## Latitude BNPL Coding Challenge

### Scenario

Suppose we could access yesterday's stock prices as a list, where:

* The indices are the time in minutes past trade opening time, which was 10:00am local time.
* The values are the price in dollars of the stock at that time.
* So if the stock cost $5 at 11:00am, stock\_prices\_yesterday[60] = 5.
* You must buy before you sell. You may not buy and sell in the same time step (at least 1 time increment must pass).

### Assumptions

1. Stock prices are in dollar values represented by integers
2. Not all minutes in a day need to be represented

### Requirements

1. Write an efficient function that takes an array of stock prices and returns the best profit could have been made from 1 purchase and 1 sale of 1 stock.

For example:

int[] stockPrices = {10, 7, 5, 8, 11, 9};

Assert.assertEquals (6, getMaxProfit(stockPrices)); // returns 6 (buy at $5 sell at $11)

1. Create an api end point that can be used to access the method from a spring boot application.
   * The API must accept and return json
   * The request object must contain:
     + Identifier
     + Start date time
     + End date time
     + Array of stock prices
   * The response object must contain:
     + The request object
     + Processed date time
     + Max profit
     + Buy value
     + Sell value

### Bonus Requirements

1. Create a docker container for the Spring boot application
2. Create a simple React SPA to fetch and display the data
3. Add Github Actions for CI
4. Create a cloud formation template that will deploy the application to ECS Fargate

### Expectations

1. Implement a solution in Java, modify the provided application as required.
2. Prove it works by creating unit tests that test the possible scenarios that the numbers could present.
3. Include any comments that you think will be relevant to provide any context around the approach taken / solution developed.