Angular模块引用

项目开发

1. 视频管理
2. 学习路径详情
3. 学习计划
4. 本地化：
    - app.module注册本地化文件

安全

1. 前端：路由守卫
    - 创建守卫服务auth.guard.ts，他要实现相应的守卫方法并返回布尔值或异步布尔值结果

```
export class AuthGuard implements CanActivate {

  constructor(private us: UserService,
              private router: Router) {
  }
  canActivate(
    next: ActivatedRouteSnapshot,
    state: RouterStateSnapshot) {
    console.log(next, state);

    return this.us.isLogin().pipe(
      tap(// 监听返回结果，如果没有登录则跳转至登录页
        isLogin => {
          if (!isLogin) {
            this.router.navigate(['/user/login']);
          }
        }
      )
    );
  }
}
```

    - 添加路由守卫

```
{
  path: 'main', component: MainComponent,
  canActivate: [AuthGuard]
}
```

2. 后端：路由保护：主要通过中间件

- 编写中间件

```
module.exports.requireUser = function (req, res, next) {
    if (!req.session.user) {
        // 用户未登录，返回401状态码
        res.sendStatus(401);
    } else {
        next()
    }
}
```

- 路由级别保护

```
router.get('/my-course/:id', requireUser,(req,res)=>{})
```

- 文件级别保护

```
app.use('/admin', requireAdmin, adminRouter);
```

3. 跨站脚本攻击XSS：攻击者尝试将有害代码插入到dom中，解决方案是组织有害代码进入dom
    - angular有一套完整的去伤害化方案，它不信任任何输入值，将任何输入值提前进行无害化处理从而避免XSS。
    - 无害化测试

```
~ ts
htmlSnippet = 'aaaa<script>alert("lala")</script><b>ffff</b>'

~ html
<h3>绑定innerHTML</h3>
<p [innerHTML]="htmlSnippet"></p>

<h3>插值绑定</h3>
<p>{{htmlSnippet}}</p>
```

    - 信任安全值：如果确信值无害，可通过api告知ng不要进行无害化处理

```
~ ts
this.trustedUrl = sanitizer.bypassSecurityTrustUrl(this.dangerousUrl);

~ html
<p><a [href]="trustedUrl">trustedUrl</a></p>
```

> 调用以下方法可将一个值标记为可信

```
bypassSecurityTrustUrl 信任url,比如a[href]
bypassSecurityTrustStyle 信任样式值，用于div[style]
bypassSecurityTrustHtml 信任html，用于p[innerHTML]
bypassSecurityTrustResourceUrl  比如iframe[src] script[src]
bypassSecurityTrustScript
```

4. 服务器端安全点：参考

注意事项：

提前熟悉ng模块懒加载和异步加载机制

提前熟悉发布、部署一些基本操作，如nginx，pm2，ftp，阿里云，反向代理等等工具概念使用