נכתב ע"י: אביה צ'ריקר :)

הנדסת תוכנה שיעור 8

בדיקות תוכנה:

למה לבדוק את התוכנה באופן כללי ?

- נכונות ותקפות – אנחנו רוצים לדעת שהתוכנה עושה את הדבר הנכון ובצורה הנכונה
- אפשר לחסוך כ-שליש מהעלויות אם עושים את הבדיקות הנכונות

איך בודקים את התוכנה?

1. מריץ בעצמי ומדבג
2. נותן לחבר/ה לבדוק (QA)
3. כותב תוכנית הדגמה שמריצה את הקוד שלי

אילו בדיקות ?

- דיבאג (ניפוי שגיאות)
- בדיקות יחידה
- בדיקות עומס, בטיחות, גישוש, שמישות A/B ועוד.

A/B - ללקוחות שונים נותנים מוצר שונה ורוצים לראות את ההתנהלות (גירסא ניסיונית ששולחים לחלק מהאנשים)

- בדיקות אינטגרציה
- בדיקות קצה לקצה
- בדיקות מערכת
- בדיקות קבלה
- בדיקות רגרסיה – לראות שלא דפקנו כלום מהגירסאות הקודמות
- סקרי קוד

נתמקד ב:

- בדיקות מערכת :
  - האם המערכת עובדת בשלמותה?
  - מנקודת מבט של המשתמש! ("קטמנדו")
  - בפרויקט: ניסוח בדיקה באמצעות תרחיש או סיפור
  - (קצה לקצה\משתמש\קבלה\פונקציונליות)
- בדיקות אינטגרציה

– האם הקוד שכתבנו עובד מול קוד אחר

– האם אי אפשר להסתפק בסוג הראשון?

- בדיקות יחידה (מפתח)

– האם המודולים עושים את הדבר הנכון? נוחים לשימוש?ע"י מי?

- בדיקת יחידה היא קוד שקורא לקוד אחר ובודק אח"כ נכונות של טענות מסוימות.
- "יחידה" היא רכיב "קטן" בד"כ מימוש של תרחיש מסוים (מצד המשתמש), מחלקה\פונקציה\מתודה
- System Under Test (SUT) – הדבר שאנו בודקים

באופן כללי – בדיקות יחידה זה שכל פונקציה סגורה על עצמה ועובדת כמו שצריך

? - אם אני עושה בדיקות קצה לקצה למה צריך לעשות אינטגרציה?

בדיקות קצה לקצה קשה לנו שהם יהיו מקיפות, אם יורדים רמה ועושים בדיקות מקיפות בקטן זה יכול לתת תמונה יותר טובה

<u>בדיקות קצה לקצה:</u>

יתרונות:

- מגיעות מקצה לקצה – יודעים שכל המערכת נבדקה החל ממשק המשתמש ועד לdb
- רואים מה שהמשתמש רואה

חסרון:

- קשה לכתוב ולהריץ

? - <u>מה מהבאים אינו יתרון של בדיקות יחידה על בדיקות אינטגרציה וקצה לקצה?</u>

1 .ניתן להריץ בדיקות שביצעתי בעבר שוב ושוב (רגרסיה)

2 .אפשר להריץ במהירות וכך לקבל משוב מהיר

3 .קל לכתוב בדיקה בודדת

4 .אוסף הבדיקות מהווה למעשה מהווה מפרט של המערכת

<u>התשובה:</u> 4

נכתב ע"י: אביה צ'ריקר :)

מי בודק את בדיקות התוכנה?

השאלה אם זו חלק מהעבודה של הפיתוח או שיש אנשים שתפקידם לבדוק את זה

המטרה: מוצר בעל ערך/איכותי

הרבה פעמים יש על מפתח אחד כמה בודקים

מתי כותבים את הבדיקות?

1. **הגישה המסורתית** – בודקים בסוף

ההנחה: המהנדסים כותבים קוד מושלם ולכן בסוף הפרויקט נבדוק רק אם זה תואם את הבדיקות

2. **באג'ייל** – לכתוב את הבדיקות אפילו לפני הקוד AGAIL

תכנון פשוט של קוד:

1. כל הבדיקות עוברות
2. ללא כפילויות
3. ברור - מבטא את כוונת המתכנת
4. קטן - מינימום של מחלקות ומתודות

הגדרות התרגיל שיש בשיעורי בית 4:

1. משכפלים את המאגר דרך ההזמנה
2. עושים קלון למכונה שלנו – ואז עושים את זה מקומי
3. עוברים על המדריך הזה
4. איפה שרואים עיגול אדום כחול ירוק כותבים באיזה שלב אנחנו בתוכנית
   אדום – קוד שנכשל
   ירוק – קוד שעבר
   כחול – ריפרקטורינג (כמעט לא יהיה)
5. דוגמא שיעור הבא