Описание решения

Содержание

Оощее описание	1
ьаза данных	2
Серверная часть	
Клиентская часть	
Тестирование	6

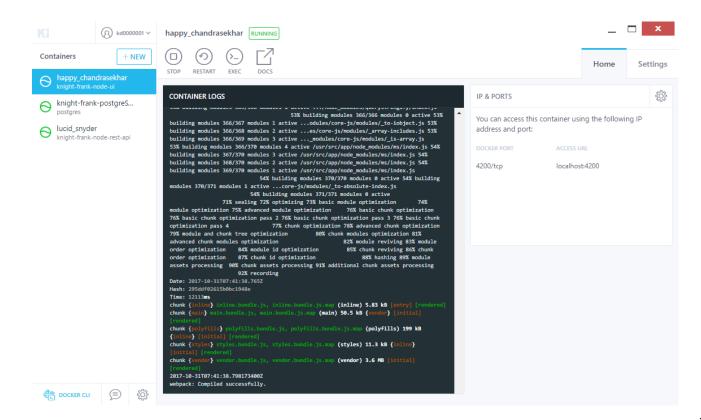
Общее описание

Приложение, опубликованное в облаке (<u>digitalocean.com</u>), можно увидеть по ссылке: http://138.68.185.190:4200

Решение размещено по трём Docker-контейнерам Linux, содержащим каждый уровень приложения. Для развёртывания достаточно запустить соответствующие Dockerfile (для запуска Dockerfile, в той же папке, имеется соответствующий ему файл DockerStart.cmd, для запуска контейнера):

- \Database\Scripts\Dockerfile сервер базы данных PostgreSQL
- \Service\Rest\Dockerfile веб-сервис Node.js
- \UI\ Dockerfile
 интерфейс пользователя Angular 4, на веб-сервере Node.js

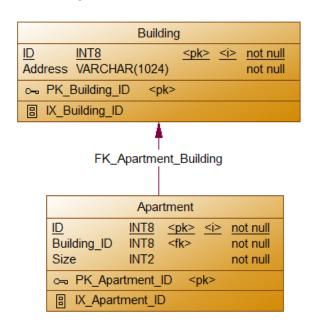
Используя Kitematic, мы можем наблюдать три запущенных контейнера:



База данных

В прилагаемом архиве папка \Database

Согласно требованию, используется PostgreSQL, привожу диаграмму физической модели (Sybase PowerDesigner):



В прилагаемом архиве (папка "Database") имеется диаграмма и скрипты pdplSQL создания структуры таблиц, а также генерации тестовых данных, со случайным их распределением.

Серверная часть

В прилагаемом архиве папка \Service\Rest

Согласно требованию, используется Node.js в качестве платформы для REST-сервиса. Используются библиотеки Express.js и основанная на ней Loopback.io, добавляющая возможности ORM-системы для реляционных данных, а также возможность генерации VIEW-модели.

Loopback позволяет как читать данные из реляционных таблиц, так и выполнять хранимые функции и SQLзапросы с параметрами, при необходимости выборок на стороне сервера.

Для данного решения это оказалось необходимым, т.к. для выборок понадобилась агрегация и соединение таблиц, что оптимально делать в реляционной СУБД, а не на стороне сервиса.

Структура серверной части проекта:

```
⊕ ÷ | ♣ - ⊩
                                               server.js × us building.js × uson building.json × uson datasources.json × uson apartment.json ×
■ Rest [WebApi] D:\Work\KnightFrank\Service\Rest
                                                       'use strict';
  ... server
                                                      var loopback = require('loopback');
                                               3
     - boot
                                               4
                                                      var boot = require('loopback-boot');
         authentication.js
                                               5
                                               6
                                                      var app = module.exports = loopback();
     .... models
         apartment.js
                                               8
                                                     papp.start = function() {
                                                        // start the web server
                                               9
          building.js
           building.json
                                                        return app.listen(function() {
                                                          app.emit('started');
        component-config.json
                                                          var baseUrl = app.get('url').replace(/\/$/, '');
        config.json
        datasources.json
                                              13
                                                          console.log('Web server listening at: %s', baseUrl);
        middleware.development.json
                                              14
                                                          // loopback routing:
         middleware.json
                                              15
                                                          if (app.get('loopback-component-explorer')) {
        model-config.json
                                                            var explorerPath = app.get('loopback-component-explorer').mountPath;
                                              16
        server.js
                                              17
                                                            console.log('Browse your REST API at %s%s', baseUrl, explorerPath);
     .editorconfig
     .eslintianore
                                              19
                                                           // custom routing:
     eslintrc
                                                          app.post('/api/requests', function(req, res) {
     gitignore.
                                                            res.send(200, 'Просмотр успешно заказан');
     .yo-rc.json
Dockerfile
                                                          1);
                                                    A};
                                              23
                                                        });
     Dockerignore
                                              24
     DockerStart.cmd
     package.json
                                              25
                                              26
                                                     -// Bootstrap the application, configure models, datasources and middleware.
       package-lock.json
  ||||| External Libraries
                                              27
                                                     // Sub-apps like REST API are mounted via boot scripts.
                                                     boot(app, __dirname, function(err) {
   if (err) throw err;
                                              28
                                              29
                                              30
                                                         // start the server if `$ node server.js`
                                              31
                                              32
                                                        if (require.main === module)
                                                         app.start();
                                              33
                                              34
                                              35
```

Клиентская часть

В прилагаемом архиве папка \UI

Структура клиентской части проекта:

```
app.module.ts ×
■ UI D:\Work\KnightFrank\UI
                                                            import { BrowserModule } from '@angular/platform-browser';
  ⊕ dist
                                                            import { NgModule } from '@angular/core';
  ... e2e
                                                            import { FormsModule, ReactiveFormsModule } from '@angular/forms';
  in node_modules library root
                                                            import { HttpModule, JsonpModule } from '@angular/http';
  i src
                                                            import { BrowserAnimationsModule } from '@angular/platform-browser/animations
                                                    5
     6
        i components
                                                           import { AppRoutes } from './app.routing';
           - apartments
                apartment-list.component.html
                                                    8
                 apartment-list.component.ts
                                                    9
                                                            import { AppComponent } from './app.component';
                                                            import { BuildingListComponent } from './components/buildings/building-list co
           🖮 🖿 buildinas
                building-list.component.html
                                                            import { ApartmentListComponent } from './components/apartments/apartment-1.st
                 abuilding-list.component.ts
           🚊 ... 📗 layout
                                                            import { AppMenuComponent } from './components/layout/app.menu.component';
               ---- = app.content.component.ts
                                                           import { AppContentComponent } from './components/layout/app.content.component
                                                   14
                 app.menu.component.ts
                                                   15
              base.component.ts
                                                            import { NgbModule } from '@ng-bootstrap/ng-bootstrap';
        models
              apartment.model.ts
                                                   18
                                                            import { BuildingService } from './services/buildings/building.service';
              base.model.ts
                                                            import { ApartmentService } from './services/apartment/apartment.service';
              🚚 building.model.ts
                                                            import { OrderService } from './services/order/order.service';
              arder.model.ts
        ... services
                                                            @NgModule({
           ≟... apartment
                                                              declarations: [
              .... apartment.service.ts
                                                   24
                                                                AppComponent,
           i- buildings
                                                                AppMenuComponent,
              building.service.ts
                                                   26
                                                                AppContentComponent.
           i order
              ---- and order.service.ts
                                                               BuildingListComponent,
              ase.service.ts
                                                               ApartmentListComponent
           app.component.css
                                                   29
                                                              1.
             app.component.html
                                                              imports: [
             app.component.spec.ts
                                                                BrowserModule,
             app.component.ts
                                                               BrowserModule,
           app.module.ts
                                                               BrowserAnimationsModule,
             app.routing.ts
                                                                FormsModule,
                                                                ReactiveFormsModule,
                           +- - 2 辛 | - - -
                                                               HttpModule,
□ UI\package.json
                                                                JsonpModule,
      o na
                                                               NgbModule.forRoot(),

    start

                                                   39
                                                               AppRoutes
      build
                                                             providers: [BuildingService, ApartmentService, OrderService],
                                                   41
     prod
                                                   42
                                                             bootstrap: [AppComponent]
     test
                                                    43
     lint
                                                   44
                                                            export class AppModule { }

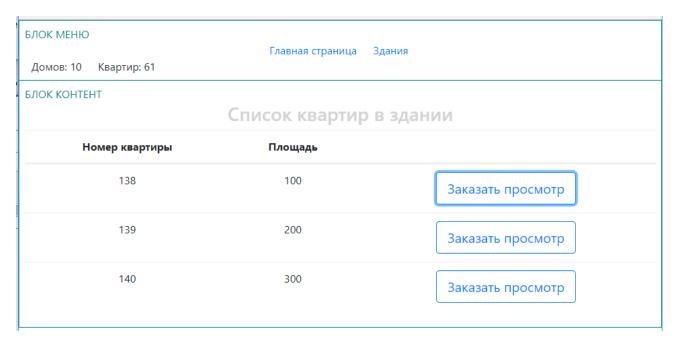
    e2e

                                                   45
```

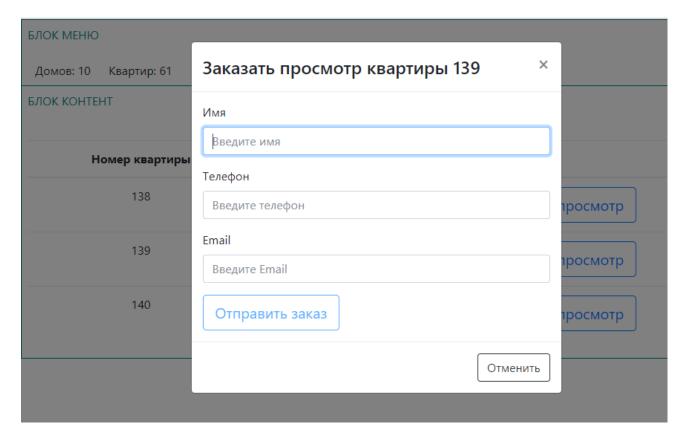
Ниже скриншот формы "Список зданий, в html-разметке специально показаны границы областей – "БЛОК МЕНЮ" и "БЛОК КОНТЕНТ" (в контенте располагается router-outlet, в который подгружаются все прочие компоненты, вызываемые из меню):

БЛОК МЕНЮ Главная страница Здания Домов: 10 Квартир: 61			
Список зданий			
Идентификатор дома	Адрес дома	Количество квартир в доме	
64	Address 64	8	
65	Address 65	3	
66	Address 66	3	
67	Address 67	6	
68	Address 68	7	
69	Address 69	3	
70	Address 70	7	
71	Address 71	8	
72	Address 72	6	
73	Address 73	10	

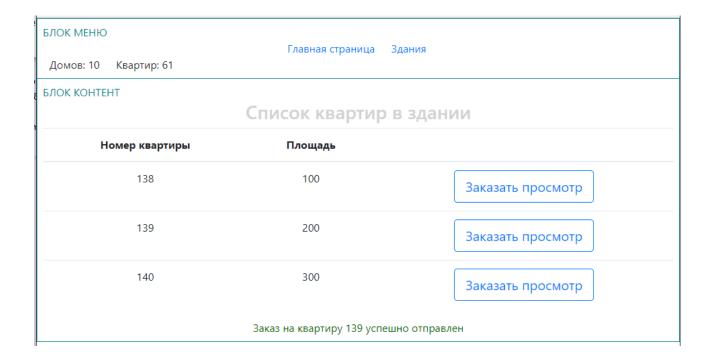
Ниже скриншот формы со списком квартир, вызываемой по клику на идентификаторе дома:



По клику на кнопке "Заказать просмотр" открывается модальное окно ng-bootstrap с формой, позволяющей оправить привязанную к форме модель Order методом Post на Rest-сервис.



В случае успешной отправки, (после отработки Promise), на форме будет показано сообщение об этом.



Тестирование