



Описание технических средств хранения исходного текста и объектного кода программы MislalabMedAI

Введение

В данном разделе будет представлено общее описание цели и необходимости управления исходным текстом и объектным кодом программного обеспечения. Управление исходным кодом и объектным кодом является важным аспектом разработки программного обеспечения, так как позволяет обеспечить контроль версий, совместную работу, отслеживание изменений и другие важные функции.

Этот документ предназначен для разработчиков программного обеспечения, менеджеров проектов, тестировщиков и других специалистов, которые работают с исходным кодом и объектным кодом программного обеспечения.

Система контроля версий

Для управления исходным кодом проекта используется система контроля версий Git.

В проекте существуют 2 основные ветки «master» и «develop». Ветка «master» содержит последнюю рабочую версию программы, которое может быть выпущено в релиз. Ветка «develop» - содержит версию программы, в которую добавлены последние изменения от разработчиков. Из ветки «develop» производится выгрузка на тестовые сервера для проверки и последующего слияния (merge request) этих изменений в ветку «master».

Каждый отдельный разработчик, приступая к работе над новой задачей, создаёт свою собственную ветку от текущего коммита ветки «develop». Коммиты (commit - операция, которая фиксирует изменения в исходном коде и сохраняет текущее состояние файлов в репозитории на определенный момент времени) в ветку разработки должны состоять из функционально связанных изменений (микроммиты). После того, как работа над задачей (и проверка работоспособности решения) закончена, разработчик формирует объединение всех внесённых изменений (merge request) для того, чтобы ведущий разработчик добавил эти изменения в ветку «master».

К репозиторию могут получить доступ только авторизованные пользователи, которые делятся на Оунеров (Owners – владельцы проекта), Майнтейнеров (Maintainers – разработчики, ответственные за поддержку проекта, включая управление кодовой базой, объединение версий, обеспечение качества кода) и Девелоперов (Developers - программисты,



занимающиеся написанием, тестированием и отладкой кода в рамках проекта). Ветка «master» является защищённой и может быть удалена только Оунером. Только Майнтейнеры и Оунеры могут изменять код в ветке «master», «develop». Разработчики добавляют код в ветку «develop» формирую итоговое слияние версий (merge request).

Хранение исходного текста

Для хранения исходного кода используются сервера и ПО, расположенное во внутренней сети компании. Доступ к сервисам хранения исходного кода Программы извне не предусмотрен.

Для обеспечения сохранности исходного кода используется методика резервного копирования при помощи зеркалирования на аналогичные жесткие диски. Так как для работы используются ОС (операционные системы) семейства Linux, то восстановление из зеркальных копий происходит в кратчайшие сроки.

Для работы с исходным кодом в компании развернут собственный приватный репозиторий на базе сервера Gitlab-CE.

Инструменты и средства разработки

Программное обеспечение написано на языке Python, в результате чего возможно избежать компиляции исходных текстов Программы в объектный код. Кроме того, проект Программы имеет стандартную структуру. Всё вышеперечисленное позволяет редактировать исходный код в любом текстовом редакторе. В связи с этим, разработчики самостоятельно выбирают удобные для себя средства разработки. Выбор средств разработки не регламентируется.

Компиляция исходного текста

Компиляция исходного текста Программы в объектный код не производится.

Управление зависимостями

Для работы Программы используется виртуальное окружение «venv» и менеджер пакетов «pip». Использование виртуального окружения позволяет избегать конфликтов версий пакетов. Для добавления исходного кода в ветку «master» и «develop» в файле с зависимостями проекта должны быть указаны точные версии пакетов.



Версионирование и выпуск продукта

Версия продукта может быть получена в любой момент работы программы при помощи запроса к соответствующему эндпоинту API (API endpoint - конечная точка веб-сервиса, к которой клиентское приложение может отправить запрос для выполнения определённой операции). Версия продукта состоит из названия компонента программы, названия ветки исходного кода, даты коммита и ID коммита. Таким образом, для каждого запущенного экземпляра программы есть возможность установить версию исходного кода.

Пример ответа на запрос версии программы:

...

```
"name": "mav-svc-ocr",
```

```
"branch": "master",
```

```
"date": "2024.03.07.14.40.26",
```

```
"commit": "64e52e1b4016bd45263947d79ceacc84a9802d6f",
```

...

Когда принимается решение о выпуске новой версии программы, то от текущего коммита в нужной ветке делается дополнительная ветка с утвержденным названием. Название ветки формируется из двух частей – название релиза и бинарный номер версии (x.y). Название выбирается исходя из целей релиза таким образом, чтобы всем сотрудниками организации было понятно назначение данного релиза, а бинарный номер версии выбирается ведущим программистом, исходя из оценки объёма изменений, которые были внесены в программу в результате доработок. Номер текущей версии должен быть больше номера предыдущей.

Приложения

Список используемого ПО и сервисов:

ПО	Пояснение
LabelStudio 1.10.0	Сервис, который используется для разметки датасетов. Свободное ПО, установленное на приватном сервере компании.



GitLab-CE 14.5.1	Сервис, который используется для хранения репозитория с исходным кодом. Свободное ПО, установленное на приватном сервере компании.
Docker 25.0.3	Среда виртуализации для запуска ПО. Свободное ПО, установленное на тестовые сервера компании, рабочие места разработчиков.

Пример конфигурационного файла («Dockerfile») для формирования образа ПО в среде виртуализации Docker:

Используем промежуточный образ, который содержит слои большого объема, чтобы не перезагружать их по-новой каждый раз

для каждого проекта.

FROM mav/svc-image:1.0

Добавляем файл с зависимостями

ADD ./requirements.txt /src

WORKDIR /src

Устанавливаем дополнительное ПО

RUN apt-get update

RUN apt-get install -y libglib2.0-0 libsm6 libxrender1 libxext6

RUN apt-get install -y tk

Устанавливаем специфические для проекта зависимости, если они не были установлены ранее

RUN pip3 install -r requirements.txt

Добавляем исходники и др., необходимые для работы проекта файлы.



ADD ./ctools /src/ctools

ADD ./locales /src/locales

ADD ./msvc /src/msvc

ADD ./ocr /src/ocr

ADD ./rest /src/rest

ADD ./version /src/version

ADD ./data/models /src/data/models

ADD ./CLI.py /src

ADD ./svc.ini /src

ADD ./svc.py /src

Даём возможность записи и чтения файлов в папке.

RUN chmod -R ugo+rw /src

Запускаем программу

CMD python3 /src/svc.py;

Публикуем порт

EXPOSE 17013