

**Инструкция по установке** **Customer delivery system (cds)**

В качестве CI-CD инструмента используется jenkins, Jenkinsfile для каждого сервиса расположен в дополнительной папке - cicd

Для каждого сервиса создаётся Job – Multubranch Pipleline, в настройках которой нужно только ввести ссылку на git репозиторий и выставить параметр Property strategy – Named branches get different properties



В Jenkins должен быть установлен плагин - Config File Management. В него нужно поместить settings.xml с настройками для подключения к Nexus серверу, пример в папке cicd.



В Jenkins в глобальных переменных должны быть указаны параметры адрес, логин и пароль от MAVEN репозитория:

* docker\_registry\_address
* \*\*\*ADDRESS\*\*\*:\*\*\*PORT\*\*\*
* maven\_login
* \*\*\*LOGIN\*\*\*
* maven\_pass
* \*\*\*PASSWORD\*\*\*

Версии используемых базовых Docker образов прописаны в Dockerfile папке cicd для каждого проекта.

Для всего проекта нужен postgresql сервер, если требуется его поднять внутри кластера kubernetes то можно использовать готовый helm шаблон, стоит только сменить пароли указанные на строках 28, 142, 151, 152, 153 и если требуется сменить nodePorts то его можно поменять на строке 592 в файле values.yaml в папке с helm шаблоном в каталоге cicd. Для разворачивания postgresql сервер а нужно скопировать папку с helm шаблоном на ноду кластера и из неё выполнить команду:

* helm install -f values.yaml -n \*\*\*NAMESPACE\*\*\*

Сервер развернётся в нужном namespace kubernetes кластера.

**auth-server**

GIT: <http://samara-soft.pro/gitlab/logistic/auth-server.git>

Сборка образа и его последующий деплой в kubernetes. В папке cicd/auth-server/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-dr-updater**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-dr-updater.git>

Сборка образа. В папке cicd/cds-dr-updater/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-fulltext-search**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-fulltext-search.git>

Сборка образа и его последующий деплой в kubernetes. В папке cicd/cds-fulltext-search/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-memcached**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-memcached.git>

Делой memcached в kubernetes для проекта.

**cds-mobile-service**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-mobile-service.git>

Сборка образа и деплой в kubernetes. В папке cicd/cds-mobile-service/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться образы для деплоя сервиса.

**cds-osrm**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-osrm.git>

Деплой в kubernetes маршрутизатора. В папке cicd/cds-osrm/ есть helm шаблон используемый в проекте. В настройках есть параметры отвечающие за скачивание карт для определённого региона, например для скачивания южного федерального округа расомментирована строка 159 в values.yaml. Для разворачивания osrm нужно скопировать папку с helm шаблоном на ноду кластера и из неё выполнить команду:

* helm install -f values.yaml -n \*\*\*NAMESPACE\*\*\*

Сервис развернётся в нужном namespace kubernetes кластера.

**cds-web-api**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-web-api.git>

Сборка зависимостей и образов для cds-ui. В файле Jenkinsfile в stage «Generate proto js» требуется образ содержащий maven, protoc, gen, grpc в каталоге cicd/cds-web-api/ лежит Dockerfile с инструкциями для сборки этого образа, а так же скаченные нужные зависимости. Готовый образ следует залить в nexus для использования его при сборке зависимостей. В строке 59 в Jenkinsfile как раз указан этот кастомный образ.

Так же для того чтобы зависимости загружались после сборки в nexus сервер в плагине Config File Management нужно создать фаил с именем - 1d629117-b147-4c06-b1bd-b0f28d5c581d (можно назвать по другому, но тогда следует в строке 83 в Jenkinsfile его тоже поменять) с содержимым как в файле cicd/cds-web-api/.npmrc, и ввести там свой пароль от репозитория NPM.

Для работы envoy в репозитории в каталоге config в файлах logistic-dev.json и logistic-prod.json (в зависимости от ветки репозитория) указаны адреса backend куда должны подключаться сервисы. А так же в корне репозитория файлы logistic-dev.config.js и logistic-prod.config.js а которых указаны что именно в файлах описанных выше прописаны адреса сервисов. Примеры всех этих файлов есть в cicd/cds-web-api/.

**cds-ui**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-ui.git>

Деплой в kubernetes сервиса cds-ui используя образ созданный после работы Job - cds-web-api.

**grpc-envoy**

GIT: <http://samara-soft.pro/gitlab/logistic/grpc-envoy.git>

Сборка sidecar grpc-envoy для использования в проекте.

**planning-time**

GIT: <http://samara-soft.pro/gitlab/logistic/planning-time.git>

Сборка образа planning-time. В папке cicd/planning-time/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера на который будет заливаться готовый образ после сборки.

**cds-android**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-android.git>

Сборка APK приложения для android. Для сборки нужен образ в котором

В файле Jenkinsfile в stage «Build Android Apk» требуется образ содержащий android sdk, gradle, java в каталоге cicd/cds-android/ лежит Dockerfile-ubuntu с инструкциями для сборки этого образа. Готовый образ следует залить в nexus для использования его при сборке зависимостей. В строке 79 в Jenkinsfile как раз указан этот кастомный образ. Далее в stage «Build nginx image with Apk» собирается образ nginx в который кладётся APK, после в stage «Deploy image in Kubernetes» этот образ деплоится в kubernetes как web страничка с которой можно скачать по прямой ссылке этот готовый APK.

**cds-dr-bridge**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-dr-bridge.git>

Сборка образа и его последующий деплой в kubernetes. В папке cicd/cds-dr-bridge/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-rfp**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-rfp.git>

Сборка образа и его последующий деплой в kubernetes. В папке cicd/cds-rfp/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-persistence**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-persistence.git>

После установки postgresql сервера для проливки миграций БД. Это первая job которую нужно запустить перед всеми остальными.

**cds-external**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-external.git>

Сборка образа и его последующий деплой в kubernetes. В папке cicd/cds-external/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-mq**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-mq.git>

Делой cds-mq в kubernetes для проекта.

**cds-notify**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-notify.git>

Сборка образа и его последующий деплой в kubernetes. В папке cicd/cds-notify/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-print**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-print.git>

Сборка образа и его последующий деплой в kubernetes. В папке cicd/cds-print/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-proto**

GIT: <http://samara-soft.pro/gitlab/logistic/cds-proto.git>

Сборка ключевых зависимостей сервисов. Эту Job стоит запускать до сборки и деплоя остальных сервисов. В папке cicd/cds-proto/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

**cds-route-optimizer**

GIT: http://samara-soft.pro/gitlab/logistic/cds-route-optimizer.git

Сборка образа и его последующий деплой в kubernetes. В папке cicd/cds-route-optimizer/ есть пример pom.xml, в нём нужно прописать адрес nexus сервера из которого будут подтягиваться зависимости для сборки проекта.

В kubernetes для тогда чтобы Jobы могли выкачивать из nexus репозитория готовые образы для деплоя нужно создать secret, где прописан адрес этого репозитория(в BASE64 кодировке). Пример манифеста лежит в cicd/deploy/ фаил regcred.yaml. Так же для установки валидного сертификата для https сервисов нужно создать секрет с содержимым TLS сертификата – пример фаил tzoom-pro.yaml. Установка этих манифестов выглядит так:

* kubectl apply –f regcred.yaml
* kubectl apply –f tzoom-pro.yaml