Stock Market Predictor App: Expert Development Roadmap & Feasibility Analysis

Executive Summary & Feasibility Assessment

Project Complexity Rating: 7/10 (Medium-High)
Success Probability: 70% for MVP, 40% for full-featured version
Recommended Approach: Focus on MVP first, then iterate

Your project idea is **ambitious but achievable** within 6 weeks if properly scoped. Based on my analysis of current technologies and your skill level, I recommend building a **Minimum Viable Product (MVP)** first, then expanding features incrementally.

Core Technology Stack Analysis

Backend & APIs

- **Web Framework**: Flask (lightweight, Python-friendly)[1] [2] [3]
- **Database**: SQLite for development, PostgreSQL for production[2] [3]
- **Stock Data**: yfinance (free), Alpha Vantage API (free tier: 500 calls/day)[4] [5] [6]
- **News Data**: NewsAPI (free tier: 1000 calls/day), Finnhub (free tier: 60 calls/minute)[7] [8]
- **ML Framework**: TensorFlow/Keras for LSTM models[9] [10] [11]

Cross-Platform GUI Options

Based on your Windows/Mac collaboration requirement:

- **Recommended: Flet** - Python-based, Flutter-powered, excellent for beginners[12]
- **Alternative: Kivy** - More mature but steeper learning curve[13] [14]
- **Advanced Option: Flask web app** - Accessible via browser on both platforms[15] [2]

Sentiment Analysis

- **VADER**: Superior for social media content, handles emojis and caps[16] [17]
- **TextBlob**: Simpler implementation, good for news articles[17] [16]

Detailed 6-Week Development Roadmap

Week	Tasks	Person A (Windows)	Person B (Mac)	Collaboration Tools
Week 1	Setup & Authentication	Flask app setup, user registration	Database design, login system	GitHub, shared repository [15] [18]
Week 2	Data Integration	Stock API integration (yfinance)	News API setup (NewsAPI)	API testing, data validation
Week 3	Sentiment Analysis	VADER sentiment implementation	News data processing pipeline	Code reviews, testing
Week 4	ML Model Development	LSTM model architecture	Data preprocessing, training	Model sharing via GitHub
Week 5	GUI & Integration	Flet/Kivy GUI development	Dashboard components	Cross-platform testing
Week 6	Testing & Deployment	Bug fixes, optimization	Documentation, deployment	Final integration testing

Required Learning Path & Time Investment

Week 1 Preparation (Before Starting)

- **Flask Fundamentals**: 15-20 hours[3] [2]
- **API Integration Basics**: 10 hours[5] [4]
- **Git/GitHub Collaboration**: 5 hours[18] [15]

Technical Skill Requirements by Week

1. **Flask + SQLite Authentication**: Medium difficulty[2] [3]
2. **API Integration**: Easy-Medium[6] [4] [5]
3. **Sentiment Analysis**: Medium[16] [17]
4. **LSTM Implementation**: High difficulty[19] [11] [9]
5. **Cross-platform GUI**: Medium-High[13] [12]

Recommended MVP Feature Set

Phase 1: Core MVP (Weeks 1-4)

✅ User authentication with SQLite database
✅ Basic stock dashboard with real-time prices
✅ Simple news sentiment analysis (VADER)
✅ Basic prediction display (linear regression first)
✅ Web-based interface (accessible on both platforms)

Phase 2: Advanced Features (Post-MVP)

⭒ LSTM neural network implementation
⭒ Advanced sentiment quantification
⭒ Native desktop app (Flet/Kivy)
⭒ Real-time WebSocket updates
⭒ Portfolio tracking features

Critical Risk Mitigation Strategies

Technical Risks & Solutions

- **LSTM Complexity**: Start with simpler linear regression, upgrade later[20] [21]
- **API Rate Limits**: Implement caching, use multiple free APIs[4] [5]
- **Cross-platform Issues**: Use web-based approach initially, then native apps[12]
- **Data Quality**: Focus on major stocks (AAPL, MSFT, GOOGL) for consistency[22] [19]

Project Management Recommendations

- **Daily standups**: 15-minute progress sync
- **Weekly sprint reviews**: Assess progress and adjust scope
- **Version control**: Branching strategy for parallel development[15] [18]
- **Testing strategy**: Continuous testing throughout development

Essential Libraries & Dependencies

```
# Core requirements.txt
flask==2.3.3
flask-sqlalchemy==3.0.5
flask-login==0.6.3
pandas==2.0.3
numpy==1.24.3
yfinance==0.2.18
requests==2.31.0
vaderSentiment==3.3.2
textblob==0.17.1
tensorflow==2.13.0
scikit-learn==1.3.0
flet==0.10.3  # For cross-platform GUI
```

Collaboration Workflow Setup

Repository Structure

```
stock-predictor-app/
├── backend/
│   ├── app.py (Flask main)
│   ├── models.py (Database)
│   ├── api_handlers.py (Stock/News APIs)
│   └── ml_models.py (LSTM/Predictions)
├── frontend/
│   ├── gui_main.py (Flet/Kivy app)
│   └── templates/ (Web templates)
├── data/
├── tests/
└── docs/
```

Development Environment Setup

1. **Shared Development**: Use GitHub Codespaces or similar[15]
2. **Local Setup**: Virtual environments on both Windows/Mac
3. **API Keys Management**: Use environment variables, shared .env files
4. **Database Sync**: SQLite for development, shared via Git

Success Metrics & Milestones

Week-by-Week Success Criteria

- **Week 1**: Working authentication system
- **Week 2**: Real-time stock data display
- **Week 3**: Basic sentiment scores from news
- **Week 4**: Simple prediction model working
- **Week 5**: Cross-platform app launched
- **Week 6**: End-to-end system functional

MVP Definition of "Done"

✅ Users can register/login securely
✅ Dashboard shows favorite stocks with real prices
✅ News sentiment affects prediction scores
✅ Basic future price predictions displayed
✅ App works on both Windows and Mac

Final Recommendation

This project is absolutely feasible if you focus on building an MVP first. Your A-Level CS background provides a solid foundation, and the 6-week timeline is realistic for a functional prototype.[23] [24]

Key Success Factors:

1. Start with web-based interface (easier cross-platform)
2. Use pre-trained sentiment analysis (VADER/TextBlob) initially[17] [16]
3. Begin with simple ML models before attempting LSTM[21] [20]
4. Implement continuous integration from day one[18] [15]
5. Plan for post-project iterations and improvements

The project will provide excellent learning opportunities in full-stack development, API integration, machine learning, and collaborative software development - all valuable skills for your computer science degree and future career.

⁂

now give me an exact step by step roadmap on how i should learn everything needed for the development of the app. so i have decided to use pyqt 6 as the cross platform GUI option and i will not be using any web framework as this will be a desktop app for both mac and windows. i also do not know how to use APIs or have any knowledge on machine learning frameworks. so using this information and also including any python libraries and other things i need to learn create a detailed step by step roadmap in order.

Here is a focused, step-by-step learning roadmap to build a cross-platform desktop app with PyQt6, stock/news APIs, sentiment analysis, and an initial ML predictor within six weeks.[119] [120] [121]

What to learn first

- PyQt6 fundamentals (widgets, layouts, signals/slots) to build a responsive desktop UI.[120] [122] [119]
- Using Python requests to call REST APIs and handle JSON responses.[121]

- Stock data APIs (Alpha Vantage for intraday/historical, yfinance for quick Yahoo data).[123] [124]
- A news API for articles (e.g., NewsData.io) to power sentiment analysis.[125]
- VADER sentiment analysis for document/article polarity scoring.[126]
- SQLite with Python for user logins and app data.[127]
- An entry-level TensorFlow LSTM time-series workflow for price forecasting.[128]
- Packaging a PyQt6 desktop app with PyInstaller for macOS/Windows.[129]

Step-by-step roadmap (in order)

1. Install prerequisites and set up a fresh virtual environment; confirm PyQt6, requests, and basic plotting run locally.[119] [121]
2. Learn PyQt6 signals/slots and events (core UI interaction model) with a simple window that reacts to button clicks and updates labels.[122] [120] [119]
3. Build a minimal PyQt6 app skeleton: a QMainWindow, central QWidget, vertical layout, and a status bar to show app messages.[122] [119]
4. Add a placeholder Dashboard tab and a Settings/Login tab in a tabbed interface so screens are separated early.[119] [122]
5. Learn Python API basics with requests: HTTP GETs, parameters, headers, JSON parsing, error handling (status codes), and timeouts.[121]
6. Register for a stock API key and test Alpha Vantage endpoints in a small script to fetch intraday or daily time series JSON/CSV for a symbol.[130] [123]
7. Use yfinance to quickly pull historical candles and metadata for several tickers to accelerate early UI demos.[124]
8. Design a small “Data layer” module: one class for Alpha Vantage calls (key, URL, function, symbol) and one for yfinance historical pulls.[123] [124]
9. Learn SQLite with Python: create a database file, define a users table, and implement insert/select/update queries with sqlite3.[127]
10. Implement a Login/Register dialog in PyQt6 that talks to SQLite, and store minimal user profile fields for favorites and preferences.[127] [119]
11. Build the Dashboard: a ticker search input, a table or list of favorites, and a simple price history plot using data fetched by the Data layer.[124] [119]
12. Learn a news API and test queries for company names/tickers to get recent articles with titles, descriptions, and publish times.[125]
13. Install and test VADER in a small script to compute pos/neu/neg/compound scores for sample headlines and short article summaries.[126]
14. Implement a News pipeline: fetch N latest articles for a symbol, score each with VADER, and aggregate into daily averages and counts.[125] [126]
15. Add a Sentiment panel in the UI that shows the distribution of pos/neu/neg and a rolling compound score for the selected symbol.[126] [119]

16. Learn the basics of TensorFlow LSTM for time-series: framing sequences, building a small Sequential LSTM model, and predicting the next step.[128]
17. Prepare price history into supervised sequences, split train/test chronologically, build a small LSTM (1–2 layers), and train/evaluate quickly.[128]
18. Display the LSTM projection alongside the latest actual prices on the Dashboard to visualize forecast versus reality for the selected symbol.[124] [128]
19. Wire periodic refresh: schedule API pulls on a timer and use signals/slots to update the UI without freezing the main thread.[120] [119]
20. Package the desktop app for each OS with PyInstaller, producing a macOS .app and Windows executable for easy distribution.[129]

Suggested weekly plan

- Week 1: PyQt6 fundamentals, UI skeleton, signals/slots, and basic event handling demos. [120] [122] [119]
- Week 2: API basics with requests, Alpha Vantage key and calls, yfinance historical pulls, and Dashboard data display.[121] [123] [124]
- Week 3: SQLite user system with a login/register dialog and storing favorites, then integrate favorites into the Dashboard.[119] [127]
- Week 4: News API integration, VADER sentiment scoring pipeline, and a sentiment panel with aggregated scores.[125] [126]
- Week 5: TensorFlow LSTM workflow on historical prices and plotting predicted vs actual in the app.[124] [128]
- Week 6: Periodic refresh, polish UX, and PyInstaller packaging for macOS/Windows builds. [129] [120] [119]

Key Python libraries to install

- PyQt6 for cross-platform desktop GUI.[120] [119]
- requests for REST API calls.[121]
- yfinance for Yahoo Finance market data.[124]
- alpha-vantage or raw Alpha Vantage endpoints for intraday/historical.[131] [123]
- vaderSentiment for news sentiment scoring.[126]
- tensorflow for LSTM time-series modeling.[128]
- sqlite3 from the standard library for data persistence.[127]
- PyInstaller for packaging the PyQt6 app.[129]

Practical milestones and checkpoints

- Milestone A: A working PyQt6 shell with tabs and reactive widgets confirms UI knowledge and event handling.[122] [119]
- Milestone B: Stock data loads for a few symbols and renders in the UI from yfinance and Alpha Vantage calls.[123] [124]
- Milestone C: Users can register and sign in, and their favorites persist to SQLite and appear on the Dashboard.[127]
- Milestone D: The app fetches articles for a ticker, computes VADER scores, and shows daily sentiment aggregates.[125] [126]
- Milestone E: An LSTM model is trained on historical prices and a simple forecast plot appears in the UI.[128] [124]
- Milestone F: The app runs timed refreshes without freezing the UI and produces packaged binaries for macOS/Windows.[120] [129]

Study tips and scope control

- Prefer yfinance for early prototypes and add Alpha Vantage when needing intraday or more structured endpoints.[123] [124]
- Keep the first LSTM small and well-regularized; focus on data preparation and stable train/test splits.[128]
- Start VADER on headlines and descriptions before attempting longer article bodies to keep latency low.[126] [125]

Minimal implementation order inside the app

- UI shell → Stock data panel → Favorites/SQLite → News fetch → VADER aggregation panel → LSTM forecast view → timers → packaging.[119] [129] [120] [125] [124] [126] [127] [128]

If this sequence is followed with daily coding and testing, a functional MVP with PyQt6, working stock/news retrieval, aggregated sentiment, and a basic LSTM forecast is realistic within six weeks.[123] [119] [124]

⁂

okay this is good, now building upon this give me a nice neat table and a list containing everything i need to learn (libraries, frameworks, applications, technologies, etc...) with the columns: name, learning time, good resources to learn, what its used for.

Below is a concise table of what to learn, with estimated time, solid learning resources, and what each item is used for, followed by a prioritized list to guide the order of study.[212] [213] [214]

Learning table

Name	Learning time	Good resources to learn	What it's used for
PyQt6 fundamentals	10–15 hours [212]	PyQt6 tutorials on pythonguis; Qt for Python docs [212] [213]	Building cross-platform desktop UIs (windows, layouts, widgets) [213]
Signals/slots & events	6–8 hours [212]	Qt signals/slots docs; PyQt6 signals/slots tutorials [213] [215]	Reactive UI logic, connecting buttons/inputs to handlers [213]
PyQt6 plotting (Matplotlib)	4–6 hours [216]	Embedding Matplotlib in PyQt6 (pythonguis); Matplotlib Embed in Qt docs [216] [217]	Rendering price charts and forecasts inside the app [217]
PyQtGraph (optional)	3–5 hours [218]	PyQtGraph plotting tutorial on pythonguis [218]	Fast interactive plots as an alternative to Matplotlib [218]
requests + JSON basics	4–6 hours [214]	"How to Use an API in Python" (Dataquest) [214]	Calling REST APIs, handling parameters, headers, timeouts, JSON [214]
Alpha Vantage API	3–5 hours [219]	Alpha Vantage documentation; Python client docs [219] [220]	Intraday/historical stock data via API key for symbols [219]
yfinance	2–4 hours [221]	yfinance documentation [221]	Quick historical market data from Yahoo for charts and training [221]
News API (NewsData.io)	2–4 hours [222]	NewsData.io documentation [222]	Fetching company news articles for sentiment analysis [222]
VADER sentiment	2–4 hours [223]	VADER GitHub repository and examples [223]	Scoring headlines/descriptions with pos/neg/compound sentiment [223]
SQLite (sqlite3)	4–6 hours [224]	SQLite and Python tutorial [224]	Local user accounts, favorites, and cached data storage [224]
TensorFlow LSTM (basics)	12–20 hours [225]	TensorFlow LSTM beginner's guide [225]	Time-series modeling for basic next-step stock forecasts [225]
PyInstaller packaging	4–6 hours [226]	Packaging PyQt6 apps for macOS/Windows (pythonguis) [226]	Creating distributable app bundles/executables for both OSes [226]

Name	Learning time	Good resources to learn	What it’s used for
Git & GitHub	4–6 hours [227]	GitHub beginner tutorial (GitHub Desktop or CLI) [227]	Version control, collaboration, branching, and code reviews [227]
Timers/threads in PyQt6	3–5 hours [217]	Matplotlib Qt example (timer usage); Qt signals/slots docs [217] [213]	Periodic refresh of data without freezing the UI [217]

Prioritized list (study order)

- PyQt6 fundamentals: windows, layouts, widgets, menus, and basic event handling to establish the app shell quickly.[213] [212]
- Signals/slots patterns: connect UI actions to application logic reliably for a responsive desktop experience.[215] [213]
- requests + JSON: learn HTTP GETs, query params, headers, timeouts, and parsing JSON into Python dicts/lists.[214]
- Stock data sources: start with yfinance for quick historical data, then add Alpha Vantage for intraday endpoints.[219] [221]
- News API: register an API key and fetch recent articles for company names/tickers to feed the sentiment layer.[222]
- VADER sentiment: score titles/descriptions and compute daily aggregates for the sentiment dashboard.[223]
- SQLite: build user login/register and store favorites and cached pulls to reduce API calls and enable persistence.[224]
- PyQt6 plotting: embed Matplotlib (or PyQtGraph) charts for prices and model forecasts in the desktop UI.[216] [217]
- TensorFlow LSTM basics: prepare sequences, build a small LSTM, and plot predicted vs actual in the app.[225]
- Timers/threads in PyQt6: add periodic data refresh via timers and avoid blocking the main event loop.[217] [213]
- Packaging: generate macOS app bundle and Windows executable with PyInstaller for easy distribution.[226]
- Git & GitHub: use a shared repo, branches per feature, and PRs for reviews to collaborate efficiently.[227]

⁂

1. https://ijsrem.com/download/stock-market-prediction-compliance-using-ai-and-ml/
2. https://www.geeksforgeeks.org/python/how-to-add-authentication-to-your-app-with-flask-login/
3. https://www.digitalocean.com/community/tutorials/how-to-add-authentication-to-your-app-with-flask-login
4. https://dev.to/williamsmithh/top-5-free-financial-data-apis-for-building-a-powerful-stock-portfolio-tracker-4dhj

5. https://marketstack.com
6. https://site.financialmodelingprep.com/developer/docs
7. https://finlight.me/blog/news-apis-for-developers-in-2025
8. https://www.intelligenthq.com/best-5-news-data-apis-in-2025/
9. https://archives.journal-grail.science/index.php/2710-3056/article/view/13
10. https://www.semanticscholar.org/paper/56ea5fee05545ca37abcc20314700e6d6927ce68
11. https://www.geeksforgeeks.org/nlp/stock-price-prediction-project-using-tensorflow/
12. https://flet.dev
13. https://www.linkedin.com/pulse/cross-platform-mobile-app-development-python-abdullah-shakir-cp4af
14. https://dev.to/mr_nova/flutter-vs-react-native-vs-kivy-best-framework-for-lightweight-app-development-5dbc
15. https://www.linkedin.com/pulse/continuously-deploying-flask-app-from-github-repository-eurico-paes-3ymlf
16. https://www.youtube.com/watch?v=V858y9L_RJM
17. https://spotintelligence.com/2022/12/16/sentiment-analysis-tools-in-python/
18. https://dev.to/sudo_anuj/automating-flask-app-deployment-with-docker-github-actions-8gh
19. https://bcpublication.org/index.php/BM/article/view/3109
20. https://www.geeksforgeeks.org/machine-learning/stock-price-prediction-using-machine-learning-in-python/
21. https://ijircst.org/DOC/66-stock-price-prediction-using-python-in-machine-learning.pdf
22. https://github.com/Py-Fi-nance/Stock-Price-Forecasting-with-Machine-Learning
23. https://appcost.ai/blog/software-development-timeline-guide
24. https://monday.com/blog/project-management/project-timeline/
25. https://publications.eai.eu/index.php/sis/article/download/4446/2704
26. https://drpress.org/ojs/index.php/fcis/article/download/10208/9933
27. https://www.mdpi.com/2227-7390/11/3/590/pdf?version=1675665621
28. https://www.ccsenet.org/journal/index.php/ijef/article/download/75705/42155
29. https://www.youtube.com/watch?v=94PIBzgeq90
30. https://drlee.io/advanced-stock-pattern-prediction-using-lstm-with-the-attention-mechanism-in-tensorflow-a-step-by-143a2e8b0e95
31. https://dev.to/adeleke123/from-flask-app-to-cicd-pipeline-with-github-actions-docker-hub-aoa
32. https://cswsolutions.com/blog/posts/2023/september/formulating-a-realistic-software-development-timeline-example/
33. https://www.youtube.com/watch?v=CbTU92pbDKw
34. https://www.honeybadger.io/blog/flask-github-actions-continuous-delivery/
35. https://kvytechnology.com/blog/software/estimate-development-time/
36. https://www.tensorflow.org/tutorials/structured_data/time_series
37. https://github.com/orgs/community/discussions/68841
38. https://stepmediasoftware.com/blog/shape-up-methodology/

39. https://www.machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/
40. https://github.com/pallets/flask
41. https://www.fuselio.com/blog/the-6-week-sme-software-development-mvp-journey-from-planning-to-post-launch-iteration
42. https://www.kaggle.com/code/faressayah/stock-market-analysis-prediction-using-lstm
43. https://stackoverflow.com/questions/77883901/how-to-use-flask-python-in-github-actions-to-serve-a-web-app
44. https://ijarsct.co.in/Paper22534.pdf
45. https://ieeexplore.ieee.org/document/10031901/
46. https://www.ewadirect.com/proceedings/aemps/article/view/18814
47. https://www.jsr.org/hs/index.php/path/article/view/6070
48. https://al-kindipublisher.com/index.php/jbms/article/view/8609
49. https://www.mdpi.com/2673-4591/56/1/254
50. https://www.mdpi.com/2227-7390/11/5/1130
51. https://ijai.iaescore.com/index.php/IJAI/article/view/23989
52. https://bcpublication.org/index.php/BM/article/download/3098/3051
53. https://www.techscience.com/iasc/v37n2/53230
54. https://www.mdpi.com/2297-8747/25/3/53/pdf
55. http://www.hrpub.org/download/20210630/UJAF15-12222396.pdf
56. https://arxiv.org/ftp/arxiv/papers/2004/2004.01497.pdf
57. https://www.techrxiv.org/doi/full/10.36227/techrxiv.16640197.v1
58. https://www.ijfmr.com/papers/2024/1/12383.pdf
59. https://dx.plos.org/10.1371/journal.pone.0319775
60. https://www.preprints.org/manuscript/202003.0256/v1/download
61. https://annals-csis.org/proceedings/rice2022/drp/pdf/02.pdf
62. https://www.youtube.com/watch?v=1O_BenficgE
63. https://sproutsocial.com/insights/sentiment-analysis-tools/
64. https://www.tigerdata.com/learn/time-series-analysis-and-forecasting-with-python
65. https://www.reddit.com/r/webdev/comments/151zk8y/is_there_any_free_stock_market_api_that_allows/
66. https://assemblyai.com/blog/best-apis-for-sentiment-analysis
67. https://python.plainenglish.io/how-i-built-an-ai-model-that-trades-the-stock-market-while-i-sleep-4501f338d89f
68. https://finnhub.io
69. https://newsapi.ai/blog/best-news-api-comparison-2025
70. https://www.sciencedirect.com/science/article/pii/S2590291124000615
71. https://www.alphavantage.co
72. https://www.edenai.co/post/best-sentiment-analysis-apis
73. https://polygon.io

74. https://finnhub.io/docs/api/news-sentiment
75. https://goldncloudpublications.com/index.php/irjaem/article/view/614
76. https://ijies.net/final-docs/final-pdf/1065.pdf
77. https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0011824500003476
78. https://www.semanticscholar.org/paper/f75662aa67e6c2ffa1337368ebf1be6924cdd7c1
79. https://dl.acm.org/doi/10.1145/3442555.3442561
80. https://www.ijrte.org/portfolio-item/A75800512123/
81. https://resources.today/en/13INOR420.html
82. https://www.semanticscholar.org/paper/02f9c4923bd7563b363ed4dbc367c9445db57b97
83. https://ieeexplore.ieee.org/document/10667693/
84. https://www.ijraset.com/best-journal/the-polyglots-playground-navigating-kmp-flutter-and-react-native-in-cross-platform-ecosystems
85. https://www.ej-eng.org/index.php/ejeng/article/download/2740/1221
86. https://dergipark.org.tr/en/download/article-file/1419794
87. https://www.mdpi.com/2078-2489/15/10/614
88. https://www.e3s-conferences.org/10.1051/e3sconf/202560100022
89. https://ejournal.bsi.ac.id/ejurnal/index.php/ijcit/article/download/9680/pdf
90. https://www.techscience.com/ueditor/files/csse/TSP_CSSE-42-2/TSP_CSSE_22519/TSP_CSSE_22519.pdf
91. https://www.mdpi.com/1424-8220/21/10/3324/pdf
92. https://www.matec-conferences.org/articles/matecconf/pdf/2018/91/matecconf_eitce2018_03044.pdf
93. https://arxiv.org/abs/2409.11667
94. https://ejournal.undiksha.ac.id/index.php/janapati/article/download/60629/26975
95. https://www.freecodecamp.org/news/how-to-setup-user-authentication-in-flask/
96. https://www.netguru.com/blog/python-sentiment-analysis-libraries
97. https://kivy.org
98. https://flask.palletsprojects.com/en/stable/tutorial/database/
99. https://neptune.ai/blog/sentiment-analysis-python-textblob-vs-vader-vs-flair
100. https://codewave.com/insights/cross-platform-app-development-languages-overview/
101. https://stackoverflow.com/questions/41323397/flask-login-with-sqlite3
102. https://www.bairesdev.com/blog/best-python-sentiment-analysis-libraries/
103. https://www.yesitlabs.com/top-programming-languages-for-mobile-app-development-in-2024/
104. https://www.youtube.com/watch?v=gbTLL6htiPg
105. https://www.ishir.com/blog/122160/the-best-8-python-js-frameworks-for-mobile-development-how-to-use-them.htm
106. https://gist.github.com/jironghuang/24e0577e58844882604c0013407bf606
107. https://www.reddit.com/r/flask/comments/5ldblw/help_with_flasklogin_and_sqlite3/
108. https://ieeexplore.ieee.org/document/11034828/
109. https://ieeexplore.ieee.org/document/9596513/

110. https://bcpublication.org/index.php/BM/article/view/4398
111. https://www.matec-conferences.org/10.1051/matecconf/202338101017
112. https://www.irjmets.com/uploadedfiles/paper//issue_8_august_2023/43978/final/fin_irjmets1693926535.pdf
113. https://www.ewadirect.com/proceedings/ace/article/view/11083
114. https://ijsrem.com/download/stock-market-price-prediction-using-machine-learning-and-deep-learning/
115. http://www.ijcte.org/vol12/1267-G1758.pdf
116. https://aemps.ewapublishing.org/media/9bd1ed43a76d422c97c69dcca2dc3b4e.marked.pdf
117. https://www.ijert.org/research/stock-market-analysis-using-lstm-in-deep-learning-IJERTV9IS040649.pdf
118. https://bcpublication.org/index.php/BM/article/download/3109/3062
119. https://www.pythonguis.com/tutorials/pyqt6-signals-slots-events/
120. https://doc.qt.io/qtforpython-6/tutorials/basictutorial/signals_and_slots.html
121. https://www.dataquest.io/blog/api-in-python/
122. https://zetcode.com/pyqt6/eventssignals/
123. https://www.alphavantage.co/documentation/
124. https://ranaroussi.github.io/yfinance/
125. https://newsdata.io/documentation
126. https://github.com/cjhutto/vaderSentiment
127. https://www.sqlitetutorial.net/sqlite-python/
128. https://pieriantraining.com/tensorflow-lstm-example-a-beginners-guide/
129. https://www.pythonguis.com/tutorials/packaging-pyqt6-applications-pyinstaller-macos-dmg/
130. https://www.alphavantage.co
131. https://alpha-vantage.readthedocs.io/en/latest/
132. https://dl.acm.org/doi/10.1145/1024916.1024920
133. https://www.semanticscholar.org/paper/b530ba091fc4b85e42e24fc9cafa9a439ddf5c11
134. http://link.springer.com/10.1007/978-981-10-1389-8_6-1
135. https://www.semanticscholar.org/paper/f335d2fd173e97d7bc711f4f78c4e2fa2ff0f57a
136. https://www.semanticscholar.org/paper/3721703ba3774e5a7454a189e6bfd1502b4fbeae
137. https://www.semanticscholar.org/paper/7384fd641b66825e4688a0ef528cefec187eb7c5
138. http://peer.asee.org/19631
139. https://www.semanticscholar.org/paper/fa7c3df2a463bc97110c55dded07febc1dfc699c
140. https://www.semanticscholar.org/paper/e32ac8c232c9f7d0f011d992c492193772321f79
141. https://www.semanticscholar.org/paper/947a7e2d9339763b7efa5a48a9b1210709c40520
142. https://www.pythonguis.com/tutorials/pyqt6-transmitting-extra-data-qt-signals/
143. https://coderslegacy.com/python/pyqt6-signals-and-slots/
144. https://www.youtube.com/watch?v=N2JfygnWJaA
145. https://www.pythonguis.com/tutorials/packaging-pyqt6-applications-windows-pyinstaller/

146. https://realpython.com/python-requests/
147. https://www.youtube.com/watch?v=upvU3d6NAvs
148. https://www.reddit.com/r/QtFramework/comments/1cu3676/whats_the_easiest_way_to_distribute_a_pyqt6_app/
149. https://requests.readthedocs.io/en/master/user/quickstart/
150. https://stackoverflow.com/questions/72213191/pyqt6-tutorial-how-to-receiving-signal-parameters
151. https://forum.qt.io/topic/156731/what-s-the-easiest-way-to-distribute-a-pyqt6-app-for-multiple-os
152. https://www.youtube.com/watch?v=JVQNywo4AbU
153. https://www.tutorialspoint.com/pyqt/pyqt_signals_and_slots.htm
154. https://github.com/pyinstaller/pyinstaller/issues/7805
155. https://www.w3schools.com/python/module_requests.asp
156. https://www.pythontutorial.net/pyqt/pyqt-signals-slots/
157. https://www.semanticscholar.org/paper/aa8348126710a3667f5179b814446a7ed36e10f6
158. https://journal.uny.ac.id/v3/jited/article/view/47
159. https://ieeexplore.ieee.org/document/10739272/
160. https://ieeexplore.ieee.org/document/11019476/
161. https://ieeexplore.ieee.org/document/11042718/
162. https://ieeexplore.ieee.org/document/11011825/
163. https://www.ijraset.com/best-journal/visualising-and-forecasting-stock-prices-with-flask
164. https://www.semanticscholar.org/paper/56ea5fee05545ca37abcc20314700e6d6927ce68
165. https://zenodo.org/record/260038
166. https://www.semanticscholar.org/paper/a7647d309e7c36f1ea009bd1d2e8895181bd5bfa
167. https://www.alphavantage.co/google_sheet_api_function_documentation/
168. https://www.postman.com/api-evangelist/blockchain/documentation/j4n0jl2/alpha-vantage
169. https://cran.r-project.org/web/packages/alphavantager/alphavantager.pdf
170. https://newsapi.org/docs
171. https://algotrading101.com/learn/yfinance-guide/
172. https://www.postman.com/api-evangelist/alpha-vantage/documentation/vx2c5fd/alpha-vantage
173. https://www.thenewsapi.com/documentation
174. https://ranaroussi.github.io/yfinance/reference/index.html
175. https://www.alphavantage.co/support/
176. https://newsapi.ai/documentation?tab=introduction
177. https://pypi.org/project/yfinance/
178. https://hexdocs.pm/alpha_vantage/api-reference.html
179. https://newsapi.org
180. https://python-yahoofinance.readthedocs.io/en/latest/api.html
181. https://apitracker.io/a/alphavantage-co
182. http://link.springer.com/10.1007/978-3-030-72240-1_56
183. https://www.worldscientific.com/doi/abs/10.1142/9789811215094_0005

184. https://doiserbia.nb.rs/Article.aspx?ID=1820-02142400040K
185. https://www.semanticscholar.org/paper/0f4eba3d1785aac3427299a8e775342d2247485f
186. https://www.ijraset.com/best-journal/integrating-textblob-and-vader-for-dynamic-sentiment-analysis-a-guibased-approach-with-emotion-visualization-and-confidence-assessment
187. https://ojs.bonviewpress.com/index.php/jdsis/article/view/2441
188. https://journal.upgris.ac.id/index.php/asset/article/view/14897
189. https://dl.acm.org/doi/10.1145/3638584.3638675
190. https://ieeexplore.ieee.org/document/10933201/
191. https://www.ijainn.latticescipub.com/portfolio-item/b39181212222/
192. https://www.mdpi.com/2227-9709/11/2/24/pdf?version=1713874475
193. https://arxiv.org/pdf/2307.14311.pdf
194. https://shmpublisher.com/index.php/joscex/article/download/193/139
195. https://github.com/rock3125/vaderSentiment
196. https://github.com/BlueFlame202/darth-vader-py
197. https://github.com/topics/vader-sentiment-analysis
198. https://github.com/greymattersblog/VADERSentiment
199. https://github.com/PetrKorab/FinVADER
200. https://www.tensorflow.org/tutorials/structured_data/time_series
201. https://docs.python.org/3/library/sqlite3.html
202. https://pypi.org/project/vaderSentiment/
203. https://www.youtube.com/watch?v=94PIBzgeq90
204. https://www.geeksforgeeks.org/python/python-sqlite/
205. https://github.com/topics/vader-sentiment-analysis?l=python&o=desc&s=updated
206. https://www.geeksforgeeks.org/deep-learning/time-series-forecasting-using-tensorflow/
207. https://www.tutorialspoint.com/sqlite/sqlite_python.htm
208. https://gist.github.com/e041443c8e5ce98712d33b61f4ace44a
209. https://www.machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/
210. https://www.youtube.com/watch?v=jsX99U8UkOo
211. https://vadersentiment.readthedocs.io/en/latest/
212. https://www.pythonguis.com/tutorials/pyqt6-signals-slots-events/
213. https://doc.qt.io/qtforpython-6/tutorials/basictutorial/signals_and_slots.html
214. https://www.dataquest.io/blog/api-in-python/
215. https://zetcode.com/pyqt6/eventssignals/
216. https://www.pythonguis.com/tutorials/pyqt6-plotting-matplotlib/
217. https://matplotlib.org/stable/gallery/user_interfaces/embedding_in_qt_sgskip.html
218. https://www.pythonguis.com/tutorials/pyqt6-plotting-pyqtgraph/
219. https://www.alphavantage.co/documentation/
220. https://alpha-vantage.readthedocs.io/en/latest/

221. https://ranaroussi.github.io/yfinance/
222. https://newsdata.io/documentation
223. https://github.com/cjhutto/vaderSentiment
224. https://www.sqlitetutorial.net/sqlite-python/
225. https://pieriantraining.com/tensorflow-lstm-example-a-beginners-guide/
226. https://www.pythonguis.com/tutorials/packaging-pyqt6-applications-pyinstaller-macos-dmg/
227. https://www.youtube.com/watch?v=v5gnvDUWqFM
228. https://www.youtube.com/watch?v=AHhcwFPQIfQ
229. https://coderslegacy.com/embed-matplotlib-graphs-into-pyqt6/
230. https://stackoverflow.com/questions/12459811/how-to-embed-matplotlib-in-pyqt
231. https://www.pythontutorial.net/pyqt/pyqt-to-exe/
232. https://www.pythonguis.com/tutorials/packaging-pyqt6-applications-windows-pyinstaller/
233. https://docs.github.com/get-started/quickstart/hello-world
234. https://matplotlib.org/stable/gallery/user_interfaces/index.html
235. https://stackoverflow.com/questions/66286229/create-pyqt6-python-project-executable
236. https://www.youtube.com/watch?v=vA5TTz6BXhY
237. https://www.reddit.com/r/Python/comments/141f88f/rendering_matplotlib_in_pyqt_6/
238. https://www.youtube.com/watch?v=U4zICx-emzc
239. https://techblog.flaviusdinu.com/are-you-still-searching-for-a-git-and-github-tutorial-in-2025-6353b6dd4bbd
240. https://doc.qt.io/qtforpython-6/examples/example_external_matplotlib_widget3d.html
241. https://doc.qt.io/qtforpython-6/deployment/deployment-pyinstaller.html
242. https://www.youtube.com/watch?v=a9u2yZvsqHA
243. https://www.geeksforgeeks.org/python/how-to-embed-matplotlib-graph-in-pyqt5/