

**אפיון פרויקט - EchoRTC**

**מגיש: אורי כהן**

**כיתה: יב 2**

**בית ספר: הנדסאים**



## תוקן עניינים

|   |    |
|---|----|
| <b>תקציר מנהלים</b>                               | 6  |
| <b>פתיחה ותיאור כללי</b>                          | 8  |
| <b>הצעת הפרויקט המקורית</b>                       | 9  |
| <b>אפיון הפרויקט</b>                              | 10 |
| הגדרה   | 10 |
| קהל יעד   | 10 |
| <b>מטרות הפרויקט</b>                              | 11 |
| מטרות עיקריות                                     | 11 |
| מטרות מתקומות                                     | 11 |
| יעדי הפרויקט                                      | 11 |
| <b>תחום הפרויקט (תיקום)</b>                       | 12 |
| תחומים הכלולים בפרויקט                            | 12 |
| תחומים שאינם כלולים בפרויקט                       | 12 |
| <b>חקר מוצרים</b>                                 | 13 |
| <b>חker פיתוחי</b>                                | 14 |
| <b>סביבה</b>                                      | 16 |
| חומרה   | 16 |
| המערכת פותחה והורצתה על מחשב אישי סטנדרטי, הכלול: | 16 |
| מערכות הפעלה                                      | 16 |
| תוכנה וכלי פיתוח                                  | 16 |
| <b>סקירה טכנולוגיות הפרויקט</b>                   | 17 |
| WebRTC  | 17 |
| מאפשר: WebRTC בשימוש ב-                           | 17 |
| צד לקוח –   | 17 |
| ממומשים: JavaScript באמצעות                       | 17 |
| Python (Signaling) צד שרת –                       | 18 |
| בלבד. השרת Python יפותח בברוב שפה                 | 18 |
| אחרראי על:  | 18 |
| WebSocket - WSS                                   | 18 |
| מוועברות: WebSocket באמצעות                       | 18 |

|   |           |
|---|-----------|
| Machine Learning .....  | 18        |
| מערכת עיבוד הווידאו מבוססת על אלגוריתמים חישוביים שפותחו מאפסו,<br>ללא שימוש בספריות חיצונית לעיבוד תמונה או למידת מכונה. המימוש<br>מתבסס על: ..... | 18        |
| מסד נתונים .....  | 19        |
| מוד הנתונים בשרת משתמש, בין היתר, ל: .....  | 19        |
| <b>תיאור מפורט של המערכת .....</b>  | <b>20</b> |
| ארQUITקטורה כללית של המערכת .....   | 20        |
| רכיבי המערכת המרכזיים .....   | 20        |
| ארQUITקטורה Client – Browser .....  | 20        |
| ארQUITקטורה Signaling Server – צד שרת .....   | 21        |
| מוד נתונים מצד השרת .....   | 21        |
| מוד נתונים מצד.Mesh Peer-to-Peer .....  | 22        |
| לדוגמא: .....   | 22        |
| תדרונות גישה זו: .....  | 22        |
| חסרונות: .....  | 22        |
| זרימת המידע במערכת .....  | 23        |
| 1. ערוץ איותה (Signaling) .....   | 23        |
| 2. ערוץ מדיה (Media) .....  | 23        |
| תשתיית עיבוד הווידאו .....  | 23        |
| פועלת מצד הלקוח, כחלק מצינור עיבוד המדיה<br>המקומי. עיבוד זה מתבצע לפני שליחת הר-<br>Peers .....  | 23        |
| <b>פרוטוקולי התקשרות במערכת .....</b>   | <b>24</b> |
| הפרוטוקולים במערכת נחלקים לשתי קבוצות עיקריות: .....  | 24        |
| WebRTC .....  | 24        |
| RTCPeerConnection .....   | 24        |
| כוללים: RTCPeerConnection .....   | 24        |
| SDP – Session Description Protocol .....  | 25        |
| הוא פורטט טקסטואלי המתאר את מאפייני החיבור בין שני SDP<br>משמש לתיאור: WebRTC, SDP .....  | 25        |
| משתתפים. במסגרת ICE – Interactive Connectivity Establishment .....  | 25        |
| Offer / Answer: .....   | 25        |
| אחראי על ICE .....  | 25        |

|  |           |
|--|-----------|
| STUN – Session Traversal Utilities for NAT.....                    | 25        |
| TURN – Traversal Using Relays around NAT.....                      | 26        |
| WebSocket Secure (WSS) – Signaling.....                            | 26        |
| - WebSocket Secure – שירת Signalign שעשו שימוש בפרוטוקול WebSocket |           |
| ... לצורך החלפת הודעות בין הלקוחות. עroz זה משמש להעברת WSS        | 26        |
| ה הפרדה בין ערוצי תקשורת.....                                      | 26        |
| אחד העקרונות המרכזיים במערכת הוא ההפרדה המלאה בין:                 | 26        |
| ה הפרדה זו:.....   | 26        |
| בזרימת התקשרות Machine Learning שילוב UIBOD וידאו מבօס             | 27        |
| ה הפרדה זו אפשרה:.....   | 27        |
| 0.יכום הפרק.....   | 27        |
| <b>אבטחת מידע והצפנה במערכת.....</b>                               | <b>28</b> |
| (Signalizing) אבטחת שכבת האיתות.....                               | 28        |
| אמצעי האבטחה בשכבות האיתות כוללים:.....                            | 28        |
| DTLS-SRTP – הצפנה מדיה.....  | 29        |
| העברת המדיה (וידאו וקול) במערכת מתבצעת באמצעות WebRTC              | 29        |
| תוך שימוש במנגנוני הצפנה מובנים: Peer-to-Peer.....                 | 29        |
| תהליך זה מבטיח:.....   | 29        |
| הפרדת תפקדים וצמצום משטח התקיפה.....                               | 29        |
| במערכת קיימת הפרדה ברורה בין רכיבי המערכת:.....                    | 29        |
| ה הפרדה זו:.....   | 29        |
| TURN ו-STUN אבטחת תקשורת מול שרת.....                              | 30        |
| TURN גם כאשר נעשה שימוש בו:.....                                   | 30        |
| פרטיות משתמשים.....  | 30        |
| שמירה על פרטיות המשתמשים הייתה שיקול מרכזי בתכנון המערכת.          |           |
| בהתאם לכך:.....  | 30        |
| 0.יכום אבטחת המידע.....  | 30        |
| <b>אלגוריתמים מרכזיים במערכת.....</b>                              | <b>31</b> |
| אלגוריתם יצרת חדר וניהול תפקדים.....                               | 31        |
| בעת יצרת חדר חדש במערכת, מתבצע תהליך לוגי הכלול:.....              | 31        |
| אלגוריתם ה가입ות משתמש לחדר קיימ.....                                | 31        |
| כאשר משתמש חדש מצטרף לחדר פעיל:.....                               | 31        |
| Mesh Peer-to-Peer.....   | 32        |
| ICE Candidates.....  | 32        |
| במהלך יצרת החיבור ולאחריו, כל משתמש:.....                          | 32        |

|  |           |
|--|-----------|
| אלגוריתם ניהול מדיה בזמן אמת   | 32        |
| לאחר יצירת החיבורים, מתבצע ניהול שוטף של זרמי המדיה:                 | 32        |
| אלגוריתם ניתוק והთואשות מחיבור שנפל                                  | 33        |
| במקרה של ניתוק משתתף או נפילת חיבור:                                 | 33        |
| אלגוריתם שילוב עיבוד וידאו בצד הלקוח                                 | 33        |
| אחרים, ניתן לשלב Peers- <i>MediaStream</i> בצד הלקוח, לפניה שילחת ה- | 33        |
| שלב עיבוד מוקדים:  | 33        |
| סיכון האלגוריתמים  | 33        |
| <b>תיאור פונקציוני של המערכת</b>                                     | <b>34</b> |
| ניהול משתמשים  | 34        |
| (Sign Up) תהליך הרשמה  | 34        |
| בעת הרשמה למערכת, מתבצע התהליך הבא:                                  | 34        |
| מנגנון זה נועד:  | 34        |
| (Login) תהליך התחברות  | 34        |
| לאחר הרשמה וAIMOT מוצלח, המשתמש יכול להתחבר למערכת:                  | 34        |
| ניהול חדרים  | 35        |
| יצירת חדר  | 35        |
| הצטרפות לחדר   | 35        |
| הרשאות ותפקידים  | 35        |
| Host   | 35        |
| Participant  | 35        |
| abinetat midu בתהליכי משתמש  | 36        |
| סיכון התיאור הפונקציוני  | 36        |
| <b>User Interface (User Interface) ממשך משתמש</b>                    | <b>37</b> |
| (Login) מסך כניסה  | 37        |
| מסך הכניסה הוא נקודת ההתחלה של המערכת. במסך זה המשתמש                | 37        |
| מתבקש להזין:   | 37        |
| (Sign Up) מסך הרשמה  | 37        |
| מסך הרשמה מאפשר יצירת חשבון חדש במערכת. המשתמש                       | 37        |
| מצין:  | 37        |
| מסך ראשי / לובי  | 38        |
| מןו, (Lobby) לאחר התחברות מוצלחת, המשתמש מועבר למסך הראשי            | 38        |
| ניתן:  | 38        |
| (Room View) מסך חדר  | 38        |
| תצוגת וידאו  | 38        |

|  |           |
|--|-----------|
| בקרי מדיה.....   | 38        |
| צ'אט טקסטואלי.....   | 38        |
| לצד שיחת הווידאו, קיים צ'אט טקסטואלי בזמן אמת, המאפשר:           | 38        |
| HOST – מסך ניהול.....  | 39        |
| מצג ממוקן ניהול נוסף, הcoil: HOST למשתמש המוגדר כ-               | 39        |
| חוויית משתמש ושיקול עיצוב  | 39        |
| בעת תכנון הממשק הושם דגש על:                                     | 39        |
| 5.יכום ממשק המשתמש   | 39        |
| <b>        מפרט תכנון מפורט – Design Detailed.....</b>           | <b>40</b> |
| המערכת בנויה בגישה מודולרית, תוך הפרדה ברורה בין:                | 40        |
| מבנה כללי של הפרויקט   | 40        |
| הפרויקט מחולק למספר תיקיות וקבצים עיקריים, כאשר כל רכיב אחראי על | 40        |
| שכבה לוגית שונה במערכת:  | 40        |
| Client – צד לקוח.....  | 41        |
| HTML קבצי.....   | 41        |
| CSS קבצי.....  | 41        |
| JavaScript – לוגיקת WebRTC.....                                  | 41        |
| ניהול חיבורים מרובים.....  | 42        |
| גישה זו מאפשרת:  | 42        |
| Signalizing Server – צד שרת.....                                 | 42        |
| תפקידים מרכזיים של השרת.....                                     | 42        |
| מבנה קוד השרת.....   | 42        |
| בסיס נתונים.....   | 43        |
| הມידע הנשמר כולל:  | 43        |
| Event Handling (טיפול באירועים).....                             | 43        |
| לצורך (Events) המערכת עשוה שימוש באירועים                        | 43        |
| Machine Learning (של עיבוד ידאו).....                            | 43        |
| תכנון זה מאפשר:  | 43        |
| 5.יכום Design Detailed.....                                      | 44        |
| <b>        עיצוב ואלגוריתמים – Machine Learning – תשתיית</b>     | <b>45</b> |
| עקרונות תכנון כלליים.....  | 45        |
| בפרויקט פותחה בהתאם לעקרונות תשתית ה- Machine Learning – הבאים:  | 45        |
| בארQUITקטורה ML מיקום תשתיית ה-.....                             | 45        |
| מבנה זה מאפשר:   | 45        |

|  |           |
|--|-----------|
| קליטת פרויומים ועיבוד בסיסי                                    | 46        |
| מتابعות פעולות חישוביות כגון: <i>PyMongo</i> באמצעות           | 46        |
| דיאוי ומוקב אחר תווים פנימית – גישה אלגוריתמית                 | 46        |
| האלגוריתם מבצע:  | 46        |
| פילטרים חישוביים בזמן אמת                                      | 46        |
| על בסיס ניתוח התמונה ניתן להפעיל פילטרים חישוביים שונים, כגון: | 46        |
| שיקולי ביצועים   | 47        |
| עיבוד וידאו בזמן אמת מהיבר התייחסות לביצועים:                  | 47        |
| מגבלי המערכת   | 47        |
| בפרויקט קיימות מגבלי ידועות: <i>Machine Learning</i> לתשתית ה- | 47        |
| הרחבות עתידיות   | 47        |
| התשתייה תוכנה כך שתאפשר בעתיד:                                 | 47        |
| <i>Machine Learning</i> סיכון תשתיית ה-                        | 47        |
| <b>בדיקות (Testing)</b>  | <b>48</b> |
| בדיקות צד לקוח (Client Testing)                                | 48        |
| בדיקות חיבור מצלמה ומייקרופון                                  | 48        |
| בדיקות WebRTC  | 48        |
| בדיקות בקרி מדיה   | 48        |
| בדיקות צד שרת (Server Testing)                                 | 49        |
| בדיקות <i>Signaling</i>  | 49        |
| בדיקות ניהול חדרים   | 49        |
| בדיקות ניהול משתמשים ואימונות                                  | 49        |
| בדיקות אבטחת מידע  | 49        |
| בדיקות אלו בוצעו באמצעות:                                      | 49        |
| בדיקות קצה (Edge Cases)  | 50        |
| בדיקות <i>Machine Learning</i>                                 | 50        |
| סיכון הבדיקה   | 50        |
| <b>הוראות התקינה והפעלה</b>                                    | <b>51</b> |
| דרישות מערכת   | 51        |
| צד לקוח  | 51        |
| צד שרת   | 51        |
| התקנת הצד שרת ( <i>Signaling Server</i> )                      | 51        |
| הפעלת הצד לקוח   | 52        |
| הפעלת המערכת בסביבה מאובטחת                                    | 52        |
| תקלות נפוצות ופתרונות  | 52        |

|                     |           |
|---------------------|-----------|
| <b>מדריך למשתמש</b> | <b>53</b> |
| ההרשמה למערכת       | 53        |
| התחברות למערכת      | 53        |
| יצירת חדר חדש       | 53        |
| הצטרפות לחדר קיימ   | 53        |
| שימוש במהלך שיחה    | 54        |
| (Host) ניהול חדר    | 54        |
| סיום שיחה           | 54        |
| סיכום מדריך המשתמש  | 54        |
| <b>סיכום ומשוב</b>  | <b>55</b> |

## תקציר מנהליים

פרויקט זה עוסק בפיתוח מערכת WebRTC מתקדמת לשיחות וידאו, קול ושיתוף מסך מאובטחות, המיועדת לתקשורת בזמן אמת בין מספר משתתפים במקביל. המערכת מבוססת על ארכיטקטורת Peer-to-Peer מרובה לקוחות (Mesh), בה כל משתתף יוצר חיבור ישיר ומוצפן מול כל משתתף אחר בחדר, ללא שימוש בשרת מדיה מרכזי.

המערכת פועלת מתוך דפדפן אינטרנט סטנדרטי, ללא צורך בהתקנת תוכנה חיונית, ומאפשרת יצירה חדרי שיחה מאובטחים, הSharperות משתמשים רבים, שיחות וידאו וקול בזמן אמת, שיתוף מסך, צ'אט טקסטואלי, וניהול הרשות מתיקdem באמצעות תפקיד מרוח (Host). תפקיד זה נוצר אוטומטית עם יצירת החדר ומאפשר שליטה על המשתתפים והפעולות המותרות במהלך השיחה.

בחירת הפרויקט נבעה מהרzon להעמק בהבנת תחום התקשורת בזמן אמת, בדגש על פרוטוקול WebRTC, רשותות מחשבים, אבטחת מידע והתחממות עם מגבלות רשת כגון NAT ו-IPv4. הפרויקט משלב ידע תיאורתי ומעשי בתחום אלו, וממש פתרון מלא מקצה לקצה הכלולצד ללקוח הצד שרת.

צד הלקוון פותח בשפת JavaScript ומנצל את ממשקי ה-WebRTC API, ובפרט את RTCPeerConnection ו-MediaStreams. לצורך יצירה חיבורו מדיה מוצפנים והעברת וידאו וקול ישירות בין המשתתפים. הצד השרת פותח בשפת Python ומשמש כשרת Signaling ייעודי המבוסס על WebSocket מאובטח (WSS), שתפקידו לנוהל את תהליכי האיתות בלבד – החלפת הודעות SDP ו-ICE Candidates – מבלי לעבוד או להעביר מדיה בפועל.

לצורך התמודדות עם מגבלות רשת ותרכישים של חסימת תקשורת 'שירות', המערכת עשויה שימוש בפרוטוקולי ICE, STUN ו-TURN, תוך הסתמכות על שירותי TURN חיצוניים במקרה הצורך. כל תüberות המדיה במערכת מוצפנת מוקצה לקצה באמצעות מנגנון RTP DTLS-SRTP המובנים ב-WebRTC, דבר המבטיח רמת אבטחה ופרטיות גבוהה במיוחד.

בנוסף ליכולות התקשרות בזמן אמת, הפרויקט כולל תכנון ומימוש של תת-מערכת יעודית לעיבוד וידאו מתקדם בצד הלוקוח, המבוססת על אלגוריתמים של Machine Learning. תת-מערכת זו מיועדת לאפשר הוספה פילטרים חכמים למצימה בזמן אמת, כגון פילטרים המבוססים על זיהוי ומעקב אחר תווי פנים.

מערכת ה-*הוֹגְרֵן* Machine Learning פותחה בגישה Low-Level, ללא שימוש בספריות עיבוד תמונה או למידת מכונה מוכנות (כגון OpenCV, TensorFlow או PyTorch), ותוך הסתמכות על ספריית PyTorch בלבד לצורך חישובים מתמטיים ואלגוריתמיים. גישה זו נבחרה במכoon על מנת להבין לעומק את עקרונות הפעולה של עיבוד תמונה, זיהוי תבניות וזרימת נתונים במודלים חישוביים, ולא כהטעה "שchorה" של ספריות חיצונית.

חלק מתשתיית זו, בנייתם מסד נתונים ייודי שנוצר במיוחד במכשיר עבור מערכת ה-*הוֹגְרֵן*, ומשמש לאחסון נתונים אימון, תיוגים, פרמטרים ומידע נדרש הדרוש לפיתוח ולבדיקה של האלגוריתמים. מסד נתונים זה מאפשר שליטה מלאה בנתונים, התאמת עתידית למודלים נוספים והרחבת הדרגתית של יכולות המערכת.

מערכת הפילטרים תוכנה להשתלב בעתיד 'שירות' בzinor עיבוד הוויידאו של WebRTC בצד הלוקוח, כך שיעיבוד התמונה יבוצע לפני שליחת ה-*MediaStream* למשתתפים האחרים, ללא פגיעה בפרטיות ולא העברת מידע חזותי לשרת מרכזי.

המערכת יכולה לבנותה בארכיטקטורה מודולרית, המאפשרת הפרדה ברורה בין שכבות התקשרות, האבטחה, ממשק המשתמש ועיבוד הוויידאו. פרויקט זה מדגים שימוש מלא של מערכת תקשורת בזמן אמת, תוך שליטה בפרוטוקולים, באבטחת המידע, בניהול חיבורים מרובים ובתשתיית Machine Learning עצמאית, ומהווה בסיס יציב להמשך פיתוח, הרחבה ושימוש מעשי במערכות תקשורת מבוזרות מתקדמות.

## פתיחה ותיאור כללי

בעשור האחרון חלה עליה משמעותית בשימוש במערכות תקשורת בזמן אמת מבוססות וידאו, הן לצרכים אישיים והן לצרכים מקצועיים וארגוניים. מערכות אלו נדרשות לספק אינטראקטיביות גבוהה, זמן השהייה נמוך, רמת אבטחה גבוהה ויכולת עבודה בסביבות רשת מגוונות, היכולות מגבלות כגון NAT, Firewalls ועוד פס משטנה.

פרויקט זה עוסק בפיתוח מערכת WebRTC לשיחות וידאו, קול ושיטוף מסך מאובטחות, הפעלת ארכיטקטורת Peer-to-Peer מרובה משתתפים (Mesh). המערכת נבנתה מtopic מטרה להבין לעומק את מנגנון הפעולה של WebRTC ברמת הפרוטוקולים, ללא הסתמכות על ספריות עזר חיצונית או שירותי מדיה מוכנים, וtopic שליטה מלאה בכל שכבות המערכת – החל משכבה הרשות והאינטראקטיבית ועד לעיבוד המדיה מצד הלkop.

בנוסף ליכולות התקשרות בזמן אמת, המערכת תוכננה כך שתאפשר הרחבת עתידית בתחום עיבוד הוידאו מצד הלkop. חלק מתכון זה, פותחה תשתיית ראשונית למערכת פילטרים למלמה המבוססת על Machine Learning, היכולת מנגן זיהוי ומעקב אחר תווים פנים, אשר תושלב בעתיד במערכת ה-WebRTC עצמה.

מערכת הפילטרים פותחה בגישה "מאפס", ללא שימוש בספריות עיבוד תמונה או למידת מכונה מוכנות (כגון OpenCV או frameworks ייעודיים), וtopic הסתמכות בסיסית בלבד על ספריית NumPy לצורכי חישובים מתמטיים. בנוסף, בניית מסד

נתוני ייוז'ר שנוצר במיוחד למערכת זו, לצורך ניהול נתונים אימון, תיוגים ונתוני עזר, חלק מהמערכות להרחבת יכולות ה-Machine Learning בפרויקט.

## הצעת הפרויקט המקורי

ההצעה המקורי של הפרויקט כוללת פיתוח מערכת תקשורת בזמן אמת מבוססת WebRTC, שתאפשר שיחות וידאו וקול בין מספר משתתפים במקביל, ללא תלות בשרת מדיה מרכזי. המערכת תוכננה לפעול מתוך דפדפן אינטרנט, תוך שימוש בפרוטוקולים סטנדרטיים ופתוחים, ובdagש על אבטחת מידע ופרטיות המשתמשים.

לשם כך, הוגדרו המרכיבים המרכזיים הבאים:

- צד לקוח מבוסיס JavaScript, האחראי על יצירת חיבור RTCPeerConnection, ניהול זרמי מדיה (וידאו, קול ושיתוף מסך) והציג ממשק משתמש אינטראקטיבי.
- צד שרת מבוסיס Python, המשמש כשרת Signaling בלבד, באמצעות WebSocket מאובטח (WSS), לצורך תיאום החיבורים והחלפת הודעות SDP ו-ICE Candidates.
- שימוש ב프וטוקולי ICE, STUN ו-TURN לצורך התמודדות עם מגבלות רשת ומעבר NAT.
- הצפנה כלל תעבורת המדיה באמצעות מנגנוני DTLS-SRTP המובנים ב-WebRTC.
- ניהול משתמשים, הרשאות וחדרים, כולל תפקיד Host עם יכולות שליטה.

בנוסף, כבר בשלב ההצעה הוגדרה מטרה עתידית לשלב במערכת יכולות ניהול יידאו מתקדמות בצד הלקלוח, ובפרט פילטרים חכמים מבוססים על ניתוח תמונה

Machine Learning. לשם כך, הוחלט לפתח תשתיית עצמאית ל<sup>לזריה</sup> תווּי פנימִי, אשר לא תישען על ספריות חיצונית כבדות, אלא תבסס על מימוש אלגוריתמי עצמאי ונתוני אימון שנבנו במיוחד עבור הפרויקט.

בחירה זו נועדה להעמיק את ההבנה בתחום עיבוד התמונה ולמידת המוכנה, ולאפשר שליטה מלאה בלוגיקה, בתנאים ובאופן האינטגרציה העתידי עם מערכת WebRTC.

## אפיון הפרויקט

### הגדרה

הפרויקט מוגדר כמערכת תקשורת בזמן אמת מבוססת WebRTC, המאפשרת שיחות וידאו, קול ושיתוף מסר מאובטחות בין מספר משתמשים במקביל, תוך שימוש בארכיטקטורת Peer-to-Peer מרובה לקוחות (Mesh). המערכת פועלת מתוך דפדף אינטרנט ואינה דורשת התקינה של תוכנה חיצונית, שרת מדיה מרכזי או תשתיית קניינית.

בנוסף למערכת התקשרות, הפרויקט כולל תת-מערכת עצמאית לעיבוד וידאו מבוססי Machine Learning, המיועדת להויספט פילטרים חכמים למצלה בזמן אמת. תת-מערכת זו פותחה בגישה אלגוריתמית נמוכה (Low-Level), ללא שימוש בספריות עזר חיצונית לעיבוד תמונה או למידת מכונה, ומ坦בסת על מימוש עצמאי -PyTorch בלבד.

שילוב שתי המערכות נועד להציג שליטה מלאה הן בתחום התקשרות בזמן אמת והן בתחום עיבוד הווידאו החישובי, תוך שמירה על אבטחה, פרטיות ויכולת הרחבה עתידית.

### קהל יעד

המערכת מיועדת למגוון רחב של משתמשי קצה וגורמים טכנולוגיים, בהם:

- משתמשים פרטיים המעוניינים בשיחות וידאו מאובטחות ללא תלות בפלטפורמות מסחריות
- צוותי עבודה מרוחקים הזקוקים לתקשורת בזמן אמת עם שליטה גבוהה בפרטיות
- מערכות לימוד מרוחק ותרגול טכנולוגי
- מפתחים וחקרנים המעוניינים למדוד ולהתנסות ב-*C* WebRTC בرمת הפרוטוקולים
- גורמים המעוניינים במערכות עיבוד וידאו חכמות הצד הלוקה, ללא העברת מדיה לשרת

המערכת אינה מיועדת לשימוש מסחרי רחוב היקף (כגון אלף משתמשים במקביל), אלא לפתרונות מבוקרים, לימודים ומודולריים, בהם נדרש שליטה מלאה בקוד ובתשתיות.

## **מטרות הפרויקט**

### **מטרות עיקריות**

- מימוש מערכת WebRTC מלאה לשיחות וידאו וקול לרבות משתתפים
- הבנה ומימוש של ארכיטקטורת Mesh Peer-to-Peer
- הקמת שרת Signaling עצמאי ב-*Python* מבוסס WebSocket
- שימוש בפרוטוקולי ICE, STUN ו-TURN להתחממות עם NAT ו-Firewalls
- הבטחת הצפנה מקצועית לפחות תעבורת המדיה (DTLS-SRTP)
- פיתוח ממשק משתמש אינטואיטיבי מבוסס דפדף

### **מטרות מתקדמות**

- פיתוח תשתיית Machine Learning לעיבוד וידאו בזמן אמת הצד הלוקה
- מימוש אלגוריתמים לזיהוי ומעקב אחר תווים פנימיים ללא ספירות עזר חיצונית
- בניית מסד נתונים עצמאי לנוטוני אيمון, תיוג ופרמטרים אלגוריתמיים
- הcnnetת תשתיית לאינטגרציה עתידית בין מערכת ה-ML ל-*MediaStream* של WebRTC
- שמירה על פרטיות מלאה – עיבוד הוידאו מתבצע מקומי בלבד

## יעדי הפרויקט

יעדי הפרויקט הוגדרו כך שייהיו מדדים וברורים:

- ייצירת חיבור WebRTC מוצפן בין כל זוג משתתפים בחדר
- ניהול מספר חיבורי RTCPeerConnection במקביל לכל משתמש
- תמייה בשיחות וידאו, קול ושיתוף מסך
- זמן השהיה (Latency) נמוכים המאפשרים שיחה טבעית
- תפקוד תקין גם בתרחישים של חסימת P2P באמצעות TURN
- זיהוי תוכי פנים בסיסי במכשיר באמצעות אלגוריתמים חישוביים
- אחסון וניהול נתונים ML בסיסי בהתאם לפרויקט
- שמירה על קוד מודולרי, קריא וניתן להרחבה

## **תחום הפרויקט (תיכון)**

### תחומים הכלולים בפרויקט

- תקשורת רשת בזמן אמת Peer-to-Peer ו-WebRTC
- אבטחת מידע והצפנה WebSocket ו-Signaling
- עיבוד וידאו בצד הלקוח Machine Learning
- אלגוריתמים של מוכנה נתונים ובסיס נתונים
- ניהול נתונים ובסיס נתונים

### תחומים שאינם כלולים בפרויקט

- עיבוד מדיה בצד שרת
- הקלטה בשיחות וידאו בשרת SFU / MCU
- שימוש בשירותי מדיה מסווג MCU / SFU
- שימוש ב-Frameworks חיצוניים ללמידה מכונה
- ניתוח Big Data או אימון מודלים כבדים בזמן אמת

## חקר מוצריים

שוק מערכות התקשרות בזמן אמת כולל מגוון רחב של פתרונות מסחריים וקוד פתוח, המאפשרים שיחות וידאו, קול ושיתוף מסך בין משתמשים. בין המערכות הנפוצות ניתן למנות את Zoom, Google Meet, Microsoft Teams ומערכות נוספות, אשר משמשות מיליוני משתמשים ברחבי העולם.

מערכות אלו מציעות חוויה משתמש נוחה ויציבה, אך מבוססות לרוב על תשתיות שרת מורכבות, הכוללות שירות מדיה מרכזיים (MCU/SFU), ניתוח ועיבוד תעבורת מדיה בצד השירות, ולעתים אחסון מידע רגיש בתשתיות צד שלישי. כתוצאה לכך, המשתמש מאבד שליטה מלאה על נתוני המדיה, על אופן הצפנותם ועל מסלול העברתם.

בנוסף, מרבית המערכות המסחריות אינן מאפשרות למפתח להבין או לשלוט עמוק בפרוטוקולים וברכיבי התקשרות הפנימיים, שכן מדובר במערכות סגורות או מושתרות מאחוריו שכבות abstraction רבות.

הפרויקט הנוכחי שואף להציג גישה שונה: מערכת תקשורת בזמן אמת המבוססת על תקנים פתוחים, ללא שירות מדיה מרכזי, ותוך שליטה מלאה בקוד, בארכיטקטורה ובפרוטוקולי התקשרות. הבחירה ב-WebRTC נעשתה מתוך

הבנה כי מדובר בטכנולוגיה סטנדרטית, נתמכת דפסנים, המאפשרת תקשורת מוצפנת בזמן אמת ללא צורך בהתקנת תוכנה חיצונית.

בניגוד לפתרונות קיימים, המערכת תוכננה כך שעיבוד המדיה, לרבות עיבוד וידאו מתקדם מבוסס Machine Learning, יבוצע לצד הלקוב בלבד. גישה זו מגדילה את רמת הפרטיות, מפחיתה תלות בתשתיות חיצונית ומאפשרת שילוב אלגוריתמים חישוביים ייעודיים, אשר אינם זמינים במערכות מסחריות סטנדרטיות.

## חקר פיתוח

פיתוח מערכת תקשורת בזמן אמת מבוססת WebRTC מציב שורה של אתגרים טכנולוגיים והנדסיים, הנוגעים להתחום הרשות והן לתוךם עיבוד המדיה. בפרויקט זה נדרש שילוב של ידע תיאורטי עם מימוש מעשי, תוך התמודדות עם מגבלות מוגנות של תקשורת מבוזרת, אבטחת מידע וביצועים בזמן אמת.

האתגר המרכזי הראשון נוגע לעצם מימוש תקשורת Peer-to-Peer בין מספר משתתפים במקביל. בניגוד לארქיטקטורות מרכזיות, בהן שרת מדיה מנהל את זרימת הוידאו והקול, ארכיטקטורת Mesh מחייבת כל לקוח לניהל מספר חיבור תקשורת בו-זמנית. משמעות הדבר היא ניהול מספר אובייקטי RTCPeerConnection, טיפול באירועי חיבור, ניתוק והთאוששות, וכן סינכרון מצב בין כל המשתתפים בחדר.

אתגר נוסף נוגע להתמודדות עם מגבלות רשת כגון NAT ו-Firewalls. תקשורת Peer-to-Peer ישירה אינה תמיד אפשרית, ולכן נדרש שימוש בפרוטוקולים ייעודיים המאפשרים גליי נתיבי תקשורת חלופיים. בפרויקט זה נבחר שימוש במנגנון ICE, המשלב שירות STUN לצורך גליי כתובת ציבורית ושירות TURN

لتיווך במקרים של חסימה מלאה. שילוב זה מאפשר ייצור חיבור יציב גם בסביבות רשת מורכבות.

נושא אבטחת המידע מהווים אתגר מרהוטי נוסף. מכיוון שהמערכת עוסקת בהעברת מידע בזמן אמיתי, נדרש להבטיח הצפנה מלאה של תעבורת הווידאו והקוול, ללא חשיפת נתונים לשרת המרכזי. WebRTC מספק מגנוני הצפנה מובנים באמצעות DTLS-SRTP, אך מיושם נכון מחייב הבנה של תהליך ייצור המפתחות, החלפת ה-SDP וניהול חיבור מאובטח לכל זוג משתתפים.

מעבר לאתגרי התקשרות, הפרויקט כולל גם אתגרי פיתוח בתחום עיבוד הווידאו ולמידת המכונה. פיתוח מערכת פילטרים למצלה בזמן אמיתי, ללא שימוש בספריות עזר חיצונית, מחייב מיושע עצמאי של אלגוריתמים לעיבוד תמונה, דיהוי התבניות ומעקב אחר תווי פנים. בנוסף, יש לקחת בחשבון מגבלות ביצועים, שכן עיבוד הווידאו מתבצע בזמן אמיתי ובצד הלקוח, במקביל לניהול חיבורו WebRTC.

אתגר משמעותי נוסף הוא בניית מסד נתונים עצמאי עבור מערכת ה-Machine Learning. מסד נתונים זה נדרש לאחסן נתונים אימון, תיוגים ופרמטרים אלגוריתמיים, ולאפשר שליטה מלאה בתהליך הלמידה והבדיקה. הבחירה לבנות מסד נתונים ייעודי, ולא להסתמך על מאגרי נתונים חיצוניים, נועדה להעמיק את ההבנה בתהליכי למידה ולשמור על התאמה מלאה לצרכי הפרויקט.

לבסוף, נדרש תכנון מוקפד של מבנה המערכת והקשר בין רכיביה. שילוב של WebRTC, שירות Signaling, אבטחת מידע ותשתיית Machine Learning מחייב ארכיטקטורה מודולרית, הפרדה ברורה בין שכבות, ויכולת הרחבה עתידית ללא פגיעה ביציבות המערכת. תכנון זה נועד לאפשר הוספה יכולות מתקדמות בהמשך, כגון פילטרים נוספים, אלגוריתמים חכמים יותר ותמייה בתרחישים מורכבים יותר.

## סביבה

פיתוח המערכת בוצע בסביבת עבודה מבוקרת, המאפשרת בדיקה, ניפוי שגיאות והרחבה עתידית של רכיבי המערכת. הסביבה נבחרה כך שתתמוך הן בפיתוח לצד הלקוח והן בפיתוח לצד השירות, תוך התאמת לעובודה עם טכנולוגיות Web ואלגוריתמים חישוביים.

## חומרה

המערכת פותחה והורצתה על מחשב אישי סטנדרטי, הכלול:

- מעבד רב-יליות
- זיכרון RAM המספיק להרצת דפדפן, שרת Python ועיבוד וידאו מקביל
- מצלמת רשת לצורכי בדיקות וידאו בזמן אמת

הפרויקט אינו דורש חומרה ייעודית או מאייצי חישוב (GPU), וזאת מתוך מטרה לשמר על נגימות ולבדק את ביצועי המערכת בתנאים ריאליים של משתמש קצה.

## מערכות הפעלה

הפתרונות בוצע על מערכות הפעלה משפחתיות Windows ו-Linux. המערכת אינה תלויה במערכת הפעלה ספציפית, כל עוד קיימת תמיכה בדפדפן מודרני ובסביבת הריצה ל-`Python`.

## תוכנה וכלי פיתוח

- עורך קוד: Visual Studio Code
- דפדפניים לבדיקות: Google Chrome, Mozilla Firefox
- סביבת שרת: Python 3
- כלים לבדיקה: Chrome DevTools, WebRTC Internals

שימוש בכלים אלו מאפשר ניתוח תעבורת רשת, בדיקת חיבורו `WebRTC`, צפיה ב-`ICE Candidates` ועקב אחר ביצועי המערכת בזמן אמת.

## 0ק'ירת טכנולוגיות הפרויקט

המערכת משלבת מספר טכנולוגיות מרכזיות, כאשר כל אחת מהן נבחרה מתוך שיקול פונקציונלי, לימודי והנדסי.

## WebRTC

תקשורת בזמן אמת של וידאו, קול ונתונים ישרות בין דפדפניים, ללא צורך בתוספים חיצוניים. בפרויקט זה `WebRTC` מהווה את לבת מערכת התקשורות, ומאפשר העברת מדיה מוצפנת בין המשתתפים בארכיטקטורת `Peer-to-Peer`.

### השימוש ב-`WebRTC` אפשר:

- תקשורת בזמן אמת עם Latency נמוך

- הצפנה מובנית של המדיה
- תאימות מלאה לדפינים מודרניים
- עבודה ללא שרת מדיה מרכזי

## צד לקלוח – JavaScript

צד הלקוח פותח בשפת JavaScript, המהווה את הבחירה הטבעית לפיתוח אפליקציות Web אינטראקטיביות. JavaScript מאפשר גישה ישירה ל-Web APIs, ניהול MediaStreams, שליטה במכשיר ובמיקרופון, והציגת משך משתמש דינמי.

**באמצעות JavaScript ממומשים:**

- ייצירת RTCPeerConnection
- ניהול זרמי וידאו וקול
- שיתוף מסר
- טיפול באירועי חיבור וניתוק
- שילוב עתידי של עיבוד וידאו מבוסס Machine Learning

## (Signaling) – הצד שרת Python

צד השרת פותח בשפת Python ומשמש כשרת Signaling בלבד. **השרת אחראי על:**

- ניהול חיבור WebSocket
- החלפת הודעות SDP
- העברת ICE Candidates בין המשתתפים
- ניהול חדרים ומשתמשים

Python נבחרה בזכות פשוטותה, קריונות הקוד והיכולת לפתח שרת קל משקל שאינו מעבד מדיה, אלא מתמקד באיתות בלבד.

## WebSocket - WSS

WebSocket מאפשר תקשורת דו-כיוונית רציפה בין לקוח לשרת. בפרויקט נעשה שימוש ב-WSS (WebSocket Secure), על מנת להבטיח הצפנה של שכבות האיתות.

**באמצעות WebSocket מועברות:**

- הודעות JSON
- נתוני SDP
- ICE Candidates
- הודעות מערכת וניהול

## Machine Learning ועיבוד וידאו

מערכת עיבוד הוידאו מבוססת על אלגוריתמים חישוביים שפותחו מאפס, ללא שימוש בספריות חיצונית לעיבוד תמונה או למידת מכונה. המימוש מתבסס על:

- NumPy לחישובים מתמטיים
- עיבוד מטריצות תמונה
- אלגוריתמים לזיהוי ומעקב אחר תווים פנימיים

גישה זו מאפשרת שליטה מלאה בלוגיקה, הבנת תהליכי הלמידה והכנה לאינטגרציה עתידית עם זרמי וידאו חיצוניים.

## מסד נתונים

מסד הנתונים של המערכת ממוקם לצד השרת ומשמש לצרכים תפעוליים וניהוליים של מערכת ה-WebRTC. מסד נתונים זה מהווה רכיב מרכזי בניהול המערכת ואחראי על אחסון וניהול מידע לוגי, שאינו כולל מדיה חזותית או קולית.

**מסד הנתונים בשרת משמש, בין היתר, ל:**

- ניהול משתמשים (הרשמה, התחברות ואיומות)
- שמירת פרטי חדרים פעילים
- שיור משתמשים לחדרים

- ניהול תפקידי משתמש (Host / Participant)
- שמירת פרטי חברים ולוגיקה פעולה
- תמייה בתהליכי בקרה, ניתוק והתחברות מחדש

הבחירה למקם את מסד הנתונים לצד השירות בלבד, ולמנוע אחסון של מדיה או נתונים חזותיים, העשתה מטור שיקולי אבטחת מידע ופרטיות. כל תעבורת הווידאו והקהל מועברת ישירות בין המשתתפים בתקשרות Peer-to-Peer מוצפנת, מלבד עبور דרך מסד הנתונים או להישמר בשרת.

מבנה זה מאפשר שמירה על ארכיטקטורה קלה ויעילה, תוך הפרדה ברורה בין שכבות האיתות והניהול לבין שכבת המדיה, ומאפשר הרחבנה עתידית של מנגנון ניהול וAIMOTOT ללא פגיעה בפרטיות המשתמשים.

## תיאור מפורט של המערכת

### ארQUITECTURA כללית של המערכת

המערכת נבנתה בארכיטקטורה משולבת מסוג Server–Client (Signalizing) ו-Peer-to-Peer להעברת מדיה, בסגנון Mesh מרובה משתתפים. ארכיטקטורה זו מפרידה באופן ברור בין שכבת ניהול והאיתות לבין שכבת העברת המדיה, ומאפשרת תקשורת מאובטחת ויעילה ללא תלות בשרת מדיה מרכזי.

במבנה זה, שרת מרכזי אחראי אך ורק על תיאום הקשרים בין המשתתפים, בעוד שכל תעבורת הוידאו והcoil מועברת ישירות בין הדפכנים של המשתתפים, בצורה מוצפנת מוקצה לenza.

## רכיבי המערכת המרכזיים

### **צד לקוח (Client – Browser)**

צד הלקוח הוא אפליקציה Web הפעלה בדף, ומפותחת בשפת JavaScript. כל לקוח אחראי על:

- גישה לצלמה ולמיקרופון
- ייצור זרמי מדיה (MediaStreams)
- ייצור חיבורו במקביל (אחד לכל משתמש בחדר)
- ניהול מספר חיבורים במקביל (אחד לכל משתמש בחדר)
- הצפנה והעברת מדיה בזמן אמת
- הציגת מסך משתמש (וידאו, צ'אט, כפתורים)

כל לקוח במערכת פועל כ-peer עצמאי, ומנהל חיבור נפרד מול כל-peer אחר בחדר.

### **צד שרת – Signaling Server**

שרת ה-signaling פתוח בשפת Python ומשמש כשרת תיאום בלבד. תפקידיו הוא:

- קבלת חיבור WebSocket מאבטחים (WSS)
- ניהול חדרים ומשתתפים
- העברת הודעות SDP בין לקוחות
- העברת ICE Candidates בין לקוחות

- **שמירת מידע לוגי וניהולי** במודם הנתונים

השרת אינו מעביר, מעבד או שומר מידע, ובכך מצטמצם משטח התקיפה ונשמרת פרטיות המשתמשים.

#### **מודם נתונים בצד השרת**

מודם הנתונים משמש את שרת ה-Signaling בלבד, ואין מעורב בהעברת מידע. הוא אחראי על:

- ניהול משתמשים והרשאות
- שיור משתמשים לחדרים
- שמירת מצב לוגי של חדרים פעילים
- תמיכה בתהליכי אימות ובקירה

הפרדה זו מבטיחה שמידע רגיש כגון ידאו וקוול לעולם אינם נשמר או עובר דרך השרת.

## ארQUITקטורת Mesh Peer-to-Peer

במערכת זו מיושמת ארQUITקטורת Mesh מלאה, בה כל משתמש יוצר חיבור Peer-to-Peer ישיר מול כל משתמש אחר.

**לדוגמה:**

- בחדר עם 2 משתמשים – חיבור אחד

- בחדר עם 3 משתתפים – כל לקוח מנהל 2 חיבורים
- בחדר עם 4 משתתפים – כל לקוח מנהל 3 חיבורים

**יתרונות גישה זו:**

- פרטיות מלאה – אין שרת מדיה
- Latency נמוך
- הצפנה נפרדת לכל חיבור
- שליטה מלאה בזרימת המדיה

**חסרונות:**

- עומס חישובי גובר עם ריבוי משתתפים
- שימוש מגבר ברוחב פס הצד הלוקוח

בחירה זו נעשתה במודע, בהתאם ליעדי הפרויקט ולדges על לימוד, שליטה  
 ובטחה.

## זרימת המידע במערכת

המערכת עשויה שימוש בשני ערוצי תקשורת נפרדים:

1. **ערוץ איתות (Signaling)**

- מבוסס WebSocket מאובטח (WSS)
- משמש להעברת:
  - SDP Offer / Answer
  - ICE Candidates
  - הودעות ניהול
- עובר דרך שירות Signalign בלבד

## 2. ערוץ מדיה (Media)

- מבוסס WebRTC Peer-to-Peer
- כולל:
  - וידאו
  - קול
  - שיתוף מסר
- מוצפן באמצעות DTLS-SRTP
- אינם עובר דרך השירות

## שילוב תשתיית עיבוד וידאו (Machine Learning)

תשתיית עיבוד הוידאו פועלת לצד הלקוח, חלק ממערכת עיבוד המדיה המקומי. עיבוד זה מתבצע לפני שליחת ה-*MediaStream* ל-peers.

- ניתוח פרטניים מהמצלמה
- הפעלת פילטרים חישוביים בזמן אמת
- שמירה על פרטיות מלאה (ללא שליחת נתונים לשרת)

תשתיית זו מופרדת לוגית ממנגנון התקשרות, אך תוכנה כך שתוכל להשתלב באופן טבעי בזרימת המדיה של WebRTC.

## **פרוטוקולי התקשרות במערכת**

מערכת ה-WebRTC מבוססת על שילוב של מספר פרוטוקולי תקשורת, אשר פועלם יחד לצורך ייצור חיבור מאובטח, יציב ובזמן אמת בין המשתתפים. בפרויקט זה נעשה שימוש ב프וטוקולים אלו בצורה ישירה ומודעת, תוך הבנה של תפקידיו של כל רכיב בתהליך.

**הפרוטוקולים במערכת נחלקים לשתי קבוצות עיקריות:**

- פרוטוקולי אינטואטיביים וניהול חיבור
- פרוטוקולי העברת מדיה והצפנה

## WebRTC – עקרונות פעולה כלליים

WebRTC אינו פרוטוקול בודד, אלא סט תקנים וממשק API המאפשרים תקשורת בזמן אמת בין דפנינים. תפקידו העיקרי הוא לאפשר העברת וידאו, קול ונתונים בצורה ישירה בין לקוחות, ללא צורך בשרת מדיה מרכזי.

**בבסיס WebRTC עומדים שלושה רכיבים עיקריים:**

- Peer-to-Peer – ניהול חיבור RTCPeerConnection
- – ניהול זרמי מדיה (וידאו, קול, מסך) MediaStream
- – העברת נתונים (לא בשימוש מרכזי בפרויקט זה) DataChannel

### RTCPeerConnection

RTCPeerConnection הוא האובייקט המרכזי האחראי על ייצור וניהול חיבור Peer-to-Peer בין שני משתתפים. בפרויקט זה, כל לקוח יוצר מופע RTCPeerConnection נפרד עבור כל משתתף אחר בחדר, בהתאם לארQUITטורת Mesh.

**תפקיד RTCPeerConnection כללים:**

- ניהול משא ומתן על תצורת החיבור
- החלפת פרטי מדיה ויכולות קידוד
- טיפול בהצפנה ובאיומות
- ניהול חיבור ICE

### SDP – Session Description Protocol

SDP הוא פורמט טקסטואלי המתאר את מאפייני החיבור בין שני משתתפים. במסגרת WebRTC, SDP משמש לתיאור:

- סוג מדיה (וידאו, קול)

- קודקים ותמכים
- פרטי הצפנה
- פרטי רשות ראשוניים

תהליך החיבור מתבצע באמצעות מנגנון **Offer / Answer**

1. אחד המשתתפים ייצור SDP Offer
2. ה-SDP Offer נשלח למשתתף השני דרך שירות ה-Signaling
3. המשתתף השני מחייב SDP Answer
4. לאחר השלמת התהליך, החיבור יכול להתחיל

## ICE – Interactive Connectivity Establishment

ICE הוא מנגנון המשמש למציאת נתיב התקשרות הטוב ביותר בין שני משתתפים. מכיוון שמשתתפים רבים נמצאים מאחורי NAT או Firewall, חיבור ישיר אינו תמיד אפשרי.

**ICE אחראי על:**

- איסוף כתובות רשות אפשריות (Candidates)
- בדיקת זמינות כל נתיב
- בחירת הנתיב היעיל ביותר

## STUN – Session Traversal Utilities for NAT

שירות STUN מאפשר ללקוח לגלוות את כתובתו הציבורית כפי שהיא נראית מחוץ לרשות המקומית. מידע זה משמש כחלק מהתהליך ICE, ומאפשר ניסיון חיבור ישיר בין שני לקוחות.

STUN אינו מעביר מדיה ואינו מתוויר את התקשרות, אלא משמש לגילוי כתובות בלבד.

## TURN – Traversal Using Relays around NAT

במקרים בהם חיבור ישיר אינו אפשרי (למשל עקב Firewall מחמייר), נעשה שימוש בשירות TURN. שירות זה משמש כמתוור, דרכו מועברת תüberת המדיה.

בפרויקט זה נעשה שימוש בשרת TURN חיצוני רק כמוצא אחרון, כאשר חיבור Peer-to-Peer ישיר אינו מתאפשר.

## WebSocket Secure (WSS) – Signaling

שרת **the signaling** עושים שימוש בפרוטוקול **WebSocket - WSS** לצורך החלפת הודעות בין הלקוחות. ערך זה משמש להעברת:

- הודעות SDP
- ICE Candidates
- הודעות ניהול וחודרים

כל התקשרות בערך זה מוצפנת באמצעות TLS, אך אינה כוללת מדיה.

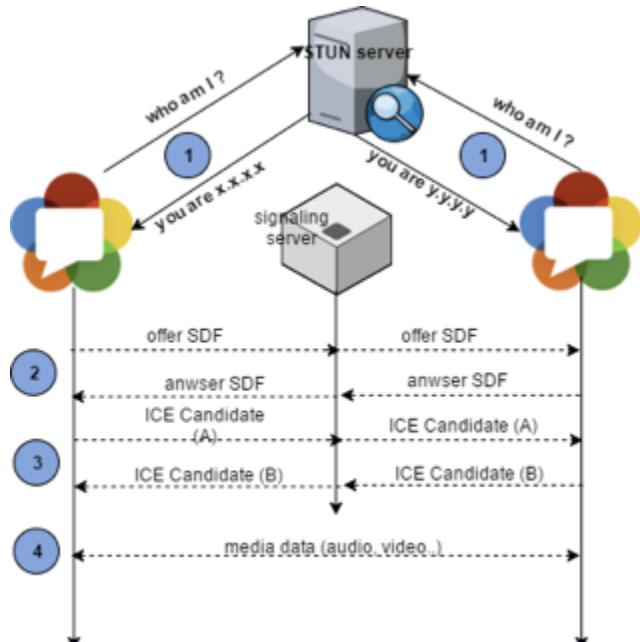
## הפרדה בין ערוצי תקשורת

אחד העקרונות המרכזיים במערכת הוא ההפרדה המלאה בין:

- ערוץ האיתות (דרך השירות)
- ערוץ המדיה (Peer-to-Peer)

הפרדה זו:

- מגדילה פרטיות
- מצמצמת עומס שרת
- משפרת אבטחה
- מאפשרת Scalability מבוקרת



## שילוב עיבוד וידאו מבוסס Machine Learning בזרימת התקשורת

לצד פרוטוקול התקשורת, המערכת תוכנונה כך שתאפשר שילוב של עיבוד וידאו חישובי לצד הלקוח, כחלק מGINOR העברת המדיה של WebRTC. עיבוד זה אינו מהוות חלק מפרוטוקולי התקשורת עצם, אלא פועל בשכבה לוגית נפרדת, לפני שליחת ה-*MediaStream* ל-peers אחרים.

במהלך פעולה המערכת, פרימיום המתקבלים מהמצלמה המקומית יכולים לעבור עיבוד מקדים באמצעות אלגוריתמים חישוביים, כגון זיהוי ומעקב אחר תווים פנים, טרם קידודם והעברתם באמצעות WebRTC. תהליך זה מתבצע באופן מקומי בדפדפן, ואני משפיע על תהליכי האיתות (Signaling), על מנגנון ICE או על תצורת החיבור בין המשתתפים.

הפרדה זו מאפשרת:

- שמירה על תקינות פרוטוקול התקשורת
- מניעת תלות בין שכבות התקשורת לשכבות העיבוד
- שמירה על פרטיות מלאה, שכן נתונים הוויידאו המועבדים אינם מועברים לשרת
- אפשרות להרחבת עתידית של אלגוריתמי העיבוד ללא שינוי במנגנון החיבור

פירוט מלא של תשתיות ה-*Networking*, האלגוריתמים והשילוב העתידי שליהם במערכת יוצג בפרק ייודי בסופו.

## סיכום הפרק

שילוב ה프וטוקולים TURN, WebRTC, SDP, ICE, STUN ו-WebSocket מאפשר למערכת ליצור חיבור תקשורת בזמן אמת, מאובטחים ועמידים למוגבלות רשת. הבנת תפקידו של כל פרוטוקול והאינטראקטיה ביניהם מהוות בסיס הכרחי למימוש נכון של מערכת Peer-to-Peer מרובה משתתפים.

## אבטחת מידע והצפנה במערכת

אבטחת מידע מהויה מרכיב מרכזי בתכנון ובשימוש המערכת, לאור העובדה שהיא עוסקת בהעברת מידע בזמן אמת, הכוללת וידאו וקול רגילים. בפרויקט זה הושם דגש על הצפנה מלאה, הפרדת שכבות תקשורת וצמצום חשיפת מידע, תוך שימוש במנגנון האבטחה המובנים בתקני WebRTC ובפרוטוקול תקשורת מאובטחים.

עקרון מנחה בתכנון המערכת הוא שמידע חזותי וקווי אינו נשמר ואין מעובד直达 השרת, אלא מועבר ישירות בין המשתתפים בתקשורת Peer-to-Peer מוצפנת.

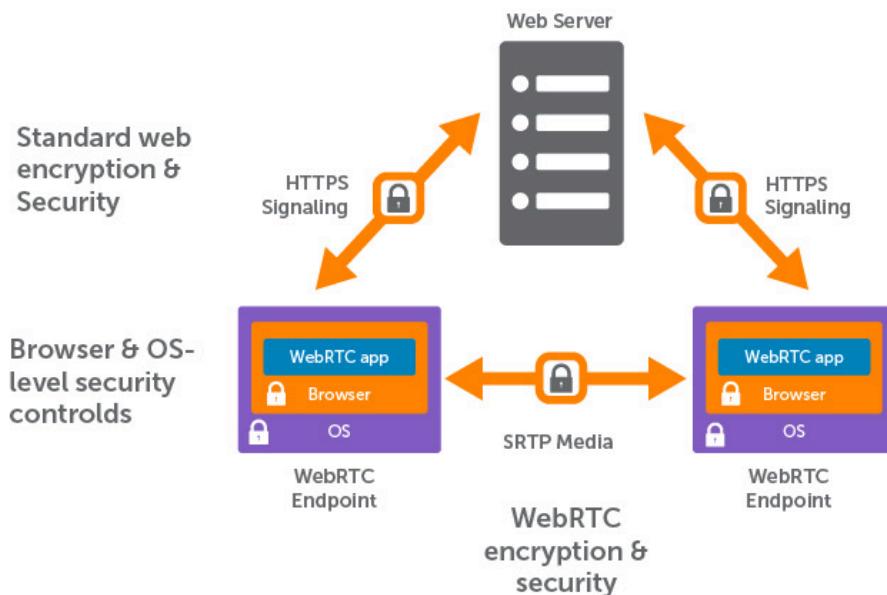
### אבטחת שכבת האיתות (Signaling)

שכבה האיתות במערכת מבוססת על פרוטוקול WebSocket Secure (WSS), הפעול מעל TLS. שכבה זו אחראית על העברת הודעות ניהול, ICE, SDP ו-*Candidates* בין הלקוחות, אך אינה כוללת העברת מידע.

#### אמצעי האבטחה בשכבת האיתות כוללים:

- שימוש ב-TLS להצפנה תעבורת WebSocket (Man-in-the-Middle)
- מניעת האזנה או שינוי הודעות איתות
- העברת מידע בפורמט JSON מבוקר
- הפרדה בין מידע ניהול לבין נתונים מידע

שימוש ב-WSS מבטיח כי גם שכבת התיאום בין המשתתפים מגנת מפני יירוט או שינוי זמני.



## הצפנה מדיה – DTLS-SRTP

העברת המדיה (וידאו וקול) במערכת מתבצעת באמצעות **WebRTC Peer-to-Peer**:  
טור שימוש במנגנון הצפנה מובנים:

- DTLS - Datagram Transport Layer Security – משמש לאימוט  
והחלפת מפתחות הצפנה בין ה-Peers
- SRTP - Secure Real-time Transport Protocol – משמש להצנת  
תעborת המדיה עצמה

**תהליך זה מבטיח:**

- הצפנה מקצה לקצה (End-to-End Encryption)
- יצרת מפתחות הצפנה נפרדים לכל חיבור Peer-to-Peer
- מניעת גישה לנוטוני המדיה מצד השרת

שרת הי-סיגנליング Signaling אינו מחזיק במפתחות ההצפנה ואין לו מסוגל לפענן את המדיה.

## הפרדת תפקידיים ומצוון משטח התקיפה

במערכת קיימת הפרדה ברורה בין רכיבי המערכת:

- השרת אחראי על איתות וניהול בלבד
- הלוקוחות אחראים על ייצור חיבורים והעברת מדיה
- מסד הנתונים מכיל מידע לוגי בלבד, ללא מדיה

**הפרדה זו:**

- מצמצמת את משטח התקיפה
- מפחיתה סיכון דליפת מידע
- מאפשרת תחזוקה מאובטחת ופושטה יותר

## אבטחת תקשורת מול שרת STUN ו-TURN

שרת STUN ו-TURN משמשים לצורך גילוי נתיבי תקשורת וה坦מודדות עם חסימות רשות. שרתים אלו אינם מעבדים או שומרים מידע באופן קבוע, ותפקידם מוגבל לティוּר חיבור בלבד במקרים חריגים.

גם כאשר נעשה שימוש ב-**TURN**:

- המדיה מועברת בצורה מוצפנת
- השירות אינו מסוגל לענוח את התוכן
- השימוש ב-**TURN** מתבצע רק כאשר אין אפשרות לחבר ישיר

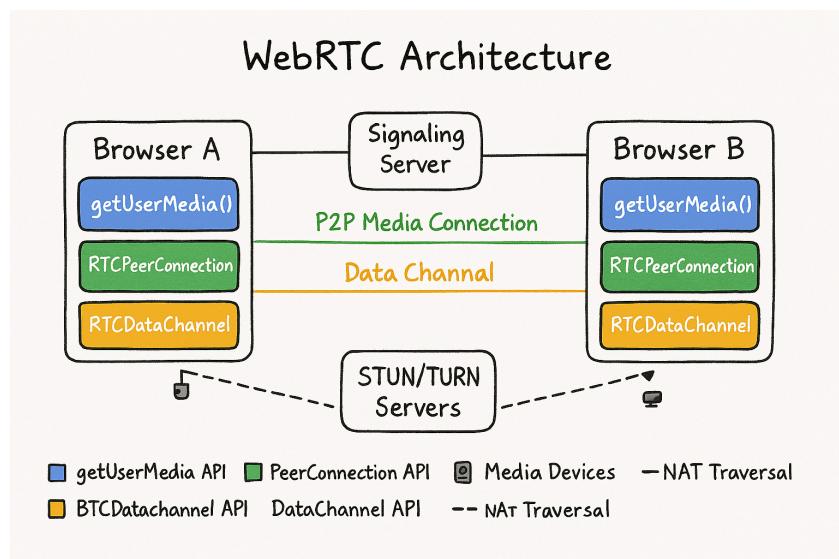
## פרטיות משתמשים

שמירה על פרטיות המשתמש הייתה שיקול מרכזי בתכנון המערכת. בהתאם לכך:

- לא מתבצע רישום או הקלה של שיחות
  - לא נשמרים נתונים יידאו או קול
  - עיבוד הוידאו (כולל פילטרים חישוביים) מתבצע מקומיות לצד הלקוח בלבד
- גישה זו מבטיחה שליטה מלאה של המשתמש במידע האישית שלו ומונעת חשיפה מיותרת.

## סיכום אבטחת המידע

מערכת זו עשויה שימוש במנגנון אבטחה תקניים ומוכחים, תוך שימוש בצפנה מקצועית, הפרדת שכבות והימנעות מאחסון מידע בשרת. תכנון זה מאפשר ייצירת מערכת תקשורת בזמן אמת מאובטחת, פרטית ועמידה בפני איום נפוצים ברשת.



## אלגוריתמים מרכזיים במערכת

מערכת ה-RTC המפותחת בפרויקט נשענת על מספר אלגוריתמים מרכזיים, אשר אחראים על ייצרת החיבורים, ניהול המשתתפים, העברת המידע וה坦מודות עם תרחישי קצה כגן ניוטק והתחברות חדש. אלגוריתמים אלו אינם מتبוססים על מנגנון יחיד, אלא על שילוב של תהליכי סינכרוניים ו-א-סינכרוניים הפעילים יחד בזמן אמת.

### אלגוריתם ייצרת חדר וניהול תפקידיים

בעת ייצרת חדר חדש במערכת, מתבצע תהליך לוגי הכלול:

1. משתמש יוזם בקשה לייצרת חדר
2. שרת ה-Signaling מקצה מזהה חדר ייחודי
3. המשתמש מוגדר אוטומטית כ-Host
4. פרטי החדר נשמרים במסד הנתונים לצד השרת
5. החדר מוקן לקבלת משתתפים נוספים

תפקיד ה-Host כולל הרשות ניהול, כגון שליטה על משתתפים, ניוטק משתמשים וניהול מצב החדר.

### אלגוריתם הצטרפות משתמש לחדר קיים

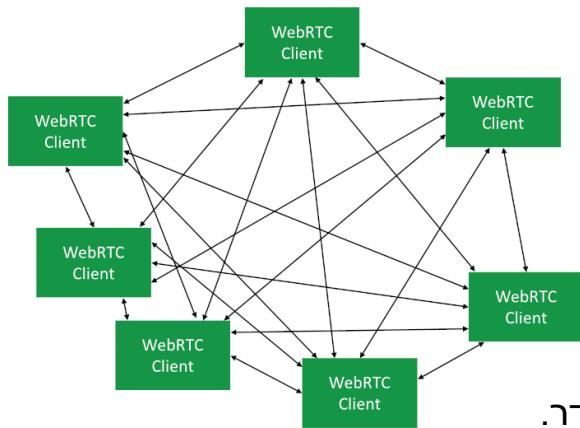
כאשר משתמש חדש מצטרף לחדר פעיל:

1. המשתמש יוצר חיבור WebSocket לשרת ה-Signaling
2. מתבצע אימות ורשהה
3. השרת שולח למשתמש את רשימת המשתתפים הקיימים בחדר
4. המשתמש מתחילה תהליך ייצרת חיבור Peer-to-Peer מול כל משתמש קיים
5. המשתתפים הקיימים מתעדכנים בהצטרפות המשתמש החדש

אלגוריתם זה מבטיח שככל המשתתפים ישארו מסונכרים לגבי מצב החדר.

## אלגוריתם ייצור Mesh Peer-to-Peer

בלב המערכת נמצא אלגוריתם ייצור רשת Mesh, בו כל משתתף יוצר חיבור RTCPeerConnection נפרד מול כל משתתף אחר.



שלבי האלגוריתם:

1. ייצור RTCPeerConnection חדש SDP Offer
2. ייצירת ה-SDP Offer דרך שרת ה-Signaling
3. שליחת ה-SDP Offer דרך שרת ה-Signaling השני
4. קבלת SDP Answer מה-Peer השני
5. התחלת החלפת ICE Candidates
6. מעבר למצב חיבור פעיל

תהליך זה חוזר על עצמו עבור כל זוג משתתפים בחדר.

## אלגוריתם החלפת ICE Candidates

במהלך ייצור החיבור ולאחריו, כל משתתף:

- אוסף ICE Candidates מקומיים
- שולח אותם ל-Peer השני דרך שרת ה-Signaling
- בודק זמינות של כל נתיב תקשורת
- בוחר את הנתיב האופטימלי

אלגוריתם זה מאפשר התאמת דינמית לתנאי הרשות וմבטיח חיבור יציב ככל האפשר.

## אלגוריתם ניהול מדיה בזמן אמת

לאחר ייצור החיבור, מתבצע ניהול שוטף של זרמי המדיה:

- קליטת וידאו וקול מהמצלמה והמיקרופון
- קידוד המדיה בהתאם לכ יכולות ה-Peer
- שליחת המדיה דרך חיבור WebRTC
- הצגת המדיה המתקבלת במשתמש

אלגוריתם זה פועל ברכזיות ודורש סyncron מדויק בין חיבורים רבים.

## אלגוריתם ניתוק והטאוששות מחיבור שנפל

במקרה של ניתוק משתף או נפילת חיבור:

1. זיהוי מצב disconnected / failed
2. עדכון המשתתפים האחרים
3. סגירת חיבור RTCPeerConnection רלוונטיים
4. ניסיון ICE Restart במידת הצורך
5. חיבור מחדש או הסרת המשתתף מהחדר

מנגן זה מאפשר המשך עבודה יציב גם בתנאי רשת משתנים.

## אלגוריתם שילוב עיבוד וידאו לצד הלקוח

בצד הלקוח, לפני שליחת MediaStream ל-peers אחרים, ניתן לשלב שלב עיבוד מוקדים:

1. קליטת פריים מהמצלמה
2. עיבוד חישובי (פילטרים, ניתוח תווים)
3. ייצור זרם וידאו מעובד
4. שליחת הזרם ל-WebRTC

אלגוריתם זה מופרד לוגית משכבות התקשרות ואינו משנה על מנגןוני החיבור עצמם.

## סיכום האלגורitmim

שילוב האלגוריתמים במערכת מאפשר יצרת פתרון תקשורת בזמן אמת, יציב, מאובטח ומודולרי. האלגוריתמים תוכנו כך שייפעלו יחד בצורה ייעילה, תוך שמירה על הפרדה בין שכבות, פרטיות המשתמש ויכולת הרחבה עתידית.

## תיאור פונקציוני של המערכת

### ניהול משתמשים

המערכת כוללת מנגנון ניהול משתמשים מלא, המאפשר הרשמה, התחברות ואימوت משתמשים בצורה מאובטחת. מנגנון זה נדרש לוודא כי רק משתמשים מורשים יכולים ליצור חדרים, להצטרף לשיחות ולבצע פעולות ניהול.

כל משתמש במערכת מזוהה באמצעות כתובת דואר אלקטרוני וסיסמה, המשמשים כאמצעי ההזדהות ייחודי.

### תהליך הרשמה (Sign Up)

בעת הרשמה למערכת, מתבצע התהליך הבא:

1. המשתמש מזין כתובת דואר אלקטרוני וסיסמה
2. הנתונים נשלחים לשרת בצורה מאובטחת
3. השרת שומר את פרטי המשתמש בסיסד הנתונים
4. נשלחת הודעה דואר אלקטרוני לאיומות החשבון
5. המשתמש נדרש לאשר את החשבון באמצעות קישור ייעודי

רק לאחר השלמת תהליך האימות, ניתן להתחבר למערכת ולהשתמש בשירותיה.

מנגנון זה נדרש:

- למנוע יצירת משתמשים מזויפים
  - להבטיח תקינות כתובות דוא"ל
- לחזק את רמת האבטחה הכללית של המערכת

### תהליך התחברות (Login)

לאחר הרשמה ואימות מוצלח, המשתמש יכול להתחבר למערכת:

1. הזנת כתובת דוא"ל וסיסמה
2. אימות הנתונים מול בסיס הנתונים
3. פתיחת Session מאובטח

4. מתן גישה לפונקציות המערכת בהתאם להרשאות המשתמש  
במקרה של כשל בהתחברות, תוצג הודעה שגיאת מתאימה למשתמש.

## ניהול חדרים

המערכת מבוססת על קונספט של חדרים (Rooms), כאשר כל חדר מייצג שמן  
תקשורות עצמאי.

### **יצירת חדר**

- משתמש מחובר יכול ליצור חדר חדש
- השירות מזמין מזהה חדר ייחודי
- יוצר החדר מוגדר אוטומטית כ-Host
- החדר נרשם כמופעל וזמין להצטרפות

### **הצטרפות לחדר**

- משתמשים אחרים יכולים הצטרף לחדר באמצעות מזהה או קישור
- מתבצע אימוץ הרשות
- לאחר הצטרפות מתחילה תהליך ייצור חיבור WebRTC מול שאר  
המשתתפים

## הרשאות ותפקידים

במערכת קיימים תפקידים משתמש שונים:

### **Host**

- ייצור חדר
- ניהול משתמשים
- שליטה על מצב החדר

### **Participant**

- הצטרפות לחדר
- שליחת וקבלת מידע וקול
- שימוש בצלט ושיתוף מסך

חלוקת זו מאפשרת שליטה ברורה על תפקידים השונים במהלך השיחה.

## **אבטחת מידע בתהליכי משתמשים**

כל תהליכי ההרשמה, ההתחברות וניהול החדרים מתבצעים דרך ערכז תקשורת מאובטחים (WSS / HTTPS). סיסמות אינן נשמרות בצורה גלויה, והמערכת אינה חשופת פרטיים רגיסטרים למשתמשים אחרים.

### **סיכום התיאור הפונקציונלי**

מנגן ניהול המשתמשים והחדרים מהוועה שכבת בסיס קריטית לפועלות המערכת. שילוב של אימות משתמשים, הרשאות וחלוקת תפקידים מאפשר שימוש מאובטח, מבוקר ומוסודר במערכת התקשרות בזמן אמת.

## מסך משתמש (User Interface)

מסך המשתמש של המערכת תוכנן כך שיהיה ברור, אינטואיטיבי ונוח לשימוש, גם עבור משתמשים שאינם בעלי רקע טכני. עיצוב המשתמש נועד לאפשר גישה מהירה לפונקציות המרכזיות של המערכת, תוך שמירה על חווית שימוש רציפה בזמןשיח.

המסך פועל מ透过 דפדן אינטרנט, ואינו דורש התקינה של תוכנה חיצונית. כל הפעולות מתבצעות בזמן אמת, תוך סyncron מלא בין המשתתפים בחדר.

### מסך כניסה (Login)

מסך הכניסה הוא נקודת ההתחלה של המערכת. **במסך זה המשתמש מתבקש להזין:**

- כתובת דואר אלקטרוני
- סיסמה

לאחר הזנת הפרטים ולחיצה על כפתור התחברות, מתבצעAIMOT מול השרת. במקרה של פרטיים שגויים, תוצג הודעה שגיאה מתאימה.

מסך זה כולל גם קישור למסך ההרשמה עבור משתמשים חדשים.

### מסך הרשמה (Sign Up)

מסך ההרשמה מאפשר יצירת חשבון חדש במערכת. **המשתמש נדרש:**

- כתובת דואר אלקטרוני
- סיסמה
- אישור סיסמה

לאחר השלמת תהליך ההרשמה, נשלחת הודעה דואר אלקטרוני לAIMOT החשבון. עד לאישור החשבון, לא ניתן להצטרף למערכת.

מנגנון זה תורם לאבטחת המערכת ולמניעת שימוש לרעה.

## מסך ראשי / לובי

לאחר התחברות מוצלחת, המשתמש מועבר למסך הראשי (Lobby), ממנו ניתן:

- לייצור חדר חדש
- להצטרף לחדר קיימ באמצעות מזזה או קישור
- לצפות במידע בסיסי על המשתמש המחבר

מסך זה מהוות מעבר בין שלב האימוח לבין שלב השיחה עצמה.

## מסך חדר (Room View)

מסך החדר הוא המסך המרכזי במערכת, בו מתבצעת השיחה בזמן אמת. מסך זה כולל:

### **תצוגת ידאו**

- תצוגת Grid של משתתפי החדר
- הציגת יידאו ח'י מכל משתמש
- התאמת גודל הווידאו למספר המשתתפים

### **בקרי מדיה**

- הפעלה / השתקה של מיקרופון
- הפעלה / כיבוי של מצלמה
- שיתוף מסך
- יציאה מהחדר

הבקרים זמינים בכל עת ומאפשרים שליטה מיידית במהלך השיחה.

## צ'אט טקסטואלי

לצד שיחת הוידאו, קיים צ'אט טקסטואלי בזמן אמת, המאפשר:

- שליחת הודעות בין המשתתפים

- הצגת היסטורית הודיעות במהלך השיחה
- תקשורת שקטה במקביל לשיחה הקולית

הצ'אט עוזה שימוש בערוצ האיות ואינו משפיע על תעבורת המדיה.

## מסך ניהול – Host

למשתמש המוגדר כ-Host מוצג ממוקן ניהול נוסף, המכול:

- רשימת משתתפים מחוברים
- זיהוי תפקיד המשתתפים
- אפשרות לניטוק משתמשים
- שליטה בהרשאות בסיסיות

ממשק זה מאפשר ניהול מסודר של השיחה ושמירה על סדר ותקינות.

## חוויית משתמש ושיתופי פעולה – עיצוב

בעת תכנון הממשק הושם דגש על:

- מינימליزم ופשטות
- תגובתיות (Responsive Design)
- הצגת מידע ברור ולא עמוס
- זמינות פונקציות קרייטיות בכל שלב

הממשק תוכנן כך שיתאים גם לשימוש במסכים שונים וברוחולציות שונות.

## **סיכום ממשק המשתמש**

מסך המשתמש מהו חלק מרכזי בהצלחת המערכת, ומאפשר למשתמשים לנצל את יכולות התקשרות והניהול בצורה נוחה, ברורה ובוטחה. שילוב של ממשק אינטואיטיבי עם תשתיות טכנולוגיות מתקדמות יוצר מערכת שימושית, יעילה ונגישה.

## – מפרט תכנון מפורט Design Detailed

פרק זה מציג פירוט מעמיק של מבנה הפרויקט, חלוקת הקוד למודולים והאחריות של כל רכיב במערכת. מטרת הפרק היא להציג את מבנה המערכת ברמת הקוד, ולהדגים כיצד הרכיבים השונים בפרקם הקודם מושגים בפועל.

**המערכת במבנה בגישה מודולרית, תוך הפרדה ברורה בין:**

- צד לקוח (Client)
- צד שרת (Signaling Server)
- מסד נתונים
- רכיבי תשתיות וצורה

### מבנה כללי של הפרויקט

הפרויקט מחולק למספר תיקיות וקבצים עיקריים, כאשר כל רכיב אחראי על שכבה לוגית שונה במערכת:

- קבצי הצד לקוח (HTML / CSS / JavaScript)
- קבצי הצד שרת (Python)
- קבצי צורה
- רכיבי מסד נתונים

חלוקת זו מאפשרת תחזוקה נוחה, קריאות קוד ורחבת עתידית של המערכת.

## צד לקוח – Client

צד הלקוח הוא אפליקציה Web הפועלת בדף, אחראית על כל הלוגיקה הקשורה לחוויות המשתמש וلتקשורת בזמן אמיתי.

### **קבצי HTML**

קובצי HTML אחראים על:

- מבנה המסכים
- הצגת רכיבי ממושך
- שילוב קבצי JavaScript ו-CSS

קובצים אלו מגדירים את שלד הממושך בלבד, בעוד הלוגיקה ממומשת בקובצי JavaScript.

### **CSS**

קובצי העיצוב אחראים על:

- פריסת מסכים
- עיצוב תצוגת וידאו
- התאמה למסכים שונים (Responsive Design)

העיצוב נשמר פשוט וברור, במטרה להבהיר את פונקציונליות המערכת.

### **קובצי JavaScript – לוגיקת WebRTC**

קובצי JavaScript מהווים את לבת הצד לקוח, וכוללים:

- ייצירת RTCPeerConnection
- ניהול MediaStreams
- טיפול ב-SDP Offer / Answer
- החלפת ICE Candidates
- ניהול חיבור WebSocket
- עדכון ממושך המשתמש בזמן אמיתי

הולוגיקה מחולקת לפונקציות ברורות, כאשר כל פונקציה אחראית על שלב אחר בתהליך החיבור.

## ניהול חיבורים מרובים

בשל ארכיטקטורת Mesh, צד הלוקו מנהל מספר חיבוריו `RTCPeerConnection` במקביל. לשם כך נעשה שימוש במבנה ותוניהם הממפה בין מזהי משתמשים לחיבוריו `PeerConnection` פעילים.

גישה זו מאפשרת:

- ייצירה והסרה דינמית של חיבורים
- ניהול חיבורים לפי משתתף
- טיפול יעיל בנזקים והתחרויות מחדש

## צד שרת – Signaling Server

שרת ה-`Signaling` ממומש בשפת Python, ואחראי על תיאום החיבורים בין הלוקוחות.

### **תפקידים מרכזיים של השרת**

- קבלת חיבורו `WebSocket`
- ניהול חדרים ומשתתפים
- העברת הודעות SDP
- העברת ICE Candidates
- אינטראקציה עם מסד הנתונים

השרת אינו מעבד מדיה ואינו שומר נתונים ידאו או קול.

### **מבנה קוד השרת**

קוד השרת מחולק למודולים לוגיים, כגון:

- מודול חיבורו `WebSocket`
- מודול ניהול חדרים
- מודול ניהול משתמשים
- מודול גישה למסד הנתונים

חלוקת זו מאפשרת תחזוקה נוחה והרחבת עתידית של רכיבי המערכת.

## מודד נתונים

מודד הנתונים משמש לאחסון מידע לוגי בלבד, ואינו כולל מדיה. המידע הנשמר כולל:

- פרטי משתמשים
- סטטוס אימות
- שיווק משתמשים לחדרים
- תפקיד משתמשים (Host / Participant)

גישה למודד הנתונים מתבצעת דרך שכבת גישה ייעודית, המונעת גישה ישירה מהלוגיקה העסקית.

## טיפול באירועים (Event Handling)

המערכת עשויה שימוש באירועים (Events) לצורך:

- חיבור וניתוק משתמשים
- קבלת הודעות WebSocket
- שינוי מצב חיבור WebRTC
- עדכון ממוקם המשתמש

גישה מבוססת אירועים מאפשרת תגובתיות גבוהה והטאמה לשינויים בזמן אמת.

## שילוב של עיבוד וידאו (Machine Learning)

תשתיית עיבוד הווידאו ממוקמת לצד הלקוח, ומופרדת מהלוגיקה של WebRTC. העיבוד מתבצע על פרויומים לפני שליחתם ל-MediaStream.

תכונן זה מאפשר:

- הוספת פילטרים חישוביים
- הרחבת אלגוריתמים

- שמירה על קוד נקי ומודולרי  
פירוט האלגוריתמים עצם יוצג בפרק "יעוד".

## **סיכום Design Detailed**

המפורט המפורט מדגים כיצד הרעיוןות הארכיטקטוניים של המערכת ממומשים בפועל בקוד. חלוקה מודולרית, הפרדת אחריות וניהול חיבורים מבקר מאפשרים מערכת יציבה, מאובטחת וניתנת להרחבה.

## תשתיית Machine Learning – עיצוב ואלגוריתמים

פרק זה מתאר את תשתיית עיבוד הווידאו והאלגוריתמים החישוביים שפותחו כחלק מהפרויקט, לצורך הוספת פילטרים חכמים למכשיר בזמן אמת. תשתיית זו תוכננה לפעול לצד הלקוח בלבד, ללא תלות בשרת ולא שימוש בספריות למידת מכונה או עיבוד תמונה חיצונית.

מטרת הפרק היא להציג את הגישה ההנדסית שנבחרה, את מבנה האלגוריתמים ואת אופן השילוב העתידי שלהם במערכת ה-WebRTC.

### עקרונות תכנון כלליים

תשתיית ה-ML בפרויקט פותחה בהתאם לעקרונות הבאים:

- עיבוד מקומי בלבד לצד הלקוח
- הימנעות מהעברת נתונים חזותיים לשרת
- שימוש אלגוריתמי עצמאי (Low-Level)
- שימוש מינימלי בספריות עזר – NumPy בלבד
- התאמה לעיבוד בזמן אמת (Real-Time)

עקרונות אלו נבחרו מtors רצון להבין לעומק את תהליכי עיבוד התמונה והחישוב, ולא להסתמך על מימושים מוכנים.

### מיקום תשתיית ה-ML בארכיטקטורה

תשתיית ה-ML ממוקמת לצד הלקוח, חלק מצינור עיבוד הווידאו המקומי. האלגוריתמים פועלים על פרימיטים המתקבלים ישירות מהמכשיר, לפני קידוד המדיה והעברתה באמצעות WebRTC.

מבנה זה מאפשר:

- עיבוד וידאו לפני יצירת MediaStream
- שליחת וידאו מעובד ל-peers אחרים
- שמירה על הפרדה מלאה בין שכבת התקשורת לשכבה העיבוד

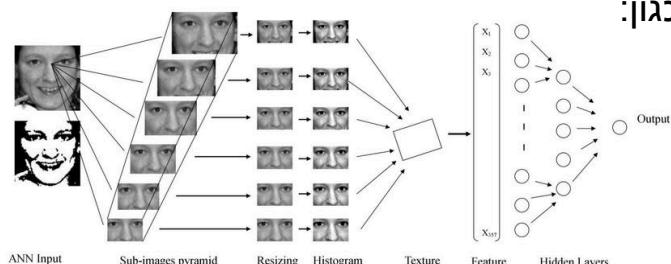
האלגוריתמים אינם משפיעים על תהליך ה-Signaling, על ICE או על יצירת החיבורים עצם.

## קליטת פרימרים ועיבוד בסיסי

תהליך עיבוד הוויידאו מתחילה בקלטת פרימרים מהמצלמה המקומית. כל פרימר מיוצג כמטריצה דו-ממדית או תלת-ממדית של ערכי פיקסלים.

באמצעות NumPy מתבצעות פעולות חישוביות כגון:

- המרת צבעים
- נרמול ערכים
- חיתוך אזורי עניין (ROI)
- חישובים סטטיסטיים על אזורים בתמונה



פעולות אלו מהוות בסיס לאלגוריתמים מתקדמים יותר.

## זיהוי ומעקב אחר תווי פנים – גישה אלגוריתמית

המערכת כוללת תשתית ראשונית לזיהוי ומעקב אחר תווי פנים, הממומנת בגישה חישובית ישירה. הגישה אינה מבוססת על רשתות נירוניים מוכנות או מודלים מאומנים מראש, אלא על ניתוח תבניות בתמונה.

האלגוריתם מבצע:

- זיהוי אזורי עניין בתמונה
- ניתוח שינויי בין פרימרים עוקבים
- מעקב אחר מיקום יחסי של תווי פנים

גישה זו מאפשרת הבנה מלאה של הלוגיקה, במחיר דיווק נמוך יותר לעומת מודלים מתקדמים – בחירה מודעת בהתאם לעדיף הפרויקט.

## פילטרים חישוביים בזמן אמת

על בסיס ניתוח התמונה ניתן להפעיל פילטרים חישוביים שונים, כגון:

- הדגשת אזורים מסוימים בתמונה
- הוספת שכבות גרפיות
- שינוי מאפיינים חזותיים של אזור פנים

פילטרים אלו פועלים בזמן אמת, ולכן האלגוריתמים תוכננו להיות קלים חישובית ולהימנע מפעולות כבדות.

## שיקולי ביצועים

עיבוד וידאו בזמן אמת מחייב התייחסות לביצועים:

- שמירה על קצב פריימרים יציב
- הימנע מלולאות כבדות
- שימוש בפעולות וקטוריות של NumPy
- עיבוד חלק של פריימרים במידה הצורך

שיקולים אלו מאפשרים הפעלת האלגוריתמים במקביל לניהול חיבור WebRTC.

## מגבלות המערכת

לתשתית ה-Machine Learning בפרויקט קיימות מגבלות ידועות:

- דיק נמוך יחסית למודלים מבוססי Deep Learning
- ריגשות לתנאי תאורה
- התאמה לסceanות פשוטות בלבד

מגבלות אלו נלקחו בחשבון כחלק מתהליכי הפיתוח, והוגדרו כחלק מההתקום ההנדסי של הפרויקט.

## הרחבות עתידיות

התשתיית תוכנונה כך שתאפשר בעתיד:

- שילוב מודלים מתקדמיים יותר
- שיפור אלגוריתמי דוחי
- אופטימיזציה נוספת לביצועים
- שילוב עמוק יותר עם MediaStream של WebRTC

כל זאת מבלי לשנות את מבנה הארכיטקטורה הקיימת.

## סיכום תשתיית ה-Machine Learning

תשתיית ה-Machine Learning בפרויקט מדגימה גישה אלגוריתמית בסיסית לעיבוד וידאו בזמן אמת, תוך שליטה מלאה בלוגיקה ובזרימת הנתונים. שילוב תשתיית זו במערכת WebRTC מdice את יכולת הרחיב מערכת תקשורת בזמן אמת לעולמות של עיבוד חישובי, תוך שמירה על פרטיות, אבטחה ומודולריות.

## בדיקות (Testing)

בדיקות המערכת מהוות שלב מרכזי בתהליכי הפיתוח, ונועדו לוודא כי כל רכיב המערכת פועלם בהתאם לאפיון, עומדים בדרישות הפונקציונליות והלא-פונקציונליות, וספקים חווית שימוש יציבה ובטוחה. הבדיקות בפרויקט בוצעו בשלבים שונים, תוך התמקדות הן לצד הלקוח והן מצד השרת.

בדיקות בוצעו בסביבה מבוקרת, תוך שימוש בדפדפניים מודרניים ובכלי ניתוח "יעודיים ל-WebRTC".

## בדיקות צד לקוח (Client Testing)

### בדיקות חיבור מצלמה ומיקרופון

- בדיקת גישה למצלמה ולמיקרופון לאחר מתן הרשאות
- בדיקת תגובה במקורה של סירוב הרשות
- בדיקת החלפת התקני קלט בזמן ריצה

### בדיקות יצירת חיבור WebRTC

- בדיקת ייצור RTCPeerConnection בין שני משתתפים
- בדיקת ייצור Mesh בחדרים עם מספר משתתפים
- בדיקת קבלת זרמי וידאו וקול מכל Peer

## **בדיקות בקרי מדיה**

- בדיקת השתקת מיקרופון והחזרתו לפעולה
- בדיקת כיבוי והפעלת מצלמה
- בדיקת שיתוף מסך והפסקתו
- בדיקת יציאה מהחדר וסגירת חיבורים

## **בדיקות צד שרת (Server Testing)**

### **בדיקות Signaling**

- בדיקת חיבור WebSocket מאובטח (WSS)
- בדיקת שליחת וקבלת הודעות SDP
- בדיקת העברת ICE Candidates בין לaptops
- בדיקת ניטוב הودעות בין משתתפים בחדר

### **בדיקות ניהול חדרים**

- בדיקת יצירת חדר חדש
- בדיקת הצטרפות ויציאה מחדר
- בדיקת עדכון רשימת משתמשים בזמן אמת

## **בדיקות ניהול משתמשים וAIMOT**

- בדיקת תהליך הרשמה (Sign Up)
- בדיקת שליחת אימייל לאימומת משתמש
- בדיקת חסימת התחברות למשתמש לא מאומת
- בדיקת תהליך התחברות (Login)
- בדיקת הרשות Host לעומת Participant

## בדיקות אבטחת מידע

- בדיקת הצפנה תקורת WebSocket (WSS)
- בדיקת קיום חיבור HTTPS בצד הלוקה
- בדיקת הצפנה מדיה באמצעות DTLS-SRTP
- ידוא שאין מעבר מדיה דרך השרת

בדיקות אלו בוצעו באמצעות:

- Chrome DevTools
- WebRTC Internals
- ניתוח תעבורות רשת

## בדיקות טרשיי קצה (Edge Cases)

- הצליפות משתמש לחדר קיימ בזמן שיחה פעילה
- יציאת משתמש פטומית מהחדר
- נפילת חיבור רשת והתאוששות
- מעבר אוטומטי ל-TURN בעת חסימת P2P
- ריבוי משתתפים ויצירת Mesh מלא

## בדיקות תשתיית Machine Learning

- בדיקת קליטת פרויימים מהמצלמה
- בדיקת עיבוד פרויימים בזמן אמיתי
- בדיקת השפעת העיבוד על קצב הפרויימים
- בדיקת יציבות המערכת בעת הפעלת 필טרים

בדיקות אלו נועדו לוודא שעיבוד הוידאו אינו פוגע ביציבות התקשרות.

## סיכום הבדיקות

תהליך הבדיקות הוכיח כי המערכת פועלת בהתאם לאפיון, מספקת חיבור תקשורת יציבם, שומרת על אבטחת מידע ופרטיות, ומתחזקת בהצלחה עם

תרחישי שימוש שונים. שילוב בדיקות פונקציונליות, בדיקות אבטחה ובדיקות ביצועים תרם ליציבות הכללית של המערכת.

## הוראות התקינה והפעלה

פרק זה מתאר את שלבי התקינה, ההגדרה וההפעלה של המערכת, ומיעוד למשתמשים המעוניינים להריץ את המערכת בסביבה מקומית לצורכי בדיקה, פיתוח או שימוש.

המערכת מבוססת על טכנולוגיות Web ושרת Signaling קל משקל, ואינה דורשת התקינה של רכיבי תוכנה קבועים או תשתיות מיוחדות.

### דרישות מערכת

#### צד לקוח

- דפדפן מודרני התומך ב-WebRTC (Google Chrome / Mozilla Firefox)
- מצלמת רשות ומיקרופון
- חיבור אינטרנט פעיל

#### צד שרת

- Python 3

- גישה לרשת לצורך חיבור WebSocket במערכת Linux או Windows

## התקנת צד שרת (Signaling Server)

1. יש לוודא ש-3 מותקן במערכת Python 3+
2. להוריד את קבצי השירות מתיקית הפרויקט
3. להתקין חבילות נדרשות (במידת הצורך)
4. להפעיל את שירות ה-Signaling באמצעות פקודת הריצה "יעודית"
5. לוודא שהשירות מازין לחיבור WebSocket מאובטחים (WSS)
6. לפתח port forwarding

לאחר הפעלת השירות, הוא מוכן לקבל חיבור ללקוחות.

## הפעלת צד לקוח

1. לפתח את קבצי ה-Client בדף
2. לאשר גישה למצלמה ולמיקרופון
3. לוודא חיבור תקין לשירות ה-Signaling
4. להתחבר למערכת באמצעות משתמש קיימ או ליצור משתמש חדש

## הפעלת המערכת בסביבה מאובטחת

המערכת פועלת באמצעות HTTPS ו-WSS, ולכן יש להפעיל אותה בסביבה מאובטחת או מקומית התומכת בפרוטוקולים אלו. הפעלה זו נדרשת לצורך גישה ל-WebRTC בדפננים מודרניים.

## תקלות נפוצות ופתרונות

- אין גישה למצלמה – יש לבדוק הרשאות בדף
- לא נוצר חיבור ידאו – לבדוק חיבור רשות ושרת Signal
- משתמש אינו שומע קול – לבדוק הגדרות מיקרופון ורמקולים
- חיבור נופל – לבדוק מעבר אוטומטי ל-TURN

## מדריך למשתמש

פרק זה מתראר את אופן השימוש במערכת מנוקודת מבטו של משתמש הקצה, ומסביר כיצד לבצע פעולות עיקריות בצורה פשוטה וברורה.

### הרשמה למערכת

1. לפתח את מסך ההרשמה
2. להזין כתובת דואר אלקטרוני וosisma
3. לאשר את הרשמה באמצעות קישור שנשלח לדוא"ל
4. להתחבר למערכת לאחר האימויות

### התחברות למערכת

1. להזין כתובת דואר אלקטרוני וosisma
2. ללחוץ על כפתור התחברות
3. במקרה של שגיאה – לבדוק פרטיים או הودעת מערכות

### יצירת חדר חדש

1. לאחר התחברות, לבחור באפשרות "יצירת חדר"

2. החדר יווצר אוטומטית
3. המשתמש יוגדר כ-**Host**
4. ניתן לשתף את קישור החדר עם משתמשים נוספים

### **הצטרפות לחדר קיימ**

1. להזין מזהה חדר או קישור
2. לאשר הצטרפות
3. להתחל שיחה לאחר יצירת החיבורים

### **שימוש במכשיר שיחה**

- הפעלה / כיבוי מצלמה
- הפעלה / השתקת מיקרופון
- שיתוף מסך
- שליחת הודעות בצ'אט
- יציאה מהחדר

### **ניהול חדר (Host)**

- צפיה ברשימת משתתפים
- ניתוק משתמש
- שליטה בהרשאות בסיסיות

### **סיום שיחה**

1. לחיצה על "יציאה מהחדר"
2. סגירת החיבורים
3. חזרה למסך הראשי

## סיכום מדריך המשתמש

המדריך מאפשר שימוש פשוט וברור במערכת, גם למשתמשים ללא ידע טכני. שילוב של ממשק אינטואיטיבי עם תיעוד ברור תורם לחוויה שימוש חיובית ולפעול תקין של המערכת.

## סיכום ומשוב

פרויקט זה מהווה מימוש מלא של מערכת תקשורת בזמן אמת מבוססת WebRTC, המשלבת ידע תיאורטי ומעשי בתחום רשתות מחשבים, אבטחת מידע, פיתוח Web ועיבוד וידאו חישובי. במהלך העבודה על הפרויקט תוכננו ונבנתה מערכת שלמה מקצה לקצה, הכוללת צד לקוחות, צד שרת, ניהול משתמשים, ניהול חדרים, הצפנה מדיה ותשתיית לעיבוד וידאו מתקדם.

אחת המטרות המרכזיות בפרויקט הייתה להבין לעומק כיצד פועלות מערכות תקשורת בזמן אמת, ולא להסתפק בשימוש בספריות או שירותים מוכנים. בחירה זו הובילה למימוש עצמאי של שירות Signaling, ניהול חיבורו Peer-to-Peer באמצעות Mesh, והיכרות מעמיקה עם פרוטוקולי STUN, ICE, WebRTC, TURN ו-SDP.

בנוסף, הפרויקט כלל פיתוח תשתיית לעיבוד וידאו מבוסס Machine Learning בצד לקוחות, בגישה אלגוריתמית נמוכה ולא שימוש בספריות חיצונית יייעודיות. תשתיית זו מדגימה הבנה של עקרונות עיבוד תמונה, ניתוח פרימיום בזמן אמת ושיקול ביצועים, ומהווה בסיס להרחבות עתידיות של המערכת.

במהלך הפיתוח ניתנה חשיבות רבה לאבטחת מידע ולשמירה על פרטיות המשתמשים. המערכת תוכננה כך שתעבורת המידע תועבר ישירות בין המשתתפים בתקשורת מוצפנת מקצה לקצה, ללא אחסון או עיבוד מידע בצד השרת. גישה זו מצמצמת את משטח התקיפה ומחזקת את רמת האבטחה הכללית.

תהליך העבודה על הפרויקט טרם לפיתוח מיזמיות תכנון, ארכיטקטורה, ניתוח בעיות והפרדת שכבות מערכת. ההתמודדות עם אתגרים כגון ניהול חיבורים מרובים, תרחישים קצה בראשות ועיבוד וידעו בזמן אמת חיזקה את ההבנה ההנדסית ואת יכולת לפתח מערכות מורכבות בצורה מסודרת ואחראית.

לסיכום, הפרויקט השיג את מטרותיו ואף מעבר לכך. הוא מדגים מערכת תקשורת בזמן אמת יציבה, מאובטחת ומודולרית, ומהווה בסיס ראוי להמשך פיתוח, הרחבה ושיפור. העבודה על הפרויקט תרמה לרבות לידע, לניסיון ולבשלות המקצועית, והייתה תהליך למידה ממשמעותי ועמוק.