Bar graph documentation

Contents

Technical introduction

Used technologies

C#, .NET, RESTful API, Entity Framework, MVC, HTML, CSS, Bootstrap, jQuery, Canvas.js

Decisions I've made

Architecture

I decided to use this architecture because it's organized/structured and easy for manage. WebAPI layer is not necessary here but I've added it just for show how we can split logic as a separate sevice. For bigger project or project where we'd like to have fast deployment/delivery I recommend using microservices architecture. Also, you can see here common and BO which shouldn't be separately for project with one class but idea is to BO contain only business object and common should contain only common things (e.g. enums, resource files, classes for communicate between layers).

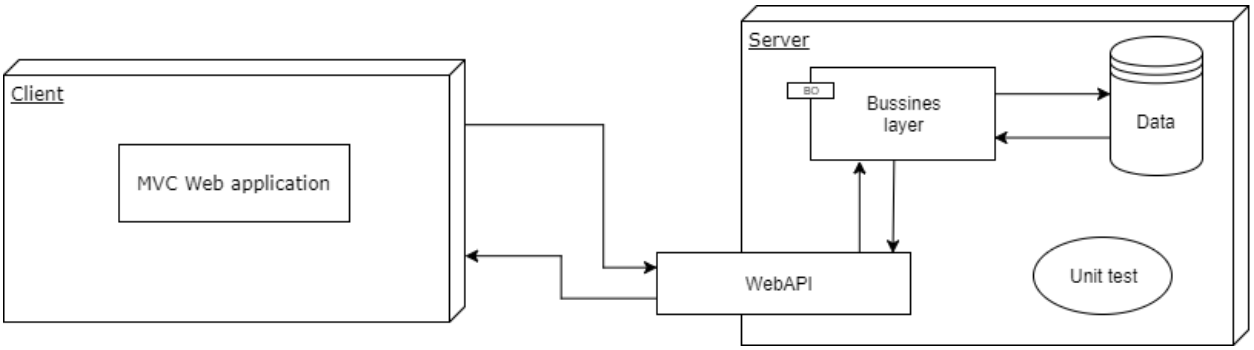

Figure1 - Arhitecture

Database

Structure

I considered two solutions, first to use single table and second one to create two tables (one for bars and one for colors) with n to n relation. I decided to use single table because it is faster to insert/read.

Also, I thought about type for BarId column between Guid and int. I used int because it's always better indexed then Guid and I will have no more data then max in value. If we have that big table that int can't cover it then I suggest using sequential Guid (it's better indexed then non sequental/regular Guid)

Insert/Delete data

I decided to use history pattern (add delete date instead of delete) because in that way of storage I'll have history of all changes. For large project with big database I don't recommend this because it will cost us speed and space, in that case we should consider some other methods for implement history pattern (crate new table for old data, using new database for saving history,…)

Message storage

I consider to use enums vs resource files. On the end, I used enums without any special reason, just to show how to read description. For my previous application I used resources files. Resources files are also better if we have multilanguage application.

Charts

I consider couple solutions:

- Highcharts
- Telerik charts
- Canvas.js charts

I can highly recommend highcarts but because I'd like to try new one I've implemented Canvas.js.

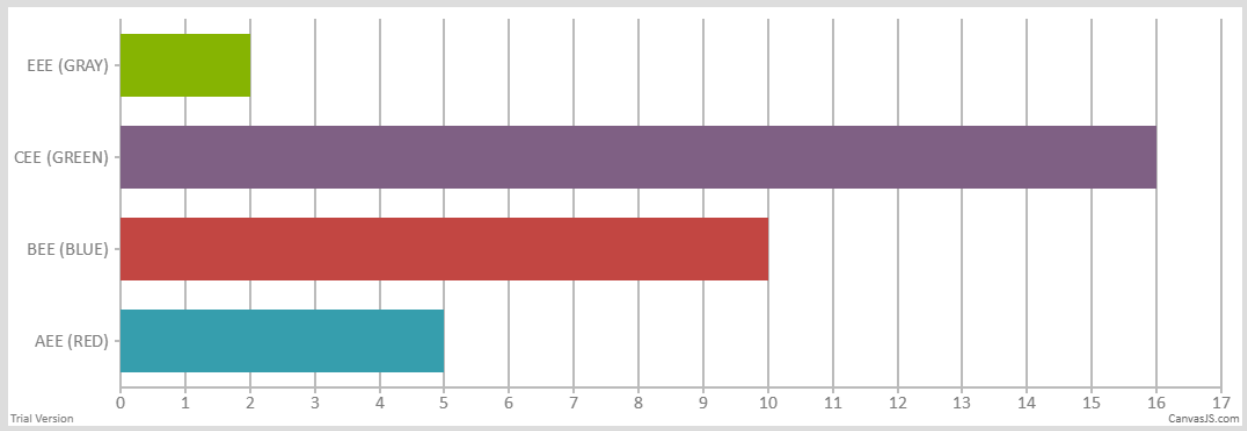

Figure2 – Canvas.js chart

Project setup

- Create database BarGraph and then create table Bars by using script or use database backup file
- Change connection strings in:
    - BarGraph.WebAPI.Tests -> App.Config
    - BarGraph.Data -> App.Config
- Publish WebAPI and change API url in BarGraph.Web - > Web.config / appSettings (BarGraph.WebAPI) with new one
- Enjoy using application

UI

Upload file

Accepted file is .txt with data in format #Name:Color:Size
Name – accept letters and numbers
Color – accept only letters
Size – accept only numbers

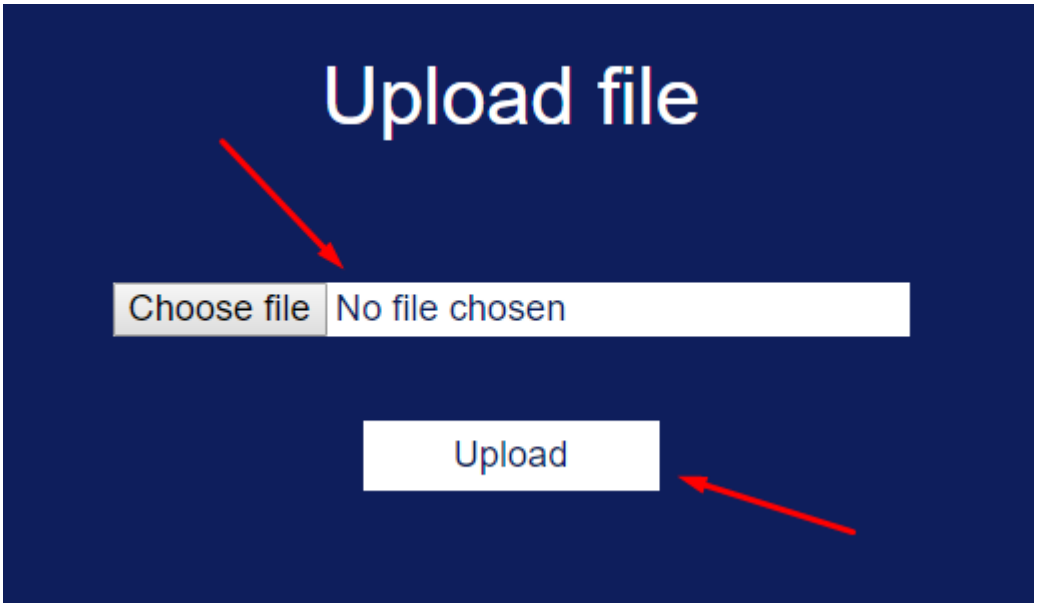

figure3 – Upload file

Results

For showing result you can choose between chart and table view simply by click on buttons “Table”/”Chart”. Because randomized size is reloading for every 10 second there is “Refresh” button which return us real data from database. For table view there are some predefined colors co colored background can be visible (as on picture below).

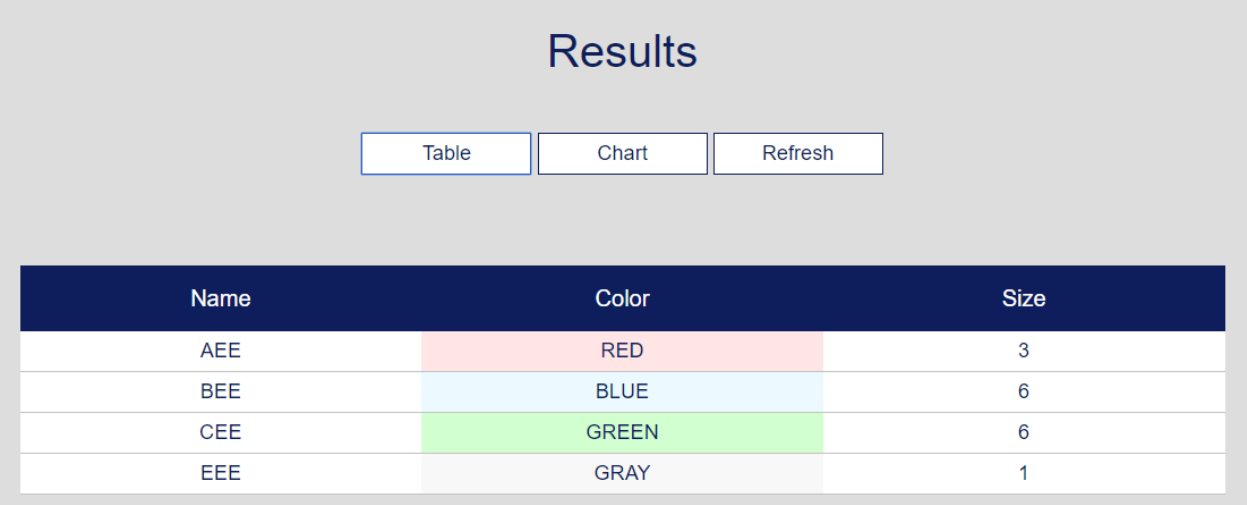

Figure4 – Results (table view)