⚡ Quick Start Guide - Automatos AI

Get up and running with the world’s most advanced multi-agent orchestration platform in under 10 minutes!

🎯 What You’ll Accomplish

By the end of this guide, you’ll have:
- ✅ **Running Platform**: Full Automatos AI stack deployed locally
- ✅ **First Workflow**: Automated deployment pipeline created
- ✅ **Live Dashboard**: Real-time monitoring and control interface
- ✅ **Agent Activity**: Multi-agent coordination in action

🚀 Option 1: Lightning Fast Setup (Recommended)

Step 1: One-Command Deployment

```
# Clone and deploy in one go
curl -sSL https://raw.githubusercontent.com/automotas-ai/automotas/main/scripts/quick-deploy.sh | bash
```

This script automatically:
- Clones the repository
- Sets up environment variables
- Starts all services with Docker Compose
- Opens the dashboard in your browser

Step 2: Access Your Platform

Once deployment completes (2-3 minutes):

- 🎨 **Dashboard**: http://localhost:3000
- 📚 **API Docs**: http://localhost:8002/docs
- 📊 **Monitoring**: http://localhost:3001 (Grafana)

Default Login:
- Username: `admin`
- Password: `automatos123`

🔧 Option 2: Manual Setup

Prerequisites Check

```
# Verify you have the required tools
docker --version    # Should be 20.10+
docker-compose --version  # Should be 1.29+
git --version      # Should be 2.30+
```

Step 1: Clone Repository

```
git clone https://github.com/automotas-ai/automotas.git
cd automotas
```

Step 2: Environment Configuration

```
# Copy and edit environment file
cp .env.example .env

# Required: Add your OpenAI API key
echo "OPENAI_API_KEY=your_openai_api_key_here" >> .env

# Optional: Customize other settings
# POSTGRES_PASSWORD=your_secure_password
# REDIS_PASSWORD=your_redis_password
# JWT_SECRET=your_jwt_secret
```

Step 3: Start Services

```
# Start all services
docker-compose up -d

# Verify services are running
docker-compose ps
```

Step 4: Initialize Database

```
# Wait for PostgreSQL to be ready (30 seconds)
sleep 30

# Initialize database with sample data
docker-compose exec backend python -c "
from context_manager import init_database
init_database()
print('Database initialized successfully!')
"
```

🎬 Create Your First Workflow

Using the Dashboard (Easiest)

1. **Open Dashboard**: Navigate to http://localhost:3000
2. **Login**: Use default credentials or create new account
3. **Create Workflow**: Click "New Workflow" button
4. **Configure**:
   - **Repository**: `https://github.com/automotas-ai/sample-app.git`
   - **Type**: AI Module
   - **Environment**: Development
5. **Deploy**: Click "Create Workflow"
6. **Monitor**: Watch real-time agent coordination!

Using the API

```
# Create workflow via API
curl -X POST http://localhost:8002/api/workflows \
  -H "Content-Type: application/json" \
  -H "X-API-Key: your_api_key" \
  -d '{
    "repository_url": "https://github.com/automotas-ai/sample-app.git",
    "workflow_type": "ai_module",
    "environment": "development",
    "priority": "normal"
  }'
```

Using the CLI

```
# Install CLI (optional)
pip install automotas-cli

# Create workflow
automotas workflow create \
  --repo https://github.com/automotas-ai/sample-app.git \
  --type ai_module \
  --env development

# Monitor progress
automotas workflow status --follow
```

👁️ Watch the Magic Happen

Real-time Dashboard

Your workflow dashboard shows:
- 🎯 **Current Stage**: Which agents are active
- 📊 **Progress**: Real-time completion percentage
- 📝 **Live Logs**: Streaming logs from all agents
- 🤖 **Agent Activity**: What each agent is doing
- ⚡ **Performance**: Resource usage and optimization

Expected Timeline

Phase	Duration	What's Happening
Analysis	1-2 min	Repository cloning and structure analysis
Planning	2-3 min	Strategy agent creates deployment plan
Security	1-2 min	Security agent validates configuration
Execution	5-10 min	Deployment agent executes the plan
Optimization	1-2 min	Performance optimization and health checks

Sample Workflow Progress

```
🎯 Workflow: sample-app-deployment
📊 Progress: 65% Complete | ⏱️ 8 minutes remaining

🤖 Active Agents:
   Strategy Agent     ✅ Plan Complete (98% confidence)
   Security Agent     ✅ Validation Passed
   Execution Agent    🔄 Deploying containers (3/5 complete)
   Monitor Agent      ⏸️ Waiting for deployment

📝 Recent Activity:
   15:23:45 | Execution | Container 'api-server' started successfully
   15:23:42 | Execution | Database migration completed
   15:23:38 | Security  | SSL certificate validated
   15:23:35 | Strategy  | Deployment plan optimized for performance
```

🔍 Explore Key Features

1. Multi-Agent Coordination

Watch your agents collaborate:
- **Strategy Agent**: Analyzes your repository and creates optimal deployment plan
- **Security Agent**: Validates security requirements and compliance
- **Execution Agent**: Handles actual deployment with monitoring
- **Analysis Agent**: Provides performance insights and optimization

2. Context Engineering

Test the knowledge system:

```
# Upload a document
curl -X POST http://localhost:8002/api/admin/documents/upload \
  -H "Content-Type: multipart/form-data" \
  -F "file=@your-deployment-guide.pdf"

# Search for context
curl -X POST http://localhost:8002/api/context/search \
  -H "Content-Type: application/json" \
  -d '{"query": "deployment best practices", "max_results": 5}'
```

3. Real-time Monitoring

Access comprehensive monitoring:
- **System Metrics**: http://localhost:3001
- **Agent Performance**: Dashboard > Analytics
- **Resource Usage**: Real-time CPU, memory, network stats
- **Error Tracking**: Automated error detection and alerting

🎯 Sample Use Cases

Web Application Deployment

```yaml
# ai-module.yaml
name: "my-web-app"
module_type: "web_app"
framework: "nodejs"

build:
  command: "npm install && npm run build"

runtime:
  start_command: "npm start"
  port: 3000

infrastructure:
  replicas:
    min: 2
    max: 10
  autoscaling:
    target_cpu: 70
```

Microservices Architecture

```
# Task prompt for complex deployment
automotas workflow create \
  --repo https://github.com/yourorg/microservices-app.git \
  --type task_prompt \
  --prompt "Deploy microservices with API gateway, user service, payment service, and database. Set up service mesh with monitoring and tracing."
```

Data Pipeline

```
# AI-powered data processing workflow
automotas workflow create \
  --repo https://github.com/yourorg/data-pipeline.git \
  --type task_prompt \
  --prompt
"Create ETL pipeline processing CSV files from S3, transforming with Python, and
storing in PostgreSQL with Airflow orchestration."
```

📚 Next Steps

🔥 Immediate Actions

1. **Upload Documents (http://localhost:3000/admin/documents)**: Add your deployment guides and documentation
2. **Configure Agents (http://localhost:3000/agents)**: Customize agent capabilities and behavior
3. **Create Team Account (http://localhost:3000/settings)**: Set up multi-user access
4. **Explore Analytics (http://localhost:3000/analytics)**: Review performance and optimization opportunities

📖 Learn More

- **Comprehensive Guide (COMPREHENSIVE_GUIDE.md)**: Deep dive into all platform capabilities
- **API Reference (API_REFERENCE.md)**: Complete API documentation
- **Architecture Guide (architecture.md)**: Understanding the system design
- **Contributing Guide (CONTRIBUTING.md)**: Join our community of contributors

🚀 Advanced Features

- **Multi-Agent Systems (multi-agent-systems.md)**: Advanced agent coordination
- **Context Engineering (CONTEXT_ENGINEERING.md)**: RAG system and knowledge management
- **Field Theory Integration (field-theory.md)**: Mathematical foundations
- **Security & Compliance (security.md)**: Enterprise-grade security setup

---

🆘 Troubleshooting

Common Issues

Port Already in Use

```
# Check what's using port 3000
lsof -ti:3000

# Kill the process
kill -9 $(lsof -ti:3000)

# Restart services
docker-compose down && docker-compose up -d
```

Database Connection Failed

```
# Reset database
docker-compose down -v
docker-compose up -d postgres
sleep 30
docker-compose up -d
```

OpenAI API Errors

```
# Verify API key is set
grep OPENAI_API_KEY .env

# Test API key
curl -H "Authorization: Bearer your_api_key_here" \
  https://api.openai.com/v1/models
```

Getting Help

- 💬 **Discord Community (https://discord.gg/automotas)**: Real-time help from the community
- 🐛 **GitHub Issues (https://github.com/automotas-ai/automotas/issues)**: Report bugs or request features
- 📧 **Support Email**: Direct technical support
- 📖 **Documentation (https://docs.automotas.ai)**: Comprehensive guides and references

---

✨ Success!

🎉 **Congratulations!** You now have a fully functional multi-agent orchestration platform running locally.

What You've Accomplished

- ✅ Deployed enterprise-grade AI automation platform
- ✅ Created and monitored your first intelligent workflow
- ✅ Witnessed multi-agent collaboration in action
- ✅ Gained hands-on experience with context engineering

You're Ready For

- 🚀 **Production Deployment**: Scale to handle real workloads
- 🤝 **Team Collaboration**: Add team members and manage permissions
- 🔧 **Custom Development**: Extend platform with custom agents and integrations
- 🏢 **Enterprise Features**: Implement advanced security and compliance

---

Ready to revolutionize your automation workflow?

🎯 **Explore Advanced Features (COMPREHENSIVE_GUIDE.md)** | 🤝 **Join Our Community (CONTRIBUTING.md)** | 🏢 **Go to Production (enterprise-deployment.md)**

Welcome to the future of intelligent automation!