**תיק למתכנת קבוצה מספר 4**

**תיאור הפרויקט:**

הפרויקט מתמקד בפיתוח פלטפורמה אינטראקטיבית לביצוע מבחנים אונליין, המאפשרת למשתמשים לבחור תשובות לשאלות רב-ברירתיות, לקבל משוב בזמן אמת ולראות את תוצאותיהם בסיום. הפלטפורמה בנויה בטכנולוגיות מודרניות המאפשרות חווית משתמש חלקה ואמינה.

**פונקציות בקובץ: `src/components/App.jsx`**

**1. `App`**

תיאור:

פונקציה ראשית שמנהלת את כל הניווט והצגת הקומפוננטות השונות באפליקציה. היא משתמשת ב-React Router כדי לנהל את הניווט בין המסכים השונים כמו מסך הכניסה, עריכת חידונים, יצירת חידונים, ורשימת החידונים.

טכנולוגיות בשימוש:

- ReactJS:לניהול רכיבי הממשק וניווט בין דפים.

- React Router:לניהול ניווט בין רכיבי האפליקציה.

**2. `handleLogin`**

תיאור:

פונקציה זו מתבצעת לאחר שהמשתמש נכנס למערכת. היא מאחסנת את המידע הנדרש בהקשר הגלובלי של האפליקציה ומעדכנת את מצב ההתחברות.

טכנולוגיות בשימוש:

- ReactJS: לניהול state והקשר גלובלי.

**3. `handleLogout`**

תיאור:

פונקציה זו מטפלת בהתנתקות של המשתמש מהאפליקציה. היא קוראת לפונקציות Firebase לצורך התנתקות ולאחר מכן מבצעת רענון לעמוד על מנת לעדכן את הממשק.

טכנולוגיות בשימוש:

- Firebase:לאימות משתמשים.

- ReactJS: לרענון הממשק לאחר ההתנתקות.

פונקציות בקובץ: `src/components/ChangeQuiz.jsx`

**4. `ChangeQuiz`**

תיאור:

פונקציה זו מנהלת את תהליך שינוי חידון קיים. היא מאפשרת למשתמש לבחור חידון ולבצע בו שינויים על ידי טעינת נתוני החידון ממסד הנתונים והצגת טופס לעריכה.

טכנולוגיות בשימוש:

- ReactJS: לניהול ממשק המשתמש.

- Firebase: לטעינת נתוני החידון לעריכה.

**5. `fetchQuiz`**

תיאור:

פונקציה זו משמשת לשליפת חידון ממסד הנתונים על פי מזהה החידון. היא מוודאת שהנתונים נטענים בהצלחה ושומרת אותם ב-state עבור המשך תהליך העריכה.

טכנולוגיות בשימוש:

- Firebase: לשליפת נתוני החידון ממסד הנתונים.

- ReactJS: לניהול ה-state של הנתונים.

**פונקציות בקובץ: `src/components/EditQuiz.jsx`**

**6. `EditQuiz`**

תיאור:

פונקציה זו מאפשרת למשתמש לערוך חידון קיים. היא מציגה את נתוני החידון בטופס עריכה ומאפשרת למשתמש לבצע שינויים ולשמור אותם בבסיס הנתונים.

טכנולוגיות בשימוש:

- ReactJS: לניהול ממשק העריכה ולשינויי state.

- Firebase: לשמירת השינויים בבסיס הנתונים.

**7. `handleEdit**`

תיאור:

פונקציה זו מנהלת את תהליך שמירת השינויים שנעשו בחידון. לאחר ביצוע השינויים, היא מאחסנת אותם במסד הנתונים ומעדכנת את הממשק.

טכנולוגיות בשימוש:

- ReactJS: לניהול ממשק המשתמש.

- Firebase: לשמירת הנתונים המעודכנים.

**פונקציות בקובץ: `src/components/QuizPlay.jsx`**

**8. `QuizPlay**`

תיאור:

פונקציה זו אחראית על ניהול חוויית המשחק של המשתמש. היא מציגה את השאלות מהחידון, מקבלת את תשובות המשתמש ומעדכנת את הממשק בהתאם לתוצאות.

טכנולוגיות בשימוש:

- ReactJS: לניהול state ותצוגת השאלות.

**9. `handleBackClick`**

תיאור:

פונקציה זו משמשת להחזרת המשתמש למסך הקודם, בדרך כלל לאחר סיום או יציאה מהחידון.

טכנולוגיות בשימוש:

- ReactJS: לניהול ניווט בממשק המשתמש.

**10. `handleNextQuestion`**

תיאור:

פונקציה זו מעבירה את המשתמש לשאלה הבאה בחידון ומעדכנת את הממשק בהתאם. היא מבטיחה שהשאלה הבאה נטענת נכון ומוצגת למשתמש.

טכנולוגיות בשימוש:

- ReactJS: לניהול state ותצוגת השאלות.

**11. `handleOptionClick`**

תיאור:

פונקציה זו מנהלת את התגובה ללחיצה על תשובה בחידון. היא מאחסנת את הבחירה של המשתמש ומתקדמת בהתאם ללוגיקה של המשחק.

טכנולוגיות בשימוש:

- ReactJS: לניהול state ואחסון התשובות הנבחרות.

**12. `closeModal`**

תיאור:

פונקציה זו סוגרת את החלון הקופץ שמופיע במהלך המשחק, בדרך כלל לאחר קבלת תשובה נכונה או שגויה.

טכנולוגיות בשימוש:

- ReactJS: לניהול ממשק המשתמש.

**פונקציות בקובץ: `src/components/QuizList.jsx`**

**13. `QuizList`**

תיאור:

פונקציה זו מציגה את רשימת החידונים הקיימים, כולל אפשרויות בחירה, עריכה או התחלה של חידון חדש.

טכנולוגיות בשימוש:

- ReactJS: לניהול תצוגת החידונים.

**14. `fetchHighScores`**

תיאור:

פונקציה זו אחראית על משיכת התוצאות הגבוהות של החידונים ממסד הנתונים והצגתן למשתמש.

טכנולוגיות בשימוש:

- Firebase: לשליפת התוצאות ממסד הנתונים.

- ReactJS: לניהול state ותצוגת התוצאות.

פונקציות בקובץ: `src/components/QuizForm.jsx`

**15. `QuizForm`**

תיאור:

פונקציה זו מנהלת את טופס יצירת החידון החדש. היא מאפשרת למשתמש להוסיף שאלות, תשובות, ולשמור את החידון במסד הנתונים.

טכנולוגיות בשימוש:

- ReactJS: לניהול תצוגת הטופס ושינויי state.

**16. `addQuestion`**

תיאור:

פונקציה זו מוסיפה שאלה חדשה לטופס החידון. היא מעדכנת את רשימת השאלות המוצגות למשתמש ומאפשרת עריכה של כל שאלה.

טכנולוגיות בשימוש:

- ReactJS: לניהול ה-state של השאלות בטופס.

**17. `removeQuestion`**

תיאור:

פונקציה זו מסירה שאלה מהחידון הנמצא בתהליך יצירה. היא מעדכנת את רשימת השאלות בהתאם לבחירת המשתמש.

טכנולוגיות בשימוש:

- ReactJS: לניהול ה-state של השאלות בטופס.

**18. `handleModalConfirm`**

תיאור:

פונקציה זו מנהלת את האישור של חלון קופץ שמופיע במהלך יצירת החידון, בדרך כלל לאחר הוספת שאלה או שמירת החידון.

טכנולוגיות בשימוש:

- ReactJS: לניהול ממשק המשתמש והחלון הקופץ.

**פונקציות בקובץ: `src/context/GlobalContext.jsx`**

**19. `GlobalProvider`**

תיאור:

פונקציה זו מספקת הקשר גלובלי לכל האפליקציה, המאפשר שיתוף מידע ומצבים בין רכיבים שונים באפליקציה.

טכנולוגיות בשימוש:

- ReactJS Context API: לניהול הקשר גלובלי ושיתוף נתונים.

**20. `useGlobal`**

תיאור:

פונקציה זו מאפשרת גישה להקשר הגלובלי שיצרנו ב-`GlobalProvider`. היא מאפשרת לכל רכיב באפליקציה לצרוך מידע מהקשר זה.

טכנולוגיות בשימוש:

- ReactJS Context API: לגישה להקשר הגלובלי באפליקציה.

Firebase: משמש לניהול מסד הנתונים שמאחסן את השאלות והתשובות, כולל אימות זהויות משתמשים ושמירת ההתקדמות.

* ReactJS -JSX: כלי פיתוח עיקרי לבניית הממשק הגרפי, הכולל יצירת קומפוננטות דינמיות וניהול מצב (state).
* ViteJS: משמש להאצת תהליך הפיתוח על ידי תמיכה ב-Hot Module Replacement והידור מהיר.

**רשימת מקורות ורפרנסים:**

* תיעוד Firebase לעבודה עם מסדי נתונים ואימות זהויות.
* מדריכים על פיתוח ReactJS ו-JSX לניהול קומפוננטות ומצב.
* מאמרים על שימוש יעיל ב-ViteJS להאצת תהליכי פיתוח.