Pre-Installation Checklist & Setup Guide

Overview

Before starting development, you need to install and configure the following components. This checklist ensures you have everything ready for Claude Code to begin autonomous development.

Required Software & Services

1. Python 3.11+ ✅ CRITICAL

What it is: Programming language for the entire project

Check if installed:

```bash
python3 --version
# or
python --version
```

Installation:

- **macOS:** `brew install python@3.11`
- **Ubuntu/Debian:** `sudo apt install python3.11 python3.11-venv`
- **Windows:** Download from python.org

Verify: Should show version 3.11.x or higher

---

2. Git ✅ CRITICAL

What it is: Version control for tracking code changes

Check if installed:

```bash
git --version
```

Installation:

- **macOS:** `brew install git`
- **Ubuntu/Debian:** `sudo apt install git`
- **Windows:** Download from git-scm.com

Configuration:

```bash
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com"
```

---

3. Interactive Brokers (IBKR) ✅ CRITICAL

What it is: Trading platform and API provider

Required Accounts:

1. **IBKR Account** (if you don't have one)
   - Sign up at: https://www.interactivebrokers.com
   - Choose "Individual" account
   - Enable "Paper Trading" account (free, no funding required)
2. **Paper Trading Account**
   - Automatically created with main account
   - Separate login credentials
   - Unlimited virtual money for testing
   - Real-time market data (may require subscription)

Required Software:

- **TWS (Trader Workstation)** OR **IB Gateway**
  - Download: https://www.interactivebrokers.com/en/trading/tws.php
  - IB Gateway is lighter (recommended for API use)

Configuration:

1. Install TWS or IB Gateway
2. Log in with **PAPER TRADING** credentials
3. Go to: File → Global Configuration → API → Settings
4. Enable:
   - ✅ "Enable ActiveX and Socket Clients"
   - ✅ "Read-Only API" (for safety during development)
   - Socket port: `7497` (paper trading)
5. Add to "Trusted IP Addresses": `127.0.0.1`

Test Connection:

```bash
# Gateway/TWS must be running
# Try connecting with ib_insync (after Python setup)
```

---

4. Anthropic API Key ✅ CRITICAL

What it is: Access to Claude AI for intelligent agents

How to get:

1. Go to: https://console.anthropic.com/
2. Sign up / Log in
3. Go to: API Keys section
4. Create new API key
5. **COPY AND SAVE SECURELY** - shown only once!

Pricing:

- Claude Sonnet 4: ~$3 per million input tokens, ~$15 per million output tokens
- Estimated cost: $30-50/month for this project

Note: You'll add this to `.env` file during setup

---

5. Database System 📊 INCLUDED (SQLite)

What it is: Storage for trades, learning history, experiments

For Development (Recommended):

- **SQLite** - Built into Python, no installation needed
- File-based, simple, perfect for paper trading
- Located in: `data/databases/trades.db`

For Production (Optional - Future):

- **PostgreSQL** - If you scale to production
- Installation:
  - **macOS:** `brew install postgresql`
  - **Ubuntu:** `sudo apt install postgresql postgresql-contrib`

- **Windows:** Download from postgresql.org

Decision: Start with SQLite, migrate to PostgreSQL only if needed later

---

6. Text Editor / IDE (Optional but Recommended)

Recommended Options:

1. **VS Code** (Most popular)
   - Download: https://code.visualstudio.com/
   - Extensions to install:
     - Python
     - Pylance
     - Black Formatter
     - Ruff
     - GitHub Copilot (optional)
2. **PyCharm Community Edition** (Python-focused)
   - Download: https://www.jetbrains.com/pycharm/download/
3. **Cursor** (AI-first editor)
   - Download: https://cursor.sh/
   - Built-in Claude integration

Why? Better than basic text editors for Python development

Optional but Recommended

7. Make (Build automation)

What it is: Simplifies running common commands

Check if installed:

```bash
make --version
```

Installation:

- **macOS:** Included with Xcode Command Line Tools: `xcode-select --install`
- **Ubuntu/Debian:** `sudo apt install build-essential`

- **Windows:** Install with `choco install make` (requires Chocolatey)

Usage: Allows commands like `make test` instead of `pytest tests/`

---

8. Docker (Future - for deployment)

What it is: Containerization for consistent environments

Not needed for development, but helpful later for:

- Running PostgreSQL in container
- Deploying to cloud
- Consistent environments across machines

Installation: https://docs.docker.com/get-docker/

Skip for now - Install later if needed

Pre-Development Setup Steps

Step 1: Create Project Directory

```bash
# Choose a location for your project
mkdir -p ~/projects/trading_agent
cd ~/projects/trading_agent

# Initialize git repository
git init

# Create basic structure
mkdir -p src tests scripts data logs docs
```

Step 2: Set Up Python Virtual Environment

```bash
```

```
# Create virtual environment
python3.11 -m venv venv

# Activate it
source venv/bin/activate  # macOS/Linux
# OR
venv\Scripts\activate     # Windows

# Upgrade pip
pip install --upgrade pip

# Install initial tools
pip install black ruff mypy pytest
```

Step 3: Create Environment File

bash

```bash
# Create .env file
touch .env

# Add to .env (edit with your values):
cat > .env << 'EOF'
# IBKR Configuration
IBKR_HOST=127.0.0.1
IBKR_PORT=7497
IBKR_CLIENT_ID=1
IBKR_ACCOUNT=DU123456

# Anthropic API
ANTHROPIC_API_KEY=your_api_key_here

# Database
DATABASE_URL=sqlite:///data/databases/trades.db

# Application Settings
PAPER_TRADING=true
LOG_LEVEL=INFO
LEARNING_ENABLED=true

# Risk Limits
MAX_DAILY_LOSS=-0.02
MAX_POSITION_SIZE=5000
EOF

# Secure the file (contains secrets)
chmod 600 .env

# Create example template
cp .env .env.example
# Edit .env.example and replace actual values with placeholders
```

Step 4: Create .gitignore

bash

```
cat > .gitignore << 'EOF'
# Python
__pycache__/
*.py[cod]
*$py.class
*.so
.Python
env/
venv/
build/
dist/
*.egg-info/

# Environment
.env
.env.local

# IDE
.vscode/
.idea/
*.swp
*.swo
*~

# Data
data/databases/*.db
data/cache/
logs/*.log

# Testing
.pytest_cache/
.coverage
htmlcov/

# OS
.DS_Store
Thumbs.db
EOF
```

Step 5: Create Initial requirements.txt

bash

```
cat > requirements.txt << 'EOF'
# Trading & Market Data
ib_insync==0.9.86
pandas==2.2.0
numpy==1.26.4

# AI & Machine Learning
anthropic==0.76.0
scikit-learn==1.4.0
scipy==1.12.0

# Database
sqlalchemy==2.0.25
alembic==1.13.1

# Utilities
python-dotenv==1.0.0
pydantic==2.5.0
loguru==0.7.2
typer==0.9.0
rich==13.7.0

# Testing
pytest==7.4.3
pytest-cov==4.1.0
pytest-asyncio==0.21.1

# Code Quality
black==23.12.1
ruff==0.1.9
mypy==1.8.0
EOF

# Install all dependencies
pip install -r requirements.txt
```

Step 6: Verify IBKR Connection

bash

```
# Create test script
cat > test_ibkr_connection.py << 'EOF'
from ib_insync import IB
import os
from dotenv import load_dotenv

load_dotenv()

print("Testing IBKR connection...")

ib = IB()
try:
    ib.connect(
        host=os.getenv("IBKR_HOST", "127.0.0.1"),
        port=int(os.getenv("IBKR_PORT", 7497)),
        clientId=int(os.getenv("IBKR_CLIENT_ID", 1))
    )

    print("✅ Connection successful!")
    print(f"   Account: {ib.managedAccounts()}")

    # Test market data
    from ib_insync import Stock
    spy = Stock('SPY', 'SMART', 'USD')
    ib.qualifyContracts(spy)
    ticker = ib.reqMktData(spy)
    ib.sleep(2)

    print(f"✅ Market data working!")
    print(f"   SPY: ${ticker.last}")

    ib.disconnect()
    print("✅ All tests passed!")

except Exception as e:
    print(f"❌ Error: {e}")
    print("\nTroubleshooting:")
    print("1. Is TWS/IB Gateway running?")
    print("2. Is it logged into PAPER TRADING account?")
    print("3. Is API enabled in settings?")
    print("4. Is port 7497 correct?")
EOF

# Run test (TWS/Gateway must be running)
python test_ibkr_connection.py
```

Step 7: Verify Anthropic API

```
bash

# Create test script
cat > test_anthropic_api.py << 'EOF'
from anthropic import Anthropic
import os
from dotenv import load_dotenv

load_dotenv()

print("Testing Anthropic API...")

client = Anthropic(api_key=os.getenv("ANTHROPIC_API_KEY"))

try:
    message = client.messages.create(
        model="claude-sonnet-4-20250514",
        max_tokens=100,
        messages=[
            {"role": "user", "content": "Say 'API test successful' and nothing else"}
        ]
    )

    print(f"✅ {message.content[0].text}")

except Exception as e:
    print(f"❌ Error: {e}")
    print("\nCheck:")
    print("1. Is ANTHROPIC_API_KEY set correctly in .env?")
    print("2. Is API key valid? (Check console.anthropic.com)")
EOF

# Run test
python test_anthropic_api.py
```

Verification Checklist

Run through this checklist before starting development:

Software Installation

- [ ] Python 3.11+ installed and working

- [ ] Git installed and configured
- [ ] TWS or IB Gateway installed
- [ ] Text editor/IDE installed (optional)

Accounts & API Keys

- [ ] IBKR account created
- [ ] IBKR Paper Trading account accessible
- [ ] Anthropic account created
- [ ] Anthropic API key obtained and saved

Project Setup

- [ ] Project directory created
- [ ] Virtual environment created and activated
- [ ] Dependencies installed (`requirements.txt`)
- [ ] `.env` file created with all credentials
- [ ] `.gitignore` created

Connection Tests

- [ ] IBKR connection test passes
- [ ] Anthropic API test passes
- [ ] Can fetch market data (SPY price, etc.)

Development Environment

- [ ] Can run Python scripts
- [ ] Can run tests with `pytest`
- [ ] Git commits working
- [ ] Logs directory exists

Troubleshooting Common Issues

IBKR Connection Fails

Error: "Connection refused"

- ✅ TWS/Gateway is running
- ✅ Logged into PAPER TRADING account (not live!)
- ✅ API settings enabled
- ✅ Port is 7497 (not 7496 for live)

Error: "No security definition found"

- ✅ Check symbol name (use 'SPY', not 'SPDR S&P 500')
- ✅ Qualify contracts before requesting data

Error: "Market data not subscribed"

- ✅ May need to subscribe to market data in IBKR
- ✅ Paper trading has some free data, live requires subscription

Python Environment Issues

Error: "Module not found"

```bash
# Make sure virtual environment is activated
which python  # Should point to venv/bin/python

# Reinstall dependencies
pip install -r requirements.txt
```

Error: "Permission denied" when installing

```bash
# Don't use sudo with pip in virtual environment
# If using system Python, create virtual environment first
```

Anthropic API Issues

Error: "Invalid API key"

- ✅ Check for typos in `.env`
- ✅ Verify key at console.anthropic.com
- ✅ Key may need billing enabled

Error: "Rate limit exceeded"

- ✅ You're making too many requests
- ✅ Wait and try again
- ✅ Check if multiple processes using API

Database Setup

SQLite (Default - No installation needed)

SQLite is included with Python - no setup required!

Test database creation:

```python
import sqlite3

# Create test database
conn = sqlite3.connect('data/databases/test.db')
cursor = conn.cursor()

# Create test table
cursor.execute('''
    CREATE TABLE test (
        id INTEGER PRIMARY KEY,
        name TEXT
    )
''')

# Insert test data
cursor.execute("INSERT INTO test (name) VALUES ('works')")
conn.commit()

# Query
result = cursor.execute("SELECT * FROM test").fetchall()
print(f"✅ Database working: {result}")

conn.close()
```

PostgreSQL (Optional - For Production)

Only install if you want to use PostgreSQL instead of SQLite:

```bash
```

```
# macOS
brew install postgresql
brew services start postgresql

# Ubuntu
sudo apt install postgresql postgresql-contrib
sudo systemctl start postgresql

# Create database
createdb trading_system

# Test connection
psql trading_system
```

Update .env for PostgreSQL:

```
bash

DATABASE_URL=postgresql://username:password@localhost/trading_system
```

Optional: Set Up Makefile

Create `Makefile` for common commands:

```
makefile
```

```
.PHONY: install test format lint clean

install:
  pip install -r requirements.txt

test:
  pytest tests/ -v

test-cov:
  pytest tests/ --cov=src --cov-report=html

format:
  black src/ tests/

lint:
  ruff check src/ tests/
  mypy src/

clean:
  find . -type d -name __pycache__ -exec rm -rf {} +
  find . -type f -name "*.pyc" -delete
  rm -rf .pytest_cache .coverage htmlcov/

run-scanner:
  python -m src.cli.main scan

run-trader:
  python -m src.cli.main trade
```

Usage:

```bash
make install    # Install dependencies
make test       # Run tests
make format     # Format code
make lint       # Check code quality
```

Ready to Start Checklist

Before telling Claude Code to begin development:

✅ Prerequisites

- [ ] All software installed (Python, Git, IBKR, etc.)

- [ ] All accounts created (IBKR Paper, Anthropic)
- [ ] All API keys obtained and saved securely
- [ ] Virtual environment created and activated
- [ ] Dependencies installed successfully

✅ Project Structure

- [ ] Project directory created
- [ ] `.env` file configured with credentials
- [ ] `.gitignore` created
- [ ] `requirements.txt` created
- [ ] Initial git commit made

✅ Connection Tests

- [ ] IBKR connection successful
- [ ] Anthropic API working
- [ ] Can fetch market data
- [ ] Database operations working

✅ Development Tools

- [ ] Can run Python scripts
- [ ] Can run tests with pytest
- [ ] Code formatting works (black)
- [ ] Linting works (ruff)

✅ Documentation

- [ ] CLAUDE.md in project root
- [ ] SPEC_TRADING_SYSTEM.md in project root
- [ ] README.md created (basic)

Cost Estimates

One-Time Costs

- **IBKR Account:** $0 (paper trading is free)
- **Market Data Subscription:** $0-10/month (paper may be free)
- **Anthropic API:** $0 initial (pay-as-you-go)

Monthly Operating Costs

- **Anthropic API:** $30-50/month (estimated)

- **IBKR Paper Trading:** $0
- **Database (SQLite):** $0
- **Server/Hosting:** $0 (running locally)

Total Estimated Monthly Cost: $30-50

Future Production Costs (if going live)

- **IBKR Live Trading:** $0 (commissions on trades only)
- **Market Data (live):** $10-30/month depending on subscriptions
- **Database (PostgreSQL):** $0 (local) or $15-50/month (cloud)
- **Server Hosting:** $0 (local) or $20-100/month (cloud)

Next Steps

Once all prerequisites are installed and verified:

1. ✅ Create `CLAUDE.md` in project root (already done)
2. ✅ Create `SPEC_TRADING_SYSTEM.md` in project root (next)
3. ✅ Verify all connection tests pass
4. ✅ Make initial git commit
5. 🚀 Tell Claude Code: "Please begin Phase 0 implementation following SPEC_TRADING_SYSTEM.md"

Quick Reference: Essential Commands

```bash
```

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

# Install dependencies
pip install -r requirements.txt

# Run tests
pytest

# Format code
black src/ tests/

# Lint code
ruff check src/ tests/

# Type check
mypy src/

# Run application
python -m src.cli.main --help

# Start development
# (after Claude Code has been given CLAUDE.md and SPEC_TRADING_SYSTEM.md)
```

Document Version: 1.0
Last Updated: January 2025
Status: Complete - Ready for project initialization