**PSG COLLEGE OF TECHNOLOGY**

(Autonomous Institution)

**COIMBATORE – 641 004**

**PROCESS AUTOMATION**

Bona fide record of work done by

**AKASH E** (17Z315)

**MIDHUNESH K** (17Z325)

**BEASHAJ P** (17Z339)

**VARSHITH T** (17Z355)

**ARUN KUMAR I** (18Z462)

Dissertation submitted in partial fulfilment of the requirements for degree of

**BACHELOR OF ENGINEERING**

**Branch: COMPUTER SCIENCE AND ENGINEERING**

Of ANNA University

**MARCH 2020**

………………… …….….…………………..

**Dr. Gopika Rani N Dr.G Sudha Sadasivam**

Faculty guide Head of Department

Certified that the candidate was examined in the viva-voce examination held on

………………….

…………………. …………………

(Internal Examiner) (ExternalExaminer)

**CERTIFICATE**

Certified that this report titled **“** **Process Automation “** , for the Innovation practices lab (15Z611) is a bona fide work of **Akash E** (17Z315), **Midhunesh K**(17Z325), **Beashaj P** (17Z339), **Varshith T** (17Z355), **Arun Kumar I** (18Z462) who have carried out the work under my supervision for the partial ful-filment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science and Engineering in Computer Science and Engineering. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion

**Place: Coimbatore Dr. Gopika rani N**

**Date: Department of Computer Science**

**and Engineering**

**PSG College of Technology**

**Coimbatore - 641004**

**COUNTERSIGNED**

**HEAD**

**Department of Computer Science and Engineering,**

**PSG College of Technology,**

**Coimbatore - 641004**

**ACKNOWLEDGEMENT**

We would like to thank our company **ProPlus Logics** for providing us the opportunity to work on such an interesting project. We would also like to thank our Principal **Dr.K.Prakasan** for providing us the opportunity to develop a project in our field of study.

It is our privilege to express our sincere regards and gratitude to our project guide, **Dr.Gopika Rani K**, Department of Computer Science and Engineering, for her valuable inputs, able guidance, encouragement, whole-hearted cooperation through the duration of our project

**ABSTRACT**

ProPlus Logics and similar companies need works to be done fast and to make things faster automation is required. Here, the project focuses on Process Automation which deals with company projects.

There are many traditional technologies or stacks to do but this project is developed using MEAN stack, which is growing now a days. Traditional technologies use multiple page application for projects like this, but MEAN stack has single page application.

MEAN stack is comparatively complex and difficult but shows greater efficiency and provides great authentication, authorization, supports portability and has flexibility

**LIST OF TABLES**

**TABLES PAGE NO**

Table 6.1…………………………………………………………………………………….. 9

Table 6.2 …………………………………………………………………………………… 13

Table 6.3 …………………………………………………………………………………… 13

Table 6.4 …………………………………………………………………………………… 13

**CHAPTER Page No.**

**Acknowledgement …………………………………………………………………………………... (I)**

**Synopsis ………………………………………………………………………………....... (II)**

**List of Tables …………………………………………………………………………………. (III)**

1. **INTRODUCTION ………………………………………………………………………………..1**
   1. MOTIVATION ……………………………………………………………………................1
   2. PROBLEM STATEMENT ………………………………………………………………….1
   3. OBJECTIVE …………………………………………………………………………………1
   4. SCOPE ………………………………………………………………………………………1
   5. OUTLINE OF PROJECT ……………………………………………………….................1
2. **SYSTEM STUDY ………………………………………………………………………………..2**
   1. LITERATURE RIVEW ……………………………………………………………………...2
      1. MONGO DB ………………………………………………………………………...2
      2. NODE .JS …………………………………………………………………………...2
      3. EXPRESS.JS ……………………………………………………………………….2
      4. ANGULAR.JS ………………………………………………………………………3
3. **SYSTEM ANALYSIS ……………………………………………………………….................4**
   1. FUNCTIONAL REQUIREMENTS ………………………………………………………...4
      1. DATABASE MAINTENANCE……………………………………………………...4
      2. TASK ASSIGNMENT………………………………………………………...........4
      3. ASSIGNING PRIORITY TO ACCESS THE DASHBOARD……………………4
      4. MAINTAINING MEETUPS AND PROGRESS…………………………………..4
      5. PROVIDING SERVICE TO CLIENT……………………………………………...4
   2. NON FUNCTIONAL REQUIREMENTS …………………………………………............4
      1. AUTHENTICATION………………………………………………………………...4
      2. FLEXIBILITY………………………………………………………………………...4
      3. HIGH SPEED AND REUSABILITY………………………………………............4
   3. HARDWARE AND SOFTWARE REQUIREMENTS…………………………………….5
      1. HARDWARE REQUIREMENTS…………………………………………………..5
      2. SOFTWARE REQUIREMENTS…………………………………………………..5
4. **SYSTEM DESIGN …………………………………………………………………..................6**
   1. BLOCK DESCRIPTION ………………………………………………………..................6
      1. ANGULAR ………………………………………………………………….............6
      2. NODE-EXPRESS ……………………………………………………………….....6
      3. MONGO DB ………………………………………………………………………...6
   2. UML USE-CASE DIAGRAM ………………………………………………………...........7
5. **SYSTEM IMPLEMENTATION…………………………………………………………...........8**
   1. UML DIAGRAM……………………………………………………………………..……….8
6. **TESTING …………………………………………………………………………………...........9**
   1. UNIT TESTING………………………………………………………………………..........9
      1. TESTING THE LOGIN PAGE……………………………………………………..9
      2. TESTING THE MANAGER………………………………………………………10
      3. TESTING THE TEAM LEADER…………………………………………….......12
      4. TESTING THE EMPLOYEE……………………………………………………..13
   2. INTEGRATION TESTING…………………………………………………………..........13
7. **RESULT…………………………………………………………………………………...........21**
   1. CHECK WITH VALID MANAGER’S USERNAME AND PASSWORD………………21
   2. CHECK WITH VALID TEAM LEADER’S USERNAME AND PASSWORD…………21
   3. CHECK WITH VALID EMPLOYEE USERNAME AND PASSWORD………….........21
   4. CHECK WITH CLICKING ON CLIENT…………………………………………..……...22
   5. CHECK WITH INVALID USERNAME/PASSWORD…………………………...………22
   6. CHECK FOR WORKING OF EXPANSION PANELS………………………………….22
   7. CHECK FOR ASSIGNMENT OF PROJECTS TO TEAM LEADERS………………..23
   8. CHECK FOR “SAVE CHANGES” BUTTON…………………………………………….23
   9. CHECKING ON “COMPLETE” BUTTON IN ONGOING PROJECTS………...……..24
   10. CHECK FOR GETTING PROJECT DETAILS…………………………………......24
   11. CHECK FOR ABLE TO CREATE TASKS/MODULES…………………………….25
   12. CHECK WHETHER THE TASK ADDED TO THE ASSIGNMENT SECTION WHEN TASK IS CREATED………………………………………………………………25
   13. CHECK FOR ABLE TO ASSIGN EMPLOYEE TO THE MODULE /TASK…......25
   14. CHECK FOR ABLE TO EDIT THE EMPLOYEE ASSIGNED TO EMPLOYE.....25
   15. CHECK FOR ABLE THAT RESPECTIVE TASK ASSIGNED IS UPDATED ONTO THEEMPLOYEE DB OR DASHBOARD………………………………………..26
   16. CHECK FOR NOTIFYING QUOTATION VIA EMAIL OR SMS……………….....26
   17. CHECK FOR ABLE TO VIEW THE STATUS OF EMPLOYEE…………………..26
   18. CHECK FOR DISPLAY TASKS……………………………………………………...27
   19. CHECK FOR THE “COMPLETED” BUTTON……………………………………....27
   20. CHECK WEATHER TASK COMPLETED UPDATED……………………………..28
   21. OVERALL VIEW OF MANAGER DASHBOARD………………………………..…28
   22. GRAPHICAL REPRESENTATION OF ENQUIRY, LEAD, CONVERSION….….28
   23. LINEAR GRAPH OF ENQUIRY WITH PREDICTED DATA………………………29
   24. LINEAR GRAPH OF LEAD WITH PREDICTED DATA……………………………29
   25. LINEAR GRAPH OF CONVERSATION WITH PREDICTED DATA……………..30
   26. GRAPH AFTER FILTERING LEAD………………………………………………….30
   27. ENQIRY CHART FOR SEARCH ENGINE OPTIMIZATION IN SIDE NAVIGATION………………………………………………………………………………31
   28. CLIENT DETAILS OF TODAY FOLLOWUP AND MISSED FOLLOR UP………31
8. **CONCLUSION………………………………………………………………………………….32**
9. **BIBILOGRAPHY……………………………………………………………………………….33**
10. **APPENDIX………………………………………………………………………………………34**
    1. APP MODULE………………………………………………………………………….34
       1. app.componet.html………………………………………………………………..34
       2. app.component.ts…………………………………………………………………35
    2. CLIENT COMPONENT……………………………………………………………….35
       1. client.component.html…………………………………………………………….35
       2. client.compponent.ts………………………………………………………………38
    3. MANAGER COMPONENT……………………………………………………………39
       1. manager.component.html………………………………………………………...39
       2. manager.component.ts…………………………………………………………...41
       3. ak.html……………………………………………………………………………...44
       4. ak.ts…………………………………………………………………………………50
    4. TEAM LEADER COMPONENT………………………………………………………55
       1. teamleader.component.html……………………………………………………..55
       2. teamleader.component.ts………………………………………………………...57
       3. task-dialog.component.html……………………………………………………...60
       4. task-dialog.component.ts…………………………………………………………60
    5. EMPLOYEE COMPONENT…………………………………………………………..60
       1. employee.component.html……………………………………………………….60
       2. employee.component.ts…………………………………………………………..61
    6. NODE PART……………………………………………………………………………62
       1. app.js……………………………………………………………………………….62
       2. clishema.js…………………………………………………………………………67
       3. managerschema.js………………………………………………………………..67
       4. post.js……………………………………………………………………………….67
       5. empscheema.js……………………………………………………………………68
       6. personscheema.js…………………………………………………………………68

**CHAPTER 1**

**INTRODUCTION**

The project is in the domain of web development where people can do their tasks without human face-to-face interaction (mostly). The project is to develop PROCESS AUTOMATION for a company. This makes the progress of the projects in a viewable and analytical way, which helps the employees to do their role in very smooth way.

**1.1 Motivation:**

As in any company people/clients approach and propose/expect some sort of requirements from company. In order to manage all those tasks a good automation should be made. So, for the sake of making their process efficiently an interface with backend should be made to make all tasks easy.

**1.2 Problem Statement:**

The project should contain dashboard for the involved people. A particular dashboard should be made for the clients in order to save the requirements who are coming to the office and giving specifications. And manager should be responsible for viewing all the manipulations of data (i.e.) Manager should keep count of enquiries, leads and converters (converting from enquiry to lead). And to view client’s arrival source (Justdial, SEO, etc). And team leader dashboard for checking status of the team. All the data should be stored in the cloud.

**1.3 Objective:**

This can be efficiently implemented using MEAN stack. This stack consists of MongoDB, Express.js, Angular.js, Node.js. Where all these are connected and communicated via developing some set of instructions onto it which helps to interact among themselves.

**1.3 Scope:**

As every company requires to track the status of their project. So, all tracking applications such as

* Web Development (Full Stack/ Mean Stack)
* App Development
* Network oriented
* Data analysts

will come under the scope.

**1.4 Outline of Project:**

The project is to analyse the tracking of all the tasks of the employees under manager and to take decisions accordingly. And assign the tasks to the employees by the team leader.

**CHAPTER 2**

**SYSTEM STUDY**

**2.1 Literature Review:**

In today’s day and age, technology is growing very drastically with the new devices in hardware thereby which increases the software growth. The key task of increasing the software is by performance. Traditionally, web development has been using technologies such as PHP, JAVA servlets, ASP.NET, ruby etc. But as compared to the MEAN these don’t show the performance as MEAN does.

**2.1.1 Mongo dB:**

Mongo is basically NoSQL database where there wont be any columns or rows. This database stores the data in format of JSON. It has dynamic schema therefore it has very good popularity to develop applications. The advantage is no need to learn separately like SQL. A package called mongoose present over the node where it handles interactions between angular and node. Data in the database is stored in documents which have format of BSON (binary encoded JSON).

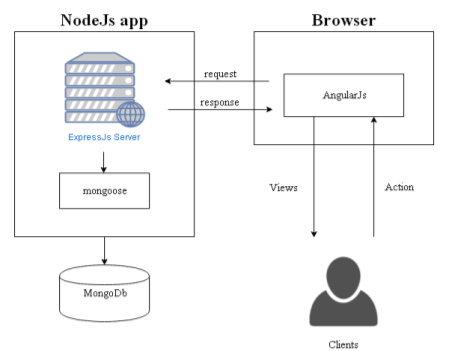
**2.1.2 Node.js:**

Node is most important component in MEAN stack. Node is a run time environment for executing JavaScript code. After 2009 v8 engine (used by google) and embedded inside C++ program and that’s called node.

And this is done for less memory consumption and greater performance. Node works in asynchronous mode where a separate thread will be assigned in order to service the request. Even though it works on a single thread it is capable of providing service to the multiple clients.

**2.1.3 Express.js:**

Express is a being a lightweight web application framework that helps developers in organizing web applications in MVC architecture on the server side. It manages everything from routing to handling requests and views. Express.JS makes it easier to write secure, modular and fast Node.JS applications with higher efficiency, reliability and less duplications. Express.JS has memory storage for sessions and organizes web applications by hiding most of the inner working details of Node.js.



**2.1.4 Angular.js:**

Angular is an open source JavaScript library developed and maintained by google. It has the power of handling the entire client’s side tasks such as viewing, raising events and etc which normally used in front-end. The main advantage of the angular is that it provides single page application, that loads entire web-page on a single-initial request. The developers have freedom to choose whichever way they want to implement the angular application. Angular uses Directives, which are HTML mark-ups which appear as html elements, attributes or even CSS classes. The directives are used to bind data, and DOM manipulations. The directive ng-app is used to define the Angular application.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 Functional Requirements:**

**3.1.1 Database Maintenance:**

Data for upcoming, ongoing and completed projects and data for clients, managers, team leaders are maintained in the cloud

**3.1.2 Task Assignment:**

Manager only responsible for assigning task to team leads. In task assignment the task is assigned to team lead automatically based on platform and current working status

**3.1.3 Assigning Priority to access the dashboard:**

Here we assign priority to the each and every entity and accessing dashboard only based on this priority. Here manager has high priority so he has rights to access all dashboards indirectly by getting lower priority data. Similarly, employee gets only his/her data as employee is least priority.

**3.1.4 Maintaining Meetups and Progress:**

After assigning the task to team lead. Team lead sequentially gets feedback from clients in particular intervals and team lead notify the progress of the project to the clients

**3.1.5 Providing service to client:**

System will provide information to client through SMS or Email when 5every module completed and it also informs the client for option about modification

**3.2 Non-Functional Requirements**

**3.2.1 Authentication**

Authentication is the crucial functionality of our project because not only for security it also maintains the session to keep the dashboard away from unwanted access.

**3.2.2 Flexibility**

MEAN allows you to test an application on cloud platform easily after successful completion of a development process. Application can be easily developed, tested and introduced in the cloud. It also allows you to add extra information simply by adding the field to your form. MongoDB, specifically designed for the cloud, provides full cluster support and automatic replications.

**3.2.3 High Speed and Reusability**

Node.js is speedy and ascendable because of its non-blocking architecture. Angular.js is an open source JavaScript framework that offers maintenance, testability, and reusability. Powerful directives of this framework progress into great testability and domain specific language.

**3.3 Hardware and software requirements**

**3.3.1 Hardware requirements**

RAM: 2GB

PROCESSOR: Intel core i5

HARD DISK: 100GB

**3.3.2 Software Requirements:**

BROWSER: Google chrome

SERVER: Node.js

BACK-END: MongoDB

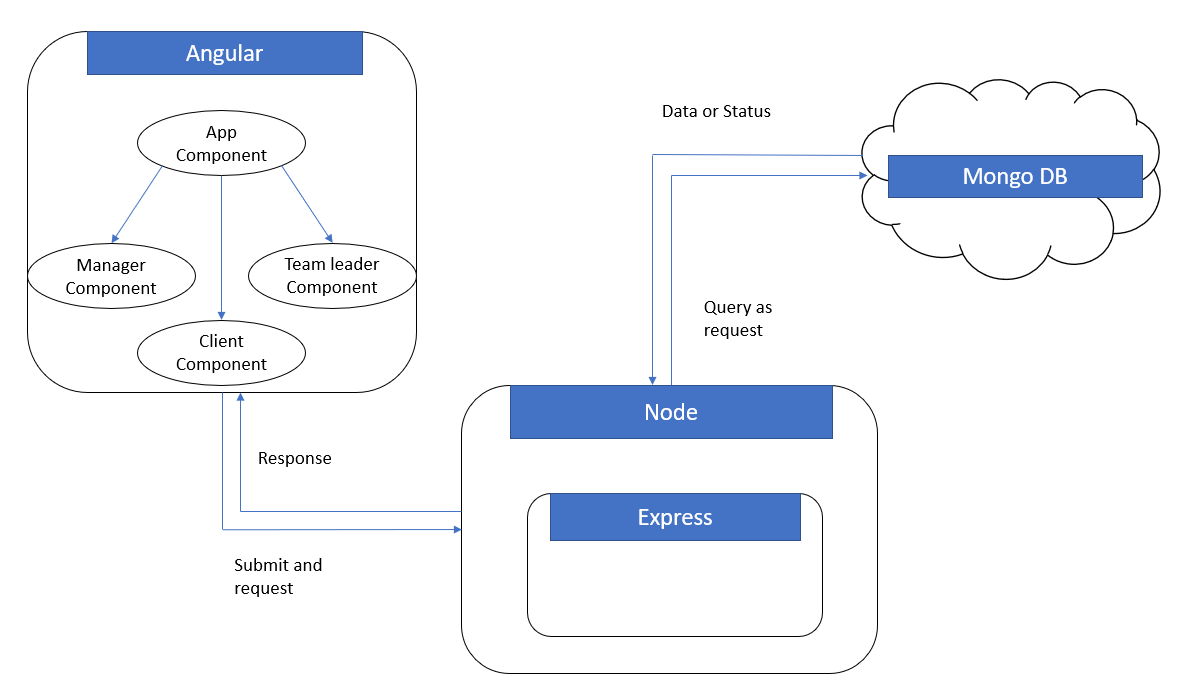
FRONT-END: Angular.js

ROUTING-LANGUAGE: Express.js

**CHAPTER 4**

**SYSTEM DESIGN**

The below flowchart describes the major steps of the project implementation



**4.1 Block Description:**

**4.1.1 Angular:**

This is the user visible block, when app component is loaded. Once its loaded login/component choosing availability is present to the end user. From there according to the validation the end user can be directed to either manager/client-form/team leader dashboards. There end user can do their own functionalities. And this takes the responsibility of viewing data requested by end-user.

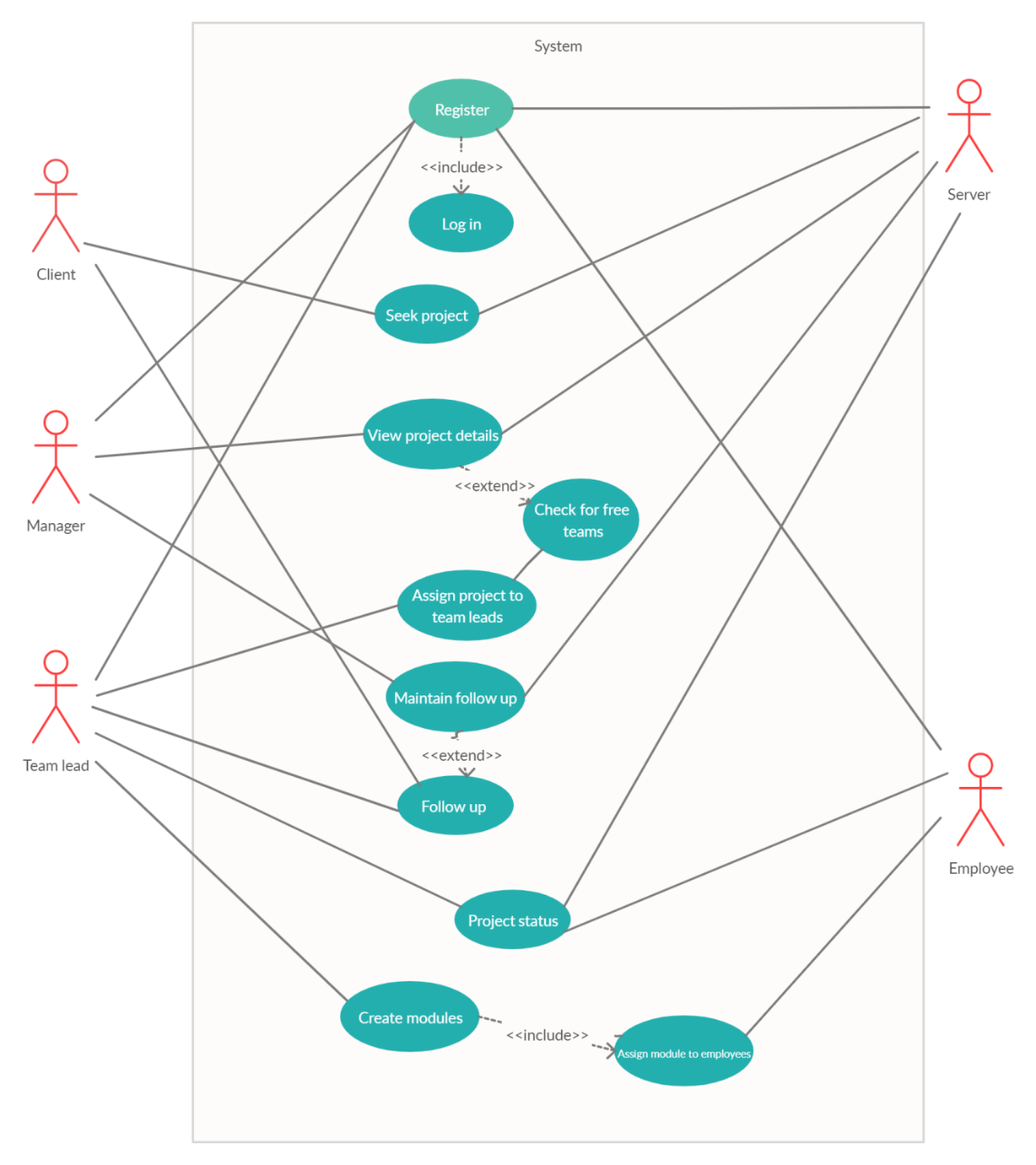
**4.1.2 Node-Express:**

When a request or data is coming from the angular part node will consider those and an express embedded in node will identify the route and handle according to that route. Then instructions in that route will be done like querying the database such as saving the content retrieving the data, etc. And node is capable of taking the data that is sent by DB and give it back to the angular component who has requested.

**4.1.3 Mongo DB:**

As the name represents this is responsible of doing all the querying process. It gets the query from the node and process it and submits the respective resultant data to the node.

**4.2 UML USE-CASE Diagram**



**CHAPTER 5**

**SYSTEM IMPLEMENTATION**

**5.1 UML Diagrams:**

****

**CHAPTER 6**

**TESTING**

# UNIT TESTING

Unit testing is a level of software testing that is used to verify the functional performance of each modular component of the entire system. The main goal of unit testing is to validate that each unit of the system performs as designed. Unit testing is conducted to verify the functional performance of each modular component of the software. The white-box testing techniques were employed for unit testing.

# TESTING THE LOGIN PAGE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MODULE NAME** | **TEST SCENARIO** | **EXPECTED RESULT** | **ACTUAL RESULT** | **PASS/FAIL** |
| Testing Login Page | Check with valid Manager’s Username and password | Redirecting to the manager dashboard and displaying the respective manager content | On clicking login redirecting to the manager dashboard with some minimal delay displaying managers details | PASS |
| Check with valid Team Leader’s Username and password | Redirecting to the team leader dashboard and displaying the respective team content | On clicking login redirecting to the team leader dashboard with some minimal delay displaying team’s details | PASS |
| Check with valid Employee Username and password | Redirecting to the employee dashboard and displaying the respective employee content | On clicking login redirecting to the employee dashboard with some minimal delay displaying employee details | PASS |
| Check with clicking on client | Loads client form to fill the requirements for developing the software | On clicking client, client form is loaded without delay | PASS |
|  | Check with invalid username/ password | Alert symbol should be displayed | Once login is clicked with invalid login and password | PASS |

**Table. 6.1** Testing the login page

# TESTING THE MANAGER

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MODULE NAME** | **TEST SCENARIO** | **EXPECTED RESULT** | **ACTUAL RESULT** | **PASS/FAIL** |
| Testing the manager dashboard  (Part 1) | Checking the functionalities of the list of buttons related to enquiry | **Today enquiry:**  It must display the list of projects for enquiry which are held in today.  **This week enquiry:**  It must display the list of projects which are enquired within this week.  **This month enquiry:**  It must display the list of projects which are enquired within this month.  **Yesterday enquiry:**  It must display the list of projects enquired yesterday.  **Last week:**  It must display the list of projects enquired last week.  **Last Month:**  It must display the list of projects enquired last month. | All the enquired and enquiring projects are listed in the dialog box and maintained in the daily manner. | PASS |
| Checking the graphical representation of data for enquiry, lead, conversion and analyzing the predicted data of upcoming months. | **Overall:**  It must display the bar chart for overall data of lead, enquiry and conversion in monthly basis.  **Enquiry:**  It must predict the data of enquiry for upcoming months and it must show the predicted result in line chart.  **Lead:**  It must predict the data of lead for upcoming months and it must show the predicted result in line chart.  **Conversion:**  It must predict the data of conversion for upcoming months and it must show the predicted result in line chart. | **Overall:**  All the data’s of the lead, enquiry and conversion are displayed using bar chart.  **Enquiry:**  It displays the predicted data of enquiry for upcoming months in line chart.  **Lead:**  It displays the predicted data of lead for upcoming months in line chart.  **Conversion:**  It displays the predicted data of conversion for upcoming months in line chart. | PASS |
| Checking the expansion panel for missed and today follow ups | **Missed follow up:**  It must display the table of client details whose follow ups are missed.  **Today follow up:**  It must display the table of client details whose follow ups are held today. | All client details are listed in table based on today follow up, missed follow up. All details are updated based on completion of follow ups | PASS |
| Checking the tab of other enquires. | **Overall:**  It must display the graph for overall data of other enquiries like SEO, Google etc. And graph must be stacked based on enquiry.  **Other graphs:**  Other tabs must display the radar graph for their individual enquiry data’s and it must display the last enquiry registered. | All data’s for enquiries are displays using graph and last enquiry also registered also displays separately. | PASS |
| Testing the manager dashboard  (Part -2) | Check for working of expansion panels | On click is toggle its state from expansion-compression | Clicking compresses and expands and displaying its own functions | PASS |
| Check for assignment of projects to team leaders | On clicking dropdown available teams are displayed. One should be chosen among them. The respective project automatically shifts from future projects to ongoing projects. | Once a dropdown is selected selection of team is possible. And it had shifted to ongoing projects | PASS |
| Check for “save changes” button | If “save changes” button is not clicked then that request won’t be sent to database | Checked without clicking “save changes” and refreshed the page changes are not updated | PASS |
| Checking on “complete” button in ongoing projects | Once the button is clicked the project should move from ongoing to “completed” projects | On clicking the “completed” button the project is moved from ongoing to completed section | PASS |
|  | | | |

**Table. 6.2** Testing the background execution

# TESTING THE TEAM LEADER

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MODULE NAME** | **TEST SCENARIO** | **EXPECTED RESULT** | **ACTUAL RESULT** | **PASS/FAIL** |
| Testing team leader dashboard | Check for getting the project details | Once the login details are verified the project details should be visualized | Project details are displayed with minimal delay. | PASS |
| Check for able to create tasks/Modules | At view tasks section if the “Create” button is clicked dialog box opens and asks for Task/Module and entered will append to the available tasks/modules | On clicking “Create”, Dialog box is displayed at center of the screen asks for task, on clicking add it appends with current tasks/modules | PASS |
| Check weather the task added to the assignment section when task is created | Once add is clicked while creating the task, the task should be added in assignment section | On clicking the add automatically task is added into the assignment section | PASS |
| Check for able to assign employee to the module/task | In assignment section a dropdown is present if selected it shows to select employees belong to particular team | On clicking dropdown all employees are listed and can able to choose one among them. | PASS |
|  | Check for able to edit the employee assigned to employee | After selecting it should not be modified because inconsistency might occur | Once an employee is selected from the drop down the dropdown will be disabled. | PASS |
| Check for able that respective task that assigned is updated onto the employee dB or dashboard | Once a task is assigned to an employee, that employee should view that task so employee can follow up the task | Once the task is assigned it is updated to employee database so once the employee login, he/she can view it. | PASS |
| Check for notifying quotation via Email or SMS | Quotation can be written on the input text area and can send the notification | Once the notify button is clicked, the text entered I text area will be sent as body of the message to client specified software | PASS |
| Check for able to view the status of the employee (free or already assigned) | Able to view who are free and busy so that leader gets an idea | Yes, Can able to view The status of the employee. | PASS |

**Table. 6.3** Testing the team leader

# TESTING THE EMPLOYEE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MODULE NAME** | **TEST SCENARIO** | **EXPECTED RESULT** | **ACTUAL RESULT** | **PASS/FAIL** |
| Testing Employee dashboard | Check for displaying tasks | Once the login formality completed, pending tasks and completed tasks to be displayed | On logging pending and completed projects are displayed | PASS |
| Check for clicking the “completed” button | Once the “completed” button is clicked the task should shift from pending to completed tasks | On clicking “completed” tasks shifts from pending to completed tasks | PASS |
| Check weather task completed updated | Once the “completed” button is clicked the respective task in team leader dashboard should be checked | On clicking the respective task in team leader dashboard is checked | PASS |

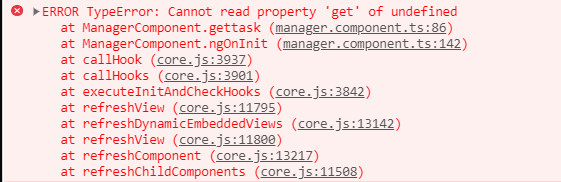
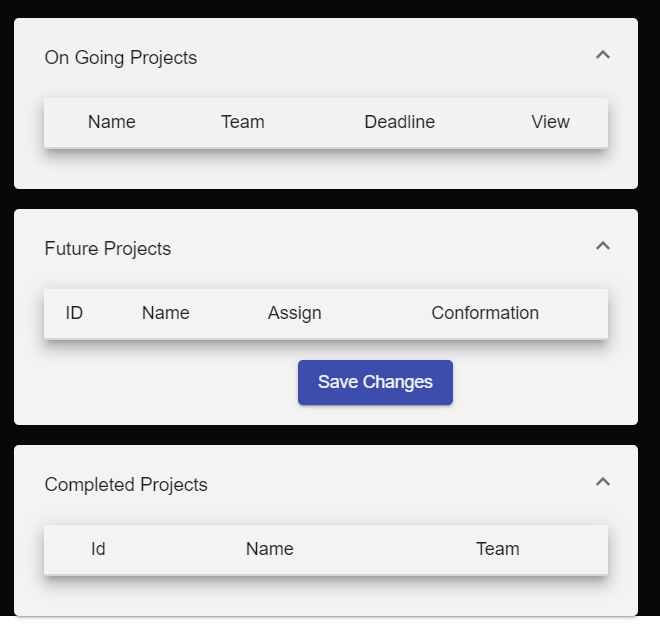
**Table. 6.4** Testing the employee

# INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. Integration testing is the complete testing of the set of modules which makes up the system. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when put them together. The problem arises in interfacing.

The modules are all integrated and the test cases mentioned above are tested. The system works effectively by satisfying the expected result for each test case.

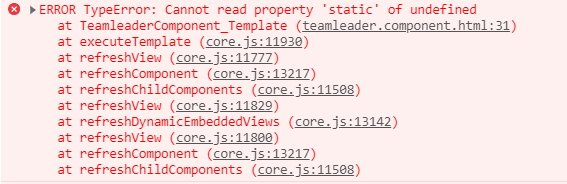
**Error:**

**Solution:**



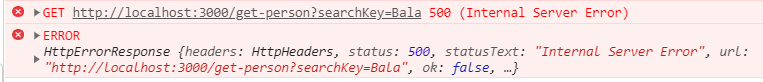
**Error:**



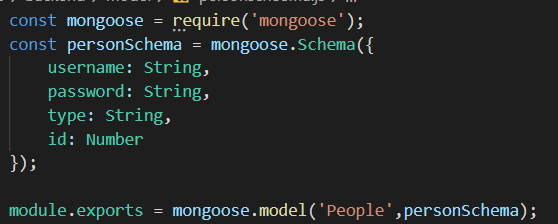
**Solution:**



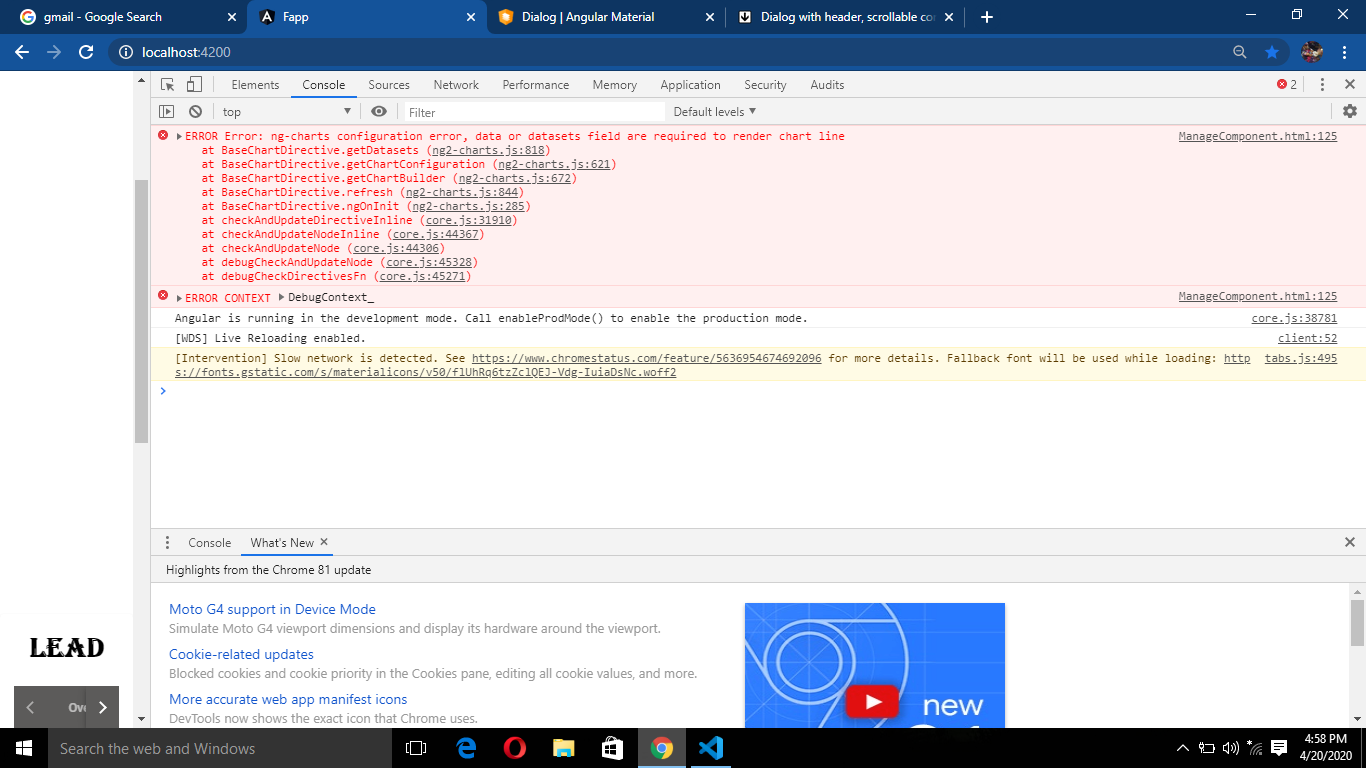
**Error:**

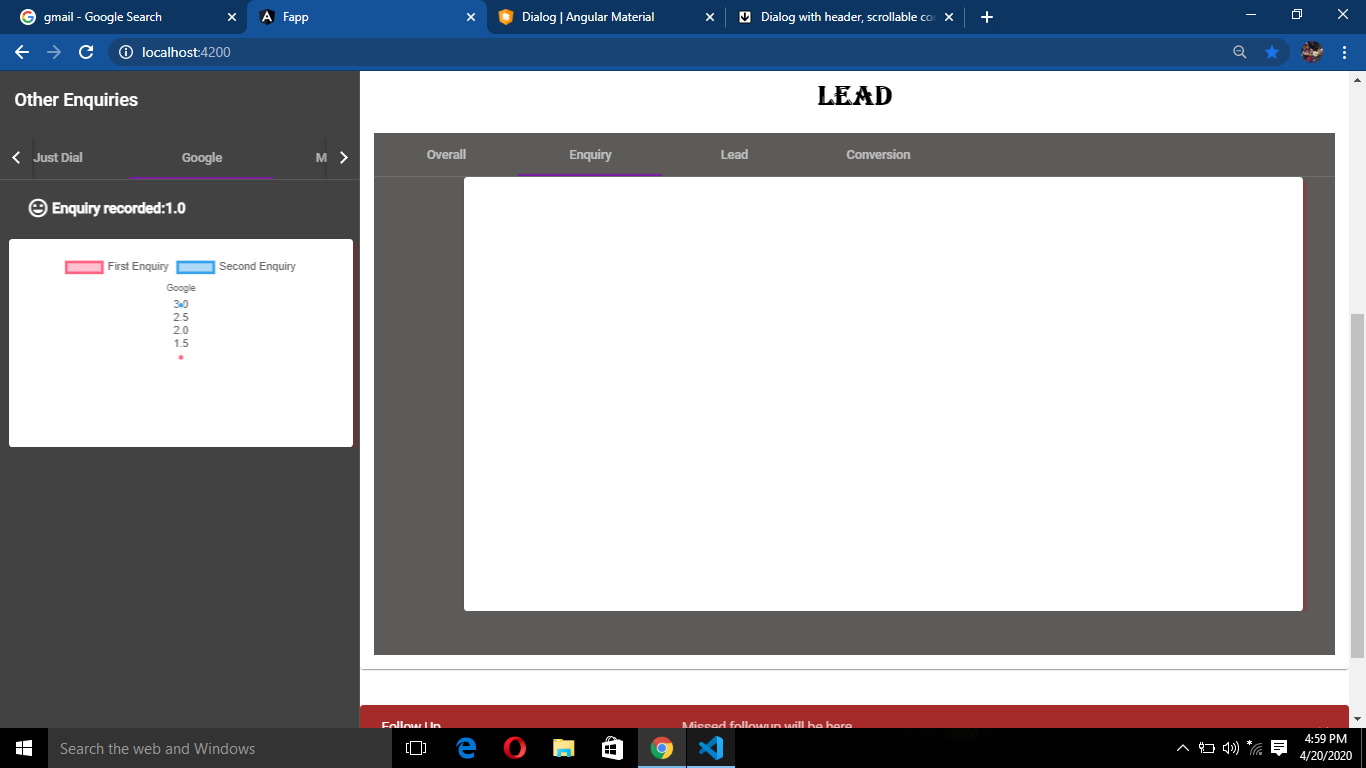


**Solution:**

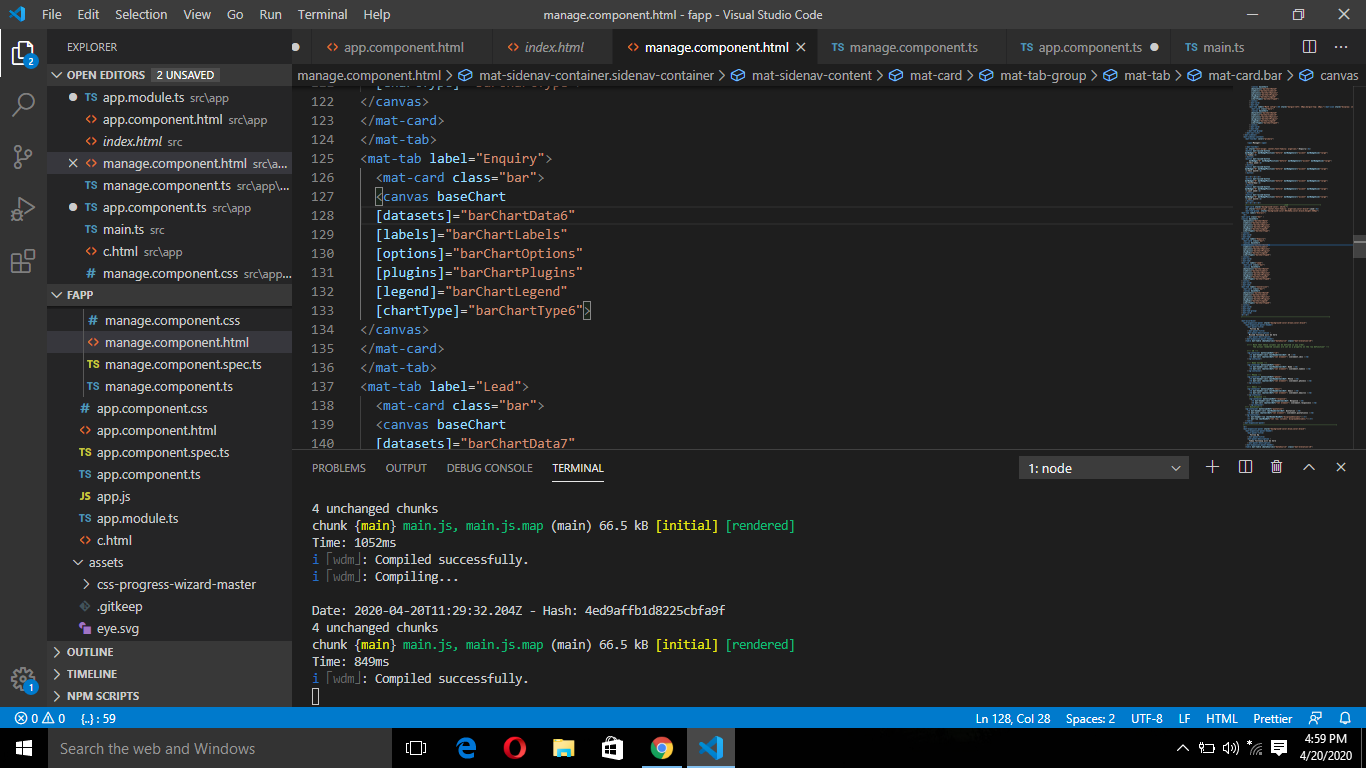


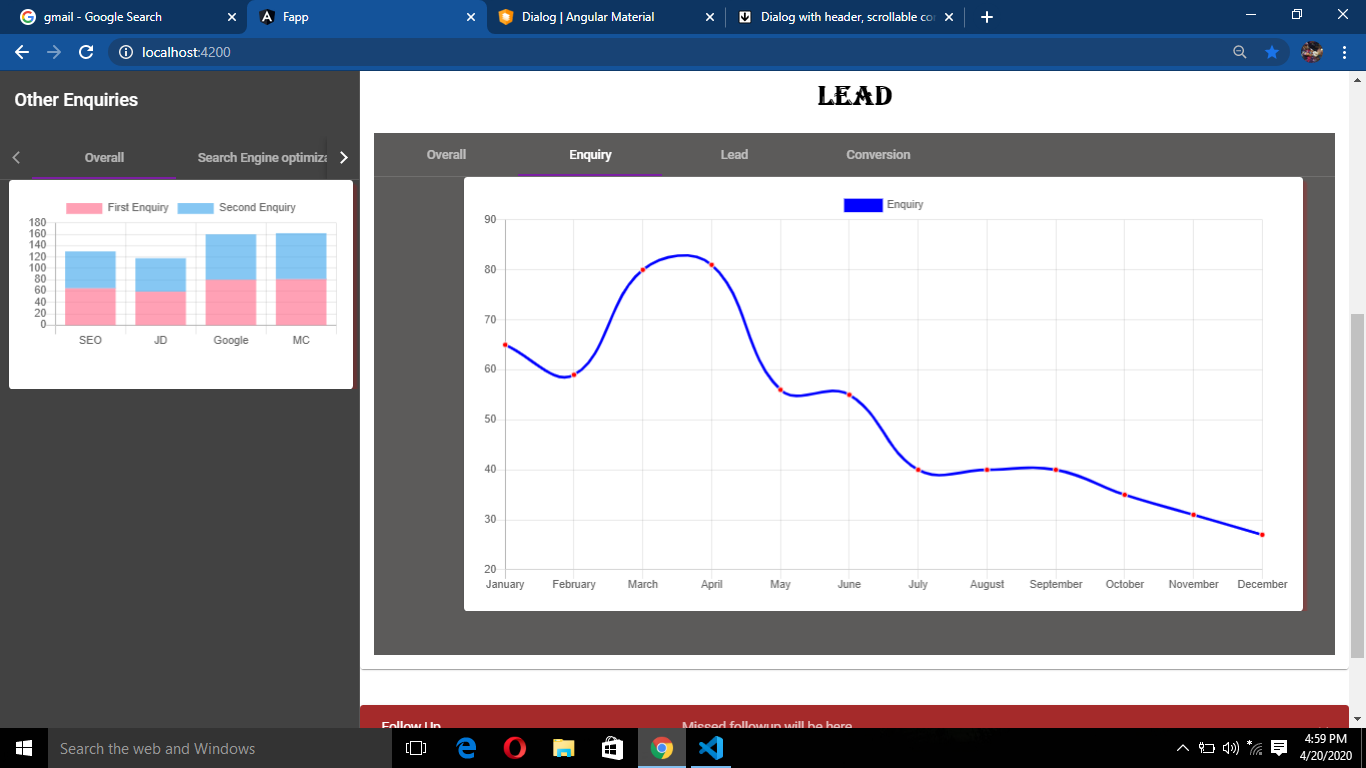
**Error:**

****

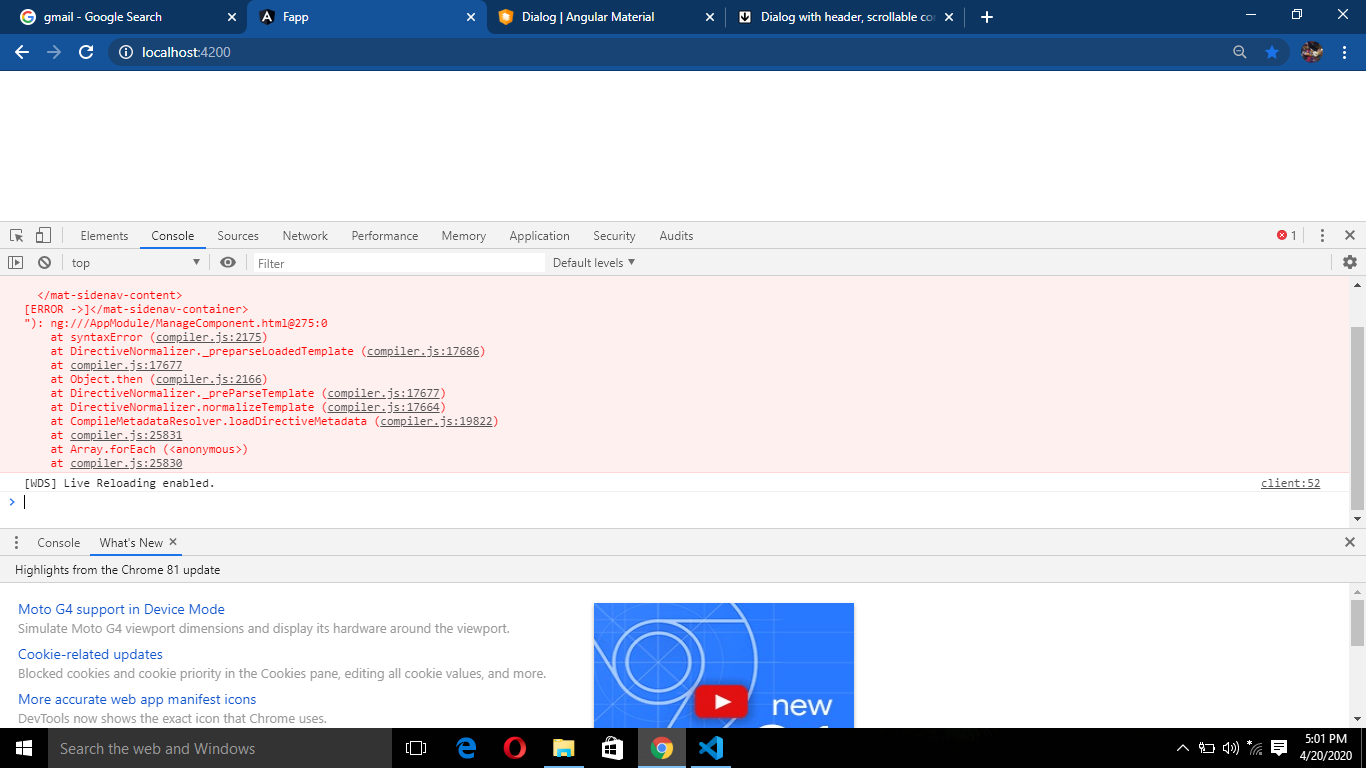
****

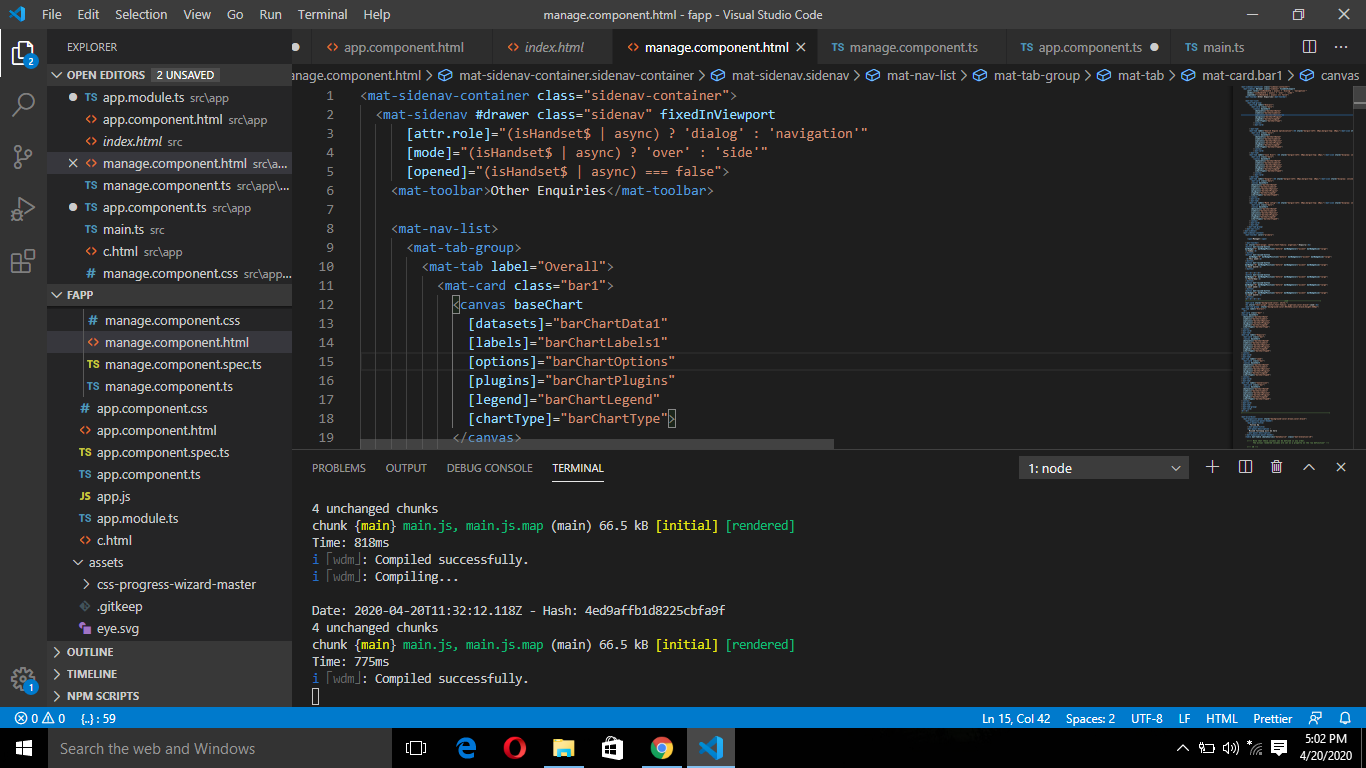
**Solution:**

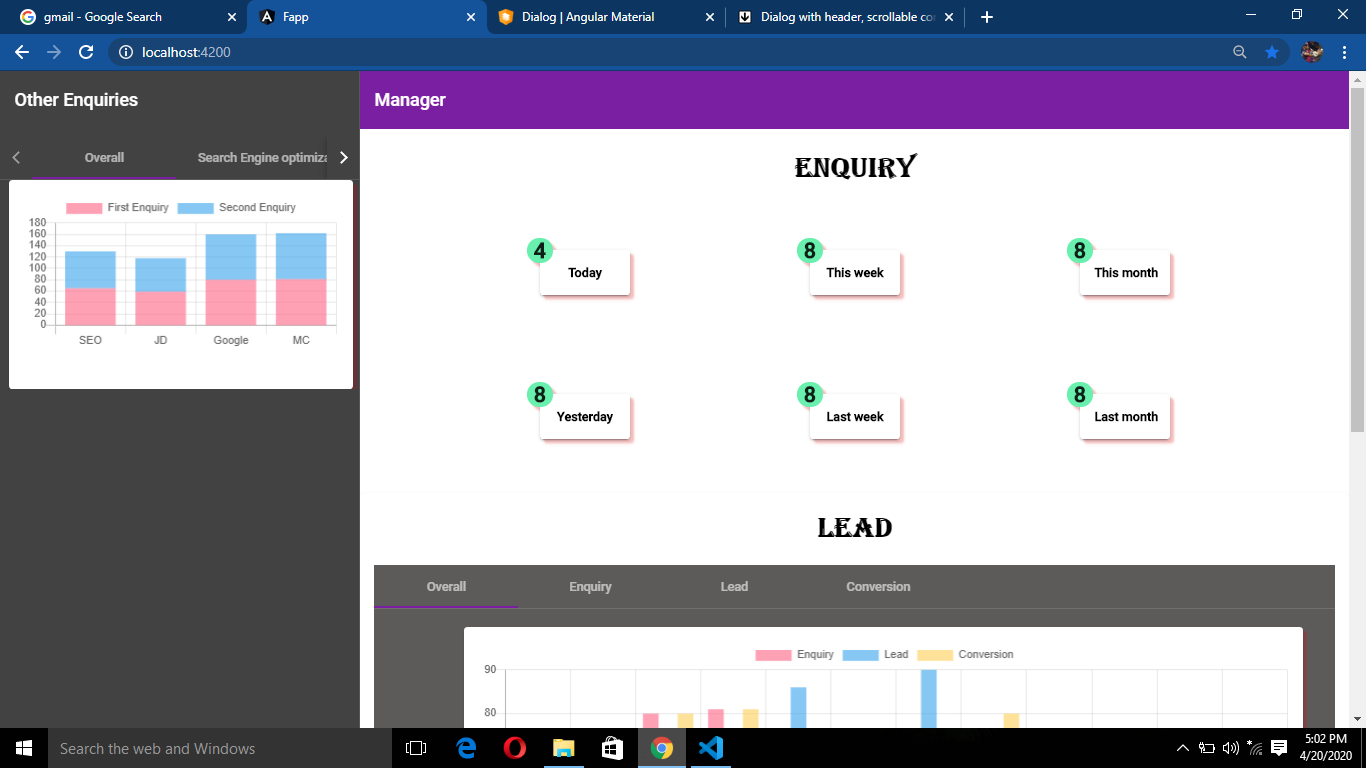
****

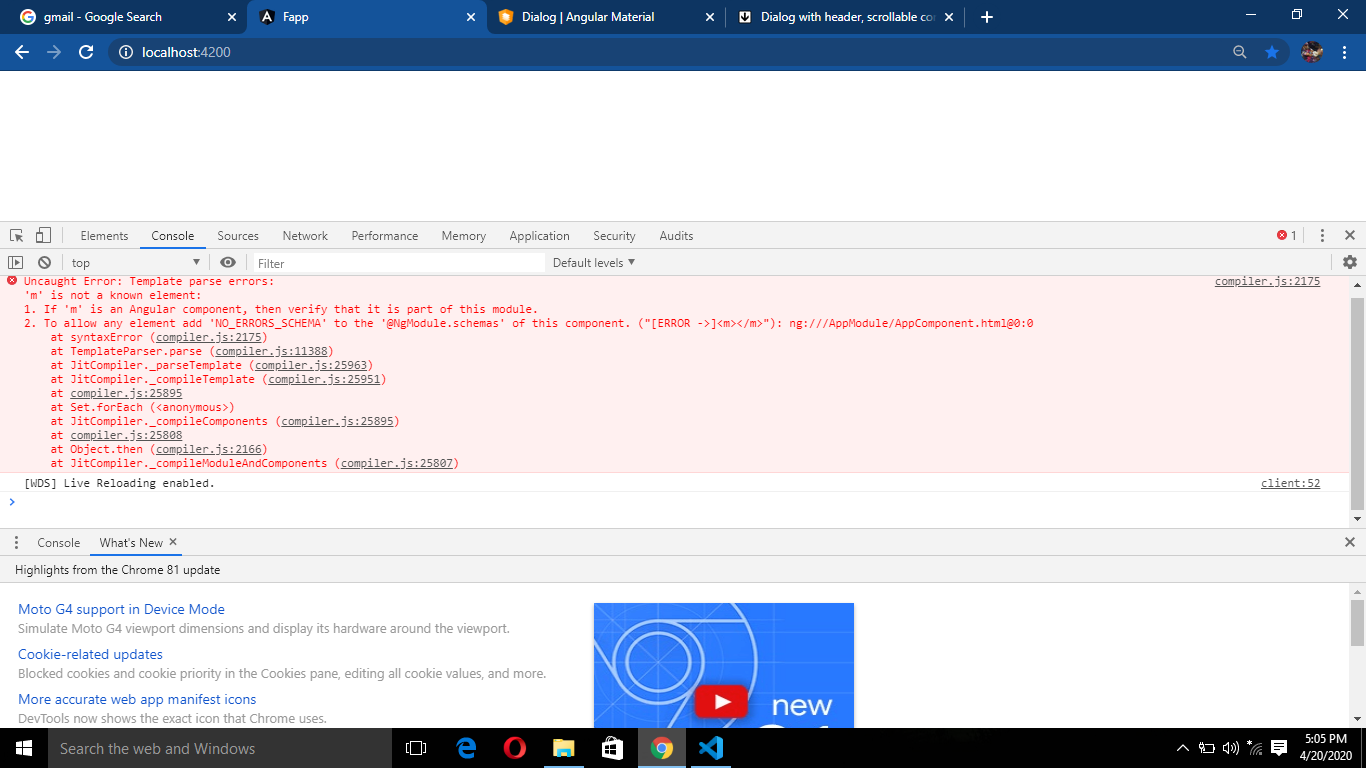
****

**Error:**

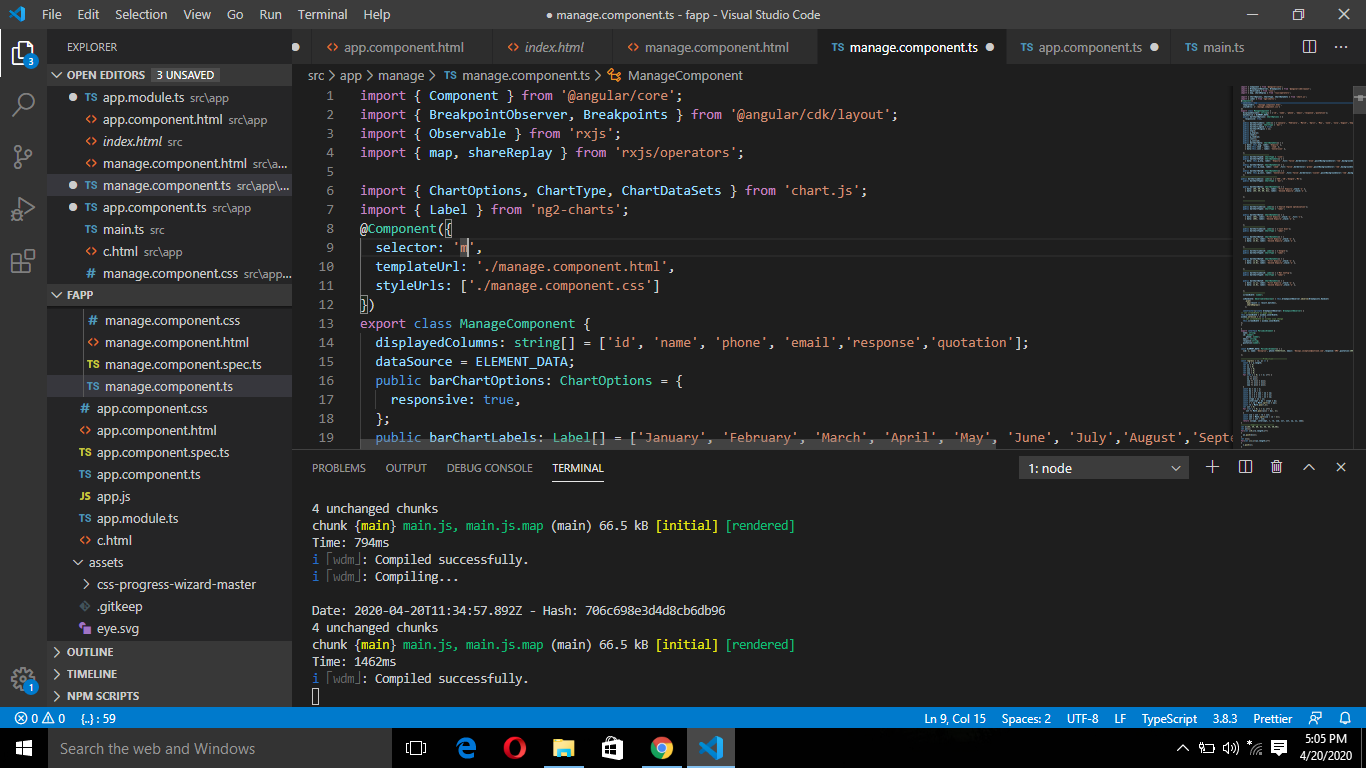
****

**Solution: **

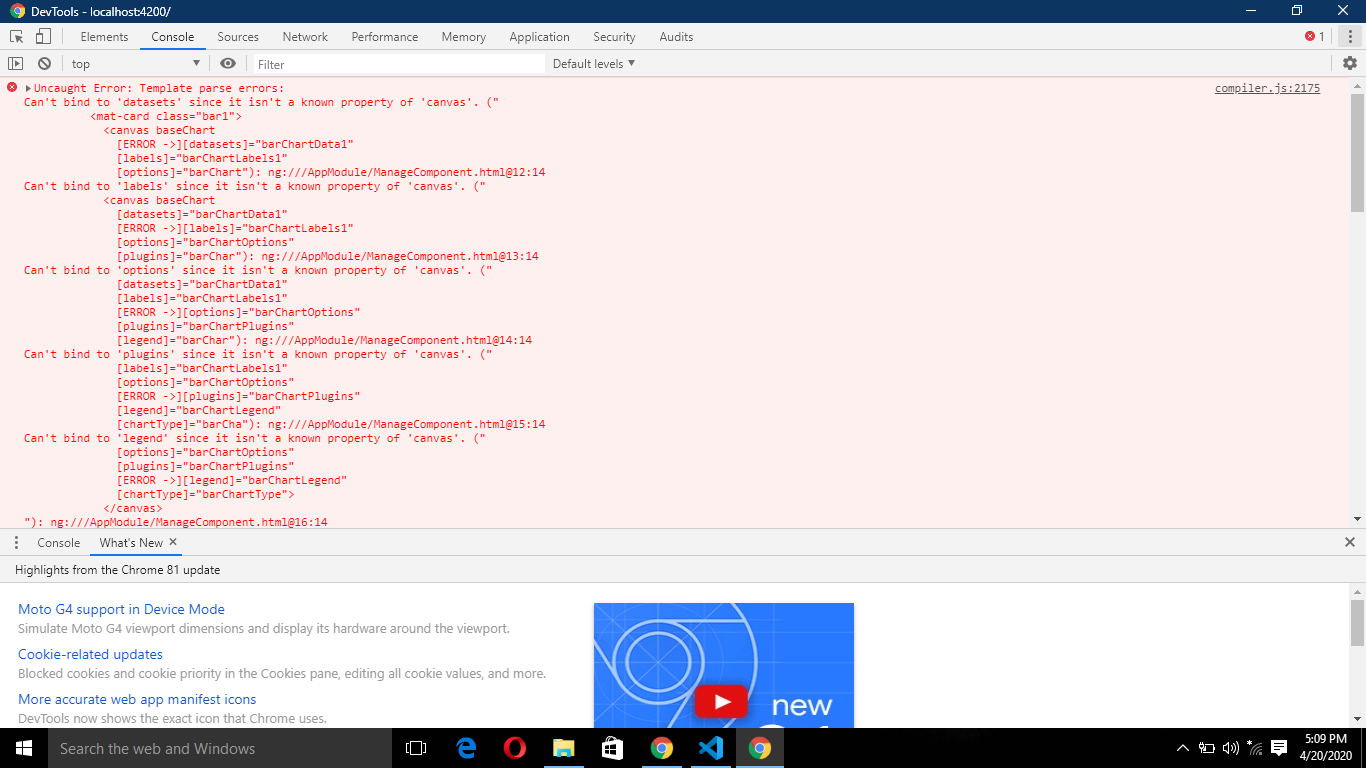
****

**Error: **

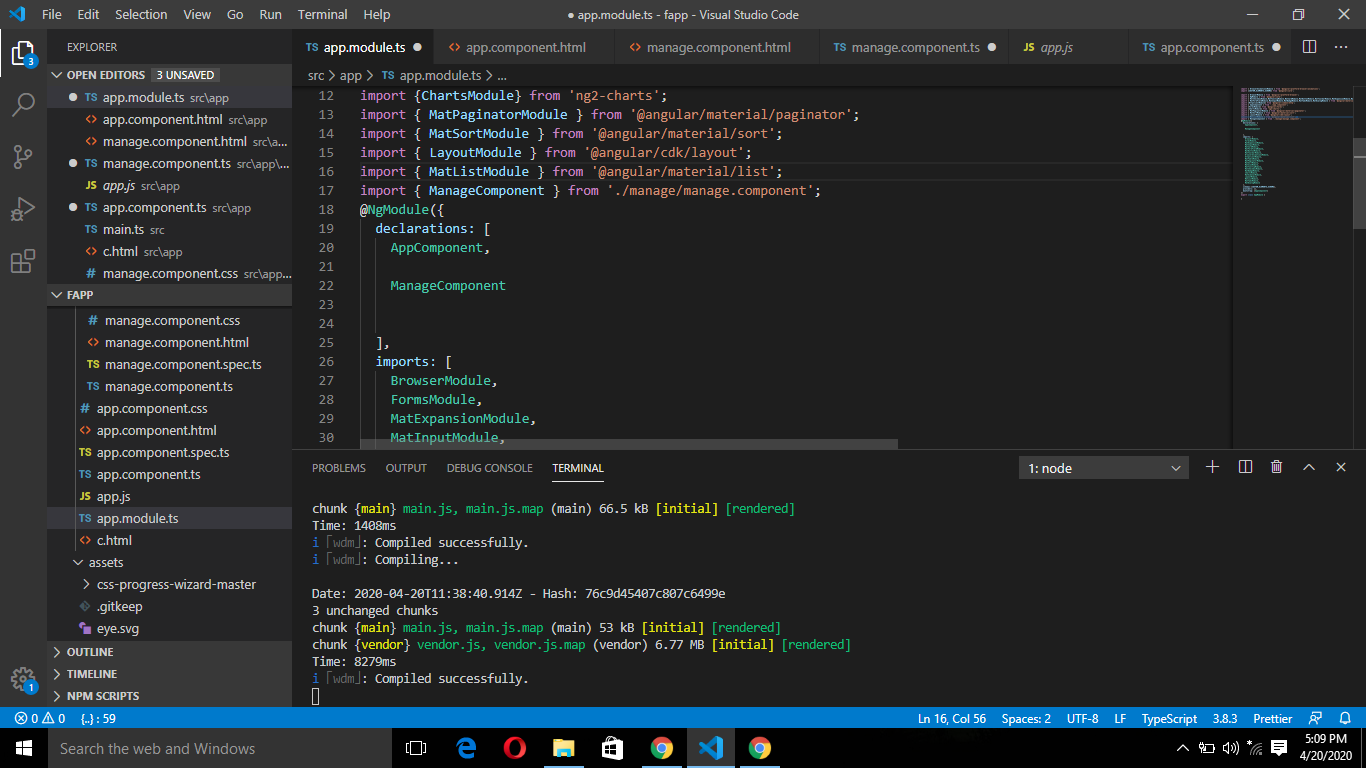
**Solution:**

****

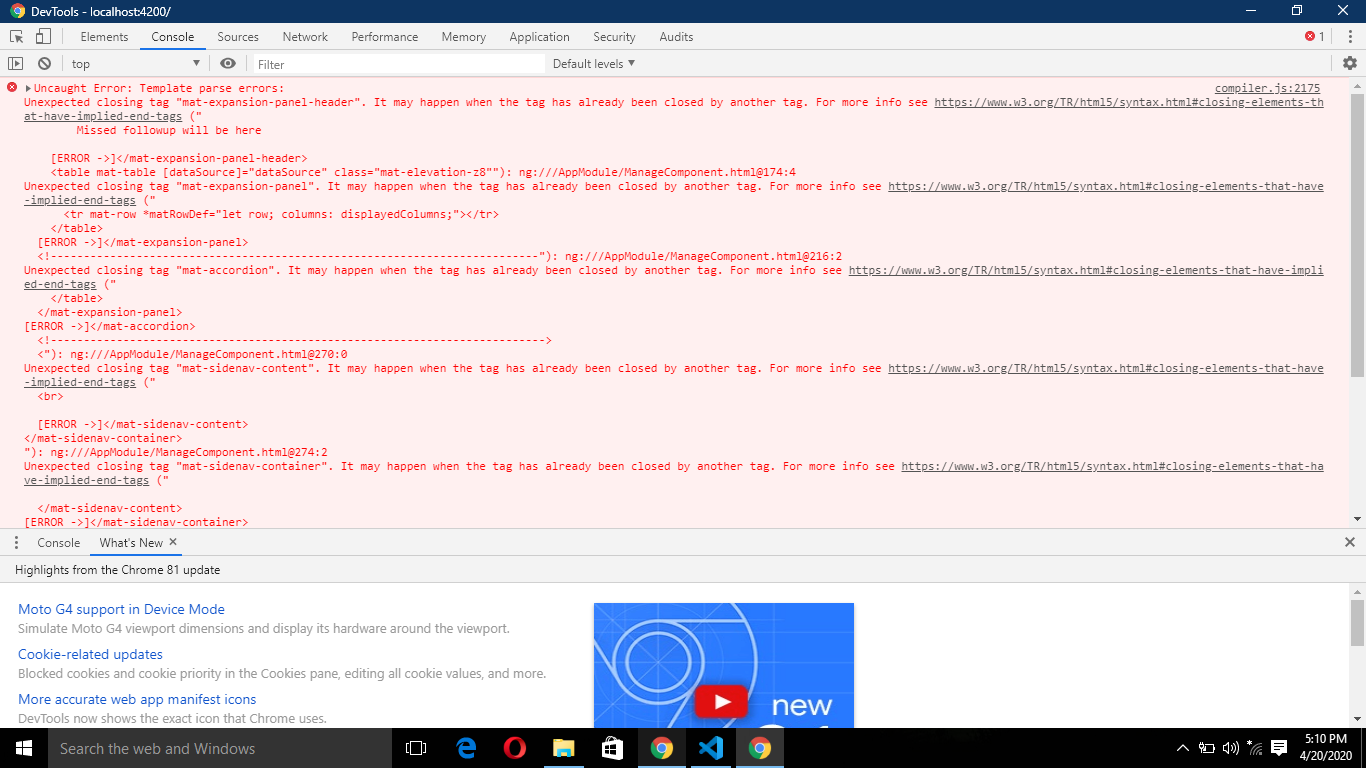
**Error:**

****

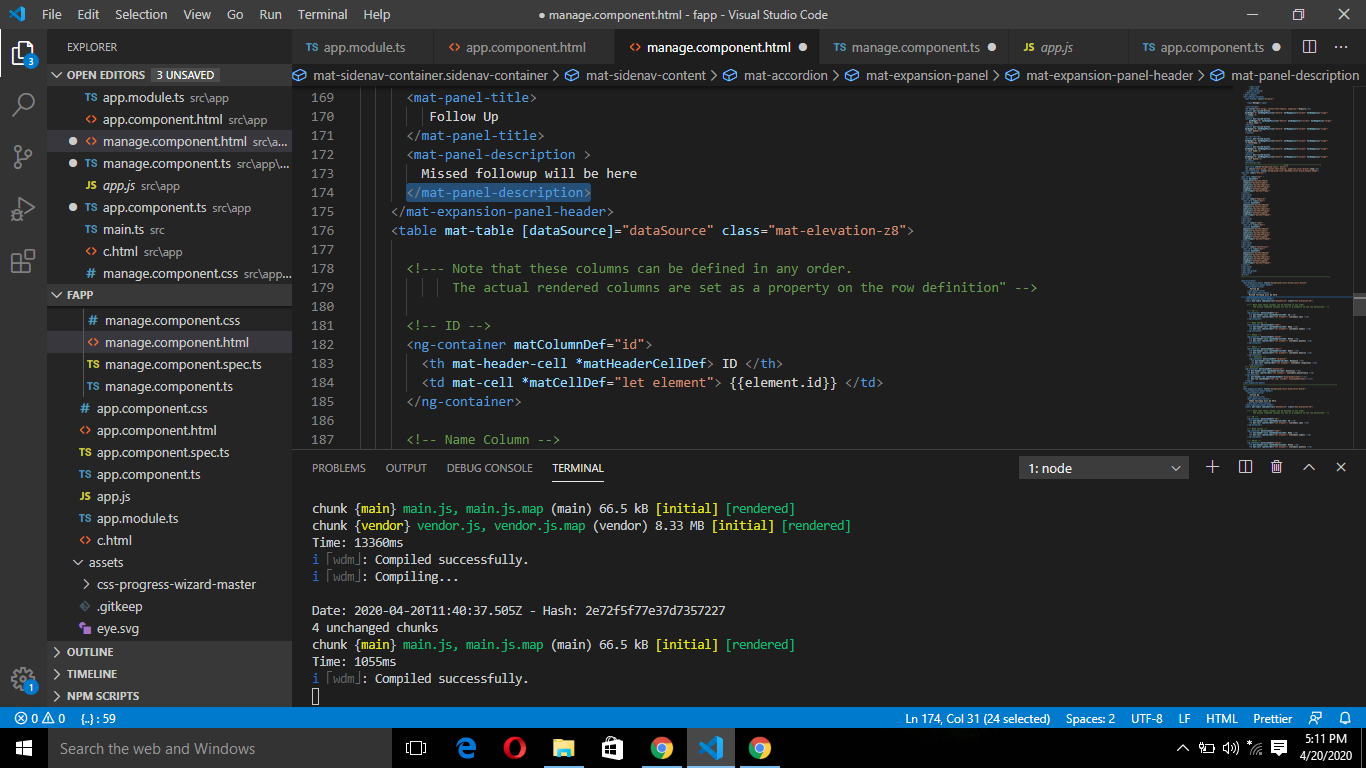
**Solution:**

****

**Error:**

****

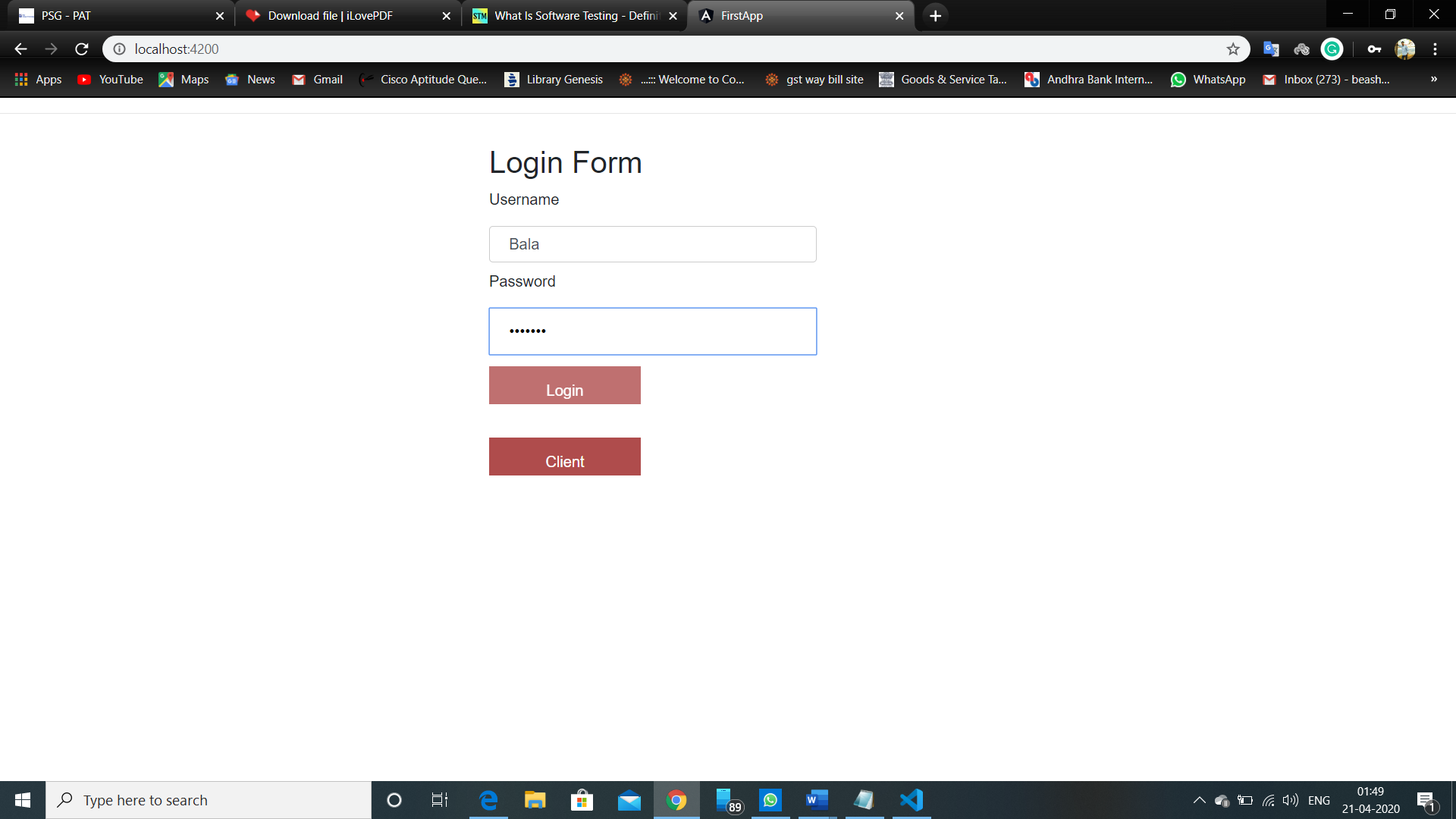
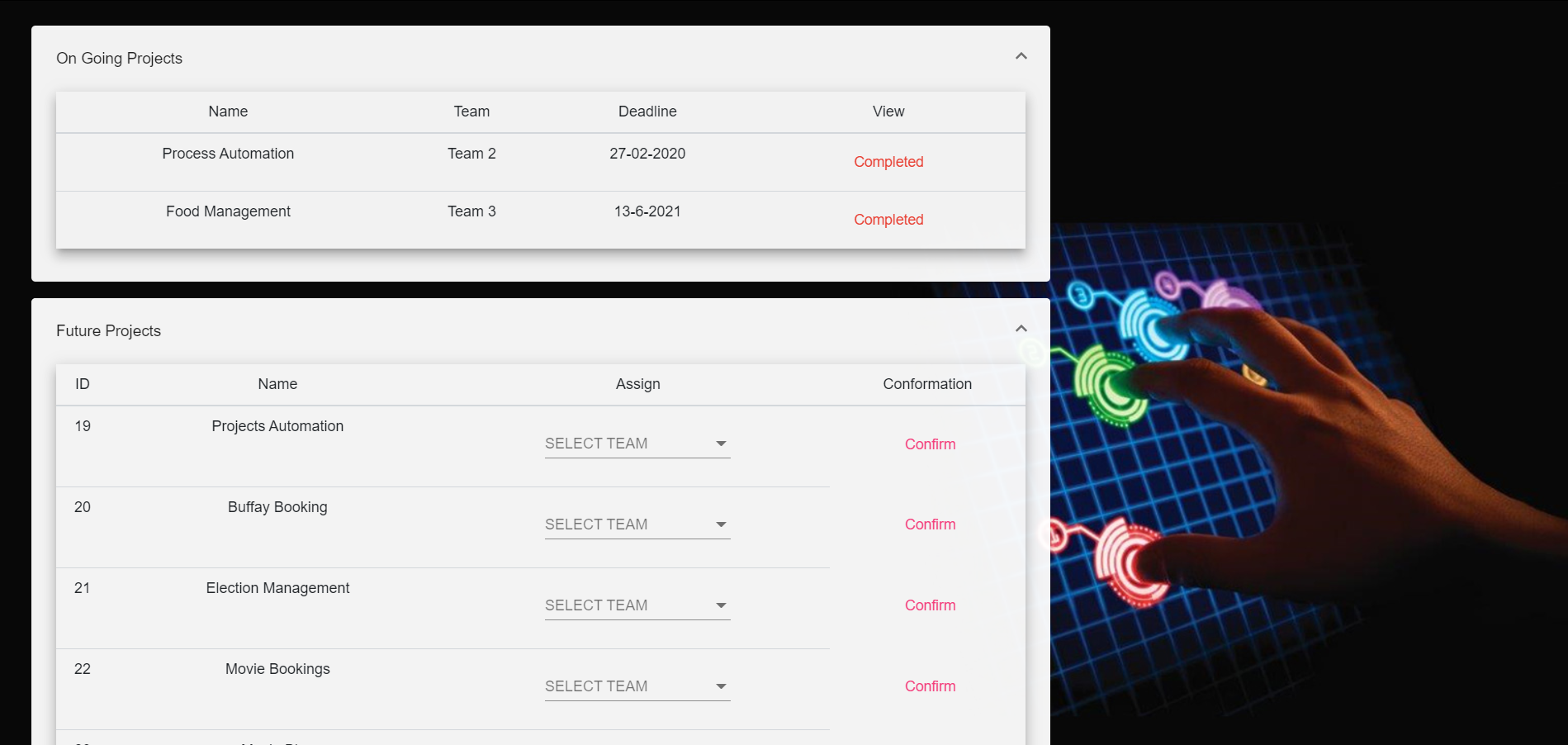
**Solution:**

****

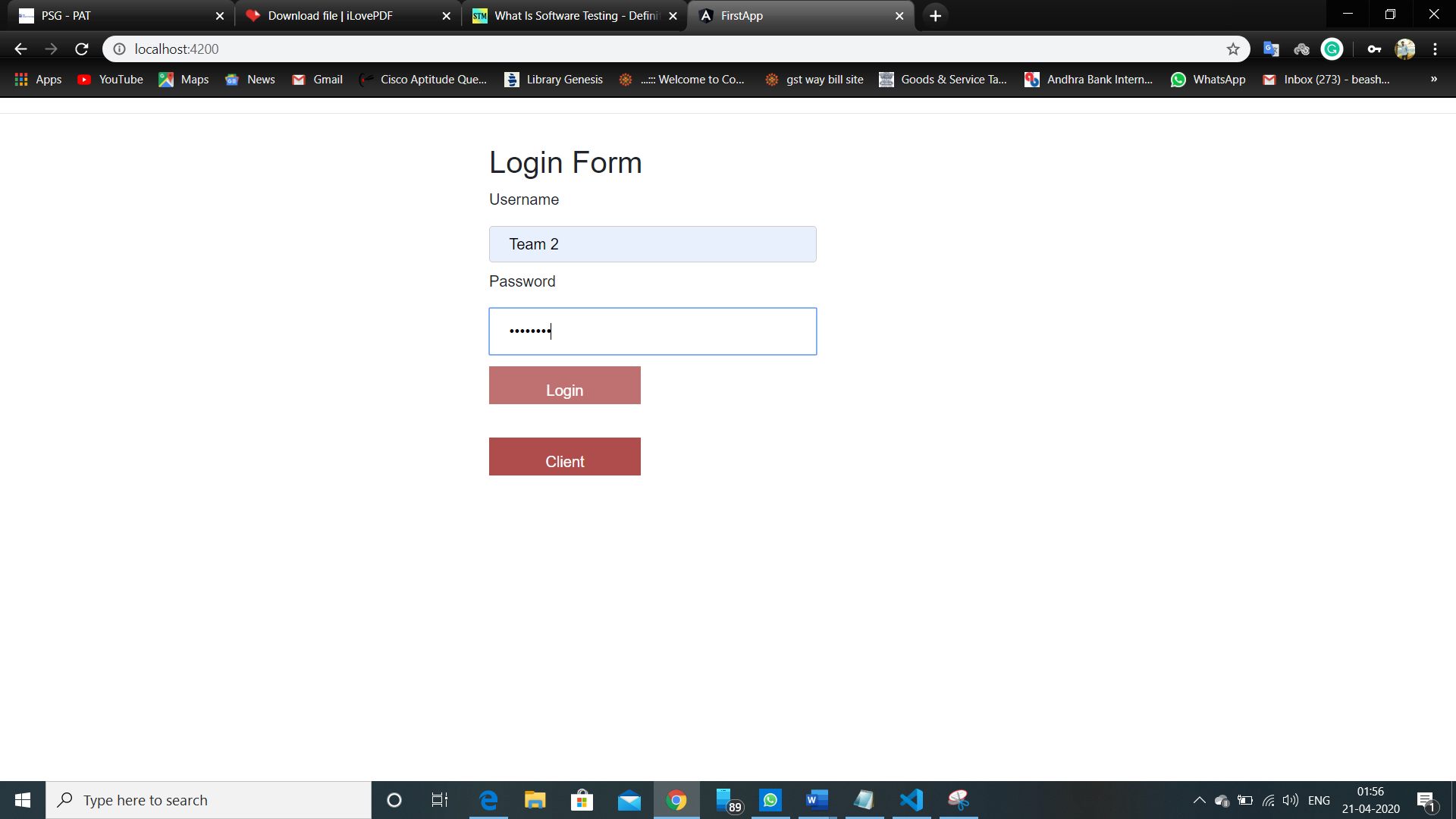
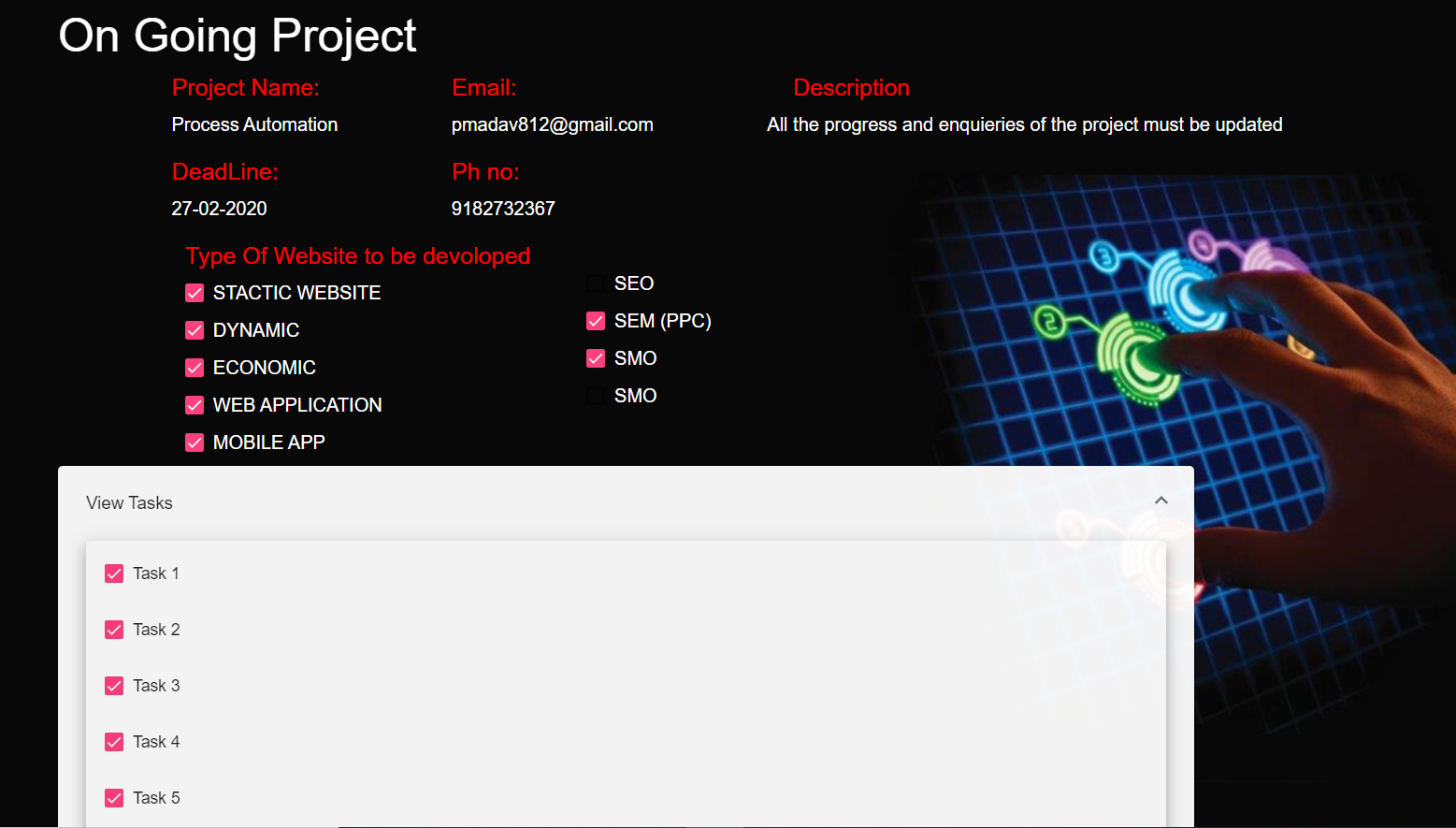
**CHAPTER 7**

**RESULT**

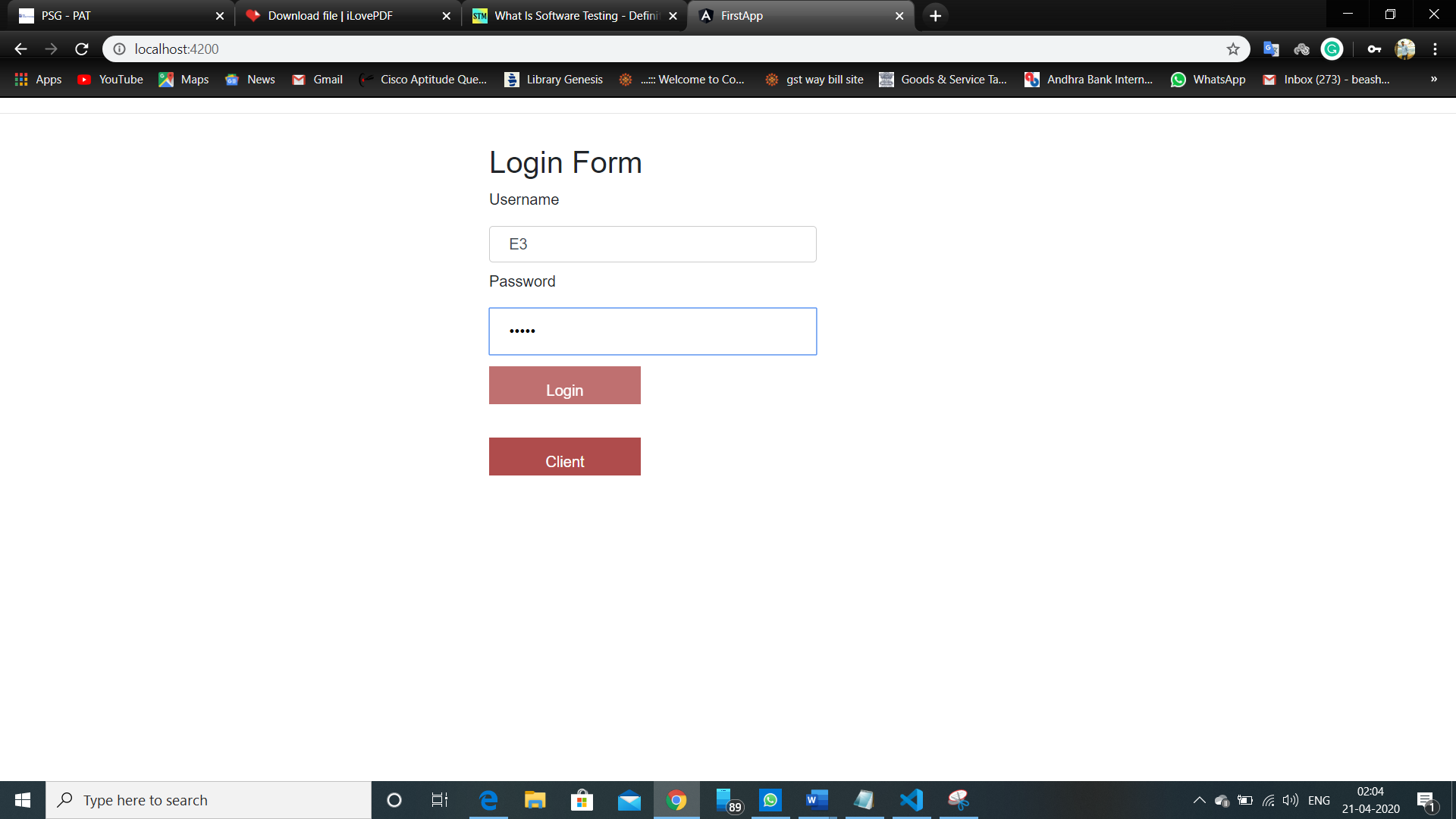
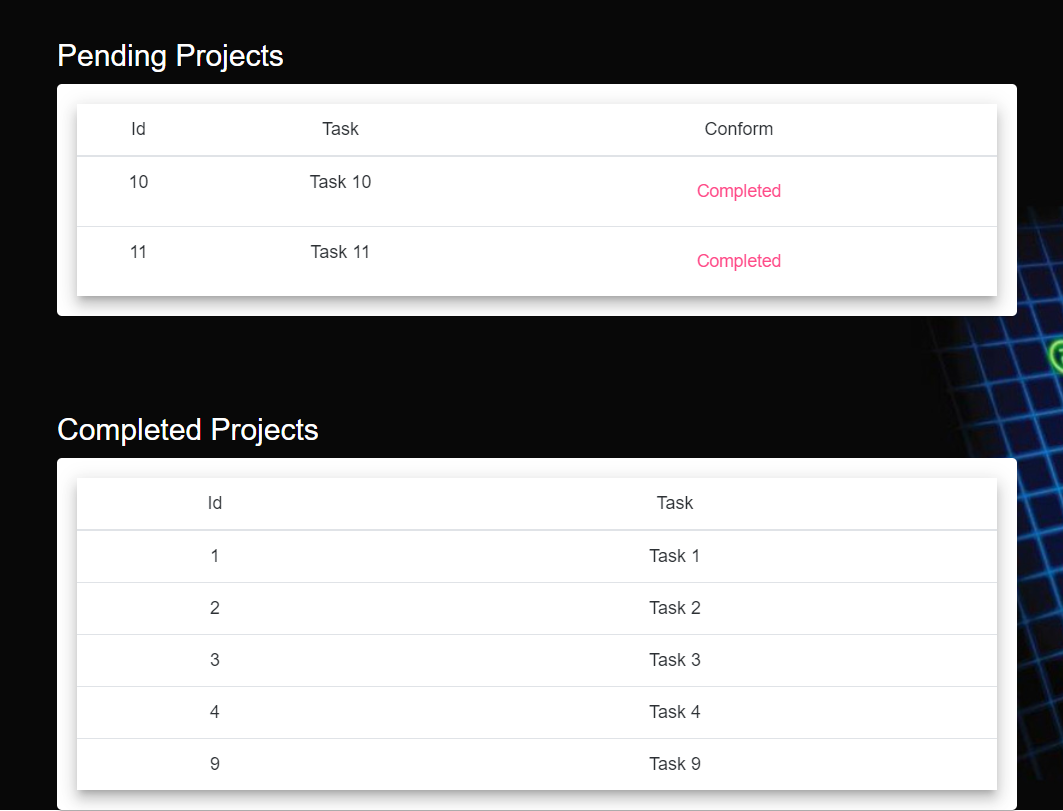
**7.1 Check with valid Manager’s Username and password**

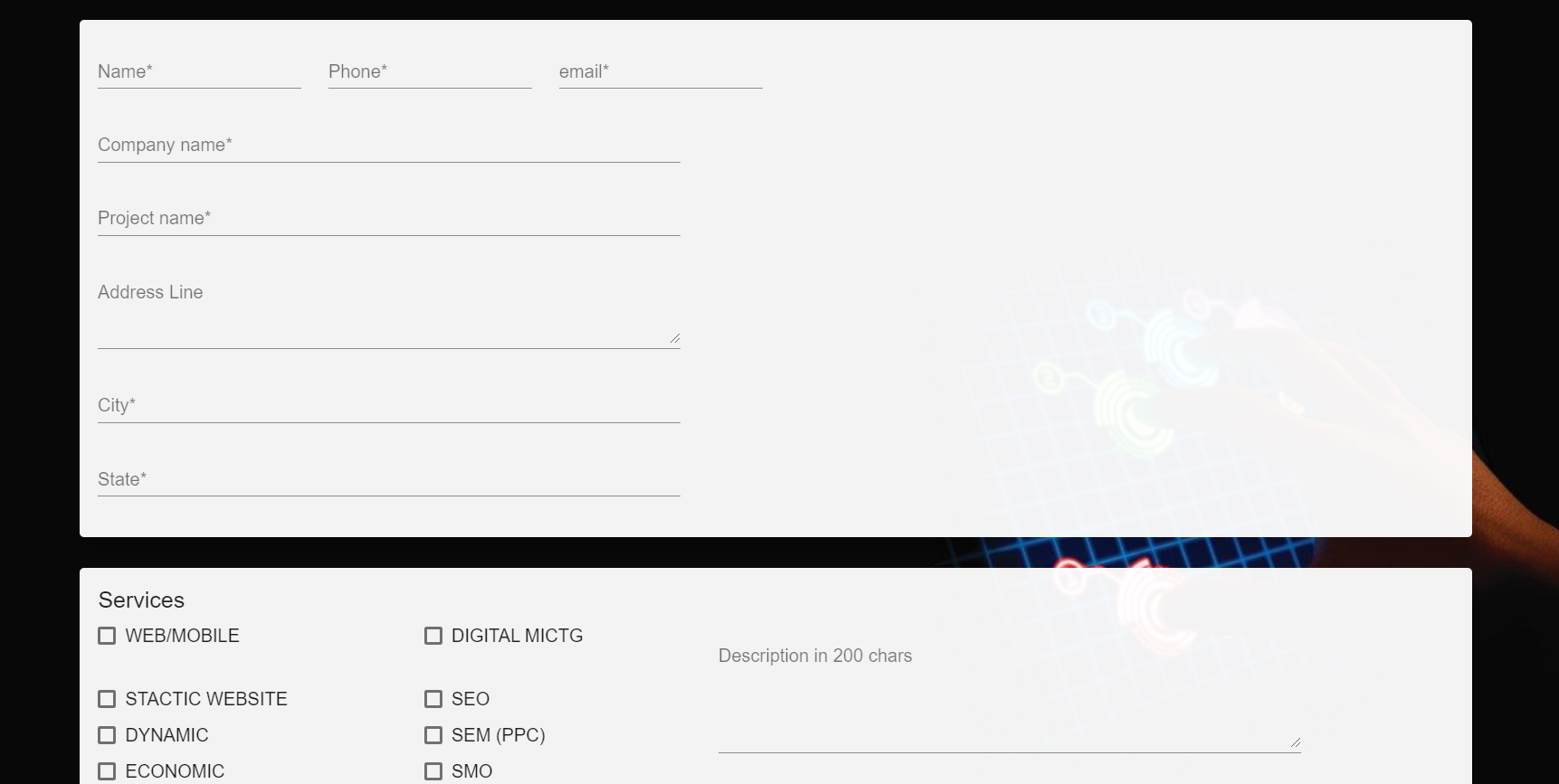
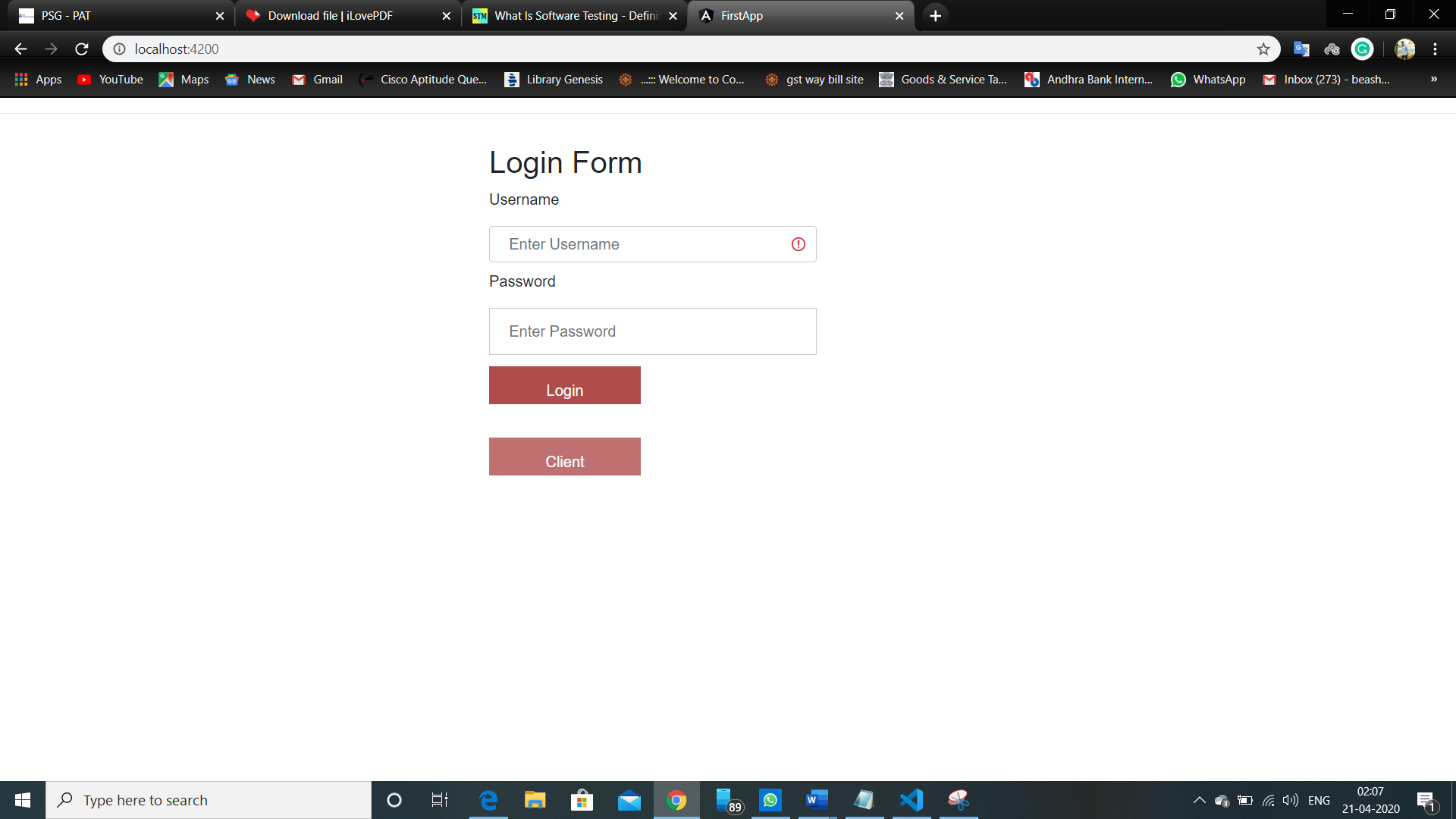
**7.2 Check with valid Team Leader’s Username and password**

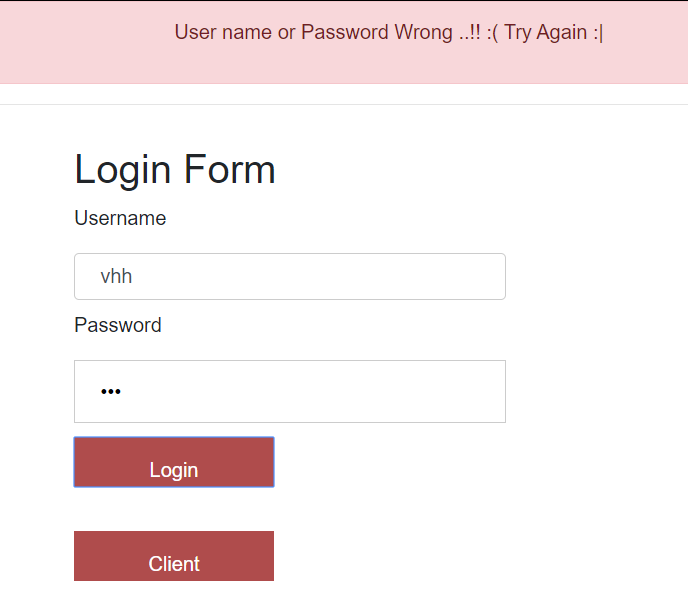
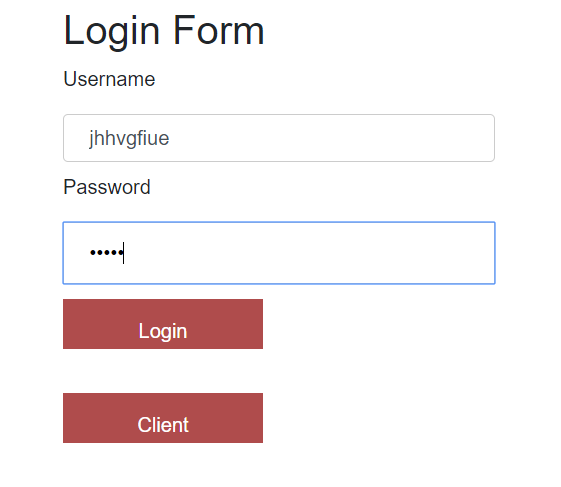
**7.3 Check with valid Employee Username and password**

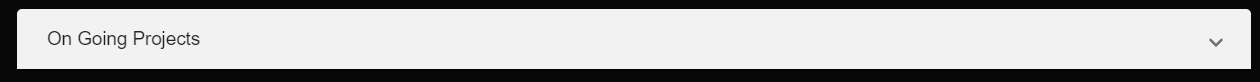
**7.4 Check with clicking on client**

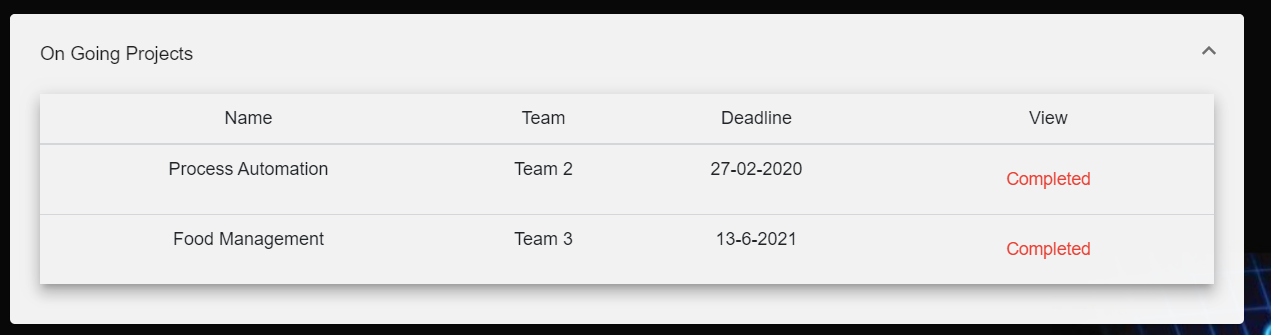


**7.5 Check with invalid username/ password**

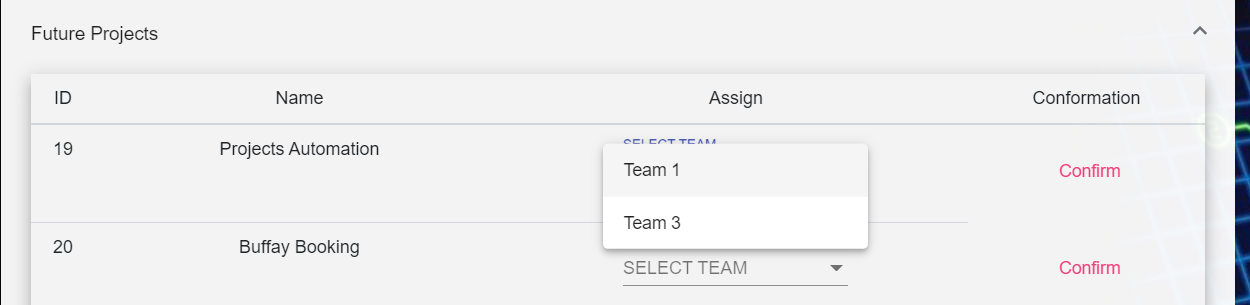


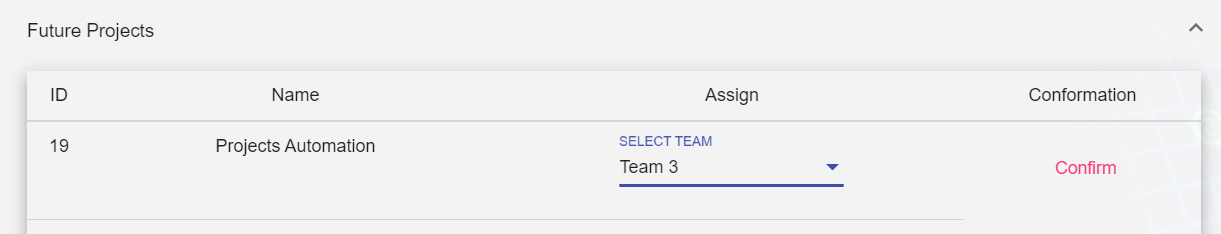
**7.6 Check for working of expansion panels**



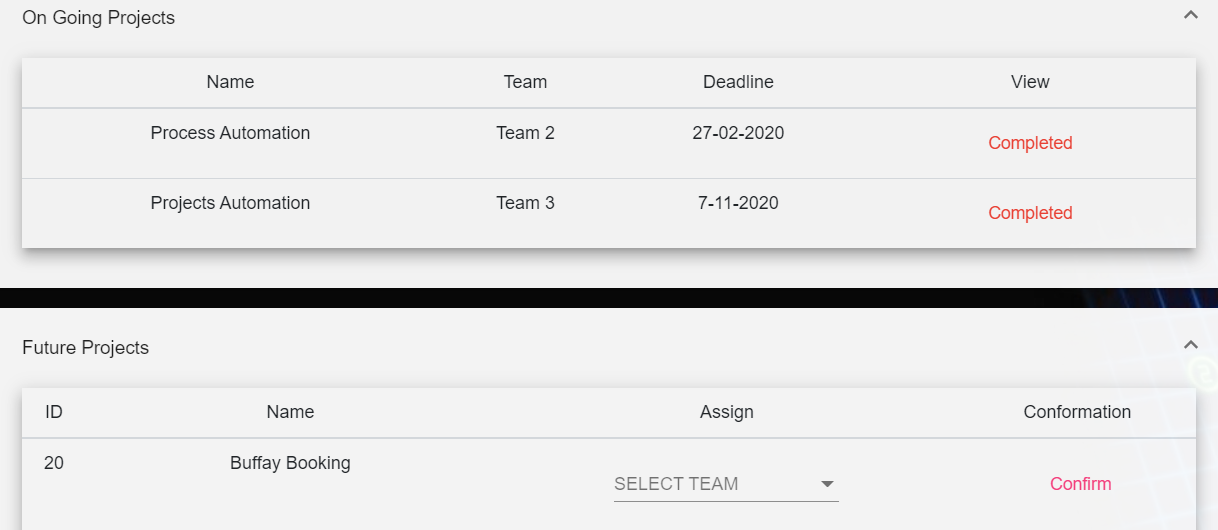


**7.7 Check for assignment of projects to team leaders**





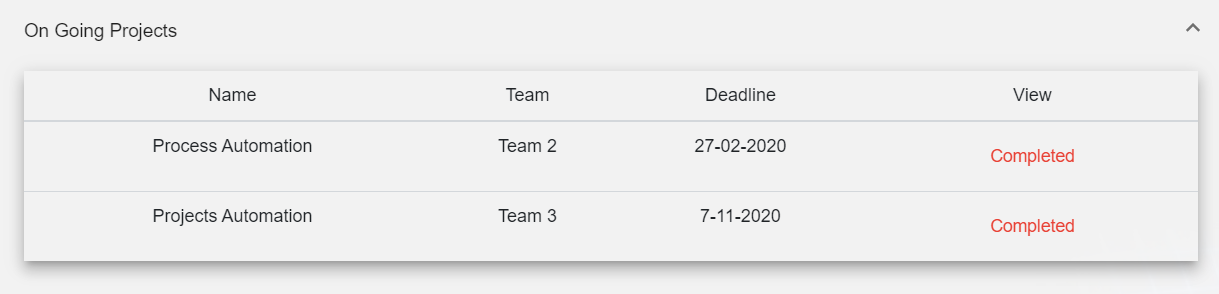
After clicking confirm



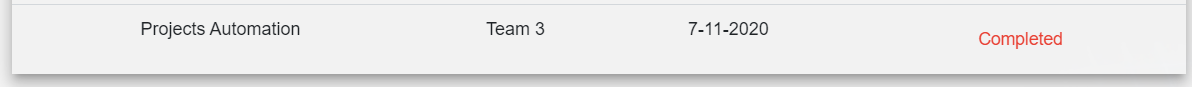
**7.8 Check for “save changes” button**



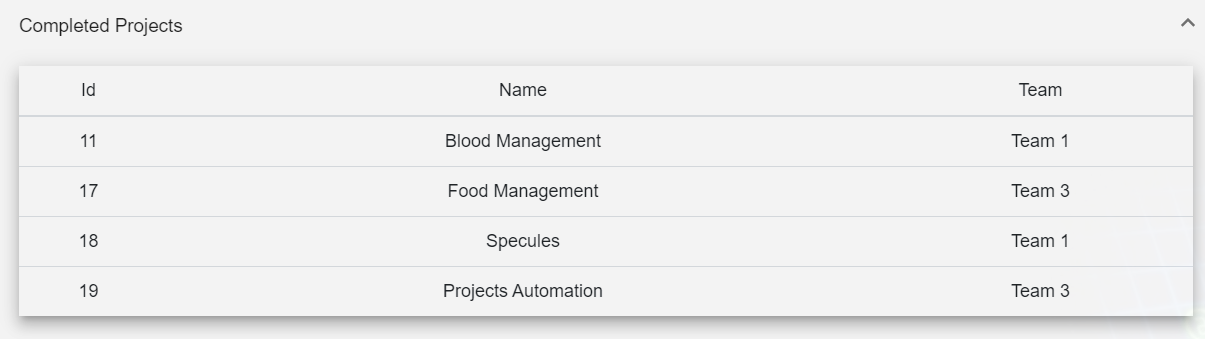
After clicking Save Changes and refreshing the page and logging again



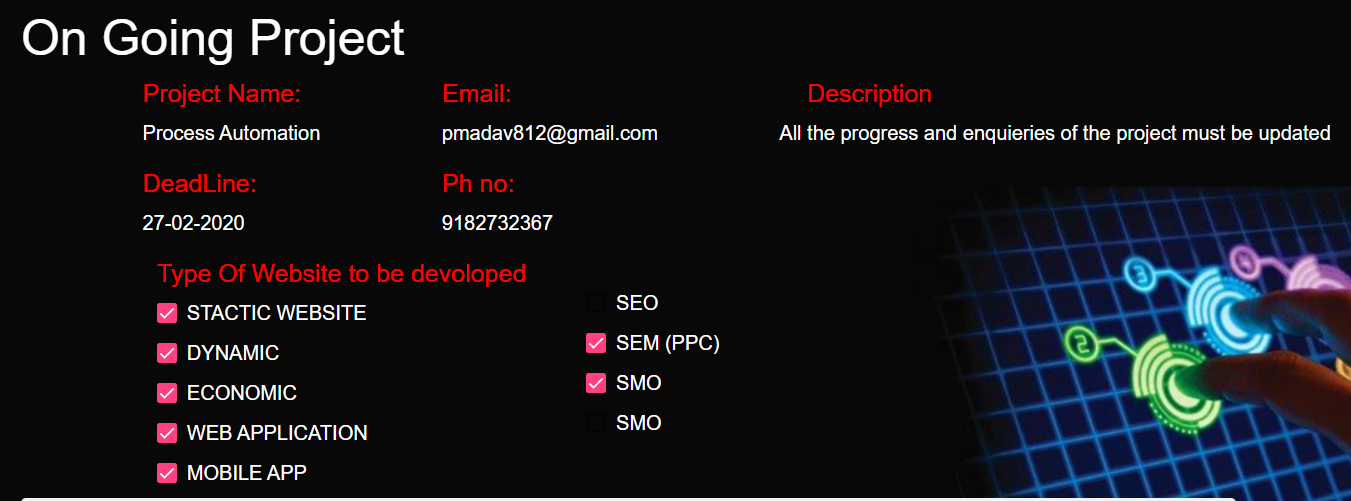
**7.9 Checking on “complete” button in ongoing projects**



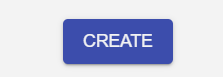
Complete button is clicked



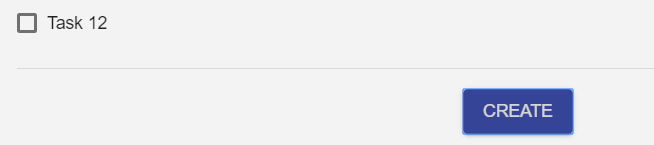
**7.10 Check for getting the project details**



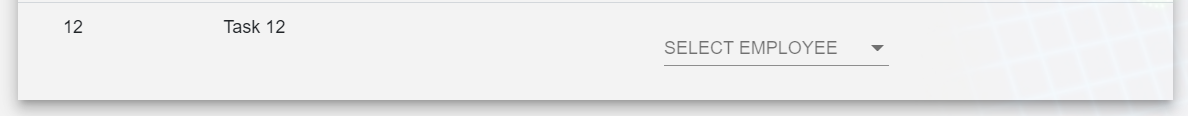
**7.11 Check for able to create tasks/Modules**



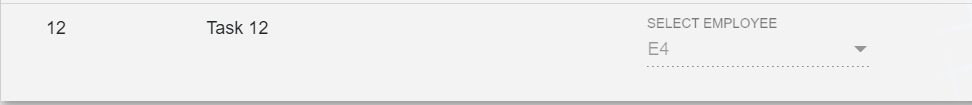
After clicking ADD



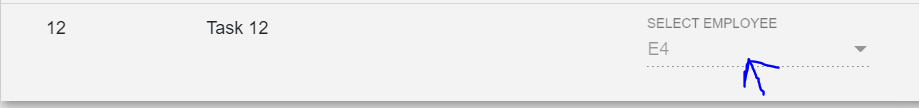
**7.12 Check whether the task added to the assignment section when task is created**



**7.13 Check for able to assign employee to the module/task**



**7.14 Check for able to edit the employee assigned to employee**



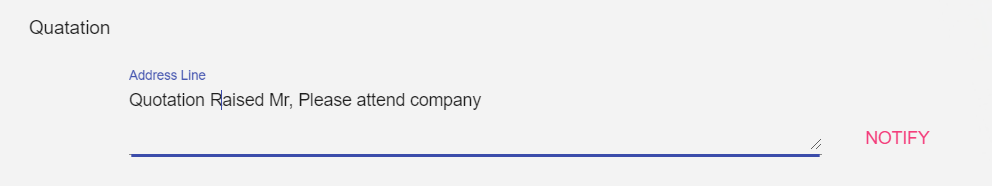
Un editable because it is disabled

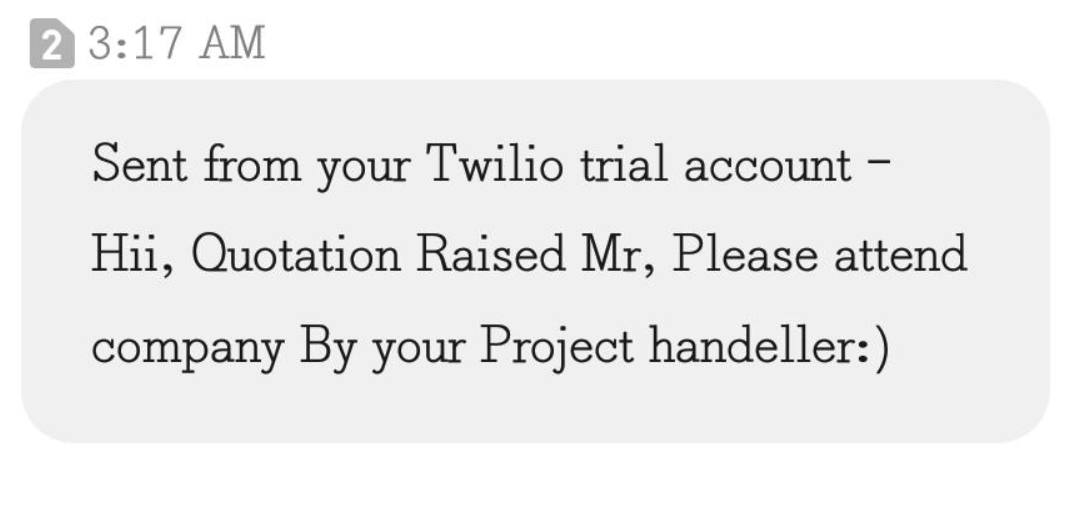
**7.15 Check for able that respective task that assigned is updated onto the employee dB or dashboard**



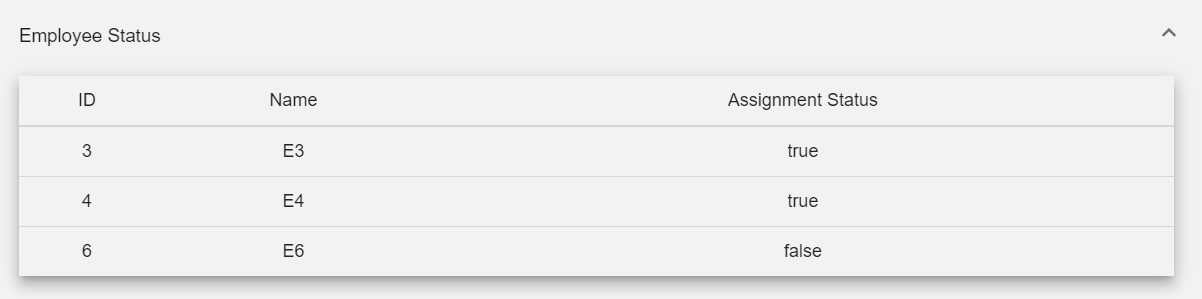
**7.16 Check for notifying quotation via Email or SMS**

In this project client asked to notify via SMS

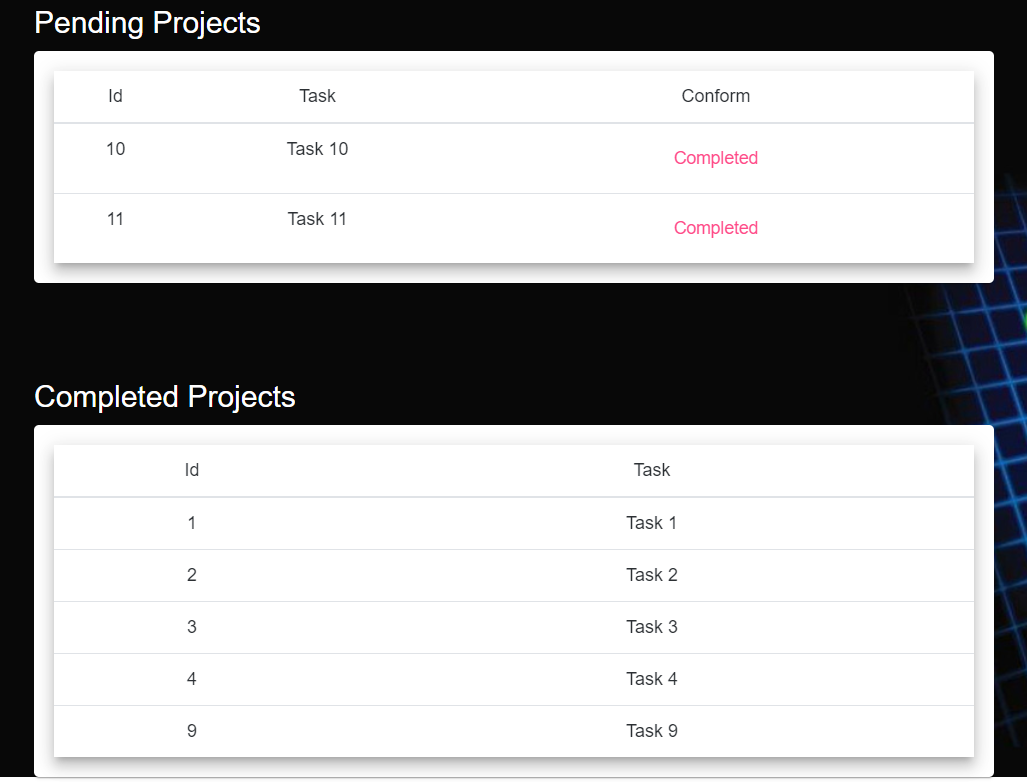




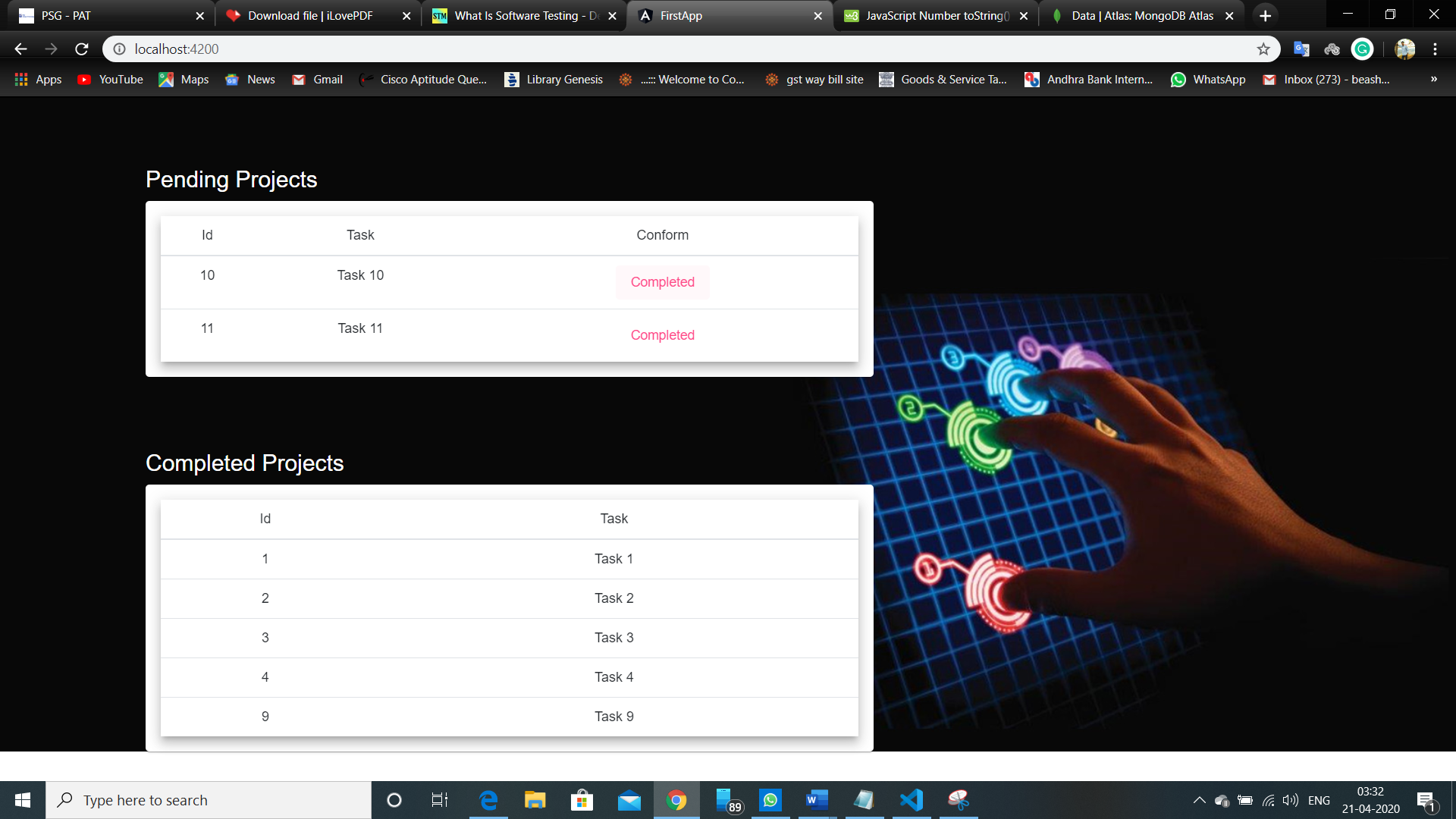
**7.17 Check for able to view the status of the employee (free or already assigned)**



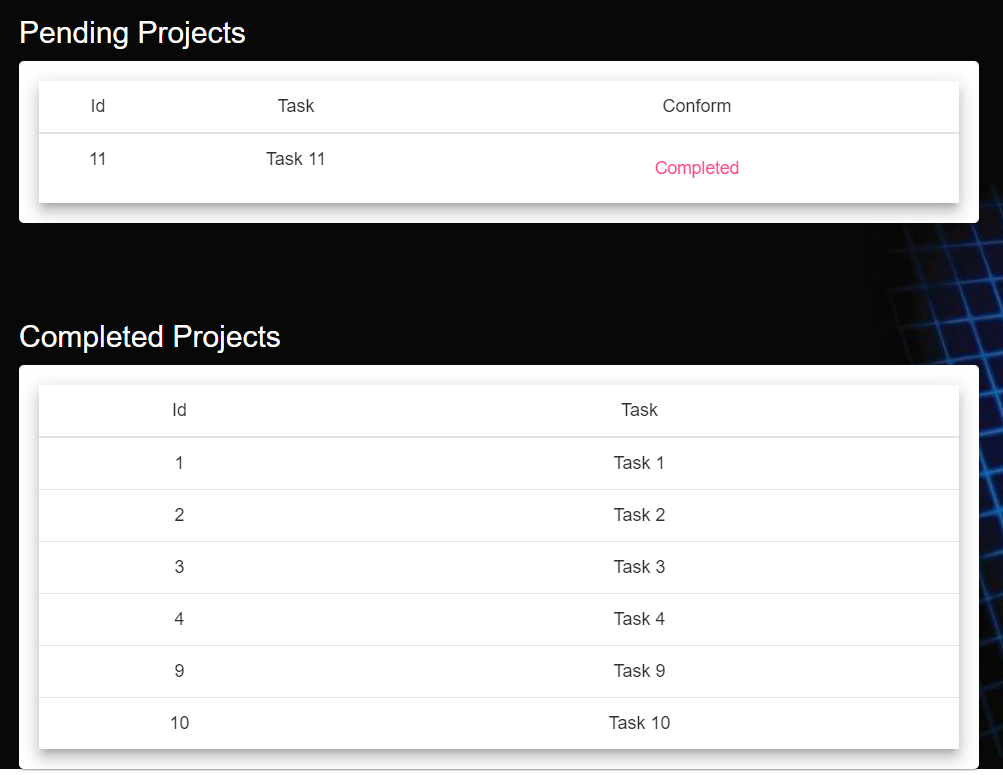
**7.18 Check for displaying tasks**



**7.19 Check for clicking the “completed” button**



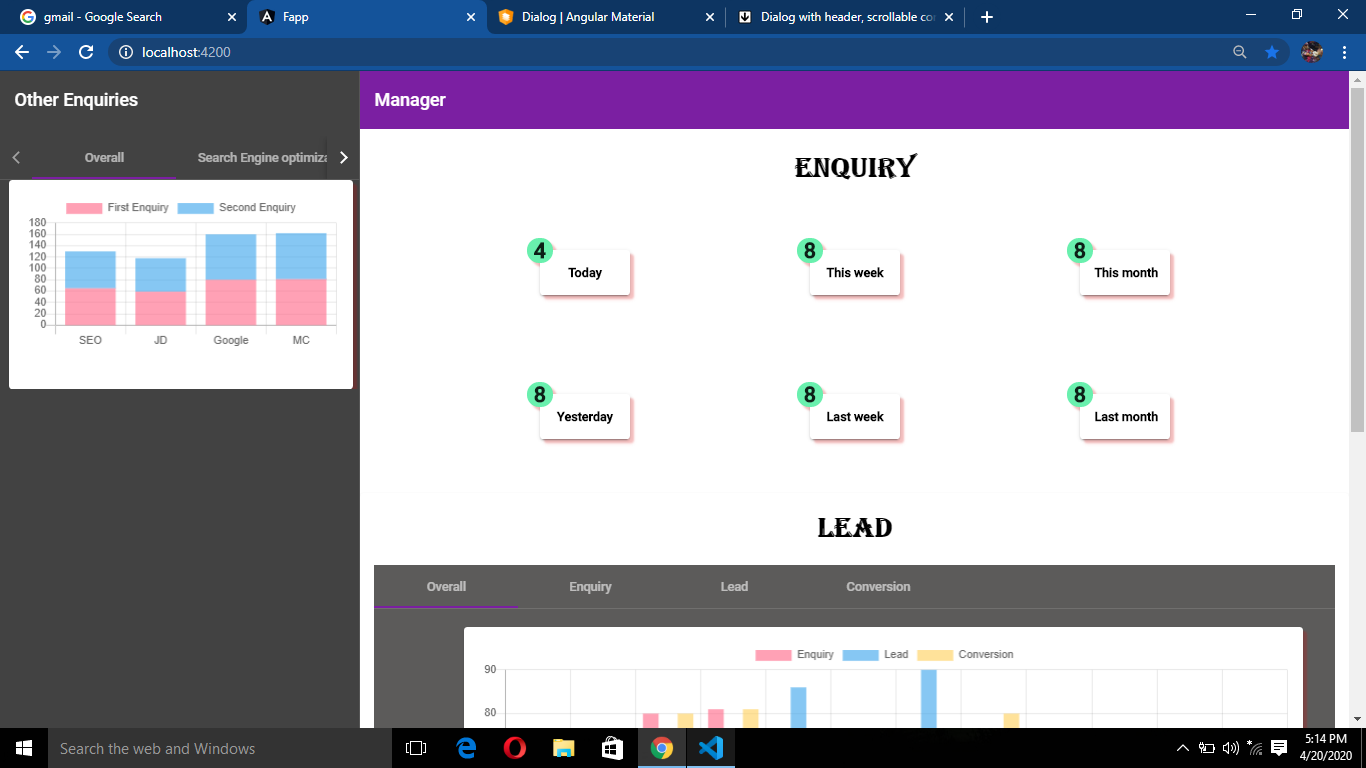
After clicking completed



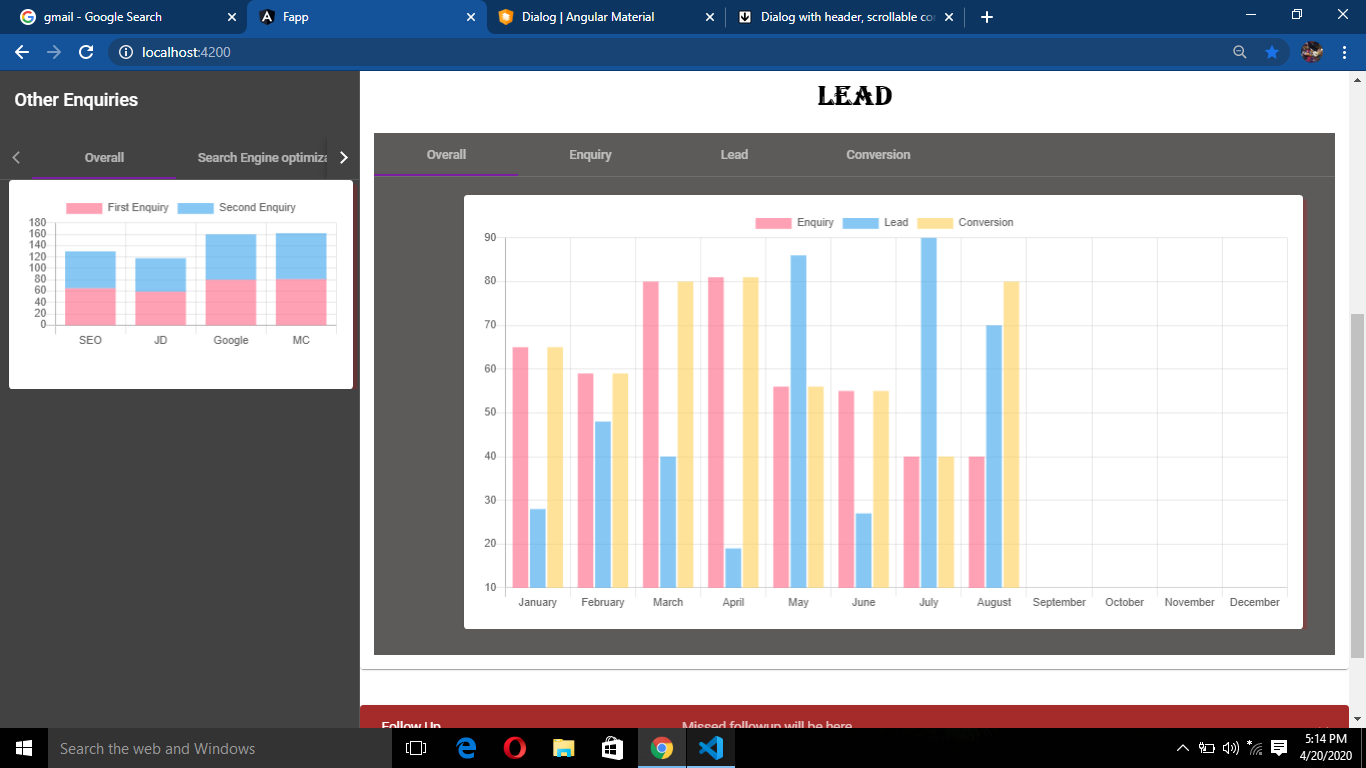
**7.20 Check weather task completed updated**



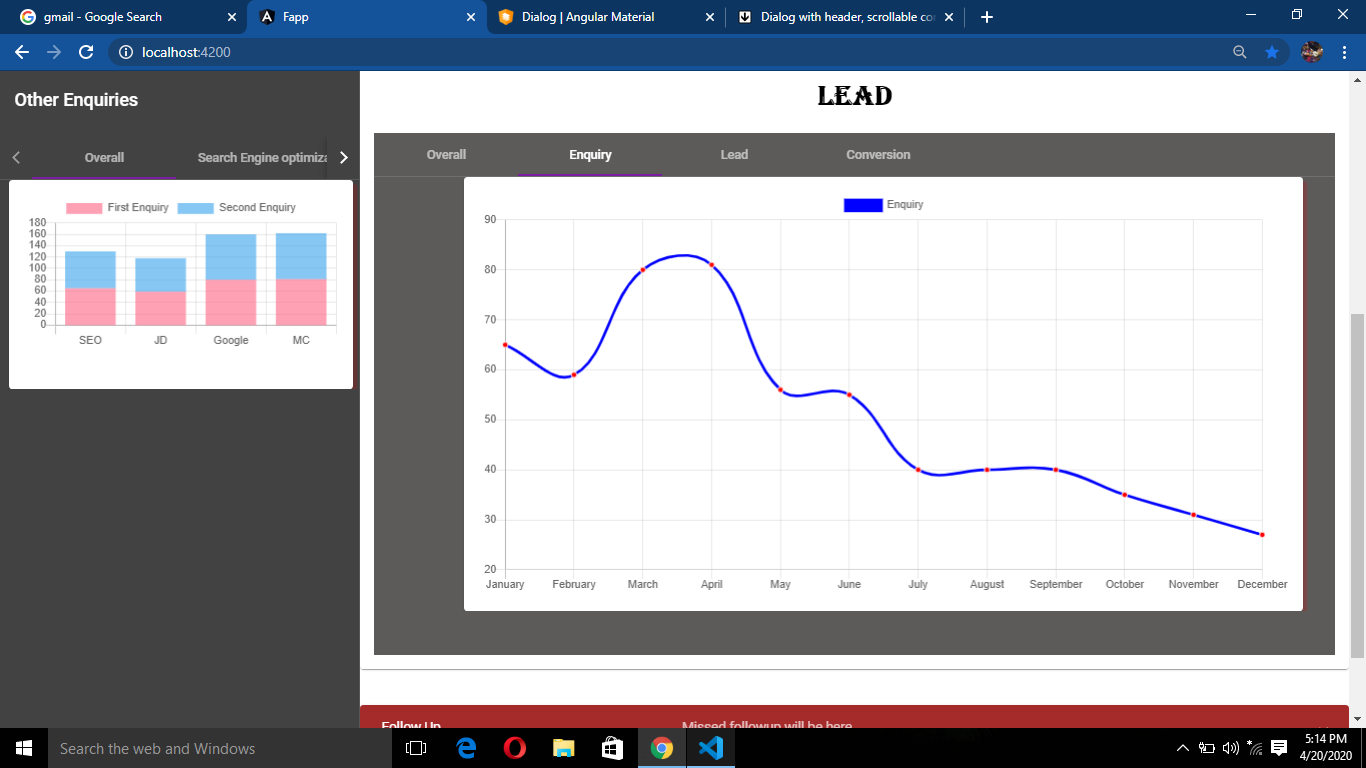
**7.21 Overall view of Manager dashboard:**

****

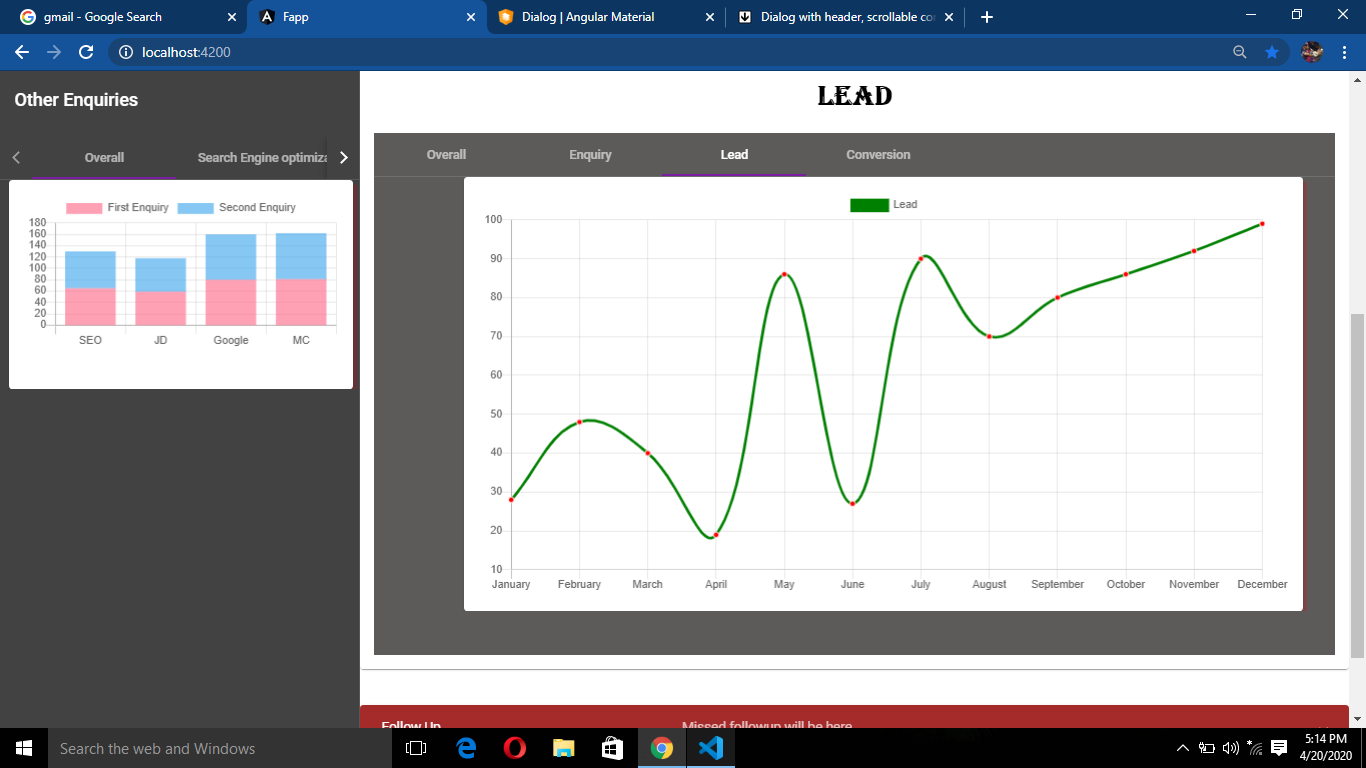
**7.22 Graphical representation of Enquiry, Lead, Conversion:**

****

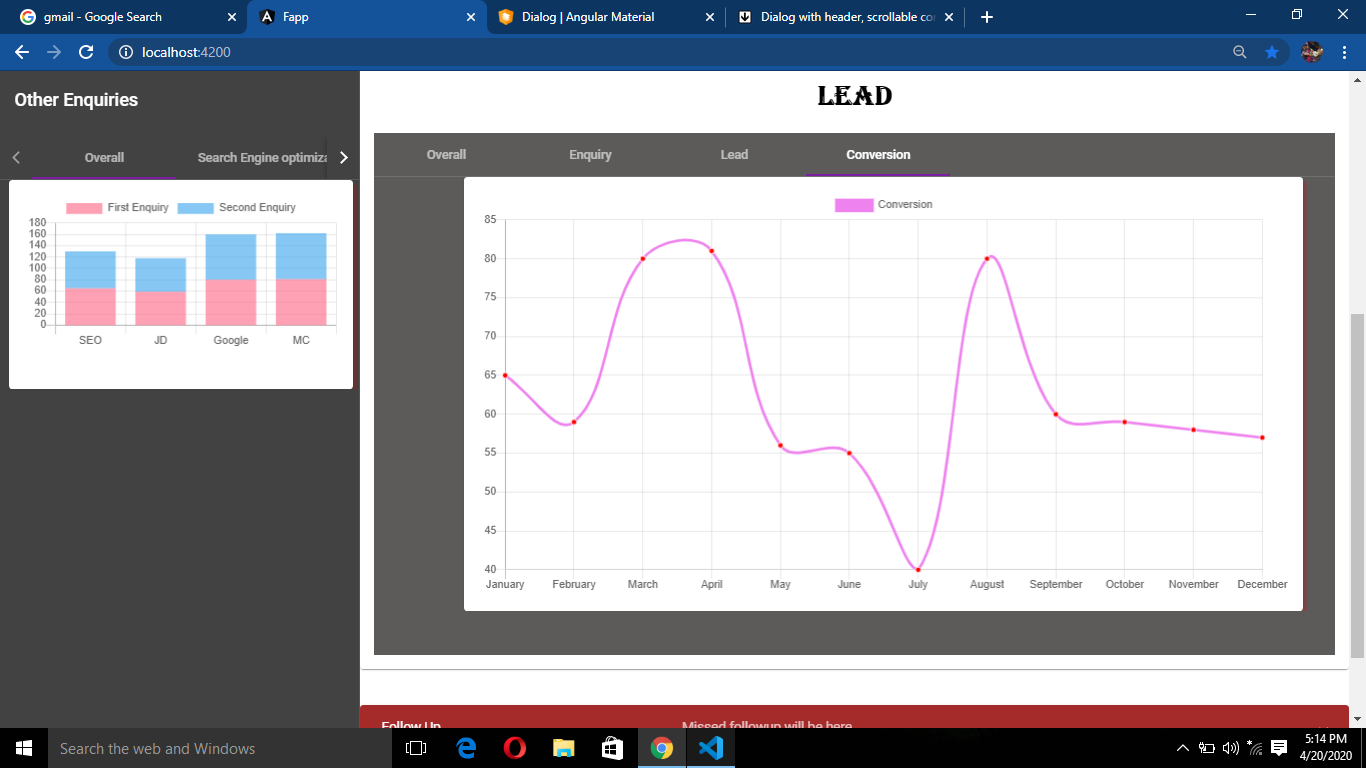
**7.23 Linear Graph of enquiry with predicted data:**

****

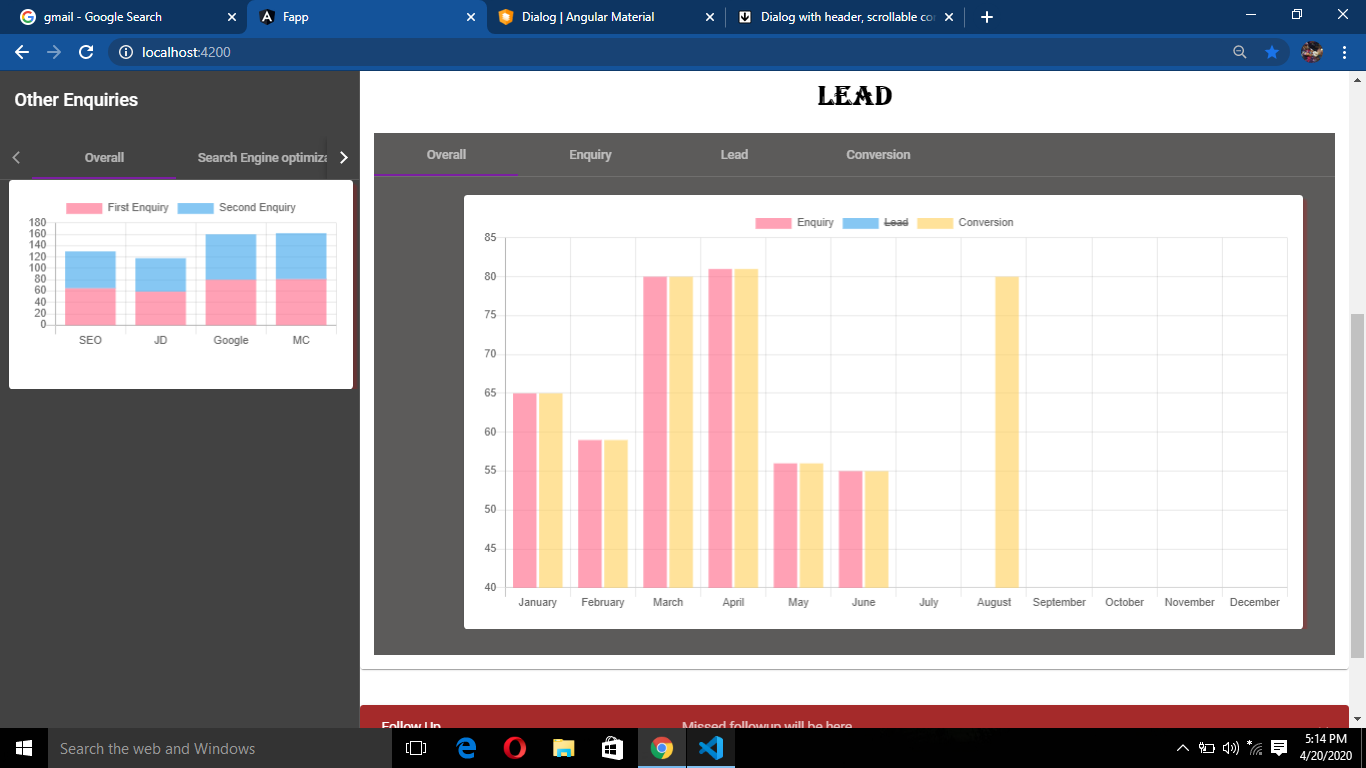
**7.24 Linear Graph of lead with predicted data:**

****

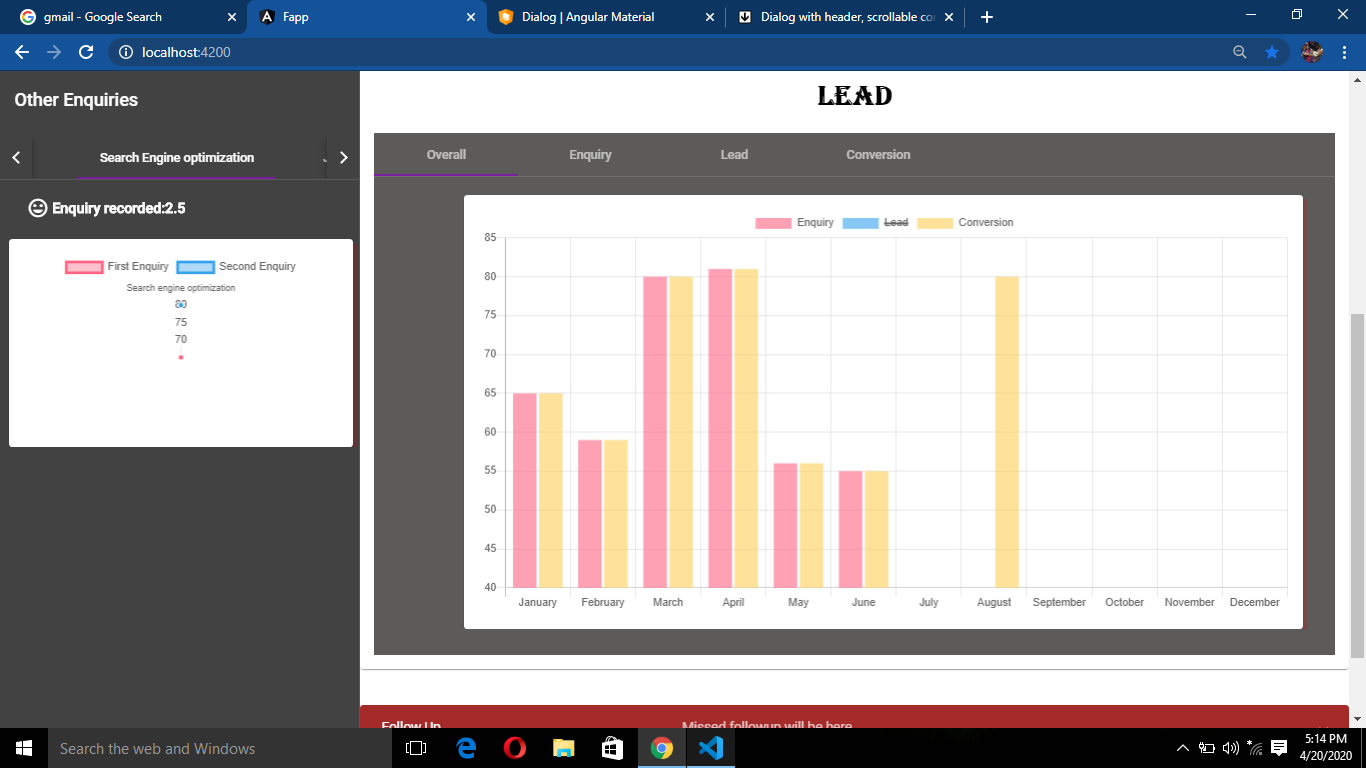
**7.25 Linear Graph of conversion with predicted data:**

****

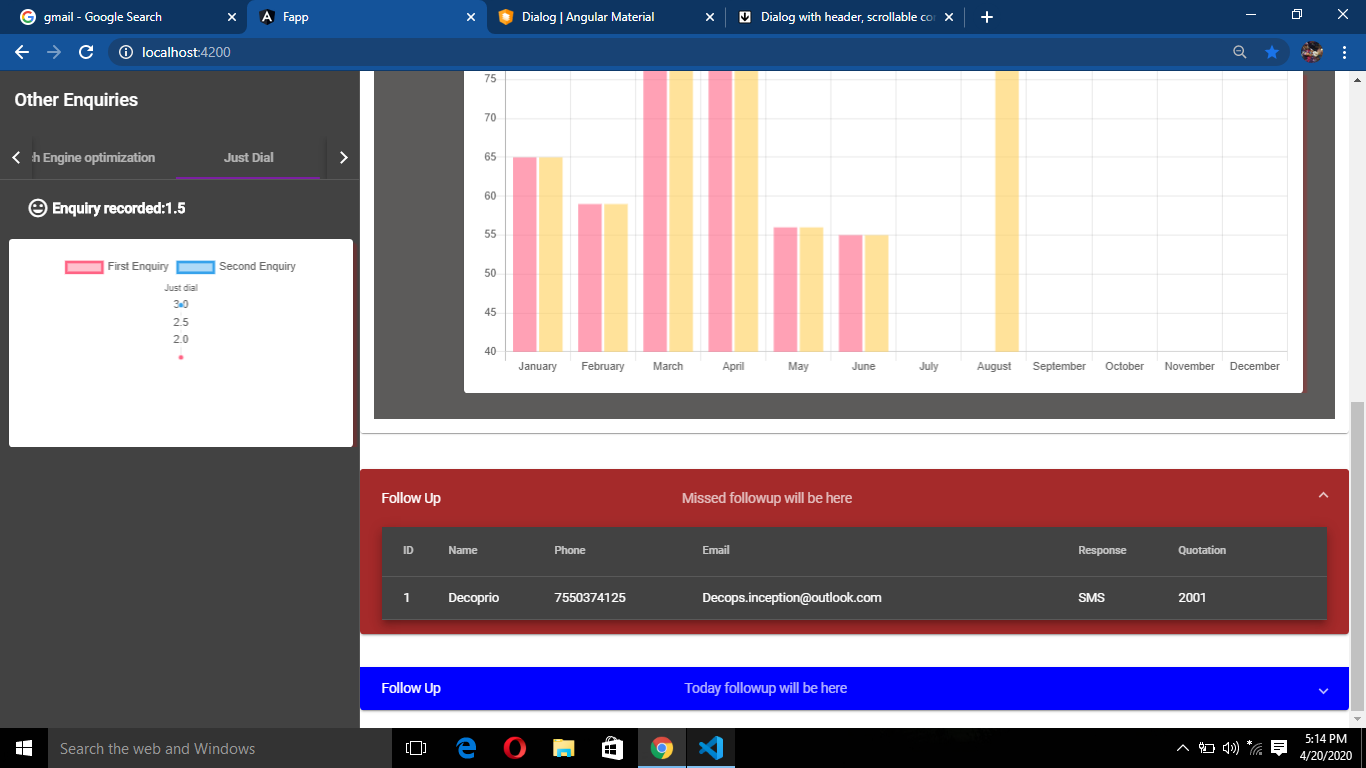
**7.26 Graph after filtering lead:**

****

**7.27 Enquiry chart for search engine optimization in side navigation:**

****

**7.28 Client details of today followup and missed follow up:**

****

**CHAPTER 8**

**CONCLUSION**

Finally, our team have developed an interface for the clients and the company members including manager and team leaders. We have achieved this with the evolving technology MEAN stack. So, with the help of the interface we developed the clients, managers and team leaders can be connected and they communicate via this interface. In order to be more systematic way in communication, we have developed some set of instructions for each individual as mentioned by the company. And with the help of Angular.js we have created the front ends(interfaces) for clients, managers and team leaders. And Mongo dB has been used for storing the data and Express.js has been helpful in routing.

In future we can add online payment for our system where you can directly make the payments with the help of digital payment systems. And we can create a repository where all our company projects are deployed with data abstraction. And every one can go through them. And if any of the clients requires any of the projects in the repository, they can buy them by following the instructions mentioned there.

**CHAPTER 9**

**BIBILOGRAPHY**

1. W3Schools. 2013. JavaScript JSON :

<https://www.w3schools.com/js/js_json.asp>

1. An Introduction to Mean Stack:

https://www.sitepoint.com/introduction-mean-stack/

1. Monteiro, F. 2014. Learning Single-page Web Application Development. Available at:

<https://www.packtpub.com/in/web-development/learning-single-page-web-application-development>

4. Holmes, S. 2015. Getting MEAN with Mongo, Express, Angular and Node.

[www.manning.com/books/getting-mean-with-mongo-express-angular-and-node](http://www.manning.com/books/getting-mean-with-mongo-express-angular-and-node%20%20%20%20%20%20%20%20%20%20%20%20%20)

5. MongoDB, Inc. 2008. Documents

<https://docs.mongodb.com/manual/core/document/>

**CHAPTER 10**

**APPENDIX**

**10.1 App Module**

**10.1.1 app.component.html**

<div \*ngIf="vis1">

    <!DOCTYPE html >

    <html >

    <head>

    <meta name="viewport" content="width=device-width, initial-scale=1">

    </head>

    <body>

      <div \*ngIf="alert" class="alert alert-danger" role="alert">

        <p style="text-align: center;">

          User name or Password Wrong ..!! :(    Try Again :|

        </p>

      </div>

    <form action="app.component.ts" #userForm="ngForm"  (ngSubmit)="onSubmit()"   novalidate  >

      <hr />

      <div class="container" >

        <h2 >Login Form</h2>

        <label for="uname"><b>Username</b></label>

        <input type="text" placeholder="Enter Username"  #name="ngModel" name="username" required [(ngModel)]="userModel.name" id="u1" [class.is-invalid]="name.invalid" class="form-control"  >

<label for="psw"><b>Password</b></label>

        <input type="password" placeholder="Enter Password" name="Password" #password="ngModel"  required [(ngModel)]="userModel.password" id="p1" class="form-group " >

        <button type="submit"   [disabled]="userForm.form.invalid"   >Login</button><br><br>

        <button (click)="clientOpen()"  >Client</button>

      </div>

    </form>

    </body>

    </html>

</div>

<cli-ent \*ngIf="clie"></cli-ent>

<man-ager \*ngIf="mana"></man-ager>

<team-leader \*ngIf="tea"></team-leader>

<employee \*ngIf="empl"></employee>

**10.1.2 app.component.ts**

userModel = new User("","");

  onSubmit() {

    this.http.get("http://localhost:3000/get-person",{params: {searchKey: this.userModel.name}}).

            subscribe(responce => {

                console.log(responce)

                let type = Object.values(responce)[0][0].type;

                if(type == 'manager') {

                  this.mana = true;

                  this.vis1= false;

                  this.alert = false;

                }

                else if(type == 'teamleader'){

                  this.tea = true;

                  this.vis1= false;

                  this.alert = false;

                }

                else if(type == 'employee'){

                  this.empl = true;

                  this.vis1= false;

                  this.alert = false;

                }

                else {

                  this.alert = true;

                }

            });

  };

  clientOpen() {

    this.clie = true;

    this.alert = false;

    this.vis1= false;

  }

}

**10.2 Client Component**

**10.2.1 client.component.html**

<form (submit)="onSave()">

    <mat-card class="shadow" id="one">

      <mat-form-field>

        <input matInput type="text" placeholder="Name\*" name="name" [(ngModel)]="name">

      </mat-form-field>

      <mat-form-field style="margin-left: 2%;">

        <input matInput type="text" placeholder="Phone\*" name="phno" [(ngModel)]="phno">

      </mat-form-field>

      <mat-form-field style="margin-left: 2%;">

        <input matInput type="email" placeholder="email\*" name="email" [(ngModel)]="email">

      </mat-form-field>

      <br>

      <mat-form-field class="full">

        <input matInput placeholder="Company name\*" name="cname" [(ngModel)]="cname">

      </mat-form-field><br>

      <mat-form-field class="full">

        <input matInput placeholder="Project name\*" name="pname" [(ngModel)]="pname">

      </mat-form-field><br>

      <mat-form-field class="full">

        <textarea matInput placeholder="Address Line" rows="3" name="address" [(ngModel)]="address"></textarea>

      </mat-form-field><br>

      <mat-form-field class="full">

        <input matInput placeholder="City\*" name="city" [(ngModel)]="city">

      </mat-form-field><br>

      <mat-form-field class="full">

        <input matInput placeholder="State\*" name="state" [(ngModel)]="state">

      </mat-form-field><br>

    </mat-card>

    <mat-card style="margin-top: 2%;height: 270px;" class="shadow">

      <h5>Services</h5>

      <div class="left">

        <mat-checkbox name="web" [(ngModel)]="web" [(indeterminate)]="indeterminate" (change)="func1()">WEB/MOBILE</mat-checkbox><br><br>

        <mat-checkbox name="static" [(ngModel)]="static" (change)="func()">STACTIC WEBSITE</mat-checkbox><br>

        <mat-checkbox name="dynamic" [(ngModel)]="dynamic" (change)="func()">DYNAMIC</mat-checkbox><br>

        <mat-checkbox name="economic" [(ngModel)]="economic" (change)="func()">ECONOMIC</mat-checkbox><br>

        <mat-checkbox name="appln" [(ngModel)]="appln" (change)="func()">WEB APPLICATION</mat-checkbox><br>

        <mat-checkbox name="mob" [(ngModel)]="mob" (change)="func()">MOBILE APP</mat-checkbox>

      </div>

      <div class="middle">

        <mat-checkbox name="digital" [(ngModel)]="digital" [(indeterminate)]="indeterminate1" (change)="check1()">DIGITAL MICTG</mat-checkbox><br><br>

        <mat-checkbox name="seo" [(ngModel)]="seo" (change)="check()">SEO</mat-checkbox><br>

        <mat-checkbox name="sem" [(ngModel)]="sem" (change)="check()">SEM (PPC)</mat-checkbox><br>

        <mat-checkbox name="smo" [(ngModel)]="smo" (change)="check()">SMO</mat-checkbox><br>

        <mat-checkbox name="smm" [(ngModel)]="smm" (change)="check()">SMM</mat-checkbox><br>

      </div>

      <div>

        <mat-form-field class="full right">

          <textarea matInput placeholder="Description in 200 chars" rows="5" name="description" [(ngModel)]="description"></textarea>

        </mat-form-field>

      </div>

    </mat-card>

    <mat-card style="margin-top: 2%;height: 200px;" class="shadow">

      <h5>ENQUIRY DETAILS</h5>

      <mat-form-field>

        <mat-label>LEAD SOURCE</mat-label>

        <mat-select name="selectoption1" [(ngModel)]="selectoption1">

          <mat-option \*ngFor= "let emp of select1" [value] = "emp.id">{{emp.name}}</mat-option>

        </mat-select>

      </mat-form-field>

      <br>

      <mat-form-field>

        <mat-label>HANDLED BY</mat-label>

        <mat-select name="selectoption2" [(ngModel)]="selectoption2">

          <mat-option \*ngFor= "let emp of select2" [value] = "emp.id">{{emp.name}}</mat-option>

        </mat-select>

      </mat-form-field>

    </mat-card>

    <mat-card style="margin-top: 2%;height: 120px;" class="shadow">

      <h5>FOLLOW UP</h5>

      <mat-form-field>

        <mat-label>QUOTE</mat-label>

        <mat-select name="selectoption3" [(ngModel)]="selectoption3">

          <mat-option \*ngFor= "let emp of select3" [value] = "emp.id">{{emp.name}}</mat-option>

        </mat-select>

      </mat-form-field>

      <mat-form-field style="margin-left: 2%;">

        <mat-label>Choose a date</mat-label>

        <input matInput [matDatepicker]="picker" name="date" [(ngModel)]="mydate">

        <mat-datepicker-toggle matSuffix [for]="picker"></mat-datepicker-toggle>

        <mat-datepicker #picker></mat-datepicker>

      </mat-form-field>

      <mat-form-field style="margin-left: 2%;">

        <mat-label>RESPONCIVE</mat-label>

        <mat-select name="selectoption4" [(ngModel)]="selectoption4">

          <mat-option \*ngFor= "let emp of select4" [value] = "emp.id">{{emp.name}}</mat-option>

        </mat-select>

      </mat-form-field><br>

    </mat-card>

    <mat-card style="margin-top: 2%;margin-bottom: 2%;height: 90px;" class="shadow">

      <div>

        <mat-checkbox name="ncustomer" [(ngModel)]="ncustomer">Notify Customer</mat-checkbox>

        <mat-checkbox name="nemail" [(ngModel)]="nemail" class="sp" [disabled]="!ncustomer" (click)="cc1()">EMAIL</mat-checkbox>

        <mat-checkbox name="nsms" [(ngModel)]="nsms" class="sp" [disabled]="!ncustomer" (click)="cc2()">SMS</mat-checkbox><br>

      </div>

    </mat-card>

    <button class="decorateButton" type="submit" mat-raised-button color="primary">SAVE</button>

  </form>

**10.2.2 client.component.ts**

onSave(){

        this.http.get("http://localhost:3000/get-count").subscribe((responce) => {

            console.log(responce);

            this.id = Object.values(responce)[0]+1;

            console.log(this.id);

            let checkbox1 = { static: this.static, dynamic: this.dynamic, economic: this.economic, appln: this.appln, mob: this.mob };

            let checkbox2 = { seo: this.seo, sem: this.sem, smo: this.smo, smm: this.smm };

            let select = {

select1: this.select1[this.selectoption1-1]['name'],

      select2: this.select2[this.selectoption2-1]['name'],

      select3: this.select3[this.selectoption3-1]['name'],

      select4: this.select4[this.selectoption4-1]['name']};

      let notify\_customer = { customer: this.ncustomer, email: this.nemail, sms: this.nsms };

      let notify\_mgmt = { mgmt: this.nmgmt, email: this.nmemail, sms: this.nmsms };

            var dd = (this.mydate.getDate()).toString()+"-"+(this.mydate.getMonth()).toString()+"-"+(this.mydate.getFullYear()).toString();

            const info = {id: this.id,

                          name: this.name,

                          phno: this.phno,

                          email: this.email,

                          cname: this.cname,

                          pname: this.pname,

                          address: this.address,

                          city: this.city,

                          state: this.state,

                          description: this.description,

                          checkbox1: checkbox1,

                          checkbox2: checkbox2,

                          select: select,

                          notifyCustomer: notify\_customer,

                          notifyMgmt: notify\_mgmt,

                          assignment: false,

                          date: dd

                        };

            console.log(info);

            this.name = "";

            this.phno = "";

            this.email = "";

            this.cname = "";

            this.pname = "";

            this.address = "";

            this.city = "";

            this.state = "";

            this.description = "";

            this.web = false;

            this.static = false;

            this.dynamic = false;

            this.economic = false;

            this.appln = false;

            this.mob = false;

            this.indeterminate = false;

            this.indeterminate1 = false;

            this.digital = false;

            this.seo = false;

            this.sem = false;

            this.smo = false;

            this.smm = false;

            this.ncustomer = false;

            this.nemail = false;

            this.nsms = false;

            this.http.post("http://localhost:3000/client-form",info).

            subscribe(responce => {

                console.log(responce)

            });

        });

    }

**10.3 Manager Component**

**10.3.1 manager.component.html**

<mat-accordion [multi]="true" style="margin-left: 2%;width: 45%">

    <mat-expansion-panel class="ex1" [expanded]="true" style="opacity: 0.95;margin-left: 2%;width: 65%">

      <mat-expansion-panel-header>

        <mat-panel-title>

          On Going Projects

        </mat-panel-title>

      </mat-expansion-panel-header>

      <table mdbTable small="true" class="mat-elevation-z8" style="width: 100%">

        <thead>

          <tr style="text-align: center;">

            <th \*ngFor="let head of ds1" scope="col">{{head}} </th>

          </tr>

        </thead>

        <tbody>

          <tr \*ngFor="let elem of onGoingArray" style="width: 100%">

            <td style="text-align: center;">{{ elem.name }}</td>

            <td style="text-align: center;">{{ elem.team }}</td>

            <td style="text-align: center;">{{ elem.deadline }}</td>

            <td style="text-align: center;">

              <button mat-button color="warn" style="text-align: center;" (click)="completeProject(elem)">Completed</button>

            </td>

          </tr>

        </tbody>

      </table>

    </mat-expansion-panel>

    <mat-expansion-panel class="ex1" [expanded]="true" style="opacity: 0.95;margin-left: 2%;width: 65%">

      <mat-expansion-panel-header>

        <mat-panel-title>

          Future Projects

        </mat-panel-title>

      </mat-expansion-panel-header>

      <table  mdbTable small="true" class="mat-elevation-z8" style="width: 100%">

        <thead>

          <tr style="text-align: center;">

            <th \*ngFor="let head of ds2" scope="col">{{head}} </th>

          </tr>

        </thead>

        <tbody>

          <tr \*ngFor="let elem of futureArray">

            <td style="text-align: center;">{{ elem.id }}</td>

            <td style="text-align: center;">{{ elem.name }}</td>

            <td style="text-align: center;">

              <mat-form-field>

                <mat-label>SELECT TEAM</mat-label>

                <mat-select [(ngModel)]="elem.team" name="categorie" style="width: 50;">

                  <mat-option \*ngFor="let e of team" [value]="e.name">{{e.name}}</mat-option>

                </mat-select>

             </mat-form-field>

            </td>

            <button mat-button color="accent" style="margin-left: 30%;margin-top: 10%;" (click)="confirm(elem)">Confirm</button>

          </tr>

        </tbody>

      </table>

      <button mat-raised-button style="margin-left: 45%;" color="primary" (click)="saveProjects()">Save Changes</button>

    </mat-expansion-panel>

   <mat-expansion-panel class="ex1" [expanded]="true" style="opacity: 0.95;margin-left: 2%;width: 65%">

      <mat-expansion-panel-header>

        <mat-panel-title>

          Completed Projects

        </mat-panel-title>

      </mat-expansion-panel-header>

      <table  mdbTable small="true" class="mat-elevation-z8" style="width: 100%">

        <thead>

          <tr style="text-align: center;">

            <th \*ngFor="let head of ds3" scope="col">{{head}} </th>

          </tr>

        </thead>

        <tbody>

          <tr \*ngFor="let elem of completeArray">

            <td style="text-align: center;">{{ elem.id }}</td>

            <td style="text-align: center;">{{ elem.name }}</td>

            <td style="text-align: center;">{{ elem.team }}</td>

          </tr>

        </tbody>

      </table>

    </mat-expansion-panel>

  </mat-accordion>

**10.3.2 manager.component.ts**

gettask()

    {

        return this.http.get("http://localhost:3000/get-projects").pipe(

          map((responce)=>{

              console.log(responce);

              this.onGoingArray = [];

              this.futureArray = [];

              console.log("Projects are")

              console.log(Object.values(responce)[0][0].projects);

              (Object.values(responce)[0][0].projects).forEach(element => {

                if(element.assignstatus == 1) {

                  this.onGoingArray.push(element);

                  this.tempteam.push(element.team); // Getting all alloted teams

                }

                else if(element.assignstatus == 0) {

                  this.futureArray.push({tempid: this.futureArray.length+1,assignstatus: element.assignstatus, deadline: element.deadline, id: element.id, name: element.name, team: element.team,

                    description: element.description,

                    checkbox1: element.checkbox1,

                    checkbox2: element.checkbox2,

                    nemail: element.nemail,

                    nsms: element.nsms});

                }

                else {

                  this.completeArray.push(element);

                  this.completedStatus = true;

                }

              });

              console.log("teams are");

              (Object.values(responce)[0][0].teams).forEach(element => {

                if(!this.tempteam.includes(element)){

                  this.team.push({name: element})

                }

              });

              console.log(this.team);

              console.log(this.onGoingArray);

              console.log(this.futureArray);

          } ));

    }

    ngOnInit()

    {

        this.gettask()

        .subscribe((data) => {

          console.log("Check1");

          console.log(data)

        });

    }

    completeProject(elem: OnTable) {

      var temp:OnTable[] = [];

      this.onGoingArray.forEach(element => {

        if(elem.id != element.id) {

          temp.push(element);

        }

        else {

          elem.assignstatus = 2;

          this.completeArray.push(elem);

          this.completedStatus = true;

        }

      });

      this.team.push({name:elem.team});

      this.team.sort(function(a,b) {

        if(a.name>b.name){

          return 1

        }

        else {

          return -1

        }

      });

      var curr = []; //Used as temp variable to remove the freed team

      this.tempteam.forEach( element => {

        if(element != elem.name) {

          curr.push(element);

        }

      });

      this.tempteam = [];

      curr.forEach(element => this.tempteam.push(element));

      console.log("The available teams are");

      console.log(this.team);

      console.log(this.tempteam);

      this.onGoingArray = [];

      temp.forEach(element => this.onGoingArray.push(element));

      this.completeArray.forEach(el => temp.push(el));

      this.futureArray.forEach(el => temp.push(el));

      console.log("Modified ones are: ");

      console.log(this.onGoingArray);

      console.log(this.futureArray);

      console.log(this.completeArray);

      temp.sort(function(a,b){

        return (a.id-b.id);

      });

      this.http.post("http://localhost:3000/update-projects1",{id: 1, team: elem.team, data: temp}).subscribe((res) => console.log(res));

    }

    confirm(element: FutureTable) {

      let array: FutureTable[] = [];

      console.log(element.tempid);

      this.tempteam.push(element.team);

      var current: Team[] = [];

      this.team.forEach((element) => {

        if(!this.tempteam.includes(element.name)){

          current.push({name: element.name})

        }

      })

      this.team = [];

      current.forEach(element => this.team.push(element));

      for(let i=0;i<this.futureArray.length;i++) {

        if(i != element.tempid-1) {

          array.push(this.futureArray[i]);

        }

      }

      this.futureArray = array;

      this.onGoingArray.push({assignstatus: 1, deadline: element.deadline, id: element.id, name: element.name, team: element.team,

        description: element.description,

        checkbox1: element.checkbox1,

        checkbox2: element.checkbox2,

        nemail: element.nemail,

        nsms: element.nsms});

      console.log("Modified project status is: ");

      console.log(this.onGoingArray);

      console.log(this.futureArray);

    }

    saveProjects() {

      var array: OnTable[] =[];

      this.onGoingArray.forEach(element => array.push(element));

      this.futureArray.forEach(element => array.push({assignstatus: element.assignstatus, deadline: element.deadline, id: element.id,

        name: element.name, team: element.team,

        description: element.description,

        checkbox1: element.checkbox1,

        checkbox2: element.checkbox2,

        nemail: element.nemail,

        nsms: element.nsms}));

      this.completeArray.forEach(element => array.push(element));

      array.sort(function(a,b){

        return a.id - b.id;

      });

      console.log(array);

      this.http.post("http://localhost:3000/update-projects",{data: array}).subscribe((response) => console.log(response));

    }

    ds1 = ['Name','Team','Deadline','View'];

    ds2 = ['ID','Name','Assign','Conformation'];

    ds3 = ['Id','Name', 'Team'];}

**10.3.3 ak.html**

<mat-sidenav-container class="sidenav-container">

  <mat-sidenav #drawer class="sidenav" fixedInViewport

      [attr.role]="(isHandset$ | async) ? 'dialog' : 'navigation'"

      [mode]="(isHandset$ | async) ? 'over' : 'side'"

      [opened]="(isHandset$ | async) === false">

    <mat-toolbar>Other Enquiries</mat-toolbar>

    <mat-nav-list>

      <mat-tab-group>

        <mat-tab label="Overall">

          <mat-card class="bar1">

            <canvas baseChart

              [datasets]="barChartData1"

              [labels]="barChartLabels1"

              [options]="barChartOptions"

              [plugins]="barChartPlugins"

              [legend]="barChartLegend"

              [chartType]="barChartType">

            </canvas>

            </mat-card>

        </mat-tab>

        <mat-tab label="Search Engine optimization"><h4 style="margin-left: 30px;margin-top: 20px;"><mat-icon style="display: inline-block;vertical-align:middle ">mood</mat-icon>    Enquiry recorded:2.5</h4>

          <mat-card class="bar1">

            <canvas baseChart

              [datasets]="barChartData2"

              [labels]="barChartLabels2"

              [options]="barChartOptions"

              [plugins]="barChartPlugins"

              [legend]="barChartLegend"

              [chartType]="barChartType2">

            </canvas>

            </mat-card>

        </mat-tab>

        <mat-tab label="Just Dial"> <h4 style="margin-left: 30px;margin-top: 20px;"><mat-icon style="display: inline-block;vertical-align:middle ">mood</mat-icon>    Enquiry recorded:1.5</h4>

          <mat-card class="bar1">

            <canvas baseChart

              [datasets]="barChartData3"

              [labels]="barChartLabels3"

              [options]="barChartOptions"

              [plugins]="barChartPlugins"

              [legend]="barChartLegend"

              [chartType]="barChartType3">

            </canvas>

            </mat-card>

        </mat-tab>

        <mat-tab label="Google"><h4 style="margin-left: 30px;margin-top: 20px;"><mat-icon style="display: inline-block;vertical-align:middle ">mood</mat-icon>    Enquiry recorded:1.0</h4>

          <mat-card class="bar1">

          <canvas baseChart

          [datasets]="barChartData4"

          [labels]="barChartLabels4"

          [options]="barChartOptions"

          [plugins]="barChartPlugins"

          [legend]="barChartLegend"

          [chartType]="barChartType4">

        </canvas>

        </mat-card>

        </mat-tab>

        <mat-tab label="Math cofig"><h4 style="margin-left: 30px;margin-top: 20px;"><mat-icon style="display: inline-block;vertical-align:middle ">mood</mat-icon>    Enquiry recorded:1.1</h4>

          <mat-card class="bar1">

          <canvas baseChart

          [datasets]="barChartData5"

          [labels]="barChartLabels5"

          [options]="barChartOptions"

          [plugins]="barChartPlugins"

          [legend]="barChartLegend"

          [chartType]="barChartType5">

        </canvas>

        </mat-card>

        </mat-tab>

      </mat-tab-group>

    </mat-nav-list>

  </mat-sidenav>

  <mat-sidenav-content>

    <mat-toolbar color="primary">

      <span>Manager</span>

    </mat-toolbar>

    <h1 style="text-align: center;font-family: algerian;">Enquiry</h1>

    <button mat-raised-button

    matBadge="4" matBadgePosition="before" matBadgeColor="accent" matBadgeSize="large">

    <b>Today</b>

    </button>

    <button mat-raised-button

        matBadge="8" matBadgePosition="before" matBadgeColor="accent" matBadgeSize="large">

     <b>This week</b>

    </button>

    <button mat-raised-button

    matBadge="8" matBadgePosition="before" matBadgeColor="accent" matBadgeSize="large">

    <b>This month</b>

    </button>

    <br><br><br><br>

    <button mat-raised-button

    matBadge="8" matBadgePosition="before" matBadgeColor="accent" matBadgeSize="large">

    <b>Yesterday</b>

    </button>

    <button mat-raised-button

    matBadge="8" matBadgePosition="before" matBadgeColor="accent" matBadgeSize="large">

    <b>Last week</b>

    </button>

    <button mat-raised-button

    matBadge="8" matBadgePosition="before" matBadgeColor="accent" matBadgeSize="large">

    <b>Last month</b>

    </button>

    <br><br><br><br>

    <!-------------------------------------Lead-------------------------------->

    <mat-card style="background-color: white;">

    <h1 style="text-align: center;font-family: algerian;color:black">Lead</h1>

    <mat-tab-group  style="background-color:#5c5b5a;color:black;height:580px">

<mat-tab label="Overall">

  <br>

<mat-card class="bar" >

<canvas baseChart

  [datasets]="barChartData"

  [labels]="barChartLabels"

  [options]="barChartOptions"

  [plugins]="barChartPlugins"

  [legend]="barChartLegend"

  [chartType]="barChartType">

</canvas>

</mat-card>

</mat-tab>

<mat-tab label="Enquiry">

  <mat-card class="bar">

  <canvas baseChart

  [datasets]="barChartData6"

  [labels]="barChartLabels"

  [options]="barChartOptions"

  [plugins]="barChartPlugins"

  [legend]="barChartLegend"

  [chartType]="barChartType6">

</canvas>

</mat-card>

</mat-tab>

<mat-tab label="Lead">

  <mat-card class="bar">

  <canvas baseChart

  [datasets]="barChartData7"

  [labels]="barChartLabels"

  [options]="barChartOptions"

  [plugins]="barChartPlugins"

  [legend]="barChartLegend"

  [chartType]="barChartType6">

</canvas>

</mat-card>

</mat-tab>

<mat-tab label="Conversion">

  <mat-card class="bar">

  <canvas baseChart

  [datasets]="barChartData8"

  [labels]="barChartLabels"

  [options]="barChartOptions"

  [plugins]="barChartPlugins"

  [legend]="barChartLegend"

  [chartType]="barChartType6">

</canvas>

</mat-card>

</mat-tab>

</mat-tab-group>

</mat-card>

<br><br>

<!-------------------------------------------------------------------------------------->

<mat-accordion>

  <mat-expansion-panel style="background-color:brown;color:black">

    <mat-expansion-panel-header>

      <mat-panel-title>

         Follow Up

      </mat-panel-title>

      <mat-panel-description >

        Missed followup will be here

      </mat-panel-description>

    </mat-expansion-panel-header>

    <table mat-table [dataSource]="dataSource" class="mat-elevation-z8">

      <!--- Note that these columns can be defined in any order.

            The actual rendered columns are set as a property on the row definition" -->

      <!-- ID -->

      <ng-container matColumnDef="id">

        <th mat-header-cell \*matHeaderCellDef> ID </th>

        <td mat-cell \*matCellDef="let element"> {{element.id}} </td>

      </ng-container>

      <!-- Name Column -->

      <ng-container matColumnDef="name">

        <th mat-header-cell \*matHeaderCellDef> Name </th>

        <td mat-cell \*matCellDef="let element"> {{element.name}} </td>

      </ng-container>

      <!-- Phone -->

      <ng-container matColumnDef="phone">

        <th mat-header-cell \*matHeaderCellDef> Phone </th>

        <td mat-cell \*matCellDef="let element"> {{element.phone}} </td>

      </ng-container>

      <!-- Email -->

      <ng-container matColumnDef="email">

        <th mat-header-cell \*matHeaderCellDef> Email </th>

        <td mat-cell \*matCellDef="let element"> {{element.email}} </td>

      </ng-container>

        <!-- Response -->

        <ng-container matColumnDef="response">

          <th mat-header-cell \*matHeaderCellDef> Response </th>

          <td mat-cell \*matCellDef="let element"> {{element.response}} </td>

        </ng-container>

    <!-- Quotation-->

    <ng-container matColumnDef="quotation">

      <th mat-header-cell \*matHeaderCellDef> Quotation </th>

      <td mat-cell \*matCellDef="let element"> {{element.quotation}} </td>

    </ng-container>

      <tr mat-header-row \*matHeaderRowDef="displayedColumns"></tr>

      <tr mat-row \*matRowDef="let row; columns: displayedColumns;"></tr>

    </table>

  </mat-expansion-panel>

  <br>

  <mat-expansion-panel style="background-color:blue;color:black">

    <mat-expansion-panel-header>

      <mat-panel-title>

         Follow Up

      </mat-panel-title>

      <mat-panel-description >

        Today followup will be here

      </mat-panel-description>

    </mat-expansion-panel-header>

    <table mat-table [dataSource]="dataSource" class="mat-elevation-z8">

      <ng-container matColumnDef="id">

        <th mat-header-cell \*matHeaderCellDef> ID </th>

        <td mat-cell \*matCellDef="let element"> {{element.id}} </td>

      </ng-container>

      <!-- Name Column -->

      <ng-container matColumnDef="name">

        <th mat-header-cell \*matHeaderCellDef> Name </th>

        <td mat-cell \*matCellDef="let element"> {{element.name}} </td>

      </ng-container>

      <!-- Phone -->

      <ng-container matColumnDef="phone">

        <th mat-header-cell \*matHeaderCellDef> Phone </th>

        <td mat-cell \*matCellDef="let element"> {{element.phone}} </td>

      </ng-container>

      <!-- Email -->

      <ng-container matColumnDef="email">

        <th mat-header-cell \*matHeaderCellDef> Email </th>

        <td mat-cell \*matCellDef="let element"> {{element.email}} </td>

      </ng-container>

        <!-- Response -->

        <ng-container matColumnDef="response">

          <th mat-header-cell \*matHeaderCellDef> Response </th>

          <td mat-cell \*matCellDef="let element"> {{element.response}} </td>

        </ng-container>

    <!-- Quotation-->

    <ng-container matColumnDef="quotation">

      <th mat-header-cell \*matHeaderCellDef> Quotation </th>

      <td mat-cell \*matCellDef="let element"> {{element.quotation}} </td>

    </ng-container>

      <tr mat-header-row \*matHeaderRowDef="displayedColumns"></tr>

      <tr mat-row \*matRowDef="let row; columns: displayedColumns;"></tr>

    </table>

  </mat-expansion-panel>

</mat-accordion>

  <!--------------------------------------------------------------------------->

  <br>

  </mat-sidenav-content>

</mat-sidenav-container>

**10.3.4 ak.ts**

import { Component } from '@angular/core';

import { BreakpointObserver, Breakpoints } from '@angular/cdk/layout';

import { Observable } from 'rxjs';

import { map, shareReplay } from 'rxjs/operators';

import { ChartOptions, ChartType, ChartDataSets } from 'chart.js';

import { Label } from 'ng2-charts';

@Component({

  selector: 'm',

  templateUrl: './manage.component.html',

  styleUrls: ['./manage.component.css']

})

export class ManageComponent {

  displayedColumns: string[] = ['id', 'name', 'phone', 'email','response','quotation'];

  dataSource = ELEMENT\_DATA;

  public barChartOptions: ChartOptions = {

    responsive: true,

  };

  public barChartLabels: Label[] = ['January', 'February', 'March', 'April', 'May', 'June', 'July','August','September','October','November','December'];

  public barChartType: ChartType = 'bar';

  public barChartLegend = true;

  public barChartPlugins = [];

  public enq=e;

  public p\_enq=y1;

  public lead=l;

  public p\_lead=y2;

  public conv=c;

  public p\_conv=y3;

  public barChartData: ChartDataSets[] = [

    { data: this.enq, label: 'Enquiry' },

    { data:this.lead , label: 'Lead' },

    { data:this.conv , label: 'Conversion' },

  ];

  //////////////////////////

  public barChartType6: ChartType = 'line';

  public barChartData6: ChartDataSets[] = [

    { data: this.p\_enq, label: 'Enquiry' ,fill:'false',borderColor:'blue',pointBackgroundColor:'red',backgroundColor:'blue'}

  ];

  public barChartData7: ChartDataSets[] = [

    { data: this.p\_lead, label: 'Lead' ,fill:'false',borderColor:'green',pointBackgroundColor:'red',backgroundColor:'green'}

  ];

  public barChartData8: ChartDataSets[] = [

    { data: this.p\_conv, label: 'Conversion' ,fill:'false',borderColor:'violet',pointBackgroundColor:'red',backgroundColor:'violet'}

  ];

////////////////////////////////////

public barChartLabels1: Label[] = ['SEO','JD','Google','MC'];

  public barChartType1: ChartType = 'bar';

  public barChartData1: ChartDataSets[] = [

    { data: [65, 59, 80, 81], label: 'First Enquiry',stack:'a' },

    { data: [65, 59, 80, 81], label: 'Second Enquiry',stack:'a' },

  ];

  //////////////////////

  //////////////////////

  public barChartLabels2: Label[] = ['Search engine optimization'];

  public barChartType2: ChartType = 'radar';

  public barChartData2: ChartDataSets[] = [

    { data: [65], label: 'First Enquiry',stack:'a' ,fill:'1'},

    { data: [80], label: 'Second Enquiry',stack:'a' },

  ];

  //////////////////////

  public barChartLabels3: Label[] = ['Just dial'];

  public barChartType3: ChartType = 'radar';

public barChartData3: ChartDataSets[] = [

    { data: [1.5], label: 'First Enquiry',stack:'a' },

    { data: [3.0], label: 'Second Enquiry',stack:'a' },

  ];

  //////////////////////

  public barChartLabels4: Label[] = ['Google'];

  public barChartType4: ChartType = 'radar';

barChartData4: ChartDataSets[] = [

    { data: [1.0], label: 'First Enquiry',stack:'a' },

    { data: [3.0], label: 'Second Enquiry',stack:'a' },

  ];

  //////////////////////

  public barChartLabels5: Label[] = ['Mat Config'];

  public barChartType5: ChartType = 'radar';

  public barChartData5: ChartDataSets[] = [

    { data: [1.1], label: 'First Enquiry',stack:'a' },

    { data: [2.0], label: 'Second Enquiry',stack:'a' },

  ];

  //////////////////////

  screenWidth: number;

  isHandset$: Observable<boolean> = this.breakpointObserver.observe(Breakpoints.Handset)

    .pipe(

      map(result => result.matches),

      shareReplay()

    );

  constructor(private breakpointObserver: BreakpointObserver) {

// set screenWidth on page load

this.screenWidth = window.innerWidth;

window.onresize = () => {

  // set screenWidth on screen size change

  this.screenWidth = window.innerWidth;

};

}

}

export interface PeriodicElement {

  name: string;

  id: number;

     phone: number;

  email: string;

  response:string;

  quotation:number;

}

const ELEMENT\_DATA: PeriodicElement[] = [

  {id: 1, name: 'Decoprio', phone:7550374125, email: 'Decops.inception@outlook.com',response:'SMS',quotation:2001},

];

//////////////////////Prediction//////////////

const regress = (x, y) => {

  const n = y.length;

  let sx = 0;

  let sy = 0;

  let sxy = 0;

  let sxx = 0;

  let syy = 0;

  for (let i = 0; i < n; i++) {

      sx += x[i];

      sy += y[i];

      sxy += x[i] \* y[i];

      sxx += x[i] \* x[i];

      syy += y[i] \* y[i];

  }

  const mx = sx / n;

  const my = sy / n;

  const yy = n \* syy - sy \* sy;

  const xx = n \* sxx - sx \* sx;

  const xy = n \* sxy - sx \* sy;

  const slope = xy / xx;

  const intercept = my - slope \* mx;

  const r = xy / Math.sqrt(xx \* yy);

  const r2 = Math.pow(r,2);

  let sst = 0;

  for (let i = 0; i < n; i++) {

     sst += Math.pow((y[i] - my), 2);

  }

  const sse = sst - r2 \* sst;

  const see = Math.sqrt(sse / (n - 2));

  const ssr = sst - sse;

  return {slope, intercept, r, r2, sse, ssr, sst, sy, sx, see};

}

//////////enquiry////////////////

let e=[65, 59, 80, 81, 56, 55, 40,40];

let y1=[];

for(let i=0;i<e.length;i++)

{

  y1.push(e[i]);

}

let x=[];

for(let i=1;i<=y1.length;i++)

{

  x.push(i);

}

for(let i=y1.length+1;i<=12;i++)

{

  let p=(regress(x,y1).intercept)+((regress(x, y1).slope)\*i);

  x.push(i);

  y1.push(Math.round(p));

}

///////////////////////////////////////Lead//////////////////////////

let l=[28, 48, 40, 19, 86, 27, 90,70];

let y2=[];

for(let i=0;i<l.length;i++)

{

  y2.push(l[i]);

}

let x1=[];

for(let i=1;i<=y2.length;i++)

{

  x1.push(i);

}

for(let i=y2.length+1;i<=12;i++)

{

  let p=(regress(x1,y2).intercept)+((regress(x1, y2).slope)\*i);

  x1.push(i);

  y2.push(Math.round(p));

}

/////////////////conversion////////////////

let c=[65, 59, 80, 81, 56, 55, 40,80];

let y3=[];

for(let i=0;i<l.length;i++)

{

  y3.push(c[i]);

}

let x2=[];

for(let i=1;i<=y3.length;i++)

{

  x2.push(i);

}

for(let i=y3.length+1;i<=12;i++)

{

  let p=(regress(x2,y3).intercept)+((regress(x2, y3).slope)\*i);

  x2.push(i);

  y3.push(Math.round(p));

}

**10.4 Team Leader component**

**10.4.1 teamleader.component.html**

<mat-accordion [multi]="true">

  <div class = "ex1">

    <br><br><br>

          <h1 style="margin-left: 0%;color: white;">On Going Project</h1>

      <div style="margin-left: 0%;">

        <div class="left">

          <h5 style="color: red;">Project Name: </h5><p style="color: white;">{{name}}</p>

          <h5 style="color: red;">DeadLine: </h5><p style="color: white;">{{deadline}}</p>

        </div>

        <div class="left">

          <h5 style="color: red;">Email: </h5><p style="color: white;">{{email}}</p>

          <h5 style="color: red;">Ph no: </h5><p style="color: white;">{{phno}}</p>

        </div>

        <div class="left">

          <h5 style="color: red;margin-left: 5%;">Description</h5><p style="color: white;">{{description}}</p>

        </div>

        <div style="width: 600px;margin-left: 5%;">

          <!-- <h5 style="color: red;">Type Of Website to be devoloped</h5> -->

        <div class="left">

          <h5 style="color: red;">Type Of Website to be devoloped</h5>

          <div style="color: white;">

            <mat-checkbox name="static" [(ngModel)]="checkbox1.static">STACTIC WEBSITE</mat-checkbox><br>

            <mat-checkbox name="dynamic" [(ngModel)]="checkbox1.dynamic">DYNAMIC</mat-checkbox><br>

            <mat-checkbox name="economic" [(ngModel)]="checkbox1.economic">ECONOMIC</mat-checkbox><br>

            <mat-checkbox name="appln" [(ngModel)]="checkbox1.appln">WEB APPLICATION</mat-checkbox><br>

            <mat-checkbox name="mob" [(ngModel)]="checkbox1.mob">MOBILE APP</mat-checkbox>

          </div>

        </div>

        <div class="middle" style="color: white;">

          <br>

          <mat-checkbox name="seo" [(ngModel)]="checkbox2.seo">SEO</mat-checkbox><br>

          <mat-checkbox name="sem" [(ngModel)]="checkbox2.sem">SEM (PPC)</mat-checkbox><br>

          <mat-checkbox name="smo" [(ngModel)]="checkbox2.smo">SMO</mat-checkbox><br>

          <mat-checkbox name="smm" [(ngModel)]="checkbox2.smm">SMO</mat-checkbox><br>

        </div></div>

      </div>

      <div style="margin-top: 3%;margin-left: 0%;"><mat-expansion-panel class="ex1" [expanded]="true" style="margin-bottom: 3%; margin-top: 3%;margin-left: 0%;opacity: 0.95;">

        <mat-expansion-panel-header>

            <mat-panel-title>

              View Tasks

            </mat-panel-title>

        </mat-expansion-panel-header>

        <mat-card class="card" class="mat-elevation-z8">

            <p \*ngFor = "let item of task">

                <mat-checkbox [(ngModel)]="item.status">{{item.content}}</mat-checkbox>

            </p>

            <hr>

            <button mat-raised-button color="primary" style="margin-left: 40%;" (click)="openDialog()">CREATE</button>

        </mat-card>

      </mat-expansion-panel>

      <div><mat-expansion-panel class="ex1" [expanded]="true" style="margin-bottom: 3%; margin-top: 3%;margin-left: 0%;opacity: 0.95">

          <mat-expansion-panel-header>

            <mat-panel-title>

                Assignment

            </mat-panel-title>

          </mat-expansion-panel-header>

          <table mdbTable small="true" class="mat-elevation-z8">

            <thead>

              <tr style="text-align: center;">

                <th \*ngFor="let head of displayedColumns" scope="col">{{head}} </th>

              </tr>

            </thead>

            <tbody>

              <tr mdbTableCol \*ngFor="let el of dataSource">

                <td scope="row" style="text-align: center;">{{el.id}}</td>

                <td style="text-align: center;">{{el.content}}</td>

                <td style="text-align: center;">

                  <mat-form-field>

                    <mat-label>SELECT EMPLOYEE</mat-label>

                    <mat-select [(ngModel)]="el.employee" [disabled]="el.empAssignStatus" (selectionChange)="saveTask(el)" name="categorie" style="width: 50;">

                      <mat-option \*ngFor="let e of emparray" [value]="e.name">{{e.name}}</mat-option>

                    </mat-select>

                 </mat-form-field>

                </td>

              </tr>

            </tbody>

          </table>

      </mat-expansion-panel></div>

       <mat-expansion-panel class="ex1" [expanded]="true" style="margin-bottom: 3%; margin-top: 3%;margin-left: 0%;opacity: 0.95">

        <mat-expansion-panel-header>

            <mat-panel-title>

              Quatation

            </mat-panel-title>

        </mat-expansion-panel-header>

        <mat-form-field style="margin-left: 80px;width: 60%">

            <textarea matInput placeholder="Address Line" rows="3" [(ngModel)]="quatation\_html"></textarea>

        </mat-form-field>

        <button mat-button color="accent" style="margin-left: 2%;" (click)="saveQuatation()">NOTIFY</button>

       </mat-expansion-panel>

       <mat-expansion-panel class="ex1" [expanded]="true" style="margin-left: 0%;opacity: 0.95">

        <mat-expansion-panel-header>

            <mat-panel-title>

              Employee Status

            </mat-panel-title>

        </mat-expansion-panel-header>

        <table mdbTable small="true" class="mat-elevation-z8">

          <thead>

            <tr>

              <th \*ngFor="let head of displayedColumns1" scope="col" style="text-align: center;">{{head}} </th>

            </tr>

          </thead>

          <tbody>

            <tr mdbTableCol \*ngFor="let el of dataSource1">

              <td scope="row" style="text-align: center;">{{el.id}}</td>

              <td style="text-align: center;">{{el.name}}</td>

              <td style="text-align: center;">{{el.assignmentStatus}}</td>

            </tr>

          </tbody>

        </table>

       </mat-expansion-panel></div>

   </div>

  </mat-accordion>

**10.4.2 teamleader.component.ts**

getdata()

    {

        return this.http.get("http://localhost:3000/team-task").pipe(

        map((responce)=>{

            console.log(responce);

            (Object.values(responce)[0][0].tasks).forEach(element => {

                this.task.push({id: element.id, content: element.content, status: element.status,empAssignStatus: element.empAssignStatus, employee: element.employee});

                this.taskStatus = true;

            });

            console.log(Object.values(responce)[0][0].employee);

            (Object.values(responce)[0][0].employee).forEach(element => {

                this.emparray.push({id: element.id, name: element.name, assignmentStatus: element.assignmentStatus});

            });

            this.team\_id = Object.values(responce)[0][0].team\_id;

            this.teamLead\_name = Object.values(responce)[0][0].teamLead\_name;

            // this.ongoingProjects = Object.values(responce)[0][0].ongoingProjects;

            this.quatation = Object.values(responce)[0][0].quatation;

            this.phno = Object.values(responce)[0][0].phno;

            this.email = Object.values(responce)[0][0].email;

            this.description = Object.values(responce)[0][0].ongoingProjects.description;

            this.checkbox1 = Object.values(responce)[0][0].ongoingProjects.checkbox1;

            this.checkbox2 = Object.values(responce)[0][0].ongoingProjects.checkbox2;

            this.nemail = Object.values(responce)[0][0].ongoingProjects.nemail;

            this.nsms = Object.values(responce)[0][0].ongoingProjects.nsms;

            this.projectid = Object.values(responce)[0][0].ongoingProjects.projectid;

            this.name = Object.values(responce)[0][0].ongoingProjects.name;

            this.deadline = Object.values(responce)[0][0].ongoingProjects.deadline;

            console.log("the checkboxes are: ");

            console.log(this.checkbox1);

            console.log(this.checkbox2.sem);

            console.log("The task array is: ");

            console.log(this.task);

            console.log(this.emparray);

            return responce;

        } ));

    }

    ngOnInit()

    {

        this.getdata()

        .subscribe((data) => {

          console.log(data)

        });

        console.log("Okk");

        this.dataSource = this.task;

        this.dataSource1 = this.emparray;

        console.log(this.dataSource1);

        console.log("Okk");

    }

    saveTask(el: any){

        console.log("Task arrived");

        console.log(el);

        let e\_id = 0;

        this.emparray.forEach(element => {

            if(element.name == el.employee) {

                element.assignmentStatus = true;

                el.empAssignStatus = true;

                e\_id = element.id;

            }

        });

        console.log("This is updated to database");

        var send = {e\_id: e\_id, t\_id: el.id, e\_task: el.content, employee: this.emparray, tasks: this.task};

        console.log(send);

        var editTask = {teamid: this.team\_id, update: send};

        this.http.post("http://localhost:3000/task-update",editTask).subscribe(responce => console.log(responce));

    }

    saveQuatation() {

        this.quatation.push({id: this.quatation.length+1, stmt: this.quatation\_html});

        console.log(this.quatation\_html);

        this.quatation\_html = "";

        console.log(this.quatation);

        this.http.post("http://localhost:3000/quatation-update",{teamid: this.team\_id,phno: this.phno, email: this.email, quatation: this.quatation,nemail: this.nemail, nsms: this.nsms}).subscribe(responce => console.log(responce));

    }

    openDialog(): void {

        const dialogRef = this.dialog.open(TaskDialogComponent,{

            width: '250px',

            data: {character: this.character}

          });

        dialogRef.afterClosed().subscribe(result => {

            this.task.push({id: this.task.length+1, content: result, status: false,empAssignStatus: false, employee: ""});

            console.log(this.task);

            this.http.post("http://localhost:3000/task-update1",{teamid: this.team\_id,tasks: this.task}).subscribe(responce => console.log(responce));

        });

    }

    displayedColumns: string[] = ['ID','Content','Select'];

    dataSource = this.task;

    displayedColumns1: string[] = ['ID','Name','Assignment Status'];

    dataSource1 = this.emparray;

**10.4.3 task-dialog.component.html**

<mat-form-field>

    <input matInput placeholder="Task" [(ngModel)]="data.character">

</mat-form-field>

<button mat-raised-button color="accent" style="margin-left: 2%;" [mat-dialog-close]="data.character">ADD</button>

**10.4.4 task-dialog.component.ts**

export class TaskDialogComponent implements OnInit {

  constructor(

    public dialogRef: MatDialogRef<TaskDialogComponent>,

    @Inject(MAT\_DIALOG\_DATA) public data: {character: string}) {}

  ngOnInit() {

  }}

**10.5 Employee Component**

**10.5.1 employee.component.html**

<div style="margin-top: 0%;margin-left: 10%;width: 50%"><br><br><br>

    <h4 style="color: white;">Pending Projects</h4>

    <mat-card>

      <table mdbTable small="true" class="mat-elevation-z8" style="opacity: 0.95;">

        <thead>

          <tr style="text-align: center;">

            <th \*ngFor="let head of ds" scope="col">{{head}} </th>

          </tr>

        </thead>

        <tbody>

          <tr mdbTableCol \*ngFor="let el of pendingTasks">

            <td scope="row" style="text-align: center;">{{el.id}}</td>

            <td style="text-align: center;">{{el.content}}</td>

            <td style="text-align: center;">

              <button mat-button color="accent" (click)="saveTask(el)">Completed</button>

            </td>

          </tr>

        </tbody>

      </table>

  </mat-card>

</div>

<div style="margin-top: 5%;margin-left: 10%;width: 50%">

    <h4 style="color: white;">Completed Projects</h4>

    <mat-card>

    <table \*ngIf="completedStatus" mdbTable small="true" class="mat-elevation-z8" style="opacity: 0.95;">

        <thead>

          <tr style="text-align: center;">

            <th \*ngFor="let head of ds1" scope="col">{{head}} </th>

          </tr>

        </thead>

        <tbody>

          <tr mdbTableCol \*ngFor="let el of completedTasks">

            <td scope="row" style="text-align: center;">{{el.id}}</td>

            <td style="text-align: center;">{{el.content}}</td>

          </tr>

        </tbody>

    </table>

  </mat-card>

</div>

**10.5.2 employee.component.ts**

getDetails() {

        return this.http.get("http://localhost:3000/get-employee").pipe(

            map((res)=>{

                console.log(res);

                this.id = Object.values(res)[0][0].id;

                this.name = Object.values(res)[0][0].name;

                this.team\_id = Object.values(res)[0][0].team\_id;

                this.tasks = Object.values(res)[0][0].tasks;

                this.tasks.forEach(element => {

                    if(element.status == true) {

                        this.completedTasks.push(element);

                        this.completedStatus = true;

                    }

                    else {

                        this.pendingTasks.push(element);

                        this.pendingStatus = true;

                    }

                });

            }));

    }

    ngOnInit()

    {

        this.getDetails().subscribe((data) => console.log(data));

    }

    saveTask(el: Task) {

        var temp: Task[] = [];

        this.pendingTasks.forEach(element =>{

            if(element.id != el.id) {

                temp.push(element);

            }

            else {

                el.status = true;

                element.status = true;

            }

        });

        this.pendingTasks = temp;

        this.completedTasks.push(el);

        console.log(this.pendingTasks);

        console.log(this.completedTasks);

        temp = [];

        this.completedTasks.forEach(element => temp.push(element));

        this.pendingTasks.forEach(element => temp.push(element));

        temp.sort(function(a,b){

            return a.id - b.id;

        });

        this.http.post("http://localhost:3000/update-employee",{id: this.id, team\_id: this.team\_id, spltask: el, update: temp}).subscribe(res => console.log(res));

        this.completedStatus = true;

    }

    ds = ['Id','Task','Conform'];

    ds1 = ['Id','Task'];

**10.6 Node part**

**10.6.1 app.js**

app.post("/client-form", (req, res, next) => {

    const info = new ClientForm({

        id: req.body.id,

        name: req.body.name,

        phno: req.body.phno,

        email: req.body.email,

        cname: req.body.cname,

        pname: req.body.pname,

        address: req.body.address,

        city: req.body.city,

        state: req.body.state,

        description: req.body.description,

        checkbox1: {static: req.body.checkbox1.static, dynamic: req.body.checkbox1.dynamic, economic:req.body.checkbox1.economic, appln: req.body.checkbox1.appln, mob: req.body.checkbox1.mob },

        checkbox2: { seo: req.body.checkbox2.seo, sem: req.body.checkbox2.sem, smo: req.body.checkbox2.smo, smm: req.body.checkbox2.smm },

        select: { select1: req.body.select.select1, select2: req.body.select.select2, select3: req.body.select.select3, select4: req.body.select.select4},

        notifyCustomer: { customer: req.body.notifyCustomer.customer, email: req.body.notifyCustomer.email, sms: req.body.notifyCustomer.sms },

        notifyMgmt: { mgmt: req.body.notifyMgmt.mgmt, email: req.body.notifyMgmt.email, sms: req.body.notifyMgmt.sms },

        assignment: req.body.assignment,

        date: req.body.date

    });

    info.save().then(() => {

      Manager.find({name: req.body.select.select2}).then(document => {

        document[0].projects.push({id: req.body.id, name: req.body.pname, assignstatus: 0,deadline:  req.body.date, team: "",

      checkbox1: req.body.checkbox1, checkbox2: req.body.checkbox2, nemail: req.body.notifyCustomer.email, nsms: req.body.notifyCustomer.sms});

      console.log(document[0]);

      Manager.updateOne({name: req.body.select.select2},{$set: document[0]}).

        then(() => console.log("Updated Sucessfully...!!")).catch((err) => console.log(err));

      });

      res.status(201).json({

        message: "Form added successfully"

      });

    });

});

app.post("/task-update", (req, res, next) => {

TeamLeaderForm.updateOne({team\_id: id},{$set: {tasks: req.body.update.tasks, employee: req.body.update.employee}})

  .then(() => {

    Employee.find({id: req.body.update.e\_id}).then((document) => {

      document[0].tasks.push({id: req.body.update.t\_id, content: req.body.update.e\_task, status: false});

      Employee.updateOne({id: req.body.update.e\_id},{$set: {tasks: document[0].tasks}}).then(() => console.log("Employee Updated"))

    })

    res.status(201).json({message: "Employee Updated sucess..!!"});

  });

});

app.post("/task-update1", (req, res, next) => {

  TeamLeaderForm.updateOne({team\_id: id},{$set: {tasks: req.body.tasks}})

  .then(() => {

    res.status(201).json({message: "Updated sucess..!!"});

  });

});

app.post("/update-employee", (req, res, next) => {

  Employee.updateOne({id: id},{$set: {tasks: req.body.update}})

  .then(() => {

    TeamLeaderForm.find({team\_id: req.body.team\_id}).then(document => {

      document[0].tasks.forEach(element => {

        if(element.id == req.body.spltask.id) {

          element.status = req.body.spltask.status;

        }

      });

      TeamLeaderForm.updateOne({team\_id: req.body.team\_id}, {$set: {tasks: document[0].tasks}}).then(() => console.log("Updated Employee"))

    });

    res.status(201).json({message: "Teamform Updated sucess..!!"});

  });

});

app.post("/get-tasks", (req,res,next) => {

  TeamLeaderForm.find({teamLead\_name: req.body.name}).then(documents => {

    res.status(201).json({tasks: documents[0].tasks});

  })

})

app.post("/update-projects", (req, res, next) => {

  Manager.updateOne({id: id},{$set: {projects: req.body.data}})

  .then(() => {

    (req.body.data).forEach(element => {

      if(element.team  != "")

      {

        ClientForm.find({id: element.id}).then((document) => {

          TeamLeaderForm.updateOne({teamLead\_name: element.team, alloted: 0},{$set: {

            alloted: 1,

            email: document[0].email,

            phno: document[0].phno,

            ongoingProjects: {projectid: element.id, name: document[0].pname, deadline: element.deadline,

              description: document[0].description,

              checkbox1: document[0].checkbox1,

              checkbox2: document[0].checkbox2,

              nemail: document[0].notifyCustomer.email,

              nsms : document[0].notifyCustomer.sms},

          }}).then(() => console.log("Updated..:)"));

        });

      }

    })

    res.status(201).json({message: "Projects Updated sucessfully..!!"});

  });

});

app.post("/update-projects1", (req, res, next) => {

  Manager.updateOne({id: id},{$set: {projects: req.body.data}})

  .then(() => {

    TeamLeaderForm.find({teamLead\_name: req.body.team}).then(document => {

      Employee.updateMany({team\_id: document[0].team\_id}, {$set: {tasks: []}}).then(() => {

        TeamLeaderForm.updateOne({teamLead\_name: req.body.team},{$set: {alloted: 0, email: "", phno: "",

          ongoingProjects: {projectid: 0, name: "", deadline: ""},

          tasks: [],

          quatation: []}}).then(() => console.log("Team DB updated..."));

      })

    })

  });

});

app.post("/quatation-update", (req, res, next) => {

  TeamLeaderForm.updateOne({team\_id: id},{$set: {quatation: req.body.quatation}})

  .then(() => {

    res.status(201).json({message: "Quataation Updated sucess..!!"});

    if(req.body.nsms) {

      const accountSid = "ACa1fb58f29be1cd23fbc1da4236b7ec43";

      const authTocken = "084d8b23b5a5fdb604763533f23a714f";

      const client = require('twilio')(accountSid, authTocken);

      req.body.phno = "+91"+req.body.phno;

      client.messages.create({

        to: req.body.phno,

        from: "+14432735073",

        body: 'Hii, '+req.body.quatation[req.body.quatation.length-1].stmt+' By your Project handeller:)'

      }).then((message) => console.log(message.sid)).catch((err) => console.log(err));

    }

    if(req.body.nemail) {

      sgMail.setApiKey('SG.eYuHpGIjRNCTeVLPXnBTrw.o3OJUW4QJzrywi8GgVFVov5fsFFB\_VRVsAN54fcM0u8');

      const msg = {

        to: req.body.email,

        from: 'beashaj2001@gmail.com',

        subject: "Quatation",

        text: 'Hii, '+req.body.quatation[req.body.quatation.length-1].stmt+' By your Project handeller:)',

      };

      sgMail.send(msg);

      console.log("Done");

    }

  })

});

app.get("/team-task", (req, res, next) => {

  TeamLeaderForm.find({team\_id: id}).then(documents => {

    res.status(200).json({

      data: documents

    });

  });

});

app.get("/get-projects", (req, res, next) => {

  Manager.find({id: id}).then(documents => {

    res.status(200).json({

      data: documents

    });

  });

});

app.get("/get-count", (req, res, next) => {

  ClientForm.find().then(documents => {

    res.status(200).json({

      value: documents.length

    });

  });

});

app.get("/get-teams", (req, res, next) => {

  Team.find().then(documents => {

    res.status(200).json({

      data: documents

    });

  });

});

app.get("/get-employee", (req, res, next) => {

  Employee.find({id: id}).then(documents => {

    res.status(200).json({

      data: documents

    });

  });

});

app.get("/get-person", (req, res, next) => {

  People.find({username: req.query.searchKey}).then(documents => {

    if(documents.length > 0){

      console.log(documents);

      id = documents[0].id;

      res.status(200).json({

        data: documents

      });

    }

    else{

      res.status(200).json({

        data: [{type: "null"}]

      });

    }

  });

});

**10.6.2 clischema.js**

const clientdataSchema = mongoose.Schema({

    id: Number,

    name: String,

    phno: String,

    email: String,

    cname: String,

    pname: String,

    address: String,

    city: String,

    state: String,

    description: String,

    checkbox1: {static: Boolean, dynamic: Boolean, economic: Boolean, appln: Boolean, mob: Boolean },

    checkbox2: { seo: Boolean, sem: Boolean, smo: Boolean, smm: Boolean },

    select: { select1: String, select2: String, select3: String, select4: String},

    notifyCustomer: { customer: Boolean, email: Boolean, sms: Boolean },

    notifyMgmt: { mgmt: Boolean, email: Boolean, sms: Boolean },

    assignment: Boolean,

    date: String

});

**10.6.3 managerschema.js**

const managerSchema = mongoose.Schema({

    id: Number,

    name: String,

    teams: [String],

    projects: [{id: Number, name: String, assignstatus: Number,deadline: String, team: String,

        description: String,

        checkbox1: {static: Boolean, dynamic: Boolean, economic: Boolean, appln: Boolean, mob: Boolean},

        checkbox2: {seo: Boolean, sem:Boolean, smo: Boolean, smm: Boolean},

        nemail: Boolean,

        nsms: Boolean}]

});

**10.6.4 post.js**

const teamleaderdataSchema = mongoose.Schema({

    alloted: Number,

    email: String,

    phno: String,

    team\_id: Number,

    teamLead\_name: String,

    employee: [{id : Number, name: String, assignmentStatus: Boolean}],

    ongoingProjects: { projectid: Number,name: String, deadline: String,

        description: String,

        checkbox1: {static: Boolean, dynamic: Boolean, economic: Boolean, appln: Boolean, mob: Boolean},

        checkbox2: {seo: Boolean, sem:Boolean, smo: Boolean, smm: Boolean},

        nemail: Boolean,

        nsms: Boolean

    },

    tasks : [{id: Number, content: String, employee\_id: Number, status: Boolean,empAssignStatus: Boolean, employee: String}],

    quatation: [{id: Number, stmt: String}]

});

**10.6.5 empscheema.js**

const empscheema = mongoose.Schema({

    id: Number,

    name: String,

    team\_id: Number,

    tasks: [{id: Number, content: String, status: Boolean}]

});

**10.6.6 personscheema.js**

const personSchema = mongoose.Schema({

    username: String,

    password: String,

    type: String,

    id: Number

});