**Промежуточный отчет по программному проекту**

**1. Основные планы и этапы проекта**

**1.1 Краткое описание проекта:**

**Название проекта**: Программа для анализа рынка труда аналитиков данных («Веб-приложение DataCareerInsight»)

**Цель проекта:** Разработать веб-приложение для анализа рынка труда аналитиков данных в России, используя заранее собранные данные о вакансиях и резюме. Приложение предоставит инструменты для глубокого анализа, включая фильтрацию по таким параметрам, как зарплата, опыт работы, требования к навыкам и т.д. Визуализация данных будет представлена в виде интерактивных графиков и диаграмм, которые пользователь сможет настроить в зависимости от своих потребностей. Ключевыми функциональными возможностями также являются прогнозирование зарплат на основе введённых данных и кластеризация информации для выявления основных тенденций на рынке труда.

**Краткое описание задач:**

1. Разработать скрипт (веб-скрейпер) для автоматического сбора данных о вакансиях и резюме с сайтов hh.ru и, при необходимости, других источников.
2. Создать базу данных для хранения и обработки собранной информации.
3. Реализовать серверную часть для взаимодействия с базой данных, обработки пользовательских запросов, выполнения анализа данных (кластеризация, линейная регрессия) и передачи данных клиенту.
4. Разработать клиентскую часть (веб-интерфейс) с возможностью выбора параметров анализа и отображения результатов в интерактивной форме (графики, таблицы), а также с разделом прогнозирования зарплаты.
5. Провести тестирование всех компонентов проекта для обеспечения корректной работы.
6. Подготовить документацию.

**1.2 Планы и этапы выполнения проекта:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Этап проекта** | **Описание работ** | **Ожидаемые результаты** | **Сроки выполнения** |
| Подготовительные работы | Формулировка целей и задач проекта. Формирование требований к программе и сбор данных. | Полное представление о структуре проекта и его реализации, а также понимание собираемой информации. | 12.11.2024 – 19.11.2024 |
| Разработка и утверждение ТЗ | Описание требований к программе (интерактивные графики, фильтры, прогнозирование зарплат), определение структуры данных и методов анализа (кластеризация, регрессия). Подготовка и утверждение ТЗ с научным руководителем. | Техническое задание с четко определенными функциональными требованиями. | 20.11.2024 – 04.12.2024 |
| Разработка веб-скрейпера | Написание Python-скрпита для автоматического сбора данных с hh.ru и ,при необходимости, других источников. Обработка данных и их интеграция в PostgreSQL. | Рабочий веб-скрейпер, способный собирать актуальные данные с сайтов и сохранять их в базу данных. Готовый датасет. | 05.12.2024 – 25.12.2024 |
| Создание серверной части | Разработка API для обработки пользовательских запросов и передачи данных из базы данных в клиентскую часть. | Серверная часть с возможностью фильтрации данных, проведения анализа, выполнения кластеризации и прогнозирования зарплат. | 26.12.2024 – 31.01.2025 |
| Разработка клиентской части | Создание интерфейса для сайта (меню, описание и т.д.)  Создание интерфейса для выбора параметров анализа (фильтры по регионам, навыкам и т.д.), для кластеризации данных, для линейной регрессии.  Визуализация данных с использованием JavaScript библиотек.  Обеспечение интерактивности (динамическое обновление графиков). | Веб-интерфейс с интерактивными графиками, возможностью выбора параметров анализа и фильтрации данных, возможностью кластеризации и прогнозированием зарплаты. | 01.02.2025 – 10.03.2025 |
| Тестирование программы | Проверка корректности работы веб-скрейпера, серверной и клиентской частей. Исправление выявленных ошибок. | Рабочая программа, корректно выполняющая все функции и устойчивая к ошибочным данным. | 11.03.2025 – 31.03.2025 |
| Подготовка к защите | Подготовка документации, презентации, демонстрации программы и итогового отчета. Финальные исправления программы. | Полностью готовый проект с презентацией и демонстрацией функционала. | 01.04.2025 – 15.04.2025 |

**2. Используемый технологический стек и его обоснование**

**2.1 Перечень используемых технологий:**

|  |  |  |
| --- | --- | --- |
| **Технология/Инструмент** | **Описание** | **Причины выбора** |
| Python | Основной язык программирования. Используется для веб-скрейпера, серверной части и анализа данных. | Универсальный синтаксически простой язык с обширной экосистемой библиотек для обработки данных, веб-разработки и ML-задач. |
| BeautifulSoup/Requests | Библиотеки для веб-скрейпинга и отправки HTTP-запросов. | Простота использования и широкие возможности для сбора и обработки данных с веб-страниц. |
| PostgreSQL | Реляционная база данных для хранения информации о вакансиях и резюме. | Надежная СУБД с мощными инструментами для обработки больших объемов данных и поддержки аналитических запросов. |
| Flask/FastAPI | Фреймворки для реализации серверной части. | Легковесные, высокопроизводительные решения для создания REST API, интегрируемые с базами данных. |
| HTML, CSS, JavaScript | Стандарты для создания клиентского интерфейса. | Обеспечивают адаптивный и удобный интерфейс для взаимодействия пользователя с веб-приложением. |
| Plotly/D3.js | Библиотеки для интерактивной визуализации данных в браузере. | Поддержка создания наглядных, интерактивных графиков и диаграмм в реальном времени. |
| Sklearn/ yellowbrick | Библиотеки Python для анализа данных | Реализация алгоритмов кластеризации (K-means) и линейной регрессии для анализа данных. |
| SQLAlchemy | ORM-библиотека для Python. | Упрощает взаимодействие между серверной частью и базой данных, улучшает читаемость и поддержку кода. |
| Git/GitHub | Система контроля версий. | Обеспечивает отслеживание изменений в коде и удобное развертывание проекта. |

**2.2 Обоснование выбранного технологического стека:**

* Python представляет обширный набор библиотек для сбора данных, анализа и построения моделей машинного обучения
* Использование Flask/FastAPI позволяет быстро создать и протестировать серверную часть.
* PostgreSQL идеально подходит для хранения больших объемов структурированных данных. Она поддерживает сложные запросы, необходимые для аналитики.
* SQLAlchemy упрощает взаимодействие между сервером и базой данных.
* Ploty и D3.js позволяют создать графики, которые динамически изменяются на основе запросов пользователя.
* Использование стандартных технологий для клиентской части (HTML, CSS, JavaScript) гарантирует корректную работу интерфейса в большинстве современных браузеров.
* Использование Git/GitHub обеспечивает надежное хранение кода и управление изменениями.

Таким образом, выбранный стек позволяет добиться поставленных целей проекта, обеспечивая надежность, производительность и удобство как для разработчика, так и для пользователя.

**3. Критерии оценивания проекта**

|  |  |
| --- | --- |
| **Критерий** | **Описание** |
| Работа с базой данных | Будут использованы/Не будут использованы |
| Функциональность - Процент выполнения функциональных требований | Выполненные требования в процентах от общего количества |
| Производительность и эффективность - Время отклика | Среднее время ответа программы на запросы пользователя (в секундах) |
| Качество кода - Покрытие кода тестами (%) | Процент строк кода, покрытых тестами |
| Документация и оформление - Полнота документации (%) | Процент от требуемого объема документации |
| Соблюдение сроков и плана - Процент выполнения работы в срок (%) | Процент задач, выполненных в срок |
| Использование технологического стека - Процент использования функциональности стека (%) | Процент использования функциональности выбранного стека технологий |
| Качество визуализации данных | Интерактивность, читабельность и информативность графиков и диаграмм (оценка визуального представления данных) |

**4. Особые пометки**

* При больших объемах данных возможны задержки в обработке запросов. Это требует оптимизации запросов к базе данных и кода.
* Выбранный стек может быть изменен в ходе разработки. Например:
  + Flask может быть заменен на FastAPI, если потребуется лучшая производительность.
  + Plotly/D3.js могут быть заменены на другую библиотеку для визуализации (например, Chart.js) в случае проблем с интеграцией.

Эти изменения будут зафиксированы в документации и учтены при планировании оставшихся этапов.

* Интеграция новых методов анализа данных (например, более сложных алгоритмов машинного обучения) может потребовать пересмотра сроков и функциональности.