**מסמך תכנון המתאר את מבנה האפליקציה:**

Baby-d הינה אפליקציית אנדרואיד המאפשרת להורה חדש לעקוב אחרי סדר יומו של התינוק.

המערכת כוללת קבצי JAVA של השרת ושל האנדרואיד וגם קבצי XML להגדרה גרפית של מסכי האפליקציה.

במערכת 3 רבדים:

1. **הלקוח** שבפניו חשופים מסכי האפליקציה ומאפשרים לו לתקשר עם השרת ומסד הנתונים.
2. **השרת** מכיל את האפשרות להבדיל בין הורים שונים וכן נותן מענה לתקשורת בין ההורה לנתונים שהוא מכניס למסד נתונים. השרת כתוב בטכנולוגית spring-boot
3. **מסד הנתונים** משתמש בטכנולוגיית postgressql הוא אוחז בתוכו את כל המידע שההורה מכניס על תינוקו.

**השרת:**

קבצי ה-controller:

בשרת ישנם שני קבצים שמשמשים לחשיפת ה-REST API לעולם החיצון (משמע האפליקציה).

קובץ ParentController:

מכיל בתוכו את השיטות הבאות:

1. loginParnt- משמש עבור הרשמת הורה. השיטה תחזיר JWT (ג'אווה תוקן) שנועד לשמש את הלקוח עבור אפשרות להשתמש באפליקציה.

2. registerParent- משמש עבור הורה להירשם לשירותי האפליקציה על ההורה לתת נתונים: שם פרטי, שם משפחה, אימייל, טלפון, וסיסמא. כל הנתונים האלו יוכנסו לתוך מאגר הנתונים.

3. getAllParents- עבור צרכי דיבאג התוכנית ההורה לא יוכל להשתמש בזה רק האדמין של התוכנית יוכל.

4. deleteAllParents- ישמש גם רק לצרכי טסט כאשר נרצה לנקות את המסד נתונים שלנו.

קובץ BabyController:

מכיל את השיטות הבאות:

1. fetchAllBabies- לצרכי בדיקת התוכנית. שיטה זו תביא את כל התינוקות הקיימים בתוכנית.
2. removeBaby- כרגע לא בשימוש יהיה ניתן לממש בעתיד מחיקת תינוק.
3. addBaby- הוספת תינוק ושיוכו להורה שיטה זו תקבל שם התינוק, שם משפחת התינוק, סוג האכלה, תאריך לידה, ומשקל התינוק.
4. findById- על מנת שנוכל להביא תינוק שמשויך להורה מסוים נצטרך את המתודה הזו היא תביא.
5. getBabyFullInfo- שיטה זו בעצם תביא את כל המידע על תינוק מסוים עבור תאריך מסוים.
6. setWeight- שיטה זו תכניס משקל תינוק חדש ותעדכן את מסד הנתונים.
7. setBabyDipper- שיטה זו תכניס מידע עבור החלפת חיתול לתינוק.
8. setBabyFormula- שיטה זו תכניס למסד הנתונים את נתוני האכלת תמ"ל לתינוק.
9. setBabyBreast- שיטה זו תכניס למסד הנתונים את נתוני האכלת מיניקה לתינוק.
10. setBabySleep- שיטה זו תכניס למסד הנתונים את נתוני השינה של התינוק.

שליחת מייל ללקוח ((MailServiceImpl:

שליחת מייל ללקוח ממשמשת את ממשק MailService במחלקה זו יש שלוש מתודות:

1. sendMailRegisterSuccessfully שליחת מייל עבור לקוח שסיים את ההרשמה לאפליקציה.
2. sendMailBabyAdded- שליחת מייל עבור הוספת תינוק להורה מסוים.
3. send- השיטה שמממשת את שליחת המייל.

מחלקות של שגיאות (exceptions):

1. EtAuthException- מעטפת עבור שגיאת זיהוי.
2. EtBadBirthdayFormat- מעטפת עבור שגיאת בקשה לא נכונה/פורמט שגוי תייצג חריגה רק אם ניתן תאריך בפורמט לא נכון.
3. EtResourceFoundException- מעטפת עבור שגיאת בקשה לא נכונה, תשמש אם היה שגיאה בהכנסת נתונים לתוך המסד נתונים.
4. EtResourceNotFoundException- מעטפת עבור שגיאת שרת, יופיע במידה ומשהו שחיפשנו במסד הנתונים לא קיים.
5. EtUnableConnectToDB- מעפטת לשגיאת שרת יופיע אם השרת לא הצליח להתחבר למסד הנתונים.

פילטר:

הפילטר משמש את השרת עבור סינון של לקוחות הסינון של הלקוחות מבוצע רק על URI מסוימים בשרת.

זיהוי הלקוחות נעשה באמצעות JWT.

מחלקות ה-repositories:

אחראיות על הכנסת הנתונים שמגיעים מהשרת אל תוך המסד נתונים וגם יצירת טבלאות חדשות במידת הצורך.

ישנם שתי מחלקות עבור כל אחד מהקונטרולרים.

1. ParentRepositoryImpl- יממש שיטות עבור כל אחת מהשאילתות הבאות:

private static final String SQL\_INSERT\_TO\_PARENT = "insert into parent (parent\_id, first\_name, last\_name, email, phone, password) " +

"values (?, ?, ?, ?, ?, ?)";

private static final String SQL\_GET\_ALL\_PARENTS = "select \*from parent";

private static final String SQL\_COUNT\_BY\_EMAIL = "select count(\*) from parent where email=?";

private static final String SQL\_FIND\_BY\_PARENT\_ID = "select parent\_id, first\_name, last\_name, email, phone, password from parent where parent\_id=?";

private static final String SQL\_FIND\_BY\_PARENT\_EMAIL = "select parent\_id, first\_name, last\_name, email, phone, password from parent where email=?";

private static final String SQL\_DELETE\_ALL\_PARENTS = "delete from parent";

1. BabyRepositoryImpl- יממש שיטות עבור כל השאילתות האלו:

// Insert SQL into baby table

private static final String SQL\_INSERT\_TO\_BABY = "insert into baby (" +

"baby\_id, first\_name, last\_name, birth\_day, food\_type)" +

"values(?, ?, ?, ?, ?)";

private static final String SQL\_INSERT\_TO\_PARENT\_BABY\_TABLE = "insert into " +

"parent\_baby\_relation (parent\_id, baby\_id) " +

"values(?,?)";

private static final String SQL\_GET\_ALL\_BABIES\_FOR\_PARENT\_ID = "" +

"select \* " +

"from baby " +

"where baby\_id in (select baby\_id from parent\_baby\_relation where parent\_id=?)";

private static final String SQL\_DELETE\_BABY\_FORM\_BABY\_TABLE = "delete from baby where baby\_id=?;";

private static final String SQL\_DELETE\_BABY\_FROM\_BABY\_PARENT\_RELATION = "delete from parent\_baby\_relation where baby\_id=?";

private static final String SQL\_FIND\_BABY\_BY\_ID = "select \* " +

"from baby " +

"where baby\_id in " +

"(select baby\_id from parent\_baby\_relation where parent\_id=? and baby\_id=?)";

private static final String SQL\_BABY\_WEIGHT\_TABLE\_CREATE = "create table %s ("+

"row\_num integer not null," +

"weight\_date varchar(50) primary key not null," +

"weight double precision not null)";

private static final String SQL\_GET\_BABY\_WEIGHT = "SELECT \* FROM %s ORDER BY row\_num DESC LIMIT 1";

private static final String SQL\_INSERT\_INTO\_WEIGHT\_BABY\_TABLE = "insert into %s (row\_num, weight\_date, weight) values (NEXTVAL('%s'), ?, ?)";

private static final String SQL\_CREATE\_TABLE\_BABY\_FULL\_INFO = "create table %s (" +

"date\_measure varchar(50) not null," +

"time\_measure varchar(50) not null," +

"baby\_weight double precision not null," +

"wet\_diaper boolean," +

"dirty\_diaper boolean," +

"feed\_amount integer," +

"breast\_side varchar(50)," +

"breast\_feeding\_time\_length integer," +

"sleeping\_time integer," +

"feed\_type varchar(50)," +

"CONSTRAINT KEY\_%s primary key (date\_measure, time\_measure)" +

");";

private static final String SQL\_INSERT\_BABY\_INFO\_TABLE = "insert into %s (date\_measure, time\_measure, baby\_weight, wet\_diaper," +

"dirty\_diaper, feed\_amount, breast\_side, breast\_feeding\_time\_length, sleeping\_time, feed\_type) " +

"values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";

private static final String SQL\_GET\_BABY\_INFO = "select \* from %s where date\_measure=?";

מחלקות services:

אחראיות לתקינות המידע לפני העברתו לאחת ממחלקות repository.

תאור מסד הנתונים:

מסד הנתונים מכיל את הטבלאות הבאות:

1. Parent- טבלה הכוללת מידע עבור ההורה:

עמודות:

* parent\_id- uuid (primary key)
* first\_name- string
* last\_name- string
* email- string
* phone- string
* password- encryption string

1. baby- טבלה הכוללת מידע על התינוק:

עמודות:

* Baby\_id – uuid
* First\_name- string
* Last\_name- string
* Birth\_day- string
* Food\_type- int

1. parent\_baby\_relation- טבלה המקשרת הורה לתינוק:

עמודות:

* parnt\_id-uuid
* baby\_id-uuid

1. baby\_<partial\_uuid>\_full\_info- מייצגת את המידע כולו עבור תינוק מסוים.

עמודות:

* Column | Type | Collation | Nullable | Default
* ----------------------------+-----------------------+-----------+----------+---------
* date\_measure | character varying(50) | | not null |
* time\_measure | character varying(50) | | not null |
* baby\_weight | double precision | | not null |
* wet\_diaper | boolean | | |
* dirty\_diaper | boolean | | |
* feed\_amount | integer | | |
* breast\_side | character varying(50) | | |
* breast\_feeding\_time\_length | integer | | |
* sleeping\_time | integer | | |
* feed\_type | character varying(50) | | |

1. baby\_<partial\_uuid>\_weight- מייצג את השינוי במשקל התינוק.

עמודות:

Column | Type | Collation | Nullable | Default

-------------+-----------------------+-----------+----------+---------

row\_num | integer | | not null |

weight\_date | character varying(50) | | not null |

weight | double precision | | not null |

Indexes:

"baby\_\_<partial\_uuid>\_weight\_pkey" PRIMARY KEY, btree (weight\_date)