NXT Robot - Dual-Purpose Educational & CyberSec Assistant

> **"The robot that teaches colors by day, runs pentests by night"**

🎯Project Overview

Building a Raspberry Pi-based robot with two distinct operational modes:

- **KID MODE**: Educational assistant for a 5-year-old (games, learning, stories)
- **CYBER MODE**: Penetration testing assistant running Kali Linux commands

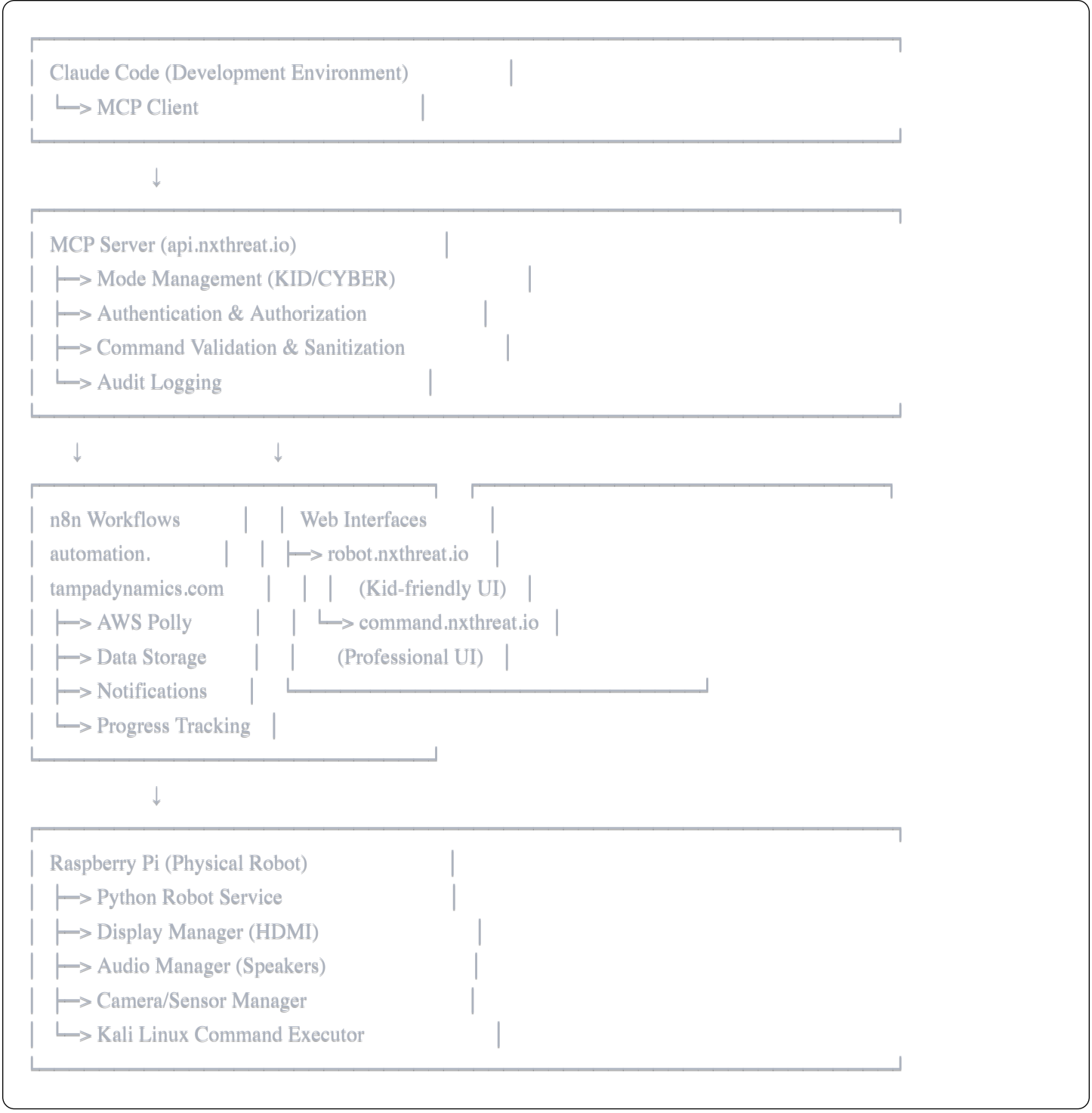
Hardware

- Raspberry Pi (running Kali Linux)
- GPT dongle
- HDMI Output (display)
- ALFA wireless card
- Camera (USB or Pi Camera)
- Speakers/Audio output

Infrastructure

- **Domain**: nxthreat.io
- **n8n Server**: automation.tampadynamics.com (with CLI access)
- **AWS Services**: Polly (text-to-speech)
- **MCP Server**: api.nxthreat.io

🏗 Architecture



📦 Project Structure

```
nxt-robot/
├── mcp-server/              # MCP Server (api.nxthreat.io)
│   ├── src/
│   │   ├── index.ts         # Main MCP server entry
│   │   ├── modes/
│   │   │   ├── kid.ts       # Kid mode tools
│   │   │   └── cyber.ts     # Cyber mode tools
│   │   ├── auth/            # Authentication middleware
│   │   ├── validators/      # Command validation
│   │   └── utils/           # Shared utilities
│   ├── package.json
│   └── tsconfig.json
│
├── web-interfaces/          # Web UIs
│   ├── kid-ui/              # robot.nxthreat.io
│   │   ├── src/
│   │   ├── public/
│   │   └── package.json
│   └── command-ui/          # command.nxthreat.io
│       ├── src/
│       ├── public/
│       └── package.json
│
├── raspberry-pi/            # Robot service for Pi
│   ├── src/
│   │   ├── main.py          # Main robot service
│   │   ├── modes/
│   │   │   ├── kid_mode.py
│   │   │   └── cyber_mode.py
│   │   ├── hardware/
│   │   │   ├── display.py
│   │   │   ├── audio.py
│   │   │   ├── camera.py
│   │   │   └── sensors.py
│   │   ├── kali/
│   │   │   └── executor.py  # Safe Kali command execution
│   │   └── api_client.py    # MCP server communication
│   ├── requirements.txt
│   └── config.yaml
│
├── n8n-workflows/           # n8n workflow exports
│   ├── kid-mode-workflows/
│   │   ├── polly-speech.json
│   │   ├── learning-tracker.json
│   │   └── story-fetcher.json
│   └── cyber-mode-workflows/
```

```
│   ├── scan-processor.json
│   ├── report-generator.json
│   └── alert-notifier.json
│
├── deployment/            # Deployment configs
│   ├── docker/
│   │   ├── mcp-server.Dockerfile
│   │   └── docker-compose.yml
│   ├── nginx/
│   │   └── nxthreat.io.conf
│   └── systemd/
│       └── robot-service.service
│
├── docs/                  # Documentation
│   ├── MCP_TOOLS.md       # MCP tool specifications
│   ├── API.md             # API documentation
│   ├── SECURITY.md        # Security guidelines
│   └── DEPLOYMENT.md      # Deployment guide
│
└── README.md              # This file
```

MCP Server Tools

Kid Mode Tools (Safe, Educational)

typescript

```typescript
// Educational activities
robot_speak(text: string, emotion?: string)
robot_teach_lesson(subject: 'colors' | 'numbers' | 'letters' | 'shapes')
robot_play_game(game: 'simon_says' | 'color_hunt' | 'counting' | 'memory')
robot_tell_story(story_type?: string)

// Interaction
robot_show_emotion(emotion: 'happy' | 'excited' | 'thinking' | 'sleepy')
robot_ask_question(category: string)
robot_give_praise()

// Status
robot_get_learning_progress()
robot_get_status()
```

Cyber Mode Tools (Authenticated, Logged)

typescript

```
// Network scanning
kali_nmap_scan(target: string, scan_type: string, options?: object)
kali_netdiscover(interface: string)

// Enumeration
kali_enum_http(target: string)
kali_enum_smb(target: string)
kali_enum_dns(domain: string)

// Vulnerability assessment
kali_nikto_scan(target: string)
kali_searchsploit(query: string)

// Reporting
kali_generate_report(scan_id: string)
kali_get_scan_results(scan_id: string)

// Reconnaissance (with voice output!)
robot_recon_announce(target: string) // Polly reads scan results

// Status & Safety
robot_get_cyber_status()
kali_validate_target(target: string) // Safety check
```

🔐Security Model

Mode Separation

```python
class RobotMode(Enum):
    KID = "educational"      # No system commands, safe only
    CYBER = "professional"   # Authenticated, full access
    LOCKED = "maintenance"   # Admin only

class SecurityPolicy:
    kid_mode:
        - No network access
        - No file system writes (except learning data)
        - Whitelisted commands only
        - No Kali tools accessible

    cyber_mode:
        - JWT authentication required
        - All commands logged to n8n
        - Target validation (no localhost/private IPs without flag)
        - Rate limiting
        - Session timeout (30 min)
```

Authentication Flow

```
1. User requests cyber mode
2. MCP server validates JWT token
3. n8n webhook logs mode change
4. Pi receives authenticated mode switch
5. LED changes color (visual confirmation)
6. All commands logged with timestamp + user
```

🎨Voice Personas (AWS Polly)

Kid Mode

- **Voice**: Joey (child) or Joanna (warm female)
- **Style**: Encouraging, patient, playful
- **SSML**: Emphasis on positive words, slower pace
- **Examples**:
  - "That's AMAZING! You found the red block!"
  - "Let's count together... One... Two... Three!"

Cyber Mode

- **Voice**: Matthew (professional) or Brian (British)
- **Style**: Technical, precise, efficient
- **SSML**: Normal pace, clear pronunciation of technical terms
- **Examples**:
  - "Scan complete. Five open ports detected."
  - "Vulnerability found: CVE-2024-1234. Severity: High."

🚀Phase 1: Foundation (Week 1)

Goals

- [ ] MCP server running with basic kid mode tools
- [ ] Raspberry Pi service connects to MCP server
- [ ] n8n workflow calls AWS Polly
- [ ] Robot speaks first words via Polly
- [ ] Basic web interface (kid mode)

Deliverables

1. **MCP Server**
    - TypeScript server with kid mode tools
    - WebSocket connection to Pi
    - Basic authentication
2. **n8n Workflows**
    - Webhook → Polly → Return audio URL
    - Learning progress tracker
    - Parent notification system
3. **Pi Service**
    - Python service listening to MCP server
    - Audio playback via speakers
    - Basic display manager (robot face)
4. **First Demo**
    - Kid presses button
    - Robot says "Hello! My name is NXT. Let's learn together!"
    - Parent gets notification via n8n

🔧Phase 2: Kid Mode Complete (Week 2)

Goals

- [ ] Full educational toolkit
- [ ] Camera-based games (color detection)
- [ ] Progress tracking via n8n
- [ ] Interactive robot face with emotions
- [ ] 5+ educational activities

Activities to Build

1. **Color Hunt**: "Show me something RED!"
2. **Counting Game**: Count objects via camera
3. **Simon Says**: Robot gives commands
4. **Story Time**: Polly reads stories with expression
5. **Shape Detective**: Identify shapes via camera
6. **Letter Learning**: Show and pronounce letters

🔴 Phase 3: Cyber Mode Foundation (Week 3)

Goals

- [ ] Secure mode switching
- [ ] Basic Kali command execution
- [ ] Command validation & sanitization
- [ ] Audit logging to n8n
- [ ] Professional web interface

Security Checklist

- [ ] JWT authentication implemented
- [ ] Command whitelist/blacklist
- [ ] Target validation (prevent internal attacks)
- [ ] Rate limiting
- [ ] All commands logged
- [ ] Session management
- [ ] Emergency kill switch

🎯Phase 4: Advanced Features (Week 4+)

Kid Mode Advanced

- [ ] Voice recognition (speech-to-text)
- [ ] Adaptive difficulty based on progress
- [ ] Multi-language support
- [ ] Parent dashboard (view progress)
- [ ] Achievement system

Cyber Mode Advanced

- [ ] Automated scanning workflows
- [ ] Report generation with Polly summaries
- [ ] Integration with vulnerability databases
- [ ] Scan scheduling
- [ ] Multi-target management
- [ ] Professional PDF reports via n8n

🎭The "Transform" Sequence

When switching from KID → CYBER mode:

```python
async def transform_to_cyber_mode():
    await robot_speak("Entering professional mode", voice="kid")
    await led_fade(from_color="blue", to_color="red", duration=3)
    await display_show("matrix_code_animation")
    await robot_speak("Cyber mode activated", voice="professional")
    await play_sound("transform.wav")
```

Visual indicators:

- **Kid Mode**: Blue LEDs, friendly face, bright colors
- **Cyber Mode**: Red LEDs, terminal display, dark theme

📋Development Commands

MCP Server

```bash
cd mcp-server
npm install
npm run dev        # Development
npm run build      # Production build
npm run start      # Start production server
```

Web Interfaces

```bash
# Kid UI
cd web-interfaces/kid-ui
npm install
npm run dev

# Command UI
cd web-interfaces/command-ui
npm install
npm run dev
```

Raspberry Pi Service

```bash
cd raspberry-pi
python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
python src/main.py
```

n8n Workflows (with CLI access)

```bash
# Export workflow
n8n export:workflow --id=<workflow_id> --output=./n8n-workflows/

# Import workflow
n8n import:workflow --input=./n8n-workflows/polly-speech.json

# List workflows
n8n list:workflow
```

🌐Domain Setup

DNS Configuration (nxthreat.io)

```
A     api.nxthreat.io      → [MCP Server IP]
A     robot.nxthreat.io    → [Kid UI Server IP]
A     command.nxthreat.io  → [Command UI Server IP]
CNAME automation           → automation.tampadynamics.com
```

SSL/TLS

```bash
# Use Let's Encrypt
certbot certonly --nginx -d api.nxthreat.io
certbot certonly --nginx -d robot.nxthreat.io
certbot certonly --nginx -d command.nxthreat.io
```

🔌 n8n Integration Points

Webhooks to Create

1. `/webhook/polly-speak` - Text → Polly → Audio URL
2. `/webhook/learning-log` - Log kid's progress
3. `/webhook/scan-result` - Log Kali scan results
4. `/webhook/mode-change` - Log mode switches
5. `/webhook/alert-parent` - Send notifications

Example n8n Flow (Polly Speech)

```
Webhook Trigger
  ↓
[Validate Input]
  ↓
AWS Polly Node
  - Text: {{$json.text}}
  - Voice: {{$json.voice}}
  - OutputFormat: mp3
  ↓
[Store Audio in S3/Local]
  ↓
Return Audio URL
```

📊 Monitoring & Logging

What to Log (via n8n)

- All mode changes (with timestamp + user)
- Every Kali command executed
- Kid mode activity completions
- Learning progress milestones
- Errors and exceptions
- API calls to MCP server

Parent Dashboard (n8n)

- Learning progress charts
- Time spent in kid mode
- Activities completed
- Favorite games
- Achievements unlocked

🧪Testing Strategy

Kid Mode Testing

```bash
# Safety tests
- Attempt to execute system commands → BLOCKED
- Attempt to access file system → BLOCKED
- Attempt network requests → BLOCKED

# Functionality tests
- All educational activities work
- Polly voices play correctly
- Camera detects colors/objects
- Progress tracked to n8n
```

Cyber Mode Testing

```bash
# Security tests
- Unauthenticated access → REJECTED
- Invalid targets → REJECTED
- Command injection attempts → BLOCKED

# Functionality tests
- Nmap scan executes correctly
- Results logged to n8n
- Polly reads results aloud
- Reports generate properly
```

🎓Educational Value

For Your Son (Age 5)

- **Now**: Interactive learning (colors, numbers, shapes)
- **Age 7-10**: Basic programming concepts, logic games
- **Age 10-13**: Understanding how computers work
- **Age 13+**: Intro to cybersecurity, ethical hacking basics

For You (Dad)

- Modern MCP server development
- n8n workflow automation
- AWS service integration
- Raspberry Pi hardware control
- TypeScript/Python fullstack
- Security best practices

🚨Safety & Ethics

Kid Mode Safety

- Zero access to dangerous commands
- All content age-appropriate
- Screen time tracking via n8n
- Parent override controls

Cyber Mode Ethics

- **ONLY** scan authorized targets
- Log everything for accountability
- No exploitation without permission
- Educational purposes only
- Comply with local laws

Legal Considerations

> ⚠️ WARNING: Penetration testing on systems you don't own
> or have explicit permission to test is ILLEGAL.
>
> This robot is for:
> - Learning cybersecurity concepts
> - Testing your own infrastructure
> - Authorized security assessments only

📚Resources

Documentation to Generate

- [ ] MCP_TOOLS.md - Complete tool specifications
- [ ] API.md - REST API documentation
- [ ] SECURITY.md - Security architecture
- [ ] DEPLOYMENT.md - Production deployment guide
- [ ] KID_ACTIVITIES.md - Educational activity ideas
- [ ] KALI_COMMANDS.md - Whitelisted Kali commands

Learning Resources

- MCP Protocol: https://modelcontextprotocol.io
- AWS Polly: https://docs.aws.amazon.com/polly/
- n8n: https://docs.n8n.io/
- Kali Linux: https://www.kali.org/docs/

🎉Success Metrics

Phase 1 Success

- ✅Robot speaks with Polly voice
- ✅Kid can trigger basic interactions
- ✅n8n workflows operational
- ✅MCP server responds to Claude Code

Final Success

- ✅Son uses robot daily for learning
- ✅Measurable educational progress
- ✅You use robot for pentesting tasks
- ✅Both learn modern dev practices
- ✅System secure and well-documented
- ✅Family impressed at demos 😎

🤝Contributing (Family Project)

This is a father-son learning project, but the architecture is solid enough to:

- Open source (if you want)
- Blog about the journey
- Present at local meetups
- Inspire other parent-child maker projects

📝Notes for Claude Code

You Have Access To:

- ✅This project structure
- ✅n8n CLI for workflow management
- ✅AWS credentials (assumed for Polly)
- ✅Domain: nxthreat.io
- ✅n8n instance: automation.tampadynamics.com

Build Priority:

1. **Start with MCP server** (TypeScript)
2. **Create n8n Polly workflow** (test voice first!)
3. **Build Pi Python service** (get it talking)
4. **First demo** (robot says hello)
5. **Iterate from there**

Key Principles:

- 🔒**Security first** - Mode separation is critical
- 👶**Kid-safe by default** - Lock down kid mode completely
- 📝**Log everything** - Audit trail via n8n
- 🎨**Make it fun** - This is for a 5-year-old!
- 🧠**Make it smart** - This is for learning

🚀Let's Build This!

Next Steps:

1. Review this architecture
2. Set up MCP server skeleton
3. Create first n8n workflow (Polly test)
4. Get robot saying "Hello World"
5. Build from there!

The robot awaits... let's bring it to life! 🤖

"The best way to predict the future is to build it... together with your son."