## OVERVIEW

This system receives SPaT messages from a traffic controller, extracts the payload, uploads it to Firebase Realtime Database, and forwards it in real time to another Python process via a UDP socket.

## Components

| Component | Role |
| --- | --- |
| Sender.py | Receives SPaT messages over UDP, extracts payload, and sends it to Firebase |
| listener.js | Listens to Firebase updates and forwards the latest SPaT to Python via UDP |
| receiver.py | Listens for SPaT messages on a UDP socket and processes or logs them |

## System-Wide Tools

### Step 1: Firebase Setup

1. Go to [https://console.firebase.google.com](https://console.firebase.google.com/)
2. Create a Firebase project (e.g., c-vision)
3. Enable Realtime Database
   * Go to Build > Realtime Database
   * Click Create Database → start in test mode
4. Go to Project Settings > Service Accounts
   * Click “Generate new private key”
   * Save the JSON file as cvision-firebase-key.json

### Step 2: Install Required Tools / Dependencies

#### 🟢 Initialize Node.js Project on Your Computer :

sudo apt update

sudo apt install npm

npm init -y

#### 🔵 Install Firebase Admin SDK:

Upgrade Node.js if your Node.js version is too old (< v18), upgrade using nvm:

node -v # should be v18+ or v20+

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash

source ~/.bashrc

nvm install 20

nvm use 20

node -v # should now show v20.x

📦 *Install Required Packages your project directory:*

npm install firebase-admin # For Node.js (listener.js)

pip3 install firebase-admin # For Python (sender.py)

⚠️ *If encounter errors in Node.js (e.g., EEXIST, symlink issues):*

rm -rf node\_modules package-lock.json

npm cache clean --force

npm install firebase-admin

## Step 3: After Setup

You can now run:

* python3 v2x-data-receiver.py → starts UDP listener
* node listener.js → listens to Firebase, forwards via UDP
* python3 send-v2x-data.py → receives V2X, pushes to Firebase

## Optional Step: Network / Firewall Notes

* Make sure port 50002 (SPaT input) is reachable on the network interface for sender.py
* Ports 5005 (local UDP for listener.js → receiver.py) must be open locally
* Use ufw allow if needed:

sudo ufw allow 50002/udp

sudo ufw allow 5005/udp

**Starting a New Project on Google Firebase from Scratch**

**Prerequisites (Confirm You Have These)**

* Google Account: You need a Google account with billing enabled for the associated Google Cloud project (even free tier usage might require billing enabled for Cloud Functions).
* Node.js & npm: Installed on your system (Node.js v18 or v20 recommended).
* Firebase CLI: Installed globally (npm install -g firebase-tools). Make sure you are logged in (firebase login).
* Python: Installed on your system (Python 3.10, 3.11, or 3.12 recommended, as these are supported by Cloud Functions).
* pip: Python's package installer (usually comes with Python).

**Step 1: Create a New Firebase Project in the Console**

1. Go to the **Firebase Console**: [https://console.firebase.google.com/](https://console.firebase.google.com/?authuser=1)
2. Click **"Add project"** or **"Create a project"**.
3. **Project Name:** Enter a new, unique name (e.g., spat-data-exchange-v2 or spat-proj-new). Let's use **spat-proj-new** for this guide.
4. **Project ID:** A unique ID will be generated (e.g., spat-proj-new-XXXXX). Make a note of this.
5. **Google Analytics:** Choose whether to enable or disable Google Analytics. It's optional for this project.
6. Click **"Create project"**. Wait for the project to provision.

Okay, let's do a complete reset and start from scratch. This will ensure we have a clean slate both locally and in your Firebase project.

**Warning:** Deleting a Firebase project is permanent and will remove all associated data (Realtime Database, Cloud Functions, Storage, Hosting, etc.). Make sure you want to delete spat-data-exchange before proceeding.

**Part 1: Deleting the Old Project**

**1. Delete the Project Locally**

1. **Close any open terminals** that are in your project directory.
2. **Navigate to the parent directory** of your cvision folder. In your case, it would be: C:\Users\ddas\Documents\debashis-workspace\src
3. **Delete the entire cvision folder.**
   * **Using Command Prompt:**

DOS

cd C:\Users\ddas\Documents\debashis-workspace\src

rmdir /s /q cvision

* + **Using File Explorer:** Right-click on the cvision folder and select "Delete".

**2. Delete the Project from Firebase**

1. Go to the **Firebase Console**: [https://console.firebase.google.com/](https://console.firebase.google.com/?authuser=1)
2. In the project selector dropdown (usually at the top left), select your project: **spat-data-exchange**.
3. Once the project dashboard loads, click on the **"Project settings" gear icon** (⚙️) next to "Project Overview" in the left-hand navigation.
4. Scroll down to the **"Your projects" card** at the bottom.
5. Click the **"Delete project" button**.
6. You will be prompted to confirm. **Read the warnings carefully**, then check all the boxes and enter your project ID (spat-data-exchange) to confirm deletion.
7. Click **"Delete project"** again.

**Part 2: Starting a New Project From Scratch**

Now that everything is clean, we'll set up a new project.

**Prerequisites (Confirm You Have These)**

* **Google Account:** You need a Google account with billing enabled for the associated Google Cloud project (even free tier usage might require billing enabled for Cloud Functions).
* **Node.js & npm:** Installed on your system (Node.js v18 or v20 recommended).
* **Firebase CLI:** Installed globally (npm install -g firebase-tools). Make sure you are logged in (firebase login).
* **Python:** Installed on your system (Python 3.10, 3.11, or 3.12 recommended, as these are supported by Cloud Functions).
* **pip:** Python's package installer (usually comes with Python).

**Step 1: Create a New Firebase Project in the Console**

1. Go to the **Firebase Console**: [https://console.firebase.google.com/](https://console.firebase.google.com/?authuser=1)
2. Click **"Add project"** or **"Create a project"**.
3. **Project Name:** Enter a new, unique name (e.g., V2X-Data-Exchange).
4. **Project ID:** A unique ID will be generated (e.g., V2X-Data-Exchange -XXXXX). Make a note of this.
5. **Google Analytics:** Choose whether to enable or disable Google Analytics. It's optional for this project.
6. Click **"Create project"**. Wait for the project to provision.

**Step 2: Set up Firebase Realtime Database:**

1. In your Firebase project console, navigate to "Build" > "Realtime Database."
2. Click "Create database."
3. Choose a location (usually closest to your users/devices).
4. Select **cc** for now. IMPORTANT: For production, you will need to secure your database with [Firebase Security Rules](https://firebase.google.com/docs/database/security?authuser=1) to prevent unauthorized access. This is critical for data integrity and security.

**Step 3: Initialize Firebase Project Locally**

1. **Open your terminal.**
2. **Navigate to where you want to create your new project folder.** Let's assume you want it directly in C:\Users\ddas\Documents\debashis-workspace\src\firebase-project-v2x-data-exchange.

cd C:\Users\ddas\Documents\debashis-workspace\src

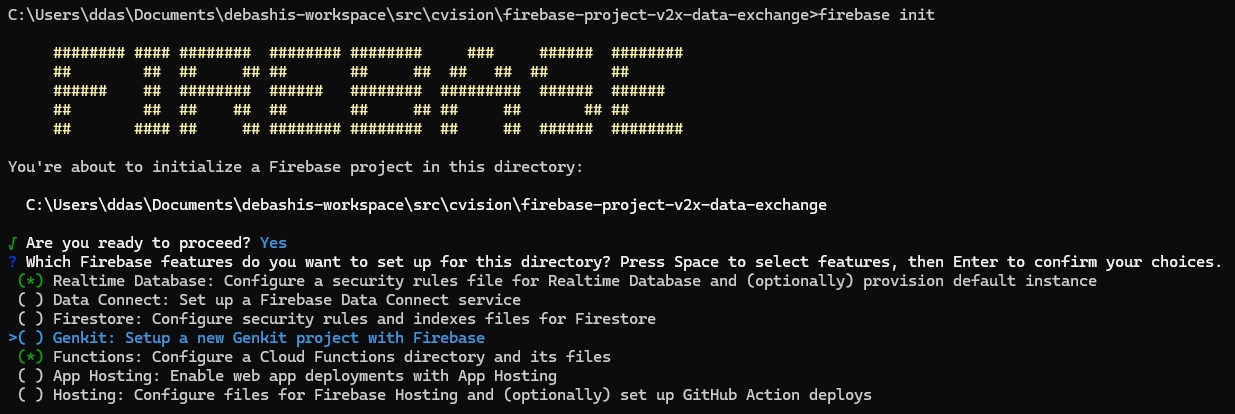
1. **Initialize a new Firebase project:**

firebase init

* + **"Are you ready to proceed? (Y/n)"**: Type Y and press Enter.
  + **"Which Firebase features do you want to set up for this directory?"**: Use the **spacebar** to select Functions and Realtime Database. Press Enter.

Text

AI-generated content may be incorrect.



* + **"Please select a project:"**: Choose **V2X-Data-Exchange** and Press Enter.
  + **"What language would you like to use for Cloud Functions?"**: Select Python. Press Enter.
  + **"Do you want to install dependencies with pip now?"**: Type Y and press Enter. (This will create a venv and requirements.txt.)
  + If system fails here follow the following:
    1. Navigate to your new project's functions directory

cd C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange\functions>

* + 1. Activate the newly created virtual environment

.\venv\scripts\activate

* + 1. Manually Upgrade pip (as suggested by the error)

python.exe -m pip install --upgrade pip

* + 1. Manually Install Python Dependencies

pip install -r requirements.txt

* + 1. Execute firebase init once again from your **project's root directory** and follow rest of the steps.
  + **"File functions/main.py already exists. Overwrite? (y/N)"**: Type N and press Enter (we'll provide the updated main.py code shortly).
  + **"File functions/requirements.txt already exists. Overwrite? (y/N)"**: Type N and press Enter (we'll provide the updated requirements.txt code shortly).
  + **"What file would you like to use as Realtime Database Rules?"**: Press Enter to accept the default (database.rules.json).
  + **"Firebase initialization complete!"**



**Step 4: Configure firebase.json for Python Runtime**

1. **Open firebase.json** located at C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange\firebase.json.
2. **Ensure its content matches this exactly**, paying close attention to the runtime line:

{

  "functions": [

    {

      "source": "functions",

      "codebase": "default",

      "ignore": [

        "venv",

        ".git",

        "firebase-debug.log",

        "firebase-debug.\*.log",

        "\*.local"

      ],

      "runtime": "python312"

    }

  ],

  "database": {

    "rules": "database.rules.json"

  },

  "hosting": {

    "public": "public",

    "ignore": [

      "firebase.json",

      "\*\*/.\*",

      "\*\*/node\_modules/\*\*"

    ],

    "rewrites": [

      {

        "source": "\*\*",

        "destination": "/index.html"

      }

    ]

  }

}

1. Save the firebase.json file.

**Step 5: Create main.py and requirements.txt for Functions**

These files contain your Cloud Function code and its Python dependencies.

1. **Open** C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange\functions\main.py and **replace its entire content** with required code
2. **Open** C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange\functions\requirements.txt and **replace its entire content** with the following:

firebase-admin

firebase-functions

requests

**Step 6: Install Local Node.js Dependencies**

1. **Ensure your terminal is in the functions directory** of your new project: C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange\functions
2. **Create a package.json file** (if firebase init didn't already create one or if you overwrote it). Open a text editor and update it
3. **Run npm install**:

npm install

1. This installs the Node.js tools required by Firebase CLI locally.

**Step 7: Deploy Your Cloud Functions**

1. Ensure your terminal is still in the functions directory.
2. Run the deployment command:

firebase deploy --only functions

This will upload your main.py and requirements.txt and build them on Google Cloud using the python312 runtime specified in firebase.json. To deploy Firebase Cloud Functions, your project needs to be on the Blaze (pay-as-you-go) billing plan.

**Step 8: Create/Update send\_spat\_data.py**

This script sends the encoded data to your deployed ingest\_spat\_data function.

1. **Navigate to your project's root directory:**

cd C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange

1. **Create a new file** named send\_spat\_data.py in this directory and update it with required code.

**IMPORTANT:** After firebase deploy completes in Step 6, the terminal will output the URL for your ingest\_spat\_data function. **You MUST copy that URL and paste it** into the CLOUD\_FUNCTION\_URL variable in this send\_spat\_data.py script.

**Step 10: Set up and Run Python Virtual Environment Locally**

1. **Navigate to your project's functions directory:**

cd C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange\functions

1. **Create a new virtual environment (if firebase init didn't already, or if you deleted it earlier):**

python -m venv venv

1. **Activate the virtual environment:**

.\venv\Scripts\activate

You should see (venv) at the beginning of your command prompt.

1. **Install Python dependencies for local use:**

pip install -r requirements.txt

This installs firebase-admin, firebase-functions, and requests into your local venv.

1. **Navigate back to your project root to run the sender script:**

(You should now be in C:\Users\ddas\Documents\debashis-workspace\src\cvision\firebase-project-v2x-data-exchange\ with (venv) still active).

1. **Run the sender script:**

python send\_spat\_data.py

This comprehensive process will get your entire system set up correctly from the ground up!