MIDTERM PRESENTATION

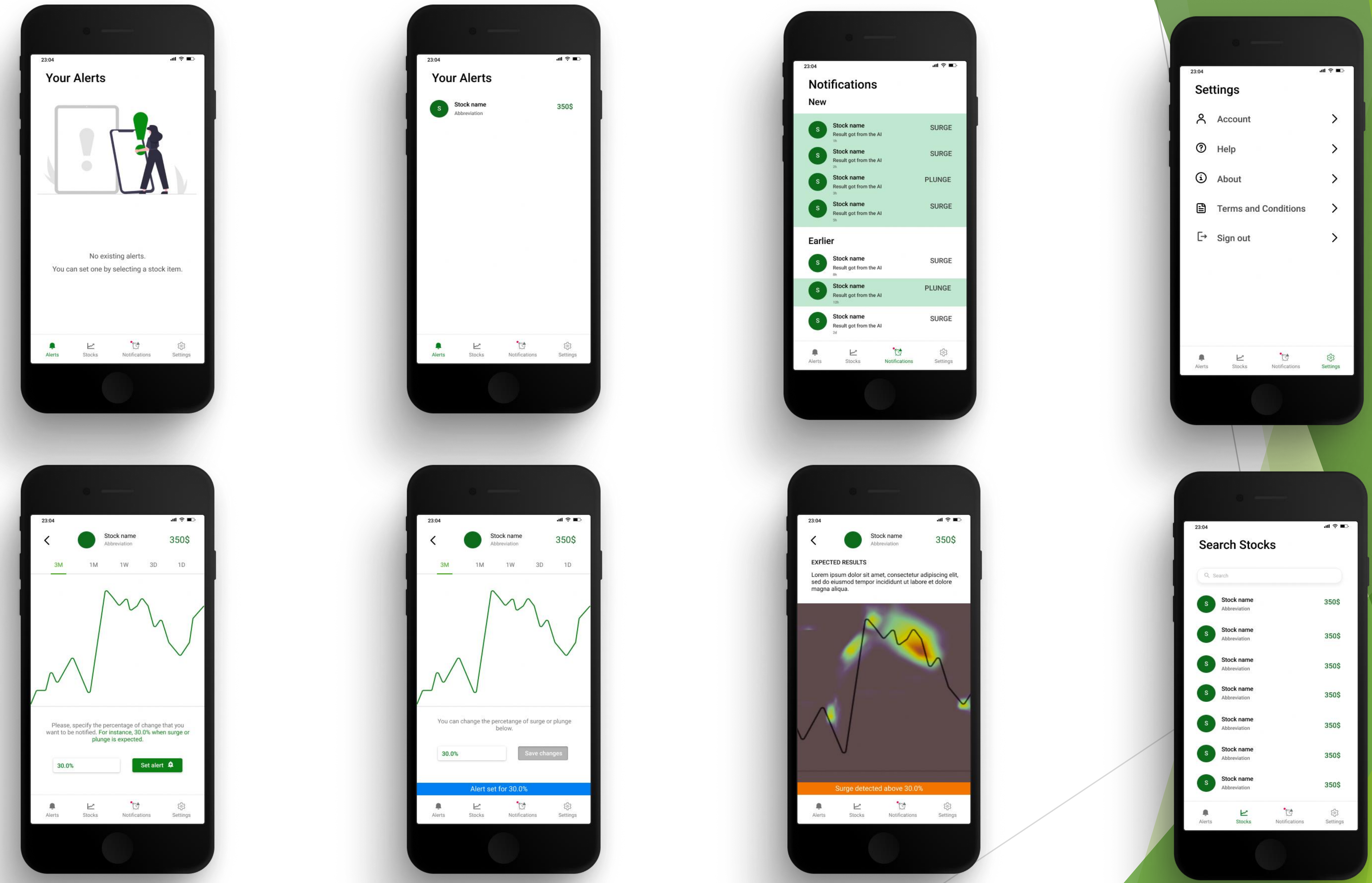
TEAM EXPONENTIAL

Borislav Pavlov, Kim Young Oh,
Park Geo Ryang, Kim Min Jae

OBJECTIVE

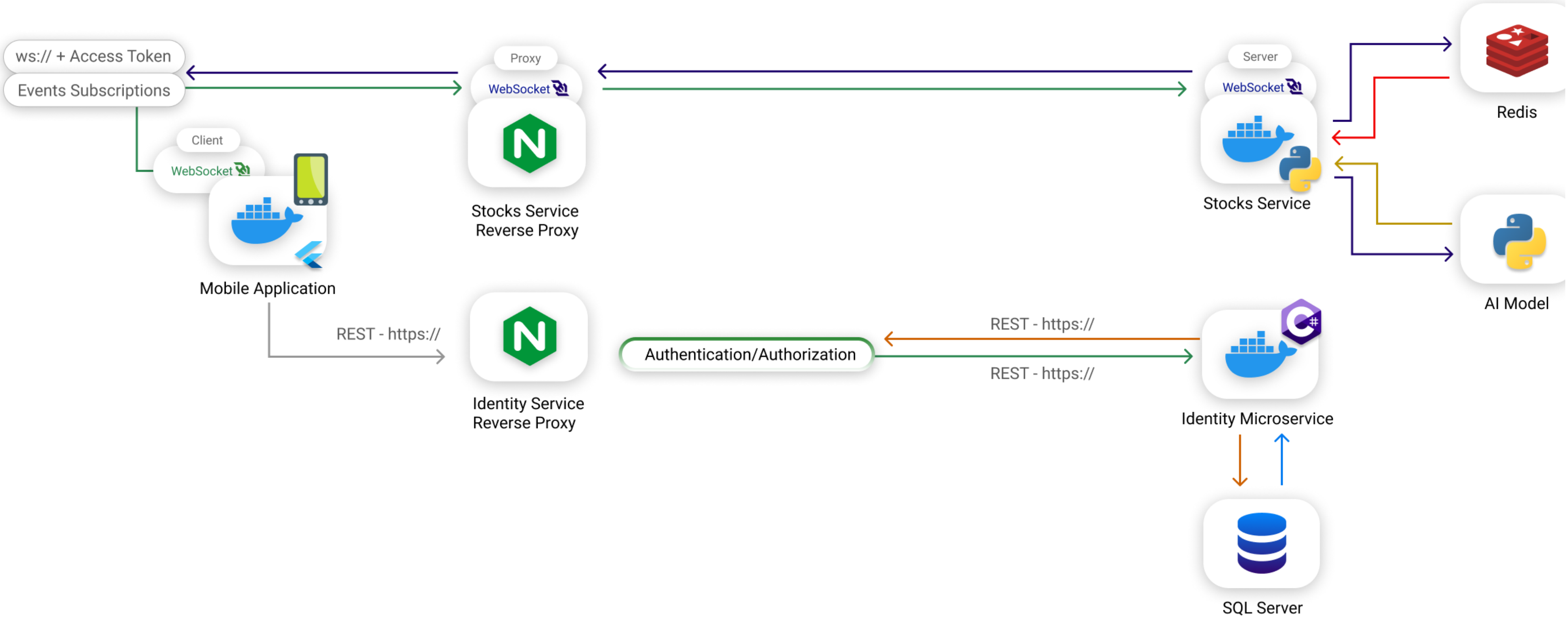
Stock-loss Prevention: Mobile Application with CNN-LSTM Model for Predicting Sharp Rises and Falls in Stock Price

INITIAL DESIGN

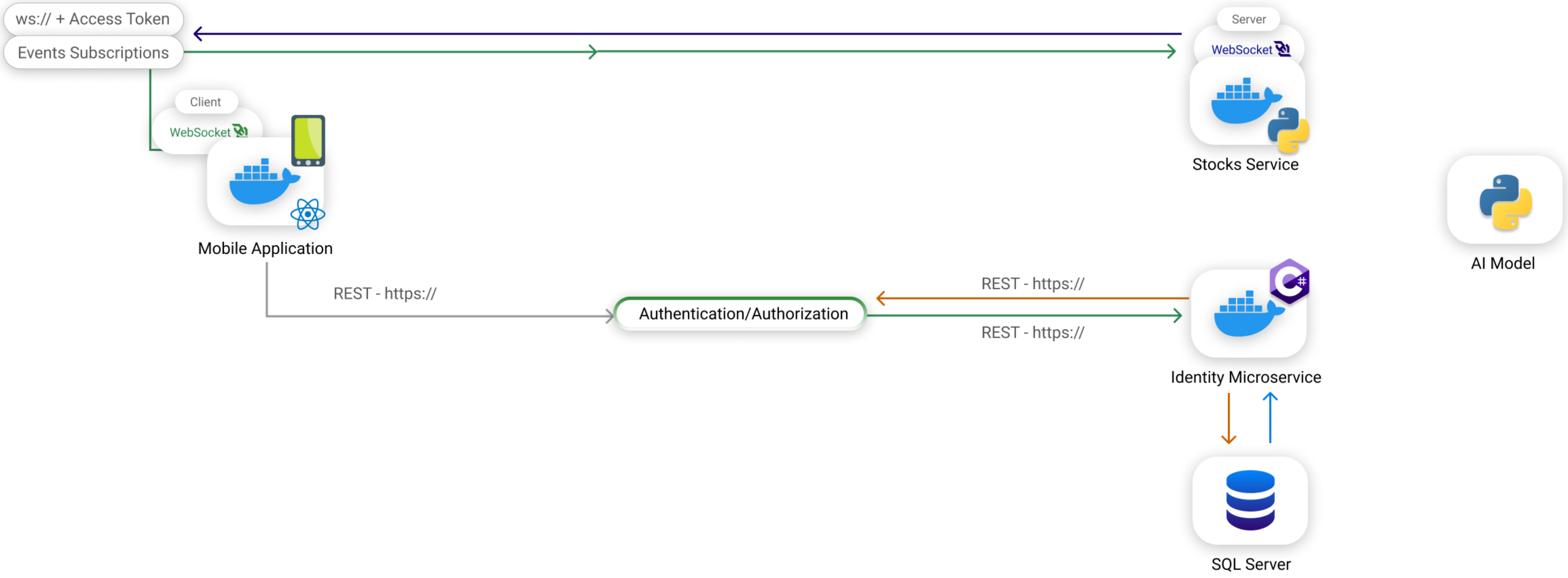


23:04
Your Alerts
No existing alerts.
You can set one by selecting a stock item.
Alerts
Stocks
Notifications
Settings
23:04
Your Alerts
Stock name
Abbreviation
350$
Alerts
Stocks
Notifications
Settings
23:04
Notifications
New
Stock name
Result got from the AI
SURGE
Stock name
Result got from the AI
SURGE
Stock name
Result got from the AI
PLUNGE
Stock name
Result got from the AI
SURGE
Earlier
Stock name
Result got from the AI
SURGE
Stock name
Result got from the AI
PLUNGE
Stock name
Result got from the AI
SURGE
Alerts
Stocks
Notifications
Settings
23:04
Settings
Account
Help
About
Terms and Conditions
Sign out
Alerts
Stocks
Notifications
Settings
23:04
Stock name
Abbreviation
350$
3M
1M
1W
3D
1D
Please, specify the percentage of change that you want to be notified. For instance, 30.0% when surge or plunge is expected.
30.0%
Set alert
Alerts
Stocks
Notifications
Settings
23:04
Stock name
Abbreviation
350$
3M
1M
1W
3D
1D
You can change the percetange of surge or plunge below.
30.0%
Save changes
Alert set for 30.0%
Alerts
Stocks
Notifications
Settings
23:04
Stock name
Abbreviation
350$
EXPECTED RESULTS
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Surge detected above 30.0%
Alerts
Stocks
Notifications
Settings
23:04
Search Stocks
Search
Stock name
Abbreviation
350$
Stock name
Abbreviation
350$
Stock name
Abbreviation
350$
Stock name
Abbreviation
350$
Stock name
Abbreviation
350$
Stock name
Abbreviation
350$
Stock name
Abbreviation
350$
Alerts
Stocks
Notifications
Settings

INITIAL DESIGN



CURRENT PROGRESS



SCHEDULE

PART	CONTENT	SCHEDULE												
		MONTH	9	10				11				12		
		WEEK	4	1	2	3	4	1	2	3	4	1	2	3
AI	Research on thesis													
	Use and modification of the AI model code													
	Apply explainable AI, GradCAM													
	performance improvement													
Mobile Application - Research & Design	Research a framework													
	Research a Websocket													
	Create initial wireframes for the mobile application													
Mobile Application - Implementation & Testing	Implement the initial design view of the application and state management													
	Implement WebSocket service													
Backend API	Research													
	Implementation													
	Testing													

WHAT IS DONE

- **MOBILE APPLICATION - React Native & TypeScript + Redux**
  - Screens navigation
  - Redux setup for state management
  - Authentication
  - WebSocket Connection
  - Real time stocks data handling
- **IDENTITY API - C# - ASP.NET 5**
  - Database Entities Setup with Initial Migration
  - Endpoints for user registration and login
  - JWT Token Generation and Validation
  - Endpoint for token refreshing
- **STOCKS SERVICE - Python**
  - WebSocket connection for handling/sending messages by/to the mobile app
  - WebSocket connection for retrieving real time stocks data – Yahoo Finances API
  - Retrieving historical data for stocks
- **AI MODEL - TBD**
  - Image dataset augmentation for CNN model
  - Making CNN, LSTM, LSTM-CNN model in 1 epoch and 100 epochs.

WHAT IS NEXT

- **MOBILE APPLICATION**
  - 1.Setting Alerts
  - 2.Handling Notifications
  - 3.Design of Screens

- **STOCKS SERVICE**
  - 1.Figure out how to store and retrieve alerts efficiently
  - 2.Integrate AI Model
  - 3.Send notifications
  - 4.Integrate Redis

- **AI MODEL**
  - Find an indicator that can explain why a prediction was made, like a heatmap.
  - Reducing the loss by adjusting the epoch.
  - Find other AIs that can replace complex code.

CHALLENGES

- **MOBILE APPLICATION**
  - Automatic token refreshing
  - State management with redux is complicated
  - Handling of WebSocket messages

- **STOCKS SERVICE**
  - Supporting two socket connections:
    - Retrieving live stocks data on one thread
    - Handling mobile application messages on another thread and sending live data
  - Giving supported stocks for the first time is a little slow – caching is needed

- **AI MODEL**
  - Overfitting
    - Adjusting hyperparameters ex) epochs
  - Searching the way to visualize
  - Performance is bad
    - Thinking about searching another models.

LIMITATIONS

- **MOBILE APPLICATION**
  - WebSocket might not be needed – e.g. refactor
- **STOCKS SERVICE**
  - Caching is needed for faster responses – Redis will fix that
  - Efficient querying is needed for getting alerts in the future – Research methods for improving query performance on sql database – e.g. indexing
  - WebSocket might not be needed – e.g. refactor
- **AI MODEL**
  - Even though the original author's code was used as it is, the loss is large, so we are thinking about whether to find another model or use it as it is.

AI MODEL IN DETAILS

- AI Model Reference
  - Paper Citation
    - Kim T, Kim HY (2019) Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data. PLoS ONE 14(2): e0212320. https://doi.org/ 10.1371/journal.pone.0212320
  - Paper code
    - https://github.com/luanft/lstm-cnn-model
  - Dataset
    - **tw_spydata_raw.csv** from Figshare
    - https://figshare.com/articles/dataset/Forecasting_Stock_Prices_with_a_Feature_Fusion_LSTM-CNN_Model_Using_Different_Representations_of_the_Same_Data/7471568
    - SPY stock price data with minute-by-minute → 98,310 datas
    - Time(minute) / Trade High Value / Trade Low Value / Trade Open Value / Trade Close Value / Trade Volume / Trade Count

	A	B	C	D	E	F	G
1	Time	Trade High	Trade Low	Trade Open	Trade Close	Trade Volume	Trade Count
2	0	214.23	214.14	214.15	214.155	1022241	2274
3	1	214.38	214.14	214.15	214.3699	582984	1902
4	2	214.37	214.18	214.37	214.28	705964	1943
5	3	214.3	214.16	214.29	214.19	430066	1321
6	4	214.2	214.09	214.18	214.1	444761	1599
7	5	214.25	214.11	214.11	214.23	284215	1193
8	6	214.3	214.22	214.235	214.22	354142	1144

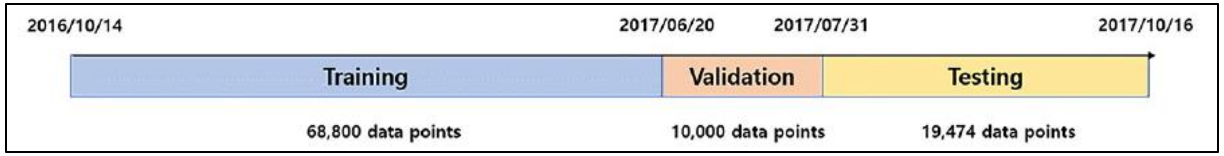
▲tw_spydata_raw.csv

AI MODEL IN DETAILS

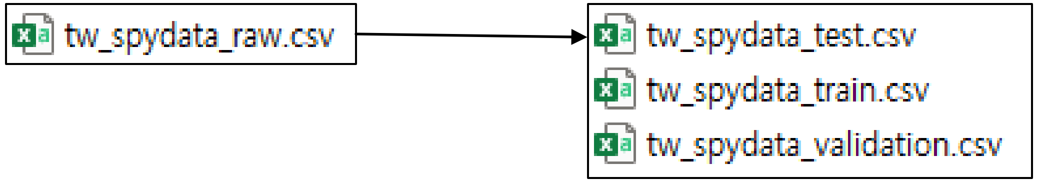
- Dataset
  - Raw-dataset (before seperated) - 98,310 datas

	A	B	C	D	E	F	G
1	Time	Trade High	Trade Low	Trade Open	Trade Close	Trade Volume	Trade Count
2	0	214.23	214.14	214.15	214.155	1022241	2274
3	1	214.38	214.14	214.15	214.3699	582984	1902
4	2	214.37	214.18	214.37	214.28	705964	1943
5	3	214.3	214.16	214.29	214.19	430066	1321
6	4	214.2	214.09	214.18	214.1	444761	1599
7	5	214.25	214.11	214.11	214.23	284215	1193
8	6	214.3	214.22	214.235	214.22	354142	1144

  - Processed-Dataset (after seperated)
    - Row-dataset to Train/Validation/Test dataset (ratio: 68,800/10,000/19,474)

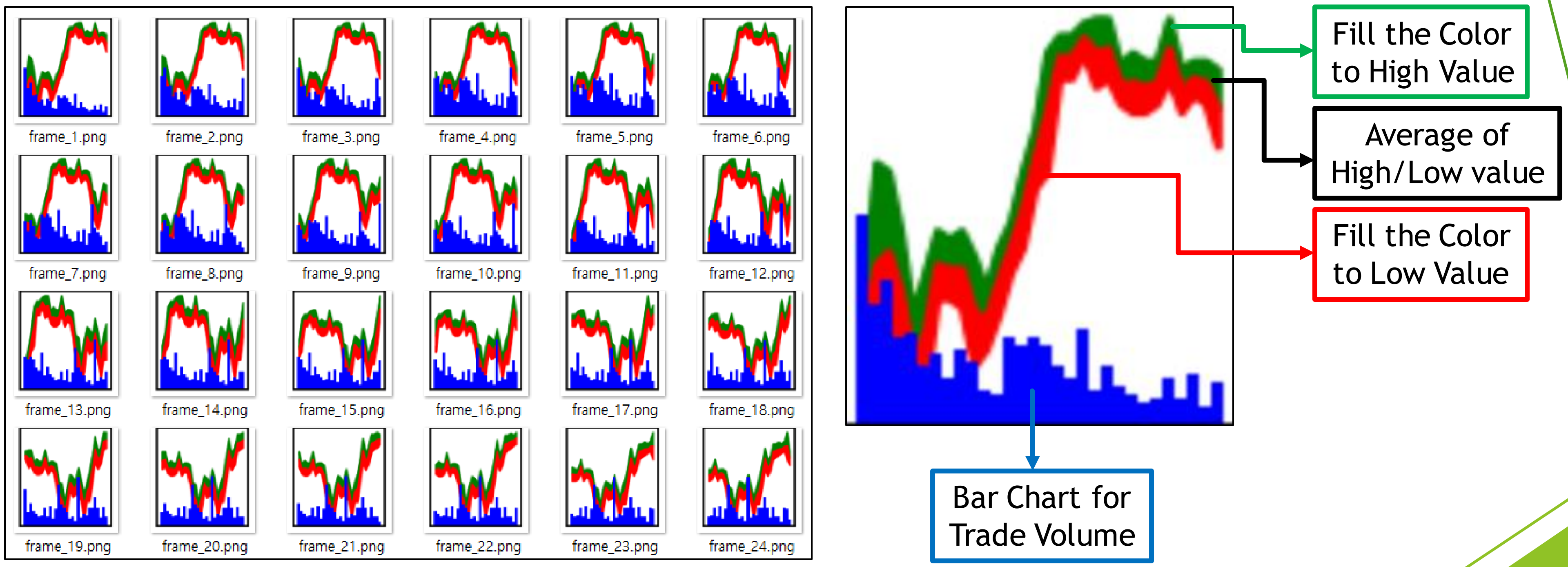


    - CSV Data split



AI MODEL IN DETAILS

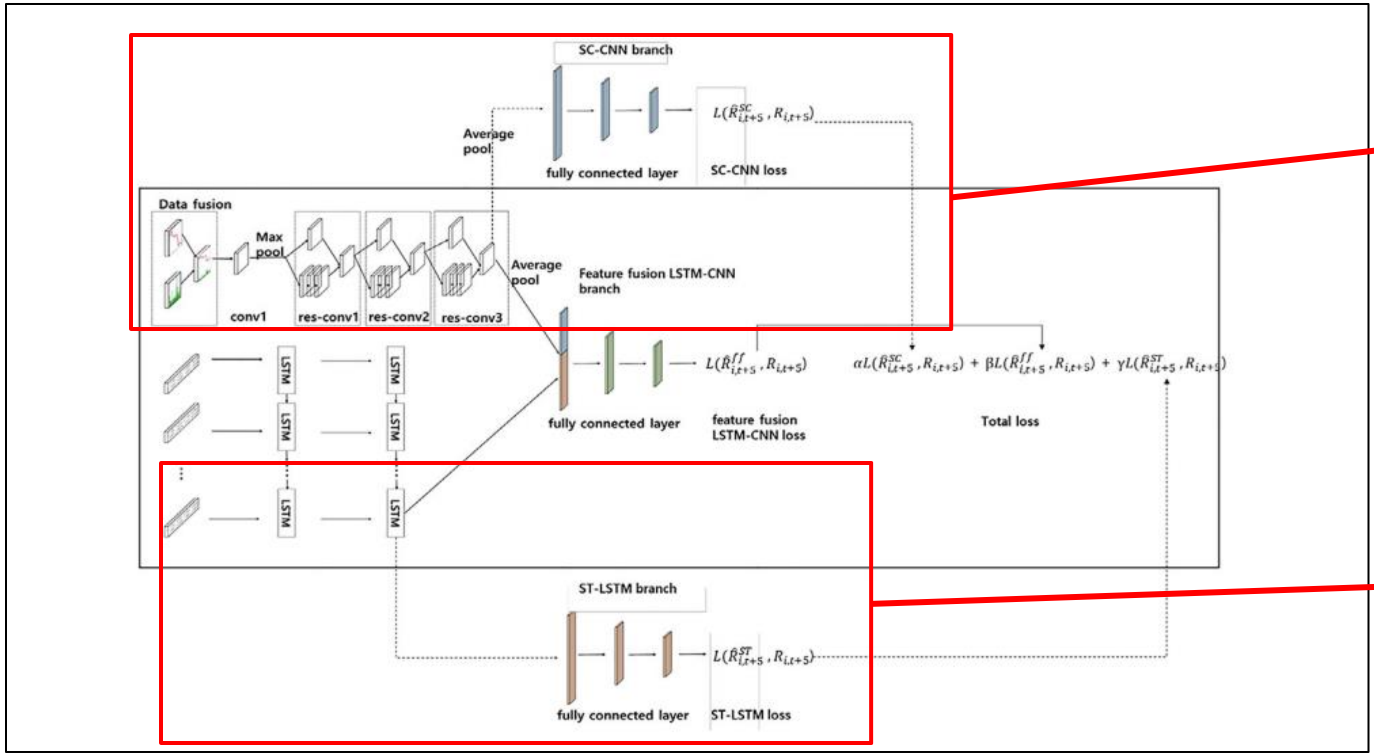
- Preprocessing Dataset



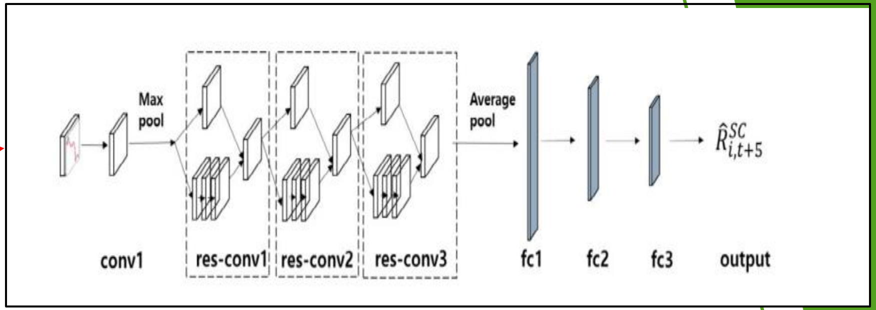
To create the model in this experiment, we incorporate a middle price by averaging the high and low prices, and we then fill the colors between the prices to provide more information to the CNN.

AI MODEL IN DETAILS

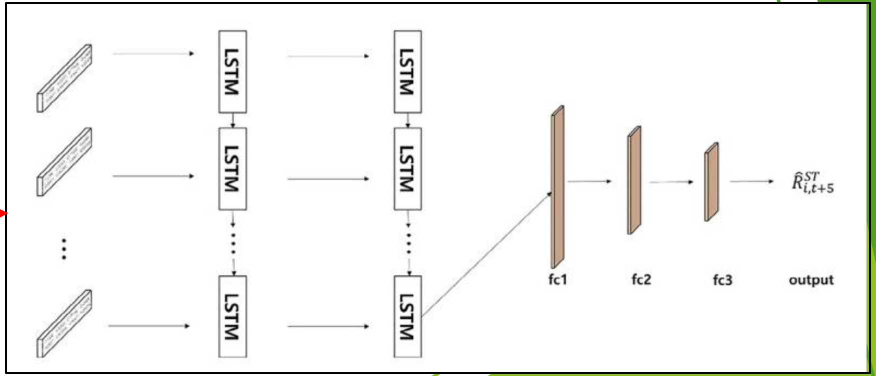
- Architecture



▲ Full Model Architecture



▲ SC-CNN Branch



▲ ST-LSTM Branch

AI MODEL IN DETAILS

- Hyperparameters & Evaluate
  - Hyperparameters

```
import easydict
train_args = easydict.EasyDict({
    "name":"train",
    "epoch":100,
    "batch_size":32,
    "learning_rate":0.003,
    "epsilon":0.1
})
```

* 100 Epochs are defualt

  - Evaluate

```
lstm_cnn_model.compile(
    optimizer=adam_optimizer, loss=tf.losses.MeanSquaredError(),
    metrics=['mape', tf.keras.metrics.RootMeanSquaredError(name='rmse'), RMAE]
)
```

    - Metrics
      - MAPE : Mean Absolute Percentage Error
      - RMSE : Root Mean Square Error (main)
      - RMAE : Root Mean Absolute Error

AI MODEL IN DETAILS

- Input & Output (batch size : 32)
  - CNN
    - Input : High prices, low prices, volumes → 112 x 112 RGB images
    - Output:

```
average_pooling2d_1 (AveragePoo (32, 1, 1, 512)      0            re_lu_5[0][0]
```

  - LSTM
    - Input : Close prices, volumes → Logarithmic return
    - Output :

```
flatten_3 (Flatten)              (32, 512)             0            average_pooling2d_1[0][0]
```

    - Logarithmic return

Input

$$\begin{bmatrix} log\frac{P_t}{P_{t-1}}, log\frac{P_{t+1}}{P_t}, \cdots, log\frac{P_{t+28}}{P_{t+27}} \\ log\frac{V_t}{V_{t-1}}, log\frac{V_{t+1}}{V_t}, \cdots, log\frac{V_{t+28}}{V_{t+27}} \end{bmatrix}$$

$$\begin{bmatrix} log\frac{P_{t+1}}{P_t}, log\frac{P_{t+2}}{P_{t+1}}, \cdots, log\frac{P_{t+29}}{P_{t+28}} \\ log\frac{V_{t+1}}{V}, log\frac{V_{t+2}}{V_{t+1}}, \cdots, log\frac{V_{t+29}}{V_{t+28}} \end{bmatrix}$$

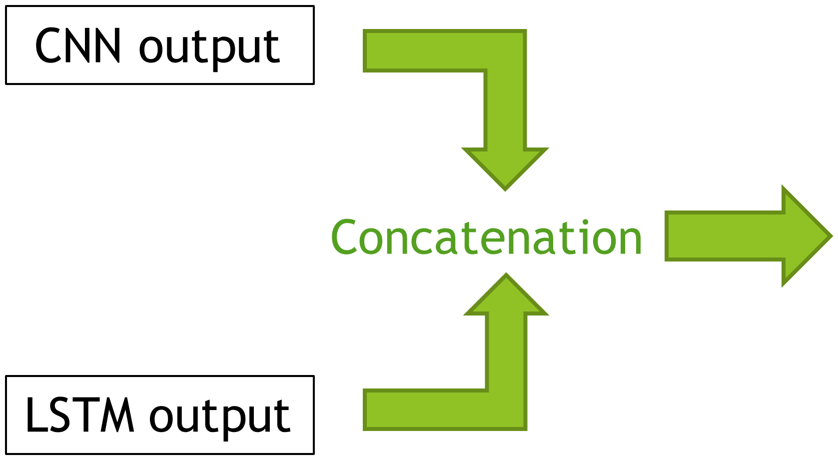
Output

$$\left[log\frac{P_{t+33}}{P_{t+28}}\right]$$

$$\left[log\frac{P_{t+33}}{P_{t+28}}\right]$$

AI MODEL IN DETAILS

- CNN-LSTM Concatenation



Fully connected layers

(combined features → 500 → 100 → 25 → 1)

concatenate_1 (Concatenate)	(32, 541)	0	flatten_4[0][0] flatten_3[0][0]
flatten_5 (Flatten)	(32, 541)	0	concatenate_1[0][0]
dense_20 (Dense)	(32, 500)	271000	flatten_5[0][0]
dropout_15 (Dropout)	(32, 500)	0	dense_20[0][0]
dense_21 (Dense)	(32, 100)	50100	dropout_15[0][0]
dropout_16 (Dropout)	(32, 100)	0	dense_21[0][0]
dense_22 (Dense)	(32, 25)	2525	dropout_16[0][0]
dropout_17 (Dropout)	(32, 25)	0	dense_22[0][0]
dense_23 (Dense)	(32, 1)	26	dropout_17[0][0]

AI MODEL IN DETAILS

- Model Predict

```
from keras.models import load_model
model = load_model('CNN_LSTM_model_epoch_1.h5', compile=False)

model.summary()

test_dataset: tf.data.Dataset = load_lstm_cnn_dataset(
            WINDOW_SIZE, PREDICT_SIZE,
            create_bar_filled_line_fusion_chart,
            TEST_SP500_DATA_FILE, 'test'
)
test_dataset = test_dataset.cache(get_cache_file('test', 'lstm_cnn'))
test_dataset = test_dataset.batch(train_args.batch_size, drop_remainder=True)
test_predict = model.predict(test_dataset)
print(type(test_predict))
print(test_predict)
```

▲ Code for predict

```
[[207.37186]
 [207.36519]
 [207.37459]
 ...
 [207.37128]
 [207.38837]
 [207.3983 ]]
```

→ Predict with model **(epoch 1)**

```
[[242.25653]
 [242.25653]
 [242.25653]
 ...
 [242.25653]
 [242.25653]
 [242.25653]]
```

→ Predict with model **(epoch 100)**

- Predict with test dataset → 19,474 datas
- 19,424 predict data for each epoch's model

- 50 differs between input data and output data
  → **Drop_remainder = True, then Batch caused the differs**

AI MODEL IN DETAILS

- Measurement on results

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_{1,i} - x_{2,i})^2}$$

$$RMAE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|x_{1,i} - x_{2,i}|}$$

$$MAPE = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{x_{2,i} - x_{1,i}}{x_{1,i}}\right|$$

We mainly focus on the root mean square error (RMSE) because the reference code use the sum of RMSE as a total loss

AI MODEL IN DETAILS

LSTM-CNN > Default epoch 100 case, val_RMSE is 9.2840

```
Epoch 97/100
2148/2148 [==============================] - 15s 7ms/step - loss: 86.0179 - mape: 2.9277 - rmse: 9.2746 - RMAE: 2.2369 - val_loss: 32.
3671 - val_mape: 2.1306 - val_rmse: 5.6892 - val_RMAE: 2.2246
Epoch 98/100
2148/2148 [==============================] - 15s 7ms/step - loss: 88.4062 - mape: 3.0699 - rmse: 9.4025 - RMAE: 2.3724 - val_loss: 19.
1613 - val_mape: 1.5434 - val_rmse: 4.3774 - val_RMAE: 1.8222
Epoch 99/100
2148/2148 [==============================] - 15s 7ms/step - loss: 94.5090 - mape: 3.1003 - rmse: 9.7216 - RMAE: 2.3475 - val_loss: 18.
9389 - val_mape: 1.5324 - val_rmse: 4.3519 - val_RMAE: 1.8144
Epoch 100/100
2148/2148 [==============================] - 15s 7ms/step - loss: 88.9032 - mape: 3.0193 - rmse: 9.4288 - RMAE: 2.3093 - val_loss: 30.
1742 - val_mape: 2.0436 - val_rmse: 5.4931 - val_RMAE: 2.1717
Evaluating model
607/607 [==============================] - 2s 3ms/step - loss: 86.1918 - mape: 3.4445 - rmse: 9.2840 - RMAE: 2.8726
Metric score:
{'RMAE': 2.8726494312286377,
 'loss': 86.19175720214844,
 'mape': 3.444488763809204,
 'rmse': 9.283951759338379}
```

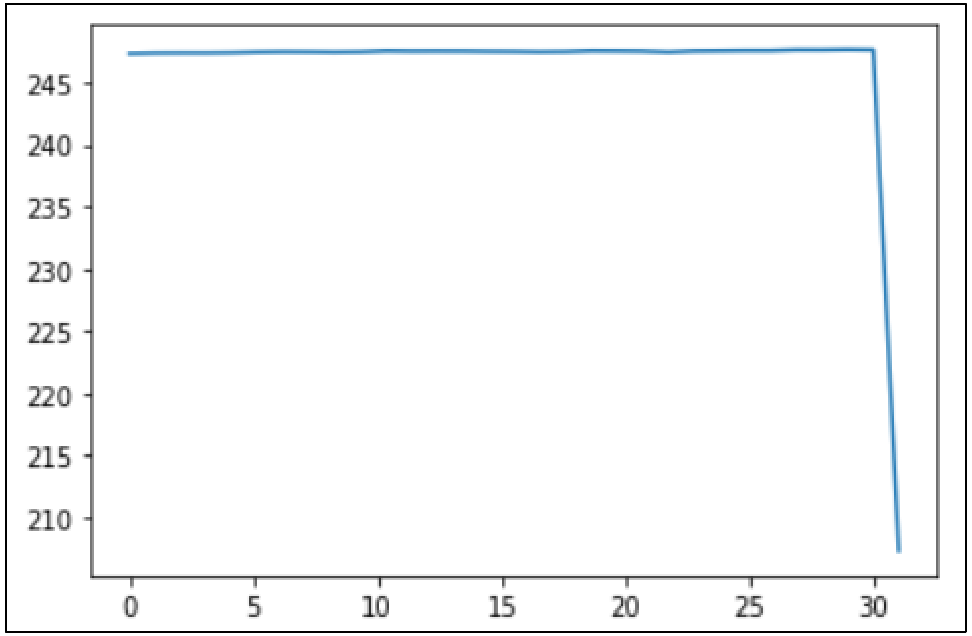
The best case is 54 epoch

```
Epoch 54/100
2148/2148 [==============================] - 952s 443ms/step - loss: 10.6601 - mape: 0.9122 - rmse: 3.2650
- RMAE: 1.2726 - val_loss: 16.4362 - val_mape: 1.4073 - val_rmse: 4.0542 - val_RMAE: 1.7209
```

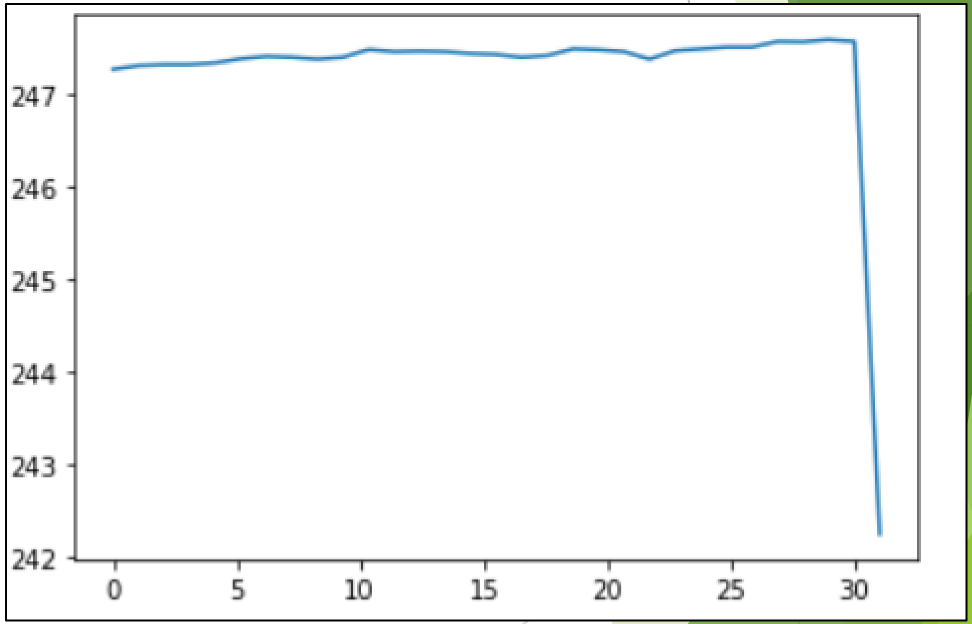
The performance problem

- Graph with [30min data] + [predict data after 30min]

Epoch 1 case



Epoch 100 case



TEAM ROLES

- Team Lead: Borislav Pavlov

- AI Algorithms:
  - Main - Kim Young Oh, Kim Min Jae
  - Supported – Borislav Pavlov, Park Geo Ryang

- Mobile Application + Additional Services
  - Main - Borislav Pavlov, Park Geo Ryang
  - Supported - Kim Young Oh, Kim Min Jae

DEMO

1398

Q & A

1398