

**SE/EE/CPRE 492**

**02/28 - 03/12**

**Project Title: Economic Home Security System**

**Group No: 42**

**Client/Advisor: Goce Trajcevski**

# **Bi-weekly Report 4**

**Team Members:**

Lucas Jedlicka - Lead Engineer DevOps

Uma Abu - Frontend



Merin Mundt - Frontend

Kamini Saldanha - Backend




Sohum Sawant - Backend




Andrew Tran - Backend








Open 5 +


- Create a wiki #70
- Status Report 5 - 04/02 #44
- Investigate memory requirements of YOLOv3 object detection. #10
- Write CRUD Tests for Simple API #49 
- Spike: Web application to stream data video from the phone. #2 







To Do 8 +

- Video Streaming in Backend #54 
- Object Detection on Backend #83
- Motion Detection on Backend #82
- Endpoint exposing all streams for a given userid #79
- View clips on client #77
- Personal user interface #72 
- Send streams on frontend #71
- Use uWSGI to deploy python instead of the development server #65 

▶ **Doing**   7 

- Peer demo/video  
#86  Wednesday 
- Store clips on server  
#76 
- Video Streaming in Frontend  
#50 
- CI/CD for backend  
#66 
- Status Report 4 - 03/12  
#43
- Endpoint for removing user's clip  
#81 
- Endpoint for adding user's clip  
#80 

▶ **Closed**  66

- Add test clips to media endpoint  
#84 
- Status Report 3 - 02/27  
#42
- Connect backend to database  
#60 
- Status Report 2 - 02/13  
#41
- User Auth with Tokens  
#55 
- Video Streaming in Backend  
#39
- Create endpoint for notifications  
#69 
- Get REST API Running on Server  
#57
- Force HTTPS for all connections via NGINX  
#67 
- Connect Django REST Project with External Database  
#45 

**Bi-weekly Summary:** We have added user authentication throughout the API and made necessary changes to accommodate that on the frontend. The frontend has been able to view clips directly from the server's file system and display it. Progress has been made for streaming data between client and server through peer-to-peer connections and websockets. WebRTC/aiortc are some of the tools we are using to do so. We were able to refactor the Django project as well and are prepared to continue adding functionality to the project.

### **Past Iteration Accomplishments:**

**Lucas Jedlicka:** Formed milestone activities and sorted out who desired which tasks. Researched Coturn, an application required for signaling and routing of WebRTC streams. For testing it appears the public servers will work, however in the internal ISU network, the google server couldn't determine our public IP (there isn't one). I've made a request to ETG to see if it's acceptable to host one or if they have one setup already for their VoIP system. I researched sending emails within Python, and found a fault in my earlier expectation. Mail sending will work from the server if we point to ISU's SMTP.

**Uma Abu:** Looked into webrtc on the frontend. I read the documentation and with some help of tutorials online, I was able to implement peer to peer communication in between two clients on the react application. After doing this, I started looking for ways to send data between two clients that are not on the same machine. I realized I needed a web socket so I implemented a websocket and got that working. I was able to achieve video streaming between two node clients on different computers. The next step was to do the same streaming but between a server and the react client. I looked into webrtc but did not find any examples of a server written in python that connected to a react client. So I started looking into aiortc and found a server written in python that implemented webrtc/aiortc and fetched streams from a js client. This was a good starting point. I started working on trying to refactor the python server to fetch the streams from a websocket connection rather than fetching it from a client that the socket is running.

**Merin Mundt:** These past two weeks I have been focusing on getting streams and clips to show up and be retrieved from the back end. I worked on retrieving a URL and then using that URL to display on the clips page of the application. Currently have up to 4 videos showing up via URL but am working on getting many to show up with time stamps.

### **Kamini Saldanha:**

These past two weeks my main focus was getting the web sockets component of our project working. I have been working with experimenting with the library WebRTC/aioRTC. I have been working to understand how the stream is sent between the client and the server. My main priority has been figuring out how to save this stream to the filesystem after it has finished recording.

**Sohum Sawant:** These past two weeks have been spent working on setting up a prototype server to receive and redistribute video streams from our client side. In order to accomplish this, I looked into different python libraries associated with WebRTC. The library that I decided to go with is called aiortc. The next step was to create an endpoint which could establish a peer to peer connection with a client. I looked into both a websocket connection and also into peer-to-peer mapping. Currently, my prototype server uses peer-to-peer mappings but the plan is to move to a websocket solution.

**Andrew Tran:** Researched possible libraries for communication and sockets for python and ran into aiortc. Fixed duplicate file issues on backend, there were many unnecessary files that could cause problems running the project. Was able to get the project working locally with the new database in order to refactor the project into separate apps for modularity and ability to split work without having a lot of merge conflicts. Created a new user for the database that works with Django both locally and on the server. User authentication is now implemented throughout the API and all the changes necessary were made to use the Django user auth model. The endpoint for getting clips given a user id has been added as well.

**Pending Issues (optional):**

**Lucas Jedlicka:** ETG email about STUN.

**Uma Abu:** N/A

**Merin Mundt:** N/A

**Kamini Saldanha:** Still figuring out how to create an mp4 file with the stream to be able to save that in the filesystem on the server.

**Sohum Sawant:** N/A

**Andrew Tran:** Lot's of backend dependency errors.

**Individual Contributions:**

<b>Name</b>	<b>Contribution</b>	<b>Biweekly hours</b>	<b>Total hours</b>
<b>Uma Abu</b>	<ul style="list-style-type: none"><li>• Created peer to peer connection between two clients on the frontend</li><li>• Implemented a signalling server to send data between the clients</li><li>• Implemented a websocket to send data between two clients that are on different computers</li><li>• Worked on the signalling server in python to see if it can fetch stream from web socket</li></ul>	<b>13</b>	<b>36</b>
<b>Lucas Jedlicka</b>	<ul style="list-style-type: none"><li>• Researched Coturn</li><li>• Tasked out milestone for basic scenario</li><li>• Put out fires caused by my poor CD for python</li><li>• Rewrote python docker container deployment</li></ul>	<b>12</b>	<b>36</b>
<b>Sohum Sawant</b>	<ul style="list-style-type: none"><li>• Created a prototype server to receive and redistribute video streams from our front end.</li><li>• Implemented asynchronous functionality in our stream server to handle connections opening and closing</li><li>• Did research on the aiortc library</li></ul>	<b>13</b>	<b>46</b>
<b>Merin Mundt</b>	<ul style="list-style-type: none"><li>• Created clips webpage</li><li>• Retrieved URLs for clips</li><li>• Clips showing up on clip page</li><li>• Made presentation for peer presentation video</li></ul>	<b>8</b>	<b>40</b>
<b>Kamini Saldanha</b>	<ul style="list-style-type: none"><li>• Created a prototype server to try to understand how the streams are sent between the</li></ul>	<b>12</b>	<b>44</b>

	<p>client and server</p> <ul style="list-style-type: none"> <li>• Tried to create and save a basic txt file on local computer and server</li> <li>• Trying to figure out how to create a mp4 file from the stream and save that locally and then on the server</li> </ul>		
<b>Andrew Tran</b>	<ul style="list-style-type: none"> <li>• Removed bad files in backend and refactored the project.</li> <li>• Found aiortc.</li> <li>• Did some code cleanup.</li> <li>• Added user authentication to the API.</li> <li>• Added endpoint to get all clips for a user.</li> </ul>	<b>12</b>	<b>36</b>

**Plans for the upcoming iteration:**

**Lucas Jedlicka:** Setup TURN/STUN server if ETG doesn't have a host, and if we're lucky, begin vision processing in the backend.

**Uma Abu:** Try to find another way to send video streams to a server if aiortc does not work.

**Merin Mundt:** Getting all clips showing up from the backend. Work on personalizing the user experience.

**Kamini Saldanha:** Help work on getting streams to be sent between client and server as well as sending emails as notifications.

**Sohum Sawant:** Move from peer-to-peer mappings to a websocket solution to handle incoming and outgoing connections from our server.

**Andrew Tran:** Work on fixing errors and look into adding motion detection to the server.

**Summary of Advisor Meeting:**

Discussed task decompositions and milestones. Focus on getting our simple scenario implemented. We also discussed creating different folders for sections of the final report in order to note things down and compile them later rather than relying on memory in the future. Considerations: reactivity to events in our system, websockets, and the expansion of use to someone you trust (like a family member) who can use the system.