

שם בית הספר: עתיד דביר בית שמש.

שם העבודה: אפליקציית צ'אטים.

שם התלמיד: שחר כהן.

ת.ז של התלמיד: 214599961.

שם המנחה: אורן אביעזר.

שם חלופה: עבודת גמר תכנון ותכנות מערכות  - טלפונים חכמים באנדרואיד

תאריך ההגשה: 1/5/2023

תוכן

[מבוא 3](#_Toc136188873)

[הרקע לפרויקט 3](#_Toc136188874)

[תהליך המחקר 3](#_Toc136188875)

[אתגרים מרכזיים 4](#_Toc136188876)

# מבוא

## הרקע לפרויקט

שם הפרויקט: אפליקציית צ'אטים.

תיאור הפרויקט: אפליקציית צ'אטים שבה כל הודעה שאתה שולחת מגיעה לכל המשתמשים באפליקציה והם יכולים לשוחח ביחד.

קהל היעד: קהל היעד שלי הוא אנשים שרוצים לתקשר עם אנשים אקראיים באינטרנט ולהכיר אנשים חדשים.

הסיבות לבחירת הנושא: רציתי ללמוד על איך להשתמש בfirebase ולראות איך אני יכול ליצור אפליקציה מגניבה וקלה לטלפון וחשבתי שיהיה מגניב ליצור אפליקציית צ'אטים שאנשים יוכלו לתקשר דרכה.

## תהליך המחקר

בפרויקט יישום הצ'אט שלי, השתמשתי במספר טכנולוגיות שלא היו חלק מתכנית הלימודים.

אחת הטכנולוגיות שהשתמשתי בהן הייתה Firebase, שהיא פלטפורמת פיתוח אפליקציות למובייל ולאינטרנט המספקת כלים ושירותים שונים לפיתוח אפליקציות, כמו מסד נתונים בזמן אמת, אימות והודעות בענן.

השתמשתי גם ב-Firestore, שהוא מסד נתונים של מסמכים NoSQL המהווה חלק מפלטפורמת Firebase ומספק סנכרון בזמן אמת ותמיכה לא מקוונת עבור אפליקציות מובייל ואינטרנט.

טכנולוגיה נוספת שהשתמשתי בה היא ה-API של Dicebear, שהוא API מחולל אווטרים חינמי ופתוח, המאפשר למפתחים ליצור אווטרים ייחודיים וניתנים להתאמה אישית עבור המשתמשים שלהם. ממשק API זה שימש ליצירת אווטרים עבור המשתמשים באפליקציית הצ'אט שלי.

כדי לחקור את הטכנולוגיות הללו, צפיתי בסרטוני YouTube שונים וקראתי את התיעוד שלהם כדי להבין את התכונות שלהם וכיצד לשלב אותם באפליקציה שלי.

בדקתי גם אפליקציות קיימות שהשתמשו בטכנולוגיות אלו כדי לקבל מושג על אופן השימוש בהן ובדקתי את המצב הנוכחי בשוק כדי לראות את הפופולריות והביקוש לטכנולוגיות אלו.

## אתגרים מרכזיים

במהלך הפיתוח של פרויקט אפליקציית הצ'אט שלי, התמודדתי עם מספר אתגרים. אחת הבעיות העיקריות שהתמודדתי איתי הייתה חוסר הידיעה איך להשתמש בטכנולוגיות מסוימות, כמו Firebase ו-Firestore. זה דרש ממני להשקיע זמן נוסף במחקר וללמוד כיצד להשתמש בטכנולוגיות הללו ביעילות.

בעיה נוספת שהתמודדתי איתי הייתה איתור באגים בקוד, דבר שעלול לקחת זמן ומתסכל. עם זאת, הצלחתי להתגבר על האתגר הזה על ידי שימוש בדיבאגר ב-Android Studio ועל ידי חיפוש של הבעיות באינטרנט ובפורומים כמו stackoverflow .

פרויקט אפליקציית הצ'אט עונה על הצורך בפלטפורמת צ'אט מרכזית שבה משתמשים יכולים לתקשר זה עם זה בזמן אמת. הפתרון שמספק פרויקט זה הוא אפליקציית צ'אט ידידותית למשתמש וניתנת להתאמה אישית המאפשרת למשתמשים להירשם, להיכנס ולשלוח הודעות לחדר צ'אט מרכזי.

כדי להתגבר על האתגרים שעמדתי בפניהם במהלך פיתוח הפרויקט, ערכתי מחקר מקדים על הטכנולוגיות בהן השתמשתי ובדקתי פתרונות שונים.

לדוגמה, צפיתי במדריכי YouTube וקראתי תיעוד כדי ללמוד כיצד להשתמש ביעילות ב-Firebase וב-Firestore. השתמשתי גם בדיבאגר ב-Android Studio וחיפשתי פתרונות לבעיות שלי בגוגל.

# תיאור תחום הידע – פרק מילולי

## אובייקטים נחוצים

בפרויקט יישום הצ'אט שלי, היו כמה אובייקטים הכרחיים שהייתי צריך לשקול וליישם. חפצים אלה כוללים:

1. משתמשים: אובייקט זה מייצג את המשתמשים באפליקציית הצ'אט. הוא כולל מידע כגון שם המשתמש, האימייל והסיסמה המשתמש.

2. הודעות: אובייקט זה מייצג את ההודעות שנשלחות ומתקבלות באפליקציית הצ'אט. הוא כולל מידע כגון תוכן ההודעה, שם השולח והשעה בה נשלחה ההודעה.

3. אווטרים של משתמש: אובייקט זה מייצג את האוואטרים שנוצרים עבור כל משתמש באפליקציית הצ'אט. הוא כולל את התמונה של האוואטר.

## סוגי נתונים

בפרויקט יישום הצ'אט שלי, השתמשתי במספר מבני נתונים כדי לייצג את המידע באפליקציה. לדוגמה:

1. רשימה: השתמשתי ברשימה כדי לשמור את כל ההודעות בצ'אט. זה איפשר לי להוסיף בקלות הודעות חדשות לצ'אט ולאחזר את כל ההודעות בעת הצורך.

2. אובייקט תמונה: השתמשתי באובייקט תמונה כדי לשמור את האווטרים של המשתמשים בצ'אט. כל הודעה מאחסנת את האווטאר של המשתמש ששלח את ההודעה, אשר נשלף מאובייקט התמונה.

3. מפה: יצרתי מחלקה סטטית ששומרת מפה שבה יש שמות משתמש, כל שם משתמש מצביע על תמונה. כשמישהו מבקש תמונה של שם משתמש, הכיתה בודקת אם השם קיים במפה ואם לא, היא מנותבת ל-dicebear API ומקבלת את התמונה, שומרת אותה במפה ומחזירה אותה. מבנה נתוני מפה זה שימש לאחסון ואחזור יעיל של אווטרים של משתמשים.

הבחירה במבני נתונים אלו התבססה על יעילותם וקלות השימוש שלהם. לדוגמה, מבנה נתוני הרשימה שימש לאחסון ואחזור הודעות בצ'אט, שהוא מקרה שימוש נפוץ ופשוט לרשימה. אובייקט התמונה שימש לאחסון ואחזור אווטארים של משתמשים, שהוא מקרה שימוש נפוץ עבור אובייקטי תמונה. מבנה נתוני המפה שימש לאחסון ואחזור יעיל של אווטרים של משתמשים בהתבסס על שמות המשתמש שלהם.

## תיאור פעולות על המידע

בפרויקט אפליקציית הצ'אט שלי, היו מספר פעולות שניתן לבצע על המידע המאוחסן במסד הנתונים. פעולות אלו כוללות:

1. הוספה: משתמשים יכולים להוסיף הודעות חדשות לצ'אט על ידי הקלדת הודעה ולחיצה על כפתור השליחה. לאחר מכן ההודעה מתווספת לרשימת ההודעות בצ'אט.

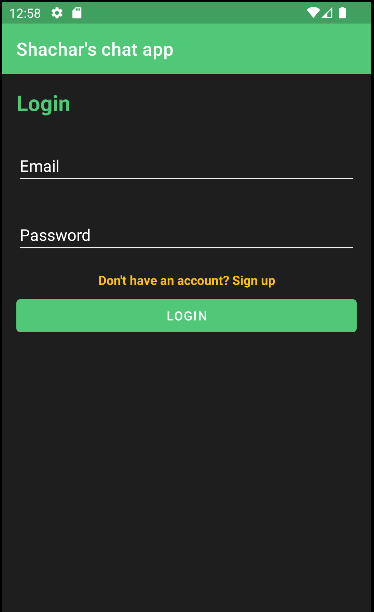
2. קריאה: משתמשים יכולים לקרוא הודעות בצ'אט על ידי גלילה ברשימת ההודעות. כל הודעה מציגה את שם השולח, הדמות והשעה שבה נשלחה ההודעה.

פעולות אלו יושמו באמצעות מסדי נתונים של Firebase ו-Firestore, המספקים ממשקי API קלים לשימוש להוספה, קריאה, עדכון ומחיקה של נתונים. הפעולות על המידע תוכננו להיות ידידותיות ואינטואיטיביות למשתמש, ולאפשר למשתמשים ליצור אינטראקציה קלה עם אפליקציית הצ'אט והתכונות שלה.

# ארכיטקטורה של הפרויקט

### מסך הפתיחה (התחברות):

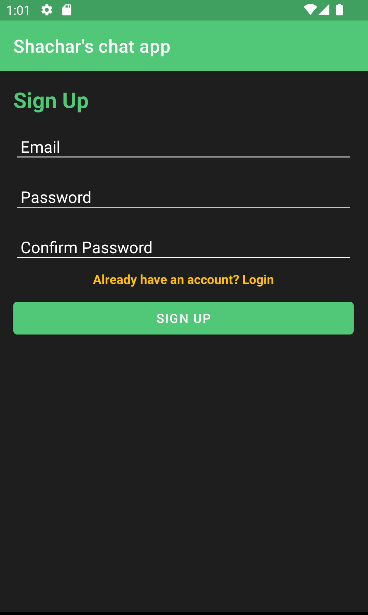
מסך הפתיחה הוא מסך התחברות והוא מכיל שדות להכנסת האימייל והסיסמה בשביל להתחבר, כפתור להתחברות וכפתור שמעביר אותך למסך ההרשמה.



### מסך ההרשמה:

במסך ההרשמה אפשר ליצור משתמש באפליקציית הצ'אטים, יש שמה שדות להכנסת האימייל הסיסמה ואישור הסיסמה וכפתור הרשמה.

יש גם כפתור שמעביר אותך למסך הפתיחה (התחברות) ובנוסף אחרי שאתה נרשם אתה גם עובר אוטומטית לסך הפתיחה.

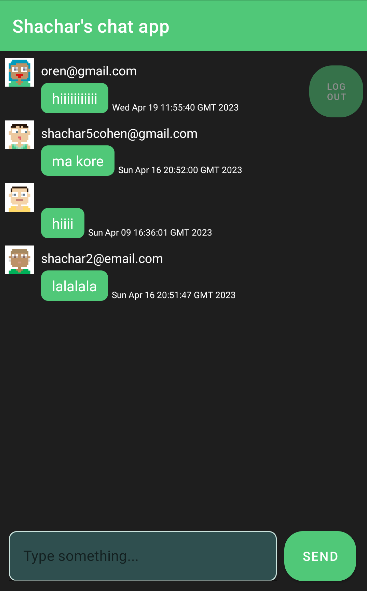


### מסך הצ'אט:

במסך הצ'אט אפשר לראות את ההודעות של כל המשתמשים, את השם משתמש את התמונה ואת זמן שליחת ההודעה.

בנוסף יש שמה שדה לשליחת הודעה וכפתור לשליחת הודעה.

יש שמה גם כפתור בשביל להתנתק מהמשתמש שלך ולחזור למסך הבית.



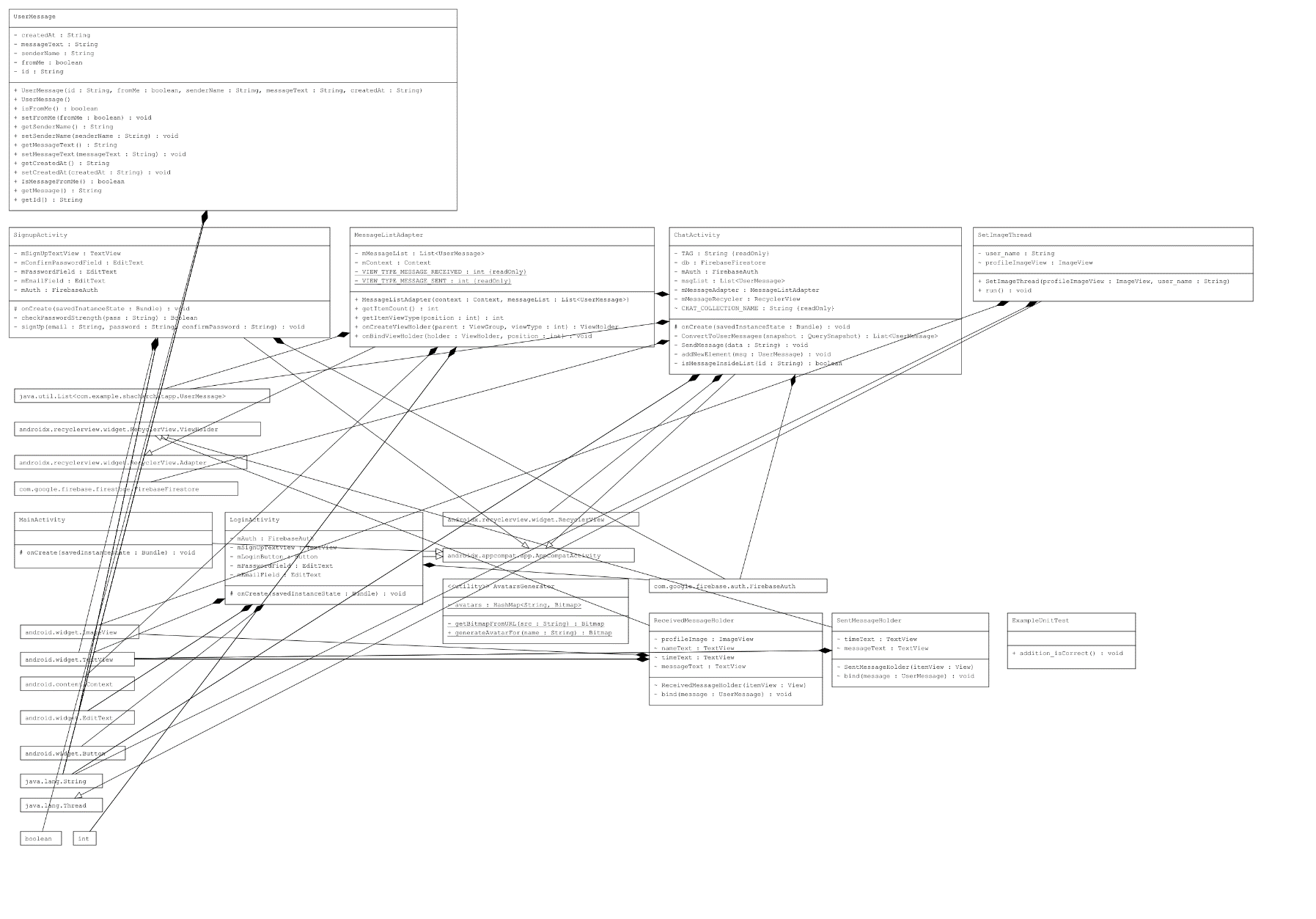
### תרשים מסכים:

מסך הצ'אט

מסך ההתחברות

מסך ההרשמה

## מחלקות הפרויקט:



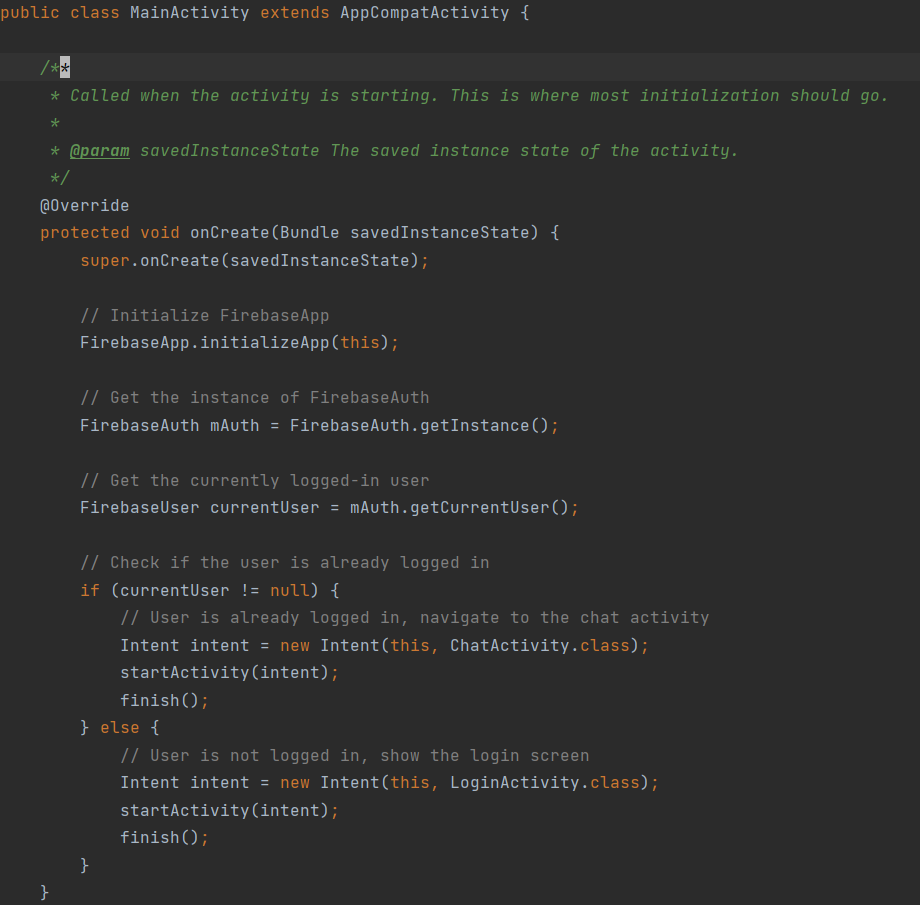
# מימוש הפרויקט

### מחלקת MainActivity:

תפקיד המחלקה הוא לבדוק אם מי שמתמש באפליקציה מחובר למשתמש שלו בfirebase, אם הוא כן להעביר אותו למסך של הchat ואם הוא לא להעביר אותו למסך ההתחברות.

למחלקה יש תכונה אחת onCreate שנקראת כאשר היא נוצרת ותפקידה הוא מה שרשמתי למעלה.

הקוד של המחלקה:



### מחלקת LoginActivity:

תפקיד המחלקה הוא לתת למשתמש באפליקציה אפשרות להתחבר למשתמש שלו בfirebase או לעבור למסך ההרשמה שמה יוכל להרשם לאפליקציה.

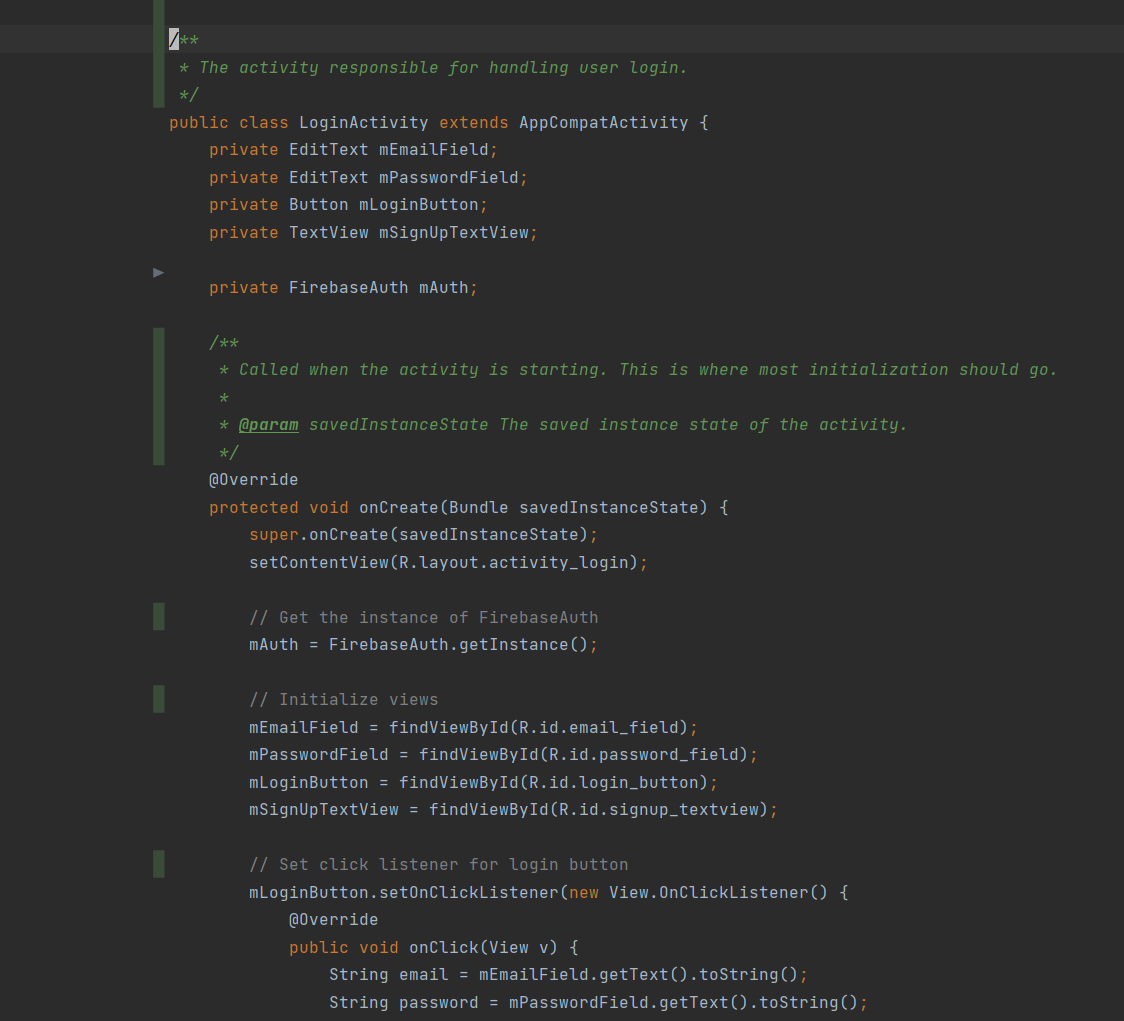
תכונות המחלקה הם:

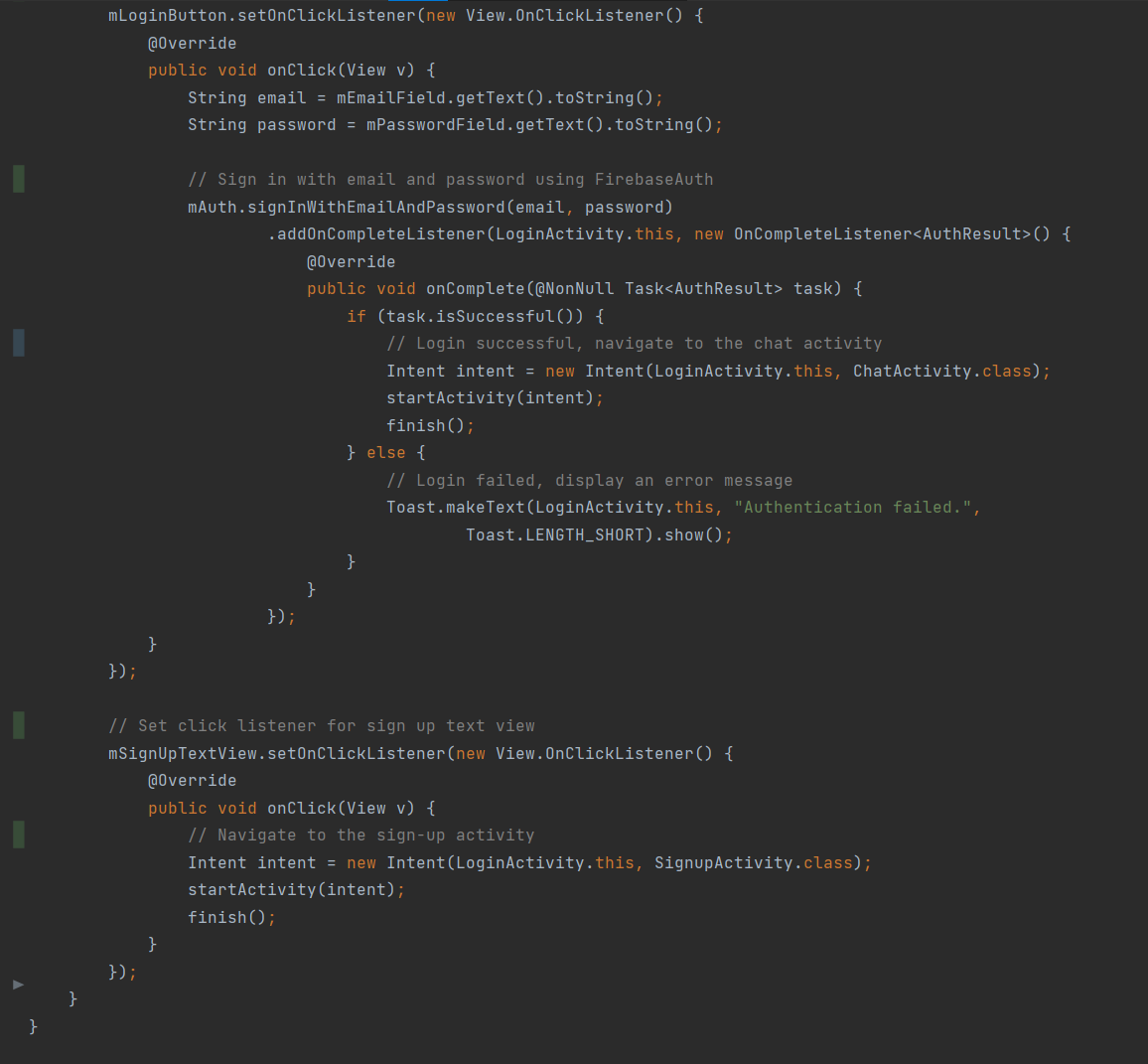
תכונת onCreate שנקראת כאשר המסך נוצר והיא מאתחלת את כל המשתנים ויוצרת eventים של onClick.

תכונה הonClick של כפתור ההתחברות – התכונה הזאתי מנסה להתחבר למשתמש שנתנו לנו ואם היא מצליחה היא עוברת לדף של הצ'אטים ואם לא היא רושמת למשתמש שהיא לא הצליחה.

תכונת הonClick של כפתור ההרשמה – התכונה הזאתי מעבירה את המשתמש למסך ההרשמה.

קוד המחלקה:





### מחלקת SignupActivity:

תפקידה של מחלקה זאתי היא לתת למשתמש ליצור משתמש חדש בfirebase.

תכונות המחלקה:

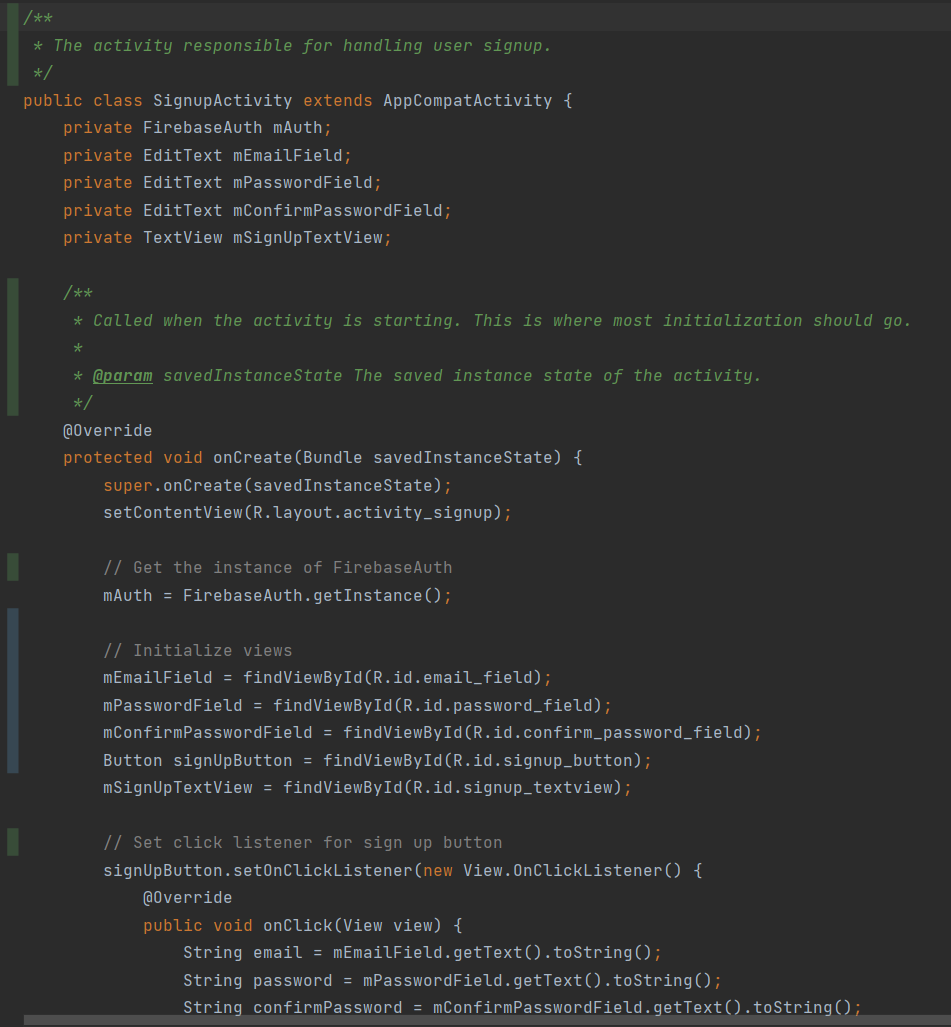
תכונת onCreate – מטרתה היא לאתחל משתנים וליצור eventים של onClick.

תכונת checkPasswordStrength – בודקת את החוזק של הסיסמה שהכניסו בעזרת regex ומחזירה אם הסיסמה חזקה מספיק או לא.

תכונת signUp – מקבלת אימייל וסיסמה ואישור סיסמה ובודקת אם הסיסמה שווה לאישור סיסמה ואם הסיסמה חזקה מספיק בעזרת הפונקציה checkPasswordStrength, אם הכל טטוב אז היא יוצרת משתמש חדש עם הפרטים האלו.

תכונת onClick של כפתור ההרשמה – משיגה את הפרטים של המשתמש מהשדות וקוראת לפונקציה signUp איתם.

תכונת onClick של כפתור ההתחברות – מעביר את המשתמש למסך ההתחברות.



### מחלקת ChatActivity:

תפקידה של המחלקה הזאתי היא להראות את הצ'אט למשתמש ולתת לו אפשרות לשלוח הודעות בצ'אט.

תכונות המחלקה:

תכונת onCreate – מטרתה היא לאתחל משתנים וליצור eventים של onClick.

תכונת ConvertToUserMessages – מקבל אובייקט של QuerySnapshot וממיר אותו לרשימה של אובייקטים של הודעות.

תכונת SendMessage – יוצר הודעה חדשה ושולח אותה לצ'אט.

תכונת addNewElement – מוסיף הודעה חדשה למסך ומציג אותה.

תכונת isMessageInsideList – בודק אם הודעה עם id כלשהו נמצאת ברשימה של הודעות.

תכונת onClick של כפתור השליחה – מקבלת את ההודעה שהמשתמש הכניס ושולחת אותה לצ'אט.

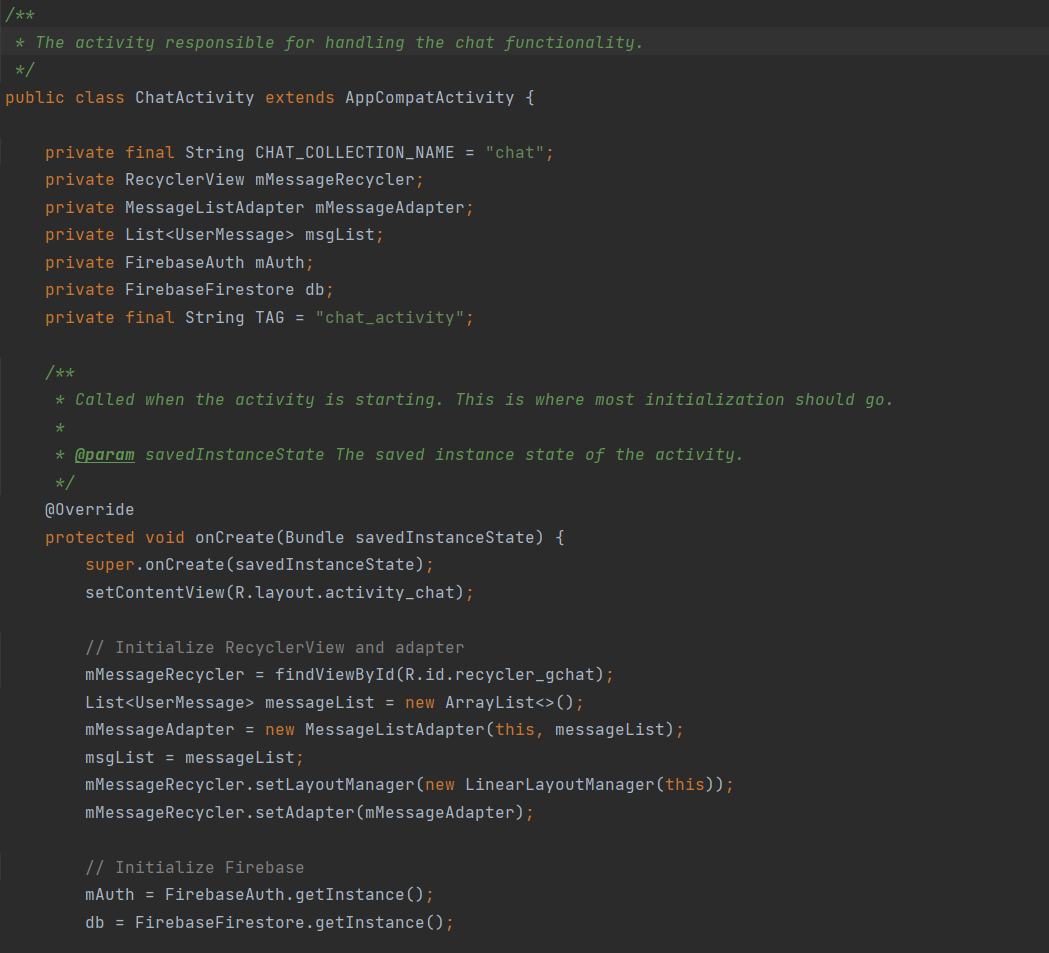
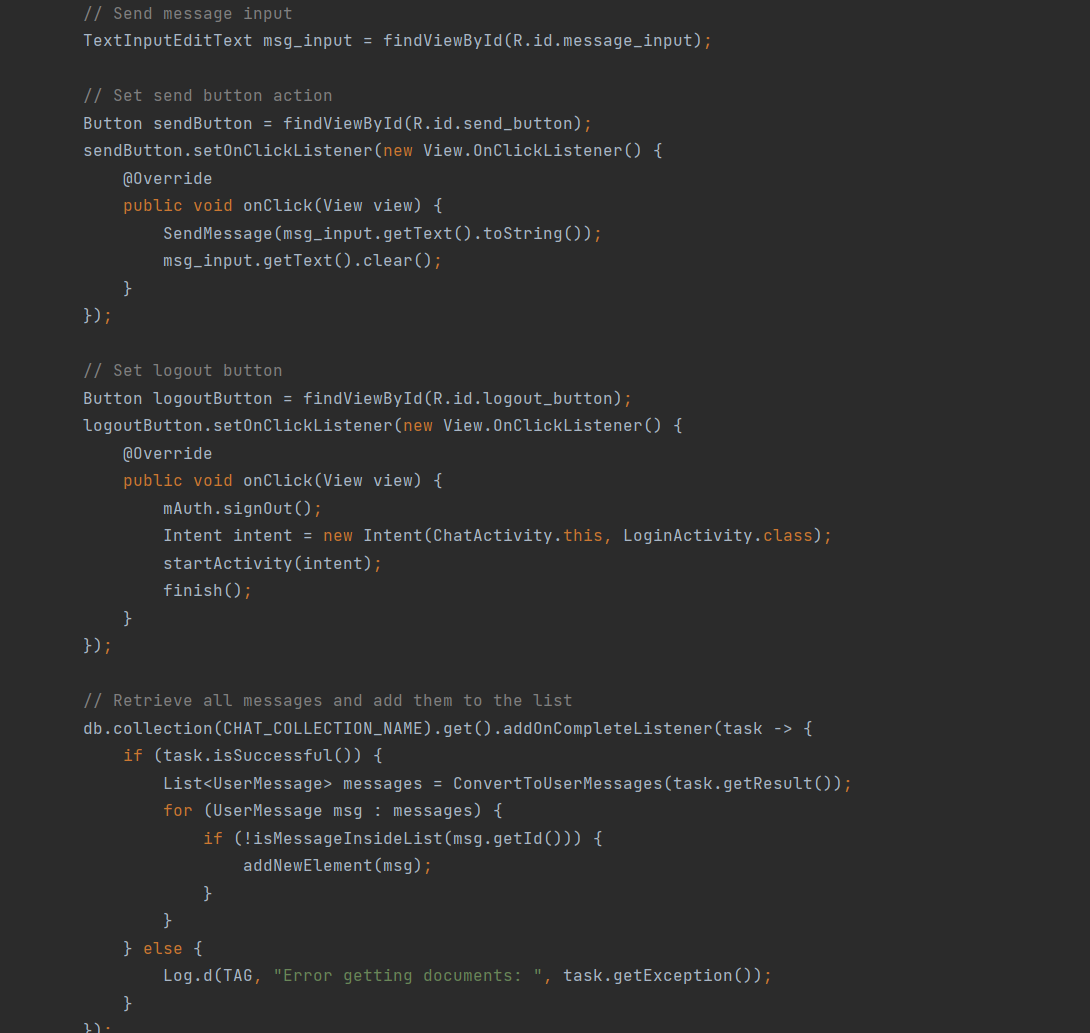
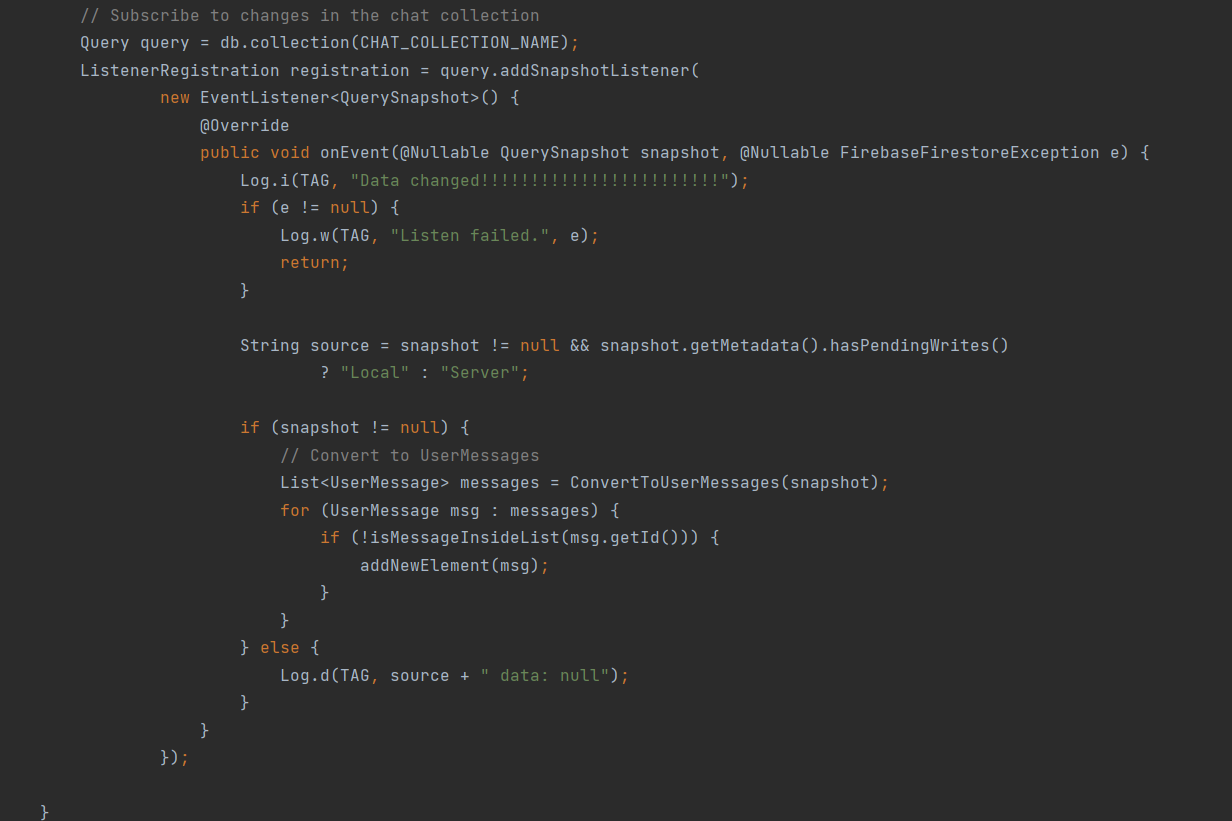
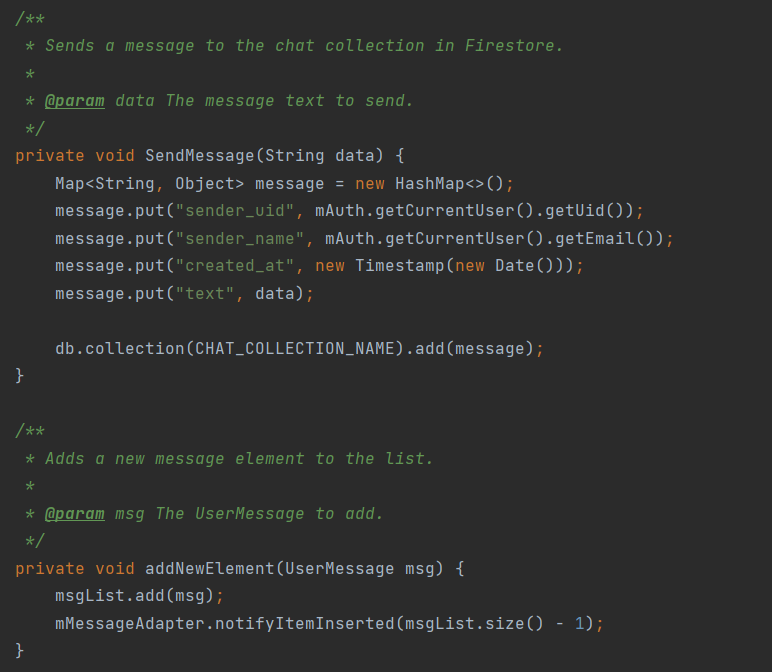
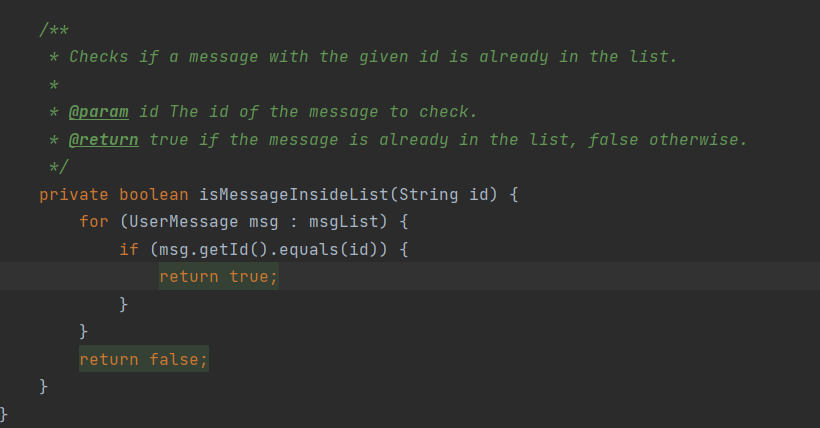
תכונת onClick של כפתור ההתנתקות – יוצא מהמשתמש וחוזר לחלון ההתחברות.

תכונת SnapshotListener של המסד נתונים – מחכה שיהיה הודעות חדשות ואז מוסיף אותם למסך.

תיאור המידע במסד נתונים firebase:

יש collection שקוראים לה chat והיא מכילה הרבה documents, כל document הוא הודעה חדשה בצ'אט שנוספה, כל הודעה מכילה: תאריך שליחה, שם השולח, הuid של השולח, הטקסט של ההודעה.

הקוד של המחלקה:

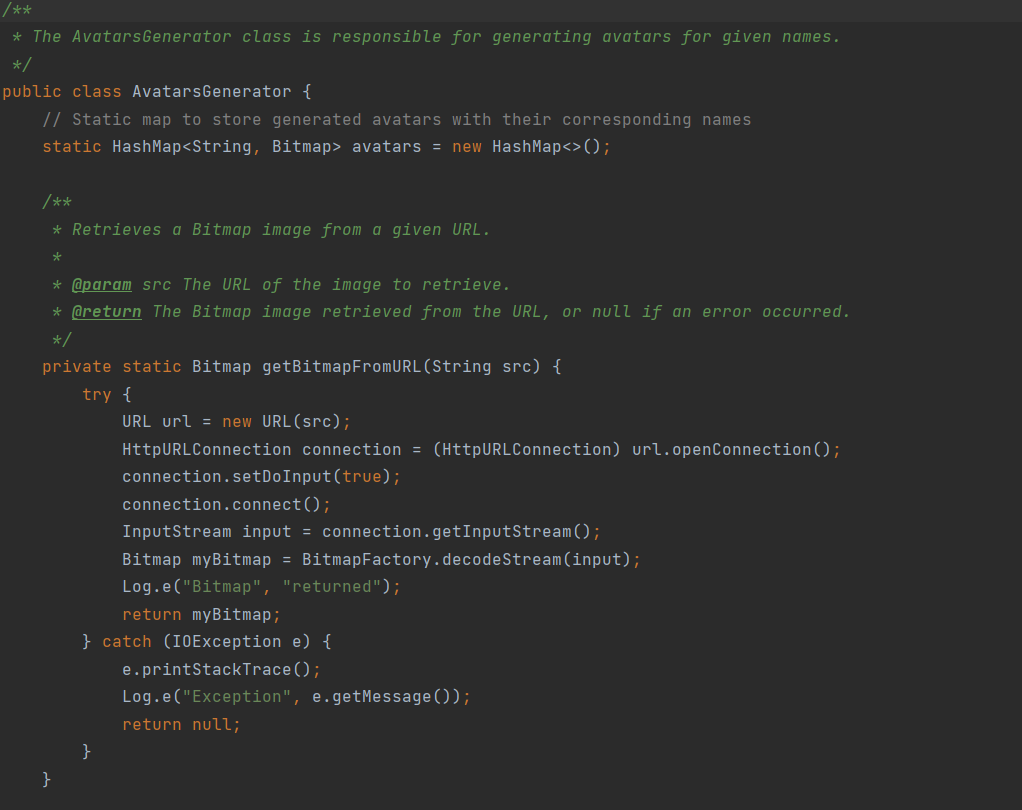
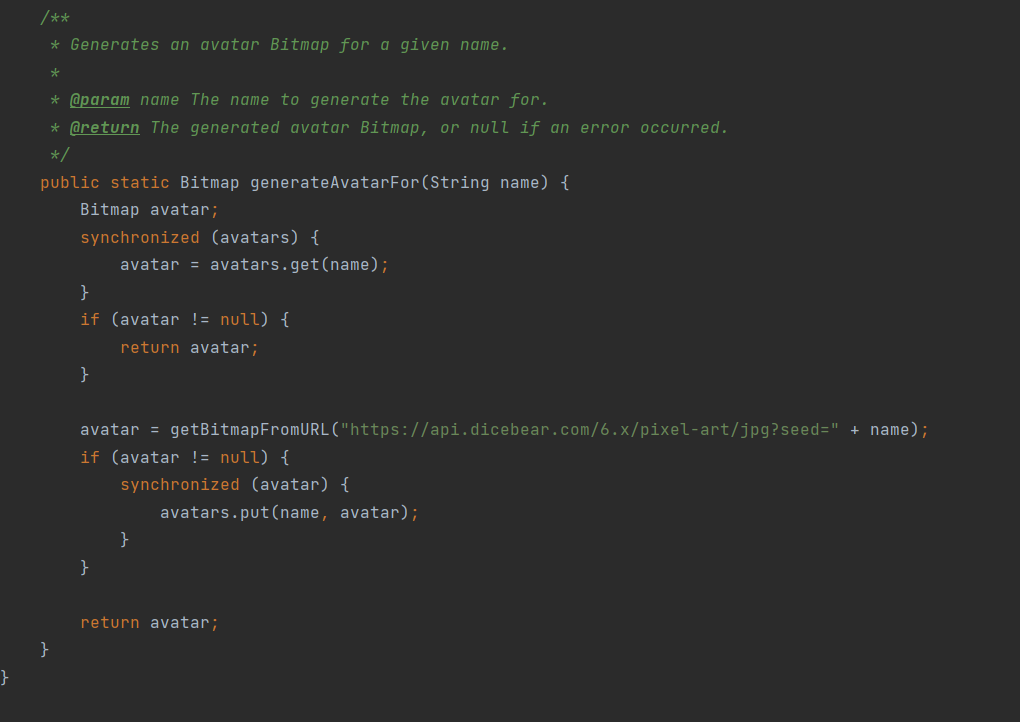
### מחלקת AvatarsGenerator:

מחלקה סטטית שמטרתה יצירה ושמירה של אוואטרים למשתמשים ששלחו הודעות.

תכונות המחלקה:

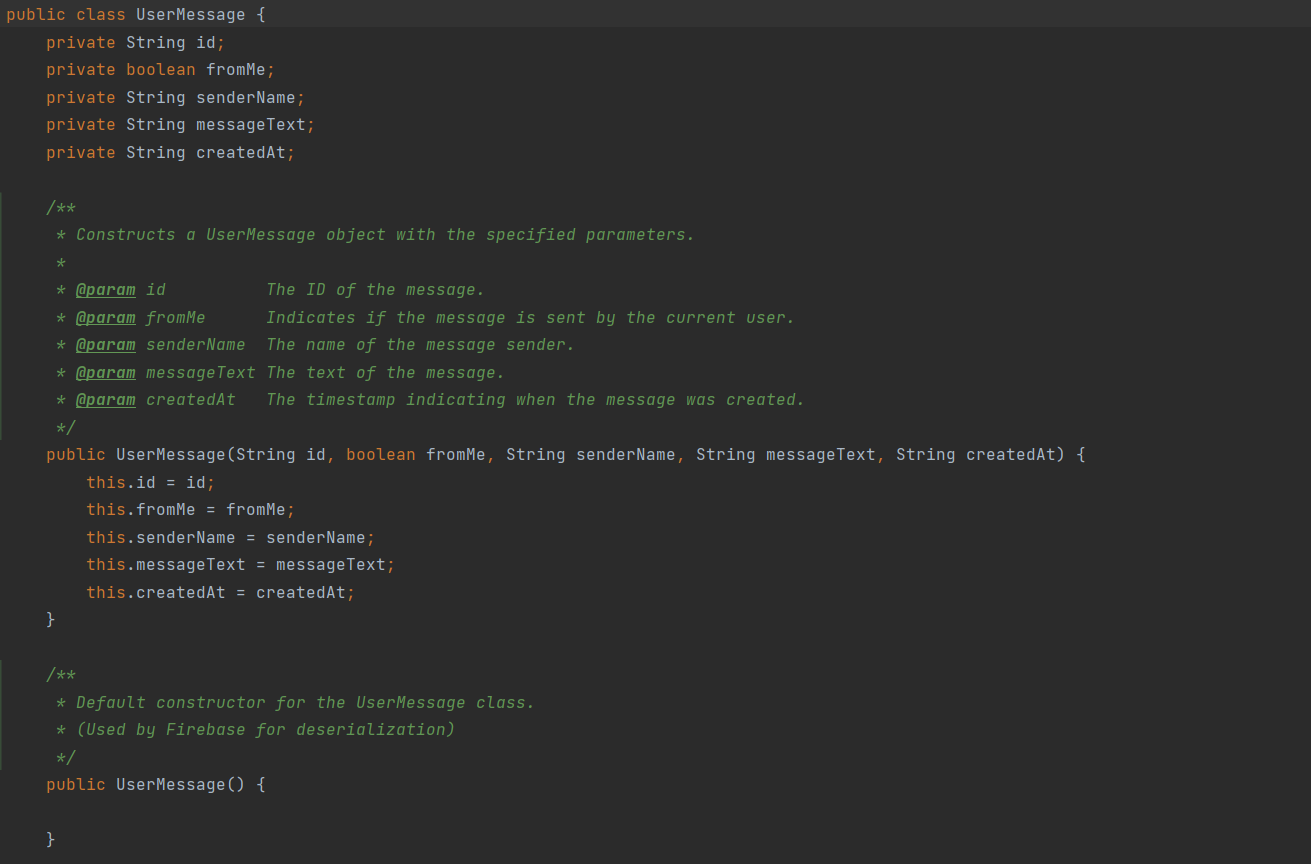
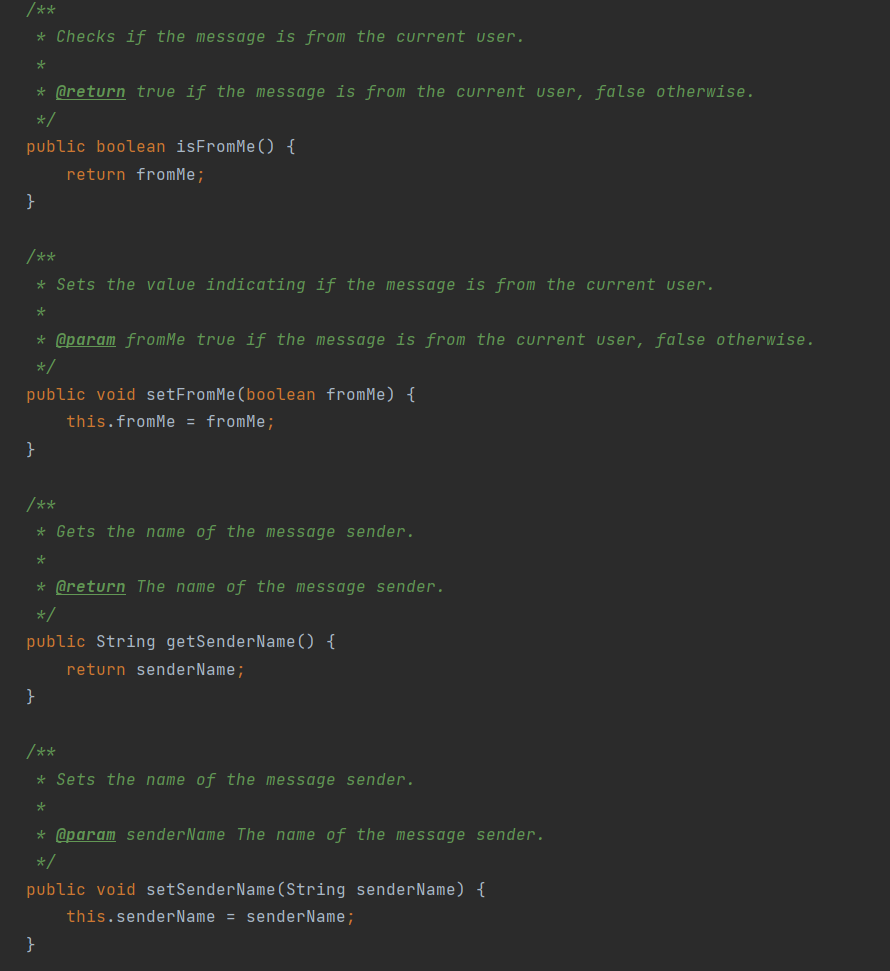
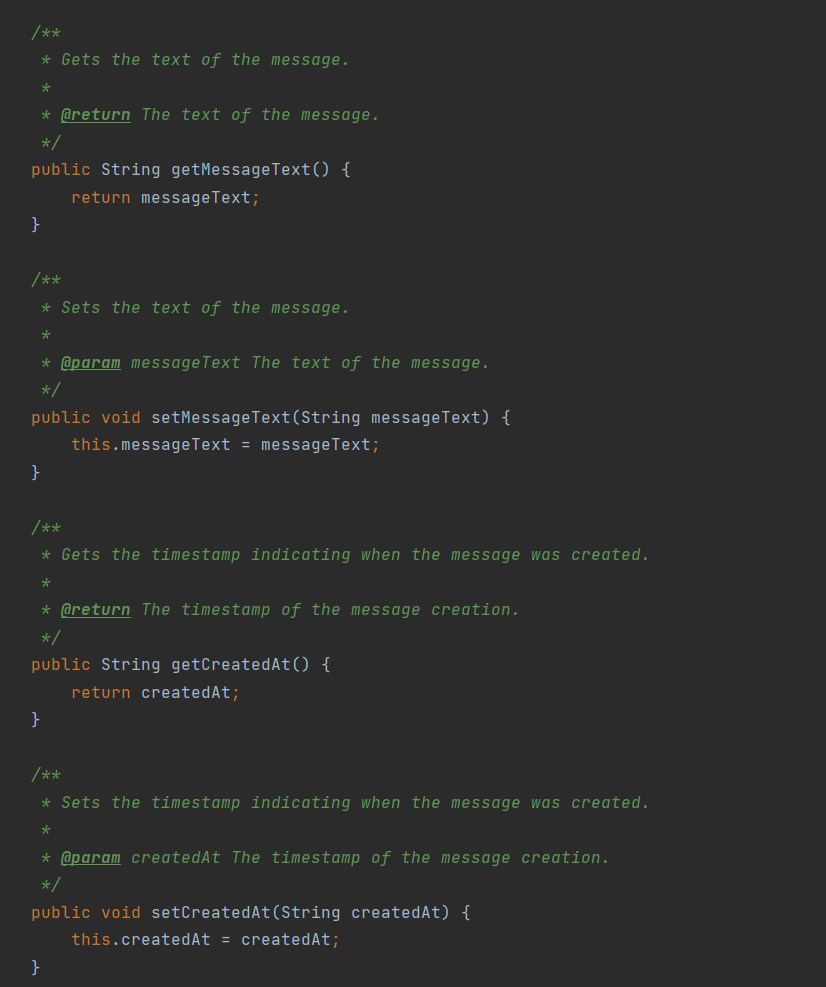
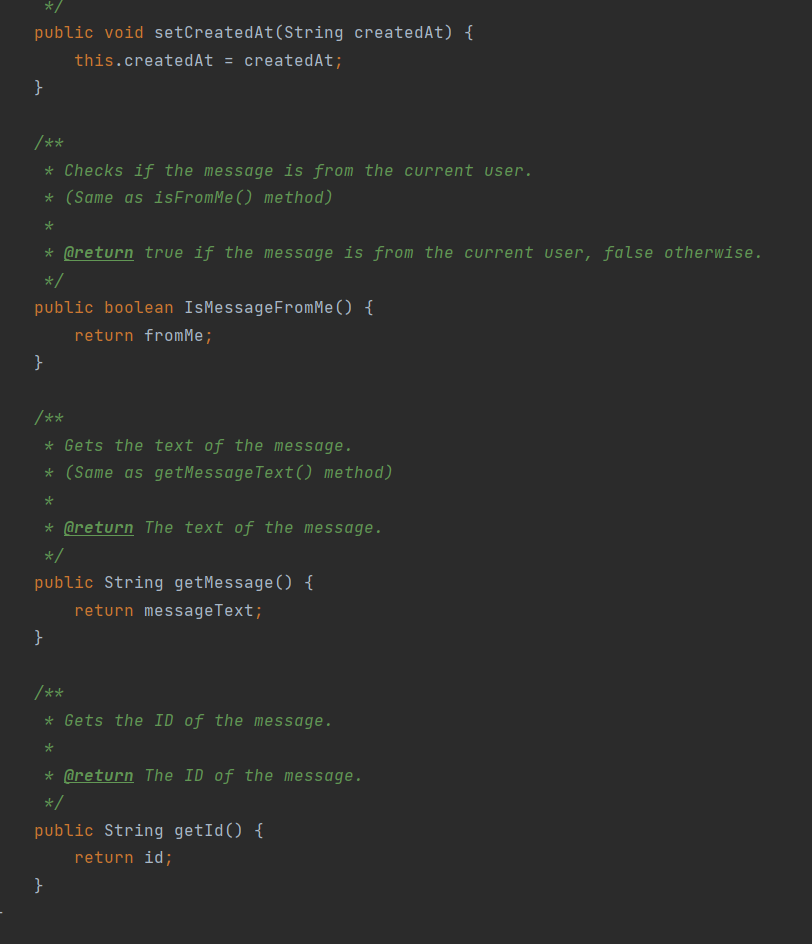
תכונת getBitmapFromURL – מוריד תמונה מכתובת אינטרנט כלשהי ומחזיר אותה בתור אובייקט של Bitmap.

תכונת generateAvatarFor – התכונה מקבלת שם של משתמש, תחילה היא בודקת אם יש לה תמונה שכבר שמורה לשם משתמש הזה אם כן היא מחזירה אותה ואם לא אז מורידים תמונה מהאתר dicebear שמתאימה לשם משתמש הזה ושומרים אותה אצלנו בhashmap ואז מחזירים את התמונה הזאתי.

### מחלקת UserMessage:

מחלקה זאת מייצגת הודעה של משתמש ומחזיקה את כל הפרטים של ההודעה.

### מחלקת MessageListAdapter:

המחלקה הזאתי מוסיפה הודעות של משתמשים שיופיעו במסך.

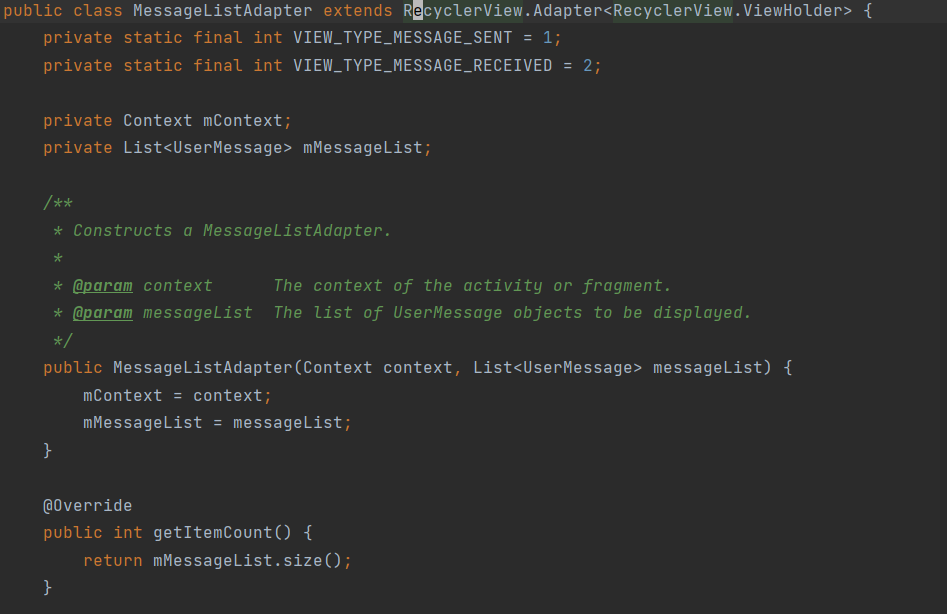
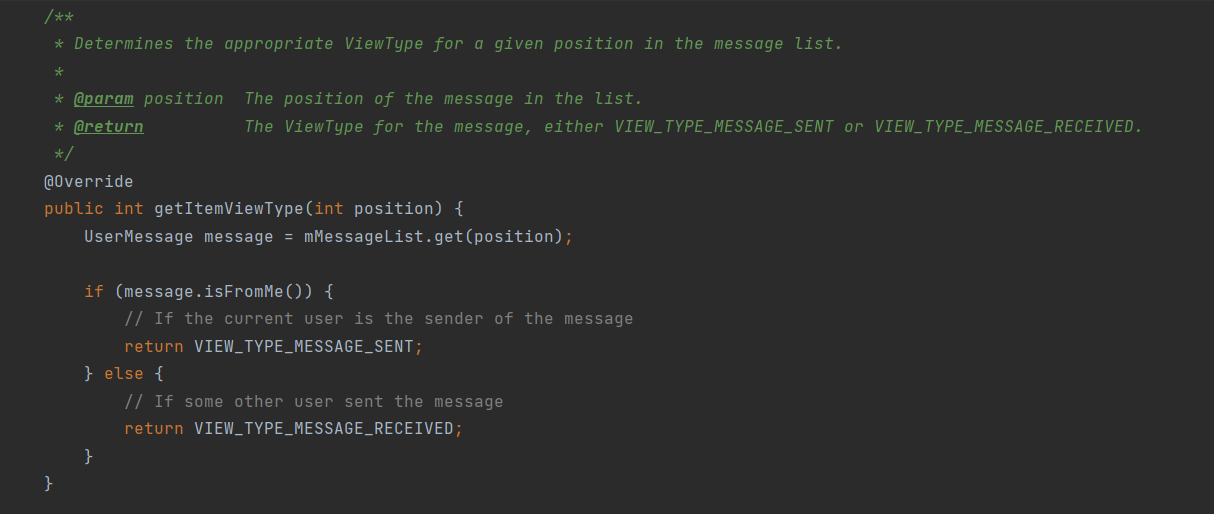
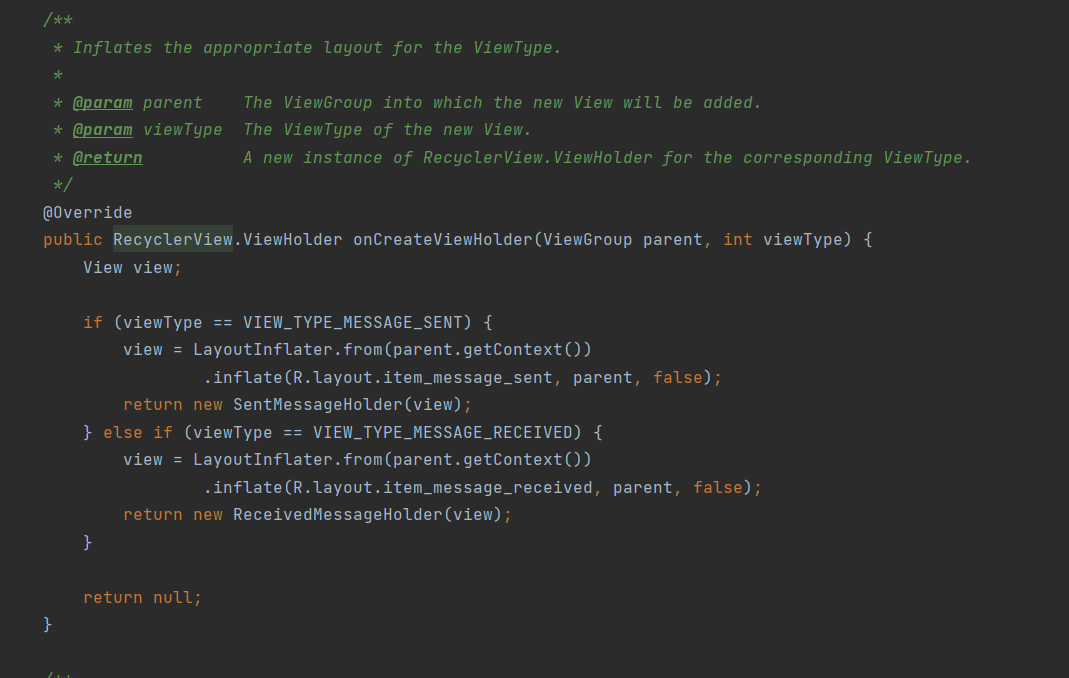
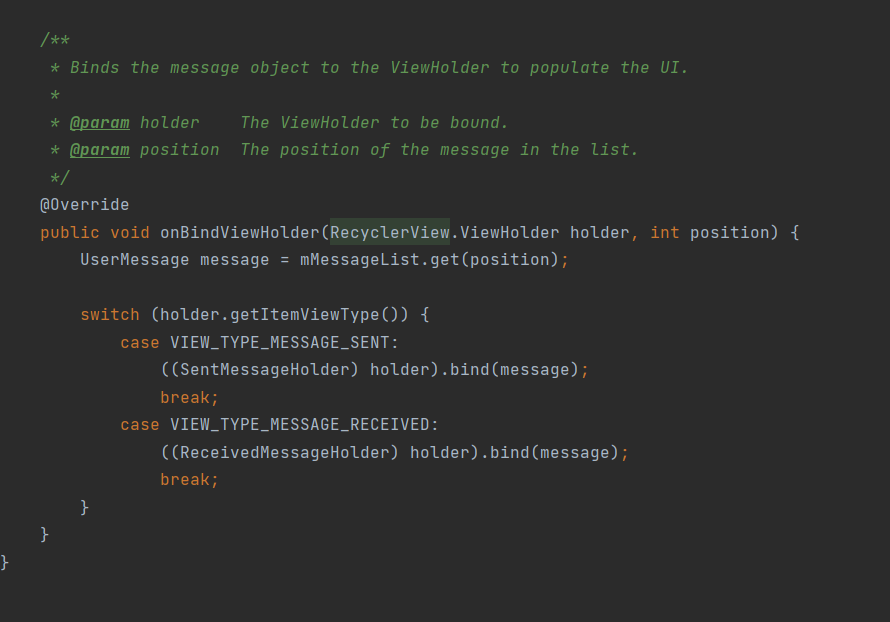
תכונות המחלקה:

תכונת getItemCount – מחזירה את כמות ההודעות שמוצגות.

תכונת getItemViewType – מחזירה את סוג ההודעה במקום כלשהו, אם הוא נשלח מאיתנו או נשלח בידי מישהו אחר.

תכונת onCreateViewHolder – בודקת את סוג ההודעה ויוצרת view חדש לפי זה.

תכונת onBindViewHolder – עושה bind לview שנוסף לפי הסוג הודעה שלו.

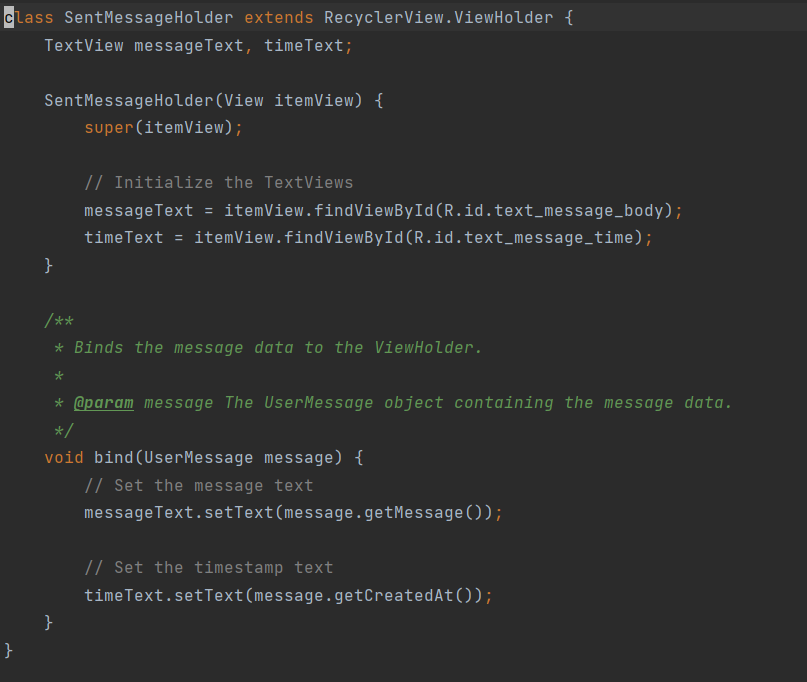
   

### מחלקת SentMessageHolder:

מחלקה זאתי מייצגת הודעה שנשלחה מאיתנו.

תכונות המחלקה:

תכונת bind – מציגה את המידע של ההודעה על המסך.



### מחלקת SetImageThread:

מחלקה שיוצרת thread שמשיג אוואטר להודעה ומשנה את האוואטר שההודעה מוצגת איתו לאוואטר שהוא השיג.

תכונות המחלקה:

תכונת run – משיגה תמונה לאוואטר ומשנה את ההודעה המוצגת שתשתמש בתמונה הזאתי.



### מחלקת ReceivedMessageHolder:

מחלקה זאתי מייצגת הודעה שנשלחה ממשתמשים אחרים.

תכונות המחלקה:

תכונת bind – מציגה את המידע של ההודעה על המסך ויוצרת thread של SetImageThread שייצור וישים אוואטר להודעה הזאתי.

