Enterprise Frameworks

Mapsago

Requirements Specification

Team Members

● Fiona Mc Andrew, Fiona.Mcandrew@student.ncirl.ie

● Andrew Burnett, Andrew.Burnett@student.ncirl.ie

● Adam Harrington, Adam.Harrington@student.ncirl.ie

Course: MSc Web Technologies (part-time)

Module: Enterprise Frameworks

Lecturer: Vikas Sahni

[Project Blog](https://enterpriseframeworks.wordpress.com/) Username and Password required

# 1.0 - Introduction

## 1.1 - Purpose

The purpose of this document is to set out the requirements for the development of an enterprise grade application to support a public map-based educational rich internet application.

The intended customers are defined as follows:

User A: *RIA*

User A is a rich internet application (separate application) designed to display data it requests from the application

User B: *Data Curator*

User B is concerned with sourcing, managing and approval of data from third party sources for use in the application.

User C: *Owner*

User C is the overarching owner of the application and the related RIA. This user is concerned with how the system works together and with managing Data Curators (User B)

## 1.2 - Project Scope

The scope of the project is to develop an application to facilitate the RIA through provision of structured data for user presentation. The application will also be concerned with the sourcing and management of the data.

The system shall also allow for abstracted data manipulation by the administrative users; the data curators. This will allow for the creation, reading, updating and deletion (CRUD) of data.

Data shall be sourced through programmable commands accessing third-party web content. Data persistence will be managed throughout the application.

### 1.2.1 - Constraints

* The client has stipulated use of the ASP.NET and C# technology stack.
* Date for completion has been tabled for the 29th April 2014.

## 1.3 - Definitions, Acronyms, and Abbreviations

Application Used to refer to the application being developed as opposed to the RIA or the client application

RIA Rich Internet Application, AKA the client Application

Client Application Used to refer to the rich internet application which will be a primary user of the applications

Management Interface Used to refer to the interface which will be operated by the Data Curator

Data Curator Used to refer to the personnel in charge of the Management Interface.

Owner Used to refer to the overarching owner of the application and the related RIA.

GUI Graphic User Interface.

UR Used to refer to the User Requirement

SR Used to refer to the System Requirement

IR Used to refer to the Interface Requirement

FR Used to refer to the Functional Requirement

DR Used to refer to the Documentation Requirement

DTR Used to refer to the Data Requirement

# 

# 2.0 - Contents

[1.0 - Introduction 2](#_Toc381787308)

[1.1 - Purpose 2](#_Toc381787309)

[1.2 - Project Scope 2](#_Toc381787310)

[1.2.1 - Constraints 2](#_Toc381787311)

[1.3 - Definitions, Acronyms, and Abbreviations 3](#_Toc381787312)

[2.0 - Contents 4](#_Toc381787313)

[3.0 - User requirements 5](#_Toc381787314)

[3.1 - User A - (Client API - Machine) 5](#_Toc381787315)

[3.2 - User B - (Data Curator - Human): 5](#_Toc381787316)

[3.3 - User C - Owner 5](#_Toc381787317)

[4.0 - System Architecture 6](#_Toc381787318)

[5.0 - Functional requirements definition 7](#_Toc381787319)

[6.0 - Interface requirements 10](#_Toc381787320)

[7.0 - Documentation Requirements 12](#_Toc381787321)

[8.0 - Data requirements 13](#_Toc381787322)

[9.0 - Non-Functional Requirements 14](#_Toc381787323)

[9.1 - Availability requirement 14](#_Toc381787324)

[9.2 - Recover requirement 14](#_Toc381787325)

[9.3 - Security requirement 14](#_Toc381787326)

[9.4 - Maintainability requirement 14](#_Toc381787327)

[9.5 - Portability requirement 14](#_Toc381787328)

[9.6 - Extendibility requirement 14](#_Toc381787329)

[9.7 - Reusability requirement 14](#_Toc381787330)

[9.8 - Resource utilization requirement 14](#_Toc381787331)

[9.9 - Robustness requirement 14](#_Toc381787332)

[10.0 - System evolution 15](#_Toc381787333)

[References 16](#_Toc381787334)

[Appendix 17](#_Toc381787335)

# 3.0 - User requirements

## 3.1 - User A - (Client API - Machine)

This user is indifferent to data in the database, their concern lies in the delivery of the queried data between the interfaces.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| User Requirement | | UR | FR | SR | IR | DR | DTR |
| UR1 | The User must to able to send queries to the dataset, based on a structured format | - | 1 |  | 1 |  |  |
| UR2 | The User must receive data back from a request in a structured and expected format | - | 2 |  | 1 |  |  |

## 3.2 - User B - (Data Curator - Human):

This user is concerned with data in the database, it is their job to populate and curate it.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| User Requirement | | UR | FR | SR | IR | DR | DTR |
| UR3 | The user must be able to source new content | - | 3 |  |  |  |  |
| UR4 | The User must be able to access and edit existing records within the application | - | 4 |  |  |  |  |
| UR5 | The User must be able to access the application, through most modern browsers | - | 8 |  |  |  |  |
| UR6 | The User must be able to log in to the admin only interface, by using secure and private authorisation | - | 6 |  |  |  |  |
| UR7 | The User should be able to view changes / conflicts to the database | - | 4 |  |  |  |  |
| UR8 | The User should be able to preview changes made by themselves and other members of the curation team | - | 3,4 |  |  |  |  |
| UR9 | The User should be able to create a repeatable web search, to source new content | - | 3 |  |  |  |  |
| UR10 | The User should not need comprehensive technical knowledge to use the application | - | 6 |  |  |  |  |
| UR11 | The User should be able to use all the system functions after a total of two hours training | - |  |  |  | 2 |  |
| UR12 | The User should be able to refer to a help document, if they are unsure of the task at hand | - |  |  |  | 1 |  |
| UR13 | The User should receive user friendly error messages to notify users of errors in the application | - | 7 |  |  |  |  |

## 3.3 - User C - Owner

This user is concerned with how the system works together and with managing Data Curators (User B).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| User Requirement | | UR | FR | SR | IR | DR | DTR |
| UR14 | The User should be able to view sensitive data in error pages, error messages, log files, and audit files | - | 5 |  |  |  |  |
| UR15 | The User could be able to the use the application for a different topics, other than history, if required | - |  |  |  |  |  |

# 4.0 - System Architecture

The application is designed to employ a number of architectures including MVC and SOA. By providing well defined interfaces and loosely coupled layers the application will leverage performance of various platforms regardless of implementation, paving the way for scalability and adaptability.

This approach has major benefits for rich internet applications plugging into the API interface as all processing can be achieved in the business layer while a simplistic abstraction is exposed.

“Design to take advantage of client processing power. RIAs run on the client computer and can take advantage of all the processing power available there. Consider moving as much functionality as possible onto the client to improve user experience. Sensitive business rules should still be executed on the server, because the client logic can be circumvented”. (Microsoft, 2013)

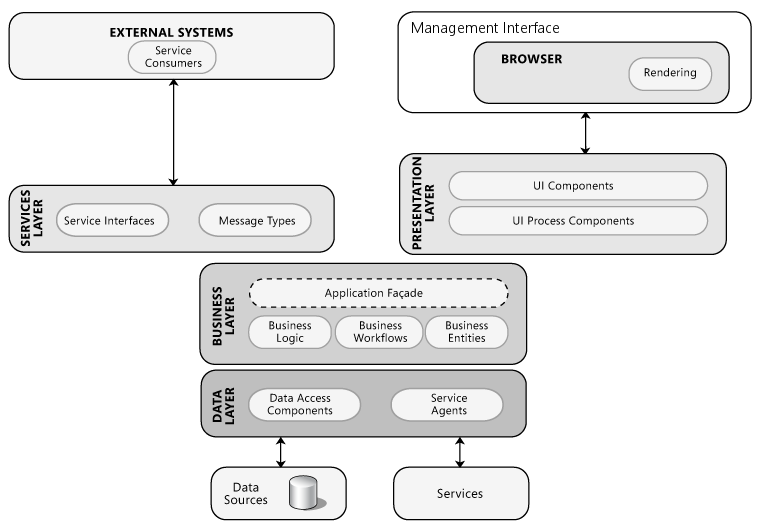


Figure 1 - Adapted from Microsoft's rich internet application and service archetypes

Aside from RIA interfaces, there is also the management interface will utilize multi-layered, or specifically an MVC architecture to provide a richer user experience (Microsoft, 2013).

Because much of the application involves sending messages without a need for a UI, the architecture will incorporate many of the characteristics of a service archetype (Microsoft, 2013). The service oriented elements include:

* Loosely coupling
* Complex
* Distributable
* Message oriented
* Technology agnostic

# 5.0 - Functional requirements definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR1 | Provide a documented RESTful API to access stored datasets | | | | |
| Description | The system must provide a documented API in order for external systems to retrieve the datasets they require. The API should be designed using best practices to cater for future changes to the system. | | | | |
| Trigger | The user (User A, client machine) sends requests using the API to acquire datasets. | | | | |
| Technical Issues | The API needs to be designed in a robust and evolvable manner. | | | | |
| Risks | Invalid API requests need to be handled gracefully. | | | | |
| Priority | Level 1, essential to system operation. | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **1** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR2 | API requests return structured data in JSON format | | | | |
| Description | Client requests to the system, using the public API, will return the requested data in JSON format. The format of the return JSON data should be fully documented along with the API specification | | | | |
| Trigger | The user (User A, client machine) gets a response in JSON format after a successful request using the API | | | | |
| Technical Issues | Need to ensure all responses are valid JSON | | | | |
| Risks | Any errors in API requests must return valid JSON data in response. The returned JSON should indicate an error has occurred, so the client can handle it correctly | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | FR1 | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **2** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR3 | Provide ability to create new data sets | | | | |
| Description | It must be possible to create new data sets easily from predefined sources | | | | |
| Trigger | The user (User B, Data Curator) creates new data sets by interacting with a simple search like interface | | | | |
| Technical Issues | It should be possible for a developer to add new sources to the system when required without altering the existing functionality | | | | |
| Risks | Any errors that may occur during the process of adding new data must be clearly communicated to the user. Incomplete additions to database must be rolled back | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **3,8,9** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR4 | Provide ability to view and edit existing data sets | | | | |
| Description | An interface should be provided to allow manipulation of existing data stored in the database. Different data sets could be displayed in a table and a simplified map to visualize the geospatial data | | | | |
| Trigger | The user (User B, Data Curator) can view and edit existing data sets by using a simple browser based interface | | | | |
| Technical Issues | Legacy browsers will not be supported | | | | |
| Risks | Errors must be handled and database integrity must be assured by rolling back any unsafe changes | | | | |
| Priority |  | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **5,7,8** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR5 | Need a secure authentication system to control access | | | | |
| Description | Only approved users should have access to the management interface. This should be achieved by requiring users to authenticate themselves with the system by using a secure login interface | | | | |
| Trigger | The user logs in to the system using secure login credentials, such as username and password | | | | |
| Technical Issues | Users need to be approved for access. An administrator user account is needed with privileges to add and remove users | | | | |
| Risks | Need to be able to recover or reset lost or forgotten credentials | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **6** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR6 | Provide documentation for all users of the system | | | | |
| Description | API documentation should be publicly available and clearly structured to allow clients use the API effectively. The management interface should include integrated help pages to explain the essential functions. The codebase should be well documented to allow future developers extend the application | | | | |
| Trigger | User A operates according to the API specification. User B can access help pages within the system, which explain how to carry out the main tasks | | | | |
| Technical Issues | None | | | | |
| Risks | Must avoid incorrect documentation. Documentation should be kept up to date with the functionality of the system as it evolves | | | | |
| Priority | Level 2, not essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **10** | **-** | **-** | **-** | **1,2,3** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR7 | Display and log informative messages when exceptions occur | | | | |
| Description | All errors and exceptions that occur in the system should be handled properly. If the exception can be handled by the system, it should be logged with a clear description of what happened. If user interactions are affected a message should be displayed to the user indicating what happened and the steps to take, if any, to correct the situation | | | | |
| Trigger | User A should be able to determine an error has occurred based on the response received from an API request. Other users should be informed of relevant errors through the interface | | | | |
| Technical Issues | As far as possible the system should deal with all errors in a graceful manner without the need for user interaction | | | | |
| Risks | Errors may occur in the exception handling system itself leading to illegal operation being performed. These modules must be tested thoroughly | | | | |
| Priority | Level 2, not essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **-** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FR8 | Deliver user content through a web browser | | | | |
| Description | The management interface should be a web application accessible from all modern web browsers. There should be no reliance on a particular physical location or operating system. Support for legacy browser will not be supported | | | | |
| Trigger | User’s B and C will interact with the management interface from a web browser | | | | |
| Technical Issues | Use cross-platform technologies to operate on all modern browsers | | | | |
| Risks | Possible difference in user experience on different platforms | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **4** | **-** | **-** | **-** | **-** |

# 6.0 - Interface requirements

This section describes how the software interfaces with other software systems and users; input and output. The diagram below illustrates the high lever goals of the system interfaces.

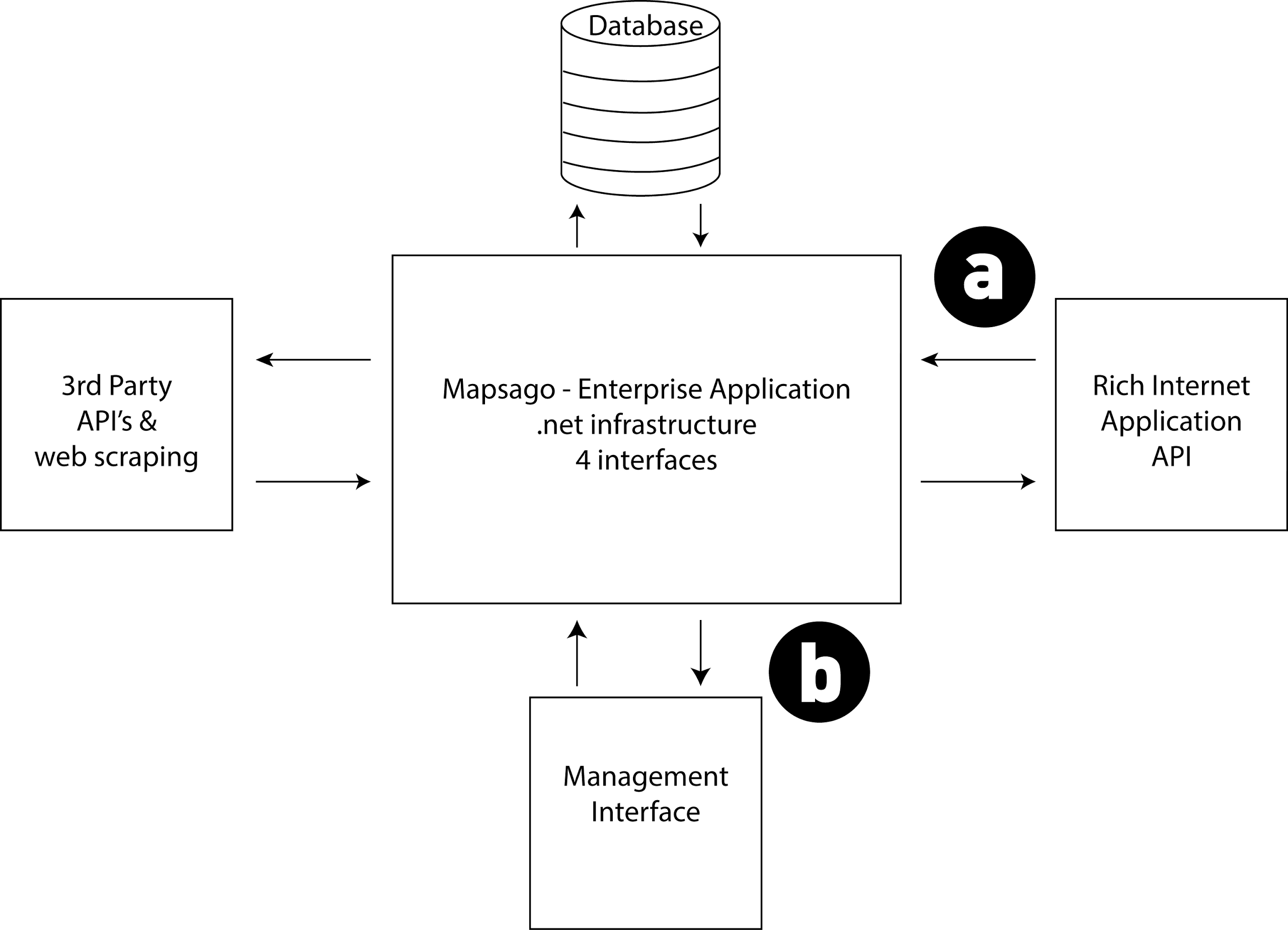


Figure 2 - Diagram of Interfaces

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| IR1 | Provide data to RIA | | | | |
| Description | Respond to structured requests with appropriate data output | | | | |
| Trigger | User A will send a structured request to a pre-defined URI | | | | |
| Technical Issues | The structure of the requests and responses must be defined and implemented in a meaningful way. Potentially using REST and JSON | | | | |
| Risks | Uncoordinated downtime of system could lead to poor service to clients | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | FR1 & FR2 | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **1,2** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| IR2 | Provide a user intuitive GUI | | | | |
| Description | The system interface should conform to usability heuristics, particularly providing for use of knowledge in the world over knowledge in the head and providing meaningful feedback | | | | |
| Trigger | Users B & C access the GUI management interface to perform tasks | | | | |
| Technical Issues | Feedback should be provided without delay which in a browser may require asynchronous communication | | | | |
| Risks | Usability testing may not be feasible so heuristic evaluation may need to be completed by the developers which can lead to bias, assumptions and oversight | | | | |
| Priority | Level 2, not essential to system operation | | | | |
| Dependencies | FR8 | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **6-13** | **-** | **-** | **-** | **-** |

# 7.0 - Documentation Requirements

There are three forms of documentation involved in the system:

* Help documentation for the Management GUI interface
* Reference documentation for the API provided for client applications
* System documentation for later extensions or modifications to the system

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DR1 | Integrate help documentation for User B into the interface | | | | |
| Description | Help for User B will be integrated into the management interface using some of the following: help page, joyrides, tour, hover tips | | | | |
| Trigger | - | | | | |
| Technical Issues | - | | | | |
| Risks | - | | | | |
| Priority | Level 2, not essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **-** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DR2 | DOcument structure and operation of API for User A | | | | |
| Description | As the API is defined the documentation must be maintained with priority placed on accuracy, relevance and completeness | | | | |
| Trigger | - | | | | |
| Technical Issues | - | | | | |
| Risks | - | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **11** | **6** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DR3 | Document the code base for extensibility and maintenance | | | | |
| Description | Need to provide a document that describes the structure of the code base. It should include details of interactions between modules and definitions of interfaces | | | | |
| Trigger | - | | | | |
| Technical Issues | - | | | | |
| Risks | - | | | | |
| Priority | Level 2, not essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **-** | **-** | **-** | **-** | **-** |

# 8.0 - Data requirements

The data requirements for the application weighed up the costs and benefits associated with local storage of data compared with those of processing frequent requests to external sites, making modifications to retrieved data and serving a response.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DTR1 | Data Acquired From External Sources Will Be Stored In The System Database | | | | |
| Description | Data sets retrieved from external sources will be processed and shaped into an appropriate format for use within the system. This will allow more efficient transfer of data between the server and User A, by allowing a single request to contain data from multiple sources | | | | |
| Trigger | - | | | | |
| Technical Issues | - | | | | |
| Risks | - | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **-** | **-** | **-** | **-** | **-** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DTR1 | Data must be manually approved for addition to the database | | | | |
| Description | Automatic addition of data into the system is fraught with danger. Incorrect or incomplete information will reduce the effectiveness of the system. User B must manually approve the data they are allowing into the system | | | | |
| Trigger | - | | | | |
| Technical Issues | - | | | | |
| Risks | - | | | | |
| Priority | Level 1, essential to system operation | | | | |
| Dependencies | None | | | | |
| Relations | **UR** | **FR** | **SR** | **IR** | **DR** |
|  | **-** | **-** | **-** | **-** | **-** |

# 

# 9.0 - Non-Functional Requirements

## 9.1 - Availability requirement

The data available to User A via an API should be available unless downtime is coordinated with the RIA.

## 9.2 - Recover requirement

Logs of changes must be kept by the system to help trace and/or undo unwanted changes to the system. Versioned backups of the database are also required to allow fast recovery of the system.

## 9.3 - Security requirement

Authentication and authorisation must be employed for the management interface (user B and user C) to protect the system from unauthorised tampering.

## 9.4 - Maintainability requirement

The code documentation should specify clearly the structure and organisation of the system to allow for speedy resolution to problems. The system should be modular and loosely coupled to ensure changes do not affect unrelated parts of the system.

## 9.5 - Portability requirement

The API should be designed to return consistent results regardless of the origin of the caller. The management interface should also work consistently across all modern browsers.

## 9.6 - Extendibility requirement

Loose coupling should ensure that adaptations or extensions can be made with confidence of not causing unexpected consequences within the system.

## 9.7 - Reusability requirement

The application should be built with reusability in mind, except where efforts to generalise the code base conflicts with other user requirements.

## 9.8 - Resource utilization requirement

Storage of data sets is restricted to only the information required to feed the API, reducing data storage in the database.

## 9.9 - Robustness requirement

The system must be scalable to tolerate a large number of request to the API. The management interface must also be able to accommodate multiple users accessing it at the same time.

# 

# 10.0 - System evolution

The purpose of the application is well defined in the present context, however, there will be a significant effort made to maintain generalisable to other similar implementations in the future. This is reflected in UR15 which states that the application could be cloned and modified for use with different topics, other than history, if required. A policy of loose coupling, DRY and avoiding hard-coding configuration should support this potential.

It is also possible that as the application develops there may be changes to the business model, requiring system-level changes to reflect new user requirements. Keeping well documented and defined interfaces should allow for these changes to be made with confidence that encapsulated or isolated changes can only have local impact.

# References

Microsoft, 2013. *Designing Rich Internet Applications.* [Online]   
Available at: http://msdn.microsoft.com/en-us/library/ee658083.aspx  
[Accessed 25 02 2014].

Microsoft, 2013. *Designing Service Applications.* [Online]   
Available at: http://msdn.microsoft.com/en-us/library/ee658114.aspx  
[Accessed 25 02 2014].

# Appendix

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| User Requirement | | UR | FR | SR | IR | DR | DTR |
| UR1 | The User must to able to send queries to the dataset, based on a structured format | - | 1 |  | 1 |  |  |
| UR2 | The User must receive data back from a request in a structured and expected format | - | 2 |  | 1 |  |  |
| UR3 | The user must be able to source new content | - | 3 |  |  |  |  |
| UR4 | The User must be able to access and edit existing records within the application | - | 4 |  |  |  |  |
| UR5 | The User must be able to access the application, through most modern browsers | - | 8 |  |  |  |  |
| UR6 | The User must be able to log in to the admin only interface, by using secure and private authorisation | - | 6 |  |  |  |  |
| UR7 | The User should be able to view changes / conflicts to the database | - | 4 |  |  |  |  |
| UR8 | The User should be able to preview changes made by themselves and other members of the curation team | - | 3,4 |  |  |  |  |
| UR9 | The User should be able to create a repeatable web search, to source new content | - | 3 |  |  |  |  |
| UR10 | The User should not need comprehensive technical knowledge to use the application | - | 6 |  |  |  |  |
| UR11 | The User should be able to use all the system functions after a total of two hours training | - |  |  |  | 2 |  |
| UR12 | The User should be able to refer to a help document, if they are unsure of the task at hand | - |  |  |  | 1 |  |
| UR13 | The User should receive user friendly error messages to notify users of errors in the application | - | 7 |  |  |  |  |
| UR14 | The User should be able to view sensitive data in error pages, error messages, log files, and audit files | - | 5 |  |  |  |  |
| UR15 | The User could be able to the use the application for a different topics, other than history, if required | - |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Functional Requirement | | **UR** | **FR** | **SR** | **IR** | **DR** | **DTR** |
| FR1 | Provide a documented RESTful API to access stored datasets | 1 | - |  |  |  |  |
| FR2 | API requests return structured data in JSON format | 2 | 1 |  |  |  |  |
| FR3 | Provide ability to create new data sets | 3,8,9 | - |  |  |  |  |
| FR4 | Provide ability to view and edit existing data sets | 5,7,8 | - |  |  |  |  |
| FR5 | Need a secure authentication system to control access | 6 | - |  |  |  |  |
| FR6 | Provide documentation for all users of the system | 10 | - |  |  | 1,2,3 |  |
| FR7 | Display and log informative messages when exceptions occur |  | - |  |  |  |  |
| FR8 | Deliver user content through a web browser | 4 | - |  |  |  |  |
|  |  |  |  |  |  |  |  |
| System Requirement | | **UR** | **FR** | **SR** | **IR** | **DR** | **DTR** |
| SR |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Interface Requirement | | **UR** | **FR** | **SR** | **IR** | **DR** | **DTR** |
| IR1 | Provide data to RIA | 1,2 | 1,2 |  |  |  |  |
| IR2 | Provide a user intuitive GUI | 6 - 13 | 8 |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Documentation Requirement | | **UR** | **FR** | **SR** | **IR** | **DR** | **DTR** |
| DR1 | Integrate help documentation for User B into the interface |  |  |  |  |  |  |
| DR2 | Define structure and operation of API documentation for User A | 11 | 6 |  |  |  |  |
| DR3 | Document the code base to allow for easier extensibility and maintenance |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Data Requirement | | **UR** | **FR** | **SR** | **IR** | **DR** | **DTR** |
| DTR1 | Data acquired from external sources will be stored in the system database |  |  |  |  |  |  |
| DTR2 | Data must be manually approved for addition to the database |  |  |  |  |  |  |