***Задание1 – Расписать ТЗ по вашей теме***

**Техническое задание на разработку программного обеспечения для приема заявок типографии "СуперКраски52"**

Название компании: Типография "СуперКраски52"

Тип бизнеса: Типографские услуги (печать визиток, листовок, буклетов, плакатов, и др.)

Целевая аудитория:

Клиенты  — физические и юридические лица, нуждающиеся в полиграфических услугах.

Внутренние пользователи — менеджеры по приему заказов типографии.

Цель: Разработать программное обеспечение для автоматизации приема заказов, обеспечивающее удобство для клиентов и повышение эффективности работы сотрудников типографии.

Общие требования:

1. Система должна быть интуитивно понятной и удобной как для клиентов, так и для сотрудников.
2. Система должна быть надежной и обеспечивать сохранность данных.
3. Система должна интегрироваться с существующими системами
4. Система должна быть масштабируемой и адаптируемой к изменению потребностей типографии.
5. Программное обеспечение должно быть кроссплатформенным

Основные требования:

Функциональность для клиентов:

1. Онлайн-калькулятор стоимости печати с учетом тиража, формата, типа бумаги и дополнительных услуг.
2. Загрузка файлов макетов (поддержка распространенных форматов: PDF, AI, PSD, JPG, TIFF).
3. Выбор параметров печати (тип бумаги, цветность, послепечатная обработка).
4. Онлайн-оплата заказа (интеграция с платежными системами).
5. Отслеживание статуса заказа.
6. Возможность связи с менеджером.

Функциональность для сотрудников:

1. Прием и обработка заказов.
2. Проверка макетов на соответствие техническим требованиям.
3. Формирование счетов и актов.
4. Управление статусами заказов.
5. Генерация отчетов по продажам.
6. Учет материалов и ресурсов.
7. Интеграция с производственным календарем.

Технические требования:

1. Разработка веб-приложения (адаптивное отображение под различные устройства).
2. Использование современной технологической стеки (React, Node.js, PostgreSQL).
3. Обеспечение безопасности данных (шифрование, защита от SQL-инъекций и других уязвимостей).
4. Регулярное резервное копирование данных.
5. Доступ к системе через логин и пароль с системой ролей и прав доступа.

Сроки выполнения:

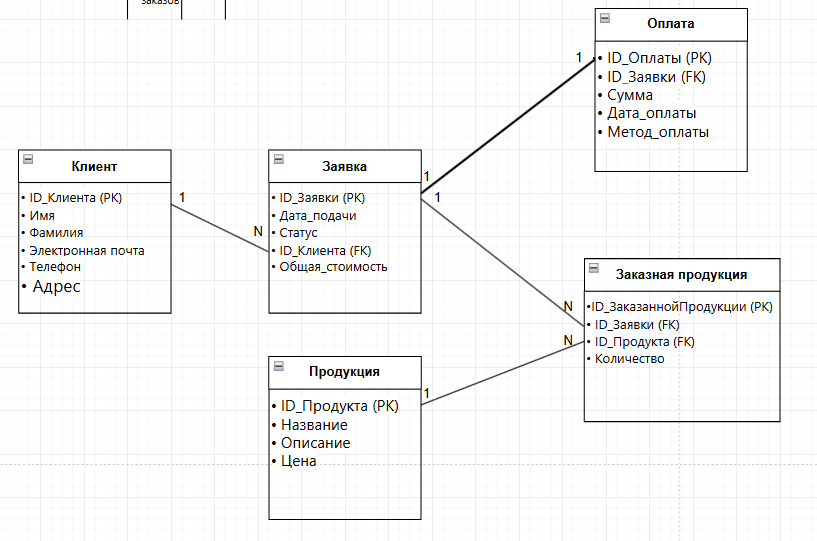
1. Предоставление первых концептов: 2 недели с момента подписания договора.
2. Внесение правок: 1 неделя после получения обратной связи по концепциям.
3. Финальная версия: 1 неделя после утверждения правок.

Контактная информация:

1. Имя контактного лица: Юлия Ижгузина
2. Email: aquarelperm@yandex.ru
3. Телефон: +7 (902) 831-31-46

***Задание2 – Составить схемы IDEF0/Er диаграммму***





***Задание3 – Описать какие языки, технологии следует использовать и почему.***

Для разработки рабочего софта приема заявок типографии можно использовать различные языки программирования и технологии.

**1. Языки программирования:**

• JavaScript (Node.js)

Почему: JavaScript является основным языком для веб-разработки. Используя Node.js, можно создавать серверные приложения, что позволяет использовать один язык как на клиентской, так и на серверной стороне. Это упрощает разработку и уменьшает время на обучение для команды.

• Python

Почему: Python известен своей простотой и читабельностью кода. Он отлично подходит для быстрой разработки и прототипирования. С помощью фреймворков, таких как Django или Flask, можно быстро создать веб-приложение с поддержкой баз данных.

• PHP

Почему: PHP является популярным языком для веб-разработки и хорошо подходит для создания серверной части приложения. Он имеет множество фреймворков (например, Laravel), которые облегчают разработку.

• Java

Почему: Java — это мощный язык с хорошей производительностью и безопасностью. Он подходит для создания крупных корпоративных приложений и может использоваться с фреймворками, такими как Spring.

**2. Фреймворки:**

• React / Angular / Vue.js

Почему: Эти фронтенд-фреймворки позволяют создавать динамичные и отзывчивые пользовательские интерфейсы. Они обеспечивают хорошую производительность и позволяют легко управлять состоянием приложения.

Django / Flask (Python)

Почему: Эти фреймворки позволяют быстро разрабатывать серверные приложения с поддержкой RESTful API, что полезно для работы с фронтендом.

• Express.js (Node.js)

Почему: Express — это минималистичный фреймворк для Node.js, который позволяет быстро создавать серверные приложения и RESTful API.

**3. Базы данных:**

• PostgreSQL

Почему: PostgreSQL — это мощная реляционная база данных с поддержкой сложных запросов и транзакций. Она хорошо подходит для работы с большими объемами данных.

• MySQL

Почему: MySQL является одной из самых популярных реляционных баз данных и хорошо подходит для большинства веб-приложений.

• MongoDB

Почему: MongoDB — это NoSQL база данных, которая хорошо подходит для хранения неструктурированных данных и может быть полезна в проектах с высокими требованиями к масштабируемости.

**4. Инструменты и технологии:**

• Git

Почему: Git — это система контроля версий, которая позволяет команде разработчиков совместно работать над кодом и отслеживать изменения.

• CI/CD (например, Jenkins, GitHub Actions)

Почему: Непрерывная интеграция и развертывание позволяют автоматизировать процесс тестирования и развертывания приложения, что повышает качество кода и ускоряет его доставку.

**5. Облачные технологии:**

• AWS / Google Cloud / Azure

Почему: Облачные платформы предлагают широкий спектр услуг для хостинга приложений, хранения данных и масштабирования, что позволяет легко управлять инфраструктурой.

***Задание4 – Создайте репозиторий git на рабочем столе.***

***Задание5 – Создайте удаленный репозиторий на github загрузите в него вашу работу.***