Veritas University Abuja
(The Catholic University of Nigeria)

Implementing a REST API for a Description of all Countries in the World Using Vue.js to Define the User Interface

Ata, Chinonso Anita
VUG/CSC/17/1914

Submitted to

The Department of Computer and Information Technology
College of Natural and Applied Sciences

In Partial Fulfilment of the Requirement of Bachelors Degree of Computer Science

November 24, 2020

Abstract

Abstract goes here...

Acknowledgements

Acknowledgement goes here...

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

In the world today, having facts at your fingertips is very useful and important. With search engines like Google, life has been made easier as we do not have to visit libraries or spend money on books to get the most basic facts or knowledge. We now have all the information in the world in one place, and it is easy to gain access to it by just typing in a keyword or two and we have what we want.

What this application will do is to take this functionality further by putting all the countries in the world and their basic information such as their capital, population, continent, etc. in one place. Just like you would have in google, you could just go to the search box and type the name of any country in the world and you get information about that country that you're looking for.

As Technology is being incorporated in our everyday lives and things are being made easier each day it shouldn't stop us from still looking for ways to improve on these technologies to further make things easier in order to keep the world moving forward.

1.2 Aims and Objectives

By having a new interactive system that consists of a layout that displays all the countries in the world, the system can help users who just need quick and simple information about any country. It can also be used by students for assignments related to the subject and it can also be a fun way to learn about different countries in the world.

Thus, the application aims to produce a simple, interesting, and easy to use application that the user can refer to and rely on at anytime to give them the basic facts about any country in the world.

The objectives for developing the Rest Countries Application are as follows:

- To design a simple system and interface that can easily be viewed by any user to search for any country in the world and view simple information about the country that was searched for.
- To develop a system that is accessible anywhere and on any device

1.3 Features of the System

The project is intended to produce an interactive system so that the user can feel interested to use the system. This this is achieved by implementing interactivity especially in the design of the system, a Graphical User Interface (GUI), and portability.

1.3.1 Interactivity

The definition of interaction is quite broad. (Aoki, 2000) stated that interactivity of a medium refers to a characteristic of communication settings a medium can create that allows users to interact.

Throughout the process of interaction design, the developer must be aware of key aspects in their design that influence emotional response in target users. The need for products to convey positive emotions and avoid negative ones is critical to any product success. These aspects include positive, negative, motivational, learning, creative, social and persuasive influences. A method that can be used to convey such aspects is the use of expressive interfaces. (Kamari, 2011)

In software, the use of of dynamic icons, animations and sound can help communicate a state of operation which in turn will create a sense of interactivity. Interface aspects such as fonts, colour palette, and graphical layouts can also influence an interface's perceived effectiveness.

1.3.2 Graphical User Interface

A graphical user interface (GUI) is a type of user interface item that allows people to interact with programs in more ways than typing. A GUI offers graphical icons, and visual indicators, as opposed to text-based interfaces, typed command labels or text navigation to fully represent the information and actions available to a user. The actions are usually performed through direct manipulation of the graphical elements.

There are several principles that need to be considered when dealing with a GUI.

- Layout
  The interface should be a series of areas on the screen that are used consistently for different purposes.
- Content Awareness
  Users should always be aware of where they are in the system and what information is being displayed.
- Aesthetic
  Interface should be functional and inviting to users through careful use of white space, colours, and fonts. In this project for example, there is an option for the user to change the colour theme from light mode to dark mode and vice versa which makes the user feel like they are in control.
- User Experience
  The interface should be built in such a way that it is both easy to use and easy to learn. Novice users or infrequent users of software will prefer ease of learning and frequent users will prefer ease of use.
- Consistency
  Consistency in interface design enables users to predict what will happen before they perform a function. It is one of the important elements in ease of learning, ease of use, and aesthetic.

1.3.3 Responsiveness

The system is a full web application and this enables the system to be viewed anywhere no matter the device and the design is also fully responsive so the layout is still engaging whether the application is viewed on a small device or a very large device like a television.

Chapter 2

Methodology

2.1 Introduction

The system will be implemented using the waterfall model of system development life cycle as the project methodology. This methodology is selected because of its advantage which allows the requirements to be specified at the start of the project and for proper documentation.

The Waterfall Methodology is a linear approach to software development. It is also known as the 'Traditional Approach' because it is time-tested and easy to understand.(Adenowo & Adenowo, 2013). The waterfall methodology breaks up the software development projects into steps: planning and analysis (Requirement Analysis), design and implementation, testing, system deployment, and maintenance. See Figure 2.1 for an illustration.

2.2 Project Activities

2.2.1 Requirement Analysis

In this phase, all the possible requirements of the system such as functional requirements, programming tools to be used, feasibility, and scope are captured and documented.(Petersen et al., 2009). The overall project is intended to come out with an interactive system that lets the user search for any country in the world

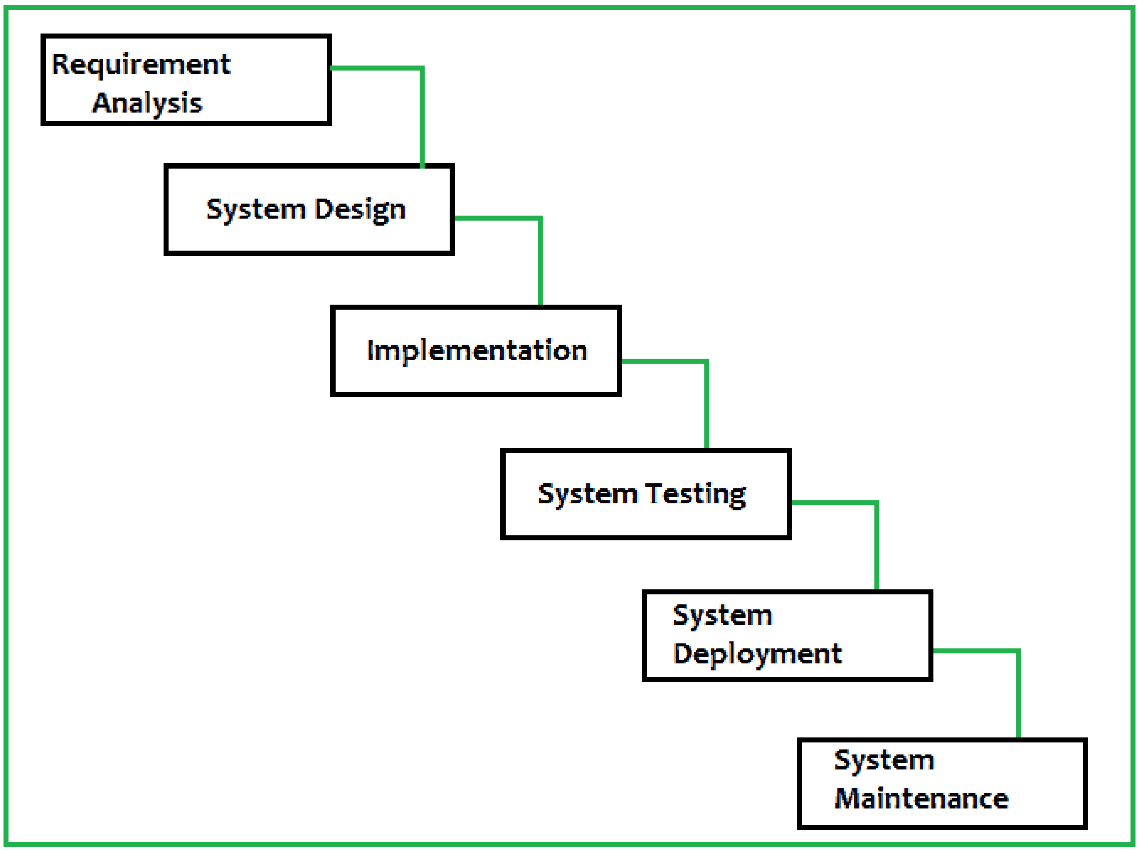

Figure 2.1: Waterfall Methodology

and get information about that country.

At the beginning of the requirements analysis phase, the first thing that has been discussed is the programming tool that will be used which is Vue.js for the client-side, Node js with Express for the server-side and MongoDb for the database.

2.2.2 Design and Implementation

In the design phase, the actual database or file structure, user interface, system inputs and outputs is designed. Thereafter, the actual developement of the system takes place. A basic unit test is also conducted to verify that each component meets its requirement before handing the developed code over to the testing phase.

2.2.3 Testing

In this phase the system integration is tested regarding quality and functional aspects. In this case, the system will be tested by different users on their devices to make sure everything is working as it should. Furthermore, any response will be taken into consideration to improve the system in future.

2.2.4 System Deployment

In this phase when the application has been fully tested, the whole system will be transferred from development mode to build mode. The system will then be brought into shippable state. The database will be set up on MongoDb Atlas and the application itself will be deployed to Heroku.

2.2.5 Maintenance

After the product has been released, there might still be some problems in the user environment. Fixing those issues will require regular maintenance and patches to be released.

Chapter 3

System Analysis

3.1 Introduction

The term *analysis* refers to breaking a whole into parts with the goal of understanding the parts' nature, function, and interrelationships. The purpose is to give a clear picture of the system in terms of capability required and what the software system is required to do.(Dennis et al., 2009)

System Analysis is a method of problem-solving that deals with the breaksing down of a system into component parts in order to study how well the individual parts work and interact to acheive their purpose.

The basic process of system analysis involves three steps:

- Understand the existing situation
- Identify improvements
- Define the requirements for the new system

3.2 Requirement Analysis

A requirement is simply a statement of what the system must do or what characterisitics it needs to have. During a systems development project, requirements will be created that describes what the business needs (business requirements); what the users need to do (user requirements); what the software should do (functional requirements); characterisitics the system should have (nonfunctional requirements); and how the system should be built (system requirements).

This section includes the requirements of the system that are categorized into user requirements, functional requirements and non-functional requirements.

3.2.1 User Requirements

User requirements are the users' expectation from the system as well as the characterisitics it possess in order to fully, effectively and efficiently interact with the system. These requirements are as follows:

- The system should be user friendly and interactive.
- The system should be accessible on all platforms.
- The system should allow the administrator to authenticate and validate user information by comparing the credentials that the user has provided and what is existing in the database.
- The system should be secured by passwords.
- The system should be able to display information about all the countries in the world.

3.2.2 Functional Requirements

Functional requirements capture the intended services, functions or tasks that the system provides and they include the following:

- The system should authenticate users by allowing only users with correct user name and password to access the system.
- The system should be able to register users with detailed information and create accounts for the users.
- With this system, the user should be able to view information about any country in the world.

3.2.3 Non-Functional Requirments

These are requirements that are not directly concerned with the specific behaviour of the system but rather the criteria that can be used to judge the operation of the system and these include:

- Maintainability of the system is easy and cheap to maintain and work with.
- Accessissility of the system is guaranteed to only authorised users by use of passwords and user name.
- The system should operate on all platforms and on any device.
- The system is portable and lightweight and does not use large storage space as well having no impact on the platform performance.
- The system interface is simple and interactive.

3.3 High-level Constituent Parts

3.3.1 Database Management

The database will be managed by the administrator and it will have the following characteristics:

- The Database will be accessible by the software.
- The Database will allow the users to store their information.
- The Database will enable user data to be queried from the database.

3.3.2 API Management

- The API will be accessible by the software.
- The API will allow users to search for data.
- The API will allow users to view information about the data searched for.

3.3.3 Software Management

- The software will be accessible from all platforms
- The software will be able to interact with the database to retrieve data.
- The software will be able to add data to the database
- The software will be able to retrieve data from the database
- The software will be able to fetch data from the API
- The software will be able to display data gotten from API

3.4 Technologies to be Used

It is recommended to use a Web-based technology for the system (using Vue.js for the client-side and Node js with Express and MongoDb for the server-side). The advantage of using web-based technologies is:

- It is easy to use (with user-friendly interfaces)
- It is free (doesn't require any liscence)
- It is cheaper
- It is easier to manage and maintain

3.4.1 Vue.js

Commonly referred to as Vue, Vue.js is an open-source front-end JavaScript framework for building interfaces and single-page applications. It features an incrementally adpatable architecture that focuses on declarative rendering and component composition.

3.4.2 Node.js

Node.js is an open-source, cross-platform, back-end, JavaScript runtime environment that executes JavaScript code outside the browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser.

3.4.3 Express.js

Express.js also referred to as Express, is a free and open-source back-end web application framework for Node.js. It is designed for building web applications and APIs.

3.4.4 MongoDB

MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

Chapter 4

System Design

4.1 Introduction

The purpose of system analysis is to discover the business needs and requirements including functional requirements and non-functional requirements while the purpose of system design is to decide how the new system will operate.

System design consists of design activities that produce system specifications which satisfy the functional requirements that have been developed in the system analysis process. System design is basically the structural implementation of system analysis. A successful design builds on what was learned or gathered from the system analysis and leads to smooth implementation by creating a clear plan of what needs to be done.

The system design determines the overall system architecture which consists of a set of physical processing components, hardware, software, people, and the communication among them that will satisfy the system's essential requirements. The various procedure of usage of the new system is given here, i.e. how to, what to and on what the system will be used on. The importance of the design is to enable the system designer or researcher to know the cost of consequence of the product on the user and developer. In that the effectiveness of the system will not be obsolete In this section the following tools were used to describe the system: context diagram and data flow diagram, flow chart and entity relationship diagrams.

4.2 Data Flow Diagram (DFD)

Data flow diagramming is a technique that diagrams the systems processes and the data that pass among them. The main focus of data flow diagrams is the processes or activities that are performed in the system. Data flow diagrams model objects, associations, and activities by describing how data flows between and around various objects, they illustrate how data is processed by a system in terms of inputs and outputs. They are pipelines through which packets of information flow. Data flow diagrams work on the premise that for every activity there is some communication, transference or flow that can be described as a data element.

Data flow diagrams describe what activities are occuring to fulfill a business relationship or accomplish a task, not how these activities are to be performed. It shows the logical sequence of associations and activities, not physical processes.

4.2.1 Context Diagram

The first DFD in every business process model, whether a manual system or a computerized system, is the context diagram. The context diagram shows the entire system in context with its environment. The context diagram shows the overall business process as just one process (i.e., the system itself) and shows the data flows to and from external entities.

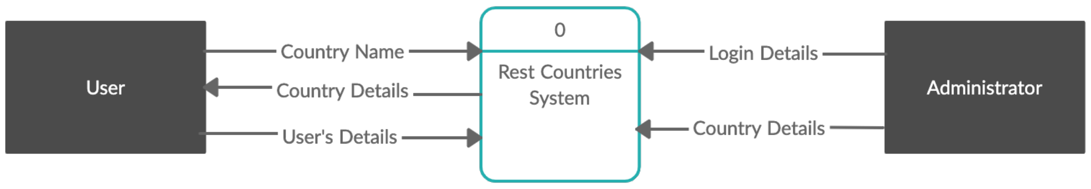

Figure 4.1: Context Flow Diagram

4.2.2 Level 1 Data FLow Diagram

Level 1 DFD is an expansion of the context diagram that shows more processes and how the users interact with the system in terms of inputs and outputs. This is the detailed description of the processes that occur in the system while related to an external entity.

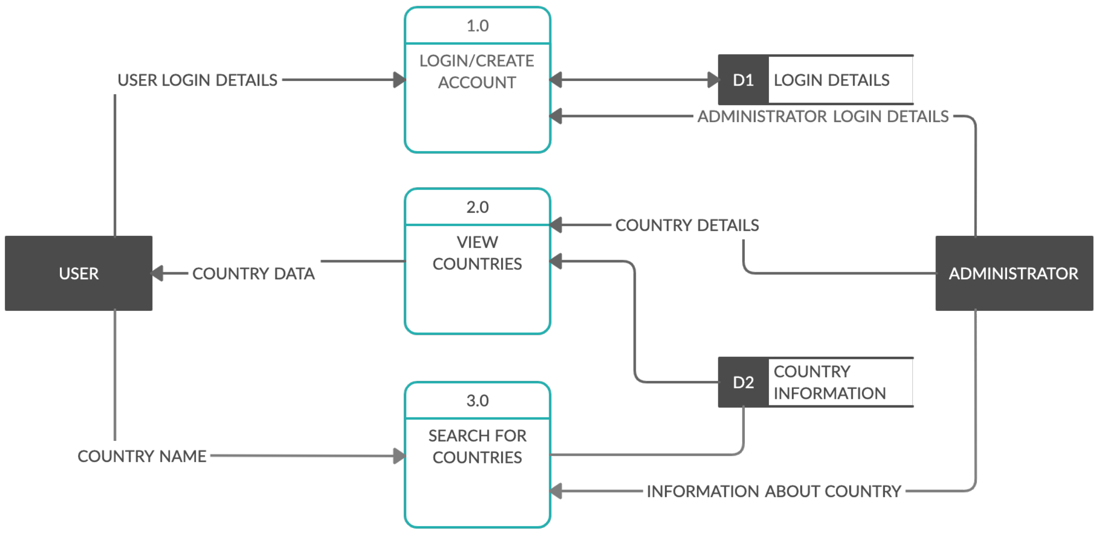

Figure 4.2: Level 1 Data Flow Diagram

4.2.3 Flow Chart

A flowchart is a type of diagram that represents a work flow or process. They are used in analysing, designing, documenting or managing a process in various fields.

Figure 4.1 shows a flowchart representing a systemic flow of how users interact with the system.

4.3 Data Modelling and Entity Relationships

4.3.1 Data Modelling

A data model is a formal way of representing the data that are used and created by a system; it illustrates people, places, or things about which information is captured and how they are related to each other.

In system design, analysts draw a physical data model to reflect how the data will be physically be stored in the databases and files.

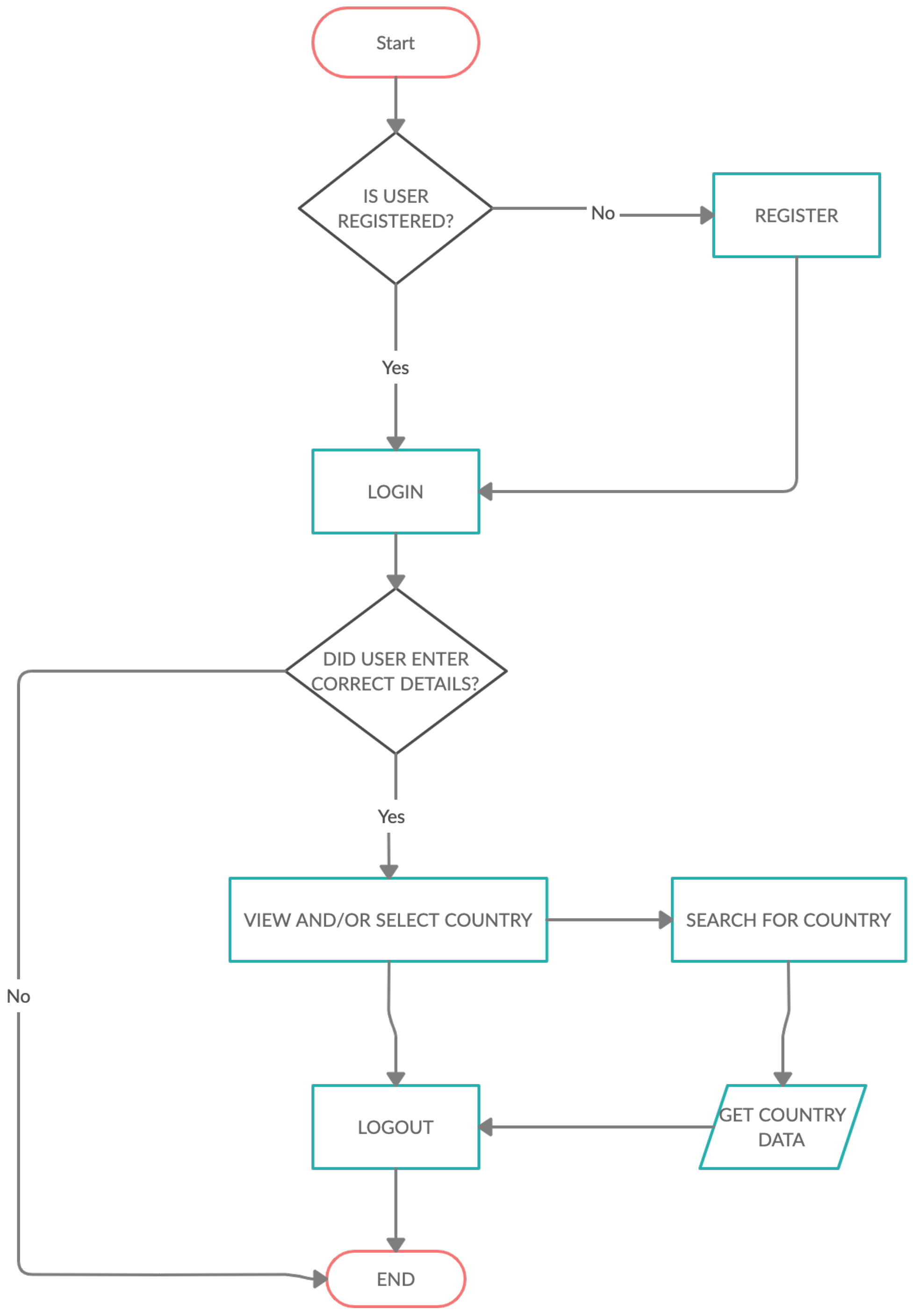

Figure 4.3: Flow Chart Diagram

4.3.2 Entity Relationship Diagram (ERD)

An entity relationship diagram is a picture which shows information that is created, stored, and used by a system. On an ERD, similar kinds of information are listed together and placed inside boxes called entities. Lines are drawn between entities to represent relationships among the data, and special symbols are added to the diagram to communicate high-level business rules that need to be supported by the system.

Entities

Entities are the basic building blocks of a data model. It is a person, place, event, or thing about which data is collected.
The entities in this application include:

- Country
- Continent
- State/Province
- Currency
- Language

Relationships

Relationships are associations between entities and they are shown by lines that connect the entities together. Figure 4.4 below show the description of the various relationships between the entities with their cardinalities.

Where:

- A Continent has many countries so it represents a One-to-Many Relationship
- A Country has many States/Provinces so it represents a One-to-Many Relationship
- A Country has only one Currency so it represents a One-to-One relationship

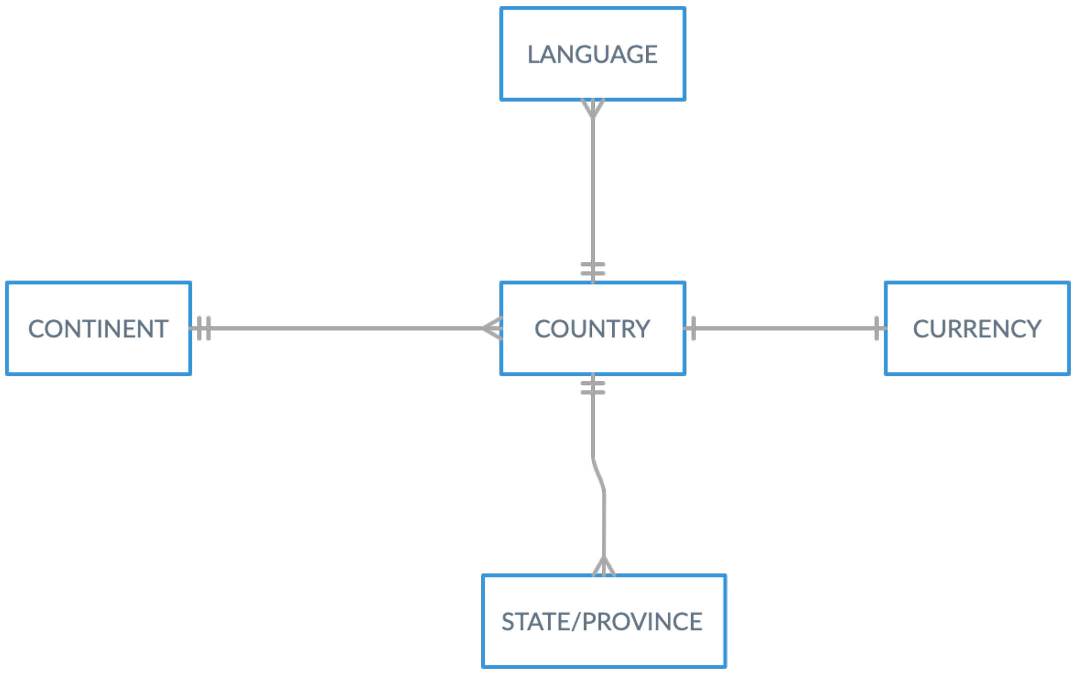

Figure 4.4: The Relationships Between the Entities

- A Country can have one or more Languages so it represents a One-to-Many relationship

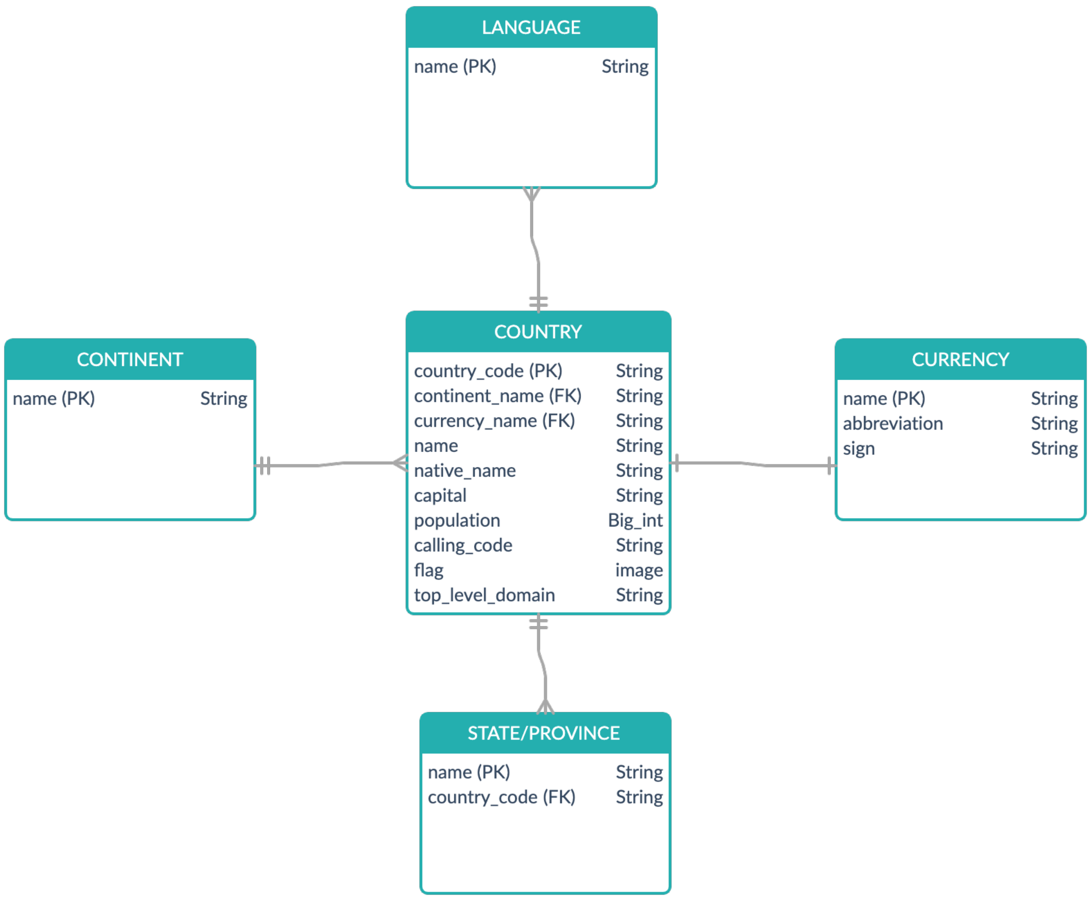

Figure 4.5: Entity Relationship Diagram

Chapter 5

Implementation

5.1 Introduction

Bibliography

Adenowo, A. A., & Adenowo, B. A. (2013). Software engineering methodologies: A review of the waterfall model and object-oriented approach. *International Journal of Scientific & Engineering Research*, *4*(7), 427–434.

Aoki, K. (2000). Taxonomy of interactivity on the web. *Esitelmä Association of Internet Researchers-konferenssissa. University of Kansas. Lawrence, Kansas.(Saatavilla: http://citeseerx.ist.psu.edu/viewdoc/download.*

Dennis, A., Wixom, B. H., & Tegarden, D. (2009). *Systems analysis and design uml version 2.0*. Wiley.

Kamari, A. (2011). Interactive hostel management system.

Petersen, K., Wohlin, C., & Baca, D. (2009). The waterfall model in large-scale development. *International Conference on Product-Focused Software Process Improvement*, 386–400.