# HIT137 Assignment 3 - AI Model Integration GUI Application

## Overview

This project is a comprehensive Tkinter GUI application that demonstrates advanced object-oriented programming concepts while integrating two different Hugging Face AI models. The application provides an intuitive interface for users to interact with **Image Classification** and **Sentiment Analysis** models, showcasing multiple OOP principles in a real-world application.

## Key Features

* **Image Classification**: Uses Vision Transformer (ViT) model for classifying images into 1000+ categories
* **Sentiment Analysis**: Uses RoBERTa-based model for analyzing text sentiment

**Interactive GUI**: User-friendly Tkinter interface with tabbed navigation

**Browse Functionality**: Built-in file browser for selecting image files

**OOP Implementation**: Comprehensive demonstration of inheritance, polymorphism, encapsulation, and more

**Model Information**: Detailed specifications and capabilities of each AI model

* **Educational Content**: Built-in explanations of OOP concepts used in the application

## Quick Start Guide

### Prerequisites

* **Python 3.7 or higher** (Python 3.8+ recommended)
* **Internet connection** (required for downloading models from Hugging Face Hub)

### Installation Instructions

1. **Clone or Download the Project**:

git clone [YOUR\_GITHUB\_REPO\_URL]

cd [REPO\_NAME]

Or download and extract the ZIP file.

1. **Create a Virtual Environment** (Recommended):Not necessary if you feel uncomfortable just ignore this step

python -m venv venv

# On Windows:

venv\Scripts\activate

# On macOS/Linux:

source venv/bin/activate

1. **Install Required Dependencies**:

pip install -r requirements.txt

1. **Verify Installation**:

python -c "import tkinter; import transformers; import torch; print('All dependencies installed successfully!')"

### 🏃‍♂️ Running the Application

**Launch the GUI Application**:

python main.py

The application window will open with three main tabs:

* **AI Models**: Main interaction interface
* **Model Information**: Detailed model specifications
* **OOP Concepts Explanation**: Educational content about implemented OOP principles

## Dependencies

The following packages are required (see requirements.txt):

tkinter # GUI framework (usually included with Python)

transformers # Hugging Face transformers library

torch # PyTorch for model execution

requests # HTTP requests for URL-based images

numpy # Numerical computations

Pillow # Image processing library

**Important Note**: tkinter is typically included with Python installations, but if you encounter issues, you may need to install it separately on some Linux distributions:

# Ubuntu/Debian:

sudo apt-get install python3-tk

# CentOS/RHEL:

sudo yum install tkinter

## 🔧 Model Handling - Important Information

**This project does NOT download or store models locally.**

* **Automatic Downloads**: Models are automatically downloaded from Hugging Face Hub on first use
* **Internet Required**: Initial model loading requires an internet connection
* **Caching**: Hugging Face automatically caches models in your system's cache directory
* **No Manual Downloads**: You don't need to manually download any model files
* **Storage Space**: Models will use approximately 850MB of cache space total
  + Image Classification (ViT): ~350MB
  + Sentiment Analysis (RoBERTa): ~500MB

### Cache Location

Models are cached in:

* **Windows**: C:\Users\[username]\.cache\huggingface\
* **macOS**: ~/.cache/huggingface/
* **Linux**: ~/.cache/huggingface/

## How to Use the Application

### 1. Image Classification

1. Select "Image Classification" from the model dropdown
2. Choose one of the following input methods:
   * **Browse for local images**: Click "Browse Image" button to select files (supports JPG, PNG, GIF, BMP, TIFF, WebP)
   * **Enter image URL**: Paste a direct image URL in the text area
   * **Use sample**: Click "Load Sample Text" for a demo URL
3. Click "Process Text" to classify the image
4. View results with confidence scores and visual progress bars

### 2. Sentiment Analysis

1. Select "Sentiment Analysis" from the model dropdown
2. Enter text in the input area (the Browse button is disabled for text models)
3. Click "Process Text" to analyze sentiment
4. View results showing sentiment classification and confidence scores

### 3. Additional Features

* **Clear Input**: Remove all text from input area
* **Model Information**: View detailed specifications of each model
* **OOP Concepts**: Learn about the programming principles implemented

## Project Structure

AI Model Integration GUI/

├── 📄 main.py # Main application entry point

├── 📄 base\_model.py # Abstract base classes, decorators, and mixins

├── 📄 image\_classification\_model.py # Image classification model implementation

├── 📄 sentiment\_analysis\_model.py # Sentiment analysis model implementation

├── 📄 gui\_application.py # Main GUI application and interface

├── 📄 requirements.txt # Project dependencies

├── 📄 README.md # 📚 This documentation file

├── 📄 INSTALLATION.md # 🔧 Additional installation notes

└── 📄 github\_link.txt # 🔗 Repository link

### File Descriptions

| **File** | **Purpose** | **Key Features** |
| --- | --- | --- |
| main.py | Application entry point | Initializes and launches the GUI |
| base\_model.py | Foundation classes | Abstract base class, decorators, mixins |
| image\_classification\_model.py | Image AI model | ViT model implementation, image processing |
| sentiment\_analysis\_model.py | Text AI model | RoBERTa model implementation, text processing |
| gui\_application.py | User interface | Tkinter GUI, tabs, buttons, file browser |

## AI Models Specifications

### 1. Image Classification Model

* **Model Name**: google/vit-base-patch16-224 (Vision Transformer)
* **Category**: Computer Vision
* **Input**: Images (JPG, PNG, GIF, BMP, TIFF, WebP)
* **Output**: Classification labels with confidence scores
* **Classes**: 1000+ ImageNet categories
* **Size**: ~350MB
* **Use Cases**:
  + Object recognition
  + Content categorization
  + Automated image tagging
  + Visual content analysis

### 2. Sentiment Analysis Model

* **Model Name**: RoBERTa-based sentiment classifier
* **Category**: Natural Language Processing
* **Input**: Text (any length)
* **Output**: Sentiment classification (Positive/Negative/Neutral) with confidence
* **Size**: ~500MB
* **Use Cases**:
  + Social media monitoring
  + Customer feedback analysis
  + Review sentiment classification
  + Content moderation

## Object-Oriented Programming Concepts Implemented

### 1. **Multiple Inheritance**

class ImageClassificationModel(BaseModel, ModelMixin):

# Inherits from both base class and mixin

### 2. **Encapsulation**

* Private attributes with underscore prefix (\_model\_name, \_pipeline)
* Property decorators for controlled access
* Method-based access to internal state

### 3. **Polymorphism**

* Same method names (process(), load\_model()) with different implementations
* Uniform interface across different model types

### 4. **Method Overriding**

* Specialized implementations of base class methods
* Custom get\_model\_info() for each model type

### 5. **Multiple Decorators**

* @log\_method\_call for execution tracking
* @validate\_input for input validation
* Decorator stacking on critical methods

### 6. **Abstraction**

* BaseModel abstract base class
* Interface definition without implementation details

### 7. **Design Patterns**

* **Factory Pattern**: Dynamic model creation
* **Mixin Pattern**: Shared functionality across classes
* **Template Method Pattern**: Algorithm structure in base classes

## 🛠️ Troubleshooting

### Common Issues and Solutions

#### 1. **ModuleNotFoundError: No module named 'tkinter'**

# Ubuntu/Debian:

sudo apt-get install python3-tk

# CentOS/RHEL:

sudo yum install tkinter

#### 2. **Model Download Fails**

* Ensure stable internet connection
* Check firewall settings
* Try running with VPN if in restricted network

#### 3. **GUI Doesn't Appear**

* Verify tkinter installation
* Try running: python -m tkinter (should show a test window)

#### 4. **Slow Model Loading**

* First-time model download can take several minutes
* Subsequent runs will be much faster (models are cached)

#### 5. **Image Classification Fails**

* Ensure image URL is accessible
* For local files, verify file path is correct
* Supported formats: JPG, PNG, GIF, BMP, TIFF, WebP

## Testing the Application

### Image Classification Testing

1. **URL Test**: Use sample URLs provided by "Load Sample Text"
2. **Local File Test**: Use the "Browse Image" button to select local images
3. **Format Test**: Try different image formats (JPG, PNG, GIF, etc.)

### Sentiment Analysis Testing

1. **Positive Text**: "I love this product! It's amazing!"
2. **Negative Text**: "This is terrible and disappointing."
3. **Neutral Text**: "The weather is cloudy today."

## Assignment Requirements Checklist

✅**GitHub Repository**: Public repository with complete source code  
✅ **Multiple Inheritance**: Implemented in model and GUI classes  
✅ **Multiple Decorators**: @log\_method\_call and @validate\_input applied  
✅ **Encapsulation**: Private attributes and controlled access methods  
✅ **Polymorphism**: Same method names with different implementations  
✅ **Method Overriding**: Specialized methods in subclasses  
✅ **Hugging Face Integration**: Two models from different categories  
✅ **GUI Components**: Dropdowns, buttons, text areas, tabs, file browser  
✅ **Input/Output Handling**: Text input, file selection, and result display  
✅ **File Organization**: Modular structure with separate files  
✅ **Documentation**: Comprehensive README and inline code documentation

## Educational Value

This project serves as a comprehensive example of:

* Advanced OOP concepts in Python
* GUI development with tkinter
* AI model integration using Hugging Face
* File handling and user input management
* Error handling and user experience design
* Code organization and documentation practices

## Development Team

## 📄 License

This project is developed for educational purposes as part of HIT137 Assignment 3. All code is provided for learning and academic use.

## 🔗 Additional Resources

* [Hugging Face Transformers Documentation](https://huggingface.co/transformers/)
* [Python tkinter Documentation](https://docs.python.org/3/library/tkinter.html)
* [Object-Oriented Programming in Python](https://docs.python.org/3/tutorial/classes.html)

**Last Updated**: September 2025  
**Version**: 1.0.0  
**Python Compatibility**: 3.7+