# System Requirements Specification Index

For

# Recognize Digits using TensorFlow

Version 1.0

- The MNIST dataset contains 70,000 grayscale images of handwritten digits (0–9).
- Each image is 28x28 pixels (784 total features when flattened).
- It is available directly from `tf.keras.datasets.mnist`.

---

## 1. `load_and_preprocess() -> tuple`

**Purpose:**
To load, flatten, normalize, and prepare the dataset for training and testing.

**Instructions:**

- Load the MNIST dataset using `tf.keras.datasets.mnist.load_data()`, which returns two tuples: training and testing data.
- Flatten the 28x28 images into vectors of size 784 using `.reshape(-1, 28 * 28)`.
- Normalize the pixel values by dividing by 255.0 to scale them between 0 and 1.
- For faster training, extract only 10,000 training samples using `train_test_split` with `stratify=y` to preserve label distribution.
- Return the `X_train`, `X_test`, `y_train`, and `y_test` arrays.

---

## 2. `build_model() -> tf.keras.Model`

**Purpose:**
To construct and compile a feedforward neural network suitable for classifying digits.

**Instructions:**

- Use `tf.keras.Sequential` to define a model consisting of:
  - A `Dense` layer with 128 units, `'relu'` activation, and `input_shape=(784,)`.
  - A second `Dense` layer with 64 units and `'relu'` activation.
  - A final `Dense` layer with 10 units and `'softmax'` activation for multi-class output (0 to 9).
- Compile the model with:
  - Optimizer: `'adam'`
  - Loss function: `'sparse_categorical_crossentropy'` (for integer labels)
  - Metric: `'accuracy'`
- Return the compiled model.

---

## 3. `train_model(model, X_train, y_train) -> tf.keras.Model`

**Purpose:**
To train the neural network on the prepared dataset.

**Instructions:**

- Train the model using `model.fit()` for 10 epochs.
- Set `batch_size=32` for small batch gradient updates.
- Set `verbose=1` to view training progress.
- Return the trained model for evaluation or further use.

---

## 4. `evaluate_model(model, X_test, y_test) -> None`

**Purpose:**
To evaluate model performance on unseen test data.

**Instructions:**

- Evaluate the model using `model.evaluate()` on the test dataset.
- Extract and print the accuracy as a percentage using Python's `f-string` formatting: `Test Accuracy: 0.9843` (for example).

---

## 5. `predict_sample(model) -> None`

**Purpose:**
To predict the digit label for a single sample from the test set.

**Instructions:**

- Load the first sample from the test set using `tf.keras.datasets.mnist.load_data()`.
- Reshape it to shape `(1, 784)` and normalize it to `[0, 1]` range.
- Use `model.predict()` to get the probability distribution.
- Use `np.argmax()` to find the class with the highest probability.
- Print the predicted digit class using formatted output.

---

## 6. `main() -> None`

**Purpose:**
To serve as the entry point for the script.

**Instructions:**

- Call `load_and_preprocess()` to get the training and test datasets.
- Build and compile the model using `build_model()`.
- Train the model using `train_model()`.

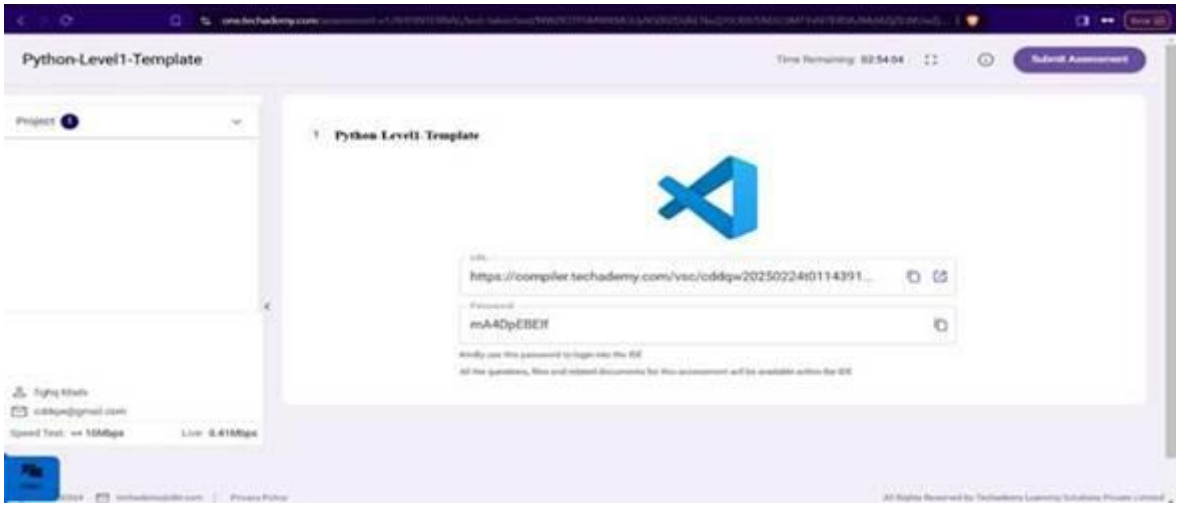
- Evaluate the model with `evaluate_model()`.
- Run prediction on a single test sample using `predict_sample()`.

Execution Steps to Follow:

- All actions like build, compile, running application, running test cases will be through Command Terminal.
- To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal
- This editor Auto Saves the code
- If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page)
- These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
- To launch application: python3 filename.py
- To run Test cases: python3 -m unittest

Screen shot to run the program

- To run the application
- python3 filename.py
- To run the testcase python3 -m unittest



- Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on “Submit Assessment” after you are done with code.