**AI-Augmented Assignment Management: A Hybrid RL-LLM Framework for Personalized Scheduling and Content Generation in Education**

|  |  |  |
| --- | --- | --- |
| Yashwardhan Khanna | Ishan Manglik | Dr. A. Revathi |
| Department of Computational Intelligence | Department of Computational Intelligence | Department of Computational Intelligence |
| SRMIST,KTR,Chennai | SRMIST,KTR,Chennai | SRMIST,KTR,Chennai |
| yd7152@srmist.edu.in | in2832@srmist.edu.in | revathia1@srmist.edu.in |

***Abstract*—** Blending reinforcement learning (RL) and large language models (LLMs), this work introduces an AI-based framework for adaptive assignment scheduling and personalized task generation for learning environments. Optimizing assignment deadlines on the fly, the system employs a Proximal Policy Optimization (PPO) agent to consider student workload patterns. LLMs generate natural language explanations for each assignment's due date, thus ensuring interpretability and allowing teachers and students to understand the rationale behind scheduling decisions. Employing past performance and topic knowledge, a clustering module tailors assignment. Current and future developments involve an AI-driven insights panel for visualizing workloads and burnout prediction as well as an "Assignment DNA" feature for classifying assignments according to fundamental competencies such as critical thinking and data analysis. Remaining open and explainable, this platform aims to foster the integration of predictive artificial intelligence and adaptive learning within intelligent educational systems.

Keywords— reinforcement learning, PPO, large language models, adaptive scheduling, explainable AI, educational technology, personalized learning, student workload, AI in education, assignment DNA.

# **Introduction**

Educational institutions find themselves increasingly with the challenge of coordinating diverse students' workloads while maintaining even academic outcomes. Fixed assignment planning systems tend to neglect individual differences in learning rates, conflicting work assignments, and student welfare to create inefficiency and academic strain. Recent technologies in artificial intelligence (AI) present a challenge to overcome the shortcomings through dynamic, data-oriented solutions. This paper presents a new AI-powered scheduling framework based on Proximal Policy Optimization (PPO) for reinforcement learning and large language models (LLMs) for producing human-understandable explanations of scheduling choices. The framework is built with Python for backend operations, MongoDB for handling dynamic workload data, and JavaScript for a user-friendly web-based interface, all containerized using Docker for horizontally scalable deployment. Through its alignment with United Nations Sustainable Development Goal 4 (Quality Education), the framework ensures equal access to individualized academic planning with a view to alleviating cognitive overload and enabling inclusive learning experiences.

# **Related Works**

The combination of reinforcement learning and large language models in educational technologies has gained growing research attention over the past few years. Previous research has investigated adaptive learning systems with reinforcement learning to maximize instructional approaches and student participation. Likewise, natural language generation by large language models has been utilized to improve the explainability of AI-based educational decisions. An effort has been placed to adopt intelligent scheduling software that dynamically reshapes assignment deadline timing based upon workload and performance data, while certain methods consider competency-based analysis and individual task tailoring.

**I. Reinforcement Learning-Based Adaptive Learning Systems:** One notable work employs reinforcement learning to construct adaptive educational settings that adapt to each student's unique and social conditions. The system presented in "A Reinforcement Learning-Based Adaptive Learning System" is centred on dynamically choosing appropriate learning resources depending on individual student states and their receptiveness towards technology. By continually evolving to meet shifting conditions in both individual and group learning environments, the system illustrates how reinforcement learning can provide a more adaptive learning experience. Although the outcomes were encouraging in test environments, the system does not solve the problems of real-time workload balancing or integration with overall classroom scheduling, which our work seeks to address.

**II. Explainability for Large Language Models: A Survey:** The black-box nature of big language models poses threats to their use in educational systems that necessitate transparency and interpretability. The survey "Explainability for Large Language Models" provides an exhaustive taxonomy of methods employed to explain the actions of Transformer-based models. It considers both fine-tuning and prompting paradigms, providing information regarding local (prediction-level) and global (model-level) explanation approaches. Second, it emphasizes the evaluation metrics of explanations and touches on their applicability in debugging and enhancing model behaviour. The paper offers an initial framework with which to appreciate LLMs, but otherwise it is geared mostly toward generalized model behaviour rather than educational implementation contexts. Our framework expands upon this by applying LLM-generate explanations in order to provide assignment scheduling choices that are intelligible to both students and educators.

**III.Raising Student Completion Rates with Adaptive Curriculum and Contextual Bandits:** The paper "Raising Student Completion Rates with Adaptive Curriculum and Contextual Bandits" by Belfer et al. presents an adaptive learning system driven by model-based reinforcement learning. With the help of contextual bandits, the system dynamically assigns learning activities according to student trajectories and continues to learn online, adapting to new content as time passes. A randomized controlled trial showed dramatically higher completion rates and engagement relative to baseline strategies. Although this system is superior in automated content allocation, it fails to provide explainability or workload balancing across topics. Our research builds on these concepts by adding natural language explanations and comprehensive student workload optimization.

**IV. Utilizing Artificial Intelligence for Competency Mapping and Personalised Skill Development in IT Organizations:** The article "Utilizing Artificial Intelligence for Competency Mapping and Personalised Skill Development in IT Organizations" discusses how AI can personalize and automate employee skills identification, competencies, and development route in IT landscapes. Machine learning and data analytics are utilized by the framework to analyze present competencies, fill skills gaps, and enable focused training programs. Although its emphasis is on organizational upskilling, the study's interest in AI-based analysis of competencies and individualized development closely mirrors educational uses. Our research takes this idea into academic environments through the "Assignment DNA" feature, which deconstructs assignments into essential competencies like critical thinking and data analysis, providing actionable feedback on student skill development over time.

# **Literature Review**

The literature review for the paper *“AI-Augmented Assignment Management: A Hybrid RL-LLM Framework for Personalized Scheduling and Content Generation in Education”* surveys key works across reinforcement learning, explainable AI, and skill-based personalization in education. Shawky and Badawi [1] introduced a reinforcement learning-based adaptive system that recommends learning materials based on students’ evolving states and technology acceptance. While effective in simulation, their system did not tackle workload balancing or calendar-level scheduling. Similarly, Belfer et al. [2] proposed an intelligent tutoring system using contextual bandits to assign learning activities. Though their model improved engagement and completion rates, it lacked explainability and deeper competency tracking.

Zhao et al. [3] offered a comprehensive survey on explainability techniques for large language models. Their taxonomy provides deep insights into Transformer behavior, but remains largely theoretical with no direct application in educational platforms. In a parallel domain, Farooqui and Talodhikar [4] explored AI-driven competency mapping within IT organizations, automating skill assessment and personalized development. Despite its corporate focus, the framework parallels educational needs in areas like assignment decomposition and individual skill tracking.

These studies highlight important advances in personalization, automation, and explainability. However, none fully integrate reinforcement learning, explainable language models, and competency-aware assignment planning in a unified academic framework. Our proposed system addresses this gap, offering a novel, interpretable, and workload-sensitive scheduling platform for intelligent educational environments.

# **Methodology**

## **Architecture Diagram**

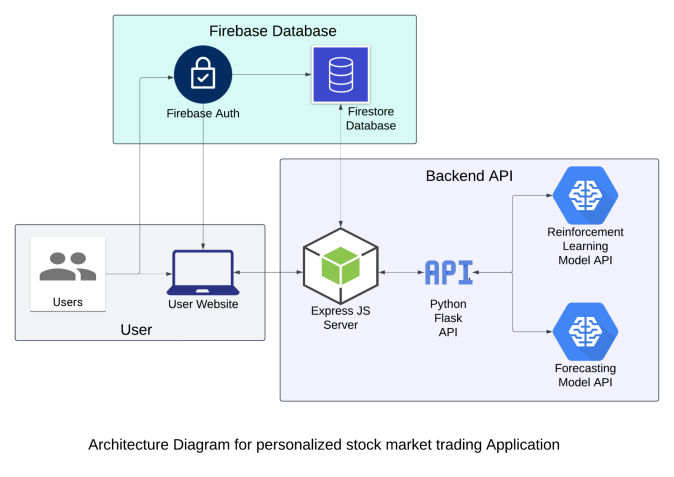


Fig.1. Architecture Diagram

The architecture diagram for the Personalized Stock Market Trading Application illustrates the system's structure, showing the interaction between the user, frontend, backend services, machine learning models, and data storage components.

### **User Interaction:** The system begins with the user, represented as the entry point for interaction. The user accesses the React App, which forms the Frontend of the application. The frontend is responsible for displaying stock data, predictions, and recommendations.

### **Authentication and Database:** The user authentication and data storage are handled by Firebase, which includes Firebase Authentication for user login and Firestore Database for storing user-related stock data. When the user interacts with the system, authentication is validated through Firebase, ensuring secure access.

### **Fetching Stock Data:** The frontend communicates with the Stock API (ytFinance) via a Flask-based API setup, fetching real-time stock prices and related information. This stock data is stored in the Firestore database for further processing and display.

### **Deep Learning Models:** For price predictions, the system integrates a TensorFlow Model API. The frontend sends a request to this API for predicted stock prices, which are processed on the Flask server and stored in the database.

### **Reinforcement Learning Models:** To provide buy/sell recommendations, the system uses a Reinforcement Learning API. Similar to the price prediction, the frontend sends a request for recommendations, which is processed and stored on the backend.

### **Localhost Flask Server:** All the interactions between the frontend and the APIs (stock, deep learning, and reinforcement learning) are facilitated by a Localhost Flask Server. This server handles the communication and ensures data is processed correctly before being stored in the Firestore Database.

### **Dataset and Libraries**: The dataset for real-time stock price forecasting is retrieved using the yFinance API, which provides historical and live stock market data. The historical data, covering one year of daily prices, is preprocessed for missing values and normalized using the MinMaxScaler library to improve model accuracy. Libraries such as TensorFlow, scikit-learn, and Flask are employed to construct machine learning models, preprocess data, and establish server-client communication.

## **Deep Learning Forecasting Model**

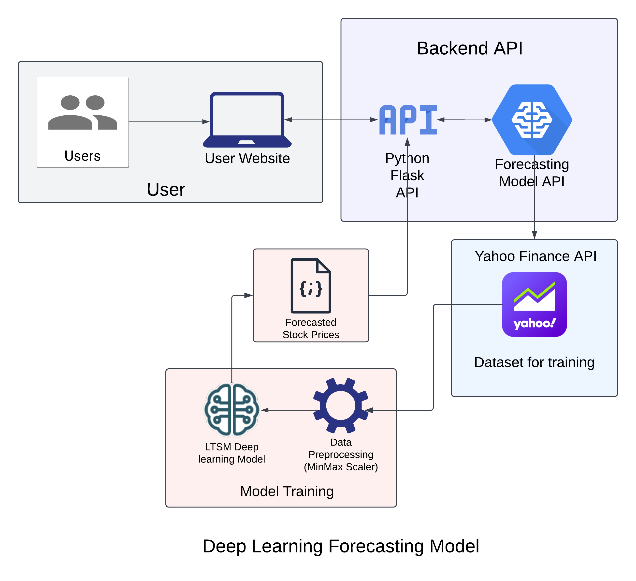


Fig.2. Deep Learning Forecasting Model

The methodology for building the deep learning forecasting model in this project is centred around the use of Long Short-Term Memory (LSTM) neural networks, a type of Recurrent Neural Network (RNN) well-suited for time series data like stock prices. The process begins by fetching historical stock data using the **yfinance** library, which pulls one year of daily stock prices for a given stock symbol. The retrieved data is pre-processed to fill any missing values and scaled using the **MinMaxScaler** from sklearn, ensuring that the data falls within the range of 0 to 1, which is crucial for LSTM model performance.

### **To prepare the data** for the LSTM model, a windowing method is applied where a sequence of 60 days of stock prices is used to predict the next day's price. This sequence is then reshaped into a three-dimensional format required for the LSTM architecture. The model is built using TensorFlow's Sequential API, with two stacked LSTM layers that capture temporal dependencies in the stock prices. The output of the LSTM layers is passed through dense layers to produce the final prediction.

### **The model is trained** on the prepared dataset using a mean squared error (MSE) loss function, optimized using the Adam optimizer. Once trained, the model is used to forecast stock prices for the next 15 days, starting from the last 60 days of available historical data. The prediction is made iteratively by updating the input sequence with each predicted value.

### **The Flask API** allows users to interact with the model by providing a stock symbol through a POST request. The API responds with a 15-day forecast, including the predicted prices and dates.

## **Reinforcement Learning Models**

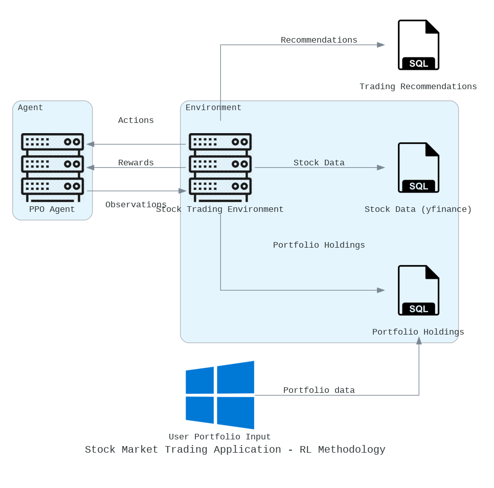


Fig.3. Reinforcement Learning Model

### **Environment Definition:** The environment simulates the stock market and includes stock prices and the user's portfolio.The state is defined by the current stock prices and the portfolio holdings.

### **Action Space**: The agent can take actions such as "Hold" or "Sell" for each stock in the portfolio.

### **Reward System**: The agent receives a reward based on the action taken. For example, selling a stock yields a reward equal to the sale price multiplied by the number of shares sold. The goal is to maximize the total reward over time, reflecting the increase in portfolio value.

### **Agent Training:** A PPO (Proximal Policy Optimization) agent is used to learn the optimal policy. The agent interacts with the environment over multiple episodes, adjusting its policy based on the rewards received.

### **Prediction:** After training, the agent predicts the best action (sell or hold) for the next 7 days based on its learned policy.

# **Results**

This section presents the outcomes of the Stock Market Trading Application, which integrates user-friendly interfaces, forecasting capabilities, and reinforcement learning-based stock recommendations. The application allows users to interact seamlessly with the stock market, enhancing their trading experience.

### **Forecast**

The forecasting feature of the application empowers users to make informed trading decisions by predicting future stock prices. Users can select any stock from their portfolios to generate forecasts. Upon selection, the application utilizes a Long Short-Term Memory (LSTM) model to analyze historical price data and produce predictions for the next 15 days and it is also shown in fig 5.1. This forecasting capability is crucial for users looking to optimize their trading strategies, as it provides insights into potential price movements..

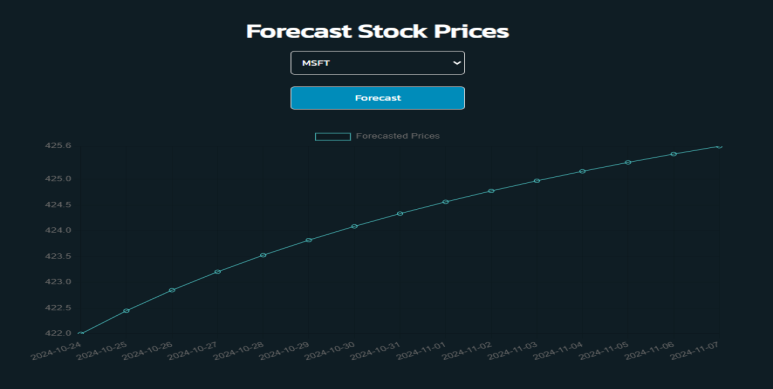


Fig.5.1 shows the forecasting of the stock values

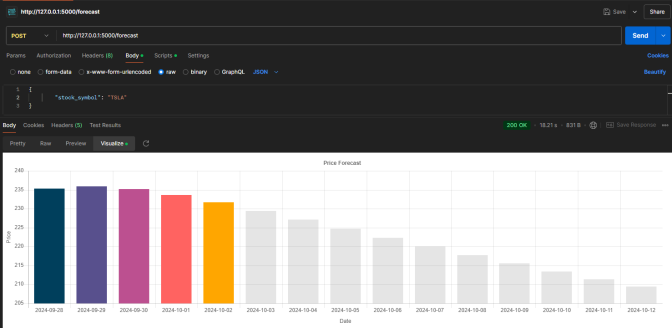


Fig 5.2 Representation of the stock values using bar chart

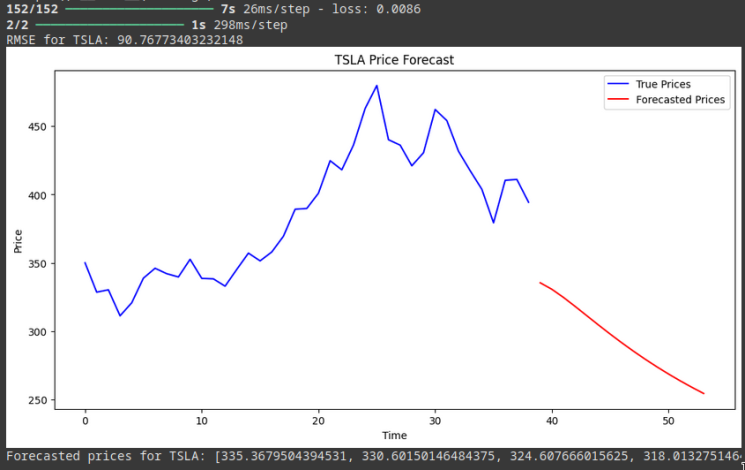


Fig 5.3, RMSE score

In above Figure5.3, RMSE Score for TSLA stock is 90.76 and Forecasting downwards based on previous 30 Days Data suggesting us the dip in value of stock.

### **Recommendations**

In addition to forecasting, the Stock Market Trading Application leverages reinforcement learning (RL) to provide users with personalized stock recommendations. Using a Proximal Policy Optimization (PPO) agent, the application analyzes users' portfolios and the historical performance of various stocks to generate actionable insights. When users interact with the recommendation feature, the RL model evaluates the current market conditions and the user’s holdings to recommend optimal actions, such as when to sell or hold stocks. These recommendations are designed to maximize the user’s portfolio value over time. By combining sophisticated machine learning techniques with user-friendly features, the application not only enhances the user experience but also equips users with the tools necessary for effective investment management.

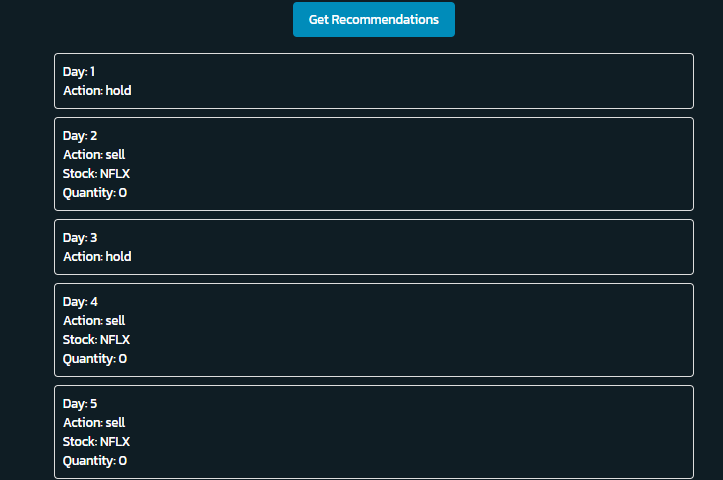


Fig 5.4 Recommendations

In above figure5.4 recommendation based on Day for a week it tell us to hold or sell based on our portfolio.

# **Conclusion**

In conclusion, the Stock Market Trading Application successfully leverages advanced technologies such as deep learning and reinforcement learning to enhance investment strategies for users. Through an intuitive user interface, the application allows users to efficiently manage their stock portfolios, utilizing Firebase for secure user authentication and seamless data storage in Fire store. The forecasting feature empowers users to make informed decisions by predicting stock price movements based on historical data, while the reinforcement learning model provides personalized recommendations, optimizing users' trading actions.

This research demonstrates the potential of integrating machine learning methodologies into financial applications, promoting better investment practices and financial literacy. Future work will focus on refining the recommendation algorithm to adapt to market changes and user behavior, as well as exploring the inclusion of additional analytical tools to further assist users in navigating the complexities of the stock market. The application aims to contribute to financial inclusion and empower both novice and experienced investors in their trading journeys.

# **Future Enhancements**

Future enhancements to the platform aim to improve performance, user experience, and predictive accuracy. Integrating additional financial data sources, such as global stock exchanges, commodities, and crypto currencies, will provide users with a more comprehensive view of market trends. User behavior analytics will enable more personalized recommendations based on trading patterns, while extending the platform’s capabilities to native mobile applications.

Looking ahead, the platform could integrate advanced AI-driven forecasting models, sentiment analysis from social media, and real-time news feeds to offer deeper insights into market trends. Enhancing security with multi-factor authentication and encryption will protect user data, while partnerships with financial institutions and regulatory bodies will ensure compliance and provide exclusive access to financial products and services.

##### References

1. D. P. Kingma and M. Welling, “Auto-Encoding Variational Bayes,” arXiv preprint arXiv:1312.6114, 2013.
2. R. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA: MIT Press, 2018.
3. Y. Zhang, Y. Chen, and L. Li, “A Hybrid Model for Stock Market Forecasting Using LSTM and Reinforcement Learning,” IEEE Access, vol. 8, pp. 45842–45854, 2020.
4. M. S. Pattnaik, “Stock Price Prediction Using Machine Learning Algorithms,” International Journal of Engineering Research and Technology, vol. 8, no. 10, pp. 1663–1668, 2019.
5. F. Chollet, Keras: The Python Deep Learning library, GitHub repository, [Online]. Available: https://github.com/fchollet/keras.
6. J. Brownlee, “How to Develop LSTM Models for Time Series Forecasting,” Machine Learning Mastery, 2018. [Online]. Available: https://machinelearningmastery.com/develop-lstm-models-time-series-forecasting-python-keras/.
7. Y. K. Meena, “Reinforcement Learning in Stock Trading: A Survey,” Journal of Financial Technology, vol. 3, no. 2, pp. 85–98, 2020.
8. R. Arora, “Deep Reinforcement Learning for Stock Trading: A Review,” International Journal of Computer Applications, vol. 182, no. 12, pp. 5–10, 2019.

[9] A. S. Shukla and S. J. S. Bansal, “Application of Machine Learning in Stock Market Forecasting: A Review,” International Journal of Computer Applications, vol. 170, no. 9, pp. 1–6, 2017