שאלות על האפלקציה

**General**

1. מה המבנה הכללי של האפלקציה? לצייר את עמוד 1 בתרשים
2. מהם ה call backs שיש לכל class במבנה שציירת וכיצד מתבצעת התקשורת של ה host activity איתם? אילו אובייקטים עוברים במקרה של הצלחה או כשלון? ממי יורש כל אחד מה classes ומה המשמעויות של הירושה הזאת?

**Analyzer**

1. צייר תרשים. הסבר אילו אובייקטים עוברים בכל שלב. לצייר את עמוד2
2. באיזה פונקציה משתמשים כדי לעדכן את ה DragRectView לאחר שה Analyzer מזהה את ה Corners ומדוע? מהן הדרכים האחרות של thread לפנות ל ui thread? תכלס איני יודע על עוד דרכים..לקרוא אולי
3. מהן הפעולות שמבצע ה Analyzer ב thread הנפרד ולאיזו פונקציה הוא קורא בסוף?

**Summarizer + UiUpdater + FirebaseUploader**

1. צייר תרשים. לצייר את עמוד 3
2. כיצד פתרנו את בעית ה race שהייתה כש 2 משתמשים בוחרים בו זמנית את אותו הפריט? תאר את הבעיה (כולל דוגמא קונקרטית – למשל ה sum של המוצרים) והסבר את הפתרון
3. תאר את כל הפונקציות שנקראות כשמשתמש מקליק על מוצר (לפי הסדר) והסבר מה תפקידה של כל אחת

**Firebase**

1. צייר את המבנה של ה DB והסבר מה הוא מכיל.
2. כיצד עובד כל ענין ה sesions וה logs? ואיך משתמש יודע לכתוב ל path הנכון? או לחילופין לאיפה למשוך את התמונות של ה products?
3. אילו פונקציות (interface) צריכה לממש הפונקציה StartCamera שמשתמשת ב PassCodeResolver?
4. צייר את המבנה של ה Storage ןהסבר מה הוא מכיל? כיצד יודע המשתמש המשני לגשת ל path הנכון לשליפת התמונות של ה product?

**Firebase – Sessions**

1. כיצד עובד כל העניין של ה sessions ואיזו בעיה הוא בא לפתור? מה ההבדל בין משתמש ראשי למשני?
2. מה המבנה של התיקיות מבחינת תאריכים?

**Passcode**

1. למה הוא נוצר וכיצד? מה ה interface שצריך לממש כדי להעלות אותו ל Firebase עבור יתר המשתמשים?

**Tests**

1. מה כוללים הטסטים שלכם? כיצד בניתם את מערך הטסטים כך שהוא יהווה בדיקה נכונה עבור כל הקומפוננטות של האפליקציה?
2. כיצד יכולתם לשפר משמעותית את מערך הטסטים? להשתמש ב Dependency Injection לאורך כל כתיבת האפלקציה. בנוסף למשל לבודד את כל התקשורת עם ה firebase ל interface נפרד כך שנוכל להחליף אותו במידת הצורך
3. מהי בעיית ה producer-consumer וכיצד ה ConcurrentLinkedQueue פותר אותה? לקרוא בוויקיפדיה על בעיית היצרן-צרכן ולקרוא בדוקומנטציה של גבה על המבנה נתונים הזה (thread safe queue based on linked nodes. Also it FIFO)
4. מה זה thread pool ולמה הוא משמש?

**החלפת tess two ב ABBYY**

1. תאר את כל התהליך שעברתם לאורך הפיתוח עם ה ocr? באילו SDKs השתמשתם ולמה חשבתם שאחוזי הזיהוי יהיו גבוהים?

**הכל על DL**

1. איפה ואיך השתמשתם ב DL באפלקציה? כיצד עובדת הארכיטקטורה של tflite וכיצד מתבצעת התקשורת עם ה model? מהו האלגוריתם שהשתמשתם בו כדי לזהות את המסגרת של החשבונית בתמונה שחזרה מהמודל?
2. (**שאלה לפי האיור על Deep Learning בתרשים**) מהי הארכיטקטורה של TF Lite? תאר את כל השלבים החל ב trained model ועד הריצה של ה kernels על המכשיר

**Camera (billslib)**

1. אילו interfaces מממש ה class הזה ולמה?
2. מדוע ממשתם custom camera ולא השתמשתם במשהו קיים? כיצד עזרה לכם הפונקציונאלית הספיציפית הזאת?

**Contracts (billslib)**

1. מדוע כדאי להגדיר interfaces לפני שמתחילים לכתוב?
2. כיצד העתקתם למכשיר את הקבצים הרלוונטיים?

**Core (billslib)**

**BillAreaDetector:**

1. כיצד GetBillCorner זיהה את החשבונית? לפני שהשתמשתם ב HED

**BillsLog:**

1. איזה מיפוי ביצענו ומה המטרה שלו? כיצד נכנס כאן עניין ה sessions לתמונה?

**ImageProcessingLib:**

1. כיצד עובד ה WarpPerspective ומה הטריק שעשינו בחישוב כדי לא לעוות את הפונט?
2. תאר מה מבצעים PreprocessingForTM/PreprocessingForParsing ואיך השימוש הנכון בפילטרים משפיע על התוצאות? מה הפונקציות של OpenCV בהם השתמשתם?

**MainActivityBase:**

1. מה המטרה שלו? אילו פעולות הוא מבצע?

**TemplateMatcher:**

1. תאר באופן כללי את התהליך של ה Match וה Parsing שאחריו
2. הסבר כיצד עובד ה Match
3. הסבר כיצד עובד ה Parsing
4. כיצד נדע איזו עמודה היא ה Price ואיזו ה Quantity

**TesseractOCREngine:**

1. מהו ה interface שמממשים ולמה חשוב להגדיר interface?

**Views**

**DragRectView:**

1. מה ה View בו השתמשת ב DragRectView וכיצד אתה מצייר את ה edges של החשבונית עליו?

**fragment\_bill\_summarizer.xml:**

1. תאר באופן כללי את ה view הזה

**Exeception Handling**

1. כיצד מתנהל כל ענין הלוגים שלכם? לאן אתם כותבים אותם? מה שם הפוקנציה ששולחת את הלוגים למייל במקרה של קריסה

**Utilities**

1. תן דוגמאות לכלים שהשתמשתם בהם לכל אורך האפלקציה והצריכו את יצירת ה class הזה

**שאלות (במקור נכתבו תוך כדי הסיכום)**

1. מה ההבדל בין extends Thread לבין implements Runnable?
2. מה ההבדל בין synchronized(this) ל synchronized(someLock)?
3. מהם כל השלבים של קריאה ל transaction ומה בעצם קורה בכל אחד מהם?
4. מה ההבדל בין fragment ל activity ולמה עדיף להשתמש ב fragment?
5. להבין את ההשוואה הבאה: LinearLayout vs RelativeLayout vs ConstraintLayout vs FrameLayout. מתי כדאי להשתמש בכל אחד?
6. איך עובד ה progress bar? (**שאלה לשינויים עתידיים אם נרצה**)
7. בפונקציה StartCamera ב MainActivity מדוע לא מחלחלים את ה relative path ל camera לצורך כתיבת לוגים? המצלמה בעצם לא מתעדת כשלון.. האם אמורים לאתחל זאת כשמייצרים אובייקט של CameraFragment? כמו שמעבירים ל Analyzer (**שאלה לשינויים עתידיים אם נרצה**)

**תשובות (במקור נכתבו תוך כדי הסיכום)**

1. כשמממשים את ה interface של runnable בעצם מתחייבים לממש פונקציה אחת שנקראת run שתרוץ ב thread נפרד. כשעושים extends ל runnable כמובן חייבים לממש גם את אותה פונקציה אבל ניתן לעשות override לעוד פונקציות כדי לשנות התנהגויות נוספות של ה thread. כמובן שנעדיף לממש runnable כיוון שג’בה לא תומכת בירושה מרובה ונרצה לשמור על האופציה של לרשת מ class אחר. מה גם שלרוב לא נרצה לשנות התנהגויות נוספות של ה thread.
2. בסנכרון עצמי כל thread אחר שמשתמש באותו עותק של class (אותו אובייקט) לא יוכל לבצע את הקטע קוד הזה(וגם את כל הקטעי קוד האחרים שמשתמשים ב synchronized(this)) עד שה thread הנוכחי ישחרר אותו. כשמסנכרנים על מנעול מסוים אז רק thread שינסה לגשת לקטע קוד אחר שמשתמש באותו מנעול, יצטרך לחכות.
3. אפשר לקרוא כאן:  
   <https://stackoverflow.com/questions/29782808/what-does-fragmentmanager-and-fragmenttransaction-exactly-do>
4. הבדל 1: בפריסה על המסך. אפשר להשתמש ב 2 fragments כדי להציג בו זמנית RecyclerView למשל ואת ההקלקות על שורה כלשהי. לא ניתן להציג בו זמנית 2 Activities בגלל ההגבלות של מערכת ההפעלה אנדרואיד

הבדל 2: הבדל קריטי בין ה lifecycle של activity לבין של fragment הוא שה lifecycle callbacks נקראים ע”י ה hosting activity ולא ע”י ה OS. כמו שנאמר בספר:

**The OS knows nothing about the fragments that an activity is using to manage things. Fragments are the activity’s internal business.**   
הבדל 3: fragment חייב להיות שייך ל activity מסויים ולא יכול להיות עצמאי.

הבדל 4: fragments נוספו רק ב api 11 לעומת activities שהיו קיימים תמיד.

הבדל 5: תקשורת בין activities די יקרה וצורכת משאבים (יצירה של activity זה דבר יקר. התקשורת בהכרח תתבצע ע”י bundles שהוא יכתב ל activity record). יצירה ותקשורת של fragments די פשוטה. התקשורת תתבצע ע"י callbacks.

1. ההבדלים הם:

With **ConstraintLayout** , instead of using nested layouts you add a series of constraints to your layout. A constraint is like a rubber band. It pulls two things toward each other. So that is the big picture: To place views where you want them to go in a ConstraintLayout , you give them constraints instead of dragging them around the screen.  
What about sizing widgets? For that, you have three options: Let the widget decide (your old friend wrap\_content ), decide for yourself, or let your widget expand to fit your constraints.

**FrameLayout** is the simplest ViewGroup and does not arrange its children in any

particular manner. In this layout, child views will be arranged according to their android:layout\_gravity attributes.

You use a **LinearLayout** when you want widgets arranged in a single column or row.