פרויקט - סביבות פיתוח באינטרנט - עבודה 3.2 - פיתוח Backend

בחלק זה של העבודה תפתחו את השרת ובסיס הנתונים, על פי ה-API אותו כתבתם והאפיון שקבלתם.

להלן דגשים שיסייעו לכם במימוש העבודה:

- בסיס הנתונים יהיה בשירות הענן ב-Azure כפי שהוצג לכם בתרגולים.
  - תחת מטלה 3.2 באתר הקורס, מופיע מסמך הסבר לפתיחת חשבון במידה וטרם פתחתם.
  - כמו כן, להלן תוסף ל-VSCODE אשר יאפשר לכם לעבוד בקלות מול ה-DB ב-AZURE:
    - https://marketplace.visualstudio.com/items?itemName=ms-mssql.mssql
    - https://docs.microsoft.com/en-us/sql/visual-studio-code/sql-server-develop-use-vscode?view=sql-server-ver15

- מידע בו תעשו שימוש בשרת:
  מקורות המידע של השרת שלכם נחלקים ל-2:
  1. שימוש ב-API חיצוני: https://docs.sportmonks.com/football
  2. שימוש בבסיס הנתונים שלכם.

שימוש בAPI חיצוני:
כל הנוגע למידע על הליגה/ הקבוצות/ השחקנים וכו' יעשה באמצעות תשאול של הAPI (דרך ה-collection שהורדתם על גבי ה-postman).
לנוחיותכם, להלן המשאבים הרלוונטיים לכם:

- Leagues endpoints
- Teams endpoints
- Players endpoints
- Coaches endpoint
- Team squads endpoints

1. לפני שאתם מתחילים לעבוד מול ה-API, הבינו איזה מידע נדרש מכם להביא, ולאחר מכן, קראו את המידע אותו מספק ה-API. שימו לב להשתמש בפרמטרים אותו ה-API מספק כך שתקבלו מידע שרלוונטי עבור הדרישות שלכם (לדוגמה: שימוש ב-Includes).
2. צאו מנקודת הנחה כי השרת יכול לתשאל תמיד את ה-API, ואין צורך לשמור מידע מיותר ב-DB
3. שימו לב, ישנה מגבלה יומית מסוימת של תשאול ה-API, מגבלה זו הינה גדולה, ולא צריכה להיות בעיה לשלוח בקשות רבות ביום.
   עם זאת, הדרך הכי פשוטה להתגבר על זה, היא לאחר שבדקתם שהקוד שלכם עובד, שמרו חלק מהמידע שחזר לכם, והשתמשו בו כמידע סטטי באפליקציה לצורך הפיתוח (בלבד).
   כלומר - שמרו את התשובה מהשרת במשתנה ב-route handler, והחזירו אותו כתשובה מיד כאשר מתקבלת כשמתקבלת בקשה, מבלי לבצע פעולת נוספות.

שימוש בבסיס הנתונים:
כל הנוגע למידע **אישי** על המועדפים של המשתמש (קבוצות/שחקנים/משחקים וכו') וכן כל המידע על אודות משחקי הקבוצות השונות בליגה ינוהל בבסיס הנתונים.

הנחיות כלליות לעבודה:

1. עליכם להקפיד על כללי הפיתוח והדגשים שנלמדו לפיתוח צד שרת בהרצאות ובתרגולים.
2. עליכם להשתמש בספריות שנלמדו בקורס בלבד - כל ספריה נוספת, נא לאשר מול הסגל.
3. **הקלה לעבודה - אין צורך לעשות בדיקות קלט בהרשמה של משתמשים בצד שרת.**
   תהיו חייבים לעשות זאת רק בצד לקוח.
4. עליכם להקפיד על כתיבת קוד נכונה ונקייה:
   - 💥 הפרדה למודולים (routes and utils).
   - 💥 הפרדה לפונקציות - לוגיקה מורכבת צריכה להיות בפונקציות, אם צריך אז לפצל למספר פונקציות - ככלל, פונקציה מוגדרת כקטע קוד המבצע משימה אחת, מעבר לכך - יש לפצל לפונקציה נוספת
   - 💥 שמות פונקציות ומשתנים משמעותיים - שמות ארוכים זה מצוין, חשבו על השמות כ"סיפור" אותו הקוד שלכם מספר, או כפסאודו קוד. מי שקורא את הקוד שלכם יכול להבין בדיוק מה הוא עושה רק מקריאת שמות הפונקציות והמשתנים.וכד'.
5. הגשת העבודה תעשה באותם זוגות שהתחילו את הפרויקט.
6. עליכם לממש את השרת וה-routes שלו, על פי ה-API אותו כתבתם.
   - 💥 עבור כל משאב:
     i. השרת יקבל בקשה, כפי שמופיע ב-API.
     ii. השרת יחזיר תשובה, כפי שמופיע ב-API.
   - 💥 במידה וברצונכם לשנות את המימוש, אין בעיה, כל זמן שאתם דואגים שמסמך ה-API תואם למימוש שלכם.
   - 💥 כדי להבטיח את התאימות בין ה-API לבין המימוש בשרת, ובכדי לסייע בעבודת הפיתוח על ידי "חיקוי" (mock) של שרת, מומלץ בחום לעבוד עם Swagger UI כפי שהוצג במעבדה 9 (הקבצים הרלוונטים מצורפים ל-template הראשוני שקיבלתם).
7. שאלות על המטלה - ניתן לשאול בפורום הייעודי של עבודה 3.2.
8. המלצה חמה :
   - 💥 חלקו את המימוש ביניכם, חלוקה טובה תהיה מימוש של routes שונים. ושתפו את הקוד ב-git יחד בחלוקה ל-branches שונים ומיזוגם בצורה מסודרת באמצעות PR (pull request).
   - 💥 ביצוע commit לgit של כל קטע קוד עובד שפיתחתם, תבטיח שלא תקרנה טעויות ותאבדו/תהרסו קוד עובד. לא לשכוח לעשות push לאחר סשן עבודה שלכם על מנת שהקוד המעודכן שלכם ישב בgithub.
     סרטון עזרה למי שלא מרגיש בנוח:
     https://www.youtube.com/watch?v=_OZVJpLHUaI

הגשה:

1. העבודה תוגש ב-GIT (שם תוכלו למצוא את ה-template הראשוני):
https://classroom.github.com/g/iKFwqro7
2. שימוש לב, כי גם בעבודה זו כמו בכל הפרויקט, הקוד ייבדק במערכת ההעתקות.
3. תאריך הגשה: **3.6 בחצות.**

בהצלחה!!

סגל "סביבות פיתוח באינטרנט" 2021
שיר, ארז, הילה, נאור, קובי ודביר.