**Creating a Python Project Repository with GitHub and Setting up Local Environment**

**Author**: Hari Prasad

**Date**: 12/09/2023

**Introduction**

This document provides an overview of the steps taken to create a Python project repository on GitHub and set up the project on a local machine.

The project involves generating a basic project structure and a setup.py file for a Python application.

**Step 1: Creating a GitHub Repository**

1.1. Visit the GitHub website (https://github.com) and log in to your account.

1.2. Click on the '+' icon in the top-right corner and select "New Repository" to create a new GitHub repository.

1.3. Fill in the repository name and provide an optional description and other settings as needed.

1.4. Click on the "Create repository" button to create the GitHub repository.

**Step 2: Cloning the Repository Locally**

2.1. Open your terminal or command prompt on your local machine.

2.2. Use the git clone command to clone the newly created GitHub repository to your local machine.

For example:

git clone https://github.com/kalehariprasad/customer-shopping-price

**Step 3: Creating the Python Project Structure**

3.1. In your local project directory, you've created a Python script called template.py. This script is responsible for generating the initial project structure.

3.2. The template.py script prompts you to enter a project name and then generates the following project structure:

project\_name/

├── \_\_init\_\_.py

├── components/

│ └── \_\_init\_\_.py

├── config/

│ └── \_\_init\_\_.py

├── constants/

│ └── \_\_init\_\_.py

├── entity/

│ └── \_\_init\_\_.py

├── exception/

│ └── \_\_init\_\_.py

├── logger/

│ └── \_\_init\_\_.py

├── pipeline/

│ └── \_\_init\_\_.py

├── Utils/

│ └── \_\_init\_\_.py

├── config/config.yaml

├── schema.yaml

├── app.py

├── logs.py

├── exception.py

├── setup.py

├── requirements.txt

└── pipeline.txt

3.3. The script creates directories for various project components and initializes empty \_\_init\_\_.py files to indicate Python packages

**Step 4: Creating the setup.py File**

4.1. The setup.py file is used to define project metadata and dependencies for packaging.

4.2. The script, setup.py, imports required modules and defines a function, get\_requirements\_list(), to read project dependencies from a requirements.txt file.

4.3. It then calls setup() from the setuptools library to configure the project with metadata such as name, version, author, and packages.

4.4. The script reads the project dependencies using the get\_requirements\_list() function and sets them as install requirements.