

RENKU - 連句

Reproduce, Reuse, Recycle Research

Rok Roškar and the SDSC Renku team

Goals of Renku

1.

Provide the means to create **reproducible** data science

2.

Facilitate the **sharing** and **reuse** of research artefacts

3.

Foster a **collaborative environment** for interactive prototyping

4.

Enable the **discovery** of relevant data and methods

5.

Allow **federated access** across institutions giving each the freedom to impose its own access controls over resources

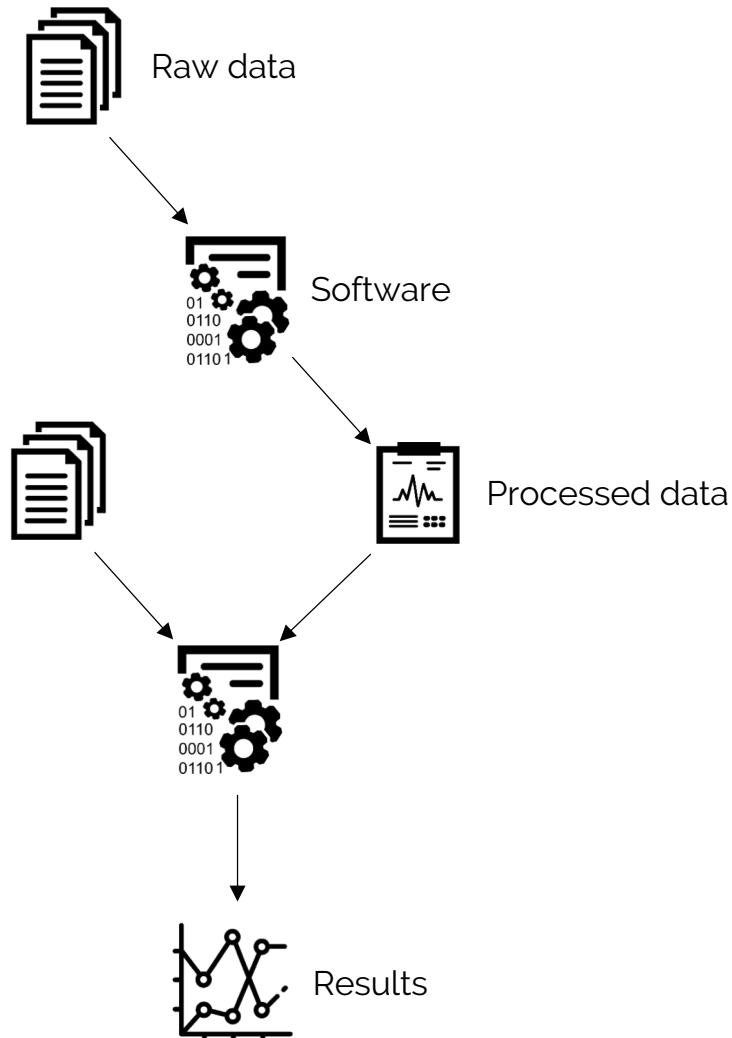
The image features a landscape background with rolling hills and a valley. A large, semi-transparent blue rectangle is overlaid on the center of the image. At the bottom of the image, there is a solid green horizontal bar. The text is centered within the blue rectangle.

Capturing, recording and utilizing the
lineage of results is the core of Renku

Terminology

- We borrow the **Renku** name from the Japanese word for *linked-verse poetry*
- A “**ku**” is a verse in a renku poem
- We use “**ku**” to mean a piece of the data analysis process – includes discussion, code, and results

Capture the scientific process

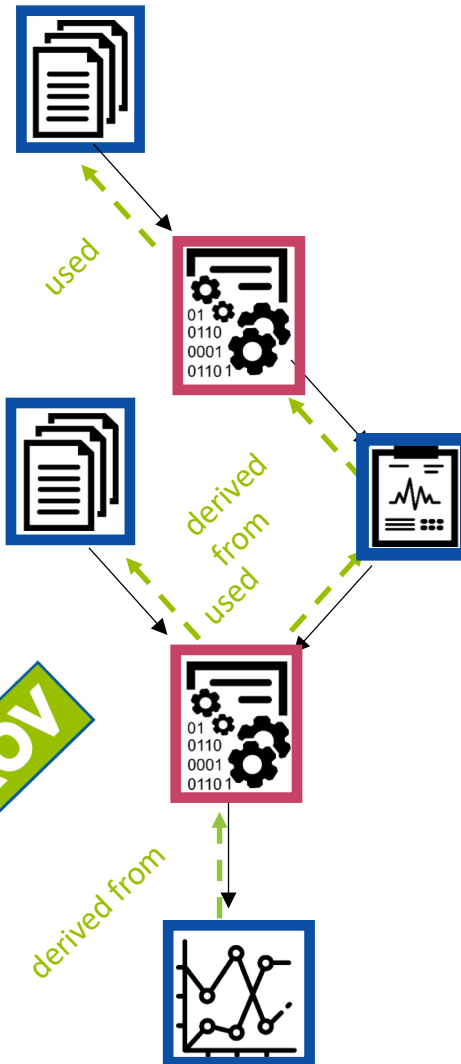


1. Lineage is recorded into a knowledge graph
2. Steps can be repeated and reused
3. Version control is built-in for data, code, and workflows
4. Lineage accessible via simple tools

Metadata

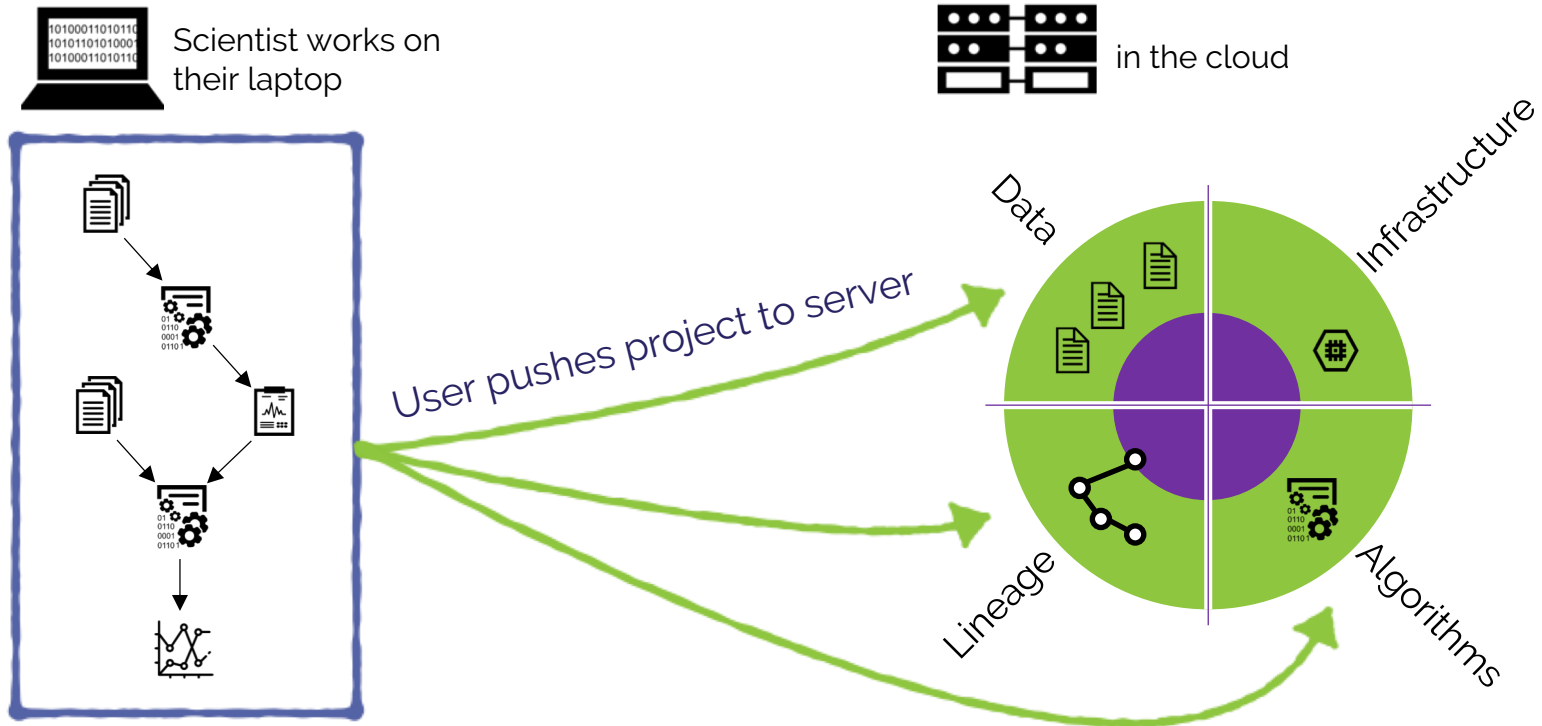


- Metadata use Dublin Core, FOAF, and Schema.org
- Provenance graph is based on PROV-O W3C recommendation



- CWL for representing all computational steps
- Capture individual steps from user input
- Tools for constructing workflows from basic pieces
- Rely on container technologies to ensure reproducibility

Verify and share results



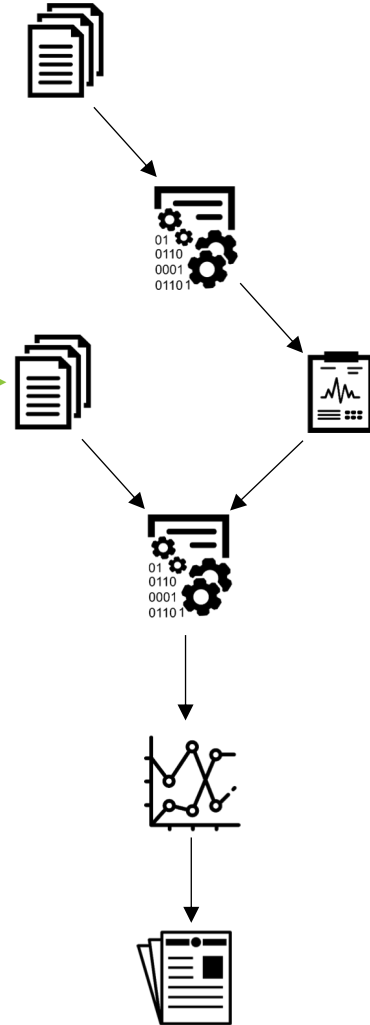
1. Results automatically verified
2. Data lineage captured "live"
3. Knowledge graph populated
4. Shareable interactive environment created

Discover and understand the work of others

Graph-based search...



Ex: search for **data** and explore the tools to efficiently generate results

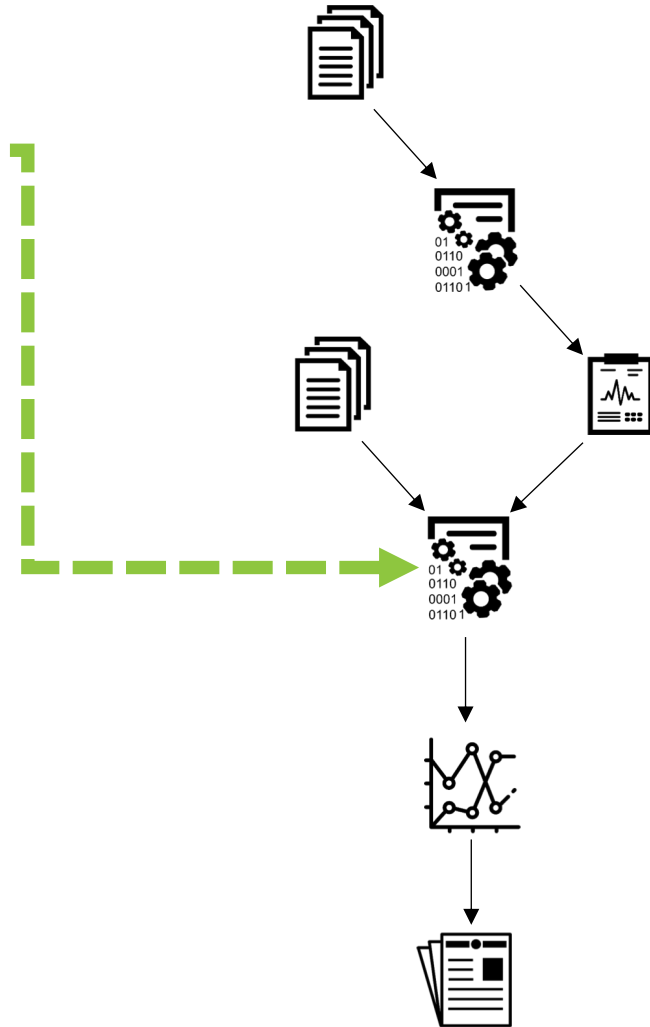


Discover and understand the work of others

Graph-based search...



Ex: search for an **algorithm** and see its applications

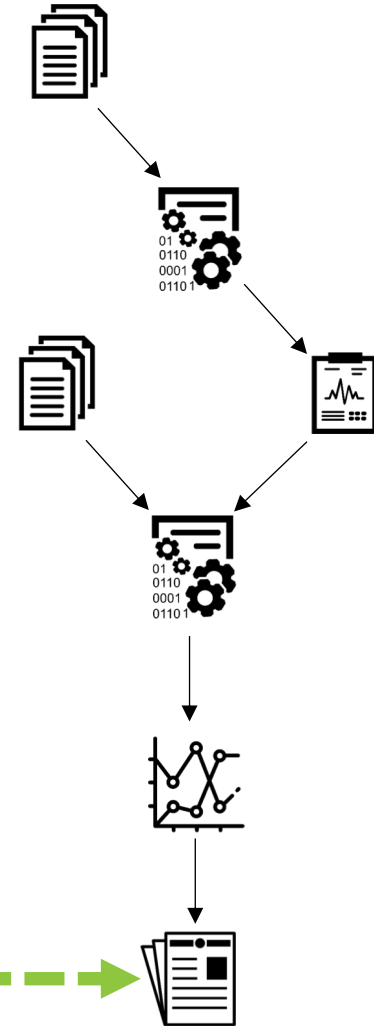


Discover and understand the work of others

Graph-based search...



Ex: search for a **publication**, obtain a full view of how the results were obtained

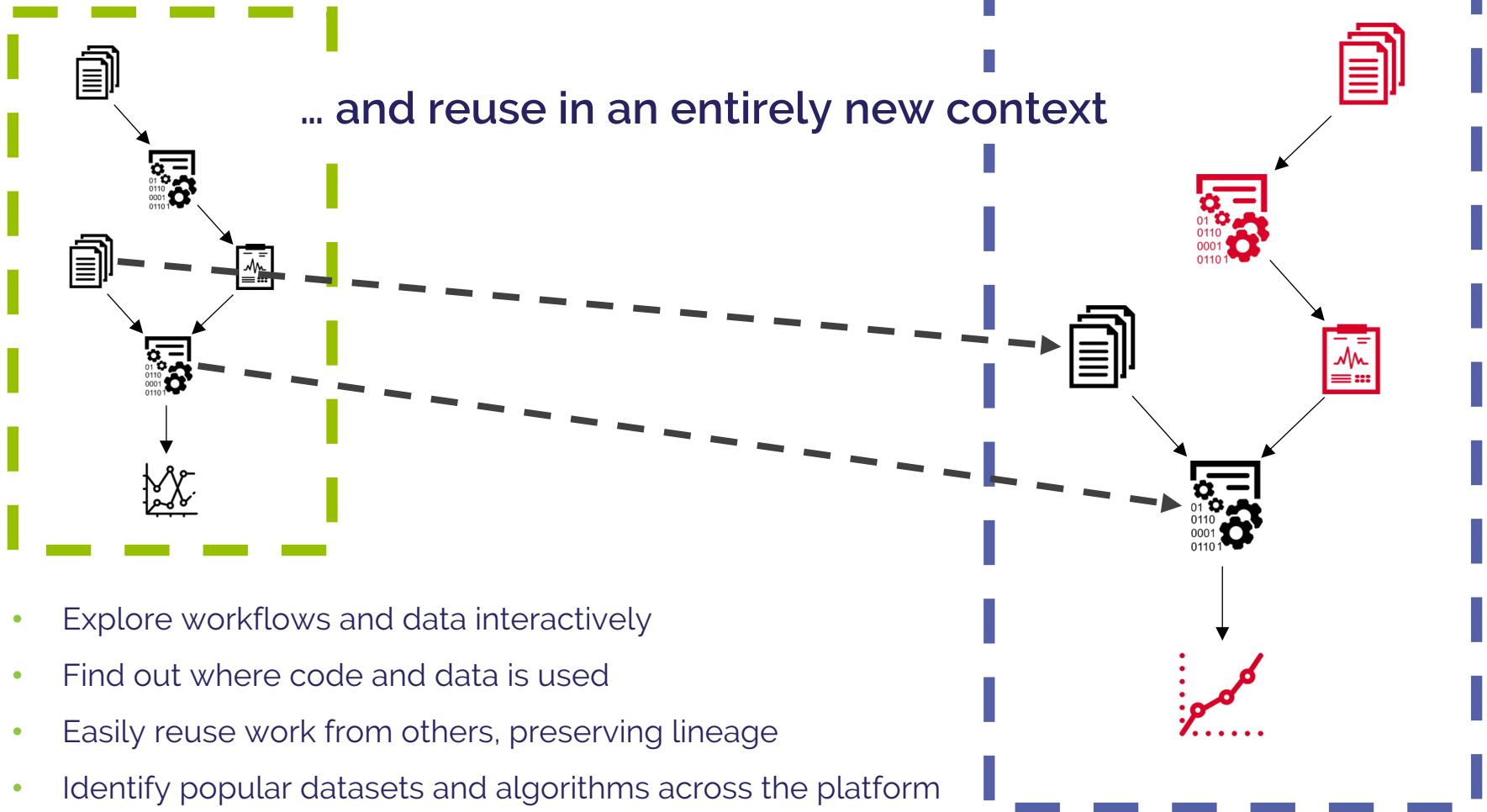


Reuse and repeat

Search for relevant data or algorithms...



... and reuse in an entirely new context



- Explore workflows and data interactively
- Find out where code and data is used
- Easily reuse work from others, preserving lineage
- Identify popular datasets and algorithms across the platform

Typical Renku Flow

1. Initialize a project in the terminal with CLI
2. Import data
 1. URL/git/local filesystem
3. Execute some work generating a lineage of results
4. Push to server
 1. Optionally verify last step (CI for the workflow/step)
 2. Build an image for the commit
5. Share link to launch hosted notebook
 1. Create changes, commit back the notebook
6. Inspect workflow
 1. Adapt inputs/parameters
 2. Rerun workflow(s) on the cloud infrastructure
 3. Create new workflows from existing pieces (could be from other projects!)

Building easy-to-use tools on top of trusted technologies

Renku consolidates the open-source data science and software engineering technologies into a single platform

```
0) Requirement already satisfied: html5lib=1.0b1,!=1.0b2,!=1.0b3,!=1.0b4,!=1.0b5,!=1.0b6,!=1.0b7,!=1.0b8,!=0.9999999pre in /opt/conda/lib/python3.6/site-packages (from bleach->nbcnvert->jupyter->weather-ch=0.1.0)
Requirement already satisfied: pyparsing=2.1.1 in /opt/conda/lib/python3.6/site-packages (from jedi=>ipython=>ipykernel->jupyter->weather-ch=0.1.0)
Requirement already satisfied: pyzmq=17.0.0 in /opt/conda/lib/python3.6/site-packages (from ipykernel->jupyter->weather-ch=0.1.0)
Requirement already satisfied: webencodings in /opt/conda/lib/python3.6/site-packages (from html5lib=1.0b1,!=1.0b2,!=1.0b3,!=1.0b4,!=1.0b5,!=1.0b6,!=1.0b7,!=1.0b8,!=0.9999999pre->bleach->nbcnvert->jupyter->weather-ch=0.1.0)
Installing collected packages: seaborn, pandas, statsmodels, weather-ch
  Running setup.py install for seaborn: started
  Running setup.py install for seaborn: finished with status 'done'
  Running setup.py develop for weather-ch
Successfully installed pandas=0.20.3 seaborn=0.8.1 statsmodels=0.8.0 weather-ch=0.1.0
Removing intermediate container 779bbd8476d8
--> 397786a99897
Step 1/7 : USER 1000
--> Running in 33ae922ee08
Removing intermediate container 33ae922ee08
--> 2eb10220ced9
Successfully built 2eb10220ced9
Successfully tagged gillab.renku.build:5081/rok/weather-ch/revise-master-phivol:25046503c71ca7b8363228eb39bfe68128b218ef
rok@master ~$ ip renku-demo ~/projects-presentation/weather-ch renku status
On branch master
all files were generated from the latest inputs.
git | master | ip renku-demo ~/projects-presentation/weather-ch | renku log | Dockerfile | README.md
gitignore | .gitignore | .ipynb_checkpoints/ | .renku/ | .renku.lock
data/ | notebooks/ | requirements.txt | src/
rok | master | ip renku-demo ~/projects-presentation/weather-ch | renku log data/zh/
homog_mo_SMA.txt | meta_data.yml | standardized.csv
rok | master | ip renku-demo ~/projects-presentation/weather-ch | renku log data/zh/standardized.csv
• 6bfcff91 data/zh/standardized.csv
• 6bfcff91 renku/workflow/974119acc835466836fc70f6ffffc-python.cwl
• 64ef36d data/zh/homog_mo_SMA.txt
rok | master | ip renku-demo ~/projects-presentation/weather-ch |
```

Command-line interface

The screenshot shows the Renku web interface for a project named 'weather-ch'. The page title is 'weather-ch' and the subtitle is 'An investigation into weather trends in Zürich, Switzerland'. The interface includes a search bar, a 'Projects' dropdown, and a 'Linear models' dropdown. The main content area is divided into two sections: 'Analyze data' and 'Preprocess data'. The 'Preprocess data' section is active and shows a notebook titled 'Preprocess data' with a 'Launch Notebook' button. The notebook content includes code for importing pandas, numpy, scipy, matplotlib, and seaborn, and a data reader function.

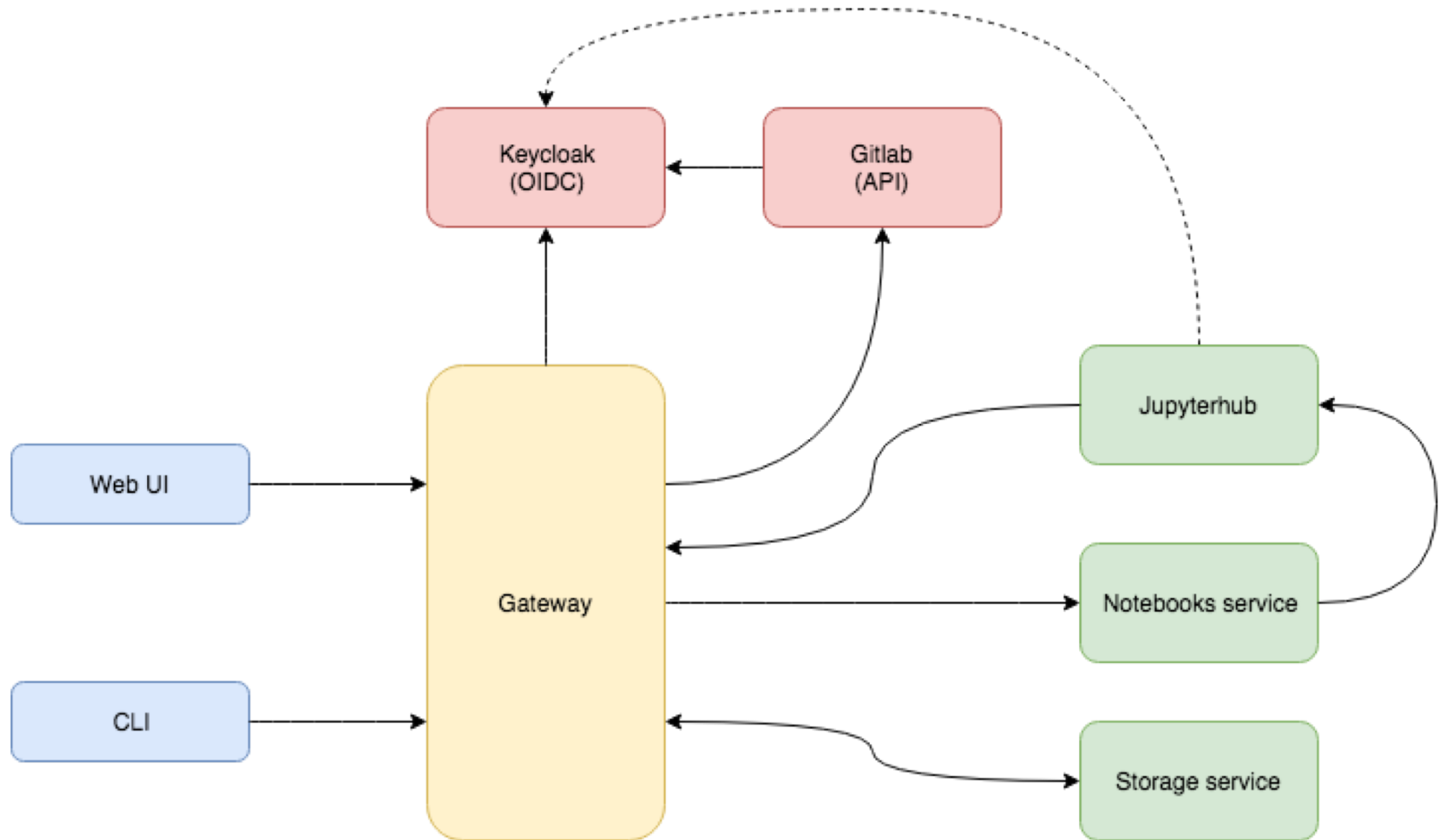
Web-based front-end

Platform components and technology

- **renku-ui**: react.js + bootstrap
- **renku-python**: click-based python CLI and API client
- **renku-notebooks**: flask app interacting with jupyterhub
- **renku-storage**: git-lfs & S3 API with scala/play
- **renku-gateway**: API Gateway (in-progress)
- **renku-graph**: Knowledge Graph (in progress)
- Workflow execution (CWL+???)
- Repository management + CI (vanilla GitLab)
- Authentication (Keycloak)
- Runtime (docker + kubernetes)



Basic Architecture



Deployment

- Deployed via kubernetes helm charts (docker-compose option for development)
- Main chart in top-level repo, but each service can be standalone
- Development charts pushed on every commit to a chart registry for continuous deployment
- Stable chart repository (upon release)
- Cloud (SWITCH OpenStack) + HPC environments (CSCS)

<https://swissdatasciencecenter.github.io/helm-charts>

Development and Roadmap

- Q1+Q2 2018: Redeveloped ground-up
 - Provide a user interface for project/discussions
 - Create interactive environments on-demand
 - Track lineage within and across projects
 - Automatic update of results based on lineage
 - Basic “cloud” workflow with interactive notebooks including custom images
- Q3 2018: solidify the base, fill in the gaps
 - Hosted platform for a limited audience
 - Collect use-cases, obtain feedback (bugfixes!)
 - Index data into an instance-wide Knowledge Graph
 - Capture component interactions on-line via events
 - Basic support for (re)executing workflows in the cloud through the UI
- Q4 2018:
 - Introduce access controls
 - Exploit the Knowledge Graph for user queries
- 2019 → federation

Current status

Platform is under very **active** development:

<https://github.com/SwissDataScienceCenter>

renku — services & deployment recipes

renku-python — CLI and Python API

renku-ui — Web front-end

renku-notebooks — external JH service

renku-storage — storage service (LFS+S3)

