**ארכיטקטורת הפרויקט:**

הפרויקט מכיל שלושה רכיבים עיקריים: שרת (Server), לקוח משדר (TC) ולקוח מקבל (RC).

לכל אחד מהרכיבים יש חלק משמעותי ושונה בפרויקט

**תיאור הארכיטקטורה המוצעת:**

שרת (Server):

השרת אחראי על העברת המידע בין הלקוח המשדר (TC) ללקוח המקבל (RC).

נוסף על כך השרת אחראי על פעולת התמלול והכתיבה למסד הנתונים.

השרת מתחלק לשלושה חלקים: TC\_Handler, RC\_Handler & Transcriptor.

כל אחד מהחלקים אחראי על אחת מהפעולות הראשיות של השרת (קבלה מלקוח משדר, תמלול וסכרון הוידאו והטקסט, העברה ללקוח המקבל).

TC\_Handler:

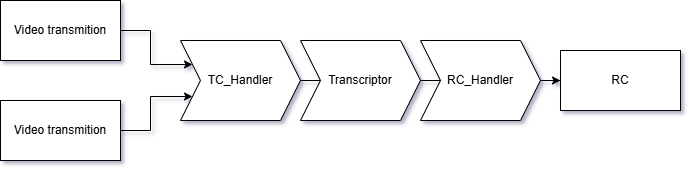
חלק זה בשרת אחראי על קבלת וידאו ואודיו מהלקוח המקבל בעזרת חיבור UDP עם פרוטוקול RTP ומסנכרן ביניהם לפני שהוא מעביר לרכיב השרת הבא.

Transcriptor:

חלק זה לוקח את האודיו ובעזרת קריאת API במטרה לתמלל את האודיו אשר התקבל מה-TC

RC\_Handler:

חלק זה אחראי על שליחת הוידאו והתמלול ל-RC בעזרת חיבור TCP, HTTP, RTP ו-WebSockets.



לקוח משדר (TC):

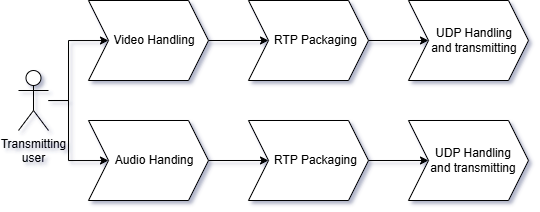
הלקוח המשדר אחראי על שידור הוידאו והאודיו לשרת.

רכיב זה הוא תוכנה לוקלית שמשדרת בעזרת חיבור UDP ופרוטוקול ה-RTP אודיו ווידאו בשני תהליכים נפרדים (פורט לוידאו ופורט לאודיו)

רכיב זה מכיל שני חלקים: Audio\_Handler, Video\_Handler.

כל חלק מהרכיב בנוי משלושה שלבים עיקריים: קליטה - הכנסה לפקטה – שליחה.

לדוגמא: קליטה של וידאו -> הכנסה לפקטת RTP -> שליחה לשרת.



לקוח מקבל (RC):

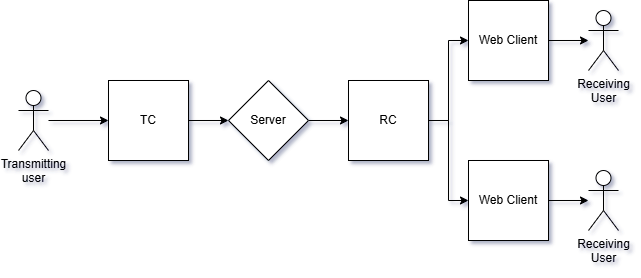
תפקידו של רכיב ה-RC הוא הצגת שידורי וידאו ותמלול המתקבלים באופן שוטף מהשרת.

יישום זה מבוסס ווב (Web-based) ומנצל את פרוטוקול WebSocket להעברת נתונים בצורה מהירה ויעילה, המבטיחה תקשורת בזמן אמת (RTC) חיונית לשידורים חיים.

היתרונות בבחירת ארכיטקטורת ווב עבור ה-RC:

* נגישות גבוהה: המערכת נגישה מכל מכשיר המצויד בדפדפן אינטרנט.
* פיתוח ופריסה יעילים: קלות המימוש והתחזוקה בסביבת ווב.
* מתאים למגון רחב של פלטפורמות (cross-platform): עבודה חלקה על פני מערכות הפעלה ודפדפנים שונים ללא צורך בהתאמות ספציפיות.
* חווית משתמש נוחה: ממשק מוכר ואינטואיטיבי המקל על השימוש.

כל שלושת הרכיבים הללו ביחד מנהלים את המוצר הסופי והכרחיים לתפעולו היעיל.



**תיאור הטכנולוגיה הרלוונטית:**

שפות תכנות: השפות השונות המשומשות בפרויקט הזה.

|  |  |  |
| --- | --- | --- |
| שם השפה | שימוש | סיבה לשימוש |
| Python | Server, RC, TC | שפה ראשית של הפרויקט.  נוחה לשימוש, כתיבה מהירה, מכילה ספריות רבות ומאוד ורסטילית. מוכרת במימוש הנוח שלה לרשתות |
| TypeScript | RC | משומשת למימוש ה-RC.  השפה משומשת בצורה נרחבת בעולם פיתוח הווב. נותנת את כל מה שיש ב-JS אבל עם בטחון של OOP ושימוש ב-Types. |
| HTML5 & CSS | RC | שפות ידועות לפיתוח ועיצוב אתרים. |
| SQL | Server | שפה ידועה ובטוחה לכתיבת מסדי נתונים. |

מערכות הפעלה: מערכות ההפעלה המשומשות בפרויקט

|  |  |  |
| --- | --- | --- |
| שם המערכת | שימוש | סיבה לשימוש |
| Windows | Server, RC, TC | מערכת הפעלה ידועה, נוחה לשימוש, ורסטילית ושומשה לפיתוח ובדיקות. |

ניתן להריץ את הפרויקט על כל מערכת הפעלה כל עוד מותקנים התוספים הרלוונטים

פרוטוקולים: הפרוטוקולים המשומשים בפרויקט

|  |  |  |
| --- | --- | --- |
| שם הפרוטוקול | שימוש | סיבה לשימוש |
| RTP | Server, RC, TC | פרוטוקול ידוע בתעשייה, נוח לשימוש וורסטילי |
| HTTP | Server, RC | פרוטוקול בו משתמשים בשביל לתקשר עם ה-browser |
| Websocket | Server, RC | פרוטוקול מעל HTTP שנותן לשרת יכולת לשלוח לקוח ווב הודעות ללא בקשה של הלקוח. |
| TCP | Server, RC | פרוטוקול אמין, מוודא הגעת הודעות, חובה לשימוש ב-HTTP |
| UDP | Server, TC | פרוטוקול מהיר יותר וידוע בשימוש בשביל סטרימינג |

תחומי עניין רלוונטים: שונות ששומשו בפרויקט.

???????