INTRODUCTION TO WANDB

Weights & Biases

By: N.S.Akshara

WHAT IS WANDB?

WandB (Weights and Biases) is a powerful ML Operations platform that helps data scientists and ML engineers track, visualize, and manage their experiments seamlessly.

📊 Track Everything

Metrics, hyperparameters, model versions, and system resources

📈 Visualize in Real-Time

Interactive dashboards updated during training

🤝 Collaborate

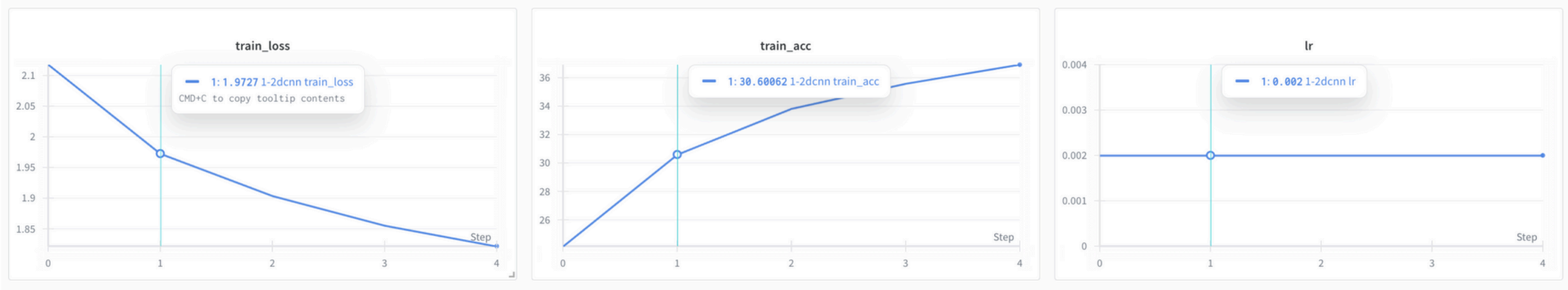
Share experiments with teammates easily

🔧 Integrates Seamlessly

Works with PyTorch, TensorFlow, scikit-learn, and more

KEY FEATURES OF WANDB

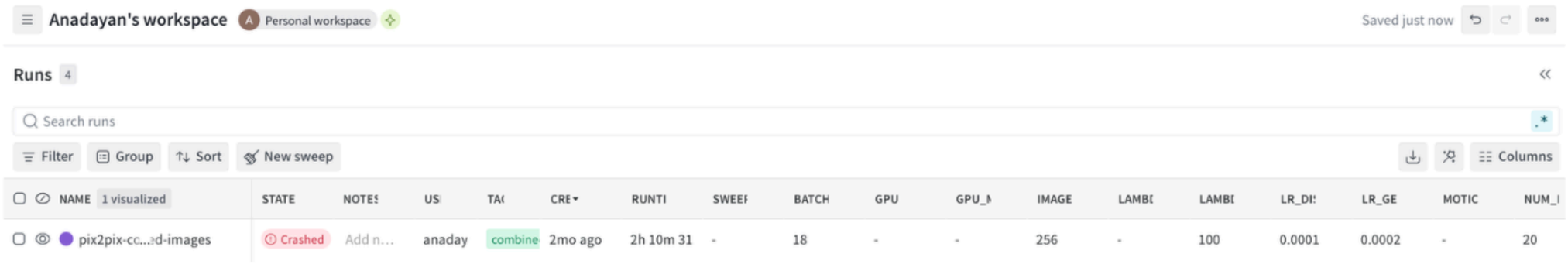
Real-time Performance Logging



- Visualize metrics like Loss and Accuracy F1 in real-time as the model trains.
- Keep track of different models and training runs.
- Spot overfitting or exploding gradients instantly using interactive charts.

KEY FEATURES OF WANDB

Crash Recovery & Resumption



- WandB can save your model's checkpoints to the cloud.
- By passing the specific Run ID, you can download the weights and resume the epoch exactly where you left off.

KEY FEATURES OF WANDB

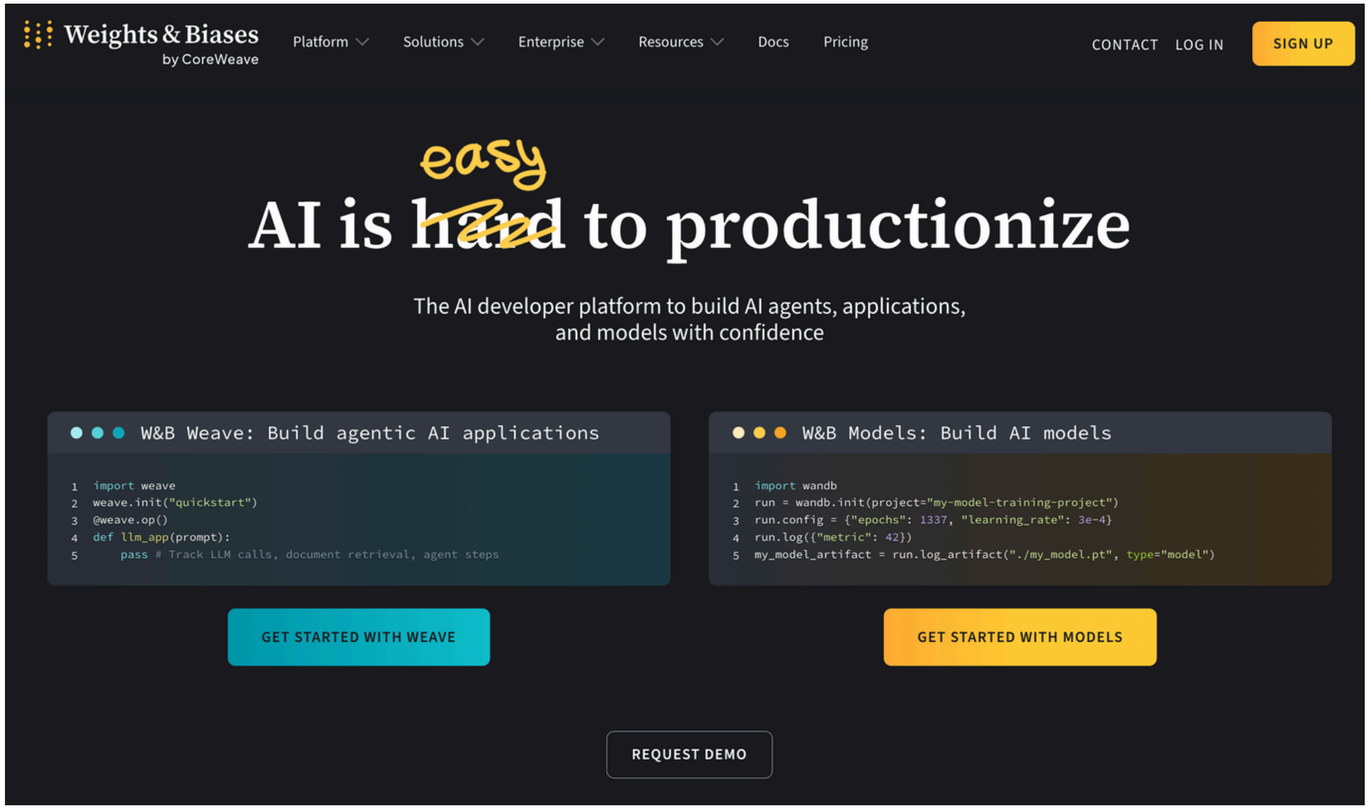
Hyperparameter Tuning

- Automating the search to find the best configuration (Learning Rate, Batch Size, Optimizer).
- Let WandB try different configurations while you sleep!

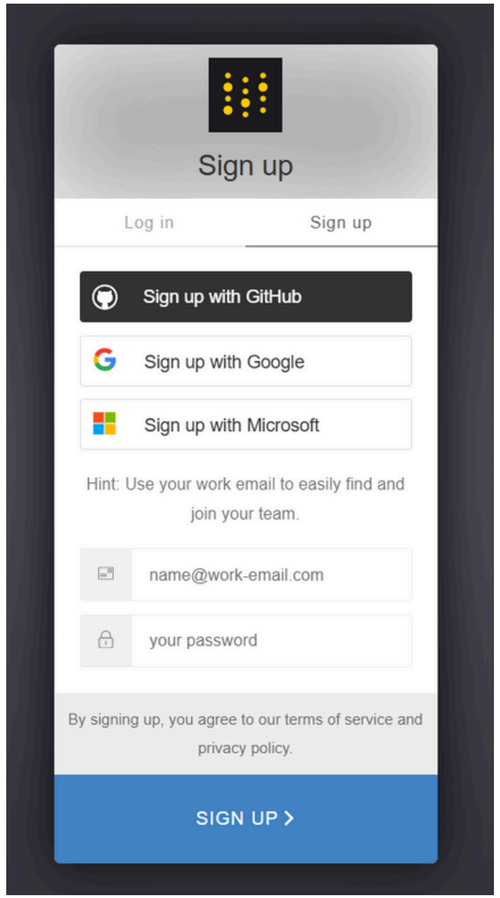
Collaboration

- All team members log runs to the same "Project" bucket.
- See everyone's runs side-by-side to determine whose model is performing best.

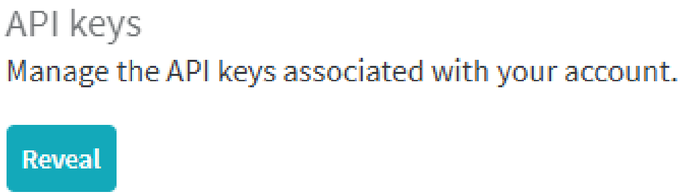
SET UP WANDB ACCOUNT



https://wandb.ai/site/



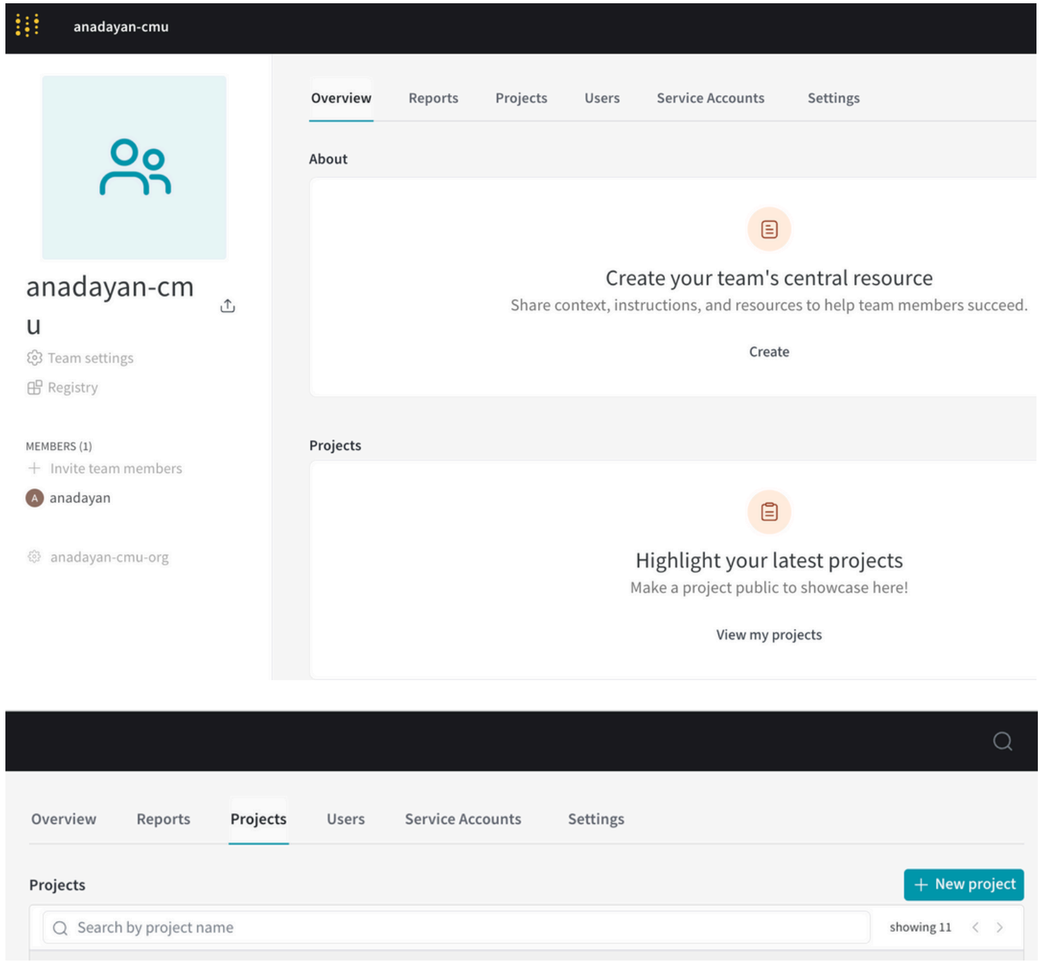
SET UP WANDB API KEY

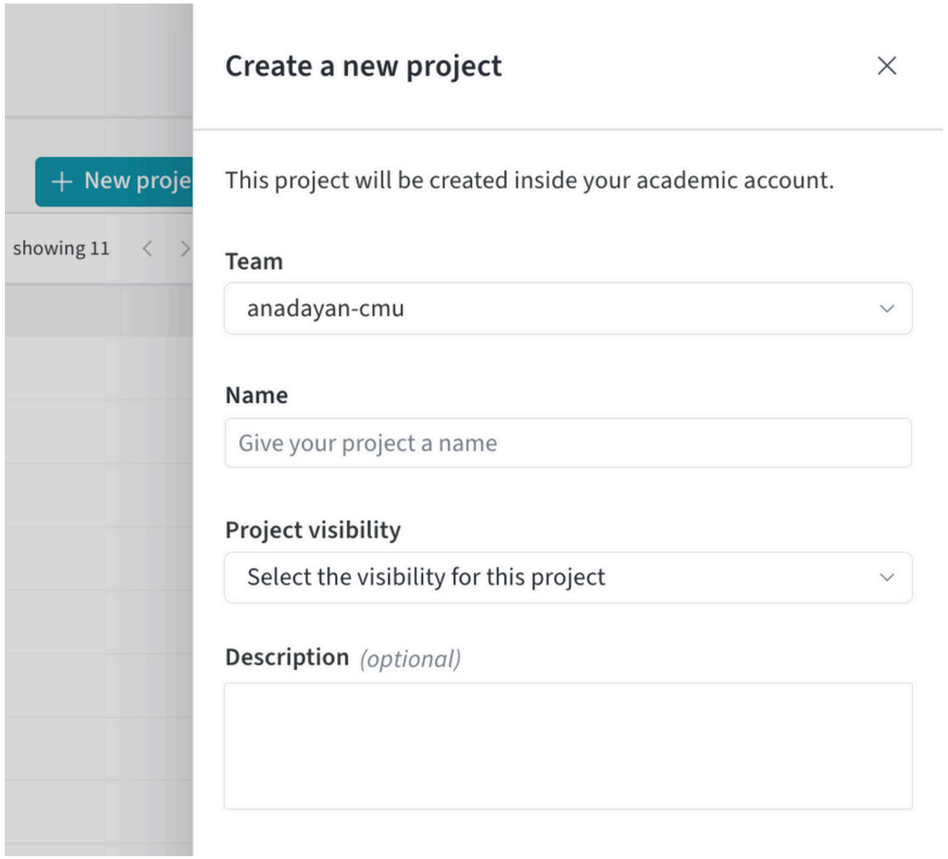


- Your API key authenticates your machine/notebook so runs can upload to your WandB dashboard.
- You can find your key in the “API Keys” section in your WandB user settings.

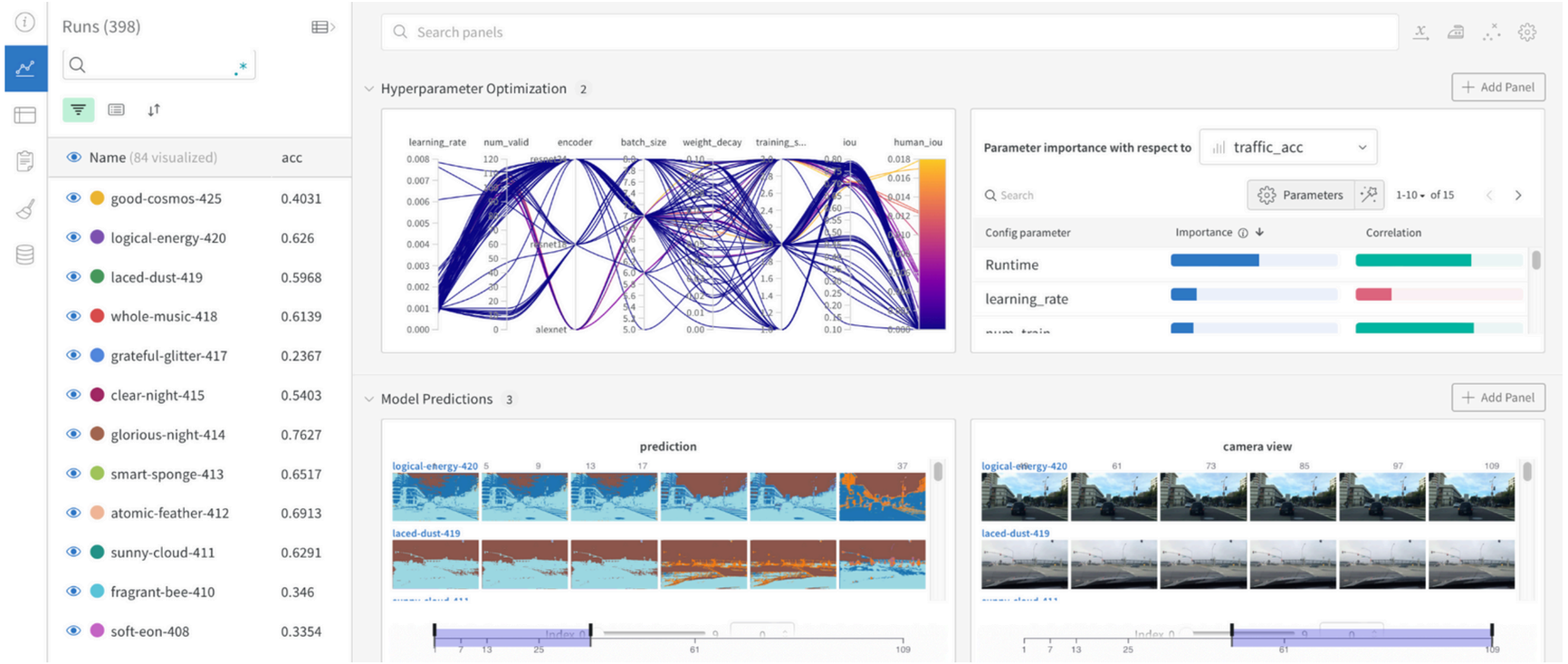
Never post your API key in GitHub, Piazza, screenshots, or shared notebooks!
→ If it does get leaked, please rotate the key immediately in WandB settings.

WANDB - NEW PROJECT

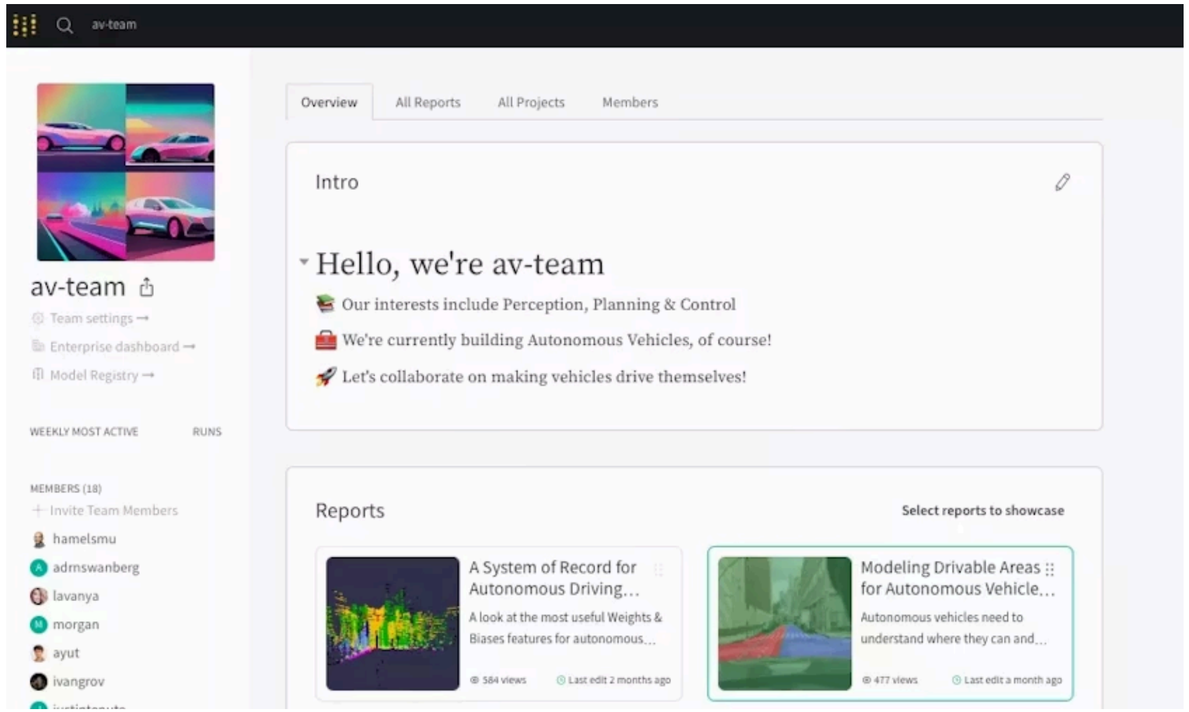




WANDB DASHBOARD



COLLABORATING ON WANDB



Source: WandB documentation

```
# "entity" = The Team Name
wandb.init(entity="intro-dl-group-4",
           project="final-project")
```

- Go to Settings > Create New Team. Invite members via email or username.
- Make sure to use the WandB init call used for the team when using it!

COLLABORATING ON WANDB

Roles

Add a new role to your organization with customized permissions.

Viewer

View-Only members can view assets within the team such as runs, reports, and workspaces. They can follow and comment on reports, but they can not create, edit, or delete project overview, reports, or runs.

Member

A regular member of the team. A team member is invited by email by the team admin. A team member cannot invite other members. Team members can only delete runs and sweep runs created by that member.

Admin

Team admins can add and remove other admins or team members. They have permissions to modify all projects and full deletion permissions.

Custom roles

Create a role by inheriting a pre-defined role and adding permissions to it.

+ Create a role

Contact sales to upgrade and access this feature.

INSTALLATION & NOTEBOOK SET UP

Wandb Installation for Python

Installation and Libraries

```
[ ]  ## Installing WandB
     !pip install wandb -qqq
```

Notebook Setup

- Dataset: CIFAR10
- Neural Network: Convolutional Neural Network
- All functionalities are built in functional blocks for automated access.

Credits to Fall 2024 TAs for the Notebook!

NOTEBOOK: WANDB LOGIN

```
[ ]  import wandb, os
     os.environ['WANDB_API_KEY'] = "7328d336610dec777bdecff906dc57e2a464d0b4"#your key here
     wandb.login()
```

```
wandb: Using wandb-core as the SDK backend.  Please refer to https://wandb.me/wandb-core for more information.
wandb: Logging into wandb.ai. (Learn how to deploy a W&B server locally: https://wandb.me/wandb-server)
wandb: You can find your API key in your browser here: https://wandb.ai/authorize
wandb: Paste an API key from your profile and hit enter, or press ctrl+c to quit: ··········
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
True
```

NOTEBOOK: INITIALIZING PROJECT

```
run_config = {
    'model': '1-2dcnn',
    'optimizer':'sgd',
    'lr': 2e-3,
    'batch_size':64,
    'epochs': 5
}

train_loader, test_loader = build_data(run_config['batch_size'],
                                       data_train, data_test)

optimizer = get_optim(run_config['optimizer'], run_config['lr'], model)

criterion = nn.CrossEntropyLoss()

scaler = torch.cuda.amp.GradScaler()
```

```
run = wandb.init(
    #entity="wandb-starter",
    project="wandb-quickstart",
    #job_type="model-training",
    name=run_config['model'],
    config=run_config
    )
```

Tracking run with wandb version 0.23.1
Run data is saved locally in `/content/wandb/run-20251214_062711-h4lcg101`
Syncing run **1-2dcnn** to Weights & Biases (docs)
View project at https://wandb.ai/anadayan-cmu/wandb-quickstart
View run at https://wandb.ai/anadayan-cmu/wandb-quickstart/runs/h4lcg101

NOTEBOOK: LOGGING & SAVING MODELS

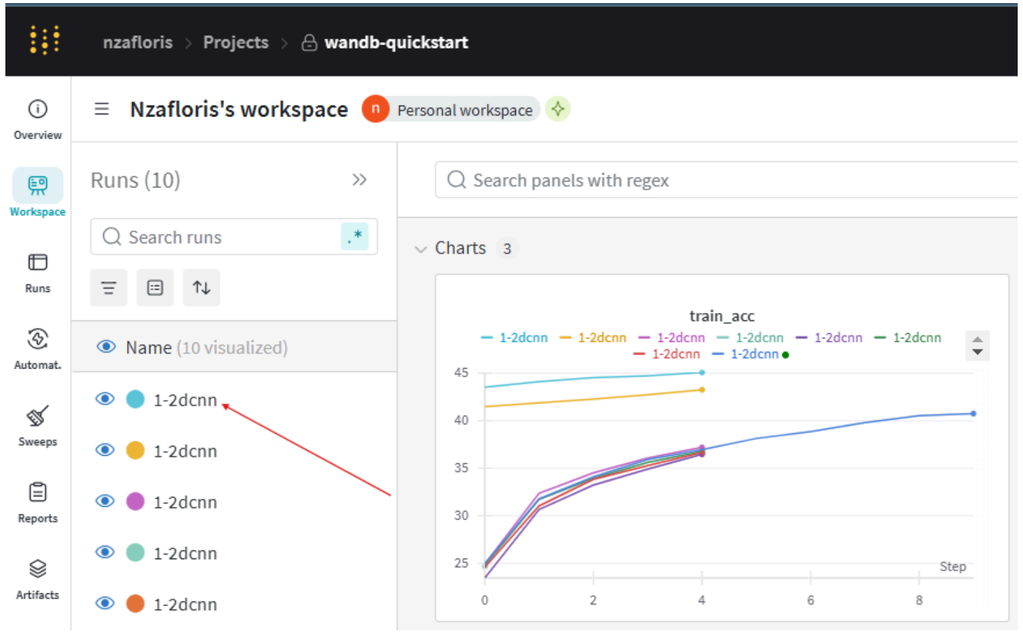
```
metrics = {
    "train_loss":train_loss,
    "train_acc": train_acc,
    'lr': lr
}
```

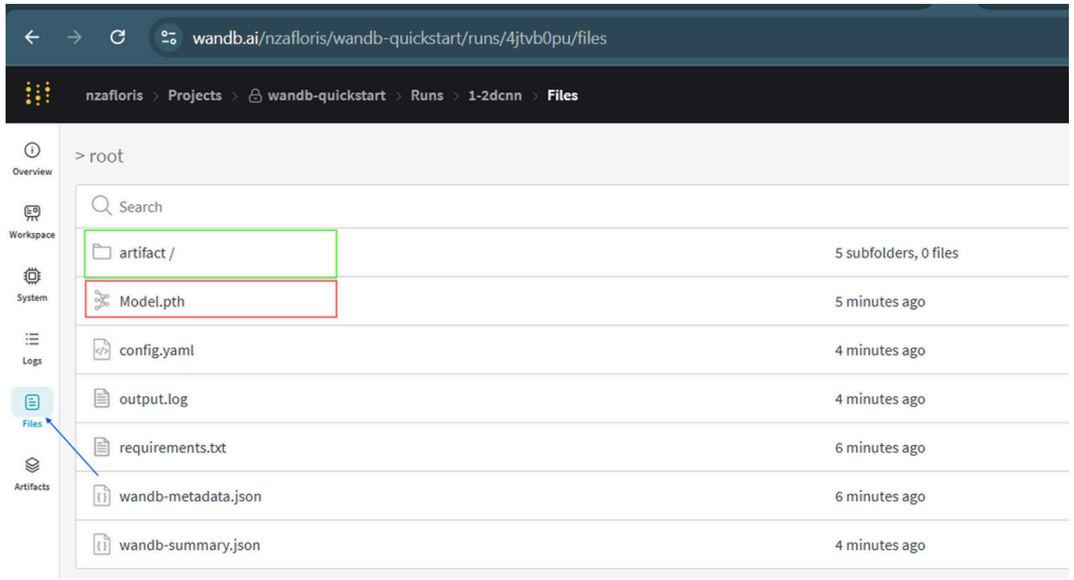
Logging metrics

```
        torch.save({
                'model_state_dict': model.state_dict(),
                'optimizer_state_dict': optimizer.state_dict()
                }, "Model.pth")

        # ALTERNATIVE 1: Saving Files as Artifacts
        # Creating Artifact
        model_artifact = wandb.Artifact(run_config['model'], type='model')

        # Adding model file to Artifact
        model_artifact.add_file("Model.pth")

        # Saving Artifact to WandB
        run.log_artifact(model_artifact)

        # ALTERNATIVE 2: Saving Files as Files
        wandb.save("Model.pth")

if finish:
  wandb.finish()
```

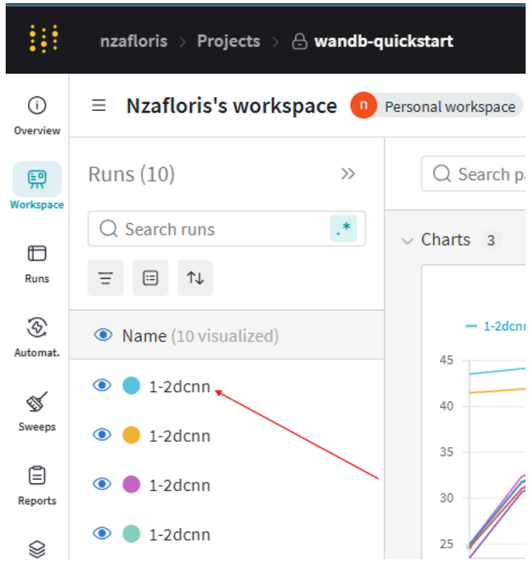
Saving models

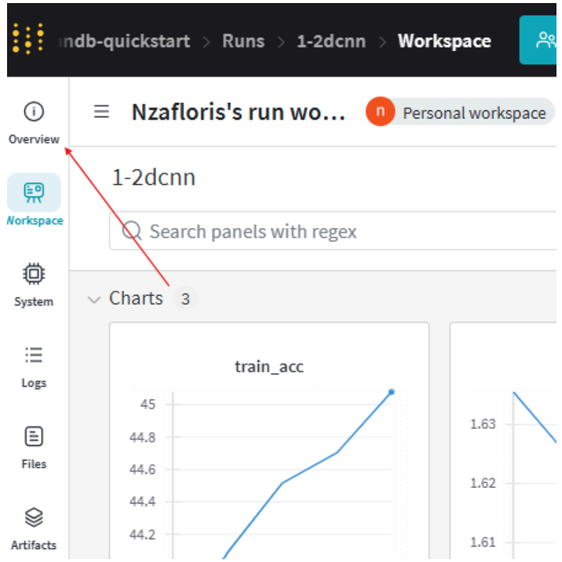
NOTEBOOK: LOGGING & SAVING MODELS

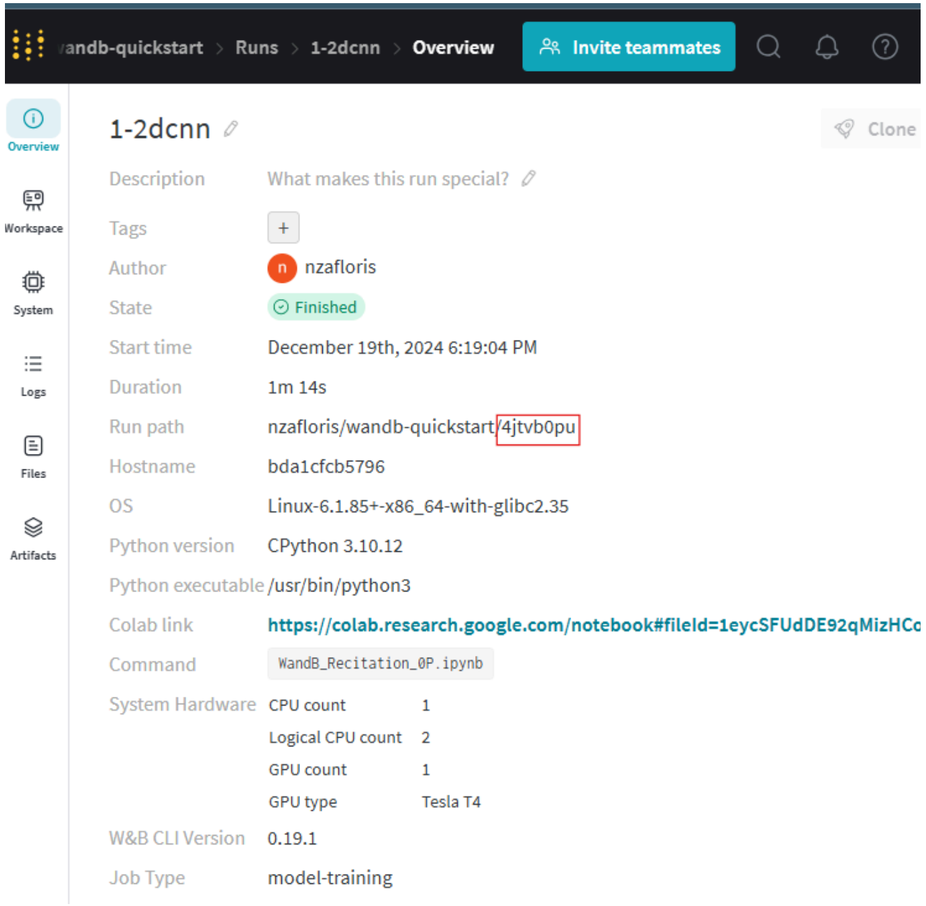




NOTEBOOK: GET RUN ID







NOTEBOOK: RESUMING RUN

```
run_id = "4jtvb0pu" ### Replace with run id string
run = wandb.init(
    id     = run_id, ### Insert specific run id here if you want to resume a previous run
    resume = "must", ### You need this to resume previous runs, but comment out reinit = True when using this
    project = "wandb-quickstart", ### Project should be created in your wandb account
)
```

- We use resume='must' in the code. This is a safety feature which prevents you from accidentally starting a new run instead of resuming.
- An often noticed mistake is loading the model.load_state_dict but forgetting optimizer.load_state_dict. If you forget the optimizer, you lose your momentum and learning rate information!

HYPERPARAMETER SWEEPING

Hyperparameters refer to parameters set before learning begins. They control the behavior of the training algorithm!

Examples

- Learning Rate
- Batch Size: Samples processed before update.
- Optimizer: Adam, SGD, RMSProp.
- Dropout Rate: Probability of ignoring neurons.

NOTEBOOK: HYPERPARAMETER SWEEPING

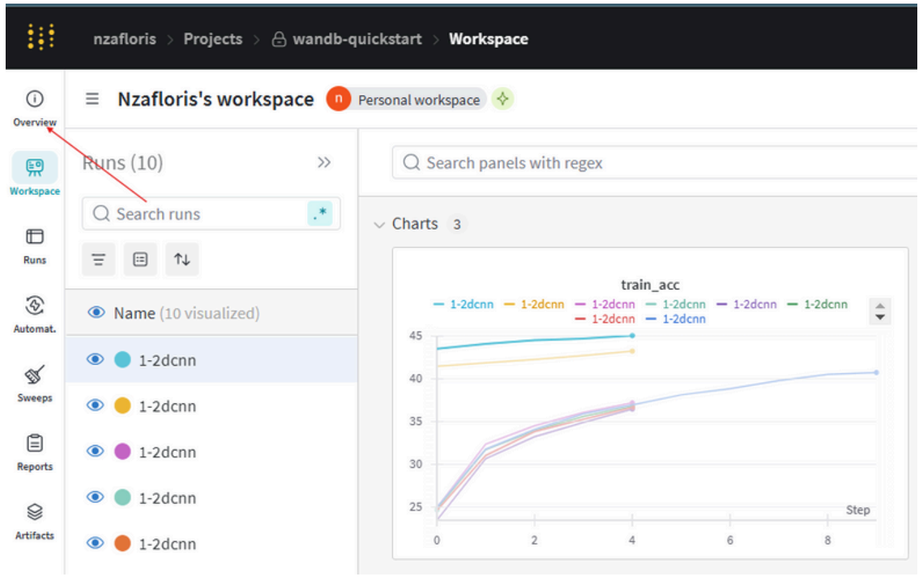
```
[ ]  # Hyperparameters to work with

     parameters_dict = {
         'optimizer':{
             'values': ['sgd', 'adam']
         },
         'learning_rate':{
             'distribution':'uniform',
             'min':2e-4,
             'max':1e-1
         },
         'batch_size': {
             'distribution': 'q_log_uniform_values',
             'q':4,
             'min': 16,
             'max': 128
         },
         'epochs':{
             'value': 5
         }
     }
     sweep_config['parameters'] = parameters_dict
```

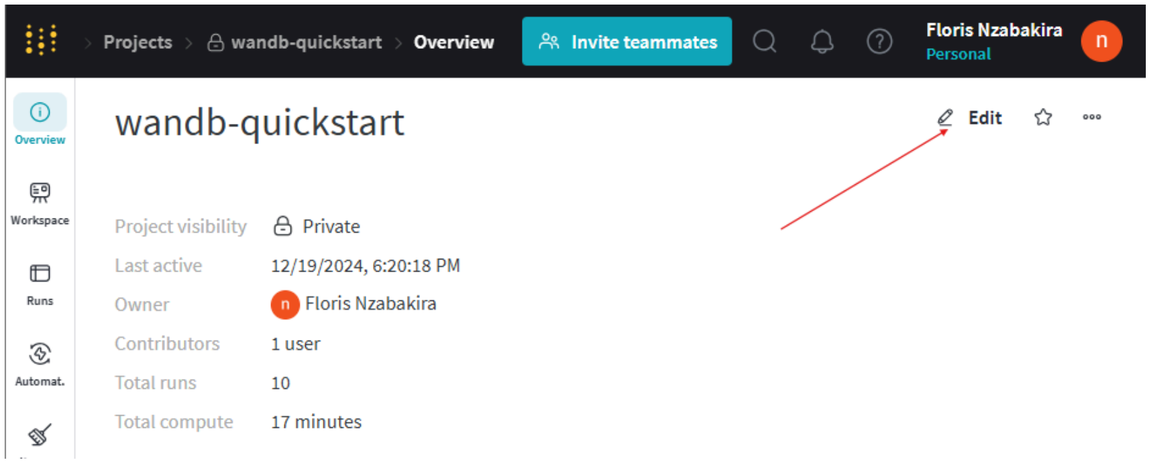
```
[ ]  # Initalizing the sweep

     sweep_id = wandb.sweep(sweep_config, project="CIFAR-Sweep2")
```

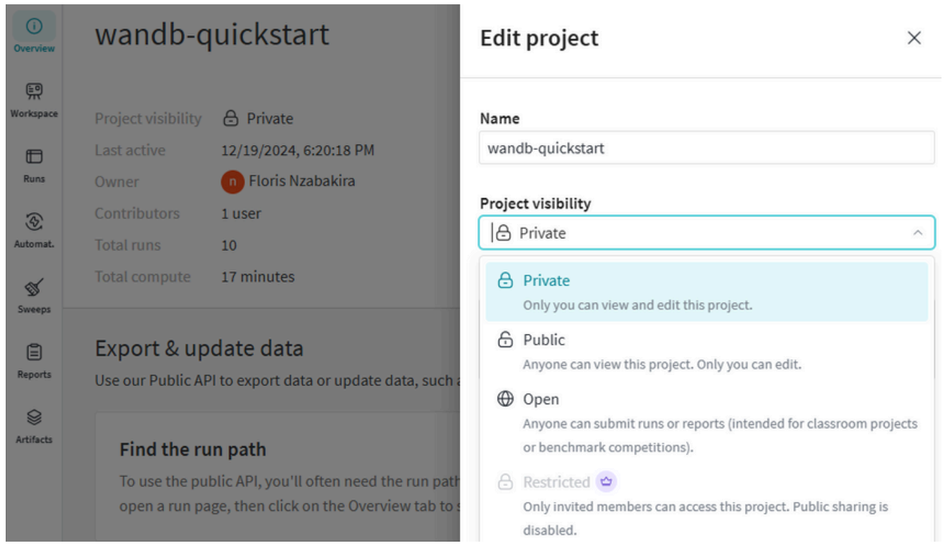
```
def train_sweep(config = None):
    with wandb.init(config=config) as run:
        run.name=f"Jeel_{wandb.config.learning_rate}_{wandb.config.batch_size}_{wandb.config.optimizer}"
        config = wandb.config

        train_loader, test_loader = build_data(config.batch_size, data_train, data_test)

        model = Network().to(device)

        optimizer = get_optim(config.optimizer, config.learning_rate, model)

        criterion = nn.CrossEntropyLoss()

        scaler = torch.cuda.amp.GradScaler()

        for epoch in range(config.epochs):

            model, loss = train_epoch(model, train_loader, optimizer, criterion, scaler)

            wandb.log({'loss': loss})
```

```
[ ]  # Running the sweep

     wandb.agent(sweep_id, train_sweep, count=2)
```

MAKING YOUR WANDB PROJECT PUBLIC







WANDB STORAGE FOR ACCOUNTS

Once your WandB free storage is full, you won't be able to access your runs!

- Individual Account:
  - Provides 100GB free
  - Can be accessed through https://wandb.ai/subscriptions

- Group Account:
  - Provides 5GB free per team
  - Can be accessed through: https://wandb.ai/account-settings/<Team-Name>/billing

THANK YOU!