# Постановка задачи

Разработать веб- и десктопное приложение "Online Calculator", которое будет объединять функции выполнения базовых математических операций, тригонометрических расчетов и конвертации единиц измерения. Приложение должно быть интуитивно понятным и доступным для пользователей всех возрастов, обеспечивая удобный интерфейс на различных устройствах.

Основной задачей является создание функционала, позволяющего пользователям легко выполнять вычисления и конверсии, а также сохранять и просматривать историю своих операций. Приложение должно поддерживать надежную обработку ошибок ввода и предоставлять пользователям возможность настраивать интерфейс под свои предпочтения.

В результате работы необходимо получить готовое решение, которое будет протестировано на различных устройствах и браузерах, с учетом отзывов пользователей для дальнейшего улучшения и оптимизации функционала.

# Стратегия дизайна

1. Заинтересованные стороны

* **Пользователи:** Обычные пользователи, студенты, профессионалы, работающие с математикой и финансами.
* **Администраторы:** Лица, управляющие пользователями и данными приложения.
* **Разработчики:** Команда, занимающаяся созданием и поддержкой приложения.
* **Заказчик:** Организация, финансирующий проект.

2. Видение продукта заинтересованными лицами

Приложение должно решать следующие задачи:

* Выполнение базовых и сложных арифметических операций.
* Конвертация систем счисления.
* Хранение и отображение истории вычислений для пользователей.
* Работа с матрицами, включая сложение и умножение.
* Поддержка авторизации для разных пользователей с сохранением индивидуальных настроек.

3. Конфликты и противоречия

* **Функциональность:** Обычные пользователи могут требовать упрощенный интерфейс, в то время как администраторы могут нуждаться в расширенных функциях для управления данными.
* **Приоритеты:** Разработчики могут предлагать различные технологии, тогда как заказчик может иметь ограничения по бюджету и срокам.
* **Пользовательский интерфейс:** Конфликты между требованиями к простоте использования и функциональностью.

4. Задачи бизнеса, маркетинга и брендинга

* **Задачи бизнеса:** Обеспечение удобного и функционального инструмента для пользователей, привлечение новых пользователей через рекламу и маркетинг.
* **Задачи маркетинга:** Определение целевой аудитории, создание бренда, формирование привлекательного имиджа приложения.
* **Задачи брендинга:** Разработка уникального стиля и логотипа, создание корпоративной идентичности.

5. Измеримые критерии успешности

* **Пользовательская активность:** Количество зарегистрированных пользователей и активных сессий.
* **Отзыв пользователей:** Оценки в магазинах приложений, отзывы и рекомендации.
* **Функциональная стабильность:** Количество ошибок и сбоев в работе приложения.
* **Финансовые показатели:** Динамика доходов от подписок или рекламы.

6. Технические возможности и ограничения

* **Технологии разработки:** Использование языков программирования (С++, Java), фреймворков (Angular, Swagger).
* **Платформы:** Веб-приложение, десктопное приложение.
* **Интеграции:** Подключение к API для конверсии систем счислений, базы данных для хранения пользовательских данных.

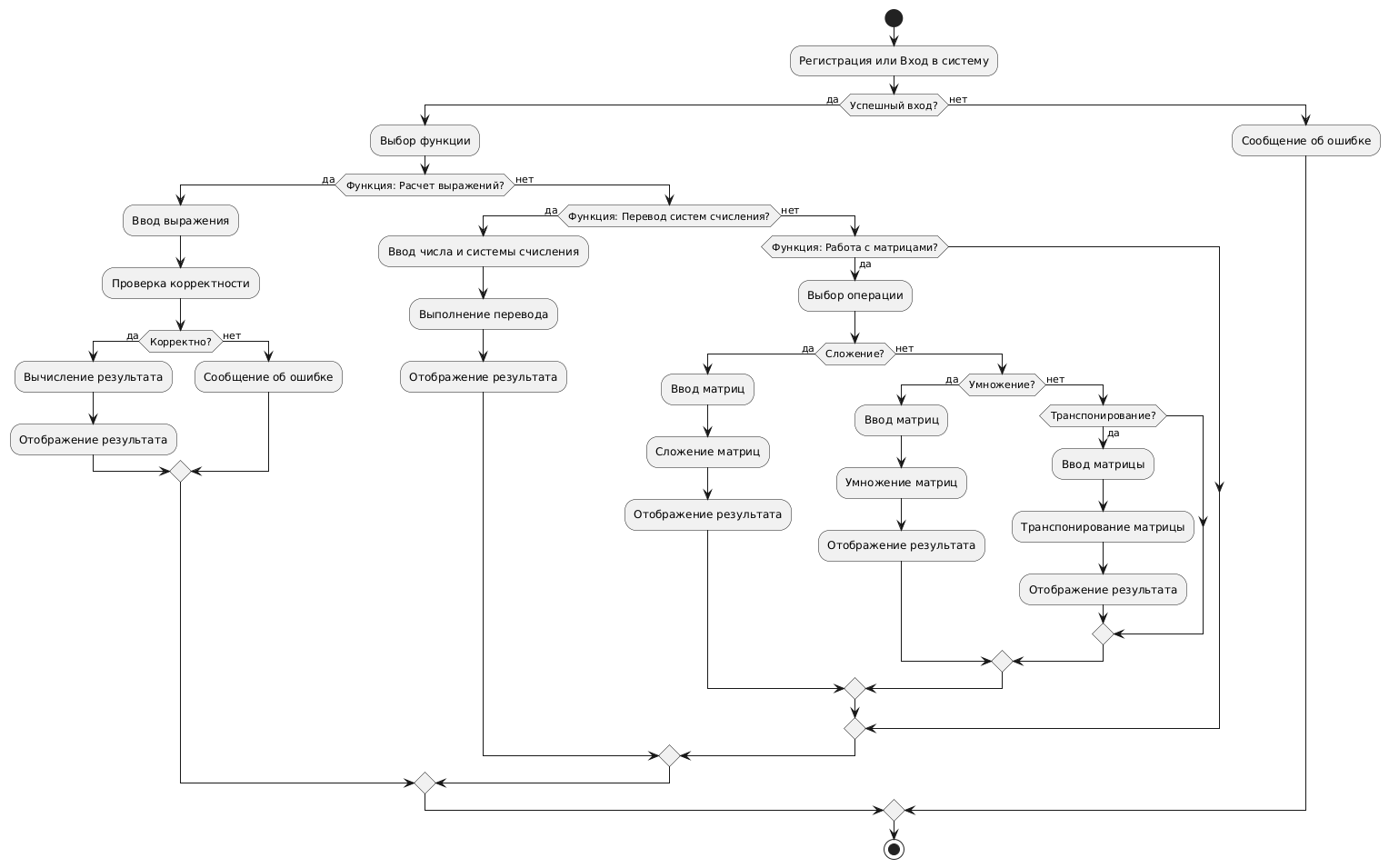
7. Представления заинтересованных лиц о пользователях

* **Целевая аудитория:** Студенты, работающие профессионалы, преподаватели, люди, занимающиеся финансами. Пользователи, которым нужны простые и доступные инструменты для вычислений.

8. Бюджет и график проекта

* **Бюджет:** Оценка затрат на разработку и поддержку. Примерная сумма — 100,000-150,000 долларов.
* **График реализации:**
  + Этап 1: Исследование и проектирование (1-2 месяца).
  + Этап 2: Разработка прототипа (2-3 месяца).
  + Этап 3: Тестирование и доработка (1-2 месяца).
  + Этап 4: Запуск и маркетинг (1 месяц).

# Диаграммы бизнес-процессов



# 

# Сценарии вариантов использования (составить сценарии для 3-5 вариантов использования) на языке PlantUML

#### **1. Расчет выражений**

* **Актор:** Пользователь
* **Предусловие:** Пользователь авторизован
* **Сценарий:**
  1. Пользователь вводит математическое выражение.
  2. Система проверяет корректность ввода.
  3. Система вычисляет результат.
  4. Система отображает результат пользователю.

#### **2. Перевод систем счисления**

* **Актор:** Пользователь
* **Предусловие:** Пользователь авторизован
* **Сценарий:**
  1. Пользователь выбирает систему счисления для перевода.
  2. Пользователь вводит число.
  3. Система выполняет перевод.
  4. Система отображает результат.

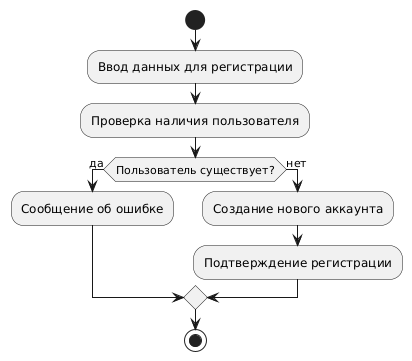
#### **3. Управление аккаунтом**

* **Актор:** Пользователь
* **Предусловие:** Пользователь авторизован
* **Сценарий:**
  1. Пользователь выбирает опцию управления аккаунтом.
  2. Пользователь может изменить пароль или удалить аккаунт.
  3. Система подтверждает изменения.

# 

# Диаграммы деятельности (1-3) на языке PlantUML

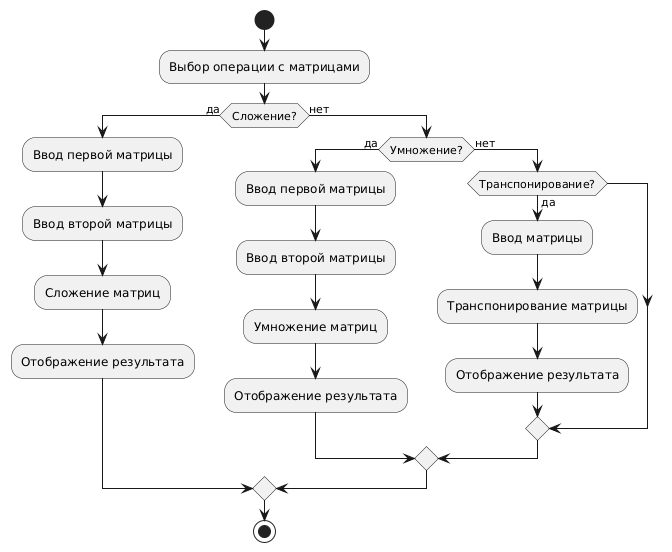
### **Диаграмма деятельности для регистрации пользователя**



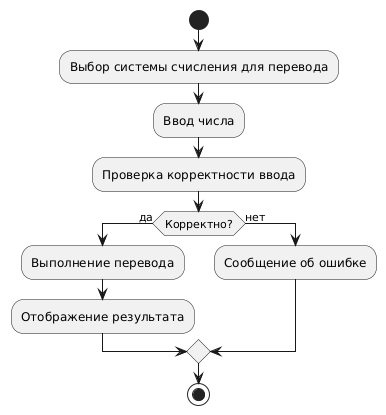
### 

### 

### **Диаграмма деятельности для работы с матрицами**

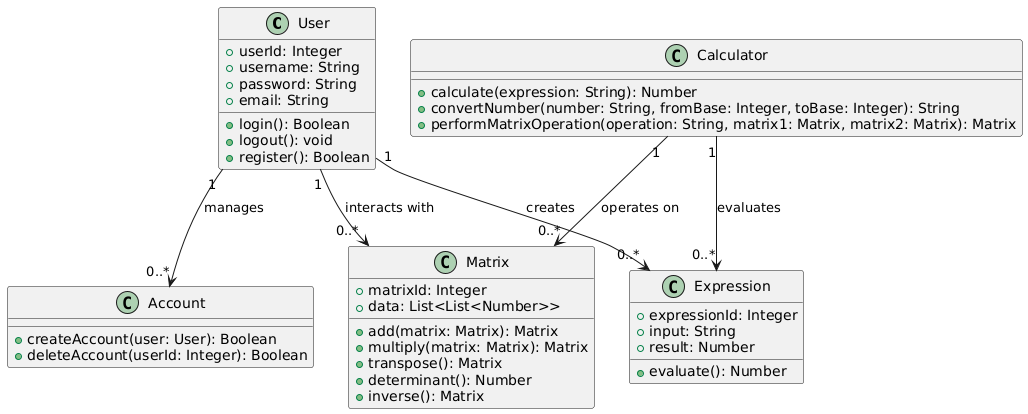


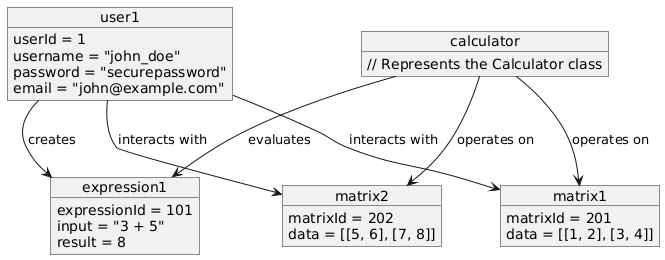
### **Диаграмма деятельности для перевода чисел между системами счисления**



# 

# Диаграммы классов и объектов (не менее 5) на языке PlantUML

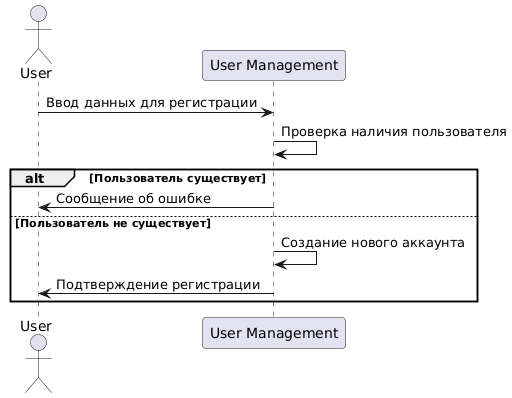




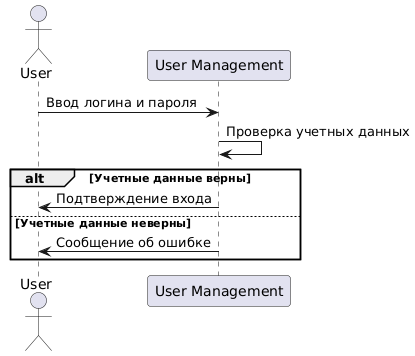
# 

# Диаграммы последовательности (не менее 5) на языке PlantUML

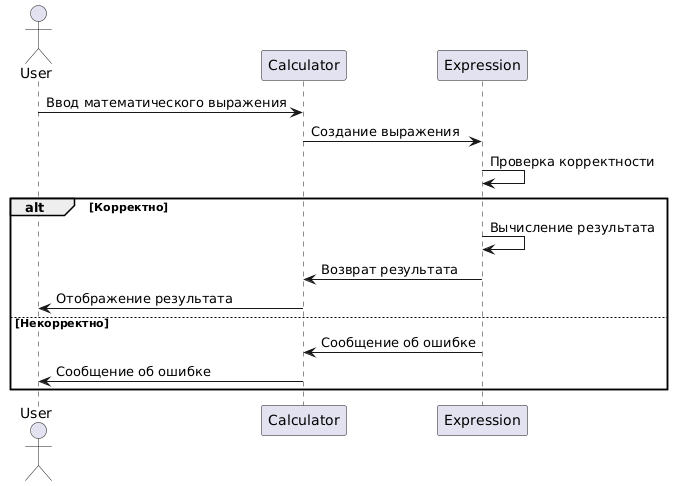
### **1. Диаграмма последовательности для регистрации пользователя**



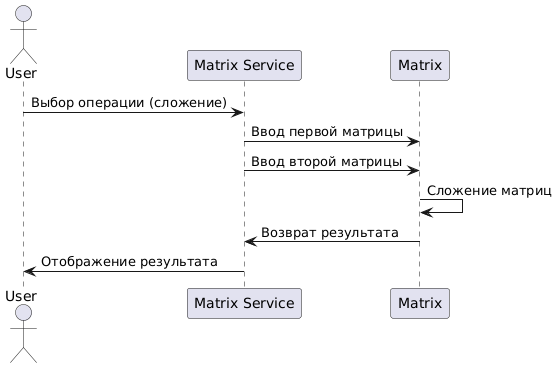
### **2. Диаграмма последовательности для входа в систему**



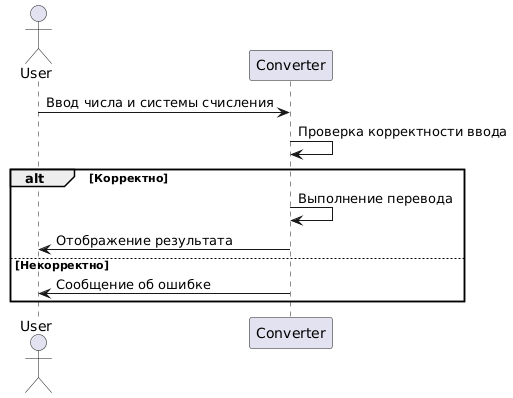
### **3. Диаграмма последовательности для вычисления выражения**



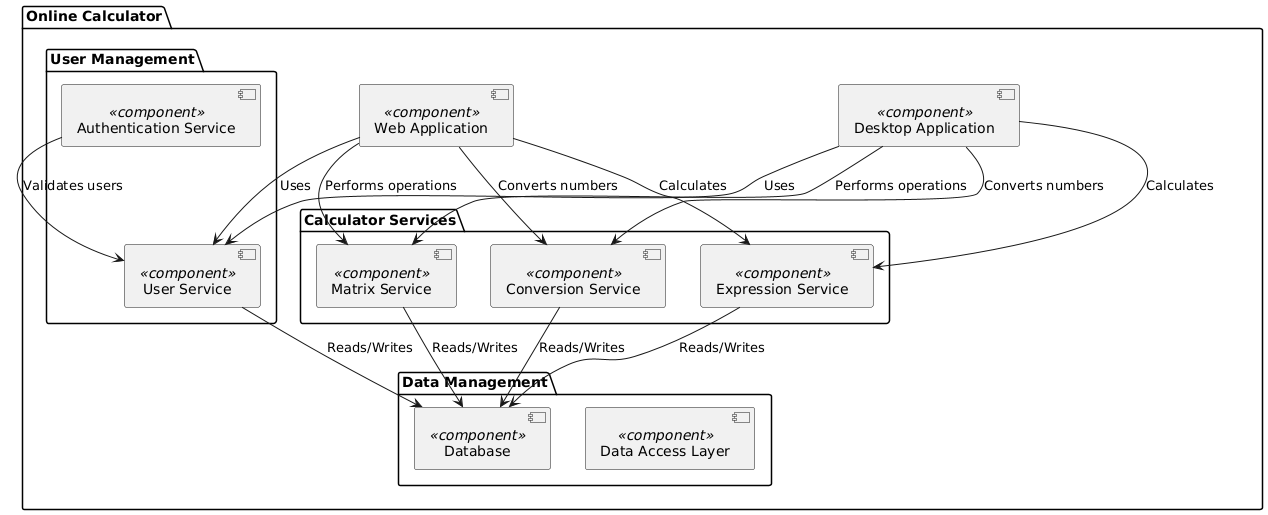
### **4. Диаграмма последовательности для работы с матрицами**



### **5. Диаграмма последовательности для перевода чисел между системами счисления**

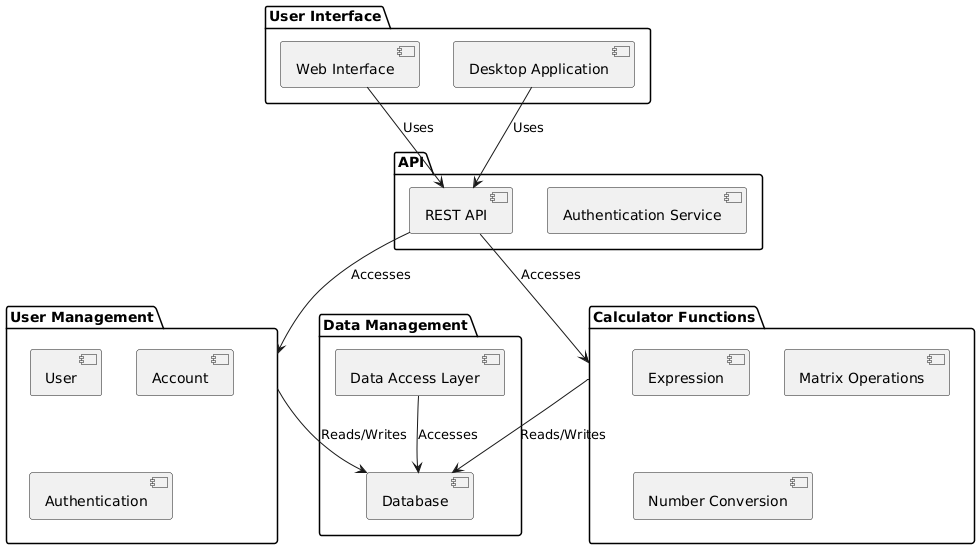


# Диаграммы компонентов на языке PlantUML

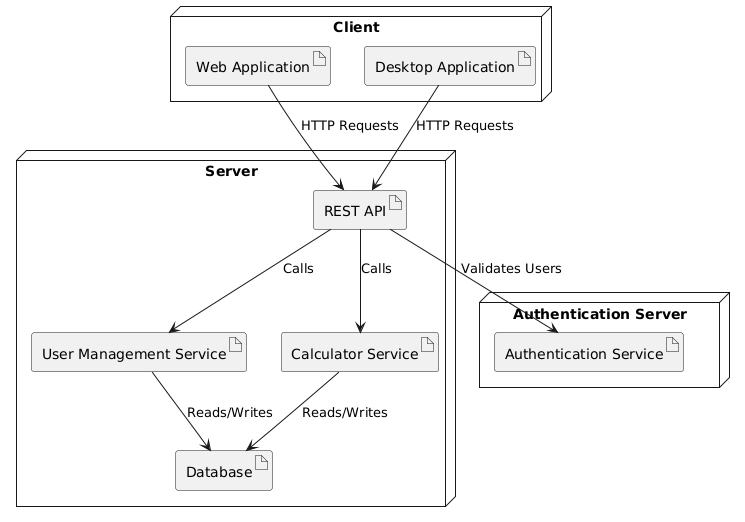


# 

# Диаграмма пакетов на языке PlantUML

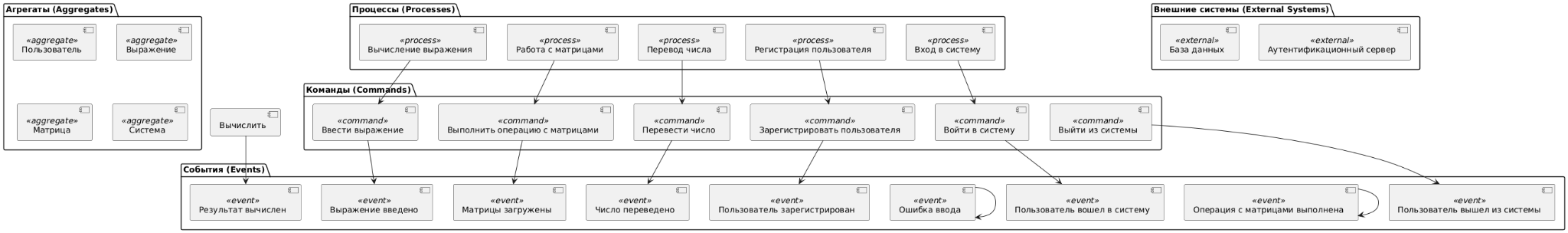


# Диаграмма развертывания на языке PlantUML



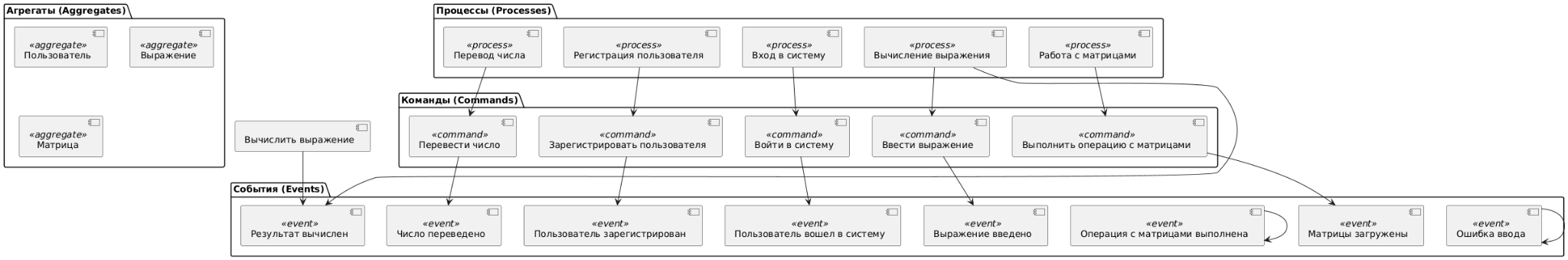
# 

# Спецификация системы с использованием метода Event Storming на языке PlantUML



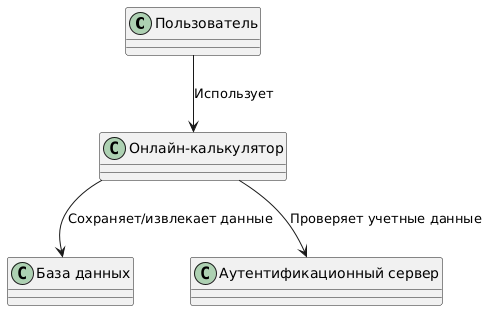
# 

# Спецификация системы с использованием метода Event Modeling на языке PlantUML



# Архитектура системы с использованием метода C4 Model на языке PlantUML

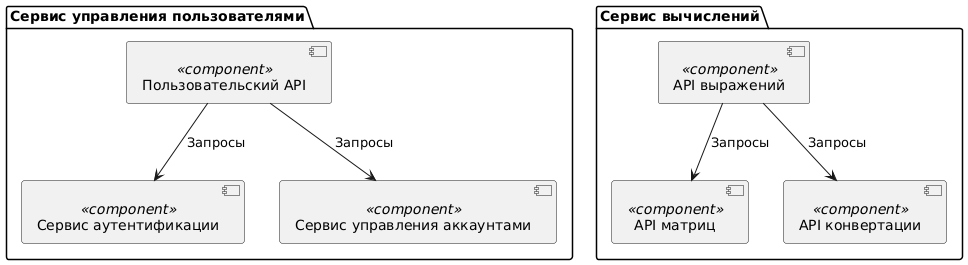
### **1. Диаграмма уровня контекста**



### 

### 

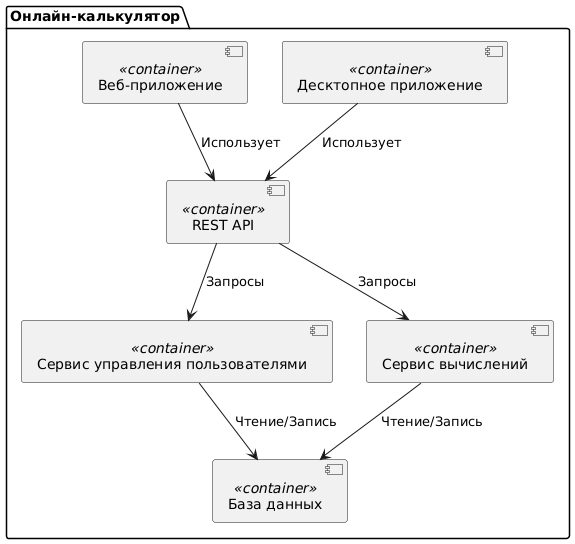
### **2. Диаграмма уровня компонентов**



### 

### 

### **3. Диаграмма уровня контейнеров**



### **4. Диаграмма уровня кода**

