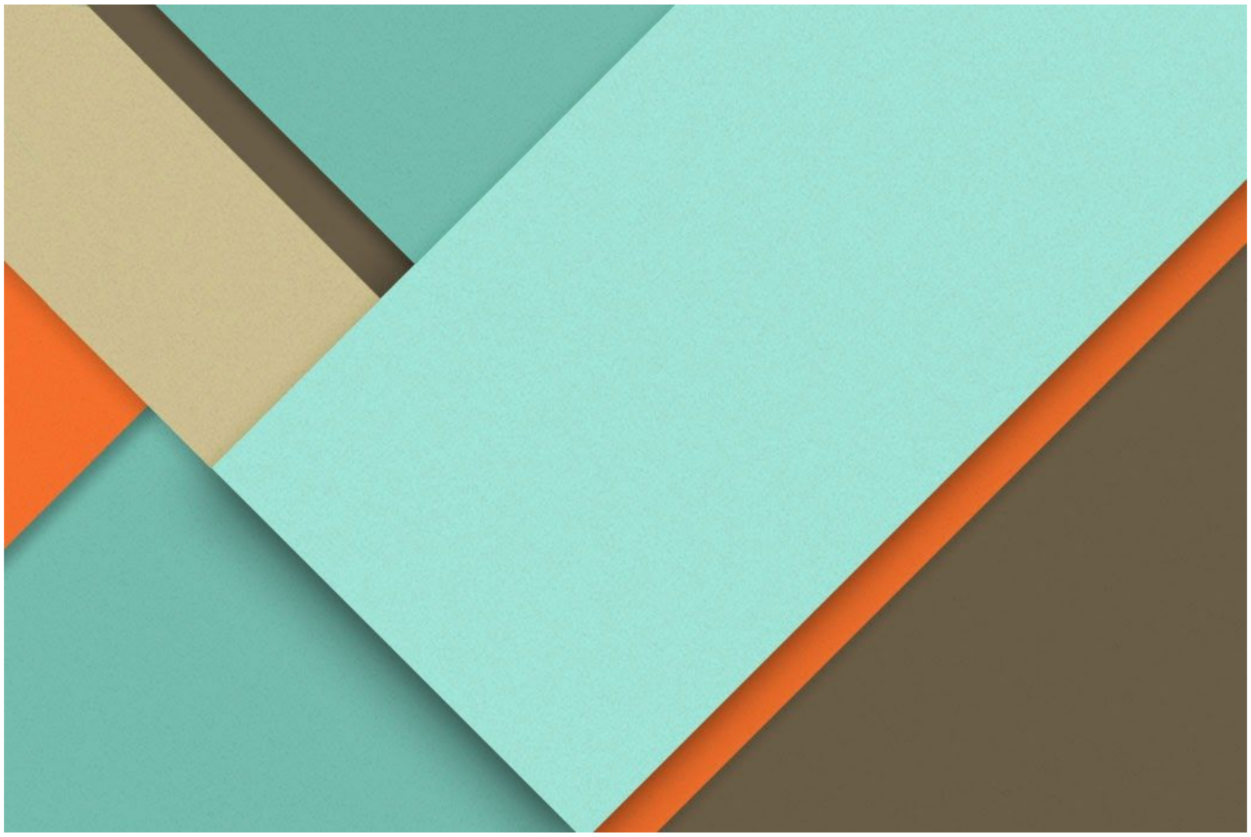
DELHI TECHNOLOGICAL UNIVERSITY

Mini Project Mid-Semester Progress Report (6th Semester)

MC-391

Anish Sachdeva

MC/2K16/013



Mini Project Mid-semester Progress Report

02.02.2020

Anish Sachdeva
Delhi Technological University
DTU/2K16/MC/13
Under the Guidance of: **Prof. Dr. S. Sivaprasad Kumar**

Acknowledgements

I would like to express my special thanks of gratitude to my teacher Prof. Dr. S. Sivaprasad Kumar of the Mathematics department at Delhi Technological University (DTU) who gave me a golden opportunity to do this wonderful project on creating a Context based communication web app, which has led me to research heavily and implement web apps that are completely RESTful and use modern technologies like MongoDb, Angular, Express, Node.js etc. and are managed in a modern way using git and maintained on Github.

Secondly i would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

Index

Introduction to The App

This app “Subtext” was designed with the sole purpose of filling in a much needed void in the current marketspace where no modern real-time communication app provides a means to relay or convey contextual information and send messages in other formats than just plain English.

Modern chatting applications are very friendly to use, and are also very social with Emojis and the ability to talk in real-time with multiple persons and also create groups, but they don’t do well in scientific or professional communities where the sender and receiver might want to send each other specific data that is very contextual in nature and the chatting app just can’t change itself to satisfy this requirement,

The non-exhaustive list of chatting applications prevalent in our current market are:-

1. Whatsapp
2. Facebook Messenger
3. Skype
4. Google Allo
5. Google Duo
6. Google Hangouts
7. Apple Facetime
8. Apple Messenger
9. Instagram
10. Tiktok
11. Slack

And many more such diverse applications where not a single one can holistically cater to the needs of professionals.

Aim, Mission and Vision

What this project aims at creating is a webapp that combines normal real-time chatting features present in applications like Whatsapp to act as a platform to showcase the power of a contextual based transmission system that can decode UTF-8 text, emojis along with LaTeX and Markdown syntax.

So people from various disciplines, professionals, scientists and Musicians can use it and come together on a common platform.

My mission was to create a context based communication app that incorporates and supports multiple data interchange formats.

My vision as to empower every organization and individual to be able to use a common platform to convey, share and access context-based relevant information that is dynamic and adaptable.

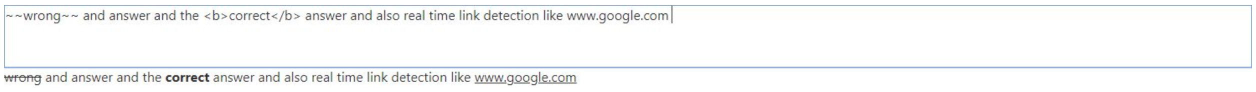
What Was accomplished this semester?

A very big chunk of this app and what this app aims to showcase was added this semester and that is LaTeX parsing and comprehension.

A new LaTeX parser was built from scratch using the existing LaTeX engine and maths symbols available for CSS and that engine was minified and designed in such a manner that it is lightweight enough to be shipped with the running front-end version of the application, so that the parsing will not use any cloud compute resources, but simply be done on the client side and can be done efficiently in an optimized manner and can also be done without any internet, so even if the user is without internet and without any connection to the server, the compression and conversion will still take place.

In fact the whole conversion and compression are completely server-side agnostic processes.

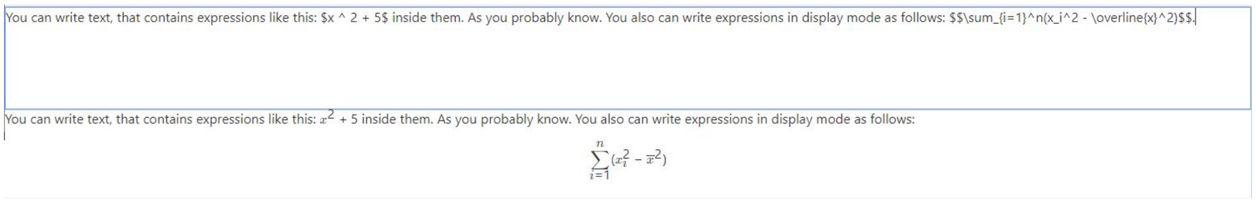
By adding the LaTeX parser and a dialog box that shows what the end output will be in real-time the user can now see what the end result will be for both Markdown and LaTeX.



```
~~wrong~~ and answer and the <b>correct</b> answer and also real time link detection like www.google.com
```

~~wrong~~ and answer and the **correct** answer and also real time link detection like www.google.com

And also as



```
You can write text, that contains expressions like this: $x ^ 2 + 5$ inside them. As you probably know. You also can write expressions in display mode as follows: $$\sum_{i=1}^n(x_i^2 - \overline{x}^2)$$.
```

You can write text, that contains expressions like this: $x^2 + 5$ inside them. As you probably know. You also can write expressions in display mode as follows:

$$\sum_{i=1}^{n}(x_i^2 - \overline{x}^2)$$

Further-more the UI layout was also improved considerably to provide the user a much better User experience akin to applications like Whatsapp, Slack etc.

The UI overhaul includes a dedicated left bar to indicate all groups that a person is part of. The left pane is divided into a messages are where all text messages are displayed and a text area where one can type a message and also see in real time how they will be transpiled in LaTeX or Markdown.

LaTeX - A Brief Introduction

LaTeX is a high-quality typesetting system; it includes features designed for the production of technical and scientific documentation. LaTeX is the de facto standard for the communication and publication of scientific documents.

LaTeX is widely used in academia for the communication and publication of scientific documents in many fields, including mathematics, statistics, computer science, engineering, chemistry, physics, economics, linguistics, quantitative psychology, philosophy, and political science. It also has a prominent role in the preparation and publication of books and articles that contain complex multilingual materials, such as Sanskrit and Greek. LaTeX uses the TeX typesetting program for formatting its output, and is itself written in the TeX macro language.

LaTeX was created in the early 1980s by Leslie Lamport, when he was working at SRI. He needed to write TeX macros for his own use, and thought that with a little extra effort he could make a general package usable by others. Peter Gordon, an editor at Addison-Wesley, convinced him to write a LaTeX user's manual for publication (Lamport was initially skeptical that anyone would pay money for it); it came out in 1986] and sold hundreds of thousands of copies. Meanwhile, Lamport released versions of his LaTeX macros in 1984 and 1985. On 21 August 1989, at a TeX Users Group (TUG) meeting at Stanford, Lamport agreed to turn over maintenance and development of LaTeX to Frank Mittelbach. Mittelbach, along with Chris Rowley and Rainer Schöpf, formed the LaTeX3 team; in 1994, they released LaTeX 2e, the current standard version, and continue working on LaTeX3.

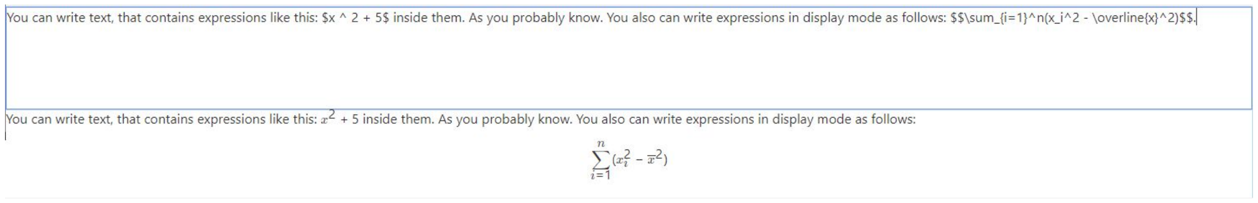
How was LaTeX Parser Implemented In the Project?

A LaTeX parser was implemented after extreme effort and painstaking amounts of code to properly parse text and convert them into LaTeX and then corresponding HTML structure that preserves both the legibility, structure and initial context that would have been generated in LaTeX.

The TeX Compiler which can compile down simple text to a tex equation was minified and embedded in the client-side code. Then the output from this compiler which is generated as a DVI file is parsed and reduced to symbols and text phrases.

This parsed output is then used to create a lexical token tree. The Lexical tree is then used to construct an HTML output by using this tree to create the same LaTeX output using HTML tags.

A special font was also imported through CSS and this and the monospace font were combined to create a union of fonts to provide space for the LaTeX typeset. The HTML output is then simply displayed in the browser which seems like LaTeX to the user of the application but just as markdown is actually just plain old HTML.



```
You can write text, that contains expressions like this: $x ^ 2 + 5$ inside them. As you probably know. You also can write expressions in display mode as follows: $$\sum_{i=1}^n(x_i^2 - \overline{x}^2)$$.
```

You can write text, that contains expressions like this: $x^2 + 5$ inside them. As you probably know. You also can write expressions in display mode as follows:

$$\sum_{i=1}^{n}(x_i^2 - \overline{x}^2)$$

Overhaul of the Already Existing Project

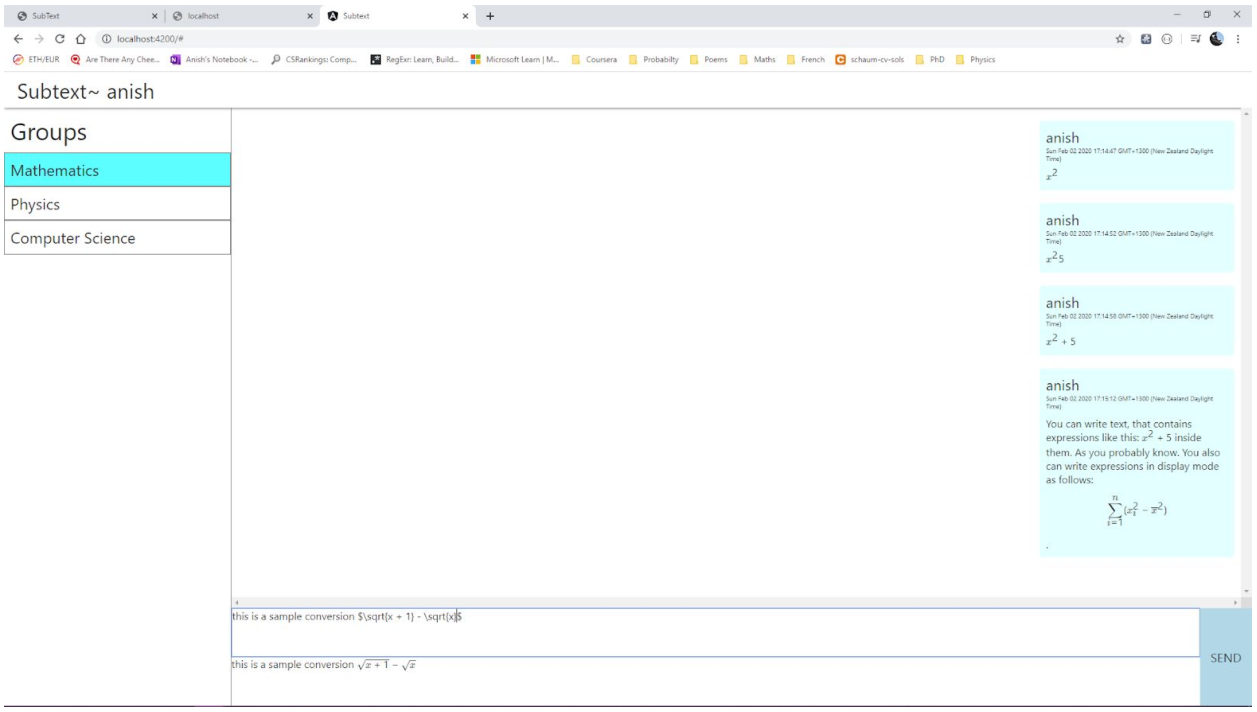
The existing project didn't have a very elegant client-facing side, so a brand new project with a brand new client-facing side was created. This new client-facing app contains a dashboard to showcase the groups on the left hand side and the chatting pane on the right hand side with all the messages and as a new textarea that can be used to type out messages and also see the compiled output before you hit send.

The new project has a much cleaner interface and also contains the storage to store different messages for different groups.

It also contains the feature to have different contextual based messages for different groups, for example Markdown for the Computer Programming Group and LaTeX for the Mathematics and Physics group.

It also has a brand new message format which shows the exact time and date of when the message was sent along with the sender and excellent formatting of the actual text message content to make it very legible and clear.

The new app with some sample message examples can be viewed as:



anish
Sun Feb 02 2020 17:14:47 GMT+1300 (New Zealand Daylight Time)
x^2

anish
Sun Feb 02 2020 17:14:52 GMT+1300 (New Zealand Daylight Time)
$x^2 5$

anish
Sun Feb 02 2020 17:14:58 GMT+1300 (New Zealand Daylight Time)
$x^2 + 5$

anish
Sun Feb 02 2020 17:15:12 GMT+1300 (New Zealand Daylight Time)
You can write text, that contains expressions like this: $x^2 + 5$ inside them. As you probably know. You also can write expressions in display mode as follows:

$$\sum_{i=1}^{n}(x_i^2 - \overline{x}^2)$$

.

this is a sample conversion $\sqrt{x + 1} - \sqrt{x}$

this is a sample conversion $\sqrt{x+1} - \sqrt{x}$

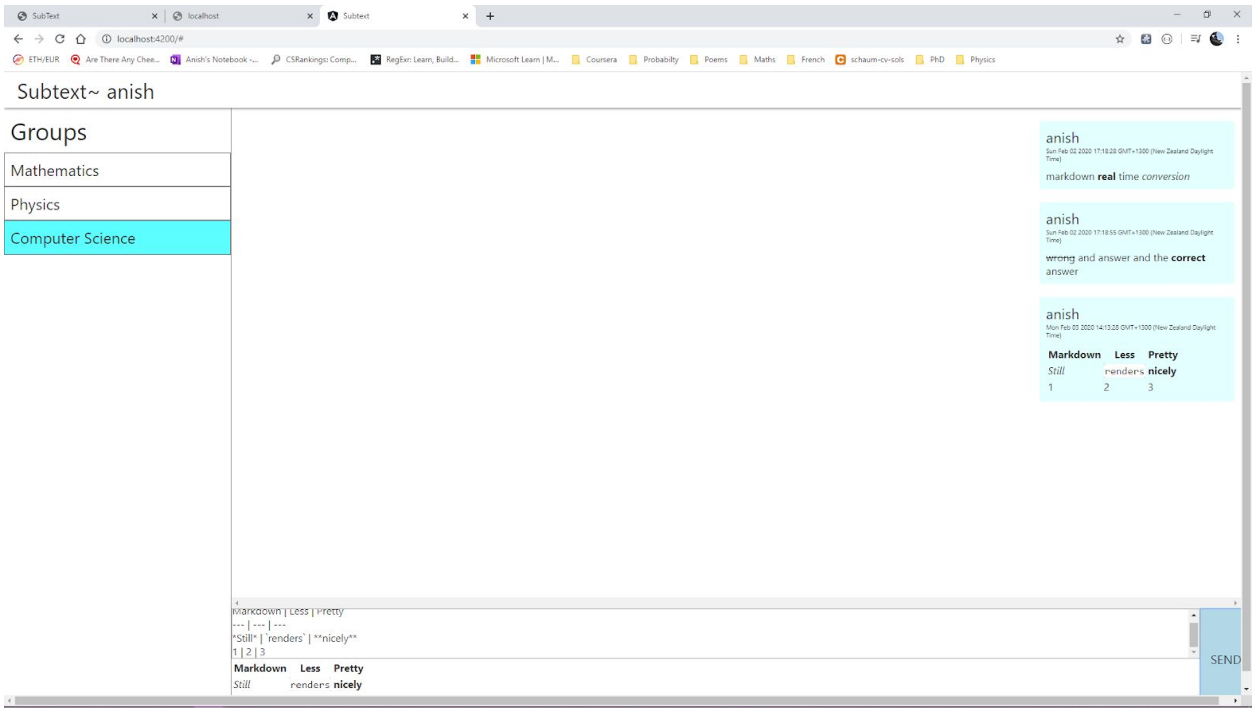
Overhaul to the Markdown Parser

The markdown parser was also overhauled and much improved upon, Using the latest angular 8 version and the new ivy engine released by Angular last year. The size of the whole parser and converter engine was further minified and reduced to just 2KB.

Many new additions were also added to the parser so as to ad many popular GitHub flavoured Markdown features such as Tables, automatic link detection and task lists.

The tables feature is one which is very widely used throughout the industry and adding it really improves the whole application by providing users with rich features that are available in high fidelity rich text format applications such as word free of cost at their fingertips in a normal standard communication app.

An example of tables is as follows:



anish
Sun Feb 02 2020 17:18:28 GMT+1300 (New Zealand Daylight Time)
markdown **real** time *conversion*

anish
Sun Feb 02 2020 17:18:55 GMT+1300 (New Zealand Daylight Time)
~~wrong~~ and answer and the **correct** answer

anish
Mon Feb 03 2020 14:13:28 GMT+1300 (New Zealand Daylight Time)

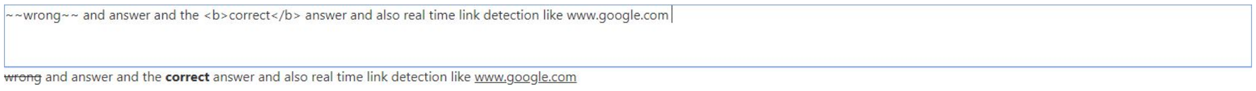
Markdown	Less	Pretty
Still	`renders`	**nicely**
1	2	3

```
Markdown | Less | Pretty
--- | --- | ---
*Still* | `renders` | **nicely**
1 | 2 | 3
```

Markdown	Less	Pretty
Still	`renders`	**nicely**

Here we can clearly see markdown converting and showing a table in realtime and also rendering it perfectly as a message and now users can send tables and items as messages.

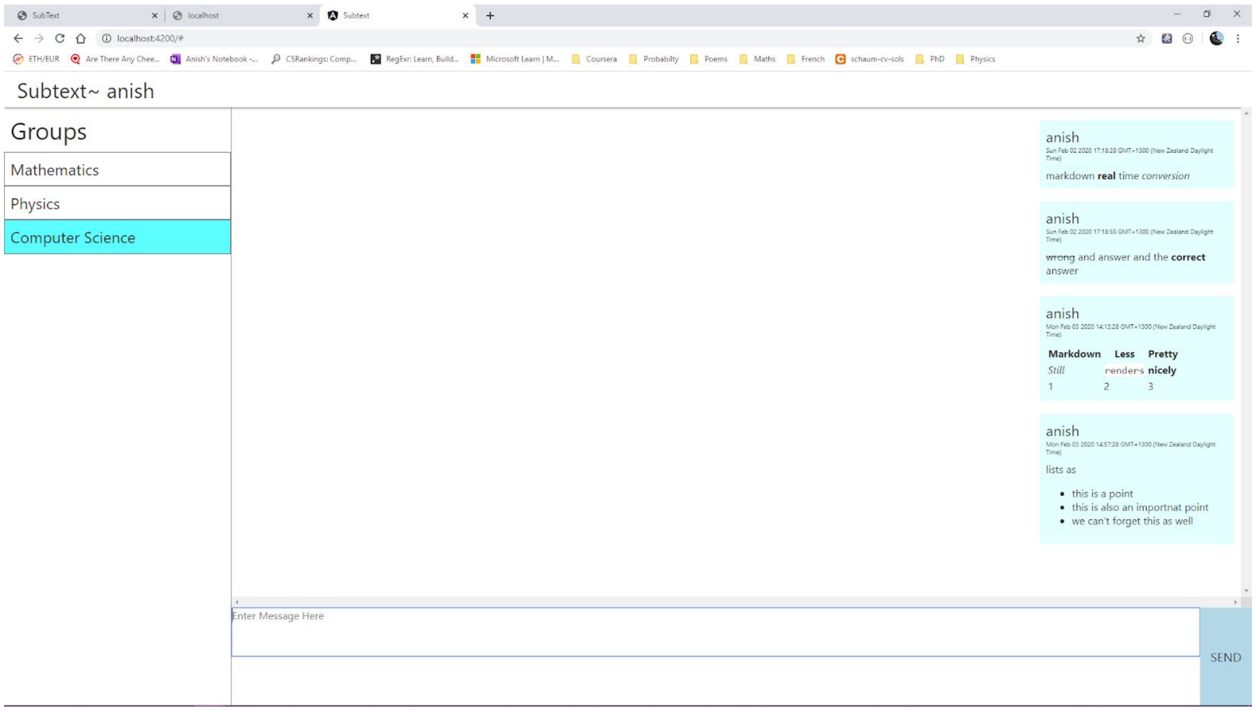
Users can now also type normal links such as www.google.com or any other such valid HTTP URL and the markdown parser will automatically parse it and convert it into a link that users can open. See functioning below as:



~~wrong~~ and answer and the <b>correct</b> answer and also real time link detection like www.google.com

~~wrong~~ and answer and the **correct** answer and also real time link detection like www.google.com

Further support for lists and unordered lists and enumerated lists was also added. Support for task lists was also added.



anish
Sun Feb 02 2020 17:18:28 GMT+1300 (New Zealand Daylight Time)

markdown **real** time *conversion*

anish
Sun Feb 02 2020 17:18:55 GMT+1300 (New Zealand Daylight Time)

~~wrong~~ and answer and the **correct** answer

anish
Mon Feb 03 2020 14:13:28 GMT+1300 (New Zealand Daylight Time)

Markdown	Less	Pretty
Still	`renders`	**nicely**
1	2	3

anish
Mon Feb 03 2020 14:57:28 GMT+1300 (New Zealand Daylight Time)

lists as

- this is a point
- this is also an importnat point
- we can't forget this as well

Using the Application

Using the application is fairly simple. Although one will need node, npm and git installed in their machine for successfully running the application. The following software can be obtained from the following links:

1. https://nodejs.org/en/
2. https://www.npmjs.com/
3. https://git-scm.com/

After installing said packages, head to y project repository on GitHub at https://github.com/anishLearnsToCode/subtext and pull this project on your local machine. Once done open a command line tool like Command Prompt or Powershell on Windows or bash on linux or terminal on Mac.

Enter the project directory and simply run the following commands.

```
>> npm i #this will install and update all project dependencies

>> ng serve # This will run a development server of the SubText Application at localhost:4200
```

Once the application is up and running, simply open any web browser at localhost:4200 and enjoy :)

For shutting down the application simply go back to your command line terminal windown and press `Ctrl` + C

Future Tasks

Now that both the LaTeX and markdown parsers have been implemented a very big chunk of what needs to be done has been completed, although there is a lot of future scope for this application and what it can become and the possibilities for features is simply endless.

What can be accomplished in the little time remaining for this semester is that currently there is no provision in this application for multiple users and is simply acting to showcase the parsers that I have created. So, what can be done is a login page can be added where people can enter different user-names to open the application and then the messages that they send will also be persisted on a No-SQL based database such as Firebase.

This will make the application much more usable and also make it very production ready as the people will now genuinely be able to login and use it for chatting rather than just the facade to showcase multi-modal parsing that it is in it's current stages.

The technologies that will be required are:

1. Google Firebase
2. MongoDb

And the concrete features that can be added are:

1. A login page
2. Persistent messages
3. Mechanism to maintain unique usernames

Future Scope

The future tasks are something that accomplishing this semester may be feasible, but the Future scope is something that this application has the potential of becoming or achieving but isn't possible in the short time period of a few months or the remaining time before this semester.

The future scope of this application includes the following features that can be added to this:

For Casual Consumers

Casual consumers (non paying) consumers can simply go to our domain name www.subtext.com and log in using any valid EMail account or through any of the given OAuth interfaces (Facebook OAuth, Google OAuth, GitHub OAuth) After logging in they will have to create a unique username and also pass through basic formalities of creating a password ad entering a few security questions. The login process will be very simple and straightforward and will only require the person selecting a unique username (Only part which can potentially take time).

Using a unique username we can overcome a very big hurdle that was brought in by whatsapp. WhatsApp forces that identification happens through the cellular provider number, but it happens very often that there is no cellular provider number or that a signal is unavailable or most common that the number used for creating the account is no longer in use and has been transferred to a third party.

Using a unique username that is not tied down to any other authentication, but just an EMail authentication makes the experience stateless and doesn’t pre inforce any knowledge of the user or requirement to use a cellular enabled mobile device for this app.

Having created a username the users can then connect with their family and friends using each other’s usernames and use all of the advanced constructs that are available at their disposal.

For Organizations (Future Growth Ideas for product - Optimistic Venture)

Organizations are much more demanding and require much more granular control over users along with authorization rules as well. Organizations also have many products and team internally so the same user in one organization should have the ability to be a part of multiple Groups.

To register an organization a user once logged in can simply click on **Create an Organization** and will be redirected to create a unique organization name that will be used to manage all the Groups and Channels that will be created within an organization and also the users that will be added in said organization.

Let us say that the organization name be ***organization-name*** and so any person logging into the space of this organization can go to ***organization-name****.subtext.com/*. Every organization based on their unique name will receive a sub domain name on the main domain space. Once logged into the organization space, a user can create Groups or chat with individual users.

Group: Group is a collection of channels that can be created by any user.

Channel: Channel is a message communication stream that can be set up inside a group by the group admin or any user based on the authorization rules present in a particular group. A channel provides a message stream for a specialized topic and users present in groups can join various channels based on where they want to contribute and what they wish to be a part of. The addition of a user to a Channel can be further controlled using authorization rules set up by administrators during Group creation and also Channel creation.

Inside a group, a user can communicate in any channel where he is a part of and also communicate with individual users in the same group.

Each Channel inside the group will have a unique name which can not collide with the Global scope of all usernames. Similarly every group name will also be a unique name that can not collide with user names, but can with channel names.

The url for the above groups and channels will be as :

For Chatting inside a particular channel

https://organization-name.**subtext.com**/{groupName}/{channelName}

For chatting with another user in the group

https://organization-name.**subtext.com**/user/{userName}

Bibliography

1. Desmos (https://www.desmos.com/)
2. Google (https://www.google.com/)
3. Facebook (https://www.facebook.com/)
4. GitHub (https://www.github.com/)
5. Microsoft Azure (https://azure.microsoft.com/)
6. Amazon Web Services (https://aws.amazon.com/)
7. Circle CI (https://circleci.com/signup/)
8. Jenkins (https://jenkins.io/)
9. Angular (https://angular.io/)
10. Nest.js (https://nestjs.com/)
11. TypeScript (https://www.typescriptlang.org/)
12. LaTeX (https://www.latex-project.org/)
13. MarkDown (https://en.wikipedia.org/wiki/Markdown)
14. OAuth (https://oauth.net/)
15. Firebase (https://firebase.google.com/)
16. MongoDB (https://www.mongodb.com/)
17. CouchDB (https://couchdb.apache.org/)
18. Reddis (https://redis.io/)