1. Оформление кода

1.1. Требования по оформлению исходного кода

1.1.1. Формат

Требуется предоставить исходный код в файлах формата используемого языка (пример: .py для языка Python). Файлы должны использовать кодировку семейства UTF-8. Имена файлов и директорий должны иметь ту же кодировку, что и сами текстовые файлы. В качестве управляющих символов для обозначения разрыва строки рекомендуется использовать LF (предпочтительно), либо CRLF.
Предоставление кода производится в виде Git-репозитория, если это возможно. В противном случае необходимо предоставить исходный код в формате архива со следующими характеристиками:

- Использование открытого (OpenSource/FreeSoftware) формата сжатия либо стандартизированного для обмена кодом конкретно с данным подрядчиком.
- Имена файлов и директорий и архивов должны иметь кодировку семейства UTF или ASCII.
- Вместе с проектом должны быть упакованы файлы системы контроля версий (директория .git и файл .gitignore).

Отсутствие системы контроля версий не допускается.

1.1.2. Модульность

Код должен быть логически разбит на модули и иметь древовидную или иную, допустимую спецификацией языка, структуру импортов модулей. При разбиении исходного кода на модули требуется следовать спецификации или best practice для используемого языка. Ввиду трудности оценки данного соответствия данному требованию рекомендуется использование автоматизированных средств контроля.
Если приложение имеет микросервисную архитектуру, исходный код каждого поставляемого сервиса должен храниться в отдельном репозитории в системе контроля версий.

1.1.3. Форматирование

Форматирование исходного кода, а именно:

- Использование определенных символов для отступов
- Формат отступов
- Оформление блоков кода
- Именование переменных и модулей
- и т.д.

Должно быть выполнено в едином стиле для всего проекта. Смешение стилей не допускается. Рекомендуется следование стандартам языка при наличии таковых (пример: PEP8, PEP257 для Python). В случае отсутствия подобных стандартов рекомендуется использование best practice для данного языка.

В случае несоответствия данным требованиям необходимо включить в состав исходного кода либо документации проекта информацию об используемом стандарте форматирования кода.
В случае наличия корпоративного стандарта форматирования кода рекомендуется включить в состав исходного кода файл документации с правилами форматирования.
Рекомендуется использование автоматических средств форматирования и/или контроля за соблюдением формата, в таком случае желательно также приложить их файлы конфигурации.

1.1.4. Сторонние библиотеки

В случае использования сторонних библиотек, они должны быть включены в состав репозитория в формате файла требований для используемого пакетного менеджера (например, файл requirements.txt для pip). Пакет, содержащий библиотеку, должен быть доступен для скачивания из Интернет. Версия пакета должна быть указана точно (не допускается использование формата “номер версии или выше”).
Требуется использовать актуальные версии библиотек, не содержащие известных уязвимостей (CVE).

1.1.5. Документирование файлов

Документирование исходного кода осуществляется на английском языке.
Требуется документирование частей исходного кода непосредственно в файлах с исходным кодом. В связи с трудностью оценки данной характеристики, ожидается возможность направить исполнителю требования о документировании той или иной части исходного кода в течении всего времени поддержки.
Рекомендуется написание документации ко всем модулям, классам и методам. Однако документирование малых методов или методов, чье предназначение понятно из названия (например, toInt()) не требуется.
Рекомендуется использование стандартов документирования для используемого языка программирования, либо, при их отсутствии, общих рекомендаций по документированию исходного кода.

1.2. Требования по оформлению Jupyter Notebook

Требования, указанные в пункте 1.1, должны соблюдаться при оформлении кода Jupyter Notebook, если данный пункт не декларирует иное.

1.2.1. Формат

Необходимо выполнение всех требований, указанных в п. 1.1.1, за исключением требования расширения файла.
Все пояснения к ячейкам кода должны быть оформлены с использованием формата Markdown. В случае необходимости в сложных математических выражениях используется LaTeX. Использование HTML допускается в случае, если функциональности Markdown недостаточно, однако не рекомендуется.

1.2.2. Модульность

В файле Jupyter Notebook должна прослеживаться четкая иерархическая структура разбиений на участки, объединенные одной темой, с использованием синтаксиса многоуровневых заголовков Markdown.
Каждая ячейка кода должна содержать предшествующее ей объяснение производимых в ней действий.
Ячейка кода должна быть небольшого размера. Большие части кода, не взаимодействующие с пользователем (не выводящие информацию) и не являющиеся ключевыми для понимания, должны быть вынесены в функцию в файле исходного кода. Данный файл должен быть подключен как библиотека. Необходимо стремиться к формату, в котором ноутбук представляет собой последовательность вызовов подобных функций.
Ячейки должны располагаться последовательно в том порядке, в котором необходимо осуществлять их запуск.

1.2.3. Рекомендуемое оформление секции

Здесь и далее пару из ячейки кода и предшествующего ей объяснения будем называть секцией.
Ячейки кода без объяснения к ним допускаются, если содержат интуитивно понятный код размером не более 3 строк.
Размер секции должен быть достаточно мал. Ячейки кода размером более 15-20 строк рекомендуется дробить на более мелкие секции. На объяснение не накладывается ограничений по объему, однако оно не должно быть излишне подробным.
Если секция выводит некоторую информацию на экран, необходимо убедиться, что данная информация не занимает много места (средний объем должен составлять 5 строк). При передаче файла ноутбука заказчику требуется удалить всю выведенную секцией информацию. Исключение - в дальнейшем на данную информацию будет опираться рассуждение.

1.2.4. Рекомендуемая структура Notebook

Рекомендуется придерживаться следующей структуры файла Jupyter Notebook:

- Заголовок, выполненный с использованием синтаксиса Markdown;
- Ячейка кода с директивами imports для подключения внешних библиотек (см. п. 1.1.4). В пояснении к данной ячейке требуется указать, для чего используется каждая из подключенных библиотек. Для библиотек, входящих в стандартную библиотеку языка, пояснение не требуется. Допускается, но не рекомендуется опускать пояснение для общеизвестных библиотек (например, matplotlib);
- Ячейка, содержащая адреса источников данных. Каждый адрес должен быть вынесен в отдельную глобальную переменную.
- Введение. Введение должно кратко описывать назначение файла ноутбука и содержать справочную информацию и(или) ссылки на нее при необходимости;

- Определения. В данном разделе должны быть введены все использующиеся в тексте файла определения, не являющиеся общепринятыми;
- Раздел загрузки данных. В ячейках кода этого раздела производится загрузка данных из источников. В описании должна быть предоставлена информация об источниках и форматах данных. Допускаются ссылки на сопровождающие документы, в том числе аналитическую записку. Если имеется возможность, стоит вывести пример загруженных данных для формирования понимания структуры данных у читателя. Подобных разделов может быть несколько в составе одного Jupyter Notebook.
- Секции с действиями над данными.
- Анализ полученных результатов.
- Любая дополнительная информация.

1.2.5. Работа с данными

По возможности, способ загрузки данных должен быть независим от окружения, в котором запускается файл ноутбука. Крайне не рекомендуется загрузка данных с жесткого диска; если подобная операция все же осуществляется, в описании необходимо указать, какие данные и в каких директориях ожидается найти
В случае, если используется загрузка данных с внешних источников, адреса источников необходимо вынести в глобальные переменные.
Временные данные, генерируемые в процессе работы, требуется хранить либо во временных файлах, либо в директории, непосредственно содержащей файл ноутбука. Файлы, не используемые при повторных запусках ноутбука, должны очищаться выполнением кода в одной из последних секций.

1.2.6. Оформление графиков

Все графики должны содержать легенду.
Оси графиков должны иметь подписи. Подписи должны иметь читаемый размер. Должны быть указаны единицы измерения, если это возможно.
Цветовое оформление должно быть достаточно контрастным, чтобы графики легко идентифицировались.