Google IoT Core sử dụng hai chuẩn bảo mật là **TLS1.2** cho tầng *transport* và **JWT** cho tầng *application* để mã hóa việc trao đổi thông tin. Điều này có nghĩa chúng ta sẽ cần phải tạo ra cặp **Public key** - **Private key** theo chuẩn TLS đưa ra. Đồng thời thông qua việc khởi tạo JWT ta sẽ bảo đảm an toàn hơn cho việc xác thực các cặp key giữa những thiết bị IoT với Google IoT Core mà không lo có kẻ thứ ba chiếm đoạt.

* Thiết bị IoT **(Client)** sẽ sử dụng ***Private key*** để tạo phần **signature** cho **JWT**. Mã JWT này sẽ được chuyển tới Google IoT Core để *chứng minh danh tính* cho thiết bị IoT này.
* Google IoT Core **(Cloud)** sẽ sử dụng ***Public key*** để *xác nhận danh tính* của thiết bị này.

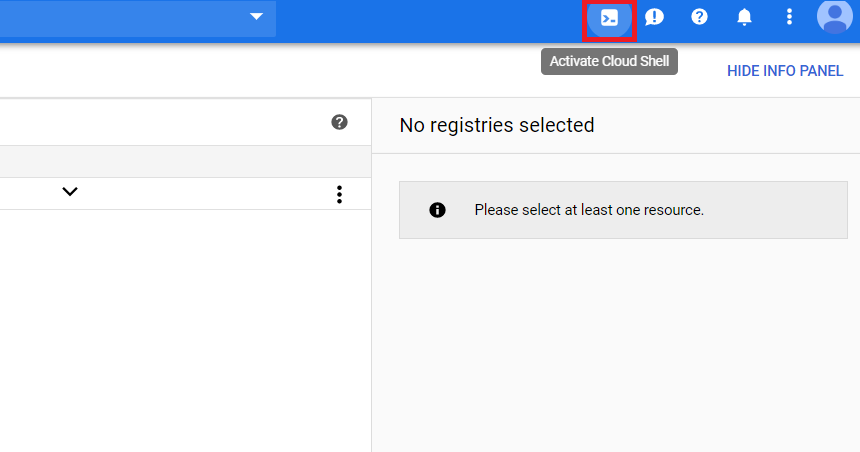
**Phần 2: Hướng dẫn tạo chứng chỉ cho thiết bị để kết nối với Google IoT Core**

**I. Hướng dẫn tạo Public/ Private Key**

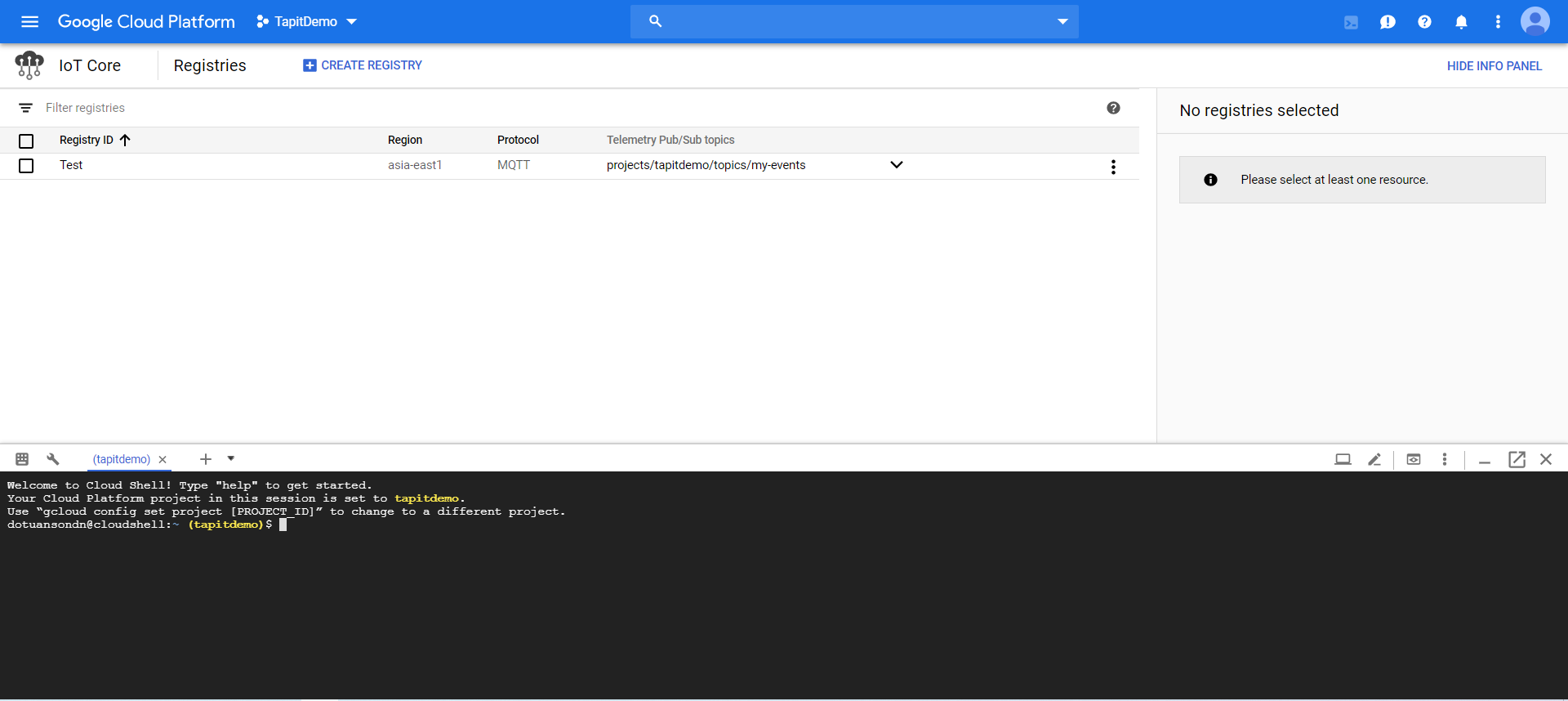
Google IoT Core hỗ trợ cho chúng ta hai thuật toán để sinh cặp public private key là RSA và Elliptic Curve.

Để tạo cặp key này chúng ta sẽ sử dụng công cụ hỗ trợ dòng lệnh gcloud mà GCP đã cung cấp cho chúng ta.

Tại giao diện Console, ở thanh công cụ phía trên ta nhấn vào biểu tượng có tên **Activate Cloud Shell** như hình bên dưới.



Giao diện Console sẽ trả về cho chúng ta cửa sổ dòng lệnh gcloud để tương tác với các nguồn tài nguyên trong GCP.



Tạo cặp key sử dụng thuật toán RSA:

openssl genpkey -algorithm RSA -out rsa\_private.pem -pkeyopt rsa\_keygen\_bits:2048

openssl rsa -in rsa\_private.pem -pubout -out rsa\_public.pem

* Các dòng lệnh này sẽ tạo cho chúng ta 2 file:
* *rsa\_private.pem*: Chứa **private key**, được lưu trữ tại thiết bị IoT và được mã hóa bởi JWT.
* *rsa\_public.pem*: Chứa **public key,** được lưu trữ tại dịch vụ Google IoT Core để xác nhận mã JWT từ thiết bị IoT gửi lên.

Tạo cặp key sử dụng thuật toán Eliptic Curve:

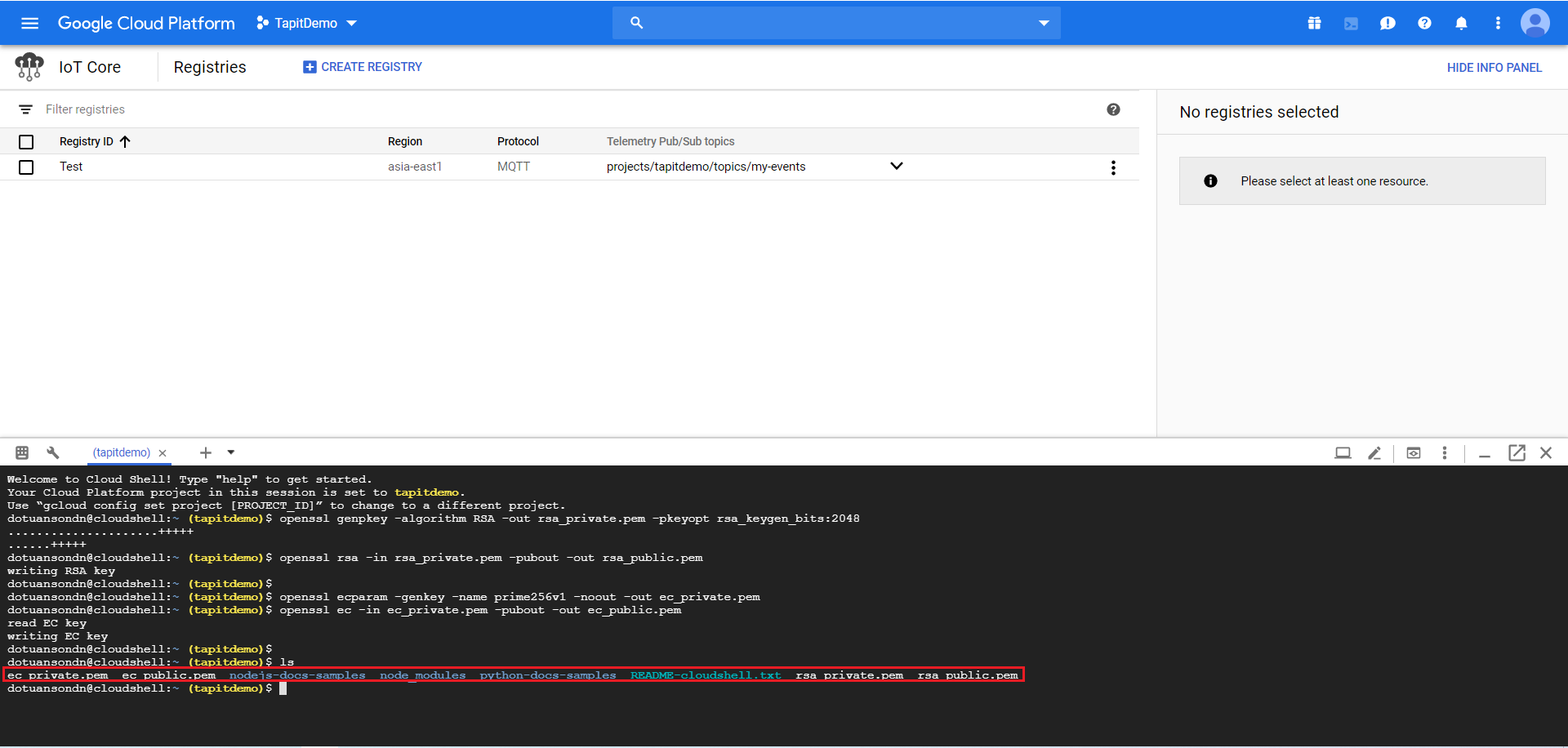
openssl ecparam -genkey -name prime256v1 -noout -out ec\_private.pem

openssl ec -in ec\_private.pem -pubout -out ec\_public.pem

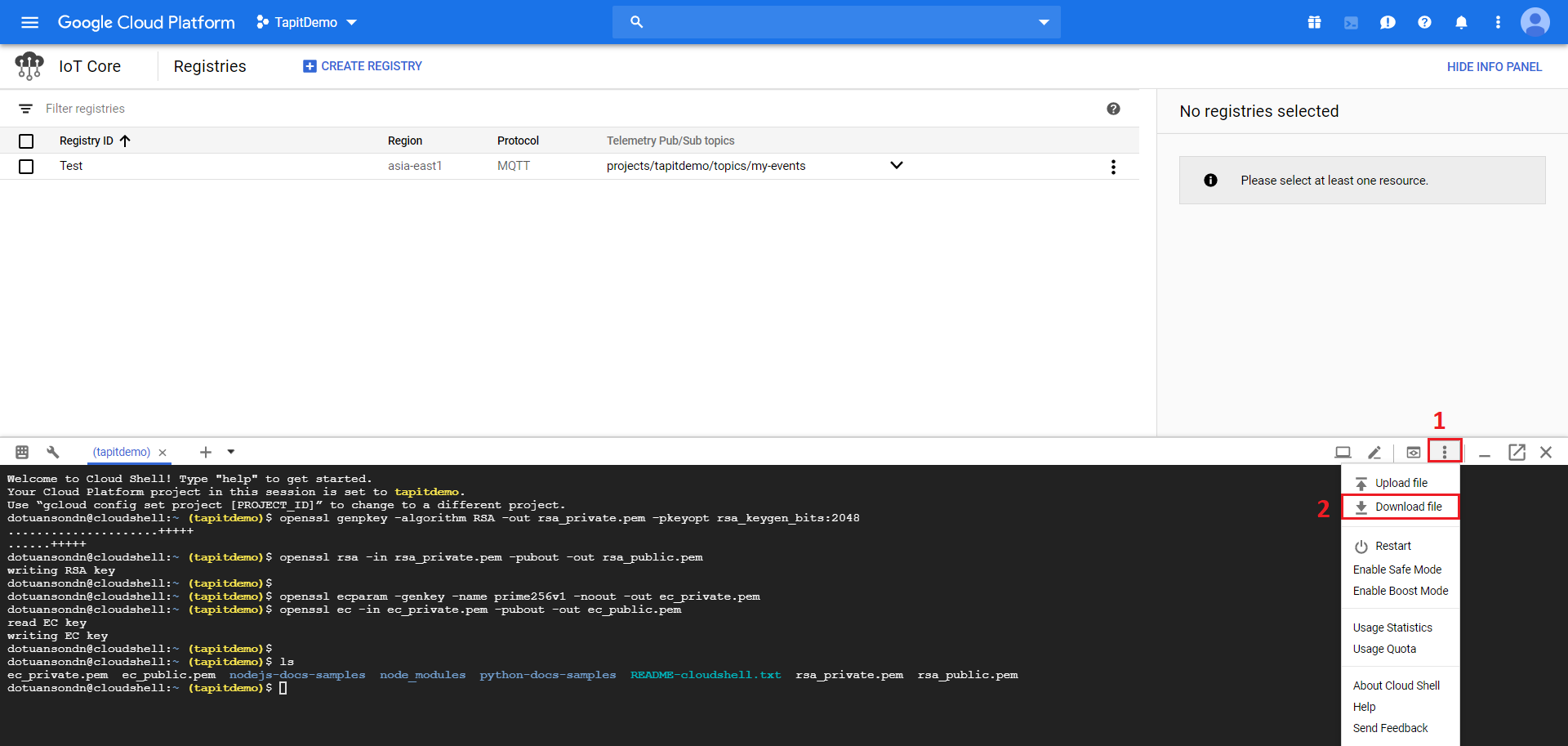
* Các dòng lệnh này sẽ tạo cho chúng ta 2 file:
* *ec\_private.pem*: Chứa **private key**, được lưu trữ tại thiết bị IoT và được mã hóa bởi JWT.
* *ec\_public.pem*: Chứa **public key,** được lưu trữ tại dịch vụ Google IoT Core để xác nhận mã JWT từ thiết bị IoT gửi lên.

*\* Ngoài ra Google IoT Core còn hỗ trợ cho chúng ta thêm cách tạo các cặp key sử dụng thuật toán RSA và EC sở hữu chứng chỉ X.509.*

Để xem các file chúng ta vừa khởi tạo, tại cửa sổ dòng lệnh gcloud, chúng ta điền lệnh *ls.*



Để tải các file này về máy, tại cửa sổ dòng lệnh gcloud, chúng ta thao tác như hình bên dưới, sau đó điền tên file (bao gồm cả đuôi file) mà chúng muốn tải về máy. VD: rsa\_public.pem, ec\_private.pem,...



Ở đây mình tải 2 file *ec\_private.pem* và *ec\_public.pem* về máy để tiện cho việc đọc file và tạo chuỗi JWT ở phần sau. Ngoài ra, các bạn còn có thể đọc trực tiếp nội dung file trên cửa sổ gcloud bằng lệnh **cat***.* VD: cat ec\_public.pem,...

**II. Hướng dẫn tạo Json Web Token (JWT)**

Để tạo JWT thì chúng ta có thể sử dụng các thư viện có sẵn do Google cung cấp với những ngôn ngữ phổ biến như C++, Java, Python,... Ở đây thì mình sẽ không sử dụng các thư viện có sẵn này mà thay vào đó, mình sẽ tự tạo một chuỗi JWT thông qua tool online trên trang jwt.io, việc này sẽ giúp chúng ta nắm rõ hơn để tạo ra một chuỗi JWT thì cần những thông số gì.

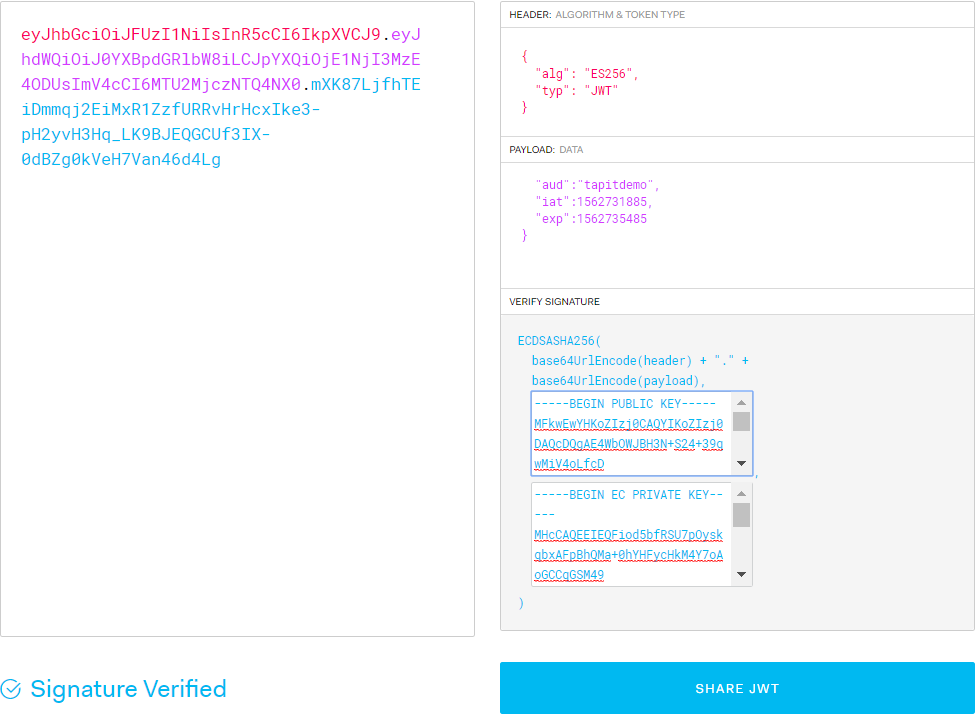
Để bắt đầu với JWT, chúng ta sẽ đi vào định nghĩa của nó:

JWT là là một đối tượng JSON được định nghĩa như một đoạn mã an toàn để trình bày một tập hợp thông tin giữa hai bên. Đoạn mã token bao gồm header, payload và signature.

Nói cách khác, JWT là một chuỗi có định dạng:

*Header.Payload.Signature*

Để có thể tạo ra một chuỗi JWT hợp lệ do Google IoT Core quy định, trong phần **Decoded** chúng ta khai báo các thông số tương ứng với 3 phần Header, Payload, Signature như sau.



*Khởi tạo chuỗi JWT sử dụng thuật toán ES256 thông qua jwt.io*

* ***Header***: Đây là thành phần tiêu đề với trường “alg” là thuật toán mã hóa và trường “typ” xác định đối tượng là JWT.
* ***Payload***: Đây là thành phần nội dung chứa thông tin mà chúng ta muốn thiết bị IoT trao đổi với Cloud. Google IoT Core yêu cầu chúng ta phải đặt vào phần này những trường sau:
* “*aud*”: Project-ID của project mà chúng ta đang làm việc.
* “*iat*” : Thời điểm chuỗi Json được tạo ra (Sử dụng định dạng timestamp).
* “*exp*” : Thời điểm chuỗi Json hết hạn (Sử dụng định dạng timestamp).
* ***Signature***: Đây là thành phần quan trọng được tạo ra bằng cách kết hợp 2 phần (Header + payload), rồi mã hóa bằng một giải thuật encode nào đó, ở đây mình dùng ES256 nên sẽ sử dụng cặp public/ private key để tạo ra signature. Các bạn lưu ý điền vào public/private key tương ứng đã tạo ở phần I.

Sau khi điền hết các thông số trên vào phần **Decoded**. Ở phần **Encoded**, chúng ta đã có một chuỗi JWT hoàn chỉnh. Chúng ta sẽ lưu chuỗi token này vào trong máy để sau này dễ dàng lấy ra sử dụng.

Như vậy là chúng ta đã có thể tạo được cặp Public/ Private key và một chuỗi JWT hoàn chỉnh cho một thiết bị muốn kết nối tới dịch vụ Google IoT Core. Ở bài viết tiếp theo nhóm TAPIT IoTs sẽ hướng dẫn các bạn cách thiết lập cặp Public/ Private key và chuỗi JWT đồng thời demo tổng quan một client muốn kết nối tới Google IoT Core sẽ trải qua những bước như thế nào. Chúc các bạn thành công!