FINAL PRESENTATION

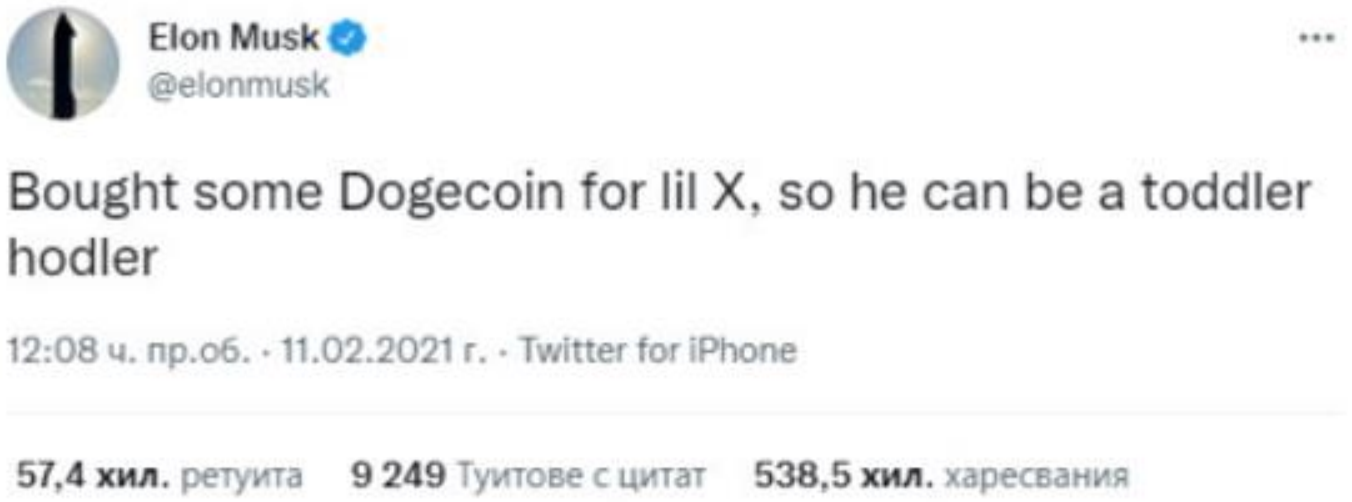
TEAM EXPONENTIAL

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

MOTIVATION



Elon Musk
@elonmusk

Bought some Dogecoin for lil X, so he can be a toddler hodler

12:08 ч. пр.об. · 11.02.2021 г. · Twitter for iPhone

57,4 хил. ретуита **9 249** Туитове с цитат **538,5 хил.** харесвания

PROJECT PROGRESS

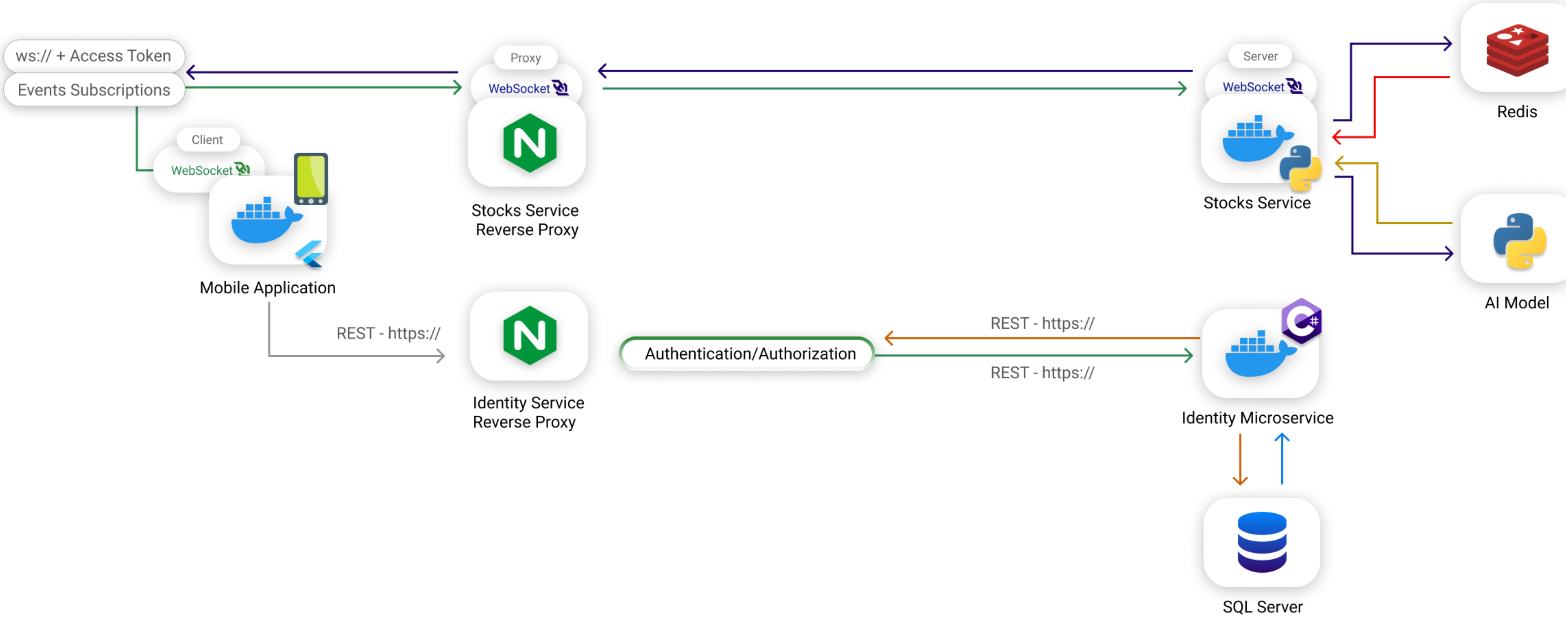
SCHEDULE

PART	CONTENT	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						🟩	🟩	🟩	🟩				
	Refactoring										🟩	🟩	🟩	🟩
Backend API	Research		🟩	🟩	🟩									
	Implementation					🟩	🟩	🟩	🟩	🟩	🟩			
	Testing											🟩	🟩	🟩

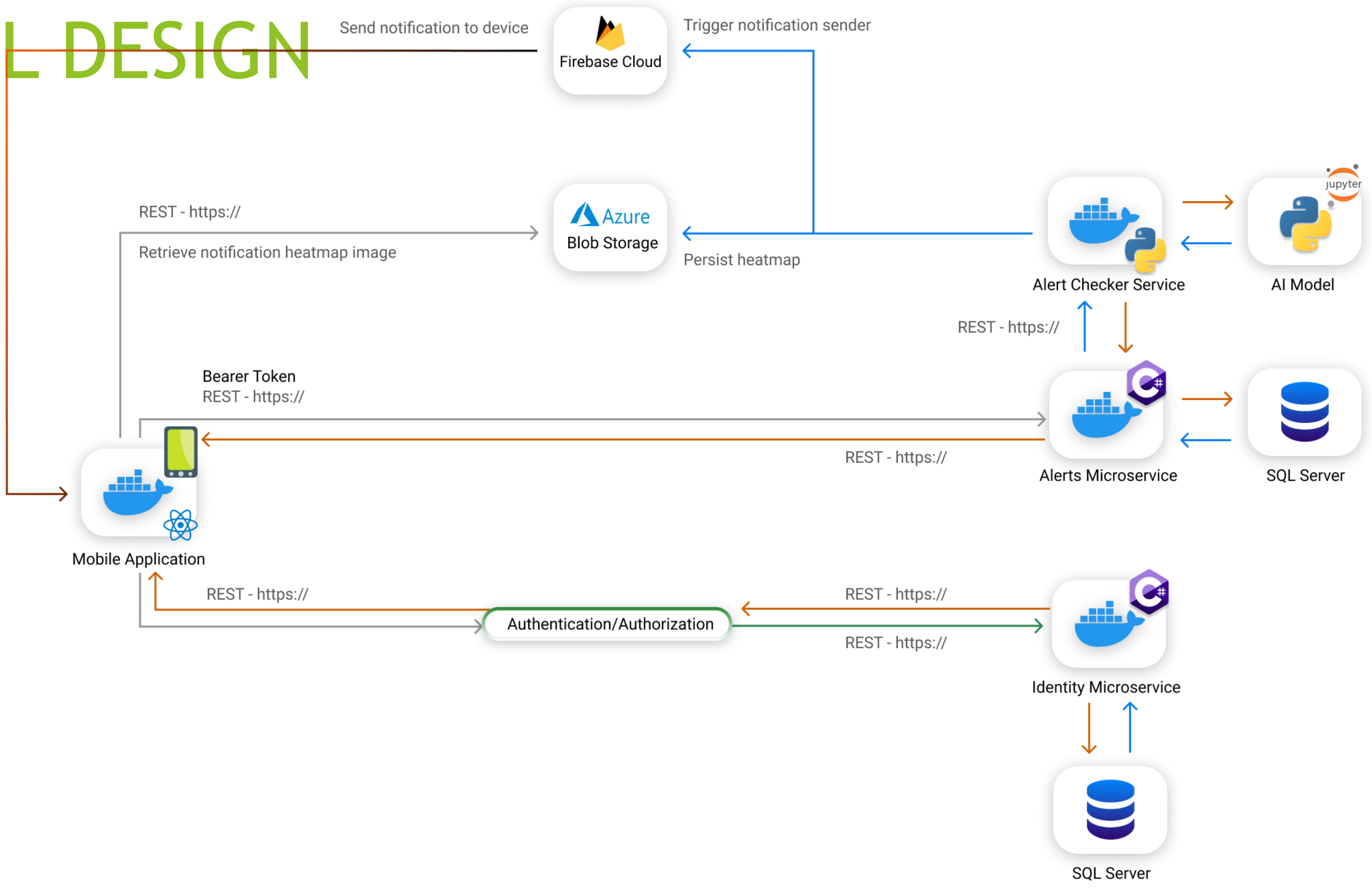
TEAM ROLES

- 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

INITIAL DESIGN

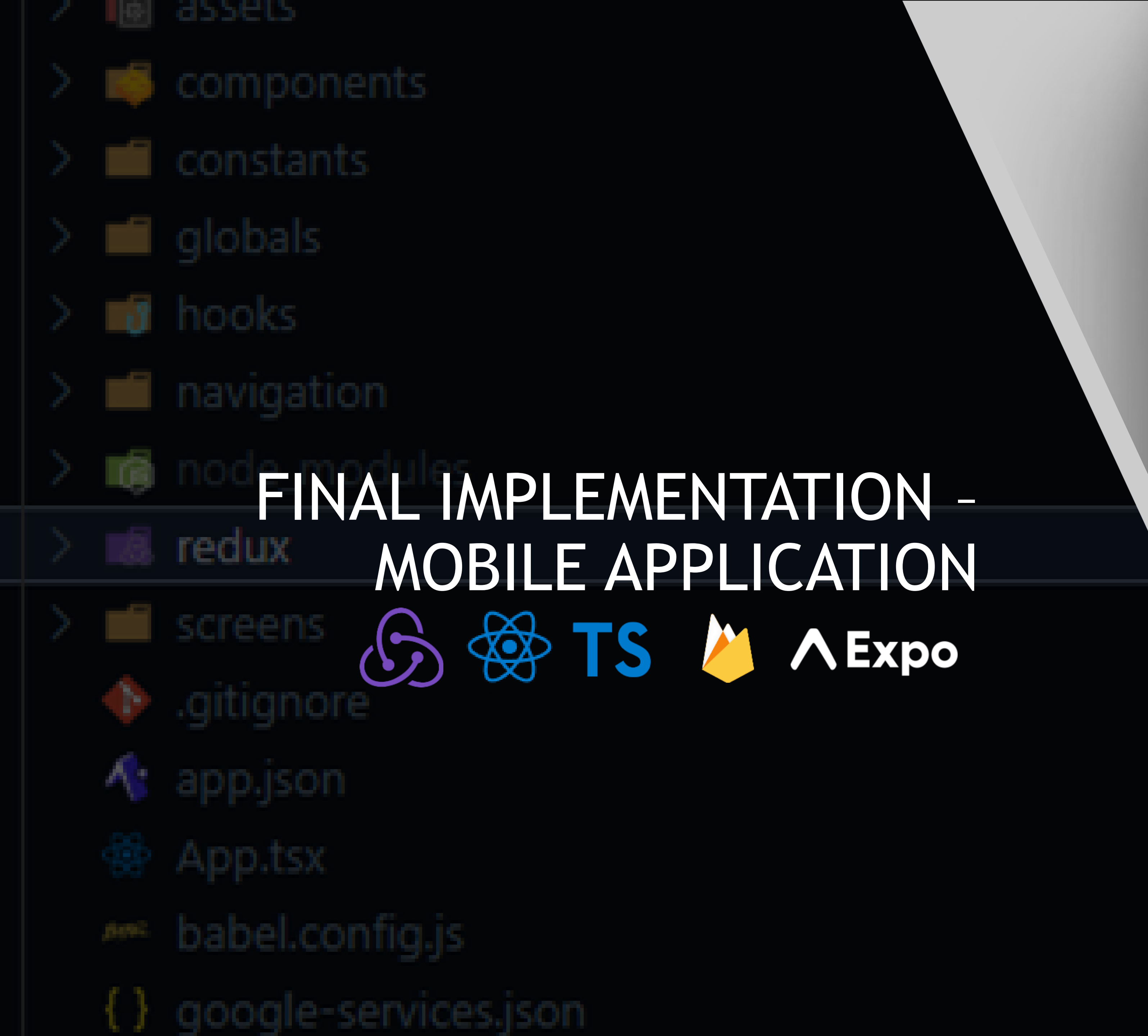


FINAL DESIGN

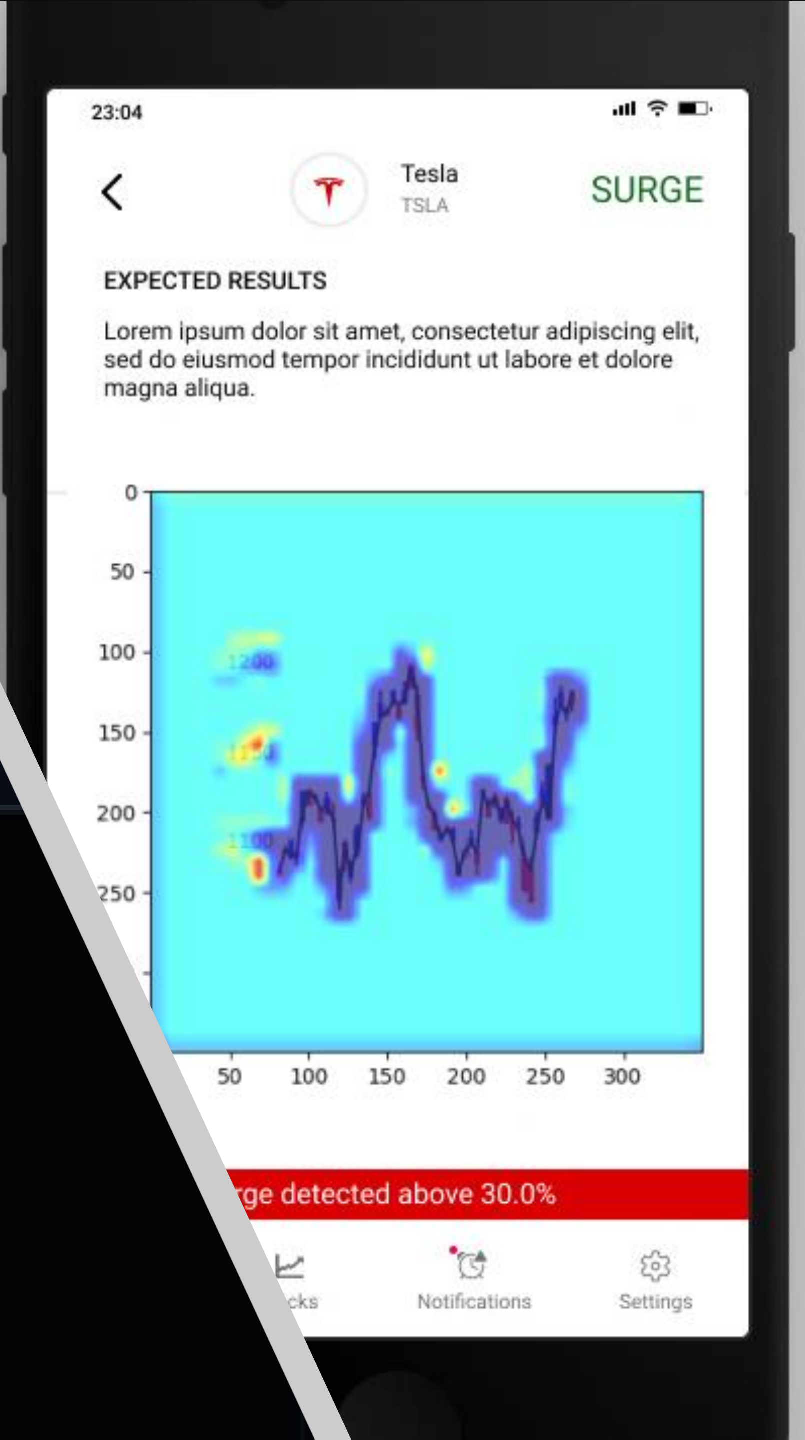


FINAL IMPLEMENTATION

FINAL IMPLEMENTATION – MOBILE APPLICATION



```
assets
components
constants
globals
hooks
navigation
node_modules
redux
screens
.gitignore
app.json
App.tsx
babel.config.js
google-services.json
```



23:04

Tesla
TSLA

SURGE

EXPECTED RESULTS

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

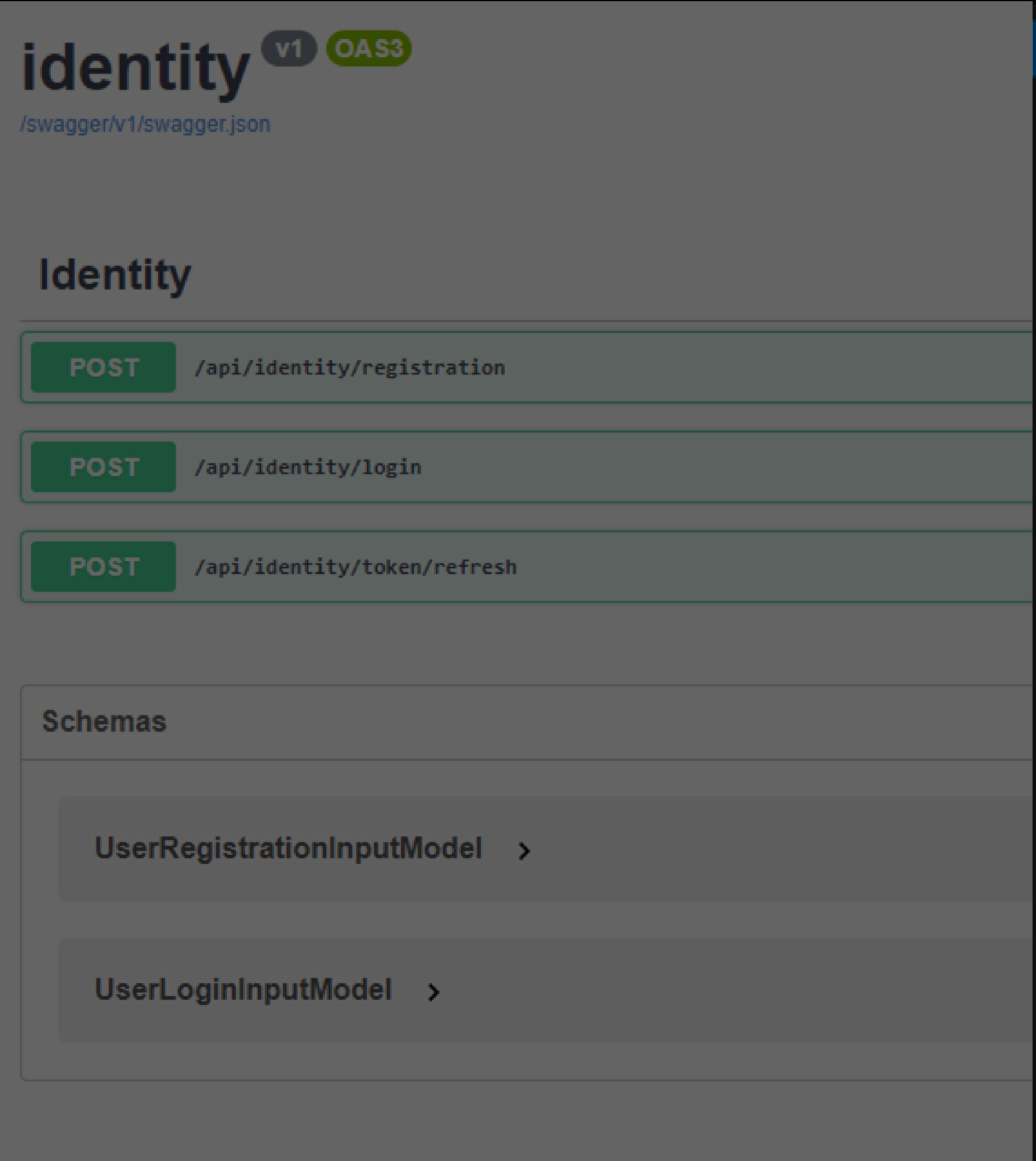
rge detected above 30.0%

FINAL IMPLEMENTATION
IDENTITY SERVICE

Microsoft SQL Server

Entity Framework

.NET



identity v1 OAS3

/swagger/v1/swagger.json

Identity

POST /api/identity/registration

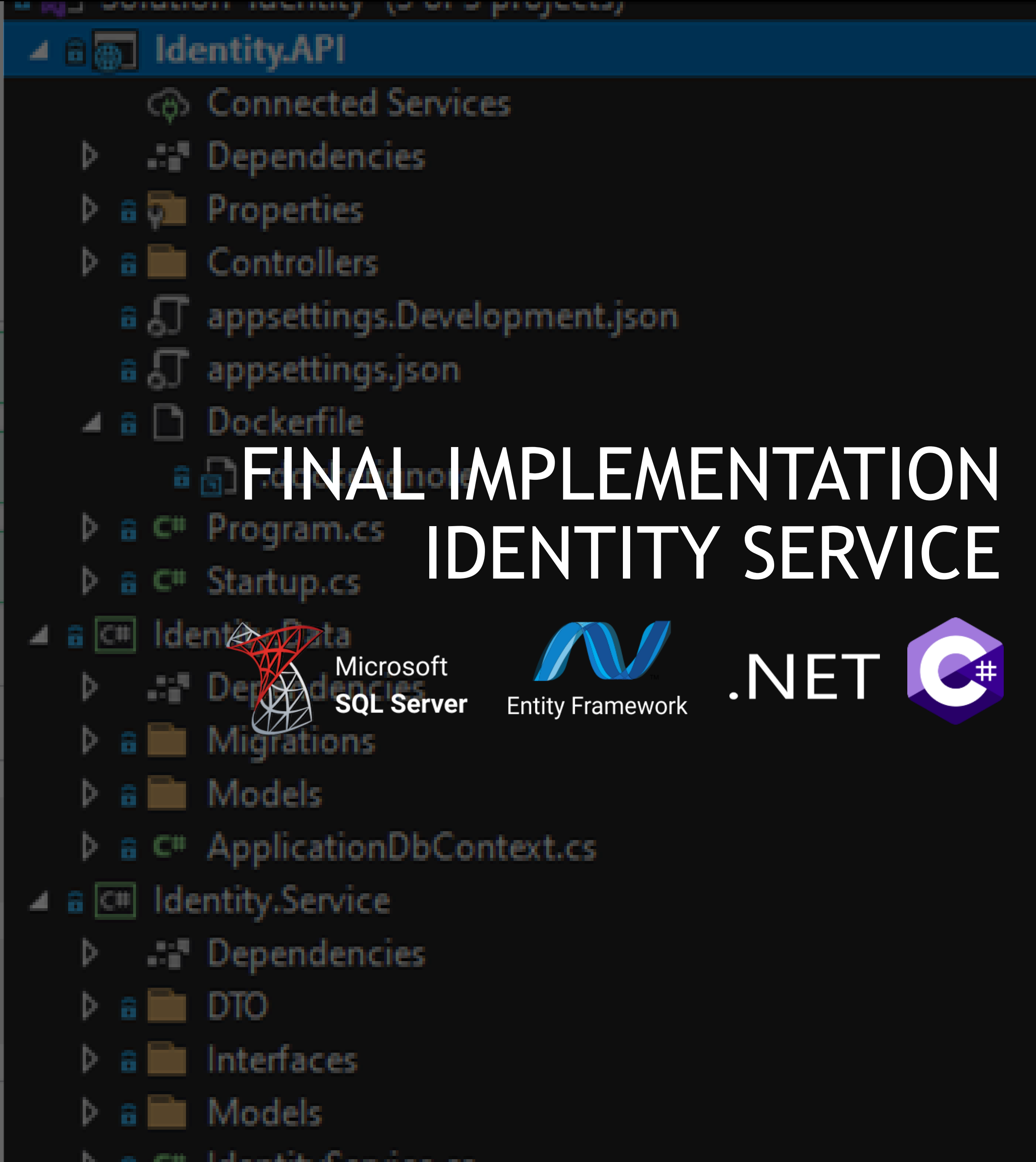
POST /api/identity/login

POST /api/identity/token/refresh

Schemas

UserRegistrationInputModel >

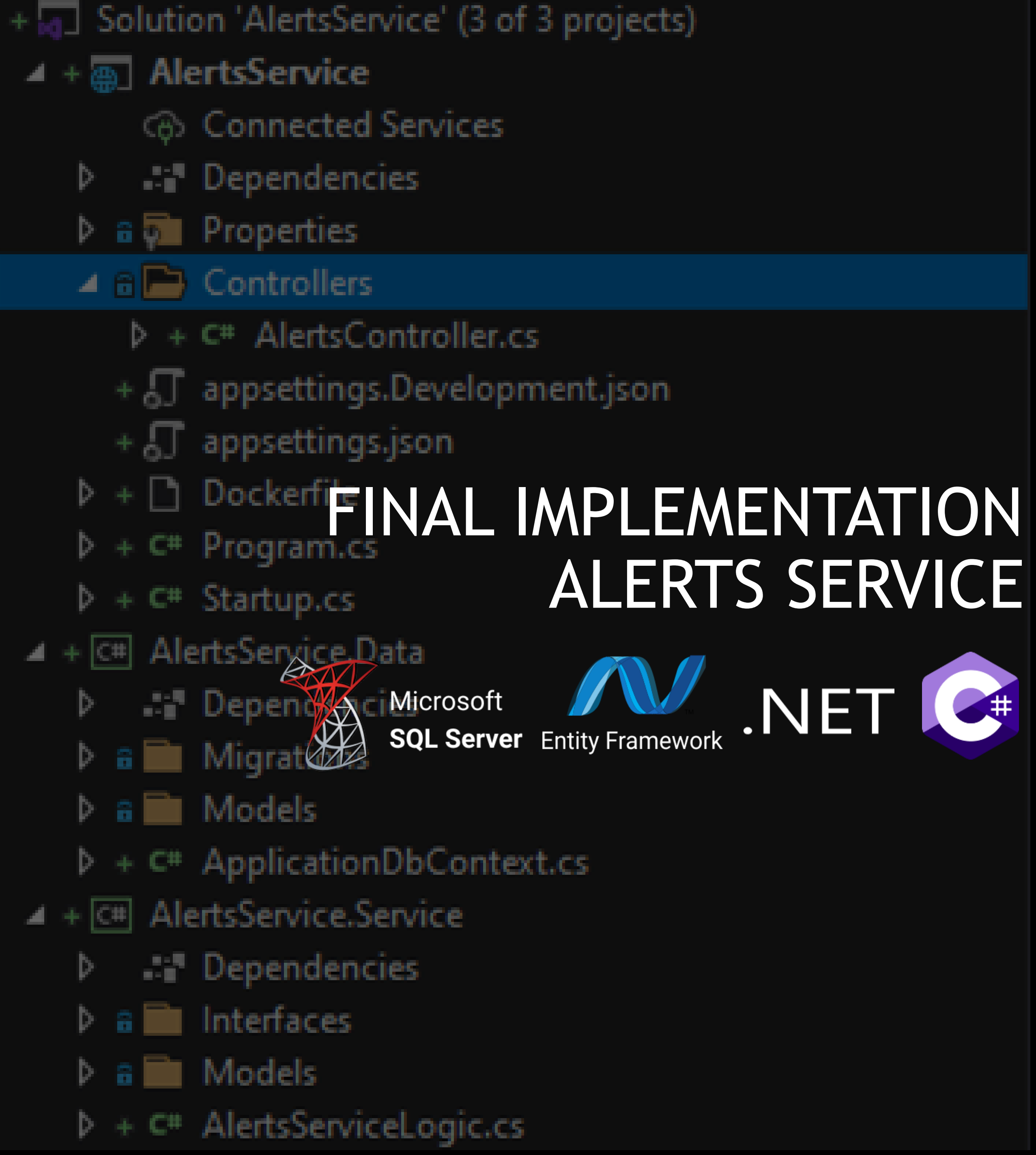
UserLoginInputModel >



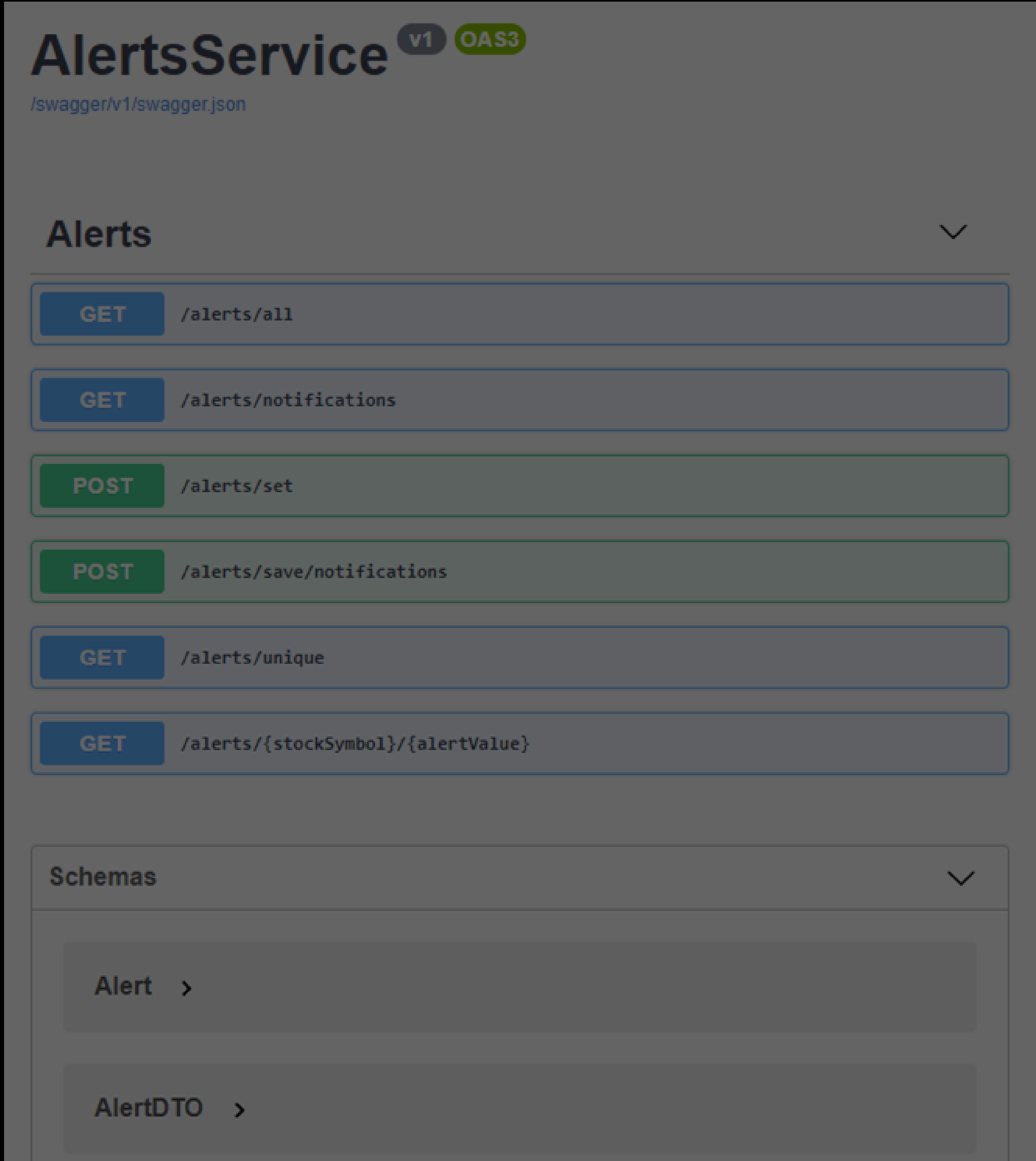
```
Identity.API
    Connected Services
    Dependencies
    Properties
    Controllers
    appsettings.Development.json
    appsettings.json
    Dockerfile
        .dockerignore
    Program.cs
    Startup.cs
Identity.Data
    Dependencies
    Migrations
    Models
    ApplicationDbContext.cs
Identity.Service
    Dependencies
    DTO
    Interfaces
    Models
```

FINAL IMPLEMENTATION
ALERTS SERVICE

Microsoft SQL Server   Entity Framework   .NET



```
Solution 'AlertsService' (3 of 3 projects)
  AlertsService
      Connected Services
      Dependencies
      Properties
      Controllers
        AlertsController.cs
      appsettings.Development.json
      appsettings.json
      Dockerfile
      Program.cs
      Startup.cs
  AlertsService.Data
      Dependencies
      Migrations
      Models
      ApplicationDbContext.cs
  AlertsService.Service
      Dependencies
      Interfaces
      Models
      AlertsServiceLogic.cs
```



AlertsService v1 OAS3

/swagger/v1/swagger.json

Alerts

GET /alerts/all

GET /alerts/notifications

POST /alerts/set

POST /alerts/save/notifications

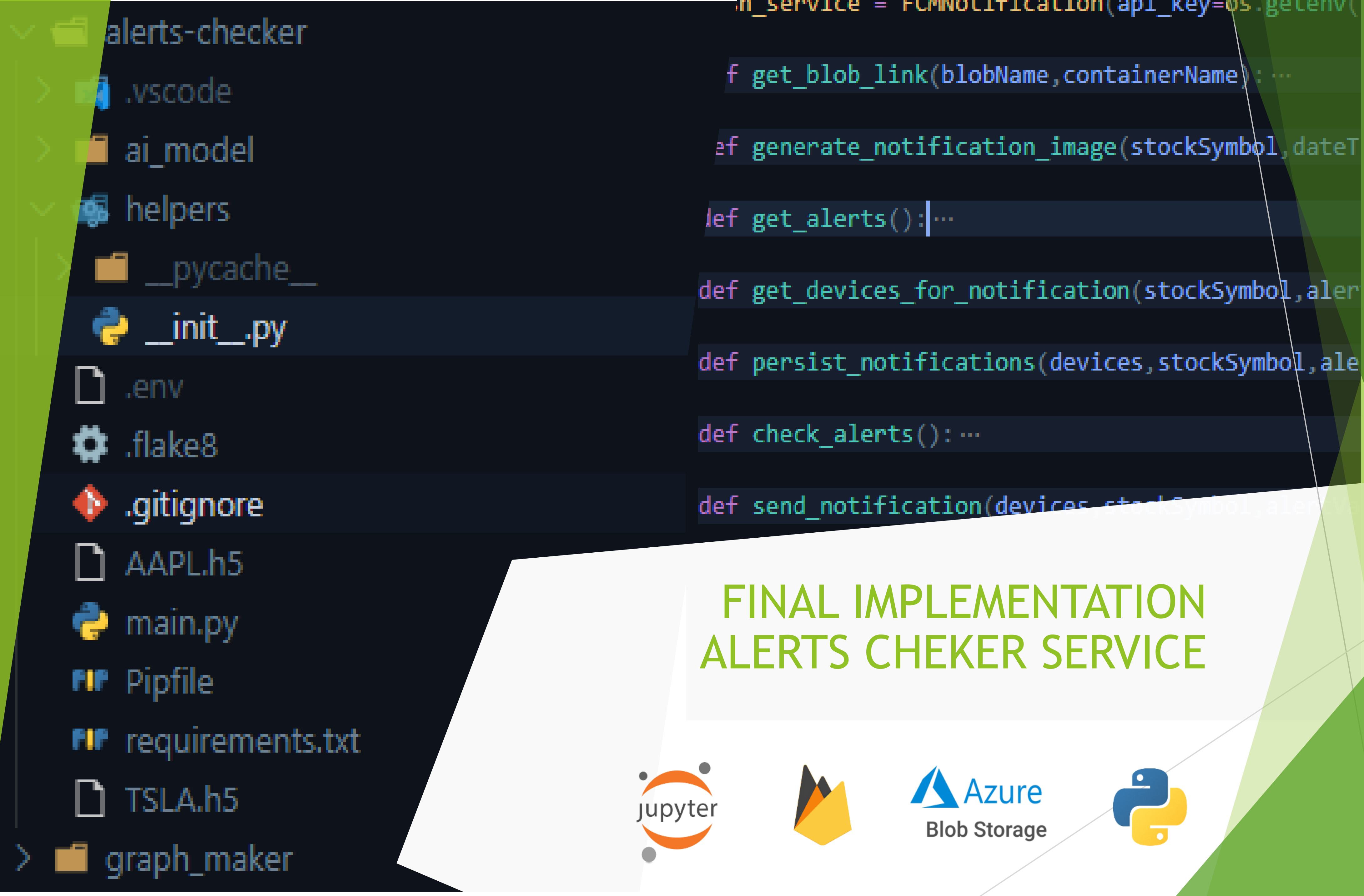
GET /alerts/unique

GET /alerts/{stockSymbol}/{alertValue}

Schemas

Alert >

AlertDTO >



```
alerts-checker
  .vscode
  ai_model
  helpers
    __pycache__
    __init__.py
  .env
  .flake8
  .gitignore
  AAPL.h5
  main.py
  Pipfile
  requirements.txt
  TSLA.h5
graph_maker
```

```
n_service = FCMNotification(api_key=os.getenv(

f get_blob_link(blobName,containerName): ···

ef generate_notification_image(stockSymbol,dateT

lef get_alerts(): ···

def get_devices_for_notification(stockSymbol,aler

def persist_notifications(devices,stockSymbol,ale

def check_alerts(): ···

def send_notification(devices,stockSymbol,alert
```

FINAL IMPLEMENTATION ALERTS CHEKER SERVICE

jupyter

Azure Blob Storage

CHALLENGES

- **MOBILE APPLICATION**
  - Refactoring because of unnecessary socket implementation
  - Integration of notifications - ejecting expo project because expo notifications does not support emulators
  - Initial state management setup

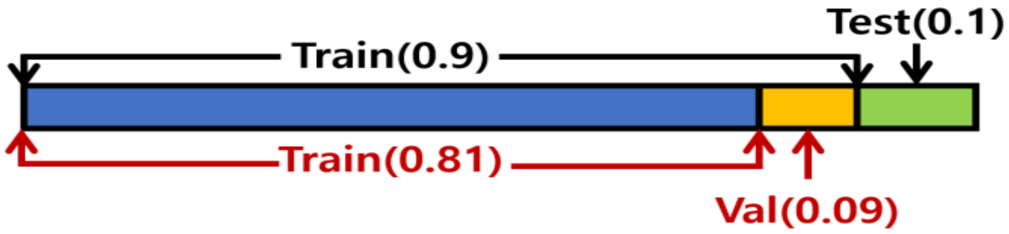
- **ALERTS CHECKER SERVICE**
  - Persisting AI Model heatmap images to external storage provider
  - Sending notifications
  - Performance optimization

- **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.

DATASET

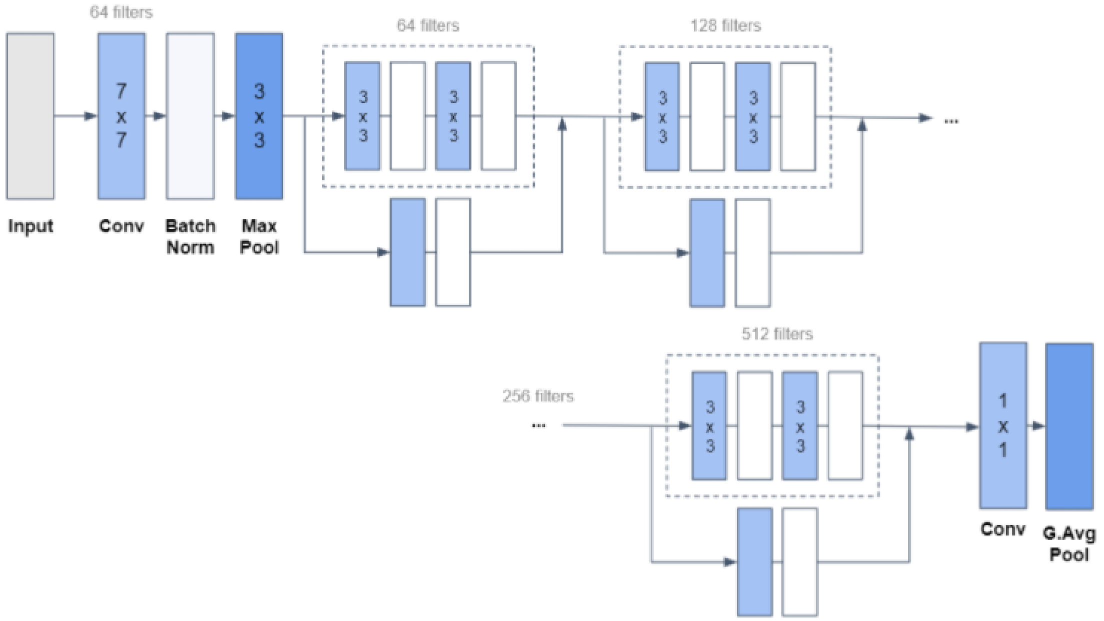
- **YFINANCE**
  - To retrieve stock price information

- **Corpus**
  - The input dataset consisted of 1744 CNNs and LSTMs
  - The ratio of train:validation:test was 0.81: 0.09: 0.1



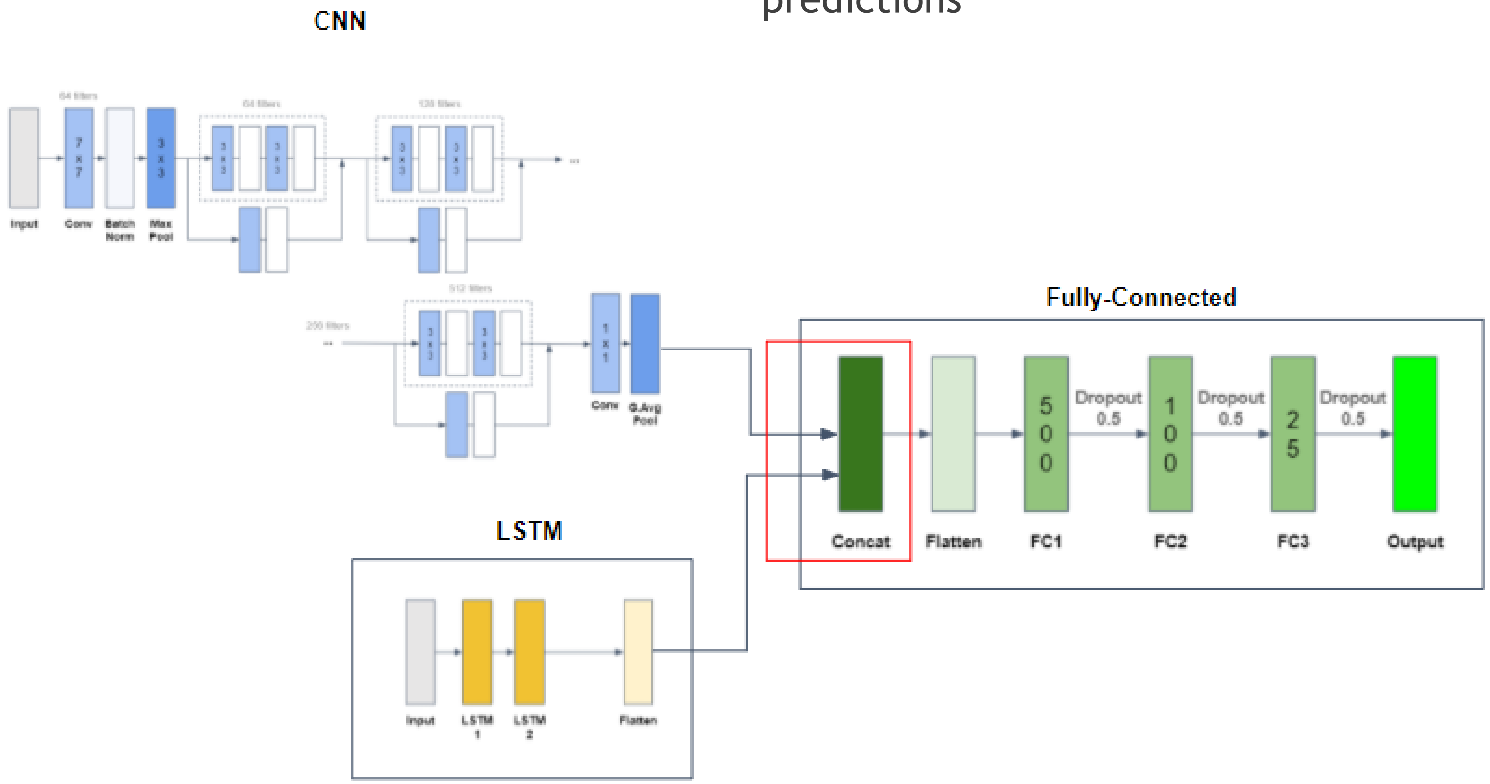
MODEL DESCRIPTION

- **CNN**
  - Resnet structure
  - 1 convolutional layer + 16 Resnet block + 1 convolution layer



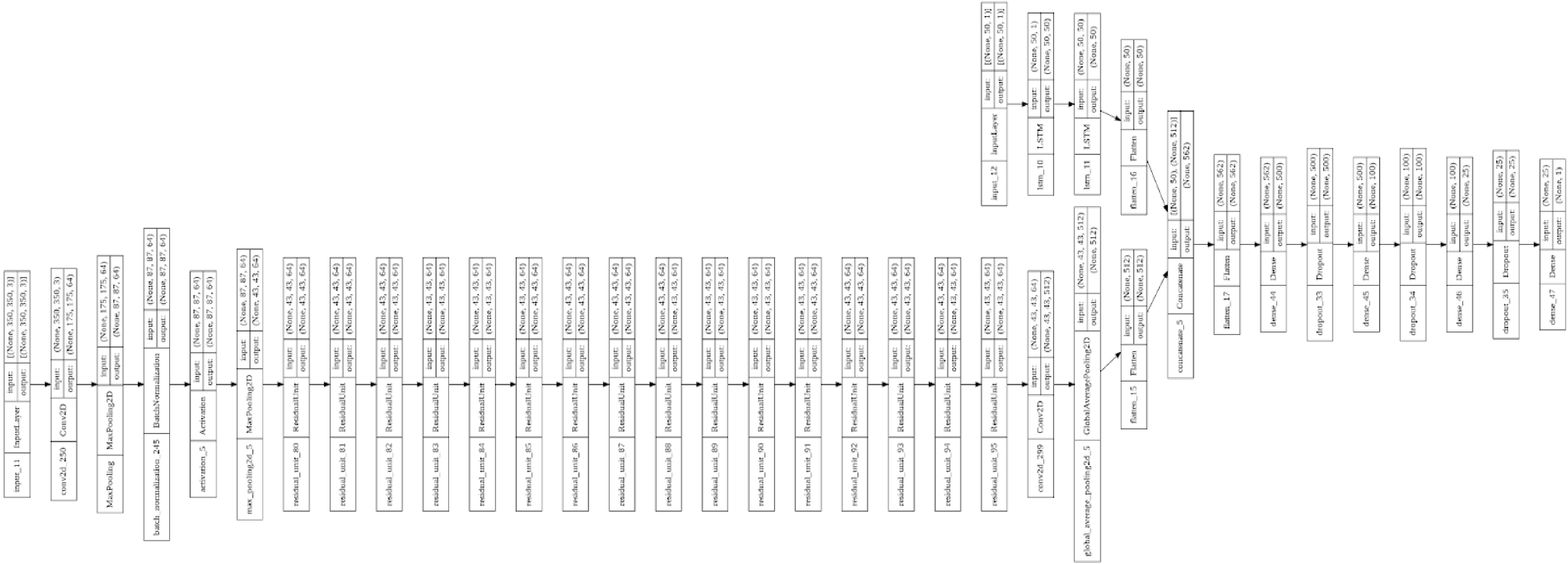
MODEL DESCRIPTION

- **LSTM**
  - 2 Layer
- **Concatenation**
  - CNN and LSTM join FNN through Flattern and produce predictions



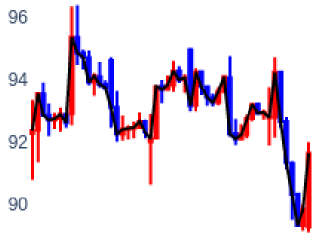
MODEL DESCRIPTION

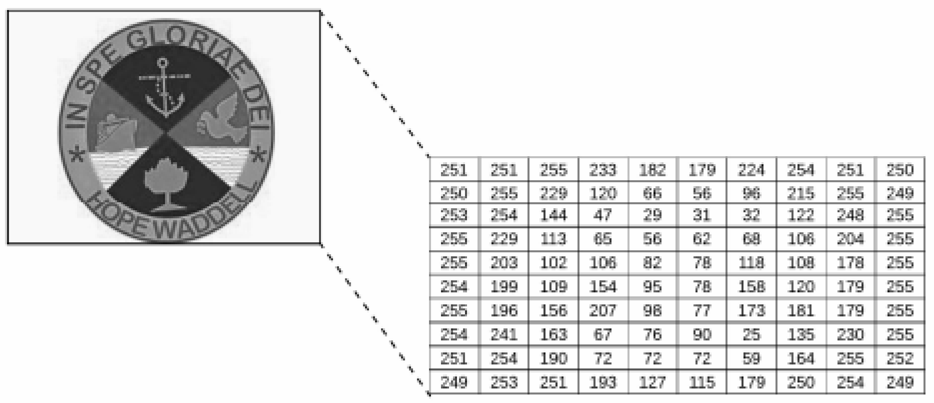
- **Schematic diagram**


input_11 | InputLayer | input: [(None, 350, 350, 3)] | output: [(None, 350, 350, 3)]
conv2d_250 | Conv2D | input: (None, 350, 350, 3) | output: (None, 175, 175, 64)
MaxPooling | MaxPooling2D | input: (None, 175, 175, 64) | output: (None, 87, 87, 64)
batch_normalization_245 | BatchNormalization | input: (None, 87, 87, 64) | output: (None, 87, 87, 64)
activation_5 | Activation | input: (None, 87, 87, 64) | output: (None, 87, 87, 64)
max_pooling2d_5 | MaxPooling2D | input: (None, 87, 87, 64) | output: (None, 43, 43, 64)
residual_unit_80 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_81 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_82 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_83 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_84 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_85 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_86 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_87 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_88 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_89 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_90 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_91 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_92 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_93 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_94 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
residual_unit_95 | ResidualUnit | input: (None, 43, 43, 64) | output: (None, 43, 43, 64)
conv2d_299 | Conv2D | input: (None, 43, 43, 64) | output: (None, 43, 43, 512)
global_average_pooling2d_5 | GlobalAveragePooling2D | input: (None, 43, 43, 512) | output: (None, 512)
flatten_15 | Flatten | input: (None, 512) | output: (None, 512)
input_12 | InputLayer | input: [(None, 50, 1)] | output: [(None, 50, 1)]
lstm_10 | LSTM | input: (None, 50, 1) | output: (None, 50, 50)
lstm_11 | LSTM | input: (None, 50, 50) | output: (None, 50)
flatten_16 | Flatten | input: (None, 50) | output: (None, 50)
concatenate_5 | Concatenate | input: [(None, 50), (None, 512)] | output: (None, 562)
flatten_17 | Flatten | input: (None, 562) | output: (None, 562)
dense_44 | Dense | input: (None, 562) | output: (None, 500)
dropout_33 | Dropout | input: (None, 500) | output: (None, 500)
dense_45 | Dense | input: (None, 500) | output: (None, 100)
dropout_34 | Dropout | input: (None, 100) | output: (None, 100)
dense_46 | Dense | input: (None, 100) | output: (None, 25)
dropout_35 | Dropout | input: (None, 25) | output: (None, 25)
dense_47 | Dense | input: (None, 25) | output: (None, 1)

INPUT AND OUTPUT

- **INPUT**
  - **CNN**
    - Candlestick chart that contains information on close, open, high, low stock prices
    - Stock chart image is converted into a numpy array
  - **LSTM**
    - Closing price data
    - Use log10 value

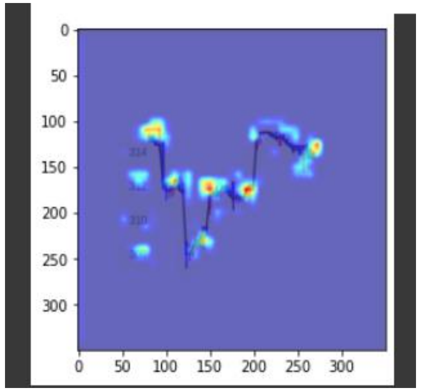




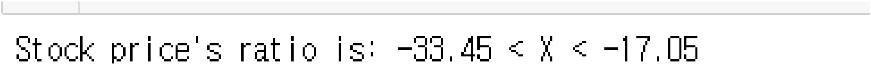
251	251	255	233	182	179	224	254	251	250
250	255	229	120	66	56	96	215	255	249
253	254	144	47	29	31	32	122	248	255
255	229	113	65	56	62	68	106	204	255
255	203	102	106	82	78	118	108	178	255
254	199	109	154	95	78	158	120	179	255
255	196	156	207	98	77	173	181	179	255
254	241	163	67	76	90	25	135	230	255
251	254	190	72	72	72	59	164	255	252
249	253	251	193	127	115	179	250	254	249

INPUT AND OUTPUT

- **OUTPUT**
  - Gradient CAM - Heat Map



  - Processing about log10 and correction are performed on the finally derived prediction



```
Stock price's ratio is: -33.45 < X < -17.05
```

EVALUATION METRICS

- **Evaluation Metrics for ML**
  - Epoch: 20~30
  - loss(mse): 0.1524
  - mape: 11.9957
  - rmse: 0.3904

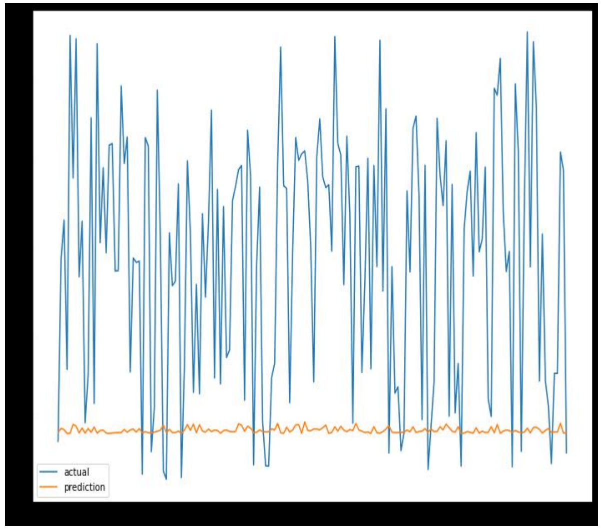
  - val_loss: 0.0050
  - val_mape: 2.3413
  - val_rmse: 0.0706

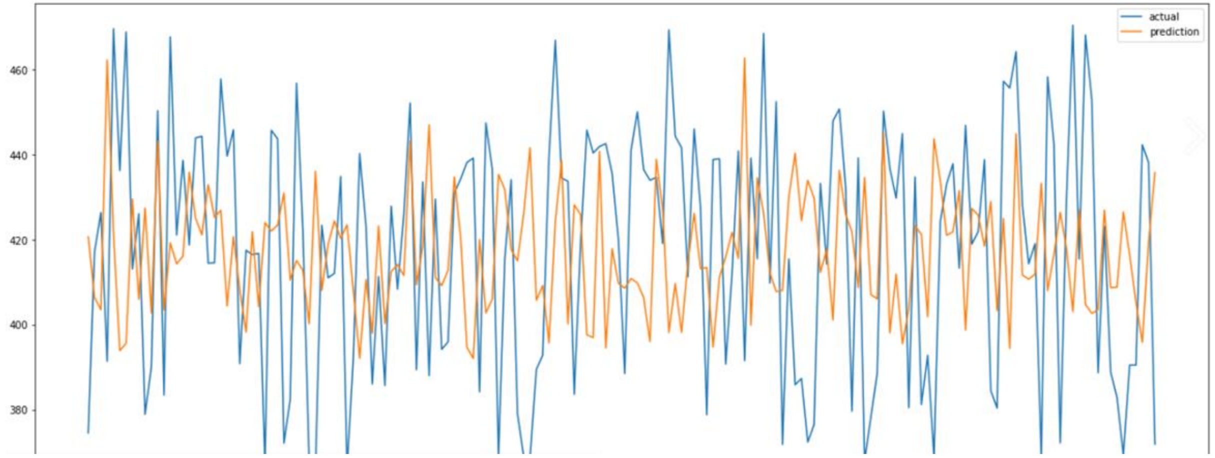
MAIN HYPERPARAMETERS

- **Batch size**
  - 32
- **Epoch**
  - It was set to 26 due to overfitting and underfitting
- **Value correction**
  - When the forecast is low
    - Multiply by gap_avg * (1+Standard Deviation)
  - When the forecast is higher
    - Devide by gap_avg * (1-Standard Deviation)

MAIN HYPERPARAMETERS

- **Result of value correction**





LIMITATIONS

- **MOBILE APPLICATION**
  - Not entirely native
- **ALERTS CHECKER SERVICE**
  - Each check must wait the previous to finish, and it can be slowly sometimes if there are many users subscribed to the same alert because the sending of notifications is included in the service
- **AI MODEL**
  - Overfitting & underfitting case
  - Approximate and consistent errors between stock price predictions and actual values

EVALUATION

- **Meet the objective of the project**
  - Application users can set alerts and receive notifications for set stock items

- **Reasons for low predictive rate**
  - Uncertain fluctuations in stock prices
  - Difference in layer depth between CNN and LSTM
  - Fast overfitting due to sequential input dataset
  - Low learning rate due to small epoch value
  - 1 hour data used in train process, but 15 minute data used in evaluate

DEMO

1398

Q & A