EUROPEAN UNIVERSITY OF LEFKE

FACULTY OF ENGINEERING

# Graduation Project I.

**Project topic**: **Chat Application.**

**Name: Chimarokem Divine ORJI.**

**Student Number: 174413.**

In the project, I will be building a chat application. The chat application will be a platform where users can communicate with one another in real time no matter how distant they are from each other.

**Supervisor**

Prof. Ezgi Deniz Ülker

**Publish Date**

January 2021

Table of Content

## 1. Introduction

### 1.1 Problem Definition

- People can’t communicate except they meet each other physically.

- Two or more people cannot share ideas with one another except they travel and gather together which will much effort, waste of time and inefficiency for a group of company workers.

- In the case of a situation where there is local or worldwide disease outbreak, people cannot talk to one another until they wait for such outbreak to cease.

### 1.2 Goals

- The project aims at eliminating the problem of distance faced by two users in the course of the need to communicate.
- My aim in this project is to reduce the need of a group of users to travel to gather together before they can communicate and share ideas, thereby saving time and energy. After the stress of traveling (for really long distances), they get tired.

- The project will make communication efficient since it provides a real time communication (RTC) through audio call. For a group of company workers who work to grow their company, real time communication will make their planning easier and faster through the conference call functionality I will be incorporating into the project.
- In the case of a global lockdown, company workers/groups will be able to work from home, so their company keeps growing.

## 2. Literature Survey

My project will enable different users to chat with another through sending text and also through real-time voice/audio communication. Below is a list of differences and similarities between my project and other existing similar applications:

**a. Facebook Messenger**:
Facebook Messenger enables users send text content to another user. Facebook has the friend suggests to a user through machine learning. My project might also have that functionality but not with machine learning. I will suggest friends to a user based on how close they are to each other geographically.
Facebook allows a user’s public posts to be viewed, liked or disliked and commented on by his friends. I will also include that functionality in my project.

**b. Whatsapp:**
Whatsapp is allows sending texts, making audio/video call. My project will have that functionality.

My project will have that functionality that allows multiple users to chat in a group.
Whatsapp has end-to-end encryption. I won't be having that in my project yet.

**c. Zoom:**
Zoom has (mainly) conference call ability but does not store chats. It is specially made for conference call. My project will store chats made in a conference call by default (I will make it the user's choice through his settings).

## 3. Background Information

### 3.1 Required Tools, Software and Technologies

Required software:

- Visual Studio Code:
  It's a good code editor because of its great support for software development.
- Google Chrome:
  To view web pages. It is fast.
- The command prompt (Window's shell):
  To run all commands including the one for React.js and Node.js that will make the app run locally in development stage.

Other software:

- Postman:
  It's great for testing requests to endpoints exposed by back-end code.
- Figma:
  It's the most modern software for UI/UX design. One can view his design live
- MongoDB Compass:
  This is the default database viewer for the MongoDB database. It is a cool GUI.
- Git:
  Used for repository

Languages:

- HTML/CSS:
  HTML/CSS for marking up text and styling them

- JavaScript
- React.js/React Native:
  React is good with building components from smaller components and this is good for a big application like my project
- Node.js/Express.js:

  Node.js is powerful and scalable as a back-end programming language.

Technologies:

- Sass:
  Sass is used to write CSS efficiently. I will use it to speed up my styling of the pages
- Redux.js:
  Redux is powerful for handling state in an application. There are many states to be handled in my project like when a user is online.

- Socket.io:

  This allows for sending text in real-time. I'll be using this because of speed

- Web RTC:

  This allows for real-time communication with audio and video call.
- Pusher:
  Pusher listens for changes in the database and trigger events based on these changes in the front-end. So it will improve speed in my project.

- JSON Web Tokens (JWT):

  I will be using JWT for authentication. It's a technology that is used to authenticate and verify a user.

<u>Databases:</u>

- MongoDB/Mongoose:
  Good database. Mongoose helps in modelling the Mongo DB database schema

# 4. Modules

## 4.1 The Back-end:

In the back-end, I will be using Node.js/Express as the language. Here, I will be connecting to the mongo DB database. I will be using the Model View Controller architecture (MVC). Below is the explanation of the significance of the MVC model in my project.

The Model refers to the required entities such as User, Room, Friend, Chat, Post, Comment, Videocall, and Audiocall. The entities will be modelled with the help of mongoose and will translate into tables in the MongoDB database.

The View will be the front-end which will be explained soon.

The Controller will be all the functions that will create and expose all routes/endpoints. It will also comprise of all functions that will be run in order to handle all requests sent through these routes. These functions will receive the route parameters (if any) and then query the database based on the nature/type of these route parameters. Once this querying is done, some checks and logic will be done – like: is the user authenticated or authorized? Afterward, this queried data is then exposed as a response.

Also, in the back-end, there will be code listening for socket.io events such as the 'connection' or 'disconnection' or 'join' (when someone joins a chat) or 'leave' (when someone leaves a chat) events. And then, there'll be code to handle these events. These events will be triggered by user actions in the front-end. In the back-end, special events will also be triggered which will be received and handled by the front-end. All of these events are only specific to text messaging.

There will also be events that will be triggered when a user starts a new call connection. There will be another event sent when another joins that call connection. This code section that will handle these types of events is where I will bring in the Web RTC technology.

## 4.1 The Front-end:

In the front-end, I will be creating the HTML and CSS mark-up and styling in React. All the buttons, input, navigation, dropdowns, text and icons that the user sees will be done in React through components.
React will be the manager in the front-end. Here, I will create complex components from smaller ones. Here, I'll be managing a lot of states including online state, light-theme/dark-theme states, button-click state, and so on. I will use Redux to help me in state management because Redux has powerful & efficient state management system.

My React code will have asynchronous functions that will get with the data endpoints (responses) given by the back-end. React will then mix this data with HTML-like code and then renders it to user.

There is a package which will be used in React called socket.io-client. Using this package will allow user actions (like clicking a call button) to trigger events which will be handled at the back-end. This package will also handle events which are sent by the back-end.

## 5. Risk Analysis:

- Messages that users send to one another will not be end-to-end encrypted. So, internet providers and other similar third parties will be able to read the messages easily. And so, a user might see his conversation with someone randomly on the internet.

- Users might rarely use my app because of already existing similar applications like Whatsapp which could have other advanced functionality.
- It could be only about 0.1% possible that connection to the database might not happen instantly since MongoDB is not a real-time database. That's why I'm using Pusher. Pusher listens to database events and notifies the front-end of those changes.

## 6. Ethics.

- I'll try my best to do the project with as much clean code as possible, so that in the future, the code will be as easy as possible to be read and easily updated if the desire to update it to a better version arises.

- I will participate in much learning which will equip me with much knowledge of the technologies I'll be working with.
  (Resource is from item 8 at: https://www.researchgate.net/publication/227991826_Software_Engineering_Ethics)

- Large files of my project containing code will be split into sub-sections.
- I will set goals of when to achieve different parts of the project in order to finish before submission time. I will be disciplined to achieve these goals.

## 7. Conclusion.

### 7.1 Benefits:

#### a. Benefits to users:

- Users will be able to communicate even when they're far apart.
- Users will have the regular audio/video call and also have conference call all in one app.
- Users will be able to send their exact location to other users.

#### b. Benefits to me:

- I will gain great confidence of being strong with the MERN development stack. MERN is MongoDB + Express.js +React.js + Node.js.
- This project can get me a job as a software developer because it is a really large project using about 12 or technologies.

I chose this project because I was curious to know how real-time messaging works and its implementation with React.js.

### 7.2 Future Works

I will continue with the project even after graduation. I want it to be one of my strongest projects on my portfolio. I desire to make users' messages end-to-end encrypted later.

## 8. References.

[a] : Whatsapp page. Retrieved from: https://www.whatsapp.com/features/

[b] : Wikipedia. Retrieved from: https://en.wikipedia.org/wiki/List_of_Facebook_features

[c] : Wikipedia. Retrieved from: https://zoom.us/docs/image/features/Welcome.png