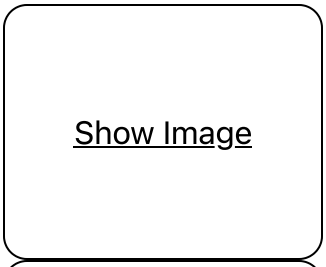
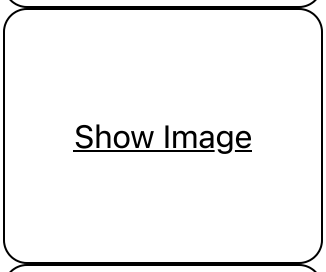
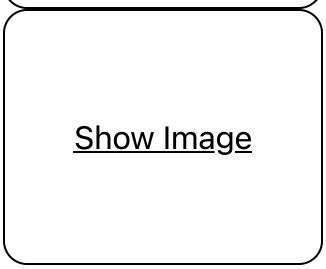
STL-PINN Processor: Production-Grade 3D Mesh Processing with Physics-Informed Neural Networks







A comprehensive, production-ready system for intelligent 3D mesh processing, repair, and physics analysis using Physics-Informed Neural Networks (PINNs) with LLM-guided optimization.

🚀 Features

Core Capabilities

- **Intelligent STL Processing**: LLM-guided mesh repair and optimization
- **PINN Integration**: Physics-informed neural networks for structural analysis
- **Advanced Mesh Repair**: FreeCAD integration for complex geometric operations
- **Quality Validation**: AI-powered mesh quality assessment and validation
- **Multiple Output Formats**: STL, STEP, PLY, OBJ support
- **Real-time Processing**: Async processing with progress tracking
- **Production API**: RESTful API with comprehensive error handling

AI-Powered Features

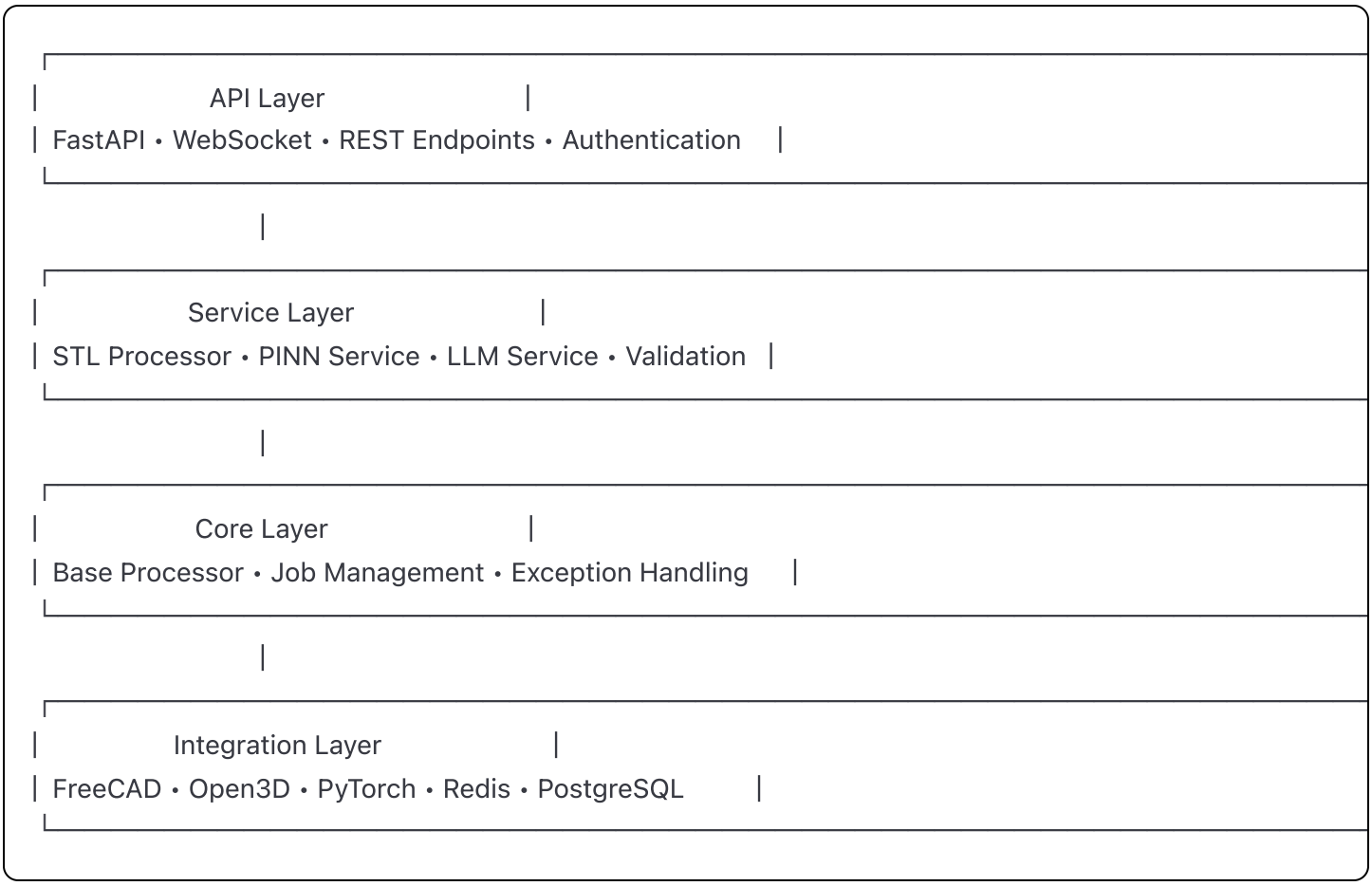
- **LLM Analysis**: GPT-4, Claude-3, or local Llama models for intelligent recommendations
- **Adaptive Processing**: Dynamic operation selection based on mesh characteristics

- **Code Generation**: Automatic PINN code generation for custom physics problems
- **Quality Assessment**: Multi-metric mesh quality evaluation

Physics Analysis

- **Structural Mechanics**: Stress, strain, and displacement analysis
- **Material Properties**: Support for various engineering materials
- **Boundary Conditions**: Flexible boundary condition specification
- **Safety Analysis**: Automatic safety factor computation
- **Visualization**: Interactive 3D result visualization

🏗️ Architecture



📦 Installation

Prerequisites

- Python 3.11+
- Docker & Docker Compose
- NVIDIA GPU (recommended for PINN training)

- FreeCAD (for advanced mesh operations)

Quick Start

```bash
# Clone the repository
git clone https://github.com/your-org/stl-pinn-processor.git
cd stl-pinn-processor

# Copy environment configuration
cp .env.example .env

# Install dependencies
pip install -r requirements.txt

# Start services
docker-compose up -d

# Run the application
python main.py --api
```

Development Setup

```bash
# Create virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv|Scripts|activate

# Install development dependencies
pip install -r requirements.txt
pip install -e .

# Run tests
pytest

# Start in development mode
python main.py --api
```

Docker Deployment

```bash
```

```
# Production deployment
docker-compose -f docker-compose.prod.yml up -d

# Scale services
docker
```