Contents

[Chapter 1: Introduction 6](#_Toc72337814)

[1.1 Introduction 6](#_Toc72337815)

[Chapter 2: Literature Review 8](#_Toc72337816)

[2.1 Introduction 8](#_Toc72337817)

[2.2 Benefits of Online Learning [3] 8](#_Toc72337818)

[2.3 Difficulties in Online Mode of Learning [2] 9](#_Toc72337819)

[2.4 Microsoft Teams 9](#_Toc72337820)

[2.4.1 Benefits of MS Teams over other technologies: [4] 10](#_Toc72337821)

[2.4.2 Challenges of teaching and learning online with MS Teams: [4] 10](#_Toc72337822)

[2.4.3 Competitors Analysis of MS Teams 11](#_Toc72337823)

[2.5 PowerShell Command line interface vs. Graphical User Interfaces 14](#_Toc72337824)

[Chapter 3: Technical Review 15](#_Toc72337825)

[3.1 Introduction 15](#_Toc72337826)

[3.2 Backend 15](#_Toc72337827)

[3.2.1 Node.js 15](#_Toc72337828)

[3.2.2 Features of Node.js 16](#_Toc72337829)

[3.2.3 Node.js vs. Other Backend Technologies 16](#_Toc72337830)

[3.3 Database 17](#_Toc72337831)

[3.3.1 SQL vs. NoSQL 17](#_Toc72337832)

[3.3.1 MongoDB 18](#_Toc72337833)

[3.4 Frontend (User Interfaces) 19](#_Toc72337834)

[3.4.1 Angular 19](#_Toc72337835)

[3.4.1 Angular vs. Other Frontend Technologies 19](#_Toc72337841)

[3.5 Express.js 20](#_Toc72337842)

[3.6 Microsoft Graph APIs 20](#_Toc72337843)

[Chapter 4: Requirements 22](#_Toc72337844)

[4.1 Introduction 22](#_Toc72337845)

[4.2 Requirement Prioritisation 22](#_Toc72337846)

[4.2.1 Moscow Method 22](#_Toc72337847)

[4.2.2 *Must-have* 22](#_Toc72337848)

[*4.2.3 Should-have* 22](#_Toc72337849)

[*4.2.4 Could-have* 23](#_Toc72337850)

[*4.2.5 Won’t have* 23](#_Toc72337851)

[4.3 Use case 23](#_Toc72337852)

[4.3.1 Use case Description 24](#_Toc72337853)

[4.4 Non-functional Requirements 27](#_Toc72337854)

[4.4.1 Scalability 27](#_Toc72337855)

[4.4.2 Availability 28](#_Toc72337856)

[4.4.3 Recoverability 28](#_Toc72337857)

[4.4.4 Security 28](#_Toc72337858)

[4.4.5 Data Integrity 28](#_Toc72337859)

[4.4.6 Usability 28](#_Toc72337860)

[4.4.7 Interoperability 28](#_Toc72337861)

[4.4.8 Maintainability 28](#_Toc72337862)

[4.4.9 Reliability 28](#_Toc72337863)

[Chapter 5: Design 29](#_Toc72337864)

[5.1 Introduction 29](#_Toc72337865)

[5.2 Class Diagram 29](#_Toc72337866)

[5.2.1 Relationship between classes: 30](#_Toc72337867)

[5.3 Sequence Diagram 31](#_Toc72337868)

[5.3.1 Sign In 31](#_Toc72337869)

[5.3.2 Create Channel 32](#_Toc72337870)

[5.3.3 Add Member 32](#_Toc72337871)

[5.4 Activity Diagram 32](#_Toc72337872)

[5.4.1 Log In 33](#_Toc72337873)

[5.4.2 Create Channel 33](#_Toc72337874)

[5.4.3 Add member to channels 33](#_Toc72337875)

[5.5 Wireframes 34](#_Toc72337876)

[5.5.1 Sign In 34](#_Toc72337877)

[5.5.2 View All Joined Channels 35](#_Toc72337878)

[5.5.3 View Channel Members 35](#_Toc72337879)

[5.5.4 Add user into the channel 36](#_Toc72337880)

[Chapter 6: Implementation 37](#_Toc72337881)

[6.1 Software Development Methodology 37](#_Toc72337882)

[6.1.1 Waterfall 37](#_Toc72337883)

[6.1.2 Rapid Application Development 37](#_Toc72337884)

[6.1.3 Prototype Development 37](#_Toc72337885)

[6.1.4 Spiral Model 37](#_Toc72337886)

[6.1.5 Agile Development 38](#_Toc72337887)

[6.2 Work Distribution 39](#_Toc72337888)

[6.2.1 Gantt Chart 39](#_Toc72337889)

[6.3 Code Implementation 41](#_Toc72337890)

[6.3.1 Microsoft Authentication Library (MSAL) 41](#_Toc72337891)

[Code Snippets: 41](#_Toc72337892)

[Chapter 7: Testing and Evaluation 48](#_Toc72337903)

[7.1 Software Testing 48](#_Toc72337904)

[1. Create Team & Channel Web service 48](#_Toc72337905)

[2. Get Teams & Channels Web services 49](#_Toc72337907)

[3. Get Users of Organisation 50](#_Toc72337909)

[4. Add Member to Channel 50](#_Toc72337911)

[5. List Members of Channel 51](#_Toc72337912)

[7.2 Final Results 53](#_Toc72337913)

[7.2.4 List Channel Members 57](#_Toc72337914)

[7.2.5 List Organization Users 58](#_Toc72337915)

[7.2.6 Add User into Channel 58](#_Toc72337916)

[Chapter 8: Conclusion and Reflection 62](#_Toc72337917)

[8.1 Future Plan 62](#_Toc72337918)

[8.2 Conclusion 62](#_Toc72337919)

[Chapter 9: References 65](#_Toc72337920)

[9.1 References 65](#_Toc72337921)

**List of Tables**

[Table 1 : SQL vs NoSQL 17](#_Toc72340219)

[Table 2 : Use Case Table-1 23](#_Toc72340220)

[Table 3 : Use Case Table-2 24](#_Toc72340221)

[Table 4 : Use Case Table-3 26](#_Toc72340222)

[Table 5 : Use Case Table-4 27](#_Toc72340223)

[Table 6 : Classes Table 31](#_Toc72340224)

[Table 7 : Gantt Chart 44](#_Toc72340225)

**List of Figures**

[Figure 1 : Use Case 23](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340253)

[Figure 2 : Class Diagram 32](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340254)

[Figure 3 : Sequence Diagram 34](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340255)

[Figure 4 : Sequence Diagram-2 35](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340256)

[Figure 5 : Sequence Diagram-3 35](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340257)

[Figure 6 : Actvity Diagram-1 36](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340258)

[Figure 7 : Activity Diagram-2 37](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340259)

[Figure 8 : Sign In Wireframe 38](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340260)

[Figure 9 : Wireframe-2 39](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340261)

[Figure 10 : Wireframe-3 40](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340262)

[Figure 11 : Wireframe-4 41](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340263)

[Figure 12 : Create Team 53](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340264)

[Figure 13 : Create Team-2 54](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340265)

[Figure 14 : Get Team-1 55](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340266)

[Figure 15 : Get Team-2 55](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340267)

[Figure 16 : Get User 56](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340268)

[Figure 17 : Add Member 57](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340269)

[Figure 18 : Add Member-2 57](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340270)

[Figure 19 : Karma Testing 58](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340271)

[Figure 20 : Login 59](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340272)

[Figure 21 : Login-2 60](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340273)

[Figure 22 : Login-3 61](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340274)

[Figure 23 : Login Enter PAssword 62](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340275)

[Figure 24 : Dashboard 62](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340276)

[Figure 25 : Create Channel 63](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340277)

[Figure 26 : Channel Created 64](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340278)

[Figure 27 : My Channel 65](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340279)

[Figure 28 : Channel members 66](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340280)

[Figure 29 : Add User 67](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340281)

[Figure 30 : Request Submitted 68](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340282)

[Figure 31 : Delete Members 69](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340283)

[Figure 32 :Logout 70](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340284)

[Figure 33 : Select Account 71](file:///D:\Sandesh%20project\Endoscopy\dissertation.docx#_Toc72340285)

# **ABSTRACT**

Online meetings have become an essential part of virtual learning and a source for collaborating with instructors, students and professionals. Especially in pandemic, where only option to work and study is to create online channels or arrange meetings with team members and instructors. It is time consuming for lecturers to make channels and allocate students, for reporting they need to spend more time. The idea is to provide a Microsoft team’s tool by building a web application that allows user or lecturer to quickly create private channels and allocate students to them.

This Project will be created as a Microsoft team application. The projects aim is to use a number of technologies for developing this tool such as, Node.js as primary technology, yeoman workflow as a client-side stack, power shell scripts as server-side script to create channels in Microsoft teams, Microsoft Graph API for reporting features and finally, JavaScript’s frame work for building an interactive user interface.

The developed application can then be used by lecturers to create private channels and allocate students by using a user-friendly interface. Students data will be inserted into application one at a time. Lecturer will then login into the application, he/she will be able to create private channels and add students to them. The application will then generate and execute a power shell script when an appropriate action has taken place. Lastly, the application will also have a reporting functionality i.e. displaying a list of students which needs to be added to the application.

# **Chapter 1: Introduction**

# **1.1 Introduction**

The concept of E-Learning has always had its importance widely in the world. Besides, the outbreak of the novel Coronavirus has exposed the world to colossal challenges and simultaneously it has brought revolutionary changes in the world i.e.

1. It has revolutionized the concept of e-learning and made it possible
2. All the traditional physical modes of learning has been shut, and almost each school is shifting to online mode

In the above scenario, all the schools and organizations are struggling to tackle the consequences of shortcomings. The total closure of educational institutions in some countries affects more than 60% of the student body. Even in countries that have only implemented localised closures, this has always impacted millions of students. Hence, there needs to be such platforms that help schools/ companies achieve their specific goals. As a solution to these challenges, we have different platforms like: Zoom Meetings, WebEx, Google Classrooms, learning management systems such as Invas' Canvas Blackboard, and a support software Proctorio (a Google Chrome browser extension that can monitor students taking online exams). But the whole process of creating rooms, and arranging meetings is tiresome and time consuming for teachers/instructors. Besides that, it is too hard for many of the teachers to get used to technical tools and understand the somewhat complex workflow of the platforms along with maintaining their focus of conveying their teachings to their students. In that case, there should be designed one such platform that is:

1. User Friendly
2. Accessible in terms of that since, it is a web based platform so any teacher can access it easily using his browser
3. Affordable in terms of that it would not require much efforts of its users
4. Flexible, for the users can access it anywhere. The government also recognizes the increasing importance of online learning in this dynamic world.
5. Usable in sense that:
   1. It would have proper and well-designed interface and aesthetic features
   2. It has other Features such as: Enrolling in a course
6. Collaborative in between teachers and students
7. Maintainable due to its extensibility, adaptability and error correction
8. Efficient in its response time performance, page generation speed and graphic generation speed
9. Reliable in terms of that instructors would be verified and their input would be validated and recovered
10. Functional in a sense that it has:
    1. Searching and retrieving capabilities
    2. Navigation and browsing features
11. Responsive

To be able to use this interface, users will have to sign into their Microsoft accounts. This interactive web interface allows users to create new channels, add new members, and view all the added members as well. This interface is quite user friendly, simple and easy to use, and hence it reduces the overall complexity of traditional processes.

# **Chapter 2: Literature Review**

# **2.1 Introduction**

This project has its focus on the limitations faced in the online mode of learning since everything is shifting to online mode, may it be trading, banking, booking services, etc. It has made it necessary for schools to shift their mode of learning to some online platform else in this COVID they would be left behind. But shifting from traditional mode to e-learning is not as smooth as is our traditional mode. It is a quite tiresome process. [1] This literature review focuses on the limitations of Microsoft Teams, the competitor analysis of Microsoft Teams, many of the widely used technologies similar to that we have created, further discusses the limitations that are faced by admins during the creation of channels in Microsoft Teams because of using PowerShell, and covers the imperfection in different technologies that are eliminated by the development of this website that we have created. Besides, it illustrates how MS Teams are complicated to learn and use because of its User Interface.

# **2.2 Benefits of Online Learning** [3]

Since the outbreak of novel Coronavirus, it has made it compulsory for online platforms to be accessible where:

1. It is possible to hold video conferences with at least 40-50 students
2. Discussions can take place with the students to keep the organic classes
3. The Internet connections are good
4. The courses are also accessible on mobile phones and not just laptops
5. The ability to view the courses already recorded
6. Comments snapshots can be obtained from students and can be tracked assignments. [12]

Much of the world is under quarantine due to the severe outbreak of this global Covid-19 pandemic and as a result, many cities have been turned into ghost towns and its effects can also be seen in schools, universities and schools. Universities affected by the circumstance can now benefit from online learning and tools provided online can be called a panacea for the crisis. The evolution of online, distant learning has revolutionized the concept of global learning, i.e. students can use online learning platforms and have access to discussion forums where they can interact with other students of different countries or continents and share their knowledge with one another. This entire learning process is efficient in terms of being student-centric. Also, students can easily learn anywhere and anytime they want to learn. So they must be provided an easy platform where they can easily access the course content without being frustrated due to the complex -User Interfaces of online platforms. Microsoft Teams, Google meet, Edmodo, and Moodle are used as learning management systems along with their applications for video conferencing. [6] Some other widely used online learning and meeting technologies are Zoom, Skype mostly used in case of business meetings, WebEx and Adobe connect, etc.

# **2.3 Difficulties in Online Mode of Learning [2]**

Though the online mode of learning has so many advantages like the number of students can be increased, while keeping the overhead the same. Yet some disadvantages remain unresolved like:

* The quality of instruction: The quality of instruction is affected by the negligent attitude of the teachers hence the quality of instruction is not much effective because the teacher's attention is always absorbed in resolving technical issues or in understanding different platforms and their functionality.
* Teachers are unable to test students based on knowledge assessment, nor were students evaluated based on their practical knowledge. The entire process of teaching and knowledge was based on theory. [7]
* Problems with the platforms: Not all the teachers are comfortable with interfaces of different platforms and find the operating of applications quite complex.

# **2.4 Microsoft Teams**

Features of Microsoft teams that are quite usable and efficient to make online learning more fun and functional. [3]

1. Online channels: Online channels are the ones that contribute to the usability of Microsoft teams on online learning platforms. Because of the channels allowing students to exchange course-related documents and work boards among one another.
2. Chat in channels: Members who add to a group can also add and see other chat forums to their shared channel.
3. File storage on SharePoint: Whatever files you share are to be stored on default storage that is SharePoint Online. This site is used a document storing and it creates private folders on site automatically for different files.
4. Online video calls and screen sharing: Microsoft Teams allow its users to arrange video calls through which users can communicate with their peers and instructors in real time and share required resources and their screen in case they need it.

In addition to all this, users can invite people to chat using @, connect with your teammates in real-time, easily find the messages and files you sent using Search, record audios, videos and screen sharing presented during the meeting and view later and supports integration with many third-party applications.

## 2.4.1 Benefits of MS Teams over other technologies: [4]

**Participants limit:** Zoom for free access allows you for 100 participants whereas Microsoft 365 members with an E3/E5/A3/A5 subscription can support admin to have 1000 listeners and 2000 attendees.

**Functionality:** Zoom just allows video calling and meeting or conference calls, whereas  Microsoft Teams is based on that it allows chatting in general and application organization that we can also experience in Slack or Discord. Not only all this, but Microsoft teams are designed to integrate it with Microsoft’s 365 apps which cover different Microsoft applications like MS Word, MS Excel MS PowerPoint,  and MS OneNote. However, Microsoft Teams gives us a complete way of organizing for team management on online platforms, resolves chatbox needs, and allows for sharing files, video conferencing, and so much more**.**

## 2.4.2 Challenges of teaching and learning online with MS Teams: [4]

* The User Interface that is provided by MS Teams is quite challenging and complex to be used by a layman or someone who is not much familiar with technology, all this creates confusion among its users. Users should be able to quickly interact with different people in different channels and Teams, share files, and make use of Office 365 applications baked into Teams. This is the reason why instructors find it difficult to make their teaching smoothly.
* MS Teams show bad performance on init generally after prolonged use.
* MS teams show bad user management because you cannot view all the members in a meeting
* MS Teams has no public channels
* There is challenging onboarding in MS teams
* Channels cannot be moved to teams but if one wants to do it then he can do it manually
* While searching for a folder uploaded in a channel requires you to go to the root folder of the channel. All this is just so confusing for a layman. In any condition, if a file is organized at a new  folder then its link from the root folder to the file would get broken
* Managing virtual groups gets quite difficult for teachers who are not much familiar with the technology. Due to this, creating channels from PowerShell, managing teams is so burdensome, and adding students manually to those channels consumes so much of the time of teachers and ultimately reduces the productivity of that teacher.

## 2.4.3 Competitors Analysis of MS Teams

Many people want professional instant messaging software to help them with their presentations. The hard part is figuring out if the software you choose is right for you. A list compiled of enterprise instant messaging software that reviewers voted the best overall compared to Microsoft Teams. Based on reviewer data, you can see how Microsoft Teams stacks up against the competition, check the reviews of current and former users, and find the best fit for your business. Cisco Jabber, Cisco WebEx Teams, Bitrix24, and Rocket. Chat is Microsoft Teams' most popular alternative and competitor. Microsoft Teams being a chat tool provides interaction and collaboration among its users. But due to its crucial environment, users switch to its alternatives. Below we have given some of the alternatives to Microsoft Teams.

**Slack**

Slack is a competitor to Microsoft Teams, Slack was bought by Microsoft and then Teams was cloned keeping in mind the functionality of Slack. Slack is user-friendly and simple which gives its user the best experience while using it because it is easy to operate. Also that, it has the functionality of segmenting the real-time messaging into channels, video calls, screen sharing, and media sharing. “Slack has such a foothold that if corporate teams are looking for a chat tool, they’re probably going to choose Slack,” says Kevin Senior, managing ex-director at Glasscubes. But Slack restricts the user to one-on-one video calls for free, 10 integrations, and access to 10,000 messages. [8]

**Google G Suite**

G Suite can be used for document management, Google Chat (previously termed as Hangouts) for the real-time chatbox. Google Meet can be used for video conferencing, Drive can organize and store your media, Google Docs can also be used for document editing, and Gmail can be used for sending mail. Nevertheless, Teams allow us channel creation, mentions, and so much more that is not provided by Google. [8] Google Classroom allows the integration of your basic desktop applications: Google Docs, Sheets and Slides. And more Google Classroom provides access to Google Drive, considered one of the best collaboration tools in the cloud by several technical reviewers. [13]

**Workplace by Facebook**

Workplace has the same User Interface as Facebook, users can see the News Feed, send messages through messenger, join desired groups, live videos, and video calls for the setting of their work. Workplace offers three subscription options:

* Essential: it is free
* Advanced: it is paid
* Enterprise: it is paid and has additional features like security

Since Workplace resembles Facebook’s consumer product, so this may be the reason, some of the users not preferring Teams, Slack, G Suite over Workplace. [8]

**Flock**

Flock allows chat, video, voice calls, and admin does all this by converting messages into tasks that we assign and keep track of, we can also set reminders, arrange polls to get feedback from other participants. Like Teams, it allows us to create channels private, public, one-way channels to broadcast and smart channels as well. Using smart channels, one would be able to be added into a channel based on certain criteria.

The amazing thing about it that it integrates 60+ applications. But Flock allows 10000 messages, video calls for four users with a duration of 20 minutes along with that it gives 5 GB of storage for teams. While there are other paid plans from Flock for its users. [8]

**Zoho Connect**

Zoho Connect is also a teams based software application through which you can collaborate and communicate with your teams. You can provide your team a shared platform where they can share their ideas and opinions with their groups. Besides, you can choose the privileges you need to assign to a group like open, closed, public, and private. In this software application, Feeds keeps a record of the conversations and lets you have a detailed view of the desired conversation that is relevant to you. You can also create posts, attach media to your posts, and share posts that you created or other people's posts on your personal or group wall. All this creates an interactive environment for workers. You can start video conferences directly within a group, set important discussions to easily access and disable channels that are not important to you, and upload files and create documents, presentations, and more. In the software and with multiple team members to collaborate. Real-time. It has a more complex user interface, and the overall design is not attractive. Standard price: $ 1 per user per month.

**Cisco WebEx Teams**

Cisco WebEx allows the functionality of audio calling, video calling and meeting, chatting, scheduling the meetings, media sharing, creating whiteboards, and visual collaboration tools.

Cisco WebEx Teams provides an AppHub and integrates different tools of famous service providers for enterprises like Google, Dropbox, Microsoft, and Salesforce. Not only that, but WebEx Teams have the functionality of bots with the help of which one can automatically perform workflows and also provides developers quite useful resources to make their system more user-friendly by required customization.

Cisco WebEx Teams focuses more on enterprise meeting organization through video calling instead of just chatting, so it has different paid plans like $13.50 per month, per host, these plans allow almost 200 attendees in a meeting and add up to 100 host licenses. [8]

All of the collaboration applications mentioned above are time-consuming, difficult to manage, and expensive due to shipping costs. Therefore, a student from an underdeveloped country cannot afford these platforms. The developed project will be easy to use, free, and will take less time to manage. Once instructors create their accounts, they will add a student to you, then there is no need to add that student repeatedly, students can easily access the resources. The application will be an efficient, user-friendly, and interactive interface.

# **2.5 PowerShell Command line interface vs. Graphical User Interfaces**

Since our project is designed for ease of admin so that the simple and usable interface of our website could allow its admin to manage his class easily, without being trapped into the frustration of not understanding the complicated commands of PowerShell. Also because: [10]

* Often admin finds it difficult to understand the environment of PowerShell because it is not visually appealing, it is based on textual commands. For each operation, you want to execute, you need to memorize its command and that’s where most of the users switch to Graphical User Interface. Since User Interface visualizes menus and buttons, it is easy to use, quick to remember.
* A Graphical User Interface has the same accessibility of files in terms of control as PowerShell, that’s why new users often prefer GUI to command line.
* With the help of GUI, one can perform multiple tasks at the same time whereas performing multiple tasks at the same time in the command-line is not as easy as in Graphical User Interface.
* Whenever there is to be any addition of computing resources like a server is to be added into your environment, in that scenario we make use of Graphical User Interface because it is easy to use, navigate and manage the system without having to worry about the commands.

Considering the entire problem domain, we have created a simple Graphical User Interface based website where admin can sign into the website using his Microsoft account credentials. Then admin can create a channel, add a user to channel, etc. using the User Interface of this website. The User interface of this application is so simple and easy to use for a layman that he can use it easily even if it is his first time interacting with the website. Thus this website enhances the usability of this website. The next thing that we have covered is to turn the admin operations of PowerShell Command Line to UI based which saves the time of the user without having to remember the commands to create a channel. Admin can also delete a particular channel and remove a student from the channel using this website.

# **Chapter 3: Technical Review**

# **3.1 Introduction**

The technical review of this web application project covers the entire architecture of this web application. Web application Architecture tells us how our web application allows its different components to interact with one another and what would be the distribution of the application logic among the client-side and server-side. So for this web application project, we have implemented making use of Backend, and Frontend technologies.

# **3.2 Backend**

Backend term is used when we are going to deal with server-side CRUD functions with databases and all server computation and logic. The backend of any application allows for backlogging and processing the syntax and operating data. Generally, we don’t allow users to access the backend of the web application. To make our websites work, there needs to be a successful interaction between the backend and frontend of the web application. All the steps included in the successful communication of frontend and backend are given below:

* User requests server for CRUD functions
* That user request for CRUD operation is done through HTTP web service
* There is a different endpoint for specific requests
* A request includes its body parameters
* The server receives requests and processes the requests and then responds to the user

Below is given the technology that we have used for the backend. Backend is how our web application processes all the information, contrary to this is the frontend, which is an interface between the user of the web application and its backend.

## 3.2.1 Node.js

We have used Node JS as a backend technology. For all the operations of the server-side and those said above, Node JS is used in this project. Node JS is an environment, which allows different frameworks to run within it. We have used Node JS here as the core technology because it is open-source, cross-platform, and it has a back-end JavaScript runtime environment that executes JavaScript code independent of a web browser. The reason why we have used Node JS is that it has a single-threaded nature which allows non-blocking and event-driven servers. This is the reason why Node JS is so lightweight, efficient, and compatible with different devices for computation on the server-side. That’s why we have used Node JS for server-side computational resources.

## 3.2.2 Features of Node.js

**Requests:** When a user requests for any operation he wants to perform on a website, that request can be blocking (complex) or non-blocking (simple).

**Node.js Server:** Node.js server is a server-side platform which process request in a sequence:

* First user requests and Server takes the user request
* Secondly, the request is processed
* In the last, the server responds to the request of the user

**Event Queue:** To manage incoming client requests, the server of Node.JS creates an Event Queue where all the incoming requests are stored and then Event Loop takes all that request from Event Queue one by one.

**Thread Pool:** All the threads that need to be processed for fulfilling a client request resides in the thread pool.

**Event Loop:** Event Loop keeps on receiving the client requests and processes all the incoming requests. After processing the requests, it returns to the client

**External Resources:** Some external resources like computing resources, storage resources are required to block client requests.

## 3.2.3 Node.js vs. Other Backend Technologies

Though there are so many other backend technologies like, PHP, .Net, Java, etc. but all these technologies have dependencies that make them less suitable to be used for this project. Below we have given some of the drawbacks of different technologies.

* **.NET**
  + It is limited in terms of that it makes use of entity framework
  + Any development that you do with it, would be dependent on Microsoft for future development
  + Migration of apps to .NET is always expensive
* **ELIXIR**
  + It is less scalable than NodeJS because using Node JS we can easily get your product on your mobile phone.
* **PHP**
  + It is less secure due to its being open-source and ASCII text file being easily accessible
  + It is not suitable for applications that are extensive and lengthy
  + It cannot allow us to modify the behaviour of an online application
  + It does not allow debugging to find errors or warnings in the program

# **3.3 Database**

Database provides us storage where we can store our files, and content related to the web application. There are two types of databases one is relational database (SQL) and other is non-relational database (NoSQL).

## 3.3.1 SQL vs. NoSQL

|  |  |  |
| --- | --- | --- |
|  | SQL | NoSQL |
| Relational | SQL is relational Database | NoSQL is relational Database |
| Querying | SQL database has a structured type query language to perform database operations | Uses other than querying languages for performing operations |
| Schema Type | SQL has a predefined schema for creating database of structured data | NoSQL has dynamic schema for creating database of unstructured data |
| Scalability Type | SQL is vertically scalable | NoSQL is horizontally scalable |
| Structure for storing data | Tables are used to manage SQL Database | Graphs are used and key-value pair is managed |
| Type of data | Better for unstructured data where they can perform operations on multiple rows for transaction | Supports unstructured data |
| Architecture | Wide Column Store | Document Store |
| Examples of databases and their limitations. | Oracle  Using Oracle is expensive in terms of hardware requirements, management, and maintenance  MYSQL  Hot backup, custom data types, procedures, and triggers are not supported by MYSQL  Slow updating makes it unsuitable for the project.  Transactions are not according to ACID-compliant  The system catalogue can be corrupted by a server crash. | MongoDB, Cassandra, HBase |

Table 1 : SQL vs NoSQL

## 3.3.1 MongoDB

Unstructured, schema-less type of data is stored in multiple collections and nodes in MongoDB and it does not burden us to need fixed table schemas, also allows limited join queries. Using MongoDB, we can add more many machines into your layers of resources because MongoDB makes use of scaling it horizontally which is the reason for its being more scalable. Whereas the relational database is generally less scalable due to its being vertically scalable. In case of increased load on a relational database in vertically scaling the data, server hardware power is increased or expensive and bigger serves are used to scale the entire database.  Though there are so many other databases like, Oracle, MYSQL, etc. but all these technologies have dependencies that make them less suitable to be used for this project. Below we have given some of the drawbacks of different technologies of NoSQL database.

* **Cassandra Drawbacks**
  + Inconsistency of data stored due to replicas, because architecture of Cassandra is distributed.
  + There can be overloading of the nodes in case of any node in a cluster goes down
  + Due to overloading, replicas can be lost
* **HBase Drawbacks**
  + Master Slave architecture can be a loophole to some of the client because failure of master node can result in data loss from both the ends.
  + One must have to learn JRuby-based HBase shell and technologies like Apache Hive (based on MapReduce) because HBase belongs to NoSQL database and it does not allow querying
  + It is interdependent for storage on others so it is less usable

# **3.4 Frontend (User Interfaces)**

The front end is how our website appears on the client’s side. The frontend is designed in such a way to:

* Make the website look great
* Build the website to load faster
* Increase responsiveness and device portability

## 3.4.1 Angular

Angular is a front-end framework that allows user interactions within the browser. Because of certain characters given below, we used them in our project:

### Typescript: Angular is a type scripting language, Typescript code allows debugging by creating just proper map file during the development in the browser or an editor. It also supports improved navigation, refactoring, and code auto-completion services.

### Declarative UI: Angular uses HTML for the UI of applications. Since HTML is a declarative language, Angular has a declarative user interface.

### Easy Testing: With module separation, the necessary services can be loaded while effectively performing automatic testing unlike the “one file-one module” principle.

### Simplified MVC Pattern: Angular provides less coding, and creating lighter and better apps because it does not burden us to split an application into different MVC components and build a code that could unite them.

### Ease of Maintenance: Code is easily maintained and updated efficiently due to decoupled components being replaced.

## 3.4.1 Angular vs. Other Frontend Technologies

Though there are so many other frontend technologies like, ReachJS, Vue.js, etc. but all these technologies have dependencies that make them less suitable to be used for this project.

Below we have given some of the drawbacks of different technologies.

* ReactJs
  + It is hard to adopt all the rapid changes in the updates since the process is too fast
  + Because of the high pace of development, It is hard to keep up with the updated documentation
* Vue.JS
  + It doesn’t allow scalability and plugins
* Ember.JS
  + It is too complex to understand the framework
  + The framework is highly adamant

## 3.5 Express.js

Express JS is extensively used for creating web services, web applications, and APIs quickly and easily, and also the whole process requires minimal development efforts, because Express JS is a free and open-source backend web application framework for Node. JS. Whereas Meteor.js has a dependency upon MongoDB for the database and uses global variables, both of which increase its development time. Furthermore, Meteor.js doesn’t use an npm-type package manager for the management of its JavaScript runtime environment. This is the reason which makes it better for projects that don’t need to be up and executing quickly. Therefore, we made use of Express JS to reduce the overall burden.

## 3.6 Microsoft Graph APIs

Microsoft Graph is a Restful web API that lets you have access to Microsoft Cloud service resources. After you have registered your application and obtained the authentication tokens for a user or service, you can send requests to the Microsoft Graph API. But, the way that conditional access policies are applied to Microsoft Graph is changing.

We have made use of some of APIs of Microsoft Graph API which are:

1. Login: It is used to authenticate users based on their Microsoft account credentials
2. Get Teams: Gets us the list of all the teams that user has in his Microsoft account
3. Get Channels: Gets us the list of channels that user has in his Microsoft account
4. View Channel Members: Gets us the list of member that a channel has in the user’s Microsoft account
5. View Organization Users: Gets the email address and name of the members of the organization
6. Add members to channel: helps to add a member to a channel

# **Chapter 4: Requirements**

# **4.1 Introduction**

The software requirements define what our software is going to perform and what features it needs to have. Requirements can also stand for what users of a system expects of it. From the perspectives of a clients, there can be different types of requirements like known or unknown, obvious or Hidden, expected or unexpected.

# **4.2 Requirement Prioritisation**

Requirement prioritisation is applied in Computer software product management for verifying which entrant conditions of a system product should be incorporated in a specific delivery. Moscow method will be used for prioritising requirements.

## 4.2.1 Moscow Method

Moscow prioritisation is also acknowledged as the Moscow method or Moscow analysis, IT is a standard ordering practice for handling conditions. The method is frequently applied to help crucial sponsors comprehend the impact of plans in a delivery.

It has the following three main components to discuss:

## 4.2.2 *Must-have*

It has secure Session login in Microsoft Team system

* User can see list of channels
* User creates channel
* User can see list of channel members
* User add members to channel
  + User can add member from the user list found in local database
  + User can add member via Microsoft-teams account email address or username.
* Development have a flexible and sustainable web application
* Produce monthly progress report
* Fulfil proposed deliverables
* Minimise the development and operational cost

## *4.2.3 Should-have*

* Should have Interactive interface
* Should be Easy to use for naïve users

## *4.2.4 Could-have*

* Allow users to join channel via invitation link
* Local database to store join request or other necessary information

## *4.2.5 Won’t have*

* Won’t have Mobile application version

# **4.3 Use case**

To understand and learn more about system requirements, we make use of a methodology that is use case, which allows us clarify, analyse, and manage our system requirements. The use case includes different set of interactions that user performs with the system and different users in the scope of specific scenario/ environment or related to a specific goal.

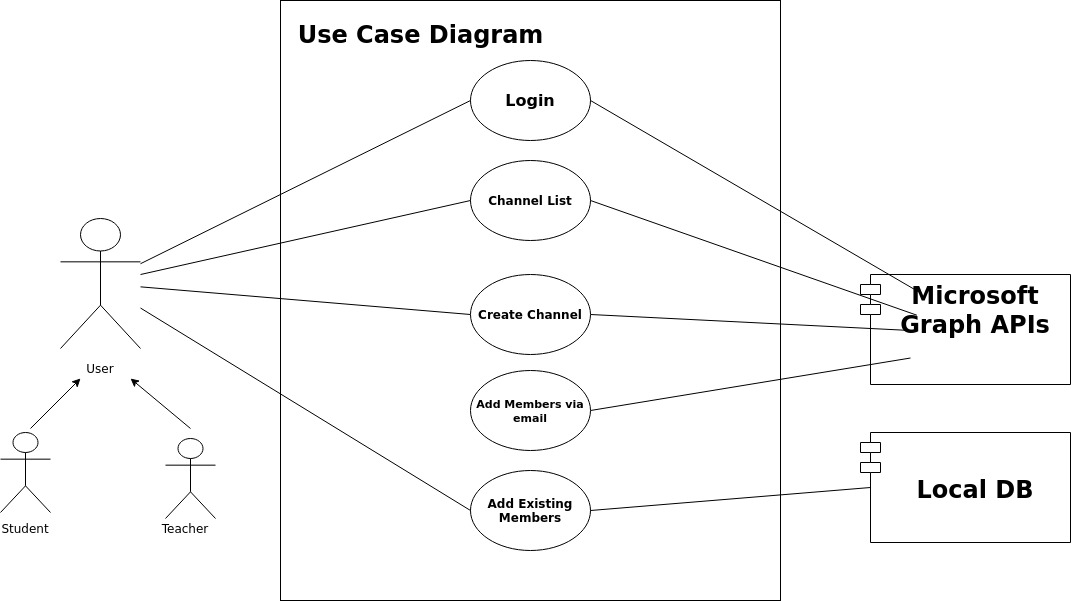


Figure 1 : Use Case

## 4.3.1 Use case Description

|  |  |
| --- | --- |
| **ID:** | **Login** |
| **Title:** | Login |
| **Description:** | This use case allows users to login to the system. To login to the system, users need to have a Microsoft team’s account. Using the credentials of Microsoft account, users will be able to use the system. |
| **Primary Actor:** | User |
| **Preconditions:** | 1. User must have a Microsoft teams account |
| **Postconditions:** | 1. The system displays a dashboard |
| **Main Flow  Success Scenario:** | 1. The user clicks the login button 2. The screen redirects to Microsoft graph authentication API link 3. The user enters his credentials (i.e. email & password) 4. The MS graph API validates the user credentials 5. The system displays the dashboard 6. The use case ends |
| **Alternative Flow** | 3a Missing email ID or password   1. The system prompts for email ID or password 2. Use case resumes at main flow step 1   3b Invalid Microsoft account credentials   1. The system displays the invalid credentials error message 2. The system prompts for email ID or password   3c Not any Microsoft team account   1. The system displays the error message (given by MS graph API) to the user regarding to that there is no such account 2. The systems redirects to the login page. |

Table 2 : Use Case Table-1

|  |  |
| --- | --- |
| **ID:** | **CreateChannel** |
| **Title:** | Create Channel |
| **Description:** | This use case allows users to create channels in the system to let students join the channel or to add students in the channel. To create a channel in the system, users must be logged in the system. |
| **Primary Actor:** | User |
| **Preconditions:** | 1. User must be logged in successfully |
| **Postconditions:** | 1. The user can view the channel in the channel list |
| **Main Flow  Success Scenario:** | 1. The user enters the dashboard on successful login 2. The user selects “**Create New Channel**” 3. The system prompts for Channel Name and Channel Description 4. The user enters the channel name and description 5. The user submits the channel name and description 6. The use case ends |
| **Alternative Flow** | 4a Missing channel name and description   1. The system prompts for channel name and description 2. Use case resumes at main flow step 1   4b User enters pre-existing channel name   1. The system displays “channel name already exists” 2. The system prompts for channel name |

Table 3 : Use Case Table-2

|  |  |
| --- | --- |
| **ID:** | **ViewChannels** |
| **Title:** | View Channel List |
| **Description:** | This use case allows users to explore the list of channels they have. Users can see the list of channels they are added in. Also, the users can see the list of channels they have created. |
| **Primary Actor:** | User |
| **Preconditions:** | 1. User must be logged in successfully |
| **Postconditions:** | 1. The user can select the channel. 2. The user can further see the list of users added in selected channel |
| **Main Flow  Success Scenario:** | 1. The system retrieves the channel list and their description from the MS graph API 2. The system displays the channel list and their description from the database 3. The user can further select the channel and view the list of users added into that channel 4. The use case ends |
| **Alternative Flow** | 1a MS Graph server down or API does not work for some reason   1. The system displays no any channel in the table 2. The system indicates an error message above the channel table. |
| **ID:** | **AddMembersByEmail** |
| **Title:** | Add members via Email |
| **Description:** | This use case allows users to add students into a channel via their email. To add a student in a channel, users must be logged in the system. |
| **Primary Actor:** | User |
| **Preconditions:** | 1. User must be logged in successfully 2. User must have a valid Microsoft team account email id to enrol in the channel |
| **Postconditions:** | 1. The system adds the user to a channel 2. The system updates the users list in database |
| **Main Flow  Success Scenario:** | 1. The user enters the email of student 2. The user submits the email 3. The system verifies the details of the student using Microsoft graph API 4. The system enlist the details of the account like:    1. Username    2. Email ID 5. The user submits the request to add the student to channel 6. The system send the request to MS teams via MS graph APIs to add new user into the channel 7. The system receives the response from MS graph APIs. 8. The system updates the database with a new user. 9. The use case ends |
| **Alternative Flow** | 3a Invalid email of student   1. The system shows error message 2. The system prompts for a valid email address 3. Use case resumes at main flow step 1   6a User enters student name pre-existing in to database   1. The system displays “User Already Exists” 2. Use case resumes at main flow step 1   6a MS Graph server down or API does not work for some reason   1. The system indicates an error message, retrieved by the API endpoint, to the user. |

Table 4 : Use Case Table-3

|  |  |
| --- | --- |
| **ID:** | **AddMembersFromList** |
| **Title:** | Add existing members |
| **Description:** | This use case allows users to add a new user (i.e. students) into a channel via selecting a member from the user list retrieved from the database. |
| **Primary Actor:** | User |
| **Preconditions:** | 1. User must be logged in successfully 2. User must select a channel to add a new user 3. System should contain a list of users in the database. |
| **Postconditions:** | 1. The system adds the user to a channel |
| **Main Flow  Success Scenario:** | 1. The system retrieves the list of users from database 2. The system displayed the list in the table 3. The user selects a new user from the list to add into the channel 4. The system request the MS Graph API to add the user into the channel 5. The system receives the response from MS graph APIs. 6. The system updates the database with a new user. 7. The use case ends |
| **Alternative Flow** | 1a Inconsistency in database   1. The system displays empty table to the user 2. The system shows the error message to the user.   1b User list not found   1. The system displays the empty table to the user   4a MS Graph server down or API does not work for some reason   1. The system indicates an error message, retrieved by the API endpoint, to the user. |

Table 5 : Use Case Table-4

# **4.4 Non-functional Requirements**

## 4.4.1 Scalability

The system will be deployed into the cloud having sufficient performance resources. The system performance with limited resources can be decreased if the number of users or traffic to the website increases. However, cloud gives us features to increase processing speed & storage at any point, without loss of any data, and take additional charges according to new performance resources.

## 4.4.2 Availability

The web dashboard would be available to users whenever they have an internet connection. In low latency conditions, the system provides media content in the shortest delay possible.

## 4.4.3 Recoverability

There will be a CRON JOB (scheduler) that will run periodically to backup the local database so the data can be recovered in case of any loss.

## 4.4.4 Security

This web app has protocol for Microsoft graph API integration, encryption and embedding of other resources. Also, the system will use the https web protocol to secure the communication between frontend and backend.

## 4.4.5 Data Integrity

The users need to authenticate themselves before using any function (requesting to join a channel or adding a channel) of the system.

## 4.4.6 Usability

The simplified interface of this web app increases its usability. So teachers can easily interact with the web app and create channels and add students to the channel. Also, students can request to join a channel easily by just clicking on a button. Apart from that, this application allows users an intuitive navigation bar between creating a channel and viewing existing channels.

## 4.4.7 Interoperability

The system will be able to exchange information security to/from the MS Graph APIs and/or LocalDB.

## 4.4.8 Maintainability

This web app will be maintained timely to resolve the issues or errors if found in a real execution environment.

## 4.4.9 Reliability

Since this web app is quite efficient in terms of availability, so the probability of unavailability, failure occurrence rate is too low.

# **Chapter 5: Design**

# **5.1 Introduction**

The Software Design is used to provide a well-defined description of the system design that is full enough evaluated so that the developer could proceed to the development of the system with an understanding of what is to be built and how that is to be built.

Our web application will allow its users to sign into their Microsoft accounts. Through its interactive web interface, users will be allowed to create new channels, add new members, and view all the added members as well.

From the requirements of our project, we could come to know that our web application should achieve below given requirements:

* A secure Session login in the Microsoft Team system
* User can see list of channels
* User can create a channel
* User can see list of channel members
* User add members to channel
* User can add member from the user list found in local database
* User can add members via Microsoft-teams account email address or username.
* Produce monthly progress report  Fulfil proposed deliverables
* Should have Interactive interface
* Should be Easy to use for naïve users

# **5.2 Class Diagram**

It helps to make a conceptual modelling and detailed modelling to understand how the structure of the application is. It describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because these can be mapped directly with object-oriented languages.

|  |  |  |
| --- | --- | --- |
| Class Name | Attributes of the class | Methods of Class and Description |
| Team | Team ID, Name, Description, Owner, Member | Create a Team: Teams can be created   Join a team: Teams can be joined |
| Channel | Channel ID, Team ID, Name, Description, Owner, Member | Create a Channel: A channel would be created  Add Members: Members would be added |
| User | First Name, Last Name, Email, Password | Login: The user can log into his account using email and Password |

Table 6 : Classes Table

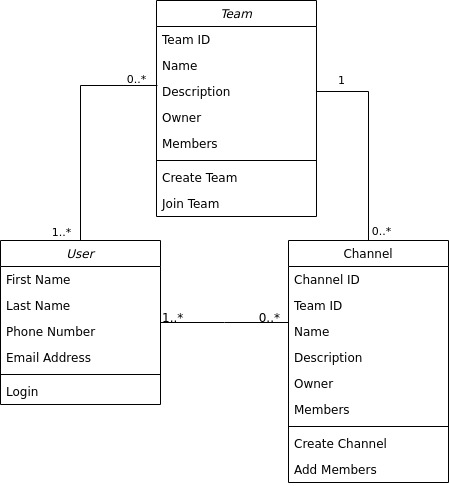


Figure 2 : Class Diagram

## 5.2.1 Relationship between classes:

A user can have any number (0, 1, 2,) of channels in his account. A user may have joined one or more than one or no channels at all. There can be zero to any number of channels for a team.

# **5.3 Sequence Diagram**

These diagram capture the interaction between class objects in the context of a collaboration. These are time focus, they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when. Below diagrams explain the interaction between class objects in an order in which the actions of the flow take place in the web application.

## 5.3.1 Sign In

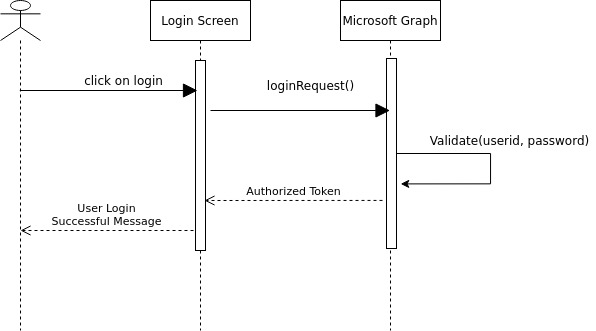


Figure 3 : Sequence Diagram

## 5.3.2 Create Channel

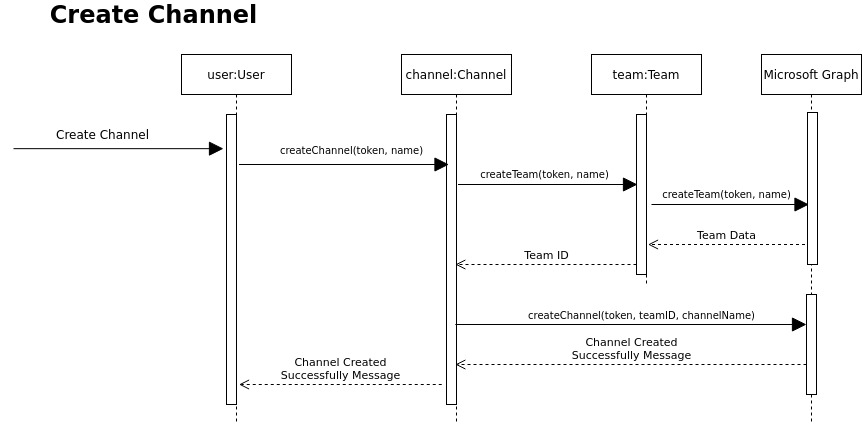


Figure 4 : Sequence Diagram-2

## 5.3.3 Add Member

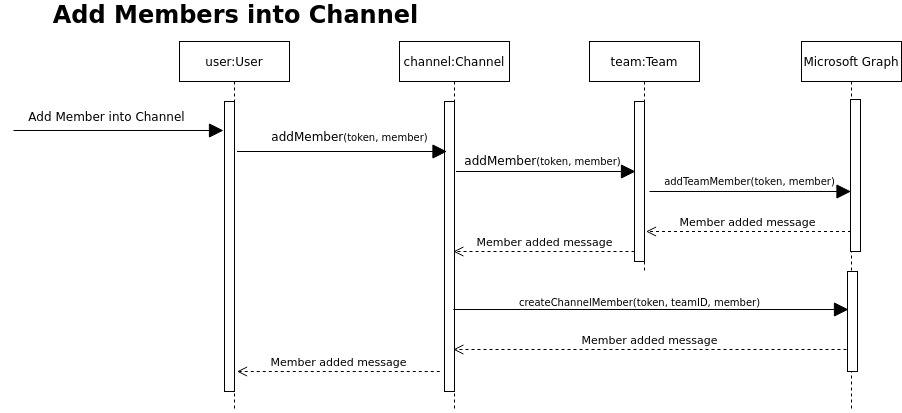


Figure 5 : Sequence Diagram-3

# 

# **5.4 Activity Diagram**

Activity diagram is another essential diagram to describe the key aspects of the system. Activity diagram is basically a flow of system to stimulate application flow from one activity to another.

## 5.4.1 Log In

## 5.4.2 Create Channel

Figure 6 : Actvity Diagram-1

## 5.4.3 Add member to channels

Figure 7 : Activity Diagram-2

# **5.5 Wireframes**

Wireframes are the simple layouts of black and white colours (kind of sketch) that outline the placement and size of html page elements, website features, and navigation for the website. These do not include the font choices, colours, logos or any real design elements that take away from purely focusing on a site's structure. Wireframe demonstrates the layout of a webpage and depicts what buttons and menus would be available on a webpage.

## 5.5.1 Sign In

Firstly, it asks the user to sign into the application using his Microsoft account. When the user clicks on the button here, he gets redirected to the MS Graph API link, where the user provides his email and password. The MS Graph API then validates the user based on his given credentials. If the user has provided correct email and password, he gets redirected to the dashboard. However, if the user has not provided the correct email and password as in MS Graph API then the system displays the error message and user gets redirected to the same sign in page.

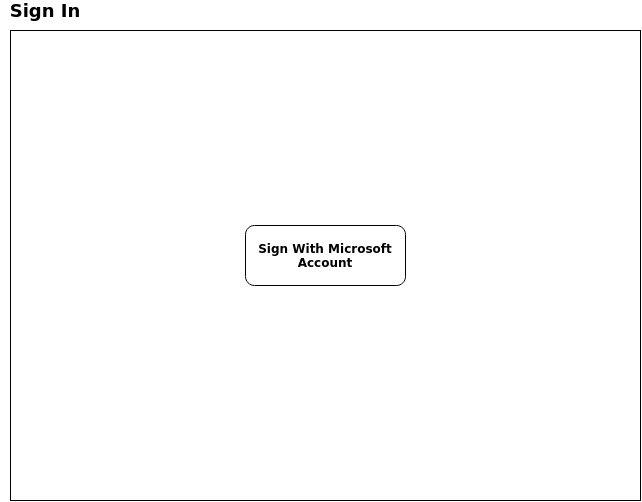


Figure 8 : Sign In Wireframe

## 5.5.2 View All Joined Channels

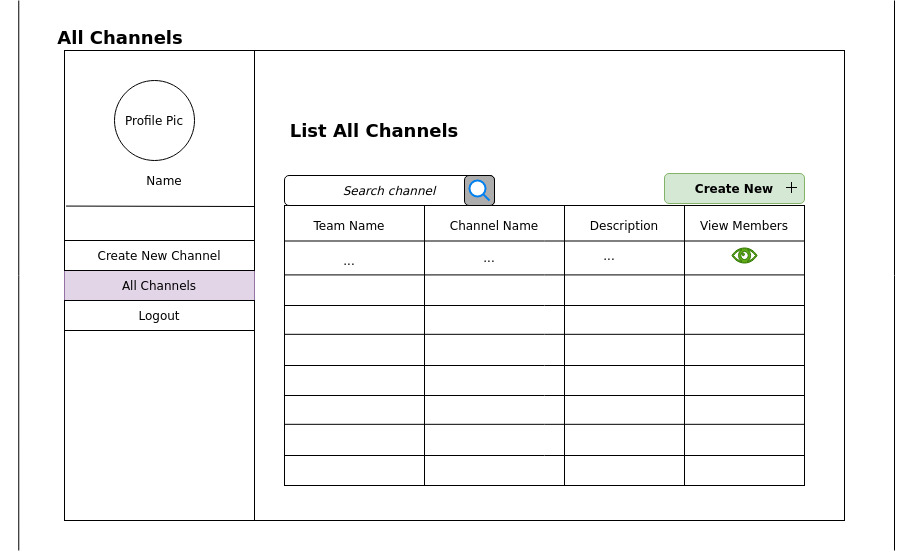
With this wireframe, the user would be able to see the created channels or the channels he is enrolled into. This list of channels would be retrieved from the MS Graph API. The web application would display the list of channels, their names, channel description, and the team name from the database. Also that its user can view the list of members enrolled in that channel. But in case of failure of retrieving the channel list from MS Graph Server API, the web application does not show any channel list. Besides that, an error message would be indicated above the empty channel list.

Figure 9 : Wireframe-2

## 5.5.3 View Channel Members

With this wireframe, the user would be able to see the list of members. This list of members would be retrieved from the MS Graph API. The web application would display the list of members, their names, and their email.

Also that its user can remove a student from the list.  But in case of failure of retrieving the member list from MS Graph Server API, the web application does not show any member list. Besides that, an error message would be indicated above the empty member list from the API endpoint.

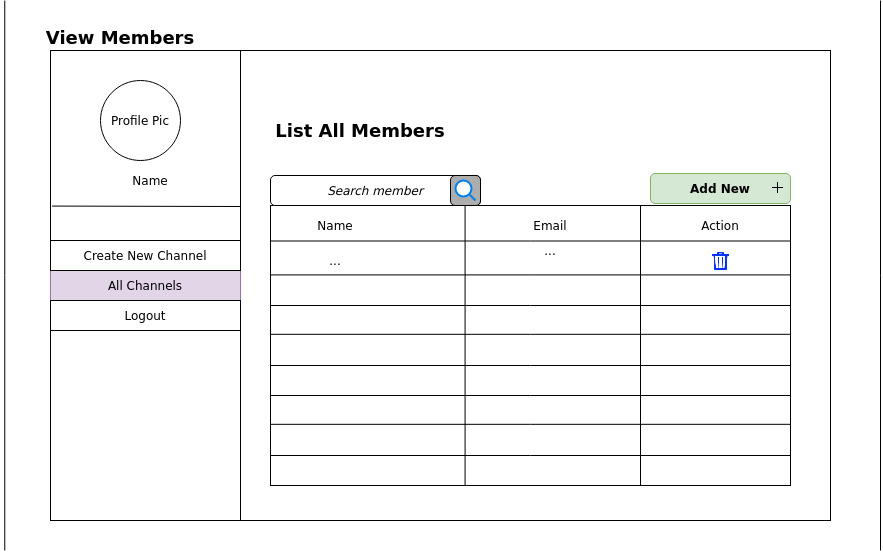


Figure 10 : Wireframe-3

## 

## 5.5.4 Add user into the channel

Apart from creating and viewing the channel, the user can add a student from the list of organisation’s user retrieved from the MS Graph API. In case of MS Graph Server down, the web application would show an error message.

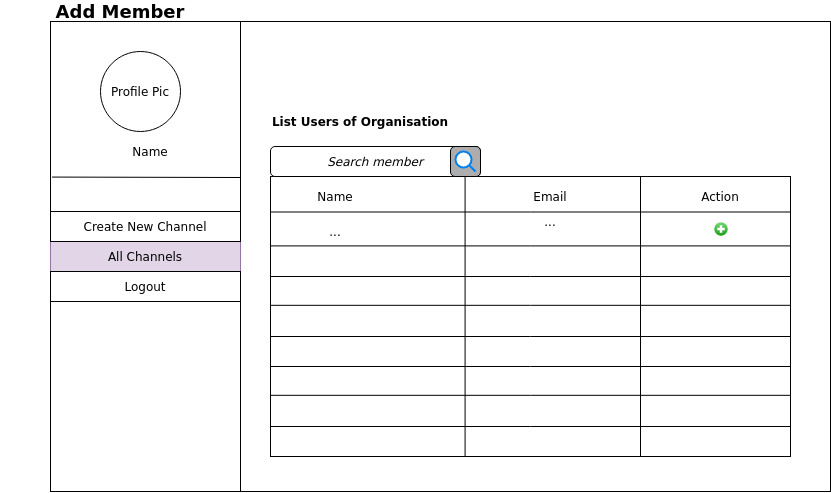


Figure 11 : Wireframe-4

# 

# **Chapter 6: Implementation**

# **6.1 Software Development Methodology**

The methodologies that help in the development process of a software development from the planning phase to the implementation and testing phase.

## 6.1.1 Waterfall

The waterfall is considered to be one of the most outdated and traditional approach used whenever the requirements of the project from the client side are crystal clear and there would be no chance of any new feature in the future. This allows a well-managed development process. It breaks down software development activities into linear sequential phases, It is to say that to switch to next phase of development there should be completed the previous phase beforehand.

## 6.1.2 Rapid Application Development

With Rapid system development, we develop a system with a low cost of development. Rapid Application Development allows coders to deal with rapidly changing requirements. Rapid Application Development includes planning, designing, constructing, and cutover.

**Disadvantages of Rapid Application Development:**

* It requires experienced developers and an extensive understanding of development.
* The approval is required after each construction of development phase

## 6.1.3 Prototype Development

A prototype is developed to validate the functional processes of the required web application.

**Disadvantages of Prototype Development:**

* Due to too many changes, and excessive client involvement causes the development process to go out of the flow.

## 6.1.4 Spiral Model

Spiral Model is a software development model based on risk-driven software development process. It is a combination of iterative model and waterfall model. Spiral Model helps to adopt software development elements of multiple process models based on unique risk patterns ensuring efficient development process.

**Disadvantages of Spiral Model Development:**

* Time management becomes hard because all the focus is provided to risk analysis, and number of loops (phases) are unknown.
* Often fails if there are no experienced professionals to make a proper risk analysis.

## 6.1.5 Agile Development

* Unlike waterfall, agile approach allows the project to be broken down into chunks of iteration and each iteration is a deliverable. In this approach changes can be easily entertained.
* There are so many advantages of Agile methodology than waterfall, Below is given some:
* Agile techniques eliminate the complete project failure chances. In Agile, we always work on products in incremental deliveries from the very first sprint, so that project failure chances are reduced.
* In agile project management, we are more focused on less re-work on projects because we found issues and changes in the early stages. On the contrary, waterfall is not an iterative approach; there are many possibilities of re-work after completion or wrong delivery of a major milestone or a feature.
* In agile, end-users often have opportunities to assess the delivery of work and make decisions and make changes throughout the software development life cycle. While waterfall doesn’t involve the end-user's suggestion in the projects.
* Stakeholders frequently check and demonstrate the project which allows for modifications to be done at a much faster pace. It is good for smaller teams to let them receive feedback fast and make it convenient for them to modify according to the needs and wants of the end-user.
* Testing in agile development is done at the sprint level to ensure that the project delivery is at an optimum state. It allows the project managers to do necessary modifications and it also allows the team to be aware of potential issues in advance. But in waterfall, there is no any option of iterative testing that makes projects vulnerable to failure.

In waterfall, at the end of the development process, the value delivery comes. If the budget and time exceeds from the agreed ones then there may be no money and time left to deliver the value according to the deal done with the client. Hence, this is the highly vulnerable in Waterfall project management to make changes in budget. In contrast, agile project management is more flexible. It allows to correct things timely.

Based on the said factors above, this project is developed using agile methodology since it is an iterative and increments based development model. Software operations are performed on a daily basis. It is a continuous requirement gathering process till the end of the development process. Users are free to ask any change at any point of the development cycle. Even late changes in requirements should be welcomed. Whereas, the waterfall model requires all the requirements to be gathered first then move to the next phase of development. As I plan to work in smaller deliveries and hope to increase scope of the project, waterfall and other methodologies does not fit for my project.

# **6.2 Work Distribution**

## 6.2.1 Gantt Chart

Whatever project related activities that we perform during the timeframe of our project are represented in the form of chart with respect to time and that is termed as Gantt chart. The left of the chart has the list of activities that are performed during the project development, while the top has the corresponding time in which the activity was performed. A bar is used to represent an activity. The start date, duration and end of the activity can be assumed with the help of the length, and position of the Gantt chart.

The Gantt chart can help us:

1. Identify different activities of the project
2. Know the beginning and ending of every particular activity
3. Identify the duration of the particular
4. Identify the duration of the entire project and also tells us the beginning and ending of the project
5. By identifying the timeframe in which an activity can overlap with the other

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Week  1 | Week 2 | Week 3 | Week  4-9 | Week 10 | Week 11-13 | Week 14-22 | Week 23-26 | Week 27 | Week 28 |
| Task 1 | Project Idea |  |  |  |  |  |  |  |  |  |
| Task 2 |  | Supervisor allocation |  |  |  |  |  |  |  |  |
| Task 3 |  |  | Fix Proposal |  |  |  |  |  |  |  |
| Task 4 |  |  |  | Requirements |  |  |  |  |  |  |
| Task 5 |  |  |  |  | Initial Report |  |  |  |  |  |
| Task 6 |  |  |  |  |  | Project plan & system designs |  |  |  |  |
| Task 7 |  |  |  |  |  |  | Complete  Development |  |  |  |
| Task 8 |  |  |  |  |  |  |  | Final  Write-up |  |  |
| Task 9 |  |  |  |  |  |  |  |  | Final Report |  |
| Task 10 |  |  |  |  |  |  |  |  |  | Viva |

Table 7 : Gantt Chart

# **6.3 Code Implementation**

## 6.3.1 Microsoft Authentication Library (MSAL)

The Microsoft Authentication Library (MSAL) is a library provided by Microsoft that enables software developers to acquire web tokens from the Microsoft identity platform. Tokens are provided to authenticate users and get secured access to web APIs. The web token is needed to access Microsoft Teams APIs. As we are using Angular for making APIs calls to Microsoft teams server, MSAL supports the angular platform to redirect to login screen so user can securely login from Microsoft portal and then access web token send to the application for further use.

### Code Snippets:

### 1. Import MSAL library module components to the application component

*Import {MsalService, MsalBroadcastService, MSAL\_GUARD\_CONFIG, and MsalGuardConfiguration} from '@azure/msal-angular';*

1. MsalService: is an authentication service that is responsible to redirect user’s login request to Microsoft server.
2. MsalBroadcastService: is a service to check whether user interaction is completed and user is successfully logged in to the application.
3. MsalGuardConfiguration: is a guard service to check whether user’s auth configuration is set or not. It also helps to request the Microsoft service and get the web token on redirect request.

### 2. Login Redirect Method *loginRedirect() {*

*if (this.msalGuardConfig.authRequest) {*

*this.authService.loginRedirect({ ...this.msalGuardConfig.authRequest } as RedirectRequest);*

*} else {*

*this.authService.loginRedirect();*

*}*

*}*

The above function is responsible to redirect the browser tab to Microsoft server so that user can login securely and application can get web token of the user.

### 3. To show all the joined channels

### In Microsoft teams, every channel is associated with a team. To get the all user joined channels. First we need to the team IDs of the joined teams of the user.

The below http request URL is responsible to retrieved all the joined teams of the user. Only authorization with access token is required to get all the joined teams of the user.

*GET* [*https://graph.microsoft.com/beta/me/joinedTeams*](https://graph.microsoft.com/beta/me/joinedTeams)

After retrieving the all the teams of user, we use the teams Ids to get the associated channels with it. Below is the http request URL to retrieve the user channels.

*GET* [*https://graph.microsoft.com/v1.0/teams/[teamID]/channels*](https://graph.microsoft.com/v1.0/teams/%5BteamID%5D/channels)*'*

We can retrieve channels of one team at a team. We have added a callback queue that to retrieve channels of team one by one and associated in to the channels array. In request URL, *teamID* will be replace by the actually id of individual team to get associated channels.

Below are the codes to retrieve joined teams and channels of the user. ‘*this.setTableData(this.tableData*);’ is used to display data of channels in the table.

**Get Teams – Code Snippet**

*getTeams() {*

*let endpoint = 'https://graph.microsoft.com/beta/me/joinedTeams';*

*this.http.get<any>(endpoint)*

*.subscribe(response => {*

*if (response.value) {*

*let teams = response.value*

*for (let x in teams) {*

*this.getChannels(teams[x].id, teams[x].displayName);*

*} } }); }*

**Get Channels – Code Snippet**

getChannels(teamID, teamName) {

let endpoint = 'https://graph.microsoft.com/v1.0/teams/' + teamID + '/channels';

this.http.get<any>(endpoint).subscribe(response => {

if (response.value) {

let channels = response.value;

for (let x in channels) {

if (channels[x].displayName == "General") continue;

let data = {} as ChannelData;

data.channelID = channels[x].id;

data.teamID = teamID;

data.teamName = teamName;

data.channelName = channels[x].displayName;

data.description = channels[x].description;

data.createDate = channels[x].createdDateTime;

this.tableData.push(data);

}

this.setTableData(this.tableData); } }); }

### 4. Create Channel

In Microsoft teams, every channel is associated with a team. So, before creating a channel we need to create a team that associate the channel within it. MS Teams API service provide a single URL request to create team and channel in single request call. Below is the http post request call to send request to Microsoft server for creating the team and channel.

*POST* [*https://graph.microsoft.com/v1.0/teams*](https://graph.microsoft.com/v1.0/teams)

The above request contains some body parameters with itself that are:

1. **displayName:** it is the display name for the team.
2. **description:** it is the description for the team.
3. **Channels []:** it is the array for the channels that we want to associate with that team. The channel array contains an array of object with following values:
   1. **displayName:** it is the display name for the channel.
   2. **description:** it is the description for the channel.

Below is the code snippet that is used to send request to Microsoft server for creating a team and channel. *FormGroup* is a group that contains the input values for channel name and description, given by the user. *FormGroup.value is an object to get the values of the form group.*

**Create Channel – Code Snippet**

*createChannel() {*

*if (this.formGroup.valid) {*

*this.spin = true;*

*let reqObj = {} as ChannelReq;*

*reqObj.displayName = this.formGroup.value.displayName;*

*reqObj.description = this.formGroup.value.description;*

*reqObj["template@odata.bind"]="https://graph.microsoft.com/v1.0/teamsTemplates('standard')";*

*reqObj.channels = [];*

*reqObj.channels.push(this.formGroup.value)*

*let endpoint = 'https://graph.microsoft.com/v1.0/teams';*

*console.log(reqObj);*

*this.http.post<any>(endpoint, reqObj).subscribe(response => {*

*alert("Request Submitted to MS Teams. Please verify in 'My Channels' after few minutes.");*

*this.resetFields();*

*},*

*error => {*

*alert(error.message);*

*}); } }*

### 5. List Organisation users

### If we want to make easy for teachers to add students in channel from a list then the list of organization’s users will be needed. In Microsoft Graph APIs, there is an API which returns users data of the organizations if admin consent is provided to the application. Below is the http get request call to retrieve list of users of an organisation.

*GET* [*https://graph.microsoft.com/v1.0/users*](https://graph.microsoft.com/v1.0/users)

**List Organization Users – Code Snippet**

*getUsers() {*

*let endpoint = 'https://graph.microsoft.com/v1.0/users';*

*this.http.get<any>(endpoint)*

*.subscribe(response => {*

*if (response.value) {*

*let users = response.value;*

*for (let x in users) {*

*if (users[x].mail == null) continue;*

*let data = {} as UserData;*

*data.serialNo = x + 1;*

*data.userID = users[x].id;*

*data.mail = users[x].mail;*

*data.displayName = users[x].displayName;*

*this.tableData.push(data);*

*}*

*this.setTableData(this.tableData);*

*}*

*});*

*}*

### 6. Add user to channel

The channel owner will add the user from the list of users of the organisation. To add member into the channel, there is an Microsoft teams API that is responsible to add user to the channel. Below is the http post request to add channel member.

*POST* [*https://graph.microsoft.com/v1.0/teams/[teamId]/channels/[channelID]/members*](https://graph.microsoft.com/v1.0/teams/%5BteamId%5D/channels/%5BchannelID%5D/members)

In request URL, *teamID* will be replace by the actual id of the team and channelId will be replaced by the actual Id of channel to get members of that channel. The request URL also contains some body parameters with itself that are:

1. **data.type:** it is a type of user that we are going to add in the channel. In our case, it will be conversation member
2. **user:** it is the UserId of a user that channel owner wants to add as a member of channel

**Add Channel Member – Code Snippet**

*addUserToChannel(userID) {*

*let channelID = localStorage.getItem("channelID");*

*let teamID = localStorage.getItem("teamID");*

*let endpoint = 'https://graph.microsoft.com/v1.0/teams/'*

*endpoint += teamID;*

*endpoint += '/members';*

*userID = "https://graph.microsoft.com/v1.0/users/"+userID;*

*let postObj = {*

*"@odata.type": "#microsoft.graph.aadUserConversationMember",*

*roles: [],*

*"user@odata.bind": userID*

*}*

*this.http.post<any>(endpoint, postObj)*

*.subscribe(response => {*

*if (response) {*

*alert("Request is Submitted");*

*this.router.navigate(['channels/members'])*

*}*

*});*

*}*

### 7. List channel members

In Microsoft teams, owner of channel can add member users to the channel to collaborate. Microsoft Graph APIs provide an API to retrieve the list of channel members by providing the teamId and channelId. Below is the http get request call to retrieve list of members of a channel.

*Get* [*https://graph.microsoft.com/v1.0/teams/[teamId]/channels/[channelID]/members*](https://graph.microsoft.com/v1.0/teams/%5BteamId%5D/channels/%5BchannelID%5D/members)

In request URL, *teamID* will be replace by the actual id of the team and channelId will be replaced by the actual Id of channel to get members of that channel.

**List Channel Members – Code Snippet**

*getMembers() {*

*let channelID = localStorage.getItem("channelID");*

*let teamID = localStorage.getItem("teamID");*

*let endpoint = 'https://graph.microsoft.com/v1.0/teams/'*

*endpoint += teamID;*

*endpoint += '/channels/';*

*endpoint += channelID;*

*endpoint += '/members';*

*this.http.get<any>(endpoint)*

*.subscribe(response => {*

*this.tableData = [];*

*if (response.value) {*

*let users = response.value;*

*for (let x in users) {*

*let data = {} as UserData;*

*data.serialNo = x+1;*

*data.memberID = users[x].id;*

*data.userID = users[x].userId*

*data.role = users[x].roles;*

*if (data.role.length == 0) {*

*data.role = "member";*

*}*

*data.mail = users[x].email;*

*data.displayName = users[x].displayName;*

*this.tableData.push(data);*

*}*

*this.setTableData(this.tableData);*

*}*

*});*

*}*

### 8. Remove channel member

The channel owner can remove the user from the list of channel members. To remove member from the channel, there is an Microsoft teams API that is responsible to remove member from the channel. Below is the http delete request to add channel member.

*Get* [*https://graph.microsoft.com/v1.0/teams/[teamId]/channels/[channelID]/members/[memberId*](https://graph.microsoft.com/v1.0/teams/%5BteamId%5D/channels/%5BchannelID%5D/members/%5BmemberId)*]*

In request URL, teamId will be replaced by the actual team Id, channelId will be replaced by the actual channel Id and memberId will be replaced by the members id of a channel members (a unique Id associates with every member when the join the channel).

**Remove Channel Member – Code Snippet**

*deleteMember(memberID) {*

*let channelID = localStorage.getItem("channelID");*

*let teamID = localStorage.getItem("teamID");*

*let endpoint = 'https://graph.microsoft.com/v1.0/teams/'*

*endpoint += teamID;*

*endpoint += '/members/'*

*endpoint += memberID;*

*this.http.delete<any>(endpoint)*

*.subscribe(response => {*

*alert("Member removed from channel!")*

*this.getMembers();*

*});*

*}*

# **Chapter 7: Testing and Evaluation**

# **7.1 Software Testing**

Software Testing is a way to evaluate whether the system software matches the expectation of requirements or not. Testing of system components can be done manually or using the automate tool for the testing. We have used both ways to test our system.

We are performed following testing to check the system according to the expectations:

1. **API Testing:** Testing is performed to test the Microsoft team’s web services.
2. **Angular Unit Testing:** Testing is performed to test the Angular (frontend) components.
3. **Usability Testing:** Testingis performed to evaluate the simplicity of the system.

**7.1.1 API Testing**

API testing is a type of testing is which tester performs the testing of web services. There are several web services of MS teams that we are using in our system. We have performed API testing using Postman tool – Postman is a tool to test the web services.

### 1. Create Team & Channel Web service

To utilize this web service, we need a web token that identifies the user and its constraints. Web token can be obtained after login with Microsoft account by using its login credentials. We added the web token in authorization tab. Also, we added the required body parameters to make the API request. After that, we submitted the request to check the results. Below are the screenshots of the API call.

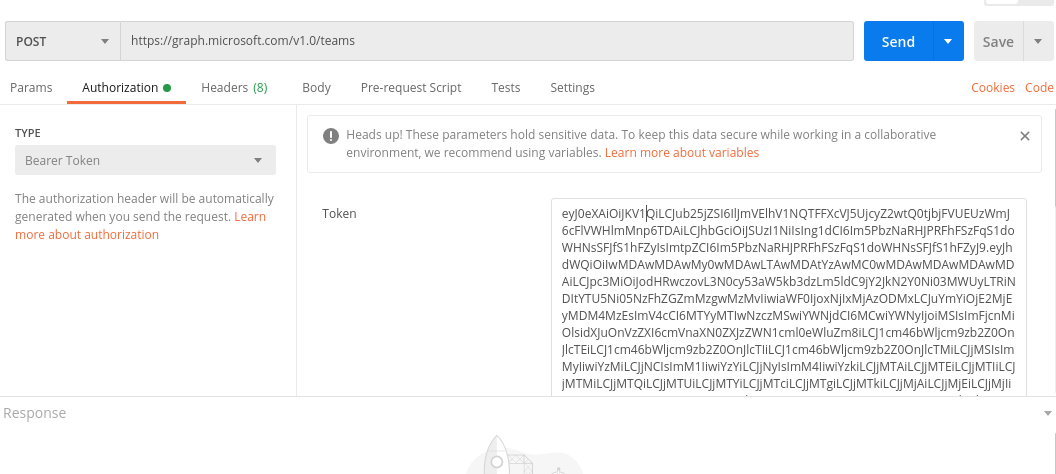


Figure 12 : Create Team

### 

Figure 13 : Create Team-2

### 2. Get Teams & Channels Web services

To obtain the user joined channel, first we need the teams in which the individual channel is associated. The teamIds are required to obtain the joined channels of the users.

To obtain the joined teams of the user, we only need the web token of the user.

To obtain the joined channels of the user, we need the web token of the user and the teamIds of the teams that user joined. Only channels of a team can be retrieved from the single request. To retrieve all the user channels, individual request to get channels should made.

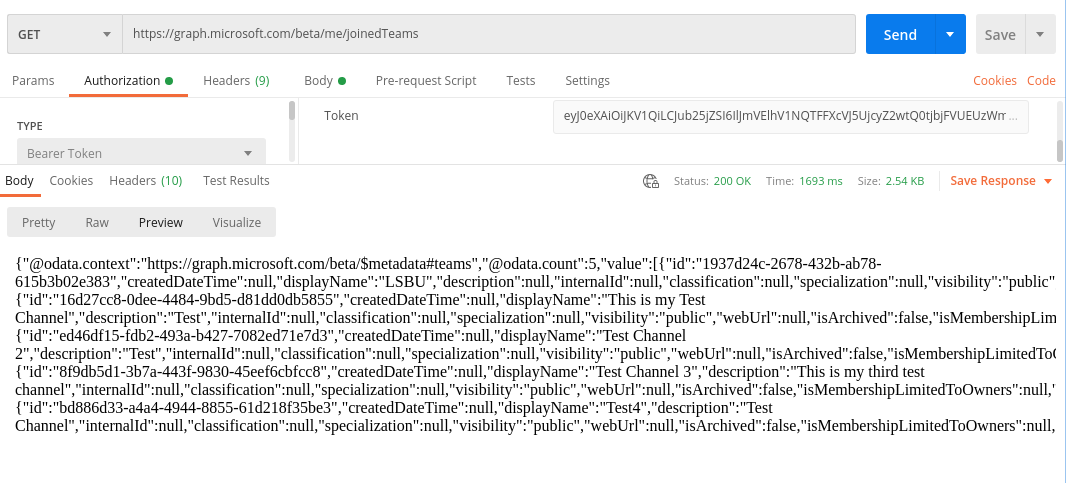


Figure 14 : Get Team-1

### 

Figure 15 : Get Team-2

### 3. Get Users of Organisation

We need the list of users of the organisation. The list will be displayed to user to add the member into the channel. User can only add member to the channel by using UserId of the member. Therefore, list of users is required to add member into the channel.

### 

Figure 16 : Get User

### 4. Add Member to Channel

To utilize this web service, we need a web token that identifies the user and its constraints. Also, we added the required body parameters to make the API request. The teamId and UserId, id of the user that we want to add into the channel, are required for this web service.

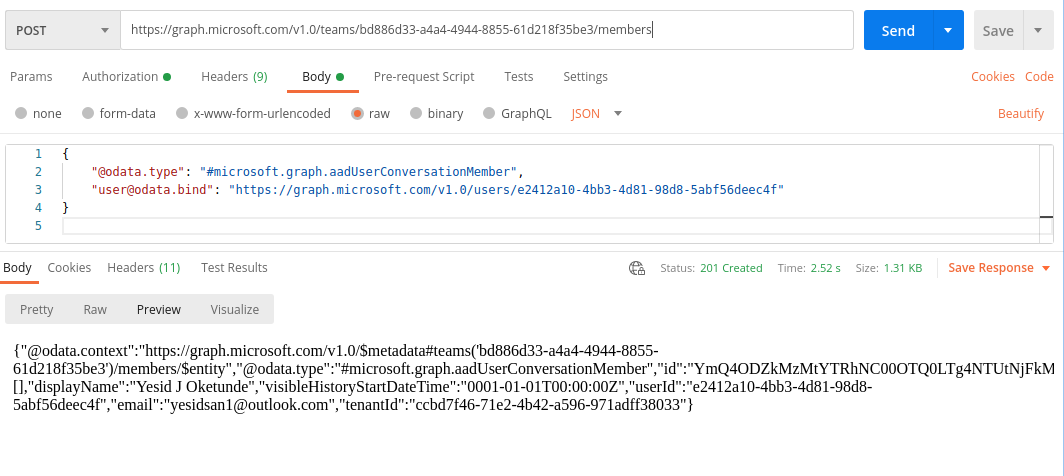


Figure 17 : Add Member

### 5. List Members of Channel

There is web service to list all the members that has joined the particular team. To utilize this web service, we need web token of user and the teamId of that team we want see the members.

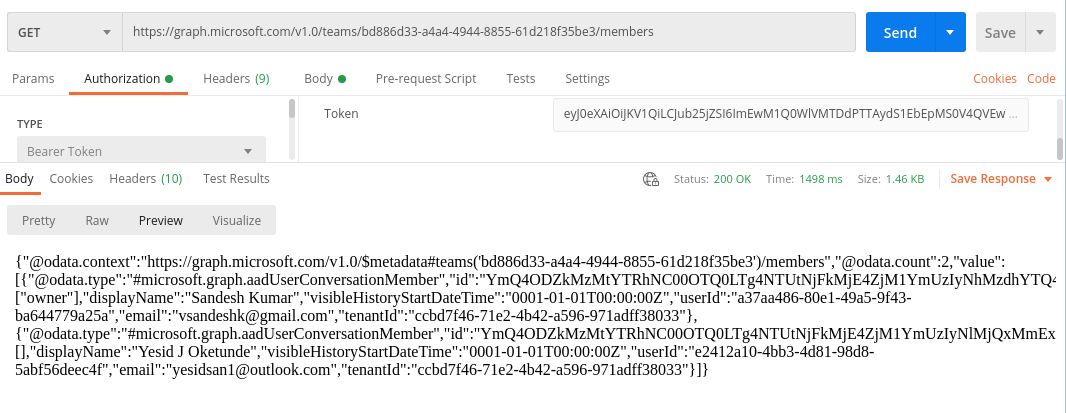


Figure 18 : Add Member-2

**7.1.2 Angular Unit Testing**

Unit testing is a type of testing in which we write test cases for individual component/unit of the system. Our application is divided in the several components such as login, list-channels, create-channels, list-users, add-user-into-channel, and list-channel-members. We have used the automated software testing tool namely ‘Karma’. Karma is a testing tool to perform unit testing of JavaScript code. Luckily, Angular contains karma as a default unit testing tool itself. We only have to write test cases to perform the testing using Karma. After executing the test cases, the browser opens and shows the success and failure of test cases.

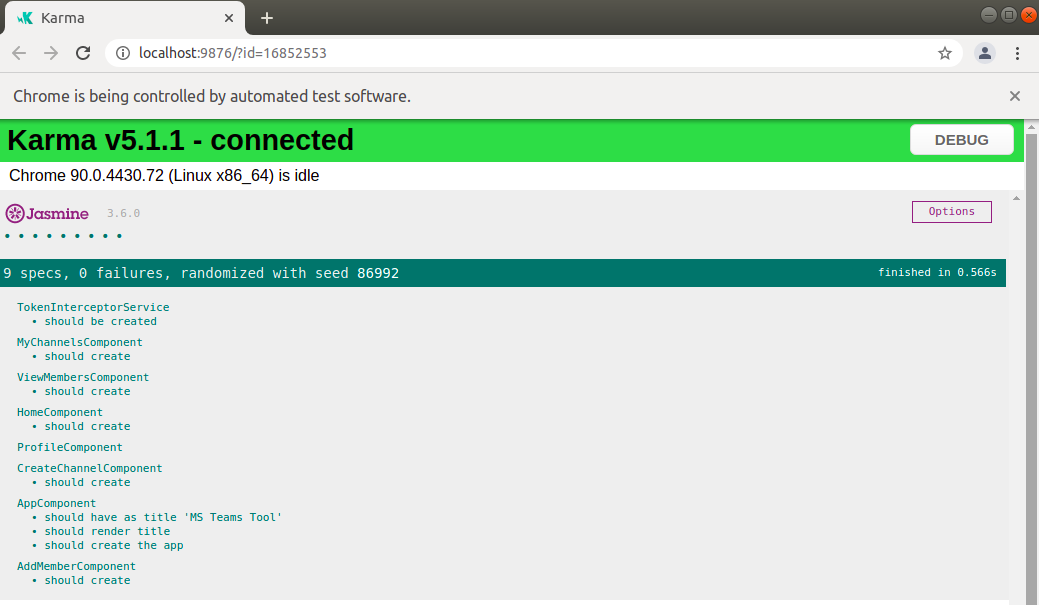


Figure 19 : Karma Testing

**7.1.3 Usability Testing**

Usability Testing is a type of testing that used to evaluate the simplicity of the system. The main objective of this system was to make a user-friendly system that also automates the most of work user by using few clicks. . In the system:

1. User can easily create the channel and add members into the channel.
2. List of organization users will be displayed in a table.
3. The data-table with sort and filter search is implemented that ease user to search the data by typing any letters related to data in the search box.
4. Channel owner user has to select the user to add into his channel. With only single click, owner user can add another user into the channel.

Usability testing is performed manually by asking the feedback of many MS Teams users to make improvements over user design.

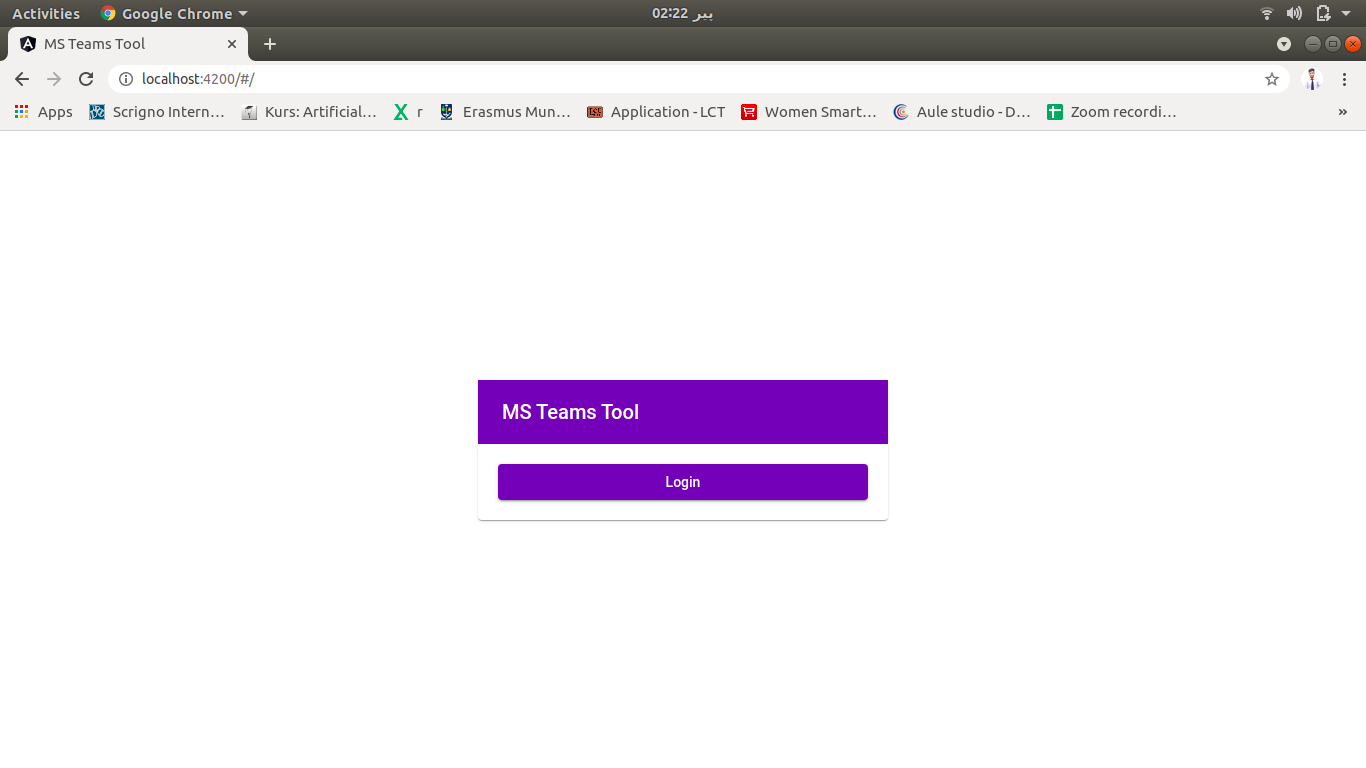
# **7.2 Final Results**

This section contains all the screenshot of the website screens evaluate the main functionality of the system.

**7.2.1 Login with MS Teams Account**

It was in the functionality that user can login with the MS teams account of the organization. To login and use the further functionality of the system, user has to open the browser and enter the particular URL to access the web system.

Figure 20 : Login



After the reaching at the login page of website, user has to click the ‘login’ button to continue using the web app. The login app will redirect to MS Teams API login screen to login using user credentials.

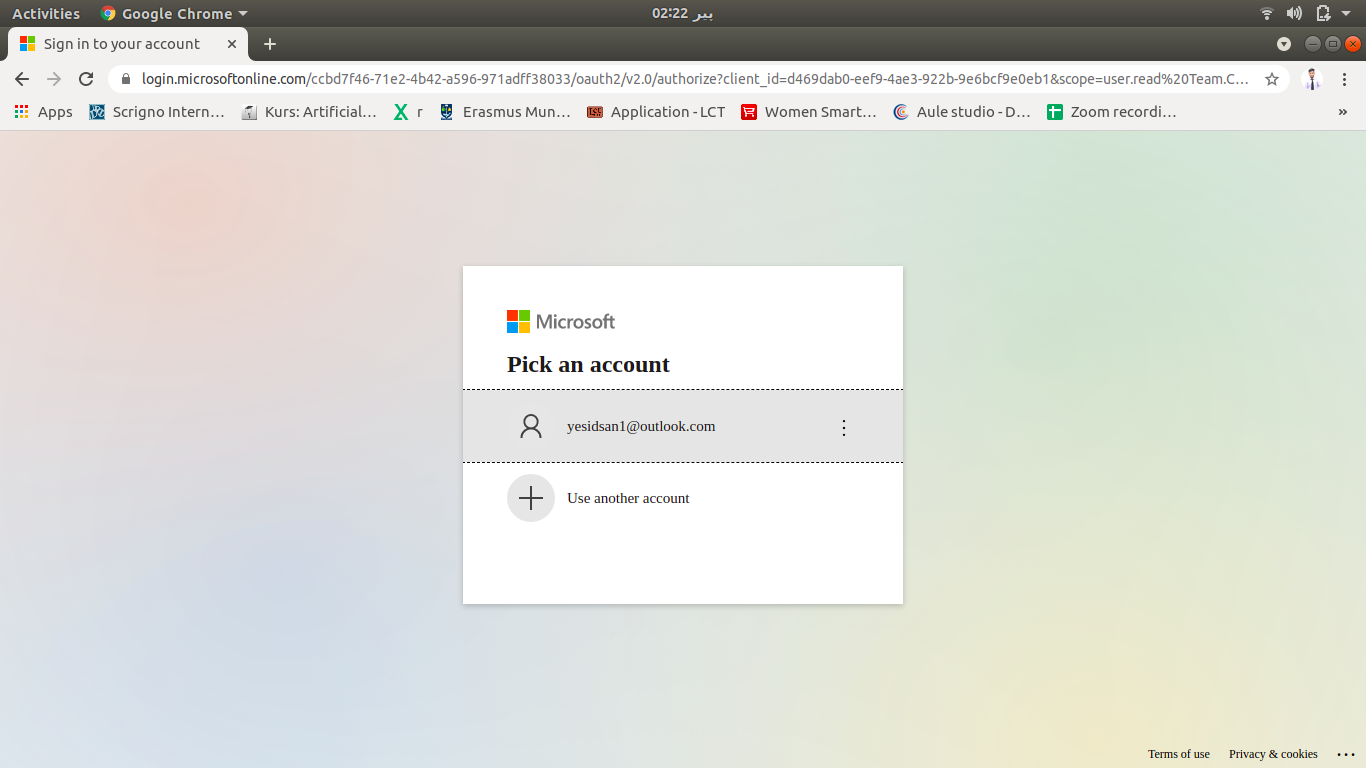


Figure 21 : Login-2

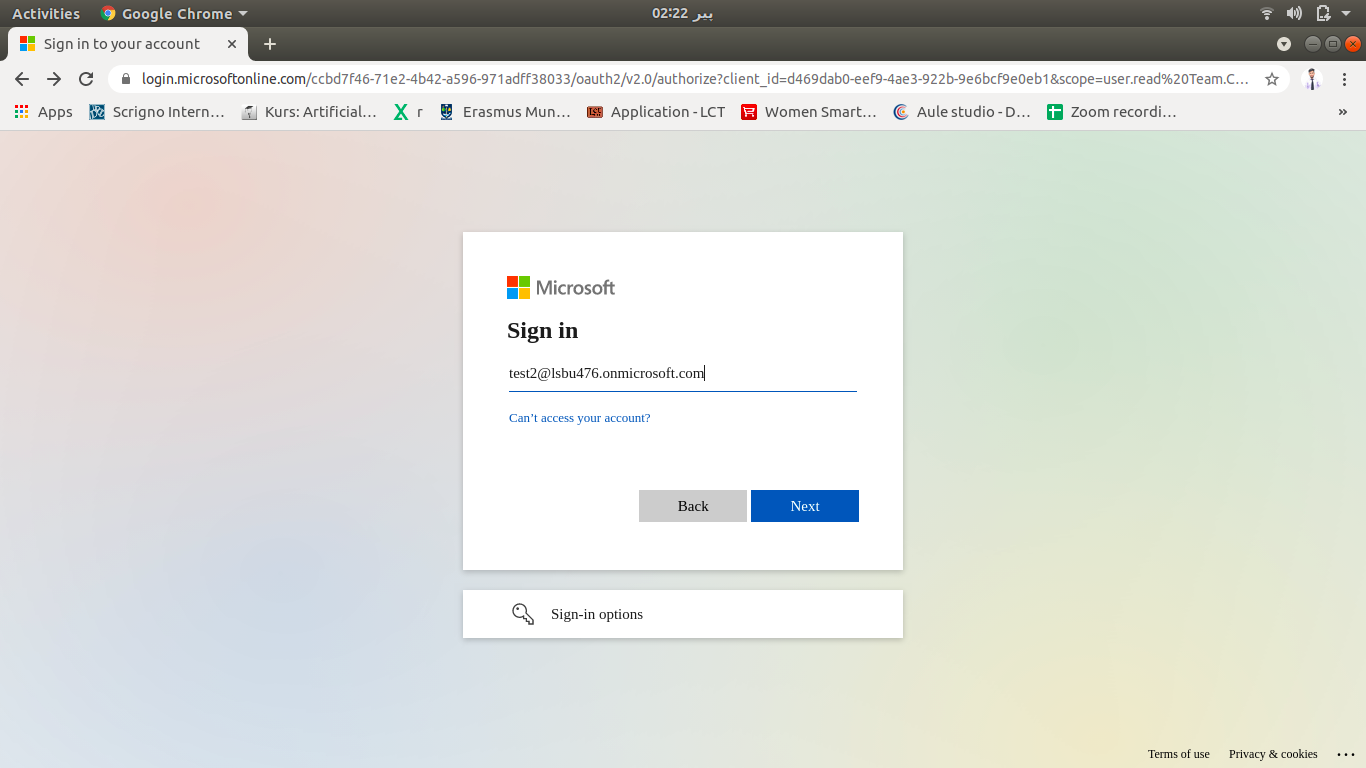


Figure 22 : Login-3

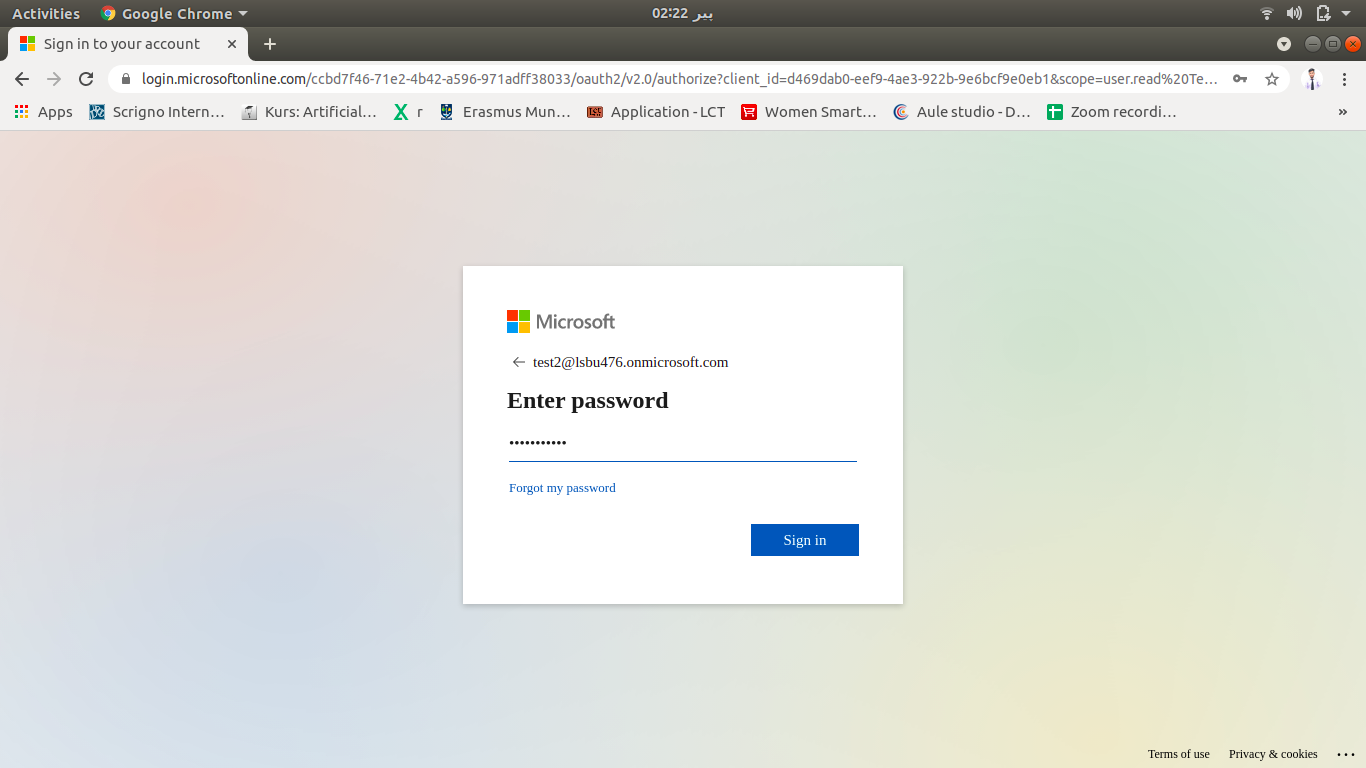
Once user entered the login credentials correctly, the application will be redirect to the dashboard of the web-app.

Figure 23 : Login Enter PAssword

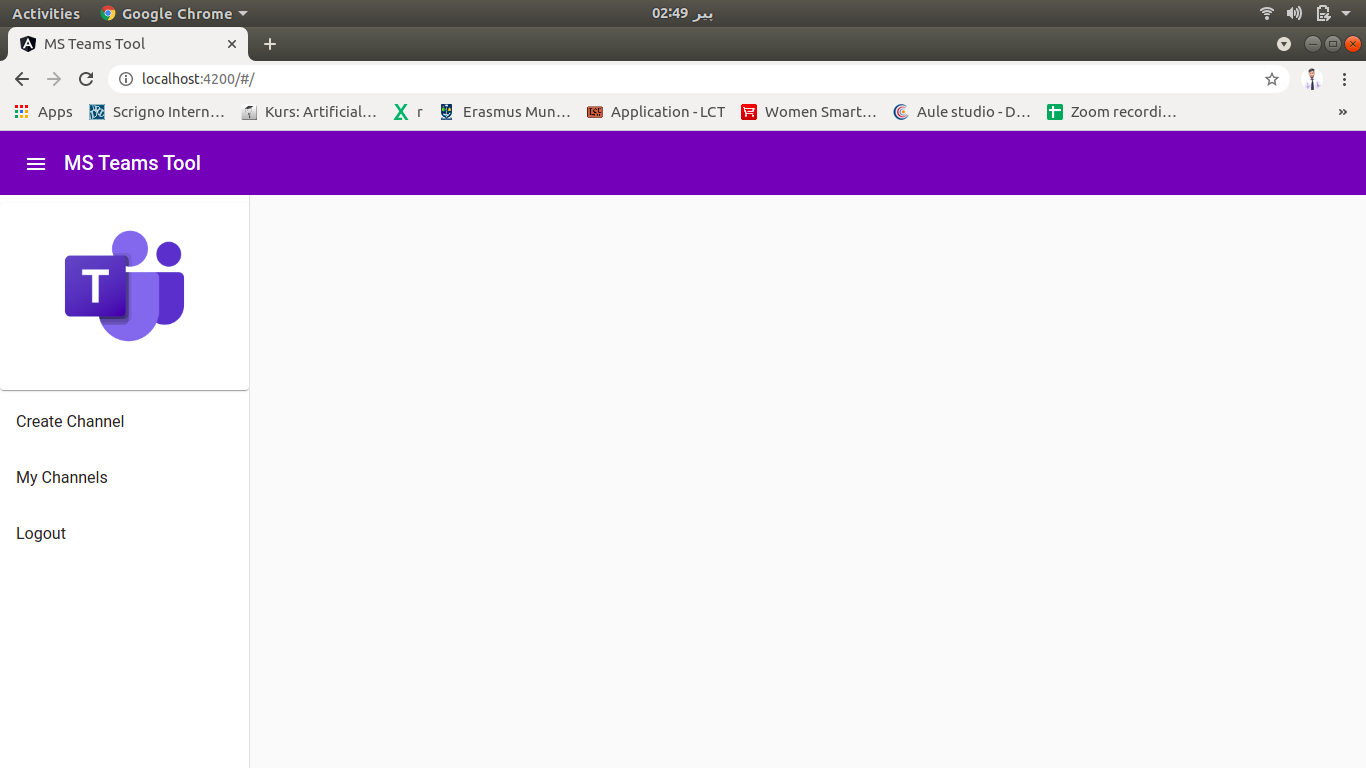


Figure 24 : Dashboard

1. **Create Channel:** It will redirect to create channel screen.
2. **My Channels:** It will redirect to the screen where user can see its joined channels.
3. **Logout:** It will logout the user account.

**7.2.2 Create Channel**

To create the channel, user has to select the “create channel” from the sidenav of the dashboard. After clicking, a screen with form will be open. User has to enter the regarding details and click on “Submit”. By clicking the submit button, the request to create channel will be send to MS Teams server using its web service. And once request completed, the respond will be shown as alert.

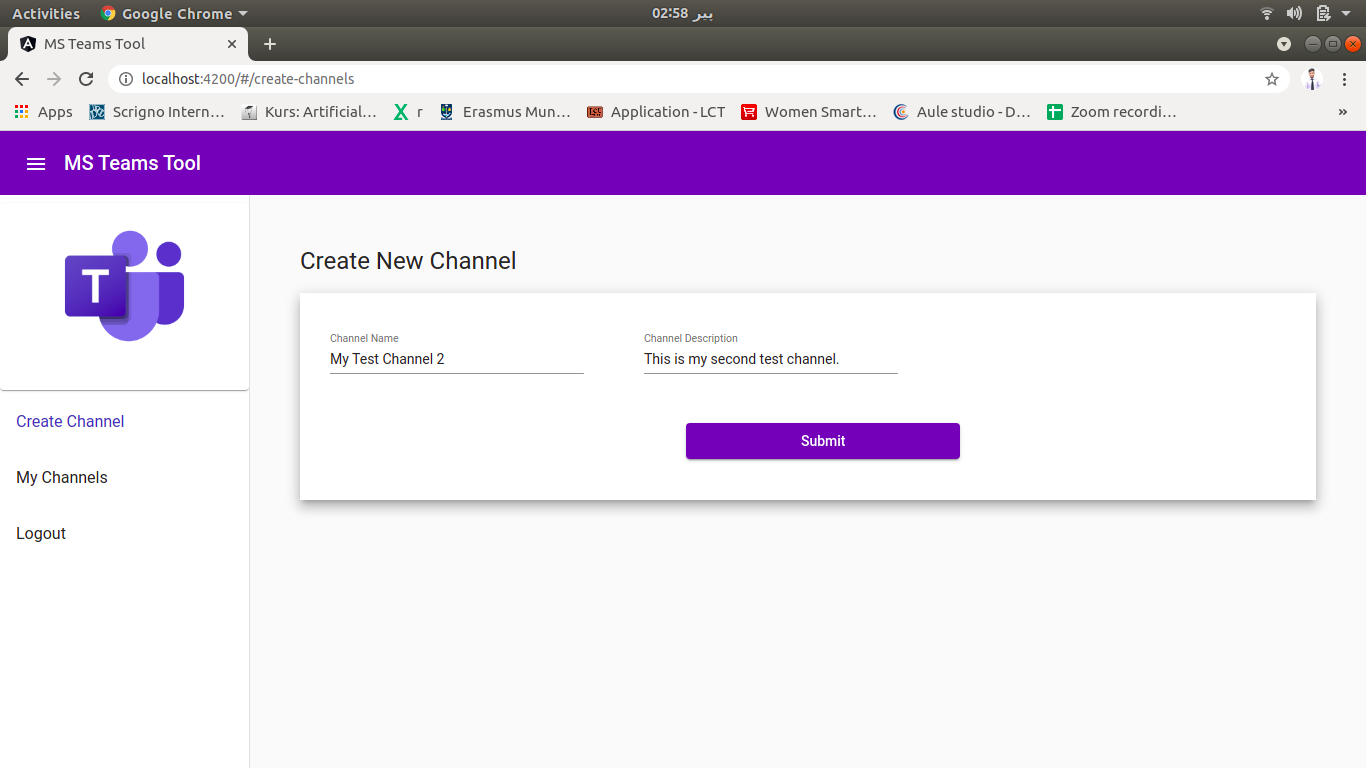


Figure 25 : Create Channel

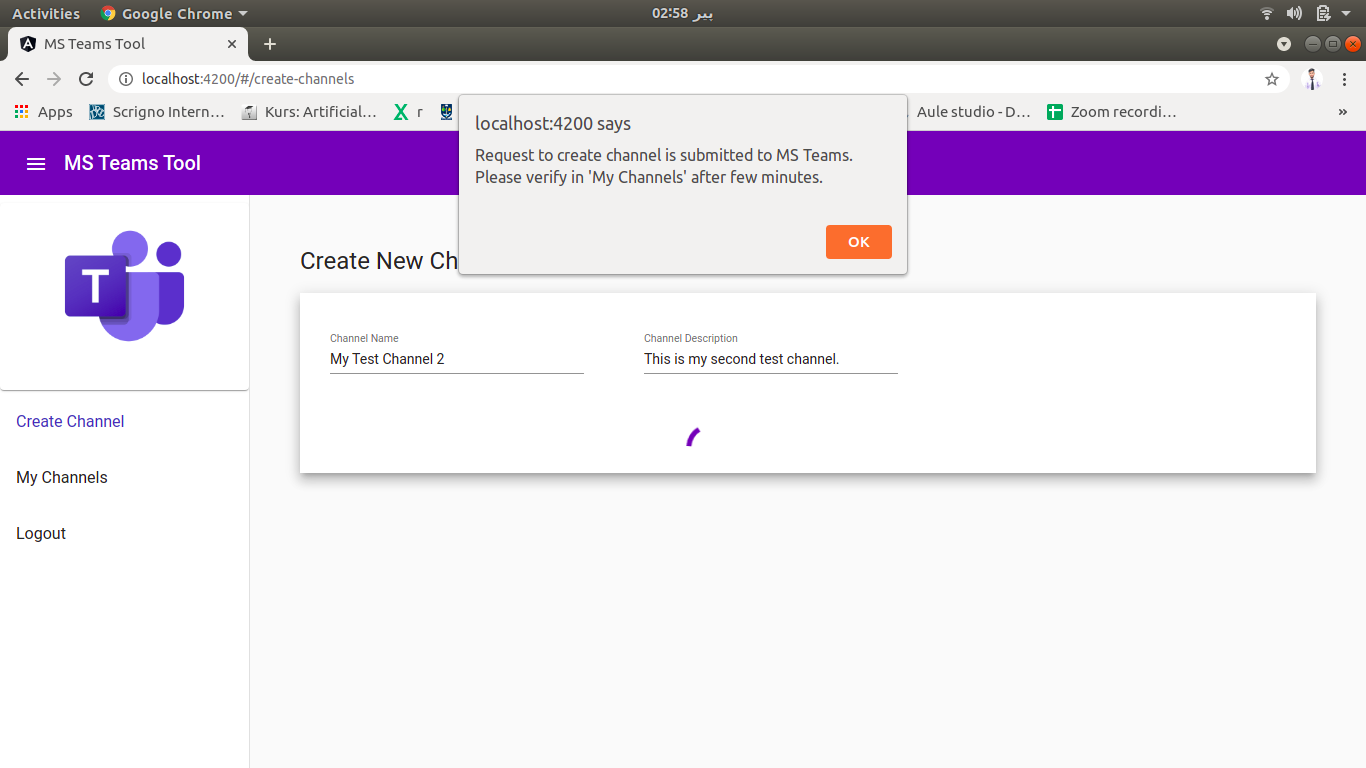


Figure 26 : Channel Created

**7.2.3 My Channel**

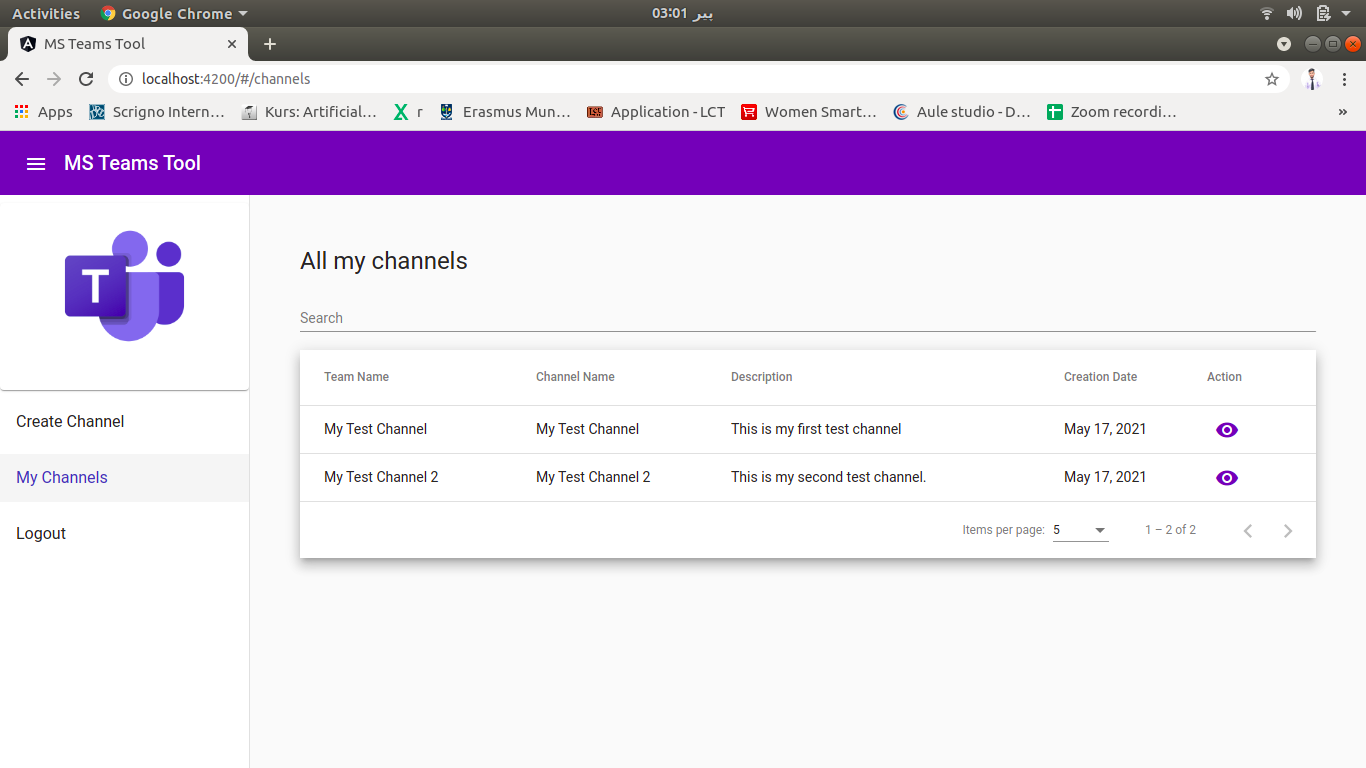
To show the list of channels, user will click the ‘my channels’ from the sidenav. After clicking this, the all the user joined channel will be displayed in the table format. User can search the channel or navigate using the pagination too. The eye icon in each row is indicates to view the members of channel. By clicking this icon, the list of channel members will be displayed.

Figure 27 : My Channel

## 7.2.4 List Channel Members

To view the list of channel members, user will click the eye icon from the channel list. List of channel members will be displayed in the table. User can search for member from the search box or user can navigate from the table list by changing the pages of pagination.

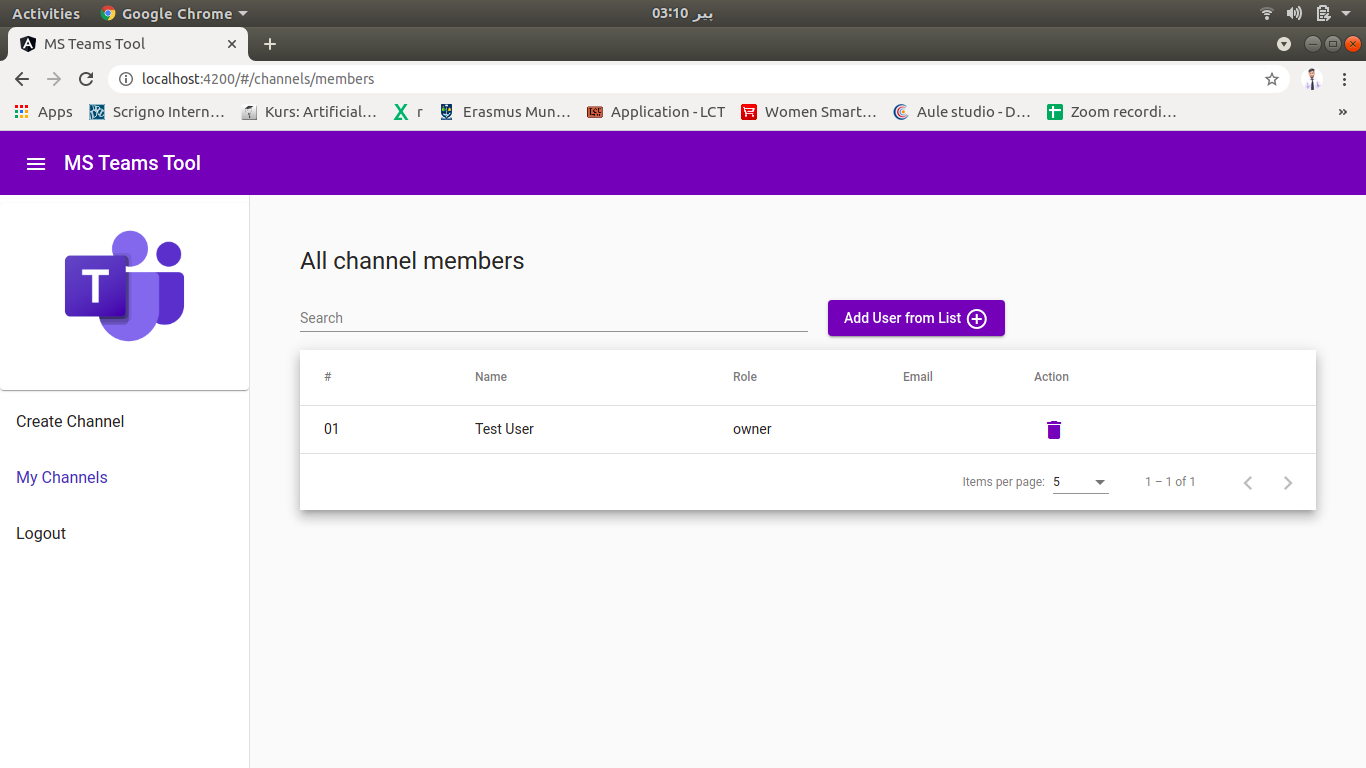


Figure 28 : Channel members

To add new member in the channel, user will click the ‘Add user....’ button. It will redirect to list of organization users to add the user into the channel.

## 7.2.5 List Organization Users

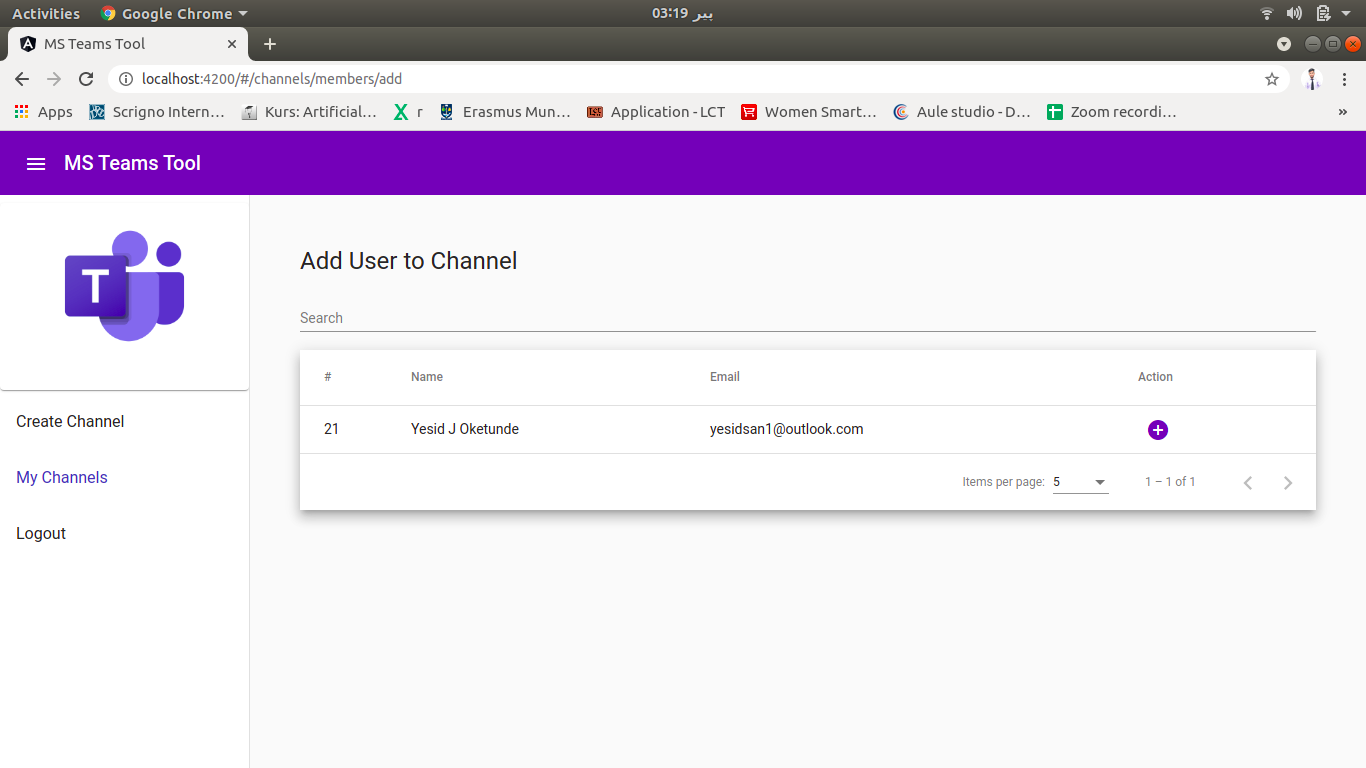
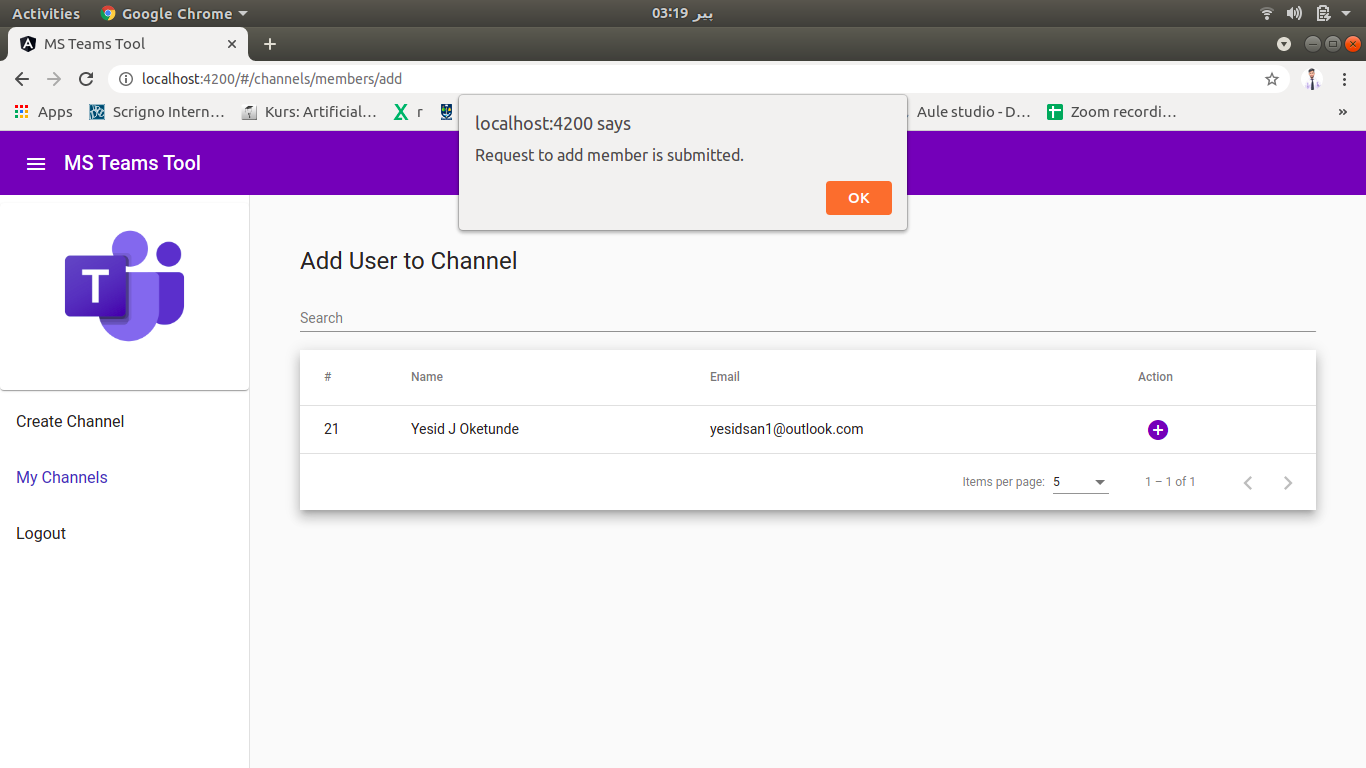
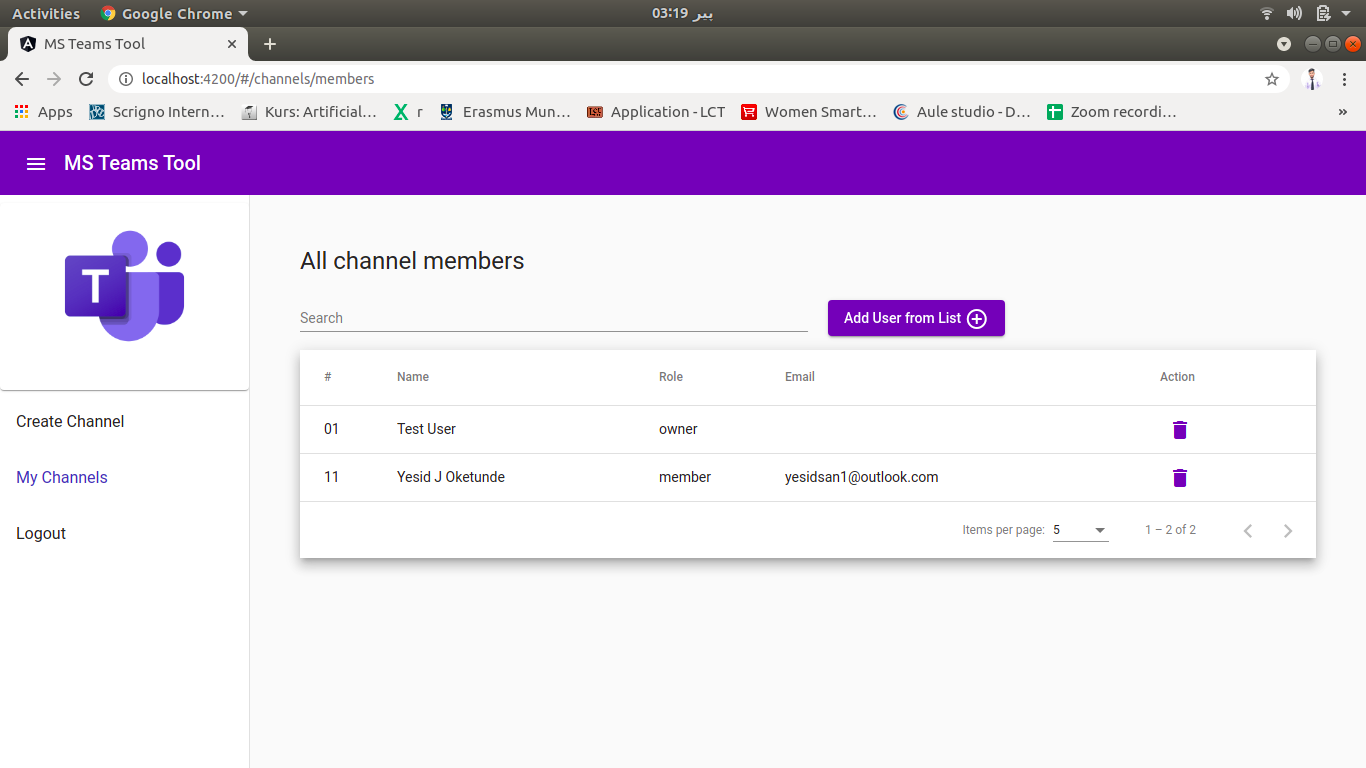
By clicking the ‘add user....’ button from the channel members screen, the app page will be redirect to view the list of organization users to add new user into the channel. The list of user will be displayed in tabular form with search box and pagination inside. Owner User can search another user by typing something related into the search box. Also, owner user can navigate by changing pages to see unseen users in the list.

Figure 29 : Add User

## 7.2.6 Add User into Channel

Owner user can add new user into the channel by clicking the ‘add/+’ button in the individual row of the individuals. By clicking this button, new user will be added into the channel.

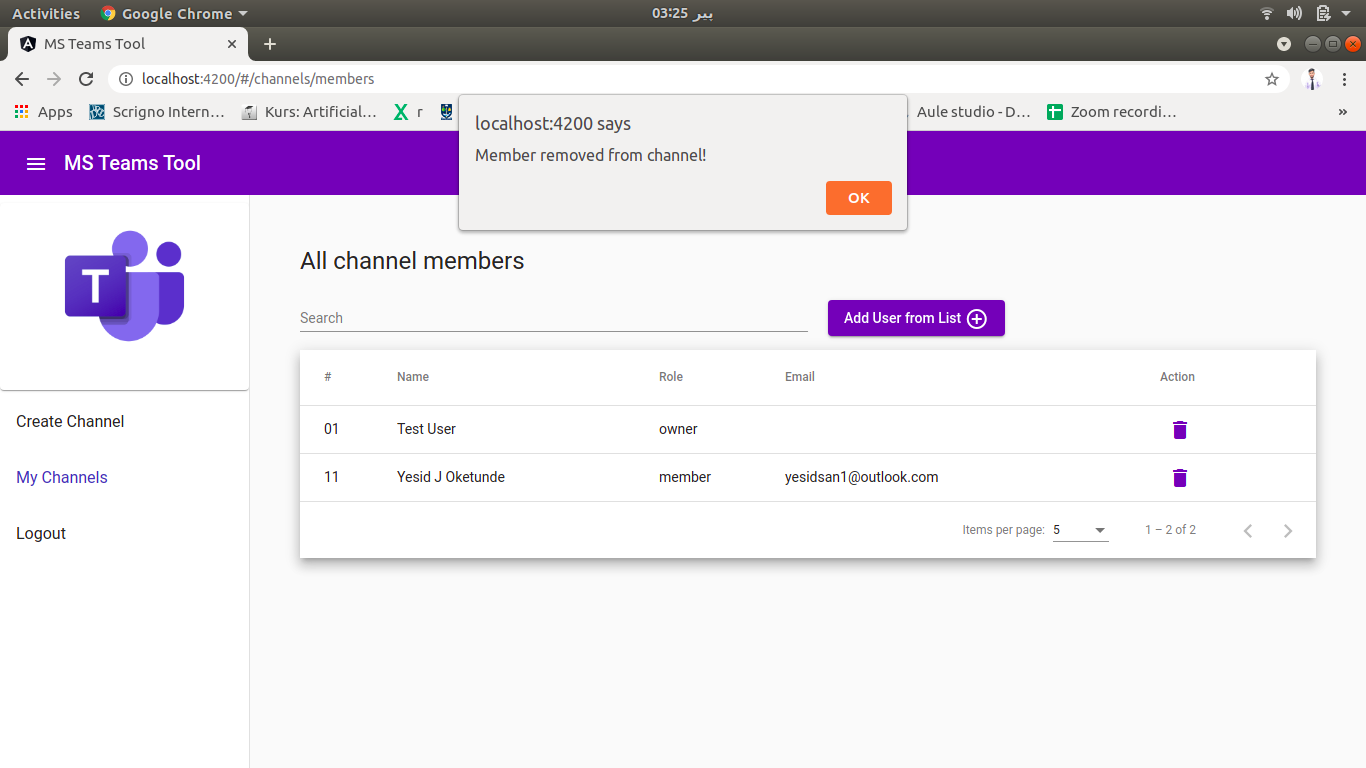




**7.2.7 Delete member from Channel**

Figure 30 : Request Submitted

To delete member from the channel, owner user will click the delete icon in the ‘list channel members’ webpage. By clicking this icon, the request to remove member from the channel will be send to MS Teams.



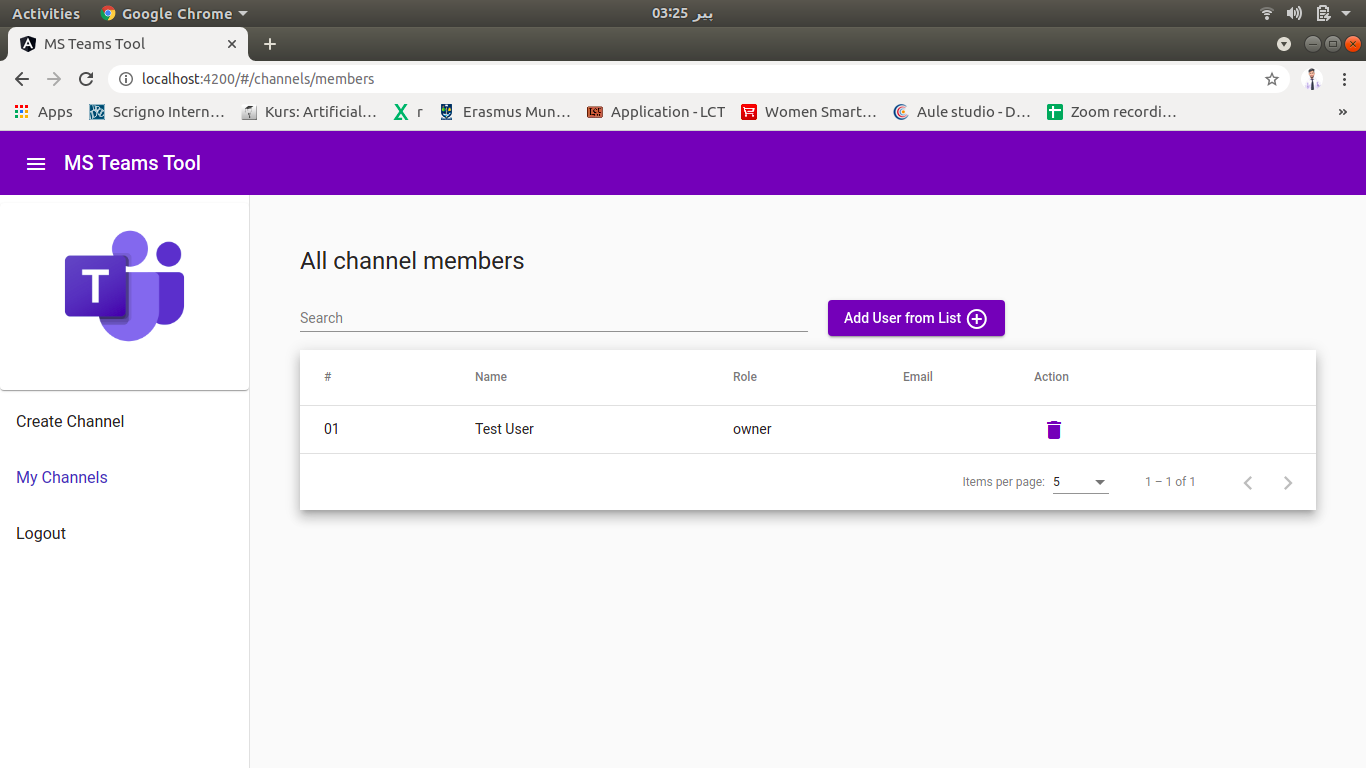
**7.2.8 Logout**

Figure 31 : Delete Members

To logout from the application, user has to click the ‘logout’ button form the sidenav. It will redirect to MS Teams APIs to logout the account and remove the access tokens from the cache of browser.

User has to select which account he has to logout (if multiple account from same browser has been loggedIn).

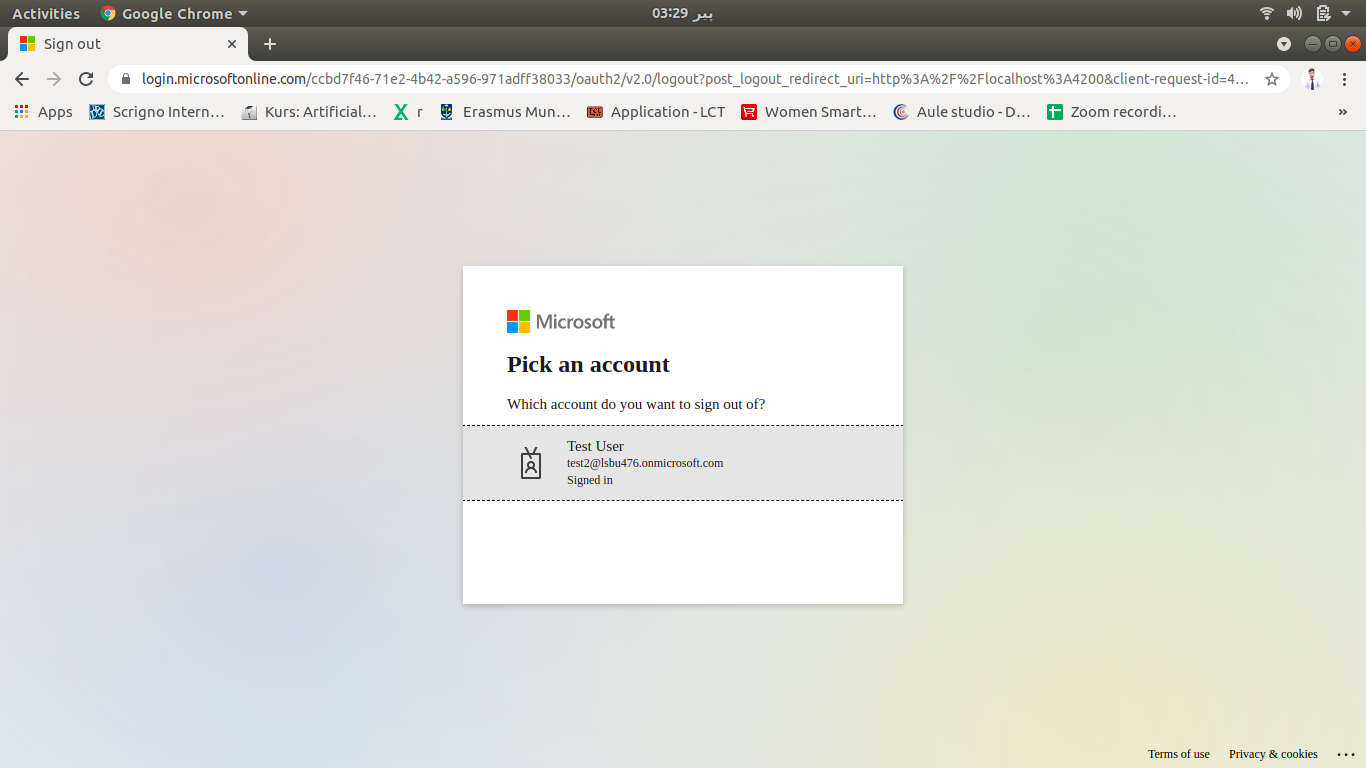


Figure 32 :Logout

By selecting the respective account, the particular account will be logged-out and the webpage will be redirected to the login webpage.

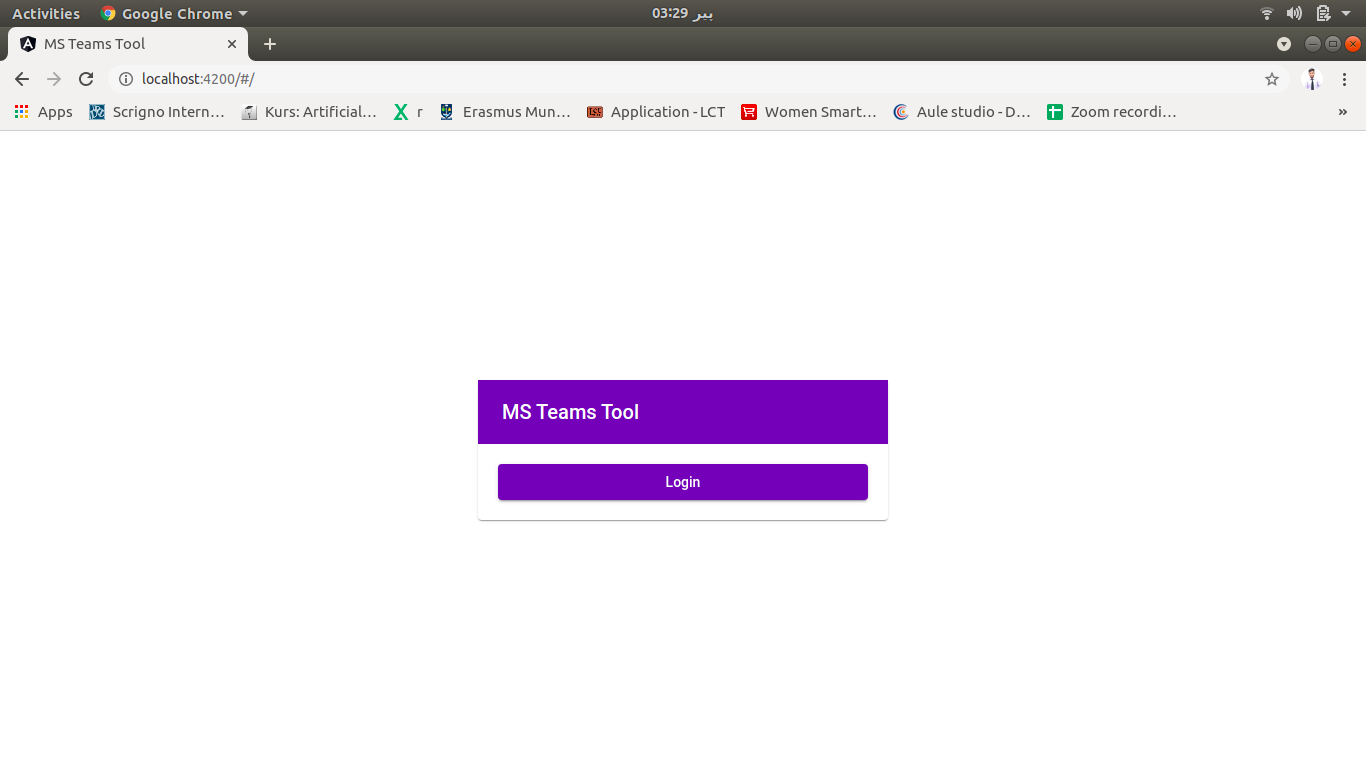


Figure 33 : Select Account

# **Chapter 8: Conclusion and Reflection**

# **8.1 Future Plan**

Though the system with the basic requirements are completely developed but it is not the end here. There are many things that has to done. We have developed our minimal software system, now it has to deploy on some cloud server and performance, load & stress testing will be done after deployment. Cloud server is a recommended to deploy the web application because the cloud solution offers flexibility, scalability & recoverability.

Moreover, software system will be extend in future. It can be extended with more functionality like resource sharing and start meeting. Also, the invitation will can be implemented in future so that user could join the channel using the invitation link of the channel.

Furthermore, the system can be extended to the mobile application too. As many people are moving towards the mobile application. Currently, we made a responsible web application that can run smoothly using mobile browser but in future the separate mobile application can also be developed.

# **8.2 Conclusion**

E-learning has always been important throughout the life but we have realized it now during the outbreak of Novel coronavirus where work was required to be done remotely. The corporate meetings, classes and other seminars were on hold and later there was a revolutionary change each and every organization switched to online mode from physical system. Due to closure of each public gathering sector such as school, offices. There was a big problem for everyone to keep the life going with their work. Academics was being interrupted and there was need for the platform which can be used by each and every organization with ease. For this challenge we have various platforms as a solution like: Zoom Meetings, WebEx, Google Classrooms, but after all they leave thee efficiency issue and along with that these tools are complex too operate and some of them are not even user friendly which causes some technical problem while addressing the real problem so that there should be such a platform which should be user friendly and easy to use reducing overall complexity.

The project mainly focus to eliminate the tiresome process for online learning and being free of complex online e-learning software. The main focus of our project is on reducing the complexity and providing a user friendly platform.

Novel Coronavirus had severe effect on the educational institutes but now they can take advantage from online learning. This time has revolutionized the global system of learning remotely such as students can easily share anything regarding their academics. Or the other organization can also be benefited from this too.

Although MS Teams is complex and is less user friendly. So therefore it has many competitors in the market which are more user friendly and less complex such as Zoom. We made up a project that has graphical user interface unlike the PowerShell command line which can be used only by commands making the usage of system more complex.

The technical review of this web application projects covers the entire architecture of this web application which includes that how our application is interacting with client and server side. We have made our web application using frontend and backend technologies where backend is totally responsible for functions of database, server computations, logic, backlogging processing syntax and operating data. Restricting user to use the backend. The successful communication between backend and frontend consist some of operations we have learnt so far. After the maximum research we find out Microsoft graph provides the API which was important for the web application to be formed. For the backend we have learnt the Node.js language for server side operations. We have used Node.js because it is open source, cross platform and it is browser independent to execute. After doing research and reviewing other various languages we chose and learnt thee Angular to code our frontend because it was more suitable than other to us.

As we collected the functional and non-functional requirements, we priorities the functional requirements using Moscow method and further specified them. After that we made them by using use case and UML diagrams then further elaborated them. Before implementing the code, we designed the software that how should we implement the software more efficiently. We made the class diagram, sequence diagram and activity diagram which purely supports the implementation.

For implementing the web application, we used the agile methodology because of it is flexible to switch from one phase (i.e. testing) to another phase (i.e. implantation). We implemented the web application using modern web technologies such as Node.js and Angular which was very new experience for us as the languages were really good to learn so many new commands and languages which allowed us to develop such web application with better functionalities. We learnt those new technologies after doing research and getting reviews for that languages that either that are perfect for our web application. After that we read the MS graph APIs and tested the APIs using the POSTMAN tool. We made angular web application and divided them into several components. MS graph APIs utilized them to complete the functionality.

In testing and evaluation we learnt so many things and experienced so many new tools of testing and validation. We used so many new tools and tested them one by one some of them were automated testing tool and others were manual testing tools. We came to know that how the testing is done, what the testing process is. We did APIs testing using postman tool and angular unit and components testing using the Karma frontend JavaScript testing tool.

# **Chapter 9: References**

# **9.1 References**

1. <https://jellyfish.tech/10-challenges-of-e-learning-during-covid-19/>
2. <https://www.westga.edu/~distance/ojdla/fall53/valentine53.html>
3. <https://asiacall.info/acoj/index.php/journal/article/view/41>
4. <https://www.androidauthority.com/zoom-vs-microsoft-teams-1109959/>
5. <https://stackshare.io/stackups/microsoft-teams-vs-zoom>
6. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7306967/#ref4>
7. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7866934/>
8. <https://www.jotform.com/blog/microsoft-teams-alternative/>
9. <https://docs.microsoft.com/en-us/microsoftteams/teams-channels-overview>
10. <https://www.computerhope.com/issues/ch000619.htm>
11. <https://uxplanet.org/redesigning-an-existing-interface-microsoft-teams-119e098a1cf9>
12. Dhawan, S. (2020). Online Learning: A Panacea in the Time of COVID-19 Crisis. *Journal of Educational Technology Systems*, 4-7.
13. <https://visibleinternet.com/microsoft-teams-vs-google-classrooms/>