1. קיים שירות שמנהל קריאות API בשם UserService שמכיל פונקציה getUsers(). מהשרת (אין צורך לממש שרת ניתן לכתוב מוק דטה) ממש בהעלאת הקומפוננטה להצגת יוזרים קריאה של פונקציה זו לא מתוך הקומפוננטה ישירות ולאחר קבלת הדטה שמור בסטור.
2. צריך לנהל רשימת משתמשים ביישום שלך בעזרת NGRX. השתמש במבנה Entity (ספריה של NGRX ) כדי להגדיר State שמנהל משתמשים:
   1. צור את ה-Adapter המתאים עבור משתמשים.
   2. הוסף את ה-state בצורה שתתמוך ב-Entity Management
3. נהל הוספה ,מחיקה ,עדכון,שמירה של משתמשים בסטור.
   1. מנע הוספה של יוזרים כפולים עם אותו ID במידה ומדובר ביוזר קיים עדכן אותו במקום להוסיף חדש)
4.  מבנה ה-state: (ניתן להחליף צורה לעבודה עם SIGNALS )
   1. כתוב סלקטור שמחזיר את המשתמש הנבחר (selectedUser) על סמך ה-selectedUserId
   2. כתוב סלקטור שמחזיר את כל ההזמנות של המשתמש הנבחר.
   3. כתוב סלקטור המחזיר אובייקט שמכיל את שם המשתמש ואת סכום כל ההזמנות שלו.
5. צור קומפוננטה בשם UserOrdersComponent שמציגה את שם המשתמש הנבחר ואת סכום כל ההזמנות שלו. ודא שהקומפוננטה מאזינה לסלקטור שיצרת ומציגה את הנתונים בצורה דינמית.
   1. הוסף כפתורים בקומפוננטה לשינוי המשתמש הנבחר ב-state, כך שתוכל לבדוק שהקומפוננטה אכן מתעדכנת אוטומטית כאשר המשתמש הנבחר משתנה.
   2. בנוסף, חלק את הקומפוננטה לשתי קומפוננטות נפרדות: אחת שמציגה את שם המשתמש ואחת שמציגה את סכום ההזמנות. ודא ששתי הקומפוננטות מאזינות לסלקטור המתאים וממשיכות להתעדכן בצורה נכונה כאשר יש שינוי במצב ה-Store.
6. נניח שאתה מפתח תכונה במערכת שבה אתה צריך להאזין לשינויים במצב המשתמשים ביישום. כאשר משתמש מסוים נבחר (selectedUserId משתנה), אתה מבצע קריאת API כדי להביא פרטים נוספים על המשתמש. עליך להפסיק את קריאת ה-API הקודמת אם המשתמש הנבחר משתנה לפני שהקריאה הסתיימה. ממש תוך שימוש בסטור ובPIPES של RXJS .