Cirrus Logic
Akash Khiste

# Art Gallery Modeling

## Problem Statement: -

Create a simple web application which does the following:

- each piece of art is tied to an artist
- artist can own more than one piece of art
- each art piece will also have a buyer
- show front end with a list of art pieces and artist + buyer name
- assume art piece is just an image file

## Summary of the feature:

The application provides feature of displaying the art piece with the name of associated artist and buyer. Application also provides a feature to upload the image into database using “upload image link”.

## Technologies Used:

Following technologies has been used for the implementation of the application.
Angular 8, HTML, CSS, Bootstrap, Firebase, Firebase Storage, Firebase Real-time Database
Visual Studio Code, Angular CLI, angularfire2

## Installation –

Angular 8 –

Requirements – Node Js

Install Node.js from www.nodejs.org

Installation –

Open Command Prompt.

Run command – npm install -g @angular/cli

Firebase –

Create a new project on www.firebase.com

Click on storage, click on get started, go to the rules tab and modify the rules to change permissions and bypass the authentication and publish the modified rule.

Modified rule –

```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write;
    }
  }
}
```

## Implementation –

Create an angular project using following command –

    ng new ArtGallery

Create three components using following command –

    ng generate component images - Main component
    ng generate component images-list – Component to display list of art pieces
    ng generate component images-upload – Component to upload an image

Create a service to upload and retrieve file from the database using following command –

    ng generate service image
    (file is available under shared folder in project)

Add components routes to the app-routing.modile.ts –

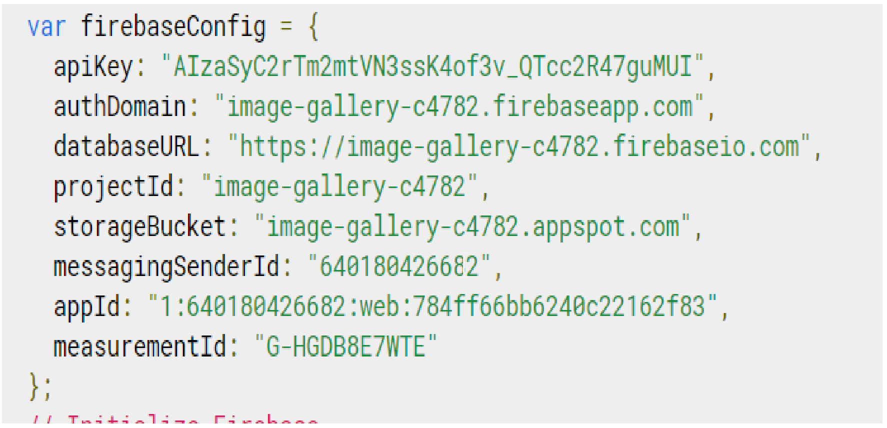
```
      const routes: Routes = [

  {path:'', redirectTo:'image/list',pathMatch:'full'},
  {path:'image', component:ImagesComponent, children:[
    {path:'upload', component:ImageComponent},
    {path:'list',component:ImageListComponent}
  ]}
];
```

Add tag <router-outlet></router-outlet> to the app.component.html and images.component.html allowing these fires to redirect to the router paths defined in the app-routing.modile.ts file.

Configure firebase with application –

Go to the firebase.com, Click on project overview. To configure firebase with project copy the generated firebase configuration script code and paste in application.



```
var firebaseConfig = {
  apiKey: "AIzaSyC2rTm2mtVN3ssK4of3v_QTcc2R47guMUI",
  authDomain: "image-gallery-c4782.firebaseapp.com",
  databaseURL: "https://image-gallery-c4782.firebaseio.com",
  projectId: "image-gallery-c4782",
  storageBucket: "image-gallery-c4782.appspot.com",
  messagingSenderId: "640180426682",
  appId: "1:640180426682:web:784ff66bb6240c22162f83",
  measurementId: "G-HGDB8E7WTE"
};
// Initialize Firebase
```

Go to environment.ts file in project and paste the firebase configuration

```
 export const environment = {

  production: false,
  firebaseConfig : {
    apiKey: "AIzaSyC2rTm2mtVN3ssK4of3v_QTcc2R47guMUI",
    authDomain: "image-gallery-c4782.firebaseapp.com",
    databaseURL: "https://image-gallery-c4782.firebaseio.com",
    projectId: "image-gallery-c4782",
    storageBucket: "image-gallery-c4782.appspot.com",
    messagingSenderId: "640180426682",
    appId: "1:640180426682:web:784ff66bb6240c22162f83",
    measurementId: "G-HGDB8E7WTE"
  }
};
```

In order to upload and retrieve images we will use angularfire2 package. To configure the package to out project run the following command –

npm install firebase @angular/fire --save

```
        ng add @angular/fire@next
```

Reference for angularfire2 package - https://github.com/angular/angularfire

Import angular fire module in app.module.ts

```
import { AngularFireModule } from '@angular/fire';
```

import the environment.ts in app.module.ts to access the firebase connection configuration.

```
import { environment } from '../environments/environment';
```

Add angular module to the import in the app.module.ts and call "intinalizeApp" function and pass the connection details of firebase as parameter.

```
AngularFireModule.initializeApp(environment.firebaseConfig),
```

In order to access the firebase storage and firebase database we will import the respective modules to app.module.ts

```
import { AngularFireStorageModule } from '@angular/fire/storage';
import { AngularFireDatabaseModule } from '@angular/fire/database';
```

now add the imported module into imports in app.module.ts –

```
imports: [
    BrowserModule,
    AppRoutingModule,
    AngularFireModule.initializeApp(environment.firebaseConfig),
    AngularFireStorageModule,
    AngularFireDatabaseModule,
  ],
```

To construct the forms and use angular forms we will import ReactiveFormsModule from @angular.forms

```
import { ReactiveFormsModule } from "@angular/forms";
```

now add the ReactiveFormsModule to the imports –

```
 imports: [
    BrowserModule,
    AppRoutingModule,
    AngularFireModule.initializeApp(environment.firebaseConfig),
    AngularFireStorageModule,
    AngularFireDatabaseModule,
    ReactiveFormsModule
  ],
```

## Components and Service functions and descriptions –

Image.service.ts –

It is a service used to upload and retrieve the data from firebase database.

getimageDetailList() - used to retrieve the details of associated with art piece which are imageUrl, artist name and buyer name.

```
      getimageDetailList(){

    this.imageDetailList = this.firebase.list('imageDetails');
  }
```

insertImageDetails() – function used to upload the details associated with an art piece to firebase database, which are imageUrl, artist name and buyer name.

```
insertImageDetails(imageDetails){
    this.imageDetailList.push(imageDetails);
  }
```

Image Component –

This component is used to upload the images to the firebase database along with the name of the artist and the name of the buyer.

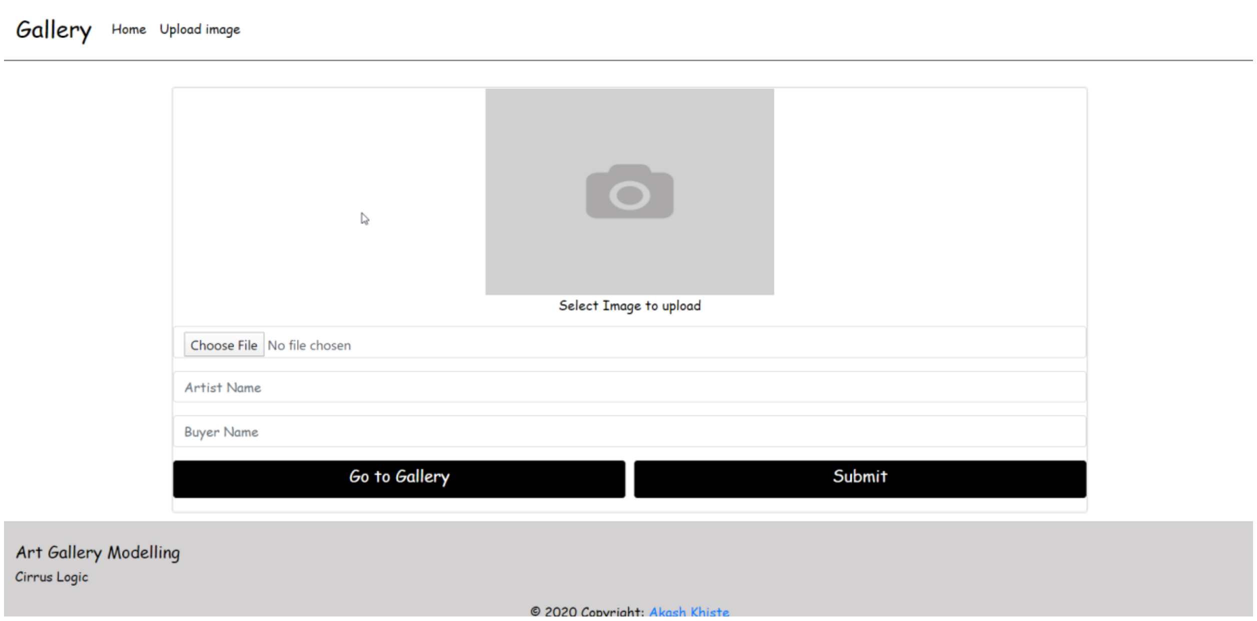


image.component.ts -

formTemplate - Variable defined to be used in the form to upload an image. Also used the Validators with the definition to apply validations on the webpage.

```
formTemplate = new FormGroup({
    artist : new FormControl('', Validators.required),
    buyer : new FormControl('', Validators.required),
    imageUrl : new FormControl('', Validators.required)
  })
```

showPreview () – function used to display the preview of the image to be uploaded to the database.

```
showPreview(event :any){
    if(event.target.files && event.target.files[0]){
      const reader = new FileReader();
      reader.onload = (e:any) => this.imgSrc = e.target.result;
      reader.readAsDataURL(event.target.files[0]);
      this.selectedImage = event.target.files[0];
    }
    else{
      this.imgSrc;
      this.selectedImage = null;
    }
  }
```

onSubmit() - onsubmit function is defined to upload the selected image and entered details to the firebase database. On clicking submit the image/upload form, the function is executed to take in the entered data from user and dump it in the database.

```
onSubmit(formValue){
    this.isSubmitted = true;
    if(this.formTemplate.valid){
      //variable filepath stores the file name and path in firestore, used split
to get rid of file extension, Date function used to uniqely
      // identify the files with same name
      var filePath = `${this.selectedImage.name.split('.').slice(0,-
1).join('.')}_${new Date().getTime()}`;
      const fileRef = this.storage.ref(filePath);
      this.storage.upload(filePath,this.selectedImage).snapshotChanges().pipe(
        finalize(()=>{
          fileRef.getDownloadURL().subscribe((url)=>{
            formValue['imageUrl']= url;
            this.service.insertImageDetails(formValue);
            this.resetForm();
          })
        })
      ).subscribe();
    }
  }
```

resetForm() – This function is used to reset the image/upload form back to its original state after an image is done uploading in the database when the submit button is clicked.

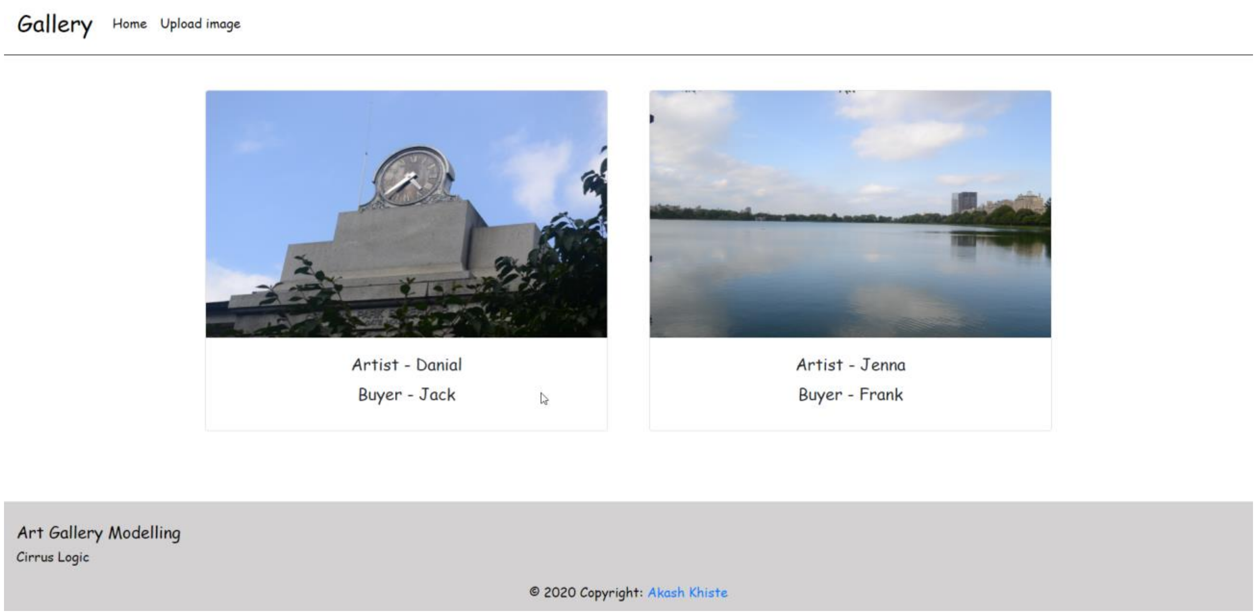
```
  resetForm(){
    this.formTemplate.reset();
    this.formTemplate.setValue({
      artist :'',
      buyer : '',
      imageUrl : ''
    });
    this.imgSrc='/assets/img/image-placeholder.png';
    this.selectedImage =null;
    this.isSubmitted = false;
  }
```

<u>Image.component.html –</u>

This html file is used to design the frontend of the image component. Used to upload images, artists' name, and buyers' name in the database. This component is designed using the <div> html elements and formgroup to successfuly implement the form in angular.

<u>Image-list component –</u>

This component is developed to display the list of art pieces and names of the associated artists and buyers.



<u>image-list.component.ts –</u>

this component is used to display the art piece (images) and the names of artist and buyers associated with the art piece. The component uses the service defined in the project "ImageService" to fetch the data from the firebase database.
A private variable service has been defined in order to inject the service into this component. Injecting service allows us to access the "imageDetailList" defined in the image.service.component

```
constructor(private service : ImageService) { }
```

ngOninit() -

Logic which is used to fetch and display the images has been written inside ngOnInit() function of angular since we want to display the images listed as the page loads.

snapshotChanges() – This function returns all the records present in the firebase database. As snapshotChanges returns an observable we can use subscribe function with it.

subscribe() – inside this function a call-back function has been implemented with a single parameter "list".

map() – This function is implemented with a call back function "item" to retrieve the details of the record.

Item.payload.val() - Returns a JSON object containing the all attributes of a record, and are stored in defined array "imageList".

rowIndexArray – It is used to render the downloaded images to html component.

```
  imageList : any[];
  rowIndexArray : any[];
```

```
ngOnInit() {
    this.service.imageDetailList.snapshotChanges().subscribe(
      list =>{
        this.imageList = list.map(item => {return item.payload.val();});
        this.rowIndexArray = Array.from(Array(Math.ceil(this.imageList.length)).k
eys());

      }
    );
  }
```

<u>image-list.component.html –</u>

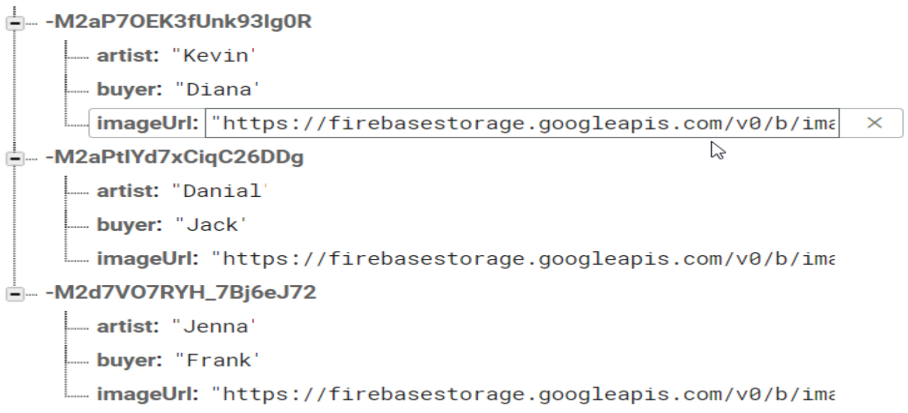
This html component is used to design the frontend of the image-list component to display the data on the webpage.
Here we are rendering the data from rowIndexArray using angular directive *ngFor.
Using string interpolation, the name of the artist and buyer is displayed.

```html
<div class="artpices col-md-12">
    <div class="container-fluid ">
        <div class="row">
            <!-- using ngIf module to check is array is empty -->
            <div *ngIf="imageList">
                <!-- parsing the data using ngFor module -->
                <div  *ngFor="let i of rowIndexArray" class="block col-md-6 .col-
md-4 .col-sm-12">
                 <div class="col-md-12 imagecard">
                    <div class="card mb-4" style="font-
family:Comic Sans MS, cursive, sans-serif ">
                        <img class="card-img-
top" [src]="imageList[i].imageUrl" style="height: 300px;">
                        <div class="card-body">
                            <h5 class="card-
title">Artist - {{imageList[i].artist}}</h5>
                            <h5 class="card-
title">Buyer - {{imageList[i].buyer}}</h5>
                        </div>
                    </div>
                </div>
            </div>
            </div>
        </div>
  </div>
</div>
```

Art Gallery Modeling – Coding Challenge
Cirrus Logic
Akash Khiste

## Structure of the firebase database: -

Following is the structure of the database used to store the items in firebase database.



```
-M2aP7OEK3fUnk93Ig0R
    artist: "Kevin"
    buyer: "Diana"
    imageUrl: "https://firebasestorage.googleapis.com/v0/b/ima
-M2aPtIYd7xCiqC26DDg
    artist: "Danial"
    buyer: "Jack"
    imageUrl: "https://firebasestorage.googleapis.com/v0/b/ima
-M2d7VO7RYH_7Bj6eJ72
    artist: "Jenna"
    buyer: "Frank"
    imageUrl: "https://firebasestorage.googleapis.com/v0/b/ima
```

## References : -

1) Reference for angularfire2 package - https://github.com/angular/angularfire
2) Firebase - https://firebase.google.com/docs/storage/web/download-files
3) Angular - https://angular.io/docs
4) Bootstrap - https://getbootstrap.com/docs/4.4/getting-started/introduction/