**Лабораторная работа № 1**

**Тема**: SOA с калькулятором

**Задание**: Разработать Web-приложение (можно Win-приложение) с формой в виде некоторого произвольного калькулятора, вычислительной системы. Все вычислительные бизнес-процессы калькулятора вынести во внешний Web-сервис.

**Ход работы**:

Сервис был размещён на бесплатном хостинге <https://heroku.com/>(рис.1), на данный момент сервис доступен по ссылке <https://calculator-2021.herokuapp.com/> (рис.2). Сам сервис был построен на языке программирования Python с помощью библиотеки flask, общение с клиентом происходит посредством Api (клиенту приходит ответ в виде json). Клиент реализован так же на языке программирования Python как descktop – приложение (рис.3).

На рисунке 4 и 5 показана работа клиентской части.

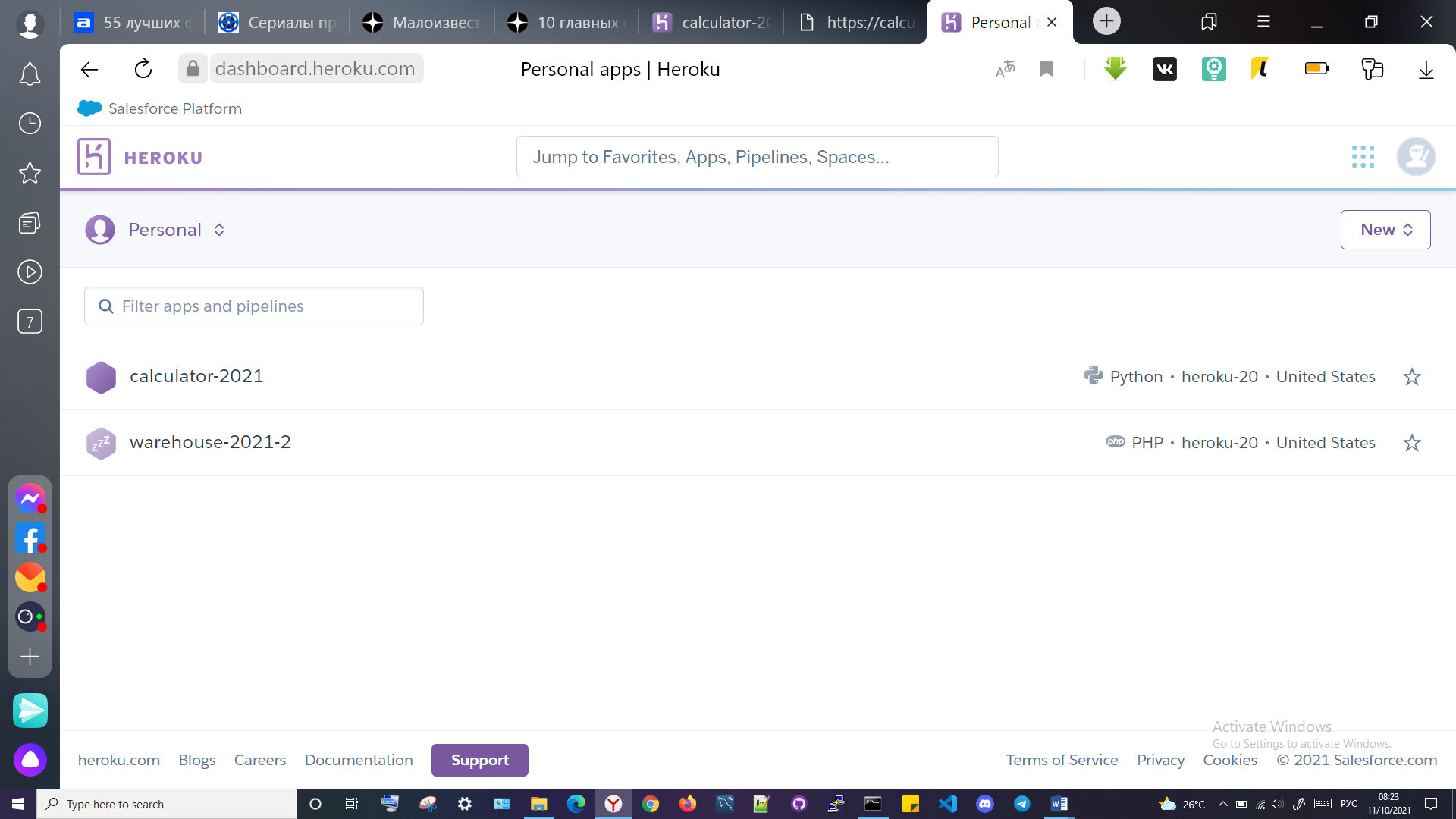


Рисунок 1 – Хостинг Heroku

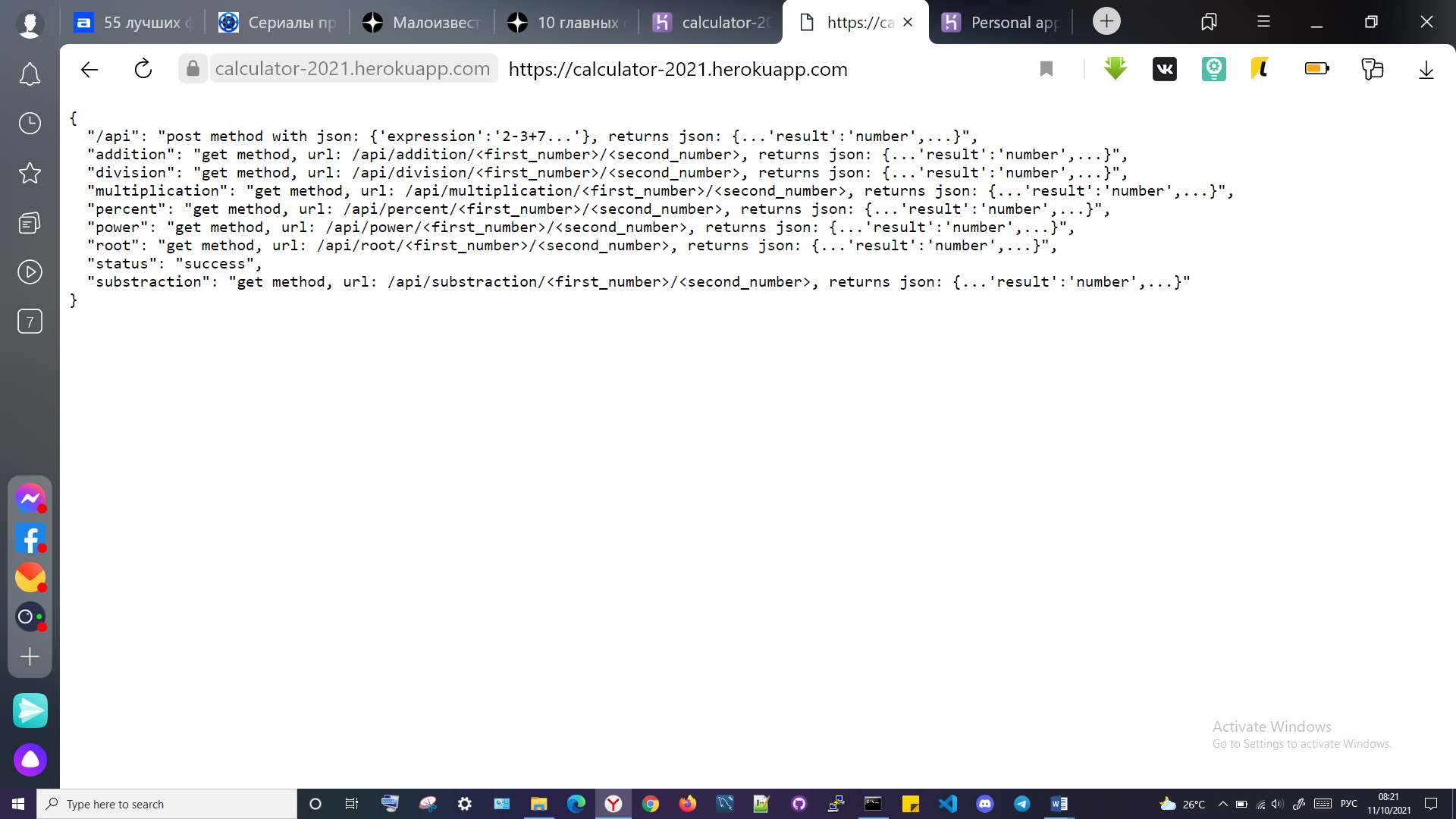


Рисунок 2 – Сервис калькулятора

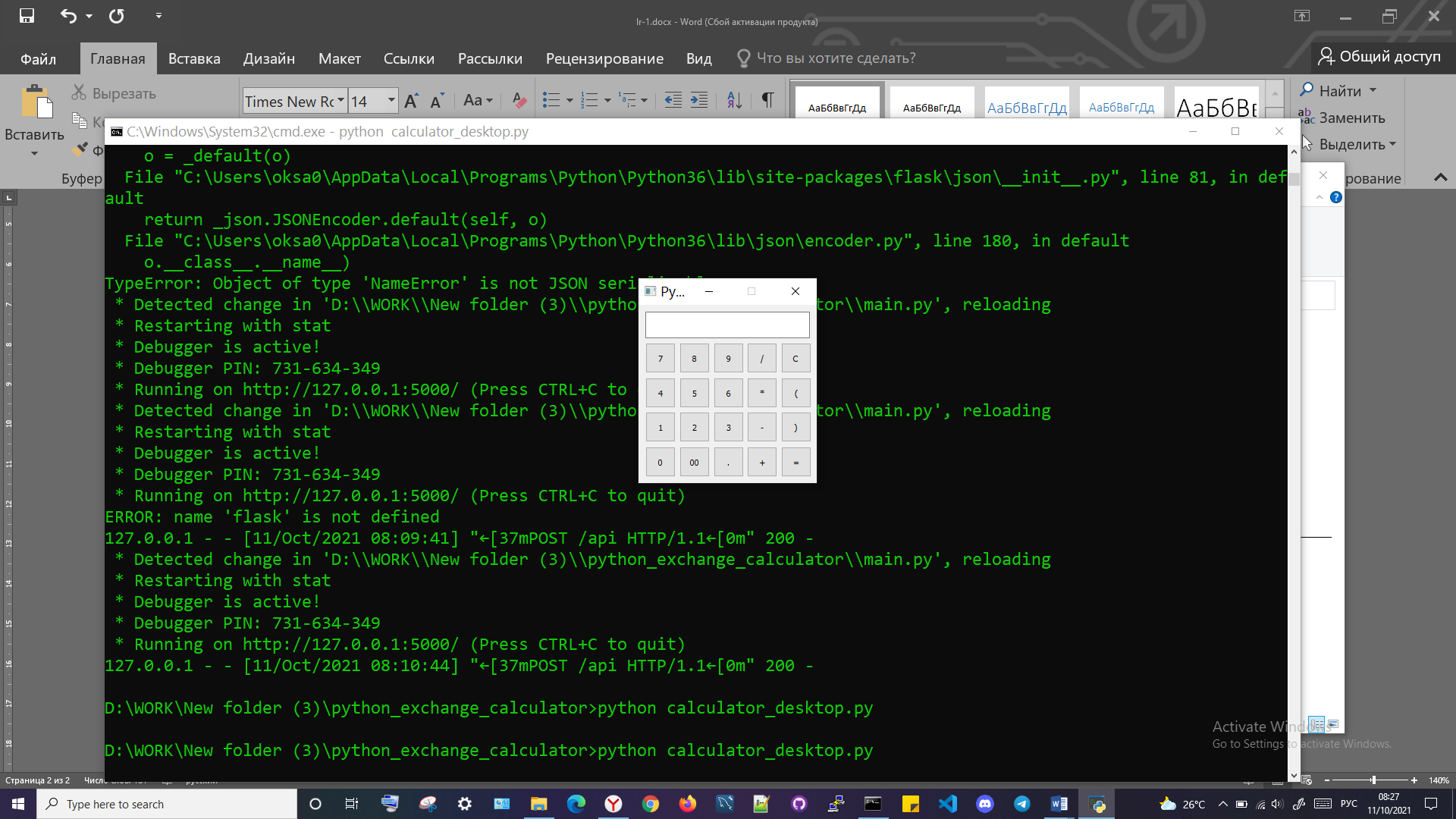


Рисунок 3 – Клиентская часть

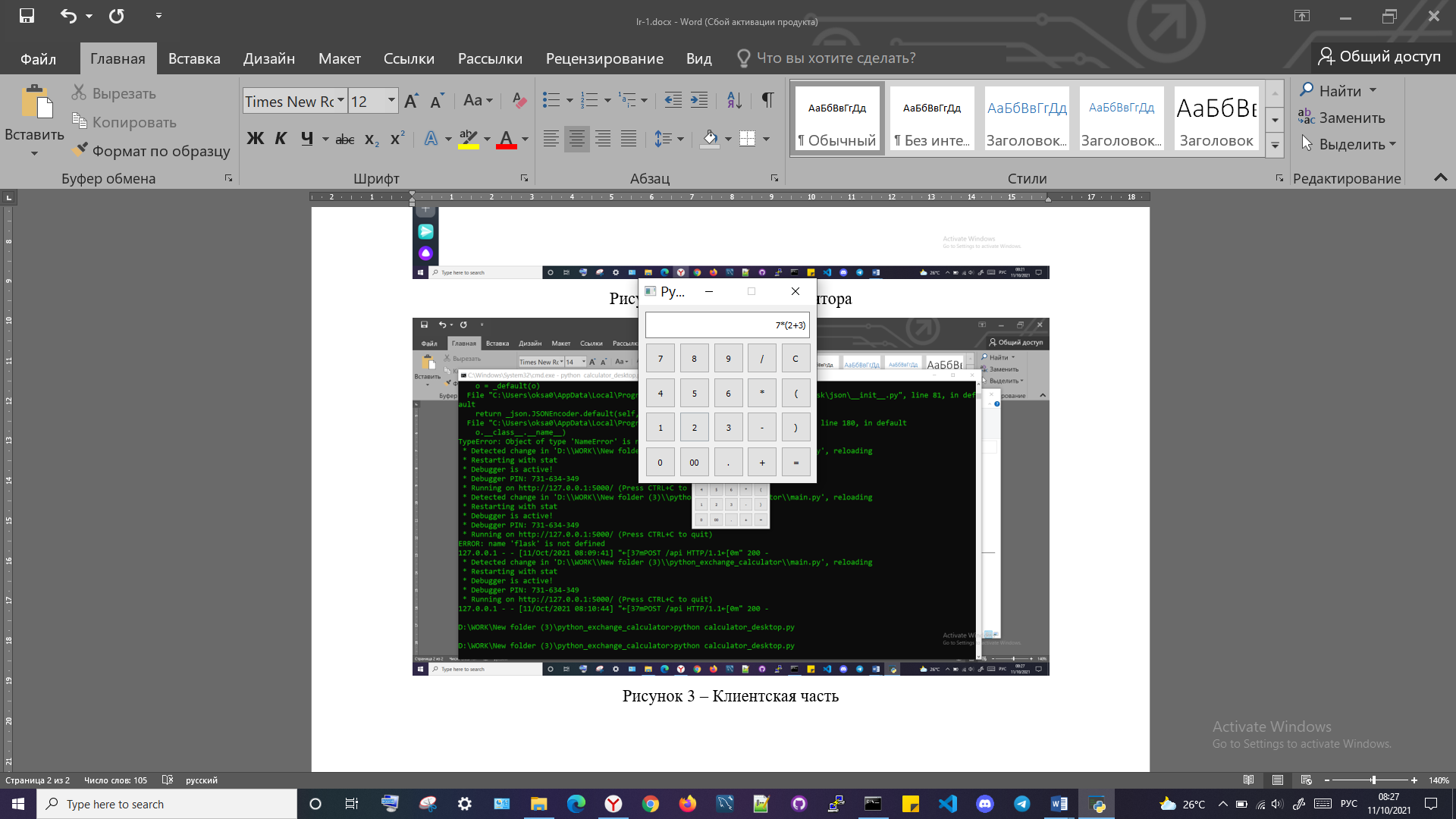


Рисунок 4 – Работа клиентской части(пример)

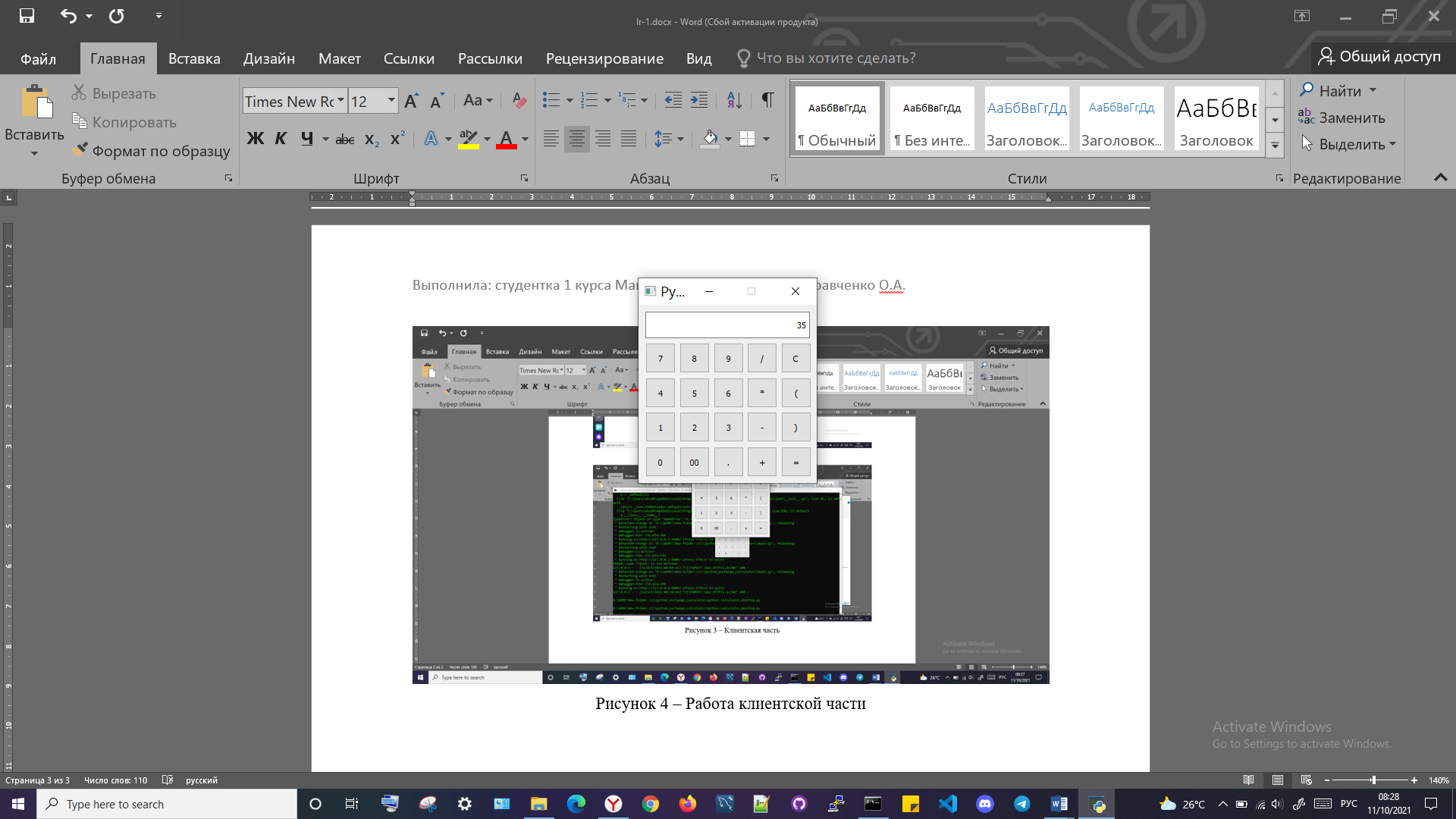


Рисунок 5 – Работа клиентской части (результат)

**Контрольные вопросы**

**1. Чем отличаются Web-сервисы от Web-сайтов?**

Веб-сервис - это любое программное обеспечение, которое доступно через Интернет и использует стандартизированную систему обмена сообщениями XML. XML используется для кодирования всех сообщений в веб-сервис. Например, клиент вызывает веб-службу, отправляя сообщение XML, а затем ожидает соответствующего ответа XML. Поскольку вся связь осуществляется в XML, веб-сервисы не привязаны к какой-либо одной операционной системе или языку программирования — Java может общаться с Perl; Приложения Windows могут общаться с приложениями Unix.

Веб-сайт - это интерфейс к веб-сервисам и доп. странички, которые, строго говоря, не относятся к веб-сервисам (например, страничка - "О нас"). Представительские функции (чаще всего). Компания через сайт взаимодействует со своими клиентами (существующими и потенциальными), партнёрами и тд. Компания публикует информацию.

**2. Каков язык передачи данных Web-сервисов и Web-сайтов?**

XML - Расширяемый язык разметки, предназначенный для хранения и передачи структурированных данных

**3. Назначение протоколов SOAP и WSDL?**

WSDL — это язык на основе XML для описания веб-сервисов и способов доступа к ним. WSDL расшифровывается как язык описания веб-сервисов. WSDL — это основанный на XML протокол для обмена информацией в децентрализованных и распределенных средах. Определение WSDL описывает, как получить доступ к веб-службе и какие операции она будет выполнять.

SOAP — это основанный на XML протокол для обмена информацией между компьютерами. SOAP для связи между приложениями. SOAP предназначен для общения через Интернет. SOAP не зависит от платформы. SOAP не зависит от языка. SOAP прост и расширяем. SOAP позволяет обойти брандмауэры. SOAP будет разработан как стандарт W3C.

**4. Каковы задачи посредников (прокси) в приложениях для общения с Web-сервисами?**

Прокси -это класс, который .NET использует для связи с удаленной веб-службой. это то, что делает тяжелую работу за кулисами, чтобы отформатировать конверт SOAP и сериализовать ваши запросы.

**5. В чём суть SOA?**

SOA — это набор архитектурных принципов, не зависящих от технологий и продуктов, совсем как полиморфизм или инкапсуляция.

Основные идеи: Сочетаемость приложений, ориентированных на пользователей. Многократное использование бизнес-сервисов. Независимость от набора технологий. Автономность (независимые эволюция, масштабируемость и развёртываемость).

**6. Каковы цели преследуются при построении SOA?**

Основные идеи: Сочетаемость приложений, ориентированных на пользователей. Многократное использование бизнес-сервисов. Независимость от набора технологий. Автономность (независимые эволюция, масштабируемость и развёртываемость).

**7. Каковы уровни абстракции SOA в информационной системе предприятия?**

Уровень 1. Реализация отдельных Web-сервисов. Это начальный уровень развертывания SOA, на котором технологии Web-сервисов используются для разработки новых приложений или преобразования существующих.

Уровень 2. Сервисно-ориентированная интеграция бизнес-функций. На этом уровне мы уже добились преобразования приложений в сервисы и хотим интегрировать их таким образом, чтобы реализовать определенную бизнес-задачу.

Уровень 3. Трансформация ИТ-инфраструктуры в масштабе предприятия. Здесь речь идет о сервисно-ориентированной интеграции приложений и процессов уже в масштабах всей компании, причем согласованный, сервисный подход к ИТ-инфраструктуре распространяется не только на внутренние подразделения, но и на партнеров и поставщиков. Здесь вступают в действие системы, обеспечивающие более глубокую детализацию разработки и интеграцию сервисов с учетом всех уже рассмотренных типов интеграции.

Уровень 4. Изменения в бизнесе. Последний уровень связан с изменениями в самих способах ведения бизнеса в ответ на глобальные трансформации ИТ-инфраструктуры.

**8. В чём суть REST и RPC подходов к построению распределённых информационных систем? Что и когда следует применять?**

Удаленный вызов процедуры (Remote Procedure Call) — это спецификация, которая позволяет удаленно выполнять функцию в другом контексте. RPC расширяет понятие локального вызова процедуры, но помещает его в контекст HTTP API. Клиент вызывает удаленную процедуру, сериализует параметры и дополнительную информацию в сообщении и отправляет это сообщение на сервер. Получив сообщение, сервер десериализует его содержимое, выполняет запрошенную операцию и отправляет результат обратно клиенту. Стаб сервера и стаб клиента берут на себя сериализацию и десериализацию параметров.

Передача состояния представления (REST): предоставление данных в качестве ресурсов. REST — самоописательный архитектурный стиль API, определяемый набором архитектурных ограничений и предназначенный для широкого внедрения многими потребителями API.