**Interim report**

Project title:

**Remotely controlled advertisement unit**

Group members:

Arbaz Ahmed (59008)

Muhammad Ammar (59051)

Mohammad Hamza (58998)

Monis Azhar (59485)

**[S.M.Ammar Ali (59051)]**

**Start**

**Overview:**

This project is a remote desktop app. Client and server correspondence is at the introduce of organization provisioning over the Internet. Client sends request to multiple servers with the help of middle ware. Middle ware control multiple servers dynamically and servers usually wait to accept invitation from client side with the help of middle ware, maintaining connectivity or handshaking. Client connect to servers and provide ads list to play on servers. Sockets is introduce for basic communication between client and servers. Clients can run ads priority base remotely on servers.

Design of our project is complete of both side’s client and server until now. Back end implementation will complete in coming weeks. Project is divided into three parts client, server and middleware. We will submit project on three releases on Github. Work is assign into group members given below:

* Ammar and Arbaz working on client application.
* Monis working on middleware application.
* Hamza working on Server application.

**End**

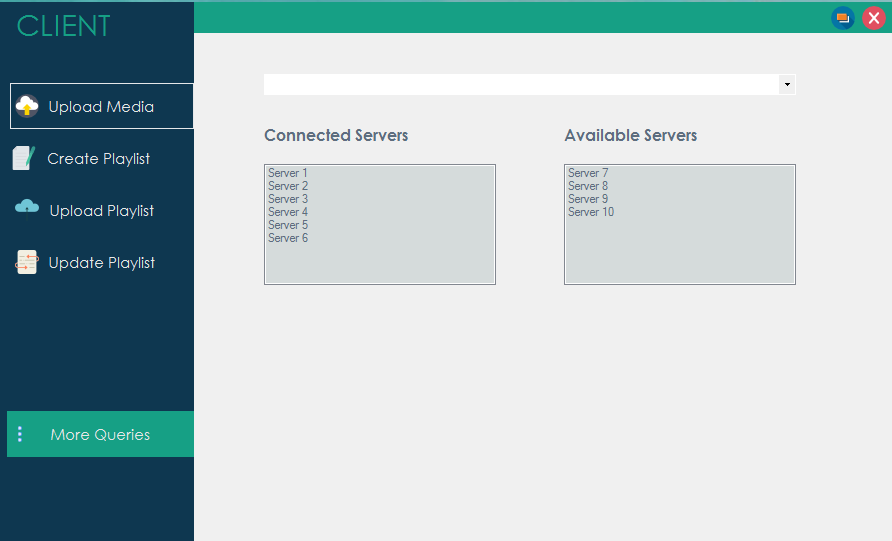
**[S.M.Ammar Ali (59051)]**

**[Arbaz Ahmed (59008)]**

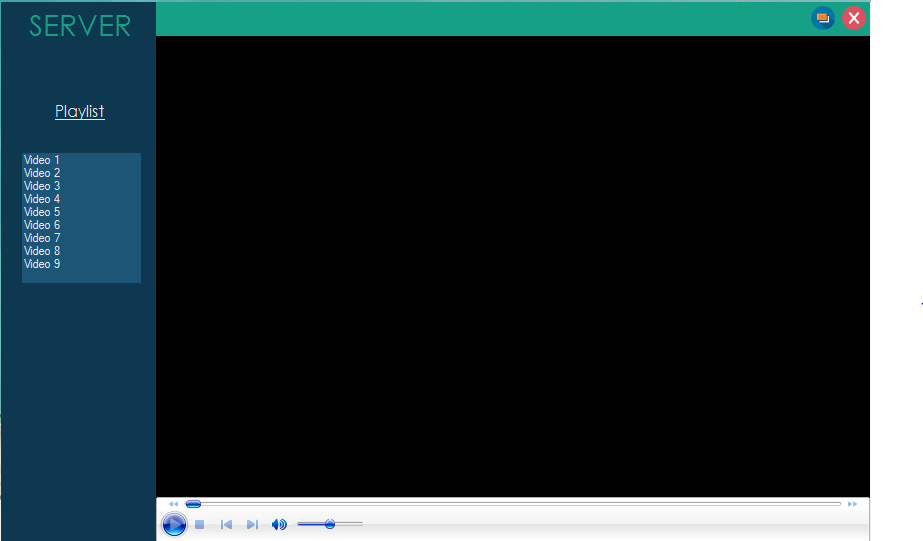
**Start**

**Features:**

* Client can upload Media
* Can create playlist
* Can upload playlist
* Can update playlist
* Client will show connected servers.
* Client will also show available servers.



* Server shows Playlist created, uploaded and updated from client side.
* Streaming ads on server from playlist with respect to time duration.



**Planning:**

We will using Tcp client and Tcp server for middle ware so it will work as distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients.

A server works as playlist streamer for several clients, that will handle playlist requests. Our client interface is connects to the server to provide playlist. The client provide a user interface that allows users to carry out actions. It forwards these requests to the server, which carries out the action and returns a response.

We will also setup FTP server on middle ware for the transfer of computer files between a client and server on a computer network. FTP is built on a client server model architecture using separate control and data connections between the client and the server.

**End**

**[Arbaz Ahmed (59008)]**