Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт математики и фундаментальной информатики СФУ

Кафедра математического анализа и дифференциальных уравнений

УТВЕРЖДАЮ Заведующий кафедрой

\_\_\_\_\_ Фроленков И.В.

подпись инициалы, фамилия

« \_\_\_\_\_» \_\_\_\_\_\_\_ 20 \_\_\_ г.

**ОТЧЕТ О ПРАКТИКЕ**

**ПО ПОЛУЧЕНИЮ ПРОФЕССИОНАЛЬНЫХ УМЕНИЙ И ОПЫТА ПРОФЕССИОНАЛЬНОЙ ДЕЯТЕЛЬНОСТИ**

Место прохождения практики:

Институт математики и фундаментальной информатики

Тема практики: **Задача идентификации функции источника в полулинейном параболическом уравнении. Разрешимость прямой задачи.**

Руководитель \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата должность, ученая степень инициалы, фамилия

Студент ИМ19-01Б 171939447 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

номер группы номер зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2021

Содержание

Общая постановка задачи ……………………………………………………………………….….3

Разработка компонентов для работы с Tinkoff Invest API ……………………………..………....3

Основные функции ………………………………………………………………………………….3

Заключение …………………………………………………………………………………………..7

Источники ……………………………………………………………………………….…………...8

**Общая постановка задачи:**

Командный проект:

Реализовать логику работы Telegram бота, который уведомляет подписчика о резких изменениях стоимости акций на фондовом рынке и пересечении SMA.

Под резким изменением будем понимать:

Рост/падение стоимости актива на X процентов за выбранный пользователем период.

Проект реализован на php с использованием фреймворка Yii2. Проект использует: API Tinkoff инвестиции sandbox, PostgreSQL. Настройка происходит через команды в ЛС телеграм боту.

**Разработка компонентов для работы с Tinkoff Invest API.**

Моя роль заключалась в том, чтобы наладить работу с Tinkoff Invest API.

Для реализации взаимодействия с Tinkoff Invest API я использовал готовый php-клиент “ PHP client for Tinkoff invest API ” свободно распространяемый через github в соответствии с лицензией MIT.

**Основные функции:**

isTokenLegit (Проверяет переданный токен tinkoff invest api на правильность. В качестве результата возвращает true/false.)

public function isTokenLegit($tinkoff\_token) {  
 try {  
 $client = new TIClient($tinkoff\_token, TISiteEnum::*SANDBOX*);  
 $client->getStocks();  
 }  
 catch (TIException $e) {  
 return false;  
 }  
 return true;  
}

isTickerExist (Проверяет существует ли переданный тикер. В качестве результата возвращает true/false.)

public function isTickerExist($tinkoff\_token=*TINKOFF\_TOKEN*, $ticker ="SBER")  
{  
 $client = new TIClient($tinkoff\_token, TISiteEnum::*SANDBOX*);  
 $all\_stocks = $client->getStocks(); //Получаем массив со всеми тикерами.  
  
 foreach ($all\_stocks as $stock) {  
 if ($stock->getTicker() == $ticker) return true;  
 }  
 return false;  
}

isPriceShift(Проверяет изменилась ли цена, в качестве параметра передается свеча и процент предполагаемого изменения. В качестве результата возвращает true/false.)

public function isPriceShift($candle, $percent=5){  
 if ($candle instanceof Candle)  
 {  
 $shift = $candle->prcopen \* $percent / 100;  
 if (abs($candle->prcopen - $candle->prcclose) > $shift) return true;  
 return false;  
 }  
}

addCandle(Создает новую свечу)

public function addCandle($stock, $interval=TIIntervalEnum::*MIN10*,$tinkoff\_token=*TINKOFF\_TOKEN*){  
 $TIclient = new TIClient($tinkoff\_token, TISiteEnum::*SANDBOX*);  
 try {  
 $date\_interval = new DateInterval('PT' . strval($this->toSeconds($interval)) . 'S');  
 $new\_date = date\_sub(new DateTime('now'), $date\_interval);  
 $now = new DateTime('now');  
 $TIcandles = $TIclient->getHistoryCandles($stock->figi, $new\_date, $now, $interval);  
 $TIcandle = end($TIcandles);  
 if ($TIcandle) {  
 $candle = new Candle();  
 $candle->prcopen = $TIcandle->getOpen();  
 $candle->prcclose = $TIcandle->getClose();  
 $candle->prcmin = $TIcandle->getLow();  
 $candle->prcmax = $TIcandle->getHigh();  
 $candle->tradevolume = $TIcandle->getVolume();  
 $candle->timeq = $now->format('Y-m-d H:i:s');  
 $candle->stock\_id = $stock->id;  
 $candle->save();  
 }  
 else {  
 $candle = new Candle();  
 $candle->timeq = $now->format('Y-m-d H:i:s');  
 $candle->stock\_id = $stock->id;  
 $candle->save();  
 }  
 }  
 catch (TIException $e) {  
 echo($e->getMessage());  
 }  
}

checkStocks(Основная функция. Для каждого пользователя отслеживает изменение выбранных активов. В качестве результата возвращает массив ассоциативных массивов.)

public function checkStocks()  
 {  
 $users = User::*find*()->all(); // Получаем массив всех юзеров  
 $result = [];  
 foreach ($users as $user)  
 {  
 $stocks = Stock::*find*()->where(['user\_id' => $user->id])->all();  
 foreach ($stocks as $stock) {  
 $sma = 'none';  
 if ($stock) {  
 $latest\_candle = $this->getLatestCandle($stock);  
  
 if (!$latest\_candle) {  
 self::*addCandle*($stock, $stock->interval, $user->token);  
 $latest\_candle = $this->getLatestCandle($stock);  
 if (self::*isPriceShift*($latest\_candle, $stock->change)) {  
 $shift\_percent = ($latest\_candle->prcclose / $latest\_candle->prcopen - 1) \* 100;  
 array\_push($result, ['user' => $user, 'stock' => $stock, 'percent' => $shift\_percent]);  
 }  
 }  
 $date = new DateTime($latest\_candle->timeq);  
 $now = new DateTime('now');  
 $diff = $now->diff($date);  
 if (-$this->dateIntervalToSeconds($diff) > $this->toSeconds($stock->interval)) {  
 self::*addCandle*($stock, $stock->interval, $user->token);  
 $latest\_candle = $this->getLatestCandle($stock);  
 if (self::*isPriceShift*($latest\_candle, $stock->change)) {  
 $shift\_percent = ($latest\_candle->prcclose / $latest\_candle->prcopen - 1) \* 100;  
 array\_push($result, ['user' => $user, 'stock' => $stock, 'percent' => $shift\_percent]);  
 }  
 }  
 if ($candles = Candle::*find*()->where(['stock\_id' => $stock->id])->count() >= $stock->period)  
 {  
 $candles = Candle::*find*()->where(['stock\_id' => $stock->id])->all();  
 $latest\_candle = $this->getLatestCandle($stock);  
 $last\_n\_candles = array\_slice($candles, -$stock->period);  
 $average = array\_sum($last\_n\_candles)/count($last\_n\_candles);  
 if ($latest\_candle->prcclose) {  
 if ($latest\_candle->prcclose - $average > 0 && $last\_n\_candles[count($last\_n\_candles)-2]->prcclose - $average < 0) $sma = 'up';  
 if ($latest\_candle->prcclose - $average < 0 && $last\_n\_candles[count($last\_n\_candles)-2]->prcclose - $average > 0) $sma = 'down';  
 }  
 }  
 array\_push($result, ['user' => $user, 'stock' => $stock, 'sma' => $sma]);  
 }  
 }  
 }  
 return $result;  
 }  
}

**Заключение:**

Я успешно выполнил свою часть работы, наладил получение и обработку данных о активах при помощи Tinkoff invest api. Благодаря общим усилиям мы получили работающего нотификационного бота, а также опыт работы с Php, Yii2, Git, PostgreSQL.

**Источники:**

1. Tinkoff Invest API [Элетронный ресурс] – Режим доступа: <https://tinkoffcreditsystems.github.io/invest-openapi>
2. PHP client for Tinkoff invest API [Элетронный ресурс] – Режим доступа: <https://github.com/jamesRUS52/tinkoff-invest>
3. Документация по PHP [Элетронный ресурс] – Режим доступа: <https://www.php.net/>
4. Руководство по использованию Yii2 [Элетронный ресурс] – Режим доступа: <https://yiiframework.com.ua/ru/doc/guide/2/structure-overview/>