Mini Project Half Yearly Presentation

Subtext: Context Based Real Time Communication App

Anish Sachdeva

DTU/2K16/MC/13

Under Prof. Dr. S. Sivaprasad Kumar

Aim: Things that my app should support

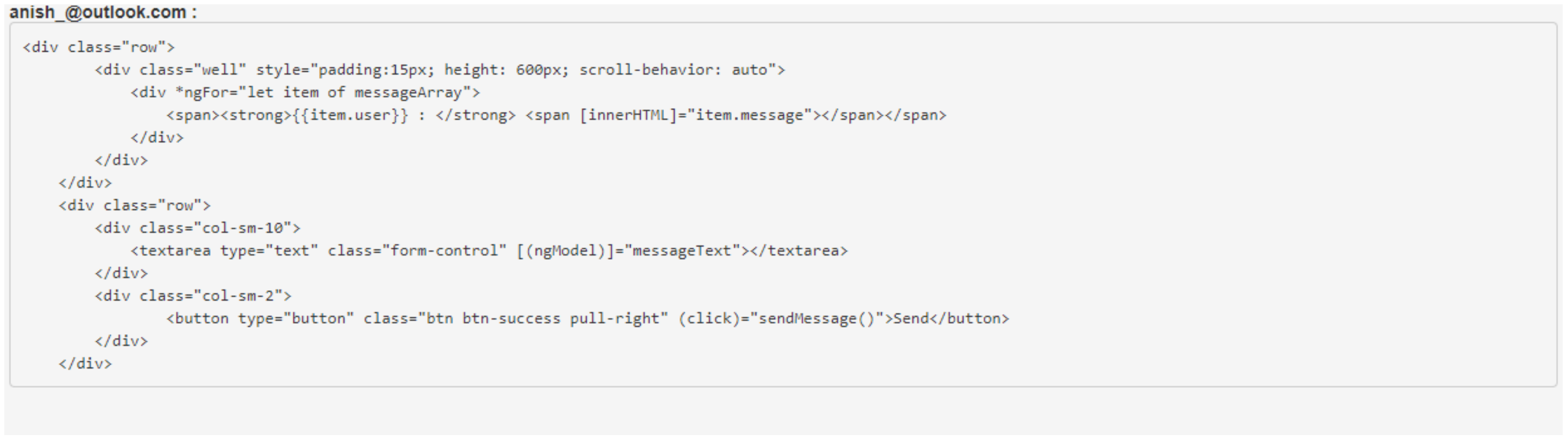
- Text
- Code
- MarkDown
- Emojis
- LaTeX

Text

- Messages like this
- And this
- Also numbers 123456
- and small case + CAPITAL
- Plus special characters /*,.<>;'":}{][!@#$%^&*()_+=-

Java
C/C++
C#
JavaScript
Swift
Python
….
….
And everything in between

```
class BinarySearchExample{
 public static void binarySearch(int arr[], int first, int last, int key){
   int mid = (first + last)/2;
   while( first <= last ){
      if ( arr[mid] < key ){
        first = mid + 1;
      }else if ( arr[mid] == key ){
        System.out.println("Element is found at index: " + mid);
        break;
      }else{
         last = mid - 1;
      }
      mid = (first + last)/2;
   }
   if ( first > last ){
      System.out.println("Element is not found!");
   }
 }
 public static void main(String args[]){
        int arr[] = {10,20,30,40,50};
        int key = 30;
        int last=arr.length-1;
        binarySearch(arr,0,last,key);
 }
}
```



anish_@outlook.com :

```
<div class="row">
        <div class="well" style="padding:15px; height: 600px; scroll-behavior: auto">
            <div *ngFor="let item of messageArray">
                <span><strong>{{item.user}} : </strong> <span [innerHTML]="item.message"></span></span>
            </div>
        </div>
    </div>
    <div class="row">
        <div class="col-sm-10">
            <textarea type="text" class="form-control" [(ngModel)]="messageText"></textarea>
        </div>
        <div class="col-sm-2">
                <button type="button" class="btn btn-success pull-right" (click)="sendMessage()">Send</button>
        </div>
    </div>
```

Markdown

- __bold__ --> **bold**
- _italic_ --> *italics*
- `code` -> code
- ```js``` --> perfomatted code
- # heading -> Main Heading etc.

Emojis

- 😀 😁 😂 🤣 😃 😄 😅
  😆 😉 😊 😋 😎 😍 😘
  ? 😗 😙 😚 ☺ 🙂 🤗 ?
  🤔 ? 😐 😑 😶 🙄 😏
  😣 😥 😮 🤐 😯 😪 😫
  😴 😌 😛 😜 😝 🤤 😒
  😓 😔 😕 🙃 🤑 😲 ☹
  🙁 😖 😞 😟 😤 😢 😭
  😦 😧 😨 😩 ? 😬 😰
  😱 ? ? 😳 ? 😵 😡 😠 ?
  😷 🤒 🤕 🤢 ? 🤧 😇
  🤠 🤡 ? ? ? 🤥 ? ? ? 🤓
  😈 👿 👹 👺 💀 👻 👽
  🤖 💩 😺 😸 😹 😻 😼
  😽 🙀 😿 😾

LaTeX

$\pm$	`\pm`	$\cap$	`\cap`	$\diamond$	`\diamond`	$\oplus$	`\oplus`
$\mp$	`\mp`	$\cup$	`\cup`	$\bigtriangleup$	`\bigtriangleup`	$\ominus$	`\ominus`
$\times$	`\times`	$\uplus$	`\uplus`	$\bigtriangledown$	`\bigtriangledown`	$\otimes$	`\otimes`
$\div$	`\div`	$\sqcap$	`\sqcap`	$\triangleleft$	`\triangleleft`	$\oslash$	`\oslash`
$\ast$	`\ast`	$\sqcup$	`\sqcup`	$\triangleright$	`\triangleright`	$\odot$	`\odot`
$\star$	`\star`	$\vee$	`\vee`	$\lhd$	`\lhd`^{b}	$\bigcirc$	`\bigcirc`
$\circ$	`\circ`	$\wedge$	`\wedge`	$\rhd$	`\rhd`^{b}	$\dagger$	`\dagger`
$\bullet$	`\bullet`	$\setminus$	`\setminus`	$\unlhd$	`\unlhd`^{b}	$\ddagger$	`\ddagger`
$\cdot$	`\cdot`	$\wr$	`\wr`	$\unrhd$	`\unrhd`^{b}	$\amalg$	`\amalg`
$+$	`+`	$-$	`-`				

Context based real time communication app that can understand information encoded with different text markup formats

$\downarrow$

SUBTEXT

Current Market and Competition

- Whatsapp
- Facebook Messenger
- Instagram
- Google Allo/Messenger
- Microsoft Office
- Skype
- Google Docs/Sheets/Slides

Text + Emojis

- Whatsapp
- Instagram
- Facebook Messenger
- Google
- Microsoft etc.

Text + Emojis + Markdown

- Slack
- Whatsapp (Italic and bold from Markdown)

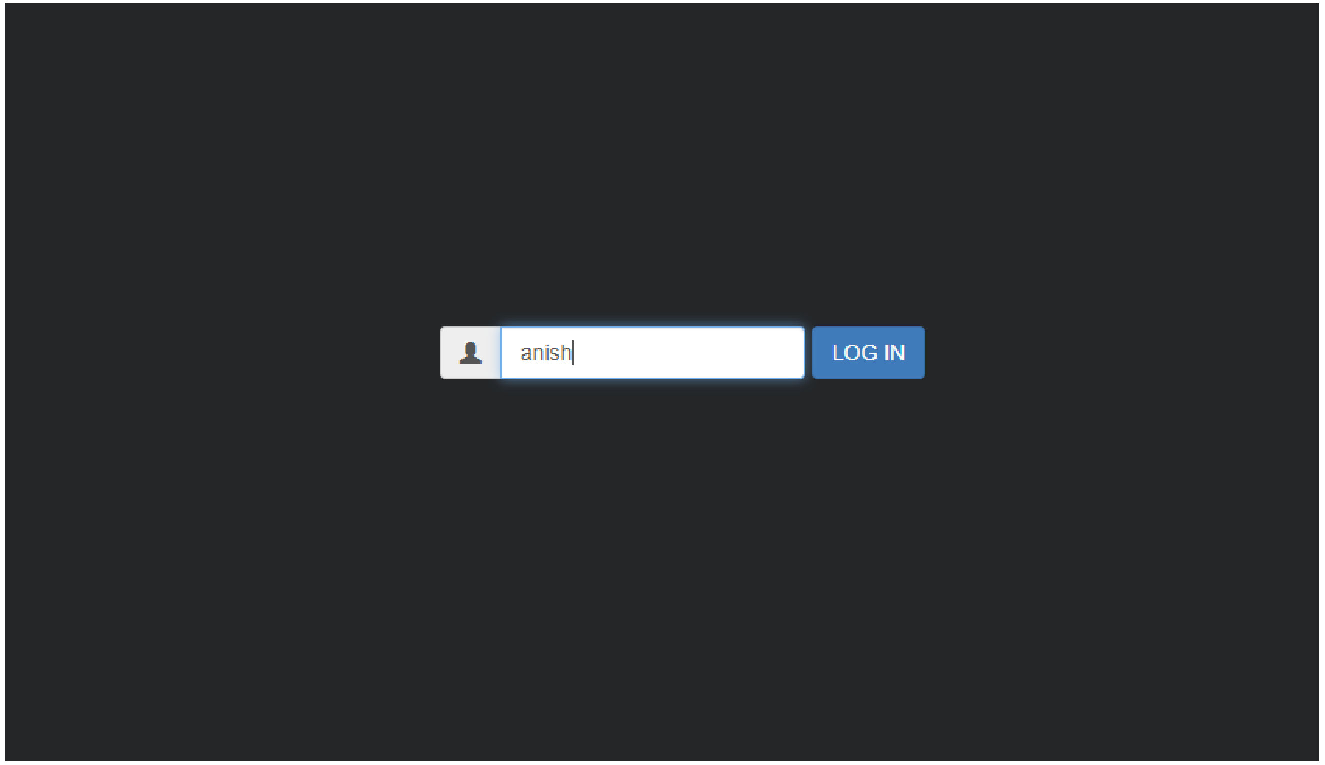
LaTeX + Others

None to my knowledge 🤷‍♀️

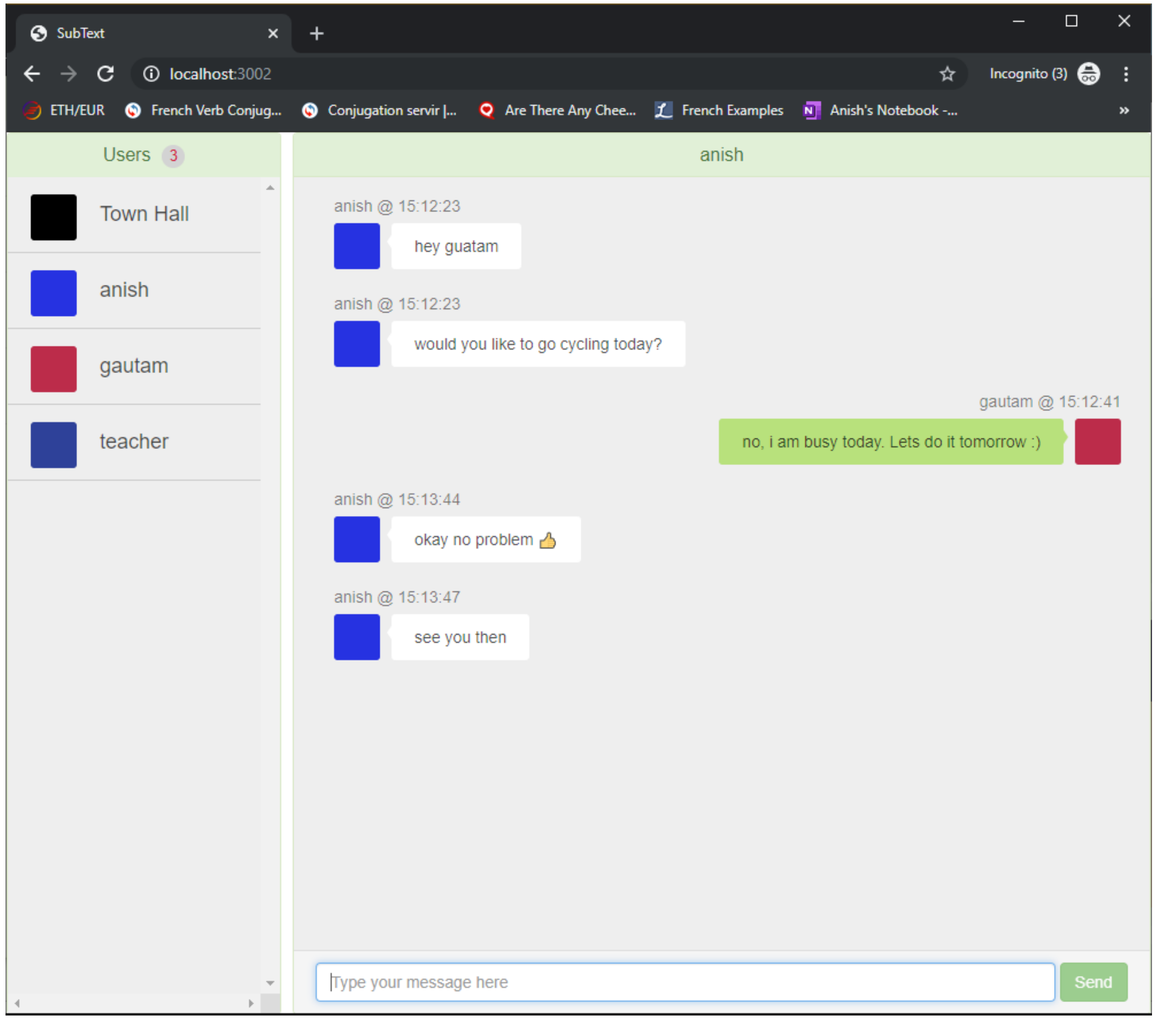
Initial Prototype

- Created using AngularJS, Express and MongoDB
- Has proper log in screen and text messaging facility people to people and also group chatting
- Also added chatbot that announces all active participants and when they leave the group etc.
- Disadvantages: does not support Markdown/LaTeX
- Another Disadvantage: was built on complete server side technology hence no client side rendering and computation available

Prototype I: Login page



Prototype I: Chatting Person/Person



anish @ 15:12:23
hey guatam

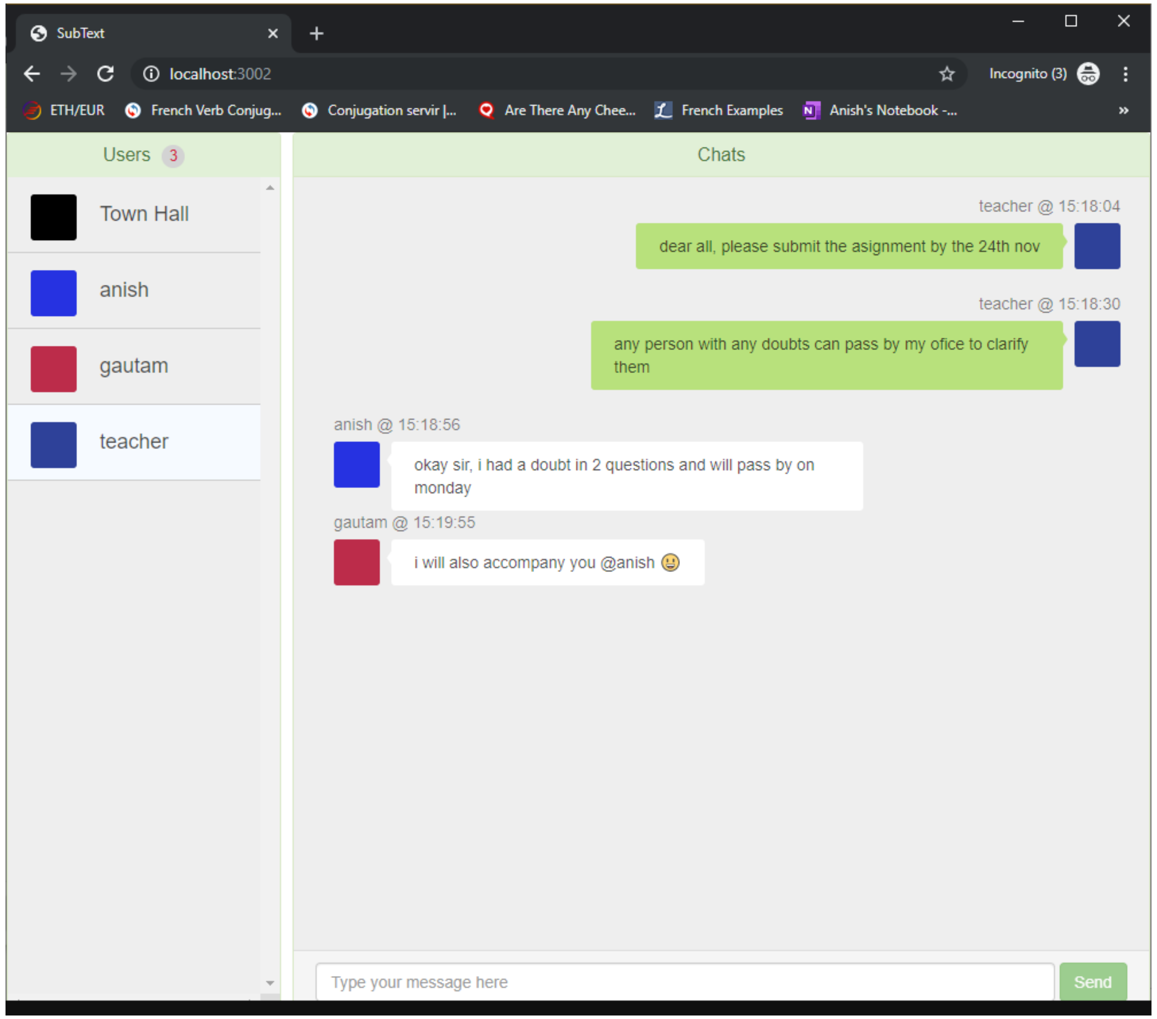
anish @ 15:12:23
would you like to go cycling today?

gautam @ 15:12:41
no, i am busy today. Lets do it tomorrow :)

anish @ 15:13:44
okay no problem 👍

anish @ 15:13:47
see you then

Prototype I: Person/Group



teacher @ 15:18:04
dear all, please submit the asignment by the 24th nov

teacher @ 15:18:30
any person with any doubts can pass by my ofice to clarify them

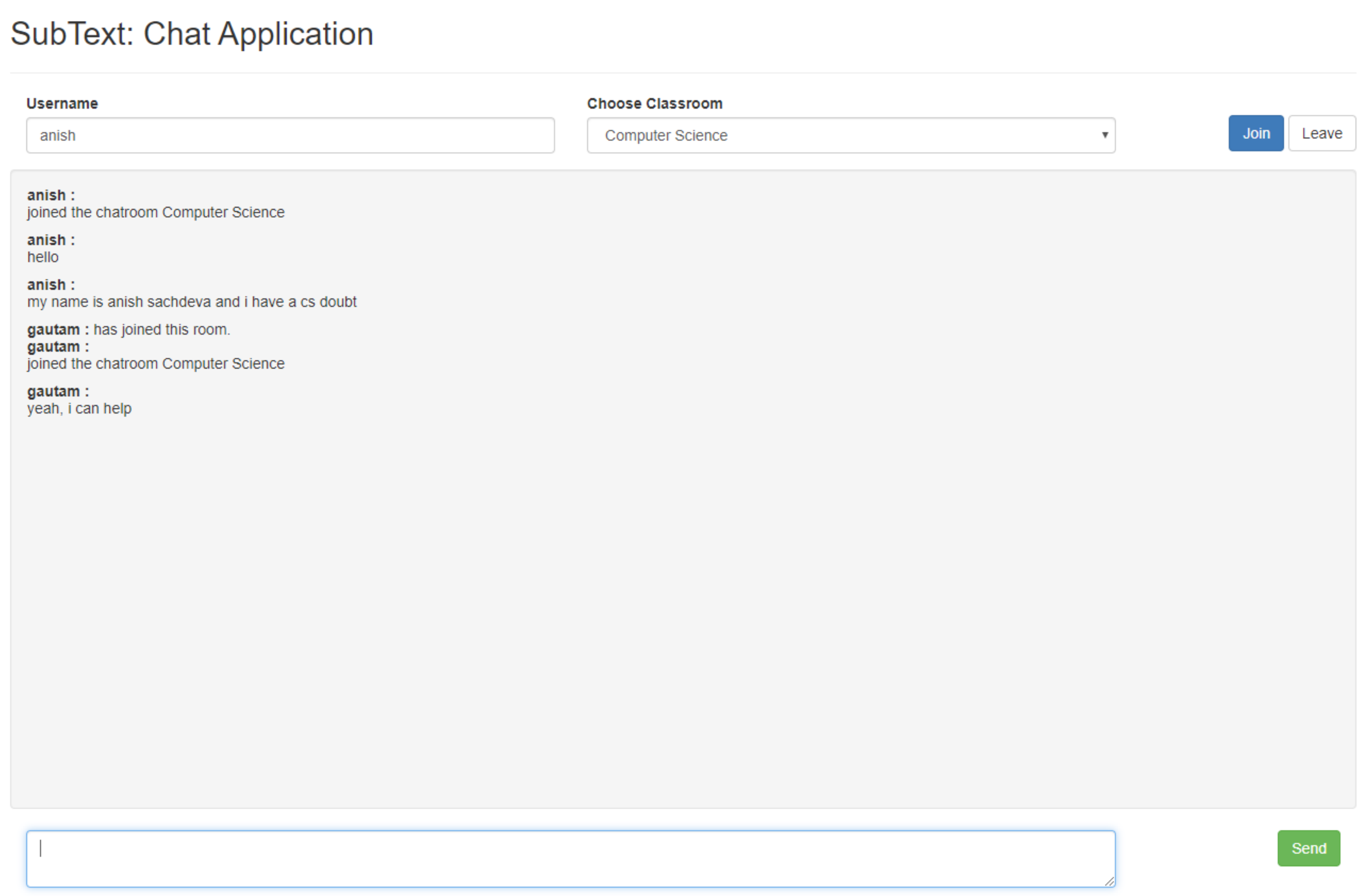
anish @ 15:18:56
okay sir, i had a doubt in 2 questions and will pass by on monday

gautam @ 15:19:55
i will also accompany you @anish 🙂

Prototype II

- Added mark down support
- Added multiple classroom support
- Created markdown analyser and text parser from scratch

Prototype II: Text interaction


SubText: Chat Application

Username: anish

Choose Classroom: Computer Science

anish :
joined the chatroom Computer Science

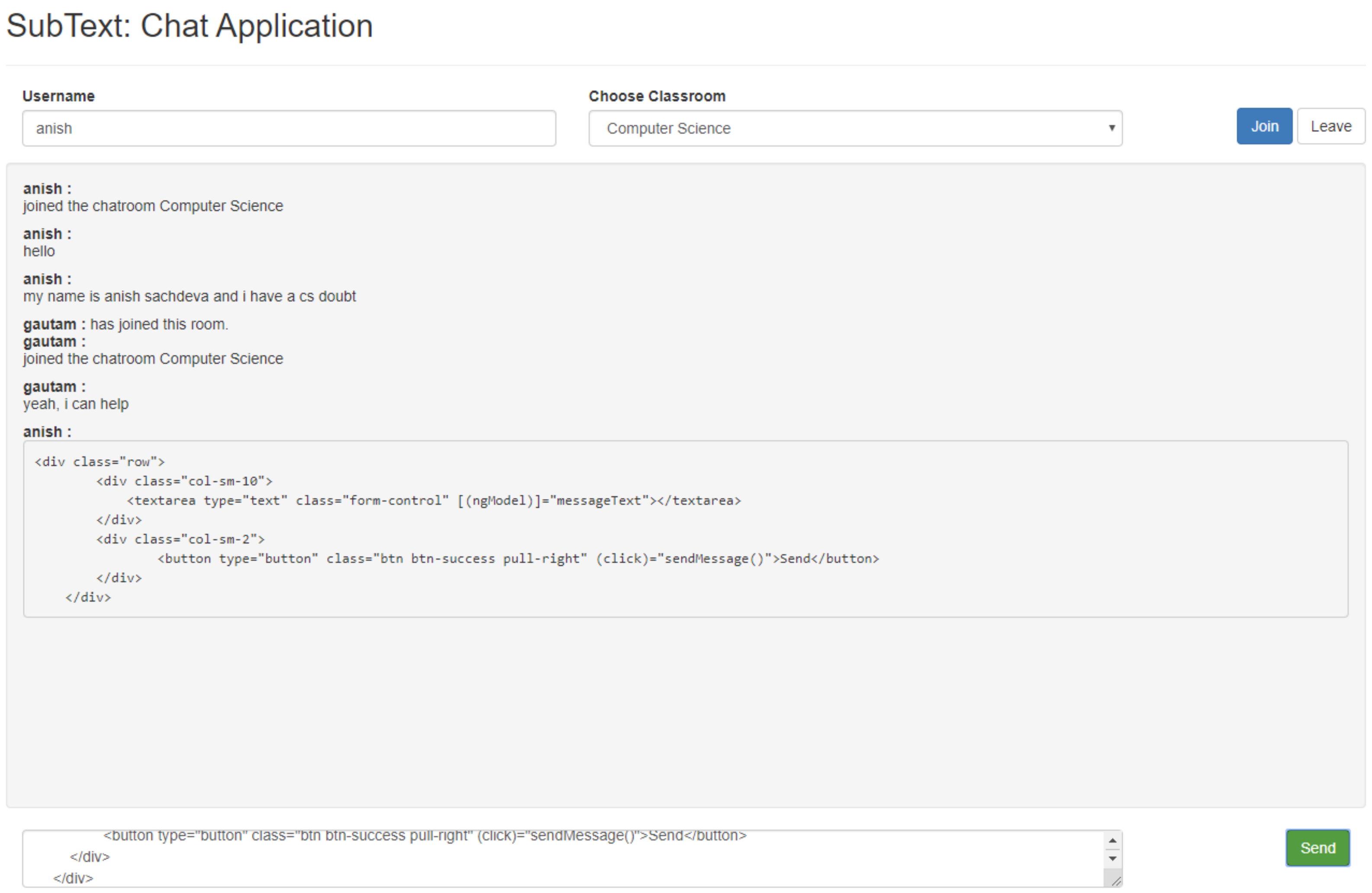
anish :
hello

anish :
my name is anish sachdeva and i have a cs doubt

gautam : has joined this room.
gautam :
joined the chatroom Computer Science

gautam :
yeah, i can help

Prototype II: Markdown + Text Interaction



SubText: Chat Application

Username: anish

Choose Classroom: Computer Science

anish :
joined the chatroom Computer Science

anish :
hello

anish :
my name is anish sachdeva and i have a cs doubt

gautam : has joined this room.
gautam :
joined the chatroom Computer Science

gautam :
yeah, i can help

anish :

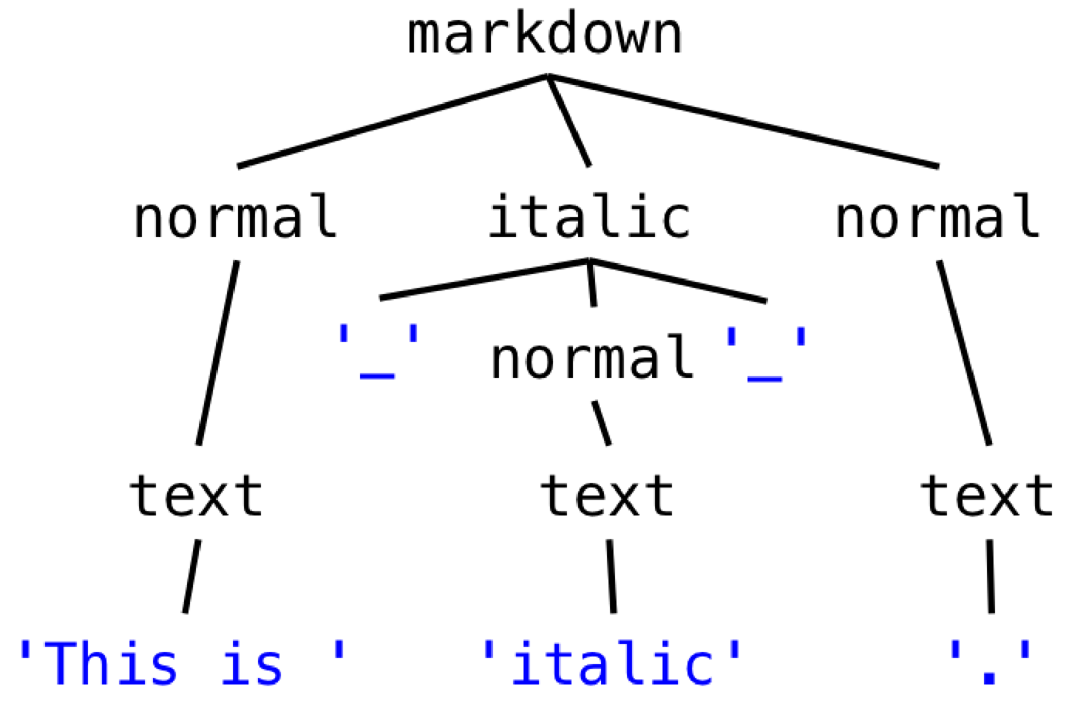
```
<div class="row">
        <div class="col-sm-10">
            <textarea type="text" class="form-control" [(ngModel)]="messageText"></textarea>
        </div>
        <div class="col-sm-2">
                <button type="button" class="btn btn-success pull-right" (click)="sendMessage()">Send</button>
        </div>
    </div>
```

```
            <button type="button" class="btn btn-success pull-right" (click)="sendMessage()">Send</button>
        </div>
    </div>
```

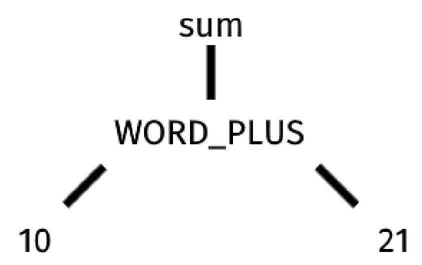
Markdown

- **Markdown** is a lightweight markup language with plain text formatting syntax. Its design allows it to be converted to many output formats, but the original tool by the same name only supports HTML. Markdown is often used to format readme files, for writing messages in online discussion forums, and to create rich text using a plain text editor.
- Developed by John Gruber and Aaron Swartz

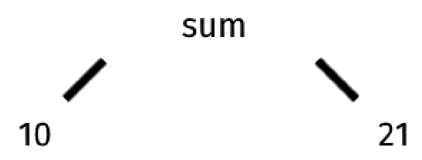
Lexical Parser



Parse Tree



Abstract Syntax Tree



Future Tasks

- Add LaTeX Support
- Improve the UI/Layout
- Add a proper login/logout screen
- Connect to a database to make the messages persistent

Possible Latex Implimentation Methods

- Creating/Importing a LaTeX to html parser similar to Markdown and saving the html translated messages and using that as interchange text
- Using a LaTeX compiler in real time to draw the images and then sending the images as interchnage between parties