|  |
| --- |
| chart-data-api |
| IndexController  /// <param name="timeframe">Time Frame; 'D', 'W', 'M'</param>  Record : : تعدادی که باید برگردانده شوند  /// https://localhost:5001/api/Index/By-Record/IRX6X60T0006-2/D/10/2020-12-19 |
| [HttpGet("By-Record/{cisin}/{timeframe}/{record}/{date}")]  public async Task<IReadOnlyCollection<IndexCandle>>GetIndexFilterByRecord(string cisin,string timeframe,DateTime date,int record) |
|  |
| [HttpGet("By-Date/{cisin}/{timeframe}/{from}/{to}")]  public async Task<IReadOnlyCollection<IndexCandle>> GetIndexFilterByDate(string cisin,string timeframe,DateTime from,DateTime to) |
| InstrumentController |
| [HttpGet("By-Date/{isin}/{timeframe}/{from}/{to}")]  public async Task<IReadOnlyCollection<Candle>> GetInstrumentFilterByDate(string isin, string timeframe,DateTime from,DateTime to)  /// <param name="timeframe">Time Frame; '1', '5', '15', '30', '60', 'D', 'W', 'M'</param>  Example url: https://localhost:5001/api/Instrument/by-date/IRO1IKCO0001-2/H/2016-12-13/2020-12-19 |
| [HttpPost("List-By-Date/{adjustment}/{timeframe}/{from}/{to}")]  public async Task<IReadOnlyCollection<Candle>> GetInstrumentsFilterByDate([FromQuery(Name = "isins")] List<string> isinList, DateTime from, DateTime to, string adjustment = "0", string timeframe = "D")  /// <param name="adjustment">0, 2</param>  /// <param name="timeframe">D,W,M</param> |
| [HttpGet("By-Record/{isin}/{timeframe}/{record}/{date}")]  public async Task<IReadOnlyCollection<Candle>> GetInstrumentFilterByRecord(string isin, string timeframe, DateTime date, int record)  /// <https://localhost:5001/api/Instrument/By-Record/IRO1IKCO0001-2/H/10/2020-12-19>  همه نمادهایی که اخر ان companyIsin  باشد که از var companyIsin = instrumentIsin.Substring(4); بدست میاید |
|  |
| [HttpGet("LastCandles")]  public async Task<IReadOnlyCollection<Candle>> GetLastCandles(string timeframe, DateTime date)  10000 تا از اخرین رکورد را برمیگرداند |

|  |
| --- |
| Admin-panelگرفتن تعدیلهای انجام شده در بازه های مشخص شده از سرویس اسا |
| 1. داده ها را براساس فیلترهایی که مرحله به مرحله از کاربر بصورت command console میپرسد   .AddChoiceGroup("All", ["Share", "None-Share"])  .AddChoiceGroup("All", ["None-Adjusted(0)", "Adjusted(2)"])  .AddChoiceGroup("All", ["Day", "Week", "Month"]) |
| 1. لیست نمادها را از الستیک سرچ رابین با تابع Task<List<string>> GetTradedInstrument(DateTime date); میگیرد |
| 1. براساس for های تو در تو براساس فیلترهای گرفته شده اطلاعات را از اسا میگیرد با سرویس زیر 2. var data = await GetInstrumentCandle(token, isin, from, to, adjustment, timeFrame); |
| 1. داده ها را با دستور \_candleRepository.IndexBulk(processedCandles); به الستیک سرچ میریزد |

نکته بند 4 و 5

|  |
| --- |
| private async Task<List<Candle>> GetInstrumentCandle(string token, string isin, DateTime from, DateTime to, string adjustment = "0", string timeFrame = "1D")  {  try  {  long fromEpoch = GetEpoch(from);  long toEpoch = GetEpoch(to);  var client = new HttpClient();  request.Headers.Add("Authorization", $"Bearer {token}");  request.Headers.Add("Cookie", "cookiesession1=678B286AC44C4AAD9B2FA9FFC3F380C6");  var request = new HttpRequestMessage(HttpMethod.Get, $"https://tseonlineapi.agah.com/api/v1/TradingViews/history?symbol=" +  $"{isin}&from={fromEpoch}&to={toEpoch}&resolution={timeFrame}&symbolType=سهام");  var response = await client.SendAsync(request);  response.EnsureSuccessStatusCode();  var str = await response.Content.ReadAsStringAsync();  var data = JsonConvert.DeserializeObject<ThirdPartyCandleResult>(str);  if(data == null || data?.isSuccess == false)  {  AnsiConsole.WriteLine("Fail to get data .");  }  var candles = toCandle(data.data, isin, adjustment, timeFrame);  return candles;  }  catch (Exception ex)  {  AnsiConsole.WriteException(ex);  return null;  } |
| public async Task IndexBulk(List<Candle> entities, CancellationToken cancellationToken = default(CancellationToken))  {  try  {  BulkResponse bulkResponse = \_client.Bulk((BulkDescriptor b) => b.IndexMany(entities, (BulkIndexDescriptor<Candle> d, Candle doc) => d.Document(doc).Index(IndexName)));  if (!bulkResponse.IsValid)  {  Console.WriteLine($"Failed to index {entities.Count} messages. Error: {bulkResponse.ServerError?.Error}");  }  }  catch (Exception ex2)  {  Exception ex = ex2;  Console.WriteLine(ex.Message + "\n" + ex.StackTrace);  }  } |

public class Candle

{

[JsonIgnore]

public string Id { get; set; }

[JsonIgnore]

[Keyword(Name = "iSIN")]

public string ISIN { get; set; }

[JsonIgnore]

[Keyword(Name = "adjustmentType")]

public string AdjustmentType { get; set; }

[JsonIgnore]

[Keyword(Name = "timeFrame")]

public string TimeFrame { get; set; }

[Date(Name = "dateTime")]

public DateTime DateTime { get; set; }

//public string Time { get; set; }

[Number(Name = "open")]

public int Open { get; set; }

[Number(Name = "high")]

public int High { get; set; }

[Number(Name = "low")]

public int Low { get; set; }

[Number(Name = "close")]

public int Close { get; set; }

[Number(Name = "volume")]

public long Volume { get; set; }

}

Public class IndexCandle

{

[JsonIgnore]

[Keyword(Name = "iSIN")]

public string ISIN { get; set; }

[JsonIgnore]

[Keyword(Name = "adjustmentType")]

public string AdjustmentType { get; set; }

[JsonIgnore]

[Keyword(Name = "timeFrame")]

public string TimeFrame { get; set; }

[Date(Name = "dateTime")]

public string DateTime { get; set; }

//public string Time { get; set; }

[Number(Name = "open")]

public int Open { get; set; }

[Number(Name = "high")]

public int High { get; set; }

[Number(Name = "low")]

public int Low { get; set; }

[Number(Name = "close")]

public int Close { get; set; }

//public long Volume { get; set; }

}