National College of Ireland

Configuration Manual -Explainable Chess AI: A Browser Extension for Move Explanation and Analysis in Real-Time Chess Interfaces

MSc Research Project
MSc in Artificial Intelligence

Sai Manohar Nanduri
Student ID: 23404621

School of Computing
National College of Ireland

Supervisor: Mohit Garg

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Sai Manohar Nanduri
Student ID:	23404621
Programme:	MSc in Artificial Intelligence
Year:	2025
Module:	MSc Research Project
Supervisor:	Mohit Garg
Submission Due Date:	11/12/2025
Project Title:	Configuration Manual -Explainable Chess AI: A Browser Extension for Move Explanation and Analysis in Real-Time Chess Interfaces
Word Count:	XXX
Page Count:	9

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Sai Manohar Nanduri
Date:	11th December 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual -Explainable Chess AI: A Browser Extension for Move Explanation and Analysis in Real-Time Chess Interfaces

Sai Manohar Nanduri
23404621

1 System Requirements

The code was built and tested on Windows 11 with Python 3.13.9. The system can also be run on Linux and mac Os environments.

1.1 Hardware Requirements

1. **CPU**: Multi-core processor
2. **GPU**: Not required for inference (optional for model training)
3. **RAM**: Minimum 4GB but recommended higher.
4. **Storage**: Minimum free space of 5GB for:
   (a) Backend dependencies and models
   (b) Stockfish chess engine service
   (c) Ollama local LLM (optional)
   (d) Remaining code files

1.2 Software Requirements

1. **Operating System:**
   (a) Windows 11 (tested)
   (b) Windows 10
   (c) Ubuntu 20.04/22.04 or macOS 11+ (compatible)
2. **Python**: Version 3.8 or higher.
3. **Browser**: Chrome or Edge (tested on chrome)
4. **Stockfish Chess Engine**: Version 15 or higher versions.
5. **Ollama:** Latest version or API ket for any model (optional)

2 Environment Setup

It's recommended to use a virtual environment to isolate project dependencies.

2.1 Using venv (Python Built-in)

Windows (Powershell):

```
# Navigate to project directory
cd E:\NCI\Thesis\explainable-chess-ai

# Create virtual environment
python -m venv venv

# Activate virtual environment
.\venv\Scripts\Activate.ps1

# If you get execution policy error, run:
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```

Figure 1: venv setup

Linux/macOS (bash):

```
# Navigate to project directory
cd /path/to/explainable-chess-ai

# Create virtual environment
python3 -m venv venv

# Activate virtual environment
source venv/bin/activate
```

Figure 2: Linux/macOS

2.2 Installing Python Dependencies

After enabling virtual environment, install all required dependencies:

```
# Install core backend dependencies
pip install -r backend/requirements.txt

# Install LLM dependencies (for AI explanations)
pip install -r requirements-llm.txt
```

Figure 3: Dependencies

Backend Dependencies(backend/requirements.txt):

- FastAPI (0.104.1) - Web framework
- uvicorn (0.24.0) - ASGI server
- python-chess (1.999) - Chess library
- PyTorch ($\geq$ 2.1.0) - Deep learning framework
- scikit-learn ($\geq$ 1.3.0) - ML utilities
- pandas, numpy - Data processing
- transformers ($\geq$ 4.35.0) - NLP models

LLM Dependencies(requirements-llm.txt):

- openai ($\geq$ 1.0.0) - OpenAI API client
- python-chess ($\geq$ 1.999)

3 Stockfish Chess Engine Setup

Stockfish engine is used for position analysis and best move generation.

3.1 Windows Installation

There are 2 options to install stockfish one from official website and other way is using Chocolatey

Option 1: Download from official website

- Visit https://stockfishchess.org/download/
- Download **Stockfish 16 for Windows** (x86-64-avx2 recommended)
- Extract the ZIP file
- Place `stockfish-windows-x86-64-avx2.exe` in the `stockfish/` directory

Option 2: Use Chocolatey

```
choco install stockfish
```

Figure 4: Stockfish installation using chocolatey

3.2 macOS Installation

Option 1:Homebrew

```
brew install stockfish
```

Figure 5: Stockfish installation using Homebrew

Option 2: Download manually

- Visit https://stockfishchess.org/download/
- Download **Stockfish 16 for macOS**
- Extract and place in `stockfish/` directory

3.3 Linux Installation

Ubuntu/Debian:

```
sudo apt-get update
sudo apt-get install stockfish
```

Figure 6:

Fedora:

```
sudo dnf install stockfish
```

Figure 7:

From source:

```
# Download from https://stockfishchess.org/download/
cd stockfish/src
make build ARCH=x86-64-modern
```

Figure 8:

3.4 Stockfish Detection

The server automatically detects stockfish in stockfish/ directory in root project.

4 LLM Setup for AI Explanations

The project supports multiple LLM providers with automatic fallback feature:

- OpenAI GPT-4 (cloud, paid, best quality)
- Ollama (Local, free, good quality)
- Rule-based feedback (No LLM required)

4.1 Option 1: Ollama Setup (Recommended for Free Usage)

4.1.1 Windows Installation

1. Download Ollama from https://ollama.com/download
2. Run `OllamaSetup.exe` installer
3. After installation, open a new terminal/PowerShell
4. Pull the llama3.2 model:

```
ollama pull llama3.2
```

5. Verify Installation:

```
ollama list
ollama run llama3.2
```

4.1.2 macOS/Linux Installation

```
# Install Ollama
curl -fsSL https://ollama.com/install.sh | sh

# Pull the model
ollama pull llama3.2

# Verify installation
ollama list
ollama run llama3.2
```

4.1.3 Ollama Service

Ollama runs as background service to manually stop the service command is **Stop-Service Ollama**

4.2 Option 2: Open AI API Setup

if you prefer cloud-based LLM with better quality and speed.

1. create an account at https://platform.openai.com
2. Generate an API key
3. Set the environment variable:

Windows (PowerShell)

```
# Temporary (current session)
$env:OPENAI_API_KEY = "sk-your-api-key-here"

# Permanent (add to user environment)
[System.Environment]::SetEnvironmentVariable('OPENAI_API_KEY', 'sk-your-api-key-here', 'User')
```

Linux/macOS(bash)

```
# Temporary (current session)
export OPENAI_API_KEY="sk-your-api-key-here"

# Permanent (add to ~/.bashrc or ~/.zshrc)
echo 'export OPENAI_API_KEY="sk-your-api-key-here"' >> ~/.bashrc
source ~/.bashrc
```

Alternatively, create/edit .env file

:

```
OPENAI_API_KEY=sk-your-api-key-here
```

5 Backend Server Configuration

5.1 Environment Variables

create or edit .env file in the project root:

```
# API Configuration
API_HOST=localhost
API_PORT=8001
DEBUG=True

# Chess Engine Configuration
STOCKFISH_PATH=stockfish/stockfish-windows-x86-64-avx2.exe
STOCKFISH_DEPTH=20
STOCKFISH_THREADS=2

# ML Configuration
MODEL_PATH=backend/models/saved_models
CACHE_SIZE=1000

# Data Configuration
DATA_PATH=backend/data
PGN_PATH=backend/data/raw
PROCESSED_PATH=backend/data/processed

# OpenAI Configuration (Optional)
OPENAI_API_KEY=sk-your-api-key-here

# Security
SECRET_KEY=your-secret-key-change-this-in-production
ALGORITHM=HS256
```

5.2 Stockfish Configuration

Edit the parameters in simple_server.py file or environment variables:

```
# Stockfish analysis depth (higher = better but slower)
STOCKFISH_DEPTH = 20  # Range: 10-25

# CPU threads for Stockfish
STOCKFISH_THREADS = 2  # Range: 1-8
```

5.3 LLM Configuration

To switch LLm providers , edit line 93 of simple_server.py file.

Force Ollama:

```
llm_explanation_service = LLMExplanationService(provider="ollama-local")
```

Force OpenAI:

```
llm_explanation_service = LLMExplanationService(
    provider="openai-gpt4",
    api_key=os.getenv("OPENAI_API_KEY")
)
```

Force fallback (no LLM)

```
llm_explanation_service = LLMExplanationService(provider="fallback")
```

6 Browser Extension Installation

6.1 Chrome/Edge Installation

1. Open the browser
2. Navigate to extensions page:
3. Enable "**Developer mode**" present in the right-corner
4. click "**Load unpacked**" button.
5. Select the **frontend/** directory from the project.
6. The extension should appear with the chess icon

6.2 Extension Configuration

The extension connects to the backend API. Configuration is in **frontend/content-scripts/content-scripts.js**

```
// API endpoint (default)
const API_URL = 'http://localhost:8001';

// Analysis refresh rate (milliseconds)
const REFRESH_INTERVAL = 2000;
```

6.3 Manifest Configuration

The extension manifest is built in **frontend/manifest.json**

```
{
  "manifest_version": 3,
  "name": "Explainable Chess AI",
  "version": "1.0.0",
  "description": "Real-time chess analysis with AI explanations",
  "permissions": [
    "activeTab",
    "storage"
  ],
  "host_permissions": [
    "https://www.chess.com/*",
    "https://lichess.org/*",
    "http://localhost:8001/*"
  ]
}
```

7 Starting the System

7.1 Start Backend Server

run the **simple_server.py** file in the directory.

Expected Output

```
[SUCCESS] Stockfish engine initialized successfully!
[SUCCESS] Tactical classifier initialized with 20 themes!
[SUCCESS] Board analyzer initialized successfully!
[SUCCESS] Move comparator initialized successfully!
[INFO] OpenAI API key found, testing GPT-4o ....
[SUCCESS] OpenAI initialized, ready to use!
INFO:     Started server process [12345]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8001
```

If no OpenAI key:

```
[INFO] Trying Ollama local LLM...
[SUCCESS] Ollama local LLM initialized successfully!
```

7.2 Verify Server Health

Open browser and visit **http://localhost:8001/health**

Expected Response:

```
{
  "status": "healthy",
  "message": "Test server is working"
}
```

References