Руководство администратора программного продукта СИП "Цифровая приемная"

1. Предисловие

Сервис СИП "Цифровая приемная" компании Ред Софт предназначен для распознавания входящих документов (напр. обращений граждан или организаций) и для автоматизации регистрации таких документов в системах АИС. Обращения могут быть в формате ГОСТ Р 7.0.97-2016, так и в нестандартной форме.

Система относится к решениям на основе технологии искусственного интеллекта в области управления бизнес-процессами и предназначена для автоматического назначения исполнителя и вынесения резолюции на основе текста входящего обращения и другой информации из обращения. Если обращение поступило в виде скан-копии, в форматах .pdf, .png, .jpg, .jpeg, .bmp, СИП "Цифровая приемная" распознаёт блоки полей: Отправитель, Информация об отправителе, Получатель, Номер документа, Тема, Основной текст, Должность, Подпись, ФИО подписанта и т.д, что исключает ввод необходимой для назначения исполнителя и вынесения резолюции информации вручную.

Система представляет собой веб сервис. Дистрибутив системы распространяется посредством системы койтенеризации docker.

Сервис использует протокол HTTP для обмена.

2. Требования

Сервис возможно установить и запустить на любую платформу (Windows, Linux, MacOs...) оборудованную:

1. x86-64 совместимый процессор;
2. Docker - платформа для койтенеризации приложений Домашняя страница (проверено для версии 20.10.18);

Далее в инструкции предполагается использование Linux (все команды приведены для Linux).

В данном руководстве не рассматривается вопрос установки docker, поскольку это зависит от платформы.

Для оптимальной производительности рекомендуется использовать

3. Установка и запуск сервиса

Предполагается, что ранее загружена версия сервиса в виде образа Docker, с именем файла:

docia-service-XXXXXXX.tar.bz2

3.1. Загрузить образ в docker

Для дальнейшего использования сервиса, образ необходимо загрузить в docker.

Загрузить образ в список доступных образов docker необходимо командой:

```
docker load --input docia-service-XXXXXXX.tar.bz2
```

3.2. Запуск сервиса

Предполагаем запуск сервиса на порте 8080. Запуск производится командой *docker run*:

```
docker run --name "docia" --rm -d -p 8080:8080 hub.red-
soft.ru/datasience/sip-fssp/docia-service:XXXXXXX
```

Для другого порта (XXXX):

```
docker run --name "docia" --rm -d -p XXXX:8080 hub.red-
soft.ru/datasience/sip-fssp/docia-service:XXXXXXX
```

3.3. Проверка сервиса

Наиболее простой способ проверить доступность сервиса выполнить команду (curl):

```
curl localhost:8080/api/v1/info
```

На что сервис должен вернуть ответ, например:

```
{"data":{"DocIA
Service":"XXXXXXX","onnxruntime":"1.12.1","pdfium":"108.0.5323","tesseract"
:"5.2.0"}}
```

3.4 Остановка сервиса

Для остановки сервиса необходимо выполнить команду *docker stop*:

```
docker stop docia
```

В данной команде предполагается, что имя контейнера сервиса дано, как в данном руководстве *docia*, в противном случае команда остановки сервиса должна выглядеть иначе:

```
docker stop NAME
```

где NAME - имя контейнера

4. Настройки сервиса

По-умолчанию в докер образе сервиса установлены наиболее оптимальные параметры, для текущей версии. И для большинства вариантов работы приемлем именно вариант запуска сервиса без настроек. Однако, в сервисе есть дополнительные настройки, которые регулируют работу сервиса.

На текущий момент все настройки сервиса производятся с помощью переменных окружения, вот список настроек:

```
PRINT_CONFIG=1 - Отобразить в консоль содержимое конфигурационного файла
LOG_LEVEL="info" - Установить уровень логирования допустимые значения
(trace|debug|info|warn|error)
LOG_SYNC_ENABLE=false - Управление буферизацией логирования
SERVICE_TIMEOUT=20 - Таймаут обработки входящего запроса
SERVICE_CONCURRENCY=10 - Максимальное количество одновременно
обрабатываемых запросов
METRICS_ENABLE=false - Добавлять или нет данные метрик в ответ
VERSIONS_ENABLE=false - Добавлять или нет информацию о версиях компонентов
в ответы
```

Все опции кроме PRINT_CONFIG, могут быть настроены через конфигурационный файл.

4.1. Опция PRINT_CONFIG

В случае, если указана данная опция, то сервис выводит в лог при запуске свой конфигурационный файл:

```
$ docker run --name "docia" -e PRINT_CONFIG=1 --rm -d -p 8080:8080 hub.red-
soft.ru/datasience/sip-fssp/docia-service:latest
7e5654b8d9a79e8d9dcd4bfadd66b36f7e6cfe8981f70035cae0d1f0c0560c7d

$ docker logs docia

[[logger]]
level = "info"
appender = "console"
targets = [
    "docia_service=info",
    "docia_detector=info",
    "docia_converter=info",
    "docia_recognizer=info",
    "mcai_onnxruntime::environment=error",
    "mcai_onnxruntime::session=warn",
]
sync = false

...
score_name = "ocr"
```

```
lang = "rus+eng"
2023-04-10T15:31:18.294004Z  INFO ThreadId(01)
docia_service::application::state: DocIA Service: 0.2.16
...
```

По-умолчанию вывод конфигурации не производится.

4.2. Опция LOG_LEVEL

Данная опция задает уровень логирования сервиса. Возможные значения:

1. error - выводится только информация об ошибках;
2. warn - помимо ошибок выводит информация о предупреждениях;
3. info - стандартный уровень вывода логов, выводится основная информация о работе сервиса и запросах;
4. debug - детальный уровень вывода логов;
5. trace - самый детальный уровень вывода логов.

По-умолчанию используется уровень *info*.

4.3. Опция LOG_SYNC_ENABLE

Данная опция определяет возможно ли буферизировать вывод логов. Если *true*, то буфер вывода логов отключается и используется синхронный вывод логов в стандартный вывод процесса. По-умолчанию *false*.

4.4. Опция SERVICE_TIMEOUT

Настройка таймаута для обработки входящего запроса. По-умолчанию отключен.

4.5. Опция SERVICE_CONCURRENCY

Настройка определяет кол-во одновременно обрабатываемых подключений к сервису. По-умолчанию отключена.

4.6. Опция METRICS_ENABLE

Определяет необходимость включения метрик скорости в ответы сервиса. По-умолчанию стоит значение *false*. Если опция включена, то ответ сервиса также содержит дополнительные метрики о скорости выполнения различных компонент системы:

```
{
    "files": {
        "doc1": {
            "metrics": {
                "macro_blocks": {
                    "secs": 1,
                    "nanos": 513879432
                },
                "tesseract": {
```

```
                    "secs": 6,
                    "nanos": 464441300
                },
                "classifier": {
                    "secs": 0,
                    "nanos": 9164421
                },
                "total": {
                    "secs": 9,
                    "nanos": 447119434
                },
                "converter": {
                    "secs": 0,
                    "nanos": 4419983
                },
                "blocks": {
                    "secs": 1,
                    "nanos": 455214298
                }
            },
            "blocks": [
                ...
            ]
            ...
        }
    }
}
```

4.7. Опция VERSIONS_ENABLE

Опция определяет будут ли включаться в ответ сервиса версии компонент системы. По-умолчанию стоит значение *false*. В случае *true* ответ сервиса также будет включать и версии компонент:

```
{
    "files": {
        ...
    },
    "versions": {
        "tesseract": "5.2.0",
        "pdfium": "108.0.5323",
        "DocIA Service": "0.2.16",
        "onnxruntime": "1.12.1"
    }
}
```

5. Конфигурационный файл

Конфигурационный файл сервиса создается на основе шаблона при старте докер образа. Шаблон конфигурационного файла находится по пути */opt/docia/config.toml.template*.

6. Логи

В зависимости от настроек уровня логи сервиса выводятся стандартным образом (зависит от настройки docker). Их можно просмотреть стандартной командой вывода логов сервиса *docker logs*:

```
$ docker logs docia

2023-04-10T15:31:18.294004Z  INFO ThreadId(01)
docia_service::application::state: DocIA Service: 0.2.16
2023-04-10T15:31:18.310917Z  INFO ThreadId(01)
docia_service::application::state: pdfium: 108.0.5323
2023-04-10T15:31:18.322579Z  INFO ThreadId(01)
docia_service::application::state: onnxruntime: 1.12.1
...
```

При этом в логи выводит информация о времени, версия компонент, информация о потоке, а также информация о каждом запросе от пользователя. При этом также выводится информация о затраченном времени ответа на каждый запрос.

7. Оптимизация

Для оптимальной производительности системы рекомендуется параллельный запуск не более N одновременных соединений, где N - кол-во ядер в системе. В случае превышения, происходит нелинейная деградация системы и в таком случае рекомендуется использовать средства распределения сетевой нагрузки на сервис (балансировщики нагрузки).