חידת נתיב הזהב

מבוא לביצוע חידת נתיב הזהב

1. חידה זו נועדה לבחון את יכולתכם להביא לכדי ביטוי אופן מחשבה מתאים לפיתוח מערכות תוכנה, וכן לאפשר לכם לקבל הצצה לאופי העבודה של הצוות.
2. החידה הינה מטלת קוד, אך שמה דגש על **אופן חשיבה** ולא על **קוד מדויק**. אלו שאינם בקיאים באופן מוחלט באופן כתיבת הקוד **חייבים** לענות על הסעיפים באופן מילולי או גרפי אשר **מבטא את התהליך החשיבתי**.
3. אתם לא בהכרח תכירו את כלל המושגים – מצופה מכם ללמוד באופן עצמאי!
4. ניתן יהיה לשלב בין מענה קוד למענה מילולי.
5. בהתאם לסעיף 2, במידה ומצורף קובץ טקסט המבטא את התהליך החשיבתי עליו להיות בסיומת md. (לצורך העלאה ל-GitHub Repository).
6. הגשת המשימה תתבצע באמצעות העלאת הקבצים כ-Repository פרטי ב-GitHub שלכם ולאפשר גישה למשתמש iafgoldenroute או למייל IAF.goldenroute@gmail.com.

   מי שאינו בקיא בשימוש ב-GitHub נדרש ללמוד זאת. מצורף בזאת סרטון YouTube אשר יהווה התחלה טובה.
7. עליכם לכתוב הוראות מדויקות לאופן ההרצה של הפרויקט על מנת שנוכל לבדוק את הפרויקט שלכם (גם אם אתם משתמשים ב-Docker).
8. המון בהצלחה!

רקע מודיעיני

1. בשלהי 2022 התקבל מודיעין על מפעל ייצור אמל"ח מתקדם במדינת אויב מרוחקת, בו מורכב נשק אויב סודי ביותר.
2. הרמטכ"ל רב-אלוף שכטר הורה ליחידת הקומנדו סיירת ברקוני לבצע מבצע פשיטה בעומק כדי לגנוב את האמל"ח המסווג לצורך מחקר נגד.
3. היחידה תחדור למדינה זו באופן רגלי.
4. בשל אופי המשימה היחידה תחזור עם מטען אשר מצריך שימוש במטוס תובלה כבד מסוג 'שמשון' כדי לחזור ארצה עם המטען.
5. משקל המטען אינו ידוע. לכן, היחידה תשמיד לפני העלייה למטוס חלק מהשלל המהווה משקל עודף.
6. היחידה יכולה לשנע את המטען למיקום ההמראה.

7. הוטל עלינו, צוות נתיב הזהב, לפתח מערכת מבצעית המחשבת ומציגה את מרחק ההמראה וזמן ההמראה.
8. על זמן ההמראה להיות לכל היותר 60 שניות על מנת לעמוד בלו"ז המשימה.
9. המסה של המטוס עם חברי היחידה וצוות האוויר הינו 35,000 ק"ג (ללא המטען הכבד).

<u>שלב 1 – לוגיקה עסקית</u>

1. השלב הראשון שנממש יהיה 'מחשבון פיזיקה' של מטוס ה'שמשון'.
2. לצורך פיתוח מהיר ובצעדים קטנים, נניח כי המשוואה בה ניתן לחשב את הנתונים הנדרשים מהדרישה המבצעית היא *משוואת תנועה שוות תאוצה* (ראה תיאור בהמשך), ממנה ניתן לתאר את מרחק וזמן ההמראה.
   א. מטוס שמשון ממריא במהירות 140 מטר לשנייה.
   ב. המנועים של מטוס שמשון מפיקים כוח בגודל השווה ל-100,000 ניוטון.
   ג. נתעלם מהשפעות הרוח (חיכוך).
3. **תנועה שוות תאוצה**:
   א. את התאוצה (המסומנת באות a) ניתן לקבל מהקשר:
   $$a = \frac{F}{m}$$
   כאשר F הינו כוח המנועים ו-m הינה המסה של המטוס כולל הצוות <u>והמטען</u>.
   ב. את המהירות (המסומנת באות V) ניתן לקבל מהקשר:
   $$V = a \cdot t$$
   כאשר a היא התאוצה ו-t הוא הזמן. מקשר זה (ומתוך מציאת התאוצה וידיעת מהירות ההמראה) ניתן לחלץ את הזמן הדרוש למטוס להמריא.
   ג. את מרחק ההמראה ניתן למצוא מתוך הקשר:
   $$X = 0.5 \cdot a \cdot t^2 + V_0 \cdot t + x_0$$
   כאשר a הינה התאוצה, V_0 הינה המהירות ההתחלתית, t הינו הזמן ו- x_0 הינו המיקום ההתחלתי (במקרה שלנו $V_0, x_0 = 0$).
4. עליכם לממש שירות אשר מבצע את חישובי הפיזיקה המתאימים. **מומלץ לקרוא את השלבים הבאים כדי להבין מהו אופן המימוש המיטבי של מחשבון פיזיקלי זה**.

<u>קלט:</u>

א. המסה של המטען הכבד.

פלט:

א. מרחק המראה.

ב. זמן ההמראה.

ג. במידה וזמן ההמראה הינו מעל 60 שניות, מהו משקל המטען העודף אותו יש להשמיד כדי לעמוד בלו"ז המשימה.

5. **שאלה:** איך נוכל לוודא כי השירות שבנינו מוסר ללקוח תשובות נכונות? אילו מקרי קצה עלולים לצוץ בעת שימוש במערכת? כיצד נתמודד איתם?
6. **שאלת בונוס:** המודל הפיזיקלי שאנחנו מממשים כאן לא מתאר בצורה מדויקת לחלוטין התנהגות מציאותית. כיצד נשפר את המודל כך שיהיה מדויק יותר? אילו תנאי סביבה של המציאות ניתן יהיה לדעתכם לממש במודל?

שלב 2 – צד שרת

1. לאחר שהצגתם את המחשבון הפיזיקלי לרמטכ"ל (שכטר) זכיתם בפרס בטחון ישראל, והוחלט כי מחשבון זה יהיה חלק מאפליקציה גדולה.
2. **דרישה** - עליכם לממש שירות HTTP אשר מאפשר גישה למחשבון הפיזיקלי שמומש בסעיף הקודם. ניתן לכתוב את השירות בכל שפת תכנות ו-Framework שתבחרו (Spring, Django, FastAPI, Flask, NestJS, Express, Node.js ועוד).
3. **בונוס** – ישנו סיכוי כי יהיו מבצעים דומים בעתיד. לכן, על המערכת לשמור את תוצאות החישוב במסד נתונים (SQL/No-SQL) על מנת לאפשר יעילות בחישובים הבאים. על מסד הנתונים לשמור:
   א. משקלו של המטען הכבד.
   ב. מרחק המראה מינימלי.
   ג. משקל המטען הכבד אותו יש להשמיד.
   ד. זמן המראה.
4. **דרישה** - עליכם להתייחס למקרה בו המערכת מקבלת קלט לא צפוי/שגוי.

שלב 3 – צד לקוח

1. לאחר יצירת המחשבון הפיזיקלי, נרצה להציג את תוצאותיו ללוחמים באמצעות ממשק משתמש נוח ופשוט.
2. בשלב זה נקים 'צד לקוח' אשר מאפשר להזין את הקלט ובעקבותיו להציג את הפלט של המחשבון הפיזיקלי.

3. **דרישה** - ממשו ממשק משתמש אשר יאפשר שימוש ויזואלי במערכת. הממשק נדרש להציג את היכולת להכניס את הקלט ולהציג את הפלט של המחשבון הפיזיקלי. יש לממש את הממשק באמצעות React, Angular, Vue.
4. יש לשים לב כי בשלבים הבאים יתכנו דרישות להציג נתונים נוספים בצד הלקוח.

שלב 4 – גישה ל-API חיצוני

1. בתדריך לקראת המבצע חיל האוויר ציין כי מטוס תובלה מסוג 'שמשון' יכול להמריא רק כאשר הטמפרטורה במקום ההמראה היא בין 15°-30° צלזיוס.
2. עליכם לגשת מהשרת שכתבתם ל-Weather API לבחירתכם ולבדוק האם ניתן לבצע את המשימה בתאריך 1/1/23 ובקורדינטות latitude: 30 longitude: 35.
   * מומלץ להשתמש ב https://open-meteo.com/
3. יש להציג ללקוח האם ניתן לבצע את המשימה בתאריך ובמיקום שנבחר, ובאיזה שעות ניתן להמריא. במידה ולא ניתן לבצע את המשימה, יש להציג הודעת שגיאה הכוללת את הטמפרטורה במקום ההמראה.
4. **בונוס** – האם לדעתכם קיימים נתונים נוספים אשר מומלץ להציג ללקוח?
5. **בונוס** – אפשרו ללקוח לבחור את התאריך דרך ממשק המשתמש.

שלב 5 – תיאור המערכת

1. **דרישה** - עליכם ליצור שרטוט המסביר את זרימת המידע של המערכת.
2. **דרישה** - עליכם לתעד כל שירות שיצרתם, ולהסביר מה הקלט והפלט שלו.

בונוס - שלב 6 – פריסה (deployment)

1. **דרישה** - על מנת לאפשר פריסה של המערכת בכל מקום, השתמשו ב-Docker על מנת 'לארוז' את המערכת (יאפשר לנו לבדוק את הפרויקט במהירות).
2. **דרישה** – יש להוסיף Docker Compose

בונוס - שלב 7 – שרידות המערכת בעת מלחמה

1. **שאלה** - מהם לדעתכם סיכונים לשרידות של מערכות מבצעיות?
2. **שאלה** – אילו טכנולוגיות ימזערו את הסיכון של המערכת?

בהצלחה!