Rockstone Interactive Dashboard

System Requirements (Version: 1.1)

Iona Pitt, Kyle Roberts, Danny Agha, Josh Clarke, Luke Wood

COM617 – Industrial Consulting Project

29/01/2024

# Administration page

|  |  |
| --- | --- |
| **Customer Information** |  |
| Project title | Rockstone Interactive Dashboard |
| Customer Organisation | Rockstone Data |
| Customer contact | Nick Thorne |
| Date due | 17/05/2024 |

|  |  |  |
| --- | --- | --- |
| **Principal authors** |  |  |
| Iona Pitt | 6pitti03@solent.ac.uk | Product Owner / Developer |
| Kyle Roberts | 6robek81@solent.ac.uk | Developer |
| Danny Agha | 5aghad30@solent.ac.uk | Developer |
| Josh Clarke | 5clarj31@solent.ac.uk | Scrum Master / Developer |
| Luke Wood | 5woodl59@solent.ac.uk | Head of Stakeholder Communications / Developer |

| **Record of changes** | | |  |
| --- | --- | --- | --- |
| Issue | Date | Detail of Changes | Changes Made By |
| 1.0 | 29/01/2024 | Document created and initial requirements added | Iona Pitt |
| 1.1 | 02/02/2024 | Additional terms added to glossary and extra references | Iona Pitt |
|  |  |  |  |
|  |  |  |  |

# Abstract / Executive Summary

## Background

Rockstone Data is a software application development company founded in 2018 by Nick Thorne. [1]

The company has recently utilised a new column-orientated database called ClickHouseDB to host very large time series data tables. This class of database outperforms traditional row-orientated databases in both speed and storage.

The project aim is to create an interactive demonstrator running on Rockstone Data’s website clearly demonstrating these benefits vs ‘traditional’ or row-orientated databases.

Table of Contents

[Administration Page 2](#_heading=h.gjdgxs)

[Abstract / Executive Summary 3](#_heading=h.30j0zll)

# 1 Introduction

## 1.1 Purpose of the document

This document is a detailed list of requirements agreed upon by the stakeholders of the project and the apprentices of Southampton Solent University. It will thoroughly explain an overview of the project, the priority of its requirements, the scope of the project and the plan of work to complete each sprint.

## 1.2 Scope of the software

The system will be a Dashboard web application hosted on Docker containers, featuring a line plot depicting comparison data from ClickHouseDB. Our group will design, develop, test and deploy the software according to the requested requirements supplied by the project sponsor, Rockstone Data.

The work will be divided into three sprints, between 22nd January and 17th May 2024, with the first sprint over three weeks, the second over four, and the third over eight. The requirements and project plan will be drafted in the first sprint, including role and task allocation.

In the second sprint, the team shall provide a working Proof of Concept, application design documentation, a test plan, and a supporting presentation to the project sponsor and course tutor. Lastly, in the third sprint, the team will conclude development and deploy the software. They will have a stand-up meeting once a week where the team will discuss project progress and any blockers.

Our group must produce a Project Initiation Document defining the project stages and the work that needs to be done.

# 2 General Description

## 2.1 Function and purpose

The system must be able to display the differences between ClickHouseDB’s column-orientated format and a traditional database’s row-orientated format using a line plot. In addition, the application could also display this information through geo-heatmaps or data analyses using Snowflake and Data Dog.

## 2.2 Environment

The application will be written using Python within a virtual environment. It will primarily be developed on Windows and Apple iOS operating systems, and will be used by the engineers and customers of Rockstone Data. It must be hosted using Docker Compose containers during development, however may later be deployed using Amazon Web Services EC2.

## 2.3 Relation to other systems

The application will provide and receive data from ClickHouseDB and Postgres. The two databases shall be hosted by separate Docker containers.

## 2.4 General Constraints

The Project Initiation Document must be produced and submitted by Friday 10th May 2024.

The software artefact must be completed and submitted by Friday 17th May 2024.

While not a heavy constraint, we were recommended not to use PyCharm for development as it may be harder to debug any potential issues.

# 3 Specific Requirements

Priority:

1 - High

2 - Medium

3 - Low

## 3.1 Functional requirements

### 3.1.1 User interface requirements

* 1 - The application shall be a one-page Dashboard web application.
* 1 - The user interface shall display a line plot of a scalar value over time.
* 1 - The user interface shall display a line plot of a start / end datetime picker.
* 1 - The user interface shall display a line plot of a database source picker.
* 1 - The user interface shall have a ‘Submit’ button which fetches the data for the line plot.
* 1 - The user interface shall have a ‘Downsampling On/Off’ toggle.
* 1 - The user interface shall have a ‘Downsampling’ text entry form.
* 3 - The user interface may have a multidimensional class of data displayed as a geo-heatmap or spectrograph or surface plot.
* 3 - The graphs may have additional animations or orientation controls.
* 3 - The multidimensional data may be shown below the line chart.
* 3 - The user interface may compare costs using Snowflake or Data Dog.
* 3 - The application may benchmark writing data to large tables.
* 3 - The application may benchmark the MongoDB time-series collections.

## 3.2 Interface requirements

### 3.2.1 Database requirements

* 1 - The database source shall use either Postgres, Postgres with TimescaleDB or ClickHouseDB.

### 3.2.2 Other interface requirements

* 1 - The team shall use the ‘Issues’ page on GitHub for logging tasks and communication.
* 1 - The application shall be developed using Visual Studio Code.

## 3.3 Operational requirements

### 3.3.1 Docker requirements

* 1 - For local development, the databases shall be run inside Docker containers.
* 1 - There shall be one Docker container for Postgres.
* 1 - There shall be two Docker containers for ClickHouseDB, and they shall be named ‘chc’ and ‘chs’ respectively.
* 1 - Docker Compose shall be used in deployment for starting and stopping the database Docker containers.
* 2 - There may be a Docker container for Postgres and TimescaleDB.

## 3.4 Resource requirements

* 1 - The application shall be deployed using a Python ‘venv’.
* 1 - The application’s packages shall be installed using Python ‘pip’.
* 3 - The application may be deployed using AWS EC2 or DigitalOcean.

## 3.5 Documentation requirements

* 2 - The team may use the ‘Wiki’ page on GitHub for system documentation.

## 3.6 Security requirements

None at present.

## 3.7 Testing requirements

* 1 - Test data shall contain visible results on a 24 hour basis
* 1 - Test data shall contain visible results on a 1 month basis
* 1 - Test data shall contain visible results on a 1 year basis
* 1 - Test data shall contain a once or second basis for 1 to 100 years

# 4 References

[1] Rockstone Data - <https://www.rockstonedata.co.uk/about>

[2] ClickHouseDB - <https://clickhouse.com/>

[3] TimescaleDB - <https://www.timescale.com/>

[4] PostgreSQL - <https://www.postgresql.org/>

[5] MongoDB - <https://www.mongodb.com/docs/>

[6] Docker Compose - <https://docs.docker.com/compose/>

[7] AWS - <https://aws.amazon.com/>

[8] EC2 - <https://aws.amazon.com/ec2/>

[9] Snowflake - <https://www.snowflake.com/en/>

[10] Data Dog - <https://www.datadoghq.com/>

[11] DigitalOcean - https://www.digitalocean.com/

# Glossary

*<List and explain specialised terms and abbreviations>*

Project Initiation Document: Summary of important information about the project, including its context, scope, milestones, objectives and requirements.

Proof of Concept: Gathering evidence to gauge the feasibility of a project.

ClickHouseDB: An open-source database which is often used for queries involving real-time and historical data.

Postgres (PostgreSQL): An open-source relational database often used for data storage for web, mobile, geospatial and analytics applications.

TimescaleDB: A database engine that uses Postgres for applications requiring time series, vector, events and analytics data.

MongoDB: A document database often used for scaling throughout application development.

Docker / Docker Compose: A tool used to run multi-container applications. Each container provides different back-end functionality for the application.

DBMS (Database Management System):

AWS (Amazon Web Services): An infrastructure platform which provides on-demand cloud computing and APIs for deploying applications.

EC2 (Amazon Elastic Compute Cloud): A compute platform used for deploying applications on the cloud.

OLAP (Online Analytical Processing): A form of computing whereby the user can extract and analyse data from different perspectives.

OLTP (Online Transaction Processing): A form of data processing whereby the user can execute many transactions at the same time.

Snowflake: Software as a service which can be used for data warehouses, lakes, sharing of data, data science and engineering, and data application management.

Data Dog: Software as a service which can be used for monitoring of servers, databases, tools and services.

DigitalOcean: Software and infrastructure as a service which can be used for deploying web applications.