Student Guide - Docker Container Access and Setup

This guide provides essential commands and instructions for connecting to your assigned Docker container and setting up your development environment.

Connecting to Your Container

SSH Connection

Connect to your container using SSH with your assigned username and password:

```
ssh <username>@<server_hostname> -p <port_number>
```

Example:

```
ssh Edward@example.edu -p 50001
```

If you receive a warning about the host key, you can confirm by typing `yes` when prompted.

SSH Connection Issues

If you encounter connection issues, try these troubleshooting steps:

1. Verify your port number and username (username is case sensitive)
2. Check that your password is correct
3. If using Windows, ensure your SSH client (PuTTY, etc.) is configured correctly
4. Ask your instructor if the container is running

Setting Up Python Environment

There are several ways to create and manage Python virtual environments. Choose the method that works best for you.

Method 1: Using standard venv and pip

```
# Create a virtual environment
python3 -m venv myenv

# Activate the virtual environment
source myenv/bin/activate
```

```
# Install packages
pip install numpy pandas matplotlib

# Install packages from requirements.txt
pip install -r requirements.txt
```

Method 2: Using pyenv for Python version management

```
# Install pyenv if not already installed

# Install A Specific Python version
pyenv install 3.12

# Set a specific version globally or locally
pyenv global 3.12  # Global setting
pyenv local 3.9.6   # Project-specific setting

# Create virtual environment with pyenv
pyenv virtualenv 3.12 myproject

# Activate the environment
pyenv activate myproject
```

Method 3: Using uv for faster package installation

```
# Install uv if not already installed


# Create an environment with a specific Python version
uv venv --python=3.12

# Activate the environment
source .venv/bin/activate

# Install packages
uv pip install numpy pandas matplotlib

# Install packages from requirements.txt
uv pip install -r requirements.txt
```

Installing Specific PyTorch Versions

Installing PyTorch

For installing specific PyTorch versions, always visit the official PyTorch website to get the correct installation command:

```
# Visit https://pytorch.org/get-started/locally/
# Select your preferences (PyTorch version, OS, package manager,
        CUDA version)
# Copy the recommended installation command

# Example command :
pip install torch torchvision torchaudio
```

Important Note: Do not modify the local CUDA environment in your container. PyTorch's CUDA support is backward compatible, meaning newer PyTorch versions can work with older CUDA installations. The containers are already configured with appropriate CUDA versions.

For CPU-only installations (when needed):

```
# Get the appropriate command from https://pytorch.org/get-
        started/locally/
# Make sure to select "CPU" for the computation platform

# Example for CPU-only installation:
pip install torch torchvision torchaudio --index-url https://
        download.pytorch.org/whl/cpu
```

Installing Specific TensorFlow Versions

For installing specific TensorFlow versions, always refer to the official TensorFlow website for the most accurate installation commands:

```
# Visit https://www.tensorflow.org/install
# Follow the instructions specific to your needs
# Copy and paste the recommended installation command

# Example of a standard installation:
pip install tensorflow
```

Important Note: Do not modify the local CUDA environment in your container. The containers are already configured with appropriate CUDA and cuDNN versions for TensorFlow. Installing a TensorFlow version compatible with the existing CUDA setup is recommended.

For GPU installations:

```
# Get the appropriate command from https://www.tensorflow.org/
        install/pip

# Example for a version-specific installation with GPU support:
python3 -m pip install 'tensorflow[and-cuda]'
```

For CPU-only installations (when needed):

```
# Visit https://www.tensorflow.org/install/pip#cpu
# Follow the CPU-specific installation instructions

# Example for CPU-only installation:
python3 -m pip install tensorflow
```

Working with the Shared Data Directory

A shared data directory is accessible in your home directory:

```
# Navigate to the shared data directory
cd ~/data

# List contents
ls -la

# Create your own work directory
mkdir ~/data/mywork
cd ~/data/mywork
```

Useful Linux Commands

```
# Check available disk space
df -h

# Check memory and CPU usage
top

# Check GPU status (if available)
nvidia-smi

# Transfer files from your local machine to container (run this
        on your local machine)
scp -P <port_number> local_file.txt
        <username>@<server_hostname>:/home/<username>/

# Transfer files from container to local machine (run this on
        your local machine)
```

```
scp -P <port_number> <username>@<server_hostname>:/home/
        <username>/remote_file.txt ./

# Check Python version
python3 --version

# Exit the SSH session
exit
```

Getting Help

If you encounter issues with your container or need assistance with the setup:

1. Check the error messages carefully
2. Consult your course materials
3. Contact your instructor through the appropriate channels

Remember that your container may have resource limitations (CPU, RAM, disk space). Monitor your usage to ensure your applications run efficiently.