МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №6

по дисциплине: Технологии Web-программирования

тема: «Работа с HTTP запросами»

Выполнил: ст. гр. ПВ-41

Мишина Ю.А.

Проверил: Картамышев С. В.

Белгород 2020 г.

**Цель работы:** изучить принципы работы отправки запросов. Изучить работу отправки запросов в React.js.

**Выполнение**

В качестве разрабатываемого проекта была выбрана тематика новостного сайта об играх. Сайт представляет из себя главную страницу со списком новостей в виде карточек, возможность перейти на новость с ее детальным просмотром и посмотреть информацию о сайте.

В приложении предполагается следующее: имеется заполненная база данных новостей. Пользователь переходить на страницы новостей, в бд хранится статический набор новостей.

**Back**

Для реализации http запросов были разработаны контроллеры

InfoController

@RestController  
@RequestMapping(ApiUtil.*INFO*)  
@RequiredArgsConstructor  
@CrossOrigin  
public class InfoController {  
  
 private final InfoService infoService;  
  
 @GetMapping("/getInfo")  
 public List<InfoDto> getInfo() {  
 //return infoService.getInfo();  
 return Collections.*singletonList*(infoService.getInfo());  
 }  
}

NewsController

@RestController  
@RequestMapping(ApiUtil.*NEWS*)  
@CrossOrigin  
@RequiredArgsConstructor  
@Transactional  
public class NewsController {  
  
 private final NewsService newsService;  
  
 @GetMapping("/getAllNews")  
 public List<NewsDto> getAllNews() {  
 return newsService.getAllNews();  
 }  
  
 @GetMapping("/getNews/{id}")  
 public NewsDto getNewsById(@PathVariable("id") Long id) {  
 return newsService.getNewsById(id);  
 }  
}

ForDBController

@RestController  
@RequestMapping(ApiUtil.*DB*)  
@RequiredArgsConstructor  
@CrossOrigin  
public class ForDBController {  
  
 private final ImageRepository imageRepository;  
  
 private final NewsRepository newsRepository;  
  
 private final InfoRepository infoRepository;  
  
  
 @GetMapping("/clean")  
 @Transactional  
 public void clearDb() {  
 newsRepository.deleteAll();  
 infoRepository.deleteAll();  
 imageRepository.deleteAll();  
 }  
  
 @GetMapping("/fill")  
 @Transactional  
 public void fillDB() {  
 addAllInfoToDB();  
 addAllNewsInDB();  
 }  
  
 void addAllNewsInDB() {  
 newsRepository.deleteAll();  
 List<News> allNews = new ArrayList<>();  
  
 News news1 = News.*builder*()  
 .title("ГЕЙМПЛЕЙ WATCH DOGS: LEGION НА XBOX SERIES X С ТРАССИРОВКОЙ ЛУЧЕЙ")  
 .lessDescription("Watch Dogs: Legion уже доступна на PC и приставках текущего поколения, однако приверженцам консольного гейминга, собирающимся перейти на некстген, есть смысл дождаться соответствующей версии игры. Ведь PlayStation 5 и Xbox Series похвастают более технологичной картинкой.\n" +  
 "Красоты некстгена Ubisoft демонстрирует в геймплее Legion на Xbox Series X, где игра использует трассировку лучей, работая в разрешении 4K при 30 fps. Ещё упоминаются быстрые загрузки, а также повышенная проработка текстур, кубических карт, теней, технологий рендеринга и детализации объектов.\n" +  
 "На Xbox Series игра выйдет в день релиза консоли — 10 ноября. Legion включена и в стартовую линейку PS5, а значит, появится там 12 ноября. Бесплатный апгрейд предусмотрен на обоих семействах приставок.\n"  
 )  
 .description("Watch Dogs: Legion уже доступна на PC и приставках текущего поколения, однако приверженцам консольного гейминга, собирающимся перейти на некстген, есть смысл дождаться соответствующей версии игры. Ведь PlayStation 5 и Xbox Series похвастают более технологичной картинкой.")  
 .image(addImage("src/main/resources/news/news1.jpg", "news1", "jpg"))  
 .urlYouTubeVideo("https://www.youtube.com/embed/imhDZGe\_bwI")  
 .game(  
 Game.*builder*()  
 .nameGame("watch Dogs: Legion")  
 .platform("PC, PS4, PS5, Stadia, XBOXS, XXONE")  
 .genre("action")  
 .cover(addImage("src/main/resources/games/game1.jpg", "game1", "jpg"))  
 .dateReleased("29 октября 2020 г.")  
 .build())  
 .build();  
  
 News news2 = News.*builder*()  
 .title("ВЫШЛА БЕТА-ВЕРСИЯ BLACK MESA DEFINITIVE EDITION, УЛУЧШАЮЩАЯ ГЕЙМПЛЕЙ И ГРАФИКУ")  
 .lessDescription("Black Mesa вышла из раннего доступа в марте, однако это не значит, что работа над игрой закончена. Студия Crowbar Collective выпустила открытую бета-версию Definitive Edition — обновления 1.5, которое улучшает геймплей и графику, а также вносит множество точечных исправлений.\n" +  
 "Нововведения касаются в том числе следующих глав:\n" +  
 "Power Up. Главу изменили так, чтобы сделать геймплей более «причинно-следственным», а общую цель — более ясной. Теперь вам нужно включить свет, убить Гаргантюа и активировать вагонетку. Во время прохождения можно оценить новую систему динамического освещения.\n" +  
 "On A Rail. Разработчики внесли крупномасштабные изменения в ракетную шахту, дабы лучше показать задачу и её последующее решение, благодаря чему глава стала более цельной.\n" +  
 "Кроме того, авторы продолжают подтягивать внешний вид локаций до уровня мира Зен.\n"  
 )  
 .description("Black Mesa вышла из раннего доступа в марте, однако это не значит, что работа над игрой закончена. Студия Crowbar Collective выпустила открытую бета-версию Definitive Edition — обновления 1.5, которое улучшает геймплей и графику, а также вносит множество точечных исправлений.\n")  
 .image(addImage("src/main/resources/news/news2.jpeg", "news2", "jpeg"))  
 .urlYouTubeVideo("https://www.youtube.com/embed/VfUU8Y0CG7E")  
 .game(  
 Game.*builder*()  
 .nameGame("Black Mesa")  
 .platform("PC")  
 .genre("action")  
 .cover(addImage("src/main/resources/games/game2.jpg", "game2", "jpg"))  
 .dateReleased("6 марта 2020 г.")  
 .build())  
 .build();  
  
 News news3 = News.*builder*()  
 .title("ЭКШЕН PRAEY FOR THE GODS, ПОХОЖИЙ НА SHADOW OF THE COLOSSUS, ВЫЙДЕТ НА XB1, PS4 И PS5 В НАЧАЛЕ 2021-ГО")  
 .lessDescription("Приключенческий боевик Praey for the Gods томится в раннем доступе на PC с начала 2019-го. Коллектив No Matter Studios, состоящий всего из трёх сотрудников, медленно, но верно ведёт свой проект к финишной черте — в первом квартале 2021-го игра дорастёт до версии 1.0 и выйдет на обоих поколениях PlayStation, а также на Xbox One.\n" +  
 "Разумеется, на PS5 экшен будет выгодно отличаться от PS4-версии. Особенности у издания для некстгена следующие:\n" +  
 "60 fps (против 30 на PS4).\n" +  
 "Максимальные графические настройки — полные тени, текстуры высокого разрешения и так далее.\n" +  
 "Значительно ускоренные загрузки.\n" +  
 "Поддержка адаптивных триггеров и тактильной отдачи DualSense.\n" +  
 "Бесплатный апгрейд с PS4.\n"  
 )  
 .description("Приключенческий боевик Praey for the Gods томится в раннем доступе на PC с начала 2019-го. Коллектив No Matter Studios, состоящий всего из трёх сотрудников, медленно, но верно ведёт свой проект к финишной черте — в первом квартале 2021-го игра дорастёт до версии 1.0 и выйдет на обоих поколениях PlayStation, а также на Xbox One.")  
 .image(addImage("src/main/resources/news/news3.jpeg", "news3", "jpeg"))  
 .urlYouTubeVideo("https://www.youtube.com/embed/xQMn4JEZmiE")  
 .game(  
 Game.*builder*()  
 .nameGame("Praey for the Gods")  
 .platform("PC, PS4, PS5, XONE")  
 .genre("action")  
 .cover(addImage("src/main/resources/games/game3.jpg", "game3", "jpg"))  
 .dateReleased("2021 г.")  
 .build())  
 .build();  
  
 News news4 = News.*builder*()  
 .title("ТОДД ГОВАРД: ТРУДНО ПРЕДСТАВИТЬ, ЧТО MICROSOFT ПОМЕШАЕТ THE ELDER SCROLLS VI ВЫЙТИ НА PLAYSTATION")  
 .lessDescription("Компания ZeniMax (и, соответственно, Bethesda) становится частью семейства Microsoft. В воздухе висят животрепещущие вопросы: например, значит ли это, что все последующие игры Bethesda, включая Starfield и The Elder Scrolls VI, окажутся эксклюзивом платформ Microsoft? Тодд Говард (Todd Howard) коснулся этой темы в интервью GamesIndustry.biz.\n" +  
 "В целом Microsoft лояльно относится к другим платформам: скажем, Minecraft выходит вообще везде, где только можно, а сделки о временной эксклюзивности Deathloop и GhostWire: Tokyo для PlayStation 5 остаются в силе. «Трудно представить, что Microsoft, несмотря на всё своё влияние над Bethesda, ограничит The Elder Scrolls VI своими платформами», — пишет интервьюер GI.biz.\n" +  
 "«Соглашусь, представить такое трудно», — подтверждает Тодд Говард. Впрочем, не рассматривайте это как гарантию того, что TESVI появится на PlayStation: формально слияние Microsoft и Bethesda пока не завершено, а некоторые детали сделки ещё обсуждаются. Говард признаётся, что и сам пока не знает, на каких условиях выйдут Starfield и TESVI: в данный момент лишь подтверждено, что их добавят в Xbox Game Pass прямо во время релиза.\n" +  
 "В то же время Говард вспоминает, что в истории The Elder Scrolls полно примеров тесного сотрудничества с Microsoft: Morrowind выходила на первой Xbox, Oblivion долгое время оставалась консольным эксклюзивом Xbox 360 (то есть появлялась на PC, но не на PlayStation 3), а DLC к Skyrim стартовали в первую очередь у «мелкомягких».\n"  
 )  
 .description("Компания ZeniMax (и, соответственно, Bethesda) становится частью семейства Microsoft. В воздухе висят животрепещущие вопросы: например, значит ли это, что все последующие игры Bethesda, включая Starfield и The Elder Scrolls VI, окажутся эксклюзивом платформ Microsoft? Тодд Говард (Todd Howard) коснулся этой темы в интервью GamesIndustry.biz.")  
 .image(addImage("src/main/resources/news/news4.jpeg", "news4", "jpeg"))  
 .urlYouTubeVideo("https://www.youtube.com/embed/OkFdqqyI8y4")  
 .game(  
 Game.*builder*()  
 .nameGame("Elder Scrolls 6, The")  
 .platform("PC, PS5, XBOXSX")  
 .genre("rpg")  
 .cover(addImage("src/main/resources/games/game4.jpg", "game4", "jpg"))  
 .dateReleased("2022 г.")  
 .build())  
 .build();  
  
 allNews.add(news1);  
 allNews.add(news2);  
 allNews.add(news3);  
 allNews.add(news4);  
 newsRepository.saveAll(allNews);  
 }  
  
 void addAllInfoToDB() {  
 infoRepository.deleteAll();  
  
 Info info = Info.*builder*()  
 .number("8 (000) 000-00-00")  
 .number2("8 (000) 000-00-00")  
 .logo(addImage("src/main/resources/info/snail.png","logo" ,"png"))  
 .headerImg(addImage("src/main/resources/info/back.jpg","headerImg" ,"jpg"))  
 .logoVk(addImage("src/main/resources/info/vk.png", "logoVk", "png"))  
 .logoInstagram(addImage("src/main/resources/info/inst.png", "logoInstagram", "png"))  
 .logoTwitter(addImage("src/main/resources/info/twit.png", "logoTwitter", "png"))  
 .logoFacebook(addImage("src/main/resources/info/faceb1.png", "logoFacebook", "png"))  
 .logoTelegram(addImage("src/main/resources/info/teleg.png", "logoTelegram", "png"))  
 .urlVk("https://vk.com/gamelnews")  
 .urlInstagram("https://www.instagram.com/blog\_games/")  
 .urlTwitter("https://twitter.com/games\_news\_")  
 .urlFacebook("https://ru-ru.facebook.com/FacebookGaming")  
 .urlTelegram("https://tlgrm.ru/channels/gaming")  
 .description("SNAIL — молодой сайт относительно своих конкурентов с запоминающимся названием будет информировать всех желающих о новинках в гейм-мире. Также за новостями можно наблюдать и в популярных соцсетях. Прямо сейчас вы зашли на нашу хаб-локацию: SNAIL.Ru — платформа для нашего и вашего контента! Со своей стороны мы можем предложить:\n" +  
 "Читать здесь статьи.\n" +  
 "Общаться в комментариях (в том числе с нами).\n" +  
 "Создавать блоги и участвовать в еженедельном розыгрыше.\n" +  
 "Подписываться на игры и разделы блогов, чтобы собрать из сайта собственную ленту предпочтительного контента.\n" +  
 "Мы тут недавно сделали редизайн, поэтому сайт в активной разработке. Если у вас есть идеи, предложения или вы нашли ужасающий баг — шлите это нам!")  
 .build();  
  
 infoRepository.save(info);  
 }  
  
  
 private Image addImage(String path, String nameImg, String typeImg) {  
 File file = new File(path);  
 byte[] bFile = new byte[(int) file.length()];  
  
 try {  
 FileInputStream fileInputStream = new FileInputStream(file);  
 fileInputStream.read(bFile);  
 fileInputStream.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 Image image = Image.*builder*()  
 .name(nameImg) // имя картинки  
 .type(typeImg) // тип  
 .imageData(bFile)  
 .build();  
  
 return imageRepository.save(image);  
 }  
}

**Front**

Для получения результата запроса был реализован сервис

Service.js

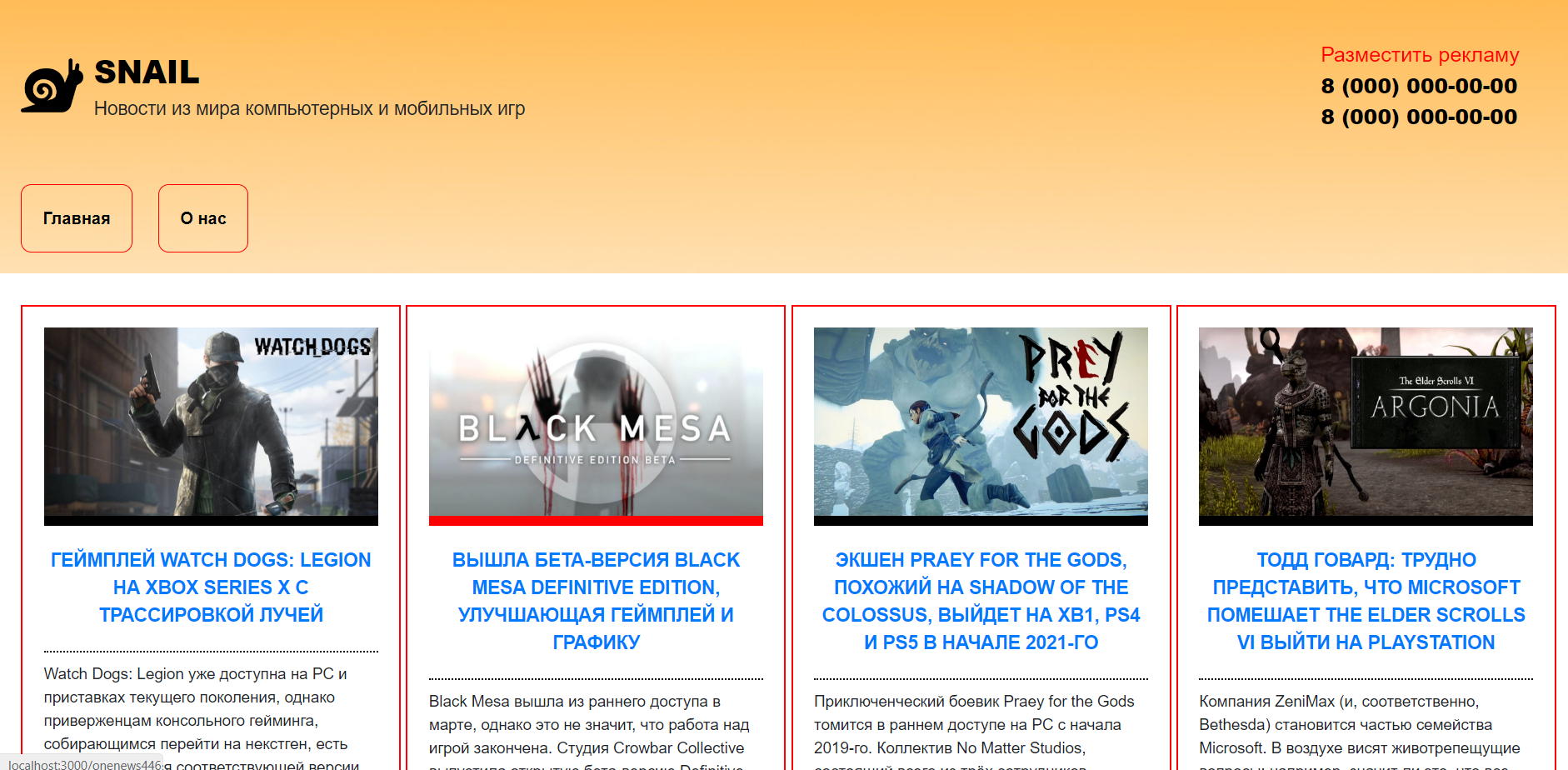
export default class RestApi {  
 constructor() {  
 this.\_apiBase = "http://localhost:8081/api/v1/";  
 }  
  
 getAllNews = async (url) => {  
 const result = await fetch(`${this.\_apiBase}${url}`);  
 if (!result.ok) {  
 throw new ***Error***(`Could not fetch ${url}, status : ${result.status}`);  
 }  
 return await result.json();  
 };  
  
 getNews = async (id) => {  
 const result = this.getAllNews(`news/getAllNews/${id}`);  
 return result;  
 };  
  
 getInfo = async (url) => {  
 const result = await fetch(`${this.\_apiBase}${url}`);  
 if (!result.ok) {  
 throw new ***Error***(`Could not fetch ${url}, status : ${result.status}`);  
 }  
 return await result.json();  
 };  
}

пример компонента news

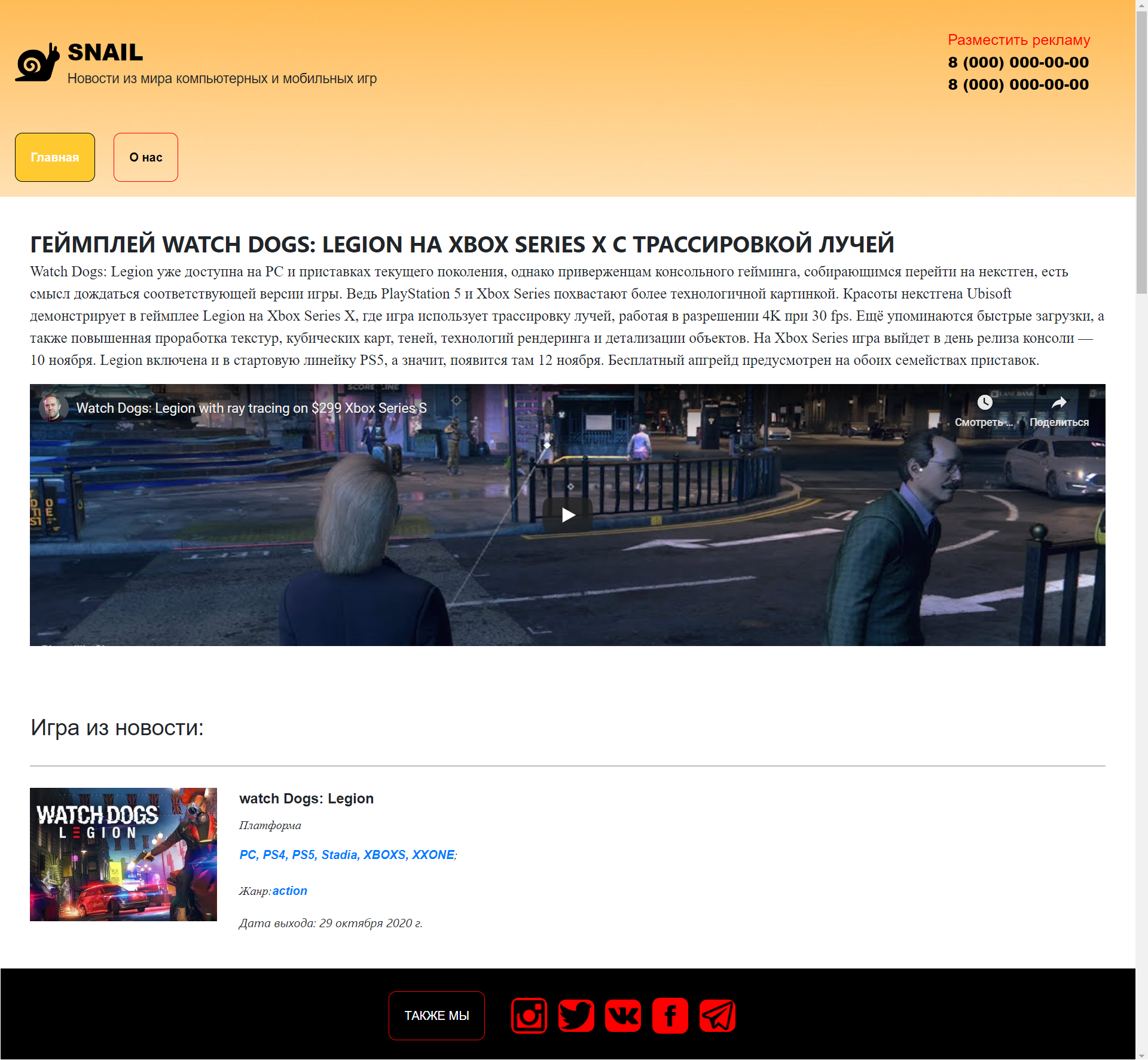
import ***React***, { useState, useEffect } from "react";  
import RestApi from "../service";  
  
const apiReq = new RestApi();  
  
const News\_main = ({ id }) => {  
 const [state, setState] = useState({ newArr: [] });  
  
 useEffect(() => {  
 apiReq.getAllNews("news/getAllNews").then((item) => {  
 return setState(() => {  
 return searchElem(***Number***(id), item);  
 });  
 });  
 }, []);  
  
 const searchElem = (id, item) => {  
 const index = item.findIndex((elem) => elem.id === id);  
 const newArr = item.slice(index, index + 1);  
 return { newArr };  
 };  
  
 ***console***.log(state.newArr);  
 return state.newArr.map((item) => {  
 const { id, title, lessDescription, urlYouTubeVideo, game } = item;  
 return (  
 <div key={id}>  
 <div className="block\_text">  
 <b>{title}</b>  
 <p>{lessDescription}</p>  
 <iframe  
 width="100%"  
 height="350px"  
 src={urlYouTubeVideo}  
 frameborder="0"  
 allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"  
 allowfullscreen  
 ></iframe>  
 </div>  
 <div className="news\_block">  
 <div className="wrap\_top">Игра за новости:</div>  
 <div className="wrap\_bottom">  
 <div className="bottom\_left">  
 <img  
 src={`data:image/jpeg;base64,${game.cover.imageBase64}`}  
 alt={game.cover.name}  
 />  
 </div>  
 <div className="bottom\_right">  
 <div className="right\_title">{game.nameGame}</div>  
 <div className="right\_description">  
 <p>Платформа</p>  
 <a href="">{game.platform}</a>;  
 <br />  
 <br />  
 Жанр:  
 <a href="">{game.genre}</a>  
 </div>  
 <div className="right\_data">Дата выхода: {game.dateReleased}</div>  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 });  
};  
  
export default News\_main;

Общий вид получившегося сайта

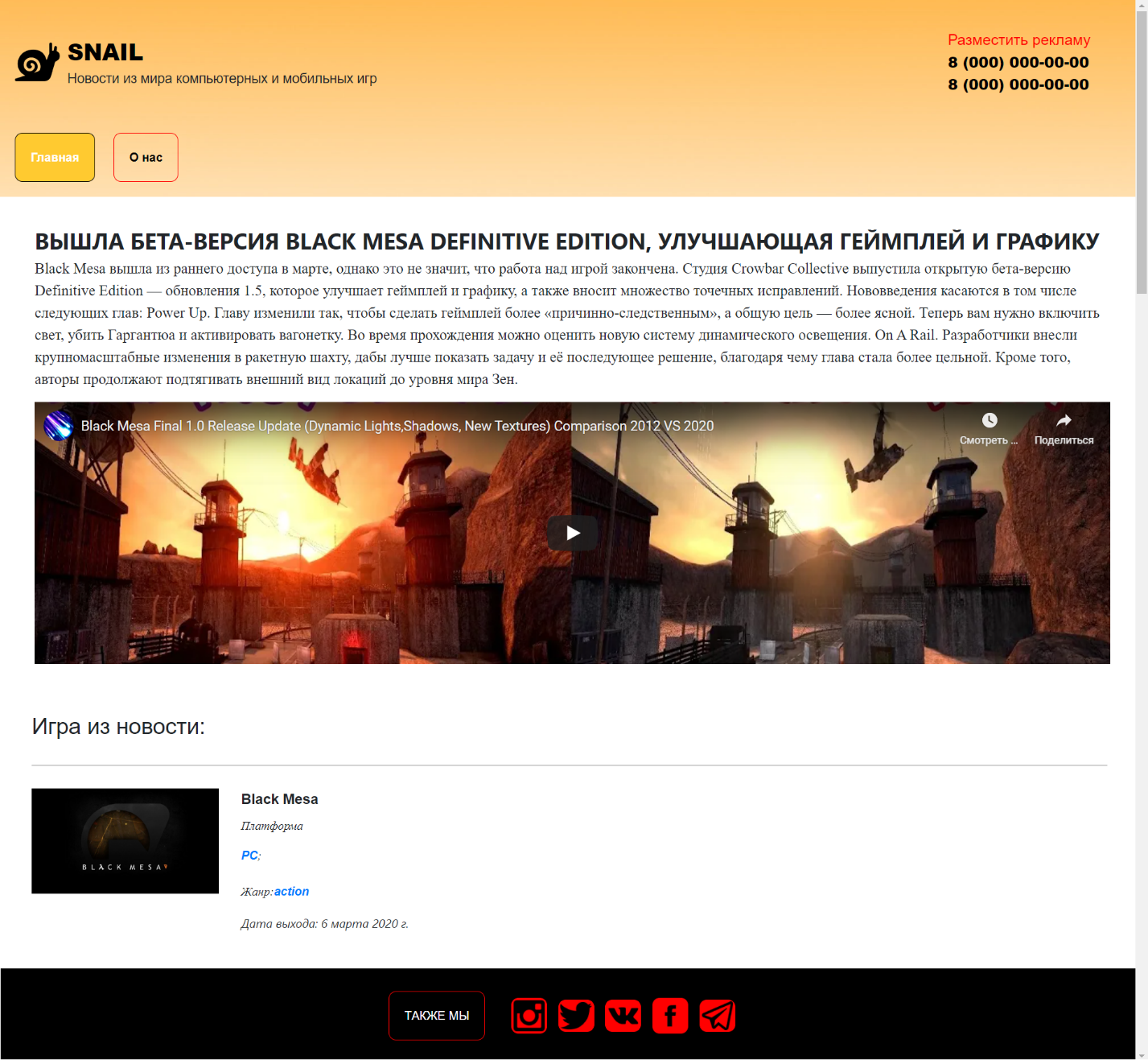
Главная страница



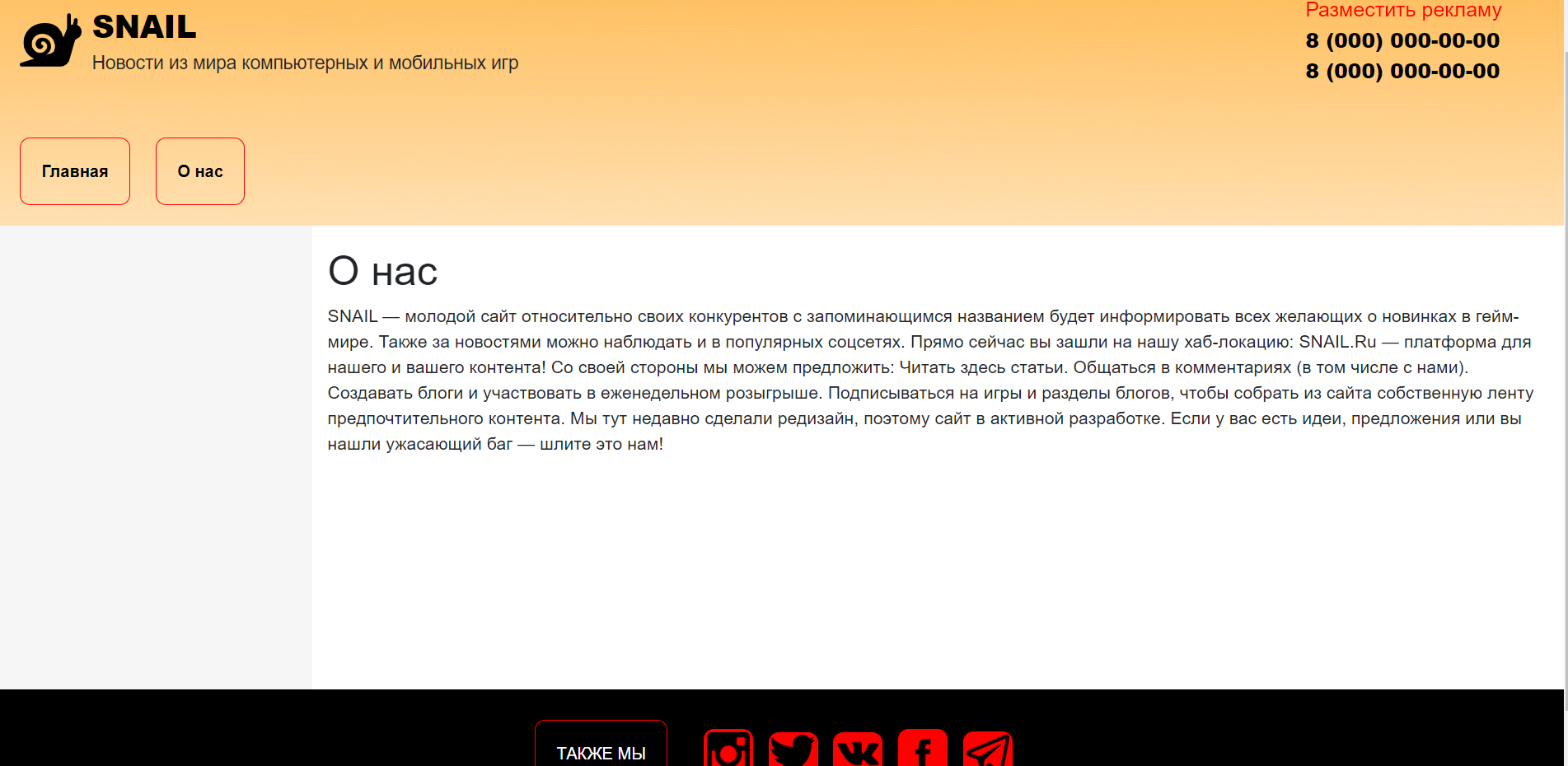
Страница детального просмотра новости 1



Страница детального просмотра новости 2



Страница информации о сайте



**Вывод**: в результате выполнения работы мы изучили принципы работы отправки запросов, изучили принцип работы отправки запросов в react.js.