

**TUGAS AKHIR – IF184802**

**RANCANG BANGUN APLIKASI BERBASIS WEB UNTUK VISUALISASI POHON KELUARGA TOKOH SEJARAH INDONESIA MENGGUNAKAN ONTOLOGI DBPEDIA DAN PELLET REASONER**

**FAIQ**

**NRP. 05111540000007**

**Dosen Pembimbing 1**

**Nurul Fajrin A.,S.Kom., M.Sc.**

**Dosen Pembimbing 2**

**Adhatus Solichah A.,S.Kom., M.Sc.**

**DEPARTEMEN INFORMATIKA**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2019**

2016

# Description: its_logo_gifHALAMAN JUDUL

**TUGAS AKHIR – IF184802**

**RANCANG BANGUN APLIKASI BERBASIS WEB UNTUK VISUALISASI POHON KELUARGA TOKOH SEJARAH INDONESIA MENGGUNAKAN ONTOLOGI DBPEDIA DAN PELLET REASONER**

**FAIQ**

**NRP. 05111540000007**

**Dosen Pembimbing 1**

**Nurul Fajrin A.,S.Kom., M.Sc.**

**Dosen Pembimbing 2**

**Adhatus Solichah A.,S.Kom., M.Sc.**

**DEPARTEMEN INFORMATIKA**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2019**

***(Halaman ini sengaja dikosongkan)***

Halaman Judul



**FINAL PROJECT – KI141502**

**FAMILY TREE VISUALIZATION DESIGN OF INDONESIAN HISTORY ACTORS USING DBPEDIA ONTOLOGY AND PELLET REASONER**

**FAIQ**

**NRP. 05111540000007**

**Supervisor 1**

**Nurul Fajrin A.,S.Kom., M.Sc.**

**Supervisor 2**

**Adhatus Solichah A.,S.Kom., M.Sc.**

**DEPARTMENT OF INFORMATICS**

**Faculty of Information Technology and Communication**

**Sepuluh Nopember Institute of Technology**

**Surabaya 2019**

***(Halaman ini sengaja dikosongkan)***

# 

# LEMBAR PENGESAHAN

**RANCANG BANGUN APLIKASI BERBASIS WEB UNTUK VISUALISASI POHON KELUARGA TOKOH SEJARAH INDONESIA MENGGUNAKAN ONTOLOGI DBPEDIA DAN PELLET REASONER**

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

pada

Rumpun Mata Kuliah Manajemen Informasi

Program Studi S-1 Departemen Informatika

Fakultas Teknologi Informasi dan Komunikasi

Institut Teknologi Sepuluh Nopember

**Oleh:**

**FAIQ**

**NRP. 05111540000007**

Disetujui oleh Pembimbing Tugas Akhir:

1. Nurul Fajrin A., S.Kom., M.Sc. ........................

NIP. 19860722 201504 2 003 (Pembimbing 1)

1. Adhatus Sholicah A., S.Kom., M.Sc. ........................

NIP. 19850826 201504 2 002 (Pembimbing 2)

**SURABAYA**

**JUNI, 2019**

***(Halaman ini sengaja dikosongkan)***

**RANCANG BANGUN APLIKASI BERBASIS WEB UNTUK VISUALISASI POHON KELUARGA TOKOH SEJARAH INDONESIA MENGGUNAKAN ONTOLOGI DBPEDIA DAN PELLET REASONER**

**Nama : Faiq**

**NRP : 0511540000007**

**Departemen : Informatika FTIK-ITS**

**Dosen Pembimbing I : Nurul Fajrin A.,S.Kom., M.Sc.**

**Dosen Pembimbing II : Adhatus Solichah A.,S.Kom., M.Sc.**

# ABSTRAK

*Tokoh sejarah dan kerajaan di Indonesia, menjadi bukti dari adanya suatu kejadian penting di masa lalu. Setiap tokoh memiliki rekan hidup dan keluarga yang berbeda. Salah satu platform ensiklopedia online yang menyediakan daftar pahlawan nasional Indonesia adalah Wikipedia. Konten dari sebuah halaman Wikipedia memiliki keterkaitan dengan DBpedia dimana DBpedia menyediakan daftar hyperlink yang memiliki keterkaitan dengan halaman Wikipedia tersebut, seperti orang tua, pasangan dan anak cucu.. Namun, seringkali halaman Wikipedia merepresentasikan data tersebut sebagai paragraf, dan halaman DBpedia sebagai tabel.*

*Dengan adanya data keluarga dari suatu halaman DBpedia, hubungan kekeluargaan tokoh sejarah dan kerajaan Indonesia dapat diketahui. Keterkaitan atau relasi tokoh bersejarah dapat digambarkan dengan ontologi. Tujuan dari pengerjaan tugas akhir ini adalah untuk melengkapi data dan relasi keluarga kerajaan di Indonesia dan merepresentasikannya dalam bentuk pohon keluarga.*

*Langkah-langkah dari pengerjaan tugas akhir ini, pertama-tama melengkapi data tokoh dengan proses reasoning lalu menyimpan data tersebut dalam suatu basis data sehingga bisa ditampilkan secara grafis hubungan keluarga tokoh sejarah dan kerajaan dalam bentuk pohon keluarga. Untuk melengkapi data keluarga, menggabungkan dan menjalankan proses reasoning pada model ontologi dengan data DBpedia sudah terbukti dapat menghasilkan fakta-fakta baru yang belum tercatat dalam DBpedia. Untuk penyimpanan data, Apache Jena-Fuseki dapat menjadi server basis data triple store. Berdasarkan uji coba yang dilakukan, aplikasi berbasis web ini dapat menampilkan pohon keluarga suatu tokoh dan lebih lengkap relasinya dibandingkan dengan DBpedia. Tugas Akhir ini dapat membantu penelitian sejarah dalam menentukan hubungan keluarga dari suatu tokoh sejarah. Hal ini dapat menambah wawasan sejarah bangsa Indonesia terhadap para pelaku sejarah beserta keluarganya.*

***Kata kunci: Keluarga kerajaan Indonesia, Ontologi, Pohon keluarga, Tokoh Sejarah, Visualisasi.***

**FAMILY TREE VISUALIZATION DESIGN OF INDONESIAN HISTORY ACTORS USING DBPEDIA ONTOLOGY AND PELLET REASONER**

**Name : Faiq**

**NRP : 05111540000007**

**Department : Informatics FTIK-ITS**

**First Advisor : Nurul Fajrin A.,S.Kom., M.Sc.**

**Second Advisor : Adhatus Solichah A.,S.Kom., M.Sc.**

# ABSTRACT

*Historical and monarch figures of Indonesia, are both proofs of important events in our history. Every figure has different partners and relatives. One of the open encyclopedia platform is Wikipedia. The pages or subjects of a Wikipedia page has a direct association with DBpedia page, whereas DBpedia provides list of hyperlinks of related things of a Wikipedia subject as table rows, such as parents, partners, and children. But sometimes a Wikipedia page represesents the data as paragraphs and DBpedia as table rows.*

*From a DBpedia page, we can get information of a person’s family and relations. This Wikipedia hyperlink relation can be modelled as an ontology. The purpose of this thesis is to complete the family data of the Indonesia’s historical and monarch figures and to represent them as a family tree.*

*The steps required to complete this thesis is first completing the figure’ data using reasoning process, store the data on a triple store database, and to display the information in a family tree graph. To complete the family data of a person, Family Relationship Ontology by Robert Stevens is used and combined with the DBpedia page and reasoned using Pellet Reasoner. It is proven that this method generates facts that are unknown to DBpedia page. To store the data, Apache Jena-Fuseki can act as a triple store database. According to test results, this web application is able to display family tree of a DBpedia subject and the relations are more complete than its DBpedia page. This thesis can help history scientist to determine the family tree of a historical figure. This thesis is also capable to educate people about Indonesia’s historical figures and their relations.*

***Key words: Family tree, Indonesian history, Indonesian kingdoms, Ontology, Visualization***

# KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT karena atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan tugas akhir yang berjudul:

**“Rancang Bangun Aplikasi Berbasis Web untuk Visualisasi Family Tree Tokoh Sejarah Indonesia Menggunakan Ontologi DBpedia dan Pellet Reasoning”**

Tugas akhir ini dilakukan untuk memenuhi salah satu syarat memperoleh gelar Sarjana Komputer di Jurusan Teknik Informatika Fakultas Teknologi Informasi dan Komunikasi Institut Teknologi Sepuluh Nopember.

Penulis mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan baik secara langsung maupun tidak langsung selama proses pengerjaan tugas akhir ini hingga selesai, antara lain:

1. Allah SWT atas segala karunia dan rahmat-Nya yang telah diberikan selama ini.
2. Orang tua, saudara serta keluarga penulis yang tiada henti-hentinya memberikan semangat, perhatian dan doa selama perkuliahan penulis di Jurusan Teknik Informatika ini.
3. Ibu Nurul Fajrin A., S.Kom., M.Sc. selaku dosen pembimbing I yang telah memberikan bimbingan dan arahan dalam pengerjaan tugas akhir ini.
4. Ibu Adhatus Sholichah A., S.Kom., M.Sc. selaku dosen pembimbing II yang telah banyak memberikan arahan dan bantuan, waktu untuk berdiskusi serta ilmu-ilmu baru sehingga penulis dapat menyelesaikan tugas akhir ini.
5. Segenap dosen Departemen Informatika ITS yang telah memberikan ilmu dalam kuliah-kuliah saya.
6. Firda Rheinalia, S.Kom yang selalu memberikan semangat dan informasi terkait pengerjaan Tugas Akhir.
7. Sahabat-sahabat Rumah Perjuangan, Fatur, Illham, Ichsan, Huda, Bimo, Dias, Azka, Adam dan Djohan, serta Tegar dan Arya.
8. Teman-teman HMTC 2016/2017 dan BEM FTIK 2016/2017 - 2017/2018.
9. Seluruh keluarga TC 2015 yang selalu menemani dan memberi semangat selama 4 tahun perkuliahan.
10. Serta semua pihak yang yang telah memberikan dukungan selama penulis menyelesaikan tugas akhir ini.

Saya mohon maaf apabila terdapat kekurangan dalam penulisan buku tugas akhir ini. Kritik dan saran saya harapkan untuk perbaikan dan pembelajaran di kemudian hari. Semoga tugas akhir ini dapat memberikan manfaat yang sebaik-baiknya.

Surabaya, Juni 2019

Penulis

# DAFTAR ISI

[HALAMAN JUDUL iii](#_Toc12492996)

[LEMBAR PENGESAHAN vii](#_Toc12492997)

[ABSTRAK ix](#_Toc12492998)

[ABSTRACT xi](#_Toc12492999)

[KATA PENGANTAR xiii](#_Toc12493000)

[DAFTAR ISI xv](#_Toc12493001)

[DAFTAR GAMBAR xix](#_Toc12493002)

[DAFTAR TABEL xxi](#_Toc12493003)

[DAFTAR KODE SUMBER xxiii](#_Toc12493004)

[1 BAB I PENDAHULUAN 1](#_Toc12493005)

[1.1. Latar Belakang 1](#_Toc12493006)

[1.2. Rumusan Masalah 2](#_Toc12493007)

[1.3. Batasan Masalah 2](#_Toc12493008)

[1.4. Tujuan 3](#_Toc12493009)

[1.5. Metodologi 3](#_Toc12493010)

[1.6. Sistematika Penulisan 4](#_Toc12493011)

[2 BAB II DASAR TEORI 7](#_Toc12493012)

[2.1. Kerajaan Indonesia 7](#_Toc12493013)

[2.2. Ontologi 7](#_Toc12493014)

[2.3. DBpedia 10](#_Toc12493015)

[2.4. Semantic Web Rule Language (SWRL) 11](#_Toc12493016)

[2.5. Family Relationships Ontology 12](#_Toc12493017)

[2.6. SPARQL 13](#_Toc12493018)

[2.7. Apache Jena Fuseki 14](#_Toc12493019)

[2.8. Pellet Reasoner 16](#_Toc12493020)

[2.9. SPARQL Lib 17](#_Toc12493021)

[2.10. Apache Jena 18](#_Toc12493022)

[3 BAB III METODOLOGI PEMECAHAN MASALAH 21](#_Toc12493023)

[3.1. Analisis Data 22](#_Toc12493024)

[3.1.1. Analisis Data dari DBpedia 23](#_Toc12493025)

[3.2. Ekstraksi Data Sebagai Model 24](#_Toc12493026)

[3.3. Pembuatan Ontologi 25](#_Toc12493027)

[3.4. Penggabungan model data dan model ontologi 26](#_Toc12493028)

[3.5. Reasoning pada Model Gabungan 27](#_Toc12493029)

[3.6. Penampilan Data 27](#_Toc12493030)

[4 BAB IV ANALISIS DAN PERANCANGAN SISTEM 29](#_Toc12493031)

[4.1. Analisis 29](#_Toc12493032)

[4.1.1. Cakupan Permasalahan 29](#_Toc12493033)

[4.1.2. Deskripsi Umum Sistem 29](#_Toc12493034)

[4.1.3. Spesifikasi Kebutuhan Perangkat Lunak 37](#_Toc12493035)

[4.1.4. Aktor 37](#_Toc12493036)

[4.1.5. Kasus Penggunaan 37](#_Toc12493037)

[4.2. Perancangan Antarmuka Pengguna 40](#_Toc12493038)

[5 BAB V IMPLEMENTASI 42](#_Toc12493039)

[5.1. Implementasi Proses Ekstraksi, Penggabungan, dan Reasoning 42](#_Toc12493040)

[5.2. Implementasi Antarmuka Pohon Keluarga 45](#_Toc12493041)

[5.2.1. Fungsi Dropdown Select 45](#_Toc12493042)

[5.2.2. Fungsi Get Family 47](#_Toc12493043)

[5.3. Implementasi Antarmuka Pengguna 62](#_Toc12493044)

[5.3.1. Implementasi Tampilan Halaman Utama 62](#_Toc12493045)

[5.3.2. Implementasi Tampilan Halaman Pohon Keluarga 63](#_Toc12493046)

[6 BAB VI PENGUJIAN DAN EVALUASI 64](#_Toc12493047)

[6.1. Lingkungan Pengujian 64](#_Toc12493048)

[6.2. Skenario Pengujian 64](#_Toc12493049)

[6.2.1. Pengujian Perbandingan Data 65](#_Toc12493050)

[6.3. Evaluasi Pengujian 94](#_Toc12493051)

[6.3.1. Evaluasi Pengujian Perbandingan Data 95](#_Toc12493052)

[7 BAB VII KESIMPULAN DAN SARAN 98](#_Toc12493053)

[**1.** 98](#_Toc12493054)

[7.1. Kesimpulan 98](#_Toc12493055)

[7.2. Saran 98](#_Toc12493056)

[DAFTAR PUSTAKA 100](#_Toc12493057)

[LAMPIRAN 1. DATA DBPEDIA 102](#_Toc12493058)

[1 102](#_Toc12493059)

[1.1. Mark Phillips 102](#_Toc12493060)

[1.2. Prince Andrew 102](#_Toc12493061)

[1.3. Meghan Markle 103](#_Toc12493062)

[1.4. Princess Diana 103](#_Toc12493063)

[1.5. Sophie, Countess of Wessex 104](#_Toc12493064)

[1.6. Prince Philip 104](#_Toc12493065)

[1.7. Sarah, Duchess of York 105](#_Toc12493066)

[1.8. Prince William 106](#_Toc12493067)

[1.9. Princess Anne 106](#_Toc12493068)

[1.10. Autumn Phillips 107](#_Toc12493069)

[1.11. Prince Charles 107](#_Toc12493070)

[1.12. Camilla Rosemary 108](#_Toc12493071)

[1.13. Princess Charlotte 108](#_Toc12493072)

[1.14. Elizabeth II 109](#_Toc12493073)

[1.15. Prince Harry 110](#_Toc12493074)

[1.16. Catherine Middleton 110](#_Toc12493075)

[1.17. Prince Edward 111](#_Toc12493076)

[1.18. Zara Phillips 112](#_Toc12493077)

[1.19. Prince George 112](#_Toc12493078)

[1.20. Mike Tindall 113](#_Toc12493079)

[1.21. Princess Eugenie 113](#_Toc12493080)

[1.22. Lady Louise Windsor 114](#_Toc12493081)

[1.23. Peter Phillips 114](#_Toc12493082)

[1.24. Timothy Laurence 115](#_Toc12493083)

[1.25. Princess Beatrice 115](#_Toc12493084)

[1.26. James, Viscount Severn 115](#_Toc12493085)

[LAMPIRAN 2. DATA DALAM FAMILY TREE APP 118](#_Toc12493086)

[1 118](#_Toc12493087)

[2 118](#_Toc12493088)

[2.1. Mark Phillips 118](#_Toc12493089)

[2.2. Prince Andrew 118](#_Toc12493090)

[2.3. Meghan Markle 119](#_Toc12493091)

[2.4. Princess Diana 119](#_Toc12493092)

[2.5. Sophie, Countess of Wessex 120](#_Toc12493093)

[2.6. Prince Philip 120](#_Toc12493094)

[2.7. Sarah, Duchess of York 122](#_Toc12493095)

[2.8. Prince William 122](#_Toc12493096)

[2.9. Princess Anne 123](#_Toc12493097)

[2.10. Autumn Phillips 124](#_Toc12493098)

[2.11. Prince Charles 124](#_Toc12493099)

[2.12. Camilla Rosemary 125](#_Toc12493100)

[2.13. Princess Charlotte 125](#_Toc12493101)

[2.14. Elizabeth II 126](#_Toc12493102)

[2.15. Prince Harry 127](#_Toc12493103)

[2.16. Catherine Middleton 128](#_Toc12493104)

[2.17. Prince Edward 128](#_Toc12493105)

[2.18. Zara Phillips 129](#_Toc12493106)

[2.19. Prince George 130](#_Toc12493107)

[2.20. Mike Tindall 130](#_Toc12493108)

[2.21. Princess Eugenie 131](#_Toc12493109)

[2.22. Lady Louise Windsor 131](#_Toc12493110)

[2.23. Peter Phillips 132](#_Toc12493111)

[2.24. Timothy Laurence 132](#_Toc12493112)

[2.25. Princess Beatrice 133](#_Toc12493113)

[2.26. James, Viscount Severn 133](#_Toc12493114)

[BIODATA PENULIS 136](#_Toc12493115)

# DAFTAR GAMBAR

[Gambar 2.1 Class Hierarchy 8](#_Toc12492965)

[Gambar 2.2 Property 9](#_Toc12492966)

[Gambar 2.3 Class, Property dan Instance 9](#_Toc12492967)

[Gambar 2.4 Contoh Halaman DBpedia 10](#_Toc12492968)

[Gambar 2.5 Ontologi FamilyTree Keluarga Robert Stevens 13](#_Toc12492969)

[Gambar 2.6 Contoh SPARQL Query 14](#_Toc12492970)

[Gambar 2.7 Database Triple Store Apache Jena Fuseki 14](#_Toc12492971)

[Gambar 2.8 Daftar API Apache Jena Fuseki 15](#_Toc12492972)

[Gambar 2.9 Arsitektur Pellet Reasoner 16](#_Toc12492973)

[Gambar 2.10 Contoh Penggunaan SPARQL Lib 18](#_Toc12492974)

[Gambar 3.1 Flowchart pengembangan sistem 21](#_Toc12492975)

[Gambar 3.2 Halaman DBpedia tentang properti keluarga Raden Wijaya 23](#_Toc12492976)

[Gambar 3.3 Hirarki Class 25](file:///D:\The-Tree-of-Heroes\5115100007-Faiq-BukuTA.docx#_Toc12492977)

[Gambar 3.4 Hirarki Data property 25](file:///D:\The-Tree-of-Heroes\5115100007-Faiq-BukuTA.docx#_Toc12492978)

[Gambar 3.5 Hirarki Object property 26](#_Toc12492979)

[Gambar 3.6 Silsilah keluarga kerajaan Singasari dan Majapahit [11] 28](#_Toc12492980)

[Gambar 4.1 Output file Hayam\_Wuruk.rdf bisa dilihat dengan tools Protégé. 32](#_Toc12492981)

[Gambar 4.2 Ontologi Hayam Wuruk (Individuals) 33](#_Toc12492982)

[Gambar 4.3 Ontologi Hayam Wuruk (Object Properties) 33](#_Toc12492983)

[Gambar 4.4 Ontologi Family Tree (Individuals) 34](#_Toc12492984)

[Gambar 4.5 Ontologi Family Tree (Object Properties) 34](#_Toc12492985)

[Gambar 4.6 Ontologi gabungan (Individuals) 35](#_Toc12492986)

[Gambar 4.7 Ontologi gabungan (Object Properties) 35](#_Toc12492987)

[Gambar 4.8 Individu Susilo Bambang Yudhoyono sebelum reasoning 36](#_Toc12492988)

[Gambar 4.9 Individu Susilo Bambang Yudhoyono setelah reasoning 36](#_Toc12492989)

[Gambar 4.10 Diagram Kasus Penggunaan Sistem 38](#_Toc12492990)

[Gambar 4.11 Diagram Aktivitas Melihat Pohon Keluarga Tokoh 40](#_Toc12492991)

[Gambar 4.12 Antarmuka Halaman Utama Family Tree App 41](#_Toc12492992)

[Gambar 4.13 Antarmuka Halaman Pohon Keluarga Family Tree App 41](#_Toc12492993)

[Gambar 5.1 Implementasi Antarmuka Halaman Utama 63](#_Toc12492994)

[Gambar 5.2 Implementasi Antarmuka Halaman Pohon Keluarga 63](#_Toc12492995)

# DAFTAR TABEL

[Tabel 2.1 Komponen SWRL 12](#_Toc12318998)

[Tabel 4.1 Daftar Kebutuhan Fungsional Perangkat Lunak 29](#_Toc12318999)

[Tabel 4.2 Daftar Kode Diagram Kasus Penggunaan 30](#_Toc12319000)

[Tabel 4.4 Spesifikasi Kasus Penggunaan Melihat Informasi Tokoh 31](#_Toc12319001)

[Tabel 4.6 Spesifikasi Atribut Rancangan Antarmuka Halaman Family Tree App 33](#_Toc12319002)

[Tabel 6.1 Pengujian Perbandingan Data Mark Phillips 57](#_Toc12319003)

[Tabel 6.2 Pengujian Perbandingan Data Prince Andrew, Duke of York 58](#_Toc12319004)

[Tabel 6.3 Pengujian Perbandingan Data Meghan Markle 59](#_Toc12319005)

[Tabel 6.4 Pengujian Perbandingan Data Diana, Princess of Wales 61](#_Toc12319006)

[Tabel 6.5 Pengujian Perbandingan Data Sophie, Countess of Wessex 62](#_Toc12319007)

[Tabel 6.6 Pengujian Perbandingan Data Prince Philip, Duke of Edinburgh 63](#_Toc12319008)

[Tabel 6.7 Pengujian Perbandingan Data Sarah, Duchess of York 64](#_Toc12319009)

[Tabel 6.8 Pengujian Perbandingan Data Prince William, Duke of Cambridge 65](#_Toc12319010)

[Tabel 6.9 Pengujian Perbandingan Data Anne, Princess Royal 66](#_Toc12319011)

[Tabel 6.10 Pengujian Perbandingan Data Autumn Phillips 67](#_Toc12319012)

[Tabel 6.11 Pengujian Perbandingan Data Charles, Prince of Wales 68](#_Toc12319013)

[Tabel 6.12 Pengujian Perbandingan Data Camilla, Duchess of Cornwall 70](#_Toc12319014)

[Tabel 6.13 Pengujian Perbandingan Data Princess Charlotte of Cambridge 71](#_Toc12319015)

[Tabel 6.14 Pengujian Perbandingan Data Elizabeth II 72](#_Toc12319016)

[Tabel 6.15 Pengujian Perbandingan Data Prince Harry 73](#_Toc12319017)

[Tabel 6.16 Pengujian Perbandingan Data Catherine, Duchess of Cambridge 74](#_Toc12319018)

[Tabel 6.17 Pengujian Perbandingan Data Prince Edward, Earl of Wessex 75](#_Toc12319019)

[Tabel 6.18 Pengujian Perbandingan Data Zara Phillips 77](#_Toc12319020)

[Tabel 6.19 Pengujian Perbandingan Data Prince George of Cambridge 78](#_Toc12319021)

[Tabel 6.20 Pengujian Perbandingan Data Mike Tindall 79](#_Toc12319022)

[Tabel 6.21 Pengujian Perbandingan Data Princess Eugenie 80](#_Toc12319023)

[Tabel 6.22 Pengujian Perbandingan Data Lady Louise 81](#_Toc12319024)

[Tabel 6.23 Pengujian Perbandingan Data Peter Phillips 82](#_Toc12319025)

[Tabel 6.24 Pengujian Perbandingan Data Timothy Laurence 83](#_Toc12319026)

[Tabel 6.25 Pengujian Perbandingan Data Princess Beatrice 84](#_Toc12319027)

[Tabel 6.26 Pengujian Perbandingan Data James, Viscount Severn 85](#_Toc12319028)

[Tabel 6.27 Rangkuman Hasil Pengujian 87](#_Toc12319029)

# DAFTAR KODE SUMBER

[Kode Sumber 3.1 Kode Java untuk memodelkan data Raden Wijaya ke dalam modelActor 24](#_Toc12492939)

[Kode Sumber 3.2 Kode sumber untuk menggabungkan dua model 27](#_Toc12492940)

[Kode Sumber 3.3 Kode sumber untuk melakukan reasoning 27](#_Toc12492941)

[Kode Sumber 4.1 Arsitektur Sistem 30](#_Toc12492942)

[Kode Sumber 4.2 URL data Hayam Wuruk 31](#_Toc12492943)

[Kode Sumber 5.1 Implementasi proses inisialisasi variabel statis 43](#_Toc12492944)

[Kode Sumber 5.2 Implementasi inisialisasi model Instance dan famonto 43](#_Toc12492945)

[Kode Sumber 5.3 Implementasi ekstraksi file RDF tokoh 44](#_Toc12492946)

[Kode Sumber 5.4 Implementasi penggabungan model 44](#_Toc12492947)

[Kode Sumber 5.5 Implementasi proses reasoning 44](#_Toc12492948)

[Kode Sumber 5.6 Implementasi print hasil reasoning sebagai file RDF 45](#_Toc12492949)

[Kode Sumber 5.7 Kode Sumber SPARQL untuk mengambil value bertipe Person dan Fungsi Dropdown Select 47](#_Toc12492950)

[Kode Sumber 5.8 Fungsi Get name 48](#_Toc12492951)

[Kode Sumber 5.9 Fungsi Get father 50](#_Toc12492952)

[Kode Sumber 5.10 Fungsi Get mother 51](#_Toc12492953)

[Kode Sumber 5.11 Fungsi Get sibling 53](#_Toc12492954)

[Kode Sumber 5.12 Fungsi Get spouse 54](#_Toc12492955)

[Kode Sumber 5.13 Fungsi Get child 56](#_Toc12492956)

[Kode Sumber 5.14 Fungsi Get child in law 57](#_Toc12492957)

[Kode Sumber 5.15 Fungsi Get grand child 59](#_Toc12492958)

[Kode Sumber 5.16 get grand child in law 60](#_Toc12492959)

[Kode Sumber 5.17 Get great grand child 62](#_Toc12492960)

***[Halaman ini sengaja dikosongkan]***

# BAB I PENDAHULUAN

Pada bab ini akan dijelaskan hal-hal yang menjadi latar belakang, permasalahan yang dihadapi, batasan masalah, tujuan, metodologi dan sistematika penulisan yang digunakan dalam pembuatan buku tugas akhir ini.

## Latar Belakang

Tokoh bersejarah adalah seseorang yang namanya dikenang karena jasanya. Sedangkan raja adalah gelar yang diberikan kepada anggota kerajaan secara turun-temurun. Tokoh bersejarah dan kerajaan, keduanya menjadi bukti dari adanya suatu kejadian penting di masa lalu. Setiap tokoh memiliki kisah serta rekan hidup yang berbeda. Rekan hidup dapat berarti keluarga, sahabat, teman, dan sebagainya. Berdasarkan pada *history* rekan hidup, tokoh yang satu dengan tokoh yang lain memiliki hubungan terkait sehingga relasi antar tokoh tersebut dapat diketahui. Selain itu, hubungan tersebut juga dapat menentukan kejadian apa yang pernah terlibat di antara mereka.

Keterkaitan antar satu tokoh dengan tokoh yang lain dapat digambarkan dengan ontologi. Ontologi adalah spesifikasi formal dari konsep-konsep yang saling berhubungan. Ontologi mendefinisikan *class, property*, *instance*, dan hubungan sebuah individu dengan individu lain untuk domain tertentu. Dengan ontologi, uraian dari seorang tokoh dapat didefinisikan. Pendefinisian tersebut berguna untuk mencari hubungan antar tokoh. Dalam *cultural heritage*, *actor* adalah salah satu domain yang dapat diontologikan. Ruang lingkup *actor* mencakup *person, group,* dan *organization*. Sedangkan tokoh bersejarah dan pahlawan termasuk dalam agen *person*.

Dalam perkembangan teknologi, pengetahuan tentang tokoh bersejarah dan pahlawan nasional tidak hanya terhimpun di dalam buku-buku sejarah. Banyak situs daring yang menyediakan informasi tentang tokoh bersejarah dan pahlawan nasional, seperti Wikipedia, DBpedia, Everything2, Quora, dan lain-lain. Akan tetapi dalam situs-situs tersebut, mayoritas informasi yang diberikan masih berupa paragraf-paragraf teks atau tabel, sedangkan otak manusia dapat memproses informasi visual 60.000 kali lebih cepat daripada informasi teks [1]. Pengerjaan tugas akhir ini akan mengembangkan ontologi data keluarga kerajaan Indonesia yang sudah ada dan melengkapinya dengan mengkombinasikan *class* dan *property* yang dimilikinya dan ditampilkan dalam sebuah situs web untuk memudahkan pemahaman terkait tokoh sejarah dan kerajaan Indonesia dan relasinya.

## Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana menentukan data property yang nantinya dapat digunakan untuk mendefinisikan relasi dalam domain tokoh sejarah dan kerajaan Indonesia?
2. Bagaimana memodelkan proses reasoning untuk melengkapi relasi tokoh sejarah pada DBpedia?
3. Bagaimana membuat aplikasi untuk menampilkan visualisasi pohon keluarga tokoh sejarah dan kerajaan Indonesia?

## Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Data yang digunakan adalah tokoh sejarah dan kerajaan Indonesia dari DBpedia.
2. Data bersumber dari artikel Wikipedia mengenai tokoh sejarah dan kerajaan Indonesia.
3. Platform pengembangan aplikasi adalah situs web.
4. Data yang digunakan sebagai *value* properti bersumber dari isi propertiDBpedia dan hasil esktraksi manual pada halaman Wikipedia Indonesia *person* terkait.
5. Aplikasi tidak dapat menangani *person* yang tidak memiliki halaman DBpedia.
6. Batas relasi adalah ayah, ibu, saudara, istri, anak, menantu, cucu, pasangan cucu, dan cicit.
7. Proses pelengkapan data dilakukan menggunakan Protege 5.2.0 yang memiliki ekstensi OWL dan Pellet *Reasoner*.
8. *Reasoner* yang digunakan adalah Pellet.
9. Aplikasi yang dibuat tidak menyediakan *form* untuk pengelolaan data (tambah, ubah, hapus).
10. Aplikasi yang dibuat hanya untuk menampilkan deskripsi *person* yang merupakan hasil dari ontologi yang dibangun.

## Tujuan

Tujuan dari pengerjaan tugas akhir ini adalah membuat aplikasi web yang dapat menampilkan pohon keluarga dari tokoh sejarah dan kerajaan Indonesia secara visual untuk membantu dan mempermudah pencarian relasi dari tokoh sejarah dan kerajaan Indonesia.

## Metodologi

Ada beberapa tahapan dalam pengerjaan tugas akhir ini, yaitu sebagai berikut:

1. Studi Literatur

Pada tahap ini, akan dilakukan studi mengenai sejumlah referensi yang diperlukan dalam pembuatan aplikasi yaitu mengenai informasi yang melekat pada tokoh bersejarah, ontologi, DBpedia, *Family Relationships Ontology*, SPARQL, Apache Jena Fuseki, SWRL (*Semantic Web Rule Language*), PHP, dan Pellet *Reasoner*.

1. Implementasi

Pada tahap ini, akan dilakukan implementasi berdasarkan rancangan yang dibuat dalam tahap sebelumnya, yaitu pelengkapan data yang dilakukan dengan *tools* Protege 5.2.0 dengan ekstensi *Web Ontology Language* (OWL). Sedangkan aplikasi sederhana untuk menampikan hasil pencarian relasi dibangun dengan bahasa PHP menggunakan *tools* PhpStorm.

1. Pengujian dan evaluasi

Tahap ini dilakukan dengan uji coba aplikasi untuk mencari dan mengetahui relasi keterkaitan antar tokoh serta mengadakan perbaikan jika ada kekurangan. Pengujian ontologi akan dilakukan dengan menggunakan Pellet *reasoner*. Selain itu, pengujian juga dilakukan dengan membandingkan data hasil uji coba yang ditampilkan pada aplikasi dengan data aslinya yang bersumber dari DBpedia. Evaluasi dilakukan untuk mengetahui karakteristik dan kecenderungan jalannya sebuah program atas sebuah rangkaian *rule* yang diberikan.

1. Penyusunan buku tugas akhir

Tahap ini merupakan tahap penyusunan laporan berupa buku sebagai dokumentasi pengerjaan tugas akhir yang mencakup seluruh dasar teori, desain, implementasi serta hasil pengujian yang telah dilakukan.

## Sistematika Penulisan

Sistematika penulisan dibuat bertujuan untuk mendapatkan gambaran umum dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

1. **Pendahuluan**

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan dan sistematika penyusunan Tugas Akhir.

1. **Dasar Teori**

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan rancang bangun aplikasi berbasis web untuk visualisasi pohon keluarga kerajaan Indonesia ini.

1. **Metode Pemecahan Masalah**

Bab ini membahas mengenai metode yang digunakan untuk memecahkan masalah yang dipaparkan pada rumusan permasalahan.

1. **Analisis dan Perancangan Sistem**

Bab ini membahas mengenai perancangan perangkat lunak. Perancangan perangkat lunak meliputi perancangan data, arsitektur, proses dan perancangan antarmuka pada perangkat lunak.

1. **Implementasi**

Bab ini berisi implementasi dari perancangan perangkat lunak dan implementasi fitur-fitur penunjang.

1. **Pengujian dan Evaluasi**

Bab ini membahas pengujian dengan metode pengujian objektif untuk mengetahui kecocokan data dan kekayaan data.

1. **Kesimpulan**

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

**Daftar Pustaka**

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

**Lampiran**

Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini.

***[Halaman ini sengaja dikosongkan]***

# BAB II DASAR TEORI

Bab ini akan membahas mengenai dasar teori dan literatur yang menjadi dasar pengerjaan tugas akhir ini.

## Kerajaan Indonesia

Sebelum negara Indonesia menganut sistem demokrasi, sistem yang dipakai adalah monarki atau kerajaan. Dalam sejarah bangsa Indonesia, terdapat banyak kerajaan yang berdiri dan jatuh. Di pulau Jawa sendiri, ada total 16 kerajaan dan beberapa masih berkuasa di beberapa daerah keraton, seperti Yogyakarta dan Surakarta [2]. Dari kerajaan yang pernah berdiri, kerajaan Majapahit adalah yang paling luas kekuasaannya, membentang dari Jawa, Sumatra, Kalimantan dan Sulawesi pada masa keemasannya. Disusul dengan kerajaan bahari terbesar Indonesia, kerajaan Sriwijaya yang daerah kekuasaanya mencapai Thailand.

Tokoh sejarah adalah seseorang yang diingat namanya atas jasanya. Setiap tokoh bersejarah memiliki pengalaman hidup yang berbeda-beda. Sering kita temui dalam biografi seorang tokoh bersejarah bahwa mereka masih memiliki relasi dengan tokoh sejarah yang lain. Biografi adalah deskripsi detail dari kehidupan seseorang dari lahir sampai meninggal dunia. Setiap jasa atau karya yang dihasilkan setiap toko sejarah dicatat dalam biografinya.

Setiap tokoh sejarah dan tokoh kerajaan memilki perjalanan hidup dan teman hidup masing-masing. Untuk tugas akhir ini, data yang digunakan adalah data tokoh sejarah dan kerajaan Indonesia yang diambil dari laman ensiklopedia bebas seperti Wikipedia.

## Ontologi

Istilah ontologi berasal dari kajian ilmu filsafat yang kemudian diresap oleh ilmu komputer. Definisi ontologi adalah sebagai studi tentang konsep yang secara sistematik menjelaskan tentang keberadaan segala sesuatu yang konkret. Terdapat tiga komponen utama dari ontologi, yaitu class, property, dan instance [3]. Berikut adalah penjelasan mengenai komponen-komponen tersebut:

1. *Class*

*Class* menspesifikasikan property yang sama dari beberapa instance dan berbentuk hierarki. Selain itu, class juga mencakup superclass dan subclass. Subclass merupakan turunan dari superclassnya yang lebih detail. Setiap subclass mewarisi fungsi dan atribut dari leluhurnya. Subclass mungkin memiliki fungsi dan atribut tambahan sendiri (yang tidak dimiliki oleh leluhurnya). Contohnya adalah class Child memiliki subclass Son dan Daughter, serta memiliki superclass Person. Hubungan antara subclass dan superclass digambarkan dengan class hierarchy yang dicontohkan pada Gambar 2.1.



Gambar 2.1 Class Hierarchy

1. *Property*

*Property* adalah atribut-atribut yang dimiliki oleh suatu Class. Property juga menghubungkan member dari suatu kelas ke member kelas lainnya. Contoh property adalah seperti yang terdapat pada Gambar 2.2.



Gambar 2.2 Property

1. *Instance*

*Instance* merupakan individual dari sebuah class atau biasa disebut dengan member dari class. Contoh hubungan dari *Class, Property* dan *Instance* ditunjukkan oleh Gambar 2.3



Gambar 2.3 Class, Property dan Instance

Selain 3 komponen penting yang telah dijelaskan di atas, terdapat beberapa istilah lain yang perlu dipahami dalam konteks ontologi antara lain d*omain* (*member* dari suatu kelas yang dapat menjadi subjek dari *property* yang diberikan), r*ange* (member dari suatu kelas yang dapat menjadi objek dari *property* yang diberikan), c*onstraint* dan *rule* (menentukan batasan dan istilah-istilah teknis untuk mendukung *reasoning*), dan *relationship* (mekanisme inferensi untuk menggenerasi pengetahuan baru).

Dalam *semantic modelling*, ontologi dapat direpresentasikan dengan berbagai bahasa yang sudah memiliki standarseperti RDF, RDFS, atau OWL. Secara umum, kegunaan ontologi adalah sebagai *controlled vocabulary, semantic interoperability, knowledge sharing,* dan *reuse* [4].

## DBpedia

DBpedia adalah situs web yang bergerak untuk mengekstrak data-data dari halaman Wikipedia dan menampilkannya sebagai informasi yang sudah terstruktur. Data dari sebuah halaman DBpedia dapat kita ambil dengan format yang kita inginkan seperti CSV, RDF, N-Triples, JSON, dan lain-lain. Data di DBpedia masih berupa tabel property dan value. Gambar 2.4adalah contoh sebuah halaman DBpedia.



Gambar 2.4 Contoh Halaman DBpedia

## Semantic Web Rule Language (SWRL)

SWRL merupakan bahasa berbentuk *unary* dan *binary rule statement* yang menjadi bagian dari OWL. Pada dasarnya, *rule* terdiri dari *antecedent* dan *consequent*, keduanya terdiri dari pasangan-pasangan atom. Jika *antecedent* bernilai benar, maka *consequent* juga akan bernilai benar [5]. Pada Tabel 2.1 berikut akan dijabarkan bentuk-bentuk atom yang didefinisikan.

Tabel 2.1 Komponen SWRL

| Atom | Deskripsi |
| --- | --- |
| C(x) | C adalah deklarasi *class* (nama *class*) dan x adalah nama individual atau variabel |
| D(y) | D adalah deklarasi *data range* dan y adalah variabel atau *data value* |
| P(x, y) | P adalah data atau *object property*, x dan y adalah variabel atau OWL individual. y adalah sebuah individual jika P adalah *object property*, sedangkan y adalah sebuah *data value* jika P adalah *data property*. |
| sameAs(x, y) | x dan y adalah variabel atau individual yang menyatakan bahwa keduanya merupakan individu yang sama |
| differentFrom(x, y) | x dan y adalah variabel atau individual yang menyatakan bahwa keduanya merupakan individu yang berbeda |

Berikut merupakan contoh SWRL *rule* yang menyatakan bahwa x3 adalah ayah (*father*) dari x1 jika x2 adalah orang tua (*parent*) dari x1 dan x3 adalah istri (*wife*) dari x2.

|  |
| --- |
| hasParent(?x1, ?x2), hasWife(?x2, ?x3) -> hasFather(?x1, ?x3) |

Tanda “->” digunakan sebagai penghubung antara *antecedent* dan *consequent* atom. Sedangkan “,” berfungsi sebagai penghubung antar atom. Sebuah variabel ditandai dengan ekspresi “?”.

## Family Relationships Ontology

*Family relationship* umumnya digambarkan dengan terstruktur melalui silsilah keluarga. Manusia membutuhkan informasi tentang silsilah keluarganya untuk berbagai hal, diantaranya adalah untuk memperat ikatan batin antar anggota keluarga, mempermudah keturunannya dalam menelusuri asal usul keluarganya, menentukan pewarisan, perkawinan, dan lain sebagainya. Silsilah keluarga adalah bagan yang menampilkan struktur keluarga dalam bentuk pohon. Silsilah keluarga menyimpan informasi yang mendeskripsikan relasi antar anggota keluarga secara kompleks [6].

Keluarga memiliki struktur garis keturunan yang panjang. Jika relasi keturunan dicari secara manual, maka dibutuhkan waktu dan analisis yang lama. Belum tentu setiap anggota keluarga mengenal kerabatnya, karena pada umunya hanya satu atau dua orang yang mengetahui detail keluarga. Semakin bertambahnya pengetahuan membuat hubungan dalam sebuah keluarga dapat diketahui dengan mudah melalui *Family Relationships Ontology*. Ontologi ini memiliki beberapa kelebihan, diantaranya adalah dapat diketahuinya keakraban, relasi, pewarisan, *domain, range, constraint,* dan kesimpulan logis dalam sebuah keluarga secara praktis.

Terdapat banyak ontologi yang telah dibangun menggunakan domain keluarga, salah satunya adalah ontologi yang digunakan pada pengerjaan tugas akhir ini, yaitu FamilyTree. Ontologi tersebut didapatkan dari portal The University of Manchester. Ontologi FamilyTree memiliki URI [*http://www.co-ode.org/roberts/family-tree.owl*](http://www.co-ode.org/roberts/family-tree.owl) [7]. Ontologi tersebut adalah sebuah ontologi sederhana dengan domain hubungan keluarga yang mendeskripsikan keluarga Robert Stevens. FamilyTree merupakan ontologi yang kompleks dan lengkap. Pembangunan ontologi tersebut dimaksudkan untuk menghasilkan suatu ontologi yang meminimalkan *relationships* dan memaksimalkan *inference*. Oleh karena itu, ontologi ini banyak menggunakan *role chain, nominal*, dan *properties hierarchy*.

Cuplikan kelas, properti, dan individu yang terdapat dalam ontologi tersebut dapat dilihat padaGambar 2.5.



Gambar 2.5 Ontologi FamilyTree Keluarga Robert Stevens

Bisa disimpulkan bahwa ontologi milik Robert Stevens adalah salah satu yang paling lengkap. Akan tetapi dalam pengerjaan tugas akhir ini, tidak semua *property* dan *class* dari ontologi tersebut hanya akan dipakai relasi yang umum, seperti hasChild, hasParent, hasGrandchild, hasSpouse, selain itu akan dihapus. Dan karena *instance* atau *individual* di tugas akhir ini adalah keluarga kerajaan Indonesia, maka *instance* di ontologi ini dihapus.

## SPARQL

SPARQL (dibaca “sparkle) adalah protokol RDF Query Language yang berfungsi untuk mengambil dan memanipulasi data dari sebuah basis data triple-store. RDF(*Resource Description Framework*) adalah tipe file yang terdiri dari statemen-statemen yang memiliki tiga variabel sebagai subjek, predikat, dan objek. Protokol SPARQL umumnya digunakan oleh peneliti Semantic Web. Contoh *syntax* SPARQL seperti yang ditunjukkan oleh Gambar 2.6:



Gambar 2.6 Contoh SPARQL Query

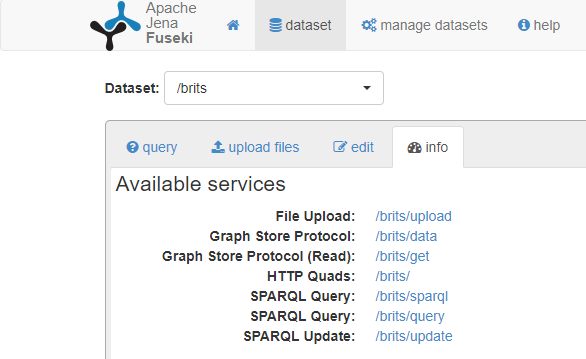
## Apache Jena Fuseki

Apache Jena Fuseki adalah server SPARQL yang juga bisa bertindak sebagai service sistem operasi dan aplikasi web berbasis java. Dalam konteks ini, Apache Jena-Fuseki bertindak sebagai basis data triple-store yang bisa diakses melalui request HTTP. Gambar 2.7. menunjukkan daftar basis data yang ada di dalam server Apache Jena Fuseki.



Gambar 2.7 Database Triple Store Apache Jena Fuseki

Apache Jena Fuseki menyediakan beberapa API untuk digunakan oleh peneliti seperti pada Gambar 2.8.



Gambar 2.8 Daftar API Apache Jena Fuseki

Beberapa fungsi API tersebut antara lain untuk *endpoint* pengunggahan file RDF (/upload), membaca data (/get), *query* SPARQL (/query), dan memperbarui data (/update). Untuk implementasi pengaksesan basis data, API yang digunakan adalah SPARQL Query. Return value dari SPARQL *query* menggunakan Apache Jena Fuseki bisa berupa JSON(JavaScript Object Notation), XML(eXtensible Markup Language), atau CSV(Comma Separated Value).

## Pellet Reasoner



Gambar 2.9 Arsitektur Pellet Reasoner

Implementasi OWL *reasoner* yang sudah ada didasarkan pada beberapa pendekatan. *Reasoner* deskripsi logika (seperti Pellet dan RacerPro) menggunakan implementasi algoritma tableaux. Penggunaan algoritma tersebut memanfaatkan penelitian yang telah dilakukan untuk kasus algoritma deskripsi logika pengetahuan berdasar pada formalitas OWL [8]. Pellet didasarkan pada algoritma tableaux yang dikembangkan untuk mengekspresikan *Description Logics*. Pellet mendukung semua konstruksi OWL DL termasuk owl:oneOf dan owl: hasValue. Saat ini, belum ada algoritma lengkap yang *decidable* dan efektif untuk semua OWL DL (khususnya, penanganan *inverse properties* dan *cardinality restrictions*). Pellet mengkombinasikan algoritma yang lengkap sebagai reasoner, yaitu OWL DL tanpa *nominals* (SHIN (D)) dan OWL DL tanpa *inverse properties* (SHON (D)). Algoritma ini dikombinasikan untuk mendapatkan penalaran yang lengkap dan berkaitan dengan semua DL. Pellet telah terbukti praktis berguna dalam berbagai pekerjaan saat ini. Gambar 2.1 menunjukkan komponen utama Pellet *reasoner*.

Ontologi OWL di*parsing* ke dalam RDF dengan pola *triple* (Sintaksis RDF / XML, N3 dan N-Triple yang mendukung). Pellet memvalidasi jenis dari ontologi dimana *triple* *RDF* dikonversi menjadi pernyataan dan *axiom* berbasis pengetahuan. Jika level ontologi adalah OWL Full karena hilangnya tipepola *triple*, maka Pellet menggunakan beberapa heuristik untuk memperbaiki ontologi. Misalnya *untyped resource* yang telah digunakan dalam predikat *position* dalam sebuah pola *triple* akan disimpulkan menjadi *datatype property* jika *triple* literal dalam posisi objek.

Pellet menyimpan *axiom* tentang kelas-kelas dalam komponen TBox dan menyimpan pernyataan tentang individu dalam komponen abox. Partisi TBox, adalah tempat penyerapan dan optimasi berlangsung. Tableau reasoner menggunakan *rule* tableau standar dan mencakup berbagai optimasi standar seperti keterkaitan yang diarahkan pada *backjumping*, percabangan semantik dan strategi pemblokiran awal. *Datatype reasoning* untuk *built-in* dan pengambilan XML *Schema datatypes* primitif didukung dalam *reasoner* ini. Pellet diimplementasikan dalam Java dan berada di bawah lisensi MIT [9].

## SPARQL Lib

SPARQL Lib adalah sebuah library PHP yang dikembangkan oleh departemen Computer Science dari University of Southampton, United Kingdom yang berfungsi untuk mengolah data bertipe RDF dalam aplikasi berbasis PHP. Dalam konteks ini, SPARQL Lib digunakan untuk mengambil data RDF dari basis data triple store melalui panggilan API. Contoh dasar penggunaan SPARQL Lib dalam proyek berbasis PHP adalah seperti Gambar 2.10. Hasil dari SPARQL Query dengan SPARQL Lib adalah array *row* dan *field*. Gambar 2.10 mencontohkan cara mengambil data ruangan dan labelnya dari situs basis data yang menyediakan SPARQL API yaitu http://sparql.data.southampton.ac.uk/.



Gambar 2.10 Contoh Penggunaan SPARQL Lib

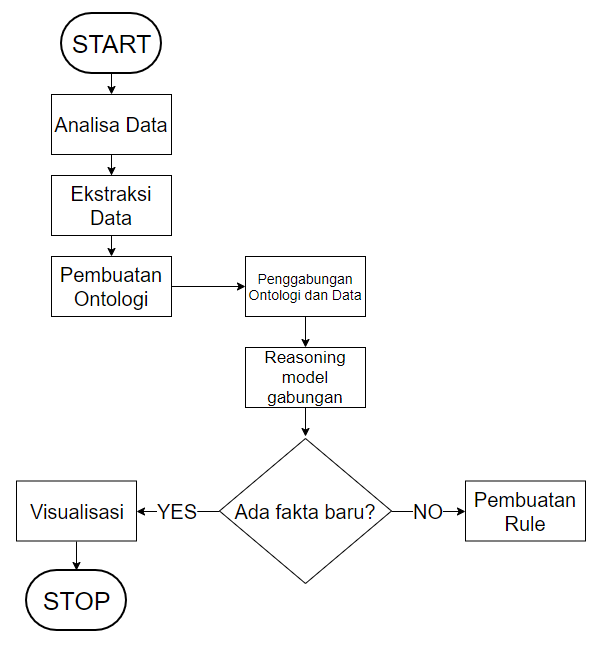
## Apache Jena

Apache Jena adalah *plugin open source* berbasis Java yang digunakan untuk membangun aplikasi *Linked Data* dan *Semantic Web*. *Framework* ini terdiri dari beberapa API yang berinteraksi secara bersamaan untuk memproses data dengan format RDF. Aplikasi yang memiliki *plugin* Apache Jena sanggup membuat model, memodelkan data dari API triple store, menggabungkan model, hingga *reasoning*. Kode sumber Apache Jena bisa diunduh di https://jena.apache.org/download/index.cgi. Dalam konteks ini, Apache Jena bertidak sebagai *plugin* dalam program Java sehingga memungkinkan aplikasi Java bisa memodelkan dan mengolah data RDF.

***(Halaman ini sengaja dikosongkan)***

# BAB III METODOLOGI PEMECAHAN MASALAH

Pada bab ini dijelaskan mengenai langkah-langkah yang dilakukan untuk mencari relasi dari suatu *person*. Mulai dari metode yang dilakukan untuk mengambil data *person* sampai menampilkan grafik pohon keluarga.



Gambar 3.1 Flowchart pengembangan sistem

Alur pemecahan masalah dapat dilihat pada Gambar 3.1. Pemecahaan masalah dimulai dengan menganalisis data DBpedia. Setelah analisis dilakukan, maka diputuskan untuk menggunakan data keluarga kerajaan Indonesia dikarenakan kurangnya relasi tokoh yang disediakan oleh DBpedia Indonesia. Kemudian, data diekstrak dengan aplikasi berbasis Jena agar dapat digabungkan dengan ontologi secara mudah. Proses ekstraksi data dan penggabungan data dilakukan menggunakan Apache Jena. Proses selanjutnya adalah melakukan reasoning dengan Pellet Reasoner di dalam aplikasi Jena. Setelah reasoning selesai, maka fakta-fakta baru akan dihasilkan, serta ontologi yang baru akan diupload ke basis data triple-store. Data di triple-store lalu ditampilkan secara grafis sebagai pohon keluarga. Deskripsi lebih detail tentang setiap proses akan dijelaskan lebih detail pada subbab bab ini.

## Analisis Data

Untuk memecahkan masalah pencarian relasi keluarga tokoh bersejarah, langkah yang pertama kali dilakukan adalah menganalisis dataset yang akan digunakan. Ontologi memiliki beberapa domain, yaitu *actor, place, time, dan event*. Domain yang menjadi topik pada pengerjaan tugas akhir ini adalah *actor*. Ruang lingkup *actor* meliputi *person, group, dan organization*. *Person* tidak dapat berdiri sendiri tanpa adanya keterkaitan dengan *place, time,* dan *event*.

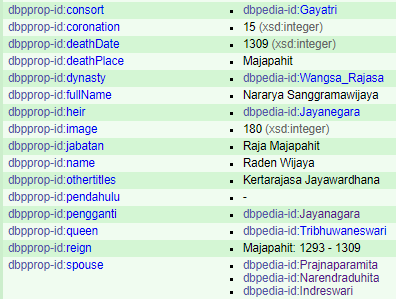
Data yang digunakan dalam perancangan ontologi ini adalah data biografi keluarga kerajaan Indonesia. Daftar kerajaan Indonesia dapat diperoleh dari ensiklopedia gratis Wikipedia Indonesia yang dapat diakses di laman *https://id.wikipedia.org/wiki/Kategori:Kerajaan\_di\_Indonesia*.

*Person* yang digunakan sebagai data adalah *person* yang dinilai memiliki banyak keterkaitan dengan *person* lain. *Person* harus memiliki atribut foaf:name. Maka dari itu, untuk pembanding, akan diolah pula data dari Presiden Indonesia, data pahlawan nasional, dan data DBpedia Inggris tentang kerajaan Inggris.

Domain inti dari sebuah ontologi menangkap konsep utama (*classes*) dan hubungan (*properties*) yang mencakup ruang lingkup domain tersebut. Bahkan ontologi dengan domain yang sama bisa heterogen karena berbagai kepentingan, perspektif pengembang, tujuan yang berbeda, dan konteks aplikasi. Untuk membuat ontologi yang lengkap dan mencakup semua inti domain akan membutuhkan *cost* yang tinggi karena ekonomi, waktu, sumber daya lainnya, serta kondisi dunia yang selalu berubah [10].

### Analisis Data dari DBpedia

Terdapat berbagai macam *open data* yang dapat diakses melalui *internet* tanpa berbayar seperti DBpedia. Data keluarga yang akan dipakai sebagai model adalah data dengan kelengkapan properti-properti utama yang diperlukan untuk mengetahui silsilah keluarga seorang *person* seperti *child, spouse,* dan *parent*. Contoh properti yang akan dipakai adalah data properti dari Raden Wijaya yang bisa dilihat di Gambar 3.2.



Gambar 3.2 Halaman DBpedia tentang properti keluarga Raden Wijaya

Berdasarkan semua *property* yang terdapat pada halaman DBpedia, dipilih *property* dalam batasan masalah seperti *name, parent, spouse,* dan *issue*(istilah resmi untuk keturunan biologis). Untuk melengkapi data, akan dibuat data properti menantu, cucu, pasangan cucu, dan cicit seperti pada subbab 1.3 dan Tabel 3.1.

Tabel 3.1 Daftar properti yang dibutuhkan dan yang akan dihasilkan

|  |  |
| --- | --- |
| Properti yang dibutuhkan | Properti yang akan dihasilkan |
| Name Spouse  Parent  Issue | Child in Law  Grandchild  Grandchild in Law  Great grand child |

## Ekstraksi Data Sebagai Model

Untuk melakukan proses ekstraksi data, diperlukan aplikasi berbasis Java yang memiliki *plugin* Apache Jena. Kode Sumber 3.1 berikut digunakan untuk ekstraksi dan pemodelan tokoh dari DBpedia.

|  |
| --- |
| Model modelActor = fManager.loadModel("http://dbpedia.org/data/Raden\_Wijaya "+ ".ttl"); |

Kode Sumber 3.1 Kode Java untuk memodelkan data Raden Wijaya ke dalam modelActor

Kode pada Kode Sumber 3.1 akan membaca semua data properti Raden Wijaya serta relasinya. Tentuya tidak hanya Raden Wijaya saja yang dijadikan model, tetapi raja-raja, tokoh sejarah dan pahlawan nasional lainnya juga dimodelkan.

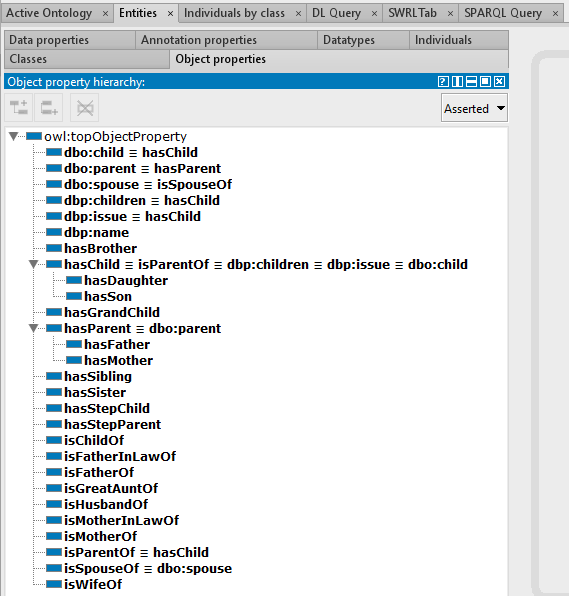
## Pembuatan Ontologi

Pada tahap ini, ontologi dibangun dengan menggabungkan ontologi yang sudah ada. Ontologi yang digunakan adalah *Family Relationships Ontology* milik Robert Stevens. Akan tetapi tidak semua *class, individual, data properties,* ataupun *object properties* akan digunakan, hanya yang benar-benar dibutuhkan saja. Beberapa *property* yang digunakan adalah hasChild, hasParent, isSpouseOf, hasChildInLaw, hasGrandChild, hasGrandChildInLaw, has GreatGrandChild. Beberapa properti yang memiliki arti yang sama akan diatur sebagai *equivalent class,* seperti properti hasChild, isParentOf, dbp:children, dbp:issue, dan dbo:child. Daftar *class* dan *property* yang akan digunakan ditunjukkan di Gambar 3.3, Gambar 3.4 dan Gambar 3.5. Dan pemetaan properti DBpedia dan Family Tree App dapat dilihat di Tabel 3.2.



Gambar 3.3 Hirarki Class

Gambar 3.4 Hirarki Data property



Gambar 3.5 Hirarki Object property

Tabel 3.2 Pemetaan properti

|  |  |
| --- | --- |
| Properti DBpedia | Properti Family Tree App |
| foaf:name | foaf:name |
| dbo:child | hasChild, isParentOf, dbp:children, dbp:issue, dbo:child |
| dbo:spouse | isSpouseOf, dbo:spouse |
| dbo:parent | hasParent, dbo:parent |

## Penggabungan model data dan model ontologi

Model yang digabungkan adalah model data DBpedia yang diperoleh dengan ekstraksi dan model ontologi *Family Relationship Ontology* yang telah dibuat pada langkah pembuatan ontologi di subbab 3.3. Penggabungan ini dilakukan dengan menggunakan fungsi *createUnion* dari class ModelFactory seperti pada Kode Sumber 3.2.

|  |
| --- |
| **final** Model union = ModelFactory.*createUnion*(modelA,modelB); |

Kode Sumber 3.2 Kode sumber untuk menggabungkan dua model

## Reasoning pada Model Gabungan

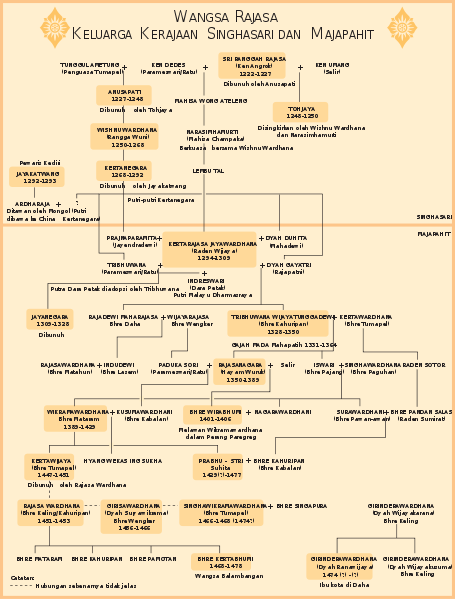
Untuk proses *reasoning*, yang digunakan adalah Pellet Reasoner. Proses ini terdiri dari tiga fase, yaitu *model reading*, *classifying*, dan *realizing*. Setelah diperoleh hasilnya, maka hasil tersebut akan diprint menjadi file RDF yang selanjutnya diunggah ke basis data Apache Jena Fuseki.

|  |
| --- |
| Reasoner reasoner = PelletReasonerFactory.*theInstance*().create(); InfModel reasonedModel = ModelFactory.*createInfModel*(reasoner,modelA); |

Kode Sumber 3.3 Kode sumber untuk melakukan reasoning

## Penampilan Data

Untuk penampilan data, ada tiga bagian, yaitu proses *query* SPARQL menggunakan SPARQL Lib, pemilihan data individu dan visualisasi sebagai pohon keluarga seperti yang dicontohkan pada Gambar 3.6 yang akan diimplementasi dengan bahasa pemrograman PHP.



Gambar 3.6 Silsilah keluarga kerajaan Singasari dan Majapahit [11]

# BAB IV ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini dijelaskan tentang analisis permasalahan dan perancangan Tugas Akhir. Analisis permasalahan membahas tentang permasalahan yang diangkat dalam Tugas Akhir ini beserta solusi yang ditawarkan. Selanjutnya dibahas juga tentang perancangan sistem yang dibuat.

## Analisis

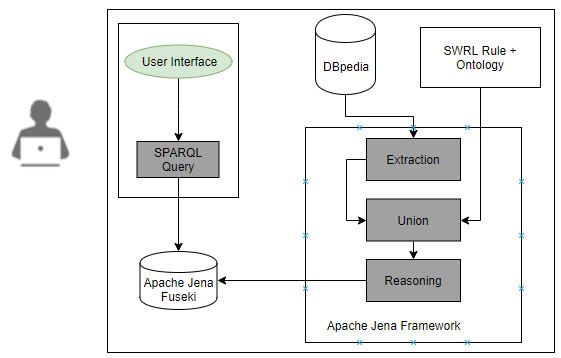
Tahap analisis dibagi menjadi beberapa bagian antara lain cakupan permasalahan, deskripsi umum sistem, kasus penggunaan sistem dan kebutuhan perangkat lunak.

### Cakupan Permasalahan

Permasalahan yang diangkat dalam tugas akhir ini adalah visualisasi pohon keluarga tokoh sejarah dan kerajaan Indonesia. Studi kasus permasalahan tersebut dipecahkan dengan ekstraksi data, penggabungan data, *reasoning* dan visualisasi. Pencarian relasi antar *person* dilakukan dengan menggunakan *property* dan SWRL *rule*. Untuk mendapatkan fakta-fakta baru, dilakukan proses *reasoning* menggunakan Pellet *reasoner*. Setelah proses *reasoning* selesai, akan didapatlam fakta-fakta baru yang kemudian disimpan sebagai ontologi baru dalam bentuk RDF. Ontologi baru tersebut lalu disimpan di dalam basis data triple store. Tentu saja hal tersebut akan menyulitkan pengguna yang ingin mengetahui fakta-fakta baru yang muncul setelah ontologi diberikan *rule*. Oleh karena itu, agar dapat dimanfaatkan secara aplikatif maka dibutuhkan sebuah sistem sederhana yang dapat menampilkan hasil *reasoning* dari ontologi yang dibangun. Untuk memudahkan pengguna, sistem sederhana tersebut akan dirancang dengan tampilan yang mudah dipahami.

### Deskripsi Umum Sistem

Perangkat lunak yang dibangun dalam pengerjaan tugas akhir ini diberi nama Family Tree App. Family Tree App dibangun dengan tujuan untuk membantu ontologi dalam menampilkan hasil-hasil yang didapatkannya.



Kode Sumber 4.1 Arsitektur Sistem

Arsitektur sistem bisa dilihat di Kode Sumber 4.1. Untuk menampilkan fakta-fakta yang didapatkan dari ontologi tersebut, perangkat lunak harus bisa membaca berkas ontologi yang telah dibangun. Family Tree App dirancang sebagai perangkat lunak berbasis *web* yang menggunakan bahasa pemrograman PHP dan *library* SPARQL Lib. Perangkat lunak ini bisa mengakses data dari basis data triple store. Sedangkan keluaran dari perangkat lunak Family Tree App adalah halaman HTML dengan tampilan pohon keluarga dari seorang tokoh yang bersumber dari basis data triple store tersebut. Berikut detail tiap komponennya.

1. User Interface

User Interface ini memungkinkan pengguna untuk beriteraksi dengan sistem dengan cara memilih URL *person* dalam dropdown. Sistem lalu menampilkan skema pohon keluarga tokoh *person* yang dipilih.

1. SPARQL Query

Setelah pengguna memilih *person*, sistem akan melakukan SPARQL *query* menggunakan *plugin* SPARQL Lib untuk mengambil data keluarga *person* yang dipilih dari basis data Apache Jena Fuseki.

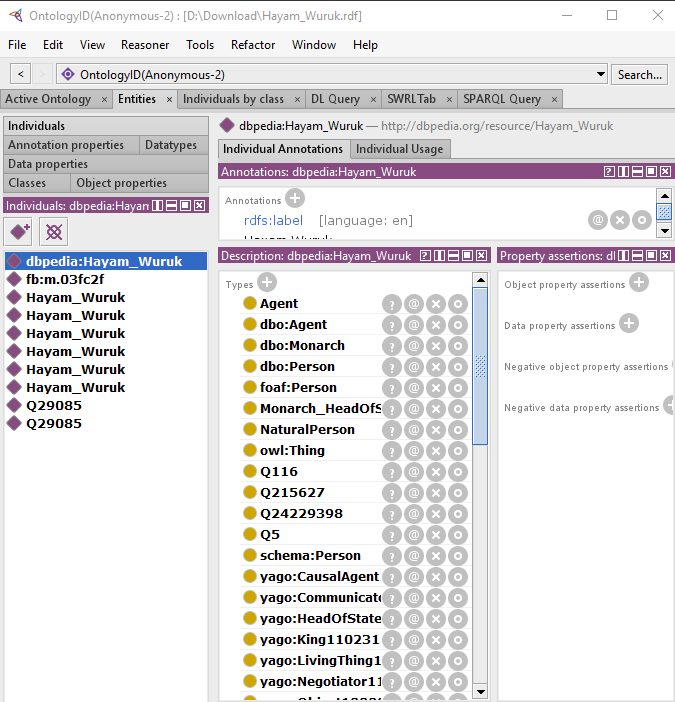
1. Apache Jena Fuseki

Apache Jena Fuseki berfungsi untuk menyimpan data berbentuk *triple-store*. Basis data ini menyediakan API untuk membaca data dan perintah *query*.

1. DBpedia  
   Data dari DBpedia akan diunduh oleh aplikasi Jena, dan dimodelkan untuk digabungkan dengan ontologi *Family Relationship*.
2. Ontology  
   *Family Relationship Ontology* yang telah direvisi seperti pada subbab 3.3 dibaca oleh aplikasi Jena dan dimodelkan, lalu digabungkan dengan model data DBpedia.
3. Extraction  
   Merupakan proses ekstraksi data *person* dari DBpedia dengan cara mengunduh file RDF. Input untuk proses ini adalah URL DBpedia, sedangkan outputnya adalah file RDF. Dapat dilihat contoh input pada dan output pada .

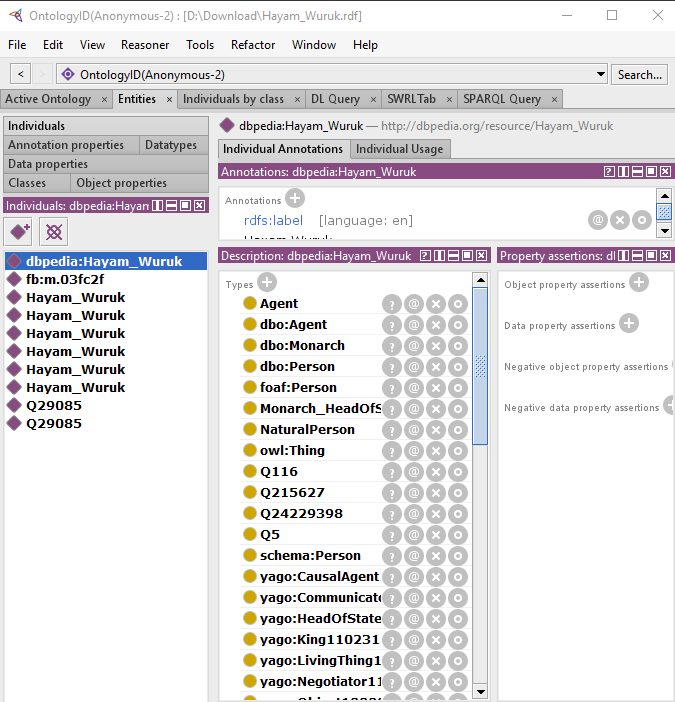
|  |
| --- |
| **http://dbpedia.org/data/Hayam\_Wuruk.rdf** |

Kode Sumber 4.2 URL data Hayam Wuruk

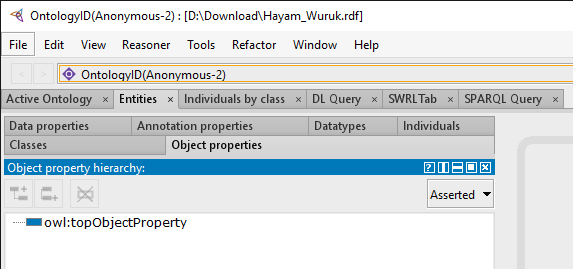


Gambar 4.1 Output file Hayam\_Wuruk.rdf bisa dilihat dengan tools Protégé.

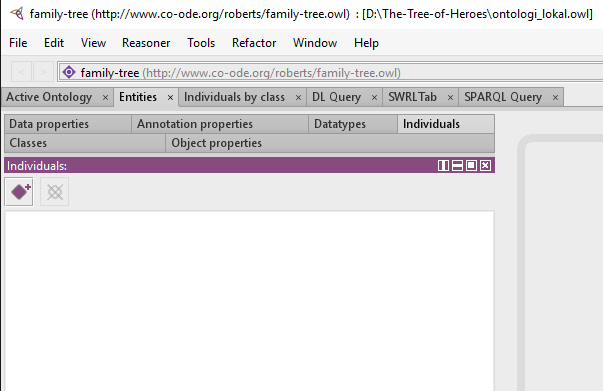
1. Union  
   Merupakan proses penggabungan model *Family Relationship* dan model data DBpedia. Input dari proses ini adalah dua model RDF dan outputnya adalah kombinasi dari keduanya. Contoh dari penggabungan model Hayam Wuruk dan model *Family Relationship Ontology*. Sebagai input, adalah data ontologi Hayam Wuruk (Gambar 4.2 dan Gambar 4.3) dan ontologi Family Tree (Gambar 4.4 dan Gambar 4.5). Sebagai output, adalah ontologi Family Tree yang baru, yang menyimpan data dan properti dari kedua ontologi pada Gambar 4.6 dan Gambar 4.7.



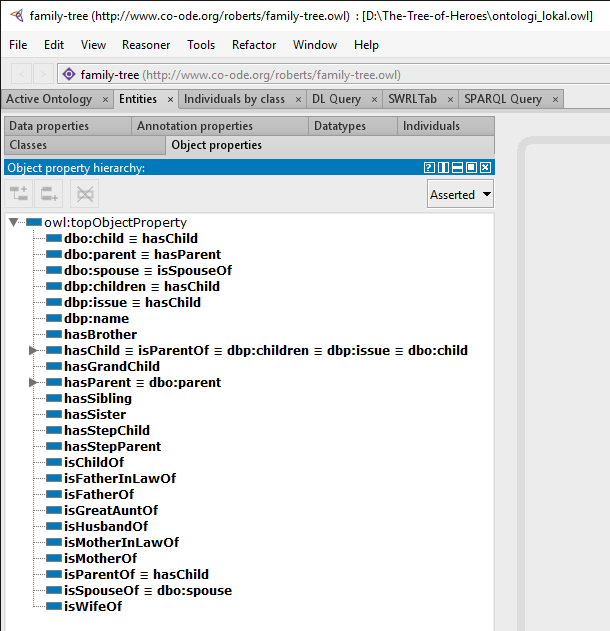
Gambar 4.2 Ontologi Hayam Wuruk (Individuals)



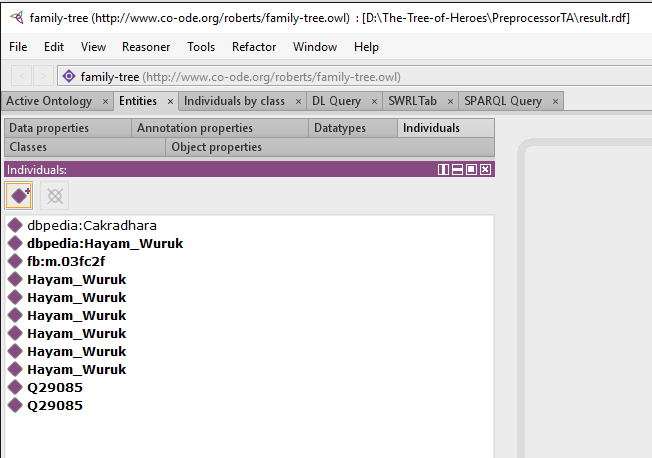
Gambar 4.3 Ontologi Hayam Wuruk (Object Properties)



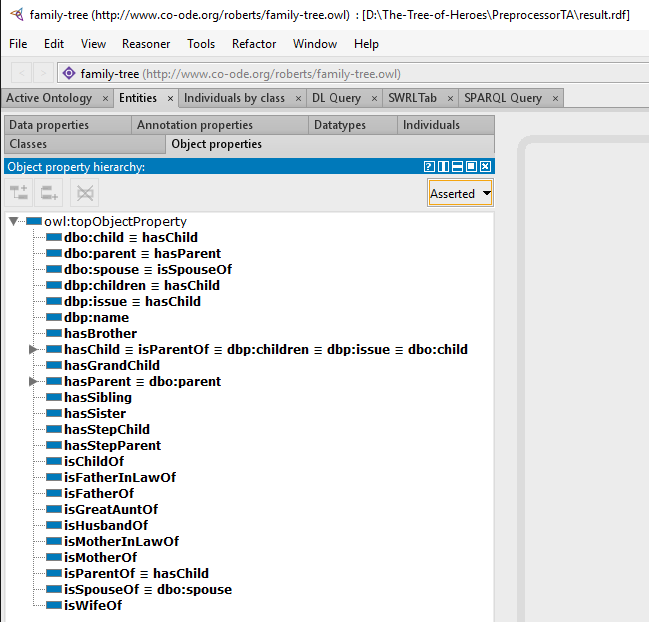
Gambar 4.4 Ontologi Family Tree (Individuals)



Gambar 4.5 Ontologi Family Tree (Object Properties)

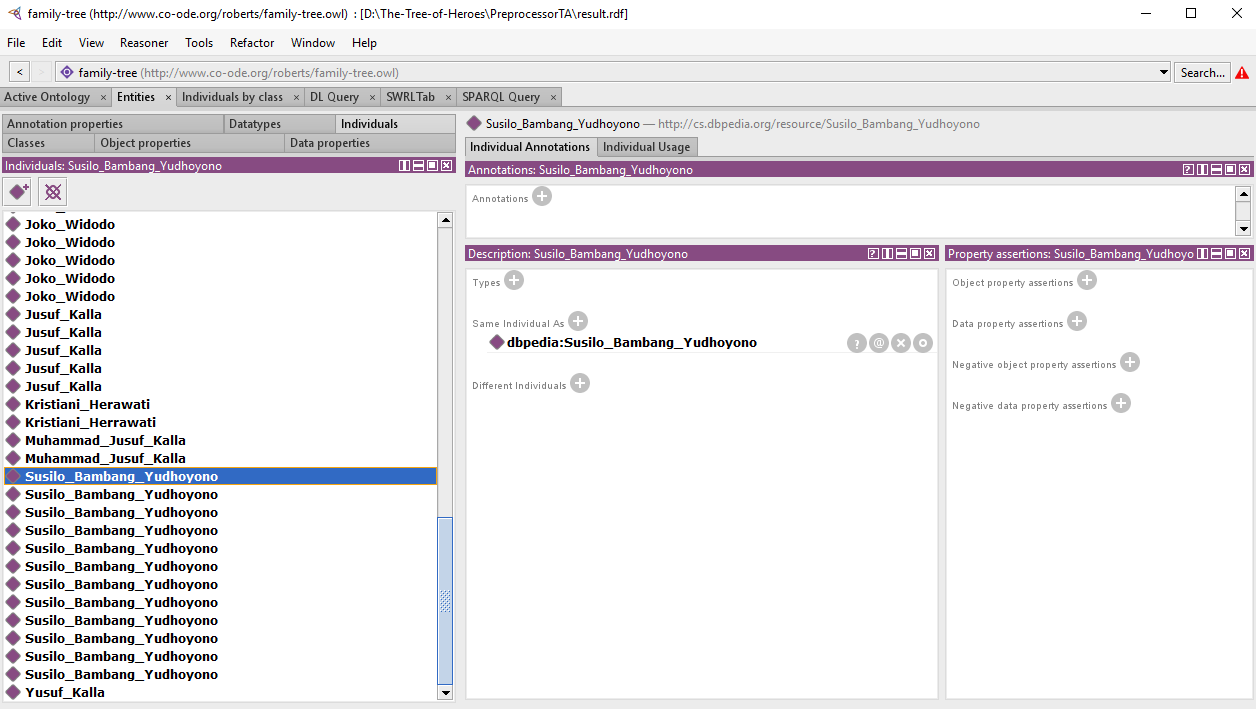


Gambar 4.6 Ontologi gabungan (Individuals)

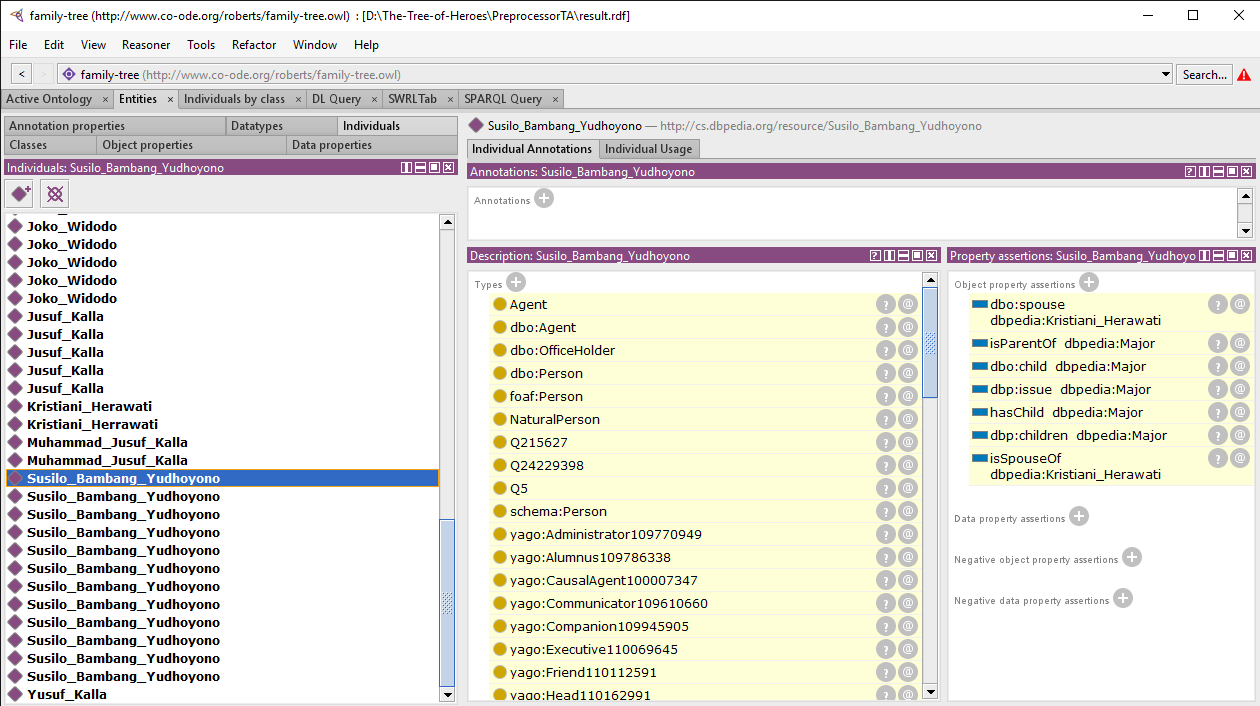


Gambar 4.7 Ontologi gabungan (Object Properties)

1. Reasoning  
   Merupakan proses untuk mencari fakta baru dari suatu model di aplikasi Jena. Input dari proses ini adalah model dan bisa dilihat di Gambar 4.8, serta outputnya juga model pada Gambar 4.9.



Gambar 4.8 Individu Susilo Bambang Yudhoyono sebelum reasoning



Gambar 4.9 Individu Susilo Bambang Yudhoyono setelah reasoning

### Spesifikasi Kebutuhan Perangkat Lunak

Bab ini menjelaskan kebutuhan perangkat lunak dalam bentuk diagram kasus dan diagram aktivitas. Masing-masing diagram menjelaskan perilaku atau sifat dari sistem ini.

#### Kebutuhan Fungsional

Kebutuhan fungsional adalah kebutuhan pokok yang harus dipenuhi agar sistem dapat berjalan dengan baik. Daftar kebutuhan fungsional dapat dilihat pada Tabel 4.1**.**

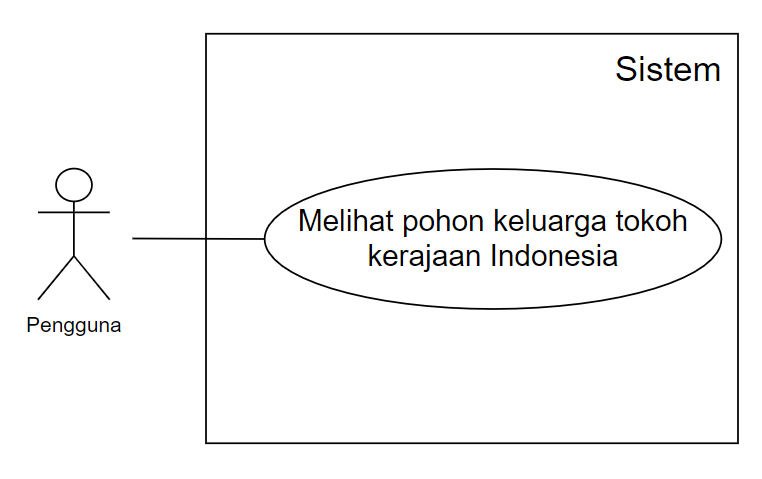
Tabel 4.1 Daftar Kebutuhan Fungsional Perangkat Lunak

| **Kode Kebutuhan** | **Kebutuhan Fungsional** | **Deskripsi** |
| --- | --- | --- |
| TA-F0001 | Menampilkan pohon keluarga tokoh | Pengguna dapat melihat pohon keluarga tokoh sejarah dan kerajaan Indonesia |

### Aktor

Aktor merupakan entitas-entitas yang terlibat dan berinteraksi langsung dengan sistem. Entitas yang dimaksud dapat berupa manusia, sistem, atau perangkat lunak yang lain. Aktor yang berinteraksi dengan Tugas Akhir ini yaitu pengguna yang diasumsikan tidak memahami bahasa pemrograman. Pengguna dapat memilih entitas melalui *dropdown select* atau memilih tautan yang disediakan oleh sistem untuk melihat informasi dari seorang tokoh sejarah Indonesia.

### Kasus Penggunaan



Gambar 4.10 Diagram Kasus Penggunaan Sistem

Kasus penggunaaan dalam Subbab ini akan dijelaskan secara rinci. Kasus penggunaan dijabarkan dalam bentuk spesifikasi kasus penggunaan dan diagram aktivitas. Diagram kasus penggunaan dapat dilihat pada Gambar 4.1. Daftar kode diagram kasus penggunaan sistem dapat dilihat pada Tabel 4.2.

Tabel 4.2 Daftar Kode Diagram Kasus Penggunaan

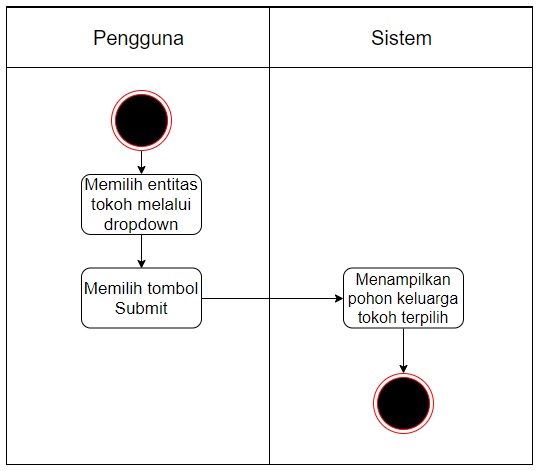
|  |  |
| --- | --- |
| **Kode Kasus Penggunaan** | **Nama** |
| TA-UC0001 | Melihat pohon keluarga tokoh sejarah dan kerajaan Indonesia |

#### Melihat Pohon Keluarga Tokoh sejarah dan kerajaan Indonesia

Pada kasus penggunaan ini, sistem membaca data yang ada di basis data Apache Jena Fuseki. Informasi yang terdapat dalam basis data tersebut selanjutnya dikonversi menjadi sebuah halaman HTML. Spesifikasi kasus penggunaannya dapat dilihat pada Tabel 4.3**Error! Reference source not found.**. Diagram aktivitasnya dapat dilihat pada **Error! Reference source not found.**.

Tabel 4.3 Spesifikasi Kasus Penggunaan Melihat Informasi Tokoh

|  |  |
| --- | --- |
| **Nama** | Melihat pohon keluarga tokoh |
| **Kode** | TA-UC0001 |
| **Deskripsi** | Pengguna bisa melihat pohon keluarga tokoh yang dipilih |
| **Tipe** | Fungsional |
| **Pemicu** | Pengguna menekan tombol *submit* |
| **Aktor** | Pengguna |
| **Kondisi Awal** | Pohon keluarga belum ditampilkan |
| **Alur:**   * **Kejadian Normal** |  |
| 1. Pengguna memilih tokoh melalui *dropdown select*. 2. Pengguna menekan tombol *submit.* 3. Sistem menampilkan halaman pohon keluarga dari tokoh yang dipilih. |
| **Kondisi Akhir** | Sistem menampilkan informasi dari tokoh yang dipilih dalam bentuk pohon keluarga |
| **Alur alternatif** | Pengguna memilih tautan.  Sistem menampilkan pohon keluarga tokoh tautan. |
| **Kebutuhan Khusus** | Tidak ada |

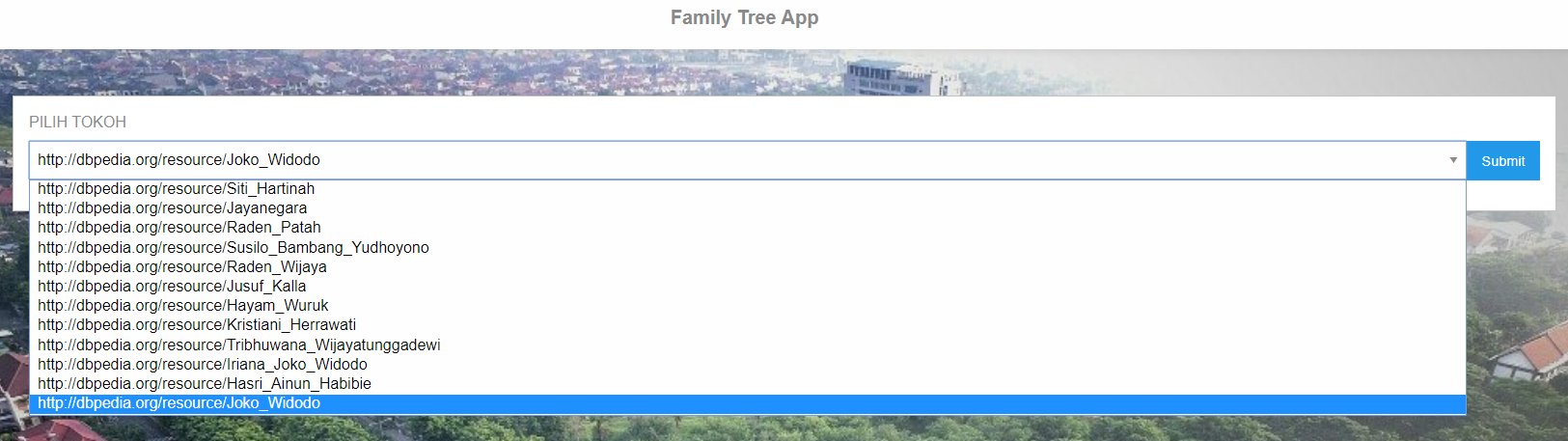


Gambar 4.11 Diagram Aktivitas Melihat Pohon Keluarga Tokoh

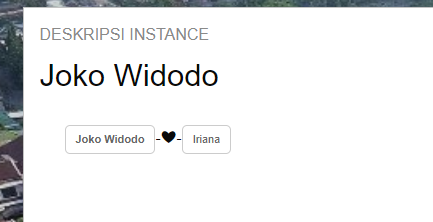
## Perancangan Antarmuka Pengguna

Bagian ini membahas mengenai perancangan antarmuka yang akan dibuat. Rancangan antarmuka dibuat agar semudah mungkin dapat dipahami dan digunakan oleh pengguna.

Antarmuka Family Tree App terdiri dari satu halaman. Di halaman tersebut, terdapat satu panel *dropdown select* dan satu panel sebagai tempat deskripsi entitas tokoh atau tautan yang dipilih. Deskripsi entitas tokoh terdiri dari satu tabel dengan sejumlah baris informasi terkait entitas tokoh yang dipilih. Rancangan antarmuka halaman utama ini dapat dilihat pada Gambar 4.3. Sedangkan rancangan antarmuka halaman informasi data tokoh dapat dilihat pada Gambar 4.4. Penjelasan mengenai atribut-atribut yang terdapat pada halaman ini bisa dilihat pada Tabel 4.4.



Gambar 4.12 Antarmuka Halaman Utama Family Tree App



Gambar 4.13 Antarmuka Halaman Pohon Keluarga Family Tree App

Tabel 4.4 Spesifikasi Atribut Rancangan Antarmuka Halaman Family Tree App

| **No.** | **Nama Atribut Antarmuka** | **Jenis Atribut** | **Kegunaan** |
| --- | --- | --- | --- |
| 1 | *Entity Dropdown Select* | *Form* | Menampilkan daftar entitas tokoh |
| 2 | *Submit Button* | *Button* | Mengeksekusi *request form* |
| 3 | *Entity Family Tree* | *Tree* | Menampilkan pohon keluarga dari entitas tokoh yang dipilih |

***[Halaman ini sengaja dikosongkan]***

# BAB V IMPLEMENTASI

Bab ini membahas tentang implementasi dari perancangan sistem yang telah dibuat. Proses implementasi dari setiap fungsi pada perangkat lunak Family Tree App akan diuraikan selangkapnya pada bab ini. Implementasi perangkat lunak Family Tree App menggunakan bahasa pemrograman PHP dengan *library* SPARQL Lib.

Agar dapat menampilkan fakta yang belum ada, pada ontologi ini diterapkan sejumlah *rule* yang telah dijelaskan pada Sub subbab 3.3. Setelah itu, dilakukan proses *reasoning* ontologi menggunakan Pellet *Reasoner*. Data model yang didapatkan dari proses *reasoning* kemudian dikonversi menjadi data RDF agar dapat dibaca oleh Apache Jena Fuseki. Apache Jena Fuseki berperan sebagai basis data untuk menyimpan data RDF dalam bentuk triple store. Lalu SPARQL Lib sebagai *query converter* yang dapat mengambil data dari Apache Jena Fuseki untuk ditampilkan di *user interface*.

## Implementasi Proses Ekstraksi, Penggabungan, dan Reasoning

Pada bagian ini dijelaskan secara terperinci mengenai implementasi proses ekstraksi, penggabungan dan *reasoning* yang digunakan untuk menghasilkan data yang akan dipakai. Implementasi dilakukan di dalam kerangka kerja Apache Jena yang ditunjukkan pada Kode Sumber 5.1 sampai dengan Kode Sumber 5.6.

|  |
| --- |
| String dbJenaFuseki=**"brits"**; String READ\_FUSEKI = **"http://localhost:3030/"**+dbJenaFuseki; String OWL\_FILE\_LOCATION = **"D:/The-Tree-of-Heroes/ontologi\_lokal.owl"**; File fileRDF = **new** File(**"D:\\The-Tree-of-Heroes\\PreprocessorTA\\result.rdf"**); |

Kode Sumber 5.1 Implementasi proses inisialisasi variabel statis

Fungsi kode sumber diatas adalah menginisialisasi variabel-variabel yang akan digunakan di fungsi lainnya. Variabel dbJenaFuseki bernilai nama basis data Apache Jena Fuseki yang digunakan. READ\_FUSEKI adalah alamat URL basis data yang digunakan. OWL\_FILE\_LOCATION adalah ontologi yang dibuat pada subbab 3.3. fileRDF adalah alamat file hasil proses reasoning.

|  |
| --- |
| FileManager.*get*().addLocatorClassLoader(Main.**class**.getClassLoader()); Model Instances = FileManager.*get*().loadModel(READ\_FUSEKI); Instances.read(READ\_FUSEKI,**"RDF/XML"**);  Model famonto = FileManager.*get*().loadModel(OWL\_FILE\_LOCATION); |

Kode Sumber 5.2 Implementasi inisialisasi model Instance dan famonto

Fungsi diatas digunakan untuk memungkinkan aplikasi Jena membaca file RDF yang akan diekstraksi. Fungsi ini juga menginisialisasi model *Instance* dan model family ontologi.

|  |
| --- |
| FileManager fManager = FileManager.*get*(); fManager.addLocatorURL();  String[] persons = {  *//Indonesian Emperors* **"Raden\_Wijaya"**,  **"Hayam\_Wuruk"**,  **"Jayanegara"**,  **"Airlangga"**,  **}**  **for** (Integer counter = 0; counter < royalFamilies.**length**; counter++) {  Model modelActor = fManager.loadModel(**"http://dbpedia.org/data/"** + persons[counter] + **".ttl"**);   Instances.add(modelActor);  System.***out***.println(persons[counter]); } |

Kode Sumber 5.3 Implementasi ekstraksi file RDF tokoh

Fungsi diatas digunakan untuk membaca file RDF yang didownload, memodelkan file tersebut, menambahkan model tersebut ke *modelActor* dan mencetak URL *person* pada setiap iterasinya.

|  |
| --- |
| **final** Model union = ModelFactory.*createUnion*(Instances,famonto); |

Kode Sumber 5.4 Implementasi penggabungan model

Fungsi diatas digunakan untuk menggabungkan dua model menjadi satu model. Model-model yang digabungkan adalah *Instances* dan *famonto*, hasil penggabungan modelnya adalah *union*.

|  |
| --- |
| Reasoner reasoner = PelletReasonerFactory.*theInstance*().create(); InfModel reasonedModel = ModelFactory.*createInfModel*(reasoner,union); |

Kode Sumber 5.5 Implementasi proses reasoning

Fungsi diatas digunakan untuk melakukan proses *reasoning* pada suatu model. Model yang di*reasoning* adalah model *union.*

|  |
| --- |
| **if**(fileRDF.delete()) {  System.***out***.println(**"The old result.rdf file deleted successfully"**); } **else** {  System.***out***.println(**"Creating new result as RDF File"**); } PrintStream fileStream = **new** PrintStream(**"result.rdf"**); System.*setOut*(fileStream);  reasonedModel.write( System.***out***, **"RDF/XML"** ); |

Kode Sumber 5.6 Implementasi print hasil reasoning sebagai file RDF

Fungsi diatas digunakan untuk mengecek apakah file RDF hasil *reasoning* sudah ada atau belum. Jika sudah ada, maka akan dihapus, dan akan membuat file RDF baru lagi.

## Implementasi Antarmuka Pohon Keluarga

Pada bagian ini dijelaskan secara terperinci mengenai implementasi fungsi-fungsi yang digunakan dalam membangun sistem.

### Fungsi Dropdown Select

Fungsi *Dropdown Select* digunakan untuk menampilkan daftar entitas tokoh. Daftar nama tokoh ditampilkan dalam bentuk *form dropdown select.* Untuk menampilkannya, digunakan *method* get. Daftar entitas tokoh yang ditampilkan memiliki ciri khusus di basis data triple storenya, yaitu memiliki tipe kelas ‘Person’. *Query* allOfType digunakan untuk mendapatkan semua tipe ‘Person’ dari basis data *triple store*. Implementasi fungsi *dropdown select* dapat dilihat pada Kode Sumber 5.7.

|  |
| --- |
| $data = sparql\_get(**"localhost:3030/brits/query"**,  **"PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>    SELECT DISTINCT ?s  WHERE {  ?s rdf:type foaf:Person.  ?s foaf:name ?name  }"**);  **if** (!**isset**($data)) {  **print "<p>Error: "** . sparql\_errno() . **": "** . sparql\_error() . **"</p>"**; }  **?>** <**div class="row content"**>  <**div class="large-up-8"**>  <**div class="callout"**>  <**h6 class="subheader"**>PILIH TOKOH</**h6**>  <**form method="GET" action="#"**>  <**div class="input-group"**>  <**select class="input-group-field" name="entity"**>  **<?php  foreach** ($data **as** $row) {  **foreach** ($data->fields() **as** $name) {  **?>** <**option selected value="<?=** $row[$name] **?>"**>**<?=** $row[$name] **?>**</**option**>  **<?php** }  } **?>** </**select**>  <**div class="input-group-button"**>  <**input type="submit" class="button" name="submit" value="Submit"**>  </**div**>  </**div**>  </**form**>  </**div**>  </**div**> </**div**> |

Kode Sumber 5.7 Kode Sumber SPARQL untuk mengambil value bertipe Person dan Fungsi Dropdown Select

### Fungsi Get Family

Fungsi *Get Description* digunakan untuk menangkap masukan dari *dropdown select*. *Value* yang ditangkap kemudian berfungsi untuk mengakses informasi data yang bukan merupakan tautan. Fungsi ini memiliki beberapa sub fungsi berdasarkan kegunaan informasi yang diambil dari ontologi.

***Get* name**

Fungsi ini digunakan untuk mendapatkan nama dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari properti foaf:name yang melekat pada entitas. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.8.

|  |
| --- |
| $data\_name = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?name  WHERE {  <'** . $selected\_val . **'> foaf:name ?name  }  LIMIT 1'**); **if** (!**isset**($data\_name)) {  **print "<p>Error: "** . sparql\_errno() . **": "** . sparql\_error() . **"</p>"**; } **foreach** ($data\_name **as** $row) {  **foreach** ($data\_name->fields() **as** $field) {  **print "<h3>**$row[$field]**</h3>"**;  } } |

Kode Sumber 5.8 Fungsi Get name

***Get* father**

Fungsi ini digunakan untuk mendapatkan ayah dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari properti hasParent yang memiliki atribut foaf:gender male. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.9.

|  |
| --- |
| $data\_father = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?name  WHERE {  <'** . $selected\_val . **'>fam:hasParent ?fatherIRI.  ?fatherIRI foaf:name ?name. ?fatherIRI foaf:gender "male"@en  }  LIMIT 1'**);  $data\_fatherIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?fatherIRI  WHERE {  <'** . $selected\_val . **'> fam:hasParent ?fatherIRI.  ?fatherIRI foaf:name ?name. ?fatherIRI foaf:gender "male"@en  }  LIMIT 1'**);  **foreach** ($data\_fatherIRI **as** $row) {  **foreach** ($data\_fatherIRI->fields() **as** $field) {  $fatherIRI = $row[$field];  }  }  **if** (!**isset**($data\_father) || $data\_father == **''**) {  **echo "<ul>"**;  **echo "<li>"**;  **echo "<a>Father Unknown</a>"**;  **echo "-❤-"**;  } **else** {  **foreach** ($data\_father **as** $row) {  **foreach** ($data\_father->fields() **as** $field) {  **echo "<ul>"**;  **echo "<li>"**;  **echo '<a href="?entity='**.*urlencode*($fatherIRI).**'">'**.*str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**,$row[$field]).**'</a>'**;  **echo "-❤-"**;  }  }  } |

Kode Sumber 5.9 Fungsi Get father

***Get* mother**

Fungsi ini digunakan untuk mendapatkan ibu dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari properti hasParent yang memiliki atribut foaf:gender female. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.10.

|  |
| --- |
| $data\_mother = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?name  WHERE {  <'** . $selected\_val . **'> fam:hasParent ?motherIRI.  ?motherIRI foaf:name ?name. ?motherIRI foaf:gender "female"@en  }  LIMIT 1'**);  $data\_motherIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?motherIRI  WHERE {  <'** . $selected\_val . **'> fam:hasParent ?motherIRI.  ?motherIRI foaf:gender "female"@en  }'**);  **foreach** ($data\_motherIRI **as** $row) {  **foreach** ($data\_motherIRI->fields() **as** $field) {  $motherIRI = $row[$field];  }  }  **if** (!**isset**($data\_mother) || $data\_mother == **''**) {  **echo "<a>Mother Unknown</a>"**;  } **else** {  **foreach** ($data\_mother **as** $row) {  **foreach** ($data\_mother->fields() **as** $field) {  **echo '<a href="?entity='**.*urlencode*($motherIRI).**'">'**.*str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**,$row[$field]).**'</a>'**;  }  }  } |

Kode Sumber 5.10 Fungsi Get mother

***Get* sibling**

Fungsi ini digunakan untuk mendapatkan saudara dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari murni *query* SPARQL. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.11.

|  |
| --- |
| $data\_siblingIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT DISTINCT ?siblingIRI  WHERE {   <'** . $selected\_val . **'> fam:hasParent ?parentIRI.  ?parentIRI foaf:name ?name.  ?parentIRI fam:hasChild ?siblingIRI.  ?siblingIRI foaf:name ?siblingname  FILTER(?siblingIRI != <'** . $selected\_val . **'>)  }LIMIT 3'**);  $i=0; **foreach** ($data\_siblingIRI **as** $rowSiblingIRI) {  **foreach** ($data\_siblingIRI->fields() **as** $field) {  $siblingIRI[$i] = $rowSiblingIRI[$field];  $i++;  } }  **if** (!**isset**($data\_siblingIRI) || $data\_siblingIRI == **''**) {  **echo "<ul>"**; } **else** {  **echo "<ul>"**;  $i=0;  **foreach** ($data\_siblingIRI **as** $rowSiblingIRI) {  **foreach** ($data\_siblingIRI->fields() **as** $field) {  **echo "<li>"**;  $data\_sibling = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?siblingname  WHERE {   <'** .$siblingIRI[$i] . **'> foaf:name ?siblingname  }LIMIT 1'**);   **foreach** ($data\_sibling **as** $row) {  **foreach** ($data\_sibling->fields() **as** $field) {  **if** (*strlen*($row[$field]) > 20)  $row[$field] = *substr*($row[$field], 0, 15) . **'...'**;  **echo '<a href="?entity='** . *urlencode*($siblingIRI[$i]) . **'">'** . *str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**, $row[$field]) . **'</a>'**;  }  }  $i++;  **echo "</li>"**;  }  } } |

Kode Sumber 5.11 Fungsi Get sibling

***Get* spouse**

Fungsi ini digunakan untuk mendapatkan pasangan suami atau dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari properti isSpouseOf yang melekat pada entitas. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.12.

|  |
| --- |
| $data\_spouseIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT DISTINCT ?spouseIRI  WHERE {  <'** . $selected\_val . **'> fam:isSpouseOf ?spouseIRI.  ?spouseIRI foaf:name ?name  } LIMIT 1'**); $i=0; **foreach** ($data\_spouseIRI **as** $rowSpouseIRI) {  **foreach** ($data\_spouseIRI->fields() **as** $field) {  $spouseIRI[$i] = $rowSpouseIRI[$field];  $i++;  } }  **if** (!**isset**($data\_spouseIRI) || $data\_spouseIRI == **''**) {  **echo "-❤-<a>Spouse Unknown</a>"**; } **else** {  $i=0;  **foreach** ($data\_spouseIRI **as** $rowSpouseIRI) {  **foreach** ($data\_spouseIRI->fields() **as** $field) {  **echo "-❤-"**;  $data\_spouse = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?name  WHERE {  <'** . $spouseIRI[$i] . **'> foaf:name ?name  } LIMIT 1  '**);  **foreach** ($data\_spouse **as** $row) {  **foreach** ($data\_spouse->fields() **as** $field) {  **echo '<a href="?entity='** . *urlencode*($spouseIRI[$i]) . **'">'** . *str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**, $row[$field]) . **'</a>'**;  }  } |

Kode Sumber 5.12 Fungsi Get spouse

***Get* child**

Fungsi ini digunakan untuk mendapatkan keturunan dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari properti hasChild yang melekat pada entitas. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.13.

|  |
| --- |
| $data\_childIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT DISTINCT ?childIRI  WHERE {   <'** . $selected\_val . **'> fam:hasChild ?childIRI.  <'** .$spouseIRI[$i] . **'> fam:hasChild ?childIRI.  ?childIRI foaf:name ?childName  }LIMIT 10'**); $i++; $j=0; **foreach** ($data\_childIRI **as** $rowChildIRI) {  **foreach** ($data\_childIRI->fields() **as** $field) {  $childIRI[$j] = $rowChildIRI[$field];  $j++;  } } $flagChild = 0; **if** (**isset**($data\_childIRI)) {  **foreach** ($data\_childIRI **as** $rowChild) {  **foreach** ($data\_childIRI->fields() **as** $field) {  **if**($rowChild[$field] == **''**){  $flagChild = 0; *//tidak punya anak* }**else** $flagChild = 1;  }  }  **if**($flagChild == 1){  $cc=0;  **echo "<ul>"**;  **foreach** ($data\_childIRI **as** $rowChildIRI) {  **foreach** ($data\_childIRI->fields() **as** $field) {  **if**(**isset**($childIRI[$cc])) {  **echo "<li>"**;  $data\_child = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?childName  WHERE {  <'** . $childIRI[$cc] . **'> foaf:name ?childName  }LIMIT 1'**);  **foreach** ($data\_child **as** $rowChild) {  **foreach** ($data\_child->fields() **as** $field) {  **if** (*strlen*($rowChild[$field]) > 20)  $rowChild[$field] = *substr*($rowChild[$field], 0, 15) . **'...'**;  **echo '<a href="?entity='**.*urlencode*($childIRI[$cc]).**'">'**.*str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**,$rowChild[$field]).**'</a>'**;  }  } |

Kode Sumber 5.13 Fungsi Get child

***Get* child in law**

Fungsi ini digunakan untuk mendapatkan menantu dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari properti isSpouseOf yang melekat pada anak entitas terpilih. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.14.

|  |
| --- |
| $data\_ChildInLaw = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#> PREFIX foaf: <http://xmlns.com/foaf/0.1/> SELECT ?name WHERE {  <'** . $childIRI[$cc] . **'> fam:isSpouseOf ?sbj.  ?sbj foaf:name ?name }LIMIT 1'**); $data\_ChildInLawIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#> PREFIX foaf: <http://xmlns.com/foaf/0.1/> SELECT ?sbj WHERE {  <'** . $childIRI[$cc] . **'> fam:isSpouseOf ?sbj.  ?sbj foaf:name ?name }LIMIT 1'**); $cc++; } **foreach** ($data\_ChildInLawIRI **as** $rowChildInLawIRI) {  **foreach** ($data\_ChildInLawIRI->fields() **as** $field) {  $childInLawIRI = $rowChildInLawIRI[$field];  } } **if** (!**isset**($data\_ChildInLaw) || $data\_ChildInLaw == **''**) {  **echo "-❤-<a>?</a>"**; }**else if**(**isset**($data\_ChildInLaw)){  **foreach** ($data\_ChildInLaw **as** $rowChildInLaw) {  **foreach** ($data\_ChildInLaw->fields() **as** $field) {  **echo "-❤-"**;  **echo '<a href="?entity='** . *urlencode*($childInLawIRI) . **'">'** . *str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**, $rowChildInLaw[$field]) . **'</a>'**; |

Kode Sumber 5.14 Fungsi Get child in law

***Get* grand child**

Fungsi ini digunakan untuk mendapatkan cucu dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari anak menantu entitas dengan cara *query* SPARQL. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.15.

|  |
| --- |
| $data\_grandchildIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#> PREFIX foaf: <http://xmlns.com/foaf/0.1/> PREFIX dbp: <http://dbpedia.org/property/> SELECT DISTINCT ?grandchildIRI WHERE {   <'** . $childInLawIRI . **'> dbp:issue ?grandchildIRI.  ?grandchildIRI foaf:name ?name }'**); $m=0; **foreach** ($data\_grandchildIRI **as** $rowgrandChildIRI) {  **foreach** ($data\_grandchildIRI->fields() **as** $field) {  $grandchildIRI[$m] = $rowgrandChildIRI[$field];  $m++;  } } $flagGrandChild=0; **if** (**isset**($data\_grandchildIRI)) {  **foreach** ($data\_grandchildIRI **as** $rowGC) {  **foreach** ($data\_grandchildIRI->fields() **as** $field) {  **if** ($rowGC[$field] == **''**) {  $flagGrandChild = 0; *//tidak punya cucu* } **else** $flagGrandChild = 1;  }  }  **if**($flagGrandChild==1){  $n=0;  **echo "<ul>"**; *//garis vertikal cucu* **foreach** ($data\_grandchildIRI **as** $rowgrandChildIRI) {  **foreach** ($data\_grandchildIRI->fields() **as** $field) {  **echo "<li>"**;  $data\_grandchild = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?grandChildName  WHERE {  <'** . $grandchildIRI[$n] . **'> foaf:name ?grandChildName  }LIMIT 1'**);  **foreach** ($data\_grandchild **as** $rowGC) {  **foreach** ($data\_grandchild->fields() **as** $field) {  **if** (*strlen*($rowGC[$field]) > 20)  $rowGC[$field] = *substr*($rowGC[$field], 0, 15) . **'...'**;  **echo '<a href="?entity='** . *urlencode*($grandchildIRI[$n]) . **'">'** . *str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**, $rowGC[$field]) . **'</a>'**;  }  } |

Kode Sumber 5.15 Fungsi Get grand child

***Get* grand child in law**

Fungsi ini digunakan untuk mendapatkan pasangan cucu dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari anak menantu entitas dengan cara *query* SPARQL. Implementasi dari fungsi ini dapat dilihat pada Kode Sumber 5.16.

|  |
| --- |
| $data\_GrandChildInLaw = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?name  WHERE {  <'** . $grandchildIRI[$n] . **'> fam:isSpouseOf ?sbj.  ?sbj foaf:name ?name  }LIMIT 1'**); $data\_GrandChildInLawIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?sbj  WHERE {  <'** . $grandchildIRI[$n] . **'> fam:isSpouseOf ?sbj.  ?sbj foaf:name ?name  }LIMIT 1'**); **foreach** ($data\_GrandChildInLawIRI **as** $rowGrandChildInLawIRI) {  **foreach** ($data\_GrandChildInLawIRI->fields() **as** $field) {  $grandChildInLawIRI = $rowGrandChildInLawIRI[$field];  } } **if** (!**isset**($data\_GrandChildInLawIRI) || $data\_GrandChildInLawIRI == **''**) {  **echo "-❤-<a>?</a>"**; }**else if**(**isset**($data\_GrandChildInLawIRI)){  **foreach** ($data\_GrandChildInLaw **as** $rowGrandChildInLaw) {  **foreach** ($data\_GrandChildInLaw->fields() **as** $field) {  **echo "-❤-"**;  **echo '<a href="?entity='** . *urlencode*($grandChildInLawIRI) . **'">'** . *str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**, $rowGrandChildInLaw[$field]) . **'</a>'**; |

Kode Sumber 5.16 get grand child in law

***Get* great grand child**

Fungsi ini digunakan untuk mendapatkan cicit dari sebuah entitas yang dipilih. *Value* yang diambil berasal dari anak menantu entitas dengan cara *query* SPARQL. Implementasi dari fungsi ini dapat dilihat padaKode Sumber 5.17.

|  |
| --- |
| $data\_greatGrandChildIRI = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  PREFIX dbp: <http://dbpedia.org/property/>  SELECT DISTINCT ?greatgrandchildIRI  WHERE {   <'** . $grandChildInLawIRI . **'> fam:hasChild ?greatgrandchildIRI.  <'** . $grandchildIRI[$n] . **'> fam:hasChild ?greatgrandchildIRI.  ?greatgrandchildIRI foaf:name ?name  }'**); $m=0; **foreach** ($data\_greatGrandChildIRI **as** $rowgreatGrandChildIRI) {  **foreach** ($data\_greatGrandChildIRI->fields() **as** $field) {  $greatGrandChildIRI[$m] = $rowgreatGrandChildIRI[$field];  $m++;  } } $flagGreatGrandChild = 0; **if**(**isset**($data\_greatGrandChildIRI)){  **foreach** ($data\_greatGrandChildIRI **as** $rowGGC) {  **foreach** ($data\_greatGrandChildIRI->fields() **as** $field) {  **if** ($rowGGC[$field] == **''**) {  $flagGreatGrandChild = 0; *//tidak punya cicit* } **else** $flagGreatGrandChild = 1;  }  }  **if**($flagGreatGrandChild == 1){  $p=0;  **echo "<ul>"**;  **foreach** ($data\_greatGrandChildIRI **as** $rowGreatGrandChildIRI) {  **foreach** ($data\_greatGrandChildIRI->fields() **as** $field) {  **echo "<li>"**;  $data\_greatGrandChild = sparql\_get(**"localhost:3030/brits/query"**, **'PREFIX fam: <http://www.co-ode.org/roberts/family-tree.owl#>  PREFIX foaf: <http://xmlns.com/foaf/0.1/>  SELECT ?name  WHERE {   <'** . $greatGrandChildIRI[$p] . **'> foaf:name ?name  }LIMIT 1'**);  **foreach** ($data\_greatGrandChild **as** $rowGGC) {  **foreach** ($data\_greatGrandChild->fields() **as** $field) {  **if** (*strlen*($rowGGC[$field]) > 20)  $rowGGC[$field] = *substr*($rowGGC[$field], 0, 15) . **'...'**;  **echo '<a href="?entity='** . *urlencode*($greatGrandChildIRI[$p]) . **'">'** . *str\_replace*(**'http://www.dbpedia.org/resource/'**, **""**, $rowGGC[$field]) . **'</a>'**;  }  }  $p++;  **echo "</li>"**;  }  }  **echo "</ul>"**; |

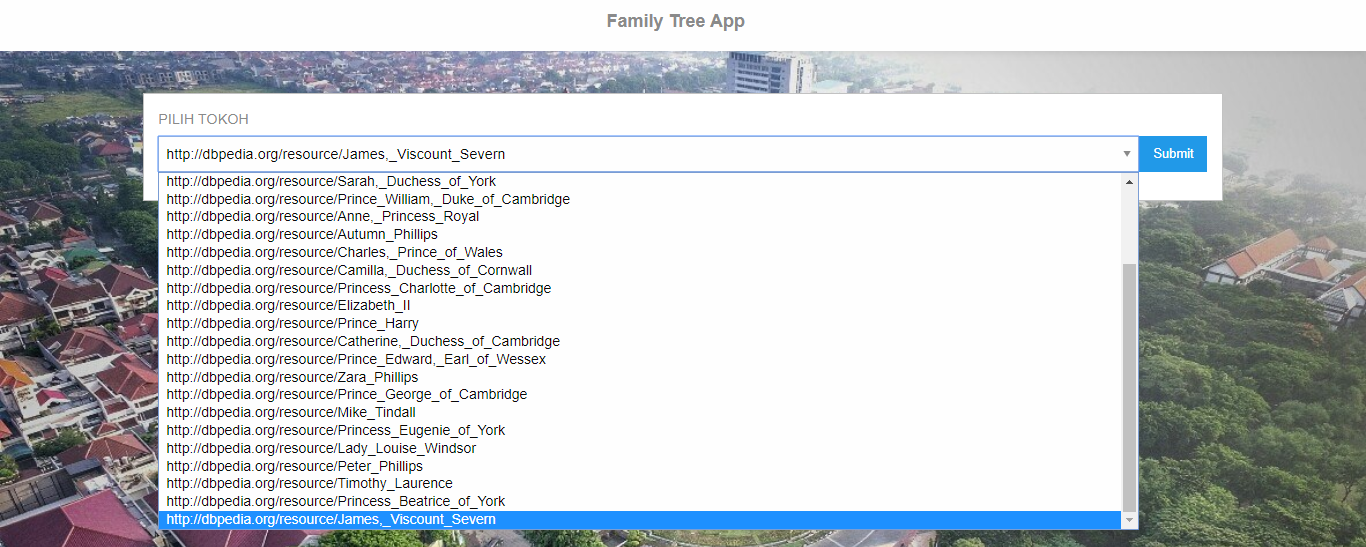
Kode Sumber 5.17 Get great grand child

## Implementasi Antarmuka Pengguna

Implementasi tampilan antarmuka pengguna pada *browser* Google Chrome dilakukan dengan menggunakan dukungan aplikasi XAMPP. XAMPP berfungsi untuk menjalankan aplikasi web dengan server Apache. Berikut ini akan dijelaskan mengenai implementasi tampilan antarmuka pengguna yang terdapat padaFamily Tree App.

### Implementasi Tampilan Halaman Utama

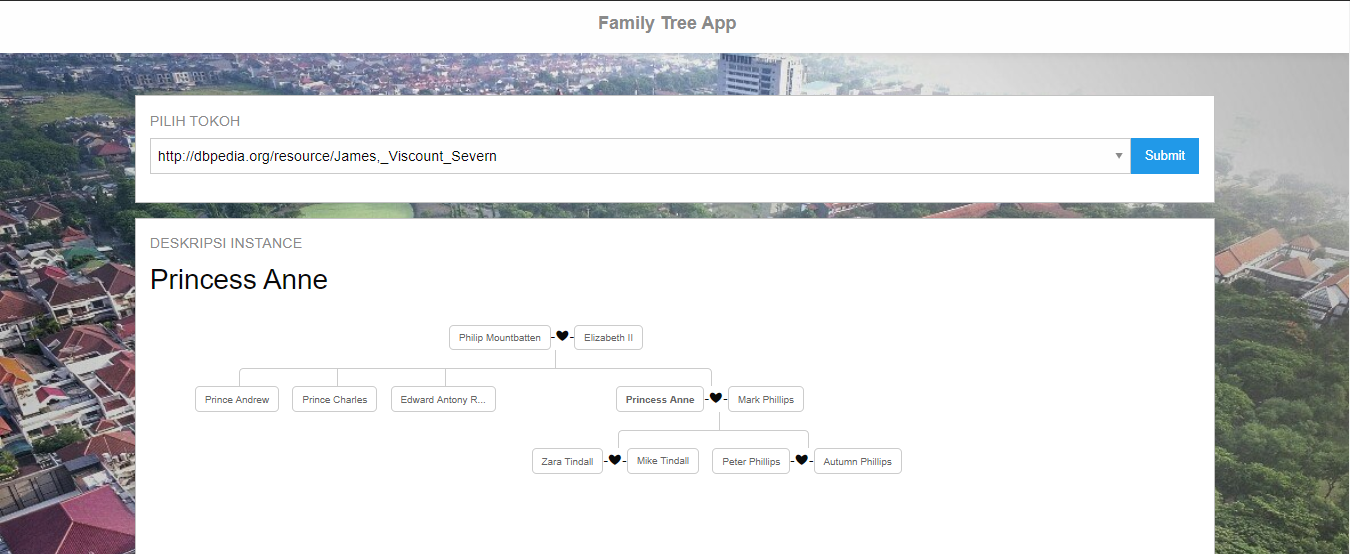
Halaman ini merupakan implementasi halaman utama dari rancangan antarmuka yang telah dijelaskan pada Subbab 4.2. Halaman utama hanya menampilkan kolom *dropdown select* yang dapat digunakan oleh pengguna untuk memilih tokoh. Daftar entitas tokoh yang ditampilkan hanya tokoh utama yang digunakan dalam pengerjaan tugas akhir ini. Tampilan antarmuka halaman utama ini dapat dilihat pada Gambar 5.1.



Gambar 5.1 Implementasi Antarmuka Halaman Utama

### Implementasi Tampilan Halaman Pohon Keluarga

Halaman ini merupakan implementasi halaman informasi untuk menampilkan data entitas yang dipilih dari rancangan antarmuka yang telah dijelaskan pada Subbab 4.2. Tampilan antarmuka halaman informasi ini dapat dilihat pada Gambar 5.2. Dan seperti yang dijelaskan di Batasan Masalah, batas generasi pendahulu adalah orang tua, sedangkan batas keturunan adalah cicit.



Gambar 5.2 Implementasi Antarmuka Halaman Pohon Keluarga

# BAB VI PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada ontologi yang dikembangkan. Pengujian yang dilakukan adalah pengujian ontologi, pengujian perbandingan data, dan pengujian kompleksitas ontologi. Pengujian ontologi mengacu pada perancangan *rule* pada Sub subbab Semantic Web Rule Language (SWRL). Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

## Lingkungan Pengujian

Lingkungan pengujian sistem pada pengerjaan Tugas Akhir ini dilakukan pada lingkungan dan alat kakas sebagai berikut:

Prosesor : Intel Core i7-6700

CPU @ 3.90GHz

Memori : 16.00 GB

Jenis *Device* : Laptop

Sistem Operasi : Microsoft Windows 10 64-bit

*Protege* : Protege 5.2

*Reasoner* : Pellet

*Browser* : Google Chrome

## Skenario Pengujian

Pada bagian ini akan dijelaskan tentang skenario pengujian yang dilakukan. Pengujian perbandingan data dilakukan dengan membandingkan data asli DBpedia dengan data yang dihasilkan dari pohon keluarga Family Tree App.

Pengujian data merupakan tahap uji kevalidan ontologi yang telah dibangun sebagai dasar proses pencarian relasi pohon keluarga tokoh sejarah dan kerajaan Indonesia. Pengujian dilakukan secara manual dengan mengecek situs DBpedia.

### Pengujian Perbandingan Data

Pengujian perbandingan data merupakan tahap uji setelah fungsionalitas *rule* atau SPARQL *query* telah dilaksanakan. Pengujian ini bertujaun untuk membandingkan kekayaan data. Data yang dibandingkan merupakan data asli DBpedia dengan data yang didapatkan dari hasil proses *reasoning*. Pengujian dilakukan secara mandiri dengan membandingkan data tokoh pada halaman DBpedia dengan data tokoh pada aplikasi web yang disertai dengan fakta-fakta baru.

#### Pengujian Perbandingan Data Mark Phillips

Pada tahap pengujian tokoh Mark Phillips, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.1.

Tabel 6.1 Pengujian Perbandingan Data Mark Phillips

|  |  |
| --- | --- |
| ID | TA-UJ.DT0001 |
| Nama | Pengujian perbandingan data Mark Phillips |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Mark Phillips pada halaman DBpedia. |
| Kondisi Awal | Individu Mark Phillips belum memiliki properti hasChildInLaw, atau hasGrandChild |
| Data Uji | Data uji merupakan data yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Mark Phillips pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia Mark Phillips yang beralamat *dbpedia.org/page/Mark\_Phillips*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Mark Philips. |
| Hasil Yang Diharapkan | Data individu Mark Phillips pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Mark Phillips, diantaranya yaitu hasChildInLaw dan hasGrandChild. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Mark Phillips pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Prince Andrew, Duke of York

Pada tahap pengujian tokoh Prince Andrew, fakta baru yang muncul dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.2.

Tabel 6.2 Pengujian Perbandingan Data Prince Andrew, Duke of York

|  |  |
| --- | --- |
| ID | TA-UJ.DT0002 |
| Nama | Pengujian perbandingan data Prince Andrew |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Prince Andrew pada halaman DBpedia. |
| Kondisi Awal | Individu Prince Andrew belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Prince Andrew pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia Prince Andrew yang beralamat *dbpedia.org/page/* *Prince\_Andrew,\_Duke\_of\_York*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Prince Andrew. |
| Hasil Yang Diharapkan | Data individu Prince Andrew pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Prince Andrew, yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Prince Andrew pada perangkat lunak Family Tree App lebih kaya dibandingkan data pada DBpedia. |

#### Pengujian Perbandingan Data Meghan Markle

Pada tahap pengujian tokoh Meghan Markle, fakta baru yang muncul dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.3.

Tabel 6.3 Pengujian Perbandingan Data Meghan Markle

|  |  |
| --- | --- |
| ID | TA-UJ.DT0003 |
| Nama | Pengujian perbandingan data Meghan Markle |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Meghan Markle pada halaman DBpedia. |
| Kondisi Awal | Individu Meghan Markle belum memiliki properti relasi keluarga. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Meghan Markle pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia Meghan Markle yang beralamat *dbpedia.org/page/Meghan\_Markle*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Meghan Markle. |
| Hasil Yang Diharapkan | Data individu Meghan Markle pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Meghan Markle, dikarenakan data DBpedia yang belum diperbarui. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Meghan Markle pada perangkat lunak Family Tree App sama kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Diana, Princess of Wales

Pada tahap pengujian tokoh Princess Diana, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.4.

Tabel 6.4 Pengujian Perbandingan Data Diana, Princess of Wales

|  |  |
| --- | --- |
| ID | TA-UJ.DT0004 |
| Nama | Pengujian perbandingan data Princess Diana |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Princess Diana pada halaman DBpedia. |
| Kondisi Awal | Individu Princess Diana belum memiliki properti hasGrandChild. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Princess Diana pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Diana,\_Princess\_of\_Wales*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Princess Diana. |
| Hasil Yang Diharapkan | Data individu Princess Diana pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Princess Diana, diantaranya yaitu hasGrandChild. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Princess Diana pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Sophie, Countess of Wessex

Pada tahap pengujian tokoh Sophie, Countess of Wessex, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.5.

Tabel 6.5 Pengujian Perbandingan Data Sophie, Countess of Wessex

|  |  |
| --- | --- |
| ID | TA-UJ.DT0005 |
| Nama | Pengujian perbandingan data Sophie, Countess of Wessex |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Sophie, Countess of Wessex pada halaman DBpedia. |
| Kondisi Awal | Individu Sophie, Countess of Wessex memiliki *value* properti *spouse* dan *issue*. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Sophie, Countess of Wessex pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Sophie,\_Countess\_of\_Wessex*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Sophie, Countess of Wessex. |
| Hasil Yang Diharapkan | Data individu Sophie, Countess of Wessex pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan *value* atau *property* baru. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Sophie, Countess of Wessex pada perangkat lunak Family Tree App sama dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Prince Philip, Duke of Edinburgh

Pada tahap pengujian tokoh Prince Philip, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.6.

Tabel 6.6 Pengujian Perbandingan Data Prince Philip, Duke of Edinburgh

|  |  |
| --- | --- |
| ID | TA-UJ.DT0006 |
| Nama | Pengujian perbandingan data Prince Philip |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Prince Philip pada halaman DBpedia. |
| Kondisi Awal | Individu Prince Philip belum memiliki properti hasChildInLaw, hasGrandChild, hasGrandChildInLaw dan hasGreatGrandChild. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Prince Philip pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Prince\_Philip,\_Duke\_of\_Edinburgh*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Prince Philip. |
| Hasil Yang Diharapkan | Data individu Prince Philip pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Prince Philip, diantaranya yaitu hasChildInLaw, hasGrandChild, hasGrandChildInLaw dan hasGreatGrandChild. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Prince Philip pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Sarah, Duchess of York

Pada tahap pengujian tokoh Sarah, Duchess of York, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.7.

Tabel 6.7 Pengujian Perbandingan Data Sarah, Duchess of York

|  |  |
| --- | --- |
| ID | TA-UJ.DT0007 |
| Nama | Pengujian perbandingan data Sarah, Duchess of York |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Mark Philips pada halaman DBpedia. |
| Kondisi Awal | Individu Sarah, Duchess of York memiliki *property* spouse dan issue. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Sarah, Duchess of York pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Sarah,\_Duchess\_of\_York.*  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Sarah, Duchess of York. |
| Hasil Yang Diharapkan | Data individu Sarah, Duchess of York pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Sarah, Duchess of York. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Sarah, Duchess of York pada perangkat lunak Family Tree App sama dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Prince William, Duke of Cambridge

Pada tahap pengujian tokoh Prince William, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.8.

Tabel 6.8 Pengujian Perbandingan Data Prince William, Duke of Cambridge

|  |  |
| --- | --- |
| ID | TA-UJ.DT0008 |
| Nama | Pengujian perbandingan data Prince William. |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Prince William pada halaman DBpedia. |
| Kondisi Awal | Individu Prince William belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Prince William pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Prince\_William,\_Duke\_of\_Cambridge*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Prince William. |
| Hasil Yang Diharapkan | Data individu Prince William pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Prince William, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Prince William pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Anne, Princess Royal

Pada tahap pengujian tokoh Princess Anne, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.9.

Tabel 6.9 Pengujian Perbandingan Data Anne, Princess Royal

|  |  |
| --- | --- |
| ID | TA-UJ.DT0009 |
| Nama | Pengujian perbandingan data Princess Anne |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Princess Anne pada halaman DBpedia. |
| Kondisi Awal | Individu Princess Anne belum memiliki properti hasSibling dan hasGrandChild. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Princess Anne pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page*/*Anne,\_Princess\_Royal*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Princess Anne. |
| Hasil Yang Diharapkan | Data individu Princess Anne pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Princess Anne, diantaranya yaitu hasSibling dan hasGrandChild. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Princess Anne pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Autumn Phillips

Pada tahap pengujian tokoh Autumn Philips, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.10.

Tabel 6.10 Pengujian Perbandingan Data Autumn Phillips

|  |  |
| --- | --- |
| ID | TA-UJ.DT0010 |
| Nama | Pengujian perbandingan data Autumn Phillips |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Autumn Phillips pada halaman DBpedia. |
| Kondisi Awal | Individu Autumn Philips memiliki properti hasChild. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Autumn Phillips pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Autumn\_Phillips*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Autumn Philips. |
| Hasil Yang Diharapkan | Data individu Autumn Phillips pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Autumn Phillips, dan properti hasChild tidak ada dikarenakan data spouse dari Autumn Phillips tidak mencantumkan properti hasChild dimana syarat hasChild adalah ketika dua orang memiliki *value* properti hasChild yang sama. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Autumn Phillips pada perangkat lunak Family Tree App sama dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Charles, Prince of Wales

Pada tahap pengujian tokoh Prince Charles, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.11.

Tabel 6.11 Pengujian Perbandingan Data Charles, Prince of Wales

|  |  |
| --- | --- |
| ID | TA-UJ.DT0011 |
| Nama | Pengujian perbandingan data Prince Charles |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Prince Charles pada halaman DBpedia. |
| Kondisi Awal | Individu Prince Charles belum memiliki properti hasSibling dan hasGrandChild. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Prince Charles pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Charles,\_Prince\_of\_Wales*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Prince Charles. |
| Hasil Yang Diharapkan | Data individu Prince Charles pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Prince Charles, diantaranya yaitu hasSibling dan hasGrandChild. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Prince Charles pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Camilla, Duchess of Cornwall

Pada tahap pengujian tokoh Camilla Rosemary, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.12.

Tabel 6.12 Pengujian Perbandingan Data Camilla, Duchess of Cornwall

|  |  |
| --- | --- |
| ID | TA-UJ.DT0012 |
| Nama | Pengujian perbandingan data Camilla Rosemary |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Mark Philips pada halaman DBpedia. |
| Kondisi Awal | Individu Camilla Rosemary memiliki properti *spouse*, *parent* dan *issue*. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Camilla Rosemary pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Camilla,\_Duchess\_of\_Cornwall*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Camilla Rosemary. |
| Hasil Yang Diharapkan | Data individu Camilla Rosemary pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Camilla Rosemary. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Camilla Rosemary pada perangkat lunak Family Tree App sama dengan data pada DBpedia dikarenakan *issue* dan *parent* dari Camilla bukanlah bagian dari *Royal Family*. |

#### Pengujian Perbandingan Data Princess Charlotte of Cambridge

Pada tahap pengujian tokoh Princess Charlotte, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.13.

Tabel 6.13 Pengujian Perbandingan Data Princess Charlotte of Cambridge

|  |  |
| --- | --- |
| ID | TA-UJ.DT0013 |
| Nama | Pengujian perbandingan data Princess Charlotte |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Princess Charlotte pada halaman DBpedia. |
| Kondisi Awal | Individu Princess Charlotte belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Princess Charlotte pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Princess\_Charlotte\_of\_Cambridge*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Princess Charlotte. |
| Hasil Yang Diharapkan | Data individu Princess Charlotte pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Princess Charlotte, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Princess Charlotte pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Elizabeth II

Pada tahap pengujian tokoh Ratu Elizabeth, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.14.

Tabel 6.14 Pengujian Perbandingan Data Elizabeth II

|  |  |
| --- | --- |
| ID | TA-UJ.DT0014 |
| Nama | Pengujian perbandingan data Ratu Elizabeth |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Ratu Elizabeth pada halaman DBpedia. |
| Kondisi Awal | Individu Ratu Elizabeth belum memiliki properti hasChildInLaw, hasGrandChild, hasGrandChildInLaw dan hasGreatGrandChild. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Ratu Elizabeth pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Elizabeth\_II*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Ratu Elizabeth. |
| Hasil Yang Diharapkan | Data individu Ratu Elizabeth pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Ratu Elizabeth, diantaranya yaitu hasChildInLaw, hasGrandChild, hasGrandChildInLaw dan hasGreatGrandChild. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Ratu Elizabeth pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Prince Harry

Pada tahap pengujian tokoh Prince Harry, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.15.

Tabel 6.15 Pengujian Perbandingan Data Prince Harry

|  |  |
| --- | --- |
| ID | TA-UJ.DT0015 |
| Nama | Pengujian perbandingan data Prince Harry |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Prince Harry pada halaman DBpedia. |
| Kondisi Awal | Individu Prince Harry belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Prince Harry pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Prince\_Harry*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Prince Harry. |
| Hasil Yang Diharapkan | Data individu Prince Harry pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Prince Harry, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Prince Harry pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Catherine, Duchess of Cambridge

Pada tahap pengujian tokoh Catherine Middleton, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.16.

Tabel 6.16 Pengujian Perbandingan Data Catherine, Duchess of Cambridge

|  |  |
| --- | --- |
| ID | TA-UJ.DT0016 |
| Nama | Pengujian perbandingan data Catherine Middleton |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Catherine Middleton pada halaman DBpedia. |
| Kondisi Awal | Individu Catherine Middleton memiliki properti hasChild, hasSpouse dan hasParent. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Catherine Middleton pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Catherine,\_Duchess\_of\_Cambridge*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Catherine Middleton. |
| Hasil Yang Diharapkan | Data individu Catherine Middleton pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Catherine Middleton |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Catherine Middleton pada perangkat lunak Family Tree App sama dengan data pada DBpedia dikarenakan properti hasParent milik Catherine Middleton bukanlah bagian dari Royal Family. |

#### Pengujian Perbandingan Data Prince Edward, Earl of Wessex

Pada tahap pengujian tokoh Prince Edward, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.17.

Tabel 6.17 Pengujian Perbandingan Data Prince Edward, Earl of Wessex

|  |  |
| --- | --- |
| ID | TA-UJ.DT0017 |
| Nama | Pengujian perbandingan data Prince Edward |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Prince Edward pada halaman DBpedia. |
| Kondisi Awal | Individu Prince Edward belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Prince Edward pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Prince\_Edward,\_Earl\_of\_Wessex*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Prince Edward. |
| Hasil Yang Diharapkan | Data individu Prince Edward pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Prince Edward, diantaranya yaitu hasSon dan hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Prince Edward pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Zara Phillips

Pada tahap pengujian tokoh Zara Phillips, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.18.

Tabel 6.18 Pengujian Perbandingan Data Zara Phillips

|  |  |
| --- | --- |
| ID | TA-UJ.DT0018 |
| Nama | Pengujian perbandingan data Zara Phillips |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Zara Phillips pada halaman DBpedia. |
| Kondisi Awal | Individu Zara Phillips belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Zara Phillips pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Zara\_Tindall*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Zara Phillips. |
| Hasil Yang Diharapkan | Data individu Zara Phillips pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Zara Phillips, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Zara Phillips pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Prince George of Cambridge

Pada tahap pengujian tokoh Prince George, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.19.

Tabel 6.19 Pengujian Perbandingan Data Prince George of Cambridge

|  |  |
| --- | --- |
| ID | TA-UJ.DT0019 |
| Nama | Pengujian perbandingan data Prince George |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Prince George pada halaman DBpedia. |
| Kondisi Awal | Individu Prince George belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Prince George pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Prince\_George\_of\_Cambridge*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Prince George. |
| Hasil Yang Diharapkan | Data individu Prince George pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Prince George, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Prince George pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Mike Tindall

Pada tahap pengujian tokoh Mike Tindall, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.20.

Tabel 6.20 Pengujian Perbandingan Data Mike Tindall

|  |  |
| --- | --- |
| ID | TA-UJ.DT0020 |
| Nama | Pengujian perbandingan data Mike Tindall |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Mark Philips pada halaman DBpedia. |
| Kondisi Awal | Individu Mike Tindall memiliki properti hasSpouse dan hasChild. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Mike Tindall pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Mike\_Tindall*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Mike Tindall. |
| Hasil Yang Diharapkan | Data individu Mike Tindall pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Mike Tindall. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Mike Tindall pada perangkat lunak Family Tree App sama dengan data pada DBpedia dikarenakan properti hasChild belum ada halaman DBpedianya. |

#### Pengujian Perbandingan Data Princess Eugenie of York

Pada tahap pengujian tokoh Princess Eugenie, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada Tabel 6.21

Tabel 6.21 Pengujian Perbandingan Data Princess Eugenie

|  |  |
| --- | --- |
| ID | TA-UJ.DT0021 |
| Nama | Pengujian perbandingan data Princess Eugenie |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Princess Eugenie pada halaman DBpedia. |
| Kondisi Awal | Individu Princess Eugenie belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Princess Eugenie pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Princess\_Eugenie\_of\_York*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Princess Eugenie. |
| Hasil Yang Diharapkan | Data individu Princess Eugenie pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Princess Eugenie, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Princess Eugenie pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Lady Louise Windsor

Pada tahap pengujian tokoh Lady Louise, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada

Tabel 6.22 Pengujian Perbandingan Data Lady Louise

|  |  |
| --- | --- |
| ID | TA-UJ.DT0022 |
| Nama | Pengujian perbandingan data Lady Louise |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Lady Louise pada halaman DBpedia. |
| Kondisi Awal | Individu Lady Louise belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Lady Louise pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Lady\_Louise\_Windsor*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Lady Louise. |
| Hasil Yang Diharapkan | Data individu Lady Louise pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Lady Louise, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Lady Louise pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Peter Phillips

Pada tahap pengujian tokoh Peter Phillips, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada

Tabel 6.23 Pengujian Perbandingan Data Peter Phillips

|  |  |
| --- | --- |
| ID | TA-UJ.DT0023 |
| Nama | Pengujian perbandingan data Peter Phillips. |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Peter Phillips pada halaman DBpedia. |
| Kondisi Awal | Individu Peter Phillips belum memiliki properti hasSibling.  Individu Peter Phillips memiliki properti *dbp:children* tetapi dalam bentuk String. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Peter Phillips pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Peter\_Phillips*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Peter Phillips. |
| Hasil Yang Diharapkan | Data individu Peter Phillips pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Peter Phillips, yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Peter Phillips pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. Data hasChild tidak ada dikarenakan data pada Peter Phillips tidak direpresentasikan sebagai DBpedia *resource*. |

#### Pengujian Perbandingan Data Timothy Laurence

Pada tahap pengujian tokoh Timothy Laurence, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada

Tabel 6.24 Pengujian Perbandingan Data Timothy Laurence

|  |  |
| --- | --- |
| ID | TA-UJ.DT0024 |
| Nama | Pengujian perbandingan data Timothy Laurence |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Timothy Laurence pada halaman DBpedia. |
| Kondisi Awal | Individu Timothy Laurence hanya memiliki properti hasSpouse. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Timothy Laurence pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Timothy\_Laurence*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Timothy Laurence. |
| Hasil Yang Diharapkan | Data individu Timothy Laurence pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Tidak didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Timothy Laurence. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Timothy Laurence pada perangkat lunak Family Tree App sama dengan data pada DBpedia. |

#### Pengujian Perbandingan Data Princess Beatrice of York

Pada tahap pengujian tokoh Princess Beatrice, fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada

Tabel 6.25 Pengujian Perbandingan Data Princess Beatrice

|  |  |
| --- | --- |
| ID | TA-UJ.DT0025 |
| Nama | Pengujian perbandingan data Princess Beatrice |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data Princess Beatrice pada halaman DBpedia. |
| Kondisi Awal | Individu Princess Beatrice belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas Princess Beatrice pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/Princess\_Beatrice\_of\_York*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia Princess Beatrice. |
| Hasil Yang Diharapkan | Data individu Princess Beatrice pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh Princess Beatrice, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu Princess Beatrice pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

#### Pengujian Perbandingan Data James, Viscount Severn

Pada tahap pengujian tokoh James, Viscount Severn fakta baru yang muncul setelah diterapkannya karakteristik dan sejumlah *rule* dibandingkan dengan data aslinya. Tujuannya adalah untuk mengetahui apakah fakta baru yang didapatkan sudah ada pada DBpedia atau belum. Jika fakta baru belum ada pada DBpedia, maka fakta tersebut akan melengkapi data yang dimiliki oleh DBpedia. Rincian pengujian perbandingan data ini dapat dilihat pada

Tabel 6.26 Pengujian Perbandingan Data James, Viscount Severn

|  |  |
| --- | --- |
| ID | TA-UJ.DT0026 |
| Nama | Pengujian perbandingan data James, Viscount Severn |
| Tujuan Pengujian | Menguji apakah fakta baru yang diperoleh dari Family Tree App sudah tersedia di DBpedia. |
| Skenario 1 | Melihat data pohon keluarga yang ditampilkan pada Family Tree App dan mencocokkannya dengan data James, Viscount Severn pada halaman DBpedia. |
| Kondisi Awal | Individu James, Viscount Severn belum memiliki properti hasSibling. |
| Data Uji | Data uji merupakan data hasil pengembangan ontologi yang ditampilkan pada perangkat lunak Family Tree App. |
| Langkah Pengujian | Pengguna memiih entitas James, Viscount Severn pada perangkat lunak Family Tree App.  Pengguna membuka halaman DBpedia yang beralamat *dbpedia.org/page/James,\_Viscount\_Severn*.  Pengguna membandingkan data pada perangkat lunak Family Tree App dengan halaman DBpedia James, Viscount Severn. |
| Hasil Yang Diharapkan | Data individu James, Viscount Severn pada perangkat lunak Family Tree App sama atau lebih kaya dibandingkan dengan data pada DBpedia. |
| Hasil Yang Didapat | Didapatkan properti dengan fakta-fakta baru yang belum dimiliki oleh DBpedia pada data tokoh James, Viscount Severn, diantaranya yaitu hasSibling. |
| Hasil Pengujian | Berhasil. |
| Kondisi Akhir | Data individu James, Viscount Severn pada perangkat lunak Family Tree App lebih kaya dibandingkan dengan data pada DBpedia. |

## Evaluasi Pengujian

Pada subbab ini akan diberikan hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang diberikan meliputi evaluasi pengujian perbandingan data yang telah dijelaskan pada Subbab 6.2.2.

### Evaluasi Pengujian Perbandingan Data

Rangkuman mengenai hasil pengujian perbandingan data dapat dilihat pada Tabel 6.27. Berdasarkan data pada tabel tersebut, semua skenario pengujian berhasil. Sehingga bisa ditarik disimpulkan bahwa ontologi yang dikembangkan telah sesuai dengan yang diharapkan.

Tabel 6.27 Rangkuman Hasil Pengujian

| ID | Nama | Properti DBpedia | Properti Family Tree App |
| --- | --- | --- | --- |
| TA-UJ.DT0001 | Mark Phillips | hasName, hasSpouse, hasChild | hasName, hasSpouse, hasChild, hasChildInLaw, hasGrandChild |
| TA-UJ.DT0002 | Prince Andrew, Duke of York | hasParent, hasName, hasSpouse, hasChild | hasParent, hasName, hasSpouse, hasChild, hasSibling |
| TA-UJ.DT0003 | Meghan Markle | hasName | hasName |
| TA-UJ.DT0004 | Diana, Princess of Wales | hasParent, hasName, hasSpouse, hasChild | hasName, hasSpouse, hasChild, hasChildInLaw, hasGrandChild |
| TA-UJ.DT0005 | Sophie, Countess of Wessex | hasParent, hasName, hasSpouse, hasChild | hasName, hasSpouse, hasChild |
| TA-UJ.DT0006 | Prince Philip, Duke of Edinburgh | hasParent, hasName, hasSpouse, hasChild | hasName, hasSpouse, hasChild, hasChildInLaw, hasGrandChild, hasGrandChildInLaw, hasGreatGrandChild |
| TA-UJ.DT0007 | Sarah, Duchess of York | hasParent, hasName, hasSpouse, hasChild | hasName, hasSpouse, hasChild |
| TA-UJ.DT0008 | Prince William | hasParent, hasName, hasSpouse, hasChild | hasParent, hasName, hasSpouse, hasChild, hasSibling |
| TA-UJ.DT0009 | Princess Anne | hasParent, hasName, hasSpouse, hasChild | hasParent, hasName, hasSpouse, hasChild, hasSibling, haChildInLaw, hasGrandChild |
| TA-UJ.DT0010 | Autumn Philips | hasParent, hasName, hasSpouse, hasChild | hasName, hasSpouse |
| TA-UJ.DT0011 | Prince Charles | hasParent, hasName, hasSpouse, hasChild | hasParent, hasName, hasSpouse, hasChild, hasSibling, hasChildInLaw, hasGrandChild |
| TA-UJ.DT0012 | Camilla Rosemary | hasParent, hasName, hasSpouse | hasName, hasSpouse |
| TA-UJ.DT0013 | Princess Charlotte | hasParent, hasName | hasParent, hasName, hasSibling |
| TA-UJ.DT0014 | Ratu Elizabeth | hasParent, hasName, hasSpouse, hasChild | hasName, hasSpouse, hasChild, hasChildInLaw, hasGrandChild, hasGrandChildInLaw, hasGreatGrandChild |
| TA-UJ.DT0015 | Prince Harry | hasParent, hasName | hasParent, hasName, hasSibling |
| TA-UJ.DT0016 | Catherine Middleton | hasParent, hasName, hasSpouse, hasChild | hasName, hasSpouse, hasChild |
| TA-UJ.DT0017 | Prince Edward | hasParent, hasName, hasSpouse, hasChild | hasParent, hasName, hasSpouse, hasChild, hasSibling |
| TA-UJ.DT0018 | Zara Phillips | hasParent, hasName, hasSpouse | hasParent, hasName, hasSpouse, hasSibling |
| TA-UJ.DT0019 | Prince George | hasParent, hasName | hasParent, hasName, hasSibling |
| TA-UJ.DT0020 | Mike Tindall | hasName, hasSpouse, hasChild | hasName, hasSpouse |
| TA-UJ.DT0021 | Princess Eugenie | hasParent, hasName | hasParent, hasName, hasSibling |
| TA-UJ.DT0022 | Lady Louise Windsor | hasParent, hasName | hasParent, hasName, hasSibling |
| TA-UJ.DT0023 | Peter Phillips | hasParent, hasName, hasSpouse, hasChild | hasParent, hasName, hasSpouse, hasSibling |
| TA-UJ.DT0024 | Timothy Laurence | hasName, hasSpouse | hasName, hasSpouse |
| TA-UJ.DT0025 | Princess Beatrice | hasParent, hasName | hasParent, hasName, hasSibling |
| TA-UJ.DT0026 | James, Viscount Severn | hasParent, hasName, hasChild | hasParent, hasName, hasSibling |

# BAB VII KESIMPULAN DAN SARAN

Pada bab ini dijelaskan mengenai kesimpulan dari hasil uji coba yang telah dilakukan dan saran mengenai hal-hal yang masih bisa untuk dikembangkan dari tugas akhir ini.



## Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi dan pengujian perangkat lunak yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Data properti yang dimiliki oleh *Family Relationship Ontology* dapat digunakan pada domain tokoh sejarah dan kerajaan Indonesia.
2. Studi kasus visualisasi pohon keluarga tokoh sejarah dan kerajaan Indonesia mampu dimodelkan dan digabungkan dengan model ontologi lokal dengan Apache Jena serta bisa melakukan proses *reasoning* dengan Pellet Reasoner.
3. Aplikasi untuk visualisasi pohon keluarga tokoh sejarah dan kerajaan Indonesia dapat dikembangkan dengan library SPARQL Lib yang mampu menghubungkan basis data Apache Jena Fuseki dengan perangkat lunak yang menggunakan bahasa pemrograman PHP.

## Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi dan pengujian yang telah dilakukan.

1. Penggunaan perangkat uji coba dengan spesfikasi kapasitas memori yang lebih besar agar waktu yang dibutuhkan untuk proses *export inferenced axiom* lebih cepat.
2. Penambahan visualisasi generasi pendahulu dan penerus.
3. Fitur penambahan data secara dinamis.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | L. Burmark, "Visual literacy: What you get is what you see," 2008. |
| [2] | Wikipedia, "Daftar Raja di Jawa," 09 06 2019. [Online]. Available: https://id.wikipedia.org/wiki/Daftar\_raja\_di\_Jawa. [Accessed 26 06 2019]. |
| [3] | S. J. Miller, Introduction to Ontology Concepts and Terminology, Lisbon, Portugal: University of Wisconsin-Milwaukee, 2013. |
| [4] | M. A. Ramadhanie, Penerapan Ontologi Objek Pembelajaran Untuk Kebutuhan Personalisasi E-Learning Berbasis Semantic Web, Depok: Universitas Indonesia, 2009. |
| [5] | S. Nikles, "Expressiveness of Enterprise Modelling Languages," University of Applied Sciences Northwestern Switzerland, Basel, 2010. |
| [6] | C. Candrabiantara, D. O. Siahaan and U. L. Yuhana, "Rancang Bangun Aplikasi Visualisasi Silsilah Keluarga Berbasis Ontologi," *Jurnal Teknik POMITS,* vol. 2, no. 1, 2013. |
| [7] | "Professor Robert Stevens," [Online]. Available: http://www.cs.man.ac.uk/~stevensr/ontology/family.rdf.owl. [Accessed 06 January 2016]. |
| [8] | G. Meditskos and N. Bassiliades, "A Rule-Based Object-Oriented OWL Reasoner," *IEEE Transactions on Knowledge and Data Engineering,* vol. 20, no. 3, pp. 397-410, 2008. |
| [9] | B. Parsia and E. Sirin, "Pellet: An OWL DL Reasoner," University of Maryland, College Park. |
| [10] | D. Wu and A. Håkansson, "A Method of Identifying Ontology Domain," *Procedia Computer Science,* vol. 35, pp. 504-513, 2014. |
| [11] | Kate Samuelson And Raisa Bruner, "Royal Family Tree," TIME, 06 05 2019. [Online]. Available: https://time.com/5238004/royal-family-tree/. [Accessed 23 06 2019]. |
| [12] | D. L. McGuinness and F. v. Harmelen, "OWL Web Ontology Language Overview," [Online]. Available: https://www.w3.org/TR/owl-features/. [Accessed 06 January 2016]. |
| [13] | Z. T. Inc., "An overview on PHP," Zend The PHP Company, 2007. |
| [14] | "XML," [Online]. Available: https://en.wikipedia.org/wiki/XML. [Accessed 10 June 2016]. |
| [15] | M. Saralita, "Pencarian Relasi Antar Tokoh Sejarah Indonesia Menggunakan Ontologi," 2016. |

# LAMPIRAN 1. DATA DBPEDIA



## Mark Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Mark Phillips |
| hasSpouse | http://dbpedia.org/resource/Anne,\_Princess\_Royal,  http://dbpedia.org/resource/Sandy\_Pflueger |
| hasChild | http://dbpedia.org/resource/Zara\_Phillips, http://dbpedia.org/resource/Peter\_Phillips |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Andrew

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | - |
| hasName | Prince Andrew, Duke of York |
| hasSpouse | http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York |
| hasChild | http://dbpedia.org/resource/Princess\_Beatrice\_of\_York, http://dbpedia.org/resource/Princess\_Eugenie\_of\_York |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Meghan Markle

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Megan Markle |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Diana

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Frances\_Shand\_Kydd, http://dbpedia.org/resource/John\_Spencer,\_8th\_Earl\_Spencer |
| hasSibling | - |
| hasName | Diana, Princess of Wales |
| hasSpouse | http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasChild | http://dbpedia.org/resource/Prince\_Henry\_of\_Wales, http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Sophie, Countess of Wessex

|  |  |
| --- | --- |
| Property | Value |
| hasParent | Mary O'Sullivan |
| hasSibling | - |
| hasName | Sophie, Countess of Wessex |
| hasSpouse | http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex |
| hasChild | http://dbpedia.org/resource/James,\_Viscount\_Severn, http://dbpedia.org/resource/Lady\_Louise\_Windsor |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Philip

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Princess\_Alice\_of\_Battenberg, http://dbpedia.org/resource/Prince\_Andrew\_of\_Greece\_and\_Denmark |
| hasSibling | - |
| hasName | Prince Philip, Duke of Edinburgh |
| hasSpouse | http://dbpedia.org/resource/Elizabeth\_II |
| hasChild | http://dbpedia.org/resource/Anne,\_Princess\_Royal, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex, http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Sarah, Duchess of York

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Susan\_Barrantes, http://dbpedia.org/resource/Ronald\_Ferguson |
| hasSibling | - |
| hasName | Sarah, Duchess of York |
| hasSpouse | http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasChild | http://dbpedia.org/resource/Princess\_Beatrice\_of\_York, http://dbpedia.org/resource/Princess\_Eugenie\_of\_York |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince William

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Diana,\_Princess\_of\_Wales, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasSibling | - |
| hasName | Prince William, Duke of Cambridge |
| hasSpouse | http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasChild | http://dbpedia.org/resource/Prince\_George\_of\_Cambridge, http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Anne

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | - |
| hasName | Anne, Princess Royal |
| hasSpouse | http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Timothy\_Laurence |
| hasChild | http://dbpedia.org/resource/Zara\_Phillips, http://dbpedia.org/resource/Peter\_Phillips |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Autumn Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | Brian Kelly  Kathleen McCarthy |
| hasSibling | - |
| hasName | Autumn Phillips |
| hasSpouse | http://dbpedia.org/resource/Peter\_Phillips |
| hasChild | http://dbpedia.org/resource/Isla\_Phillips, http://dbpedia.org/resource/Savannah\_Phillips |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Charles

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | - |
| hasName | Charles, Prince of Wales |
| hasSpouse | http://dbpedia.org/resource/Lady\_Diana\_Spencer, http://dbpedia.org/resource/Camilla\_Parker\_Bowles |
| hasChild | \* Prince William, Duke of Cambridge \* Prince Henry of Wales |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Camilla Rosemary

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Bruce\_Shand, http://dbpedia.org/resource/Rosalind\_Cubitt |
| hasSibling | - |
| hasName | Camilla, Duchess of Cornwall |
| hasSpouse | http://dbpedia.org/resource/Andrew\_Parker\_Bowles, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasChild | \* Tom Parker Bowles \* Laura Lopes |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Charlotte

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge, http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasSibling | - |
| hasName | Princess Charlotte |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Elizabeth II

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/George\_VI, http://dbpedia.org/resource/Queen\_Elizabeth\_The\_Queen\_Mother |
| hasSibling | - |
| hasName | Elizabeth II |
| hasSpouse | http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasChild | http://dbpedia.org/resource/Anne,\_Princess\_Royal, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex, http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Harry

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Diana,\_Princess\_of\_Wales, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasSibling | - |
| hasName | Prince Harry |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Catherine Middleton

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Carole\_Middleton, http://dbpedia.org/resource/Michael\_Francis\_Middleton |
| hasSibling | - |
| hasName | Catherine, Duchess of Cambridge |
| hasSpouse | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge |
| hasChild | http://dbpedia.org/resource/Prince\_George\_of\_Cambridge, http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Edward

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | - |
| hasName | Prince Edward, Earl of Wessex |
| hasSpouse | http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex |
| hasChild | http://dbpedia.org/resource/James,\_Viscount\_Severn, http://dbpedia.org/resource/Lady\_Louise\_Windsor |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Zara Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Anne,\_Princess\_Royal |
| hasSibling | - |
| hasName | Zara Phillips |
| hasSpouse | http://dbpedia.org/resource/Mike\_Tindall |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince George

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge, http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasSibling | - |
| hasName | Prince George |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Mike Tindall

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Mike Tindall |
| hasSpouse | http://dbpedia.org/resource/Zara\_Phillips |
| hasChild | mai tindall |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Eugenie

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York, http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York |
| hasSibling | - |
| hasName | Princess Eugenie of York |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Lady Louise Windsor

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex |
| hasSibling | - |
| hasName | Lady Louise Windsor |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Peter Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Anne,\_Princess\_Royal |
| hasSibling | - |
| hasName | Peter Phillips |
| hasSpouse | http://dbpedia.org/resource/Autumn\_Phillips |
| hasChild | Isla Phillips  Savannah Phillips |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Timothy Laurence

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Timothy Laurence |
| hasSpouse | http://dbpedia.org/resource/Anne,\_Princess\_Royal |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Beatrice

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York, http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York |
| hasSibling | - |
| hasName | Princess Beatrice of York |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## James, Viscount Severn

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex |
| hasSibling | - |
| hasName | James, Viscount Severn |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

***[Halaman ini sengaja dikosongkan]***

# LAMPIRAN 2. DATA DALAM FAMILY TREE APP



## Mark Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Mark Phillips |
| hasSpouse | http://dbpedia.org/resource/Anne,\_Princess\_Royal,  http://dbpedia.org/resource/Sandy\_Pflueger |
| hasChild | http://dbpedia.org/resource/Zara\_Phillips, http://dbpedia.org/resource/Peter\_Phillips |
| hasChildInLaw | http://dbpedia.org/resource/Autumn\_Phillips, http://dbpedia.org/resource/Mike\_Tindall |
| hasGrandChild | http://dbpedia.org/resource/Savannah\_Phillips,  http://dbpedia.org/resource/Isla\_Phillips |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Andrew

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales, http://dbpedia.org/resource/Anne,\_Princess\_Royal, |
| hasName | Prince Andrew, Duke of York |
| hasSpouse | http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York |
| hasChild | http://dbpedia.org/resource/Princess\_Beatrice\_of\_York, http://dbpedia.org/resource/Princess\_Eugenie\_of\_York |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Meghan Markle

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Megan Markle |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Diana

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Diana, Princess of Wales |
| hasSpouse | http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasChild | http://dbpedia.org/resource/Prince\_Henry\_of\_Wales, http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge |
| hasChildInLaw | http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasGrandChild | http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge, http://dbpedia.org/resource/Prince\_George\_of\_Cambridge |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Sophie, Countess of Wessex

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Sophie, Countess of Wessex |
| hasSpouse | http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex |
| hasChild | http://dbpedia.org/resource/James,\_Viscount\_Severn, http://dbpedia.org/resource/Lady\_Louise\_Windsor |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Philip

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Prince Philip, Duke of Edinburgh |
| hasSpouse | http://dbpedia.org/resource/Elizabeth\_II |
| hasChild | http://dbpedia.org/resource/Anne,\_Princess\_Royal, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex, http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasChildInLaw | http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York, http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex, http://dbpedia.org/resource/Diana,\_Princess\_of\_Wales |
| hasGrandChild | http://dbpedia.org/resource/Princess\_Beatrice\_of\_York,  http://dbpedia.org/resource/Princess\_Eugenie\_of\_York, http://dbpedia.org/resource/Zara\_Phillips, http://dbpedia.org/resource/Peter\_Phillips,  http://dbpedia.org/resource/Lady\_Louise\_Windsor, http://dbpedia.org/resource/James,\_Viscount\_Severn, http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge,  http://dbpedia.org/resource/Prince\_Harry |
| hasGrandChildInLaw | http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasGreatGrandChild | http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge, http://dbpedia.org/resource/Prince\_George\_of\_Cambridge |

## Sarah, Duchess of York

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Sarah, Duchess of York |
| hasSpouse | http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasChild | http://dbpedia.org/resource/Princess\_Beatrice\_of\_York, http://dbpedia.org/resource/Princess\_Eugenie\_of\_York |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince William

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Diana,\_Princess\_of\_Wales, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasSibling | http://dbpedia.org/resource/Prince\_Harry |
| hasName | Prince William, Duke of Cambridge |
| hasSpouse | http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasChild | http://dbpedia.org/resource/Prince\_George\_of\_Cambridge, http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Anne

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex,  http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasName | Anne, Princess Royal |
| hasSpouse | http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Timothy\_Laurence |
| hasChild | http://dbpedia.org/resource/Zara\_Phillips, http://dbpedia.org/resource/Peter\_Phillips |
| hasChildInLaw | http://dbpedia.org/resource/Autumn\_Phillips, http://dbpedia.org/resource/Mike\_Tindall |
| hasGrandChild | http://dbpedia.org/resource/Savannah\_Phillips,  http://dbpedia.org/resource/Isla\_Phillips |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Autumn Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Autumn Phillips |
| hasSpouse | http://dbpedia.org/resource/Peter\_Phillips |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Charles

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | http://dbpedia.org/resource/Anne,\_Princess\_Royal,  http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex,  http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasName | Charles, Prince of Wales |
| hasSpouse | http://dbpedia.org/resource/Lady\_Diana\_Spencer, http://dbpedia.org/resource/Camilla\_Parker\_Bowles |
| hasChild | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge,  http://dbpedia.org/resource/Prince\_Harry |
| hasChildInLaw | http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasGrandChild | http://dbpedia.org/resource/Prince\_George\_of\_Cambridge,  http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Camilla Rosemary

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Camilla, Duchess of Cornwall |
| hasSpouse | http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Charlotte

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge, http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasSibling | http://dbpedia.org/resource/Prince\_George\_of\_Cambridge |
| hasName | Princess Charlotte |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Elizabeth II

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Elizabeth II |
| hasSpouse | http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasChild | http://dbpedia.org/resource/Anne,\_Princess\_Royal, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex, http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasChildInLaw | http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York, http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex, http://dbpedia.org/resource/Diana,\_Princess\_of\_Wales |
| hasGrandChild | http://dbpedia.org/resource/Princess\_Beatrice\_of\_York,  http://dbpedia.org/resource/Princess\_Eugenie\_of\_York, http://dbpedia.org/resource/Zara\_Phillips, http://dbpedia.org/resource/Peter\_Phillips,  http://dbpedia.org/resource/Lady\_Louise\_Windsor, http://dbpedia.org/resource/James,\_Viscount\_Severn, http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge,  http://dbpedia.org/resource/Prince\_Harry |
| hasGrandChildInLaw | http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasGreatGrandChild | http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge, http://dbpedia.org/resource/Prince\_George\_of\_Cambridge |

## Prince Harry

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Diana,\_Princess\_of\_Wales, http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales |
| hasSibling | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge |
| hasName | Prince Harry |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Catherine Middleton

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Catherine Elizabeth |
| hasSpouse | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge |
| hasChild | http://dbpedia.org/resource/Prince\_George\_of\_Cambridge, http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince Edward

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Elizabeth\_II, http://dbpedia.org/resource/Prince\_Philip,\_Duke\_of\_Edinburgh |
| hasSibling | http://dbpedia.org/resource/Charles,\_Prince\_of\_Wales,  http://dbpedia.org/resource/Anne,\_Princess\_Royal, http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York |
| hasName | Prince Edward, Earl of Wessex |
| hasSpouse | http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex |
| hasChild | http://dbpedia.org/resource/James,\_Viscount\_Severn, http://dbpedia.org/resource/Lady\_Louise\_Windsor |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Zara Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Anne,\_Princess\_Royal |
| hasSibling | http://dbpedia.org/resource/Peter\_Phillips |
| hasName | Zara Phillips |
| hasSpouse | http://dbpedia.org/resource/Mike\_Tindall |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Prince George

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_William,\_Duke\_of\_Cambridge, http://dbpedia.org/resource/Catherine,\_Duchess\_of\_Cambridge |
| hasSibling | http://dbpedia.org/resource/Princess\_Charlotte\_of\_Cambridge |
| hasName | Prince George |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Mike Tindall

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Mike Tindall |
| hasSpouse | http://dbpedia.org/resource/Zara\_Phillips |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Eugenie

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York, http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York |
| hasSibling | http://dbpedia.org/resource/Princess\_Beatrice\_of\_York |
| hasName | Princess Eugenie of York |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Lady Louise Windsor

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex |
| hasSibling | http://dbpedia.org/resource/James,\_Viscount\_Severn |
| hasName | Lady Louise Windsor |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Peter Phillips

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Mark\_Phillips, http://dbpedia.org/resource/Anne,\_Princess\_Royal |
| hasSibling | http://dbpedia.org/resource/Zara\_Phillips |
| hasName | Peter Phillips |
| hasSpouse | http://dbpedia.org/resource/Autumn\_Phillips |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Timothy Laurence

|  |  |
| --- | --- |
| Property | Value |
| hasParent | - |
| hasSibling | - |
| hasName | Timothy Laurence |
| hasSpouse | http://dbpedia.org/resource/Anne,\_Princess\_Royal |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## Princess Beatrice

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Prince\_Andrew,\_Duke\_of\_York, http://dbpedia.org/resource/Sarah,\_Duchess\_of\_York |
| hasSibling | http://dbpedia.org/resource/Princess\_Eugenie\_of\_York |
| hasName | Princess Beatrice of York |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

## James, Viscount Severn

|  |  |
| --- | --- |
| Property | Value |
| hasParent | http://dbpedia.org/resource/Sophie,\_Countess\_of\_Wessex, http://dbpedia.org/resource/Prince\_Edward,\_Earl\_of\_Wessex |
| hasSibling | http://dbpedia.org/resource/Lady\_Louise\_Windsor |
| hasName | James, Viscount Severn |
| hasSpouse | - |
| hasChild | - |
| hasChildInLaw | - |
| hasGrandChild | - |
| hasGrandChildInLaw | - |
| hasGreatGrandChild | - |

***(Halaman ini sengaja dikosongkan)***

# BIODATA PENULIS

Faiq, lahir pada tanggal 8 Juli 1997 di Kediri. Penulis pernah menempuh pendidikan di SDIT Nurul Islam Pare (2003-2007) SD Islam Ar-Robithoh (2007-2009), MTs Negeri 1 Pare (2009-2012), dan SMA Negeri 2 Kediri (2013-2015).

Saat ini penulis sedang menempuh pendidikan perguruan tinggi di Institut Teknologi Sepuluh Nopember Surabaya di departemen Informatika Fakultas Teknologi Informasi dan Komunikasi angkatan tahun 2015. Dalam menyelesaikan pendidikan S1 penulis mengambil bidang minat Manajemen Informasi (MI). Penulis juga pernah terlibat aktif dalam organisasi kemahasiswaan serta kepanitiaan selama perkuliahan, antara lain staff Departemen Hubungan Luar di Himpunan Mahasiswa Teknik Computer-Informatika ITS, dan menjadi kabinet dalam organisasi BEM FTIK ITS. Di sisi profesional, penulis pernah melakukan kerja praktek di Blibli.com, Direktorat Pengembangan Teknologi dan Sistem Informasi (DPTSI) – ITS, dan PT. Aku Pintar Indonesia. Penulis dapat dihubungi melalui alamat *email* karyoutomoo@gmail.com.