Following these instructions should allow the User to create the necessary database features and static URL to confirure RISSPriv. Admin

Helpful Commands

In Finder - Show hidden files
Press **Command + Shift + .**
(That's Command + Shift + period)
This will show all hidden files (files starting with a dot).

.firebaserc is hidden by default.

✅ **Where is .firebaserc?**
It's a hidden file located in the root of any Firebase-initialized project directory.
If you ran firebase init from your **home directory**, then it's at:
/Users/alohaedo/.firebaserc
You can **delete it** with this command in Terminal:
rm ~/.firebaserc

This will try to uninstall any existing Homebrew.
Run: /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/uninstall.sh)"

This will exit any command when using Terminal:
Control C (^C)

RISSPriv.Admin is designed to work with Google Firebase, a Google account is required to create a Project in Firebase

✅ **PART 1: Creating a Firebase Project with RTDB & Email Authentication (Approval-based)**

🔷 **A. Log into the Google account of your choice (a Google account is required to create a Firebase Project)**

🔷 **B. Create the Firebase Project**

1. Go to https://console.firebase.google.com
2. Click **"Create a Firebase project"**
3. Name your project, suggest the group or location name for which you will be the **Privileged Administrator**
4. **<Continue>**
5. **Disable "**Enable Gemini in Firebase"
6. **<Continue>**
7. **Disable** "Enable Google Analytics for this project"
8. **<Continue>**
9. Preparing your project please wait…
10. Your Firebase project is ready
11. **<Continue>**
12. **Register App**
    1. Above "Add an app to get started", **Click** "ios+"
    2. For Apple bundle ID **input** com.ed.RISSPrivAdmin
    3. For App nickname **input** RISSPrivAdmin

4. For App Store ID **input** TBDXXXXXX
5. **<Register app>**

13. **Download config file**
    1. **Select** <Download GoogleService-Info.plist>
        1. **Copy Down the following:**
        2. GOOGLE_APP_ID: Example "1:926543173077:ios:30c949a16c653c0425d95c"
        3. GCM_SENDER_ID: Example "926543173077"
        4. API_KEY: Example "AIzaSyCV9Xa5HIBidlO-U6NqLa6578fiI_JVYAo"
        5. PROJECT_ID: Example “temp-14441”
    2. **Select** <Next>
14. **Add Firebase SDK**
    1. **Select** <Next>, nothing else
15. **Add initialization code**
    1. **Select** <Next>, nothing else
16. **Next Steps**
    1. <Continue to console>
17. On the left panel **Click** “Build”
18. **Select** “Authentication”
19. **Select** “Get Started”
20. **Select** “Email/Password”
21. **Enable** “Email/Password”
22. **<Save>** Note: a verification mail is sent those who create accounts to assure that the person uses the email name owns the email name.
23. On the left side panel, **Select** “Build”, again
24. **Select** “Realtime Database” – NOT Firestore Database
25. **Click** “Create Database”
26. **Under Set up database**
    1. **Select** “United States (us-cental1)”
    2. <**Next**>
27. **Select** “Start in Locked”
    1. <**Enable**>
28. Goes to Realtime Database in your project, **Select** second tab, “Rules”
    1. Copy the text below and paste into “Rules” :

```
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null"
  }
}
```

    2. **Select** <Publish>
29. Dismiss Alert
30. **Select** first tab, “Data”
31. At the end of https://projectname / :null**, Select** “+”
    1. **Type** in “PrivAdminCodes”, the **Select** “+”
    2. **Type** in your name, id, whatever, then for value, **Type** the code you will use as a Privileged Administrator, and **Select** “ABC” (for string), then **Press** “Return”

Your database and authentication are set up, everything else can be controlled from the App.

RISS requires access to a static web page, that stores the information that links your RISS app to your Realtime Database Project. The below is one way of creating a static webpage for this purpose at no cost.

✅ **PART 2: Setup Firebase Hosting to Store JSON Config**
Information: https://firebase.google.com/docs/hosting/quickstart

- **Homebrew** is a popular package manager for macOS (and Linux).

- **When you install Homebrew on Apple Silicon Macs, it's installed in /opt/homebrew/ instead of the traditional /usr/local/.**
- **Your shell (like zsh or bash) needs to know where to find the brew command.**
- **This is done by adding Homebrew's binary path to the PATH environment variable.**
- **The eval "$(/opt/homebrew/bin/brew shellenv)" command sets up the environment variables Homebrew needs (including adding its bin directory to PATH).**

https://brew.sh/
https://github.com/Homebrew/brew/releases/tag/4.5.8
Homebrew-4.5.8.pkg

After reinstalling Homebrew, Homebrew package installer will print:
To add Homebrew to your PATH, add to your shell profile (e.g. ~/.bash_profile or ~/.zprofile): **-** eval "$(/opt/homebrew/bin/brew shellenv)"

✅ **Here's exactly how to do it:**

1. Open Terminal and run this command to open .zprofile in a text editor:
nano ~/.zprofile

2. Add this line to the file (or make sure it's the only one like it):
eval "$(/opt/homebrew/bin/brew shellenv)"
Why?
Adding this line ensures that when you open a terminal, your shell knows where to find Homebrew and its installed commands, so you can run brew and other tools without errors.

3. Save and exit:
- Press Control + O (to write the file),
- Press Enter (to confirm),
- Then press Control + X (to exit).

4. Apply it to your current Terminal session immediately:
eval "$(/opt/homebrew/bin/brew shellenv)"

🔄 **After that:**
Try:
brew doctor

and:
firebase –version

You should see responses that indicates a Version or that your computer is Ready to Brew.
If not, stop and consult AI, cut and paste your terminal info into AI and it should be able to find out what needs correction.

Use Homebrew to install Node.js:
brew install node

Then install Firebase CLI via npm:
brew install firebase-cli

Verify installations
npm -v
firebase –version

Check that Homebrew works:
brew doctor
You should see something like:
Your system is ready to brew.
You should see responses that indicates a Version or that your computer is Ready to Brew.
If not, stop and consult AI, cut and paste your terminal info into AI and it should be able to find out what needs correction.

✅ **Fix Global npm Permissions the Right Way**
This changes ownership of your .npm cache folder back to your user.
sudo chown -R $(id -u):$(id -g) ~/.npm

✅ **Step-by-Step: Host a .json file with Firebase Hosting**
Change Directory to your Project Directory
cd ~/Public/RISS/RISSA

---

2. Initialize Firebase Hosting
firebase init hosting
When prompted:

- **Select Firebase project** → Choose one or create a new one.
- **Public directory** → Enter public
- **Configure as a single-page app?** → NO
- **Set up automatic builds/deploys with GitHub?** → NO
- Confirm overwrite of index.html → NO

👉 **Optional, if run previously the above will link to a default Project, to choose a different Project other than the default, run:**
firebase use --add

- **This will let you select the correct Firebase project from your Firebase Console**
- **You can give it an alias (like rissa, site2, etc.)**
- **It updates .firebaserc in the current folder**

---

3. Clear the Users/Username/public/ folder except for your configuration.json file.

- **It should look like the below and has the configuration information from Part 1. If you have trouble making a .json file, suggest using AI.**

```
"RISSB": {
  "GOOGLE_APP_ID": "1:926543173077:ios:30c949a16c653c0425d95c",
  "GCM_SENDER_ID": "926543173077",
  "API_KEY": "AIzaSyCV9Xa5HIBidlO-U6NqLa6578fiI_JVYAo",
  "PROJECT_ID": "riss-elementary",
  "DATABASE_URL": "https://riss-elementary-default-rtdb.firebaseio.com/"
}
}
```

- RISSB – **name of the Configuration as defined by the User (you), this can be any name including those of a Project, Group, or Location, all caps, text only, no special characters.**
- GOOGLE_APP_ID, GCM_SENDER_ID, API_KEY, PROJECT_ID **are from Par1 1.**
- DATABASE_URL – **this is the https from the Realtime Database on your Project, go to the Firebase console, select your Project, in your Project select the Realtime Database, and on the right you will see an https to the Realtime Database.**

---

4. Deploy to Firebase Hosting
firebase deploy

After a few seconds, it will output a URL like:
Hosting URL: https://your-project-id.web.app

Your JSON file will be available at:
https://your-project-id.web.app/yourjson.json

✅ **Done!**