PEMBANGUNAN MUZIUM SEMANTIK BAGI MENYOKONG CARIAN MAKLUMAT SEMANTIK WARISAN BUDAYA

Arisya Nurqamarina binti Md Isa
Dr Lailatul Qadri

Fakulti Teknologi & Sains Maklumat, Universiti Kebangsaan Malaysia

ABSTRAK

Muzium secara umumnya hanya menyebarkan maklumat mengenai koleksi dan artifak melalui pameran yang hanya terhad di tempat pameran itu sendiri. Dan disebabkan pelbagai faktor seperti lokasi pameran, tempat dan jadual menyebabkan bilangan orang ke pameran berada dalam minoriti. Selalunya, kita hanya akan jumpa gambar koleksi dan bukannya maklumat lengkap mengenai warisan-warisan kebangsaan ini. Capaian maklumat untuk koleksi muzium amatlah terhad kerana tidak diekspoitasi secara betul. Teknologi seperti 'Internet of Things' atau lebih dikenali sebagai IOT dan ruang pintar menyediakan asas yang berkesan bagi membina muzium pintar berdasarkan infrastruktur digital dan teknologi maklumat. Web Semantik pula adalah evolusi yang logic dari laman sesawang yang sedia ada. Ia adalah berdasarkan kepada model data konseptual umum yang membolehkan kedua-dua manusia dan mesin berkerja secara tidak langsung tetapi berkait seakan akan ia adalah satu pangkalan data global. Dengan adanya sistem web muzium semantic ini di negara kita, bukan sahaja kita akan jadi negara pertama di Asia Tenggara yang menggunakan sistem ini malah kita dapat mengurangkan beban maklumat. Oleh itu, penyampaian ini memberi tumpuan kepada pembangunan muzium yang melibatkan metadata untuk muzium, koleksi muzium ontologi dan enjin carian semantik bagi menyokong carian maklumat semantik warisan budaya.

1 PENGENALAN

Idea web semantik pertama kali didedahkan pada tahun 2001 dalam artikel "*Scientific America*", di mana *Tim Berners-Lee* memainkan peranan yang amat penting dalam penubuhan Web Semantik. Selain *Tim*, penyelidik "Artificial Intelligence" *James Hendler* dan saintis komputer *Ora Lassila* turut bekerjasama dalam membangunkan idea untuk menyambung maklumat menggunakan rangkaian yang boleh dibaca oleh mesin. Menurut "*World Wide Web Consortium*" (W3C), Web Semantik adalah "kerangka umum yang membolehkan data dikongsi dan digunakan semula meliputi aplikasi, perusahaan, dan sempadan komuniti." Kita juga boleh mengatakan bahawa secara umumnya, Web Semantik adalah mengenai penambahan deskripsi data untuk kandungan dan data yang sedia ada di web agar mesin mempunyai keupayaan untuk membuat tafsiran yang sama seperti cara pemprosesan maklumat manusia untuk mencapai matlamat mereka. Matlamat utama web semantik adalah untuk

mencetuskan evolusi web yang sedia ada untuk membolehkan pengguna mencari, menemui, berkongsi dan menyertai maklumat dengan usaha yang kurang.

Menurut ICOMOS, (2002) Warisan Budaya adalah ungkapan cara hidup yang dibangunkan oleh komuniti dan diteruskan dari generasi ke generasi, termasuk adat, amalan, tempat, objek, ungkapan artistik dan nilai-nilai. Warisan budaya memainkan peranan yang sangat penting dalam kehidupan kita, terutama untuk menjadikan kita mempunyai pegangan pada agama, tradisi, dan kepercayaan kita. Bagi memastikan ia tidak pupus atau hilang dek zaman, warisan budaya yang ketara harus tetap relevan dengan budaya dan diamalkan secara kerap dan dipelajari dalam masyarakat dan antara generasi. Menerusi kertas ini akan diterangkan satu kajian tentang bagaimana teknologi Web Semantik boleh digunakan untuk mendedahkan maklumat warisan budaya dengan cara yang berskala.

2 PERNYATAAN MASALAH

Sumber maklumat berkaitan warisan negara boleh didapati di laman web, muzium, buku-buku di perpustakaan dan sebagainya. Namun, permasalahannya di sini adalah kekangan masa dan kesukaran untuk mendapatkan maklumat lengkap dengan pantas *dan* mudah.

a. Web: Maklumat yang ada di web tidak lengkap selain butiran yang diberikan terlalu ringkas.

b. Muzium: Pengguna perlu hadir ke muzium sendiri untuk mendapatkan maklumat.

c. Buku-buku rujukan: Pengguna perlu mencari buku di perpustakaan bagi mendapatkan maklumat.

Sistem capaian maklumat semantic kebangsaan telah pun dibangunkan sebelum ini oleh salah satu pelajar Universiti Kebangsaan Malaysia (UKM) sendiri namun terdapat sedikit komplikasi. Walaupun, ontologi sudahpun wujud, namun terdapat kekurangan di bahagian paparan maklumat pada antara muka sistem yang dibina. Penggunaan templat yang terdahulu menyebabkan maklumat tidak dapat dipaparkan secara menyeluruh. Selain itu, tidak ada

maklumat tambahan seperti gambar rajah untuk menunjukkan maklumat agar kelihatan lebih menarik dan jelas.

3 OBJEKTIF KAJIAN

Projek ini bertujuan untuk membangunkan muzium semantik untuk warisan ketara dan tidak ketara Malaysia. Ia bertujuan untuk memberi pendedahan mengenai artifak ini kepada orang ramai tanpa perlu pergi ke muzium.

Selain itu, objektif kajian ini juga adalah untuk menguji muzium semantik untuk warisan ketara dan tidak ketara Malaysia bagi memastikan ia dapat digunakan oleh orang ramai tanpa sebarang masalah.

4 METOD KAJIAN

Bagi membangunkan projek ini dengan lancar, penggunaan model pembangunan amatlah penting bagi memastikan projek berjalan tanpa sebarang masalah. Metod yang sesuai digunakan adalah metod Model SDLC. Ia juga merupakan kitaran yang berulang bagi membolehkan kaedah atau fasa yang sebelumnya berulang bagi memperbaiki fungsi sedia ada yang masih mempunyai kelemahan atau kekurangan.

4.1 Fasa Perancangan

Merupakan fasa pertama atau terawal yang tertumpu kepada pengolahan masalah, memastikan berkait dengan skop projek serta mengenalpasti kepentingan dan tujuan projek dilaksanakan. Kekangan juga perlulah dikenalpasti agar dapat elak dari menganggu projek.

4.2 Fasa Analisis

Fasa ini menerangkan tentang apa yang perlu dicapai oleh sistem, jangkaan prestasi dan halangan rekabentuk sistem. Selain dari itu, analisis tentang perisian juga dijalankan untuk memastikan perisian yang digunakan adalah sesuai untuk membangunkan projek ini.

4.3 Fasa Reka Bentuk

Rekabentuk membantu dalam menentukan keperluan perkakasan dan perisian dan juga membantu dalam menentukan seni bina sistem secara keseluruhan.Bagi membina laman web semantic, perisian Eclipse Java EE Developer digunakan memandangkan bagi membina laman web semantik, aplikasi Java harus dibina melalui penggunaan Apache Jena untuk mengakses ontologi dan membaca data dari ontologi.

4.4 Fasa Pengujian

Ini merupakan fasa di mana sistem perlu diuji bagi tujuan merekodkan apa yang serba kurang bagi memastikan sistem tidak ralat dan berfungsi seperti dirancang. Banyak faktor yang perlu difikirkan bagi memastikan web yang dibina digelar Muzium Web Semantik seperti seberapa banyak artifak yang perlu ada dan bagaimana penggunaan Semantik dalam Muzium Web Semantik ini.

Berikut merupakan perkakasan dan perisian yang digunakan bangi membangunkan web ini:

Jenis	Item	Tujuan	Spesifikasi
Perkakasan	Komputer Riba	Digunakan untuk membangunkan aplikasi	Macbook Pro 2017 Intel Core i5 macOS Mojave 2.3GHz Intel 8GB ROM
Perisian	Jena Apache	Digunakan untuk menghubungkan ontologi kepada server	
	Eclipse IDE for Java EE Developers	Digunakan untuk membuat aplikasi Java EE dan Web, termasuk Java IDE, alat untuk Java EE, JPA, JSF, Mylyn, EGit dan lain-lain.	
	Tomcat Apache	m\Melaksanakan beberapa spesifikasi Java EE termasuk Jawa Servlet, JavaServer Pages, Java EL, dan WebSocket, dan menyediakan persekitaran pelayan web "tulen Java" HTTP di mana kode Java dapat dijalankan.	

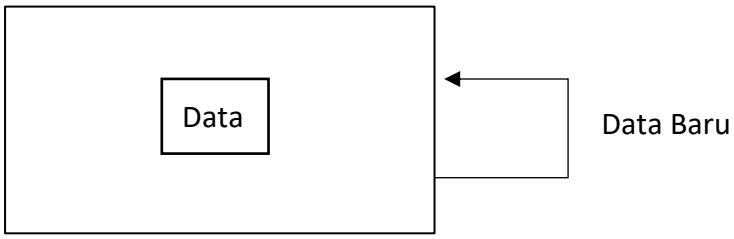
Rajah 1, Keperluan Perkakasan dan Perisian

5 HASIL KAJIAN

Hasil kajian membincangkan konklusi dan hasil kepada proses pembangunan muzium semantik ini. Penerangan yang teliti tentang pembangunan muzium semantic dibincanng. Ia merupakan fasa yang paling penting. Muzium Web Semantik ini terdiri kepada dua jenis komponen iaitu komponen Web Warisan Kebangsaan yang mempunyai komponen asas web

seperti laman muka mengenai kami,galeri kepada artifak- artifak yang tersedia dan laman muka carian. Galeri dibahagikan kepada beberapa kategori untuk memudahkan penggunaan orang ramai untuk mengakses koleksi tertentu. Pembangunan kepada bahagian komponen ini menggunakan HTML, Bootstrap, Javascript dan CSS untuk membina antara muka yang responsif dan menarik perhatian.

Untuk bahagian disebalik tadbir bahagian carian, kita memfokuskan kepada penggunaan ontologi sebagai '*relational database*' untuk menjadikan web ini Web Semantik. Web Semantik menggunakan kelebihan yang terdapat pada skala web sedia ada, kepelbagaian dan pengagihan dengan menetapkan piawaian yang bersesuaian untuk menyatakan maklumat. Web Semantik perlulah meletakkan data di bahagian tengah seperti Rajah 2. Semantik perlulah terletak di bahagian data dan tidak ditinggalkan untuk tafsiran pengguna ataupun diletak di dalam bahagian pengekodan. Aplikasi Web Semantik perlu berkongsi dan mengakses sumber data yang terdapat di internet. Aplikasi web semantik perlulah cukup dinamik untuk memahami perubahan kontent dan struktur informasi untuk menangani keperluan dan matlamat organisasi semasa.



Rajah 2, Aplikasi Web Semantik

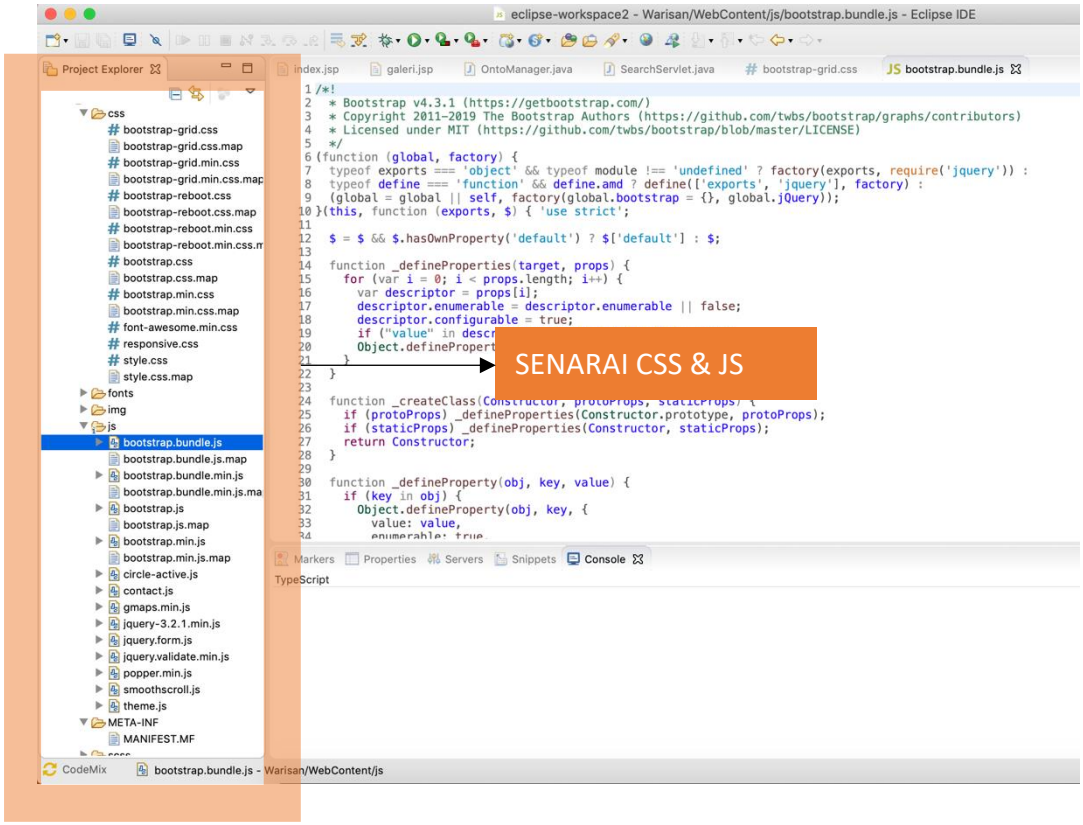
Disebabkan projek yang kita ingin bangunkan adalah berasaskan web, jadi pengekodan teramatlah penting dalam setiap aspek ketika membangunakan Muzium Web Semantik ini. Pengekodan dilakukan dari awal hingga akhir untuk:

i. Membina antaramuka yang mesra pengguna

ii. Mmebina capaian maklumat (cara menghubungkan ontologi kepada web)

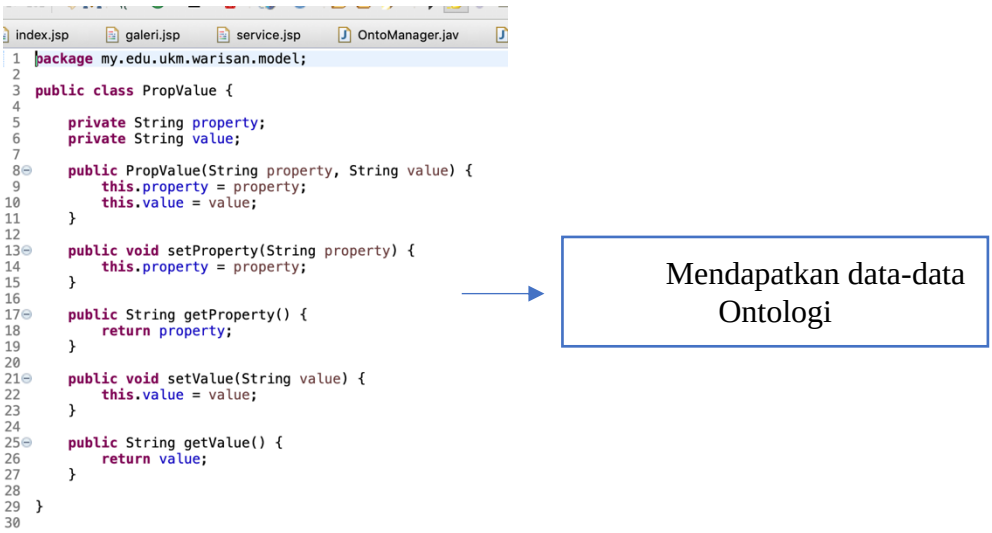
iii. Membina bahagian galeri.

Bagi pembinaan antaramuka, bahasa pengekodan CSS, SCSS dan JS digunakan untuk mendapatkan hasil yang lebih menarik ditambah melalui beberapa aksi animasi selain menjadikan web lebih responsif. Berikut merupakan contoh pengekodan :



Rajah 3, Pengekodan Antara Muka

Bagi bahagian capaian maklumat, bahagian pengekodan dia terbahagi kepada beberapa cara iaitu dengan memastikan web sudah pun disambung ke Apache Tomcat agar web boleh membaca aplikasi Java. Penggunaan Jena Apache untuk mengakses dan membaca data dari Ontologi menjadikan web ini web berasaskan aplikasi Java.Berikut merupakan proses - proses untuk mendapatkan capaian maklumat:



Rajah 4, Pengekodan untuk fungsi kepada capaian maklumat

Bahagian pengekodan dibawah terdiri daripada pengekodan SPARQL memandangkan data Ontologi hanya boleh dibaca melaui pengekodan SPARQL dan juga pengekodan untuk membaca kueri. Berikut merupakan cara-cara mendapatkan data tersebut:

Rajah 5, Pengekodan untuk fungsi kepada capaian maklumat

Urutan adalah seperti berikut :

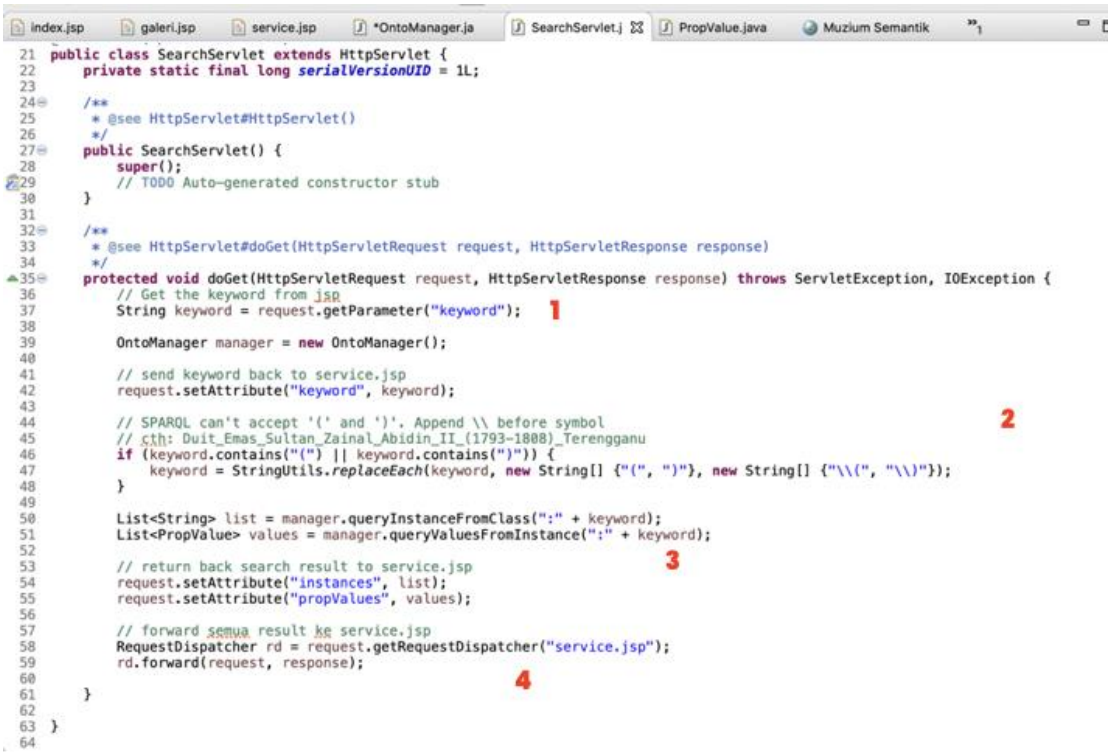
1. Senarai ontologi yang sedia ada di web termasuk Ontologi Warisan Budaya.

2. Baca data dari ontologi.

3. Kueri SPARQL untuk tujuan membaca segala jenis dan kelas dari Ontologi kecuali sifat objek.

4. Pengekodan untuk membaca kueri dan mendapatkan capaian.

Seterusnya, merupakan pengekodan untuk memaparkan hasil carian kueri yang diminta oleh pengguna ke web semula:




```
public class SearchServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public SearchServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // Get the keyword from jsp
        String keyword = request.getParameter("keyword");

        OntoManager manager = new OntoManager();

        // send keyword back to service.jsp
        request.setAttribute("keyword", keyword);

        // SPARQL can't accept '(' and ')'. Append \\ before symbol
        // cth: Duit_Emas_Sultan_Zainal_Abidin_II_(1793-1808)_Terengganu
        if (keyword.contains("(") || keyword.contains(")")) {
            keyword = StringUtils.replaceEach(keyword, new String[] {"(", ")"}, new String[] {"\\(", "\\)"});
        }

        List<String> list = manager.queryInstanceFromClass(":" + keyword);
        List<PropValue> values = manager.queryValuesFromInstance(":" + keyword);

        // return back search result to service.jsp
        request.setAttribute("instances", list);
        request.setAttribute("propValues", values);

        // forward semua result ke service.jsp
        RequestDispatcher rd = request.getRequestDispatcher("service.jsp");
        rd.forward(request, response);

    }
}
```

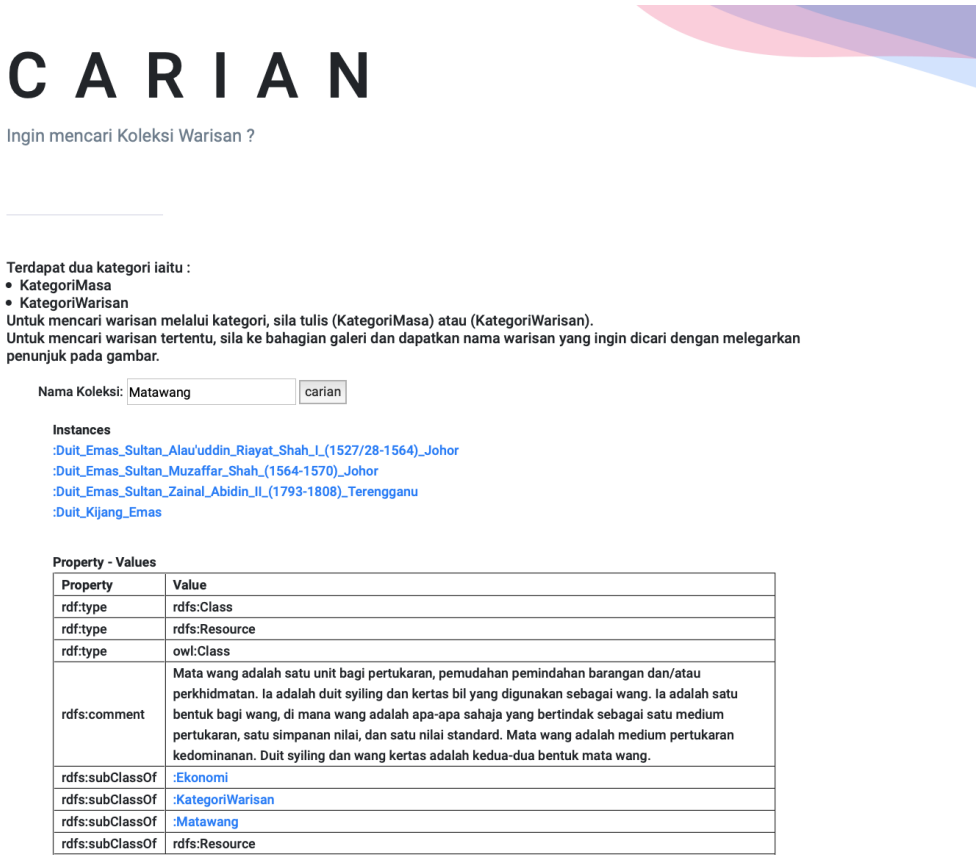
Rajah 6, Pengekodan untuk fungsi kepada capaian maklumat

Berikut merupakan urutan :

1. Mendapatkan kueri.
2. Memberitahu bahawa data yang dicapai yang mempunyai karakter yang tidak diketahui seperti (‘) dan (_) yang terdapat dalam ontologi boleh difahami oleh mesin .
3. Meminta data antara objek dan nilai .
4. Memaparkan hasil ke laman carian.

Contoh untuk melakukan carian : **MATAWANG**

Untuk memberi gambaran yang lebih jelas, kita akan membuat contoh carian kepada kelas Matawang. Memandangkan ontologi merupakan kes sensitif, jadi pengguna perlu masukkan carian seperti teks yang di-*hover* iaitu teks yang sama dalam ontologi sendiri.



Rajah 7, Contoh carian matawang.

Rajah di atas merupakan contoh untuk mencari maklumat mengenai Matawang. Sudah dimaklum di bahagian atas carian agar pengguna dapatkan nama warisan dengan melegarkan

penunjuk pada setiap gambar dan isi ruang carian sebagaimana nama warisan ditulis di setiap gambar.

```
<div class="search-area">
    <%
        String keyword = (String) request.getAttribute("keyword");
        List<String> instances = (ArrayList<String>) request.getAttribute("instances");
        List<PropValue> propValues = (ArrayList<PropValue>) request.getAttribute("propValues");

        if (keyword == null) {
            keyword = "";
        }

        if (instances == null) {
            instances = new ArrayList<String>();
        }

        if (propValues == null) {
            propValues = new ArrayList<PropValue>();
        }
    %>

    <div class="search-container">
        <div class="search-form">
            <form name="searchForm" action="SearchServlet" method="get">
                Nama Koleksi: <input type="text" name="keyword" value="<%=keyword %>" />
                <input type="submit" value="carian"/>
            </form>
        </div>

        <div id="instance-result" class="result-box">
            <div class="result-box-title">Instances</div>

            <% if (instances.isEmpty()) { %>
                Tiada "Instance"
            <% } else { %>
                <div>
                <%
                    for (String instance: instances) {
                %>
                        <a href="SearchServlet?keyword=<%=StringUtils.removeStart(instance, ":") %>"><%=instance %></a></br>
                <%
                    }
                %>
                </div>
            <% } %>
        </div>
        <div id="property-values-result" class="result-box">
            <div class="result-box-title">Property - Values</div>
            <table border="1">
                <tr>
                    <th>Property</th>
                    <th>Value</th>
                </tr>
                <%
                    for (PropValue propValue: propValues) {
                %>
                    <tr>
                        <td><%=propValue.getProperty() %></td>
                        <td>
                        <% if (propValue.getValue().startsWith(":")) { %>
                            <a href="SearchServlet?keyword=<%=StringUtils.removeStart(propValue.getValue(), ":") %>"><%=propValue.getValue() %></a>
                        <% } else { %>
                            <%=propValue.getValue() %>
                        <% } %>
                        </td>
                    </tr>
                <% } %>
            </table>

        </div>
    </div>
</div>
```

Rajah 8, Pengekodan di bahagian laman carian

Rajah di atas menunjukkan pengekodan di bahagian laman carian. Di sini, kita akan lihat di bahagian atas sekali adalah untuk mengisytiharkan bahawa perkataan *keyword* adalah teks Str*ing* dan queri *keyword* yang bakal dimasukkan perlulah dibuat carian di array instance dan propvalues. Dalam kelas search-form merupakan bagaimana kueri *keyword* itu kemudiannya dibuat carian manakala *instance-results* dan *property-value results* memaparkan senarai contoh kepada jenis dan apa ciri-ciri yang terhubung sekali dengan kueri *keyword* yang dimasukkan.

```java
package my.edu.ukm.warisan;

import java.io.IOException;

/**
 * Servlet implementation class SearchServlet
 */
@WebServlet("/SearchServlet")
public class SearchServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public SearchServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // Get the keyword from jsp
        String keyword = request.getParameter("keyword");

        OntoManager manager = new OntoManager();

        // send keyword back to service.jsp
        request.setAttribute("keyword", keyword);

        // SPARQL can't accept '(' and ')'. Append \\ before symbol
        // cth: Duit_Emas_Sultan_Zainal_Abidin_II_(1793-1808)_Terengganu
        if (keyword.contains("(") || keyword.contains(")")) {
            keyword = StringUtils.replaceEach(keyword, new String[] {"(", ")"}, new String[] {"\\(", "\\)"});
        }

        List<String> list = manager.queryInstanceFromClass(":" + keyword);
        List<PropValue> values = manager.queryValuesFromInstance(":" + keyword);

        // return back search result to service.jsp
        request.setAttribute("instances", list);
        request.setAttribute("propValues", values);

        // forward semua result ke service.jsp
        RequestDispatcher rd = request.getRequestDispatcher("service.jsp");
        rd.forward(request, response);

    }

}
```

Rajah 9, Pengekodan di bahagian carian

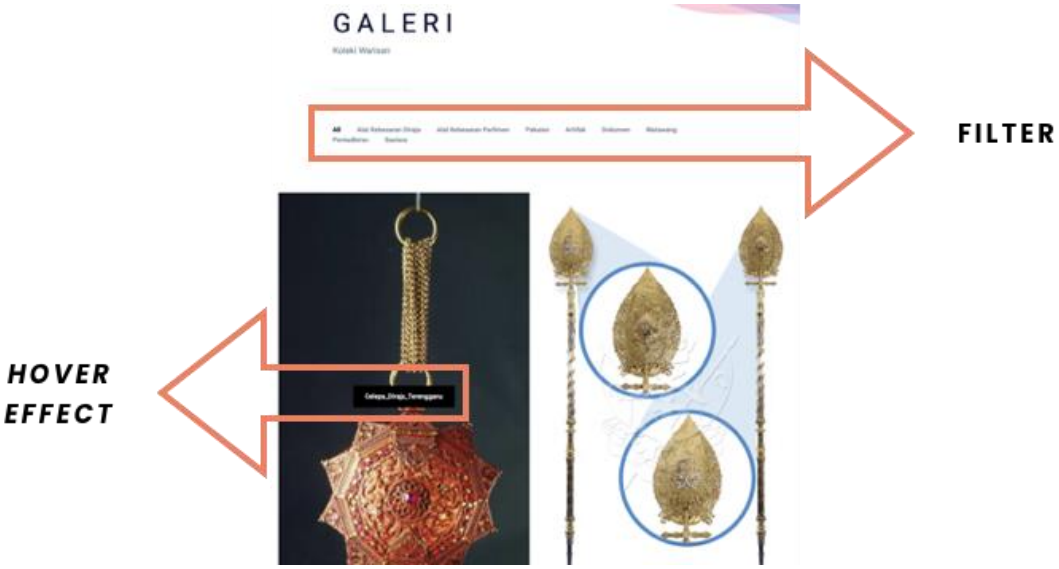
Rajah di atas merupakan sambungan kepada rajah sebelumnya dimana kueri keyword itu pula diproses supaya tidak terkeliru. Karakter (_) dan (‘) tidak boleh difahami oleh SPARQL oleh itu kita perlulah menukarkannya ke ruang kosong antara dua perkataan. OntoManager adalah manager kepada ontologi dan disinilah berlakunya perkongsian maklumat dari OntoManager ke bahagian carian maklumat. Setelah selesai, ia akan menghantar semula segala data dan maklumat ke bahagian pengekodan laman carian.

```
 */
public List<PropValue> queryValuesFromInstance(String keyword) {
    List<PropValue> propValues = new ArrayList<PropValue>();

    String sparql = PREFIXES + "SELECT DISTINCT ?property ?value " + "WHERE " + "{  " + keyword
            + " ?property ?value " + "FILTER (?property != owl:topDataProperty) "
            + "} orderBy ?property";

    QueryExecution qe = null;

    try {
        Query query = QueryFactory.create(sparql);

        qe = QueryExecutionFactory.create(query, model);

        // Use rewindable result set instead of normal resultset. Normal resultset can't
        // go back to the first result. Use reset() to rewind the resultset to the beginning
        ResultSetRewindable resultRewindable = ResultSetFactory.makeRewindable(qe.execSelect());

        while (resultRewindable.hasNext()) {
            QuerySolution querySolution = (QuerySolution) resultRewindable.next();

            OntProperty prop = model.createOntProperty(querySolution.getResource("property").getURI());

            RDFNode rdfNode = querySolution.get("value");

            // check if the value is Literal or URI Resource, because we handle it differently
            if (rdfNode.isLiteral()) {
                propValues.add(new PropValue(model.shortForm(prop.getURI()), String.valueOf(querySolution.getLiteral("value").getValue())));
            } else if (rdfNode.isURIResource()) {
                propValues.add(new PropValue(model.shortForm(prop.getURI()), model.shortForm(querySolution.getResource("value").getURI())));
            }

        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        if (qe != null) {
            qe.close();
            System.gc();
        }
    }

    return propValues;
}
```

Rajah 10, Pengekodan SPARQL di bahagian

Rajah di atas merupakan pengekodan SPARQL untuk mendapatkan data dari Ontologi. Mula-mula sekali kita perlu mengisytiharkan senarai ontologi yang terlibat jika kita mempunyai data yang dimiliki oleh berbagai ontologi dan kemudiannya kita perlu memasukkan pengekodan SPARQL memandangkan SPARQL adalah antara kueri yang difahami oleh ontologi. Kemudian, data akan dihantar semula ke bahagian carian dan dihantar pula ke bahagian antaramuka carian.

Pengekodan di bahagian galeri pula memfokuskan kepada dua ciri utama iaitu:



Rajah 11, Ciri-ciri utama galeri

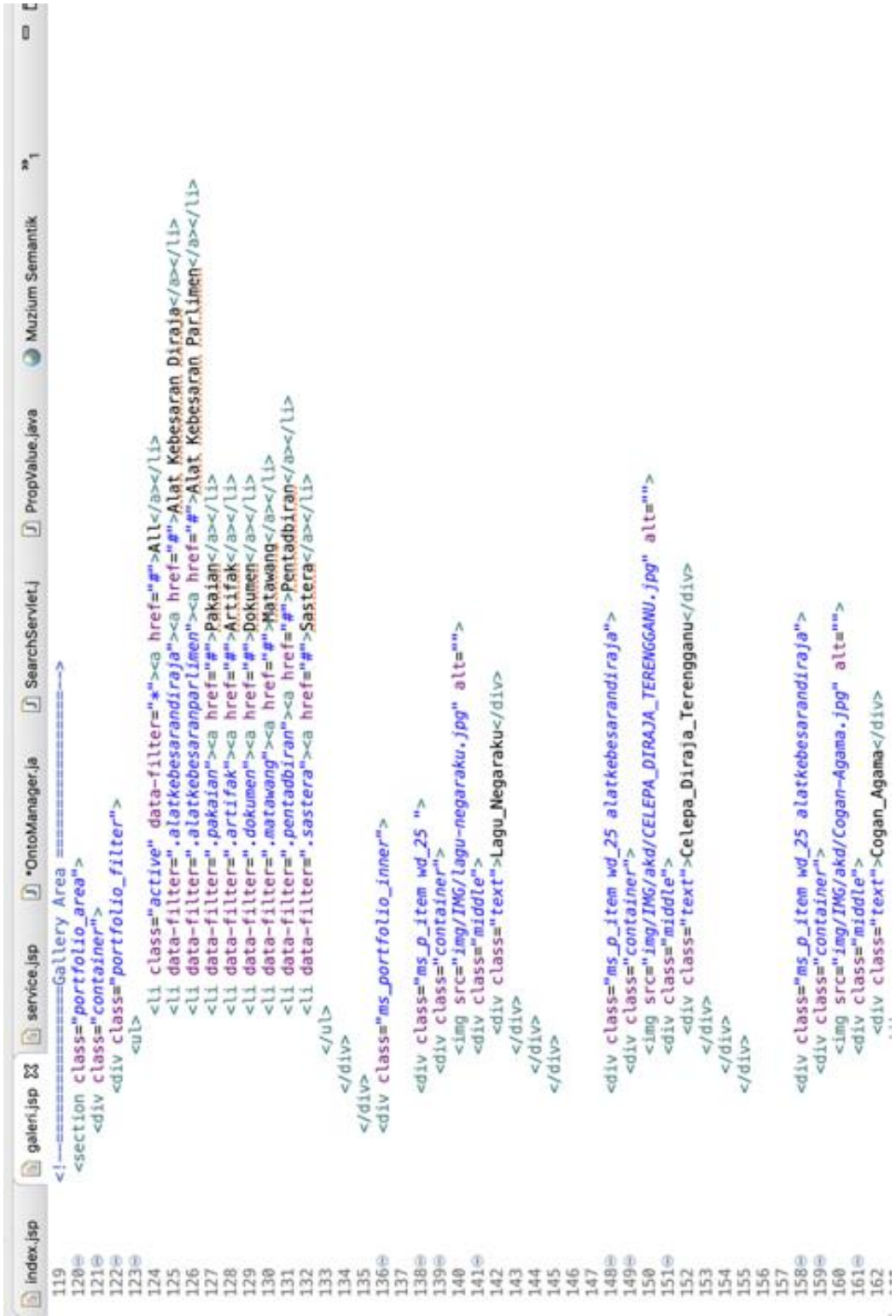
Tujuan adanya ciri filter adalah untuk mengasingkan senarai koleksi mengikut kategori tersendiri agar pengguna tidak keliru. Berikut merupakan pengekodan untuk filter:

```
/*-------------------------------------------------------------*/
/*  Galeri js
/*-------------------------------------------------------------*/
function portfolio_isotope(){
    if ( $('.portfolio_filter ul li').length ){
        // Add isotope click function
        $(".portfolio_filter ul li").on('click',function(){
            $(".portfolio_filter ul li").removeClass("active");
            $(this).addClass("active");

            var selector = $(this).attr("data-filter");
            $(".ms_portfolio_inner").isotope({
                filter: selector,
                animationOptions: {
                    duration: 450,
                    easing: "linear",
                    queue: false,
                }
            });
            return false;
        });
    }
}
```

Rajah 12a, Pengekodan JavaScript untuk Filter

```
1745 .portfolio_area {
1746   border-bottom: 2px solid #fff;
1747 }
1748
1749 .portfolio_filter {
1750   padding: 95px 0px 130px 0px;
1751 }
1752
1753 .portfolio_filter ul li {
1754   display: inline-block;
1755   margin-right: 40px;
1756 }
1757
1758 .portfolio_filter ul li a {
1759   font-size: 18px;
1760   font-family: "Roboto", sans-serif;
1761   font-weight: bold;
1762   color: #7e8d9c;
1763   -webkit-transition: all 400ms linear 0s;
1764   -o-transition: all 400ms linear 0s;
1765   transition: all 400ms linear 0s;
1766 }
1767
1768 .portfolio_filter ul li:last-child {
1769   margin-right: 0px;
1770 }
1771
1772 .portfolio_filter ul li:hover a, .portfolio_filter ul li.active a {
1773   color: #0b1033;
1774 }
1775
1776 .wd_25 {
1777   width: 25%;
1778 }
1779
1780 .wd_50 {
1781   width: 50%;
1782 }
1783
1784 .ms_portfolio_inner {
1785   overflow: hidden;
1786 }
1787
1788 .ms_portfolio_inner .ms_p_item {
```

Rajah 12b, Pengekodan CSS untuk Filter



Rajah 12c, Pengekodan HTML untuk Filter

Bagi ciri Efek Hover di mana teks hanya akan muncul apabila kursor berada di atas kawasan yang dikodkan, hanya pengekodan HTML dan CSS digunakan. Berikut merupakan sebahagian daripada pengekodan:

index.jsp | galeri.jsp | service.jsp | *OntoManager.ja | SearchServlet.j | PropValue.java | Muzium Semantik

```
<!--================Gallery Area ================-->
 <section class="portfolio_area">
     <div class="container">
         <div class="portfolio_filter">
             <ul>
                 <li class="active" data-filter="*"><a href="#">All</a></li>
                 <li data-filter=".alatkebesarandiraja"><a href="#">Alat Kebesaran Diraja</a></li>
                 <li data-filter=".alatkebesaranparlimen"><a href="#">Alat Kebesaran Parlimen</a></li>
                 <li data-filter=".pakaian"><a href="#">Pakaian</a></li>
                 <li data-filter=".artifak"><a href="#">Artifak</a></li>
                 <li data-filter=".dokumen"><a href="#">Dokumen</a></li>
                 <li data-filter=".matawang"><a href="#">Matawang</a></li>
                 <li data-filter=".pentadbiran"><a href="#">Pentadbiran</a></li>
                 <li data-filter=".sastera"><a href="#">Sastera</a></li>
             </ul>
         </div>
     </div>
     <div class="ms_portfolio_inner">

         <div class="ms_p_item wd_25 ">
           <div class="container">
             <img src="img/IMG/lagu-negaraku.jpg" alt="">
             <div class="middle">
               <div class="text">Lagu_Negaraku</div>
             </div>
           </div>
         </div>


         <div class="ms_p_item wd_25 alatkebesarandiraja">
           <div class="container">
             <img src="img/IMG/akd/CELEPA_DIRAJA_TERENGGANU.jpg" alt="">
             <div class="middle">
               <div class="text">Celepa_Diraja_Terengganu</div>
             </div>
           </div>
         </div>


         <div class="ms_p_item wd_25 alatkebesarandiraja">
           <div class="container">
             <img src="img/IMG/akd/Cogan-Agama.jpg" alt="">
             <div class="middle">
               <div class="text">Cogan_Agama</div>
```

Rajah 13a, Pengekodan HTML5 untuk *Hover Effect*

index.jsp | galeri.jsp | service.jsp | *OntoManager.ja | SearchServlet.j | PropValue.java | Muzium Semantik

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">

<style>
.container {
  position: relative;
  width: 50%;
}

.image {
  opacity: 1;
  display: block;
  width: 100%;
  height: auto;
  transition: .5s ease;
  backface-visibility: hidden;
}

.middle {
  transition: .5s ease;
  opacity: 0;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  -ms-transform: translate(-50%, -50%);
  text-align: center;
}

.container:hover .image {
  opacity: 0.3;
}

.container:hover .middle {
  opacity: 1;
}

.text {
  background-color: black;
  color: white;
  font-size: 16px;
  padding: 16px 32px;
}
```

Rajah 13b, Pengekodan CSS untuk *Hover Effect*

Setelah selesai bahagian pengekodan, pengujian pun dilakukan untuk memastikan web berfungsi dengan baik dan menepati objektif yang ditetapkan dari awal pendokumentasi. Pengujian pertama dilakukan untuk memastikan kebolehgunaan Muzium Web Semantik. Setelah itu, pengujian dilakukan agar ia menyelesaikan masalah yang timbul sebelum ini.

6 KESIMPULAN

Sistem Muzium Web Semanti kini merupakan sistem yang membantu para pengguna untuk mendapatkan maklumat dan data secara lebih terperinci dengan capaian maklumat yang lebih ringkas dan mesra pengguna. Sistem ini juga merupakan sistem yang dibangunkan untuk menambahbaikan sistem capaian maklumat yang pernah dibina sebelum ini. Tetapi sistem sebelum ini lebih memfokuskan kepada pembangunan ontologi dan bukannya sistem web semantik. Web Semantik kali ini mempunyai galeri yang menarik dan mudah dikendalikan. Galeri terdiri daripada beberapa kategori yang berkaitan dengan Warisan Malaysia. Selain itu, capaian Maklumat kini lebih mudah berbanding daripada sistem yang terdahulu.

Namun, dengan kurangnya pengalaman dan pengetahuan, pelbagai kajian dilakukan bagi menyiapkan dan memenuhi objektif projek ini.

7 RUJUKAN

Douglas McCarthy, Europeana Foundation, 2018, Hello World ! The Finnish Museum National Gallery Opens up its collections, https://pro.europeana.eu/post/hello- cc0-the-finnish-national-gallery-opens-up-its-collections

Matthew Lincoln, Programming historian, 2018-07-16, https://programminghistorian.org/en/lessons/graph-databases-and-SPARQL

Gangemi, A., N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. (2002)Sweetening ontologies with DOLCE. In: Proc. of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, Springer–Verlag (2002).

Hyvönen, E., T. Ruotsalo, T. Häggström, M. Salminen, M. Junnila, M. Virkkilä, M. Haaramo, T. Kauppinen, E. Mäkelä, K. Viljanen (2008). "CultureSampo– Finnish culture on the semantic web. The vision and first results". In: Klaus Robering (Ed.). Information Technology for the Virtual Museum. LIT Verlag. (2008a).

Hyvönen, E., K, Viljanen, J. Tuominen, K. Seppälä (2008). "Building a national semantic web ontology and ontology service infrastructure–the FinnONTO approach". In: Proceedings of the ESWC 2008. Tenerife, Spain, Springer– Verlag (2008b).

Hyvönen (2006)."Towards a National Finnish Semantic Web Infrastructure for eCulture"https://seco.cs.aalto.fi/events/2006/2006-05-04-websemantique/presentations/thursday-1140-Hyvonen.pdf

Li. Ma and Mei. A. Fokoue,"Semantic Web Technologies and Data Management" , Li Ma, Jing Mei, Yue Pan, Krishna Kulkarni , AchilleFokoue, Anand Ranganathan.2007

Urvi Shah, James Mayfield," Information Retrieval on the Semantic Web", "ACM CIKM International Conference on Information Management", Nov 2002.

Berners Lee, J.Lassila, "Ontologies in Semantic Web", "Scientific American", May (2001) 34-43.

U.Shah. T.Finin and A.Joshi. "Information Retrieval on the semantic web", "Scientific American", pages 34-43, 2003.

Kiryakov, A. Popov, L. Manov, "Semantic annotation, indexing and retrieval", "Journal of Web Semantics", 2005-2006.

David Vallet, M.Fernandes, “An Ontology-Based Information Retrieval Model”, “European Semantic Web Symposium (ESWS)”, 2006.

Su-Kyoung, Kim. 2007, “Implementation of Web Ontology for Semantic Web Application”International Conference on Advanced Language Processing and Web Information Technology,0-7695-2930-5/07 $25.00 © 2007 IEEE

Decker, S. 2000. “The semantic Web roles of XML and RDF”. IEEE Internet Comp., 4(5), 63–74.

G.Deepak, J. Sheeba, and M S HareeshBabu. 2016 “A Differential Semantic Algorithm for Query Relevant Web Page Recommendation”. IEEE International Conference on Advances in Computer Applications (ICACA).2016 IEEE.