

CRCs

Class Name: Live Chatroom	
Parent: None Sub: None	
Responsibilities <ul style="list-style-type: none">- Enables chatting in the stream room- allows streamer to control the live chat	Collaborators <ul style="list-style-type: none">- Users- Database

Class Name: Users	
Parent: None Sub: Viewer, Streamer	
Responsibilities <ul style="list-style-type: none">- Allows user activities- Gives user information to classes that needs them	Collaborators <ul style="list-style-type: none">- Live Chatroom- Database- Social Media

Class Name Streamer	
Parent: User Sub: None	
Responsibilities <ul style="list-style-type: none">- Allows streaming setup- Allows giving selected viewers roles	Collaborators <ul style="list-style-type: none">- Live Chatroom- Database- Social Media- Stream

Class Name Viewer	
Parent: User Sub: None	
Responsibilities <ul style="list-style-type: none">- Allows subscribing to a streamer- Allows login and access streaming rooms	Collaborators <ul style="list-style-type: none">- Live Chatroom- Database- Social Media- Stream

Class Name Social Media	
Parent: None Sub: Direct Message	
Responsibilities <ul style="list-style-type: none">- Interchange post data with Database- Allows user to post, reply, comment, like	Collaborators <ul style="list-style-type: none">- Database- Users

Class Name Database	
Parent: None Sub: None	
Responsibilities <ul style="list-style-type: none"> - stores User data - stores Social Media data - allows data access from other classes (e.g. Live Chatroom, Login) 	Collaborators <ul style="list-style-type: none"> - Social Media - Live Chatroom - Users - Login - Register

Class Name Login	
Parent: None Sub: None	
Responsibilities <ul style="list-style-type: none"> - gives permission to certain user activities - links a client to a User 	Collaborators <ul style="list-style-type: none"> - Database - Users - Stream

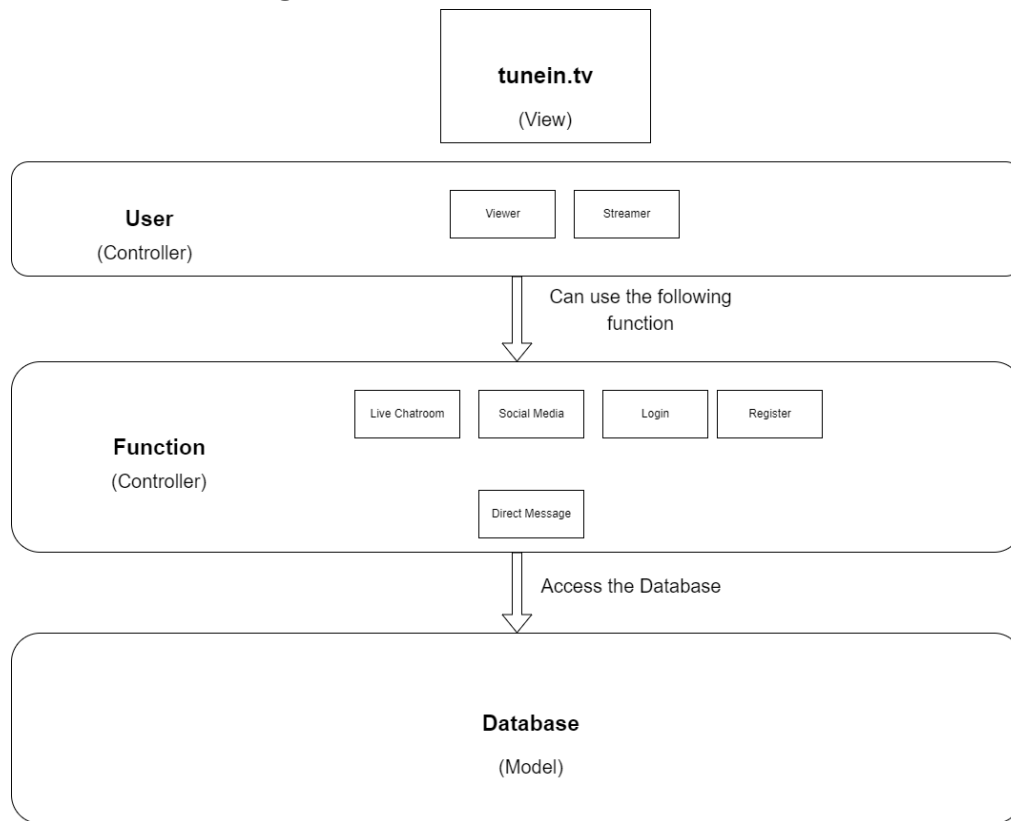
Class Name Register	
Parent: None Sub: None	
Responsibilities <ul style="list-style-type: none"> - generates new User - insert new user info in database 	Collaborators <ul style="list-style-type: none"> - Database - Users

Class Name Direct Message	
Parent: Social Media Sub: None	
Responsibilities <ul style="list-style-type: none"> - allows direct message between two users - stores direct message in database - notifies the user receiving the DM 	Collaborators <ul style="list-style-type: none"> - Database - Users

Description of System Interaction with the Environment

The system is expected to run on a PC with a widely accepted OS like Windows, macOS and Linux. We do not expect full support for mobile devices. The machines should have an active network connection for new browsing. The client machine should have a modern web browser such as Chrome or Firefox. The server machine should have python 3.10.2 and Node.js runtime installed, and have the environment set up according to the instructions in README.md included in the project.

Architecture Diagram



User

- Chat -> Live Chatroom
- Send -> Direct Message -> User
- Post, Reply, Comment -> Social Media

Viewer

- Subscribe -> Streamer
- Watch -> Stream

Streamer

- Setup -> Stream
- Assign roles -> Viewer

Database

- Store -> Users
- Store -> Social Media

- Give user Information -> Live Chatroom
- Give user Information -> Login

Register

- info put in -> Database
- Create -> User

System Decomposition

Model: database

View: frontend website / html pages

Controller:

The system is separated into two parts, one focusing on stream and the other focusing on social media. Both parts have their data stored in the database and are connected through Users. Users are created by Register, after which the user data are stored in the database. When login the user passes authentication, after which their clients are linked to the user using browser cookie.

In the streaming part, depending on which subclass of Users, they can either set up a stream in their room or join a room that is active. They can also chat in that room's live chatroom. Streamers have many powers in their room, including assigning special roles exclusive to the room to viewers.

In the social media part, users can make posts, do activities related to posts, or send a direct message to another user.

Viewers can subscribe to Streamers, and they can follow other users to receive notifications on their social media updates.

Error Handling:

Invalid inputs when logging in result in authentication failure, in which case the client would not be able to log in and have access to user activities.

A stream error causes the streamer to disconnect from their stream room, where the stream would end. The room will not be shut down.

We expect the database to be stable without error since all database insert/query are done in the controller.