# Hệ Thống Triển Khai Mô Hình ML Tiết Kiệm Năng Lượng cho Thiết Bị IoT

Dự án đồ án chuyên ngành: Hệ thống quản lý và triển khai mô hình Machine Learning lên thiết bị IoT với giám sát **telemetry thực** theo thời gian thực.

Lưu ý quan trọng: nếu không có phần cứng/nguồn đo công suất (power sensor / INA219 / powercap…) hoặc pipeline telemetry năng lượng từ thiết bị, hệ thống **không thể** hiển thị "mWh tiêu thụ" một cách chính xác. Trong trường hợp đó, dashboard sẽ hiển thị các metrics **thực** sẵn có từ hệ điều hành như CPU/RAM/Temperature/Storage.

## Tổng Quan

Hệ thống bao gồm 2 thành phần chính:

1. **ML Controller (Server quản lý)**: Flask web server chạy trên máy tính chủ, cung cấp dashboard web để quản lý models, dự đoán năng lượng, và triển khai models lên thiết bị IoT.
2. **BBB ML Agent (Agent trên thiết bị)**: Service chạy trên thiết bị IoT, nhận models từ controller, thực thi inference, và báo cáo telemetry **thực** (CPU/RAM/Temperature/Storage). Nếu có tích hợp nguồn đo năng lượng bên ngoài, agent có thể nhận telemetry năng lượng qua API.

## Tính Năng Chính

- **Energy Prediction**: Dự đoán năng lượng tiêu thụ của mô hình ML trước khi triển khai
  - Sử dụng Gradient Boosting Regressor trained trên 274 models thực tế
  - Jetson Nano: MAPE 18.69%, $R^2$=0.86 (247 models)
  - Raspberry Pi 5: MAPE 15.88%, $R^2$=0.95 (27 models)
  - 15+ popular models sẵn có (MobileNet, ResNet, EfficientNet, etc.)
  - EXCELLENT models đã được tải sẵn trong `model_store` để deploy ngay
  - 📖 **User Guide**: ENERGY_PREDICTION_USER_GUIDE.md
  - 🔧 **Technical Guide**: ENERGY_PREDICTION_TECHNICAL_GUIDE.md
- Đề xuất top models phù hợp nhất dựa trên energy budget, kích thước, và latency
- Triển khai models lên thiết bị IoT qua HTTP API
- Tự động download model artifacts từ timm (PyTorch Image Models) khi cần
- Giám sát telemetry thiết bị real-time (CPU/RAM/Temperature/Storage) với biểu đồ và metrics
- Tích hợp Balena Cloud để quản lý fleet thiết bị IoT
- Dashboard web đầy đủ với 2 chế độ: Deployment và Monitoring

## Yêu Cầu Hệ Thống

### Máy chủ (ML Controller)

- Python 3.8 trở lên
- Flask 3.0.0
- scikit-learn, pandas, numpy
- PyTorch và timm (để download models)

- 2GB RAM tối thiểu
- Hệ điều hành: Windows/Linux/MacOS

### Thiết bị IoT (Target Device)

- Jetson Nano Dev Kit 2GB (hoặc tương tự)
- Python 3.x
- Flask framework
- Kết nối mạng LAN/WiFi với máy chủ
- 100MB storage trống (để lưu model artifacts)

### (Tuỳ chọn) Nguồn đo năng lượng

- Nếu cần số liệu năng lượng (mWh/mW) **thực**, phải có phần cứng/nguồn đo hoặc cơ chế telemetry (ví dụ INA219/INA226, powercap trên nền tảng hỗ trợ, hoặc một service đọc cảm biến và POST về agent).

## Cấu Trúc Thư Mục

```
DACN/
├── ml-controller/                    # Server quản lý chính
│   ├── python/                       # Source code Python
│   │   ├── app.py                    # Flask server chính (API endpoints)
│   │   ├── model_analyzer.py         # Phân tích benchmark và đề xuất models
│   │   ├── energy_predictor_service.py # Service dự đoán năng lượng
│   │   └── download_models.py        # Tool download model artifacts
│   │
│   ├── data/                         # Dữ liệu benchmark
│   │   └── 124_models_benchmark_jetson.csv   # Dataset 124 models benchmark trên
Jetson Nano
│   │   └── 27_models_benchmark_rpi5.csv      # Dataset 27 models benchmark trên
Raspberry Pi 5
│   │
│   ├── artifacts/                    # Model ML đã train
│   │   ├── energy_predictor.pkl      # Gradient Boosting model
│   │   ├── energy_scaler.pkl         # StandardScaler cho features
│   │   ├── feature_names.pkl         # Danh sách features
│   │   └── model_info.pkl            # Metadata của model
│   │
│   ├── model_store/                  # Lưu trữ model artifacts (.pth, .onnx)
│   │   ├── mobilenetv3_small_075.pth
│   │   ├── ghostnet_100.pth
│   │   └── ...                       # (14 models đã download)
│   │
│   ├── notebooks/                    # Jupyter notebooks
│   │   └── energy_prediction_model.ipynb  # Training notebook
│   │
│   ├── templates/                    # Giao diện web
│   │   └── index.html                # IoT ML Energy Manager dashboard
│   │
│   └── requirements.txt              # Dependencies Python
│
```

```
├── bbb-ml-agent/                    # Agent chạy trên thiết bị IoT
│   └── app/
│       ├── server.py                # Flask server trên IoT device
│       ├── requirements.txt         # Dependencies cho agent
│       └── Dockerfile.template      # Template Dockerfile (Balena)
│
├── README.md                        # File này
└── USER_GUIDE.md                    # Hướng dẫn sử dụng chi tiết
```

# Cài Đặt

## Bước 1: Clone Repository

```
git clone <repository-url>
cd DACN
```

## Bước 2: Cài Đặt Dependencies cho ML Controller

```
cd ml-controller

# Tạo virtual environment (khuyến nghị)
python -m venv venv

# Kích hoạt virtual environment
# Windows PowerShell:
.\venv\Scripts\Activate.ps1
# Windows CMD:
venv\Scripts\activate.bat
# Linux/MacOS:
source venv/bin/activate

# Cài đặt packages
pip install flask==3.0.0 requests==2.32.0
pip install scikit-learn pandas numpy
pip install torch timm  # Để download models
```

## Bước 3: Cài Đặt Agent trên Thiết Bị IoT

**Cách 1: Chạy trực tiếp (Development)**

```
# SSH vào thiết bị IoT
ssh user@<device-ip>

# Copy folder bbb-ml-agent lên thiết bị
# Sau đó:
```

```
cd bbb-ml-agent/app
pip install -r requirements.txt
python server.py
```

**Cách 2: Deploy qua Balena Cloud (Production)**

1. Tạo application trên Balena Cloud
2. Push code lên Balena:

```
cd bbb-ml-agent
balena push <app-name>
```

3. Thiết bị sẽ tự động download và chạy container

### Bước 4: Kiểm Tra Cài Đặt

Verify các file quan trọng tồn tại:

```
# Kiểm tra artifacts (model đã train)
ls ml-controller/artifacts/
# Phải có: energy_predictor.pkl, energy_scaler.pkl, feature_names.pkl,
model_info.pkl

# Kiểm tra dataset benchmark
ls ml-controller/data/
# Phải có: 124_models_benchmark_jetson.csv, 27_models_benchmark_rpi5.csv

# Kiểm tra model store (có thể trống ban đầu, sẽ download khi cần)
ls ml-controller/model_store/
```

## Sử Dụng

### 1. Khởi Động ML Controller

```
cd ml-controller/python
python app.py
```

Server sẽ chạy tại: **http://localhost:5000**

Output mẫu:

```
 * Running on http://0.0.0.0:5000
Press CTRL+C to quit
```

## 2. Mở Dashboard Web

Truy cập: **http://localhost:5000** trên trình duyệt

Dashboard **IoT ML Energy Manager** sẽ hiển thị với 2 tab:

- **Deployment**: Quản lý và deploy models
- **Monitoring**: Giám sát telemetry thiết bị real-time (CPU/RAM/Temperature/Storage)

## 3. Workflow Triển Khai Model

#### Bước 3.1: Tải Danh Sách Models

Trên dashboard, nhấn nút **"Đề xuất năng lượng thấp"** để tải top 10 models được đề xuất (< 100 mWh, < 100 MB, latency < 0.5s).

Hoặc nhấn **"Tải tất cả"** để xem toàn bộ 126 models trong benchmark.

#### Bước 3.2: Chọn Model

- Cuộn danh sách "Model library" ở cột bên phải
- Click vào model bạn muốn deploy (ví dụ: `mobilenetv3_small_075`)
- Xem thông tin chi tiết trong "Selection preview":
  - Năng lượng tiêu thụ: 58.32 mWh
  - Kích thước: 8.21 MB
  - Latency: 0.026s
  - Parameters: 2.04M

#### Bước 3.3: Cấu Hình Thiết Bị

- Nhập IP thiết bị IoT vào ô "Địa chỉ IP thiết bị" (mặc định: 192.168.137.10)
- Hoặc chọn thiết bị từ panel "Balena fleet" và nhấn "Connect"

#### Bước 3.4: Deploy Model

- Nhấn nút **"Deploy model đã chọn"** (màu xanh)
- Hệ thống sẽ:
  1. Kiểm tra năng lượng dự đoán vs ngân sách (100 mWh)
  2. Download model từ timm nếu chưa có (tự động)
  3. Transfer model lên thiết bị IoT
  4. Thiết bị tải model về và khởi động inference

#### Bước 3.5: Giám Sát

- Xem "Energy watch" để theo dõi năng lượng real-time
- Xem "Deployment log" để kiểm tra trạng thái deploy
- Chuyển sang tab "Monitoring" để xem biểu đồ chi tiết

## 4. Download Model Artifacts Thủ Công

Nếu muốn download model trước khi deploy:

```
cd ml-controller/python
python download_models.py <model_name>

# Ví dụ:
python download_models.py mobilenetv3_small_075
python download_models.py ghostnet_100
python download_models.py efficientnet_b0
```

Model sẽ được lưu vào `ml-controller/model_store/`.

## 5. Dự Đoán Năng Lượng cho Model Mới

**Cách 1: Sử dụng Dashboard**

1. Mở panel "Energy predictor" ở cột phải
2. Nhập các thông số model:
   - Params (M): Số lượng parameters (triệu)
   - GFLOPs: Floating-point operations (tỷ)
   - GMACs: Multiply-accumulate operations (tỷ)
   - Size (MB): Kích thước file
   - Latency (s): Thời gian inference
   - Throughput (iter/s): Số iterations/giây
3. Nhấn "Dự đoán"
4. Kết quả hiển thị: Energy Est + Confidence Interval

**Cách 2: Sử dụng API**

```
curl -X POST http://localhost:5000/api/predict-energy \
  -H "Content-Type: application/json" \
  -d '{
    "params_m": 5.0,
    "gflops": 1.5,
    "gmacs": 0.75,
    "size_mb": 20.0,
    "latency_avg_s": 0.05,
    "throughput_iter_per_s": 20.0
  }'
```

Response:

```
{
  "prediction_mwh": 58.32,
  "confidence_interval": [40.82, 75.82],
  "model_r2": 0.943
}
```

# API Reference

## 1. GET /api/models/all

Lấy danh sách tất cả 126 models từ benchmark.

**Response:**

```
{
  "models": [
    {
      "model": "mobilenetv3_small_075",
      "params_m": 2.04,
      "gflops": 0.04,
      "size_mb": 8.21,
      "latency_avg_s": 0.026,
      "energy_mwh": 58.32
    }
  ]
}
```

## 2. GET /api/models/recommended

Lấy danh sách models được đề xuất dựa trên constraints.

**Query Parameters:**

- `device_type` (optional): Loại thiết bị (mặc định: "BBB")
- `max_energy` (optional): Ngân sách năng lượng tối đa (mặc định: 100 mWh)
- `max_size` (optional): Kích thước tối đa (mặc định: 100 MB)
- `max_latency` (optional): Latency tối đa (mặc định: 0.5s)

**Example:**

```
curl "http://localhost:5000/api/models/recommended?max_energy=100&max_size=50"
```

## 3. POST /api/predict-energy

Dự đoán năng lượng tiêu thụ cho model mới.

**Request Body:**

```
{
  "params_m": 5.0,
  "gflops": 1.5,
  "gmacs": 0.75,
```

```
  "size_mb": 20.0,
  "latency_avg_s": 0.05,
  "throughput_iter_per_s": 20.0
}
```

**Response:**

```
{
  "prediction_mwh": 85.43,
  "confidence_interval": [65.2, 105.66],
  "features_used": ["params_m", "gflops", "gmacs", "size_mb", "latency_avg_s",
"throughput_iter_per_s", "params_per_gflop", "efficiency_score"]
}
```

### 4. POST /api/deploy

Deploy model lên thiết bị IoT.

**Request Body:**

```
{
  "bbb_ip": "192.168.137.10",
  "model_name": "mobilenetv3_small_075",
  "max_energy": 100
}
```

**Response (Success):**

```
{
  "status": "success",
  "message": "Deploy thành công",
  "model": "mobilenetv3_small_075",
  "predicted_energy": 58.32,
  "device_ip": "192.168.137.10"
}
```

### 5. GET /api/balena/devices

Lấy danh sách thiết bị từ Balena Cloud.

**Query Parameters:**

- `app_slug` (optional): Lọc theo application
- `online_only` (optional): true/false (chỉ hiển thị thiết bị online)

**Response:**

```
{
  "devices": [
    {
      "device_name": "proud-star",
      "uuid": "abc123...",
      "is_online": true,
      "ip_address": "192.168.1.100",
      "os_version": "balenaOS 2.88.5",
      "device_type": "jetson-nano"
    }
  ]
}
```

### 6. GET /models/

Download model artifact từ model_store.

**Example:**

```
curl -O http://localhost:5000/models/mobilenetv3_small_075.pth
```

## Model Predictor

### Thông Tin Model

- **Loại model**: Gradient Boosting Regressor
- **Độ chính xác**: $R^2$ = 0.943 trên test set
- **MAE**: 70.39 mWh
- **MAPE**: 29.28%
- **Dataset**: 126 models benchmark trên Jetson Nano Dev Kit 2GB

### Features Sử Dụng

**Primary Features** (6 features):

1. `params_m`: Số lượng parameters (triệu)
2. `gflops`: Floating-point operations (tỷ)
3. `gmacs`: Multiply-accumulate operations (tỷ)
4. `size_mb`: Kích thước file model (MB)
5. `latency_avg_s`: Thời gian inference trung bình (giây)
6. `throughput_iter_per_s`: Số iterations mỗi giây

**Derived Features** (2 features - tự động tính): 7. `params_per_gflop`: Tỷ lệ params/gflops 8. `efficiency_score`: latency × size_mb

### Training Process

Xem chi tiết trong notebook: `ml-controller/notebooks/energy_prediction_model.ipynb`

6 thuật toán được so sánh:

- Linear Regression
- Ridge Regression
- Random Forest
- Gradient Boosting (BEST)
- XGBoost
- Neural Network

## Dataset Benchmark

File: `ml-controller/data/124_models_benchmark_jetson.csv`

**Columns:**

- `model`: Tên model (timm architecture name)
- `params_m`: Parameters (triệu)
- `gflops`: FLOPs (tỷ)
- `gmacs`: MACs (tỷ) - có thể null
- `size_mb`: Kích thước file (.pth)
- `latency_avg_s`: Latency trung bình
- `latency_std_s`: Latency std dev
- `throughput_iter_per_s`: Throughput
- `energy_mwh`: Năng lượng tiêu thụ (mWh) - TARGET

**Thống kê:**

- Tổng số models: 126
- Models < 50 mWh: 18 models (cực kỳ tiết kiệm)
- Models < 100 mWh: 45 models (phù hợp IoT)
- Models < 200 mWh: 78 models
- Range năng lượng: 35.5 - 850 mWh

**Top 5 models tiết kiệm nhất:**

1. ghostnet_100: 35.5 mWh, 5.18 MB, 2.59M params
2. mnasnet_small: 37.2 mWh, 8.39 MB, 2.03M params
3. mobilenetv3_small_075: 58.3 mWh, 8.21 MB, 2.04M params
4. mobilenetv3_small_100: 64.7 mWh, 9.46 MB, 2.54M params
5. mobilenetv2_050: 67.8 mWh, 7.72 MB, 1.97M params

## Model Store

Thư mục: `ml-controller/model_store/`

Lưu trữ model artifacts (.pth, .onnx, .tflite, .bin) để deploy lên thiết bị.

**Models hiện có** (14 models đã download):

- ghostnet_100.pth
- mnasnet_small.pth
- mobilenetv3_small_075.pth
- mobilenetv3_small_100.pth
- mobilenetv2_050.pth
- efficientnet_b0.pth
- efficientnet_lite0.pth
- resnet18.pth
- squeezenet1_0.pth
- shufflenet_v2_x0_5.pth
- (và thêm 4 models khác)

**Cách thêm models mới:**

```
# Cách 1: Download tự động từ timm
cd ml-controller/python
python download_models.py <model_name>

# Cách 2: Copy thủ công
cp /path/to/your/model.pth ml-controller/model_store/

# Lưu ý: Tên file phải khớp với tên model trong CSV
# Ví dụ: model "mobilenetv3_small_075" → file "mobilenetv3_small_075.pth"
```

**Định dạng hỗ trợ:**

- .pth, .pt (PyTorch)
- .onnx (ONNX)
- .tflite (TensorFlow Lite)
- .bin (Generic binary)

# Balena Cloud Integration

Hệ thống tích hợp Balena Cloud để quản lý fleet thiết bị IoT.

## Cấu Hình Balena Token

**Cách 1: Hardcode trong code (Development)**

File: `ml-controller/python/app.py`

```
BALENA_API_TOKEN = "your_token_here"
```

**Cách 2: Environment Variable (Production)**

```
export BALENA_API_TOKEN="your_token_here"
python app.py
```

### Lấy Balena API Token

1. Đăng nhập Balena Cloud: https://dashboard.balena-cloud.com
2. Vào **Account Settings** → **Access Tokens**
3. Tạo token mới với scope: "Read-Write"
4. Copy token và cấu hình

### Sử Dụng Balena Fleet

1. Mở dashboard tại http://localhost:5000
2. Scroll xuống panel "Balena fleet"
3. (Optional) Nhập app slug để lọc thiết bị
4. (Optional) Check "Chỉ hiển thị online"
5. Nhấn "Làm mới" để tải danh sách thiết bị
6. Click "Connect" bên cạnh thiết bị để dùng IP của thiết bị đó
7. Deploy model như bình thường

## Troubleshooting

### Lỗi 1: ModuleNotFoundError

**Triệu chứng:**

```
ModuleNotFoundError: No module named 'flask'
```

**Giải pháp:**

```
pip install flask requests scikit-learn pandas numpy
```

### Lỗi 2: FileNotFoundError - artifacts not found

**Triệu chứng:**

```
FileNotFoundError: ml-controller/artifacts/energy_predictor.pkl
```

**Nguyên nhân:** Các file artifacts chưa được tạo.

**Giải pháp:**

1. Chạy notebook training: `ml-controller/notebooks/energy_prediction_model.ipynb`

2. Hoặc copy artifacts từ backup (nếu có)

### Lỗi 3: Model artifact not found khi deploy

**Triệu chứng:**

```
Error: Model file mobilenetv3_small_075.pth not found in model_store
```

**Giải pháp:**

```
# Download model tự động
cd ml-controller/python
python download_models.py mobilenetv3_small_075

# Hoặc hệ thống sẽ tự động download khi bạn nhấn deploy
```

### Lỗi 4: Cannot connect to device

**Triệu chứng:**

```
Error: Không thể kết nối BBB tại 192.168.137.10
```

**Giải pháp:**

1. Kiểm tra thiết bị đã bật và chạy agent:

```
ssh user@192.168.137.10
cd bbb-ml-agent/app
python server.py
```

2. Kiểm tra kết nối mạng:

```
ping 192.168.137.10
```

3. Kiểm tra firewall cho phép port 5000

### Lỗi 5: BALENA_API_TOKEN not configured

**Triệu chứng:**

```
Warning: BALENA_API_TOKEN chưa được cấu hình
```

**Giải pháp:**

Chức năng Balena Fleet sẽ không hoạt động nhưng các chức năng khác vẫn bình thường.

Để fix: Cấu hình token theo hướng dẫn ở phần "Balena Cloud Integration".

### Lỗi 6: Port 5000 already in use

**Triệu chứng:**

```
OSError: [Errno 98] Address already in use
```

**Giải pháp:**

```
# Tìm process đang dùng port 5000
# Windows:
netstat -ano | findstr :5000
taskkill /PID <PID> /F

# Linux/MacOS:
lsof -i :5000
kill -9 <PID>

# Hoặc đổi port trong app.py:
app.run(host='0.0.0.0', port=5001)
```

### Lỗi 7: CSV file not found

**Triệu chứng:**

```
FileNotFoundError: 124_models_benchmark_jetson.csv
```

**Giải pháp:**

Đảm bảo file CSV tồn tại tại: `ml-controller/data/124_models_benchmark_jetson.csv`

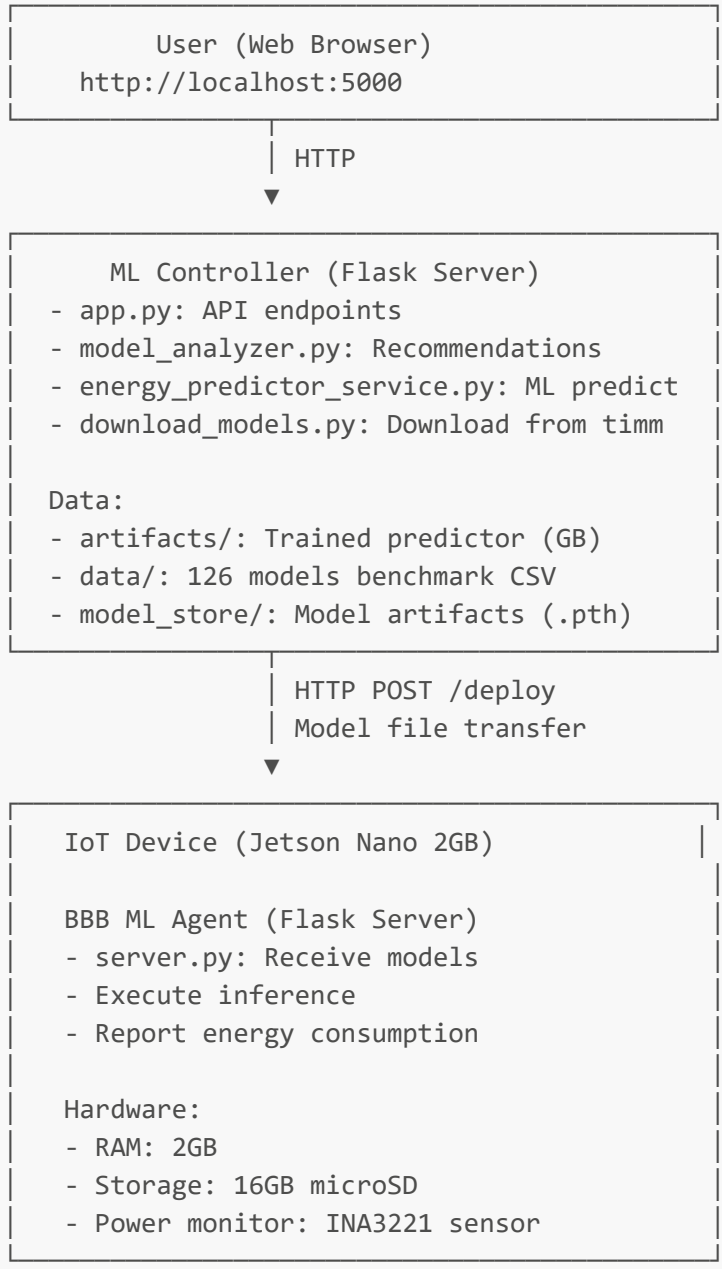
Nếu thiếu, copy từ backup hoặc tạo lại bằng notebook benchmark.

## Tài Liệu Bổ Sung

- **USER_GUIDE.md**: Hướng dẫn sử dụng dashboard web chi tiết từng bước
- **Notebook Training**: `ml-controller/notebooks/energy_prediction_model.ipynb` - Chi tiết quá trình train model predictor
- **Dataset**: `ml-controller/data/124_models_benchmark_jetson.csv` - Dữ liệu benchmark Jetson Nano

- **Dataset**: `ml-controller/data/27_models_benchmark_rpi5.csv` - Dữ liệu benchmark Raspberry Pi 5

## Kiến Trúc Hệ Thống




## Performance Benchmarks

### Model Predictor Performance

- **Training Time**: ~5 phút (126 samples, 6 models comparison)
- **Inference Time**: < 1ms per prediction
- **Accuracy**: $R^2$ = 0.943, MAE = 70.39 mWh
- **Best Model**: Gradient Boosting Regressor

### System Performance

- **Model List Load**: < 500ms (126 models)
- **Recommendation Load**: < 200ms (top 10)
- **Predict API**: < 50ms per request
- **Deploy Time**: 5-30 giây (tùy kích thước model và network)
- **Model Download**: 2-10 giây (tùy kích thước, 5-50 MB)

### Device Requirements

**Jetson Nano Dev Kit 2GB:**

- RAM Usage: ~500MB (idle) + model size
- CPU Usage: ~20% (monitoring mode)
- Network: 100Mbps LAN khuyến nghị
- Power: 5V/2A adapter

## Roadmap

### Version 1.0 (Current)

- Energy prediction với Gradient Boosting
- Web dashboard deployment
- Balena Fleet integration
- 126 models benchmark

### Future Enhancements

- ☐ Support thêm device types (Raspberry Pi, Coral, etc.)
- ☐ Real-time energy monitoring với chart streaming
- ☐ Auto-scaling models dựa trên battery level
- ☐ Model versioning và rollback
- ☐ A/B testing framework
- ☐ Mobile app để giám sát
- ☐ Cảnh báo qua email/Slack khi vượt ngưỡng
- ☐ Export báo cáo PDF/Excel

## Đóng Góp

Nếu bạn muốn đóng góp vào dự án:

1. Fork repository
2. Tạo branch mới: `git checkout -b feature/your-feature`
3. Commit changes: `git commit -m 'Add some feature'`
4. Push to branch: `git push origin feature/your-feature`
5. Tạo Pull Request

## License

Dự án đồ án chuyên ngành - Educational purpose.

## Liên Hệ

Nếu có câu hỏi hoặc gặp vấn đề, vui lòng:

1. Xem file USER_GUIDE.md để biết cách sử dụng chi tiết
2. Check phần Troubleshooting ở trên
3. Xem logs trong "Deployment log" panel trên dashboard
4. Xem terminal console nơi chạy `python app.py`

---

**Chúc bạn triển khai thành công!**