מסמך תיעוד –Handy

מערכת השאלת חפצים

מגישים:

יניב קרים 205731193

אביב בדיאן 313550550

אוקטובר 2019

**תוכן עניינים:**

תיאור המערכת והפונקציונליות 3

טכנולוגיות 4

מבני נתונים 5

ארכיטקטורה 6

# **תיאור המערכת והפונקציונליות**

זוהי מערכת להשאלת חפצים מחנות (שם החנות Handy) אשר מציעה חפצים להשאלה. ישנם שני סוגים של משתמשים במערכת, לקוח ומנהל. לקוח אינו דורש משתמש וצריך רק להוריד את האפליקציה, מנהל לעומת זאת צריך להזדהות עם שם משתמש וסיסמה על מנת לגשת לאפשרויות הפתוחות בפני מנהל

**לקוח:**

כאשר לקוח פותח את האפליקציה הוא רואה את קטלוג הפריטים אשר קיימים להשאלה בחנות, במידה והוא רואה פריט שהוא מעוניין להשאיל הוא בוחר בו ועובר למסך הבא שבו הוא בוחר מאיזה סניף לעשות את הפריט ואז הוא צריך לבחור תאריך שבו המוצר זמין. המוצר זמין אם מנהל הזין עבורו זמינות בטווח תאריכים קבוע מראש.

לאחר שהלקוח בוחר פריט, סניף ותאריכים הוא ממלא את הפרטים שלו (שם, מייל ומספר פלאפון) וממתין שמנהל יאשר לו את ההזמנה, כאשר מנהל מאשר הזמנה האפליקציה שולחת ללקוח סמס שההזמנה אושרה.

**מנהל:**

לאחר שמנהל מתחבר ועובר למסך הניהול יש לו מספר יכולות

1. **הוספת/מחיקת חפצים:** מנהל יכול להוסיף חפץ חדש למערכת ולהעלות אותו עם תמונה שתשמר בשרת. כמו כן הוא יכול להסיר חפץ מהמערכת אם הוא לא זמין יותר להשאלה
2. **הוספת/הסרת זמינויות של חפצים:** לכל חפץ יש תאריכים בהם הוא זמין. זוהי רשימה של טווחי תאריכים בהם ניתן להשאיל את החפץ (אם למשל יש אותו חפץ מספר פעמים ניתן להזין זמינויות חופפות כדי שכמה אנשים יוכלו להשאיל את החפץ בו זמנית). למנהל יש אפשרות עבור כל חפץ להוסיף ולהסיר זמינויות
3. **אישור הזמנות של לקוחות:** כאשר לקוח מבקש להשאיל חפץ, מנהל צריך לאשר לו את ההזמנה. במסך אישור הזמנות המנהל רואה את כל הבקשות הפתוחות והוא יכול לאשר ולדחות אותן בהתאם להחלטתו. במידה ונהל בוחר לאשר/לדחות הזמנה הלקוח שביקש את ההזמנה מקבל סמס שמתריעה על כך

# **טכנולוגיות**

הטכנולוגיות שנבחרו לכל חלק במערכת הן:

1. **בסיס נתונים:** נבחר בסיס הנתונים הטבלאי PostgreSQL
2. **צד שרת:**
   1. גישה לבסיס הנתונים: דרייבר JDBC שמתחבר לPostgreSQL
   2. תקשורת שרת לקוח: שרת Spring אשר חושף ממשק REST כך שהלקוח יכול לפנות בHTTP לבקשות השונות
   3. קונפיגורציה: ההגדרות השונות של השרת נשמרות בקובץ XML אשר נטען. לדוגמה כתובת בסיס הנתונים ופרטי ההתחברות
3. **צד לקוח:** קליינט אנדרואיד
   1. שמירת תמונות: תשתית **Firebase-storage** של גוגל אשר מספקת שירותי אחסון.
   2. טעינה ותצוגת תמונות: ספריית **Glide** אשר מספקת יכולות טעינה ו-Caching של תמונות.
   3. גישה לשרת: ספריית **Retrofit** אשר מספקת שירותי Http client וממפה API Rest לממשק Java
   4. תשתית פיתוח אנדרואיד: **AndroidX** ספריות הרחבה עבור ספריות התשתית הישנות של אנדרואיד מעל Java.
   5. ניהול ממשק תאריכים: **savvisingh/DateRangePicker**- תצוגת Calendar בעלת תצוגה משופרת ויכולות מורחבות.
   6. הזרקת תלויות אובייקטים תצוגתיים: **Butterknife** – הזרקת אובייקטים ויזואליים לאובייקטים מאחורי הקלעים בצורה נוחה.
   7. ואלידציות: **saripaar** – מאפשרת ואלידציה אינטואיטיבית עבור שדות טפסים.
   8. העברת נתונים: **gson** – ספריית Serialize/Deserialize של אובייקטי Java ל-Json ולהיפך.

# **מבני נתונים**

מבני הנתונים במערכת כפי שהם נשמרים בבסיס הנתונים, בשרת ובלקוח (מפתחות ראשיים **בבולד וקו תחתון**, מפתחות זרים בבולד**)**:

1. Admins(**username,** password) – פרטי המנהלים השונים במערכת
2. Items(**id,** title, category, description, path) - החפצים השונים שניתנים להשאלה במערכת, יחד עם הקטגוריה, התיאור והנתיב לתמונה שלהם
3. Branches(**title**, address) – הסניפים השונים מהם ניתן לאסוף את החפצים והכתובת שלהם
4. Availabilities(**id**, **item\_id**, start\_date, end\_date) – זמינות עבור חפץ כלשהו בטווח תאריכים מסויימים
5. Borrows(**id**, **availability**, **branch**, phone, email, first\_name, last\_name, status)בקשות השאלה לזמינות מסוימת (של חפץ מסוים) בסניף מסוים יחד עם פרטי המבקש וסטטוס הבקשה, כאשר סטטוס יכול להיות
   1. Pending - ממתין לאישור או דחייה של מנהל
   2. Approved – בקשה מאושרת
   3. Declined – בקשה נדחתה

# **ארכיטקטורה**

1. Server side:
   1. Application: Main class that starts the spring application which start all the REST controllers
   2. AdminController: A controller which exposes REST api to log in to management
   3. AvailabilityController: A controller which exposes REST api to read, create and delete availabilities
   4. BorrowsController: A controller which exposes REST api to read, create and change the status of borrow requests
   5. BranchesController: A controller which exposes REST api to read, create and delete branches
   6. ItemsController: A controller which exposes REST api to read, create and delete items
   7. DataStore: A class which is responsible to create the connection to the database and make queries.
   8. DBConfig: A class that reads the system config XML in order to expose the connection data to the database
   9. ViewModels: A package that contains all the view models used in the systems
2. Client Side:

# **מדריך לשימוש באפליקציה**

## מנהל המערכת