Develop a microservice-based Web API applications

Description:

* Yellow text – not required tasks, but would be a plus if you implement it.
* Develop a simple microservice-based Web API application (with controllers) that performs CRUD (Create, Read, Update, Delete) operations on a database.
* The application should have two microservices: one for handling user information, and another for handling projects. use the FluentRest library for this task. It can be found on NuGet
* Microservices should use a SQL database (PostgreSQL) for users and a NoSQL database (MongoDB) for projects.
* The application should be containerized using Docker and docker-compose.
* Write a few unit tests for the application using xUnit and Moq.
* Write 1 integration tests for the application to ensure that the microservices are communicating correctly and the data is being stored and retrieved correctly from the databases.

Deliverables:

* The source code of the application.
* Solution that runs 2 Web API services and databases in docker-compose.

Additional information:

PostgreSQL database should contains 2 tables:  
users:  
- id  
- name  
- email  
- subscriptionid  
  
Samples of data:

|  |  |  |  |
| --- | --- | --- | --- |
| id | name | email | subscriptionId |
| 1 | John Doe | John@example.com | 2 |
| 2 | Mark Shimko | Mark@example.com | 5 |
| 3 | Taras Ovruch | Taras@example.com | 6 |

subscriptions:  
- id  
- type (“Free”/”Trial”/”Super”)  
- startDate  
- endDate

Samples of data:

|  |  |  |  |
| --- | --- | --- | --- |
| id | type | startDate | endDate |
| 1 | Free | 2022-05-17 15:28:19 | 2099-01-01 00:00:00 |
| 2 | Super | 2022-05-18 15:28:19 | 2022-08-18 15:28:19 |
| 3 | Trial | 2022-05-19 15:28:19 | 2022-06-19 15:28:19 |
| 4 | Free | 2022-05-20 15:28:19 | 2099-01-01 00:00:00 |
| 5 | Trial | 2022-05-21 15:28:19 | 2022-06-21 15:28:19 |
| 6 | Super | 2022-05-22 15:28:19 | 2023-05-22 15:28:19 |
| 7 | Super | 2022-05-23 15:28:19 | 2023-05-23 15:28:19 |

-------------------------------------------------  
MongoDB database should contains 2 collections:  
user.settings  
- userId  
- language (“English”/”Spanish”)  
- theme (“light”/”dark”)  
  
projects  
- userId  
- name  
- charts (array)  
 - symbol (“EURUSD”/“USDJPY”)  
 - timeframe (“M1”/”M5”/”H1”)  
 - indicators (array)  
 - name (“MA”/ “BB”/ “RSI”/ “Ichimoku”)  
 - parameters (string)  
  
Samples of data:

|  |  |
| --- | --- |
| {  "userId": 3,  "name": "my super project 1",  "charts": [  {  "symbol":"EURUSD",  "timeframe": "M5",  "indicators": []  },  {  "symbol":"USDJPY",  "timeframe": "H1",  "indicators": [  {  "name": "BB",  "parameters" : "a=1;b=2;c=3"  },  {  "name": "MA",  "parameters" : "a=1;b=2;c=3"  }  ]  }  ]  } | {  "userId": 3,  "name": "my super project 2",  "charts": [  {  "symbol":"EURUSD",  "timeframe": "M5",  "indicators": [  {  "name": "MA",  "parameters" : "a=1;b=2;c=3"  }  ]  }  ]  } |
| {  "userId": 3,  "name": "my super project 3",  "charts": []  } | {  "userId": 2,  "name": "project 1",  "charts": [  {  "symbol":"EURUSD",  "timeframe": "H1",  "indicators": [  {  "name": "RSI",  "parameters" : "a=1;b=2;c=3"  }  ]  }  ]  } |
| {  "userId": 2,  "name": "project 2",  "charts": [  {  "symbol":"USDJPY",  "timeframe": "H1",  "indicators": [  {  "name": "MA",  "parameters" : "a=1;b=2;c=3"  }  ]  }  ]  } | {  "userId": 1,  "name": "project 3",  "charts": [  {  "symbol":"EURUSD",  "timeframe": "M5",  "indicators": [  {  "name": "RSI",  "parameters" : "a=1;b=2;c=3"  },  {  "name": "MA",  "parameters" : "a=1;b=2;c=3"  }  ]  }  ]  } |

In second API (which work with MongoDb) add endpoint that gives an answer on a question: Top 3 most used indicator names (in projects) by subscription type Super

Url  
/api/popularIndicators/{sybscriptionType}  
  
parameter subscriptionType: string with values “Free”/”Trial”/”Super”  
  
Required answer:  
{

"indicators": [

{

"name":"MA",

"used": 3

},

{

"name":"BB",

"used": 1

},

{

"name":"RSI",

"used": 1

}

]

}