חלק ג - פרויקט WEB

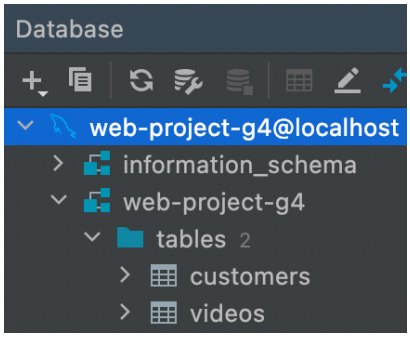
Avivit Ben Hayun 323010058
Eden Levi 315544114
Noa Elharar 318746146
Yarden Felz 311449706

קישור ל-git:
https://github.com/noae123/team4.git

תוכן עניינים

שימוש בפרויקט הבסיס הנתון במודל

הורדנו את הפרויקט המצורף במודל בשם Flask Skeleton Project
ועבדנו ע"פ המדריך המצורף במודל "הגדרת סביבת עבודה עם flaks"



חשוב!

איך לפתוח את העבודה בpycharm:
כדי לפתוח את העבודה שלנו יש לפתוח את pycharm. לאחר מכן לפתוח את file - >openfile . לחפש במחשב את תיקיית web-priject-g4 ולפתוח אותה, היא תיקיית flask שלנו. בתוכה יש קובץ בשם app.py יש להריץ את הקובץ בפיתון.

יצירת בסיס נתונים

יצרנו בסיס נתונים בשם web-project-g4

בבסיס נתונים זה יצרנו טבלאות בשם customers ו- videos.
טבלת customers מכילה את כל המידע על המשתמשים הרשומים באתר שלנו:

יצירת הטבלה התבצעה כך:

```
create table customers
(
    id        int auto_increment
        primary key,
    user_name varchar(30) not null,
    nickname  varchar(30) not null,
    password  varchar(30) not null,
    email     varchar(50) not null,
    constraint customers_id_uindex
        unique (id),
    constraint ck_email
        check (`email` like '%@%.%'),
   constraint ck_min_password
      check (length(`password`) >= 6)
```

	id	user_name	nickname	password	email
1	2	EDEN	ED77E77	4567890	E@LJDN.T
2	4	Aviv	Aviv123	E&T22/08/22	aviv12@gmail.com
3	5	Yarden	Yardi90	Aa123456	yar@gmail.com
4	6	Noa	Nuni66	Nn123456	Noa@gmail.com
5	7	Gal	Galim15/08	Gg123456	Gal@gmail.com
6	8	Don	dondon5	Don12345	don@gmail.com
7	9	Tali	Talic536	Tt12345	tal@gmail.com

```
);
```

	id_video	user_id	audio	image	sender_name	card_banner	color	shape	video_name	dir
1	1	2	<null>	<null>	eden	tami	red	triangle	var	<null>
2	3	4	<null>	<null>	Aviv	Tomi	green	squre	songi	<null>
3	4	5	<null>	<null>	Dan	Hili	yellow	circle	stars	<null>
4	5	6	<null>	<null>	Yotam	Roni	blue	rectangle	moon	<null>
5	6	7	<null>	<null>	Yossi	Avi	orange	star	sunshine	<null>

שדה id – מסוג int נוצר אוטומטית ומייצג את המספר הסידורי של כל משתמש, אשר ייחודי לו ולכן הוא גם מפתח ראשי בטבלת users.
שדה user_name-מסוג varchar(30), מייצג את שם המשתמש שנרשם/רשום לאתר, שדה זה גם הוגדר להיות מפתח ראשי, לכן לא יהיו 2 משתמשים בעלי אותו שם.
שדה nickname- מסוג varchar(30), מייצג את הכינוי של המשתמש באתר.

שדה password- מסוג varchar(30), מייצג את הסיסמא של המשתמש בעת כניסתו לאתר, הסיסמא אולצה באמצעות אילוץ בשם: ck_min_password להיות גדולה מ6 תווים, אחרת תופיע התראה למשתמש בעת ההרשמה הראשונית לאתר. שדה זה חייב להיות מלא לכן not null.

שדה email- מסוג varchar(50), מייצג את כתובת האימייל של משתמש, שעמה הוא נכנס לאתר. אילצנו את שדה זה בעזרת אילוץ בשם: ck_email להיות בפורמט מסוים של אימייל: עם סימן @ ו- '.' לסיומת .

אופן יצירת הטבלאות והכנסת הנתונים מופיע בקובץ שנקרא: sql.sql:

יצירת הטבלה users התבצעה כך:

```
create table customers(
    id        int auto_increment
        primary key,
    user_name varchar(30) not null,
    nickname  varchar(30) not null,
    password  varchar(30) not null,
    email     varchar(50) not null,
    constraint customers_id_uindex
        unique (id),
    constraint ck_email
        check (`email` like '%@%.%'),
    constraint ck_min_password
       check (length(`password`) >= 6)
);
```

הכנסת הנתונים לטבלת המשתמשים (users) התבצעה כך:

```
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (1, 'Card4You', 'card4u', 'CD123456', 'card4u@c4u.io');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (2, 'EDEN', 'ED77E77', '4567890', 'E@LJDN.T');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (4, 'Aviv', 'Aviv123', 'E&T22/08/22', 'aviv12@gmail.com');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (5, 'Yarden', 'Yardi90', 'Aa123456', 'yar@gmail.com');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (6, 'Noa', 'Nuni66', 'Nn123456', 'Noa@gmail.com');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (7, 'Gal', 'Galim15/08', 'Gg123456', 'Gal@gmail.com');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (8, 'Don', 'dondon5', 'Don12345', 'don@gmail.com');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (9, 'Tali', 'Talic536', 'Tt12345', 'tal@gmail.com');
INSERT INTO `web-project-g4`.customers (id, user_name, nickname, password, email)
VALUES (13, 'Roei', 'roei2107', 'Rr123456', 'Roei@gmail.com');
```

המשתמש הראשון שהוכנס לטבלה הוא המשתמש הראשי האתר: Card4You .
בעזרתו יצרנו את הקטלוג של הסרטונים.
יצירת הטבלה videos התבצעה כך:

```
create table videos(
    id_video    int auto_increment
        primary key,
    user_id     int           not null,
    audio       varchar(200)  null,
    image       varchar(200)  null,
    sender_name varchar(30)   null,
```


```
    card_banner varchar(2500) null,
    color       varchar(20)   null,
    shape       varchar(20)   null,
    video_name  varchar(250)  null,
    dir         bit           null,
    constraint videos_id_video_uindex
        unique (id_video),
    constraint fk_user_id
        foreign key (user_id) references customers (id),
    constraint ck_max_card
        check (octet_length(`card_banner`) <= 2500),
    constraint ck_max_sender
        check (octet_length(`sender_name`) <= 30)
);
```

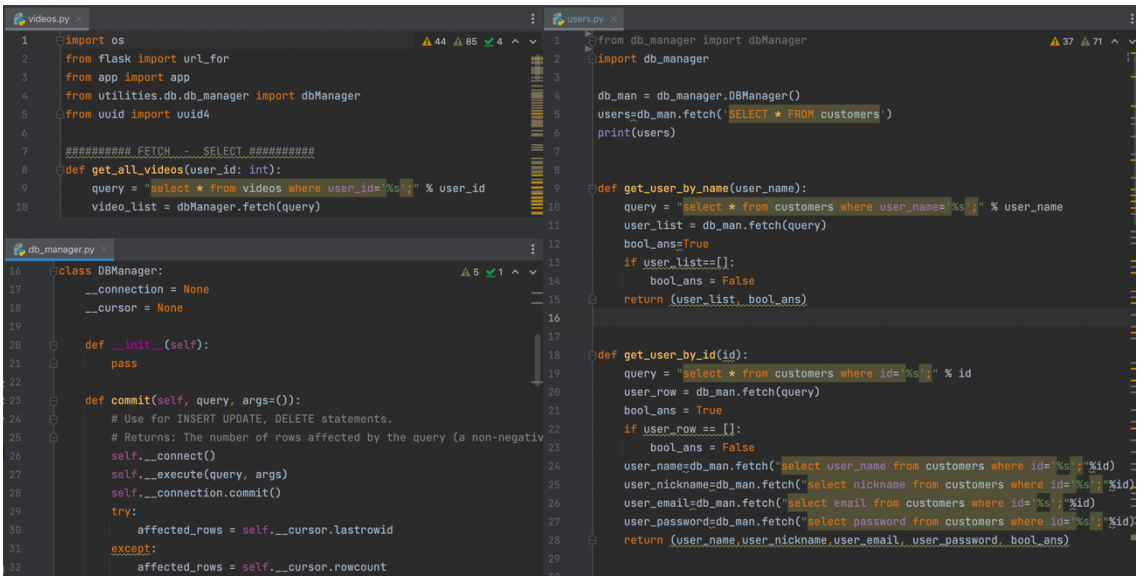
הכנסת הנתונים לטבלת הסרטונים (videos) התבצעה כך :

```
INSERT INTO `web-project-g4`.videos (id_video, user_id, audio, image,
sender_name, card_banner, color, shape, video_name, dir) VALUES (1, 1, '',
'happybdaycard.jpeg', 'card4u', 'Happy birthday, dear friend', 'pink',
'butterfly', 'Happy Birthday Card', false);
INSERT INTO `web-project-g4`.videos (id_video, user_id, audio, image,
sender_name, card_banner, color, shape, video_name, dir) VALUES (2, 1,
'wedding.mp3', 'wedding.jpeg', 'card4u', 'You are invited to our wedding', '',
'heart', 'Wedding Invitation', false);
INSERT INTO `web-project-g4`.videos (id_video, user_id, audio, image,
sender_name, card_banner, color, shape, video_name, dir) VALUES (3, 1,
'web.mp3', 'web.jpeg', 'team4', 'Thank you to all the web course teachers, for a
great experience! we really enjoyed the course and we learnt a-lot! :)',
'brown', 'heart', 'Thank you web team', false);
INSERT INTO `web-project-g4`.videos (id_video, user_id, audio, image,
sender_name, card_banner, color, shape, video_name, dir) VALUES (4, 6, null,
null, null, null, null, null, null, null);
INSERT INTO `web-project-g4`.videos (id_video, user_id, audio, image,
sender_name, card_banner, color, shape, video_name, dir) VALUES (5, 6, null,
null, 'noa', 'team4check', 'green', 'heart', null, null);
INSERT INTO `web-project-g4`.videos (id_video, user_id, audio, image,
sender_name, card_banner, color, shape, video_name, dir) VALUES (6, 5, null,
null, 'card4u', 'someblessinghere', null, null, null, null);
```


יצירת מחלקות

נתונה לנו מחלקת הבסיס db_manager שדרכה נתבקשנו לבצע את כל ההתנהלות מול בסיס הנתונים.
בנוסף יצרנו את המחלקות videos ו- users שמשתמשות במחלקה זו.
בכל מחלקה כזו יצרנו את השיטות הרלוונטיות לקבלת המידע שרצינו וזה יבתצע תוך כדי קריאה למחלקה db_manager שנקבל ממנה את הנתונים שאנו רוצים, תעבד אותם ותחזיר את התוצא הספציפית שביקשנו.

מחלקת users:



השיטות שכתבנו במחלקה users:

```
def get_user_by_name(user_name):
def get_user_by_id(id):
def create_user(name,user_nickname,user_password,user_email):
def update_user(id,user_name,user_nickname,user_email,user_password):
def delete_user(id):
def get_user_id_by_name_password(user_name,password):
def get_user_id_by_name(user_name):
```

get_user_by_name
פונקציה זו מקבלת כקלט את שם המשתמש(user_name), ומחזירה כפלט את כל הפרטים של המשתמש-כל השדות של המשתמש : שם משתמש, כינוי, סיסמא ואימייל ובנוסף היא מחזירה תשובה בעזרת משתנה בוליאני בשם: bool_ans אם הפונקציה הצליחה או לא.

Get_user_by_id
פונקציה זו מקבלת כקלט את ה id של משתמש מסוים ומחזירה כפלט את כל אחד מהשדות של משתמש מסוים לפי ה Id שהכנסנו: id, שם משתמש, כינוי, סיסמא ואימייל , בנוסף הפונקציה מחזירה משתנה בוליאני בשם: bool_ans אשר מציין אם הפונקציה הצליחה או לא.

Creat_user
פונקציה זו מקבלת כקלט את כל השדות הדרושים ליצירת משתמש חדש : שם משתמש, כינוי, סיסמא ואימייל . הפונקציה מייצרת את המשתמש החדש, מייצרת עבורו גם id חדש כפי שהגדרנו בעת יצירת הטבלה users בבסיס הנתונים ומכניסה אותו לטבלת users .
הפונקציה מחזירה כפלט את מספר השורות שהושפעו (משתנה בשם: last_row_effected) בבסיס הנתונים ובנוסף מחזירה את המשתנה הבוליאני : bool_ans אשר מציין אם הפעולה הצליחה או לא.

Update_user
הפונקציה מקבלת כקלט את כל השדות של המשתמש: id, שם משתמש, כינוי, סיסמא ואימייל. את השדות היא מקבלת מהאתר אשר בו מוגדרת ברירת מחדל עבור שדות שאין המשתמש רוצה לעדכן, לכן הפונקציה בנויה לקבל את כל השדות. הפונקציה משתמשת בפונקציה get_user_by_id על מנת לחפש את המשתמש שאותו רוצים לעדכן ולבדוק אם המשתמש קיים בבסיס הנתונים. אם המשתמש נמצא אז נעדכן אותו בבסיס הנתונים שלנו, אחרת נחזיר תשובה בעזרת משתנה:bool_ans שהפעולה נכשלה. הפונקציה מחזירה כפלט את המשתנה שעודכן (לאחר העדכון) בנוסף למשתנה bool_ans שכבר צוין קודם לכן.

Delete_user
הפונקציה מקבלת כקלט id של משתמש ומבצעת פעולת מחיקה מבסיס הנתונים באמצעות שאילתת DELETE לאחר מכן מתבצעת בדיקה להימצאות המשתמש הנ"ל בבסיס הנתונים, באמצעות שאילתה נוספת שהוכנסה למשתנה בשם: check_if_user_exist.
פונקציה זו מחזירה משתנה בוליאני bool_ans אשר מחזיר trueאם המשתמש נמחק, אחרת false.

Get_user_id_by_name_password
פונקציה זו מקבלת כקלט : שם משתמש וסיסמא ומחפשת בבסיס הנתונים את המשתמש על מנת להחזיר את ה id שלו. אם המשתמש שהוכנס לא נמצא הפונקציה תחזיר משתנה bool_ansשערכו false אחרת ערכו יהיה true. בנוסף הפונקציה מחזירה את ה id של המשתמש שהכנסנו.

Get_user_id_by_name
הפונקציה מקבלת כקלט שם משתמש כלשהו , מחפשת את המשתמש הנ"ל בבסיס הנתונים שלנו ומחזירה כפלט את הid של המשתמש, בנוסף הפונקציה מחזירה משתנה בוליאני בשם bool_ans אשר מציין האם הפעולה הצליחה או לא.

מחלקת videos:

המחלקה הזו קישור בין האתרים השונים לdb_mannager ומסד הנתונים. תפקידה לעשות את הקישורים בין סרטוני האתר בהם הלקוחות באתר משתמשים.

פונקציית get_all_videos:

הפונקציה מקבלת כקלט את id המשתמש בצורת int. ראשית, הפונקציה מחזירה את רשימת הסרטונים של המשתמש, היא מחזירה אותם כמילון שבתוכו יש מילונים כשכל מילון מייצג סרטון של המשתמש. את הפרטים החסרים של הסרטונים הפונקציה משלימה בעזרת קריאה לפונקציית generate_default_pararms. בנוסף, הפונקציה מחזירה האם הצליחה להחזיר לפחות סרטון אחד.

פונקציית generate_default_pararms:

הפונקציה מקבלת כקלט סרטון שזה מילון עם כל הפרמטרים של הסרטון, ופרמטר הקובע האם יש. להחזיר בצורה דפולטיבית את חלק מהטקסטים, משתנה זה הוא בוליאני ומאותחל להיות שלישי אם פרמטר כזה לא נשלח בקריאה לפונקציה. הפונקציה מחזירה סרטון בו החלקים החסרים והריקים של הסרטון שיקבלה יאותחלו להיות הפרמטרים הדיפולטיבים. בין הפרמטרים הגנריים: אם לא נשלח מוזיקה עם הסרטון, הסרטון יאותחל עם שיר גנרי של יומולדת שמח. אם לא נשלחה תמונה עם הסרטון, הסרטון יקבל רקע שחור מאחוריו. אם לא נשלח השם של השולח, ונבחר להשתמש בטקסטים הדיפולטיביים יירשם Sender Name כשם השולח אחרת, אם היה ריק יישלח סטרינג ריק. אם לא נשלח תאור הברכה, ונבחר להשתמש בטקסטים הדיפולטיביים יירשם Card Banner כתאור הברכה אחרת, אם היה ריק יישלח סטרינג ריק. אם הסרטון לא שמר צבע, הצבע הדיפולטיבי של האנימציה יהיה לבן. אם הסרטון לא שמר צורה, הצורה הדיפולטיבית של האנימציה תהיה עיגול. אם לא נשלח כיוון, הכיוון יהיה משמאל לימין, ואם אין כותרת לסרטון היא תהיה סטרינג ריק.

פונקציית get_video:

הפונקציה מקבלת כקלט id של סרטון וid של משתמש. הפונקציה מחזירה סרטון כמילון. את הפרטים החסרים של הסרטון הפונקציה משלימה בעזרת קריאה לפונקציית generate_default_pararms. בנוסף, הפונקציה מחזירה האם הצליחה להחזיר את הסרטון המבוקש.

פונקציית create_empty_video:

הפונקציה יוצרת סרטון ריק ומשלימה אותו לסרטון גנרי עם קריאה לפונקציה generate_default_pararms ועם בקשה שגם הטקסטים יהיו דפולטיביים. לבסוף הפונקצייה מחזירה את הסרטון הנוצר.

פונקציית saveFile:

הפונקציה מקבלת שם של קובץ, בודקת אם הסיומת שלו שייכת לסיומת של קבצי מוזיקה או קבצי תמונה, בפונקציה משנה את שם הקובץ כדי לא לדרוס קבצים אחרים ושומרת אותו בתיקייה המתאימה בתוך static, media_users. הפונקציה מחזירה את שם הקובץ החדש.

פונקציית create_a_new_video:

הפונקציה מקבלת את הפרטים מטופס יצירת סרטון חדש ואת id המשתמש שיצר אותו. אם המשתמש בעת היצירה העלה קובץ מוזיקה או תמונה הפונקציה תבצע קריאה לsaveFile

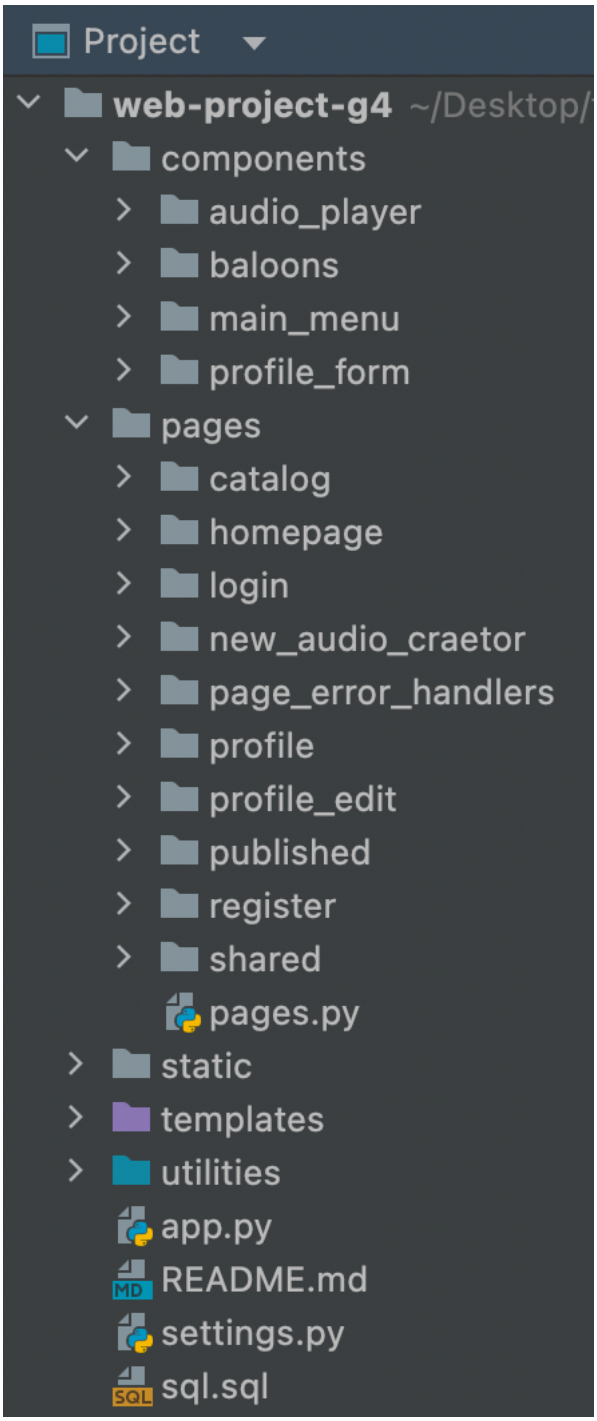
ותשמור את הקבצים, לאחר מכן הפונקציה תשמור את הסרטון החדש על מסד הנתונים ותחזיר את הid של הסרטון.

פונקציית update_video:
הפונקציה מקבלת את הפרטים מטופס עדכון סרטון ואת id המשתמש שיצר אותו. אם המשתמש בעת היצירה העלה קובץ מוזיקה או תמונה הפונקציה תבצע קריאה לsaveFile ותשמור את הקבצים, לאחר מכן הפונקציה תשמור את עדכוני הסרטון על מסד הנתונים.

פונקציית delete_video:
הפונקציה מקבלת id של סרטון ומוחקת אותו ממסד הנתונים.

פונקציית delete_all_videos:
הפונקציה מקבלת id של משתמש ומוחקת את כל הסרטונים שלו ממסד הנתונים.

קבצי הפרויקט:



חלוקת הקבצים:

Components >> Profile_from
Components >> balloons
Components >> audio_player
Pages >> login
Pages >> new_audio_creator
Pages >> profile
Pages >> profile_edit
Pages >> published
Pages >> register
Pages >> shared
Static >> lib
Static >> media_users
Utilities >> db >> users
Utilities >> db >> videos

דפים ראשיים:

App.py
הקובץ המרכזי של האפליקציה, את קובץ זה קיבלנו עם שלד העבודה. הרחבנו אותו כך שייצור blueprints מכל הדפים שיצרנו באפליקציה ועליהם נדבר בהמשך, בנוסף בקובץ זה יש את הקישור לכל הcomponents שיצרנו שגם הם כתובים בהמשך הדוח.

Base
דף הבסיס אותו קיבלנו עם שלד העבודה, שינינו את title האתר כך שבהתחלה יופיע שם החברה, 'Card4U', באזור הcss הראשי חיברנו לcss של רכיב האודיו כדי שקוד זה לא ייטען על כל סרטון חדש. שמופיע באתר, כי באתר יכולים להיות כמה סרטונים בכל דף. לאחר מכן הוספנו קישורים לקבצי lib שלp5 שעל תיקייה זו יצרנו את האפקטים של האנימציה של הסרטונים. לאחר מכן קישרנו לפונטים שאספנו בחלק השני של העבודה מגוגל. בheader של האתר הוספנו את הלוגו הלבן של החברה. והוספנו בלוקים לחלקי section, aside.

Pages:

Pages, new_audio_creator

דף audio new הדף עושה הרחבה של האתר הבסיסי base.html, האתר משנה את הtitle להכיל את השם 'Audio Creator', בנוסף האתר מתחבר לקובץ הcss המתאים לו.
בחלק הmain של האתר page_body_main_content הוספנו את הרכיב audio_player.html, כדי שיוצג סרטון אותו יערוך המשתמש. בחלק aside ישנו form מסוג post שמגיע בהתאם לפונקציית יצירת סרטון חדש או עדכון סרטון חדש, את חלק זה האתר מקבל מקבוץ הpy במידה והמשתמש נכנס לעדכון קיים הטופס יהיה מעודכן כך שפרטים כמו כיוון הכתב, שם השולח והברכה כבר יהיו רשומים באתר. בסוף האתר הוספנו את הscripts הראשון מחבר את האתר לקובץ הjs והscript השני מייצר אנימציה לסרטון המופיע באתר.
בקובץ הפייתון יש שני פונקציות, הפונקציה הראשונה היא ליצירת או עדכון סרטון של המשתמש, אם המשתמש לא מחובר היא תעביר את המשתמש לדף ההרשמה. אחרת אם לא נשלח לפונקציה id של סרטון היא תיצור סרטון גנרי חדש ריק, אם לא היא תנסה לחפש את הסרטון במסד הנתונים ואם היא לא מצאה סרטון היא תקרא לעצמה רקורסיבית עם סרטון ריק והודעה שהסרטון לא נמצא, לבסוף הפונקציה תיצור את הדף עם הסרטון כדי שהמשתמש יוכל לעבוד עליו.
הפונקציה השנייה לוקחת פרמטרים מסרטונים השייכים למשתמש 1 שלנו, שזה המשתמש של האתר card4u היא נותנת למשתמש במידה ומצאה את הסרטון המתאים לערוך סרטון חדש עם פרמטרים שמאותחלים מהקלוג. במידה והפונקציה לא מצאה את הסרטון היא תחזור לפונקציה הראשונה עם הודעה שהסרטון לא נמצא, אבל שיש סרטון חדש ריק אותו המשתמש יכול לערוך ולעבוד עליו.

Pages, catalog

את דף catalog קיבלנו מהשלד של פרויקט הבסיס. הדף עושה הרחבה של האתר הבסיסי base.html, האתר משנה את הtitle להכיל את השם 'Catalog', בנוסף האתר מתחבר לקבצי הcss והjs המתאים לו.
בחלק הmain של האתר page_body_main_content הוספנו את הרכיב audio_player.html, בלולאת for עבור כל הסרטונים של משתמש 1 שהוא המשתמש הראשי של האתר. לכל רכיב הוספנו שם לסרטון ומתחתיו סקריפט שמיועד לו כדי ליצור את האנמיציה כפי שהיא שמורה במסד הנתונים. בנוסף אם הצופה באתר רשום לאתר תופיע כפתור עריכה לסרטון שתוביל ליצירת העתק סרטון זה אצל המשתמש כדי שהוא יוכל ליצור ברכה חדשה מהסגנון של הברכות בקטלוג.
בקובץ הפיתון יש פונקציה שקוראת למסד הנתונים עם הid של המשתמש ומקבלת ממנו את כל הסרטונים של הקטלוג לבסוף היא מציגה את הדף.

Pages, homepage

את דף homepage קיבלנו מהשלד של פרויקט הבסיס. הדף עושה הרחבה של האתר הבסיסי base.html, האתר משנה את הtitle להכיל את השם 'Homepage', בנוסף האתר מתחבר לקבצי הcss המתאים לו.
בחלק הmain של האתר page_body_main_content הוספנו את הפרטים הסטטיים של אתר homepage מחלק 2, בנוסף אם המשתמש לא מחובר לאתר יופיע כפתור התחברות לאתר.
בקובץ הpy הוספנו אפשרות לשלוח עם החזרת האתר הודעה למשתמש.

Pages, shared

דף shared הדף עושה הרחבה של האתר הבסיסי base.html, האתר מתחבר לקובץ הcss המתאים לו.
בחלק הmain של האתר page_body_main_content הוספנו את הרכיב audio_player.html, כדי שיוצג סרטון ששותף עם הצופה. בסוף האתר הוספנו את הscripts המייצר את האנימציה לסרטון המופיע באתר.
בקובץ הפייתון יש פונקצייה המקבלת מכתובת האתר מספר של סרטון ומספר של משתמש ומנבה להוציא את פרטי הסרטון מבסיס הנתונים, אם נמצא הסרטון הוא יוצג לצופה באתר. אחרת, האתר יועבר להודעת שגיאה שהדף לא נמצא.

Pages, published

דף published הדף עושה הרחבה של האתר הבסיסי base.html, האתר משנה את הtitle להכיל את השם 'Published', בנוסף האתר מתחבר לקובץ הcss המתאים לו.
בחלק הmain של האתר page_body_main_content הוספנו את הרכיב audio_player.html, כדי שיוצג סרטון אותו המשתמש מפרסם. בחלק aside ישנו רשמנו למשתמש שזה הסרטון שהו הולך לפרסם, הוספנו כפתור להעתקת הקישור לדף הshared ומתחתיו כפתור לחזרה אם המשתמש ירצה לערוך את הסרטון. בסוף האתר הוספנו את הscripts הראשון מחבר את האתר לקובץ הjsn והscript השני מייצר אנימציה לסרטון המופיע באתר.
בקובץ הפייתון ישנם שלוש פונקציות, הפונקציה הראשונה מקבלת כקלט את מספר הסרטון שהמשתמש יצר, בודקת אם הסרטון נמצא במסד הנתונים, אם כן הדף יוצג למשתמש עם הסרטון שהוא יצר אחרת זה יגיע לדף page not found. הפונקציה השנייה היא במתודת post היא מקבלת id של הסרטון מעדכנת אותו במסד הנתונים וקוראת לפונקצייה הראשונה שמחזירה את האתר. הפונקצייה השלישית גם היא במתודת פוסט, היא יוצרת את הסרטון במסד הנתונים ולאחר מכן קוראת לפונקציה הראשונה שמחזירה את הדף עם הסרטון החדש שנוצר.

Pages, login

<u>צד לקוח-</u>
בעמוד זה מתבצעת כניסת המשתמש לאתר שלנו, כאשר הוא מכניס את שם המשתמש שלו ואת הסיסמה. קודם מתבצעת בדיקה אם בכלל שם המשתמש קיים בדאטה בייס ולאחר מכן מתבצעת בדיקה אם יש שם משתמש וסיסמא תואמים.
במידה והמשתמש אכן הקליד פרטים נכונים, החיבור יתבצע בהצלחה והמשתמש יובל ישירות לעמוד הפרופיל שלו.
במידה והמשתמש הכניס פרטים לא נכונים הוא יישאר בעמוד ה-login עד שיזין את הפרטים הנכונים.
בנוסף בעמוד זה ישנה אופציה של JOIN US וזאת למשתמשים שאינם רשומים עדיין ורוצים להצטרף לאתר. בלחיצה על כפתור זה הם יובלו לעמוד ה- register ליצירת חשבון חדש.
<u>צד שרת-</u>

```
def index():
```

פונקציה זו מחזירה אותי לעמוד ה-html של login

```
def try_login():
```


פונקציה זו מבצעת בדיקה האם שם המשתמש שהוכנס קיים בבסיס הנתונים שלנו, במידה וכן הוא מבצע גם בדיקה על הסמה שנכנסה, האם היא תואמת לסמה שרשומה גם בבסיס הנתונים ולאמת שאכן מדובר באותו משתמש. במידה וגם הסמה נכונה הוא מעדכן את כל ה-session של פרטי המשתמש שלנו שנכנס זה עתה לאתר ומעביר את המשתמש לעמוד הפרופיל שלו (profile.index)
במידה ומשתמש זה לא קיים בבסיס הנתונים שלנו או הסמה לא נכונה הוא יישאר בעמוד ה-login שם יוכל לנסות להזין את פרטיו מחדש או ללחוץ על כפתור ה-join us וליצור משתמש חדש בעמוד ה-register.

כאשר המשתמש מצליח להתחבר לאתר וממלא את הפרטים כמו שצריך אז ה-session['logedIn']=True
ובהתאם session['userId']=get_user_id_by_name(request.form['user_name'])[0] שזה בעצם ה-id של המשתמש המשויך לשם המשתמש שלו
Session['nickname'] = request.args['nickname']
Session['email']=request.args['email']
כל משך הזמן שהמשתמש מחובר באתר כלומר session['logedIn']=Trur כל שאר הנתונים מחוברים ל-session גם. כאשר המשתמש מבצע logout מהאתר נתונים אלה מתאפסים.

Pages, register

<u>צד לקוח-</u>
בעמוד זה מתבצע הרישום של משתמש חדש לאתר, כאשר הוא ממלא את הפרטים המבוקשים ליצירת חשבון.
במידה והמשתמש מילא את כל השדות המבוקשים ע"פ הדרישות (במידה ולא-מופיעה הודעה בהתאם בכל שדה) ושם המשתמש שהוא בחר לא קיים במערכת (שמות המשתמשים שלנו ייחודים) אז נאפשר את יצירת המשתמש הזה והוא יובל לעמוד הבית של האתר.
במידה ושם המשתמש קיים במערכת, המשתמש יישאר בעמוד ה-register עד למילוי הפרטים כנדרש.

<u>צד שרת-</u>

```
def index():
```

פונקציה זו מחזירה אותי לעמוד ה-html של register

```
def try_register():
```

פונקציה זו קוראת לפונקציית ה-

```
def create_user(name,user_nickname,user_password,user_email):
```

אשר מקבלת את כל הפרטים שהמשתמש מכניס, במידה והוא בחר שם משתמש שלא קיים במערכת, כלומר אין מניעה ליצור את המשתמש הזה האו יחזיר ערך בוליאני "אמת", ופונקציה זו תוביל אותנו לעמוד ה-login כדי לבצע כניסה לאתר.
במידה ויש מניעה ליצור את המשתמש הזה פונקציה זו תשאיר אותנו בעמוד ה-register עד לבחירת שם משתמש מתאים.

Pages, profile

<u>צד לקוח-</u>
בעמוד זה יופיעו הברכות שהמשתמש יצר בתוך המain לכל ברכה יש כפתור למחיקת הברכה וכפתור לעריכת הברכה. . בaside יופיע טקסט עם שם המשתמש שהתחבר. בנוסף, יופיע כפתור edit profile שיוביל לדף עריכת פרטי משתמש וכפתור log out להתנתקות.

<u>צד שרת-</u>

```
def index():
```


פונקציה זו בודקת שהמשתמש אכן מחובר ולאחר מכן תחזיר את profile.html ביחד עם כל ברכות הוידאו, שם המשתמש והID של המשתמש.

```
def logout():
```

פונקציה זו מנקה את הsession כלומר מוחקת את כל הנתונים בו ומחזירה לדף הבית.

```
def delete_a_video():
```

פונקציה זו מוחקת את קובץ הוידאו בעזרת פונקציה שבנינו במחלקת user - delete_video ומחזירה אותנו לעמוד הפרופיל.

```
def delete_account():
```

פונקציה שמוחקת את הנתונים מהsession, מוחקת את כל הסרטונים של המשתמש מהדאטה בייס ואז מוחקת את המשתמש מהדאטה בייס. לבסוף מחזירה לעמוד הבית.

Pages, profile_edit

<u>צד לקוח-</u>

בדומה לעמוד הprofile בתוך הmain יופיעו הברכות עם אפשרות למחוק ולערוך. בaside יופיעו תיבות עם פרטי המשתמש שממולאים באופן אוטומטי. המשתמש יוכל למלא במידת הצורך ואז ללחוץ על כפתור update profile לעדכון. כפתור נוסף שמופיע זה כפתור delete your user המקשר לפונקציה של מחיקת חשבון (הפונקציה כתובה בעמוד profile)

<u>צד שרת-</u>

```
def index():
```

הפונקציה בודקת אם המשתמש בsession אם לא מחזירה לדף הlogin ואם כן מחזירה את profile_edit.html עם כל פרטי המשתמש על מנת שנוכל להציג אותם בprofile.form

```
def update_profile():
```

פונקציה שמקבלת את הנתונים מהprofile.form שהם פרטי המשתמש ומעדכנת את הפרטים לתוך הדאטה בייס בעזרת פונקציה ממחלקת user [update_user]. לאחר מכן מחזירה לעמוד הפרופיל.

Pages, new_audio_creator

דף audio new הדף עושה הרחבה של האתר הבסיסי base.html, האתר משנה את הtitle להכיל את השם 'Audio Creator', בנוסף האתר מתחבר לקובץ הcss המתאים לו.
בחלק הmain של האתר page_body_main_content הוספנו את הרכיב audio_player.html, כדי שיוצג סרטון אותו יערוך המשתמש. בחלק aside ישנו form מסוג post שמגיע בהתאם לפונקציית יצירת סרטון חדש או עדכון סרטון חדש, את חלק זה האתר מקבל מקבוץ הpy במידה והמשתמש נכנס לעדכון קיים הטופס יהיה מעודכן כך שפרטים כמו כיוון הכתב, שם השולח והברכה כבר יהיו רשומים באתר. בסוף האתר הוספנו את הscripts הראשון מחבר את האתר לקובץ הjs והscript השני מייצר אנימציה לסרטון המופיע באתר.
בקובץ הפייתון יש שני פונקציות, הפונקציה הראשונה היא ליצירת או עדכון סרטון של המשתמש, אם המשתמש לא מחובר היא תעביר את המשתמש לדף ההרשמה. אחרת אם לא נשלח לפונקציה id של סרטון היא תיצור סרטון גנרי חדש ריק, אם לא היא תנסה לחפש את הסרטון במסד הנתונים ואם היא לא מצאה סרטון היא תקרא לעצמה רקורסיבית עם סרטון ריק והודעה שהסרטון לא נמצא, לבסוף הפונקציה תיצור את הדף עם הסרטון כדי שהמשתמש יוכל לעבוד עליו.

הפונקציה השנייה לוקחת פרמטרים מסרטונים השייכים למשתמש 1 שלנו, שזה המשתמש של האתר card4u היא נותנת למשתמש במידה ומצאה את הסרטון המתאים לערוך סרטון חדש עם פרמטרים שמאותחלים מהקלוג. במידה והפונקציה לא מצאה את הסרטון היא תחזור לפונקציה הראשונה עם הודעה שהסרטון לא נמצא, אבל שיש סרטון חדש ריק אותו המשתמש יכול לערוך ולעבוד עליו.

Components:

Components, profile_form

תבנית המשותפת לדף register ולדף profile_edit (זוהי תבנית לטיפול בטפס -form)בה ניתן למלא את פרטי המשתמש למטרת עריכה או למטרת הרשמה לאתר. ההבדל נעשה בעזרת בדיקה האם המשתנה update=true- משתנה זה נותן לנו אינדיקציה אם התבנית מוצגת בעמוד עריכת פרופיל.
לאחר מכן מעודכנים הערכים לפי העמוד בו מופיע התבנית.

```
{% if  update %}
    {% set value_user_name =user_name %}
    {% set value_Nickname = nickname  %}
    {% set value_Password = password %}
    {% set value_Email = email %}
    {% set value_submit = "Update Profile" %}
    {% set value_action = "/update_profile" %}

{% else %}
    {% set value_user_name ="" %}
    {% set value_Nickname = ""  %}
    {% set value_Password = "" %}
    {% set value_Email = "" %}
    {% set value_submit = "Register" %}
    {% set value_action = "/register/try_register" %}
{% endif %}
```

Components, baloons

קומפוננת הבלונים של האתר, החלטנו ליצור אותה כי היא משותפת לכמה דפים באתר: דף הבית, כניסה לאתר והרשמה לאתר.
היא מורכבת מדף עם תמונות הבלונים שמחובר לcss שתפקידו להראות בלונים עולים ויורדים באתר.

בקובץ הpy של הקומפוננטה הגדרנו blueprint כדי שאתרים אחרי יוכלו להשתמש ברכיב זה.

Components, audio_player

קומפוננת הנגן הראשי של האתר, הרכיב המרכזי באתר שלנו שנמצא בכמעט כל חלקי האתר והוא השירות שמציע האתר.
הקומפוננטה מורכבת מתיקייה של קובץ השמע והתמונה הגינרית, קובץ הhtml של הרכיב וקובץ הpy.
בגלל שהרכיב יכול לחזור באותו עמוד מספר פעמים החלטנו להפריד את קבצי הcss והjs שלו כדי שהאתר לא יירשום אותם שוב כשנציג באתר כמה סרטונים כאלה. רכיבים אלו נמצאים בתיקיית static הראשית בcss ובjs המשותפים לכל האתרים, בנוסף בתיקיית static הראשית נמצא קובץ הlib של תיקיית p5 בה הסרטון משתמש כדי לייצר את האנימציה, קבצים אלו הם קבצים סטטיים ותועדו בעבודה הקודמת.
קובץ הhtml : הרכיב צריך לקבל שני פרמטרים video_id שזה הid של הסרטון בטבלת video, ואת video_info אובייקט זה הינו מילון המכיל את פרטי הסרטון מטבלת videos עם הid של video_id. הרכיב עושה שימוש בפרמטרים הבאים: תמונת הסרטון במסד הנתונים, הכיוון של המילים בסרטון, שם השולח וההודעה שנשלחת עם הסרטון והמוזיקה המלווה עם הסרטון.
קובץ הpy בקובץ זה הגדרנו blueprint כדי שאתרים אחרי יוכלו להשתמש ברכיב זה.

Components, main_menu

רכיב דף הניווט אותו קיבלנו מהשלד של הפרויקט הבסיסי שינינו את הקישורים, כך שאם המשתמש לא מחובר יופיעו: דף הבית, קטלוג, דף הרישום לאתר ודף כניסה לאתר. אם המשתמש כן רשום יופיעו דפי במקום הרשמה וכניסה לאתר דף יצירת סרטון חדש ודף פרופיל של המשתמש.