Threat Model Information

Application Version: 1.0

Description: The miniYoutube is the first implementation of a video sharing website that will provide the following features for the users:

- Upload and delete videos
- Get video streams with desired quality
- Search available videos (filtering and sorting by some attributes)
- Display user profile and uploaded videos
- Some videos are private (only specific user) and some are hidden (not shown in search and on user page).
- Display view history
- Display, create and delete comments on video

Document Owner: Naghmeh Mohammadi Far

Entry Points

ID	Name	Description	Trust Levels
1	HTTPS Port	The miniYoutube website will only be accessible via TLS. All pages within the miniYoutube website are layered on this entry point.	(1) Not logged in user (2) Logged in user
1.1	Main Page	The main entry point for all users	(1) Not logged in user (2) Logged in user
1.2	Login Page	Users must login to their account to upload videos, visit their history, add/delete comment on videos, visit their profile and information	(1) Not logged in user
1.2.1	Login Function	The login function accepts user supplied credentials and compares them with those in the database.	(1) Not logged in user
1.3	Register Page	Users can create an account. Both those who don't have an account and who want to create their next account (with different name, purpose, etc.)	(1) Not logged in user (2) Logged in user
1.3.1	Register Function	The login function accepts user supplied credentials and does the checks (length of the password, not same phone/email address, etc.) then create an account and update the database	(1) Not logged in user (2) Logged in user
1.4	Search Entry Page	The page used to enter a search query to search a video with some filters and sorting	(1) Not logged in user (2) Logged in user
1.5	Video Display Page	This page is used to stream the video and users can set the quality of it to play.	(1) Not logged in user (2) Logged in user
1.5.1	Comment entry page	This part is for each video and any logged in person can add comments, delete and edit their previous comments. Also, all the comments will be presented in this page	(1) Logged in user

1.6	User Profile Page	This page consists of user's information	(1) Logged in user
1.7	User Uploaded Videos Page	This page consist of user's uploaded videos that both logged in and anonymous users can visit	(1) Not logged in user (2) Logged in user
1.7.1	User manage Video Function	User can upload a new video or delete one.	(1) Logged in user
1.8	User History Page	User can visit his watch/search/upload history in this page	(1) Logged in user
2	HTTP requests	It will use HTTP requests to retrieve images, ..	(1) Not logged in user (2) Logged in user
2.1	Req user profile images		(1) Logged in user
2.2	Req video preview images		(1) Logged in user
2.3	Req video stream playlist		(2) Logged in user
2.4	Req video stream chunks		(3) Logged in user

Assets

ID	Name	Description	Trust Levels
1	**MiniYoutube users**	**Assets relating to the website users**	
1.1	User Login Details	The login credentials that a user will use to log into the website	(3) Logged in user (4) Database Admin (5) Database read user (6) Database write user
2	**System**	**Assets relating to the underlying system.**	
2.1	Availability of the website	This video streaming website should be available 24 hours a day and work properly that means app servers should be alive all the time.	(3) Database Admin (5) Website Admin
2.2	Video Storage	Video storage should hold the videos and datas for a long time and they shouldn't change	(3) Database Admin (4) Website Admin
2.3	Ability to Execute SQL as a Database Read User	This is the ability to execute SQL select queries on the database, and thus retrieve any information stored in database	(3) Database Admin (6) Database read user
2.4	Ability to Execute	This is the ability to execute SQL.	(3) Database

	SQL as a Database write User	insert, and update queries on the database and thus have read and write access to any information stored	Admin (7) Database write user
3	**Website**	**Assets relating to the MiniYoutube website.**	
3.1	Access to the Database Server	Access to the database server allows you to administer the database, giving you full access to the database users and all data contained within the database.	(6) Database write user

Trust Levels

ID	Name	Description
1	Not logged in user	User who is Anonymous (didn't try to login) or had failed the login process.
2	Logged in user	User who successfully logged in.
3	Database admin	The database admin has read and write access to the database.
4	Website admin	The Website admin can configure the website
5	Database read user	The database user account used to access the database for read access.
6	Database write user	The database user account used to access the database for write access.

Data Flow Diagrams

Clarification:

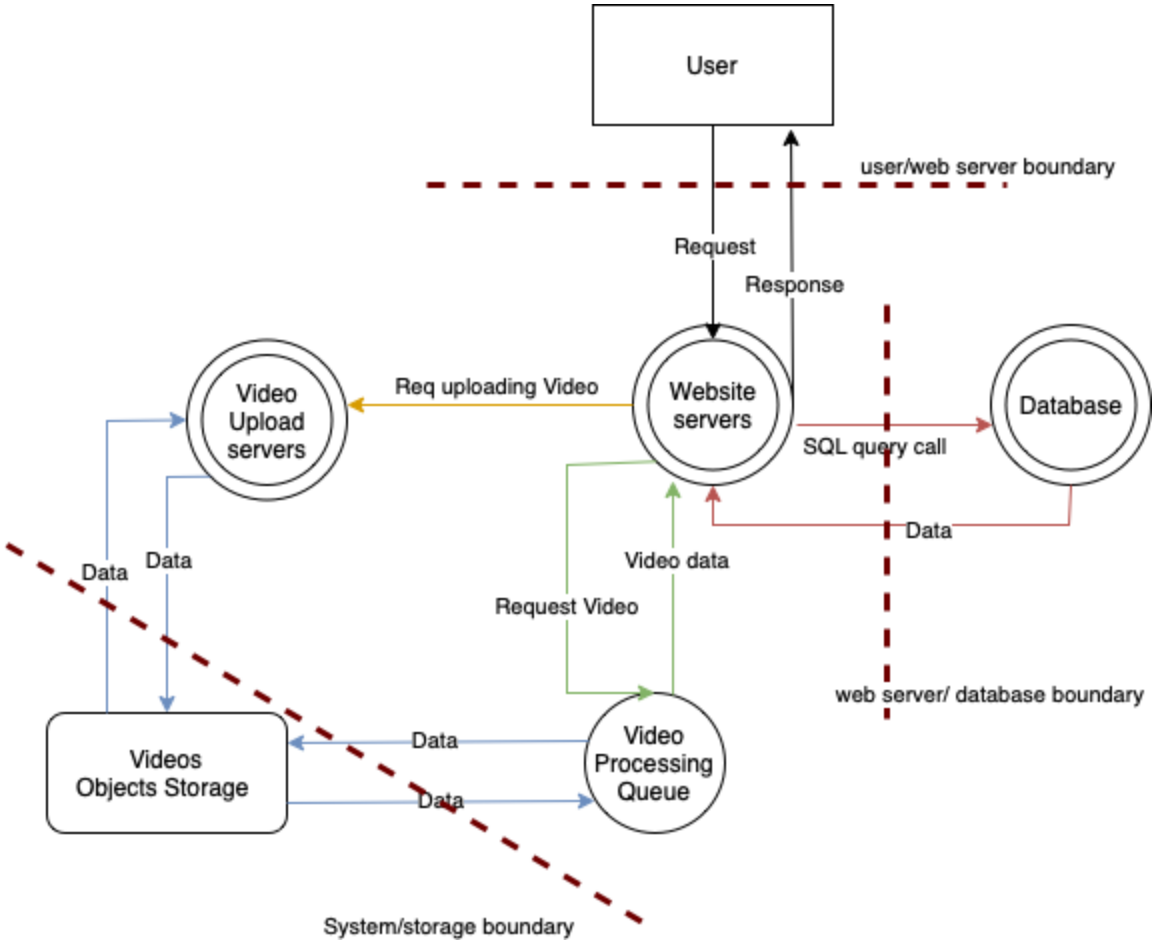
I assumed that my service is working the following way:

First client (user) connects to the website and sends a request to it. Then the main page is shown to her. In the main page there are different videos, which means, our web service should send browsing requests to the database. Database has the information about the videos (Name, id, owner, video private status, ...) Then the database sends the id of the videos (kind of a link to the actual storage of these objects) to the web server. After that, the web server requests objects from the video processing queue (with the requested quality). Video processing queue asks for the video from the storage and returns it back to the web server.

If the user wants to upload a video, the web server will send the data of this object (Name, owner, ..) to the database and connect to the video uploading server. Video uploading server will be responsible for storing the object in storage.

User sign-in, login and managing comments the credential is also possible with contacting the database.

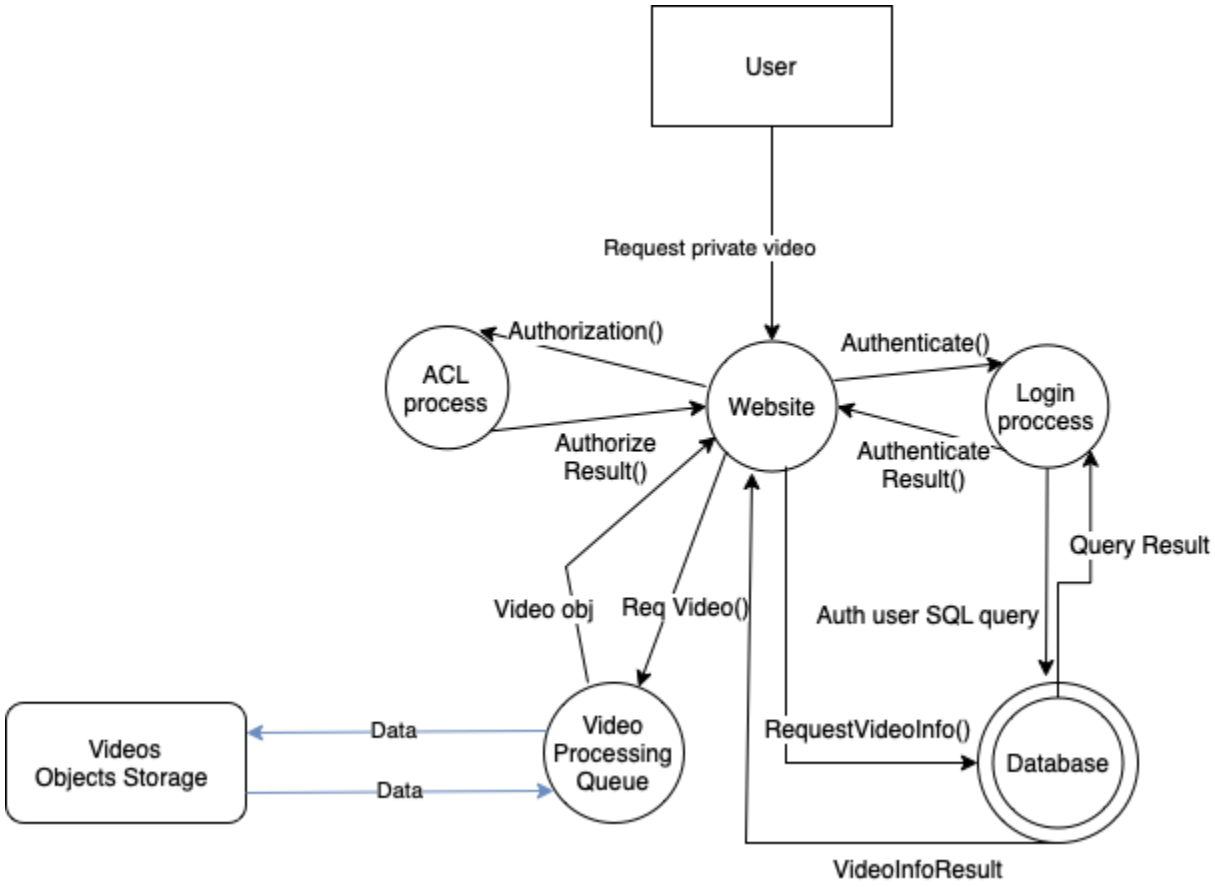
The following diagram will support the data flow of the desired system:


User
user/web server boundary
Request
Response
Video Upload servers
Req uploading Video
Website servers
Database
SQL query call
Data
Data
Data
Video data
Request Video
web server/ database boundary
Videos Objects Storage
Data
Data
Video Processing Queue
System/storage boundary

Use Case 1:
When a not logged in user has a link to a private video and wants to open it.

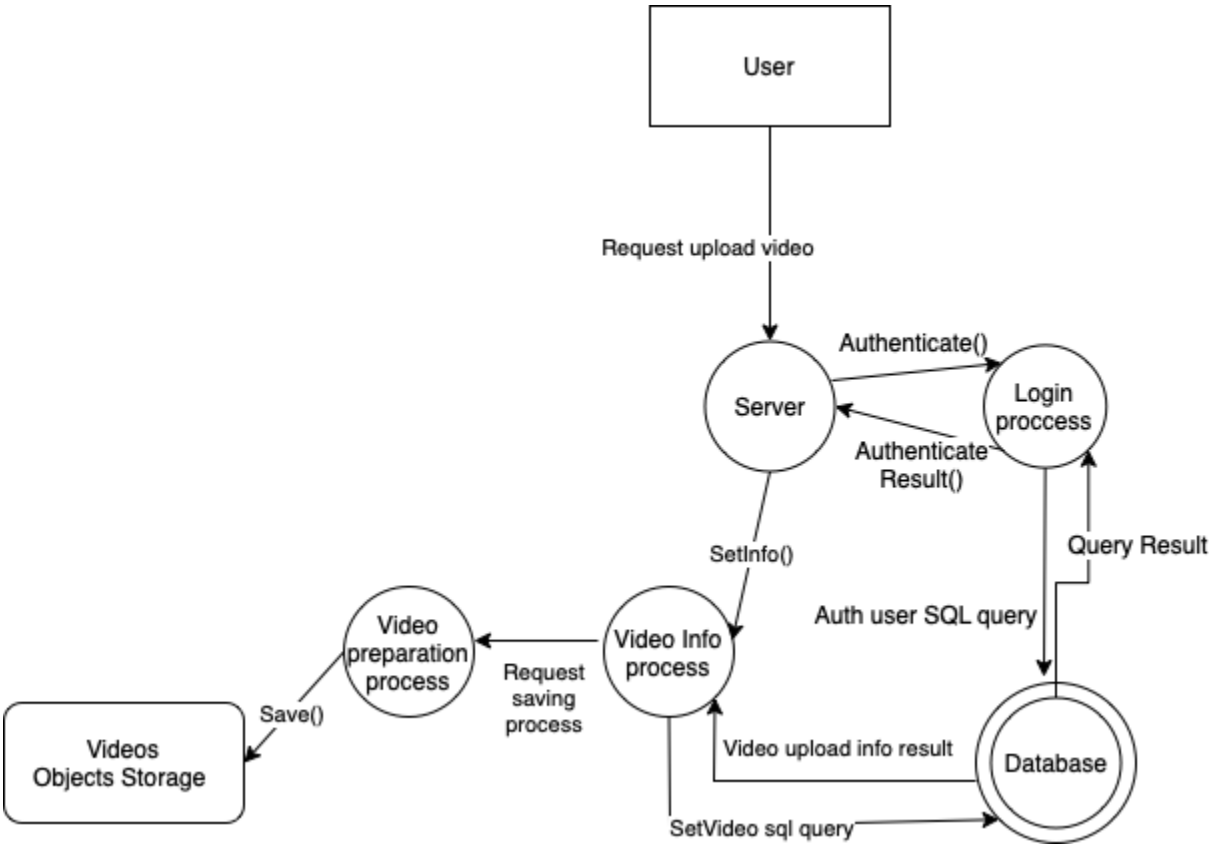
- Authenticate user -> login process
- Authorize user if he has access rights to the private video
- If yes, request for the video through database
- Ask for video with the ID and info the database gave us, from Video processing queue

P.s: with the website I ment, the server responsible for the backend of the site.

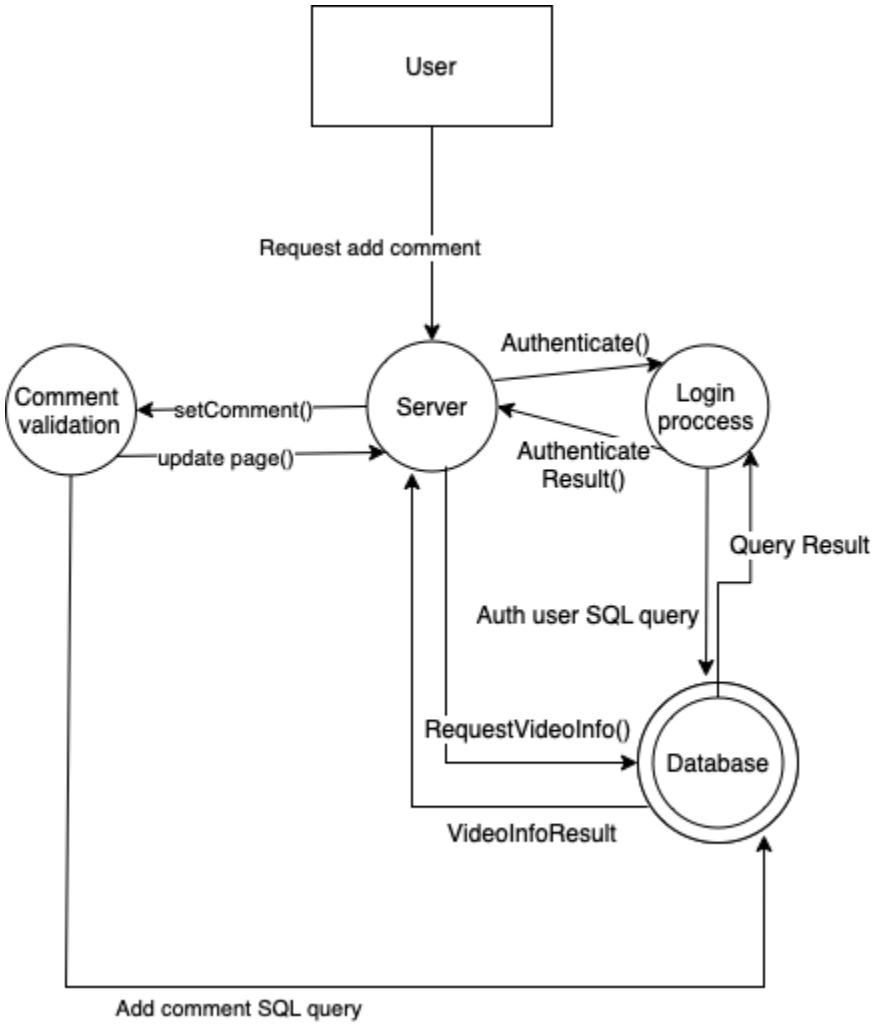


Use case 2:
An Unlogged in user wants to upload a file.

- First she should login
- Then the server will accept the info from the user, process and check the info the user entered (Name, content, ..)
- Then video preparation for storing starts



Use case 3:
An Unlogged in user wants to comment on a video.



Determine threats:

Asset	Category	Threat	Vulnerability	Score	Countermeasure
User Login Details	Information disclosure	User credentials are exposed and obtained by an attacker	-Passwords are storing in plain text -Website is open for XSS and SQL injection - Bruteforce/Dictionary attack	6.5	-Use a more secure way of storing the passwords - strong password policies, limited number of attempts -Sanitize data inputs
Availability of the website	Denial of service	The website is not responding	-Doesn't block ips after multiple requests in row	7.5	-Block ips who are accessing the website for more than 5 times in short time, for about 15 min -Have blacklist, whitelist for ips
Video Storage	-Tampering -Denial of service	Data is changed there or the storage is not available	-Does not check if request is from authorized user or not.	8.8	-Check who is contacting to the storage -Have duplicated servers
Ability to Execute SQL as a Database Read User	Information disclosure	All the website information are exposed and obtained by an attacker	-Plain text password for connecting to the database - Open access for XSS and SQLi - Insecure functions used to connect to the database	7.5	-Use a secure way of connection (functions, passwords, ..) -Input Sanitization

Ability to Execute SQL as a Database write User	-Tampering -Elevation of -Privilege -Repudiation	-Data will change in database - Attacker can escalate his privileges - Attacker will do everything he wants and delete all the traces.	-Plain text password for connecting to the database - Open access for XSS and SQLi - Insecure functions used to connect to the database	7.5	-Use a secure way of connection (functions, passwords, ..) -Input Sanitization
Access to the Database Server	Denial of service -Spoofing -Tampering	Attackers will make the database unresponsive and change the data there. Attacker send emails to gain access to the database admin account	-Does not check if request is from authorized user or not. - Human fault	8.0	-Check who is contacting to the storage -Inform the admins about threats