

# Build a Serverless Web App with Firebase

## Prerequisites

This is a **intermediate level** lab. This assumes familiarity with the Cloud Console and shell environments. Experience with Firebase will be helpful, but it is not required. Before taking this lab it is recommended that you have completed the following lab:

- [Importing Data to a Firestore Database](#)

You should also be comfortable editing files. You can use your favorite text editor (like `nano`, `vi`, etc.) or you can launch the code editor from Cloud Shell, which can be found in the top ribbon:



Once you're ready, scroll down and follow the steps below to setup your lab environment.

## Firebase Backend

Ruby sends Patrick an email:

Ruby sends Patrick some background information in an email, and they hold a meeting to work out the key activities. From this meeting they determine that he needs to:

- Configure a Firebase project.
  - Establish security policies.
  - Add the Firestore CLI to the Google Cloud project.
- Next, help Patrick accomplish these tasks.

## Task 1. Register a Firebase Application

Open an incognito window to access the Firebase Console url [Firebase Console](#).  
When requested enter the:

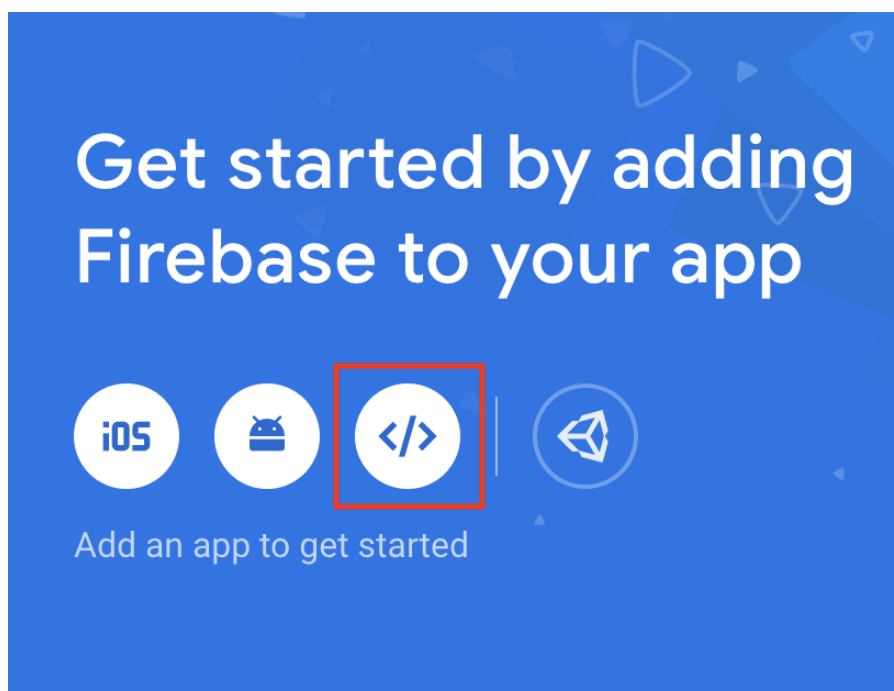
- username as `student-01-1af306466e17@qwiklabs.net`
- password as `34J2BwPozFhM`.

**Note:**

A Firebase project labelled `qwiklabs-gcp-00-3b2dbcb4861a` has been provisioned. Select this project to access Firebase products and follow this tutorial.

From the Firebase Project Overview screen, enter the following commands:

1. Select the **web icon** (highlighted below) from the list of "Get started by adding Firebase to your app" icons:



2. When prompted for an "App nickname", type in **Pet Theory**.

Pet Theory

3. Check the box next to "Also set up **Firebase Hosting** for this app".
4. Click the deploy dropdown and select `Create a new site`.
5. Amend the default to include the student prefix.

```
student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1
```

**Note:**

Ensure the site dropdown contains the student-bucket-`qwiklabs-gcp-00-3b2dbcb4861a-1` before continuing. The site domain will be based on this setting.

6. Click on the **Register app** button.
7. Click **Next > Next > Continue to console**.

**Note:**

Changing the default Firebase site domain will require changes to the local environment.

Specifically we need to:

1. Update `firebase.json` to include the custom site ID. [Set up deploy targets for your sites](#)
2. Use the site ID when using the command `firebase deploy --only hosting:student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1`

You have now configured the Firebase application.

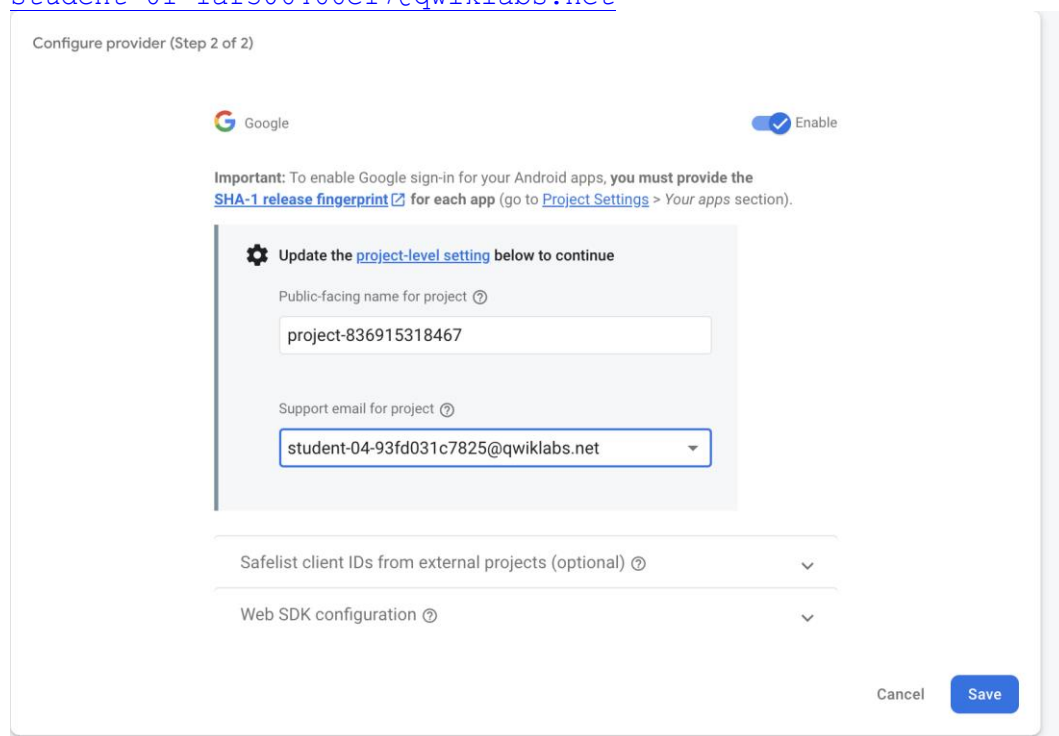
## Task 2. Enable Firebase Products

### Firebase Authentication

In the Firebase Console we will setup Firebase Authentication.

1. Click on the **Build** dropdown button in the left-hand navigation panel.
2. Select **Authentication** tile and then click on **Get Started**:
3. Click on **Sign-in method** tab and then, click on the **Google** item.
4. Click the **enable** toggle in the top right corner and for the **Support email for project** select your lab account from the drop down list.

[student-01-1af306466e17@qwiklabs.net](mailto:student-01-1af306466e17@qwiklabs.net)



The screenshot shows the 'Configure provider (Step 2 of 2)' screen for Google Authentication in the Firebase Console. At the top, the Google logo is on the left and an 'Enable' toggle switch is on the right, which is currently turned on. Below this, an important note states: 'Important: To enable Google sign-in for your Android apps, you must provide the SHA-1 release fingerprint for each app (go to Project Settings > Your apps section)'. A gear icon indicates that project-level settings must be updated. The 'Public-facing name for project' field contains 'project-836915318467'. The 'Support email for project' dropdown menu is open, showing 'student-04-93fd031c7825@qwiklabs.net' as the selected option. At the bottom, there are two expandable sections: 'Safelist client IDs from external projects (optional)' and 'Web SDK configuration'. 'Cancel' and 'Save' buttons are located at the bottom right.

5. Once you have verified the above, click on the **Save** button.

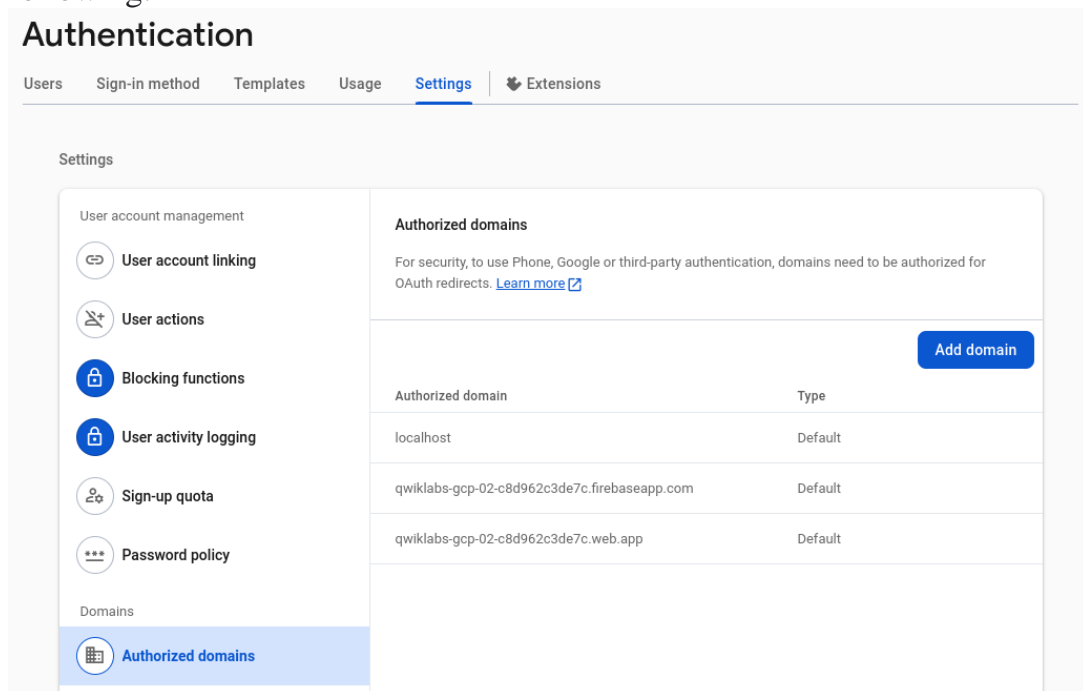
#### Note:

When using a custom domain with Firebase, you will also need to amend the Authorized domains setting.

6. Click the **Settings** tab

7. Under the **Domains** heading, click the **Authorized domains** menu item

Your page should now resemble the following:



8. Click the **Add domain** button

9. Enter the following domain:

`student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1.web.app`

**Note:**

Adding the custom domain will allow the Firebase OAuth flow to operate correctly.

10. Click the **Add** button

## Firestore

In the Firebase Console we will setup Firebase Authentication.

1. Click on the **Build** dropdown button in the left-hand navigation panel.
2. Select **Firestore Database** tile and then click on **Create database**:
3. Accept the default settings and click **Next**
4. Click **Create** to provision Cloud Firestore
5. Click the **Rules** tab

### Note:

We want to update the security rules for the Firestore database to use authentication. In the new rules, we allow read/write on the database if the user is authenticated.

6. Update the rules as follows:

```
7. rules_version = '2';
8. service cloud.firestore {
9.   match /databases/{database}/documents {
10.    match /customers/{email} {
11.      allow read, write: if request.auth.token.email == email;
12.    }
13.    match /customers/{email}/{document=**} {
14.      allow read, write: if request.auth.token.email == email;
15.    }
16.  }
```

You have now set up Firebase Authentication and Firestore with a custom site identifier. The backend configuration is now complete.

# Firebase Localhost

## Task 3. Install the Firebase CLI

Use the IDE to connect to Firebase and deploy your application.

**Note:**

The development environment has been preconfigured with Firebase Tools.

1. Copy the **IDE** link from the Lab Details panel

```
https://ide-service-ksdncxciwa-ez.a.run.app
```

2. Paste the link into a new incognito browser tab to open Cloud Code.
3. Open a Terminal by Clicking the **Application menu** (≡) > **Terminal** > **New terminal**.
4. Clone the GitHub repository from the command line:

```
git clone https://github.com/rosera/pet-theory.git
```

5. In the left panel, click the **Explorer** icon, and then **Open Folder** > **pet-theory** > **lab02**. Click **OK**.

**Note:**

If you get pop-up Do you trust the authors of the files in this folder? then check the box and click **Yes, I trust the authors**.

6. Open a terminal again by clicking the **Application menu** (≡) > **Terminal** > **New terminal**.
7. Update the node packages:

```
npm i
```

### Example Output:

```
added 630 packages, and audited 631 packages in 42s

69 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New minor version of npm available! 10.7.0 -> 10.8.3
npm notice Changelog:
https://github.com/npm/cli/releases/tag/v10.8.3
npm notice To update run: npm install -g npm@10.8.3
npm notice
npm warn using --force Recommended protections disabled.

up to date, audited 631 packages in 2s

69 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

You are now ready to link the application to the backend Firebase project.

## Task 4. Authorize Firebase Access

In the IDE connect Firebase and deploy your application.

1. Type the following command to authorize Firebase project access:

```
firebase login --no-localhost
```

### Note:

Firebase will attempt to authorize the local environment to connect with the Firebase project. Therefore ensure browser activity is performed in an incognito window to facilitate this action with the appropriate credentials.

2. Enter in **Y** if asked if Firebase can collect error reporting information and press **Enter**.
3. **Copy and paste the URL** generated into a new **incognito browser tab** then press **Enter** (directly clicking on the link results in an error).



4. Select your lab account then click **Allow**.
5. Click on **Yes, I just ran this command** to proceed
- 6 Confirm your session ID by clicking **Yes, this is my session ID**.
7. You will then be given an access code:
8. Copy the access code then paste it in the Cloud Shell prompt **Enter authorization code:**, and press **Enter**.

You should receive output similar to the following response:

### Example Output:

```
✓ Success! Logged in as student-02-21ab4a7ce0d1@qwiklabs.net
```

#### Note:

At this point the localhost is linked to the backend Firebase project.

Firebase is now connected and authorized. The next step is to initialize the Firebase products to be used.

## Task 5. Initialize Firebase Products

In the IDE let Firebase know which products are required.

1. Initialize a new Firebase project in your current working directory:

```
firebase init
```

#### Note:

Running this command will step you through setting up your project directory and Firebase products. You will be asked to select the Firebase CLI features you want set up in this folder. Use the arrow keys to move up and down the list. Use the spacebar to select the product option.

2. We need the following products:

- Firestore
- Hosting

3. Use the arrow keys and the spacebar to select **Firestore** and **Hosting**. Ensure your shell matches the following and then hit **Enter**:

```
4. ? Which Firebase CLI features do you want to set up for this
   folder? Press Space to select features, then Enter to confirm
   your choices.
5.   ○ Realtime Database: Configure a security rules file for
   Realtime Database and (optionally) provision default insta
6.   ● Firestore: Configure security rules and indexes files for
   Firestore
7.   ○ Functions: Configure a Cloud Functions directory and its files
8.   >● Hosting: Configure files for Firebase Hosting and (optionally)
   set up GitHub Action deploys
9.   ○ Hosting: Set up GitHub Action deploys
   ○ Storage: Configure a security rules file for Cloud Storage
```

10. Run through the rest of the steps to configure Firebase:

- Key down to **Use an existing project** and press **Enter**.
- Select your Project ID from the list `qwiklabs-gcp-00-3b2dbcb4861a` then **Enter**.
- Press **Enter** and then **N** to keep your `firestore.rules` file.
- Press **Enter** and then **N** to keep your `firestore.indexes.json` file.
- Press **Enter** to keep your public directory and then **N** to disallow rewrites to your `/index.html` file.
- Press **Enter** to Set up automatic builds and deploys with GitHub? and press **N**.
- Enter in **N** when prompted to overwrite your `404.html` file.
- Enter in **N** when prompted to overwrite your `index.html` file.

**Example Output:**

```
✓ Wrote public/404.html
✓ Wrote public/index.html

i Writing configuration info to firebase.json...
i Writing project information to .firebase...
i Writing gitignore file to .gitignore...

✓ Firebase initialization complete!
```

The local configuration is now complete.

## Task 6. Deploying to Firebase

Continue in the Terminal for this step. Ensure you are still in the **pet-theory/lab02** folder.

1. Edit `firebase.json` and update the hosting section with the **site**

### Expected Output

```
{
  ...
  "hosting": {
    "site": "student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1",
    ...
  }
}
```

2. Deploy your firebase application, using your site ID value e.g.:

```
firebase deploy --only hosting:student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1
```

Output:

```
✓ Deploy complete!

Project Console:
https://console.firebase.google.com/project/project_0/project_id|
"PROJECT_ID"/overview
Hosting URL: https://student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1.web.app
```

3. Copy the hosting URL (should resemble `qwiklabs-gcp-00-3b2dbcb4861a.web.app`) and open it in a new incognito tab.

```
https://student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1.web.app
```

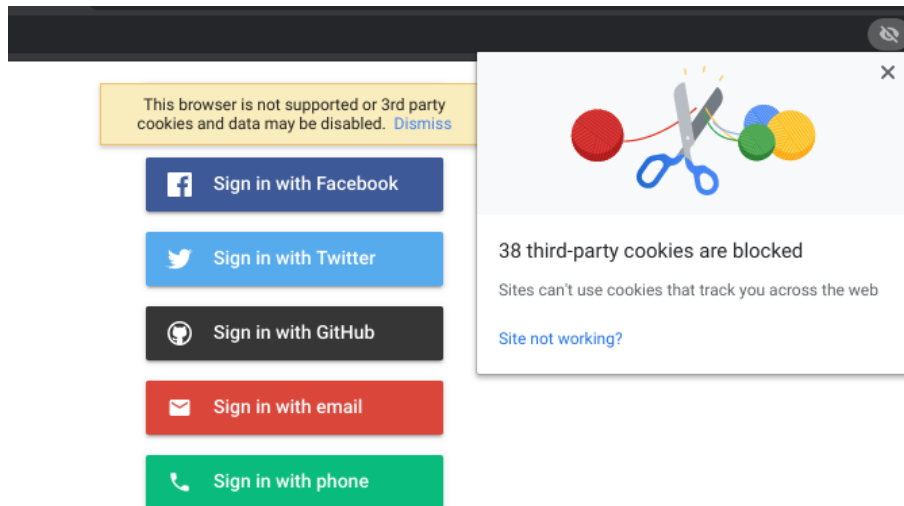
### Important:

Personal credentials should not be used as authentication in this lab. Please use account: `student-01-1af306466e17@qwiklabs.net` to perform the Google Sign in!

4. Click on the **Sign in with Google** button:

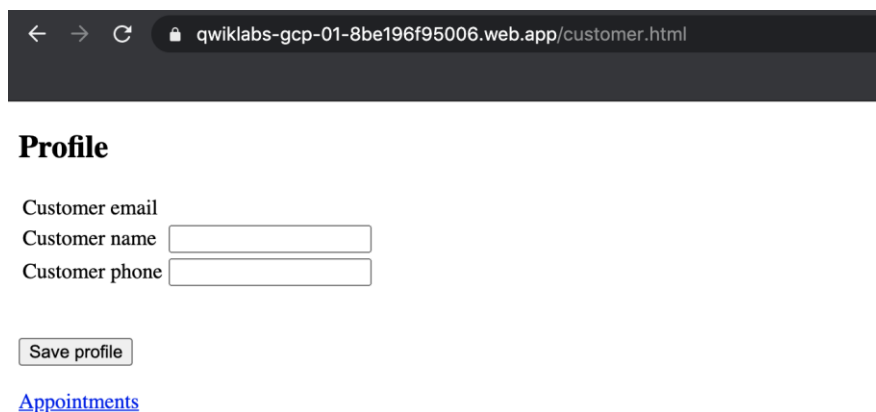
**Note:**

You see a **browser is not supported or 3rd party cookies and data may be disabled** error, make sure to enable cookies in your browser.



This can be done in Chrome by clicking the **eye** icon at the far right of the url tab and following the blue links in the popup. Click on the **site not working?** link to update your browser settings to accept cookies.

5. Login with your username provided i.e. `student-01-1af306466e17@qwiklabs.net`. The following page opens:



A small company like Pet Theory doesn't have the resources or requisite skill set to do this. In this instance it can be beneficial to let application users log in with their existing Google account (or any other identity providers)!

**Note:**

Managing passwords is a difficult task and could expose your company to additional risk. Also, users don't want to create yet another user id and password.

You have now deployed code to let users use Google authentication to access the appointments app.

## Task 7. Add a customer page to your web app

Return to the Terminal and use the editor to view the files in the **public** folder.

1. Open the `public/customer.js` file and copy and paste the following code:

```
2. let user;
3.
4. firebase.auth().onAuthStateChanged(function(newUser) {
5.   user = newUser;
6.   if (user) {
7.     const db = firebase.firestore();
8.
9.     db.collection("customers").doc(user.email).onSnapshot(function(doc) {
10.      const cust = doc.data();
11.      if (cust) {
12.        document.getElementById('customerName').setAttribute('value',
13.          cust.name);
14.        document.getElementById('customerPhone').setAttribute('value',
15.          cust.phone);
16.      }
17.      document.getElementById('customerEmail').innerText =
18.        user.email;
19.    });
20.  });
21.
22. document.getElementById('saveProfile').addEventListener('click',
23.   function(ev) {
24.     const db = firebase.firestore();
25.     var docRef = db.collection('customers').doc(user.email);
26.     docRef.set({
27.       name: document.getElementById('customerName').value,
28.       email: user.email,
29.       phone: document.getElementById('customerPhone').value,
30.     })
31.   })
32. }
```

27. })

28. Open the `public/styles.css` file and paste in the following code:

```
29. body { background: #ECEFF1; color: rgba(0,0,0,0.87); font-family: Roboto, Helvetica, Arial, sans-serif; margin: 0; padding: 0; }
30. #message { background: white; max-width: 360px; margin: 100px auto 16px; padding: 32px 24px 16px; border-radius: 3px; }
31. #message h3 { color: #888; font-weight: normal; font-size: 16px; margin: 16px 0 12px; }
32. #message h2 { color: #ffa100; font-weight: bold; font-size: 16px; margin: 0 0 8px; }
33. #message h1 { font-size: 22px; font-weight: 300; color: rgba(0,0,0,0.6); margin: 0 0 16px; }
34. #message p { line-height: 140%; margin: 16px 0 24px; font-size: 14px; }
35. #message a { display: block; text-align: center; background: #039be5; text-transform: uppercase; text-decoration: none; color: white; padding: 16px; border-radius: 4px; }
36. #message, #message a { box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24); }
37. #load { color: rgba(0,0,0,0.4); text-align: center; font-size: 13px; }
38. @media (max-width: 600px) {
39.   body, #message { margin-top: 0; background: white; box-shadow: none; }
40.   body { border-top: 16px solid #ffa100; }
}
```

41. From the Terminal command line run the following command:

```
firebase deploy --only hosting:student-bucket-qwiklabs-gcp-00-3b2dbcb4861a-1
```

Output:

```
✓ Deploy complete!

Project Console:
https://console.firebase.google.com/project/project_0.project_id|
"PROJECT_ID"/overview
Hosting URL: https://student-bucket-{{ project_0.project_id|
"site" }}-1.web.app
```

42. Go to your application tab and **hard refresh the page with CMND+SHIFT+R (Mac) or CTRL+SHIFT+R (Windows)**. Simple refreshing will not display the needed updates.

The screenshot shows a web form titled "Profile" in orange text. It contains three input fields: "Customer email" with the value "student-00-2a82d9eefcb4@qwiklabs.net", "Customer name" which is empty, and "Customer phone" which is empty. Below the inputs is a "Save profile" button. At the bottom of the form is a large blue button labeled "APPOINTMENTS".

Customer email	student-00-2a82d9eefcb4@qwiklabs.net
Customer name	<input type="text"/>
Customer phone	<input type="text"/>

43. Enter some customer info—make up a name and phone number

**Customer name:**

John

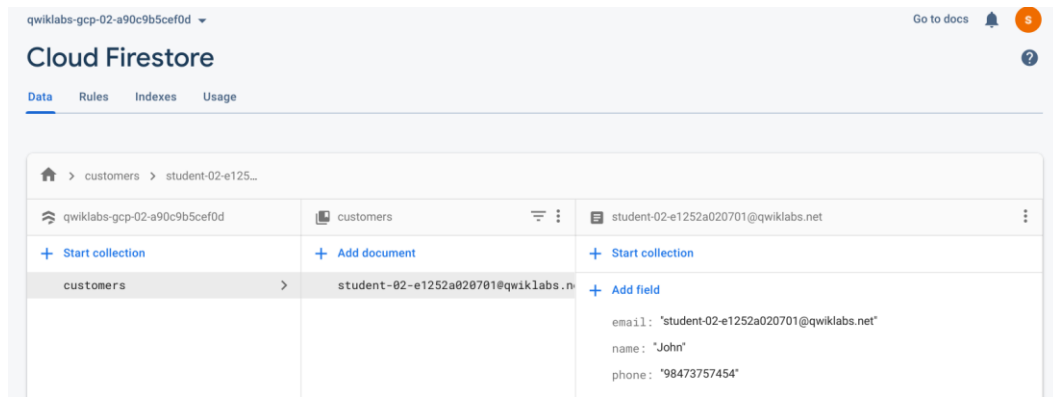
**Customer phone:**

98473757454

44. Click **Save profile**.

45. Return to the [Firebase Console](#)

46. Click **Build > Firestore Database** to view the profile information saved:



47. Return to the web app page and click on the **Appointments** link. You will see a blank page since it has not deployed the appointments code yet.