The company XtremeSecurity has grown tremendously in the last five years expanding its points of presence throughout the world. XtremeSecurity acts a contractor for a large number of governmental and military organizations. The company has a large number of officies located in many countries and therefore requires various collaboration tools that allow the employees to communicate from different geographic locations. In particular, the employees find chat tools such as Skype and Yahoo Messenger particularly useful. Because XtremeSecurity offers security services they are offen in possession of confidential information from their customers. This makes them very paranoid about security breaches where 3rd parties may access the confidential data. As a result, they don't trust the regular chat tools and would rather make their own from scratch (no 3rd party components/libraries other than what follows with standard compilers/interpretors).

XtremeSecurity would like that your group builds an enterprise chat system for them. The chat system must support text messages of arbitrary length (up to 4KB), but it is not required to provide any audio or video features. A central database is required to store user information metadata associated with each sent or received message, so that every user can see the date and time a message was sent as part of this conversations, the name of the sender and recipient, if the message was read by the recipient and when it was read and can browse the history of their conversations with various peers. To provide fast service the messages should be sent to offline users as e-mails to reduce the delay. User can see a message status (read/unread, message time stamp). A call record, timestamped information about the sender and receiver of messages, their corresponding IP addresses, along with metadata such as message size, type (message or file) should be kept in a SQL database and only authorized users (administrators) should be able to access it. The customer requirement also includes an administrative role: delete a user, block and unblock a user, emergency broadcast message like some crash problem. The administrator can view statistics of records graphically as tables/graphs (GUI) available on main database server (SQL). The messages and the metadata must remain accessible so a user can browse the history of their conversations with various peers.

The sales and support departments of XtremeSecurity have determined that traveling salesmen often require access to the latest marketing materials, white papers, company broschures and user manuals in order to showcase the products to potential customers. Furthermore, support engineers out on the field demand the ability to securely download patches and service packs to resolve customer issues on location. Therefore they need the ability to exchange binary files of arbitrary size in addition to text messages.

Obviously, it is expected that the enterprise chat system must have a friendly graphical user interface (GUI) providing all necessities to edit and read messages, configure the user account, add/remove/browse users in a local address book. The users in the adress book have an associated state (offline, available, idle, busy) that must be shown somehow in the GUI. The user must be able to switch manually between the states "available" and "busy", whereas the system keeps track if the user is "offline" or "idle". Furthermore, a user should be able to temporarily block a user from the address book.

Since XtremeSecurity is paranoid about security they require that each user trying to sign up for the service must authenticate himself against a user database maintained in a central server. The authentication is based on user name (or e-mail address) and password. It's up to you to decide on a suitable way to add/remove users to the central server: you can either provide a minimalistic GUI or offer a script. Furthermore, when a user is added to the local address book of the chat system (as described in the previous paragraphs), the system must verify if the user exists in the database. If that is not case, the system must prevent the operation and present a friendly message in the GUI explaining that only authorized XtremeSecurity employees are allowed to use the system. When an account is created an e-mail is sent to the owner of the account with login details and validity period for the account.

Since the chat messages can cross the Internet in order to reach remote sites, XtremeSecurity insists that both authentication communication as well as chat messages are encrypted. As security experts, they suggest that certificates should be used for each user as well as for the central server hosting the database. Certificates can be self-signed in the release to the customer. Encryption features provided by well-known libraries such as OpenSSL/LibreSSL, libcurl or standard Java/Python libraries (for example, HttpsURLConnection, httplib.py, ssl.py) are allowed to be used. It should be possible to restart the system with encryption disabled for debugging purposes.

Finally, XtremeSecurity requests that all interaction user-to-user and user-to-server be based on a RESTful API with JSON data encoding. Such an API should be easily testable by using HTTP(S) and the command-line utility curl, from the package libcurl.

You must be able to demonstrate that your product supports at least 3 peers. XtremeSecurity is looking forward to receiving your project proposal!