EscrowAI 2.0 DRAFT User Manual

EscrowAI 2.0 DRAFT User Manual

Exported on 02/02/2024

Table of Contents

1 Common workflows that remain unchanged: 11
2 New public doc space for BETA docs 12
3 Updates for 1:1 Training 13
4 Validation criteria 14
5 Updates for 1:Many Validation 15
6 EscrowAI 16
6.1 Data Steward and Algorithm Owner 16
6.2 EscrowAI's Trusted Execution Environment (TEE) 16
6.2.1 EscrowAI security advantages 16
6.2.2 Confidential Computing technology options 17
6.2.2.1 Confidential Containers in an Intel Software Guard Extensions (SGX) Enclave 17
6.2.2.2 VM-isolated Confidential Containers on Azure Container Instances (CC ACI) 17
6.2.2.3 Confidential Virtual Machines (CVMs) 17
6.3 Model training and validation in EscrowAI 18
6.4 How EscrowAI works 18
7 Onboard your project to EscrowAI 20
7.1 Accounts 20
7.2 User Management 20
7.3 Algorithm owner model setup 20
7.3.1 Use EnclaveAPI to access secrets and post reports 20
7.3.1.1 Data Endpoints 20
7.3.1.2 Reporting Endpoints 23
7.3.2 Optional runtime parameters 29
7.3.3 Work with MLflow in EscrowAI 30
7.3.3.1 EscrowAI and MLFlow Integration 30
7.3.3.2 How to Get Started with MLFlow 31

7.3.3.3 Create your MLFlow model 31
7.3.3.4 Logging your experiment with MLFlow 31
7.3.3.5 Review experiment run 31
7.3.4 Examples of models for use with EscrowAI 31
7.3.5 Package your algorithm for upload to EscrowAI 31
7.3.5.1 Contents of the algorithm package 31
7.3.5.2 Folder structure for algorithm package 34
7.3.5.3 Encrypt and upload algorithm package 34
7.3.6 EscrowAI multiple concurrent algorithm runs 34
7.4 Data Steward Azure setup 35
7.4.1 Azure Cloud 35
7.4.2 Data Steward cloud services provisioning workflow 36
7.4.3 Create a Virtual Network 37
7.4.3.1 Send VNET details to BeeKeeperAI 37
7.4.4 Import EscrowAI application into Azure 37
7.4.4.1 Prerequisites 38
7.4.4.2 Steps 39
7.4.5 Create a Storage Account 41
7.4.6 Set up and manage Blob Storage 42
7.4.7 Related Microsoft documentation 43
8 Project creation by BeeKeeperAI 44
8.1 Data Steward VNET details for project creation 44
9 EscrowAI workflows 45
9.1 Collaboration workflow 45
9.2 Common encryption steps 47
10 EscrowAI dashboard 48
10.1 Home page 48
10.2 Project metadata search 48
10.3 Home page elements 49

10.4 High-level project states 50
10.5 Project event notifications via the bell icon 50
10.5.1 Types of events 50
10.5.2 View notifications 51
11 EscrowAI projects 52
11.1 Project page elements 52
11.2 Intra-project chat between users 54
11.3 Common elements of artifact cards 56
12 EscrowAI cryptography 58
12.1 Key storage 58
12.2 Types of keys 58
12.3 EscrowAI encryption tool 60
12.4 Sequence of key use 60
12.5 Key file naming convention 60
12.6 Common encryption steps 61
12.6.1 Download Key Encryption Key from EscrowAI 61
12.6.2 Generate a Content Encryption Key (CEK) 62
12.6.3 Generate and upload a Wrapped Content Encryption Key (WCEK) 62
12.6.3.1 Steps 62
13 Data steward step-by-step in EscrowAI 64
13.1 Project page 64
13.2 Data Steward steps 64
13.3 Curate data per AO Data Specification 66
13.3.1 Download data specification 66
13.3.2 Curate the data 67
13.4 Attest conformance to data specification 67
13.4.1 Upload the data attestation report 67
13.4.2 Upload a new version of a data attestation report 68
13.5 Encrypt dataset 68

13.6 Upload encrypted dataset to Azure Blob storage 69
13.6.1 Background on Azure Blob storage 69
13.6.1.1 Storage accounts 69
13.6.1.2 Containers 69
13.6.1.3 Blobs 70
13.6.2 Instructions to upload encrypted data 70
13.6.2.1 Azure Portal 70
13.6.2.2 Azure Storage Explorer 73
13.6.2.3 Azure Command Line Interface 73
13.6.3 Related Microsoft documentation 76
13.7 Add a dataset URL to EscrowAI 77
13.7.1 Add a new dataset 77
13.7.2 Create a new dataset version 78
13.7.3 Updating the SAS URL 78
13.8 Generate a Signed URL for the Dataset 78
13.8.1 Steps 79
13.9 Review run request and initiate or reject a run 80
13.9.1 Review a run request 80
13.9.2 Reject a run request 80
13.9.3 Initiate a run 80
13.10 Monitor and cancel a run 80
13.10.1 Run status 80
13.10.2 View the status of a run in progress 81
13.10.3 Cancel a run in progress 81
13.10.4 Completed runs 81
14 Algorithm owner step-by-step in EscrowAI 83
14.1 How algorithm runs are identified: version combinations 84
14.2 Create and upload Data Specification 85
14.2.1 Supported file formats for Data Specification 85

14.2.2 Suggested outline of Data Specification 85
14.2.2.1 Section 1: Hypothesis Test and Methodology 85
14.2.2.2 Section 2: Population Description 85
14.2.2.3 Section 3: Data Specification 85
14.2.2.4 Section 4: Truth Standard 86
14.2.2.5 Section 5: Data Set Validation 86
14.2.3 Upload Data Specification 86
14.3 Package your algorithm for upload 86
14.4 Encrypt algorithm files 86
14.5 Upload algorithm with encryption 87
14.5.1 Planning for uploading your algorithm 87
14.5.2 Common steps 88
14.5.3 Encrypt and upload the algorithm 88
14.5.4 Upload a previously encrypted algorithm 89
14.5.5 View Upload Status 89
14.5.5.1 Validating files 90
14.5.5.2 Building algorithm container 91
14.5.5.3 Pushing algorithm to container registry 91
14.5.5.4 Enclave OS build 91
14.6 For validation runs: upload the Validation Criteria JSON 92
14.6.1 Steps 92
14.6.1.1 New Validation Criteria 92
14.6.1.2 New Version of Validation Criteria 93
14.6.2 Example of JSON validation criteria 93
14.7 Define run configuration, send run request 94
14.7.1 How algorithm runs are identified: version combinations 94
14.7.2 Multiple concurrent algorithm runs 94
14.7.3 Planning 95
14.7.3.1 Run configuration: algorithm version and dataset version 95

14.7.3.2 Run configuration: active, non-expired dataset SAS URL 95
14.7.3.3 Run request: RAM size and number of vCPUs on run request 95
14.7.3.4 Run request: optional runtime parameters 95
14.7.4 Create run configuration 95
14.7.5 Send run request 96
14.8 View run in progress 97
14.9 Get run results 97
14.9.1 Download trained model 97
14.9.2 Make successive model training runs 97
14.9.3 Get validation report 97
14.9.4 View performance report for model training run 98
14.9.5 View run logs 98
14.9.6 See run-specific runtime parameters 98
14.10 Make successive training runs 99
14.10.1 Example 99
14.10.1.1 Add training algorithm v1 to EscrowAI 99
15 Model building test cases 101
15.1 <BIGNOTE> 101
15.2 </BIGNOTE> 101
16 Glossary 102
16.1 A 102
16.1.1 Algorithm Owner 102
16.1.2 Attestation 102
16.1.3 Azure Blob Storage 102
16.1.4 Azure Subscription 102
16.2 C 102
16.2.1 Compute Enclave 102
16.2.2 Confidential computing 102
16.2.3 Confidential containers on Azure Container Instances (CC ACI) 103

16.2.4 Confidential Virtual Machine (CVM) 103
16.2.5 Container 103
16.2.6 Content Encryption Key (CEK) 103
16.3 D 103
16.3.1 Data Attestation Report 103
16.3.2 Data Set Version 104
16.3.3 Data Specification 104
16.3.4 Data Steward 104
16.4 E 104
16.4.1 Enclave 104
16.4.2 Enclave Agent 104
16.4.3 Encryption 104
16.5 I 104
16.5.1 Intel SGX 104
16.6 J 105
16.6.1 Just in time model decryption 105
16.7 K 105
16.7.1 Key Encryption Key (KEK) 105
16.8 L 105
16.8.1 Linux VM 105
16.8.2 Managed Identity 105
16.9 M 106
16.9.1 Manifest 106
16.10 N 106
16.10.1 Network Interface 106
16.10.2 Network Security Group 106
16.10.3 Node 106
16.11 P 106
16.11.1 Package manager 106

16.11.2 Pre-encrypted Model 106
16.11.3 Project 107
16.12 R 107
16.12.1 Resource Group 107
16.13 S 107
16.13.1 Sealed Enclave Key 107
16.13.2 Secure Enclave 107
16.13.3 SSL 107
16.13.4 Site 107
16.13.5 Storage Account 107
16.14 T 108
16.14.1 Trusted Computing Base 108
16.14.2 Trusted Execution Environment 108
16.15 U 108
16.16 V 108
16.16.1 Virtual Machine 108
16.16.2 Virtual Network 108
16.16.3 VM Disk 108
16.17 W 109
16.17.1 Wrapped Content Encryption Key (WCEK) 109
16.18 Z 109
16.18.1 Zero Trust Architecture 109
17 EscrowAI Help Center 110
17.1 Login to the Help Center 110
17.2 View all resources on the Help Center 110
17.3 Search the Help Center 110
17.4 Contact BeeKeeperAI technical support 111

This is the DRAFT documentation space for the EscrowAI 2.0 User Manual.

Version 2.0 introduces two significant changes to EscrowAI.

- a one-to-many framework for validation projects
- 1:1 training

Both of these changes are specific to the Algorithm Owner role, and so the changes to the user manual are targeted in the AO section.

1 Common workflows that remain unchanged:

- Create and upload Data Specification

2 New public doc space for BETA docs

Add “Doc feedback” button on same

3 Updates for 1:1 Training

- Integrate Enclave API…
  - I assume this needs updates, based on
    - EAI-2465[1] - MLFlow routing to Enclave API DONE
    - EAI-2466[2] - Report Token Authorization to Enclave API DONE
    - EAI-2467[3] - Validation of the traffic coming from Enclave API DONE
    - Any new REST API endpoints? with Python program examples?
- MLFlow
  - How to Get Started with MLFlow (see page 31).
  - EscrowAI and MLFlow Integration (see page 30)
  - Create your MLFlow model (see page 31)
  - Logging your experiment with MLFlow (see page 31) - what to integrate into your algo code.
  - Review experiment run (see page 31)
- Training parameters added at run request time. -
  - EAI-2706[4] - Allow run parameters to be changed TO DO
  - EAI-2712[5] - Documentation should be added to describe how this functionality (algo upload) works DONE

    .
  - ~~Example program that tests/reads a training parameter on a running enclave and how it affects the algo's behavi~~or. supplied by alex and ridwan
- Additional changes/guidance?
  - Create the algorithm package - does this have additional changes?
  - Encrypt algorithm - any new guidance needed in this section?
- Future Topics:
  - Model exfiltration - hold on this section until we know if/how/when we allow this.

1 https://beekeeperai.atlassian.net/browse/EAI-2465
2 https://beekeeperai.atlassian.net/browse/EAI-2466
3 https://beekeeperai.atlassian.net/browse/EAI-2467
4 https://beekeeperai.atlassian.net/browse/EAI-2706
5 https://beekeeperai.atlassian.net/browse/EAI-2712

4 Validation criteria

Steal from Sudish's page and enhance explanation of validation JSON, including tying the validation back to an algo. https://beekeeperai.atlassian.net/wiki/spaces/BEEKEEPERA/pages/662667265/Algo+Models

5 Updates for 1:Many Validation

- Separate data spec/data attestation for each DS?

6 EscrowAI

EscrowAI's zero-trust confidential compute technology secures the collaboration between an algorithm owner's model and a data steward's protected data.

With EscrowAI, data remains in the data steward's secure environment, securely available in a hardware-based Trusted Execution Environment (TEE) instance for model analysis, training, and validation.

6.1 Data Steward and Algorithm Owner

A *data steward* (DS) is a generic data governance term for an organization charged with protecting by law to safeguard their clients' information while making it available to a third-party AI algorithm owner for analysis.

An *algorithm owner* (AO) is a generic data governance term for the creators of AI models to produce results for the data steward.

6.2 EscrowAI's Trusted Execution Environment (TEE)

Encrypted algorithms and encrypted data are brought into the TEE instance.

EscrowAI's TEE enforces the following essential aspects of confidential computing:

- **Data confidentiality:** Unauthorized entities cannot view data while it is in use in the TEE.
- **Data integrity:** Unauthorized entities cannot add, remove, or alter data while it is in use in the TEE.
- **Code integrity:** Unauthorized entities cannot add, remove, or alter code executing in the TEE.
- **Code confidentiality:** Unauthorized entities cannot view code while it is in use. code while it is in use in the TEE.

An EscrowAI TEE instance runs in the data steward's cloud, enabled by cutting edge confidential computing technology.

Encrypted algorithms are brought into the TEE along with the encrypted data. In the TEE instance's protected memory, the algorithms and data are then decrypted with user-created private keys stored in an HSM key vault. The computation is executed, and only a predetermined output is allowed out of the TEE instance after verification by EscrowAI. After the computation is complete, the TEE instance is decommissioned and terminated.

6.2.1 EscrowAI security advantages

EscrowAI has the following important security features:

- **Data Stewards retain control of their data.** The Data Stewards' organizations retain full control of the data. With EscrowAI, data stays in the Data Steward's organizations' infrastructure.
- **Strong encryption protects data at rest, in transit, and in use.**
  Data is encrypted with AES-256-based ciphers and all network connections use TLS.
- **Data is protected in use.**
  Data and algorithms achieve high-isolation and memory encryption through hardware-based

assurances. Secure attestation ensures the authenticity and integrity of the TEE execution environments.

- **Algorithm intellectual property is protected.**
  Algorithm owners retain protection of their intellectual property. Algorithm owners themselves encrypt their models, which are decrypted only in a TEE instance.

6.2.2 Confidential Computing technology options

EscrowAI offers the following types of confidential computing[6] technologies. These technologies underlie the virtual machines (VMs) and other secure resources in the TEE instance that EscrowAI manages. The TEE technology is assigned during the project creation process.

6.2.2.1 Confidential Containers in an Intel Software Guard Extensions (SGX) Enclave

EscrowAI offers Microsoft Azure's confidential containers that are based on Intel® Software Guard Extensions (Intel® SGX)[7]. SGX is a set of security-related instruction codes built into some Intel Central Processing Units (CPUs). On a hardware-based Trusted Execution Environment (TEE) instance, application code runs in private regions of memory, called *enclaves*[8], which are protected from all other processes running at higher privilege levels.

6.2.2.2 VM-isolated Confidential Containers on Azure Container Instances (CC ACI)

Confidential containers on Azure Container Instances are deployed in a container group with a Hyper-Visor isolated TEE, which includes a memory encryption key that is generated and managed by an AMD SEV-SNP capable processor. Data in use in memory is encrypted with this key to help provide protection against data replay, corruption, remapping, and aliasing-based attacks.

6.2.2.3 Confidential Virtual Machines (CVMs)

EscrowAI supports CVMs based on AMD Secure Encrypted Virtualization-Secure Nested Paging (SEV-SNP)[9] technology. A confidential virtual machine (CVM) provides a high level of security derived from a hardware-based TEE combined with the flexibility of operating within a full virtual machine at the operating system level. Confidential VMs provide significantly elevated protections for customer data from the underlying infrastructure and cloud operators, and BeeKeeperAI. Unlike other approaches and solutions, you don't have to adapt your existing workloads to fit the platform's technical needs. The CVM is isolated from other VMs, the hypervisor, and the host management code. CVM memory is encrypted by the CPU-managed keys stored within a dedicated virtual Trusted Platform Module (TPM) instance. The vTPM is a virtualized version of a hardware TPM that complies with the TPM 2.0 specification and runs in a secure environment outside the reach of any VM. The CVM boots only after successful attestation of the platform's critical components and security settings.

---

6 https://learn.microsoft.com/en-us/azure/confidential-computing/overview
7 https://www.intel.com/content/www/us/en/architecture-and-technology/software-guard-extensions.html
8 https://beekeeperai.atlassian.net/wiki/spaces/EVDUM/pages/694124566/Glossary#Compute-Enclave
9 https://www.amd.com/system/files/TechDocs/SEV-SNP-strengthening-vm-isolation-with-integrity-protection-and-more.pdf

6.3 Model training and validation in EscrowAI

In EscrowAI's secured model processing, an algorithm owner's model is run against a data steward's target dataset either for successive training of the model or to validate it.

An algorithm owner can run a model in two different modes.

- *Training mode*: Successive model runs against the target dataset can help refine the model. For any algorithm/dataset version combination, algorithm owners can train the model in repeated run requests.
  The algorithm owner receives the trained model after each run. This trained model can then be fed back into EscrowAI for more training as many times as desired or for validation.
- *Validation mode*: The algorithm owner can verify an already trained model's expected results against the validation criteria she associates with the model.

You can ask BeeKeeperAI for a project that has both training and validation or a project that is for validation only. Training is not supported for Intel SGX EnclaveOS confidential compute technology.

To alter the behavior of either training or validation runs, you can optionally define externalized runtime parameters that your algorithm reads during a run, without hard-coding the parameter values in your model. See Optional runtime parameters (see page 29).

6.4 How EscrowAI works

The collaboration summarized below between data stewards and algorithm owners is shown graphically in EscrowAI workflows (see page 45) and detailed step-by-step in Data steward step-by-step in EscrowAI (see page 64) and Algorithm owner step-by-step in EscrowAI (see page 83).

1. An algorithm owner uploads their encrypted algorithm to EscrowAI, specifying either training mode or validation mode. EscrowAI builds the algorithm into a secure computing container.
2. A data steward curates a data set to meet the algorithm owner's requirements. The data set is encrypted and uploaded to an EscrowAI accessible zone within their secure cloud.
3. The algorithm owner creates a run configuration for a unique combination of algorithm version and dataset version.
4. The algorithm owner sends a run request to the data steward, for as many runs as desired for that particular run configuration.
5. The data steward either initiates the run or rejects it.
6. To run the algorithm, EscrowAI initiates an attested, hardware-based Trusted Execution Environment (TEE) in the Data Steward cloud and loads the secure algorithm container and encrypted data into the TEE. In the attested enclave:
   - The algorithm and data are decrypted in protected memory.
   - The algorithm runs.

- A confidential report is created containing the algorithm's performance. For training mode, the trained model is also securely accessible to the algorithm owner for more training or for validation.
- The run outputs are the only things that leave that secure computing enclave.

7. The enclave is decommissioned, shut down, and deleted.

---

7 Onboard your project to EscrowAI

7.1 Accounts

Accounts are established by BeeKeeperAI Customer Service as the first post-contract work. Setting up the account involves:

- Creating your organization in EscrowAI.
- Creating users identified by your organization.

7.2 User Management

BeeKeeperAI will set up account users in EscrowAI at the direction of the account's point of contact. We can also configure Single Sign On (SSO) authentication to allow accounts direct access control for improved user convenience. EscrowAI supports common standards such as SAML and has out-of-the-box support for more than 15 cloud applications, including Microsoft Azure Active Directory.

7.3 Algorithm owner model setup

Before you add your model/algorithm to EscrowAI, there are several things you can to do to prepare.

7.3.1 Use EnclaveAPI to access secrets and post reports

The Enclave API is the interface the Algorithm Owner integrates to allow their algorithm to interact with data within an EscrowAI secured Trusted Execution Environment (TEE). There are several endpoints that are available inside an EscrowAI TEE that allow you to obtain data, list data, validate outputs, and post reports to EscrowAI.

7.3.1.1 Data Endpoints

The data endpoints provide you interfaces to securely interact with the Data Stewards encrypted data within the TEE. In the following subsections, you will find the details of some of the operations that can be performed with the data endpoints.

7.3.1.1.1 Data Listing [/api/v1/data/files]

This endpoint allows you to enumerate the data contained in a Data Steward's container in a blob storage.

7.3.1.1.1.1 Method

GET

7.3.1.1.1.2 Parameters

No parameters

7.3.1.1.1.3 Responses

Code	Description

200	File list retrieved successfully
	Media type
	application/json
	Controls `Accept` header.
	• Example Value
	• Schema
	<pre>{
	  "files": [
	  {
	  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
	  "name": "string",
	  "size": 0,
	  "uploadDate": "2023-12-08T21:30:58.830Z"
	  }
	  ],
	  "totalFiles": 1
	}</pre>

400	Bad Request (if applicable, e.g., invalid page or limit value)
500	Internal Server Error

```
import requests

SERVER_URL="https://localhost:5000"
ENDPOINT = "/api/v1/data/files"

def get_file_list():
    response = requests.get(f"{SERVER_URL}{ENDPOINT}", verify=False)  # verify=False
is to ignore SSL certificate validation

    if response.status_code == 200:
        data = response.json()
        print(json.dumps(data, indent=4))
        return data['files']
    else:
        # Handle potential errors based on status code
        if response.status_code == 400:
            print("Bad Request. Check your parameters.")
        elif response.status_code == 500:
            print("Internal Server Error on the API side.")
        else:
            print(f"Unexpected error with status code: {response.status_code}")
        return None
```

7.3.1.1.2 Data Downloading [/api/v1/data/file]

This endpoint allows you to download files by providing the “name” as the `filepath` variable.

7.3.1.1.2.1 Parameters

Name	Description
filepath *	This corresponds to the `name` key of each file in the files that are returned by the `/api/v1/data/files` endpoint.

7.3.1.1.2.2 Responses

Code	Description

200	File retrieved successfully Media type application/octet-stream Controls `Accept` header. • Example Value • Schema `string`
404	File not found

```python
import requests
from io import BytesIO

SERVER_URL="https://localhost:5000"
ENDPOINT = "/api/v1/data/file"

def download_file(file_name):
    response = requests.get(
        f"{SERVER_URL}{ENDPOINT}", params={"filepath": file_name}, verify=False
    )
    if response.status_code == 200:
        return response.content
    elif response.status_code == 404:
        print(f"File {file_name} not found")
        return None
    else:
        print(f"Unexpected error with status code: {response.status_code}")
        return None

file_name = "test_file.txt"
file_content = download_file(file_name)
# You can either use the file_content directly or convert it to a Binary I/O file
if file_content:
  data_io = BytesIO(file_content)
with open(data_io, "rb") as f:
  file_content = f.read() # get back to the binary file content that was returned
from the endpoint
```

7.3.1.2 Reporting Endpoints

The reporting endpoints are a collection of endpoints that allow you to post reports or logs to be displayed on the UI.

7.3.1.2.1 Report Validation [/api/v1/validate]

This endpoint validates the final report that is posted at the end of the training or validation flow. The endpoint validates the report and ensures that it matches a pre-defined schema provided by the Algorithm Owner and agreed upon by the Data Steward.

7.3.1.2.1.1 Parameters

No parameters

7.3.1.2.1.2 Request body

application/json

- Schema

```
{
  "name": "string",
  "status": "In Progress",
  "json_data": "string",
  "upload_location": "string",
  "project": "string",
  "run_configuration": "string",
  "run": "string"
}
```

7.3.1.2.1.3 Responses

Code	Description

200	Validation was successful Media type application/json Controls `Accept` header. • Example Value • Schema `{ "status": "successful", "message": { "additionalProp1": [ "string" ], "additionalProp2": [ "string" ], "additionalProp3": [ "string" ] } }`
400	Bad request or validation failed Media type application/json • Schema `{ "status": "failed", "message": { "additionalProp1": [ "string" ], "additionalProp2": [ "string" ], "additionalProp3": [ "string" ] } }`

Python Reference Implementation

```python
import requests
import json
import base64

SERVER_URL="https://localhost:5000"
ENDPOINT="/api/v1/validate"

def validate_report(finalReport):
  """
  Post a report to the server for validation.
  :param finalReport: The final report as a JSON object.
  :return: The server response as a JSON object.
  """

  # define the api endpoint
  url = f"{SERVER_URL}{ENDPOINT}"

  # send the post request
  response = requests.post(url, json=finalReport, verify=False)

  # check the response
  if response.status_code == 200:
      print('Validation was successful')
      print(response.json())
  else:
      print('Bad request or validation failed')
      print(response.json())

  return response.json()

class_rep = "Any object can be here... can be text, can be dict, etc."
finalReport = {
  "json_data": base64.b64encode(json.dumps(class_rep).encode()).decode(),
  "name": "Sklearn Model",
  "project": os.environ.get('ESCROW_PROJECT_ID'),
  "run": os.environ.get('ESCROW_RUN_ID'),
  "run_configuration": os.environ.get('ESCROW_RUN_CONFIG_ID'),
  "status": "Completed",
  "upload_location": "your_upload_location"
}
validate_report(finalReport)
```

Sample Response

```
{
  "status": "successful",
  "message": {
    "additionalProp1": [
      "string"
    ],
```

```
    "additionalProp2": [
      "string"
    ],
    "additionalProp3": [
      "string"
    ]
  }
}
```

7.3.1.2.2 Report Posting [/api/v1/report]

This endpoints allows you to post a log or report to the UI. When a log/report is posted with the status, “Completed”, then the log/report is validated against a predefined schema as described in Report Validation subsection.

7.3.1.2.2.1 Parameters

No parameters

7.3.1.2.2.2 Request body

application/json

- Schema

```
{
  "name": "string",
  "status": "In Progress",
  "json_data": "string",
  "upload_location": "string",
  "project": "string",
  "run_configuration": "string",
  "run": "string"
}
```

7.3.1.2.2.3 Responses

Code	Description

200	Report posted successfully Media type application/json Controls `Accept` header. • Example Value • Schema `{ "message": "string" }`

Python Reference Implementation

```
import requests
import json
import base64

SERVER_URL="https://localhost:5000"
ENDPOINT="/api/v1/report"

def post_report(report):
  """
  Post a report to the server for submission.
  :param report: The final report as a JSON object.
  :return: The server response as a JSON object.
  """
  # define the api endpoint
  url = f"{SERVER_URL}{ENDPOINT}"

  # send the post request
  response = requests.post(url, json=report, verify=False)

  # check the response
  if response.status_code == 200:
      print('Validation was successful')
      print(response.json())
  else:
      print('Bad request or validation failed')
      print(response.json())


  return response.json()

content_to_be_posted = "Any object can be here... can be text, can be dict, etc."
report = {
  "json_data": base64.b64encode(json.dumps(content_to_be_posted).encode()).decode(),
```

```
  "name": "Sklearn Model",
  "project": os.environ.get('ESCROW_PROJECT_ID'),
  "run": os.environ.get('ESCROW_RUN_ID'),
  "run_configuration": os.environ.get('ESCROW_RUN_CONFIG_ID'),
  "status": "In Progress",
  "upload_location": "Not in use currently. can be anything"
}
post_report(report)
```

Sample Response

```
{
  "message": "string"
}
```

7.3.2 Optional runtime parameters

With EscrowAI's optional runtime parameters, you can design your model to accept externalized variable inputs when it is run. You yourself design the parameters you want and the algorithm logic that operates with those parameters. You have complete control over what parameters you might want to use.

To affect the behavior of your algorithm, you change the values of the parameter for various run requests without having to hardcode the values directly in your algorithm.

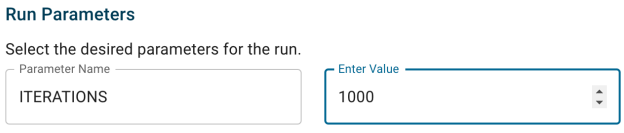
Runtime parameters can be used for training runs, validation runs, or any other need.

When you send a run request that takes advantage of EscrowAI's runtime parameters, you specify the names and values of those parameters.

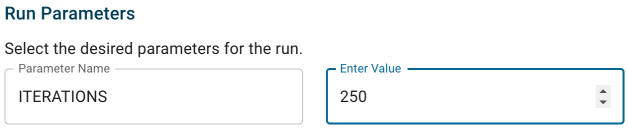
Runtime parameters are available as environment variables on the running enclave.

For example, suppose you have the parameter `ITERATIONS` that your algorithm relies on to indicate the number of times your algorithm should cycle through the target dataset. For instance, Training Model v1 needs to run 1,000 times, Training Model v2 should run 250 times, and Training Model v3 should run 5 times.

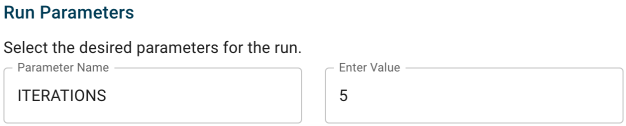
1. When you create a run request for Training Model v1, you specify the parameter `ITERATIONS` with value 1000.



2. When you create a run request for Training Model v2, you specify `ITERATIONS` with value 250.



3. When you create a run request for Training Model v3, you specify `ITERATIONS` with value 5.



In each run, your algorithm gets the environment variable's value and uses the same logic to vary its behavior based on the value of the externalized parameter, as in this simple code snippet.

```
.
.
.
# Import Python os module to get
# environment variable ITERATIONS
import os
#
# Incude scikit-learn module Py-PI package https://pypi.org/
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression

# Get value of environment variable ITERATIONS
# that we set as runtime parameter
# on EscrowAI's "Send Run Request"
iterations = os.environ.get['ITERATIONS']

# Run the algorithm
X, y = load_iris(return_X_y=True)
clf = LogisticRegression(random_state=0, max_iter=iterations).fit(X, y)
clf.predict(X[:2, :])
clf.predict_proba(X[:2, :])
clf.score(X, y)
.
.
.
```

7.3.3 Work with MLflow in EscrowAI

7.3.3.1 EscrowAI and MLFlow Integration

what is the level of integration?

7.3.3.2 How to Get Started with MLFlow

7.3.3.3 Create your MLFlow model

7.3.3.4 Logging your experiment with MLFlow

- what to integrate into your algo code.

7.3.3.5 Review experiment run

- MLFlow tracking server information
- relate to the section on logging experiment
- tie back to the core MLFlow documentation
- explicit doc on how I do multiple successive training runs

7.3.4 Examples of models for use with EscrowAI

[@Christer Smith this is the topic where we will put the examples of various models that you're cranking on]

[Xref to Model building test cases (see page 101)]

7.3.5 Package your algorithm for upload to EscrowAI

In EscrowAI, an algorithm is your program (such as data query, inference, or model training) that processes data in an EscrowAI-managed TEE running in the Data Steward's environment.

For proper execution, your algorithm code must be packaged according to the guidelines detailed here. The package is converted to a container by EscrowAI.

These details include sample code you can use as a template for your algorithm package and how to create the algorithm package.

After you create your algorithm package, you will encrypt[10] and upload[11] it to EscrowAI.

7.3.5.1 Contents of the algorithm package

The package has the following components:

- `Dockerfile` - A set of instructions to define your algorithm's container environment.
- Entry point: `run.sh` - A file that defines the starting point of your code.

10 https://beekeeperai.atlassian.net/wiki/spaces/EVDUM/pages/694288589/Encrypt+algorithm+files
11 https://beekeeperai.atlassian.net/wiki/spaces/EVDUM/pages/754188296/Upload+the+algorithm

- Algorithm code - Your code.

7.3.5.1.1 Dockerfile

Dockerfile is a set of instructions to define your algorithm's container environment.

See the example Dockerfile

```
# Define a base image you want to build FROM
FROM python:3.9.16-slim

# Set the working directory
WORKDIR /app

# Use COPY to keep your local folder and file structure
COPY . .

# Write reuirements.txt
to resolve required libraries to run the model
...

RUN pip install EnclaveSDK

# Make the run.sh script executable
RUN chmod +x run.sh

# Execute the template script inside a convenient wrapper
ENTRYPOINT ["run.sh"]
```

7.3.5.1.2 Entry point: example run.sh

When your container starts, it executes an entry point script that actually starts your algorithm program. The example entry point here is called `run.sh`.

The script name `run.sh` is only an example. You can name your entry point script whatever you like, as long as that same script name is specified for the value of the `ENTRYPOINT` variable in your Dockerfile.

See the example run.sh

```
#!/bin/sh

# Run your algorithm code
python3 algo-template.py
```

7.3.5.1.3 Example algorithm code

The example `algo-template.py` below calls the REST-API-based EscrowAI Enclave API for the following tasks. These API endpoints are discussed in Use EnclaveAPI to access secrets and post reports (see page 20).

1. List available data files.
2. Fetch data files.
3. Post report.

See the example algorithm.py

```
import EnclaveSDK

configuration = EnclaveSDK.Configuration("https://localhost:5000")
api_client = EnclaveSDK.ApiClient(configuration)

def main():
    """Main function demonstrating how to use EnclaveSDK"""

    ####
    # 1. List available data files
    ###
    # Create an instance of Data API class and list files in blob storage
    api_instance = EnclaveSDK.DataApi(api_client)
    api_response = api_instance.api_v1_data_files_get()

    ###
    # 2. Fetch data files
    ###
    for file in api_response.files:
      api_response = api_instance.api_v1_data_file_get(file.name)
      break

    ###
    # 3. Post report
    ###
    # Create an instance of Report API class
    api_instance = EnclaveSDK.ReportApi(api_client)
    report = {"report": "Performance Report"}
    finalReport = {"json_data": report, "name": "EscrowAI Algorithm Package",
"status": "Completed"}
    report = EnclaveSDK.Report.from_dict(finalReport)
    api_response = api_instance.api_v1_report_post(report)

if __name__ == "__main__":
    main()
```

7.3.5.2 Folder structure for algorithm package

Put your files at the top level of a folder, as in these examples:

- `Dockerfile`
- `run.sh`
- `algo-template.py`

The structure of your folder should look like this example `algorithm-package` folder:

```
algorithm-package/
├── Dockerfile
├── run.sh
└── algo-template.py
# other files or subdirectories
```

Your actual algorithm package must also contain all of the files or subdirectories needed to successfully run your algorithm

7.3.5.3 Encrypt and upload algorithm package

After you create your algorithm package, encrypt[12] and upload[13] it to EscrowAI

7.3.6 EscrowAI multiple concurrent algorithm runs

The algorithm owner can request multiple runs but submit only one request at a time for each individual run configuration. EscrowAI processes multiple runs concurrently, each run operating on its unique combination of dataset version and algorithm version.

Example

As algorithm owner, you might have three different versions of an algorithm that you want to test against a certain dataset version. You create three different run configurations, each with the same dataset version but with the unique algorithm versions.

dataset version 1	algorithm version 1	Unique run configuration #1
	algorithm version 2	Unique run configuration #2

12 https://beekeeperai.atlassian.net/wiki/spaces/EVDUM/pages/694288589/Encrypt+algorithm+files
13 https://beekeeperai.atlassian.net/wiki/spaces/EVDUM/pages/754188296/Upload+the+algorithm

	algorithm version 3	Unique run configuration #3

You then submit run requests for all three unique run configurations, which EscrowAI processes concurrently after the data steward initiates the runs.

Continuing the example, after a request to start run configuration #1, you need to wait until that run finishes before you can request a new run for it. Likewise, the data steward can initiate the new run at the algorithm owner's request, but only after that run configuration's previous run has completed.

7.4 Data Steward Azure setup

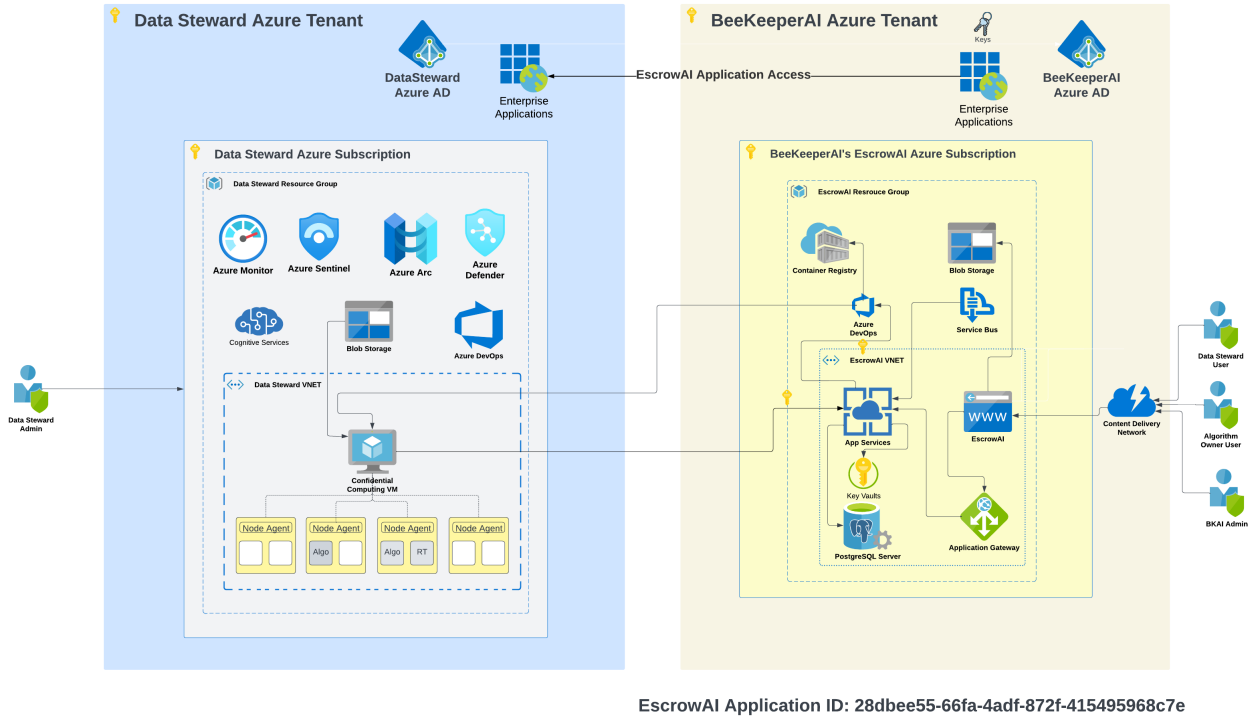
Before working in EscrowAI, there are prerequisites you need to do to prepare.

> Before BeeKeeperAI can create an EscrowAI project for you and the algorithm owner, you need to send Azure VNET details to BeeKeeperAI so that enclaves can run in your own Azure environment. See Send VNET details to BeeKeeperAI (see page 35).

7.4.1 Azure Cloud

A Data Steward (DS) must have the required Azure infrastructure to support data access and deployment of confidential compute nodes within their hardened Azure cloud environment.

The configuration in the Data Steward Azure Subscription, shown below, represents the minimum amount of infrastructure required to operate. Institutions may have additional policies to layer on this configuration such as resources related to VPN and on-premises access to this content.



1 EscrowAI Architecture

7.4.2 Data Steward cloud services provisioning workflow

> You will need an Azure account with an active subscription to complete these steps.

> The following steps are for the DS to prepare the Azure environment for EscrowAI. The steps below are provided for setting up the needed services using the Azure Portal.

The following instructions describe a one-time provisioning of Azure services in the Data Steward's Azure subscription that is required by EscrowAI. These Azure services are needed to allow EscrowAI to provision and to decommission a Trusted Execution Environment in the Data Steward's Azure subscription automatically (including confidential virtual machines and confidential container groups).

Summary of data steward's one-time setup work in Azure:

- Create a resource group for your Confidential Computing VMs.
- Provision a VNET.
- Import EscrowAI application into Azure.

7.4.3 Create a Virtual Network

1. Sign in to the Azure portal.
2. Search for and select Virtual networks.
3. On the Virtual networks page, select **Create**.
4. On the **Basics** tab of the Create virtual network screen, enter or select the following information:
   • **Subscription:** Keep the default or select a different subscription.
   • **Resource group:** Select Create new, and then name the resource group <resource group name>.
   • **Virtual network name:** Enter the desried VNET name.
   • **Region:** Keep the default or select a different region for the network and all its resources.
5. Select **Next: IP Addresses** at the bottom of the page.
6. On the **IP Addresses** tab, under **IPv4 address space**, select the garbage can icon to remove any address space that already appears, and then enter a VNET address. For example `10.0.0.0/16`.
7. Select **Add subnet**.
8. On the **Add subnet** screen, enter the following information, and then select **Add**:
   • **Subnet name**
   • **Subnet address range:** <subnet address>. For example `10.0.0.0/24`.
9. **The following additional steps in this sub-section are required to enable Confidential Containers on Azure Container Instances (ACI):**
   a. An additional subnet must be created by following the steps in 8
   b. Delegate this subnet to the Container Instance/Container Group

7.4.3.1 Send VNET details to BeeKeeperAI

Share the following values with the BeekeeperAI team.

1. VNET Name
2. VNET Resource Group Name
3. For **Virtual Machines** Subnet Name created in step 8
4. For **Azure Confidential Container Instances (ACI)** Subnet Name created in step 9
5. Resource Group Name for VM
6. Region

7.4.4 Import EscrowAI application into Azure

EscrowAI relies on an application that you securely import into your Azure environment so that EscrowAI can launch confidential containers for you and take care of other housekeeping.

The general sequence of steps in Azure is as follows:

1. Import the EscrowAI application.
2. Create a custom role to assign to the EscrowAI application.
3. Set the permissions of the custom role for the EscrowAI application.
4. Apply the custom role to the EscrowAI application.

7.4.4.1 Prerequisites

Have these prerequisites ready or make note of them. You will need them in the steps that follow.

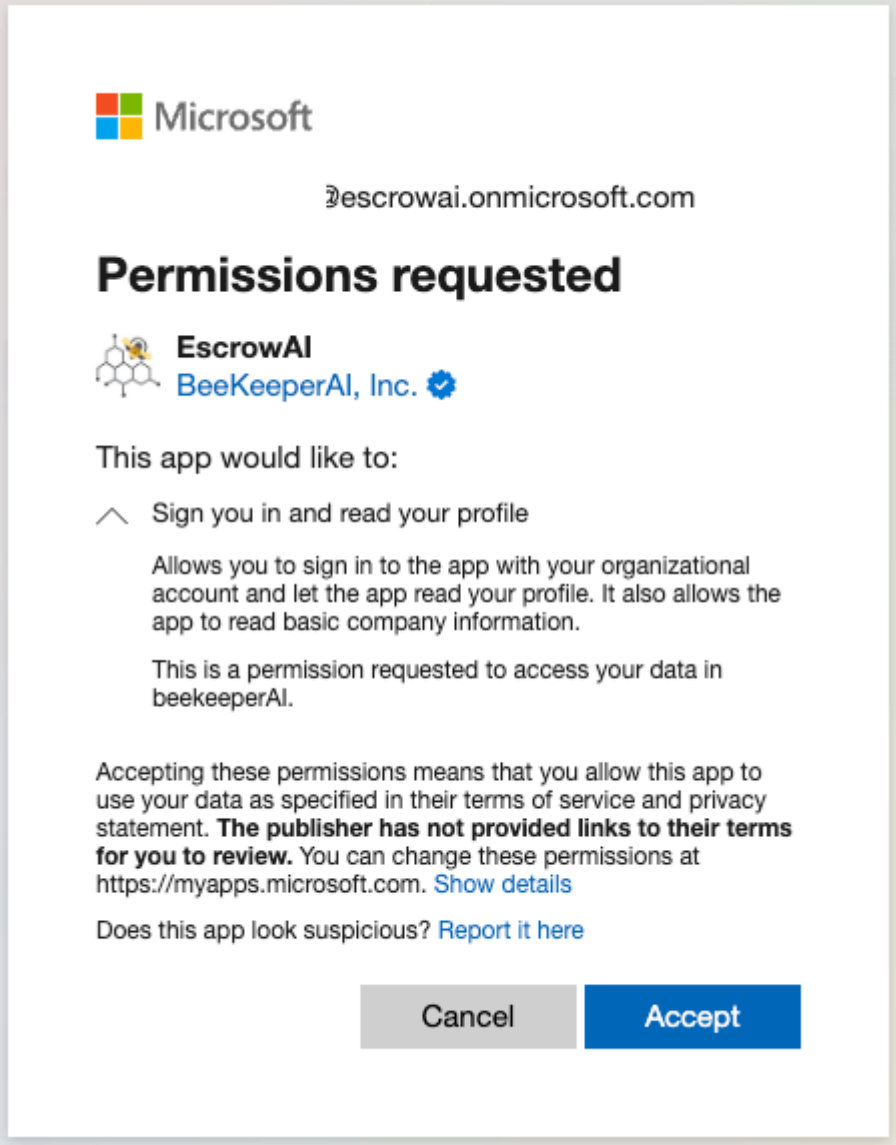
- You must be an Azure administrator to import the EscrowAI Enterprise Application
- You need to know your Azure tenant ID. This value is indicated in these steps as `<your_azure_tenant_id>`.
- You need to know your Azure subscription ID. This value is indicated in these steps as `<your_azure_subscription_id>`.
- URL to import EscrowAI application into Azure. The following URL is the EscrowAI application for importing into Azure. Notice the `<your_azure_tenant_id>` variable you must specify in your browser:
- ```
  https://login.microsoftonline.com/<your_azure_tenant_id>/oauth2/authorize?client_id=28dbee55-66fa-4adf-872f-415495968c7e&response_type=code&redirect_uri=https%3A%2F%2Fwww.microsoft.com%2F
  ```
- JSON to assign permissions to EscrowAI application.
  Create a JSON file with the following content, which defines permissions you will assign to the EscrowAI application.
  - On line 3, do not change the value of `roleName`. Leave it as `escrow-vm-launch`. You will assign this role to the EscrowAI application.
  - On line 6, be sure to specify your Azure subscription ID for the variable `<your_azure_subscription_id>`.
  ```

```
{
    "properties": {
        "roleName": "escrow-vm-launch",
        "description": "Escrow VM Launch Custom Role",
        "assignableScopes": [
            "/subscriptions/<your_azure_subscription_id>"
        ],
        "permissions": [
            {
                "actions": [
                    "Microsoft.Network/virtualNetworks/subnets/read",
                    "Microsoft.Network/networkSecurityGroups/read",
                    "Microsoft.Network/networkSecurityGroups/write",
                    "Microsoft.Network/networkSecurityGroups/delete",
                    "Microsoft.ContainerInstance/containerGroups/*",
                    "Microsoft.Resources/subscriptions/resourceGroups/read",
                    "Microsoft.Resources/subscriptions/resourceGroups/write",
                    "Microsoft.Resources/subscriptions/resourceGroups/delete",
                    "Microsoft.Resources/deployments/*"
                ],
                "notActions": [],
                "dataActions": [],
                "notDataActions": []
            }
        ]
    }
}
```

7.4.4.2 Steps

7.4.4.2.1 Import the EscrowAI application into Azure

1. As an Azure administrator, in your web browser, open the URL shown in URL to import EscrowAI application into Azure (see page 38).
   On the URL, be sure to specify your own Azure tenant ID for the variable `<your_azure_tenant_id>`.
2. At the prompt from Microsoft, check **Consent on behalf of your organization**, and accept the requested permissions.



Result: The EscrowAI application has been imported into Azure.

7.4.4.2.2 Create a custom Azure role, add permissions, and assign to the EscrowAI application

1. In Azure, go to the appropriate **Subscription.**
2. Under **Access Control (IAM),** click **+Add** in the top menu, then select **Add custom role**.

Add permissions from JSON file

1. On the **Basics** tab, in **Baseline permissions**, select **Start from JSON**.
2. Open and select the contents of JSON file you created in JSON to assign permissions to EscrowAI application (see page 38).

3. Copy the above JSON content and paste it to overwrite the stub displayed in the JSON box, substituting your own Azure subscription ID in place of the variable `<your_azure_subscription_id>`.
    a. Click Save
    b. Click **Review and Create**.
    c. Click **Create**.
4. Assign the Virtual Machine Contributor role (Built-In Role) to the EscrowAI application.
    a. Navigate to the appropriate **Subscription.**
    b. Under **Access control (IAM)** click **+Add** in the top menu, then select **Add role assignment**.
    c. On the **Role** tab, under **Job Function roles**, select the **Virtual Machine Contributor** role.
    d. On the **Members** tab, select **+Select members,** select the **EscrowAI application,** then click **Review + Assign**.

Assign custom role to EscrowAI application

1. Navigate to the appropriate **Subscription.**
2. Under **Access control (IAM)** click **+Add** in the top menu, then select **Add role assignment**.
3. On the **Role** tab, under **Job Function roles**, select the custom role , which is named `escrow-vm-launch`**.**
4. On the **Members** tab, select **+Select members,** select the **EscrowAI application,** then click **Review + Assign**.

Result: The EscrowAI application has been given a role with the appropriate permissions to do its function for you.

7.4.5 Create a Storage Account

1. Sign in to the Azure portal.
2. In the Azure portal, click **Create a resource**.
3. Search for **Storage account** in the search bar and select it from the list of options.
4. Click **Create** to start creating a new storage account.
5. In the **Basics** tab, select your subscription, create a new resource group or select an existing one, and give your storage account a unique name.
6. Select the region for the storage account.
7. Choose the performance tier and Redundancy.
8. Click **Next: Advanced** to move to the next tab.
9. Choose your desired settings for advanced options like virtual networks, data protection, and data transfer.
10. Click **Review + create** to review your storage account settings.
11. After reviewing your settings, click **Create** to create your storage account.

7.4.6 Set up and manage Blob Storage

The setup and management of Blob storage is repeated for each project. A project specific Container is created, and encrypted project data is uploaded as a blob into this Container. The workflows in these instructions assume the Storage Account and project Container are configured at the start of any project.

See Upload encrypted dataset to Azure Blob container[14].

It is recommended that the data site designate a role or individual that creates the necessary project containers and who can grant permission to the project-specific data analyst 1) for uploading the encrypted data set to blob storage and 2) for creating the SAS URL.

Please consider creating a private endpoint for Blob Storage, which enables traffic to flow between the VNET and Blob Storage via a private network.

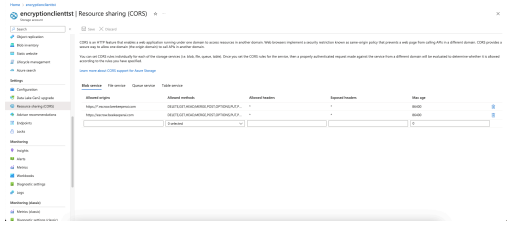
Generate a SAS URL for the Azure Storage Container

In order to upload to an Azure Blob Container, you'll need the container's Shared Access Signature (SAS) URL[15]. To generate a valid SAS URL follow the instructions below:

Update Storage Account CORS Settings

First, you will need to update the CORS settings of your Azure Storage Account[16] with these origins:

1. *https://*.escrow.beekeeperai.com*
2. *https://escrow.beekeeperai.com*
3. To do this, visit the Azure Portal[17] and select your storage account, navigate to the **Settings** section, and select **CORS**.



14 https://beekeeperai.atlassian.net/wiki/spaces/EVDUM/pages/696287238/Upload+encrypted+dataset+to+Azure+Blob+container

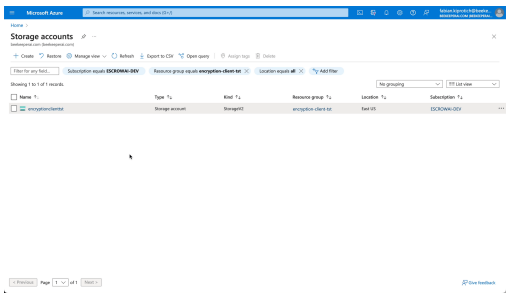
15 https://learn.microsoft.com/en-us/azure/applied-ai-services/form-recognizer/create-sas-tokens?view=form-recog-3.0.0

16 https://learn.microsoft.com/en-us/rest/api/storageservices/cross-origin-resource-sharing--cors--support-for-the-azure-storage-services

17 https://portal.azure.com/

Generate SAS URL

1. Navigate to the **Containers** section of your storage account.
2. From the containers table, click on the settings icon of your container and select **Generate SAS** from the menu.
3. Set the relevant permissions of the SAS URL and click the “Generate SAS token and URL” button.
4. You can then copy the Blob SAS URL for use in the encryption client.



7.4.7 Related Microsoft documentation

- Create a Linux virtual machine in the Azure portal[18]
- Manage resource groups in the Azure portal[19]
- Use the Azure portal to create a virtual network[20]
- Create a Windows VM in the Azure portal[21]
- Create a storage account[22]
- Upload, download, and list blobs in the Azure portal[23]
- Use a managed identity to access Azure Storage - Linux[24]
- Configure Azure Storage Firewalls and Virtual Networks[25]
- Private Endpoint for Azure Storage[26]

---

18 https://learn.microsoft.com/en-us/azure/virtual-machines/linux/quick-create-portal?tabs=ubuntu
19 https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/manage-resource-groups-portal
20 https://docs.microsoft.com/en-us/azure/virtual-network/quick-create-portal
21 https://docs.microsoft.com/en-us/azure/virtual-machines/windows/quick-create-portal
22 https://docs.microsoft.com/en-us/azure/storage/common/storage-account-create?tabs=azure-portal
23 https://docs.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-portal
24 https://docs.microsoft.com/en-us/azure/active-directory/managed-identities-azure-resources/tutorial-linux-vm-access-storage
25 https://docs.microsoft.com/en-us/azure/storage/common/storage-network-security?tabs=azure-portal
26 https://learn.microsoft.com/en-us/azure/storage/common/storage-private-endpoints

8 Project creation by BeeKeeperAI

BeeKeeperAI creates your EscrowAI projects.

The project:

- Brings together the data set(s) and algorithm(s).
- Assigns roles to each organization (Algorithm Owner or Data Steward).
- Adds designated users from each organization to the project.
- Assigns the desired Confidential Computing technology platform to the project.
  After a project is created, its Confidential Computing technology cannot be changed. If you want a different technology a new project must be created.

The new project appears in the Home Page of the assigned user, with the name given by the project owner and with placeholders for the project artifacts.

8.1 Data Steward VNET details for project creation

Data steward needs to give details about a VNET needed in Azure. See Data Steward Azure setup (see page 35).

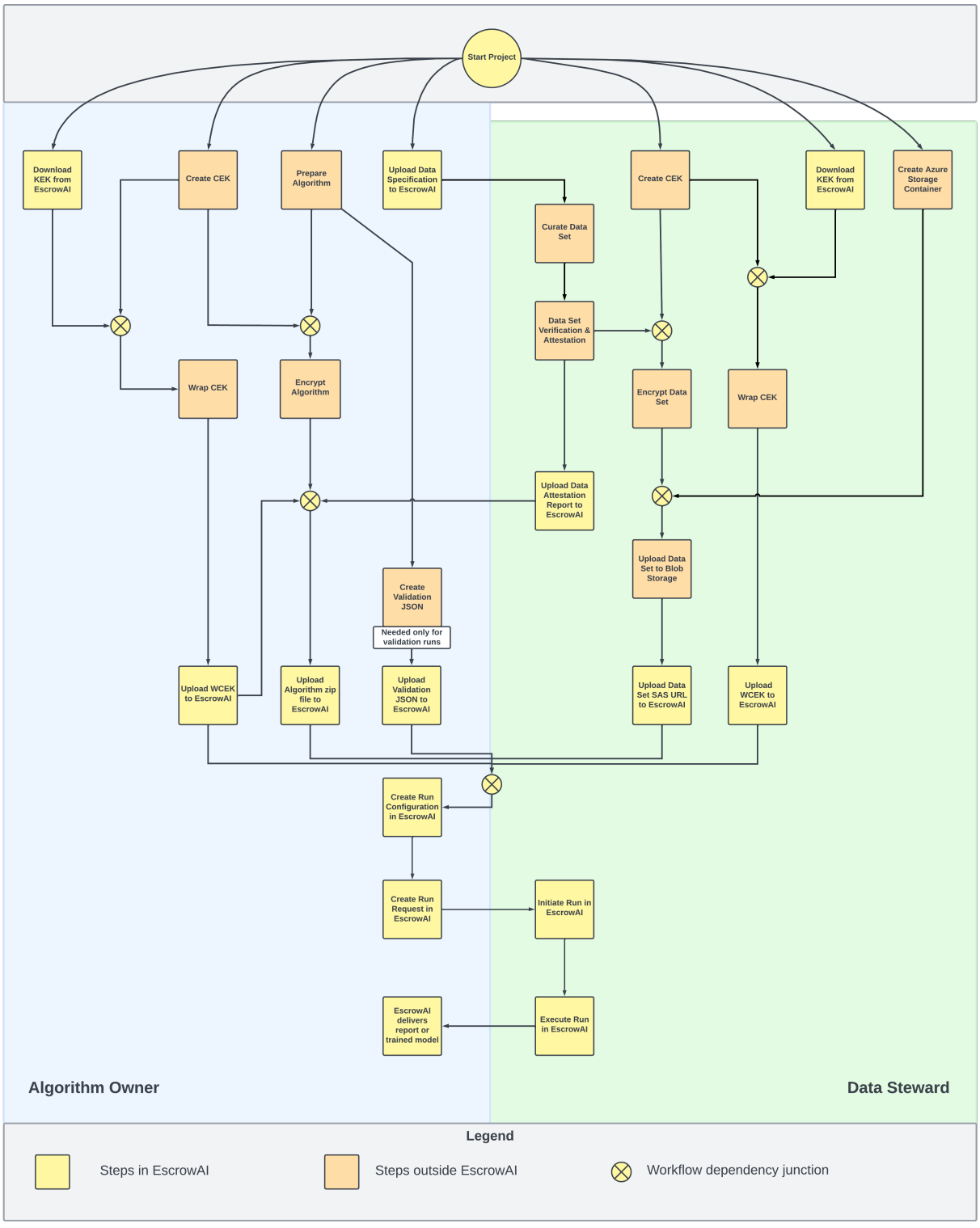
---

9 EscrowAI workflows

9.1 Collaboration workflow

The following diagram illustrates the collaboration in an EscrowAI project.

Most tasks in the project workflow are completed in EscrowAI. Preparation of project artifacts to upload to EscrowAI takes place on your own local computers. In this workflow diagram the differentiation between the in- and out-of-EscrowAI application steps are color coded. The diagram also shows dependencies between workflow steps using a workflow dependency junction. All of the workflow steps connected into the junction must be completed before the step after the junction can be completed.



2 EscrowAI collaboration workflow between algorithm owner and data steward

9.2 Common encryption steps

Data stewards and algorithm owners both independently use EscrowAI's encryption tool complete the cryptography workflow.

- EscrowAI cryptography (see page 58)
- Common encryption steps (see page 61)

---

10 EscrowAI dashboard

10.1 Home page

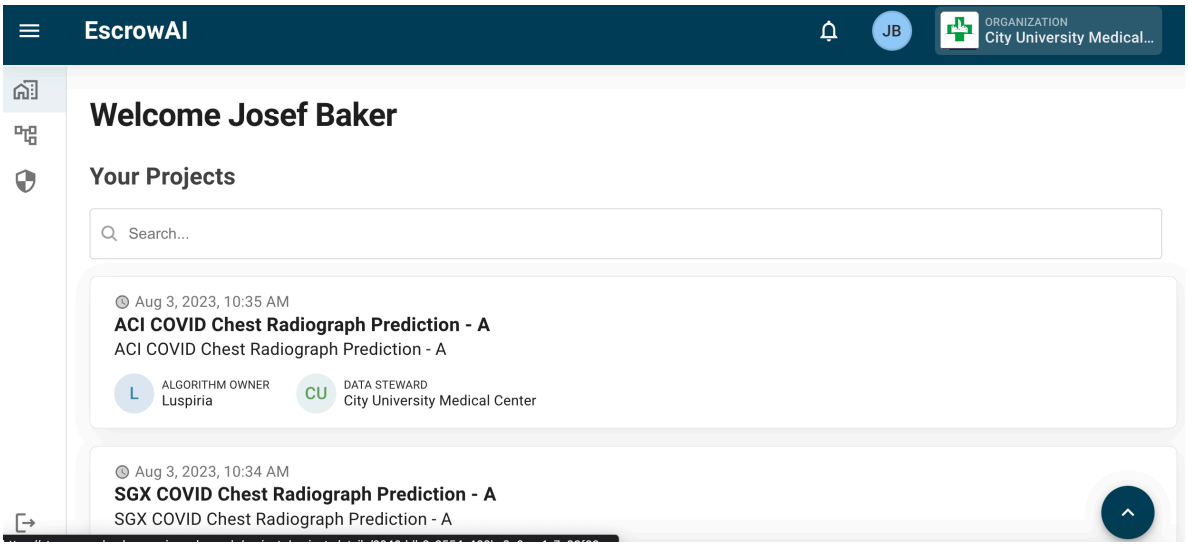
The EscrowAI home page is the user's initial landing page after logging in. It displays the list of Projects that have been assigned to the user by their organization.

The user's organization name and logo and the user's initials icon are displayed in the page heading banner.

The home page displays the list of Projects as cards with the most recent projects at the top. Each Project card includes descriptive metadata:

- Project name
- Project description
- Project creation date
- Project collaborator organizations and roles
- User initials icons

Clicking on a Project card will open the EscrowAI project page (see page 52) with the details of the collaboration.



10.2 Project metadata search

You can search for content within projects from the home page using keywords. The search query returns a content list based on the metadata in the following fields.

- Project names and descriptions.
- Dataset names, descriptions, version names, and version descriptions.
- Algorithm names, descriptions, version tags.
- Project artifacts names, descriptions, version tags, and version descriptions.

- Run configuration names and descriptions.

10.3 Home page elements

Element	Description
	System Notification icon (also called the “bell icon”). System notifications related to runs are displayed by clicking this icon. New notifications are indicated by a red dot on the icon.
	User initials icon. This element is used to identify the user to other collaborators throughout EscrowAI.
	Company logo. This is specific to the user that is logged in.
	Expand bar. Clicking this icon will display the Home, Project, and Log Out icons.
	Home icon. This navigates to the user’s home page.
	Project icon. This navigates to the user’s project listing.
	EscrowAI encryption tool for encrypting datasets and algorithms. For details, see EscrowAI encryption tool.

Element	Description
	On the lower right of the EscrowAI home page, the quick actions icon has several uses: • A link to EscrowAI help and other information resources. For details, see EscrowAI Help Center (see page 110). • Chat between the project's data stewards and algorithm owners. For details, see Intra-project chat between users (see page 54).
	Log out
App Version	EscrowAI release version number

10.4 High-level project states

An EscrowAI project is in one of the following states. On the EscrowAI home page, the project state is shown the upper right of each project card. The states are a direct reflection of the progress of the project as recorded by EscrowAI.

- **Active**: project's workflow is actively in progress by the data steward and algorithm owner, as described in EscrowAI workflows (see page 45).
- **Paused**: project workflow activity is suspended.
- **Deactivated**: project has been retired.

In any of these states, the project metadata is still searchable.

10.5 Project event notifications via the bell icon

On your EscrowAI home page, the bell icon near the upper right lists notifications about project-related events.

- Unread notifications are indicated by a dot above the bell.
- Notifications are removed from the bell after 48 hours.

10.5.1 Types of events

Bell notifications include the following:

- **Run requested**: Event recorded for the data steward: a run has been requested by the algorithm owner
- **Run request rejected**: Event recorded for algorithm owner: the data steward has rejected a run request. This event is displayed in EscrowAI for 48 hours to allow the algorithm owner to see it.

10.5.2 View notifications

To view event notifications:

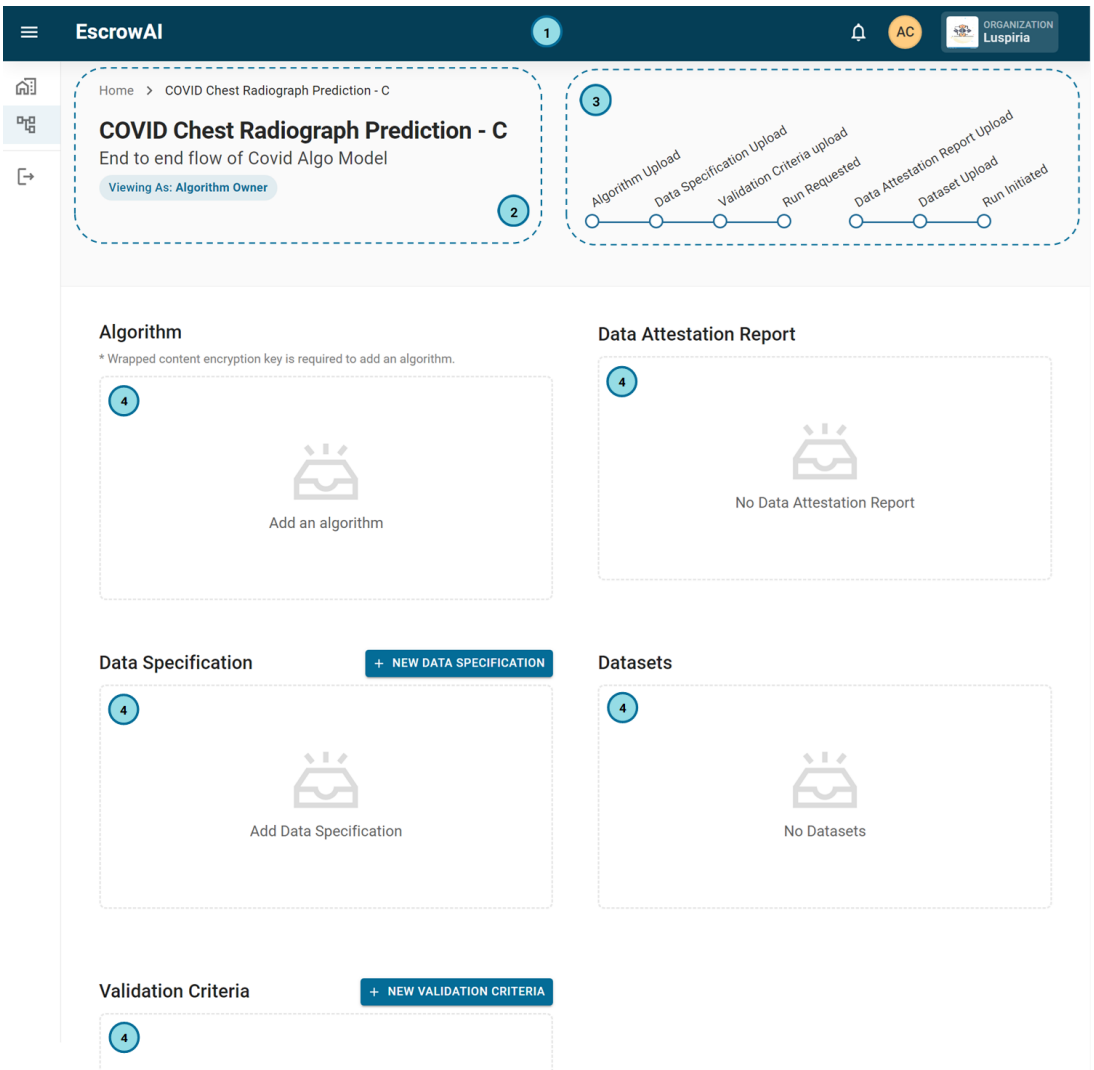
1. Go to your EscrowAI home page.
2. Near the upper right, click the bell.
3. View the flagged events.

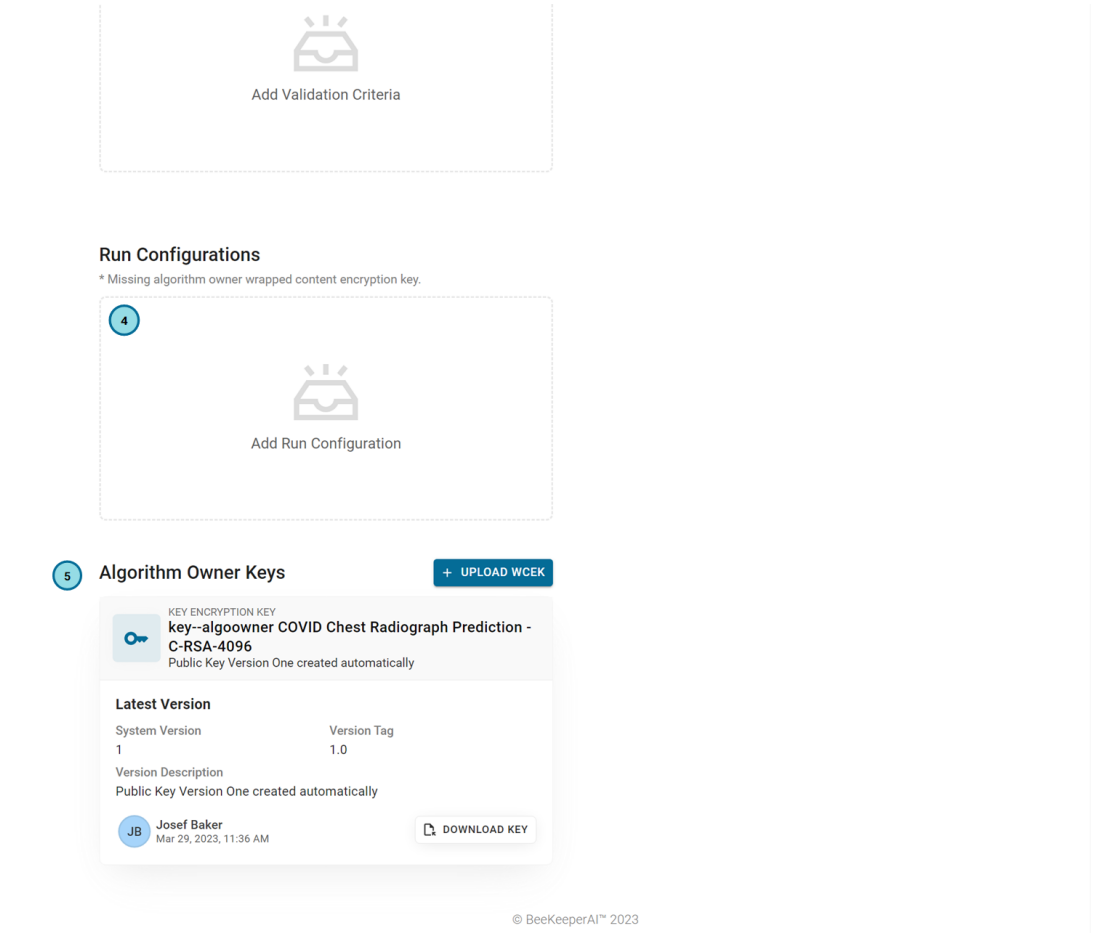
---

11 EscrowAI projects

The EscrowAI Project Page is a central hub for all information related to your project. It provides an overview of the project's goals, progress, and key milestones, including the project name, a brief description, and relevant files. The workflow tracker, located on the upper right-hand side of the page, displays the progress of completed and incomplete tasks. In the center of the page, project cards show a title and the completion status of each step (e.g., 'Data Specification Uploaded,' 'Algorithm Added,' etc.). The Project Page is a critical resource for the collaboration of the Data Steward and Algorithm Owner to stay informed and on track toward successful completion. Permission to access or change artifacts is limited by role. For instance, the Algorithm Owner can add and change the algorithm, but the Data Steward access is limited to the information in the artifact card.

11.1 Project page elements





1	EscrowAI Title Bar Contains the active user's initials, the organization's identification, and the notification icon.
2	Project identification section. Contains the project name, description, user's role, and navigation breadcrumbs.
3	Project workflow milestones. As project progresses, the milestones are color-coded: • **Green**: milestone achieved. • **Yellow**: milestone is pending completion. • **Red**: milestone attempted but not achieved.

4	Project artifact cards. These hold the named artifact, meta data, and version information.
5	Cryptographic key card. Keys for decrypting the data or algorithm content are uploaded here.

11.2 Intra-project chat between users

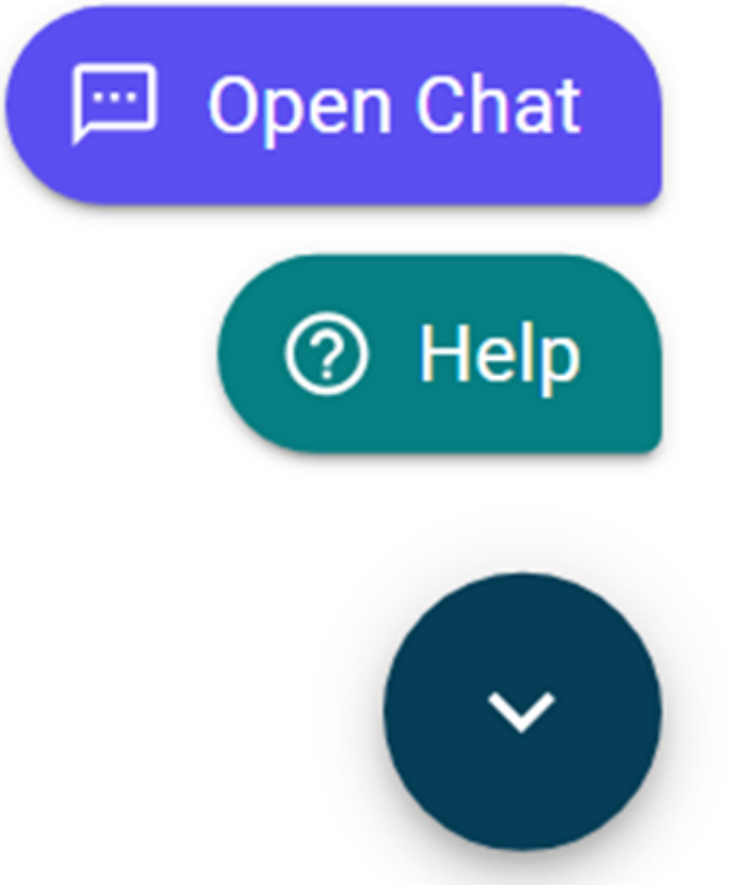
To enhance your collaboration, you can communicate with your project counterparts directly from within EscrowAI. Chat is limited to intra-project communications.

To start or view a chat:

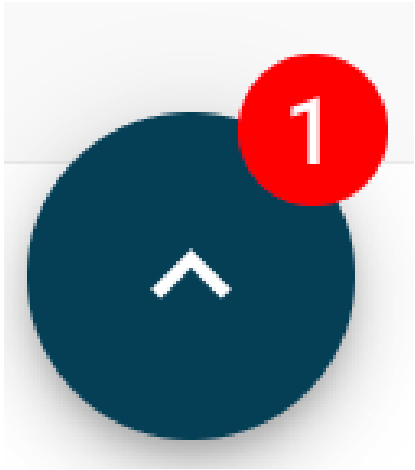
1. Go to the desired project page.
2. The chat widget is initiated by clicking on the **Quick actions** icon in the lower right corner.
3. Select **Open Chat**.

> A red bubble with number indicates that there are unread messages in Chat.

3 Quick actions icon



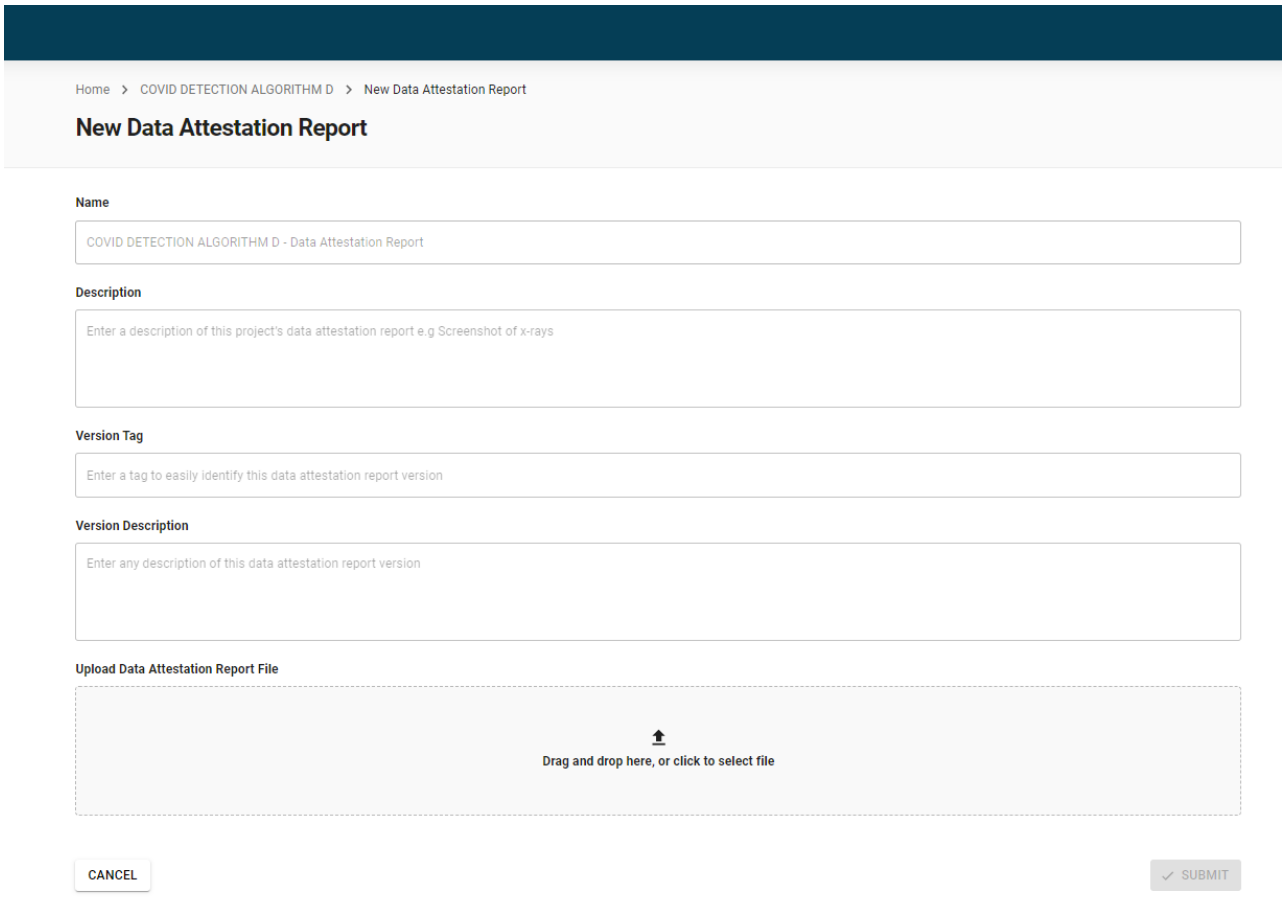
4 Expanded action menu



5 One unread message

11.3 Common elements of artifact cards

The artifact cards of the EscrowAI Project page contain common meta data elements: the Name field, Description field, Version Tag field, and Version Description fields display within each card. They are used to provide a description of what is different or new about this project. There is also an area where you can upload the file associated with the artifact. These elements are repeated on each Project page, and users are expected to fill in the appropriate information for each field.



Field	Description
Name	Name of the artifact. This is auto-populated by EscrowAI.
Description	Further describe the artifact as needed.

Field	Description
Version Tag	The user's version for the artifact. This can be any form of alphanumeric version tagging. EscrowAI creates fully unique and sequential version numbers for each artifact to ensure relational consistency and accurate record keeping. This is shown as the "System Version" associated with each artifact.
Version Description	A description of the version. This is useful for adding details about the uniqueness of the version. For example: • "Initial version" • "Changes in v2 include corrections to the gender classification in the data set to include genetic gender only." • "v2.0.3 of the algorithm corrects a defect that interrupted processing of images with less than 800x600 pixel resolution."
Upload File	This is a place to drag and drop a file, or to open your file navigator to select and upload a file.

EscrowAI assigns a system version to each artifact version, in addition to the version tag entered by the user. This ensures versions are uniquely numbered.

---

12 EscrowAI cryptography

Encryption is intrinsic to EscrowAI's confidential computing.

Data and algorithms are encrypted at rest, in transit, and in process. EscrowAI uses both customer-centric and machine-centric encryption to enforce absolute confidentiality and Zero Trust.

12.1 Key storage

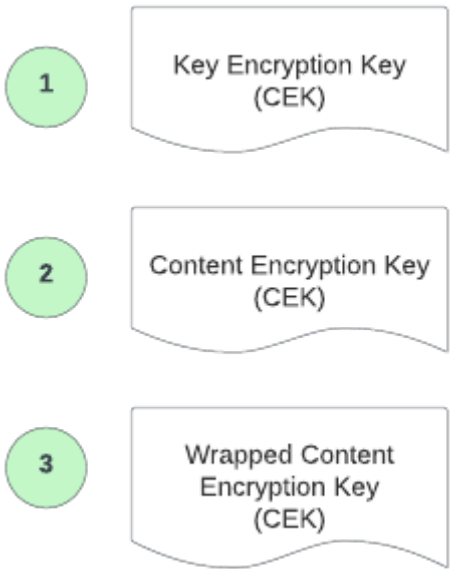
Backing up the platform is a data security management system that performs a number of functions. It is a FIPS 140-2 Level 3 Hardware Security Module (HSM) key manager that is outside of the management control of EscrowAI or BeeKeeperAI, and is not accessible by any user. The key vault generates and stores all the keys and secrets used within EscrowAI, with the exception of:

- The customer's Content Encryption Key (CEK). This is a data encryption key that they create with the Escrow AI encryption tool or with their own key management solution. It is stored in their own key manager.
- The Wrapped Content Encryption Key (WCEK) that the customer uploads to EscrowAI as part of the project. Because this is ciphertext, it can be stored in the EscrowAI platform.

EscrowAI encryption is based on certificates and public/private key encryption with TLS-encrypted communication and AES-256 ciphers.

12.2 Types of keys

Encryption of datasets and algorithms in EscrowAI relies on the following types of keys.



Key	Description
Key Encryption Key (KEK)	The EscrowAI generates a unique, separate asynchronous key pair for the data steward and algorithm owner. The KEK[27] is the public half of the asynchronous key pair. In a public-key encryption system, anyone with a public key can encrypt data yielding a ciphertext, but only those with the corresponding private key can decrypt the ciphertext to obtain the original data. The KEK is the cryptographic key that is used for the encryption of the CEK ("wrapping") to provide confidentiality and protection for that key, allowing it to be sent to EscrowAI as ciphertext. The KEK is available for download from the project page. The private half of this key pair (the half that can decrypt) is retained in the key vault after generation and is only available for decrypting the wrapped CEK within an attested Trusted Execution Environment initiated from the associated project. The private key is only released to the corresponding project confidential container operating in the TEE after an attestation process enables the TEE to cryptographically prove that: • A genuine TEE is running the code, built and signed by the user, unmodified. • The TEE platform is secure and running the necessary microcode updates at runtime. • The configuration requirements of the TEE are met by the hardware and software.
Content Encryption Key (CEK)	A CEK is a private, synchronous data-encryption key[28] used to encrypt and decrypt data or algorithms. You create your CEK using EscrowAI's encryption tool or your organization's key manager. The CEK is used to encrypt a data set and the intellectual property within the algorithm. You should secure your private CEK to guard them against unwanted access. Consider using a password manager or your organizations key manager.
Wrapped Content Encryption Key (WCEK)	A WCEK is a CEK that has been encrypted ("wrapped") by a KEK. The WCEK is ciphertext. You create your WCEK with EscrowAI's encryption tool.

27 https://csrc.nist.gov/glossary/term/kek
28 https://csrc.nist.gov/glossary/term/data_encryption_key

12.3 EscrowAI encryption tool

EscrowAI has an encryption tool to make cryptography more easy.

The encryption tool runs on your local computer through the web browser using your local computer's resources. This is indicated by a message from the encryption tool that it is running offline.

EscrowAI's encryption tool is based on the well-known Web Crypto API. However, if you like, you can use a different key generator to create a Content Encryption Key, as long as it is a 32 byte AES-256-cipher-based key.

12.4 Sequence of key use

1. A project is created in EscrowAI. An asynchronous key pair is generated for each role in the project. The public half (KEK) is available to download from the project page.
2. Each role generates a private Content Encryption Key (CEK) with either the EscrowAI encryption tool or their organization's key manager.
3. Use the CEK generated in (2) to encrypt your dataset or algorithm or using the EscrowAI encryption tool.
4. Use the EscrowAI encryption tool to wrap the CEK created in (2) with the KEK downloaded in (1) to create your Wrapped Content Encryption Key (WCEK).
5. Upload the WCEK to the project page.

12.5 Key file naming convention

EscrowAI's encryption key file names follow the naming convention detailed here, where:

- `<project-name>` is the hyphenated title of your project
- `*` is a system-generated random string.

Type of Key	File Naming Convention
Key Encryption Key (KEK)	The `.der` suffix: • For Data Steward: `<project-name>*-ds.der` • For Algorithm Owner: `<project-name>*-ao.der`
Content Encryption Key (CEK)	`cek_*.key`

Type of Key	File Naming Convention
Wrapped Content Encryption Key (WCEK)	`wcek_*.key.bkenc`

12.6 Common encryption steps

EscrowAI's encryption tool is used to create the keys needed for encrypting a dataset or algorithm.

The encryption tool is accessible by the shield icon in the lefthand toolbar.

Both the data steward and the algorithm owner follow these same steps to encrypt either a dataset or an algorithm.

- Download Key Encryption Key from EscrowAI (see page 61)
- Generate a Content Encryption Key (CEK) (see page 62)
- Generate and upload a Wrapped Content Encryption Key (WCEK) (see page 62)

12.6.1 Download Key Encryption Key from EscrowAI

1. Go to the desired project.
2. Scroll to the bottom.
3. On the right side of the project card, click **DOWNLOAD KEY**.

Result: The KEK is downloaded to your computer.

12.6.2 Generate a Content Encryption Key (CEK)

To generate a CEK:

1. Go to the desired project.
2. Select the shield icon in the lefthand sidebar.
3. Select **Generate CEK**.

Result: The CEK is downloaded to your computer.

12.6.3 Generate and upload a Wrapped Content Encryption Key (WCEK)

The Content Encryption Key (CEK) itself must be encrypted and stored in EscrowAI securely. This is accomplished using the Key Encryption Key (KEK) downloaded from EscrowAI. (see page 61) The KEK is used to encrypt or “wrap” the CEK you previously created. This WCEK is a cyphertext version of your CEK that is unwrapped only within an EscrowAI Trusted Execution Environment associated with the project.

EscrowAI retains only the last WCEK uploaded.

> ⚠ Datasets or algorithm files encrypted with a version of the CEK not encrypted within the current WCEK version cannot be decrypted in the TEE.

Two workflows support WCEK generation. One uses the Encryption Tool independently, and the other is integrated within the EscrowAI project page.

12.6.3.1 Steps

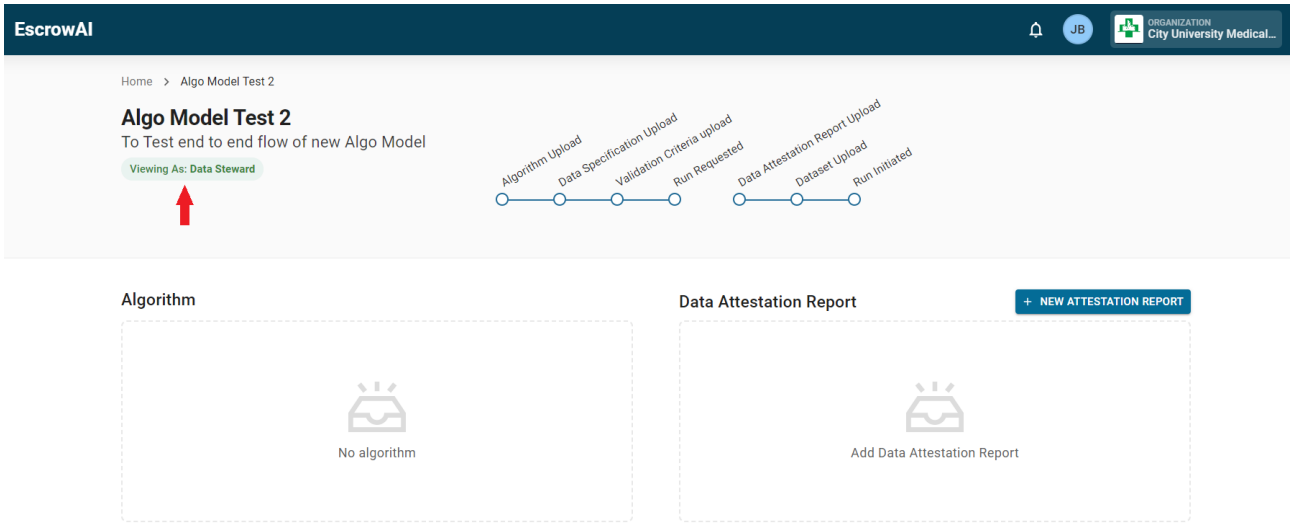
Independent	Integrated
1. Click on the Shield icon in the side menu bar to open the EscrowAI Encryption Client.	1. Under AO or DS Keys on the project page, click either **UPLOAD WCEK** (first upload) or **NEW VERSION** (subsequent change). Fill out **Version Tag** and **Version Description**.
2. Click on **Generate WCEK** from the menu.	2. Click on **GENERATE WCEK** in the Wrapped Content Encryption Key File section.

<table>
<tr><th>Independent</th><th>Integrated</th></tr>
<tr><td>3. Add your Key Encryption Key (drag and drop into the window or click on the window to select a file).</td><td>3. The Key Encryption Key is automatically populated.</td></tr>
<tr><td colspan="2">4. Add your Content Encryption Key
a. Drag and drop a previously created CEK into the window or click on the window to select a file.
b. Alternatively, click GENERATE CEK for the Encryption client to generate a CEK, which is saved in your default local Downloads folder. Once the file is downloaded, drag and drop or select a file to add to the CEK window.</td></tr>
<tr><td colspan="2">5. Select ENCRYPT to create the WCEK.</td></tr>
<tr><td>6. The WCEK is downloaded to your local computer.</td><td>6. The WCEK is uploaded to the AO/DS Keys section of the project page.</td></tr>
<tr><td>7. Under AO or DS Keys on the project page, click either UPLOAD WCEK (first upload) or NEW VERSION (subsequent change).
Fill out Version Tag and Version Description.</td><td rowspan="2"></td></tr>
<tr><td>8. Drag and drop the WCEK downloaded in step 6 into the window or click on the window to select a file.</td></tr>
<tr><td colspan="2">You can also CANCEL to exit the page or RESET FORM to start again with a blank page.</td></tr>
</table>

13 Data steward step-by-step in EscrowAI

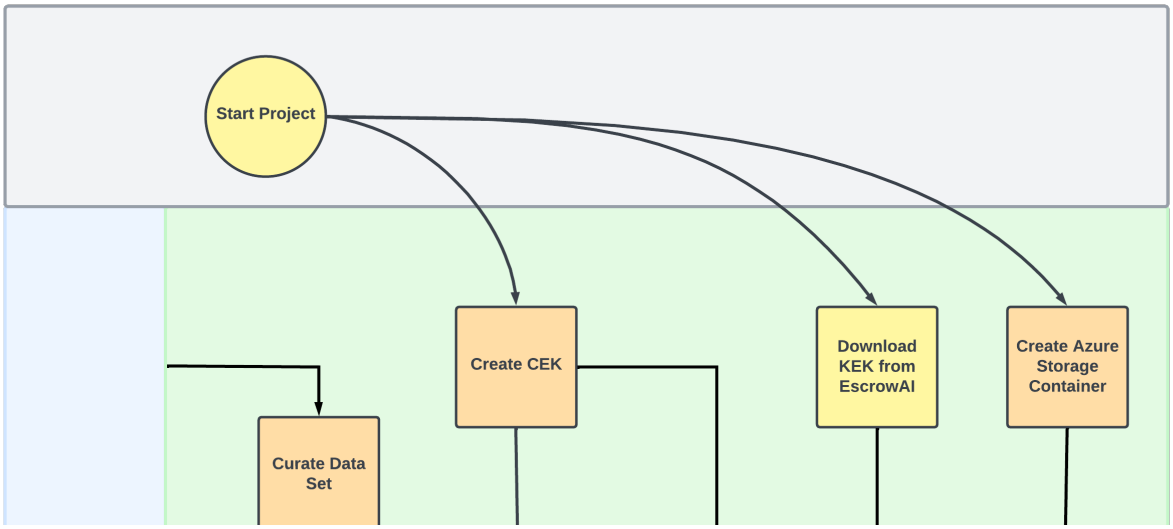
13.1 Project page

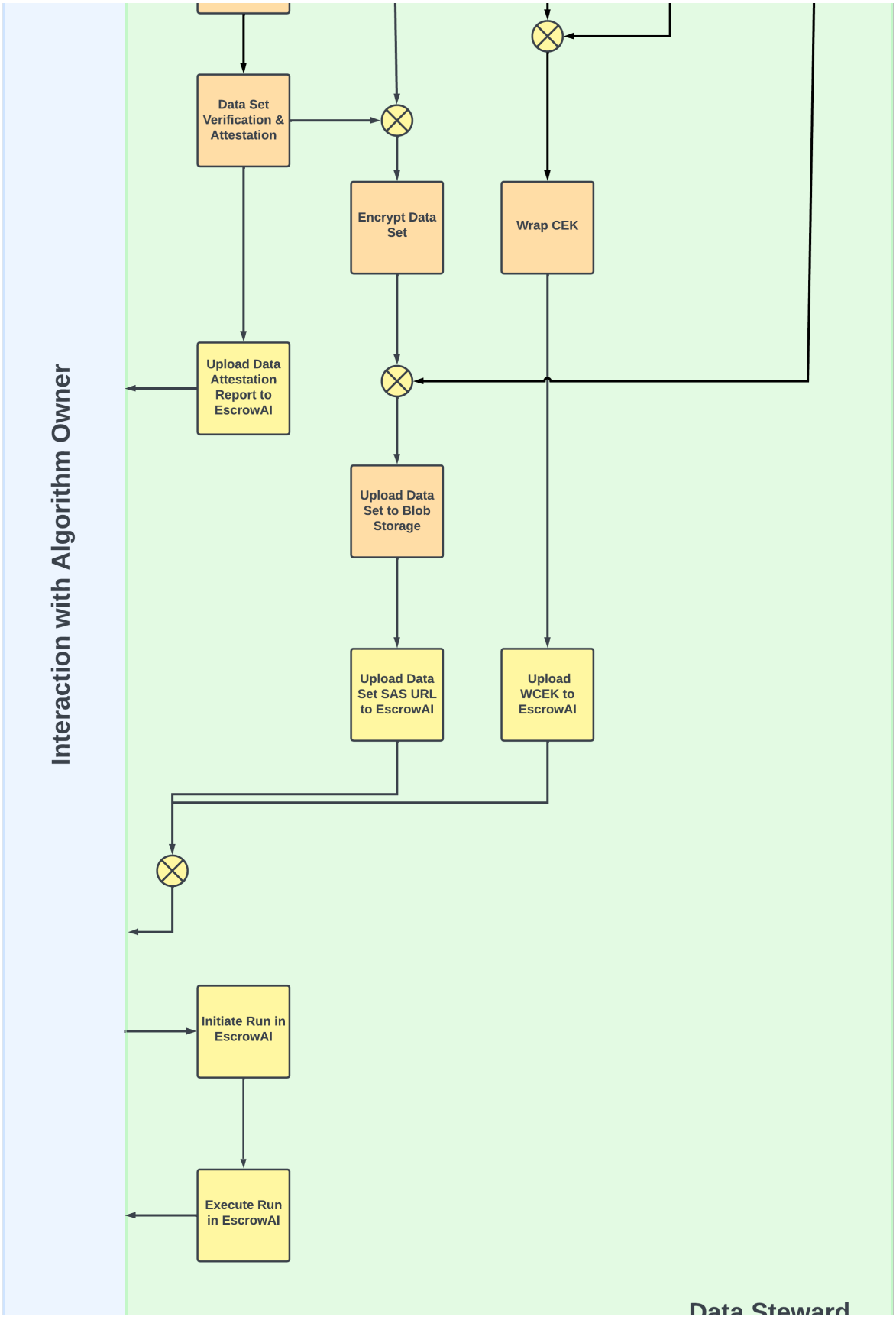
The Data Steward role is identified in the project page below the project name and description. The Data Steward's project interaction is focused on the right side of the project page.



13.2 Data Steward steps

The Data Steward's half of the collaboration workflow with the algorithm owner is shown in the following flowchart and detailed in the sections shown on the left.




Interaction with Algorithm Owner
Data Set Verification & Attestation
Encrypt Data Set
Wrap CEK
Upload Data Attestation Report to EscrowAI
Upload Data Set to Blob Storage
Upload Data Set SAS URL to EscrowAI
Upload WCEK to EscrowAI
Initiate Run in EscrowAI
Execute Run in EscrowAI
Data Steward

6 Focus on data steward's steps of collaboration with algorithm owner

---

13.3 Curate data per AO Data Specification

The data specification is a critical component of a project within which the Algorithm Owner defines the specific requirements for the data that will be used in the algorithm. This includes details such as the subject population (e.g., inclusion and exclusion criteria), the number of records, the data content and form specification, and the data set validation requirements. The data specification ensures that the data used in the algorithm is accurate, complete, and meets the necessary quality standards. It also helps to ensure that the algorithm produces reliable results that can be used to support decision-making or research. In summary, the data specification is a vital part of the collaboration between the Data Steward and Algorithm Owner, and it helps ensure the project's success by providing clear and concise requirements for the data used in the algorithm.

13.3.1 Download data specification

As a Data Steward, you can download a data specification from the Project Page.

1. Go to the Data Specification section from the Project Details page.
2. Click **PREVIEW**.
   A preview window is displayed.
3. Click **DOWNLOAD**.

Result: The data specificatio is downloaded to your computer.

13.3.2 Curate the data

The Data Steward uses their organization's tools and processes to query, combine, form, and clean the data according to the Data Specification.

The creation of a *truth standard* for model training and inference validation is an important step of data curation. The form of the truth standard is part of the Data Specification. The truth standard may be based on a subject matter expert's adjudication of a classification (e.g., confirming a diagnosis through record or image review), or the SME's expert labeling of an area of interest in a medical image. One or more SMEs may be needed to establish the truth standard.

13.4 Attest conformance to data specification

Because patient data is never visible to the algorithm owner, the data steward can certify that the data meets the algorithm owner's stated requirements using a data attestation report. The report includes a summary of the population statistics that demonstrates compliance with the specification, along with the truth standard and the acceptance criteria used to validate the data set.

The Data Steward (DS) retrieves data from various data repositories to fulfill the data specification. The data is structured as required by the Data Specification. The DS verifies the data set by using the method specified in the Data Specification. The DS creates a Data Attestation Report that is approved by the Algorithm Owner (AO). Once approved, the DS encrypts the data set using a DS-controlled content encryption key and deploys it to Azure Blob storage within its Azure compliant cloud.

13.4.1 Upload the data attestation report

1. Go to the Project Details page and click on the **NEW ATTESTATION REPORT** button to begin adding a new report.
2. The **Name** field is automatically populated based on the Data Specification name.
3. Provide a brief description of the data attestation report you are uploading in the **Description** field.
4. Enter a version number in the **Version Tag**. This version must attest to a data set that will be added later in the workflow. Ideally this is a unique number or alpha-numeric you can use to identify and search.
5. Add a version description in the text box provided.
6. You can either drag and drop your data attestation report into the designated area, or click on the gray box to open your file browser and select and upload the file from your computer or device.
7. Once the file has been uploaded, click the **SUBMIT** button to save.

Result: The data attestation report is added to the project.

13.4.2 Upload a new version of a data attestation report

1. Go to the Project Details page for the relevant project.
2. Select **NEW VERSION** from the Data Attestation Report card.

   The name and description fields are automatically filled.
3. Add the information and upload the new file, as described above.
4. Click **SUBMIT**.

Result: the data attestation is added to the project.

13.5 Encrypt dataset

Before you encrypt it, be sure that your dataset is not zipped. You need to encrypt and upload unzipped files or folders, not zipfiles.

To encrypt datasets using the encryption tool:

1. Go to the desired project page.
2. On the left, click the shield icon to go to the encryption client page.
3. Select **Encrypt Dataset.**
4. In the **Dataset** field, select the files or folder that are your dataset. Compressed (zipped) files or folders are not allowed.
5. Select or omit files to encrypt.
6. In the Content Encryption Key field, select your CEK.
7. Select **Encrypt**.

Result: the encrypted dataset is downloaded to your computer as a zipfile.

13.6 Upload encrypted dataset to Azure Blob storage

13.6.1 Background on Azure Blob storage

Azure Blob Storage is the storage mechanism used by EscrowAI to make the data available to the Azure-based confidential compute resource. The Data Steward (DS) must transfer the encrypted data to a Azure Blob Storage container within a storage account located in their Azure subscription.

13.6.1.1 Storage accounts[29]

A storage account provides a unique namespace in Azure for your data. Every object that you store in Azure Storage has an address that includes your unique account name. The combination of the account name and the Blob Storage endpoint forms the base address for the objects in your storage account.

13.6.1.2 Containers[30]

A container organizes a set of blobs, similar to a directory in a file system. A storage account can include an unlimited number of containers, and a container can store an unlimited number of blobs.

EscrowAI pulls a comprehensive data set out of blob storage into the confidential compute node. The data set is version controlled within the application so that there is traceability of results based on data and algorithm version. For this reason, a container must contain data at the Project-Data Set Version level. For example,

- DS-subscription
    - Projects-Storage-Account
        - Project-A-dataset-v1-container
            - blobs (folders, files)
        - Project-A-dataset-v2-container
        - Project-B-dataset-v1-container

Ensure data sets are put in unique containers at the Project-Data Set Version level.

29 https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction#storage-accounts
30 https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction#containers

13.6.1.3 Blobs[31]

Azure Storage supports three types of blobs, Block, Append, and Page. The specific type of blob used may depend on the project type. The most common type for use with EscrowAI is a **Block** blob.

- **Block blobs** store text and binary data. Block blobs are made up of blocks of data that can be managed individually. Block blobs can store up to about 190.7 TiB.
- **Append blobs** are made up of blocks like block blobs, but are optimized for append operations. Append blobs are ideal for scenarios such as logging data from virtual machines.
- **Page blobs** store random access files up to 8 TiB in size. Page blobs store virtual hard drive (VHD) files and serve as disks for Azure virtual machines.

13.6.2 Instructions to upload encrypted data

Encrypted data can be uploaded in Azure blob storage in several ways, depending on your own operations/ requirements.

1. Azure Portal
2. Azure Storage Explorer
3. Azure CLI
4. Encryption tool in EscrowAI

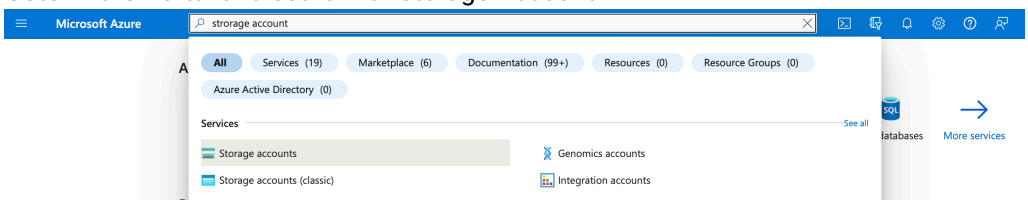
13.6.2.1 Azure Portal

Azure Portal can be used to create and organize storage accounts and containers, and upload and manage Blobs.

1. Create a Storage Account[32]

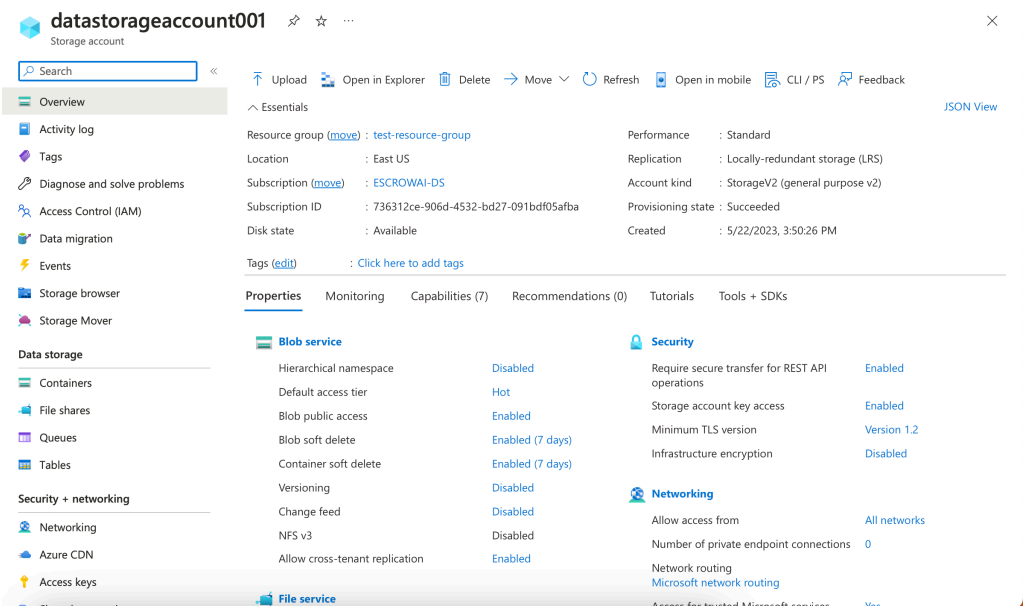
Creation of the Storage Account may be a one-time operation, depending on your organization's policies (e.g., you can create one Storage Account for all EscrowAI projects).

2. Create a Container[33]
    a. Goto Azure Portal and search for Storage Account

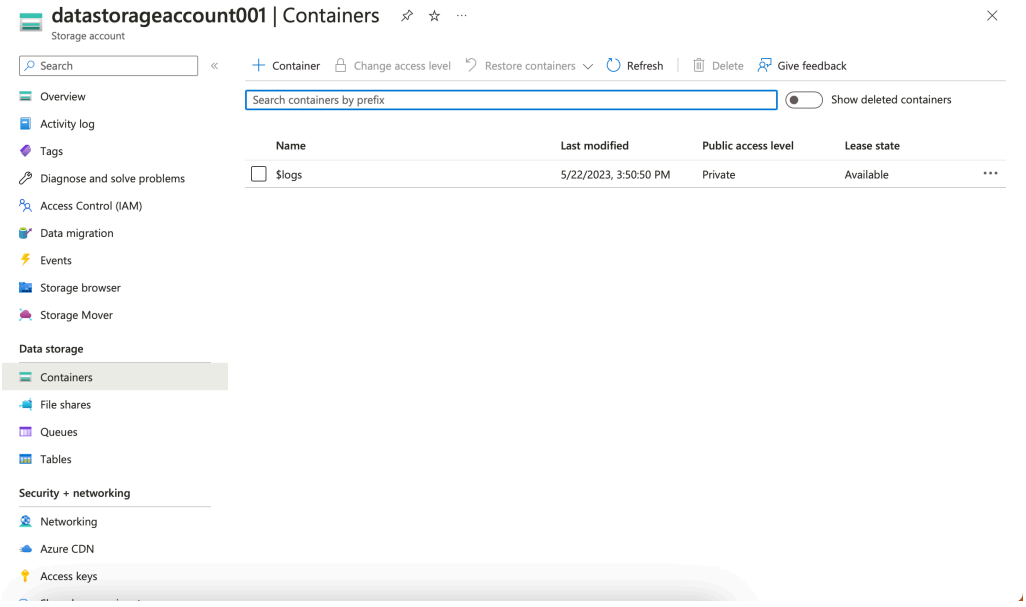


31 https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction#blobs
32 https://learn.microsoft.com/en-us/azure/storage/common/storage-account-create?tabs=azure-portal#create-a-storage-account-1
33 https://learn.microsoft.com/en-us/azure/storage/blobs/blob-containers-portal#create-a-container

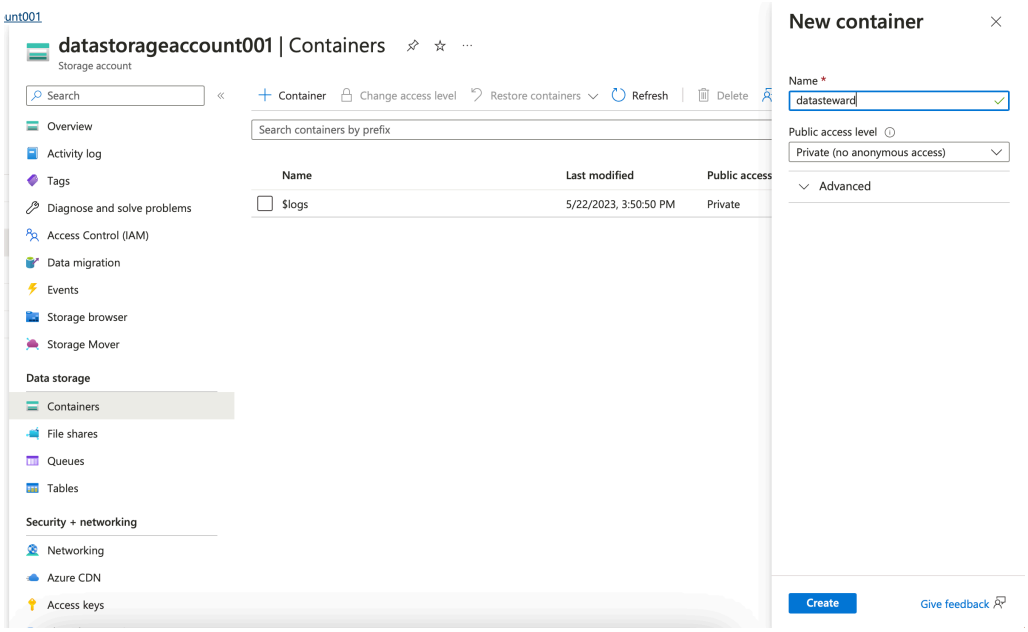
b. Select the Storage Account used for EscrowAI projects.



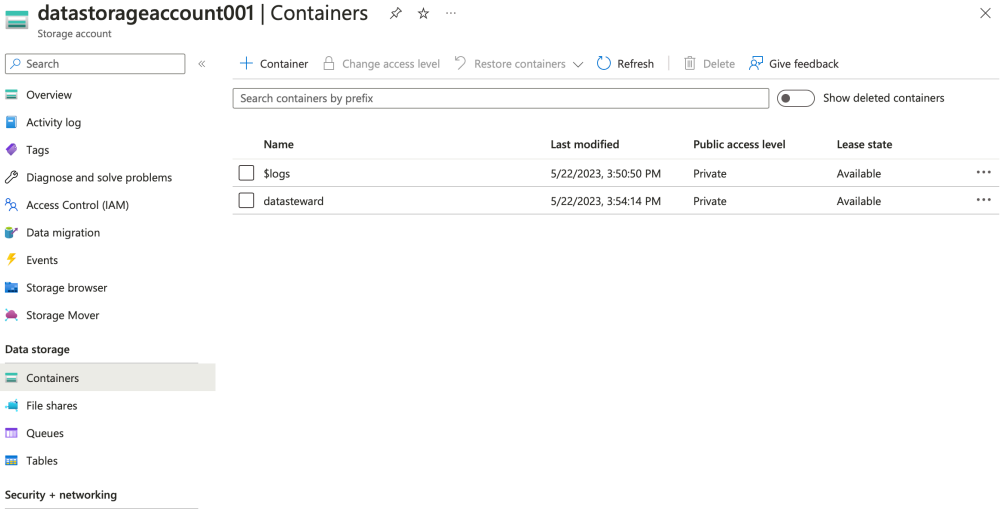
c. Select the **Containers** from the left panel and click on **+ Container** to create a container in storage account.



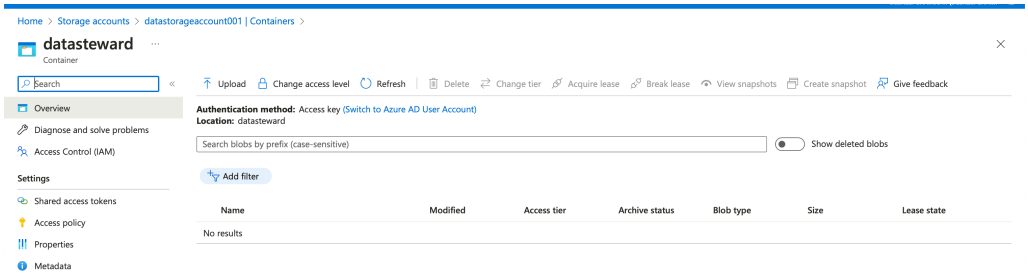
d. Fill in the required details in the right panel and click on **Create.**



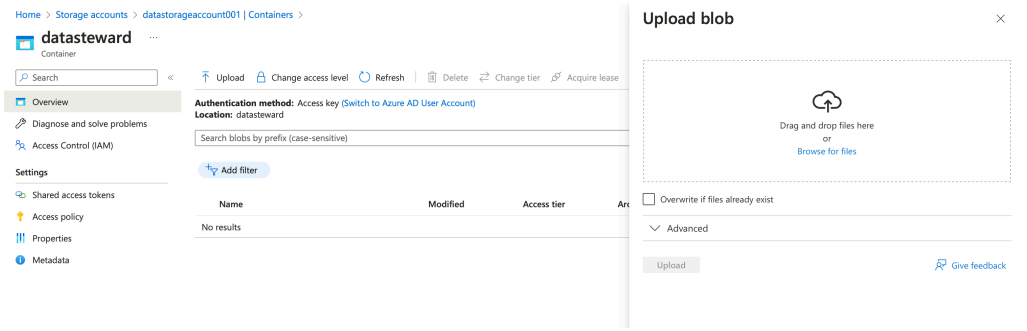
e. Once the container gets created, you will be able to see the container in the Storage account.



3. Upload Blobs using Azure Portal
    a. Select the container that was created in previous step.



b. Click on **Upload** to upload the files to the container. Use the Advanced tab to refine the Blob details.



13.6.2.2 Azure Storage Explorer

Follow Microsoft's instructions from the below link to upload the files to the blob container via Azure Storage Explorer:

Create a blob with Azure Storage Explorer - Azure Storage[34]

13.6.2.3 Azure Command Line Interface

Azure CLI can be used to perform the above operations. The Azure CLI can be invoked from the Bash environment in Azure Cloud[35]Shell or run locally using a locally installed Azure CLI[36].

1. Open a command prompt or terminal window.
2. Log in to your Azure account using the following command:

```
az login
```

3. Create a storage account using the following command (optional):

34 https://learn.microsoft.com/en-us/azure/storage/blobs/quickstart-storage-explorer
35 https://portal.azure.com/#cloudshell/
36 https://learn.microsoft.com/en-us/cli/azure/install-azure-cli

```
az storage account create --name <account-name> --resource-group <resource-group-name> --location <location> --sku Standard_LRS
```

4. Create a container in the storage account using the following command (optional):

```
az storage container create --name <container-name> --account-name <account-name> --account-key <account-key>
```

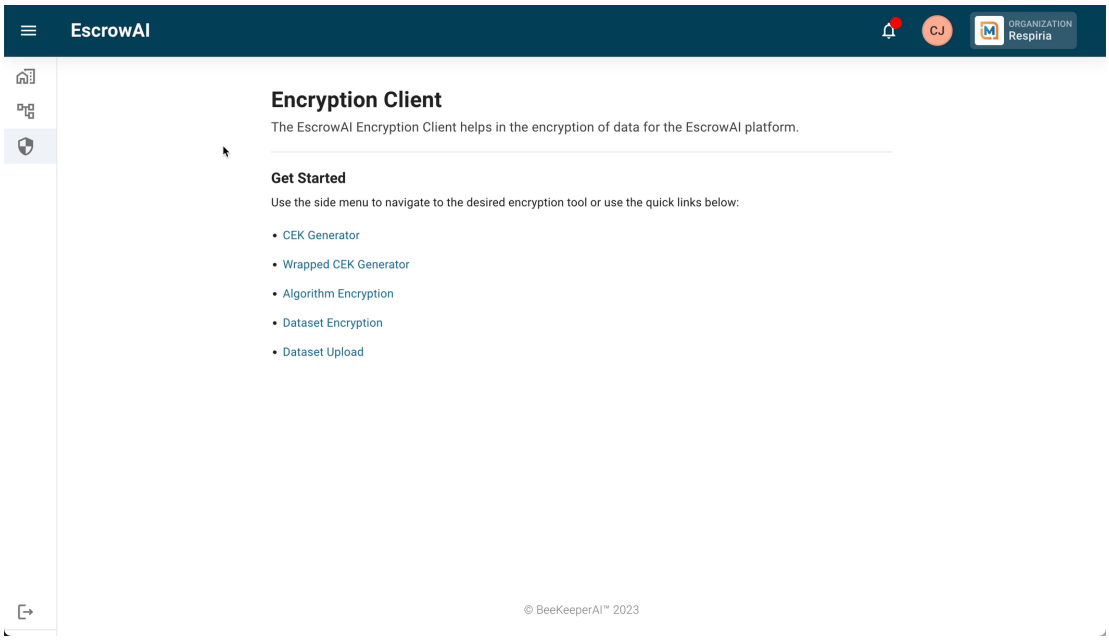
5. If you have storage account-key, upload data to the container using the following command. Replace **<container-name>**, **<path-to-data>**, **<account-name>**, and **<account-key>** with your actual values.

```
az storage blob upload-batch --destination <container-name> --source <path-to-data> --account-name <account-name> --account-key <account-key>
```

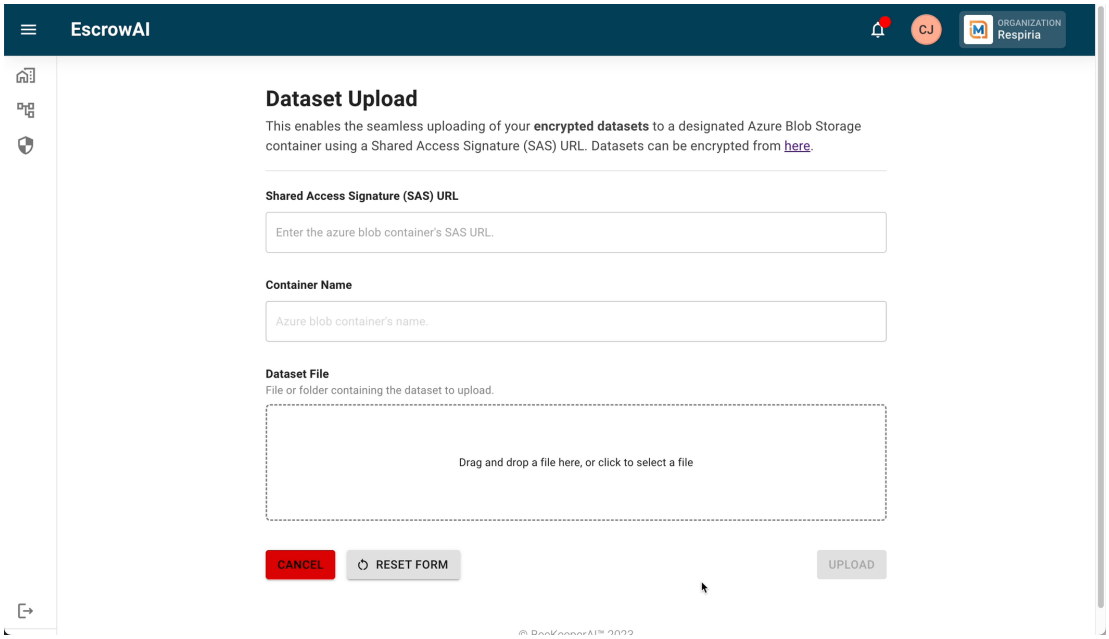
13.6.2.3.1 Upload the encrypted data

After you've obtained your SAS URL, in the EscrowAI UI, you can upload your dataset to the Azure blob container you defined as a prerequisite.

1. Select Dataset Upload from the Encryption Client's home page or side menu.

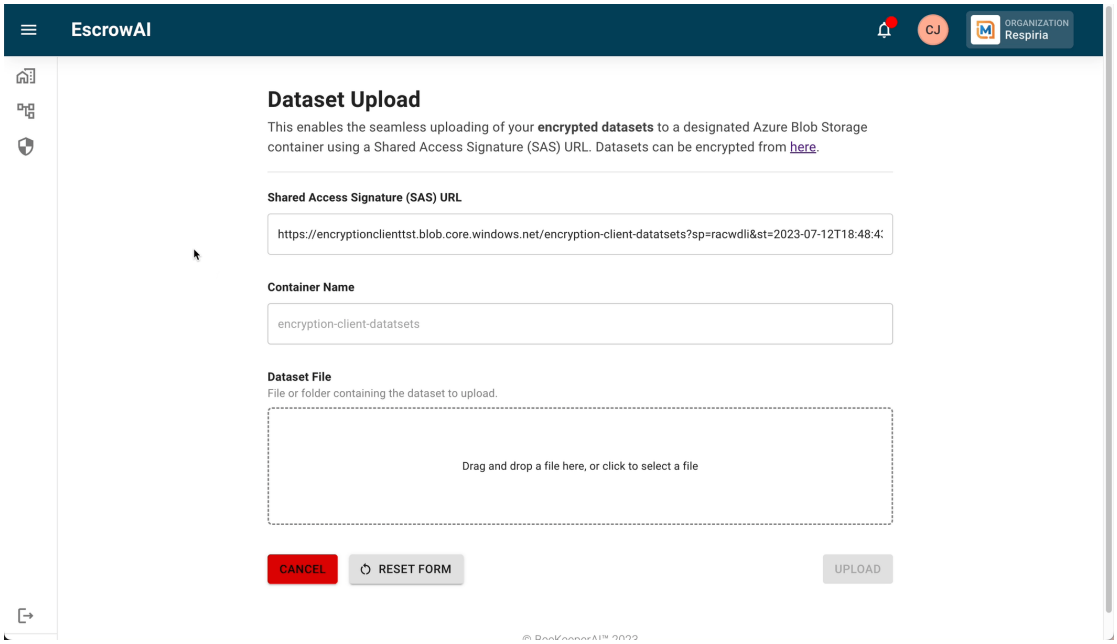


2. In the Shared Access Signature (SAS) URL field enter the SAS URL you generated.
   The validity of the SAS URL is checked by EscrowAI.
   The **Container Name** field is filled in by the system.

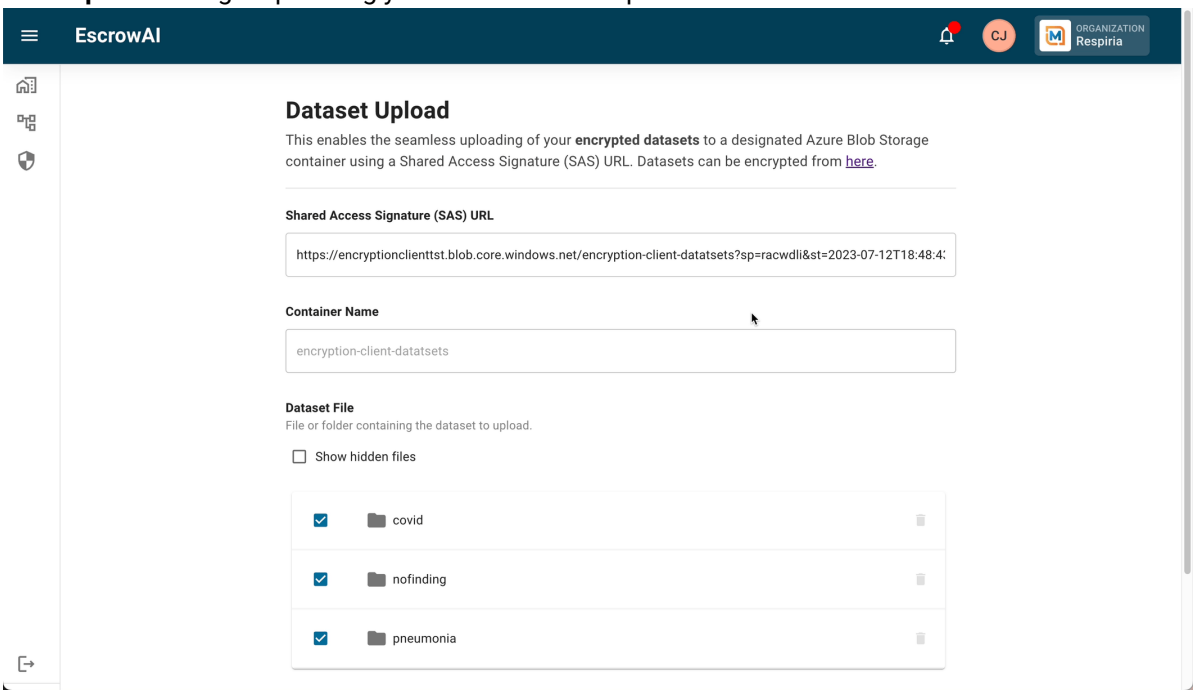
- 

3. In the Dataset File field, select/drop your unzipped and encrypted dataset file.
   After that, you can use the EscrowAI UI to omit any files you don't want to be uploaded from the rendered directory view.

> Do not move your files on your local system until you complete and submit the form in EscrowAI.



4. Click **Upload** to begin uploading your dataset to the specified Azure Blob Container.



13.6.3 Related Microsoft documentation

Azure Storage Explorer – cloud storage management[37]

37 https://azure.microsoft.com/en-us/products/storage/storage-explorer

Create a blob with Azure Storage Explorer - Azure Storage[38]

Upload, download, and list blobs - Azure CLI - Azure Storage[39]

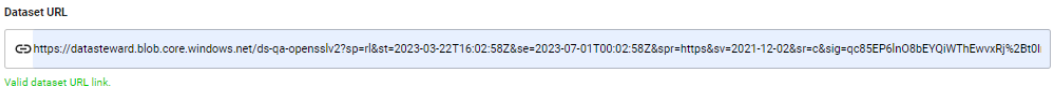
Azure Cloud Shell Overview[40]

13.7 Add a dataset URL to EscrowAI

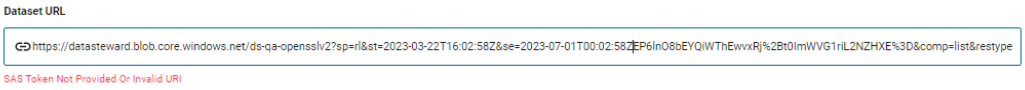
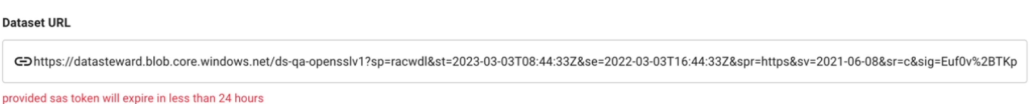
Adding a dataset to EscrowAI does not transfer any data to the EscrowAI. Instead, the SAS URL is recorded in EscrowAI along with dataset metadata. No data leaves the Data Steward environment because the confidential compute node is spun up within the Data Steward's environment and the dataset is streamed into the confidential compute node from the Data Stewards Azure BLOB storage.

13.7.1 Add a new dataset

1. Go to the desired project page.
2. Click **NEW DATASET**.
3. Fill in the required fields.
4. Select the proper Data Specification and Data Attestation Report versions.
   This communicates to the Algorithm Owner the exact dataset contents.
5. Enter your Dataset SAS URL.
   EscrowAI checks the SAS URL for validity, with status shown below.
6. If the token is invalid, generate a new SAS URL and re-enter it in the **Dataset URL** field. See Generate a Signed URL for the Dataset (see page 78).
7. Check the confirmation box that the associated dataset does not contain any unencrypted privacy-protected or confidential information.
8. Click **SUMBIT** to save.

Dataset URL	Description
Valid	The Dataset URL link is valid and can be successfully saved. 

38 https://learn.microsoft.com/en-us/azure/storage/blobs/quickstart-storage-explorer
39 https://learn.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-cli
40 https://learn.microsoft.com/en-us/azure/cloud-shell/overview

Dataset URL	Description
Invalid	Either the SAS token was not provided or an invalid URL was entered. 
Expired	The provided SAS token will expire in less than 24 hours. 

13.7.2 Create a new dataset version

If the new dataset is the result of a change in the Data Specification or is associated with a new Data Attestation Report, ensure that those artifacts are added to the project before adding the new dataset, so that the appropriate relationships can be selected.

A new version of a dataset can be added by clicking on the **New Version** button on the Dataset card. Enter the information as described above.

13.7.3 Updating the SAS URL

A SAS URL has a specific expiration date. If this expiration will occur in 24 hours or less, the project page indicates that the SAS URL is expired. This means that the dataset itself has been marked as expired.

A new SAS URL must be added after the expiration date of an existing SAS URL. Adding a new SAS URL requires that a new dataset version be added.

For clarity, add a comment in the Version Description field that the new version was created only to update the SAS URL. Select the appropriate Data Specification and Data Attestation Report.

13.8 Generate a Signed URL for the Dataset

Generating a shared access signature for a dataset allows you to securely and easily share access to your data with EscrowAI. A shared access signature (SAS) is a URI that grants restricted access rights to a specific dataset stored in the Data Steward's Azure Storage subscription. You need the appropriate permissions and access to the dataset to generate a signed URI.

EscrowAI requires an account SAS to be entered for the **Container** holding the data set. This allows the platform to pull the data set into the confidential compute node during a run. The SAS URI must be valid within the start and expiry dates and times when a run is initiated and in progress.

Data in the Container will be available to EscrowAI for the duration of the SAS token. The duration specified is at the discretion of the Data Steward and should balance the security needs with access over the project duration. When a SAS token expires, a new token must be generated and entered into EscrowAI before a new run can be initiated.

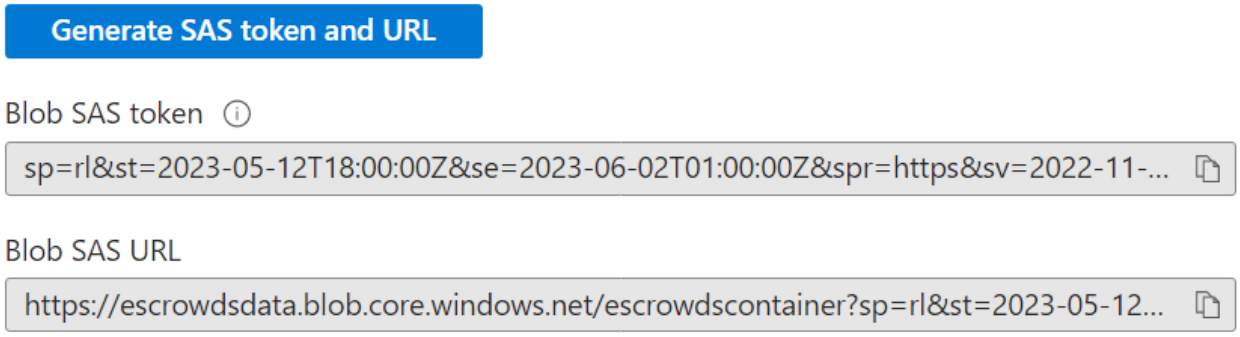
13.8.1 Steps

1. Log in to your Microsoft Azure portal[41] and select the desired subscription.
2. Select the target Storage account.
3. Select **Containers** in the Resource menu.
    a. Click on the desired Container in the working pane.
    b. Click on the three dots on the right side of the row to expose the activity options menu.
    c. Select **Generate SAS**.

4. In the **Generate SAS** window, complete the options.

- Signing method: **Account key**
- Signing key: select desired key.
- Permissions dropdown: select **Read** and **List**.
- Specify the **Start** and **Expiry** dates and time for URI.
- Select **HTTPS only**.

5. Click **Generate SAS token and URL**. Copy the **Blob SAS URL**, which includes the token (shown below), and paste it into the required field in the EscrowAI project.



41 https://portal.azure.com/

13.9 Review run request and initiate or reject a run

A run request is configured and sent by the Algorithm Owner when the necessary algorithm, data set version, and validation criteria are available on the Project page.

Each run configuration is a unique combination of algorithm and data set versions. The AO can send a request for either a new or existing configuration.

13.9.1 Review a run request

When the Algorithm Owner (AO) sends a Run Request, the Data Steward (DS) receives an email notification.

The request is indicated by the **Run Requested** status on the DS view of the **Run Configurations** card.

13.9.2 Reject a run request

You can either accept the run request or reject it. If you reject, enter a reason for the rejection.

- The algorithm owner receives an email message stating the reason.
- A notification is posted to the bell on the algorithm owner's home page.

13.9.3 Initiate a run

1. Go to the project in EscrowAI and select the **Run Configuration** card.
2. Review the configuration of the Run Request and ensure it is appropriate.
3. Click **INITIATE RUN**.

EscrowAI allows only one run of a particular Run Configuration at a time. The **INITIATE RUN** button associated with any Run Configuration is active only when there are no confidential computing resources processing that same configuration.

13.10 Monitor and cancel a run

13.10.1 Run status

Once the Data Steward (DS) initiates the run, EscrowAI will start the requested TEE in the DS's environment and initiate the Run.

The run progresses through stages from building the runtime, launching the confidential computing virtual machine, and running the desired task or program. Monitor the top-level status of the Run in the **Run Updates** sidebar. This progression shows the progress of bringing up the TEE and initiating the run.

Run Updates	Description
Run Initiated	The run configuration has been submitted and the process of initiating the runtime environment is about to begin.
Runtime Build	The runtime environment is being built according to the specifications of the run configuration.
Workload Launch	The virtual machine (VM) that will run the algorithm is launched, the TEE is attested, and the run artifacts are brought into the TEE.
Run	The algorithm is currently running on the virtual machine.

13.10.2 View the status of a run in progress

Status updates, run logs, and the final report or model are displayed in the run configuration's **Reports** window.

For a **Run in Progress** the Report window display a running status of the workload, updated at periodic intervals.

1. Go to the desired project.
2. Select the **Run Configurations** card. The **Run Details** tab for the last initiated Run is displayed.
3. The **Run Details** tab displays the Run Configuration details, the status of the **Run in Progress**, and a list of **Run Reports** associated

   with the configuration.
4. Click on the **Report** link in the Actions column for the Run In Progress to view the detailed run progress logs.

13.10.3 Cancel a run in progress

The Data Steward can cancel a run in progress by clicking the **CANCEL RUN** link in the Actions column under Run Reports.

13.10.4 Completed runs

After a run is finished, its status changes to either **Run Completed** or **Run Failed**.

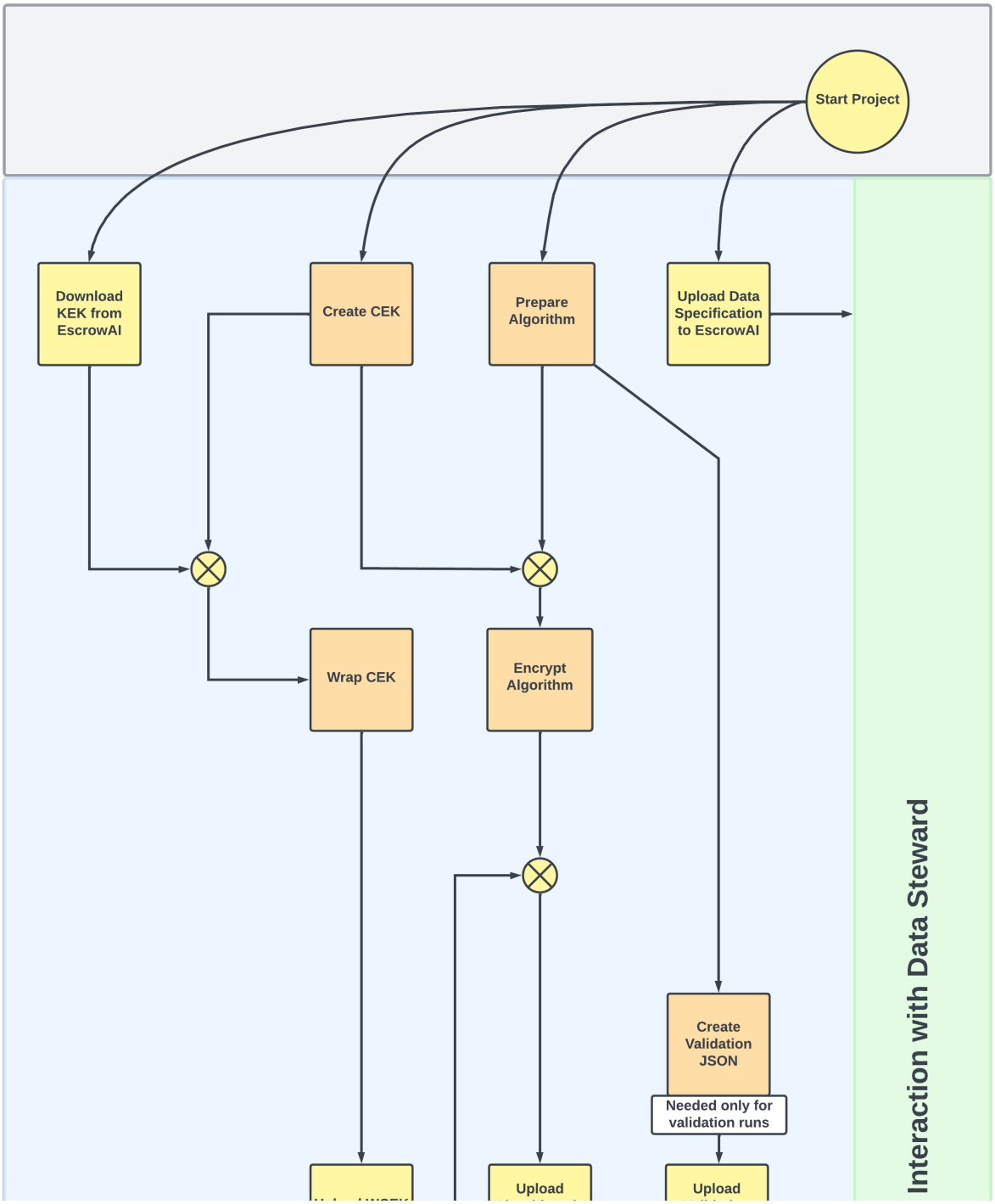
- The final report or trained model is viewable only by the algorithm owner.

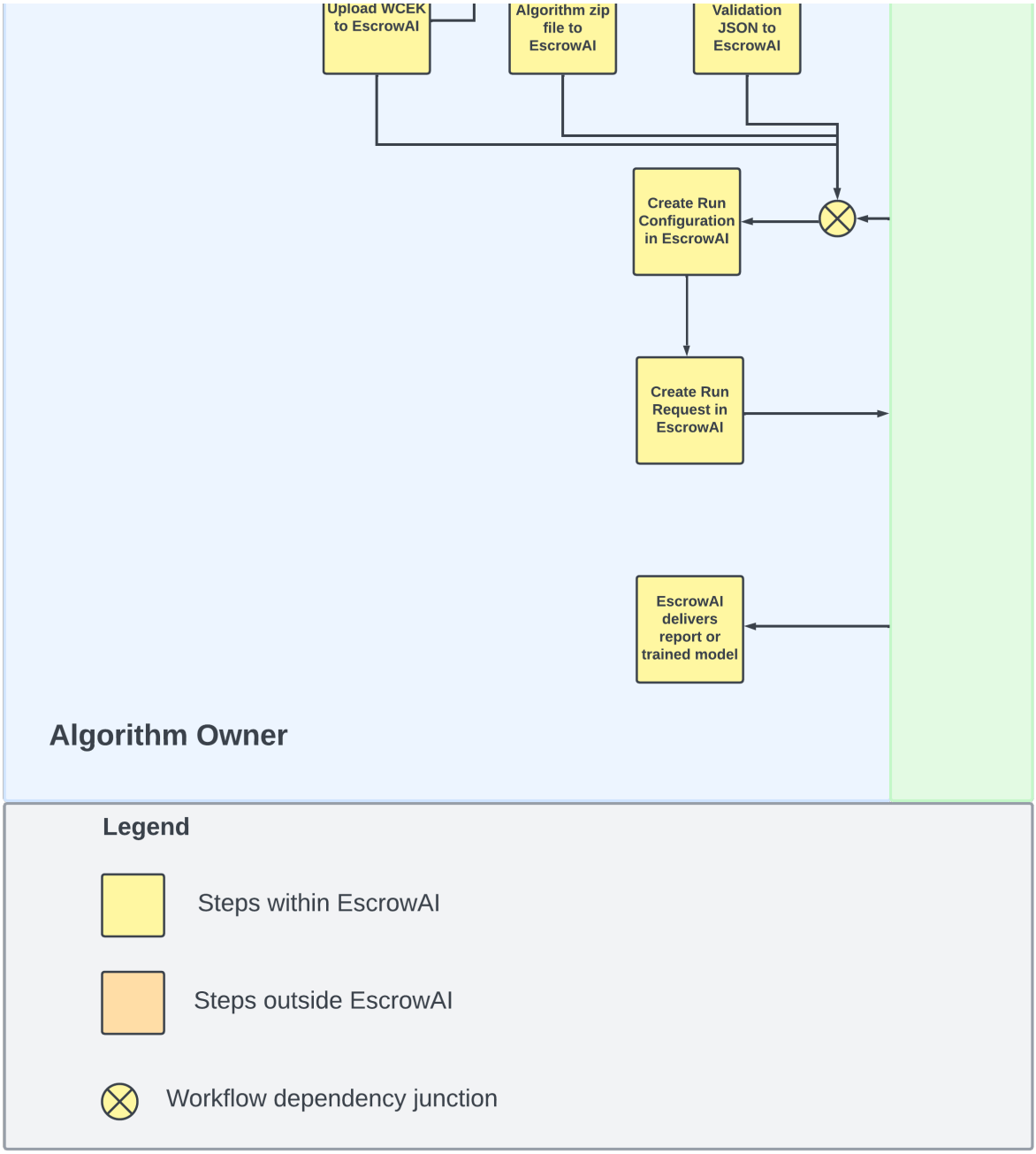
- Both the algorithm owner and the data steward can see the run logs.

See Get run results (see page 97).

14 Algorithm owner step-by-step in EscrowAI

The algorithm owner's half of the collaboration workflow with the data steward is shown in the following flowchart and detailed in the sections shown on the left.





7 Focus on algorithm owner's steps of collaboration with data steward

14.1 How algorithm runs are identified: version combinations

Each run configuration created by the algorithm owner must have a unique combination of dataset version and algorithm version. Together, this is the unique identifier for a particular run configuration.

14.2 Create and upload Data Specification

The Algorithm Owner (AO) proposes the Data Specification that supports the intent of the project and the implementation needs of the algorithm. In EscrowAI, the Data Specification is vital for the AO and Data Steward (DS) as it ensures that the data meets the requirements without known issues or errors and complies with industry standards and regulations.

For more information about the Data Specification from a Data Steward's perspective, see Curate data per AO Data Specification (see page 66).

14.2.1 Supported file formats for Data Specification

The Data Specification must be in one of the following file formats:

- Microsoft Word .doc file.
- PDF.

14.2.2 Suggested outline of Data Specification

The topics highlighted below should form the basis of a Data Specification, although the exact form and content are based on the needs and requirements of the collaborating partners. These sections are provided as a guide only.

14.2.2.1 Section 1: Hypothesis Test and Methodology

This section defines the purpose of the algorithm or query and its expected output.

14.2.2.2 Section 2: Population Description

This section describes the target population for the algorithm, including demographic information such as age, gender, and ethnicity. It also outlines the inclusion and exclusion criteria for individuals to be included in the dataset.

14.2.2.3 Section 3: Data Specification

This section details the required data for the algorithm, including the types of data (e.g., text, images, audio) and any specific format requirements (e.g., file type, encoding, ontology). It also describes any pre-processing steps that may be necessary.

14.2.2.4 Section 4: Truth Standard

This section describes the method for determining the ground truth for the dataset: the correct classification or label for each data record. It also outlines the criteria for each classification or label.

For example, a dataset of chest X-ray images labeled as either "normal" or "abnormal" for a pneumonia detection algorithm. The ground truth for this dataset could be determined by having a team of radiologists visually inspect each image and classify it as either normal or abnormal based on specific criteria, such as the presence of infiltrates, consolidation, or pleural effusion.

14.2.2.5 Section 5: Data Set Validation

This section outlines the method for validating the dataset, which may involve randomly sampling and inspecting a subset of data points to ensure accuracy and consistency with the population, inclusion and exclusion, data content and form, and ground truth.

14.2.3 Upload Data Specification

To upload the data specification:

1. Go to the desired project page.
2. Click **NEW DATA SPECIFICATION**.
3. Fill in the required fields and upload your Data Specification report.
   The **Name** field is auto-populated from the project name.
4. From your local computer, select or drag-and-drop the desired data specification.
5. Click **SUMBIT** and confirm to save or **CANCEL** to discard the details.

Result: The data specification is uploaded to EscrowAI.

14.3 Package your algorithm for upload

Before you encrypt and upload your algorithm you need to prepare it for the upload.

For details, see Package your algorithm for upload to EscrowAI (see page 31).

14.4 Encrypt algorithm files

Algorithm encryption safeguards the confidentiality of your algorithm's sensitive content during storage, transmission, and execution. Files you want to safeguard are encrypted using your Content Encryption Key (CEK).

This page describes the workflow that encrypts algorithm files using the Encryption Tool as a stand-alone function. The algorithm files can also be encrypted in the New Algorithm page as part of the Upload algorithm with encryption (see page 87) workflow, which facilitates integrated uploading of the algorithm zip file.

To encrypt algorithm files:

1. From the left, click the Shield icon to bring up the Encryption Tool.
2. Select **Encrypt Algorithm**.
3. In the **Algorithm Directory** field, select/drop the folder containing your algorithm files. The top-level folder will be displayed. Click on the folder to expand the directory.
4. From the **Algorithm Files** section, select the files you want encrypted. *Only* the selected files will be encrypted.
5. In the **Content Encryption Key** field, select/drop your CEK.
6. Click **ENCRYPT** to begin algorithm encryption. The selected files are encrypted, and the entire algorithm set is zipped.

Result: The algorithm zipfile is downloaded to your computer.

14.5 Upload algorithm with encryption

You can upload your algorithm after the following prerequisite steps are completed:

1. The AO has created the algorithm package (see page 31).
2. Both the algorithm WCEK and the Validation Criteria are uploaded by the AO.
3. The Data Attestation Report is uploaded by the DS.

The AO uploads the algorithm package zip file to the EscrowAI project space. EscrowAI builds the container, prepares it for use in the TEE, and stores it in the platform's container registry. The workflow described on this page supports both encrypting the algorithm files asynchronously or from within the upload algorithm sequence of steps.

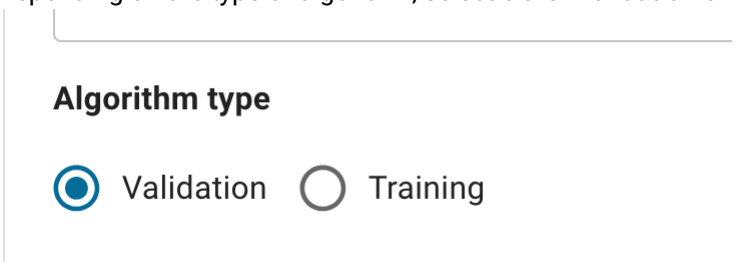
14.5.1 Planning for uploading your algorithm

- **Minimum RAM for Intel SGX EnclaveOS**: If your project is based Intel SGX, when you upload your algorithm, you can specify the minimum amount of RAM you want on the running enclave. This RAM specification is needed for EscrowAI to build the EnclaveOS container. This setting is required only for projects based on Intel SGX EnclaveOS. The RAM specification cannot be changed after you have uploaded the algorithm. To change the RAM setting, you need to upload your algorithm again.
- Decide if your algorithm is for training or validation.

14.5.2 Common steps

To upload an algorithm:

1. Go to the desired project page.
2. Click **NEW ALGORITHM** or **NEW VERSION.**
   The first time, use **NEW ALGORITHM**. For later versions, use **NEW VERSION**.
3. Depending on the type of algorithm, select either **Validation** or **Training**.



4. Fill in the required metadata fields.
   The versions of the dataset and the algorithm must be a unique pair.
5. Select the Data Attestation Report Version. This represents the dataset version your algorithm uses in the TEE.
6. **Only for Intel SGX EnclaveOS**: Select the desired minimum amount of RAM your algorithm requires. This RAM specification is needed for EscrowAI to build the EnclaveOS container. This setting is required only for projects based on Intel SGX EnclaveOS.

You have several options for algorithm encryption and upload:

- Encrypt and upload in a single step.
- Encrypt and upload later.

14.5.3 Encrypt and upload the algorithm

In this use case, you have not already encrypted your algorithm.

Algorithm encryption safeguards the confidentiality of your algorithm's sensitive content during storage, transmission, and execution. Contents are encrypted using your Content Encryption Key (CEK).

1. In the **Upload algorithm** section, select **ENCRYPT ALGORITHM**. The interface shifts to display the **Encrypt Algorithm** page from the Encryption Tool. Follow the instructions below for encrypting your algorithm files.

1. From the left, click the Shield icon to bring up the Encryption Tool.
2. Select **Encrypt Algorithm**.
3. In the **Algorithm Directory** field, select/drop the folder containing your algorithm files. The top-level folder will be displayed. Click on the folder to expand the directory.

4. From the **Algorithm Files** section, select the files you want encrypted. *Only* the selected files will be encrypted.
5. In the **Content Encryption Key** field, select/drop your CEK.
6. Click **ENCRYPT** to begin algorithm encryption. The selected files are encrypted, and the entire algorithm set is bundled into a zip file.

2. Once encrypted, the algorithm zip file will be automatically added to the Upload Algorithm section.
3. Check the box certifying that your algorithm files do not contain privacy protected data.
4. Click **SUBMIT** to upload the algorithm to EscrowAI.

After the zip file is uploaded a success message is displayed.

14.5.4 Upload a previously encrypted algorithm

In this use case, you have already encrypted your algorithm using the Encrypt algorithm files (see page 86) step.

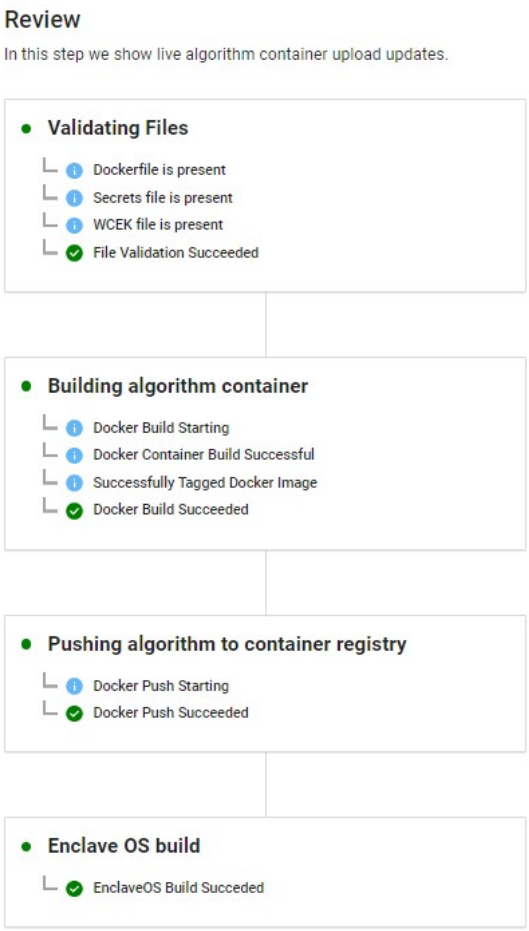
1. Drag and drop or browse and select your *already encrypted* algorithm zip file.
2. Check the box certifying that your algorithm files do not contain privacy protected data.
3. Click **SUBMIT** to upload the algorithm to EscrowAI.

Result: fter the zip file is uploaded a success message is displayed.

14.5.5 View Upload Status

To see the status of the upload:

1. Go to the desired project.
2. The upload status is shown on the right sidebar of the **New Algorithm** page.
3. If you navigate away from the **New Algorithm** page before the upload is complete, or to check the upload details of any previous algorithm upload, follow these steps.
    a. Click on the **Algorithm** card in the project page.
    b. Click the **Version Tag**.
    The Version Details window is displayed.
    c. On the upper right, click **VIEW UPLOAD STATUS**.
    The **Review** panel displays each step of the upload status.



14.5.5.1 Validating files

This is the process of checking if the required files are present in the upload.

Status	Description
Dockerfile is present	A script that contains instructions for building a Docker container image. This step checks if the file is present.
Secrets file is present	A secrets file contains the list of files that have been encrypted by the AO. This file lets the enclave know which files need to be decrypted in the enclave during confidential compute. This step checks if the file is present.
WCEK file is present	A WCEK file contains the encryption key for confidential data. This step checks if the file is present.

Status	Description
File Validation Succeeded	Confirms that all required files are present and the validation has been successful.

14.5.5.2 Building algorithm container

This is the process of building a Docker container image that contains the algorithm.

Status	Description
Docker Build Starting	Starts the Docker container image building process.
Docker Container Build Successful	Confirms that the Docker container image building process has been successful.
Successfully Tagged Docker Image	Attaching a specific version tag to the built Docker container image.
Docker Build Succeeded	Confirms that the entire Docker container image building process has been successful.

14.5.5.3 Pushing algorithm to container registry

This is the process of pushing the built Docker container image to a container registry, which is a central location for storing and managing container images.

Status	Description
Docker Push Starting	The process of pushing the built Docker container image to the container registry.
Docker Push Succeeded	Confirms that the built Docker container image has been successfully pushed to the container registry.

14.5.5.4 Enclave OS build

This is the process of building the secure environment where the algorithm will run when using the Intel® Software Guard Extensions (Intel® SGX) confidential computing enclave.

Status	Description
Enclave OS Build Succeeded	Confirms the process of converting the Docker container in the container registry to include EnclaveOS and pushed it back to the container registry.

14.6 For validation runs: upload the Validation Criteria JSON

The validation criteria in JSON format are required only if you plan on requesting a validation run. For training runs, validation criteria are not required.

The algorithm output must be generalized to show performance or data characteristics without including protected privacy information, and the Validation Criteria is the policy that enforces this output. To ensure that the algorithm's report does not include any protected or private information, the Algorithm Owner (AO) must create a Validation Criteria policy in the form of a JSON file that exactly represents the algorithm's output. The Data Steward must agree to the Validation Criteria form and content in order to initiate a Run. EscrowAI checks the algorithm output within the TEE to ensure compliance with the Validation Criteria before pushing the report out to the AO project space.

For an example of validation criteria in JSON, see Example of JSON Validation Criteria (see page 93).

The validation criteria must be accepted by the Data Steward (DS) prior to execution of run.

14.6.1 Steps

14.6.1.1 New Validation Criteria

To upload validation criteria:

1. Go to the desired project.
2. Click **NEW VALIDATION CRITERIA**.
3. Fill in the required fields and upload your validation criteria.
   The **Name** and **Description** fields will be auto-populated based on the project name.
4. Click **SUMBIT** to save.

14.6.1.2 New Version of Validation Criteria

To add another version of validation criteria after the first upload:

1. Go to the desired project.
2. Under **Validation Criteria**, click **NEW VERSION**.
3. Fill in the required fields and upload your new validation criteria.
4. Click **SUBMIT** to save.

14.6.2 Example of JSON validation criteria

This is an example of a JSON structure for data validation.

```
{
    "report": {
        "type": "dict",
        "schema": {
            "accuracy": {
                "type": "dict",
                "schema": {
                    "value": {"type": "float", "min": 0, "max": 1},
                    "CI": {"type": "float", "min": 0, "max": 1},
                    "n": {"type": "integer", "min": 1}
                }
            },
            "specificity": {
                "type": "dict",
                "schema": {
                    "value": {"type": "float", "min": 0, "max": 1},
                    "CI": {"type": "float", "min": 0, "max": 1},
                    "n": {"type": "integer", "min": 1}
                }
            },
            "sensitivity": {
                "type": "dict",
                "schema": {
                    "value": {"type": "float", "min": 0, "max": 1},
                    "CI": {"type": "float", "min": 0, "max": 1},
                    "n": {"type": "integer", "min": 1}
                }
            }
        }
    }
}
```

14.7 Define run configuration, send run request

For background, see EscrowAI multiple concurrent algorithm runs (see page 34).

After the algorithm owner and data steward have uploaded the various required artifacts, the algorithm owner can create a run configuration and submit a run request to the data steward for that run configuration.

- A *run configuration* is a unique combination of the algorithm version and the dataset version. You cannot have more than one run configuration for any algorithm/dataset version combination.
- A *run request* is based on a run configuration for EscrowAI to execute a particular algorithm/dataset version combination in a confidential computing container whose vCPU and RAM size you specify when you send the run request. You can create as many run requests as you like for any run configuration.

14.7.1 How algorithm runs are identified: version combinations

Each run configuration created by the algorithm owner must have a unique combination of dataset version and algorithm version. Together, this is the unique identifier for a particular run configuration.

14.7.2 Multiple concurrent algorithm runs

The algorithm owner can request multiple runs but submit only one request at a time for each individual run configuration. EscrowAI processes multiple runs concurrently, each run operating on its unique combination of dataset version and algorithm version.

Example

As algorithm owner, you might have three different versions of an algorithm that you want to test against a certain dataset version. You create three different run configurations, each with the same dataset version but with the unique algorithm versions.

dataset version 1	algorithm version 1	Unique run configuration #1
	algorithm version 2	Unique run configuration #2
	algorithm version 3	Unique run configuration #3

You then submit run requests for all three unique run configurations, which EscrowAI processes concurrently after the data steward initiates the runs.

Continuing the example, after a request to start run configuration #1, you need to wait until that run finishes before you can request a new run for it. Likewise, the data steward can initiate the new run at the algorithm owner's request, but only after that run configuration's previous run has completed.

14.7.3 Planning

Before making a run configuration and any run requests, consider these planning details.

14.7.3.1 Run configuration: algorithm version and dataset version

For the run configuration, identify the version of the dataset and the version of the algorithm that you will create a run configuration for.

14.7.3.2 Run configuration: active, non-expired dataset SAS URL

The data steward creates an Azure SAS URL so that your algorithm can access the necessary datasets for your run. The SAS URL has an expiry date.

If the SAS URL has reached the expiry date, you cannot create a run configuration for the desired dataset.

Check with the data steward that the SAS URL is still active or needs to be refreshed.

14.7.3.3 Run request: RAM size and number of vCPUs on run request

For the run request, consider the number of vCPUs and RAM size that you want for the run. These settings depend on the type of confidential computing for the project.

14.7.3.4 Run request: optional runtime parameters

If you have designed your algorithm to take advantage of EscrowAI's externalized runtime parameters, you can specify the name/value pairs of those parameters when you create your run request. For more details and examples, see Optional runtime parameters (see page 29).

14.7.4 Create run configuration

See the background details about run configurations and run requests, including planning for both: Define run configuration, send run request (see page 94).

1. Go to the desired project.
2. Scroll to find **RUN CONFIGURATION** on the lower left.

3. Enter the metadata.

   The confidential compute technology is populated with the type of technology configured when the project was created for you.

4. Select the **Run Artifacts** for this run configuration: **Algorithm Version** and **Dataset Version**.

   If you have already created a run configuration for this combination of algorithm and dataset, an error is displayed.

   If the data steward's SAS URL for the desired dataset is expired, the dataset version is greyed out and shown as **Expired**. Contact the data steward to refresh the dataset URL.

5. Click **SUBMIT** to save.

Result: The run configuration is created and its status set to Pending.

14.7.5 Send run request

See the background details about run configurations and run requests, including planning for both: Define run configuration, send run request (see page 94).

1. Go to the desired project.
2. Scroll to find **RUN CONFIGURATION** on the lower left.
3. Click **Send Run Request**.
4. If you are using EscrowAI optional runtime parameters, enter the parameter names and values.

   The parameter name must conform to the following regular expression:
   `/^[a-zA-Z][a-zA-Z0-9]*([_\-\/.S][a-zA-Z0-9]+)*$/g`

   The parameter value is freeform.

5. Select the desired number of vCPUs and associated RAM.

   The available choices depend on the type of confidential computing technology established for the project when the project was created. The list of available settings also takes into consideration the needs of EscrowAI's processing the run and any limitations put in place by the cloud service provider.

6. Click **SEND RUN REQUEST**.

Result:

- The status changes to **Run Requested**.
- An email message with the run request is sent to the project's data steward.

The authorized data steward reviews the request and either initiates the run or cancels the request.

When the DS initiates the run, the status changes to **In Progress**.

If the DS rejects the run, the AO receives a System Notification[42].

42 https://beekeeperai.atlassian.net/wiki/spaces/EVDUM/pages/694322655/EscrowAI+home+page#Page-Elements

14.8 View run in progress

For details, see View the status of a run in progress (see page 81).

14.9 Get run results

EscrowAI run results have several parts under these headings:

- **Output**: The results here are for a trained model or for a validation report.
- **Performance Report**: This is to work with your experiment in MLflow.
- **Logs**: These are EscrowAI log messages specific to the run.
- **Run Parameters**: If you have taken advantage of EscrowAI's optional runtime parameters (see page 29) for this run, they are detailed here.

14.9.1 Download trained model

1. Go to the desired project.
2. Scroll to find and click **Run Configuration** on the lower left.
3. Scroll to the bottom to find **Run Details** for the latest run.
   You can also find your runs under **All Run Configurations**.
4. On the far right of the desired run under **Actions**, click **REPORT**.
5. Click the **Output** tab..
6. Click **DOWNLOAD MODEL**.

Result: A zipfile of the trained model is downloaded.

14.9.2 Make successive model training runs

See Make successive training runs (see page 99).

14.9.3 Get validation report

1. Go to the desired project.
2. Scroll to find and click **Run Configuration** on the lower left.
3. Scroll to the bottom to find **Run Details** for the latest run.
   You can also find your runs under **All Run Configurations**.

4. On the far right of the desired run under **Actions**, click **REPORT**.
5. Click the **Output** tab.
   The validation report is displayed.
6. On the lower right, click **CSV** or **JSON** to download the report in these formats

Result: The report is downloaded to a file named:

`<project_name><some_unique_identifier>.csv` or `.json`

14.9.4 View performance report for model training run

1. Go to the desired project.
2. Scroll to find and click **Run Configuration** on the lower left.
3. On the right, click the **Performance Report** tab.

Result: The MLflow window is displayed.

[This subsection should be a cross-reference to wherever Christer and Alan want to put the write-up about working in MLflow.]

14.9.5 View run logs

1. Go to the desired project.
2. Scroll to find and click **Run Configuration** on the lower left.
3. Scroll to the bottom to find **Run Details** for the latest run.
   You can also find your runs under **All Run Configurations**.
4. On the far right of the desired run under **Actions**, click **REPORT**.
5. Click the **Logs** tab.

Result: Log messages specific to this run are displayed.

14.9.6 See run-specific runtime parameters

For background, see Optional runtime parameters (see page 29).

1. Go to the desired project.
2. Scroll to find and click **Run Configuration** on the lower left.
3. Scroll to the bottom to find **Run Details** for the latest run.
   You can also find your runs under **All Run Configurations**.
4. On the far right of the desired run under **Actions**, click **REPORT**.
5. Click the **Run Parameters** tab.

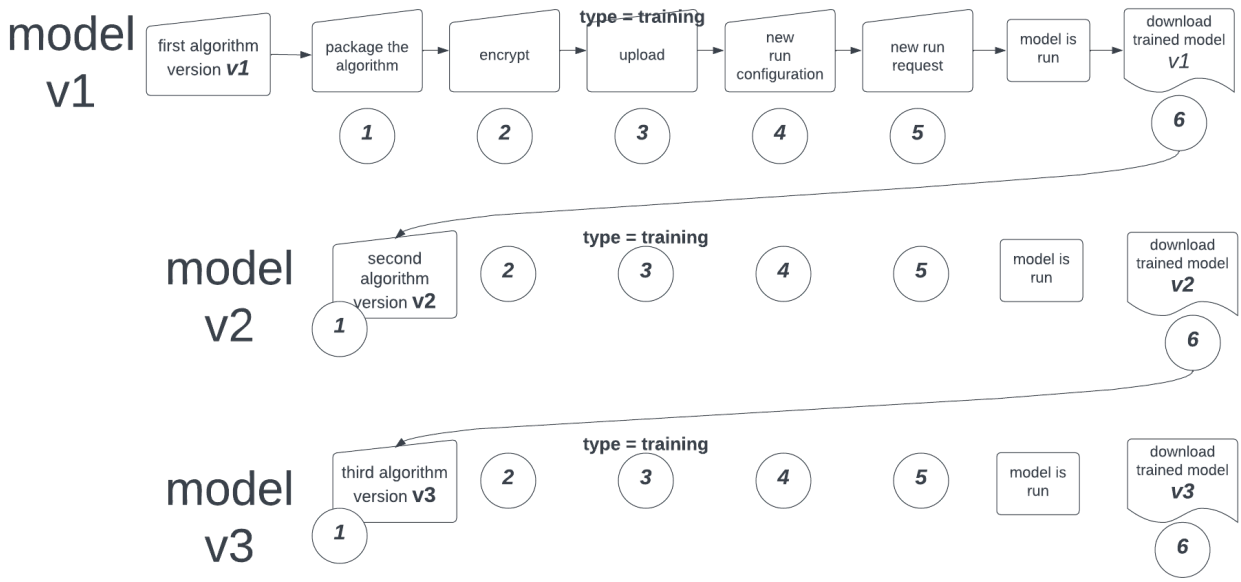
Result: Runtime parameters used in this run are displayed.

14.10 Make successive training runs

For successive model training runs, in EscrowAI, follow the same process as for your first algorithm.

14.10.1 Example

For example, suppose you want to make three training runs.



8 Example successive EscrowAI training runs

14.10.1.1 Add training algorithm v1 to EscrowAI

Prepare the first version of model.

1. Package your algorithm for upload to EscrowAI (see page 31).
2. Encrypt algorithm files (see page 86).
3. Upload algorithm with encryption (see page 87) with algorithm type set to ***training***.
4. Create run configuration (see page 95) for this combination of algorithm version and dataset version.
5. Send run request (see page 96).
6. Get run results (see page 97).

14.10.1.1.1 New version v2 training model

Follow the same steps in EscrowAI but add a *new version of the algorithm*.

- Make sure your algorithm package includes *the trained model v1* from the previous run.
- Upload a new algorithm package for the *new unique combination* of algorithm v2 and dataset.
- Create a *new run configuration* for the new combination of algorithm v2 and dataset.
- Send a run request for algorithm v2.

14.10.1.1.1.1 New version v3 training model

Follow these same steps, and so on, for as many training runs as you want.

15 Model building test cases

15.1 <BIGNOTE>

This should include all concrete ***TESTED*** models' algorithm code such as from Model building test cases[43] by Christer Smith.

15.2 </BIGNOTE>

43 https://beekeeperai.atlassian.net/wiki/spaces/TRAIN/pages/586022913/Model+building+test+cases

16 Glossary

16.1 A

16.1.1 Algorithm Owner

The developer of the mathematical algorithms for analyzing the data maintained by a Data Steward.

16.1.2 Attestation

The process of authenticating a Trusted Execution Environment instance. In Confidential Computing an attestation is the validation of a hardware signed report (an “attestation report”) of the measurements of the Trusted Computing Base.

16.1.3 Azure Blob Storage

Azure Blob storage is Microsoft's object storage solution for the cloud. Blob storage is optimized for storing massive amounts of unstructured data. A container organizes a set of blobs, similar to a directory in a file system. A storage account can include an unlimited number of containers, and a container can store an unlimited number of blobs.

16.1.4 Azure Subscription

An Azure subscription is a logical container used to provision resources in Azure

16.2 C

16.2.1 Compute Enclave

An ephemeral enclave that is initialized for the purpose of a confidential compute unit of work and destroyed after the confidential compute task ends.

16.2.2 Confidential computing

The protection of data in use by performing computation in a hardware-based, attested Trusted Execution Environment (TEE).

16.2.3 Confidential containers on Azure Container Instances (CC ACI)

Confidential containers on Azure Container Instances enable customers to run Linux containers within a hardware-based and attested Trusted Execution Environment (TEE). Customers can lift and shift their containerized Linux applications or build new confidential computing applications without needing to adopt any specialized programming models to achieve the benefits of confidentiality in a TEE. Confidential containers on Azure Container Instances protect data-in-use and encrypts data being used in memory. Azure Container Instances extends this capability through verifiable execution policies, and verifiable hardware root of trust assurances through guest attestation.

16.2.4 Confidential Virtual Machine (CVM)

The AMD Azure EPYC SEV-SNP[44] DCasV5 and ECasv5-series confidential VM series provides a hardware-based Trusted Execution Environment (TEE) with attestation capability by leveraging AMD SEV-SNP security features. Azure confidential VMs (CVMs) offer VM memory encryption with integrity protection, which strengthens guest protections to deny the hypervisor and other host management components code access to the VM memory and state.

See more information from Microsoft about CVM[45].

16.2.5 Container

A container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another.

16.2.6 Content Encryption Key (CEK)

A CEK is a private, synchronous data-encryption key[46] used to encrypt **and** decrypt data. You create your CEKs using EscrowAI's encryption tool or your organization's key manager. The CEK is used to encrypt a data set and the intellectual property within the algorithm.

16.3 D

16.3.1 Data Attestation Report

A report by the Data Steward affirming that the data set curated for the project meets the requirements of the data specification.

---

44 https://techcommunity.microsoft.com/t5/azure-confidential-computing/azure-confidential-vms-using-sev-snp-dcasv5-ecasv5-are-now/ba-p/3573747

45 https://techcommunity.microsoft.com/t5/azure-virtual-desktop-blog/announcing-general-availability-of-confidential-vms-in-azure/ba-p/3857974

46 https://csrc.nist.gov/glossary/term/data_encryption_key

16.3.2 Data Set Version

A snapshot in time of elements that make up a data set. The data set version will correspond to specific Data Specification and Data Attestation Report versions.

16.3.3 Data Specification

A document created by the Algorithm Owner that defines the population of the data set (e.g., the patient group) and the data elements needed for each member of the population, the form of each data element, and the form of the data set in its entirety. The data specification also includes the means of identifying the truth by which an inference is tested (as needed), and how the data set must be validated.

16.3.4 Data Steward

Legal entities with the responsibility/liability to protect privacy information in a legal and ethical way.

16.4 E

16.4.1 Enclave

An *enclave* is a protected memory region that provides confidentiality for data and code execution. It is an instance of a Trusted Execution Environment (TEE) which is usually secured by hardware.

16.4.2 Enclave Agent

A piece of code that runs in an enclave and checks for carefully controlled tasks to run within an enclave.

16.4.3 Encryption

Encryption is a way of hiding data from unauthorized parties by transforming it into a secret code that only the intended recipients can decipher.

16.5 I

16.5.1 Intel SGX

Intel Software Guard Extensions or Intel SGX helps protect data in use via application isolation technology.

16.6 J

16.6.1 Just in time model decryption

Pre-signed models are decrypted just in time by the model owner releasing sealed keys after they prove that the correct code is running in an authorized enclave.

16.7 K

16.7.1 Key Encryption Key (KEK)

The KEK is the cryptographic key that is used for the encryption of the CEK (“wrapping”) to provide confidentiality and protection for that key, allowing it to be sent to EscrowAI as ciphertext. The KEK[47] is the public half of an asynchronous key pair. In a public-key encryption system, anyone with a public key can encrypt data yielding a ciphertext, but only those with the corresponding private key can decrypt the ciphertext to obtain the original data. In EscrowAI, the private half of this key pair (the half that can decrypt) is retained in the key vault after generation and is only available for decrypting the wrapped CEK within an attested Trusted Execution Environment initiated from the associated project.

16.8 L

16.8.1 Linux VM

Azure Virtual Machines are image service instances that provide on-demand and scalable computing resources with usage-based pricing. Linux is a family of operating systems commonly used on servers.

16.8.2 Managed Identity

Managed identities provide an identity for applications to use when connecting to resources that support Azure Active Directory (Azure AD) authentication

47 https://csrc.nist.gov/glossary/term/kek

16.9 M

16.9.1 Manifest

A description of what assets a test run includes. Manifest is an adapted term from “ship’s manifest” which means a list of the shipments or cargo that a vessel is carrying

16.10 N

16.10.1 Network Interface

A network interface connects a virtual machine with other services in the virtual network.

16.10.2 Network Security Group

A network security group filters traffic to and from Azure resources

16.10.3 Node

A confidential computing machine within the data steward's security perimeter that has been joined into an enclave and is connected to EscrowAI core infrastructure. The computer communicates with EscrowAI through enclave agent software and has capabilities to setup secure enclaves for data science tasks performed against PHI data.

16.11 P

16.11.1 Package manager

A package manager or package-management system is a collection of software tools that automate the process of installing, upgrading, configuring, and removing content on a computer in a consistent manner.

16.11.2 Pre-encrypted Model

A machine learning model that is signing using a mechanism outside of the Beekeeper ecosystem.

16.11.3 Project

A canonical name used across sites to define a relationship between parties collaborating on a joint statement of work. Typically this involves a particular algorithm and a corresponding data set.

16.12 R

16.12.1 Resource Group

A resource group is a container that holds related resources for an Azure solution.

16.13 S

16.13.1 Sealed Enclave Key

A key that has been encrypted in such a way that only a single enclave can decrypt the key.

16.13.2 Secure Enclave

A protected black box where confidential computing occurs

16.13.3 SSL

[description]

16.13.4 Site

A canonical name for a counter party (e.g., `UCSF` for the University of California, San Francisco)

16.13.5 Storage Account

An Azure storage account contains all Azure Storage data objects, including blobs, file shares, queues, tables, and disks.

16.14 T

16.14.1 Trusted Computing Base

Totality of protection mechanisms within a computer system, including hardware, firmware, and software, the combination responsible for enforcing a security policy.

16.14.2 Trusted Execution Environment

A Trusted Execution Environment (TEE) is an environment in which the executed code and the data that is accessed are physically isolated and confidentially protected so that no one without integrity can access the data or change the code or its behavior. A TEE has three primary attributes: data integrity, data confidentiality, and code integrity. Four additional attributes may be present (code confidentiality, programmability, recoverability, and attestability) but only attestability is strictly necessary for a computational environment to be classified as Confidential Computing.

16.15 U

16.16 V

16.16.1 Virtual Machine

A computer system created using software on one physical computer in order to emulate the functionality of another separate physical computer.

16.16.2 Virtual Network

An Azure Virtual Network or VNet is a fundamental building block for building a private network in the cloud.

16.16.3 VM Disk

A physical disk, such as a hard driver or solid-state driver that is connected to a virtual machine.

16.17 W

16.17.1 Wrapped Content Encryption Key (WCEK)

A WCEK is a CEK that has been encrypted (“wrapped”) by a KEK. The WCEK is ciphertext (encoded information).

16.18 Z

16.18.1 Zero Trust Architecture

Zero Trust[48] is a security model, a set of system design principles, and a coordinated cybersecurity and system management strategy based on an acknowledgement that threats exist both inside and outside traditional network boundaries. The Zero Trust security model eliminates implicit trust in any one element, node, or service and instead requires continuous verification of the operational picture via real-time information from multiple sources to determine access and other system responses. In essence, a Zero Trust Architecture allows users full access but only to the bare minimum they need to perform their jobs. If a device is compromised, zero trust can ensure that the damage is contained. The Zero Trust Architecture security model assumes that a breach is inevitable or has likely already occurred, so it constantly limits access to only what is needed and looks for anomalous or malicious activity. Zero Trust Architecture embeds comprehensive security monitoring; granular risk-based access controls; and system security automation in a coordinated manner throughout all aspects of the infrastructure in order to focus on protecting data in real-time within a dynamic threat environment. This data-centric security model allows the concept of least-privileged access to be applied for every access decision, where the answers to the questions of who, what, when, where, and how are critical for appropriately allowing or denying access to resources based on the combination of several factors.

48 https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/

17 EscrowAI Help Center

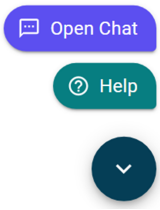
Online help for EscrowAI (this user manual) and other information resources are available via the **Quick actions** on any page in EscrowAI. From the help center, you can also submit questions/issues to BeeKeeperAI technical support.

17.1 Login to the Help Center

1. In EscrowAI, in the lower right, click the **Quick actions** icon.

9 Quick actions icon

2. Select **Help**.



10 Expanded action menu

You are redirected to the EscrowAI Help Center in a new browser tab.

17.2 View all resources on the Help Center

Resources within the Help Center are arranged by tiles on the home page. Click on any tile to access that resource. To see all the resources available on the help center, click on the BeeKeeperAI logo in upper left corner of the page.

After logging in to the help center, you might be directed to the EscrowAI User Manual directly, rather than the Help Center home page. Click the BeeKeeperAI logo to go the home page.

17.3 Search the Help Center

You can search for information across all resources in the Help Center, including the EscrowAI User Manual.

1. Login to the EscrowAI help center.

2. Go to the help center main page.
3. In the displayed **Search** box, enter the desired keywords.
4. Hit **Return**.

Results: Matching hits are displayed.

17.4 Contact BeeKeeperAI technical support

You can contact BeeKeeperAI technical support if you do not find the information you are looking for or encounter a technical issue.

1. Login to the EscrowAI help center.
2. Go to the help center main page.
3. In the lower right, click **Help**.
4. Enter the details in the displayed form:
    - So BeeKeeperAI can best help you, be as detailed as possible.
    - Attach any relevant files you think can help isolate and resolve the issue.
5. At the bottom of the form, click **Send**.
