

Supporting the Creation of Markup for Web Resources

Final Year Honours Dissertation Project

SEAN TURNBULL

H00225489

BSc (Hons) Computer Science

Supervisor: Dr. Alasdair J G Gray

Second Reader: Dr. Arash Eshghi



HERIOT-WATT UNIVERSITY

School of Mathematical and Computer Sciences

Department of Computer Science

April 23, 2019

Declaration

I, Sean Turnbull confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed:

Date:

Acknowledgements

I would like to thank a number of people for their contributions and assistance throughout this project.

- My project supervisor, Dr Alasdair J G Gray, for his invaluable guidance and constructive feedback over the course of the project.
- My second reader, Dr Arash Eshghi, for his constructive feedback on the first deliverable.
- The Bioschemas and life science community, for taking the time to evaluate and provide feedback on my prototype system.

Abstract

Webmasters can increase the visibility of a web page by adding machine-readable data to the page. This allows applications like search engines to understand the content of the web page and increase search visibility which can indirectly impact a web page's search ranking.

Schema.org provides a shared vocabulary for webmasters to use in order to create the machine-readable data markup. However, due to the large number of terms in Schema.org's vocabulary webmasters find difficulty in choosing which terms to use. Schema.org's vocabulary also has limited coverage and does not provide terms in every area for example life sciences, requiring extensions to fill in the missing gaps. Bioschemas proposes new types of information from life sciences to be added to Schema.org's core vocabulary to better support biological types not included in Schema.org. Additionally, it provides a profile layer that enhances the Schema.org model by recommending the most relevant terms, to help webmasters in creating their machine-readable data markup.

Over the course of this project, I have developed a system that allows users to generate markups based on Bioschemas.org profiles to increase the visibility of data from the life science community. Through an initial usability evaluation, members of the life science community found the prototype system to be appealing, easy to use and navigate but fed back that more information needed to be provided to the user. A further usability evaluation was carried out by comparing the final system against a second similar system. Results showed that all test subjects preferred the final system and on average were able to complete the tasks faster compared to the second system.

Contents

1	Introduction	1
1.1	Introduction	1
1.2	Aim & Objectives	2
2	Background	3
2.1	Introduction	3
2.2	Machine-readable Data	3
2.2.1	Machine-readable Data on the Web	4
2.3	Schema.org	9
2.4	Profiles over Schema.org	10
2.4.1	Google Dataset Search	10
2.4.2	Bioschemas.org	11
2.5	Schema.org Markup Support Tools	11
2.5.1	Generation	12
2.5.2	Validation	14
2.6	Summary	16
3	Requirements	17
3.1	Introduction	17
3.2	System Requirements	17
3.2.1	Overview	17
3.2.2	Bioschemas	18
3.2.3	Prototyping	18

3.3	MoSCoW Requirements Analysis	20
3.3.1	Functional Requirements	20
3.3.2	Non-Functional Requirements	21
4	Implementation	22
4.1	Introduction	22
4.2	System Overview	22
4.3	Scripting Tool	24
4.3.1	Choice of Language	24
4.3.2	Declarative Description Processing	25
4.4	Web Application	27
4.4.1	User Interface Design	27
4.4.2	Scripts, Libraries and APIs	31
4.4.3	Hosting the Web Application	34
4.5	Implementation Challenges	35
4.5.1	Issue 1: Data Format Inconsistencies	35
4.5.2	Issue 2: JSON-Editor Documentation	36
4.5.3	Issue 3: Large JSON-Schema	38
4.6	Comparison System	39
4.7	Maintainability	40
4.8	Summary	40
5	Evaluation	41
5.1	Usability Evaluation	41
5.1.1	Prototype Usability Evaluation	41
5.1.2	Final Usability Evaluation	45
5.1.3	Summary of Usability Evaluation	50
5.2	Requirements Evaluation	51
5.2.1	Functional Requirements Evaluation	51

5.2.2	Non-Functional Requirements Evaluation	54
5.2.3	Summary of Requirements Evaluation	55
6	Conclusion	56
6.1	Project Aims	56
6.2	Project Objectives	56
6.3	Future Work	57
A	Prototype Usability Study	61
A.1	Questionnaire	62
A.2	Second Prototype Usability Results	64
B	Final Usability Study	65
B.1	Consent Form	66
B.2	Task Sheets	67
B.3	Questionnaire	69
B.4	Task Times	71
C	Files	74
C.1	Gene YAML	75
C.2	Gene Python Object	84
C.3	BioChemEntity JSON-LD	92
C.4	Gene JSON-Schema	107
C.5	Gene Additional Information JSON	137

Chapter 1

Introduction

1.1 Introduction

The success of machine-readable data on the web can be seen with the rate of adoption from webmasters. From a sample of 3.2 billion web pages in 2017 [32], 38.9% contained machine-readable data. This is up from 30% in 2014 [31] from 2 billion web pages. With the excess number of machine-readable data formats and the markup vocabularies available, webmasters have the difficulty of choosing which format and vocabulary to use.

As each machine-readable data format has their own advantages and disadvantages, webmasters have the difficulty in choosing which format and subsequently which vocabulary best suits their needs. Schema.org aims to make it easier for webmasters by providing a single vocabulary [25] for the following machine-readable formats: RDFa; Microdata; and JSON-LD. Although Schema.org's vocabulary is extensive it relies on extensions to provide terms in areas that are lacking [26].

Section 2.3 describes how properties and types are linked through a single hierarchy. However, due to the extensive number of properties each type can have, webmasters have a difficult time choosing which properties to include. Profiles like Google Dataset Search and Bioschemas aim to help by recommending and suggesting properties that webmasters should include when marking up resources. Bioschemas goes one step further and enhances the Schema.org model by providing an additional layer of constraints (marginality, cardinality and controlled vocabularies)[10]. Additionally, Bioschemas aims to recommend 6 properties in a type to users in order to get the best result possible in discovery searches.

Although there exists many systems to help webmasters generate Schema.org markups, from my research all systems found are limited to a small selection of types. The Bioschemas community are interested in a reconfigurable system which takes a declarative description of a Bioschemas specification and dynamically generates a form to help users create their markup. This is due to Bioschemas rapidly evolving, so ultimately needs a system that is able to keep up with new profiles and types.

1.2 Aim & Objectives

The aim of this project is to develop a system to support users in the creation of Bioschemas markup for their web resources.

The aim will be met through the following objectives:

1. Review existing tools for generating and validating Schema.org markup.
2. Allow users to easily generate Bioschemas markup through an intuitive and aesthetically pleasing user interface.
3. To evaluate the final system's usefulness and validate that the generated data is syntactically correct and compliant with the Bioschemas.org specifications.

Chapter 2

Background

2.1 Introduction

The aim of this project is to develop a system that helps generate markup based on the specifications from Bioschemas.org and subsequently Schema.org. The structured data produced from the system should be in a format that is widely supported and usable on the web. This chapter will explore the topics relevant to the system and its requirements: machine-readable data; Schema.org; profiles over Schema.org; and existing tools for creating and validating Schema.org markups will be reviewed.

2.2 Machine-readable Data

Machine-readable data is structured data that is in a format that can be easily processed by a machine and can be categorised into two types: Human-readable data and Data file formats. Human-readable data is marked up in a way that can be read and understood by humans but also by machines (e.g microformats and RDFa). Data file formats on the other hand is more complex and is intended to only be processed by machines (e.g RDF and JSON-LD).

2.2.1 Machine-readable Data on the Web

Before machine-readable data was introduced to the web, applications that utilised underlying data on web pages had to use custom data extractors to convert the plain HTML into machine-readable data. These extractors were difficult to implement and were prone to breaking when a website changed its layout [11]. By adding machine-readable data to the web it made it easier for applications and webmasters to create and read machine-readable data, enabling search engines to interpret the contents of the web page reliably. This gives users a more rich and interactive experience through features such as Google Rich Snippets¹ and Knowledge Graph². In the following subsections I will discuss the four main formats of machine-readable data on the web:

1. Microformats
2. RDFa
3. HTML Microdata
4. JSON-LD

2.2.1.1 Microformats

Introduced in 2005 [21], Microformats was the first approach to introduce machine-readable data to the web. Using HTML/XHTML standards, microformats adds metadata into the markups tags, this is indicated by the following attributes: `class`, `rel` and `rev`. This allows applications to easily extract and process information on a web page without the need for custom extraction tools (mentioned in Section 2.2.1).

Many varieties of microformats were developed to describe particular types of information such as events, news or products. However, only hCalander [19] and hCard [20] were formally published, with the others remaining as drafts.

¹<https://developers.google.com/search/docs/guides/mark-up-content> (accessed 3/11/18)

²<https://developers.google.com/knowledge-graph/> (accessed 3/11/18)

Although microformats are simple and easy to use, they come with limitations. One limitation is that microformats have issues with scoping [18]. Meaning if one type overlaps another, microformats does not have a way to identify which properties belongs to which type. Another limitation is name spacing [18]. Different microformats may use the same property name, leading to ambiguity of the property definitions. With these limitations and the numerous formats available it can become confusing for the webmasters [25]. Therefore, webmasters have begun to favour new and more advanced machine-readable formats.

The hCard example shown in Listing 2.1 represents a person. The HTML element `link` specifies which type of microformat is used through the attribute `href`, in this case hCard. The `span` element with the attribute `class="fn"` was used to specify the person's full name of "John Doe". The `img` element with attribute `class="photo"` was used to specify a photo of the person, "johndoe.jpg". Finally, the `a` element with attribute `class="url"` was used to specify the homepage for this person, "http://www.johndoe.com" [20].

hCard Example:

```
<link rel="profile" href="http://microformats.org/profile/hcard" />
  <div class="vcard">
    <span class="fn">John Doe</span>
    
    John 's Website:
    <a class="url" href="http://www.johndoe.com">johndoe.com</a>
  </div>
</div>
```

Listing 2.1: A hCard example

2.2.1.2 RDFa

RDFa (Resource Description Framework in attributes) is a human-readable markup of RDF. It allows webmasters to add machine-readable data to XHTML, XML or SVG by adding the ability to write RDF triples as attribute values [28]. There are currently 4 versions of RDFa:

1. RDFa Lite 1.1 (A small subset of RDFa for simple to moderate complexity of structured data) [29]
2. HTML+RDFa 1.1 (Adapting RDFa Core 1.1 and RDFa Lite 1.1 specifications for use in HTML5 and XHTML5) [17]
3. RDFa Core 1.1 (Specification for attributes to express structured data in any markup language) [1]
4. XHTML+RDFa 1.1 (Defined attributes and syntax for embedding semantic markup in XHTML) [2]

The advantage of RDFa compared to microformats, is that RDFa has the ability to declare the name space of the of the property. This eliminates any ambiguity of the property. RDFa also allows the use of more than one vocabulary. This aids webmasters in describing information that may not covered in certain vocabularies.

The RDFa example shown in Listing 2.2 represents a person. The element `p` specifies the vocabulary used with attribute `vocab`, in this case Schema.org. It also specifies the type of property is being described, a Person, through the `typeof` attribute. The `span` element with the attribute `property="name"` was used to specify the person's name of "John Doe". The `img` element with attribute `property="image"` was used to specify a photo of the person, "johndoe.jpg". Finally, the `a` element with attribute `property="url"` was used to specify the URL for this person, "http://www.johndoe.com".

RDFa Example:

```
<p vocab="http://schema.org/" typeof="Person">
  <span property="name">John Doe</span>
  
  John's web site :
  <a property="url" href="http://www.johndoe.com">johndoe.com</a>
</p>
```

Listing 2.2: A RDFa Example

2.2.1.3 HTML Microdata

HTML Microdata is defined as a "HTML extension that defines new HTML attributes to embed simple machine-readable data in HTML documents" [6]. HTML microdata represents machine-readable data as a group of name-value pairs [28] where groups are called items, and each name-value pair is a property.

The main advantage of HTML microdata is that it is simple for webmasters to learn and process [6], but due to its simplicity it has limitations. Limitations of HTML microdata include: only supporting two data types (strings and URLs), not retaining markup structure and not preserving internationalisation-related information except when encoded as microdata [6]. The World Wide Web Consortium (W3C) is a international community, that develops protocols and guidelines for the web to ensure long-term growth [35]. W3C currently recommends webmasters to choose JSON-LD or RDFa over HTML microdata [6] if they need to support the limitations previously discussed.

The Microdata example shown in Listing 2.3 represents a Person. The `div` element specifies the property type and vocabulary through the attribute `itemtype`, in this example the vocabulary is Schema.org and is of type Person. The `span` element with the attribute `itemprop="name"` was used to specify the person's name of "John Doe". The `img` element with attribute `itemprop="image"` was used to specify a photo of the person, "john-doe.jpg". Finally, the `a` element with attribute `itemprop="url"` was used to specify the URL for this person, "http://www.johndoe.com".

Microdata Example:

```
<div itemscope="itemscope" itemtype="http://schema.org/Person">
  <span itemprop="name">John Doe</span>
  
  John's web site:
  <a href="http://www.johndoe.com" itemprop="url">johndoe.com</a>
</div>
```

Listing 2.3: A Microdata Example

2.2.1.4 JSON-LD

The latest machine-readable data format introduced for the web was JSON-LD (JavaScript Object Notation for Linked Data). Unlike other web formats JSON-LD is "described as JavaScript code rather than markup elements and attributes" [28], meaning that JSON-LD is completely detached from the HTML code.

Due to JSON-LD being in a human-readable format, it is easy for webmasters to create and read the data without the aid of any tools. With JSON-LD being separate from the markup, it can be easily reused and interchanged without having to edit the markup. However, the disadvantage to this is that if the webmaster changes the content of web page they may forget to update the JSON-LD capturing this change.

The JSON-LD example shown in Listing 2.4 represents a person. The key value pair `@context` specifies the vocabulary being used, in this case it is Schema.org. The key value pair `@type` is used to specify the property type from the specified vocabulary, in this case it is Person. The key value pair `name` specifies the name of the person. The key value pair `image` was used to specify a photo of the person. Finally, the key value pair `url` specifies the URL of the person.

JSON-LD Example:

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "Person",
  "name": "John Doe",
  "image": "johndoe.jpg",
  "url": "http://www.johndoe.com"
}
</script>
```

Listing 2.4: A JSON-LD Example

2.2.1.5 Adoption Statistics

The Web Data Commons project [30] extracts machine-readable structured data from the Common Crawl [7], the largest copy of the web publicly available. The project provides statistics on the formats of structured data used on the web from 2009 onwards.

Format	Domains 2017	Domains 2016
HTML Microdata	3,743,822	2,537,539
Microformats (hCard)	2,758,884	1,668,039
JSON-LD	2,685,738	2,116,755
RDFa	1,209,430	938,830

Table 2.1: Top 4 used machine-readable data formats from 2016/2017 Common Crawl [32, 33]

Table 2.1 shows the significant growth of machine-readable data on the web over the course of a year between 2016 and 2017. This growth is due to more companies supporting and utilising machine-readable data on the web. For example, during this time Google implemented Rich Cards³ and both Google⁴ and Bing⁵ began to support Schema.orgs fact checking type ClaimReview⁶. It is also due to availability of tools making it easier for webmasters to generate and validate machine-readable data (mentioned in Section 2.5). Table 2.1 also shows us the rise of the newest format JSON-LD. The use of JSON-LD is likely to surpass all other formats. This is due to large companies like Bing⁷ and Google beginning to support the format as well as Google now recommending it over the other formats [9].

³<https://webmasters.googleblog.com/2016/05/introducing-rich-cards.html> (accessed 21/11/2018)

⁴<https://www.seroundtable.com/google-news-fact-check-22884.html> (accessed 21/11/2018)

⁵<https://www.seroundtable.com/bing-claimreview-fact-checking-24209.html> (accessed 21/11/2018)

⁶<https://toolbox.google.com/datasetsearch> (accessed 21/11/2018)

⁷<https://blogs.bing.com/webmaster/august-2018/Introducing-JSON-LD-Support-in-Bing-Webmaster-Tools> (accessed 04/11/2018)

2.3 Schema.org

Schema.org aims to improve the web by creating a shared collection of machine-readable data markup schema that is supported by the major search engines providers. "It was created in 2011 by the major search engines Bing, Google and Yahoo (later joined by Yandex)" [12] to combat the rising number of markup schemas causing confusion and type duplication with webmasters. By creating a shared markup vocabulary it makes it easier for web masters to create and use machine-readable data on the web.

Schema.org also tried to solve which format of machine-readable data to recommend to webmasters. In the end due to each format having their own pros and cons they decided that multiple formats would be the best approach [11]. Schema.org currently recommends, supports and provides examples for the following formats of structured data: RDFa, HTML Microdata and JSON-LD.

When launched, Schema.org provided 297 types and 187 properties [11] which has now grown to 598 types, 862 properties and 144 enumeration values in version 3.4 [26]. Each type has a set of properties where a property can be a data type or another type, creating a hierarchical structure. For example, Person⁸ is a type for representing "a person (alive, dead, undead or fictional)" [27]. Person has 54 properties unique to it. For example, "givenName for their firstname; familyName for their last name; and birthDate for their date of birth" [27]. Due to the singular hierarchy structure, types can also contain properties from other types. For example, the type Person contains 12 properties from the type Thing⁹ like "alternateName for an alias; description for a description of the item; and image for an image of the item" [27]. This leads to less property duplication and confusion for web masters.

Although the amount of types provided by Schema.org is extensive, they rely on extensions to provide vocabularies on topics they do not cover. There are two types of extensions: Hosted and External. Hosted extensions are "managed and published as part of the Schema.org project, with their design often led by one or more dedicated community groups" [26] e.g Auto¹⁰ and Bib¹¹. External extensions are not officially supported by Schema.org and are typically managed by other organisations with their own processes and collaboration mechanisms [26] e.g GS1¹².

⁸<https://schema.org/Person> (accessed 03/11/2018)

⁹<https://schema.org/Thing> (accessed 03/11/2018)

¹⁰<https://auto.schema.org> (accessed 01/11/2018)

¹¹<https://bib.schema.org> (accessed 03/11/2018)

¹²<https://gs1.org/voc> (accessed 01/11/2018)

2.4 Profiles over Schema.org

Unlike an extension of Schema.org, profiles focus on recommendations and suggestions for properties in a type. This allows web masters to better create their markup without being overwhelmed with the amount of properties and types provided by Schema.org. Two organisations that provide profile definitions are Google Dataset Search and Bioschemas.org.

2.4.1 Google Dataset Search

Google's Dataset search¹³ is a search engine that allows users to find a dataset regardless of whether it is located on a publishers site, digital library or on a personal web page [24]. In order for their search engine (and others) to find the datasets in these mediums they created a profile to better aid web masters in creating their markups.

The profile adds a layer over the existing Schema.org model that categorises properties of a type into two categories: required and recommended. This helps the web masters pick which properties to include in their markup. For example, in Google's Dataset¹⁴ they require 2 properties and recommend 10. While Schema.org's Dataset¹⁵ only provide a list of 103 properties with no indication to web masters which ones to use. With this information they can analyse where the dataset has come from and find which publications are describing or discussing that particular dataset [5].

Additionally, the profile suggests best practices to follow when creating the markup. For example, Google suggests to use the `isBasedOn` property if the dataset is a republished dataset that has been significantly changed, or if the dataset derives from or aggregates several originals [8]. This allows Google to identify where datasets have originated from and potentially find overlapping studies effectively making a larger dataset. With a larger dataset the effect of any outliers and bad data could be reduced or even negated [23].

¹³<https://toolbox.google.com/datasetsearch> (accessed 20/11/2018)

¹⁴<https://developers.google.com/search/docs/data-types/dataset> (accessed 20/11/2018)

¹⁵<https://schema.org/Dataset> (accessed 20/11/2018)

2.4.2 Bioschemas.org

In the life science community there is demand to add new types and properties to Schema.org. This is due to generic types like Dataset and Event from Schema.org not being able to properly represent the different types of biological resources available e.g information on genes and proteins. Bioschemas identifies new properties and types from life sciences through the help of life science community and proposes their adoption by Schema.org [4].

Bioschemas not only extends Schema.org with new types and properties for biological entities but also adds a profile layer over the existing Schema.org model [10] providing additional constraints. The additional constraints are:

1. The marginality of the properties agreed by the community which are: minimum, recommended or optional.
2. The cardinality of the property, i.e whether the properties occurs once or many times.
3. Associated controlled vocabulary terms drawn from existing ontologies

Finally, Bioschemas aims to require just 6 properties in a type based on their ability to support indexing and snippet generation and provide the most information to the consumer [10]. The number of properties chosen to provide the most information is the average number of properties contained in a Schema.org markup [12]. Bioschemas believes this number of properties is the ideal number to get the most information without asking the webmaster for information not necessarily required.

2.5 Schema.org Markup Support Tools

While researching the topics previously discussed, I took a look at existing tools that have similar functionality to the system I will be developing. As my project involves generating markup based on Bioschemas.org profiles, I examined tools that generate Schema.org markup. Due to my project generating data, I also took a look at validation systems to make sure the data produced by my system is correct according to syntax and conformance with a profile. A brief description of each tool is given below.

2.5.1 Generation

Technical SEO

Technical SEO's Schema Markup Generator¹⁶ shown in Figure 2.1 allows users to create Schema.org markup. By selecting a Schema.org markup from a list of 10, it then displays a form with the required field that the user will then fill out in order to create said markup. It then generates JSON-LD and Microdata for the user to then use in their website. Additionally it provides quick access to Google's structured data testing tool to validate the generated data.

</> Schema Markup Generator (JSON-LD)

Which [Schema.org](#) markup would you like to create?

Article

Use this Schema.org structured data generator to create JSON-LD & Microdata markups, including all of the required item properties and more. Click on "Validate" to test your newly created markup with Google's [Structured Data Testing Tool](#).

Article Markup: NewsArticle, BlogPosting

JSON-LD Microdata [G Validate](#)

NewsArticle

Accelerated Mobile Page (AMP)? ☒

Headline (≤ 110 characters)

Image URL

Image width (≥ 696px) px

Image height px

Schema.org's references:

Google's documentation:

- Article
- NewsArticle
- BlogPosting

- Articles

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "NewsArticle",
  "headline": "",
  "image": {
    "@type": "ImageObject",
    "url": "",
    "width": ,
    "height":
  }
}
</script>
```

Comments

+

Figure 2.1: Technical SEO Schema Markup Generator

¹⁶<https://technicalseo.com/seo-tools/schema-markup-generator/>(accessed 07/11/2018)

Kaizen

Kaizen¹⁷ shown in Figure 2.2 is a Google sheets¹⁸ template to help create markups from Schema.org. Kaizen supports the markup generation of 4 Schema.org types: organisation, video object, article and product rating. Due to its nature of being a spreadsheet it is more complex and not as user friendly as Technical SEOs. To generate the markup the user selects the sheet of the type they would like to generate and fill in the appropriate cells. It also provides an example markup to help the users fill out the sheet.

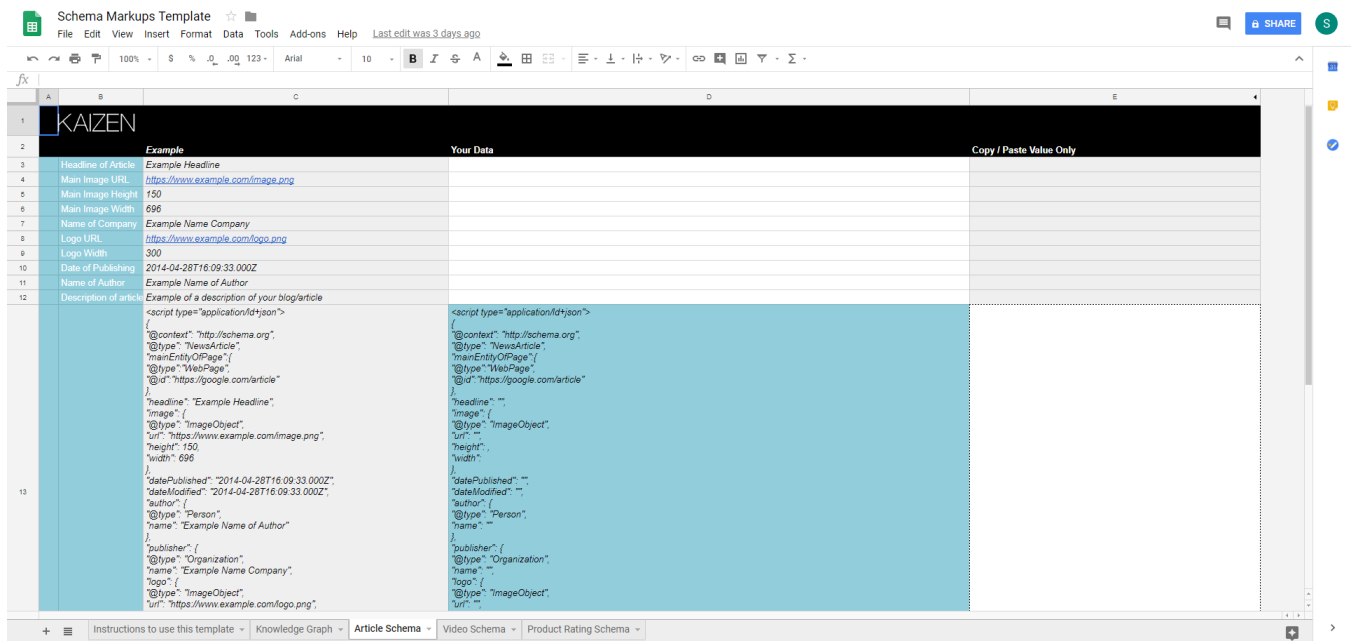


Figure 2.2: Kaizen Template

Customisability

Although these systems are extremely useful in helping the user generate their Schema.org markup, both have the limitation of the number of Schema.org markup types they support. Technical SEO's markup generator is a private system, so in order to add new types, a user would have to contact Technical SEO in hopes they would update the system to support the types they would like. On the other hand, Kaizen's system is completely open source due to its nature of being a spreadsheet template. This allows users to use the provided markup templates as an example and customise them to create a new type or add in additional properties.

¹⁷<https://www.kaizen.co.uk/blog/seo/the-ultimate-template-for-structured-data-markup/> (accessed 03/11/2018)

¹⁸<https://www.google.co.uk/sheets/about/> (accessed 03/11/2018)

2.5.2 Validation

Structured Data Testing Tool

Google's structured data testing tool¹⁹ shown in Figure 2.3 allows users to test their structured data. The user can supply the structured data in one of two ways. The first is through a code snippet. The second is through a URL, by giving a URL it also gives an indication if the structured data on their web page can be accessed and used by applications. Once the structured data is supplied it then validates the structured data is syntactically correct as well as testing it against the Schema.org specification.

https://www.proteinatlas.org/ NEW TEST

DataCatalog

All (1)

DataCatalog

0 ERRORS 1 WARNING

@type	DataCatalog
name	Human Protein Atlas
license	http://www.proteinatlas.org/about/licence
thumbnailUrl	http://www.proteinatlas.org/images_static/logo.gif

logo

The property logo is not recognized by Google for an object of type DataCatalog)

The Human Protein Atlas is a Swedish-based program initiated in 2003 with the aim to map all the human proteins in cells, tissues and organs using integration of various omics technologies, including antibody-based imaging, mass spectrometry-based proteomics, transcriptomics and systems biology. All the data in the knowledge resource is open access to allow scientists both in academia and industry to freely access the data for exploration of the human proteome. The Human Protein Atlas consists of three separate parts, each focusing on a particular aspect of the genome-wide analysis of the human proteins; the Tissue Atlas showing the distribution of the proteins across all major tissues and organs in the human body, the Cell Atlas showing the subcellular localization of proteins in single cells, and finally the Pathology Atlas showing the impact of protein levels for survival of patients with cancer. The Human Protein Atlas program has already contributed to several thousands of publications in the field of human biology and disease and it is selected by the organization ELIXIR (www.elixir-europe.org) as a European core resource due to its fundamental importance for a wider life science community. The Human Protein Atlas consortium is funded by the Knut and Alice Wallenberg

description

Figure 2.3: Google Structured Data Testing Tool

¹⁹<https://search.google.com/structured-data/testing-tool> (accessed 03/11/2018)

Validata

Validata²⁰ shown in Figure 2.4 is a tool for "validating a RDF document against a set of constraints"[13]. Constraints are expressed as a Shape Expression (ShEx) schema [3]. The user can select a schema that they would like their data to be validated against, currently supported schemas are Open PHACTS Dataset Description, HCLS Community Profile, DCAT or custom schema. The user then supplies either through a file or input box and then provides the results of the validation. Although not yet supported, this tool will eventually allow users to validate their JSON-LD against a description, providing another method for users to validate their Schema.org / Bioschemas markups.

1 Select Schema

2 Input Data

3 Configure Options

4 Validation Results

Load Demo Data

▶ DCAT Example

Select Schema

Select which schema your data should be validated against:

DCAT

Show Source

Description

W3C Data Catalog Vocabulary <http://www.w3.org/TR/2014/REC-vocab-dcat-20140116/> DCAT is an RDF vocabulary designed to facilitate interoperability between data catalogs published on the Web. This document defines the schema and provides examples for its use. By using DCAT to describe datasets in data catalogs, publishers increase discoverability and enable applications easily to consume metadata from multiple catalogs. It further enables decentralized publishing of catalogs and facilitates federated dataset search across sites. Aggregated DCAT metadata can serve as a manifest file to facilitate digital preservation. Note that the example will generate a single error due to the use of the `rdfs:label` property. Validating without the closed world assumption will enable this to pass.

Creation Date

14/09/2015 13:55

Input Data (Turtle, TriG, N-Triples or N-Quads)

Upload Data File

Choose file No file chosen

Select a data file from your computer.

Directly Input Data

1 ### Example taken from the DCAT specification document.

2 ### Example will validate with a single error, that of the use of `rdfs:label`.

3 ### The use of `rdfs:label` is not specified in the DCAT document.

4 ### Validating without the closed world assumption set will enable the example to validate successfully.

5

6 # `http://example.com/dcat#dataset-001` validates as `<Dataset>`

7 # `http://example.com/dcat#catalog` validates as `<Catalog>`

8 # `http://example.com/dcat#transparency-office` validates as `<Agent>`

9 # `http://example.com/dcat#dataset-001.csv` validates as `<Distribution>`

10

11 PREFIX : <`http://example.com/dcat#`>

12 PREFIX dcat: <`http://www.w3.org/ns/dcat#`>

Status Summary

Schema

✓

Data

✓

Validation

✓

Figure 2.4: Validata: RDF Validator

²⁰<http://hw-swel.github.io/Validata/>(accessed 03/11/2018)

2.6 Summary

In conclusion, this chapter has examined Machine-readable data, more specifically the number of formats that are currently available for use on the web. Shown is the rise in use of machine-readable data on the web from webmasters, as large companies like Google and Bing have begun to provide tools and services to warrant their use.

Schema.org was covered, describing their solution to the vast amount of vocabularies for machine-readable data on the web by provide a single shared markup vocabulary. Due to the large number of properties and types available, webmasters struggle to choose decide which properties and types to best describe their web pages. Bioschemas.org Profiles and Google Dataset Search were discussed as a solution to this issue, as they research and recommend the most useful types and properties to include.

Finally, existing tools that aided in the generation and validation of Schema.org markup's were given and discussed. This is due to the proposed system having similar features to current Schema.org markup generators, and having a system to validate the data being produced will be vital.

Chapter 3

Requirements

3.1 Introduction

This chapter will explore the requirements of the final system. In this project I am being assisted by Dr. Alasdair J G Gray, an Associate Professor in Computer Science at Heriot-Watt University. Dr. Gray leads the Bioschemas community, meaning he can provide detailed requirements of what the system should be able to provide, additionally providing feedback as a user and encouraging the community to try the tool.

In this chapter, I will give an overview of the systems desired functionality, discuss the requirements gathered from Bioschemas as well as my initial prototype systems and how they aided in establishing the final system requirements. Finally, a MoSCoW analysis of the system requirements will be completed.

3.2 System Requirements

3.2.1 Overview

Once the system is complete it should allow the user to select a Bioschemas profile that they would like to generate a markup for. A form should then appear allowing the user to enter the data required to create the markup for the selected profile. The system then provides the user with all the information they need to complete the form without having to go to any external sources or read any documentation. Once complete, the system then generates the markup with the provided data. The user will then be able to validate the generated markup is correct, both syntactically and against the profile.

3.2.2 Bioschemas

As mentioned in Section 2.4.2, Bioschemas adds not only new biological types to Schema.org but also a profile layer over the Schema.org model with additional constraints. To support Bioschemas profiles, the system must be able to support these additional constraints: cardinality (one or many items for a specified property), marginality (Minimum, Recommended and Optional properties) and use of controlled vocabularies.

Through discussions with Dr. Gray, I was able to establish that the system should use a declarative description of the Bioschemas profiles in order to generate the form. The form should also be ordered in a way that makes most sense to the user. i.e Minimum then Recommended then Optional properties. Additionally, the generated markup should be in the format of JSON-LD (See Section 2.2.1.4).

Finally, the system should display all the information required for the user to fill out the form. Where the information is available, each property should provide a description, examples and controlled vocabularies, which are provided by Bioschemas and Schema.org.

3.2.3 Prototyping

To gain a better understanding of the system and its requirements I developed a prototype system. The goal of the prototype was to create a form that allowed users to generate a markup for the Dataset (v0.2)¹ Bioschemas profile. Through developing the prototype one functional requirement was discovered and another's importance was highlighted.

During development I discovered that in order to generate markup based on the Bioschemas Dataset (v0.2) profile, the form must be able to handle recursive property types. This is due to Schema.org types having the ability to contain themselves as an expected type for a property. For example, the Schema.org type PropertyValue² has a property valueReference, where the expected type for this property can either be Enumeration, QualitativeValue, QuantitativeValue, StructuredValue or itself a PropertyValue. This requirement was then added to the list of functional requirements as FR 7 in Table 3.1. The requirement's prioritisation was set to should as unfortunately at this stage I was not able to produce this functionality, later discussed in Section 4.4.2.1.

¹<http://bioschemas.org/specifications/Dataset/> (accessed 03/11/2018)

²<https://schema.org/PropertyValue> (accessed 03/11/2018)

The requirement highlighted during prototype development was FR 5, the system must be able to handle cardinality and marginality, shown in Table 3.1. The cardinality portion of this requirement previously mentioned in Section 3.2.2, was not possible during the first iteration of the prototype due to limited functionality with the library used to display the form, this limitation will be later discussed in Section 4.4.2.1.

Once the prototype system was complete, shown in Figure 3.1, a initial prototype usability evaluation was carried out to tailor the system requirements towards to the target user-group (members of the Bioschemas and life science communities). The feedback showed us that the functional requirements of the system achieved the desired functionality although more information needed to be provided thus FR 10 (Descriptions and Examples) and 11 Controlled Vocabulary) were added. Additionally, one test subject recommended to include different formats of markup and so FR 12 (Microdata and RDFa) was included. A more detailed discussion of the prototype usability evaluation and the results can be seen in Section 5.1.1.

Bioschemas Markup Generator

The screenshot displays the 'Bioschemas Markup Generator' interface. It features a blue header bar with the title. Below the header, there are two main sections: 'Input' and 'Generated Form'. The 'Generated Form' section contains a 'Dataset Profile' header and a description: 'A form to help describe datasets in the life-sciences using Schema.org-like annotation. Marginality: Minimum + Recommended , No recursive definitions.' The form includes several input fields with labels and descriptions: 'Name', 'Description', 'URL', 'Identifiers', 'Keywords', 'Citation', 'Creator', 'Distribution', 'Included In Data Catalog', 'License', 'Measurement Technique', 'Variable Measured', and 'Version'. Each field has a small 'i' icon and a description. The 'Identifiers', 'Citation', 'Creator', 'Included In Data Catalog', 'Measurement Technique', and 'Variable Measured' fields have an 'Add item' button. The 'Version' field has a dropdown arrow. At the bottom of the form, there are two buttons: 'Generate JSON-LD' and 'Validate Form Data'. Below the form, there is a blue bar labeled 'Form Data'.

Figure 3.1: Bioschemas Markup Generate Prototype Second Iteration

3.3 MoSCoW Requirements Analysis

From the requirements in Section 3.2.2 above along with any other requirements gathered, a MoSCoW analysis was performed on the requirements detailing the priority of each. The requirements have been split into two categories: Functional Requirements Table (3.1) and Non-Functional Requirements Table (3.2).

3.3.1 Functional Requirements

Functional Requirements			
ID	Priority	Requirement	Reasoning
FR 1	Must	The system must allow users to enter data into the form.	This is core functionality of the system.
FR 2	Must	The system must generate JSON-LD compliant data.	This is core functionality of the system.
FR 3	Must	The system must allow users to select the Bioschemas profile.	This allows users to be able to select the Bioschemas profile that they would like to generate the markup for.
FR 4	Must	The system must be able to handle Schema.org data types ³ .	This is necessary to be able to support the user in entering the data types that Bioschemas allows.
FR 5	Must	The system must be able to handle cardinality and marginality.	This is core functionality of the system to support BioSchemas profiles.
FR 6	Should	The system should generate the forms based on a declarative specification.	This allows the system to be more dynamic and not be rely on static form definitions. Supporting the evolution of Bioschemas.
FR 7	Should	The system should be able to handle recursive properties.	This allows recursion that may occur in Schema.org properties.
FR 8	Should	The system should be able to prioritise form inputs.	The form should be ordered in a way that is intuitive for the user.
FR 9	Should	The system should be able validate the generated markup against the Bioschemas profile .	The user should be certain that the generated markup is correct.
FR 10	Should	The system should display a description of each property, as well as examples.	To allow users to get more information without going to external sources.

³<https://schema.org/DataType> (accessed 03/11/2018)

FR 11	Should	The system should display the controlled vocabulary of each property.	To allow users to see the controlled vocabulary to use without going to external sources.
FR 12	Could	The system could generate Microdata and RDFa along side JSON-LD.	Allow users to have a choice of format generated.
FR 13	Could	The system could allow the user to download the data generated.	Allow users to easily save their results without having to use external software.
FR 14	Wont	The system wont be able to dynamically update declarative specifications from an online source.	To stop the systems functionality from going down if the declarative specification is incorrect.

Table 3.1: Table of Functional Requirements

3.3.2 Non-Functional Requirements

Non-Functional Requirements			
ID	Priority	Requirement	Reasoning
NFR 1	Must	The system must be easy to use through a simple user interface.	A systems interface should be intuitive to use.
NFR 2	Must	The system must be available 24/7	The system should be available at all times to support potential users across the globe.
NFR 3	Should	The system should be accessible through the top internet browsers: Google Chrome, Mozilla Firefox and Apple Safari.	To allow the most users access to the system.
NFR 4	Could	The system could have two interfaces. One for standard users. Another for administrative users.	To allow more advanced users greater functionality to generate more complex markups.

Table 3.2: Table of Non-Functional Requirements

Chapter 4

Implementation

4.1 Introduction

This chapter will explore the final implementation of the proposed system by giving an overview of the final system as well as discussing the languages and libraries used. It will also highlight any challenges that were encountered during the development process. Furthermore, it will discuss the comparison system used in the final usability evaluation and the maintainability of each of the systems.

4.2 System Overview

As shown in Figure 4.1, the final system was separated into two individual systems: a scripting tool and a web application. The scripting tool generates the required files for the web application based on information from Bioschemas and Schema.org. The web application then uses these files to display a form and additional information to allow users to markup Bioschemas profiles.

Separating the file generation from the web application allowed the load times to be kept as low as possible. By generating the files separately and having the web application fetch the generated files, it removes the time needed to generate the files each time the web application is loaded.

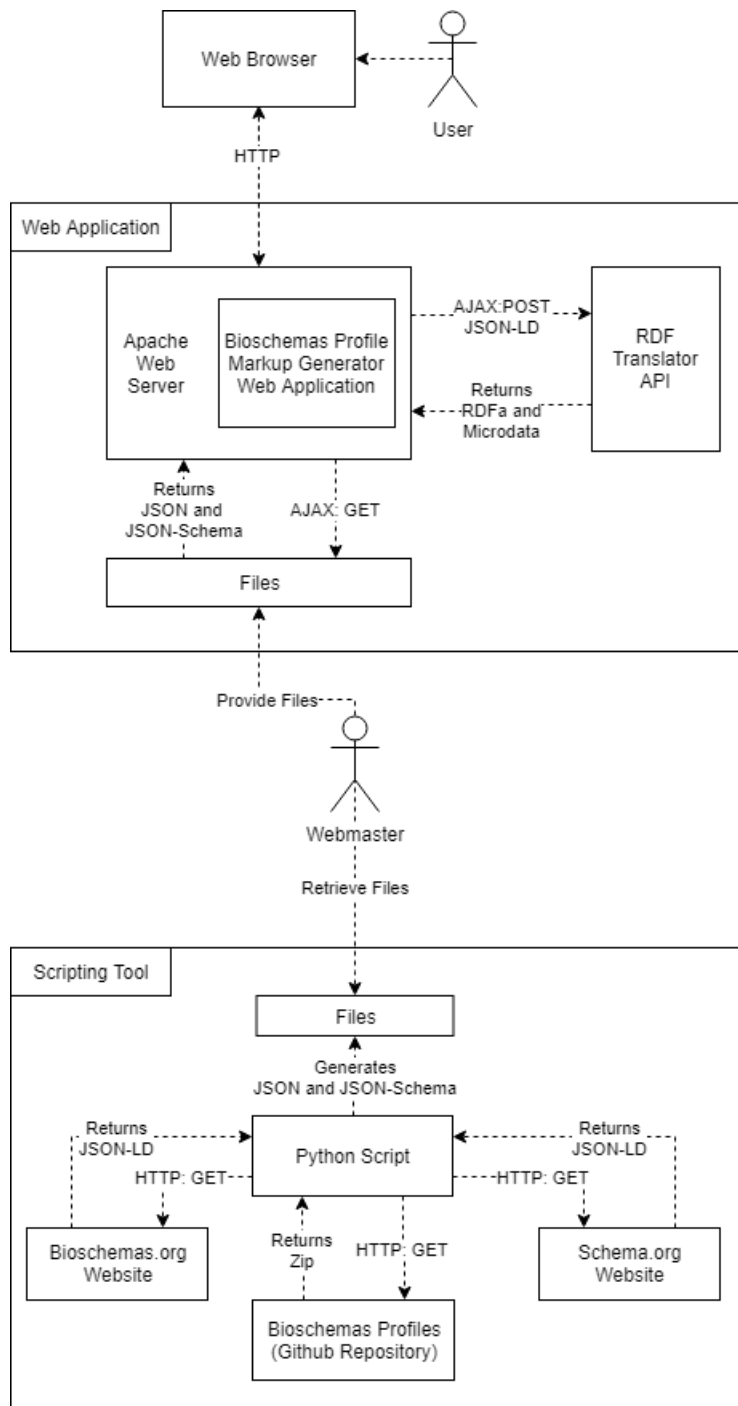


Figure 4.1: Component Diagram

4.3 Scripting Tool

4.3.1 Choice of Language

After developing the initial prototype to help gather requirements and ensuring I was satisfied with the approach to utilise JSON-Schema to generate the form. The next step was to begin developing a tool that takes the declarative descriptions of the Bioschemas profiles and generates the required files (JSON and JSON-Schema).

To implement this tool, I had two different options available. I could either extend the Bioschemas GoWeb tool which generates the declarative description of the Bioschemas Profiles or create a separate tool in Python that takes the output of GoWeb and further processes it.

4.3.1.1 Bioschemas GoWeb

Go¹, also known as GoLang, is an open source language developed by Google that is similarly modelled after C [16] and is also considered to be lightly object-orientated [36]. While I do not have any experience with this language specifically, I have a lot of experience working with object-orientated languages but have very limited experience with C.

Bioschemas GoWeb², developed in Go, is a tool that generates a declarative description of the Bioschemas profiles. The declarative descriptions that it generates is a human friendly but machine readable format called YAML³. To generate the YAML, the tool uses CSVs which contains information about the Bioschemas profiles. To aid the webmasters in creating the CSVs, a Google Sheets template and detailed instructions are provided.

This is a valid option for our scripting tool as we could extend Bioschemas GoWeb by taking the YAML it produces and then generating the files needed directly from it. Furthermore, by generating the files directly from the YAML, it mitigates the need to retrieve the generated YAML from the Bioschemas Github⁴.

¹<https://golang.org> (accessed 28/02/2019)

²<https://github.com/BioSchemas/bioschemas-goweb> (accessed 28/02/2019)

³<https://yaml.org/> (accessed 28/02/2019)

⁴<https://github.com/BioSchemas/bioschemas.github.io> (accessed 28/02/2019)

4.3.1.2 Python Tool

Python⁵ is a object-orientated language, high-level programming language that is often used for rapid prototyping. Through university coursework and personal projects I have a lot of experience with this language as well as other object-orientated languages.

This is a valid option as the YAML generated by the GoWeb tool for the Bioschemas profiles is available through the Bioschemas Github. In this case we would be able to retrieve the YAML directly from the Github and then further process the YAML into the desired files, instead of processing the CSV's directly to generate the YAML first like in the GoWeb tool.

4.3.1.3 Selection

While the Bioschemas GoWeb tool is a good starting point in developing the scripting tool, I can not neglect my experience and knowledge with Python and object-orientated programming. As time management is critical in this project with Python being utilised in rapid prototype development and having a more extensive selection of libraries, the tool will be written in Python.

4.3.2 Declarative Description Processing

The first stage of generating the required files for the web application was retrieving information about and surrounding the Bioschemas profiles and converting it into a usable format. As the information is collected from multiple sources (Shown in Figure 4.1), two methods of retrieval and conversion were used.

For the first method, the information about the Bioschemas profiles were located on the Bioschemas Github and were in the YAML format. Using Python library Requests⁶ the YAML was retrieved using HTTP GET requests. The Python library PyYAML⁷ was then used to convert the YAML into a Python object to be further processed. Shown in Appendix C.1 is an example of the YAML retrieved for the Bioschemas profile Gene (v0.4)⁸ and the resulting Python object displayed as JSON is shown in Appendix C.2.

⁵<https://python.org> (accessed 28/02/2019)

⁶<http://docs.python-requests.org/en/master/> (accessed 28/02/2019)

⁷<https://pyyaml.org/> (accessed 28/02/2019)

⁸<https://bioschemas.org/specifications/Gene/> (accessed 28/02/2019)

The second method was used to fetch additional information about Bioschemas and Schema.org types. Similar to the first method, this was achieved using the Request Python library to perform HTTP GET requests to retrieve the information in the format of JSON-LD. As JSON-LD only adds specific key pair values in comparison to JSON, I was able to use the Python library JSON⁹ to convert the JSON-LD into a Python object to be further processed. Shown in Appendix C.3 is the JSON-LD for the Bioschemas type BioChemEntity¹⁰.

Once the methods of retrieving the information was figured out, the next stage of the process was to generate the JSON-Schema¹¹ used to display the form on the web application. To create the JSON-Schema, I was able to create a Python dictionary of key value pairs and then using the JSON library I was able to output the dictionary as JSON-Schema. During this stage, I encountered three major issues that are discussed in Section 4.5. Through using the iterative development process and help from my supervisor Dr. Alasdair Gray, I was able to work through these issues to create the JSON-Schema describing the Bioschemas profiles. Shown in Appendix C.4 is an example of the JSON-Schema generated for the Bioschemas Gene (v0.4)¹² profile.

The final stage of the process was to generate JSON that is used to display additional information (examples and controlled vocabulary) along side the form on the web application. This used the same method as creating the JSON-Schema, by creating a Python dictionary of key value pairs and then utilising the Python library JSON to output the dictionary as JSON. Shown in Appendix C.5 is example of the generated JSON for the Bioschemas Gene (v0.4) profile.

To summarise, the scripting tool generates two files for each Bioschemas profile: A JSON-Schema file that is used by the JavaScript library JSON-Editor (See Section 4.4.2.1) to display a form that the user can use to markup Bioschemas profiles and a JSON file that contains the controlled vocabularies and examples of the profiles properties to be displayed along side the form (See Section 4.4.2.2).

⁹<https://docs.python.org/2/library/json.html> (accessed 28/02/2019)

¹⁰<http://bio.sdo-bioschemas-227516.appspot.com/BioChemEntity> (accessed 05/04/2019)

¹¹<https://json-schema.org/> (accessed 05/04/2019)

¹²<http://bioschemas.org/specifications/Gene/> (accessed 02/03/2019)

4.4 Web Application

The web application consists of an Apache Web Server¹³ (Version 2.2.15), containing the front-end user interface as well as JavaScript scripts and libraries for client-side processing. The web application retrieves the generated files from the scripting tool (See Section 4.3.2) that are stored on the Apache Web Server and using a multitude of libraries (See Section 4.4.2) provides the desired functionality of the proposed system.

4.4.1 User Interface Design

The user interface provides an easy and intuitive way for users to select, provide information and generate the markup for a Bioschemas profile. It was developed using HTML5, CSS3 and Bootstrap 4. I used Bootstrap¹⁴ as a front-end web framework to handle the layout, styling and fonts of the HTML elements that are displayed to the user. Bootstrap also supports the latest, stable releases of all major browsers and platforms, which forms part of the system requirements (Non-Functional Requirement NFR3: The system should be accessible through the top internet browsers: Google Chrome, Mozilla Firefox and Apple Safari.).

Select Profile

The Select Profile section is designed to allow users to select which Bioschemas profile they would like to generate a markup for through a drop-down list, shown in Figure 4.2. Unfortunately, as the profiles are added to the drop-down list asynchronously, the resulting drop-down list is not alphabetically ordered, possibly adding time for a user to find their desired profile. The "Generate Form" button then displays the form for the Bioschemas profile they selected.

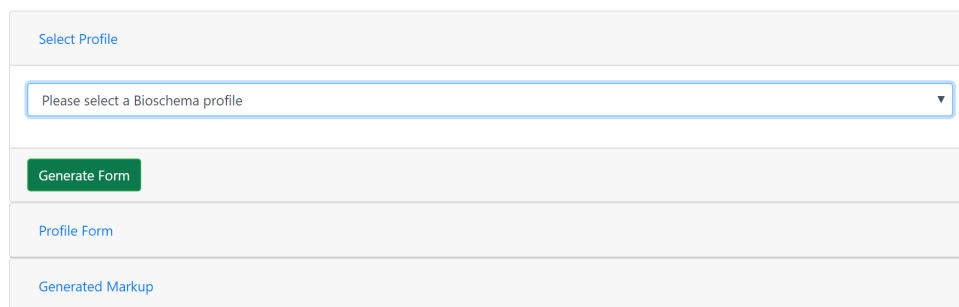
The image shows a web form titled "Select Profile" in a light blue header. Below the header is a white input field with a blue border and a dropdown arrow on the right, containing the text "Please select a Bioschema profile". Underneath the input field is a green button with white text that says "Generate Form". Below the button is another light blue header section with the text "Profile Form". At the bottom of the form is a light blue header section with the text "Generated Markup".

Figure 4.2: Select Profile

¹³<https://httpd.apache.org/> (accessed 12/03/2019)

¹⁴<https://getbootstrap.com/> (accessed 12/03/2019)

Profile Form

The Profile Form section is designed to allow the user to provide information to markup a Bioschemas profile they previously selected, shown in Figure 4.3. The page is designed as two parts: the left hand side displaying the form for users to provide the information with the right hand side providing additional information like examples and controlled vocabularies.

Profile Form

Gene (v0.4)

Additional Properties

This Gene profile specification presents the markup for describing a Gene.

identifier (Minimum) Property/Value ▾

Additional Properties

A property-value pair, e.g. representing a feature of a product or place. Use the 'name' property for the name of the property. If there is an additional human-readable version of the value, put that into the 'description' property.

Always use specific schema.org properties when a) they exist and b) you can populate them. Using Property/Value as a substitute will typically not trigger the same effect as using the original, specific property.

@id

name (Minimum) Text ▾

Schema.org: The name of the item.

Property Tips

Minimum

Recommended

Description

Example: [Show](#)

Encodes

Example: [Show](#)

Controlled Vocabulary: SIO

Hasrepresentation

Example: [Show](#)

Image

Example: [Show](#)

Isparstofbiochemistry

Example: [Show](#)

Controlled Vocabulary: For subcellular locations branch from GO, please use [Gene Ontology (GO)] (<http://www.geneontology.org>)

Url

Example: [Show](#)

Optional

Generate Markup

Figure 4.3: Bioschemas Profile Gene Form

The form is displayed using the JSON-Editor JavaScript library, which will be discussed in Section 4.4.2.1 and is styled using Bootstrap 4. By default it displays the Minimum required properties of the Bioschemas Profile. To add the Recommended and Optional properties to the form, select the "Additional Properties" button, shown in Figure 4.4 and then select the check box next to the desired properties.

Gene (v0.4)

This Gene profile specifies the following properties:

- ☒ identifier (Minimum)
- ☒ name (Minimum)
- ☐ description (Recommended)
- ☒ encodes (Recommended)

Additional Properties

A property-value pair, name of the property, 'description' property.

Property name... **add**

encodes (Recommended)

Bioschemas.org: For genes, this property is used to express in a generic way gene products encoded by this gene. Two more specific properties SIO:010082 (is translated into) and SIO:010080 (is transcribed into) should be used for (protein) translation and (RNA) transcription respectively. Note: bioschemas:encodes skos:closeMatch SIO:010078 Schema.org: This property is used to link to gene products encoded (even indirectly) from this gene such as RNA or proteins.

Figure 4.4: Bioschemas Profile Gene Additional Properties

For Properties that have a cardinality of many, items can be added through the "Add item" button shown in Figure 4.5. Once an item has been added, the type of item can then be selected through the drop-down, the example shown in 4.6 is of type URL¹⁵. To remove an item, select the "Delete item" button and then confirm through the browser pop-up.

encodes (Recommended)

Bioschemas.org: For genes, this property is used to express in a generic way gene products encoded by this gene. Two more specific properties SIO:010082 (is translated into) and SIO:010080 (is transcribed into) should be used for (protein) translation and (RNA) transcription respectively. Note: bioschemas:encodes skos:closeMatch SIO:010078 Schema.org: This property is used to link to gene products encoded (even indirectly) from this gene such as RNA or proteins.

Add item

Figure 4.5: Bioschemas Profile Gene Encodes Array

item 1

URL

Delete item

Add item **Delete Last item**

Figure 4.6: Bioschemas Profile Gene Encodes Array Item

¹⁵<https://schema.org/URL> (accessed 12/03/2019)

The Property Tips are displayed using a custom JavaScript script, see Section 4.4.2.2 and is styled using Bootstrap 4. I decided to split the additional information into three collapsible sections, by marginality of the properties (Minimum, Recommended and Optional). Taking inspiration from the Bioschemas website¹⁶, I decided that the best way to display the examples to the user was to use a modal pop-up, shown in Figure 4.7. The decision to separate and nest the additional information was taken to reduce the information overload to the user and to hopefully not confuse them straight away.

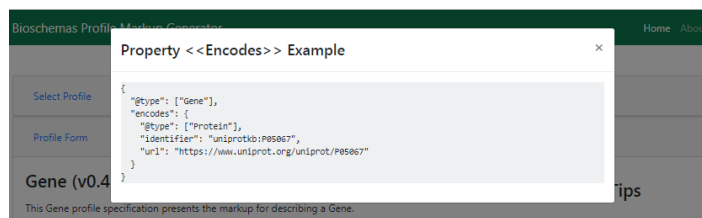


Figure 4.7: Bioschemas Profile Gene Encodes Example

Generated Markup

The Generated Markup section shows the user the generated markup based on the Bioschemas profile they selected and the information they provided, shown in Figure 4.8. The web application generates three formats of machine-readable data: JSON-LD, Microdata and RDFa, which can be accessed through the appropriate navigation tabs. The "Download" button also allows the user to download the generated markup (which ever format is currently being displayed) into a HTML file.

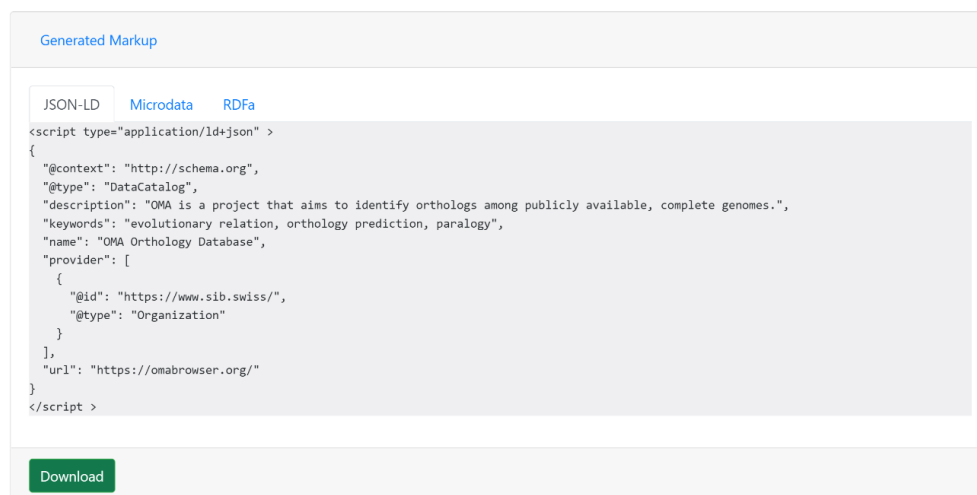


Figure 4.8: Bioschemas Profile Markup DataCatalog Example

¹⁶<https://bioschemas.org/> (accessed 07/04/2019)

4.4.2 Scripts, Libraries and APIs

4.4.2.1 JSON-Editor

To allow users to markup Bioschemas profiles, JSON-Schema¹⁷ was used to dynamically generate forms using a JavaScript library. The JSON-Schema is generated by the scripting tool (See Section 4.3) that is then used by a library to generate and display a HTML form. Through the iterative development cycle, three libraries were used as requirements and limitations of the different libraries came to light. I will now discuss the two libraries used for the prototype system in comparison to the library used in the final system.

The first iteration of the prototype was built using the JavaScript library SchemaForm.io¹⁸. It was quickly discovered that this library was inadequate for the system due to not supporting arrays of multiple types. Discussed in Section 3.2.3, properties that have a cardinality of many and multiple expected types can produce arrays of multiple types leading to an incompatibility between the library and system requirements. Listing 4.1 is an example markup of the Bioschemas profile DataCatalog¹⁹ in which the provider property has an array of multiple types (Organization²⁰ and Person²¹) which ultimately could not be produced by this library.

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "DataCatalog",
  "provider": [
    {
      "@type": "Organization",
      "name": "The International Union of Basic and Clinical Pharmacology",
      "url": "http://www.iuphar.org/"
    },
    {
      "@type": "Person",
      "givenName": "John",
      "familyName": "Doe",
      "url": "http://www.johndoe.com"
    }
  ]
}
</script>
```

Listing 4.1: DataCatalog Example Array of Multiple Types

¹⁷<https://json-schema.org/> (accessed 07/04/2019)

¹⁸<http://www.schemaform.io/> (accessed 03/11/2018)

¹⁹<https://bioschemas.org/specifications/DataCatalog/> (accessed 07/04/2019)

²⁰<https://schema.org/Organization> (accessed 07/04/2019)

²¹<https://schema.org/Person> (accessed 07/04/2019)

The second iteration of the prototype was built using the JavaScript library JSON-Forms²². In comparison to the first library used, this library was able to support arrays of multiple types but did not support the requirement of recursion. As previously discussed in Section 3.2.3, that in order to generate markup based on the Bioschemas profiles, the form should be able to handle recursive property types. This is due to properties in Schema.org types having the ability to contain themselves as an expected type.

The JSON-Schema specification has the ability to define schemas that can be reused and combined elsewhere in the schema [15]. Once a schema has been defined it can then be referenced using the keyword `$ref` where the value is a Universal Resource Identifier (URI) of the schema. Shown in Listing 4.2, the `PropertyValue` schema is defined and is then referenced within the `DataCatalog` schema using a local URI. The ability to reference schema allows schemas to be self-referential and can therefore create recursive schemas [15]. Shown in Listing 4.2 is an example of a recursive schema in which the schema `PropertyValue` can contain itself as a value.

Although JSON-Schema specification supports recursive schemas which in turn can handle recursive Schema.org property types that is needed to support the markup of Bioschemas profiles, the library JSON-Forms was unable to support this specification and therefore unable to support the markup of Bioschemas profiles.

```
"schema": {
  "title": "DataCatalog",
  "type": "object",
  "properties": {
    "identifier": {
      "title": "identifier",
      "oneOf": [
        {
          "$ref": "#/definitions/PropertyValue"
        }
      ]
    }
  },
  "definitions": {
    "PropertyValue": {
      "title": "PropertyValue",
      "type": "object",
      "properties": {
        "valueReference": {
          "title": "valueReference",
          "oneOf": [
            {
              "$ref": "#/definitions/PropertyValue"
            }
          ]
        }
      }
    }
  }
}
```

Listing 4.2: DataCatalog Recursive Schema Example

²²<https://github.com/brutusin/json-forms> (accessed 03/11/2018)

The final system was built using the JavaScript library JSON-Editor²³. The following list of features are supported by JSON-Editor and highlights how they were utilised to allow users to markup Bioschemas profiles:

Descriptions: Supports the creation of markup by displaying a detailed description of each property to the user.

Arrays: Supports the creation of arrays allowing properties that have a cardinality of many to be marked up.

One Of: Supports the ability for users to choose which expected type for a property to markup.

Recursion: Supports the markup of Schema.org types where properties have a recursive expected type.

Required Properties: Supports marginality of properties by requiring minimum properties to be marked up.

Property Ordering: Supports ordering of form inputs / properties to display the most relevant properties first. i.e Minimum then Recommended then Optional properties.

Definitions & References: Supports the ability to create efficient JSON-Schema through the use definitions and references of schemas.

Default & Hidden Properties: Supports the creation of JSON-LD by providing JSON-LD key value pairs like `@type` and `@context` without the unnecessary form inputs being displayed to the user.

4.4.2.2 Additional Information Panel

A custom JavaScript script was used to display additional information (examples and controlled vocabulary) to the user to help markup Bioschemas profiles. Originally, I had planned to use a JavaScript library like Tabular²⁴ to display the JSON file containing the additional information in a table. As previously mentioned in Section 4.4.1, I decided that displaying all the information at once in a table could overwhelm the user. Therefore, I created a custom script to display the information in a controlled way which can be seen in Figure 4.3.

4.4.2.3 RDF Translator API

The RDF Translator API²⁵ was used to convert the generated JSON-LD into additional machine-readable formats for the web: RDFa and Microdata. The client-side makes a AJAX request containing the JSON-LD that is to be converted to the desired formats. The resulting data is returned in either RDFa or Microdata, dependant on the format specified in the request URL.

²³<https://github.com/json-editor/json-editor> (accessed 16/03/2019)

²⁴<https://github.com/json-editor/json-editor> (accessed 16/03/2019)

²⁵<https://rdf-translator.appspot.com/> (accessed 16/03/2019)

As RDFa and Microdata use HTML as a basis to add machine-readable data to the web, the data returned is in the HTML format. To display the HTML to the user, it first needs to be escaped so that it will be displayed correctly on the web page. This was achieved by using jQuery's `text`²⁶ function to set the content of the selected element to the HTML as it automatically escapes the necessary characters for the HTML to be displayed.

4.4.2.4 Bootbox.js

Bootbox.js²⁷ was used to programmatically create and display Bootstrap pop-up modals to provide the user with example markups, shown in Figure 4.7. Without the library, a system would have to be implemented to create, manage and remove any of the required DOM elements or JavaScript event handlers to produce the pop-up modals which can be difficult and time consuming to implement. Due to the predicted difficulty in implementing the same functionality provided by Bootbox.js and the time constraints associated with the project, Bootbox.js was used to produce the desired functionality.

4.4.2.5 Download.js

Download.js²⁸ was used to allow the user to download the generated markup of their selected Bioschemas profile. An additional library was needed as the W3C standard File API: Writer that can be used to store files on a client's machine is currently only supported by the browser Google Chrome [34][22]. This issue was not inline with the requirements outlined at the beginning of the project (Non-Functional Requirement NFR3: The system should be accessible through the top internet browsers: Google Chrome, Mozilla Firefox and Apple Safari.), so an additional library was used to provide the desired functionality.

4.4.3 Hosting the Web Application

The implementation began on my personal machine, with the intention of deploying the web application on a publicly accessible web server. The decision to make it publicly accessible was to enable the Bioschemas community during meetings and workshops to provide feedback (See Section 5.1.1) on the system whilst the system was still in development, without having Dr. Alasdair Gray host the web application himself.

²⁶<http://api.jquery.com/text/> (accessed 16/03/2019)

²⁷<https://github.com/makeusabrew/bootbox> (accessed 16/03/2019)

²⁸<http://danml.com/download.html> (accessed 16/03/2019)

The web application built is a HTML/JavaScript application and because of this it can be deployed on any standard web server. For this reason, I was able to host the application on the the Heriot-Watt MACS Student/Development web server. The choice to use this server was made as a student at Heriot-Watt University I was already given access and along with the ability to host the application on any web server no additional setup was required.

4.5 Implementation Challenges

As the system relies heavily on the generation of JSON-Schema and an open-source library to utilise the generated JSON-Schema, it was inevitable that there would a few issues surrounding this area. The key use and generation issues encountered during development with JSON-Schema were:

1. Bioschemas profile data format inconsistencies.
2. JSON-Editor unable to handle documented features.
3. JSON-Editor unable to handle large JSON-Schema.

4.5.1 Issue 1: Data Format Inconsistencies

This issue was caused by the inconsistent formatting of data within the declarative descriptions of the Bioschemas profiles. As Bioschemas is currently in a stage where they are rapidly evolving and refining their profiles, older formats of data still exists in a number of current profiles.

Each Bioschemas profiles has a list of properties in which each property has a list of expected types. Once the declarative description of a profile has been converted from YAML to JSON to be further processed, each property should contain an array of expected types. Although, in some cases the list of expected types are conjoined into a singular string making it difficult to process. For example in Bioschemas profile ProteinAnnotation (v0.4)²⁹, the expected output of expected types for the property location is shown in Listing 4.3, while the actual output is shown in Listing 4.4.

²⁹<http://bioschemas.org/specifications/ProteinAnnotation/> (accessed 01/03/2019)

```
[ " Place " , " PostalAddress " , " PropertyValue " , " Text " , " URL " ]
```

Listing 4.3: Expected Output

```
[ " Place orPostalAddress orPropertyValue orText orURL " ]
```

Listing 4.4: Actual Output

To solve this issue, the singular string was split into an array of individual types and the additional "or" before each type was removed, keeping the format consistent with the expected output. This kept the format of the expected types consistent across the different Bioschemas profiles and ready to be further processed.

4.5.2 Issue 2: JSON-Editor Documentation

The inconsistency in the documentation of the JavaScript library JSON-Editor in comparison to what could be implemented with the library caused two issues. Both issues relate to the libraries documentation stating that "it has full support for JSON-Schema Version 3 and 4" [14]. Meaning the library should fully support the use of keyword `oneOf` specified in JSON-Schema version 4³⁰, although this seems not to be the case.

The first issue encountered was due specifying the formats of strings. By specifying the format of a string, the form input generated only accepts the format specified. For example, specifying the format email will only allow valid emails to be accepted. According to the documentation, JSON-Editor supports the necessary formats to allow users to enter the necessary data types specified by Schema.org DataTypes³¹. Although when formats date or date-time are used in conjunction with the keyword `oneOf` (Example shown in Listing 4.5) it leads to an error in which the form input would not be displayed to the user.

```
"schema": {
  "title": "String Format Issue",
  "oneOf": [
    {
      "title": "Date",
      "type": "string",
      "format": "date"
    },
    {
      "title": "Date-Time",
      "type": "string",
      "format": "date-time"
    }
  ]
}
```

Listing 4.5: Example Schema String Format Issue

³⁰<https://tools.ietf.org/html/draft-fge-json-schema-validation-00> (accessed 01/03/2019)

³¹<https://schema.org/DataType> (accessed 01/03/2019)

To handle this issue, I decided that the user must be able to provide data for these data types as this issue is detrimental to the systems functionality. Until a solution is found, the offending formats were removed from the JSON-Schema to allow the system to function correctly. This compromise unfortunately leads to the possibility that user can now provide incorrect formats of date and date-time when creating a markup for a profile.

The second issue encountered was due to recursive definitions. As previously discussed in Section 4.4.2.1, the JSON-Schema specification supports definitions of a recursive nature. JSON-Editor supports these recursive definitions, although when used in conjunction with the `oneOf` specification (Example shown in Listing 4.6) it leads to the recursive form inputs not being displayed to the user.

```
"schema": {
  "title": "String Format Issue",
  "$ref": "#/definitions/recursion",
  "definitions": {
    "recursion": {
      "title": "recursion",
      "oneOf": [
        {
          "title": "example",
          "$ref": "#/definitions/recursion"
        }
      ]
    }
  }
}
```

Listing 4.6: Example Schema Recursion Issue

This issue was fortunately negated as a result of solving the issue of JSON-Editor being unable to handle large JSON-Schema. Discussed in Section 4.5.3, to reduce the size of the JSON-Schema, nested Schema.org types were removed and in place an International Resource Identifier (IRI) was given. As a result of this solution, no recursive definitions were used and so no solution was needed for this issue.

To contribute to the open-source community and help the development of JSON-Editor, I fed back these issue to the developers by submitting the issues on their Github repository³². To help the developers in understanding the issues I provided a detailed explanation of the errors and gave examples of the error prone JSON-Schemas. Issue 323³³ was submitted in relation to the string format issue and Issue 248³⁴ was submitted in relation to the issue of recursion.

³²<https://github.com/json-editor/json-editor/issues> (accessed 02/03/2019)

³³<https://github.com/json-editor/json-editor/issues/323> (accessed 02/03/2019)

³⁴<https://github.com/json-editor/json-editor/issues/248> (accessed 02/03/2019)

4.5.3 Issue 3: Large JSON-Schema

This issue was due to the JavaScript library JSON-Editor being unable to handle JSON-Schemas of a large size. The large size of the JSON-Schema comes from the extensive number of types and properties provided by the Schema.org vocabulary. For example, the Bioschemas profile DataCatalog (v0.1)³⁵ contains the property called identifier, where the expected types are: PropertyValue³⁶, Text³⁷ or URL³⁸. Where PropertyValue has 20 properties where the expected types are 12 additional Schema.org types. As Schema.org types can reference other types, the overall size of the JSON-Schema can grow exponentially, thus becoming unusable by JSON-Editor.

To allow the JSON-Schema to work with JSON-Editor, a compromise needed to be made to limit the overall size. We decided that instead of nesting Schema.org types (Example shown in Listing 4.7) the JSON-LD object for the type would be externally referenced through an Internationalised Resource Identifier (Example shown in Listing 4.8). This dramatically reduced the size of the JSON-Schema allowing it to work with JSON-Editor. Unfortunately this compromise stops the user from being able to markup Schema.org types through the web application and instead having to create, host and reference the markup for the Schema.org types externally.

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "DataCatalog",
  "provider": {
    "@type": "Person",
    "givenName": "John",
    "familyName": "Doe",
    "gender": "Male",
    "birthDate": "01/01/1970"
  }
}
</script>
```

Listing 4.7: Example Markup Nested

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "DataCatalog",
  "provider": {
    "@type": "Person",
    "@id": "http://www.example.com/JohnDoe"
  }
}
</script>
```

Listing 4.8: Example Markup IRI

³⁵<http://bioschemas.org/specifications/DataCatalog/> (accessed 02/03/2019)

³⁶<https://schema.org/PropertyValue> (accessed 02/03/2019)

³⁷<https://schema.org/Text> (accessed 02/03/2019)

³⁸<https://schema.org/URL> (accessed 02/03/2019)

4.6 Comparison System

The final usability evaluation (See Section 5.1.2) consisted of comparing the developed system against a second similar system. The system that was chosen for comparison was Kaizen, see Section 2.5.1. As Kaizen is a Google Sheets template it can be easily modified, making it the best choice for the comparison system as it can be customised to suit the needs of the evaluation.

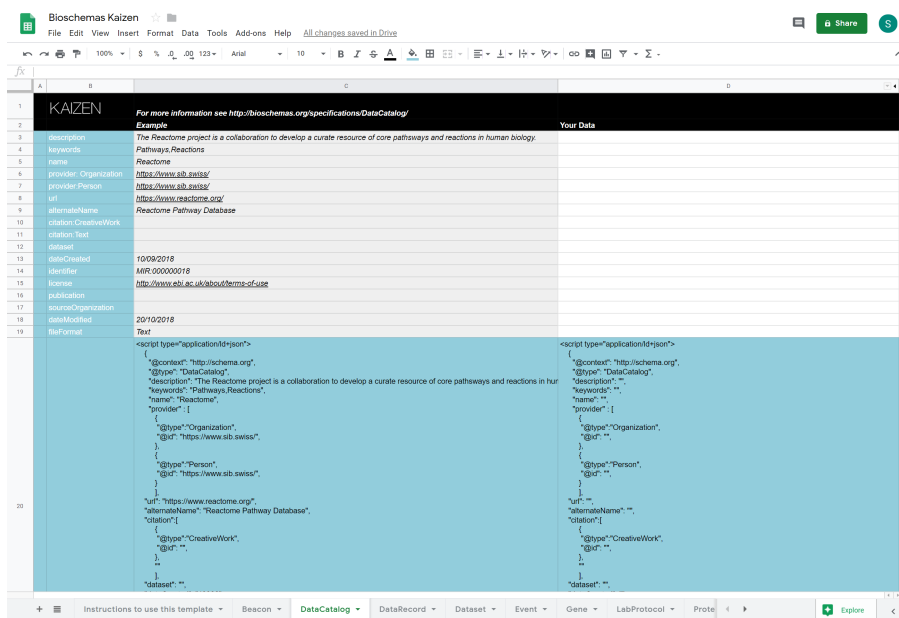


Figure 4.9: Comparison System

The usability evaluation consisted of participants completing a set of three tasks with the comparison system, therefore it only needed to support the ability to perform these tasks. This led to participants only being able to markup the DataCatalog (v0.1)³⁹ and Gene (v0.4)⁴⁰ profiles as these were the only profiles needed for the tasks. Although only DataCatalog and Gene were needed, dummy tabs for other profiles were created to keep the system inline with what the final system would be like to use. As Kaizen provides no additional information like descriptions and controlled vocabularies to the user, I took the approach of directing the user to an external source (Bioschemas Website⁴¹) for this information. The final modified version of the comparison system can be seen in Figure 4.9.

³⁹<http://bioschemas.org/specifications/DataCatalog/> (accessed 02/03/2019)

⁴⁰<http://bioschemas.org/specifications/Gene/> (accessed 02/03/2019)

⁴¹<http://bioschemas.org/specifications/> (accessed 02/03/2019)

4.7 Maintainability

After the implementation of both systems, main and comparison, the maintainability of each system was compared in order to see how easy each system was to keep up to date with the latest Bioschemas profiles.

To update the main system to the latest Bioschemas profiles, the webmaster would simply have to re-run the scripting tool (See Section 4.3) to retrieve the updated files. Once the updated files have been generated the webmaster would then replace the files located on the web server with the newly generated files. Once the files have been replaced, the system will automatically be ready to use with the latest Bioschemas profiles.

As the main system uses a multitude of libraries (See Section 4.4.2), the stability of these libraries are a key concern for the maintainability of the system. Libraries JSON-Editor and Bootbox.js see regular updates and fixes and as a result only stable releases are published. However, Download.js has not received any updates in over 10 months. This is a cause for concern as browsers are updated regularly which could lead to the library becoming unusable in the near future.

To update the comparison system to the latest Bioschemas profiles, the webmasters would have to manually check and update each profile as there is currently no automated way of doing so. Additionally, as users create a local copy of the Google Sheet template to use, to receive the latest Bioschemas profiles the users have to check that they are using the latest version and download the newest version if available to keep up to date.

4.8 Summary

Although there were a few issues surrounding the generation and use of the JSON-Schema, the core functionality of the proposed system was successfully developed. A user can select which Bioschemas profile they wish to generate markup for, then provide the data using the form with the help of descriptions, controlled vocabularies and examples. Once the form is filled out they can then generate JSON-LD, Microdata or RDFa to use on their web page. Furthermore, a second comparison system was implemented for the final usability evaluation and the maintainability of each of the system was compared.

Chapter 5

Evaluation

5.1 Usability Evaluation

Once development of the system had come to a conclusion a usability evaluation was carried out in order to find out how well users interacted with the system. The system is aimed towards members of the Bioschemas and life science communities. These users will likely have varying knowledge and technical abilities, making it vital to design an interface that is as user-friendly and intuitive as possible. For these reasons, members of these communities would have been the ideal test subjects for this evaluation. However, due to logistical issues the ideal test subjects would not be able to participate in A/B testing within a controlled environment. To receive feedback from the ideal test subjects as well as conduct A/B testing to determine the usability of the system, the usability evaluation was carried out in two stages:

1. An initial evaluation of the prototype system with the ideal test subjects.
2. A final evaluation of the complete system with alternate test subjects.

5.1.1 Prototype Usability Evaluation

The prototype usability evaluation subjects consisted of participants from various Bioschemas Meetings¹ that were run by Dr. Alasdair Gray. The evaluation was initially conducted with the first prototype system (Section 3.2.3) that was used to help gathered the initial system requirements and later a second prototype system comparable to the final system (Section 4.4.1) without the additional information panel.

¹<http://bioschemas.org/meetings/> (accessed 07/03/2019)

The prototype usability test did not consist of a set task. The test subjects were given a demonstration of the prototype and then were given the opportunity to create markup for their own data resource. Once they tested the prototype they were asked to fill out an evaluation questionnaire shown in Appendix A.1 on various aspects of the system and any additional features they would like to see. This information was then used to improve the final system and tailor it towards the end user. Although useful, the information collected may not be definitive. By not having a set task and control over the conditions the test was conducted in, the resulting information may be affected in comparison to a controlled usability test.

5.1.1.1 First Prototype Usability Results

The prototype usability questionnaire was completed by eleven test subjects. The questionnaire presented test subjects with a total of seven questions. Questions one to four, used a scale, ranging from one to ten, to assess their current knowledge and how they felt about the interface of the prototype. Question five and six, asked the subjects about the information provided and the results. Finally, question seven asked the test subjects to provide written feedback on what features they would like added to the final system.

As shown in Figures 5.1 and 5.2, the test subjects had a wide range of familiarity with Schema.org and Bioschemas.org, the underlying premise of the system. This information highlighted the necessity to create the final system to be as user-friendly as possible, by providing all the information necessary to use the system without any previous knowledge.

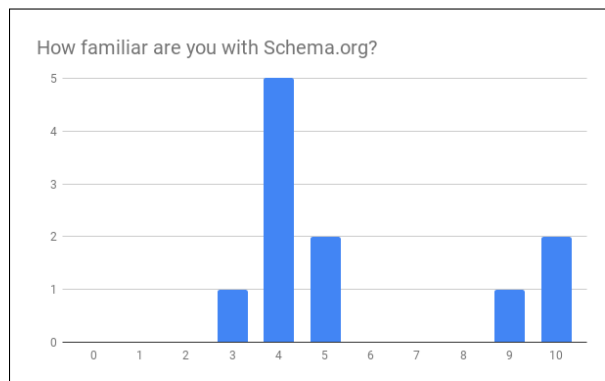


Figure 5.1: Prototype Evaluation Question 1

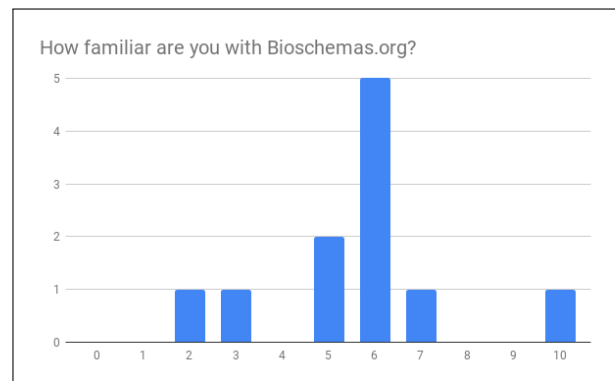


Figure 5.2: Prototype Evaluation Question 2

From questions three and four, the overall consensus was that the interface of the prototype was easy to use and appealing, as shown in Figures 5.3 and 5.4. With the positive reaction of the prototype, the final system was designed using the same layout and appearance. However, a few test subjects commented that they found it difficult to distinguish the marginality of a property (Minimum, Recommend or Optional). Only the minimum properties had a signifier of a small red asterisk next to the property name, but test subjects did not know the meaning of the asterisk until they validated their data. For the final system, this feedback was taken into consideration and a more complete and visible marginality system was implemented.

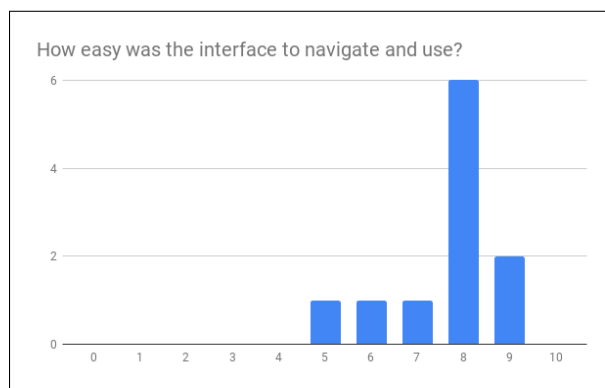


Figure 5.3: Prototype Evaluation Question 3

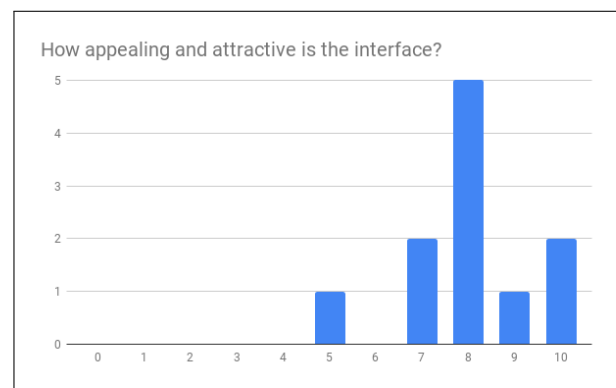


Figure 5.4: Prototype Evaluation Question 4

Question five had a clear divide in answers from test subjects, as shown in Figure 5.5. This divide may be due to the nature of the prototype usability test, as test subjects were instructed to create their own markups. This meant test subjects without any personal data or previous knowledge looked for examples to use, but none were provided on the prototype system. For the final system, this feedback was taken into consideration and where possible additional information like examples and controlled vocabularies were provided.

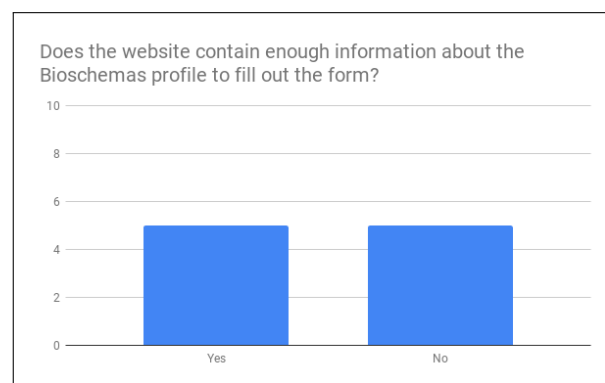


Figure 5.5: Prototype Evaluation Question 5

Question six asked test subjects if the system generated the expected result from the data users had provided, as shown in Figure 5.6. In some cases the expected result was unfortunately incorrect. A couple users found that on validation that some properties in the DataCatalog profile were required, but were not actually required in the profiles definition. This was due to human error when manually creating the JSON-Schema to generate the form, as some properties had the required attribute when they should not have. For the final system, these errors will be less likely to occur as the JSON-Schema will be generated directly from the Bioschemas profile without any human intervention.

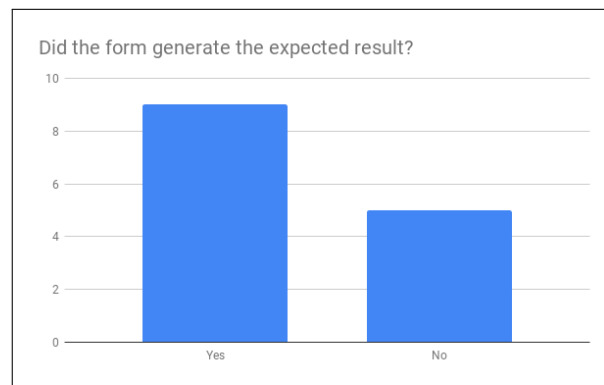


Figure 5.6: Prototype Evaluation Question 6

Question seven asked test subjects if there was any extra features that they would like to see in the final system. This question had overlapping information with the previous questions in which the test subjects stated that they would like to more information to help the users fill out the forms through examples and controlled vocabularies. An additional feature that a test subject requested was to have the resulting data in different formats like RDFa and Microdata, this feature was then included in the system requirements prioritised as a could feature.

In summary, this prototype usability evaluation has been successful in showing that the core functionality and usability of the system works well; although the information provided to the users could be vastly improved. The prototype system was used as a starting platform for the final system and with the feedback provided, the final system was tailored directly towards the end user.

5.1.1.2 Second Prototype Usability Results

The prototype usability questionnaire was completed by two test subjects for the second version of the prototype. The questionnaire presented contained the same questions previously given to test subjects in the first prototype usability evaluation (See Section 5.1.1). Unfortunately as only two test subjects completed the questionnaire in which no qualitative feedback was given, no definitive conclusion can be drawn to compare the first prototype to the second. From the quantitative data collected, shown in Appendix A.2, we can see that test subjects thought the interface was easy to navigate and use and that the prototype was appealing and attractive as well as providing the desired functionality, but these conclusions can not be considered definitive due to the environment in which the usability evaluation was conducted in and the low participation rate.

5.1.2 Final Usability Evaluation

The final usability evaluation subjects consisted of students from Heriot-Watt University, mainly consisting of students with a Computer Science background. To test the usability of the system developed, a second similar system was used for comparison. In the following sections I will refer to the system developed as **System A**, which can be seen in Section 4.4.1, and the comparison system as **System B**, which can be seen in Section 4.6.

For each of the systems, the subjects were given a set of tasks to complete. The users were timed for each task, to understand how easily they figured out how to use the systems and any difficulties during the tasks were noted. To eliminate any starting bias towards a system, test subjects alternated between starting on each of systems. Also, to reduce the diversity in times affected by typing speed, test subjects copy and pasted data from the task sheets into each of the systems.

Once the users completed their tasks they were asked to fill out a short questionnaire on each system. This questionnaire asked the users to evaluate various aspects of the system as well as leaving any recommendations to improve the system. The task sheet, questionnaire and consent form signed by the test subjects can be found in Appendix B.

5.1.2.1 Tasks

Each test subject was assigned the same three tasks for both systems, although an additional task was required for System A. Test subjects will begin Tasks 1, 3 and 4 at the respective Home Page of the system. Task 2 will start where Task 1 left off.

Task 1: Users were asked to markup a DataCatalog profile using the data supplied on their task sheet.

Task 2: Users were asked to save the resulting data from Task 1.

Task 3: Users were asked to find the Controlled Vocabulary for a property in the Gene profile.

Task 4: Users were asked to markup a more complex Gene profile using the data supplied on their task sheet.

5.1.2.2 Final Usability Results

The final usability test was completed by eight test subjects. The set of the tasks for each system were completed easily by all the test subjects, as reflected by the relatively consistent times taken with the exception of a few outliers, shown in Appendix B.4. With a faster average task completion time (Shown in Figure 5.7) as well as feedback gathered from the questionnaire, it shows that System A was easy to use and was preferred by all test subjects, but could use some minor improvements.

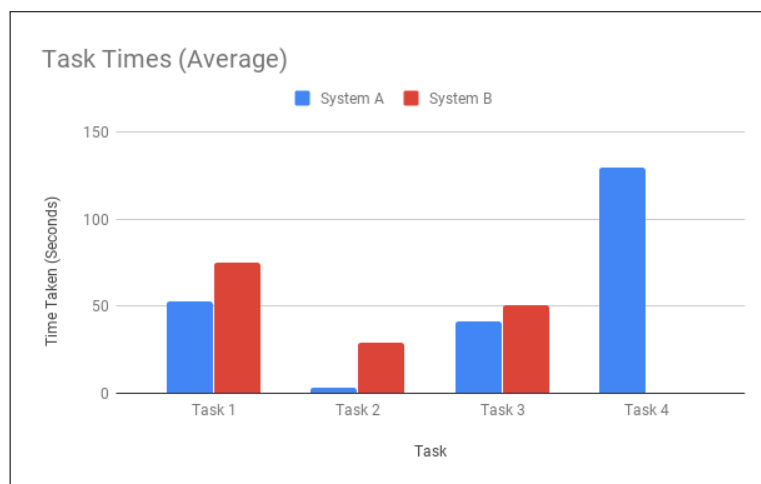


Figure 5.7: Average Task Completion Times

The questionnaire presented test subjects with a total of seven questions. Questions one to four used a Likert scale, ranging from "Strongly Disagree" to "Strongly Agree", to assess how the subjects felt about each of the interfaces and the amount of information provided by the systems. Additionally, after each question a comment section was also provided to collect any additional information from the test subjects. Question six and seven asked the test subjects to provide written feedback on what could be added to each system or what could be changed. Question eight then asked which system would they prefer to use.

As shown in Figure 5.8, the majority of test subjects considered System A to be easy to use and navigate. The test subject who selected "Neither Agree nor Disagree" left further feedback stating that they found the forms nested design along with the buttons to add and remove items confusing. Two other test subjects also noted that they found the forms buttons to be confusing and another test subject suggested that they should be better separated to help with this issue.

In comparison, only a few test subjects felt that the interface of System B was easy to use, shown in Figure 5.9. One test subject stated that they found it easy to use as they already had experience with Google Sheets and were familiar with the interface and layout. However, an issue that was apparent in most evaluations, stemmed from Task Two. Users were asked to save the resulting data from Task One into a HTML file. Users expected that double-clicking would highlight the data to be copied, although in Google Sheets double-clicking shows the user the formula behind the data. This often resulted in test subjects taking multiple attempts to copy to data successfully, increasing the time taken to complete the task, which can be seen in Appendix B.4 with Figures B.3 and B.4.

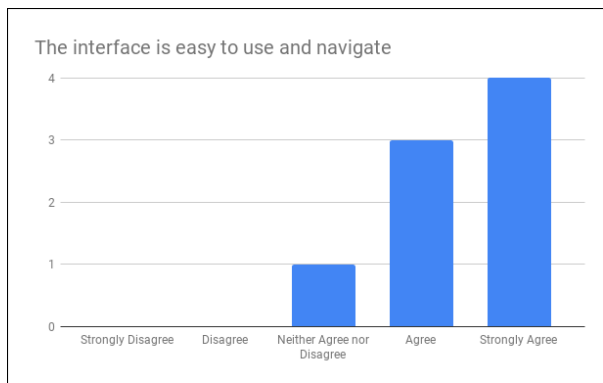


Figure 5.8: Question 1: System A

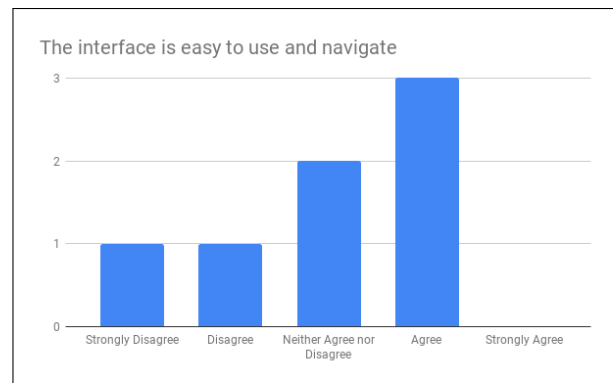


Figure 5.9: Question 1: System B

Figure 5.10 shows that every test subject agreed that they found System A to be aesthetically pleasing. One test subject commented that they liked the flat design of the system but felt that navigation bar could utilise some graphics. Originally, I had intended to use the Bioschemas logo in the navigation bar. Unfortunately, due to selecting the same green for the navigation bar used in the logo, it would have blended in too well and would not have been clearly seen.

In comparison, most test subjects agreed that System B was not aesthetically pleasing, shown in Figure 5.11. Two test subjects commented that a separate application would have been a better choice for the design of the interface rather using Google Sheets. Another test subject commented that even though the system was functional, they thought that the system was poorly designed and not aesthetically pleasing.

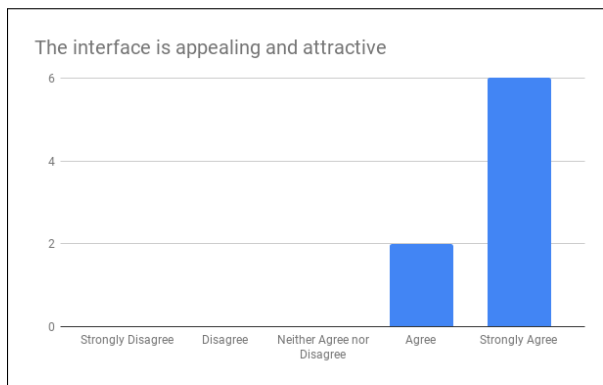


Figure 5.10: Question 2: System A

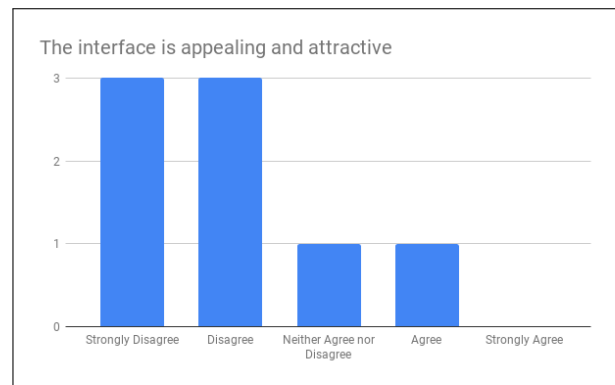


Figure 5.11: Question 2: System B

Question three has the most diverse range in answers from test subjects, as shown in Figures 5.12 and 5.13, which stemmed from Task Three of the usability test. Users were tasked to find the Controlled Vocabulary of a property in the Bioschemas profile Gene.

System A, contained all information necessary to complete Task Three. Although most test subjects found it easy to find the information, some test subjects found it slightly more difficult. One test subject commented that they found the information hard to find as it was hidden within the collapsible sections. To increase the visibility of the information another test subject suggested to label what was contained in the collapsible sections.

System B, did not contain the information necessary to complete Task Three, instead it pointed to an external website. A couple test subjects felt that the website that contained information needed was well formatted, but because they had to visit an external website they selected "Neither Agree nor Disagree". Whereas other test subjects felt that they were not able to find the information quickly as they had to visit an external website and disagreed with the question.

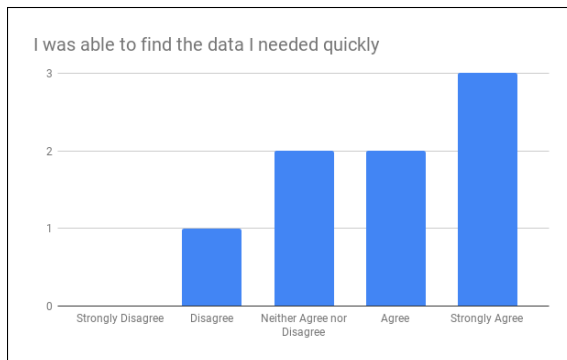


Figure 5.12: Question 3: System A

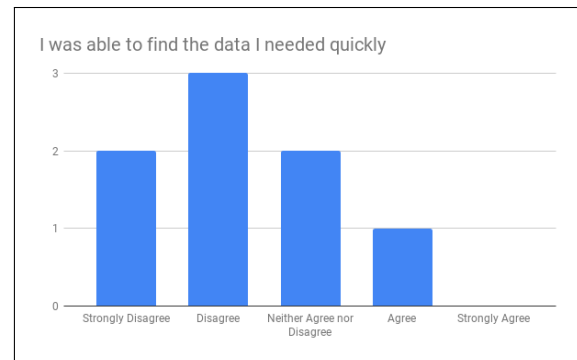


Figure 5.13: Question 3: System B

Figure 5.14 shows that every test subject agreed that System A contained enough relevant information, whereas test subjects agree System B did not agree, shown in Figure 5.15. The test subjects felt that to complete the tasks provided that all the information should be provided within the system. The results from this question showed this opinion, as System A contained all the information necessary whereas System B pointed to external website.

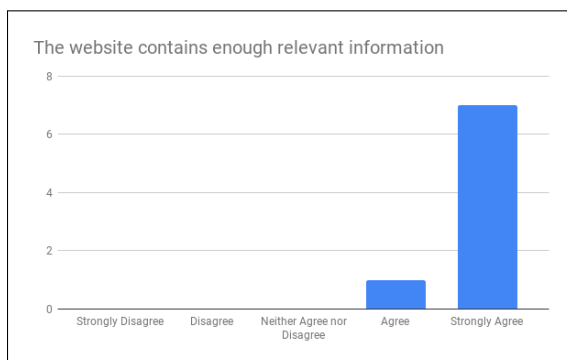


Figure 5.14: Question 4: System A

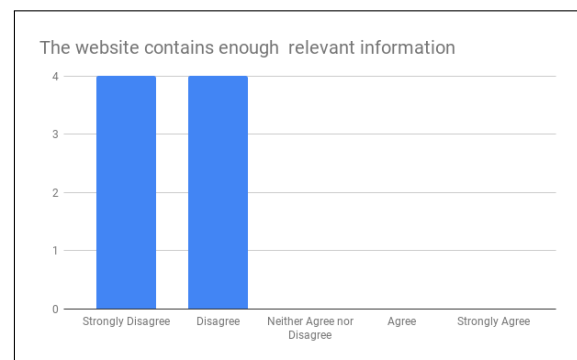


Figure 5.15: Question 4: System B

Question five asked test subjects "Is there anything that you would change about the system?". For System A, test subjects reiterated that they found the forms nested design along with the add item to be confusing and would like these features to be redesigned to be more user friendly. For System B, test subjects reiterated that they would have preferred a separate application to Google Sheets and commented that they would like all the information required to be available and not point to an external website.

Question six asked test subjects "Is there anything that you like to add to the system?". For System A, only one test subject suggested an addition. Their suggestion was to add graphical feedback when performing actions on the form. For example, when an item is added the area behind the item could change colour letting the user know where the item was added. For System B, a couple test subjects suggested that they would like a feature to easily download the resulting data to a HTML, like in System A.

In Summary, this final usability evaluation has been successful in comparing the functionality and the usability of the final system (**System A**) against a similar system (**System B**). We can see from Figure 5.16 and previous questions, that the test subjects preferred to use system A as it was easy to use, more aesthetically pleasing and provided all the information where necessary; although labelling the information provided could be improved to better the visibility.

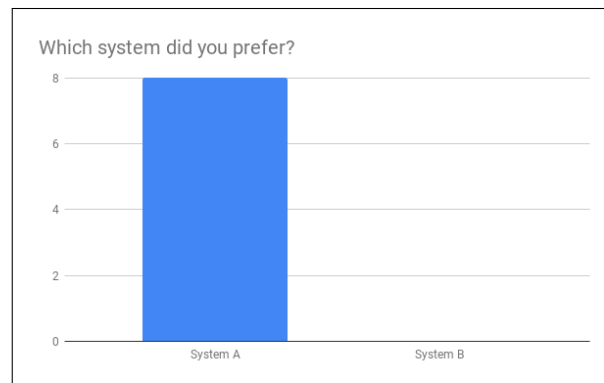


Figure 5.16: Question 7: System A vs System B

5.1.3 Summary of Usability Evaluation

To summarise, both the final and prototype evaluation were successful in showing that the systems provided the desired functionality outlined at the beginning of the project. It also highlighted the design of the systems were successful in providing an effective, intuitive and aesthetically pleasing user interface and thus test subjects preferred System A over System B. Feedback given during the prototype evaluation was used to further refine the final system towards the target user and feedback given during the final evaluation has been included as future works in Section 6.3.

5.2 Requirements Evaluation

A requirements evaluation was carried out in order to analyse how many system requirements were implemented in the final system compared to the proposed requirements. The evaluation identified which system requirements were successfully implemented, and which were not, along with a description of what was achieved.

5.2.1 Functional Requirements Evaluation

FR 1 The system must allow users to enter data into the form.

- Priority: Must
- Status: Complete
- Explanation: The system allows users to enter data through form inputs.

FR 2 The system must generate JSON-LD compliant data.

- Priority: Must
- Status: Complete
- Explanation: The system generates JSON-LD compliant data as tested by the Google Structured Data Testing Tool.

FR 3 The system must allow users to select the Bioschemas profile.

- Priority: Must
- Status: Complete
- Explanation: The users can select which Bioschemas profile they would like to generate a markup for through a drop-down list.

FR 4 The system must be able to handle Schema.org data types.

- Priority: Must
- Status: Complete
- Explanation: The form inputs can handle each of the Schema.org data types.

FR 5 The system must be able to handle cardinality and marginality.

- Priority: Must
- Status: Complete
- Explanation: The system can handle cardinality by requiring the minimum properties and displaying the cardinality next to the property name. It can also handle marginality through the JSON-Editors ability to support arrays of multiple types of objects.

FR 6 The system should generate the forms based on a declarative specification.

- Priority: Should
- Status: Complete
- Explanation: The system generates the JSON-Schema required to generate the form based on declarative specifications from Bioschemas and Schema.org.

FR 7 The system should be able to handle recursive properties.

- Priority: Should
- Status: Complete
- Explanation: The JSON-Editor library used for the form is able to handle the recursive properties.

FR 8 The system should be able to prioritise form inputs.

- Priority: Should
- Status: Complete
- Explanation: The form inputs / properties are prioritised by their marginality. First being minimum, then recommended, then optional.

FR 9 The system should be able to validate the generated markup against the Bioschemas profile.

- Priority: Should
- Status: Partially
- Explanation: JSON-Editor allows the form inputs to be validated against the JSON-Schema but due to an unknown issue the validation does not work. As JSON-Schema only validates the structure of the data produced and not the data provided, controlled vocabularies would not be able to be validated. Validation could be done through the Validata tool as it would validate both structure and data provided but currently does not have the functionality to do so.

FR 10 The system should display descriptions of each property, as well as examples.

- Priority: Should
- Status: Complete
- Explanation: Descriptions of each property are displayed next to the relevant form input. Examples are displayed through the Property Tips section and are displayed as a pop-up modal.

FR 11 The system should display the controlled vocabulary of each property.

- Priority: Could
- Status: Complete
- Explanation: Control Vocabularies are displayed through the Property Tips Section.

FR 12 The system could generate Microdata and RDFa along side JSON-LD.

- Priority: Could
- Status: Complete
- Explanation: The system generates Microdata and RDFa by converting the generated JSON-LD using the RDF Translator API.

FR 13 The system could allow the user to download the data generated.

- Priority: Could
- Status: Complete
- Explanation: The user can download a HTML file containing either the generated JSON-LD, Microdata or RDFa.

FR 14 The system wont be able to dynamically update declarative specifications from an online source.

- Priority: Wont
- Status: Incomplete
- Explanation: This requirement was not expected to be complete as expected, due to time constraints and is included as a future work.

5.2.2 Non-Functional Requirements Evaluation

NFR 1 The system must be easy to use through a simple user interface.

- Priority: Must
- Status: Complete
- Explanation: Results from the usability evaluation that the test subjects were happy with the interface and its ease of use, although some minor improvements are needed.

NFR 2 The system must be available 24/7.

- Priority: Must
- Status: Complete
- Explanation: The web application is publicly accessible 24/7, hosted on the Heriot-Watt MACS Student/Development web server.

NFR 3 The system should be accessible through the top internet browsers: Google Chrome, Mozilla Firefox and Apple Safari.

- Priority: Should
- Status: Complete
- Explanation: The web application is accessible through the top web browsers, tested functionality on all three browsers specified.

NFR 4 The system could have two interfaces. One for standard users. Another for administrative users.

- Priority: Could
- Status: Incomplete
- Explanation: Due to time constraints, I was not able to implement this requirement but is included as future work.

5.2.3 Summary of Requirements Evaluation

All but three of the requirements outlined at the beginning of the project were completed, with only one partially complete and two incomplete. Of the two incomplete requirements, one was a functional requirement that was prioritised as a wont, while the other was non-functional that was prioritised as a could. While not vital for the systems functionality, it would be beneficial for these requirements to be complete, thus they have been included as future development tasks in Section 6.3.

Chapter 6

Conclusion

6.1 Project Aims

As stated in Chapter 1, the aim of this project was to "develop a system to support users in the creation of Bioschemas markup for their web resources". This aim was successfully completed as the final system allows the user to select, provide data and generate a markup based on a Bioschemas profile with the aid of additional information like examples and controlled vocabularies.

6.2 Project Objectives

At the start of the project, I set three objectives for this project. I will restate these objectives and explain to what extent they have been achieved.

The first objective was to "review existing tools for generating and validating Schema.org markup". This objective was successfully completed as the resulting information can be seen in Section 2.5. The information gathered was extremely useful in helping design the final system and finding a comparison system to use in the usability test.

The second objective was to "allow users to easily generate Bioschemas markup through an intuitive and aesthetically pleasing user interface". This objective was met by the implementation of the proposed system and was verified through an evaluation. The evaluation concluded that the proposed system was successfully implemented as all but three of the initial requirements were incomplete. Additionally, it was verified through a usability test (See Section 5.1.2.2) where members of the Bioschemas community and Computer Science students from Heriot-Watt University evaluated the final system. The evaluation showed that interface was aesthetically pleasing and easy to use, but there are a few minor areas in which the system could be improved upon and have been included as future works, discussed in Section 6.3.

The third and final objective was "to evaluate the final system's usefulness and validate that the generated data is syntactically correct and compliant with the Bioschemas.org specification". As the prototype system is currently deployed and in use, I believe that the final system will be just as if not more useful as it allows many more Bioschemas profiles to be marked up and contains additional information to help the user. Although, the markup produced is syntactically correct as tested with Google's Structured Data Test Tool, I was unable to validate the markup against the Bioschemas profile. JSON-Schema produced by the system could have been used to validate the markup but due to an unknown issue with JSON-Editor the functionality did not work correctly. A drawback to using JSON-Schema would not allow for the markups controlled vocabularies to be validated. However, Validata mentioned in Section 2.5, could have provided the functionality needed but at this moment the functionality is only planned and has not yet been implemented.

6.3 Future Work

After developing the system and conducting evaluations during different stages of the development, there are a few ways in which the final system could be further developed and improved upon. A couple of these future works are system requirements, but due to technical issues and time constraints, I was unable to implement them. Future development of the system would involve:

- **Profile Validation:** Implement the ability to validate the generated markup against the selected Bioschemas Profile.
- **Advanced User:** Allow more knowledgeable users to have more control over the form, by allowing modifications of the form to better suit their needs.
- **More Bioschemas Profiles:** To best support the community in creating markups for Bioschemas profiles, the system should support all the profiles available including draft profiles¹.
- **Auto Generate Files:** Automatically detect new and updated Bioschemas profiles and generate the required files to keep the system always up to date.
- **User Interface Improvements:** Feedback from the final usability evaluation highlighted a few ways to enhance the usability of the interface. Future improvements would involve:
 - Information provided needs to be better labelled in order to improve visibility.
 - Buttons to add and remove items on the form need to be better spaced out.
 - A better nested design of the form to decrease confusion.

¹<http://bioschemas.org/specifications/drafts> (accessed 13/03/2019)

Bibliography

- [1] Ben Adida, Mark Birbeck, Shane McCarron, and Ivan Herman. Rdfa core 1.1, 2007.
- [2] Ben Adida, Mark Birbeck, Shane McCarron, and Steven Pemberton. Rdfa in xhtml: Syntax and processing. *Recommendation, W3C*, 7, 2008.
- [3] Alasdair J G Gray. Validata: An Online Tool For Testing RDF Data Conformance. <http://www.macs.hw.ac.uk/~ajg33/validata-swat4ls/>. Date Accessed: 07/02/2019.
- [4] Bioschemas.org. What is Bioschemas? <https://bioschemas.org>. Date Accessed: 3/11/2018.
- [5] Matthew Burgess and Natasha Noy. Building Google Dataset Search and FOustering an Open Data Ecosystem. <https://ai.googleblog.com/2018/09/building-google-dataset-search-and.html>. Date Accessed: 20/11/2018.
- [6] Chaals McCatchie Nevile and Dan Brickley. HTML Microdata W3C Working Draft. <https://www.w3.org/TR/microdata/>. Date Accessed: 27/10/18.
- [7] Common Crawl. Common Crawl. <http://commoncrawl.org/>. Date Accessed: 27/10/2018.
- [8] Google. Dataset. <https://developers.google.com/search/docs/data-types/dataset>. Date Accessed: 20/11/2018.
- [9] Google. Introduction to Structured Data. <https://developers.google.com/search/docs/guides/intro-structured-data>. Date Accessed: 15/10/18.
- [10] Alasdair JG Gray, Carole A Goble, and Rafael Jimenez. Bioschemas: From potato salad to protein annotation. In *International Semantic Web Conference (Posters, Demos & Industry Tracks)*, 2017.
- [11] R Guha, Dan Brickley, and Steve Macbeth. Schema.org: evolution of structured data on the web. *Communications of the ACM*, 59(2):44–51, 2016.

- [12] Ramanathan V Guha, Dan Brickley, and Steve MacBeth. Schema. org: Evolution of structured data on the web. *Queue*, 13(9):10, 2015.
- [13] Jacob Baungard Hansen, Andrew Beveridge, Roisin Farmer, Leif Gehrman, Alasdair JG Gray, Sunil Khutan, Tomas Robertson, and Johnny Val. Validata: An online tool for testing rdf data conformance. In *SWAT4LS*, pages 157–166, 2015.
- [14] Jeremy Dorn. JSON Editor. <https://github.com/json-editor/json-editor>. Date Accessed: 28/02/2019.
- [15] JSON Schema. Structuring a complex schema. <https://json-schema.org/understanding-json-schema/structuring.html>. Date Accessed: 09/04/2019.
- [16] Margaret Rouse. Go (Programming Language). <https://searchitoperations.techtarget.com/definition/Go-programming-language>. Date Accessed: 28/02/2019.
- [17] Shane McCarron, Ben Adida, M Birbeck, K Gregg, I Herman, and S Pemberton. Html+ rdfa 1.1, 2013.
- [18] Microformats. Accepted Limitations Of Microformats. <http://microformats.org/wiki/accepted-limitations-of-microformats>. Date Accessed: 1/11/2018.
- [19] Microformats.org. hCalender 1.0. <http://microformats.org/wiki/hcalendar>. Date Accessed: 27/10/2018.
- [20] Microformats.org. hcard 1.0. <http://microformats.org/wiki/hcard>. Date Accessed: 27/10/2018.
- [21] Microformats.org. Revision history of "Main Page". http://microformats.org/wiki/index.php?title=Main_Page&dir=prev&action=history. Date Accessed: 27/10/2018.
- [22] Microsoft. EdgeHTML Platform Status. <https://developer.microsoft.com/en-us/microsoft-edge/platform/status/filewriter/>. Date Accessed: 09/04/2019.
- [23] Erika Morphy. What Is Google Dataset Search? <https://www.cmswire.com/big-data/what-is-google-dataset-search/>. Date Accessed: 20/11/2018.
- [24] Natasha Noy. Making it easier to discover datasets. <https://www.blog.google/products/search/making-it-easier-discover-datasets/>. Date Accessed: 20/11/2018.

- [25] Schema.org. About Schema.org. <https://schema.org/docs/about.html>. Date Accessed: 15/10/2018.
- [26] Schema.org. Organization of Schemas. <https://schema.org/docs/schemas.html>. Date Accessed: 3/11/2018.
- [27] Schema.org. Person. <https://schema.org/Person>. Date Accessed: 27/02/2019.
- [28] Leslie Sikos. *Mastering structured data on the Semantic Web: From HTML5 microdata to linked open data*. Apress, 2015.
- [29] Maun Sporny, I Herman, and S Pemberton. Rdfa lite 1.1. 2015.
- [30] University of Mannheim. Web Data Commons. <http://webdatacommons.org/>. Date Accessed: 27/10/2018.
- [31] University of Mannheim. Web Data Commons - RDFa, Microdata, Embedded JSON-LD, and Microformats Data Sets - November 2014. <http://webdatacommons.org/structureddata/2014-12/stats/stats.html>. Date Accessed: 07/11/2018.
- [32] University of Mannheim. Web Data Commons - RDFa, Microdata, Embedded JSON-LD, and Microformats Data Sets - November 2017. <http://webdatacommons.org/structureddata/2017-12/stats/stats.html>. Date Accessed: 27/10/2018.
- [33] University of Mannheim. Web Data Commons - RDFa, Microdata, Embedded JSON-LD, and Microformats Data Sets - October 2016. <http://webdatacommons.org/structureddata/2016-10/stats/stats.html>. Date Accessed: 27/10/2018.
- [34] W3C. File API: Writer W3C Working Draft 17 April 2012. <https://www.w3.org/TR/2012/WD-file-writer-api-20120417/>. Date Accessed: 09/04/2019.
- [35] W3C. What does W3C do? <https://www.w3.org/Help/#activity>. Date Accessed: 07/02/2019.
- [36] William Kenny. Object Oriented Programming in Go. <https://www.ardanlabs.com/blog/2013/07/object-oriented-programming-in-go.html>. Date Accessed: 28/02/2019.

Appendix A

Prototype Usability Study

A.1 Questionnaire

Bioschemas Prototype Evaluation Questionnaire

1. How familiar are you with Schema.org?

1	2	3	4	5	6	7	8	9	10
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

2. How familiar are you with Bioschemas.org?

1	2	3	4	5	6	7	8	9	10
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

3. How easy was the interface to navigate and use?

1	2	3	4	5	6	7	8	9	10
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

4. How appealing and attractive is the interface?

1	2	3	4	5	6	7	8	9	10
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

5. Does the website contain enough information about the Bioschema profile to fill out the form?

☐ Yes

☐ No

6. If you answered No in the previous question, what other information could be added?

7. Did the form generate the expected result?

☐ Yes

☐ No

8. If you answered No in the previous question, could you please provide the example and what the expected result should have been.

9. Is there any extra features you would like to see in the final product.

A.2 Second Prototype Usability Results

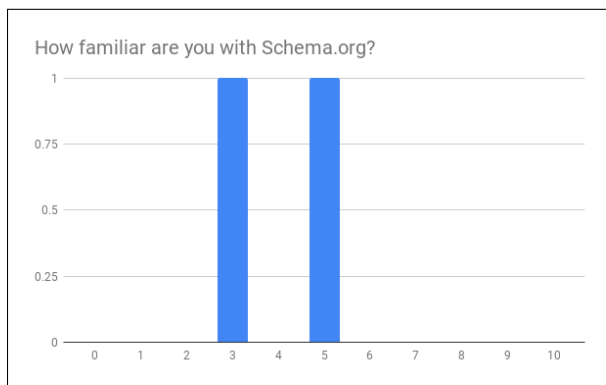


Figure A.1: Final Evaluation Task 3: Average

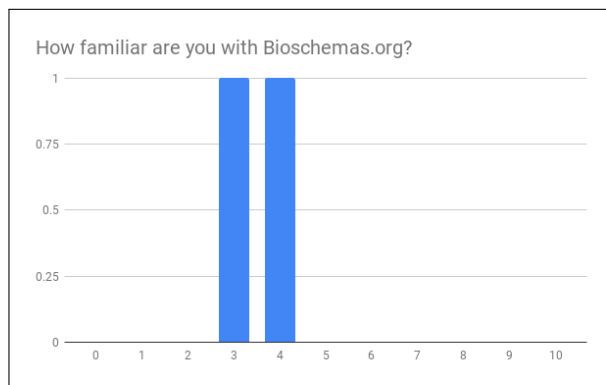


Figure A.2: Final Evaluation Task 4: Average

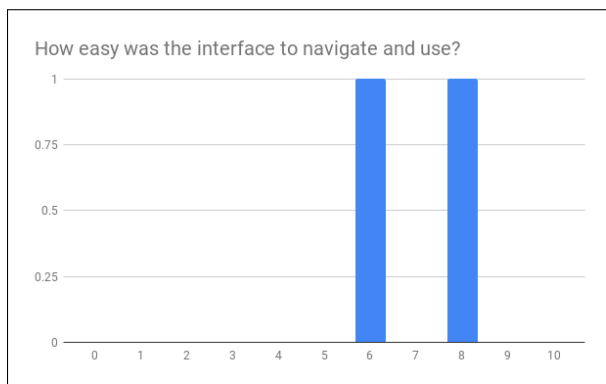


Figure A.3: Final Evaluation Task 3: Average

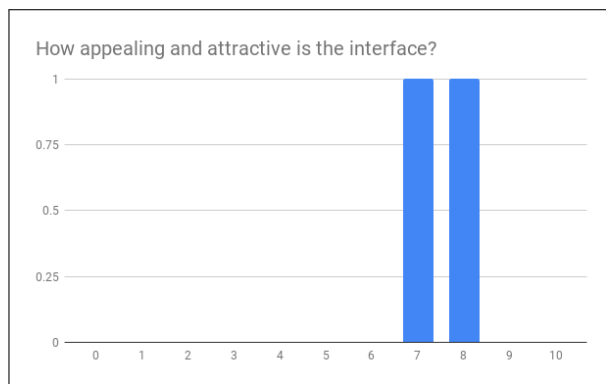


Figure A.4: Final Evaluation Task 4: Average

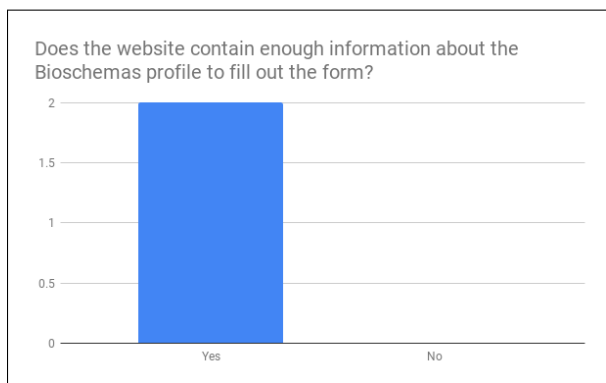


Figure A.5: Final Evaluation Task 3: Average

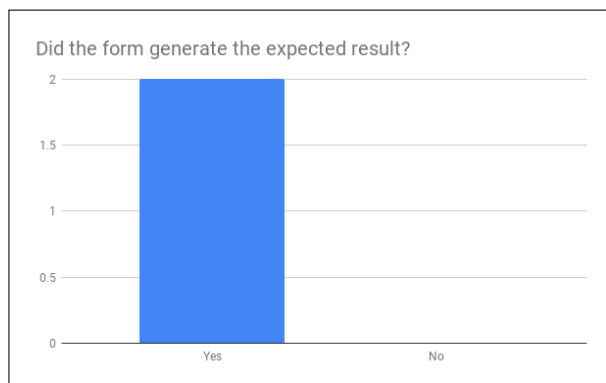


Figure A.6: Final Evaluation Task 4: Average

Appendix B

Final Usability Study

B.1 Consent Form

Consent Form for Usability Experiment

Consent to Act as a Subject in an Experimental Study

Principal Investigator: Sean Turnbull

Description: The purpose of this study is to test two applications, Bioschemas Profile Markup Generator and Kaizen. As the subject, you may be asked several questions on the usability of the applications before being asked about your own person and honest opinion. You may be tested on the the time taken to complete a few simple tasks that may include filling out the a form with provided example data. The ease of usability may also be tested using the time taken to fully complete a task given. There are minimal risks for you to participate in this study. No personal information will be taken. Your participation will not affect how well you do in your courses (if you are a student) or affect your relationship with the university in any way. You are free to decline to participate in this study. Should you decided to participate, you are free to end your participation at any time. Such a decision by you will not adversely affect or alter your status with the university in any way.

Voluntary Consent: I certify that I have read the preceding and that I understand its contents. Any questions I have pertaining to the research have been and will be answered. My signature below means that I have freely agreed to participate in this study, and that I agree to the publication of the results for scientific purposes so long as my identity is not revealed.

Date

Subject Signature

Inv. Initials

Investigator's Certification: I certify that I have explained to the above individual in this research study, have answered any questions that have been raised and have witnessed the above signature.

Date

Investigator Signature

B.2 Task Sheets

System A

System A Usability Test

The purpose of this test is to evaluate how easy the developed Bioschemas Profile Markup Generator web application is to use.

You will be timed for each task.

Please complete the tasks below.

Task 1: Generate a markup for the DataCatalog profile. Insert the following data:

Name:

OMA Orthology Database

Description:

OMA is a project that aims to identify orthologs among publicly available, complete genomes.

Keywords:

evolutionary relation, orthology prediction, paralogy

Provider:

Organization:

URL: <https://www.sib.swiss/>

Task 2: Download the result from Task 1 into a HTML File.

Task 3: Find the Controlled Vocabulary for the Recommended property “encodes” in the Gene Profile.

Task 4: Generate a markup for the Gene profile. Insert the following data:

Identifier:

Property Value:

URL: <http://identifiers.org/ensembl:ENSG00000142192>

Name:

APP

Encodes:

BioChemEntity:

Name: Protein P05067

Identifier: uniprotkb:P05067

URL: <http://identifiers.org/ensembl:ENSG00000142192>

System B

System B Usability Test

The purpose of this test is to evaluate how easy the developed Kaizen web application is to use.

You will be timed for each task.

Please complete the tasks below.

Task 1: Generate a markup for the DataCatalog profile. Insert the following data:

Name:

OMA Orthology Database

Description:

OMA is a project that aims to identify orthologs among publicly available, complete genomes.

Keywords:

evolutionary relation, orthology prediction, paralogy

Provider:

Organization:

URL: <https://www.sib.swiss/>

Task 2: Download the result from Task 1 into a HTML File.

Task 3: Find the Controlled Vocabulary for the Recommended property “encodes” in the Gene Profile.

B.3 Questionnaire

Usability Evaluation Questionnaire

1. The interface is easy to use and navigate.

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

Comment:

2. The interface is appealing and attractive.

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

Comment:

3. The website contains enough relevant information.

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

Comment:

4. I was able to find the data I needed quickly.

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

Comment:

5. Is there anything that you would change about the system?

6. Is there anything that you would like to add to the system?

7. Which system did you prefer?

☐ System A

☐ System B

B.4 Task Times

Task 1

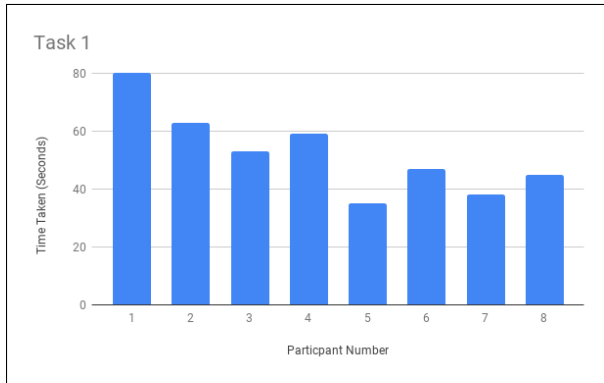


Figure B.1: Final Evaluation Task 1: System A

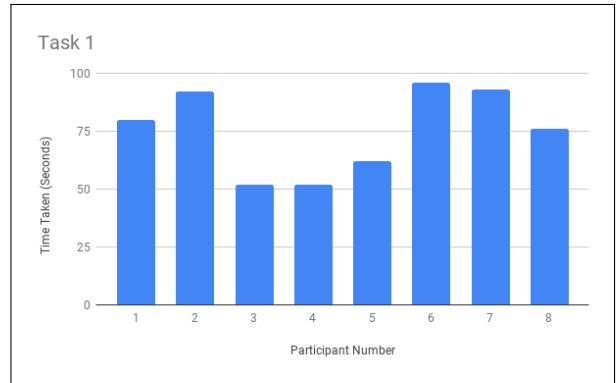


Figure B.2: Final Evaluation Task 1: System B

Task 2

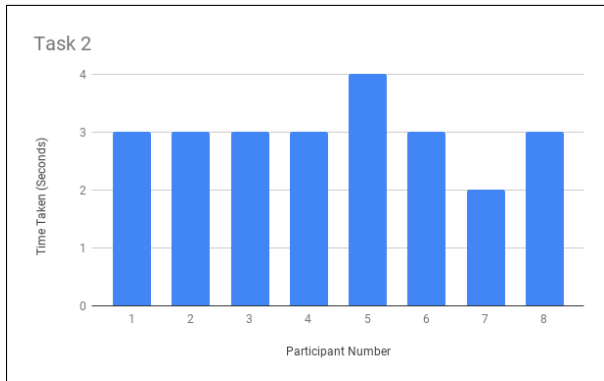


Figure B.3: Final Evaluation Task 2: System A

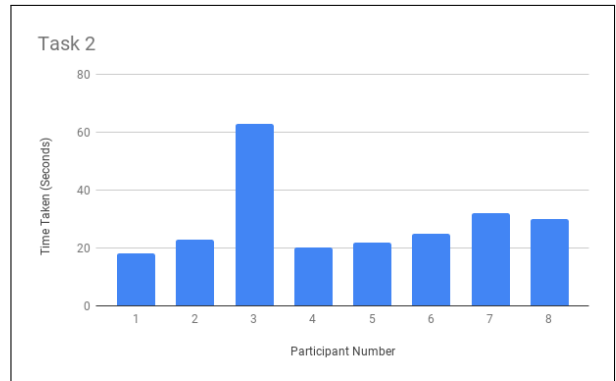


Figure B.4: Final Evaluation Task 2: System B

Task 3

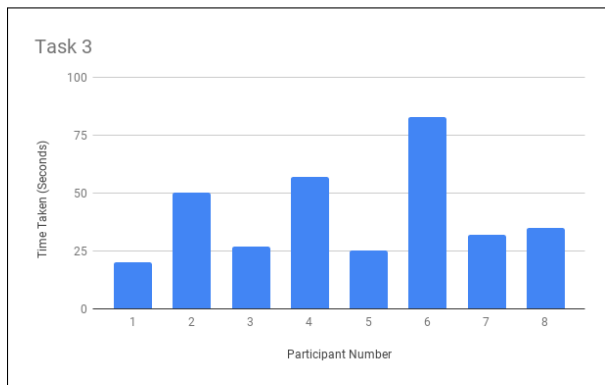


Figure B.5: Final Evaluation Task 3: System A

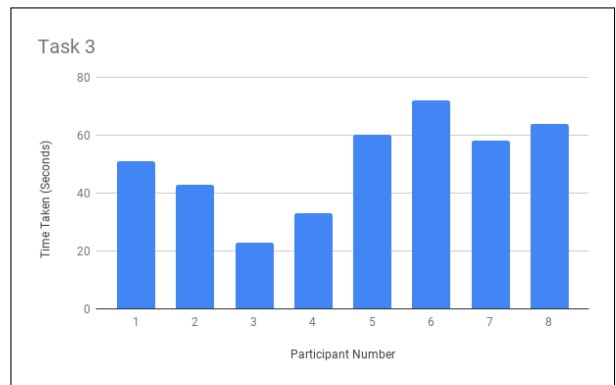


Figure B.6: Final Evaluation Task 3: System B

Task 4

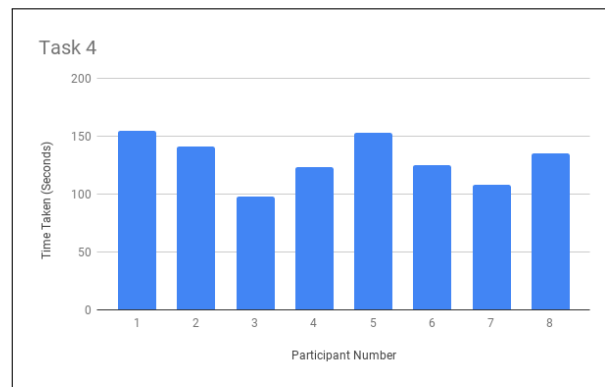


Figure B.7: Final Evaluation Task 4: System A

Task Averages

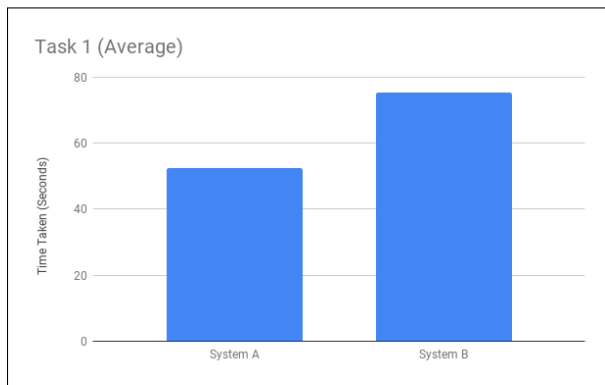


Figure B.8: Final Evaluation Task 1: Average

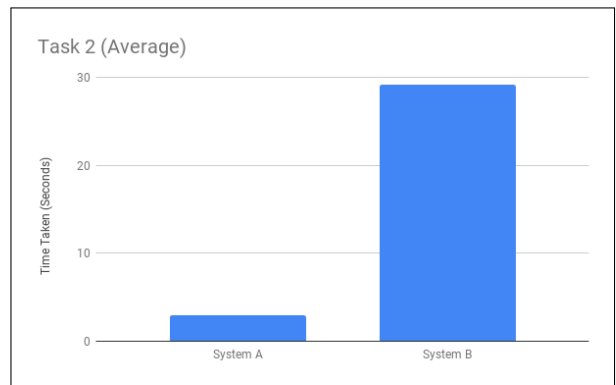


Figure B.9: Final Evaluation Task 2: Average

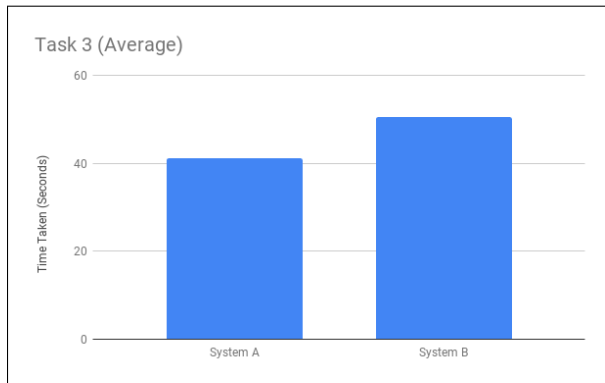


Figure B.10: Final Evaluation Task 3: Average

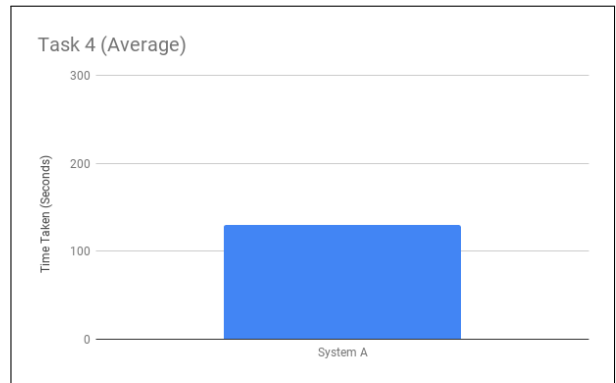


Figure B.11: Final Evaluation Task 4: Average

Appendix C

Files

C.1 Gene YAML

```
1 ---
2
3 name: Gene
4 url: Gene
5
6 status: release
7 spec_type: Profile
8 group: genes
9 use_cases_url: /useCases/Genes/
10 cross_walk_url: https://docs.google.com/spreadsheets/d/15yAvj5m-Ak_hbkSGrBx2fuhTfcQh-
   ↪ gY1mLT1jd0u5a0/edit#gid=1483018794
11 gh_tasks: https://github.com/BioSchemas/bioschemas/labels/type%3A%20Gene
12 live_deploy: ''
13
14 parent_type: Gene
15 hierarchy:
16 - Thing
17 - BioChemEntity
18 - Gene
19
20 spec_mapping_url: https://docs.google.com/spreadsheets/d/1WGP1VPElboWsKnASQwbp19wcvTg-
   ↪ _pi0CoIJ6Xe7rB8/edit#gid=1483018794
21
22 # spec_info content generated using GOWeb
23 # DO NOT MANUALLY EDIT THE CONTENT
24 spec_info:
25   title: Gene
26   subtitle: Bioschemas profile describing a Gene in Life Sciences
27   description: This Gene profile specification presents the markup for describing
28     a Gene.
29   version: 0.4
30   version_date: 20181110T092310
31   official_type: http://bioschemas.org/Gene
32   full_example: https://github.com/BioSchemas/Specifications/tree/master/Gene/
33 mapping:
34 - property: additionalProperty
35   expected_types:
36   - PropertyValue
37   description: |-
38     A property-value pair representing an additional characteristics of the entity, e.g.
39     ↪ a product feature or another characteristic for which there is no matching
40     ↪ property in schema.org.
41
42     Note: Publishers should be aware that applications designed to use specific schema.org
43     ↪ properties (e.g. http://schema.org/width, http://schema.org/color,
44     ↪ http://schema.org/gtin13, ...) will typically expect such data to be provided
45     ↪ using those properties, rather than using the generic property/value mechanism.
46 type: ""
```

```

42 type_url: ""
43 bsc_description: Whenever possible, please use a property coined in a third-party
44 well-know vocabulary. For instance, you can directly use
45   ↪ http://purl.obolibrary.org/obo/RO_0002327
46 as a property to express how a protein or gene enables some GO molecular function.
47 If you still want or need to use additionalProperty, please use (i) property name
48 to specify the name of the property, (ii) additionalType (if possible) to better
49 specify the nature of the property, and (iii) value to link to the object/range
50 of this property.
51 marginality: Optional
52 cardinality: MANY
53 controlled_vocab: ""
54 example: ""
55 - property: additionalType
56 expected_types:
57   - URL
58 description: An additional type for the item, typically used for adding more specific
59 types from external vocabularies in microdata syntax. This is a relationship between
60 something and a class that the thing is in. In RDFa syntax, it is better to use
61 the native RDFa syntax - the 'typeof' attribute - for multiple types. Schema.org
62 tools may have only weaker understanding of extra types, in particular those defined
63 externally.
64 type: ""
65 type_url: ""
66 bsc_description: |-
67   Any ontology term describing the gene concept. This is in addition to the official type
68   ↪ used in Bioschemas.
69
70   **Bioschemas Gene**: Official proposed term for the profile describing a gene:
71   ↪ [SO:gene](http://purl.obolibrary.org/obo/SO_0000704).
72 marginality: Optional
73 cardinality: MANY
74 controlled_vocab: ""
75 example: |-
76   {
77     "@type": ["Gene"],
78     "additionalType": "http://semanticscience.org/resource/SIO_010035"
79   }
80 - property: alternateName
81 expected_types:
82   - Text
83 description: An alias for the item.
84 type: ""
85 type_url: ""
86 bsc_description: ""
87 marginality: Optional
88 cardinality: MANY
89 controlled_vocab: ""
90 example: |-
91   {
92     "@type": ["Gene"],
93     "alternateName": "AD1"
94   }

```

```

92 - property: associatedDisease
93   expected_types:
94     - MedicalCondition
95     - URL
96   description: Disease associated to this BioChemEntity
97   type: bioschemas
98   type_url: http://bioschemas.org/associatedDisease
99   bsc_description: ""
100   marginality: Optional
101   cardinality: ""
102   controlled_vocab: ""
103   example: ""
104 - property: description
105   expected_types:
106     - Text
107   description: A description of the item.
108   type: ""
109   type_url: ""
110   bsc_description: ""
111   marginality: Recommended
112   cardinality: ONE
113   controlled_vocab: ""
114   example: |-
115     {
116       "@type": ["Gene"],
117       "description": "amyloid beta precursor"
118     }
119 - property: enablesMF
120   expected_types:
121     - CategoryCode
122     - PropertyValue
123   description: ""
124   type: external
125   type_url: ' http://purl.obolibrary.org/obo/RO_0002327'
126   bsc_description: "RO:0002327 (enables). GO molecular function enabled by the
127     ↪ gene/protein.
128     \nRecommended range: BioChemEntity or CategoryCode, ProteinAnnotation if evidence
129     should be included."
130   marginality: Optional
131   cardinality: MANY
132   controlled_vocab: Only the molecular function branch of [Gene Ontology
133     ↪ (GO)](http://www.geneontology.org)
134   example: |-
135     //Gene enabling a GO molecular function
136     {
137       "@type": ["Gene"],
138       "enablesMF": {
139         "@type": "CategoryCode",
140         "url": "http://purl.obolibrary.org/obo/GO_0000166",
141         "codeValue": "GO:0000166",
142         "name": "nucleotide binding"
143       }
144     }

```

```

143 - property: encodes
144   expected_types:
145     - BioChemEntity
146     - Protein
147     - URL
148   description: This property is used to link to gene products encoded (even indirectly)
149     from this gene such as RNA or proteins.
150   type: bioschemas
151   type_url: http://bioschemas.org/encodes
152   bsc_description: |-
153     For genes, this property is used to express in a generic way gene products encoded by
154     ↪ this gene. Two more specific properties SIO:010082 (is translated into) and
155     ↪ SIO:010080 (is transcribed into) should be used for (protein) translation and
156     ↪ (RNA) transcription respectively.
157     Note: bioschemas:encodes skos:closeMatch SIO:010078
158   marginality: Recommended
159   cardinality: MANY
160   controlled_vocab: SIO
161   example: "{\n  \"@type\": [\"Gene\"],\n  \"encodes\": {\n    \"@type\": [\"Protein\"],\n
162    \"identifier\": \"uniprotkb:P05067\", \n    \"url\":\n
163    ↪ \"https://www.uniprot.org/uniprot/P05067\"\n
164    }\n}"
165 - property: hasBioChemEntityPart
166   expected_types:
167     - BioChemEntity
168   description: ""
169   type: bioschemas
170   type_url: http://bioschemas.org/hasBioChemEntityPart
171   bsc_description: |+
172     For genes, it can be used to link to gene sequence annotations such as variants or so.
173   marginality: Optional
174   cardinality: MANY
175   controlled_vocab: ""
176   example: "//Contains a variant\n{\n  \"@type\": [\"Gene\"],\n  \"hasBioChemEntityPart\":\n
177   {\n    \"@type\": \"BioChemEntity\", \n    \"additionalType\":\n
178   ↪ \"http://semanticscience.org/resource/SIO_001381\", \n
179   \"identifier\": \"rs577882423\", \n    }, \n    \"url\":\n
180   ↪ \"https://www.ensembl.org/Homo_sapiens/Variation/\n
181   Summary?db=core;g=ENSG00000142192;r=21:25880550-26171128;vf=110312838\"\n
182   }\n}\n"
183 - property: hasCategoryCode
184   expected_types:
185     - CategoryCode
186   description: A Category code contained in this code set.
187   type: pending
188   type_url: http://pending.schema.org/hasCategoryCode
189   bsc_description: ""
190   marginality: Optional
191   cardinality: MANY
192   controlled_vocab: ""
193   example: ""
194 - property: hasRepresentation

```

```

190 expected_types:
191 - PropertyValue
192 - Text
193 - URL
194 description: A common representation such as a protein sequence or chemical structure
195 for this entity. For images use schema.org/image.
196 type: bioschemas
197 type_url: http://bioschemas.org/hasRepresentation
198 bsc_description: For genes, this property could be used, for instance, to register
199 a gene sequence as its representation. If you want to better define the nature
200 of the representation, use a PropertyValue as described in additionalProperty
201 or a third-party ontology predicate.
202 marginality: Recommended
203 cardinality: MANY
204 controlled_vocab: ""
205 example: |-
206 {
207     "@type": ["Gene"],
208     "hasRepresentation": "https://www.ncbi.nlm.nih.gov/
209     nuccore/NC_000021.9?report=fasta&from=25880550&to=26171128&strand=true"
210 }
211 - property: identifier
212 expected_types:
213 - PropertyValue
214 - Text
215 - URL
216 description: The identifier property represents any kind of identifier for any kind
217 of Thing, such as ISBNs, GTIN codes, UUIDs etc. Schema.org provides dedicated
218 properties for representing many of these, either as textual strings or as URL
219 (URI) links. See [background notes](http://schema.org/docs/datamodel.html#identifierBg)
220 for more details.
221 type: ""
222 type_url: ""
223 bsc_description: ""
224 marginality: Minimum
225 cardinality: ONE
226 controlled_vocab: ""
227 example: |-
228 //Using a text
229 {
230     "@type": ["Gene"],
231     "identifier": "ENSG00000142192"
232 }
233 //Using a URL directly (looks like a text)
234 {
235     "@type": ["Gene"],
236     "identifier": "http://identifiers.org/ensembl:ENSG00000142192"
237 }
238 //Using a URL modeled as schema:URL (equivalent to the previous one but more verbose)
239 {
240     "@type": ["Gene"],
241     "identifier": {
242         "@type": "URL",

```

```

243         "url": "http://identifiers.org/ensembl:ENSG00000142192"
244     }
245 }
246 - property: image
247   expected_types:
248   - ImageObject
249   - URL
250   description: An image of the item. This can be a URL or a fully described ImageObject.
251   type: ""
252   type_url: ""
253   bsc_description: ""
254   marginality: Recommended
255   cardinality: ONE
256   controlled_vocab: ""
257   example: |
258     //Using a URL directly (looks like a text)
259     {
260       "@type": ["Gene"],
261       "image": "https://en.wikipedia.org/wiki/Amyloid_precursor_protein#/
262       media/File:Ideogram_human_chromosome_21.svg"
263     }
264 - property: involvedInBP
265   expected_types:
266   - CategoryCode
267   - PropertyValue
268   description: ""
269   type: external
270   type_url: ' http://purl.obolibrary.org/obo/RO_0002331'
271   bsc_description: |-
272     RO:0002331 (is involved in). GO biological process this gene/protein is involved in.
273     Recommended range: BioChemEntity or CategoryCode, ProteinAnnotation if evidence should
274     ↪ be included.
275   marginality: Optional
276   cardinality: MANY
277   controlled_vocab: Only the biological process branch of [Gene Ontology
278     ↪ (GO)](http://www.geneontology.org)
279   example: |
280     //Gene involved in a GO biological process
281     {
282       "@type": ["Gene"],
283       "involvedInBP": {
284         "@type": "CategoryCode",
285         "url": "http://purl.obolibrary.org/obo/GO_0000278",
286         "codeValue": "GO:0000278",
287         "name": "mitotic cell cycle"
288       }
289     }
290 - property: isPartOfBioChemEntity
291   expected_types:
292   - BioChemEntity
293   description: |-
294     Indicates a BioChemEntity that is (in some sense) a part of this BioChemEntity.
295     Inverse property: hasBioChemEntity

```

```

294 type: bioschemas
295 type_url: http://bioschemas.org/isPartOfBioChemEntity
296 bsc_description: |-
297     **Bioschemas Gene**:
298     For genes, it is recommended to at least specify the DNA/chromosome containing this
299     ↪ gene and the taxon/organism associated to it. For taxon/organism, it is a good
300     ↪ practice to use categoryCode to point to a controlled vocabulary such as NCBI
301     ↪ taxon or UniProt Taxonomy.
302 marginality: Recommended
303 cardinality: MANY
304 controlled_vocab: For subcellular locations branch from GO, please use [Gene Ontology
305 (GO)](http://www.geneontology.org)
306 example: "/Is contained in a chromosome in positions X to Y\n{\n \@type\":
307     ↪ ["Gene\"],\n
308     ↪ \ \"isPartOfBioChemEntity\": {\n \@type\": \"BioChemEntity\", \n \"name\":
309     ↪ \"Chromosome 21\", \n } \n} \n//Is contained in an organism\n{\n \@type\": [\"Gene\"], \n
310     ↪ \ \"isPartOf\": {\n \@type\": \"BioChemEntity\", \n \"name\": \"Homo sapiens\", \n
311     ↪ \ \"alternateName\": \"Human\", \n \"codeCategory\": {\n \@type\":
312     ↪ \"CategoryCode\", \n
313     ↪ \ \"codeValue\": \"9606\", \n \"url\":
314     ↪ \"http://purl.bioontology.org/ontology/NCBITAXON/9606\", \n
315     ↪ \ \"sameAs\": \"http://purl.uniprot.org/taxonomy/9606\", \n \"inCodeSet\":
316     ↪ {\n \@type\": \"CategoryCodeSet\", \n \"name\": \"NCBI taxon\" \n
317     ↪ \ } \n } \n } \n}
318 - property: isTranscribedInto
319 expected_types:
320 - BioChemEntity
321 description: ""
322 type: external
323 type_url: http://semanticscience.org/resource/SIO_010080
324 bsc_description: SIO:010080 (is transcribed into). For genes, this property is used
325 to link to gene products transcribed from this gene such as RNA.
326 marginality: Optional
327 cardinality: MANY
328 controlled_vocab: SIO
329 example: ""
330 - property: isVariantOf
331 expected_types:
332 - BioChemEntity
333 - bioschemas:Gene
334 description: ""
335 type: external
336 type_url: http://semanticscience.org/resource/SIO_000272
337 bsc_description: 'SIO: 000272 (is variant of). Use this property to express when
338 a gene is a variant of any other gene.'
339 marginality: Optional
340 cardinality: MANY
341 controlled_vocab: SIO
342 example: ""
343 - property: mainEntityOfPage
344 expected_types:
345 - CreativeWork
346 - URL

```



```

description: |-
    Indicates a page (or other CreativeWork) for which this thing is the main entity being
    ↪ described. See [background
    ↪ notes](http://schema.org/docs/datamodel.html#mainEntityBackground) for details.
Inverse property: mainEntity.
type: ""
type_url: ""
bsc_description: Link via DataRecord to the main DataRecord representing this entity
    in a dataset. It is usually preferred to use mainEntity from a DataRecord to point
    to its corresponding entity.
marginality: Optional
cardinality: ONE
controlled_vocab: ""
example: |-
    //Preferred way from DataRecord
    {
        "@type": "DataRecord",
        "@id": "http://rdf.ebi.ac.uk/resource/ensembl/ENSG00000142192"
        "mainEntity": {
            "@type": ["Gene"],
            "name": "Name and any other property for this Gene entity"
        }
    }

    //Also possible from Gene
    {
        "@type": ["Gene"],
        "mainEntityOfPage": { "@id": "http://rdf.ebi.ac.uk/resource/ensembl/ENSG00000142192"
            ↪ }
    }

- property: name
  expected_types:
  - Text
  description: The name of the item.
  type: ""
  type_url: ""
  bsc_description: ""
  marginality: Minimum
  cardinality: ONE
  controlled_vocab: ""
  example: |-
      {
          "@type": ["Gene"],
          "name": "APP"
      }

- property: sameAs
  expected_types:
  - URL
  description: URL of a reference Web page that unambiguously indicates the item's
    identity. E.g. the URL of the item's Wikipedia page, Wikidata entry, or official
    website.
  type: ""

```

```

391 type_url: ""
392 bsc_description: ""
393 marginality: Optional
394 cardinality: MANY
395 controlled_vocab: ""
396 example: |-
397     {
398         "@type": ["Gene"],
399         "sameAs": "https://www.wikidata.org/wiki/Q14865870"
400     }
401 - property: taxonomicRange
402   expected_types:
403   - Taxon
404   - Text
405   - URL
406   description: The taxonomic grouping of the organism that expresses, encodes, or
407     in someway related to the BioChemEntity.
408   type: bioschemas
409   type_url: taxonomicRange
410   bsc_description: ""
411   marginality: Optional
412   cardinality: ""
413   controlled_vocab: ""
414   example: ""
415 - property: url
416   expected_types:
417   - URL
418   description: URL of the item.
419   type: ""
420   type_url: ""
421   bsc_description: Link to the official webpage associated to this entity.
422   marginality: Recommended
423   cardinality: ONE
424   controlled_vocab: ""
425   example: |-
426       {
427         "@type": ["Gene"],
428         "url": "https://www.ensembl.org/Homo_sapiens/Gene/
429           Summary?g=ENSG00000142192;r=21:25880550-26171128"
430       }
431
432 ---

```

C.2 Gene Python Object

```
1 {
2   "name": "Gene",
3   "url": "Gene",
4   "status": "release",
5   "spec_type": "Profile",
6   "group": "genes",
7   "use_cases_url": "/useCases/Genes/",
8   "cross_walk_url": "https://docs.google.com/spreadsheets/d/15yAvj5m-
   ↪ Ak_hbkSGrBx2fuhTfcQh-gY1mLT1jdOu5a0/edit#gid=1483018794",
9   "gh_tasks": "https://github.com/BioSchemas/bioschemas/labels/type%3A%20Gene",
10  "live_deploy": "",
11  "parent_type": "Gene",
12  "hierarchy": [
13    "Thing",
14    "BioChemEntity",
15    "Gene"
16  ],
17  "spec_mapping_url":
   ↪ "https://docs.google.com/spreadsheets/d/1WGP1VPElbowSKnASQwbp19wcvTg-
   ↪ _piOCoiJ6Xe7rB8/edit#gid=1483018794",
18  "spec_info": {
19    "title": "Gene",
20    "subtitle": "Bioschemas profile describing a Gene in Life Sciences",
21    "description": "This Gene profile specification presents the markup for describing a
   ↪ Gene.",
22    "version": 0.4,
23    "version_date": "20181110T092310",
24    "official_type": "http://bioschemas.org/Gene",
25    "full_example": "https://github.com/BioSchemas/Specifications/tree/master/Gene/"
26  },
27  "mapping": [
28    {
29      "property": "additionalProperty",
30      "expected_types": [
31        "PropertyValue"
32      ],
33      "description": "A property-value pair representing an additional characteristics of
   ↪ the entity, e.g. a product feature or another characteristic for which there
   ↪ is no matching property in schema.org.\n\nNote: Publishers should be aware that
   ↪ applications designed to use specific schema.org properties (e.g.
   ↪ http://schema.org/width, http://schema.org/color, http://schema.org/gtin13,
   ↪ ...) will typically expect such data to be provided using those properties,
   ↪ rather than using the generic property/value mechanism.",
34      "type": "",
35      "type_url": "",
36      "bsc_description": "Whenever possible, please use a property coined in a
   ↪ third-party well-know vocabulary. For instance, you can directly use
   ↪ http://purl.obolibrary.org/obo/RO_0002327 as a property to express how a protein
   ↪ or gene enables some GO molecular function. If you still want or need to use
   ↪ additionalProperty, please use (i) property name to specify the name of the
   ↪ property, (ii) additionalType (if possible) to better specify the nature of the
   ↪ property, and (iii) value to link to the object/range of this property.",
```

```

37     "marginality": "Optional",
38     "cardinality": "MANY",
39     "controlled_vocab": "",
40     "example": ""
41 },
42 {
43     "property": "additionalType",
44     "expected_types": [
45         "URL"
46     ],
47     "description": "An additional type for the item, typically used for adding more
    ↳ specific types from external vocabularies in microdata syntax. This is a
    ↳ relationship between something and a class that the thing is in. In RDFa syntax,
    ↳ it is better to use the native RDFa syntax - the 'typeof' attribute - for
    ↳ multiple types. Schema.org tools may have only weaker understanding of extra
    ↳ types, in particular those defined externally.",
48     "type": "",
49     "type_url": "",
50     "bsc_description": "Any ontology term describing the gene concept. This is in
    ↳ addition to the official type used in Bioschemas.\n\n**Bioschemas Gene**:\n
    ↳ Official proposed term for the profile describing a gene:\n
    ↳ [SO:gene](http://purl.obolibrary.org/obo/SO_0000704).",
51     "marginality": "Optional",
52     "cardinality": "MANY",
53     "controlled_vocab": "",
54     "example": "{\n    \@type\: [\"Gene\"],\n    \@additionalType\:
    ↳ \"http://semanticscience.org/resource/SIO_010035\"\n}"
55 },
56 {
57     "property": "alternateName",
58     "expected_types": [
59         "Text"
60     ],
61     "description": "An alias for the item.",
62     "type": "",
63     "type_url": "",
64     "bsc_description": "",
65     "marginality": "Optional",
66     "cardinality": "MANY",
67     "controlled_vocab": "",
68     "example": "{\n    \@type\: [\"Gene\"],\n    \@alternateName\: \"AD1\"\n}"
69 },
70 {
71     "property": "associatedDisease",
72     "expected_types": [
73         "MedicalCondition",
74         "URL"
75     ],
76     "description": "Disease associated to this BioChemEntity",
77     "type": "bioschemas",
78     "type_url": "http://bioschemas.org/associatedDisease",
79     "bsc_description": "",
80     "marginality": "Optional",
81     "cardinality": "",
82     "controlled_vocab": "",
83     "example": ""
84 },

```

```

85 {
86   "property": "description",
87   "expected_types": [
88     "Text"
89   ],
90   "description": "A description of the item.",
91   "type": "",
92   "type_url": "",
93   "bsc_description": "",
94   "marginality": "Recommended",
95   "cardinality": "ONE",
96   "controlled_vocab": "",
97   "example": "{\n    \"@type\": [\"Gene\"],\n    \"description\": \"amyloid beta\n    ↳ precursor\"\n}"
98 },
99 {
100   "property": "enablesMF",
101   "expected_types": [
102     "CategoryCode",
103     "PropertyValue"
104   ],
105   "description": "",
106   "type": "external",
107   "type_url": " http://purl.obolibrary.org/obo/RO_0002327",
108   "bsc_description": "RO:0002327 (enables). GO molecular function enabled by the\n    ↳ gene/protein. \nRecommended range: BioChemEntity or CategoryCode,\n    ↳ ProteinAnnotation if evidence should be included.",
109   "marginality": "Optional",
110   "cardinality": "MANY",
111   "controlled_vocab": "Only the molecular function branch of [Gene Ontology\n    ↳ (GO)](http://www.geneontology.org)",
112   "example": "//Gene enabling a GO molecular function\n{\n  \"@type\": [\"Gene\"],\n  ↳ \"enablesMF\": {\n    \"@type\": \"CategoryCode\", \n    \"url\":\n    ↳ \"http://purl.obolibrary.org/obo/GO_0000166\", \n    \"codeValue\":\n    ↳ \"GO:0000166\", \n    \"name\": \"nucleotide binding\"\n  }\n}"
113 },
114 {
115   "property": "encodes",
116   "expected_types": [
117     "BioChemEntity",
118     "Protein",
119     "URL"
120   ],
121   "description": "This property is used to link to gene products encoded (even\n    ↳ indirectly) from this gene such as RNA or proteins.",
122   "type": "bioschemas",
123   "type_url": "http://bioschemas.org/encodes",
124   "bsc_description": "For genes, this property is used to express in a generic way\n    ↳ gene products encoded by this gene. Two more specific properties SIO:010082 (is\n    ↳ translated into) and SIO:010080 (is transcribed into) should be used for\n    ↳ (protein) translation and (RNA) transcription respectively.\nNote:\n    ↳ bioschemas:encodes skos:closeMatch SIO:010078",
125   "marginality": "Recommended",
126   "cardinality": "MANY",
127   "controlled_vocab": "SIO",
128   "example": "{\n  \"@type\": [\"Gene\"], \n  \"encodes\": {\n    \"@type\":\n    ↳ [\"Protein\"], \n    \"identifier\": \"uniprotkb:P05067\", \n    \"url\":\n    ↳ \"https://www.uniprot.org/uniprot/P05067\"\n  }\n}"

```

```

129 },
130 {
131     "property": "hasBioChemEntityPart",
132     "expected_types": [
133         "BioChemEntity"
134     ],
135     "description": "",
136     "type": "bioschemas",
137     "type_url": "http://bioschemas.org/hasBioChemEntityPart",
138     "bsc_description": "For genes, it can be used to link to gene sequence annotations
        ↳ such as variants or so.\n\n",
139     "marginality": "Optional",
140     "cardinality": "MANY",
141     "controlled_vocab": "",
142     "example": "//Contains a variant\n{\n  \"@type\": [\"Gene\"],\n
        ↳ \"hasBioChemEntityPart\": { \n    \"@type\": \"BioChemEntity\", \n
        ↳ \"additionalType\": \"http://semanticscience.org/resource/SIO_001381\", \n
        ↳ \"identifier\": \"rs577882423\", \n    }, \n    \"url\":
        ↳ \"https://www.ensembl.org/Homo_sapiens/Variation/
143     Summary?db=core;g=ENSG00000142192;r=21:25880550-26171128;vf=110312838\" \n  } \n} \n"
144 },
145 {
146     "property": "hasCategoryCode",
147     "expected_types": [
148         "CategoryCode"
149     ],
150     "description": "A Category code contained in this code set.",
151     "type": "pending",
152     "type_url": "http://pending.schema.org/hasCategoryCode",
153     "bsc_description": "",
154     "marginality": "Optional",
155     "cardinality": "MANY",
156     "controlled_vocab": "",
157     "example": ""
158 },
159 {
160     "property": "hasRepresentation",
161     "expected_types": [
162         "PropertyValue",
163         "Text",
164         "URL"
165     ],
166     "description": "A common representation such as a protein sequence or chemical
        ↳ structure for this entity. For images use schema.org/image.",
167     "type": "bioschemas",
168     "type_url": "http://bioschemas.org/hasRepresentation",
169     "bsc_description": "For genes, this property could be used, for instance, to
        ↳ register a gene sequence as its representation. If you want to better define the
        ↳ nature of the representation, use a PropertyValue as described in
        ↳ additionalProperty or a third-party ontology predicate.",
170     "marginality": "Recommended",
171     "cardinality": "MANY",
172     "controlled_vocab": "",
173     "example": "{\n  \"@type\": [\"Gene\"], \n  \"hasRepresentation\":
        ↳ \"https://www.ncbi.nlm.nih.gov/nuccore/NC_000021.9?
174     report=fasta&from=25880550&to=26171128&strand=true\" \n}"
175 },

```

```

{
  "property": "identifier",
  "expected_types": [
    "PropertyValue",
    "Text",
    "URL"
  ],
  "description": "The identifier property represents any kind of identifier for any
    ↪ kind of Thing, such as ISBNs, GTIN codes, UUIDs etc. Schema.org provides
    ↪ dedicated properties for representing many of these, either as textual strings
    ↪ or as URL (URI) links. See [background
    ↪ notes](http://schema.org/docs/datamodel.html#identifierBg) for more details.",
  "type": "",
  "type_url": "",
  "bsc_description": "",
  "marginality": "Minimum",
  "cardinality": "ONE",
  "controlled_vocab": "",
  "example": "//Using a text\n{\n  \@type\": [\"Gene\"],\n  \"identifier\":
    ↪ \"ENSG00000142192\"\n}\n//Using a URL directly (looks like a text)\n{\n
    ↪ \@type\": [\"Gene\"],\n  \"identifier\":
    ↪ \"http://identifiers.org/ensembl:ENSG00000142192\"\n}\n//Using a URL modeled as
    ↪ schema:URL (equivalent to the previous one but more verbose)\n{\n  \@type\":
    ↪ [\"Gene\"],\n  \"identifier\": {\n    \@type\": \"URL\", \n    \"url\":
    ↪ \"http://identifiers.org/ensembl:ENSG00000142192\"\n  }\n}
},
{
  "property": "image",
  "expected_types": [
    "ImageObject",
    "URL"
  ],
  "description": "An image of the item. This can be a URL or a fully described
    ↪ ImageObject.",
  "type": "",
  "type_url": "",
  "bsc_description": "",
  "marginality": "Recommended",
  "cardinality": "ONE",
  "controlled_vocab": "",
  "example": "//Using a URL directly (looks like a text)\n{\n  \@type\": [\"Gene\"],\n
    ↪ \"image\": \"https://en.wikipedia.org/wiki/Amyloid_precursor_protein#/
    ↪ media/File:Ideogram_human_chromosome_21.svg\"\n}\n}
},
{
  "property": "involvedInBP",
  "expected_types": [
    "CategoryCode",
    "PropertyValue"
  ],
  "description": "",
  "type": "external",
  "type_url": "http://purl.obolibrary.org/obo/RO_0002331",
  "bsc_description": "RO:0002331 (is involved in). GO biological process this
    ↪ gene/protein is involved in.\nRecommended range: BioChemEntity or CategoryCode,
    ↪ ProteinAnnotation if evidence should be included.",

```

```

219     "marginality": "Optional",
220     "cardinality": "MANY",
221     "controlled_vocab": "Only the biological process branch of [Gene Ontology
    ↪ (GO)](http://www.geneontology.org)",
222     "example": "//Gene involved in a GO biological process\n{\n  \@type\:
    ↪ [\"Gene\"],\n  \"involvedInBP\": {\n    \@type\: \"CategoryCode\", \n
    ↪ \"url\": \"http://purl.obolibrary.org/obo/GO_0000278\", \n    \"codeValue\":
    ↪ \"GO:0000278\", \n    \"name\": \"mitotic cell cycle\"\n  }\n}
223   },
224   {
225     "property": "isPartOfBioChemEntity",
226     "expected_types": [
227       "BioChemEntity"
228     ],
229     "description": "Indicates a BioChemEntity that is (in some sense) a part of this
    ↪ BioChemEntity.\nInverse property: hasBioChemEntity",
230     "type": "bioschemas",
231     "type_url": "http://bioschemas.org/isPartOfBioChemEntity",
232     "bsc_description": "***Bioschemas Gene**:\nFor genes, it is recommended to at least
    ↪ specify the DNA/chromosome containing this gene and the taxon/organism
    ↪ associated to it. For taxon/organism, it is a good practice to use categoryCode
    ↪ to point to a controlled vocabulary such as NCBI taxon or UniProt Taxonomy.",
233     "marginality": "Recommended",
234     "cardinality": "MANY",
235     "controlled_vocab": "For subcellular locations branch from GO, please use [Gene
    ↪ Ontology (GO)](http://www.geneontology.org)",
236
237
238
239     "example": "//Is contained in a chromosome in positions X to Y\n{\n  \@type\:
    ↪ [\"Gene\"],\n  \"isPartOfBioChemEntity\": {\n    \@type\: \"BioChemEntity\", \n
    ↪ \"name\": \"Chromosome 21\", \n  }\n}\n//Is contained in an organism\n{\n
    ↪ \@type\: [\"Gene\"],\n  \"isPartOf\": {\n    \@type\: \"BioChemEntity\", \n
    ↪ \"name\": \"Homo sapiens\", \n    \"alternateName\": \"Human\", \n
    ↪ \"codeCategory\": {\n    \@type\: \"CategoryCode\", \n    \"codeValue\":
    ↪ \"9606\", \n    \"url\":
    ↪ \"http://purl.bioontology.org/ontology/NCBITAXON/9606\", \n    \"sameAs\":
    ↪ \"http://purl.uniprot.org/taxonomy/9606\", \n    \"inCodeSet\": {\n
    ↪ \@type\: \"CategoryCodeSet\", \n    \"name\": \"NCBI taxon\" \n  } \n
    ↪ }\n  }\n}
240   },
241   {
242     "property": "isTranscribedInto",
243     "expected_types": [
244       "BioChemEntity"
245     ],
246     "description": "",
247     "type": "external",
248     "type_url": "http://semanticscience.org/resource/SIO_010080",
249     "bsc_description": "SIO:010080 (is transcribed into). For genes, this property is
    ↪ used to link to gene products transcribed from this gene such as RNA.",
250     "marginality": "Optional",
251     "cardinality": "MANY",
252     "controlled_vocab": "SIO",
253     "example": ""
254   },
255   {

```



```

256     "property": "isVariantOf",
257     "expected_types": [
258         "BioChemEntity",
259         "bioschemas:Gene"
260     ],
261     "description": "",
262     "type": "external",
263     "type_url": "http://semanticscience.org/resource/SIO_000272",
264     "bsc_description": "SIO: 000272 (is variant of). Use this property to express when
    ↪ a gene is a variant of any other gene.",
265     "marginality": "Optional",
266     "cardinality": "MANY",
267     "controlled_vocab": "SIO",
268     "example": ""
269 },
270 {
271     "property": "mainEntityOfPage",
272     "expected_types": [
273         "CreativeWork",
274         "URL"
275     ],
276     "description": "Indicates a page (or other CreativeWork) for which this thing is
    ↪ the main entity being described. See [background
    ↪ notes](http://schema.org/docs/datamodel.html#mainEntityBackground) for
    ↪ details.\nInverse property: mainEntity.",
277     "type": "",
278     "type_url": "",
279     "bsc_description": "Link via DataRecord to the main DataRecord representing this
    ↪ entity in a dataset. It is usually preferred to use mainEntity from a DataRecord
    ↪ to point to its corresponding entity.",
280     "marginality": "Optional",
281     "cardinality": "ONE",
282     "controlled_vocab": "",
283     "example": "//Preferred way from DataRecord\n{\n\n  \@type\: \"DataRecord\", \n
    ↪ \@id\: \"http://rdf.ebi.ac.uk/resource/ensembl/ENSG00000142192\" \n
    ↪ \"mainEntity\": {\n    \@type\: [\"Gene\"], \n    \"name\: \"Name and any
    ↪ other property for this Gene entity\" \n  } \n} \n\n//Also possible from Gene\n{\n
    ↪ \@type\: [\"Gene\"], \n  \"mainEntityOfPage\: { \@id\:
    ↪ \"http://rdf.ebi.ac.uk/resource/ensembl/ENSG00000142192\" } \n}"
284 },
285 {
286     "property": "name",
287     "expected_types": [
288         "Text"
289     ],
290     "description": "The name of the item.",
291     "type": "",
292     "type_url": "",
293     "bsc_description": "",
294     "marginality": "Minimum",
295     "cardinality": "ONE",
296     "controlled_vocab": "",
297     "example": "{\n  \@type\: [\"Gene\"], \n  \"name\: \"APP\" \n}"
298 },
299 {
300     "property": "sameAs",
301     "expected_types": [

```

```

302     "URL"
303 ],
304     "description": "URL of a reference Web page that unambiguously indicates the item's
    ↪ identity. E.g. the URL of the item's Wikipedia page, Wikidata entry, or official
    ↪ website.",
305     "type": "",
306     "type_url": "",
307     "bsc_description": "",
308     "marginality": "Optional",
309     "cardinality": "MANY",
310     "controlled_vocab": "",
311     "example": "{\n  \@type\: [\"Gene\"],\n  \\"sameAs\:\"
    ↪  \\"https://www.wikidata.org/wiki/Q14865870\""}"
312 },
313 {
314     "property": "taxonomicRange",
315     "expected_types": [
316         "Taxon",
317         "Text",
318         "URL"
319     ],
320     "description": "The taxonomic grouping of the organism that expresses, encodes, or
    ↪ in someway related to the BioChemEntity.",
321     "type": "bioschemas",
322     "type_url": "taxonomicRange",
323     "bsc_description": "",
324     "marginality": "Optional",
325     "cardinality": "",
326     "controlled_vocab": "",
327     "example": ""
328 },
329 {
330     "property": "url",
331     "expected_types": [
332         "URL"
333     ],
334     "description": "URL of the item.",
335     "type": "",
336     "type_url": "",
337     "bsc_description": "Link to the official webpage associated to this entity.",
338     "marginality": "Recommended",
339     "cardinality": "ONE",
340     "controlled_vocab": "",
341     "example": "{\n  \@type\: [\"Gene\"],\n  \\"url\:\"
    ↪  \\"https://www.ensembl.org/Homo_sapiens/Gene/
342     Summary?g=ENSG00000142192;r=21:25880550-26171128\""}"
343 }
344 ]
345 }

```

C.3 BioChemEntity JSON-LD

```
1 {
2   "@context": {
3     "dct": "http://purl.org/dc/terms/",
4     "owl": "http://www.w3.org/2002/07/owl#",
5     "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
6     "rdfa": "http://www.w3.org/ns/rdfa#",
7     "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
8     "schema": "http://schema.org/",
9     "xsd": "http://www.w3.org/2001/XMLSchema#"
10  },
11  "@graph": [
12    {
13      "@id": "schema:ProteinStructure",
14      "rdfs:subClassOf": {
15        "@id": "schema:BioChemEntity"
16      }
17    },
18    {
19      "@id": "schema:sampleUsed",
20      "schema:rangeIncludes": {
21        "@id": "schema:BioChemEntity"
22      }
23    },
24    {
25      "@id": "schema:url",
26      "@type": "rdf:Property",
27      "rdfs:comment": "URL of the item.",
28      "rdfs:label": "url",
29      "schema:domainIncludes": {
30        "@id": "schema:Thing"
31      },
32      "schema:rangeIncludes": {
33        "@id": "schema:URL"
34      },
35      "schema:sameAs": {
36        "@id": "https://schema.org/url"
37      }
38    },
39    {
40      "@id": "schema:hasHomolog",
41      "schema:rangeIncludes": {
42        "@id": "schema:BioChemEntity"
43      }
44    },
45    {
46      "@id": "schema:RNA",
47      "rdfs:subClassOf": {
48        "@id": "schema:BioChemEntity"
```

```

49     }
50 },
51 {
52     "@id": "schema:expressedIn",
53     "schema:rangeIncludes": {
54         "@id": "schema:BioChemEntity"
55     }
56 },
57 {
58     "@id": "schema:SequenceRange",
59     "rdfs:subClassOf": {
60         "@id": "schema:BioChemEntity"
61     }
62 },
63 {
64     "@id": "schema:reagent",
65     "schema:rangeIncludes": {
66         "@id": "schema:BioChemEntity"
67     }
68 },
69 {
70     "@id": "schema:isEncodedByBioChemEntity",
71     "@type": "rdf:Property",
72     "dct:source": {
73         "@id": "http://www.bioschemas.org/Gene"
74     },
75     "rdfs:comment": "Another BioChemEntity encoding by this one. Inverse property: <a
76     ↪ href=\"http://schema.org/encodesBioChemEntity\">encodesBioChemEntity</a>.",
77     "rdfs:label": "isEncodedByBioChemEntity",
78     "schema:domainIncludes": {
79         "@id": "schema:BioChemEntity"
80     },
81     "schema:inverseOf": {
82         "@id": "schema:encodesBioChemEntity"
83     },
84     "schema:isPartOf": {
85         "@id": "http://bio.schema.org"
86     },
87     "schema:rangeIncludes": [
88         {
89             "@id": "schema:DNA"
90         },
91         {
92             "@id": "schema:RNA"
93         },
94         {
95             "@id": "schema:Gene"
96         }
97     ],
98     "schema:sameAs": {
99         "@id": "https://schema.org/isEncodedByBioChemEntity"
100     }
101 },

```

```

101 {
102   "@id": "schema:hasCoenzyme",
103   "schema:rangeIncludes": {
104     "@id": "schema:BioChemEntity"
105   }
106 },
107 {
108   "@id": "schema:disambiguatingDescription",
109   "@type": "rdf:Property",
110   "rdfs:comment": "A sub property of description. A short description of the item
    ↪ used to disambiguate from other, similar items. Information from other
    ↪ properties (in particular, name) may be necessary for the description to be
    ↪ useful for disambiguation.",
111   "rdfs:label": "disambiguatingDescription",
112   "rdfs:subPropertyOf": {
113     "@id": "schema:description"
114   },
115   "schema:domainIncludes": {
116     "@id": "schema:Thing"
117   },
118   "schema:rangeIncludes": {
119     "@id": "schema:Text"
120   },
121   "schema:sameAs": {
122     "@id": "https://schema.org/disambiguatingDescription"
123   }
124 },
125 {
126   "@id": "schema:SequenceAnnotation",
127   "rdfs:subClassOf": {
128     "@id": "schema:BioChemEntity"
129   }
130 },
131 {
132   "@id": "schema:Gene",
133   "rdfs:subClassOf": {
134     "@id": "schema:BioChemEntity"
135   }
136 },
137 {
138   "@id": "schema:associatedDisease",
139   "@type": "rdf:Property",
140   "dct:source": {
141     "@id": "http://www.bioschemas.org/BioChemEntity"
142   },
143   "rdfs:comment": "Disease associated to this BioChemEntity. Such disease can be a
    ↪ MedicalCondition or a URL. If you want to add an evidence supporting the
    ↪ association, please use PropertyValue.",
144   "rdfs:label": "associatedDisease",
145   "schema:domainIncludes": {
146     "@id": "schema:BioChemEntity"
147   },
148   "schema:isPartOf": {

```

```

149     "@id": "http://bio.schema.org"
150 },
151 "schema:rangeIncludes": [
152     {
153         "@id": "schema:MedicalCondition"
154     },
155     {
156         "@id": "schema:URL"
157     },
158     {
159         "@id": "schema:PropertyValue"
160     }
161 ],
162 "schema:sameAs": {
163     "@id": "https://schema.org/associatedDisease"
164 }
165 },
166 {
167     "@id": "schema:hasOrtholog",
168     "schema:rangeIncludes": {
169         "@id": "schema:BioChemEntity"
170     }
171 },
172 {
173     "@id": "schema:Protein",
174     "rdfs:subClassOf": {
175         "@id": "schema:BioChemEntity"
176     }
177 },
178 {
179     "@id": "schema:hasBioChemEntityPart",
180     "@type": "rdf:Property",
181     "dct:source": {
182         "@id": "http://www.bioschemas.org"
183     },
184     "rdfs:comment": "Indicates a BioChemEntity that (in some sense) has this
185         ↪ BioChemEntity as a part. Inverse property: <a
186         ↪ href=\"http://schema.org/isPartOfBioChemEntity\">isPartOfBioChemEntity</a>.",
187     "rdfs:label": "hasBioChemEntityPart",
188     "schema:domainIncludes": {
189         "@id": "schema:BioChemEntity"
190     },
191     "schema:inverseOf": {
192         "@id": "schema:isPartOfBioChemEntity"
193     },
194     "schema:isPartOf": {
195         "@id": "http://bio.schema.org"
196     },
197     "schema:rangeIncludes": {
198         "@id": "schema:BioChemEntity"
199     },
200     "schema:sameAs": {
201         "@id": "https://schema.org/hasBioChemEntityPart"
202     }
203 }

```

```

200     }
201   },
202   {
203     "@id": "schema:boundMolecule",
204     "schema:rangeIncludes": {
205       "@id": "schema:BioChemEntity"
206     }
207   },
208   {
209     "@id": "schema:potentialAction",
210     "@type": "rdf:Property",
211     "rdfs:comment": "Indicates a potential Action, which describes an idealized action
212       ↪ in which this thing would play an 'object' role.",
213     "rdfs:label": "potentialAction",
214     "schema:domainIncludes": {
215       "@id": "schema:Thing"
216     },
217     "schema:rangeIncludes": {
218       "@id": "schema:Action"
219     },
220     "schema:sameAs": {
221       "@id": "https://schema.org/potentialAction"
222     }
223   },
224   {
225     "@id": "schema:hasSequenceAnnotation",
226     "@type": "rdf:Property",
227     "dct:source": {
228       "@id": "http://www.bioschemas.org/BioChemEntity"
229     },
230     "rdfs:comment": "Pointer to a sequence annotation; please use PropertyValue if you
231       ↪ want to include any evidence.",
232     "rdfs:label": "hasSequenceAnnotation",
233     "schema:domainIncludes": {
234       "@id": "schema:BioChemEntity"
235     },
236     "schema:isPartOf": {
237       "@id": "http://bio.schema.org"
238     },
239     "schema:rangeIncludes": [
240       {
241         "@id": "schema:SequeneceAnnotation"
242       },
243       {
244         "@id": "schema:PropertyValue"
245       }
246     ],
247     "schema:sameAs": {
248       "@id": "https://schema.org/hasSequenceAnnotation"
249     }
250   },
251   {
252     "@id": "schema:biologicalType",

```

```

251     "schema:rangeIncludes": {
252         "@id": "schema:BioChemEntity"
253     }
254 },
255 {
256     "@id": "schema:isInvolvedInBiologicalProcess",
257     "@type": "rdf:Property",
258     "dct:source": {
259         "@id": "http://www.bioschemas.org/BioChemEntity"
260     },
261     "rdfs:comment": "Biological process this BioChemEntity is involved in; please use
    ↪ PropertyValue if you want to include any evidence.",
262     "rdfs:label": "isInvolvedInBiologicalProcess",
263     "schema:domainIncludes": {
264         "@id": "schema:BioChemEntity"
265     },
266     "schema:isPartOf": {
267         "@id": "http://bio.schema.org"
268     },
269     "schema:rangeIncludes": [
270         {
271             "@id": "schema:CategoryCode"
272         },
273         {
274             "@id": "schema:PropertyValue"
275         }
276     ],
277     "schema:sameAs": {
278         "@id": "https://schema.org/isInvolvedInBiologicalProcess"
279     }
280 },
281 {
282     "@id": "schema:name",
283     "@type": "rdf:Property",
284     "owl:equivalentProperty": {
285         "@id": "dct:title"
286     },
287     "rdfs:comment": "The name of the item.",
288     "rdfs:label": "name",
289     "rdfs:subPropertyOf": {
290         "@id": "rdfs:label"
291     },
292     "schema:domainIncludes": {
293         "@id": "schema:Thing"
294     },
295     "schema:rangeIncludes": {
296         "@id": "schema:Text"
297     },
298     "schema:sameAs": {
299         "@id": "https://schema.org/name"
300     }
301 },
302 {

```



```

303     "@id": "schema:ChemicalSubstance",
304     "rdfs:subClassOf": {
305         "@id": "schema:BioChemEntity"
306     }
307 },
308 {
309     "@id": "schema:taxonomicRange",
310     "@type": "rdf:Property",
311     "dct:source": {
312         "@id": "http://www.bioschemas.org"
313     },
314     "rdfs:comment": "The taxonomic grouping of the organism that expresses, encodes, or
    ↪ in someway related to the BioChemEntity.",
315     "rdfs:label": "taxonomicRange",
316     "schema:domainIncludes": [
317         {
318             "@id": "schema:Phenotype"
319         },
320         {
321             "@id": "schema:BioChemEntity"
322         }
323     ],
324     "schema:isPartOf": {
325         "@id": "http://bio.schema.org"
326     },
327     "schema:rangeIncludes": [
328         {
329             "@id": "schema:URL"
330         },
331         {
332             "@id": "schema:CategoryCode"
333         },
334         {
335             "@id": "schema:Taxon"
336         },
337         {
338             "@id": "schema:Text"
339         }
340     ],
341     "schema:sameAs": {
342         "@id": "https://schema.org/taxonomicRange"
343     }
344 },
345 {
346     "@id": "schema:identifier",
347     "@type": "rdf:Property",
348     "owl:equivalentProperty": {
349         "@id": "dct:identifier"
350     },
351     "rdfs:comment": "The identifier property represents any kind of identifier for any
    ↪ kind of <a class=\"localLink\" href=\"http://schema.org/Thing\">Thing</a>, such
    ↪ as ISBNs, GTIN codes, UUIDs etc. Schema.org provides dedicated properties for
    ↪ representing many of these, either as textual strings or as URL (URI) links.

```

```

352     See <a href="/docs/datamodel.html#identifierBg">background notes</a> for more
↪ details.",
353     "rdfs:label": "identifier",
354     "schema:domainIncludes": {
355         "@id": "schema:Thing"
356     },
357     "schema:rangeIncludes": [
358         {
359             "@id": "schema:PropertyValue"
360         },
361         {
362             "@id": "schema:URL"
363         },
364         {
365             "@id": "schema:Text"
366         }
367     ],
368     "schema:sameAs": {
369         "@id": "https://schema.org/identifier"
370     }
371 },
372 {
373     "@id": "schema:DNA",
374     "rdfs:subClassOf": {
375         "@id": "schema:BioChemEntity"
376     }
377 },
378 {
379     "@id": "schema:isPartOfBioChemEntity",
380     "@type": "rdf:Property",
381     "dct:source": {
382         "@id": "http://www.bioschemas.org"
383     },
384     "rdfs:comment": "Indicates a BioChemEntity that is (in some sense) a part of this
↪ BioChemEntity. Inverse property: <a
↪ href="/http://schema.org/hasBioChemEntityPart">hasBioChemEntityPart</a>.",
385     "rdfs:label": "isPartOfBioChemEntity",
386     "schema:domainIncludes": {
387         "@id": "schema:BioChemEntity"
388     },
389     "schema:inverseOf": {
390         "@id": "schema:hasBioChemEntityPart"
391     },
392     "schema:isPartOf": {
393         "@id": "http://bio.schema.org"
394     },
395     "schema:rangeIncludes": {
396         "@id": "schema:BioChemEntity"
397     },
398     "schema:sameAs": {
399         "@id": "https://schema.org/isPartOfBioChemEntity"
400     }
401 },

```

```

402 {
403   "@id": "schema:bioChemSimilarity",
404   "@type": "rdf:Property",
405   "dct:source": {
406     "@id": "http://www.bioschemas.org"
407   },
408   "rdfs:comment": "A similar molecular entity, e.g., obtained by fingerprint
    ↪ similarity algorithms.",
409   "rdfs:label": "bioChemSimilarity",
410   "schema:domainIncludes": {
411     "@id": "schema:BioChemEntity"
412   },
413   "schema:isPartOf": {
414     "@id": "http://bio.schema.org"
415   },
416   "schema:rangeIncludes": {
417     "@id": "schema:BioChemEntity"
418   },
419   "schema:sameAs": {
420     "@id": "https://schema.org/bioChemSimilarity"
421   }
422 },
423 {
424   "@id": "schema:description",
425   "@type": "rdf:Property",
426   "owl:equivalentProperty": {
427     "@id": "dct:description"
428   },
429   "rdfs:comment": "A description of the item.",
430   "rdfs:label": "description",
431   "schema:domainIncludes": {
432     "@id": "schema:Thing"
433   },
434   "schema:rangeIncludes": {
435     "@id": "schema:Text"
436   },
437   "schema:sameAs": {
438     "@id": "https://schema.org/description"
439   }
440 },
441 {
442   "@id": "schema:hasMolecularFunction",
443   "@type": "rdf:Property",
444   "dct:source": {
445     "@id": "http://www.bioschemas.org/BioChemEntity"
446   },
447   "rdfs:comment": "Molecular function performed by this BioChemEntity; please use
    ↪ PropertyValue if you want to include any evidence.",
448   "rdfs:label": "hasMolecularFunction",
449   "schema:domainIncludes": {
450     "@id": "schema:BioChemEntity"
451   },
452   "schema:isPartOf": {

```

```

453     "@id": "http://bio.schema.org"
454 },
455 "schema:rangeIncludes": [
456     {
457         "@id": "schema:CategoryCode"
458     },
459     {
460         "@id": "schema:PropertyValue"
461     }
462 ],
463 "schema:sameAs": {
464     "@id": "https://schema.org/hasMolecularFunction"
465 }
466 },
467 {
468     "@id": "schema:isMatchedBy",
469     "@type": "rdf:Property",
470     "dct:source": {
471         "@id": "http://www.bioschemas.org"
472     },
473     "rdfs:comment": "A model matching this BioChemEntity.",
474     "rdfs:label": "isMatchedBy",
475     "schema:domainIncludes": {
476         "@id": "schema:BioChemEntity"
477     },
478     "schema:isPartOf": {
479         "@id": "http://bio.schema.org"
480     },
481     "schema:rangeIncludes": {
482         "@id": "schema:SequenceMatchingModel"
483     },
484     "schema:sameAs": {
485         "@id": "https://schema.org/isMatchedBy"
486     }
487 },
488 {
489     "@id": "schema:additionalType",
490     "@type": "rdf:Property",
491     "rdfs:comment": "An additional type for the item, typically used for adding more
492     ↪ specific types from external vocabularies in microdata syntax. This is a
493     ↪ relationship between something and a class that the thing is in. In RDFa syntax,
494     ↪ it is better to use the native RDFa syntax - the 'typeof' attribute - for
495     ↪ multiple types. Schema.org tools may have only weaker understanding of extra
496     ↪ types, in particular those defined externally.",
497     "rdfs:label": "additionalType",
498     "rdfs:subPropertyOf": {
499         "@id": "rdf:type"
500     },
501     "schema:domainIncludes": {
502         "@id": "schema:Thing"
503     },
504     "schema:rangeIncludes": {
505         "@id": "schema:URL"
506     }
507 }

```

```

501     },
502     "schema:sameAs": {
503         "@id": "https://schema.org/additionalType"
504     }
505 },
506 {
507     "@id": "schema:image",
508     "@type": "rdf:Property",
509     "rdfs:comment": "An image of the item. This can be a <a class=\"localLink\"
510         ↪ href=\"http://schema.org/URL\">URL</a> or a fully described <a
511         ↪ class=\"localLink\" href=\"http://schema.org/ImageObject\">ImageObject</a>.",
512     "rdfs:label": "image",
513     "schema:domainIncludes": {
514         "@id": "schema:Thing"
515     },
516     "schema:rangeIncludes": [
517         {
518             "@id": "schema:ImageObject"
519         },
520         {
521             "@id": "schema:URL"
522         }
523     ],
524     "schema:sameAs": {
525         "@id": "https://schema.org/image"
526     }
527 },
528 {
529     "@id": "schema:MolecularEntity",
530     "rdfs:subClassOf": {
531         "@id": "schema:BioChemEntity"
532     }
533 },
534 {
535     "@id": "schema:Thing",
536     "@type": "rdfs:Class",
537     "rdfs:comment": "The most generic type of item.",
538     "rdfs:label": "Thing",
539     "schema:sameAs": {
540         "@id": "https://schema.org/Thing"
541     }
542 },
543 {
544     "@id": "schema:isLocatedInSubcellularLocation",
545     "@type": "rdf:Property",
546     "dct:source": {
547         "@id": "http://www.bioschemas.org/BioChemEntity"
548     },
549     "rdfs:comment": "Subcellular location where this BioChemEntity is located; please
550         ↪ use PropertyValue if you want to include any evidence.",
551     "rdfs:label": "isLocatedInSubcellularLocation",
552     "schema:domainIncludes": {

```

```

551     "@id": "schema:BioChemEntity"
552 },
553     "schema:isPartOf": {
554         "@id": "http://bio.schema.org"
555     },
556     "schema:rangeIncludes": [
557         {
558             "@id": "schema:CategoryCode"
559         },
560         {
561             "@id": "schema:PropertyValue"
562         }
563     ],
564     "schema:sameAs": {
565         "@id": "https://schema.org/isLocatedInSubcellularLocation"
566     }
567 },
568 {
569     "@id": "schema:Enzyme",
570     "rdfs:subClassOf": {
571         "@id": "schema:BioChemEntity"
572     }
573 },
574 {
575     "@id": "schema:subjectOf",
576     "@type": "rdf:Property",
577     "dct:source": {
578         "@id": "https://github.com/schemaorg/schemaorg/issues/1670"
579     },
580     "rdfs:comment": "A CreativeWork or Event about this Thing..",
581     "rdfs:label": "subjectOf",
582     "schema:category": "issue-1670",
583     "schema:domainIncludes": {
584         "@id": "schema:Thing"
585     },
586     "schema:inverseOf": {
587         "@id": "schema:about"
588     },
589     "schema:isPartOf": {
590         "@id": "http://pending.schema.org"
591     },
592     "schema:rangeIncludes": [
593         {
594             "@id": "schema:CreativeWork"
595         },
596         {
597             "@id": "schema:Event"
598         }
599     ],
600     "schema:sameAs": {
601         "@id": "https://schema.org/subjectOf"
602     }
603 },

```

```

604 {
605   "@id": "schema:biologicalRole",
606   "@type": "rdf:Property",
607   "dct:source": {
608     "@id": "http://www.bioschemas.org"
609   },
610   "rdfs:comment": "A role played by the molecular entity within a biological
    ↪ context.",
611   "rdfs:label": "biologicalRole",
612   "schema:domainIncludes": {
613     "@id": "schema:BioChemEntity"
614   },
615   "schema:isPartOf": {
616     "@id": "http://bio.schema.org"
617   },
618   "schema:rangeIncludes": {
619     "@id": "schema:DefinedTerm"
620   },
621   "schema:sameAs": {
622     "@id": "https://schema.org/biologicalRole"
623   }
624 },
625 {
626   "@id": "schema:hasRepresentation",
627   "@type": "rdf:Property",
628   "dct:source": {
629     "@id": "http://www.bioschemas.org"
630   },
631   "rdfs:comment": "A common representation such as a protein sequence or chemical
    ↪ structure for this entity. For images use schema.org/image.",
632   "rdfs:label": "hasRepresentation",
633   "schema:domainIncludes": {
634     "@id": "schema:BioChemEntity"
635   },
636   "schema:isPartOf": {
637     "@id": "http://bio.schema.org"
638   },
639   "schema:rangeIncludes": [
640     {
641       "@id": "schema:Text"
642     },
643     {
644       "@id": "schema:PropertyValue"
645     },
646     {
647       "@id": "schema:URL"
648     }
649   ],
650   "schema:sameAs": {
651     "@id": "https://schema.org/hasRepresentation"
652   }
653 },
654 {

```

```

655     "@id": "schema:encodesBioChemEntity",
656     "schema:rangeIncludes": {
657         "@id": "schema:BioChemEntity"
658     }
659 },
660 {
661     "@id": "schema:mainEntityOfPage",
662     "@type": "rdf:Property",
663     "rdfs:comment": "Indicates a page (or other CreativeWork) for which this thing is
        ↳ the main entity being described. See <a
        ↳ href=\"/docs/datamodel.html#mainEntityBackground\">background notes</a> for
        ↳ details.",
664     "rdfs:label": "mainEntityOfPage",
665     "schema:domainIncludes": {
666         "@id": "schema:Thing"
667     },
668     "schema:inverseOf": {
669         "@id": "schema:mainEntity"
670     },
671     "schema:rangeIncludes": [
672         {
673             "@id": "schema:CreativeWork"
674         },
675         {
676             "@id": "schema:URL"
677         }
678     ],
679     "schema:sameAs": {
680         "@id": "https://schema.org/mainEntityOfPage"
681     }
682 },
683 {
684     "@id": "schema:BioChemEntity",
685     "@type": "rdfs:Class",
686     "dct:source": {
687         "@id": "http://bioschemas.org"
688     },
689     "rdfs:comment": "Any biological, chemical, or biochemical thing. For example: a
        ↳ protein; a gene; a chemical; a synthetic chemical.",
690     "rdfs:label": "BioChemEntity",
691     "rdfs:subClassOf": {
692         "@id": "schema:Thing"
693     },
694     "schema:isPartOf": {
695         "@id": "http://bio.schema.org"
696     },
697     "schema:sameAs": {
698         "@id": "https://schema.org/BioChemEntity"
699     }
700 },
701 {
702     "@id": "schema:alternateName",
703     "@type": "rdf:Property",

```



```

704     "rdfs:comment": "An alias for the item.",
705     "rdfs:label": "alternateName",
706     "schema:domainIncludes": {
707         "@id": "schema:Thing"
708     },
709     "schema:rangeIncludes": {
710         "@id": "schema:Text"
711     },
712     "schema:sameAs": {
713         "@id": "https://schema.org/alternateName"
714     }
715 },
716 {
717     "@id": "schema:sameAs",
718     "@type": "rdf:Property",
719     "rdfs:comment": "URL of a reference Web page that unambiguously indicates the
    ↪ item's identity. E.g. the URL of the item's Wikipedia page, Wikidata entry, or
    ↪ official website.",
720     "rdfs:label": "sameAs",
721     "schema:domainIncludes": {
722         "@id": "schema:Thing"
723     },
724     "schema:rangeIncludes": {
725         "@id": "schema:URL"
726     },
727     "schema:sameAs": {
728         "@id": "https://schema.org/sameAs"
729     }
730 },
731 {
732     "@id": "schema:bioChemInteraction",
733     "@type": "rdf:Property",
734     "dct:source": {
735         "@id": "http://www.bioschemas.org"
736     },
737     "rdfs:comment": "A BioChemEntity that is known to interact with this item.",
738     "rdfs:label": "bioChemInteraction",
739     "schema:domainIncludes": {
740         "@id": "schema:BioChemEntity"
741     },
742     "schema:isPartOf": {
743         "@id": "http://bio.schema.org"
744     },
745     "schema:rangeIncludes": {
746         "@id": "schema:BioChemEntity"
747     },
748     "schema:sameAs": {
749         "@id": "https://schema.org/bioChemInteraction"
750     }
751 }
752 ]
753 }

```

C.4 Gene JSON-Schema

```
1 {
2   "definitions": {
3     "BioChemEntity": {
4       "description": "Any biological, chemical, or biochemical thing. For example: a
5         ⇨ protein; a gene; a chemical; a synthetic chemical.",
6       "properties": {
7         "@context": {
8           "default": "http://schema.org",
9           "options": {
10             "hidden": "true"
11           },
12           "type": "string"
13         },
14         "@type": {
15           "default": "BioChemEntity",
16           "options": {
17             "hidden": "true"
18           },
19           "type": "string"
20         },
21         "additionalType": {
22           "items": {
23             "oneOf": [
24               {
25                 "$ref": "#/definitions/URL",
26                 "title": "URL"
27               }
28             ],
29             "title": "additionalType",
30             "type": "array"
31           },
32           "alternateName": {
33             "items": {
34               "oneOf": [
35                 {
36                 "$ref": "#/definitions/Text",
37                 "title": "Text"
38               }
39             ],
40             "title": "alternateName",
41             "type": "array"
42           },
43           "associatedDisease": {
44             "items": {
45               "oneOf": [
46               {
```

```

48         "properties": {
49             "@id": {
50                 "$ref": "#/definitions/URL",
51                 "title": "@id"
52             },
53             "@type": {
54                 "default": "BioChemEntity",
55                 "options": {
56                     "hidden": "true"
57                 },
58                 "type": "string"
59             }
60         },
61         "required": [
62             "@id",
63             "@type"
64         ],
65         "title": "BioChemEntity"
66     },
67     {
68         "$ref": "#/definitions/URL",
69         "title": "URL"
70     },
71     {
72         "properties": {
73             "@id": {
74                 "$ref": "#/definitions/URL",
75                 "title": "@id"
76             },
77             "@type": {
78                 "default": "BioChemEntity",
79                 "options": {
80                     "hidden": "true"
81                 },
82                 "type": "string"
83             }
84         },
85         "required": [
86             "@id",
87             "@type"
88         ],
89         "title": "BioChemEntity"
90     }
91 ]
92 },
93 "title": "associatedDisease",
94 "type": "array"
95 },
96 "bioChemInteraction": {
97     "items": {
98         "oneOf": [
99             {
100                 "properties": {

```

```

101         "@id": {
102             "$ref": "#/definitions/URL",
103             "title": "@id"
104         },
105         "@type": {
106             "default": "BioChemEntity",
107             "options": {
108                 "hidden": "true"
109             },
110             "type": "string"
111         }
112     },
113     "required": [
114         "@id",
115         "@type"
116     ],
117     "title": "BioChemEntity"
118 }
119 ]
120 },
121 "title": "bioChemInteraction",
122 "type": "array"
123 },
124 "bioChemSimilarity": {
125     "items": {
126         "oneOf": [
127             {
128                 "properties": {
129                     "@id": {
130                         "$ref": "#/definitions/URL",
131                         "title": "@id"
132                     },
133                     "@type": {
134                         "default": "BioChemEntity",
135                         "options": {
136                             "hidden": "true"
137                         },
138                         "type": "string"
139                     }
140                 },
141                 "required": [
142                     "@id",
143                     "@type"
144                 ],
145                 "title": "BioChemEntity"
146             }
147         ]
148     },
149     "title": "bioChemSimilarity",
150     "type": "array"
151 },
152 "biologicalRole": {
153     "items": {

```

```

154         "oneOf": [
155             {
156                 "properties": {
157                     "@id": {
158                         "$ref": "#/definitions/URL",
159                         "title": "@id"
160                     },
161                     "@type": {
162                         "default": "BioChemEntity",
163                         "options": {
164                             "hidden": "true"
165                         },
166                         "type": "string"
167                     }
168                 },
169                 "required": [
170                     "@id",
171                     "@type"
172                 ],
173                 "title": "BioChemEntity"
174             }
175         ]
176     },
177     "title": "biologicalRole",
178     "type": "array"
179 },
180 "description": {
181     "items": {
182         "oneOf": [
183             {
184                 "$ref": "#/definitions/Text",
185                 "title": "Text"
186             }
187         ]
188     },
189     "title": "description",
190     "type": "array"
191 },
192 "disambiguatingDescription": {
193     "items": {
194         "oneOf": [
195             {
196                 "$ref": "#/definitions/Text",
197                 "title": "Text"
198             }
199         ]
200     },
201     "title": "disambiguatingDescription",
202     "type": "array"
203 },
204 "hasBioChemEntityPart": {
205     "items": {
206         "oneOf": [

```

```

207         {
208             "properties": {
209                 "@id": {
210                     "$ref": "#/definitions/URL",
211                     "title": "@id"
212                 },
213                 "@type": {
214                     "default": "BioChemEntity",
215                     "options": {
216                         "hidden": "true"
217                     },
218                     "type": "string"
219                 }
220             },
221             "required": [
222                 "@id",
223                 "@type"
224             ],
225             "title": "BioChemEntity"
226         }
227     ]
228 },
229     "title": "hasBioChemEntityPart",
230     "type": "array"
231 },
232 "hasMolecularFunction": {
233     "items": {
234         "oneOf": [
235             {
236                 "properties": {
237                     "@id": {
238                         "$ref": "#/definitions/URL",
239                         "title": "@id"
240                     },
241                     "@type": {
242                         "default": "BioChemEntity",
243                         "options": {
244                             "hidden": "true"
245                         },
246                         "type": "string"
247                     }
248                 },
249                 "required": [
250                     "@id",
251                     "@type"
252                 ],
253                 "title": "BioChemEntity"
254             },
255             {
256                 "properties": {
257                     "@id": {
258                         "$ref": "#/definitions/URL",
259                         "title": "@id"

```

```

260         },
261         "@type": {
262             "default": "BioChemEntity",
263             "options": {
264                 "hidden": "true"
265             },
266             "type": "string"
267         }
268     },
269     "required": [
270         "@id",
271         "@type"
272     ],
273     "title": "BioChemEntity"
274 }
275 ]
276 },
277 "title": "hasMolecularFunction",
278 "type": "array"
279 },
280 "hasRepresentation": {
281     "items": {
282         "oneOf": [
283             {
284                 "$ref": "#/definitions/Text",
285                 "title": "Text"
286             },
287             {
288                 "properties": {
289                     "@id": {
290                         "$ref": "#/definitions/URL",
291                         "title": "@id"
292                     },
293                     "@type": {
294                         "default": "BioChemEntity",
295                         "options": {
296                             "hidden": "true"
297                         },
298                         "type": "string"
299                     }
300                 },
301                 "required": [
302                     "@id",
303                     "@type"
304                 ],
305                 "title": "BioChemEntity"
306             },
307             {
308                 "$ref": "#/definitions/URL",
309                 "title": "URL"
310             }
311         ]
312     },

```

```

313     "title": "hasRepresentation",
314     "type": "array"
315 },
316 "hasSequenceAnnotation": {
317     "items": {
318         "oneOf": [
319             {
320                 "properties": {
321                     "@id": {
322                         "$ref": "#/definitions/URL",
323                         "title": "@id"
324                     },
325                     "@type": {
326                         "default": "BioChemEntity",
327                         "options": {
328                             "hidden": "true"
329                         },
330                         "type": "string"
331                     }
332                 },
333                 "required": [
334                     "@id",
335                     "@type"
336                 ],
337                 "title": "BioChemEntity"
338             },
339             {
340                 "properties": {
341                     "@id": {
342                         "$ref": "#/definitions/URL",
343                         "title": "@id"
344                     },
345                     "@type": {
346                         "default": "BioChemEntity",
347                         "options": {
348                             "hidden": "true"
349                         },
350                         "type": "string"
351                     }
352                 },
353                 "required": [
354                     "@id",
355                     "@type"
356                 ],
357                 "title": "BioChemEntity"
358             }
359         ]
360     },
361     "title": "hasSequenceAnnotation",
362     "type": "array"
363 },
364 "identifier": {
365     "items": {

```



```

366         "oneOf": [
367             {
368                 "properties": {
369                     "@id": {
370                         "$ref": "#/definitions/URL",
371                         "title": "@id"
372                     },
373                     "@type": {
374                         "default": "BioChemEntity",
375                         "options": {
376                             "hidden": "true"
377                         },
378                         "type": "string"
379                     }
380                 },
381                 "required": [
382                     "@id",
383                     "@type"
384                 ],
385                 "title": "BioChemEntity"
386             },
387             {
388                 "$ref": "#/definitions/URL",
389                 "title": "URL"
390             },
391             {
392                 "$ref": "#/definitions/Text",
393                 "title": "Text"
394             }
395         ]
396     },
397     "title": "identifier",
398     "type": "array"
399 },
400 "image": {
401     "items": {
402         "oneOf": [
403             {
404                 "properties": {
405                     "@id": {
406                         "$ref": "#/definitions/URL",
407                         "title": "@id"
408                     },
409                     "@type": {
410                         "default": "BioChemEntity",
411                         "options": {
412                             "hidden": "true"
413                         },
414                         "type": "string"
415                     }
416                 },
417                 "required": [
418                     "@id",

```

```

419         "@type"
420     ],
421     "title": "BioChemEntity"
422 },
423 {
424     "$ref": "#/definitions/URL",
425     "title": "URL"
426 }
427 ]
428 },
429 "title": "image",
430 "type": "array"
431 },
432 "isEncodedByBioChemEntity": {
433     "items": {
434         "oneOf": [
435             {
436                 "properties": {
437                     "@id": {
438                         "$ref": "#/definitions/URL",
439                         "title": "@id"
440                     },
441                     "@type": {
442                         "default": "BioChemEntity",
443                         "options": {
444                             "hidden": "true"
445                         },
446                         "type": "string"
447                     }
448                 },
449                 "required": [
450                     "@id",
451                     "@type"
452                 ],
453                 "title": "BioChemEntity"
454             },
455             {
456                 "properties": {
457                     "@id": {
458                         "$ref": "#/definitions/URL",
459                         "title": "@id"
460                     },
461                     "@type": {
462                         "default": "BioChemEntity",
463                         "options": {
464                             "hidden": "true"
465                         },
466                         "type": "string"
467                     }
468                 },
469                 "required": [
470                     "@id",
471                     "@type"

```

```

472         ],
473         "title": "BioChemEntity"
474     },
475     {
476         "properties": {
477             "@id": {
478                 "$ref": "#/definitions/URL",
479                 "title": "@id"
480             },
481             "@type": {
482                 "default": "BioChemEntity",
483                 "options": {
484                     "hidden": "true"
485                 },
486                 "type": "string"
487             }
488         },
489         "required": [
490             "@id",
491             "@type"
492         ],
493         "title": "BioChemEntity"
494     }
495 ]
496 },
497 "title": "isEncodedByBioChemEntity",
498 "type": "array"
499 },
500 "isInvolvedInBiologicalProcess": {
501     "items": {
502         "oneOf": [
503             {
504                 "properties": {
505                     "@id": {
506                         "$ref": "#/definitions/URL",
507                         "title": "@id"
508                     },
509                     "@type": {
510                         "default": "BioChemEntity",
511                         "options": {
512                             "hidden": "true"
513                         },
514                         "type": "string"
515                     }
516                 },
517                 "required": [
518                     "@id",
519                     "@type"
520                 ],
521                 "title": "BioChemEntity"
522             },
523             {
524                 "properties": {

```

```

525         "@id": {
526             "$ref": "#/definitions/URL",
527             "title": "@id"
528         },
529         "@type": {
530             "default": "BioChemEntity",
531             "options": {
532                 "hidden": "true"
533             },
534             "type": "string"
535         }
536     },
537     "required": [
538         "@id",
539         "@type"
540     ],
541     "title": "BioChemEntity"
542 }
543 ]
544 },
545 "title": "isInvolvedInBiologicalProcess",
546 "type": "array"
547 },
548 "isLocatedInSubcellularLocation": {
549     "items": {
550         "oneOf": [
551             {
552                 "properties": {
553                     "@id": {
554                         "$ref": "#/definitions/URL",
555                         "title": "@id"
556                     },
557                     "@type": {
558                         "default": "BioChemEntity",
559                         "options": {
560                             "hidden": "true"
561                         },
562                         "type": "string"
563                     }
564                 },
565                 "required": [
566                     "@id",
567                     "@type"
568                 ],
569                 "title": "BioChemEntity"
570             },
571             {
572                 "properties": {
573                     "@id": {
574                         "$ref": "#/definitions/URL",
575                         "title": "@id"
576                     },
577                     "@type": {

```

```

578         "default": "BioChemEntity",
579         "options": {
580             "hidden": "true"
581         },
582         "type": "string"
583     }
584 },
585     "required": [
586         "@id",
587         "@type"
588     ],
589     "title": "BioChemEntity"
590 }
591 ]
592 },
593     "title": "isLocatedInSubcellularLocation",
594     "type": "array"
595 },
596 "isMatchedBy": {
597     "items": {
598         "oneOf": [
599             {
600                 "properties": {
601                     "@id": {
602                         "$ref": "#/definitions/URL",
603                         "title": "@id"
604                     },
605                     "@type": {
606                         "default": "BioChemEntity",
607                         "options": {
608                             "hidden": "true"
609                         },
610                         "type": "string"
611                     }
612                 },
613                 "required": [
614                     "@id",
615                     "@type"
616                 ],
617                 "title": "BioChemEntity"
618             }
619         ]
620     },
621     "title": "isMatchedBy",
622     "type": "array"
623 },
624 "isPartOfBioChemEntity": {
625     "items": {
626         "oneOf": [
627             {
628                 "properties": {
629                     "@id": {
630                         "$ref": "#/definitions/URL",

```

```

        "title": "@id"
    },
    "@type": {
        "default": "BioChemEntity",
        "options": {
            "hidden": "true"
        },
        "type": "string"
    }
},
"required": [
    "@id",
    "@type"
],
"title": "BioChemEntity"
}
]
},
"title": "isPartOfBioChemEntity",
"type": "array"
},
"mainEntityOfPage": {
    "items": {
        "oneOf": [
            {
                "properties": {
                    "@id": {
                        "$ref": "#/definitions/URL",
                        "title": "@id"
                    },
                    "@type": {
                        "default": "BioChemEntity",
                        "options": {
                            "hidden": "true"
                        },
                        "type": "string"
                    }
                },
                "required": [
                    "@id",
                    "@type"
                ],
                "title": "BioChemEntity"
            },
            {
                "$ref": "#/definitions/URL",
                "title": "URL"
            }
        ]
    },
    "title": "mainEntityOfPage",
    "type": "array"
},

```

```

684     "name": {
685         "items": {
686             "oneOf": [
687                 {
688                     "$ref": "#/definitions/Text",
689                     "title": "Text"
690                 }
691             ]
692         },
693         "title": "name",
694         "type": "array"
695     },
696     "potentialAction": {
697         "items": {
698             "oneOf": [
699                 {
700                     "properties": {
701                         "@id": {
702                             "$ref": "#/definitions/URL",
703                             "title": "@id"
704                         },
705                         "@type": {
706                             "default": "BioChemEntity",
707                             "options": {
708                                 "hidden": "true"
709                             },
710                             "type": "string"
711                         }
712                     },
713                     "required": [
714                         "@id",
715                         "@type"
716                     ],
717                     "title": "BioChemEntity"
718                 }
719             ]
720         },
721         "title": "potentialAction",
722         "type": "array"
723     },
724     "sameAs": {
725         "items": {
726             "oneOf": [
727                 {
728                     "$ref": "#/definitions/URL",
729                     "title": "URL"
730                 }
731             ]
732         },
733         "title": "sameAs",
734         "type": "array"
735     },
736     "subjectOf": {

```

```

737     "items": {
738         "oneOf": [
739             {
740                 "properties": {
741                     "@id": {
742                         "$ref": "#/definitions/URL",
743                         "title": "@id"
744                     },
745                     "@type": {
746                         "default": "BioChemEntity",
747                         "options": {
748                             "hidden": "true"
749                         },
750                         "type": "string"
751                     }
752                 },
753                 "required": [
754                     "@id",
755                     "@type"
756                 ],
757                 "title": "BioChemEntity"
758             },
759             {
760                 "properties": {
761                     "@id": {
762                         "$ref": "#/definitions/URL",
763                         "title": "@id"
764                     },
765                     "@type": {
766                         "default": "BioChemEntity",
767                         "options": {
768                             "hidden": "true"
769                         },
770                         "type": "string"
771                     }
772                 },
773                 "required": [
774                     "@id",
775                     "@type"
776                 ],
777                 "title": "BioChemEntity"
778             }
779         ]
780     },
781     "title": "subjectOf",
782     "type": "array"
783 },
784 "taxonomicRange": {
785     "items": {
786         "oneOf": [
787             {
788                 "$ref": "#/definitions/URL",
789                 "title": "URL"

```



```

790     },
791     {
792         "properties": {
793             "@id": {
794                 "$ref": "#/definitions/URL",
795                 "title": "@id"
796             },
797             "@type": {
798                 "default": "BioChemEntity",
799                 "options": {
800                     "hidden": "true"
801                 },
802                 "type": "string"
803             }
804         },
805         "required": [
806             "@id",
807             "@type"
808         ],
809         "title": "BioChemEntity"
810     },
811     {
812         "properties": {
813             "@id": {
814                 "$ref": "#/definitions/URL",
815                 "title": "@id"
816             },
817             "@type": {
818                 "default": "BioChemEntity",
819                 "options": {
820                     "hidden": "true"
821                 },
822                 "type": "string"
823             }
824         },
825         "required": [
826             "@id",
827             "@type"
828         ],
829         "title": "BioChemEntity"
830     },
831     {
832         "$ref": "#/definitions/Text",
833         "title": "Text"
834     }
835 ]
836 },
837 "title": "taxonomicRange",
838 "type": "array"
839 },
840 "url": {
841     "items": {
842         "oneOf": [

```

```

843         {
844             "$ref": "#/definitions/URL",
845             "title": "URL"
846         }
847     ]
848 },
849     "title": "url",
850     "type": "array"
851 }
852 },
853 "required": [
854     "@type",
855     "@context"
856 ],
857 "title": "BioChemEntity"
858 },
859 "Boolean": {
860     "type": "boolean"
861 },
862 "CategoryCode": {
863     "description": "A Category Code.",
864     "properties": {
865         "@id": {
866             "$ref": "#/definitions/URL",
867             "title": "@id"
868         },
869         "@type": {
870             "default": "CategoryCode",
871             "options": {
872                 "hidden": "true"
873             },
874             "type": "string"
875         }
876     },
877     "required": [
878         "@id",
879         "@type"
880     ],
881     "title": "CategoryCode"
882 },
883 "CreativeWork": {
884     "description": "The most generic kind of creative work, including books,
885 ↪ movies, photographs, software programs, etc.",
886     "properties": {
887         "@id": {
888             "$ref": "#/definitions/URL",
889             "title": "@id"
890         },
891         "@type": {
892             "default": "CreativeWork",
893             "options": {
894                 "hidden": "true"

```

```

895         "type": "string"
896     }
897 },
898 "required": [
899     "@id",
900     "@type"
901 ],
902 "title": "CreativeWork"
903 },
904 "Date": {
905     "type": "string"
906 },
907 "DateTime": {
908     "type": "string"
909 },
910 "Gene": {
911     "description": "This Gene profile specification presents the markup for
912     ↪ describing a Gene.",
913     "properties": {
914         "@context": {
915             "default": "http://schema.org",
916             "options": {
917                 "hidden": "true"
918             },
919             "type": "string"
920         },
921         "@type": {
922             "default": "Gene",
923             "options": {
924                 "hidden": "true"
925             },
926             "type": "string"
927         },
928         "identifier": {
929             "description": " Schema.org: The identifier property represents any
930             ↪ kind of identifier for any kind of Thing, such as ISBNs, GTIN
931             ↪ codes, UUIDs etc. Schema.org provides dedicated properties for
932             ↪ representing many of these, either as textual strings or as URL
933             ↪ (URI) links. See [background
934             ↪ notes](http://schema.org/docs/datamodel.html#identifierBg) for
935             ↪ more details.",
936             "oneOf": [
937                 {
938                     "$ref": "#/definitions/PropertyValue",
939                     "title": "PropertyValue"
940                 },
941                 {
942                     "$ref": "#/definitions/Text",
943                     "title": "Text"
944                 },
945                 {
946                     "$ref": "#/definitions/URL",
947                     "title": "URL"
948                 }
949             ]
950         }
951     }
952 }

```

```

941         }
942     ],
943     "title": "identifier (Minimum)"
944 },
945     "name": {
946         "description": " Schema.org: The name of the item.",
947         "oneOf": [
948             {
949                 "$ref": "#/definitions/Text",
950                 "title": "Text"
951             }
952         ],
953         "title": "name (Minimum)"
954     }
955 },
956 "required": [
957     "@type",
958     "@context",
959     "identifier",
960     "name"
961 ],
962 "title": "Gene",
963 "type": "object"
964 },
965 "ImageObject": {
966     "description": "An image file.",
967     "properties": {
968         "@id": {
969             "$ref": "#/definitions/URL",
970             "title": "@id"
971         },
972         "@type": {
973             "default": "ImageObject",
974             "options": {
975                 "hidden": "true"
976             },
977             "type": "string"
978         }
979     },
980     "required": [
981         "@id",
982         "@type"
983     ],
984     "title": "ImageObject"
985 },
986 "Integer": {
987     "type": "integer"
988 },
989 "MedicalCondition": {
990     "description": "Any condition of the human body that affects the normal
991         ⇨ functioning of a person, whether physically or mentally. Includes
992         ⇨ diseases, injuries, disabilities, disorders, syndromes, etc.",
993     "properties": {

```

```

992         "@id": {
993             "$ref": "#/definitions/URL",
994             "title": "@id"
995         },
996         "@type": {
997             "default": "MedicalCondition",
998             "options": {
999                 "hidden": "true"
1000             },
1001             "type": "string"
1002         }
1003     },
1004     "required": [
1005         "@id",
1006         "@type"
1007     ],
1008     "title": "MedicalCondition"
1009 },
1010 "Number": {
1011     "type": "number"
1012 },
1013 "PropertyValue": {
1014     "description": "A property-value pair, e.g. representing a feature of a
1015         ↳ product or place. Use the 'name' property for the name of the property. If
1016         ↳ there is an additional human-readable version of the value, put that into
1017         ↳ the 'description' property.<br/><br/>\n\nAlways use specific schema.org
1018         ↳ properties when a) they exist and b) you can populate them. Using
1019         ↳ PropertyValue as a substitute will typically not trigger the same effect
1020         ↳ as using the original, specific property.",
1021     "properties": {
1022         "@id": {
1023             "$ref": "#/definitions/URL",
1024             "title": "@id"
1025         },
1026         "@type": {
1027             "default": "PropertyValue",
1028             "options": {
1029                 "hidden": "true"
1030             },
1031             "type": "string"
1032         }
1033     },
1034     "required": [
1035         "@id",
1036         "@type"
1037     ],
1038     "title": "PropertyValue"
1039 },
1040 "Protein": {
1041     "description": "This Protein profile specification presents the markup when
1042         ↳ describing a Protein.",
1043     "properties": {
1044         "@context": {

```

```

1038     "default": "http://schema.org",
1039     "options": {
1040         "hidden": "true"
1041     },
1042     "type": "string"
1043 },
1044 "@type": {
1045     "default": "Protein",
1046     "options": {
1047         "hidden": "true"
1048     },
1049     "type": "string"
1050 },
1051 "identifier": {
1052     "description": " Schema.org: The identifier property represents any
        ⇒ kind of identifier for any kind of Thing, such as ISBNs, GTIN
        ⇒ codes, UUIDs etc. Schema.org provides dedicated properties for
        ⇒ representing many of these, either as textual strings or as URL
        ⇒ (URI) links. See [background
        ⇒ notes](http://schema.org/docs/datamodel.html#identifierBg) for
        ⇒ more details.",
1053     "oneOf": [
1054         {
1055             "$ref": "#/definitions/PropertyValue",
1056             "title": "PropertyValue"
1057         },
1058         {
1059             "$ref": "#/definitions/Text",
1060             "title": "Text"
1061         },
1062         {
1063             "$ref": "#/definitions/URL",
1064             "title": "URL"
1065         }
1066     ],
1067     "title": "identifier (Minimum)"
1068 },
1069 "name": {
1070     "description": " Schema.org: The name of the item.",
1071     "oneOf": [
1072         {
1073             "$ref": "#/definitions/Text",
1074             "title": "Text"
1075         }
1076     ],
1077     "title": "name (Minimum)"
1078 }
1079 },
1080 "required": [
1081     "@type",
1082     "@context",
1083     "identifier",
1084     "name"

```

```

1085     ],
1086     "title": "Protein",
1087     "type": "object"
1088 },
1089 "String": {
1090     "type": "string"
1091 },
1092 "Taxon": {
1093     "description": "This profile aims to denote a taxon by common properties such
        ↳ as its scientific name, authority, taxonomic rank and vernacular names. It
        ↳ is also a means to link to existing taxonomic registers where each taxon
        ↳ has a URI.",
1094     "properties": {
1095         "@context": {
1096             "default": "http://schema.org",
1097             "options": {
1098                 "hidden": "true"
1099             },
1100             "type": "string"
1101         },
1102         "@type": {
1103             "default": "Taxon",
1104             "options": {
1105                 "hidden": "true"
1106             },
1107             "type": "string"
1108         },
1109         "name": {
1110             "description": " Bioschemas.org: Taxon name without authorship nor
        ↳ date information of the currently valid (zoological) or accepted
        ↳ (botanical) taxon. Schema.org: The name of the item.",
1111             "oneOf": [
1112                 {
1113                     "$ref": "#/definitions/Text",
1114                     "title": "Text"
1115                 }
1116             ],
1117             "title": "name (Minimum)"
1118         }
1119     },
1120     "required": [
1121         "@type",
1122         "@context",
1123         "name"
1124     ],
1125     "title": "Taxon",
1126     "type": "object"
1127 },
1128 "Text": {
1129     "format": "text",
1130     "type": "string"
1131 },
1132 "URL": {

```

```

1133     "format": "url",
1134     "type": "string"
1135   }
1136 },
1137 "description": "This Gene profile specification presents the markup for describing a
    ↪ Gene.",
1138 "properties": {
1139   "@context": {
1140     "default": "http://schema.org",
1141     "options": {
1142       "hidden": "true"
1143     },
1144     "type": "string"
1145   },
1146   "@type": {
1147     "default": "Gene",
1148     "options": {
1149       "hidden": "true"
1150     },
1151     "type": "string"
1152   },
1153   "additionalProperty": {
1154     "description": " Bioschemas.org: Whenever possible, please use a property
    ↪ coined in a third-party well-know vocabulary. For instance, you can
    ↪ directly use http://purl.obolibrary.org/obo/RO_0002327 as a property to
    ↪ express how a protein or gene enables some GO molecular function. If you
    ↪ still want or need to use additionalProperty, please use (i) property name
    ↪ to specify the name of the property, (ii) additionalType (if possible) to
    ↪ better specify the nature of the property, and (iii) value to link to the
    ↪ object/range of this property. Schema.org: A property-value pair
    ↪ representing an additional characteristics of the entity, e.g. a product
    ↪ feature or another characteristic for which there is no matching property
    ↪ in schema.org.\n\nNote: Publishers should be aware that applications
    ↪ designed to use specific schema.org properties (e.g.
    ↪ http://schema.org/width, http://schema.org/color,
    ↪ http://schema.org/gtin13, ...) will typically expect such data to be
    ↪ provided using those properties, rather than using the generic
    ↪ property/value mechanism.",
1155     "items": {
1156       "oneOf": [
1157         {
1158           "$ref": "#/definitions/PropertyValue",
1159           "title": "PropertyValue"
1160         }
1161       ]
1162     },
1163     "propertyOrder": 3,
1164     "title": "additionalProperty (Optional)",
1165     "type": "array"
1166   },
1167
1168
1169

```



```

1170     "additionalType": {
1171         "description": " Bioschemas.org: Any ontology term describing the gene
            ↳ concept. This is in addition to the official type used in
            ↳ Bioschemas.\n\n**Bioschemas Gene**.: Official proposed term for the profile
            ↳ describing a gene: [SO:gene](http://purl.obolibrary.org/obo/SO_0000704).
            ↳ Schema.org: An additional type for the item, typically used for adding
            ↳ more specific types from external vocabularies in microdata syntax. This
            ↳ is a relationship between something and a class that the thing is in. In
            ↳ RDFa syntax, it is better to use the native RDFa syntax - the 'typeof'
            ↳ attribute - for multiple types. Schema.org tools may have only weaker
            ↳ understanding of extra types, in particular those defined externally.",
1172         "items": {
1173             "oneOf": [
1174                 {
1175                     "$ref": "#/definitions/URL",
1176                     "title": "URL"
1177                 }
1178             ]
1179         },
1180         "propertyOrder": 3,
1181         "title": "additionalType (Optional)",
1182         "type": "array"
1183     },
1184     "alternateName": {
1185         "description": " Schema.org: An alias for the item.",
1186         "items": {
1187             "oneOf": [
1188                 {
1189                     "$ref": "#/definitions/Text",
1190                     "title": "Text"
1191                 }
1192             ]
1193         },
1194         "propertyOrder": 3,
1195         "title": "alternateName (Optional)",
1196         "type": "array"
1197     },
1198     "associatedDisease": {
1199         "description": " Schema.org: Disease associated to this BioChemEntity",
1200         "propertyOrder": 3,
1201         "title": "associatedDisease (Optional)"
1202     },
1203     "description": {
1204         "description": " Schema.org: A description of the item.",
1205         "oneOf": [
1206             {
1207                 "$ref": "#/definitions/Text",
1208                 "title": "Text"
1209             }
1210         ],
1211         "propertyOrder": 2,
1212         "title": "description (Recommended)"
1213     },

```

```

1214     "enablesMF": {
1215         "description": " Bioschemas.org: RO:0002327 (enables). GO molecular function
            ↳ enabled by the gene/protein. \nRecommended range: BioChemEntity or
            ↳ CategoryCode, ProteinAnnotation if evidence should be included.",
1216         "items": {
1217             "oneOf": [
1218                 {
1219                     "$ref": "#/definitions/CategoryCode",
1220                     "title": "CategoryCode"
1221                 },
1222                 {
1223                     "$ref": "#/definitions/PropertyValue",
1224                     "title": "PropertyValue"
1225                 }
1226             ]
1227         },
1228         "propertyOrder": 3,
1229         "title": "enablesMF (Optional)",
1230         "type": "array"
1231     },
1232     "encodes": {
1233         "description": " Bioschemas.org: For genes, this property is used to express
            ↳ in a generic way gene products encoded by this gene. Two more specific
            ↳ properties SIO:010082 (is translated into) and SIO:010080 (is transcribed
            ↳ into) should be used for (protein) translation and (RNA) transcription
            ↳ respectively.\nNote: bioschemas:encodes skos:closeMatch SIO:010078
            ↳ Schema.org: This property is used to link to gene products encoded (even
            ↳ indirectly) from this gene such as RNA or proteins.",
1234         "items": {
1235             "oneOf": [
1236                 {
1237                     "$ref": "#/definitions/BioChemEntity",
1238                     "title": "BioChemEntity"
1239                 },
1240                 {
1241                     "$ref": "#/definitions/Protein",
1242                     "title": "Protein"
1243                 },
1244                 {
1245                     "$ref": "#/definitions/URL",
1246                     "title": "URL"
1247                 }
1248             ]
1249         },
1250         "propertyOrder": 2,
1251         "title": "encodes (Recommended)",
1252         "type": "array"
1253     },
1254     "hasBioChemEntityPart": {
1255         "description": " Bioschemas.org: For genes, it can be used to link to gene
            ↳ sequence annotations such as variants or so.\n\n",
1256         "items": {
1257             "oneOf": [

```

```

1258         {
1259             "$ref": "#/definitions/BioChemEntity",
1260             "title": "BioChemEntity"
1261         }
1262     ]
1263 },
1264 "propertyOrder": 3,
1265 "title": "hasBioChemEntityPart (Optional)",
1266 "type": "array"
1267 },
1268 "hasCategoryCode": {
1269     "description": " Schema.org: A Category code contained in this code set.",
1270     "items": {
1271         "oneOf": [
1272             {
1273                 "$ref": "#/definitions/CategoryCode",
1274                 "title": "CategoryCode"
1275             }
1276         ]
1277     },
1278     "propertyOrder": 3,
1279     "title": "hasCategoryCode (Optional)",
1280     "type": "array"
1281 },
1282 "hasRepresentation": {
1283     "description": " Bioschemas.org: For genes, this property could be used, for
1284         ↪ instance, to register a gene sequence as its representation. If you want
1285         ↪ to better define the nature of the representation, use a PropertyValue as
1286         ↪ described in additionalProperty or a third-party ontology predicate.
1287         ↪ Schema.org: A common representation such as a protein sequence or chemical
1288         ↪ structure for this entity. For images use schema.org/image.",
1289     "items": {
1290         "oneOf": [
1291             {
1292                 "$ref": "#/definitions/PropertyValue",
1293                 "title": "PropertyValue"
1294             },
1295             {
1296                 "$ref": "#/definitions/Text",
1297                 "title": "Text"
1298             },
1299             {
1300                 "$ref": "#/definitions/URL",
1301                 "title": "URL"
1302             }
1303         ]
1304     },
1305     "propertyOrder": 2,
1306     "title": "hasRepresentation (Recommended)",
1307     "type": "array"
1308 },

```

```

1306     "identifier": {
1307         "description": " Schema.org: The identifier property represents any kind of
            ↳ identifier for any kind of Thing, such as ISBNs, GTIN codes, UUIDs etc.
            ↳ Schema.org provides dedicated properties for representing many of these,
            ↳ either as textual strings or as URL (URI) links. See [background
            ↳ notes](http://schema.org/docs/datamodel.html#identifierBg) for more
            ↳ details.",
1308         "oneOf": [
1309             {
1310                 "$ref": "#/definitions/PropertyValue",
1311                 "title": "PropertyValue"
1312             },
1313             {
1314                 "$ref": "#/definitions/Text",
1315                 "title": "Text"
1316             },
1317             {
1318                 "$ref": "#/definitions/URL",
1319                 "title": "URL"
1320             }
1321         ],
1322         "propertyOrder": 1,
1323         "title": "identifier (Minimum)"
1324     },
1325     "image": {
1326         "description": " Schema.org: An image of the item. This can be a URL or a
            ↳ fully described ImageObject.",
1327         "oneOf": [
1328             {
1329                 "$ref": "#/definitions/ImageObject",
1330                 "title": "ImageObject"
1331             },
1332             {
1333                 "$ref": "#/definitions/URL",
1334                 "title": "URL"
1335             }
1336         ],
1337         "propertyOrder": 2,
1338         "title": "image (Recommended)"
1339     },
1340     "involvedInBP": {
1341         "description": " Bioschemas.org: RO:0002331 (is involved in). GO biological
            ↳ process this gene/protein is involved in.\nRecommended range:
            ↳ BioChemEntity or CategoryCode, ProteinAnnotation if evidence should be
            ↳ included.",
1342         "items": {
1343             "oneOf": [
1344                 {
1345                     "$ref": "#/definitions/CategoryCode",
1346                     "title": "CategoryCode"
1347                 },
1348                 {
1349                     "$ref": "#/definitions/PropertyValue",

```

```

1350         "title": "PropertyValue"
1351     }
1352 ]
1353 },
1354 "propertyOrder": 3,
1355 "title": "involvedInBP (Optional)",
1356 "type": "array"
1357 },
1358 "isPartOfBioChemEntity": {
1359     "description": " Bioschemas.org: **Bioschemas Gene**:\nFor genes, it is
        ↳ recommended to at least specify the DNA/chromosome containing this gene
        ↳ and the taxon/organism associated to it. For taxon/organism, it is a good
        ↳ practice to use categoryCode to point to a controlled vocabulary such as
        ↳ NCBI taxon or UniProt Taxonomy. Schema.org: Indicates a BioChemEntity that
        ↳ is (in some sense) a part of this BioChemEntity.\nInverse property:
        ↳ hasBioChemEntity",
1360     "items": {
1361         "oneOf": [
1362             {
1363                 "$ref": "#/definitions/BioChemEntity",
1364                 "title": "BioChemEntity"
1365             }
1366         ]
1367     },
1368     "propertyOrder": 2,
1369     "title": "isPartOfBioChemEntity (Recommended)",
1370     "type": "array"
1371 },
1372 "isTranscribedInto": {
1373     "description": " Bioschemas.org: SIO:010080 (is transcribed into). For genes,
        ↳ this property is used to link to gene products transcribed from this gene
        ↳ such as RNA.",
1374     "items": {
1375         "oneOf": [
1376             {
1377                 "$ref": "#/definitions/BioChemEntity",
1378                 "title": "BioChemEntity"
1379             }
1380         ]
1381     },
1382     "propertyOrder": 3,
1383     "title": "isTranscribedInto (Optional)",
1384     "type": "array"
1385 },
1386 "isVariantOf": {
1387     "description": " Bioschemas.org: SIO: 000272 (is variant of). Use this
        ↳ property to express when a gene is a variant of any other gene.",
1388     "items": {
1389         "oneOf": [
1390             {
1391                 "$ref": "#/definitions/BioChemEntity",
1392                 "title": "BioChemEntity"
1393             }

```

```

1394         {
1395             "$ref": "#/definitions/Gene",
1396             "title": "Gene"
1397         }
1398     ]
1399 },
1400 "propertyOrder": 3,
1401 "title": "isVariantOf (Optional)",
1402 "type": "array"
1403 },
1404 "mainEntityOfPage": {
1405     "description": " Bioschemas.org: Link via DataRecord to the main DataRecord
        ↳ representing this entity in a dataset. It is usually preferred to use
        ↳ mainEntity from a DataRecord to point to its corresponding entity.
        ↳ Schema.org: Indicates a page (or other CreativeWork) for which this thing
        ↳ is the main entity being described. See [background
        ↳ notes](http://schema.org/docs/datamodel.html#mainEntityBackground) for
        ↳ details.\nInverse property: mainEntity.",
1406     "oneOf": [
1407         {
1408             "$ref": "#/definitions/CreativeWork",
1409             "title": "CreativeWork"
1410         },
1411         {
1412             "$ref": "#/definitions/URL",
1413             "title": "URL"
1414         }
1415     ],
1416     "propertyOrder": 3,
1417     "title": "mainEntityOfPage (Optional)"
1418 },
1419 "name": {
1420     "description": " Schema.org: The name of the item.",
1421     "oneOf": [
1422         {
1423             "$ref": "#/definitions/Text",
1424             "title": "Text"
1425         }
1426     ],
1427     "propertyOrder": 1,
1428     "title": "name (Minimum)"
1429 },
1430 "sameAs": {
1431     "description": " Schema.org: URL of a reference Web page that unambiguously
        ↳ indicates the item's identity. E.g. the URL of the item's Wikipedia page,
        ↳ Wikidata entry, or official website.",
1432     "items": {
1433         "oneOf": [
1434             {
1435                 "$ref": "#/definitions/URL",
1436                 "title": "URL"
1437             }
1438         ]

```

```

1439     },
1440     "propertyOrder": 3,
1441     "title": "sameAs (Optional)",
1442     "type": "array"
1443   },
1444   "taxonomicRange": {
1445     "description": " Schema.org: The taxonomic grouping of the organism that
      ↳ expresses, encodes, or in someway related to the BioChemEntity.",
1446     "propertyOrder": 3,
1447     "title": "taxonomicRange (Optional)"
1448   },
1449   "url": {
1450     "description": " Bioschemas.org: Link to the official webpage associated to
      ↳ this entity. Schema.org: URL of the item.",
1451     "oneOf": [
1452       {
1453         "$ref": "#/definitions/URL",
1454         "title": "URL"
1455       }
1456     ],
1457     "propertyOrder": 2,
1458     "title": "url (Recommended)"
1459   }
1460 },
1461 "required": [
1462   "@type",
1463   "@context",
1464   "identifier",
1465   "name"
1466 ],
1467 "title": "Gene (v0.4)",
1468 "type": "object"
1469 }

```

C.5 Gene Additional Information JSON

```
{
  "additionalType": {
    "example": "{\n  \@type\": [\"Gene\"],\n  \"additionalType\":\n    ↪ \"http://semanticscience.org/resource/SIO_010035\"\n}\",
    "marginality": "Optional"
  },
  "alternateName": {
    "example": "{\n  \@type\": [\"Gene\"],\n  \"alternateName\": \"AD1\"\n}\",
    "marginality": "Optional"
  },
  "description": {
    "example": "{\n  \@type\": [\"Gene\"],\n  \"description\": \"amyloid beta\n    ↪ precursor\"\n}\",
    "marginality": "Recommended"
  },
  "enablesMF": {
    "controlled_vocab": "Only the molecular function branch of [Gene Ontology\n    ↪ (GO)](http://www.geneontology.org)",
    "example": "//Gene enabling a GO molecular function\n{\n  \@type\": [\"Gene\"],\n  ↪ \"enablesMF\": {\n    \@type\": \"CategoryCode\", \n    \"url\":\n    ↪ \"http://purl.obolibrary.org/obo/GO_0000166\", \n    \"codeValue\":\n    ↪ \"GO:0000166\", \n    \"name\": \"nucleotide binding\"\n  }\n}\",
    "marginality": "Optional"
  },
  "encodes": {
    "controlled_vocab": "SIO",
    "example": "{\n  \@type\": [\"Gene\"],\n  \"encodes\": {\n    \@type\":\n    ↪ [\"Protein\"], \n    \"identifier\": \"uniprotkb:P05067\", \n    \"url\":\n    ↪ \"https://www.uniprot.org/uniprot/P05067\"\n  }\n}\",
    "marginality": "Recommended"
  },
  "hasBioChemEntityPart": {
    "example": "//Contains a variant\n{\n  \@type\": [\"Gene\"],\n  ↪ \"hasBioChemEntityPart\": {\n    \@type\": \"BioChemEntity\", \n    \"additionalType\": \"http://semanticscience.org/resource/SIO_001381\", \n    ↪ \"identifier\": \"rs577882423\", \n    }, \n    \"url\":\n    ↪ \"https://www.ensembl.org/Homo_sapiens/Variation/\n    Summary?db=core;g=ENSG00000142192;r=21:25880550-26171128;vf=110312838\"\n  }\n}\n",
    "marginality": "Optional"
  },
  "hasRepresentation": {
    "example": "{\n  \@type\": [\"Gene\"],\n  \"hasRepresentation\":\n    ↪ \"https://www.ncbi.nlm.nih.gov/nuccore/\n    NC_000021.9?report=fasta&from=25880550&to=26171128&strand=true\"\n}\",
    "marginality": "Recommended"
  },
}
```



```

36  "identifier": {
37      "example": "//Using a text\n{\n    \@type\: [\"Gene\"],\n    \"identifier\":
        ↳ \"ENSG00000142192\"}\n//Using a URL directly (looks like a text)\n{\n
        ↳ \@type\: [\"Gene\"],\n    \"identifier\":
        ↳ \"http://identifiers.org/ensembl:ENSG00000142192\"}\n//Using a URL modeled
        ↳ as schema:URL (equivalent to the previous one but more verbose)\n{\n
        ↳ \@type\: [\"Gene\"],\n    \"identifier\": {\n        ↳ \@type\: \"URL\", \n
        ↳ \"url\": \"http://identifiers.org/ensembl:ENSG00000142192\"}\n    }\n}\",
38      "marginality": "Minimum"
39  },
40  "image": {
41      "example": "//Using a URL directly (looks like a text)\n{\n    \@type\:
        ↳ [\"Gene\"],\n    \"image\":
        ↳ \"https://en.wikipedia.org/wiki/Amyloid_precursor_protein#
42      /media/File:Ideogram_human_chromosome_21.svg\"}\n",
43      "marginality": "Recommended"
44  },
45  "involvedInBP": {
46      "controlled_vocab": "Only the biological process branch of [Gene Ontology
        ↳ (GO)](http://www.geneontology.org)",
47      "example": "//Gene involved in a GO biological process\n{\n    \@type\:
        ↳ [\"Gene\"],\n    \"involvedInBP\": {\n        ↳ \@type\: \"CategoryCode\", \n
        ↳ \"url\": \"http://purl.obolibrary.org/obo/GO_0000278\", \n        ↳ \"codeValue\":
        ↳ \"GO:0000278\", \n        ↳ \"name\": \"mitotic cell cycle\"}\n    }\n",
48      "marginality": "Optional"
49  },
50  "isPartOfBioChemEntity": {
51      "controlled_vocab": "For subcellular locations branch from GO, please use [Gene
        ↳ Ontology (GO)](http://www.geneontology.org)",
52      "example": "//Is contained in a chromosome in positions X to Y\n{\n    \@type\:
        ↳ [\"Gene\"],\n    \"isPartOfBioChemEntity\": {\n        ↳ \@type\:
        ↳ \"BioChemEntity\", \n        ↳ \"name\": \"Chromosome 21\", \n    }\n//Is contained
        ↳ in an organism\n{\n    \@type\: [\"Gene\"],\n    \"isPartOf\": {\n        ↳ \@type\:
        ↳ \"BioChemEntity\", \n        ↳ \"name\": \"Homo sapiens\", \n        ↳ \"alternateName\":
        ↳ \"Human\", \n        ↳ \"codeCategory\": {\n            ↳ \@type\: \"CategoryCode\", \n
        ↳ \"codeValue\": \"9606\", \n            ↳ \"url\":
        ↳ \"http://purl.bioontology.org/ontology/NCBITAXON/9606\", \n            ↳ \"sameAs\":
        ↳ \"http://purl.uniprot.org/taxonomy/9606\", \n            ↳ \"inCodeSet\": {\n
        ↳ \@type\: \"CategoryCodeSet\", \n            ↳ \"name\": \"NCBI taxon\"}\n        }\n    }\n    }\n}\",
53      "marginality": "Recommended"
54  },
55  "isTranscribedInto": {
56      "controlled_vocab": "SIO",
57      "marginality": "Optional"
58  },
59  "isVariantOf": {
60      "controlled_vocab": "SIO",
61      "marginality": "Optional"
62  },
63
64
65

```

```

66 "mainEntityOfPage": {
67   "example": "//Preferred way from DataRecord\n{\n\n  \@type\: \"DataRecord\", \n
    ↪   \@id\: \"http://rdf.ebi.ac.uk/resource/ensembl/ENSG00000142192\" \n
    ↪   \"mainEntity\": {\n    \@type\: [\"Gene\"], \n    \"name\: \"Name and any
    ↪   other property for this Gene entity\" \n  }\n}\n\n/Also possible from
    ↪   Gene\n{\n  \@type\: [\"Gene\"], \n  \"mainEntityOfPage\: { \@id\:
    ↪   \"http://rdf.ebi.ac.uk/resource/ensembl/ENSG00000142192\" }\n}\",
68   "marginallity": "Optional"
69 },
70 "name": {
71   "example": "{\n  \@type\: [\"Gene\"], \n  \"name\: \"APP\" \n}",
72   "marginallity": "Minimum"
73 },
74 "sameAs": {
75   "example": "{\n  \@type\: [\"Gene\"], \n  \"sameAs\:
    ↪   \"https://www.wikidata.org/wiki/Q14865870\" \n}",
76   "marginallity": "Optional"
77 },
78 "url": {
79   "example": "{\n  \@type\: [\"Gene\"], \n  \"url\:
    ↪   \"https://www.ensembl.org/Homo_sapiens
80   /Gene/Summary?g=ENSG00000142192;r=21:25880550-26171128\" \n}",
81   "marginallity": "Recommended"
82 }
83 }

```