Aruppi, Japan in one place

Xiomara Salome Arias Arias-20222020028, Member 1
Carlos Andres Celis Herrera-20222020051, Member 2
Universidad Distrital
Professor: Carlos Andrés Sierra Virgüez
Bogotá, Colombia
Date: April 10, 2024

ABSTRACT– Japanese culture has become a worldwide interest, so many of those interested want to know a little more about it, especially anime, manga and music. However, access to this type of content is often limited or has some problems, such as publicity. In contrast to this, Aruppi is an application that looks to collect all the Japanese culture in an application, that is to say, the already mentioned contents, with news that are also related to them. To implement this, a rigorous planning and documentation is required, making use of different tools and methods, such as design patterns, which in this context, the factory design will be used. This paper shows all the diagrams made to understand the operation of the backend of Aruppi, and how the data flow, activities, states and sequence between objects and classes; in addition, a database mapping is made, with the respective entities and relationships, this is based on the class diagram that synthesizes all the platform's logic. The previous, will allow a correct implementation on the application, looking for a solution to the problem raised, by using the available technologies.

I. INTRODUCTION

The world has become a space where there are a lot of cultures, with different traditions and customs. In that context, Japanese Culture has been a social interesting around the world, because that culture stands out for its rich historical depth, its vibrant traditions, and its global influence on art, music and literature.

This is due to the fusion of modernity and traditions, which are preserved over time, even on a daily basis. A clear example of their traditions is how celebrations are held throughout the year, where their origins, values and vision of life are reflected. They also highlight those places where they go to clear their minds after a long time of work, such as the hot springs of volcanic origin, located in the onsen villages, where they highlight their Ryokan inns; it is worth mentioning that these are a very old tradition that is still preserved today. As for the Japanese gastronomy, it can be said that it is a pleasure for the senses, since UNESCO declared in 2013 the washoku (traditional Japanese cuisine) as intangible cultural heritage of humanity, this has made it popular worldwide; its most outstanding dishes are sushi, ramen, yakisoba, okonomiyaki, these can be consumed in restaurants, street stalls, taverns and in the food of the Buddhist temples in Japan.

Another important aspect of Japanese culture is cinema, where interesting plots and excellent film production stand out, however, it has shown that the most demanded is the animated film, that is anime, this has been successful worldwide; an interesting fact is that all the information given above, is reflected in its cinema, which allows the world to know a little more about japanese culture, customs and point of view about the life. Finally, manga originated in Japan, what began as a sketch of scenes of daily life, became a form of art and literature, which conquered the world and have been good reception, to the point that some have their own animated adaptation.

Given the context, japonese culture catch the world's attention, however, sometimes access to this content is limited, also if the user want to see some content have to watch a lot of publicity wich will be annoying; at the same time, websites where users can watch anime or manga often lack organized content, making it difficult to keep track of what user have watched or determine which content interest to them. For that, Aruppi aims to solve this issue by providing a free platform that offers access to different contents of Japanese culture, everything in a same place, such as: anime, music, manga, news, among others interest about japanese culture.

Aruppi will offer this service through sections and functions such as: news about the latest releases and upcoming release dates, podCast and music; it will also allow you to view content, add to favorites or if you want to play it at another time. It will also allow the user to know their progress in the content watched, so when they want to continue watching it and not lose it. It should be noted that the application does not store its own content, but redirects to other pages that provide these services, such as AnimeFlv, Jikan API, Crunchyroll, SomosKudasai, Palomitron and Ramen para Dos. However, there is a database with all the content that will be provided to users.

In this project the stakeholders are principally the final users, who are a community known particularly as "otakus", who are the main consumers of Japanese cultural content. The term

otaku in Japan is used to refer to a fan of anything, while outside Japan it is used to refer to all those who have a great interest in manga and anime of Japanese origin. However, they are not the only ones to whom the application is designed for, because, a great part of teenagers and children are interested in the consume of this content, without being otaku. That is to say, the target audience of this project will be those interested in Japanese culture, taking this into account, there are features or functionalities that they want to do in the application, that is, the user stories, these are:

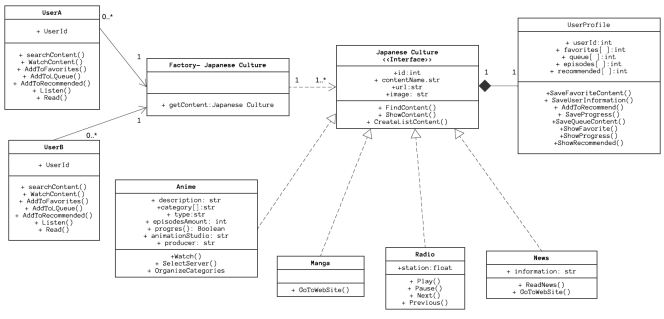
- As a user, I want to watch anime, so what I hang out.
- As a user, I want read manga, so what entertainment.
- As a user, I want listen Japanese stations, so what I learn Japanese music and know facts of their country.
- As a user, I want know the news about Japanese culture, so what I keep me informed.
- As a user, I want add my favorite content, so what I categorize them according to my preferences.
- As a user, I want see my history of episodes, so what I don't lose the thread of it.
- As a user, I want add content to the list queue, so what I can see it after.

These allow to synthesize the processes and to understand how they work, in order to perform a correct solution of the problem.

II. METHODS AND MATERIALS

The tools that will be used in the construction of this project are: Visual Studio code, a code editor through which will program with the python programming language, docker to facilitate the implementation and execution of the project, Lucidchart for the creation of the various, conceptual model of project and finally Github through which will make deliveries and relevant information will be shared with others. Finally, phpMyAdmin to create and make test in databases with SQL. First of all, it is important to understand which design pattern fits to solve the problem of this project. In this context, the factory design was used to define the construction of the Aruppi backend.

Figure 1: Class Diagram



As can be seen in Figure 2, a relationship of association is formed between the user and the factory, since the user can create objects from the factory, in this case Japanese content, because the factory is of Japanese culture. In this factory, a method is established in which the contents the user wants are simply obtained, hiding the whole process from the user; the client has methods that allow him to search, watch, listen, read and add to favorite lists, the queue or recommended.

Also, the factory has a dependency relationship with the abstract class of Japanese culture, in other words, it does not exist without the interface. As for the Japanese culture interface, it has variables and methods that must be implemented, but it is not specified how, which means that each object created can implement them as required.
Also, there are different types of Japanese culture content, such as Anime, Manga, Radio and News, these implement the interface; this means, they use the established methods in different ways, for example, the method of showing content is abstract, but when the user wants to watch Anime, it will be implemented in one way, while in News it will use another logic, and so successively. It is also evident that other variables are added to each object, which are necessary for its correct implementation. It should be clarified that all content will have ids that identify them, a content name, url that redirect to view the content, and the paths of images to display in the interface. Finally, there is a composition class called user profile, which has instances of the Japanese content objects, since it stores all the information regarding the user's preferences, in addition to the user's progress.

UML (Unified Modeling Language) is a standard modeling language used to visualize, specify, build and document software systems, which is very useful for understanding Aruppi's processes. The diagrams used will be presented below.
One of the methods used corresponds to the UML diagrams that allow the analysis of Aruupi's functionality and processes.
Activity Diagrams: is used to visualize and specify the dynamic behavior of Aruppi.In this case, there are various activities that can be performed in the application, so a diagram was created for each activity. The figures used are the circle, which represents the beginning of a process, while the end is a circle with a filler; in the same time, the arrows show the flow of the different activities or processes, where decisions or different ways to take can be found, which are joined to a rhombus.

Figure 2, shows the process flow that occurs when the user wants to watch Anime, where it can be seen that two types of search can be performed, by title or category, Aruppi shows the content found, there the user can choose the server to watch it, then it will redirect to this one.
As for the Anime activity diagram, it is really simple, as shown in Figure 3, because the user, when selecting this option, Aruppi takes him to the page of Tu Manga Online, where he can read Manga.

Figure 2: Anime Activity Diagram

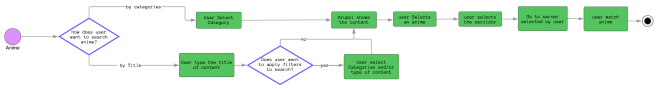
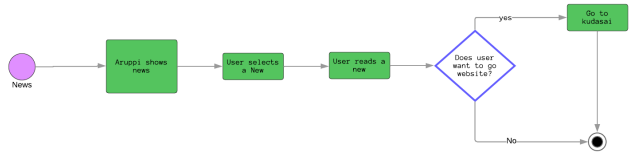


Figure 3: Manga Activity Diagram

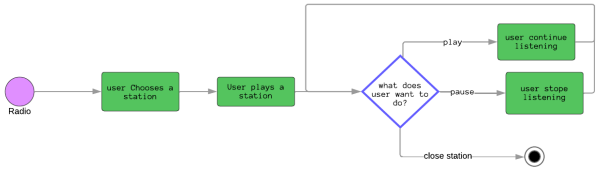
In the activity diagram shown in Figure 4, which corresponds to News, it can be seen that the process flow is given with the news available, the user selects one and reads it, there is the possibility of visiting the website of the news, so the user can perform this action if he/she wishes.

Figure 4: News Activity Diagram



On the radio side, the application offers different stations, so the user chooses one and starts listening, he/she can pause, play or close the station, then the process is finished, as shown in figure 5.

Figure 5: Radio Activity Diagram



When the user wants to add an Anime to favorites, to the queue or to recommended, the process that follows, according to figure 6, is to select an anime and select to which list to add it, Aruppi will save all the user's preferences.

The last activity diagram is the one in figure 7, which shows the user's history, where it can be seen that if the user wants to know his history, Aruppi will obtain his history and will show it to the user.

Figure 6: Preferences Activity Diagram

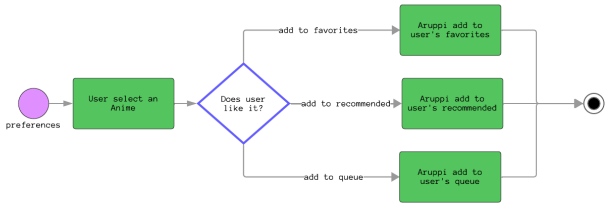
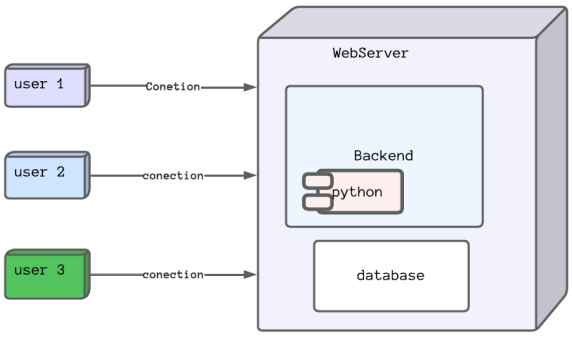


Figure 7: User History Activity Diagram

Deployment Diagram: The project will be executed locally, that is to say, in the computer where the program is installed, in Figure 8 the backend is observed in the same space as the database, this in order to contemplate a database inside the machine and not external to the computer, it is hosted and it is only possible to access from the same program located in the same machine. This in order not to use an external service. In the same way, the python component is inside the backend making allusion to the construction of this by means of this programminglanguage that today is one of the most used.

Finally, the diagram gives us an idea of the relationship between software and hardware. In this project the hardware is minimal and the execution of the software is reinforced with tools like docker.

Figure 8: Deployment Diagram

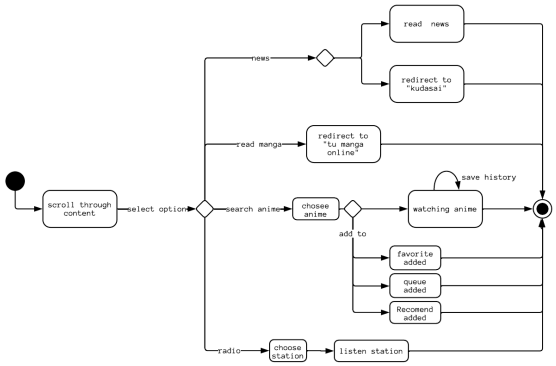


State Diagram: It is possible to observe the different states through which the "user" object is involved in the use of the project.

As shown in Figure 9, the user, after inspecting Arupi, is ready to select one of the four options. By closing the initial state, the user can start four other different events within Arupi. It is worth mentioning one of the events with

more work within Arupi and is the option of anime, since, within this there is a special event that is to add content that is divided into three states: add to favorites, queue and recommended.

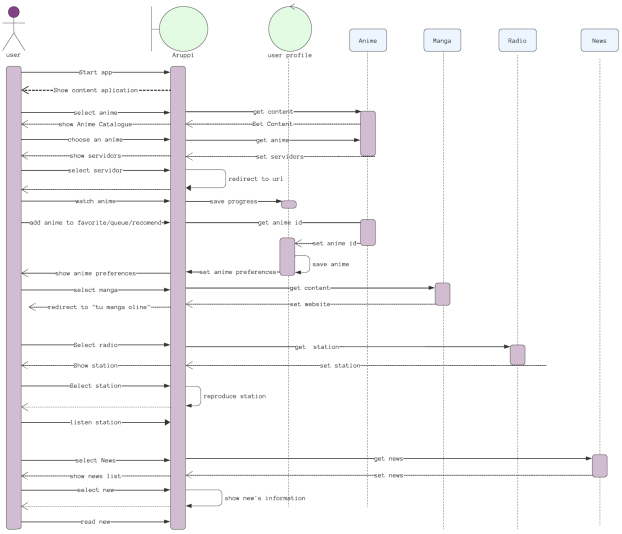
Figure 9: State Diagram



Sequence Diagram: The sequence diagram, allows to give a solution focused on the lifelines together with the objects that exist together, along with the messages exchanged between them. This allows to know the requirements and functionality that Aruppi must have.

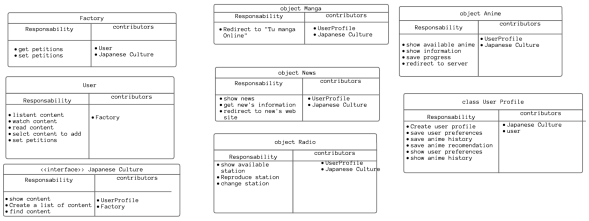
In the diagram in Figure 10, the representation of classes, objects and users is shown; in this case, Aruppi represents a boundary, because it is the interface with the user will interact; on the other hand, the user profile is represented as a control class, because it has a management of the user's information. The different activation boxes are also visible on the lifelines of each object or class, where the user will always be active while inside the application as well as the interface, while the content objects will only have an activation once the user selects an option related to it. The messages that describe the process show how the content interacts with the user, and how the user's required information will be saved.

Figure 10: Sequence Diagram



CRC Cards: In the context of Arupi we use these cards to represent the objects and classes that make up this project. Focused on the responsibilities and collaborations of each one of them. In order to have a clearer idea of the system.

Figure 11: CRC Cards



ER Diagram: Is a tool used in database design to model the relationships between the entities that compose Aruppi.

Figure 12: ER Diagram

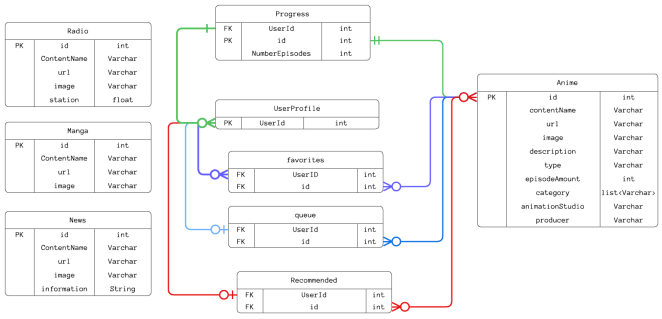


Figure 12 shows the Aruppi entity-relationship diagram, where the different tables of the database are displayed, with their respective names, fields, type of data and the primary and foreign key of each one of them is specified. In general, it is known that the primary key of the content is the id of the content. As the user has a profile, in which only its id is taken into account, together with its favorites, queue, recommended and progress, which are associated to another database, in which an association with the primary and foreign keys can be

made. For example, the progress, has a primary key that is the id of the anime, since this allows to obtain the chapters of the anime that are in the Anime database, in order to determine the state of each one, if it has been watched or not, that is to say, true and false respectively; additionally this associated to the table of the user's profile, by means of the id of the user. This is how the other bases work, the diagram shows the type of relationships they have, and those that have no relationship, but will serve to display the contents of the application.

III. EXPERIMENTS AND RESULTS

Some tests were made on the database, which was created from the Entity-Relationship diagram (Figure 12). The Xamp tool was used together with phpMyAdmin, to create a local database, together with data randomly generated with the help of an AI, in order to be able to perform tests.

The tests consist of queries, considering the user stories, Aruppi needs to perform 4 main queries on the database.

Get Name content of user's recommendation.:

Figure 13: Recommendation query

```
SELECT a.ContentName FROM userprofile u INNER JOIN recommended r ON
u.userId = r.userId INNER JOIN anime a ON r.id = a.id WHERE u.userId = '782996';
```

The SQL query (Figure 13)returns the name of the content of a specific user's recommendations.

It uses the 'userprofile', 'recommended', and 'anime' tables. First, the 'userprofile' and 'recommended' tables are joined in the 'userId' column, then the anime table is joined using the 'id' column of the 'recommended' table. Finally, filter by the 'userId'. The result is the name of the content of the recommendations for that user. In this case, the results are in Figure 14.

Figure 14: Recommendation query result

ContentName
One Piece
Demon Slayer: Kimetsu no Yaiba

Get Name content of user's favorites: The SQL favorites query (Figure 15) retrieves the 'ContentName' of the user's favorites. It joins the 'userprofile' and 'favorites' tables on the 'userId', and then joins the 'anime' table on the 'id'. Finally, it filters by the user's 'userId'. The result of the SQL query will be the 'ContentName' of the anime that are favorites of the user with the 'userId'(Figure 16).

Figure 15: Favorites query

```
SELECT a.ContentName FROM userprofile u INNER JOIN favorites r ON
u.userId = r.userId INNER JOIN anime a ON r.id = a.id WHERE u.userId = '123811';
```

Figure 16: Favorite query result

ContentName
Naruto
My Hero Academia

Get Name content of user's queue.: Figure 17 SQL query retrieves the 'ContentName' of the anime in the user's queue. It joins the 'userprofile' and 'queue' tables on the 'userId', and then joins the 'anime' table on the 'id'. Finally, it filters by the user's 'userId'.

Figure 17: Queue query

```
SELECT a.ContentName FROM userprofile u INNER JOIN queue r ON u.userId = r.userId INNER JOIN anime a ON
r.id = a.id WHERE u.userId = '262664';
```

The result is the name of the content of the anime in the user's queue (figure 18).

Figure 18: Queue query result

ContentName
One Piece
Dragon Ball Z
My Hero Academia
Death Note

Get Name content and user's episodes watched:

Figure 19: Progress query

```
SELECT a.ContentName, p.NumberEpisodes FROM userprofile u INNER JOIN progress p ON
 u.userId = p.userId INNER JOIN anime a ON p.id = a.id WHERE u.userId = '495407';
```

Figure 20: Progress query result

ContentName	NumberEpisodes
Dragon Ball Z	3
Dragon Ball Z	4
Dragon Ball Z	5
Death Note	1
Death Note	4
Attack on Titan	20
Death Note	1
Death Note	4
Attack on Titan	20

Figure 19 SQL query retrieves the ‘ContentName‘ of the anime and the number of episodes watched by the user. It joins the ‘userprofile‘ and ‘progress‘ tables on the ‘userId‘, and then joins the ‘anime‘ table on the ‘id‘. Finally, it filters by the user’s ‘userId‘.
The result is the name of the content of the anime and the number of episodes watched by the user(Figure 20).
The tests that have been realized are one of the many that must be done, but it is possible to check that what was planned is being done, by obtaining the user’s data with his preferences and progress. The information of content, and user profile will persistent default in order to apply test, it means, data that is permanently stored in a storage system, so that it can be accessed at any time.

IV. CONCLUSIONS

To conclude, these are some of the conclusions that were obtained from the design of the Arupi backend:

- Factory is the model with which it is possible to give a solution to the raised problem. Since, it is a pattern to create objects from a superclass, that allows its subclasses to alter the type of object that will be created.
- From the visual representation of the UML diagrams, an idea was given of how Aruppi should work, the requirements, structure, components and interactions, which allow it to fulfill its purpose: in turn, establish a solid foundation for the development and implementation of Aruppi.
- Arupi is scalable by design, allowing to add more products without changing the core logic of the application. This means that as the platform grows and expands to include more products and services, it continues to maintain its efficiency and performance, without compromising user experience or system stability.

V. BIBLIOGRAPHY

[1] Arisín, J. and Rodriguez, P. (2023). Japon Secreto. Available at: https://japon-secreto.com/cultura/. Accessed on: April 4, 2024.

[2] Lucidchart. (2024). State Machine Diagram Tutorial. Available at: https://www.lucidchart.com/pages/uml-state-machine-diagram. Accessed on: April 5, 2024.

[3] Lucidchart. (2024). UML Sequence Diagram Tutorial. Available at: https://www.lucidchart.com/pages/uml-sequence-diagram. Accessed on: April 5, 2024.

[4] Lucidchart. (2024). UML Class Diagram Tutorial. Available at:https://www.lucidchart.com/pages/uml-class-diagram. Accessed on: April 5, 2024.

[5] Lucidchart. (2024). UML Activity Diagram Tutorial. Available at: https://www.lucidchart.com/pages/uml-activity-diagram UML. Accessed on: April 5, 2024.

[6] Lucidchart. (2024). Deployment Diagram Tutorial. Available at: https://www.lucidchart.com/pages/uml-deployment-diagram. Accessed on: April 7, 2024.

[7] Lucidchart. (2024). What is an Entity Relationship Diagram (ERD). Available at: https://www.lucidchart.com/pages/er-diagrams. Accessed on: April 7, 2024.