



西安电子科技大学
XIDIAN UNIVERSITY

Web 安全 产品文档

作业内容:

小组作业 1

小组成员:

姜欣悦

李颖

职泉

陈静怡

盖乐

实验日期:

2023.12.27

目录

web 应用开发用到的技术	3
一、 前端---Angular	3
二、 依赖管理相关---NVM 与 NPM	6
三、 后端---Spring Boot	8
四、 数据库---SQL	10
部署与环境配置	11
应用功能的使用步骤图解	15
一、 登录页面	15
二、 E-commerce（电子商务）	15
三、 lot Dashboard(物联网控制面板)	18
四、 登录界面	20
接口设计	20
作业 1 小组分工	24

web 应用开发用到的技术

一、 前端---Angular

1.Angular 架构设计

Angular 框架属于 MEAN 框架，是如今创业公司最热门的技术栈。Angular 是一个完整的基于 TypeScript 的 Web 应用开发框架，主要用于构建单页 Web 应用（SPA）。

与 AngularJS 这一早期的框架不同，Angular2 是基于组件的，与 MV* 模式没有什么关联。Angular 的结构方式包括模块、组件和服务。

在 Angular 框架中，每个组件都有一个类或模板，定义了应用逻辑和 MetaData（装饰器）。组件的这些元数据为创建和呈现其视图所需的构件在哪里提供了指引。

Angular 架构的另一个重要因素是，模板是用 HTML 编写的。它们还可以包含 Angular 模板语法，并带有特殊指令以输出响应式数据，并且可以渲染多个元素。

服务 —— Angular 应用中的一个独特元素，被 Components 用于委托业务逻辑任务，如获取数据或验证输入。虽然使用服务并没有严格执行，但是将应用程序结构作为一组可复用的不同服务则是比较明智的。

2. Angular 适用目标和范围

Angular 最适合大型和高级项目。这些可能包括但不限于：

用于开发渐进式 Web 应用程序（PWA）。

用于重新设计网站应用程序。

用于建立基于内容的动态网页设计。

用于创建有着复杂基础架构的大型企业应用程序。

Angular 性能方面的一些亮点包括：

有无缝的第三方集成，以增强产品或应用程序的功能。

提供强大的组件集合，从而简化了编写，更改和使用代码的过程。

它的“提前编译器”赋予了应用程序更快的加载时间和安全性。

MVC 模型通过允许视图分离来帮助减少后台查询。

促进使用将依赖项注入的外部元素来让组件解耦，从而为可复用性以及简化管理和测试铺平了道路。

通过将任务分成逻辑块来减少网页的初始加载时间。

可以完全自定义的设计。

Angular 是一个开发平台，建立在打字稿。作为一个平台，Angular 包括：

基于组件的框架，用于构建可扩展的 Web 应用程序

一系列集成良好的库，涵盖各种功能，包括路由、表单管理、客户端-服务器通信等

一套开发人员工具，可帮助您开发、构建、测试和更新代码

借助 Angular，您可以利用一个可以从单个开发人员项目扩展到企业级应用程序的平台。Angular 旨在使更新尽可能简单，因此请以最小的努力利用最新的发展。最重要的是，Angular 生态系统由超过 170 万开发人员、库作者和内容创建者组成的多元化群体组成。

下面是一个最小的 Angular 组件。

```
import { Component } from '@angular/core';
```

```
@Component({
```

```
  selector: 'hello-world',
```

```
  template: `
```

```
    <h2>Hello World</h2>
```

```
    <p>This is my first component!</p>
```

```
  `
```

```
});
```

```
export class HelloWorldComponent {
```

```
  // The code in this class drives the component's behavior.
```

```
}
```

要使用此组件，请在模板中编写以下内容：

```
content_copy<hello-world></hello-world>
```

当 Angular 渲染这个组件时，生成的 DOM 如下所示：

```
content_copy<hello-world>
```

```
  <h2>Hello World</h2>
```

```
  <p>This is my first component!</p>
```

```
</hello-world>
```

Angular 的组件模型提供了强大的封装和直观的应用程序结构。组件还可以使应用程序无需单元测试，并且可以提高代码的一般可读性。

二、 依赖管理相关---NVM 与 NPM

nvm 全英文也叫 node.js version management，是一个 nodejs 的版本管理工具。nvm 和 n 都是 node.js 版本管理工具，为了解决 node.js 各种版本存在不兼容现象可以通过它可以安装和切换不同版本的 node.js。

Node.js 是 2009 年的时候由大神 Ryan Dahl 开发的。Ryan 的本职工作是用 C++ 写服务器，后来他总结出一个经验，一个高性能服务器应该是满足“事件驱动，非阻塞 I/O”模型的。C++ 开发起来比较麻烦，于是 Ryan 就想找一种更高级的语言，以便快速开发。

可以说有两点促成了 Nodejs 的诞生。首先第一点，Ryan 发现 JS 语言本身的特点就是事件驱动并且是非阻塞 I/O 的，跟他的思路正是绝配。第二点，Chrome 的 JS 引擎，也就是 V8 引擎是开源的，而且性能特别棒。于是 Ryan 就基于 V8 开发了 Node.js，注意 Node.js 听名字好像是个 JS 库，其实不是的，Node.js 是 C++ 开发的，到官网 <http://nodejs.org> 可以看到 Node.js 是一个基于 Chrome V8 引擎的 Javascript 运行环境

所以说 Node.js 不是库，是一个运行环境，或者说是一个 JS 语言解释器。

Node.js 最初的定位是提升 Ryan 自己的日常工作效率，也就是用来写服务器代码的，但是后来没有想到的是 Node.js 在前端领域却大放异彩。

先说几句 Node.js 在服务器端的发展。Node.js 的诞生带给人们的是个大大的惊喜，传统上 Web 开发者，前端用 JS 写，但是写服务器端代码的时候还必须用另外一种语言，类似 Ruby/Java/PHP 等。但是 Node.js 出现之后，JS 前后通吃了。如果去网上搜 Node.js 的资料，很多都是用 Node.js 去写服

务器代码的。

但是，Node.js 今天也成为了 Web 前端开发必不可少的基础设施。注意，Web 前端的 JS 代码最终还是运行在浏览器中的，所以运行的时候，或者说在产品环境下，不依赖于 Node.js。但是，Node.js 诞生以后，前端大爆发，类似 React/Vuejs 这样的前端框架的开发环境变得非常强大和负责，Node.js 是这些开发环境运行的基础。

npm 是两件事：首先，它是发布开源 Node.js 项目的在线存储库；其次，它是一个命令行实用程序，用于与所述存储库进行交互，有助于包安装、版本管理和依赖项管理。大量的 Node.js 库和应用程序发布在 npm 上，并且每天都在添加更多。可以在 <https://www.npmjs.com/> 上搜索这些应用程序。有了要安装的包后，可以使用单个命令行命令安装它。

假设您有一天正在努力工作，开发下一个伟大的应用程序。你遇到了一个问题，你决定是时候使用你不断听到的那个很酷的库了 - 让我们以 Caolan McMahon 的异步为例。值得庆幸的是，使用起来非常简单：你只需要运行，指定的模块就会安装在当前目录下。安装到您的文件夹后，您将能够像内置一样在它们上使用它们

npm 的另一个重要用途是依赖关系管理。当你有一个包含 package.json 文件的节点项目时，你可以从项目根目录运行，npm 将安装 package.json 中列出的所有依赖项。这使得从 git 存储库安装 Node.js 项目变得更加容易！例如，可以从 git 安装一个 Node.js 测试框架，并且可以自动处理其单个依赖项：

```
npm installvowseyes
```

例：

```
git clone https://github.com/cloudhead/vows.git
```

```
cd vows
```

```
npm install
```

运行这些命令后，您将看到一个文件夹，其中包含 `package.json` 中指定的所有项目依赖项。

三、 后端---Spring Boot

下面我们大概来了解一下 Spring Boot 的核心模块。

1、spring-boot

这是 Spring Boot 的主模块，也是支持其他模块的核心模块，主要包含以下几点：

提供了一个启动 Spring 应用的主类，并提供了一个相当方便的静态方法，它的主要作用是负责创建和刷新 Spring 容器的上下文；

内嵌式的并可自由选择搭配的 WEB 应用容器，如：Tomcat, Jetty, Undertow 等；

对配置外部化的支持；

提供一个很方便的 Spring 容器上下文初始化器，包括合理记录日志默认参数的支持。

2、spring-boot-autoconfigure

Spring Boot 能根据类路径下的内容自动配置一些公共大型应用，提供的 `@EnableAutoConfiguration` 注解就能启用 Spring 功能的自动配置。

自动配置功能可以推断用户可能需要加载哪些 Spring Bean，如：如果类路径下有 `HicariCP` 这个连接池的包，此时并未提供任何有效连接池的配置，那么 Spring Boot 就知道您可能需要一个连接池，并做相应配置。如果用户配置了其

他连接池，那么 Spring Boot 会放弃自动配置。

3、spring-boot-starters

Starters，我们叫它启动器好了，它是包括一系列依赖的描述符。简单的说就是，它可以一站式的帮你打包 Spring 及相关技术应用，而不需要你到处找依赖和示例配置代码，它都帮你做好了。

例如，第一章我们在介绍 Spring Boot 的时候就说了 spring-boot-starter-web 这个启动器，你只要引用了这个启动器应用，就会自动配置 WEB 应用的能力。

spring-boot-starters 这个启动器这主要提供了 spring-boot, spring-context, spring-beans 这三个 Spring 模块而已。

4、spring-boot-cli

这是 Spring Boot 的命令行工具，用于编译和运行 Groovy 源程序，可以十分简单的编写并运行一个应用程序。它也能监控你的文件，一旦有变动就会自动重新编译和重新启动应用程序。

5、spring-boot-actuator

这是 Spring Boot 提供的执行端点，你可以更好的监控及和你的应用程序交互。这个模块提供了像健康端点、环境端点、Spring Bean 端点等。

6、spring-boot-actuator-autoconfigure

这个原理同上，为 Spring Boot 执行端点提供自动配置。

7、spring-boot-test

Spring Boot 测试模块，为应用测试提供了许多非常有用的核心功能。

8、spring-boot-test-autoconfigure

这个原理同上，为 Spring Boot 测试模块提供自动配置。

9、spring-boot-loader

这个模块可以用来构建一个单独可执行的 jar 包,使用 `java -jar` 就能直接运行。一般不会直接使用这个来打包,使用 Spring Boot 提供的 Maven 或者 Gradle 插件就行了。

10、spring-boot-devtools

开发者工具模块,主要为 Spring Boot 开发阶段提供一些特性,如修改了代码自动重启应用等。这个模块的功能是可选的,只限于本地开发阶段,当打成整包运行时这些功能会被禁用。

四、 数据库---SQL

SQL (Structured Query Language) 是具有数据操纵和数据定义等多种功能的数据库语言,这种语言具有交互性特点,能为用户提供极大的便利,数据库管理系统应充分利用 SQL 语言提高计算机应用系统的工作质量与效率。SQL 语言不仅能独立应用于终端,还可以作为子语言为其他程序设计提供有效助力,该程序应用中,SQL 可与其他程序语言一起优化程序功能,进而为用户提供更多更全面的信息。 [1]

SQL Server 数据库包括 Microsoft SQL Server 以及 Sybase SQL Server 两个子数据库,该数据库能否正常运行直接关系着整个计算机系统的运行安全。

SQL 语言的组成:

- 1.一个 SQL 数据库是表(Table)的集合,它由一个或多个 SQL 模式定义。
- 2.一个 SQL 表由行集构成,一行是列的序列(集合),每列与行对应一个数据项。
- 3.一个表或者是一个基本表或者是一个视图。基本表是实际存储在数据库

的表，而视图是由若干基本表或其他视图构成的表的定义。

4.一个基本表可以跨一个或多个存储文件，一个存储文件也可存放一个或多个基本表。每个存储文件与外部存储上一个物理文件对应。

5.用户可以用 SQL 语句对视图和基本表进行查询等操作。在用户角度来看，视图和基本表是一样的，没有区别，都是关系(表格)。

6.SQL 用户可以是应用程序，也可以是终端用户。SQL 语句可嵌入在宿主语言的程序中使用，宿主语言有 FORTRAN, COBOL, PASCAL, PL/I, C 和 Ada 语言等。SQL 用户也能作为独立的用户接口，供交互环境下的终端用户使用。

部署与环境配置

一. (推荐)安装 nvm (node.js version management) 以便进行版本控制，但也可以省略这一步，直接进行 nodejs 与 npm 的安装

二. 在中 nvm 安装 nodejs、npm(运行环境+包管理)，此处使用的 npm 版本为 6.14.11，nodejs 版本为 v14.16.0，可以保证项目文件在此环境下正常运行。通过 “-version” 验证安装如下：

```
PS C:\Users\Kyriota> npm --version
6.14.11
PS C:\Users\Kyriota> node --version
v14.16.0
```

三. 进入项目文件夹，并运行如下命令来安装依赖文件并编译项目：

```
npm install --registry=http://registry.npm.taobao.org
```

本过程通常需要较长(约 5~10 分钟)，完整输出过长，在此截几个中间过程以作参考，只要全程不报 Error 即可：

```

ngcc
> node scripts/build.js

Building: E:\Program Files\nodejs\node.exe E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-gyp\bin\node-gyp.js rebuild --verbose --libsass_ext= --libsass_cflags= --libsass_ldflags= --libsass_library=
gyp info it worked if it ends with ok
gyp verb cli [
gyp verb cli   'E:\Program Files\nodejs\node.exe',
gyp verb cli   'E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-gyp\bin\node-gyp.js',
gyp verb cli   'rebuild',
gyp verb cli   '--verbose',
gyp verb cli   '--libsass_ext=',
gyp verb cli   '--libsass_cflags=',
gyp verb cli   '--libsass_ldflags=',
gyp verb cli   '--libsass_library='
gyp verb cli ]
gyp info using node-gyp@3.8.0
gyp info using node@14.16.0 | win32 | x64
gyp verb command rebuild []
gyp verb command clean []
gyp verb clean removing "build" directory
gyp verb command configure []
gyp verb check python checking for Python executable "python2" in the PATH
gyp verb `which` succeeded python2 E:\CTFTOOL\Python27\python2.EXE
gyp verb check python version 'E:\CTFTOOL\Python27\python2.EXE -c "import sys; print "2.7.18"' returned: %j
gyp verb check python version %.%.s" % sys.version_info[:3];" returned: %j
gyp verb get node dir no --target version specified, falling back to host node version: 14.16.0
gyp verb command install [ '14.16.0' ]
gyp verb install input version string "14.16.0"
gyp verb install installing version: 14.16.0
gyp verb install --ensure was passed, so won't reinstall if already installed
gyp verb install version is already installed, need to check "installVersion"

```

```

ngcc
Link:
E:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Tools\MSVC\14.16.27023\bin\HostX64\x64\link.exe /ERR
ORREPORT:QUEUE /OUT:"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\Release\binding.node" /NOLOGO k
ernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib o
dbc32.lib DelayImp.lib "C:\Users\Kyriota\node-gyp\14.16.0\x64\node.lib" Delayimp.lib /DELAYLOAD:iojs.exe /DELAYLOAD:
node.exe /MANIFEST /MANIFESTUAC:"Level='asInvoker' uiAccess='false'" /manifest:embed /DEBUG /PDB:"E:\DL-Tencent\ngx-a
dmin-starter-kit\node_modules\node-sass\build\Release\binding.pdb" /TLBID:1 /RELEASE /DYNAMICBASE /NXCOMPAT /MACHINE:
X64 /ignore:4199 /DLL Release\obj\binding\binding.obj
Release\obj\binding\create_string.obj
Release\obj\binding\custom_function_bridge.obj
Release\obj\binding\custom_importer_bridge.obj
Release\obj\binding\sass_context_wrapper.obj
Release\obj\binding\boolean.obj
Release\obj\binding\color.obj
Release\obj\binding\error.obj
Release\obj\binding\factory.obj
Release\obj\binding\list.obj
Release\obj\binding\map.obj
Release\obj\binding\null.obj
Release\obj\binding\number.obj
Release\obj\binding|string.obj
Release\obj\binding\win_delay_load_hook.obj
"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\Release\libsass.lib"
正在创建库 E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\Release\binding.lib 和对象 E:\DL-Tencent\
ngx-ad
min-starter-kit\node_modules\node-sass\build\Release\binding.exp
binding.vcxproj -> E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\Release\binding.node
FinalizeBuildStatus:
正在删除文件"Release\obj\binding\binding.tlog\unsuccessfulbuild"。
正在对"Release\obj\binding\binding.tlog\binding.lastbuildstate"执行 Touch 任务。
已完成生成项目"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\binding.vcxproj"(默认目标)的操作。

```

```

ngeo
已用时间 00:00:57.75
gyp info ok
Installed to E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\vendor\win32-x64-83\binding.node

> ngx-admin@6.0.0 postinstall E:\DL-Tencent\ngx-admin-starter-kit
> ngcc --properties es2015 es5 browser module main --first-only --create-ivy-entry-points --tsconfig "./src/tsconfig.app.json"

Compiling @angular/core : es2015 as esm2015
Compiling @angular/cdk/keycodes : es2015 as esm2015
Compiling @angular/animations : es2015 as esm2015
Compiling @angular/compiler/testing : es2015 as esm2015
Compiling @angular/cdk/collections : es2015 as esm2015
Compiling @angular/common : es2015 as esm2015
Compiling @angular/cdk/observers : es2015 as esm2015
Compiling @angular/animations/browser : es2015 as esm2015
Compiling @angular/core/testing : es2015 as esm2015
Compiling @angular/cdk/platform : es2015 as esm2015
Compiling @angular/platform-browser : es2015 as esm2015
Compiling @angular/cdk/bidi : es2015 as esm2015
Compiling @angular/cdk/portal : es2015 as esm2015
Compiling @angular/forms : es2015 as esm2015
Compiling @angular/cdk/a11y : es2015 as esm2015
Compiling @angular/router : es2015 as esm2015
Compiling @angular/common/http : es2015 as esm2015
Compiling @angular/platform-browser-dynamic : es2015 as esm2015
Compiling @angular/platform-browser/testing : es2015 as esm2015
Compiling @angular/cdk/scrolling : es2015 as esm2015
Compiling @angular/cdk/table : es2015 as esm2015
Compiling @angular/common/testing : es2015 as esm2015

```

常见问题:

- 需要 Visual Studio 2017 的生成工具(平台工具集 =“v141”)等编译器, 参考如下博客 <https://blog.csdn.net/logocool/article/details/122303991>

```

ngeo
admin-s
starter-kit\node_modules\node-sass\build\binding.vcxproj.metaproj"(2) (默认目标)。
项目"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\binding.vcxproj.metaproj"(2)正在节点 1 上生成"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\src\libsass.vcxproj"(3) (默认目标)。
E:\Program Files (x86)\Microsoft Visual Studio\2019\Community\MSBuild\Microsoft\VC\v160\Microsoft.CppBuild.targets(439, 5): error MSB8020: 无法找到 Visual Studio 2017 的生成工具(平台工具集 =“v141”)。若要使用 v141 生成工具进行生成, 请安装 Visual Studio 2017 生成工具。或者, 可以升级到当前 Visual Studio 工具, 方式是通过选择“项目”菜单或右键单击该解决方案, 然后选择“重定解决方案目标”。 [E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\src\libsass.vcxproj]
已完成生成项目"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\src\libsass.vcxproj"(默认目标)的操作 - 失败。

已完成生成项目"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\binding.vcxproj.metaproj"(默认目标)的操作 - 失败。

已完成生成项目"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\binding.sln"(默认目标)的操作 - 失败。

生成失败。

"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\binding.sln"(默认目标) (1) ->
"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\binding.vcxproj.metaproj"(默认目标) (2) ->
"E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sass\build\src\libsass.vcxproj"(默认目标) (3) ->
(PreparesForBuild 目标) ->
E:\Program Files (x86)\Microsoft Visual Studio\2019\Community\MSBuild\Microsoft\VC\v160\Microsoft.CppBuild.targets(439, 5): error MSB8020: 无法找到 Visual Studio 2017 的生成工具(平台工具集 =“v141”)。若要使用 v141 生成工具进行生成, 请安装 Visual Studio 2017 生成工具。或者, 可以升级到当前 Visual Studio 工具, 方式是通过选择“项目”菜单或右键单击该解决方案, 然后选择“重定解决方案目标”。 [E:\DL-Tencent\ngx-admin-starter-kit\node_modules\node-sa

```

安装的具体依赖列表如下:

```

"@angular/animations": "^10.0.10",
"@angular/cdk": "10.1.1",
"@angular/common": "^10.0.10",
"@angular/compiler": "^10.0.10",
"@angular/core": "^10.0.10",
"@angular/forms": "^10.0.10",
"@angular/platform-browser": "^10.0.10",
"@angular/platform-browser-dynamic": "^10.0.10",
"@angular/router": "^10.0.10",
"@nebular/auth": "6.0.0",
"@nebular/eva-icons": "6.0.0",

```

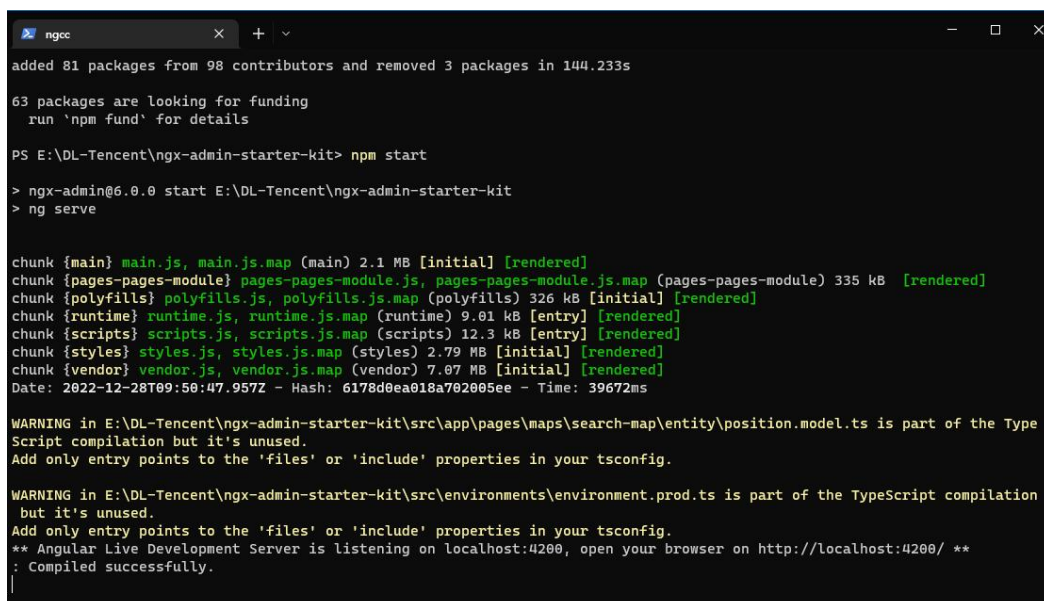
```
"@nebular/security": "6.0.0",
"@nebular/theme": "6.0.0",
"bootstrap": "4.3.1",
"classlist.js": "1.1.20150312",
"core-js": "2.5.1",
"eva-icons": "^1.1.3",
"intl": "1.2.5",
"ionicons": "2.0.1",
"nebular-icons": "1.1.0",
"node-sass": "^4.12.0",
"normalize.css": "6.0.0",
"pace-js": "1.0.2",
"roboto-fontface": "0.8.0",
"rxjs": "6.6.2",
"rxjs-compat": "6.3.0",
"socicon": "3.0.5",
"style-loader": "^1.1.3",
"tslib": "^2.0.0",
"typeface-exo": "0.0.22",
"web-animations-js": "^2.3.2",
"zone.js": "~0.10.2"
```

1. 以上 package 安装完成后，使用如下命令来启动项目

```
npm start
```

"Compiled successfully"字样后，访问 <http://localhost:4200/pages/dashboard>

测试是否启动成功



```
ngcc
added 81 packages from 98 contributors and removed 3 packages in 144.233s

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

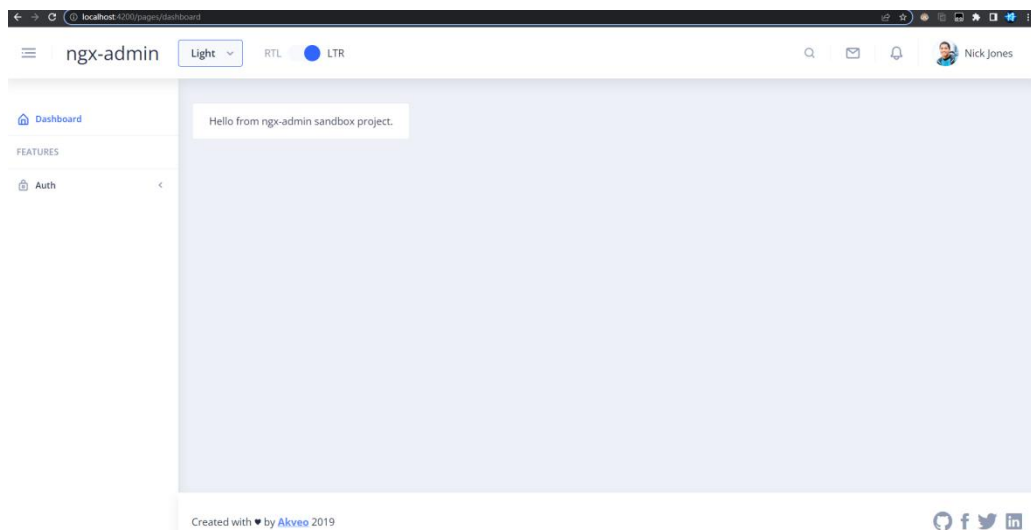
PS E:\DL-Tencent\ngx-admin-starter-kit> npm start

> ngx-admin@6.0.0 start E:\DL-Tencent\ngx-admin-starter-kit
> ng serve

chunk {main} main.js, main.js.map (main) 2.1 MB [initial] [rendered]
chunk {pages-pages-module} pages-pages-module.js, pages-pages-module.js.map (pages-pages-module) 335 kB [rendered]
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 326 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 9.01 kB [entry] [rendered]
chunk {scripts} scripts.js, scripts.js.map (scripts) 12.3 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 2.79 MB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 7.07 MB [initial] [rendered]
Date: 2022-12-28T09:50:47.957Z - Hash: 6178d0ea018a702005ee - Time: 39672ms

WARNING in E:\DL-Tencent\ngx-admin-starter-kit\src\app\pages\maps\search-map\entity\position.model.ts is part of the Type
Script compilation but it's unused.
Add only entry points to the 'files' or 'include' properties in your tsconfig.

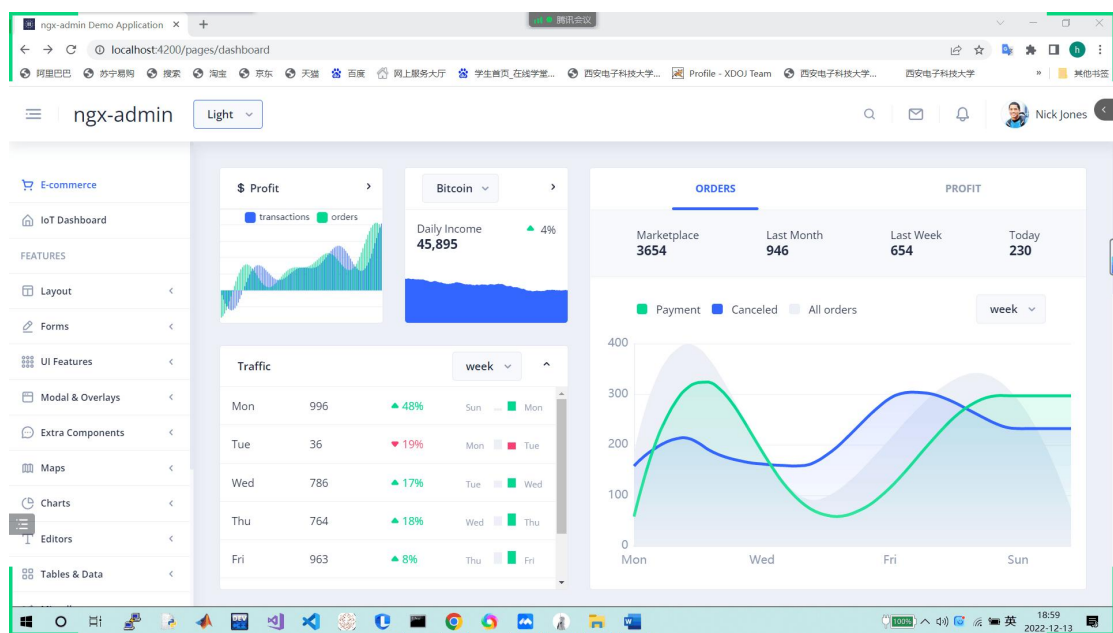
WARNING in E:\DL-Tencent\ngx-admin-starter-kit\src\environments\environment.prod.ts is part of the TypeScript compilation
but it's unused.
Add only entry points to the 'files' or 'include' properties in your tsconfig.
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
: Compiled successfully.
```

至此，项目的安装与部署工作全部完成

应用功能的使用步骤图解

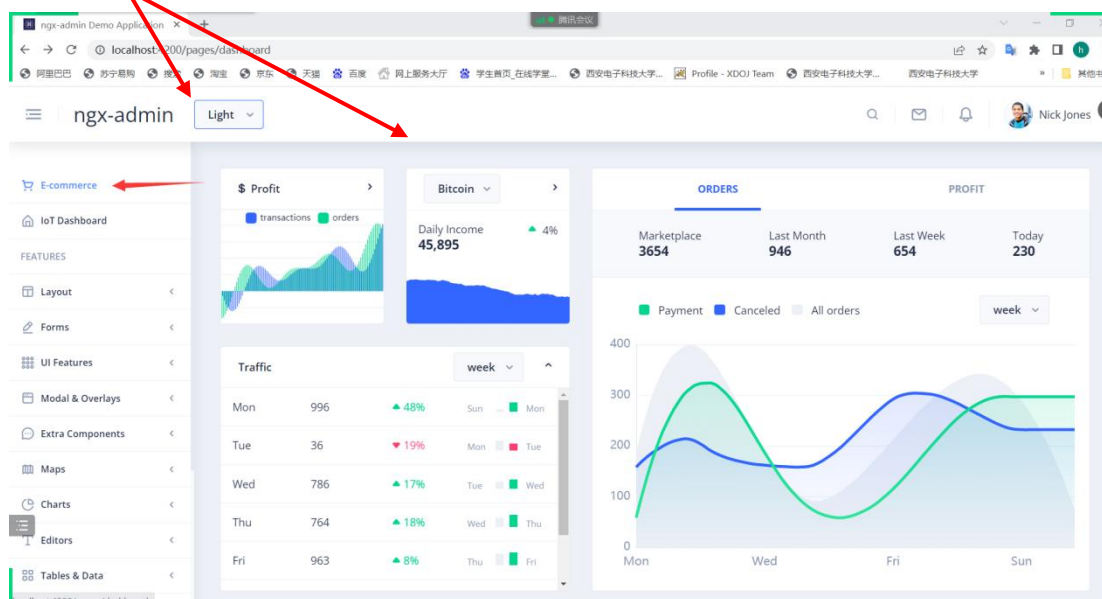
一、 登录页面



二、 E-commerce（电子商务）

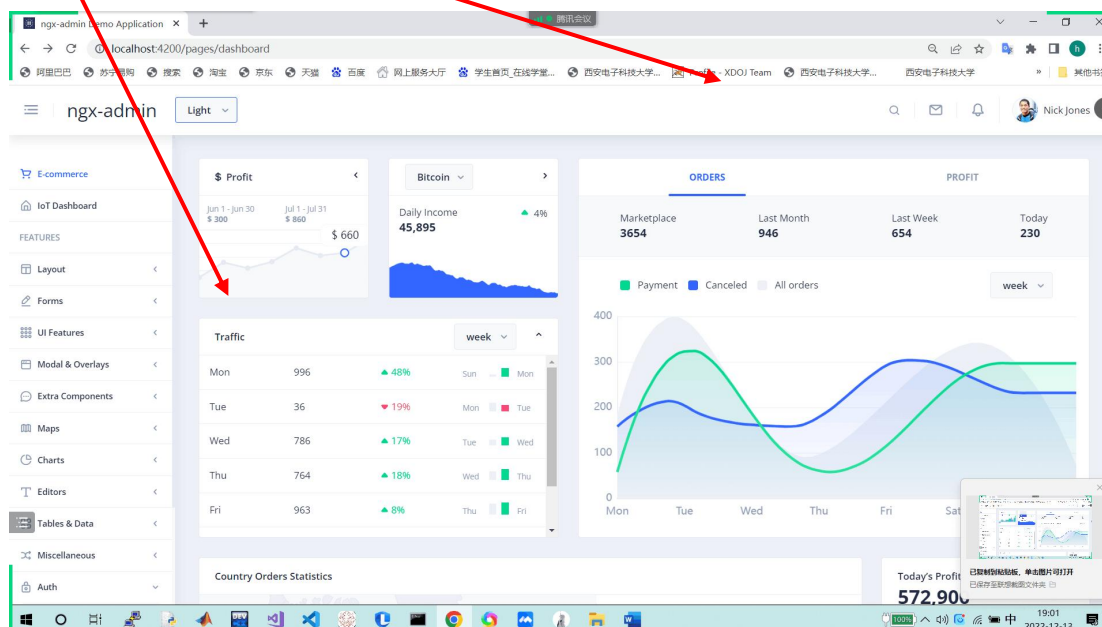
1. 内含一段时间的交易量和订单量的变化图

2. 选择货币的种类及当天收入曲线

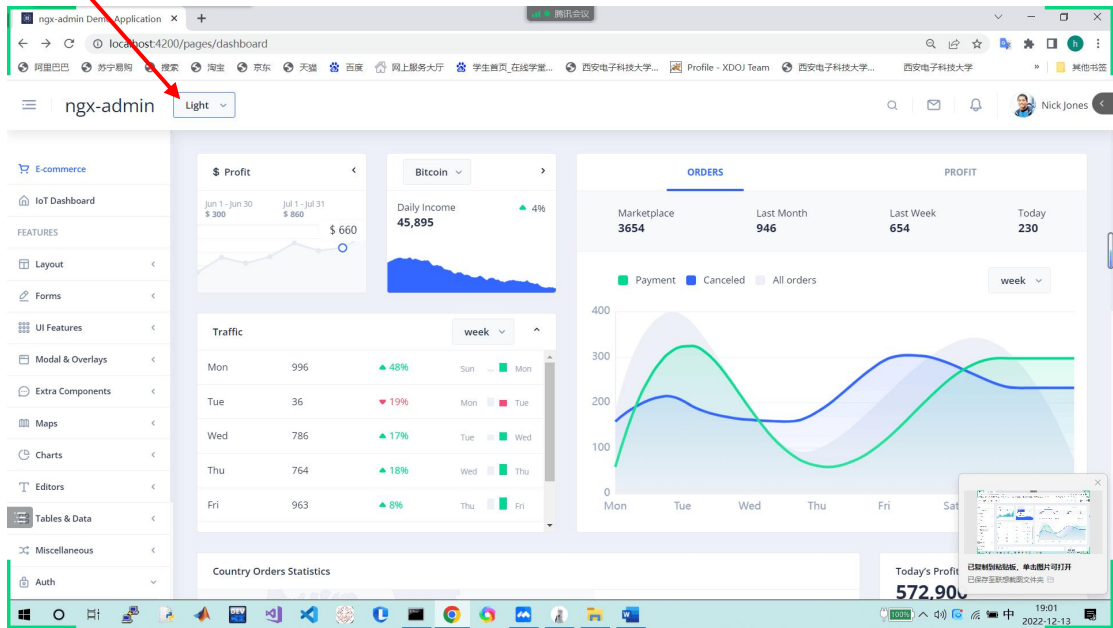


3. 市场具体的交易情况图

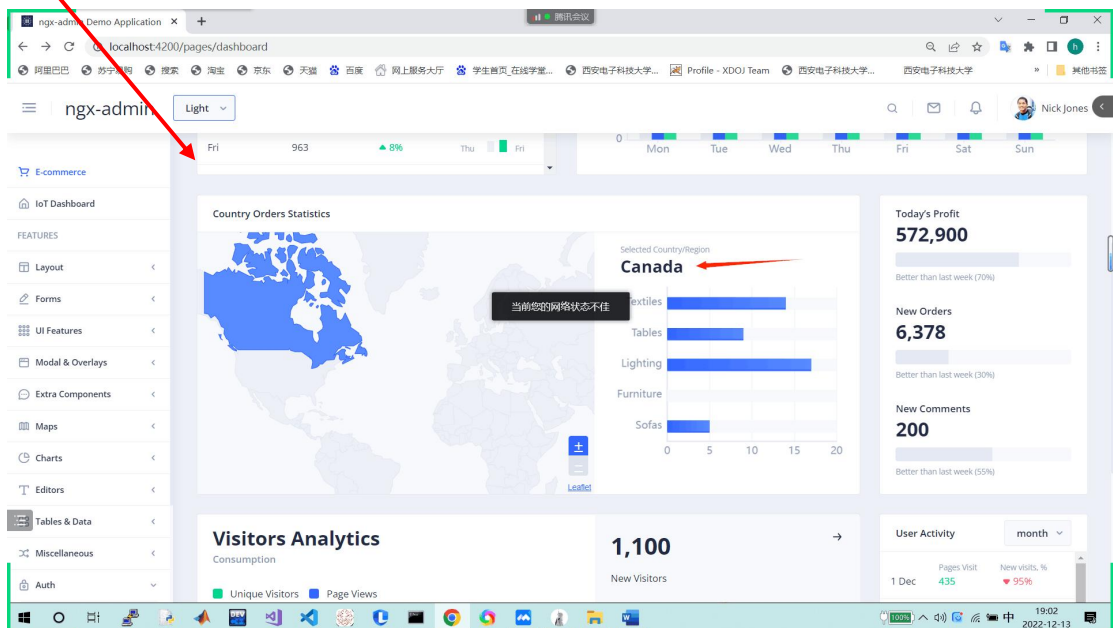
4. 每周每日的流量图



点击后可查看具体情况

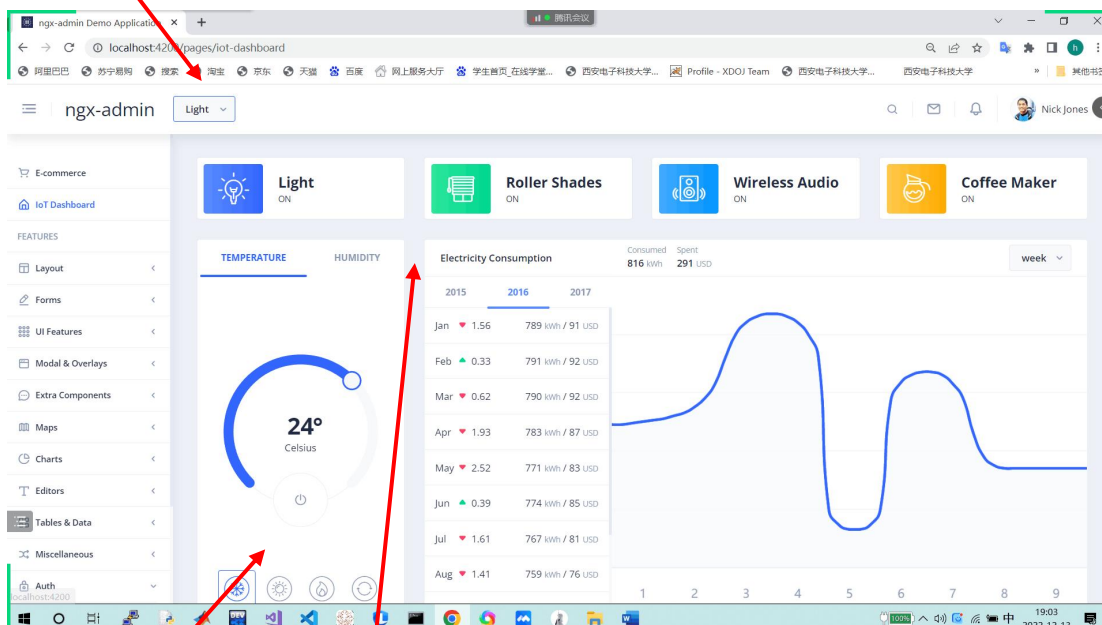


5. 选择城市数据



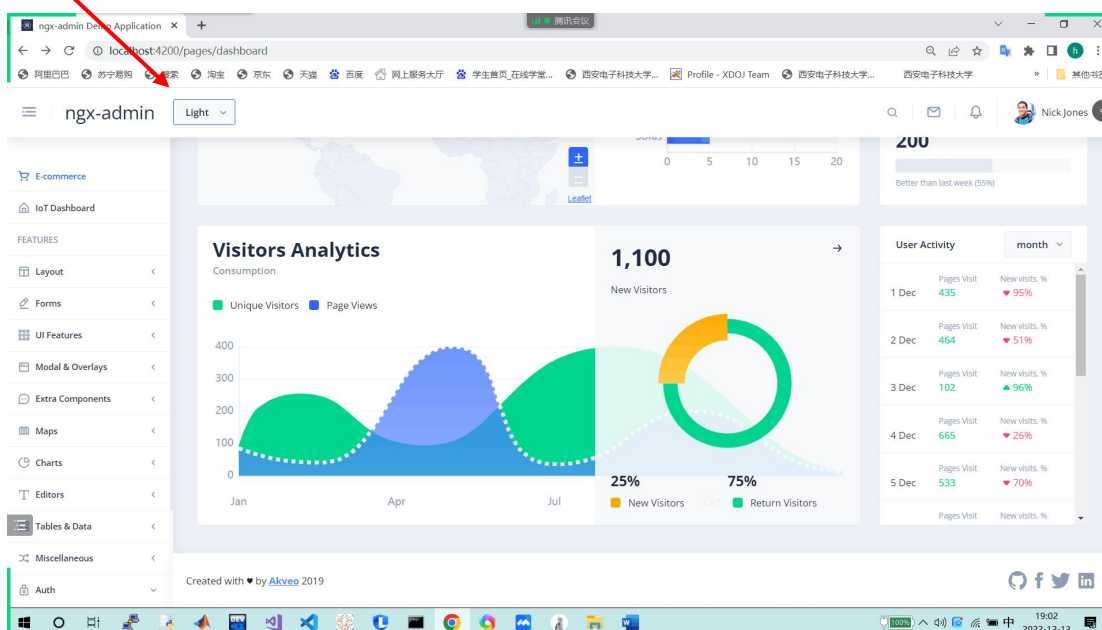
三、 IoT Dashboard(物联网控制面板)

a. 灯光控制 b.窗帘 c.无限音频 d.咖啡机

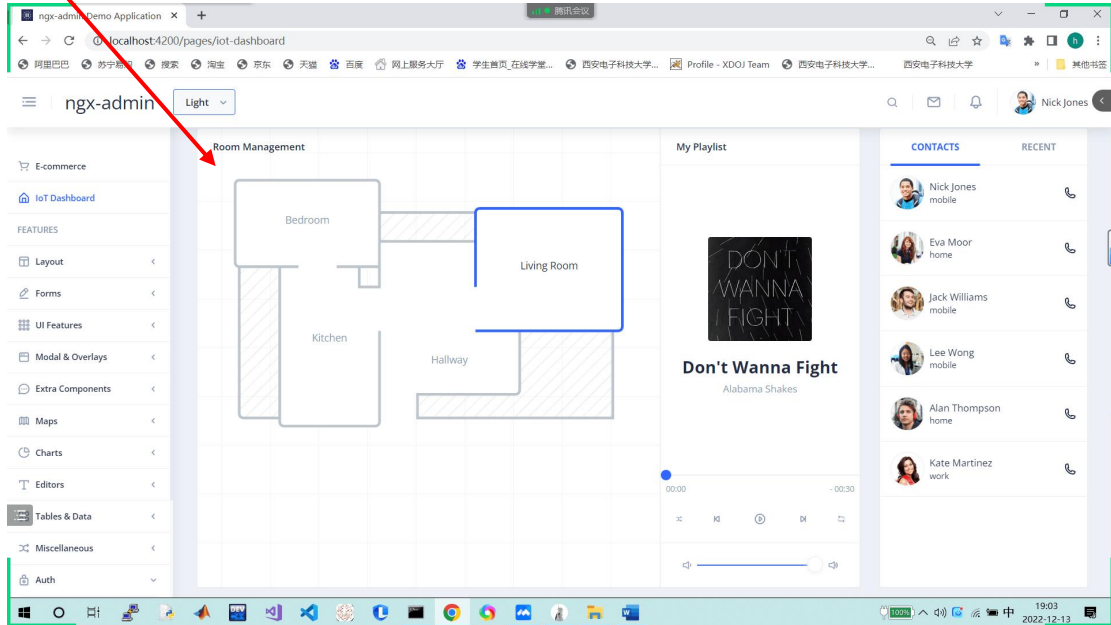


温度控制 电量使用情况图

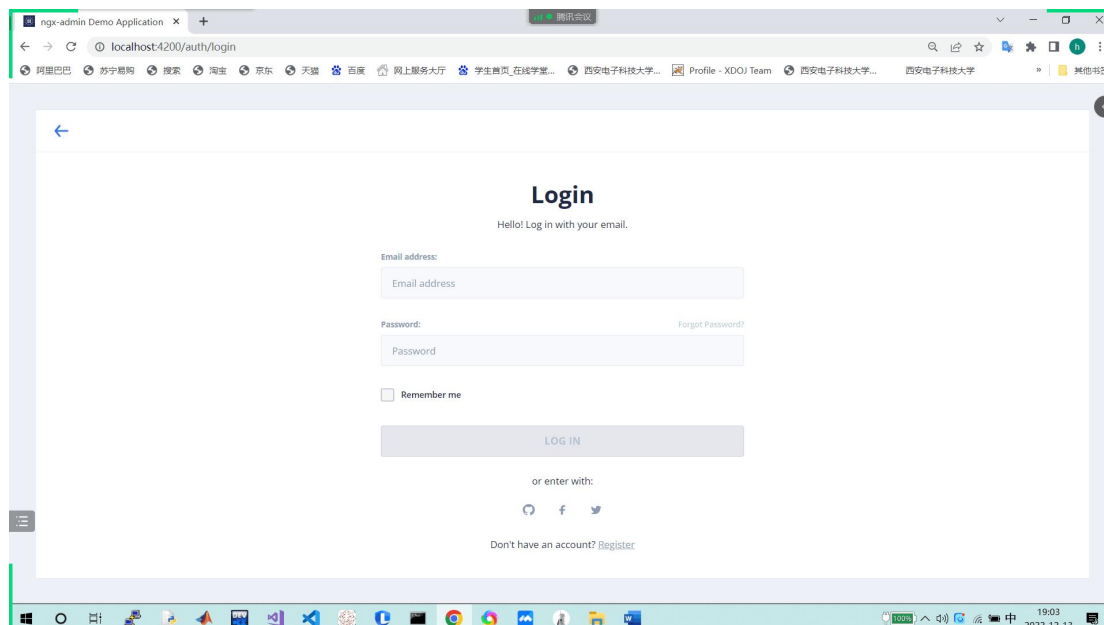
控制整个页面的灯光



房间管理示意图



四、 登录界面

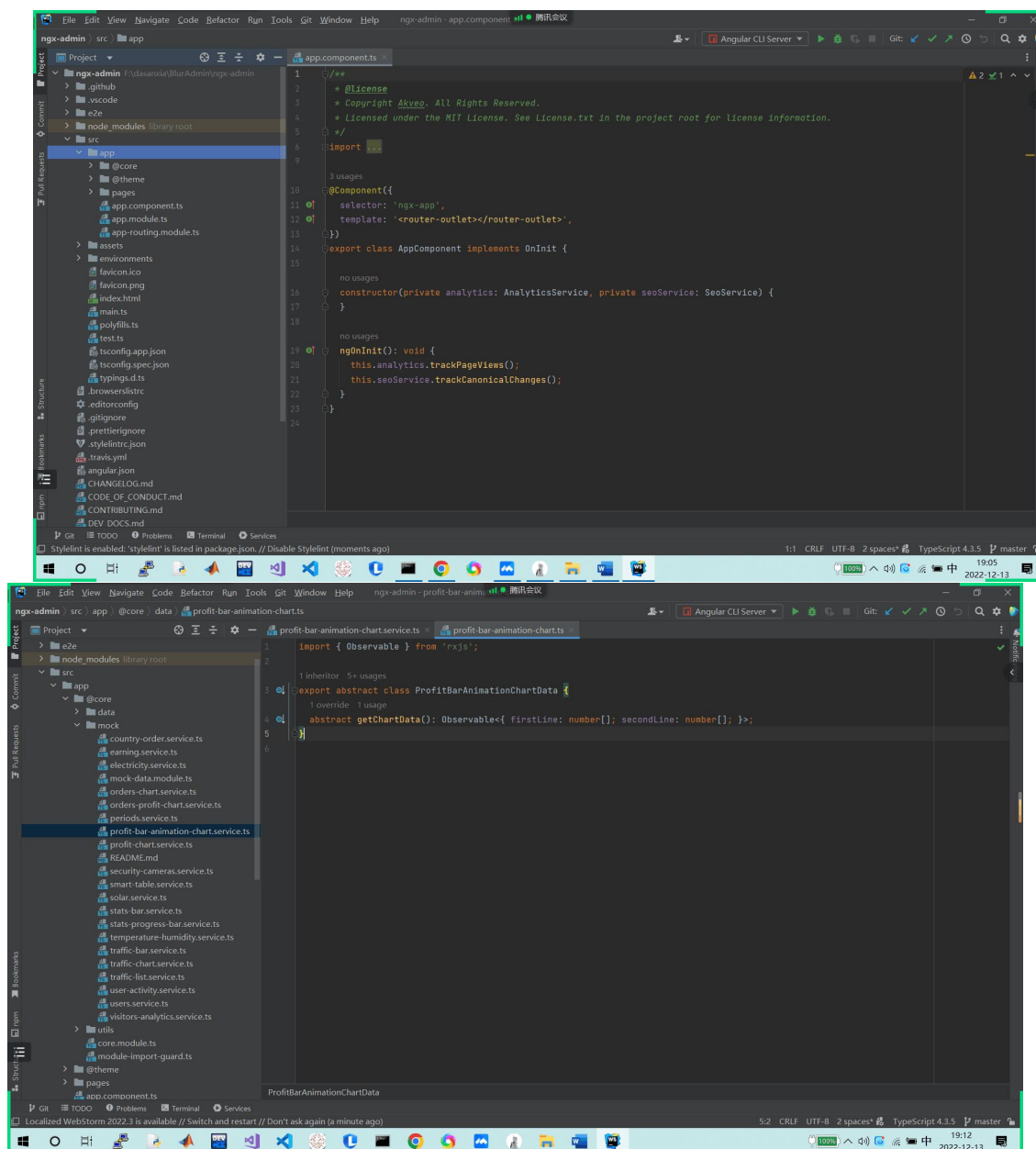


接口设计

（由于本项目的接口设计比较复杂，在下举例解释说明三个最具代表性的思想）

接口一：

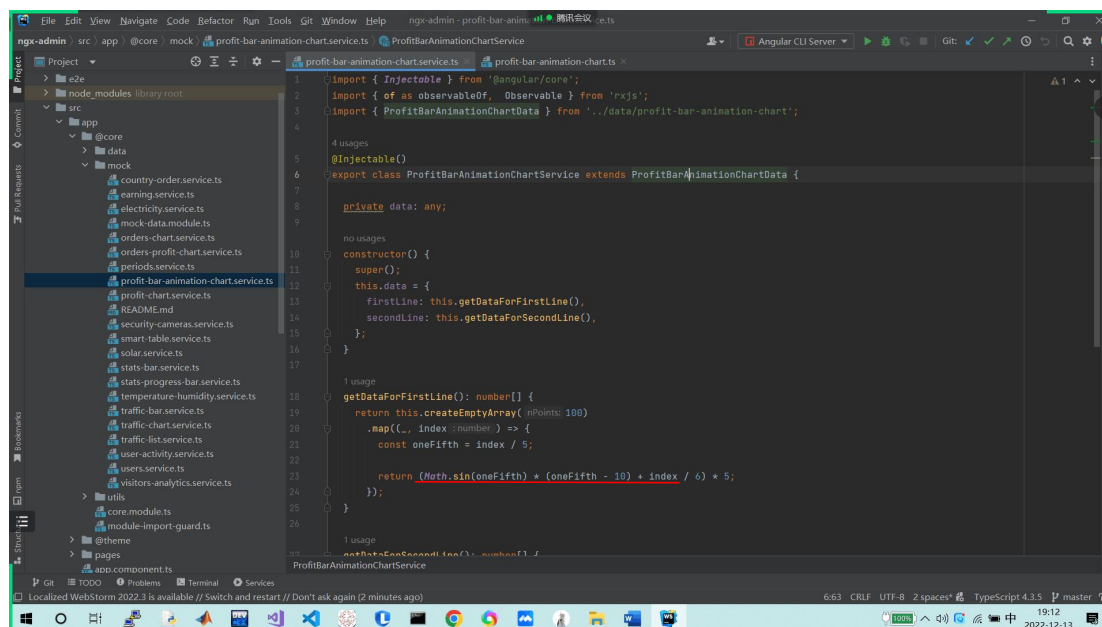
Src 是最大的文件夹，所有的文件都保存在该文件夹，剩余的都是支撑 Src 的文件。App 内包含 Core Data Mock 等等。其中 Core 内包含核心代码，Data 内涵数据，Mock 包含动态模拟数据 物联网数据等等 Theme 包含 web 的主题：颜色 风格等等 Utils 以.Ts 为后缀是 web 的类其中 Scss 规定样式以及 Css+Html 来规定网页语言的展示格式以及标记语言的格式。



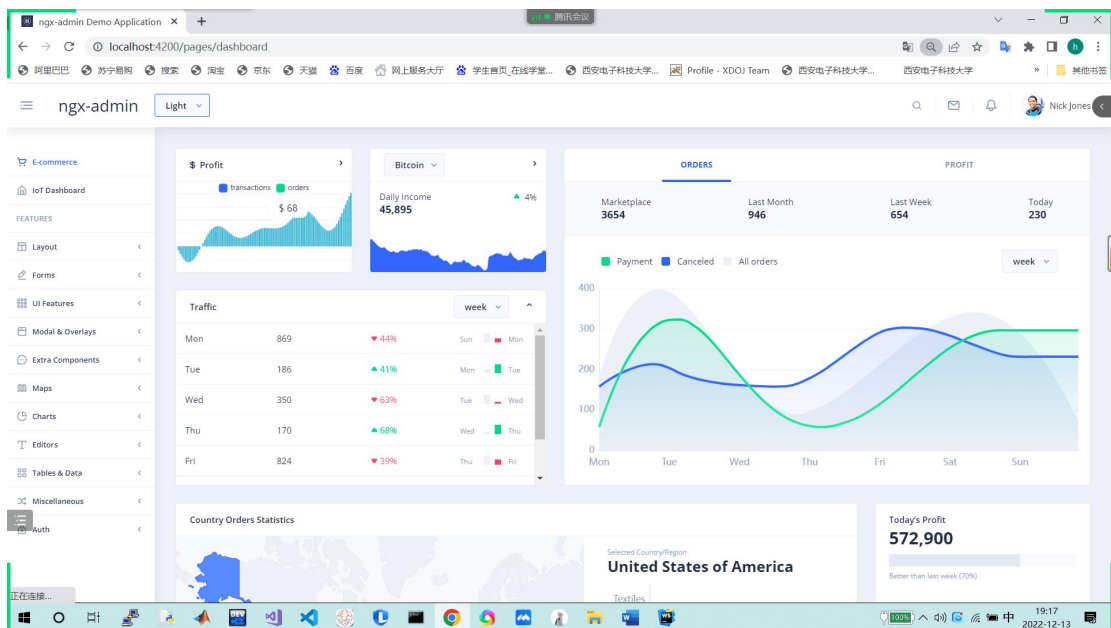
接口二：

在 date 文件内含有获取数据的总方法，为了满足需求创建 mock 文件夹，其内部也是一些获取数据的方法，每一个方法都继承了 data 的内方法，在此基础上进行扩展，这样即可满足要求。在前端获取数据的过程中，需要进行修改，则只需修改 mock 内部的函数，data 内部的不需要修改。

此方法便于前端在连接本地数据库或者从网页中获取数据亦或实时输入数据，则仅需在 mock 文件夹进行修改，与原有的 data 不冲突，进行特异性改变来适应程序需求。



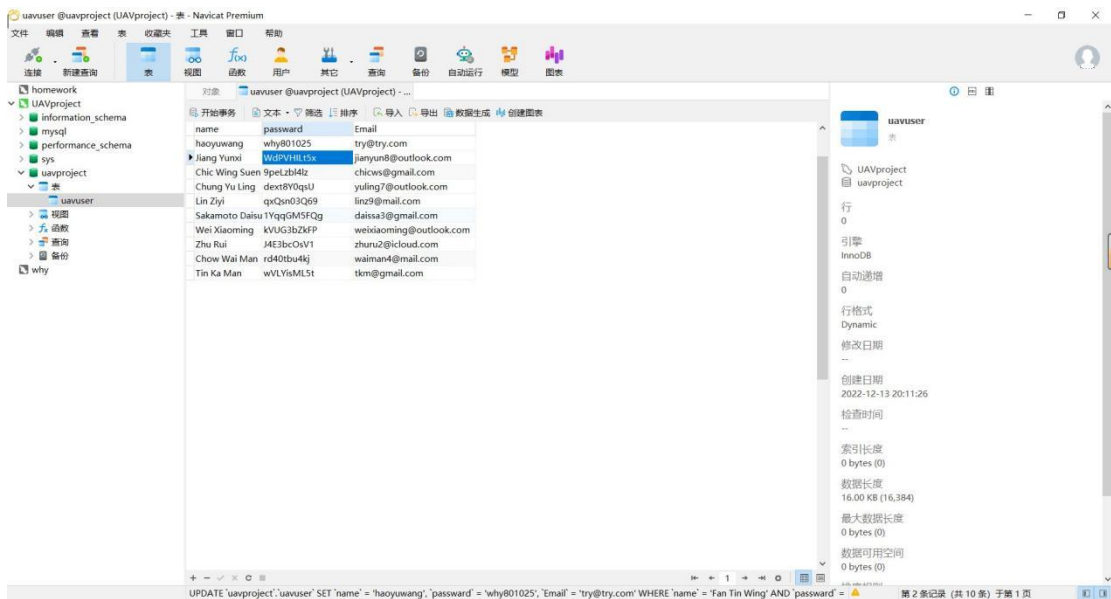
举例如下：在需要满足需求时直接在 mock 文件夹内部修改函数：cos 改为 sin 来达到函数图像重合的目的。



接口三：后端与数据库

下图是建立的数据库，我们将后端与数据库相连实现登录功能

在前端先向后端发送前端收到的用户名以及密码，在后端向数据库发送在数据库内进行 select 是否有相应的用户名及密码。若有，则向前端发送一个 true 的 https 协议，则实现登录过程。



作业 1 小组分工

姜欣悦 20%	Web 页面主体制作
李颖 20%	前端的信息搜集
职泉 20%	后端的信息搜集
陈静怡 20%	产品文档信息收集与查找 作业版式制作
盖乐 20%	调试与运行