

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Вятский государственный университет»
(ВятГУ)

ОТЧЕТ
ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Кудяшев Ярослав Юрьевич
(Ф.И.О. обучающегося)

09.01.01 Информатика и вычислительная техника. Программное и
аппаратное обеспечение вычислительной техники
(направление подготовки (специальность), направленность (профиль))

Место прохождения практики ООО "ЭЛМА Вятка"
(наименование организации, структурного подразделения организации)

Итоговая оценка:

Зачислено

Руководитель

практики от университета

23.04.2022
(дата)

[Подпись]
(подпись)

Караваева О.В.
(Ф.И.О.)

Киров, 2022 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	3
1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ	4
1.1 Сведения о работе, выполненной в период прохождения учебной практики	4
1.2 Организационная структура предприятия	4
2. ФОРМУЛИРОВКА ИНДИВИДУАЛЬНОГО ЗАДАНИЯ	6
3. ОПИСАНИЕ ВЫПОЛНЕНИЯ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ	9
3.1 Анализ предметной области и обзор аналогов.....	9
3.2 Разработка структур, алгоритмов, интерфейсных решений, в соответствии с индивидуальным заданием.....	10
3.3 Программная реализация в соответствии с индивидуальным заданием	12
3.4 Экспериментальная апробация в соответствии с индивидуальным заданием	13
ЗАКЛЮЧЕНИЕ.....	14
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	15
ПРИЛОЖЕНИЯ.....	16
Приложение А.....	16
Приложение Б.....	21

ВВЕДЕНИЕ

На сегодняшний день многим коммерческим организациям требуется новое и современное решение оптимизации работы компании. Если работа налажена и хорошо оптимизирована, рабочий процесс проходит быстрее и комфортнее. Каждый работник будет связан одной системой, в которой приходят задачи конкретным работникам (менеджерам, бухгалтерам), группам людей или отделам. При выполнении текущей задачи работа либо завершается, либо последовательно ставятся другие задачи сотрудникам. Возможности разработки таких систем полностью встроены в компании. Например, могут быть налажены процессы выхода сотрудника на обед, процесс командировки сотрудника и другие.

В данный момент есть много компаний, занимающиеся построением бизнес-процессов и переходам компаниями на цифровые платформы. Такие системы помогают автоматизировать бизнес-процессы. Так же они могут роботизировать процессы, то есть снижать затраты на выполнение рутинных операций, благодаря чему ускоряется ход выполнения любых задач.

1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ

1.1 Сведения о работе, выполненной в период прохождения учебной практики

Таблица 1 – Сведения о работе, выполненной в период практики

Дата	Краткое содержание выполненных работ
27.06.2022	Пройти инструктаж по ознакомлению с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также вводный инструктаж и инструктаж на рабочем месте
27.06.2022-01.07.2022	Прохождение курсов по настройке приложения, изучение основ
01.07.2022-8.07.2022	Моделирование бизнес-процесса движения Кредитной заявки в небольшом финансовом учреждении, добавление динамики на формы, при этом учесть требования изложенные в техническом задании
08.07.2022-15.07.2022	В разработанном бизнес-процессе выполнить доработки новых форм, следуя техническому заданию
15.07.2022-21.07.2022	Доработка новых заданий с помощью «скриптов»
21.07.2022-23.07.2022	Подготовить и оформить отчёт

(дата)

(подпись)

1.2 Организационная структура предприятия

Организационную структуру практики можно описать как на Рисунке 1. Главным является основатель, он имеет связь с вице-президентом, директором ELMA, коммерческим директором ELMA, директором по маркетингу и HR ELMA. Директор связан с офис-менеджером HR, low-code специалистами и с архитекторами решений.

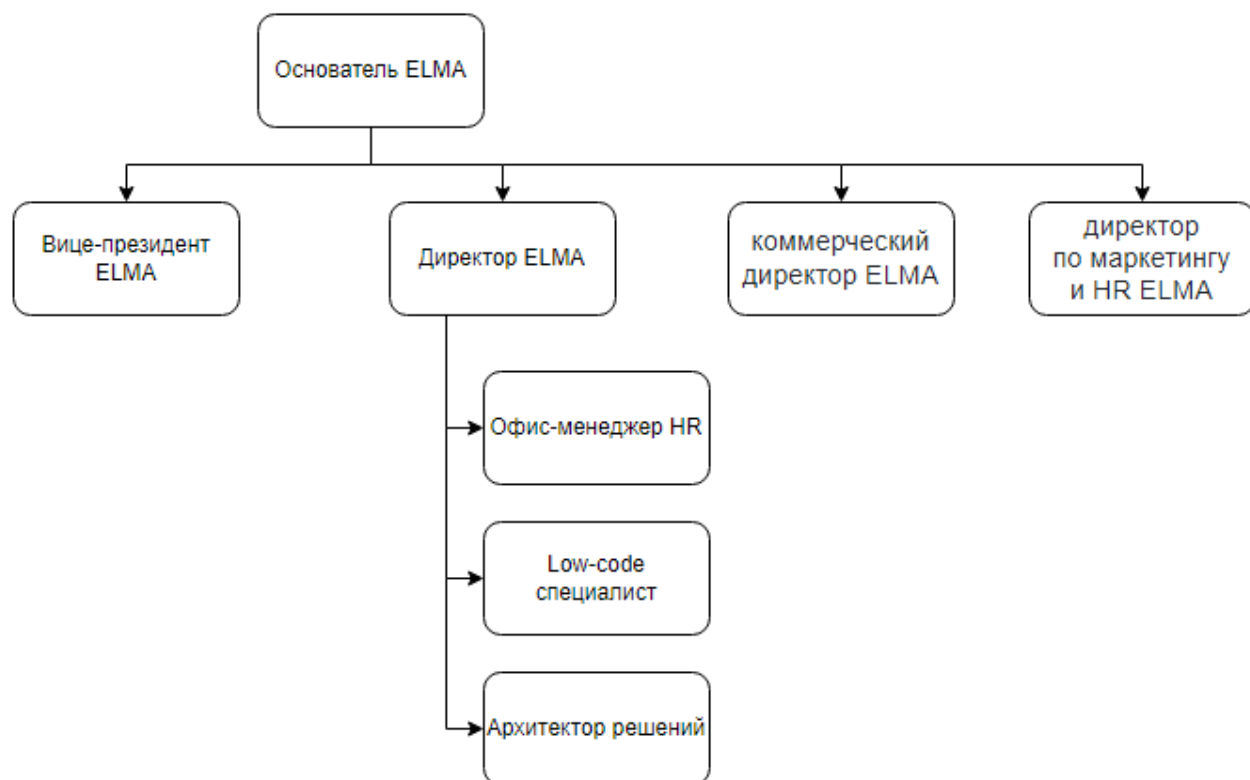


Рисунок 1 – Организационная структура предприятия

2. ФОРМУЛИРОВКА ИНДИВИДУАЛЬНОГО ЗАДАНИЯ

Содержание индивидуального задания:

1. Смоделировать бизнес-процесс движения Кредитной Заявки в небольшом финансовом учреждении. Требования к логике процесса:
 - 1.1. Запустить процесс может любой сотрудник отдела продаж, указав клиента из справочника с клиентами (можно использовать справочник из раздела CRM либо создать свой справочник - выбрать более подходящий вариант и обосновать свой выбор). Также следует указать сумму, информацию о цели кредита и залоге (добавить другие поля, отражающие важную информацию по кредитной заявке).
 - 1.2. Если сумма заявки:
 - 1.2.1. Меньше 100 тыс. р. – заявка поступает на согласование начальнику кредитного отдела.
 - 1.2.2. Меньше 1 млн. р. – заявка поступает на согласование начальнику кредитного отдела и группе безопасности (у любого сотрудника группы должна быть возможность выполнить задачу).
 - 1.2.3. Свыше 1 млн. р. – согласование направляется всем вышеперечисленным и начальнику кредитного комитета. Начальник должен организовать совещание кредитного комитета, создав событие и выполнив задачу. По окончании совещания Начальнику кредитного комитета должна прийти задача отражения результатов работы комитета. Необходимо реализовать автоматическое создание события встречи кредитного комитета.
 - 1.3. По окончании согласования – поставить сотруднику отдела продаж задачу уведомления клиента.
 - 1.4. Передать заявку на выплату к бухгалтеру.
 - 1.5. Отобразить информацию о кредитной истории в карточке клиента.
2. Добавить динамику на формах.
 - 2.1. Сумма ежемесячного платежа и сумма переплат по кредиту рассчитываются автоматически.
 - 2.2. Добавить отображение-скрытие полей в зависимости от значений в поле, например, при выборе типа кредита «Автокредит» у нас отображается поле «Тип транспорта» (изначально поле не отображается), при выборе типа кредита «Ипотека» отображается

поле «Тип недвижимости» (изначально поле не отображается).
Можно придумать свои поля.

2.3. Добавить фильтрацию типов кредитов в зависимости от юридической формы заявителя. Например, заявитель – юр. лицо – при выборе типа кредита отображаются только те, которые относятся к юр. лицам (лизинг, бизнес и т.д.); заявитель – физ. лицо – типы кредита для физ. лиц (потребительский, ипотека, автокредит и т.д.).

3. В задаче обязательно предусмотреть выполнение следующих пунктов:

3.1. Статусы приложения, смена статуса по ходу бизнес-процесса.

3.2. Согласование кредитной заявки проходит последовательно (сначала согласует Начальник кредитного отдела, затем Сотрудник группы безопасности, затем Начальник кредитного комитета).

3.3. Сроки выполнения задач.

3.4. Добавить контекстные переменные: тип кредита, ставка, срок, ежемесячный платеж и сумма переплат.

3.5. В зоне ответственности **Группы безопасности** нужно применить механизм «Кто первый».

3.6. Задать название элемента приложения по шаблону.

3.7. Название задач по шаблону.

3.8. Кредитный комитет - в зоне ответственности **Начальника кредитного комитета** 2 задачи:

- «Организовать кредитный комитет» - создание элемента приложения типа Событие.
- Вынести решение по кредиту».

Предполагается, что комитет прошел очно, далее по результатам совещания **Начальник кредитного комитета** выносит окончательное решение.

3.9. В задаче «Организовать кредитный комитет» надо создать и вынести переменные для совещания (Тема (строка), Место (строка), Участники (пользователь, связь многие ко многим), Дата начала (дата/время), Дата окончания (дата/время), Описание (текст)), ссылка на кредитную заявку.

3.10. Для каждого согласующего добавить переменные с комментариями. Сделать обязательными для заполнения в случае отказа.

3.11. Организовать хранение кредитной истории в карточке Заемщика с помощью виджета «Связанные элементы».

3.12. Генерация документа по шаблону (любого на выбор аналитика, выполняющего задание).

- 3.13. Эскалация
- 3.14. Проставить сроки выполнения задач.

Скриптовые задания по процессу

Задание №1

Начальник кредитного комитета организует совещание кредитного комитета. В этой же задаче необходимо ему дать возможность добавить поручения своим подчиненным. Поручения он вносит в отдельную таблицу, которая находится на форме задачи организации совещания. После выполнения задачи организации совещания система по каждой строке создает поручение.

Рекомендации к выполнению задания:

1. Для задачи организации совещания использовать блок Задача. Вынести туда поля для создания совещания. Совещание создавать автоматически с помощью блока «Создание элемента».
2. Добавить в контекст процесса таблицу. Строка таблицы – это отдельное поручение.
3. С помощью сценария (блок Сценарий) получить данные из таблицы (из строки таблицы), записать это в переменные процесса. Полученные данные передать из основного процесса в подпроцесс.
4. Использовать шлюз, в котором проверяется, из всех ли строк была передана информация в подпроцессы. Как только все строки проверены, выходить из цикла.

Задание №2


Менеджер по качеству периодически проводит анализ кредитных заявок, по которым отказался сам клиент после этапа согласования. 1 числа каждого месяца в 10:00 он проходит по списку отказанных заявок за предыдущий месяц и отбирает те, которые требуют отдельного рассмотрения. Выбранные заявки отправляются на рассмотрение руководителю отдела качества. Руководитель отдела качества рассматривает заявки и принимает по ним решение. Заявки, не требующие отдельного рассмотрения, попадают в архив.

Реализовать данные требования в разделе с кредитными заявками.

3. ОПИСАНИЕ ВЫПОЛНЕНИЯ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ

3.1 Анализ предметной области и обзор аналогов

На сегодняшний день есть много вариантов заполнения кредитной заявки. Практически у каждого банка есть собственный сайт или страница заполнения кредитной заявки. Содержание какой-либо заявки должно иметь определенные поля для заполнения, так же могут содержаться дополнительные поля, которые не обязательны для заполнения. Каждая организация сама решает добавлять или нет. Примером для кредитной заявки может быть онлайн заявка на кредит Сбера показанного на рисунке 2. Так же на рисунке 3 показаны примеры заявки в электронном виде.

 **СБЕРБАНК**
Всегда рядом

НЕ ЯВЛЯЕТСЯ ДОГОВОРом

Регистрационный номер: _____

от «___» _____ г.

Заявление – анкета*

1. Запрашиваемый кредит

Сумма	Срок (мес.)	Вид кредитования	Цель кредитования / предполагаемая цель кредитования	Способ погашения кредита Аннуитетные платежи	Периодичность погашения кредита Ежемесячно
В качестве обеспечения предлагаю:					
<input type="checkbox"/> Поручительство физического лица					
<input type="checkbox"/> Поручительство юридического лица					
<input type="checkbox"/> Гарантия субъекта Российской Федерации					
<input type="checkbox"/> Гарантия муниципального образования					
<input type="checkbox"/> Залог недвижимого имущества					
<input type="checkbox"/> Залог транспортных средств					
<input type="checkbox"/> Залог мерных слитков драгоценных металлов					
<input type="checkbox"/> Залог ценных бумаг					
<input type="checkbox"/> Залог иного имущества					
Начальный капитал, направляемый Заемщиком в качестве оплаты части стоимости приобретаемого за счет кредита имущества:					

2. Сведения о Заемщике

Ф.И.О.		Образование	
Менялись ли Ф.И.О.		В случае их изменения указать предыдущие Ф.И.О. с указанием причины и даты изменения:	
<input type="checkbox"/> Да			
<input type="checkbox"/> Нет			
Представлен паспорт:			
серия [] [] [] [] № [] [] [] [] [] []			
Кем выдан:			
Дата выдачи: [] [] [] [] [] []			
Код подразделения [] [] [] [] [] []			
Семейное положение:		Иждивенцы	
<input type="checkbox"/> Холост / не замужем		кол-во	
<input type="checkbox"/> Женат / замужем		их возраст	
<input type="checkbox"/> Разведен / Разведена		<input type="checkbox"/> Вдовец / Вдова	
Из них детей		Из них детей	
кол-во		кол-во	
их возраст		их возраст	
Адрес регистрации: [] [] [] [] [] []		телефон (вкл. код)	
Адрес проживания: [] [] [] [] [] []		телефон (вкл. код)	
<input type="checkbox"/> Собственное		Время проживания в населенном пункте на момент заполнения анкеты	
<input type="checkbox"/> По найму			
<input type="checkbox"/> У родственников			
<input type="checkbox"/> Иное			
мобильный телефон		E-mail	
Предпочтительный вид контакта		<input type="checkbox"/> Мобильный телефон	
		<input type="checkbox"/> Городской телефон	
		<input type="checkbox"/> E-mail	

* Если количество Созаемщиков и/или Поручителей (физических лиц) больше предусмотренного в заявлении, ее необходимо дополнить соответствующими листами, разделами и графами. Если Созаемщик и/или Поручители по кредиту отсутствуют, а предоставление информации по Разделу 4 не требуется, соответствующие листы могут не предоставляться. При этом изменение нумерации разделов не требуется.

Рисунок 2 – Пример кредитной заявки с сайта www.cbkg.ru

Заявка на получение кредита

Статус документа: создан

Данные о заявителе

Форма организации Наименование

Место нахождение бизнеса

Город

Улица

Дом Корпус:

Вид деятельности Время работы в данном направлении деятельности

Регистрационные данные

ОКВЭД ИНН

ОГРН

№ свидетельства Дата регистрации

Среднемесячный оборот по рас/сч (в руб.)

Средняя стоимость РКО др. банка (з мес.)

Наличие среднемесячных остатков на рас/счёте

Опыт использования банковских продуктов

Наличие филиалов/представительств ☐ да (укажите город) ☐ нет

Опыт получения кредитов в ВТБ 24 (ЗАО) ☐ Опыт есть ☐ Опыта нет

Опыт получения кредитов в других банках ☐ Опыта нет

Банк	Дата получения	Дата возврата по договору	Дата возврата по графику	% ставка	Сумма
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Наличие банковских счетов и осудной задолжности в других банках

Банк	Номер счёта	Осудная задолжность (при наличии)
<input type="text"/>	<input type="text"/>	<input type="text"/>

Рисунок 3 – Пример калькулятора калорий с сайта calc.ru

3.2 Разработка структур, алгоритмов, интерфейсных решений, в соответствии с индивидуальным заданием

Заявка в зависимости от суммы кредита должна быть одобрена разным набором сотрудников. Поэтому на рисунке 4 представлена диаграмма получения задач сотрудниками банка

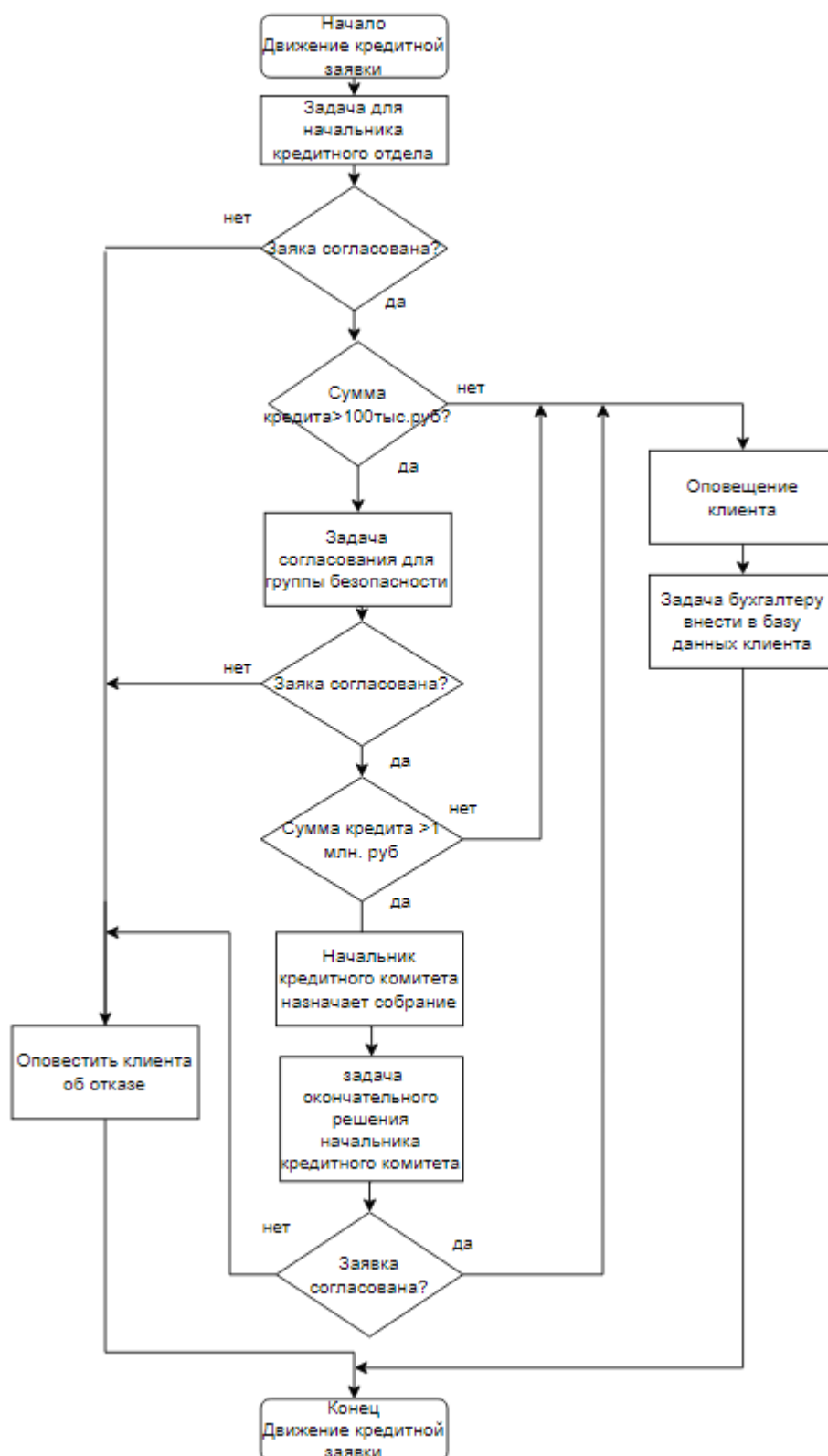


Рисунок 4 – Схема движения кредитной заявки

Некоторые поля могут быть доступны только при выполнении некоторых условий или выбора конкретных полей. Например, при выборе типа кредита Автокредит, автоматически отображается поле с типом транспорта. Для этого можно было использовать как встроенные в систему методы, которые бы не требовали написания скриптов, которые связывали определенные поля, но для

дальнейшей работы и удобства были написаны небольшие программы для отображения полей. Пример схемы алгоритма отображения полей изображен на рисунке 5.

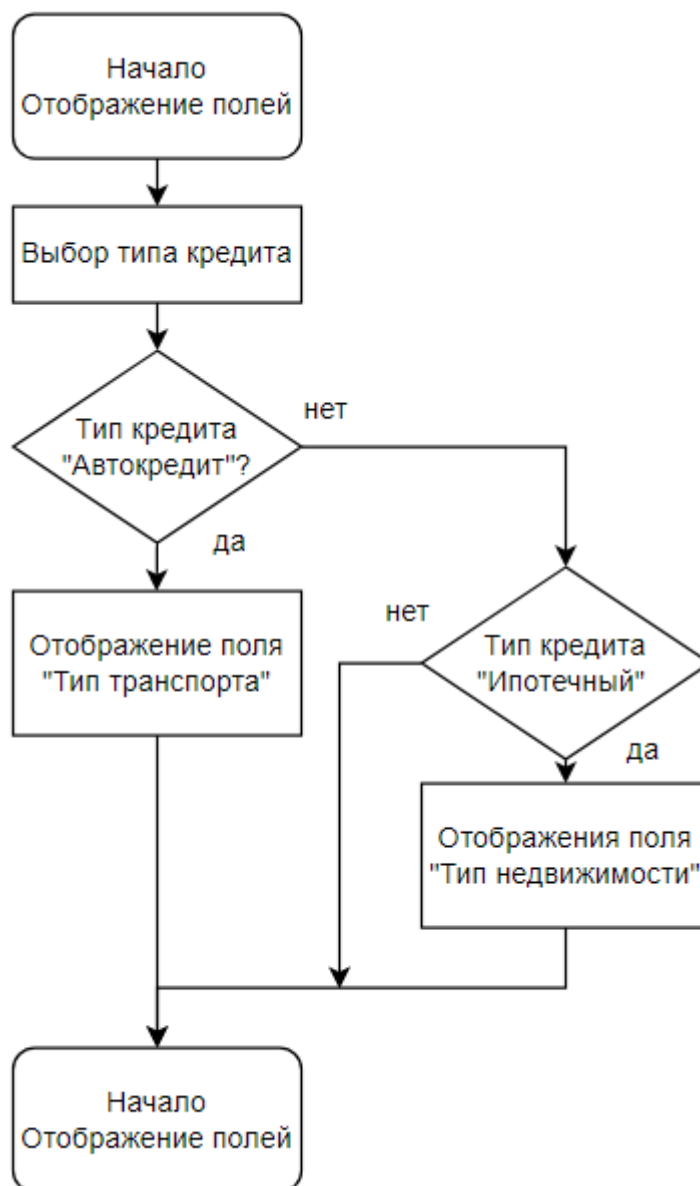


Рисунок 5 - Схема алгоритма проверки и отображения специальных полей

Так же при помощи небольших программ, называемых «скриптами» или же сценариями, были произведены расчеты ежемесячной платы за кредит, изменения процентной ставки в зависимости от типа кредита, сумма переплаты. Так же производилось изъятие курса валют с официального сайта Сбербанка для случаев, если клиент берет кредит в иностранной валюте.

3.3 Программная реализация в соответствии с индивидуальным заданием

Код сценариев программы представлен в Приложении А, который содержит реализацию интерфейса и алгоритмов в соответствии с заданием. Экранные формы помещены в Приложение Б.

3.4 Экспериментальная апробация в соответствии с индивидуальным заданием

После написания программного обеспечения было проведено её тестирование эффективности вместе с кураторами практики. В ходе которой были найдены и исправлены допущенные ошибки. Реализованы новые идеи для удобства как работника, так и клиента банка. Полная проверка задания производилась раз в неделю.

ЗАКЛЮЧЕНИЕ

В ходе производственной практики были получены практические знания и навыки построения бизнес-процессов, было изучено приложение для составления процессов, построения форм, передач обязанностей и задач, которые нужно выполнить определенном порядке в зависимости от условий, работникам. Для выполнения внутренних действий и вычислений был изучен язык TS SDK, который как раз предназначен для выполнения небольших задач.

Разработанное приложение полностью функционирует и удовлетворяет условиям технического задания. С помощью такого приложения задачи будет приходить последовательно в зависимости от условий, что сильно позволит упростить рабочий процесс, а именно ускорить и автоматизировать его. Ненадежные бумажные носители будет заменены цифровым вариантом, которым хранится на нескольких базах данных для сохранения информации. Данное приложение можно улучшить или изменить в любое время по просьбе заказчика.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. tssdk.elma365.com[Электронный ресурс]. – Режим доступа:
<https://tssdk.elma365.com/ru/index.html>
2. elma-academy.com [Электронный ресурс]. – Режим доступа:
<https://elma-academy.com/ru/elma365/CoursePassage/Wizard?id=24ebbb27-ee65-4d79-9327-e1edad54ca05>

ПРИЛОЖЕНИЯ

Приложение А

Сценарии

```
/* Client scripts module */

let PersentFlagDate = true;
let PersentFlagSum = true;
let FirstTime = false
let numberOfMonths = 3
var arrayDate = new Array()

/*Создание новой строки в таблице */
async function FillingInTheTable(remains: number, summa: number, persent: number):
Promise<void> {
    if (Context.data.CreditSum) {
        for (var i = 0; i < parseInt(Context.data.CreditTime!.name); i++) {
            const row = Context.data.MonthlyPayment!.insert()
            row.Month = String(i + 1) + " месяц";
            row.RemainingDebt = new Money(summa, "RUB");
            row.MainDebt = new Money(remains, "RUB");
            row.Persents = new Money(summa * (persent * 0.01), "RUB");
            row.Payment = new Money(summa * (persent * 0.01) + remains, "RUB");
            summa = summa - remains;
            Context.data.MonthlyPayment = Context.data.MonthlyPayment
        }
    }
}

/*Замена всех значений */
async function DataChange(): Promise<void> {
    if (Context.data.CreditTime && Context.data.Percent) {
        const persent = Context.data.Percent;
        const months = Context.data.CreditTime.name;
        for (var i = 0; i < numberOfMonths; i++) {
            Context.data.MonthlyPayment!.delete(0)
        }
        numberOfMonths = parseInt(Context.data.CreditTime.name)
        if (Context.data.Currency!.name == "Рубль") {
            const summa = Context.data.CreditSum!.cents * 0.01;
            const currentMoney = (persent * 0.01 + 1) * summa;
            const remains = summa / parseInt(months);
            Context.data.TotalAmount = new Money(currentMoney, 'RUB');
            Context.data.Overpayment = new Money(currentMoney - summa, 'RUB');
            Context.data.MonthlyPay = new Money(currentMoney / parseInt(months),
'RUB');
            FillingInTheTable(remains, summa, persent);
        } else if (Context.data.Currency!.name == "Доллар") {
            const summa = Context.data.dollar!.cents * 0.01;
            const currentMoney = (persent * 0.01 + 1) * summa;
```



```

        const remains = summa / parseInt(months);
        Context.data.TotalSumDollar = new Money(currentMoney, 'USD');
        Context.data.OverpaymentDollar = new Money(currentMoney - summa,
'USD');
        Context.data.MonthlyPaymentDollar = new Money(currentMoney /
parseInt(months), 'USD');
        FillingInTheTable(remains, summa ,persent);
    } else if (Context.data.Currency!.name == "Евро") {
        const summa = Context.data.evro!.cents * 0.01;
        const currentMoney = (persent * 0.01 + 1) * summa;
        const remains = summa / parseInt(months);
        Context.data.TotalAmountEuro = new Money(currentMoney, 'EUR');
        Context.data.OverpaymentEuro = new Money(currentMoney - summa, 'EUR');
        Context.data.MonthlyPaymentEuro = new Money(currentMoney /
parseInt(months), 'EUR');
        FillingInTheTable(remains, summa, persent);
    }
}
}

/*Пенсионер ты или нет */
async function onRetireeChange(): Promise<void> {
    if (Context.data.Percent) {
        if (Context.data.Grandparent) {
            Context.data.Percent = Context.data.Percent - 2.0;
            DataChange();
        } else {
            Context.data.Percent = Context.data.Percent + 2.0;
            DataChange();
        }
    }
}

/*Смена суммы для взятия кредита (РУБЛИ) */
async function onSumChange(): Promise<void> {
    if (Context.data.Percent && Context.data.Currency!.name == "Рубль") {
        Context.data.ErrorsInformation = undefined;
        if (Context.data.CreditSum!.cents * 0.01 >= 10000 &&
Context.data.CreditSum!.cents * 0.01 <= 5000000) {
            if (Context.data.CreditSum!.cents * 0.01 < 300000 && PersentFlagSum ==
true) {
                Context.data.Percent = Context.data.Percent + 4.0;
                PersentFlagSum = false;
            } else if (Context.data.CreditSum!.cents * 0.01 >= 300000 &&
PersentFlagSum == false) {
                Context.data.Percent = Context.data.Percent - 4.0;
                PersentFlagSum = true;
            }
            DataChange();
        } else {

```

```

        Context.data.ErrorsInformation = "Сумма должна быть не менее 10 тыс.
руб. и не более 5 млн. руб.";
    }
}

/*Смена суммы для взятия кредита (ДОЛЛАР) */
async function onSumChangeDollar(): Promise<void> {
    if (Context.data.Percent && Context.data.Currency!.name == "Доллар") {
        Context.data.ErrorsInformation = undefined;
        //const dollarSum = Context.data.Dollar.cents * 0.01 *
        if (Context.data.dollar!.cents * 0.01 >= 200 && Context.data.dollar!.cents
* 0.01 <= 100000) {
            if (Context.data.dollar!.cents * 0.01 < 5000 && PersentFlagSum ==
true) {
                Context.data.Percent = Context.data.Percent + 4.0;
                PersentFlagSum = false;
            } else if (Context.data.dollar!.cents * 0.01 >= 50000 &&
PersentFlagSum == false) {
                Context.data.Percent = Context.data.Percent - 4.0;
                PersentFlagSum = true;
            }
            DataChange();
        } else {
            Context.data.ErrorsInformation = "Сумма должна быть не менее 200$ и не
более 100 тыс.$";
        }
    }
}

/*Смена суммы для взятия кредита (ЕВРО) */
async function onSumChangeEuro(): Promise<void> {
    if (Context.data.Percent && Context.data.Currency!.name == "Евро") {
        Context.data.ErrorsInformation = undefined;
        //const dollarSum = Context.data.Dollar.cents * 0.01 *
        if (Context.data.evro!.cents * 0.01 >= 200 && Context.data.evro!.cents *
0.01 <= 100000) {
            if (Context.data.evro!.cents * 0.01 < 5000 && PersentFlagSum == true)
{
                Context.data.Percent = Context.data.Percent + 4.0;
                PersentFlagSum = false;
            } else if (Context.data.evro!.cents * 0.01 >= 50000 && PersentFlagSum
== false) {
                Context.data.Percent = Context.data.Percent - 4.0;
                PersentFlagSum = true;
            }
            DataChange();
        } else {
            Context.data.ErrorsInformation = "Сумма должна быть не менее 200€ и не
более 100 тыс.€";
        }
    }
}

```

```

    }
}

/*Смена типа физического и юридического кредитования*/
async function onFizClientChange(): Promise<void> {
    if (Context.data.CreditTypeFiz) {
        const TypeOfCredit = Context.data.CreditTypeFiz.name;
        switch (TypeOfCredit) {
            case "Автокредит":
                Context.data.InfoAboutTypeOfCredit = "Автокредит – это деньги,
которые вы занимаете у банка с целью покупки нового или поддержанного автомобиля.
Его особенностью является залог, которым выступает приобретаемая машина. Это даёт
возможность купить машину с первоначальным взносом или вообще без него.";
                break;
            case "Ипотека":
                Context.data.InfoAboutTypeOfCredit = "Ипотека – это целевой займ.
Он оформляется на крупную сумму, поэтому чаще всего ее оформляют при необходимости
приобретения дорогостоящих товаров: недвижимость, автомобиль, оплата обучения,
лечение, предметы роскоши.";
                break;
            case "Потребительский":
                Context.data.InfoAboutTypeOfCredit = "Потребительский кредит – это
кредит, предоставляемый банком на приобретение товаров (работ, услуг) для личных,
бытовых и иных непроизводственных нужд.";
                break;
            case "Обычный":
                Context.data.InfoAboutTypeOfCredit = "Обычный кредит – это кредит,
предоставляемый банком организации на приобретение товаров (работ, услуг) для
личных, бытовых и иных непроизводственных нужд.";
                break;
            case "Лизинг":
                Context.data.InfoAboutTypeOfCredit = "Лизинг – это вид финансовых
услуг, суть которого заключается в финансировании приобретения основных фондов
посредством приобретения компанией-лизингодателем имущества у поставщика и
передачи его в долгосрочную аренду клиенту-лизингополучателю (юридическому лицу) с
последующим выкупом.";
                break;
            case "Факторинг":
                Context.data.InfoAboutTypeOfCredit = "Факторинг – это финансовая
услуга для компаний, работающих на условиях отсрочки платежа. С её помощью
продавец получает деньги за товар сразу после отгрузки, может устранить кассовые
разрывы и пополнить оборотный капитал.";
                break;
        }
    }
}

/*Смена количества месяцев кредитования */
async function onDateChange(): Promise<void> {
    if (Context.data.CreditTime && Context.data.Percent) {

```

```

        if (parseInt(Context.data.CreditTime.name) < 12 && PersentFlagDate ==
true) {
            PersentFlagDate = false;
            Context.data.Percent = Context.data.Percent + 5.0;
        } else if (parseInt(Context.data.CreditTime.name) >= 12 && PersentFlagDate
== false) {
            PersentFlagDate = true;
            Context.data.Percent = Context.data.Percent - 5.0;
        }
        DataChange();
        FirstTime = true
    }
}

```

```

// "2022-07-01", "2022-07-05",
// startingYear:number, startingMonth:number, startingDay:number, endYear:number,
endMonth:number, endDay:number
async function CurrencyArray() {
    // const sDate = new Datetime()
    let valute = (Context.data.DollarOrEuro == true) ? 'USD' : 'EUR'
    let startingDate = Context.data.startingDateCurse!.asDatetime(new
TTime(0,0,0,0))
    // const eDate = new Datetime()
    const endDate = Context.data.endDateCurse!.asDatetime(new TTime(0,0,0,0))
    const duration = Math.floor(endDate!.sub(startingDate!).hours / 24)
    let moneyArray = new Array //Массив для курса валюты за промежутки времени
    let counter = 0
    for (let i = 0; i < duration; i++) {
        let month = startingDate!.month < 10 ? ('0' + String(startingDate!.month))
: String(startingDate!.month)
        let day = startingDate!.day < 10 ? ('0' + String(startingDate!.day)) :
String(startingDate!.day)
        const res = await fetch("https://www.cbr-xml-daily.ru//archive//" +
String(startingDate!.year) + "/" + month + "/" + day + "//daily_json.js")
        if (res.ok) {
            arrayDate[counter] = String(startingDate!.year) + '-' +
String(startingDate!.month) + '-' + String(startingDate!.day)
            const cbrData = await res.json();
            const price = await cbrData["Valute"][valute]["Value"] as number;
            moneyArray[counter] = String(price)
            counter++
        }
        startingDate = startingDate!.addDate(0, 0, 1)
    }
    return moneyArray.concat(arrayDate)
}

```

```

async function onInit() {
    Context.data.startingDateCurse = Context.data.endDateCurse!.addDate(0,0,-7)
    Context.data.PreviousDate = Context.data.CurrentDate!.addDate(0, 0, -1)
}

```

}

Экранные формы

Рисунок 6 – форма кредитной заявки

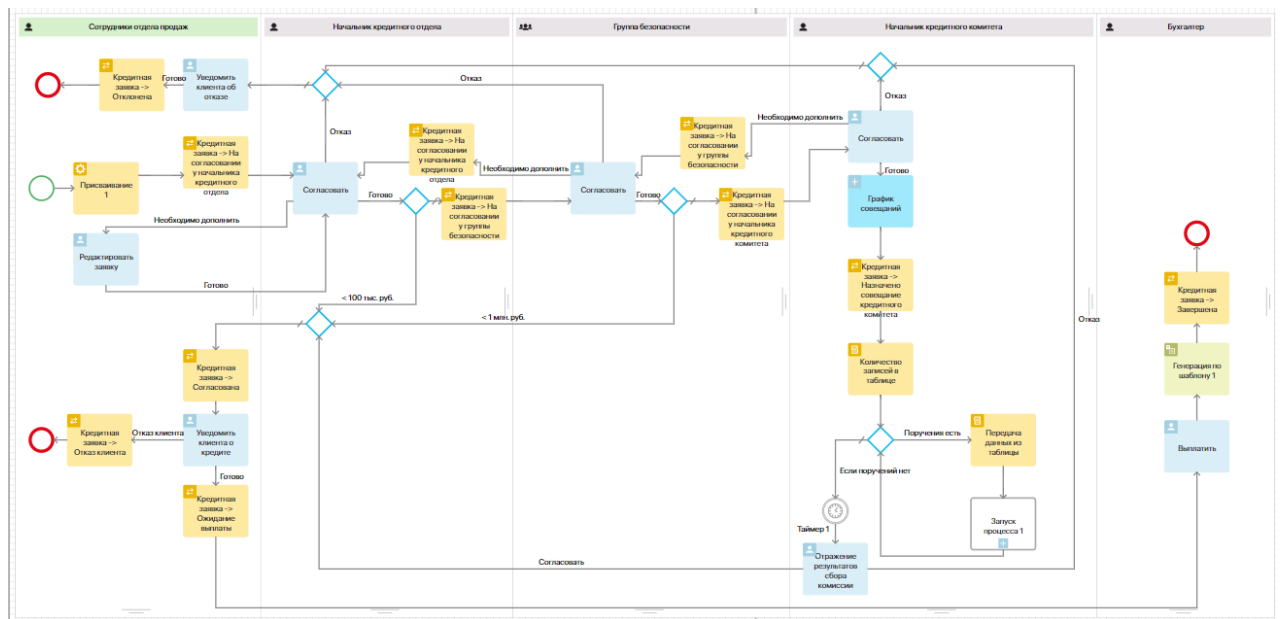


Рисунок 7 – составленный бизнес-процесс