

Angular 参考 Demo 项目启动步骤

前期准备

首先保证你的 Git, Node.js 和 AngularCLI 已经安装完成, 参考文档【01.Git 安装及配置.pdf】,【02.Node.js 安装配置及 Angular CLI 的安装.pdf】。

我们的 Angular demo 项目名称是: e-bidding-system

下载运行参考 demo 项目:

1. 从 Gitee 上面 clone 项目

创建一个项目路径, 例如: D:\01.Programming\01.Angular, 然后打开命令行工具, 并切换到这个路径中, 然后执行:

```
>git clone https://gitee.com/dlut2025/e-bidding-system.git
```

或者如果你在本地配置了 SSH, 则运行下面的命令, 这两个命令都是用来将代码 clone 到本地

```
>git clone git@gitee.com:dlut2025/e-bidding-system.git
```

```
D:\01.Programming\01.Angular>git clone git@gitee.com:luanxiyuan/e-bidding-system.git
Cloning into 'e-bidding-system'...
remote: Enumerating objects: 432, done.
remote: Counting objects: 100% (432/432), done.
remote: Compressing objects: 100% (376/376), done.
Receiving objects: 100% (432/432), 2.32 MiB | 7.23 MiB/s, done.
Resolving deltas: 100% (155/155), done.
```

2. 切换 demo 项目分支到 staging

参考 demo 在 git 的另一个分支 staging 上, 因此我们需要切换到 staging 分支, 在项目根路径中执行

```
>git checkout staging
```

```
PS D:\01.Programming\01.Angular\test\e-bidding-system> git checkout staging
Switched to a new branch 'staging'
branch 'staging' set up to track 'origin/staging'.
```

再执行一次>npm install

```
PS D:\01.Programming\01.Angular\test\e-bidding-system> npm install

removed 1 package, and audited 902 packages in 2s

92 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

3. 安装依赖包

项目想要跑起来的前提是所有的依赖包都安装在本地，在命令行工具中将路径切换到 clone 下来的这个项目的根路径中：

```
D:\01.Programming\01.Angular>cd e-bidding-system
D:\01.Programming\01.Angular\e-bidding-system>
```

然后在项目根路径中执行 `> npm install` 安装依赖包，这里需要等待 npm 下载对应的依赖包：

```
D:\01.Programming\01.Angular\e-bidding-system>npm install
npm WARN deprecated @npmcli/move-file@2.0.1: This functionality has been moved to @npmcli/fs
added 901 packages, and audited 902 packages in 12s

92 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

【说明】：参考 demo 需要基于 API response，将拿到的 API response 数据显示在页面上，因此我们需要先启动后台的服务。如果你的后台 API 接口已经开发完成，直接跳到第 5 步；如果 API 接口没完成，也可以通过启动 mock server 模拟 API 接口返回数据，继续第 4 步。

4. 启动 mock server

参考文档【05.启动 mock server.docx】

5. 启动后台项目 API 接口服务

参考第一部分后台开发所学到的步骤启动

6. 配置 API 接口地址

用 VSCode 将项目打开，API 接口地址在文件 `src/app/consts/constants.ts` 中配置，默认 `domainUri=http://localhost:5000` 是 mock server 的接口地址，如果你使用的是后台项目的 API 接口，

在这里做相应的更改即可。

```
src > app > consts > TS constants.ts > ...  
1  const domainUri = 'http://localhost:5000';  
2
```

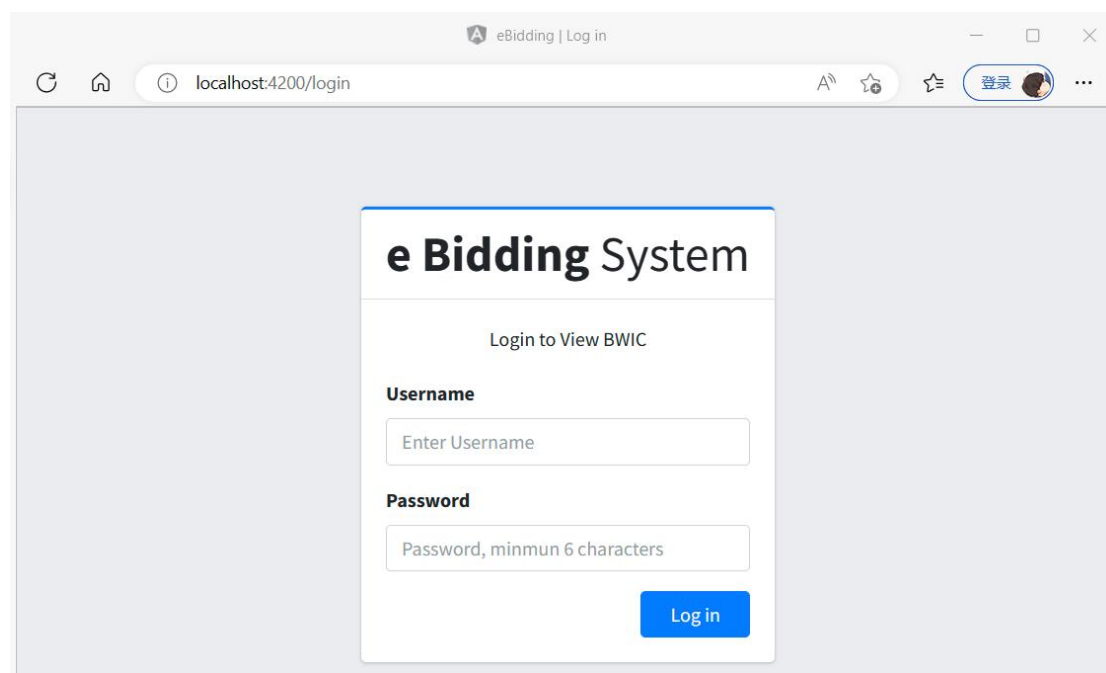
7. 启动项目

在命令行中执行`>ng serve`，启动项目。Angular 项目默认端口号是 4200，如果想要指定特定端口（例如 4201）则执行`>ng serve --port 4201`

```
D:\01.Programming\01.Angular\e-bidding-system>ng serve  
✓ Browser application bundle generation complete.  
  
Initial Chunk Files | Names | Raw Size  
vendor.js | vendor | 2.33 MB  
polyfills.js | polyfills | 314.29 kB  
styles.css, styles.js | styles | 209.42 kB  
main.js | main | 63.39 kB  
runtime.js | runtime | 6.53 kB  
Initial Total | 2.91 MB  
  
Build at: 2023-03-06T15:41:55.580Z - Hash: 3899d7abf3524336 - Time: 7249ms  
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **  
  
✓ Compiled successfully.
```

8. 访问项目

打开浏览器（推荐使用 chrome 或 edge）输入：<http://localhost:4200>，回车。如果见到如下登录页面，代表你的参考 demo 项目已经可以正常运行。



接下来我们查看 API 请求的发送情况。例如当你输入完用户名 sasa 和密码，点击 “Log in” 按钮，我们可以在浏览器中看到一个登录请求已经发出。

The screenshot displays a web browser at `localhost:4200/dashboard`. The page shows a user profile for 'Alexander Pierce' and a table titled 'BWIC Publish List'. The table contains one entry with a 'Winner: Alpha' status. The browser's developer tools are open, showing the 'Network' tab. A request to `http://localhost:5000/client/login/sasa` is highlighted, with a status of 200 OK. The request details show the URL, method (GET), status code, remote address, and referer policy.

#	Cusip	Due Date	Position	Market Price(\$)	Staring Market Value(\$)	NO. of Bids	My Ranking	My Bid Market Value(\$)	Action
1	037833100	2023-02-20	400	30	10000	10	3/5	10500	Winner: Alpha

Network Tab Details:

- 名称: sasa
- 请求 URL: `http://localhost:5000/client/login/sasa`
- 请求方法: GET
- 状态代码: 200 OK
- 远程地址: `[::1]:5000`
- 引用者策略: strict-origin-when-cross-origin