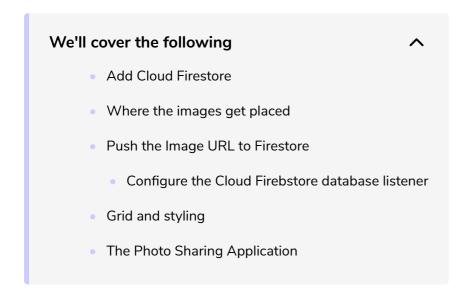
### Showing Images in the Browser

In this lesson, you will learn how to show images you uploaded to Firebase Storage in your app. After each upload, we will take the download URL and place it in Cloud Firestore. Then we will make a database listener for that collection. That is the final step that allows us to show the images to the user in the browser.



#### Add Cloud Firestore #

We need a way to keep track of files we want to show the user so we will use the Cloud Firestore database for this. Add the CDN script to the head of your HTML file.

**HTML** 

We make a div that holds all the images. We will inject are images into this div soon.

# Push the Image URL to Firestore #

When files or images get uploaded to Firebase Storage, we will store a reference to them in Cloud Firestore.

We will insert the code inside .then() of the getDownloadURL().

```
// Every time we upload a image we also need to add a reference to the database
firebase.firestore().collection('images').add({
   url: url
})
.then(success => console.log(success))
.catch(error => console.log(error))
JavaScript
```

#### Configure the Cloud Firebstore database listener #

We use a Firebase query to make a real-time listener of the **images** collection. We will get a snapshot back to which we attach a **forEach** loop. Inside **forEach**, we create **img** element dynamically with JavaScript using data coming from Cloud Firestore as the source of the image.

**JavaScript** 

### Grid and styling #

The images will possibly be different sizes if uploaded by many users so lets put them in a <code>grid</code> and set <code>width = 100%</code>. This will ensure that their width will constrain to the grid giving us some consistency and keep our application looking good.

```
img{
    width: 100%;
}

#images{
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
    grid-gap: 30px;
    margin-bottom: 25px;
}

#images > div {
    background-color: #EFEFEF;
    padding: 10px;
    border-radius: 10px;
}
```

# The Photo Sharing Application #

Press the **Run** button to see how our real-time listener shows every uploaded image. Add a few new images to see it in action!

CSS

Key:	Value:
apiKev	Not Specified
authDomain	Not Specified
databaseURL	Not Specified
projectld	Not Specified
storageBucket	Not Specified
messagingSenderld	Not Specified
appld	Not Specified
Output	
JavaScript	
	HTML
CSS (SCSS)	
Choose File	
<b>&gt;</b>	

In the next lesson, I am going to show you how to remove all the images from the database by deleting the entire collection.