1. Проектно-конструкторская часть
   1. Разработка структуры приложения.

В состав компонентов системы входят:

– Источник данных;

– Клиентское приложение для отображения данных;

– Серверная компонента для обработки исходных данных.

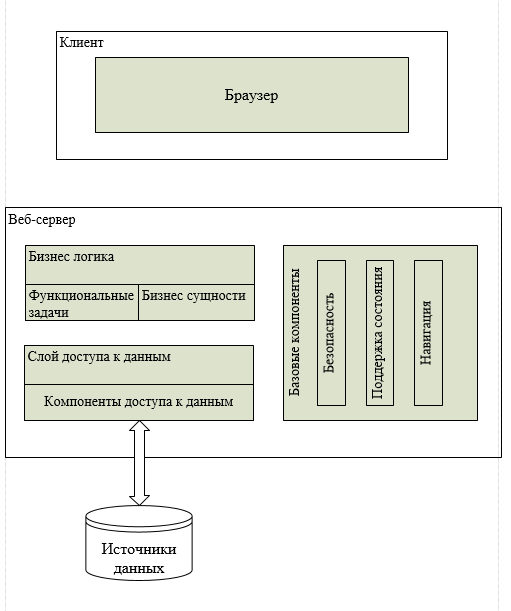


Рисунок – структурная схема приложения

* 1. Логическая структура источника данных

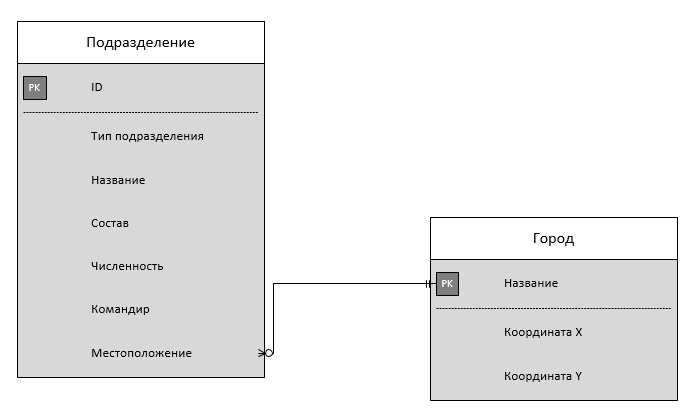


Рисунок – логическая структура источника данных

* 1. Разработка интерфейса взаимодействия пользователя с системой.

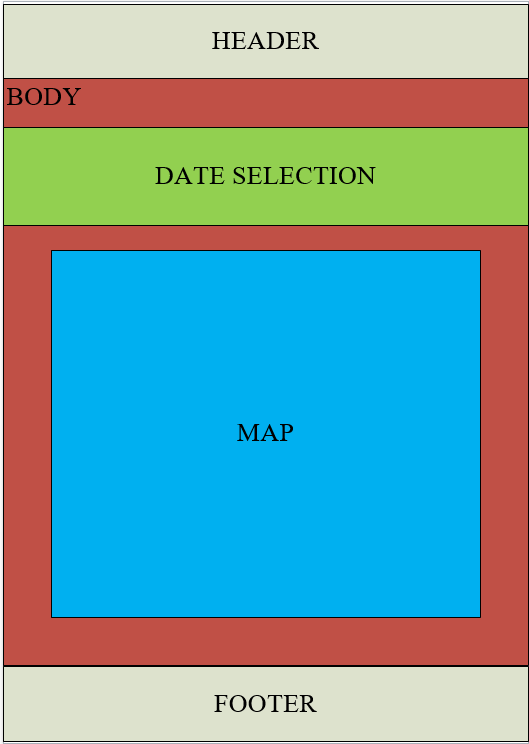
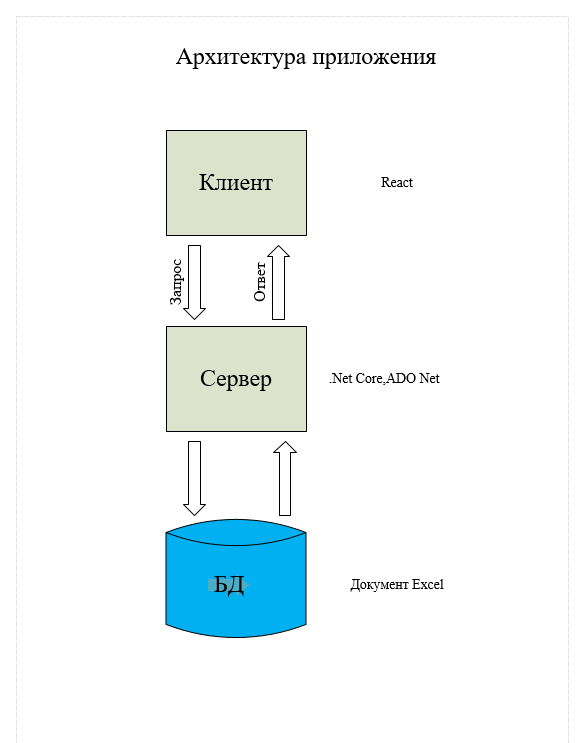


Рисунок – прототип интерфейса

* 1. Разработка архитектуры приложения.



.NET — это бесплатная платформа разработки с открытым исходным кодом для создания различных типов приложений, таких как следующие:

* [Веб-приложения, веб-API и микрослужбы](https://docs.microsoft.com/ru-ru/aspnet/core/introduction-to-aspnet-core#recommended-learning-path)
* [Бессерверные функции в облаке](https://docs.microsoft.com/ru-ru/azure/azure-functions/functions-create-first-function-vs-code?pivots=programming-language-csharp)
* [Полностью облачные приложения](https://docs.microsoft.com/ru-ru/dotnet/architecture/cloud-native/)
* [Мобильные приложения.](https://dotnet.microsoft.com/learn/xamarin/hello-world-tutorial/intro)
* Классические приложения
* [Windows WPF](https://docs.microsoft.com/ru-ru/dotnet/desktop/wpf/)
* [Windows Forms](https://docs.microsoft.com/ru-ru/dotnet/desktop/winforms/)
* [Универсальная платформа Windows (UWP)](https://docs.microsoft.com/ru-ru/windows/uwp/get-started/create-a-hello-world-app-xaml-universal)
* [Игры](https://dotnet.microsoft.com/apps/games)
* [Интернет вещей.](https://docs.microsoft.com/ru-ru/dotnet/iot)
* [Машинное обучение](https://docs.microsoft.com/ru-ru/dotnet/machine-learning/)
* [Консольные приложения](https://docs.microsoft.com/ru-ru/dotnet/core/tutorials/with-visual-studio-code)
* [Службы Windows](https://docs.microsoft.com/ru-ru/aspnet/core/host-and-deploy/windows-service)

Для совместного использования функциональных возможностей различных приложений и типов приложений используются [библиотеки классов](https://docs.microsoft.com/ru-ru/dotnet/standard/class-libraries).

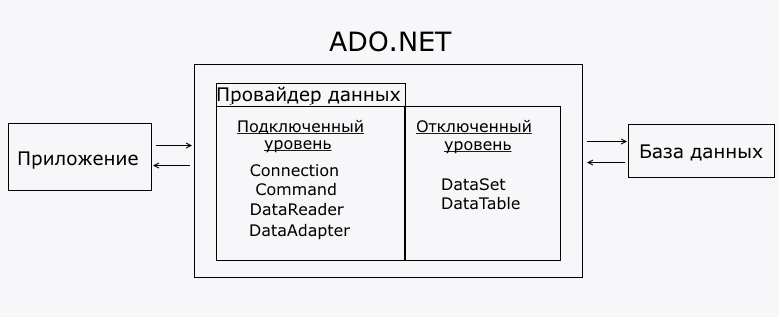


Рисунок – представление технологии передачи данных между компонентами

ADO.NET предоставляет согласованный доступ к таким источникам данных, как SQL Server и XML, а также к источникам данных, предоставляемым при помощи OLE DB и ODBC. Пользовательские приложения, использующие общие данные, могут использовать ADO.NET для соединения с этими источниками данных и для получения, обработки и обновления имеющихся в них данных.

OLE DB ([англ.](https://dic.academic.ru/dic.nsf/ruwiki/6161) Object Linking and Embedding, Database) — набор [интерфейсов](https://dic.academic.ru/dic.nsf/ruwiki/6202), основанных на [COM](https://dic.academic.ru/dic.nsf/ruwiki/35034), которые позволяют [приложениям](https://dic.academic.ru/dic.nsf/ruwiki/4631) обращаться к [данным](https://dic.academic.ru/dic.nsf/ruwiki/71919), хранимым в разных источниках информации или хранилищах данных с помощью унифицированного доступа.

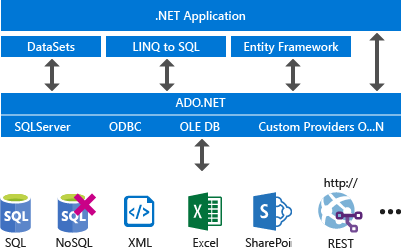


Рисунок – представление технологии передачи данных между компонентами

* 1. Реализация функционирующего приложения.
     1. Серверный компонент

Для получения информации из таблиц Excel используется класс контекста данных. Считанные строки хранятся в объекте List\_of\_strings.

Для представления городов по паре ключ-значение используется словарь (объект Coordinates\_of\_city).

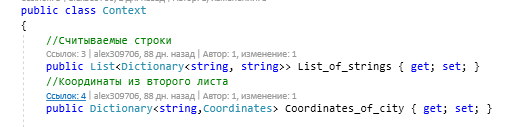


Рисунок – листинг свойств класса Context

Для получения данных требуется установить связь с объектом источника данных. Строка подключения возвращается методом GetConnectionString.



Рисунок – генерация строки подключения

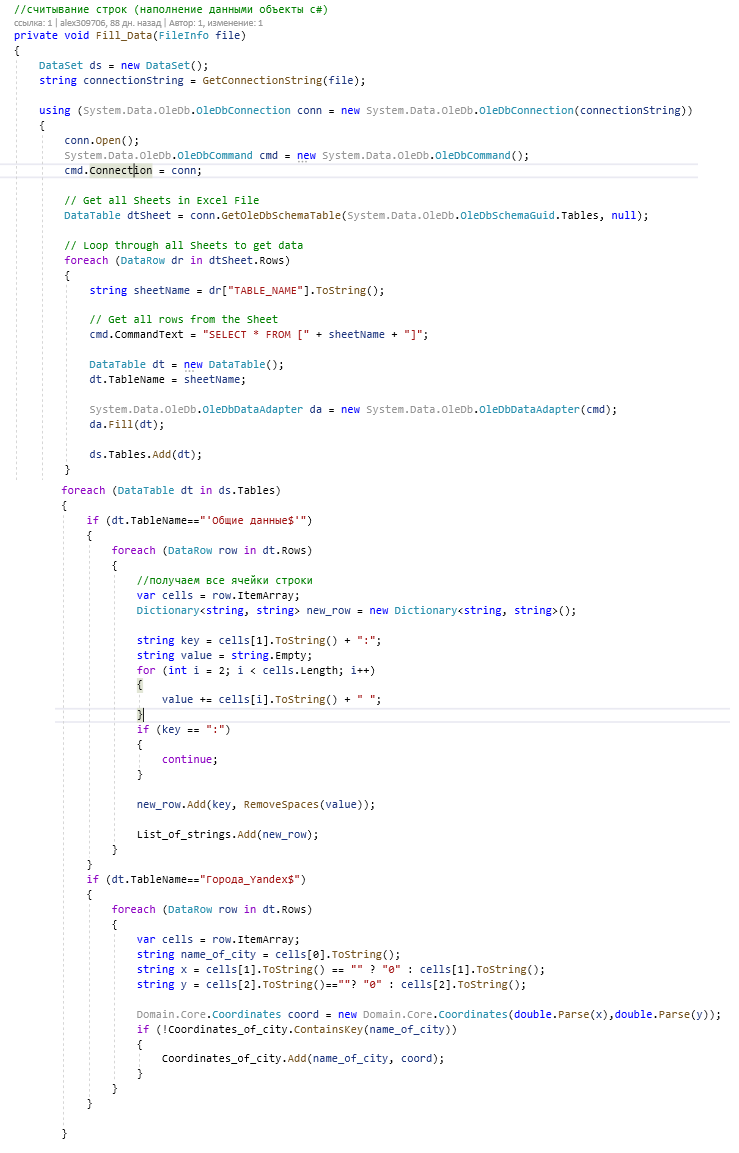


Рисунок – наполнение данными объекты C#



Рисунок – обработка данных и создание объектов предметной области на языке C#



Рисунок – использование объектов в серверной компоненте проекта

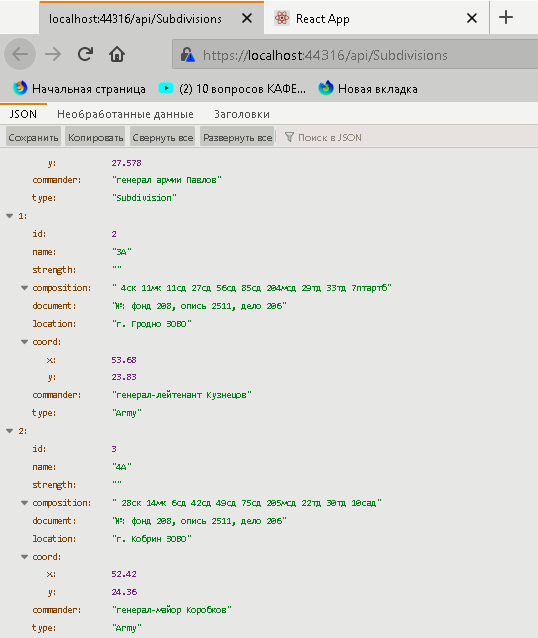


Рисунок – возвращаемые данные с сервера

* + 1. Клиентское приложение

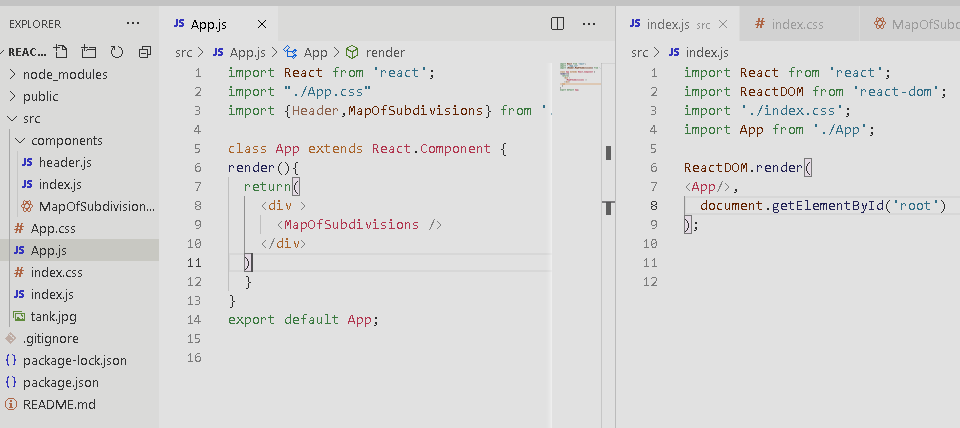


Рисунок – корневые элементы клиентского приложения

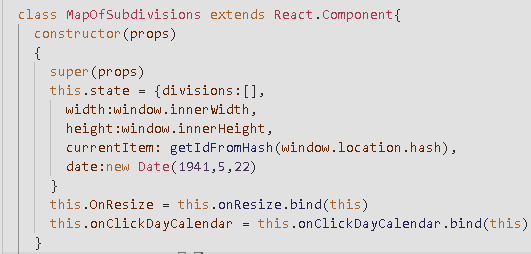


Рисунок – конструктор карты (React – компонента)

На стадии жизненного цикла «монтирование компонента» необходимо загрузить данные с сервера в и определить состояние приложения.



Рисунок – посылка запроса и установка состояния приложения

В методе render формируется код отрисовки элементов на странице веб-браузера.

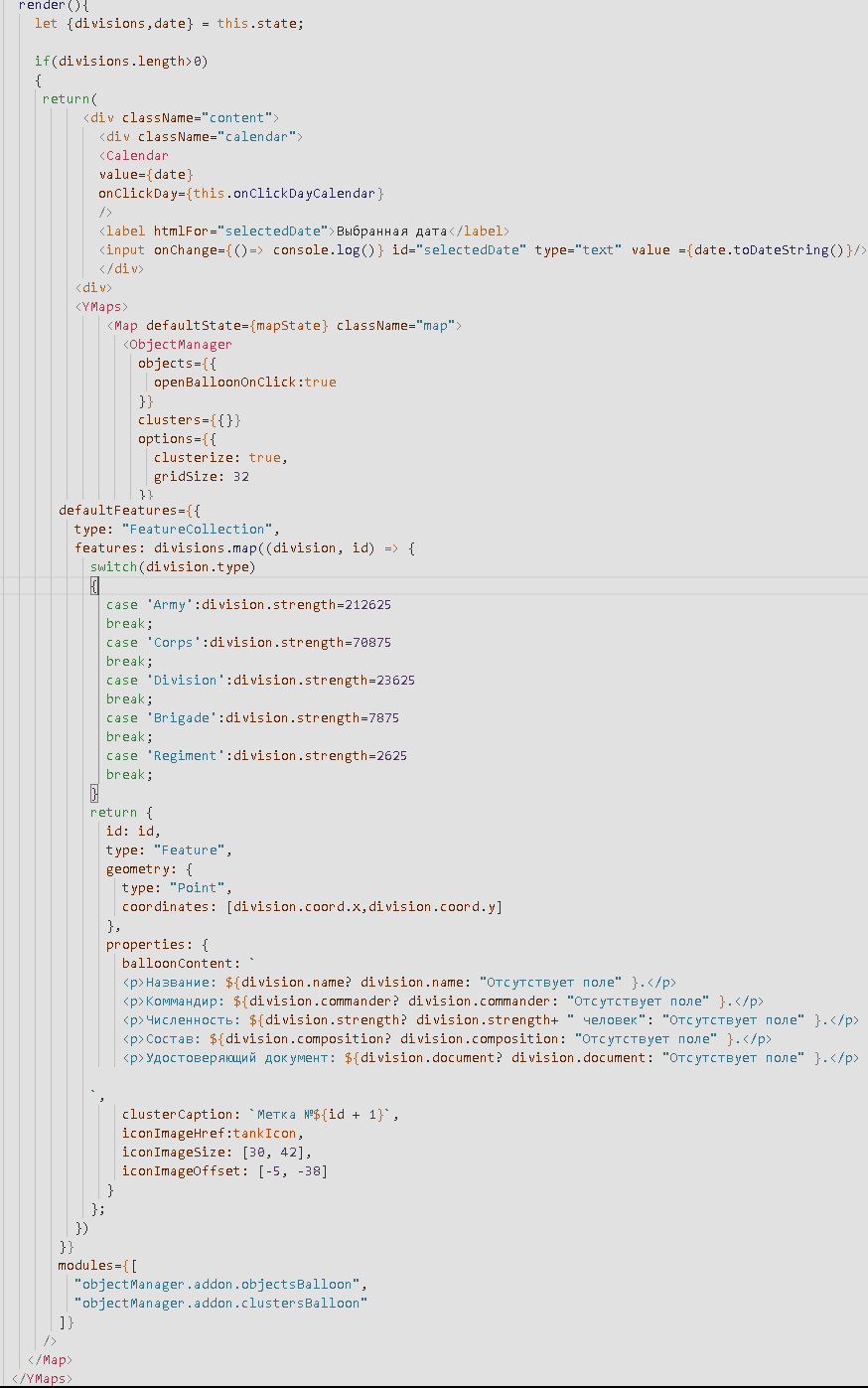


Рисунок – формирование отрисовки элеметнов

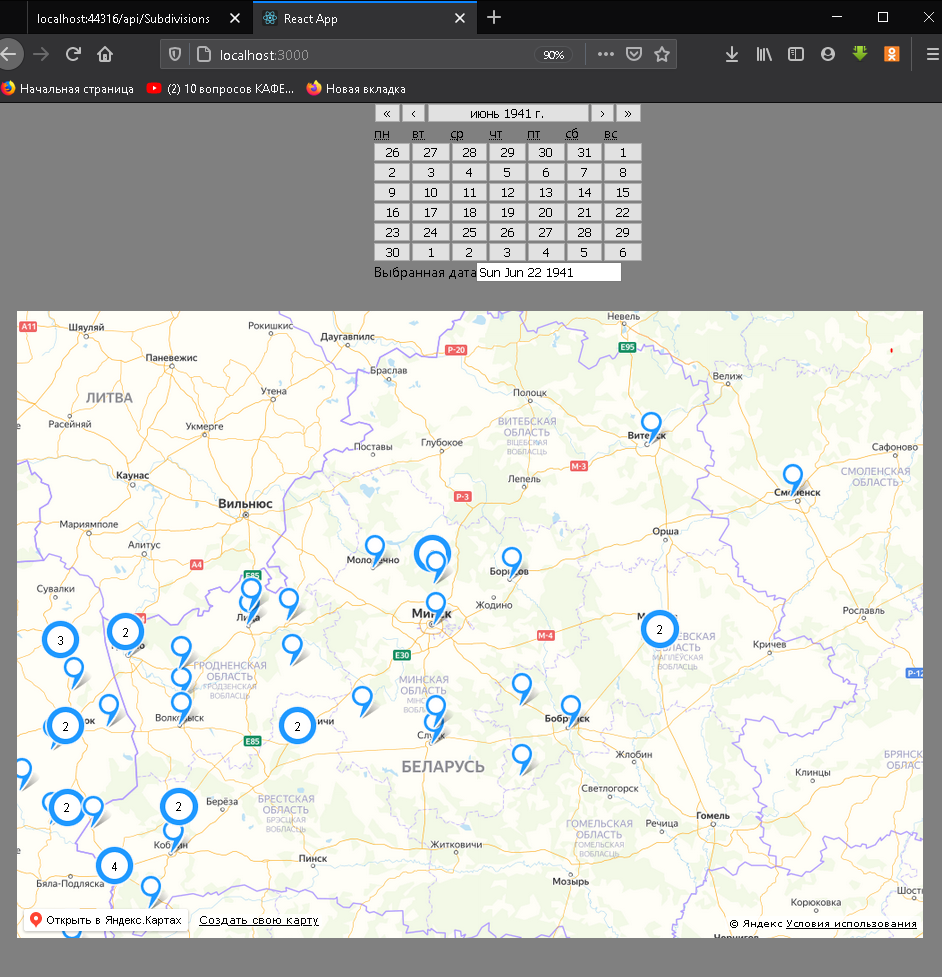
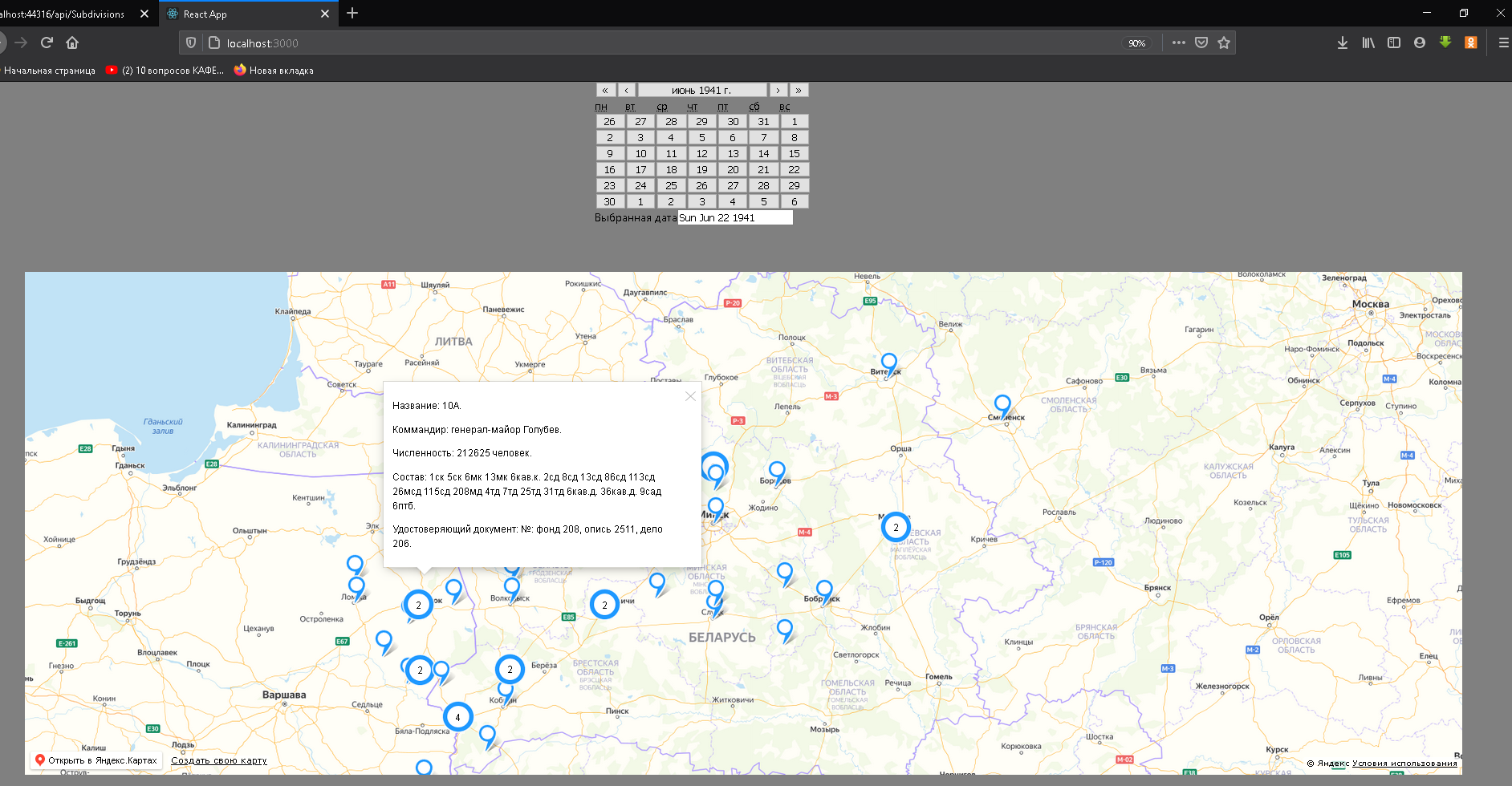


Рисунок – интерфейс приложения

 Рисунок – интерфейс приложения