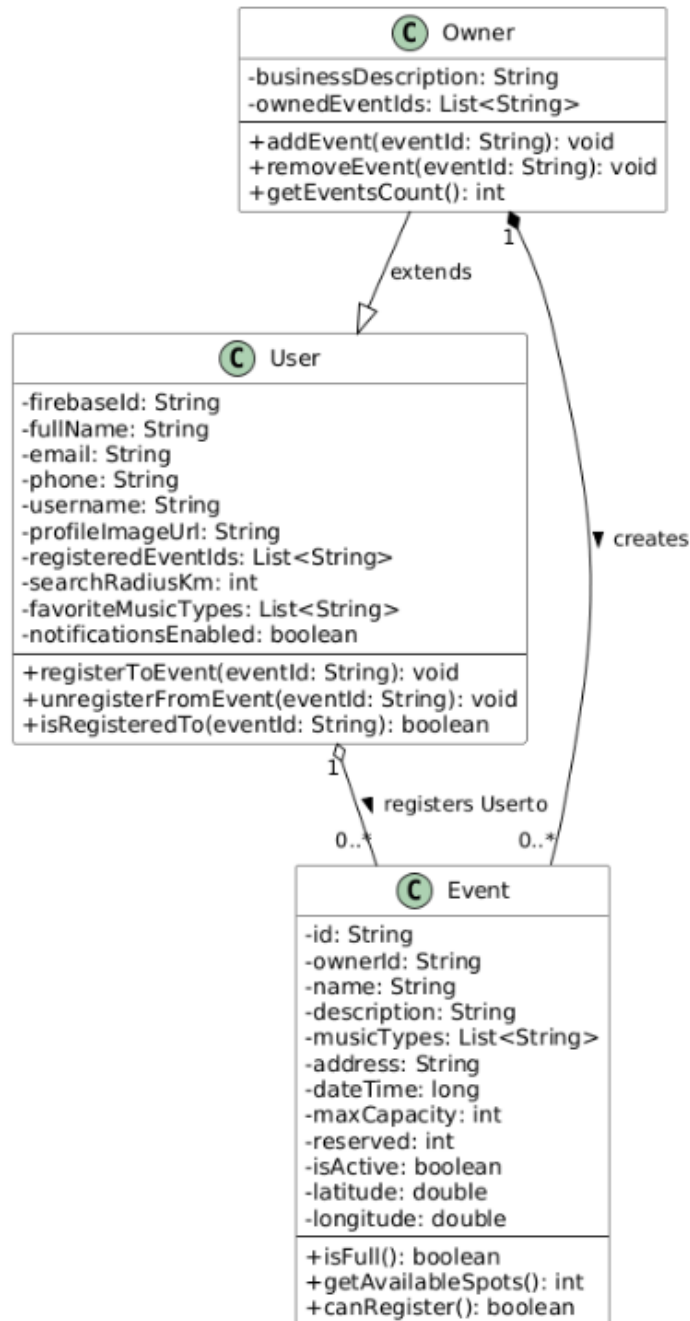


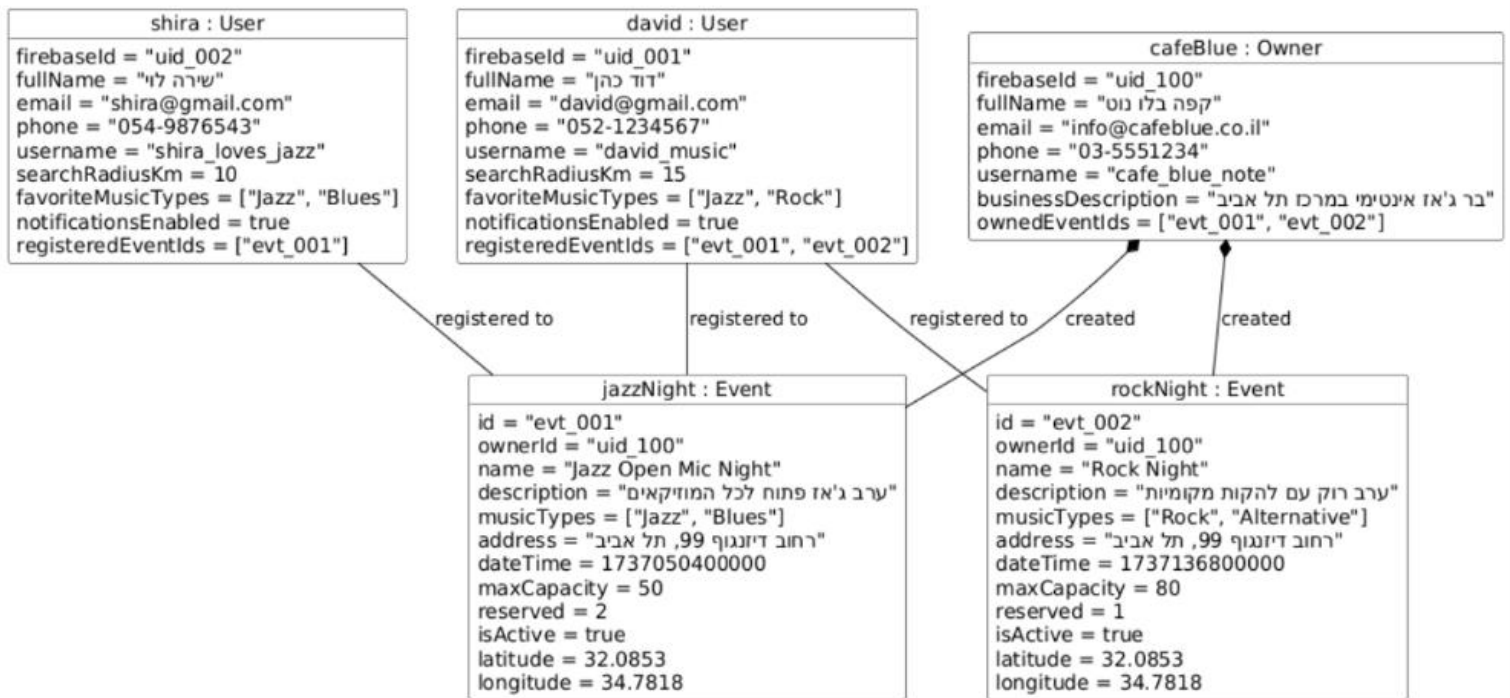
חברי הצוות: רעות גרבר, אור דרעי, שירת הוטרר.

מסמך ניתוח - דיאגרמות:

: Class Diagram.1

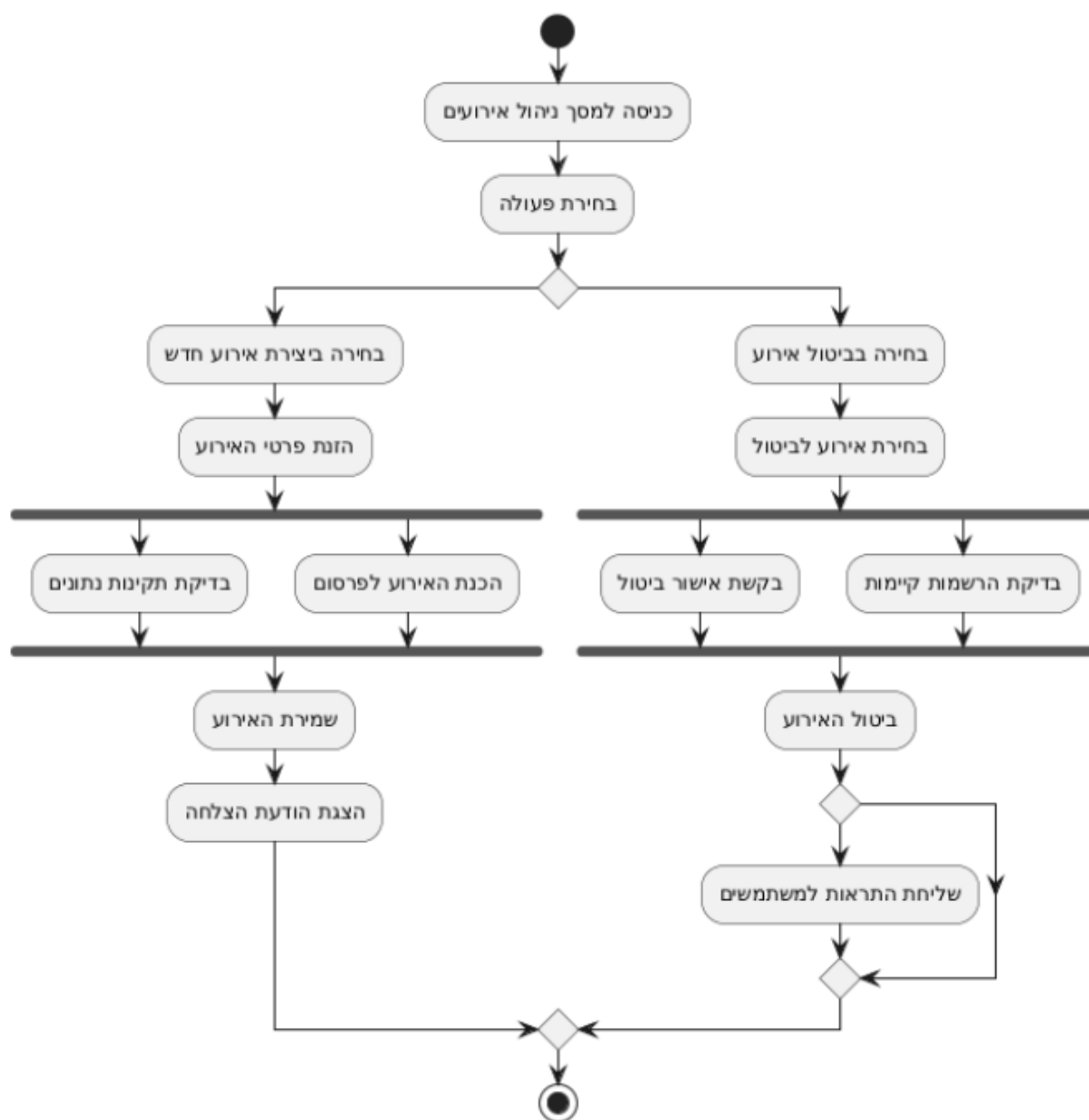


:Object Diagram .2

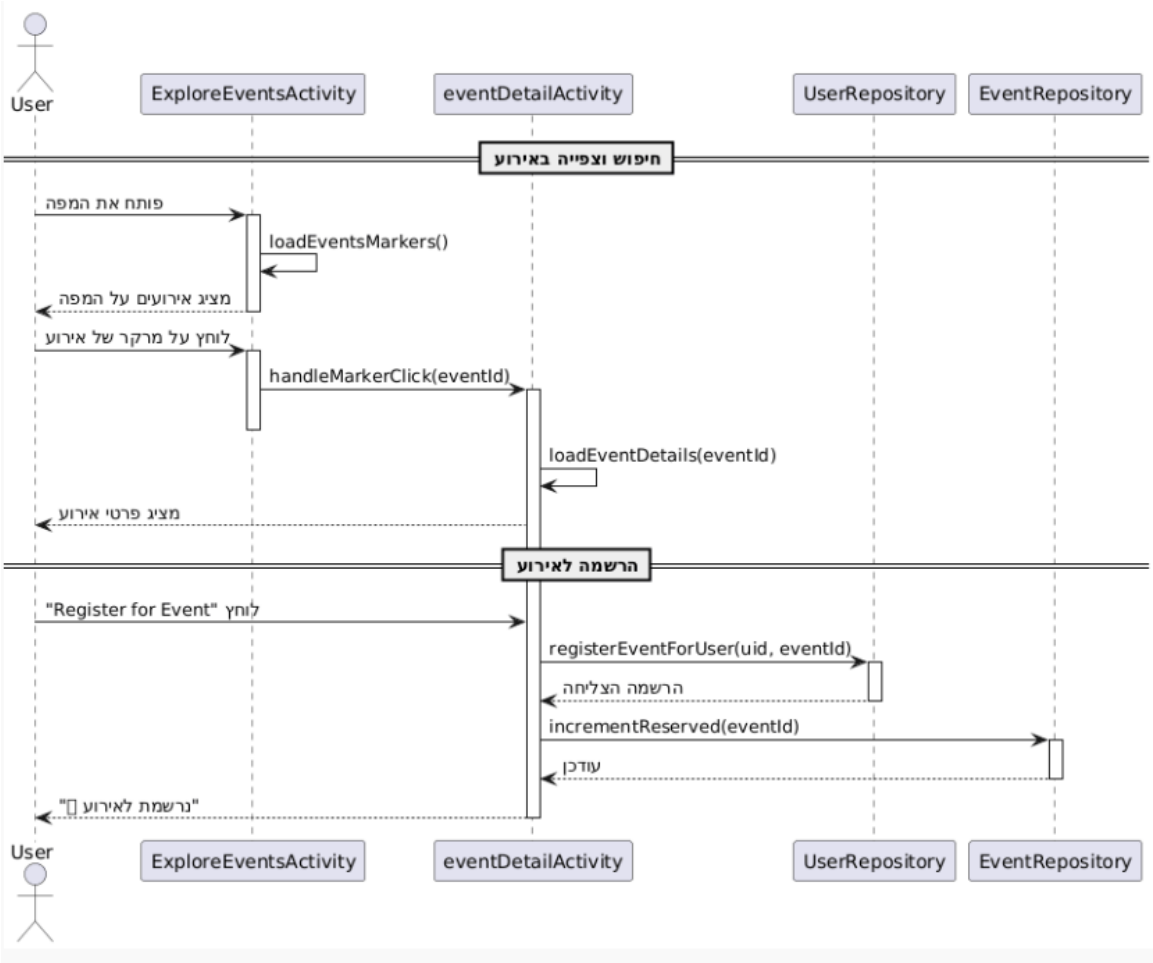


:Activity Diagram.3

הדיאגרמה מתארת את תהליך ניהול האירועים על ידי בעל עסק, הכולל יצירת אירוע חדש או ביטול אירוע קיים.

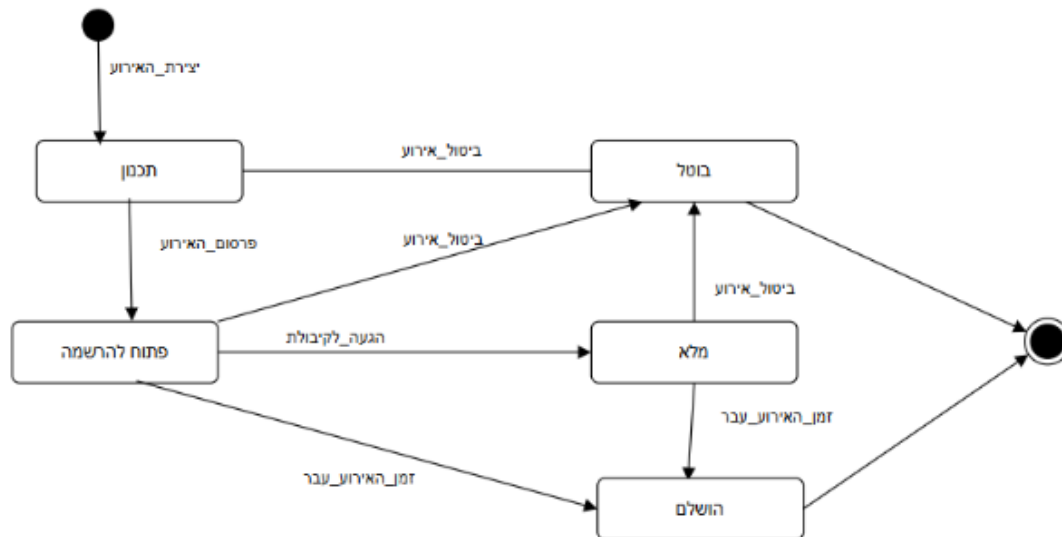


.Sequence Diagram.4

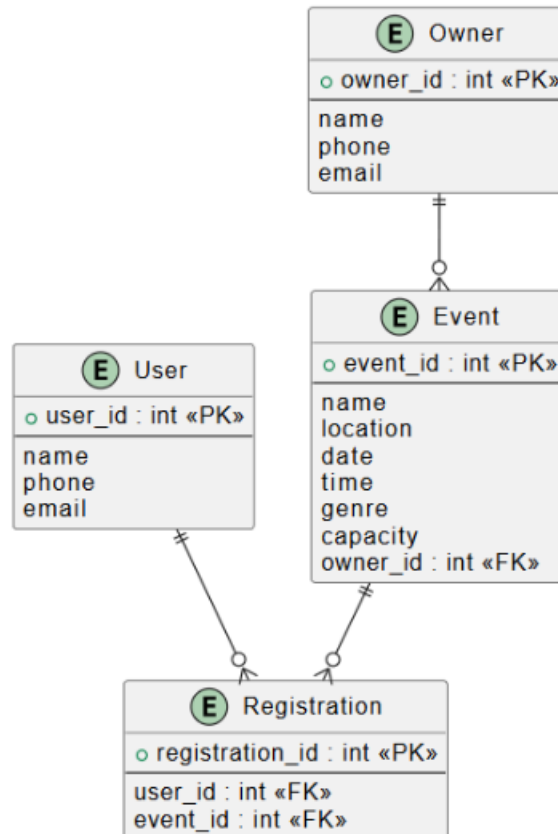


: State Machine Diagram.5

תיאור של אירוע (Event)



:ERD – Entity Relationship Diagram .6



הדיאגרמה מציגה את מבנה בסיס הנתונים של מערכת לניהול אירועים, המנומל לרמה NF3.

המערכת כוללת ארבע ישויות עיקריות:

User - מייצגת משתמש רגיל במערכת. הטבלה שומרת פרטי זיהוי בסיסיים כגון שם, טלפון ודוא"ל, ולכל משתמש מזהה ייחודי (user_id).

Owner - מייצגת מנהל אירועים. למרות הדמיון במאפייני הזיהוי ל-User, מדובר בישות נפרדת עם תפקיד שונה במערכת. לכל Owner מזהה ייחודי (owner_id).

Event - מייצגת אירוע במערכת וכוללת פרטים כגון שם, מיקום, תאריך, שעה, ז'אנר וקיבולת. השדה owner_id הוא מפתח זר המצביע על בעל העסק שיצר את האירוע.

Registration - מייצגת הרשמה של משתמש לאירוע. טבלה זו משמשת כטבלת קשר בין User ל-Event. קשרים בין הטבלאות:

Owner one-to-many Event - בעל עסק אחד יכול ליצור מספר אירועים, ולכל אירוע יש בעל עסק אחד.

User one-to-many Registration - משתמש יכול להירשם למספר אירועים.

Event one-to-many Registration - לאירוע יכולים להיות מספר נרשמים.

אין קשר ישיר בין User ל-Event, כל הרשמה מתבצעת דרך טבלת Registration.

רמות נרמול:

1NF כל הטבלאות מכילות שדות אטומיים בלבד, אין שדות המכילים רשימות או ערכים מרובים, ולכל טבלה מפתח ראשי המזהה באופן חד משמעי כל רשומה.

2NF כל שדה שאינו מפתח תלוי ישירות במפתח הראשי של הטבלה, ולכן אין תלות חלקית בין שדות.

3NF אין תלות טרנזיטיבית בין שדות שאינם מפתחות, כל שדה שאינו מפתח תלוי ישירות במפתח הראשי של הטבלה שלו בלבד, והמידע על משתמשים, בעלי עסקים ואירועים מופרד לישויות נפרדות.

system screens structure

רשימת מסכים מלאה:

מסכים למשתמש רגיל (User):

Login Screen, Register Screen, Home Screen, Event Details Screen, Profile Screen, My Events Screen, Notifications Screen.

מסכים למנהל אירועים (Owner):

Owner Home, Create Event Screen, Edit Event Screen, Owner Notifications Screen, Update profile screen.

פירוט על תפקיד כל מסך:

מסכי משתמש רגיל:

1. Login Page

עמוד התחברות שבו המשתמש מזין אימייל וסיסמה, עם אפשרות לעבור להרשמה או לשחזור סיסמה.

Register Page .2

עמוד יצירת חשבון חדש, שבו המשתמש מזין פרטים בסיסיים כמו שם, אימייל וסיסמה כדי לפתוח משתמש במערכת.

Home Page.3

עמוד הבית המציג רשימת אירועים זמינים. מכאן המשתמש יכול לעיין באירועים, לפתוח פרטי אירוע, לבצע חיפוש ולהפעיל סינונים.

Event Details Page.4

עמוד המציג פרטים מלאים על אירוע מסוים - תיאור, מיקום, תאריך, מושבים פנויים ועוד. מאפשר להירשם לאירוע או לבטל הרשמה.

User Profile Page.5

עמוד פרופיל אישי שבו המשתמש יכול לערוך את פרטיו, לעדכן רדיוס חיפוש מועדף, ולבחור ז'אנרים מועדפים.

My Event Page.6

עמוד המציג למשתמש את כל האירועים אליהם הוא נרשם. מאפשר לפתוח פרטי אירוע או לבטל הרשמה.

Notification Page .7

עמוד המציג התראות שהמערכת שלחה למשתמש: ביטול אירוע, שינוי בפרטי אירוע, סולד אאוט, מקום שהתפנה וכדומה.

מסכי בעל אירוע:

Owner Dashboard Page .8

עמוד ניהול עבור בעל האירוע, המציג את כל האירועים שיצר ומאפשר לעבור לניהול האירועים השונים.

Create Event Page .9

עמוד יצירת אירוע חדש - כולל טופס להזנת שם האירוע, תיאור, תאריך, שעה, מיקום וכמות מושבים.

Edit Event Page .10

עמוד לעריכת אירוע קיים, המאפשר לעדכן פרטים או לבטל את האירוע לחלוטין.

Owner Notifications Page .11

עמוד המציג לבעל האירוע התראות כגון: סולד אאוט, מושבים שהתפנו, שינויים הקשורים לנרשמים ועוד.

Update profile page.12

עמוד המציג את פרטי מנהל האירועים אותם הכניס כשנרשם, ומאפשר לו כעת לערוך את הפרטים (כמו למשל שם, תמונת פרופיל או כתובת).

זרימת ניווט למשתמש רגיל:

1. Login Page → Home Page

לאחר שהמשתמש מזין אימייל וסיסמה תקפים ולוחץ על כפתור "Login", המערכת מבצעת אימות מול Firebase Authentication. אם ההתחברות הצליחה, המשתמש מועבר אוטומטית למסך הבית (Home Page).

Login Page → Register Page → Home Page

אם המשתמש לוחץ על כפתור "Create Account", הוא עובר למסך ההרשמה (Register Page). לאחר הקמת חשבון חדש בהצלחה, המערכת מפנה אותו ישירות למסך הבית.

2. Home Page → Event Details Page

כאשר המשתמש לוחץ על אירוע מסוים מרשימת האירועים במסך הבית או על אחד האייקונים המופיעים במפה, המסמנים אירועים שקיימים, המערכת פותחת את מסך פרטי האירוע. שם הוא רואה תיאור מלא, מיקום, תאריך ומספר מקומות זמינים.

3. שתי אפשרויות פעולה:

א. Event Details → Register (הרשמה לאירוע)

אם המשתמש לא רשום לאירוע והוא לוחץ על כפתור "Register", המערכת תבצע בדיקת זמינות מקום ועדכון מצב האירוע. לאחר הפעולה, המשתמש נשאר באותו מסך והכפתור משתנה ל "Cancel Registration".

ב. Event Details → Cancel Registration

אם המשתמש לוחץ על "Cancel Registration", המערכת מוחקת את ההרשמה, מעדכנת את מספר המקומות הפנויים, המשתמש נשאר במסך, אך הכפתור חוזר להיות "Register".

4. Home Page → Profile Page

בלחיצה על כפתור התפריט (☰) במסך הבית, נפתח תפריט ניווט צדדי. דרך התפריט, המשתמש יכול לבחור באפשרות "Profile", ולהגיע למסך הפרופיל שבו ניתן לערוך פרטים אישיים, לעדכן רדיוס חיפוש ולהגדיר ז'אנרים מועדפים..

5. Home Page → My Events Page → Event Details Page

בלחיצה על הכפתור "My Events", המשתמש מגיע למסך המרכז את כל האירועים אליהם הוא נרשם. משם ניתן ללחוץ על כל אירוע ולפתוח שוב את מסך פרטי האירוע (Event Details Page).

6. Home Page → Notifications Page

דרך תפריט הניווט, המשתמש בוחר באפשרות "Notifications". המסך מציג התראות כגון ביטול אירוע, שינוי בפרטי אירוע או עדכון בזמינות מקומות.

זרימת ניווט - בעל עסק (Owner)

1. Login Page → Owner Dashboard Page

כאשר בעל אירוע נכנס עם חשבון שמקושר לתפקיד Owner, המערכת מפנה אותו ללוח הבית שלו.

Login → Register → Owner Dashboard
אם הוא לוחץ על "Create Account" ופותח חשבון חדש, ולאחר מכן מוגדר כבעל עסק - הוא מגיע ללוח הבית.

Owner Dashboard → Create Event Page .2
כאשר בעל העסק לוחץ על כפתור "Create Event", הוא מגיע למסך יצירת אירוע חדש. שם הוא מזין פרטים (שם האירוע, תאריך, מיקום, כמות מקומות וכו') ושומר את האירוע.

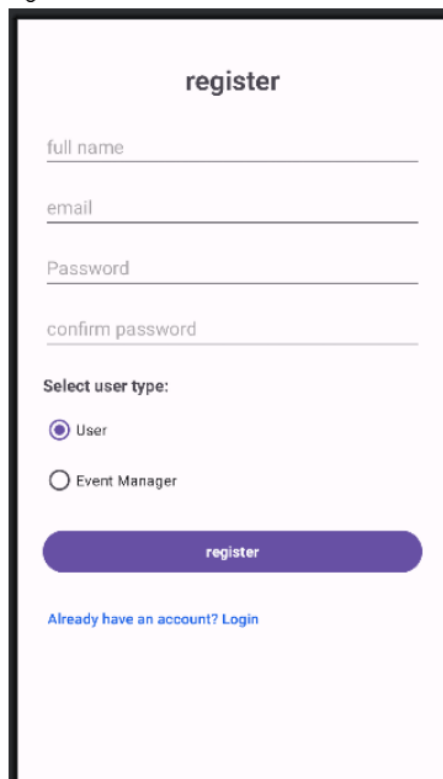
Owner Dashboard → Edit Event Page .3
לחיצה על אירוע קיים בלוח הבית, שם מופיעים האירועים אותם יצר הowner, פותחת את מסך עריכת האירוע.
מכאן ניתן לעדכן פרטים, לשנות כמות מקומות, לבטל/למחוק את האירוע.
לאחר השמירה, המשתמש חוזר ללוח הבית.

Owner Dashboard → Owner Notifications Page .4
דרך תפריט הניווט, בעל העסק בוחר באפשרות "Notifications".
המסך מציג התראות כגון:
אירוע שהגיע לסולד אאוט, אירוע שהתפנו בו מקומות, שינויים שהמערכת זיהתה ברישומים, התראות תפעוליות נוספות.

Owner Dashboard → Update profile Page .5
דרך תפריט הניווט, בעל העסק בוחר באפשרות "Edit Profile". המערכת פותחת מסך עדכון פרטים, בו ניתן לערוך פרטי בעל העסק, פרטי המקום ותיאור כללי.

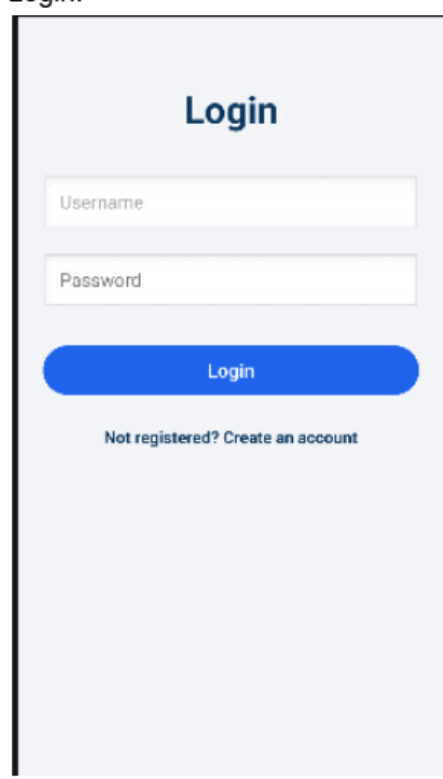
ייצוג ויזואלי:

register:



The register form is titled "register" in bold. It contains four input fields: "full name", "email", "Password", and "confirm password". Below these fields is a section titled "Select user type:" with two radio button options: "User" (selected) and "Event Manager". At the bottom, there is a purple "register" button and a link that says "Already have an account? Login".


Login:





The login form is titled "Login" in bold. It contains two input fields: "Username" and "Password". Below these fields is a blue "Login" button. At the bottom, there is a link that says "Not registered? Create an account".

User Profile:

Profile & Preferences


**John Smith**


 Edit Profile




Distance

25 km

 Favorite Genres



Notification Settings





My Registered Events

Logout







Home:

Hello User Name



My Events

Your location



Events Near Me

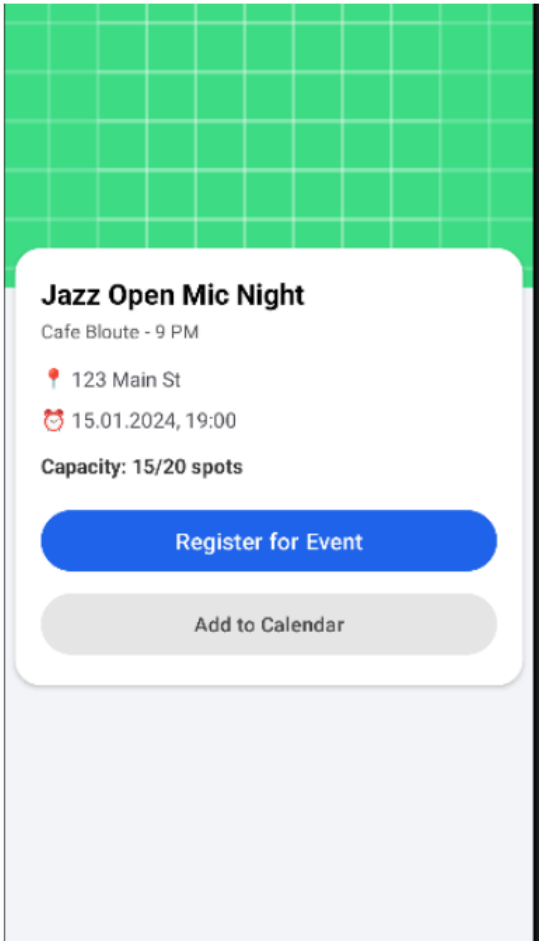
Map Placeholder

Jazz Open Mic
Cafe Blue Note – 7 PM
Jazz

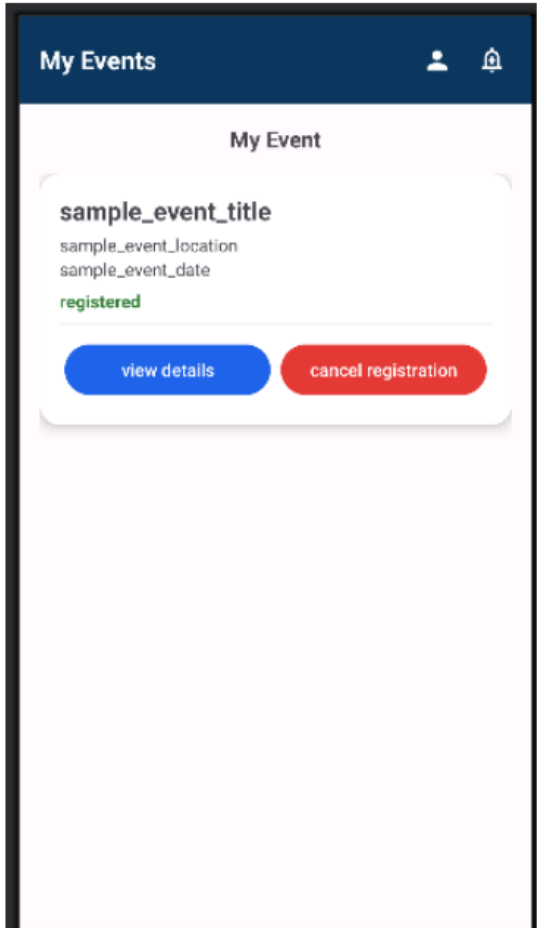
Rock Night
The Cavern – Blues
Rock

Open Jam Session

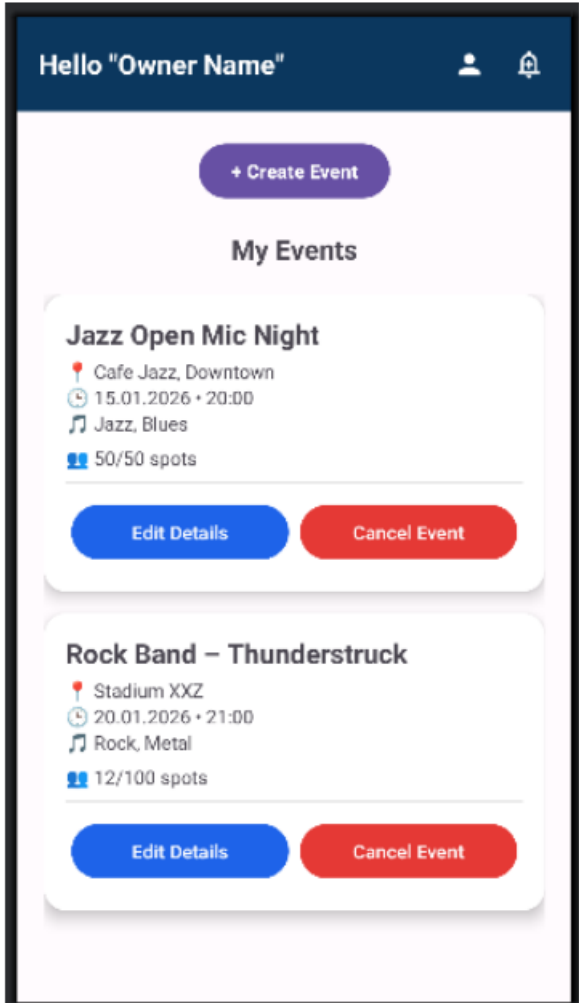
Event details:



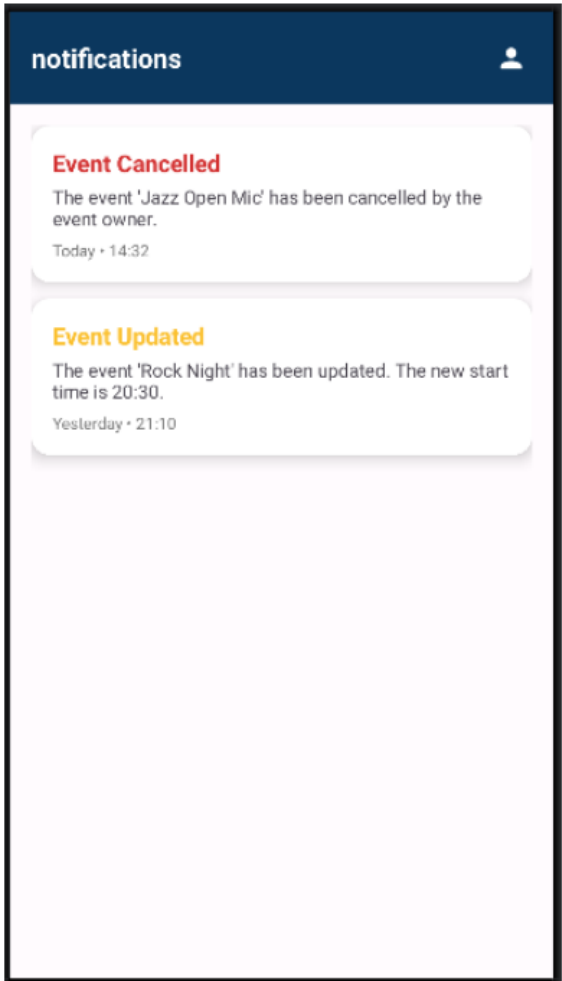
My Event (User)



Owner Dashboard:



Notifications (User)



Create Event:

Create New Event

Event Name

Location

Date

Time

Music genre

Max Capacity


Event Description

Publish Event

Cancel

update owner details:

Edit Profile

 owner name

Business / Owner Name

Email

Phone Number

Address

Save Changes

Notification (owner)

notifications

Event Sold Out

Your event 'Jazz Open Mic' has reached full capacity (sold out).

Today • 13:10

Seats Available

Several seats have opened for your event 'Rock Night' following cancellations.

Yesterday • 18:45

Event Updated

You updated details for 'Acoustic Night'. All registered users were notified.

05/01/2025 • 16:57

Edit Event:

Edit Event

Event Title

Event Description

Location

Date (DD/MM/YYYY)

Time (HH:MM)

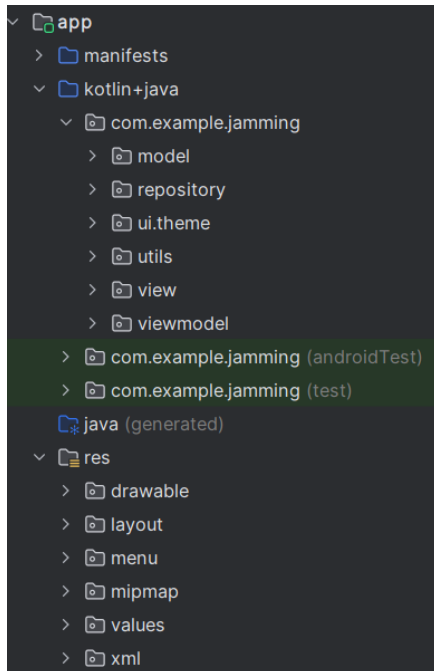
Capacity

Save Changes

Delete Event

Task Details: Design Patterns for Android Studio

בפרויקט בחרתי במבנה תיקיות שמבוסס על עקרונות של הפרדת אחריות וארכיטקטורה שכבתית, במטרה לשמור על קוד קריא, מודולרי, וקל לתחזוקה ולהרחבה.



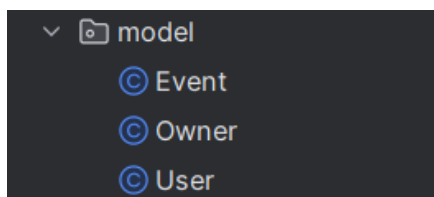
מבנה כללי של הפרויקט:

בתיקית app נמצאים כל רכיבי האפליקציה:

manifests מכיל את AndroidManifest.xml שבו מוגדרים רכיבי האפליקציה, Activities, הרשאות services וכו'.

Kotlin+java מכיל את קוד המקור של האפליקציה.

Res מכיל את כל משאבי ה ui (xml, תמונות, צבעים, מחרוזות וכו').



מבנה קוד המקור מכיל תתי תיקיות:

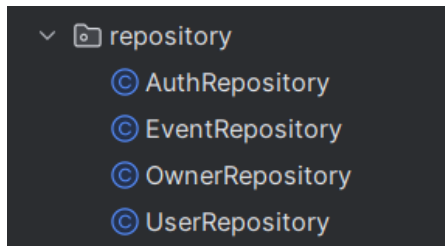
model

תיקייה זו מכילה את מחלקות המודל:

משתמשים (User), אירועים (Event) ובעלי אירועים (Owner)

אלו מחלקות שמייצגות ישויות לוגיות באפליקציה, לרוב עם שדות, בנאים ו- getters. הן אינן תלויות ב־ UI או במסד הנתונים.

הבחירה במבנה זה מאפשרת שימוש אחיד בנתונים בכל שכבות האפליקציה.
בנוסף, המבנה יוצר חלוקה נוחה של עבודה בינינו נפרט על כך בהמשך.



repository

תיקיה זו אחראית על גישה לנתונים:

תקשורת עם (Authentication, Firestore) Firebase שלילת נתונים, שמירה ועדכון שלהם.

ה-UI אינו ניגש ישירות למסד הנתונים, אלא דרך שכבה זו.

AuthRepository

אחראי על פעולות אימות משתמשים (Authentication), כגון: התחברות (Login) הרשמה (Register) ניהול משתמש מחובר (FirebaseAuth)

UserRepository

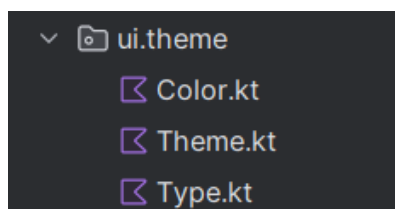
אחראי על ניהול נתוני משתמשים כגון: שלילת פרטי משתמשים מ-Firestore, חיפוש משתמשים לפי מזהים או שדות, עדכון נתוני משתמש ועוד.

OwnerRepository

אחראי על נתונים של בעלי אירועים (Owners) כגון: שלילת פרטי Owner לפי UID, ניהול מידע ייעודי לבעלי אירועים.

EventRepository

אחראי על ניהול אירועים כגון: יצירה, עדכון ומחיקה של אירועים, שלילת אירועים ממסד הנתונים שאילתות לפי מיקום, בעלים, או מאפיינים אחרים.

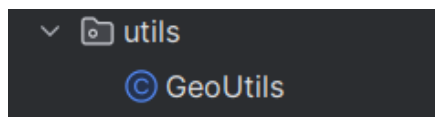


ui.theme

תיקיה ייעודית לנושא העיצוב: (Theme)

- צבעים
- טיפוגרפיה
- סגנונות כלליים

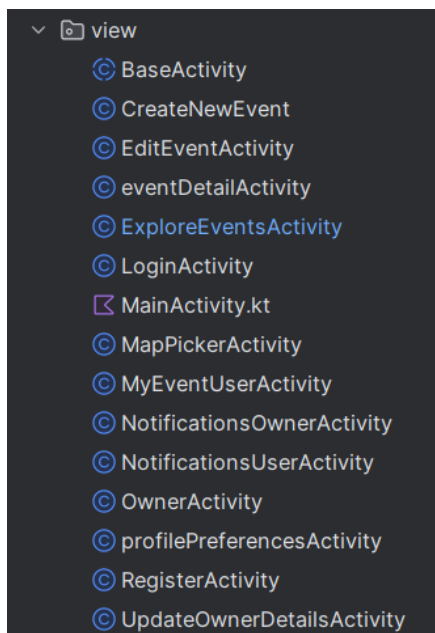
utils



תיקיית עזר שמכילה מחלקות כלליות:פונקציות עזר

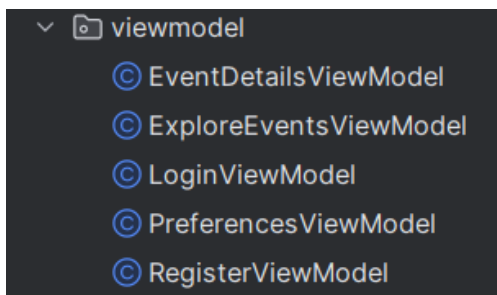
- GeoUtils מחלקת עזר גיאוגרפית מחשבת מרחק בין שני זוגות של קואורדינטות.

view



שכבת התצוגה:(UI)

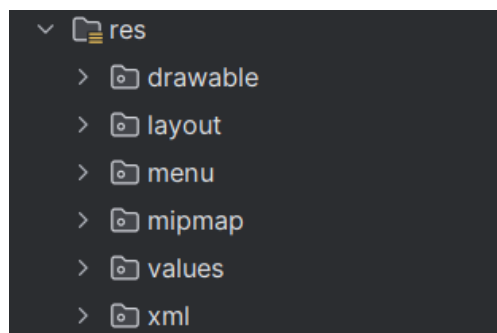
- Activities / Fragments
 - אחראית על הצגת המידע למשתמש ותגובה לאינטראקציות (לחיצות, ניווט)
- שכבה זו אינה מכילה לוגיקה עסקית או קוד של גישה לנתונים.
- BaseActivity** מחלקת בסיס משותפת לכל ה-Activities, מרכזת לוגיקה והתנהגות כללית שחוזרת במסכים שונים.
- CreateNewEvent** מסך ליצירת אירוע חדש והזנת פרטי האירוע על ידי המשתמש.
- EditEventActivity** מסך לעריכת פרטים של אירוע קיים.
- eventDetailActivity** מסך להצגת פרטי אירוע בודד, כולל מידע מלא ואפשרויות אינטראקציה.
- ExploreEventsActivity** מסך חיפוש וגלישה באירועים זמינים, כולל הצגה על גבי מפה.
- LoginActivity** מסך התחברות משתמשים באמצעות מערכת האימות.
- MainActivity.kt** נקודת הכניסה הראשית לאפליקציה וניהול הניווט הראשוני.
- MapPickerActivity** מסך לבחירת מיקום גיאוגרפי על גבי מפה.
- MyEventUserActivity** מסך להצגת האירועים שהמשתמש רשום אליהם או משתתף בהם.
- NotificationsOwnerActivity** מסך התראות ייעודי לבעלי אירועים.
- NotificationsUserActivity** מסך התראות ייעודי למשתמשים רגילים.
- OwnerActivity** מסך ראשי לניהול פעולות של בעל אירוע.
- profilePreferencesActivity** מסך להגדרת והעדפת פרטי פרופיל המשתמש.
- RegisterActivity** מסך הרשמת משתמש חדש למערכת.
- UpdateOwnerDetailsActivity** מסך לעדכון פרטים אישיים של בעל אירוע.



viewmodel

Repository: שכבת ביניים בין ה-View ל-Repository

- מכילה **לוגיקה** שקשורה ל-UI
- מנהלת state של מסכים



תיקיית res – משאבי UI

בתיקיית res נשמרים כל משאבי הממשק:

Layout קבצי XML של מסכים

drawable אייקונים ותמונות

menu תפריטים

values מחרוזות, צבעים, סגנונות

xml קבצי קונפיגורציה (כמו backup_rules.xml)

בחרנו במבנה זה משום שהוא מאפשר הפרדת אחריות ברורה בין שכבות האפליקציה (UI, לוגיקה ונתונים), (משפר את קריאות הקוד ותחזוקתו, מקל על הרחבה עתידית והוספת פיצ'רים, ומונע תלות ישירה בין מסכי המשתמש לבין מקורות הנתונים, בהתאם לעקרונות פיתוח מקובלים. בנוסף, חלוקה ברורה זו אפשרה לנו כקבוצה לעבוד במקביל ובשיתוף פעולה מבלי לדרוס קוד של חבר אחר.

היתרונות המרכזיים של הארכיטקטורה שבחרנו (MVVM + Repository) הם:

תחזוקתיות גבוהה (Maintainability): הפרדת אחריות ברורה—ה-view מציג UI בלבד,

ה- viewmodel מחזיק לוגיקה ומצב, וה- repository מטפל בנתונים (Firebase/DB) כך שינוי ב- UI כמעט לא "שובר" שכבות אחרות ולהפך.

סקיילביליות (Scalability): קל להוסיף מסכים/פיצ'רים חדשים כי יש תבנית עבודה קבועה :
Data → Repository → ViewModel → Activity/Fragment מונע "ספגטי" כשכמות הקוד גדלה.

בדיקות קלות יותר (Testability): לוגיקה נמצאת ב- ViewModel/Repositories ולא בתוך Activities, ולכן אפשר לבדוק אותה בצורה נקייה יותר בלי להסתמך על UI.

ביצועים וחווית משתמש (Performance/CX): ה- UI נשאר "רזה" ולא חוסם את ה- main thread גישה ישירה לנתונים; בנוסף אפשר לשפר בקלות/caching מניעת קריאות כפולות דרך ה- Repository.

גמישות להחלפת תשתיות: אם בעתיד מחליפים Firebase ב- API אחר DB/אחר—השינוי מתרכז בעיקר ב- repository ולא בכל המסכים.

חלוקת התפקידים בפרויקט בוצעה באופן ברור אך גמיש: אור היה אחראי בעיקר על צד בעל האירוע, רעות על חוויית המשתמש, ושירת על ניהול הנתונים ושכבת ה- Repository-עם זאת, העבודה התנהלה בשיתוף פעולה מלא, וכל אחד מחברי הצוות חרג מתחום אחריותו העיקרי ותרם גם לחלקים נוספים בפרויקט, בהתאם לצרכים שעלו במהלך הפיתוח.