**System Architecture and System**

**Design**

**Architectural Styles:**

Our system utilize a three-tier architecture system and consists of 3 layers: presentation, application and data tiers. Our presentation layer is fundamentally residing in the website User Interface (UI) which allows the user to interact with our application through command-based components to enter the input and accept the output. Then comes our application tier which contains the logical operations and data access, for instance a User searches for specific field’s experts then our Selection from database command resides in this layer. Finally, our data tier consists of our database where our information is stored and retrieved.

These three tiers are separated from each other to allow for encapsulation and data abstraction. We want each tier to hide its usage from implementation and to preserve the integrity of our data. We also want to reduce the overall complexity of our system. However, each tier must maintain a sufficient level of communication and be able to retrieve needed data from each other. In common scenario for our system, our application layer may request information from the data tier. It then processes this information and returns it to the presentation tier in response to the user request. A visual diagram was provided in our earlier stage of planning in the section titled System Architecture Diagram.

**Identifying Subsystems:**

Our software is designed around four primary subsystems. The core subsystem is the

UI Subsystem, responsible for interfacing with the user and other subsystems. The Data

Logging Subsystem saves users authentication data and sessions data questions’ post data.

of this data. The mail system we use to notify users on sessions reservations and confirmation. Finally, the Chatting subsystem including (Video/Audio/Texting).

**Persistent Data Storage:**

Since mongodb is the database supported in MEAN stack web development stack, we used mongo as our database. Mongodb provides an interface between the server side (Node.js) and the database (mongodb) called Mongodb-Node.js driver which offer predefined methods to deal with the data in the collections.

**Network Protocol:**

The Chatting subsystem implemented on HTTP Live streaming to provide real-time chatting between the users and a client-side API WEB API to handle the sockets and the connection in peer-peer connection.

