A System for User Friendly Stock Price Prediction and Sentiment Analysis [STOCK]

Contents

Figures

Tables

1 OVERVIEW

This document contains the Software Project Specification for the **Stock Price Prediction and Sentiment Analysis System.** The members of the team are as follows:

Harun Ćerim	**Backend** – Web Services
Ilda Balliu	**Frontend** – Web Design and help with Web Services
Kaan Yöş	**Backend** – Machine Learning
Mahran Emeiri	**Frontend** – Web Design and help with Machine Learning

Table 1 – Team Members and Responsibilities

The team members already have experience with software development, including building front-end and back-end applications. Taking this into account, we decided to implement a system in which we could integrate these skills and also learn more about machine learning, sentiment analysis and the stock market.

The main purpose of this project is to create a system, served to the users as a web application, through which they will be able to get information about stock prices of companies they choose. System provides an easy access to:

- Company data (information & stock price),
- Articles published about the company,
- Predicted future stock prices,

Most importantly, STOCK Application allows users to create their custom prediction models and forecast stock prices based on their own adjustments. After the custom model is created, the application will train the model and notify the user once the resulting report is ready. Additionally, users will be able to compare their custom models with each other and export the results to a PDF document.

Note that we are aware of other similar software services in the market, such as the Wallet Investor [1]. STOCK Application significantly differs from other services because it provides users with the opportunity to create their custom prediction models and tweak them according to their needs.

In general, this system will be useful for individuals/companies that would like to adjust important parameters of stock prediction and current/future investors who want to track companies' progress in the stock market.

The web application will offer the following functionalities:

- searching for a specific company that the user is interested in,
- getting an overview of past and current stock prices of this company (their graph visualizations),

[1] https://walletinvestor.com/

- predicting the estimated future stock price based on user-defined parameters (e.g. time period, prediction variables, kernel selection),
- positive and negative classification of the articles about companies.

STOCK Application is planned to be modular and scalable. If a company which a user is looking for is still not in the database, the user must contact the administrator directly from the web application and ask the administrators to add it. The administrator will then perform all the necessary steps (adding company to the database, training of the machine learning modules etc.). Those steps do not require implementing a new module into the system. The administrator will simply add a new company name into existing company list and the system will automatically gather information regarding the new company. After the new company is added and is ready to be viewed and used by the user, they will be notified in the application UI and via email about it.

Additionally, the modularity will enable future extensions of the selected parts of the project e.g., within Master theses of some of the participating students.

2 SYSTEM UNITS

The system will be divided into smaller implementation modules as depicted in Figure 1.

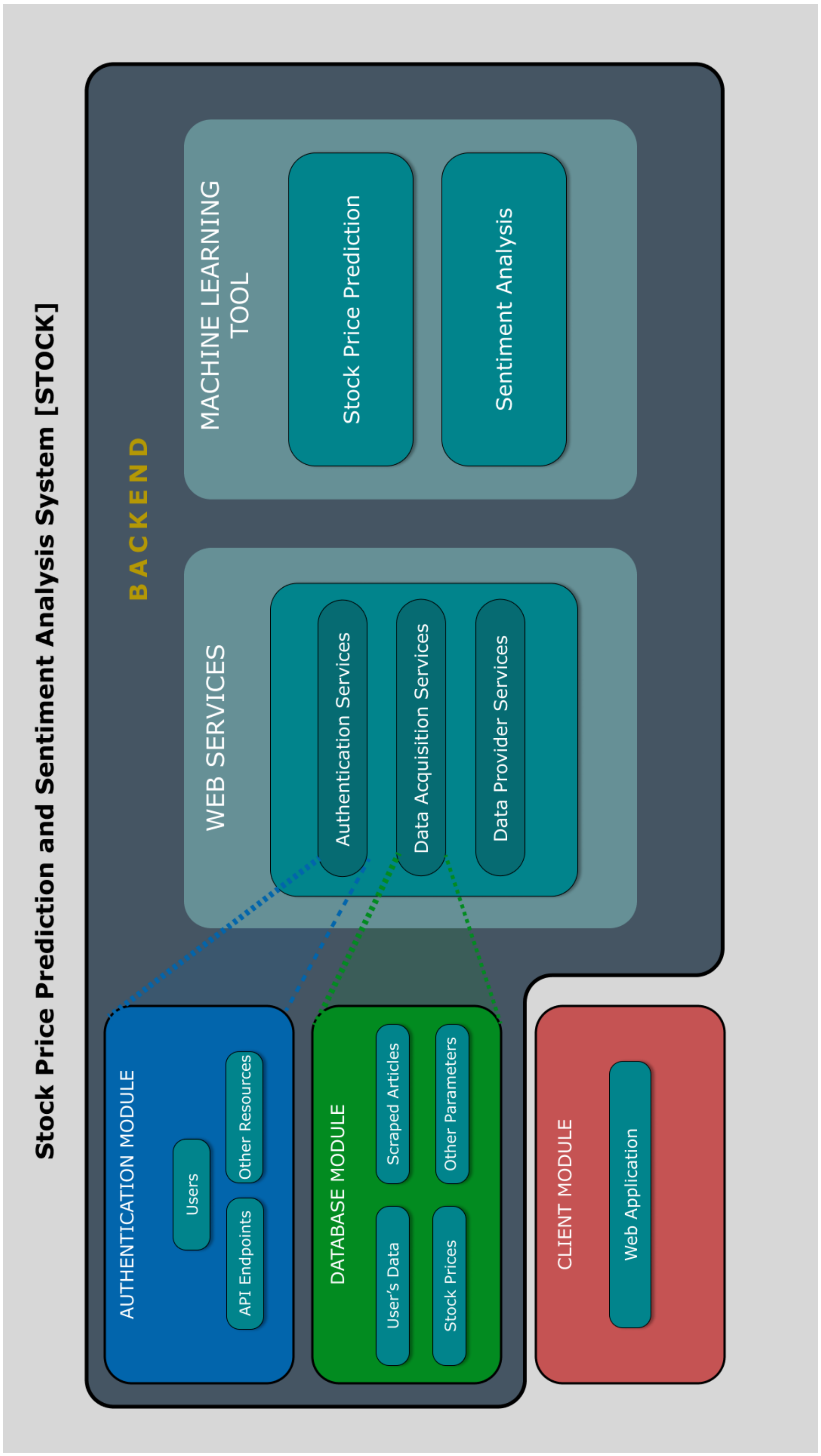


Figure 1 – Stock Price and Sentiment Analysis System Decomposition View

2.1 Client Module

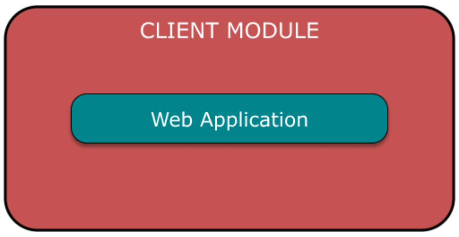


Figure 2 – Client Module Diagram

An interface of the system and for the initial version the system will focus on the web application. It is going to have all functionalities of the system available to the end user.

2.2 Authentication Server Module

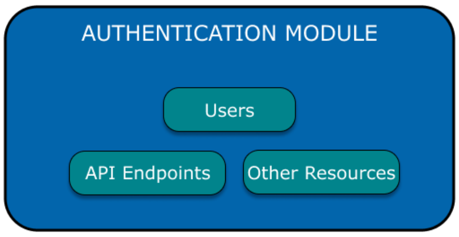


Figure 3 – Authentication Module Diagram

An individual module will be responsible for keeping resources, users, and APIs secure. The main purpose of this module is to provide token issuing functionality where the end user will get access tokens to use other services provided through the interface.

2.3 Database Module

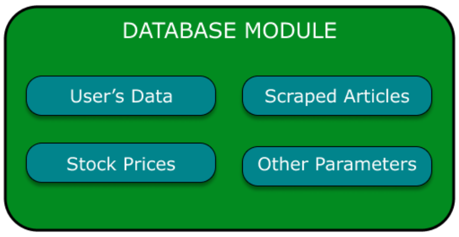


Figure 4 – Database Module Diagram

We will keep all the information necessary for the System Back End module. There will be stored user's data, financial articles, stock prices, companies' information, custom machine learning models that the user has created and other parameters. All the data will be stored in a structured way which means stored by the specification, so it can be easily used by the back-end services and machine learning (ML) tools.

2.4 Machine Learning Tool Module

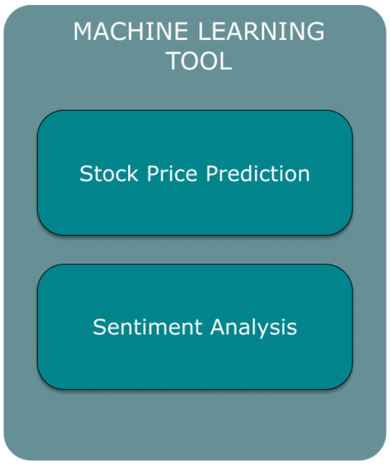


Figure 5 – Machine Learning Tool Diagram

The core part of the system will be responsible for predicting stock prices and classification of articles which will help the end user in deciding whether some stocks are good to be bought or sold depending on the current stock price, the predicted stock price for a specified date, the average price from a specified period, etc. Classified articles will help the user to check, e.g., whether financial analysts have said something about the company of users' interest. Stock price prediction tool will focus on predicting the stock prices and training users' custom prediction models. Sentiment Analysis tool will focus on classifying articles from analysts in the financial world as, e.g., positive, negative.

2.5 Web Services Module

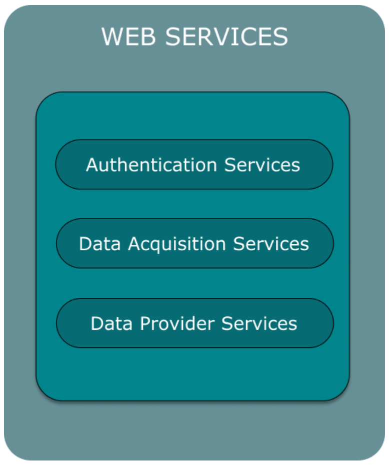


Figure 6 – Web Services Diagram

This module is going to deal with providing back-end logic and functionality to front-end application and to ML Tool.

3 HIGH-END FUNCTIONALITIES

The historical data will be obtained from financial API and a custom scraper developed by the team that will get current information from relevant websites for a sufficient time interval. The data will be transformed and stored in a predefined structured way to be used by the prediction tool and for information visualization in the client application. The prediction tool and sentiment analysis tool will consist of algorithms that will be continuously trained by the team during development and new information gathering.

As depicted in Figure 2, the end application will have the following functionalities:

- **Authentication and authorization**
  This involves the registration, logging in/out.
- **Searching, searching history**
  Searching capability for a specified company/set of companies using selected criteria and parameters. Ability to see the previously searched companies/parameters.
- **Custom prediction models**
  Ability to create prediction models, customize variables and parameters, and compare models.
- **Check predicted prices**
  Check the predicted price for a specified time and company using visual graph representations.
- **History of prices for a specific period using visual graph representations**
  Ability to see the prices in the past, the trends etc., using visual graph representations.
- **Showing the prices of user's saved companies**
  The application will enable users to save their favorite companies and see them right after logging in, without the need to search again.
- **Showing the related articles and respective sentiment**
  Users will be able to see new analyzed articles for a specific company and to navigate to the source from which these articles are acquired.

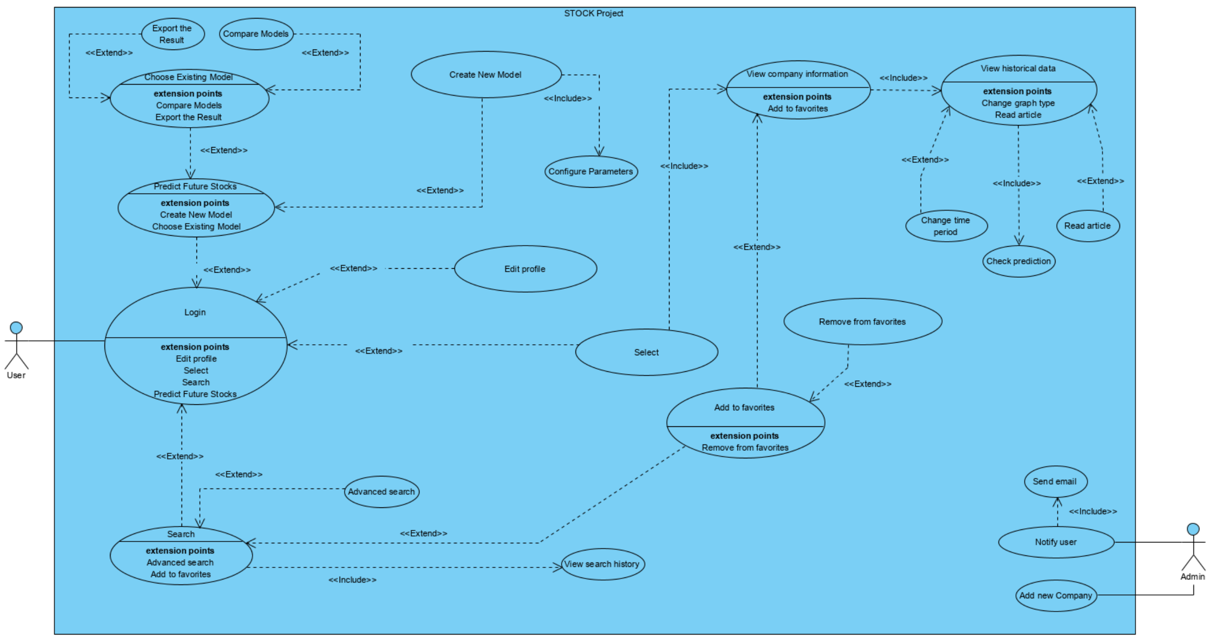


Figure 7 – STOCK Project Use Case Diagram Example

3.1 Use Case Example

The user first needs to register and then log in to the system. From that point, the user will be able to modify his/her profile, specify favorite companies that will be available in the dashboard and see the search history. The user will be able to search for a specific company and see its details including; stock prices, latest analyzed articles and choose the date of the prediction. User can create custom Regression Models to predict stock prices, compare the results of his/her custom models and export the results. The user can also request a new company to be added to the dataset.

4 SOFTWARE MODULES

In this part, there will be described technologies that are going to be used to develop this system.

4.1 Web Services

Web services consist of backend and core services of the web application. The back-end is going to be developed in .NET Core using C# and will be deployed as Lambda application in Amazon Web Services (AWS). For long performing tasks, AWS EC2 will be used as a virtual machine. Data Acquisition and Data Provider Services which belong to Core Services will be developed using .NET Core and C#. Public APIs, such as IEX, will be used to gather historical data about the stock prices and articles from Market Watch. Also, the Python scripts and models will be made available as http services to be provided to the client application in order to show the user models that have already been created and they can process them further.

Basically, all these services will be developed as Rest APIs that will either function as serverless functions or normal server functions depending on the duration of their execution. As the limit of AWS Web API is 30 seconds long running tasks like scraping will be running on EC2 which is a virtual machine in AWS cloud.

4.2 Machine Learning Tool

4.2.1 Stock Price Prediction

Stock price prediction is a binary classification problem which tries to find an answer if given stock price will be lower or higher in the future. However, we will implement forecasting algorithms that show what will be the closing price of specific company in the future.

Firstly, the system will predict future stocks by using Recurrent Neural Network (RNN) and show the result in the company page. We will use TensorFlow [1] and Keras [2] to create and train the RNN. This prediction will be done automatically in the back-end and shown to the user as a default prediction. Besides RNN, we will use SkLearn [3] Library to implement different Regression Models (Ridge, Lasso etc. [4]) for user specific models and necessary data normalization.

STOCK Application will give users the opportunity of creating their custom RNN and Regression Predictors and predict stock prices based on their adjustments. Since stock price predictions depends on multiple variables, RNN and Regression Models will be implemented with adjustable parameters (prediction variables, time interval, learning rate, penalty rate, hidden layers etc.) in a way that the user can tweak them to create custom models.

Additionally, Pandas [5] will be used for analyzing data and matplotlib [6] will be used for necessary plotting.

All in all, while providing default predictions on company page, STOCK Application will also allow users to create custom Regression and RNN Models, adjust prediction parameters and predict stock prices based on their criteria.

4.2.1.1 Model Customization

The user will be able to tweak the learning and hyper parameters such as random variables, evaluation metrics, penalty rate, epoch, learning rate etc. and train their custom Regression or RNN Models. In order to do that, the user will be given a customize page to adjust parameters based on their preferences. After customization is done and user presses the create button, the system will train the users' model. Training custom models will be handled in the same way of default prediction models. Back-End will be working with the same structure but different learning parameters. Once the training is done, the user will get notified.

Also, since our resources are limited, we will add a queuing system to adjust resource allocation for training new models.

4.2.2 Sentiment Analysis

Financial articles will be classified as positive and negative depending on their sentiment scores. We will show the classification score to users for the companies that they are checking and help them to understand if financial articles are saying positive/negative opinions about the company.

In order to do that, we will use TextBlob [7] and VADER [8] Libraries since they are providing good results for polarity classification. NLTK [9] will be used for necessary text pre-processing.

We will create a neural network that is financial domain specific. This includes several steps as described below:

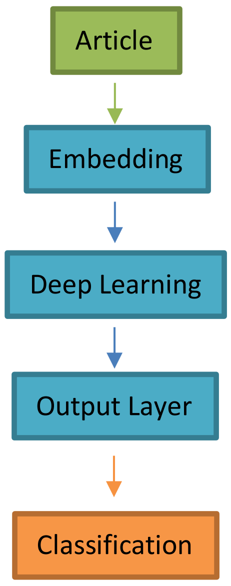


Figure 8 – Classification Steps

Scraped articles will be used to feed the network. After doing the pre-processing and classifying our articles, we will use TensorFlow and Keras to create our own neural network.

Deep Learning will take the sequence of embedded word vectors. We will use Recurrent Neural Network (RNN) with Long Short-Term Short Memory (LSTM) for this part.

Finally, we will train the sentiment classification model.

The aim of creating a custom neural network is that our system will be based on only financial articles, not like other networks or libraries that are trained by general articles.

4.3 *Database*

For the database, DynamoDB [10] will be used. DynamoDB as a managed cloud NoSQL database can provide flexibility and scalability that we are going to need as we will store all the information related to companies, articles, news, stocks, customers, etc. there. Since DynamoDB is scalable and also schema-less, there will be one table that will represent our database. All items stored in the database will have primary partition key as Id as well as primary sort key as Type for faster and easier querying. AWS SDK will be used as a toolkit to work with all AWS services so DynamoDB libraries for .NET will be used to work with DynamoDB.

4.4 *Authentication Server*

We will be using Amazon Cognito [11] and its User Pools as the authorization server. Cognito libraries for .NET will be used to work directly with Cognito. All information related to customers authentication will be stored directly there but for backup in DynamoDB as well. Cognito User Pools work on token principle and verification codes. Basically, when there is an action that required authenticated user, token is used to verify whether certain user has authority to perform that action. Verification codes will be used via email to verify certain actions like deletion of account or changing the password.

4.5 *Client Application*

The team will build a client application for the user to be able to access the data in an easy way. It will follow the admin platform approach for building Single Page Applications (SPA). The application will be built with JavaScript and the React Material Design Framework [12].

Firstly, the user will log in to the system or register if he does not have an account yet. These two scenarios will be similar to the views below.

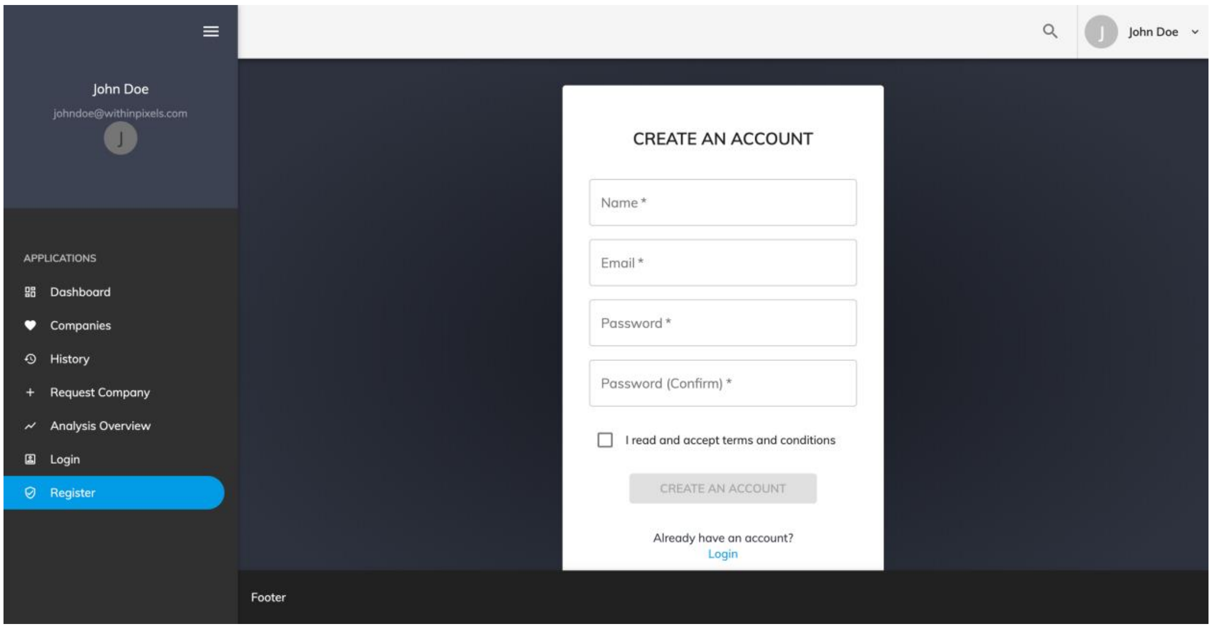


Figure 9 – Create Account Page

To create an account, we will ask for information such as full name, email address and password. However, if the user chooses not to register and use the platform without logging in, he will still be able to see pages such as the Dashboard, Companies and the Company profile.

If the user already has an account, then he can log in by using email and password which was set previously. Also, the user will have the ‘Remember me’ and ‘Forgot password?’ options for easier login experience.

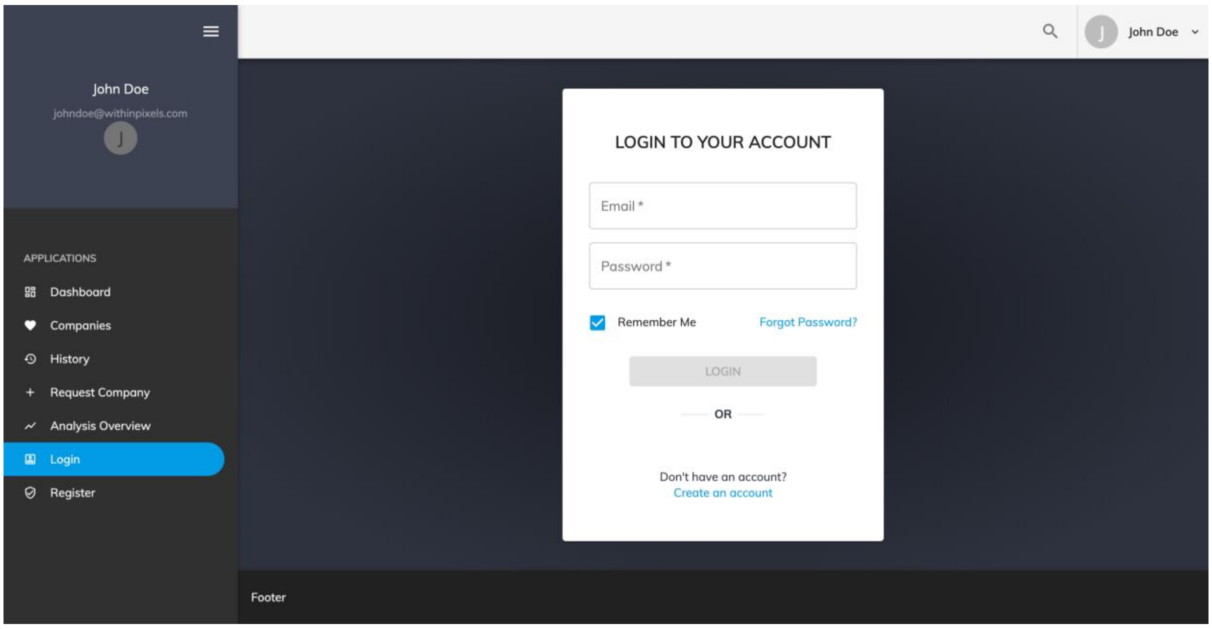
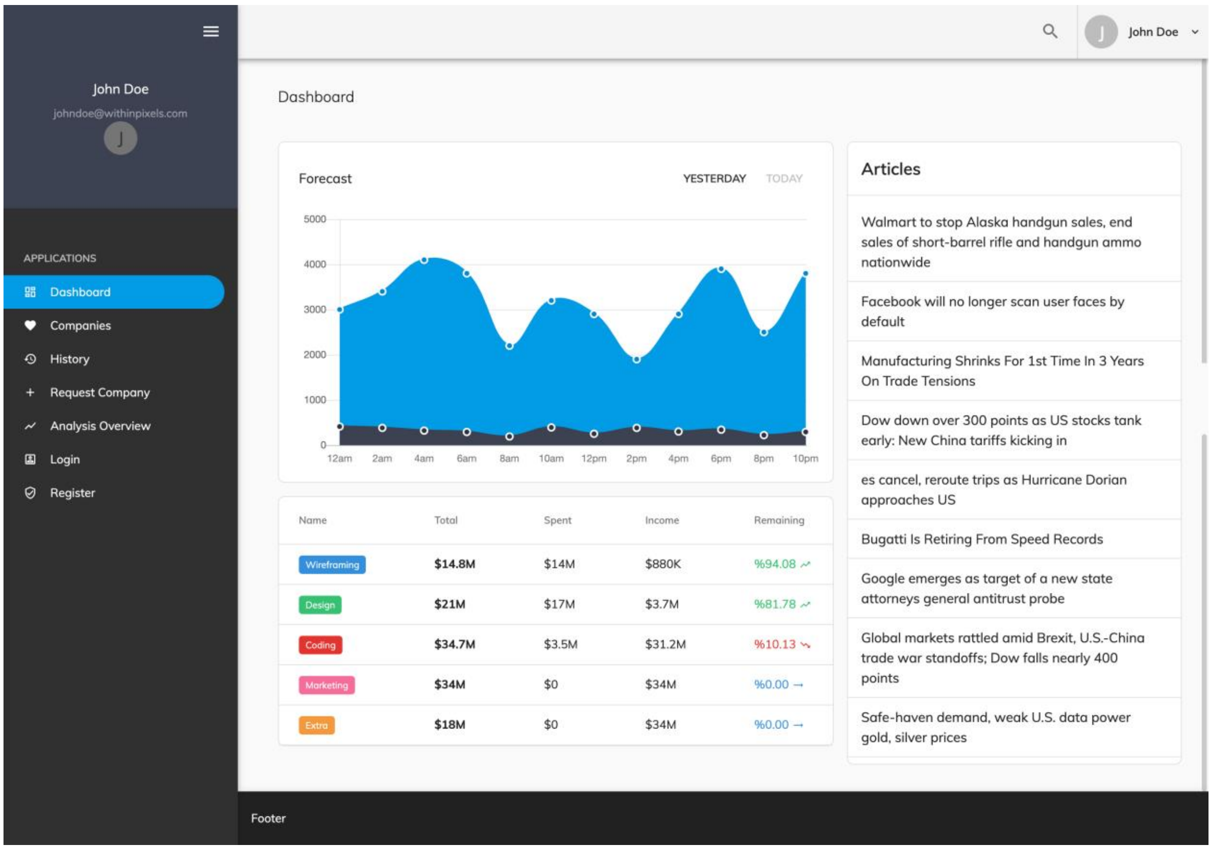


Figure 10 – Log in Page

However, if the user chooses not to log in or register, they will still be able to see the main page which in this case is the Dashboard, where the general information about stock prices are shown and also related articles but they will not be able to create custom prediction models. Here is an example of what the dashboard will look like after the user logs in:



Name	Total	Spent	Income	Remaining
Wireframing	$14.8M	$14M	$880K	%94.08
Design	$21M	$17M	$3.7M	%81.78
Coding	$34.7M	$3.5M	$31.2M	%10.13
Marketing	$34M	$0	$34M	%0.00 →
Extra	$18M	$0	$34M	%0.00 →

Articles

- Walmart to stop Alaska handgun sales, end sales of short-barrel rifle and handgun ammo nationwide
- Facebook will no longer scan user faces by default
- Manufacturing Shrinks For 1st Time In 3 Years On Trade Tensions
- Dow down over 300 points as US stocks tank early: New China tariffs kicking in
- es cancel, reroute trips as Hurricane Dorian approaches US
- Bugatti Is Retiring From Speed Records
- Google emerges as target of a new state attorneys general antitrust probe
- Global markets rattled amid Brexit, U.S.-China trade war standoffs; Dow falls nearly 400 points
- Safe-haven demand, weak U.S. data power gold, silver prices

Figure 11 – Dashboard Design

On the left sidebar, the user will be able to see a list of companies that are available to check stock prices for and also check their view and search history. These will be explained in detail later on. The main part of the dashboard will show a chart of stock prices for a set period of time for companies specified by the user. If the user has not logged in or has not specified any companies, there will be shown top 5 companies based on their current prices. Below the chart, all this data will be shown in tabular way, so the user will be able to see the current price, the previous price difference and the prediction of the price for each company on the list. On the right, they will be able to see the latest articles related to these companies which will be scraped by the services in the back end. Also, there will be a search bar implemented, in which the user will be able to search for a company that they want to see.

Below is shown how the user profile looks like. It is separated into two main parts. First, user's general information, including full name, email and phone number if he chooses to add it where he can update personal details as needed. Second, account information where the user can change the password or delete the account definitely. When the account is deleted, all his records are discarded, and this action cannot be reverted.

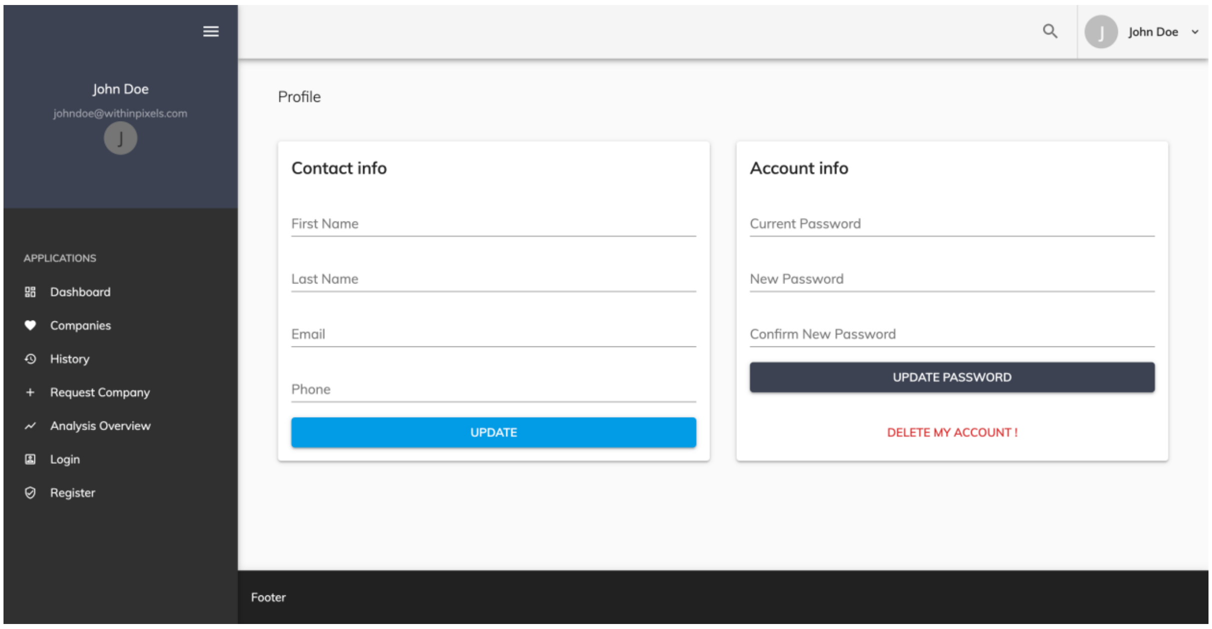
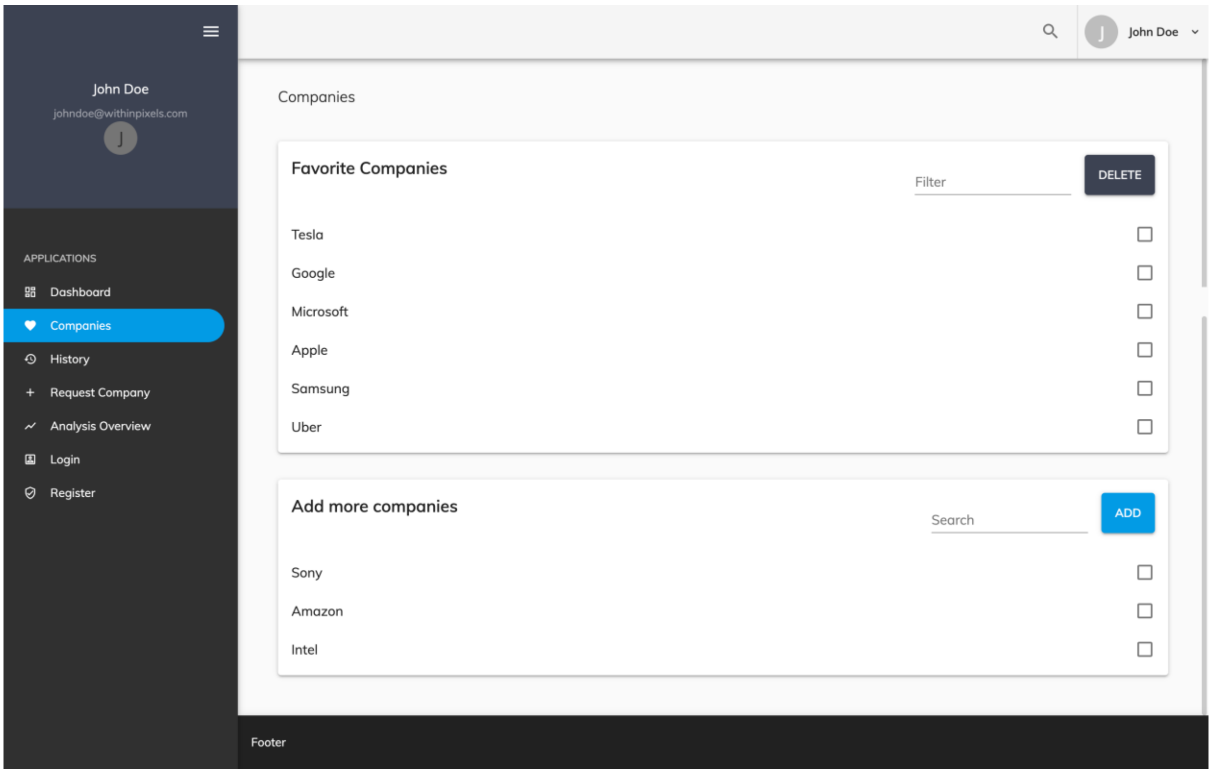


Figure 12 – Profile Page

Going further, the user will be able to go to Companies page. Initially it will be displayed as a list of companies which the user already added to his favorites and another list from which the user can search for new companies and add them to the list. The user can also remove existing companies from the list. Brief data will be displayed for each company, such as name, current price etc. On both lists, users will be able to search for companies, sort the results and modify them.



Companies

Favorite Companies

Filter

DELETE

- Tesla
- Google
- Microsoft
- Apple
- Samsung
- Uber

Add more companies

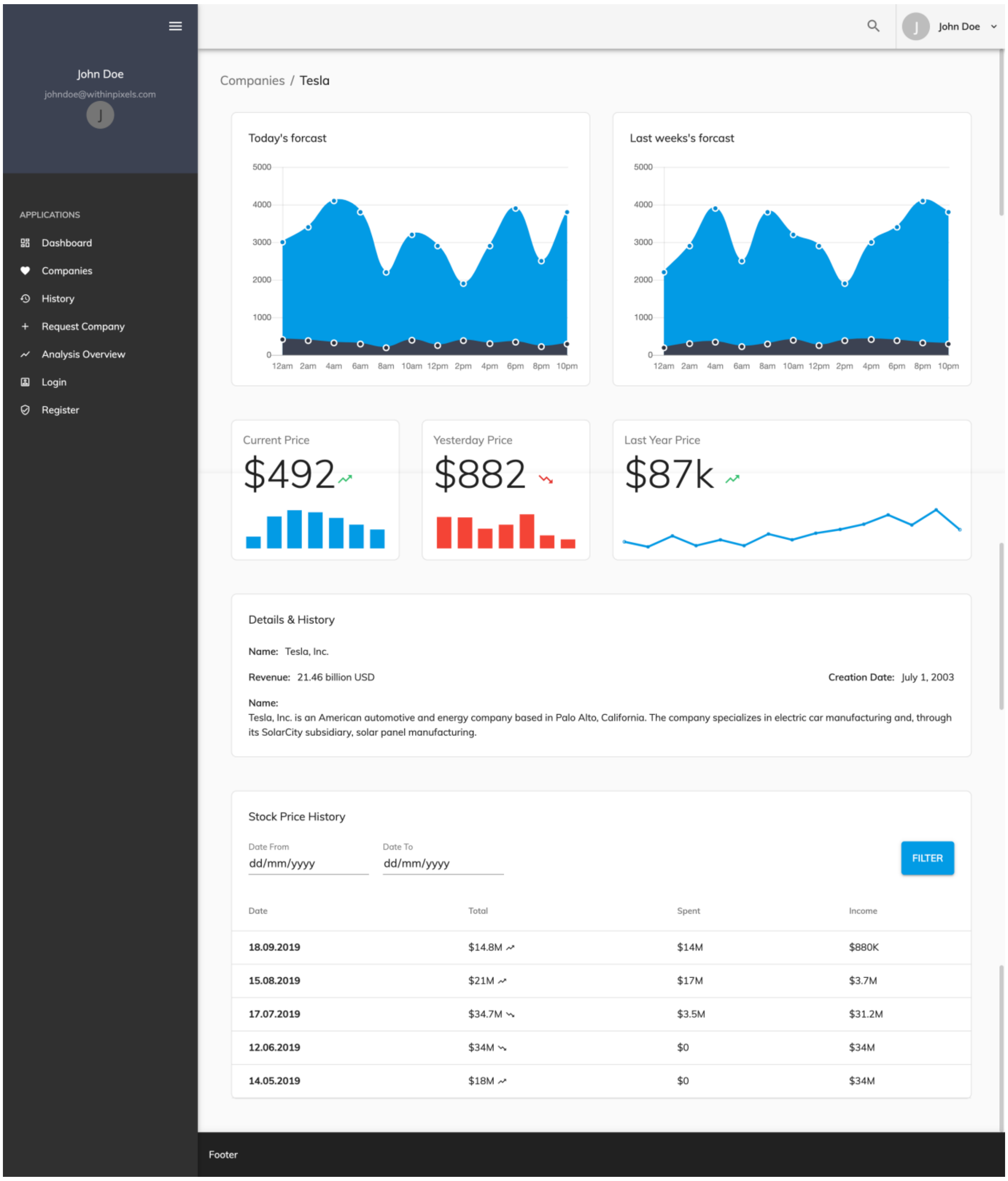
Search

ADD

- Sony
- Amazon
- Intel

Figure 13 – Companies Page

If the user wants to see details of one company, he can click on the name to be redirected to the profile of that company within the platform. In this interface all details regarding each company will be displayed. An example is below.



Details & History

Name: Tesla, Inc.

Revenue: 21.46 billion USD

Creation Date: July 1, 2003

Name:
Tesla, Inc. is an American automotive and energy company based in Palo Alto, California. The company specializes in electric car manufacturing and, through its SolarCity subsidiary, solar panel manufacturing.

Stock Price History

Date From: dd/mm/yyyy

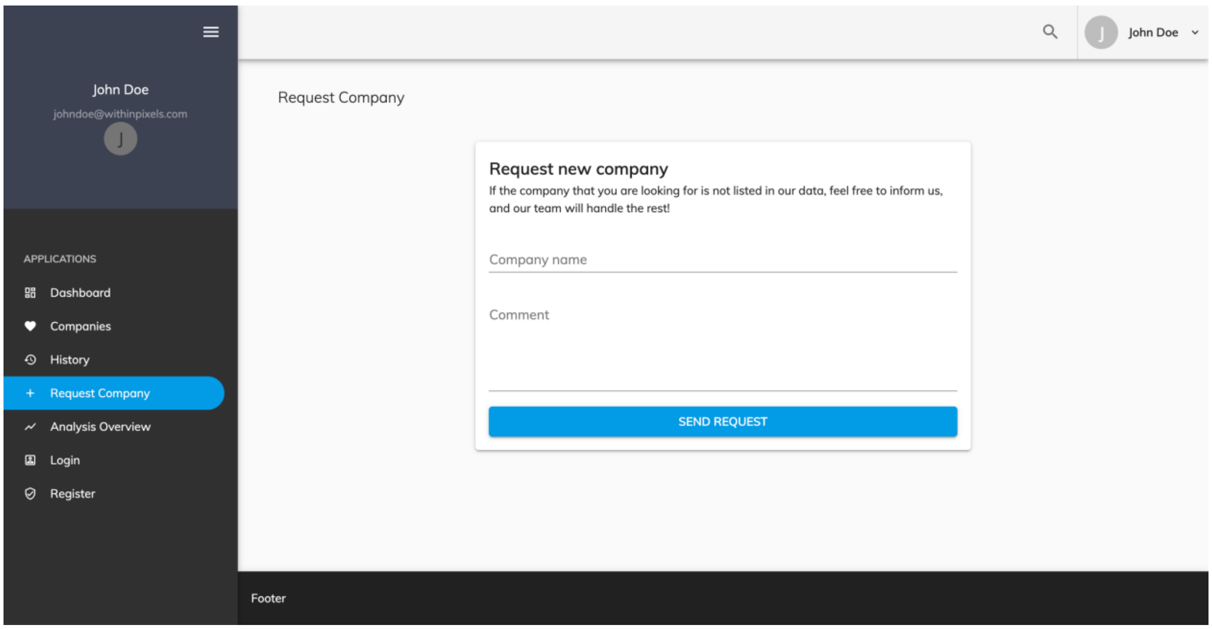
Date To: dd/mm/yyyy

Date	Total	Spent	Income
18.09.2019	$14.8M	$14M	$880K
15.08.2019	$21M	$17M	$3.7M
17.07.2019	$34.7M	$3.5M	$31.2M
12.06.2019	$34M	$0	$34M
14.05.2019	$18M	$0	$34M

Figure 14 – Company Profile Page

The company profile will be divided into a few sections. First, there will be charts of today's forecast and last week's forecast, correspondingly displaying data hourly and daily. Below that will be the current price, 1 year forecasted price and the next 7 days forecast which will be highlighted into separated sections. Scrolling down, the user will also be able to see details regarding the history of the company, when it was founded, by whom, the area in which it operates, revenues etc. like a short bio for each company. The last section will include an interactive table in which the user can specify the time period for which they want to check the stock prices, the trend (up/down), difference in prices, percentage and so on. The user will also be able to sort the results by values, high to low and vice versa. By default, the results will be sorted by date. There could also be an option for the user to export this data into a .csv or .pdf file ready to print.

If the user wants to check prices for a company that is currently not a part of the data set, then he will be able to request this company so the administrator can take the necessary measures and add it. That will be done from the Request Company page which looks like below:



Request new company

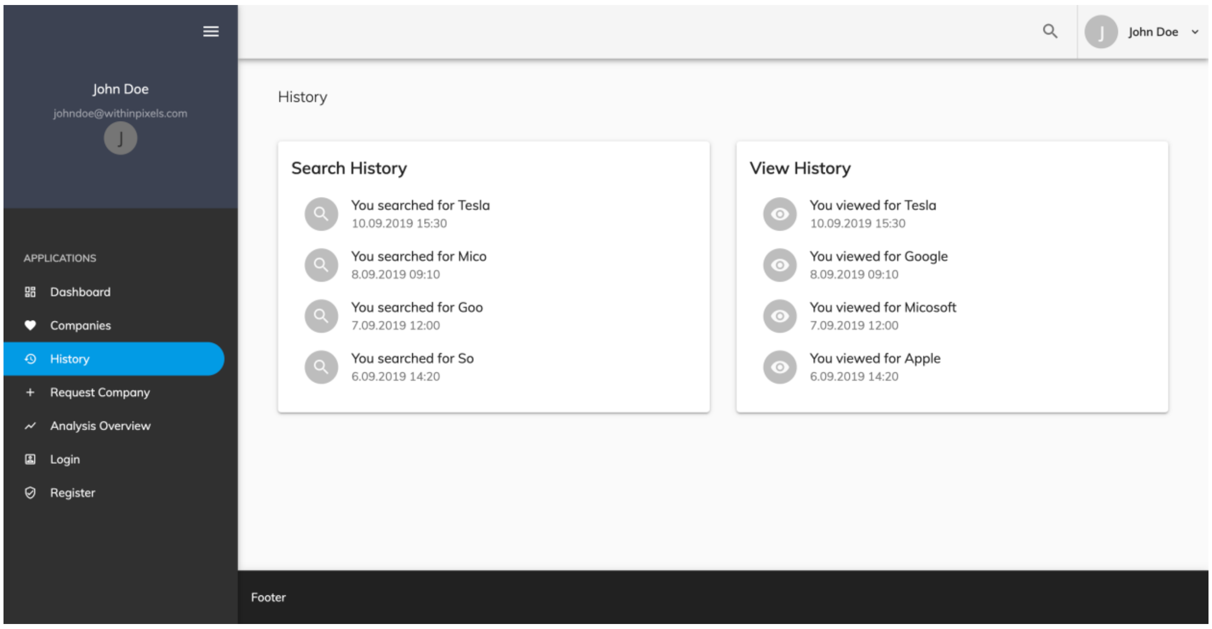
If the company that you are looking for is not listed in our data, feel free to inform us, and our team will handle the rest!

Figure 15 – Request Company

Once the administrator receives the request, he will add the company details into the scraping script, then after having enough data about the company, the chosen prediction model will be trained with the company data. No other actions will be required from administrator since the web service is handling data access through the UI.

The user will receive a notification within the application UI and also via email once the process of adding the new company has finished and is ready for the user to see it and use it as necessary.

The user will also be able to see the companies that he recently searched for or viewed, displayed accordingly in two sections as below.


John Doe

johndoe@withinpixels.com

APPLICATIONS

- Dashboard
- Companies
- History
- Request Company
- Analysis Overview
- Login
- Register

History

Search History

- You searched for Tesla 10.09.2019 15:30
- You searched for Mico 8.09.2019 09:10
- You searched for Goo 7.09.2019 12:00
- You searched for So 6.09.2019 14:20

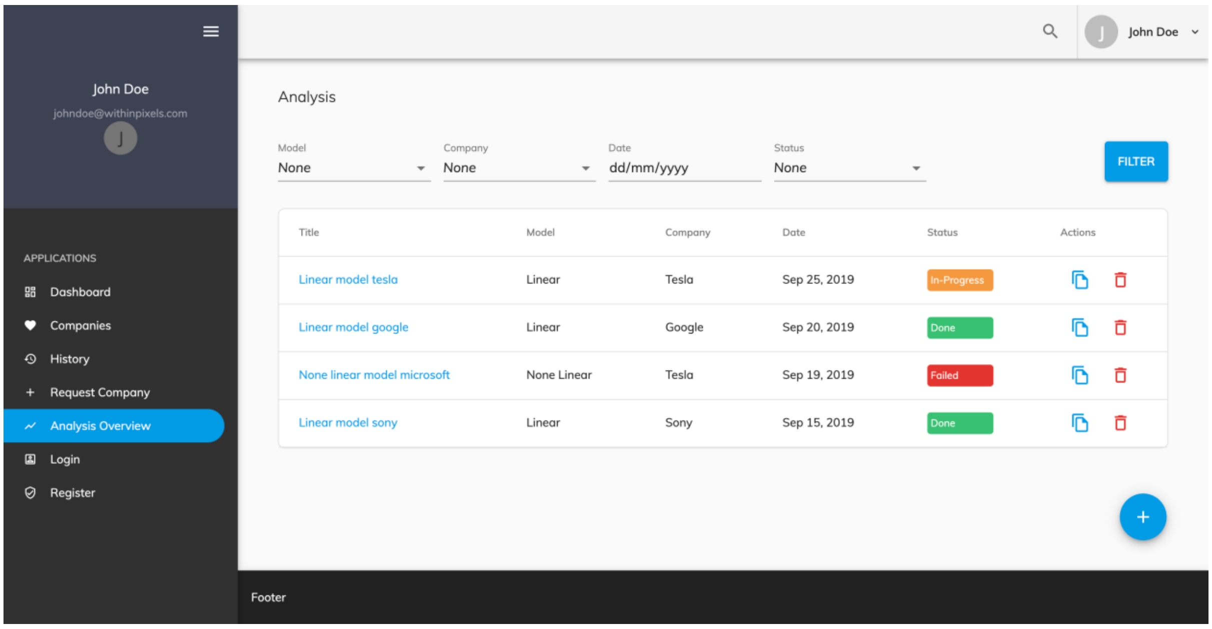
View History

- You viewed for Tesla 10.09.2019 15:30
- You viewed for Google 8.09.2019 09:10
- You viewed for Micosoft 7.09.2019 12:00
- You viewed for Apple 6.09.2019 14:20

Footer

Figure 16 – History

To give the user more flexibility, he will also be able to create own models to be trained. For this, in the next page 'Analysis Overview', the user will be able to see a list of models that he has created for corresponding companies. In this list, he will be able to filter the created models by model, company, date of creation and status of the model which can be 'Done', 'In progress' or 'Failed'.


John Doe

johndoe@withinpixels.com

APPLICATIONS

- Dashboard
- Companies
- History
- Request Company
- Analysis Overview
- Login
- Register

Analysis

Model: None · Company: None · Date: dd/mm/yyyy · Status: None · FILTER

Title	Model	Company	Date	Status	Actions
Linear model tesla	Linear	Tesla	Sep 25, 2019	In-Progress	
Linear model google	Linear	Google	Sep 20, 2019	Done	
None linear model microsoft	None Linear	Tesla	Sep 19, 2019	Failed	
Linear model sony	Linear	Sony	Sep 15, 2019	Done	

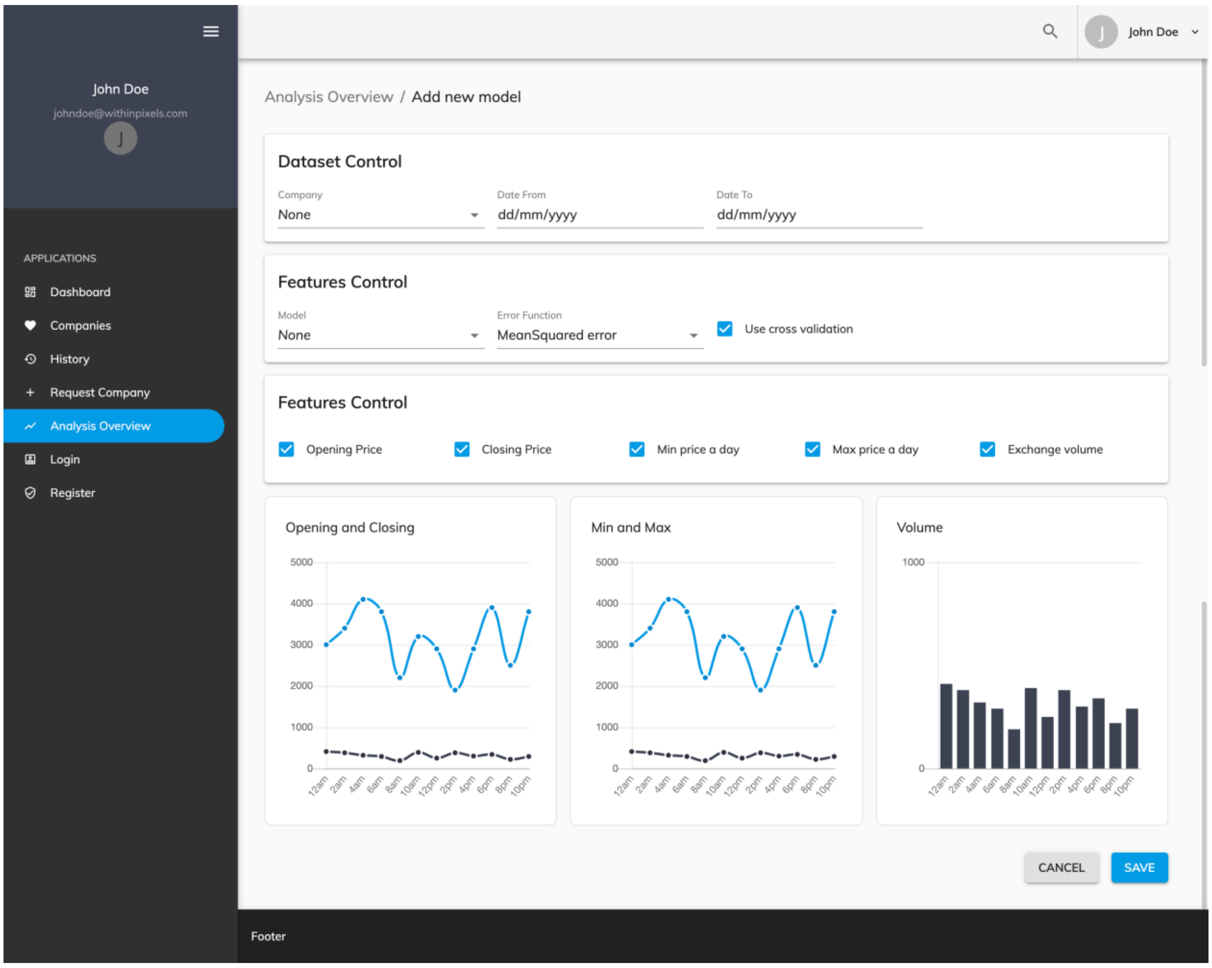
Footer

Figure 17 – Analysis Overview

In the list of already created models, the user will be able to view the model report in which the predicted prices and other information will be shown. Also, the user will be able to create a duplicate of the model and delete it by clicking the buttons under Actions section. When the user wants to view the report, he will also have the option to export it and download it and furthermore compare it to another already created model.

On the bottom of the page where is the round blue button, he will be able to create a new model. Clicking on it will redirect the user to a new page where he will be able to choose the model and set all related parameters in order to create a fully customized prediction model according to his preferences.

In this view, as depicted below in Figure 17, the user will choose the company and the time interval for which to predict the prices, the model and the error function to be used and based on the chosen model he will also be able to choose to enable or disable the corresponding features such as opening/closing price, min/max price of the day, exchange volume, epochs and so on.



Dataset Control

Company: None
Date From: dd/mm/yyyy
Date To: dd/mm/yyyy

Features Control

Model: None
Error Function: MeanSquared error
✓ Use cross validation

Features Control

✓ Opening Price ✓ Closing Price ✓ Min price a day ✓ Max price a day ✓ Exchange volume

Figure 18 – Add New Model

Once the user clicks on Save, this model will be sent to back end where the model will be trained according to parameters set by the user. Before the training, the custom model will be added to the queue which has other custom model training jobs. Once the training is done and the report is ready, the user will be notified via email and also in the UI.

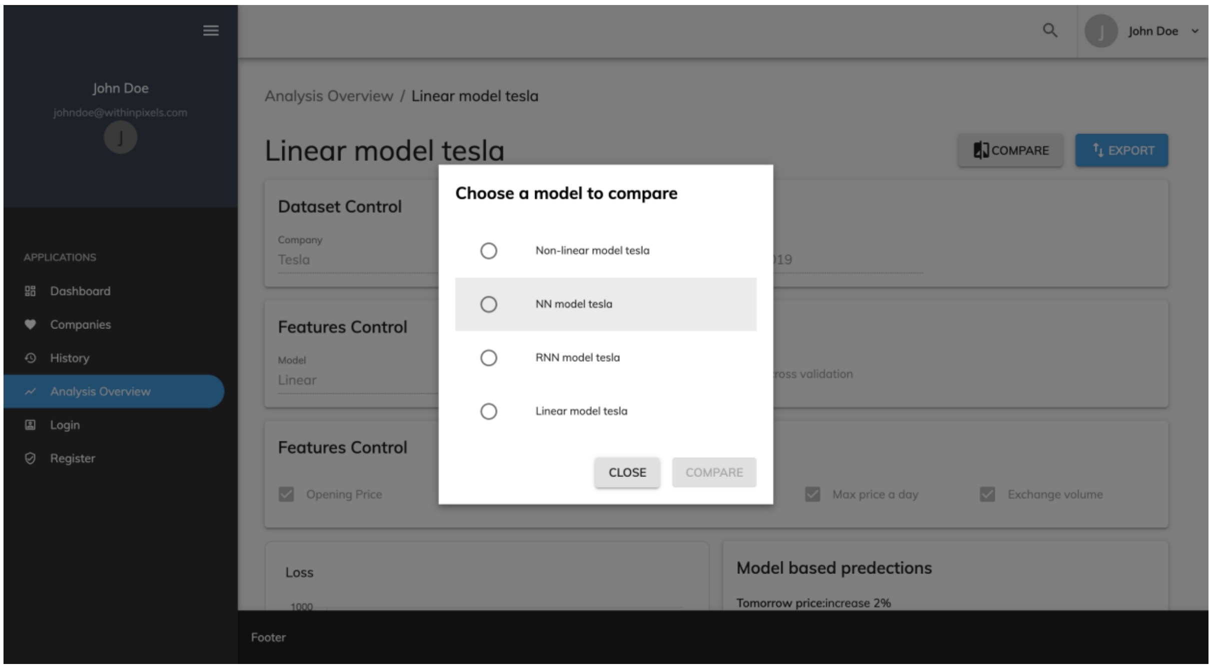
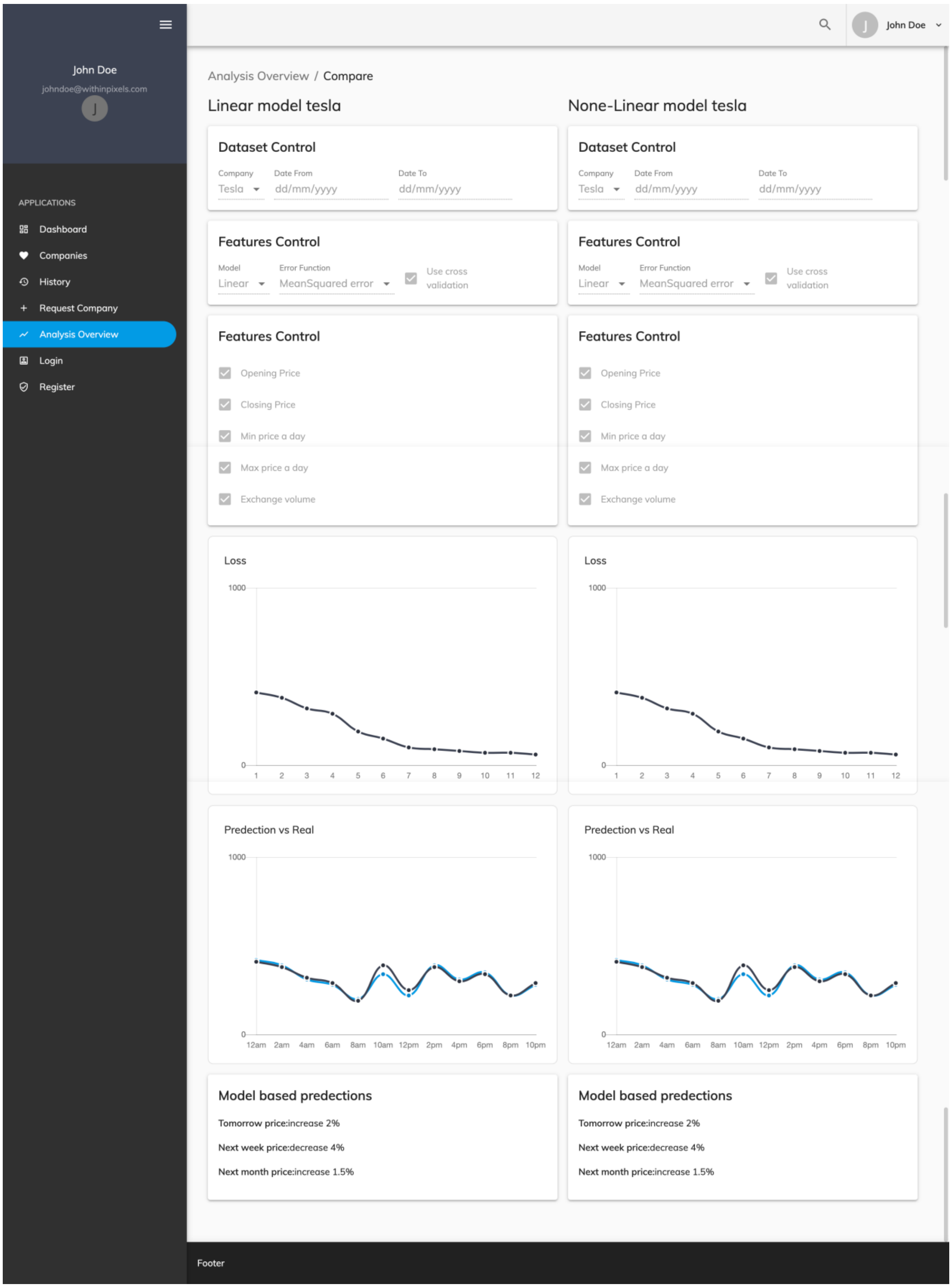


Figure 19 – Choose Comparison

In the list of already created models, clicking on the name of a model will take the user to a page where he can see all details regarding that prediction, including the model and other variables or parameters chosen during creation. The user will be able to export the report and save it for further use or compare it to other models already created.



Linear model tesla

Dataset Control

Company: Tesla | Date From: dd/mm/yyyy | Date To: dd/mm/yyyy

Features Control

Model: Linear | Error Function: MeanSquared error | ✓ Use cross validation

Features Control

✓ Opening Price
✓ Closing Price
✓ Min price a day
✓ Max price a day
✓ Exchange volume

Model based predections

Tomorrow price:increase 2%
Next week price:decrease 4%
Next month price:increase 1.5%

None-Linear model tesla

Dataset Control

Company: Tesla | Date From: dd/mm/yyyy | Date To: dd/mm/yyyy

Features Control

Model: Linear | Error Function: MeanSquared error | ✓ Use cross validation

Features Control

✓ Opening Price
✓ Closing Price
✓ Min price a day
✓ Max price a day
✓ Exchange volume

Model based predections

Tomorrow price:increase 2%
Next week price:decrease 4%
Next month price:increase 1.5%

Figure 20 – Models Comparison

In the comparison page, as shown in Figure 19, the user will be shown both models that he decided to compare.

5 SOFTWARE ENGINEERING METHODS

The software will be developed using the agile development model with several key iterations. The first iteration will focus on data scraping and storing data in the database in a structured way. The rest of the iterations will focus on building the project in small portions.

This section will provide an overview of what methods we will use to create the product and how we will perform quality assurance testing.

5.1 Coding Standard

We will follow a coding standard for the development of the project that will increase readability and maintainability. (The coding standard will be presented in the Implementation Plan.)

5.2 Code Reviews

As a team, we will meet the last day of every week to conduct code reviews.

5.3 Object-Oriented Design

We will use the object-oriented method for program design and keep this design updated with fixes for any potential problems we may catch. We will also perform design reviews throughout the project to ensure that the existing design will meet the criteria for the product.

5.4 Change Log

We will keep a track of every change in the project by using *bitbucket* version controller to get a better understatement of versions.

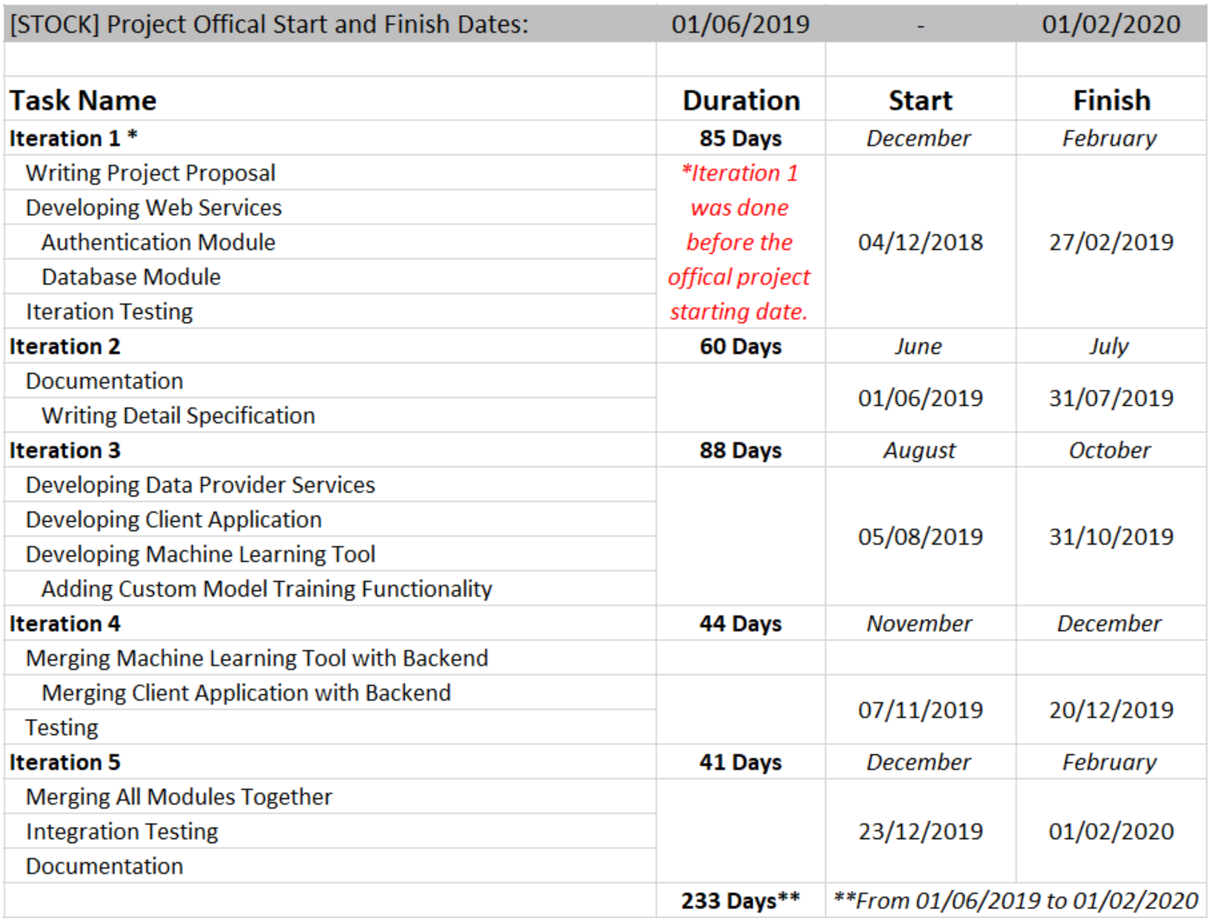
6 PROJECT AREAS

The project focuses on the following areas:

Discrete Mathematics and Algorithms	
	Discrete Mathematics and Algorithms
	Geometry and Mathematical Structure in Computer Science
	Optimization
Theoretical Computer Science	
	Theoretical Computer Science
Software and Data Engineering	
✓	Software Engineering
✓	Software Development
✓	Web Engineering
✓	Database Systems
✓	Analysis and Processing of Large Data
Software Systems	
	System Programming
	Reliable Systems
	Powerful Systems
Mathematical Linguistics	
	Computer and Format Linguistic
	Statistical Methods and Machine Learning in Computer Linguistics
Artificial Intelligence	
	Intelligent Agents
✓	Machine Learning
	Robotics
Computer Games and Computer Graphics	
	Computer Graphics
	Development of Computer Games

Table 2 – Project Areas

7 PROJECT SCHEDULE



[STOCK] Project Offical Start and Finish Dates:	01/06/2019	-	01/02/2020
Task Name	**Duration**	**Start**	**Finish**
Iteration 1 *	**85 Days**	*December*	*February*
Writing Project Proposal	**Iteration 1 was done before the offical project starting date.*	04/12/2018	27/02/2019
Developing Web Services			
Authentication Module			
Database Module			
Iteration Testing			
Iteration 2	**60 Days**	*June*	*July*
Documentation		01/06/2019	31/07/2019
Writing Detail Specification			
Iteration 3	**88 Days**	*August*	*October*
Developing Data Provider Services		05/08/2019	31/10/2019
Developing Client Application			
Developing Machine Learning Tool			
Adding Custom Model Training Functionality			
Iteration 4	**44 Days**	*November*	*December*
Merging Machine Learning Tool with Backend			
Merging Client Application with Backend		07/11/2019	20/12/2019
Testing			
Iteration 5	**41 Days**	*December*	*February*
Merging All Modules Together		23/12/2019	01/02/2020
Integration Testing			
Documentation			
	233 Days**	***From 01/06/2019 to 01/02/2020*	

Table 3 – Project Schedule

8 RESOURCE INFORMATION

All resources are available for educational purposes and there are no usage restrictions.

[1] https://www.tensorflow.org/

[2] https://keras.io/

[3] https://scikit-learn.org/

[4] https://scikit-learn.org/stable/supervised_learning.html

[5] https://pandas.pydata.org/

[6] https://matplotlib.org/

[7] https://textblob.readthedocs.io/en/dev/

[8] https://pypi.org/project/vaderSentiment/

[9] https://www.nltk.org/

[10] https://aws.amazon.com/dynamodb/

[11] https://aws.amazon.com/cognito/

[12] https://material-ui.com/