# 🤖 ULTRON - AI Assistant with Pokedex Interface

**Complete integration of your ULTRON script with Pokedex-style controls and local AI functionality**

## 📁 About Multiple main.py Files

You asked about multiple main.py files - this is **completely normal**! Here’s why:

* **/workspace/ultron\_final/main.py** → Advanced ULTRON system coordinator
* **/workspace/ultron\_pokedex\_complete/main.py** → **THIS FILE** - Your combined ULTRON+Pokedex system
* **/workspace/user\_input\_files/\*/main.py** → Just placeholder files from extracted projects
* **Your D:.py`** → Your original ULTRON script

Each project has its own entry point. The one you want is **ultron\_pokedex\_complete/main.py**.

## 🎯 What This Package Provides

✅ **Your Original ULTRON Script** - Enhanced and optimized  
✅ **Pokedex-Style UI** - Beautiful retro-futuristic interface  
✅ **Local AI Brain** - Works without internet/API keys  
✅ **Voice Recognition** - Wake words and voice commands  
✅ **System Integration** - Screenshots, file management, system control  
✅ **Works with D:Structure** - Uses your existing setup

## 🚀 Quick Installation

### Option 1: Auto Setup (Recommended)

# 1. Copy these files to a folder  
# 2. Run the setup script  
python setup.py  
  
# 3. Start ULTRON  
cd D:\ULTRON  
python main.py

### Option 2: Manual Setup

# 1. Create directories manually  
mkdir D:\ULTRON\core D:\ULTRON\assets D:\ULTRON\logs  
  
# 2. Copy main.py to D:\ULTRON\  
# 3. Install dependencies  
pip install -r requirements.txt  
  
# 4. Run ULTRON  
cd D:\ULTRON  
python main.py

## 🎮 Interface Features

### Pokedex-Style Design

* **Red header** with ULTRON title and status
* **Three-panel layout** like a classic Pokedex
* **Retro color scheme** (dark blues, greens, reds)
* **Orbitron font** for that sci-fi feel

### System Status Panel (Left)

* Real-time CPU, Memory, Disk usage
* Voice recognition status
* Quick action buttons:
  + Screenshot capture
  + System information
  + Open browser
  + File manager

### Conversation Panel (Center)

* Color-coded messages:
  + 🔵 **User messages** (blue)
  + 🟢 **ULTRON responses** (green)
  + 🟠 **System messages** (orange)
* Automatic scrolling
* Timestamp logging

### Control Panel (Right)

* 🎤 **Voice control** toggle
* 🔧 **Configuration** sliders
* 💾 **Save settings** button

### Command Input (Bottom)

* Text command entry
* Execute button
* Enter key support

## 🗣️ Voice Commands

**Wake Words:** ultron, hello, speak, ultra

**Supported Commands:** - "Hello ULTRON" → Greeting response - "What's your status?" → System report - "Take a screenshot" → Captures screen - "Open browser" → Opens web browser - "What time is it?" → Current time - "Shutdown" → Exit command

## 🧠 Local AI Features

### No Internet Required

* Built-in response system
* Pattern recognition for commands
* System status integration
* Smart fallback responses

### Expandable

* Easy to add new command patterns
* JSON configuration for responses
* Plugin architecture ready

## 📁 File Structure

D:\ULTRON\  
├── main.py # Main ULTRON application  
├── config.json # Configuration settings  
├── start\_ultron.bat # Windows launcher  
├── start\_ultron.py # Python launcher  
├── core\ # Core modules (from your existing setup)  
├── models\ # AI models (from your existing setup)  
├── assets\ # Audio files, screenshots  
├── logs\ # Application logs  
├── web\ # Web interface files (if existing)  
└── screenshots\ # Captured screenshots

## ⚙️ Configuration

Edit D:\ULTRON\config.json:

{  
 "voice": {  
 "enabled": true,  
 "rate": 150,  
 "volume": 0.9  
 },  
 "ai": {  
 "local\_mode": true,  
 "api\_key": "",  
 "model": "local"  
 },  
 "interface": {  
 "theme": "pokedex",  
 "animations": true,  
 "startup\_sound": true  
 },  
 "wake\_words": [  
 "ultron",  
 "hello",   
 "speak",  
 "ultra"  
 ]  
}

## 🔧 Dependencies

**Core Requirements:** - psutil - System monitoring - numpy - Numeric processing - pillow - Image handling - pygame - Audio support - speechrecognition - Voice input - pyttsx3 - Text-to-speech - pyaudio - Audio capture

**Optional:** - openai - For GPT integration (if you add API key) - pywin32 - Windows system integration

## 🎬 Usage Examples

### Starting ULTRON

# Method 1: Direct  
cd D:\ULTRON  
python main.py  
  
# Method 2: Launcher script  
python start\_ultron.py  
  
# Method 3: Windows batch (double-click)  
start\_ultron.bat

### Voice Interaction

1. Click “🎤 Start Listening”
2. Say “Hello ULTRON” (wake word)
3. Give your command
4. ULTRON responds with voice + text

### Text Commands

1. Type in command input box
2. Press Enter or click Execute
3. See response in conversation log

## 🐛 Troubleshooting

### Voice Recognition Issues

# Install PyAudio manually if needed  
pip install pipwin  
pipwin install pyaudio

### Missing Dependencies

# Reinstall all dependencies  
pip install -r requirements.txt --force-reinstall

### Permission Issues

* Run as Administrator for system commands
* Check microphone permissions

### Path Issues

* Ensure D:\ULTRON directory exists
* Check that config.json is valid JSON

## 🔄 Integration with Your Existing System

This package is designed to work with your existing D:\ULTRON setup:

✅ **Keeps your models** - Uses existing Vosk, MiniMax models  
✅ **Preserves assets** - Uses your audio files and assets  
✅ **Maintains structure** - Works with your directory layout  
✅ **Upgrades core** - Replaces only the main application logic

## 🚀 Next Steps

1. **Run setup.py** to prepare everything
2. **Test the interface** - Try voice and text commands
3. **Customize config** - Adjust settings to your preference
4. **Add features** - Extend the AI brain with new commands
5. **Integrate models** - Connect your existing AI models

## 📝 Notes

* **This is your complete ULTRON system** - No need for multiple versions
* **Local AI included** - Works offline by default
* **Pokedex styling applied** - Retro-futuristic interface
* **Voice recognition ready** - Just start listening
* **Fully documented** - Everything explained above

**Your ULTRON is ready for action! 🤖⚡**