#### Министерство науки и высшего образования Российской Федерации

#### Федеральное государственное бюджетное образовательное учреждение

#### высшего образования

#### «Владимирский государственный университет

#### имени Александра Григорьевича и Николая Григорьевича Столетовых»

**(ВлГУ)**

**Кафедра информационных систем и программной инженерии**

Лабораторная работа №8

по дисциплине

"Основы разработки веб-приложений"

Выполнил:

ст. гр. ПРИ-117

Подолец А.А.

Принял:

Козаев А.

Владимир, 2020 г.

## **Цель работы**

Научиться работать с Headless CMS.

**Ход работы**

Вариант 13 (1)

Материалы находятся: <https://github.com/taki-jeden-karol/ORWP>

Устанавливаем Strapi (фреймворк для управления контентом). Для этого устанавливаем Node 8, если не установлен, MongoDB. Вводим в командную строку: npm i strapi -g

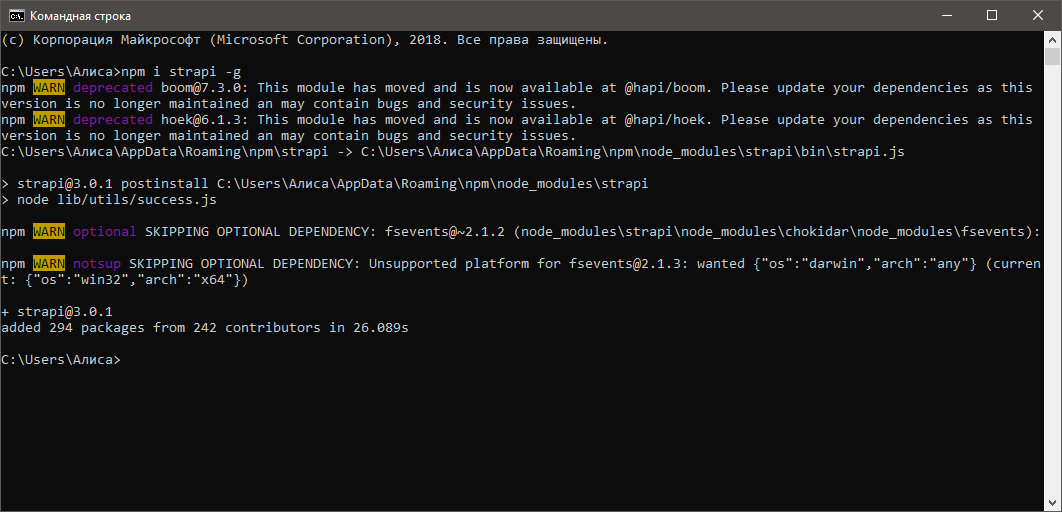


Рисунок 1. Установка strapi

Далее создаем директорию для лабораторной, переходим в нее и выполним команду strapi new strapi. После чего запускаем сервер командой strapi start

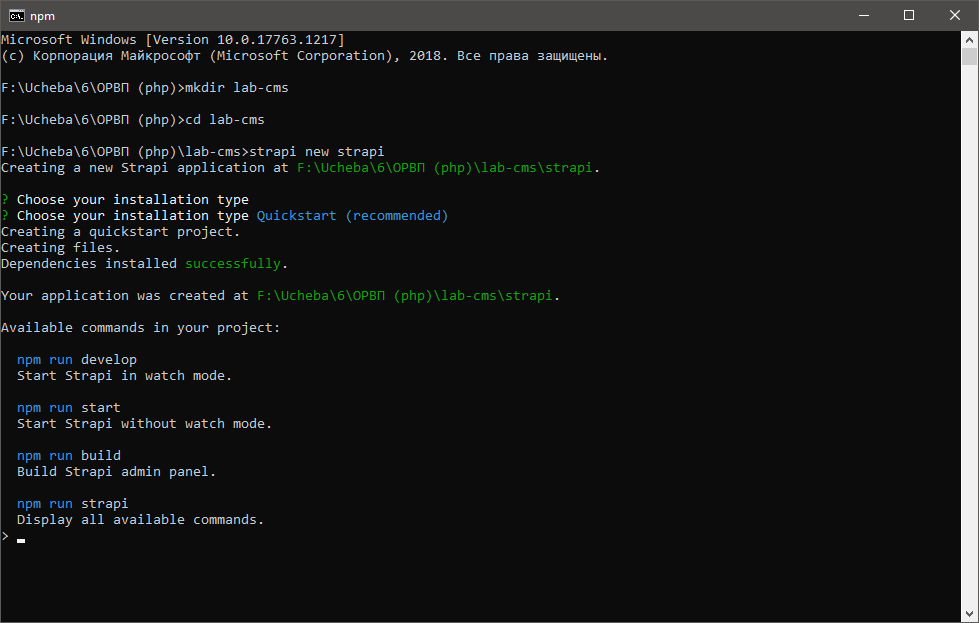


Рисунок 2. Установка директории и запуск сервера

Откроется панель администрирования проекта, создаем своего пользователя.

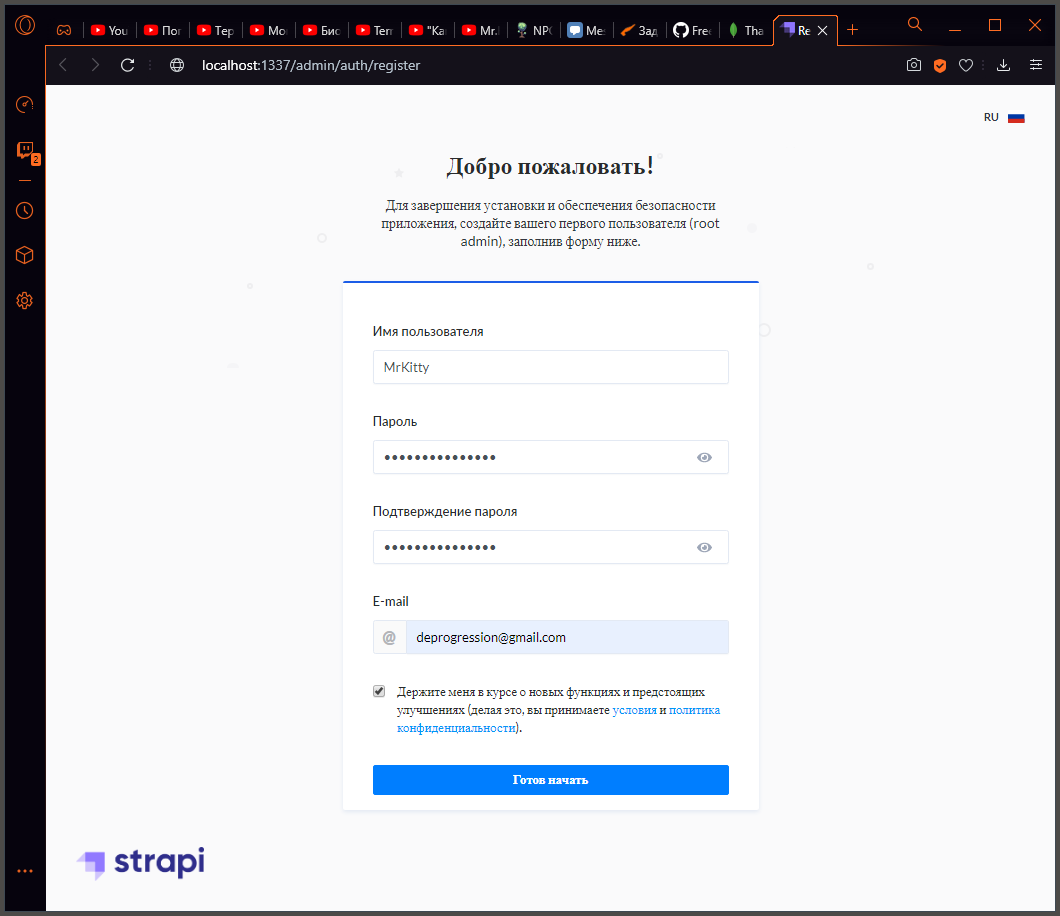


Рисунок 3. Создание пользователя

Далее нажимаем кнопку «Создание типа контента», задаем ему имя

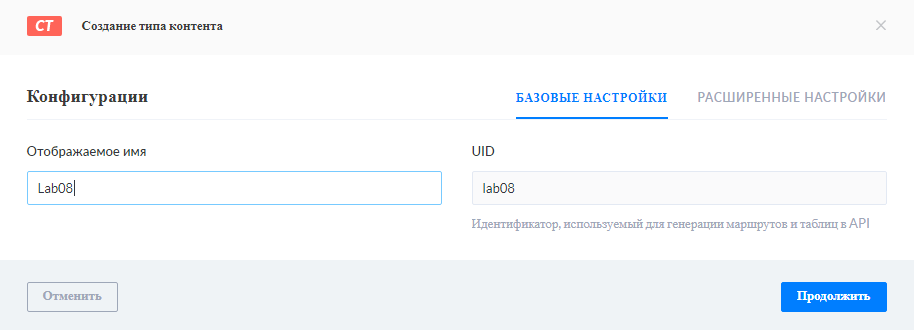


Рисунок 5. Создание типа контента

Выбираем тип поля text

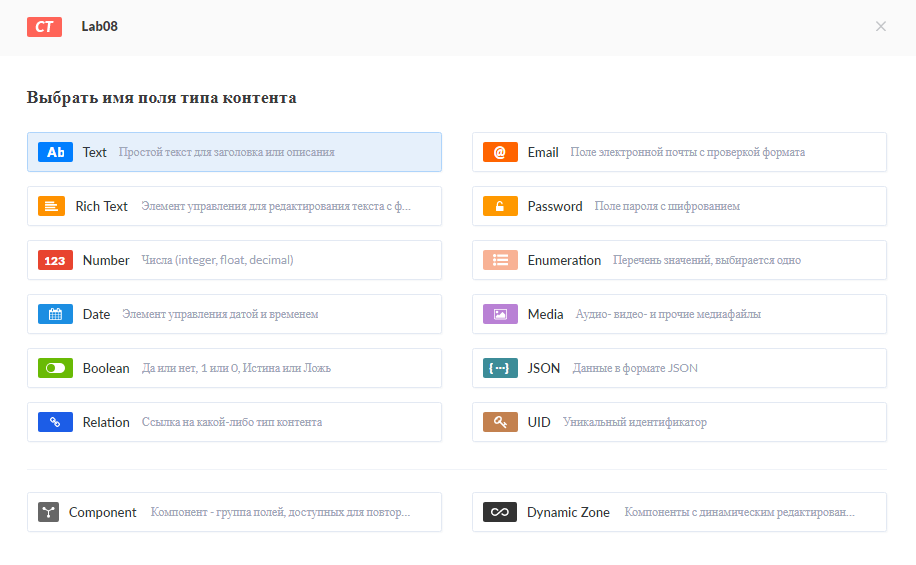


Рисунок 9. Задание типа поля контента

Вводим название для поля

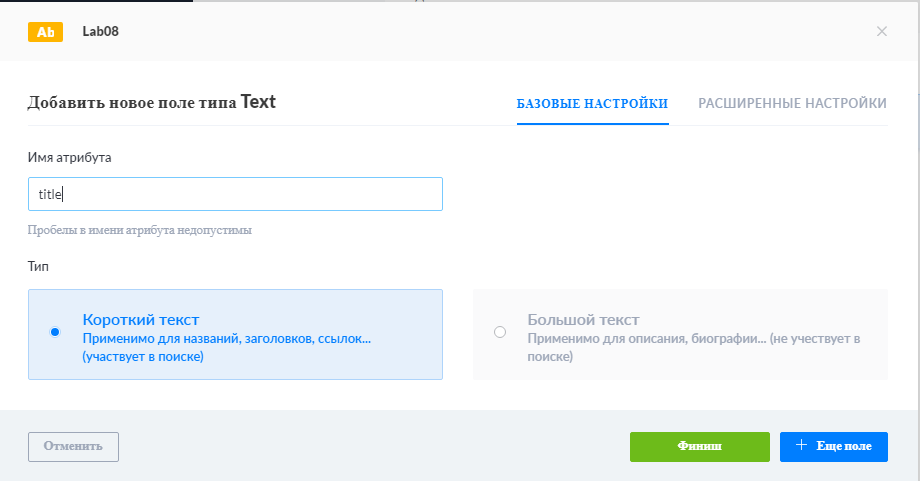


Рисунок 10. Ввод названия

В ролях добавляем доступ для данного контента

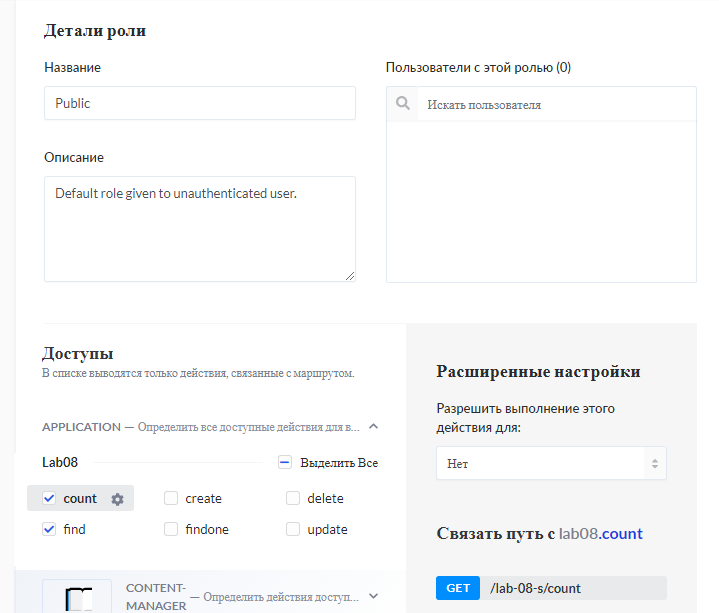


Рисунок 11. Задание доступа

Проверяем работоспособность

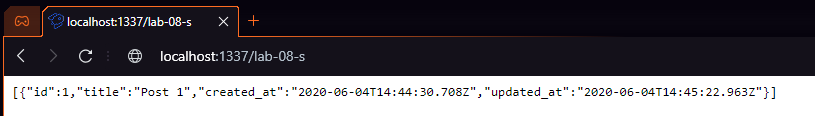


Рисунок 12. Проверка работоспособности

Далее создаем три таблицы, связываем их. В моем случае это таблицы Пациент (имя, вид), Врач (имя), Лист болезней (ссылается на пациента и врача)

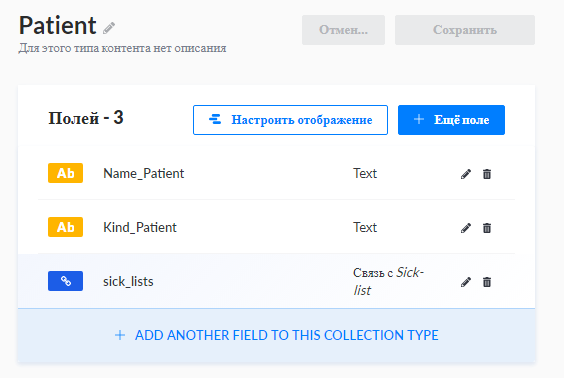


Рисунок 13. Таблица «Пациент»

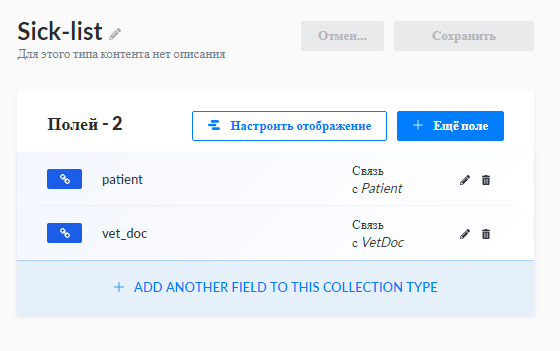


Рисунок 13. Таблица «Лист болезней»

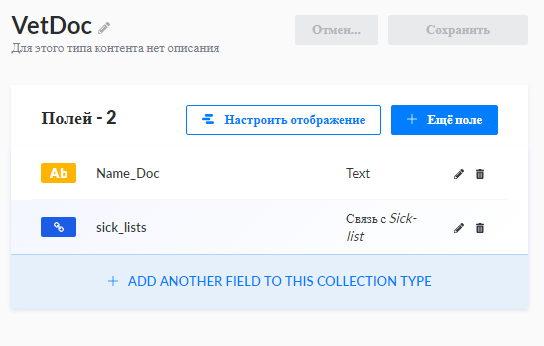


Рисунок 14. Таблица «ВетВрач»

Для фронтенда используем Angular. Создаем компоненты для сущностей. Создаем интерфейсы сущностей.

export interface IVetDoc {

    id?: number;

    Name\_Doc?: string;

    sick\_lists?: ISickList;

}

export interface ISickList {

    id?: number;

    patient?: IPatient;

    vetdoc?: IVetDoc;

}

export interface IPatient {

    id?: number;

    Name\_Patient?: string;

    Kind\_Patient?: string;

    sick\_lists?: ISickList;

}

Настраиваем роут

const routes: Routes = [

  {

    path: 'patient',

    component: PatientComponent

  },

  {

    path: 'sick-list',

    component: SickListComponent

  },

  {

    path: 'vet-doc',

    component: VetDocComponent

  },

  {

    path: 'patients/:id',

    component: OnePatientComponent

  },

  {

    path: 'sick-lists/:id',

    component: OneSickListComponent

  },

  {

    path: 'vet-docs/:id',

    component: OneVetDocComponent

  }

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

Компонент получения всех пациентов

import { Component, OnInit } from '@angular/core';

import { IPatient } from './patient.model';

import { HttpResponse, HttpClient } from '@angular/common/http';

@Component({

  selector: 'patient',

  templateUrl: './patient.component.html',

  styleUrls: ['./patient.component.css']

})

export class PatientComponent implements OnInit {

  patients: IPatient[];

  constructor(private http: HttpClient) { }

  ngOnInit(): void {

    this.http.get<IPatient[]>("http://localhost:1337/patients", {observe: "response"}).subscribe(

      (res: HttpResponse<IPatient[]>) => {

        console.log(res.body);

        this.patients = res.body}

      );

  }

}

Компонент получения всех вет.врачей

import { Component, OnInit } from '@angular/core';

import { IVetDoc } from './vet-doc.model';

import { HttpResponse, HttpClient } from '@angular/common/http';

@Component({

  selector: 'vet-doc',

  templateUrl: './vet-doc.component.html',

  styleUrls: ['./vet-doc.component.css']

})

export class VetDocComponent implements OnInit {

  vetdoc: IVetDoc[];

  constructor(private http: HttpClient) { }

  ngOnInit(): void {

    this.http.get<IVetDoc[]>("http://localhost:1337/vet-docs", {observe: "response"}).subscribe(

      (res: HttpResponse<IVetDoc[]>) => {

        console.log(res.body);

        this.vetdoc = res.body}

      );

  }

}

Компонент получения всех больничных листов

import { Component, OnInit } from '@angular/core';

import { HttpResponse, HttpClient} from '@angular/common/http';

import { ISickList } from './sick-list.model';

@Component({

  selector: 'sick-list',

  templateUrl: './sick-list.component.html',

  styleUrls: ['./sick-list.component.css']

})

export class SickListComponent implements OnInit {

  sickLists: ISickList[];

  constructor(private http: HttpClient) { }

  ngOnInit(): void {

    this.http.get<ISickList[]>("http://localhost:1337/sick-lists", {observe: "response"}).subscribe(

      (res: HttpResponse<ISickList[]>) => {

        console.log(res.body);

        this.sickLists = res.body;

        console.log(this.sickLists);

      }

      );

  }

}

Компонент получения одного пациента

import { Component, OnInit } from '@angular/core';

import { HttpResponse, HttpClient } from '@angular/common/http';

import { ActivatedRoute } from '@angular/router';

import { IPatient } from '../patient/patient.model';

@Component({

  selector: 'one-patient',

  templateUrl: './one-patient.component.html',

  styleUrls: ['./one-patient.component.css']

})

export class OnePatientComponent implements OnInit {

  patient: IPatient;

  id: string;

  constructor(protected http: HttpClient, private route: ActivatedRoute) { }

  ngOnInit(): void {

    this.id = this.route.snapshot.paramMap.get('id');

    this.http.get<IPatient>("http://localhost:1337/patients/" + this.id, {observe: "response"}).subscribe(

      (res: HttpResponse<IPatient>) => {

        this.patient = res.body});

  }

  }

Компонент получения одного вет.врача

import { Component, OnInit } from '@angular/core';

import { HttpClient, HttpResponse } from '@angular/common/http';

import { ActivatedRoute } from '@angular/router';

import { IVetDoc } from '../vet-doc/vet-doc.model';

@Component({

  selector: 'one-vet-doc',

  templateUrl: './one-vet-doc.component.html',

  styleUrls: ['./one-vet-doc.component.css']

})

export class OneVetDocComponent implements OnInit {

  vetdoc: IVetDoc;

  id: string;

  constructor(protected http: HttpClient, private route: ActivatedRoute) { }

  ngOnInit(): void {

    this.id = this.route.snapshot.paramMap.get('id');

    this.http.get<IVetDoc>("http://localhost:1337/vet-docs/" + this.id, {observe: "response"}).subscribe(

      (res: HttpResponse<IVetDoc>) => {

        this.vetdoc = res.body});

  }

}

Компонент получения одного больничного листа

import { Component, OnInit } from '@angular/core';

import { HttpClient, HttpResponse } from '@angular/common/http';

import { ActivatedRoute } from '@angular/router';

import { ISickList } from '../sick-list/sick-list.model';

@Component({

  selector: 'one-sick-list',

  templateUrl: './one-sick-list.component.html',

  styleUrls: ['./one-sick-list.component.css']

})

export class OneSickListComponent implements OnInit {

  sickLists: ISickList;

  id: string;

  constructor(protected http: HttpClient, private route: ActivatedRoute) { }

  ngOnInit(): void {

    this.id = this.route.snapshot.paramMap.get('id');

    this.http.get<ISickList>("http://localhost:1337/sick-lists/" + this.id, {observe: "response"}).subscribe(

      (res: HttpResponse<ISickList>) => {

        this.sickLists = res.body});

  }

}

Страница для списка пациентов

<div>

<p>List of patient</p>

<table>

    <tbody>

        <tr>

            <td>id</td>

            <td>name</td>

            <td>kind</td>

            <td>sick list</td>

        </tr>

        <tr \*ngFor="let patient of patients">

            <td><a routerLink="/patients/{{patient.id}}">{{patient.id}}</a></td>

            <td>{{patient.Name\_Patient}}</td>

            <td>{{patient.Kind\_Patient}}</td>

            <td>{{patient.sick\_lists[0].id}}</td>

        </tr>

    </tbody>

</table>

</div>

Страница для списка больничного листа

<div>

    <p>Lists of sick-list</p>

    <table>

        <tbody>

            <tr>

                <td>id</td>

                <td>name</td>

                <td>doc</td>

            </tr>

            <tr \*ngFor="let sicklist of sickLists">

                <td><a routerLink="/sick-lists/{{sicklist.id}}">{{sicklist.id}}</a></td>

                <td>{{sicklist.patient.Name\_Patient}}</td>

                <td>{{sicklist.vet\_doc.Name\_Doc}}</td>

            </tr>

        </tbody>

    </table>

</div>

Страница для списка вет.врачей

<div>

    <p>Lists of vet-docs</p>

    <table>

        <tbody>

            <tr>

                <td>id</td>

                <td>name</td>

                <td>sick list</td>

            </tr>

            <tr \*ngFor="let vetdocs of vetdoc">

                <td><a routerLink="/vet-docs/{{vetdocs.id}}">{{vetdocs.id}}</a></td>

                <td>{{vetdocs.Name\_Doc}}</td>

                <td>{{vetdocs.sick\_lists[0].id}}</td>

            </tr>

        </tbody>

    </table>

</div>

Страница для одного пациента

<div>

    <p>Details of patient</p>

    <table>

        <tbody>

            <tr>

                <td>id</td>

                <td>name</td>

                <td>kind</td>

                <td>sick list</td>

            </tr>

            <tr>

                <td>{{patient.id}}</td>

                <td>{{patient.Name\_Patient}}</td>

                <td>{{patient.Kind\_Patient}}</td>

                <td>{{patient.sick\_lists[0].id}}</td>

            </tr>

        </tbody>

    </table>

    </div>

Страница для одного вет.врача

<div>

    <p>Details of vet doc</p>

    <table>

        <tbody>

            <tr>

                <td>id</td>

                <td>name</td>

            </tr>

            <tr>

                <td>{{vetdoc.id}}</td>

                <td>{{vetdoc.Name\_Doc}}</td>

            </tr>

        </tbody>

    </table>

</div>

Страница для одного больничного листа

<div>

    <p>Details of sick list</p>

    <table>

        <tbody>

            <tr>

                <td>id</td>

                <td>name patient</td>

                <td>name doc</td>

            </tr>

            <tr>

                <td>{{sickLists.id}}</td>

                <td>{{sickLists.patient.Name\_Patient}}</td>

                <td>{{sickLists.vetdoc.Name\_Doc}}</td>

            </tr>

        </tbody>

    </table>

</div>

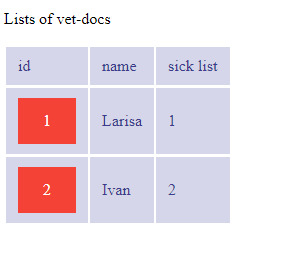


Рисунок 15. Список вет.врачей

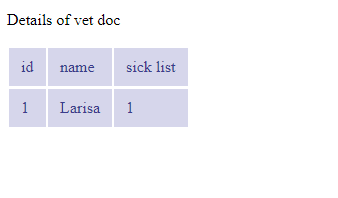


Рисунок 16. Информация о вет.враче

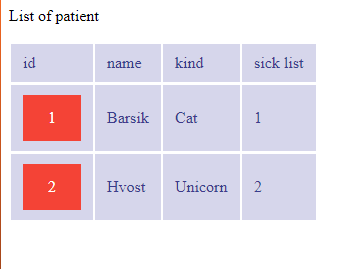


Рисунок 17. Список пациентов

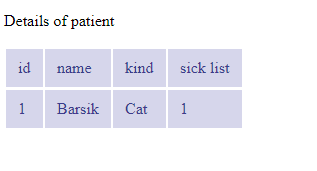


Рисунок 18. Информация о пациенте

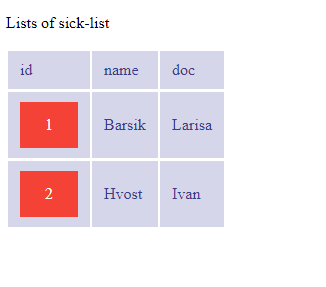


Рисунок 19. Список больничного листа

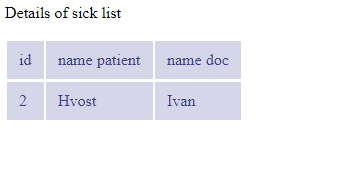


Рисунок 20. Информация больничного листа

**ВЫВОД**

В ходе лабораторной работы были приобретены навыки работы с Headless CMS.