INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA

Development of an Ontology of Tourist Attractions for Recommending Points of Interest in a Group Recommender System for Tourism

DAVID LUÍS CORREIA DA SILVA

Outubro de 2023

POLITÉCNICO
DO PORTO

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Development of an Ontology of Tourist Attractions for Recommending Points of Interest in a Group Recommender System for Tourism

David Silva

**A dissertation submitted in partial fulfillment of
the requirements for the degree of Master of Science,
Specialisation Area of Graphics Systems and Multimedia**

Supervisor: Goreti Marreiros
Co-Supervisor: Patrícia Alves

Evaluation Committee:
President:
Dulce Mota

Members:
Gabriel Santos

Porto, October 16, 2023

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I have not plagiarised or applied any form of undue use of information or falsification of results along the process leading to its elaboration.

Therefore the work presented in this document is original and authored by me, having not previously been used for any other end.

I further declare that I have fully acknowledged the Code of Ethical Conduct of P.PORTO.

ISEP, Porto, October 16, 2023

Dedicatory

To my family, my friends, professors and classmates: thank you all for being an integral part of my academic journey!

Abstract

In recent years, the tourism industry has witnessed substantial growth, thanks to the proliferation of digital technology and online platforms. Tourists now have greater access to information and the ability to make informed travel decisions. However, the abundance of available information often leaves tourists overwhelmed when selecting points of interest (POI) that align with their preferences. Recommender Systems (RS) have emerged as a solution, personalising recommendations based on tourist behaviour, social networks, and contextual factors. To enhance RS efficacy, researchers have begun exploring the integration of psychological factors, such as personality traits. Yet, to meet the demands of modern tourists, a robust knowledge base, such as a tourist attractions ontology, is essential for seamless and rapid matching of tourist characteristics and preferences with available POI.

With that in mind, this project aims to enhance a Group Recommender System (GRS) prototype, GrouPlanner, by creating a robust tourist attractions ontology. This ontology will facilitate rapid and accurate matching of points of interest with tourists' characteristics, including personality, preferences, and demographic data, ultimately improving POI recommendations.

First, there needs to be an understanding of the personality of tourists and how it influences their choices when it comes to picking the best point of interest based on their personality. With that knowledge acquired, it is time to choose a way to represent this knowledge in the form of an ontology.

In this project, the Protégé ontology editor was used to design the ontology and the relationships between the tourists' personality and the points of interest. After designing the ontology, it had to be converted to a database so the Grouplanner system could access it. So, to do that, a solution was designed to integrate the designed ontology in a triple store data base, in this case, Apache Fuseki.

With the database implemented, several tests were made to verify if the database would give the recommended points of interests based on the tourists' preferences. This tests were later analysed.

Keywords: Point of interest, Recommendation system, Tourism, Ontology, Personality

Resumo

Nos anos mais recentes, a indústria do turismo presenciou um crescimento substancial devido à tecnologia digital e plataformas online. Cada vez mais, os turistas têm acesso a uma abundância de informação que influencia a habilidade de tomar decisões sobre viajar. No entanto, esta informação pode complicar a seleção dos pontos de interesse que alinhem com as preferências dos turistas. Para combater isso, sistemas de recomendação (SR) emergiram como uma solução, personalizando as recomendações com base no comportamento do turista, redes socias e outros fatores. Para aumentar a eficácia destes sistemas, os investigadores começaram a explorar a possibilidade de integração com fatores psicológicos, como traços de personalidade. Apesar disso, para cumprir as exigências dos turistas modernos, uma base de conhecimento robusta, como uma ontologia de atrações turísticas, é essencial para, de forma eficaz e eficiente, corresponder as características dos turistas com os pontos de interesse disponíveis.

Com isso em mente, este projeto tem como objetivo melhorar um protótipo de um sistema de recomendação (GrouPlanner), criando uma ontologia robusta de atrações turísticas. Essa ontologia facilitará a correspondência rápida e precisa de pontos de interesse com as características dos turistas, incluindo a sua personalidade e as suas preferências, melhorando assim as recomendações de pontos de interesse.

Em primeiro lugar, é necessário compreender a personalidade dos turistas e como ela influencia as suas escolhas ao selecionar o melhor ponto de interesse com base na sua personalidade. Com esse ponto adquirido, é necessário escolher uma maneira de representar esse conhecimento na forma de uma ontologia.

Neste projeto, o editor de ontologias Protégé foi utilizado para projetar a ontologia e as relações entre a personalidade dos turistas e os pontos de interesse. Após a construção da ontologia, foi necessário convertê-la numa base de dados para que o sistema Grouplanner pudesse ter acesso. Para isso, foi desenhada uma solução para integrar a ontologia projetada numa base de dados "triple store", neste caso, o Apache Fuseki.

Com a base de dados implementada, foram realizados vários testes para verificar se esta forneceria os pontos de interesse recomendados com base nas preferências dos turistas. Esses testes foram depois analisados.

Palavras-chave: Ponto de interesse, Sistema de recomendação, Turismo, Ontologia, Personalidade

Acknowledgement

First, I would like to thank my friends and family for always pushing me forward and supporting me.

Second, I would like to thank all my professors and classmates at Instituto Superior de Engenharia do Porto for everything I have learned throughout this journey. In particular, I would like to thank my supervisor Goreti Marreiros and my co-supervisor Patrícia Alves for helping me accomplish the development of this project and finishing this dissertation.

I have nothing but gratitude to express and I want to thank you all for being an integral part of my academic journey.

Contents

List of Figures

List of Tables

List of Source Code

List of Acronyms

ACID	Atomicity, Consistency, Isolation, Durability.
API	Application Programming Interface.
CAFOB	Context-aware fuzzy-ontology-based tourism recommendation system.
ETL	Extract, Transform, Load.
FFE	Fuzzy Front End.
GIS	Geographic Information System.
HTTPS	Hyper Text Transfer Protocol Secure.
JSON	JavaScript Object Notation.
NPD	New Product Development.
OWL	Web Ontology Language.
POI	Point of Interest.
RDF	Resource Description Framework.
REST	Representational State Transfer.
RS	Recommendation System.
SPARQL	SPARQL Protocol and RDF Query Language.
SQL	Structured Query Language.
SWOT	Strengths, Weaknesses, Opportunities, and Threats.
UI	User Interface.
URI	Uniform Resource Identifier.

Chapter 1

Introduction

1.1 Context

This dissertation was conducted within the TMDEI program at the Research Group on Intelligent Engineering and Computing for Advanced Innovation and Development (GECAD), Instituto Superior de Engenharia do Porto (ISEP). It forms part of a larger project aimed at developing the GrouPlanner prototype (GECAD 2023), a sophisticated tourism recommendation system tailored for group travel. The system capitalises on digital technology to deliver personalised recommendations, addressing the challenge of information overload in the tourism industry.

1.2 Problem

In recent years, the tourism industry has experienced a significant growth in both volume and diversity of tourists. With the rise of digital technologies and online platforms, tourists have more access to information and more power to make informed decisions about their travel choices (Company 2023). However, with the overwhelming amount of information available, tourists often face difficulties in selecting the best points of interest (POI) that match their preferences and expectations.

To address this issue, Recommender Systems (RS) have emerged as a promising solution to provide personalised recommendations to tourists. These systems are designed to analyse the preferences and characteristics of each tourist and suggest relevant POI based on their past behaviour, social network, or other contextual factors. However, traditional RS methods that only consider tourist behaviour and social factors may not be sufficient to provide accurate and effective recommendations.

To improve the effectiveness of RS, researchers have started to explore the use of psychological factors, such as personality traits, as an additional input for the recommendation algorithms (Tkalcic and Chen 2015). By incorporating psychological factors into the RS, the system can understand the emotional and sensory experiences that tourists are seeking and provide them with more personalised and satisfactory recommendations.

However, that alone is not enough to satisfy the increasing demands of modern tourists. To provide faster and more accurate recommendations, a solid knowledge base is needed to match the tourist's characteristics and preferences with the available POI. To achieve this, a tourist attractions ontology (a structured database of concepts and relationships that can be used to identify and match POI with tourists' preferences and characteristics) can be created. With the use of such an ontology, RS can more seamlessly and rapidly provide accurate recommendations to satisfy the growing expectations of tourists.

Recommender Systems (RS) are increasingly being studied and developed to help users make more satisfactory choices, being the tourism domain one of the most widely studied (Massimo and Ricci 2022). As leisure tourism involves emotional and sensorial experiences, personalization is increasingly being perceived as an important factor for the effectiveness of RS, and therefore, the more about the tourist is known, better recommendations of points of interest (POI) can be made. The use of psychological aspects to generate recommendations, such as personality, is a growing trend in RS and they are being studied to provide more personalised approaches, since they have shown to produce better results than generic approaches and can even help in the cold-start problem. However, just to know the users' characteristics is not enough. With increasingly demanding consumers, the RS need to provide faster and more accurate recommendations. Therefore, it is important to have a solid knowledge base so the POI can be rapidly and seamlessly matched with the tourists' characteristics and preferences, which can be solved with the creation of a tourist attractions ontology.

1.3 Objectives

This work aims to improve RS for tourism, namely an existing Group RS (GRS) prototype (GrouPlanner), by providing a robust ontology of tourist attractions so the POI can be rapidly, diversely,and accurately matched to the tourists' characteristics (personality, fears, tourist preferences, and demographic information), and consequently improve the POI recommendation lists. In this work, the main objective was to develop a tourist attractions ontology and integrate it in the Recommendation Engine and POI Microservices of the GRS prototype to improve the recommendation process. The following results are expected:

- Contextualisation on the State of the Art of:
  - (Group) Recommender Systems;
  - Psychology of Tourism and Destination Personality;
  - Personality and User Preferences;
  - Knowledge representation and reasoning (Ontologies).
- Formalisation and development of an ontology of tourist attractions, including all the relevant POI attributes needed to provide more accurate and personalised recommendations;
- Codification and integration of the ontology into the Recommendation Engine Microservices of the existing GRS prototype;
- Testing of the ontology and POI recommendations using real use-case scenarios;
- Results analysis and dissertation writing.

1.4 Approach

With the problem in hand, the goal was to develop an ontology of tourist attractions that can recommend points of interest based on the tourist's characteristics and preferences and implement it in a group recommender system for tourism.

To do this, several studies about recommendation systems, the psychology of tourism and destination personality, user's preferences and their personality and how to make knowledge representation and reasoning (including ontologies) was made. After these studies, an ontology was formalised and developed to include all the relevant POI attributes needed to provide more accurate and personalised recommendations. After the formalisation and development, it will was expected to be coded and integrated into the Recommendation Engine and POI Microservices of the existing GRS prototype and tested using real use-case scenarios.

The solution development followed an agile method for software development using SCRUM, with bimonthly sprints, guided by defined objectives and requirements. At the end of each "sprint", the functionalities were continuously tested and, if necessary, the defined objectives will be adjusted.

This solution will also be evaluated using real use-case scenarios, which means it will be tested using different users with different personalities and see if the recommendation system was providing them with the accurate POI.

1.5 Document Structure

The report is broken down into seven chapters. Each contains a set of divisions that organise the content for easier and more enjoyable reading. The following chapters are included: Introduction; Current State of the Art; Value Analysis; Analysis and Design; Implementation; Evaluation and Experimentation; Conclusions.

In the first (the current one), it is given an introduction of the current problem relating to recommendation systems and tourism preferences, the objectives that are going to be attempted, the approach that is going to be used and the structure of the document.

The State of the Art is were all the studying, reading and investigating is documented to help the development process. First there is an introduction to recommendation systems in tourism and how tourists' personality influences their decisions. After that, in focuses in understanding knowledge representation and reasoning, searching also examples of recommendation systems that use ontologies. Finally, this chapter focuses on searching the right tools to implement the previous research.

The value analysis chapter follows, in which a few solutions are found, measured, and tied to the major problem and research questions, in order to determine which solutions perform best to provide both the best results and the highest level of user happiness.

Following the value analysis, the analysis and design chapter will analyse, refine, and arrange the previously chosen solution in a position where the developer can take it up and apply it. This is accomplished by developing both functional and non-functional criteria. The design of the ontology was also a key component that was documented in this chapter.

After that is the implementation chapter, which explains and desiccates the development in a way that shows how each of the user stories were produced with the help of development photos and code samples of the most general portions.

With the solution implemented, a new chapter called evaluation and experiment occurs, in which the author conducts extensive tests that will serve to assess the performance of the recommendation system on both a quantitative and qualitative level. The measurements and requirements for the tests are also listed here for a better testing experience.

Finally, the conclusions chapter summarises all that was done, created, and tested during the project's building. It is also a place to consider all of the advantages and downsides of the development process, what could have gone better, and the project's future.

Chapter 2

Context and State of the Art

2.1 Recommender Systems In Tourism

In the modern day, travelling as become more of a norm due to globalisation (Elena 2020). Tourism is becoming more of an activity that almost everyone takes part in. That being said, there is a lot of information to be accounted for when planning a holiday trip. A lot of time can be consumed just by searching and organising activities, places to dine, hotels to stay. A solution to this problem could be a recommender system.

Recommender systems are information filtering systems that use data and machine learning algorithms to suggest items to users based on their preferences and past behaviours. These systems are designed to help users navigate an overwhelming amount of information by filtering out irrelevant items and presenting personalised recommendations (Borràs, Moreno, and Valls 2014).

In the context of tourism, recommender systems can be particularly helpful in assisting travellers with the search and organisation of activities during a holiday trip. Tourists are often unfamiliar with the places they are visiting, which can make it challenging to select activities that match their preferences. Recommender systems can help with this by providing personalised recommendations based on a user's profile. User profiles typically include information about the user's preferences, which can be gathered in a variety of ways, including explicit questionnaires, social group associations, or implicit observation of the user's interaction with the system. The quality of the recommendations can be improved through the use of Artificial Intelligence representation languages and inference tools. The distribution of recommended activities can also be influenced by the spatial distribution of activities and visitors (Alves, Carneiro, Novais, et al. 2019). In order to deal with this, the combination of Artificial Intelligence and Geographic Information Systems (GIS) can be used in a Recommender System to provide an appropriate way to deal with spatial data during the recommendation process. This technology can reduce travel planning time by providing personalised assistance (Moreno et al. 2013).

Recommendation systems in tourism can be based on a variety of algorithms, including collaborative filtering, content-based filtering, and hybrid algorithms that combine elements of both (Amzad and Vijayalakshmi 2021).

- Collaborative filtering uses the past behaviour of a group of users to make recommendations. For example, if multiple users have rated a particular hotel highly, the system might recommend that hotel to a new user who has similar preferences.
- Content-based filtering analyses the attributes of items to make recommendations. For example, a content-based recommendation system for travel might analyse the

features of a hotel (e.g. location, amenities) to suggest other hotels with similar features.

- Hybrid filtering combines both collaborative filtering and content-based filtering to make recommendations. Hybrid systems can be more accurate than either type of system used alone.

One of the key benefits of recommendation systems in tourism is that they can help travellers save time and effort by filtering out destinations, attractions, and travel experiences that are unlikely to be of interest. This makes the travel planning process more efficient and enjoyable for the traveller, as they are able to focus on a smaller number of options that are more likely to meet their needs and interests. They also have the potential to help travellers discover new destinations, attractions, and travel experiences that they may not have considered otherwise. By analysing large amounts of data about traveller preferences, travel behaviours, and destination characteristics, these systems can generate recommendations that are highly personalised and tailored to each traveller's specific needs and interests (Alves, Saraiva, et al. 2022).

2.2 Psychology of Tourism and Personality

Understanding the psychology of tourism is essential to design effective marketing strategies and developing successful tourist destinations. By understanding the personality traits and preferences of potential visitors, tourist destinations can cater to their needs and preferences and create an immersive and engaging experience that they will want to return to.

Destination personality plays a critical role in this process, as it refers to the unique set of qualities and characteristics that define a place and make it stand out from other destinations. These qualities can include the physical landscape, cultural attractions, historical landmarks, and other unique features that contribute to the overall appeal of the destination.

Tourists' personality influences their preferences for destinations and attractions, and this relationship has been explored using five broad dimensions of personality: openness, conscientiousness, extroversion, agreeableness, and neuroticism. These dimensions are widely used to describe individual differences in personality and have been found to play a role in tourists' decision-making processes (Alves, Domingos, et al. 2020).

- Openness refers to the degree to which an individual is imaginative, curious, and open to new experiences. Tourists who score high on this dimension tend to seek unique and diverse destinations, and are interested in cultural, historical, and artistic attractions.

- Conscientiousness refers to the degree to which an individual is organised, dependable, and responsible. Tourists who score high on this dimension tend to prefer well-planned and structured itineraries and may be more interested in visiting theme parks and natural parks.

- Extroversion refers to the degree to which an individual is sociable, talkative, and assertive. Tourists who score high on this dimension tend to enjoy lively and bustling destinations, and may be interested in visiting shopping centres, night clubs, and entertainment parks.

- Agreeableness refers to the degree to which an individual is friendly, cooperative, and compassionate. Tourists who score high on this dimension tend to prefer harmonious and friendly destinations, and may be interested in visiting museums, parks, and wildlife reserves.

- Neuroticism refers to the degree to which an individual experiences negative emotion, such as anxiety and stress. Tourists who score high on this dimension tend to prefer relaxing and tranquil destinations, and may be interested in visiting spas, beaches, and gardens.

As mentioned above, destination personality is a crucial factor in the selection of tourist attractions and destinations. However, there is also a need to understand the tourists' personality dimensions, to help in the development of more personalised and effective recommender systems for the tourism industry.

That being said, personality can be defined as an individual's stable and distinct traits that shape their behaviour, attitudes, and preferences. In the context of tourism, a person's personality can have a significant impact on their preferences for different types of tourist attractions. To assess that, a study was conducted by (Alves, Domingos, et al. 2020), and later extended (Alves, Carneiro, Saraiva, et al. 2023) that collected data from tourists through a personality assessment tool and a survey. The personality assessment tool was used to gather information about the tourists' personality traits, while the survey was used to gather information about their preferences for different types of tourist attractions. The data was then analysed to determine the relationship between personality traits and preferences for tourist attractions.

The study found that personality traits significantly influence tourists' preferences for tourist attractions. For instance, individuals with high levels of openness tend to prefer cultural and natural attractions, while those with high levels of extroversion prefer social activities and nightlife. Tourists who score high in agreeableness prefer leisure and entertainment, while those who score high in conscientiousness prefer cultural attractions and outdoor activities. Finally, tourists who score high in neuroticism tend to prefer relaxing activities and places with a low level of stimulation (see figure 2.1).

Based on these findings, it is possible to conclude that incorporating personality information into Recommender systems can improve the accuracy of tourist attraction recommendations and provide a more personalised and effective experience for tourists.

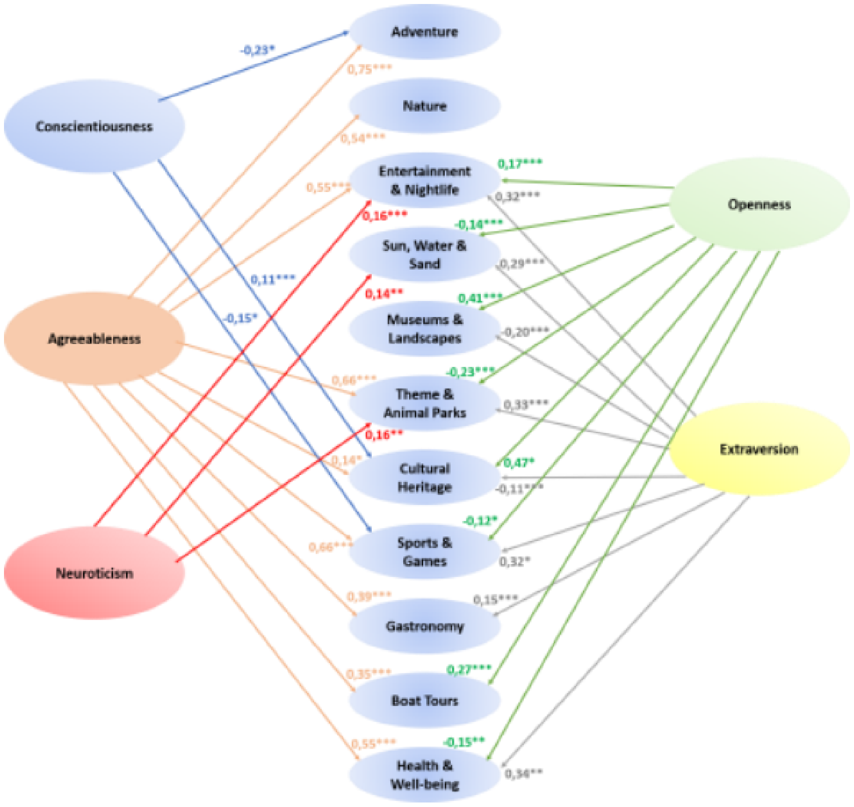


Figure 2.1: Simplified Structural Equation Model for the proposed "Personality-Tourist Attractions Preference" model. For readability, only the statistically significant values are presented (* p < 0.05 (2-tailed), ** p < 0.01 (2-tailed), *** p < 0.001 (2-tailed)) (retrieved from Alves, Domingos, et al. 2020)

2.3 Knowledge representation and reasoning

Artificial intelligence and computational reasoning rely heavily on knowledge representation and reasoning. They entail the creation of computational models capable of representing, manipulating, and reasoning about knowledge. These models help with problem solving, decision making, and other cognitive tasks (Autoblocks 2023).

Knowledge representation is the process of creating a symbolic representation of knowledge that can be manipulated by a computer program, typically in the form of a formal language. This representation captures the relationships and constraints that govern concepts, entities, and events. Semantic networks, frames, description logics, and rule-based systems are examples of knowledge representation formalisms.

The process of drawing inferences from knowledge to support problem-solving and decision-making is known as reasoning. There are various kinds of reasoning, including deductive, inductive, and abductive reasoning. Deductive reasoning infers conclusions from premises using logical rules. Inductive reasoning makes generalisations and predictions by using examples. Abductive reasoning begins with observations and proceeds to propose the most likely explanation for those observations (JavaPoint 2021).

Ontology, the study of the categories and concepts used to describe the world, is an important area of knowledge representation and reasoning. An ontology is a formal definition of the concepts and relationships in a domain that can be used to create a shared understanding of the domain among humans and computers.

2.3.1 Ontologies

An ontology can be defined as a hierarchical structure of concepts that define the terms and relationships in a specific domain. It can also be seen as a formal specification of a shared conceptualisation of a particular domain (V. Siricharoen 2007). It is a representation of knowledge that specifies the concepts and relationships within a particular domain, and is used to facilitate knowledge sharing and reuse. It seeks to reduce or eliminate conceptual and terminological confusion among the members of a user community who need to share various kinds of electronic documents and information (Navigli, Velardi, and Gangemi 2003).

In terms of the representation of the information, an ontology has a set of interrelated concepts and their relationships, organised in a hierarchical or network structure (Borràs, Moreno, and Valls 2014). The concepts are usually represented as classes, and the relationships between them are typically taxonomical and non-taxonomical. Taxonomical relationships are those that express subsumption or inheritance, such as the "is-a" relationship between a "car" and a "vehicle". Non-taxonomical relationships are those that express other types of semantic relationships between concepts, such as "part-of" or "located-in". Ontologies can be represented using formal languages, such as the Web Ontology Language (OWL) (Group 2012), which is used on the Semantic Web. These languages provide a way to formally define concepts, relationships, and constraints, and they enable computers to reason about and manipulate the knowledge represented in an ontology. To construct an ontology, specialists from several fields must thoroughly analyse the domain by examining the vocabulary that describes the entities that populate it, developing formal descriptions of the terms in that vocabulary, and characterising the conceptual relations that hold among or within those terms.

An ontology may also include axioms, which are logical statements that express constraints or rules that govern the domain, in addition to concepts and relationships. Instances are another important aspect of an ontology. They are used to populate the ontology with real-world data by representing specific objects or entities in the domain.

Ontologies make it easier to share and integrate knowledge from various sources, as well as reason about the relationships between concepts in a formal and well-defined manner (Borràs, Moreno, and Valls 2014). It also promotes interpersonal communication and cooperation, improved enterprise organisation, and system interoperability. It offers advantages in system engineering such as reusability, reliability, and specification. Another significant advantage of ontologies is their ability to support the automatic discovery of new knowledge. An ontology-based system, for example, can use the relationships and constraints specified in an ontology to discover new connections between concepts and infer new knowledge.

Ontologies are typically used in intelligent systems, including recommender systems, to represent the knowledge about the domain that the system needs to reason and make recommendations. In the context of e-tourism, an ontology can be used to represent the knowledge and concepts related to tourism, such as tourist attractions, events, and activities, and can help to facilitate the exchange and sharing of tourism-related information between different systems and applications.

2.4 Recommendation Systems that use Ontologies

There are already some recommendation systems in place that give a customised experience to the user using ontologies. The first one that is going to be presented is SigTur/E-Destination.

2.4.1 SigTur/E-Destination

SigTur is an ontology-based tourism recommendation system that takes into account various sources of information, such as demographic data, contextual details of the travel group, geographical aspects, and user feedback, to provide personalised recommendations to users (Moreno et al. 2013). The system uses a variety of recommendation techniques, including stereotypes, content-based filtering, and collaborative filtering.

One of SigTur's distinguishing features is its use of ontologies to guide the recommendation process. A domain ontology is used in SigTur to represent knowledge about the tourism domain, such as information about the various activities, attractions, and resources available in a specific area.

The domain ontology is designed to fit the specification of the particular area where the system is deployed. For example, the ontology used by SigTur for the "Camp de Tarragona and Terres de l'Ebre" area includes a detailed level of concepts related to wine, reflecting the importance of enotourism in the region.

SigTur makes a knowledge-level analysis of the user preferences, including processes that make bottom-up and top-down propagation of the preferences over the concepts of the ontology. This allows the system to make inferences about the correspondence between the characteristics of an activity and a certain user profile. The system also associates a certain degree of confidence to each specific recommendation, which is useful in order to take the final decision of which activities to show to the user.

Finally, the system includes GIS tools to store the main tourism and leisure resources with geospatial information, which is used to recommend the activities and to show the results in a user-friendly map-based Web application.

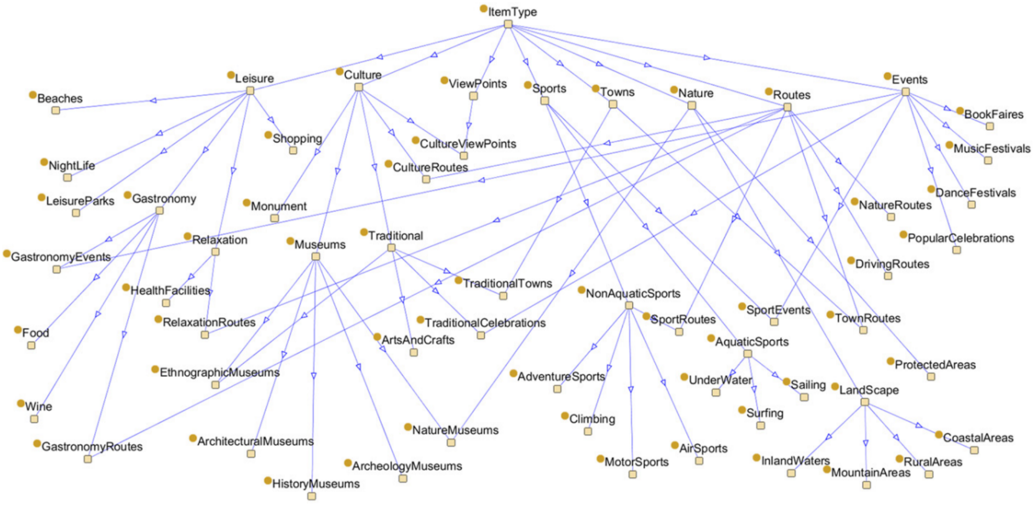


Figure 2.2: Partial view of the SigTur's Tourism ontology (retrieved from Moreno et al. 2013)

In terms of architecture, SigTur/E-Destination is divided into four main layers: the data layer, the domain layer, the recommendation layer, and the user interface layer.

- The data layer is where all the information about the tourism resources and the user data is stored. This includes demographic data, travel context data, user preferences and feedback, and the GIS database of tourism resources with geospatial information.
- The domain layer includes the domain ontology, which represents the concepts and relationships of the tourism domain. It also includes the ontology management system, which is responsible for managing the ontology, and the inference engine, which uses the ontology to make recommendations.
- The recommendation layer includes the recommendation engine, which uses various recommendation techniques, such as stereotypes, content-based filtering, and collaborative filtering, to generate recommendations for the user. The recommendation engine also uses the ontology and the user data to make inferences and generate personalised recommendations.
- The user interface layer provides the interface for the user to interact with the system. This includes the map-based web application, which displays the recommended activities and attractions, and allows the user to interact with the system by providing feedback and preferences.

Overall, SigTur/E-Destination is an integrated system that uses a variety of techniques and technologies to provide personalised tourism recommendations to users. The use of ontologies and GIS tools are unique features of the system that allow it to take into account both the spatial and conceptual aspects of the tourism domain.

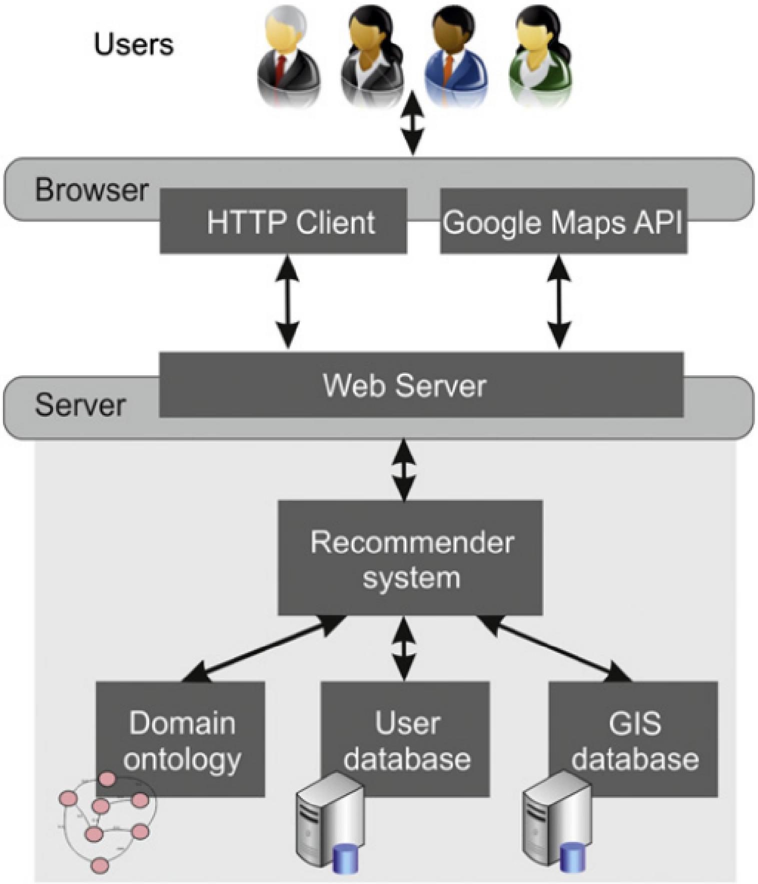


Figure 2.3: Architecture of SigTur/E-Destination (retrieved from Moreno et al. 2013)

2.4.2 CAFOB (Context-aware fuzzy-ontology-based tourism recommendation system)

CAFOB (Context-aware fuzzy-ontology-based tourism recommendation system) is a computer-based system designed to provide tourism recommendations to users based on their preferences and contextual information (Abbasi-Moud et al. 2022). The system uses fuzzy ontology to handle uncertainty and imprecision in user preferences and contextual data, and to represent knowledge about the tourism domain. It takes into account the user's location, time, and other contextual information to provide personalised recommendations for tourism activities, such as sightseeing, dining, and entertainment. The system also considers the user's preferences and interests, as well as their budget constraints.

The ontology used in the CAFOB system represents the tourism domain by defining concepts, relationships, and properties that are relevant to tourism activities. For example, the ontology may define concepts such as "tourist attraction", "restaurant", and "hotel", as well as their relationships, such as "is located in", "has a rating of", and "offers a discount for".

This ontology is then used by the system to model the user's preferences and contextual information. For example, the system could represent the user's preference for "romantic" restaurants in the ontology as a fuzzy concept defined by a set of rules and membership functions that determine how strongly the user's preferences match different restaurants.

In this way, the use of ontologies allows the CAFOB system to provide personalised and context-aware recommendations based on the user's preferences and contextual information, while also handling uncertainty and imprecision in the data.

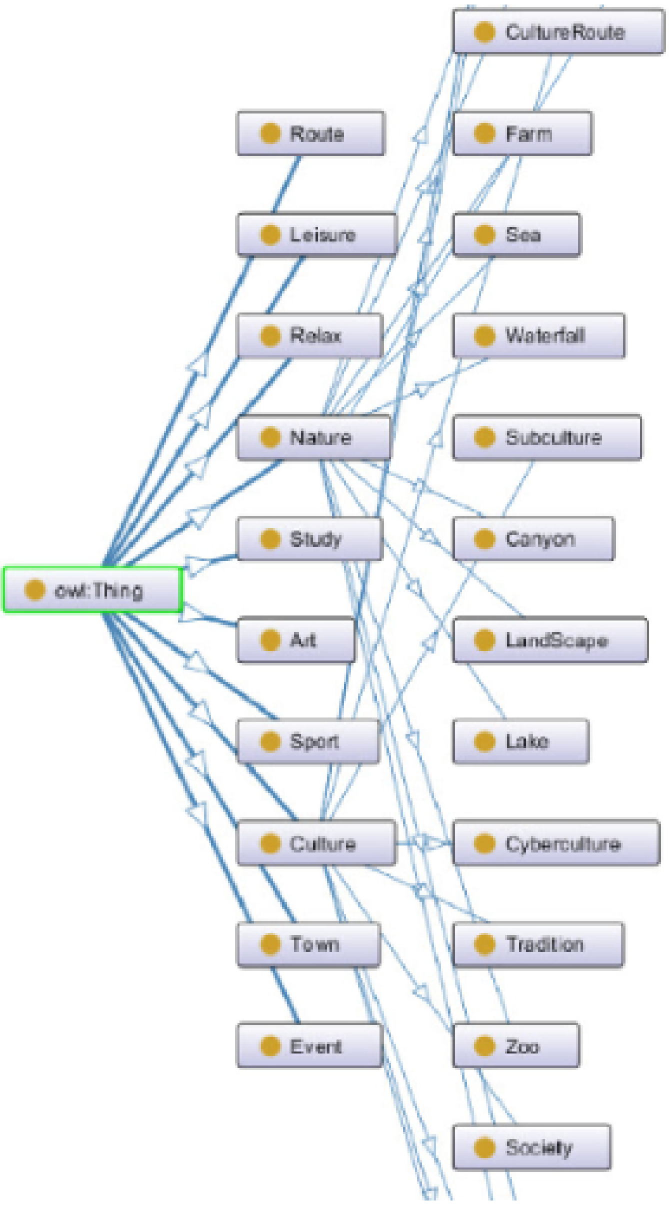


Figure 2.4: A partial visualisation of the proposed conventional ontology (retrieved from Abbasi-Moud et al. 2022)

In terms of architecture, it may vary depending on the implementation. However, the general architecture can be described as a combination of three main components: data sources, the recommendation engine, and the user interface.

- In terms of data sources, the CAFOB system relies on several data sources to provide recommendations to users. These data sources include tourism data, such as information about tourist attractions, restaurants, and hotels, as well as contextual data, such as the user's location, time, and weather. The system may also incorporate data from social media and other external sources to enhance its recommendations.

- The recommendation engine is the CAFOB system's core component, which processes data from various sources and generates personalised recommendations for the user. The recommendation engine employs fuzzy ontology to model tourism domain knowledge and represent the user's preferences and contextual information. It uses machine learning algorithms to refine its recommendations over time by applying a set of rules to infer recommendations based on the user's input and contextual data.
- The user interface is the component of the CAFOB system that allows the user to interact with the system and receive recommendations. The user interface can take many forms, such as a mobile application or a web-based interface, and may incorporate features such as search, filtering, and social sharing. The user interface may also provide additional information about the recommended tourism activities, such as reviews, ratings, and directions.

Overall, the CAFOB system's architecture is intended to provide personalised and context-aware recommendations to users by combining data from various sources, modelling domain knowledge with fuzzy ontology, and providing an intuitive and user-friendly interface for interaction.

2.5 Protégé

To represent the ontology and construct its domain model, there are a multitude of tools that stand out. However, due to previous experience and a familiarity with the tool, the one that was chosen for that purpose was Protégé.

Protégé is a widely used and powerful open-source software tool for creating and managing ontologies. It is primarily used in the field of knowledge representation and the development of semantic web applications (University 2020). Here's what Protégé is useful for:

- **Knowledge Representation**: Users can use Protégé to describe and organise knowledge in a machine-processable format. This is required for a wide range of applications, including artificial intelligence, natural language processing, data integration, and expert systems.
- **Customised Ontology Creation**: Protégé features a straightforward interface for developing ontologies. Users can specify classes, properties, and their relationships. It supports a variety of ontology languages, including OWL (Web Ontology Language) and RDF (Resource Description Framework), allowing users to select the formalism that is best suited to their project.
- **Collaboration and Sharing**: Protégé facilitates collaboration among individuals or teams working on ontology projects. Multiple users can work on the same ontology concurrently, making complex ontologies easier to build and administer. It also enables community version control and ontology sharing.
- **Plug-in Architecture**: One of Protégé's assets is its extensibility via a large ecosystem of plugins. Users can expand its functionality by installing plugins that add features like as automated reasoning, visualization, and data integration.

- **Semantic Reasoning**: Protégé offers automated reasoning, allowing users to extract new information from existing ontologies. This is essential for spotting errors, discovering implicit relationships, and preserving the integrity of the ontology.
- **Application Areas**: Protégé is utilised in many sectors, including biology, health, finance, and engineering. It is used in biology, for example, to develop ontologies for characterising genes, proteins, and biological processes, which aids in data integration and biomedical research.
- **Education and Training**: Protégé is extensively used in academics to teach knowledge representation and ontology construction. It is accessible to both students and scholars due to its user-friendly design and open-source nature.
- **Community Support**: Protégé has an active user community, and tools such as documentation, tutorials, and forums are accessible to help users and share information.

Protégé is a powerful and widely used tool for creating ontologies, enriching data with semantics, and enabling the Semantic Web. Its user-friendly design, extensibility, and comprehensive features make it a great resource for knowledge engineers, academics, and developers working across multiple areas. However, it does share some negatives points that need to be mentioned:

- **Learning Curve**: For newcomers, especially those unfamiliar with ontology modeling and semantic web technologies, Protégé might have a high learning curve. It may take some time and effort to understand the numerous tabs, features, and ontology languages.
- **Resource Intensive**: Protégé can be resource-intensive when working with big ontologies, requiring a significant amount of memory and computing power. On less powerful systems, this can cause performance concerns.
- **Limited Collaboration Features**: While Protégé has collaboration functions, they are not as sophisticated as some other collaborative programs. When numerous people are editing the same ontology at the same time, collaborative work might be difficult. On top of that, Protégé does not provide built-in collaboration features for real-time collaboration, version control, and conflict resolution, which can be critical for large-scale collaborative ontology projects.
- **Complexity for Complex Ontologies**: Creating complex ontologies with deep class hierarchies and multiple axioms can be time-consuming and difficult to manage with the Protégé interface.
- **Plugin Compatibility**: Some plugins may not be compatible with all Protégé versions, resulting in potential conflicts or limitations in extending the tool's capability.
- **User Interface**: Although Protégé's user interface has evolved over time, some users may find it less visually appealing or intuitive when compared to other ontology creation tools.
- **Performance with Large Datasets**: While Protégé is appropriate for ontology modelling, it may not be the ideal choice for managing very large datasets or applications that require fast data processing.

- **Limited Built-In Reasoning**: Protégé has rudimentary reasoning capabilities, however users may need to incorporate external reasoners for higher reasoning tasks. Reasoners can be difficult to set up and configure.
- **Limited Visualisation**: While Protégé provides visualisation capabilities, they may not be as robust as specialised visualisation tools, making it difficult to develop interactive and aesthetically appealing ontology representations.
- **Updates and Maintenance**: As technology progresses, users may meet challenges with software updates and compatibility, as well as long-term maintenance of ontology projects.

Despite these drawbacks, Protégé remains an important tool for ontology construction and knowledge representation, particularly for researchers, domain specialists, and organisations seeking to create organised and semantically rich ontologies. In this project in specific, most of these concerns don't apply. The learning curve is reduced due to previous experience, the machine on which the ontology is being built has a good computing power and because it is developed by one developer, all the versioning and collaboration tools end up being irrelevant. However, the other concerns still need to be address in the development process.

2.6 Triple Store Databases

With Protégé, the process of modelling and building an ontology is made. However, that by it self is not enough. The various relationships, classes and properties need to be queried and consulted in a fast and efficient way, something that Protégé does not provide. That being said, to facilitate this process, a RDF triple store database can be utilized.

An RDF triplestore, also known as a semantic graph database, is a customized database system that uses the Resource Description Framework (RDF) architecture to store and manage semantic information (Ontotext 2023a). Data is stored in RDF triplestores as a network of interconnected objects with clear relationships or linkages between them. Because of this, they are well-suited to handle heavily interconnected data, making them a preferred choice in scenarios with complicated data relationships.

RDF triplestores provide greater flexibility and cost-effectiveness for maintaining structured data than standard relational databases. RDF databases can handle sophisticated semantic queries effectively, allowing users to obtain data depending on its semantics. Furthermore, they can use inference methods to generate new insights and information from existing data linkages, making them valuable tools for improved knowledge representation and discovery in the context of semantic web applications.

For these type of databases, SPARQL is the standard query language and protocol. SPARQL (SPARQL Protocol and RDF Query Language) is a query language and protocol that is used in Linked Open Data and RDF (Resource Description Framework) databases. This adaptable language, pronounced'sparkle,' is intended to query a wide range of data rapidly, allowing users to extract information concealed within non-uniform data stored in diverse forms and sources (Ontotext 2023b).

It is the industry standard for searching and modifying RDF data, making it appropriate for Linked Open Data on the web and RDF triplestores. The World Wide Web Consortium (W3C) has endorsed and designed it, emphasising its dependability and widespread acceptance.

SPARQL allows users to query information from databases or any RDF-mapped data source. It is more concerned with what users want to know than with how the underlying database is arranged. This technique simplifies searching and enables developers and users to rapidly access and retrieve specific knowledge.

SPARQL is similar to SQL (Structured Query Language), but it is designed specifically for NoSQL graph databases like Ontotext's GraphDB. SPARQL searches can also be conducted on databases that can be seen as RDF via middleware, increasing their utility. SPARQL is a versatile language that may be used for computations, filtering, aggregating, subqueries, and more.

SPARQL is distinguished by its ability to run federated queries, which allow access to numerous data stores (endpoints) rather than being limited to a single database. This is possible because SPARQL serves as both a query language and an HTTP-based transport protocol accessible via a defined transport layer. RDF results can be returned in a variety of data forms, and RDF entities are recognized using Universal Resource Identifiers (URIs). The use of URIs to reference data facilitates clear data referencing across apps, overcoming the limits of local searches. These design principles are consistent with SPARQL's main purpose of enabling Linked Data for the Semantic Web, enriching data by establishing links with global semantic resources, and facilitating meaningful data sharing, merging, and reuse.

With all that information, it is necessary to investigate the best SPARQL and RDF databases to use in this project.

2.6.1 Apache Fuseki

Apache Jena Fuseki is a powerful SPARQL server with a variety of deployment options. It can run as an operating system service, a Java web application (WAR file), or as a standalone server. Fuseki comes in two flavors: a user-friendly "webapp" with administration and query interfaces, and a "main" server for bigger deployments, including Docker and embedded systems. Both setups use the same fundamental protocol engine and file format. Fuseki supports the SPARQL 1.1 and SPARQL Graph Store protocols for requesting and modifying RDF data. It smoothly interacts with TDB for robust, transactional storage and has Jena text query capabilities (Jena 2023a). Some of it's qualities and best features are the following:

- **SPARQL Server**: As a specialized SPARQL server, Apache Jena Fuseki is well-suited for searching and managing RDF data using the SPARQL query language.
- **Flexible Deployment Options**: Fuseki provides a variety of deployment choices, allowing customers to select the configuration that best meets their needs. It can run as an operating system service, a Java web application (WAR file), or as a standalone server, giving it versatility for a variety of use scenarios.
- **Web Application with User Interface**: The "webapp" configuration of Fuseki provides a user-friendly web app with a graphical user interface (UI). This UI simplifies administration duties and allows users to query and manage RDF data interactively.
- **Scalable Standalone Server**: Fuseki's "main" configuration is intended for scalability and incorporation into bigger deployments. It may be integrated into sophisticated systems such as Docker-based installations and embedded deployments.

- **Standard SPARQL Support**: Fuseki offers full support for SPARQL 1.1 protocols, including querying and modifying RDF data. This standardization provides compatibility and interoperability with other SPARQL-compliant tools and libraries.
- **SPARQL Graph Store Protocol**: Fuseki supports the SPARQL Graph Store protocol, which allows for efficient administration of RDF graphs on the server.
- **Integration with TDB**: TDB (Triple Database), which provides a strong and transactional persistent storage layer for RDF data, is intimately linked with Fuseki. This connection improves data consistency and reliability.
- **Text Query Capabilities**: Fuseki includes Jena text query functionality, allowing users to do efficient text-based searches within RDF data.
- **Community Support**: Fuseki benefits from an active and supportive community as part of the Apache Jena project. Documentation, tutorials, and forums are available to users for assistance and collaboration.
- **Open Source and Apache License**: Apache Jena Fuseki is open-source software distributed under the Apache License, which allows it to be used for both business and non-commercial purposes.
- **Scalability**: Fuseki is intended to manage large-scale RDF datasets and can be utilized in scenarios with enormous data volumes.
- **Configurability**: Fuseki's configuration options enable customers to tailor the server to their individual needs, such as security settings, dataset setups, and query optimisation.

While Apache Jena Fuseki is a robust SPARQL server with numerous benefits, there are also potential downsides or restrictions to consider:

- **Learning Curve**: Working with Fuseki and SPARQL might be challenging, especially for people who are new to Semantic Web technology. Users must become acquainted with RDF data modeling and the SPARQL query language.
- **Resource Intensive**: Large RDF datasets can be resource-intensive, necessitating ample memory and processing capacity. Users should be aware of system requirements and scalability issues.
- **Complex Configurations**: Configuring Fuseki for specific use cases might be difficult, especially when working with sophisticated settings or security settings. Users may require knowledge of server administration and RDF data management.
- **No Built-In Inference**: Fuseki does not have built-in RDF inference capability. Users who want to use ontological reasoning or inferencing may need to integrate extra tools or libraries.
- **Limited Text Search**: While Fuseki supports text queries, it may not provide the same advanced text search options as dedicated full-text search engines.
- **Performance Considerations**: The intricacy of SPARQL queries and the quantity of the dataset can affect query performance. For huge datasets, users should optimize searches and explore indexing solutions.

- **Community Size**: While Apache Jena has an active community, it may be smaller than those of competing RDF and SPARQL solutions, which may limit the availability of third-party extensions and support resources.
- **Maintenance and Updates**: Users must keep their Fuseki deployments up to date with the newest versions and security fixes, which may necessitate continuous maintenance.
- **Security Concerns**: Security configurations and practices, as with any server software, are critical for securing RDF data and server resources. Access controls and configurations must be properly managed by users.
- **Integration Challenges**: Depending on the deployment's specific requirements, integrating Fuseki into current IT environments may necessitate additional development and integration activities.

It is crucial to note that many of these disadvantages are frequent issues when working with RDF data and SPARQL servers in general. Careful planning, resource allocation, and knowledge with Semantic Web technologies might help reduce some of these issues.

2.6.2 Blazegraph

Blazegraph is an extraordinarily fast graph database that supports Blueprints and RDF/SPARQL API. It has been used in commercial environments by Fortune 500 firms such as EMC and Autodesk, among others, and is capable of managing up to 50 billion edges on a single system. Blazegraph is widely used in the biological sciences and plays an important role in enabling Precision Medicine applications. It is also widely used to improve Cyber analytics in both commercial and government applications (Blazegraph 2023). Some of the positives are the following:

- **High Performance**: Blazegraph is designed for fast querying and updating, making it particularly adept at managing large RDF datasets.
- **Scalability**: Blazegraph provides a scalable approach through horizontal sharding. It enables RDF data to be distributed across numerous servers, ensuring seamless operation as data volume grows.
- **ACID Transactions**: Blazegraph supports ACID transactions, which provide data integrity and consistency in complicated operations by providing atomicity, consistency, isolation, and durability.
- **Geospatial and Temporal Support**: The database has extensive support for geographic and temporal data, making it a good solution for applications involving location and time-related data.
- **Open Source**: Blazegraph is open-source, allowing users to utilise and adjust it to their own needs without license restrictions.
- **Community and Support**: A vibrant Blazegraph community provides support, documentation, and resources to make adoption and troubleshooting easier.
- **Query Language Compatibility**: Blazegraph is SPARQL-compatible, which implies that it interacts effortlessly with the widely known SPARQL query language, appealing to users who are familiar with semantic web technologies.

- **Versatile Use Cases**: Blazegraph is adaptable and can be used in a variety of scenarios, including data integration, knowledge graph creation, and semantic web technology research.

These items listed above are the main positives for using Blazegraph. However, there is also some negatives to consider:

- **Complexity**: Users who are unfamiliar with RDF databases or semantic web technologies may find it difficult to set up and configure Blazegraph.
- **Resource Intensive**: Blazegraph's great performance and scalability may necessitate extensive hardware resources, especially for large datasets, raising operational expenses.
- **Learning Curve**: When dealing with Blazegraph, users who are unfamiliar with RDF or SPARQL may confront a steep learning curve.
- **Limited Ecosystem**: While Blazegraph is a sophisticated RDF database, its ecosystem is smaller, with fewer third-party tools and connectors than more frequently used database systems.
- **Maintenance**: Managing and maintaining a distributed Blazegraph cluster can be difficult, particularly for organizations with little experience with RDF databases.
- **Lack of Built-in Data Ingestion**: Blazegraph lacks robust built-in ETL (Extract, Transform, Load) capabilities, which may necessitate the use of external data import and transformation tools.
- **Costs**: Although Blazegraph is free and open source, organisations may spend expenditures for hardware, infrastructure, and support in order to implement and maintain the database.

With all things considered, Blazegraph proves to be a powerful tool that, with the right study and development process, might be useful for this project.

2.6.3 Virtuoso

OpenLink Virtuoso is a revolutionary platform that combines open standards for data access, integration, and management with the game-changing powers of AI and AGI. It dismantles data silos by utilising Hyperlinks as robust data source identifiers serving as Super Keys, opening the way for unparalleled data interaction flexibility for both individuals and enterprises (Software 2019). Some of its characteristics and positives are the following:

- **Multi-Model Support**: Virtuoso supports several data models, allowing users to deal with RDF, relational, XML, and JSON data all in the same database.
- **SPARQL and SQL Querying**: It supports both SPARQL and SQL query languages, making it adaptable for querying structured and RDF data and accommodating users who are comfortable with various query languages.
- **Data Integration**: Virtuoso allows for the smooth integration of disparate data sources, assisting organisations in developing unified data solutions.
- **Linked Data Publishing**: It comes with a built-in Web server for publishing RDF data as Linked Data, making it simple to exchange structured data on the web.

- **Geospatial Capabilities**: Virtuoso supports GeoSPARQL, making it suited for managing and analysing geospatial data.
- **Scalability**: The database is built for scalability and can manage huge datasets as well as heavy query volumes.
- **Community and Support**: Virtuoso has an active community and provides users with assistance, documentation, and resources to help them maximise its possibilities.
- **Use Cases**: Virtuoso is used in a variety of applications, including government data integration, data virtualisation, and the construction of RDF-based knowledge graphs.

These items listed above are the main positives for using Virtuoso. However, there is also some negatives to consider:

- **Complexity**: Setting up and customising Virtuoso can be difficult, especially for users who are unfamiliar with RDF databases and multi-model databases.
- **Resource Intensive**: Virtuoso, like many multi-model databases, can be resource-intensive, especially when dealing with huge datasets that may need significant hardware resources.
- **Learning Curve**: When working with Virtuoso, users who are unfamiliar with RDF and multi-model databases may have a high learning curve, especially when dealing with its many data models and query languages.
- **Licensing Costs**: While a free version (Open Source Edition) is accessible, the Enterprise Edition of Virtuoso may involve license costs, which may be a consideration for some organisations.
- **Maintenance**: Managing and maintaining a complicated database system like Virtuoso, especially in a production setting, can be difficult and may necessitate specialized knowledge.
- **Limited Ecosystem**: When compared to more frequently used databases, Virtuoso may have a smaller ecosystem and fewer third-party tools and connectors.
- **Costs**: Aside from licensing, there may be extra expenditures associated with hardware, infrastructure, and support for Virtuoso deployment and maintenance.

With all things considered, Virtuoso poses as a good alternative to implement a RDF-based system and a SPARQL database.

2.6.4 Chosen triple store

After carefully studying and analysing all the different triple-store databases, the one that ended up being superior is Apache Jena Fuseki. It was the one who provided the most familiar and accessible user interface and in which development seemed the most accessible. On top of that, it also presents more tutorials and information online for free, which is very important due to the unfamiliarity of triple store databases when setting out to develop this project.

Chapter 3

Value Analysis

3.1 Innovation Process

The Innovation Process wants to minimise the uncertainty or vagueness of concepts prior to advancing the project. Koen proposed that the process of innovation can be segmented into three distinct stages: The Fuzzy Front End (FFE), the New Product Development (NPD) process, and Commercialization. A visual representation of these stages can be observed in the figure bellow.

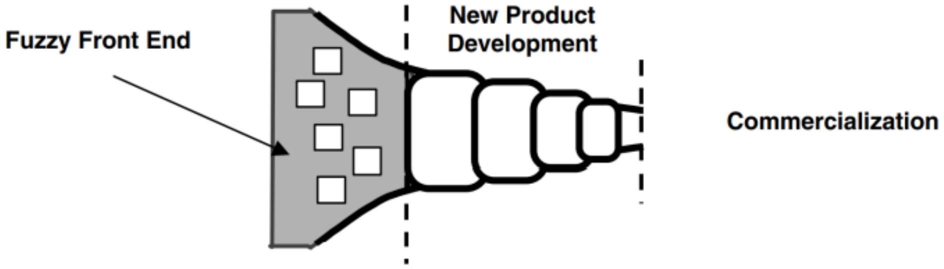


Figure 3.1: Division of the Innovation Process (retrieved from Koen 2001)

- **Fuzzy Front End (FFE)**: Typically, this stage marks the initiation of the innovation process and is characterised by low organisation and unstructured ideas. There are numerous variables related to the commercialisation and financing of the project that are still uncertain at this point. The primary objective of this phase is to explore and enhance the business requirements and concepts.
- **New Product Development (NPD)**: In the second phase, the innovation process has advanced to a favourable position where the plans are well-defined and geared towards specific objectives. During this stage, there is a clear establishment of timelines and objectives, and structured teams are assigned to the continuous development and implementation of the product.
- **Commercialisation**: The final phase represents the end result of the development completed during the preceding two stages. At this point, the product has been completed and it is prepared for commercialisation and realised to its fullest on the market.

3.2 New Concept Development

This technique developed by Peter Koen and his team, has the objective to create knowledge and concepts of the ideas. The development of New Concept Development was intended to address the issue of ambiguity that arises during the first phase of the innovation process (Fuzzy Front End) in a more systematic manner. This innovation model is non-linear and consists of three main components, namely the Motor, the Wheel, and the Rim, as illustrated in the figure below (Koen et al. 2002).

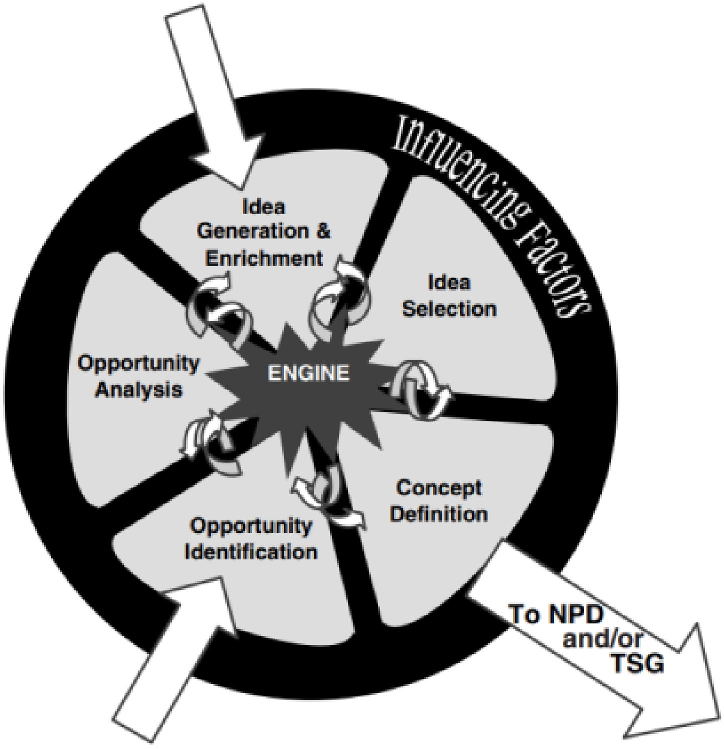


Figure 3.2: New Concept Development (retrieved from Koen 2001)

- **The Motor**: At the heart of the model, the motor is linked to the company's strategic goal and culture, and it will impact and power the inner wheel directly.
- **The Wheel**: The five front end influencing components that make up the inner wheel are opportunity identification, opportunity analysis, idea generation, idea selection, and concept definition.
- **The Rim**: Being the outermost component of the model, it stands in for the external, uncontrollable forces that have an impact on and influence the entire innovation process. Law, politics, and the economics are a few examples of these.

3.2.1 Opportunity Identification

Typically, the first influencing factor to be realised is the opportunity identification, as it provides the company with a clear understanding of which opportunities should be pursued.

During this stage, there is a significant emphasis on presenting future scenarios and engaging in discussions about the future requirements of the organisation.

In this case, the issue was previously identified by the team that was developing the GrouPlanner recommender system, where they realised that the recommendations where not tailored to the tourists personality traits.

3.2.2 Opportunity Analysis

As the second step of the innovation process, opportunity analysis involves a thorough evaluation and analysis of the previously identified opportunity to determine the potential value that can be obtained from investing in it. Various detailed methods can be used to identify opportunities.

Once the organisation has identified an opportunity, it is crucial to assess its potential value to potential customers. In this case, a SWOT diagram was used to facilitate the evaluation process. A SWOT diagram is a tool commonly used by companies to organise internal and external factors that are either positive or negative. It is typically divided into four sections.

- **Strengths**: These are the positive internal factors that contribute value to the proposition.
- **Weaknesses**: These are the negative internal characteristics that hinder the opportunity and create disadvantages when compared to others.
- **Opportunities**: These are the external situations that bring value to the opportunity and propel it forward. Any outside force that facilitates the development or success of the solution falls under this category.
- **Threats**: These are the external negative forces that may stop the project's progress and bring it to a halt.

Strengths	Weaknesses
• Increase customer satisfaction and engagement due to personalised recommendations. • Tailored recommendations that fit tourists unique preferences and their personality. • The use of personality traits differentiates this system from other tourism recommendation systems.	• Tourists hesitation to provide personal information or discomfort to share their personality traits. • Not being able to cater to the preferences of all tourists, especially those with unique or obscure interests.
Opportunities	**Threats**
• Integration with social media platforms or travel blogs to reach a wider audience. • Partnerships with tourism businesses to offer discounts or promotions based on recommendations. • Increase of the amount of tourists in the world.	• Highly Competitive Market. • Privacy concerns.

Table 3.1: SWOT Analysis

3.2.3 Idea Generation and Enrichment

The process of generating ideas it's where these are created, developed and implemented. However, first, they need to be enriched with the knowledge and solutions that were developed in the previous steps.

In this project, there was a need to find a relationship between the tourists personality traits and their favourite leisure and tourism activities, as well as travel destinations. To achieve that, a study was made that connected the five personality dimensions to different tourism activities (that study is described in section 2.2). With the way the study was conducted, the plan was to always use an ontology to develop the recommendations that will be given to tourists.

3.2.4 Idea Selection

After the process of creating and developing ideas, it becomes necessary to select the most effective idea to solve the problem. This step is crucial for successfully implementing a solution. However, in this project, the objective has always been the development of an ontology that would give recommendations based on the tourists personality traits, so there was no need to analyse, evaluate and pick the most valuable idea. s

3.3 Value Proposition

The concept of a business model represents a theoretical construct that encompasses a variety of elements and their interactions, providing a framework to articulate the underlying business logic of a company. A conceptual approach towards business models allows for their capture, modelling, explanation, communication, tracking across time, and, potentially, measurement and simulation. Several authors consider business models as a novel analytical unit and a valuable tool for innovation. The four fundamental pillars of a company's "what," "who," "how," and "how much," as postulated by Osterwalder and Pigneur, are key components that can be broken down into simpler parts. These pillars enable a business to express its offerings, target audience, implementation strategy, and revenue generation potential (Osterwalder and Pigneur 2003).

In this project, the value proposition resides in the implementation of an ontology in an existing tourism recommender system (GrouPlanner) to enable travelling recommendations based on tourists' unique personality traits, thus providing a more personalised experience, more tailored to the users' preferences and that matches their travelling style. The users who adopt this system will discover new cultures, cuisines, and activities that align with their personal preferences, having a more fulfilling and enjoyable travelling experience.

3.4 Customer Value

The Value Proposition Canvas was developed by Dr. Alexander Osterwalder as a tool to assess the compatibility between a product and its market. It closely examines the relationship between two key elements of the Business Model Canvas: client segmentation and value propositions. The canvas can be utilised to enhance an existing service or product or to construct a new one from scratch.

3.4.1 Customer Profile

Customer profiling consists in tree major aspects: gains, pains and customer jobs.

- **Gains**: Refers to the advantages and perks that customers anticipate and demand, as well as the factors that would delight customers and increase the likelihood of accepting a value proposition. As a result, the client, a tourist, will be able to get tourism recommendations that are catered to his preferences and personality, increasing his satisfaction and engagement.
- **Pains**: Represents the adverse experiences, emotions, and potential hazards that consumers encounter while accomplishing a task. In this dissertation, they are represented by a potential lack of interest or fear of the tourists to give their personality traits and preferences in the recommender system. Without this feedback, it is impossible to give accurate and precise recommendations.
- **Customer Jobs**: Represents the activities that consumers aim to accomplish, along with their associated challenges and needs, encompassing functional, social, and emotional dimensions. As a result, the tourists require a method that recommends them the best tourism destinations and activities based on their personality and preferences.

3.4.2 Value Map

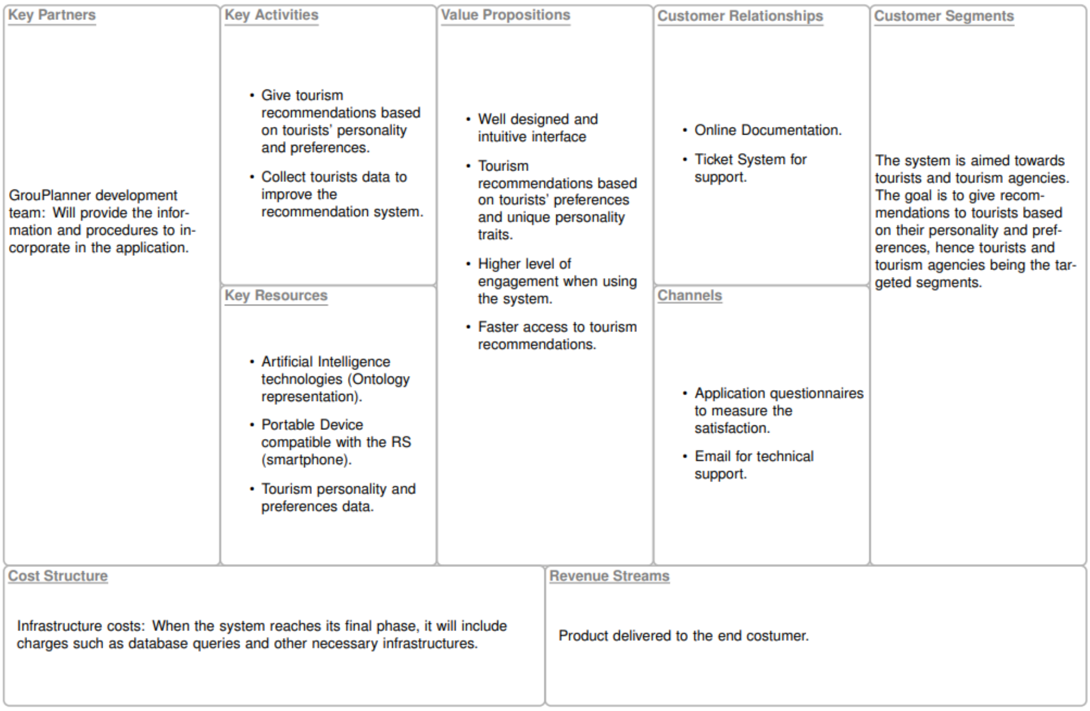
The value provided to the customer can be categorised into three sectors: Gain Creators, Pain Relievers and Products and Services.

- **Gain Creators**: The way in which a product or service enhances the customer benefits and generates gains for them. In this dissertation, the system developed aims to give a solution to the client that gives tourism recommendations that are catered to tourists preferences and personality, increasing their satisfaction and engagement.

- **Pain Relievers**: How a product or service addresses customers difficulties and how it aids them with their pains. In this case, it will provide a well-designed and user-friendly interface to make the process of filling in the preferences and personality traits of the tourists easier. It will also store previously suggested locations and activities to facilitate the search process and to reduce downtime.
- **Products and Services**: Includes products and services that generate profit and alleviate suffering, as well as those that support the production of value for customers. In this case, the system will provide the customer with an ontology that gives tourism recommendations based on users preferences and personality, which will be incorporated in the aforementioned solution referred to in Section 3.2, which is a tourism recommender system.

3.5 Canvas Business Model

The Canvas Business Model has the goal of describing how the product developed operates and fits into a business. In the case of this dissertation, this model is represented in the figure bellow.



Key Partners

GrouPlanner development team: Will provide the information and procedures to incorporate in the application.

Key Activities

- Give tourism recommendations based on tourists' personality and preferences.
- Collect tourists data to improve the recommendation system.

Key Resources

- Artificial Intelligence technologies (Ontology representation).
- Portable Device compatible with the RS (smartphone).
- Tourism personality and preferences data.

Value Propositions

- Well designed and intuitive interface
- Tourism recommendations based on tourists' preferences and unique personality traits.
- Higher level of engagement when using the system.
- Faster access to tourism recommendations.

Customer Relationships

- Online Documentation.
- Ticket System for support.

Channels

- Application questionnaires to measure the satisfaction.
- Email for technical support.

Customer Segments

The system is aimed towards tourists and tourism agencies. The goal is to give recommendations to tourists based on their personality and preferences, hence tourists and tourism agencies being the targeted segments.

Cost Structure

Infrastructure costs: When the system reaches its final phase, it will include charges such as database queries and other necessary infrastructures.

Revenue Streams

Product delivered to the end costumer.

Figure 3.3: Canvas Business Model

Chapter 4

Analysis and Design

4.1 Requirements Analysis

The most critical issue and requirement for the project is the ontology's integration with the other existing systems in the GrouPlanner prototype. The solution was intended to work and integrate with the recommendation engine and in the POI service. Both of them have different types of connection properties. However, there is already and existing database, in PostgreSQL, that has the different points of interest. This new one is meant to replace the old one using a faster system (with SPARQL) and considering the user's preferences.

4.1.1 Functional Requirements

After the analysis on the requirements that were gathered above, two use cases were idealised:

- **UC1**: Manage the available Data Sources
- **UC2**: Visualise the recommended point of interest

The Use Case diagram is represented bellow.

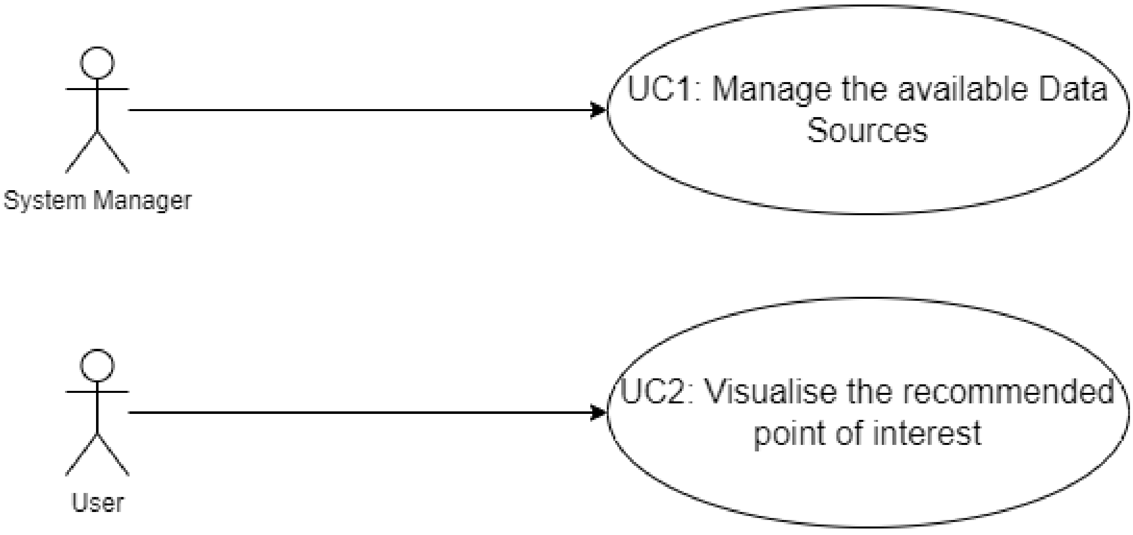

System Manager
UC1: Manage the available Data Sources
User
UC2: Visualise the recommended point of interest

Figure 4.1: Use Case Diagram

4.1.2 Non-Functional Requirements

To assess and gauge the non-functional requirements, the FURPS+ model was employed (1000sourcecodes 2023). This model classifies non-functional requirements into five primary categories: Functionality, Usability, Reliability, Performance, and Supportability. The "plus" signifies additional considerations such as Design Constraints, Implementation requirements, Interface requirements, and Physical requirements. This model considers the already existing GrouPlanner interface, as well as the new database implementation.

Functionality:

- **NFF01**: HTTPS protocol should be used in the communications between components.

Usability:

- **NFU01**: The system's interface must be intuitive and user-friendly for adult users of all technical skill levels.
- **NFU02**: The User Interface for displaying the recommendations should not display technical data that users may not understand.

Reliability:

- **NFR01**: The system should have high availability, which means it should be available to users 24 hours a day, seven days a week, with minimal downtime for maintenance.
- **NFR02**: Implement strong error handling techniques to manage system faults or failures graciously without causing data loss or user inconvenience.

Performance:

- **NFP01**: : The Data Retrieval from the existent systems shouldn't affect their performance.

Supportability:

- **NFS01**: The system should support any SPARQL database, needing the specification of all data needed for the connection (such as IP Address, username, password, and port number).

4.2 Ontology Design

The first steps of designing the ontology were to represent it in a diagram in order to see the different categories of tourism, their hierarchy and how they interact with each other. First, the ontology was represented in the format of a mind map (check appendix A). The main categories represented were based according to the study of Alves, Carneiro, Saraiva, et al. 2023 and there are 11 main categories that are related with various personality dimensions. Here's a summary of each category:

- **Adrenaline Activities**: These are activities that require a high level of excitement, risk, and physical exertion and are generally performed in natural surroundings. Extroverted people who seek adventure and excitement prefer them.
- **Sun, Water, and Sand**: This category comprises activities linked to enjoying sunny and beachside destinations, such as swimming, resting at the beach, and vacationing on islands. People who prefer these activities are typically outgoing and slightly neurotic.

- **Sports and Games**: This category includes a wide range of activities such as sporting events, casino games, hunting, and fishing. Those that like these activities are usually quite outgoing, but they lack agreeableness, openness, and conscientiousness.
- **Cultural Heritage**: Visiting historical landmarks, monuments, and touring old cities or towns are all part of this category. It is favoured by people who enjoy joining friends and family to cultural venues.
- **Museums, Boat Trips, and Viewpoints**: This type of activities focuses on viewing and appreciating natural or historical scenery, which frequently includes boat rides and visits to vistas. These activities are predicted by four personality traits, with positive agreeableness and negative conscientiousness having considerable influences.
- **Natural Phenomena**: This category contains activities involving the observation of natural phenomena such as caves, volcanoes, and the northern lights. Those who favour these activities, like those who enjoy wild nature activities, have positive agreeableness, extroversion, and negative conscientiousness.
- **Health and Well-being**: Attending health and wellness centers, spas, and having wellness treatments are all examples of activities in this category. Individuals that chose these hobbies are frequently highly extroverted, modestly conscientious, and neurotic, with low openness and agreeableness.
- **Gastronomy Events**: This category includes food and wine-related events such as food festivals and wine tasting tours. Gastronomy event attendees are often extroverted and agreeable, with some negative qualities in openness and conscientiousness.
- **Wild Nature Activities**: These activities take place in natural surroundings and include adventurous activities such as mountaineering, gorge visits, and safaris. Extroverted and amiable people with low conscientiousness prefer them.
- **Party, Music, and Nightlife**: This category includes nightlife activities such as clubbing and attending music events. Those who enjoy these activities have negative values for openness to experience, conscientiousness, and agreeableness, with extroversion having a major influence.
- **Theme and Animal Parks**: Theme parks and zoos are examples of activities in this category. Those who favor theme parks and animal parks are very extroverted people who seek excitement and enjoy spending time with friends and family, with some negative tendencies in agreeableness, openness, and conscientiousness.

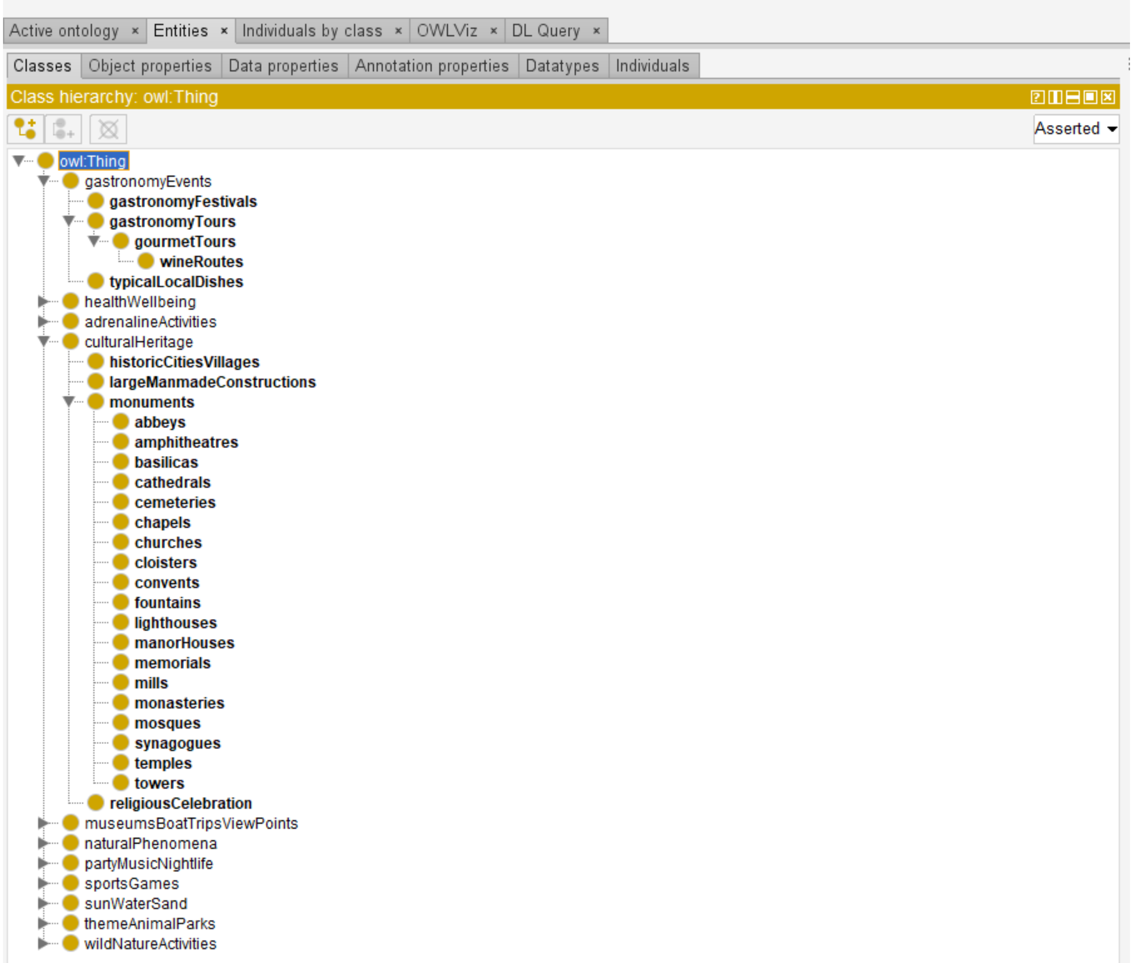
These categories and their associations with personality dimensions help understand the preferences of tourists and how certain personality traits may influence their choices when it comes to tourist attractions.

4.3 Protégé Development

After the representation with mind map (check appendix A), the next step was to use that design and re-create it with the OWL 2 Web Ontology Language. To do this, the Protégé platform was used.

4.3.1 Classes

First, the different classes related to the previously referred tourism categories and their hierarchies were represented in the classes tab that allows users to define and manage classes within the ontology, which represent concepts or categories within a domain. Users can create new classes, subclass existing ones, and specify class hierarchies.

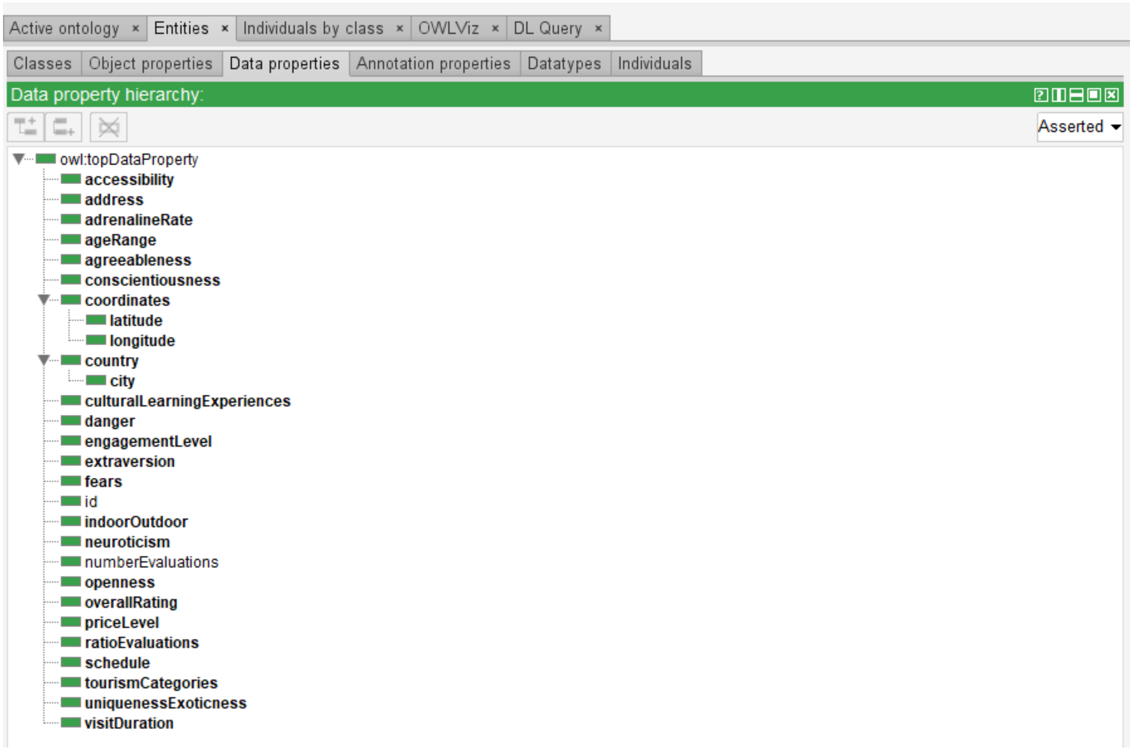


- owl:Thing
  - gastronomyEvents
    - gastronomyFestivals
    - gastronomyTours
      - gourmetTours
        - wineRoutes
    - typicalLocalDishes
  - healthWellbeing
  - adrenalineActivities
  - culturalHeritage
    - historicCitiesVillages
    - largeManmadeConstructions
    - monuments
      - abbeys
      - amphitheatres
      - basilicas
      - cathedrals
      - cemeteries
      - chapels
      - churches
      - cloisters
      - convents
      - fountains
      - lighthouses
      - manorHouses
      - memorials
      - mills
      - monasteries
      - mosques
      - synagogues
      - temples
      - towers
    - religiousCelebration
  - museumsBoatTripsViewPoints
  - naturalPhenomena
  - partyMusicNightlife
  - sportsGames
  - sunWaterSand
  - themeAnimalParks
  - wildNatureActivities

Figure 4.2: Classes Tab in Protégé representing the tourism categories

4.3.2 Data Properties

After defining the classes, the data within the individuals or classes also needed to be defined. They were already defined in the original database of GrouPlanner and now they needed to be transported to Protégé in the "Data Properties" tab. This tab allows users to define relationships between individuals or classes. However, data properties are used to associate data values (e.g., strings, numbers) with individuals or classes.



```
owl:topDataProperty
    accessibility
    address
    adrenalineRate
    ageRange
    agreeableness
    conscientiousness
    coordinates
        latitude
        longitude
    country
        city
    culturalLearningExperiences
    danger
    engagementLevel
    extraversion
    fears
    id
    indoorOutdoor
    neuroticism
    numberEvaluations
    openness
    overallRating
    priceLevel
    ratioEvaluations
    schedule
    tourismCategories
    uniquenessExoticness
    visitDuration
```

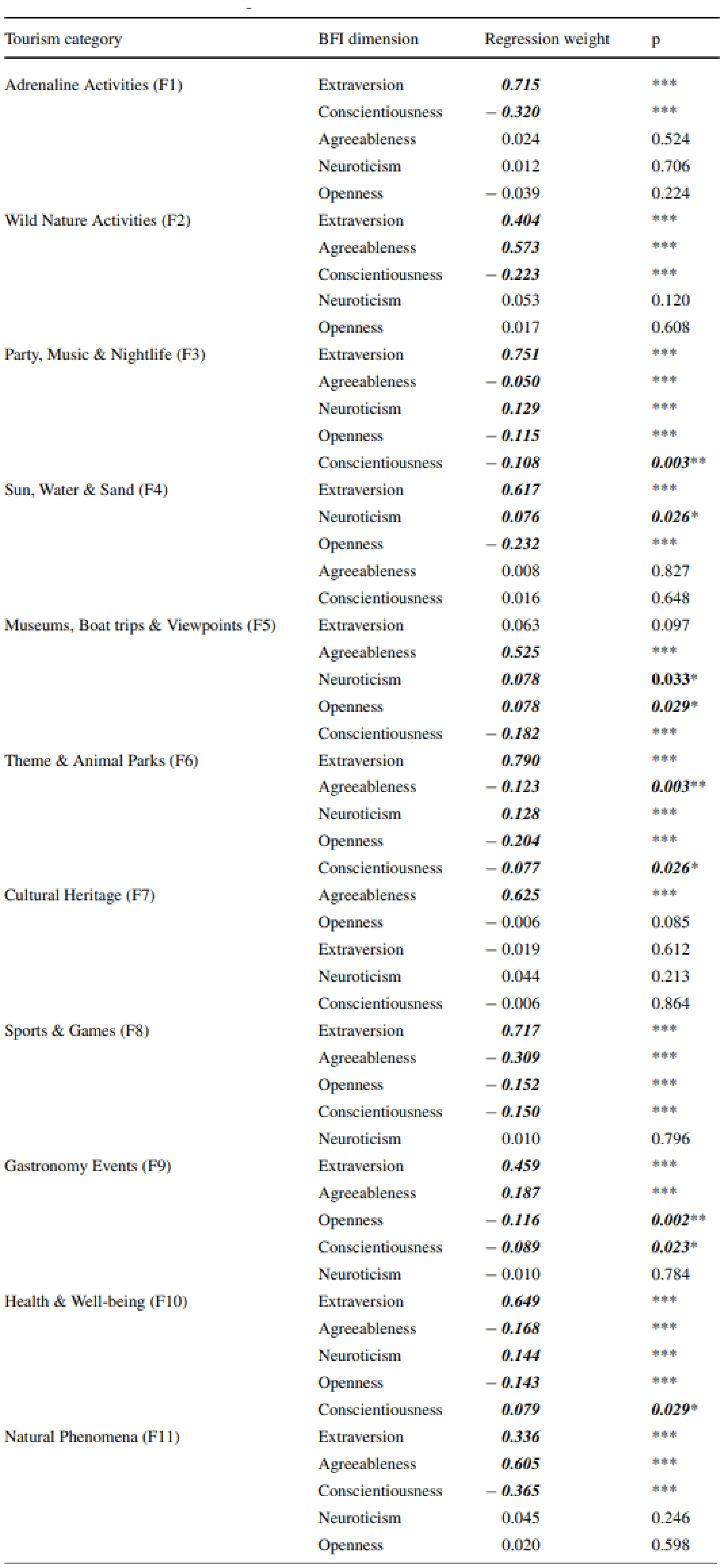
Figure 4.3: Data Properties for the POI in Protégé

These properties were defined based on several parameters. Some of them are essential for classifying a point of interest, them being the following:

- **id**: Represents id of the point of interest in the database. It is represented by an int value.
- **adrenalineRate**: Represents the level of excitement or adrenaline-inducing activities available at the point of interest. It is represented by a double value between the range of 0-1.
- **danger**: Indicates the level of risk associated with visiting the point of interest. It is represented by a double value between the range of 0-1.
- **visitDuration**: Specifies the typical amount of time tourists spend at the point of interest. It is represented by a dataTime value between two ranges depending on the POI.
- **overallRating**: Based on user evaluations and ratings, this value represents the overall rating or satisfaction level of the point of interest, indicating its popularity and quality. It is represented by a double value between the range of 0-5.
- **numberEvaluations**: Based on user evaluations and ratings, this value represents the number of evaluations done to the POI. It is represented by a int value.
- **ratioEvaluations**: This value represents the ratio between overallRating and numberEvaluations. It is represented by a double value.
- **priceLevel**: Indicates the cost or price range associated with visiting the point of interest, helping tourists budget their trips. It is represented by a double value.

- **engagementLevel**: Describes how engaging or interactive the point of interest is for visitors. It is represented by a double value between the range of 0-1.
- **city**: Represents the city where the point of interest is located. It is represented by a string value.
- **country**: Represents the country where the point of interest is located. It is represented by a string value.
- **tourismCategories**: Specifies the type of tourism experience offered by the point of interest. It is represented by a string value.
- **ageRange**: Indicates the suitable age range for visitors, whether it's for children, adults, or all age groups. It is represented by a int value between two ranges depending on the POI.
- **indoorOutdoor**: Specifies whether the attraction is indoors, outdoors, or a combination of both, helping travellers plan for weather conditions. It is represented by 3 int values, 0 being indoors, 1 being outdoors and 2 being both.
- **schedule**: Provides information about the opening and closing hours, as well as any special schedules or events associated with the point of interest. It is represented by a dataTime value between two ranges depending on the POI
- **coordinates**: Includes the geographical coordinates (latitude and longitude) of the attraction's location. It is represented by two double values (latitude and longitude).
- **address**: Provides the physical address or location details of the point of interest. It is represented by a string value.
- **accessibility**: Describes the level of accessibility and accommodations available for individuals with disabilities, ensuring that the point of interest is inclusive and welcoming to all visitors. It is represented by an Enum with different conditions (Wheelchair, Blind, Deaf, Heart Conditions...).
- **fears**: Provides information on common fears or concerns that visitors may have while visiting a point of interest. It is represented by an Enum with different fears (Heights, Water, Wildlife...).

Other properties are more subjective and are related to the personality of the tourists. Despite being classified in the POI, their values are based on studies conducted that associate human characteristics to an activity and POI. These were already defined in chapter 2. These data properties are openness, conscientiousness, extraversion, agreeableness and neuroticism. The following table explains their values and how they relate to the POI.

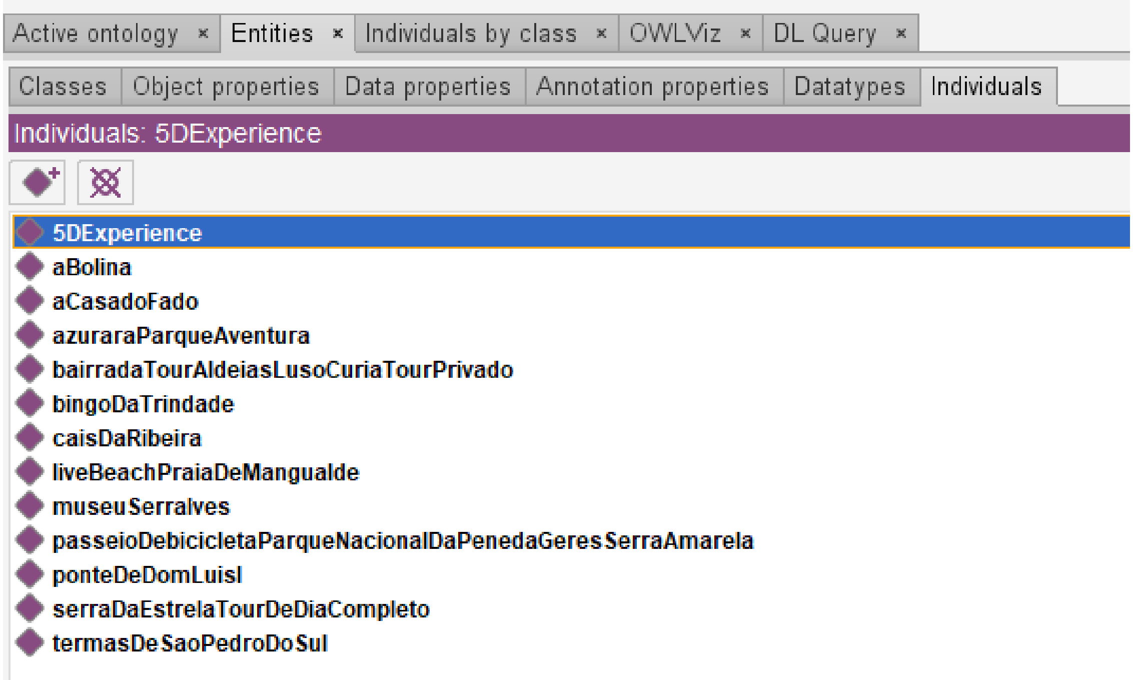


Tourism category	BFI dimension	Regression weight	p
Adrenaline Activities (F1)	Extraversion	0.715	***
	Conscientiousness	− 0.320	***
	Agreeableness	0.024	0.524
	Neuroticism	0.012	0.706
	Openness	− 0.039	0.224
Wild Nature Activities (F2)	Extraversion	0.404	***
	Agreeableness	0.573	***
	Conscientiousness	− 0.223	***
	Neuroticism	0.053	0.120
	Openness	0.017	0.608
Party, Music & Nightlife (F3)	Extraversion	0.751	***
	Agreeableness	− 0.050	***
	Neuroticism	0.129	***
	Openness	− 0.115	***
	Conscientiousness	− 0.108	0.003**
Sun, Water & Sand (F4)	Extraversion	0.617	***
	Neuroticism	0.076	0.026*
	Openness	− 0.232	***
	Agreeableness	0.008	0.827
	Conscientiousness	0.016	0.648
Museums, Boat trips & Viewpoints (F5)	Extraversion	0.063	0.097
	Agreeableness	0.525	***
	Neuroticism	0.078	0.033*
	Openness	0.078	0.029*
	Conscientiousness	− 0.182	***
Theme & Animal Parks (F6)	Extraversion	0.790	***
	Agreeableness	− 0.123	0.003**
	Neuroticism	0.128	***
	Openness	− 0.204	***
	Conscientiousness	− 0.077	0.026*
Cultural Heritage (F7)	Agreeableness	0.625	***
	Openness	− 0.006	0.085
	Extraversion	− 0.019	0.612
	Neuroticism	0.044	0.213
	Conscientiousness	− 0.006	0.864
Sports & Games (F8)	Extraversion	0.717	***
	Agreeableness	− 0.309	***
	Openness	− 0.152	***
	Conscientiousness	− 0.150	***
	Neuroticism	0.010	0.796
Gastronomy Events (F9)	Extraversion	0.459	***
	Agreeableness	0.187	***
	Openness	− 0.116	0.002**
	Conscientiousness	− 0.089	0.023*
	Neuroticism	− 0.010	0.784
Health & Well-being (F10)	Extraversion	0.649	***
	Agreeableness	− 0.168	***
	Neuroticism	0.144	***
	Openness	− 0.143	***
	Conscientiousness	0.079	0.029*
Natural Phenomena (F11)	Extraversion	0.336	***
	Agreeableness	0.605	***
	Conscientiousness	− 0.365	***
	Neuroticism	0.045	0.246
	Openness	0.020	0.598

Table 4.1: Relationship between Tourist Personalities and Categories (retrieved from Alves, Carneiro, Saraiva, et al. 2023)

4.3.3 Individuals

For each class and subclass, there needs to be a real point of interest that falls in those categories. This was done in Protégé, in the individuals tab.

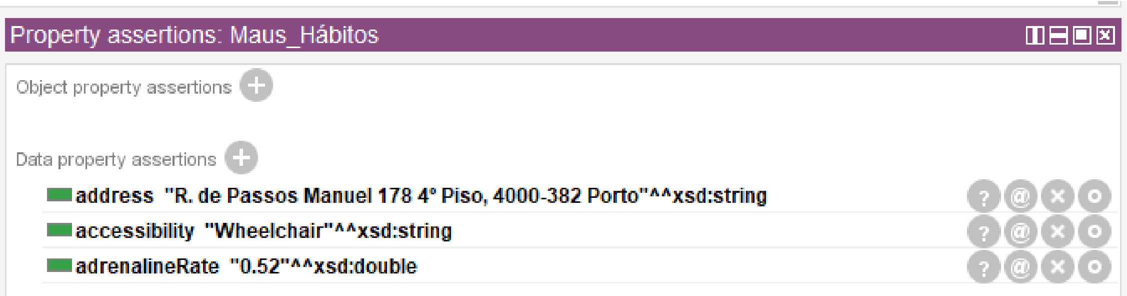


Individuals: 5DExperience

- 5DExperience
- aBolina
- aCasadoFado
- azuraraParqueAventura
- bairradaTourAldeiasLusoCuriaTourPrivado
- bingoDaTrindade
- caisDaRibeira
- liveBeachPraiaDeMangualde
- museuSerralves
- passeioDebicicletaParqueNacionalDaPenedaGeresSerraAmarela
- ponteDeDomLuisI
- serraDaEstrelaTourDeDiaCompleto
- termasDeSaoPedroDoSul

Figure 4.4: Individuals (POI) in Protégé

For each individual, there needs to be a classification of its data properties, so that when the ontology is queried, specific attributes and characteristics that are unique (or not) to the individual can also be queried. A tab in Protégé named Property Assertions was used to do that.



Property assertions: Maus_Hábitos

Object property assertions

Data property assertions

- address "R. de Passos Manuel 178 4º Piso, 4000-382 Porto"^^xsd:string
- accessibility "Wheelchair"^^xsd:string
- adrenalineRate "0.52"^^xsd:double

Figure 4.5: Property Assertion in Protégé

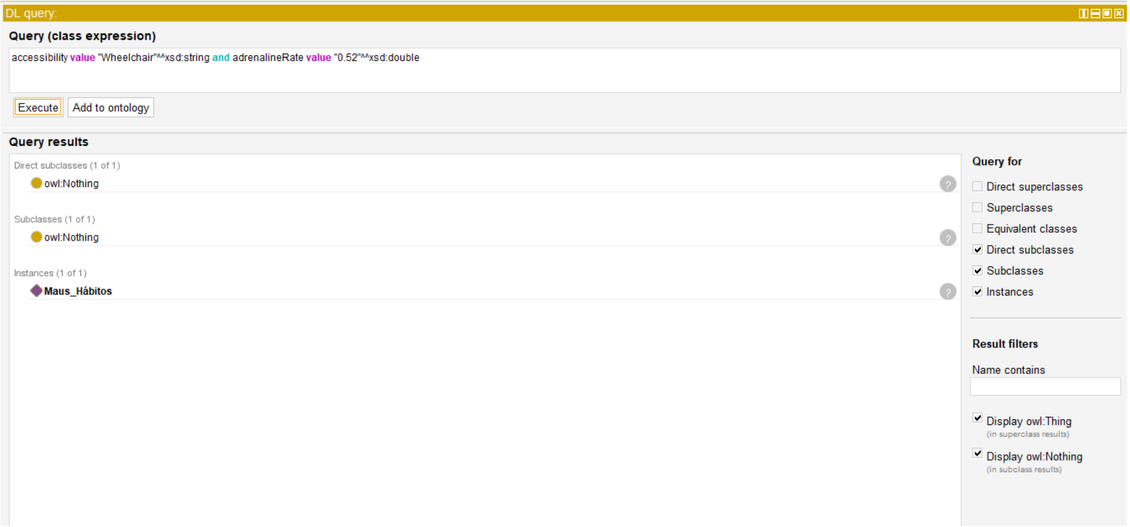
In the example above, the point of interest (represented as individual in Protégé) "Maus Hábitos" has an address defined by a string, it is accessible to people in Wheelchairs due to the data property accessibility and an adrenalineRate of 0.52 (from an interval between 0 and 1), which means just above average.

4.3.4 DL Query

To test the query and to check if it was functioning correctly, some searches were made to the classified ontology. To do that, a section called DL Query was used.

With each query, what to present in the results tab can be chosen (Direct superclasses, superclasses, equivalent classes, direct subclasses, subclasses and instances).

In the image below, there was a query was made with the intent of searching specific characteristics of an individual, not specifying its class but just its data properties.



Query (class expression)

```
accessibility value "Wheelchair"^^xsd:string and adrenalineRate value "0.52"^^xsd:double
```

Query results

Direct subclasses (1 of 1)
owl:Nothing

Subclasses (1 of 1)
owl:Nothing

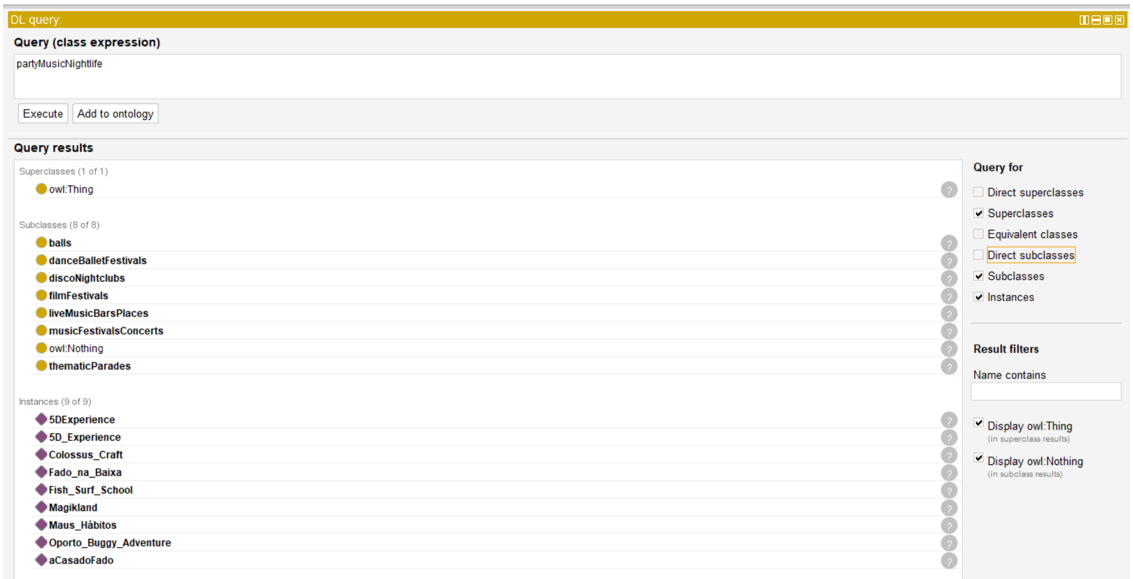
Instances (1 of 1)
Maus_Hábitos

Query for
□ Direct superclasses
□ Superclasses
□ Equivalent classes
☑ Direct subclasses
☑ Subclasses
☑ Instances

Result filters
Name contains
☑ Display owl:Thing (in superclass results)
☑ Display owl:Nothing (in subclass results)

Figure 4.6: Search for Data Properties

Another query that can be made is just to find the individuals related to a specific class, its subclasses and its superclasses. To do that, the following query was made as shown in the image bellow.



Query (class expression)

```
partyMusicNightlife
```

Query results

Superclasses (1 of 1)
owl:Thing

Subclasses (8 of 8)
balls
danceBalletFestivals
discoNightclubs
filmFestivals
liveMusicBarsPlaces
musicFestivalsConcerts
owl:Nothing
thematicParades

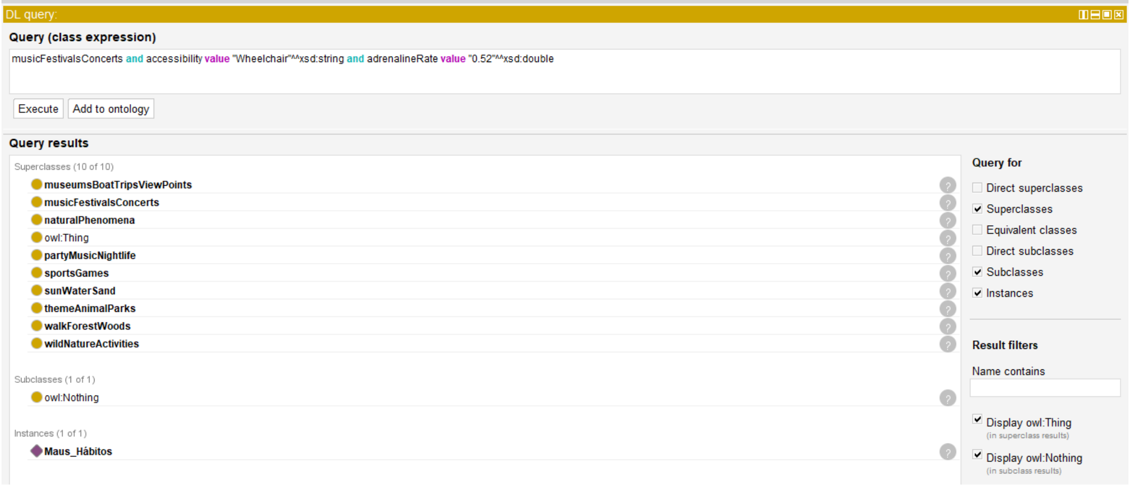
Instances (9 of 9)
5DExperience
5D_Experience
Colossus_Craft
Fado_na_Baixa
Fish_Surf_School
Magikland
Maus_Hábitos
Oporto_Buggy_Adventure
aCasadoFado

Query for
□ Direct superclasses
☑ Superclasses
□ Equivalent classes
□ Direct subclasses
☑ Subclasses
☑ Instances

Result filters
Name contains
☑ Display owl:Thing (in superclass results)
☑ Display owl:Nothing (in subclass results)

Figure 4.7: Search for Individuals, Superclasses and Subclasses

Finally, the two can be combine and get an individual that corresponds to a defined class and with the following attributes as shown in the image bellow.



DL query:

Query (class expression)

```
musicFestivalsConcerts and accessibility value "Wheelchair"^^xsd:string and adrenalineRate value "0.52"^^xsd:double
```

Query results

Superclasses (10 of 10)

- museumsBoatTripsViewPoints
- musicFestivalsConcerts
- naturalPhenomena
- owl:Thing
- partyMusicNightlife
- sportsGames
- sunWaterSand
- themeAnimalParks
- walkForestWoods
- wildNatureActivities

Subclasses (1 of 1)

- owl:Nothing

Instances (1 of 1)

- Maus_Hábitos

Query for

- □ Direct superclasses
- ☑ Superclasses
- □ Equivalent classes
- □ Direct subclasses
- ☑ Subclasses
- ☑ Instances

Result filters

Name contains

☑ Display owl:Thing (in superclass results)

☑ Display owl:Nothing (in subclass results)

Figure 4.8: Search for Individuals with both queries

4.4 GrouPlanner App

As said in previous chapters, GrouPlanner is an already existing app constituted by microservices that communicate between themselves.

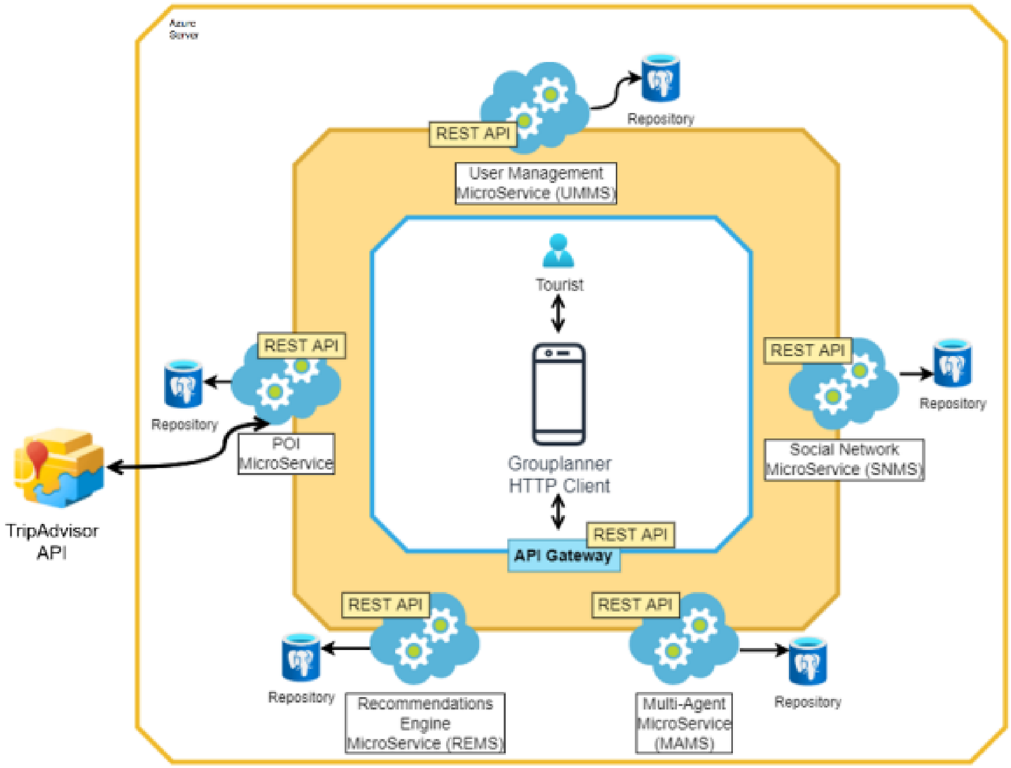


Figure 4.9: GrouPlanner Architecture (Alves, Saraiva, et al. 2022)

The Recommendations Engine Microservice is the one responsible for making the current recomendations to the user. These recommendations are based on users preferences predicted by their personality, by using the referred model in Alves, Carneiro, Saraiva, et al. 2023. At the moment, the list of suggested POI are retrieved from a PostgreSQL database which do not uses the full potential an ontology can provide. One solution was to change the recommendation engine in order to accommodate the triple store database. To do this, the current database repository of the microservice needs to be replaced by a triple store database, as well as the services that query it such as the recommendation engine. Another solution could be just to implement another microservice that queries the triple store database. Because all the microservices are independent, a new one can be created and then the mobile application would query it. Due to time restrains, the approach that was chosen was the first one, due to being easier to modify an existing microservice than to create a whole new one.

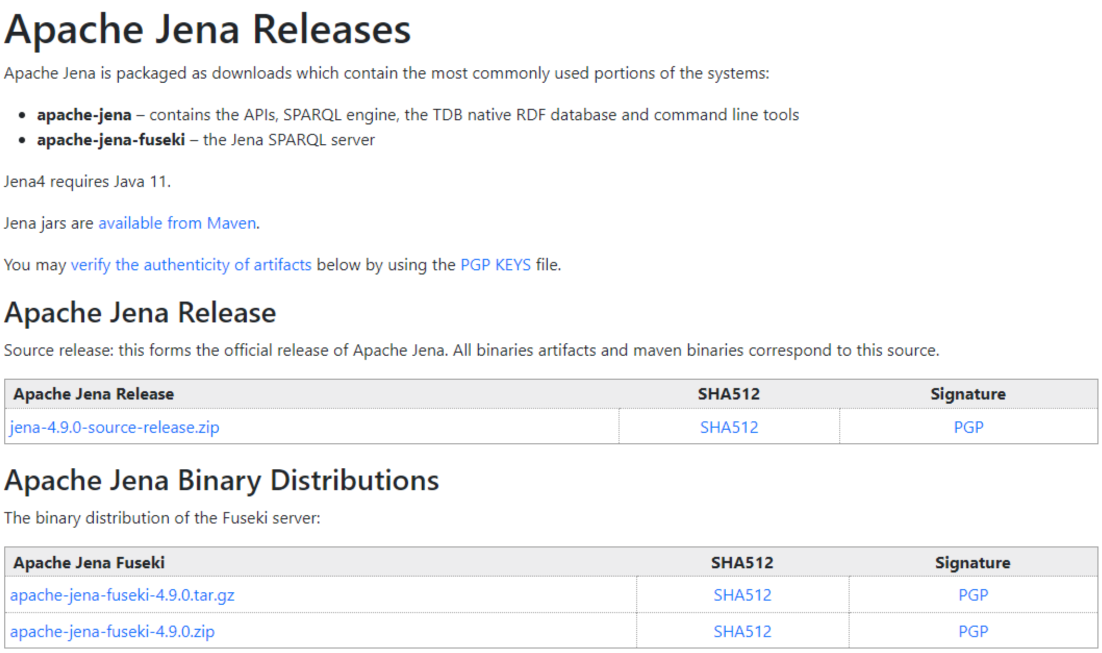
Chapter 5

Implementation

5.1 Apache Jenna

5.1.1 Download

To implement the ontology in a SPARQL database, in this case, in Apache Jenna Fuseki, the first step was to download the correct distribution from the official Apache Jena website (Jena 2023b). There are several versions that are available to download. For this scenario, the Apache Jena distribution downloaded was the binary distribution of the Fuseki server in the .zip format. One thing to consider is that to run Apache Jena it was necessary to have Java installed with the lowest version being Java 11.



Apache Jena Releases

Apache Jena is packaged as downloads which contain the most commonly used portions of the systems:

- **apache-jena** – contains the APIs, SPARQL engine, the TDB native RDF database and command line tools
- **apache-jena-fuseki** – the Jena SPARQL server

Jena4 requires Java 11.

Jena jars are available from Maven.

You may verify the authenticity of artifacts below by using the PGP KEYS file.

Apache Jena Release

Source release: this forms the official release of Apache Jena. All binaries artifacts and maven binaries correspond to this source.

Apache Jena Release	SHA512	Signature
jena-4.9.0-source-release.zip	SHA512	PGP

Apache Jena Binary Distributions

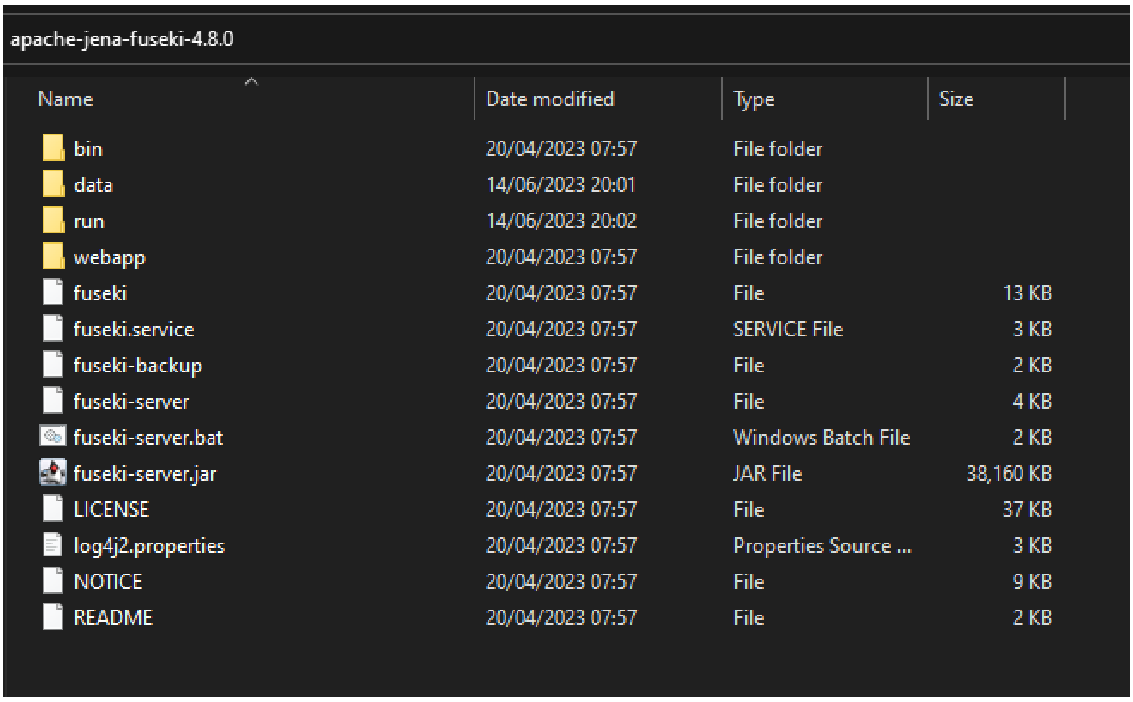
The binary distribution of the Fuseki server:

Apache Jena Fuseki	SHA512	Signature
apache-jena-fuseki-4.9.0.tar.gz	SHA512	PGP
apache-jena-fuseki-4.9.0.zip	SHA512	PGP

Figure 5.1: Download options Apache Jenna (retrieved from Jena 2023b)

5.1.2 Running the Server

After the download process and extracting the .zip file, the following folder was presented.

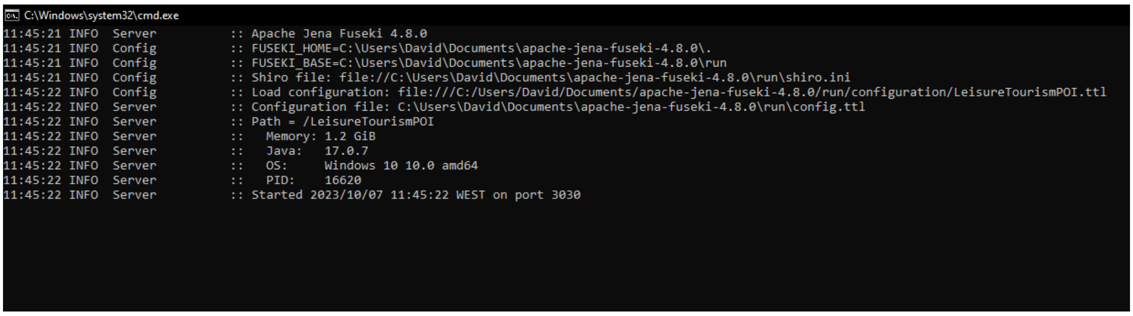


apache-jena-fuseki-4.8.0

Name	Date modified	Type	Size
bin	20/04/2023 07:57	File folder	
data	14/06/2023 20:01	File folder	
run	14/06/2023 20:02	File folder	
webapp	20/04/2023 07:57	File folder	
fuseki	20/04/2023 07:57	File	13 KB
fuseki.service	20/04/2023 07:57	SERVICE File	3 KB
fuseki-backup	20/04/2023 07:57	File	2 KB
fuseki-server	20/04/2023 07:57	File	4 KB
fuseki-server.bat	20/04/2023 07:57	Windows Batch File	2 KB
fuseki-server.jar	20/04/2023 07:57	JAR File	38,160 KB
LICENSE	20/04/2023 07:57	File	37 KB
log4j2.properties	20/04/2023 07:57	Properties Source ...	3 KB
NOTICE	20/04/2023 07:57	File	9 KB
README	20/04/2023 07:57	File	2 KB

Figure 5.2: Apache Jenna Fuseki Folder

There, all the configurations needed to run the Apache Jenna Fuseki are present. To run the server, the fuseki-server.bat windows file needs to be executed. That command will open a command prompt window that runs the server in *http://localhost:3030/*.



C:\Windows\system32\cmd.exe

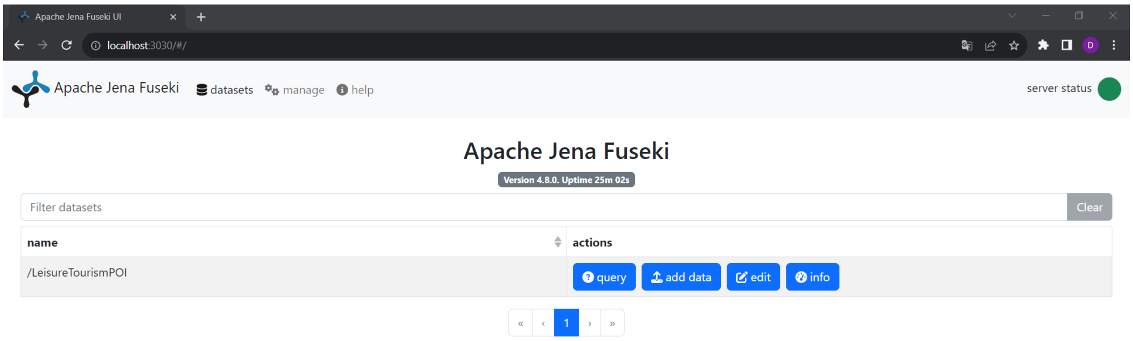
```
11:45:21 INFO  Server          :: Apache Jena Fuseki 4.8.0
11:45:21 INFO  Config          :: FUSEKI_HOME=C:\Users\David\Documents\apache-jena-fuseki-4.8.0\.
11:45:21 INFO  Config          :: FUSEKI_BASE=C:\Users\David\Documents\apache-jena-fuseki-4.8.0\run
11:45:21 INFO  Config          :: Shiro file: file://C:\Users\David\Documents\apache-jena-fuseki-4.8.0\run\shiro.ini
11:45:22 INFO  Config          :: Load configuration: file:///C:/Users/David/Documents/apache-jena-fuseki-4.8.0/run/configuration/LeisureTourismPOI.ttl
11:45:22 INFO  Server          :: Configuration file: C:\Users\David\Documents\apache-jena-fuseki-4.8.0\run\config.ttl
11:45:22 INFO  Server          :: Path = /LeisureTourismPOI
11:45:22 INFO  Server          ::   Memory: 1.2 GiB
11:45:22 INFO  Server          ::   Java:   17.0.7
11:45:22 INFO  Server          ::   OS:     Windows 10 10.0 amd64
11:45:22 INFO  Server          ::   PID:    16620
11:45:22 INFO  Server          :: Started 2023/10/07 11:45:22 WEST on port 3030
```

Figure 5.3: Apache Jenna Cmd window

In this window, all the queries, requests and activities are registered.

5.1.3 User Interface

Apache Jena Fuseki's default screen is the one presented bellow (also corresponds to the datasets tab).



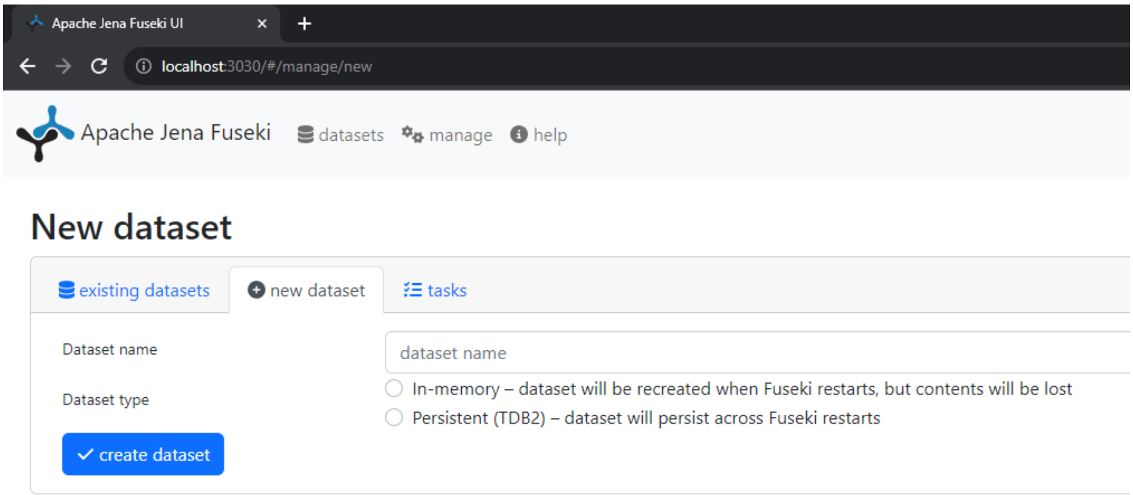
name	actions
/LeisureTourismPOI	query add data edit info

Figure 5.4: Apache Jenna Datasets

In here is where the datasets are presented. In the image above there already is one defined called "/LeisureTourismPOI".

5.1.4 Create and manage a dataset

To create a dataset, there is an option called "manage" in the tab above. Clicking on that, the user can define a new dataset or manage existing ones. In the create option, users can choose the name of the dataset and if they want to persist the data in-memory or persistent. With these factors chosen, the dataset was created.

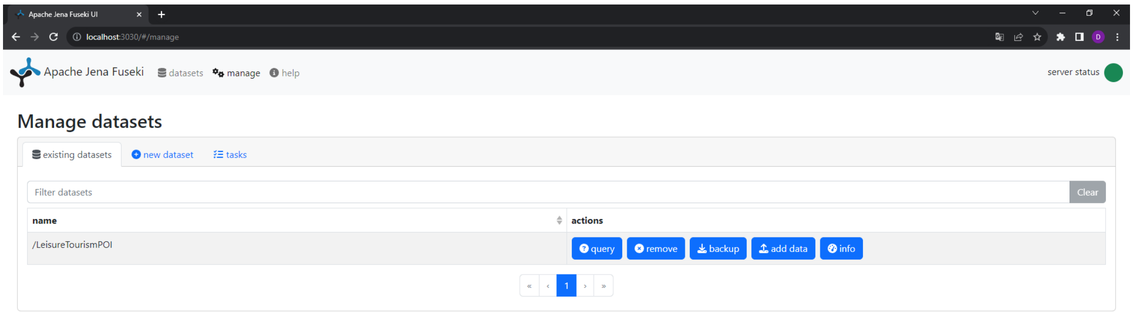


Dataset name: dataset name

Dataset type:
○ In-memory – dataset will be recreated when Fuseki restarts, but contents will be lost
○ Persistent (TDB2) – dataset will persist across Fuseki restarts

Figure 5.5: Create Dataset in the Apache Jena Fuseki UI

With the dataset created, the heavy lifting of creating classes, sub-classes, individuals, data properties, etc, was already done by Protégé. That being said, Apache Jena can upload a .owl file and use that information to fill in the dataset. To do that, there is an option in the existing datasets tab that allows to add data (see figure 5.6).

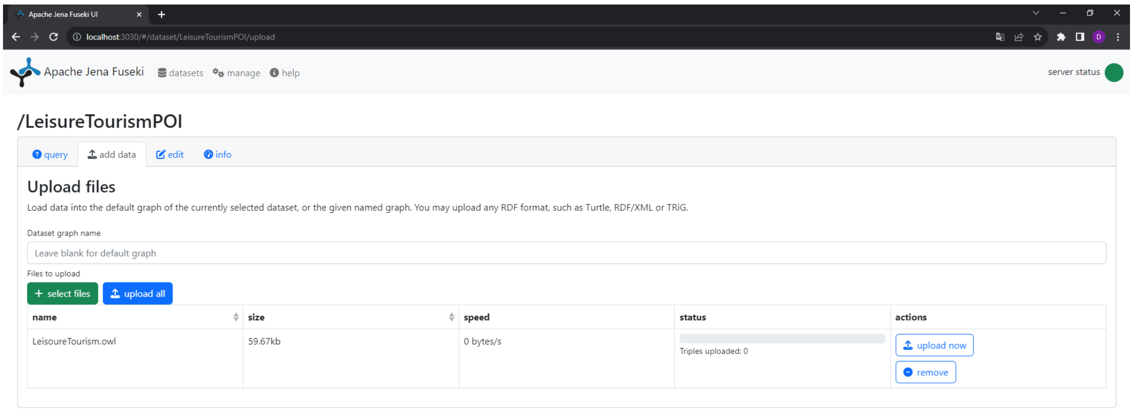


Manage datasets

name	actions
/LeisureTourismPOI	query remove backup add data info

Figure 5.6: Add data to dataset

Clicking on that option, a file can be uploaded and then used to create a graph with the information initially designed in Protégé.



/LeisureTourismPOI

Upload files

Load data into the default graph of the currently selected dataset, or the given named graph. You may upload any RDF format, such as Turtle, RDF/XML or TRiG.

Dataset graph name: Leave blank for default graph

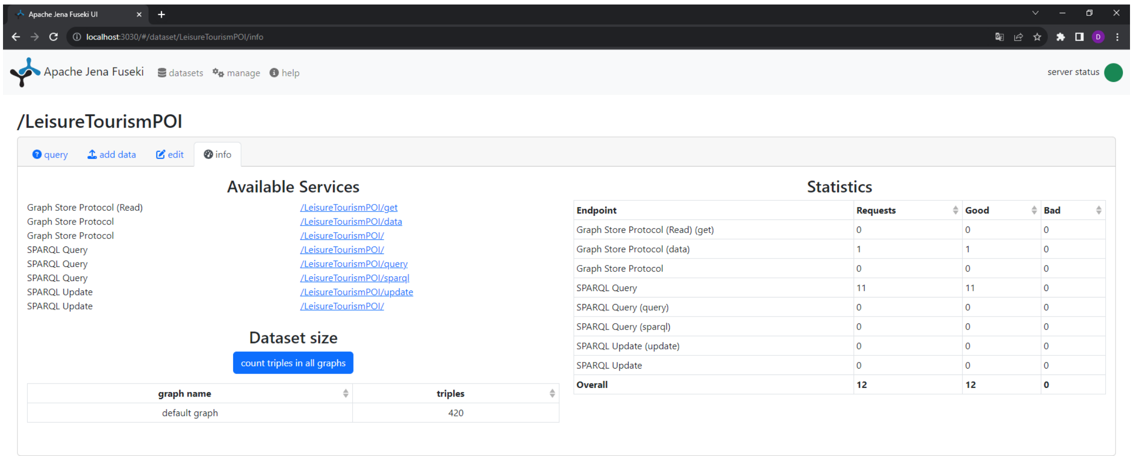
Files to upload

name	size	speed	status	actions
LeisoureTourism.owl	59.67kb	0 bytes/s	Triples uploaded: 0	upload now remove

Figure 5.7: Upload data to dataset

The user can use the same dataset with different graphs and in the upload process the graph can be defined. If not, it will choose the default graph, which is the graph that is going to be used for this project.

Still in the managing datasets section, there is also the info tab, which allows users to see the details of the corresponding dataset, which includes how many requests where made (good and bad), the available services that later are going to be called with REST to consult and query the dataset and its size, that includes the existing graphs and their amount of triplets.



/LeisureTourismPOI

Available Services

Graph Store Protocol (Read)	/LeisureTourismPOI/get
Graph Store Protocol	/LeisureTourismPOI/data
Graph Store Protocol	/LeisureTourismPOI/
SPARQL Query	/LeisureTourismPOI/
SPARQL Query	/LeisureTourismPOI/query
SPARQL Query	/LeisureTourismPOI/sparql
SPARQL Update	/LeisureTourismPOI/update
SPARQL Update	/LeisureTourismPOI/

Dataset size

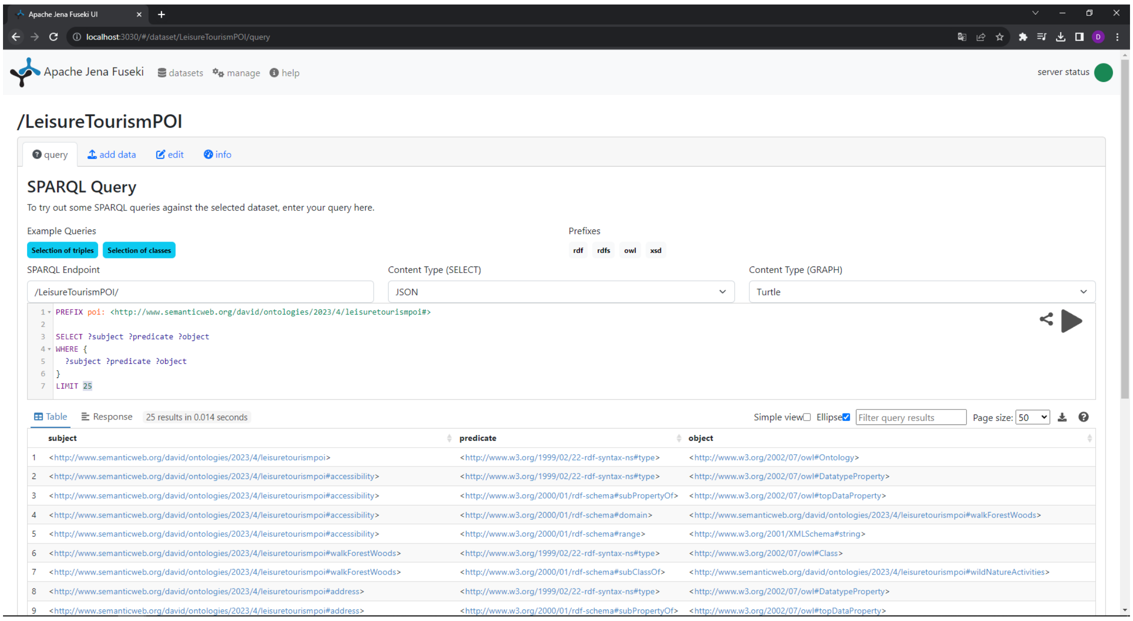
graph name	triples
default graph	420

Statistics

Endpoint	Requests	Good	Bad
Graph Store Protocol (Read) (get)	0	0	0
Graph Store Protocol (data)	1	1	0
Graph Store Protocol	0	0	0
SPARQL Query	11	11	0
SPARQL Query (query)	0	0	0
SPARQL Query (sparql)	0	0	0
SPARQL Update (update)	0	0	0
SPARQL Update	0	0	0
Overall	**12**	**12**	**0**

Figure 5.8: Information about the dataset

There is also the option of querying the database with the interface. This provides the user with feedback and a good way of testing if the graph is well integrated.



/LeisureTourismPOI

SPARQL Query

To try out some SPARQL queries against the selected dataset, enter your query here.

Example Queries

Prefixes

SPARQL Endpoint: /LeisureTourismPOI/

Content Type (SELECT): JSON

Content Type (GRAPH): Turtle

```
PREFIX poi: <http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#>

SELECT ?subject ?predicate ?object
WHERE {
  ?subject ?predicate ?object
}
LIMIT 25
```

25 results in 0.014 seconds

Simple view☐ Ellipse☑ Page size: 50

	subject	predicate	object
1	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#Ontology>
2	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#accessibility>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#DatatypeProperty>
3	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#accessibility>	<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>	<http://www.w3.org/2002/07/owl#topDataProperty>
4	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#accessibility>	<http://www.w3.org/2000/01/rdf-schema#domain>	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#walkForestWoods>
5	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#accessibility>	<http://www.w3.org/2000/01/rdf-schema#range>	<http://www.w3.org/2001/XMLSchema#string>
6	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#walkForestWoods>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#Class>
7	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#walkForestWoods>	<http://www.w3.org/2000/01/rdf-schema#subClassOf>	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#wildNatureActivities>
8	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#address>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#DatatypeProperty>
9	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#address>	<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>	<http://www.w3.org/2002/07/owl#topDataProperty>

Figure 5.9: Querying the dataset

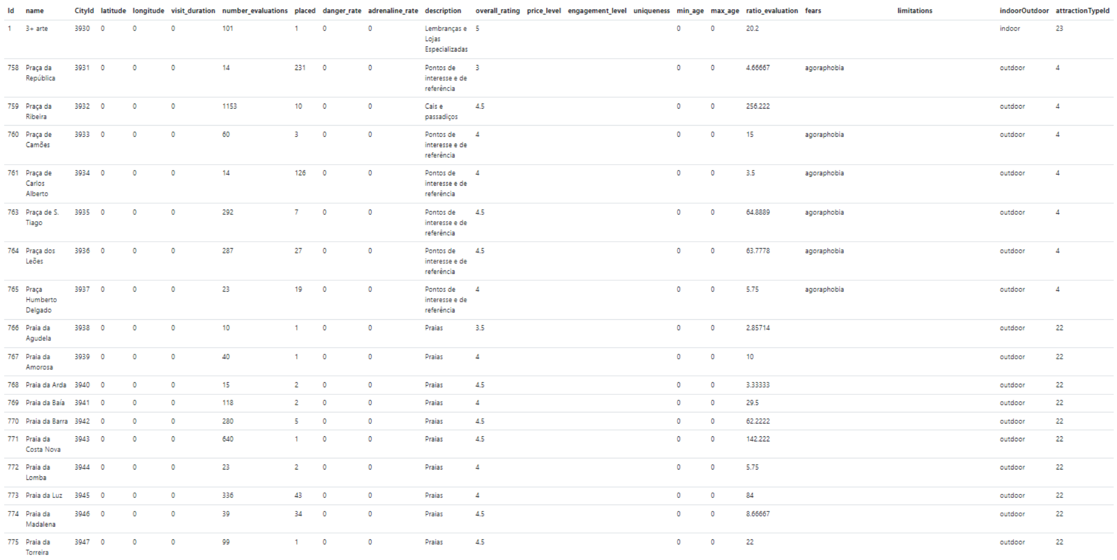
As shown in the image above, the query editor allows to define a SPARQL query and test it on the existing dataset.

5.2 GrouPlanner Integration

After configuring the triple store database, it was necessary to integrate it in the existing GrouPlanner system, more specifically, in the recommendation engine.

5.2.1 Database

Currently, this project is consulting a relational database on Elephant SQL (ElephantSQL 2023) that has the points of interest with some attributes like evaluations, fears, danger rate, adrenaline rate, etc. However, it does not have the relationship with the most important part that needs to be studied, which are the different dimensions of personality.



Id	name	CityId	latitude	longitude	visit_duration	number_evaluations	placed	danger_rate	adrenaline_rate	description	overall_rating	price_level	engagement_level	uniqueness	min_age	max_age	ratio_evaluation	fears	limitations	indoorOutdoor	attractionTypeId
1	3+ arte	3930	0	0	0	101	1	0	0	Lembranças e Lojas Especializadas	5				0	0	20.2			indoor	23
758	Praça da República	3931	0	0	0	14	231	0	0	Pontos de interesse e de referência	3				0	0	4.66667	agoraphobia		outdoor	4
759	Praça da Ribeira	3932	0	0	0	1153	10	0	0	Cais e passadiços	4.5				0	0	256.222			outdoor	4
760	Praça de Camões	3933	0	0	0	60	3	0	0	Pontos de interesse e de referência	4				0	0	15	agoraphobia		outdoor	4
761	Praça de Carlos Alberto	3934	0	0	0	14	126	0	0	Pontos de interesse e de referência	4				0	0	3.5	agoraphobia		outdoor	4
763	Praça de S. Tiago	3935	0	0	0	292	7	0	0	Pontos de interesse e de referência	4.5				0	0	64.8889	agoraphobia		outdoor	4
764	Praça dos Leões	3936	0	0	0	287	27	0	0	Pontos de interesse e de referência	4.5				0	0	63.7778	agoraphobia		outdoor	4
765	Praça Humberto Delgado	3937	0	0	0	23	19	0	0	Pontos de interesse e de referência	4				0	0	5.75	agoraphobia		outdoor	4
766	Praia da Agudela	3938	0	0	0	10	1	0	0	Praias	3.5				0	0	2.85714			outdoor	22
767	Praia da Amorosa	3939	0	0	0	40	1	0	0	Praias	4				0	0	10			outdoor	22
768	Praia da Arda	3940	0	0	0	15	2	0	0	Praias	4.5				0	0	3.33333			outdoor	22
769	Praia da Baía	3941	0	0	0	118	2	0	0	Praias	4				0	0	29.5			outdoor	22
770	Praia da Barra	3942	0	0	0	280	5	0	0	Praias	4.5				0	0	62.2222			outdoor	22
771	Praia da Costa Nova	3943	0	0	0	640	1	0	0	Praias	4.5				0	0	142.222			outdoor	22
772	Praia da Lomba	3944	0	0	0	23	2	0	0	Praias	4				0	0	5.75			outdoor	22
773	Praia da Luz	3945	0	0	0	336	43	0	0	Praias	4				0	0	84			outdoor	22
774	Praia da Madalena	3946	0	0	0	39	34	0	0	Praias	4.5				0	0	8.66667			outdoor	22
775	Praia da Torreira	3947	0	0	0	99	1	0	0	Praias	4.5				0	0	22			outdoor	22

Table 5.1: Current POI table

All the points of interest represented in this database were exported to Protégé in the form of individuals, and some of the columns from the table seen above were replaced as data properties.

5.2.2 Recommendation Engine

The recommendation engine is the part of the GrouPlanner application that generates the recommendation list of points of interest based on preferences predicted by the tourists' personality and ratings given to visited or suggested POI. As said above, currently it retrieves the POI to recommend from a relational database hosted on an Elephant SQL server. Now, with the new triple store SPARQL database, it should consult it instead.

This engine has a standard controller, service, repository pattern. It has two controllers, one for bootstrapping and filling the database with the points of interest, and another to do the recommendation. In the first one, named "POIController" the current methods responsible for loading all things necessary to give a recommendation needed to be replaced by a request to the Apache Jena Fuseki server with an RDF formatted file to import the graph with the new information. This graph will be updated and designed in Protégé and then this new function will only import it. Bellow, there is one example of the current controller.

```
[HttpPost("bootstrapCategories")]
        [AllowAnonymous]
        public async Task<IActionResult> BootstrapCategories()
        {
            try
            {
                await service.BootstrapCategories();

                return Ok("Categories added to the Recommendation System Service!");
            }
            catch (Exception e)
            {
                return BadRequest(new { message = e.Message });
            }
        }
```

Listing 5.1: Current bootstrapper controller for POI categories

Here is the service that it also calls, saving all the categories in the relational database.

```
public async Task BootstrapCategories()
        {
            List<Category> listCategories = new List<Category>(){
                new Category(ADRENALINE_ACTIVITIES, "Activities related to aquatic, air and motorsports, climbing, snorkeling, escape games, etc."),
                new Category(SPORTS_AND_GAMES, "Activities related to fishing/hunting, casino games, sports competitions."),
                new Category(MUSEUMS_BOATTRIPS_VIEWPOINTS, "Activities related to visiting museums, boat trips and viewpoints."),
                new Category(WILD_NATURE_ACTIVITIES, "Activities related to hiking/walking along rivers, forests, mountain and wildlife nature."),
                new Category(GASTRONOMY_EVENTS, "Activities related to gastronomy food tours and festivals and typical dishes."),
                new Category(CULTURAL_HERITAGE, "Activities related to visit of monuments, historic cities/villages."),
                new Category(PARTY_MUSIC_NIGHTLIFE, "Activities related to dancing, music and parades."),
                new Category(HEALTH_AND_WELL_BEING, "Activities related to health and wellness, beauty centers and SPAs."),
                new Category(NATURAL_PHENOMENA, "Activities related to watching a natural phenomenon and visit of caves/caverns/volcanoes."),
                new Category(THEME_AND_ANIMAL_PARKS, "Activities related to theme and water parks, oceanarium and zoos."),
                new Category(SUN_WATER_AND_SAND, "Activities related to beach, island and swimming.")
            };

            foreach (var category in listCategories)
            {
                await categoryRepository.AddObjectAsync(category);
            }

        }
```

Listing 5.2: Current bootstrapper service for POI categories

As for the other controller, it was where the current recommendation system lies. The requests made for this one return one or several points of interest based on users preferences. Currently, the methods used to give this recommendation are very processor heavy, consulting the points of interest first and then making the calculations with the code. That is why using a triple store database with SPARQL and sending the recommendation process to it was a much more efficient and less time consuming way to give recommendations.

```
[HttpPost("groupOpinionRecommendations")]
        [AllowAnonymous]
        public async Task<IActionResult> AssociateOpinionRecommendations
    (GroupDivisionPOIMAMSAfterOpinionDTO dto)
        {
            try
            {
                var result = await service.
    AssociateOpinionRecommendations(dto);
                return Ok(result);
            }
            catch (Exception e)
            {
                return BadRequest(new { message = e.Message });
            }
        }
```

Listing 5.3: Current recommendation controller for POI categories

Chapter 6

Evaluation and Experimentation

Unfortunately, at the moment of writing this document, there is no integration of the database with the GrouPlanner microservice. Due to that, the testing process was conducted differently. So, it focused on the triple store database results, made by querying the database and how they were related to the tourists different personalities and preferences. On top of that, the criteria defined in the Functional and Non-Functional was also tested.

6.1 Meeting Functional and Non-Functional Requirements

In the analysis and design chapter, several functional and non-functional were defined. In general, most of them were achieved, but some were not possible to test.

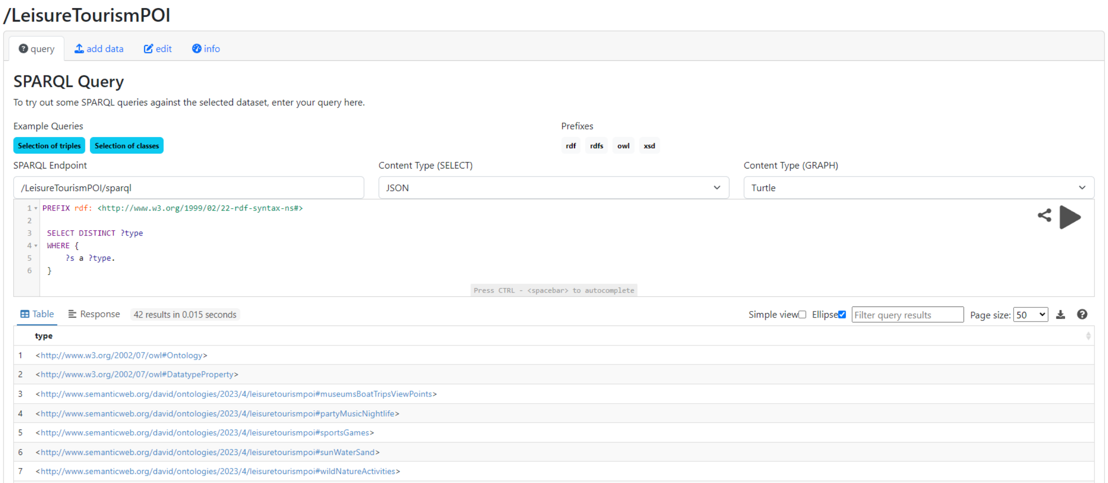
For the functional requirements, is safe to say that both of them were achieved. In the first use case, the system manager can manage the available data sources by using the Apache Fuseki interface. As it was already mentioned in the implementation chapter, the UI was used to manage the data sources, add new information and query the existing graphs and data sets. As for the second functional requirement, it will be better demonstrated ahead in this chapter. But, to summarise it, Apache Jena Fuseki provides several endpoints that were called to give information about the dataset, including the recommended points of interest based on the users personality.

For the non-functional requirements, some were already covered by the GrouPlanner prototype and others were not possible to verify if they were accomplished. In terms of functionality, as shown in the implementation chapter in the GrouPlanner architecture, the different microservices all communicate with the HTTPS protocol. For usability, GrouPlanner's interface is already user-friendly and it does not show the technical data that users may not understand when showing recommendations. However, because the new SPARQL database was not implemented in the GrouPlanner app, the requirement is not fully completed, because the recommendations are not made by the new engine. As for the systems reliability, the current prototype is always available and does not have minimal downtime for maintenance. Again, the new database was not implemented, but, with all the information gathered in the development process of this project, it is possible to conclude that this requirement would be maintained and even improved due to the nature of the triple store database. Due to this same nature, the performance was improved and the data retrieval was faster. Finally, in terms of supportability, this requirement was not achieved, again, because the database was not integrated and the connection was not tested.

6.2 POI Recommendations in Fuseki UI

As mentioned previously in the implementation chapter, the UI of Apache Jena Fuseki allows users to run queries. It is with these queries that the information can be consulted by the recommendation engine and then used to give accurate recommendations to users.

The first query that was tested was one to give information about all classes and subclasses presented in the graph of the dataset. For a positive outcome, the query should return all the classes and subclasses that compose the dataset, with no individuals or data properties.



To try out some SPARQL queries against the selected dataset, enter your query here.

SPARQL Endpoint: /LeisureTourismPOI/sparql

Content Type (SELECT): JSON

Content Type (GRAPH): Turtle

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

 SELECT DISTINCT ?type
 WHERE {
     ?s a ?type.
 }
```

42 results in 0.015 seconds

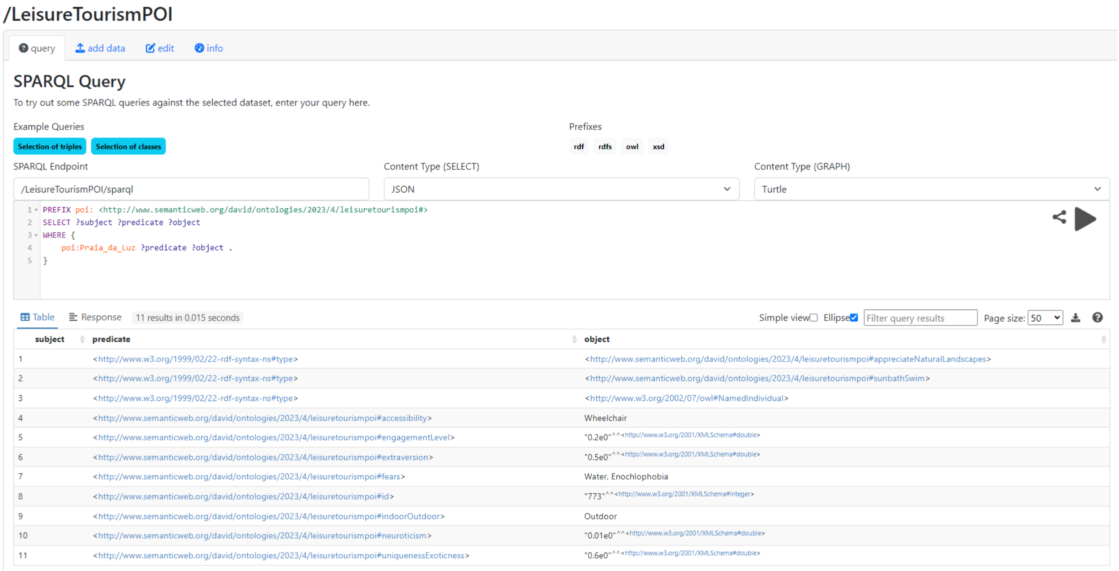
Simple view☐ Ellipse☑ Filter query results Page size: 50

	type
1	<http://www.w3.org/2002/07/owl#Ontology>
2	<http://www.w3.org/2002/07/owl#DatatypeProperty>
3	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#museumsBoatTripsViewPoints>
4	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#partyMusicNightlife>
5	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#sportsGames>
6	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#sunWaterSand>
7	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#wildNatureActivities>

Figure 6.1: Query to test classes and subclasses

As shown in the results tab, there are 42 results and all of them correspond to the classes and subclasses of the ontology, as expected.

To search for a specific individual, the query presented in the image bellow was tested. In this case, the POI chosen was "Praia da Luz" and the query should return all its properties and the classes that it belongs to.



To try out some SPARQL queries against the selected dataset, enter your query here.

SPARQL Endpoint: /LeisureTourismPOI/sparql

Content Type (SELECT): JSON

Content Type (GRAPH): Turtle

```
PREFIX poi: <http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#>
SELECT ?subject ?predicate ?object
WHERE {
    poi:Praia_da_Luz ?predicate ?object .
}
```

11 results in 0.015 seconds

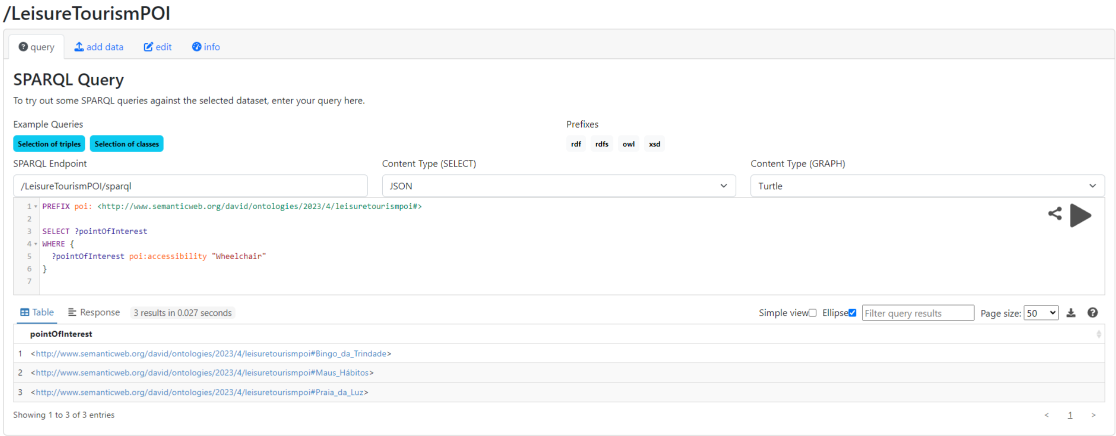
Simple view☐ Ellipse☑ Filter query results Page size: 50

	subject	predicate	object
1		<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#appreciateNaturalLandscapes>
2		<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#sunbathSwim>
3		<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
4		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#accessibility>	Wheelchair
5		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#engagementLevel>	"0.2e0"^^<http://www.w3.org/2001/XMLSchema#double>
6		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#extraversion>	"0.5e0"^^<http://www.w3.org/2001/XMLSchema#double>
7		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#fears>	Water, Enochlophobia
8		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#id>	"773"^^<http://www.w3.org/2001/XMLSchema#integer>
9		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#indoorOutdoor>	Outdoor
10		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#neuroticism>	"0.01e0"^^<http://www.w3.org/2001/XMLSchema#double>
11		<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#uniquenessExoticness>	"0.6e0"^^<http://www.w3.org/2001/XMLSchema#double>

Figure 6.2: Query to test the specific individual

In the results tab, it is possible to see all the properties that the POI has, as well as both categories that it belongs to: "appreciateNaturalLandscapes" and "sunbathSwim". This also shows the potential of the triple store databases in that you can have an individual that is classified to two different categories.

On this example, the query was made to check if the database will return places that are accessible to people in wheelchair. For a positive outcome, the results should be the points of interest "Maus Hábitos", "Bingo da Trindade" and "Praia da Luz".



SPARQL Query

To try out some SPARQL queries against the selected dataset, enter your query here.

SPARQL Endpoint: /LeisureTourismPOI/sparql

Content Type (SELECT): JSON

Content Type (GRAPH): Turtle

```
PREFIX poi: <http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#>

SELECT ?pointOfInterest
WHERE {
  ?pointOfInterest poi:accessibility "Wheelchair"
}

```

3 results in 0.027 seconds

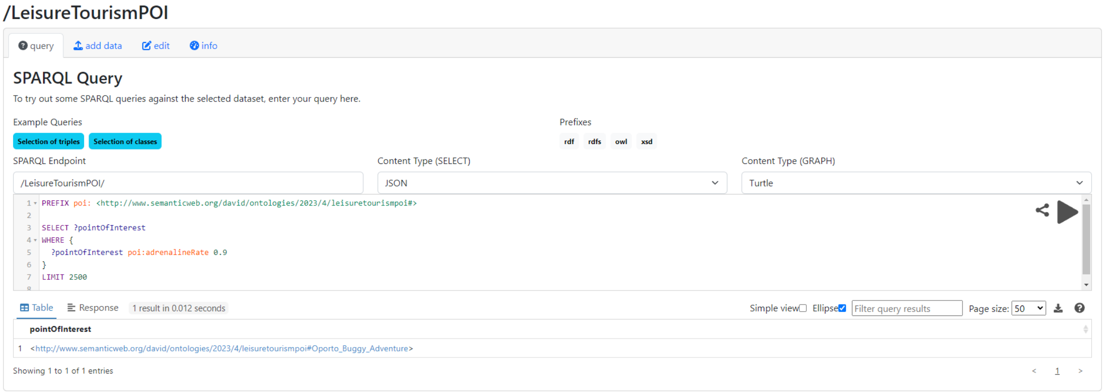
	pointOfInterest
1	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#Bingo_da_Trindade>
2	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#Maus_Hábitos>
3	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#Praia_da_Luz>

Showing 1 to 3 of 3 entries

Figure 6.3: Query to test the accessibility

In the results tab, it is possible to see the points of interest represented by their URI. As predicted, the results were "Maus Hábitos" and "Bingo da Trindade" and "Praia da Luz".

Another test that was made to check a specific value of the adrenaline rate. The value used was 0.9 and the only attraction/POI with that value was "Oporto Buggy Adventure".



SPARQL Query

To try out some SPARQL queries against the selected dataset, enter your query here.

SPARQL Endpoint: /LeisureTourismPOI/

Content Type (SELECT): JSON

Content Type (GRAPH): Turtle

```
PREFIX poi: <http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#>

SELECT ?pointOfInterest
WHERE {
  ?pointOfInterest poi:adrenalineRate 0.9
}
LIMIT 2500
```

1 result in 0.012 seconds

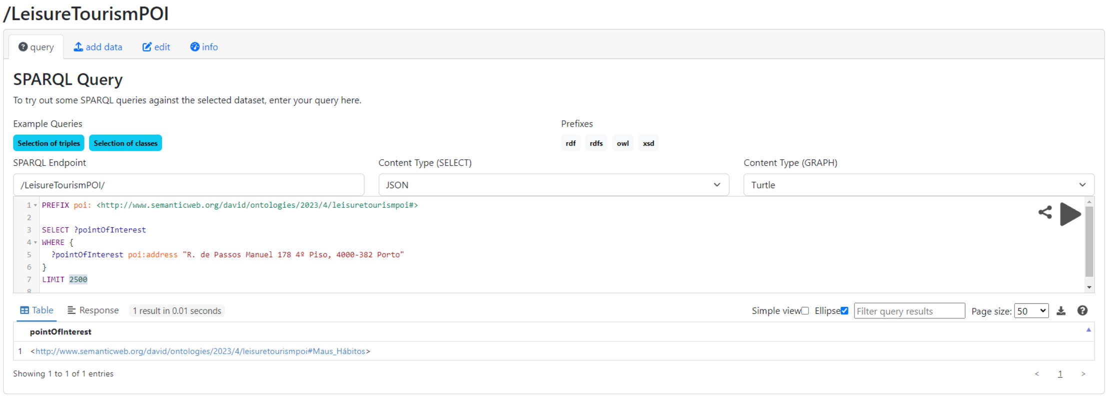
	pointOfInterest
1	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#Oporto_Buggy_Adventure>

Showing 1 to 1 of 1 entries

Figure 6.4: Query to test the adrenaline rate

Again, the results were as expected. It returned one URI and it was the one corresponding to "Oporto Buggy Adventure".

The fifth test was one to just simply search by an attribute that doesn't relate to the individual's characteristics by itself, but rather to the POI's address. In this case, the address "R. de Passos Manuel 178 4º Piso, 4000-382 Porto" corresponds to the point of interest "Maus Hábitos".



To try out some SPARQL queries against the selected dataset, enter your query here.

Example Queries

Prefixes

SPARQL Endpoint: /LeisureTourismPOI/

Content Type (SELECT): JSON

Content Type (GRAPH): Turtle

```
PREFIX poi: <http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#>

SELECT ?pointOfInterest
WHERE {
  ?pointOfInterest poi:address "R. de Passos Manuel 178 4º Piso, 4000-382 Porto"
}
LIMIT 2500
```

1 result in 0.01 seconds

Simple view□ Ellipse☑ Filter query results Page size: 50

	pointOfInterest
1	<http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#Maus_Hábitos>

Showing 1 to 1 of 1 entries

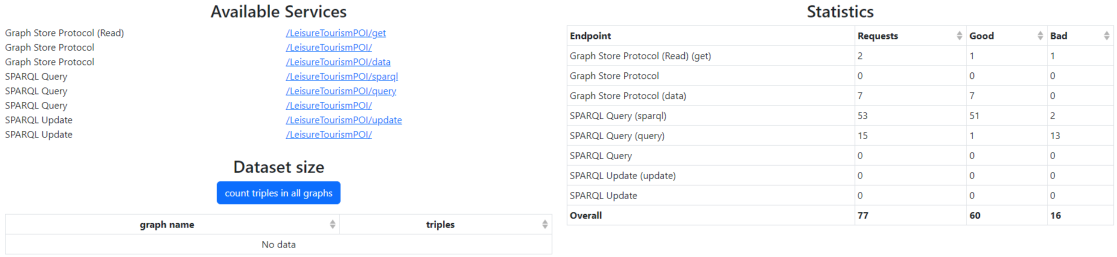
< 1 >

Figure 6.5: Query to test the address

The results were what was expected.

6.3 POI Recommendations in REST Services

For an easier explanation and viewing in the querying process, the tests were made using the Apache Jena Fuseki UI. However, to really implement the application and make it function in the GrouPlanner prototype, the querying process needs to be made in the API of Apache Jena Fuseki. The different endpoints available for querying and getting information about the database are show in the image below.



Available Services

Graph Store Protocol (Read)	/LeisureTourismPOI/get
Graph Store Protocol	/LeisureTourismPOI/
Graph Store Protocol	/LeisureTourismPOI/data
SPARQL Query	/LeisureTourismPOI/sparql
SPARQL Query	/LeisureTourismPOI/query
SPARQL Query	/LeisureTourismPOI/
SPARQL Update	/LeisureTourismPOI/update
SPARQL Update	/LeisureTourismPOI/

Dataset size

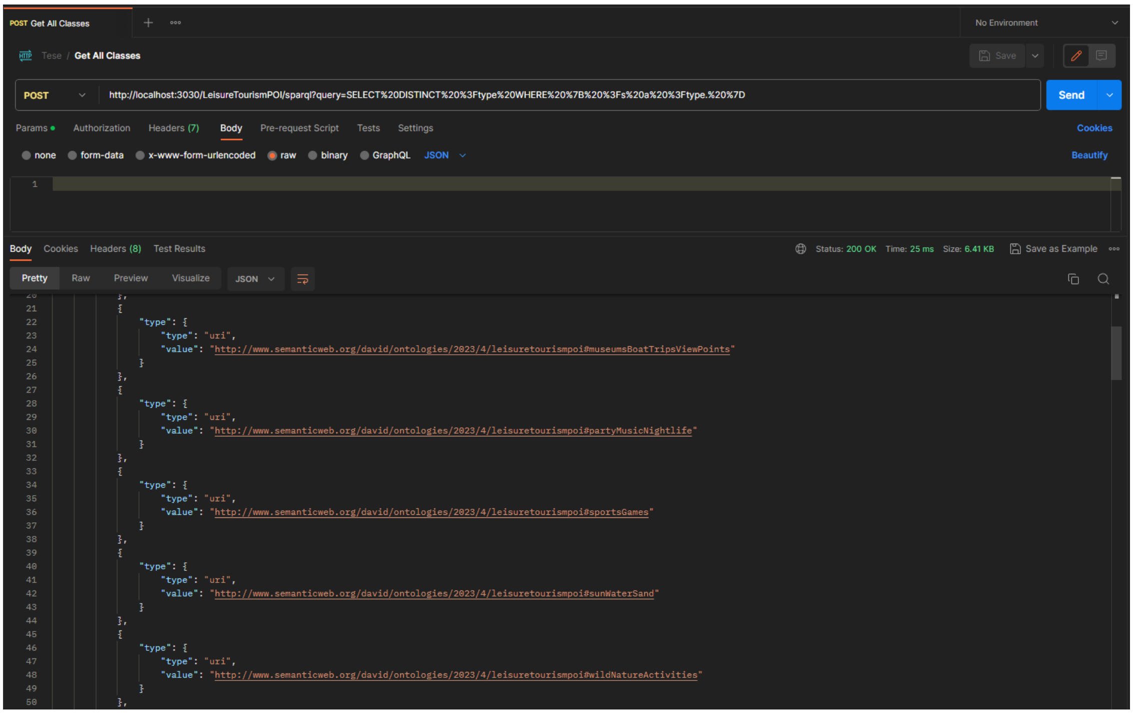
graph name	triples
No data	

Statistics

Endpoint	Requests	Good	Bad
Graph Store Protocol (Read) (get)	2	1	1
Graph Store Protocol	0	0	0
Graph Store Protocol (data)	7	7	0
SPARQL Query (sparql)	53	51	2
SPARQL Query (query)	15	1	13
SPARQL Query	0	0	0
SPARQL Update (update)	0	0	0
SPARQL Update	0	0	0
Overall	**77**	**60**	**16**

Figure 6.6: Different endpoints for Apache Jena Fuseki

To get the same results shown in the querying UI (in this case it was used the first example shown in point 6.2 of the chapter), an HTTP POST request must be sent to the Endpoint "/LeisureTourismPOI/SPARQL", as shown in the image below.



POST http://localhost:3030/LeisureTourismPOI/sparql?query=SELECT%20DISTINCT%20%3Ftype%20WHERE%20%7B%20%3Fs%20a%20%3Ftype.%20%7D

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

Status: 200 OK Time: 25 ms Size: 6.41 KB

JSON

```
            },
            {
                "type": {
                    "type": "uri",
                    "value": "http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#museumsBoatTripsViewPoints"
                }
            },
            {
                "type": {
                    "type": "uri",
                    "value": "http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#partyMusicNightlife"
                }
            },
            {
                "type": {
                    "type": "uri",
                    "value": "http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#sportsGames"
                }
            },
            {
                "type": {
                    "type": "uri",
                    "value": "http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#sunWaterSand"
                }
            },
            {
                "type": {
                    "type": "uri",
                    "value": "http://www.semanticweb.org/david/ontologies/2023/4/leisuretourismpoi#wildNatureActivities"
                }
            },
```

Figure 6.7: Postman request to check all classes and subclasses

The results are presented in the form of a JSON file, were it is possible to check the different results, in this case, them being the classes and subclasses. The query is sent in the header of the request as a query parameter and it has to be URL encoded. In the future, the recommendation microservice would take the JSON file received and treat the data to feed the user with information.

Chapter 7

Conclusions

7.1 Objectives Achieved

The objectives of this research project were aimed at advancing the understanding of recommender systems, the psychology of tourism and destination personality, personality-driven user preferences, and knowledge representation through ontologies. Unfortunately, due to time constraints and limited knowledge in certain areas, not all planned objectives were accomplished.

However, it is important to note that significant progress was made in the initial stages of the project. The first four objectives, which involved contextualising the state of the art in recommender systems, psychology of tourism, and user preferences, and the formalisation and development of an ontology for tourist attractions, were successfully concluded. This foundation of knowledge has provided valuable insights into the field and laid the groundwork for future research in these areas as well as for future ontology development.

Regrettably, the latter objectives related to, its integration into the existing recommendation engine, and real-world testing using use-case scenarios were not fully realised. These tasks proved to be more complex and time-consuming than initially anticipated, and the limited expertise in ontology development further compounded the challenges. However, there is still a good ground foundation for the implementation due to the creation and understanding of a SPARQL database.

While not all objectives were completed as initially planned, this research project has nonetheless contributed to the advancement of knowledge in the fields of recommender systems and tourism psychology. It serves as a stepping stone for future endeavours in these areas and highlights the importance of effective time management and expertise in achieving research goals.

7.2 Limitations and Future Work

While this research project has made significant strides in understanding recommender systems, the psychology of tourism, and related areas, there are limitations and promising avenues for future work, particularly concerning the implementation of these findings in the GroupPlanner App and conducting practical testing.

The first limitation lies in the actual integration of the developed ontology and recommendation engine into the GroupPlanner App. Due to time constraints and the complexities involved in seamless integration with an existing application, this crucial step could not be realised as initially planned. However, this remains a critical next step.

In the future, efforts should focus on a robust integration process that ensures the smooth functioning of the recommendation engine within the GrouPlanner App. This entails not only the technical aspects but also ensuring that user experiences are enhanced through personalised recommendations for tourist attractions and destinations.

Another limitation pertains to the lack of empirical validation through real-world testing scenarios. While the development of the ontology and recommendation engine has been undertaken, actual testing in real use-case scenarios has not been executed. This is crucial for assessing the practical applicability and effectiveness of the proposed system.

Future work should prioritise the design and execution of real-world testing scenarios within the GrouPlanner App. This would involve collecting user feedback and evaluating the system's performance in providing accurate and personalised recommendations. Such testing is essential to validate the research findings and ensure that the recommendations align with the preferences and needs of users.

Furthermore, it is important to recognise that user feedback and iterative improvements are integral to the success of any recommendation system. Implementing a feedback mechanism within the GroupPlanner App will allow for continuous refinement of the ontology and recommendation algorithms. This iterative process will enhance the system's ability to provide increasingly accurate and tailored recommendations over time.

Addressing the limitation related to limited expertise in ontology development is also essential. Future work should include training and skill development for the research team or collaboration with experts in ontology engineering to ensure a more effective development process.

7.3 Final Consideration

In retrospect, this research project has been a journey marked by both accomplishments and challenges. The objectives set out to advance the understanding of recommender systems, the psychology of tourism, user preferences, and knowledge representation through ontologies. In this regard, the project can be viewed as a success.

The initial phases of the research project were fruitful, as they allowed us to contextualise the state of the art in the relevant fields and accumulate valuable knowledge. This foundational work not only contributed to understand but also laid the groundwork for future research endeavours in the domains of recommender systems and tourism psychology.

However, as with any ambitious research project, there were limitations and unfulfilled objectives. The integration of the developed ontology and recommendation engine into the GroupPlanner App and real-world testing were areas where constraints, including time and expertise, posed significant challenges. These uncompleted tasks highlight the complexity and intricacy of translating research findings into practical applications.

Nonetheless, these challenges should not overshadow the achievements and progress made during the project. It is important to recognise that research, particularly in multidisciplinary domains like this, often unfolds in iterative stages. The insights gained and the groundwork laid in this project provide a solid foundation for future work.

Bibliography

1000sourcecodes (2023). *Software Engineering-FURPS*. url: https://www.1000sourcecodes.com/2012/05/software-engineering-furps.html (visited on 08/25/2023).

Abbasi-Moud, Z. et al. (Aug. 2022). "CAFOB: Context-aware fuzzy-ontology-based tourism recommendation system". In: *Expert Systems with Applications* 199. url: https://www.sciencedirect.com/science/article/pii/S0957417422003232.

Alves, P., J. Carneiro, P. Novais, et al. (2019). "Modeling a Mobile Group Recommender System for Tourism with Intelligent Agents and Gamification". In: url: https://www.gecad.isep.ipp.pt/grouplanner/assets/files/HAIS2019.pdf.

Alves, P., J. Carneiro, P. Saraiva, et al. (Mar. 2023). "Group recommender systems for tourism: how does personality predict preferences for attractions, travel motivations, preferences and concerns?" In: url: https://www.gecad.isep.ipp.pt/grouplanner/assets/files/UMUAI2023.pdf.

Alves, P., G. Domingos, et al. (July 2020). "Modeling Tourists' Personality in Recommender Systems". In: url: http://www.gecad.isep.ipp.pt/grouplanner/assets/files/UMAP2020.pdf.

Alves, P., P. Saraiva, et al. (2022). "Grouplanner: a Group Recommender System for Tourism with Multi-Agent MicroServices". In: pp. 1–6. url: https://www.gecad.isep.ipp.pt/grouplanner/assets/files/PAAMS2022.pdf.

Amzad, H. and K. Vijayalakshmi (Sept. 2021). "Tourism Recommendation System: A Systematic Review". In: *International Journal of Engineering Research Technology (IJERT)* 10. url: https://www.ijert.org/research/tourism-recommendation-system-a-systematic-review-IJERTV10IS090100.pdf.

Autoblocks (2023). *knowledge representation and reasoning*. url: https://www.autoblocks.ai/glossary/knowledge-representation-and-reasoning (visited on 08/25/2023).

Blazegraph (2023). *Welcome to Blazegraph*. url: https://blazegraph.com/ (visited on 08/25/2023).

Borràs, J., A. Moreno, and A. Valls (2014). "Intelligent tourism recommender systems: A survey". In: *Expert Systems with Applications* 41, pp. 7370–7389. url: https://www.sciencedirect.com/science/article/pii/S0957417414003431.

Company, McKinsey (2023). *The future of tourism: Bridging the labor gap, enhancing customer experience*. url: https://www.mckinsey.com/industries/travel-logistics-and-infrastructure/our-insights/future-of-tourism-bridging-the-labor-gap-enhancing-customer-experience (visited on 08/01/2023).

Elena, Ş (2020). "International Tourism In Globalization Conditions". In: pp. 1–10. url: https://anale.steconomiceuoradea.ro/volume/2020/n1/012.pdf.

ElephantSQL (2023). *Documentation*. url: https://www.elephantsql.com/docs/index.html (visited on 08/25/2023).

GECAD (2023). *GrouPlanner*. url: https://www.gecad.isep.ipp.pt/grouplanner/ (visited on 08/25/2023).

Group, OWL Working (2012). *Web Ontology Language (OWL)*. url: https://www.w3.org/OWL/ (visited on 12/11/2012).

JavaPoint (2021). *knowledge representation and reasoning*. url: https://www.javatpoint.com/knowledge-representation-in-ai (visited on 08/25/2023).

Jena, Apache (2023a). *Apache Jena Fuseki*. url: https://jena.apache.org/documentation/fuseki2/ (visited on 08/25/2023).

– (2023b). *Apache Jena Releases*. url: https://jena.apache.org/download/ (visited on 08/25/2023).

Koen, P. (Mar. 2001). "Providing Clarity and Common Language to the "Fuzzy Front End"". In: *Research Technology Management* 42.2, pp. 46–56.

Massimo, M and F Ricci (June 2022). "Building effective recommender systems for tourists". In: *AI Magazine* 26, pp. 209–224. url: https://onlinelibrary.wiley.com/doi/10.1002/aaai.12057.

Moreno, A. et al. (2013). "SigTur/E-Destination: Ontology-based personalized recommendation of Tourism and Leisure Activities". In: *Engineering Applications of Artificial Intelligence* 26, pp. 633–651. url: https://www.sciencedirect.com/science/article/pii/S0957417414003431.

Navigli, R., P. Velardi, and A. Gangemi (Feb. 2003). "Ontology learning and its automated terminology translation". In: url: https://www.researchgate.net/publication/3454078_Ontology_learning_and_its_automated_terminology_translation.

Ontotext (2023a). *What is an RDF Triplestore?* url: https://www.ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/ (visited on 08/25/2023).

– (2023b). *What is SPARQL?* url: https://www.ontotext.com/knowledgehub/fundamentals/what-is-sparql/ (visited on 08/25/2023).

Osterwalder, P. and Y. Pigneur (Sept. 2003). "Modeling value propositions in e-Business". In: *Proceedings of the 5th international conference on Electronic commerce*, pp. 429–436. url: https://dl.acm.org/doi/10.1145/948005.948061.

Software, OpenLink (2019). *Data-driven agility without compromise*. url: https://virtuoso.openlinksw.com/ (visited on 08/25/2023).

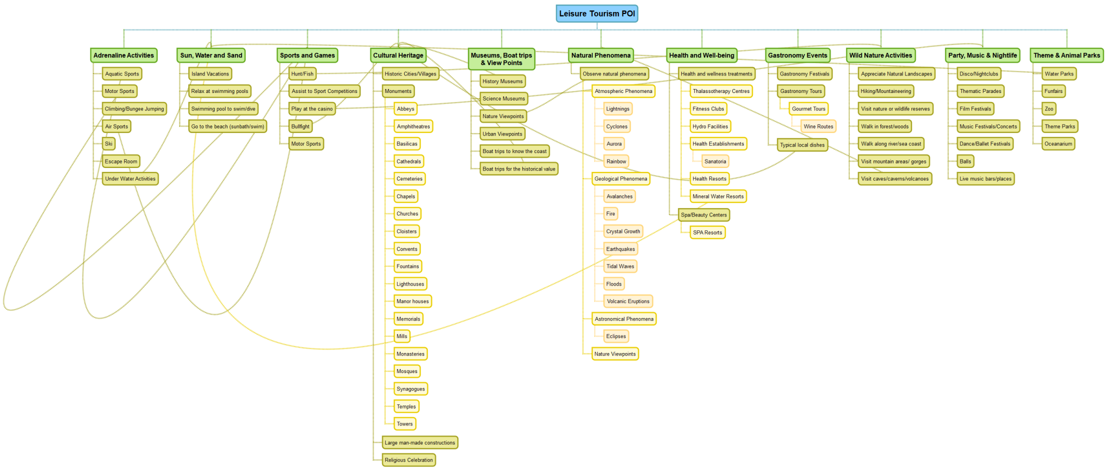
Tkalcic, M. and L. Chen (Sept. 2015). "Recommender Systems Handbook". In: *Recommender Systems Handbook*, pp. 715–739.

University, Stanford (2020). *Protégé*. url: https://protege.stanford.edu/software.php (visited on 08/25/2023).

V. Siricharoen, W. (July 2007). "Using Ontologies for E-tourism". In.

Appendix A

Mind Map Ontology


Leisure Tourism POI
Adrenaline Activities
Aquatic Sports
Motor Sports
Climbing/Bungee Jumping
Air Sports
Ski
Escape Room
Under Water Activities
Sun, Water and Sand
Island Vacations
Relax at swimming pools
Swimming pool to swim/dive
Go to the beach (sunbath/swim)
Sports and Games
Hunt/Fish
Assist to Sport Competitions
Play at the casino
Bullfight
Motor Sports
Cultural Heritage
Historic Cities/Villages
Monuments
Abbeys
Amphitheatres
Basilicas
Cathedrals
Cemeteries
Chapels
Churches
Cloisters
Convents
Fountains
Lighthouses
Manor houses
Memorials
Mills
Monasteries
Mosques
Synagogues
Temples
Towers
Large man-made constructions
Religious Celebration
Museums, Boat trips & View Points
History Museums
Science Museums
Nature Viewpoints
Urban Viewpoints
Boat trips to know the coast
Boat trips for the historical value
Natural Phenomena
Observe natural phenomena
Atmospheric Phenomena
Lightnings
Cyclones
Aurora
Rainbow
Geological Phenomena
Avalanches
Fire
Crystal Growth
Earthquakes
Tidal Waves
Floods
Volcanic Eruptions
Astronomical Phenomena
Eclipses
Nature Viewpoints
Health and Well-being
Health and wellness treatments
Thalassotherapy Centres
Fitness Clubs
Hydro Facilities
Health Establishments
Sanatoria
Health Resorts
Mineral Water Resorts
Spa/Beauty Centers
SPA Resorts
Gastronomy Events
Gastronomy Festivals
Gastronomy Tours
Gourmet Tours
Wine Routes
Typical local dishes
Wild Nature Activities
Appreciate Natural Landscapes
Hiking/Mountaineering
Visit nature or wildlife reserves
Walk in forest/woods
Walk along river/sea coast
Visit mountain areas/ gorges
Visit caves/caverns/volcanoes
Party, Music & Nightlife
Disco/Nightclubs
Thematic Parades
Film Festivals
Music Festivals/Concerts
Dance/Ballet Festivals
Balls
Live music bars/places
Theme & Animal Parks
Water Parks
Funfairs
Zoo
Theme Parks
Oceanarium