**REST API and NoSQL**

Documentation

Created by

**UPTOWN IT**

For

**<<CUSTOMER>>**

**PROJECT REFERENCE:**

**DATE:**

**CONTENTS**

[PART ONE – NoSQL Solution 2](#_Toc103206013)

[NoSQL research and technology selection 2](#_Toc103206014)

[MongoDB setup 3](#_Toc103206015)

[Partitioning 3](#_Toc103206016)

[Indexing and TTL in NoSQL 3](#_Toc103206017)

[PART TWO – REST API 4](#_Toc103206018)

[Build and document REST API Project 4](#_Toc103206019)

[Review Project Requirements 4](#_Toc103206020)

[Selecting REST API framework and IDE 4](#_Toc103206021)

[Build the REST API: Endpoints and methods 4](#_Toc103206022)

[Test methods 5](#_Toc103206023)

[Enable Cross-origin Resource Sharing (CORS) 5](#_Toc103206024)

[Evaluate and secure REST API 6](#_Toc103206025)

[Documenting REST API endpoints 6](#_Toc103206026)

[PART THREE – REST API and MongoDB 7](#_Toc103206027)

[REST API and MongoDB Integration 7](#_Toc103206028)

[Further queries or object interactions 7](#_Toc103206029)

[Finalising the project 8](#_Toc103206030)

[Appendix 9](#_Toc103206031)

# PART ONE – NoSQL Solution

# NoSQL research and technology selection

Introductory paragraph

* Review and analyse the scenario presented with the aim of identifying the business data needs and confirming that NoSQL is the best option for this project.
* Outline in detail at least three (3) reasons to justify the use of NoSQL for the scenario presented.
* Research vertical (scale-up) and horizontal (scale-out) scaling methods and identify the reasons why horizontal scaling is better suited for the scenario presented. Review the business requirements. Explain in detail at least two (2) benefits of using horizontal scaling for the project.
* Research and compare relational databases (SQL-based) and non-relational (NoSQL) databases. Complete the table below. Write your appraisal in the corresponding cells. A “tick” will not be considered an appropriate answer.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TYPE of DATABASE | SUITABILITY by DATA VOLUME | SUITABILITY by DATA TYPE  Structured, Semi-Structured, Unstructured | QUERY COMPLEXITY | ACID COMPLIANCE |
| Relational: SQL |  |  |  |  |
| Non-relational: NoSQL |  |  |  |  |

* Research and review at least three (3) NoSQL interfaces or vendor technologies. This information will be used by the manager and/or relevant personnel to discuss and select the most appropriate NoSQL solution.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NoSQL Interface | MAIN FEATURES | DATA FORMATS | EASE OF USE | BEST FOR  Types of Projects |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Add rows as necessary |  |  |  |  |

You have been informed that the NoSQL interface selected for the project is MongoDB. Keep that in mind while completing the tasks below.

## MongoDB setup

Introductory paragraph

* Identify and design the data storage requirements for the type of data used in this project.
* Review the business needs presented, and select the most appropriate data (formats) types for the NoSQL data store for this project.
* Create the necessary document/collection schema in MongoDB for the scenario presented. Document the datastore structure and its notation, e.g. BSON. It can be presented as a combination of text and graphics.
* Source the data from the dataset provided and populate the data stores.

## Partitioning

Introductory paragraph

* Identify and design the data storage requirements.
* Identify at least one (1) document/collection that could benefit from partitioning. Determine the partition key and the sort key to control the order in which the data will be stored in the partition. Document your solution.
* Create the partition ensuring that the partition key and the sort key match the business requirements. Provide screenshots as evidence.
* Review the partition created above. Outline a strategy to achieve an effective distribution of storage across the partition.

## Indexing and TTL in NoSQL

Introductory paragraph

* Calculate and determine how read and write through-puts are going to be implemented in the NoSQL database. Document the process and provide calculation details.
* Determine indexing needs to suit the scenario presented and complete the following:
* Configure, and create a single field index in a collection to optimise data retrieval.
* Configure and create either a multikey index or a compound index to optimise data retrieval.
* Determine and implement a time-to-live (TTL) or special single-field index on a field in a collection. The TTL index will automatically remove the document after a certain amount of time has passed or a specific clock time. Refer to the scenario presented for the required TTL actions.
* Present the completed MongoDB database setup to your manager or relevant person in the organisation for approval and signoff.

|  |  |
| --- | --- |
| **MongoDB database SIGNOFF**  Signing off on this document signifies that the **MongoDB database setup** presented **complies** with the Client’s Business **requirements.** | |
| Project Manager or relevant stakeholder  Signature:  Date: | Web Developer  Signature:  Date: |
| **Documentation NOT APPROVED**  Please provide feedback on the changes needed. | |
| **APPROVAL**  Granted  Not Granted | |

# PART TWO – REST API

# Build and document REST API Project

## Review Project Requirements

Introductory paragraph

Read and analyse the scenario presented and justify the need for a REST API solution for the given scenario. In the justification address the following points:

* Suitability to the scenario presented
* Flexibility and portability
* Scalability
* Cacheable

## Selecting REST API framework and IDE

Introductory paragraph

Review Web API frameworks available to your chosen programming language, select a framework that you feel is suitable and justify your selection by providing a detailed description of how this Framework will meet the needs described in 1.1.

Identify the framework selected and the IDE environment that you are going to use.

## Build the REST API: Endpoints and methods

Introductory paragraph

Create and configure at least one endpoint for each of the following points - For each method, create an endpoint(s) that corresponds with the requirements specified in the scenario provided.

1. Return a single value, using the GET method
2. Return a collection of values, using the GET method
3. Add a provided value to a collection, using the POST method
4. Update an existing value with a provided value, using the PUT method
5. Remove a specified item from a collection using the DELETE method

## Test methods

Introductory paragraph

Test methods (GET, POST, PUT & DELETE) and check that the response status codes obtained match the business requirements/needs.

If necessary, modify and re-test until all tests pass. Complete the table below and provide screenshots as evidence.

|  |  |  |  |
| --- | --- | --- | --- |
| METHOD | ENDPOINT | METHODS RESPONSE STATUS CODE | MATCH OCCURRED  Yes/No |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Add rows as necessary |  |  |  |

## Enable Cross-origin Resource Sharing (CORS)

Introductory paragraph

Enable CORS for GET, POST, PUT & DELETE methods and confirm that cross-origin requests are allowed. Provide screenshots of the actions.

Test cross-origin requests on client with at least four (4) instances. One per method will be sufficient.

Complete the table below and provide screenshots as evidence.

|  |  |  |  |
| --- | --- | --- | --- |
| INSTANCE/METHOD  GET, POST, PUT & DELETE | CROSS-ORIGIN REQUEST | OUTPUT  RESULT | COMMENTS |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Add rows as necessary |  |  |  |

Update the requests completed in the previous task to enable pre-flight cross-origin using the OPTION method. Test the pre-flight requests. Complete the table below and provide screenshots as evidence.

|  |  |  |  |
| --- | --- | --- | --- |
| INSTANCE/METHOD  GET, Post, PUT and Delete | PRE-FLIGHT REQUEST | OUTPUT  RESULT | COMMENTS |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Add rows as necessary |  |  |  |

## Evaluate and secure REST API

Introductory paragraph

Describe the method chosen to secure the application, as well as the individual endpoints that will require authentication and authorisation as per the business requirements provided.

|  |  |  |  |
| --- | --- | --- | --- |
| ENDPOINT NAME | HTTP METHOD | AUTHORISATION DETAILS | AUTHENTICATION REQUIRED |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Add rows as necessary |  |  |  |

Implement the authentication and authorisation method proposed in the previous task.

Test the method and provide screenshots as evidence.

## Documenting REST API endpoints

Introductory paragraph

Compare and evaluate at least three (3) API frameworks that provide the necessary tools to document endpoints and it is compatible with OPEN REST API specification.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TOOLS | MAIN FEATURES | COST  Free/Paid | EASE OF USE | BEST FOR  Type of Project | OVERALL RATING  (L)1-5 (H) |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  | Add rows as necessary |  |  |  |

In discussion with your manager and/or relevant personnel, it has been decided that Swagger will be the application used for API definition and documentation.

Use the selected application for the REST API endpoints documentation. Extend the documentation to include more information, e.g., methods, resources, parameters, requests/responses and several.

Provide screenshots of the endpoints documentation or URL and access details.

Validate documentation against endpoints and submit it to your manager or relevant person in the organisation for approval.

|  |  |
| --- | --- |
| **REST API Endpoints Documentation** **SIGNOFF**  Signing off on this document signifies that the **documentation** presented **complies** with the Client’s Business **requirements.** | |
| Project Manager or relevant stakeholder  Signature:  Date: | Web Developer  Signature:  Date: |
| Documentation NOT APPROVED  Please provide feedback on the changes needed. | |
| **APPROVAL**  Granted  Not Granted | |

# PART THREE – REST API and MongoDB

# REST API and MongoDB Integration

Introductory paragraph

* Using the REST API researched, designed, created and tested in PART 1, use the client API to interact with the NoSQL database for this project. Test connection.
* Perform the operations listed below. All the operations must be as per the business requirements specified in the scenario presented. If you completed some of these queries in PART 2, while testing the API, you can reuse them in this section.

For each operation, use the corresponding method and endpoint(s). Provide a screenshot of each query and its output.

1. Insert a single data object
2. Perform multiple inserts in a single operation
3. Query a single object
4. Use batch object to retrieve multiple objects in one operation
5. Include an index in a query
6. Create and run a query that returns multiple attributes
7. Delete a single object
8. Delete multiple objects
9. Update a single object
10. Update multiple objects

## Further queries or object interactions

Introductory paragraph

To ensure data persistence, configure the NoSQL database to accept persistence of objects including objects of different data types. Set the configuration as per business requirements. Test and modify the configuration until the desired result is achieved.

Provide screenshots of the configuration file and options.

In accordance with the scenario provided:

1. Propose at least two (2) triggers for the project and identify the corresponding events and notifications required. Refer to the scenario presented for the required trigger actions. Email your manager or relevant stakeholder seeking confirmation of the triggers before implementing them. Provide email as evidence.
2. Modify triggers, if necessary, based on feedback received. Then, implement the triggers and test them. For each trigger, provide screenshots as evidence.

Review and confirm that the following features are working according to business access requirements. Modify and retest as necessary. Test each one of them and provide screenshots.

1. Data encryption
2. Database authorisation
3. Database authentication

## Finalising the project

Complete documentation.

Notify the manager or relevant person in the organisation of the project completion and seek approval.

|  |  |
| --- | --- |
| **REST API and NoSQL Projects**  **SIGNOFF**  Signing off on this document signifies that the project and **documentation** presented **complies** with the Client’s Business **requirements.** | |
| Project Manager or relevant Stakeholder  Signature:  Date: | Web Developer  Signature:  Date: |
| Documentation NOT APPROVED  Please provide feedback on the changes needed. | |
| **APPROVAL**  Granted  Not Granted | |

# Appendix

You can add copies of coding to the appendix