על הניהול והתחזוקה שלהם.

בדיקה סלקטיבית וחסכון בזמני ריצה

יתרון נוסף של חלוקת בדיקות למספר קבצי XML הוא שהיא מאפשרת בדיקה סלקטיבית. במקרים מסוימים, ייתכן שלא יהיה צורך להפעיל את כל הבדיקות, והפעלת כל הבדיקות עשויה להימשך זמן רב, ולהאט את תהליך הריצה. על ידי חלוקת בדיקות למספר קבצים, ניתן להפעיל באופן סלקטיבי בדיקות ספציפיות או קבוצות של בדיקות, ולחסוך זמן ומשאבים.

שיתוף בין פרויקטים

חלוקת בדיקות למספר קבצי XML מקל גם על שיתוף ושימוש חוזר בבדיקות בין פרויקטים. בפרויקט גדול ומורכב, מקובל לקיים טסטים המשותפים בין חלקים שונים של הפרויקט. על ידי הצבת בדיקות משותפות בקבצי XML נפרדים, ניתן לפנות אליהם בקלות ולעשות בהם שימוש חוזר בחלקים שונים של הפרויקט. זה יכול לחסוך זמן ומאמץ, מכיוון שאין צורך לשכפל בדיקות על פני חלקים שונים של הפרויקט.

הרצה במקביל

יתרה מכך, על ידי חלוקת בדיקות למספר קובצי XML, ניתן להריץ בדיקות במקביל, מה שיכול להפחית משמעותית את הזמן שלוקח להפעיל בדיקות. בדיקה מקבילה כוללת הפעלת מספר בדיקות בו-זמנית על מכונות שונות, מה שמאפשר להריץ מספר רב של בדיקות בפרק זמן קצר יותר. זה יכול לעזור להאיץ את תהליך הריצה, להוביל למשוב מהיר יותר ותיקוני באגים מהירים.

ניפוי באגים ובידוד כשלי בדיקות

יתרון נוסף של חלוקת בדיקות למספר קבצי XML הוא בכך שהיא מספקת דרך לנפות באגים ולפתור תקלות בבדיקות בצורה יעילה יותר. כאשר כל הבדיקות כלולות בקובץ XML בודד, זה יכול להיות מאתגר לקבוע איזו בדיקה גרמה לכשל, במיוחד אם הקובץ מכיל מספר רב של בדיקות.

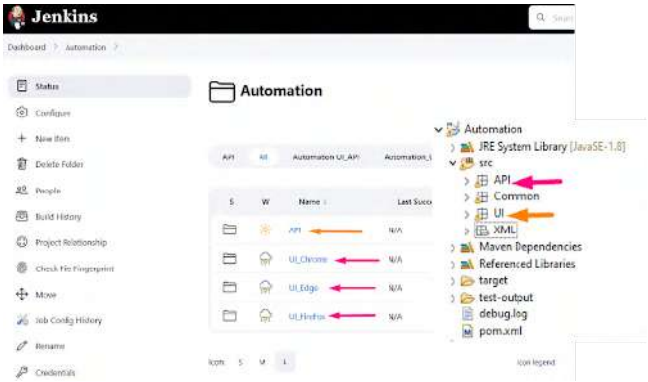
על ידי חלוקת בדיקות למספר קבצי XML, קל יותר לבודד ולזהות את המקור לכשל בבדיקה. אם בדיקה נכשלת, יכולים להסתכל על קובץ ה-XML המכיל את הבדיקה המסוימת ולמקד את מאמצי ניפוי הבאגים בקובץ הספציפי הזה. זה חוסך זמן ומאמץ על ידי מניעת הצורך לסרוק קובץ גדול כדי לזהות את הבדיקה הספציפית שנכשלה. בנוסף, אם ידוע כי בדיקה מסוימת נכשל או נוטה לכישלון, אנו יכולים להשבית את הבדיקה הספציפית הזו על ידי הוספת הערה בקובץ ה-XML הרלוונטי. זה מונע את הפעלת הבדיקה מבלי להשפיע על הבדיקות האחרות בפרויקט.

עבודה מול Jenkins

בעת שימוש בכלי אינטגרציה מתמשכת ופריסה רציפה (CI/CD) כגון Jenkins, חלוקת בדיקות למספר קבצי XML יכולה לספק אפשרויות ניפוי באגים נוספות. ניתן להגדיר את Jenkins להריץ תת-קבוצה של בדיקות בהתבסס על הקובץ הספציפי ששונה או נכשל בבנייה קודמת. המשמעות היא שאם בדיקה נכשלת, אנו יכולים להריץ מחדש במהירות רק את הבדיקות בקובץ המושפע, במקום להפעיל מחדש את כל הבדיקות.

חלוקה נכונה לתיקיות בג'נקינס

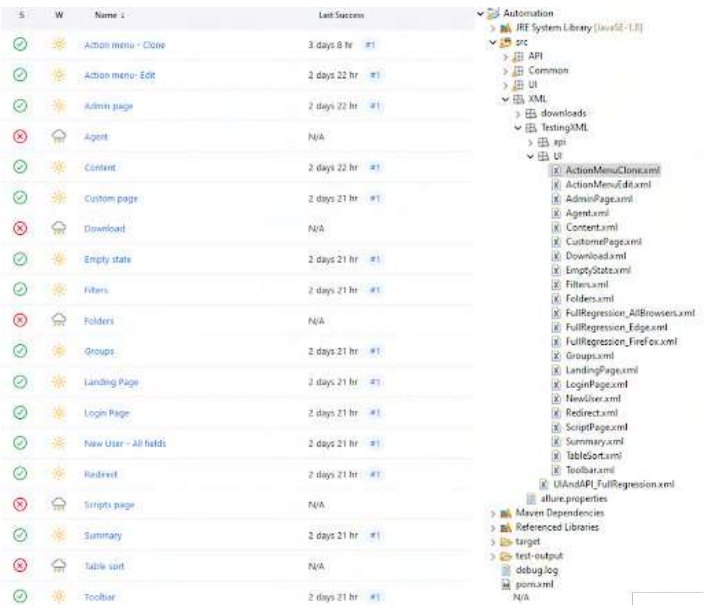
ניצור מבניות זהה בין תשתית הקוד של האוטומציה לבין התיקיות בג'נקינס, נשכפל את בדיקות ה-UI לתיקיות לפי דפדפנים שאנחנו תומכים, במידה ויש לנו בדיקות API נחלק את הבדיקות לתיקיה נפרדת.



בכל תיקייה של UI ירוצו הטסטים הקיימים באוטומציה אך עם flag -Dbrowser של הדפדפן הספציפי, מספר התיקיות יהיה על פי מספר הדפדפנים שאנחנו תומכים. שורת ההרצה של ג'וב מתוך תיקיית UI_Edge לדוגמא תראה כך:

מקביליות xml וג'ובים בג'נקינס

עבור כל קובץ xml המכיל סוויטה של טסטים ניצור עבורו ג'וב בג'נקינס שיריץ רק אותו.



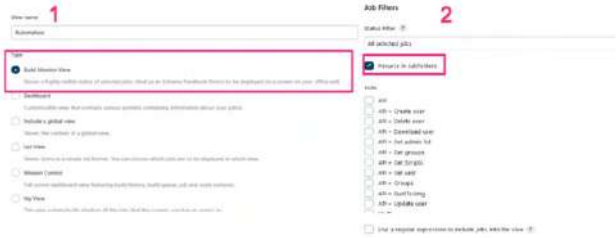
תשתית אוטומציה יעילה בעבודה נכונה עם XML נתי צדוק

שורת הפקודה תהיה :

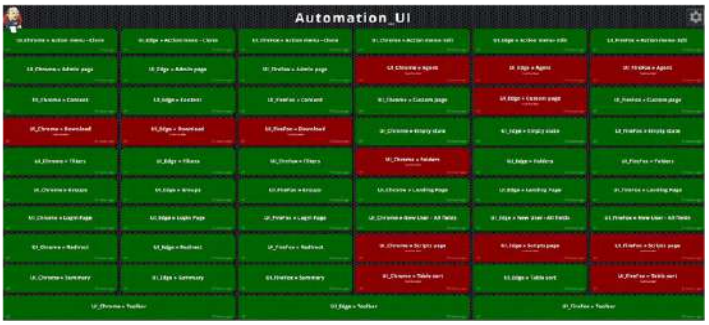
Monitor view

כחלק מהיתרון שאנחנו מקבלים עם חלוקת קבצי xml ללא שימוש בקובץ אחד שמכיל את כל הטסטים ולאחר התקנת monitor view plugin ב-Jenkins אנחנו יכולים ליצור לוחות לקבלת סקירה ברמה גבוהה של התקינות והסטטוס של מספר ג'ובים, עם יכולת הרצה של טסטים בודדים, יכולת דיבוג גבוהה יותר ויכולת אפיון בעיות בצורה רחבה ופרטנית כאחד. נוכל ליצור לוחות לפי הצורך - לוח שיכיל טסטים לפי דפדפן מסוים, לוח שיכיל רק בדיקות UI ואחד שיכיל הכל. תצוגה זו שימושית למעקב אחר ההתקדמות של מספר ריצות או ג'ובים במבט חטוף, מבלי צורך לנווט לכל אחד בנפרד.

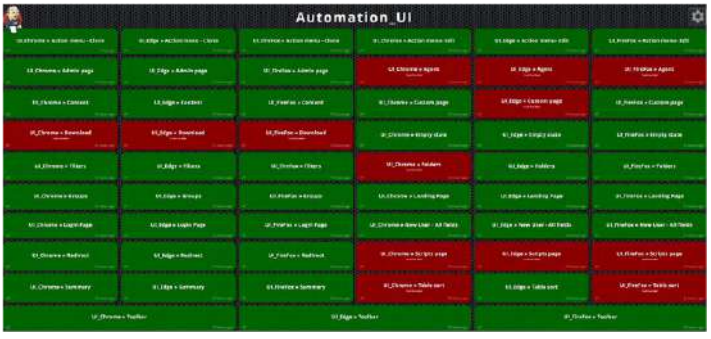
אחד היתרונות המרכזיים של השימוש ב-Monitor View הוא שהיא מספקת נראות בזמן אמת למצב של מספר ריצות או ג'ובים, ומאפשרת לצוותים לזהות במהירות כל בעיה או כשל שדורשים התייחסות. על ידי מעקב בזמן אמת, צוותים יכולים לנקוט בפעולה מיידית כדי לפתור בעיות ולמנוע מהן להסלים לבעיות משמעותיות יותר.



ניצור בורד אחד עבור כל האוטומציה של ה-UI

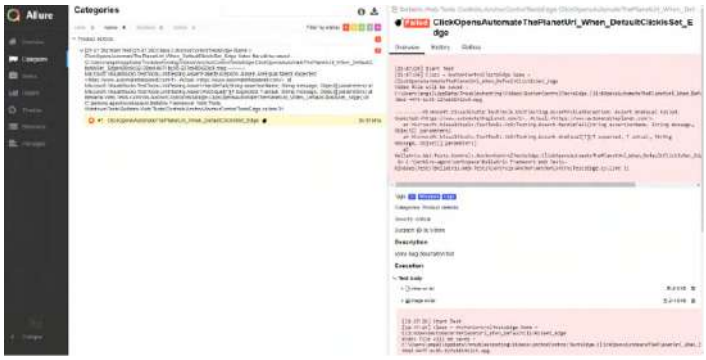


ונשתמש ביכולת הזאת ליצור גם בורד עבור אוטומציה של UI ו-API



כל לחיצה על ריבוע מתוך הבורד תוביל אותנו אל הג'וב הספציפי, אם צבעו אדום (משמעות שנכשל) ונוכל לחקור. ניגש אל דוח ה-allure שלו, ונחקור בקלות את הנפילה של הג'וב הספציפי הזה.

לדוגמא עבור הטסט Agent נקליק עליו ונקבל את הדוח, ששם נוכל להגיע לנפילה שגרמה לטסט להכשל כולל אפשרות לצילום מסך מתוך הריצה בזמן אמת.



קבצי FullRegression

למרות שהעבודה השוטפת תהיה עם קבצי xml קטנים נרצה לשמור על קובץ מקיף וריכוזי שכולל את כל הטסטים, כדי לקבל תמונה כוללת של האוטומציה, נבנה 3 קבצים שיתנו לנו מבט ב-high-level לפי סדר גודלם.

1. Full_regression.xml

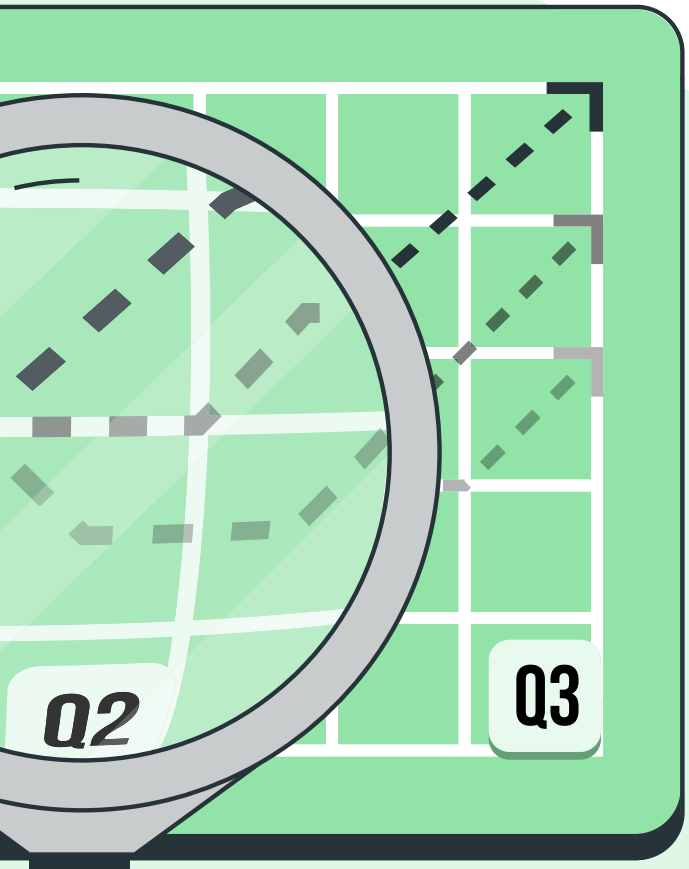
יחזיק את הסוויטה שמכילה את כל הטסטים הקיימים והרצה שלו מול ג'וב מותאם בג'נקינס ייתן לנו בעצם אפשרות למבט ב-high-level על כל הטסטים של ה-UI עבור דפדפן ספציפי.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="FullRegression_UI_FireFox">
<parameter name="browser" value="FireFox"/>
    <listeners>
        <listener class-name="utils.ListenerClass"></listener>
    </listeners>
    <!--ActionMenuClone -->
    <test name="ActionMenuClone">
        <classes>
            <class name="ui.test.ActionMenuClone"/>
        </classes>
    </test> <!--ActionMenuClone-->

    <!--NewUser -->
    <test name="NewUser">
        <classes>
            <class name="ui.test.NewUser"/>
        </classes>
    </test> <!--NewUser-->
```

2. Full_regression_AllBrowsers.xml

קובץ שלא מכיל טסטים אלא הוא מכיל סוויטות של טסטים. וייתן לנו מבט גדול יותר על **כל** האוטומציה שלנו עבור כל הדפדפנים שאנו תומכים




```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Full_Regression_AllBrowsers">
    <suite-files>
        <suite-file path="./FullRegression_FireFox.xml"> </suite-file>
        <suite-file path="./FullRegression_Edge.xml"> </suite-file>
        <suite-file path="./FullRegression.xml"> </suite-file>
    </suite-files>
</suite> <!-- Suite -->
```

3. UIAndAPI_FullRegression.xml

למעשה זה קובץ שנותן תמונה כללית ורלוונטי לפרויקטים שמכילים בדיקות UI וגם בדיקות API

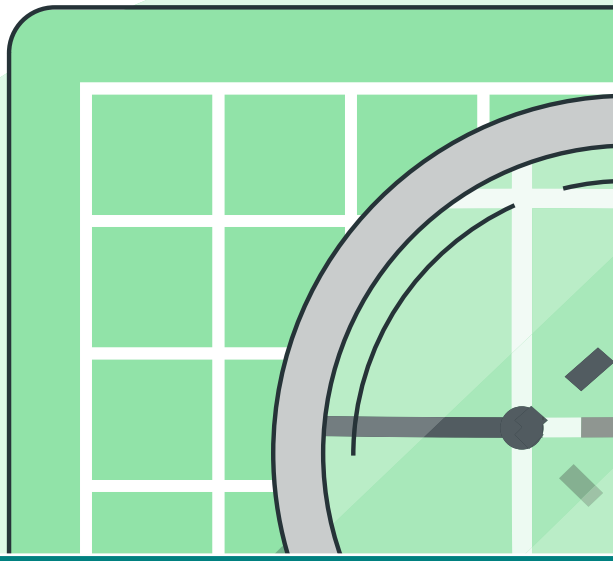
```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="UI and API">
    <suite-files>
        <suite-file path="UI/FullRegression_FireFox.xml"> </suite-file>
        <suite-file path="UI/FullRegression_Edge.xml"> </suite-file>
        <suite-file path="UI/FullRegression.xml"> </suite-file>
        <suite-file path="api/FullRegression.xml"> </suite-file>
    </suite-files>
</suite> <!-- Suite -->
```

דוח allure של הרצה מסוג קובץ כזה יראה כך

לסיכום

חלוקת בדיקות למספר קבצי XML בפרויקט של Maven מביאה יתרונות רבים. זה מקל על ניהול בדיקות, מאפשר חלוקה לוגית נכונה של הרצת טסטים, מאפשר בדיקות סלקטיביות וחסכון בזמני ריצה , מקדם שימוש חוזר בבדיקות בין פרויקטים ומאפשר בדיקות מקבילות, מאפשר הרצה במקביל, אפשרות לניפוי באגים מהיר יותר, ויכולת ניצל האפשרויות הקיימות ב-Jenkins בצורה עמוקה יותר. כל היתרונות הללו יכולים לעזור לשפר את היעילות של תהליך פיתוח האוטומציה, תחזוקה באופן שוטף, וכתוצאה מכך לאוטומציה איכותית יותר ולשחרור מהיר יותר של קוד.

לכל שאלה – ניתן לפנות אלי בלינקדאין

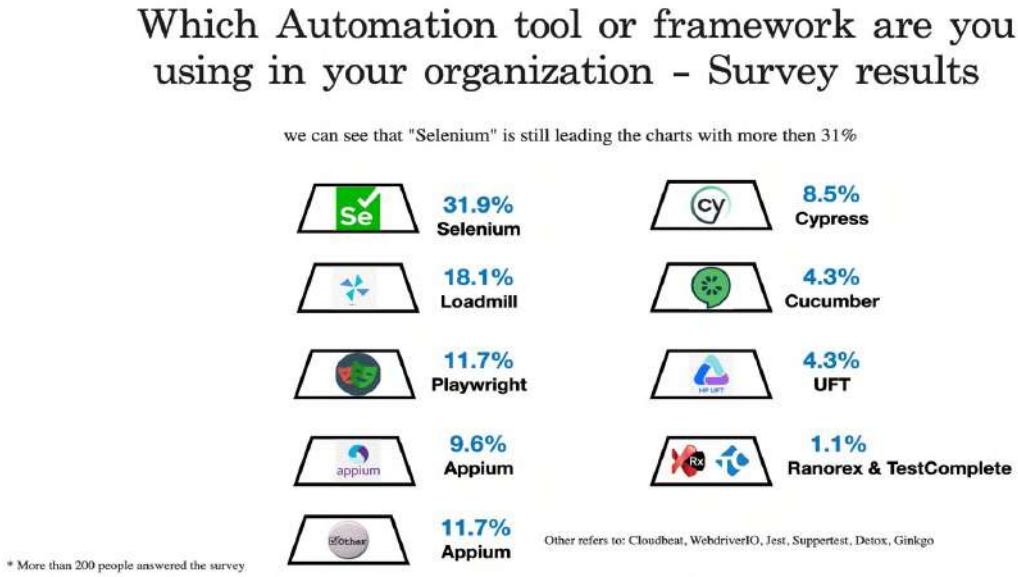


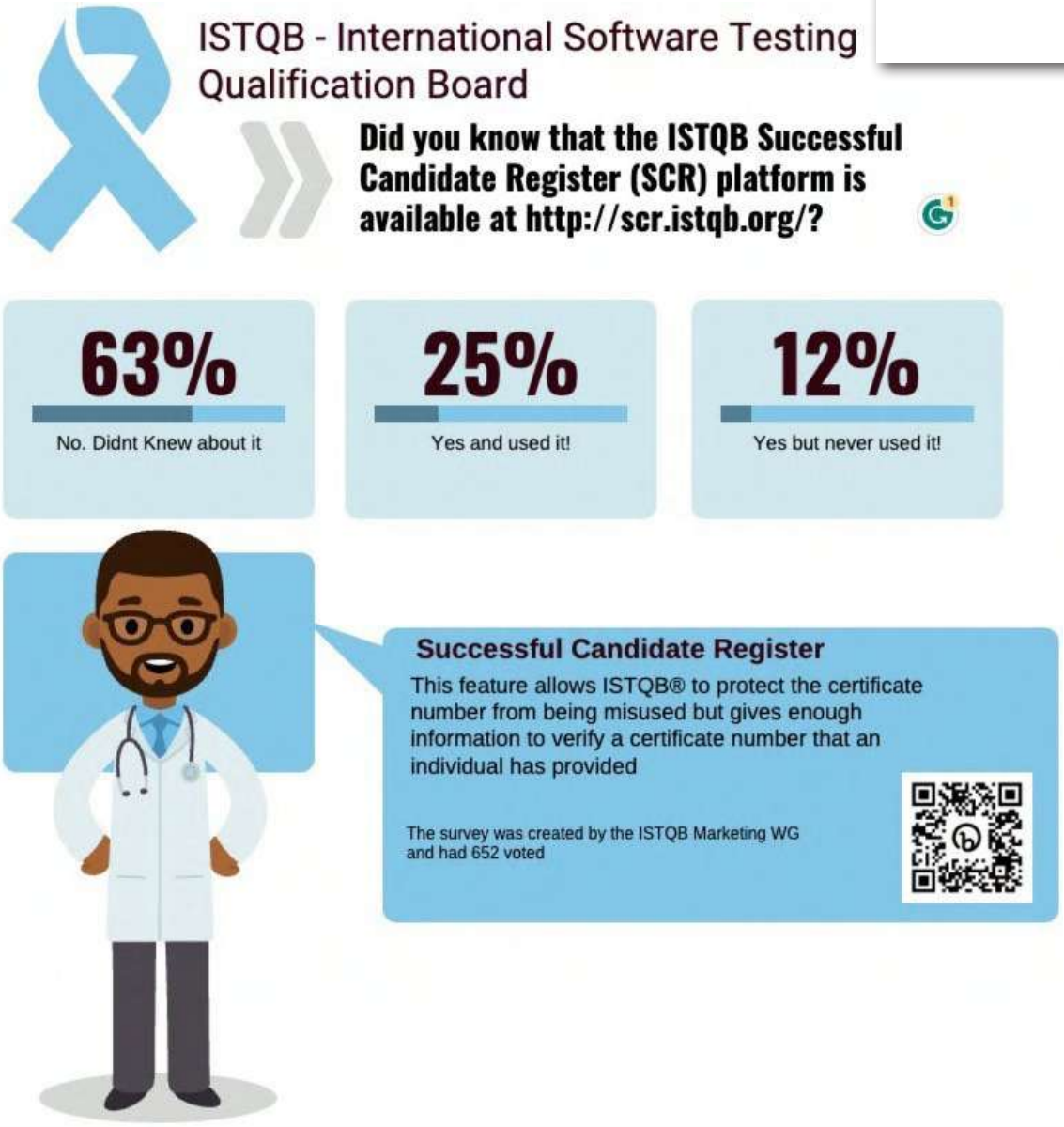
מה הסקרים אומרים?...

ניצן גולדנברג

מזה 7 שנים בתחום בדיקות התוכנה, תפקיד נוכחי מהנדס בדיקות בכיר בחברת SeatGeek, מנהל קבוצת ה-AB של ארגון ITCB® המוביל הראשי של קבוצת המיטאפ TestIL, מרצה בקורסים לבודקי תוכנה

טור זה מציג ממצאי סקרים המציגים מגמות מרחבי העולם לעיונכם.





משולחנו של שביט -
אל תפתח מטרייה לפני שהתחיל לרדת גשם

בכתבה הזו אני רוצה להתייחס לנושא של תיעוד וכתיבת טסטים.
אבל אני רוצה לדבר יותר על ה-"אזורים האפורים" שמסתתרים בו – בחלק מהמקרים נקרא לבדיקות הללו Error handling שזה למעשה 'טיפול בחריגות'. ובחלק אחר מהמקרים לא תהיה להן הגדרה ברורה משום שכמעט בלתי אפשרי לכתוב בדיקות לבעיות שטרם קרו ('בעיה' היא **תוצאה** של בדיקה).

רבים מהבודקים המתחילים נוטים להתבלבל ולכתוב בדיקות של "מה יקרה אם.."
הבעיה בכך היא שאנחנו לא יכולים להרשות לעצמנו לכתוב עכשיו כמה עשרות טסטים על משהו שעוד לא קרה וספק סביר שגם יקרה.
אנחנו בעצם מנחשים או תוהים באפלה ובכך אנחנו עלולים לבזבז את זמננו היקר מפז על טסטים רבים ש "ילכו לפח" משום שלא תהיה דרך להריץ את אותם טסטים.

כלומר, אנחנו לא יכולים לחזות או לצפות תקלות. עדיין לא.
בנוסף חשוב לחדד שגם לכלי בינה מלאכותית כ-ChatGPT אין עדיין את היכולת לחזות תקלות.
עם כל הכבוד ויש כבוד.

אם כך, נשאלת השאלה מה ניתן ונכון לבדוק?
ניתן לבדוק שהמהלך התקין של הטסט מתבצע.
ניתן לבדוק שאנחנו מקבלים ב-UI את המצופה.
ניתן לוודא את תגובת השרת המצופה.
לוודא שאנחנו מקבלים בחזרה מהשרת Data מסוים ע"י - API.
ניתן לוודא בדיקות ספציפיות ומכוונות של המערכת.

הדברים תלויים מאוד בסוג הבדיקה, המערכות הקשורות לאפיון שהוגדרו ע"י הפרודקט.

אז איפה כאן הקטגוריה של Error handling נכנסת לתמונה כאשר לא מעט פעמים גורמת לחלקנו לבלבול מסוים?
ובכן, אנחנו קודם כל יוזמים התקלה באופן זדוני. ומכוונים לכך שאולי ונצליח לגרום למערכת להגיב בצורה לא טובה ואף גם לגרום לה אולי להיתקע, ליפול או לקרוס ושאר ירקות.

כאשר נתקדם שלב אחד קדימה. בהנחה ואכן הצלחנו להתקיל את המערכת.

האם כדאי לנו "לשמוח" שמצאנו באג ו"לדפוק ספרינט מהיר" עם העכבר לעבר הטאב הפתוח של ה-JIRA הממתין לנו בציפייה?
או שמא כדי לנו לספור עד 10, לנשום עמוק ולנסות להבין מה קורה כאן.

אנחנו גם נעצור כאן? או שהוא קורה ומשתחזר בעוד 3 מקומות שונים במערכת, גורם לנו לעוד 5-6 באגים אחרים שבעקבות כך "יעזרו" לנו להזיע הרבה בשבוע הקרוב?

זו דילמה חשובה משום שבודקים רבים בתחום זה שאת חלקם הכרתי באופן אישי, גם כאלו שהם חדשים לתחום הזה וגם כאלו שכבר יותר ותיקים ונמצאים במערכת 3-4 שנים ומעלה והם כבר "עייפים".. רוב הסיכויים שיעצרו כאן, יתעדו את התקלה ולא ימשיכו הלאה מתוך איזשהו ויתור, מתוך עצלות ומתוך עוד סיבות שונות שאינן ידועות לנו.

כל זה ייקח ויגבה מאיתנו זמן, כל זה מצריך מאיתנו סבלנות.
כל זה יצריך מאיתנו "להיכנס לאירוע!"

הצורך ברגעים אלו הוא לזכור שאנחנו בעצם רצים למרחקים ארוכים בתחום הזה ולא נמצאים כאן כדי למצוא באג אחד ליום סגריר ו"עשינו את היומית שלנו".

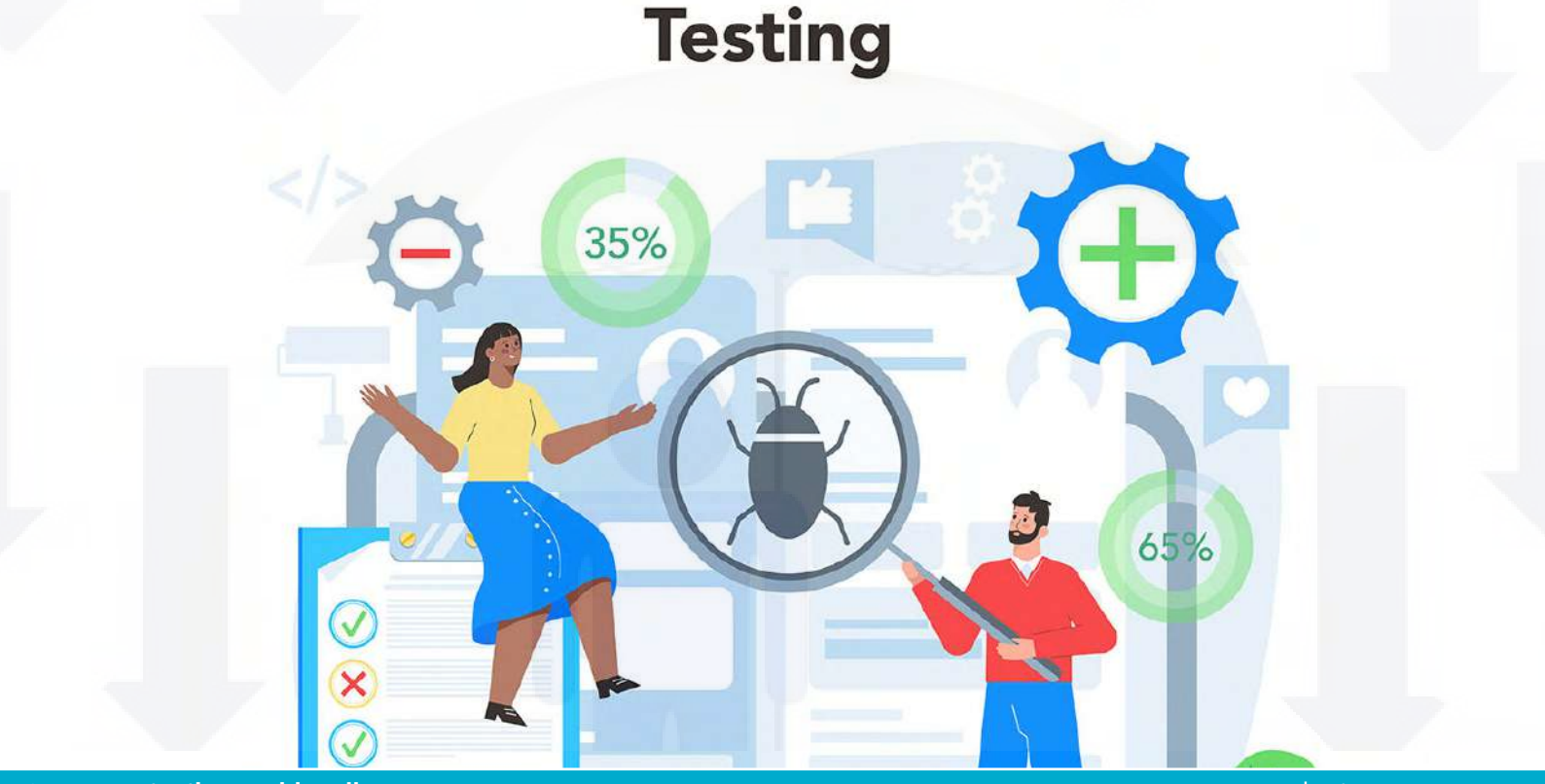
חזרה לענייננו,
בכל זאת.. איך אפשר ללא קצת רוח קרב QA? 😊
חשוב שנשים לב לפרטים האלו ונדע גם לנתח במערכת במתודה לא מתודה, מה שנקרא Explore משום שאם הצלחנו להתקיל את המערכת לפי טסט כתוב – זה נהדר כי תפסנו באג. ומצד שני, אנחנו צריכים לשים לב ולהיות ערים ל-איך המערכת מגיבה לתקלות האלו משום שלא נוכל לכתוב טסטים שצופים את התקלות האלו מראש ולא נדע להיערך לכל דבר מבעוד מועד, ולמעשה...לא נוכל לפתוח מטרייה לפני שהתחיל לרדת גשם.

זכרו, הדבר הנחמד בגשם זה שהוא תמיד נעצר לבסוף.

אני מאחל לכם המשך שבוע נעים
ושלעולם לא תצעדו לבד

שביט ג'רסי

מנהל בדיקות בחברת Wisetamp, בעל תואר ראשון בהנדסת תעשייה וניהול מהטכניון, בעל 10 שנות נסיון בתחום הבדיקות, מתוכם מעל 7 שנים מדריך עצמאי ל-QA.
בעל סדרת הסרטונים השבועית "QA ללא הפסקה"



מזל טוב לאלופי הבדיקות של ישראל

זו השנה השביעית לתחרות הבדיקות הישראלית ISTC שהתקיימה בזכות יוזמתה של אנה לוקובסקי ובתמיכתם של - ITCB העמותה הישראלית להסמכת בודקים, Matrix Testing & Automation, ג'ון ברייס הדרכה, Inflectra, Cloudbeat, Practitest, Jam.dev.

התחרות מיועדת לבודקים מנוסים וגם לאלו החדשים בתחום, המשוגעים לדבר אשר אוהבים אתגרים ונהנים לבצע בדיקות. במסגרת התחרות התבקשו המתמודדים לבדוק מוצרים אמיתים בתנאים קיצוניים ובזמן מוגבל.

המשתתפים נדרשו לזהות באגים, לדווח עליהם באופן ברור המאפשר תיקון, לערוך בדיקות לא פונקציונליות לפי הצורך ולסיום גם להגיש דוח סיכום מצב המוצר אשר יהווה ערך נוסף לחברות המוצר מבחינת שחרור הגרסה.

במוקדמות שנערכו השנה בשני מועדים, ב-19 באפריל 2023 וב-21 באפריל 2023 השתתפו 35 צוותים ודווחו 370 תקלות.

בגמר שנערך ב-18 ליוני 2023 השתתפו 5 צוותים ונמצאו 80 תקלות.

צוות השופטים במוקדמות ובגמר כלל מנהלים ויועצי בדיקות בכירים המייצגים את חברות ה-IT וההייטק המובילות בישראל: **ירון צוברי** – נשיא ITCB@, **ניצן גולדנברג** – העורך הראשי של מגזין "עולם הבדיקות", יו"ר ומנהל התחרות, **אסתר צבר** - מקימת AQA, **קובי יונסי** - מייסד QAMasters.co.il, **מור קורם** - ר"צ ב-Webbing, **סלבה פשנין מוביל בדיקות** - חברת Palo alto networks, **גיל שקל** - ראש צוות אוטומציה, **הדס חסידים** – ראש מחלקת הנדסת תוכנה – סמי שמעון, **מיכל טל** – מנהלת ומדריכת בדיקות

פרסים יקרי ערך הוענקו לזוכים בגמר

מקום – I
טיסה ,אירוח והשתתפות בכנס Agile Testing Days 2023 שיתקיים בפוטסדאם, גרמניה בנובמבר 2023 בחסות:
ITCB, PractiTest, Cloudbeat, Jam.dev

מקום – II
קורס מקצועי (עד 40 שעות) ואוזניות Airpods pro בחסות מכללת גו'ן ברייס

מקום – III
קורס מקצועי (עד 40 שעות) בחסות מכללת גו'ן ברייס

תואר Best Bug Award במוקדמות ובגמר
שוברים בסך 500 ₪ כל אחד בחסות ITCB & Blirokratia

העולים לגמר
זכו בכרטיס חינם **לכנס Testing & Automation GeekWeek 2023**

שלושת הזוכים קיבלו שוברי הנחות לבחינות הסמכה בסיסיות או מתקדמות בחסות ITCB

הזוכים של ISTC 2023

מקום ראשון

אלכסנדר קומנוב ועומרי ממן - צוות KoMamanov

שנינו עובדים ביחד בחברת סיטגיק. עד לפני שנה וחצי היינו באותו צוות אוטומציה.
אנחנו הגענו לעולם ה-QA אחרי הסבה ממקצוע אחר: עומרי היה שופט כדורסל ומורה לחינוך גופני ואלכס היה שוטר. אני חשוב שאנו חולקים את המוטיבציה והדחף ללמוד ולהיחשף לנושאים חדשים. תחום ההייטק בכלל ועולם הבדיקות ופיתוח התוכנה בפרט מהווים קרקע פורייה ללמידה והתפתחות מתמדת.

אלכסנדר קומנוב

עומרי ממן

מקום שני

אלכסנדר זבולוקו וכינר אלקיים - צוות Automation Avengers

Our team, the Automation Avengers, is a dynamic duo on a mission to conquer the world of software testing! We first united at SeatGeek and have since become an unstoppable force in the world of automation. With a passion for hunting down bugs and a hunger for innovation, we live for the thrill of competitions like the ISTC. We are ready to take on any challenge and bring home the trophy as champions

אלכסנדר זבולוקו

כינר אלקיים

מקום שלישי

דנית דור ונתנאל הרוש – צוות The Killers

שני בודקי תוכנה צעירים ומוכשרים, יצרו קשר מדהים מהרגע שנפגשו במהלך קורס QA שלהם במכללת ג'ון ברייס עוד בימי תקופת הקורונה המורכבת. משיתוף הפעולה הראשון שלהם בפרויקט בלימודים, דרך אינספור שיחות זום שנמשכו עד השעות הקטנות של הלילה, השותפות שלהם הפכה לבלתי שבירה ועד היום הם כמו משפחה. בהשראת המרצה הנערץ שלהם, מגדלור של מומחיות ודוגמת מופת עבור תלמידיו הרבים, אשר במקרה גם הוא כמובם עשה הסבה מקצועית לעולם הבדיקות תוכנה, הם הרגישו חייבים להשתתף בתחרות ISTC. החזון המשותף שלהם הפך לחוד החנית של הבודקים בארץ, תוך קביעת סטנדרט חדש למצוינות בתחום ההייטק, עם שאפתנות בלתי מעורערת להוביל את הדרך המקצועית עבור כלל הבודקים הקיימים והמצטרפים בדרך ארץ, ביושרה, באהבה וקבלת האחר.

דנית דור

נתנאל הרוש

ברכות גם לשאר הצוותים שעלו והתחרו בגמר גביע וזכו בתואר Best Bug Award

ארז לסמן וגיא לוין – צוות ThunderBugs
אריאל לזריאן – צוות Ariel

	סטטיסטיקות המוקדמות	סטטיסטיקות הגמר
משתתפים	35 צוותים – 3 צוותי יחיד ו-32 צוותי זוגות	4 צוותים של זוג וצוות אחד של יחיד
המוצר הנבדק	Fillfaster – מערכת להחתמת מסמכים און ליין בלירוקרטיה – מערכת לניהול מסמכים ובטיחות	New-Ledge – מערכת הדרכות וחדרי בריחה המשלבת משקפי VR עם אפליקציית וב ומובייל
כמות שעות בדיקה	201 שעות אדם	24 שעות אדם
כמות תקלות שדווחו	370	80
כמות ממוצעת של תקלות שנמצאו לצוות	10.5	16
מקסימום של תקלות שנמצאו ע"י צוות אחד	36	21
מינימום של תקלות שנמצאו לצוות אחד	1	4

Word Search

A V I G N I T S E T C I M A N Y D Q Q E G A T
T A E S F L Y Q V Z L Z N X R S A U Z M U M S
L Y L T M N U H K V Y E O P X T D A E V T Z Z
U X I A Y O V B M B C T I G N D D L O I R G T
T B G T T S Z R N T E S T C A S E I I E A T U
P H A I I D Y V Y Y P T A G I Q B T E B C D M
E F E C R Y F F S T Y E C S D J B Y M Y E V I
J B A T U S T P T I Z S I S X T U E Z Y A F X
B R N E C D M I R L O T F Z Y J G E B Y B Z C
Q R R S E I J G V I X O I Q V F I M U B I L H
T C V T S K W K U B S R R I V N N I G Y L R X
S F L I F U H L O A W A E D I G G S D R I R U
I R O N M H Q C R S N C V E E J M V V O T J Z
N R D G H L T K P U Q L S B S L T H U R Y Q E
U N I T T E S T Y T U E L J O N M E Q R Z Q Q
U L V A L I D A T I O N T M F V U M A E P U K

Find the following words in the puzzle.
Words are hidden ↑ ↓ → ← and ↘ .

DYNAMIC TESTING
STATIC TESTING
TRACEABILITY
VERIFICATION
VALIDATION
TEST ORACLE
DEBBUGING
USABILITY
SECURITY
UNIT TEST
TEST CASE
QUALITY
ISTQB
ERROR
AGILE
ITCB
STR
STP
STD
BUG

את הפיתרון יש לשלוח במייל
MAGAZINE@TESTINGWORLD.CO.IL
הראשון שישלח את הפיתרון נכון יפורסם בגליון הבא

ניצן גולדנברג

מזה 7 שנים בתחום בדיקות התוכנה, תפקיד נוכחי מהנדס בדיקות בכיר בחברת SeatGeek, מנהל קבוצת ה-AB של ארגון ITCB® המוביל הראשי של קבוצת המיטאפ TestIL, מרצה בקורסים לבודקי תוכנה

עורכי הגיליון

תמרה מוסונובה

בודקת תוכנה ואינטגרציה בחברת Varonis. בעלת תואר בתקשורת וקולנוע והסמכות פיתוח אפליקציותWeb של מיקרוסופט, כך משלבת חשיבה יצירתית ואנליסטית בעבודה. בונה אתרים ועורכת סרטים כתחביב. שחקנית כדורשת בליגה ארצית, תופסת כדורים ובאגים מקצועית.

אפרת וינברג

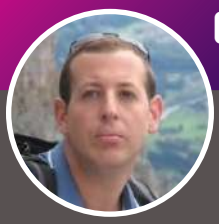
עוסקת בבדיקות תוכנה קרוב ל-20 שנה. עבדה במספר ארגונים בתפקידי בדיקות וניהול בדיקות. בשנים האחרונות עוסקת בפיתוח והוראת קורסים בבדיקות תוכנה ונושאים נוספים

אסתר צבר

מהנדסת (M.Sc.) בעלת 23 שנות ניסיון בפיתוח ובדיקות תוכנה, מתוכן 11 שנים בניהול QA בחברות ECI ו BMC - ובנוסף חברה ב-Advisory Board של ITCB® הארגון הישראלי להסמכת בודקי תוכנה. בתשע השנים האחרונות – יזמית ומנהלת של AQA המכשירה ומשלבת אנשים עם תסמונת אספרגר בעבודה בהייטק כבודקי תוכנה.

משה מאמיה

בעל 17 שנות ניסיון כמהנדס, מתוכן מעל עשר שנות ניסיון ניהולי, מתמחה בפיתוח אוטומציה ובבדיקות ביצועים. עובד מעל 5 שנים ב-HP כמנהל קבוצות QA ו-DevOps. חבר מייעץ למועצת מנהלים של ISTQB® ומרצה בפקולטה להנדסת תוכנה במכללת SCE.

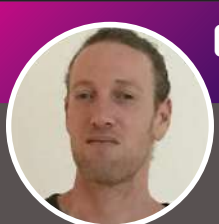


עמית ורטהיימר

בודק תוכנה ב-Deep Instinct.

רחל ברוך

עובדת כבודקת תוכנה בכלל ביטוח. לאחר הפסקה של עשור מעולם התוכנה חזרה למקצוע הכי אהוב אליה. כשעובדים בעבודה שנהנים בה הזמן טס.

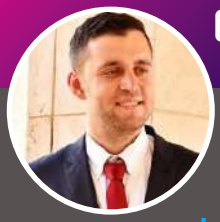


דור רוזנברג

לפני כמה שנים החלטתי לעשות שינוי במקצוע שלי. לאחר מספר תחומי לימוד הכי הרבה התלהבתי מלימודי בדיקות תוכנה. כרגע עובד בשרות לקוחות בישראכרט.

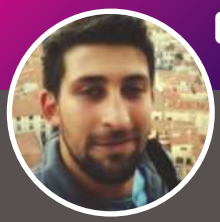
שי ביטון

על 12 שנות ניסיון בפיתוח אוטומציות ובדיקות. עובד כיום ב- Qwilt.



אלכס קומנוב

בעל מספר שנים בתחום הבדיקות האוטומטיות. עובד כמפתח בדיקות ותשתיות אוטומציה בכיר בחברת SeatGeek נשוי+1 חובב נגינה על גיטרה וטיולים.



רוביק סביאנץ

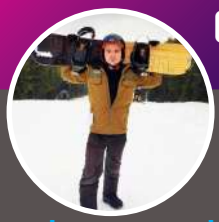
בודק תוכנה, נמצא בתחום מעל 4 שנים את דרכו התחיל בחברת CARAMBOLA נכון להיום מחזיק את מערך הבדיקות בחברת OOLO בזמן הפנוי - ספורט, טיולים ומחשבים

טל פאר

בעל ניסיון של יותר מ-20 שנים כבודק ומנהל בדיקות במגוון חברות וטכנולוגיות במודלי פיתוח שונים. כיום טל יועץ ומדריך בדיקות עצמאי.
טל חבר ב-ITCB® וגזבר הארגון העולמי ISTQB®.

שירה נוסבוים

הייטקיסטית ואמא במשרה מלאה, בדקות הבודדות שנשארות ביום בלוגרית אפייה.
בעלת תואר ראשון במדעי המחשב ובכימיה, מעל 10 שנות ניסיון כמפתחת תשתיות אוטומציה וכלים אוטומטיים בחברות גדולות, בסטארטאפים שונים בתעשייה במגוון תחומים. מובילת תחום, מרצה ומפתחת קורסי אוטומציה. בשבילה החיים זה לא מספיק.



אלכסנדר זבולוקו

אני מפתח תשתיות אוטומציה בחברת SeatGeeK. בנוסף אני מתמחה בדוואופס וגם מכיר טכנולוגיית ענן בעומק. עם רקע רחב בשני התחומים, אני מנצל את הידע והמיומנות שלי כדי לפתור בעיות ולהביא לחדירה מרבית של טכנולוגיות חדשות בסביבת התשתיות. ביכולתי להפוך את התשתיות לגמישות ויעילות יותר באמצעות ענן המאפשר גיוס משאבים והתקנה מהירה של סביבות חדשות. מחבר את המקצועיות שלי עם תשוקתי לטייל ברחבי העולם, אני מתרגש לחוות חוויות ותרבויות חדשות בזמן שאני עובד במיזם שלי.

מגזין עולם הבדיקות