

נושא פרויקט : סביבת הרצה מקוונת לשפת C

שם מלא : שרה يولி סמיישקיס

ת"ז : 216764803

מנחה : אופיר שביט

שם החלופה : סייבר ומערכות הפעלה

תיקון ע"ש חיים הרצוג כפר סבא



**תוכן עניינים**

<b>מבוא.....</b>	<b>ייזום.....</b>
4.....	תיאור ראשי של המערכת..... הגדרת הלקוּח..... הגדרת יעדים ומטרות..... בעיות תועלות וחסכנות..... סקירת פתרונות קיימים..... סקירת טכנולוגיית הפרויקט..... סיגים בהגדרת המערכת..... תיחום הפרויקט.....
<b>אפיון.....</b>	<b>תיאור מפורט של המערכת.....</b>
8.....	פירוט יכולות השרת..... פירוט יכולות הלקוּח..... פירוט יכולות המשותפות לשרת וללקוּח..... פירוט בדיקות..... תכנוּן וניהול זמנים..... ניהול הסיכון.....
<b>תיאור תחום הידע.....</b>	<b>יכולות בצד השרת.....</b>
14.....	יכולות בצד הלקוּח.....
18.....	יכולות משותפות.....
21.....	<b>ארכיטקטורה של הפרויקט.....</b>
23.....	<b>הארכיטקטורה המבוצעת תחולק לשולש שכבות עיקריות:.....</b>
23.....	Frontend.....
23.....	Backend.....
23.....	Database.....
24.....	הchromera.....
25.....	<b>טכנולוגיה הרלוונטית.....</b>
25.....	שפות תכנות.....
25.....	מסגרות ותשויות.....
25.....	כלים ומנגנונים נוספים.....
26.....	<b>זרימת המידע במערכת הרשמה והתחברות.....</b>
27.....	עריכת קוד.....
28.....	היזור והרצת קוד.....
29.....	עבודה על תרגילים.....
30.....	<b>האלגוריתמים.....</b>
30.....	עריכה שיטופית בזמן אמת.....
30.....	בדיקות קוד מסוכן.....
31.....	הגבלת משאבים בהרצת קוד.....
31.....	בדיקות אוטומטיות של פתרונות תרגילים.....
32.....	אלגוריתמים קיימים.....
32.....	מקורות רלוונטיים.....
32.....	<b>סביבה הפיתוח.....</b>
32.....	כלי פיתוח.....
33.....	פירוט הסביבה וכלי בדיקה.....
33.....	<b>פרוטוקול התקשרות.....</b>
33.....	פירוט כלל החודעות.....
34.....	<b>MSCSI המערכת.....</b>
34.....	מסך כניסה (Login).....
35.....	מסך הרשמה (Register).....
36.....	לוח מחוונים (Dashboard).....
37.....	מסך עריכת פרויקט.....

38.....	מסך תרגילים .....
39.....	מסך תרגיל .....
40.....	מסך ניהול .....
41.....	תרשים מסכימים .....
<b>41.....</b>	<b>מבנה הנתונים .....</b>
41.....	פירוט מבני הנתונים .....
44.....	פירוט מאגרי המידע .....
45.....	מסדי נתונים .....
<b>46.....</b>	<b>סקירה חולשות והאינומים .....</b>
46.....	שכבות האפליקציה .....
46.....	שכבות התעבורה .....
46.....	שכבות המערכת .....
<b>47.....</b>	<b>הימוש בפרויקט .....</b>
<b>47.....</b>	<b> חלק א' - סקירת המודולים והמחלקות של המערכת .....</b>
47.....	מודולים מיובאים .....
47.....	מחלקות שפותחו .....
<b>50.....</b>	<b> חלק ב' - אלגוריתמים מרכזיים בפרויקט .....</b>
50.....	אלגוריתם הרצת קוד C מאובטח .....
51.....	אלגוריתם Operational Transformation לעריכה שיטופית .....
52.....	אלגוריתם בדיקת תרגילי קוד אוטומטית .....
<b>52.....</b>	<b> חלק ג' - מסמך בדיקות .....</b>
54.....	בדיקות אבטחה .....
55.....	בדיקות ביצועים .....
55.....	בדיקות נספנות שבוצעו .....
<b>57.....</b>	<b> מדדיך למשתמש .....</b>
<b>57.....</b>	<b> פירוט קבצי המערכת - עץ קבצים .....</b>
<b>58.....</b>	<b> התקנת המערכת .....</b>
58.....	סבירה נדרשת .....
58.....	כלים נדרשים .....
<b>59.....</b>	<b> משתמשי המערכת .....</b>
<b>64.....</b>	<b> סיכום אישי / רפלקציה .....</b>
64.....	תהליך העבודה על הפרויקט .....
64.....	תהליך הלמידה .....
64.....	כלים לעתיד .....
65.....	תובנות מהתהליך .....
65.....	סיכום .....
<b>66.....</b>	<b>ביבליוגרפיה .....</b>
<b>67.....</b>	<b>נספחים .....</b>
<b>67.....</b>	<b>נספח א': פרוטוקול תקשורת WebSocket .....</b>
67.....	הודעות מהלקוח לשרת .....
68.....	הודעות מהשרת ללוקוח .....
<b>69.....</b>	<b>נספח ב': שמות תיקיות וקבצים שהוגבלו לשימוש בסנדבוקס .....</b>
<b>70.....</b>	<b>נספח ג': אלגוריתם בדיקת קוד לתבניות מסוכנות .....</b>
<b>72.....</b>	<b>קוד הפרויקט : .....</b>

## מבוא

### יעזום

#### תיאור ראשוני של המערכת

הפרויקט כולל פיתוח סביבת הרצה מקוונת לשפת תכנות C, המאפשרת למשתמשים לעבוד בשיתוף פעולה בזמן אמת. בסביבת הרצה ניתן לכתוב קוד, להדר אותו ולהריץ אותו, לראות את הפלט וכן לתקשר עם משתמשים אחרים. המערכת כוללת גם אוצר עם תרגילי לימוד, המאפשרת למשתמשים לתרגל את ידיעותיהם בשפת C באמצעות סדרת תרגילים ברמות קושי שונות.

המערכת מורכבת מצד שרת (Backend) שנכתב בשפת Python עם המסתגרת Flask, וצד לקוח (Frontend) שנכתב באמצעות HTML, CSS ו-JavaScript. התקשורת בזמן אמת מתבצעת באמצעות IO Socket.IO. המאפשר תקשורת דו-כיוונית בין הלוקו לשרת. מידע של משתמשים, פרויקטים ותרגילים נשמר במסד נתונים באמצעות SQLAlchemy.

אחד היתרונות המרכזים של המערכת הוא יכולת לעבוד בשיתוף פעולה בזמן אמת, כאשר כל המשתמשים יכולים לראות את השינויים שנעשים על ידי משתמשים אחרים באופן מיידי, וכן יכולת להדר ולהריץ קוד C בסביבה מאובטחת.

החלטי לבחור בפרויקט זה מהחר ובעת לימודי אוניברסיטה, יצא לי ללמידה ולתרגל את שפת C, ואחד מהדברים שזכורים לי מהקורס בו למדתי את השפה, זה שבאחד מהתרגילים הייתה אפשרות להריץ את הקוד ולראות כמה זיכרונו דלפ בהקצאות הזיכרון שביצעו בקוד, אך עם הנוחות שספק הכליל הוא מסובך לשימוש ורוב התלמידים ייתרו על השימוש בו, אך אני רוצה ליצור סביבת עבודה אשר תהיה זמינה בכל מחשב בלי התקנות מסוימות, אשר תגשים את הכלים המייעלים ומשמעות תכונות ב-C.

#### אתגר פיתוח

פרויקט זה כלל מספר אתגרים טכניים משמעותיים:

- יצירת מנגן עריכה שיתופית בזמן אמת עם טיפול בתנשויות הקמת סביבת ריצה מאובטחת ( Sandbox ) להרצה קוד C, תוך מניעת קוד זמני
- פיתוח תקשורת בזמן אמת בין לקוחות רבים
- הגנה על המערכת מפני איומי אבטחה שונים, כולל ניסיונות SQL Injection ו-Cross-Site Scripting

#### הגדרת הלוקו

המערכת מיועדת למשתמש המומצא שימושי להתחילה ללמידה או לתרגל תכונות בשפת C, כגון:

- **סטודנטים למדעי המחשב** - המתמודדים עם שפת C במסגרת קורסי מבוא לתכונות או קורסים ממוקדים כמו מערכות הפעלה ורשתות.
  - **מורים ומרצים** - המעניינים בכלי שיאפשר להם להציג קוד C באופן אינטראקטיבי ולא צריך בתקנות מסוימות.
  - **מתכננים מתחילה** - המעניינים למדוד שפת C באופן עצמאי.
  - **צוותי פיתוח** - העובדים על פרויקטים ממשותיים ומתחשים סביבת עבודה שיתופית מקוונת.
- הלוקוחות העיקריים הם אנשים החסרים גישה למשאים יקרים כגון סביבות פיתוח מקצועיות, או אלה שמעדיפים סביבת עבודה פשוטה ואיינטואיטיבית ללא צורך בתקינות מסוימות.

#### הגדרת יעדים ומטרות

- **עורץ קוד שיתופי** - יצירת עורץ קוד שיאפשר למספר משתמשים לעורך את אותו הקוד בו-זמן, עם מנגן שיתוף בזמן אמת שיציג את מיקום הסמן של כל משתמש והשינויים שהוא מבצע.

- סביבה הדרה והרצה קוד** - פיתוח מנגנון בטוח להדרת והרצה קוד C, עם הצגת פלט ההרצה בצורה ברורה.
- תמיכה במלידה** - ייצור מערכת תרגילים ברמות קושי שונות, עם בדיקה אוטומטית של פתרונות וממןuable לשימוש.
- תקשורות בין משתמשים** - שימוש מנגנון צ'אט בזמן אמת בין משתמשים העובדים על אותו פרויקט, ואינדיקטור המציג אילו משתמשים מחובריםCurrently.
- ניהול משתמשים ופרויקטים** - ייצור מערכת לניהול משתמשים, כולל הרשמה, התחברות ושמירת פרויקטים, עם אפשרות לשתף פרויקטים עם משתמשים אחרים.
- בטחה** - הטמעת מנגנוני בטחה להגנה על המערכת ונתוני המשתמשים, כולל הצפנה סיסמאות והרצה קוד בסביבה מבודדת.

### בעיות תועלות וחסכנות

#### מהן הבעייה

- מורכבות הקמת סביבת פיתוח** - הקמת סביבת פיתוח לשפת C דורשת לרוב התקנת מהדר IDE, כלים נוספים, תחילה שלול להיות מרכיב עבור מתחילה.
- קושי בעבודה שיתופית** - כלים מסורתיים לפיתוח בשפת C אינם מאפשרים עבודה שיתופית בזמן אמת, ודורשים שימוש במערכות ניהול גרסאות מורכבות.
- חוסר בכלי למידה אינטראקטיביים** - קיימים מחסור בכלי למידה אינטראקטיביים לשפת C שימושייםurrect, הרצה ותרגול בסביבה אחת.
- מוגבלות זמינות** - כלי פיתוח מסורתיים מוגבלים למחשב אחד ואין זמינים מכל מקום.

#### מהן התיעלת

- נגישות** - המערכת זמינה מכל מחשב עם חיבור לאינטרנט, ללא צורך בהתקנות מסובכות.
- סביבת עבודה אחידה** - כל המשתמשים עובדים באותה סביבה, מה שמנוע בעיות תאימות בין מערכות הפעלה וגרסאות מהדר שונות.
- שיתוף פעולה** - המערכת מאפשרת עבודה שיתופית בזמן אמת, כולל תקשורת ישירה בין משתמשים.
- למידה אפקטיבית** - שילוב של הרצה קוד, הרצה ותרגול באותה סביבה מייעל את תהליך הלמידה.
- בטחה** - הרצה קוד בסביבה מבודדת מגנה על המערכת ועל המשתמשים מפני קוד זדוני.

### אילו שירותים תציע המערכת

המערכת מציעה את השירותים הבאים:

- עריכת קוד** - עורך קוד עם הדגשת תחביר, השלמת קוד אוטומטית ואפשרויות עריכה מתקדמות.
- עריכה שיתופית** - אפשרות לעבוד על אותו קוד בו-זמנית עם משתמשים אחרים, עם הצגת שינויים בזמן אמת.
- הדרה והרצה קוד** - אפשרות להדר ולהרץ קוד C באופן מיידי, עם הצגת פלט ההרצה.
- תרגול** - סדרת תרגילים ברמות קושי שונות, עם בדיקה אוטומטית של פתרונות וממןuable.
- תקשורות** - צ'אט בזמן אמת בין משתמשים העובדים על אותו פרויקט.
- ניהול פרויקטים** - שמירת פרויקטים, גישה לפרוייקטים קודמים ושיתוף פרויקטים עם משתמשים אחרים.

## סקירת פתרונות קיימים

- מאחר והפרויקט שלי עוסק בנושאים אשר רבים מימשו כבר, אך לא שילבו בתוצר אחד קשה למצוא פתרון קיים אך אסקור את ההשראות שלי לפרויקט :
- כאשר זה מגע למזהדרי שפת C מוקונים, שני פתרונות קיימים הם :
- – **Repl.it** – סביבת פיתוח מקוונת התומכת בתוכנות בסיס ומציעה שיתוף עריכה בזמן אמת, הרצת קוד בסביבה sandbox, אימוט משתמשים ושיתוף פרויקטים. אך עם זאת איןנה מעניקה את ההתקדמות הישראלית בלימוד C ולא מכילה אופציה לשלב valgrind.
  - – **CoderPad** – סביבת תכנות מקוונת בעבר ראיונות טכניים, תומכת ב-C, מאפשרת שיתוף בזמן אמת בעבר תוכנות בזוגות, אך גם בה לא קיימת ההתקדמות בתרגול C ושימוש ב-valgrind.
  - – **Codeanywhere** – סביבת פיתוח מבוססת ענן בעבר תוכנות ב-C בה כוללות אפשרות לעריכה משותפת, טרמינל פנימי, סביבת פיתוח מוגבלות אך גם לה אין שום אזכור למידה המתקדם ב-C.
  - – **GitHub Codespaces** – סביבת פיתוח מבוססת ענן אשר תומכת ב-C, מאפשרת עריכה בשיתוף, אך גם לא מתמקד בתרגול C.
  - – **CodePen** – למרות שהיא שטח פיתוח במקור בפיטוח web, האתר כן מציע שיתוף בזמן אמת, version control וairyoch משאבים. עם זאת לא תומך ב-C.
  - בהשוואה לפתרונות אלו, הפרויקט שלי מציע יתרונות ייחודיים :
    1. ההתקדמות בשפת C עם כלים ייעודיים להדרה והרצה.
    2. שילוב של עריכה שיתופית, הרצת קוד ותרגול באוטומאטי.
    3. משקל פשוט וឥינטואיטיבי המותאם במיוחד למתחילה.
    4. מערכת תרגול מובנית עם משוב אוטומטי.
    5. אפשרויות להרחבה עתידית עם כלים נוספים כמו Valgrind לניצוח זיכרון.

## סקירת טכנולוגיות הפרויקט

הפרויקט משתמש במגוון טכנולוגיות קיימות ומשלב אותן באופן מיטבי :

### 1. צד שרת: (Backend)

- שפת Python עם מסגרת Flask לניהול בקשות HTTP
- Socket.IO לתקשורת בזמן אמת
- SQLAlchemy לzechtnת מערכות נתונים
- Flask-Bcrypt להצפנה סיסמות
- Resource, Subprocess לניהול הרצת קוד בסביבה מבודדת

### 2. צד לקוח: (Frontend)

- HTML, CSS ו-JavaScript לתקשורת עם-
- Socket.IO-client לתקשורת בזמן אמת
- CodeMirror לעריכת קוד עם הדגשת תחביר
- Bootstrap לעיצוב משקל המשתמש

### 3. תשתיות :

- Docker לניהול סביבת הפיתוח והפרישה
- PostgreSQL כבסיס נתונים

הפרויקט אינו מנסה להציג טכנולוגיות חדשות, אלא משלב טכנולוגיות קיימות באופן אפקטיבי כדי ליצור פתרון מקיים ומוכoon משתמש.

**סיגים בהגדרת המערכת****דרישות חומרה :**

- שרתים עם ביצועים גבוהים נדרשים לתמיכה בריבוי משתמשים וקומפילציה מקבילה
- רוחב פס רשת משמעותי נדרש לתמיכה בתקשורת Socket.IO בזמן אמת

**מגבליות אבטחה :**

- איזון בין פונקציונליות לאבטחה בסביבת Sandbox
- הגנה מפני התקפות DDOS בשירות מקוון

**מורכבות טכנית :**

- סyncron מדויק נדרש בין לקוחות רבים בעריכה שיתופית
- ניהול עומסים בקומפילציה והרצה קוד במקביל

**מגבליות שפה :**

- התמקדות בשפת C בלבד מגבילה את קהל היעד הפוטנציאלי

**תלות בתשתיות חיונית :**

- הסתמכות על מהדר C חיוני (GCC) יוצרת תלות בגרסאות ועדכונים

**תיכון הפרויקט****1. רשותות :**

- תקשורת בזמן אמת באמצעות IO.Socket
- REST API באמצעות HTTP/HTTPS
- אבטחת תקשורת באמצעות TLS/SSL
- סyncron נוטונים בין משתמשים רבים

**2. מערכות הפעלה :**

- סביבת הרצה מבודדת (Sandbox) לביצוע קוד
- אינטגרציה עם מערכת הפעלה לקומפילציה והרצה קוד C
- ניהול משאבי מערכת לתמיכה במשתמשים רבים

**3. אבטחת מידע :**

- יישום סביבת הרצה מבודדת (Sandbox)
- מניעת SQL Injection
- אימומות משתמשים והצפנה סיסמות
- הגבלת משאבים לקוד שמורץ במערכת

**4. עriticת קוד שיתופית :**

- יישום עriticה שיתופית בזמן אמת
- סyncron Shinonim וטיפול בהתנגשויות
- הצגת מיקום הסמן של משתמשים אחרים

**תחומיים בהם המערכת אינה עוסקת :**

1. פיתוח מהדר C עצמאי
2. יצרת סביבת פיתוח מלאה (IDE) עם כל התכונות המתקדמות
3. ניהול גרסאות קוד
4. אופטימיזציה של קוד C
5. דיבוג מתקדם של תוכניות C
6. תמיכה בשפות תכנות בלבד C

## אפיון

### תיאור מפורט של המערכת

המערכת היא סביבת עריכה והרצה מקוונת לקוד C עם יכולות עריכה שיתופית. בכניסה למערכת, המשתמש מתבקש להתחבר או להירשם. לאחר ההתחברות, המשתמש מוצג בפניו לוח מחוונים (dashboard) המציג את הפרויקטים שלו, פרויקטים מסווגים ותרגילי תכנות בהם החל.

המערכת מאפשרת ייצור פרויקטים חדשים, שיתוף פרויקטים עם משתמשים אחרים, ועובדת על תרגילים תכנות מובנים. בעת עבודה על פרויקט, המערכת מציגה עורך קוד עם הדגשת תחביר, רשימת משתמשים מחוברים, אפשרות לתקשורת באמצעות צ'אט, וכפתורים להדרכה והרצה הקוד.

הידור והרצה קוד מתבצעים בצד השרת בסביבה מבודדת, עם הגבלות על משאבי מערכת כדי למנוע שימוש לרעה. פלט ההידור והרצה מוצג למשתמש, כולל הודעות שגיאה במידת הצורך.

לצד פרויקטים אישיים, המערכת מציעה סדרת תרגילים ברמות קושי שונות. כל תרגיל כולל תיאור, קוד התחלתי, ובדיקה אוטומטית של פתרונות.

**פונקציונליות המערכת כוללת:**

1. הרשמה והתחברות משתמשים
2. יצרה וניהול פרויקטים
3. עריכת קוד C עם הדגשת תחביר
4. עריכה שיתופית בזמן אמת
5. הידור והרצה קוד C
6. צ'אט בין משתמשים
7. תרגול מובנה עם בדיקה אוטומטית של פתרונות
8. ניהול הרשאות גישה לפרויקטים

אבטחת המערכת מtaboo מושגים, הצנת סיסמאות, הרצת קוד בסביבה מבודדת, וניקוי קלט למניעת התקפות injection.

### פירוט יכולות השרת תקשות Socket.IO בזמן אמת

- ניהול חיבורים מרובים של לקוחות
- העברת עדכוני עריכה בין משתמשים בזמן אמת
- סנכרון מצב הפרויקט בין כל המשתמשים
- העברת הודעות צ'אט בין משתמשים

**הידור וביצוע קוד C בסביבה מאובטחת**

- קבלת קוד מהלקוח וניקוי קלט
- בדיקת קוד לזרויי דפוסים מסוכנים
- הידור הקוד באמצעות GCC בסביבה מבודדת
- הרצה הקוד המהוורד עם הגבלות משאבים
- איסוף פלט ההידור וההרצה והחזרתו ללוקה

**איומות משתמשים וניהול הרשאות**

- רישום משתמשים חדשים והצפת סיסמאות
- איומות משתמשים בכניסה למערכת
- יצירה וניהול של שניים
- בקרת גישה לפרויקטים לפי הרשאות

**ניהול נתוניים**

- שימירת נתונים משתמשים, פרויקטים ותרגילים במסד נתונים
- שימירת היסטוריית קומפילציה
- ניהול פרויקטים משותפים וההרשאות גישה

**ניהול עריכה משותפת**

- סנכרון שינויי עריכה בין כל המשתמשים
- טיפול בהתנסויות בעריכה מקבילית
- שימירת מיקומי סמנטים של כל המשתמשים

**אבטחה**

- הצפתה תקשורת באמצעות TLS/SSL
- בדיקת קוד לזרויי דפוסים מסוכנים
- הגבלת משאבי מערכת בהרצה קוד
- מניעת SQL Injection

**פירוט יכולות הלכה****עריכת קוד C עם הזגתת תחביר**

- עורך קוד מבוסס CodeMirror
- הדגשת תחביר C
- מספור שורות
- השלמת קוד אוטומטית

**כפיה בשיתוף פעולה בזמן אמת**

- הצגת שינוי קוד של משתמשים אחרים בזמן אמת
- הצגת מיקום הסמן של משתמשים אחרים
- רישימת משתמשים מחוברים עם מחוון זמינות

### **הפעלת הייזור והרצת קוד**

- כפטור הייזור והרצאה
- הכנסת קלט לתוכנית
- הצגת פלט ההיזור וההרצאה

### **ניהול פרויקטים**

- ייצרת פרויקט חדש
- שמירת פרויקט
- הזמנת משתמשים לשיתוף פעולה

### **תקשורות בין משתמשים**

- צ'יאט בזמן אמת בין משתמשים בפרויקט
- התראות על התחברות/התנטקות משתמשים

### **עבודה על תרגילים**

- גישה לרשימת תרגילים
- סינון תרגילים לפי רמת קושי וסטטוס
- הגשת פתרונות וקבלת משוב אוטומטי

### **פירוט יכולות המשותפות לשרת וללקוח**

#### **סנכרון שינויים עריכה בזמן אמת**

- זיהוי שינויים בקוד מצד הלקוח
- העברת השינויים לשרת
- עיבוד השינויים והפצתם לכל הלקוחות
- יישום השינויים בעורך הקוד של כל לקוח

### **טיפול בתנשויות**

- זיהוי התנשויות בעריכה מקבילית
- יישום אלגוריתם למיזוג שינויים מתנשאים
- עדכון כל הלקוחות בתוצאות המיזוג

### **שיתוף פרויקטים**

- הזמנת משתמשים לשיתוף פעולה

- הגדרת הרשות גישה
- סנכרון מצב הפרויקט בין כל המשתתפים

### ניהול תרגילים

- הצגת תרגילים ובדיקה פתרונות בצד השרת
- הצגת תרגילים וממשק הגשה בצד הלוקה
- מעקב אחר התקדמות המשתמש

### פירוט בדיקות

#### בדיקות פונקציונליות

בדיקה	מטרה	תהליכי	תוצאה מצופה	תוצאה בפועל
ערכות קוד ותוצאות ת לחבר	לודא שורץ הקוד מציג את הקוד בצורה נכונה עם הדגשת ת לחבר	הזנת קטעי קוד C שונים ובדיקה הציגם	הקוד מוצג ע"מ הדגשת תחביר מתאימה לפי סטנדרט C	הקוד מוצג כראוי ע"מ הדגשת תחביר נכונה
קומפילציה והרכבת קוד תיקין	לודא שקוד תיקין מהוורץ ומורץ כראוי	הזנת קוד C תיקין, הידור והרצה	הקוד מהוורץ בהצלחה, הפלט הוצג במסך הפלט כראוי	הקוד הורדר והוורץ בהצלחה, הפלט הוצג במסך הפלט כראוי
קומפילציה של קוד שנוי	לודא שקוד שגוי מייצר הודעות שגיאה	הזנת קוד C עם שגיאות תחביר,	הציגו הודעות שגיאה מדוקינות מההדר מתאימות	הציגו הודעות שגיאה מדוקינות מההדר מתאימות
עריכה שיתופית	לודא שמספר משתמשים יוכלים לערוך את אותו הקוד בזמן אמת	חיבור מספר משתמשים לאותו אצל כל המשתמשים שינויים	השינויים נראים אמת אצל כל המשתמשים המוחברים בזמן אמת	השינויים עוצבו בזמן אמת אצל כל המשתמשים המוחברים בזמן אמת
פתרון התנגשויות	לודא שהמערכת מטפלת נכון בהתנגשויות עrica ההתנגשויות טופלו כראוי, השינויים מוגשו באופן תקין	עריכה של אותו קטע קוד על ידי שני משתמשים בו-זמנית חלוקת, ללא איבוד מידע	התנגשויות מטפלות נכון בהתנגשויות עrica ההתנגשויות טופלו כראוי, השינויים מוגשו באופן תקין	ההתנגשויות מטפלת נכון בהתנגשויות עrica ההתנגשויות טופלו כראוי, השינויים מוגשו באופן תקין
תרגול תכנות	לודא שמערכת פתרונות, קבלת משוב גישה לתרגיל, הגשת פתרונות, קבלת משוב נבדק, קבלת משוב נבדק ומקבל משמעות	גישה לתרגיל, הגשת פתרונות, קבלת משוב נבדק, קבלת משוב משמעות	התרגיל נתן, כראוי, הפתרון נבדק והתקבל משוב מדויק	התרגיל נתן, כראוי, הפתרון נבדק והתקבל משוב מדויק
יציאת	לודא שימושים יכולים לתקשר באמצעות היציאת	שליחת הודעות בין משתמשים בפרויקט כל המשתמשים בפרויקט	ההודעות התקבלו בזמן אמת אצל כל המשתמשים בפרויקט	ההודעות התקבלו בזמן אמת אצל כל המשתמשים בפרויקט
יצירת פרויקט חדש	לודא שימושים יכול ליצור פרויקט חדש	לחיצה על "פרויקט חדש", הזנת שם ויצירה	פרויקט חדש נוצר ומוצג בلوح המחוונים	פרויקט חדש נוצר בהצלחה ומוצג בلوح המחוונים
הזנת משתמש לפרויקט	לודא שמשתמש יכול בטופס ההזמנה לפרויקט ויוכל לגשת אליו	הזנת שם משתמש בטופס ההזמנה לפרויקט ויוכל לגשת אליו	המשתמש מוזמן בהצלחה והפרויקט מופיע ברשימת	המשתמש מוזמן בהצלחה והפרויקט מופיע ברשימת

הפרויקטים המשותפים						לודא שתהליך הרישום עובד	רישיון משתמש חדש
שלו	משתמש חדש נוצר ומורבר להצלחה ומועבר ללוח המחוונים	הזנת פרטי משתמש חדש בטופס הרישום	לודא שתהליך הרישום עובד	רישיון משתמש חדש			
<b>בדיקות אבטחה</b>							
בדיקה	תוצאה בפועל	תוצאה מצופה	תהליך	מטרה	לודא שהתקשרות בין הלקוח לשרת מוצפנת	בידוקת העפננת תקשורת	רישיון משתמש חדש
העפננת תקשורת	כל התקשרות מוצפנתTLS/SSL	כל התקשרות מוצפנתTLS/SSL	בבדיקה תעבורת הרשות באמצעות כלי ניטור	לודא שתהליך הרישום עובד	לודא שהתקשרות בין הלקוח לשרת מוצפנת	לודא שהתקשרות בין הלקוח לשרת מוצפנת	רישיון משתמש חדש
אימות משתמשים	ניסיונות כנראה ניתנו לקרו את התוכן פרטם שגוים נדחוו כנדרש	המערכת דוחה ניסיונות כנישה עם פרטם שגוים	ניסיון כנישה עם כניסה לא מורשים	לודא שרק משתמשים מורשים יכולים לגשת למערכת	לודא שרק משתמשים מורשים יכולים לגשת למערכת	לודא שרק משתמשים יכולים לגשת למערכת	רישיון משתמש חדש
הרשאות גישה	ניסיונות גישה לפרוייקטים ללא הרשאה נחסמו	המערכת מונעת גישה לפרוייקטים ללא הרשאה	ניסיון גישה לפROYיקט ללא הרשאה שלם או שהוזמנו אליו	לודא שימושים יקרים לגשת רק לפרויקטים זמינים	לודא שימושים יקרים לגשת רק לפרויקטים זמינים	לודא שימושים יקרים לגשת רק לפרויקטים זמינים	רישיון גישה
Sandbox לקוד	קוד מסוכן זווה ונחמס, עם הودעה מתאימה לשימוש נחסמו, הקלט נוקה כראוי תוכניות שניסו לזרוק משאים יוטר מדי הופסקו אוטומטית	המערכת מזזה קוד מסוכן ומנע את הרצתו	ניסיון הרצת קוד זמני	לודא שקוד מסוכן לא יכול לפגוע במערכת	לודא שקוד מסוכן לא יכול לפגוע במערכת	לודא שקוד מסוכן לא יכול לפגוע במערכת	רישיון גישה
SQL Injection	ניסיונות SQL Injection נחסמו, הקלט נוקה כראוי תוכניות שניסו לזרוק משאים יוטר מדי הופסקו אוטומטית	המערכת מנקה את הקלט SQL Injection ומנעת טופס	ניסיון הזנת קלט זמני בשודות טופס	לודא שהמערכת מוגנת מפני SQL Injection	לודא שהמערכת מוגנת מפני SQL Injection	לודא שהמערכת מוגנת מפני SQL Injection	רישיון גישה
הגבלת משאים	תוכניות שניסו לזרוק מדי המשאים יוטר מדי הופסקו אוטומטית	המערכת מגבילה את צריכת המשאים וmpsika תוכניות חריגות המשאים הופסקו אוטומטית	הרצת קוד עם לולאה אינסופית או צריכת זיכרון גבואה	לודא שתוכניות לא יכולות לצורך יותר מדי המשאים יוטר מדי הופסקו אוטומטית	לודא שתוכניות לא יכולות לצורך יותר מדי המשאים יוטר מדי הופסקו אוטומטית	לודא שתוכניות לא יכולות לצורך יותר מדי המשאים יוטר מדי הופסקו אוטומטית	רישיון גישה
בדיקות ביצועים	<b>בדיקות ביצועים</b>						רישיון גישה
בדיקה	תוצאה בפועל	תוצאה מצופה	תהליך	מטרה	לודא שהמערכת מוגנת מפני SQL Injection	לודא שהמערכת מוגנת מפני SQL Injection	רישיון גישה
עומסים זמיית תנובה	המערכת תפקדה באופן יטיב לתפקיד יכיב עד 50 משתמשים בו-זמנית ביציבות זמנית זמני תגובה קצרים ועקביים 500 מילישניות לרוב הפעולות	סימולציה של מספר רב של משתמשים בו-זמנית	מתפקדת היטב תחת עומס זמיית תנובה	לודא שהמערכת מוגנת מפני SQL Injection	לודא שהמערכת מוגנת מפני SQL Injection	לודא שהמערכת מוגנת מפני SQL Injection	רישיון גישה

רשות רגילים	פחות משנהיה	מסוכנים תוך	השינויים הופיעו בתוך	mdiedit	לודא שעדכוני
300	מילישניות בתנאי			עריכה מסוכנים	בזמן אמת

## תכנון וניהול לוח זמנים

שלב	תיאור	זמן מתוכנן	זמן בפועל	סטטוס
1	תכנון וחקר	2 שבועות	3 שבועות	הושלם
2	הקמת תשתיית בסיסית	1 שבוע	1 שבוע	הושלם
3	מערכת אימוט וניהול משתמשים	1.5 שבוע	1 שבוע	הושלם
4	עורך קוד בסיסי	2 שבועות	2 שבועות	הושלם
5	תקשורת WebSocket	1 שבוע	2 שבועות	הושלם
6	עריכה שיתופית	3 שבועות	4 שבועות	הושלם
7	קומפילציה והרצה	2 שבועות	3 שבועות	הושלם
8	מערכת תרגילים	2.5 שבועות	2.5 שבועות	הושלם
9	פאנל ניהול	1 שבוע	1 שבוע	הושלם
10	בדיקות ותיקונים	2 שבועות	3 שבועות	הושלם

## ניהול הסיכון

הסיכון	פירוט הסיכון	רמת הסיכון	דרכי התמודדות	איך טופל בפועל
קורשי בסנסרין	הנתגשויות בעריכה עלולות לגרום לאיבוד מידע	גבוהה	שימוש באlgorigthms מיוזג מוקדם, ניטור ערכיות בזמן אמת	יעיל, נספה תוצאות סטטיסטicas של משתמשים אחרים
עריכה שיתופית	קוד זמני עלול לפרוץ מה Sandbox	גבוהה	הגבלת משאבים, סיינון בסביבה מבודדת	ሞמש מנגנון בדיקת קוד מסוכן, הוגבלו משאבי מערכת, נספו בדיקות אבטחה
אבטחת Sandbox	ריבוי משתמשים על להעמיס על הרשת	בינונית	תכנון לסקלibility, מנגנוני איזון עומסים	שימוש בתכורת Docker, מוכנה לסקלibility, מיטוב שאלות מס' נתונים -Bootstrap-
עומס על הרשת	נמוכה	בדיקות תאימות, שימוש בספריות סטנדרטיות בין דפדפניים	יתכנו הבדלים בתցוגה וב��פוקוד	תאימות, בדיקות ב-3 דפדים נפוצים
בעיות תאימות בדףנים	חיבורם בזמן אמת עלולים להיות מאתגרים	בינונית	למידה עמוקה של Socket.IO, קדני של פרוטוקול ההודעות	הוקדש זמן למידת Socket.IO נכתב פרוטוקול מבנה ומתחודד היבט
פתרונות בטעמאות Socket.IO	שימוש בטכנולוגיות גיבויים תכופים	בינונית	שימוש בטרנזקציות, גיבויים תכופים	ישמו טרנזקציות לכל פעולות העדכון, הוקם מנגנון גיבוי אוטומטי
בעיות במסד הנתונים	mdiupdate	mdiupdate	mdiupdate	

## תיאור תחום הידע

### יכולות מצד השירות

#### ניהול משתמשים ואמוות

**שם היכולת:** הרשמה למערכת

**מהות:** רישום משתמש חדש במערכת עם קליטת פרטיים אישיים נדרשים ובדיקה תקינות הנתונים.

**אוסף יכולות נדרשות:**

- ממשק משתמש – מסך הרשמה עם שדות קלט
- קליטת נתונים – שם משתמש, אימייל וסיסמה
- בדיקת תקינות – ולידציה של פורמט אימייל ועוצמת סיסמה
- הצפנה – הצפנה הסיסמה באמצעות bcrypt
- שליחה לשרת – העברת הנתונים לשרת באמצעות HTTP POST
- בדיקת תקינות מול בסיס הנתונים – בדיקת ייחודיות שם משתמש ואימייל
- קבלת תשובה מהשרת – קבלת אישור או הודעה שגיאה
- הצגת התשובה למשתמש – הצגת הודעה הצלחה או שגיאה

**אובייקטים נחוצים:** מודל user, ממשק הרשמה, מנגן הצפנה, מסד נתונים

**שם היכולת:** התחברות למערכת

**מהות:** אימוות זהות משתמש קיים וכניסה למערכת.

**אוסף יכולות נדרשות:**

- ממשק התחברות עם שדות שם משתמש וסיסמה
- ולידציה בסיסית של נתונים
- שליחת בקשה לשרת
- בדיקת קיום המשתמש במסד הנתונים
- אימוות הסיסמה מול הגרסה המוצפנת
- יצרת הפעלה (session) למשתמש
- הפניה לדף הראשי או הצגת שגיאה

**אובייקטים נחוצים:** מודל user, מנגן session, מנגן אימוות

#### ניהול פרויקטים

**שם היכולת:** יצרת פרויקט חדש

**מהות:** יצרת פרויקט קוד חדש עם הגדרת בעלות והרשות.

**אוסף יכולות נדרשות:**

- קליטת שם פרויקט ותיאור
- ולידציה של ייחודיות שם הפרויקט
- ייצור רשומה חדשה ב景德 הנתונים
- קביעת בעלות על הפרויקט
- הגדרת רשות ברירת מחדל
- החזרת מזהה פרויקט למשתמש

**אובייקטים נחוצים:** מודל project, מודל user, מנגנון הרשות

**שם היכולת:** הזמנת משתמשים

**מהות:** הוספה משתמשים נוספים לפרויקט קיימים עם הגדרת הרשות.

**אוסף יכולות נדרשות:**

- בדיקת הרשות המזמין (בעליים או מנהל)
- חיפוש המשתמש המזמין ב景德 הנתונים
- בדיקה שהמשתמש לא כבר חבר בפרויקט
- הוספה המשתמש לרשימת המשתמשים
- הגדרת רמת הרשות (קריאיה/ כתיבה/ניהול)
- שליחת הודעה למשתמש המזמין

**אובייקטים נחוצים:** מודל user, מודל project, מנגנון הרשות, מערכת הודעות

**עריכה שיתופית בזמן אמת**

**שם היכולת:** סנכרון שינויי קוד

**מהות:** העברת שינויי קוד בין כל המשתמשים בפרויקט בזמן אמת תוך שמירה על עקביות.

**אוסף יכולות נדרשות:**

- קליטת פעולות עריכה מהמשתמש (הוספה/מחיקה/החלפה)
- הפיכת הפעולה לפורמט Operational Transformation
- העברת הפעולה לכל המשתמשים בפרויקט
- ישום הפעולה על המסמך המקומי
- פתרון התנגשויות בעריכה
- שמירת המסמך המעודכן

**אובייקטים נחוצים:** מנגנון ניהול מצב WebSocket, אלגוריתם OT, מנגנון ניהול ניהול

**שם היכולת:** מעקב מיקום cursor

**מהות:** הצגת מיקום cursor של כל משתמש אחרים.

**אוסף יכולות נדרשות:**

- קליטתת מיקום cursor מהמשתמש
- שידור המיקום למשתמשים אחרים
- הצגת cursor משמשים אחרים
- עדכון מיקום בזמן אמת
- הסרת cursor עם התנטקות משתמש

**אובייקטים נחוצים:** מנגןו WebSocket , ממשק עורך הקוד

**קומPILEציה והרצה של קוד C**

**שם היכולת:** קומPILEציה מאובטחת

**מהות:** קומPILEציה קוד C שנכתב על ידי משתמש תוך הבטחת אבטחה.

**אוסף יכולות נדרשות:**

- קבלת קוד מהמשתמש
- בדיקת הקוד לתבניות מסווגות
- ייצור קובץ זמני בסביבה מבודדת
- הפעלת מקמפל GCC עם פרמטרים מוגבלים
- קליטת הודעות שגיאה וזמן קומPILEציה
- מהיקת קבצים זמניים
- החזרת תוצאות קומPILEציה

**אובייקטים נחוצים:** מנגןו sandbox , מקמפל GCC , מעבד קבצים זמניים

**שם היכולת:** הרצה מבודדת

**מהות:** הרצת הקוד הקומפל בסביבה מבודדת ומוגבלת.

**אוסף יכולות נדרשות:**

- ייצור תהליך מבודד
- הגבלת משאבים (זיכרון, זמן)
- הפעלת הקובץ הבינארי
- קליטת פלט ושגיאות
- הפסקת תהליך במקרה של חריגה
- מהיקת קבצים זמניים
- החזרת תוצאות הרצה

**אובייקטים נחוצים:** מנגןו sandbox , מגביל משאבים, מעבד O/I

**מערכת תרגילים****שם היכולת:** ייצרת תרגיל חדש**מהות:** ייצרת תרגיל תכנות עם הגדרות מקרבי בדיקה וקריטריונים.**אוסף יכולות נדרשות:**

- ממשק ייצרת תרגיל עם שדות מתאימים
- הגדרת רמת קושי וקטgorיה
- הוספת מקרי בדיקה (קלט/פלט צפוי)
- הגדרת קוד התחלתי אופציונלי
- שימירת פתרון מלא לתרגיל
- ולידציה של נתוני התרגיל

**אובייקטים נחוצים:** מודל TestCase , מודל Exercise , ממשק ייצירה**שם היכולת:** בדיקה אוטומטית של פתרון**מהות:** בדיקת פתרון תלמיד מול מקרי הבדיקה של התרגיל.**אוסף יכולות נדרשות:**

- קבלת קוד פתרון מהתלמיד
- קומפילציה של הפתרון
- הרצת הפתרון עם כל מקורה בדיקה
- השוואת הפלט לפلت הצפוי
- חישוב ציון או אחוזי הצלחה
- שימירת תוכאות הבדיקה
- עדכון התקדמות התלמיד

**אובייקטים נחוצים:** מנגן קומפילציה, מעבד מקרי בדיקה, מודל Progress**פאנל ניהול****שם היכולת:** ניהול משתמשים**מהות:** מתן כלים למנהל לניהול משתמשי המערכת.**אוסף יכולות נדרשות:**

- הצגת רשימה כל המשתמשים
- חיפוש וסינון משתמשים
- צפייה בפרטי משתמש ופעולותיו
- חסימה או ביטול חסימה של משתמש

- איפוס סיסמה
- מחיקת משתמש

**אובייקטים נחוצים:** מודל User , ממשק ניהול, מגנון הרשאות

**שם היכולת:** סטטיסטיות מערכת

**מהות:** הצגת נתונים סטטיסטיים על שימוש המערכת.

**אוסף יכולות נדרשות:**

- ספירת משתמשים פעילים
- ניתוח דפוסי שימוש
- מעקב ביצועי מערכת
- גרפים וטבלאות סטטיסטיקה
- דוחות תקופתיים
- התראות על בעיות

**אובייקטים נחוצים:** מגנון analytics , גנרטור דוחות, מסד נתונים.

## יכולות מצד הלקו

**ממשק משתמש אינטראקטיבי**

**שם היכולת:** ניווט המערכת

**מהות:** מתן ממשק נוח וידידותי לניווט בין חלקי המערכת השונים.

**אוסף יכולות נדרשות:**

- תפריט ראשי עם קישורים לחלקים שונים
- breadcrumb navigation למעקב מיקום
- תפריטי צד ודרופ-דאון
- כפתורי חזרה וAMPLE
- חיפוש גלובלי במערכת
- עיצוב רספונסיבי

**אובייקטים נחוצים:** UI, CSS framework, JavaScript events

**שם היכולת:** הודעות מושב

**מהות:** הצגת הודעות מושב למשתמש על פעולות שביצע.

**אוסף יכולות נדרשות:**

- הודעות הצלחה בצבע ירוק

- הודעות שגיאה בצבע אדום
- הודעות אזהרה בצבע כתום
- הודעות מידע בצבע כחול
- אנימציות הופעה והעלמה
- סגירה אוטומטית לאחר זמן

**אובייקטים נחוצים:** notification, CSS animations, JavaScript timers

#### עורך קוד מתקדם

**שם היכולת:** הדגשת syntax

**מהות:** הדגשת קוד C עם צבעים המתאימים לרכיבי השפה השונים.

**אוסף יכולות נדרשות:**

- זיהוי מילים מפתח של C
- הדגשת מחרוזות ומספריים
- הדגשת הערות בקוד
- הדגשת סוגרים תואמים
- הדגשת שגיאות syntax
- תמיכה בעדכון בזמן אמת

**אובייקטים נחוצים:** מנוע syntax highlighting , רגולר אקספרסנס , עורך קוד expressions

**שם היכולת:** השלמה אוטומטית

**מהות:** הצעת השלמות לקוד בזמן הקלדה.

**אוסף יכולות נדרשות:**

- זיהוי הקשר הקלדה
- הצעת מילים מפתח רלוונטיות
- הצעת שמות פונקציות ומשתנים
- הצגת רשימה אפשרויות
- בחירה באמצעות מקלדת או עכבר
- הכנסת ההשלמה לקוד

**אובייקטים נחוצים:** מאגר מילים מפתח UI popup , autocomplete engine,

#### תקשורות בזמן אמת

**שם היכולת :** חיבור WebSocket

**מהות:** ייצור וניהול חיבור WebSocket יציב לשרת.

**אוסף יכולות נדרשות:**

- יצרת חיבור WebSocket לשרת
- טיפול באירועי חיבור ונטק
- שליחת הודעות לשרת
- קבלת הודעות מהשרת
- אוטומטי במקרה נטק Reconnection
- ניהול מצב החיבור

**אובייקטים נחוצים:** WebSocket API, event handlers, reconnection logic

**שם היכולת:** סנכרון עדכוני

**מהות:** קבלת עדכוני קוד ממשתמשים אחרים וישום בעורך המקומי.

**אוסף יכולות נדרשות:**

- קבלת פעולות עריכה מהשרת
- ישום הפעולות על הקוד המקומי
- שימירת מיקום cursor נוכחי
- עדכון הקוד ללא הפרעה לעריכה
- התמודדות עם התנשויות consistency
- שימירת consistency

**אובייקטים נחוצים:** code editor, API event handlers, OT algorithm

**ניהול מצב אפליקציה****שם היכולת:** ניהול state מקומי

**מהות:** ניהול מצב האפליקציה מצד הלוקוט.

**אוסף יכולות נדרשות:**

- שימירת נתונים משתמש מחובר
- שימירת רשימת פרויקטים
- שימירת מצב עורך הקוד
- שימירת העדפות משתמש
- סנכרון עם שרת בעת הצורך
- ניקוי נתונים בהתקנות

**אובייקטים נחוצים:** state management, local storage, session management

## יכולות משותפות

*WebSocket*  
פרוטוקול תקשורת

**שם היכולת:** ניהול הודעות WebSocket

**מהות:** הגדרה ויישום של פרוטוקול תקשורת מובנה בין לקוח ושרת.

**אוסף יכולות נדרשות:**

- הגדרת פורמט הודעות JSON
- routing של הודעות לפי type
- ולידציה של תוכן הודעות
- טיפול בשגיאות תקשורת
- logging של הודעות
- spam throttling למניעת

**אובייקטים נחוצים:** JSON parser, message router, validator

## ניהול הפעולות משותפות

**שם היכולת:** Operational Transformation

**מהות:** אלגוריתם לסינכרון עריכה שיתופית ללא התנגשויות.

**אוסף יכולות נדרשות:**

- הגדרת פעולות עריכה (insert/delete/retain)
- transformation של פעולות מתנגשות
- composition של פעולות רצופות
- שמירת היסטוריה לביטול
- ולידציה של עקביות
- recovery מכישלונים

**אובייקטים נחוצים:** Operation classes, Transform engine, History manager

## אבטחת מידע

**שם היכולת:** validation ו- sanitization

**מהות:** ויזוא תקינות ובטיחות של נתונים הנכנסים למערכת.

**אוסף יכולות נדרשות:**

- ולידציה של פורמט נתונים
- sanitization של תוכן מסוכן

- בדיקת אורך ותוכן שדות
- מניעת XSS וSQL injection
- הצפנה נתוניים רגשיים
- logging של ניסיונות התקפה

**אובייקטים נחוצים:** validator functions, sanitizer, encryption utilities

## ארכיטקטורה של הפרויקט

המערכת מתוכננת כסביבת הרצה מקוונת לשפת C עם דגש על למידה, תרגול ועבודה שיתופית.

### הארכיטקטורה

המערכת מחולק לשולש שכבות עיקריות:

#### Frontend

שכבה הציגת מיושמת באמצעות CSS, HTML ו- JavaScript וספקת ממשק משתמש ידידותי עם העורך קוד, אוצר תרגול, אוצר פלט וצ'אט. היא מתקשרת עם שכבה היישום באמצעות HTTP ו-Socket.IO.

#### רכיבים עיקריים:

- **עורך קוד:** מבוסס על CodeMirror עם תמיכה בהדgestה תחביר C וסנכרון בזמן אמת
- **אוצר תרגולים:** מציג תרגילים ומאפשר למשתמשים לעבוד עליהם
- **אוצר פלט:** מציג את תוכאות ההיוזר וההרצת צ'אט
- **צ'אט:** מאפשר תקשורת בין משתמשים
- **לוח מחוונים:** מציג את הפרויקטים של המשתמש ומאפשר ניהול שלהם

#### Backend

שכבה היישום מיושמת באמצעות Python עם מסגרת Flask ומטפלת בבקשתות מהלקוחות, בהידור והרצה קוד, בסנכרון שניינים ובתקשורת עם מסד הנתונים.

#### רכיבים עיקריים:

- **בקר משתמשים:** מטפל ברישום, התחברות וניהול משתמשים
- **בקר פרויקטים:** מטפל ביצירה, עERICA וניהול של פרויקטים
- **בקר תרגולים:** מטפל בהציגה ובדיקה של תרגילים
- **מנגנון היוזר והרצתה:** מהדר ומריצ קוד C בסביבה מאובטחת
- **מנגנון סנכרון:** מטפל בסנכרון שניינים בין ל��וחות
- **Socket.IO Server:** מאפשר תקשורת בזמן אמת בין ל��וחות לבין עצם

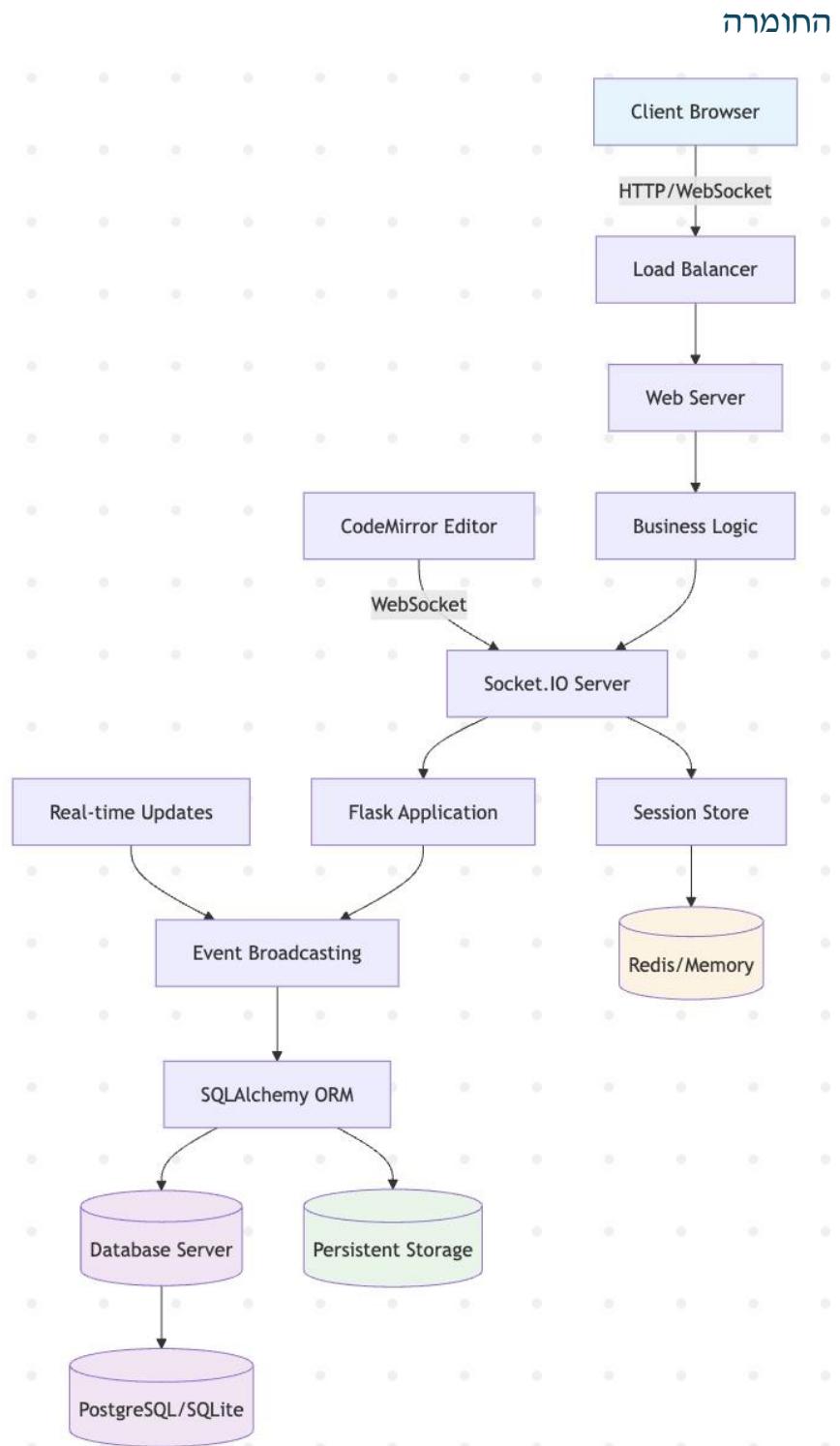
#### Database

שכבה הנתונים מיושמת באמצעות מסד נתונים SQL עם ממשק SQLAlchemy ומ אחסנת את כל הנתונים של המערכת.

#### רכיבים עיקריים:

- **טבלת משתמשים:** שומרת מידע על משתמשי המערכת
- **טבלת פרויקטים:** שומרת מידע על פרויקטים של משתמשים
- **טבלת תרגולים:** שומרת מידע על תרגילי תוכנות
- **טבלת התקדמות:** שומרת מידע על התקדמות משתמשים בתרגילים

- **טבלת ההיסטוריה קומפילציה:** שומרת מידע על ניסיונות קומפילציה והרצתה



המערכת מתוכננת לזרע על שרת עם משאבים מסוימים לתמיכה בריבוי משתמשים וקומפילציה מקבילית.  
הדרישות המינימליות הן :

- מעבד : 4 ליביות או יותר
- זיכרון : 8 GB RAM או יותר
- אחסון : 50 GB או יותר
- רוחב פס רשת : לפחות 100 Mbps

בסביבת הפיתוח, המערכת יכולה לזרע על מחשב מקומי עם Docker שמספק סביבה מבודדת להרצת השרת ומסד הנתונים.

## הטכנולוגיה הפלטפורמית

### שפות תכנות

שפה	שימוש	nymok הבחירה
<b>Python</b>	Backend development	גמישות, ספריות עשירות, קהילה גדולה
<b>JavaScript</b>	Frontend development	שפת הדף הסטנדרטית, אкосיסטם עשיר
<b>HTML5</b>	מבנה דפים	סטנדרט מודרני עם תמיכה בתכונות מתקדמות
<b>CSS3</b>	עיצוב ממשק	עיצוב מתקדם וRESPONSIVE
<b>C</b>	קוד המורץ במערכת	השפה שהמערכת מיועדת ללמד

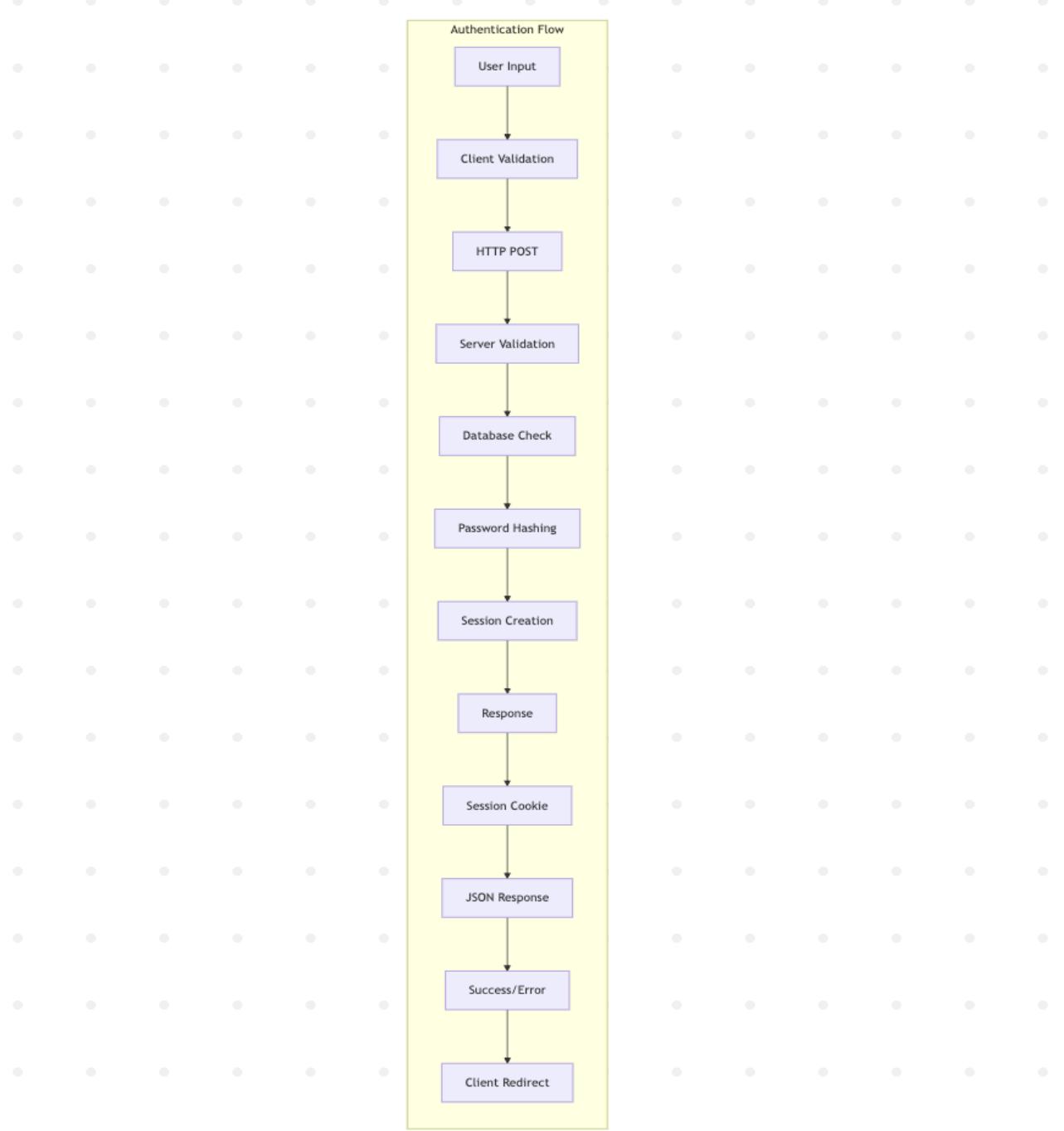
### מסגרות ותשתיות

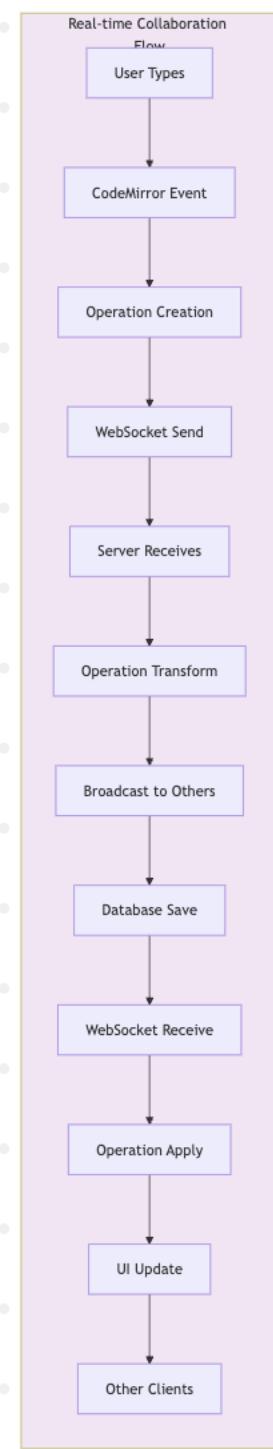
מסגרת	תפקיד	יתרונות
<b>Flask</b>	Web framework	פשוט, גמיש, מהיר לפיתוח
<b>Socket.IO</b>	Real-time communication	תקשורת דוחה-זמןית יציבה
<b>SQLAlchemy</b>	ORM	ניהול מסד נתונים מובנה
<b>CodeMirror</b>	Code editor	עיצוב קוד מתקדם וモותאם
<b>Bootstrap</b>	CSS framework	עיצובレスpondive מהיר

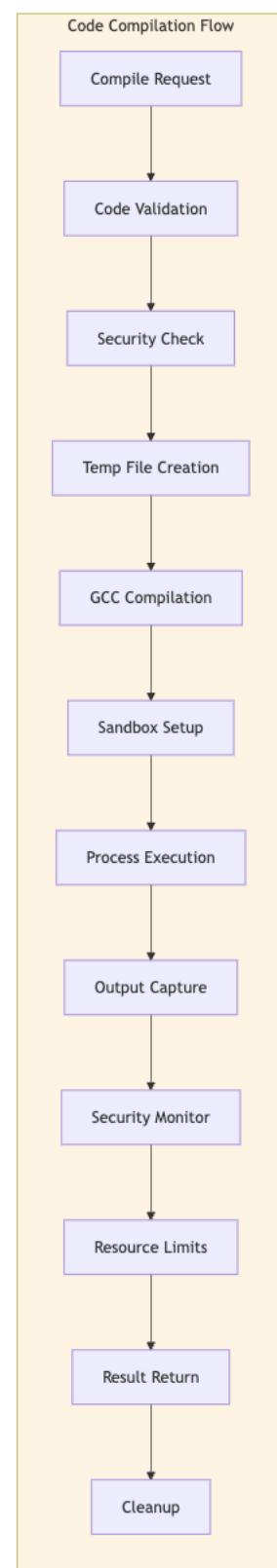
### כליים ומנגנונים נוספים

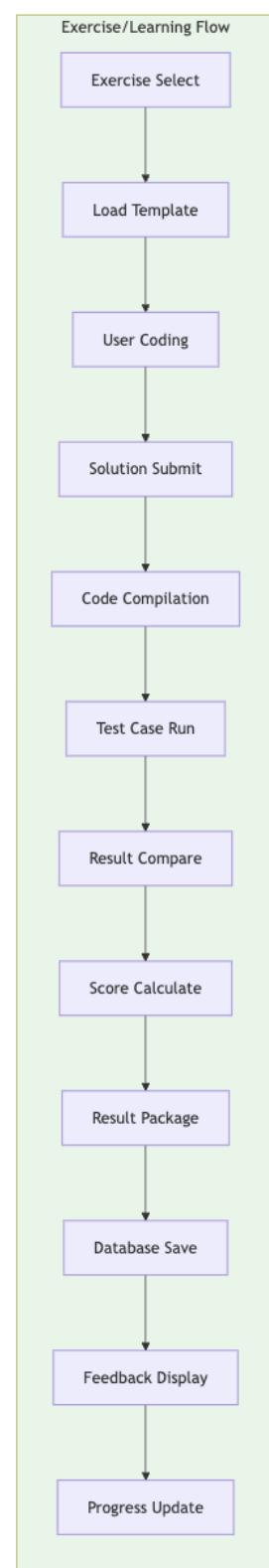
- Docker : לארגון סביבת הפיתוח והריצה
- nginx : כשרת HTTP וכ-HTTP reverse proxy
- Git : לביקורת גרסאות
- pytest : לבדיקות אוטומטיות
- bcrypt : להצפנה סיסמות
- Valgrind : לבדיקת זליגות זיכרון בקוד C

**זרימת המידע במערכת**  
**הרשמה והתחברות**



**עריכת קוד**

**הידור והרצת קוד**

**עבודה על תרגיל**

## האלגוריתמים

### עריכה שיתופית בזמן אמת

האתגר המרכזי בעריכה שיתופית הוא לשמור על סyncron בין כל המשתמשים, תוך טיפול בשינויים מקבילים והתנגשויות. המערכת משתמשת במנגנון מבוסס על שינויים מקומיים, שנשלחים לשרת ומופצים לכל המשתמשים האחרים.

השרת מנהל מצב מרכזי של כל פרויקט, ומטפל בשינויים לפי הסדר בהם הגיעו. כאשר מתרחשת התנגשות, השרת מיישם אלגוריתם מיזוג שמנסה לשלב את השינויים של כל המשתמשים באופן היגיוני.

האלגוריתם מבוסס על שלושה סוגים פועלות ערך:

1. **הוספה**: הוספת טקסט חדש בנקודת מסויימת
2. **מחיקה**: מחיקת טקסט קיים מנקודה מסוימת
3. **החלפה**: החלפת כל הטקסט בטקסט חדש

כל פעולה, השרת מחשב את ההשפעה שלה על מצב הפרויקט הנוכחי, וمعدכן את המצב בהתאם. השינויים המעודכנים מופצים לכל המשתמשים, כך שכולם רואים את אותו מצב.

### בדיקות קוד מסוכן

כדי להגן על המערכת מפני קוד זוני, המערכת מיישמת אלגוריתם לבדיקת קוד מסוכן. האלגוריתם מוחפש דפוסים ידועים של קוד C שעלול לסכן את המערכת, כגון:

- קריאות מערכת כמו `system()`, `popen()`
- פעולות על קבצים כמו `fopen()`, `fwrite()`, `fprintf()`
- פעולות תהליכיים כמו `exec()`, `fork()`
- פעולות על מערכות הקבצים כמו `unlink()`, `remove()`, `rename()`

האלגוריתם משתמש בביטויים רגולריים כדי לזהות את הדפוסים האלה בקוד המוגש. אם נמצא דפוס מסוכן, המערכת מסרבת להריץ את הקוד ומציגת הודעה מתאימה למשתמש.

יתרונו של פתרון זה הוא שהוא פשוט ויעיל בזיהוי דפוסים ידועים. חסרונו הוא שהוא עלול לסנן גם קוד לגיטימי שימוש בפונקציות אלה למטרות חוקיות, ושהוא עלול להחמיר דפוסים מסוכנים שאינם מכוסים על ידי הביטויים הרגולריים.

```
def check_for_dangerous_code(code):
    dangerous_patterns = [
        r'system\s*\(\w+',      # קריאות system()
        r'popen\s*\(\w+',       # popen()
        r'fopen\s*\(\w+',       # פעולות קובץ
        r'fwrite\s*\(\w+',      # פעולות כתיבה לקובץ
        r'fprintf\s*\(\w+',     # פעולות כתיבה לקובץ
        r'exec[\w][pe]\[\w*',   # משפחת פונקציות exec
```

```
r'fork\b|\(|,      #יצירת תחביבים
r'unlink\b|\(|,   #מחיקת קבצים #
r'remove\b|\(|,   #מחיקת קבצים #
r'rename\b|\(|,   #שינוי שם קבצים #
r'mkdir\b|\(|,    #יצירת תיקיות #
r'rmdir\b|\(|,    #מחיקת תיקיות #
]
```

```
import re
for pattern in dangerous_patterns:
    if re.search(pattern, code):
        return True
return False
```

### הגבלת משאבים בהרצה קוד

כדי למנוע שימוש לרעה במשאבי המערכת, המערכת מיישמת אלגוריתם להגבלת משאבים בהרצה קוד. האלגוריתם משתמש במודול Python של `resource` כדי להגביל:

- זמן CPU מניעת לולאות אינסופיות
- שימוש בזכרון: מניעת דליפות זיכרון ומתקפות "שייטפון זיכרון"
- גודל קובץ: מניעת ייצור קבצים גדולים

```
def set_resource_limits():
    resource.setrlimit(resource.RLIMIT_AS, (32 * 1024 * 1024, 32 * 1024 * 1024))
    resource.setrlimit(resource.RLIMIT_CPU, (2, 2))
    resource.setrlimit(resource.RLIMIT_FSIZE, (1024 * 1024, 1024 * 1024))
```

מוגבלות אלה מיושמות בכל הרצת קוד, באמצעות הפקט `popen` של `subprocess`. פונקציה בתהיליך הבן לפני הרצת התוכנית.

### בדיקות אוטומטיות של פתרונות תרגילים

המערכת מיישמת אלגוריתם לבדיקה אוטומטית של פתרונות לתרגילים. האלגוריתם מריץ את הפתרון של המשמש עם מספר מקרי בדיקה מוגדרים מראש, ומשווה את הפלט של הפתרון עם הפלט המוצופה.

התהיליך כולל את השלבים הבאים:

1. הידור הקוד של המשתמש
2. הרצת הקוד עם כל מקרה בדיקה
3. השוואת הפלט בפועל עם הפלט המוצופה
4. ייצור דוח תוצאות עם סטטוס כל מקרה בדיקה

מקרי הבדיקה מוגדרים בפורמט JSON שכלל:

- קלט למקורה הבדיקה
- פלט מצופה
- תיאור אופציונלי של מקורה הבדיקה

## אלגוריתמים קיימים

**עבור עריכה שיתופית:**

- Google's Differential Synchronization
- Apache Wave's Operational Transformation
- ShareJS Implementation

**עבור הרצה מאובטחת:**

- Docker Containers
- LXC (Linux Containers)
- chroot jails

## מקורות רלוונטיים

### למה ? Operational Transformation

- **יציבות:** אלגוריתם מוכח בשימוש במערכות כמו Google Docs
- **פשטות:** יחסית פשוט ליישום לעומת אלטרנטיבות
- **ビizioוים:** תגובה מהירה לשינויים

### למה ? Linux rlimit

- **יעילות:** מוגנה במערכת הפעלה, אמין ומהיר
- **גמישות:** אפשרות להגדיר הגבלות מדויקות
- **בטחה:** הגנה ברמת הליבה

## סביבת הפיתוח

### כלי פיתוח

- **Visual Studio Code :** בעבר בו לכתיבה קוד Frontend, Backend ובדיקות.
- **GCC (מהדר C) :** לביצוע קומpileציה של קוד C-ב-Backend.
- **Git (בקורת גרסאות) :** לניהול השינויים בקוד.
- **Make (כלי בניה) :** להגדרת תהליכי ההידור והבנייה שלuproיקט.

- CUnit (ספריית בדיקות C) : לבדיקות יחידה לקוד C.

### פירוט הסביבה וכלי בדיקה

- macOS : מערכת ההפעלה של השרת.
- Frontend Apache/Nginx (שרת אינטרנט) : להגשת ה-HTML.
- MySQL/PostgreSQL (בסיס נתונים) : לאחסון נתונים.
- Jest/Mocha (ספריות בדיקות JavaScript) : לבדיקות יחידה וaintegration ל-JavaScript.
- Cypress/Selenium (כלי בדיקות E2E) : לבדיקות מקצה למקצה.
- Valgrind : לניתוח זיכרון.

### פרוטוקול התקשרות

המערכת משתמש בשני פרוטוקולי תקשורת עיקריים :

1. HTTP/HTTPS (REST API) לבקשת סינכרונית
2. WebSocket לתקשורת אסינכרונית בזמן אמת

**מבנה הודעה WebSocket :** כל הודעה היא אובייקט JSON עם המבנה :

```
{
  "type": "message_type",
  "data": {
    "field1": "value1",
    "field2": "value2"
  },
  "timestamp": "ISO8601_timestamp",
  "user_id": "user_identifier"
}
```

### פירוט כלל החודעות

מחלקו לשרת :

שם הודעה	מטרה	שלחת מ	שלחת אל	מבנה השדות
connect	חיבור לשרת	Client	Server	{project_id: string}
join_project	הצטרפות לפרויקט	Client	Server	{project_id: string}
edit	פעולות עריכה	Client	Server	{type: string, position: number, text: string, project_id: string}
cursor_move	תוווזות cursor	Client	Server	{project_id: string, position: {line: number, ch: number} }
chat_message	הודעת צ'אט	Client	Server	{project_id: string, message: string}

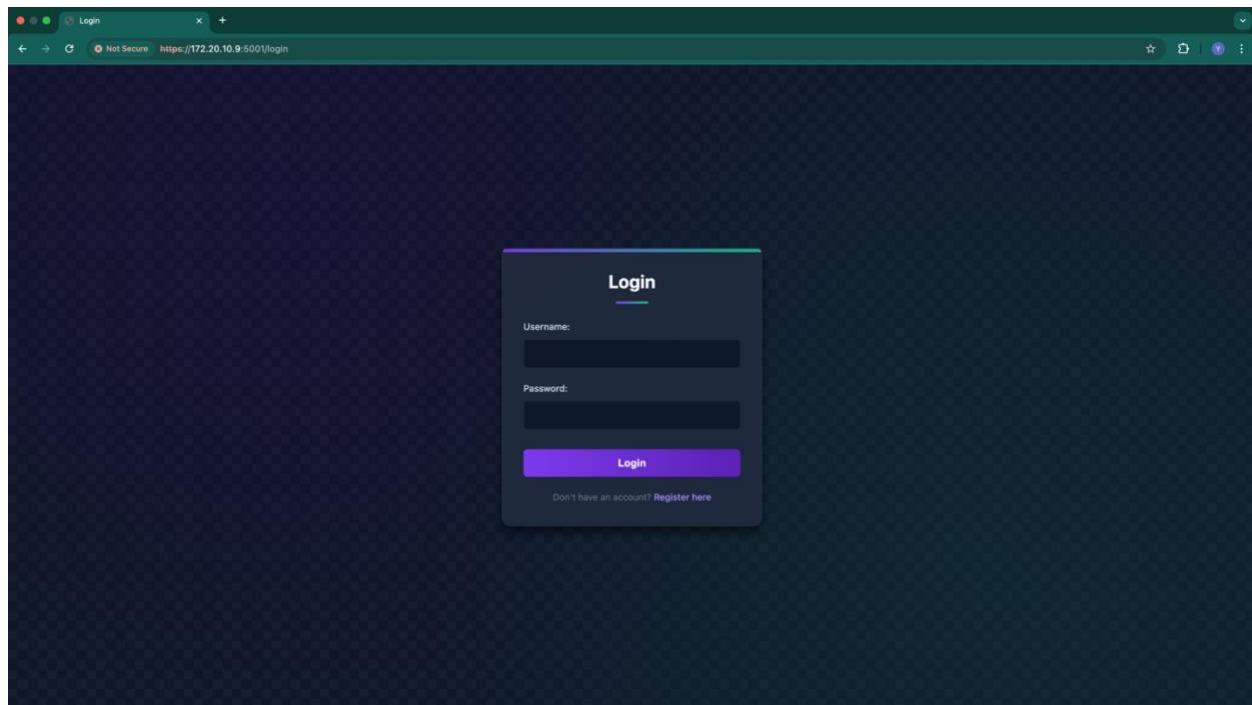
## מהירות ללקוח:

מבנה השדות	נשלחת אל	נשלחת מ	מטרה	שם ההודעה
{text: string}	Client	Server	סינכרון מסמך	document
{username: string, user_id: string, sid: string}	All Clients	Server	משתמש התחבר	user_connected
{username: string, sid: string}	All Clients	Server	משתמש התנתק	user_disconnected
{user_id: string, username: string, position: object}	Other Clients	Server	עדכון קורסור	cursor_update
{user_id: string, username: string, message: string, timestamp: string} {message: string}	All Clients	Server	הודעת צ'אט	new_chat_message
	Client	Server	שגיאות עריכה	edit_error

## מסכי המערכת

המערכת כוללת מספר מסכים מרכזיים :

## מסך כניסה (Login)



**תפקיד המסך:** מותן אפשרות למשתמשים רשומים להתחבר למערכת.

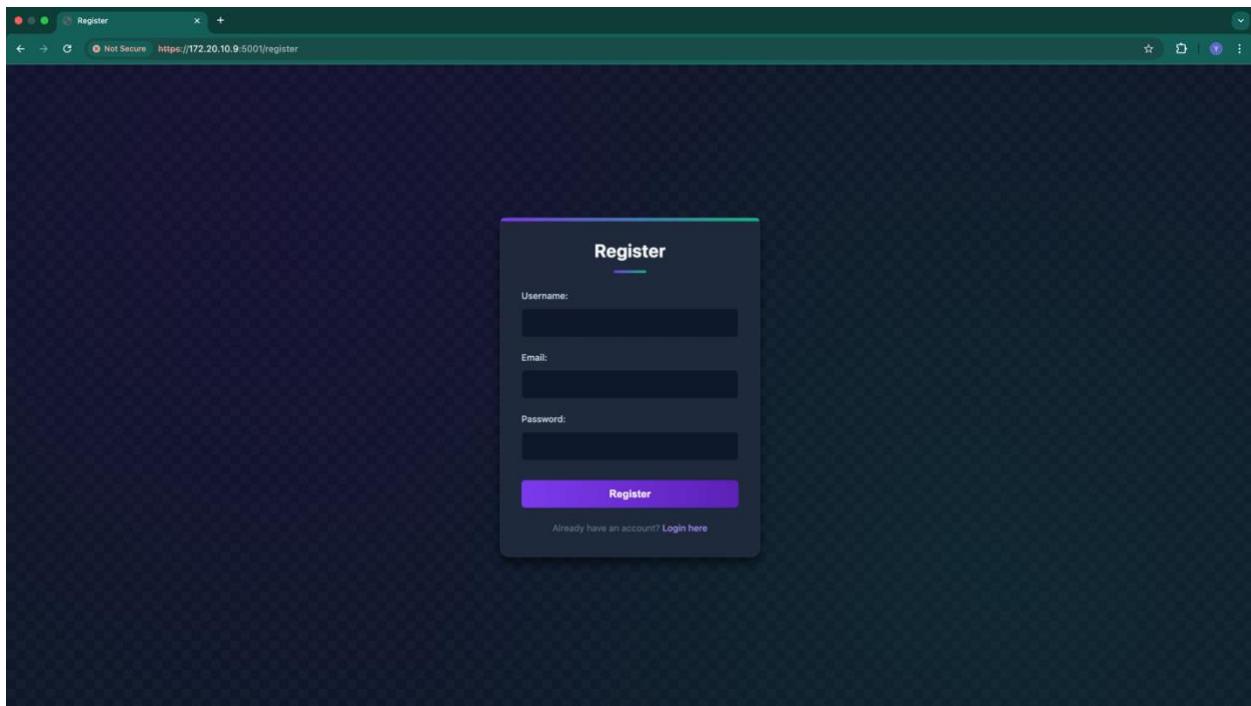
**מידע מוצג:**

- שדה קלט לשם משתמש
- שדה קלט לסיסמה
- כפטור התחברות
- קישור למסך הרשמה
- הודעות שגיאה במידת הצורך

**פעולות זמינות:**

- הזנת פרטי התחברות
- לחיצה על כפטור התחברות
- מעבר למסך הרשמה

### מסך הרשמה (Register)



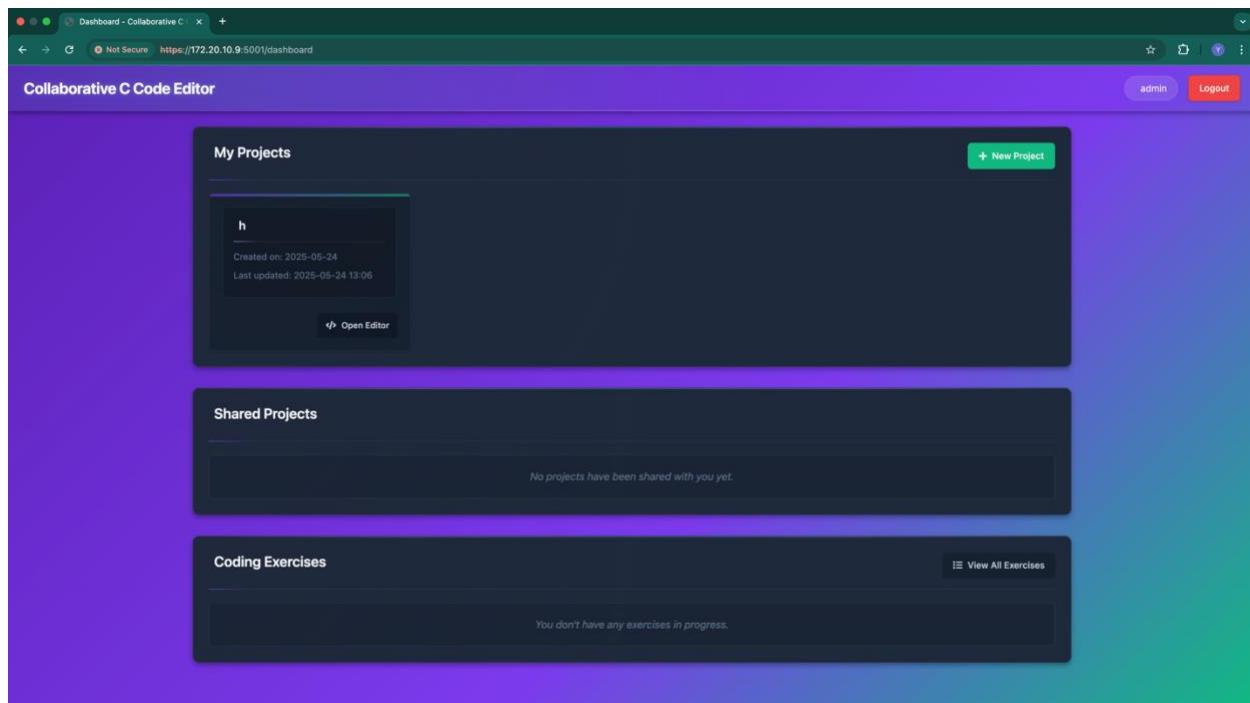
**תפקיד המסך:** מtran אפשרות למשתמשים חדשים להירשם למערכת.

**מידע מוצג:**

- שדה קלט לשם משתמש
- שדה קלט לכתובת אימייל
- שדה קלט לסיסמה
- כפטור הרשמה
- קישור למסך התחברות
- הודעות ולידציה

**פעולות זמיינות:**

- הונת פרטי הרשמה
- ולידציה בזמן אמת של השדות
- לחיצה על כפתור הרשמה
- מעבר למסך התחברות

**לוח מחוונים (Dashboard)**

**תפקיד המסך:** מרכז השליטה הראשי של המשתמש במערכת.

**מידע מוצר:**

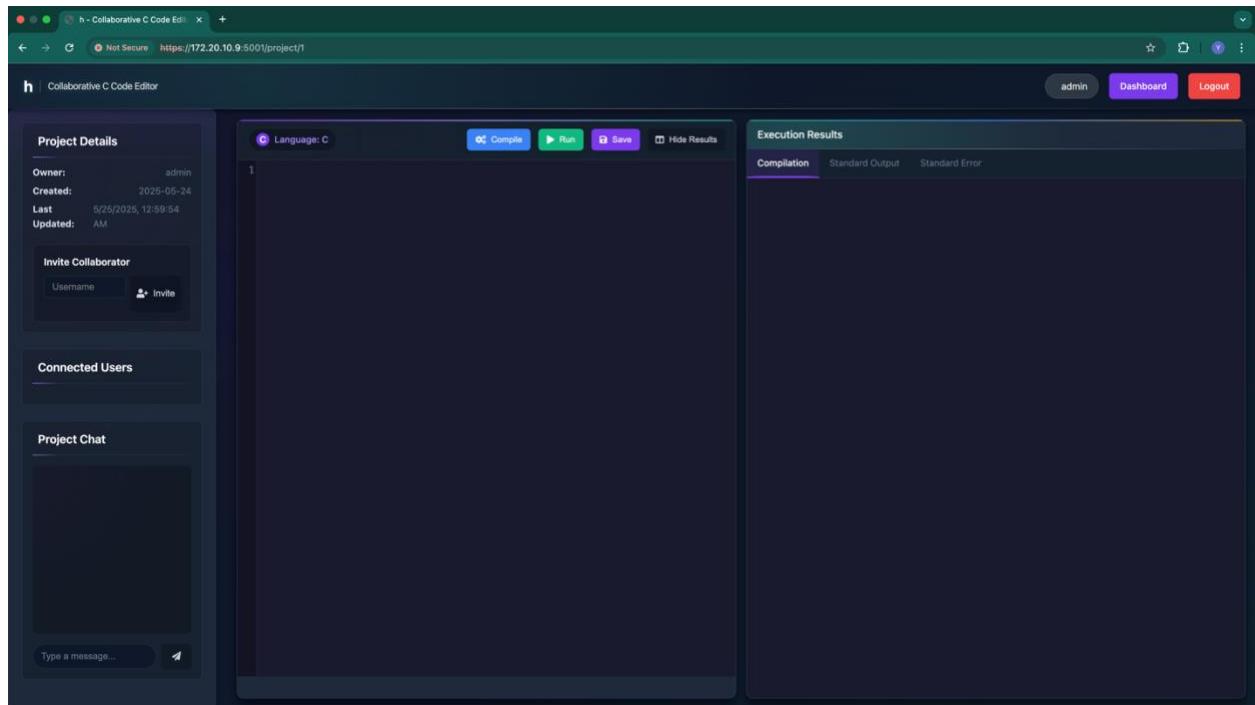
- רשימת פרויקטים בעלות המשתמש
- רשימת פרויקטים משותפים
- רשימת תרגילים בתהילה
- סטטיסטיות אישיות
- תפריט ניווט ראשי

**פעולות זמיינות:**

- ייצירת פרויקט חדש
- פתיחת פרויקט קיים לעריכה
- מעבר לדף תרגילים
- מעבר לפאנל ניהול

- התנטקות מהמערכת

## מסך ערך פרויקט



**תפקיד המסך:** הלב של המערכת - עריכה שיתופית של קוד C

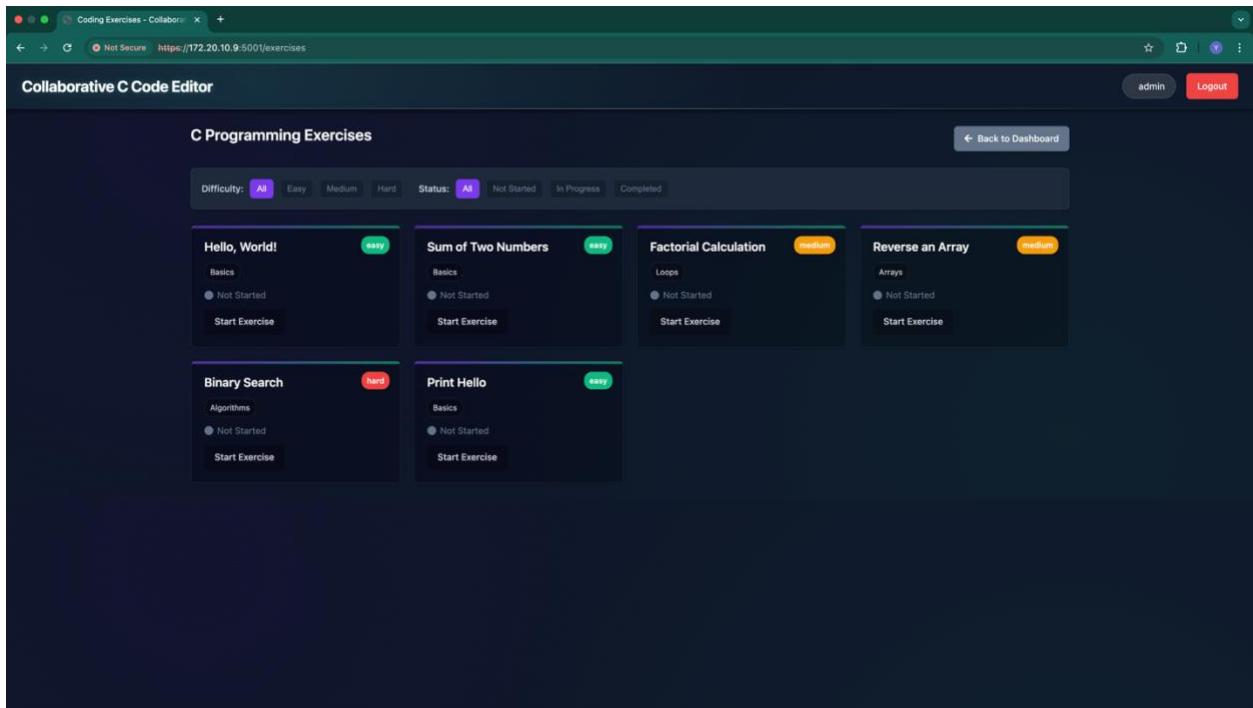
### מידע מוצג:

- עורך קוד עם syntax highlighting
- רשימת משתמשים מחוברים
- אוצר צ'אט לתקשורת
- פאנל תוצאות הרצה
- כפturny פעולה (קומpileציה, הרצה, שמירה)

### פעולות זמינות:

- ערך קוד בזמן אמת
- קומPILEציה של הקוד
- הרצת הקוד עם או בלי קלט
- שליחת הודעות צ'אט
- שמירת הפרויקט
- הזמנת משתתפים נוספים

## מסמך תרגילים



**תפקיד המשך:** הציג רשימות התרגילים הזמינים במערכת.

**מידע מוצג:**

- רשימת כל התרגילים
- מידע על כל תרגיל
- סטודנט תתקדמות
- אפשרויות סינון וחיפוש
- כפורי פועלה לכל תרגיל

**פעולות זמינות:**

- סינון תרגילים לפי קושי
- סינון תרגילים לפי סטטוס
- חיפוש תרגילים
- תחילת עבודה על תרגיל
- המשך עבודה על תרגיל קיים
- חוזה לדשبورד

## מסמך תרגיל

```

1 #include <stdio.h>
2
3 int main() {
4     int n;
5     unsigned long long factorial = 1;
6
7     // Read the input
8     scanf("%d", &n);
9
10    // Calculate factorial
11    // Write your code here
12
13    // Print the result
14    printf("%llu", factorial);
15
16    return 0;
17 }

```

**תפקיד המשך:** פתרו תרגיל ספציפי עם בדיקה אוטומטית.

## מיצע מוצג:

- תיאור התרגיל ודרישותיו
- דירוג קושי וקטגוריה
- עורך קוד עם קוד התחלתי
- אוצר תוכאות בדיקה
- פאנל התקדמות אישית
- פתרון

## פעולות זמינות:

- עריכת קוד הפתרון
- הרצה ובדיקה של הפתרון
- הצגת פתרון מלא
- חזרה לרשימת תרגילים
- מעבר לתרגיל הבא/קודם

**מסך ניהול**

The screenshot shows the Admin Dashboard interface for a Collaborative C Code Editor. The dashboard has a dark theme with purple highlights. On the left, a sidebar titled 'Administration' includes 'Dashboard' (selected), 'Users', 'Exercises', and 'Statistics'. The main area is titled 'System Overview' and contains three cards: 'Users' (3), 'Exercises' (6), and 'Compilations' (1). Below these are two tables: 'Recent Users' (listing admin, a, and smishky) and 'Recent Compilations' (listing one entry for user 'a').

ID	Username	Email	Created
3	admin	admin@gmail.com	2025-05-24 12:33
2	a	a@gmail.com	2025-05-24 12:29
1	smishky	ysmishkis@gmail.com	2025-05-24 11:04

ID	User	Status	Compiled
1	a	compilation_error	2025-05-24 12:31

**תפקיד המסך:** כלי ניהול למנהל המערכת.

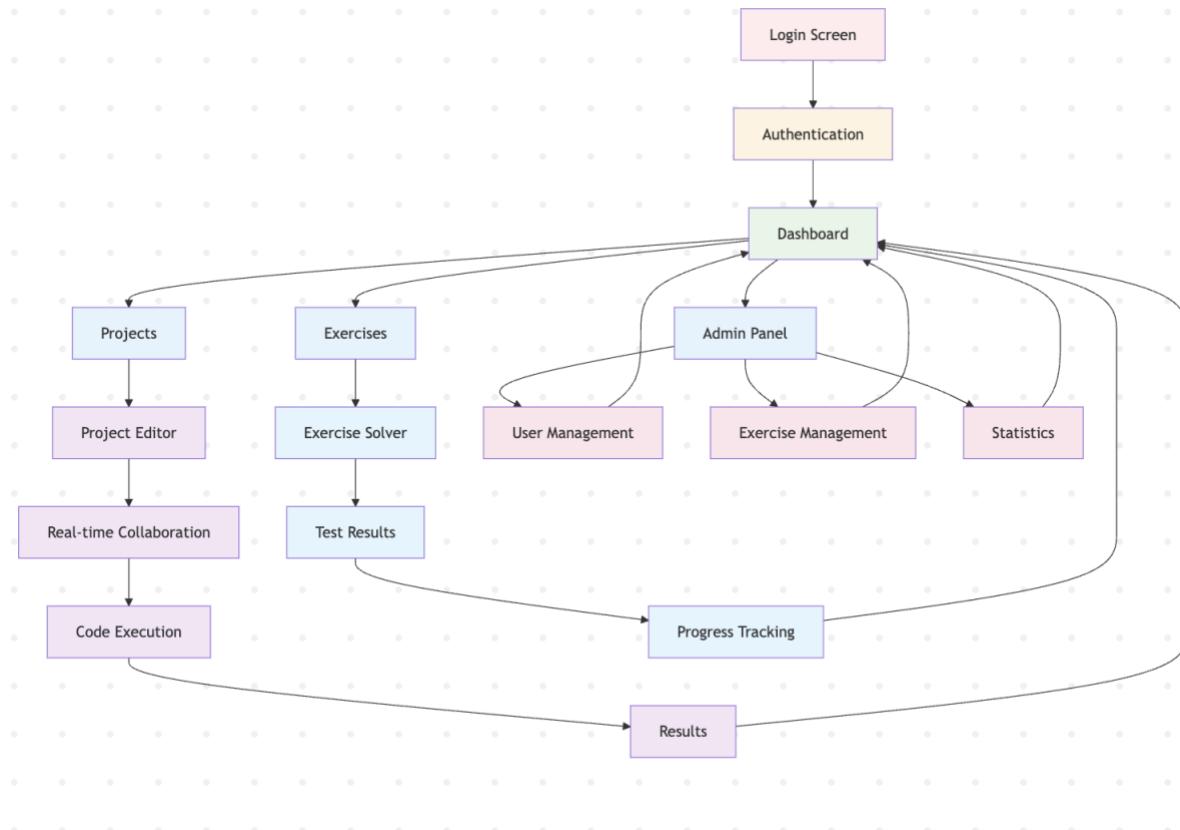
**מידע מוצר:**

- סטטיסטיות מערכת כלליות
- רשימת משתמשים
- רשימת תרגילים
- גרפים ודווחות
- לוגים ומעקב פעילות

**פעולות זמינות:**

- ניהול משתמשים
- ייצירה ועריכה של תרגילים
- צפיה בסטטיסטיות מפורטות
- ניהול תוכן ומורכזיה
- ייצוא נתונים ודווחות של תרגילים

## תרשים מסכימים



## מבנה הנתונים

## פירוט מבני הנתונים

המערכת משתמש במספר מבנים נתונים מרכזים, המוצגים כמודלים ב-SQLAlchemy:

*User*

מייצג משתמש במערכת.

שדה	סוג	תיאור
<i>id</i>	Integer	מזהה ייחודי אוטומטי
<i>username</i>	String(20)	שם משתמש, ייחודי
<i>email</i>	String(120)	כתובת דוא"ל, ייחודית
<i>password</i>	String(60)	סיסמה מוצפנת
<i>created_at</i>	DateTime	תאריך ייצירת המשתמש

קשרים:

- פרויקטים שהמשתמש יצר `owned_projects`
- פרויקטים שהמשתמש משתתף בהם `collaborated_projects`
- התקדמות המשמש בתרגילים `exercise_progress`

### *Project*

מייצג פרויקט במערכת.

שדה	סוג	תיאור
מזהה ייחודי אוטומטי	Integer	<i>id</i>
שם הפרויקט	String(100)	<i>name</i>
תוכן הקוד של הפרויקט	Text	<i>content</i>
תאריך יצירת הפרויקט	DateTime	<i>created_at</i>
תאריך עדכון אחרון	DateTime	<i>updated_at</i>
מזהה הבעלים של הפרויקט	Integer (FK)	<i>owner_id</i>

קשרים:

- המשתמש שיצר את הפרויקט `owner`
- משתמשים המשתתפים בפרויקט `Collaborators`
- מסמכים השייכים לפרויקט `Documents`

### *Document*

מייצג מסמך בפרויקט.

שדה	סוג	תיאור
מזהה ייחודי אוטומטי	Integer	<i>id</i>
שם המסמך	String(100)	<i>name</i>
תוכן המסמך	Text	<i>content</i>
מזהה הפרויקט	Integer (FK)	<i>project_id</i>
מזהה המשתמש שיצר את המסמך	Integer (FK)	<i>user_id</i>
תאריך יצירת המסמך	DateTime	<i>created_at</i>
תאריך עדכון אחרון	DateTime	<i>updated_at</i>

קשרים:

- הפרויקט שהמסמך שייך אליו `Project`

### *Exercise*

מייצג תרגיל תכנות.

שדה	סוג	תיאור
מזהה ייחודי אוטומטי	Integer	<i>id</i>

כותרת התרגיל	String(200)	<i>title</i>
תיאור התרגיל	Text	<i>description</i>
רמת קושי(easy, medium, hard)	String(20)	<i>difficulty</i>
קטגוריה	String(50)	<i>category</i>
קוד התחלתי	Text	<i>initial_code</i>
קוד פתרון	Text	<i>solution_code</i>
מקרי בדיקה בפורמט JSON	Text	<i>test_cases</i>

קשרים:

• התקדמות משתמשים בתרגיל Progress

*ExerciseProgress*

מייצג התקדמות של משתמש בתרגיל.

שדה	סוג	תיאור
<i>id</i>	Integer	מזהה ייחודי אוטומטי
<i>user_id</i>	Integer (FK)	מזהה המשתמש
<i>exercise_id</i>	Integer (FK)	מזהה התרגיל
<i>status</i>	String(20)	סטטוס(not_started, in_progress, completed)
<i>user_code</i>	Text	הקוד של המשתמש
<i>attempts</i>	Integer	מספר ניסיונות
<i>last_attempt</i>	DateTime	זמן הניסיון האחרון
<i>completed_at</i>	DateTime	זמן השלמת התרגיל

קשרים:

• משתמש User

• התרגיל Exercise

*CompilationHistory*

מייצג היסטוריית קומpileציה והרצה.

שדה	סוג	תיאור
<i>id</i>	Integer	מזהה ייחודי אוטומטי
<i>user_id</i>	Integer (FK)	מזהה המשתמש
<i>project_id</i>	Integer (FK)	מזהה הפרויקט
<i>document_id</i>	Integer (FK)	מזהה המסמך
<i>exercise_id</i>	Integer (FK)	מזהה התרגיל
<i>code</i>	Text	הקוד שהודר/הורץ
<i>compiled_at</i>	DateTime	זמן הקומPILEציה
<i>compilation_output</i>	Text	פלט הקומPILEציה
<i>execution_output</i>	Text	פלט ההרצה

(success, σιωπή)	String(20)	<i>status</i>
compilation_error,		
runtime_error)		<i>ChatMessage</i>

מייצג הودעת צ'אט.

שדה	סוג	תיאור
<i>id</i>	Integer	מזהה ייחודי אוטומטי
<i>user_id</i>	Integer (FK)	מזהה המשתמש
<i>project_id</i>	Integer (FK)	מזהה הפרויקט
<i>message</i>	Text	תוכן ההודעה
<i>sent_at</i>	DateTime	זמן שליחת ההודעה

## קשיים:

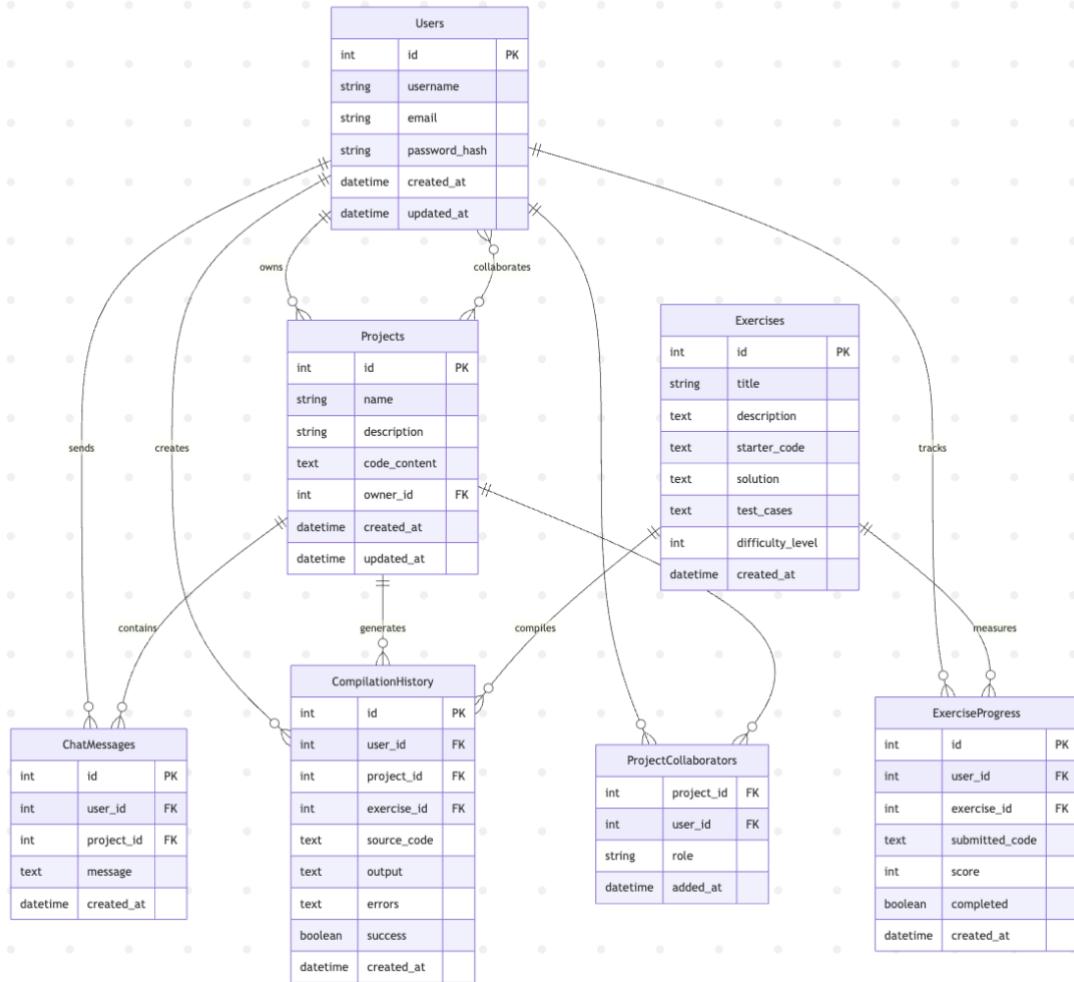
- User המשמש ששלח את הודעה  
Project הפרויקט שהודעת שicity ליאו

פירוט מאגרי המידע

המערכת משתמשת בסיס נתונים SQL, מיושם באמצעות מנגנון ORM של SQLAlchemy כולל את הבעיות הבאות:

- |    |  |
|----|--|
| 1. | <b>user</b> שומרת מידע על משתמשי המערכת<br><b>project</b> שומרת מידע על פרויקטים<br><b>project_collaborators</b> טבלת קשר בין משתמשים לפרויקטים (many-to-many)<br><b>document</b> שומרת מידע על מסמכים בפרויקטים<br><b>exercise</b> שומרת מידע על תרגילי תוכנות<br><b>exercise_progress</b> שומרת מידע על התקדמות משתמשים בתרגילים<br><b>compilation_history</b> שומרת מידע על ניסיונות קומpileציה והרצתה<br><b>chat message</b> שומרת מידע על הודעות צ'אט |
|----|--|

## בסיס נתונים



## אינדקסים לביצועים:

- users.username : Username •
- users.email : Email •
- projects.owner\_id : Owner •
- exercise\_progress(user\_id, exercise\_id) : User-Exercise •
- compilation\_history.compiled\_at : Compilation-Time •

## סקירת חולשות והאיומים

המערכת חשופה למספר איומים אבטחה פוטנציאליים, והוא מימוש מנגוני הגנה כדי להתמודד איתם :

### שכבות האפליקציה

מנגנון הגנה	תיאור	איום
SQLAlchemy ORM אוטומטית	זרקת קוד SQL זמני לשאליות שמנקה קלט	<i>SQL Injection</i>
NIKKI קלט, הגבלת תגיוט HTML מותרונות	זרקת קוד JavaScript זמני ניקוי קלט, הגבלת תגיוט HTML	<i>Cross-Site Scripting (XSS)</i>
CSRF token	ביצוע פעולות בשם משתמש מאומת	<i>Cross-Site Request Forgery (CSRF)</i>
nisyonot לחש סיסמות עם bcrypt מוגבלות על ניסיונות כניסה	הצפנה סיסמות או לעקוφ את מערכת האimotoות	התקפות על מערכת האimotoות
סינון קוד מסוכן, סביבת הריצה מבודדת, הגבלת משאבים	הרצת קוד C שמנסה לפגוע במערכת	הרצת קוד זמני
הגבלת קצב בקשנות, הגבלת משאבים לכל משתמש	התקפות מניעת שירות	<i>DOS/DDOS</i>

### שכבות התעבורה

מנגנון הגנה	תיאור	איום
שימוש ב- TLS/SSL להצפנה תקשורת לשרת	ירוט תקשורת בין לקוח לשרת גניבת זהות חדש	<i>Man-in-the-Middle Session Hijacking</i>
שימוש בטוקנים מוצפנים, תופוגת שני	הצפנה כל התקשורת באמצעות SSL/SSL רשות האזנה לתקשורת רשת	<i>Packet Sniffing</i>

### שכבות המערכת

מנגנון הגנה	תיאור	איום
סביבת הריצה מבודדת, הגבלת משאבים, סינון קוד מסוכן	הרצת קוד שמנסה לגשת למערכת הקבצים או למשאים אחרים ניסיון להשיג הרשות גבוהה יותר	<i>הרצת קוד זמני Privilege Escalation</i>
הרצת קוד בהרשאות מינימליות, סביבת הריצה מבודדת	ניסיון לצרוך יותר מדי משאבי מערכת לכל הרצת קוד	שימוש יתר במשאבים

המערכת מימוש שכבות הגנה מרובות כדי להתמודד עם איומים אלה. היא משתמשת בכלים ושיטות סטנדרטיים ומוכחים, כמו SQLAlchemy, Flask-Bcrypt ו-TLS/SSL ומימוש מנגוני אבטחה ייחודיים כמו סינון קוד מסוכן וסביבה הריצה מבודדת.

## מימוש הפרויקט

### חלק א' - סקירת המודולים והמחלקות של המערכת

#### מודולים מיובאים

שם המודול	תפקידו
<i>Flask</i>	פריאמוורק לפיתוח אפליקציית ווב בשפת Python
<i>Flask-SocketIO</i>	הרחבה של Flask המאפשרת תקשורת בזמן אמת באמצעות WebSockets
<i>SQLAlchemy</i>	ORM (Object Relational Mapper) להטמעשות עם מסד נתונים
<i>Bcrypt</i>	ספירה להצפנה סיסמות
<i>JSON</i>	ספירה לטיפול בפורמט JSON
<i>Subprocess</i>	ספירה להרצת פקודות מערכת הפעלה
<i>Resource</i>	ספירה לניהול משאבי מערכת הפעלה

#### מחלקות שפותחו

*User*

#### תפקיד המחלקה : ניהול פרטי משתמש במערכת

#### תכונות המחלקה :

- id : מזהה ייחודי למשתמש
- Username : שם המשתמש
- Email : כתובת הדוא"ל של המשתמש
- password : סיסמה מוצפנת
- created\_at : תאריך יצירת החשבון
- owned\_projects : קשר לפרויקטים שהמשתמש יצר
- collaborated\_projects : קשר לפרויקטים שהמשתמש משתתף בהם
- exercise\_progress : קשר לתקדמות המשתמש בתרגילים

#### פעולות במחלקה :

- \_\_repr\_\_ : מחזיר מחרוזת המייצגת את המשתמש

*Project*

#### תפקיד המחלקה : שמירה וניהול של פרויקטי קוד

#### תכונות המחלקה :

- Id : מזהה ייחודי לפרויקט
- Name : שם הפרויקט
- Content : תוכן הפרויקט
- created\_at : תאריך יצירת הפרויקט

- updated\_at : תאריך עדכון אחרון של הפרויקט
- owner\_id : מזהה המשתמש שיצר את הפרויקט

**פעולות במחלקה:**

- \_\_repr\_\_ : מוחזר מחוوظת המייצגת את הפרויקט

### *Exercise*

**תפקיד המחלקה:** ניהול תרגילי קוד במערכת

**תכונות המחלקה:**

- Id : מזהה ייחודי לתרגיל
- Title : כותרת התרגילים
- Description : תיאור התרגילים
- difficulty : רמת קושי
- category : קטגוריה
- initial\_code : קוד התחלתי לתרגיל
- solution\_code : פתרון לתרגיל
- test\_cases : מקרים בדיקה
- progress : קשר להתקדמות משתמשים בתרגיל

**פעולות במחלקה:**

- \_\_repr\_\_ : מוחזר מחוوظת המייצגת את התרגילים

### *ExerciseProgress*

**תפקיד המחלקה:** מעקב אחר התקדמות משתמשים בתרגילים

**תכונות המחלקה:**

- Id : מזהה ייחודי
- user\_id: מזהה המשתמש
- exercise\_id : מזהה התרגילים
- status : סטטוס
- user\_code : הקוד שהמשתמש כתב
- attempts : מספר הניסיונות
- last\_attempt : תאריך הניסיון האחרון
- completed\_at : תאריך השלמת התרגילים

**פעולות במחלקה:**

- \_\_repr\_\_ : מוחזר מחוوظת המייצגת את ההתקדמות

*CompilationHistory***תפקיד המחלקה:** שמירת ההיסטוריה קומpileציה והרצה קוד**תכונות המחלקה:**

- Id : מזהה ייחודי
- user\_id : מזהה המשתמש
- project\_id : מזהה הפרויקט
- document\_id : מזהה המסמך
- exercise\_id : מזהה התרגיל
- code : הקוד שהורץ
- compiled\_at : תאריך הקומPILEציה
- compilation\_output : פלט הקומPILEציה
- execution\_output : פלט ההרצה
- status : סטטוס

**פעולות במחלקה:**

- \_\_repr\_\_ : מחזיר מחרוזת המייצגת את ההיסטוריה

*ChatMessage***תפקיד המחלקה:** ניהול הודעות ציאט בין משתמשים בפרויקט**תכונות המחלקה:**

- Id : מזהה ייחודי להודעה
- user\_id : מזהה המשתמש שלח את ההודעה
- project\_id : מזהה הפרויקט
- message : תוכן ההודעה
- sent\_at : זמן שליחת ההודעה
- User : קשר למשתמש שלח את ההודעה
- Project : קשר לפרויקט

**פעולות במחלקה:**

- \_\_repr\_\_ : מחזיר מחרוזת המייצגת את ההודעה

*AdminBlueprint***תפקיד המחלקה:** ניהול דפי האדמינ'**תכונות המחלקה:**

- `admin_required` : דקורתור לבדיקת הרשותות אדמין

**פעולות במחלקה:**

- `check_admin()` : בדיקה האם המשתמש הוא אדמין
- `index()` : דף הבית של האדמין
- `users()` : ניהול משתמשים
- `exercises()` : ניהול תרגילים
- `stats()` : הצגת סטטיסטיות

**פונקציות עזר מרכזיות:**

#### `execute_code_impl()`

- **תפקיד:** הרצה מאובטחת של קוד C
- **פרמטרים:** `code, project_id, user_input, compile_only`
- **החזרה:** תוכאות קומpileציה והרצה
- **תכונות מיוחדות:** sandboxing, resource limiting, security checks

#### `check_for_dangerous_code()`

- **תפקיד:** בדיקת קוד לתבניות מסוכנות
- **פרמטרים:** `code (string)`
- **החזרה:** True אם נמצא קוד מסוכן
- **תכונות מיוחדות:** regex patterns לזיהוי פונקציות מערכת

#### `set_resource_limits()`

- **תפקיד:** הגדרת הגבלות משאבים לתהיליך
- **פרמטרים:** לא
- **החזרה:** לא
- **תכונות מיוחדות:** הגבלת זיכרון CPU, וקבצים

## חלק ב' - אלגוריתמים מרכזיים בפרויקט

### אלגוריתם הרצת קוד C מאובטח

**הסבר על היכולת:** אלגוריתם זה מאפשר למשתמש להריץ קוד C בסביבה מאובטחת תוך הגנה על המערכת מפני קוד זדוני. האלגוריתם כולל שלושה שלבים עיקריים:

1. בדיקת אבטחה של הקוד
2. קומPILEציה של הקוד

### 3. הרצה הקוד בסביבה מבודדת עם הגבלת משאבים

#### תיאור מפורט של האלגוריתם :

1. המערכת יוצרת תיקייה זמנית לאחסון קבצי הקוד והקובץ הבינארי המקומפל
2. הקוד נכתב לקובץ זמני
3. מתבצעת בדיקת אבטחה לאייתור תבניות קוד מסווגות למשל ()(), popen() ו cocci system()
4. אם מתגלות תבניות מסווגות, התהיליך נעצר ומוחזרת הודעה שגיאה
5. הקוד מקומפל באמצעות GCC עם דגלים ספציפיים להגדרת סטנדרט, ספריות וכו'
6. אם יש שגיאות קומפילציה, הן מוחזרות למשתמש
7. לפני הרצה הקוד מוגדרות מגבלות משאבים באמצעות resource.setrlimit:
  - o הגבלת זיכרון ל- MB32
  - o הגבלת זמן CPU ל- 2- 2 שניות
  - o הגבלת גודל קבצים ל- MB1
8. הקוד מורץ בתהיליך נפרד עם קלט מהמשתמש (אם יש)
9. פلت ההרצה נאוסף ומוחזר למשתמש
10. התקיינה הזמנית נמחקת

קוד זה מיישם עקרונות של "סביבת חול (Sandbox)" להרצה קוד לא אמין תוך הגנה על המערכת המארחת.

## אלגוריתם Operational Transformation לעריכה שיתופית

**הסבר על היכולת:** אלגוריתם זה מנהל עיריכת קוד שיתופית בזמן אמת בין מספר משתמשים. הוא מאפשר סyncron תוכן העורך בין משתמשים שונים ופתרון התנגשויות.

#### תיאור מפורט של האלגוריתם :

1. כל שינוי בעורך הטקסט מייצר פעולה עיריצה (operation) מסווג :
  - o הוספה (insert): הוספת טקסט במקומות מסוימים
  - o מחיקה (delete): מחיקת טקסט ממקומות מסוימים
  - o החלפה (replace): החלפת כל התוכן
2. פעולה העיריצה נשלחת לשרת עם המידע הבא :
  - o סוג הפעולה
  - o מיקום בטקסט
  - o הטקסט הרלוונטי
  - o מזזה הפרוייקט
3. השרת מקבל את הפעולה ומבצע את השינוי במסמך המשותף
4. השרת שדר את המסמך המעודכן לכל המשתמשים החוברים לאותו פרויקט
5. לקוחות מקבלים את המסמך המעודכן וمعدכנים את העורך שלהם
6. כדי למנוע "עיריכות מרווחות" (race conditions), השרת מבצע את הערכות בסדר קבלתן

אלגוריתם זה מבטיח שככל המשתמשים רואים אותו תוכן בכל רגע נתון, גם כאשר מספר משתמשים עורכים את המסמך במקביל.

## אלגוריתם בדיקת תרגילי קוד אוטומטית

**הסבר על היכולות:** אלגוריתם זה מאפשר בדיקה אוטומטית של פתרונות משתמשים לתרגום קוד באמצעות מקרים בדיקה מוגדרים מראש.

### תיאור מפורט של האלגוריתם:

1. המערכת מקבלת את הקוד שכתב המשתמש ואת רשימת מקרים הבדיקה
2. הקוד נבדק תחילה מבחינת אבטחה להגנה מפני קוד זמני
3. הקוד מוקומפל עם אותן הגדירות כמו באלגוריתם הרצת הקוד
4. אם הקומpileציה נכשלה, מוחזרת שגיאת הקומpileציה
5. עברו כל מקרה בדיקה:
  - קוד המשמש מורץ עם הקלט המוגדר במקרה הבדיקה
  - הפלט מהריצהמושווה לפלט המוצופה
  - נשמרת תוצאה הבדיקה ( עבר/נכשל/תקוע )
6. המערכת מחזירה סיכום של כל מקרים הבדיקה, כולל מידע על מה עבר ומה נכשל
7. אם כל מקרים הבדיקה עברו בהצלחה, התרגום מסומן כ"הושלם"

האלגוריתם מאפשר משוב מיידי למשתמשים על פתרונותיהם ומאפשר למידה אינטראקטיבית.

## חלק ג' - מסמך בדיקות בדיקות שתוכננו בשלב האפיון בדיקת ערך קוד ותוכנת תחביר

**מטרת הבדיקה:** לוודא שהעורך מאפשר הזנת קוד C ומציג את התchapir בצורה נכונה.

**מה בוצע בפועל:** הזנת קטעי קוד שונים ובדיקה תצוגת הצבעים והפורמט.

**תוצאות הבדיקה:** העורך מציג כראוי את התchapir באמצעות ספריית CodeMirror. מילים מפתח, משתנים, מחרוזות ופונקציות מוצגים בצבעים שונים לשיפור הקריאה.

**בעיות שהתגלו ופתרונו:** לא התגלו בעיות משמעותיות.

### בדיקות קומPILEציה והרצת קוד

**מטרת הבדיקה:** לוודא שהמערכת מסוגלת להדר ולהריץ קוד C תקין ולהציג הודעות שגיאה מתאימות עבור קוד לא תקין.

**מה בוצע בפועל:**

- נכתבו והורצטו תוכניות C פשוטות (כגוןHello World)
- נכתבו והורצטו תוכניות עם לולאות ותנאים
- נכתבו והורצטו תוכניות עם שגיאות תchapir
- נכתבו והורצטו תוכניות עם שגיאות זמן ריצה

**תוצאות הבדיקה:**

- קוד תקין הודר והורץ בהצלחה
- המערכת הציגה הודעות שגיאה ברורות לקוד לא תקין
- המערכת זיהתה ותיעדה שגיאות זמן ריצה

**בעיות שהתגלו ופתרונו:**

- בעיות ראשוניות בפרמטרים של - GCC נפתרו על ידי הוספת דגלים מתאימים לGCC
- בעיות בזיהוי ספירות - נפתרו על ידי הוספת נתיבי include מפורשים
- תקיעות בתוכניות אינסופיות - נפתרו על ידי הוספת הגבלת זמן ריצה

**בדיקות ערךיה שיתופית**

**מטרת הבדיקה:** לוודא שמספר משתמשים יכולים לעורך את אותו הקוד במקביל ולראות את השינויים בזמן אמת.

**מה בוצע בפועל:** חיבור של מספר משתמשים לאותו פרויקט ובידיקת סyncronization השינויים.

**תוצאות הבדיקה:** המשמשים יכולים לראות את השינויים של אחד השני בזמן אמת. מיקום הסמן של כל משתמש הועזב לשימוש אחרים.

**בעיות שהתגלו ופתרונו:**

- ניתוקים מזדמנים בחיבור - WebSocket נפתרו על ידי הוספת מגנון התחברות חדש אוטומטי
- עיכובים בסyncronization בעט עומס - נפתרו על ידי ניהול העברת המתוונים

**בדיקה פתרון התנשויות**

**מטרת הבדיקה:** לוודא שהמערכת מסוגלת לטפל נכון בעריכה מקבילית של אותו קטע קוד.

**מה בוצע בפועל:** יצירת התנשויות מכוונות על ידי עריכה מקבילית של אותו קטע קוד על ידי שני משתמשים שונים.

**תוצאות הבדיקה:** המערכת הצליחה לפתרן התנשויות על ידי החלטת השינויים לפי סדר הגעתם לשורת.

**בעיות שהתגלו ופתרונו:**

- בעיות בסדר החלטת שינויים - נפתרו על ידי שיפור אלגוריתם ה- TO
- איבוד שינויים בתנאי תזמון מסוימים - נפתר על ידי שיפור מגנון הסyncronization

**בדיקה אזכור תרגול תכנות C**

**מטרת הבדיקה:** לוודא שהמשתמשים יכולים לגשת לתרגילים, להגיש פתרונות ולקבל משוב.

**מה בוצע בפועל:** פתרון מספר תרגילים ובידיקת המשוב והדירוג.

**תוצאות הבדיקה:** המשמשים הצלחו לגשת לתרגילים, להגיש פתרונות ולקבל משוב אוטומטי על פתרונוניהם.

#### בעיות שהתגלו ופתרונו :

- בעיות במקרי בדיקה מורכבים - נפתרו על ידי שיפור מנגנון השוואת הפלט
- בעיות בתצוגת התקדמות המשתמש - נפתרו על ידי עדכון ממשך המשתמש

#### בדיקות אבטחה

##### בדיקות העפנת תקשורת

**מטרת הבדיקה:** לוודא שהתקשרות בין הלקוח לשרת מוצפנת באמצעות TLS/SSL .

**מה בוצע בפועל:** מעקב אחר תעבורת הרשות וידוא שהוא מוצפנת.

**תוצאות הבדיקה:** כל התקשרות בין הלקוח לשרת מוצפנת כנדרש באמצעות TLS/SSL .

#### בעיות שהתגלו ופתרונו :

- בעיות בתצורת SSL - נפתרו על ידי עדכון תצורת השרת והתעודות

#### בדיקות אינומת משתמשים

**מטרת הבדיקה:** לוודא שתהליכי הרשמה, כניסה ואייפוס סיסמה פועלים כראוי.

#### מה בוצע בפועל :

- ניסיון כניסה עם פרטיים נכונים ושגוויים
- בדיקת תהליך הרשמה
- בדיקת חזוק הסיסמאות

**תוצאות הבדיקה:** מערכת האינומת פועלת כראוי, סיסמאות מאוחסנות בצורה מוצפנת ומأובטחת.

#### בעיות שהתגלו ופתרונו :

- חולשה בתהליכי אייפוס סיסמה - נפתרה על ידי הוספת שכבת אינומת נוספת
- אפשרות ליצור סיסמאות חלשות - נפתרה על ידי הוספת בדיקת חזק סיסמה

#### בדיקות הרשות גישה

**מטרת הבדיקה:** לוודא שימושים יכולים לגשת רק לפרויקטים שיש להם הרשאה אליהם.

**מה בוצע בפועל:** ניסיון גישה למשאים ללא הרשאה מתאימה.

**תוצאות הבדיקה:** המערכת מונעת גישה לפרויקטים שלמשמש אין הרשאה אליהם.

**בעיות שהתגלו ופתרוּן:**

- מספר נקודות גישה ללא אכיפת הרשות - נפתרו על ידי הוספת בדיקות הרשות בכל הנקודות הクリיטיות

**בדיקות ביצועים****בדיקות עומסים**

**מטרת הבדיקה:** לוודא שהמערכת מתפקדת תחת עומס של משתמשים רבים.

**מה בוצע בפועל:** סימולציה של פעילות מרובה משתמשים ומידת זמני תגובה.

**תוצאות הבדיקה:** המערכת תפקדה באופן סביר עד 100 משתמשים במקביל.

**בעיות שהתגלו ופתרוּן:**

- עומס על בסיס הנתונים - נפתר על ידי הוספת cache
- עומס בעת קומפילציה מרובה - נפתר על ידי הגבלת כמות הקומpileציות במקביל

**2. בדיקת זמני תגובה**

**מטרת הבדיקה:** לוודא שערכונים בעירכה השיתופית מופצים במהירות.

**מה בוצע בפועל:** מדידת הזמן בין ביצוע שינוי להופעתו אצל משתמשים אחרים.

**תוצאות הבדיקה:** זמן התגובה הממוצע היה פחות מ-200 מילישניות בתנאים רגילים.

**בעיות שהתגלו ופתרוּן:**

- עיכובים ברשומות איטיות - נפתרו באמצעות אופטימיזציה של גודל ההודעות המועברות

**בדיקות נוספות שבוצעו****בדיקות מגוון דפדףנים**

**מטרת הבדיקה:** לוודא שהמערכת פועלת באופן תקין ב.TODO דפדףנים.

**מה בוצע בפועל:** בדיקת המערכת בדףדףנים Edge, Chrome, Firefox, Safari.

**תוצאות הבדיקה:** המערכת תפקדה כראוי בכל הדפדףנים, עם הבדלים מינוריים בתצוגה.

**בעיות שהתגלו ופתרוּן:**

- בעיות תאימות ב-Safari - נפתרו על ידי התאמת ה-CSS-והתסריטים

**בדיקות תגובתיות במכשורים ניידים**

**מטרת הבדיקה:** לוודא שההממשק מותאים למכשורים ניידים ומסכים קטנים.

**מה בוצע בפועל:** בדיקת המערכת במגוון מכשורים ורזרולוציות מסך.

**תוצאות הבדיקה:** הממשק הותאם למכשורים ניידים באמצעות CSS רסתפונסיבי.

**בעיות שהתגלו ופתרונו:**

- בעיות בתצוגת עורך הקוד במסכים קטנים - נפתרו על ידי התאמת העיצוב והוספה אפשרות אפליאות לגיליה

**בדיקות שימוש בזכרון וCPU**

**מטרת הבדיקה:** לוודא שהמערכת לא צורכת יותר מדי משאבים.

**מה בוצע בפועל:** מדידת צריכה זיכרון וCPU במצבים שימוש שונים.

**תוצאות הבדיקה:** צריכת המשאבים הייתה בגבולות הסבירים, עם עליות צפויות בעת קומפילציה.

**בעיות שהתגלו ופתרונו:**

- דליפות זיכרון בתהליכי קומפילציה - נפתרו על ידי שיפור ניהול המשאבים

## מדריך למשתמש

### פירוט קבצי המערכת - עץ קבצים

```

VCCE/
|
├── server.py      # קובץ השרת הראשי
├── models.py       # הדרות מודלים של מסד הנתונים
├── admin.py        # Blueprint לאנגל האדמינ
├── exercise_manager.py   # ניהול תרגילים ופונקציות עזר
├── requirements.txt # Requirements dependencies
├── docker-compose.yml # הדרות Docker
├── Dockerfile       # הדרות לבניית container
├── .gitignore       # קבצים להתעלמות ב-Git
├── README.md        # תיעוד בסיסי
├── cert.pem         # תעודה SSL
└── key.pem          # פרטי SSL מפתח

|
├── static/          # קבצים סטטיים
│   ├── style.css    # עיצוב ראשי של הממשק
│   └── script.js    # JavaScript קיד לוח
|
├── templates/       # תבניות HTML
│   ├── index.html   # דף הבית עורך קוד
│   ├── login.html   # דףהתחברות
│   ├── register.html # דף הרשמה
│   ├── dashboard.html # לוח מחוונים
│   ├── editor.html   # עורך פרויקט
│   ├── exercises.html # רשימת תרגילים
│   ├── exercise.html # דף תרגיל בודד
│   ├── new_project.html # יצירת פרויקט חדש
|
│   └── admin/        # תבניות אングל אדמין
│       ├── index.html # דASHBOARD אדמין
│       ├── users.html  # ניהול משתמשים
│       ├── exercises.html # ניהול תרגילים
│       ├── new_exercise.html # יצירת תרגיל חדש
│       ├── edit_exercise.html # עריכת תרגיל
│       └── stats.html   # סטטיסティקות
|
├── certs/           # תיקיית תעוזות SSL
│   ├── cert.pem
│   └── key.pem
|
└── migrations/     # קבצי מיגרציה של מסד הנתונים
    └── versions/

```

**התקנת המערכת****סביבת נדרשת**

- Python 3.9 או גרסה חדשה יותר
- מותקן על השרת (GNU Compiler Collection) GCC
- PostgreSQLSCLite
- Docker and Docker compose

**כלים נדרשים****התקנת תלויות בסיסיות (Ubuntu) :**

```
sudo apt update
sudo apt install python3 python3-pip python3-venv
sudo apt install postgresql postgresql-contrib
sudo apt install gcc build-essential
sudo apt install git
sudo apt install docker.io docker-compose
```

**התקנת תלויות Python :**

```
# ייצירת virtual environment
python3 -m venv venv
source venv/bin/activate
```

```
# התקנת packages
pip install -r requirements.txt
```

**הוראות התקנה****שלב 1 : שכפול הפרויקט**

```
git clone <repository-url>
cd VCCE
```

**שלב 2 : הגדרת מסד נתונים**

```
# PostgreSQL-ל
התחברות ל PostgreSQL
sudo -u postgres psql

# Configuration
# Configuration file for PostgreSQL
CREATE DATABASE code_editor;
CREATE USER code_editor_user WITH PASSWORD 'your_password';
GRANT ALL PRIVILEGES ON DATABASE code_editor TO code_editor_user;
```

\q

**שלב 3 : הגדרת משתני סביבה**יצירת קובץ `env` :

```
DATABASE_URL=postgresql://code_editor_user:your_password@localhost:5432/code_editor
SECRET_KEY=your_very_secret_key_here
ALLOWED_ORIGINS=*
DEBUG=True
```

**שלב 4 : אתחול מס'ד הנתונים**

python models.py

**שלב 5 : הרצת השירות**

python server.py

התקנה עם Docker (מומלץ!!!!!!):

docker-compose up -d

נתונים התחלה

תרגילים לדוגמה : המערכת יוצרת אוטומטית 5 תרגילי דוגמה :

Hello, World!	.1
Sum of Two Numbers	.2
Factorial Calculation	.3
Reverse an Array	.4
Binary Search	.5

הגדרות בירית מחדל:

- פורט שירות : 5001
- מאפשר חיבורים HTTPS
- לוגים ברמת DEBUG
- גיבוי אוטומטי יומי

משתמשי המערכת

משתמש רגיל

יצירת חשבון והתחברות:

1. גישה למערכת :
  - فتح صفحة وغش لكتوبت : <https://localhost:5001>

**2. הרשמה למערכת :**

- לחץ על "Register here" בדף הכנסה
- מלא את הפרטים הנדרשים :

- שם משתמש (באנגלית, ללא רווחים)
- כתובת אימייל תקפה
- סיסמה (מינימום 6 תווים)

**3. התחברות :**

- הזן את שם המשתמש והסיסמה
- לחץ על "Login"
- תועבר ללוח המחוונים האישי

**עבודה עם פרויקטים:**

**1. יצרת פרויקט חדש :**

- בלוח המחוונים, לחץ על "New Project"
- הזן שם לפרוייקט
- לחץ על "Create Project"
- תועבר לעורך הקוד

**2. עריכת קוד :**

- כתוב את הקוד בעורך
- השתמש ב- Ctrl+Enter להרצת מהירה
- השתמש ב- Ctrl+S לשמירה
- עקוב אחר משתמשים מחוברים ברשימת הימנית

**3. קומpileציה והרצתה :**

- לחץ על "Compile" לקומpileציה בלבד
- לחץ על "Run" להרצת מלאה
- אם התוכנית מתחילה בקלט, הזן אותו בשדה הקלט
- צפה בתוצאות בלשוניות המתאימות

**4. שיתוף פרויקט :**

- בעורך הפרויקט, הזן שם משתמש בשדה "Invite Collaborator"
- לחץ על "Invite"
- המשתמש המוזמן יראה את הפרויקט ברשימת הפרויקטים המשותפים

**עבודה עם תרגילים:**

**1. גישה לתרגילים :**

- מלוח המחוונים, לחץ על "View All Exercises"
- או לחץ על תרגיל ספציפי בראשימת התרגילים בתחילת

**2. סינון וחיפוש :**

- השתמש בכפטוריו הסינון לפי קושי (Easy/Medium/Hard)
- השתמש בכפטוריו הסינון לפי סטטוס (Not Started/In Progress/Completed)
- כל הסינונים ניתנים לשילוב

**3. פתרון תרגיל :**

- לחץ על "Start Exercise" או "Continue" בתרגיל הרצוי

- קרא את התיאור והדרישות מצד שמאל

- כתוב את הפתרון בעורך הקוד

- לחץ על "Run & Check" לבדיקה אוטומטית

**4. הצגת תוצאות :**

- לאחר הריצה, תוצג טבלת תוצאות מפורטת

- כל מקרה בדיקה מציג: קלט, פלט צפוי, פלט בפועל, סטטוס

- תרגיל מושלם כאשר כל מקרי הבדיקה עברו

**5. הצגת פתרון :**

- לאחר השלמת התרגילים (או בכל עת), ניתן לחוץ על "Reveal Solution"

- יוצג הפתרון המלא עם הסברים

**תקשורות וצ'אט :**

**1. צ'אט בפרויקט :**

- בעורך הפרויקט, השתמש באזורי הצ'אט מצד ימין

- כתוב הודעה ולחץ Enter או על כפתור השליחה

- כל המשתמשים בפרויקט יראו את ההודעה

**2. מעקב אחר משתמשים :**

- בראשימת המשתמשים המוחברים, תוכל לראות מי מחוברCurrently

- צבעי הusers מציגים מיקום כל משתמש בקוד

**ניהול חשבון:**

**1. שינוי פרטיים :**

- כרגע לא קיימת אפשרות לשינוי פרטיים דרך הממשק

- ניתן לפנות למנהל המערכת

**2. התנתקות :**

- לחץ על "Logout" בפינה הימנית העליונה

- תועבר לדף הכניסה

**משתמש מנהל**

**גישה לפannel ניהול:**

**1. התחברות כמנהל :**

- השתמש בחשבון עם הרשאות מנהל

- מלוח המחוונים, תראה כפתור "Admin Panel"

- לחץ עליו לכינסה לפannel ניהול

**ניהול משתמשים:**

**1. צפייה במשתמשים :**

- בפannel האדמינין, לחץ על "Users"

- תוצג רשימה כל המשתמשים במערכת

- ניתן לחפש משתמש ספציפי

**2. ניהול משתמש :**

- לחץ על כפתור העין לצפייה בפרטי המשתמש

**ניהול תרגילים:****1. צפיה בתרגילים :**

- לחץ על "Exercises" בתפריט האדמין
- תוצג רשימת כל התרגילים
- ניתן לחפש ולסנן תרגילים

**2. ייצירת תרגיל חדש :**

- לחץ על "New Exercise"
- מלא את הפרטים הנדרשים :

- כוורתת התרגיל  
(Easy/Medium/Hard)
- רמת קושי (Easy/Medium/Hard)
- קטגוריה
- תיאור מפורט
- קוד התחלתי
- קוד הפתרון המלא
- מקורי בדיקה בפורמט JSON

**3. עריכת תרגיל :**

- לחץ על כפתור העריכה ליד התרגיל
- עורך את הפרטים הנדרשים
- שומר את השינויים

**4. מחיקת תרגיל :**

- לחץ על כפתור המחיקה
- אשר את המחיקה בחלון המאשר

**פורמט מקרי בדיקה:**

```
[
  {
    "input": "5 7",
    "expected_output": "12"
  },
  {
    "input": "10 -3",
    "expected_output": "7"
  }
]
```

**צפיה בסטטיסטיות:****1. דשborad אדמין :**

- דף הבית של האדמין מציג סטטיסטיות כלליות :
  - מספר משתמשים במערכת
  - מספר תרגילים
  - מספר קומפיילציות שבוצעו

**2. סטטיסטיות מפורטות :**

- לחץ על "Statistics" לנתחונים מפורטים יותר

- גרפים המציגים חלוקת תרגילים לפי קושי
- נתוני הצלחת קומפילציות

**ניטור פעילות:**

**1. משתמשים מחוברים :**

- צפייה במשתמשים הפעילים כרגע
- מעקב אחר פעילותם האחורונה

## סיכום אישי / רפלקציה

### תהליך העבודה על הפרויקט

עבודה על פרויקט זה הייתה מסע מרתק של למידה והתפתחות. בתחילת הדרך, הצבתי לעצמי אתגר ממשמעותי: ליצור סביבת עבודה מקוונת לשפת C שתיה ידידותית למשתמש, בטוחה, ומאפשרת עבודה שיתופית בזמן אמת. היעד היה להציג את הכלים המתקדמיים לפיתוח בשפת C גם למתכנתים מתחילה.

התחלתי בחקירה עמוקה של הטכנולוגיות הקיימות. מצאתי שקיימות מספר פלטפורמות לעריכת קוד מקוונת, אך אף אחת לא התמקדה ספציפית בשפת C או לא הציעה את השימוש של עrica שיתופית, סביבת הרצה מאובטחת ואוצר תרגול ייודי.

הצלחות רבות היוו את הפרויקט. הצלחתי ליישם מערכת התחברות מאובטחת, עורך קוד שיתופי עם סנכרון בזמן אמת, וסביבת הרצה מבודדת לקוד C. במיוחד אני גאה במערכת התרגילים האוטומטית, שמאפשרת למשתמשים לתרגל את כישורי התכנות שלהם ולקלב משוב מיידי.

האתגרים היו רבים גם הם. ההתמודדות עם שיתוף קוד בזמן אמת הציבה אתגרים טכניים מורכבים, במיוחד בהיבטי סנכרון ופתרון התנשויות. יישום סביבת הרצה מאובטחת היה מוגבל גם כן, שכן שפת C מאפשרת גישה ישירה למשאבי המערכת, מה שהיא סיכון אבטחה ממשמעותי. פתרתי את הבעיה הללו באמצעות מחקר عميق, התניות עם מומחים, ויישום של פתרונות מתקדמיים כמו Operational Transformation וمبرגות משאבים.

### תהליך הלמידה

הפרויקט דרש מני להעמיק בתחוםים רבים ומגוונים:

1. **טכנולוגיות Full-Stack:** למדתי לעומק את Flask ו- SQLAlchemy בצד השרת, ו- JavaScript בצד הלקוח. עם SocketIO בצד הלקוח.
2. **תכנות בזמן אמת:** למדתי על תקשורת WebSocket והאלגוריתמים המורכבים הנדרשים לסנכרון עrica שיתופית.
3. **אבטחת מידע:** העמكتי בנושאי סביבות מבודדות (Sandboxing) הגבלה משאים, ומניעת התקפות כגון SQL Injection.
4. **הרצת קוד מרוחק:** למדתי כיצד להרכיב ולהנגן תחilibci קומפילציה והרצה באוּפן מאובטח.
5. **עיצוב ממשחק משתמש:** שיפרתי את מיזמיות ה- CSS, HTML ו- JavaScript שלי לייצור ממשחק ידידותי ותגובהית.

כל השלבים האלה דרשו במידה עצמית נרחבת. קראתי מאמרים, צפיתי בהרצאות, התיעצתי בפורומים, וערכתי ניסויים רבים. זו הייתה למעשה ההזדמנות מוגבלת במסגרת הלימודים הרגילה ביתי.

### כליים לעתיד

הפרויקט העניק לי כלים וידע רב שאקח איתי להמשך דרכי:

1. **מיומניות פיתוח Full-Stack:** הבנה عمוקה של פיתוח שירות-ללקוח API, ותקשורת בזמן אמת.
2. **יכולת תכנון ופיתוח מערכות מורכבות:** למדתי כיצד לחלק בעיה גדולה למודולים ולבאים, ולתכנן ארכיטקטורה מסודרת.

3. **הבנה عمוקה של אבטחת מידע:** רכשתי ידע מעשי רב על אבטחת יישומי ווב ומונעת איומים.
4. **ניהול פרויקט:** פיתחתי מיומנויות בתכנון, מעקב וניהול של פרויקט מורכב לאורך זמן.
5. **למידה עצמית:** שיפרתי את יכולת הלמידה העצמית שלי, שתשרה אותה בכלאתגר עתידי.

## תובנות מהתחלה

במהלך הפרויקט נעזרתי רבות בקהילה המפתחים. הצלרפי לפורומים של Flask ו- SocketIO, התיעצתי עם מומחי אבטחה, ושיתפתי קטעי קוד עם עמיתים לקבלת משוב. המידע הפתוח והנכונות של מפתחים לעזור זה לזה היו ממשמעותיים מאוד להצלחת הפרויקט.

נתקלתי בעיות שלא צפיתי מראש, כמו למשל הקושי בפתרון התנשויות בעריכה שיתופית או האגרים בהגנה על המערכת מפני קוד זדוני. אך כל בעיה הייתה הזדמנות ללמידה ולשיפור.

במברט לאחר, אם הייתה מתחילה את הפרויקט מחדש, הייתי מקדים יותר זמן לתכנון הארכיטקטורה והבנת האלגוריתמים של עריכה שיתופית לפני שהתחלה בכתיבת הקוד. הייתה גם שם דגש רב יותר על כתיבת בדיקות אוטומטיות מהתחלה.

אילו היו לי משאבים נוספים, הייתה מרחיב את הפרויקט עם תכונות כמו:

1. תמייה בשפות תכונות נוספות מלבד C
2. מערכת להערכת קוד איכותית יותר
3. אפשרות לדיבאג מתקדם
4. אינטגרציה עם מערכות ניהול גרסאות כמו Git
5. אפשרות להוראה מרוחק עם יכולות שיתוף מסך ווידיאו

## סיכום

הפרויקט "סביבת הרצה מקוונת לשפת C" היה מסע של התפתחות והעצמה. למדתי לא רק על טכנולוגיות ויישומים, אלא גם על עצמי כפתח וכЛОם. פיתחתי סבלנות, התמדה ויכולת להתמודד עם אתגרים מורכבים.

אני מאמין שהтвор הוסף מגשים את המטרה העיקרית: הנגשת כלי פיתוח מתקדמים לשפת C למתקנים בכל הרמות. המערכת היא ידידותית למשתמש, בטוחה, ומאפשרת שיתוף פעולה - בדיקת מה שחזיתי בתחילת הדרך.

אני מודה למנהל שלי, אופיר שביט, על ההכוונה והתמיכה לאורך כל הדרך. תודה גם למשפחה וחבריי שהתמכו בי ועזרו לי להתמיד גם ברגעים הקשים. ותודה מיוחדת לקהילה המפתחים הפתוחה, שהידע והניסיון שלהם היו משאב יקר ערך עבורו.

## ביבליוגרפיה

1. **Flask Project.** (2023). *Flask Quickstart - Request Routing and Templates*. Flask Documentation. <https://flask.palletsprojects.com/en/2.3.x/quickstart/#routing>
2. **Flask Project.** (2023). *Flask Sessions and User Authentication*. Flask Documentation. <https://flask.palletsprojects.com/en/2.3.x/quickstart/#sessions>
3. **Flask Project.** (2023). *Flask Blueprints for Application Modularization*. Flask Documentation. <https://flask.palletsprojects.com/en/2.3.x/blueprints/>
4. **Socket.IO.** (2023). *Server API - Rooms and Event Handling*. Socket.IO Documentation. <https://socket.io/docs/v4/server-api/#socketjoin>
5. **Socket.IO.** (2023). *Emitting Events and Broadcasting*. Socket.IO Documentation. <https://socket.io/docs/v4/emitting-events/>
6. **SQLAlchemy.** (2023). *ORM Tutorial - Declaring Models and Relationships*. SQLAlchemy Documentation. [https://docs.sqlalchemy.org/en/20/tutorial/orm\\_data\\_manipulation.html](https://docs.sqlalchemy.org/en/20/tutorial/orm_data_manipulation.html)
7. **SQLAlchemy.** (2023). *Relationship Configuration - Many-to-Many*. SQLAlchemy Documentation. [https://docs.sqlalchemy.org/en/20/orm/basic\\_relationships.html#many-to-many](https://docs.sqlalchemy.org/en/20/orm/basic_relationships.html#many-to-many)
8. **CodeMirror.** (2023). *User Manual - Basic Usage and C-like Mode*. CodeMirror Documentation. <https://codemirror.net/5/doc/manual.html#usage>
9. **CodeMirror.** (2023). *C-like Mode for Syntax Highlighting*. CodeMirror Documentation. <https://codemirror.net/5/mode/clike/>
10. **Mozilla Developer Network.** (2023). *Using WebSockets for Real-time Communication*. MDN Web Docs. [https://developer.mozilla.org/en-US/docs/Web/API/WebSockets\\_API/Writing\\_WebSocket\\_client\\_applications](https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API/Writing_WebSocket_client_applications)
11. **PostgreSQL Global Development Group.** (2023). *PostgreSQL Tutorial - Creating Tables and Relationships*. PostgreSQL Documentation. <https://www.postgresql.org/docs/current/tutorial-table.html>

**נספחים****נספח א' : פרוטוקול תקשורת WebSocket**

הודעות מהלкова לשרת

**connect**

```
{
  "type": "connect",
  "data": {
    "project_id": "string"
  }
}
```

**join\_project**

json

```
{
  "type": "join_project",
  "data": {
    "project_id": "string"
  }
}
```

**edit**

```
{
  "type": "edit",
  "data": {
    "type": "insert|delete|retain",
    "position": "number",
    "text": "string",
    "project_id": "string"
  }
}
```

**cursor\_move**

```
{
  "type": "cursor_move",
  "data": {
    "x": 100,
    "y": 200
  }
}
```

```
"project_id": "string",
"position": {
  "line": "number",
  "ch": "number"
}
}
```

### chat\_message

```
{
  "type": "chat_message",
  "data": {
    "project_id": "string",
    "message": "string"
  }
}
```

הודעות מהשרת ללקוח

### document

```
json
{
  "type": "document",
  "data": {
    "text": "string"
  }
}
```

### user\_connected

```
{
  "type": "user_connected",
  "data": {
    "username": "string",
    "user_id": "string",
    "sid": "string"
  }
}
```

**user\_disconnected**

```
{
  "type": "user_disconnected",
  "data": {
    "username": "string",
    "sid": "string"
  }
}
```

**cursor\_update**

```
{
  "type": "cursor_update",
  "data": {
    "user_id": "string",
    "username": "string",
    "position": {
      "line": "number",
      "ch": "number"
    }
  }
}
```

**new\_chat\_message**

```
{
  "type": "new_chat_message",
  "data": {
    "user_id": "string",
    "username": "string",
    "message": "string",
    "timestamp": "ISO8601_string"
  }
}
```

**נספח ב' :** שמות תיקיות וקבצים שהוגבלו לשימוש בסנדבוקס

**תיקיות אסורה:**

- קובצי הגדרות מערכת - /etc/

- נתונים משתנים של המערכת - /var
- (תוכנות משתמש (קריאה בלבד מותרת - /usr)
- תיקיות בית משתמשים אחרים - /home
- תיקיות מנהל המערכת - /root
- מידע על תהליכי - /proc
- מידע על מערכת - /sys
- קובצי התקנים - /dev

**קבצים מוגבלים:**

- רשימת משתמשים - /etc/passwd
- סיסמאות מוצפנות - /etc/shadow
- הגדירות רשת - /etc/hosts
- כל קובץ ההתחלה ב-

**הרחבות קבצים אסורות:**

- .sh קוביshell script
- .py קוביPython
- .pl קוביPerl
- .php קוביPHP

**נספח ג' : אלגוריתם בדיקת קוד לתבניות מסוכנות****רגקסים לזיהוי פונקציות מסוכנות:**

```
dangerous_patterns = [
    r'system\s*\()',      # system() calls
    r'popen\s*\()',      # popen() calls
    r'fopen\s*\()',      # File operations
    r'fwrite\s*\()',      # File write operations
    r'fprintf\s*\()',     # File write operations
    r'exec[lv][pe]\s*\()', # exec family functions
    r'fork\s*\()',        # Process creation
    r'unlink\s*\()',      # File deletion
    r'remove\s*\()',      # File deletion
    r'rename\s*\()',      # File renaming
    r'mkdir\s*\()',       # Directory creation
    r'rmdir\s*\()'         # Directory removal
]
```

**אלגוריתם הבדיקה:**

```
def check_for_dangerous_code(code):
    import re
    for pattern in dangerous_patterns:
        if re.search(pattern, code):
            return True
    return False
```

**קוד הפרויקט :**

קוד הפרויקט יוצרף בעמודים הבאים, בתחילת כל section של קוד נמצא שם הקובץ.