LLM Development Assistant GUI

A simple, clean graphical interface for the LLM Development Assistant that automates iterative development tasks with Large Language Models.

Features

Core Functionality

- **Project Initialization**: Browse and initialize project directories
- **Code Generation**: Request code implementations with task descriptions
- **Code Verification**: Verify if implementations meet requirements
- **Change Implementation**: Apply code changes to files
- **Test Management**: Verify test coverage and run integration tests
- **Development Planning**: Generate next development steps based on goals
- **Workflow Persistence**: Save and load workflow states

GUI Features

- **Clean, Raw Interface**: Simple and functional design focusing on clarity
- **Real-time Logging**: Scrollable log viewer with severity filters (ALL, DEBUG, INFO, WARNING, ERROR)
- **Progress Indicators**: Visual feedback for long-running operations
- **Output Display**: Large text area for viewing operation results
- **Menu System**: File, Edit, Tools, and Help menus for easy navigation
- **Keyboard Shortcuts**: Standard shortcuts for common operations

Running the GUI

Option 1: Direct GUI Launch

```
bash

python -m llm_dev_assistant gui
```

Option 2: Using --gui Flag

```
bash

python -m llm_dev_assistant --gui
```

Option 3: Running the GUI Module Directly

```bash
python -m llm_dev_assistant.gui_main
```

Interface Overview

Main Window Layout

1. **Controls Panel (Left)**
   - Project initialization section
   - Task description text area
   - Optional file path input
   - Action buttons for all operations
   - Workflow management buttons
2. **Output Panel (Top Right)**
   - Displays results from operations
   - JSON-formatted output for clarity
   - Scrollable for large results
3. **Log Panel (Bottom Right)**
   - Real-time log display
   - Severity filter dropdown
   - Clear logs button
   - Color-coded log levels
4. **Status Bar (Bottom)**
   - Current operation status
   - Progress indicator for async operations

Using the GUI

1. Initialize a Project

1. Enter or browse for a project directory path
2. Click "Initialize Project"
3. View the project structure in the output panel

2. Request Code Implementation

1. Enter a task description in the text area
2. Optionally specify a file path to modify

3. Click "Request Code"
4. View the generated code in the output panel

3. Verify Implementation

1. Click "Verify Implementation"
2. In the dialog:
   - Paste original code
   - Paste new code
   - Enter requirements
3. Click "Verify"
4. View verification results in the output panel

4. Implement Changes

1. Click "Implement Changes"
2. In the dialog:
   - Enter or browse for the file path
   - Paste the new code
3. Click "Implement"
4. Check the output for success/failure

5. Test Management

- **Verify Tests**: Enter a file path and click "Verify Tests"
- **Run Tests**: Click "Run Tests" to run all integration tests

6. Planning

1. Click "Plan Next Steps"
2. Enter project goals (one per line)
3. Click "Plan"
4. View the development plan in the output panel

Menu Options

File Menu

- **Initialize Project**: Same as the button
- **Save Workflow**: Save current workflow state to JSON
- **Load Workflow**: Load a previously saved workflow

- **Exit**: Close the application

Edit Menu

- **Clear Output**: Clear the output display
- **Clear Logs**: Clear the log display

Tools Menu

- **Configure LLM**: Set up LLM provider (OpenAI, Local, LM Studio)

Help Menu

- **About**: Display application information

LLM Configuration

The GUI supports three LLM providers:

1. **OpenAI**
   - Select model (gpt-4, gpt-3.5-turbo, gpt-4-vision-preview)
   - Enter API key
2. **Local**
   - Browse for local model file
3. **LM Studio**
   - Enter model name
   - Configure host and port

Log Filtering

Use the filter dropdown to show:

- **ALL**: All log messages
- **DEBUG**: Detailed debugging information
- **INFO**: General information messages
- **WARNING**: Warning messages
- **ERROR**: Error messages only

Tips

1. **Watch the Logs**: The log panel provides valuable information about what's happening
2. **Save Your Work**: Use File > Save Workflow regularly to preserve your progress
3. **Check Status Bar**: The status bar shows current operation status

4. **Use Filters**: Filter logs by severity to focus on important messages
5. **Keyboard Navigation**: Use Tab to move between fields

Troubleshooting

GUI Won't Start

- Ensure all dependencies are installed
- Check that Python tkinter is available: `python -m tkinter`

Operations Fail

- Check the log panel for error messages
- Ensure your OpenAI API key is set (if using OpenAI)
- Verify file paths are correct

Slow Performance

- Long operations run in background threads
- Progress bar indicates ongoing operations
- Check logs for detailed progress information

Requirements

- Python 3.7+
- tkinter (usually comes with Python)
- All dependencies from the main project