**המחלקה להנדסת תוכנה - קורס מבוא למחשוב ענן**

**סמסטר אביב התשפ"ד**

**תרגיל בית 3**- עבודה **בצוותי העבודה**

מועד הגשה: 15.8.24

המשימה בתרגיל זה: סיום בניית המערכת, ביצוע קוד ריוויו לצוות אחר - איטרצית פיתוח מס 2

שימו לב: למטלה זו שלושה חלקים

קישור לקולאב :

: [Snake\_Project.ipynb - Colab (google.com)](https://colab.research.google.com/drive/1qvIYofjyHQwWZaUIUMYEgRvtifklC9nS?usp=sharing#scrollTo=9TNghIVp4VQA)

**חלק ראשון.**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם חבר הצוות** | **משימות שהוקצו** | **משימות שהושלמו** | **Acceptance test –** בדיקת קבלה |
| מהנדס צוות -   אלכס בבושין | -צ'אט בוט  -יצירת גרף נוסף(תרומה של כל סטודנט במטלה ספציפית) | כל המשימות הושלמו בהצלחה | -הכנסנו את כל הprompts שהצאט תומך בהם וקיבלנו את התשובה הרצויה  -הכנסנו קבצי JSON שונים על מנת לבדוק שהגרף תקין |
| שי אוסמו | -ביצוע בדיקות קבלה | כל המשימות הושלמו בהצלחה | בוצע |
| רימי חלבי | - התראות | כל המשימות הושלמו בהצלחה | -בדקנו ש5 ההתראות שמופיעות במסך ההתראות הן באמת ההתראות האחרונות שבוצעו בקובץ |
| אלון ברק | -תיקון CSS | כל המשימות הושלמו בהצלחה | -הרצת כל המסכים ובדיקה שהכל מסודר כפי שרצינו |
| רונן פרידמן | -סידור הקוד בצורה מודולרית | כל המשימות הושלמו בהצלחה | -לאחר סידור הקוד, בדיקה שהכל עובד כמו שצריך |
| דימטרי סימון | * מילואים |  |  |

***חלק שני: בניית המערכת*** (70 נקודות)

|  |  |  |
| --- | --- | --- |
| **הערה** | **תגובה** | **האם בכוונת הצוות לבצע שינוי בעקבות ההערה? יש לפרט. אם כן, מה השינוי. אם לא, מדוע** |
| אין וידאו | חלק מהדרישות בתרגיל 3, יבוצע. | בוצע וידאו |
| לא ניתן לבצע חלוקת פעולות לפי יום בגרף האחרון | הצוות יבחן הוספת אפשרות לסינון פעולות בהתאם לתאריך. | לא בוצע, לדעתנו גרף לפי תאריך הוא לא רלוונטי. הוספנו גרף נוסף המתאר את תרומת כל סטודנט |
| יעילות- אפשר לעשות את הקוד יותר מודולרי לפרק לחלקים יותר קטנים | מסכימים עם ההערה. | חילקנו את הקוד למקטעים והוספנו תיעוד מפורט. |
| אפשר להשתמש בספריות שיפשטו יותר את הקוד . | מסכימים עם ההערה , נבדוק אם ניתן לפשט את הקוד בעזרת הוספת ספרייה | לא בוצע,  לא מצאנו ספרייה שתפשט את הקוד. |
| אם הועלה קובץ שהוא לא JSON -אין הערה מתאימה | מסכימים עם ההערה. | בוצע. נוספה התראה. |
|  |  |  |

***חלק שלישי : סגירת הפרויקט \_( 20 נקודות)***

בחלק זה תכינו מסמכים המתארים את הפרויקט שלכם:

1. ~~יש לבנות תיק למתכנת הכולל את שמות כל הקבצים המרכזיים, פונקציות מרכזיות, קטעי קוד/תבניות עיצוב מעניינים שהשתמשתם בהם.~~ **בוצע**
2. יש לבנות תיק למשתמש , הכולל הסבר כללי על המערכת , פירוט מסכים, מעברים בין מסכים והסבר על טעויות אפשרויות. **בוצע**
3. יש להכין סרטון קצר של 30-60 שניות, המתאר את השימוש במערכת. הסרטון משמש כ – elevator pitch  למערכת שלכם, כלומר יש לכלול בו הסבר מקצועי ועם זאת שיווקי , המדגיש את האלמנטים המיוחדים של המערכת שבניתם. יש להגיש את הסרטון בפורמט mp4.

**תיק למתכנת:**

מערכת לניתוח פעולות משתמשים ב-Google Colab באמצעות Firebase ו-Chatbot

המערכת מיועדת לניתוח ולמעקב אחר פעולות משתמשים בשימוש ב-Google Colab ו-Firebase. בנוסף, היא כוללת צ'אטבוט שמספק תמיכה למשתמשים. להלן פירוט של המבנה והתפקידים של המערכת:

המחברת בgoogle colab כוללת:

מכיל את כל הקוד המרכזי של המערכת כולל את ה-HTML, CSS, JavaScript, וקוד Python.

פונקציות מרכזיות

upload\_json:

אחראית על קבלת קבצי JSON, ניתוחם ושמירתם ב-Firebase.

classifyData:

מסווגת את הנתונים מה-JSON וסופרת את הפעולות שבוצעו על ידי כל משתמש.

generate\_combined\_js:

יוצרת את הקוד JavaScript שמציג את הגרפים והנתונים המנותחים בדף ה-HTML.

get\_bot\_response:

אחראית על קבלת תגובות מהצ'אטבוט בהתבסס על קלט המשתמש.

handle\_bot\_response:

מטפלת בתגובות של הצ'אטבוט ומציגה אותן בממשק המשתמש.

move\_upload, move\_glossary, move\_team\_stats, move\_notifications:

פונקציות שאחראיות על ניהול דפי המערכת והצגת הנתונים הרלוונטיים.

קטעי קוד ותבניות עיצוב מעניינים

קטע קוד המחשב את הציון הכולל של כל הצוות:

action\_weights **=** **{**

‘add’**:** 6**,**

‘insert’**:** 5**,**

‘commit’**:** 4**,**

‘change’**:** 3**,**

‘edit’**:** 3**,**

‘update’**:** 3**,**

‘fix’**:** 4**,**

‘start’**:** 2**,**

‘delete’**:** **-**2**,**

‘cancel’**:** **-**2

**}**

**def** calculate\_individual\_scores**(**student\_data**):**

scores **=** **{}**

**for** student**,** actions **in** student\_data**.**items**():**

score **=** 0

**for** action**,** count **in** actions**.**items**():**

score **+=** action\_weights**.**get**(**action**,** 0**)** **\*** count

scores**[**student**]** **=** score

**return** scores

**def** calculate\_team\_score**(**individual\_scores**):**

**return** sum**(**individual\_scores**.**values**())**

**def** normalize\_score**(**score**,** max\_score**):**

**return** round**((**score **/** max\_score**)** **\*** 10**,** 1**)**

הקוד מציג פרמטרים הניתנים לשינוי למתכנת, לכל פעולה קיים משקל הניתן לשינוי לבחירת המתכנת, סכום הפעולות של כל המשתמשים לפי המשקל מנורמל וניתן ציון קבוצתי לצוות.

קוד של הצ'אט בוט:

הקוד של הבוט, בהתחלה אנחנו מקבלים input מהמשתמש, בודקים האם קיימת תבנית כזאת לתגובה, אם כן שומרים את המפתח שלה בבסיס הנתונים. ניגשים לבסיס הנתונים ושולפים לפי המפתח אחת מהתשובות לשאלה רנדומאלי. מקבלים את התשובה ומחזירים למשתמש אותה.

# Function to fetch responses from Firebase based on a pattern key

**def** fetch\_response\_from\_firebase**(**pattern\_key**):**

***# Create a reference to the specified pattern key in the Firebase database***

ref **=** db**.**reference**(**f'chatbotResponses/{pattern\_key}'**)**

***# Retrieve the responses associated with the pattern key from Firebase***

responses **=** ref**.**get**()**

***# If responses are found, return them***

**if** responses**:**

**return** responses

**else:**

***# If no responses are found, fetch the default response from Firebase***

ref\_default **=** db**.**reference**(**'chatbotResponses/default'**)**

default\_response **=** ref\_default**.**get**()**

**return** default\_response

***# Example function to match user input with a pattern key***

**def** match\_pattern**(**user\_input**):**

***# Define a dictionary of pattern keys and their corresponding regex patterns***

patterns **=** **{**

'hi'**:** r'hi|hello|hey'**,**

'how\_are\_you'**:** r'how are you?'**,**

'upload\_json'**:** r'how to upload JSON file?'**,**

'filter\_graphs'**:** r'how to filter the Team Stats graphs?'**,**

'calculate\_stats'**:** r'how the overall team stats are calculated?'

**}**

***# Iterate through the patterns to find a match with the user's input***

**for** key**,** pattern **in** patterns**.**items**():**

**if** re**.**search**(**pattern**,** user\_input**,** re**.**IGNORECASE**):**

**return** key ***# Return the matching pattern key if a match is found***

**return** 'default' ***# Return 'default' if no pattern matches the user input***

***# Function to get the chatbot's response based on user input***

**def** get\_bot\_response**(**user\_input**):**

***# Match the user input with a pattern key***

pattern\_key **=** match\_pattern**(**user\_input**)**

***# Fetch the corresponding responses from Firebase***

responses **=** fetch\_response\_from\_firebase**(**pattern\_key**)**

***# If responses are found, return a random choice among them***

**if** responses**:**

**return** random**.**choice**(**responses**)**

**else:**

***# Return a fallback message if no responses are found***

**return** "Sorry, I did not understand that. Can you rephrase?"

***# Function to handle the display of the chatbot's response in the UI***

**def** handle\_bot\_response**(**user\_input**):**

***# Get the bot's response based on the user's input***

response **=** get\_bot\_response**(**user\_input**)**

***# Execute JavaScript to display the response in the chat interface***

output**.**eval\_js**(**f"""

const botMessage = `{response}`;

setTimeout(() => {{

const messageDiv = document.createElement('div');

messageDiv.classList.add('message', 'bot');

messageDiv.textContent = botMessage;

document.getElementById('messages').appendChild(messageDiv);

document.getElementById('messages').scrollTop = document.getElementById('messages').scrollHeight;

}}, 500); # Delay the message display slightly for a more natural conversation flow

"""**)**

פירוט הפונקציה upload\_json

תיאור כללי

הפונקציה upload\_json אחראית על קבלת קובץ JSON מהממשק המשתמש, ניתוח הנתונים בקובץ, שמירתו ב-Firebase והצגת הנתונים בדף הסטטיסטיקות. בנוסף, הפונקציה כוללת בדיקה לוודא שהקובץ שהועלה הוא מסוג JSON בלבד.

פירוט הפונקציה שלב אחרי שלב

בדיקת סוג הקובץ:

**if** **not** filename**.**endswith**(**'.json'**):**

output**.**eval\_js**(**"""

alert('Error: Only JSON files are allowed!');

"""**)**

**return**

הפונקציה מתחילה בבדיקה אם הקובץ שהועלה הוא קובץ JSON על פי סיומת הקובץ. אם הקובץ אינו קובץ JSON, נשלחת הודעת שגיאה למשתמש והפונקציה מסתיימת.

הצהרה על משתנים גלובליים:

**global** global\_processed\_data

**global** taskData

הצהרה על משתנים גלובליים שישמשו לאחסון הנתונים המנותחים.

פריסת תוכן ה-JSON:

**try:**

data **=** json**.**loads**(**content**)**

פריסת תוכן הקובץ שהועלה למבנה נתונים (רשימה או מילון) באמצעות פונקציה json.loads.

שמירת קובץ ה-JSON:

filepath **=** os**.**path**.**join**(**os**.**getcwd**(),** filename**)**

**with** open**(**filepath**,** 'w'**)** **as** f**:**

json**.**dump**(**data**,** f**,** indent**=**2**)**

שמירת הקובץ שהועלה בתיקיית העבודה הנוכחית.

הצגת פריט ראשון מה-JSON:

print**(**f"File '{filename}' has been saved successfully in the Colab folder."**)**

**if** isinstance**(**data**,** dict**):**

first\_key **=** next**(**iter**(**data**))**

print**(**f"First item in the JSON: {first\_key}: {data**[**first\_key**]**}"**)**

**elif** isinstance**(**data**,** list**)** **and** len**(**data**)** **>** 0**:**

print**(**f"First item in the JSON: {data**[**0**]**}"**)**

**else:**

print**(**"The JSON file is empty or has an unexpected structure."**)**

בדיקת סוג הנתונים והצגת הפריט הראשון שבקובץ ה-JSON.

העלאת הנתונים ל-Firebase:

upload\_path **=** '/Json/'

upload\_cloud**(**data**,** upload\_path**)**

העלאת הנתונים ל-Firebase באמצעות פונקציה upload\_cloud.

ניתוח הנתונים:

all\_users **=** get\_all\_users**()**

users\_string **=** ' '**.**join**(**all\_users**)**

actionData **=** searchJson**(**data**,** 'Add Insert Commit Delete Change Edit Update Cancel Fix Start'**,** '/SearchResults/'**)**

studentData **=** searchJson**(**data**,** users\_string**,** '/StudentResults/'**)**

global\_processed\_data**,** taskData **=** classifyData**(**data**)**

ניתוח הנתונים מה-JSON לשני סוגי נתונים עיקריים:

actionData - חיפוש פעולות מסוימות במבנה הנתונים.

studentData - חיפוש נתונים על כל משתמש.

classifyData - סיווג הנתונים ושמירתם במשתנים גלובליים.

מעבר לדף הסטטיסטיקות:

move\_team\_stats**()**

קריאה לפונקציה המעבירה את המשתמש לדף הסטטיסטיקות ומציגה את הנתונים המנותחים.

התמודדות עם שגיאות:

**except** json**.**JSONDecodeError**:**

print**(**"Error: The file content is not a valid JSON."**)**

output**.**eval\_js

alert**(**'Error: The file content is not a valid JSON!'**);**

טיפול בשגיאות במקרה שהתוכן שהועלה אינו קובץ JSON תקין והצגת הודעת שגיאה למשתמש.

הפונקציה move\_notifications מקבלת את ההודעות האחרונות מ-Firebase, מעבדת אותן לרשימה מסודרת, יוצרת קוד JavaScript להצגת ההודעות המוגבלות ל-5 האחרונות, ואז מציגה אותן בעמוד האינטרנט.

**def** move\_notifications**():**

#clearing the page

clear\_output**()**

***#displayHtml(notificationPage, notificationPageJs)***

***# Fetch the latest notifications from Firebase***

notifications\_data **=** FBconn**.**get**(**'/Json'**,** None**)**

***#if notifications\_data:***

***# Flatten the notifications data***

notifications\_for\_display **=[]**

**for** notification **in** notifications\_data**:**

notifications\_for\_display**.**append**(**

**{**

"Time"**:** notification**.**get**(**'Time'**),**

"User"**:** notification**.**get**(**'User'**),**

"Document"**:** notification**.**get**(**'Document'**),**

"Tab"**:** notification**.**get**(**'Tab'**),**

"Description"**:** notification**.**get**(**'Description'**)**

**}**

**)**

notificationJs**=**createNotification**(**notifications\_for\_display**[:**5**])**

displayHtml**(**notificationPage**,**notificationJs**)**

**תיק למשתמש**

**הסבר כללי**

מערכת לניהול וניתוח פעולות משתמשים בתוכנת OnShape.

המערכת מאפשרת למנהלים לנהל ולנתח פעולות של משתמשים באמצעות העלאת קבצי JSON המכילים נתונים על הפעולות שבוצעו על ידי המשתמשים. הנתונים כוללים פרטים כמו שם משתמש, תיאור הפעולה, ומשימה זמן וכו.

תכונות המערכת:

העלאת קבצי JSON:

המערכת מאפשרת למנהל להעלות קבצי JSON המכילים נתונים על פעולות המשתמשים.

אם הקובץ שהועלה אינו קובץ JSON, המערכת מציגה הודעת שגיאה למשתמש.

שמידה והקובץ חוקי, המערכת תעלה את התוכן לבסיס נתונים ותשמור אותו.

שימוש ב-Firebase:

הנתונים מועלים ל-Firebase לצורך אחסון.

המערכת משתמשת בבסיס הנתונים Firebase כדי לשמור ולעבד את הנתונים.

ניתוח נתונים:

המערכת מסווגת את הנתונים לפי משתמשים ומשימות.

היא סופרת את מספר הפעולות שבוצעו על ידי כל משתמש ומחלקת את הפעולות לפי סוגים (הוספה, עריכה, מחיקה וכו').

המערכת גם מסווגת את הנתונים לפי משימות ומציגה את התרומה של כל משתמש לכל משימה.

הצגת נתונים:

המערכת מציגה את הנתונים בצורה גרפית.

גרפים מציגים את מספר הפעולות שבוצעו על ידי כל משתמש, ואת התרומה של כל משתמש לכל משימה.

המערכת מציגה את הניקוד הכללי של הצוות על בסיס הפעולות שבוצעו.

ממשק משתמש:

הממשק כולל טופס להעלאת קבצים וכפתורים למעבר בין עמודים שונים (העלאת JSON, מילון מונחים, סטטיסטיקות צוות, התראות).

ישנה מערכת צ'אטבוט שמסייעת למשתמשים ומאפשרת להם לקבל תשובות לשאלות בנוגע למערכת.

הממשק עוצב בצורה מודרנית וידידותית למשתמש.

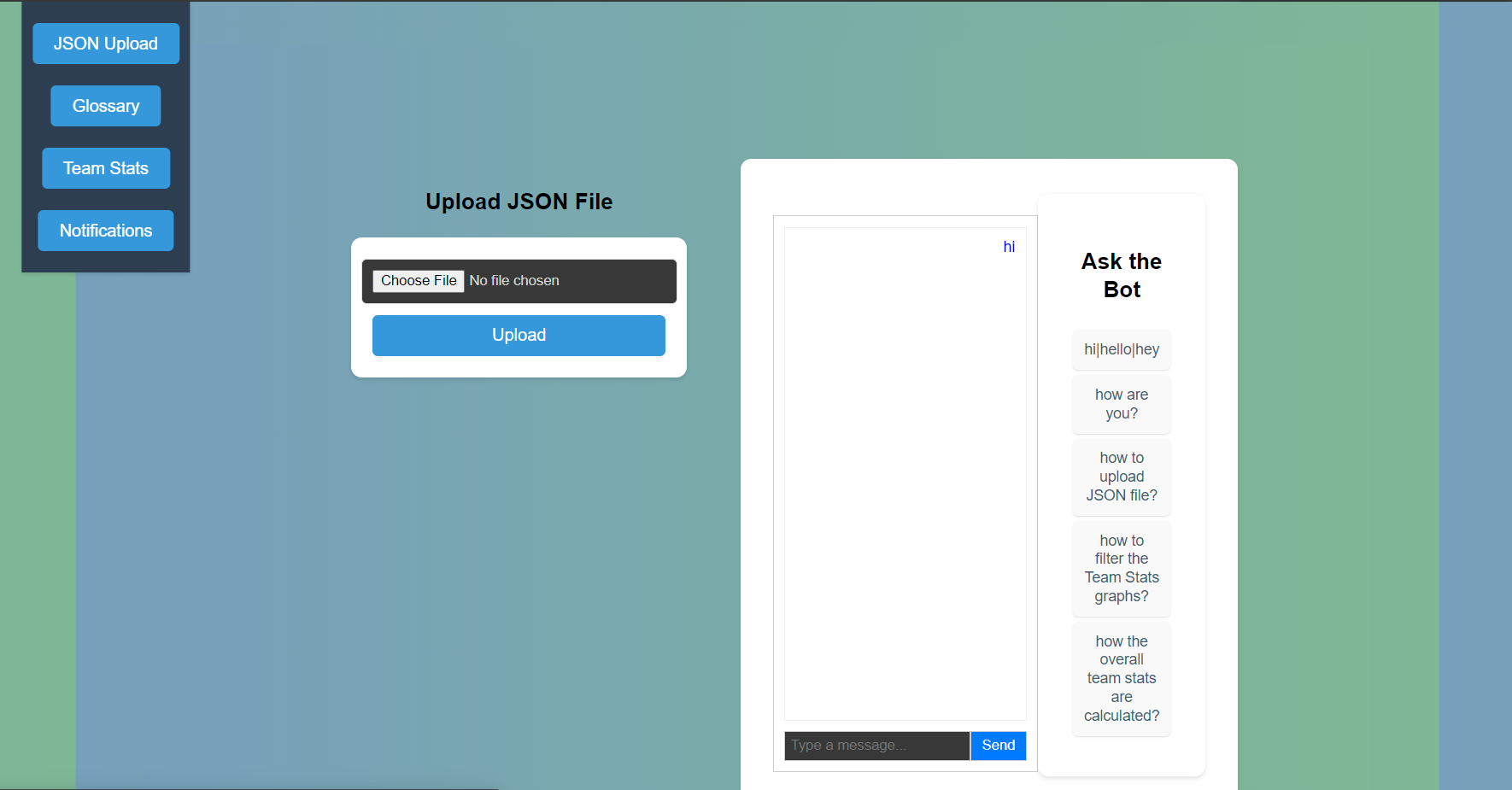
**מסכים**

**תמונה שמכילה טקסט, צילום מסך, גופן, מספר

התיאור נוצר באופן אוטומטי**

סרגל לבחירת המסך הרצוי.

**JSON Upload**



במסך נדרש להעלות קובץ JSON .   
כאשר נבחר קובץ מסוג שונה תקפוץ הודעת שגיאה.

**Glossary**

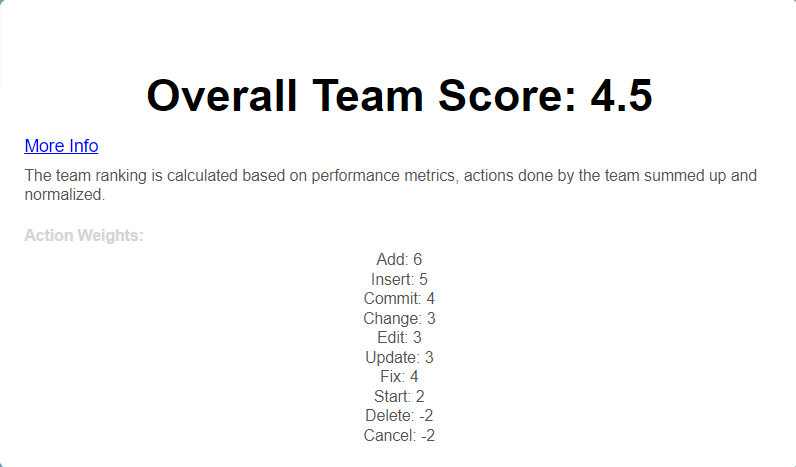
טבלת אינדקסים של הפעולות המשמעותיות, לכל מילה קיים תיאור ומספר הפעמים שהיא מופיעה בקובץ.

A screenshot of a computer

Description automatically generated

**Team Stats**

בלוק זה מראה למשתמש את הציון הכללי של הצוות המחושב לפי פעולות חיוביות ושליליות, הציון ניתן באופן מנורמל.



בלוק זה מציג לכל משתמש את מספר הפעולות שהוא ביצע, כל פעולה נספרת כנקודה אחת.

תמונה שמכילה טקסט, צילום מסך, מספר, גופן

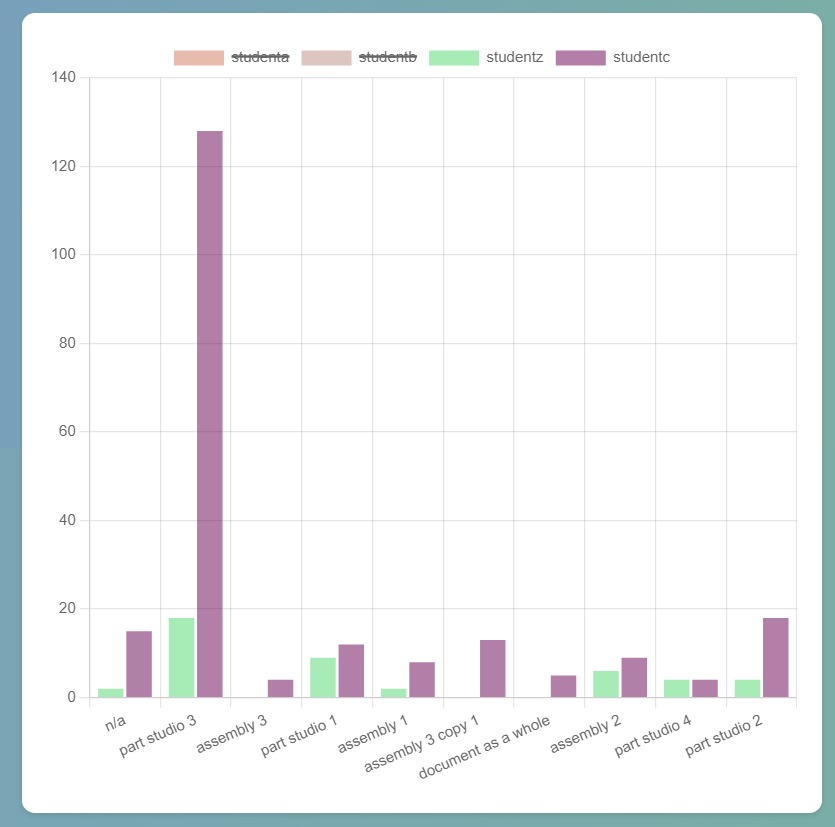
התיאור נוצר באופן אוטומטי

בלוק שלישי מציג גרף המפרט עבור כל משתמש את סוג הפעולות שהוא ביצע וכמות הפעמים שבוצעה כל פעולה. את הגרף ניתן לסנן לפי פעולות רצויות ולא רצויות.

תמונה שמכילה טקסט, צילום מסך, מספר, עלילה

התיאור נוצר באופן אוטומטי

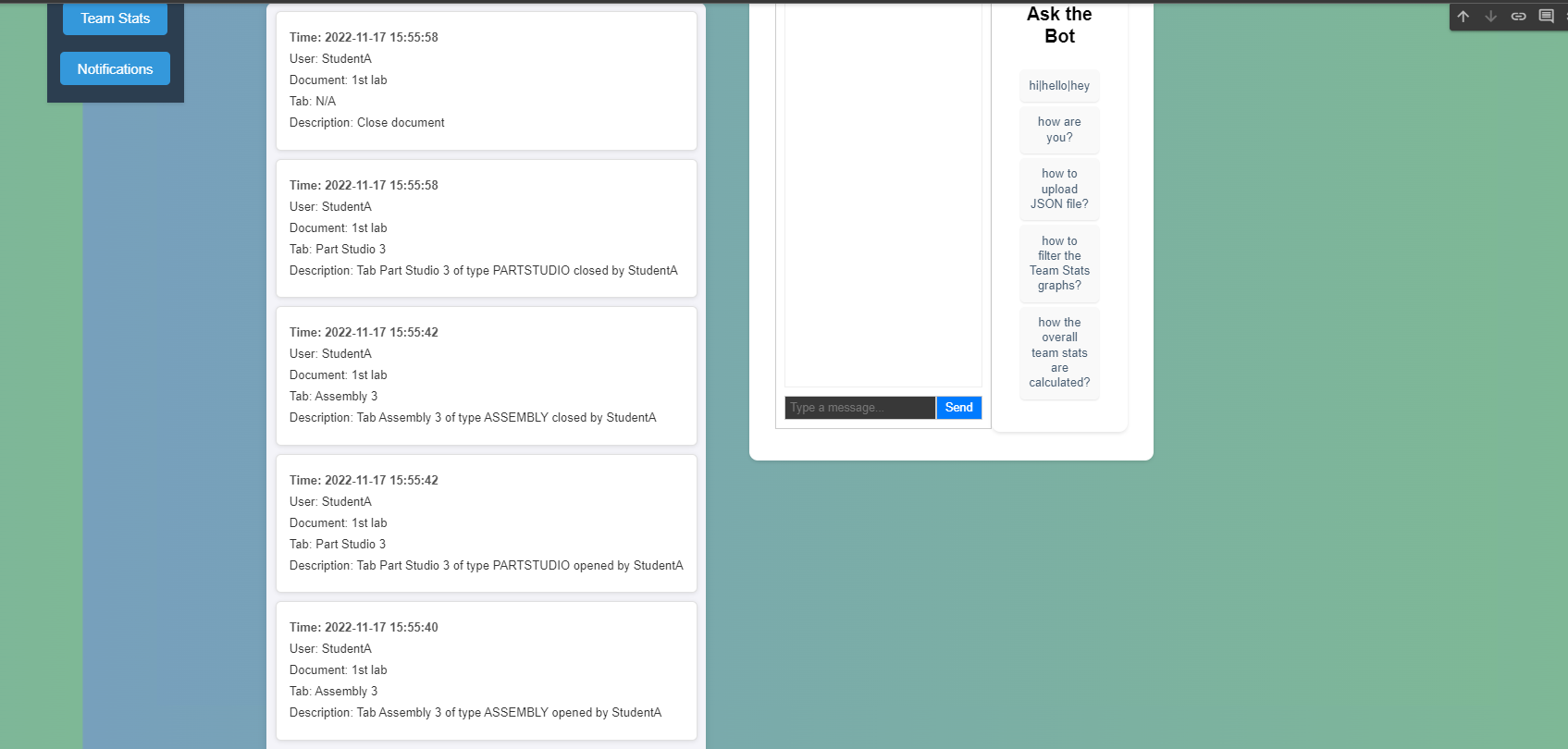
בלוק רביעי מציג תרומת כל סטודנט לכל מטלה TASK כך שניתן להשוות בקלות כמה כל סטודנט תרם בפעולות שביצע לכל אחת מהמטלות. כל עמודה מייצגת את התרומה של סטודנט מסוים עבור מטלה מסוימת, עם אפשרות לסינון לפי הסטודנטים הרלוונטיים.



**Notification**

מסך התראות המציג את חמש הפעולות האחרונות שבוצעו במערכת.

לכל פעולה נשמר שם המבצע, הפעולה שבוצעה, שם הפרויקט, תאריך ושעה.



הוראות הגשה:

1. יש להגיש את התרגיל בצוותים, בתיקיית ה –GIT שלכם, וכן בתיקייית התרגיל ב moodle. **חובה לכלול קישור לתיקיית הגיט בקובץ במוודל.**
2. יש להגיש במוודל קובץ זיפ הכולל קובץ וורד ובו מענה לשאלות, וקישור ל- notebook ובו הקוד שלכם (יש לוודא שהקישור פומבי ונגיש).
3. כותרתו של הקובץ תהיה HW3\_groupName
4. שימו לב כי כל העבודות חייבות להיות שונות זו מזו.אנו מריצות תוכנה לבדיקת עבודות זהות. עבודות שייראו דומות ייפסלו ויינתן עליהן ציון 0.

בהצלחה!