Кратко о приложении

Форма обратной связи дает пользователю возможность вводить некоторое сообщение, выбирать тему сообщения, а также вводить свои личный данные, такие как имя, email и номер телефона, также присутствует recaptcha.

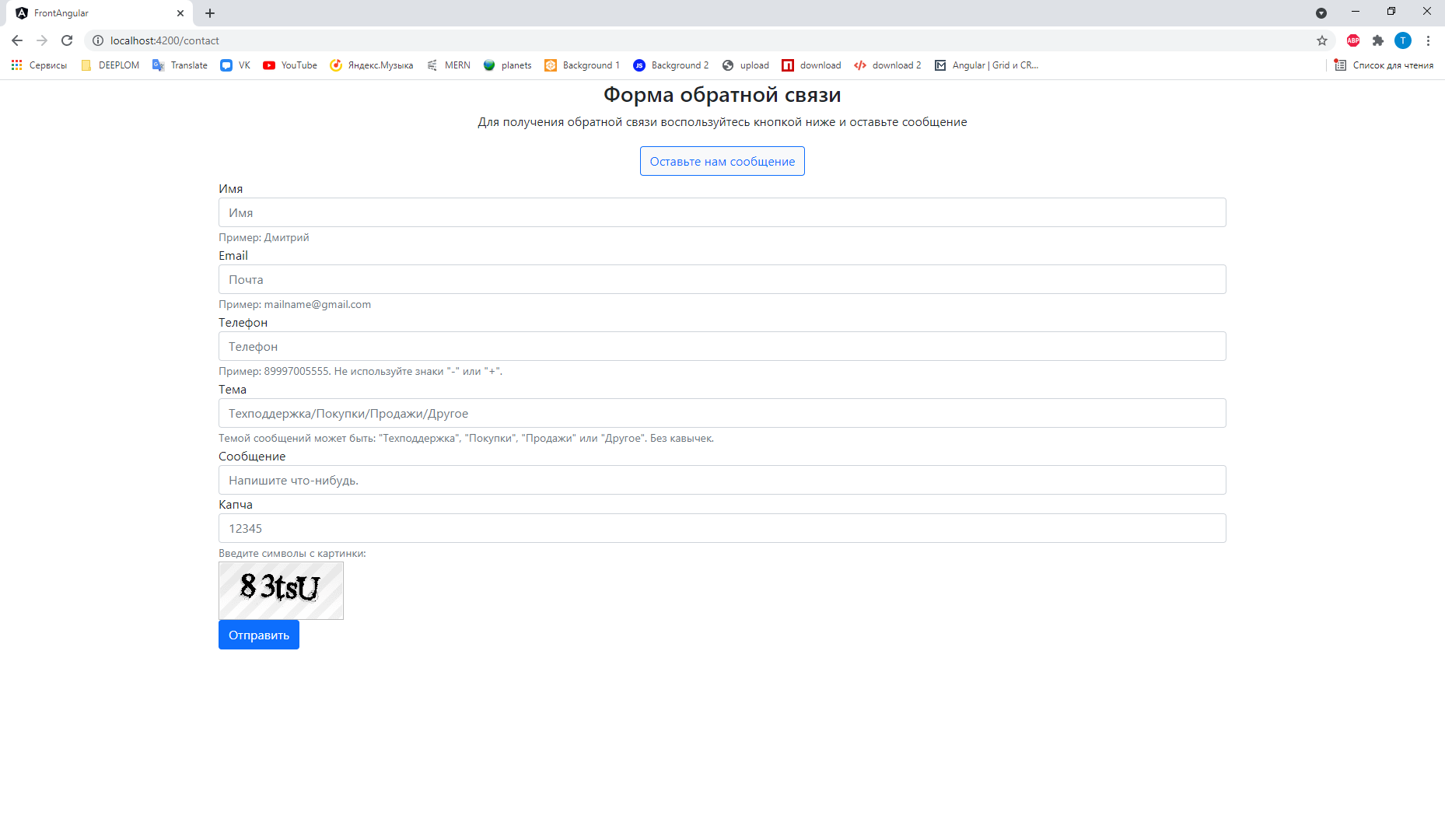


Рис. 1 Внешний вид приложения

Что включает в себя приложение?

Приложение состоит из двух частей: ui и api. Api, он же бэкэнд, создан для того, чтобы вносить информацию в БД MS SQL при помощи POST запросов. В БД есть три таблицы: таблица Контактов, таблица Текстов сообщений и таблица Тем сообщений. В таблице текстов сообщений есть два вторичных ключа, один ссылается на таблицу контактов, второй ссылается на таблицу Тем.

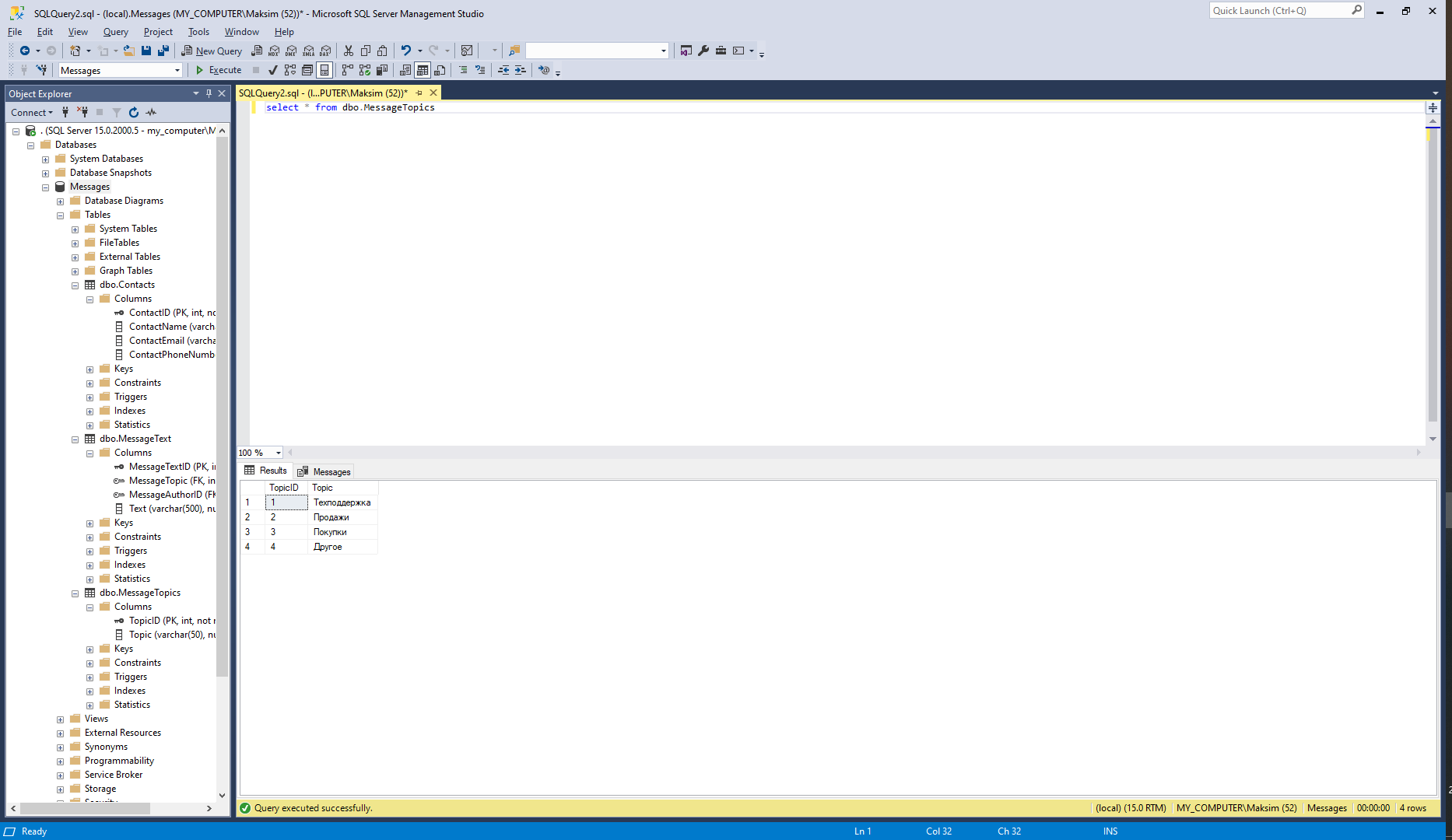


Рис. 2 MS SQL, select запрос и структура таблиц(слева)

Формирование запроса в БД зависит от данных, которые ввел пользователь. Если введенный пользователем Email или телефон уже существуют в БД, то новая запись в таблице контактов не создается, в данном случае создается только запись в таблицу Текстов сообщений, которая будет ссылаться на уже существующую запись в таблице Контактов. В случае, если Email или телефон не были найдены среди уже существующих, производится запись и в таблицу контактов и в Таблицу пользователей.

Ui, он же фронтэнд, это приложение на Angular, которое реализует пользовательский интерфейс. Ui не связывается с БД напрямую, Ui посылает запрос в API, для этого реализован сервис save-data, которого есть метод addMessage(), код которого представлен ниже.

addMessage(val:any){

    return this.http.post(this.APIUrl+'/DB',val);

}

Где APIUrl это константа, которая хранит ссылку по которой можно обратиться к приложению в текстовом виде: readonly APIUrl="https://localhost:44357/api";

Какие были проблемы в процессе разработки?

Процесс разработки шел в следующем порядке: БД – бэк – фронт. С БД проблем не возникло, а вот с бэкендом была одна проблема. В VS 2019 пакеты устанавливаются/обновляются при помощи NuGet(всего пару раз с ней сталкивался) и в начале разработки мне понадобилось поставить несколько пакетов, но для каждого из них Get выдавал ошибку «пакет не найден» я пробовал устанавливать пакеты разными способами, но не вышло. В итоге я заметил, что у NuGet есть дефолтный источник пакетов, по которому он ищет пакеты и обновления для них. Этот дефолтный источник пакетов вел в мои же локальные файлы… Заметив это я добавил новый источник пакетов, который ссылается на nuget.org и все заработало.

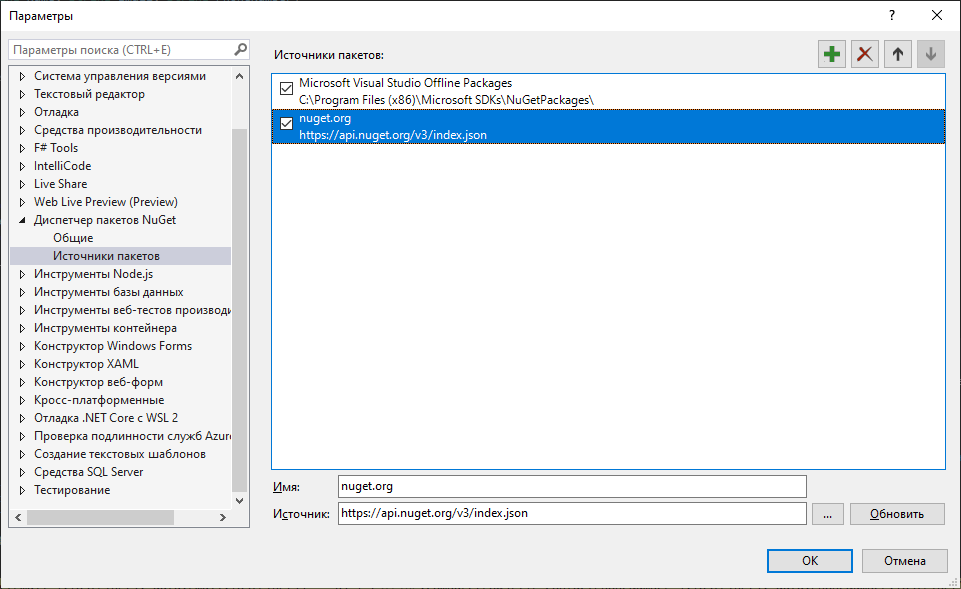


Рис. 3 Средства -> Диспетчер пакетов NuGet -> Параметры диспетчера пакетов -> правильный источник пакетов

Вторая проблема возникла при разработке фронтенда и связана она с добавлением сторонних компонентов, а именно dropdownMenu и reCaptcha. Проблема была вызвана тем, что Angular это относительно новая для меня платформа и я не сразу разобрался как правильно добавлять сторонние компоненты.

Приложение

Здесь представлен код из контроллера, который отправляет данные в БД. Я не хотел вставлять его выше, т.к. он занимает много места.

Если вкратце. Метод GetContacts() реализует select запрос для получения всех контактов, это нужно чтобы проверить существует ли контакт и используется в методе Post(). Методы InsertIntoMessageTextTable() и InsertIntoContactsTable() реализуют POST запросы в соответствующие таблицы, они были вынесена отдельно во избежание повторения блоков кода. Метод Post() использует все перечисленные выше методы с целью проверки существования контакта и помещения данных в соответствующие таблицы.

public class DBController : ApiController

{

public DataTable GetContacts()

{

string query = @"select ContactID, ContactEmail, ContactPhoneNumber from dbo.Contacts";

DataTable table = new DataTable();

using (var con = new SqlConnection(ConfigurationManager.ConnectionStrings["Messages"].ConnectionString))

using (var cmd = new SqlCommand(query, con))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return table;

}

public Boolean InsertIntoMessageTextTable(int messageTopic, int contactID, string text)

{

try{

Console.WriteLine(messageTopic.ToString());

string query = @"

insert into dbo.MessageText values

(

'" + messageTopic + @"',

'" + contactID + @"',

'" + text + @"'

)";

DataTable table = new DataTable();

using (var con = new SqlConnection(ConfigurationManager.ConnectionStrings["Messages"].ConnectionString))

using (var cmd = new SqlCommand(query, con))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return true;

}

catch (Exception)

{

return false;

}

}

public Boolean InsertIntoContactsTable(String Name, String Email, String PhoneNumber)

{

try

{

string query = @"

insert into dbo.Contacts values

(

'" + Name + @"',

'" + Email + @"',

'" + PhoneNumber + @"'

)";

DataTable table = new DataTable();

using (var con = new SqlConnection(ConfigurationManager.ConnectionStrings["Messages"].ConnectionString))

using (var cmd = new SqlCommand(query, con))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return true;

}

catch (Exception)

{

return false;

}

}

public string Post(Data data)

{

DataTable existingTable = GetContacts();

DataRow[] rows = existingTable.Select();

int contactID = 0;

int lastContactID = (int)rows[rows.Length-1]["ContactID"];

for (int i = 0; i < rows.Length; i++)

{

if ((String.Compare(rows[i]["ContactEmail"].ToString(), data.Email.ToString()) == 0) | (String.Compare(rows[i]["ContactPhoneNumber"].ToString(), data.PhoneNumber.ToString()) == 0))

{

contactID = (int)rows[i]["ContactID"];

}

}

if (contactID != 0)

{

if (InsertIntoMessageTextTable(data.MessageTopic, contactID, data.MessageText))

{

return "Ваше сообщение отправлено!";

}

else

{

return "Что-то не так с отправкой вашего сообщения!";

}

}

else

{

if (InsertIntoContactsTable(data.Name, data.Email, data.PhoneNumber) & InsertIntoMessageTextTable(data.MessageTopic, lastContactID+1, data.MessageText))

{

return "Ваше сообщение и контакт отправлены!";

}

else

{

return "Что-то не так с отправкой Вашего контакта и сообщения!";

}

}

}

}