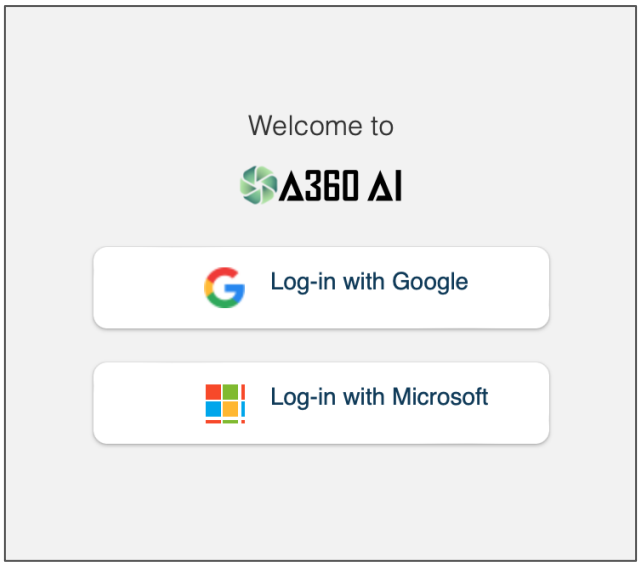
A360 AI

[User Manual]

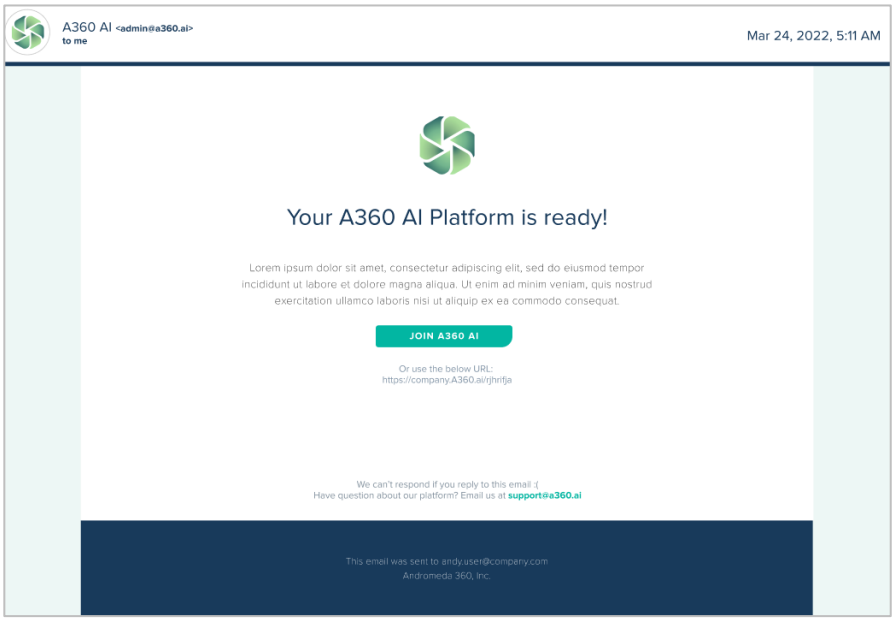
Table of Contents

HOW TO LOG IN

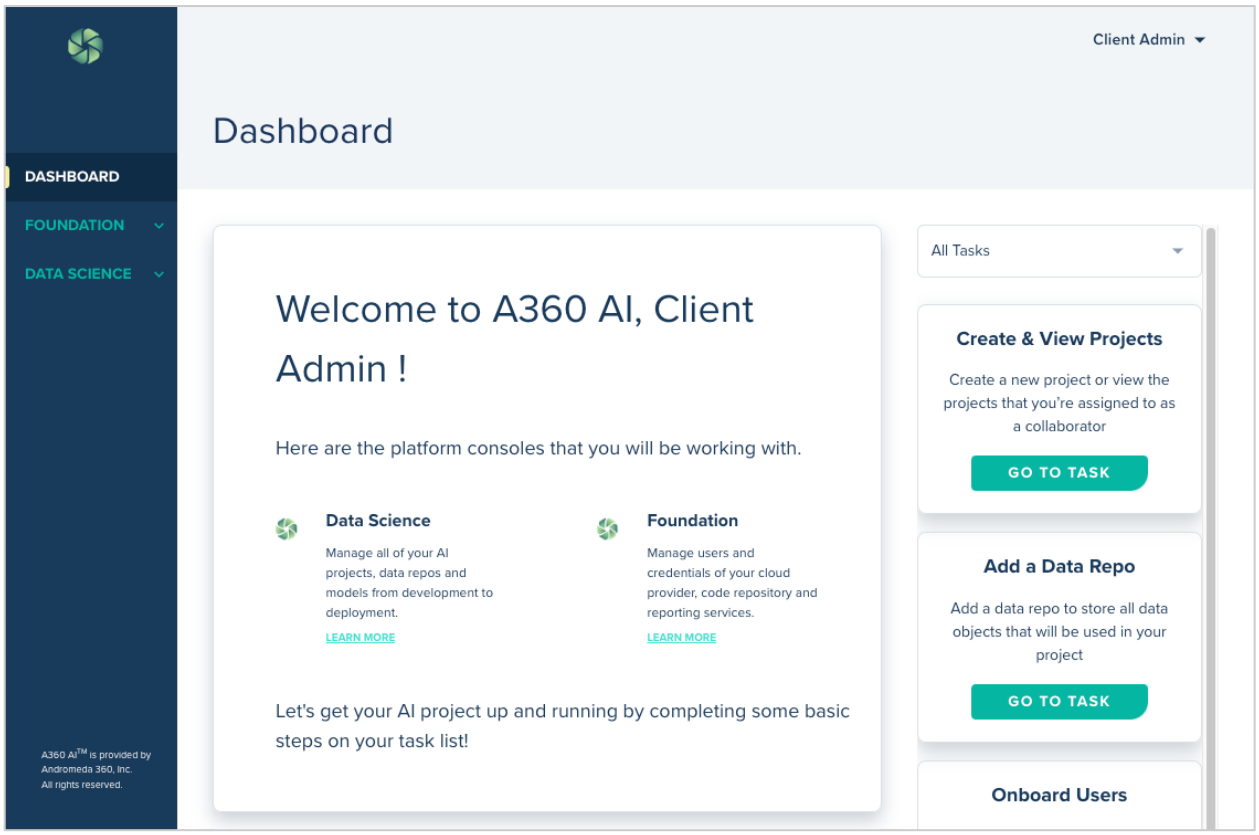


A360 AI supports two authentication mechanisms to access the platform. Users can authenticate via Microsoft or Google single sign-on (SSO). Users receive an invitation email inviting them to the platform once the user has been successfully added using the User Management pane by the platform administrator.

Example Invitation Email:



DASHBOARD



The Dashboard serves as the landing page for all users. The center section showcases summaries of each console the user has access to. The right most section contains a Task list that shows actions a user may take within the platform with each link directing them to a pertinent topic.

FOUNDATION CONSOLE

What is the Foundation Console?

The Foundation console is the central administrative hub for the A360 AI platform and is a dedicated user interface for your organization's SaaS administration needs. Activities available in the console include onboarding users to the platform, assigning roles, and managing access. In addition, the console allows for the secure setup and configuration of the integrations supported by A360 AI.

Note, in future releases, the Foundation console will grow in scope and function to cover audit logs, billing, data retention policies, platform health & usage, in-app notifications, and much more.

Role-Based Access Control (RBAC)

Within the Foundation console, administrators can assign one-to-many roles to user groups containing one-to-many platform users. Roles determine what access users have on the platform.

RBAC hierarchy is as follows:
User -> User Group -> Role

User - An individual involved in data science or machine learning workflows that requires access to the platform to perform some activity.

User Group - A user group defines a user's role-based access and credentials. Credentials are NOT assigned directly to users and are assigned to user groups. User groups can also be used to logically separate users into teams representing your organization's structure.

Roles - A role defines the user group access and permissions within the platform.

Role Definitions

Client Admin - Client Admin is the administrator roll. Users with this role have access to the Foundation console to conduct user management and manage credentials for all other roles. Client Admins cannot access consoles other than Foundation without assigning themselves another role.

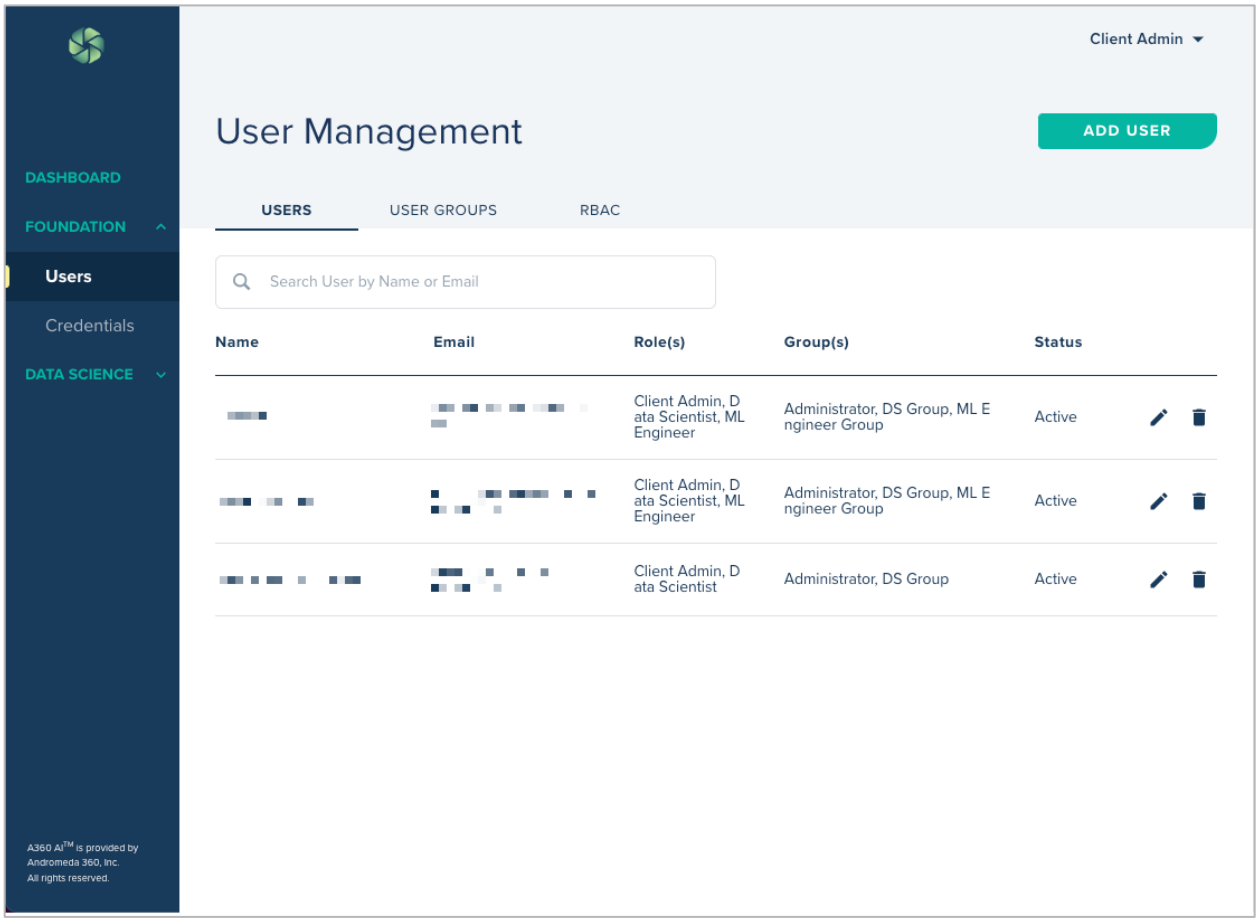
Data Scientist - Data Scientists have access to the Data Science console to participate in the projects to which they are assigned. Within those projects, the Data Scientist has access to Data Sources, Models, monitoring of those models, and any Jupyter Notebook asset.

ML Engineer – In future releases, users with the ML Engineer role will have access to the Model Ops console to operationalize models. ML Engineers can package models, manage deployment, and monitor them in production.

Permissions Matrix

Role	Pane	Description	Permissions
Client Admin	Users	Client Admins can add, delete, and assign role-based access to platform users	Create, edit, and delete
	User Groups	Client Admins can create user groups to manage access utilizing a group rather than on an individual user level	Create, edit, and delete
	Credentials	Client Admins can create connections to data sources, API tokens, deployment targets and monitoring services, and manage group-based access	Create, edit, and delete
Data Scientist	Projects	Data Scientists can create and manage projects which their user group is assigned to	Create, edit, and delete
	Data	Data Scientists can access to data sources which their user group is assigned to	Create, edit, and delete
	Notebook Servers	Data Scientists can spin up notebook servers and associate them to projects	Create, edit, and delete
		Get credential	
		Add model	
		Edit model	
		List data repos	
	Models	Data Scientists can develop machine learning models to drive business insights	Create and edit
		Review model	
		Package model	
		Publish model	
	Monitoring	Data Scientists can monitor model performance metrics via dashboards	View

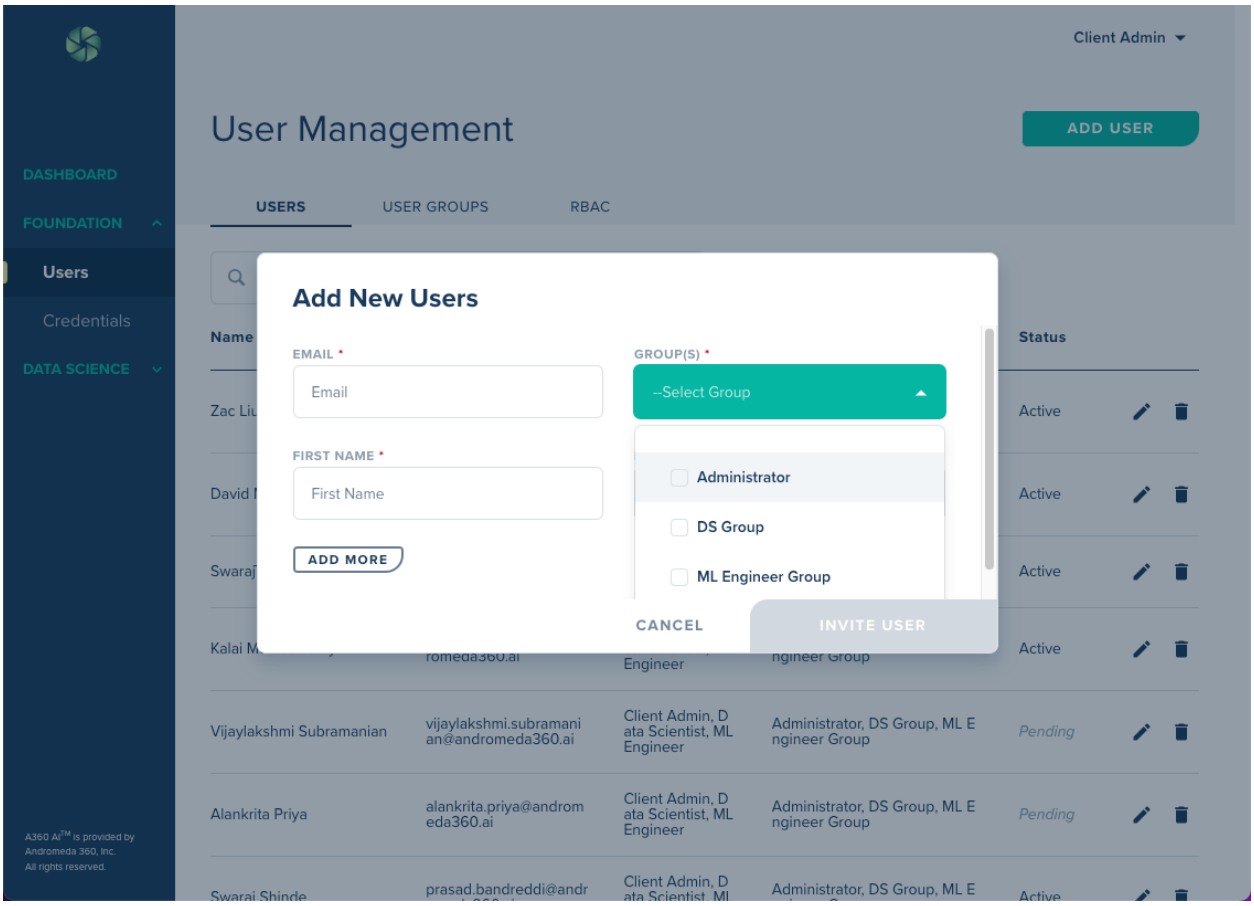
User Management Pane



The User Management pane is displays platform users in tabular form. The fields included are:

- Name - The first and last name of the user.
- Email - The email of the user.
- Role - Based on user groups assigned, the user inherits roles (Data Scientist or Client Administrator).
- Group(s) - The user group(s) associated with the user.
- Status
    - Active - Users who have accepted their invitation to join the platform.
    - Pending - Users who have been invited to the platform but not yet accepted.

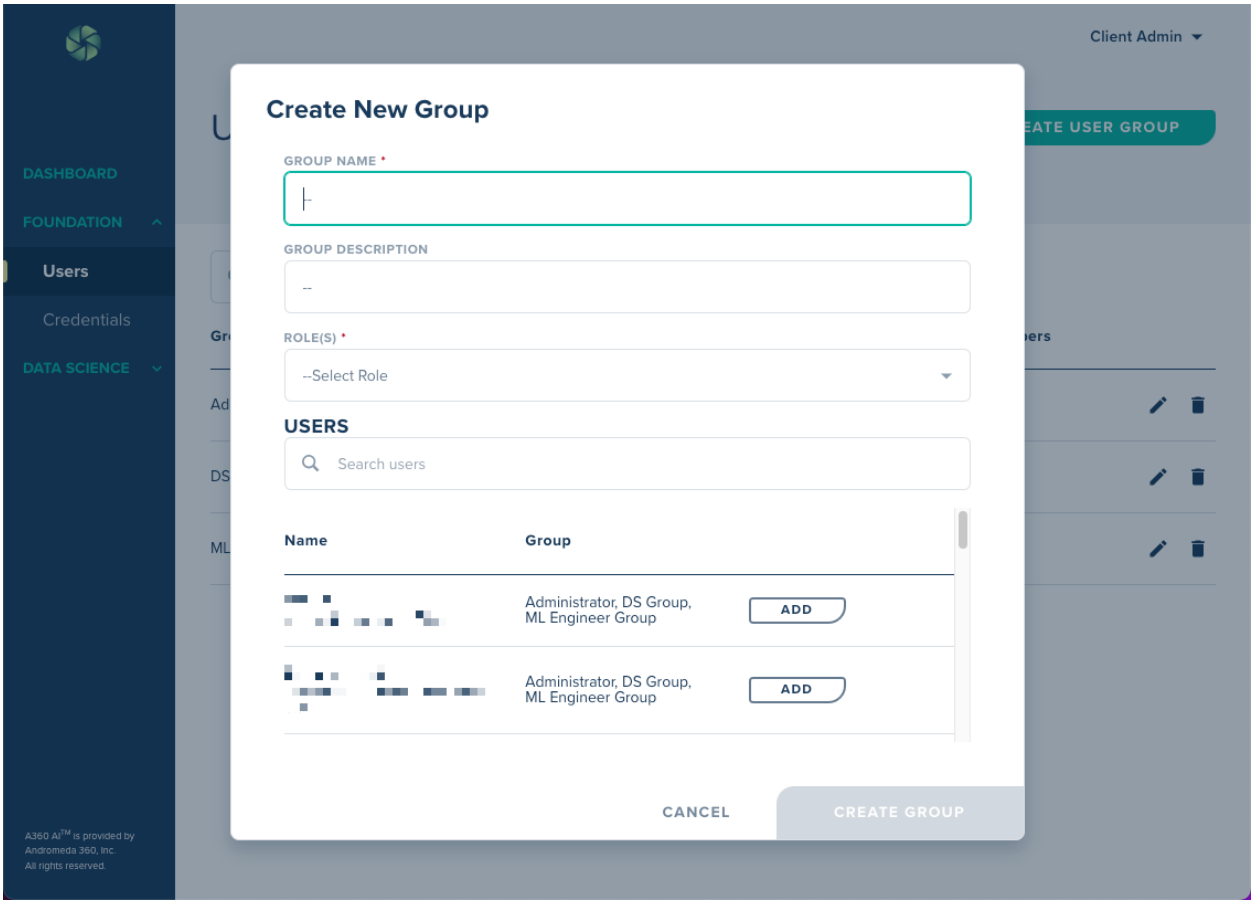
Add Users



To add a new user to the platform, the users with the client administrator role must fill out a short form containing the following details:

- Email (required)
- First Name (required)
- Last Name (required)
- Group (required) - The user group(s) the user will be assigned to.

Create User Groups

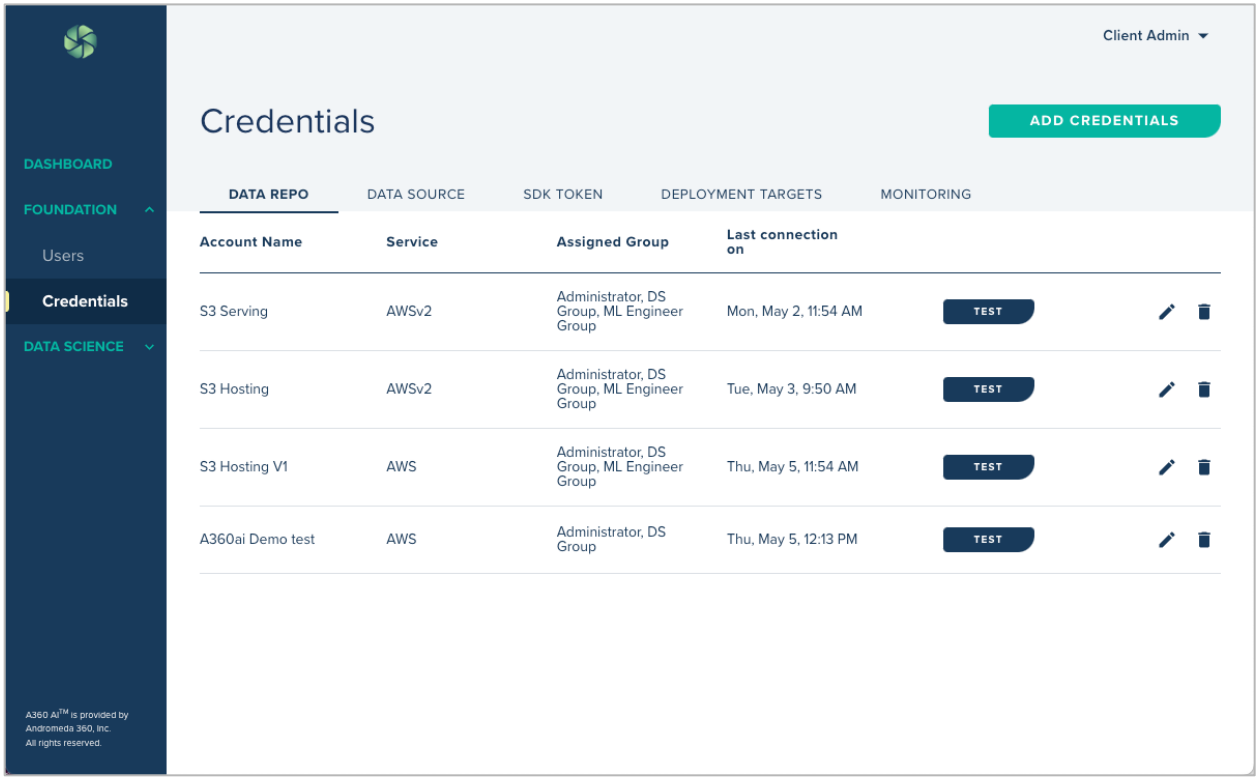


User groups must be created before any user can access the platform. A default group named "Administrator" is created for the first user of the platform with the role: Client Admin. Best practice is to create and manage user groups for business users containing the right level of permission and access controls.

To create a new group, choose Manage Group > Create Group. The administrator is presented with a form requiring the following information:

- Group Name (required)
- Group Description
- Role (required)
- List of users to Add to user group with the ability to search for specific users

Credentials

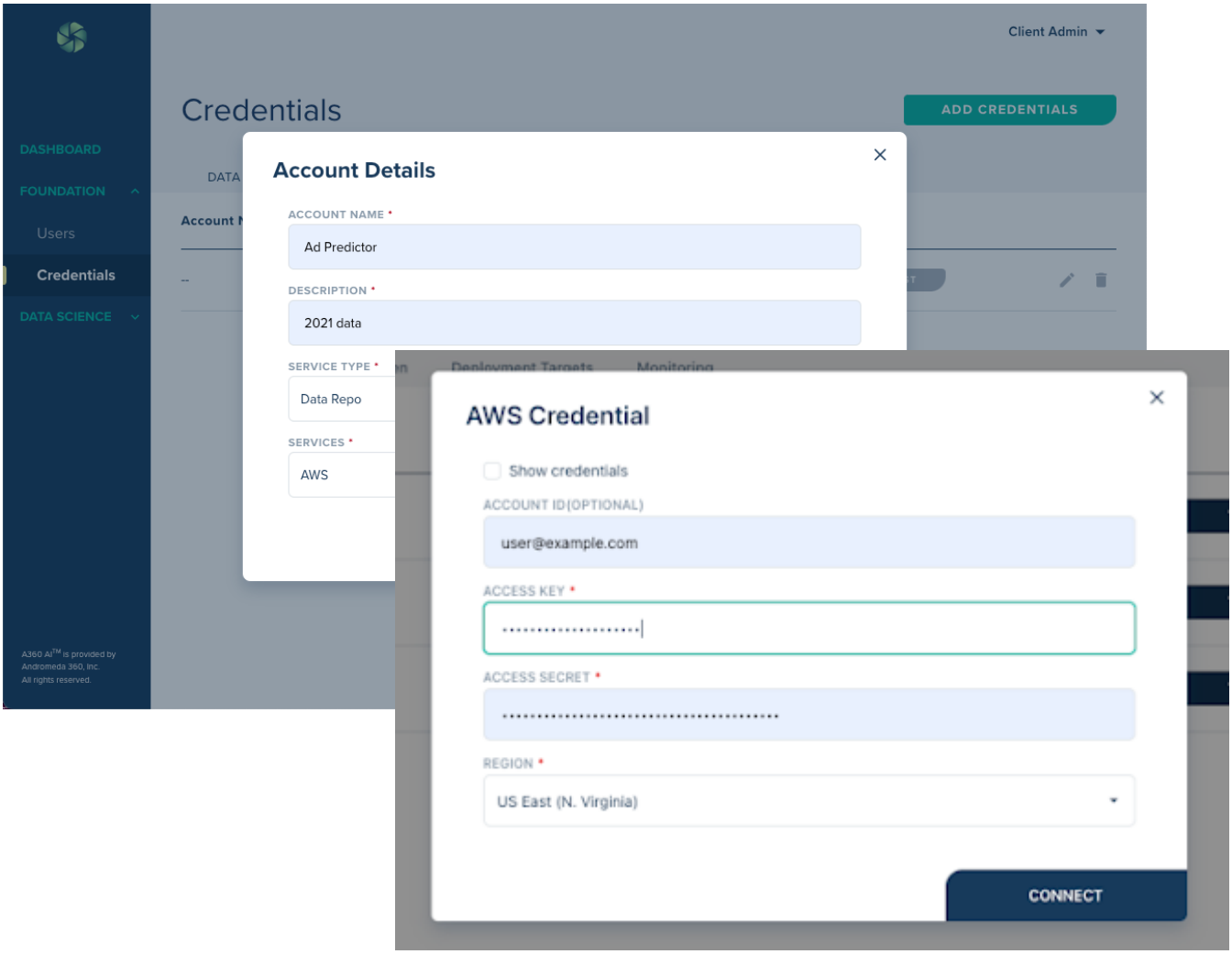


The Credentials pane displays credentials in tabular form across five categories:

- **Data Repo** - Credentials to connect to AWS S3 bucket for saving model artifacts. Credentials are automatically created for a data repo when creating a new project.
- **Data Source** - Credentials to connect to external data sources A360 AI can consume such as AWS S3 buckets.
- **API Token** - Create and manage API tokens.
- **Deployment Targets** - Credentials to deployment target environments such as EKS clusters.
- **Monitoring** - Visualize your deployed model's performance using a personalized Grafana dashboard.

Note: *Each service requires different inputs to be able to authenticate and successfully integrate with A360 AI. These differences are detailed in the following sections.*

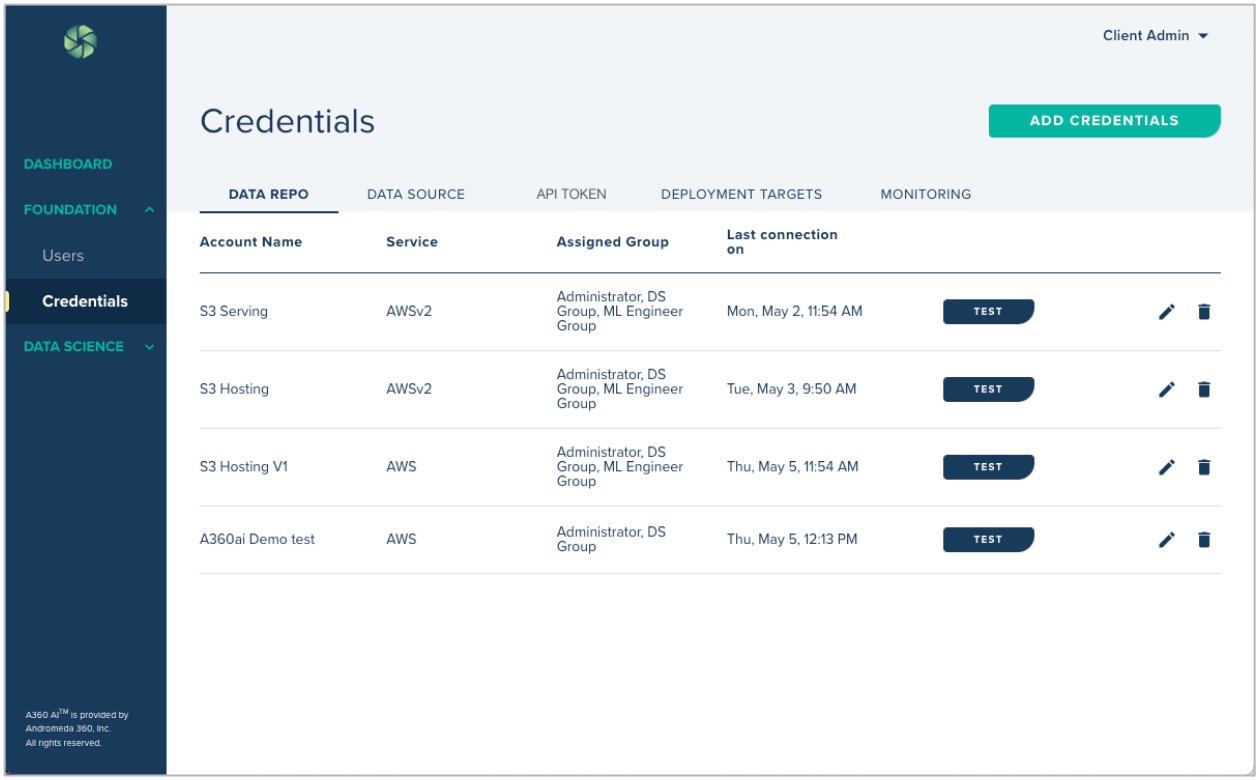
Adding Credentials



When adding credentials, the user is presented with a form containing the following fields:

- Account Name (required) - The name of the account.
- Description (required) – A short description of the account.
- Service Type (required) – Choose Data Source, API Token, Deployment Target, or Monitoring
- Services (required) - Auto-populates with the supported service after the service type has been selected.

Data Sources

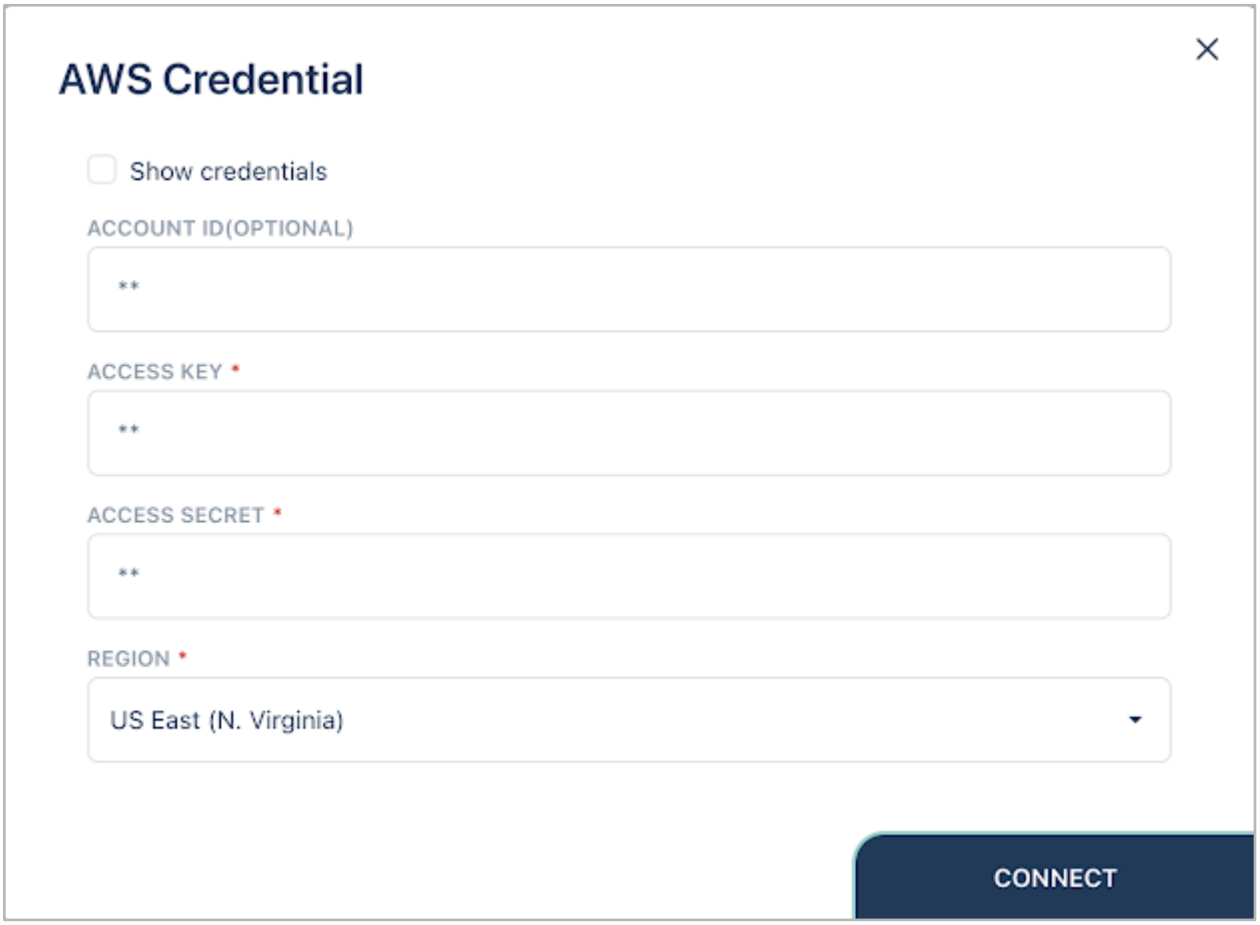


Data Source list view shows credentials added to access data sources. The following fields are present and sortable:

- Account Name
- Service
- Assigned Group - Only user groups assigned to the credential will have access in other consoles.
- Last connection on - The last connection datetime when the credential was tested.

The administrator can test the connection to see if the credential is still valid or the service is online. The user can edit or remove the credential by selecting the two most-right positioned icons per credential row.

Adding an AWS S3 Credential



Field options:

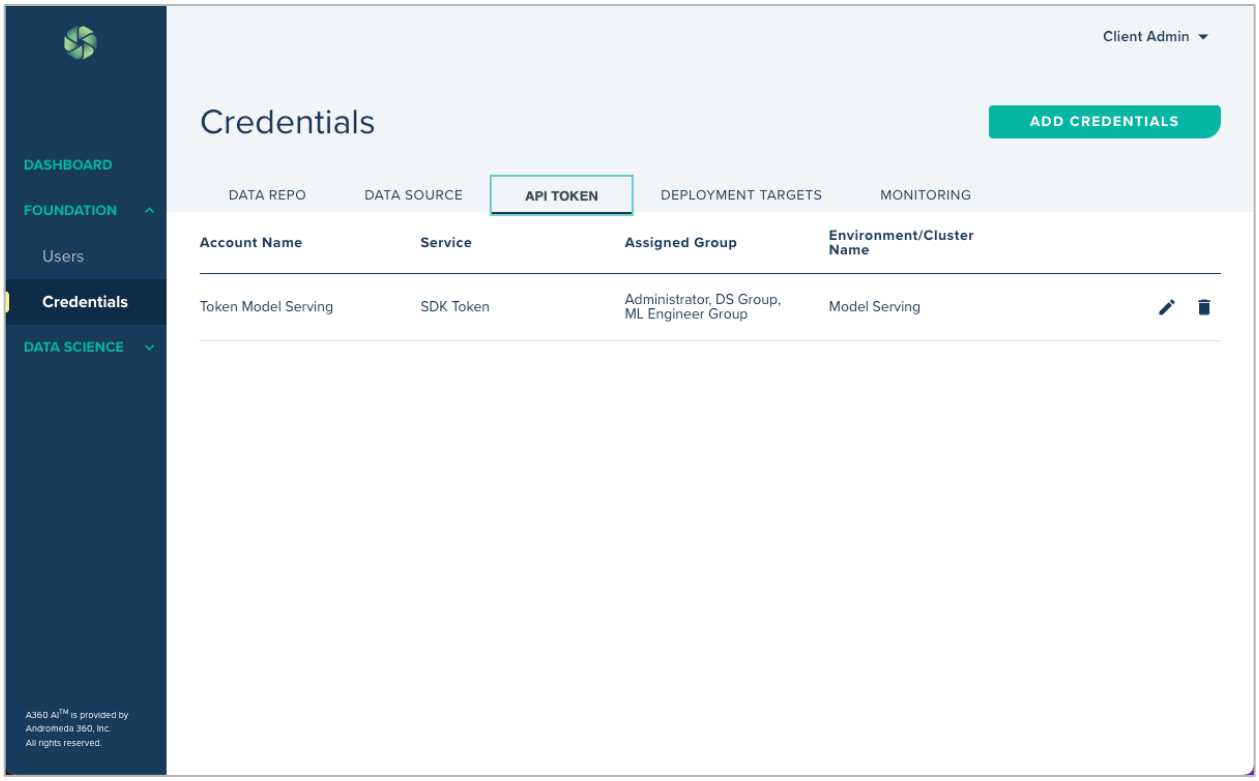
- Account ID - The AWS account ID number.
- Access Key (required) - Long-term credentials for an IAM user or the AWS account root user.
- Access Secret (required) - The secret for the access key acting as complete credential.
- Region (required) - The AWS account resource region.

Note: *The access key and access secret must be associated with complete access permissions to the S3 bucket.*

- Required permissions for connection to S3:

```
List*
Get*
CreateBucket
PutBucket*
DeleteBucket
PutObject*
DeleteObject*
```

API Tokens

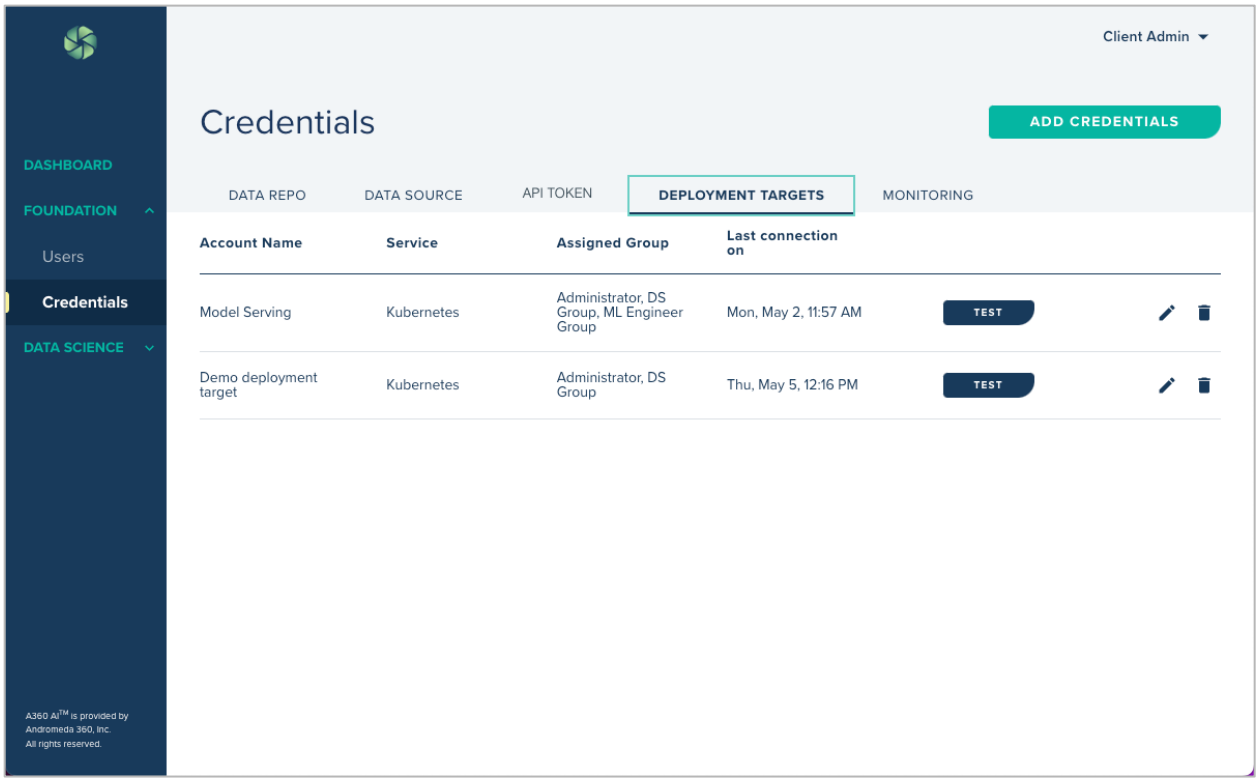


Any credential added for API tokens will be visible in the API Token list view. The following fields are present and sortable:

- Account Name
- Service
- Assigned Group - Only user groups assigned to the credential will have access in other consoles.
- Environment/Cluster Name - The environment which the token is designated for.

The user can edit or remove the credential by selecting the two most-right positioned icons per credential row.

Deployment Targets

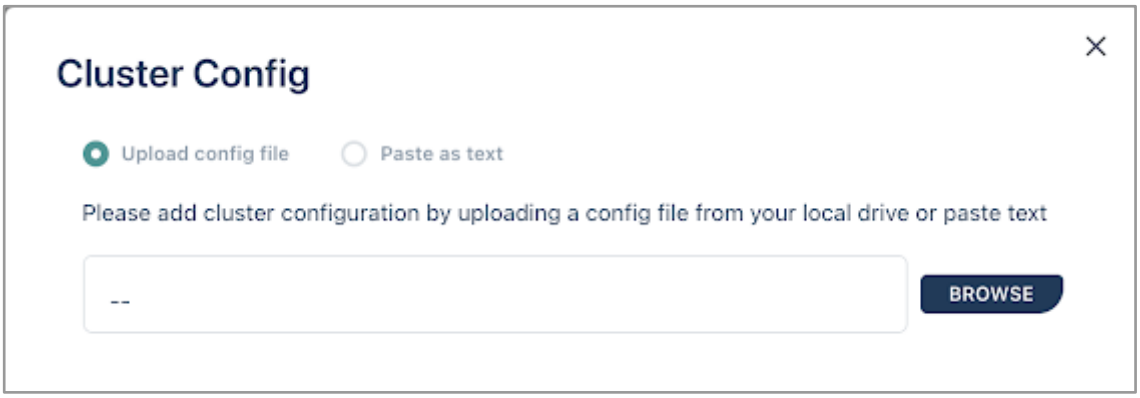


Any credential added with for deployment targets will be visible in the Deployment Targets list view. The following fields are present and sortable:

- Account Name
- Service
- Assigned Group - Only user groups assigned to the credential will have access in other consoles.
- Last Connection On - The last connection datetime when the credential was used or tested.

The administrator can test the connection to see if the credential is still valid or the service is online. The user can edit or remove the credential by selecting the two most-right positioned icons per credential row.

Configuring an EKS Cluster



Required Inputs:

- Kubeconfig File via file upload or pasted as text

Note: *The access key and secret which are included in the KubeConfig must belong to an IAM user in the model serving AWS account, and (1) this IAM user needs to have the access to the EKS model serving cluster, and (2) if RBAC is enabled in the Kubernetes, the user should have the permission to create/bind cluster role and manage the Bitnami controller in* kube-system *namespace, and permission to manage new namespace and resources in the new namespace (for model).*

How to Generate the Kubeconfig File From AWS:

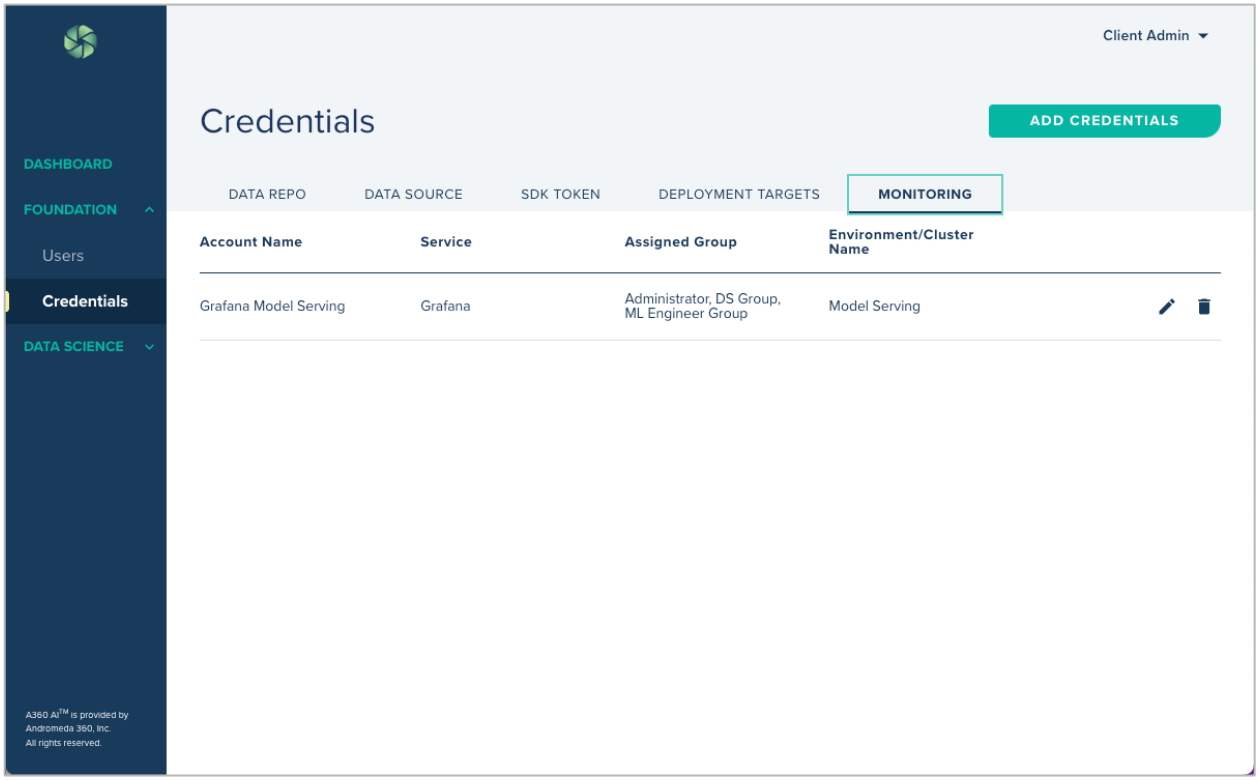
- Kubeconfig file template

```
apiVersion: v1
clusters:
- cluster:
    server: replace-with-actual-value
    certificate-authority-data: replace-with-actual-value
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: aws
  name: aws
current-context: aws
kind: Config
preferences: {}
users:
- name: aws
  user:
    exec:
```

```
apiVersion: client.authentication.k8s.io/v1alpha1
command: aws
args:
  - eks
  - get-token
  - --cluster-name
  - replace-with-actual-value
env:
  - name: AWS_DEFAULT_REGION
    value: replace-with-actual-value
  - name: AWS_ACCESS_KEY_ID
    value: replace-with-actual-value
  - name: AWS_SECRET_ACCESS_KEY
    value: replace-with-actual-value
```

- The client admin is responsible for generating the Kubeconfig file from the template above by replacing the value for *replace-with-actual-value*
- For the *replace-with-actual-value* for `AWS_DEFAULT_REGION`, `AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY`, which are the credentials of IAM user model-serving in the model serving AWS account. The owner of this model serving AWS account needs to share those credentials with the client admin.
- For the *replace-with-actual-value* for `server`, `certificate-authority-data`, and `--cluster-name`, the owner of this model serving AWS account can run `aws eks update-kubeconfig --region $clusterRegion --name $clusterName` and get those values from the generated Kubeconfig file, which is usually `~/.kube/config`

Monitoring



Any credential added for monitoring will be visible in the Monitoring list view. The following fields are present and sortable:

- Account Name
- Service
- Assigned Group – Only user groups assigned to the credential will have access in other consoles.

The user can edit or remove the credential by selecting the two most-right positioned icons per credential row.

Connecting to Grafana

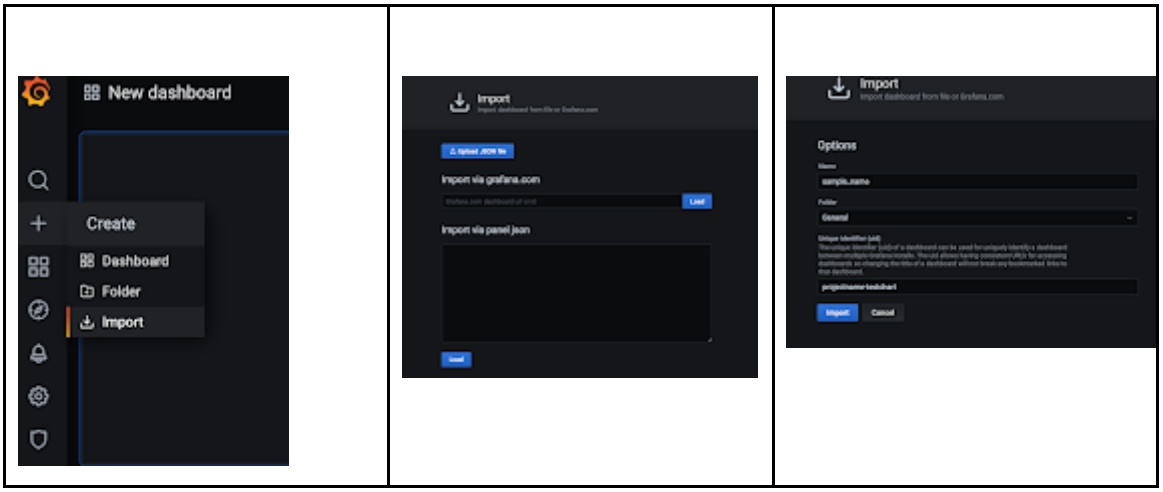
If your organization will deploy models on environment outside of an A360 AI instance, such as an EKS Cluster or On-Prem, the following steps are required to enable resource usage and availability monitoring and using Grafana. Grafana must be installed and configured on your target

environment before taking these steps to integrate the Grafana service on your instance to the A360 AI platform.

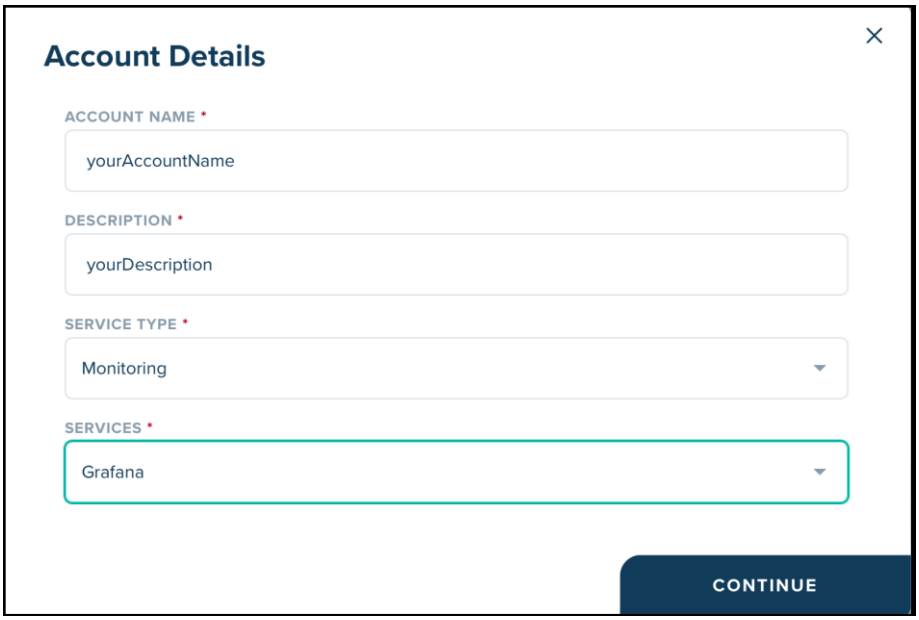
1. Locate the two Grafana JSON files that were included with your onboarding materials:
    - The json files are named: `a360ai_resource_usage` and `a360ai_availability`. These represent the dashboard configurations files required for Step 2.

`Resource Usage`
`UID: a360ai_resource_usage`
`Dashboard Name: resource_usage`
	`Availability`
	`UID: a360ai_availability`
	`Dashboard Name: availability`

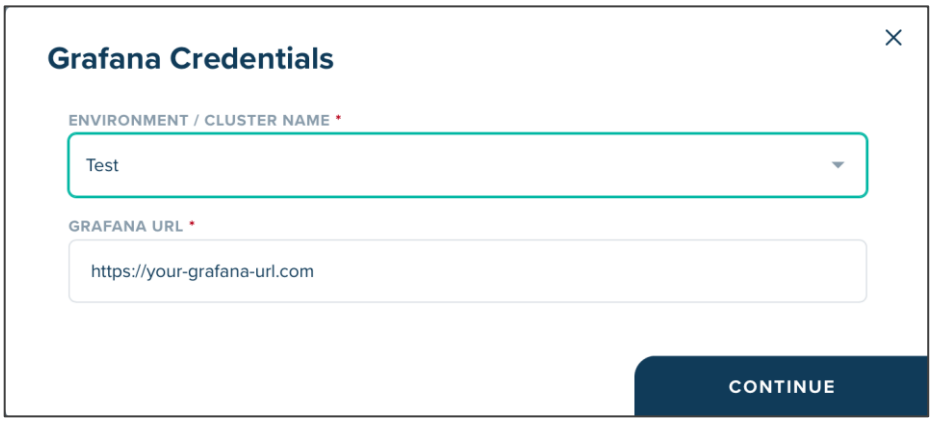
2. In your Grafana instance on the deployment target environment: Once the JSON file is downloaded or copied, go to '+' sign to Create > Import. From the Import pane either Upload or Paste your JSON file in the text box. Enter UID, Name, Folder to complete the import step.



3. Go to the Foundation console and select the Credentials pane. While in the Credentials pane on Add Credentials to bring up a new modal where you will fill out your Account Details:
    - Account Name (required): A unique name for your Grafana account in A360 AI.
    - Description (required): A description of the Grafana account.
    - Service Type (required): Select Monitoring as the Service Type.
    - Services (required): Select Grafana.



4. Click Continue to enter your Grafana Credentials:



5. Select your Environment/Cluster name from the available environments in the drop-down menu. All environments associated with your A360 AI instance will appear.
6. Enter your Grafana base URL:
    - Example Grafana base URL: `https://grafana.ms-dev.a360ai.solutions/`
7. Navigate back to the Monitoring page under the Data Science console to confirm successful integration.

Note: *The default option for A360 AI authentication with Grafana is through the same IDP provided during customer onboarding. A360 AI will use Microsoft or Google SSO to authenticate unless*

explicitly stated it is not required. If it is not required, the customer must set anonymous access on the Grafana dashboards.

If your organization elected to have A360 AI host model deployment environments, this section is not necessary. Model monitoring will be visible in the Data Science console and a Grafana dashboard enabled without the need to manage credentials.

Github App

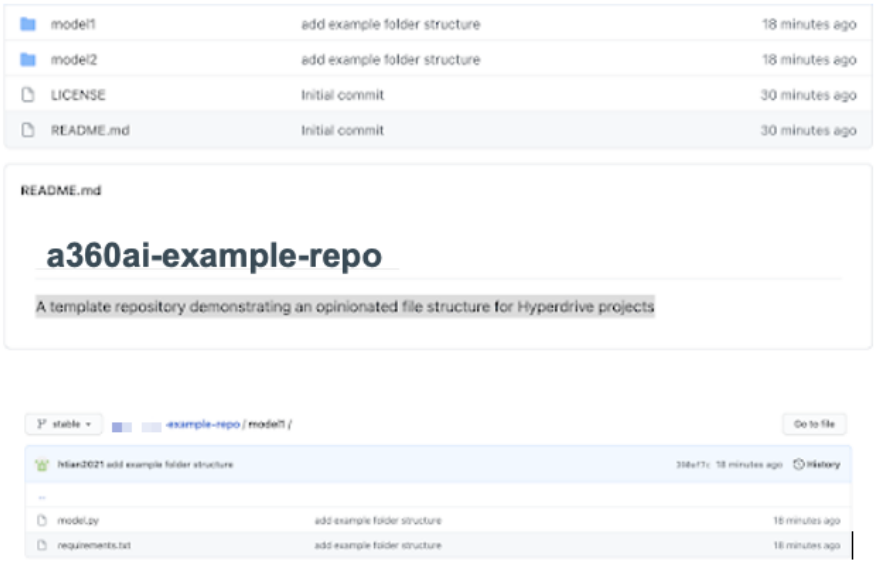
For enterprise customers who wish to integrate their GitHub repositories with A360 AI, your organization's GitHub admin must perform the following steps:

1. Login to your GitHub account.
2. Install the A360 AI GitHub app by going to the following URL and clicking Install or Configure:

   `a360-read-app`
3. Select the appropriate GitHub repository and grant read-only permission.

Andromeda 360 makes an example Github repository available for users that demonstates the recommended file structure for A360 AI projects:
`https://github.com/andromeda360/a360-example-repo`

The suggested file structure inside your repository should start at the model level with models separated into folders. Within each model folder, there should be a `model.py` and `requirements.txt` file. The `model.py` file contains the prediction script to run inference (make predictions) with an associated deployed model.



Note: *Users may complete Github integration at any time before packaging and publishing model stages. Users do not have to utilize A360 AI's preferred file structure within their Github repository. For users who elected to have A360 AI host their Github repo for model serving, no action is required.*

DATA SCIENCE CONSOLE

What is the Data Science Console?

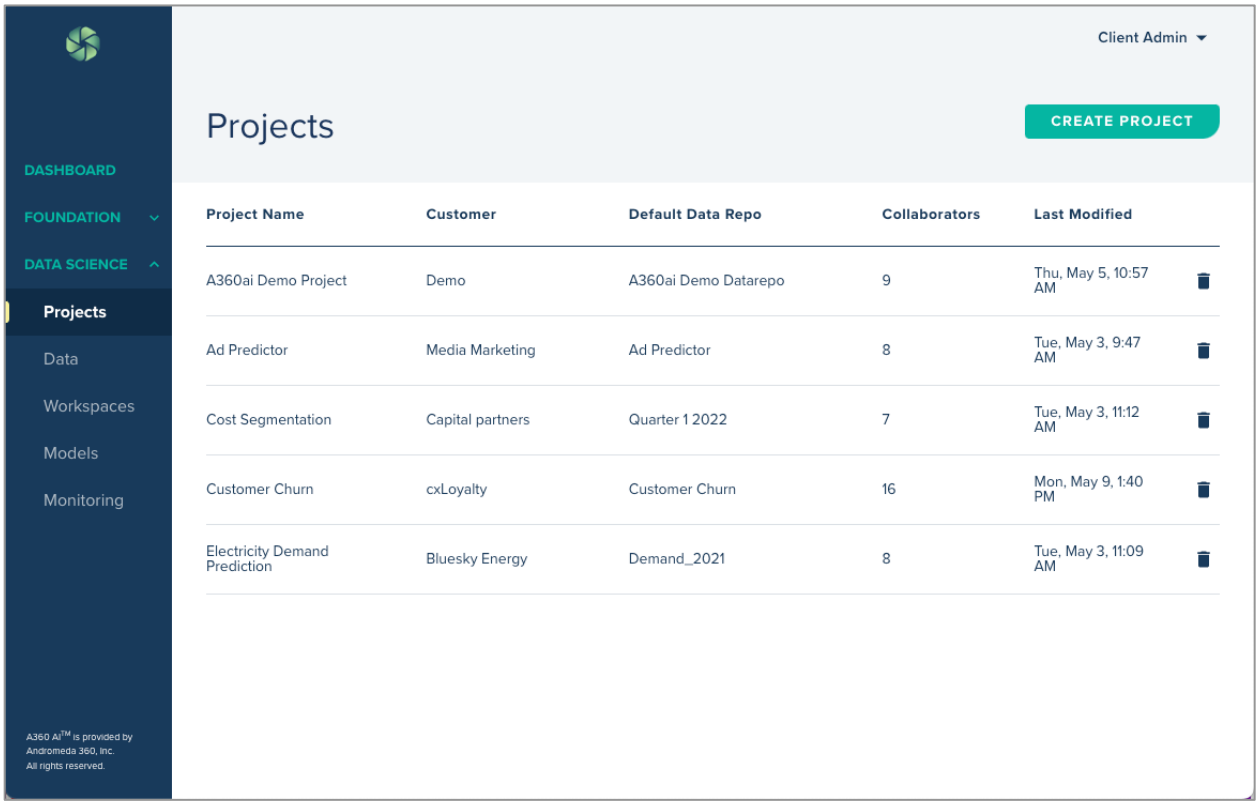
A360 AI empowers data scientists them to quickly spin-up workspaces, provision data repositories, and build models efficiently, by removing daily pain points.

How A360 AI solves these problems:

- **ACCELERATE MODEL DEVELOPMENT**
  - Easy and seamless access to model development environment and data, significantly reducing model development cycle time from months to days.
- **DRIVES AGILE MODEL DEVELOPMENT**
  - Package and deploy ML Models across Cloud and Edge platforms leveraging A360 AI's Starpack AI/ML application, Proprietary and declarative specification.
- **AUTOMATES MANUAL STEPS**
  - Execution scripts and state of data are automatically stored as snapshots. MDK functionality automates manual steps in building, experimenting and testing models.
- **SUPPORTS COLLABORATION**
  - Intuitive workflow for reviewing experiments / models similar to code review for GitHub, reducing dependency on 3rd party providers.
- **TRACKS EXPERIMENTS AND RUNS**
  - Jupyter Notebooks are available for review at runtime for reproducibility and organization.
- **ORGANIZES DATA**
  - Organize and track the training / test data sets in one platform.

Projects

Projects are the highest organization level in delivering an AI application. Projects contain collaborators, data connectors, models and their artifacts, workspaces, and monitoring dashboards. Access to models, data, and deployed applications is controlled through the Project structure. You can view the Project pane by first navigating to the Data Science console, then selecting Projects. This is where users can see projects they have created and/or are collaborators on. If no projects have been created, the page will display an empty state.



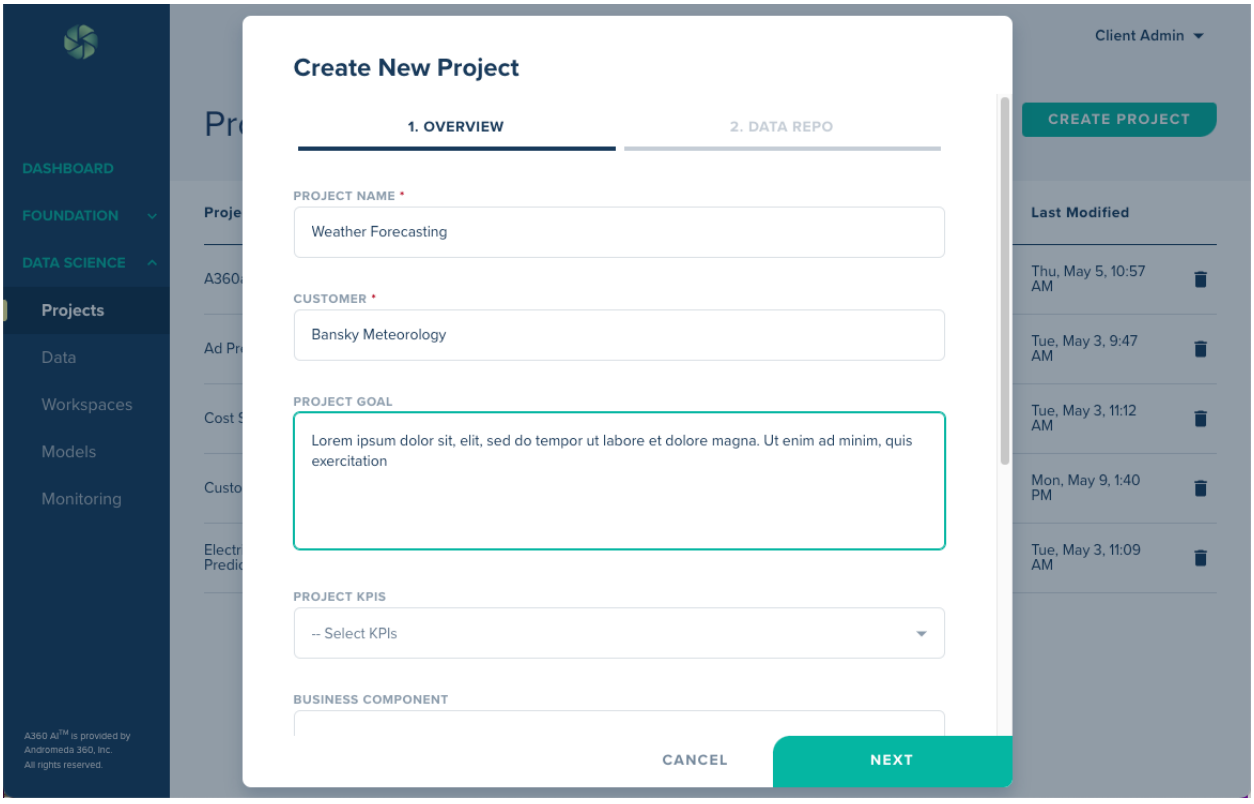
Creating a Project

From the Project pane, click on Create Project which to open a modal for you to create the project and its default data repository.

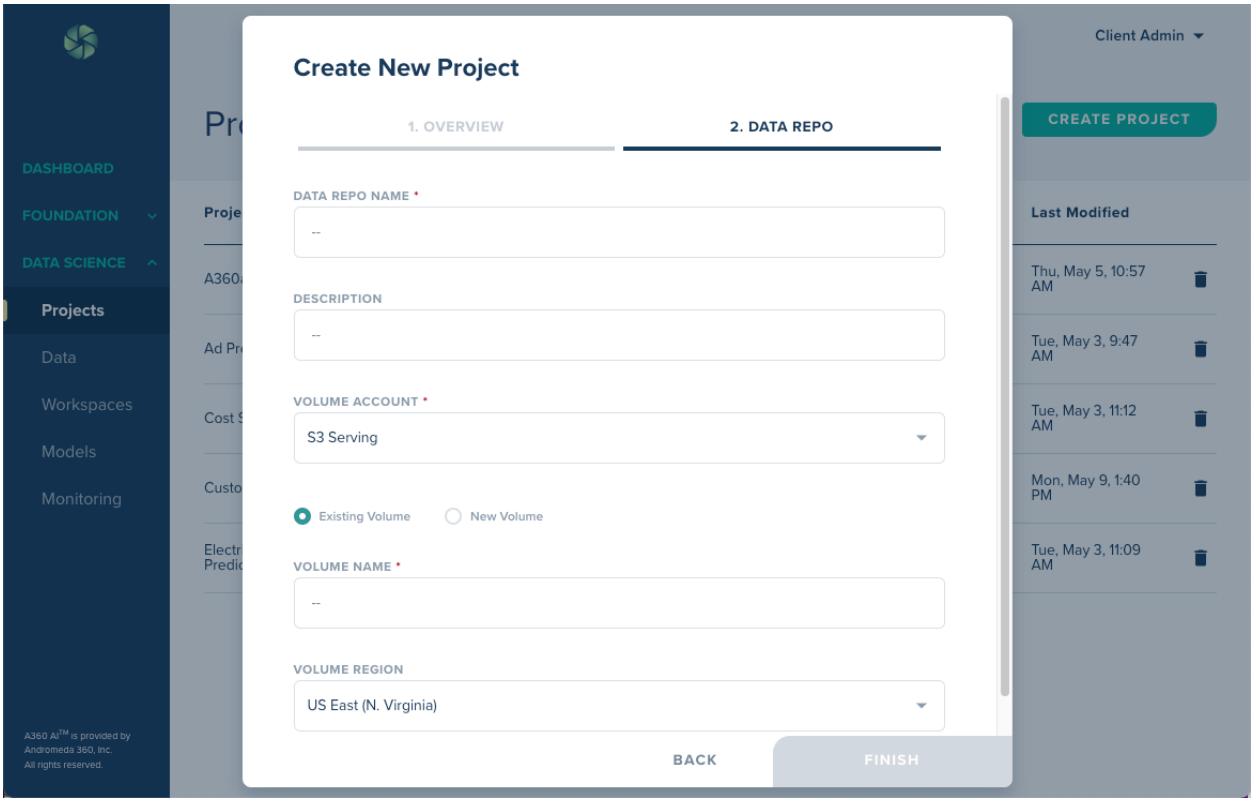
The information at the project level assures that the intent of the project is clearly defined and captured from the business stakeholder's perspective.

To create your project:

1. Enter project name (required)
2. Enter customer name (required)
3. Project Goal - What is the desired outcome, in what timeline?
4. Business Process - Is this tied to a specific business service or process?
5. Project KPIs - Multi-select drop down option
    a. Time to deployment
    b. Models in Production
    c. Models deployed per month
    d. Cost savings
    e. API calls
6. Business Component - Is this tied/affect a specific business, function or product?
7. Business Function - Is this tied to a specific activity?



Each Project requires a Data Repo - a centralized repository in an Amazon S3 bucket hosted by A360 AI – that will contain all of the model artifacts associated with that project. Data Repos are also where you'll store local data used during the model training process.



You will be required to enter or select the following information:

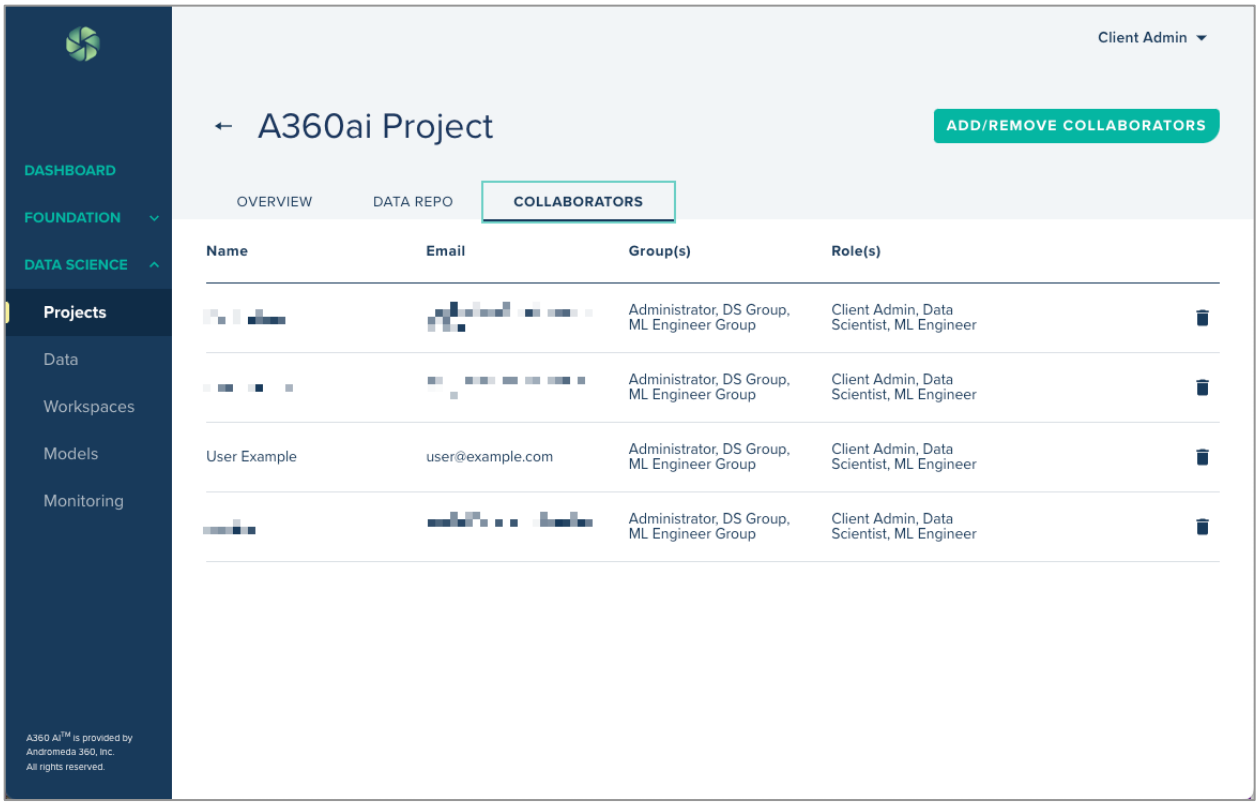
1. Data Repo Name (required) - Cannot be more than 64 characters.
2. Description
3. Volume Account (required) - This drop-down option will pull any data source accounts in which you have permissions. If the drop down is empty, you will need to contact your administrator.
4. Volume Option - Volume represents a destination that is available on the network to store your data. Select from either an Existing Volume or New Volume.
5. Volume Name (required) - Must be lowercase and numeric with dashes or spaces.
6. Volume Region (required) - Select the proper region tied to your account. If the incorrect volume name is selected, creating the repo will fail.

When you create a data repo during project creation, the data repo you've created is automatically set as the default data repo. If you create additional data repos for your project, you can change which is set to default.

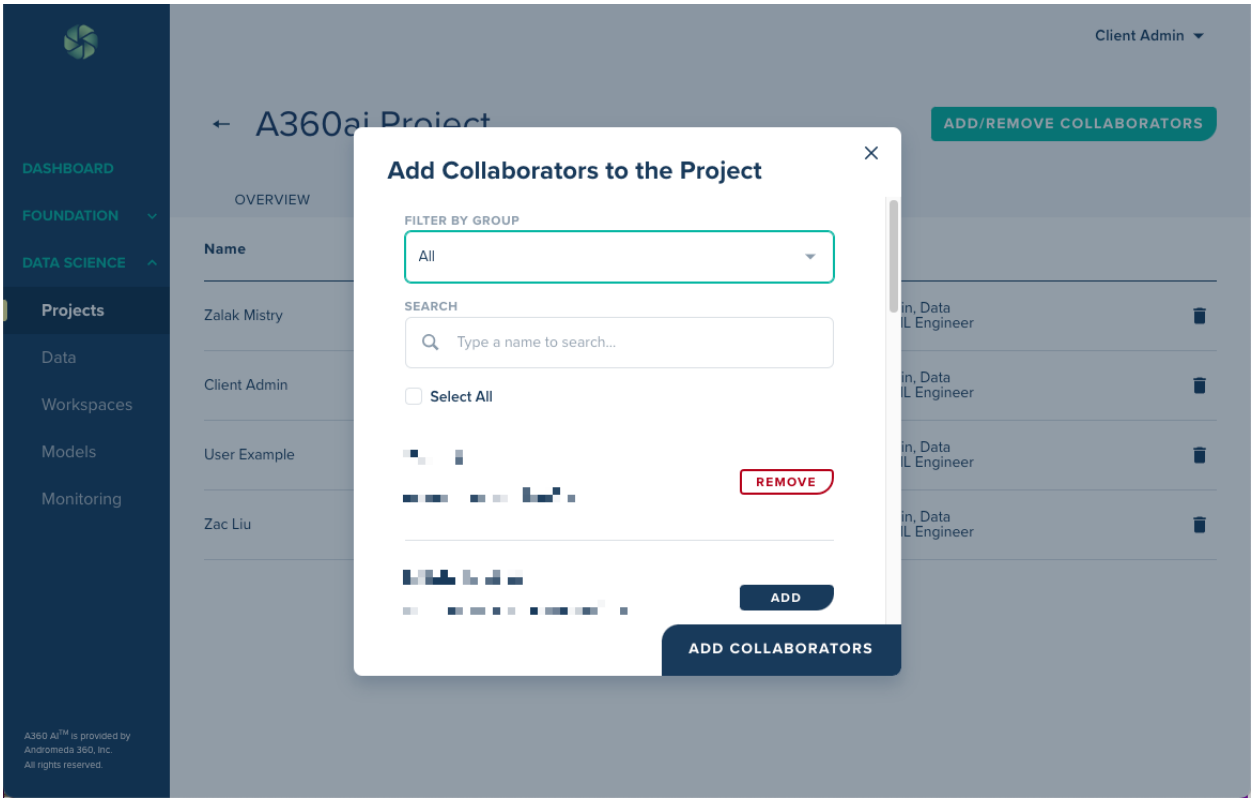
Collaborators

Users with the Data Scientist role can be added to projects as collaborators. Becoming a collaborator on a project allows access to the project and all its data repos, workspaces, Jupyter notebooks, and models.

To edit project collaborators, navigate to the Projects pane in the Data Science console. Click on the Project row you wish to edit collaborators for. You'll be taken to Overview page for that Project. Navigate to the Collaborators tab to edit project collaborators.



1. Click on Add/Remove Collaborators



2. Use the drop-down menu to Filter by Group or you can Search by name:
    a. Once a group is selected all the users within the group will be displayed in a list.
    b. Add one or multiple users to be collaborators for this project.
    c. Click Add Collaborators to finalize.
3. The users will then display in a list on the collaborators tab.

NOTE - *Users that become inactive will automatically be removed from the collaborators list, unless they are the only user tied to the project. In this situation, at least one new collaborator will need to be added before the inactive collaborator can be removed.*

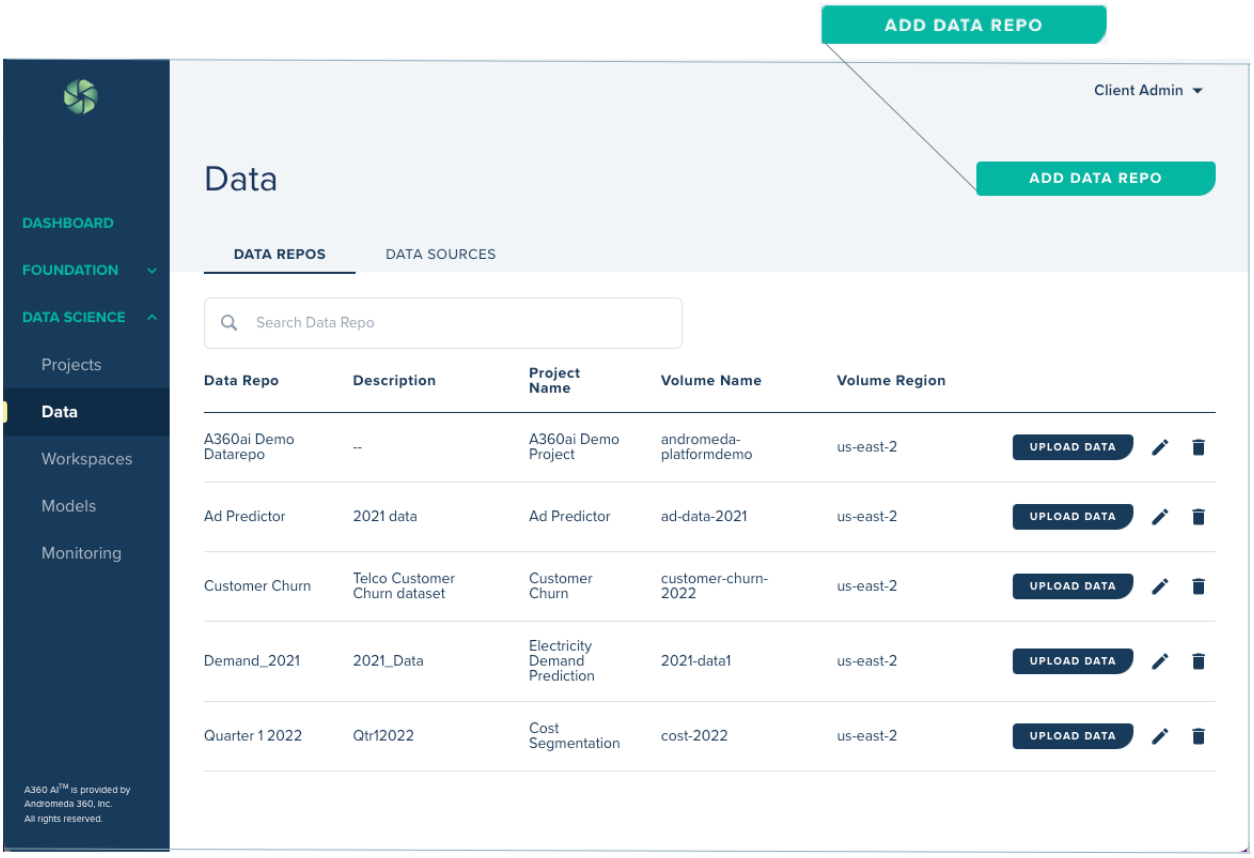
Data Repos and Data Sources

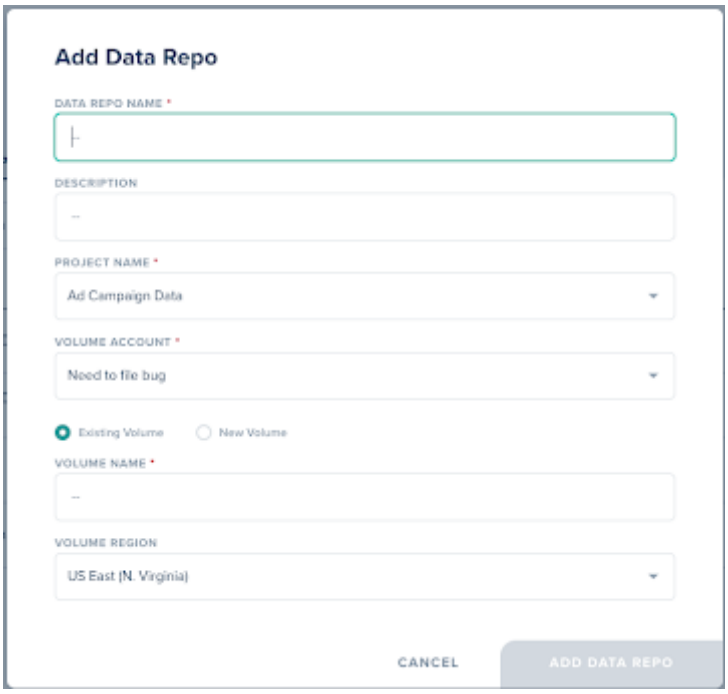
On the Data pane, users can create additional repositories for data that can be queried via the A360 AI Model Development Kit (MDK) from within a Jupyter notebook. Select Add Data Repo to connect to an existing or create a new data repo as an AWS S3 bucket.

NOTE - *Default data repos are created with each project. Any new repo created, will have to be set as default at the project level, if you want your project artifacts to automatically be saved to the new repo.*

Adding a Data Repo

To add a data repo, navigate to the Data pane under the Data Science console and click Add Data Repo.

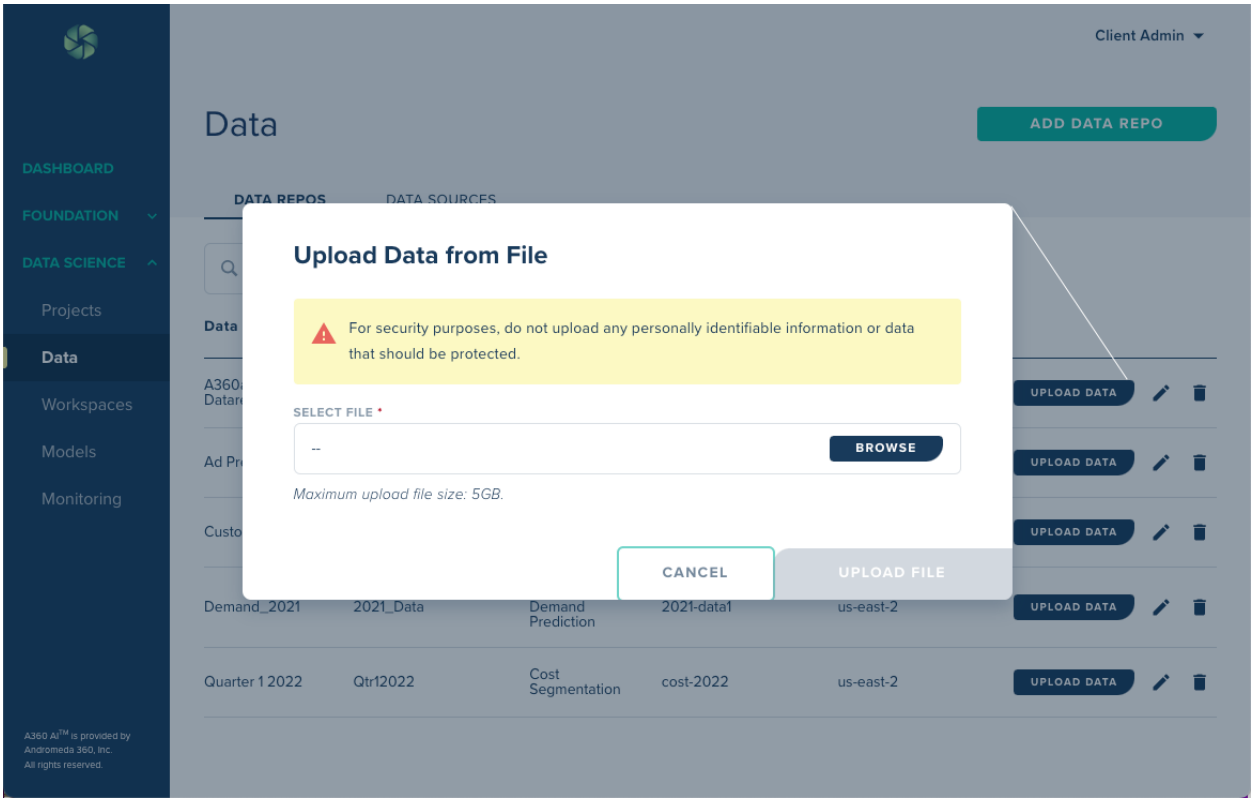




1. A popup modal will require you to enter or select the following information:
    a. Data Repo Name (required) - Cannot be more than 64 characters.
    b. Description
    c. Project Name (required) - Drop-down selection.
        i. If you have not created a project or are not a collaborator on any projects, the drop down will be empty. You must create a new project in order to add a data repo.
    d. Volume Account (required) - This drop-down option will pull any data source accounts in which you have permissions. If the drop down is empty, you will need to contact your administrator.
    e. Volume Option - Volume represents a destination that is available on the network to store your data. Select either Existing Volume or New Volume.
    f. Volume Name (required) - Must be lowercase and numeric with dashes and no spaces.
    g. Volume Region (required) – Select the proper region tied to your account. If the incorrect volume name is selected, creating the repo will fail.
2. Click Add Data Repo to close the modal and complete the process.

Upload Data From a File

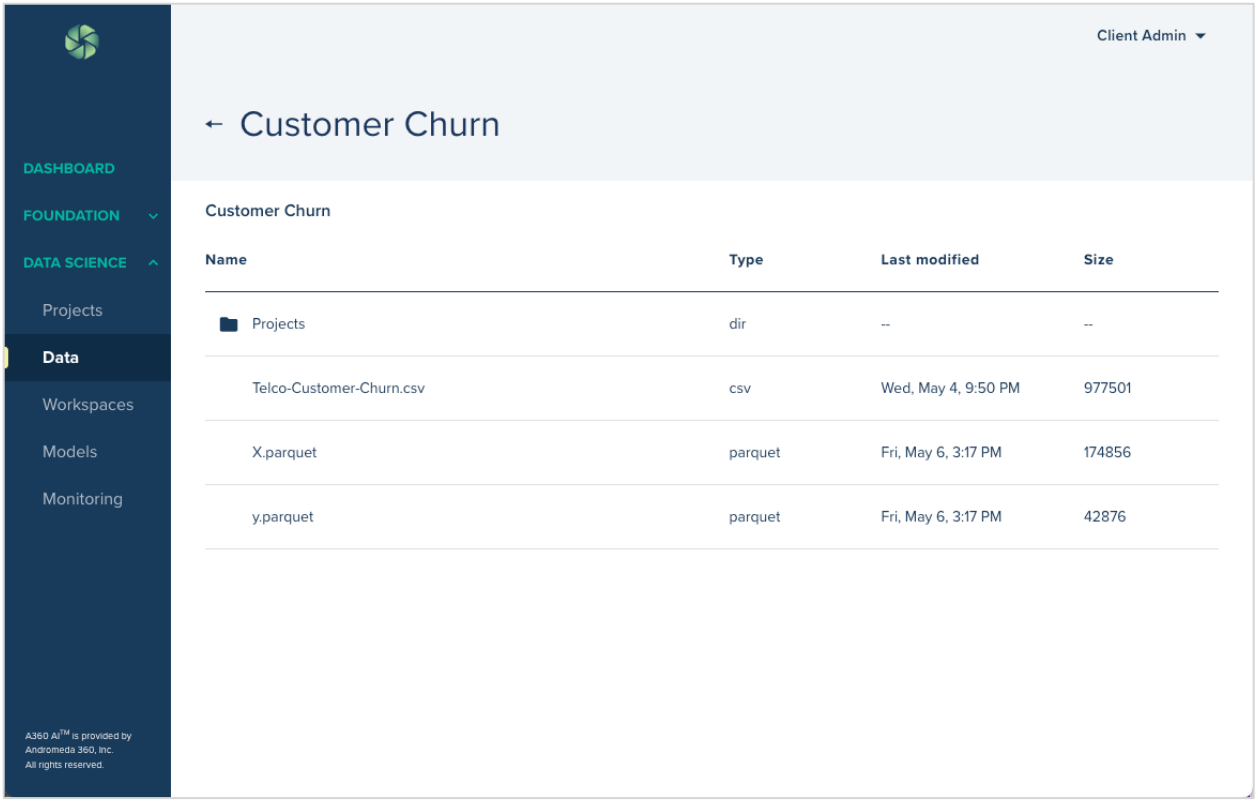
Files can be uploaded to a data repo for access with the MDK and Jupyter Notebooks.



Adding data from file:

1. Click on the Upload Data button.
2. A popup modal will require you to enter or select the following information:
    a. File(s) (required) - Browse and select one or many files to attach for the upload.
        - For security reasons you should not upload files with Personally Identifying Information. A360 recommends anonymizing PII data before uploading to a data repo.
        - File size restrictions - files size is limited to 5GB. Files larger than 5GB should be broken into smaller files.
3. Click Upload File to start the data upload:
    - Once the upload is complete, you can click on the data repo row and view the files which are loaded into the directory.
    - The data in the file can be found in the volume bucket of the selected data repo.

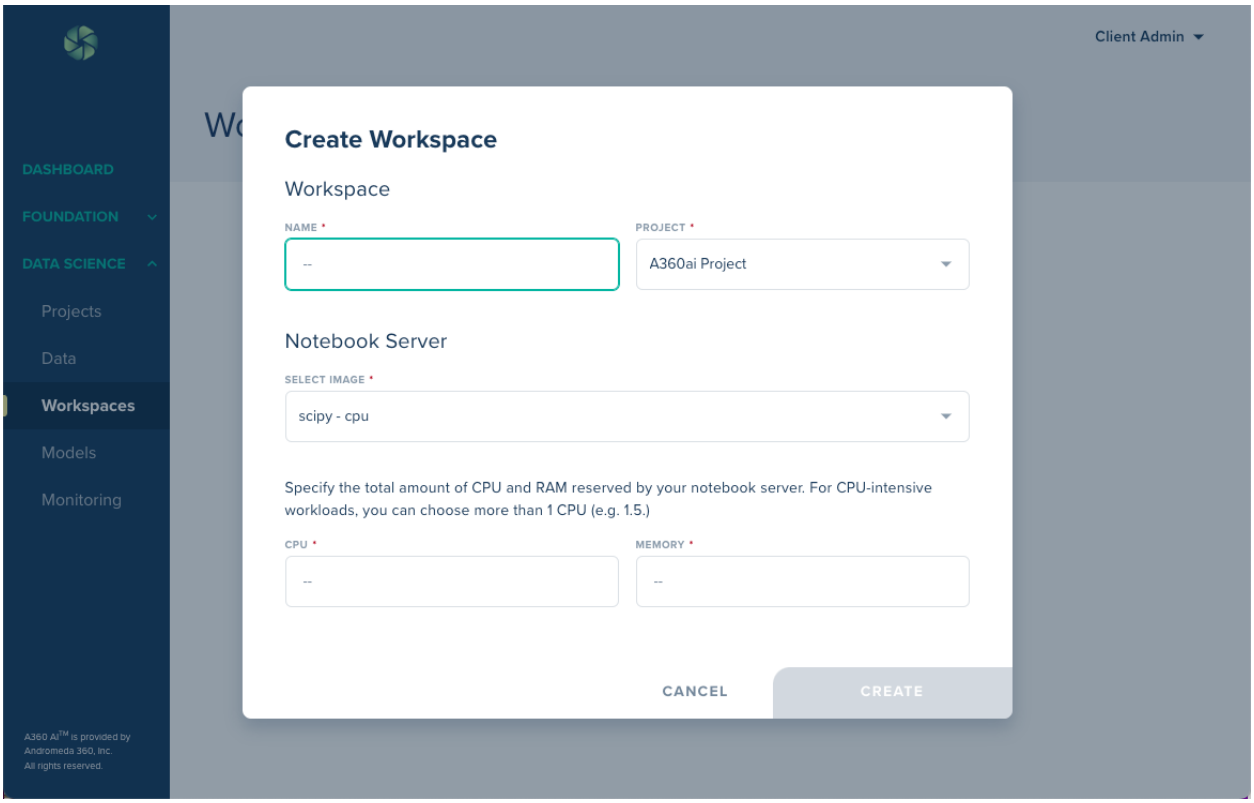
Clicking on an existing data repo will show the repo contents in tabular format, where you can click and sort by Name, Type, Last Modified, or Size.



Workspaces

Once you're ready to begin building models you can create your workspace and notebook server. Workspaces host the Jupyter notebook environment and may be customized with a Notebook Server image and compute resources.

Creating a Workspace

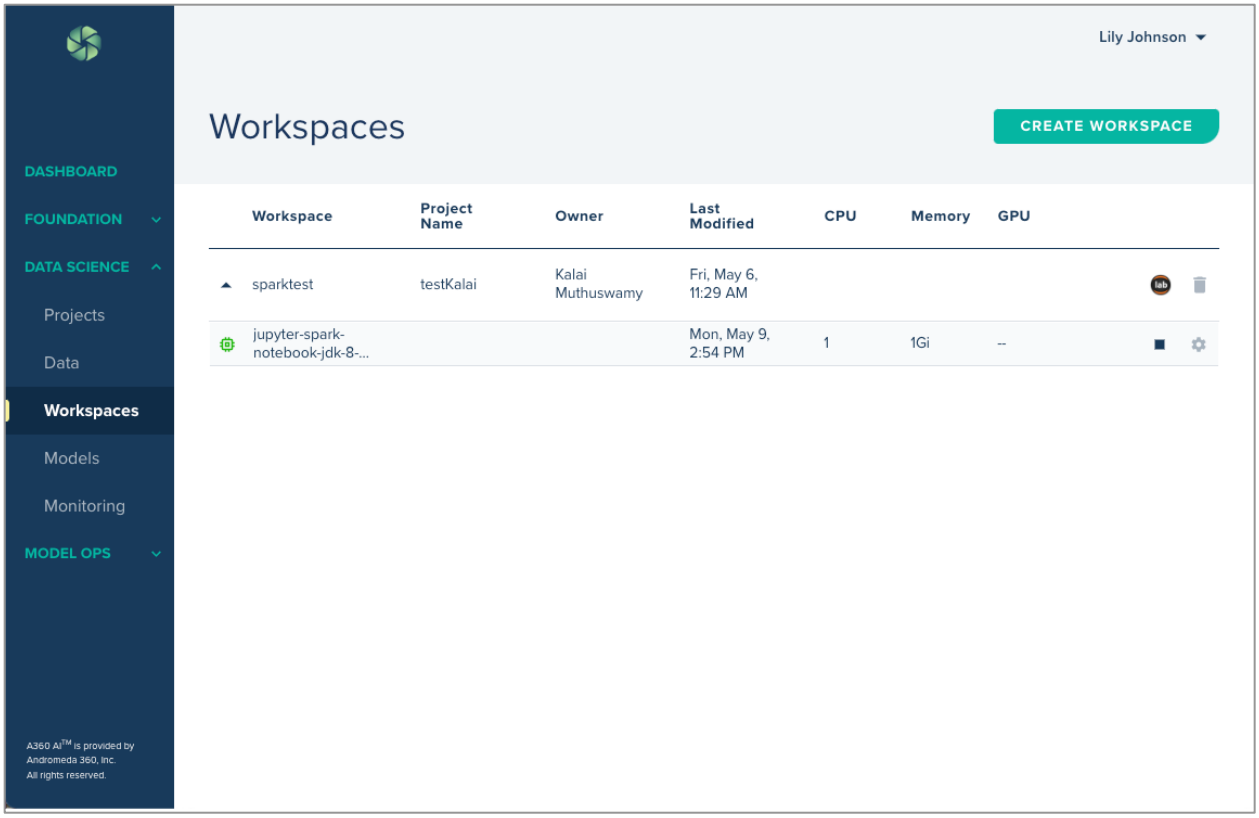


To create a workspace:

1. Click on the Create Workspace button.
2. A popup modal will require you to enter or select the following information:
    a. Name (required) - Enter a name for your workspace.
    b. Project (required) - Select which project is tied to this notebook.
    c. Image (required) - Select from predefined notebook images images.
        i. Select the image based on the type (tensorflow, pytorch, etc.) of model to be trained.
    d. CPU (required) - Typical value is 1.

e. Memory (required) – Typical value is 1Gi
    i. Memory must be typed with *memorySize*Gi
f. Number of GPUs - Defaults to None, currently limited to model training functionality.
    i. If GPUs are not available, the drop down will not display.

3. Clicking Create will generate the request. Once complete, the Jupyterlab button will be enabled. This may take a few minutes.



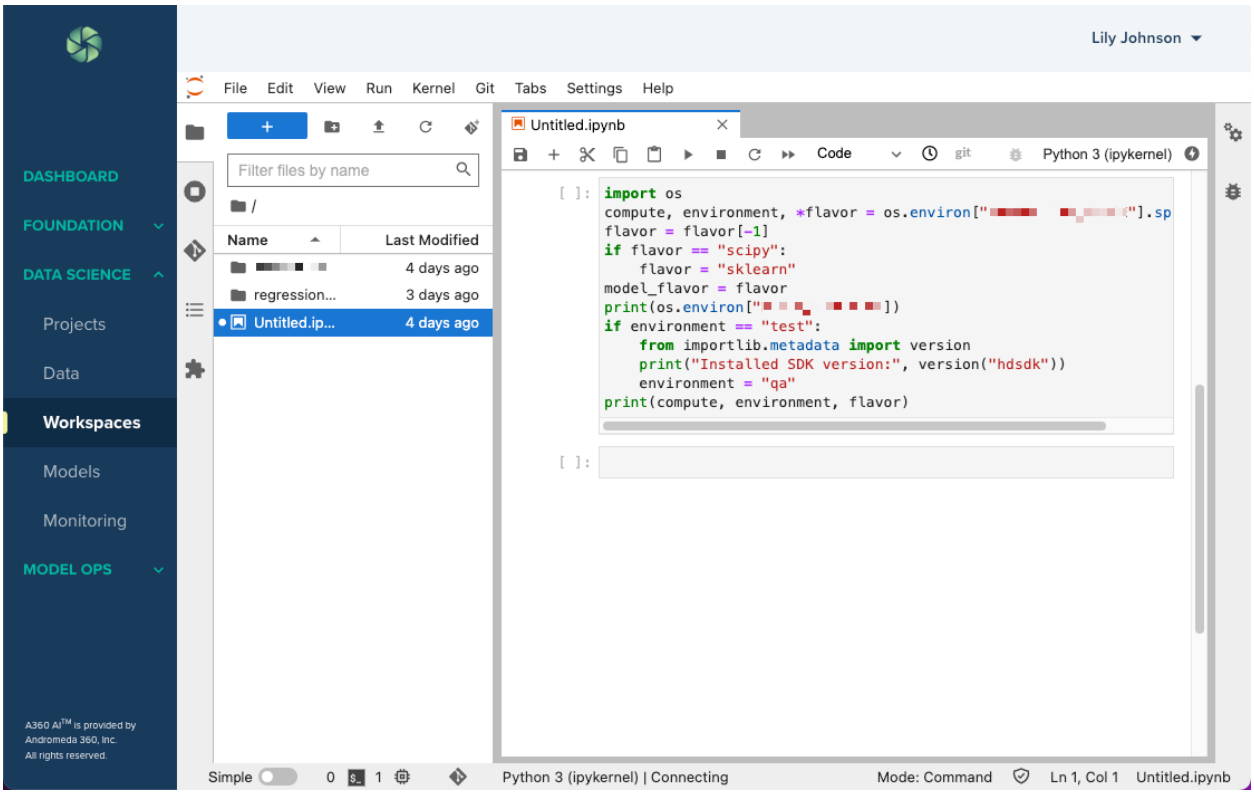
Notebook status is indicated by the icon to the left of the workspace instance name with the following icons:

- Notebook is RUNNING
- Server STOPPED
- ERROR starting Server

Workspaces are persisted on A360 AI whether they are in a running or stopped state. You can safely Stop your notebook by clicking the square Stop icon without losing your work. If your notebook is idle for an hour, it will automatically shut down.

Notebook instances can also be edited, in order to change the image or update the compute requirements, *but* you'll need to stop the notebook before the Edit button (gear icon) is enabled .

Clicking the Jupyeterlab button will take you to the workspace where you can work within an interactive Jupyter notebook environment.

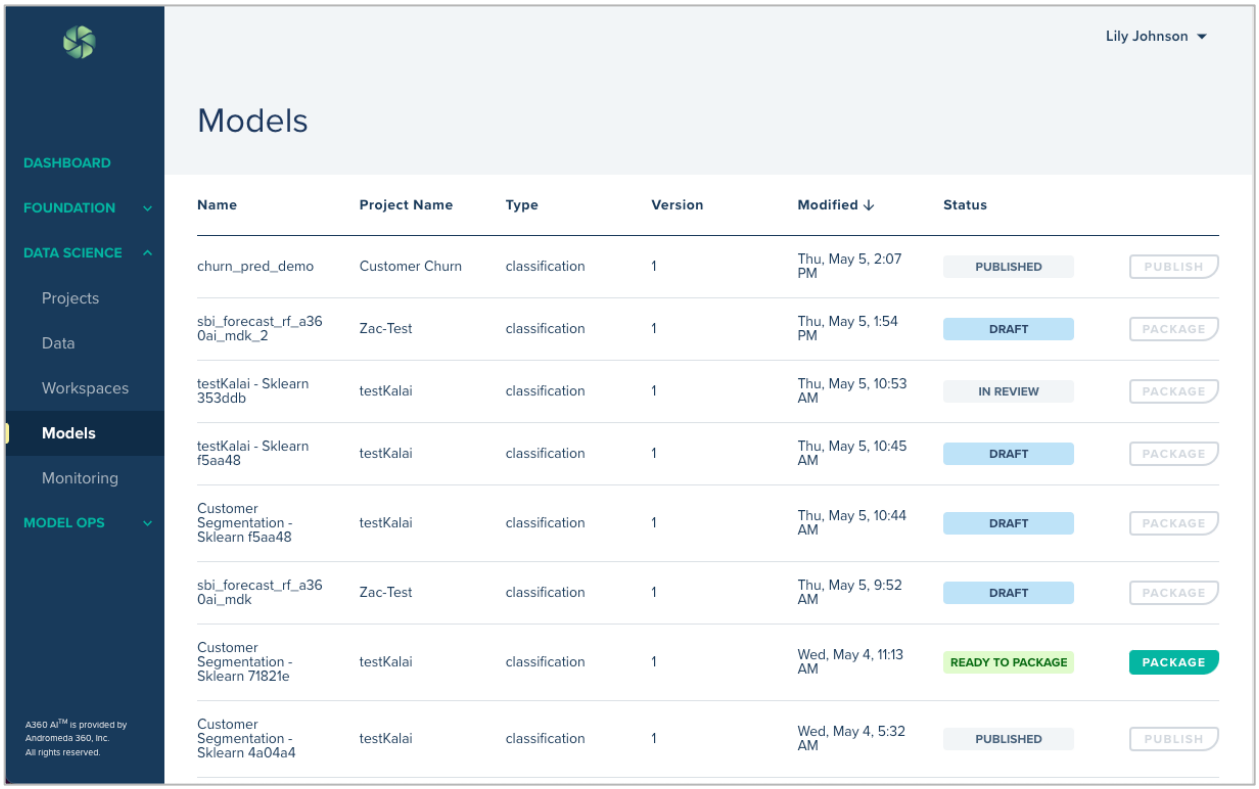


Deleting a Workspace

Deleting a workspace will *permanently* delete active notebooks and files in the workspace. Static notebooks and other model artifacts will remain in the project's default data repository after you delete a workspace..

Models

All models created by you or for projects in which you are a collaborator will be listed on the Models pane. Navigate to the Models pane under the Data Science console to see all of the models you've been added as a collaborator to.



Model Status shows which stage of the model creation process that model is in:

- Draft – Models are currently being developed. Submitting for Review will move a model from the Draft stage to the In Review stage.
- In Review - Upon completion of experimentation and testing, users can select the best model to be peer reviewed by clicking Submit for Review. The model's status is then changed to In Review. This action can be taken from the model's detail page.
- Need Revisions - If the reviewer of the model selects Deny during the review process, the model's status updates to Needs Revision and a notification will be sent to the user who submitted the model for review.
- Ready to Package - If the reviewer selects Approve the model is then ready to be packaged. The packaging process combines the model, model artifacts, and necessary Python

packages and libraries in preparation for publishing to a test or production environment into a Docker container image.

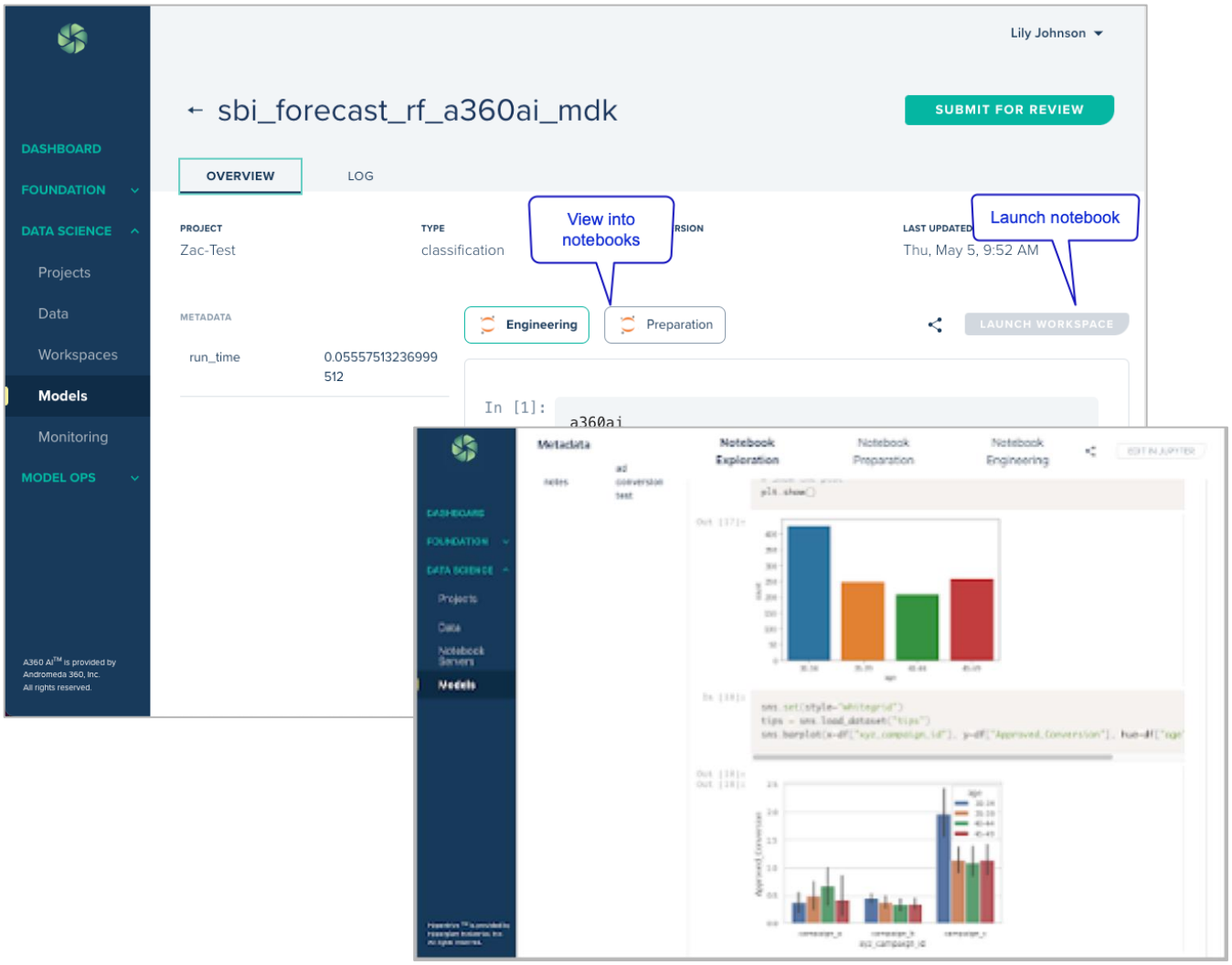
- Package Failed - Occurs when the process encounters an issue. Details can be found on the model detail logs. The option to Package will be enabled so you can run the request again.
    - Package failure can occur for a number of reasons, including:
        - The model type is not compatible with the A360 AI model serialization framework.
        - The model Docker container image has Severe or High risk security vulnerabilities.
- Ready to Publish - Your model is now ready to be deployed to a test environment.
- Publish Failed - Occurs when the process encounters an issue. Details can be found on the model detail logs. The option to Publish will be enabled so you can run the request again.
- Publish - The model has successfully been deployed in a test environment. An endpoint is available to expose the model to external applications.

Model Details

For each model, the detail pane has a view of general information, as well as a view into the notebook(s) tied to the model. If there are multiple notebooks, each can be viewed by clicking on their header.

Notebooks are available in the model detail pane after the first training experiment run. If there are multiple runs in the experiment only the best one will display here. There is also a direct link into Jupyter while the model is in Draft and In Review statuses.

All collaborators within each project have access to these details.

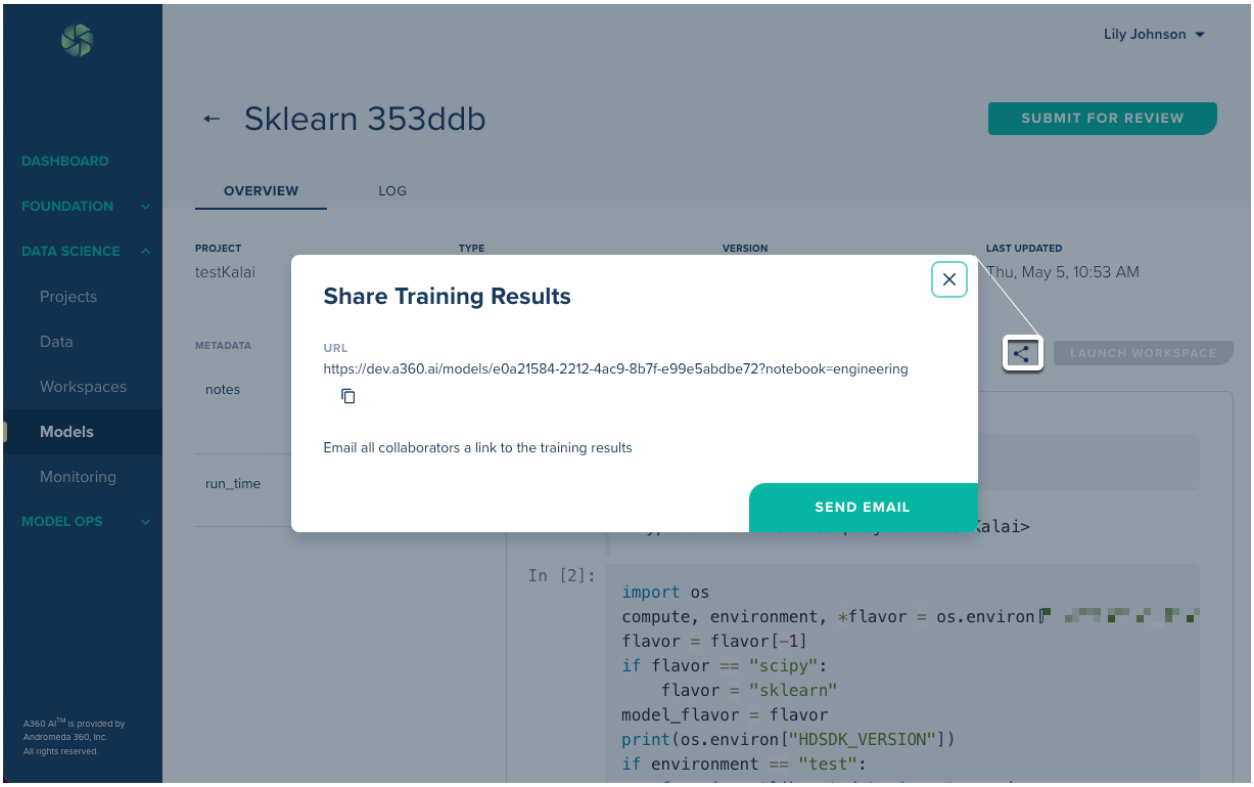


Sharing Training Results

Easily share your notebook training results with collaborators.

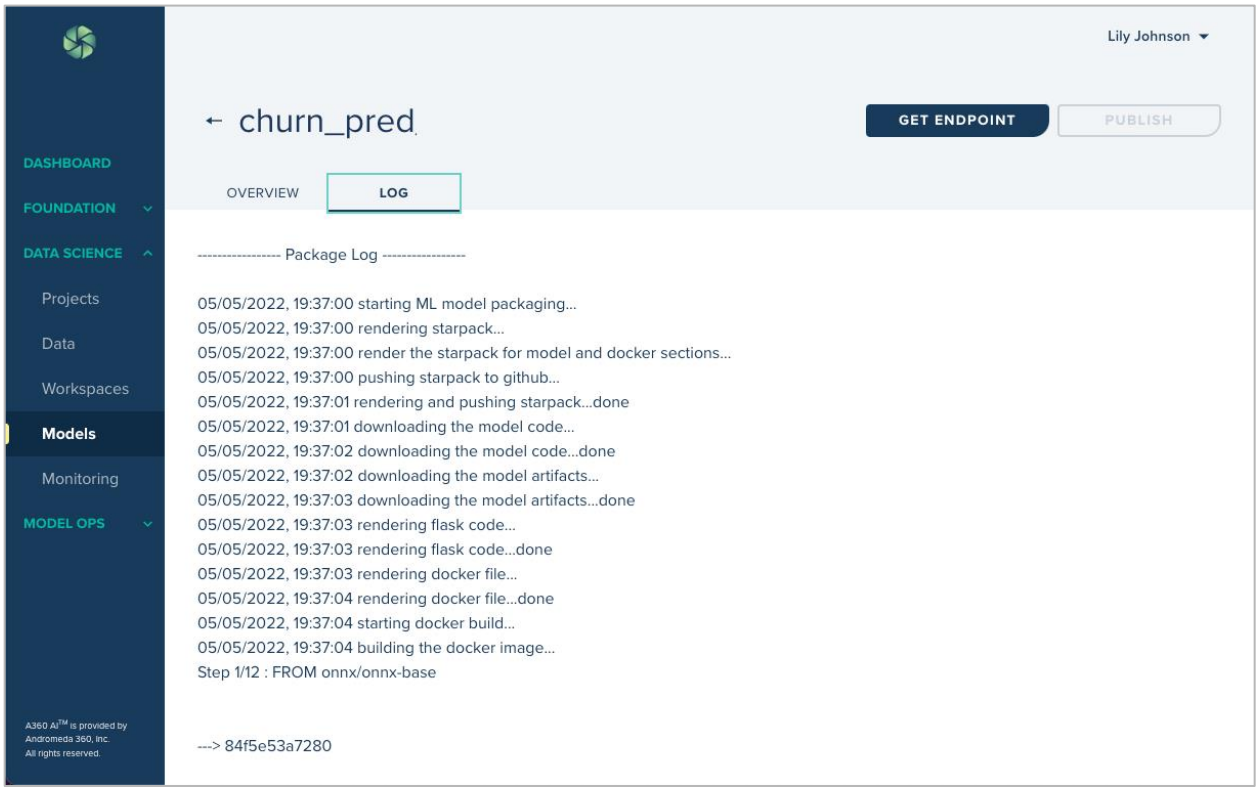
Share results by:

1. Clicking on the share icon
    a. You can share each notebook individually, so if you have multiple notebooks associated with a model, first click on the notebook in which you want to share the results.
2. From the popup you can either:
    a. Copy the URL to share manually.
    b. Click Send Email which will send a system email to all collaborators on the project.



Model Logs

Logs currently capture the details tied to packaging and publishing models.

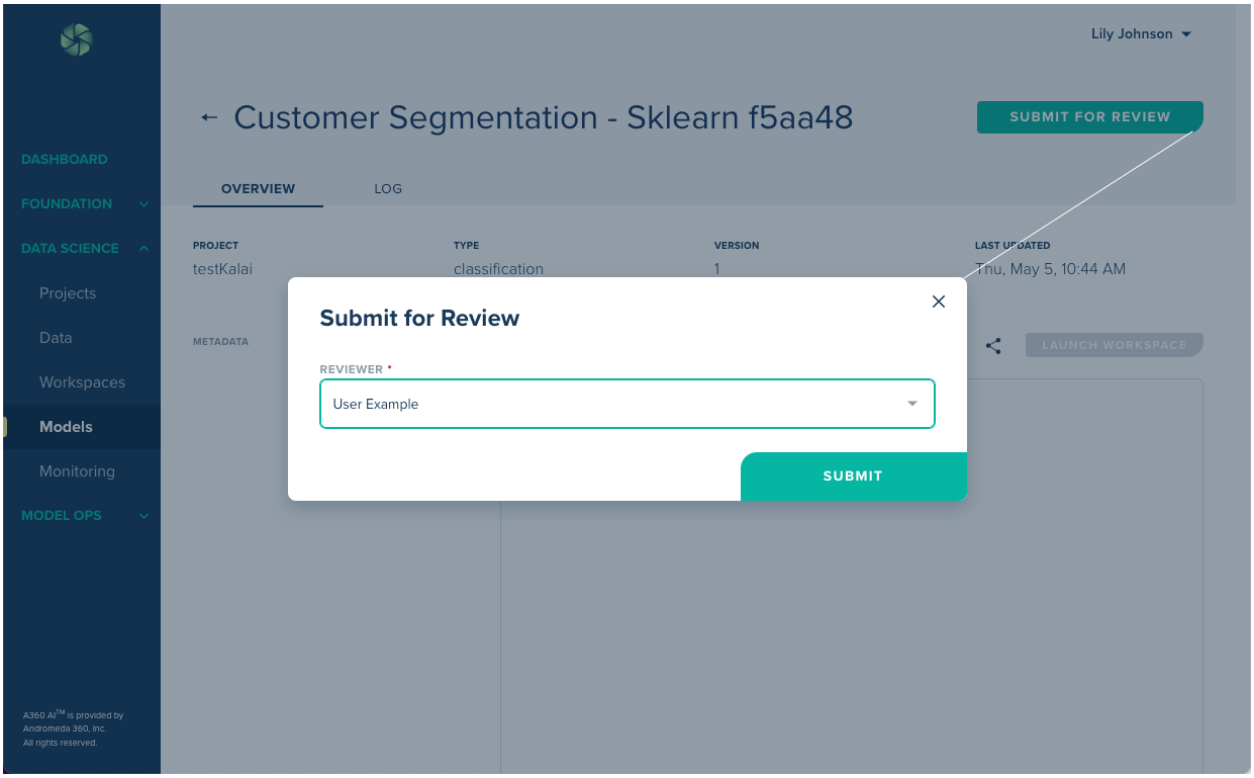


Model Review

Once the selection of the best model has been made, it can be sent for peer review by clicking on Submit for Review.

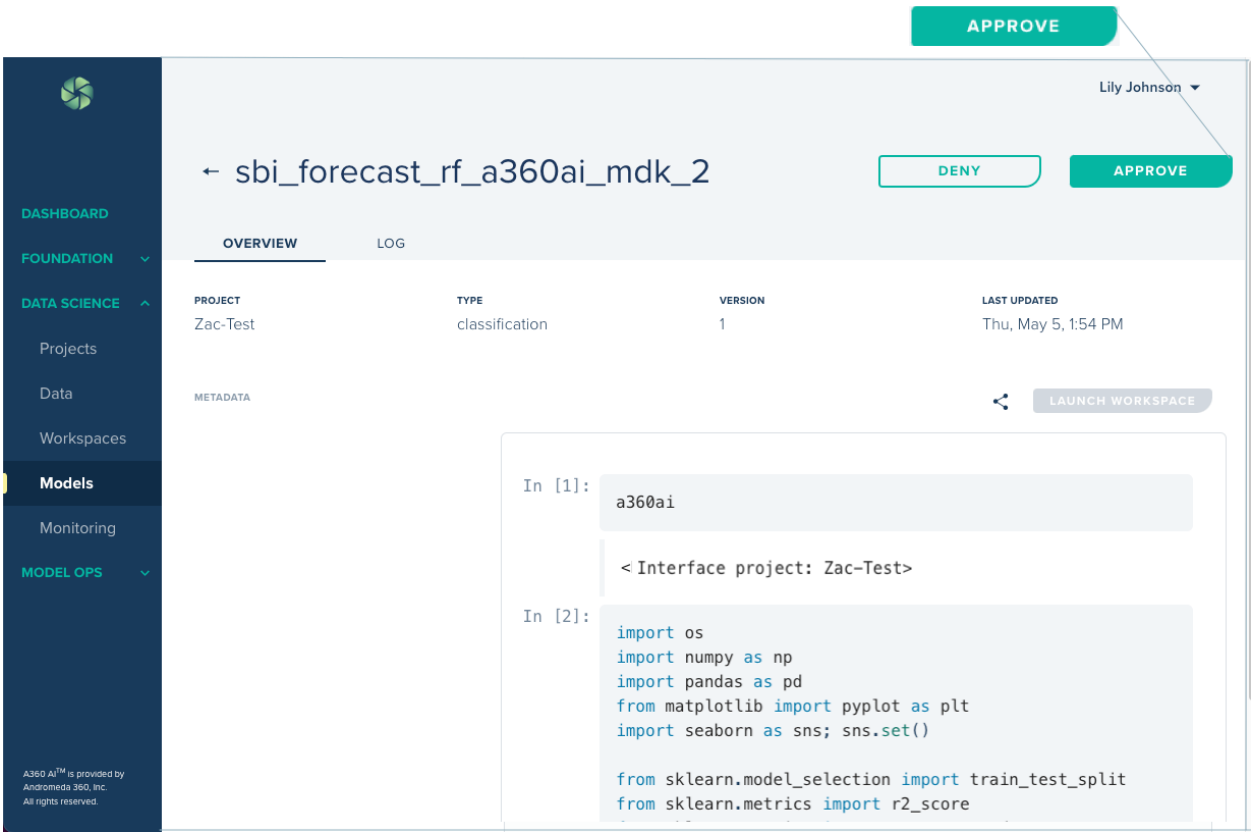
To submit a model for review:

1. Click on the Submit for Review button.
2. Select a reviewer from the drop-down menu, which displays a list of the collaborators on the project.
3. Slick Submit to submit the model for review.
    a. A system-generated email will notify the reviewer and will provide a link to the model's detail page.
4. The status of the model is updated to In Review.



Review Model

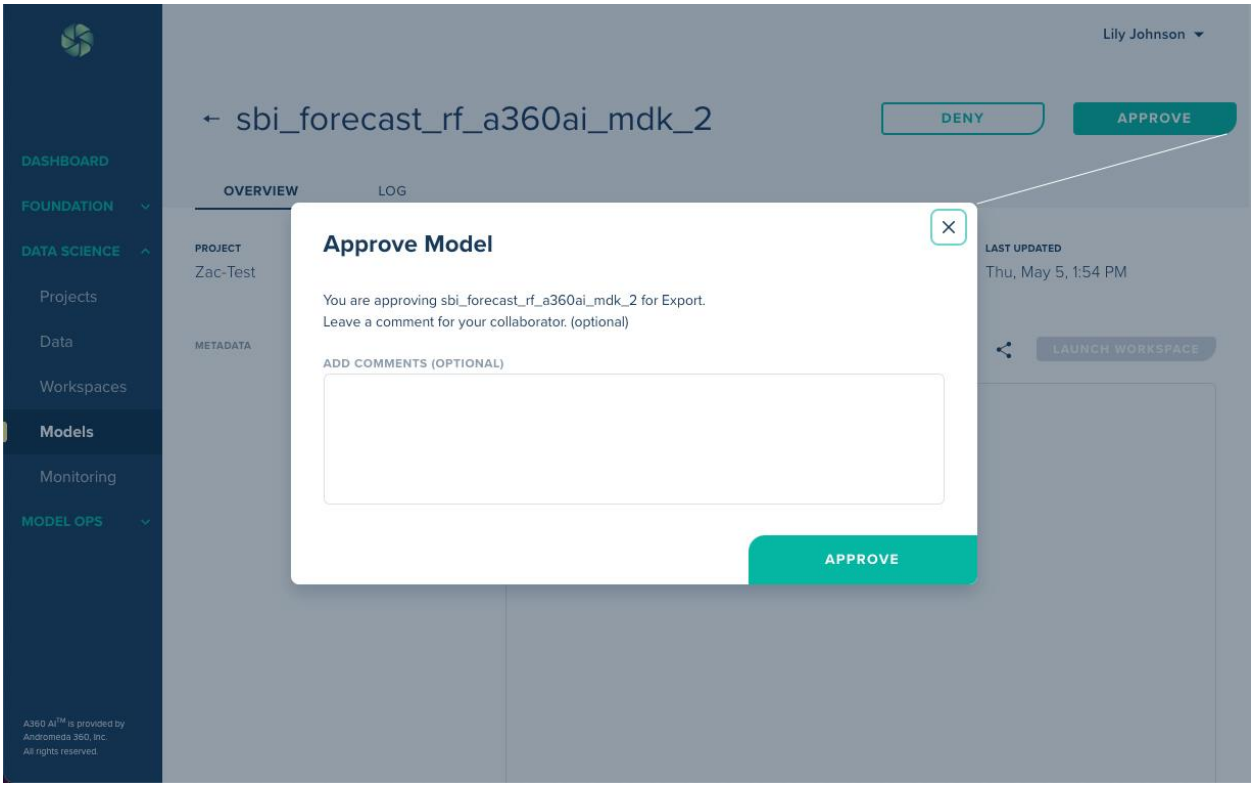
The reviewer assigned in the previous step will receive an email notification that contains a link that directs them into the model detail page. Here the user can review the information in the notebooks and either approve or deny the request.



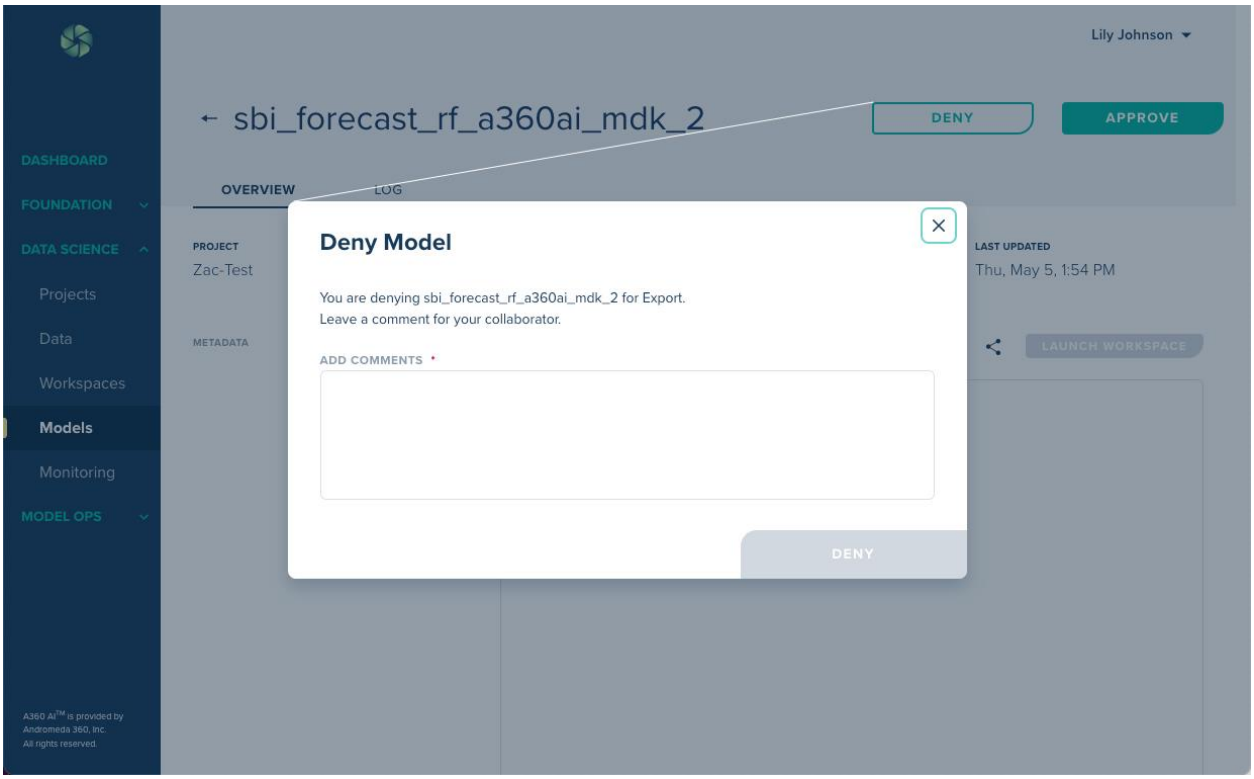
To Approve a model that has been submitted for review:

1. Click on the Approve button in the top-right corner of the model detail pane.
2. Enter comment (optional).
3. Click Submit to approve the model.

This completes the approval process and notifies the user through a system-generated email that the request has been approved. Once approved, the model is ready to be packaged in preparation for publishing.



Denying a Model Submission



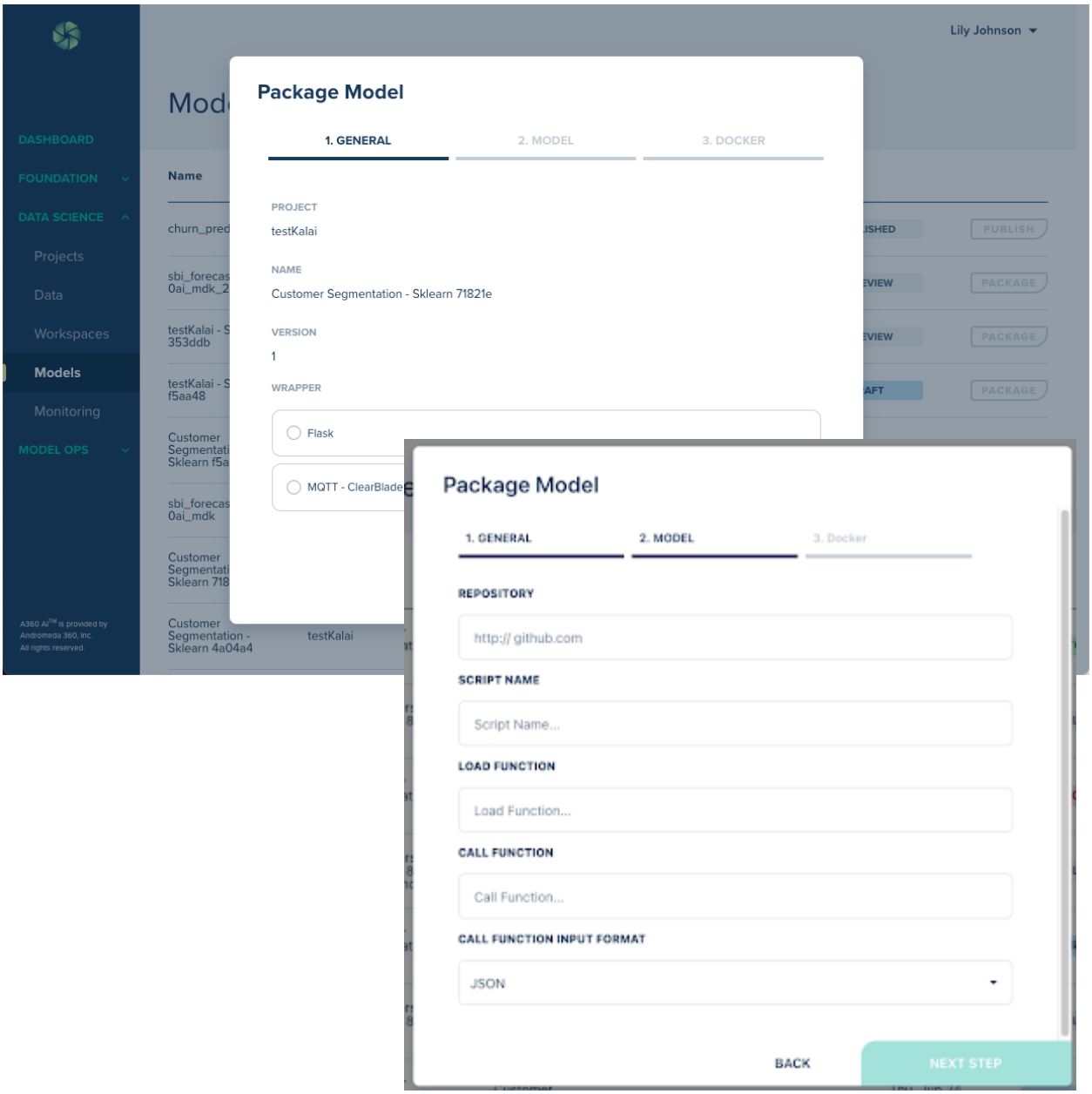
Steps to a model that has been submitted for review:

1. Click on the Deny button at the top-right corner of the model detail pane.
2. Enter comment (required).
3. Click

This completes the review process and notifies the user through a system generated email that the request has not been approved and includes the reviewer's comments. After denying a model the status of the model is updated to Needs Revisions and the user can resubmit for review after making any suggested updates.

Packaging a Model

After a model has been approved through the review process it may be packaged. Packaging a model builds a secure Docker container image containing the model, model artifacts, and all necessary libraries and dependencies for running the model as an AI application in a RESTful endpoint.



Packaging

Steps to Package a model:

1. Click on the Package button.
2. A popup modal will require you to enter or select the following information:
    a. General information tab:
        i. Project - The project associated with the model will auto-populate.
        ii. Name - The model's name will auto-populate.
        iii. Wrapper (required) – Model artifacts will be wrapped into a web application, which is built into secure Docker container image for deployment. The typical wrapper will be Flask unless you know the model will be published to Clearblade platform.
            - Starpack supports the following:
                a. Flask
                b. MQTT - Clearblade
    b. Model information tab:
        i. Repository (required) - Should be a Github repository link which ends in `.git`
        ii. Script Name (required) - Link to your python file ending in `.py`
        iii. Call Function (required) – The call function is the URL of the prediction script used for model inference. The predictions script must be located in the same Github repository that was entered under Repository.
        iv. Call Function Input Format - Defaults to JSON.
        v. Call Function Output Format - Defaults to JSON.
        vi. Call Function Validation Input
        vii. Call Function Validation Output
        viii. *Listen Topic (required) - Only applies to MQTT Clearblade wrapper.
        ix. *Respond Topic (required) - Only applies to MQTT Clearblade wrapper.
    c. Docker information tab:
        i. Base Image (required) - Defaults to `onnx/onnx-base`.
            - Customers can use their own base image, but their the image must contain the ONNX Python 3.x library.

ii. Dependencies (required)

iii. Registry Type - Defaults to AWS ECR.

iv. Registry URL (required)

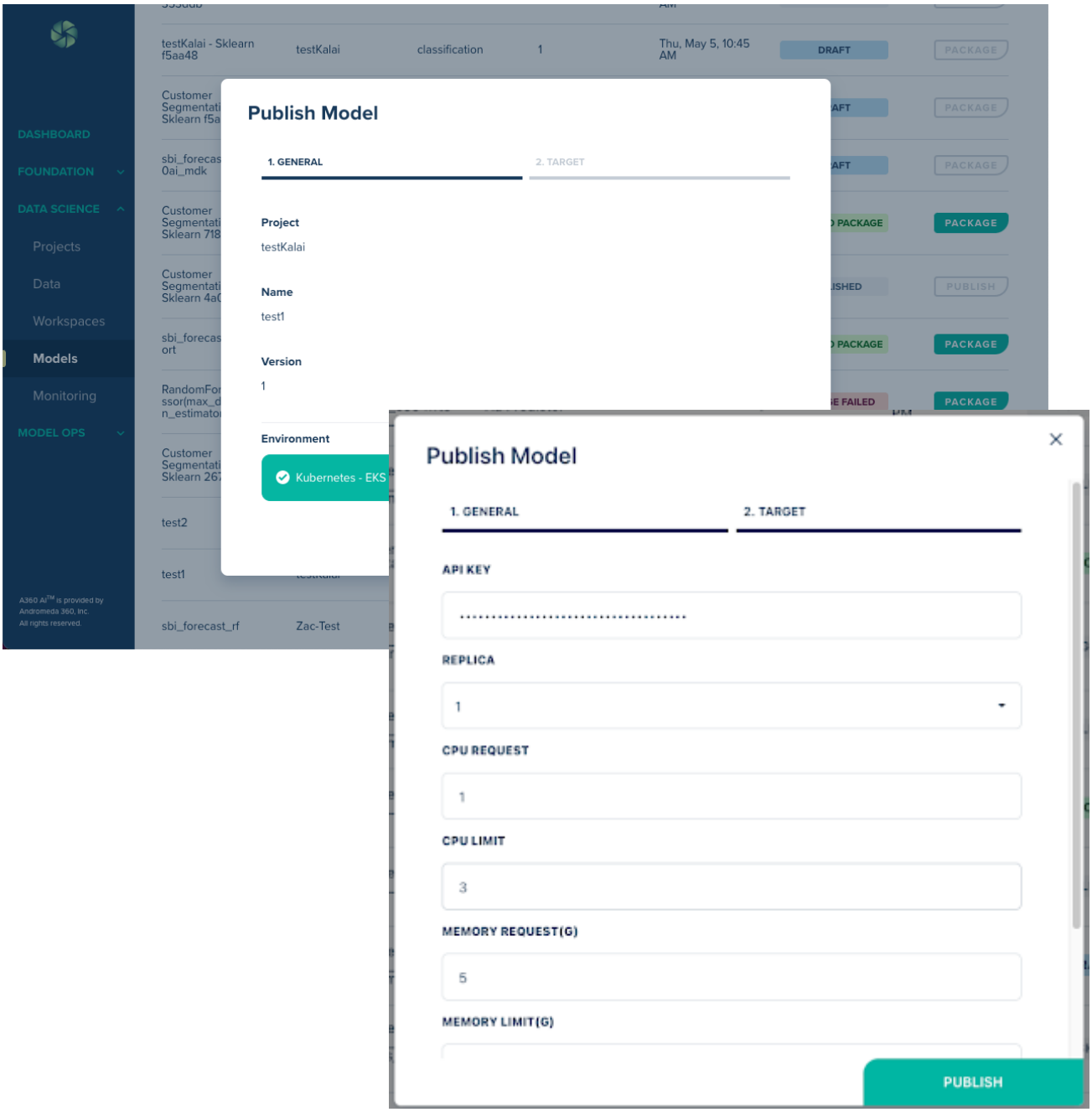
d. Package - Submits the request and processes the packaging operation.

When the operation has successfully been completed, the status of the model will be updated to Ready to Publish and the option to Publish the model will be available.

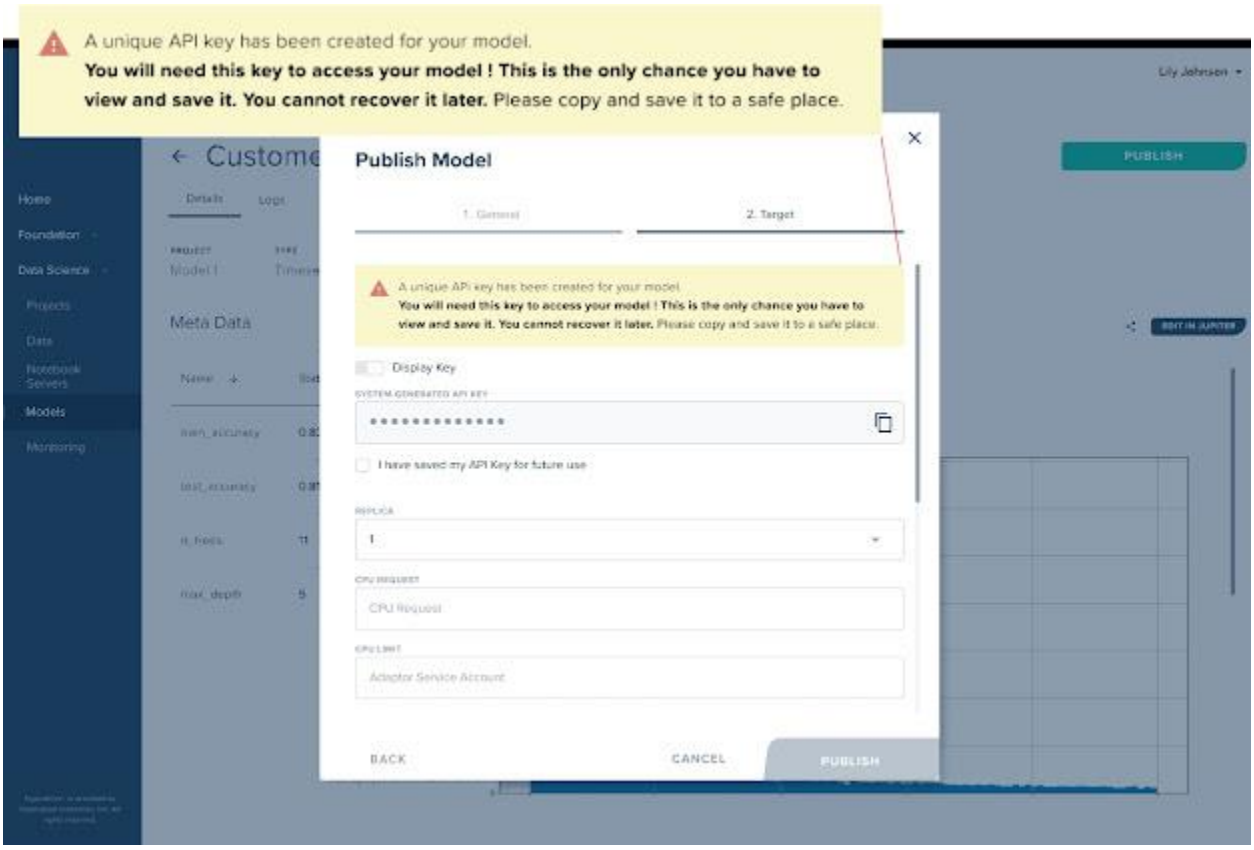
If the operation encounters an issue and fails, the status displayed will be Package Failed and the option to Package will be enabled so you can try again. Logs are viewable from the model detail page for troubleshooting purposes.

Publishing a Model

Once Packaging is complete, the a model can be deployed to a test or production environment for inference.



Publishing a Model



To Publish a model:

3. Click on the Publish button.
4. A popup modal will require you to enter or select the following information:
    a. General information tab:
        i. Project - The project associated with the model will auto-populate.
        ii. Name - The model's name will auto-populate.
        iii. Environment (required) – The target deployment environment. Publishing to an environment other than Clearblade requires that the target environment is a Kubernetes cluster. Options for deployment are:
            1. Kubernetes - EKS
            2. Clearblade
    b. Target information tab:

i. System-Generated API Key - Users can view the key by toggling the Display Key slider.
   - Users *must* copy this key in order to access the model.
   - **NOTE** - *Paste the key to a word or .txt document during this step. In following steps, you will copy the endpoint information and if you haven't saved the API key, it will be overwritten.*

ii. Replica - A reference to how many pods of the model to be published will be running on the model serving cluster.

iii. CPU Request (required) - Minimum number of CPU cores Kubernetes will allocate it to the model pods.

iv. CPU Limit (required) - Maximum number of CPU cores kubernetes can allocate it to the model pods.

v. Memory Request (G) (required) - Minimum number of Gigabytes of memory Kubernetes will allocate it to the model pods.

vi. Memory Limit (G) (required) - Maximum number of Gigabytes of memory Kubernetes can allocate it to the model pods.

vii. Environment

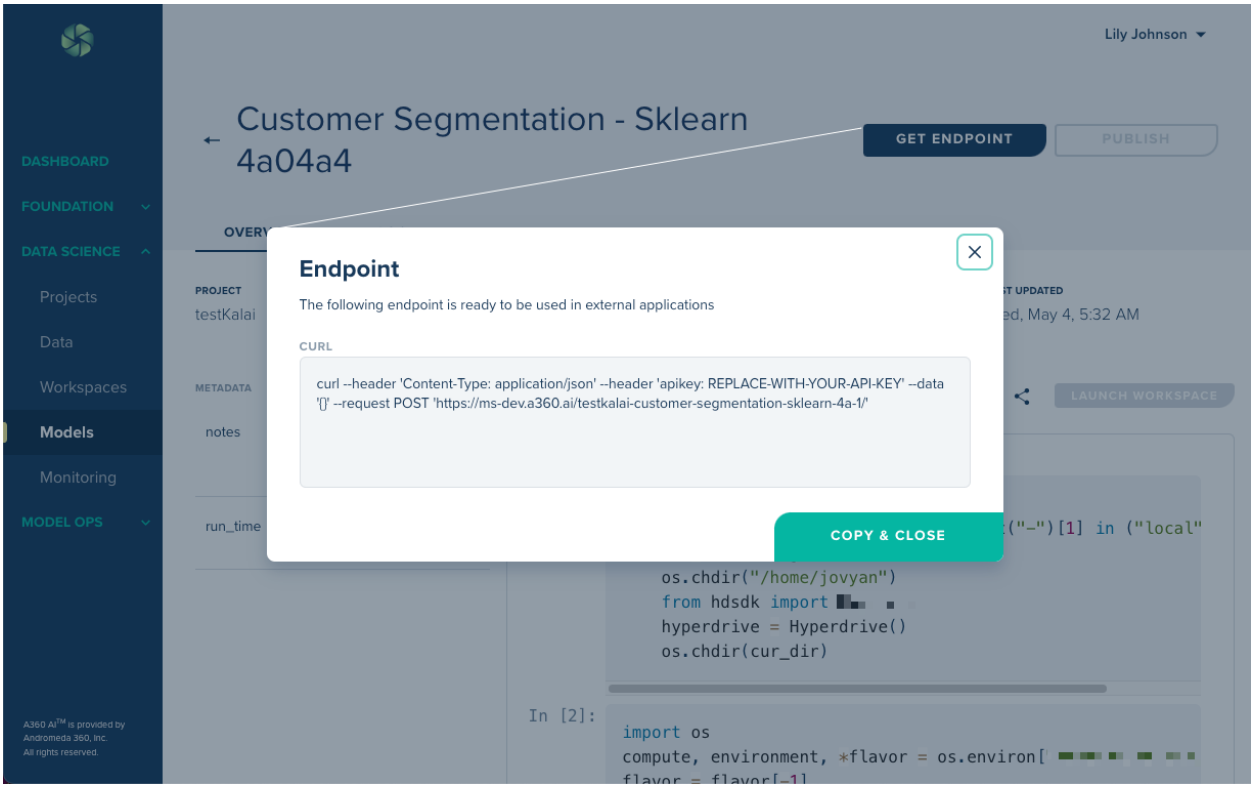
c. Publish - Submits the request and processes the publishing request.

When the operation has successfully been completed, the status of the model will be updated to Published and the Ai application endpoint will become available.

If the operation encounters an issue and fails, the status displayed will be Publish Failed and the option to Publish will be enabled so you can try again. Logs are viewable from the model detail page for troubleshooting purposes.

Endpoint Retrieval

Once you've successfully published a model, you will be able to access the endpoint information from the model detail page. A URL from the system-generated "Successful Publishing" email notification will direct you to the model detail page as well.



NOTE - Make sure you have saved your API key somewhere safe before you copy the endpoint. Clicking on Copy & Close will overwrite anything you've previously copied to your clipboard.

1. Copy and paste the endpoint to a word or .txt file.
2. Take your previously saved API key and replace *REPLACE-WITH-YOUR-API-KEY* in the endpoint URL.
3. Then replace the { } in the endpoint URL with your input data string in JSON format.

The endpoint URL can now be used to send and receive data from the deployed AI application.

Example of Endpoint URL Before Editing:

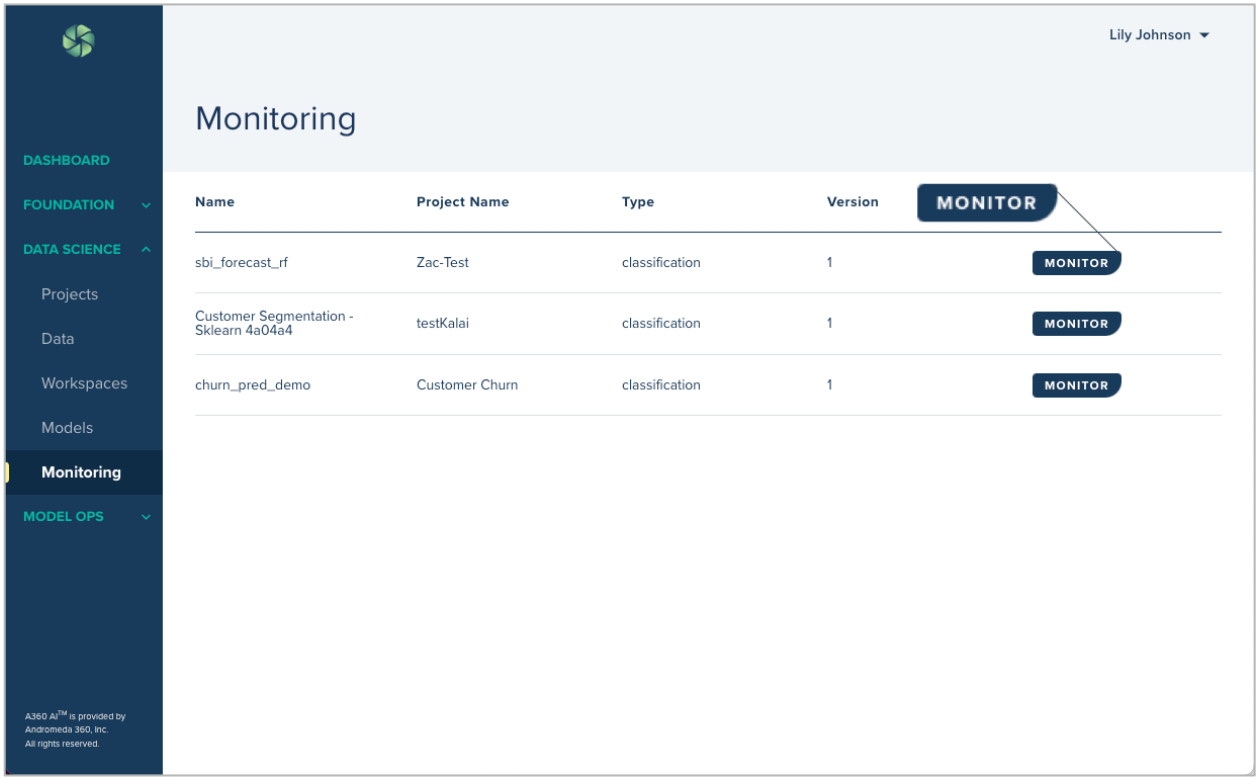
curl --header 'Content-Type: application/json' --header 'apikey: REPLACE-WITH-YOUR-API-KEY' --data '{}' --request POST 'http://af78546dfb34f49cb8ef7d2fef80ed2a-5fc3a25da333f67e.elb.us-east-2.amazonaws.com/monitoring-test-boston-housing-demo-2-1/'

Example Endpoint URL After Editing with API Key:

```
curl --header 'Content-Type: application/json' --header 'apikey: 1efea9a7-3dd3-4ede-92a0-7c9afc554a1d'
--data '{
        "avg(BMI)": "30",
        "avg(active_heartrate)": "100",
        "avg(resting_heartrate)": "86",
        "avg(V02_max)": "18"
}' --request POST 'https://ms.a360ai.solutions/grocery-chain-progre-grocery-chain-xgboost-539672-1/'
```

Monitoring Models

Once a model has been successfully published to a dev or test environment, some monitoring data will start to populate the monitoring dashboard. However, model monitoring metrics will not be populated until the actual model endpoint is called by an external function. You can access the monitoring dashboard by navigating to Monitoring in the Data Science console.

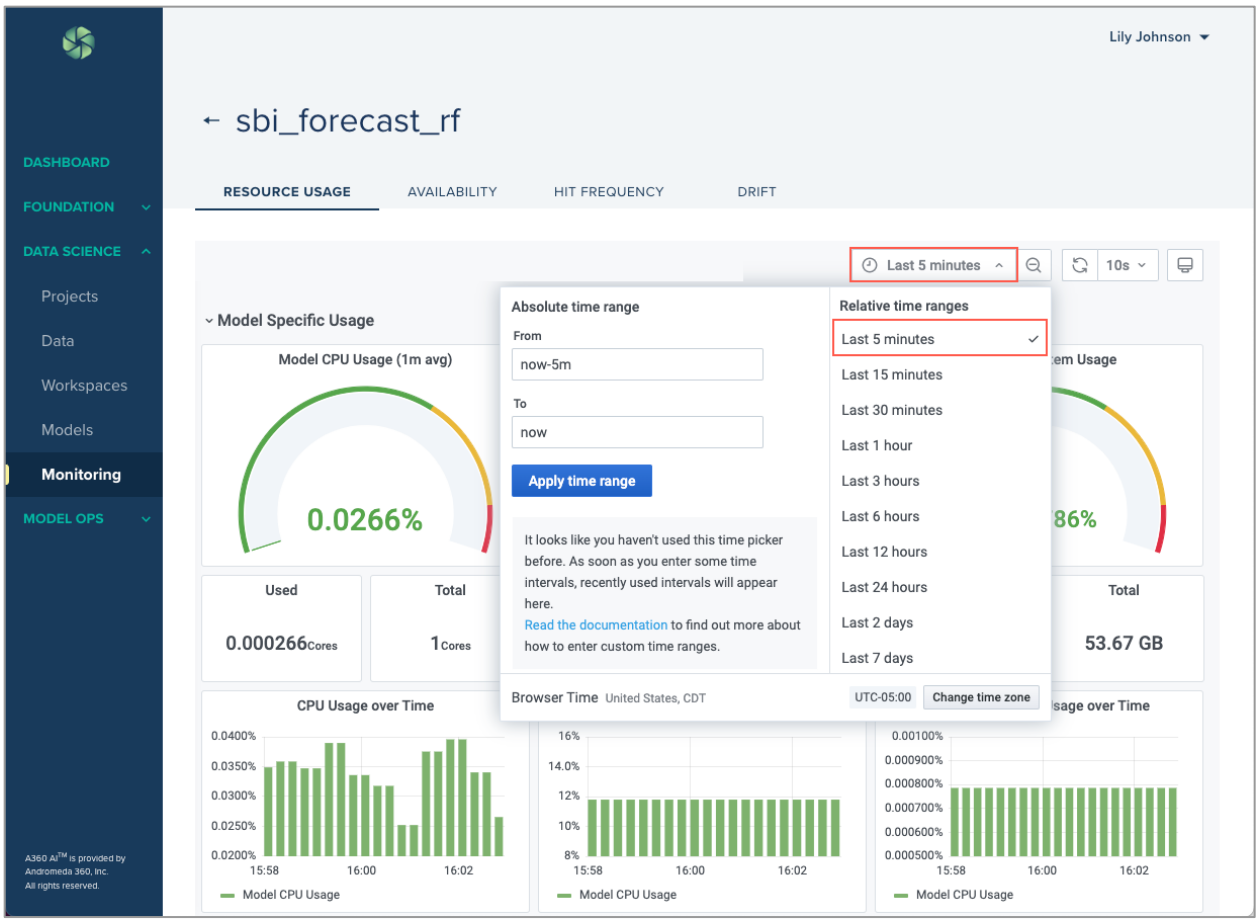


The Monitoring pane has four tabs contain dashboards for the following:

- Resource Usage - Cluster CPU, Memory, and Disk.
- Availability – Uptime.
- Hit Frequency
- Drift (Data and Concept Drift)

Selecting the Monitoring Time Range

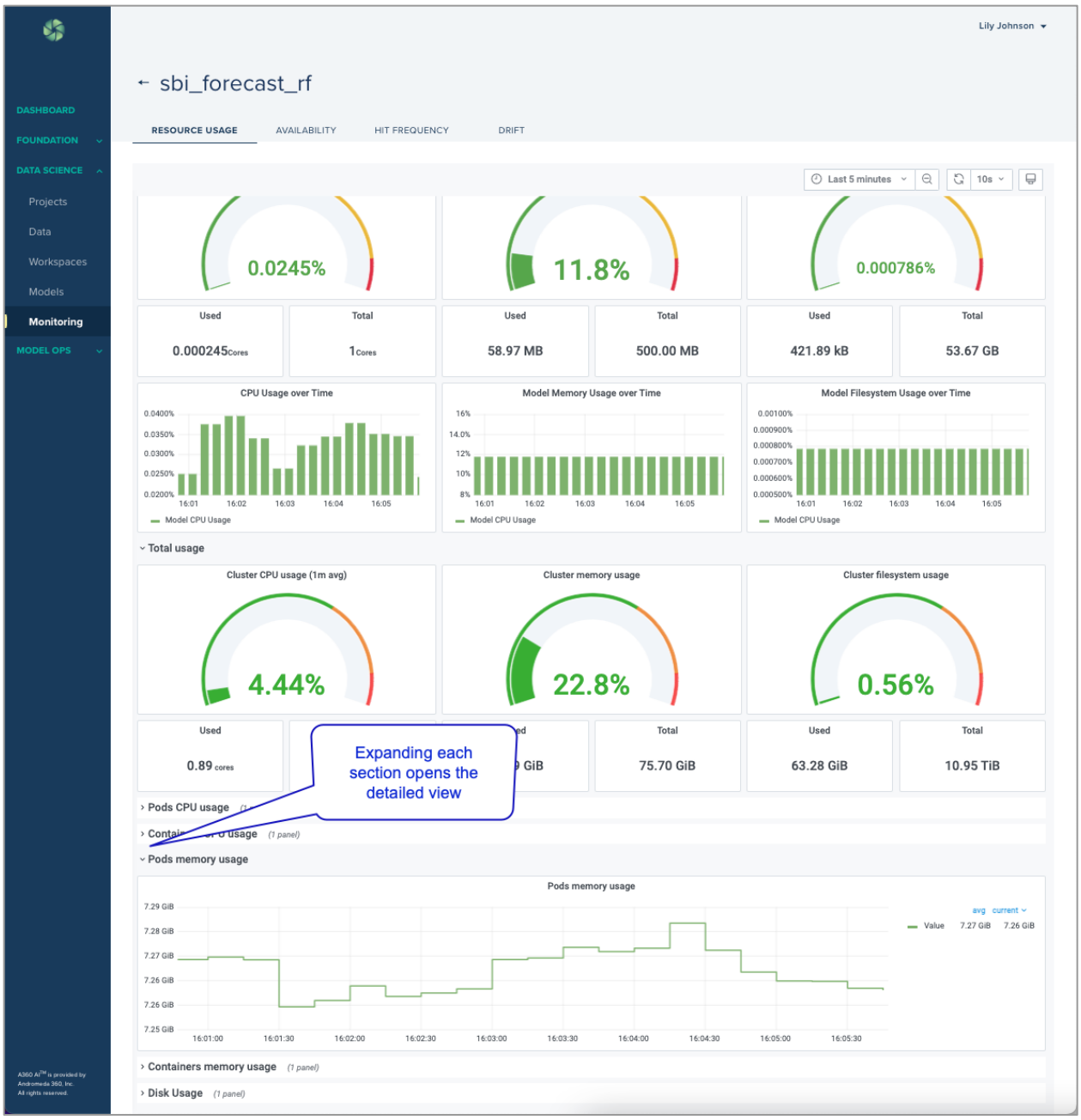
At the top of each of the monitoring pages you can adjust the time period for which you would like to see results.



Resource Usage

The Resource Usage dashboard displays the Memory, CPU, and File system usage of the models and the model serving cluster in a gauge view. To see more detailed information, each section can be expanded to view a time series chart of each metric.

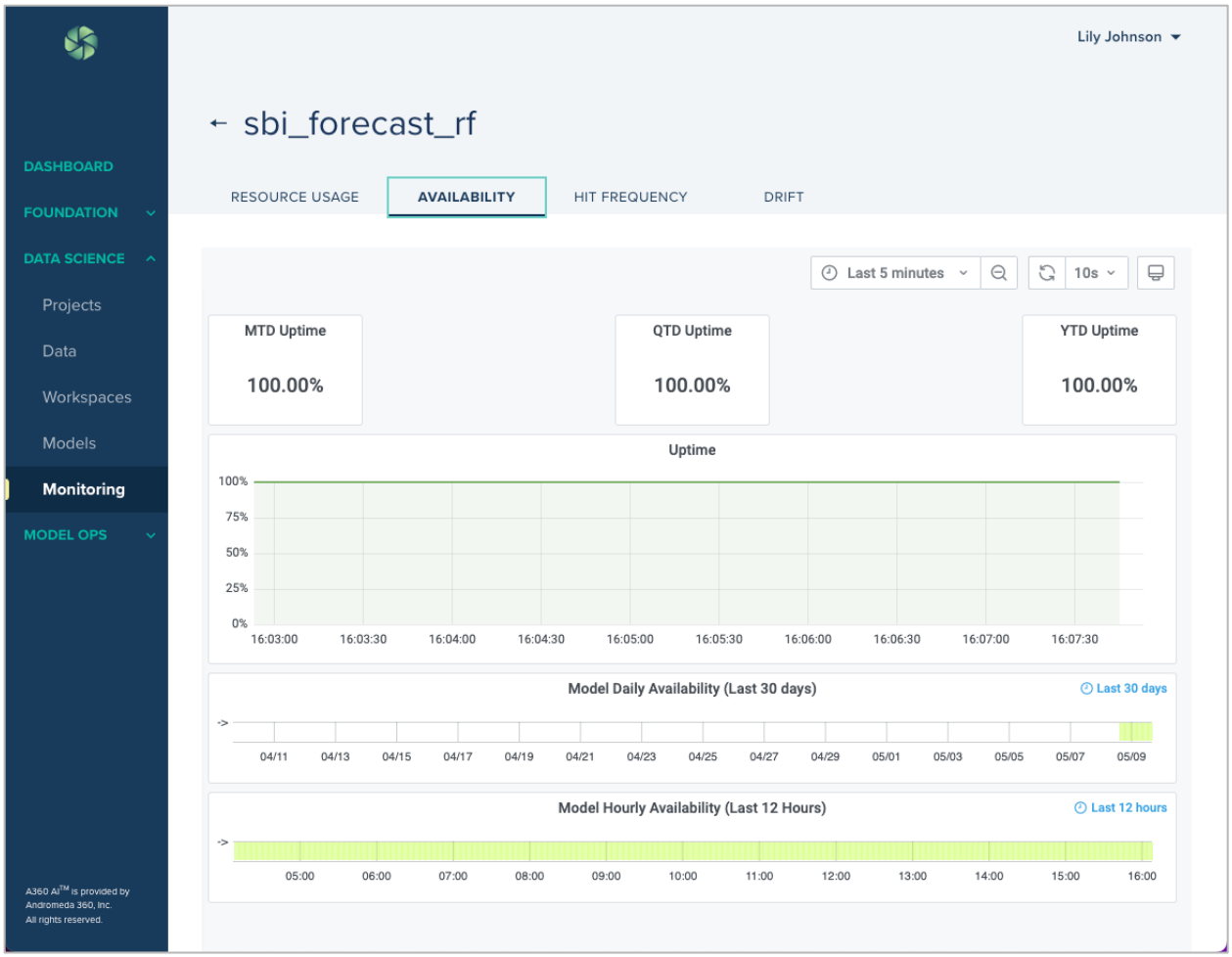
Each of the gauge metrics provide a quick view of usage for the time period selected, as well as the total usage recorded.



Availability

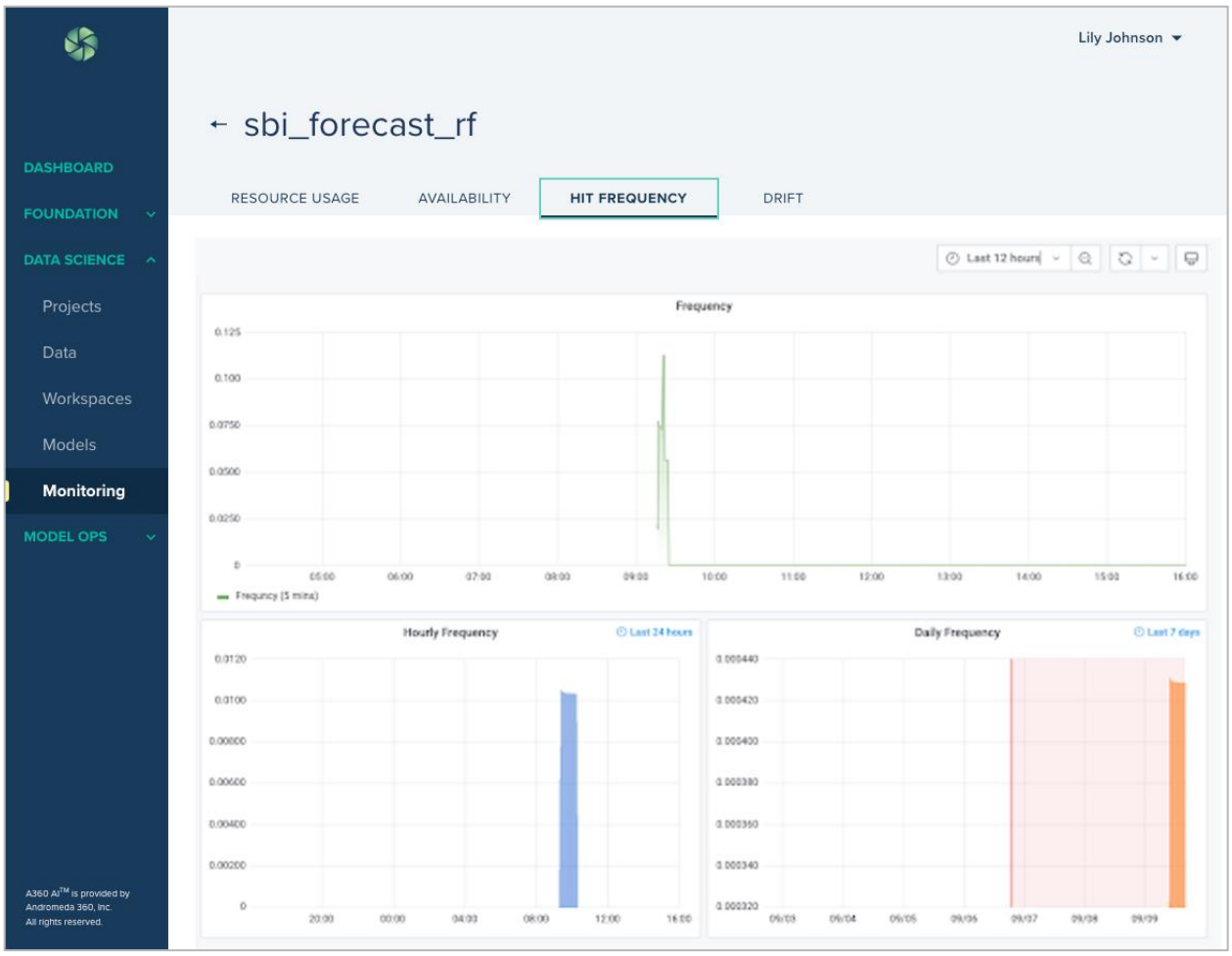
Availability helps users track infrastructure status, i.e. the serving environment is on-line and the endpoint is being reached successfully. This dashboard shows the Year-to-Date (YTD), Quarter-to-Date (QTD), and Month-to-Date (MTD) availability.

NOTE - *There may be a delay of up to 30 minutes after models have been deployed before availability is displayed.*



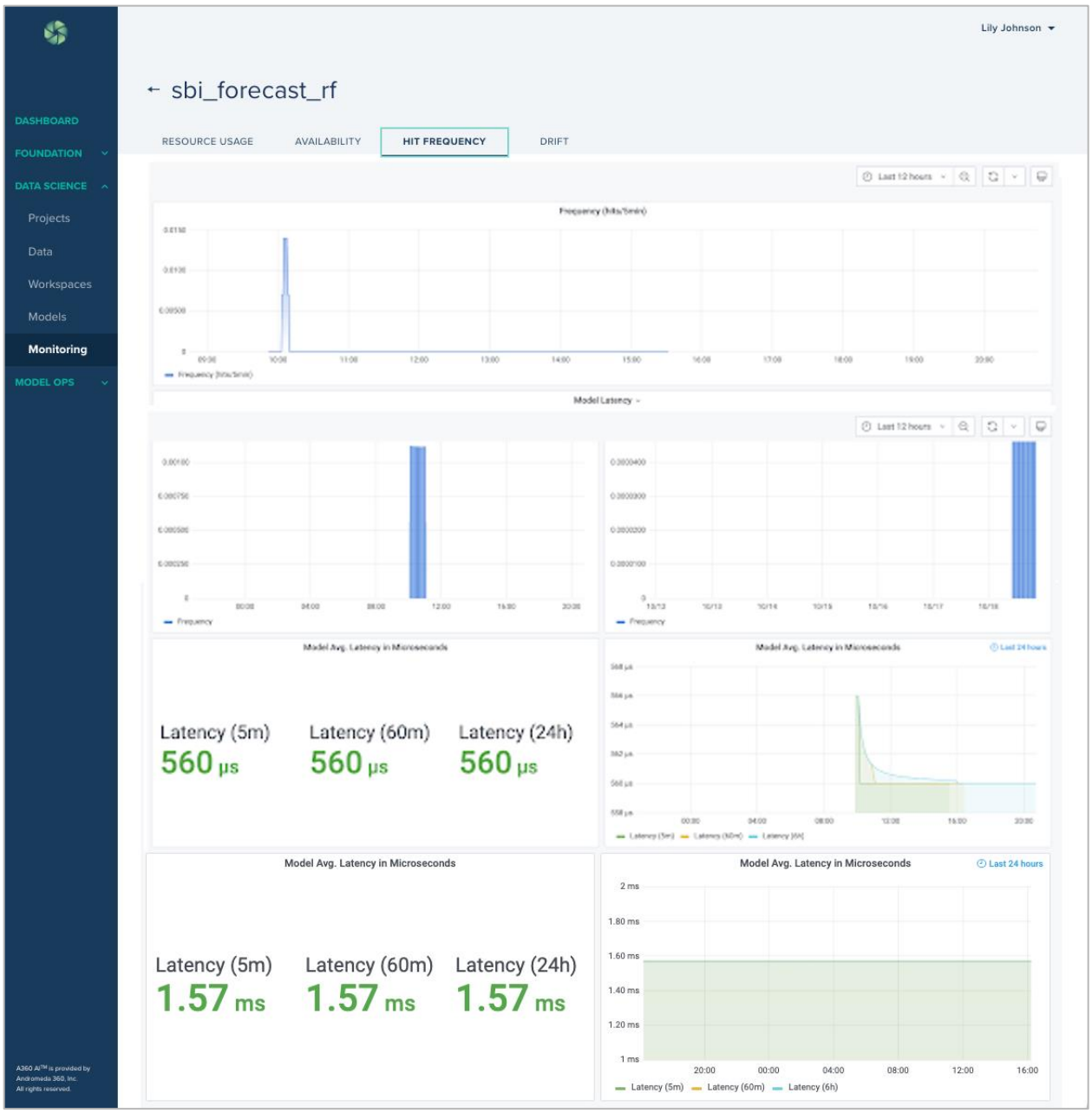
Hit Frequency

The model's fit frequency is a measure of how many times the model endpoint is invoked by the user over time. The Hit Frequency tab displays three panels: 1) Hit per every 5 minutes, 2) Hits per hour over the last 24 hours, and 3) Hits per day over the last 7 days.



Latency

The latency metric measured in microseconds allows the Data Scientist to monitor the overall time it takes to get a result when they call the model prediction function. The Data Scientist can measure Latency at any given time over the available relative time ranges. Average Latency measured over 5m, 60m and 24-hour time period is also provided.

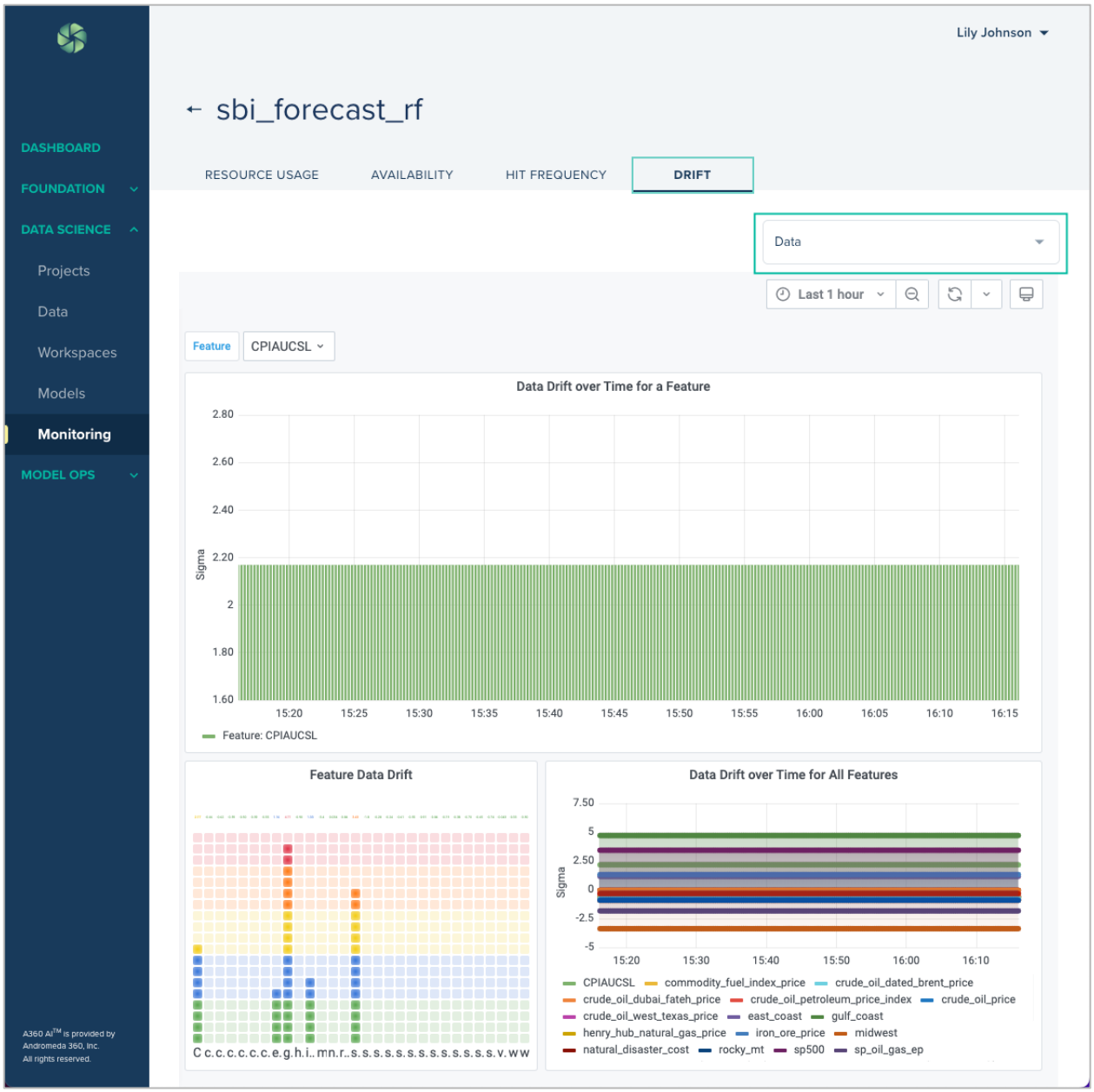


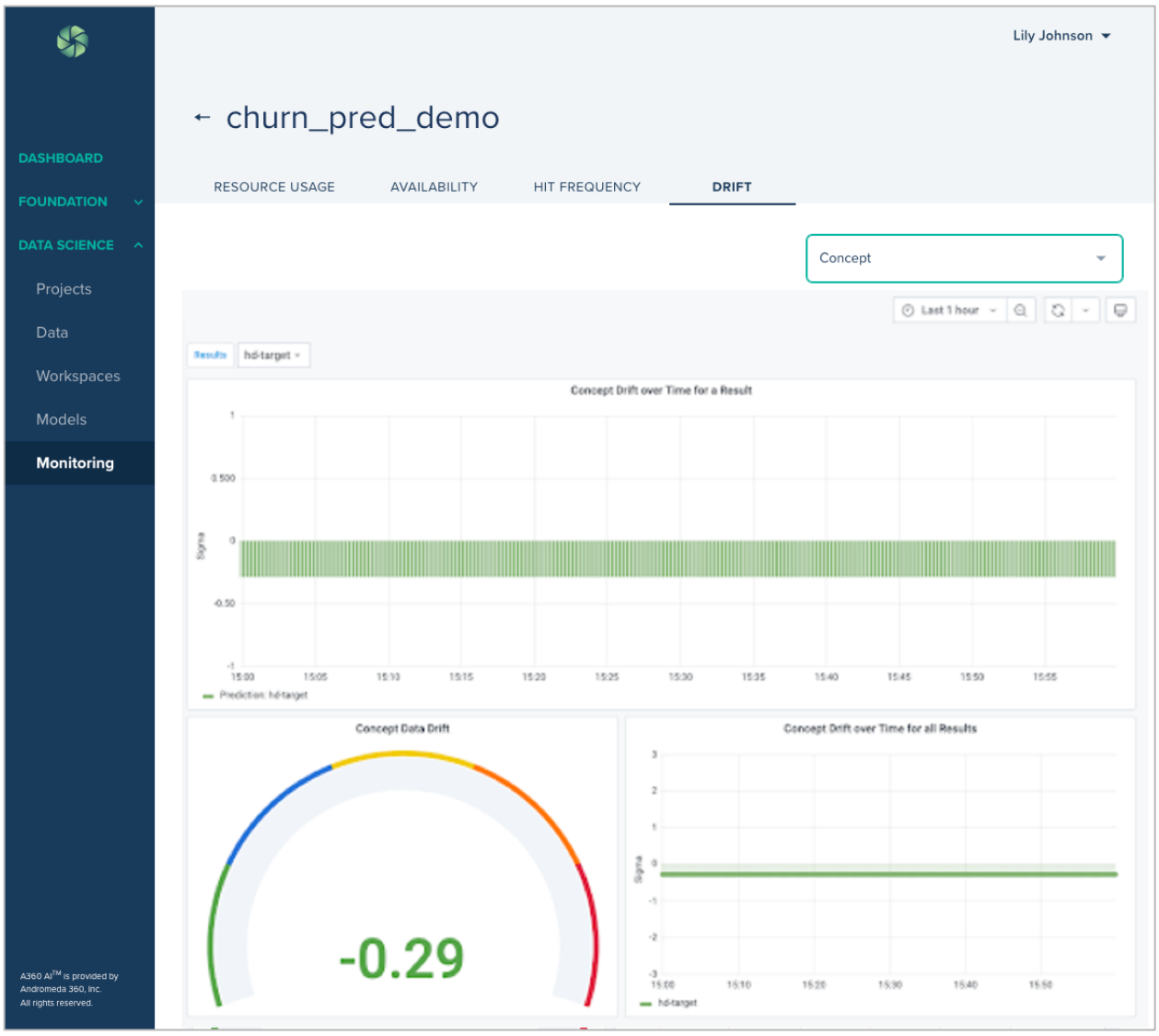
Drift

Data drift and concept drift are both an option in this view. You can toggle between them by selecting the drop down above the top chart.

- Data Drift – Data drift identifies the deviation of the current data stream from the mean data values used to train the model. Data drift is not available for imported models that where training data was not uploaded.

- Concept Drift – Concept drift identifies the deviation of the current model prediction output from the mean prediction values outputted during model training.





Glossary

API Token - An environment-specific authorization token to communicate with the deployed model, i.e. the Docker container/AI application.

Data Repo - A collection of persistent data assets kept in customer object storage with metadata (incl. versioning/branching) managed by A360 AI.

Data Source - External data source/integration e.g., Snowflake database, Big Query, Postgres via ODBC.

Endpoint - A URL for web services and APIs for the unique string location where the API places calls for information.

MDK – Abbreviation of Model Development Kit.

ONNX - An opensource library built to represent machine learning models in serialized format. All A360 AI models are serialized as ONNX models during the packaging step and then included in the deployed Docker container. The deployment runtime then does not need to worry about package dependencies because all models are ONNX models and only require the ONNX runtime at deployment.

Package - The process of compiling the model and collecting its dependencies before publishing.

Publish - The process of deploying a packaged model in a target environment which includes the RESTful endpoint.