Konveyor / Tackle

Tackle Config Discovery

Konveyor Projects

Rehost virtual machines to KubeVirt

Rehost apps between Kubernetes clusters

Replatform applications to Kubernetes

Refactor applications for Kubernetes

Measure software delivery performance

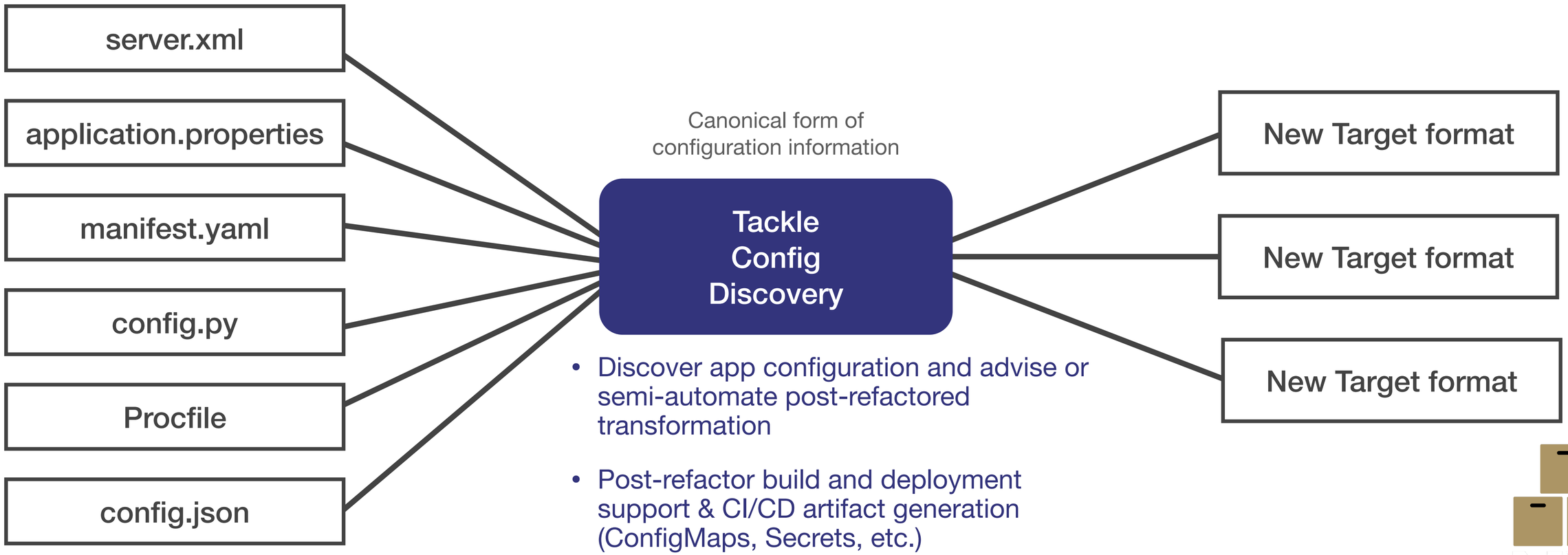
You just broke up your monolith into microservices!

Where does the configuration info go?

Tackle Config Discovery

Overview

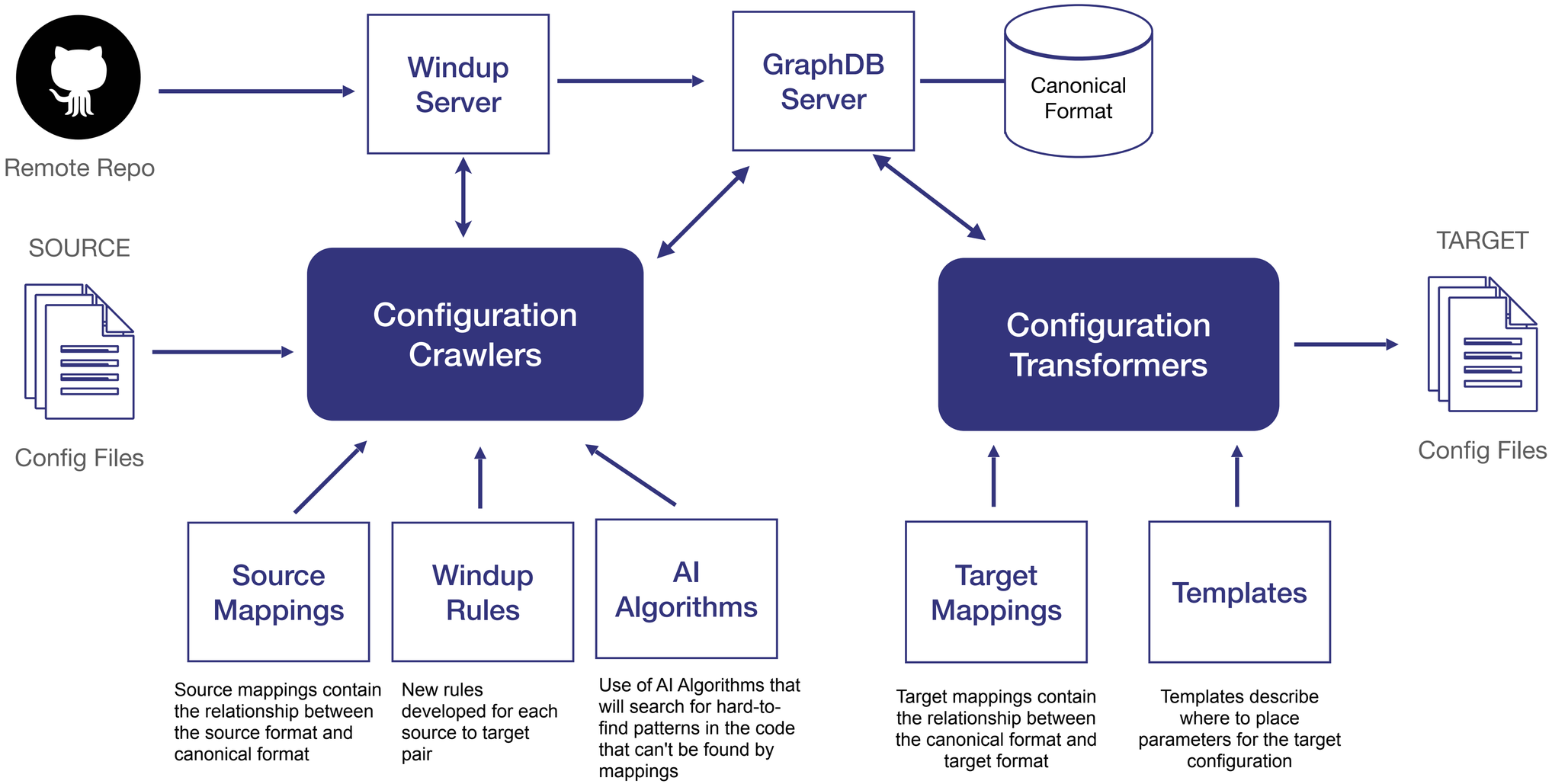
As an Architect I'd like to discover existing configuration parameters so that I can semi-automate the post-refactor build



- Discover app configuration and advise or semi-automate post-refactored transformation
- Post-refactor build and deployment support & CI/CD artifact generation (ConfigMaps, Secrets, etc.)

Tackle Config Discovery

Architecture



Config Discovery Example

DayTrader 7 Java EE xml config file example:

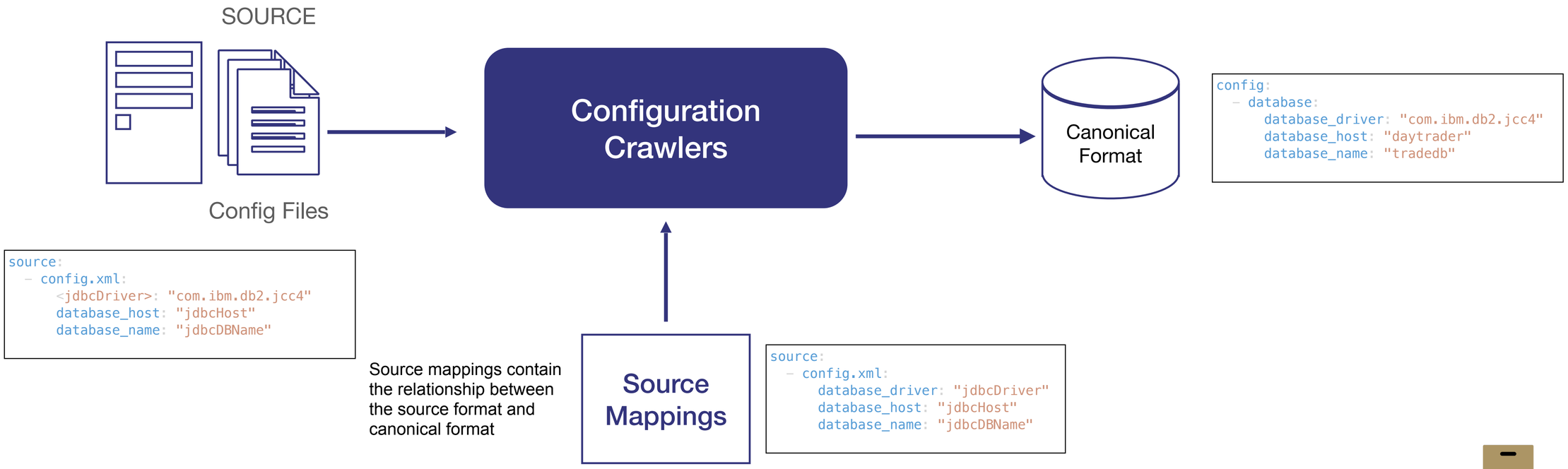
```
./pom.xml
./daytrader-ee7/pom.xml
./daytrader-ee7/bin/main/META-INF/application.xml
./daytrader-ee7/src/main/application/META-INF/application.xml
./daytrader-ee7-ejb/pom.xml
./daytrader-ee7-ejb/bin/main/META-INF/ibm-ejb-jar-bnd.xml
./daytrader-ee7-ejb/bin/main/META-INF/persistence.xml
./daytrader-ee7-ejb/bin/main/META-INF/ejb-jar.xml
./daytrader-ee7-ejb/src/main/resources/META-INF/ibm-ejb-jar-bnd.xml
./daytrader-ee7-ejb/src/main/resources/META-INF/persistence.xml
./daytrader-ee7-ejb/src/main/resources/META-INF/ejb-jar.xml
./daytrader-ee7-web/pom.xml
./daytrader-ee7-web/bin/main/META-INF/persistence.xml
./daytrader-ee7-web/src/main/webapp/WEB-INF/ibm-web-bnd.xml
./daytrader-ee7-web/src/main/webapp/WEB-INF/classes/META-INF/persistence.xml
./daytrader-ee7-web/src/main/webapp/WEB-INF/beans.xml
./daytrader-ee7-web/src/main/webapp/WEB-INF/faces-config.xml
./daytrader-ee7-web/src/main/webapp/WEB-INF/web.xml
./daytrader-ee7-web/src/main/webapp/WEB-INF/ibm-web-ext.xml
./daytrader-ee7-web/src/main/java/META-INF/persistence.xml
./daytrader-ee7-wlpcfg/pom.xml
./daytrader-ee7-wlpcfg/servers/daytrader7Sample/server.xml
```

Summary	
pom.xml	5
persistence.xml	5
application.xml	2
ejb-jar.xml	2
ibm-ejb-jar-bnd.xml	2
beans.xml	1
faces-config.xml	1
ibm-web-bnd.xml	1
ibm-web-ext.xml	1
server.xml	1
web.xml	1

Configuration Crawlers

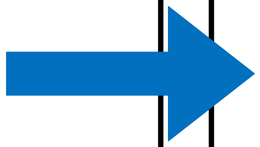
Source Mapping

Config crawler uses the source mapping file to determine how the values in each of the source files gets mapped into the canonical format and stores it.



Mapping Example

```
<server description="DayTrader Processor Server">

    <httpEndpoint httpPort="9080" httpsPort="9443" id="defaultHttpEndpoint" />

    <keyStore id="defaultKeyStore" password="Liberty" />

    <webApplication id="DayTraderProcessor" location="DayTraderProcessor.war"
       name="DayTraderProcessor"/>

    <connectionManager agedTimeout="-1" connectionTimeout="0" id="conMgr1"
       maxIdleTime="-1" maxPoolSize="100" minPoolSize="100"
       purgePolicy="FailingConnectionOnly" reapTime="-1"/>

    <jdbcDriver id="DB2Driver" libraryRef="DB2JCC4Lib"/>

    <library id="DB2JCC4Lib" filesetRef="DB2Fileset" />

    <fileset id="DB2Fileset"  dir="${shared.resource.dir}/db2jars"
       includes="db2jcc4-10.1.jar"/>

    <authData id="TradeDataSourceAuthData" password="${env.DB_PASSWORD}"
       user="${env.DB_USER}"/>

    <dataSource jndiName="jdbc/TradeDataSource" jdbcDriverRef="DB2Driver"
       connectionManagerRef="conMgr1" id="DefaultDataSource"
       isolationLevel="TRANSACTION_READ_COMMITTED" statementCacheSize="60">
        <properties.db2.jcc databaseName="${env.DB_DATABASE}" serverName="$
       {env.DB_HOST}" portNumber="${env.DB_PORT}" user="${env.DB_USER}"
       password="${env.DB_PASSWORD}" />
    </dataSource>
</server>
```



```
server:
  description: "DayTrader Processor Server"
  httpEndpoint:
    httpPort: 9080
    httpsPort: 9443
    id: defaultHttpEndpoint
  keyStore:
    id: defaultKeyStore
    password: Liberty
  webApplication:
    id: DayTraderProcessor
    location: "DayTraderProcessor.war"
    name: DayTraderProcessor
  jdbcDriver:
    id: DB2Driver
    libraryRef: DB2JCC4Lib
  library:
    id: DB2JCC4Lib
    filesetRef: DB2Fileset
  fileset:
    id: DB2Fileset
    dir: "${shared.resource.dir}/db2jars"
    includes: "db2jcc4-10.1.jar"
  authData:
    id: TradeDataSourceAuthData
    password: "${env.DB_PASSWORD}"
    user: "${env.DB_USER}"
  dataSource:
    jndiName: "jdbc/TradeDataSource"
    jdbcDriverRef: DB2Driver
    connectionManagerRef: conMgr1
    id: DefaultDataSource
    isolationLevel: TRANSACTION_READ_COMMITTED
    statementCacheSize: 60
    properties.db2.jcc:
      databaseName: "${env.DB_DATABASE}"
      serverName: "${env.DB_HOST}"
      portNumber: "${env.DB_PORT}"
      user: "${env.DB_USER}"
      password: "${env.DB_PASSWORD}"
```

Quarkus Database Config

application.properties

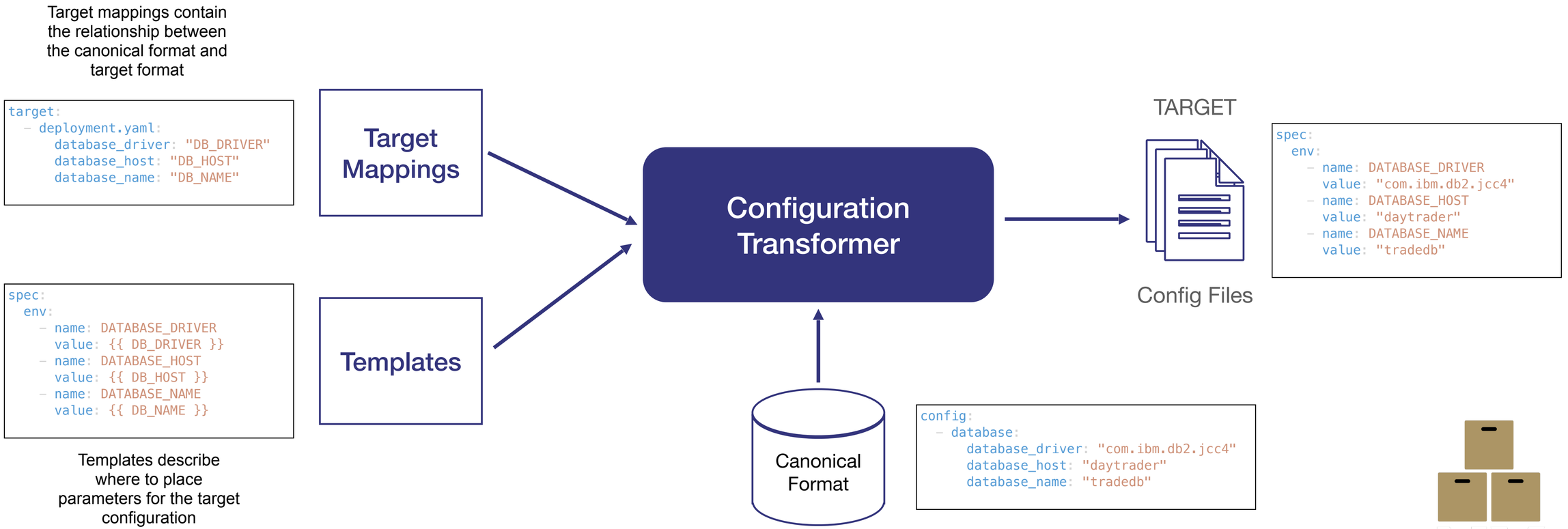
https://quarkus.io/guides/all-config

quarkus.datasource.jdbc	boolean
quarkus.datasource.jdbc.driver	string
quarkus.datasource.jdbc.transactions	enabled, xa, disabled
quarkus.datasource.jdbc.enable-metrics	boolean
quarkus.datasource.jdbc.url	string
quarkus.datasource.jdbc.initial-size	int
quarkus.datasource.jdbc.min-size	int
quarkus.datasource.jdbc.max-size	int
quarkus.datasource.jdbc.background-validation-interval	Duration
quarkus.datasource.jdbc.acquisition-timeout	Duration
quarkus.datasource.jdbc.leak-detection-interval	Duration
quarkus.datasource.jdbc.idle-removal-interval	Duration
quarkus.datasource.jdbc.max-lifetime	Duration
quarkus.datasource.jdbc.transaction-isolation-level	undefined, none, read-uncommitted, read-committed, repeatable-read, serializable
quarkus.datasource.jdbc.detect-statement-leaks	boolean
quarkus.datasource.jdbc.new-connection-sql.	string
quarkus.datasource.jdbc.validation-query-sql	string
quarkus.datasource.jdbc.pooling-enabled	boolean
quarkus.datasource.jdbc.additional-jdbc-properties	Map<String,String>

Configuration Transformers

Artifact Generation

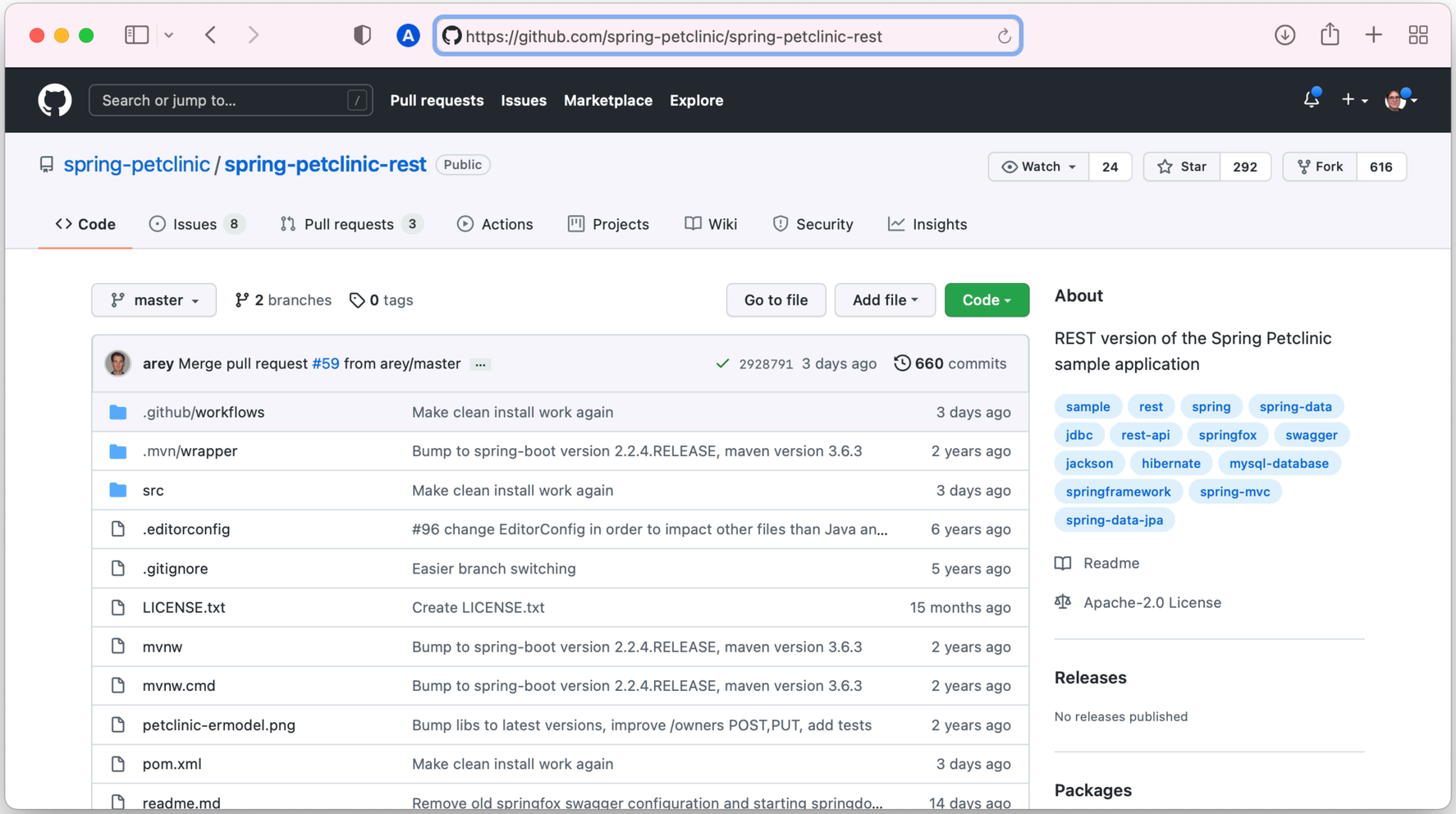
Config transformer takes the mapping file to get the configuration information out of the canonical store and uses the template file to create a new configuration file that inserts the correct values into the template



Live Demo

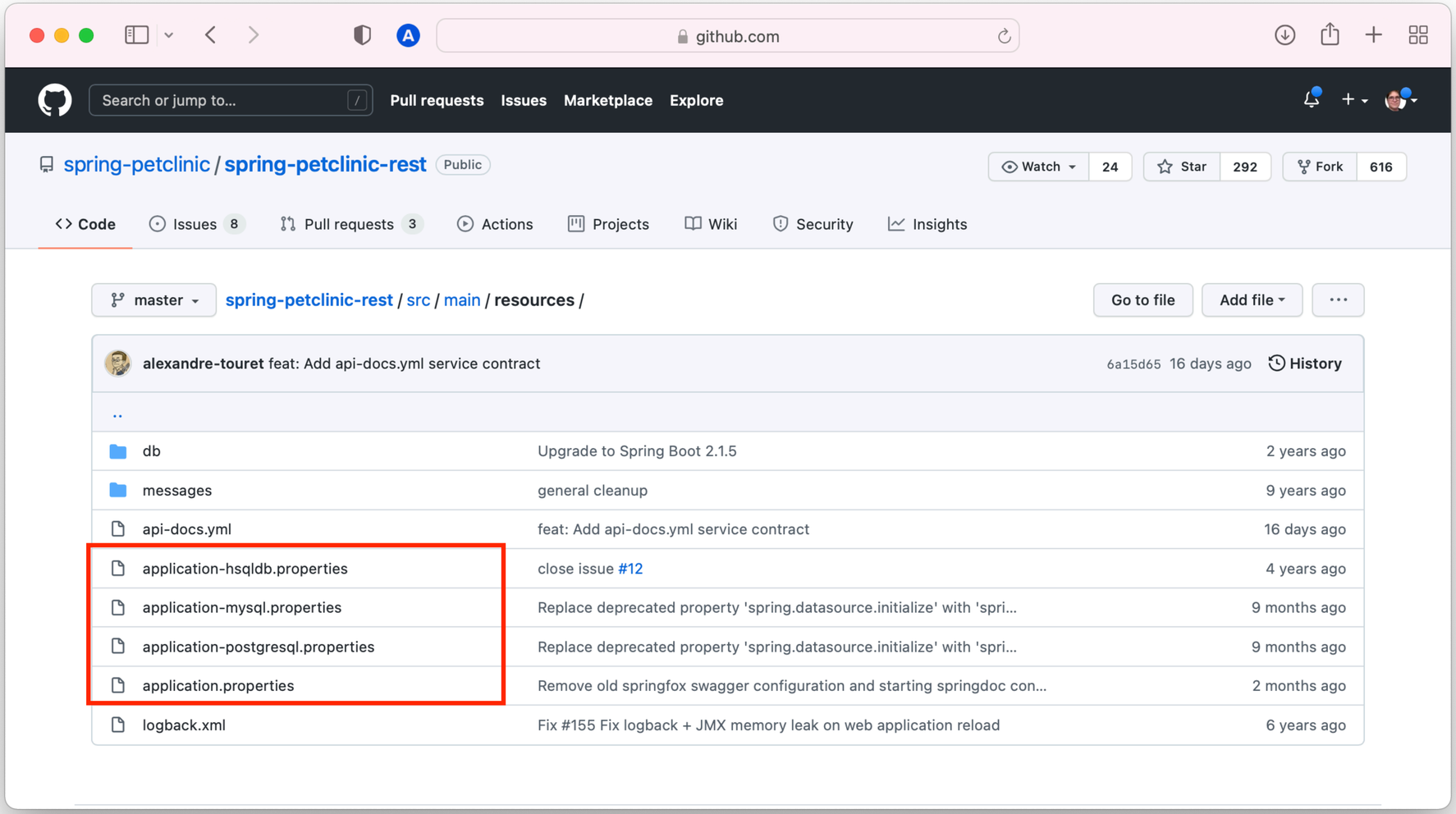
Convert SpringBoot to Quarkus

https://github.com/spring-petclinic/spring-petclinic-rest



Convert SpringBoot to Quarkus

Several properties files in this repo



TCD Command Line Interface


```
$ tcd --help
Usage: tcd [OPTIONS] COMMAND [ARGS]...

  Tackle Configuration Discovery.

Options:
  -v, --verbose  Show verbose output
  --help         Show this message and exit.

Commands:
  collect    Collect configuration files for a given framework.
  config     Displays the current configuration
  init       Initialize a new configuration.
  list       List the available frameworks.
  translate  Translate configuration into target framework.
```

TCD Init CLI

Initializes the common parameters for running discovery including setting the location of the windup server

```
$ tcd init -d ./out -w 127.0.0.1:8180
```

```
$ tcd init --help
Usage: tcd init [OPTIONS]

  Initialize a new configuration.

Options:
  -d, --data-directory PATH  Directory of configuration database
  -w, --windup TEXT          tcd-windup server host
  -s, --windup-ssl           Enable SSL for tcd-windup server
  --help                     Show this message and exit.
```

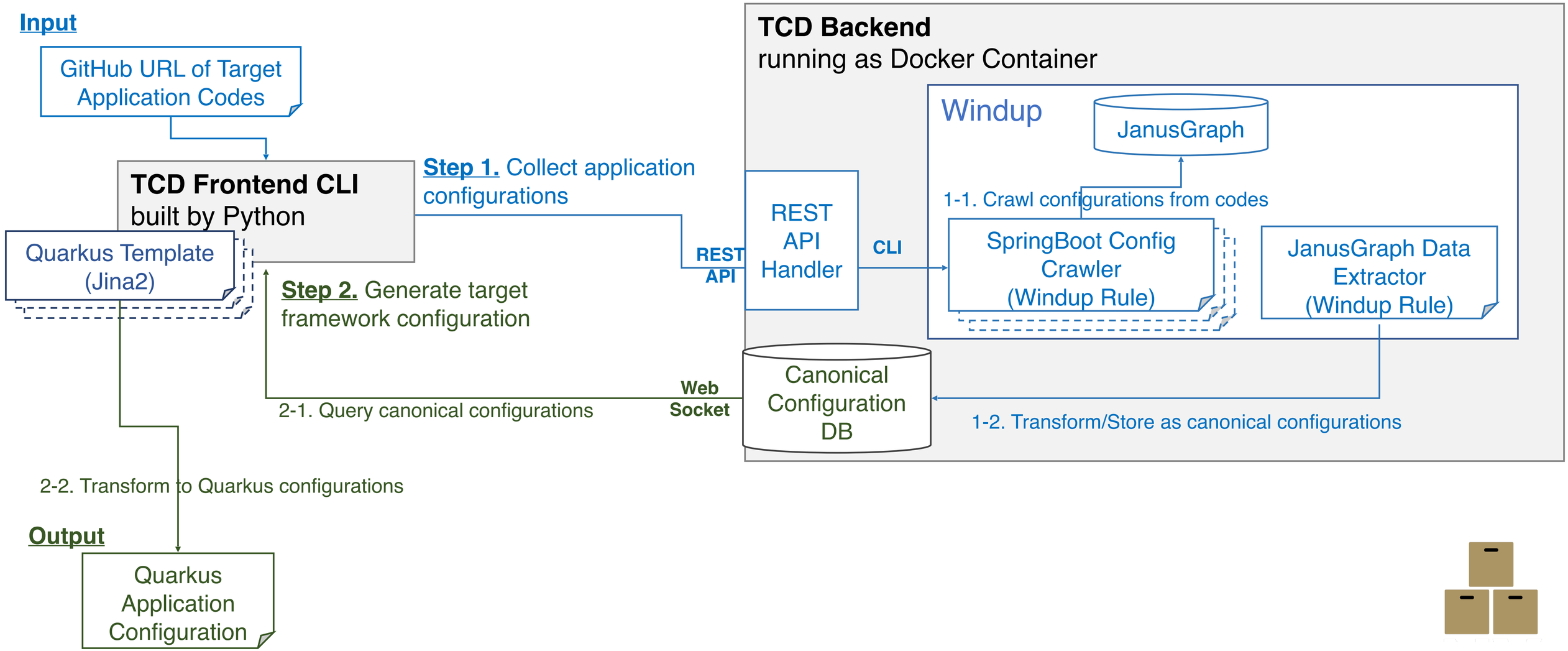
TCD Collect CLI

Collects all of the configuration files in the source folder for a given framework.

```
$ tcd collect ––framework liberty ––source ./src
```

```
$ tcd collect --help
Usage: tcd collect [OPTIONS]

  Collect configuration files for a given framework.

Options:
  -f, --framework TEXT            The application framework to look for
                                  [required]
  -c, --collector TEXT            The collector to use [file | windup]
  -s, --source TEXT               Source folder to scan
  -r, --repo TEXT                 Public git repository to scan
  -a, --all                       Collect all filenames
  -o, --output [yaml|json|ndjson]
                                  Output format for collected data
  --help                          Show this message and exit.
```

TCD Windup Collector



TCD Translate CLI

Translates configuration information found by a discovery run into a new target configuration

```
$ tcd translate ––framework quarkus ––target ./out
```

```
$ tcd translate --help
Usage: tcd translate [OPTIONS]

  Translate configuration into target framework.

Options:
  -f, --framework TEXT  The application framework to translate to  [required]
  -t, --target TEXT     Target folder to place config files
  --help                Show this message and exit.
```

Join the Konveyor Community

- **Chat**

  #konveyor on slack.k8s.io

  Join the Konveyor-community list

- **Share**

  Propose a meetup talk (form)

- **Contribute**

  Join the next quarterly project planning

  Invites sent to Konveyor-community list

www.konveyor.io

Thank you!

www.konveyor.io