HINDUSTAN
INSTITUTE OF TECHNOLOGY & SCIENCE
(DEEMED TO BE UNIVERSITY)
CHENNAI

Department of Information Technology					
CRUD OPERATIOIN WITH REST API IN ANGULAR					
Name of the Candidate:	VITUL CHAUHAN				
Register Number:	18132023		**Academic Year/Sem:**	2021-22/ VII	
Email id:	18132023@student.hindustanuniv.ac.in		**Mobile No:**	9790759563	
Mentor Name and Signature:	Ms. I. JUVANNA				
Industry Name :	Disgen International Private Limited.				
Duration of the Internship:	**From:**	01st July,2021	**To:**	30th September,2021	
Address of the Industry:	DiSGen International Pvt. Ltd. L-102, Dua Complex,24 Veer Savarkar Block, Shakarpur, Delhi-110092,India				
Contact Person in Industry:	**Name :**	M.Vaishali (Regional Manager – HR)			
	Mobile No:	+91 9550029721			
	Email Id:	hr@disgen.co			
Signature of the Candidate	M. Vaishali **Internship Incharge**		**HOD/IT**		

CRUD OPERATIOIN WITH REST API IN ANGULAR

INDUSTRY INTERNSHIP PROGRAMME REPORT

Submitted by

VITUL CHAUHAN (18132023)

in partial fulfillment of the Course- Industry Internship Programme-IIP

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

HINDUSTAN INSTITUTE OF TECHNOLOGY AND SCIENCE,

CHENNAI – 603 103

(September, 2021)

HINDUSTAN
INSTITUTE OF TECHNOLOGY & SCIENCE
(DEEMED TO BE UNIVERSITY)
CHENNAI

BONAFIDE CERTIFICATE

This is to certify that **Mr. VITUL CHAUHAN (18132023)** completed the report titled "CRUD OPERATIOIN WITH REST API IN ANGULAR" under my guidance for the partial fulfillment of the Course: Industry Internship Programme (IIP) in Semester VII of the Bachelor of Technology in Information Technology

[illegible]

SIGNATURE
ROHAN CHAUDHARY
Technical Architect
Regional Manager-HR
Disgen International

SIGNATURE
MS. I. JUVAANNA
Student Mentor
Assistant Professor
Information Technology

SIGNATURE
Dr. CERONAMANI SHARMILA
Head of the department
Professor
Information Technology

SIGNATURE
Dr. B.V BAIJU
Internship In-charge
Assistant Professor
Information Technology

HINDUSTAN
INSTITUTE OF TECHNOLOGY & SCIENCE
(DEEMED TO BE UNIVERSITY)
CHENNAI

TABLE OF CONTENTS

1. ABSTRACT

This report describes my internship at Disgen International. Disgen International IT firm, offering a wide gamut of Software Solutions services and other value-added services related to the IT domain by maintaining quality, efficiency, and sensitivity in most cost-effective way. The scope of this document is to identify and describe the analysis caried out, projects completed, experience gained and focuses on the achievement as a software engineer Intern. Hindustan University provides that glorious opportunity to their students of having an internship within their bachelor program. During my internship at Disgen International, I was introduced to some new technologies, languages, and frameworks. But the most amazing experience was to work in a multicultural work environment. Overall, I am very satisfied with the results of my internship. I was able to use my software development knowledge and apply it in a real organization working in a real- life problem I was able to see some differences in functioning that resulted from my efforts. Due to the character of the internship and the short time period spent at Disgen International, I took this experience as an opportunity to provide Disgen International with the skills which I earned while a information technology student at Hindustan University.

2. INTRODUCTION

Angular is a Typescript-based open-source framework for building client-side web applications. Since Typescript is a superset of JavaScript, let us first understand JavaScript briefly. JavaScript runs on the client-side of the web, which can be used to design or program how the web pages behave on the occurrence of an event.

Typically, JavaScript is used for interface interactions, slideshows, and other interactive components. JavaScript evolved quickly and has also been used for server-side programming (like in Node.js), game development, etc.

JavaScript deals with dynamic content, which is an important aspect of web development. Dynamic content refers to constantly changing content and it adapts to specific users. For example, JavaScript can be used to determine whether or not to render the mobile version of the website by checking the device, which is accessing the website.

This encouraged web developers to start creating their own custom JavaScript libraries for reducing the number of code lines and implementing complex functionalities easily.

jQuery is a fast, small, and feature-rich JavaScript library, which makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API. jQuery became the most popular one because it was easy to use and extremely powerful..

SINGLE-PAGE APPLICATIONS

• **Single-Page Applications** (or **SPA's**) are applications that are accessed via a web browser like other websites but offer more dynamic interactions resembling native mobile and desktop apps. The most notable difference between a regular website and SPA is the reduced amount of page refreshes. SPAs have a heavier usage of AJAX- a way to communicate with back-end servers without doing a full page refresh to get data loaded into our application. As a result, the process of rendering pages happens mostly on the client-side.

Progressive web apps: It uses modern web platform capabilities to deliver an app-like experience. It gives high performance, offline, and zero-step installation. So, working with Angular is pretty much easy.

Native: You can build native mobile apps with strategies using Ionic Framework, NativeScript, and React Native.

Desktop: Create desktop-installed apps across Mac, Windows, and Linux using the same Angular methods you've learned for the web plus.

Speed and Performance

Code generation: Angular turns your templates into code that's highly optimized for JavaScript virtual machines, giving you all the benefits of hand-written code with the productivity of a framework.

Universal: You can use any technology with Angular for serving the application like node.js, .NET, PHP and other servers

Code splitting: Angular apps load quickly with the new Component Router, which delivers automatic code-splitting, so users only load code required to render the view they request.

Productivity

Templates: Quickly create UI views with simple and powerful template syntax

Angular CLI: Command-line tools: You can easily and quickly start building components, adding components, testing them, and then, instantly deploy them using Angular CLI.

IDEs: Get intelligent code completion, instant errors, and other feedback in popular editors and IDEs like Microsoft's VS Code.

Testing: With the Jasmine test framework, the Angular CLI downloads and installs everything required for testing your application. When you run *ng test* command, the app will be built in watch mode and the Karma test runner will be launched. Once that is done, you will see the output in the terminal plus, the chrome browser will open up showing the output in Jasmine HTML Reporter.

Building Blocks of Angular

The main building blocks of Angular are:

- Modules
- Components
- Templates
- Metadata
- Data binding
- Directives
- Services
- Dependency injection

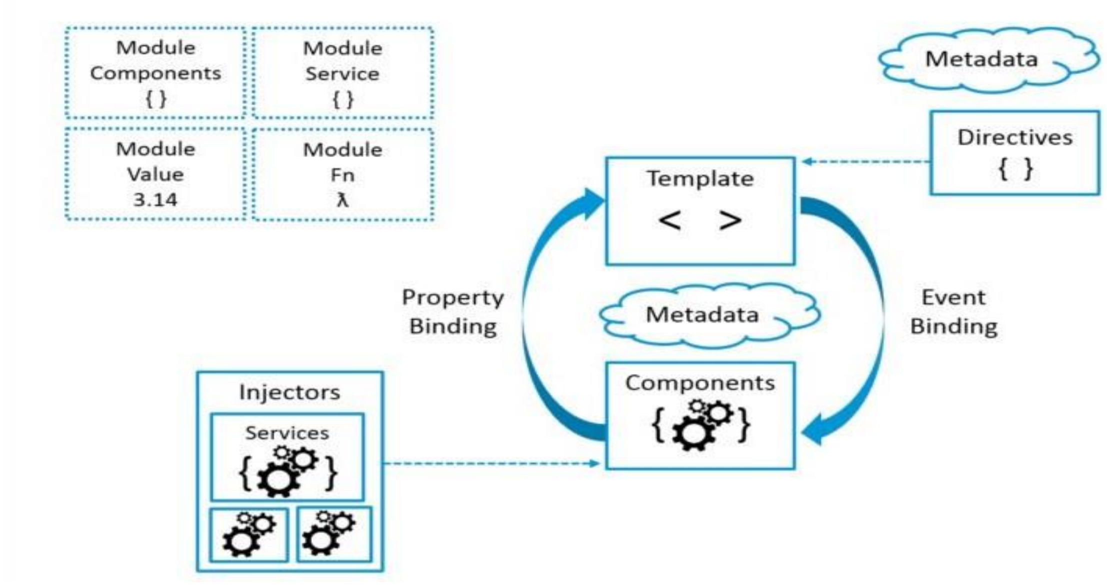


Figure 1

As Show in fig 1 the architecture of angular application based upon MVC framework

Modules

Angular apps are modular and to maintain modularity, we have *Angular modules* or you can say *NgModules*. Every Angular app contains at least one Angular module, i.e. the root module. Generally, it is named as *AppModule*. The *root module* can be the only module in a small application. While most of the apps have multiple modules. You can say, a module is a cohesive block of code with a related set of capabilities that have a specific application domain or a workflow. Any angular module is a class with @NgModule decorator.

3. COMPANY OVERVIEW

DiSGen International Private Limited is an ISO 9001-2015 IT firm, offering a wide gamut of Software Solutions services and other value-added services related to the IT domain by maintaining quality, efficiency and sensitivity that define our service standards in the most cost-effective way.

DISGEN INTERNATIONAL PRIVATE LIMITED is a Global Information Technology Solution Provider based in New Delhi, India. We are An ISO 9001:2015 & ISO/IEC 27001:2013 certified company engaged in discovering new ways to transform and empower global businesses. With powerful and innovative It solutions DiSGen has been powering businesses worldwide. Idea & Innovation Re-engineering, Re-Designing to make the solution optimize, up to date.

4. OBJECTIVES OF THE STUDY

- To Enable to create software quicker and with less effort
- Result in a more maintainable software
- Encourage good programming practices and design patterns like MVC
- Allow you to collaborate easier with other people
- Allow you to become proficient in a reasonable time
- Address problems that may arise in your software architecture such as Dependency Injection, DRY (Don't Repeat Yourself), etc
- Utilizing AngularJS formats adequately
- Questioning and adjusting information in various databases and getting to be plainly gifted with the API
- Quickly making perplexing structures
- Understanding two-way (proportional) information authoritative
- Presenting route usefulness in web applications
- Overseeing conditions with Injection frameworks
- Confining web applications to take into account worldwide groups of onlookers
- Securing web applications from dangers and pernicious clients
- Building different Angular orders
- Understanding the compiler for building better and more propelled orders
- Utilizing the testing system (Jasmine BDD) to test the web applications
- Organizing the web application utilizing the vigorous index structure
- Organizing, composing, and ultimately sending the application

5. METHODOLOGY

1. Question

REST CRUD APIs & HTTP GET Requests with HTTP Client.

Step 1 — Setting up Angular CLI 11

```
$ npm install -g @angular/cli
```

Output:

```
Angular CLI: 10.0.0
Node: 12.14.0
OS: linux x64

Angular:
...
Ivy Workspace:

Package                      Version
------------------------------------------------------------
@angular-devkit/architect    0.1100.0-next.0
@angular-devkit/core         11.0.0-next.0
@angular-devkit/schematics   11.0.0-next.0
@angular/cli                 11.0.0-next.0
rxjs                         6.5.4
```

Step 2 — Initializing a New Angular 11 Example Project

```
$ cd ~
$ ng new angular-httpclient-example
```

This will instruct the CLI to automatically set up routing in our project

Output:

```
Angular CLI: 11.0.2
Node: 12.14.0
OS: linux x64

Angular: 11.0.1
... animations, common, compiler, compiler-cli, core, forms
... language-service, platform-browser, platform-browser-dynamic
... router
Ivy Workspace: Yes

Package                           Version
------------------------------------------------------------------
@angular-devkit/architect         0.1100.2
```

```
@angular-devkit/build-angular     0.1100.2
@angular-devkit/build-optimizer   0.1100.2
@angular-devkit/build-webpack     0.1100.2
rxjs                              6.5.4
typescript                        3.7.5
webpack                           4.41.2
```

Next, navigate to our project's folder and run the local development server

```
$ cd angular-httpclient-example
$ ng serve
```

http://localhost:4200/ address.

Step 3 — Setting up a (Fake) JSON REST API

Use an external API service, create a real REST API server or create a fake API using json-server.

```
$ cd ~/angular-httpclient-example
$ npm install --save json-server
```

Next, create a server folder in the root folder of our Angular project:

```
$ mkdir server
$ cd server
```

In the server folder, create a database.json file and add the following JSON object:

```
{
    "products": []
}
```

This JSON file will act as a database for our REST API server.

```
$ cd ..
$ npm install faker --save
```

Now, create a generate.js file and add the following code:

```
var faker = require('faker');

var database = { products: []};

for (var i = 1; i<= 300; i++) {
  database.products.push({
    id: i,
    name: faker.commerce.productName(),
    description: faker.lorem.sentences(),
    price: faker.commerce.price(),
    imageUrl: "https://source.unsplash.com/1600x900/?product",
    quantity: faker.random.number()
  });
}

console.log(JSON.stringify(database));
```

Next, add the generate and server scripts to the package.json file:

```
  "scripts": {
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",

    "generate": "node ./server/generate.js > ./server/database.json",

  },
```

Next, head back to our command-line interface and run the generate script using the following command:

```
$ npm run generate
```

Finally, run the REST API server by executing the following command:

```

$ npm run server
```

These are the API endpoints we'll be able to use via our JSON REST API server:

- GET /products for getting the products,
- GET /products/<id> for getting a single product by id,
- POST /products for creating a new product,
- PUT /products/<id> for updating a product by id,
- PATCH /products/<id> for partially updating a product by id,
- DELETE /products/<id> for deleting a product by id.

Step 4 — Setting up Angular 11 HttpClient in our Project

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { HttpClientModule } from '@angular/common/http';

@NgModule({
  declarations: [
    AppComponent,
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    HttpClientModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Step 5 — Creating Angular 11 Components

```
$ cd ~/angular-httpclient-example
$ ng generate component home
```

This is the output of the command:

```
CREATE src/app/home/home.component.html (19 bytes)
CREATE src/app/home/home.component.ts (261 bytes)
CREATE src/app/home/home.component.css (0 bytes)
UPDATE src/app/app.module.ts (467 bytes)
```

Next, open the src/app/about/about.component.html and add the following code:

```
<p style="padding: 13px;">
An Angular 11 example application that demonstrates how to use HttpClient to consume REST APIs
</p>
```

We'll update the home component in the following steps.

Step 6 — Adding Angular 11 Routing

Head back to the src/app/app-routing.module.ts file, that was automatically created by Angular CLI for routing configuration, and import the components then add the routes as follows:

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { HomeComponent } from './home/home.component';
import { AboutComponent } from './about/about.component';


const routes: Routes = [
  { path: '', redirectTo: 'home', pathMatch: 'full'},
  { path: 'home', component: HomeComponent },
  { path: 'about', component: AboutComponent },
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

Step 7 — Styling the UI with Angular Material 11

Angular Material to our project and style our application UI.

```
<mat-toolbar color="primary">
  <h1>
    ngStore
  </h1>
  <button mat-button routerLink="/">Home</button>
  <button mat-button routerLink="/about">About</button>

</mat-toolbar>

<router-outlet></router-outlet>
```

Step 8 — Consuming the JSON REST API with Angular HttpClient 11

To consume JSON data from our REST API server in our application.

```
$ ng generate service data
```

Next, open the src/app/data.service.ts file, import and inject HttpClient as follows:

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Injectable({
  providedIn: 'root'
})
export class DataService {

  private REST_API_SERVER = "http://localhost:3000";

  constructor(private httpClient: HttpClient) { }
}
```

add a sendGetRequest() method that sends a GET request to the REST API endpoint to retrieve JSON data:

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Injectable({
  providedIn: 'root'
})
export class DataService {

  private REST_API_SERVER = "http://localhost:3000";

  constructor(private httpClient: HttpClient) { }

  public sendGetRequest(){
    return this.httpClient.get(this.REST_API_SERVER);
  }
}
```

The method simply invokes the get() method of HttpClient to send GET requests to the REST API server.

Next, open the src/app/home/home.component.html file and update it as follows:

```
<div style="padding: 13px;">
    <mat-spinner *ngIf="products.length === 0"></mat-spinner>

    <mat-card *ngFor="let product of products" style="margin-top:11px;">
        <mat-card-header>
            <mat-card-title>{{product.name}}</mat-card-title>
            <mat-card-subtitle>{{product.price}} $/ {{product.quantity}}
            </mat-card-subtitle>
        </mat-card-header>
        <mat-card-content>
            <p>
                {{product.description}}
            </p>
            <img style="height:100%; width: 100%;" src="{{ product.imageUrl }}" />
        </mat-card-content>
        <mat-card-actions>
      <button mat-button> Buy product</button>
    </mat-card-actions>
    </mat-card>
</div>
```

Step 9 — Adding URL Query Parameters to the HttpClient get() Method

```
import { HttpClient, HttpErrorResponse, HttpParams } from "@angular/common/http";
```

Step 10 — Getting the Full HTTP Response with Angular HttpClient 11

Go to the src/app/data.service.ts file and import the RxJS tap() operator:

```
import { retry, catchError, tap } from 'rxjs/operators';
```

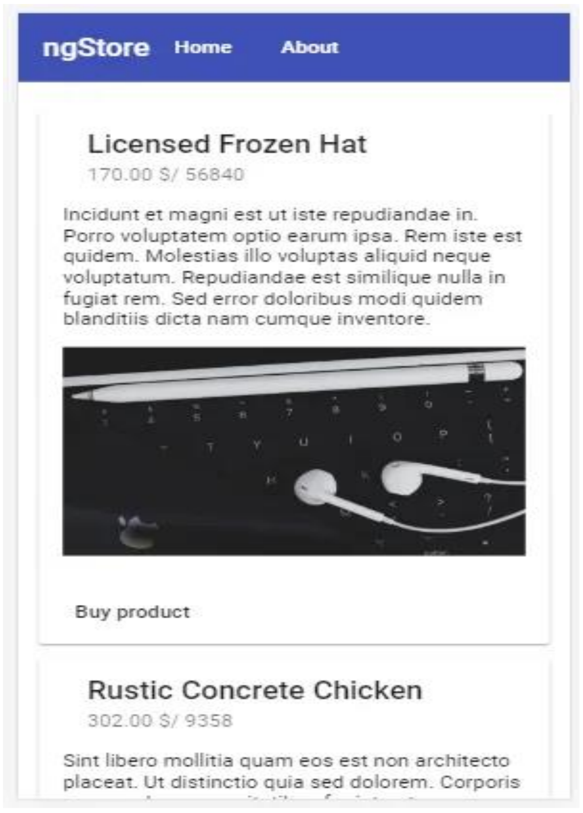
Step 11 — Requesting a Typed HTTP Response with Angular HttpClient 11

Step 12 — Building and Deploying our Angular 11 Application to Firebase Hosting

```
$ ng deploy
```

The command will produce an optimized build of your application it will upload the production assets to Firebase hosting.

6. Project Output:



Licensed Frozen Hat

170.00 $/ 56840

Incidunt et magni est ut iste repudiandae in. Porro voluptatem optio earum ipsa. Rem iste est quidem. Molestias illo voluptas aliquid neque voluptatum. Repudiandae est similique nulla in fugiat rem. Sed error doloribus modi quidem blanditiis dicta nam cumque inventore.

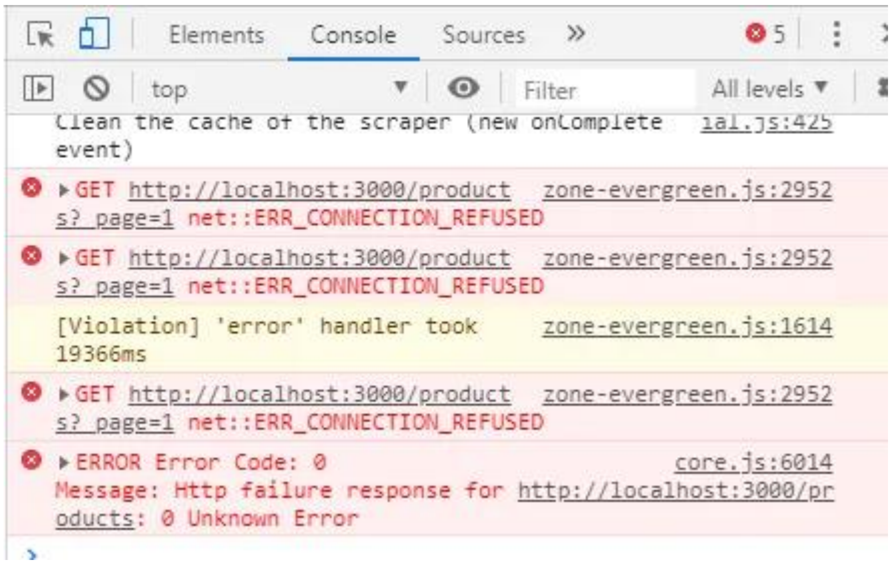
Rustic Concrete Chicken

302.00 $/ 9358

Sint libero mollitia quam eos est non architecto placeat. Ut distinctio quia sed dolorem. Corporis

Figure 2

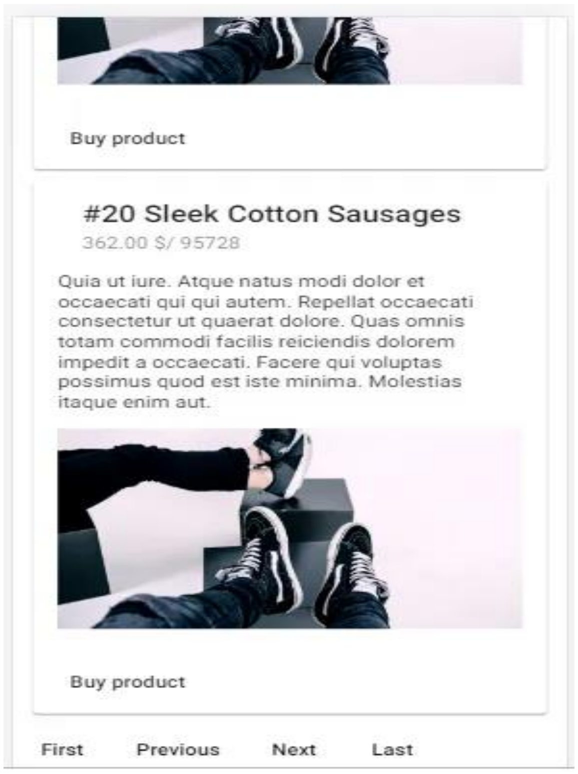
As showing in the figure2 the description and header information coming from the API And showing in the mat card with image card and replicated same card for other data also.



```
Clean the cache of the scraper (new onComplete   ial.js:425
event)
▶GET http://localhost:3000/product   zone-evergreen.js:2952
s? page=1 net::ERR_CONNECTION_REFUSED
▶GET http://localhost:3000/product   zone-evergreen.js:2952
s? page=1 net::ERR_CONNECTION_REFUSED
[Violation] 'error' handler took     zone-evergreen.js:1614
19366ms
▶GET http://localhost:3000/product   zone-evergreen.js:2952
s? page=1 net::ERR_CONNECTION_REFUSED
▶ERROR Error Code: 0                          core.js:6014
Message: Http failure response for http://localhost:3000/pr
oducts: 0 Unknown Error
```

Figure 3

As showing figure 3 this is the error came if the API response failed or serve is stop or not running



Buy product

#20 Sleek Cotton Sausages

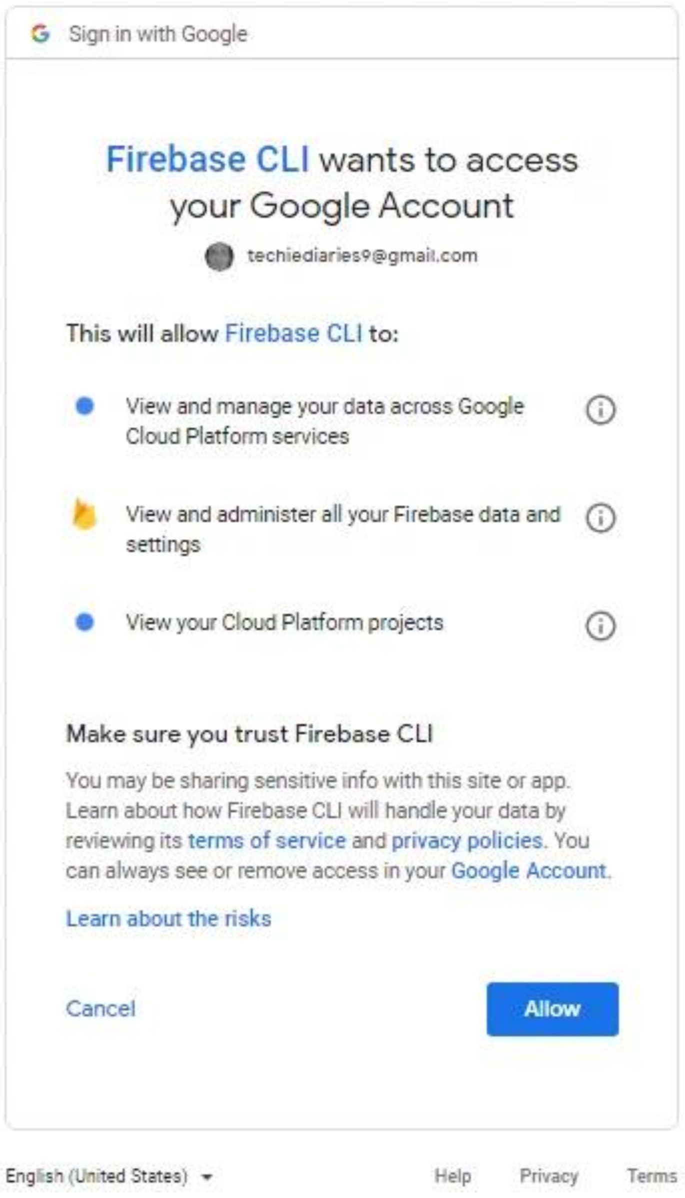
362.00 $/ 95728

Quia ut iure. Atque natus modi dolor et occaecati qui qui autem. Repellat occaecati consectetur ut quaerat dolore. Quas omnis totam commodi facilis reiciendis dolorem impedit a occaecati. Facere qui voluptas possimus quod est iste minima. Molestias itaque enim aut.

Buy product

Figure 4

As showing in figure 4 first, previous, next, last button implemented with routing concept to navigate page from one component to another without page load .


Sign in with Google

Firebase CLI wants to access your Google Account

techiediaries9@gmail.com

This will allow Firebase CLI to:

- View and manage your data across Google Cloud Platform services
- View and administer all your Firebase data and settings
- View your Cloud Platform projects

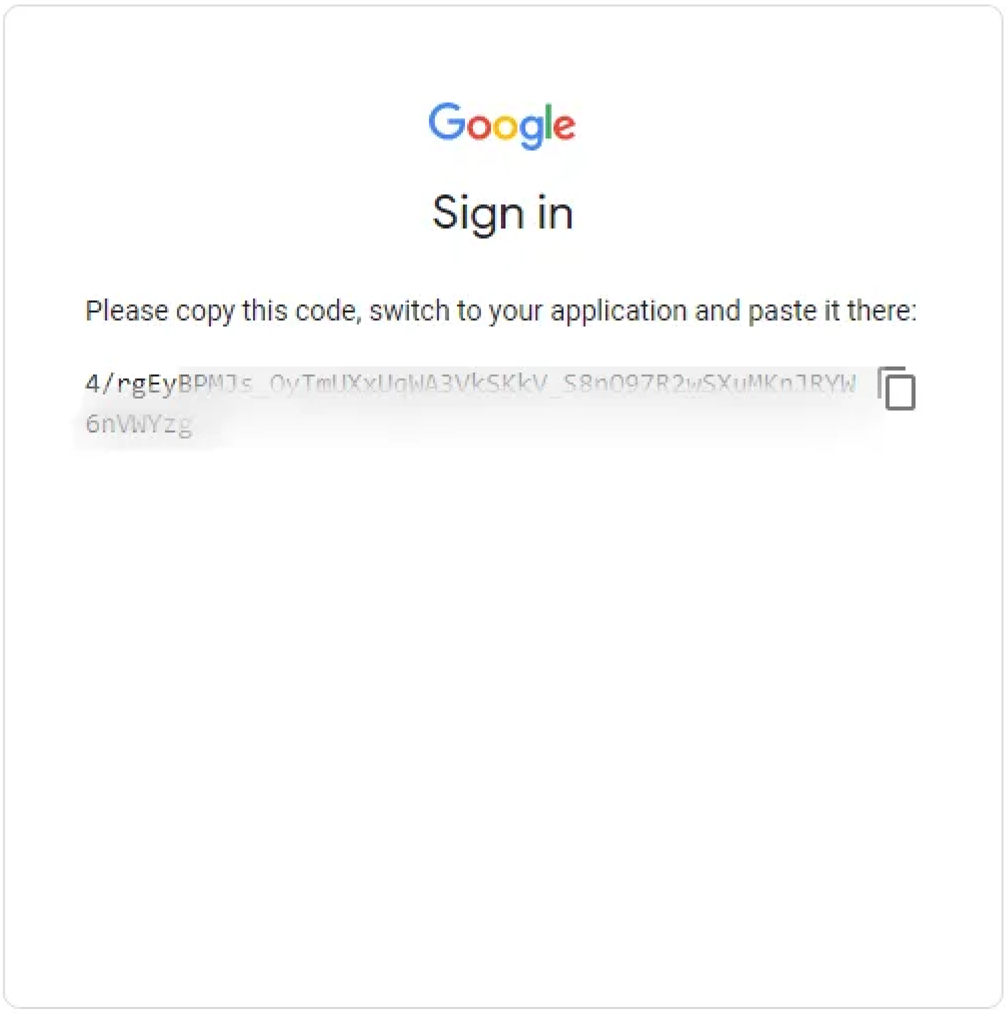
Make sure you trust Firebase CLI

You may be sharing sensitive info with this site or app. Learn about how Firebase CLI will handle your data by reviewing its terms of service and privacy policies. You can always see or remove access in your Google Account.

Learn about the risks

Figure 5

As shown in the figure 5 for access the login information in firebase console is asking allow the permission to store as json object in database.



Google

Sign in

Please copy this code, switch to your application and paste it there:

4/rgEy[redacted]

Figure 6

As showing in the figure 6 this is the Token generate after the authentication successful and Databases connect establishing with application

7. CONCLUSION

I believe the trial has shown conclusively that it is both possible and desirable to use Angular as the principal front end language:

- The Angular Framework by Google is very creative and easy to use.
- It is Single Page Applications (SPA) which can be helpful to load fast.
- ngModel directive in it can be used to implement two-way data binding.
- Code is organized into buckets which are known as "modules".
- Short and Effective Coding using Model View Controller (MVC).
- User Interface, which is Declarative which, uses HTML.
- Easy Integration in Angular is very useful advantage to do integration of third-party features with Angular.
- Angular can be used to make Mobile App, Desktop App and Web App which make it Cross- Platform Framework.
- Angular has been used in many large and popular websites so demand of Angular is high.

The training program having three destination was a lot more useful than staying at one place throughout the whole 12 weeks. In my opinion. I have gained lots of knowledge and experience needed to be successful in great engineering challenge as in my opinion, Engineering is after all a Challenge and not a job.

8. LEARNING OUTCOME

- Understand the principles of creating an effective web page, including an in-depth consideration of information architecture.
- Become familiar with graphic design principles that relate to web design and learn how to implement theories into practice.
- Develop skills in analyzing the usability of a web site.
- Understand how to plan and conduct user research related to web usability.
- Learn the language of the web: HTML and CSS.
- Learn CSS grid layout and flexbox.
- Learn techniques of responsive web design, including media queries.
- Develop skills in digital imaging (Adobe Photoshop.)
- Develop basic programming skills using Javascript and jQuery.
- Be able to embed social media content into web pages.

Verification Report			
Report - submitted	YES	NO	Internship Incharge sign:
Certificate -Submitted	YES	NO	HOD sign: