**What is API Key**

An \***API key**\* is a **unique identifier** used to authenticate requests made to an **API (Application Programming Interface)**. It's like a **password** or **token** that grants access to a particular service or resource. Here's a breakdown of its uses:

### **Key Uses of an API Key**:

1. \***Authentication and Authorization**\*:- The API key ensures that the user making the request has the necessary permissions to access the service. It helps authenticate the client or user making requests to the API, ensuring that only authorized users can interact with the service.

2. \***Access Control**\*:- By using API keys, API providers can control who has access to the resources, manage usage limits, and monitor activity. It helps service providers restrict or allow access to specific users, based on the key associated with the request.

**How to Generate API Keys for BigQuery & Gemini in Google Cloud Platform (GCP)**

To generate an API key for **Google Cloud Platform (GCP)** services like **BigQuery** and **Gemini**, follow these steps:

**Step 1**. **\*Sign in to Google Cloud Console\***

- Go to [Google Cloud Console](https://console.cloud.google.com/).

- Log in with your Google account.

**Step 2. \*Create a Project (if you don't have one)\***

- In the Cloud Console, click the \*project drop-down\* on the top-left side of the page.

- Select \*New Project\*.

- Name your project and click \*Create\*.

**Step 3. \*Enable the Required APIs\***

- Navigate to the \*API & Services > Library\* section.

- Search for \*BigQuery API\* and \*Gemini API\*.

- Click on the APIs and then enable them for your project.

**Step 4. \*Create an API Key\***

- In the \*API & Services > Credentials\* section, click \*Create Credentials\*.

- Select \*API Key\*.

- Your new API key will appear. You can copy it and use it to authenticate requests to BigQuery or Gemini.

**Step 5. \*Restrict the API Key (optional but recommended)\***

- Click \*Restrict Key\* after creating the API key.

- Choose the restrictions based on your use case, such as restricting it to specific IP addresses, or specific APIs like BigQuery.

**Step 6. \*Save Your API Key\***

- Keep your API key secure and do not expose it in public code repositories. You may want to store it in a secure environment variable or a secrets management tool.

**How setup Git -**

**Git Setup & GitHub Integration**

Installing Git, configuring it, setting up GitHub, and pushing your code to a repository.

**Step 1:**

* Download & Install Git
* Go to the official website: <https://git-scm.com/download/win>
* Download the Git for Windows installer.
* Run the installer and follow the prompts.

**During installation, select these options:**

* "Git Bash Here" (enables right-click to open Git Bash).
* "Use Git from Git Bash only" (under PATH environment).
* Set default editor as VS Code when prompted.

**Step 2:**

* Open Git Bash & Configure User Info
* Open Git Bash
* Search for "Git Bash" in the Start menu.

**Set your name and email:**

* bash
* Copy
* Edit
* git config --global user.name "Your Name"
* git config --global user.email "your\_email@example.com"
* Check configuration (optional):
* bash
* Copy
* Edit
* git config --global --list

**Step 3**: **Create a GitHub Account**

* Visit [https://github.com](https://github.com/)

**## Create an account if you don't have one ##**

**Step 4: Generate GitHub Token (Classic)**

* Go to Settings > Developer Settings > Personal Access Tokens > Tokens (classic).
* Click Generate new token.
* Select scopes like repo, workflow, read:org, etc.
* Copy the token and store it securely (you won’t be able to see it again).

**Step 5**: **Clone a GitHub Repository**

* bash
* Copy
* Edit
* git clone <https://github.com/ermannsriv9/Revil.git>
* cd Revil

**Step 6: Check Git Status**

* bash
* Copy
* Edit
* git status

**Step 7: Initialize & Push Project to GitHub**

* bash
* Copy
* Edit
* echo "# Revel" >> README.md # Add README file
* git init # Initialize local repo
* git add README.md # Stage file
* git commit -m "first commit" # Commit
* git branch -M main # Rename default branch to main
* git remote add origin <https://github.com/ermannsriv9/Revel.git> # Add remote
* git push -u origin main # Push to remote

**Step 8: Sync Branch & Add Files**

* bash
* Copy
* Edit
* git pull origin main # Pull latest changes
* git checkout main # Switch to main branch
* Add or update your documentation (e.g. in README.md), then:
* bash
* Copy
* Edit
* git add filename # Stage changes
* git commit -m "document for API key and Git setup" # Commit changes
* git push origin main # Push to GitHub