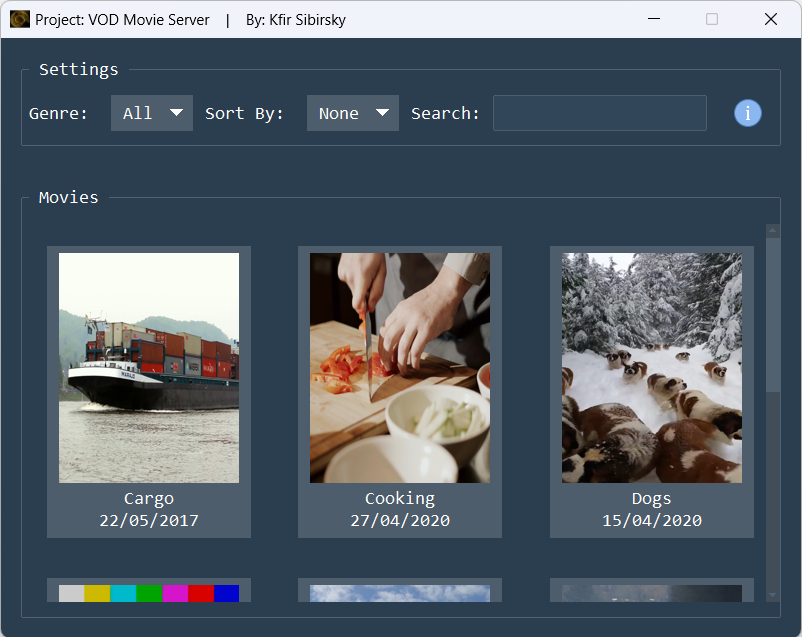
תיק פרויקט גמר

שרת סרטים  
(VOD)



**תוכן עניינים**

[1. מבוא 3](#_Toc157588050)

[2. דרישות קדם 3](#_Toc157588051)

[3. הוראות התקנה 4](#_Toc157588052)

[4. הוראות שימוש 4](#_Toc157588053)

[4.1. הפעלת האפליקציה 4](#_Toc157588054)

[4.2. סינון סרטים לפי ז'אנר 4](#_Toc157588055)

[4.3. מיון סרטים לפי קטגוריה 5](#_Toc157588056)

[4.4. חיפוש סרטים לפי שם 6](#_Toc157588057)

[4.5. תצוגת אודות 7](#_Toc157588058)

[4.6. הצגת פרטי סרט רצוי 9](#_Toc157588059)

[4.7. בחירת סרט לנגינה 10](#_Toc157588060)

[4.8. ניגון סרט 12](#_Toc157588061)

[4.9. יציאה מהאפליקציה 12](#_Toc157588062)

[5. ארכיטקטורת תוכנה 13](#_Toc157588063)

[5.1. כללי 13](#_Toc157588064)

[5.2. צד שרת 13](#_Toc157588065)

[5.3. צד לקוח 14](#_Toc157588066)

[5.4. מבנה המחלקות 14](#_Toc157588067)

[5.5. תיאור המחלקות 15](#_Toc157588068)

[5.6. הקשר בין המחלקות 16](#_Toc157588069)

[6. תהליכים עיקריים 17](#_Toc157588070)

[6.1. תהליך החיבור לשרת 17](#_Toc157588071)

[6.2. תהליך הצגת הסרטים הזמינים לצפייה 17](#_Toc157588072)

[6.3. תהליך הצגת פרטי הסרט הרצוי 18](#_Toc157588073)

[6.4. תהליך התחלת צפייה בסרט 18](#_Toc157588074)

[6.5. תהליך קפיצה לנקודת זמן במהלך צפייה בסרט 19](#_Toc157588075)

[6.6. תהליך סיום צפייה בסרט 20](#_Toc157588076)

[6.7. תהליך הורדת קובץ תיק פרויקט 20](#_Toc157588077)

[6.8. תהליך הניתוק מהשרת 20](#_Toc157588078)

[7. מקרי בדיקה 21](#_Toc157588079)

[8. פערים במצב הקיים 21](#_Toc157588080)

[9. CD של הפרויקט 22](#_Toc157588081)

# מבוא

בעידן הדיגיטלי של היום, הדרך בה אנו צורכים תוכן מדיה התפתחה באופן משמעותי.  
המודל המסורתי של צפייה בסרטים ובתוכניות טלוויזיה באמצעות כבלים או מדיה פיזית פינה את מקומו לשירותי הזרמת מדיה (בלועזית, Media Streaming) המספקים גישה מיידית לספריית תוכן עצומה.  
כחלק מהטרנספורמציה הדיגיטלית הזו, אני מציג את **"פרויקט שרת סרטים (VOD)",** מאמץ טכנולוגי שמטרתו לספק למשתמשים חווית צפייה מהנה, מהירה, ואמינה.

פקטור מרכזי בפרויקט הוא יישום תשתית סטרימינג חזקה ויעילה. לשם כך, נעשה שימוש בספריית נגן המדיה של VLC, המאפשרת ללקוחות להזרים סרטים בזמן אמת באמצעות כתובת RTSP URL ייחודית. טכנולוגיה זו מבטיחה למשתמשים חווית סטרימינג איכותית וללא הפרעות.

פרויקט זה מיושם כארכיטקטורת שרת-לקוח, כאשר הלקוחות מצוידים בנגן מדיה מותאם להזרמת הסרטים.

ממשק משתמש נקי, אינטואיטיבי וידידותי למשתמש תוכנן בקפידה כדי לשפר את חווית המשתמש, פקדי ניגון ומעבר חלק בין סרטים הם כולם חלק מחוויית המשתמש הזורמת.

נגן המדיה מציע מערך של תכונות שליטה, כולל הפעלה, השהייה, עצירה, התאמת עוצמת הקול וקפיצה לנקודת זמן ספציפית. כל אלו מספקים למשתמש שליטה מלאה בחוויית הצפייה שלהם.

לחוויה קולנועית, הפרויקט מציע מצב מסך מלא המסתיר אוטומטית את פקדי הניגון כדי להפיק את המירב מהנדל"ן במסך, ומספק חווית צפייה סוחפת.

בצד השרת, הפרויקט משתמש ב-backend מבוסס Python בסביבת Flask לפיתוח Web, תוך שימוש בממשק API RESTful לתקשורת בין הלקוח לשרת. כמו כן, נעשה שימוש בספריה  
flask-sqlalchemy למען גישה קלה לבסיס הנתונים (מסוג sqlite) בו מאוחסנים פרטי הסרטים באמצעות מיפוי אובייקטיבי-רלציוני (ORM). הדברים האלו מאפשרים ללקוחות לבקש ולהזרים סרטים ביעילות ומספק פתרון מדרגי (scalable) שניתן להרחבה.

למאמץ הזה יש השלכות מרחיקות לכת, לא רק עבור משתמשים בודדים אלא גם עבור עסקים הפועלים בתעשיית הסטרימינג והמדיה. ככל שהביקוש למדיה דיגיטלית ממשיך לעלות, פרויקטים כמו אלה סוללים את הדרך לחוויות משתמש מכילות ומספקות יותר. **"פרויקט שרת סרטים (VOD)"** הוא הצצה לעתיד הבידור הדיגיטלי, שבו התאמה אישית ונגישות עומדות בחזית החדשנות.

# דרישות קדם

האפליקציה מותאמת לעבודה בסביבת Windows – **נבדק על Windows 11**.

מסך בעל תמיכה ברזולוציה של 800x600 לפחות.

כמו כן, רצוי להתקין על עמדות בעלות גישה לרשת בא מפעילים את האפליקציה בעמדות נוספות בכדי להפיק את מירב התועלת מהאפליקציה.

# הוראות התקנה

בכדי להתקין את האפליקציה על תחנת קצה מסוימת יש להעתיק את קובץ ההרצה של האפליקציה **client.exe** המספק את כל היכולות של האפליקציה.

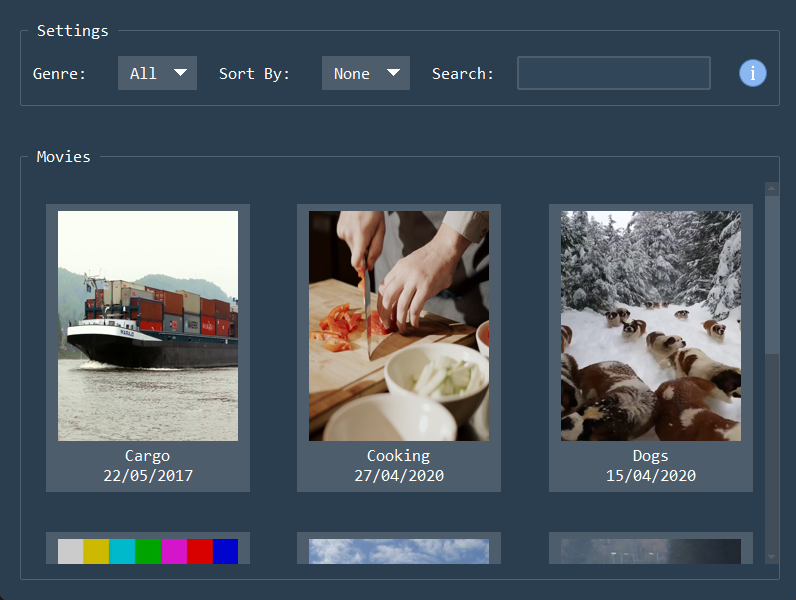
# הוראות שימוש

## הפעלת האפליקציה

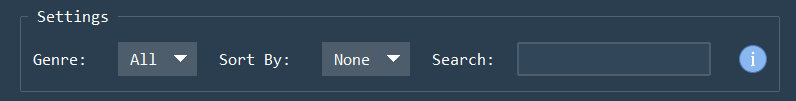
פתח את הקובץ **client.exe** באמצעות "דאבל קליק" על האייקון  .

## סינון סרטים לפי ז'אנר

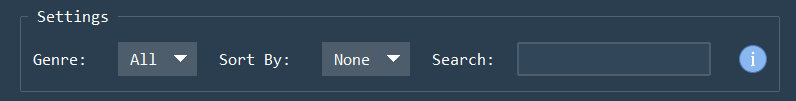
* גש לחלון הראשי ואתר את אזור 'הגדרות' (Settings).



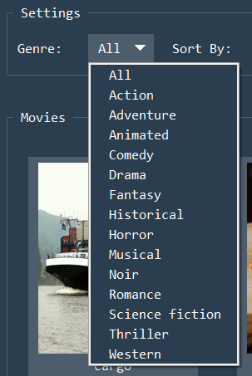
* גש לאזור 'הגדרות' (Settings) ואתר את התווית 'ז'אנר' (Genre).



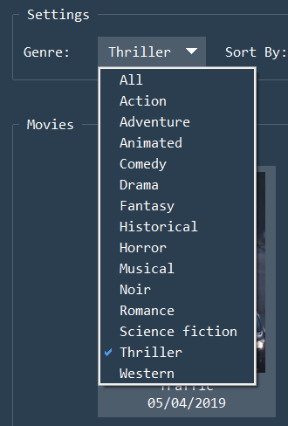
* מימין לתווית תמצא לשונית ובה כיתוב ז'אנר ברירת המחדל 'ALL':



* לחיצה על הלשונית מגלה תפריט ובו הז'אנרים השונים. בחר את הז'אנר הרצוי:

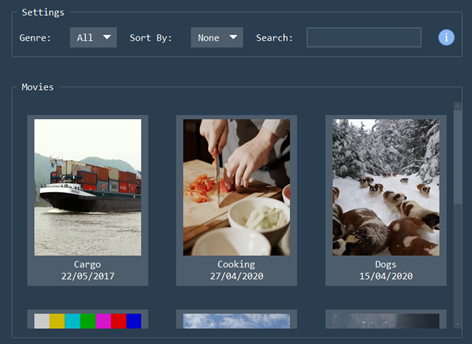


* לאחר בחירת הז'אנר, מתבצע סינון הסרטים והז'אנר הנבחר מסומן:

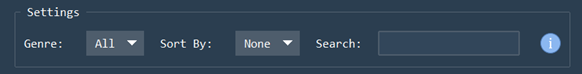


## מיון סרטים לפי קטגוריה

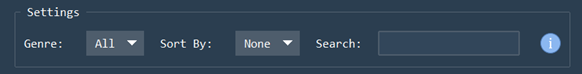
* גש לחלון הראשי ואתר את אזור 'הגדרות' (Settings).



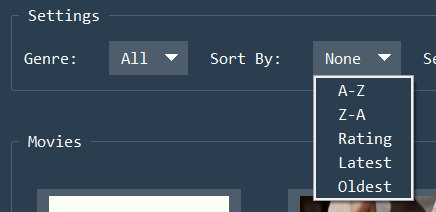
* גש לאזור 'הגדרות' (Settings) ואתר את התווית 'מיון לפי' (Sort By).



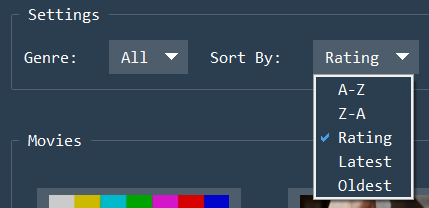
* מימין לתווית תמצא לשונית ובה כיתוב קטגורית ברירת המחדל 'None':



* לחיצה על הלשונית מגלה תפריט ובו הקטגוריות השונות. בחר את הקטגוריה הרצויה:

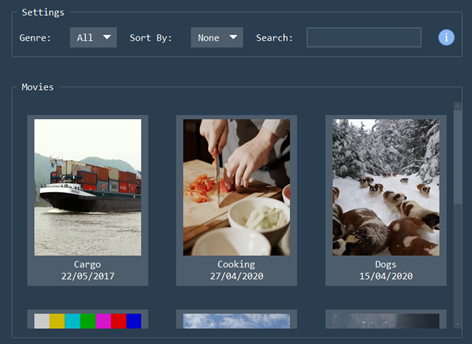


* לאחר בחירת הקטגוריה, מתבצע מיון הסרטים והקטגוריה הנבחרת מסומנת:

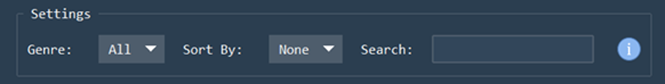


## חיפוש סרטים לפי שם

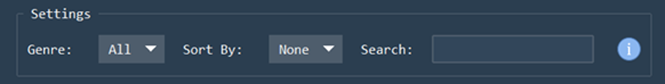
* גש לחלון הראשי ואתר את אזור 'הגדרות' (Settings).



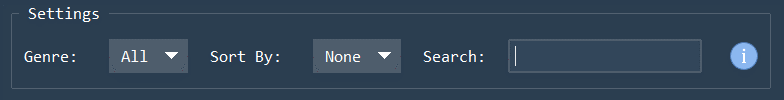
* גש לאזור 'הגדרות' (Settings) ואתר את התווית 'חיפוש' (Search).



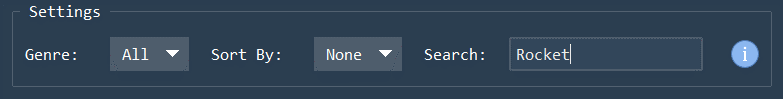
* מימין לתווית תמצא תיבת חיפוש:



* לחיצה על הלשונית מגלה סמן. הקלד את שם הסרט הרצוי:

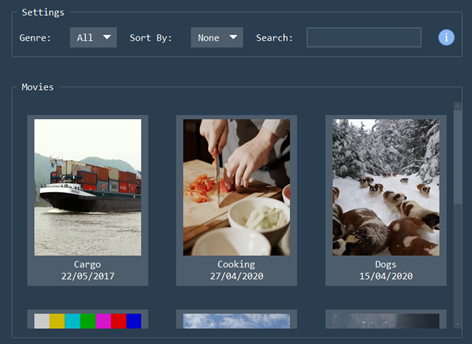


* תוך כדי ההקלדה, מתבצע חיפוש וסינון בזמן אמת:



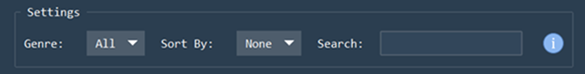
## תצוגת אודות

* גש לחלון הראשי ואתר את אזור 'הגדרות' (Settings).

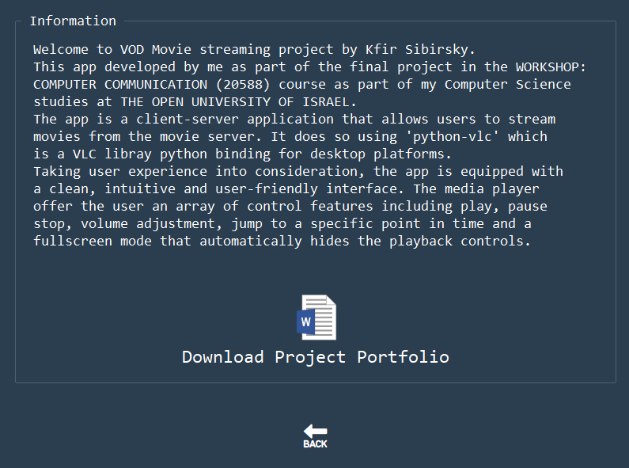


* גש לאזור 'הגדרות' (Settings), אתר את האייקון A blue circle with white letter i

  Description automatically generated ולחץ עליו.



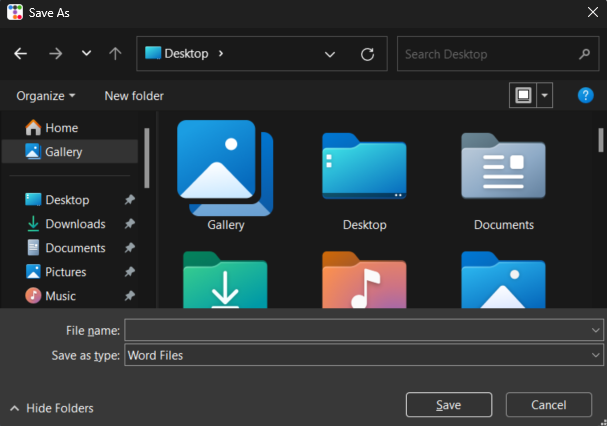
* לאחר הלחיצה, יוצגו אודות האפליקציה, כפתור להורדת תיק הפרויקט וכפתור חזרה לעמוד הראשי:



* להורדת הקובץ יש ללחוץ על כפתור ההורדה 'Download Project Portfolio' בימין עמוד 'אודות'.



* + לאחר הלחיצה על הכפתור יפתח דיאלוג לשמירת הקובץ. יש לבחור מיקום ושם לקובץ:

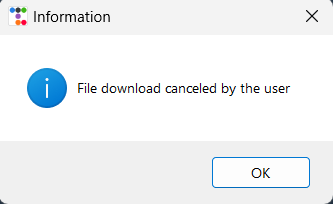


* + לשמירה יש ללחוץ על כפתור 'שמירה' (Save), הודעה על שמירת הקובץ תוצג:

A screenshot of a computer error

Description automatically generated

* + לביטול יש ללחוץ על כפתור 'ביטול' (Cancel), הודעה על ביטול ההורדה תוצג:

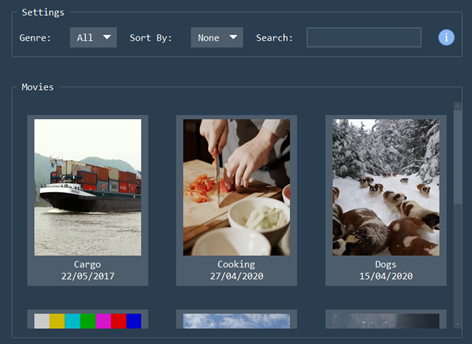


* לחזרה לחלון הראשי יש ללחוץ על כפתור החזרה 'BACK' בתחתית עמוד 'אודות'.

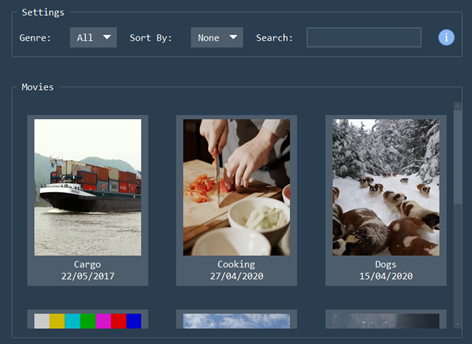


## הצגת פרטי סרט רצוי

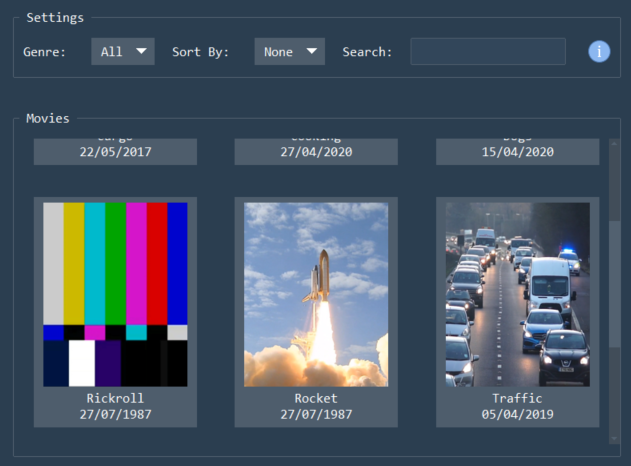
* גש לחלון הראשי ואתר את אזור 'סרטים' (Movies). באזור זה ניתן לראות מידע ראשוני על כל סרט, כגון שם הסרט, תאריך היציאה של הסרט לאקרנים ותמונת הפוסטר של הסרט:



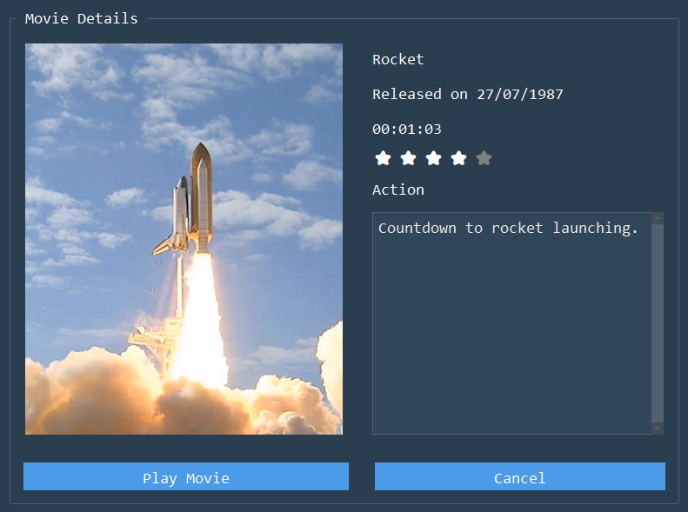
* גש לאזור 'סרטים' (Movies), אתר את פס הגלילה וגלול בגלריית הסרטים:



* בהגיעך לסרט הרצוי, לחץ על כפתור הסרט התואם:

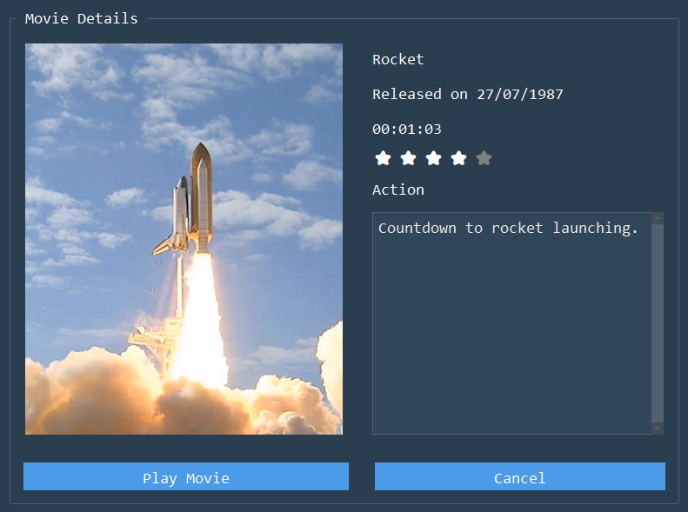


* לאחר הלחיצה, יוצגו פרטי הסרט – שם הסרט, תאריך יציאה לאקרנים, משך זמן, דירוג (1-5 כוכבים), ז'אנר, תקציר ותמונת הפוסטר. כמו כן, כפתור 'ביטול' (Cancel – לחיצה עליו מחזירה לחלון הראשי) וכפתור 'ניגון סרט' (Play Movie – לחיצה עליו מעבירה לחלון נגן הסרט):

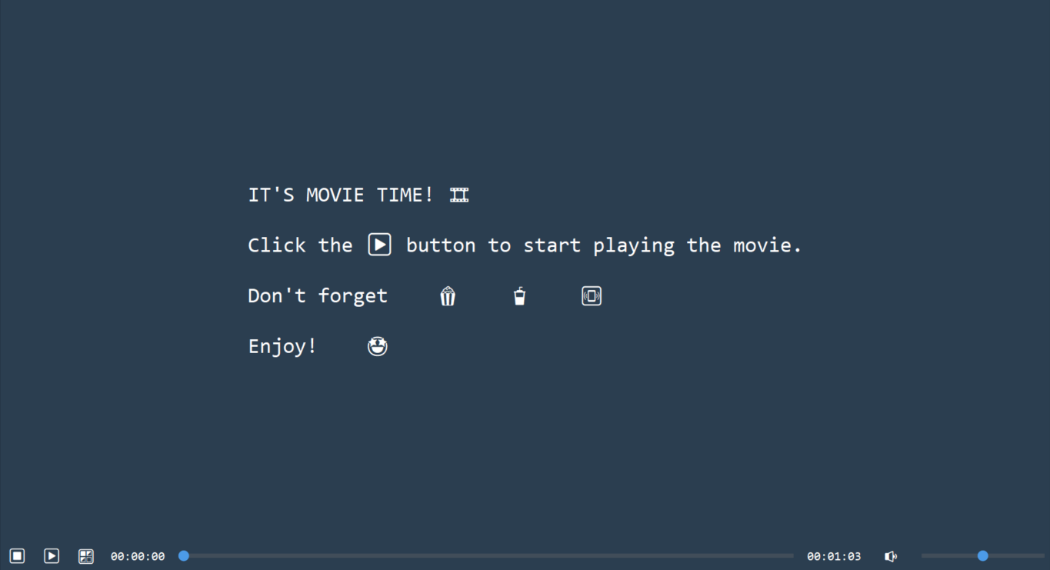


## בחירת סרט לנגינה

* בצע את שלב 4.6 במדויק. כעת יש ללחוץ על כפתור 'ניגון סרט' (Play Movie):



* כעת, יוצג נגן המדיה ובו אזור נגינת הסרט (המכיל הוראות להתחלת הניגון) ושורת פקדי ניגון:



* שורת פקדי הניגון מכילה מספר כלים – כפתור עצירה, כפתור הפעלה/השהייה, כפתור מסך מלא, תווית זמן עבר, סליידר ניגון, תווית זמן נותר, כפתור השתקה וסליידר שמע :



* + כפתור העצירה  – לחיצה על כפתור זה עוצרת (Stop) את הסרט וחוזרת לגלריית הסרטים.
  + כפתור ההפעלה/השהייה  – לחיצה על כפתור זה לראשונה מפעילה (Play) את הסרט ומחליפה את האייקון (לסירוגין) לאייקון . בכל לחיצה נוספת מבצעת לסירוגין השהייה (Pause) וביטול השהייה (Un-Pause) ומחליפה את האייקון בהתאם.
  + כפתור מסך מלא  – לחיצה על כפתור זה מבצעת לסירוגין הכנסה למצב מסך מלא (Fullscreen ON) וביטול מסך מצב מלא (Fullscreen OFF).

כאשר מצב מסך מלא מופעל (Fullscreen ON), שורת פקדי הניגון נעלמת בצורה אוטומטית. לחיצה על מקש  במקלדת תציג/תסתיר לסירוגין את שורת פקדי הניגון.

שים לב כי המקלדת צריכה להיות במצב שיטת קלט אנגלית.

* + תווית זמן עבר  – מציגה את הזמן שעבר מתחילת ניגון הסרט בפורמט HH:MM:SS, כאשר כל שנייה ערך תווית זה מתעדכן.
  + סליידר ניגון  – מציג את נקודת הזמן הנוכחית המתנגנת כרגע, כאשר כל שנייה ערכה מתעדכן.

לחיצה והחזקה של הידית (העיגול הכחול) יחד עם הזזה לנקודת זמן רצויה ושחרור הידית, תגרום לניגון הסרט החל מאותה נקודת זמן. כלומר קפיצה לנקודת זמן רצויה (קדימה או אחורה).

* + תווית זמן נותר  – מציגה את הזמן שנותר עד לסוף ניגון הסרט בפורמט HH:MM:SS, כאשר כל שנייה ערך תווית זה מתעדכן.
  + כפתור השתקה  – לחיצה על כפתור זה מבצעת לסירוגין השתקה (Mute) יחד עם החלפת האייקון לאייקון  וביטול השתקה (Un-Mute) יחד עם הגדרת עוצמת השמע ל-50% והחלפת האייקון לאייקון . כמו כן אייקון הכפתור משתנה בהתאם לעוצמת השמע המוגדרת על ידי סליידר השמע באופן הבא:  
     – עבור ערך עוצמת השמע השווה ל- 0%.

 – עבור ערך עוצמת השמע בין 1% ל- 33%.

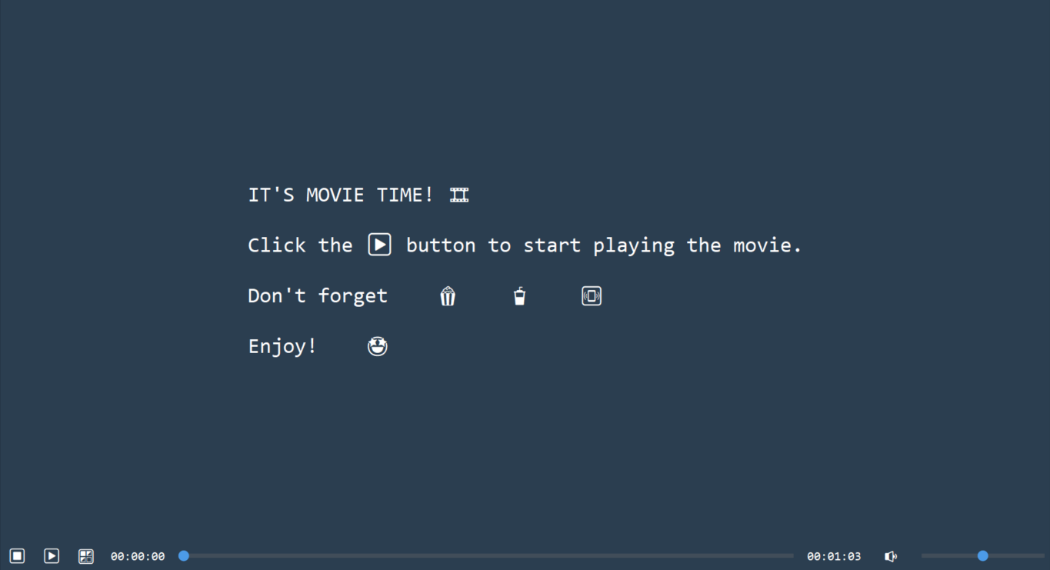
 – עבור ערך עוצמת השמע בין 33% ל- 66%.

 – עבור ערך עוצמת השמע בין 66% ל- 100%.

* + סליידר שמע  – מציג את ערך עוצמת השמע הנוכחי. לחיצה והחזקה של הידית (העיגול הכחול) יחד עם הזזה לערך עוצמת שמע רצוי ושחרור הידית, תגרום לעדכון ערך עוצמת השמע.

## ניגון סרט

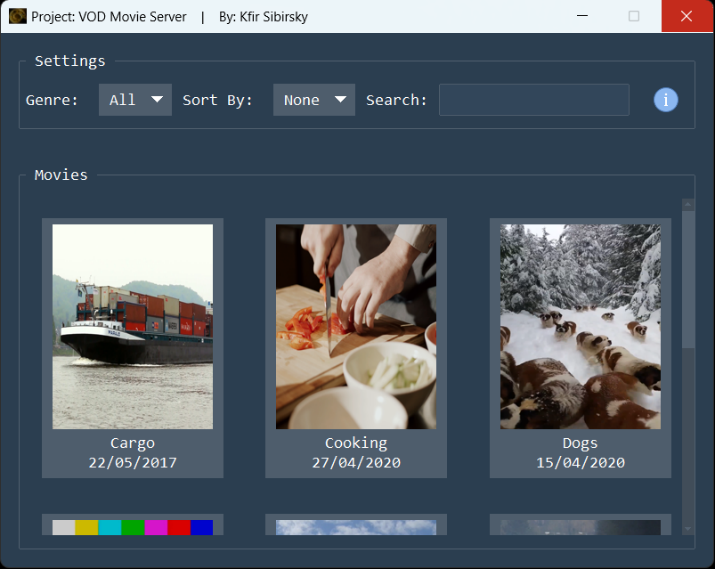
* בצע את שלב 4.7 במדויק. כעת יש ללחוץ על כפתור ההפעלה/השהייה  להתחלת ניגון הסרט.  
  הסבר זה נמצא גם כהערה על גבי נגן המדיה – הערה זו תיעלם באופן אוטומטי עם תחילת ניגון הסרט. אל תשכח פופקורן🍿, שתייה🥤, טלפון נייד על מצב רטט📳 ושתהיה צפייה נעימה🤩



* כעת, הסרט מתחיל להתנגן, על מנת לעצור/להשהות/להפעיל את הסרט או להיכנס/לצאת ממצב מסך מלא או לקפוץ לנקודת זמן רצויה או לשנות את עוצמת השמע יש לעיין בשלב 4.7 בקטע המסביר את הפעולה הרצויה.

## יציאה מהאפליקציה

* על מנת לצאת מהאפליקציה יש ללחוץ על סימן **❎** בפינה העליונה של חלון התוכנה:



# ארכיטקטורת תוכנה

## כללי

**"פרויקט שרת סרטים (VOD)"** (Video-on-Demand) היא אפליקציית שרת-לקוח שנועדה לאפשר למשתמשים להזרים סרטים משרת מרוחק, להבטיח אפליקציה מדרגית, ניתנת לתחזוקה ויעילה כאשר צד השרת מארח נתוני סרטים ומזרים תוכן, בעוד שצד הלקוח מספק ממשק ידידותי למשתמש לגלישה ולהפעלת סרטים.

האינטראקציה בין הלקוח והשרת מתבצעת בעיקר באמצעות פרוטוקול HTTP, תוך שימוש בממשק  
API RESTful לחילופי נתונים והזרמת הסרטים. השימוש בספריות כמו Flask, VLC ו- Tkinter (ובפרט ttkbootstrap שהיא הרחבת עיצוב מודרנית ל-Tkinter), יחד עם ניהול מסד נתונים נכון, מייצר חווית משתמש וצפייה סוחפות.

libVLC הוא מנוע הליבה (Core Engine) והממשק למסגרת המולטימדיה עליה מבוסס נגן המדיה VLC. הן צד השרת והן צד הלקוח עושים שימוש ב-'python-vlc' שהיא כריכה (Binding) עבור libVLC לפלטפורמת שולחן עבודה (Desktop).

האפליקציה מעוצבת בארכיטקטורת Object Oriented Programming שכן קוד הפרויקט נבנה עם מחלקות, ירושה, אנקפסולציה ועקרונות OOP אחרים. בכך למעשה האפליקציה מאורגנת בדרך יעילה שבין היתר מקדמת שימוש חוזר בקוד.

## צד שרת

צד השרת (Server-side) אחראי על אירוח (Hosting) והגשת (Serving) סרטים ללקוחות, כאשר הוא תוכנן לספק זמינות גבוהה, אחסון נתונים יעיל ומדרגיות (Scalability).

בלב ארכיטקטורת צד השרת נמצא שרת web שנוצר באמצעות שימוש בספרייה Flask שהיא מסגרת קלת משקל וגמישה המתאימה ליישומים קטנים עד בינוניים. כמו כן, היא מאפשרת לשרת לקבל בקשות (Requests) HTTP מלקוחות, לנהל מסלולים (Routes) ולהחזיר תגובות (Responses) מתאימות.

השרת מקיים אינטראקציה עם מסד נתונים SQLite המכיל את פרטי הסרטים.  
יתרונותיו של מסד נתונים זה שקובץ ההרצה שלו מאוד קומפקטי (קובץ בודד בנפח של כ- 0.5MB), הוא אינו מצריך התקנה וקונפיגורציה, ותומך בנפחים גדולים של מידע (140TB למסד).

הסרטים עצמם ותמונות הפוסטר שלהם מאוחסנים בתיקיית הנכסים assetsשבצד השרת.

כדי להגיש תוכן סרטים ללקוחות ביעילות, השרת משתמש ב-'python-vlc' ובכך למעשה מתבצע הטיפול בהזרמת הסרטים (Movie Streaming) והבטחת הפעלה חלקה.

בעזרת שימוש בכריכה זו ניתן לייצר מופע (Instance) של VLC, נגן מדיה (Media Player) לאותו המופע ולהגדיר מדיה (Media) שאותה יזרים השרת (באמצעות הפרוטוקולים RTP/RTCP/RTSP) לכתובת (URL) RTSP שתישלח ללקוח.

צד השרת חושף ממשק API לתקשורת עם צד הלקוח. הדבר מאפשר ללקוחות לבקש רשימות סרטים, פרטי סרטים ומידע ספציפי למשתמש. זהו מרכיב מרכזי בהפעלת תקשורת בין הלקוח לשרת.

## צד לקוח

האפליקציה מבוססת על קובץ הרצה (Executable) בודד, כאשר קובץ זה מכיל הן את הלוגיקה והן את הממשק הגרפי למשתמש.

צד הלקוח (Client-side) אחראי על אינטראקציה עם המשתמש והצגת מידע על סרטים והפעלתם, כאשר הוא תוכנן עם חווית משתמש כעדיפות עליונה.

הממשק המשתמש הגרפי (UIG) בפרויקט זה נוצר באמצעות Tkinter, ספריית Python פופולרית לבניית יישומי שולחן עבודה (Desktop) וההרחבה ttkbootstrap עבורה לעיצוב מודרני. הוא מספק למשתמשים ממשק נקי, אחיד, עקבי ואינטואיטיבי לגלישה ולהפעלת סרטים.

הלקוח מתקשר עם השרת באמצעות בקשות HTTP, בדרך כלל מבצע בקשות GET לאחזור פרטי סרט ובקשות POST לפעולות משתמש כגון הזרמת סרט. הוא משתמש בספריית הבקשות (Requests) של Python כדי לבצע פעולות אלה.

כדי לנגן תוכן סרטים מהשרת ביעילות, הלקוח משתמש ב-'python-vlc' ובכך למעשה מתבצע הטיפול בניגון הסרטים (Movie Streaming) והבטחת הפעלה חלקה.

נגן המדיה מציע מערך פקדי ניגון הכולל הפעלה, השהייה, עצירה, התאמת עוצמת הקול, קפיצה לנקודת זמן ספציפית ומצב מסך מלא.

Threading – כדי להבטיח זמני תגובה מהירים למשתמש, הלקוח משתמש ב-thread נפרד לעדכון (בכל שנייה) ערכי תווית זמן עבר/נותר ומיקום ידית סליידר ניגון במקביל לנגן את הסרט.

## מבנה המחלקות

האפליקציה מורכבת ממספר מחלקות. להלן תרשים המציג את מחלקות האפליקציה השונות והקשר ביניהן:

**Main  
Window**

**Client**

**Server**

**Base  
Window**

**Movie  
Gallery  
Window**

**Information  
Window**

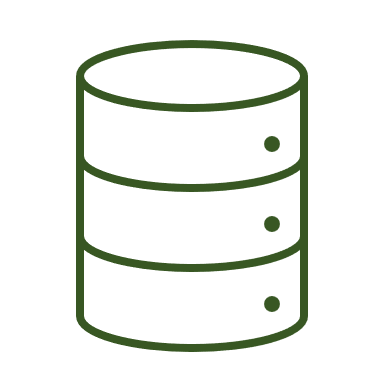
**Movie  
Player  
Window**

**Selected  
Movie  
Window**

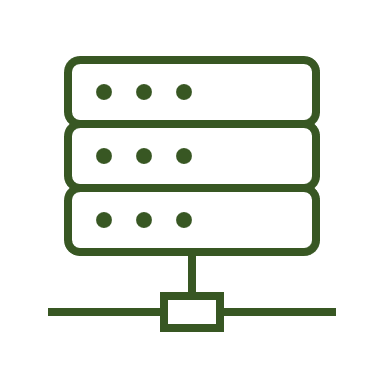
**Clients**

**Movies**

**Client**



**Database**



## תיאור המחלקות

|  |  |  |
| --- | --- | --- |
| צד | מחלקה | תפקיד |
| שרת | Client | מחלקה זו שייכת לצד השרת וחולקת את אותו שם כמו המחלקה Client שבצד הלקוח, אך היא מחלקה אחרת עם מטרה אחרת.  מחלקה זו מייצגת לקוח המחובר לשרת הסרטים. לכל לקוח כזה יש מזהה לקוח ייחודי איתו הלקוח פונה לשרת בבקשה כלשהי. כך למעשה השרת יודע איזה לקוח פנה אליו. כמו כן, לכל לקוח כזה יש תכונות VLC שבאמצעותם השרת מזרים את הסרט המתאים ללקוח המתאים. |
| שרת | Clients | מחלקה המייצגת מאגר לקוחות המחוברים לשרת הסרטים. כל לקוח במאגר יכול להיות מחובר בלבד או מחובר וצופה. כמו כן, למחלקה זו מספר מונים שבאמצעותם מתבצעים הקצאת מזהה לקוח, הוספה והסרה של לקוחות, עדכון סטטוס צפייה ועוד. |
| שרת | Movies | מחלקה המייצגת (באמצעות SQLAlchemy) את טבלת הסרטים בבסיס הנתונים, כלומר משקפת את מבנה רשומות הסרטים בטבלה. בסיס הנתונים מאחסן מידע על הסרטים, כולל שם הסרט, תאריך שחרור לאקרנים, אורך, ז'אנר, תקציר, דירוג ועוד. המחלקה מספקת שיטה להחזרת מידע על סרט כמילון (מבנה נתונים של Python). מחלקה זו יורשת מ-db.Model כאשר db=SQLAlchemy(app) ו-app היא אפליקציית Flask של השרת. |
| לקוח | MainWindow | מחלקה זו מייצגת את החלון הראשי של האפליקציה בצד הלקוח.  בין היתר מאתחלת וממרכזת על המסך את החלון הראשי, מגדירה את כותרת החלון ורשימת מסגרות חלון המשמשת למעבר בין החלונות השונים, כאשר כל מסגרת היא סוג שונה של חלון. |
| לקוח | BaseWindow | מחלקה זו מהווה מחלקת בסיס מופשטת המייצגת חלון בסיסי באמצעות ירושה מ- 'ווידג'ט מסגרת' (ttk.Frame) של Tkinter. למעשה מספקת תבנית ליצירת חלונות שונים באפליקציית הלקוח כך שמחלקות משנה (סוגי חלונות שונים) יורשות ממנה ומיישמות פונקציות ספציפיות לחלון. |
| לקוח | MovieGalleryWindow | מחלקה זו מייצגת את חלון גלריית הסרטים בצד הלקוח. היא מציגה רשימה של סרטים זמינים להזרמה משרת הסרטים. כמו כן, מטפלת באינטראקציות עם המשתמש כמו סינון סרטים לפי ז'אנר, מיון הסרטים לפי קטגוריה, חיפוש סרט, בחירת סרט למען הצגת פרטיו ומציעה אפשרות לעבור לחלון אודות. מחלקה זו יורשת מ-BaseWindow. |
| לקוח | SelectedMovieWindow | מחלקה זו מייצגת את חלון פרטי הסרט הנבחר. היא מציגה את פרטי הסרט הנבחר – שם הסרט, תאריך יציאה לאקרנים, משך זמן, דירוג (1-5 כוכבים), ז'אנר, תקציר ותמונת הפוסטר. כמו כן, מטפלת באינטראקציות עם המשתמש כמו ביטול הפעולה וחזרה לגלריה או המשך הפעולה ומעבר לחלון נגן הסרט. מחלקה זו יורשת מ-BaseWindow. |
| לקוח | MoviePlayerWindow | מחלקה זו מייצגת את חלון נגן הסרט. היא מציגה את אזור נגינת הסרט (המכיל את הנגן עצמו והוראות להתחלת הניגון) ושורת פקדי ניגון. כמו כן, מטפלת באינטראקציות עם המשתמש כמו הפעלה/השהייה/עצירה של הסרט, כניסה/יציאה ממצב מסך מלא, קפיצה לנקודת זמן רצויה ושינוי עוצמת השמע. מחלקה זו יורשת מ-BaseWindow. |
| לקוח | InformationWindow | מחלקה זו מייצגת את חלון האודות. היא מציגה את אודות הפרויקט ומטפלת באינטראקציות עם המשתמש כמו הורדת קובץ תיק פרויקט וחזרה לחלון הגלריה. מחלקה זו יורשת מ-BaseWindow. |
| לקוח | Client | מחלקה זו שייכת לצד הלקוח וחולקת את אותו שם כמו המחלקה Client שבצד השרת, אך היא מחלקה אחרת עם מטרה אחרת.  מחלקה זו מייצגת את אפליקציית הלקוח והחיבור שלה לשרת הסרטים. היא מטפלת בפונקציות כמו התחברות לשרת, בקשת נתוני סרטים, הפעלת הזרמות סרטים וניהול אינטראקציה עם המשתמש. |

## הקשר בין המחלקות

אפליקציית הלקוח מופעלת בפונקציה main() של הלקוח, היוזמת את החיבור לשרת ויוצרת מופע של MainWindow המשמש כחלון 'הורה' עבור חלונות אחרים.

למעשה מופע זה יוצר מופעים של מחלקות חלון שונות (MovieGalleryWindow, SelectedMovieWindow, MoviePlayerWindow, InformationWindow) כך שהוא מציג/מסתיר חלונות לפי הצורך בהתבסס על אינטראקציות המשתמש.

מחלקות אלה שונות בהרבה מובנים אך יש להן בסיס משותף המוגדר במחלקה BaseWindow שהן יורשות ממנה. למעשה, המשתמשים מנווטים בין חלונות שונים (חלון הגלריה, חלון פרטי הסרט שנבחר, חלון נגן הסרטים וחלון אודות) על ידי מעבר בין מופעים של מחלקות חלונות שונות.

המחלקה Client (בצד לקוח) מטפלת בחיבור לשרת ובבקשות השונות (למשל בקשת נתוני הסרטים הזמינים לצפייה, מה שמאפשר למשתמשים לבחור ולהזרים לצפייה סרטים).

אפליקציית השרת מופעלת בפונקציה main() של השרת. היא מנוהלת על ידי Flask ובאמצעות SQLAlchemy יוזמת חיבור לבסיס הנתונים של פרטי הסרטים. (קבצי הסרטים עצמם אינם מאוחסנים בתוך הטבלה, אלא בתיקיית assets שבצד השרת).

למעשה היכולת לתשאל (Query) את בסיס הנתונים מושגת באמצעות אובייקט מחלקת Movies שמהווה מודל לבסיס הנתונים ובפרט לטבלת פרטי הסרטים.

המחלקה Clients מנהלת את מאגר הלקוחות המחוברים לשרת. יחד עם המחלקה Client (בצד שרת) מנהלות את החיבור, ההזרמה, והאינטראקציה של הלקוחות עם נגן המדיה של VLC.

לכל Client ב-Clients יש מופע VLC ייחודי, כך שבהתאם לבקשת אותו לקוח לניגון סרט מסוים, מיקום קובץ הסרט נשלף מבסיס הנתונים ותוכנו מוזרם באמצעות מופע VLC הייחודי לכתובת RTSP ייחודית שמוחזרת ללקוח המבקש.

# תהליכים עיקריים

## תהליך החיבור לשרת

התהליך מופעל באופן אוטומטי בעליית האפליקציה (לפני שהממשק הגרפי עולה).

* בצד לקוח:
  + נשלחת בקשת HTTP מסוג GET לשרת לכתובת מהצורה 'server\_url/connect\_new\_client\_to\_server' ומחכה לתגובה.
  + לאחר קבלת התגובה, מחולץ ממנה מזהה הלקוח שהשרת הקצה עבור המשתמש הנוכחי והוא נשמר בתכונה client\_id של Client.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה '/connect\_new\_client\_to\_server'.
  + מיוצר לקוח מחובר חדש (Client חדש של צד שרת מתווסף לרשימה הלקוחות של Clients) בעל מזהה לקוח ייחודי.
  + מזהה הלקוח מוחזר בפורמט JSON כתגובה לבקשה.

## תהליך הצגת הסרטים הזמינים לצפייה

התהליך מופעל באופן אוטומטי בעליית האפליקציה ומתבצע בשני חלקים.

החלק שמבוצע מיד לאחר ביצוע ההתחברות לשרת:

* בצד לקוח:
  + נשלחת בקשת HTTP מסוג GET לשרת לכתובת מהצורה 'server\_url/get\_movies' ומחכה לתגובה.
  + לאחר קבלת התגובה, מחולצת ממנה רשימת פרטי הסרטים והיא נשמרת בתכונה movies של Client.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה '/get\_movies'.
  + הרצת שאילתה על בסיס הנתונים המחזירה את פרטי הסרטים של כל רשומות טבלת הסרטים.
  + פרטי הסרטים שהוחזרו מוכנסים לרשימת מילונים המוחזרת בפורמט JSON כתגובה לבקשה.

החלק שמבוצע בעת יצירת חלון גלריית הסרטים (MovieGalleryWindow):

* בצד לקוח:
  + עבור כל סרט ברשימה פרטי הסרטים (תכונת movies של Client) מיוצר כפתור (ווידג'ט של Tkinter) המכיל פרטים ראשוניים בלבד (שם הסרט, תאריך שחרור לאקרנים, תמונת פוסטר).

## תהליך הצגת פרטי הסרט הרצוי

התהליך מופעל באופן יזום על ידי המשתמש (ראה סעיף 4.6) לאחר לחיצה על כפתור הסרט הרצוי בחלון גלריית הסרטים (MovieGalleryWindow).

* בצד לקוח:
  + עדכון ערך התכונה selected\_movie\_id של Client.
  + עדכון של כל שדות פרטי הסרט בחלון פרטי הסרט הנבחר (SelectedMovieWindow) ומעבר אליו.

## תהליך התחלת צפייה בסרט

התהליך מופעל באופן יזום על ידי המשתמש (ראה סעיפים 4.7 ו-4.8) לאחר לחיצה על כפתור ניגון הסרט בחלון פרטי הסרט הנבחר (SelectedMovieWindow) ומתבצע במספר חלקים.

החלק שמתבצע מיד לאחר לחיצה על כפתור ניגון הסרט בחלון פרטי הסרט הנבחר:

* בצד לקוח:
  + נשלחת בקשת HTTP מסוג POST לשרת לכתובת מהצורה 'server\_url/get\_movie\_rtsp\_url/client\_id/selected\_movie\_id' ומחכה לתגובה.
  + לאחר קבלת התגובה, מחולצת ממנה כתובת ה-RTSP והיא נשמרת בתכונה rtsp\_url של Client.
  + צירוף נגן המדיה למסגרת (ווידג'ט של Tkinter) נגן הסרטים.
  + יצירת מדיה חדשה בהתאם למיקום הסרט (rtsp\_url).
  + השמתה לנגן השמתה לנגן המתאים ללקוח.
  + יצירת שורת פקדי ניגון.
  + התחלת thread חדש לעדכון בכל שנייה של ערכי תוויות זמן עבר/נותר וסליידר הניגון.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה   
    '/get\_movie\_rtsp\_url/client\_id/movie\_id'.
  + הרצת שאילתה המחזירה את פרטי הסרט בעל מזהה הסרט הנתון.
  + הבאת הלקוח (Client ברשימת Clients) בעל מזהה הלקוח הנתון.
  + יצירת מדיה חדשה בהתאם למיקום הסרט (נמצא באחד השדות שחזרו מהשאילתה).
  + הגדרת המדיה להזרמה באמצעות פרוטוקול RTP/RTCP/RTSP לכתובת ייחודית (מוטמעים בה מזהה הלקוח ומזהה הסרט).
  + השמתה לנגן המתאים ללקוח.
  + כתובת ה-RTPS מוחזרת בפורמט JSON כתגובה לבקשה.

החלק שמתבצע מיד לאחר לחיצה על כפתור ניגון הסרט בחלון נגן הסרטים:

* בצד לקוח:
  + נשלחת בקשת HTTP מסוג POST לשרת לכתובת מהצורה   
    'server\_url/start\_streaming/client\_id ' ומחכה לתגובה.
  + לאחר קבלת התגובה, הפעלת הנגן של הלקוח (Client) המתאים (כלומר התחלת ניגון הסרט המוזרם מהשרת).
  + כעת המשתמש יכול לבצע שלל פעולות לשליטה על חווית הצפייה באמצעות שורת פקדי הניגון.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה '/start\_streaming/client\_id'.
  + הבאת הלקוח (Client ברשימת Clients) בעל מזהה הלקוח הנתון.
  + עדכון ערך דגל ההזרמה ועדכון מונה הלקוחות הצופים.
  + הפעלת הנגן של הלקוח המתאים (כלומר הפעלת ההזרמה מהשרת ללקוח המתאים).
  + הודעת חיווי מוחזרת בפורמט JSON כתגובה לבקשה.

## תהליך קפיצה לנקודת זמן במהלך צפייה בסרט

התהליך מופעל באופן יזום על ידי המשתמש (ראה סעיפים 4.7 ו-4.8) לאחר הזזת ידית סליידר הניגון בחלון נגן הסרט (MoviePlayerWindow).

* בצד לקוח:
  + המרת נקודת הזמן אליה יש לקפוץ לפורמט HH:MM:SS.
  + נשלחת בקשת HTTP מסוג POST לשרת לכתובת מהצורה   
    'server\_url/skip\_to\_timestamp/client\_id/hours:minutes:seconds.
  + כעת נקודת הזמן התעדכנה והמשתמש יכול לבצע שלל פעולות לשליטה על חווית הצפייה באמצעות שורת פקדי הניגון.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה 'skip\_to\_timestamp/client\_id/hours:minutes:seconds/'.
  + הבאת הלקוח (Client ברשימת Clients) בעל מזהה הלקוח הנתון.
  + המרת נקודת הזמן אליה יש לקפוץ לאלפיות שנייה (VLC משתמש באלפיות שנייה).
  + עדכון נקודת הזמן והפעלת הנגן של הלקוח המתאים החל מנקודת הזמן הזו.
  + הודעת חיווי מוחזרת בפורמט JSON כתגובה לבקשה.

## תהליך סיום צפייה בסרט

התהליך מופעל באופן יזום על ידי המשתמש (ראה סעיפים 4.7 ו-4.8) לאחר לחיצה על כפתור עצירת הסרט בחלון נגן הסרט (MoviePlayerWindow) או באופן אוטומטי לאחר סיום הסרט.

* בצד לקוח:
  + איתות (באמצעות אירוע) לעצירת ה-thread המעדכן בכל שנייה את ערכי שורת פקדי הניגון.
  + עצירת הנגן של הלקוח (Client) המתאים (כלומר עצירת ניגון הסרט המוזרם מהשרת).
  + נשלחת בקשת HTTP מסוג POST לשרת לכתובת מהצורה   
    'server\_url/stop\_streaming/client\_id/selected\_movie\_id'.
  + הריסת שורת פקדי הנגינה.
  + במידת הצורך, יציאה ממצב מסך מלא.
  + חזרה לחלון גלריית הסרטים.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה '/stop\_streaming/client\_id/movie\_id'.
  + הבאת הלקוח (Client ברשימת Clients) בעל מזהה הלקוח הנתון.
  + עצירת הנגן שבצד השרת.
  + עדכון ערך דגל ההזרמה.
  + איפוס תכונות VLC של הלקוח (Client) .
  + עדכון מונה הלקוחות הצופים.

## תהליך הורדת קובץ תיק פרויקט

התהליך מופעל באופן יזום על ידי המשתמש (ראה סעיפים 4.5) לאחר לחיצה על כפתור הורדת תיק פרויקט בחלון אודות (InformationWindow).

* בצד לקוח:
  + נשלחת בקשת HTTP מסוג GET לשרת לכתובת מהצורה   
    'server\_url/download\_project\_portfolio ומחכה לתגובה.
  + לאחר קבלת התגובה, נפתח דיאלוג לשמירת הקובץ.
  + לאחר שמירתו או ביטול השמירה, הודעת חיווי תקפוץ בהתאם.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה 'download\_project\_portfolio/'.
  + שליחת קובץ תיק פרויקט כתגובה לבקשה.

## תהליך הניתוק מהשרת

התהליך מופעל באופן אוטומטי ביציאת האפליקציה.

* בצד לקוח:
  + נשלחת בקשת HTTP מסוג POST לשרת לכתובת מהצורה   
    'server\_url/client\_exit/client\_id'.
* בצד שרת:
  + מתקבלת הבקשה בנתיב מהצורה '/client\_exit/client\_id'.
  + עדכון מונה הלקוחות המחוברים.
  + נבדק סטטוס הנגן של הלקוח בעל מזהה הלקוח הנתון (Client ברשימת Clients) ובהתאם, אם הבקשה להתנתקות (מהלקוח Client) התקבלה בעת שהוא צופה בסרט או בעת שהשרת מזרים לו את הסרט, אז מתבצעים עצירת הנגן של שבצד השרת, עדכון ערך דגל ההזרמה ועדכון מונה הלקוחות הצופים.

# מקרי בדיקה

האפליקציה עברה סדרת בדיקות טרם הגשתה, כאשר היא נבדקה במסגרת רשת מקומית באמצעות localhost המחוברת לרשת האינטרנט והבדיקות בוצעו על מספר לקוחות במקביל.

בצד הלקוח בין היתר נבדקו החלונות השונים ותקשורת שרת-לקוח.  
בצד השרת בין היתר נבדקו ה-routes השונים, בסיס הנתונים והמחלקות ופונקציות העזר השונות.

הבדיקות מרוכזות בשתי קבצי python נפרדים הנמצאים בתיקיית הפרויקט:  
 [test\_client.py](../../VOD_Client/test_client.py) (נמצא בתת תיקייה VOD\_Client)  
[test\_server.py](../test_server.py) (נמצא בתת תיקייה VOD\_Server)

תוצאות הבדיקות מרוכזות בשתי קבצי HTML נפרדים הנמצאים בתיקיית הפרויקט:  
[Test Results - client-side.html](../../VOD_Client/Test%20Results%20-%20client-side.html) (נמצא בתת תיקייה VOD\_Client)  
[Test Results - server-side.html](../Test%20Results%20-%20server-side.html) (נמצא בתת תיקייה VOD\_Server)

# פערים במצב הקיים

האפליקציה **לא** תומכת בשלב זה ביכולות הבאות:

* זמינות לציבור:

האפליקציה עובדת (באמצעות **localhost**) בתצורה **מקומית בלבד**.

* שילוב כתוביות (הפעלה/נטרול כתוביות, מעבר בין כתוביות שונות):

האפליקציה מניחה כי הסרטים בשרת הם בעלי כתוביות מובנות (Hardcoded subtitles),  
כלומר קיימת שכבת-על (Overlay) של טקסט המוטבעת לצמיתות בסרט כך שהם נשארות גלויות ללא קשר להגדרות הצופה או לפלטפורמת הניגון.

**יש לציין כי VLC (ובפרט 'python-vlc') אכן תומכות בשילוב כתוביות ולכן יש מקום לשיפור.**

* שינוי שפת הממשק הגרפי:

הממשק הגרפי של האפליקציה תומך בשפה האנגלית בלבד.

* דירוג סרטים על ידי המשתמש:

דירוג הסרטים ניתן לשינוי על ידי תשאול בסיס הנתונים בצד השרת בלבד.

* הפעלת האפליקציה ללא בסיס הנתונים:

האפליקציה מוגבלת מבחינת הפעלה ללא בסיס הנתונים.

כרגע האפליקציה מניחה כי קיים הקובץ **movies\_database.db** בתיקייה **instance** שבתיקיית פרויקט

השרת ובפרט, קיים בסיס נתונים בשם **movies\_database** ובו טבלה בשם **movies**.

* היעדרות תיקיות assets:

האפליקציה מניחה כי קיימות בצד הלקוח ובצד השרת תיקיות **assets** ובהם כל "הנכסים" – הקבצים שניגשים אליהם במהלך הריצה באמצעות ה- **RESTful API** והם המיקומים היחידים בהם נמצאים הקבצים הרלוונטיים.

מיקומי הסרטים (קבצי 4MP) והפוסטרים שלהם (קבצי PNG) בתיקייה **assets** שבצד השרת תואמים את ערכי עמודות 'מיקום הסרט' ו-'מיקום תמונת הפוסטר' בבסיס הנתונים.

תיק הפרויקט (**project\_portfolio.docx**) נמצא בתיקייה שבצד השרת.

תמונת לוגו Word (**WordLogo.png**) נמצאת בתיקייה שבצד הלקוח.

* ממשק גרפי לאפליקציית צד השרת:

**הוספה/הסרה** של סרט לא נעשית על גבי טופס **הוספה/הסרה** גרפי, אלא על ידי שינוי תוכן תיקיית  **assets** שבצד השרת, במקביל לביצוע שינוי הרשומה בטבלה **movies** בבסיס הנתונים **movies\_database**.

כמו כן, **עריכת** פרטי סרט לא ממומשת כרגע ולכך יש להסיר את הסרט ולהוסיף אותו עם פרטיו הנכונים.

* אוטומציה באפליקציית צד השרת:

**הוספה/הסרה** של סרט נעשה באופן **ידני** כך:

* **הוספה/הסרה** של הסרט ותמונת הפוסטר שלו בתיקיית **assets** שבצד השרת.
* **הוספה/הסרה** של הרשומה בטבלה **movies** בבסיס הנתונים **movies\_database** על ידי שימוש

בפונקציות **remove\_movie\_data/insert\_movie\_data** שבקובץ **database.py**.

* הפעלת אפליקציית צד השרת באמצעות "דאבל קליק" על אייקון:

אפליקציית צד השרת ניתנת להפעלה על ידי הרצת הקובץ **setup.py** בלבד.  
קובץ זה מטפל בבדיקת קיום הספריות הנחוצות עבור השרת ('python-vlc', 'Flask' ו- 'Flask-SQLAlchemy') והתקנתם במידת הצורך, הרצת הקובץ **database.py** (היוצר את בסיס הנתונים)

במידה ובסיס הנתונים לא קיים והרצת הקובץ **server.py** (אפליקציית השרת).

# CD של הפרויקט

|  |  |
| --- | --- |
| מצ"ב CD המכיל את **"פרויקט שרת סרטים (VOD)"** (כולל קובצי ההתקנה והבדיקה של הפרויקט). |  |