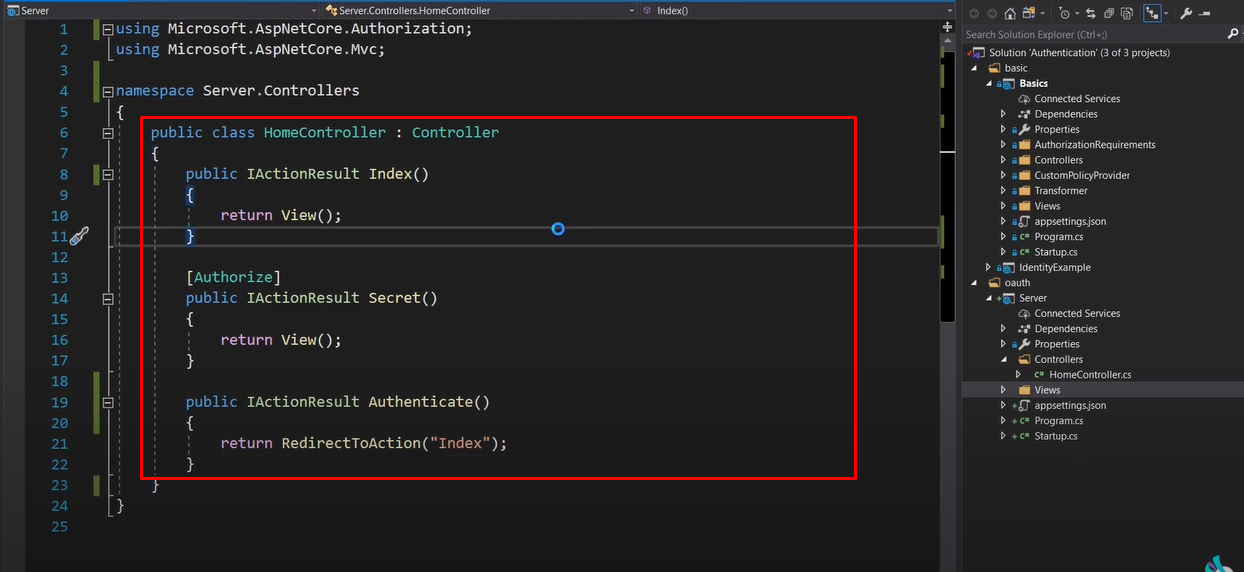
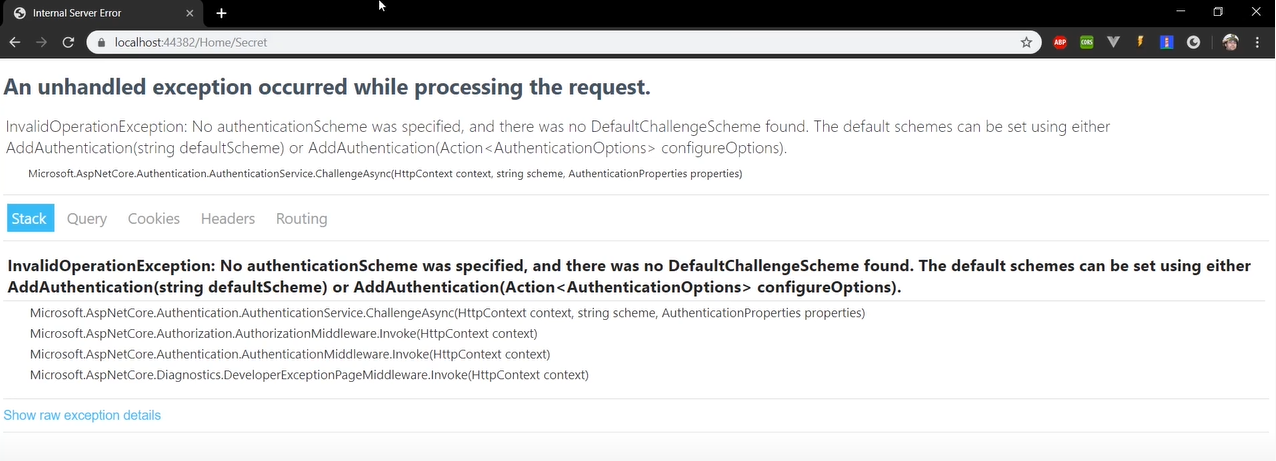


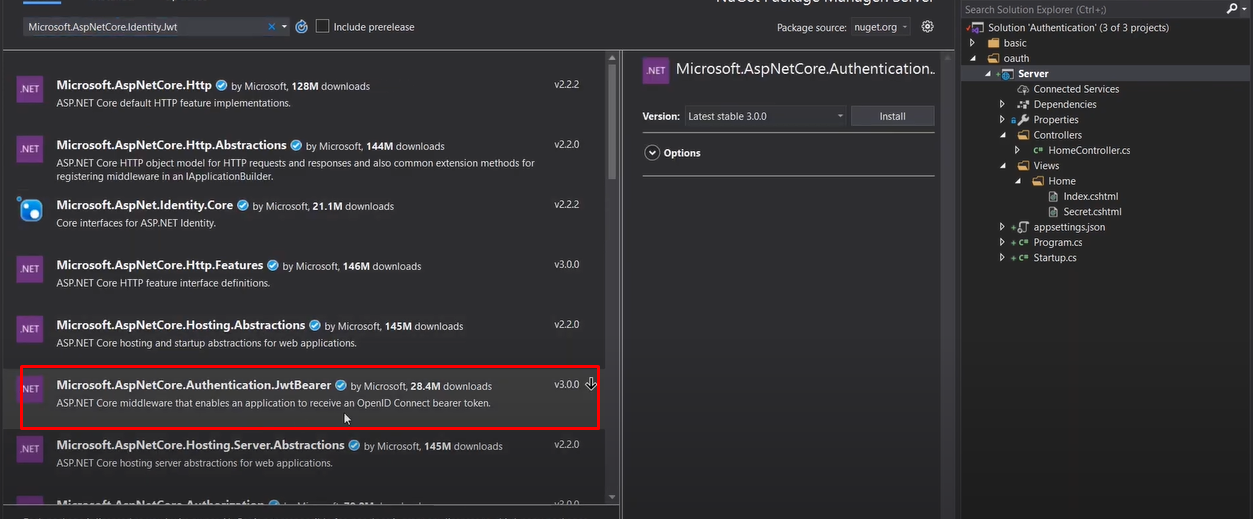
middleware thì vẫn tương tự



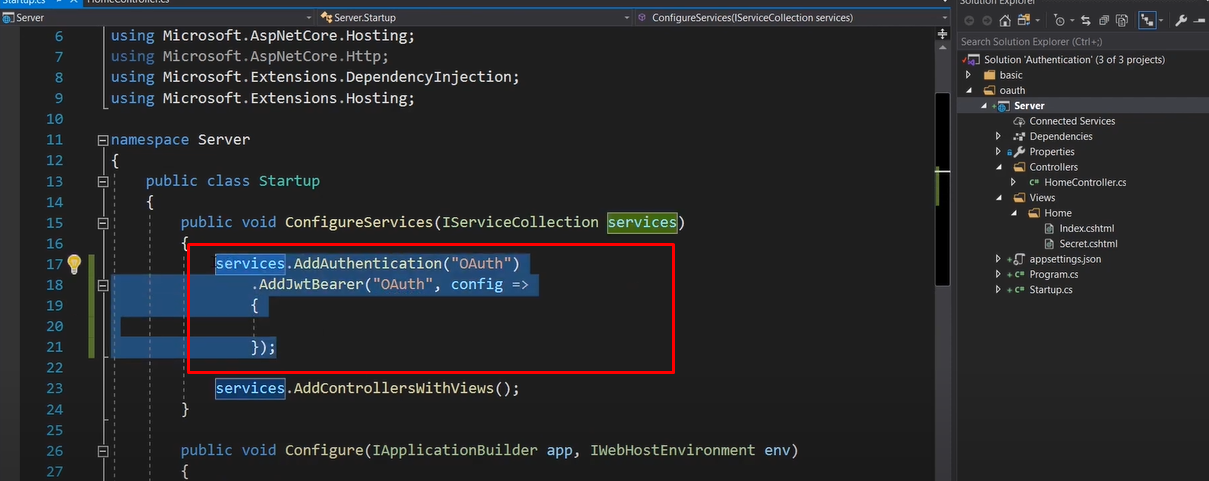
Controller giữ lại 3 cái này

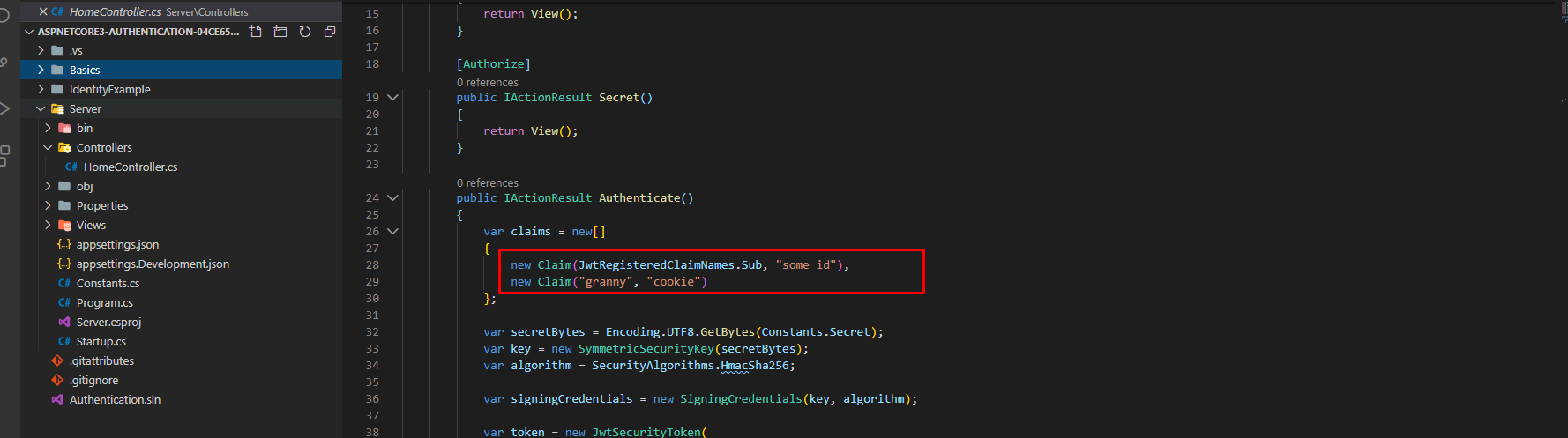


Hiện tại sẽ lỗi vì chưa được authen



Cài nuget

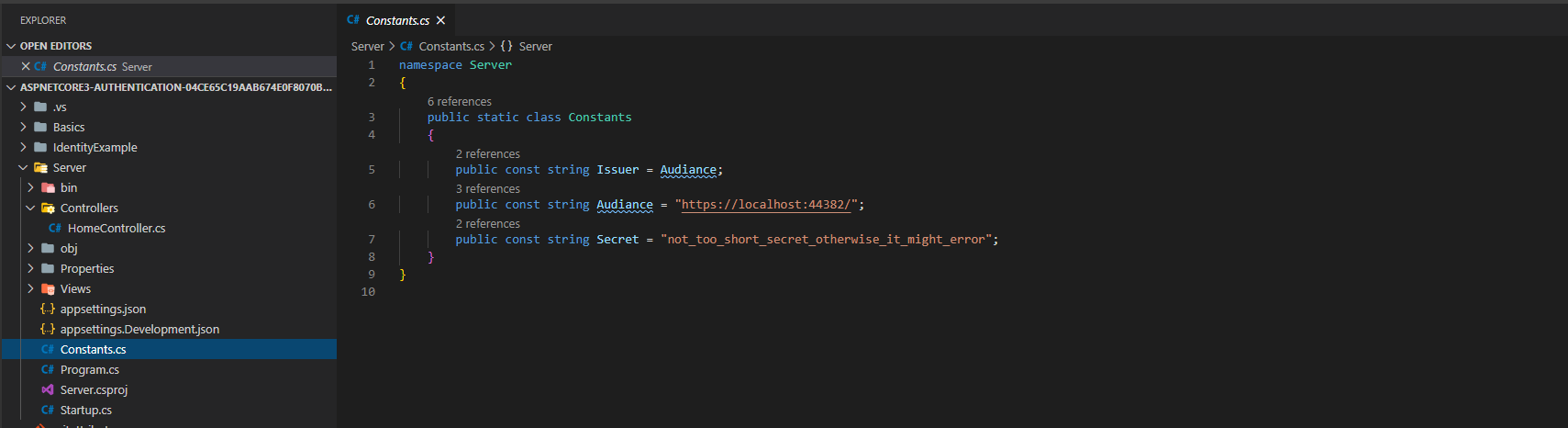
hiện tại nếu request vào secret sẽ nhận lỗi 401 nghĩa là chưa authen



Tạo claim array

Sub ở đây chứa id của user, sub là 1 claim được implement sẵn trong doc rồi, nên dùng những claim được implement sẵn này, vì nó quy chuẩn chung và được sử dụng chung

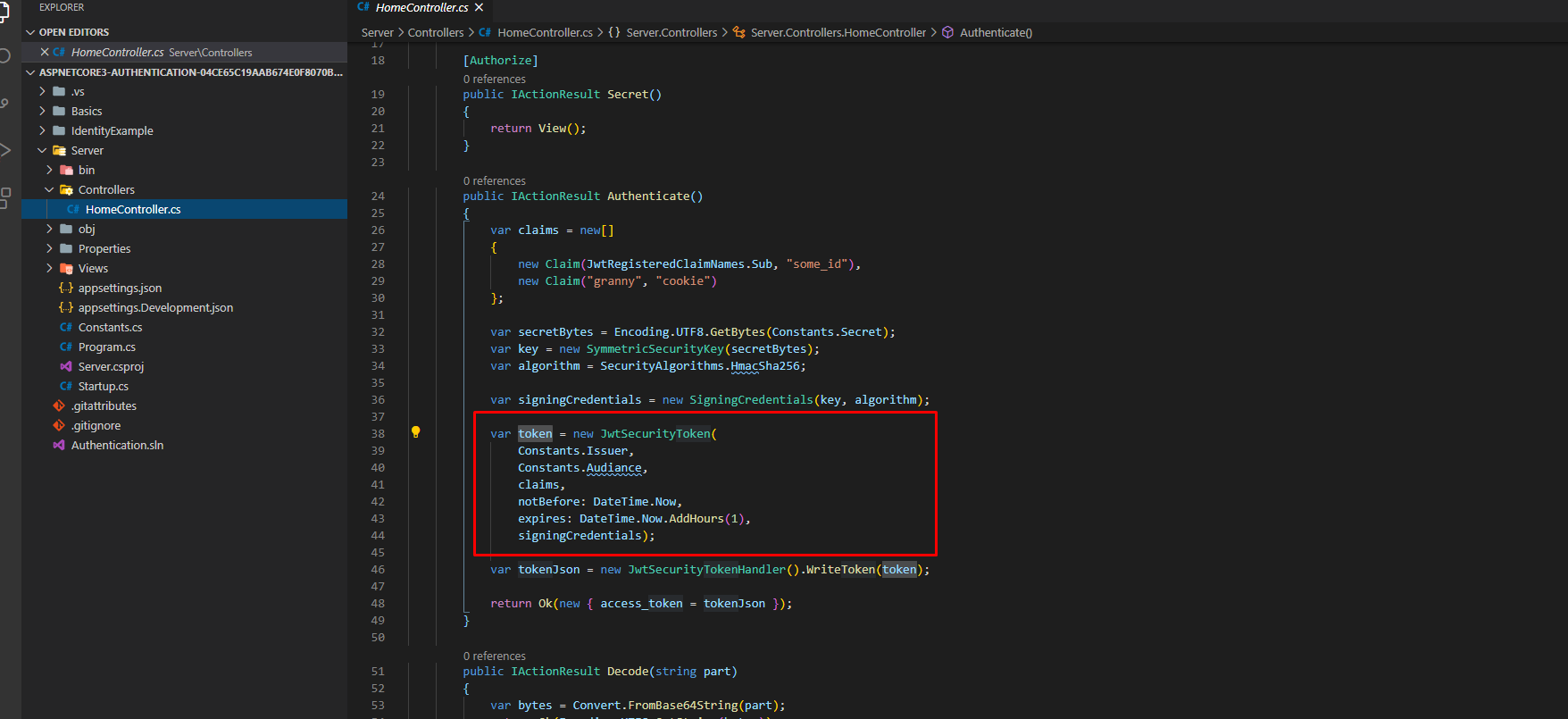
Ganny ở đây là 1 custom claim, và chứa giá trị do dev quyết định



Tạo class constant chứa contant

Issuer là nơi phát hành tocken

Audience là nơi mà token sẽ được sử dụng

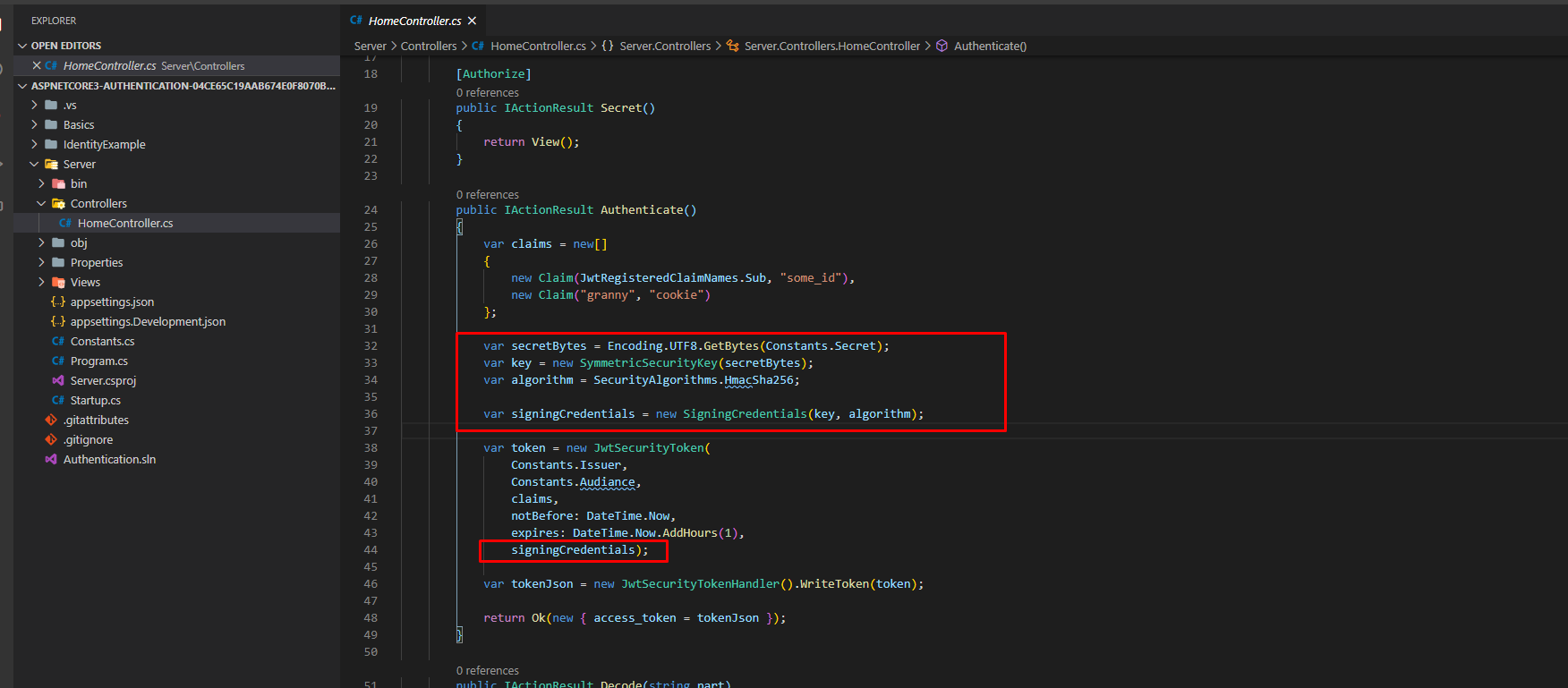


Bắt đầu bằng issuer, audience

Đến claim

notBefore: thời điểm token bắt đầu có tác dụng

expires: thời điểm token hết hạn



Conver secret key to binary lưu vào secretBytes

Tạo ra key từ secretkey đã convert

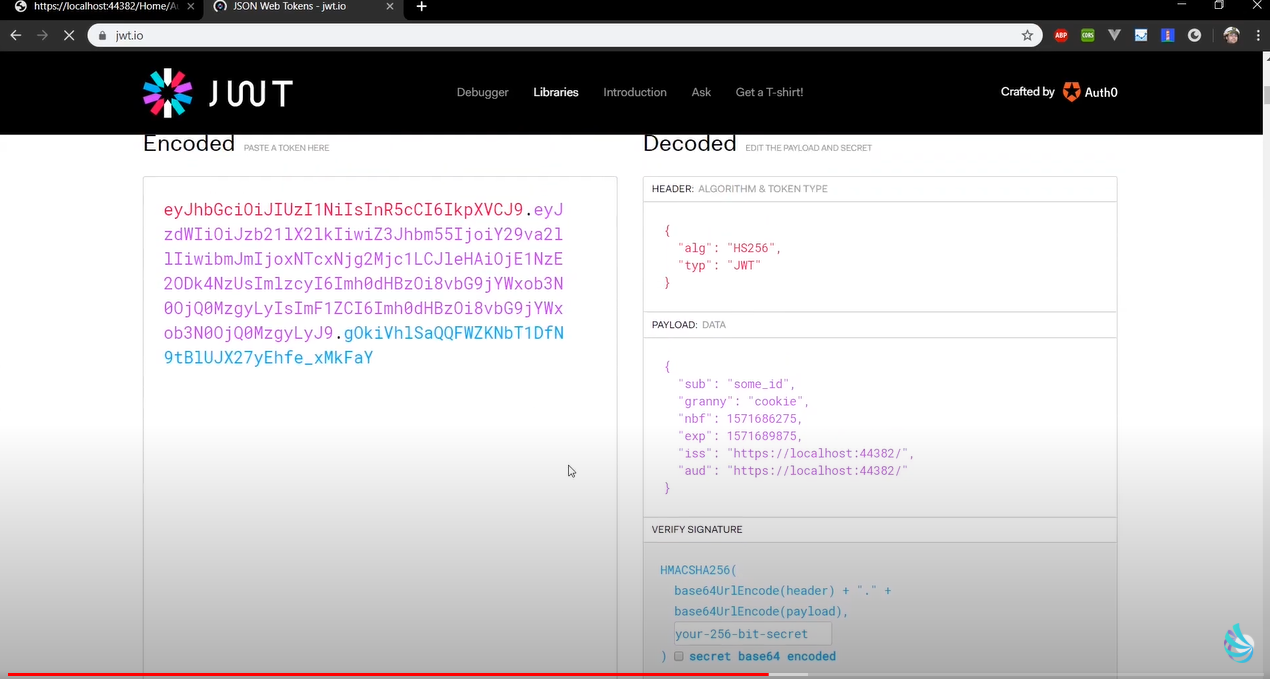
Algorithm chọn thuật toán HmacSha256

Dùng cái signingCredentials vừa tạo đưa vào token

Sau đó write token và trả về kết quả là token



Khi vào link sẽ nhận được token như sau



Biên dịch ra để xem thông tin của token

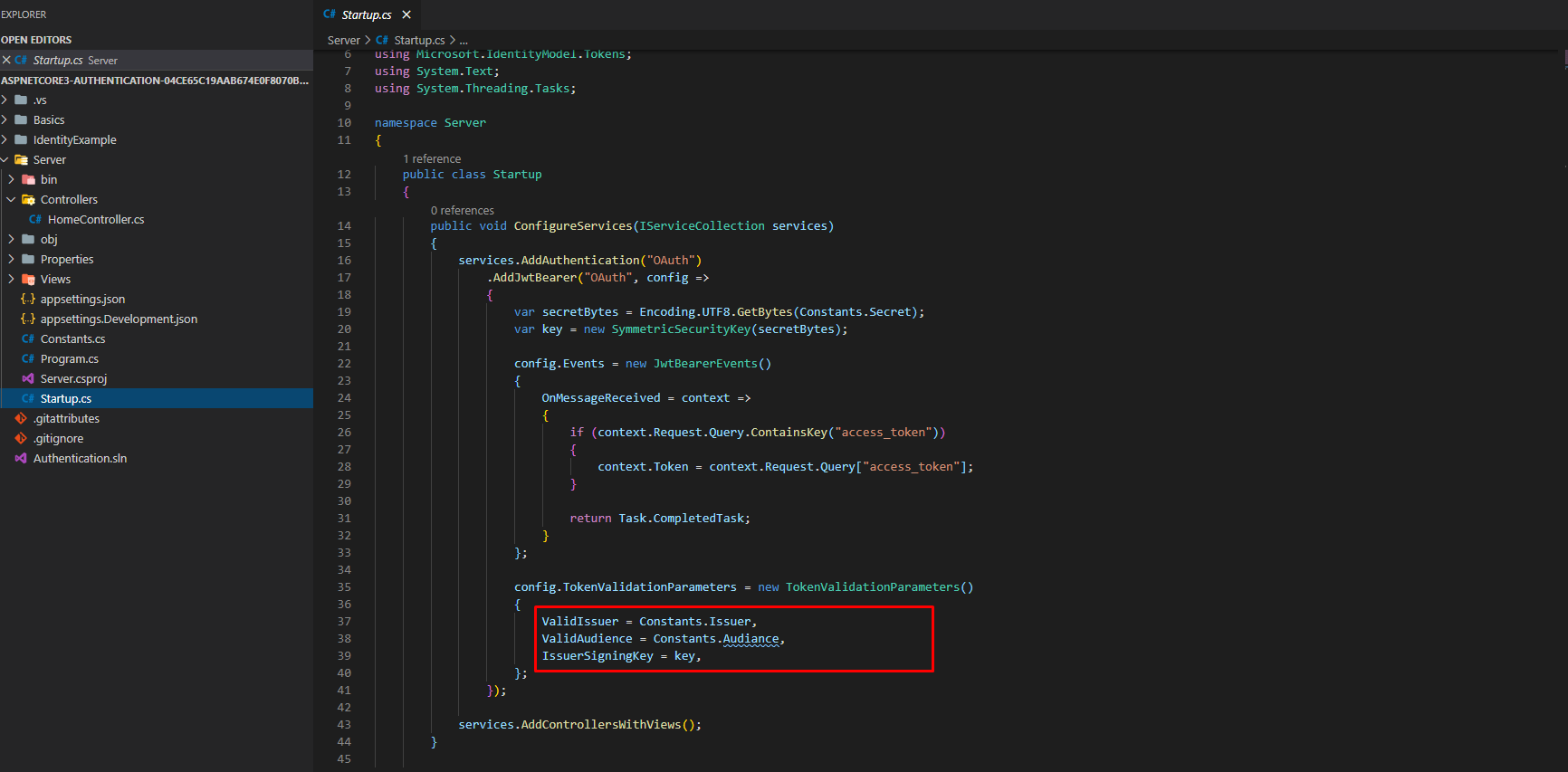
Token gồm 3 phân

Header

Payload

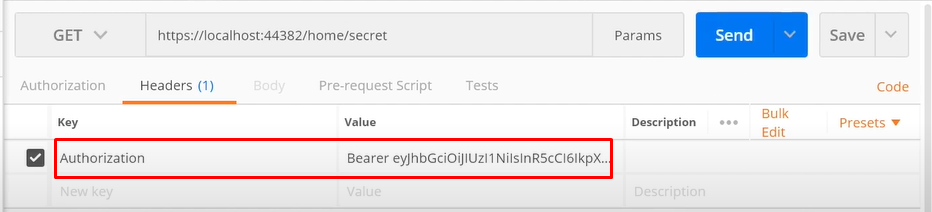
Verify signature: cái này là secret

User request tới dùng token thì token đó phải có Verify signature đúng thì thông tin mới đúng, còn 2 phần còn lại public ko vấn đề

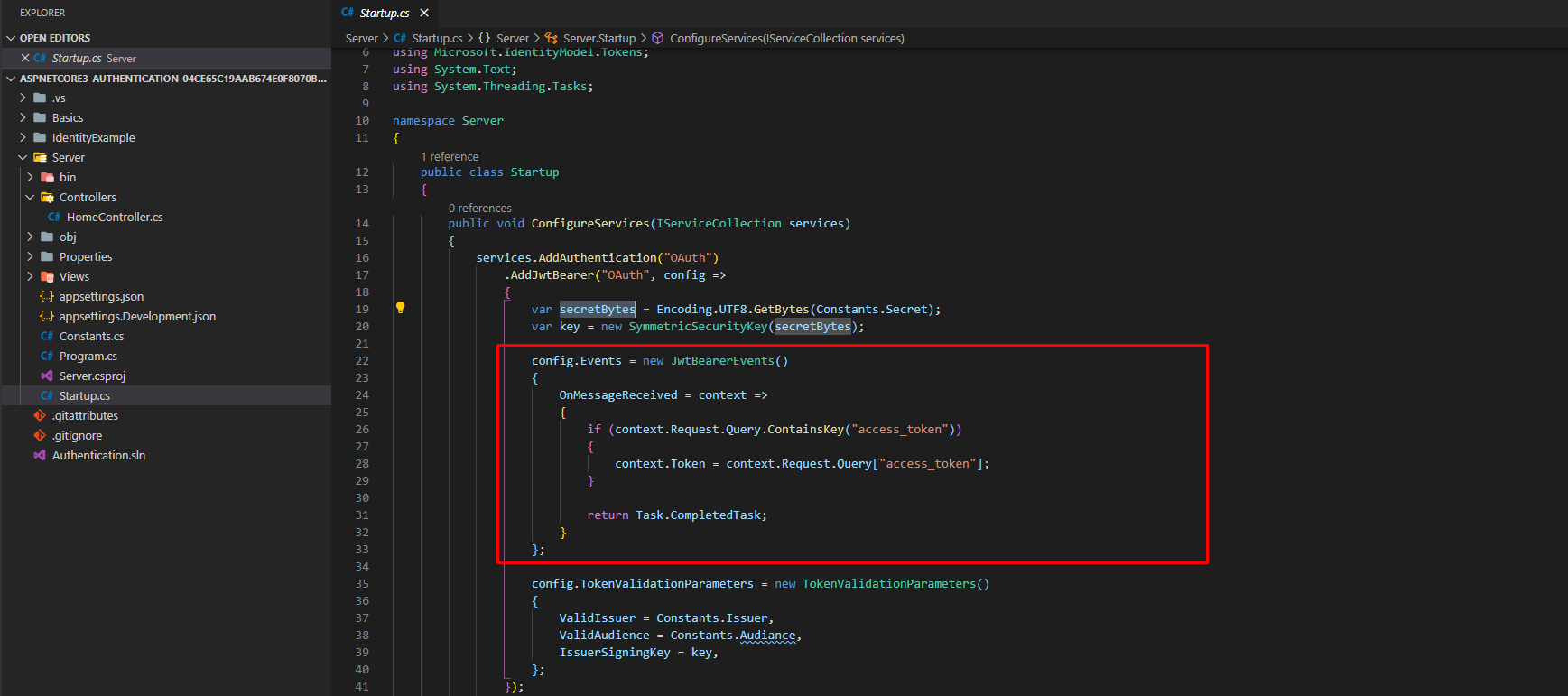


Token request vào phải có 3 trường này giống với giá trị đưa vào ở đây thì token đó mới được validate

Key phải dùng method sinh ra lại key như trên, vì key trong secret là key string, nên phải đưa về binary rồi dùng hàm sinh ra key

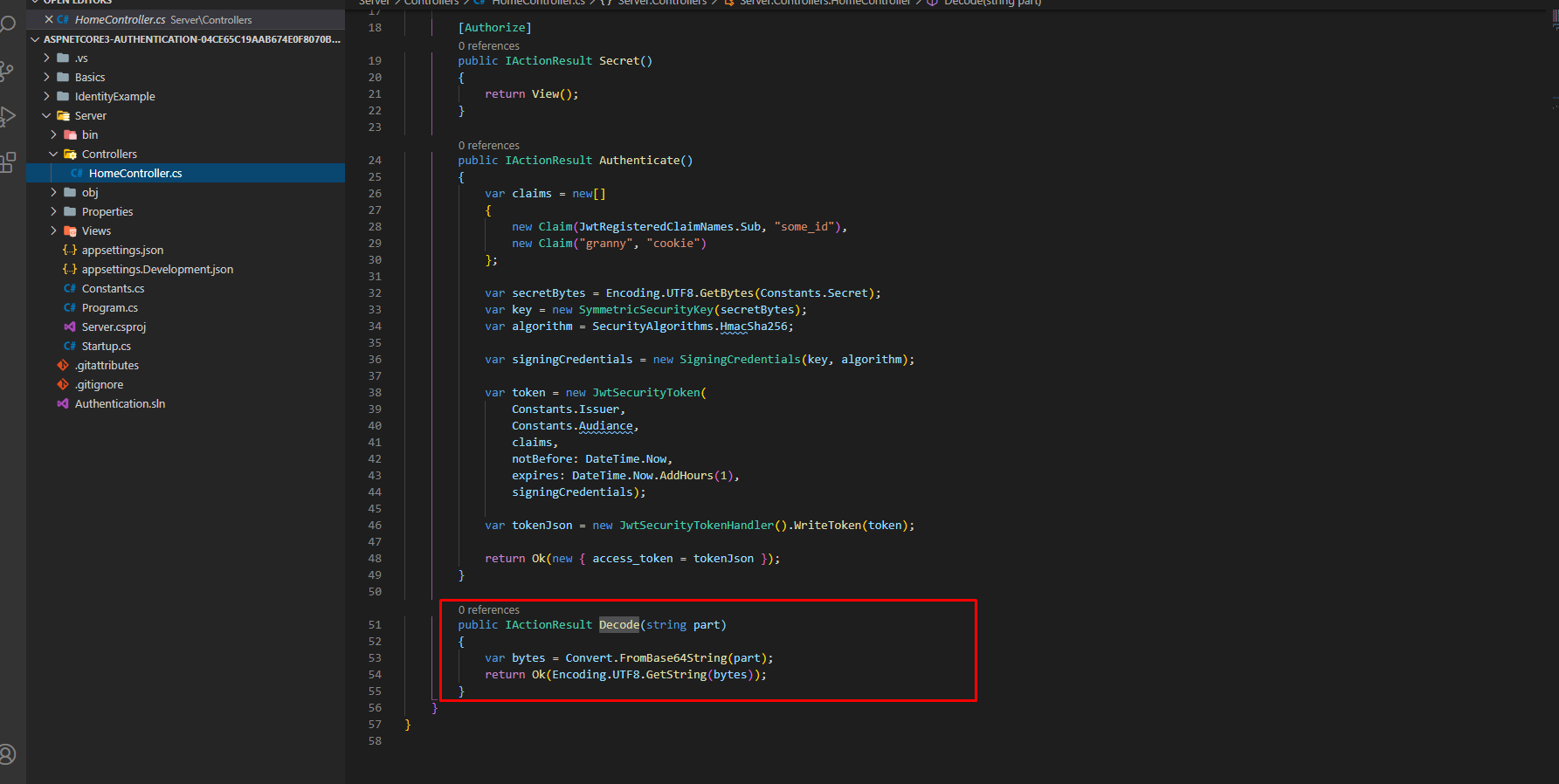


Reques đưa token vào header là request được



Cấu hình để đưa jwt vào bằng url query





Decode



Như vậy là decode từng part lấy được từng giá trị, decode dùng cho từng path

Split string ra bằng dấy chấm, dùng mãng đó decode từng path

Hoặc là dùng identiy để truy cập tới các claim cũng được ( cách này xem trong khóa microservice)