**נתיב הזהב**

**מגיש – בן טאו**

**שלב 1:**

**5)** בכדי לוודא שהשירות שבנינו מוסר ללקוח תשובות נכונות נוודא שהקוד שכתבנו אכן תואם את הרעיון בשלב התכנון, נבדוק האם המערכת מוציאה ערכים לא הגיוניים (לדוגמה במשימה הזאת מספר שלילי) ונבדוק ידנית עבור 2-3 ערכים אם המערכת אכן מצליחה "לפגוע" בתשובה הנכונה. במערכת עלולים לצוץ מקרי קצה כאשר המשתמש יזין קלט אשר לא תואם את דרישות המערכת כגון מספרים שלילים או אותיות או ישלח שדה ריק. בכדי להתמודד עם בעיות אלו השתמשתי בtry and except אשר לא ייתן למערכת לקרוס מצד אחד ומאידך ייתן ללקוח פידבק על הערכים שהוא הכניס כדי שיתקן להבא. בנוסף, צמצמתי כמה שיותר את האפשרות של הלקוח להכניס ערכים שגויים על ידי הפיכת חריץ הקלט למספרים בלבד ללא אפשרות של שליחת ערכים ריקים.

6**)** המודל הפיזיקלי הנוכחי מזניח כוחות חשובים כגון חיכוך עם הרוח (גודל וכיוון) ותנאי מזג האוויר, טמפ' סביבה ובזבוז של דלק שמקל על המטוס להמריא. ניתן להוסיף את שלושת הפרמטרים למודל, חיכון עם הרוח ניתן להוסיף על ידי הוספת API חיצוני של גודל וכיוון הרוח ולחשב את זה עם התאוצה (כמובן שיש צורך להתחשב גם בשטח הפנים של המטוס ובעוצמת החיכוך), טמפ' הסביבה על ידי API חיצוני, ובזבוז דלק ניתן לחשב על ידי בזבוז הדלק כל שניה ולהוסיף למודל שיהיה מדוייק יותר.

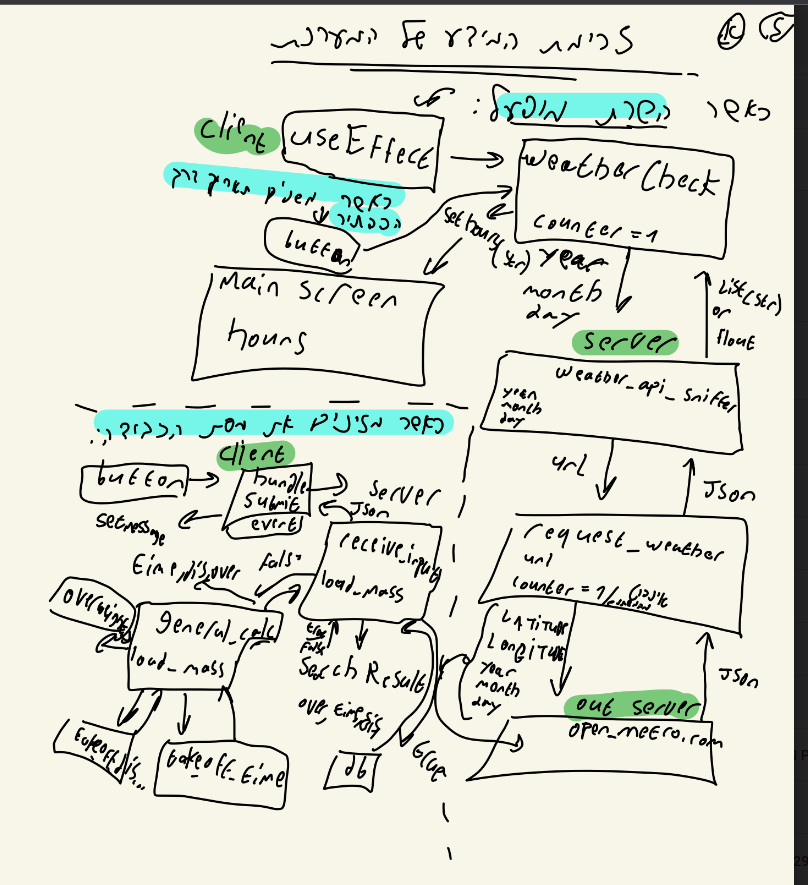
**שלב 4:**

**4)** מומלץ להציג ללקוח את מזג האויר במקום (משקעים, ראות וכו) ורשימה של כל התאריכים בהם ניתן לקיים את המבצע בכדי שלא יצטרכו לחפש אחד אחד (אפילו ניתן לעשות חיפוש לפי שעה).

**שלב 7:**

1. סיכונים לשרידות של מערכות מבצעיות הינם הצפה של הסרברים על ידי גורמים חיצוניים ופנימיים, נפילת חשמל כללית של המחשבים או הסרברים על ידי התקפת אויב, ופגימה ביכולת לקבל מידע מהסרבר החיצוני (של המזג האויר) על ידי הספמתו מגורמים עוינים.
2. ניתן למזער את הסיכון של המערכת על ידי אישור בשרת רק כתובות IP ספציפיות לשימוש בשרת ולבדיקה תדירה שאין שימוש חריג בו. ובנוסף יש צורך בחיבור הסרבר הראשי לגנרטור גיבוי למקרה של הפסקת חשמל בכדי שהמערכת לא תקרוס**.**

שלב 5:

1) 

**2)**

**Server.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **מטרה** | **output** | **input** | **name** |
| **לטפל בזה שאם הסרבר החיצוני לא מגיב זה לא יקריס לי את הסרבר** | **Error message or weather result** | **url – the url of the server.**  **counter – counter for break the recursion.** | requests\_weather |
| **לשמור את הנתונים בדטא בייס** | **אין** | **Result – physics calc result**  **Load\_mass – המטען הכבד.** | db\_save |
| **להחזיר ללקוח את הנתונים מהמחשבון הפיזיקלי** | **List[time,distance,overweight]- מחזיר ללקוח את הנתונים מהphysics\_calc** | **Load\_mass – המטען הכבד.** | receive\_input |

**Physics\_calc.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **מטרה** | **output** | **input** | **name** |
| **מרכז לכל התת פונקציות ומחזיר את כל התוצאות שלהם בצורה מסודרת** | time, distance, overweight  **או -9 במקרה של ערך לא צפוי** | load\_mass | general\_calc |
| **מחשב את זמן ההמראה** | time | acceleration | take\_off\_time |
| **מחשב את המרחק** | distance | acceleration, time | take\_off\_distance |
| **מחשב את המסה שיש להשמיד בכדי להגיע למתחת ל 60 שניות** | **מסה להשמדה** | time, imass | overweight\_reduce |

**Client:**

**App.js**

|  |  |  |  |
| --- | --- | --- | --- |
| **מטרה** | **output** | **input** | **name** |
| **לעדכן את ערך המטען הכבד בכל שינוי של הלקוח** | **אין** | Event-מטען כבד | handleInputChange |
| **לבקש מהשרת את הטמפ של האיזור או השעות בהם ניתן לקיים את המבצע** | Html-השעות בהם ניתן לקיים את במבצע או הטמפ של האיזור | selectedDate-תאריך הנבחר לקיום המבצע | weatherCheck |
| **להחזיר ללקוח את הנתונים מהמחשבון הפיזיקלי** | **List[time,distance,overweight]** | Event-מטען כבד | handleSubmit |
| **משנה את התאריך הנבחר** | **אין** | Event-תאריך | handleDateChange |