**Pirimid Graduate Training Program**

[Introduction](#_o8rolirvm3mi)

[Project](#_r0tx6krbtx9h)

[Standup](#_frliat6t0qb3)

[Slack](#_kwllg48xlhzv)

[Finance Domain](#_bubmkeb1ec52)

[Day 1 - Introduction](#_4g090fabkjhh)

[Week 1 - General](#_nhxm0eu1h0cs)

[Week 2 - Basics of backend and Frontend](#_1zhfw9m9n9fo)

[Learning Milestone:](#_9c28aacusi2k)

[Week 3 - Scaffolding of backend application](#_d99iv6xlpu3h)

[Learning Milestone:](#_4dqje8z6o0aw)

[Project Milestone:](#_ydjlajfdhnij)

[Week 4 - Application APIs (Back-end Implementation)](#_najb50z829oz)

[Learning Milestone:](#_cgoknzdfi6p5)

[Project Milestone:](#_ugsyt6d7dnm0)

[Week 5, 6 - Integrate Backend APIs in Front-end](#_lof185mtes91)

[Learning Milestone:](#_1q1c628poj0m)

[Project Milestone:](#_3kknlmjc1d7e)

[Week 7-10 - Implement Product Backlog](#_e1cg3in30mod)

[Learning Milestone:](#_qd0t7uh304xo)

[Project Milestone:](#_iz40pgof5o3k)

[Cloud & Security (Optional)](#_wj22096y9qx9)

[AI/ML Introduction (Optional)](#_v6jyo2ycv474)

[Mobile Technologies (Optional)](#_9kbzc6ks84li)

[Investment Platform Features (TBD)](#_yi2y0ymwftck)

# Introduction

Purpose of this exercise is to enable interns for an enterprise ready financial solution. Aim of this 3 months activity is to get familiar with the Finance domain along with the tech stack which we use for our products.

## Project

* A MVP for a trading platform like <https://groww.in/>. Detailed scope of a project is mentioned at the end of this document.

## Standup

* Daily standup to track progress, roadblocks, tasks @ 10 am
* Frequent demos of progress

## Slack

* Use #gtp channel for any communication/ query resolution / discussions

# Day 1 - Introduction

* Introduction of company, various projects, business, teams, tech (Nirav)
* Introduction of company, culture, policies, events (Rishi)
* Introduction to the structure of the program
* A call with all the mentors

# 

# Week 1 - General

* Linux and CLI
  + Setup Ubuntu
  + <https://ubuntu.com/tutorials/command-line-for-beginners#1-overview>
  + <https://linuxjourney.com/>
  + <https://www.learnenough.com/command-line-tutorial/basics>
* Getting familiar with an IDE, IntelliJ and VSCode
* Git Version Control
  + Setup git locally
  + [Learn Git Branching](https://learngitbranching.js.org/)
  + [A successful Git branching model](https://nvie.com/posts/a-successful-git-branching-model/)
* How the Internet Works
  + Server/Client
  + HTTP/HTTPS
  + API (REST)
  + Data Formats
    - <https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON>
  + <https://web.stanford.edu/class/msande91si/www-spr04/readings/week1/InternetWhitepaper.htm>
  + <https://tleyden.github.io/blog/2016/09/30/the-lifecycle-of-an-http-request/>
  + <https://github.com/alex/what-happens-when>

# 

# Week 2 - Basics of backend and Frontend

## Learning Milestone:

* Core Java
  + Implementing OOP concepts in Java
    - <https://medium.com/@kkajasu/object-oriented-programming-with-real-world-example-63f69f917d62>
  + [Enum](https://www.geeksforgeeks.org/enum-in-java/), Class, abstract class, Interface and when to use what
  + Java collections - List, Set, Map, Stack, Queue (Important)
    - <https://docs.oracle.com/javase/tutorial/collections/intro/index.html>
  + JDK 8 features - Stream , Optional, Lambda expressions (Important)
    - <https://stackify.com/streams-guide-java-8/>
    - <https://winterbe.com/posts/2014/03/16/java-8-tutorial/>
  + Multithreading and Concurrency (Basic Understanding)
    - <https://winterbe.com/posts/2015/04/07/java8-concurrency-tutorial-thread-executor-examples/>
* JavaScript fundamentals
  + How to write and run Javascript code, How the browser runs the js code.
  + JS fundamentals ([Link](https://javascript.info/first-steps))
  + Understand the difference between typed(Java) and typeless language (JS).
  + Javascript Arrays and different APIs of arrays
  + Event loop ([Link](https://www.youtube.com/watch?v=8aGhZQkoFbQ))
  + Callbacks, Promises ([Link](https://javascript.info/async))
  + Optional: [javascript-questions](https://github.com/lydiahallie/javascript-questions)
  + Debugging using chrome inspector ([Link](https://javascript.info/debugging-chrome))
* HTML, CSS (Basics)
  + Understanding of basic or most common html elements
  + Responsive Design
  + Understanding of basic or most commonly used events on html elements like onchange, onclick etc..
  + Basics of css like using css with css class, id, combination of class and id, combination of multiple classes etc. Try to learn how to use the css selectors.
  + Basics of display: flex ([Link](https://css-tricks.com/snippets/css/a-guide-to-flexbox/)).
  + CSS Specificity ([Link](https://www.w3schools.com/css/css_specificity.asp))
* SQL
  + Basic understanding of RDBMS
    - Entity design keeping normalization into the consideration
    - Constraints - Primary Key, Unique key, Foreign key
    - Joins, Aggregate functions, Grouping
    - Optional - Index and its trade-off
    - Association between tables - OneToOne, ManyToOne, OneToMany, ManyToMany
  + Setup any RDBMS like MySQL locally
    - <https://www.mysqltutorial.org/>

# 

# Week 3 - Scaffolding of backend application

## Demo/Session From Pirimid Team:

* + - * [Capital Markets](https://www.investopedia.com/terms/c/capitalmarkets.asp)
        + [Stocks trading](https://www.investopedia.com/university/stocks/stocks3.asp)
        + <https://www.investopedia.com/terms/h/hedgefund.asp> Dhanraj
        + [Positions](https://www.investopedia.com/terms/p/position.asp) Richa
        + [Portfolio Management](https://www.investopedia.com/terms/p/portfoliomanagement.asp) Richa

## Learning Milestone:

* Intro to agile development
  + Scrum, Kanban,Sprints
  + Reference Link - <https://www.atlassian.com/agile>
* Maven (basics)
  + Dependency management using maven
  + Structure of pom.xml
  + Maven goals and plug-ins lifecycle
  + How to build a project using Maven?
  + Reference
    - Basics <https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html>
    - Details <https://maven.apache.org/guides/getting-started/index.html>
* Spring and Spring Boot
  + Basic concepts of Spring
    - * Dependency injection
      * Spring annotations - Component,Service
      * Bean scopes
      * Bean lifecycle
      * Optional - How does spring work internally
      * Reference - <https://docs.spring.io/spring-framework/docs/current/reference/html/core.html#spring-core>
  + Spring Boot
    - * Advantage of spring boot
      * Refer starters libraries for JPA,Web applications and REST services
  + Create a new Spring boot project and run it
    - * Use <https://start.spring.io/> to generate application
* Hibernate, Spring Data and JPA
  + Connect spring boot application with database using datasource
  + Hibernate and JPA annotations.
  + Design entities
  + Spring data JPA
    - * JPARepository, CrudRepository
  + Optional - Transactional boundary

## Project Milestone:

* Divide product into User stories
  + Create Stories or Kanban board to track all the tasks
  + Identify assignees and if needed, create sub-tasks out of a feature
  + Reference - <https://medium.com/dev-genius/the-atom-of-agile-ff0b3537643f>
* Identify data source to pull stocks/mutual fund data from
  + Historical stock data and mutual fund data
  + Real time/ Delayed stock data
* Database Schema for Project Modules
* Scaffolding of Spring boot application which will serve APIs
* JPA entity design and its repositories

# 

# Week 4 - Application APIs (Back-end Implementation)

## Learning Milestone:

* REST API & Websocket with Spring Boot
  + Basic difference between REST API and Websocket.
  + When to use websocket over REST API?
  + How to implement both in Spring Boot?
* Testing (Unit and Integration)
  + Difference between Unit and Integration testing
  + Junit, Mockito etc.

## Project Milestone:

* Identify REST endpoints to develop for stock and mutual funds listing page
* Implementation of above endpoints
* Write unit tests and integration tests for above implementation.

# 

# Week 5, 6 - Integrate Backend APIs in Front-end

## Learning Milestone:

* ECMAScript Standards
  + Arrow functions, object destructuring, array destructuring, object initializer, spread operators, let, const etc..
* TypeScript
* React
  + What is React and how it works?
    - Virtual DOM
    - It's only a view part in M**V**C etc.
  + React component Lifecycle. When to use which?
  + Using react component state and props
  + React router
  + Using create react app or any other similar cli
  + Debugging using chrome React plugin
* Redux
  + What is Redux and why do we need it?
  + How to integrate/use redux with my React application?
  + Reducers, Actions, Dispatch, Store, State (Application State/Redux State), mapStateToProps, mapDispatchToProps, Connect etc.
  + One way data flow in React+Redux applications.
  + Debugging using chrome Redux plugin
* REST APIs, Websocket and Integration with backend
* Testing
  + Jest, Cypress, (Mocha, Chai, Sinon etc..)

## Project Milestone:

* Scaffolding of front end application
* Develop stock and mutual fund listing page and integrate it with APIs
* Develop stock profile and mutual fund profile page showing all the details of each

# 

# Week 7-10 - Implement Product Backlog

## Learning Milestone:

* Security and Authentication/ Authorization
  + Authentication and Authorization. Difference between these two.
  + Basic of most commonly used hashing techniques - Bcrypt.
  + How to implement this in Spring boot?
  + How to integrate this Authentication and Authorization in the front-end?
  + What is OAuth2 and different entities involved in it
* Design Principles and Patterns (Optional)
  + SOLID Principles
  + Basics of design patterns, what different type of design patterns are there, when to use them etc.([Link](https://refactoring.guru/design-patterns))

## Project Milestone:

* Login, User profile and change password
* Watchlist
* Screener view of mutual fund and stocks
* Websocket integration for real time price update of a stock.
* Trade option which will perform mock trading.

# Demo/Talk From Pirimid Team (Optional)

* + - * [Crypto](https://www.coinbase.com/learn)
      * FX - Anirudha
      * Banking
        + [Open Banking](https://www.investopedia.com/terms/o/open-banking.asp) Pathik/Chintan
        + <https://www.openbanking.org.uk/customers/what-is-open-banking/> Pathik/Chintan
        + [Account Aggregation](https://sahamati.org.in/)

# Mobile Technologies (Optional)

* React Native
  + Need understanding of all basic react fundamentals
  + What is the difference between React and React Native?
  + What will be different when actually creating components and all in the application between React and React native app?
  + How to create a React Native app and how to run it?
  + How to build a React Native app for Android and IOS?
  + Expo or similar tools to run and build the application
  + How to write android or ios specific code in a React Native application?

# Cloud & Security (Optional)

* AWS Cloud overview, tutorials
* IT Security best practices
* Deployment

# AI/ML Introduction (Optional)

This week, we will introduce you to ML/AI and create some cool projects. The idea is to learn the theory and practice the ML at the same time. There are going to be separate video series or teaching for theory and coding sessions for practical.

* Introduction to AI/ML
  + Introduction to AI/ML/DL/RL [link](https://towardsdatascience.com/introduction-to-machine-learning-for-beginners-eed6024fdb08)
  + Types of ML algorithms and tasks
  + Introduction to pandas/numpy ([Pandas](https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html)/[Numpy](https://numpy.org/doc/stable/user/quickstart.html))
* Classical ML algorithms ([ML Course](https://www.youtube.com/watch?v=PPLop4L2eGk&list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN) theory only)
  + Linear regression ([Link](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html))
  + Logistic regression ([Link](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html))
  + SVM
  + KNN
  + K-means clustering
  + PCA
  + Project for classification/regression
* Introduction to Deep Learning
  + Simple Neural Networks ([Link1](https://www.tensorflow.org/tutorials/quickstart/beginner)/[Link2](https://www.tensorflow.org/tutorials/keras/classification)/[Link3](https://www.tensorflow.org/tutorials/keras/regression)) ([Theory Videos](https://www.youtube.com/watch?v=CS4cs9xVecg&list=PLkDaE6sCZn6Ec-XTbcX1uRg2_u4xOEky0))
  + CNN ([Link1](https://keras.io/examples/vision/captcha_ocr/)/[Link2](https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/)) ([Theory](https://www.youtube.com/watch?v=ArPaAX_PhIs&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF))
  + RNN/LSTM/GRU ([Link1](https://keras.io/examples/nlp/lstm_seq2seq/)/[Link2](https://keras.io/examples/nlp/bidirectional_lstm_imdb/)) ([Theory](https://www.youtube.com/watch?v=_i3aqgKVNQI&list=PLkDaE6sCZn6F6wUI9tvS_Gw1vaFAx6rd6))
  + Extra theory on parameters tuning, bias and variance trade off and many other topics [Part1](https://www.youtube.com/watch?v=dFX8k1kXhOw&list=PLkDaE6sCZn6E7jZ9sN_xHwSHOdjUxUW_b)/[Part2](https://www.youtube.com/watch?v=1waHlpKiNyY&list=PLkDaE6sCZn6Hn0vK8co82zjQtt3T2Nkqc)
  + Transformers ([Link](https://www.tensorflow.org/tutorials/text/transformer)) ([Blog](http://jalammar.github.io/illustrated-transformer/)/[Video](https://www.youtube.com/watch?v=iDulhoQ2pro&list=PL1v8zpldgH3pQwRz1FORZdChMaNZaR3pu&index=4))
  + BERT/GPT 3 ([Link](https://keras.io/examples/nlp/masked_language_modeling/)) ([Paper](https://arxiv.org/abs/1810.04805)/[Video](https://www.youtube.com/watch?v=-9evrZnBorM&list=PL1v8zpldgH3pQwRz1FORZdChMaNZaR3pu&index=2))
  + Image Classification/News Sentiment Prediction Project
* Introduction to Model Deployment
  + Docker for ML/DS ([Link](https://www.youtube.com/watch?v=0qG_0CPQhpg))
  + TF Serving ([Link](https://www.tensorflow.org/tfx/guide/serving))
  + AWS Sagemaker ([Link](https://github.com/mobassir94/Deploy-trained-TensorFlow-2.x-models-using-Amazon-SageMaker))
* End to End ML Project
  + UI
  + Backend
  + APIs
  + Model training and deployment
  + Integration of model with app via backend

Python Basics(Optional)

* [Installation and configuration](https://docs.conda.io/en/latest/miniconda.html)
* [Quick Python Intro](https://www.programiz.com/python-programming/tutorial)
* Python DS:- [List](https://www.tutorialspoint.com/python/python_lists.htm), [Tuple](https://www.tutorialspoint.com/python/python_tuples.htm), [Dictionary](https://www.tutorialspoint.com/python/python_dictionary.htm)
* [Python OOPs](https://www.javatpoint.com/python-oops-concepts)
* [Introduction to Flask](https://code.tutsplus.com/tutorials/an-introduction-to-pythons-flask-framework--net-28822#:~:text=Flask%20is%20a%20small%20and,a%20short%20amount%20of%20time.)
  + [Introduction to Crud Operation in Flask](https://www.codementor.io/@garethdwyer/building-a-crud-application-with-flask-and-sqlalchemy-dm3wv7yu2)
  + [Ajax call to Flask Endpoint Video](https://www.youtube.com/watch?v=IZWtHsM3Y5A), [Simple Blog](https://medium.com/@yameday/python-flask-ajax-simple-example-2302424401de)
* [Design Patterns in Python](https://refactoring.guru/design-patterns/python)

# Investment Platform Features (TBD)

1. Why do people do the trading and how do the end users use it? Show some example platform.
2. Security
   1. Equity & MF
   2. Import Security
   3. Create Security
   4. List Security
   5. Delete Security
3. Prices
   1. ~~Yahoo Finance~~ (Find alternatives)
   2. Fetch Prices
   3. Historical Prices
4. Watchlist
   1. Create Watchlist
   2. Delete Watchlist
   3. Add Security
   4. Delete Security
   5. View Price
5. Position
   1. Fetch Positions
   2. Update Position
   3. Profit and Loss calculation
6. Authentication
   1. SIgnup
   2. Login