**איפיון פרויקט – משחק בול פגיעה (Bull & Pgia)**

**הגשה עד י"ג סיוון מזכה ב5 נקודות בונוס.**

**הגשה אחרונה: כ"א סיוון, אין הגשה לאחר תאריך זה.**

**אין לעשות את הפרויקט ביותר מזוג בנות.**

**📝 מטרת הפרויקט:**

בניית משחק בול פגיעה מול המחשב. לכל שחקן נוצר קוד סודי רנדומלי, והוא מנסה לנחש אותו. כל ניחוש מחזיר בול (ניחוש מדויק במקום הנכון) ופגיעה (ניחוש של ספרה נכונה במקום שגוי).

**🧩 מטרות לימודיות:**

* עבודה עם Express + TypeScript.
* חיבור ל-MongoDB ושימוש ב־Mongoose.
* ניהול לוגיקה במשחק אמיתי.
* תרגול מחלקות, שירותים ו־API לפי תחום (features).
* עבודה עם Middlewares וטיפוסים מותאמים.
* קריאות API מסודרות.

**🗂 מבנה תיקיות מוצע:**

project-root/

│**node\_modules <- לא להגשה**

├── src/

│ ├── app.ts

│ ├── db/

│ │ └── connection.ts

│ ├── games/ ← תכונה: משחקים

│ │ ├── game.controller.ts

│ │ ├── game.service.ts

│ │ ├── game.model.ts

│ │ ├── game.logic.ts ← מחלקת BullPgia

│ ├── players/ ← שחקנים

│ │ ├── player.controller.ts

│ │ ├── player.service.ts

│ │ └── player.model.ts

│ └── middleware/← יש להוסיף לפי הצורך

│ └── validateGame.ts

│ └── validateParameters.ts

│ ├── server.ts

├── package.json

└── tsconfig.json

**📘 תיאור המשחק:**

* שחקן נכנס למערכת בעזרת שם וסיסמא
* לכל שחקן נוצר קוד סודי רנדומלי של 4 ספרות, מ־1 עד 9 (ללא חזרות).
* המשתמש שולח ניחוש – מערך מספרים באותו אורך.
* כל תגובה מכילה:
  + bulls: מספר ספרות נכונות במקום הנכון.
  + pgias: מספר ספרות נכונות במקום הלא נכון.
* לכל משחק מוקצב מספר ניסיונות (למשל 10).
* המערכת שומרת את היסטוריית הניחושים והתוצאה.
* כל התוצאות נרשמות ב־MongoDB.

**🧠 קריאות API**

|  | **פעולה** | **ערך חוזר** | **שיטה** | **כתובת** | **תיאור** |
| --- | --- | --- | --- | --- | --- |
|  | יצירת משחק | GAME\_ID | POST | /api/games/start | יצירת משחק חדש עם קוד סודי –  יש לשלוח שם משתמש וסיסמא בBODY |
|  | שליחת ניחוש | JSON | POST | /api/games/:gameId/guess | שליחת ניחוש ומקבל בול/פגיעה-  שליחת הערך BODY |
|  | סטטוס משחק | JSON [] | GET | /api/games/:gameId | מחזיר את מצב המשחק וניחושים קודמים |
|  | סיום משחק | STRING | POST | /api/games/:gameId/end | מסיים את המשחק מוקדם (למשל ויתור) |
|  | תוצאות אחרונות | JSON [] | GET | /api/players/:playerid/recent | תוצאות אחרונות לשחקן |
|  | דירוג שחקנים | JSON [] | GET | /api/players/leaderboard | 10 הניצחונות המהירים ביותר |
|  | הוספת שחקן | PLAYER | POST | /api/players/add | הוספת שחקן לרשימת שחקים |
|  | עריכת שחקן | PLAYER | PUT | /api/players/:playerid | עריכת שחקן – שינוי פרטים |
|  | מחיקת שחקן | PLAYER | DELETE | /api/players/: playerid | מחיקת שחקן |
|  | הצגת שחקן | PLAYER+ | GET | /api/players/: playerid | הצגת שחקן- עם כל הנתונים כולל המשחקים שלו |
|  |

**🧱 מבני נתונים:**

**🎲 סכמת משחק (Game)**

{

\_id: ObjectId,

playerId: ObjectId,

secretCode: number[], // למשל: [4, 2, 8, 1]

attempts: [

{

guess: number[], // למשל: [4, 3, 2, 9]

bulls: number,

pgias: number,

createdAt: Date

}

],

status: 'in-progress' | 'won' | 'lost' | 'ended',

maxAttempts: number,

winner: boolean, // האם הצליח לנחש

createdAt: Date

}

**🧍 שחקן (Player) –**

{

\_id: ObjectId,

name: string,

password: string,

mail: string,

totalGames: number,

wins: number

}

**🌐 דוגמת קריאת API – שליחת ניחוש:**

POST /api/games/64e8a1234567/guess

Content-Type: application/json

{

"guess": [4, 3, 2, 9]

}

תגובה:{

"bulls": 1,

"pgias": 2,

"remainingAttempts": 6,

"status": "in-progress"

}

**🛡 הוספת middleware:**

יש להוסיף middlewares עבור בדיקה שהערכים המגיעים הם תקינים,

לכל הפחות:

1. בדיקת ניחוש
   1. שהוא רק מספרים
   2. שאינו מכיל מספר יותר מפעם אחת
   3. אורך 4
2. בדיקת משתני שחקן
   1. מייל
   2. סיסמא

**POSTMAN**

חובה להוסיף קובץ POSTMAN המכיל הוספה של לפחות 2 שחקנים, עריכה, מחיקה, והצגה

ושימוש במשחק כמה פעמים, כולל הצגות, ובקשות שנכשלות בגלל בדיקות בmiddleware

יש לקרוא לcollection בשם של המגישות  
אנא מספרו את הבקשות כך אוכל להפעיל לפי הסדר שהתכוונתם (לדוגמא 1\_adduser, 2\_edit, 3\_remove)