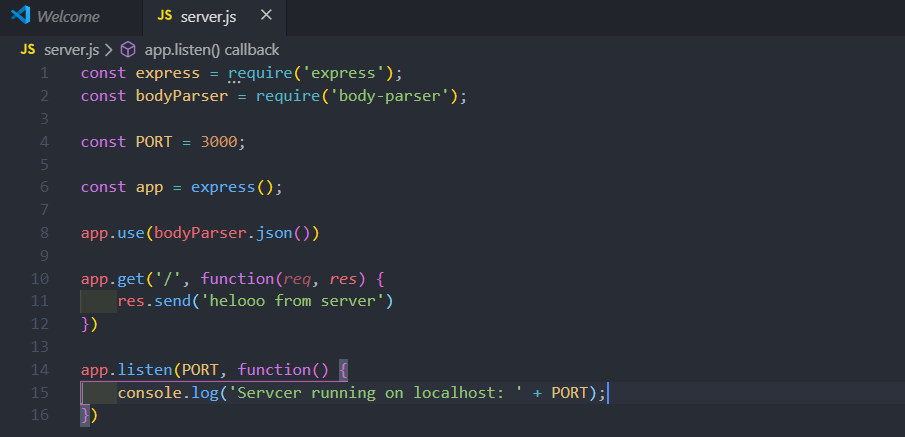
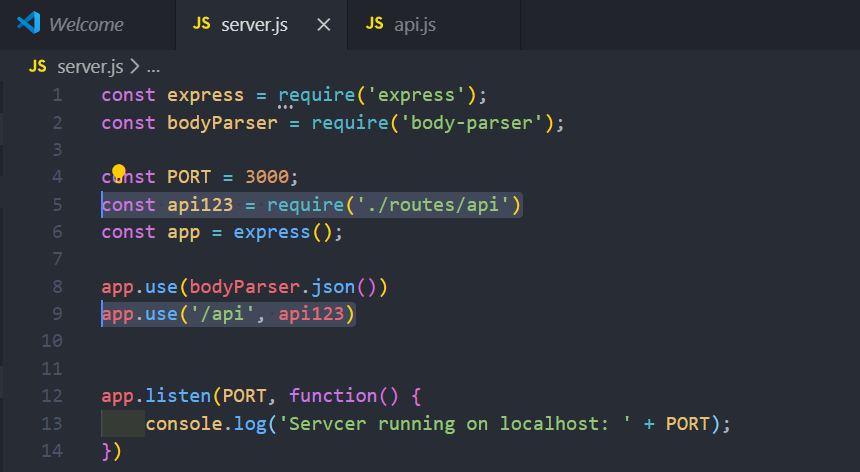
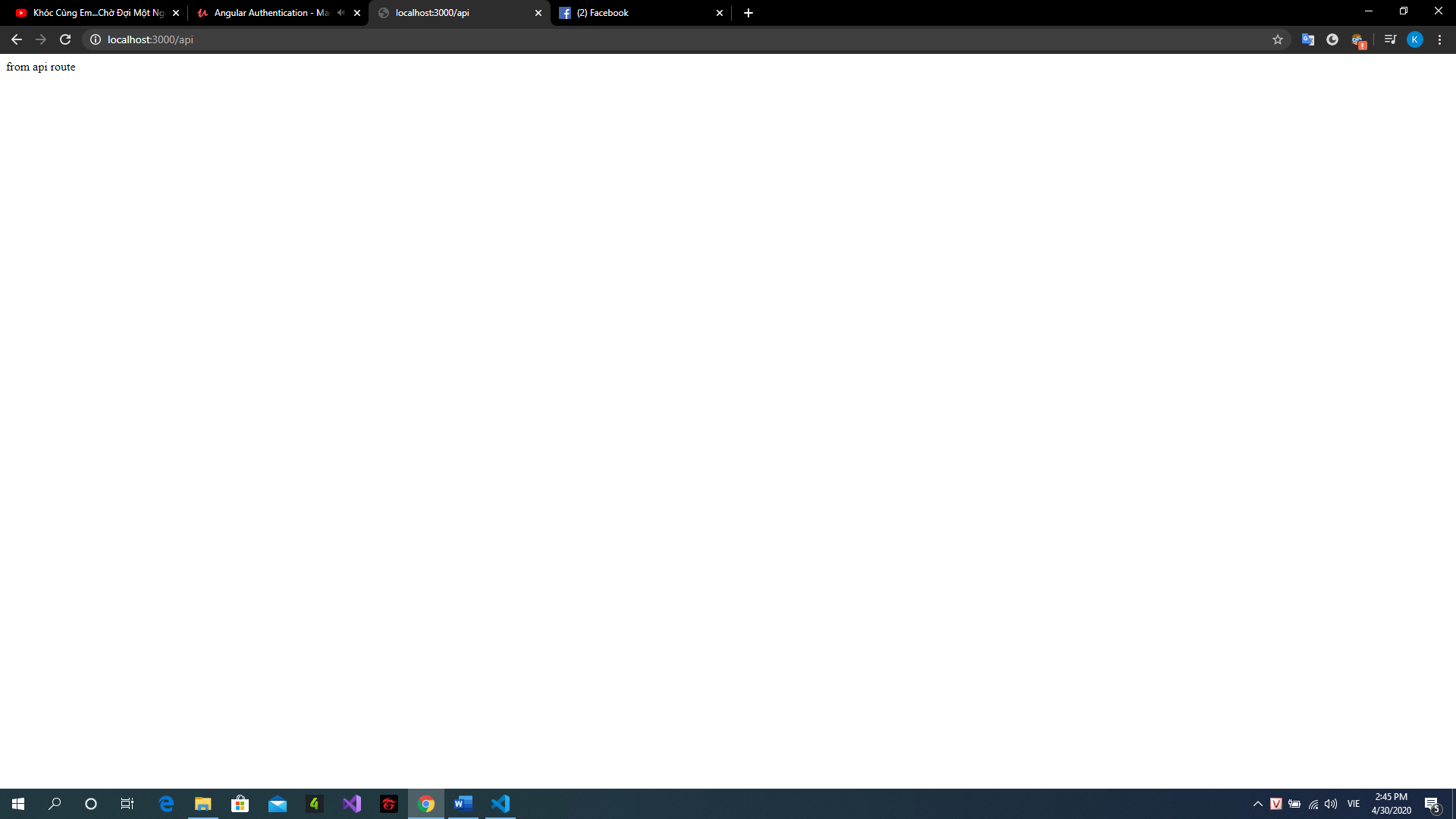
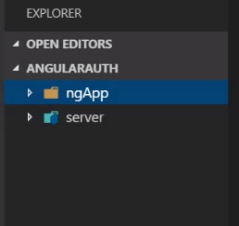
MEAN: MongoDb, Express, Angular, Nodejs

-Express Server:

1. Khởi tạo project “npm init” => “npm install express body-parser –save”.
2. Tạo file Server.js:
   1. 
3. Tạo folder Routes, tạo file “api.js”:
   1. 
4. Tại Server.js, import api vào và sử dụng use để thiết lập API :
   1. 
5. Chạy node server.js:
   1. 

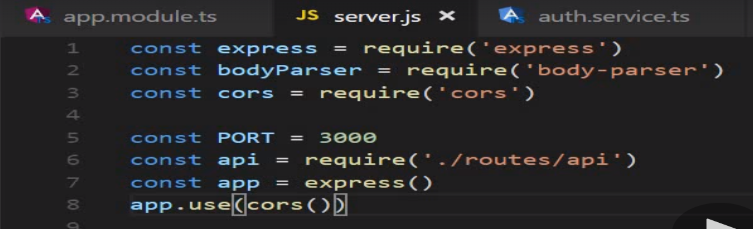
-Setup MongoDb database: TỰ KẾT NỐI VỚI DATABASE THÍCH HỢP CỦA MÌNH.

-Setup Front-end project: Tạo mới thư mục Angular như bình thường, làm như bình thường không liên quan gì với server project.

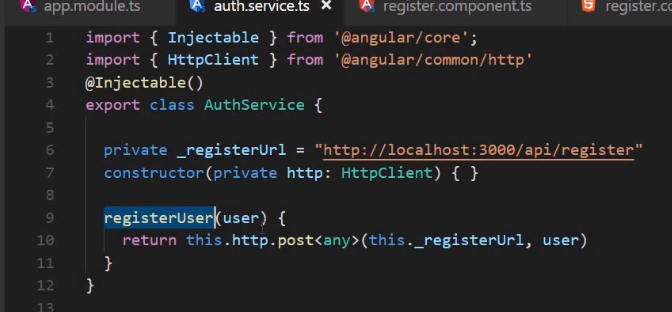
1.  => 2 dự án server và angular front-end nằm chung trong 1 folder.

-Nhưng khi gọi API thì thất bại vì chưa có sự liên kết giữa 2 project, cần sử dụng middleware npm “cors” bên server project để kết nối 2 project:

 => cài đặt CORS

 => Sử dụng “cors” vào server.js (index.js).

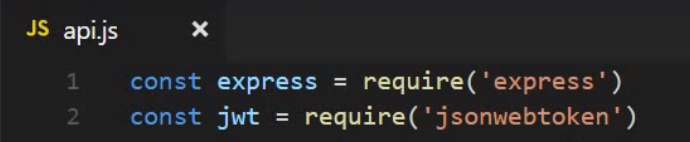
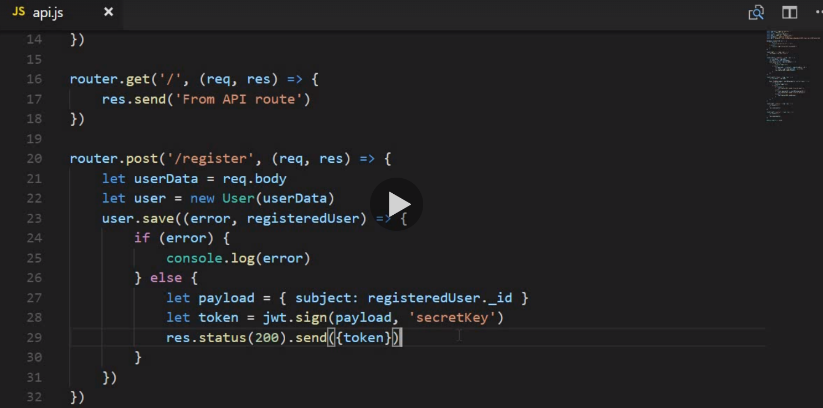
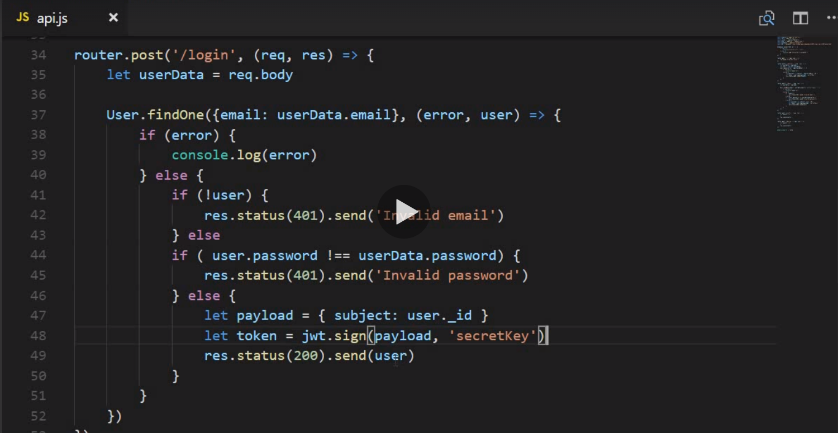
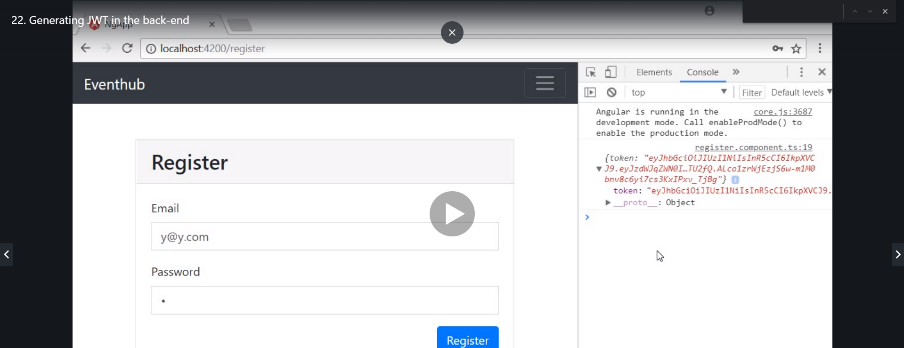
- Tạo RegisterUser API bên phía client, auth.service.ts:

 => trả về 1 observable

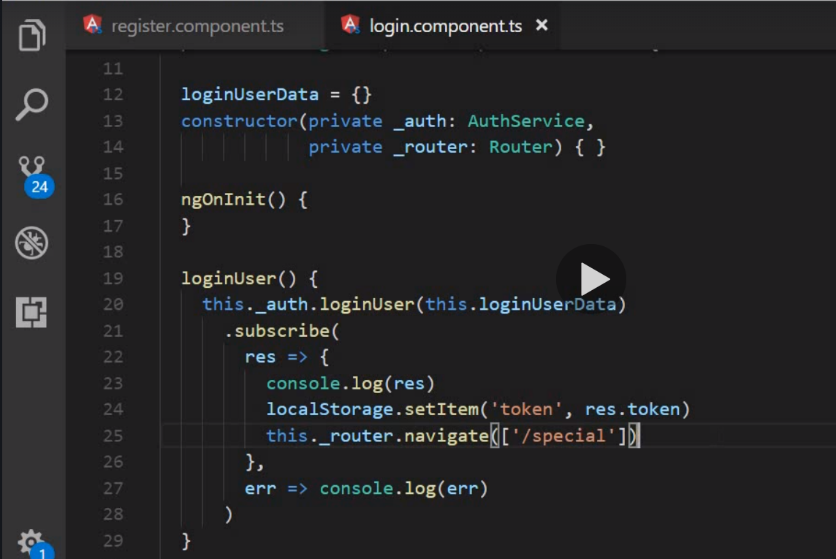
-Làm tương tự với Login.

-AUTHENTICATION OVERVIEWS

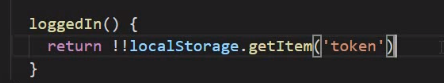
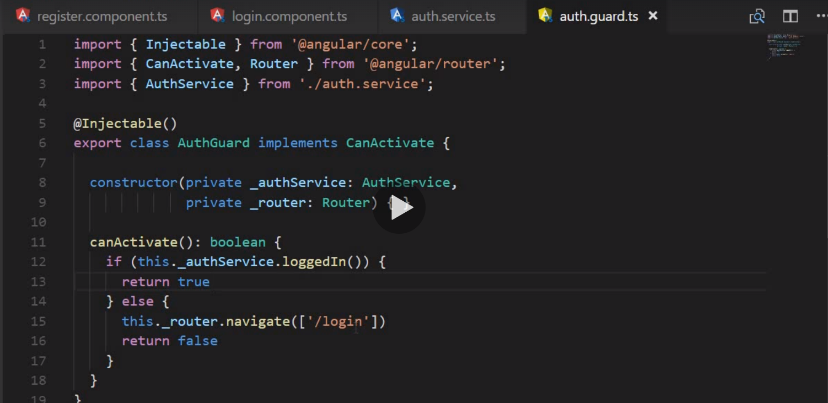
-Generating JWT in the Back-end với chức năng “Login” và “Register”:

1. Cài đặt gói npm jsonwebtoken: 
2. Khai báo JWT trong api.js: 
3.  => Tạo token với phương thức .sign() sau đó gửi về theo res khi đăng ký user.
4.  => Tương tự với Login.
5.  => Kết quả: nhận được token khi login hoặc register thành công.

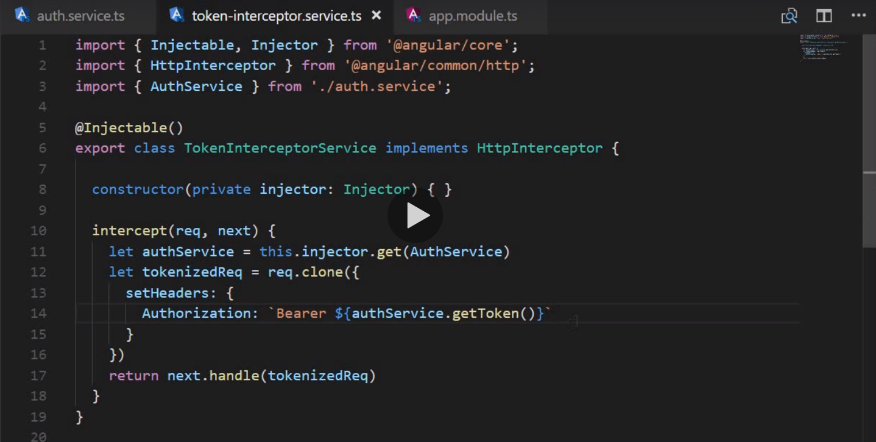
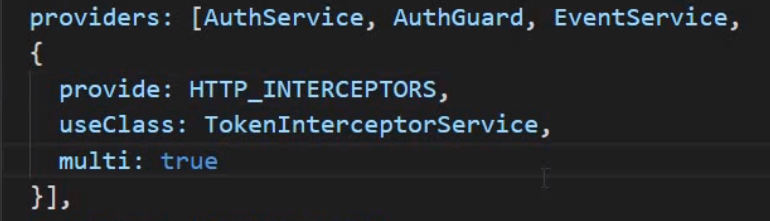
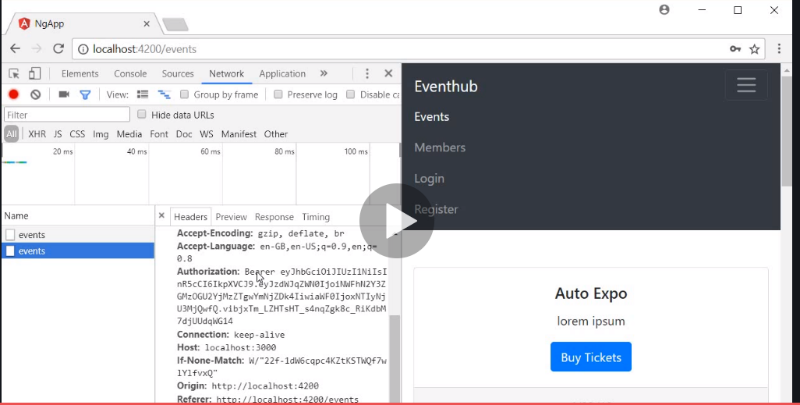
-Stroring JWT in the Local Storage

 => Login thành công => Lưu token vào localStorage => navigate vào route ‘/special’.

-Special Events Route Guard: Dùng guard để hạn chế việc để các user chưa được xác thực vào cái trang cần phải được xacs thực như admin, ....:

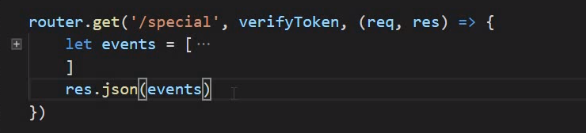
1. Trong auth.service.ts:  => Hàm này trả về true hoặc false, localStorage có item Token thì true, không có thì False.
2. Tạo file Auth.guard.ts:  => localStorage có item Token thì true (được truy cập vào component ấy), không có thì False(không được truy cập vào component và navigate router ‘login’).
3. => **VẤN ĐỀ:** Chưa giải quyết thỏa đáng được vấn đề nếu chỉ dùng mỗi auth guard bởi user có thể **tự động** setItem(‘token’) trên màn hình console, vậy là họ sẽ **có thể truy cập** vào component ấy.Do đó ta cần phải tạo **HTTP-INTERCEPTOR.**

-Tạo file ‘*token-interceptor.service.ts’*:

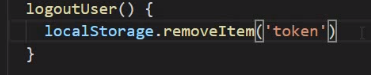
1. **INTERCEPTOR**: Mục đích: dùng để chạy 1 số code ngay trước khi gửi 1 Request nào đó về server. (giả sử lấy token để xác thực user trước khi gửi request về server để thực hiện 1 event nào đó).
2. Trước khi mỗi HTTP Request được gửi lên server, ta gán token vào header để server xác thực token có hợp lệ không rồi mới thực hiện theo API. Ta dùng injector để import AuthService vào đây(Lý do chưa hiểu):
   1. 
3. Các bước import cần thiết ở App.module.ts:
   1. 
   2. 
   3. 
   4.  => Khi gửi API lên server thì Token sẽ được gán vào đây.

* OKEY thiết lập thành công ở Client.

-Middleware to Verify Token: Sau khi Client gửi Token lên thành công, ta cần phải xác thực Token ấy hợp lệ, chính xác không thì mới tiếp tục:

1. Tạo middleware function verifyToken() tại api.ts server: 
2.  => inject nó vào những API cần tính xác thực bằng token. Như vậy mỗi khi client gửi API “/special” thì nó sẽ chạy hàm verifyToken() trước, nếu thành công thì mới chạy tiếp.

-Logout:

1. Remove Token tại localStorage khi click Logout Button: 

KẾT THÚC.

Tham Khảo thêm :

<https://www.youtube.com/watch?v=2jqok-WgelI>

<https://www.youtube.com/watch?v=0g0Of8jlhN8>