**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«СЕВЕРО-ОСЕТИНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ КОСТА ЛЕВАНОВИЧА ХЕТАГУРОВА»**

**Факультет математики и компьютерных наук**

**Кафедра прикладной математики и информатики**

**Направление подготовки «Прикладная математика и информатика»**

**КУРСОВАЯ РАБОТА**

**Биллинговая система для интернет провайдера**

Выполнил

студент 2 курса

направления «Прикладная математика и информатика»

**Габараев Андрей Валентинович**

**Научный руководитель**

**Гудиев Тимур Владимирович**

**«Работа допущена к защите»**

**Заведующий кафедрой**

**к. ф-м. н., доцент**

**Басаева Е.К. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Владикавказ 2023

**Оглавление**

Введение

Глава 1. Теоретическая Часть

Глава 2. Практическая Часть

Руководство пользователя

Руководство системного администратора

Руководство программиста

Глава 3. Список использованных инструментов и литературы

**Введение**

Биллинговая система, также известная как автоматизированная система расчетов (АСР), это программное обеспечение, которое позволяет определять стоимость услуг для каждого клиента. Она играет важную роль в различных сферах бизнеса, особенно в сфере телекоммуникаций.

Прежде всего, биллинг представляет собой процесс расчетов, информационного и финансового обслуживания.

В биллинговую систему вводится информация, на основе которой оператор связи выставляет счет абоненту за использование услуг связи за определенный период (обычно это один календарный месяц). Эта информация включает тарифы, стоимость минуты звонка в зависимости от направления (междугородние и международные звонки), стоимость интернет-трафика, стоимость подключенных телевизионных каналов и другие услуги. Биллинг также должен учитывать индивидуальные и общие скидки, акции, временное приостановление услуг по желанию абонента, налоговые начисления, дополнительные платежи и другие факторы, которые влияют на бизнес оператора связи. В биллинг также вносятся первичные данные о тарификации, включающие информацию о всех звонках, сделанных абонентами этого оператора связи, и объеме интернет-трафика. Основываясь на этой информации, биллинговая система рассчитывает стоимость услуг за определенный период для каждого клиента и создает платежные документы.

Биллинговая система должна обладать удобным интерфейсом для ввода и отслеживания всей необходимой информации, такой как тарифы, тарифные планы, пакеты услуг, опции, скидки, стоимость звонков в зависимости от направления и так далее. Она также должна иметь возможность создавать отчеты и взаимодействовать с абонентами, отправлять документы, управлять уведомлениями и т.д.

Автоматизированная система расчетов должна не только корректно выставлять счета абонентам, но и обрабатывать поступающие платежи, связывать их с выставленными счетами, управлять задолженностями абонентов и, при необходимости, взаимодействовать с бухгалтерскими программами. Она выполняет функции финансового учета и обработки платежей, обеспечивая точность и своевременность финансовых операций.

Биллинговая система обеспечивает комплексное информационное и финансовое обслуживание, обеспечивая эффективную работу операторов связи. Она позволяет точно определять стоимость услуг для каждого клиента, обрабатывать платежи и контролировать задолженности, а также предоставлять необходимую информацию для управленческого анализа и принятия решений.

Важно отметить, что биллинговая система является неотъемлемой частью операционной деятельности операторов связи, обеспечивая прозрачность и эффективность процессов расчетов и финансового учета.

**Глава 1: Теоретическая часть**

Моей целью было создать биллинговую систему для интернет-провайдера, которая поможет управлять и обрабатывать платежи и счета клиентов.

Провайдерам предлагается функциональность для создания счетов, отслеживания оплаты и управления услугами, предоставляемыми клиентам. Основная цель - автоматизация процесса биллинга, уменьшение ошибок и обеспечение удобного и эффективного взаимодействия с клиентами.

В системе реализованы базовые функции, такие как генерация платежа на основе использования интернет-услуг, получение истории оплат, отслеживание задолженностей, управление тарифными планами и предоставление детализированной информации о расходах клиентов. Система в дальнейшем также будет предоставлять аналитические отчеты и возможность создания пользовательских настроек.

Эта система является важным инструментом для интернет-провайдера, так как она облегчает учет и управление финансовыми операциями, связанными с предоставлением интернет-услуг. Она также помогает провайдерам улучшить качество обслуживания клиентов и повысить эффективность своей деятельности.

Итак, определившись с системой, передо мной стоял выбор. Надо было определиться в виде чего ее создать: веб-приложение, десктопное приложение или мобильное приложение. Отталкиваясь от моих знаний и опыта в разработке программного обеспечения, я решил создать веб-приложение с использованием современных веб-технологий.

Веб-приложение предоставляет ряд преимуществ, таких как доступность из любого устройства с доступом в Интернет, легкая масштабируемость и возможность обновления системы без необходимости установки обновлений на каждом клиентском компьютере.

Я выбрал веб-приложение, чтобы обеспечить простоту использования и удобство для клиентов и персонала интернет-провайдера. Пользователи смогут легко получить доступ к своим счетам, проверить свои платежи, изменить свои тарифные планы и получить детализированную информацию о своем использовании интернет-услуг.

Для разработки веб-приложения я выбрал современные веб-технологии, такие как HTML, CSS и JavaScript

С помощью веб-приложения интернет-провайдер сможет эффективно управлять своими клиентскими счетами, улучшить процесс биллинга и повысить удовлетворенность клиентов. Оно также обеспечит надежную и безопасную обработку платежей, а также защиту конфиденциальности данных клиентов.

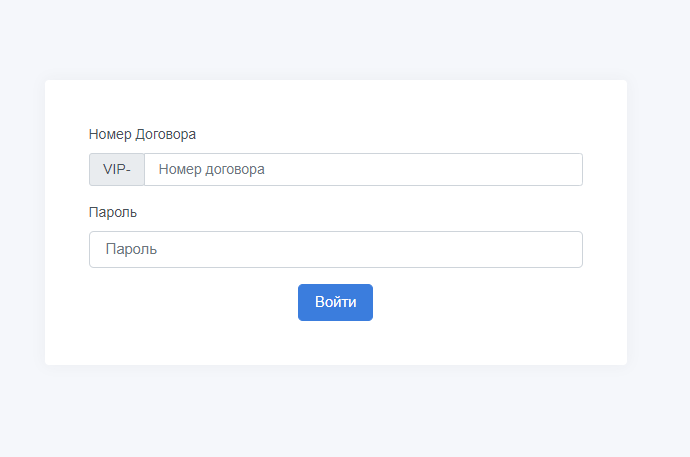
Таким образом, создание веб-приложения на основе современных веб-технологий позволит интернет-провайдеру эффективно управлять своими финансовыми операциями и обеспечить качественное обслуживание своих клиентам.

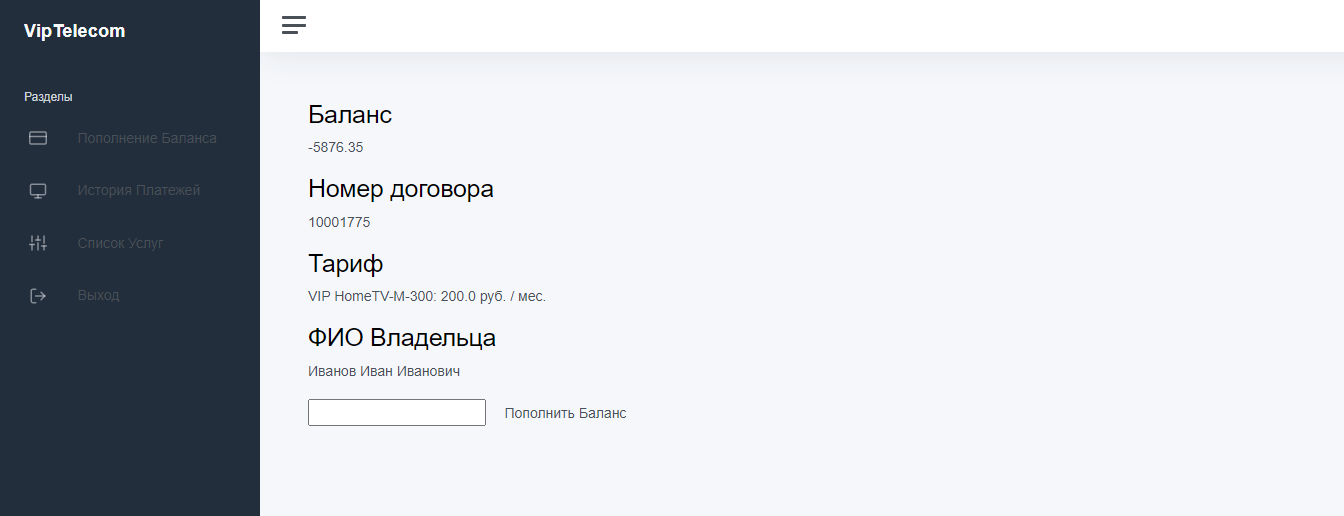
.

**Глава 2. Практическая часть**

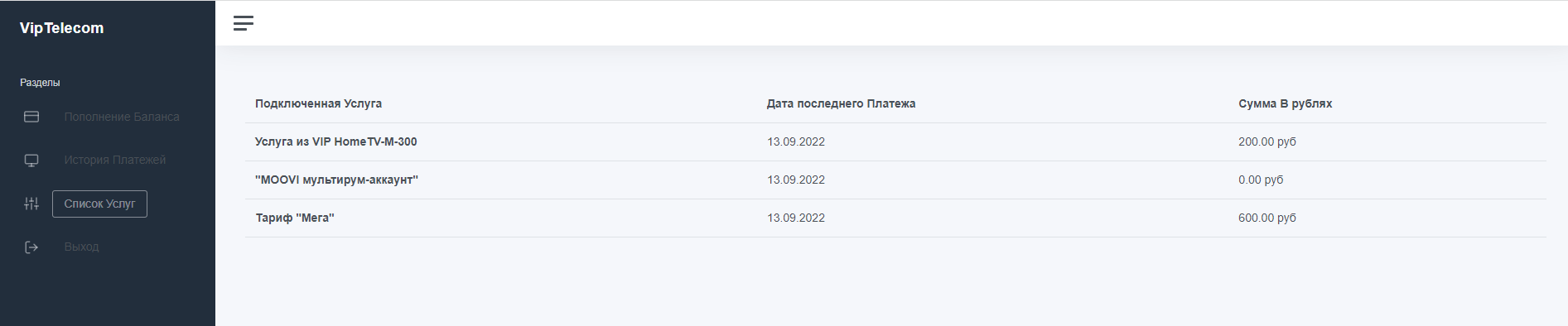
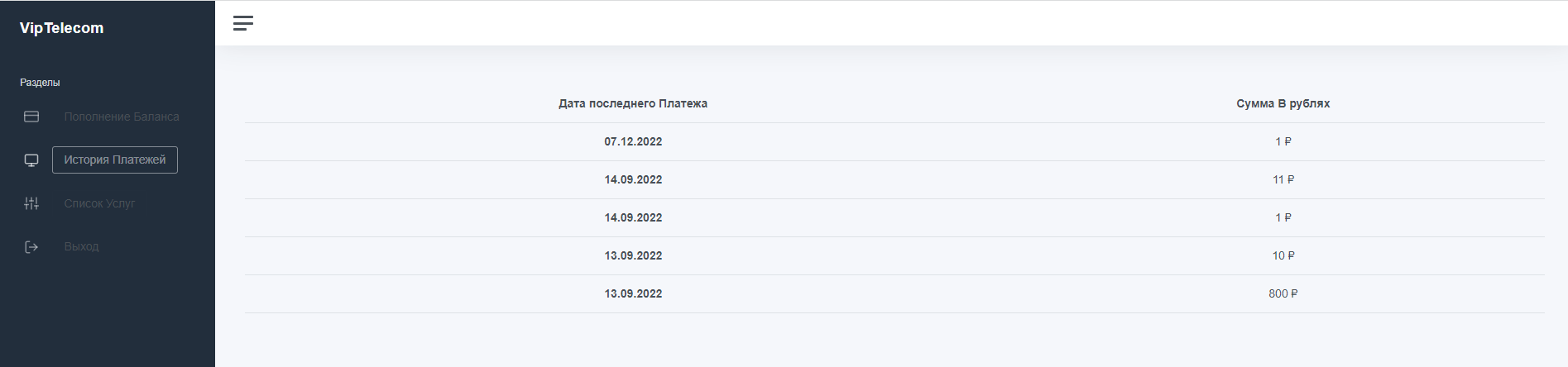
**Руководство пользователя**

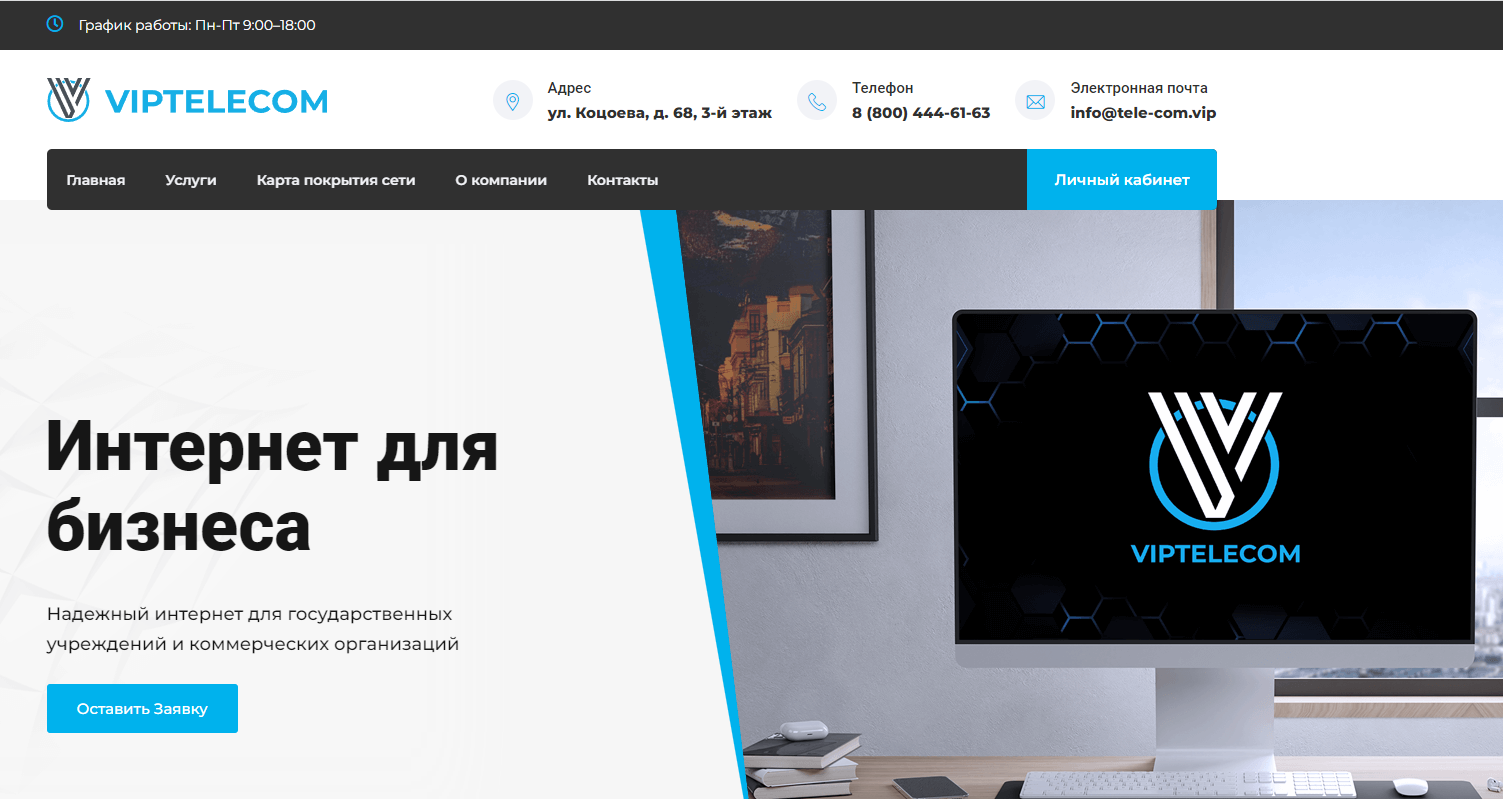
При открытии приложения вас встречает данное окно авторизации, где нужно ввести актуальный логин и пароль, далее после верификации происходит редирект на главную страницу, с пользовательскими данными и возможностью пополнения баланса.

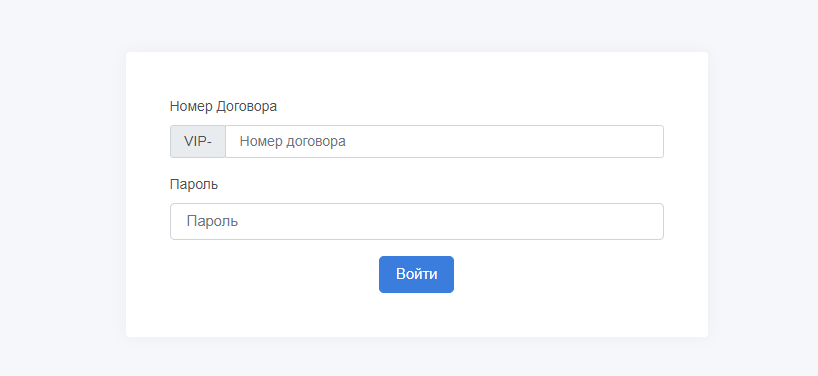




Для пополнения баланса требуется ввести желаемую сумму и нажать на кнопку “Пополнить Баланс”, после чего следует редирект на сервис эквайринга.  
 Можно также просмотреть историю платежей и список подключенных услуг, кликнув на “История платежей” и “Список услуг” соответственно.



Имеется возможность перейти на официальный сайт интернет провайдера по клику на “VipTelecom” сверху страницы и выйти из приложения и вернуться к полю авторизации по клику на “Выход” внизу страницы.



**Руководство системного администратора**

Минимальные требования к программным и техническим средствам для работы программы:

1. .NET Framework 4.5
2. ОЗУ - 1 ГБ
3. свободного места на диске 128 МБ
4. ОС - Windows 7
5. Процессор - Intel Pentium 4

**Руководство программиста**

Вся разметка содержится в 2 компонентах:  
1)index.html(она же страница авторизации)  
2)main.html(страница с остальным функционалом)

Логика авторизации прописана в файле “login.js” который находится в папке “ИМЯПРОЕКТА/src/js”

Конкретно за проверку полей и ответ с бэкэнда отвечает код с c 9-46 строку

**if** (login.value == "" || password.value == "") {  
 alert("Пожалуйста заполните поля!");  
 } **else** {  
 **const** loginUrl = `http://45.84.68.38:8082/system\_api/?format=json&context=web&model=users&method1=web\_cabinet.login&arg1={"login":"VIP-${login.value}","passwd":"${password.value}"}`;  
 console.log(123, loginUrl);  
 fetch("https://tele-com.vip/carbon/api.php?req=" + btoa(loginUrl))  
 .then((response) => response.json())  
 .then((clientInfo) => {  
 **if** (!clientInfo.error) {  
 console.log("clientInfo", clientInfo);  
 **const** sessionID = clientInfo.session\_id;  
 **const** clientName = clientInfo.user.\_\_abonent;  
 **const** clientAccount = clientInfo.user.abonent.\_\_account;  
 **const** clientAccountSplit = clientAccount.split(" ");  
 **const** clientAccountNumber = clientAccountSplit[1];  
 **const** clientAccountDebt = clientAccountSplit[3];  
 **const** clientContractNumber = clientInfo.user.abonent.contract\_number;  
 **const** clientBalance = clientInfo.user.abonent.recomend\_pay\_sum;  
 **const** clientActivationDate = clientInfo.user.abonent.activate\_date;  
 **const** clientTariff = clientInfo.user.abonent.\_\_tarif;  
  
 **const** clientInfoObject = {  
 name: clientName,  
 account: clientAccountNumber,  
 contractNumber: clientContractNumber,  
 balance: clientAccountDebt,  
 activationDate: clientActivationDate,  
 tariff: clientTariff,  
 clientName: clientName,  
 };  
 console.log(clientInfo);  
 localStorage.setItem("sessionID", sessionID);  
 localStorage.setItem("clientInfo", JSON.stringify(clientInfoObject));  
 window.location.assign("main.html");  
 } **else** {  
 console.error(clientInfo.error);  
 }  
 });

Логика компонентов прописана в файле “main.js” который находится в папке “ИМЯПРОЕКТА/src/js”

Перерендер компонента происходит при клике на соответствующее поле на сайдбаре (“Пополнение Баланса”,”История Платежей ”,”Активные услуги”)

Логика под каждый компонент написана на 30-46,48-85,87-129,строках соответственно

| payBtn.addEventListener("click", **async** (e) => {  e.preventDefault();  **const** scoreInput = document.getElementById("score-input").value;   **if** (!scoreInput) {  alert("Введите сумму платежа");  } **else** {  **const** payScoreUrl = `http://45.84.68.38:8082/system\_api/?format=json&context=web&model=users&method1=web\_cabinet.add\_payment\_operation&arg1={"suid":\"${sessionId}\","src\_ip":"10.20.30.41", "contract\_number": "${scoreId}", "summa\_in": "${scoreInput}", "operator": "SBERBANK\_ACQ"}`;   **const** fetchData = **await** fetch(  "https://tele-com.vip/carbon/api.php?req=" + btoa(payScoreUrl)  ).then((response) => console.log(response));  *// .then((payScore) => {*  *// return payScore;*  *// });*  } }); |
| --- |

| payHistoryBtn.addEventListener("click", **async** () => {  allWrappers.forEach((element) => {  element.classList.remove("active");  });   allBtns.forEach((element) => {  element.classList.remove("active");  });   payHistoryWrapper.classList.add("active");  payHistoryBtn.classList.add("active");   **const** fetchData = **await** fetch(  "https://tele-com.vip/carbon/api.php?req=" + btoa(payHistoryUrl)  )  .then((response) => response.json())  .then((payHistory) => {  **return** payHistory;  });   **const** tableBody = document.getElementById("payhistory-table-body");   **const** myChildData = fetchData.items  .map((element) => {  **return** `  <tr>  <th class="text-center" scope="row">${**new** Date(  element.op\_date  ).toLocaleDateString("ru-RU")} </th>  <td class="text-center">${element.op\_summa} ₽</td>  </tr>  `;  })  .join("");    tableBody.innerHTML = myChildData; }); |
| --- |

| activeServicesBtn.addEventListener("click", **async** () => {  allWrappers.forEach((element) => {  element.classList.remove("active");  });   allBtns.forEach((element) => {  element.classList.remove("active");  });   activeServicesWrapper.classList.add("active");  activeServicesBtn.classList.add("active");   **const** fetchData = **await** fetch(  "https://tele-com.vip/carbon/api.php?req=" + btoa(servicesUrl)  )  .then((response) => response.json())  .then((serviceHistory) => {  console.log(serviceHistory);  **return** serviceHistory[1];  });   **const** tableBody = document.getElementById("services-table-body");   **const** myChildData = fetchData  .map((element) => {  **const** serviceName = element.\_\_self;  **const** serviceNameResult = serviceName.match(regexName)[0];  **const** serviceCost = regexCost.exec(element.\_\_self);  **const** serviceCostResult = serviceCost ? serviceCost[1] + "руб" : null;  **return** `  <tr>  <th scope="row">${serviceNameResult}</th>  <td>  ${**new** Date(element.create\_date).toLocaleDateString("ru-RU")}  </td>  <td>${serviceCostResult}</td>  </tr>  `;  })  .join("");   tableBody.innerHTML = myChildData; }); |
| --- |

Логика выхода из страницы реализована встроках 131-135

| logoutBtn.addEventListener("click", () => {  location.href = "login.html";  localStorage.clear(); }); |
| --- |

**Глава 3: Список источников и литературы**

1. <https://docs.carbonsoft.ru/display/CarbonBilling/Home>
2. <https://docs.carbonsoft.ru/pages/viewpage.action?pageId=187433098>
3. <https://developer.mozilla.org/ru/>