If you are in ubuntu/macos, you can use the default terminal.
If you are in windows, you may have to use google cloud SDK shell.

Also, you can use the CLOUD SHELL Terminal that appears in the cloud console.

Install gcloud

See the instructions here.
Make sure you set the default project where you have received the allocation from TRC.

Start a TPU VM

Here is a sample command that starts a TPU VM

```
gcloud alpha compute tpus tpu-vm create vm_name --zone=us-central1-f
--accelerator-type=v2-8 --version=tpu-vm-tf-2.8.0
```

vm_name->the name I am assigning to the vm
zone-> one of my assigned zones
accelerator type-> depends on your quota, usage, and availability
version=it's advised that you use the latest version of the machine learning libraries (tensorflow or torch)

Check out this link for more detailed instructions

Connect to VM

```
gcloud alpha compute tpus tpu-vm ssh vm_name --zone=us-central1-f
```

Storage buckets

Use storage buckets to store all your files. If you go to the cloud console and search for Cloud Storage, you will see the create bucket option on the top left.

Transfer files to your VM

Keep all your files in a cloud bucket and transfer files to your VM as necessary.

```
gsutil cp gs://bucket_name/filename /home/username/folder_name
```

You can also transfer files from your VM to the bucket as necessary.

```
gsutil cp filename gs://bucket_name
```

Or you can also directly transfer files between your local computer and the VM

Run code in vm

Here is a tensorflow code snippet that utilizes the TPU.

```
# detect TPUs
tpu = tf.distribute.cluster_resolver.TPUClusterResolver.connect()
tpu_strategy = tf.distribute.experimental.TPUStrategy(tpu)

with tpu_strategy.scope():

    #take input
    inputs = tf.keras.Input(shape=[seq_len, 4,])

    # add layers

    # final prediciton layer
    pred = tf.keras.layers.Dense(1)(flatten)

    # define model
    model = tf.keras.Model(inputs=[inputs], outputs= [pred])

    opt = tf.keras.optimizers.Adam(lr) #tf.keras.optimizers.Adam(lr=lr)#
    model.compile(optimizer=opt, loss='mean_squared_error',
                                    metrics = [r_square])
    model.summary()

model.fit(train_data, validation_data = val_data, batch_size = batch_size,
epochs=epochs ,callbacks = callbacks_list, steps_per_epoch =
```

```
len(train_data), validation_steps = len(val_data),validation_batch_size =
batch_size)
```

You can find more information here.

Share the resource with team members

Use IAM to provide access to your team members. Go to the cloud console, search for IAM & Admin and use the add option on the top left.
See the details about different kinds of roles here.