Setting Up a Virtual Environment for Mankind Matrix

This guide provides step-by-step instructions for creating a separate virtual environment specifically for the Mankind Matrix AI Recommendation Prototype.

What is a Virtual Environment?

A virtual environment is an isolated Python environment that allows you to install packages for a specific project without affecting your system-wide Python installation. This helps avoid dependency conflicts between different projects.

Step-by-Step Instructions

Step 1: Install Python (if not already installed)

Make sure you have Python installed on your system. The prototype should work with Python 3.6 or newer.

- **Windows**: Download from python.org
- **macOS**: `brew install python3` (using Homebrew) or download from python.org
- **Linux**: `sudo apt install python3 python3-pip` (Ubuntu/Debian) or `sudo dnf install python3` (Fedora)

Step 2: Create a Project Directory

```
mkdir mankind_matrix_project
cd mankind_matrix_project
```

Step 3: Create a Virtual Environment

Windows:

```
python -m venv mankind_env
```

macOS/Linux:

```
python3 -m venv mankind_env
```

Step 4: Activate the Virtual Environment

Windows:

```
mankind_env\Scripts\activate
```

macOS/Linux:

```
source mankind_env/bin/activate
```

After activation, your command prompt should change to indicate you're working in the virtual environment, showing something like `(mankind_env)`.

Step 5: Create Project Structure

```
# Create project directories
mkdir -p mankind_matrix_prototype/data
```

Step 6: Install Dependencies

With the virtual environment activated:

```
pip install typing
```

Note: The basic prototype has minimal dependencies, but typing is used for type hints.

Step 7: Verify Installation

Verify that the packages were installed in your virtual environment:

```
pip list
```

You should see `typing` in the list of installed packages.

Step 8: Deactivate the Virtual Environment When Done

When you're finished working with the project:

```
deactivate
```

Troubleshooting

Issue: "Command not found: venv"

- Make sure Python is installed properly
- For some Linux distributions, you may need to install venv separately: `sudo apt install python3-venv`

Issue: Permission Denied (Linux/macOS)

- If you encounter permission issues, try using `sudo` or check your directory permissions

Issue: Virtual Environment Not Activating

- Windows: Make sure you're using the correct path separator (`\`)
- Linux/macOS: Make sure you use `source` before the path
- Check that you're in the correct directory

Advanced: Creating a Requirements File

For future expansion, create a `requirements.txt` file:

```
echo typing > requirements.txt
```

Then install dependencies with:

```
pip install -r requirements.txt
```

Working with the Virtual Environment

Activating the environment each time:

Before working on your project, always activate the virtual environment:

Windows:

```
cd mankind_matrix_project
mankind_env\Scripts\activate
```

macOS/Linux:

```
cd mankind_matrix_project
source mankind_env/bin/activate
```

Adding new packages:

With the environment activated:

```
pip install package_name
```

Saving your environment configuration:

After installing new packages:

```
pip freeze > requirements.txt
```

Recreating the environment elsewhere:

```
python -m venv new_env
source new_env/bin/activate  # or new_env\Scripts\activate on Windows
pip install -r requirements.txt
```