‘WINGS ON’ USER STORY

**Technical Assessment**

|  |  |
| --- | --- |
|  | |
| Customer Name | Travix |
| Project/Sub-Project | Technical Assessment documentation |
| File Name | Wings Documentation.docx |
| Version | 0.1 |
| Compatibility Version | 1.0 |
| Author | Mohamed Salah |
| Publish Date | 15-JUN-2019 |
| Reviewer |  |
| Review Date |  |

Disclaimer

© 2019 All rights reserved. This document has been created for use by Travix International. These rights are not transferable to any third party without the prior written consent of the author.

Table of contents

[Record of Changes 5](#_Toc11687399)

[Approvals 6](#_Toc11687400)

[Distribution List 6](#_Toc11687401)

[Documents References 7](#_Toc11687402)

[1 Sum of Multiple 8](#_Toc11687403)

[1.1 Problem statement 8](#_Toc11687404)

[1.2 Assumption 8](#_Toc11687405)

[1.3 Solution Approaches 8](#_Toc11687406)

[1.3.1 Approach A 8](#_Toc11687407)

[1.3.2 Approach B 9](#_Toc11687408)

[1.3.3 Approach C 10](#_Toc11687409)

[1.4 Design Pattern 11](#_Toc11687410)

[1.4.1 Template Method Pattern 11](#_Toc11687411)

[1.4.2 Dependency Injection 12](#_Toc11687412)

[1.5 Statistical Running 12](#_Toc11687413)

[2 Sequence Analysis 13](#_Toc11687414)

[2.1 Problem Statement 13](#_Toc11687415)

[2.2 Assumption 13](#_Toc11687416)

[2.3 Solution Approaches 13](#_Toc11687417)

[2.3.1 Approach A 13](#_Toc11687418)

[2.3.1.1 Pseudocode and Complexity 13](#_Toc11687419)

[2.3.2 Approach B 13](#_Toc11687420)

[2.3.2.1 Pseudocode and Complexity 14](#_Toc11687421)

[2.4 Design patterns 15](#_Toc11687422)

[2.4.1 Factory Method Pattern 15](#_Toc11687423)

[2.4.2 Dependency Injection 16](#_Toc11687424)

[2.5 Statistical Running 16](#_Toc11687425)

[3 Testing Solution 18](#_Toc11687426)

# Record of Changes

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Change Reference |
| 0.1 | 05-Jan-2019 | Mohamed Salah | Initial version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Approvals

This document has been approved by the following people

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Function | Date of Approval | Signature |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Distribution List

This document has been distributed to:

|  |  |
| --- | --- |
| Name | Function |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Documents References

|  |  |  |
| --- | --- | --- |
| Document Name | Document Reference | Version |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 

# Purpose

The purpose of this document is presenting the **design** and **development** approaches to achieve the requirement in the technical assignment.

I did my best to cover most common topic in design Restful API.

# Problem statement

As a Front-End Developer, I need a REST-ful Web API for my ticketing website, so that I can access and manage information related to the passengers.

Website need the following:

1. Endpoint that returns a person by Id.

2. Endpoint that returns all passengers on the flight by number for example ‘PZ696’.

3. Endpoint that updates a person’s email address.

4. Endpoint that creates a booking of an existing flight for a new passenger.

5. Endpoint that lists all the male passengers.

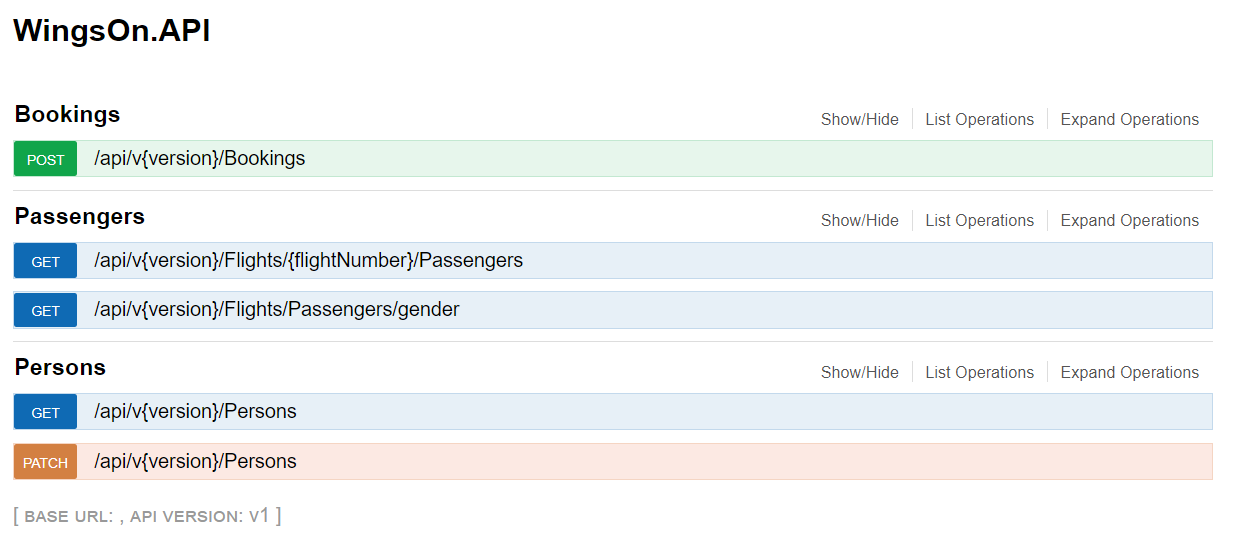
# Endpoint Table

The following is the Endpoint Table

|  |  |  |
| --- | --- | --- |
|  | Endpoint | Usage |
| POST | /api/v1/Bookings |  |
| GET | /api/v1/Flights/{flightNumber}/Passengers |  |
| GET | [/api/v1/Flights/Passengers/gender](http://localhost:63312/swagger/ui/index#!/Passengers/Passengers_Get) |  |
| GET | [/api/v1/Persons](http://localhost:63312/swagger/ui/index#!/Persons/Persons_Get) |  |
| PATCH | [/api/v1/Persons](http://localhost:63312/swagger/ui/index#!/Persons/Persons_Get) |  |

Also, you can find this using Swagger url

http://<hostname>:<port>/swagger/ui/index



## Assumption

The following assumption will apply for the different approaches

* 3 or 5 will be as part of the input so that user feel free to change them any time and test again
* The result must be representable number and not exceed the max number of long type.
* The numbers of Divisors (3,5,..) is small list so that its complexity is order of constant.
* For Approaches, B&C Divisors are not able to dividable on each other. For example Correct Divisors [2,3,5] and wrong Divisors example [2,3,6]

## Best Practice for Building REST API

Use nouns but no verbs

## Solution Approaches

There are 3 approaches and one final solution describe as following

### Approach A

The easy solution is to loop on the range and check every number of it able division by one of the multiplier or not then add to total sum.

Pseudocode

*Loop from 1 to limit*

*If number x able to divide with any of Divisors and remainder is 0*

*Then*

*Add number x to the accumulated result*

*Else*

*Continue*

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Time Complexity | O(N) | N : limit provided |
| Space Complexity | O(1) |  |

### Approach B

The problem in Approach A is the need to check every number from 1 to the upper limit. When we will calculate only the sequence of divisors

For example divisor 3

It is need to calculate 3 + 6 + 9 + to N

Therefore, the sequence is like k+2K+3k+... Where K is any divisor

Therefore, it is good to modify the loop to be jump into the correct number directly and calculate the number that will match the condition.

So if there are 2 divisor 3 & 5

Our sequence will be

5 =>: 5(1+2+3+4+5+….N) 5, 10, 15, 20,

3 =>: 3(1+2+3+4+5+….M) 3, 6, 9, 12, 15,

Where N & M are the max number, allow division without remainder less that limit provided in the input parameter.

So the sequence will be 5(1+2+3+4+5+….N) + 3(1+2+3+4+5+….M)

Notice that 15 is common between two sequence and it calculated twice

Now it is need to remove the common number from the equation.

Therefore, the final sequence will be

5(1+2+3+4+5+….N) + 3(1+2+3+4+5+….M) – 15 (1+2+...K)

**Note:**

* For two divisors [x, y] the total sum must subtract the common from the results because it will calculate twice for each number.
* For three divisors [x, y, k] and after use the calculation in the previous point, then it is need to add common of three divisors
* And so on, 4 divisors, it is to subtract common of three divisors
* For Divisors with count L if L Odd, it is need to add common of L divisors
* Else if L is even, it is need to add common of L divisors

For example

Divisors [2, 3] for limit N there are common 6, 12, 24, …. 6K where K:1,2,3

The previous sequence will repeat when calculated sum for both divisors separate

Then it is need to remove common sequence once form the calculation.

If the Divisor [2, 3, 5] the common sequence is 15, 30 ,45,….15K where K:1,2,3

When apply subtract the sequence of twins the sequence of the triple will subtract triple so it is need to added once more.

Pseudocode

*Foreach int d in divisors*

*Loop from d to (limit/d) step d*

*Add number x to the accumulated result*

*Multiplies the divisors for each other and then*

*Loop from d to (limit/d) step d*

*Subtract number x to the accumulated result*

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Time Complexity | O(N/K+N/L – N/(L\*K)) | N : limit provided  K,L, .. are *divisors* |
| Space Complexity | O(1) |  |

### Approach C

After approach B, it is noticed that there is equation can be used to get the sum of multipliers instead on jump on certain sequence. The benefit of this the complexity can narrow down to constant number.

Pseudocode

*Foreach int d in divisors*

*n:= limit/d*

*result += ((1 + n) \* n / 2) \* m;*

*Multiplies the divisors for each other and then*

*Loop from d to*

*n:= limit/d*

*result -= ((1 + n) \* n / 2) \* m;*

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Time Complexity | O(C) ~ O(1) | C: is constant |
| Space Complexity | O(1) |  |

## Statistical Running

After running the approaches for different data input, here are the result.

Reference file Statistics.xlsx

|  |  |  |  |
| --- | --- | --- | --- |
|  | Sum Of Multiple | | |
|  | Time In MS | | |
| **To Limit** | **Approach A** | **Approach B** | **Approach C** |
| 1000 | 1 | 0 | 0 |
| 10000 | 1 | 0 | 0 |
| 100000 | 9 | 3 | 0 |
| 1000000 | 84 | 29 | 0 |
| 10000000 | 757 | 293 | 0 |
| 100000000 | 7715 | 2819 | 0 |

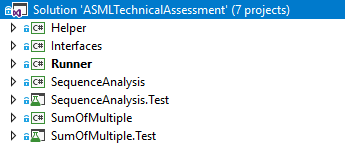
# Testing Solution

To prove the solution for both problem and according to what is minimum required.

New Solution “ASMLTechnicalAssessment” contains the following projects

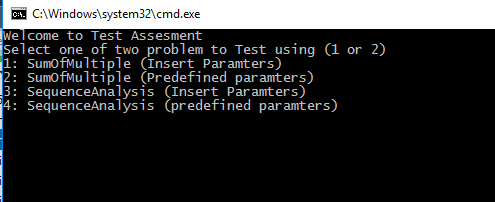
Using Visual studio 17 and .Net Framework 4.7

|  |  |
| --- | --- |
| Project Name | Usage |
| Helper | Contains any common helper methods like  Time tracker , string generator, … |
| Interface | Interfaces for Sequence Analysis & Sum Of Multiple |
| Runner | It will ask the user which of the problems below to solve |
| SequenceAnalysis | Solve the issue of Sequence Analysis with two approaches |
| SequenceAnalysis.Test | Test Solution of the issue of Sequence Analysis with two approaches |
| SumOfMultiple | Solve the issue of Sum Of Multiple with two approaches |
| SumOfMultiple.Test | Test Solution of the issue of Sum Of Multiple with two approaches |



For the Runner project

When user run it, the next screen will appear



|  |  |
| --- | --- |
| Selection Number | Description |
| 1 | this method is used to test Sum Of Multiple  it allow user to test with inserting limit parameters  and also defined the divisors |
| 2 | this method is used to test Sum Of Multiple  it uses predefined limit parameters start from 1000 ,10000,...., 100000000  Also predefined the divisors [2, 3, 5, 7]. |
| 3 | this method is used to test Sequence Analysis  it allow user to test with inserting string parameter |
| 4 | this method is used to test Sequence Analysis it uses  a predefined string parameter that is randomly generated  by method Random String that takes a length of string  In addition, returns a random string. |
| Esc | To exist |