SPINTA 2
SPINTA Tiesioginis diegimas į OS 5
SPINTA diegimas naudojant konteinerius 7
Šaltinių konfigūravimas 10
SPINTA agento konfigūravimas 11
Duomenų gavimo testavimas 17
Papildoma informacija 19

SPINTA

- SPINTA Tiesioginis diegimas į OS
- SPINTA diegimas naudojant konteinerius
- Šaltinių konfigūravimas
- SPINTA agento konfigūravimas
- Duomenų gavimo testavimas
- Tinklo konfigūravimas
- Papildoma informacija

Spinta yra Python biblioteka ir įrankių komplektas, skirtas duomenų struktūrų aprašymui, transformavimui ir teikimui pagal UDTS specifikaciją.

Pagrindiniai įrankiai:

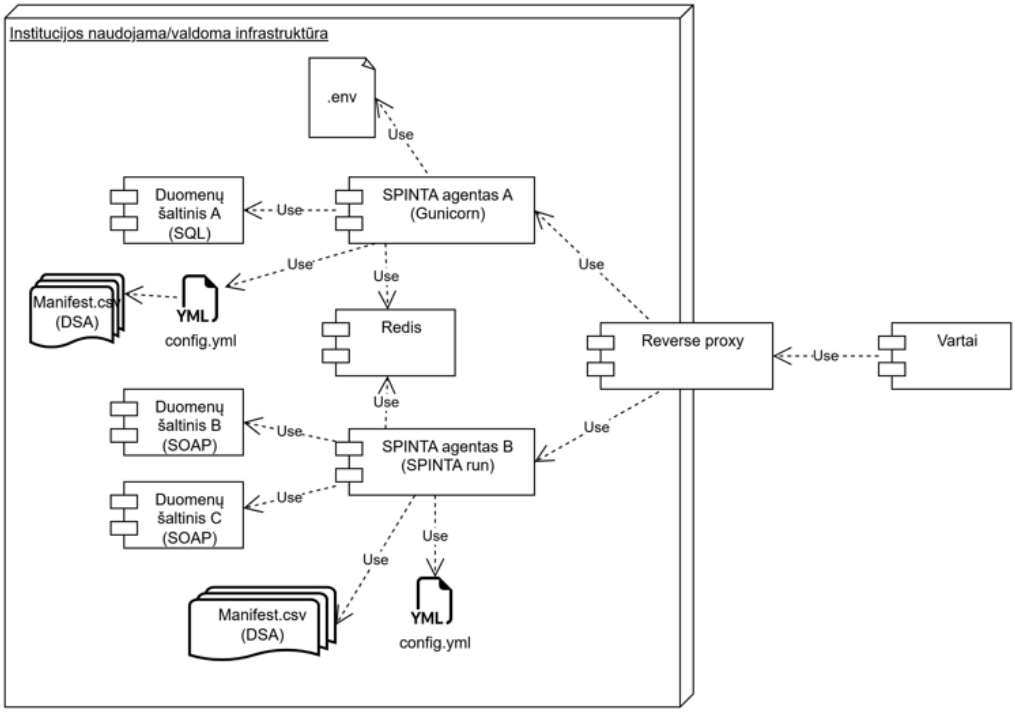
- **spinta inspect** – automatiškai nuskaito duomenų struktūras iš įvairių šaltinių (SQL, CSV, XML) ir generuoja pirminį DSA (duomenų struktūros aprašo) failą
- **spinta copy** – transformuoja ir valo DSA failus (pvz., generuoja DSA iš XSD schemos, ar iš JSON Schema, pašalina konfidencialius source duomenis prieš publikavimą, nustato prieigos lygius)
- **spinta run** – paleidžia Spinta serverį agento režimu, kuris teikia duomenis per REST JSON API pagal UDTS
- Kiti pagalbiniai įrankiai duomenų migravimui, testavimui ir WSDL/SOAP integracijoms

Diegimo alternatyvos

Spinta gali būti diegiama keliais būdais pagal projekto poreikius:

- **Tiesioginis diegimas į OS** – Python pip paketu (pip install spinta)
- **Docker konteineris**

Nepriklausomai nuo diegimo pateikiame abstraktų diegiamų komponentų vaizdą



Komponentas	Paskirtis	Komentaras
Reverse proxy	SSL layer. Spinta agentas veikia kaip webserveris tačiau jis neužtikrina TLS.	Institucija turi pati priimti sprendimą kaip užtikrinti HTTPS tarp Vartų ir SPINTA agento. Šiame dokumente pateikiama instrukcija kaip konfigūruoti nginx diegiant jį OS. Naudojant docker instrukcija nepateikiama.
SPINTA agentas	SPINTA run komandos arba `gunicorn` bibliotekos pagalba paleistas servisas	Duomenų tvarkytojas gali nuspręsti diegti daugiau nei viena agenta aptarnauti užklausas. Kaip matyti diagramoje SPINTA agentas gali dirbti su vienu ar daugiau šaltinių ir naudoti viena ar daugiau DSA failų. Daugeliu atveju rekomenduojama leisti spintos serverį `gunicorn` pagalba, tačiau WSDL serviso atveju reikia konfigūruoti ir leisti servisą `spinta run`
SPINTA config.yml	Tai pagrindinis konfigūracinis failas, kurį naudoja agentas. Failo vieta ir pavadinimas nurodomas sisteminiame	Failą sukurti ir užpildyti turi sistemos adminitratoriai diegimo metu. Šiame dokumente pateikiami pavyzdžiai bei konfigūracijos parametrų reikšmės.

	kintamajame `SPINTA_CONFIG`	DSA failai susiejami per manifestus.
REDIS ⚠ KRITINIS ELEMENTAS	Cache bei keymap.db. Keymap.db saugo globalių ir šaltinio raktų poras. Globalūs raktai sukuriami pirmo rakto panaudojimo metu. Praradus ar sugadinus failus bus prarasti išoriniai raktai.	Adminitratorius privalo užtikrinti, kad redis failai nebūtų prarasti ar sugadinti. Jei institucija jau tutri Redis ar gali naudoti SaaS pagrindais rekomenduojame tą daryti, tačiau redis turi veikti persistant rėžimu. Kitu atveju redis reikia susidiegti.
Duomenų šaltiniai	Tai jau egzistuojančios duomenų tvarkytojo duomenų paslaugos ar duomenų bazės.	Priklausomai nuo duomenų šaltinio, gali tekti papildomai diegti python bibliotekas.
.env	Failas nurodantis gunicorn aplinkos parametrus	Aktualu tik gunicorn (SQL šaltinių atvejais)

Diegiant sistemą tiek tiesiogiai į OS, tiek per Docker institucijų sistemų administratoriai turi priimti sprendimus, o dokumente toliau pateikiamos diegimo instrukcijos yra rekomendacinio pobūdžio pavyzdys.

Naudojant konteinerius sistemų administratoriai turi pasirūpinti, kad duomenys ir konfigūraciniai failai būtų saugomi failinėje sistemoje, daromi backpup.

SPINTA Tiesioginis diegimas į OS

Agentas veikia ir yra testuotas Linux operacinėse sistemose, konkrečiai naudojant Debian/Ubuntu distribucijas, todėl instrukcijos, kaip pavyzdys bus pateiktos būtent Debian/Ubuntu aplinkai. Diegimą galima atlikti ir kitose Linux distribucijose, tačiau tam tikros vietos nurodytos šioje dokumentacijoje turėtu būti priderintos taip, kad veiktų kitoje distribucijoje.

> Pateikiama instrukcija yra kaip pavyzdys, kai naudojama Debian/Ubuntu OS

Spinta yra sukurta naudojant Python programavimo kalbą, veikia su Python versijomis 3.10-3.13. Naujose Agento versijose reikalavimas Python versijai gali keistis.

Dėl serverio resursų, tokių kaip CPU, RAM ir HDD, reikalingi resursai tiesiogiai priklauso nuo publikuojamų duomenų kiekio ir naudotojų srauto, kurie naudosis duomenų publikavimo paslauga.

Minimalūs reikalavimai Agentui, be duomenų ir su 5 naudotojais vienu metu besinaudojančiais duomenų teikimo paslauga būtų 1 CPU, 1 GB RAM ir 5 GB HDD laisvos vietos, kuri lieka pilnai įdiegus operacinę sistemą ir visas reikalingas priklausomybes.

> Taip pat rekomenduojama bent 10 GB HDD laisvos vietos, kuri lieka pilnai įdiegus operacinę sistemą ir visas reikalingas priklausomybes. Ši vieta gali būti reikalinga log'ų įrašams.

Pats savaime Agentas su visomis Python priklausomybėmis diske užima apie 2 GB vietos, tačiau sunaudojamos vietos skaičius gali skirtis, skirtingose distribucijose.

Agento veikimas turėtu būti nuolat stebimas ir reikiami resursai didinami, pagal poreikį.

Operacinės sistemos paruošimas

> Agentas turėtu būti diegiamas ir leidžiamas spinta naudotojo teisėmis (ar kito **ne root** naudotojo teisėmis), todėl reikia sukurti sisteminį naudotoją:

```
sudo useradd --system -g www-data --create-home --home-dir /opt/spinta s
```

Atkreipkite dėmesį, kad visose komandose, kurios prasideda sudo, komanda turi būti vykdoma administratoriaus teisėmis, tačiau visur kur nėra sudo, komanda turi būti vykdoma spinta naudotojo teisėmis. Tai yra svarbu, todėl nesupainiokite kokio naudotojo teisėmis vykdote komandas, priešingu atveju susidursite su sunkumais susijusiais su failų teisėmis.

Python diegimas

Daugelis Linux distribucijų ateina su įdiegta Python versija, tačiau reikia įsitikinti, ar distribucijos Python versija yra tinkama:

```
python3 --version
```

Jei Python versija yra tarp 3.10 ir 3.13, tada galite pereiti prie sekančio žingsnio.

Jei versija yra žemesnė nei 3.10, ar aukštesnė nei 3.13, tuomet reikės įsidiegti palaikomą Python versiją.

Paprasčiausias būdas tai padaryti yra naudojantis distribucijos teikiamais paketais, Ubuntu atveju galite daryti taip:

```
sudo apt update
sudo apt upgrade
sudo apt install -y \
git make build-essential libssl-dev zlib1g-dev \
libbz2-dev libreadline-dev libsqlite3-dev wget \
curl llvm libncurses5-dev libncursesw5-dev \
xz-utils tk-dev libffi-dev liblzma-dev
sudo git clone https://github.com/pyenv/pyenv.git /opt/pyenv/
export PYENV_ROOT=/opt/pyenv
/opt/pyenv/bin/pyenv install --list | grep -v - | tail
sudo PYENV_ROOT=/opt/pyenv /opt/pyenv/bin/pyenv install 3.13.9
```

Naujausia python versija bus pasiekiama `python3.13` komandos pagalba.

Jei sisteminė Python versija jums tinkama, gali būti, kad joje nėra Python virtualių aplinkų kūrimo įrankio. Jį galite sudiegti taip:

```
sudo apt update
sudo apt upgrade
sudo apt install python3-venv
```

Spinta diegimas

Atkreipkite dėmesį, kad visos komandos diegiant Spinta turi būti vykdoma spinta naudotojo teisėmis ir iš spinta naudotojo namų katalogo /opt/spinta.

Aktyvų naudotoją ir katalogą galite pasikeisti taip:

```
sudo -Hsu spinta
```

```
cd
```

Agentas veikia *Spinta_* priemonės pagalba, kuriai reikia Python. Rekomenduojama visus Python paketus diegti taip vadinamoje izoliuotoje Python aplinkoje, kurią galima susikurti taip:

```
python3 -m venv env
```

Jei diegėte naujesnę python versiją - nepamiškite nurodyti jūsų naudojamos Python versijos numerį, kuris gali skirtis:

```
/opt/pyenv/versions/3.13.9/bin/python -m venv env
```

Toliau diekite spinta:

DVMS partneriams projekto vystymo metu reikia diegti naujausią pre-release versiją, kadangi joje yra naujausi projekto vystymui ir partnerių darbui atlikti pakeitimai. Spintą diegti rekomenduojama su VSSA patvirtintais paketais:

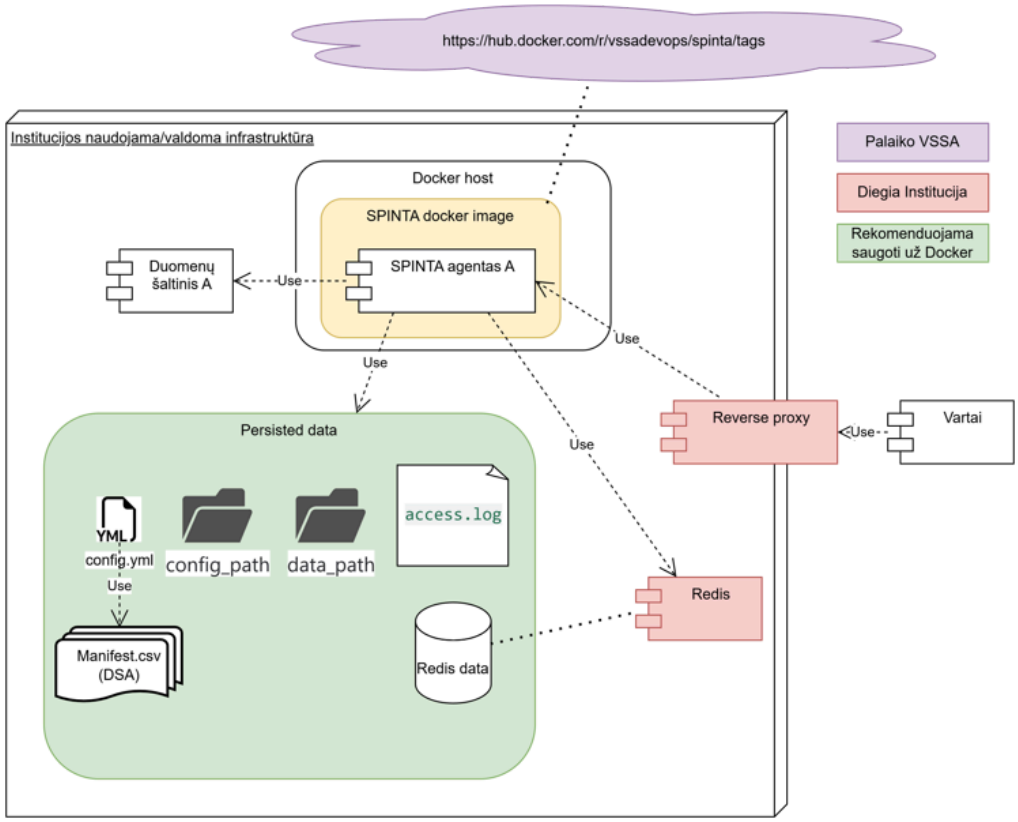
```
env/bin/pip install --require-hashes -r https://raw.githubusercontent.co
```

Sudiegę, galit patikrinti sudiegtą spintos versiją:

```
env/bin/spinta --version
```

SPINTA diegimas naudojant konteinerius

Norint naudoti SPINTA agentą kaip konteinerį galima konteinerį kurtis patiems arba naudoti oficialų VSSA palaikomą docker image. Tačiau verta atkreipti dėmesį, kad naudojant VSSA image apima tik SPINTA dalį, tačiau Redis ir Reverse proxy (SSL užtikrinimui) diegimas ir konfigūravimas lieka institucijos atsakomybėje.



VSSA palaikomą oficialią Docker image galima parsisiųsti iš Docker hub: https://hub.docker.com/r/vssadevops/spinta/tags

Sudiegus image - SPINTA agento konfigūravimas atliekamas iš dalies taip pat kaip ir diegiant SPINTA į virtualią ar fizinę mašiną: SPINTA agento konfigūravimas

Taigi siūlome susieti docker kintamąjį `SPINTA_CONFIG` nurodant kaip vadinsis ir kur bus pasiekiamas pagrindinis Spinta agento konfigūracijos failas.

Jo default reikšmė turėtų būti `SPINTA_CONFIG=/opt/spinta/config.yml` tačiau tai priklauso nuo to kaip bus primontuota failinė sistema prie docker image.

Docker duomenų saugyklos (Volumes)

Primontuotas išteklius	Tipas	Paskirtis
config_path	Katalogas	
data_path	Katalogas	
access.log	Failas ar jo saugojimo katalogas	
manifest.csv(DSA)	Katalogas kuriame saugomi DSA	Nebūtina atskirai, jei bus saugomas config_data ar data_path kataloguose, o config.yml nukreips į juos.

Redis

⚠ Redis komponentas yra kritinis komponentas. Jis naudojamas GUID ir vidinių pirminių šaltinio raktų susiejimui. Kadangi SPINTA agentas GUID sukuria naudodamas atsitiktinės generacijos algoritmus, praradus Redis duomenis raktus reiktų generuoti naujai, tačiau duomenų teikėjai būtų išsaugoję senus raktus ir duomenų integralumas nebūtų užtikrinamas.

ℹ Institucijos IT administratoriai turi užtikrinti, kad Redis duomenys būtų apsaugoti nuo praradimo ar sugadinimo.

Redis komponentą galima naudoti tiek SaaS paslaugą, tiek diegti savo infrastruktūroje.

Kaip pasiekti Redis nurodoma config.yml ir plačiau aprašyta skyriuje SPINTA agento konfigūravimas

SSL ir reverse proxy

Spinta agentas veikia HTTP protokolų, tad institucijos IT administratoriai turi pasirūpinti SSL. Tam gali naudoti jiems tinkamus ar naudojamus sprendimus ir komponentus.

Šaltinių konfigūravimas

Tam, kad Agentas galėtų pasiekti duomenis ir teikti juos UDTS formatu, reikia nurodyti, kokie duomenys bus teikiami, ir kaip juos pasiekti. Tai daroma naudojantis duomenų struktūros aprašais (DSA). Šiuo atveju, mums reikės Šaltinio duomenų struktūros aprašų (ŠDSA).

Kaip nurodyta konfigūracijos faile, turite pateikti duomenų struktūros aprašą, kurio pagrindu veiks agentas.

SPINTA palaikomi šaltiniai:

- SQL
- WSDL/SOAP
- XML
- JSON

WSDL ir SOAP šaltiniai

WSDL ir SOAP šaltinio struktūros parengimas aprašytas čia:

Duomenų šaltiniai — DSA

Struktūros aprašo, skirto WSDL duomenims gauti, sudarymo pavyzdys:

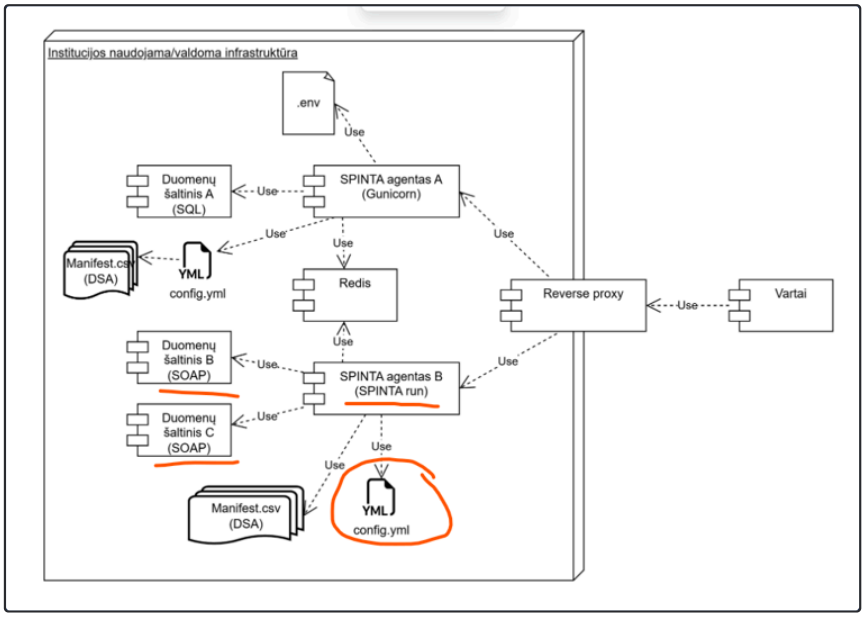
```
cat > manifest.csv <<'EOF'
id,dataset,resource,base,model,property,type,ref,source,prepare,level,s
,datasets/gov/vssa/demo/rctest,,,,,dataset,,,,,,,,,,
,,rc_wsdl,,,,wsdl,,https://test-data.data.gov.lt/api/v1/rc/get-data/?ws
,,get_data,,,,soap,,Get.GetPort.GetPort.GetData,wsdl(rc_wsdl),,,,,,,,
,,,,,,param,action_type,input/ActionType,input(),,,,,,,,
,,,,,,param,caller_code,input/CallerCode,input(),,,,,,,,
,,,,,,param,end_user_info,input/EndUserInfo,input(),,,,,,,,
,,,,,,param,parameters,input/Parameters,input(),,,,,,,,
,,,,,,param,time,input/Time,input(),,,,,,,,
,,,,,,param,signature,input/Signature,"creds(""sub"").input()",,,,,,,,
,,,,,,param,caller_signature,input/CallerSignature,input(),,,,,,,,
,,,,GetData,,,,/,,,,,,,,,
,,,,,response_code,string,,ResponseCode,,,,,,,,
,,,,,response_data,string,,ResponseData,base64(),,,,,,,,
 ,,,,,decoded_parameters,string,,DecodedParameters,,,,,,,,
 ,,,,,action_type,string,,,param(action_type),,,,,,,,
,,,,,end_user_info,string,,,param(end_user_info),,,,,,,,
,,,,,caller_code,string,,,param(caller_code),,,,,,,,
,,,,,parameters,string,,,param(parameters),,,,,,,,
,,,,,time,string,,,param(time),,,,,,,,
,,,,,signature,string,,,param(signature),,,,,,,,
,,,,,caller_signature,string,,,param(caller_signature),,,,,,,,
,,,,,,,,,,,,,,,,,
,,nested_read,,,,dask/xml,,,eval(param(nested_xml)),,,,,,,,
 ,,,,,,param,nested_xml,GetData,read().response_data,,,,,,,,
,,,,Country,,,,countries/countryData,,,,,,,,
,,,,,id,string,,id,,,,,,,,
,,,,,title,string,,title,,,,,,,,
EOF
```

SPINTA agento konfigūravimas

Spinta agento konfigūravimas susideda iš kelių žingsnių

1 SPINTA Agento konfigūracijos failų paruošimas
2 Keymap DB
3 Spinta agento B web serverio serviso konfigūravimas
4 SPINTA agento serviso aktyvavimas
5 Web serverio (nginx) konfigūravimas
6 SSL konfigūracija

SPINTA Agento konfigūracijos failų paruošimas



Pavyzdyje pateikiamas SOAP šaltinio konfigūravimas

SPINTA agentą galima konfigūruoti ne vienu būdu ir žemiau pateiktos instrukcijos yra bazinės, kai agentas konfigūruojamas darbui su SOAP/WSDL šaltiniu (*paveikslėlyje atitinka agentą B*).

Spinta yra konfigūruojama konfigūracijos failo pagalba, kurio, pagal nutylėjimą ieškoma aktyviame kataloge. Kur Spinta ieško konfigūracijos failo, galima patikrinti taip:

```
env/bin/spinta config config
```

Kuriuos, taip pat galima pateikti ir .env faile:

```
cat > .env <<'EOF'
```

```
SPINTA_CONFIG=/opt/spinta/config.yml
EOF
```

Kadangi failas neegzistuoja, reikia jį sukurti:

```
cat > config.yml <<'EOF'
config_path: /opt/spinta/config
env: production

keymaps:
  default:
    type: redis
    dsn: redis://localhost:6379/1

accesslog:
  type: file
  file: /opt/spinta/logs/access.log
EOF
```

Failo aprašas

Kintamasis	Privalomas ar neprivalomas	Aprašas
config_path	Privalomas	Nurodo kelią į direktoriją, kurioje yra įvairūs konfigūracijos duomenys
env		Nurodo, ar aplinka testinė, ar vystymo, ar produkcinė. Rekomenduojama diegti su `production`, nes `dev` arba `test` įrašo papildomų konfigūracijų.
keymaps	privalomas	Aprašyta žemiau
.default		Nurodo numatytąjį
..type		Tipas - `redis`
..dsn		Nurodo, kaip pasiekti servisą
accesslog		Nurodo, kur bus saugomi prieigos žurnalai
.type		Tipas - ar saugoma faile, ar kitaip
.file		Jei saugoma faile, nurodo, kuriame faile

Prieš testuojant ar konfigūracija veikia, sukuriame reikalingus katalogus:

```
#config ir logs katalogai šiuo atveju kuriami atitinkamai pagal config.y
mkdir /opt/spinta/config
mkdir /opt/spinta/logs
#optional jei naudojama sqlite
mkdir /opt/spinta/var
```

Konfigūracijos tikrinimas

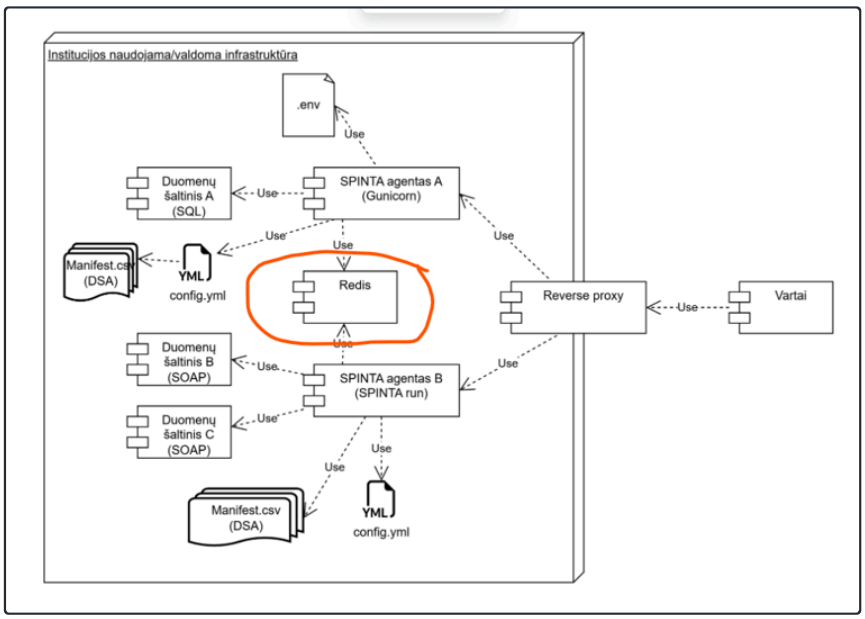
```
env/bin/spinta config config backends manifests accesslog
```

Jei jūsų duomenų šaltinis - reliacinė duomenų bazė, patikriname ar Spinta gali prisijungti prie duomenų bazės.

```
env/bin/spinta wait 1
```

Keymap DB

Keymap naudojamas susieti išorinius identifikatorius su vidiniais identifikatoriais. Gali būti konfigūruojama.



Pagal numatytuosius nustatymus naudojama SQLite, tačiau **būtina pakeisti** į Redis **persistent**

SPINTA agentui nurodome kokį keymap komponentą naudoti ir kaip jį pasiektį per config.yml failą. Pateikiame config.yml failo fragmentą:

```
keymaps:
  default:
    type: redis
    dsn: redis://redis-address:6379/1
```

Jei neturite **Redis** serviso, rekomenduojame diegti Docker konteineryje.

Pavyzdinė Redis diegimo instrukcija

Instaliuojam Docker ir pridedam `spinta` naudotoją į `docker` grupę:

```
exit
sudo apt install docker.io docker-compose-v2
sudo usermod -aG docker spinta
sudo -Hsu spinta
```

```
cd
```

Sukuriam Docker Compose failą:

```
cat > docker-compose.yml << "EOF"
services:
  redis-keymap: # sqlite faster alternative
    image: valkey/valkey:9
    restart: always
    command: [ "redis-server", "--appendonly", "yes", "--appendfsync",
    volumes:
      - redis_keymap_data:/data
    ports:
      - "6379:6379"
volumes:
  redis_keymap_data:
EOF
```

SVARBU! Redis turi būti būtinai leidžiamas persistent režimu (appendonly:yes ir appendfsync:always parametrai) Yra keli persistent režimai (žr. Redis/Valkey dokumentaciją). Numatytasis režimas (appendonly:yes ir appendfsync:always) užtikrina didžiausią duomenų nepraradimo patikimumą, tačiau turi mažiausią greitį naujo rakto kūrimo metu, lyginant su kitais režimais.

Paleidžiam ir patikrinam:

```
docker compose up -d
docker ps
```

Pakeičiame konfigūracijos failo vietą aplinkos kintamojo pagalba:

```
export SPINTA_CONFIG=/opt/spinta/config.yml
```

Patikriname ar konfigūracijoje ir pateiktame struktūros apraše nėra klaidų.

```
env/bin/spinta check
```

```
exit
```

Keymap (Redis) yra kritinis elementas, tad privalote užtikrinti, kad duomenys nebūtų prarasti as sugadinti, nes tokiu atveju bus pažeistas perduodamų duomenų integralumas.

Spinta agento B web serverio serviso konfigūravimas

Jei nėra, reikia sudiegti UV tools biblioteką.

```
sudo -Hsu spinta
cd
env/bin/pip install uvicorn uvloop httptools
```

Su `root` teisėmis sukuriame *SystemD* servisą (atkreipkite dėmesį, kad jūsų pasirinkta distribucija gali naudoti kitą servisų valdymo priemonę, tuomet šis pavyzdys netiks):

```
cat > /etc/systemd/system/spinta.service << "EOF"
[Unit]
Description=Spinta external WSDL service
After=network.target

[Service]
Type=simple
User=spinta
Group=www-data
WorkingDirectory=/opt/spinta
Environment="SPINTA_CONFIG=/opt/spinta/config.yml"

ExecStart=/opt/spinta/env/bin/spinta \
    -o config=/opt/spinta/config.yml \
    run /opt/spinta/manifest.csv \
    --mode external \
    --host 127.0.0.1 \
    --port 8000

Restart=on-failure
RestartSec=5

[Install]
WantedBy=multi-user.target
EOF
```

SPINTA agento serviso aktyvavimas

Aktyvuokite servisą:

```
sudo systemctl enable spinta
sudo systemctl daemon-reload
sudo systemctl start spinta
sudo systemctl status spinta
```

Web serverio (nginx) konfigūravimas

⚠ SPINTA agentas veikia tik SSL ryšiu, tačiau jo užtikrinimu turi rūpintis institucijos sistemų administratoriai. Žemiau pateikiame tik bendrojo pobūdžio aprašą.

Įdiegiame pasirinktą Web serverio paketą, šiuo atveju pavyzdys pateiktas *Nginx*:

```
sudo apt update
sudo apt install nginx
```

Sukuriame pasirinkto Web serverio, šiuo atveju Nginx, konfigūracijos failą (pakeiskite `example.com` į jūsų domeno pavadinimą):

```
cat > /etc/nginx/sites-available/example.com << "EOF"
server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
```

```
}
EOF
```

Aktyvuojame konfigūracijos failą:

```
sudo ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-enabl
```

Patikriname ar konfigūracija veikia:

```
sudo nginx -t
```

Perkraukite Nginx:

```
sudo systemctl restart nginx
```

Patikrinkite ar servisas veikia:

```
sudo systemctl status nginx
```

SSL konfigūracija

Detalesnės instrukcijos apie tai, kaip konfigūruoti SSL sertifikatus ir kitus Gunicorn ar Nginx parametrus rasite minėtų projektų dokumentacijoje.

Jei naudojate Let’s Encrypt sertifikatus, jų diegimą galima daryti `certbot` pagalba:

```
sudo snap install --classic certbot
```

```
sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

```
sudo certbot --nginx
```

Duomenų gavimo testavimas

Norint ištestuoti duomenų gavimą, reikia sukonfigūruoti klientus ir jų leidimus.

Daugiau infromacijos apie leidimus galit rasti UDTS specifikacijoj: UAPI

Klientus ir jų leidimus galima pridėti tokia komanda:

```
sudo -Hsu spinta
cd
export SPINTA_CONFIG=/opt/spinta/config.yml
```

```
env/bin/spinta client add -n test --scope - <<EOF
uapi:/:getone
uapi:/:getall
uapi:/:search
uapi:/:changes
EOF
```

Šios komandos atsakymas bus kažkas panašaus į:

```
New client created and saved to:

    /opt/spinta/config/clients/id/f4/0e/76b7-f3f4-4a4c-b2e9-c03a147d65f

Client secret:

    wjhl5sKB0YkE994yXue8rX0E-dQadKcF

Remember this client secret, because only a secure hash of
client secret will be stored in the config file.
```

Būtinai išsisaugokit šį Secret, nes daugiau jo pamatyti nebegalėsit.

Jei norim susieti šį klientą su šaltinio klientu, turim pakoreguoti šį sukurtą kliento failą, ir pridėti `backends` dalį, kurioje aprašysim jungimuisi prie šaltinio reikalingus duomenis:

```
backends:
  backends:
  get_data:
    sub: MTAwMQ==
```

Pilnas kliento failas turėtų atrodyti panašiai į šį:

```
client_id: ae571e79-a826-4e10-b792-cc1a4d1e94ca
client_name: test
client_secret_hash:
```

```
  pbkdf2$sha256$952803$anM6ka5E5CuKoDMaj90f4Q$NNymDRKxL_JOwHL8z9GA6uume
scopes:
  - uapi:/:getone
  - uapi:/:getall
  - uapi:/:search
  - uapi:/:changes
backends:
  get_data:
    sub: MTAwMQ==
```

Pridėję klientą, pirmiausiai turite išsiimti autorizacijos žetoną.

Žemiau nurodytas pavyzdys, kaip galima ištestuoti duomenų gavimą komandinės eilutės pagalba, bet taip pat galit naudoti ir REST API įrankius, kaip Postman, Insomnia ar panašius.

Kompiuteryje, kuriame norime gauti duomenis, pirmiausiai nurodom kliento duomenis, įskaitant norimus leidimus (jei testuojame iš kito kompiuterio, nei sudiegėme agentą, vietoje `localhost` nurodome agento URL arba IP adresą):

```
SERVER=http://localhost:8000
CLIENT=test
SECRET=secret
SCOPES="
  uapi:/:getone
  uapi:/:getall
  uapi:/:search
  uapi:/:changes
"
```

Tada reikia paleisti komandą, kurios pagalba gautome autorizacijos žetoną:

```
TOKEN=$(
  curl -sS -f \
    -u "$CLIENT:$SECRET" \
    -d "grant_type=client_credentials" \
    -d "scope=$SCOPES" \
    "$SERVER/auth/token" \
  | jq -r .access_token
)
```

Pasitikrinam, ar gavom žetoną:

```
AUTH="Authorization: Bearer $TOKEN"
echo "$AUTH"
```

Turėtume gauti kažką panašaus į:

```
{
"token_type":"Bearer",
"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzUxmiIsImtpZCI6IlVPb1M0dXU5
"expires_in":864000,
"scope":"uapi:/:search uapi:/:changes uapi:/:getall uapi:/:getone"}
```

Darom užklausą į serverį:

```
curl --location 'https://example.com/datasets/gov/vssa/demo/rctest/Count
  --header 'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzUxMiIsIm
```

Papildoma informacija

Spintos naujinimas

Norint pasitikrinti, ar Spinta versija yra naujausia, galima paleisti šias dvi eilutes:

```
spinta –version

pip index versions --pre spinta
```

Jei versijos sutampa - naujinti nereikia.

Jei versijos skiriasi - reikėtų atnaujinti Spintą.

Visus naujausius ir kitoje versijoje numatomus pakeitimus galima rasti čia: CHANGES.rst

Norint atnaujinti Spinta versiją, reikia įvykdyti tokias komandas:

```
sudo -Hsu spinta
cd
env/bin/pip install --require-hashes -r https://raw.githubusercontent.co
```

Struktūros aprašo naujinimas

Pasikeitus struktūros aprašui

```
sudo cp manifest.csv /opt/spinta/manifest-new.csv

sudo -Hsu spinta
cd

env/bin/spinta check manifest-new.csv

cp manifest.csv manifest-old.csv
mv manifest-new.csv manifest.csv

diff -y --suppress-common-lines manifest-old.csv manifest.csv

exit

sudo systemctl restart gunicorn
sudo systemctl status gunicorn
```

Kelių struktūros aprašų naudojimas

Jei naudojate daugiau, nei vieną manifestą, ir norite, kad Agentas pasiektų juos visus, galima tai padaryti operatoriaus sync pagalba (konfigūracijos faile) (neaktualu su SOAP/WSDL):

```
manifests:
  default:
    type: tabular
    path: /opt/spinta/manifest.csv
    backend: default
    mode: external
    sync: additional
  additional:
    type: tabular
    path: /opt/spinta/manifest2.csv
    backend: default
    mode: external
```

Prieš testuojant ar konfigūracija veikia, sukuriame reikalingus katalogus:

Problemos ir sprendimai

Jei kažkas neveikia, pirmiausiai reikėtų žiūrėti servisų žurnalus, pavyzdžiui:

```
journalctl -u gunicorn -xe
journalctl -u nginx -xe
```

Žurnaluose dažniausiai būna pateikta informacija, leidžianti suprasti kas ir kodėl neveikia.

Naujausią instrukciją galit rasti čia:

Agento diegimas — Duomenų atvėrimo vadovas documentation