Wavecell
An 8x8 Company

“Take home” test for C# engineer: “Advanced Feature Filter”

Objective

Determine the candidate's efficiency, speed and code quality in producing well designed app code and with .

Description

There is set of rules with following structure: RuleId, Rule Priority, Filter1.. Filter4, OutputValue.
Sample:

RuleId	Priority	Filter1	Filter2	Filter3	Filter4	OutputValue
1	80	AAA	<ANY>	CCC	DDD	8
2	10	<ANY>	<ANY>	AAA	<ANY>	1
3	70	BBB	<ANY>	CCC	<ANY>	7
4	100	AAA	BBB	CCC	<ANY>	10
5	50	CCC	AAA	<ANY>	CCC	5
6	0	<ANY>	<ANY>	<ANY>	<ANY>	0

Purpose of your library/application to find best matching rules for datasets with filtering values (Filter1, ..., Filter4). If record matches multiple rules, rule with highest priority must be selected (100 – highest priority, 0 – lowest priority). Value “*<ANY>*” means, that any value is accepted for this filter.
Initially rules are stored in CSV file (see attached *SampleData.csv*).
In real production environment quantity of rules expected to be 1000+. Quantity of data to pass through filter is 10x-100x larger or might be even infinity data stream in future.

Must have features:

1) To load data from a file to array of **strongly typed entities.**
   For example:

```
class StrategyRule : BaseRule
{
   public int RuleId;
   public int Priority;
   public string Filter1;
   public string Filter2;
   public string Filter3;
   public string Filter4;
   public int? OutputValue;
}
```

2) To create a library that will be finding best matching rule (respecting rule priority) and its *OutputValue* for the set of input values.
   Sample to describe idea:

```
BaseRule? r = engineStrategy.FindRule(string Val1, string Val2, string Val3, string Val4);
```

Expected, that this module will be used to find matching rules with the frequency 100 calls/second over loaded dataset.

Should have features:

1) Taking into account that collection can store 1000+ records it should be performant enough.
2) Code should be reusable as much as possible to implement different Rules models.
   For instance, to write another set of strongly typed rules with Filters: Filter1(int), Filter2(bool), Filter3(string).

Validation

In case of provided sample data (see *Description*), expected results will be following:

Filter1	Filter2	Filter3	Filter4	RuleId	OutputValue
AAA	BBB	CCC	AAA	4	10
AAA	BBB	CCC	DDD	4	10
AAA	AAA	AAA	AAA	2	1
BBB	BBB	BBB	BBB	6	0
BBB	CCC	CCC	CCC	3	7

Demo

- Provide zipped Visual Studio solution with source code of app and working Unit tests.

Good luck!