ООО «Новые программные системы»

Инструкция по установке экземпляра программного обеспечения

«Программная система NGSWizard 1.3»

(версия 1.3)

Листов 25

2022 год

АННОТАЦИЯ

Настоящий документ является инструкцией системного программиста (администратора) программы анализа молекулярно-генетических характеристик по данным высокопроизводительного секвенирования ДНК (далее – Программа), предназначенной для мониторинга за молекулярно-генетическими показателями и поддержки принятия решений медицинских работников по лечению.

В разделе «Общие сведения о Программе» указаны назначение и функции Программы и сведения о технических и программных средствах, необходимых для работы Программы.

В разделе «Структура Программы» приведены сведения о структуре Программы, порядке взаимодействия составных частей, связях с другими программами.

В разделе «Настройка Программы» приведено описание действий по подготовке технических средств к установке Программы, а также установке и настройке Программы.

В разделе «Проверка Программы» приведено описание порядка проверки Программы.

В разделе «Дополнительные возможности» приведено описание порядка настройки системы сбора логов и мониторинга, настройки системы сбора и анализа метрик, а также порядок работы с резервными копиями.

В разделе «Сообщения системному программисту» указаны сообщения, выдаваемые в ходе настройки Программы, описание их содержания и действий, которые необходимо предпринять системному программисту (администратору).

СОДЕРЖАНИЕ

1. НАСТРОЙКА ПРОГРАММЫ

1.1 Запуск демонстрационной версии, используя образ виртуальной машины

В ознакомительных целях самый быстрый способ развернуть программу — запустить ее в виртуальной машине, используя подготовленный образ (снимок диска виртуальной машины в формате VMDK). В данном разделе описаны шаги для запуска виртуальной машины с помощью программы Oracle VirtualBox (проверка осуществлялась с использованием версии 6.1.30). (***Внимание: данный способ установки не предназначен для промышленного использования программы.)***

Образ содержит сконфигурированные и автоматически запускаемые при старте контейнеры сервера приложений, одной вычислительной ноды и клиентской части под управлением веб-сервера Nginx, а также используемые в работе программы файлы референсных геномов человека и данные из открытых баз данных для аннотации однонуклеотидных вариантов и коротких инсерций/делеций (Ensembl: ver 103, 1KG: 2019-02-26, ClinVar: 2021-03-15, dbNSFP: 2021-04-06 (ver 4.2c), dbSnp: 2020-05-01 (ver 154), gnomAD3: ver 3.1).

Минимальные технические требования к рабочей станции для запуска демонстрационной версии:

— не менее 16 Гб оперативной памяти;

— не менее 500 Гб доступного пространства на жестком диске;

— сетевая карта, работающая на скорости не ниже 100 Мбит/сек;

— процессор, имеющий не менее 4-х вычислительных ядер с тактовой частотой не ниже 2 ГГц;

— рабочая станция должна поддерживать технологию аппаратной виртуализации VT-x или AMD-x, поддержка данных опций должна быть включена в BIOS рабочей станции.

На рабочей станции должно быть установлено следующее программное обеспечение:

— Oracle VirtualBox, версии 6, с поддержкой 64-битной архитектуры

— WEB-браузер (Google Chrome версии не ниже 50.0.2661, Mozilla Firefox версии не ниже 52, Safari версии не ниже 10.0, Microsoft Edge версии не ниже 41.16299.15, Microsoft Internet Explorer версии не ниже 11).

После запуска программы VirtualBox, нажмите на кнопку «Создать» для создания новой виртуальной машины.

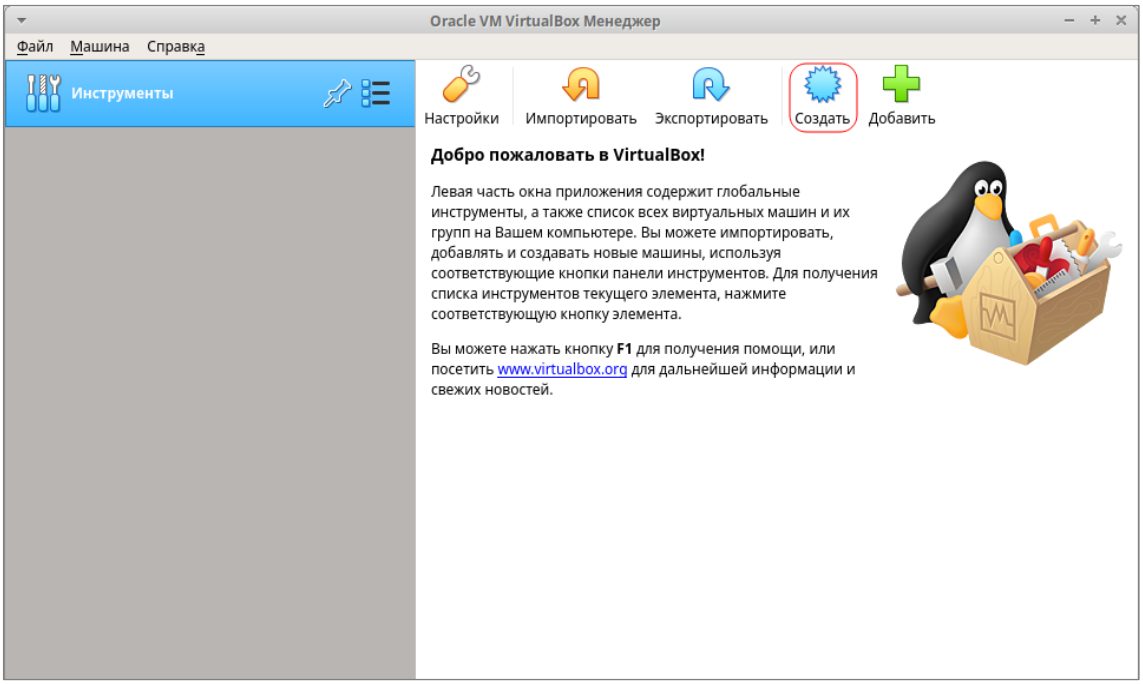

Рисунок 1 — Окно программы VirtualBox

После этого будет отображено диалоговое окно, в котором нужно указать произвольное имя для виртуальной машины, указать папку, в которой будут хранится файлы созданной виртуальной машины, выбрать тип Linux и версию Ubuntu (64-bit). Затем нажать кнопку «Далее».

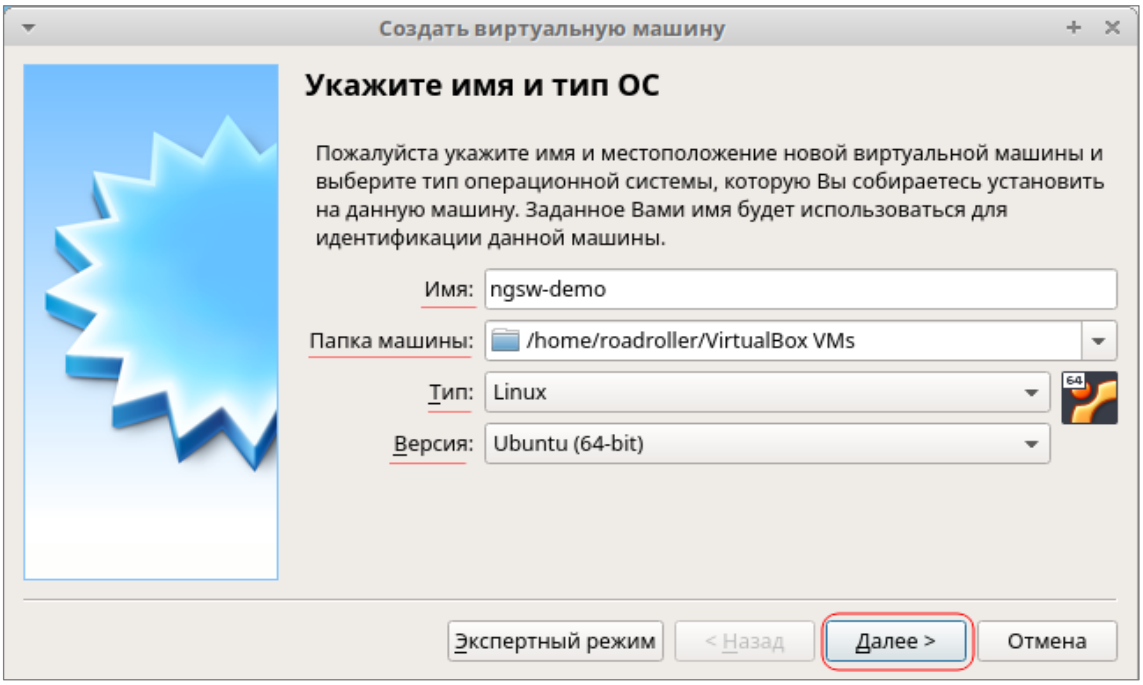

Рисунок 2 — Окно задания параметров виртуальной машины

На следующем шаге указывается объем оперативной памяти, которая будет выделена для виртуальной машины, минимальное рекомендуемое значение 16384 MB (16GB) и нажать кнопку «Далее».

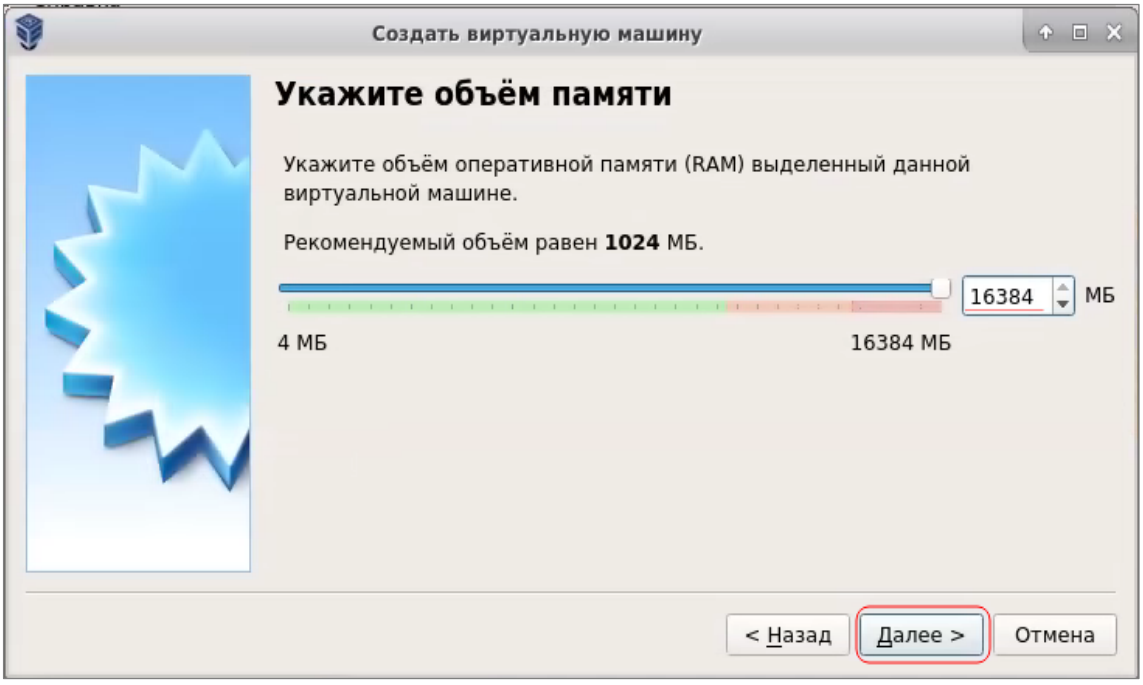

Рисунок 3 — Окно задания оперативной памяти

На следующем шаге, нужно выбрать опцию «Использовать существующий жесткий диск», затем нажать на кнопку для выбора пути к файлу жесткого диска виртуальной машины.

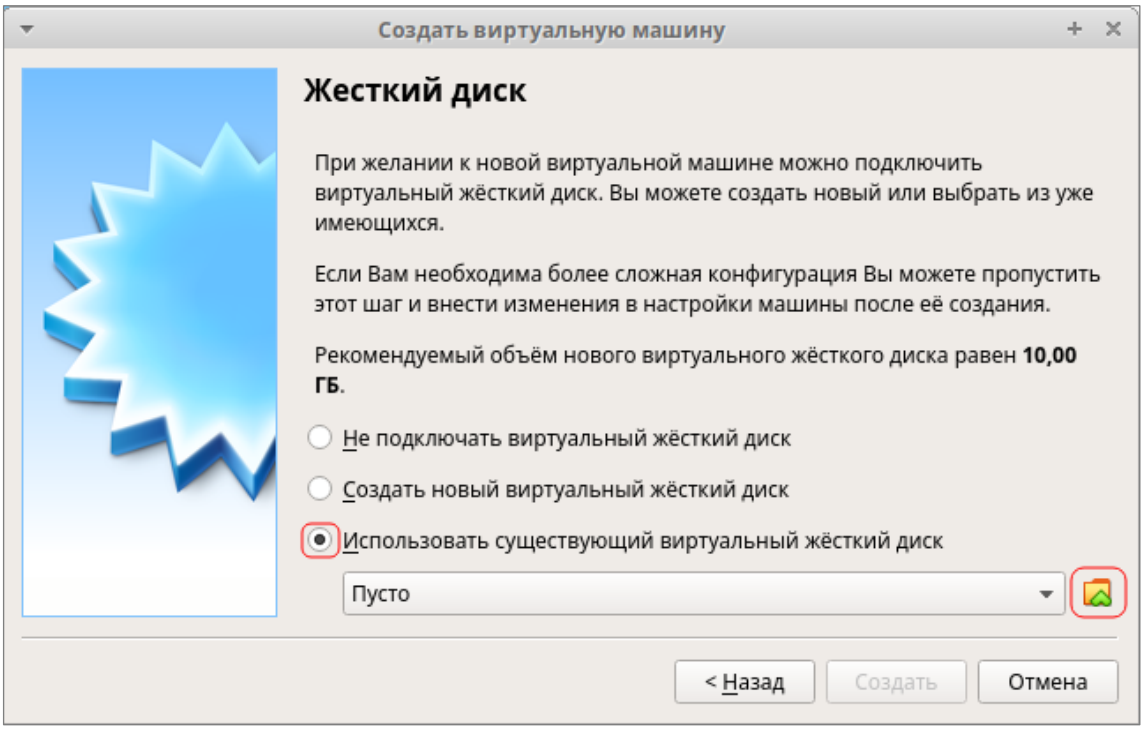

Рисунок 4 —Окно выбора опций для жёсткого диска

В появившемся окне «Выбор жесткого диска», нажать кнопку «Добавить» и выбрать файл жесткого диска, содержащий демонстрационную версию. Затем нажать на кнопку «Выбрать».

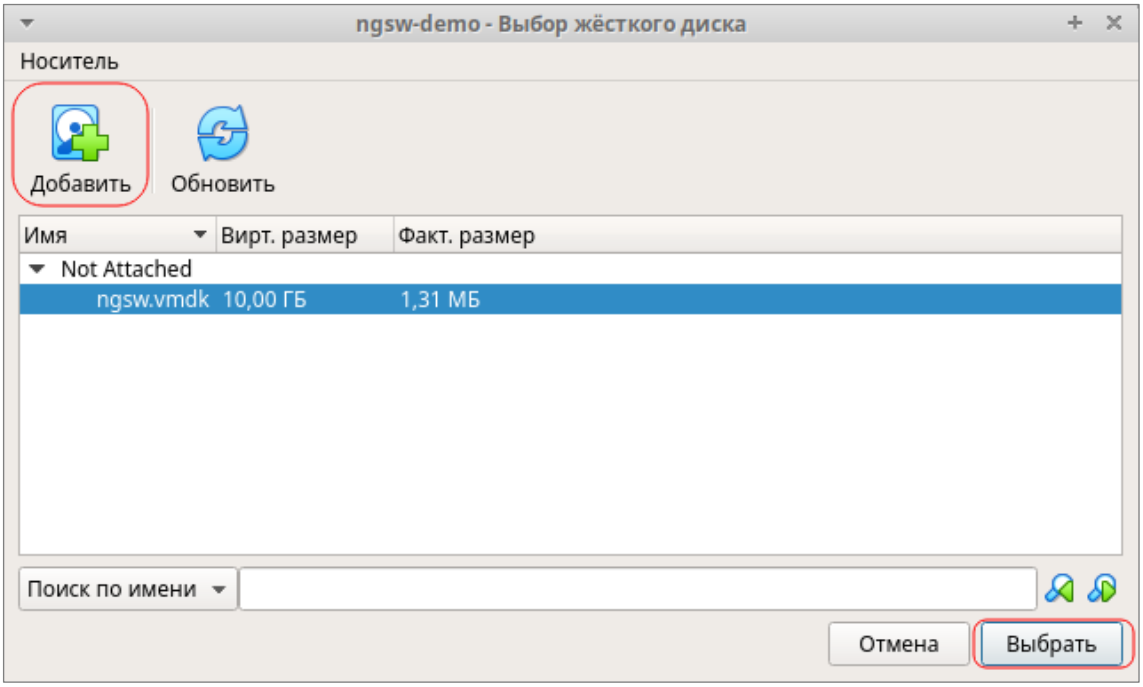

Рисунок 5 — Окно выбора жесткого диска

После возвращения в мастер создания виртуальной машины нажать там кнопку «Создать». Виртуальная машина будет создана и отображена в списке

виртуальных машин. Для запуска нужно выбрать виртуальную машину из списка и нажать на кнопку «Запустить».

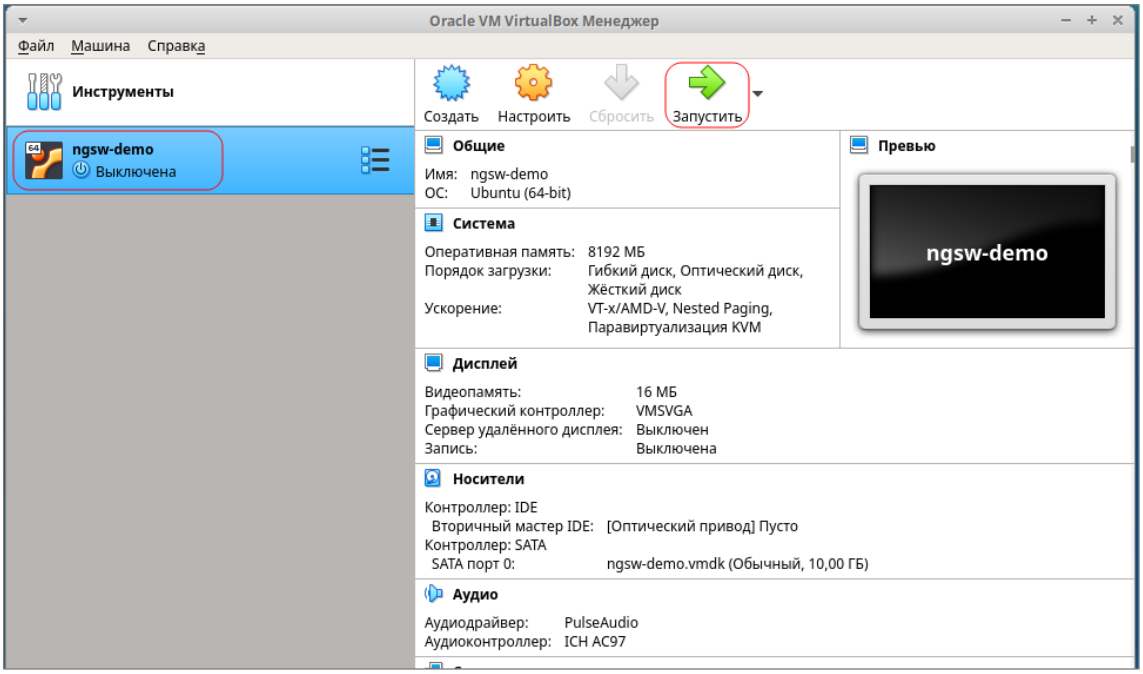

Рисунок 6 —Окно мастера создания виртуальной машины

После того, как виртуальная машина запуститься, приложение будет доступно на рабочей станции в веб-браузере по адресу http://localhost.

В следующих разделах описаны подготовка и установка программы для промышленного использования.

1.2 Подготовка к установке Программы

Компоненты программы поставляются в виде контейнеров. В качестве средства контейнеризации используется СПО Docker (https://www.docker.com). Имеется четыре типа образов:

Образ сервера БД — развертывается в одном экземпляре (при развертывании системы для промышленной эксплуатации, рекомендуется установка PostgreSQL без использования контейнера).

Образ сервера приложений — развертывается в одном экземпляре.

Образ вычислительного узла — как правило развертываются несколько экземпляров, количество зависит от требований к пропускной способности системы.

Образ Jupyter Hub — развертывается в одном экземпляре, позволяет открывать и анализировать файлы результатов в Jupyter Notebook.

Образы сервера БД и вычислительных узлов не требуют обновления при изменении кода системы: инкрементальное обновление БД происходит при старте сервера приложений, код задач загружается на вычислительные узлы, благодаря механизму «Zero Deployment» реализованному в фреймворке Apache Ignite.

1.2.1 Установка Docker

СПО Docker должно быть установлено на всех серверах Программы, в том числе на вычислительных узлах. Для установки Docker можно воспользоваться скриптом install-docker.sh, который находиться в корне папки проекта бэкенда. В рамках промышленной эксплуатации, скрипт был протестирован разработчиками на дистрибутивах Debian 10 LTS и Ubuntu 18 LTS, для других дистрибутивов процедура может отличаться, в этом случае нужно следовать инструкциям из официальной документации Docker https://docs.docker.com/engine/install.

Проверка успешной установки СПО Docker осуществляется с помощью команды:

sudo docker run hello-world

В случае успешной установки, в консоли будет отображен следующий текст:

Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
0e03bdcc26d7: Pull complete
Digest:
sha256:4cf9c47f86df71d48364001ede3a4fcd85ae80ce02ebad74156906caff5378bc
Status: Downloaded newer image for hello-world:latest
Hello from Docker!
This message shows that your installation appears to be working correctly.
To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub. (amd64)
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash
Share images, automate workflows, and more with a free Docker ID: https://hub.docker.com/
For more examples and ideas, visit: https://docs.docker.com/get-started/

1.2.2 Установка PostgreSQL

Для использования Программы в промышленных целях рекомендуется установка PostgresSQL без использования образа Docker, в случае тестовой установки можно разворачивать PostgreSQL из контейнера.

Для установки PostgreSQL можно воспользоваться скриптом **install-postgres.sh**, который находиться в корне папки проекта бэкенда. В рамках промышленной эксплуатации, скрипт был протестирован разработчиками на дистрибутивах Debian 10 LTS и Ubuntu 18 LTS, для других дистрибутивов процедура может отличаться, в этом случае нужно следовать инструкциям из официальной документации PostgreSQL https://www.postgresql.org/download.

После установки необходимо подключиться к PostgreSQL с помощью утилиты psql (либо используя pgAdmin), создать пользователя, базу данных и подключить расширение pg_trgm, для чего выполнить следующий код:

```
CREATE ROLE gliotrace LOGIN
PASSWORD '<DB_PASSWORD>';
CREATE DATABASE gliotrace
WITH OWNER = gliotrace
ENCODING = 'UTF8'
LC_COLLATE = 'en_US.UTF-8'
LC_CTYPE = 'en_US.UTF-8';
\c gliotrace
```

CREATE EXTENSION IF NOT EXISTS pg_trgm;

Если PostgreSQL физически устанавливается на отдельном сервере, то в конфигурационном файле postgresql.conf, расположенный в папке /etc/postgresql/12/main/ необходимо добавить адрес сервера приложений в список адресов, от которых PostgreSQL будет принимать соединения. Для этого необходимо отредактировать строку файла конфигурации и указать IP сервера приложений:

listen_addresses = 'localhost,<IP сервера приложений>'

1.2.3 Сборка Docker образов из исходного кода

При сборке разработчиком образов из исходного кода необходимо установить следующее программное обеспечение:

— Docker 19 (https://docs.docker.com/get-docker);

— PostgreSQL 12.5 (https://www.postgresql.org/download);

— Java 8 SDK (http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html);

— Gradle 5.6 (https://gradle.org/install);

— Nodejs 10.22 (https://nodejs.org/en/download);

— Yarn 1.12. (https://classic.yarnpkg.com/en/docs/install).

Администратор Программы развертывает Программу из готовых образов, переданных ему разработчиками Программы.

После установки необходимо подключиться к PostgreSQL, создать пользователя, базу данных и подключить расширение pg_trgm таким же образом как это описано в разделе «Установка PostgreSQL».

После выполнения указанных операций необходимо клонировать репозитории проектов бэкенда и фронтенда. Скрипт сборки образов предполагает, что репозитории были клонированы в одну папку и проект фронтенда находиться в папке gtgo-frontend.

Сборка образов запускается с помощью скрипта **build-images.sh**, находящегося в корне папки проекта бэкенда. Данный скрипт собирает проекты фронтенда и бэкенда и создает четыре Docker образа:

— glio-trace-db – образ PostgreSQL с созданной БД;

— glio-trace – образ сервера приложений;

— glio-trace-node – образ вычислительного узла;

— gtgo-frontend – образ, содержащий клиентскую часть под управлением веб-сервера Nginx

Образы могут быть сохранены в файлы для переноса на сервера, где необходимо развернуть систему, примеры команд сохранения/загрузки образов:

— сохранение образа в файл: sudo docker save -o glio-trace.img glio-trace;

— загрузка образа из файла: sudo docker load -i glio-trace.img glio-trace.

Для развертывания системы для промышленной эксплуатации рекомендуется использовать образы, которые автоматически собираются на системе сборки Gitlab и сохраняются, как артефакты задач сборки.

1.2.4 Получение и настройка SSL сертификатов

Для работы сервера приложений по защищенному протоколу HTTPS требуется получить и настроить SSL сертификаты. Получение и настройка SSL сертификатов осуществляется администратором Программы, в зону ответственности разработчика не входит.

В качестве примера в данном разделе рассмотрено получение и настройка сертификатов, выдаваемых сервисом Let’s encrypt, который выдает бесплатные сертификаты SSL со сроком действия 3 месяца. При истечении срока действия может быть получен новый сертификат.

На сервере, на котором будет развернут сервер приложений, необходимо установить утилиту certbot для взаимодействия с сервисом Let’s encrypt. Для установки в терминале необходимо выполнить следующие команды:

sudo apt-get update
sudo apt-get install software-properties-common

sudo add-apt-repository universe
sudo add-apt-repository ppa:certbot/certbot
sudo apt-get update
sudo apt-get install certbot

Для получения сертификата необходимо выполнить команду:

sudo certbot certonly --standalone -d <DOMAIN_NAME>

Здесь <DOMAIN_NAME> должно быть заменено на доменное имя, к которому привязан сервер приложений. Также при запуске этой команды должен быть свободен 80-й, порт, который используется утилитой для проверки доменного имени.

Для обновления сертификата требуется остановить контейнер клиентской части, содержащий веб-сервер командой:

sudo docker stop gtgo-frontend

Обновить сертификат с помощью команды:

certbot renew

После успешного обновления сертификата запустить сервер приложений командой:

sudo docker start gtgo-frontend

При успешном обновлении система будет доступна по протоколу HTTPS по адресу: https://<DOMAIN_NAME>

1.3 Установка и настройка Программы

Для развертывания Программы необходимо предварительно выполнить операции, описанные в подразделе 1.21.

После выполнения операций, описанных в подразделе 1.21 необходимо скопировать в файловое хранилище (произвольная папка, которая должна быть доступна по сети серверу приложений и всем вычислительным узлам) файлы, необходимые для работы приложения:

— файлы референсного генома и их индексы;

— файлы баз данных, необходимые для аннотации;

— конфигурационные файлы построения клинического отчета.

Набор этих файлов поддерживается в актуальном состоянии на тестовом сервере ox.nprog.local.

Произвести запуск вычислительных узлов командой:

```
docker run --network=host --detach \
--mount 'type=bind,src=<PATH_TO_STORAGE>,dst=/root/gliotrace' \
-e "IGNITE_TCP_DISCOVERY_ADDRESSES=<BASE_CLUSTER_IP>" \
--restart=unless-stopped \
--name glio-trace-node glio-trace-node
```

Здесь <PATH_TO_STORAGE> должен быть заменен на путь к файловому хранилищу. Минимум один вычислительный узел должен быть выбран в качестве координатора кластера. В соответствии с документацией на Apache Ignite в качестве координатора кластера рекомендуется использовать 2-3 вычислительных узла. IP адреса данных узлов (перечисленные через запятую) должны быть подставлены вместо <BASE_CLUSTER_IP>.

Если несколько вычислительных узлов запускаются на одном сервере, то нужно дать им различные имена и добавить параметры запуска, ограничивающие использование памяти и количество используемых ядер, чтобы предотвратить возможность захвата большей части вычислительных ресурсов одним узлом. Для этого необходимо выполнить команду:

```
docker run --network=host --detach \
--mount 'type=bind,src=<PATH_TO_STORAGE>,dst=/root/gliotrace' \
-e "IGNITE_TCP_DISCOVERY_ADDRESSES=<BASE_CLUSTER_IP>" \
-m="32g" --cpus="4" \
--restart=unless-stopped \
--name glio-trace-node glio-trace-node
```

В данном примере вычислительный узел ограничивается использованием 32Gb оперативной памяти и 4 ядер процессора.

Выполнить запуск сервера приложений командой:

```
docker run --network=host --detach \
--mount  'type=bind,src=<PATH_TO_STORAGE>,dst=/root/gliotrace' \
--mount 'type=bind,src=<PATH_TO_CERTIFICATES>,dst=/etc/letsencrypt/live/genom
enal.com' \
-e "IGNITE_TCP_DISCOVERY_ADDRESSES=<BASE_CLUSTER_IP>" \
-e "DB_PWD=<DATABASE_PASSWORD>" \
-e "PROFILES=prod,ssl" \
--restart=unless-stopped \
--name glio-trace glio-trace
```

Для выполнения команды следует заменить текст:

— <PATH_TO_STORAGE> – путь к файловому хранилищу;

— <PATH_TO_CERTIFICATES> – путь к файлам SSL сертификата (получение и установка описаны в подпункте 1.2.4.), по умолчанию /etc/letsencrypt/live/<DOMAIN_NAME>;

— <BASE_CLUSTER_IP> – список IP адресов вычислительных узлов, которые используются в качестве координаторов кластера;

— <DATABASE_PASSWORD> – пароль к базе данных, который был указан при ее создании. Если PostgreSQL установлен на отдельном сервере отличном от сервера приложений, то также нужно передать следующий параметр:

-e DB_URL="jdbc:postgresql://<HOST>:<PORT>/gliotrace",

где в качестве <HOST> и <PORT> нужно указать адрес и порт, на котором СУБД PostgreSQL принимает соединения.

При первом запуске Программы будет создана структура базы данных, загружены данные по хромосомам и генам, а также создан пользователь admin с паролем P@ssword.

Запуск образа Jupyter Hub выполняется с помощью команды:

```
docker run --network=host --detach \
--mount 'type=bind,src=<PATH_TO_STORAGE>/jupyter,dst=/notebooks' \
--restart=unless-stopped \
--name jupyter jupyter
```

2. ПРОВЕРКА ПРОГРАММЫ

Для проверки корректности установки Программы необходимо в адресной строке web-браузера ввести адрес сервера приложений https://<DOMAIN_NAME>. При этом должна открыться страница входа в Программу (Рисунок 7).

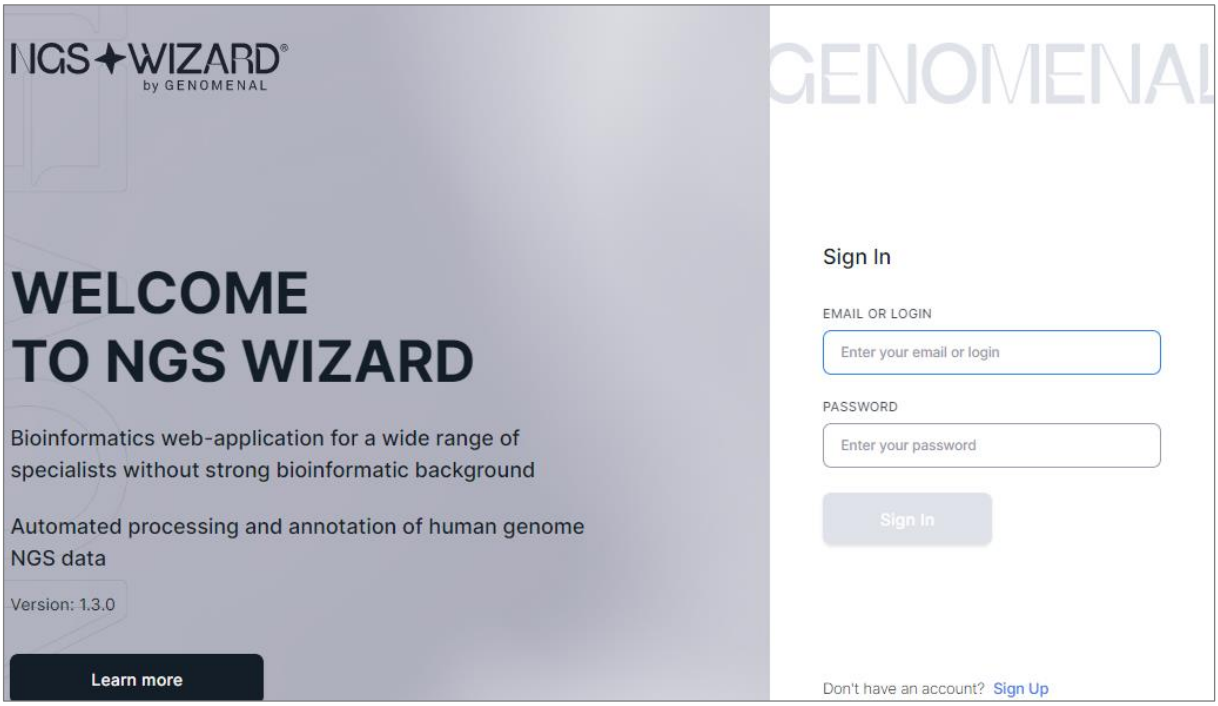

Рисунок 7 — Страница входа в Программу

Для программной проверки доступности Программы (например, для систем мониторинга и нотификации администрации о недоступности системы) может использоваться метод API, доступный по адресу https://<DOMAIN_NAME>/api/health, который при нормальной работе системы, возвращает HTTP статус 200 и JSON ответ следующего содержания:

{
"annotation": {
"1KG": "2019-02-26",
"CLINVAR": "2020-07-06","COSMIC": "2020-04-07 (ver 91)",
"DBNSFP": "2019-12-05 (ver 4.0c)",
"DBSNP": "2019-07-22 (ver 153)",
"ENSEMBL": "ver 99",
"GNOMAD3": "ver 0.2.24"
},

"commitHash": "ce20919a",
"status": "OK",
"timestamp": "2020-10-06T08:47:59.870+0000",
"version": "1.3.0"
}

3. ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ

3.1 Система сбора логов и мониторинга

Система централизованного сбора логов основывается на следующих продуктах:

— Filebeat - Сбор и отправка логов в хранилище;

— Elasticsearch - Хранение и обработка логов;

— Kibana - пользовательский интерфейс для поиска и анализа логов.

На серверах, на которых необходимо собирать логи, требуется развернуть filebeat, для чего следует:

— добавить в /etc/hosts адрес elasticsearch, где будут храниться логи;

— выполнить команду:

docker-compose up -d filebeat.

На сервере, который планируются использовать для хранения и обработки логов, необходимо развернуть elasticsearch и kibana. Для этого:

— поверить, что vm.max_map_count >= 262144. Для проверки использовать инструкцию с официального сайта https://www.elastic.co/guide/en/elasticsearch/reference/current/vm-max-map-count.html

— выполнить команду:

docker-compose up -d elasticsearch kibana

При этом UI Kibana будет доступен на порту 5061.

Конфигурации Kibana можно загрузить в Management -> Saved Objects -> Import. Имеется возможность выбрать конфигурацию из папки /glio-trace-monitoring/log-collector.

3.2 Система сбора и анализа метрик

Система сбора и анализа метрик состоит из трех компонентов:

— Экспортеры (https://github.com/google/cadvisor, https://github.com/prometheus/node_exporter, https://github.com/wrouesnel/postgres_exporter);

— Prometheus (https://prometheus.io/);

— Grafana (https://grafana.com/).

На серверах, с которых требуется собирать метрики необходимо:

— запустить cadvisor командой:

docker-compose up -d cadvisor

— установить node-exporter следуя инструкциям официального сайта: https://devopscube.com/monitor-linux-servers-prometheus-node-exporter/

— на сервере с базой данных запустить postgres-exporter командой:

docker-compose up –d postgres-exporter

— развернуть Prometheus, для чего в конфигурационном файле Prometheus.yml указать IP-адреса, с которых требуется собирать метрики и выполнить команду:

docker-compose up -d

UI Prometheus будет доступен на порту 9090, UI Grafana будет доступен на порту 3000. Стандартный логин/пароль UI Grafana: admin/admin. Перейти в Configuration-> Data Sources-> Add data source -> Prometheus.

— указать адрес url prometheus;

— перейти в Create -> Dashboard -> Dashboard Settings -> JSON Model и вставить содержимое файла:

glio-trace-monitoring/metrics-monitor/grafana-dashboards/ox-dashboard.json

Конфигурационные файлы и актуальная инструкция по настройкам этих систем находится в репозитории: http://relativity.nprog.local/novel-dev/glio-trace-monitoring.

3.3 Создание резервных копий и восстановление из резервной копии

Для создания резервной копии базы данных используется утилита pg_dump. Для создания резервной копии необходимо выполнить команду, от имени пользователя postgres:

sudo -u postgres -i
pg_dump -U gliotrace -W -d gliotrace > <BACKUP_FILE_PATH>

При выполнении команды, необходимо будет указать пароль к базе данных, заданный при ее создании. В качестве <BACKUP_FILE_PATH> указать путь к создаваемому файлу резервной копии.

Для восстановления базы данных из созданной резервной копии необходимо выполнить команду от имени пользователя postgres:

sudo -i -u postgres
psql -U gliotrace -d gliotrace -W < <BACKUP_FILE_PATH>

При выполнении команды, необходимо будет указать пароль к базе данных, заданный при ее создании. В качестве <BACKUP_FILE_PATH> указать путь к созданному ранее файлу резервной копии.

Резервная копия данных хранилища, может быть выполнена простым копированием файлов хранилища на резервный носитель. Так как файлы образцов и результатов могут занимать большие объемы, при создании резервной копии хранилища рекомендуется архивация данных.

4. СООБЩЕНИЯ СИСТЕМНОМУ ПРОГРАММИСТУ

Коды ошибок, которые может вернуть API NGS Wizard при обращении к Программе (данные ошибки также регистрируются в логах приложения):

ACCESS_DENIED — пользователь пытается выполнить операцию, на которую у него нет прав (например, пользователь, не имеющий прав администратора, пытается получить список активных выполняющихся задач);

AUTHENTICATION_REQUIRED — попытка доступа без JWT токена авторизации, либо с токеном, у которого истек срок годности;

BAD_CREDENTIALS — попытка входа в систему с неправильным логином или паролем;

VIEWER_REJECTED — попытка пользователя, у которого есть только права на чтение (используется для демонстрационных аккаунтов), выполнить операцию на создание или изменение объекта;

LOGIN_IN_USE — при регистрации пользователь указывает логин, который уже используется другим пользователем Программы;

EMAIL_IN_USE — при регистрации пользователь указывает email, который уже используется другим пользователем Программы;

INVITATION_CODE_INVALID — при регистрации указан неверный секретный код приглашения для регистрации;

UNABLE_TO_DELETE_DEFAULT_PRESET — попытка удалить набор настроек, который выбран как набор настроек для использования по умолчанию;

UNKNOWN_SETTING_ID — при попытке обновить значение настройки, передан неверный идентификатор настройки;

WRONG_SETTING_VALUE — при попытке обновить значение настройки, передано некорректное значение (например, строка, для настройки числового типа);

CONFIGURATION_PRESETS_IS_EMPTY — попытка создать конфигурацию настроек, в которую не добавлено ни одного набора настроек;

PATIENT_IS_ARCHIVED — попытка запуска на обработку образца пациента, который был перемещен в архив;

AMBIGUOUS_GENE_REQUEST — при создании панели генов из списка, в списке есть имя гена, по которому невозможно однозначно определить ген (существуют гены имеющие одинаковые имена по классификации Ensembl), в этом случае запрос нужно уточнить, заменив неоднозначное имя гена на Ensembl ID;

REDUNDANT_GENE_REQUEST — при создании панели генов из списка, в списке один и тот же ген встречается несколько раз;

PATIENT_CODE_IN_USE — при добавлении нового пациента указан код пациента, который уже используется для созданного ранее пациента;

COHORT_NAME_IN_USE — при добавлении новой когорты указано имя, которое уже используется для имеющейся в системе когорты;

BAD_REQUEST — общая ошибка, указывающая на некорректность запроса (подробности передаются в тексте сообщения);

INTERNAL_SERVER_ERROR — внутренняя ошибка сервера, которая как правило означает ошибку в коде приложения (подробности передаются в тексте сообщения);

<ENTITY_TYPE>_NOT_FOUND — – набор ошибок, которые возвращаются при попытке выполнить операцию с сущностью по идентификатору, но в системе нет сущности с указанным идентификатором, полный список ошибок данного типа: SOURCE_FILE_NOT_FOUND, RESULT_FILE_NOT_FOUND, TASK_NOT_FOUND, PANEL_NOT_FOUND, SAMPLE_SET_NOT_FOUND, SETTINGS_PRESET_NOT_FOUND, CONFIGURATION_NOT_FOUND, DEFAULT_SETTINGS_PRESET_NOT_FOUND, PATIENT_NOT_FOUND, COHORT_NOT_FOUND, GROUP_ANALYSIS_NOT_FOUND, GENE_PANEL_NOT_FOUND, USER_FILTER_CONDITION_NOT_FOUND, GENE_NOT_FOUND.

Перечень сокращений

Сокращение	Расшифровка
ДНК	Дизоксирибонуклеиновая кислота
СПО	Специальное программное обеспечение
СУБД	Система управления базами данных
БД	База данных
API	Application program interface
REST	Representation State Transfer
JSON	JavaScript Object Notation
JWT	JSON Web Token
SSL	Secure Sockets Layers

Лист регистрации изменений

Изм	Номера листов (страниц)				Всего листов (страниц) в документе	№ документа	Входящий № сопроводит. документа и дата	Подп.	Дата
	Измененных	Замененных	Новых	Аннулированных					