

Министерство образования и науки Российской Федерации  
Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б.Н. Ельцина» (УрФУ)

Институт фундаментального образования

Кафедра «Технической физики»

Оценка

Руководитель курсового

проектирования

Члены комиссии

Дата защиты

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по теме: Создание автоматизированной информационной системы личного кабинета для сайта компании «ЛУКОЙЛ»

по дисциплине: Алгоритмизация и программирование

Студент: Бузаньяров Эльдар Салаватович

(Подпись)

Группа: ФТ−140007

Екатеринбург

2024



Министерство образования и науки Российской Федерации  
Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б.Н. Ельцина» (УрФУ)

Институт фундаментального образования

Кафедра «Технической физики»

**Задание на курсовую работу**

Студент Бузаньяров Эльдар Салаватович

Группа ФТ−140007

специальность/направление подготовки 09.03.02 Информационные системы и технологии

1.Тема курсовой работы: Создание автоматизированной информационной системы личного кабинета для сайта компании «ЛУКОЙЛ»

2.Содержание работы, в том числе состав графических работ и расчетов: описание и реализация алгоритмов, блок−схемы алгоритмов, таблицы, листинг кода

3.Дополнительные сведения: работа выполнена на языке программирования python.

4. План выполнения курсовой работы

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование элементов  проектной работы | Сроки | Примечания | Отметка о выполнении |
| Изучение методов и библиотек | 4.09-10.10 |  |  |
| Знакомство с заказчиком, определение потребностей | 10.10 |  |  |
| Планирование | 10.10-10.11 |  |  |
| Реализация алгоритма | 10.11-6.12 |  |  |
| Предварительная проверка | 6.12-13.12 |  |  |
| Оформление пояснительной записки | 13.12-26.12 |  |  |
| Защита работы | 26.12-27.12 |  |  |

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/М.Н. Ятченко

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 3](#_Toc185960360)

[ВВЕДЕНИЕ 4](#_Toc185960361)

[**1 Методология создания алгоритма** 5](#_Toc185960362)

[**1.1 Сведения о заказчике** 5](#_Toc185960363)

[**1.2 Определение потребностей** 6](#_Toc185960364)

[**1.3 Описание алгоритмов** 7](#_Toc185960365)

[**1.3.1 Создание и обозначение блок-схем** 7](#_Toc185960366)

[**1.3.2 Страница навигации** 8](#_Toc185960367)

[**1.3.3 Личный кабинет неавторизованного пользователя** 9](#_Toc185960368)

[**1.3.4 Личный кабинет авторизованного пользователя** 10](#_Toc185960369)

[**1.3.5 Бонусы и скидки** 11](#_Toc185960370)

[**1.3.6 История транзакций** 12](#_Toc185960371)

[**1.3.7 Добавление транзакции** 13](#_Toc185960372)

[**1.4 Средства и методы создания системы** 14](#_Toc185960373)

[**1.4.1 Библиотеки** 14](#_Toc185960374)

[**1.4.2 Методы создания программным путем** 15](#_Toc185960375)

[**1.4.3. Реализация отдельных объектов** 15](#_Toc185960376)

[**1.4.4 Реализация смены темы** 19](#_Toc185960377)

[**1.4.5 Реализация регистрации и входа** 20](#_Toc185960378)

[**1.4.6 Реализация добавления транзакций** 21](#_Toc185960379)

[**2 Реализация алгоритма** 23](#_Toc185960380)

[**2.1 Состав информационной системы** 23](#_Toc185960381)

[**2.2 Страница навигации** 24](#_Toc185960382)

[**2.3 Личный кабинет неавторизованного пользователя** 26](#_Toc185960383)

[**2.4 Личный кабинет авторизованного пользователя** 27](#_Toc185960384)

[**2.5 Бонусы и акции** 27](#_Toc185960385)

[**2.6 История транзакций** 28](#_Toc185960386)

[**2.7 Добавление транзакции** 29](#_Toc185960387)

[ЗАКЛЮЧЕНИЕ 30](#_Toc185960388)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 31](#_Toc185960389)

[ПРИЛОЖЕНИЕ 32](#_Toc185960390)

# ВВЕДЕНИЕ

В современном цифровом мире наличие удобного и информативного веб−сайта является важным для любой крупной компании, особенно для такой, как ПАО «ЛУКОЙЛ», являющейся глобальным лидером в нефтегазовой отрасли. Эффективное онлайн−представительство обеспечивает улучшение взаимодействия с клиентами, партнерами и инвесторами, а также способствует укреплению имиджа компании. Разработка и постоянное обновление пользовательского интерфейса сайта – неотъемлемая часть стратегии успешного функционирования в условиях жесткой конкуренции и растущих требований пользователей к удобству и функциональности онлайн−ресурсов. Данная работа посвящена разработке и улучшению пользовательского интерфейса веб−сайта компании, что является актуальной задачей для повышения эффективности коммуникации и привлечения аудитории.

Целью данной работы является разработка прототипа интуитивно понятного интерфейса личного кабинета, обеспечивающего удобный доступ к истории транзакций и бонусам.

Для достижения поставленной цели необходимо решить следующие задачи:

− Изучить функционал сайта заказчика;

− Определить потребности заказчика;

− Изучить программные методы и средства для реализации системы;

− Разработать информационную систему;

− Реализовать информационную систему.

− Протестировать информационную систему

# **1 Методология создания алгоритма**

## **1.1 Сведения о заказчике**

ПАО «ЛУКОЙЛ» − одна из крупнейших публичных вертикально интегрированных нефтегазовых компаний в мире, на долю которой приходится около 2% мировой добычи нефти и около 1% доказанных запасов углеводородов.

Ежедневно продукцию, энергию и тепло компании покупают миллионы людей более чем в 100 странах мира, улучшая качество своей жизни. Более 100 тысяч человек объединяют свои усилия и таланты, чтобы обеспечить эффективное развитие компании и её передовые позиции на рынке.

Бизнес−модель компании основана на принципе эффективной вертикальной интеграции в целях создания добавленной стоимости и обеспечения высокой устойчивости нашего бизнеса путем диверсификации рисков.

В 2017 году Совет директоров утвердил Программу стратегического развития Группы «ЛУКОЙЛ» на 2018 – 2027 годы. Обновленная стратегия направлена на обеспечение устойчивого роста ключевых показателей и выполнение прогрессивной дивидендной политики при консервативном сценарии цены на нефть, а также на дополнительное развитие бизнеса и распределение средств акционерам в случае более благоприятной конъюнктуры.

Являясь одним из крупнейших пользователей природных ресурсов в мире, «ЛУКОЙЛ» неукоснительно следует высочайшим мировым экологическим стандартам и гордится выстроенной системой управления охраной окружающей среды по всей вертикали управления.

Следуя своему предназначению, ЛУКОЙЛ осуществляет деятельность на основе честности и справедливости, уважительности и порядочности. Понимая степень ответственности перед обществом за рациональное использование природных ресурсов и сохранение благоприятной экологической ситуации, ЛУКОЙЛ в своей работе руководствуется самыми высокими стандартами охраны окружающей среды и обеспечения промышленной безопасности. В своей деятельности компания руководствуется принципами устойчивого развития и старается достичь равновесия между социально−экономическим и природно−экологическим развитием.

В компании функционирует развитая система корпоративного управления, которая строится на основе международных стандартов корпоративного поведения и деловой этики, требований российского законодательства, требований Правил листинга Московской биржи, принципов Кодекса корпоративного управления, рекомендованного к применению Банком России [1].

На сайте компании «ЛУКОЙЛ» реализованы кнопки для перехода на соответствующие разделы, такие как навигация, информационные сообщения, маркетинговые материалы и контактная информация и др. Так же представлен логотип компании, баннеры. Динамические изменения цвета при наведении курсора на элементы навигации улучшают пользовательский опыт, обеспечивая визуальную обратную связь. Поле поиска позволяет пользователям быстро находить необходимую информацию на сайте. Выпадающий список языков обеспечивает выбор для носителей разных языков.

## **1.2 Определение потребностей**

В данной работе определение потребностей базируется на анализе существующего веб−ресурса и выявлении областей для улучшения пользовательского опыта. Система необходима заказчику для повышения эффективности взаимодействия с пользователями, улучшения их лояльности и оптимизации бизнес−процессов. Разработанный прототип призван решить следующие задачи:

Оптимизация бизнес−процессов. Система сбора данных об истории транзакций позволяет компании анализировать покупательское поведение, выявлять тренды, оценить эффективность различных акций и адаптировать свою маркетинговую стратегию, автоматизация учета бонусов и их начисления минимизирует ошибки и снизит нагрузку на персонал.

Повышение лояльности клиентов. Возможность отслеживания накопленных бонусов и истории покупок мотивирует клиентов совершать повторные покупки на АЗС ЛУКОЙЛ, используя бонусы и участвуя в акциях, а личный кабинет упрощает процесс участия в бонусной программе, позволяя клиентам легко отслеживать накопленные баллы, использовать их для оплаты товаров и услуг и получать информацию об условиях программы.

Система, представленная в данном проекте, вносит следующие новшества:

− Смена темы

− Возможность создания личного кабинета

− История транзакций

− Раздел бонусов и промокодов

В целом, создание данной системы отвечает стратегическим целям компании «ЛУКОЙЛ» по повышению лояльности клиентов и оптимизации бизнес−процессов.

## **1.3 Описание алгоритмов**

### **1.3.1 Создание и обозначение блок**−**схем**

В данной работе для описания отдельных процессов и алгоритмов взаимодействия с графическим интерфейсом будут использоваться блок−схемы, построенные с использованием приложения draw.io в соответствии с обозначениями, представленными в таблице 1:

Таблица 1 – Обозначение блок−схем

|  |  |
| --- | --- |
| Блок | Значение |
|  | Начало/конец алгоритма,  переход на следующую/предыдущую страницу |
|  | Процесс |
|  | Запоминаемые данные |
|  | Условие |
|  | Ввод/вывод данных |
|  | Показывают направление потока выполнения алгоритма |

### **1.3.2 Страница навигации**

На странице навигации пользователю предоставляется возможность произвести смену темы, пролистать баннеры и открыть личный кабинет. При смене темы меняется иконка смены темы и цветовая схема всех окон. При наведении на надписи с навигацией цвет шрифта меняется. Схема взаимодействия представлена на рисунке 1.

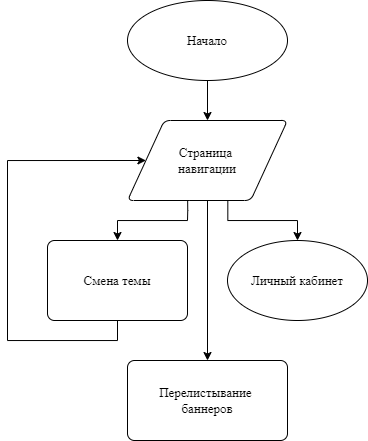


Рисунок 1 – Страница навигации

### **1.3.3 Личный кабинет неавторизованного пользователя**

В личном кабинете по умолчанию пользователь не авторизован, и пользователю предоставляется возможность зарегистрировать новый аккаунт, либо войти в уже существующий. При неправильном вводе данных пользователь оповещается о наличии ошибки. При успешной авторизации открывается интерфейс авторизованного пользователя. Схема взаимодействия представлена на рисунке 2.

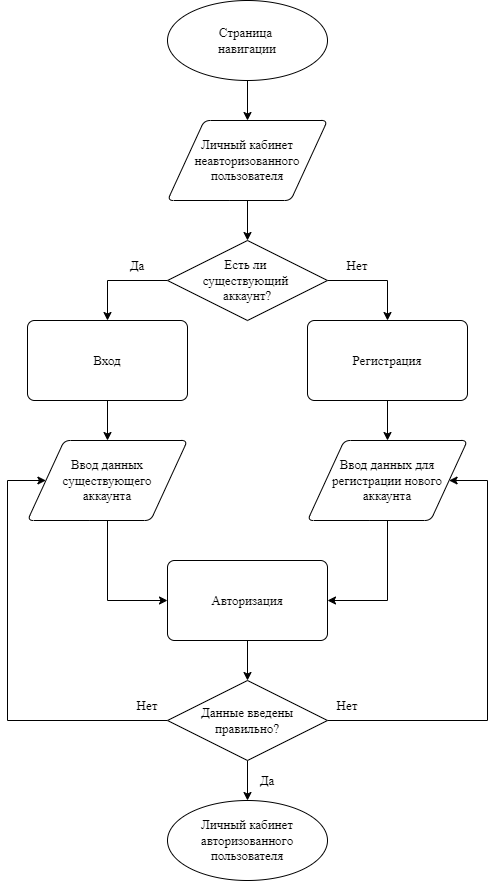


Рисунок 2 – Неавторизованный пользователь

### **1.3.4 Личный кабинет авторизованного пользователя**

В личном кабинете авторизованного пользователя предоставляется возможность выйти из аккаунта, открыть историю транзакций или посмотреть баланс бонусов и скидки. Схема взаимодействия представлена на рисунке 3.

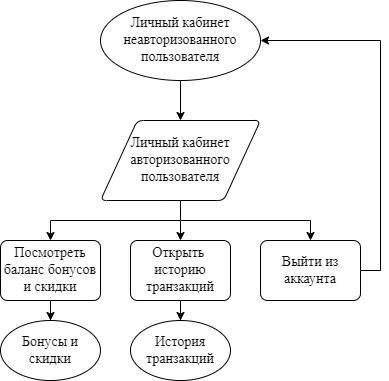


Рисунок 3 –Авторизованный пользователь

### **1.3.5 Бонусы и скидки**

На данной странице пользователю предоставляется возможность посмотреть текущий баланс бонусных баллов и скидки. Баланс бонусов зависит от данных о существующих транзакциях. Схема взаимодействия представлена на рисунке 4.

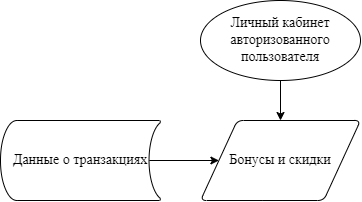


Рисунок 4 – Бонусы и скидки

### **1.3.6 История транзакций**

На данной странице пользователю предоставляется возможность посмотреть историю транзакций, предоставляющую следующую информацию о транзакции:

− Дата транзакции;

− Количество приобретенных литров;

− Тип приобретенного бензина;

− Общая сумма транзакции;

− Начисленные бонусы.

Кроме просмотра уже существующих транзакций, пользователю предоставляется возможность добавить новую, перейдя на другую страницу. Схема взаимодействия представлена на рисунке 5.

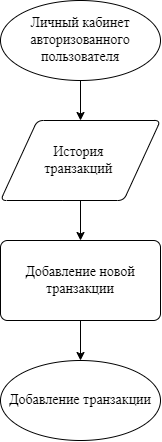


Рисунок 5 – История транзакций

### **1.3.7 Добавление транзакции**

На данной странице пользователю предоставляется возможность ввести данные о новой транзакции, используя поле ввода, выпадающий список и интерактивный календарь. При успешном вводе данных, они сохраняются, в обратном случае пользователь получает уведомление об ошибке. Схема взаимодействия представлена на рисунке 6.

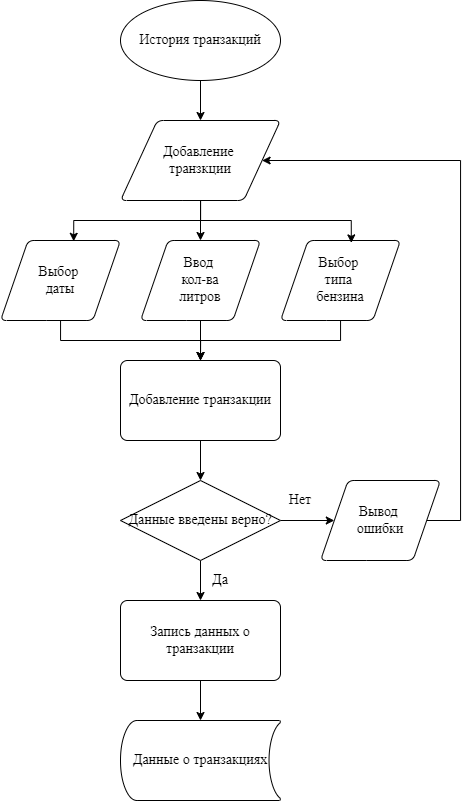


Рисунок 6 – Добавление транзакции

## **1.4 Средства и методы создания системы**

**1.4.1 Библиотеки**

Разработка автоматизированной информационной системы личного кабинета для сайта компании «ЛУКОЙЛ» осуществляется с использованием языка программирования Python и следующих библиотек:

Tkinter – является стандартной библиотекой Python для создания интерфейсов. Она предоставляет возможность создания целого набора элементов интерфейса. В данной системе с помощью данной библиотеки созданы:

− Окна;

− Надписи;

− Изображения;

− Кнопки;

− Поля ввода;

− Выпадающие списки;

− Таблицы;

− Графические элементы

Кроме создания объектов библиотека предоставляет инструменты для их размещения и обработки событий, а также для управления геометрией и внешним видом окон.

tkcalendar – Сторонняя библиотека для Tkinter, предоставляющая возможность создания объекта Calendar [2], который используется в окне добавления транзакции для выбора даты. Библиотека упрощает создание интерактивного календаря, избавляя от необходимости ручного написания кода для его реализации.

ttk – Сторонняя библиотека для Tkinter, позволяющая создавать дополнительные типы объектов и стилизовать отдельные объекты библиотеки Tkinter [3]. В проекте она используется для изменения цветов текста и фона в выпадающих списках и таблицах.

### **1.4.2 Методы создания программным путем**

Автоматизированная информационная система личного кабинета для сайта компании «ЛУКОИЛ» реализуется программным методом с использованием линейных разветвляющихся, и цикличных алгоритмов. Также используются математические расчёты.

Для реализации окон, надписей, изображений и т.д. используются соответствующие объекты и методы из импортированных библиотек.

Для размещения объектов применяется метод place, в нем указываются координаты в окне.

Для обработки событий применяется метод bind, он позволяет вызывать разные функции при разных событиях, происходящих с объектом. Пример − изменение цвета кнопки при наведении курсора.

### **1.4.3 Реализация отдельных объектов**

Для создания окон используется объект Tk для создания стартового окна, и объекты TopLevel для вторичных окон. Для их настройки используются методы:

− config для конфигурации разрешения и фона;

− title для названия окон;

− resizable для настройки доступности изменения размера окна;

− iconbitmap для установки значков в шапке окон.

С помощью этих объектов и методов реализуется стартовое окно:

main\_window = Tk()  
main\_window.config(width=1280, height=747, bg=main\_color)  
main\_window.resizable(False, False)  
main\_window.title('ЛУКОИЛ - Официальный сайт Компании «ЛУКОИЛ»')  
main\_window.iconbitmap('logo.ico')

И вторичные окна, например окно личного кабинета:

acc\_window = Toplevel(main\_window)  
acc\_window.title('Личный кабинет')  
acc\_window.config(width=400, height=480, bg=main\_color)  
acc\_window.resizable(False, False)  
acc\_window.iconbitmap('logo.ico')

Для реализации надписей используется объект Label, позволяющий конфигурировать текст надписи, её шрифт, размер, цвета надписи. При наведении или отведении курсора происходит обработка события и вызывается соответствующая функция, меняющая цвет текста:

def global\_business\_gray(event):  
 global\_business\_lb['fg'] = red\_gray\_color  
def global\_business\_white(event):  
 global\_business\_lb['fg'] = white\_color  
global\_business\_lb = Label(text='Глобальный бизнес',font=font15b, fg=white\_color, bg=red\_color)  
global\_business\_lb.place(x=30, y=7)  
global\_business\_lb.bind('<Enter>', global\_business\_gray)  
global\_business\_lb.bind('<Leave>', global\_business\_white)

Для создания изображений используются объект PhotoImage для сохранения файла в переменной, и объект Label, позволяющий размещать не только надписи, но и изображения:

logo\_blackbg = PhotoImage(file='logo\_blackbg.png')  
logo\_graybg = PhotoImage(file='logo\_graybg.png')  
lukoil\_logo = Label(image=logo\_blackbg, bg=black\_color, borderwidth=0, highlightthickness=0)

Для реализации кнопок используется объект Button, позволяющий конфигурировать внешний вид кнопки, предоставляя возможность сделать кнопкой как текст, так и изображение, изменять размер, цвета, настраивать реакцию кнопки на нажатие, устанавливать команду, которую будет вызывать кнопка. При наведении или отведении курсора происходит обработка события и вызывается соответствующая функция, меняющая картинки с разными цветами:

def account\_gray(event):  
 account\_button['image'] = account\_image\_gray  
def account\_white(event):  
 account\_button['image'] = account\_image\_white  
account\_image\_white = PhotoImage(file='user\_white.png')  
account\_image\_gray = PhotoImage(file='user\_gray.png')

account\_button = Button(image=account\_image\_white, bg=red\_color, borderwidth=0, highlightthickness=0, activebackground=red\_color, command=account\_window)  
account\_button.place(x=1245, y=9)  
account\_button.bind('<Enter>', account\_gray)  
account\_button.bind('<Leave>', account\_white)

Для создания полей ввода используется объект Entry, позволяющий конфигурировать внешний вид поля, устанавливать базовое значение в поле, настраивать реакцию поля на нажатие и параметры отображения данных. В неактивном состоянии поле ввода отображает подсказку о необходимости ввести пароль. При активации поля подсказка пропадает, оставляя место для ввода пароля. Если пользователь начинает вводить символы, поле начинает их скрывать:

def password\_clear(event):  
 if password\_entry.get() and password\_entry.get() != 'Придумайте пароль' and password\_entry.get() != 'Введите пароль':  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 else:  
 password\_entry.delete(0, END)  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
def password\_start(event):  
 if password\_entry.get():  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 else:  
 password\_entry.delete(0, END)  
 if reg\_log == 'reg':  
 password\_entry.insert(0, 'Придумайте пароль')  
 if reg\_log == 'log':  
 password\_entry.insert(0, 'Введите пароль')  
def password\_enter(event):  
 password\_entry['show'] = '\*'  
password = StringVar()  
password\_entry = Entry(acc\_window, textvariable=password, width=28, fg=black\_color, font=font17,  
 justify='center', show='',  
 relief='groove', highlightcolor=red\_color, highlightthickness=2, borderwidth=0,  
 selectbackground=red\_color)  
password\_entry.bind('<FocusIn>', password\_clear)  
password\_entry.bind('<FocusOut>', password\_start)  
password\_entry.bind('<KeyPress>', password\_enter)

Для создания выпадающих списков используется объект Combobox, позволяющий устанавливать возможность ручного ввода данных, перечень выбираемых опций и внешний вид списка. Список возможных опций для выбора задается заранее. С помощью объекта Style и метода configure осуществляется настройка цветов в списке:

gas\_types = ['92', '95', '100', 'ДТ']  
gas\_var = StringVar()  
gas\_style = ttk.Style()  
gas\_style.configure('TCombobox', selectbackground=white\_color, selectforeground=red\_color, background=white\_color, foreground=red\_color, fieldbackground=white\_color, darkcolor=red\_color, lightcolor=white\_color)  
gas\_choose = ttk.Combobox(add\_window, state='readonly', textvariable=gas\_var, font=font17b, values=gas\_types, width=4)  
gas\_choose.place(x=185, y=235)

Для создания таблиц используется объект Treeview, позволяющий настраивать, какие столбцы будут отображены в таблице, а так же конфигурировать параметры их отображения. Объект так же предполагает создание заголовков и столбцов, применяя методы heading и column:

transactions\_list = ttk.Treeview(exp\_window, columns=('Дата', 'Литры', 'Бензин', 'Сумма', 'Бонусы'), show="headings", height=12)  
transactions\_list.column('#1', anchor=CENTER, stretch=NO, width=100)  
transactions\_list.heading('#1', text='Дата')

Для создания графических элементов используется объект Canvas, позволяющий настраивать ширину, длину, фон, границы полотна. С помощью метода create\_polygon создаются различные геометрические фигуры внутри полотна:

red\_header = Canvas(width=1280, height=40, bg=red\_color, borderwidth=0, highlightthickness=0)  
red\_header.place(x=0, y=0)  
red\_header.create\_polygon(210, 20, 220, 20, 215, 25, fill=white\_color, activefill=red\_gray\_color)

### **1.4.4 Реализация смены темы**

По умолчанию устанавливается светлая тема, при вызове функции смены темы меняется значок, символизирующий текущую тему, а цвета интерфейса меняются на соответствующие текущей теме:

theme = 'light'  
def dark\_theme(event):  
 global main\_color, red\_color, theme  
 theme = 'dark'  
 main\_color = dark\_gray\_color  
 red\_color = red\_color\_night  
 main\_window.configure(bg=main\_color)  
 red\_header['bg'] = red\_color\_night  
 global\_business\_lb['bg'] = red\_color\_night

...  
 network\_lb['bg'] = red\_color\_night  
 tenders\_lb['bg'] = red\_color\_night

vacancies\_lb['bg'] = red\_color\_night  
contacts\_lb['bg'] = red\_color\_night  
search\_button['image'] = search\_img\_night  
div\_search['bg'] = red\_color\_night  
lang\_style.configure('TCombobox', selectforeground=red\_color\_night, foreground=red\_color\_night)

...  
poster1\_img['file'] = 'poster1\_dark.png'  
poster2\_img['file'] = 'poster2\_dark.png'  
poster3\_img['file'] = 'poster3\_dark.png'  
poster4\_img['file'] = 'poster4\_dark.png'  
poster5\_img['file'] = 'poster5\_dark.png'  
poster6\_img['file'] = 'poster6\_dark.png'

Так же при открытии каждого окна проверяется текущая тема и в соответствии с этим устанавливаются цвета в окне:

if theme == 'light':  
 main\_color = white\_color  
 red\_color = red\_color\_day  
if theme == 'dark':  
 main\_color = dark\_gray\_color  
 red\_color = red\_color\_night

### **1.4.5 Реализация регистрации и входа**

В начале работы программы данных о пользователе и транзакциях нет, статус личного кабинета установлен на режим регистрации, баланс бонусов равен нулю. При вводе в поля данные сохраняются.

При регистрации создаётся новый пользователь и запускается интерфейс авторизованного пользователя:

if u\_name != 'Введите ваше имя' and u\_login != 'Придумайте логин' and u\_password != 'Придумайте пароль':  
 user = [u\_name, u\_login, u\_password]  
 authorise\_status = True  
 reg\_log\_lb.destroy()  
 user\_name\_entry.destroy()  
 login\_entry.destroy()  
 password\_entry.destroy()  
 authorise\_button.destroy()  
 reg\_log\_button.destroy()  
 authorised()

При попытке входа в уже существующий аккаунт проверяется наличие такого аккаунта и соответствие введенных данных с данными о существующем аккаунте. При успешном входе запускается интерфейс авторизованного пользователя:

if user!=[] and user[1] == u\_login and user[2] == u\_password:  
 authorise\_status = True  
 reg\_log\_lb.destroy()  
 user\_name\_entry.destroy()  
 login\_entry.destroy()  
 password\_entry.destroy()  
 authorise\_button.destroy()  
 reg\_log\_button.destroy()  
 authorised()

### **1.4.6 Реализация добавления транзакций**

При отображении истории транзакций данные передаются из списка с сохраненными транзакциями в таблицу:

for transaction in transactions:  
 transactions\_list.insert('', END, values=(transaction['Дата'], transaction['Литры'], transaction['Бензин'], transaction['Сумма'], transaction['Бонусы'],))

При добавлении транзакции проверяется наличие всех требуемых данных и производится передача их в переменные и сохранение в списке. Данные о транзакции передаются в таблицу, баланс бонусов обновляется. В случае неверно введенных данных или их отсутствии пользователь получает уведомление об ошибке:

def get\_transaction():  
 global bonuses  
 price = 0  
 add\_button['text'] = 'Добавить транзакцию'  
 date = calendar.get\_date()  
 try:  
 litres = int(litres\_entry.get())  
 except:  
 add\_button['text'] = 'Ошибка'  
 gas = gas\_choose.get()  
 if gas and date:  
 match gas:  
 case '92':  
 price = 53  
 case '95':  
 price = 57  
 case '100':  
 price = 82  
 case 'ДТ':  
 price = 68  
 summ = price\*litres  
 transactions.append({'Дата': date, 'Литры': litres, 'Бензин': gas, 'Сумма': summ, 'Бонусы': round(summ\*0.1, 2)})  
 bonuses += round(summ\*0.1, 2)  
 transactions\_list.insert('', END, values=(transactions[-1]['Дата'], transactions[-1]['Литры'], transactions[-1]['Бензин'], transactions[-1]['Сумма'], transactions[-1]['Бонусы'],))  
 else:  
 add\_button['text'] = 'Ошибка'

# **2 Реализация алгоритма**

## **2.1 Состав информационной системы**

Разработанная информационная система является прототипом главной страницы сайта компании ЛУКОЙЛ и включает в себя пять окон:

− Страница навигации

− Личный кабинет неавторизованного и авторизованного пользователя

− Бонусы и скидки

− История транзакций

− Добавление транзакций

Для удобства пользователя и интерактивности во всех разделах кнопки подсвечиваются при наведении, есть возможность смены темы. Во всех разделах применена корпоративная цветовая схема, в которую включены цвета, представленные в таблице 2:

Таблица 2 – Цветовая схема

|  |  |  |  |
| --- | --- | --- | --- |
| Цвет | Визуальное  представление | HEX – код  цвета | Назначение цвета |
| Белый |  | #ffffff | Цвет фона для светлой темы,  цвет фона полей ввода, таблиц, выпадающих списков |
| Черный |  | #000000 | Цвет фона некоторых объектов, цвет шрифта |
| Серый |  | #555555 | Цвет фона для некоторых объектов в темной теме |

Продолжение таблицы 2

|  |  |  |  |
| --- | --- | --- | --- |
| Темно−  серый |  | #404040 | Цвет фона для темной темы |
| Розовый |  | #e9928d | Цвет шрифта при наведении курсора |
| Красный |  | #db2b36 | Основной цвет для объектов в светлой теме |
| Темно−красный |  | #800000 | Основной цвет для объектов в темной теме |

## **2.2 Страница навигации**

Страница навигации представляет собой имитацию внешнего вида главной страницы сайта компании «ЛУКОИЛ». По умолчанию применена светлая тема с соблюдением корпоративной цветовой схемы. Для пользователя предоставляется возможность поменять тему, пролистать баннеры и перейти в личный кабинет. Смена темы применяет другую тему во всех окнах с соблюдением корпоративного стиля. Перелистывание баннеров производится с помощью кнопок, отвечающих за каждый из них. Переход в личный кабинет открывает новое окно с соответствующим функционалом Для интерактивности и улучшения пользовательского опыта реализованы функции смены цвета шрифта при наведении на кнопки. В светлой теме используются белый, черный, розовый и красный цвета. Внешний вид страницы навигации в светлой теме представлен на рисунке 8.

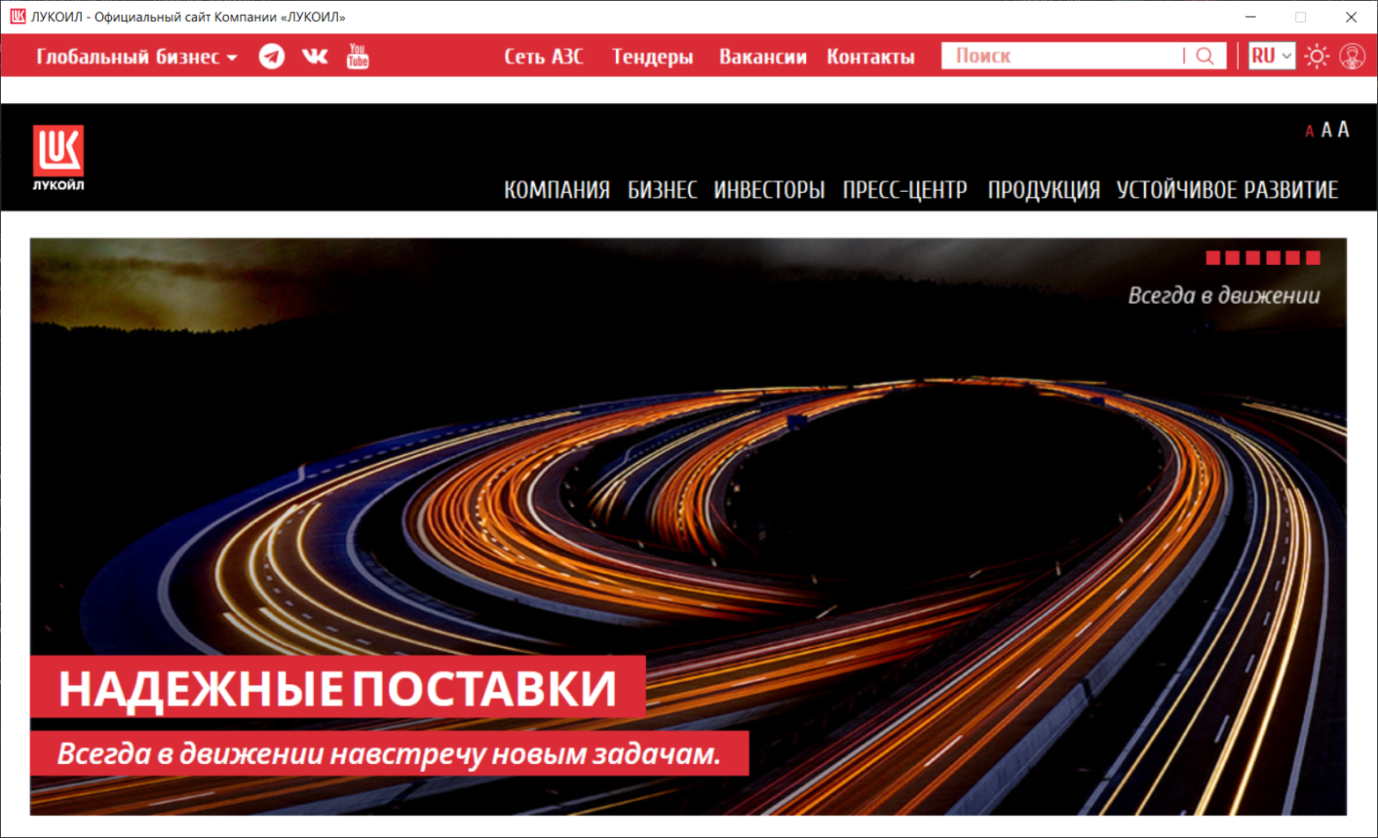


Рисунок 8 − Страница навигации в светлой теме

В светлой теме используются белый, черный, серый, темно−серый, розовый и темно−красный цвета. Внешний вид страницы навигации в темной теме представлен на рисунке 9.

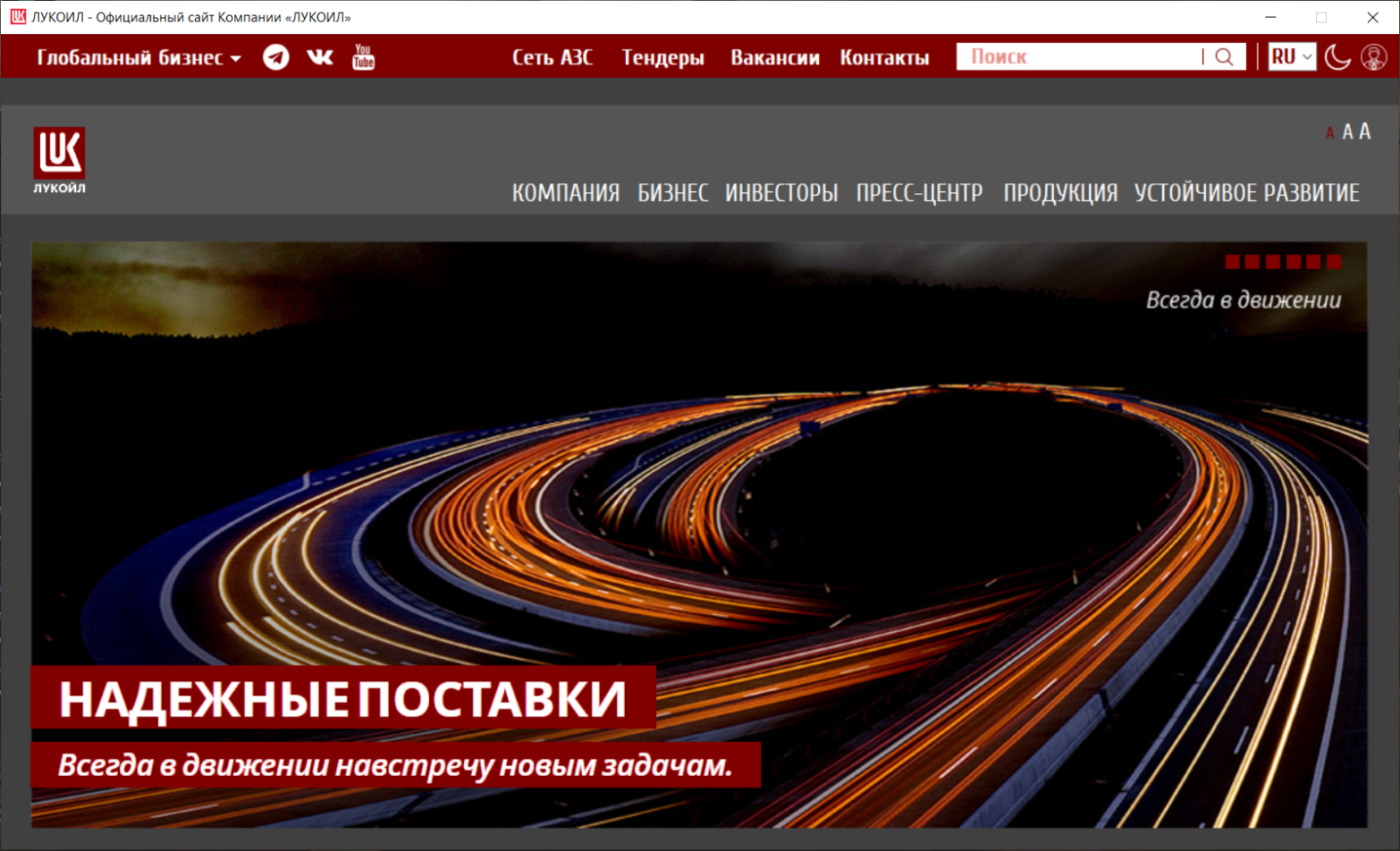


Рисунок 9 − Страница навигации в темной теме

## **2.3 Личный кабинет неавторизованного пользователя**

Личный кабинет по умолчанию предлагает пользователю зарегистрироваться, но также предоставляет возможность выполнить вход в уже существующий аккаунт. При неверно введённых данных пользователь получает оповещение в виде надписи на кнопке авторизации. Внешний вид личного кабинета неавторизованного пользователя при регистрации в обеих темах представлен на рисунке 10. Внешний вид личного кабинета неавторизованного пользователя при входе в обеих темах представлен на рисунке 11.

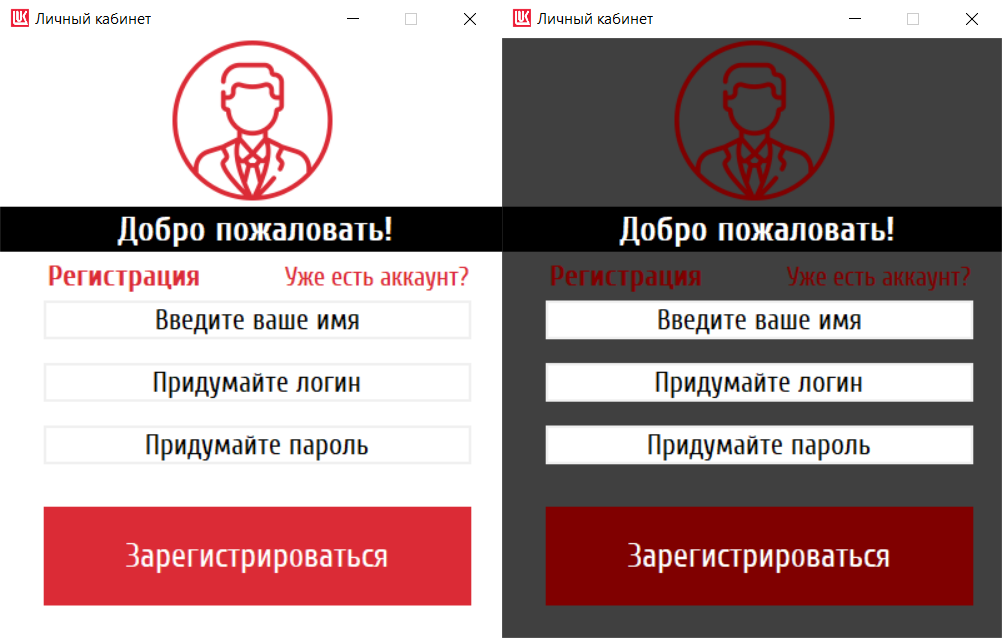


Рисунок 10 – Регистрация

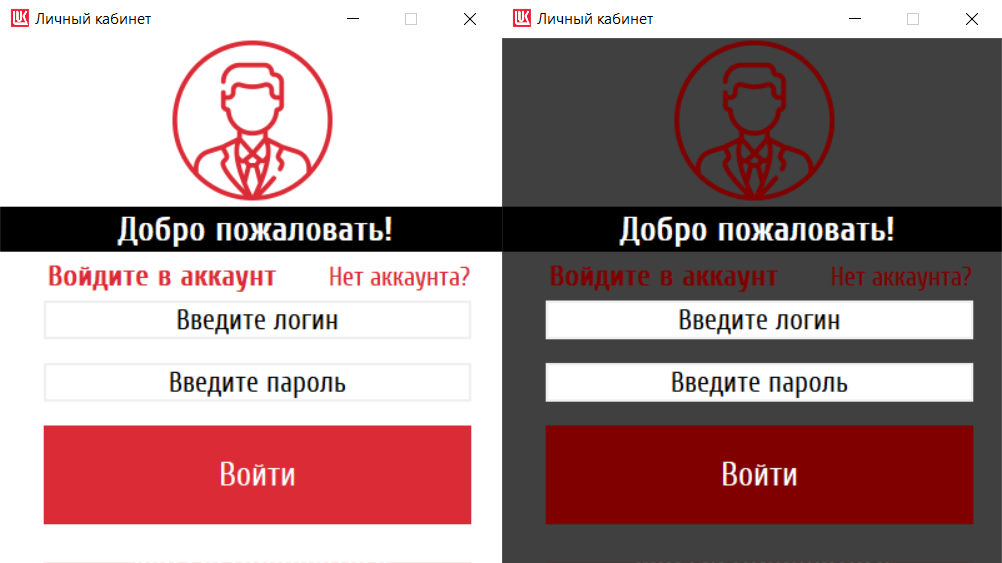


Рисунок 11 – Вход

## **2.4 Личный кабинет авторизованного пользователя**

Личный кабинет при успешной авторизации предоставляет пользователю возможность открыть историю транзакций, баланс бонусных баллов и выйти из аккаунта. При этом все данные об аккаунте, транзакциях и бонусах сохраняются. Внешний вид личного кабинета авторизованного пользователя в обеих темах представлен на рисунке 12.

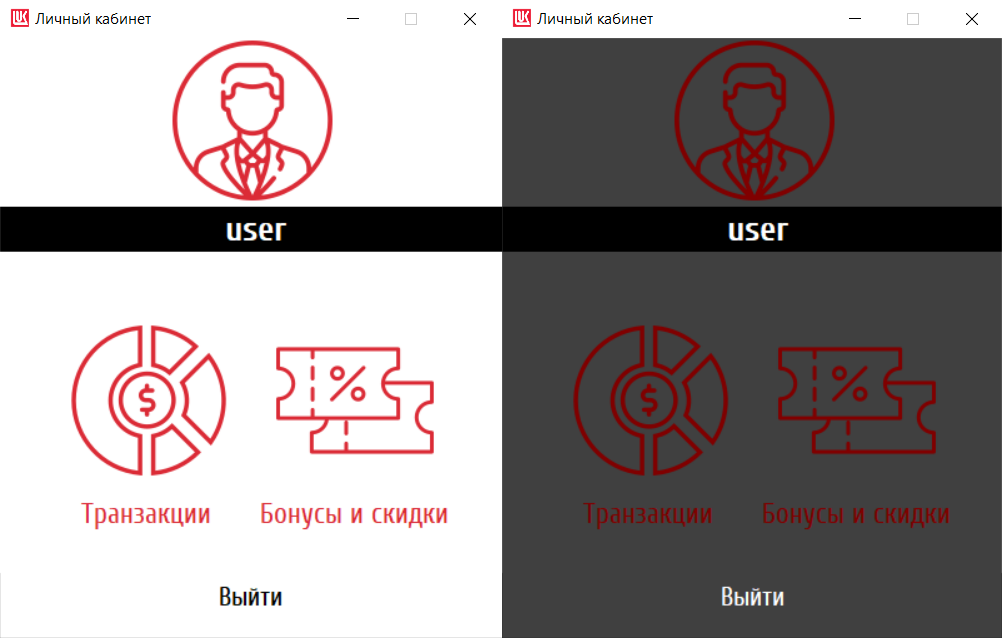


Рисунок 12 – Личный кабинет

## **2.5 Бонусы и акции**

Данный раздел предоставляет пользователю возможность получить информацию о текущем балансе бонусных баллов, а также получить информацию о промокодах. Внешний вид раздела бонусов и скидок в обеих темах представлен на рисунке 13.



Рисунок 13 – Бонусы и скидки

## **2.6 История транзакций**

Данный раздел предоставляет пользователю информацию о транзакциях, такую как:

− Дата транзакции;

− Количество приобретенных литров;

− Тип приобретенного бензина;

− Общая сумма транзакции;

− Начисленные бонусы.

Также пользователь имеет возможность добавить новую транзакции, нажав соответствующую кнопку, открывающую специальное окно. Внешний вид раздела истории транзакций представлен на рисунке 14.

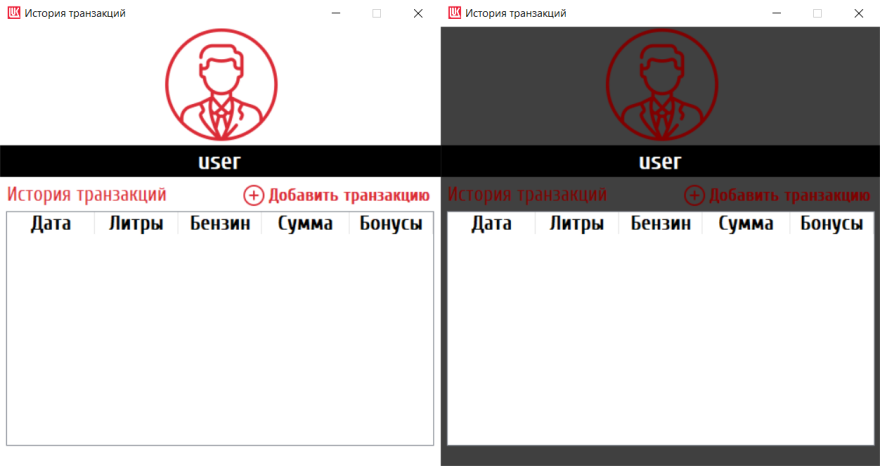


Рисунок 14 – История транзакций

## **2.7 Добавление транзакции**

Данный раздел предоставляет пользователю возможность добавить новую транзакцию в историю, позволяя выбрать дату в интерактивном интерфейсе календаря, ввести количество купленных литров и выбрать тип топлива в выпадающем списке. При неверно введённых данных пользователь получает оповещение в виде надписи на кнопке добавления транзакции. Внешний вид раздела добавления транзакции представлен на рисунке 15.

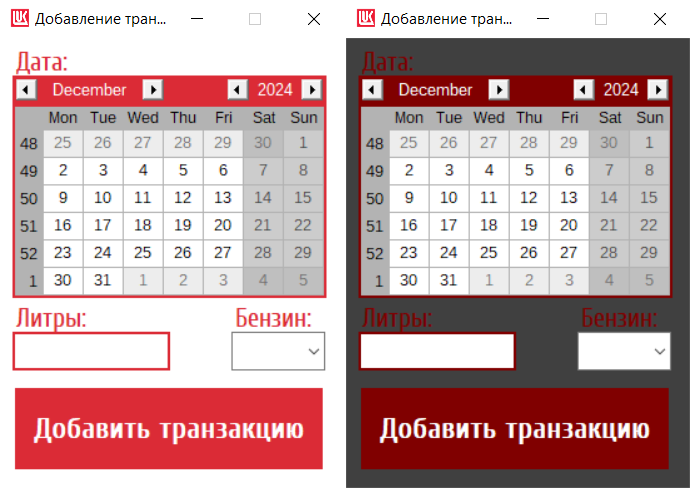


Рисунок 15 – Добавление транзакции

# ЗАКЛЮЧЕНИЕ

В заключение следует отметить, что в рамках данной работы был разработан прототип пользовательского интерфейса, имитирующий главную страницу и личный кабинет веб−сайта ПАО «ЛУКОЙЛ». Целью работы являлась разработка прототипа интуитивно понятного интерфейса личного кабинета, обеспечивающего удобный доступ к истории транзакций и бонусам.

Для достижения поставленной цели были успешно решены следующие задачи:

− Проведен анализ функционала сайта заказчика, что позволило определить ключевые элементы, необходимые для реализации в прототипе;

− Определена потребность заказчика в добавлении функционала личного кабинета;

− Изучены программные методы и средства для реализации системы;

− Разработана и реализована информационная система в виде прототипа, включающего смену цветовой темы и интерфейс личного кабинета с отображением истории транзакций и бонусной программой;

− Проведено тестирование информационной системы, что позволило убедиться в ее работоспособности и выявить потенциальные области для улучшения.

Таким образом, все поставленные задачи были выполнены, и цель работы была достигнута. Несмотря на то, что разработанный прототип не является полноценным веб−сайтом и не включает в себя весь спектр функций реального веб−приложения, он демонстрирует успешную реализацию ключевых аспектов пользовательского опыта. Прототип может служить в качестве основы для дальнейшей разработки и доработки, демонстрируя потенциал использования Tkinter для создания привлекательных и эргономичных пользовательских интерфейсов.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. ЛУКОЙЛ – О Компании [Электронный ресурс]. Режим доступа: <https://lukoil.ru/Company/CorporateProfile>. Дата обращения (10.10.2024).

2. tkcalendar PyPI [Электронный ресурс]. Режим доступа: https://pypi.org/project/tkcalendar. Дата обращения (03.12.2024).

3 tkinter.ttk — Tk themed widgets [Электронный ресурс]. Режим доступа: <https://docs.python.org/3/library/tkinter.ttk.html> Дата обращение (4.12.2024)

# ПРИЛОЖЕНИЕ

ПРИЛОЖЕНИЕ А

from tkinter import \*  
from tkinter import ttk  
from tkcalendar import Calendar  
  
white\_color = '#ffffff'  
black\_color = '#000000'  
gray\_color = '#555555'  
dark\_gray\_color = '#404040'  
red\_gray\_color = '#e9928d'  
red\_color\_day = '#db2b36'  
red\_color\_night = '#800000'  
main\_color = white\_color  
red\_color = red\_color\_day  
  
font12 = 'Cuprum 12 normal'  
font14 = 'Cuprum 14 normal'  
font14b = 'Cuprum 14 bold'  
font15b = 'Cuprum 15 bold'  
font16 = 'Cuprum 16 normal'  
font17 = 'Cuprum 17 normal'  
font17b = 'Cuprum 17 bold'  
font20 = 'Cuprum 20 normal'  
font20b = 'Cuprum 20 bold'  
  
main\_window = Tk()  
main\_window.config(width=1280, height=747, bg=main\_color)  
main\_window.resizable(False, False)  
main\_window.title('ЛУКОИЛ - Официальный сайт Компании «ЛУКОИЛ»')  
main\_window.iconbitmap('logo.ico')  
  
red\_header = Canvas(width=1280, height=40, bg=red\_color, borderwidth=0, highlightthickness=0)  
red\_header.place(x=0, y=0)  
red\_header.create\_polygon(210, 20, 220, 20, 215, 25, fill=white\_color, activefill=red\_gray\_color)  
  
def global\_business\_gray(event):  
 global\_business\_lb['fg'] = red\_gray\_color  
def global\_business\_white(event):  
 global\_business\_lb['fg'] = white\_color  
global\_business\_lb = Label(text='Глобальный бизнес',font=font15b, fg=white\_color, bg=red\_color)  
global\_business\_lb.place(x=30, y=7)  
global\_business\_lb.bind('<Enter>', global\_business\_gray)  
global\_business\_lb.bind('<Leave>', global\_business\_white)  
  
def tg\_red(event):  
 tg['image'] = tg\_logo2

Продолжение ПРИЛОЖЕНИЯ А

def tg\_white(event):  
 tg['image'] = tg\_logo1  
tg\_logo1 = PhotoImage(file='tg\_logo1.png')  
tg\_logo2 = PhotoImage(file='tg\_logo2.png')  
tg = Label(image=tg\_logo1, bg=red\_color, borderwidth=0, highlightthickness=0)  
tg.place(x=240, y=9)  
tg.bind('<Enter>', tg\_red)  
tg.bind('<Leave>', tg\_white)  
  
def vk\_red(event):  
 vk['image'] = vk\_logo2  
def vk\_white(event):  
 vk['image'] = vk\_logo1  
vk\_logo1 = PhotoImage(file='vk\_logo1.png')  
vk\_logo2 = PhotoImage(file='vk\_logo2.png')  
vk = Label(image=vk\_logo1, bg=red\_color, borderwidth=0, highlightthickness=0)  
vk.place(x=280, y=9)  
vk.bind('<Enter>', vk\_red)  
vk.bind('<Leave>', vk\_white)  
  
def yt\_red(event):  
 yt['image'] = yt\_logo2  
def yt\_white(event):  
 yt['image'] = yt\_logo1  
yt\_logo1 = PhotoImage(file='yt\_logo1.png')  
yt\_logo2 = PhotoImage(file='yt\_logo2.png')  
yt = Label(image=yt\_logo1, bg=red\_color, borderwidth=0, highlightthickness=0)  
yt.place(x=320, y=9)  
yt.bind('<Enter>', yt\_red)  
yt.bind('<Leave>', yt\_white)  
  
def network\_gray(event):  
 network\_lb['fg'] = red\_gray\_color  
def network\_white(event):  
 network\_lb['fg'] = white\_color  
network\_lb = Label(text='Сеть АЗС',font=font15b, fg=white\_color, bg=red\_color)  
network\_lb.place(x=465, y=7)  
network\_lb.bind('<Enter>', network\_gray)  
network\_lb.bind('<Leave>', network\_white)  
  
def tenders\_gray(event):  
 tenders\_lb['fg'] = red\_gray\_color  
def tenders\_white(event):  
 tenders\_lb['fg'] = white\_color  
tenders\_lb = Label(text='Тендеры',font=font15b, fg=white\_color, bg=red\_color)

Продолжение ПРИЛОЖЕНИЯ А

tenders\_lb.place(x=565, y=7)  
tenders\_lb.bind('<Enter>', tenders\_gray)  
tenders\_lb.bind('<Leave>', tenders\_white)  
  
def vacancies\_gray(event):  
 vacancies\_lb['fg'] = red\_gray\_color  
def vacancies\_white(event):  
 vacancies\_lb['fg'] = white\_color  
vacancies\_lb = Label(text='Вакансии',font=font15b, fg=white\_color, bg=red\_color)  
vacancies\_lb.place(x=665, y=7)  
vacancies\_lb.bind('<Enter>', vacancies\_gray)  
vacancies\_lb.bind('<Leave>', vacancies\_white)  
  
def contacts\_gray(event):  
 contacts\_lb['fg'] = red\_gray\_color  
def contacts\_white(event):  
 contacts\_lb['fg'] = white\_color  
contacts\_lb = Label(text='Контакты',font=font15b, fg=white\_color, bg=red\_color)  
contacts\_lb.place(x=765, y=7)  
contacts\_lb.bind('<Enter>', contacts\_gray)  
contacts\_lb.bind('<Leave>', contacts\_white)  
  
def search\_clear(event):  
 search.delete(0, END)  
def search\_start(event):  
 if search == '':  
 search.insert(0, ' Поиск')  
  
search = Entry(main\_window, font=font15b, fg=red\_gray\_color, bg=main\_color, borderwidth=0, highlightthickness=0)  
search.place(x=875, y=8, width=265, height=25)  
search.insert(0, ' Поиск')  
search.bind('<ButtonPress>', search\_clear)  
search.bind('<FocusOut>', search\_start)  
div\_search = Canvas(width=1, height=15, bg=red\_color, borderwidth=0, highlightthickness=0, relief='flat')  
div\_search.place(x=1100, y=13)  
search\_img\_day = PhotoImage(file='search\_light.png')  
search\_img\_night = PhotoImage(file='search\_dark.png')  
search\_button = Button(image=search\_img\_day, bg=main\_color, borderwidth=0, highlightthickness=0, relief='flat')  
search\_button.place(x=1112, y=13)  
  
div\_lang = Canvas(width=1, height=25, bg=main\_color, borderwidth=0, highlightthickness=0)  
div\_lang.place(x=1150, y=8)  
languages = ['RU', 'EN']  
languages\_default = StringVar(value=languages[0])

Продолжение ПРИЛОЖЕНИЯ А

lang\_style = ttk.Style()  
lang\_style.configure('TCombobox', selectbackground=white\_color, selectforeground=red\_color,  
 background=white\_color, foreground=red\_color, fieldbackground=white\_color,  
 darkcolor=white\_color,  
 lightcolor=white\_color)  
languages\_choose = ttk.Combobox(main\_window, state='readonly', textvariable=languages\_default, values=languages,  
 height=35, width=2, font=font14b)  
languages\_choose.place(x=1160, y=7)  
  
black\_header = Canvas(width=1280, height=100, bg=black\_color, borderwidth=0, highlightthickness=0)  
black\_header.place(x=0, y=65)  
logo\_blackbg = PhotoImage(file='logo\_blackbg.png')  
logo\_graybg = PhotoImage(file='logo\_graybg.png')  
lukoil\_logo = Label(image=logo\_blackbg, bg=black\_color, borderwidth=0, highlightthickness=0)  
lukoil\_logo.place(x=30, y=85)  
  
def company\_red(event):  
 company\_lb['fg'] = red\_color  
def company\_white(event):  
 company\_lb['fg'] = white\_color  
company\_lb = Label(text='КОМПАНИЯ',font=font17, fg=white\_color, bg=black\_color)  
company\_lb.place(x=465, y=130)  
company\_lb.bind('<Enter>', company\_red)  
company\_lb.bind('<Leave>', company\_white)  
  
def business\_red(event):  
 business\_lb['fg'] = red\_color  
def business\_white(event):  
 business\_lb['fg'] = white\_color  
business\_lb = Label(text='БИЗНЕС',font=font17, fg=white\_color, bg=black\_color)  
business\_lb.place(x=580, y=130)  
business\_lb.bind('<Enter>', business\_red)  
business\_lb.bind('<Leave>', business\_white)  
  
def investors\_red(event):  
 investors\_lb['fg'] = red\_color  
def investors\_white(event):  
 investors\_lb['fg'] = white\_color  
investors\_lb = Label(text='ИНВЕСТОРЫ',font=font17, fg=white\_color, bg=black\_color)

Продолжение ПРИЛОЖЕНИЯ А

investors\_lb.place(x=660, y=130)  
investors\_lb.bind('<Enter>', investors\_red)  
investors\_lb.bind('<Leave>', investors\_white)

def press\_red(event):  
 press\_lb['fg'] = red\_color  
def press\_white(event):  
 press\_lb['fg'] = white\_color  
press\_lb = Label(text='ПРЕСС-ЦЕНТР',font=font17, fg=white\_color, bg=black\_color)  
press\_lb.place(x=780, y=130)  
press\_lb.bind('<Enter>', press\_red)  
press\_lb.bind('<Leave>', press\_white)  
  
def products\_red(event):  
 products\_lb['fg'] = red\_color  
def products\_white(event):  
 products\_lb['fg'] = white\_color  
products\_lb = Label(text='ПРОДУКЦИЯ',font=font17, fg=white\_color, bg=black\_color)  
products\_lb.place(x=915, y=130)  
products\_lb.bind('<Enter>', products\_red)  
products\_lb.bind('<Leave>', products\_white)  
  
def improvement\_red(event):  
 improvement\_lb['fg'] = red\_color  
def improvement\_white(event):  
 improvement\_lb['fg'] = white\_color  
improvement\_lb = Label(text='УСТОЙЧИВОЕ РАЗВИТИЕ',font=font17, fg=white\_color, bg=black\_color)  
improvement\_lb.place(x=1035, y=130)  
improvement\_lb.bind('<Enter>', improvement\_red)  
improvement\_lb.bind('<Leave>', improvement\_white)  
  
a1\_lb = Label(text='А', font=font12, fg=red\_color, bg=black\_color)  
a1\_lb.place(x=1210, y=79)  
  
def a2\_red(event):  
 if theme == 'light':  
 a2\_lb['fg'] = red\_color\_day  
 if theme == 'dark':  
 a2\_lb['fg'] = red\_color\_night  
def a2\_white(event):  
 a2\_lb['fg'] = white\_color  
a2\_lb = Label(text='А',font=font14, fg=white\_color, bg=black\_color)  
a2\_lb.place(x=1225, y=76)  
a2\_lb.bind('<Enter>', a2\_red)

Продолжение ПРИЛОЖЕНИЯ А

a2\_lb.bind('<Leave>', a2\_white)  
def a3\_red(event):  
 if theme == 'light':  
 a3\_lb['fg'] = red\_color\_day  
 if theme == 'dark':  
 a3\_lb['fg'] = red\_color\_night  
def a3\_white(event):  
 a3\_lb['fg'] = white\_color  
a3\_lb = Label(text='А',font=font16, fg=white\_color, bg=black\_color)  
a3\_lb.place(x=1240, y=75)  
a3\_lb.bind('<Enter>', a3\_red)  
a3\_lb.bind('<Leave>', a3\_white)  
  
poster1\_img = PhotoImage(file='poster1\_light.png')  
poster2\_img = PhotoImage(file='poster2\_light.png')  
poster3\_img = PhotoImage(file='poster3\_light.png')  
poster4\_img = PhotoImage(file='poster4\_light.png')  
poster5\_img = PhotoImage(file='poster5\_light.png')  
poster6\_img = PhotoImage(file='poster6\_light.png')  
poster = Label(image=poster4\_img, bg=main\_color, borderwidth=0, highlightthickness=0)  
poster.place(x=27, y=190)  
def poster1():  
 global poster  
 poster['image'] = poster1\_img  
def poster2():  
 global poster  
 poster['image'] = poster2\_img  
def poster3():  
 global poster  
 poster['image'] = poster3\_img  
def poster4():  
 global poster  
 poster['image'] = poster4\_img  
def poster5():  
 global poster  
 poster['image'] = poster5\_img  
def poster6():  
 global poster  
 poster['image'] = poster6\_img  
  
empty = PhotoImage(file='empty.png')  
button1 = Button(image=empty, bg=red\_color, borderwidth=0, highlightthickness=0, command=poster1)  
button1.place(x=1121, y=202)  
button2 = Button(image=empty, bg=red\_color, borderwidth=0, highlightthickness=0, command=poster2)  
button2.place(x=1139, y=202)  
button3 = Button(image=empty, bg=red\_color, borderwidth=0,

Продолжение ПРИЛОЖЕНИЯ А

highlightthickness=0, command=poster3)  
button3.place(x=1158, y=202)  
button4 = Button(image=empty, bg=red\_color, borderwidth=0, highlightthickness=0, command=poster4)

button4.place(x=1177, y=202)  
button5 = Button(image=empty, bg=red\_color, borderwidth=0, highlightthickness=0, command=poster5)  
button5.place(x=1195, y=202)  
button6 = Button(image=empty, bg=red\_color, borderwidth=0, highlightthickness=0, command=poster6)  
button6.place(x=1214, y=202)  
  
def sun\_gray(event):  
 theme\_button['image'] = sun\_gray\_img  
def sun\_white(event):  
 theme\_button['image'] = sun\_img  
def moon\_gray(event):  
 theme\_button['image'] = moon\_gray\_img  
def moon\_white(event):  
 theme\_button['image'] = moon\_img  
  
theme = 'light'  
def dark\_theme(event):  
 global main\_color, red\_color, theme  
 theme = 'dark'  
 main\_color = dark\_gray\_color  
 red\_color = red\_color\_night  
 main\_window.configure(bg=main\_color)  
 red\_header['bg'] = red\_color\_night  
 global\_business\_lb['bg'] = red\_color\_night  
 tg['bg'] = red\_color\_night  
 vk['bg'] = red\_color\_night  
 yt['bg'] = red\_color\_night  
 network\_lb['bg'] = red\_color\_night  
 tenders\_lb['bg'] = red\_color\_night  
 vacancies\_lb['bg'] = red\_color\_night  
 contacts\_lb['bg'] = red\_color\_night  
 search\_button['image'] = search\_img\_night  
 div\_search['bg'] = red\_color\_night  
 lang\_style.configure('TCombobox', selectforeground=red\_color\_night, foreground=red\_color\_night)  
 theme\_button['bg'] = red\_color\_night  
 theme\_button['image'] = moon\_img  
 account\_button['bg'] = red\_color\_night  
 account\_button['activebackground'] = red\_color\_night  
 black\_header['bg'] = gray\_color  
 lukoil\_logo['image'] = logo\_graybg  
 company\_lb['bg'] = gray\_color  
 business\_lb['bg'] = gray\_color  
 investors\_lb['bg'] = gray\_color

Продолжение ПРИЛОЖЕНИЯ А

press\_lb['bg'] = gray\_color  
 products\_lb['bg'] = gray\_color  
 improvement\_lb['bg'] = gray\_color

a1\_lb['bg'] = gray\_color  
 a1\_lb['fg'] = red\_color\_night  
 a2\_lb['bg'] = gray\_color  
 a3\_lb['bg'] = gray\_color  
 button1['bg'] = red\_color\_night  
 button2['bg'] = red\_color\_night  
 button3['bg'] = red\_color\_night  
 button4['bg'] = red\_color\_night  
 button5['bg'] = red\_color\_night  
 button6['bg'] = red\_color\_night  
 poster1\_img['file'] = 'poster1\_dark.png'  
 poster2\_img['file'] = 'poster2\_dark.png'  
 poster3\_img['file'] = 'poster3\_dark.png'  
 poster4\_img['file'] = 'poster4\_dark.png'  
 poster5\_img['file'] = 'poster5\_dark.png'  
 poster6\_img['file'] = 'poster6\_dark.png'  
 theme\_button.bind('<Enter>', moon\_gray)  
 theme\_button.bind('<Leave>', moon\_white)  
 theme\_button.bind('<Button>', light\_theme)  
  
def light\_theme(event):  
 global main\_color, red\_color, theme  
 theme = 'light'  
 main\_color = white\_color  
 red\_color = red\_color\_day  
 main\_window.configure(bg=main\_color)  
 red\_header['bg'] = red\_color\_day  
 global\_business\_lb['bg'] = red\_color\_day  
 tg['bg'] = red\_color\_day  
 vk['bg'] = red\_color\_day  
 yt['bg'] = red\_color\_day  
 network\_lb['bg'] = red\_color\_day  
 tenders\_lb['bg'] = red\_color\_day  
 vacancies\_lb['bg'] = red\_color\_day  
 contacts\_lb['bg'] = red\_color\_day  
 search\_button['image'] = search\_img\_day  
 div\_search['bg'] = red\_color\_day  
 lang\_style.configure('TCombobox', selectforeground=red\_color\_day, foreground=red\_color\_day)  
 theme\_button['bg'] = red\_color\_day  
 theme\_button['image'] = sun\_img  
 account\_button['bg'] = red\_color\_day  
 account\_button['activebackground'] = red\_color\_day  
 black\_header['bg'] = black\_color  
 lukoil\_logo['image'] = logo\_blackbg  
 company\_lb['bg'] = black\_color  
 business\_lb['bg'] = black\_color

Продолжение ПРИЛОЖЕНИЯ А

investors\_lb['bg'] = black\_color  
 press\_lb['bg'] = black\_color  
 products\_lb['bg'] = black\_color

improvement\_lb['bg'] = black\_color  
 a1\_lb['bg'] = black\_color  
 a1\_lb['fg'] = red\_color\_day  
 a2\_lb['bg'] = black\_color  
 a3\_lb['bg'] = black\_color  
 button1['bg'] = red\_color\_day  
 button2['bg'] = red\_color\_day  
 button3['bg'] = red\_color\_day  
 button4['bg'] = red\_color\_day  
 button5['bg'] = red\_color\_day  
 button6['bg'] = red\_color\_day  
 poster1\_img['file'] = 'poster1\_light.png'  
 poster2\_img['file'] = 'poster2\_light.png'  
 poster3\_img['file'] = 'poster3\_light.png'  
 poster4\_img['file'] = 'poster4\_light.png'  
 poster5\_img['file'] = 'poster5\_light.png'  
 poster6\_img['file'] = 'poster6\_light.png'  
 theme\_button.bind('<Enter>', sun\_gray)  
 theme\_button.bind('<Leave>', sun\_white)  
 theme\_button.bind('<Button>', dark\_theme)  
  
  
sun\_img = PhotoImage(file='sun.png')  
sun\_gray\_img = PhotoImage(file='sun\_gray.png')  
moon\_img = PhotoImage(file='moon.png')  
moon\_gray\_img = PhotoImage(file='moon\_gray.png')  
theme\_button = Label(image=sun\_img, bg=red\_color,  
 borderwidth=0, highlightthickness=0, activebackground=red\_color)  
theme\_button.bind('<Enter>', sun\_gray)  
theme\_button.bind('<Leave>', sun\_white)  
theme\_button.bind('<Button>', dark\_theme)  
theme\_button.place(x=1212, y=9)  
  
def account\_gray(event):  
 account\_button['image'] = account\_image\_gray  
def account\_white(event):  
 account\_button['image'] = account\_image\_white  
account\_image\_white = PhotoImage(file='user\_white.png')  
account\_image\_gray = PhotoImage(file='user\_gray.png')  
  
user\_logo\_img = PhotoImage(file='user\_big\_light.png')  
add\_image = PhotoImage(file='add\_light.png')  
add\_image\_gray = PhotoImage(file='add\_gray.png')  
coin\_img = PhotoImage(file='coin\_light.png')  
user = []  
authorise\_status = False

Продолжение ПРИЛОЖЕНИЯ А

reg\_log = 'reg'  
transactions = []  
bonuses = 0

def account\_window():  
 global authorise\_status  
 if theme == 'light':  
 main\_color = white\_color  
 red\_color = red\_color\_day  
 user\_logo\_img['file'] = 'user\_big\_light.png'  
 if theme == 'dark':  
 main\_color = dark\_gray\_color  
 red\_color = red\_color\_night  
 user\_logo\_img['file'] = 'user\_big\_dark.png'  
  
 acc\_window = Toplevel(main\_window)  
 acc\_window.title('Личный кабинет')  
 acc\_window.config(width=400, height=480, bg=main\_color)  
 acc\_window.resizable(False, False)  
 acc\_window.iconbitmap('logo.ico')  
  
 user\_logo = Label(acc\_window, image=user\_logo\_img, bg=main\_color, highlightthickness=0)  
 user\_logo.place(x=136, y=0)  
 welcome\_lb = Label(acc\_window, text='', width=31, justify='center', font=font20b, bg=black\_color, fg=white\_color)  
 welcome\_lb.place(x=0, y=135)  
  
 def unauthorised():  
 if theme == 'light':  
 main\_color = white\_color  
 red\_color = red\_color\_day  
 if theme == 'dark':  
 main\_color = dark\_gray\_color  
 red\_color = red\_color\_night  
 global reg\_log  
 welcome\_lb['text'] = 'Добро пожаловать!'  
 reg\_log\_lb = Label(acc\_window, text='Регистрация', font=font17b, bg=main\_color, fg=red\_color)  
 reg\_log\_lb.place(x=35, y=175)  
  
 def authorise():  
 acc\_window.config(width=400, height=480)  
 global user, authorise\_status  
 if reg\_log == 'reg':  
 u\_name = user\_name\_entry.get()  
 u\_login = login\_entry.get()  
 u\_password = password\_entry.get()  
 if u\_name != 'Введите ваше имя' and u\_login != 'Придумайте логин' and u\_password != 'Придумайте пароль':

Продолжение ПРИЛОЖЕНИЯ А

user = [u\_name, u\_login, u\_password]  
 authorise\_status = True  
 reg\_log\_lb.destroy()

user\_name\_entry.destroy()  
 login\_entry.destroy()  
 password\_entry.destroy()  
 authorise\_button.destroy()  
 reg\_log\_button.destroy()  
 authorised()  
 else:  
 authorise\_button['text'] = 'Ошибка'  
  
 if reg\_log == 'log':  
 u\_login = login\_entry.get()  
 u\_password = password\_entry.get()  
 if user!=[] and user[1] == u\_login and user[2] == u\_password:  
 authorise\_status = True  
 reg\_log\_lb.destroy()  
 user\_name\_entry.destroy()  
 login\_entry.destroy()  
 password\_entry.destroy()  
 authorise\_button.destroy()  
 reg\_log\_button.destroy()  
 authorised()  
 else:  
 acc\_window.configure(width=400, height=420)  
 authorise\_button['text'] = 'Ошибка'  
  
 def user\_name\_clear(event):  
 if user\_name\_entry.get() and user\_name\_entry.get() != 'Введите ваше имя' and user\_name\_entry.get() != (  
 '\*' \* 17):  
 authorise\_button['text'] = 'Ошибка'  
 else:  
 user\_name\_entry.delete(0, END)  
 def user\_name\_start(event):  
 if user\_name\_entry.get():  
 authorise\_button['text'] = 'Ошибка'  
 else:  
 user\_name\_entry.delete(0, END)  
 user\_name\_entry.insert(0, 'Введите ваше имя')  
 user\_name = StringVar()  
 user\_name\_entry = Entry(acc\_window, textvariable=user\_name, width=28, fg=black\_color, font=font17,  
 justify='center',

Продолжение ПРИЛОЖЕНИЯ А

relief='groove', highlightcolor=red\_color, highlightthickness=2, borderwidth=0,

selectbackground=red\_color)  
 user\_name\_entry.bind('<FocusIn>', user\_name\_clear)  
 user\_name\_entry.bind('<FocusOut>', user\_name\_start)  
  
 def login\_clear(event):  
 if login\_entry.get() and login\_entry.get() != 'Придумайте логин' and login\_entry.get() != 'Введите логин':  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 else:  
 login\_entry.delete(0, END)  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 def login\_start(event):  
 if login\_entry.get():  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 else:  
 login\_entry.delete(0, END)  
 if reg\_log == 'reg':  
 login\_entry.insert(0, 'Придумайте логин')  
 if reg\_log == 'log':  
 login\_entry.insert(0, 'Введите логин')  
  
 login = StringVar()  
 login\_entry = Entry(acc\_window, textvariable=login, width=28, fg=black\_color, font=font17, justify='center',  
 relief='groove', highlightcolor=red\_color, highlightthickness=2, borderwidth=0,  
 selectbackground=red\_color)  
 login\_entry.bind('<FocusIn>', login\_clear)  
 login\_entry.bind('<FocusOut>', login\_start)  
  
 def password\_clear(event):  
 if password\_entry.get() and password\_entry.get() != 'Придумайте пароль' and password\_entry.get() != 'Введите

Продолжение ПРИЛОЖЕНИЯ А

пароль':  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'

if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 else:  
 password\_entry.delete(0, END)  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 def password\_start(event):  
 if password\_entry.get():  
 if reg\_log == 'log':  
 authorise\_button['text'] = 'Войти'  
 if reg\_log == 'reg':  
 authorise\_button['text'] = 'Зарегистрироваться'  
 else:  
 password\_entry.delete(0, END)  
 if reg\_log == 'reg':  
 password\_entry.insert(0, 'Придумайте пароль')  
 if reg\_log == 'log':  
 password\_entry.insert(0, 'Введите пароль')  
 def password\_enter(event):  
 password\_entry['show'] = '\*'  
 password = StringVar()  
 password\_entry = Entry(acc\_window, textvariable=password, width=28, fg=black\_color, font=font17,  
 justify='center', show='',  
 relief='groove', highlightcolor=red\_color, highlightthickness=2, borderwidth=0,  
 selectbackground=red\_color)  
 password\_entry.bind('<FocusIn>', password\_clear)  
 password\_entry.bind('<FocusOut>', password\_start)  
 password\_entry.bind('<KeyPress>', password\_enter)  
  
 authorise\_button = Button(acc\_window, text='Зарегистрироваться', font=font20, bg=red\_color, fg=white\_color,  
 width=28, height=2, relief='flat', borderwidth=0, command=authorise,  
 activebackground=red\_gray\_color,

Продолжение ПРИЛОЖЕНИЯ А

activeforeground=white\_color)  
  
 def reg(event):

global reg\_log  
 reg\_log = 'reg'  
 reg\_log\_lb['text'] = 'Регистрация'  
 reg\_log\_button['text'] = 'Уже есть аккаунт?'  
 reg\_log\_button.place(x=225, y=177)  
 reg\_log\_button.bind('<Button>', log)  
 acc\_window.config(width=400, height=480, bg=main\_color)  
 registration()  
 def registration():  
 user\_name\_entry.place(x=35, y=210)  
 user\_name\_entry.delete(0, END)  
 user\_name\_entry.insert(0, 'Введите ваше имя')  
 login\_entry.place(x=35, y=260)  
 login\_entry.delete(0, END)  
 login\_entry.insert(0, 'Придумайте логин')  
 password\_entry.place(x=35, y=310)  
 password\_entry.delete(0, END)  
 password\_entry['show'] = ''  
 password\_entry.insert(0, 'Придумайте пароль')  
 authorise\_button['text'] = 'Зарегистрироваться'  
 authorise\_button.place(x=35, y=375)  
  
 def log(event):  
 global reg\_log  
 reg\_log = 'log'  
 reg\_log\_lb['text'] = 'Войдите в аккаунт'  
 reg\_log\_button['text'] = 'Нет аккаунта?'  
 reg\_log\_button.place(x=260, y=177)  
 reg\_log\_button.bind('<Button>', reg)  
 acc\_window.config(width=400, height=420, bg=main\_color)  
 log\_in()  
 def log\_in():  
 user\_name\_entry.place(x=10000, y=10000)  
 login\_entry.place(x=35, y=210)  
 login\_entry.delete(0, END)  
 login\_entry.insert(0, 'Введите логин')  
 password\_entry.place(x=35, y=260)  
 password\_entry.delete(0, END)  
 password\_entry.insert(0, 'Введите пароль')  
 authorise\_button['text'] = 'Войти'  
 authorise\_button.place(x=35, y=310)  
  
 if reg\_log == 'reg':  
 registration()  
 if reg\_log == 'log':  
 log\_in()

Продолжение ПРИЛОЖЕНИЯ А  
  
 def login\_gray(event):

reg\_log\_button['fg'] = red\_gray\_color  
 def login\_red(event):  
 if theme == 'light':  
 reg\_log\_button['fg'] = red\_color\_day  
 if theme == 'dark':  
 reg\_log\_button['fg'] = red\_color\_night  
 reg\_log\_button = Label(acc\_window, text='Уже есть аккаунт?', font=font16, bg=main\_color, fg=red\_color)  
 reg\_log\_button.place(x=225, y=177)  
 reg\_log\_button.bind('<Enter>', login\_gray)  
 reg\_log\_button.bind('<Leave>', login\_red)  
 reg\_log\_button.bind('<Button>', log)  
  
 def authorised():  
 def log\_out():  
 global authorise\_status, reg\_log  
 authorise\_status = False  
 expenses\_img\_lb.destroy()  
 expenses\_lb.destroy()  
 bonuses\_img\_lb.destroy()  
 bonuses\_lb.destroy()  
 log\_out\_button.destroy()  
 reg\_log = 'reg'  
 unauthorised()  
  
 def expenses\_window():  
 if theme == 'light':  
 main\_color = white\_color  
 sec\_color = black\_color  
 red\_color = red\_color\_day  
 add\_image['file'] = 'add\_light.png'  
 if theme == 'dark':  
 main\_color = dark\_gray\_color  
 sec\_color = white\_color  
 red\_color = red\_color\_night  
 add\_image['file'] = 'add\_dark.png'  
  
 def add\_transaction(event):  
 if theme == 'light':  
 main\_color = white\_color  
 sec\_color = black\_color  
 red\_color = red\_color\_day  
 if theme == 'dark':  
 main\_color = dark\_gray\_color  
 sec\_color = white\_color  
 red\_color = red\_color\_night  
  
 def get\_transaction():

Продолжение ПРИЛОЖЕНИЯ А

global bonuses  
 price = 0

add\_button['text'] = 'Добавить транзакцию'  
 date = calendar.get\_date()  
 try:  
 litres = int(litres\_entry.get())  
 except:  
 add\_button['text'] = 'Ошибка'  
 gas = gas\_choose.get()  
 if gas and date:  
 match gas:  
 case '92':  
 price = 53  
 case '95':  
 price = 57  
 case '100':  
 price = 82  
 case 'ДТ':  
 price = 68  
 summ = price\*litres  
 transactions.append({'Дата': date, 'Литры': litres, 'Бензин': gas, 'Сумма': summ, 'Бонусы': round(summ\*0.1, 2)})  
 bonuses += round(summ\*0.1, 2)  
 transactions\_list.insert('', END, values=(transactions[-1]['Дата'],  
 transactions[-1]['Литры'],  
 transactions[-1]['Бензин'],  
 transactions[-1]['Сумма'],  
 transactions[-1]['Бонусы'],))  
 else:  
 add\_button['text'] = 'Ошибка'  
  
  
 add\_window = Toplevel(exp\_window)  
 add\_window.title('Добавление транзакции')  
 add\_window.config(width=275, height=360, bg=main\_color)  
 add\_window.resizable(False, False)  
 add\_window.iconbitmap('logo.ico')  
  
 date\_label = Label(add\_window, text="Дата:", font=font16, bg=main\_color, fg=red\_color)  
 date\_label.place(x=10, y=5)  
 calendar = Calendar(add\_window,

Продолжение ПРИЛОЖЕНИЯ А

selectmode='day', year=2024, background=red\_color, foreground=white\_color,

selectbackground=red\_color, selectforeground=sec\_color)  
 calendar.place(x=10, y=30)  
  
 litres\_label = Label(add\_window, text="Литры:", font=font16, bg=main\_color, fg=red\_color)  
 litres\_label.place(x=10, y=210)  
 litres\_entry = Entry(add\_window, width=10, fg=black\_color, font=font17, justify='center',  
 relief='groove', highlightcolor=red\_color, highlightthickness=2, borderwidth=0,  
 selectbackground=black\_color, highlightbackground=red\_color)  
 litres\_entry.place(x=10, y=235)  
  
 gas\_label = Label(add\_window, text="Бензин:", font=font16, bg=main\_color, fg=red\_color)  
 gas\_label.place(x=185, y=210)  
 gas\_types = ['92', '95', '100', 'ДТ']  
 gas\_var = StringVar()  
 gas\_style = ttk.Style()  
 gas\_style.configure('TCombobox', selectbackground=white\_color, selectforeground=red\_color,  
 background=white\_color, foreground=red\_color, fieldbackground=white\_color,  
 darkcolor=red\_color,  
 lightcolor=white\_color)  
 gas\_choose = ttk.Combobox(add\_window, state='readonly', textvariable=gas\_var, font=font17b, values=gas\_types, width=4)  
 gas\_choose.place(x=185, y=235)  
  
 add\_button = Button(add\_window, text="Добавить транзакцию", font=font17b, width=20, bg=red\_color, fg=white\_color,  
 height=2, relief='flat', borderwidth=0, command=get\_transaction,  
 activebackground=red\_gray\_color, activeforeground=white\_color)  
 add\_button.place(x=12, y=280)  
  
 def add\_tr\_gray(event):  
 add\_transact\_lb['fg'] = red\_gray\_color  
 add\_image\_lb['image'] = add\_image\_gray  
 def add\_tr\_red(event):  
 add\_transact\_lb['fg'] = red\_color

Продолжение ПРИЛОЖЕНИЯ А

add\_image\_lb['image'] = add\_image  
  
 exp\_window = Toplevel(main\_window)

exp\_window.title('История транзакций')  
 exp\_window.config(width=500, height=500, bg=main\_color)  
 exp\_window.resizable(False, False)  
 exp\_window.iconbitmap('logo.ico')  
  
 user\_logo = Label(exp\_window, image=user\_logo\_img, bg=main\_color, highlightthickness=0)  
 user\_logo.place(x=186, y=0)  
 name\_lb = Label(exp\_window, text=user[0], width=38, justify='center', font=font20b, bg=black\_color, fg=white\_color)  
 name\_lb.place(x=0, y=135)  
  
 transactions\_lb = Label(exp\_window, text='История транзакций', font=font17, bg=main\_color, fg=red\_color)  
 transactions\_lb.place(x=5, y=175)  
 add\_image\_lb = Label(exp\_window, image=add\_image, bg=main\_color, activebackground=sec\_color)  
 add\_image\_lb.place(x=275, y=178)  
 add\_transact\_lb = Label(exp\_window, text='Добавить транзакцию', font=font14b, bg=main\_color, fg=red\_color)  
 add\_transact\_lb.place(x=303, y=178)  
 add\_transact\_lb.bind('<Enter>', add\_tr\_gray)  
 add\_transact\_lb.bind('<Leave>', add\_tr\_red)  
 add\_transact\_lb.bind('<Button>', add\_transaction)  
 add\_image\_lb.bind('<Enter>', add\_tr\_gray)  
 add\_image\_lb.bind('<Leave>', add\_tr\_red)  
 add\_image\_lb.bind('<Button>', add\_transaction)  
  
 headerstyle = ttk.Style()  
 headerstyle.configure('Treeview.Heading', font=font17b)  
 columnstyle = ttk.Style()  
 columnstyle.configure('Treeview', font=font16, background=main\_color, foreground=sec\_color, fieldbackground=main\_color)  
 columnstyle.map('Treeview', background=[('selected', red\_color)])  
 transactions\_list = ttk.Treeview(exp\_window, columns=('Дата', 'Литры', 'Бензин', 'Сумма', 'Бонусы'), show="headings", height=12)  
 transactions\_list.column('#1', anchor=CENTER, stretch=NO, width=100)

Продолжение ПРИЛОЖЕНИЯ А

transactions\_list.heading('#1', text='Дата')  
 transactions\_list.column('#2', anchor=CENTER, stretch=NO, width=95)

transactions\_list.heading('#2', text="Литры")  
 transactions\_list.column('#3', anchor=CENTER, stretch=NO, width=95)  
 transactions\_list.heading('#3', text="Бензин")  
 transactions\_list.column('#4', anchor=CENTER, stretch=NO, width=100)  
 transactions\_list.heading('#4', text="Сумма")  
 transactions\_list.column('#5', anchor=CENTER, stretch=NO, width=95)  
 transactions\_list.heading('#5', text="Бонусы")  
 for transaction in transactions:  
 transactions\_list.insert('', END, values=(transaction['Дата'],  
 transaction['Литры'],  
 transaction['Бензин'],  
 transaction['Сумма'],  
 transaction['Бонусы'],))  
 transactions\_list.place(x=7, y=210)  
  
 def bonuses\_window():  
 if theme == 'light':  
 main\_color = white\_color  
 sec\_color = black\_color  
 red\_color = red\_color\_day  
 coin\_img['file'] = 'coin\_light.png'  
 if theme == 'dark':  
 main\_color = dark\_gray\_color  
 sec\_color = white\_color  
 red\_color = red\_color\_night  
 coin\_img['file'] = 'coin\_dark.png'  
 bonus\_window = Toplevel(main\_window)  
 bonus\_window.title('Бонусы')  
 bonus\_window.config(width=500, height=500, bg=main\_color)  
 bonus\_window.resizable(False, False)  
 bonus\_window.iconbitmap('logo.ico')  
  
 user\_logo = Label(bonus\_window, image=user\_logo\_img, bg=main\_color, highlightthickness=0)  
 user\_logo.place(x=186, y=0)  
 name\_lb = Label(bonus\_window, text=user[0], width=38, justify='center', font=font20b, bg=black\_color,  
 fg=white\_color)

Продолжение ПРИЛОЖЕНИЯ А

name\_lb.place(x=0, y=135)  
  
 coin\_img\_lb = Label(bonus\_window,

image=coin\_img, bg=main\_color)  
 coin\_img\_lb.place(x=50, y=220)  
 user\_bonuses\_lb = Label(bonus\_window, text=f'Вы накопили \n{bonuses} бонусов', fg=red\_color, bg=main\_color, font=font17b)  
 user\_bonuses\_lb.place(x=50, y=370)  
  
 promocodes = ['LUKOIL-DRIVE20: \nСкидка 20% на топливо \nпри заправке от 1000 рублей',  
 'LUKOIL-BONUS100: \n100 бонусных баллов \nпри первой заправке.',  
 'LUKOIL-WEEKEND5: \nСкидка 5% на все товары \nв магазинах сети "Лукойл" \nпо выходным дням.']  
 y\_pos = 185  
 for promocode in promocodes:  
 promolabel = Label(bonus\_window, text=promocode, width=25, font=font16, bg=main\_color, fg=sec\_color,  
 relief='solid', highlightthickness=2, borderwidth=0, highlightbackground=red\_color)  
 promolabel.place(x=225, y=y\_pos)  
 y\_pos += 100  
  
 welcome\_lb['text'] = user[0]  
 acc\_window.expenses\_img = PhotoImage(file='expenses\_light.png')  
 acc\_window.bonuses\_img = PhotoImage(file='bonuses\_dark.png')  
  
 if theme == 'light':  
 main\_color = white\_color  
 sec\_color = black\_color  
 red\_color = red\_color\_day  
 acc\_window.expenses\_img['file'] = 'expenses\_light.png'  
 acc\_window.bonuses\_img['file'] = 'bonuses\_light.png'  
 if theme == 'dark':  
 main\_color = dark\_gray\_color  
 sec\_color = white\_color  
 red\_color = red\_color\_night  
 acc\_window.expenses\_img['file'] = 'expenses\_dark.png'  
 acc\_window.bonuses\_img['file'] = 'bonuses\_dark.png'

Продолжение ПРИЛОЖЕНИЯ А

expenses\_img\_lb = Button(acc\_window, image=acc\_window.expenses\_img, bg=main\_color, relief='flat',  
 borderwidth=0,

activebackground=main\_color, command=expenses\_window)  
 expenses\_img\_lb.place(x=54, y=225)  
 expenses\_lb = Label(acc\_window, text='Транзакции', font=font17, fg=red\_color, bg=main\_color, justify='center',  
 width=11)  
 expenses\_lb.place(x=54, y=365)  
 bonuses\_img\_lb = Button(acc\_window, image=acc\_window.bonuses\_img, bg=main\_color, relief='flat',  
 borderwidth=0, activebackground=main\_color, command=bonuses\_window)  
 bonuses\_img\_lb.place(x=219, y=225)  
 bonuses\_lb = Label(acc\_window, text='Бонусы и скидки', font=font17, fg=red\_color, bg=main\_color,  
 width=13)  
 bonuses\_lb.place(x=209, y=365)  
  
 log\_out\_button = Button(acc\_window, text='Выйти', bg=main\_color, fg=sec\_color, font=font16,  
 relief='flat', borderwidth=0, activebackground=main\_color, command=log\_out)  
 log\_out\_button.place(x=170, y=430)  
  
 if authorise\_status == False:  
 unauthorised()  
 if authorise\_status == True:  
 authorised()  
  
account\_button = Button(image=account\_image\_white, bg=red\_color,  
 borderwidth=0, highlightthickness=0, activebackground=red\_color,  
 command=account\_window)  
account\_button.place(x=1245, y=9)  
account\_button.bind('<Enter>', account\_gray)  
account\_button.bind('<Leave>', account\_white)  
  
main\_window.mainloop()