МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук

Кафедра технологий обработки и защиты информации

Курсовая работа

Ведение домашнего бюджета. Рекомендации по сокращению расходов

09.03.02 Информационные системы и технологии
Защита информации в компьютерных системах

Обучающийся _____ Зеленев А.А., 3 курс, д/о

Обучающийся _____ Михно Д.А., 3 курс, д/о

Обучающийся _____ Ганигин К.И., 3 курс, д/о

Руководитель _____ Тарасов В.С., старший преподаватель

Воронеж 2021

Содержание

Введение

Деньги – один из важнейших ресурсов в нашем мире. Люди трудятся, чтобы заработать, и тратят, чтобы получить блага: продукты, техника, образование, недвижимость. Отсюда возникает два вопроса: как заработать деньги и как их оптимально потратит. В рамках данного проекта мы собираемся ответить именно на второй вопрос. Ежедневно человек совершает около 10-15 операций по карте, зачастую даже не контролируя, сколько он тратит. В этой суете тяжело проконтролировать не просто месячные расходы, а даже дневные. Впрочем, проблему эту решили довольно просто. Ключ к спасению - ведение дневника расходов. Запись доходов, расходов, классификация трат, подведение итогов в конце месяц. В конце концов общий анализ и решения по дальнейшим действиям оставались на человеке. Но мы живем в век высоких технологий, мало кому захочется вести и хранить бумажную сводку покупок, да еще к тому же не актуальную. Вероятнее, люди предпочтут приложение, в которое можно будет быстро ввести информацию по тратам, чтобы оно обрабатывало информацию об изменениях бюджета и показывало графики, прогнозы и рекомендации об улучшении текущего финансового положения. Такому приложению и посвящена данная курсовая работа.

1 Постановка задачи.

Цель данной курсовой работы – создать веб-приложение для ведения и контроля семейного бюджета. Бюджет должен представлять из себя статьи доходов и статьи расходов, соответственно, в процессе ведения домашнего бюджета необходимо вести учет данных показателей, чтобы потом соотносить фактические траты с планируемыми.

Веб-приложение должно обладать следующей функциональностью:

— Регистрация расходов/доходов.

— Составление плана расходов на месяц.

— Предоставление рекомендаций по снижению расходов.

— Создание финансовых целей.

— Расчет размера финансового резерва.

— Формирование отчета по расходам за определенный период времени.

Для этого потребуется решить следующие задачи:

— Изучить предметную область

— Провести анализ существующих решений

— Определить какие будут использованы средства и методы для данного приложения

— Разработать приложение, которое реализует вышеперечисленные функциональности.

2 Анализ предметной области

2.1 Глоссарий

Расходы - затраты, которые ведут к уменьшению экономических выгод в результате выбытия денежных средств.

Доходы - денежные средства, полученные физическим лицом в результате какой-либо деятельности за определённый период времени.

Счет – учетная позиция, создаваемая для хранения информация о текущем денежном состоянии пользователя.

Бюджет - финансовый план определённого субъекта (семьи, бизнеса, организации, государства и т. д.), устанавливаемый на определённый период времени.

Семейный бюджет - план доходов, расходов и накоплений, описывающий возможности всех членов семьи в определённый период времени.

Виджет - примитив графического интерфейса пользователя, имеющий стандартный внешний вид и выполняющий стандартные действия.

Веб-приложение - клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Логика вебприложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети.

2.2 Анализ существующих решений

2.2.1 Деньги ОК

Количество установок: более 100 000. Имеется разовая подписка (PRO-версия), расширяющая функциональность.

Тенденция развития: Стабильно 1-2 раза в месяц разработчики выпускают обновления с исправленными ошибками. Так же приложение периодически получает масштабные обновления, но связаны они чаще всего с внешним видом.

Оценка с позиции клиента:

Приложение имеет оценку 4.8 в AppStore. Огромное количество пользователей IOS хвалит интерфейс за его простоту и удобство, за возможность синхронизации и перевод валюты. Однако сообщество Android устройств встретило приложение менее тепло. Оценка 4.2, вечные жалобы на неисправности, никаких ответов от разработчиков.

Функциональности:

— Учет расходов и доходов.

— Неограниченное число счетов и статей (категорий).

— Подстатьи.

— Переводы между счетами.

— Дневные, месячные и годовые отчеты.

— Синхронизация.

— Дневные, недельные, годовые отчеты (PRO-версия).

— Составление бюджета (PRO-версия).

— Виджет для ввода расходов и доходов (PRO-версия).

2.2.2 CoinKeeper

Количество установок: более 1 000 000. Имеется годовая и месячная подписки, а также разовая (навсегда).

Тенденция развития: Стабильно 1-2 раза в месяц разработчики выпускают обновления с исправленными ошибками и добавлением новой функциональности. Каких-то масштабных обновлений, приложение не получало с даты выхода.

Оценка с позиции клиента:

AppStore: Оценка 4.6

GooglePlay: Оценка 4.5

Несмотря на высокую оценку, пользователи постоянно жалуются на разные ошибки, рекламу после покупки подписки и др.

Функциональности:

— Автоматически распознает СМС-сообщения от любых банков.

— Поддержка всех мировых валют.

— Защита данных паролем.

— Общий профиль для семьи.

2.2.3 Monefy

Количество установок: более 5 000 000. Имеется разовая подписка (PRO-версия) расширяющая функциональность.

Тенденция развития: Разработчики выпускают каждые 4-5 дней обновления с исправлением ошибок, улучшением интерфейса.

Оценка с позиции клиента:

AppStore: Оценка 4.8

GooglePlay: Оценка 4.5

Пользователи хвалят приложение практически со всех сторон. Когда клиенты сообщают об ошибках разработчики быстро реагируют на отзыв.

Функциональности:

— Можно просматривать общий журнал расходов или детализировать данные.

— Безопасная синхронизация через учетную запись Google Drive или Dropbox.

— Можно контролировать кредиты, ежедневные и ежемесячные траты.

— Поддержка различных валют.

— Виджеты для быстрого доступа.

— Можно управлять категориями. При этом изменения синхронизируются между устройствами.

— Простое резервное копирование и экспорт данных.

— Можно использовать несколько учетных записей и считать расходы вместе с членами семьи.

— Простой подсчет расходов и доходов при помощи встроенного калькулятора.

— Возможность вести учет счетов в разных валютах (PRO-версия).

— Наличие пароля обеспечит безопасность данных (PRO-версия).

2.2.4 Moneon

Количество установок: более 50 000. Имеется месячная и годовая расширяющая функциональность подписка.

Тенденция развития: Разработчики быстро решают проблемы. 2-3 обновления в месяц с исправлением ошибок и каждые 6-7 месяцев глобальные обновления, которые расширяют функционал программы.

Оценка с позиции клиента:

AppStore: Оценка 4.7

GooglePlay: Оценка 3.9

Клиенты IOS хвалят приложения со всех сторон и дают советы разработчикам. Клиенты Android очень сильно критикуют приложение, разработчики создают иллюзию реакции на отзывы, но не сильно заинтересованы в своих пользователях.

Функциональности:

— Учет личных расходов и расчет бюджета.

— Трекер расходов и контроль доходов.

— Совместный онлайн кошелек с семьей или друзьями (PRO версия).

— Личный бюджет, его планирование.

— Категории для анализа расходов и доходов.

— Множество валют на выбор.

— Выбор периода для финансового анализа.

— Защита паролем.

2.2.5 Wallet

Количество установок: более 5 000 000. Имеется месячная, годовая и разовая покупка полной функциональности приложения.

Тенденция развития: Разработчики выпускают 3-4 обновления с исправленными ошибками в месяц, каждые 4-6 месяцев выходят крупные обновления, расширяющие функционал.

Оценка с позиции клиента:

AppStore: Оценка 4.0

GooglePlay: Оценка 4.7

Пользователи Android высоко оценивают приложение, при этом не скупы на критику и очень категорично высказываются о недоработках. Разработчики выборочно реагируют на отзывы.

Пользователи IOS приняли приложение чуть хуже, однако разработчики быстро адаптировали приложение под сообщество.

Функциональность:

— Импорт банковских транзакций.

— Экспорт в CSV/XLS/PDF.

— Умный помощник.

— Поддержка нескольких валют.

— Карты лояльности.

— Автоматическая синхронизация с банком (PRO-версия).

— Чеки и гарантии.

— Категории и шаблоны.

— Списки покупок.

— Управление долгами.

2.3 Анализ поставленной задачи

2.3.1 Use case

Рисунок 1 – Use case

При работе с приложением Budger неавторизованный пользователь, именуемый далее гостем, имеет следующие возможности:

— Просмотр гостевой страницы

— Зарегистрироваться

— Войти в систему

В свою очередь, авторизованный пользователь данного приложения, то есть пользователь, который зарегистрировался и вошел в систему под своими учетными данными, обладает приведенными ниже возможностями:

— Добавить расходы.

— Добавить финансовые цели.

— Получить рекомендации по расходам.

— Получить статистику расходов.

— Выйти из системы.

2.3.2 Class diagram

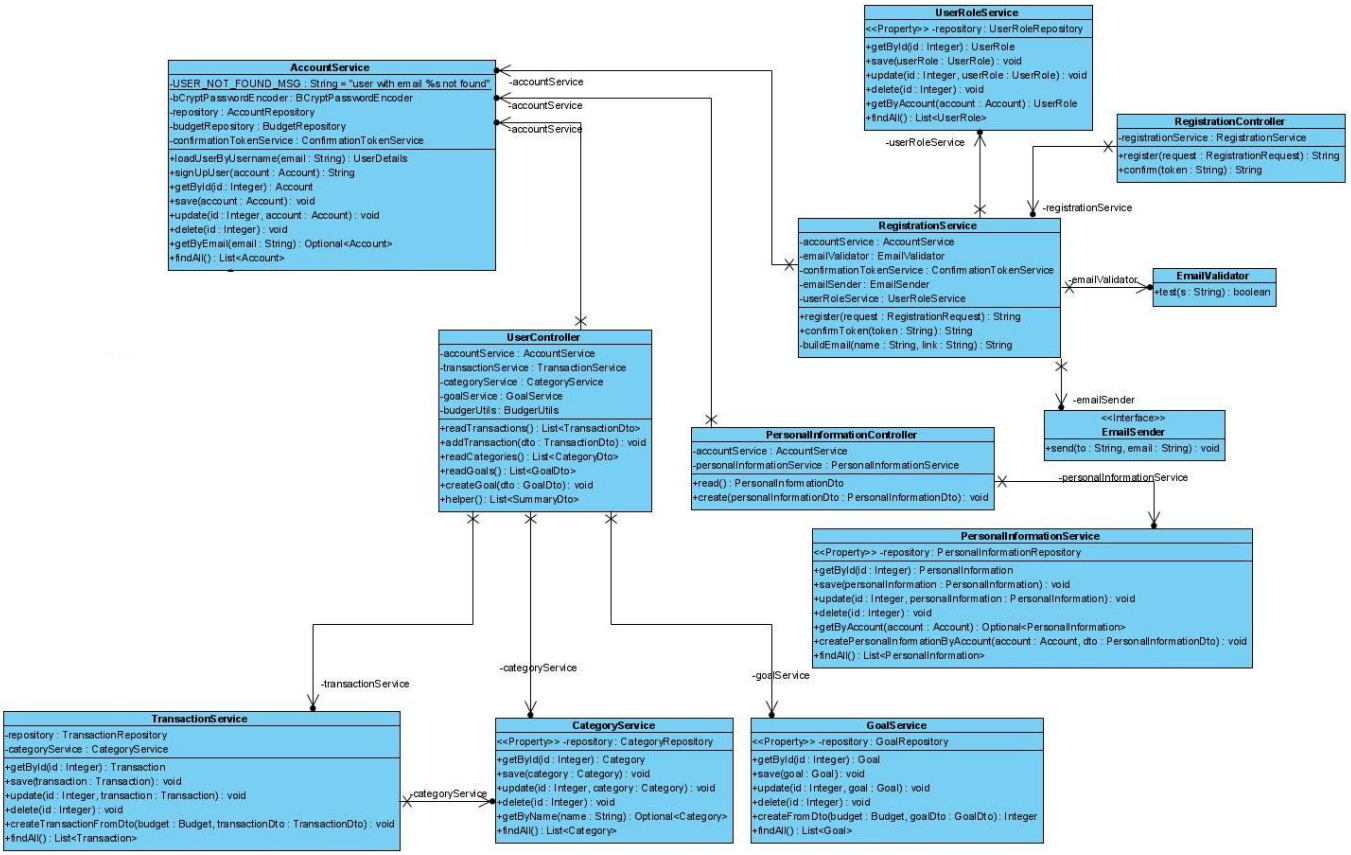

Рисунок 2 – Class diagram

Серверная часть приложения организована следующим образом: классы AccountService, CategoryService, RegistrationService и т.д. обеспечивают получение и запись данных в БД. Для общения с сервером существуют UserController, PersonalInformationController и RedistrationController. Они реализуют основные методы получение данных, которые запрашивает клиент. Контроллеры используют

соответствующие сервисы, которые реализуют основной набор методов для работы с репозиториями, которые работают с базой данных.

2.3.3 Sequence diagram

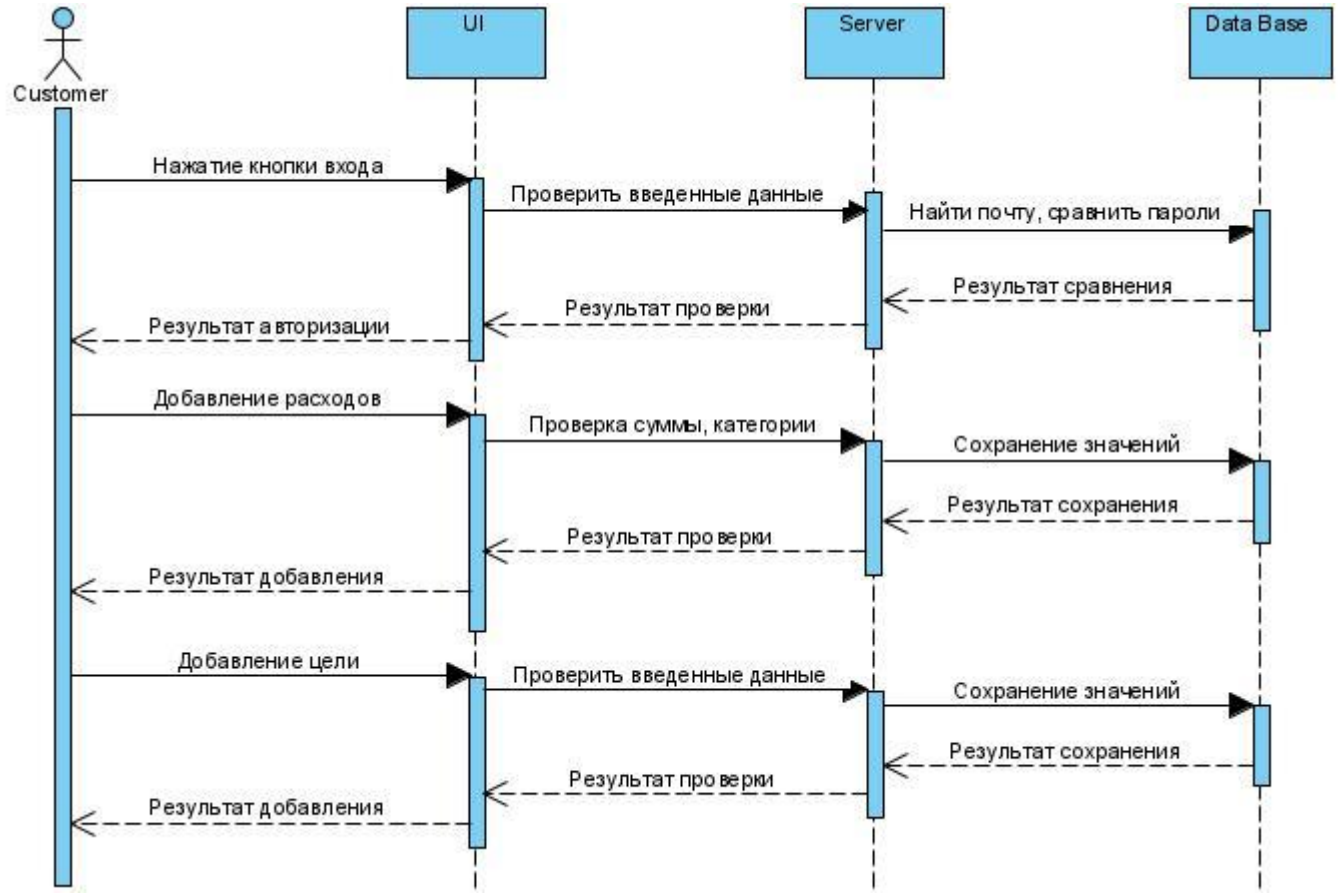

Рисунок 3 – Sequence diagram 1

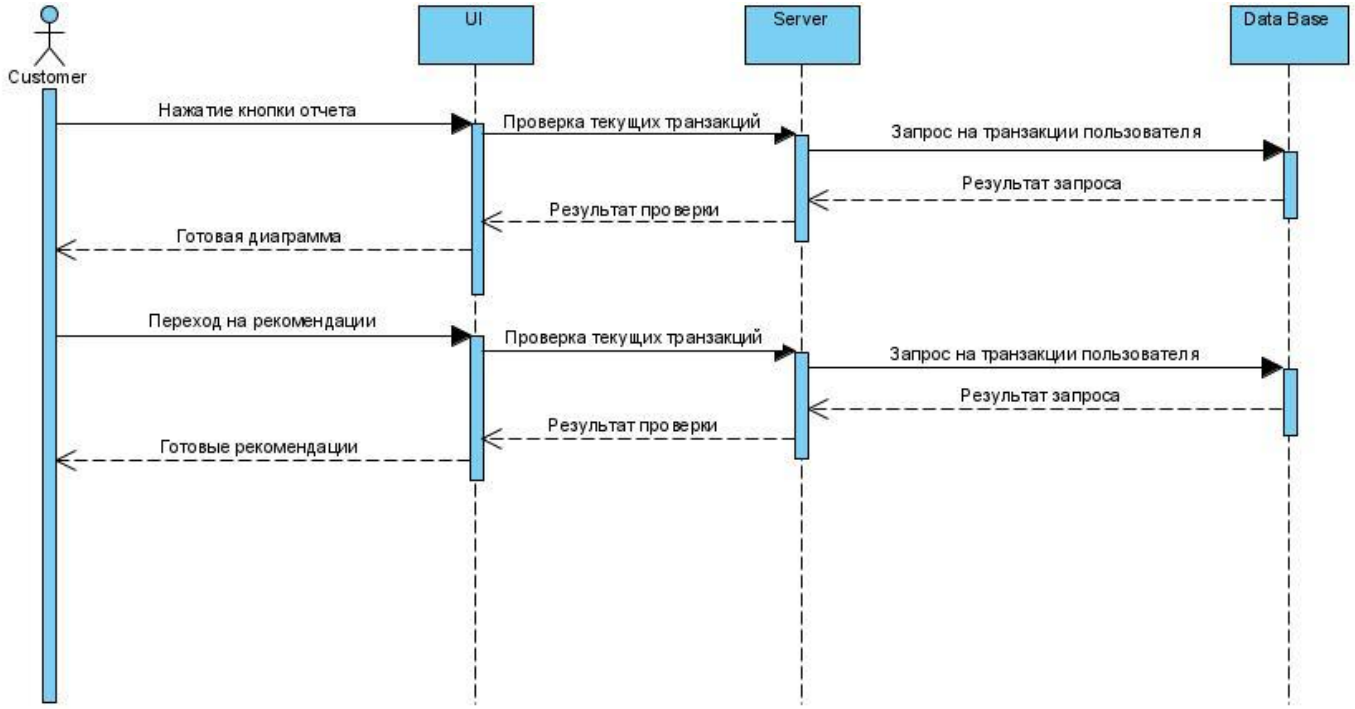

Рисунок 4 – Sequence diagram 2

На приведенном рисунке продемонстрирована диаграмма последовательностей и взаимодействий, на которых показаны отношения между объектами при входе в систему и при создании семейного профиля.

Пользователь использует графический интерфейс, чтобы войти в систему. Интерфейс в свою очередь передает запрос на добавление серверу, а сервер впоследствии проверяет и сравнивает данные введенные пользователем и данные в базе данных. База данных возвращает результат запроса, и в случае, если запрос прошел неудачно, сервер передает сообщение об ошибке графическому интерфейсу, а тот в свою очередь отображает сообщение об ошибке.

2.3.4 State diagram

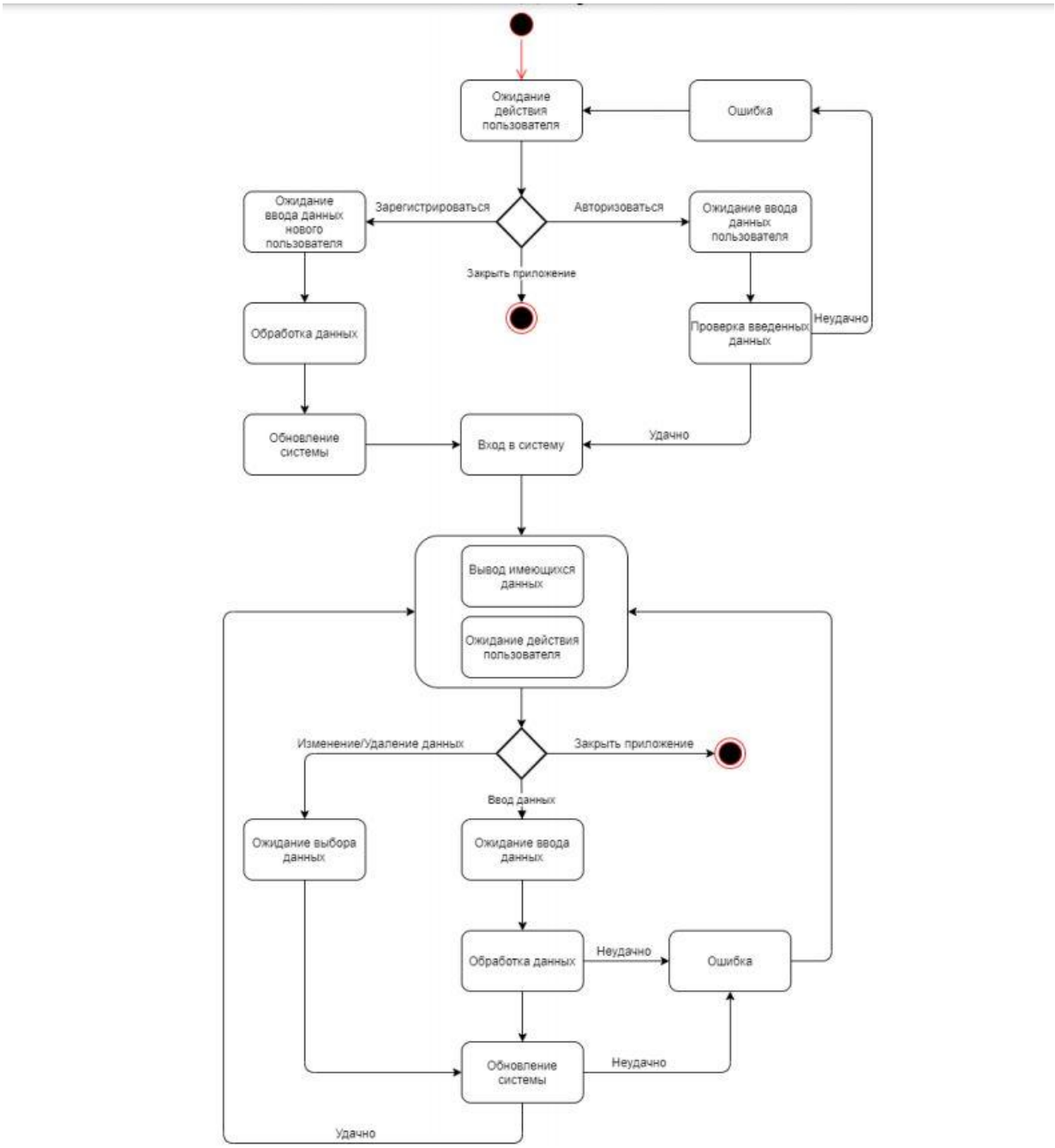

Рисунок 5 – State diagram

Данная диаграмма состояний отражает все возможные состояния системы, в которых она может находиться. При входе в приложение, система находится в

состоянии ожидания действий пользователя. Из этого состояния она может перейти в одно из следующих в зависимости от конкретных действий пользователя:

— Зарегистрироваться.

— Авторизоваться.

— Закрыть приложение.

В случае, если пользователь принял решение зарегистрироваться, система переходит в состояние ожидания ввода данных нового пользователя, затем введенные пользователем данные обрабатываются, то есть система находится в состоянии обработки данных, после чего она (система) обновляется и в последствии происходит вход в систему.

При авторизации пользователя система также входит в состояние ожидания ввода данных пользователя, а затем в состояние проверки введенных данных и, если введенные данные корректны, происходит вход в систему. В противном случае система входит в состояние ошибки и возвращается в начальное состояние ожиданий действий пользователя как при входе в систему. Если пользователь закроет приложение, произойдет выход из системы, то есть из самого приложения.

После входа в систему, приложение переходит в состояние ожидания действия пользователя, при котором у него (пользователя) есть следующие возможности:

— Изменение/Удаление данных

— Ввод данных

— Закрыть приложение

При изменении или удалении данных, система переходит в состояние ожидания выбора тех существующих в базе данных, которые хочет изменить или удалить пользователь приложения. После чего система входит в состояние обновления данных. Если данные обновились успешно, система переходит в состояние ожидания тех же действий пользователя, как и после входа в систему, иначе — произойдет ошибка. Если пользователь выберет ввод новых данных,

система будет ожидать ввода данных, которые впоследствии нужно будет обработать. В случае удачной обработки введенной пользователем информации, произойдет обновление системы, однако, если полученные данные были некорректны, приложение выдаст ошибку и предложит пользователю заново их ввести. При успешном обновлении системы, приложение будет ожидать дальнейших действий пользователя, в противном случае произойдет ошибка, информация не сохранится и пользователю приложения будет необходимо проделать действия заново.

2.3.5 Activity diagram

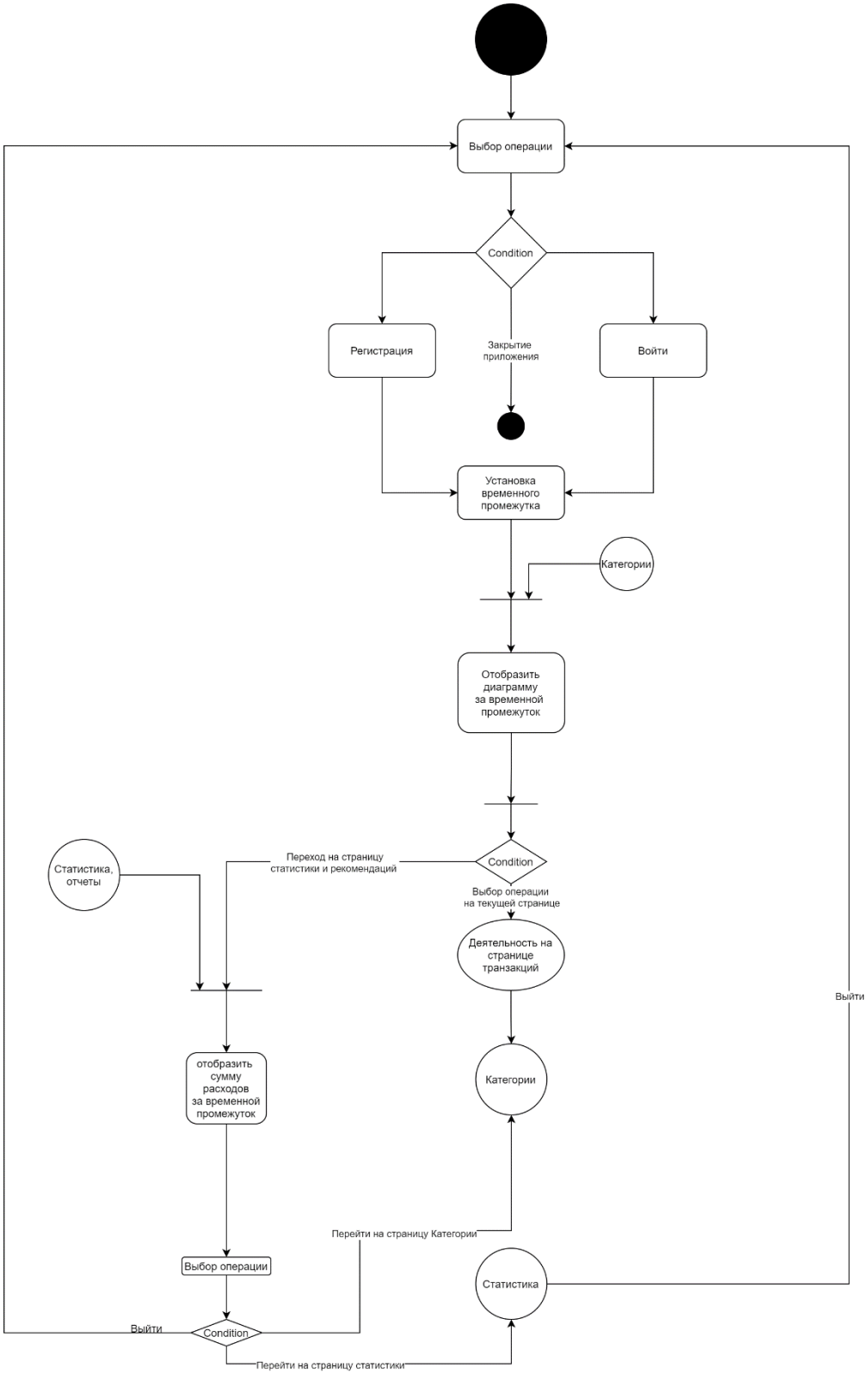

Рисунок 6 – Activity diagram

2.3.6 Deploy diagram

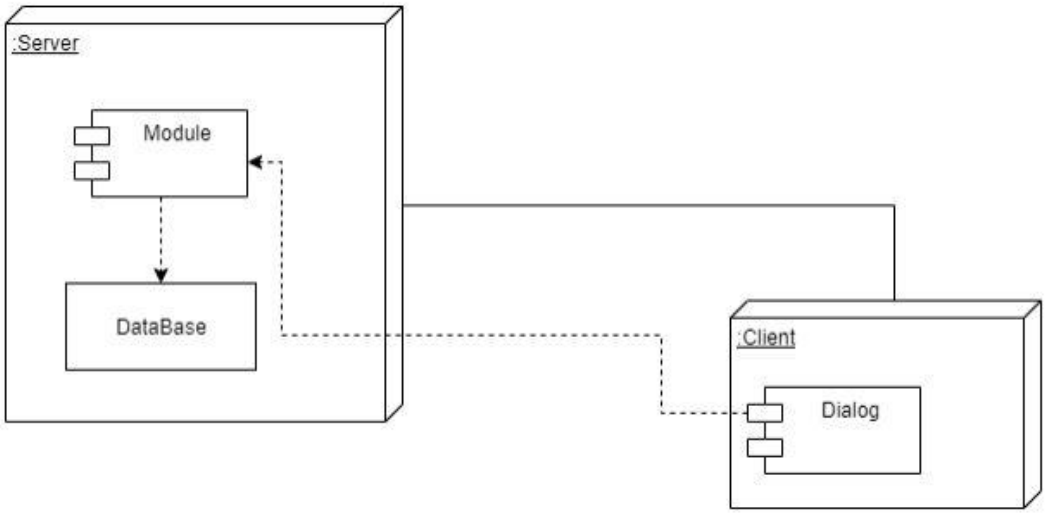

Рисунок 7 – Deploy diagram

Представленная нами диаграмма развертывания служит для демонстрации существующих аппаратных и программных компонентов приложения и взаимодействия между ними. На стороне сервера реализован весь необходимый функционал для успешной работы разрабатываемой системы, который неразрывно связан с базой данных, расположенной здесь же на сервере. На клиентской стороне приложения находится диалоговое окно, которое необходимо для обработки всех действий пользователя и передачи введенных или выбранных данных.

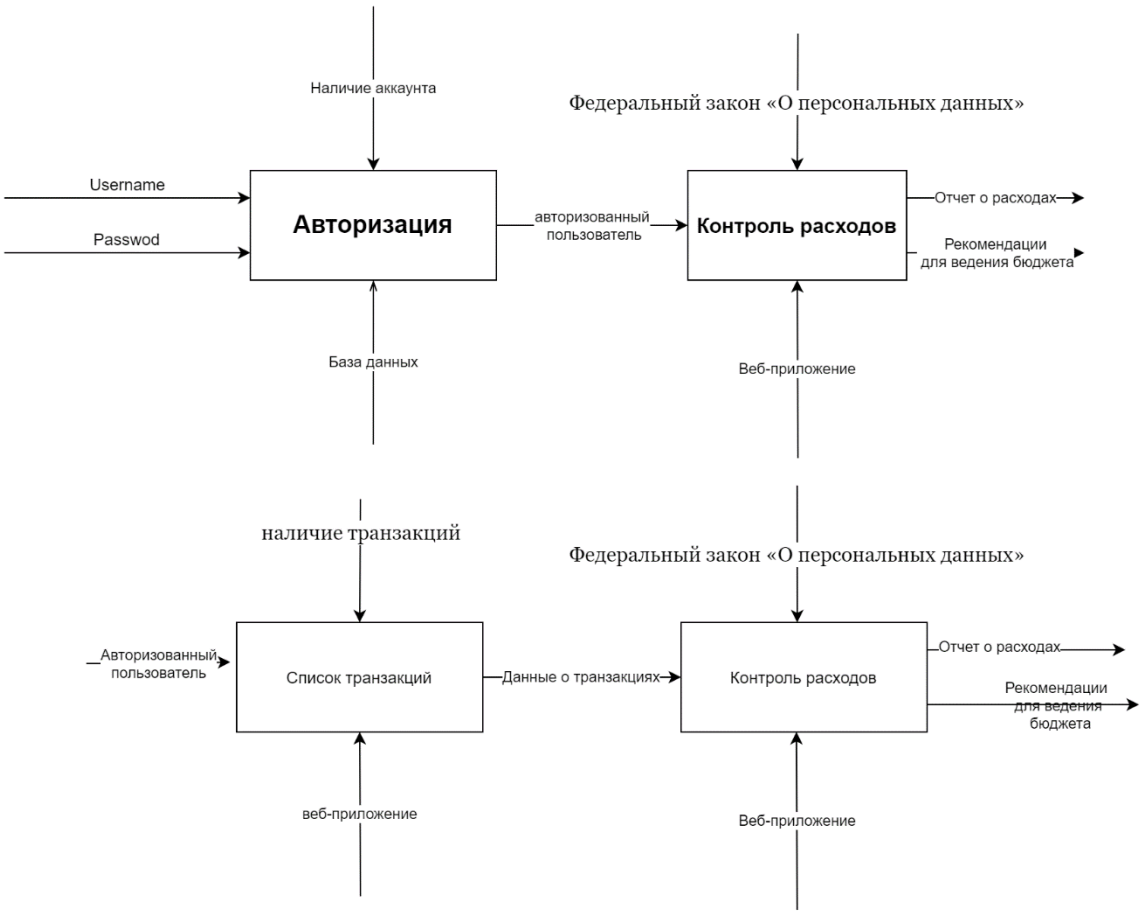

Рисунок 8 – IDEF0

2.4 Анализ средств реализации

Для реализации и разработки приложения Budger были выбраны следующие технологии:

Java в качестве основного языка программирования вследствие следующих причин:

Все члены команды имеют опыт работы с языком программирования

Подходит для разработки веб приложений.

JavaScript как встраиваемый язык для программного доступа к объектам приложений. Нашел применение в приложении как язык для придания интерактивности веб-страницам

HTML как стандартизированный язык текстовой разметки

CSS в качестве формального языка описания внешнего вида веб-приложения

В качестве СУБД был выбран PostgreSQL по нескольким причинам:

— Масштабируемость.

— Многофункционально

— Простота в использовании

Взаимодействие между Front-end и Back-end осуществляется за счет REST API

Обмен данными между Front-end и Back-end происходит с помощью передачи JSON файлов.

3 Реализация

3.1 Серверная часть приложения

Серверная часть приложения реализована с использованием языка Java, а также базы данных PostgreSQL. Серверная часть приложения предоставляет клиентской части доступ к набору методов, необходимые для реализации основных клиентских сценариев. Данный набор методов называется API. Организация взаимодействия между клиентом и сервером с использованием данного подхода позволяет иметь возможность доступа к набору методов данного приложения из других систем.

3.2 Графический интерфейс

Клиентская часть представляет собой SPA-приложение, реализованное с помощью javascript библиотеки React. Для разметки использовалось расширение javascript – jsx. Стили реализованы с помощью языка css.

3.2.1 Главная страница

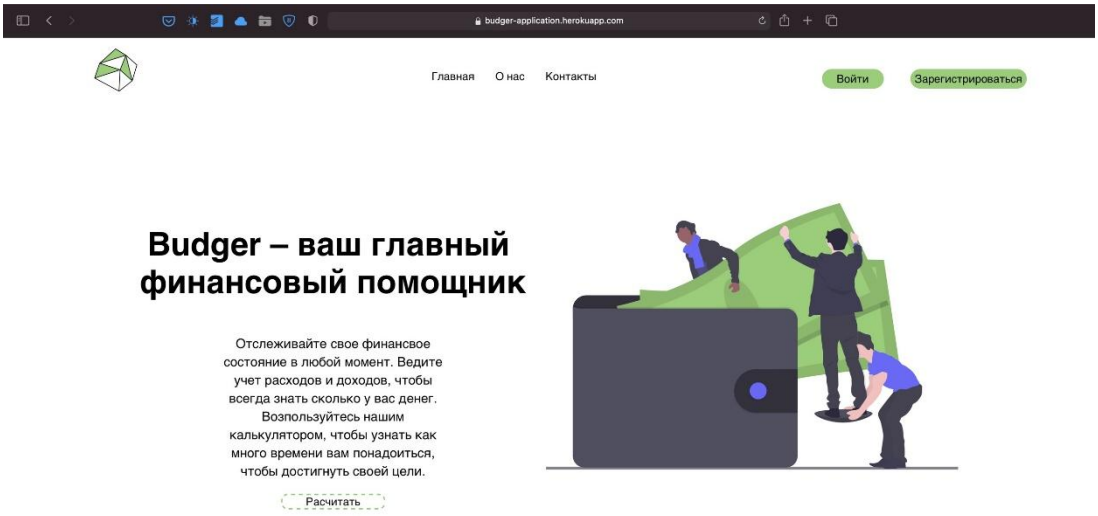

Рисунок 9 – Главная страница

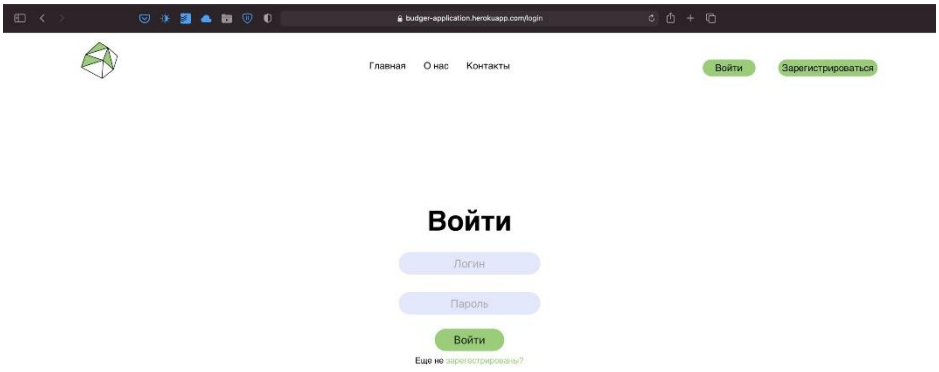

Рисунок 10 – Страница логина

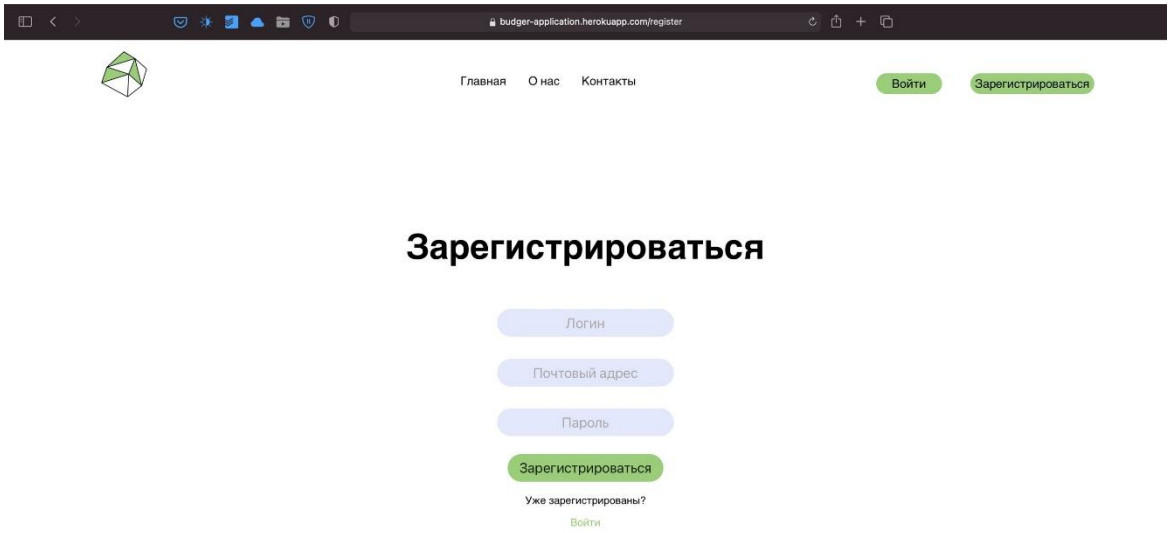

Рисунок 11 – Страница регистрации

Для регистрации необходимо ввести имя пользователя, почту и пароль. Если пользователя с таким именем и почтой в системе нет, то будет создан новый.

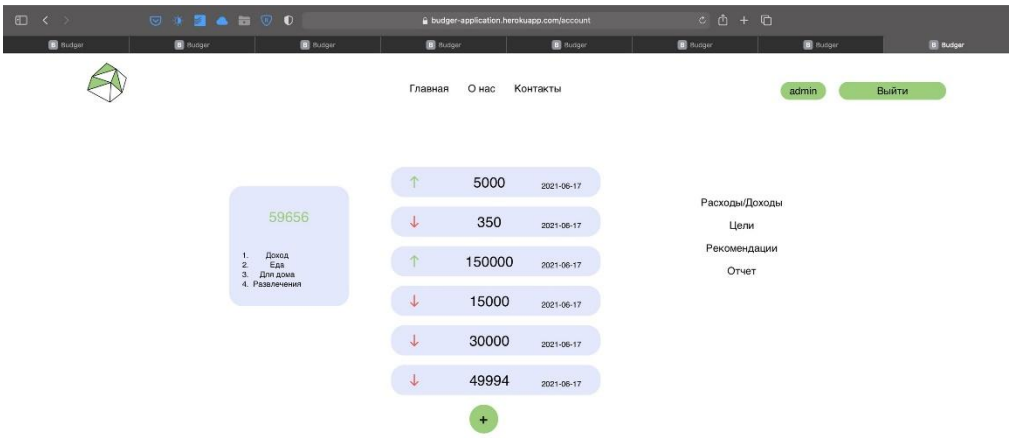

Рисунок 12 – Страница транзакций

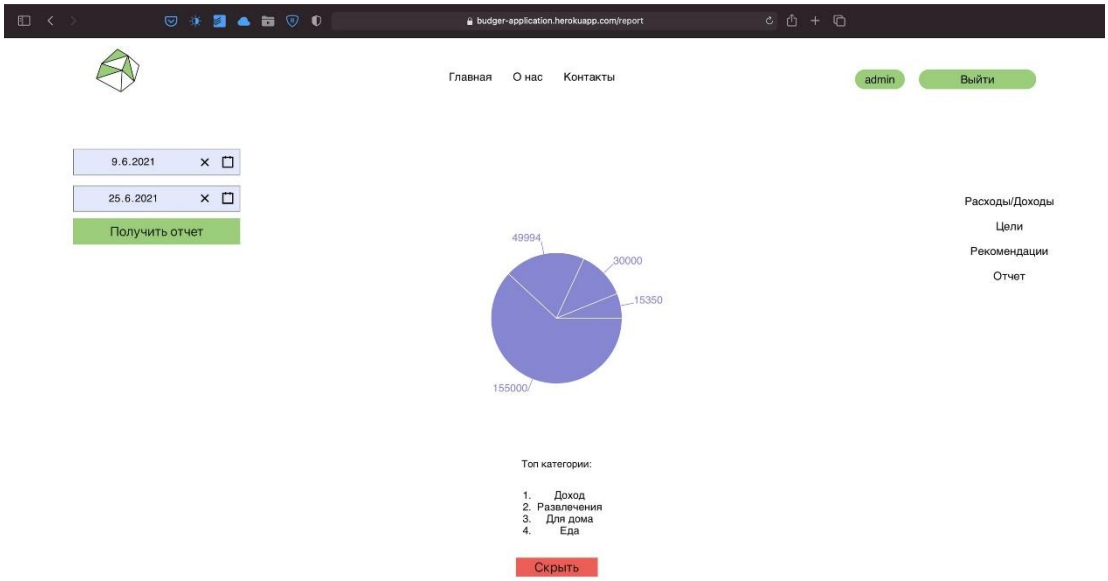

Рисунок 13 – Страница отчета

4 Тестирование

Во процессе разработки приложения были созданы тестовые процедуры. По завершении разработки приложения, было проведено тестирование по работе всего функционала веб-приложения Budger для проверки соответствия ожидаемого результата работы, указанного в техническом задании, с фактическим. Были выбран следующие процедуры ручного тестирования:

— System test

— Smoke test

4.1 Smoke testing

Smoke testing представляет собой короткий цикл тестов, выполняемых для подтверждения того, что приложение выполняет основные функции. Данная процедура была использована при каждой итерации деплоя приложения на хостинг.

#	Test	Verdict
	Registration	
1	GUI (layout, colors, static text, images)	passed
	Events	
2	"Зарегистрироваться" button click	passed
	Fields	
	Логин	
3	valid value >MIN & <MAX	passed
4	empty value	passed
	Почтовый адрес	
5	unique check	passed
6	email format	passed
7	not email format	passed
	Пароль	
8	empty value	passed
9	valid value	passed
	Login	
10	GUI (layout, colors, static text, images)	passed
	Events	
11	"Войти" button click	passed
	Fields	
	Логин	
12	correct login	passed
13	incorrect login	passed
	Пароль	
14	asterisks	passed
15	correct password	passed
16	incorrect password	passed
	Transaction create	
17	GUI (layout, colors, static text, images)	passed
	Events	
18	"Create" button click	passed
	Fields	
	Income switch	
19	changing type	passed
	Amount	
20	valid value	passed
21	empty value	passed
	Date	
22	calendar apperience	passed
23	empty value	passed
	Category	
24	valid value	passed
	Transactions view	
25	GUI (layout, colors, static text, images)	passed
	Events	
26	add transation button	passed
	Fields	
	Transactions list	
27	empty list	passed
	Date bounds	
28	upper bound calendar apperience	passed
29	bottom bound calendar apperience	passed
	Helper	
30	sum	passed
31	non empty category list	passed
32	non empty diagram	passed
	Goals view	
33	GUI (layout, colors, static text, images)	passed
	Events	
34	add goal button	passed
	Fields	
	Goals list	
35	empty list	passed
36	non empty list	passed
	Goals create	
37	GUI (layout, colors, static text, images)	passed
	Events	
38	add goal button	passed
	Fields	
	Name	
39	MIN<length<MAX	passed
	Amount	
40	MIN< value <MAX	passed
41	empty value	passed
	Date	
42	calendar apperience	passed
	Recommendations	
43	GUI (layout, colors, static text, images)	passed
	Fields	
	Category list	
44	MIN elements	passed
45	MAX elements	passed
	Report	
46	GUI (layout, colors, static text, images)	passed
	Events	
47	show button	passed
	Fields	
	Date	
48	calendar apperience	passed
	Diagram	
49	MIN<elements<MAX	passed

Рисунок 14 – Smoke testing

4.2 System testing

System testing использовалась при первом деплое, чтобы подтвердить работоспособность всех функций приложения.

#	Test	Priority	Verdict
	Registration		
1	GUI (layout, colors, static text, images)	High	passed
	Events		
1	"Зарегистрироваться" button click	High	passed
3	"Войти" link click	Medium	passed
4	Browser's back button	Low	passed
	Fields		
	Логин		
5	placeholder	High	passed
6	valid value		summary
7	MIN length	Medium	passed
8	MIN<length<MAX	High	passed
9	MAX length	Low	passed
10	special characters	Low	passed
11	unicode characters	Low	passed
12	invalid value		summary
13	empty value	High	passed
14	length<MIN	High	passed
15	length>MAX	Low	passed
	Почтовый адрес		
16	placeholder	High	passed
17	unique check	High	passed
18	valid value		summary
19	email format	High	passed
20	invalid value		summary
21	not email format	High	passed
	Пароль		
22	placeholder	High	passed
23	asterisks	High	passed
24	valid value		summary
25	MIN length	Medium	passed
26	MAX length	Medium	passed
27	MIN<length<MAX	High	passed
28	invalid value		summary
29	length<MIN	High	passed
30	length>MAX	High	passed
	Login		
1	GUI (layout, colors, static text, images)	High	passed
	Events		
1	"Войти" button click	High	passed
3	"Зарегистрироваться" link click	Medium	passed
4	Browser's back button	Low	passed
	Fields		
	Логин		
5	placeholder	High	passed
6	valid value		summary
7	MIN length	Medium	passed
8	MIN<length<MAX	High	passed
9	MAX length	Low	passed
10	special characters	Low	passed
11	unicode characters	Low	passed
12	invalid value		summary
13	empty value	High	passed
14	length<MIN	High	passed
15	length>MAX	Low	passed
	Пароль		
16	placeholder	High	passed
17	asterisks	High	passed
18	valid value		summary
19	MIN length	Medium	passed
20	MAX length	Medium	passed
21	MIN<length<MAX	High	passed
22	invalid value		summary
23	length<MIN	High	passed
24	length>MAX	High	passed
	Transaction create		
25	GUI (layout, colors, static text, images)	High	passed
	Events		
26	"Create" button click	High	passed
27	Browser's back button	Low	passed
	Fields		
	Amount		
28	valid value		summary
29	MIN value	Medium	passed
30	MAX value	Medium	passed
31	MIN< value <MAX	High	passed
32	invalid value		summary
33	value <MIN	High	passed
34	value >MAX	Medium	passed
35	value exluding numbers and point	High	passed
36	empty value	High	passed
	Date		
36	calendar apperience	High	passed
37	empty value	High	passed
	Category		
38	dropdown scroll	High	passed
	Transactions view		
39	GUI (layout, colors, static text, images)	High	passed
	Events		
40	add transation button	High	passed
41	goals button	High	passed
42	recommendations button	High	passed
43	report button	High	passed
	Fields		
	Transactions list		
44	list		summary
45	empty list	High	passed
46	list < screen	High	passed
47	list = screen	Medium	passed
48	list > screen	Medium	passed
	Helper		
49	sum	High	passed
50	category list		summary
51	empty list(no transactions)	High	passed
52	non empty list	High	passed
52	list < MAX	High	passed
53	list = MAX	Medium	passed
54	list > MAX	Low	passed
	Goals view		
55	GUI (layout, colors, static text, images)	High	passed
	Events		
56	add goal button	High	passed
57	transactions button	High	passed
58	recommendations button	High	passed
59	report button	High	passed
	Fields		
	Goals list		
60	list		summary
61	empty list	High	passed
62	list < screen	High	passed
63	list = screen	Medium	passed
64	list > screen	Medium	passed
	Goals create		
65	GUI (layout, colors, static text, images)	High	passed
	Events		
66	add goal button	High	passed
	Fields		
	Name		
67	valid value		summary
68	MIN length	Medium	passed
69	MIN<length<MAX	High	passed
70	MAX length	Low	passed
71	special characters	Low	passed
72	unicode characters	Low	passed
73	invalid value		summary
74	length<MIN	High	passed
75	length>MAX	Low	passed
	Amount		
76	valid value		summary
77	MIN value	Medium	passed
78	MAX value	Medium	passed
79	MIN< value <MAX	High	passed
80	invalid value		summary
81	value <MIN	High	passed
82	value >MAX	Medium	passed
83	empty value	High	passed
	Date		
84	calendar apperience	High	passed
85	empty value	High	passed
	Description		
86	valid value		summary
87	MIN length	Medium	passed
88	MIN<length<MAX	High	passed
89	MAX length	Low	passed
90	special characters	Low	passed
91	unicode characters	Low	passed
92	invalid value		summary
93	length>MAX	Low	passed
	Recommendations		
94	GUI (layout, colors, static text, images)	High	passed
	Events		
95	transactions button	High	passed
96	goals button	High	passed
97	report button	High	passed
	Fields		
	Category list		
98	MIN elements	High	passed
99	MAX elements	High	passed
	Report		
100	GUI (layout, colors, static text, images)	High	passed
	Events		
101	show button	High	passed
102	report button	High	passed
103	transactions button	High	passed
104	goals button	High	passed
105	report button	High	passed
	Fields		
	Date		
106	calendar apperience	High	passed
107	empty value	High	passed
	Diagram		
108	MIN elements	High	passed
109	MAX elements	High	passed

Рисунок 15 – System testing

5 Аналитика

Для аналитики приложения использовался сервис Яндекс.Метрика. С его помощью можно отслеживать активность пользователь за разные промежутки времени, посещаемость определенных роутов и др.

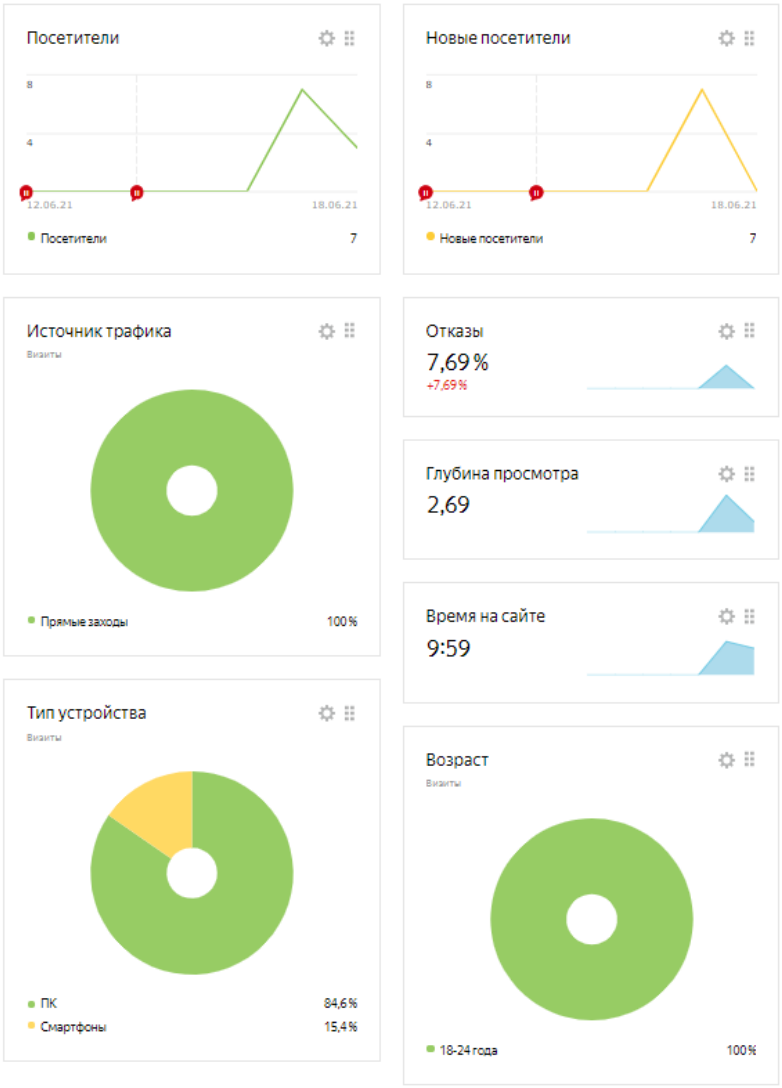

Рисунок 16 – Сводка Яндекс.Метрики

Так же средствами данного сервиса были созданы три основных пользовательских сценария, чтобы определить насколько пользователи заинтересованы в тех или иных функциях.

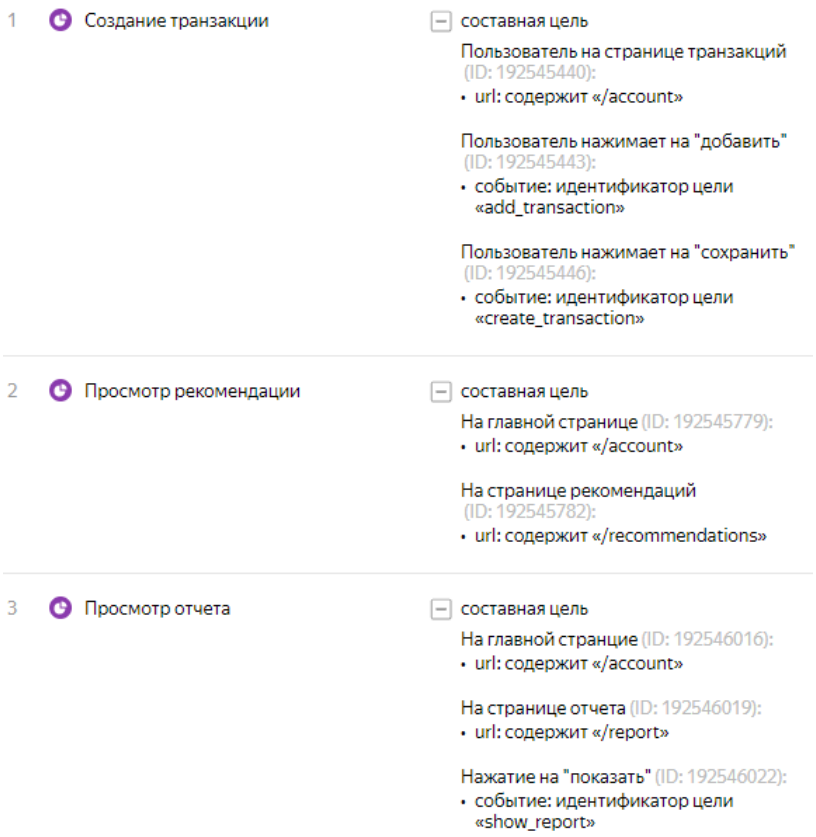

Рисунок 17 – 3 воронкки коверсии

Заключение

В результате проделанной работы было реализовано приложение Budger, которое позволяет пользователю осуществлять регистрацию и систематизацию своих расходов, а также просматривать различную статистику за определенный временной промежуток. Веб-приложение Budger обладает визуальным интерактивным интерфейсом, который необходим для взаимодействия пользователя с приложением и позволяет выводить на экран всю нужную информацию для удобства работы пользователя с сервисом. Были выполнены следующие задачи:

- Разработана база данных для хранения информации о пользователе, его расходах, доходах, расположенная на удаленном сервере.
- Разработан Back-end приложения необходимы для работы всего функционала сервиса развернутый на удаленном сервере приложения
- Разработан Front-end приложения, который необходим для взаимодействия пользователя с сервисом приложения
- С помощью API была реализована связь между Back-end и Front-end приложения