SwingTrader Backend — Development Task List

Project: AI-Ready Swing Trading Backtesting System for Swedish Equities
Phase: Backend Development
Created: January 2025

Overview

This task list covers the complete backend implementation for the SwingTrader system. Tasks are organized by module, with dependencies, priorities, and estimated effort clearly marked.

Priority Legend:

- 🔴 **P0** — Critical / Blocker
- 🟠 **P1** — High Priority
- 🟡 **P2** — Medium Priority
- 🟢 **P3** — Nice to Have

Status Legend:

- ⬜ Not Started
- 🔄 In Progress
- ✅ Complete
- ⏸️ Blocked

Module 1: Data Pipeline

1.1 Data Fetching

ID	Task	Priority	Status	Effort	Dependencies
DP-001	Set up yfinance integration for Swedish stocks	🔴 P0	⬜	2h	None
DP-002	Create list of OMX30 stock symbols with metadata	🔴 P0	⬜	1h	None
DP-003	Implement daily OHLCV data fetcher	🔴 P0	⬜	3h	DP-001
DP-004	Add support for multiple timeframes (daily, weekly)	🟡 P2	⬜	2h	DP-003

ID	Task	Priority	Status	Effort	Dependencies
DP-005	Implement rate limiting and retry logic	🟠 P1	⬜	2h	DP-003
DP-006	Create data fetching scheduler (daily updates)	🟡 P2	⬜	3h	DP-003

1.2 Data Storage

ID	Task	Priority	Status	Effort	Dependencies
DP-007	Design database schema for price data	🔴 P0	⬜	2h	None
DP-008	Set up SQLite/PostgreSQL database	🔴 P0	⬜	2h	DP-007
DP-009	Implement price data models (SQLAlchemy/Peewee)	🔴 P0	⬜	3h	DP-008
DP-010	Create data insertion/upsert functions	🔴 P0	⬜	2h	DP-009
DP-011	Implement data retrieval queries (by symbol, date range)	🔴 P0	⬜	2h	DP-009
DP-012	Add database indexing for performance	🟡 P2	⬜	1h	DP-009

1.3 Data Validation & Cleaning

ID	Task	Priority	Status	Effort	Dependencies
DP-013	Implement OHLCV data validation (price sanity checks)	🟠 P1	⬜	2h	DP-003
DP-014	Handle missing data (weekends, holidays, gaps)	🟠 P1	⬜	2h	DP-013
DP-015	Detect and handle stock splits/dividends	🟡 P2	⬜	3h	DP-013
DP-016	Implement outlier detection for price anomalies	🟡 P2	⬜	2h	DP-013
DP-017	Create data quality reporting function	🟢 P3	⬜	2h	DP-013

Module 2: Technical Indicators

2.1 Moving Averages

ID	Task	Priority	Status	Effort	Dependencies
TI-001	Implement Simple Moving Average (SMA)	🔴 P0	⬜	1h	DP-011
TI-002	Implement Exponential Moving Average (EMA)	🔴 P0	⬜	1h	DP-011
TI-003	Create configurable MA periods (10, 20, 50, 200)	🔴 P0	⬜	1h	TI-001, TI-002
TI-004	Implement MA crossover detection	🟠 P1	⬜	2h	TI-003
TI-005	Calculate price distance from MA (% deviation)	🟠 P1	⬜	1h	TI-003

2.2 Momentum Indicators

ID	Task	Priority	Status	Effort	Dependencies
TI-006	Implement RSI calculation (14-period default)	🔴 P0	⬜	2h	DP-011
TI-007	Add configurable RSI periods	🟠 P1	⬜	1h	TI-006
TI-008	Implement RSI divergence detection	🟡 P2	⬜	3h	TI-006
TI-009	Implement MACD indicator	🟡 P2	⬜	2h	TI-002
TI-010	Implement Stochastic Oscillator	🟢 P3	⬜	2h	DP-011

2.3 Volatility Indicators

ID	Task	Priority	Status	Effort	Dependencies
TI-011	Implement ATR (Average True Range)	🔴 P0	⬜	2h	DP-011
TI-012	Create ATR-based stop loss calculator	🟠 P1	⬜	2h	TI-011
TI-013	Implement Bollinger Bands	🟡 P2	⬜	2h	TI-001
TI-014	Calculate historical volatility (std dev)	🟡 P2	⬜	1h	DP-011

2.4 Volume Analysis

ID	Task	Priority	Status	Effort	Dependencies
TI-015	Calculate volume moving average	🟠 P1	⬜	1h	DP-011
TI-016	Implement volume ratio (current/average)	🟠 P1	⬜	1h	TI-015
TI-017	Detect volume spikes/anomalies	🟠 P1	⬜	2h	TI-016
TI-018	Implement On-Balance Volume (OBV)	🟢 P3	⬜	2h	DP-011
TI-019	Implement Volume-Price Trend (VPT)	🟢 P3	⬜	2h	DP-011

2.5 Indicator Framework

ID	Task	Priority	Status	Effort	Dependencies
TI-020	Create base Indicator class/interface	🟠 P1	⬜	2h	None
TI-021	Implement indicator caching mechanism	🟡 P2	⬜	2h	TI-020
TI-022	Create indicator factory for dynamic loading	🟡 P2	⬜	2h	TI-020
TI-023	Add unit tests for all indicators	🟠 P1	⬜	4h	TI-001 to TI-019

Module 3: Signal Generation

3.1 Entry Signals

ID	Task	Priority	Status	Effort	Dependencies
SG-001	Implement MA pullback entry signal	🔴 P0	⬜	3h	TI-003, TI-005
SG-002	Implement MA crossover entry signal	🟠 P1	⬜	2h	TI-004
SG-003	Implement RSI oversold entry signal	🔴 P0	⬜	2h	TI-006
SG-004	Implement volume breakout entry signal	🟠 P1	⬜	3h	TI-017
SG-005	Create composite entry signal (multiple conditions)	🟠 P1	⬜	3h	SG-001 to SG-004

3.2 Exit Signals

ID	Task	Priority	Status	Effort	Dependencies
SG-006	Implement fixed percentage stop loss	🔴 P0	⬜	2h	None
SG-007	Implement ATR-based trailing stop	🟠 P1	⬜	3h	TI-011
SG-008	Implement fixed percentage take profit	🔴 P0	⬜	2h	None
SG-009	Implement time-based exit (max holding period)	🟡 P2	⬜	2h	None
SG-010	Implement MA crossover exit signal	🟡 P2	⬜	2h	TI-004
SG-011	Implement RSI overbought exit signal	🟡 P2	⬜	2h	TI-006

3.3 Market Phase Detection

ID	Task	Priority	Status	Effort	Dependencies
SG-012	Define market phase criteria (accumulation, markup, distribution, markdown)	🟠 P1	⬜	2h	None
SG-013	Implement trend detection (using MA slope)	🟠 P1	⬜	2h	TI-003
SG-014	Implement accumulation phase detector	🟡 P2	⬜	3h	SG-012, TI-015
SG-015	Implement distribution phase detector	🟡 P2	⬜	3h	SG-012, TI-015
SG-016	Create phase-based trade filter	🟡 P2	⬜	2h	SG-013 to SG-015

3.4 Signal Framework

ID	Task	Priority	Status	Effort	Dependencies
SG-017	Create base Signal class/interface	🔴 P0	⬜	2h	None
SG-018	Implement signal combiner (AND/OR logic)	🟠 P1	⬜	2h	SG-017
SG-019	Create signal configuration schema (JSON/YAML)	🟠 P1	⬜	2h	SG-017

ID	Task	Priority	Status	Effort	Dependencies
SG-020	Add signal logging and debugging output	🟡 P2	⬜	2h	SG-017

Module 4: Backtesting Engine

4.1 Core Engine

ID	Task	Priority	Status	Effort	Dependencies
BT-001	Create BacktestEngine class structure	🔴 P0	⬜	3h	None
BT-002	Implement event-driven simulation loop	🔴 P0	⬜	4h	BT-001
BT-003	Create Trade class (entry, exit, P&L tracking)	🔴 P0	⬜	2h	None
BT-004	Implement position tracking (open/closed)	🔴 P0	⬜	2h	BT-003
BT-005	Create Portfolio class (cash, positions, equity)	🔴 P0	⬜	3h	BT-003

4.2 Trade Execution

ID	Task	Priority	Status	Effort	Dependencies
BT-006	Implement market order execution	🔴 P0	⬜	2h	BT-002
BT-007	Implement limit order execution	🟡 P2	⬜	2h	BT-002
BT-008	Implement stop order execution	🟠 P1	⬜	2h	BT-002
BT-009	Add execution slippage modeling	🟠 P1	⬜	2h	BT-006
BT-010	Implement fill price simulation (open, close, VWAP)	🟡 P2	⬜	2h	BT-006

4.3 Risk Management

ID	Task	Priority	Status	Effort	Dependencies
BT-011	Implement fixed position sizing	🔴 P0	⬜	2h	BT-005
BT-012	Implement percentage-of-equity position sizing	🟠 P1	⬜	2h	BT-005

ID	Task	Priority	Status	Effort	Dependencies
BT-013	Implement ATR-based position sizing	🟠 P1	⬜	2h	TI-011, BT-005
BT-014	Add maximum position limit per stock	🟠 P1	⬜	1h	BT-005
BT-015	Add maximum portfolio exposure limit	🟠 P1	⬜	1h	BT-005
BT-016	Implement Kelly Criterion position sizing	🟢 P3	⬜	3h	BT-005

4.4 Transaction Costs

ID	Task	Priority	Status	Effort	Dependencies
BT-017	Implement commission modeling (fixed + percentage)	🔴 P0	⬜	2h	BT-006
BT-018	Add spread cost modeling	🟠 P1	⬜	1h	BT-006
BT-019	Implement tax calculation (Swedish capital gains)	🟢 P3	⬜	3h	BT-003
BT-020	Create configurable cost profiles	🟡 P2	⬜	2h	BT-017, BT-018

4.5 Performance Metrics

ID	Task	Priority	Status	Effort	Dependencies
BT-021	Calculate total return	🔴 P0	⬜	1h	BT-005
BT-022	Calculate win rate	🔴 P0	⬜	1h	BT-003
BT-023	Calculate profit factor	🔴 P0	⬜	1h	BT-003
BT-024	Calculate expectancy (average R)	🔴 P0	⬜	1h	BT-003
BT-025	Calculate maximum drawdown	🔴 P0	⬜	2h	BT-005
BT-026	Calculate Sharpe ratio	🟠 P1	⬜	2h	BT-005
BT-027	Calculate Sortino ratio	🟡 P2	⬜	2h	BT-005
BT-028	Calculate Calmar ratio	🟢 P3	⬜	1h	BT-025, BT-021
BT-029	Generate equity curve data	🔴 P0	⬜	2h	BT-005
BT-030	Generate drawdown curve data	🟠 P1	⬜	1h	BT-025

ID	Task	Priority	Status	Effort	Dependencies
BT-031	Create performance summary report	🟠 P1	⬜	2h	BT-021 to BT-030

4.6 Validation & Testing

ID	Task	Priority	Status	Effort	Dependencies
BT-032	Implement look-ahead bias prevention	🔴 P0	⬜	2h	BT-002
BT-033	Add survivorship bias warning	🟠 P1	⬜	1h	DP-002
BT-034	Implement walk-forward validation	🟡 P2	⬜	4h	BT-002
BT-035	Create backtest reproducibility (seed control)	🟠 P1	⬜	2h	BT-002
BT-036	Add unit tests for backtest engine	🔴 P0	⬜	4h	BT-001 to BT-031

Module 5: API Endpoints

5.1 Stock Data Endpoints

ID	Task	Priority	Status	Effort	Dependencies
API-001	`GET /api/stocks` — List available OMX30 stocks	🔴 P0	✅	1h	DP-002
API-002	`GET /api/stocks/<symbol>` — Get stock details	🟠 P1	⬜	1h	DP-002
API-003	`GET /api/price-data/<symbol>` — Get historical prices	🔴 P0	✅	2h	DP-011
API-004	`GET /api/indicators/<symbol>` — Get calculated indicators	🟡 P2	⬜	2h	TI-020

5.2 Backtest Endpoints

ID	Task	Priority	Status	Effort	Dependencies
API-005	`POST /api/backtest/run` — Execute backtest	🔴 P0	✅	3h	BT-002
API-006	`GET /api/backtest/<id>` — Get backtest results	🟠 P1	⬜	2h	API-005
API-007	`GET /api/backtest/<id>/trades` — Get trade list	🔴 P0	⬜	1h	API-005

ID	Task	Priority	Status	Effort	Dependencies
API-008	`GET /api/backtest/<id>/metrics` — Get performance stats	🔴 P0	⬜	1h	BT-031
API-009	`GET /api/backtest/<id>/equity-curve` — Get equity data	🔴 P0	⬜	1h	BT-029
API-010	`DELETE /api/backtest/<id>` — Delete backtest	🟢 P3	⬜	1h	API-005

5.3 Trade Endpoints

ID	Task	Priority	Status	Effort	Dependencies
API-011	`GET /api/trades` — List all trades (with filters)	🔴 P0	✅	2h	BT-003
API-012	`GET /api/trades/<id>` — Get trade details	🟡 P2	⬜	1h	BT-003
API-013	`GET /api/trades/export` — Export trades as CSV	🟠 P1	⬜	2h	API-011

5.4 AI Endpoints

ID	Task	Priority	Status	Effort	Dependencies
API-014	`POST /api/ai/predict` — Get trade probability	🟠 P1	✅	2h	AI-005
API-015	`POST /api/ai/phase` — Detect market phase	🟡 P2	⬜	2h	AI-001
API-016	`GET /api/ai/sentiment/<symbol>` — Get sentiment score	🟢 P3	⬜	2h	AI-003

5.5 API Infrastructure

ID	Task	Priority	Status	Effort	Dependencies
API-017	Add request validation (Pydantic/Marshmallow)	🟠 P1	⬜	3h	None
API-018	Implement error handling and status codes	🔴 P0	⬜	2h	None
API-019	Add API rate limiting	🟡 P2	⬜	2h	None
API-020	Implement response caching	🟡 P2	⬜	2h	None
API-021	Add API documentation (Swagger/OpenAPI)	🟠 P1	⬜	3h	None
API-022	Implement async endpoints (for long backtests)	🟡 P2	⬜	4h	API-005

Module 6: AI Integration (Phase 2)

6.1 LLM Phase Detection

ID	Task	Priority	Status	Effort	Dependencies
AI-001	Design prompt template for phase detection	🟠 P1	⬜	3h	SG-012
AI-002	Implement LLM API integration (Claude/GPT)	🟠 P1	⬜	3h	None
AI-003	Create phase detection pipeline	🟡 P2	⬜	4h	AI-001, AI-002
AI-004	Add response parsing and validation	🟡 P2	⬜	2h	AI-003

6.2 ML Trade Filter

ID	Task	Priority	Status	Effort	Dependencies
AI-005	Design feature set for trade prediction	🟠 P1	⬜	3h	TI-020
AI-006	Implement feature engineering pipeline	🟠 P1	⬜	4h	AI-005
AI-007	Train Logistic Regression baseline model	🟠 P1	⬜	3h	AI-006
AI-008	Train Decision Tree model	🟡 P2	⬜	2h	AI-006
AI-009	Implement model evaluation (precision, recall, F1)	🟠 P1	⬜	2h	AI-007
AI-010	Create model selection framework	🟡 P2	⬜	2h	AI-007, AI-008
AI-011	Implement probability threshold optimization	🟡 P2	⬜	2h	AI-009
AI-012	Create model persistence (save/load)	🟠 P1	⬜	2h	AI-007

6.3 Sentiment Analysis

ID	Task	Priority	Status	Effort	Dependencies
AI-013	Identify news sources for Swedish stocks	🟢 P3	⬜	2h	None
AI-014	Implement news data fetcher	🟢 P3	⬜	3h	AI-013

ID	Task	Priority	Status	Effort	Dependencies
AI-015	Create sentiment analysis pipeline	🟢 P3	⬜	4h	AI-014
AI-016	Integrate sentiment as trading feature	🟢 P3	⬜	2h	AI-015, AI-006

6.4 Confidence Scoring

ID	Task	Priority	Status	Effort	Dependencies
AI-017	Design confidence score formula	🟡 P2	⬜	2h	AI-007
AI-018	Implement multi-factor confidence scoring	🟡 P2	⬜	3h	AI-017
AI-019	Add confidence-based position sizing	🟢 P3	⬜	2h	AI-018, BT-012

Sprint Planning

Sprint 1: Foundation (Week 1-2)

Goal: Basic data pipeline and indicator calculation

Task ID	Task	Owner	Status
DP-001	Set up yfinance integration		⬜
DP-002	Create OMX30 stock list		⬜
DP-003	Implement daily OHLCV fetcher		⬜
DP-007	Design database schema		⬜
DP-008	Set up database		⬜
DP-009	Implement data models		⬜
DP-010	Create data insertion functions		⬜
DP-011	Implement data retrieval queries		⬜
TI-001	Implement SMA		⬜
TI-002	Implement EMA		⬜
TI-006	Implement RSI		⬜
TI-011	Implement ATR		⬜

Deliverable: Working data pipeline that fetches and stores price data, calculates basic indicators.

Sprint 2: Signals & Core Backtest (Week 3-4)

Goal: Signal generation and basic backtesting

Task ID	Task	Owner	Status
TI-003	Configurable MA periods		⬜
TI-004	MA crossover detection		⬜
TI-015	Volume moving average		⬜
TI-016	Volume ratio		⬜
SG-001	MA pullback entry signal		⬜
SG-003	RSI oversold entry signal		⬜
SG-006	Fixed stop loss		⬜
SG-008	Fixed take profit		⬜
BT-001	BacktestEngine class		⬜
BT-002	Simulation loop		⬜
BT-003	Trade class		⬜
BT-004	Position tracking		⬜
BT-005	Portfolio class		⬜
BT-006	Market order execution		⬜

Deliverable: Basic backtesting engine that can run a simple MA pullback strategy.

Sprint 3: Risk & Metrics (Week 5-6)

Goal: Risk management and performance measurement

Task ID	Task	Owner	Status
BT-011	Fixed position sizing		⬜
BT-012	% equity position sizing		⬜
BT-017	Commission modeling		⬜
BT-021	Total return		⬜
BT-022	Win rate		⬜
BT-023	Profit factor		⬜
BT-024	Expectancy		⬜
BT-025	Maximum drawdown		⬜
BT-029	Equity curve		⬜
BT-031	Performance summary		⬜
BT-032	Look-ahead bias prevention		⬜
SG-017	Signal base class		⬜

Deliverable: Complete backtesting engine with risk management and performance metrics.

Sprint 4: API & Integration (Week 7-8)

Goal: API endpoints and frontend integration

Task ID	Task	Owner	Status
API-005	POST /backtest/run		⬜
API-007	GET /backtest/trades		⬜
API-008	GET /backtest/metrics		⬜
API-009	GET /backtest/equity-curve		⬜
API-013	Trade CSV export		⬜
API-017	Request validation		⬜
API-018	Error handling		⬜
API-021	API documentation		⬜
DP-013	Data validation		⬜
DP-014	Handle missing data		⬜

Deliverable: Production-ready API that powers the frontend dashboard.

Sprint 5: AI Integration (Week 9-10)

Goal: ML-based trade filter

Task ID	Task	Owner	Status
AI-005	Feature set design		⬜
AI-006	Feature engineering pipeline		⬜
AI-007	Logistic Regression model		⬜
AI-009	Model evaluation		⬜
AI-012	Model persistence		⬜
API-014	POST /ai/predict		⬜
SG-005	Composite entry signal		⬜
SG-018	Signal combiner		⬜

Deliverable: AI-powered trade filter integrated with backtesting engine.

Progress Tracking

Module Completion

Module	Total Tasks	Complete	Progress
Data Pipeline	17	0	0%
Technical Indicators	23	0	0%
Signal Generation	20	0	0%
Backtesting Engine	36	0	0%
API Endpoints	22	5	23%
AI Integration	19	0	0%
Total	**137**	**5**	**4%**

Notes

Technical Decisions

- [ ] Database: SQLite (dev) vs PostgreSQL (prod)?
- [ ] ORM: SQLAlchemy vs Peewee vs raw SQL?
- [ ] Async framework: Flask vs FastAPI?
- [ ] ML framework: scikit-learn vs XGBoost?

Blockers & Risks

1. Data availability for delisted Swedish stocks
2. API rate limits on free data sources
3. Model overfitting on limited historical data

Resources

- yfinance Documentation
- TA-Lib Python
- Backtrader Framework
- Zipline (Quantopian)

Last Updated: January 2025