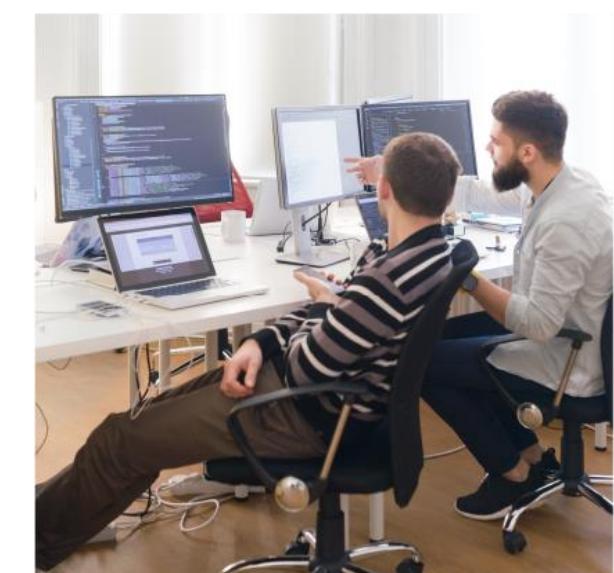
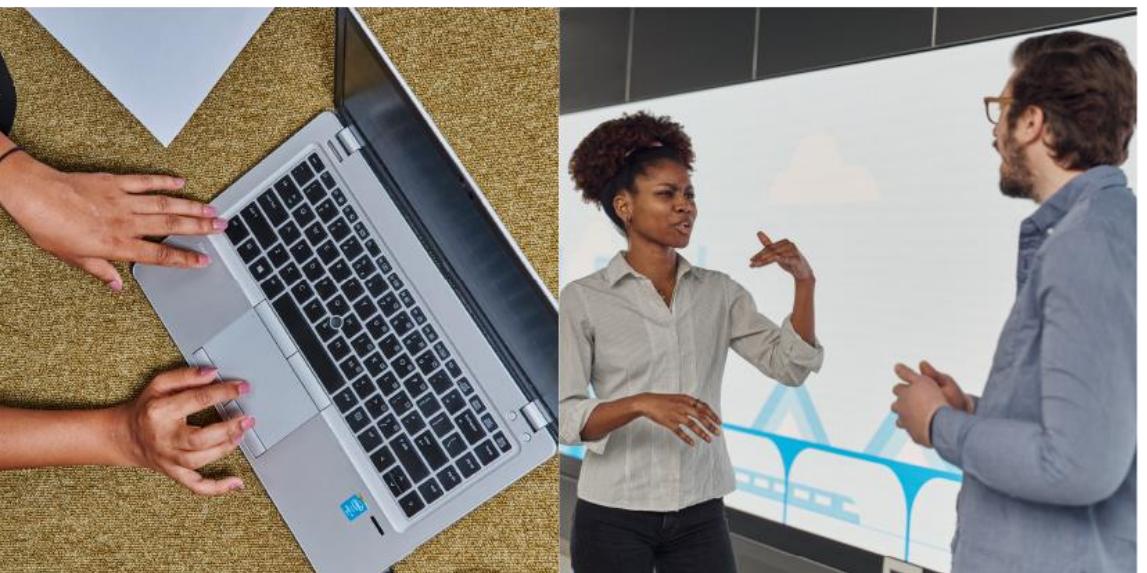
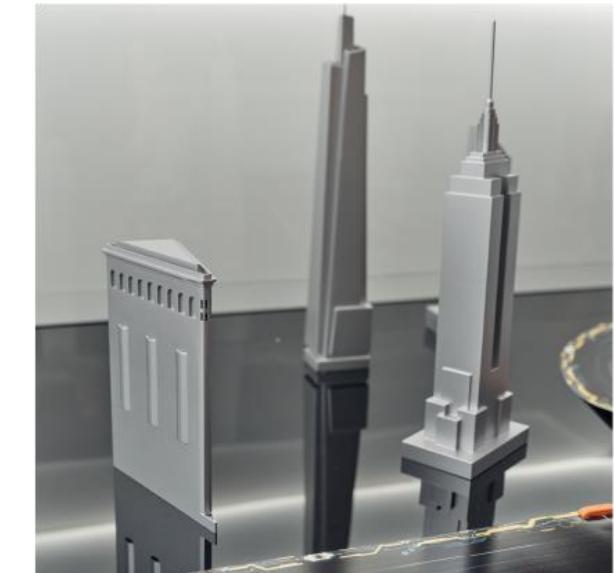


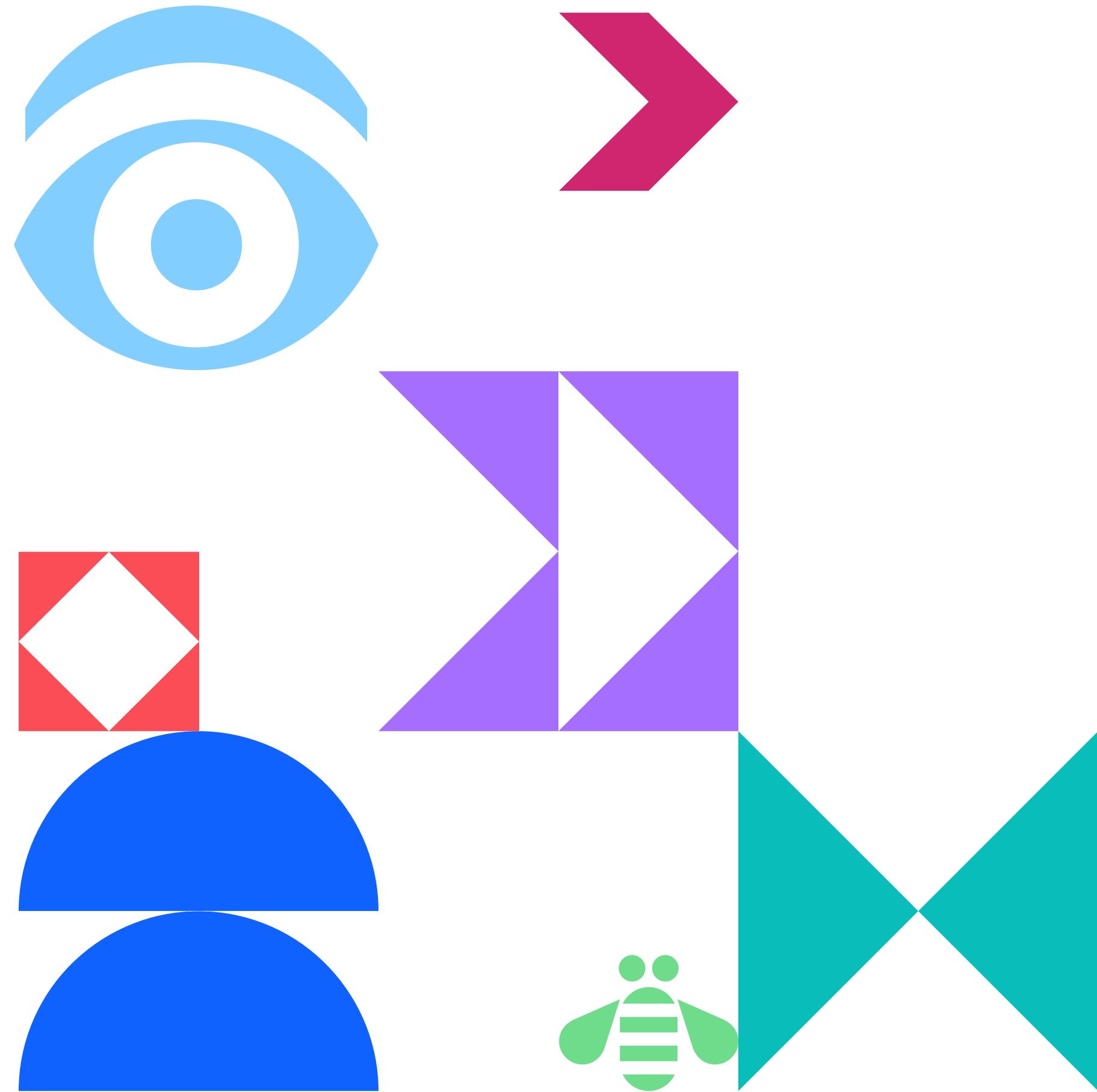
Open Source Data Science and MLOps in **watsonx.ai**

Ryan Kather – Public Market
IBM, Principal AI Engineer – IBM Data, AI, Automation
Session Code: 2494

October 23, 2024
Mandalay Bay, Las Vegas



Agenda



- 01 Simpler Times
- 02 *Ops
- 03 [watsonx.ai](#)
- 04 - Secure Coding
- 05 - Resources
- 06 - Environments

Simpler Times

Data Science is now a team sport.

Data and ML landscapes have evolved.

Factors to Account

- Repeatability
- Accountability
- Transparency
- Explainability
- Risk Exposure
- Robustness
- Performance



+BP4DS

1. Understand Business Context
2. Prioritize Data Cleaning
3. Conduct Exploratory Data Analysis
4. Validate Models Properly
5. Communicate Results Clearly

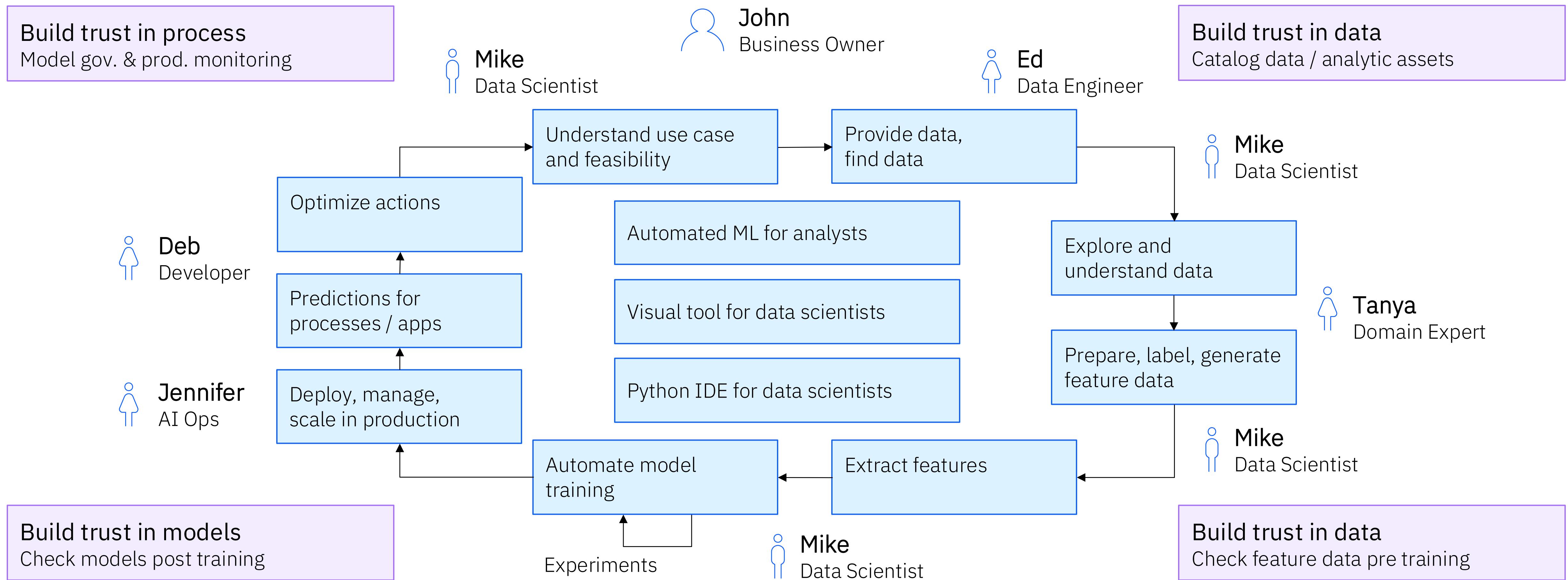
-- GPT4

1. Start with a Clear Problem Statement
2. Use Version Control
3. Use Reproducible Methods
4. Use Appropriate Visualizations
5. Continuously Learn and Improve

-- Llama2

DS Process

CRISP-DM NIST Methodology



What is MLOps?

MLOps, also known as Machine Learning Operations, is a set of practices that aim to streamline and automate the process of taking machine learning (ML) models from development to production. It involves collaboration between data scientists, engineers, and operations teams to ensure the smooth deployment, monitoring, and maintenance of ML models in a production environment.

*Ops All the Things

Relationship to DevOps

DevOps focuses on bridging the gap between software development and operations teams, while MLOps does the same for machine learning development and operations teams. Both practices aim to:

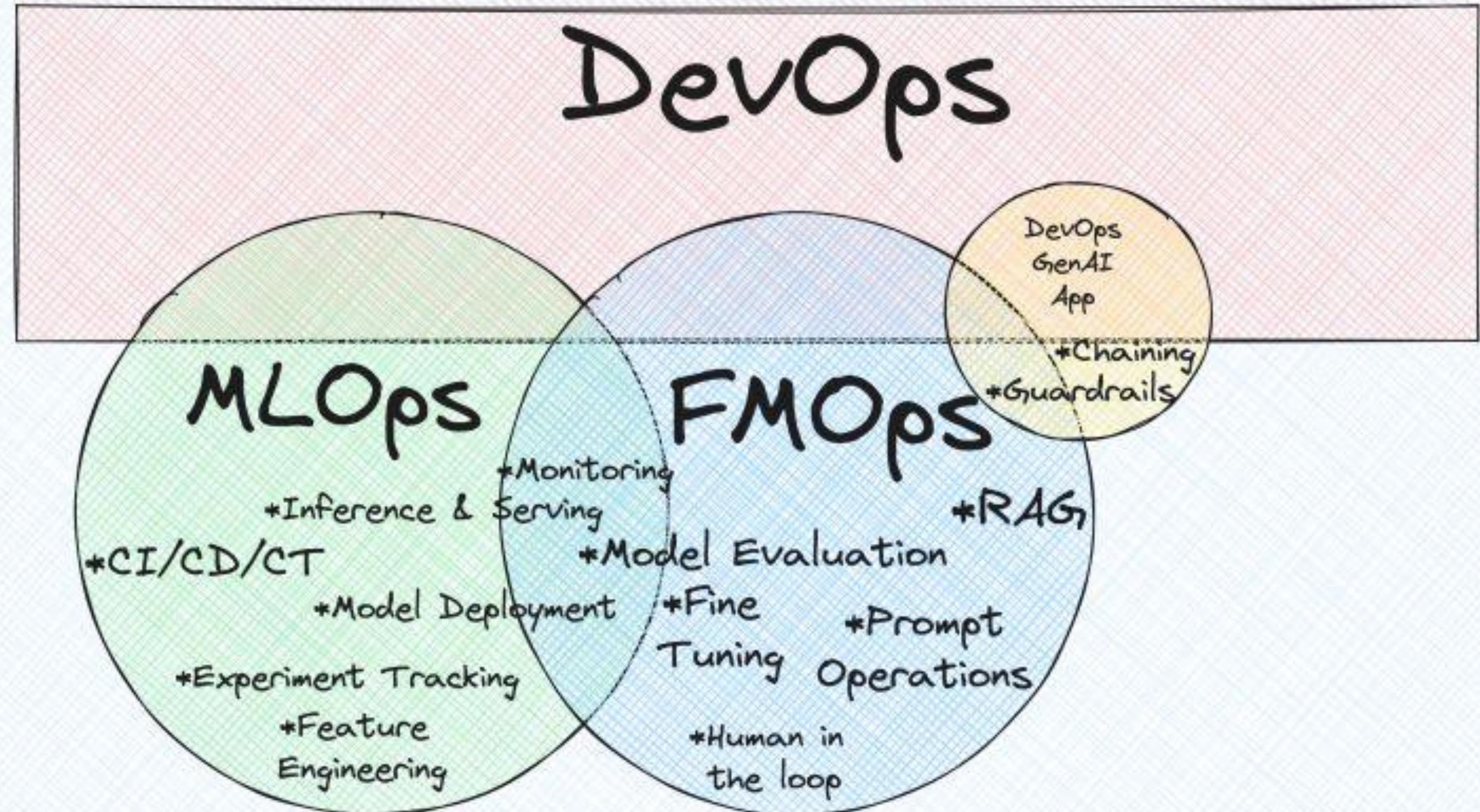
- Improve collaboration and communication between teams
- Automate processes to increase efficiency and reduce errors
- Ensure smooth deployment and monitoring of systems/models
- Foster a culture of continuous improvement and learning

AI Ops

Common alignment on Core Tenants:

- LLMOps and MLOps aim to streamline the deployment, monitoring, and maintenance of machine learning models in production environments.
- Involve collaboration between data scientists, engineers, and operations teams.
- Focus on ensuring model performance, scalability, and reliability.

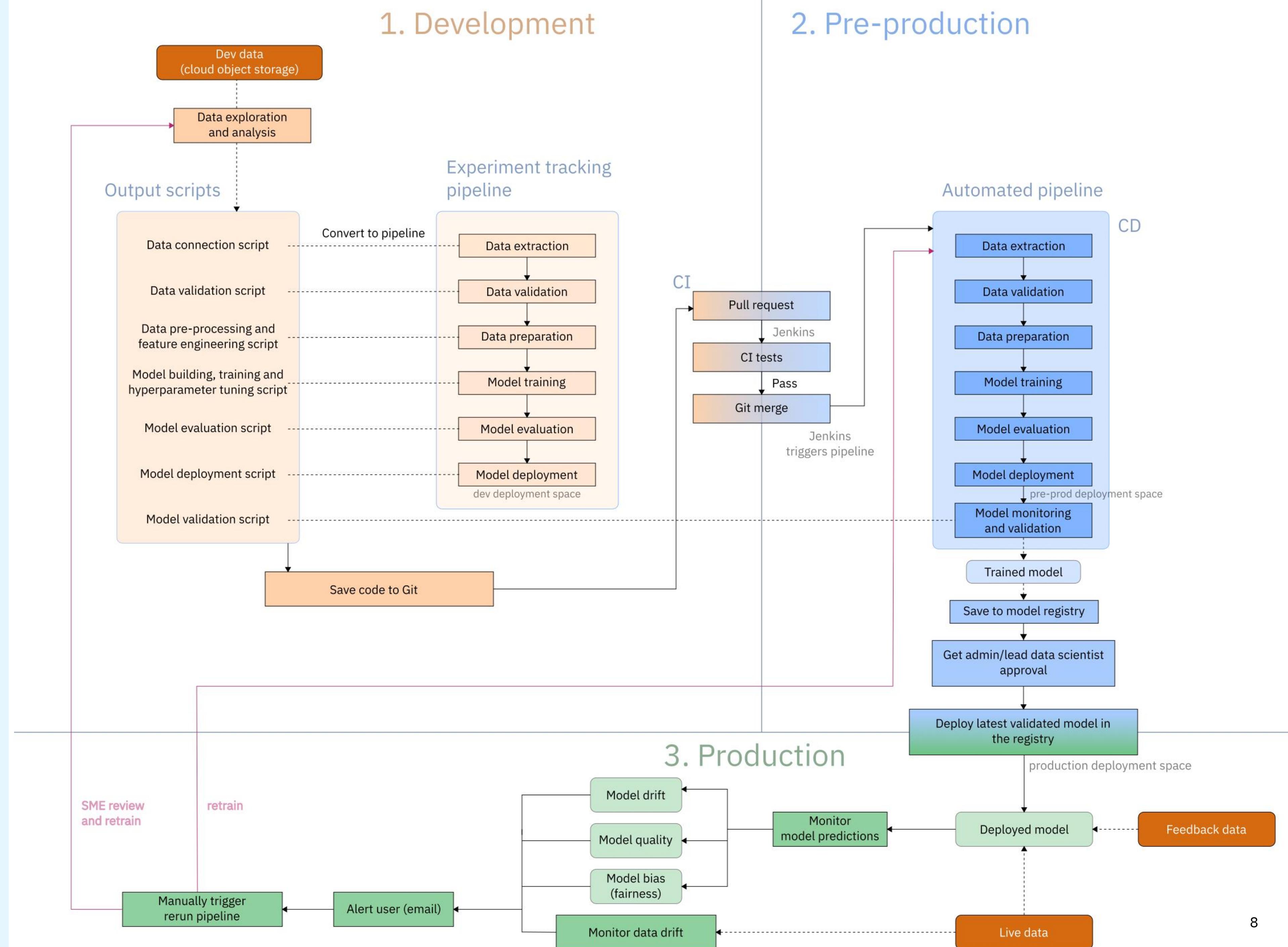
xOps in the world of Generative AI



Visualisation by Eduardo Ordax (<https://www.linkedin.com/in/eordax/>)
Original Idea by Eduardo Ordax

MLOps

Modern Watson Studio
MLOps Workflow from
Dev to Prod with CI-CD



Where does IBM's platform fit?

Development Environments

Flexible microservices based development architecture with customization support.

The screenshot shows the 'Orchestrate an AI pipeline' project in the IBM Cloud Pak for Data interface. The left sidebar lists assets like 'Mortgage approval pipeline' and 'Integrate Mortgage Data'. The main area displays a 'Storage' section with 186.86 KB used, and a 'Project history' section showing recent activity from 'Data Scientist' and 'Jan 26, 2023 01:23 PM'. A 'Scatter plot chart' is visible, showing a correlation between 'Income' and 'Loan Amount'.

Production Hosting

Secure, auditable, and production hosting for analytic assets and modeling products.

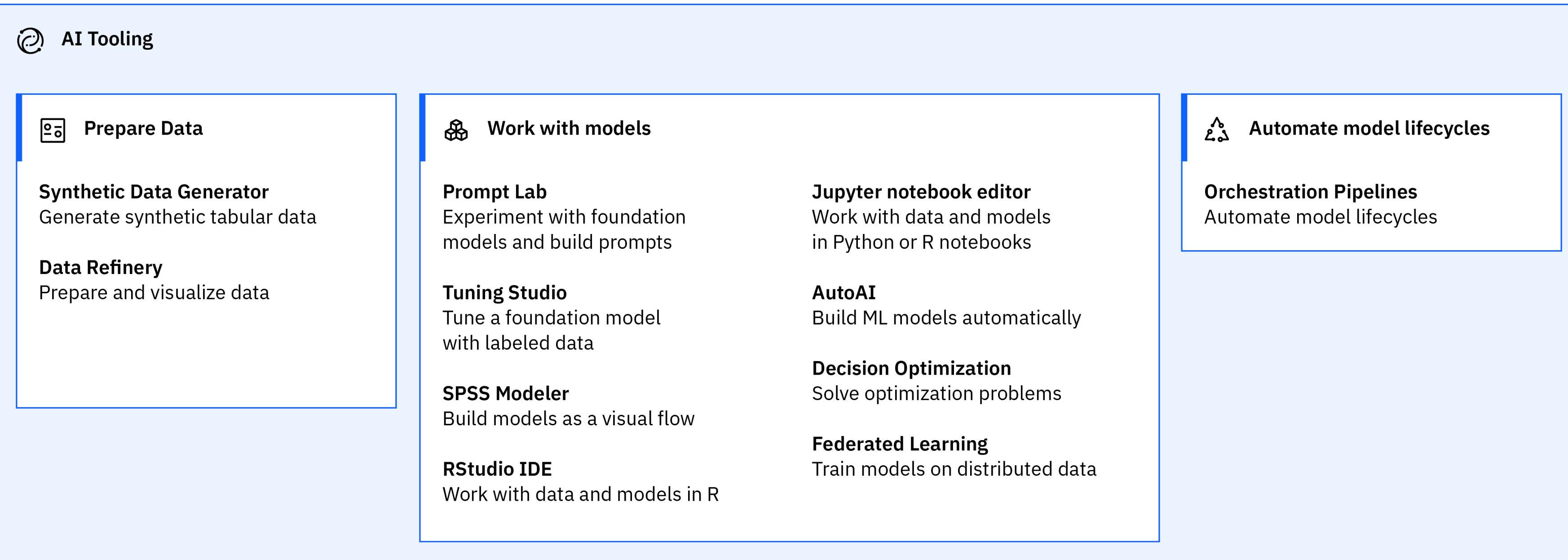
The screenshot shows the 'Mortgage_Customer.csv_flow' project in the IBM Cloud Pak for Data interface. It features a 'SCATTER PLOT CHART' with various filters and settings. The main view displays a scatter plot of 'Loan Amount' versus 'Income' with bubbles of different sizes and colors representing other variables.

Data Handling

Abstracted data layer for governed, traceable, and collaborative data integration.

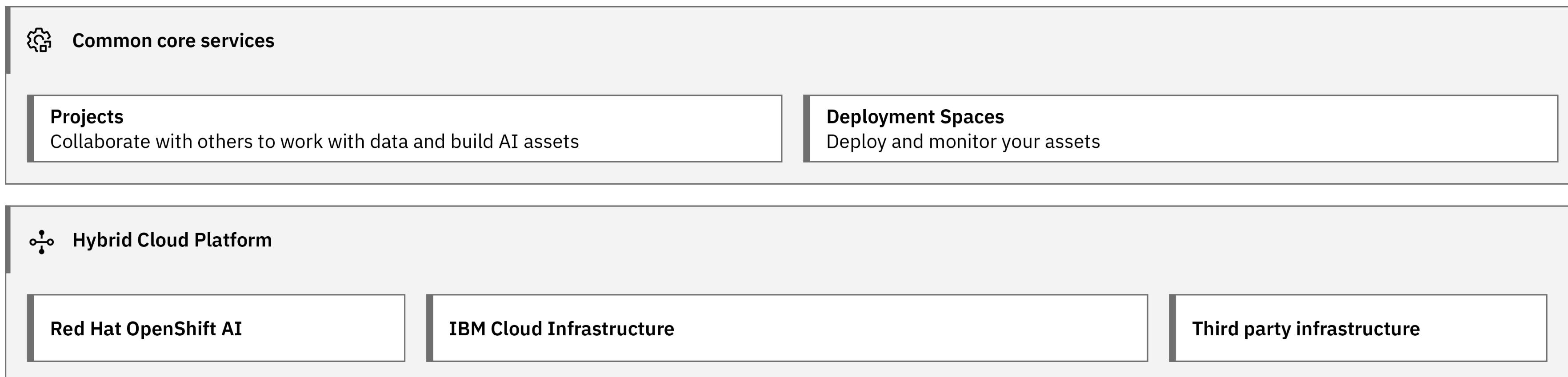
The screenshot shows the 'titanic_train.csv' profile in the IBM Cloud Pak for Data interface. It includes a 'Profile' tab with detailed statistics for columns like 'PassengerId', 'Survived', and 'Pclass'. Below this are three bar charts showing the frequency of values for 'Frequency', 'Survived', and 'Pclass'.

IBM watsonx.ai architecture

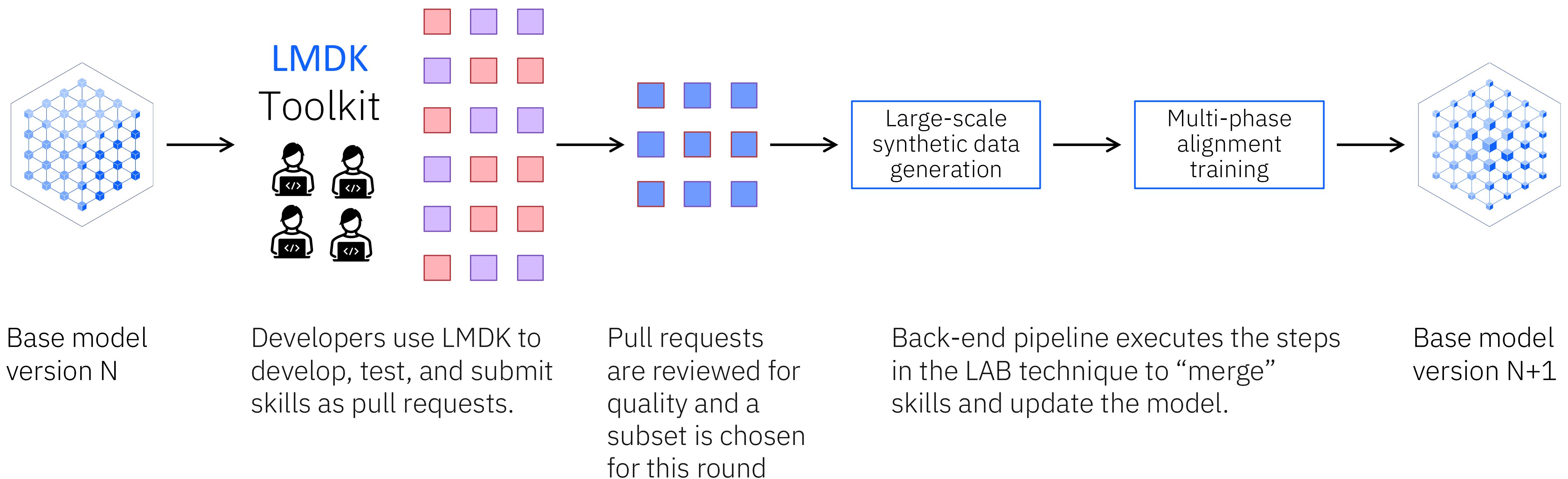


Common core services

- Collaborative projects
- Deployment spaces
- Jobs
- Notifications
- Common connectivity
- Access and Authentication
- Resource management
- Central asset management system



InstructLab Workflow for Model Training



Model inventory

Model use cases Manage

Filter by Tags Status Alert Cata

Find model use cases

Healthcare
Sepsis Onset

Status
Promoted to pre-production

Business terms

Tags

healthcare sepsis ml classification
phi emr

[View details](#)

Model Registry

- Use Case Details
- Associated Assets
- Approval Workflow
- Model Metadata
 - Training
 - Algorithm
 - Schema

Preview asset Profile Data quality Visualization

Feature group β

Features

This table shows features in your feature group. Click a feature name to view and edit details.

<input type="checkbox"/>	Name	Role	Description	Fairness information
<input type="checkbox"/>	Age	Input	Age	Yes
<input type="checkbox"/>	HR_delta_fe	Input	Heartrate delta between last monitored interval	-
<input type="checkbox"/>	Temp_delta_fe	Input	Temperature Delta of Change in the Last Hour	-

Feature Store / Catalog / EDA

- Engineered Features for Data
- Repeatability / Understanding
- Metadata Enrichment
- Lineage

MLOps

DataOps

Facts and lineage

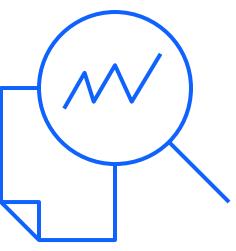
Promote models to a catalog

- Add metadata – tags, reviews, classifications, business terms, description
- Fully searchable and governed

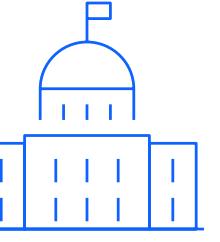
Lineage

- Track consumption and version changes
- Identify related datasets

Model governance in the knowledge catalog



The screenshots illustrate the governance features of the IBM Cloud Pak for Data knowledge catalog. The top screenshot shows the 'Activities' tab for a model named 'AutoAI Attrition 2019 - P7 DecisionTreeClassifier'. It displays metrics such as 'Added to projects 1', 'Last event Model published to project', and 'Last updated 36 days ago'. The bottom screenshot shows the 'Overview' tab for the same model, providing detailed information such as 'Added: May 26, 2021 3:15 AM', 'Size: 12 KB', 'Tags: attrition, employee, termination', 'Reviews: 1 review', and a list of 'Relationships' including 'Is implemented by Employee'.



Model registry and lifecycle tracking

- Entry point for keeping track of model attributes and status
- Track deployments, versions, and status

The screenshot displays two main interfaces of IBM Cloud Pak for Data:

Model registry: This interface shows a list of models with various filters (Tags, Status, Owner, Catalog, Classification, Business term). A specific model, "Credit Risk", is highlighted. Its details are shown in a modal overlay, including Fairness (80%), Quality (0.80), and Drift (10%). Other models listed include "Mortgage Risk" and "Cost estimator".

Catalogs / Banking catalog / Credit Risk: This interface shows the "Credit Risk" model entry. It includes tabs for Overview, Models (selected), Access, Review, and Activities. The "Models" tab displays the model's status across different environments: Trained in project, In development, In pre-production, and In production. The "In production" environment shows the model deployed to "Banking production" with sub-components "Credit Risk Model" and "Credit Risk Deployment". An "Approved" status is indicated for the deployment. The "Published models" section shows the model published to the "Banking catalog".

Model validation



Deployment monitoring

- Validate pre-production models
- Compare models performance
- Continual gathering of quality and trust metrics

Model Risk Governance

- Integrates with Watson Studio model monitoring metrics
- Store, manage, and monitor model metrics
- Calculate risk scores and impact for models

Dashboard / Credit Risk Evaluation

Model Credit Risk Pre-production

Description Evaluates credit applications for risk signals and applies a RISK or NO RISK verdict.

Model ID fad9a34f-8e48-4231-a718-131...

Fairness 90% Green within threshold

Fairness by feature

Age	90%
Sex	91%
Race	92%

Quality .99 Green within threshold

Quality metrics

- Area under ROC
- Area under PR
- Accuracy
- True positive rate (TPR)
- False positive rate (FPR)
- Recall

3 tests run

3 Tests passed

Evaluate Compare model Description Select a model for comparison. Credit Risk V2

Send to OpenPages Select the metrics to send to OpenPages. If the metric is not yet available in OpenPages, a new metric will be created automatically.

Quality measures

- Area under ROC
- Area under PR
- Accuracy
- True positive rate (TPR)
- False positive rate (FPR)
- Recall
- Precision
- F1-measure
- Logarithmic loss

Fairness measures

- Fairness
- Age
- Sex

Performance measures

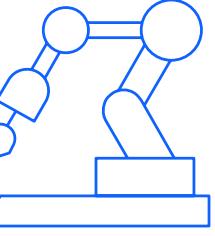
- Throughput

Drift measures

- Drop in accuracy
- Drop in data consistency
- Estimated accuracy
- Base accuracy

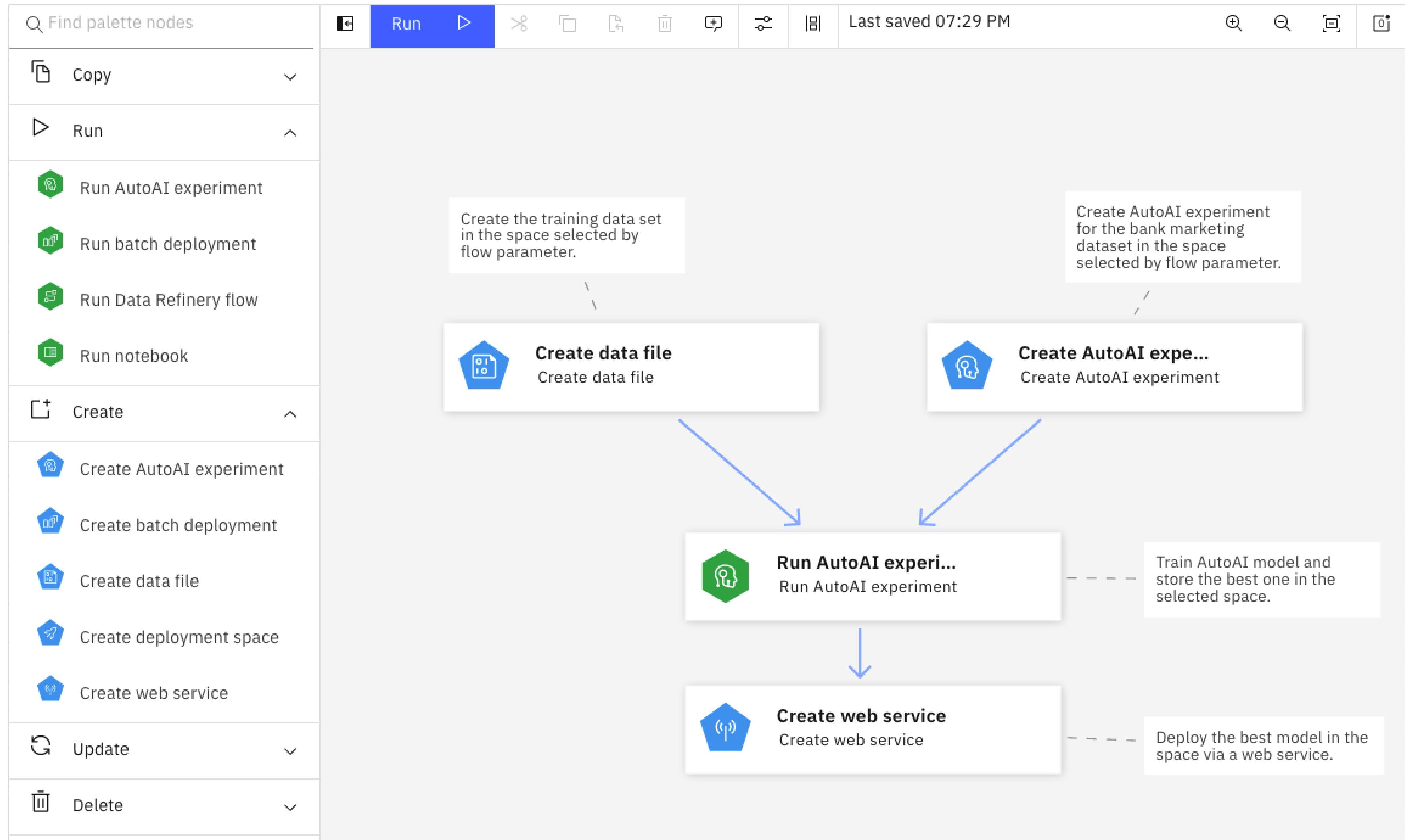
Cancel Schedule

Orchestrate AI modeling



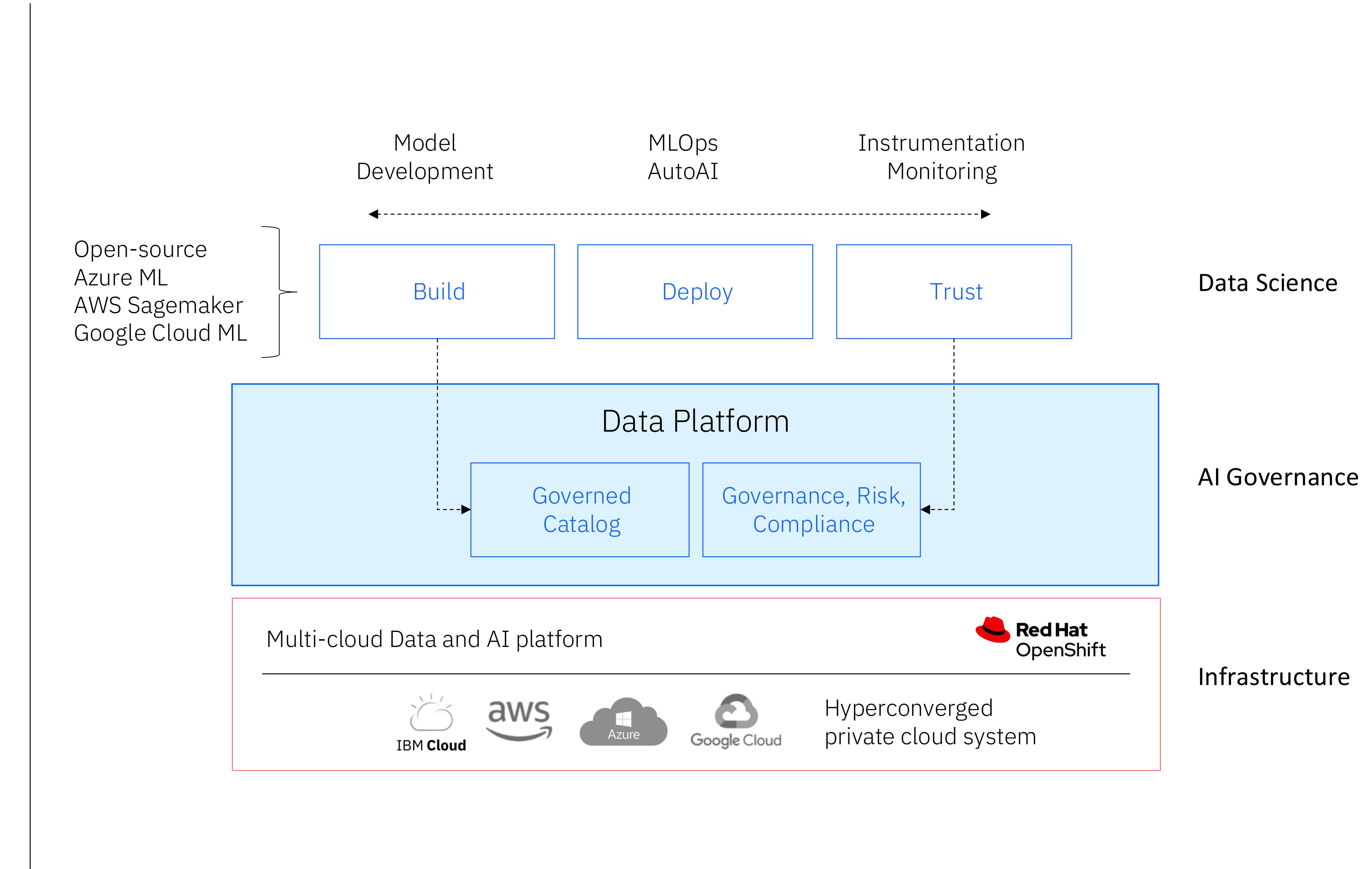
Automated pipeline interface

- Combine tasks from different tools
 - Notebooks
 - Data refinery
 - Model creation
 - Model deployments
- Automate model training with new data sets
- Automatically deploy better challenger models

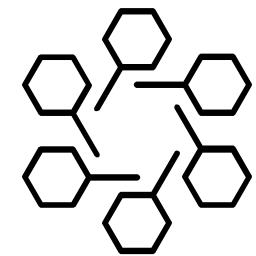


MLOps and Trustworthy AI platform

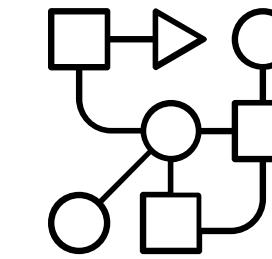
Modular components running on any technology stack



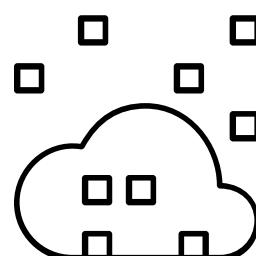
watsonx.ai Usage Best Practices



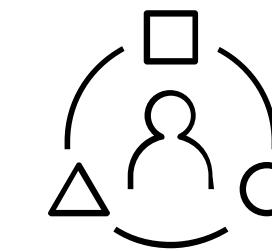
Decide early if you are going to use Git or internal collaboration



Automate with Jobs, Pipelines, and Deployment Spaces



Define Projects by Use Case and Identify Stakeholders and Required Data



Define Roles for Participants, with an emphasis on Project Owner

Secure Coding (Collaboration)



Collaborating with Sensitive Credentials – 4 Methods (Non Exhaustive)

- Insecure Example

```
import types
import pandas as pd
from botocore.client import Config
import ibm_boto3
def __iter__(self): return 0
# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes
# You might want to remove those credentials before you share the notebook.
client_01da3b8d07aa40ca85ec5cee0637167f = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='KEYID-EXAMPLE',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3-api.us-geo.objectstorage.service.networklayer.com')
body = client_01da3b8d07aa40ca85ec5cee0637167f.get_object(Bucket='cloudproject-don'
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType(__iter__, body)
df = pd.read_csv(body)
df.head()
```

1 Secrets

Preferred Method:

<https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=2-managing-secrets-vaults>

3 File

Import credentials from a resident file. Add to .gitignore.

2 Dotenv

Load environment variables from .env file(untested)

[\(https://pypi.org/project/python-dotenv/\)](https://pypi.org/project/python-dotenv/)

4 Getpass

Interactively prompt for user input.

Disrupts automated flows.

[\(https://docs.python.org/3/library/getpass.html\)](https://docs.python.org/3/library/getpass.html)

Overview of secrets and vaults

A *secret* contains sensitive data

The information in the secret is stored in a secure and encrypted environment that conforms to your organization's policies.

The services and connections that use the secret do not have direct access to the information in the secret.

A *vault* is a secure place to store and manage secrets.

The information in the secret can be updated once. The change is automatically picked up by all services or connections that use the secret.

Cloud Pak for Data includes an internal vault. If you have a supported enterprise-grade vault, you can also connect to external vaults.

Secrets (API and GUI)

*Not Available on aaS

Add secret

Secret overview

Name

Display name for the secret

Description

0/100

Description of the secret and how the secret is used

Vault (i)

INTERNAL_VAULT

Secret details

Secret type (i)

Type of content in secret

Username and password

Key

Share secret (optional)

You can allow other users and groups to use this secret to enter sensitive information. The user or group will not be able to edit, delete or see the contents of the secret.



Name

Email

No users selected

You haven't shared anything yet

```
# DOING SECRET (Note: This code example)
secret = {
    'secret_name': 'example_ibm_api_key',
    'description': 'IBM Cloud API Key',
    'secret': {'apikey': 'INSERT YOUR KEY HERE'},
    'type': 'credentials',
    'vault_name': 'internal'}
```

Create Vault Secret

Store a secret in the vault so we can retrieve later for IBM Cloud API Key

```
# @hidden_cell

import os
import json
import requests
from requests.packages.urllib3.exceptions import InsecureRequestWarning

requests.packages.urllib3.disable_warnings(InsecureRequestWarning)

url = "https://cpd-cpd-instance.apps.cpd47pub2.tec.ihost.com/zen-data/v1/secrets"
token = os.environ.get('USER_ACCESS_TOKEN')

headers = {
    'Authorization': 'Bearer {}'.format(token),
    'Content-Type': 'application/json',
}

payload = json.dumps(secret)
response = requests.post(
    url,
    headers=headers,
    data=payload,
```

External File

Parse an external file hidden from Github

```
: import secrets  
api_key = secrets.api_key
```

Jupyter Notebook Script File Reference

Unlike Jupyterlab with Git, Jupyter Notebooks local data directories are ephemeral and destroyed after directory here;

Jupyter Notebook Working Directory (Deleted on Environment Shutdown)

```
: !pwd  
/home/wsuser/work
```

As a result, script files you place in that folder will disappear when the environment is shut down. Our doc in the project data assets.

I would rather just import the script from the data assets folder directly. You can browse the data assets environments.

Project Data Assets

```
: !ls /project_data/data_asset/  
__pycache__ script.py
```

You cannot simply reference a script above the working directory though without it being included in the path. Let's use the `sys` library to append the data assets directory to the path and then import our script.

Add Project Data to the Search Path for Python

```
: import sys  
sys.path.append('/project_data/data_asset/')
```

File – Import and Reference

- Does not Prevent Access from Project Members
- Extra Method Required for CPDaaS

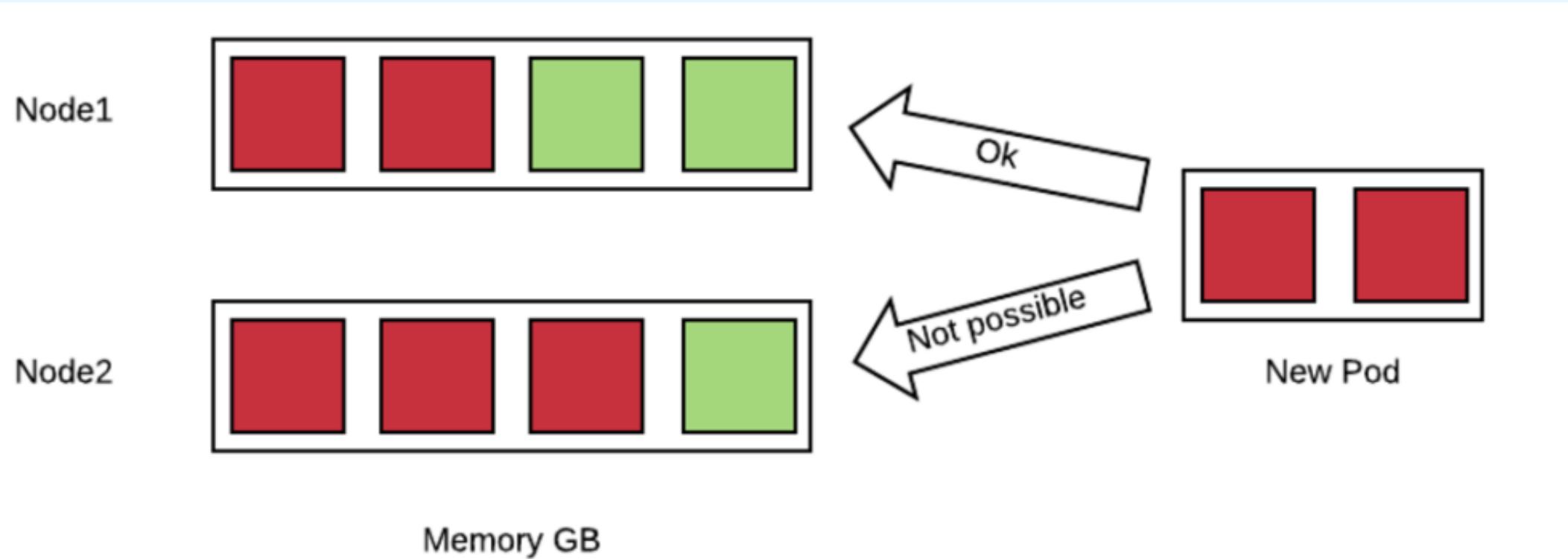
* Add to .gitignore

Resources (Capacity Management)



Levels of Resource Considerations (Non-Exhaustive)

- Openshift Scheduler



1 Monitoring

Resource overhead understanding.

<https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=2-best-practices-monitoring>

2 Code

Memory management,
Frameworks, Code Optimization

3 Load

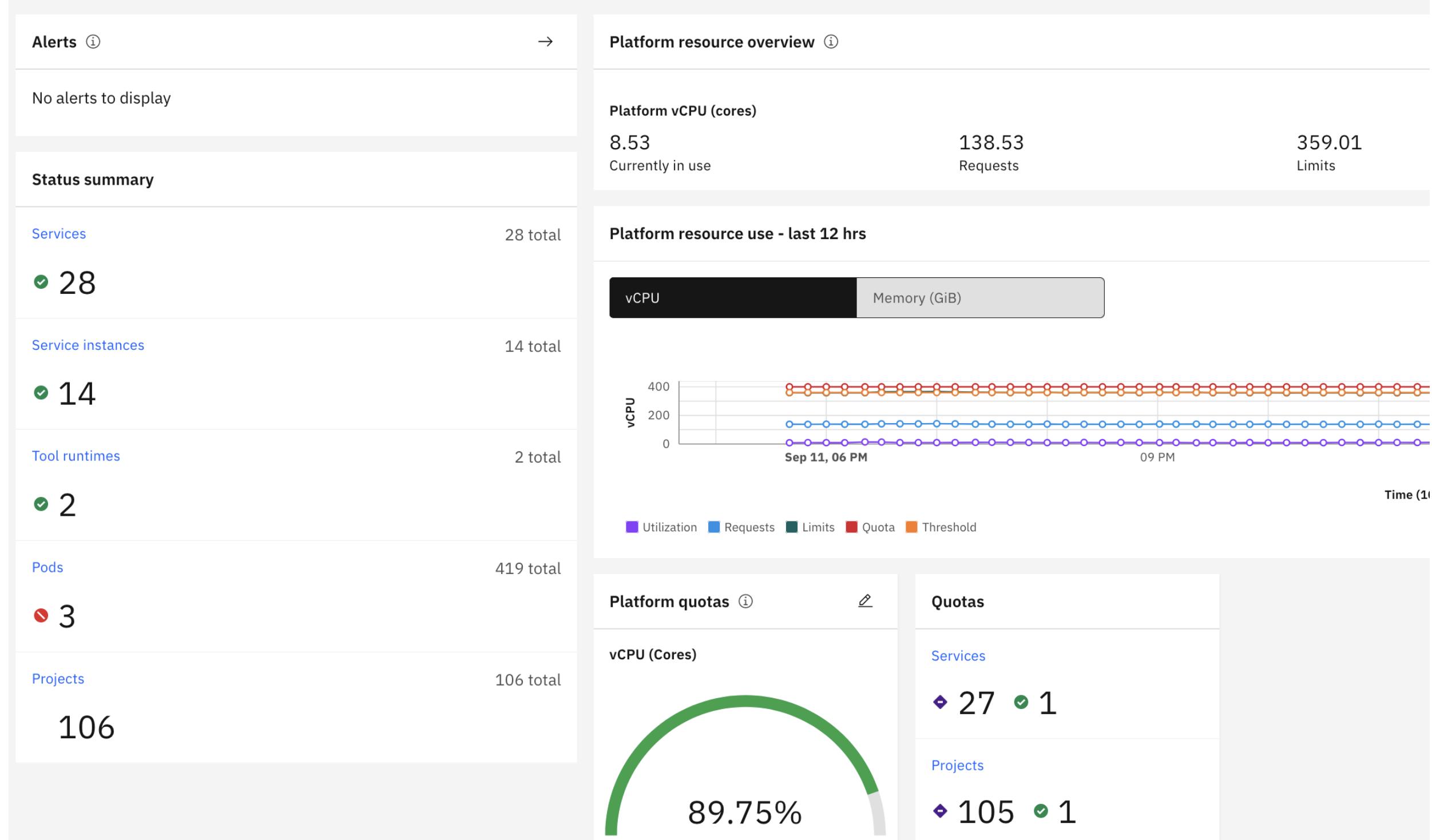
Competing workloads,
user volumes, active projects

4 Data

Volume of data, type of data, operations required.

Overview of Monitoring and Controls

Monitoring



Cluster Capacity Reporting native to CP4D. CPDaaS Usage Monitoring for CuH with workload runtime reporting. OpenShift monitors and Prometheus Integration

Resource Quota Support:
Per Namespace, Per Service, Per Project. Per User Threshold

Autoscaling services using the Horizontal Pod Autoscaler (HPA). Changes the resource allocation of services by increasing or decreasing the number of pods in response to CPU or memory consumption.

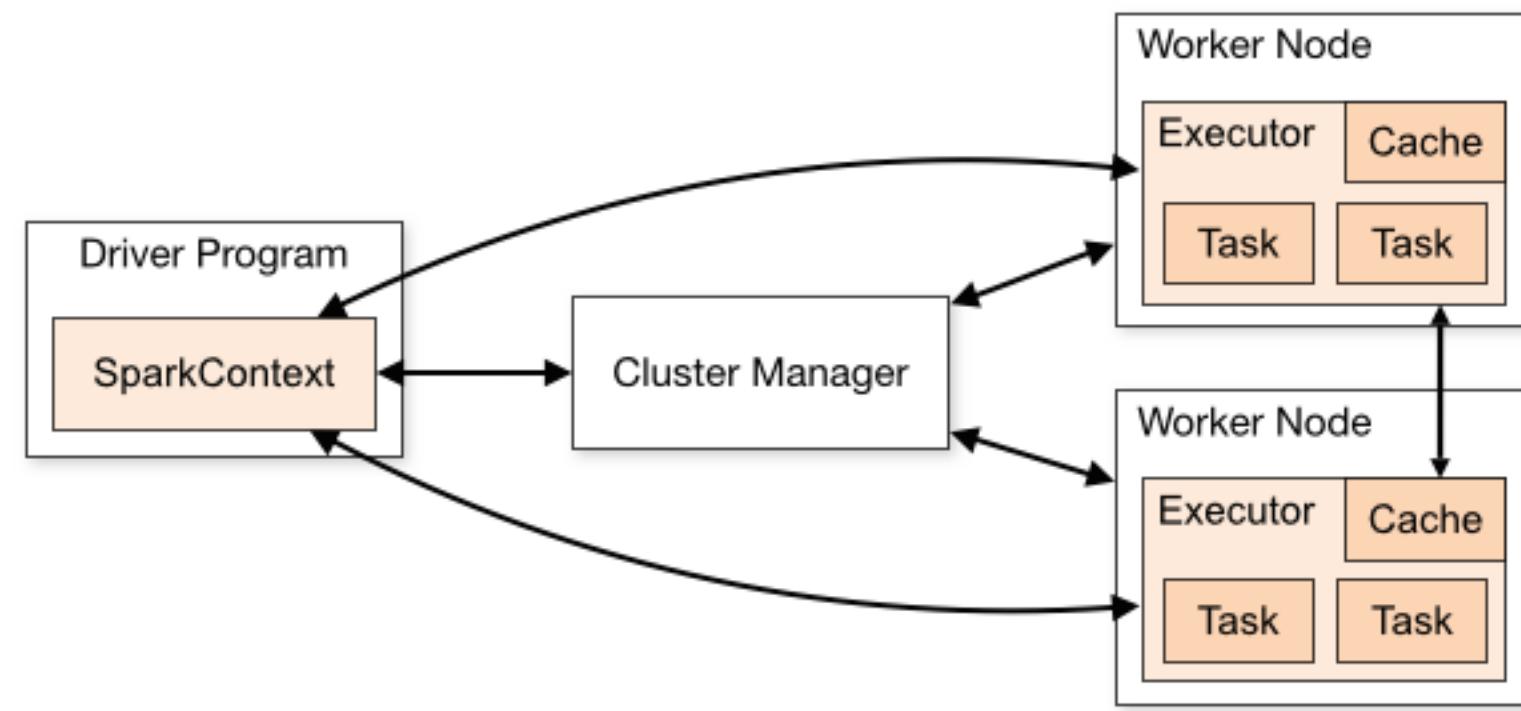
Cluster Scaling support with Machinesets, Cloud Providers instrumented autoscalers with Terraform

WMLA GPU management Options

MIG (Multiple GPU Instances)

Code – Optimization, Framework Considerations

Spark:



- Arrow with Inferred Schema

```
import itc_utils.flight_service as itcfs
readClient = itcfs.get_flight_client()
RDK_COS_WL_data_request = {
    'connection_name': '""RDK-COS-WL""',
    'interaction_properties': {
        '#row_limit': 5000,
        'file_name': 'labels.csv',
        'infer_schema': 'true',
        'infer_as_varchar': 'false'
    }
}
flightInfo = itcfs.get_flight_info(readClient, nb_data_request=RDK_COS_WL_data_request)
data_df_1 = itcfs.read_pandas_and_concat(readClient, flightInfo, timeout=240)
data_df_1.head(10)
```

Apache Arrow reduces the time to serialize data in memory, and it can also reduce the footprint and time to read/process DataFrames
<https://arrow.apache.org/>

Memory Management:
Garbage Collection,
functions, data
structures (Pandas, R
Dataframe), interim
sets, and sampling

Alternate frameworks:
Spark, larger than
memory datasets.
Distributed compute
with executors. Other
options.

Memory Monitoring in
Jupyter. Dead Kernel,
pod logs
(administration/
monitoring)

Trusted | Python 3.10 O ↗
Memory:320.1 MB / 2 GB

Load – Managing with MLOps, Understanding Limits/Requests

Details [YAML](#)

```
64      | | | | | 'f:requests.memory': {}
65  spec:
66    hard:
67      limits.cpu: '4'
68      limits.memory: 4Gi
69      requests.cpu: '1'
70      requests.memory: 2Gi
71  status:
```

Resource	Requests	Limits
cpu	3861m (96%)	3240m (81%)
memory	14396Mi (91%)	17302Mi (109%)
---	---	---
Resource	Requests	Limits
cpu	3746m (93%)	190m (4%)
memory	12336Mi (78%)	13274Mi (84%)
---	---	---
Resource	Requests	Limits
cpu	3362m (84%)	1720m (43%)
memory	12518Mi (79%)	13728Mi (87%)

Limits and Requests:

Limits are a hard reservation for pod resources, requests are an overcommit up to. Environment Definitions.

Dev/QA/UAT/Prod Acceptance asset movement. Git, CI-CD, APIs and SDK

<https://github.com/IBM/cpdctl>

Multiple Workloads: Namespace isolation, quota assignment and other solution stacks.

Environment Sizing specific to need. Destroy and recreate within bounds.

APIs and SDK

Related but Different Purposes

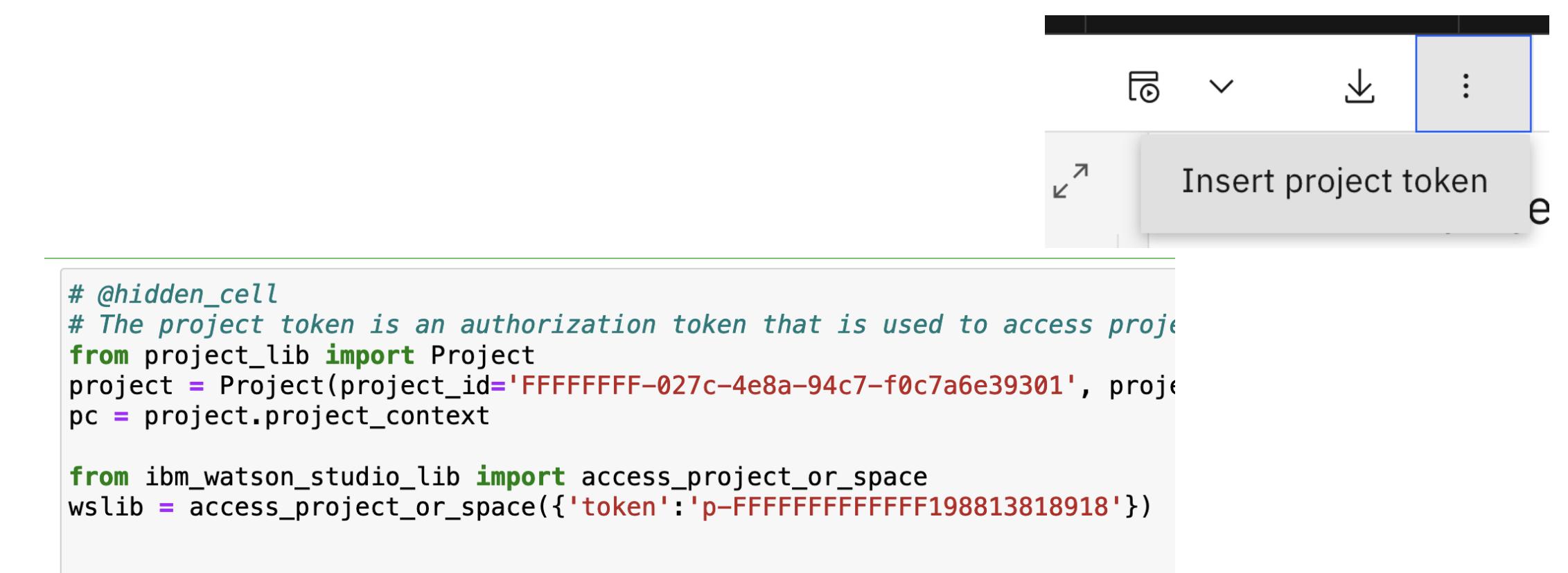
Programmatically interface with projects, data, and assets within project spaces. *Project Lib
Deprecated

Samples Repo:

<https://github.com/IBMDatascience/sample-notebooks/blob/master/CloudPakForData/notebooks/4.0/IPYNB/Working%20with%20ibm-watson-studio-lib%20in%20CPD.ipynb>

Documentation:

<https://www.ibm.com/docs/en/cloud-paks/cpdata/5.0.x?topic=lib-watson-studio-python>



A screenshot of a code editor interface. A tooltip window is open at the top right, containing the text "Insert project token". The main code area shows Python code for interacting with a project space using the Project Lib and IBM Watson Studio Lib.

```
# @hidden_cell
# The project token is an authorization token that is used to access projects
from project_lib import Project
project = Project(project_id='FFFFFFFFFF-027c-4e8a-94c7-f0c7a6e39301', project_token='p-FFFFFFFFFF198813818918')
pc = project.project_context

from ibm_watson_studio_lib import access_project_or_space
wslib = access_project_or_space({'token': 'p-FFFFFFFFFF198813818918'})
```

Programmatically instantiate models, package metadata, manage deployments, and deployment spaces.

Samples Repo: <https://github.com/IBM/watson-machine-learning-samples>

Documentation: <https://ibm.github.io/watson-machine-learning-sdk/>

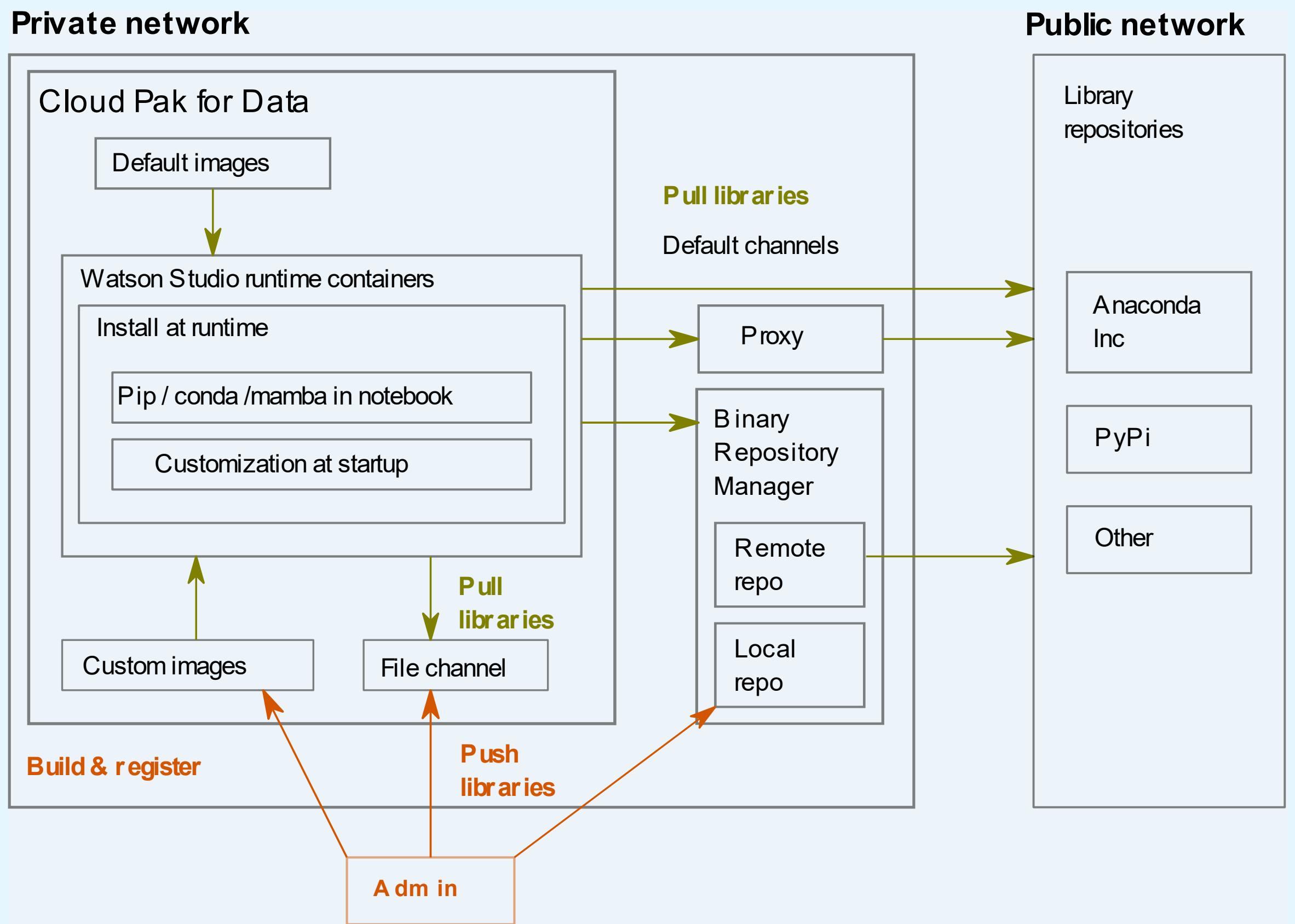
- Persist Data without Exposing in UI: `wslib.storage.save_data()` `wslib.upload_file()` `wslib.save_data()`

Environments (container management)



Environment Tuning & Customization

– Image Customization



1 Libraries

Installing required libraries from the internet or trusted repositories

2 Preload

Environment template customizations and channel specifications

3 Images

Custom built images for specific functionality and use case

4 Labels

Restrict pods to nodes by requirement. Say GPU, extra memory?

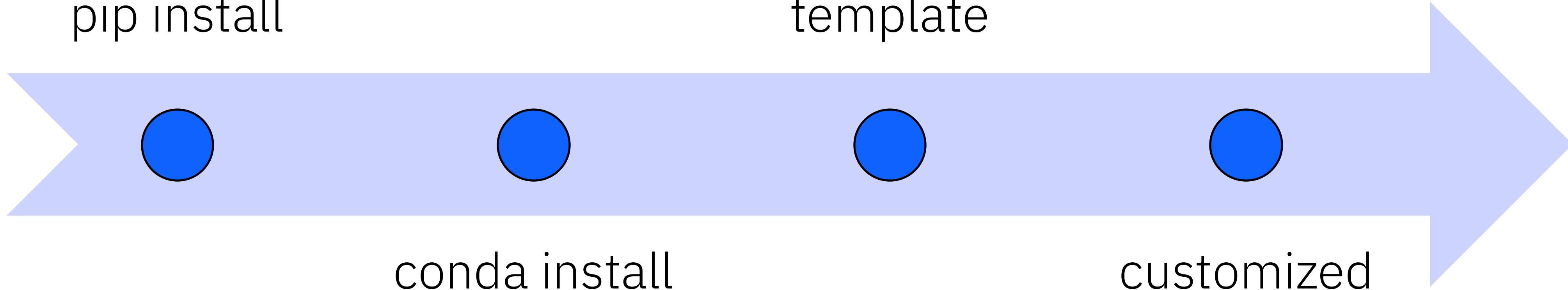
Library Management (in order of increasing pain)

pip install

customized
environment
template

conda install

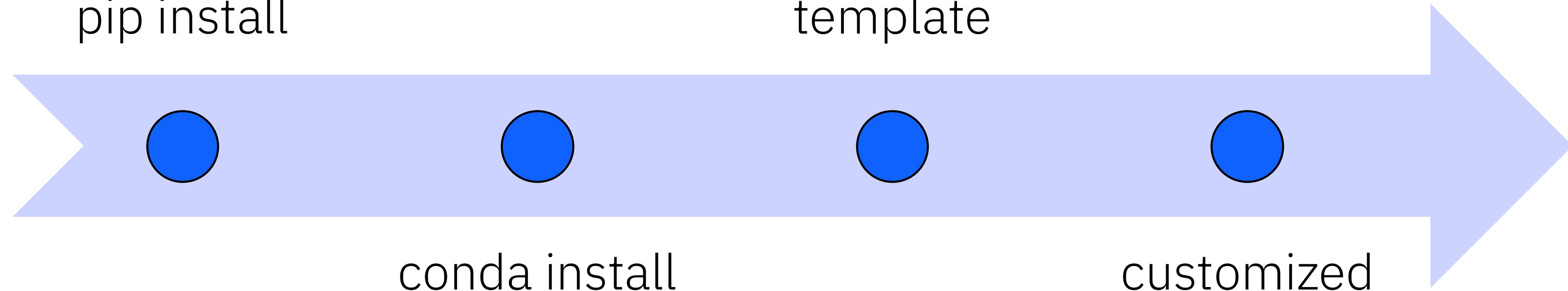
customized
image



Library Management (in order of increasing pain)

pip install

customized
environment
template



conda install

customized
image

Jupyter Magic Commands: !pip %pip !conda

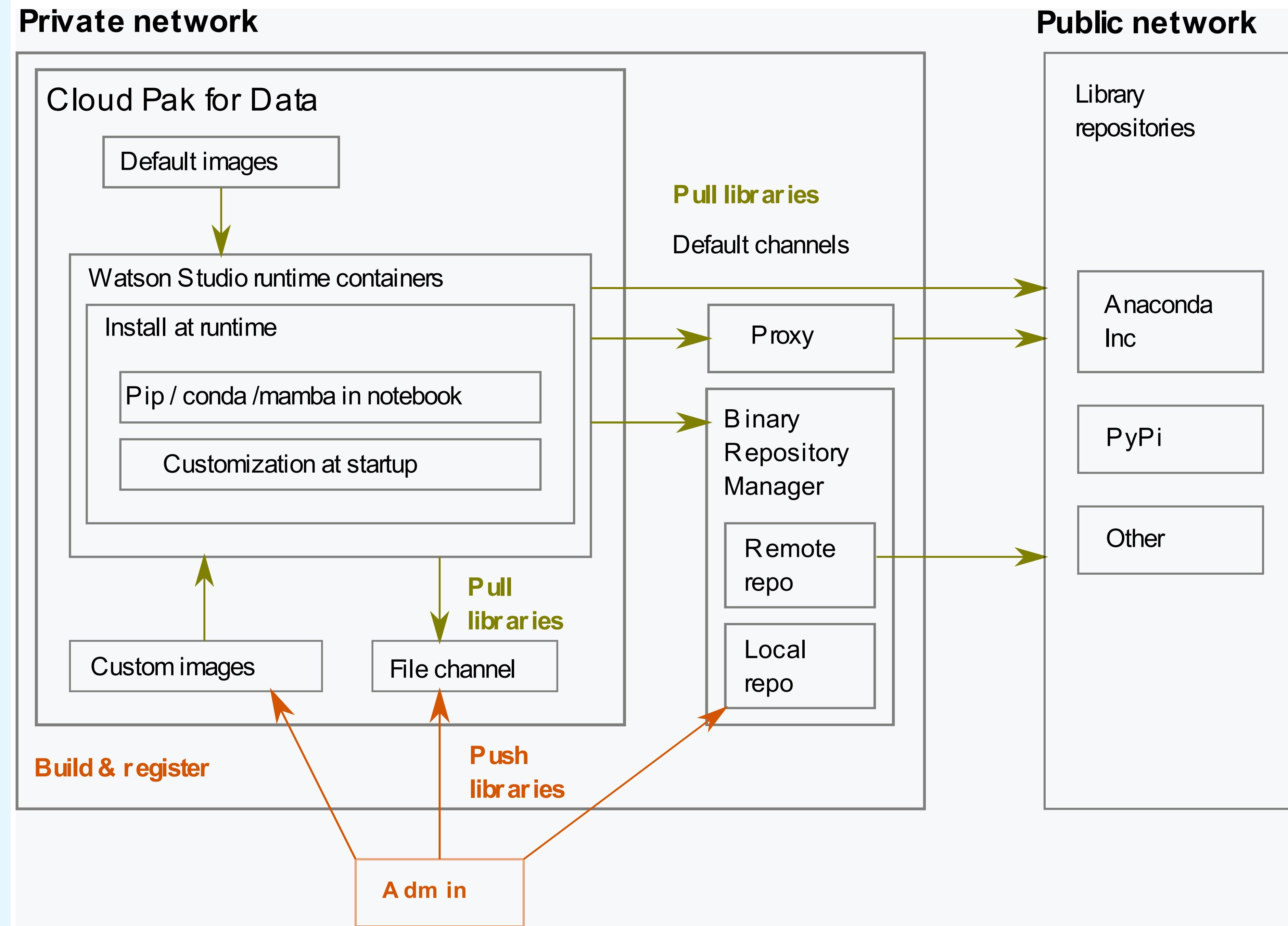
pip and conda install when you want to quickly add a few libraries in the running environment
conda solver is much longer and memory intensive

Environment templates execute per container initialization (watch that start time)

Custom images for more invasive libraries (especially system library requirements) and LARGE frameworks

Custom Images

- Admin Required*
- Download Configuration File
- Download the Image
- Build Image
- Register the Image
- Push the Image
- Update Configuration



Process Outline

1. Pull Instance Configuration from Data API
2. Fetch Available Image Runtimes
3. Pull Specific Runtime Configuration
4. In runtime configuration find image id from registry:
such as; wslocal-runtime-py310main@sha256:491c86cb5604a1fe0e11d1...
5. Login to Registry and Pull Image ID
6. Edit Dockerfile with Customizations
7. Build updated image
8. Push image to registry
9. Update configuration file with updated image id
10. Push configuration file to cluster.

<https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=environments-building-custom-images>

Dockerfile Best Practices

1.Run an image build that installs the packages you require.

2.From the build log output , collect a list of all packages and versions that were installed.

3.Add comments to the install commands without package versions. These comments will be useful when packages need to be updated.

4.Add new install commands with the full list of packages and their versions.

5.Rebuild the image.

```
FROM ${base_image_tag}

# =====
# root section
# =====
USER root:root

COPY install-as-root.sh /tmp/install/
RUN umask 022 \
    && bash /tmp/install/install-as-root.sh \
    && /opt/ibm/build/bin/microdnf-clean.sh \
    && rm -rf /tmp/install
# The microdnf-clean.sh script also grants required permissions.

# -----
# Change display name of the Jupyter kernel
# -----
# Changing the display name of the kernel allows for distinguishing different
# custom images from the default runtime images when editing a notebook.
# It is good practice, but not strictly required.

RUN sed -i -e '/display_name/{s//, / with modifications"/}' \
    /opt/ibm/run/kernels/*/kernel.json

# =====
# user section
# =====
USER wsbuild:wsbuild

COPY --chown=wsbuild install-as-user.sh /tmp/install/
RUN umask 002 \
    && bash /tmp/install/install-as-user.sh \
    && /opt/ibm/build/bin/fix-conda-permissions.sh \
    && conda run -n $DSX_KERNEL_CONDENV pip cache purge \
    && conda clean --tarballs \
    && rm -rf /tmp/install

# ====
# conda
# ====
USER nobody:wsbuild
```

Labels and Pod Customizations

RSI (Resource Specification Injection): Customize Specifications of Pods

- Labels
- Environment Variables
- Annotations
- Node Affinity
- Init Containers or Sidecar Containers
- Custom Resource Requests

– OCP Authentication

```
cpd-cli manage login-to-ocp \
--username=${OCP_USERNAME} \
--password=${OCP_PASSWORD} \
--server=${OCP_URL}
```

– Install RSI (public registry)

```
cpd-cli manage install-rsi \
--cpd_instance_ns=${PROJECT_CPD_INST_OPERANDS}
```

<https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=2-customizing-pod-specifications-rsi>

Additional Cool Features

VSCode Extension:

Must be enabled by admin on the cluster. Develop in VSCode and remotely execute in CP4D.

SSH over secure websocket tunnel. Uses cluster configured auth.

<https://www.ibm.com/docs/en/cld-paks/cp-data/4.7.x?topic=scripts-visual-studio-code>

Storage Volumes:

Access NFS and CIFS Shares. Access permissions operates oddly, however. Ownership for CIFS derived from user account creating, NFS from openshift UIDs

Killer use case is, local storage volumes. Use a common shared storage space for cross project data (Library concept).

<https://www.ibm.com/docs/en/cld-paks/cp-data/4.7.x?topic=2-managing-storage-volumes>

Watson Pipelines:

Instrument flows and interactions from assets. Derive a data product, validate quality, use to train model, compare to existing model, promote challenger to production.

Can work across project and deployment spaces.

<https://www.ibm.com/docs/en/cld-paks/cp-data/4.7.x?topic=assets-watson-pipelines>

Q&A



Next Steps

Try watsonx.ai in [IBM Cloud](#)

Join us for hands-on labs at
[IBM TechXChange conference](#)

Fuel Your AI
at the ultimate
IBM learning event

IBM TechXchange Conference

October 21-24, 2024

Mandalay Bay – Las Vegas



Thank You



Ryan Kather
IBM, Sr. AI Engineer – Client Engineering

ryan.kather@ibm.com

@krondor

<https://github.com/krondor/techxchange2023-ws-bp>

Notices and disclaimers

© 2024 International Business Machines Corporation.
All rights reserved.

This document is distributed “as is” without any warranty, either express or implied. In no event shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.

Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM.

Not all offerings are available in every country in which IBM operates.

Any statements regarding IBM’s future direction, intent or product plans are subject to change or withdrawal without notice.

IBM, the IBM logo, and ibm.com are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at: www.ibm.com/legal/copytrade.shtml.

Certain comments made in this presentation may be characterized as forward looking under the Private Securities Litigation Reform Act of 1995.

Forward-looking statements are based on the company’s current assumptions regarding future business and financial performance. Those statements by their nature address matters that are uncertain to different degrees and involve a number of factors that could cause actual results to differ materially. Additional information concerning these factors is contained in the Company’s filings with the SEC.

Copies are available from the SEC, from the IBM website, or from IBM Investor Relations.

Any forward-looking statement made during this presentation speaks only as of the date on which it is made. The company assumes no obligation to update or revise any forward-looking statements except as required by law; these charts and the associated remarks and comments are integrally related and are intended to be presented and understood together.

