שגיא יעקב

פרויקט פיתוח מערכת טריוויה

# פרויקט גמר 2021

תיאור כללי

מערכת טריוויה

המערכת המפותחת במסגרת הפרויקט הינה מערכת מבוססת שרת ולקוח.

המערכת כוללת שני מודי עבודה:

1. מוד תרומת שאלות למאגר שאלות ההמונים
   1. המערכת כוללת בסיס נתונים האוגר שאלות ממשתמשים שונים
   2. מבנה השאלות והתשובות – שאלון אמריקאי (סימון תשובה נכונה)
   3. מבנה התשובות גמיש – ניתן להכיל מספר תשובות שונה לכל שאלה
   4. תשובה נכונה הינה תשובה בודדת
2. מוד משחק
   1. בסיס נתונים של משתמשי המשחק
      1. כולל שיאים אישיים
      2. ציונים ממוצעים
   2. חישוב ציון לכל משחק ורישום בבסיס הנתונים של המשתמשים

רקע

חיפשתי נושא בסיסי שמאפשר להרחיבו לאיזורים שונים באופן גמיש.

אפשר לממש משחק טריוויה באופן פשוט ועצמאי ומשם ניתן להרחיב את הפרויקט לתחומים שונים:

* שילוב שרת וקליינט
* שילוב בסיסי נתונים
* עטיפה כאפליקציית WEB
* דוחות ( משתמשים , ציונים , ממוצעים , הישגים ..)
* שילוב חכמת המונים – הפיכת המשחק לכזה שמאפשר גידול אינסופי של מגוון השאלות – ע"H כל משתמשי האפליקציה...
* שילוב שפות שונות ( HTML , CSS , JAVASCRIPT ועוד ...)

לצורך הפרויקט נדרשתי ללמוד סביבות פיתוח ושפות שונות כולל שימוש בבסיס נתונים, ליצור שרת אינטרנט והתקשורת איתו, JAVASCRIPT , שרת NODE JS , כלי ממשק משתמש ( כמו ספריית VUE JS ) ועוד...

מרכיבים שונים בפרויקט התגלו ככאלו שלא היה לי בהם כל ניסיון או ידע מקדים. כך שחלק גדול מתהליך הפיתוח כלל לימוד כלים חדשים.

[ עם המשך הפרויקט אסיים את תיאור הרקע ... ]

תרשים תהליכי הפרויקט:

כניסה/רישום משתמש

בסיס נתונים שאלות ותשובות

בסיס נתונים משתמשים

תוצאות / הרשאות

תרומת שאלות

משחק

מבנה מרכיבי התוכנה:

שרת ה NODE JS

הקמתי שרת Node JS שכרגע משמש ל 3 מטרות:

1. החצנה של ספריית PUBLIC ל STATIC FILES  
   קישור index.html ל localhost:3000.
2. API לבסיס הנתונים  
   כרגע מימשתי את פונקציית GET QUESTION (לצורך הצגת השאלות למשתמש) , בהמשך אממש את פונקציית ה SET QUESTION ( לאפשר הוספת שאלות למערך)
3. מחצין API-ים ( get + post ) המאפשרים קריאה מה FRONT END (Index.html) לשרת.  
   כרגע ממשתי רק API ל getquestion שמגיע לשרת , מפעיל את הפונקציה של getquestion בבסיס הנתונים שמחזיר את פרטי השאלה לקליינט.

הסבר מפורט :

1. בקובץ server.js בשורת : app.use('/', express.static(\_\_dirname + '/public'));   
   ניתן לראות שכניסה ל ROOT "/" מריצה את index.html שנמצא בספריית PUBLIC.

const express = require('express')

const app = express()

const port = 3000

const db = require('./server/db');

const run = async () => {

    await db.connect();

    app.use('/', express.static(\_\_dirname + '/public'));

    app.get('/api/questions', async (req, res) => {

        const questions = await db.getQuestions();

        res.send({questions})

    })

    app.listen(port, () => {

      console.log(`Example app listening at ${port}`);

    });

}

run();

חיבור למסד הנתונים. שימוש בקובץ MONGOOSE .   
הגדרת מודלים. כרגע הגדרתי מודל אחד (בקובץ db.js ) : QUESTIONS בלבד שכולל:

Question = טקסט לשאלה

Answers = מערך תשובות

Correct = התשובה הנכונה

QuestionModel = mongoose.model('Question', new mongoose.Schema({

        question: String,

        answers: [],

        correct: Number

בהמשך אוסיף פרמטר SELECT שישמש לאגירת התשובה שנבחרה ע"י המשתמש במשחק.

בקובץ ה db.js יצרתי בינתיים שני מודולים נוספים:

Getquestion (לקריאת שאלה ממסד הנתונים ) ו addquestion (להוספת שאלה למסד הנתונים) :

addQuestion: async question => {

const newQuestion = new QuestionModel();

newQuestion.question = question.question;

newQuestion.answers = question.answers;

newQuestion.correct = question.correct;

newQuestion.save()

},

getQuestions: async () => {

const res = await QuestionModel.find({});

return res;

},

ספריית ה PUBLIC:

קובץ ה index.html :

בחרתי להשתמש בסיפריית UI שנקראת : VUE JS .   
הסיבה: מצאתי שהוא הכי קל לשימוש ולא מצריך קומפילציה ... כלים אחרים כמו REACT ו ANGULAR מצריכים קומפילציה. ולכן קל יותר לתפעול.

<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>

בקובץ index.html יש קישור לקובץ index.js שבו מוגדרות ה JAVASCRIPT ששולט במה שרואים על המסך.

בנוסף אלמנטים שונים.

Question – טקסט ( מאופיין כאלמנט של VUE JS )

Answer - מערך בגודל משתנה - לולאת for :

 <ul id="answers">

            <li v-for="answer in answers" :key="answer">

                {{ answer }}

            </li>

        </ul>

* כרגע אני עובד על הפיכת ה ANSWER למצב שיאפשר בחירת תשובה ( SELECT) באמצעות מודול מסוג RADIO BUTTON – בתהליך.

לחצני הניווט:

Index.html כולל גם את הגדרת לחצני הניווט בין השאלות :

<div id="navigation">

            <button v-on:click="prevQuestion" v-bind:disabled='isPrevDisabled'>prev</button>

            <button v-on:click="nextQuestion" v-bind:disabled='isNextDisabled'>next</button>

        </div>

כפתור NEXT וכפתור PREV.

חיברתי פונקציית DISABLE לכל לחצן לצורך התאמה בתחילת ובסוף רשימת השאלות (האפרת הכפתור PREV בשאלה הראשונה והאפרת כפתור NEXT בהצגת השאלה האחרונה)

VUE אפשר לקשור את ה ATTRIBUTE כפונקציה כדי שאוכל להפעיל את הפונקציה בהתאם לשינוי הנדרש. (סימון כ COMPUTED המאפשר האזה לשינוי ב DATABASE והתאמת ה ATTRIBUTE ל TRUE או FALSE בהתאם )