WikiTime - Collaborative Timelines

Daniel Teles Arrais, 69675, MEEC
Instituto Superior Técnico, Universidade de Lisboa
E-mail: daniel.arrais@tecnico.ulisboa.pt

Abstract—The organisation and visualisation of chronological data in Web pages needs to be properly handled to provide the reader with a clear understanding of the examined topic. Despite the existence of modern chronographic tools such as web-timelines, the majority of history-related content on the Web are merely lists, or tables, in which each entry is granted just enough space to display its information. Furthermore, even though the distinct contents are usually related, these links are typically not explored and remain unused, hence limiting the user’s access to a fluid experience when studying a particular subject. Currently, some Web-based technologies for creating timelines have already been developed, but these still face some problems regarding the inability to edit or reuse previously created contents, and the non-existence of a mechanism that encourages collaboration among users. The work herein presented defines the architecture of a collaborative infrastructure, entitled WikiTime, that addresses these issues. The developed system not only allows for an easy creation and editing of timelines and its events but also grants the user with the possibility to clone any existing information. When compared with the existing methods for chronological presentation, WikiTime showed exceptional results, providing the user with a dynamic and self-consistent experience. In the context of an organisation, WikiTime also allows the integration of scattered information.

Index Terms—History, Timelines, Web Application, Collaborative Software, ReSTful API

I. INTRODUCTION

NOWADAYS, people are always so concerned about the present, and even more worried about the future, that anything from the past seems to become irrelevant and outdated. However, history has shown us that the past should not be taken lightly. To better understand the present, sometimes we should look into what has happened and comprehend the factors that have caused change. By analysing past events, we allow ourselves to anticipate what has yet to come, to understand how society will evolve through time, and what consequences should be expected. Since it was first established as an academic discipline in the year of 1870 [1], history claims to have been pursuing these same objectives, and the arrival of the World Wide Web (WWW) has opened its doors to those who never had access to it before. In a few steps, historians, or simply history enthusiasts, can share their perspectives and experiences with practically anyone, using for example blogs, forums, or a personal Web page [2].

Despite being considered a relatively recent technology, the WWW is now a huge source of information, one that has challenged historians to rethink not only how to perform research, but mostly how to present and teach about history. Over the years, researchers have been putting to practice the multiple advantages provided by the Web, and history has never been so strongly disseminated as it is today. Nonetheless, the majority of the presented content on the Web still lacks the design and functionality required to provide the readers with a richer and more fluid experience, mainly regarding aspects such as interaction, visualisation, and exploration. The chronological data is usually displayed as just lists or tables, surrounded by blocks of text, making it difficult for Internet users to get the information they are looking for and to feel really involved in the subject. Additionally, the multiple current data is usually connected and only at a later stage are these links found, exposed, and used, which defeats one of the purposes of studying history, that is, to not only look for when and where an event has happened, but also which other events are related to it.

The proper visualisation of data is of utmost importance, and an appropriate display tool is one that facilitates the readers in understanding the presented contents, communicating information to the users clearly and efficiently, while also simplifying the process of relating content and recognising patterns. Chronology, in its most basic definition, is the arrangement of a series of events over a definite time period, and time itself can be visualised in multiple different ways. Even so, the idea of portraying time as a line is so embedded into peoples’ conceptions and notions that using the timeline for representing historical data prevails as the correct fitting. Stephen Davis supports this same view and states that: “a graphical timeline is assumed to be more informative than a list of dates, however poorly it is designed” [3]. Digital timelines offer a large deal of interactivity and engagement, supporting visually attractive displays of information, and allowing users to move quickly through time, to efficiently analyse and compare different historical periods, and to easily change between closer or wider perspectives.

A. Problem Statement

The areas of study in which timeline tools can be applied are countless, and although some advances have been made in creating software that can transform chronological data into vibrant displays of properly organised information, these are still limited and do not allow to interpret the presented historical contents correctly. Furthermore, some of the features that we consider to be truly necessary are also lacking, namely the existence of a mechanism that promotes collaboration among users, or the possibility to reuse or embed any of the previously generated timelines and events.

With the constant evolution of digital technologies, it would be expected, at this moment, that a system capable of suppressing the referred problems would already be available. However, considering that to perform such task a sizable amount of
programming knowledge is required, the development of these applications is hindered by another issue: interest usually lies within people with professional backgrounds in areas of study unrelated to software development, incapable of designing such product. History needs to be seen as a field of great value, from which lessons can be taken to help us comprehend the present and to be more aware of the future. It is absolutely necessary that we take the time to think about how we can use and extend the existing systems to create a tool worthy of a historical presentation, a tool that serves the purposes of history best.

B. Objectives and Results

The primary objective of this work is to define and implement a Web-based collaborative infrastructure for the easy creation and editing of timelines. The developed software application, entitled WikiTime, should have a visually appealing graphical interface that provides users with a dynamic and interactive experience, and should also contain a simple mechanism supporting the production and editing of contents on a collaborative basis.

The design and development of a Web Service Application Programming Interface (API), which allows the full complete integration of WikiTime’s services with other external systems, is also in our agenda. Having a robust, scalable services interface which allows resources to be consumed by other developers, is a step towards fostering the creation of an ecosystem around the designed application, thus making it more desirable and valuable.

The development of such application led to the creation of more than 110 events distributed among 17 timelines, depicting about 100 years of history regarding Instituto Superior Técnico. Our primary goal was to gather all the data included in the websites created by Instituto Superior Técnico [4]–[7], combine it with additional sources of information, and provide the readers with an interactive and dynamic experience, in which every relation is shown to allow for a fluid, more complete study on each topic.

C. Document Structure

The work herein presented is organised into six sections. The next section ("Timelines") will start by providing an historical background for the development of timelines, followed by an analysis of the currently used methods for the creation of such displays. In Section "Data Design", a few use cases are defined to understand the design requirements and specifications that the application must meet, which will mainly serve as the basis to construct both the supporting data model and the software platform. Section "Implementation" describes the implementation details concerning the envisaged approach. In Section "Evaluation", we will evaluate the performance of the developed solution by performing a comparison with the existing methods for visualising and organising historical data in the Web. Finally, the Section "Conclusions" will conclude the document by addressing the most significant contributions of the proposed system.

II. Cartographies of Time

The conceptual visualisation of time-related data is not an easy exercise. The idea of time is highly subjective and usually difficult to define since it can contain distinctive meanings depending on peoples’ perspectives, beliefs, and feelings. The idea behind timelines is to provide an intuitive and comprehensive method for temporal data display, allowing to visualise how multiple different events are organised in time and what the relationships between them are. Essentially, timelines are an efficient way of telling a (chronological) story.

A. The History of Timelines

Even though chronology has been identified as a field of study since the ancient times, it was only in the middle of the eighteenth-century that a common vocabulary for time-maps was introduced, capable of displaying temporal data in a linear and continuous format [8]. Until then, the visual representations of time always followed the traditional tabular structure, in which the information about history was shrunk to fill in the least amount of space as possible. But in 1753, Jacques Barbeu-Dubourg, driven by the belief that visual representations of time could and should be designed using a uniform scale, published his Chronography or Depiction of Time, a 16.5 metres chart depicting history since the creation until his own days. However, considering the substantial length of the diagram, the posed challenge to display all at once impelled the production of an apparatus, entitled Chronographie Universelle (Fig. 1), in which the paper could be mounted and viewed one section a time (roughly 150 years). This invention allowed the reader to scroll through a continuous timeline back and forth by using the small crank handles.

Besides Barbeu-Dubourg’s publication, another work of serious relevance was A Chart of Biography, developed by Joseph Priestley in 1765. While teaching history at the Warrington Dissenting Academy, Priestley needed to make sense of the lives of various historical figures, which led him to design a diagram that was long enough to record the lifespan of about two thousand individuals (Fig. 2). Even though he cannot be considered the one responsible for the creation of the timeline, what he added to Barbeu-Dubourg’s map was the unique concept of using a printed line to represent the lifespan of each individual [10]. Since different amounts of time could be described by using a longer or a shorter line, the readers
were now able to obtain clear and precise information about every individual, instead of just a rough estimation.

Fig. 2. Joseph Priestley’s A Chart of Biography, from 1765 [11]

The eighteenth century was an era of extreme importance regarding the development of instruments for temporal visualisation. But ever since Barbeu-Dubourg’s creation of the timeline, followed by Priestley’s unprecedented improvement, that chronology has not received any serious attention and remained essentially in the domain of those who do not have the required knowledge that would allow further developments [12]. Despite the evolution of modern technologies, current historians have been more preoccupied in recovering the lost history of chronographics rather than in developing new systems for history visualisation.

B. The Emergence of Modern Chronographics

When applied to historical data, a timeline’s primary purpose is to enable users in obtaining both a general, and a detailed perspective of events, and patterns. Nonetheless, the study of history is not only about examining the existence of events in the past but also about identifying and studying the relations of cause and effect that determine these. History is best understood as a series of interconnected facts and competing narratives that do not mean much when in isolation [13]. Once we accept that historical relations are an important part of our comprehension of the world’s past, present, and future, we should take into better consideration any device that facilitates the discernment of these associations.

When both Barbeu-Dubourg and Priestly published their charts, it was already evident that they had profound knowledge about the importance of relationships in the representation of history. Even so, due to the lack of technologies powerful enough for representing the historical relations accurately, these timelines did not allow readers to understand efficiently if and how the distinct subjects were related to one another. With the evolution of digital media, and specifically of the Web technologies, it would be expected that a solution for this problem would already exist, either by using hyperlinks or by employing a more dynamic and interactive form of presentation. However, like in the ancient times, the majority of new chronologies, limited by the early versions of Hypertext Markup Language (HTML), are really still lists or tables [12].

The timeline still occupies a position of little relevance and is rarely given any serious thought. Our comprehension of these tools is still weak, and it is only when all other alternatives have already been used that we think of them as a necessary instrument for the graphic instantiation of history [13].

III. WEB-BASED TIMELINE SOFTWARE

In recent years, new methods of aggregating and integrating chronological data from multiple sources have been suggested. Methods that elevate the timeline’s purpose and consider it as the central organising structure for the graphical display of historical data. By working on the principles of collaboration, and by taking advantage of recent technologies, new modern tools may be able to achieve what the time charts of the past could not: gather and display large amounts of data while presenting all the different existing relationships inherent to the exhibited information.

A. TimelineJS

TimelineJS [14] is an open-source, Web-based tool for creating and visualising attractive and dynamic timelines. Released in March of 2012, it is currently in its third version, being widely used mostly by news organisations from all around the world. Among the many features it includes, the following ones should be highlighted:

- Possibility to incorporate media of different types;
- Supports the customisation of the displaying elements;
- Possibility to group events into a similar category;
- Events can be punctual or spread across a span of time.

Users can generate the desired timeline objects either by using the authoring tool available at TimelineJS’ website, which accepts a properly formatted Google spreadsheet as input, or by introducing contents in JavaScript Object Notation (JSON) format. Although the first described method may be sufficient to generate singular and independent timelines, it is still considered to be a relatively primitive technique if one needs to create multiple distinct timelines, interconnected around the same particular topic. In this case, there is the possibility to download and modify the source code supporting the application, which enables programmers to design custom installations and to mould the software’s functionalities and characteristics into their requirements and goals.

B. Sutori

Defined as a “free digital learning tool” [15], the principal purpose of Sutori is to offer an instrument, devised mainly for the classroom, which enables teachers and students to create interactive stories using an intuitive collaborative interface. Each timeline can be customised with a description and a header illustration, and different types of event objects can be generated, containing either an image, video, audio, or even a quiz question. Nonetheless, upon further observation, we noted that the existing free package is rather limited since it only allows the creation of at most 200 stories and offers no possibility for collaborative work. Users are required to pay
for either the premium package (supports countless stories and collaboration in pairs) or the unlimited package, in order to be able to enjoy such features.

C. Neatline

Defined as a "light-weight, extensible framework for creating interactive editions of visual materials" [16], Neatline is a project of Scholars’ Lab, a team of experienced students and researchers, from the University of Virginia Library, who mainly focus on developing solutions in the fields of digital humanities and geospatial information. Onto the core Omeka’s feature set, Neatline adds new functionalities that allow not only to create beautiful and interactive visual objects quickly but also to explicitly represent geospatial information as a collection of records plotted in both time and space. Nonetheless, even though current versions of Neatline are able to provide the methods for representing any contents that require some visual display, it is still too much focused mostly on maps, putting aside the aspects related to the temporal representation of information.

For establishing and using Neatline, one must first install the technological stack supporting Omeka, which is composed of the famous but rather old-fashioned software bundle entitled LAMP (Linux, Apache, MySQL, and PHP). After having installed the LAMP stack server, the Neatline plugin must be added and configured. Additionally, it may be necessary to download and incorporate any extra libraries, in order to take full advantage of Neatline’s features and functionalities.

D. Tiki-Toki

Tiki-Toki [17] is a Web-based application for creating interactive and dynamic timelines which can be shared with anyone on the Internet. It was set up by a company called Webalon which launched the product in the year 2011. Similar to Sutori in practically every aspect, it also makes use of Web-based forms to allow the easy creation and editing of the contents, while supporting the possibility to add data of multiple different types, namely images, text, and videos (Youtube or Vimeo).

In addition to accommodating the visualisation of timelines in the traditional horizontal format, Tiki-Toki’s application also permits to display the contents in a 3D display. No other software platform provides this feature, which combines both the observational and immersional modes that were so dearly referred by Priestley in the eighteenth century. The problem with this application however, is that, unless users are willing to pay, only one timeline can be created using a free account, with no possibility to perform any collaborative work. Furthermore, even the paid versions still contain a rather small limit on the number of timelines allowed to produce.

E. Overview of Modern Technologies

After searching for some of the existing solutions, we came to the conclusion that, despite the existence of some particular differences, most of the Web-based applications for creating timelines share a lot of common features. Even so, for us to make a more correct and fitting decision, the described applications should be evaluated based on a set of specific parameters:

1) Is the software open-source?
2) How good is the documentation?
3) How is the community support?
4) How easy it is to integrate the software with other technologies?

Considering the first aspect of evaluation, both Sutori and Tiki-Toki can be immediately discarded. These systems were developed with the intention of functioning mainly as standalone applications, hence offering no possibility to download and reuse the source code supporting the software. On the other hand, TimelineJS and Neatline are classified as libraries, or frameworks, created using Web-based technologies to allow the simple integration of these technologies into larger systems and, consequently, develop more complex software products.

Since Neatline was designed as a built-in plugin for Omeka, the decision of using this library would also imply the development of our Web application using the technological stack employed by the referred publishing framework. As such, we feel that this obligation would limit the potential of developing a more attractive and dynamic software since, at some point, we would be pushed towards using a set of technologies with which we are not comfortable. Moreover, besides the fact that the official documentation lacks the fundamental information required to understand how to use this library efficiently, the existing online community still counts with a small number of supporters. In the event of any serious obstacle, we would be entirely on our own when trying to solve the problem.

Despite the existence of other Web-based timeline software tools, such as Timeglider [18], SIMILE Timeline [19], or Timesheet.js [20], these were still rather immature, requiring the design of more appealing visual displays and the implementation of more powerful features. In comparison with the numerous Web-based libraries for creating timelines, TimelineJS proved to be the one with the most potential to be utilised and modified according to our application’s requirements. In fact, when evaluating this tool, we verified that there are not any significant downsides associated with the library: it contains an impressive set of features, displays an attractive and interactive interface, and allows the simple integration with other components.

IV. Requirements Analysis

There is more to software engineering than just writing the required code to make an application work as expected. Logical thinking and design also take part in this long-lasting process that usually begins with the analysis of the project’s requirements and functionalities which then give order to the conception of the data structure that serves as the basis for sustained incremental development. It is at this stage that engineers decide on the key system’s specifications and functionalities, which should be clear, consistent, measurable, and testable, providing a sufficient level of detail for the system to be efficiently designed, implemented, and improved.
A. Product Functions

When entering the home page of the developed Web application, the users will be faced with a list of the timelines stored in the system. Moreover, besides being able view the displayed timelines, the most basic operation one will be able to perform is registering an account and, after completing the registration process, log in. Authentication should be possible either via a local account (username and password) or via integration with a Google’s account.

1) Events:
Table I contains a description of all the required functionalities to be implemented within our software application regarding events, that are essentially related to creating and updating an object structure composed of a set of properties regarding their information (headline and description), the associated media file, and the starting and ending dates (identified by year, month, day, hour, and minute).

<table>
<thead>
<tr>
<th>Functionality</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a new event’s object</td>
<td>Create a copy of the event’s object</td>
<td>Associate a media file with the event</td>
<td>Modify the event’s headline and description</td>
<td>Update the event’s starting and ending date</td>
<td>Change the event’s domain</td>
<td>Delete the event</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Events should be either punctual or spread across a span of time. While the starting date should always be required, it should be up to the user’s choice to decide if the ending date is defined or not. Moreover, the user will be entitled to classify its domain as either public or private: a public event means that it should be visualised by anyone accessing the application, while a private one should only be exposed to the users responsible for it.

2) Timelines:
Editing a particular timeline means editing the events contained in it. Even so, it should be possible to modify certain aspects of a timeline that are unrelated to the objects it contains, such as its headline and description, its media, or its domain. Furthermore, access to the timeline should only be allowed upon authorisation through the specification of an access list. Table II presents an overview of the functionalities that are required to be implemented regarding timelines’ objects.

Table II

<table>
<thead>
<tr>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>19</td>
</tr>
<tr>
<td>20</td>
</tr>
</tbody>
</table>

B. User Characteristics

Depending on the context in which they are described, there are several types of users that will interact with the system. In a first instance, it is possible to identify three principal types: administrators, registered users of the application, and guests. After a user performing the registration process, the account will still need to be evaluated and accepted by an administrator. During this time, the user will be considered as inactive and, therefore, he will still be rather limited in the number of actions allowed to execute. This type of users is then included in the same category as guests, since the kind of interaction that they will be able to perform regarding timelines or events will be more or less the same.

1) Administrators:
Administrators are the type of users who are completely free of any constraints, hence being able to access all aspects of the application and carry out all kinds of operations (1-8, and 9-20, from Tables I and II, respectively). In addition to all the described functionalities regarding the creation, editing, and deletion of timelines or events, administrators will be entitled to execute actions related to the management of all the other users of the application. These operations include modifying the status (inactive or active) and the role (normal or administrator) of a user or deleting a user’s account. In general, administrators are required to manage the entire application, ensuring that there is no incorrect or prejudicial information within it and that registered users are behaving within the bounds of civility.

2) Registered Users:
Registered users are those who have an account which was already accepted and marked as active. Primarily, this type of users will have the authority to create and edit multiple timelines and events (1 and 9), as well as to visualise and reuse other already created objects (2, 3, 10, and 11). However, the latter two operations will always be susceptible to if a particular timeline or event is classified as either public or private: if marked as private, only users included in the object’s access list will be entitled to view and reutilize it. Public timelines and events, however, will be exposed to everyone and will be totally available for reuse.
The one who decides to create a timeline’s object is classified as its owner, responsible for inviting the contributors to collaborate on the creation and editing of contents. In addition to the typical operations (12-16), any modification related to more sensitive information shall be the responsibility of the owner, as well as any operations about deleting a timeline or managing its access list (17-20). Contributors will then only be entitled to add or remove events and sub-timelines, create or delete tags, modify the media file, or group multiple events and sub-timelines into a specified category (12-16). Regarding events, if these belong to a particular timeline, there will be no differentiation between the type of users that are included in the access list, i.e., both the owner and the contributors of a timeline will be entitled to perform the same kind of operations on the events it contains (4-8).

3) Guests:
Every user who accesses the application and does not have a registered account is included in the guests’ category. Besides being able to visualise the current timelines and events (1 and 9), and to undergo the registration process, there is nothing more that they will be able to perform. Even the visualisation of timelines and events will be limited according to either the particular object’s domain value is defined as public or private: only public timelines will be exposed to guest users. On the other hand, registered inactive users will be able to visualise private timelines and events (1 and 9), as long as they are included in the defined access list, but will not be able to perform any editing or management of the displayed contents.

V. DATA DESIGN
Up until this point, we have been planning and analysing all the fundamental requirements that will guide us during the development process of our software application. Transforming some of these specifications into distinct data entities that compose the database system supporting the project’s logical and physical structure represents the next step in the software development cycle. Data design is about identifying the type of information that requires being persistently stored, determining the relationships between the different elements and, ultimately, organise and model this information into a diagram representing all the data structures and processes [21].

A. WikiTime’s Data Model
A data model is a conceptual, visual representation of the data structures, including the distinct elements and the several associations between them, that compose the database supporting the application. The primary focus when designing a data model is to represent the information in such way that it should be easily understood by anyone with knowledge of these concepts, serving as a bridge between the real-world components and their physical representation in a database [22]. The process of designing the software’s data model is comprised of four major steps:

1) Identify Entity Types - an entity is typically referred to as a collection of objects that share common features and aspects;
2) Identify Attributes and Keys - each entity type may contain one or more attributes, which are commonly used to describe and define the referred entities;
3) Identity Relationships - the relationships purpose is to describe the existing associations between distinct entities. These are typically classified concerning their cardinality (may be either One-to-N or N-to-N);
4) Apply Data Model Techniques - upon performing all the previous steps, some patterns can be applied to the identified elements.

By following these guidelines, and with the help of a Unified Modeling Language (UML) class diagram [23], we were able to design the data model supporting WikiTime’s software application (Fig. 3). The first aspect that needs to be highlighted is the definition of the three principal data entities: user, event, and timeline. Each of these entities contains a different set of attributes related to the information that is required to be stored, according to the specifications and requirements previously defined. As an example, both the events and timelines contain fields that are related to their visibility, or to their starting and ending dates. The user entity, on the other hand, allows storing information about any registered client (e.g. full name, email, etc.).

Regarding the associations, the cardinality, as well as the role that each entity represents in a relationship, is displayed next to the corresponding element. With the exception of the users’ entity, both timelines and events are entities that cannot be described on their own, as these must always be associated with one or more users.

![WikiTime's UML data model](image)

VI. SOFTWARE DEVELOPMENT
The process of creating a Web application can be quite demanding and exhausting. With such a large number of frameworks and tools to choose from, and many more being launched practically every week, narrowing down our options
can be a frustrating process. A careful and thought selection can make a significant impact on the success and efficiency of a software project. The key question is not how to choose the best options, but instead how to choose those that are most appropriate for the application’s purpose and goals [24].

A. WikiTime’s Technological Stack

When deciding on the technological stack that was going to be employed for the development of our software project, we mainly focused on technologies that would allow us to meet the predefined requirements and specifications. Above all, our desire was to create a robust, maintainable, and stable application that could provide the users with a rich, interactive, and engaging experience. The MEAN stack is a set of technologies that have been found to work remarkably well when used together, and is comprised of the following frameworks:

1) **MongoDB** – is an open-source document-oriented database management system that provides high performance and availability while supporting a rich query language syntax for executing all the required operations [25];
2) **Express** – is a light-weight, open-source JavaScript (JS) framework for Node.js, designed to provide a set of common procedures and methods for developing Web applications [26];
3) **AngularJS** – is an open-source Web application framework, developed by Google, aimed to simplify the development of client-side applications capable of providing an interactive and dynamic experience [27].
4) **Node.js** – is a light-weight, efficient server runtime environment, built on top of Chrome’s V8 JS engine, for designing and implementing scalable Web applications [28].

The MEAN software bundle is a full-stack collection of Web-based JS technologies that helps to simplify and accelerate the process of creating a Web application [29]. By coding with JS throughout both the client and the server’s application, developers are entitled to obtain clear performance gains in both the software itself and the productivity of development. Furthermore, given the fact that documents are stored in JSON-like format, using a set of technologies that are JS-based allows the data to be transferred seamlessly from the database all over to the client without having to proceed to its serialization or deserialization.

B. WikiTime’s Architecture

Fig. 4 represents the general architecture of WikiTime’s Web application. As expected, it consists of the three distinct components that compose the 3-Tier Architectural Model [30]. The Web browser, responsible for displaying the user interface and managing all of its interaction, was developed with the help of three technologies, commonly used to develop the client-side application: HTML, Cascading Style Sheets (CSS), and JavaScript (JS). While HTML and CSS provide, respectively, the structure and styling of the Web pages, JS is accountable for adding the interactivity. In this type of system, the rendering of the Web pages that compose the user interface occurs only within the browser.

![WikiTime’s Architecture](image)

Upon each user interaction, the client’s application performs a HyperText Transfer Protocol (HTTP) request to the server, using the server’s exposed API. There are four distinct types of components included within the Web server, each having its own specific purpose and being accountable for different tasks:

- **Operational Management** - includes the components that perform input data validation, error handling, and email messaging;
- **User Management** - as the name suggests, covers the components that are responsible for authenticating users and controlling accesses to data and executable functionalities. Operations concerning the administration of users’ accounts are also included in this category;
- **Timelines/Events Management** - includes the components that are accountable for the creation and editing of timelines/events, as well as for the automatic construction of the inherent relationships between the multiple existing objects;
- **Data Access** - includes the components that allow communicating with the database server.

By using this type of architecture, it was possible to implement the client-side application to be completely decoupled from the server-side. The client only needs to have knowledge about the routes designed at the server, and which type of data these return. This decoupling allows software engineers to implement a server-side script that functions mainly as a Web Service API, meaning that it can be consumed not only by the developed client-side application but also from other external systems [31], hence increasing the overall flexibility and manageability of the platform.

C. WikiTime’s ReSTful API

When used in the context of Web software development, an API is typically defined as a collection of rules and specifications that describe how multiple different systems can access a specific Web-based software application [32]. By working on the principles of information hiding, Web Service APIs allow developers to encapsulate the implementation details of the server’s application: while consuming its resources, developers only need to know how and which routes they can use to access the desired information and services.

As we designed and developed our Web application, we tried to make sure that the server-side program, functioning mostly as a Web Service API, was versatile enough to serve
both the client-side code as well as other external applications that look to integrate the implemented services. The developed API platform, organised according to the principles of a ReSTful architectural style [33], contains a total of 78 different routes that provide programmatic access to read and write the application’s data. Accordingly, the response given by the server is structured in JSON format.

In Table III are displayed only the most relevant endpoints, which allow the proper and substantial use of the system’s functionalities. To correctly execute any of the desired operations, it is necessary to prepend the base Uniform Resource Locator (URL) of the server, which is given by: ~ / api /. Another important aspect that can be withdrawn from this is the presence of URL parameters, which can be either : timeline_id, : event_id, or : user_id, that allow us to obtain specific resources by specifying its primary key as an identifier. These are automatically set by MongoDB when storing the document in the database and are composed of a 12-byte hexadecimal string.

<table>
<thead>
<tr>
<th>HTTP Verb</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>~ / users / all</td>
</tr>
<tr>
<td>GET</td>
<td>~ / users / user_id</td>
</tr>
<tr>
<td>GET</td>
<td>~ / users / user_id / events</td>
</tr>
<tr>
<td>GET</td>
<td>~ / users / user_id / timelines</td>
</tr>
<tr>
<td>POST</td>
<td>~ / events / new</td>
</tr>
<tr>
<td>GET</td>
<td>~ / events / all</td>
</tr>
<tr>
<td>GET</td>
<td>~ / events / event_id</td>
</tr>
<tr>
<td>POST</td>
<td>~ / events / event_id / reuse</td>
</tr>
<tr>
<td>DELETE</td>
<td>~ / events / event_id / delete</td>
</tr>
<tr>
<td>GET</td>
<td>~ / events / event_id / collaborators</td>
</tr>
<tr>
<td>GET</td>
<td>~ / events / event_id / timeline</td>
</tr>
<tr>
<td>GET</td>
<td>~ / events / event_id / related</td>
</tr>
<tr>
<td>POST</td>
<td>~ / timelines / new</td>
</tr>
<tr>
<td>GET</td>
<td>~ / timelines / all</td>
</tr>
<tr>
<td>GET</td>
<td>~ / timelines / timeline_id</td>
</tr>
<tr>
<td>POST</td>
<td>~ / timelines / timeline_id / delete</td>
</tr>
<tr>
<td>DELETE</td>
<td>~ / timelines / timeline_id / delete</td>
</tr>
<tr>
<td>GET</td>
<td>~ / timelines / timeline_id / collaborators</td>
</tr>
<tr>
<td>GET</td>
<td>~ / timelines / timeline_id / events</td>
</tr>
<tr>
<td>PUT</td>
<td>~ / timelines / timeline_id / events / event_id / add</td>
</tr>
<tr>
<td>PUT</td>
<td>~ / timelines / timeline_id / events / event_id / remove</td>
</tr>
<tr>
<td>GET</td>
<td>~ / timelines / timeline_id / related</td>
</tr>
</tbody>
</table>

TABLE III
OVERVIEW OF THE IMPLEMENTED API ENDPOINTS

VII. SOLUTION’S EVALUATION

Anyone accessing WikiTime’s application (available at http://146.193.41.162.nip.io/wikitime/), regardless of having an account or not, is entitled to visualise any of the already created timelines and events, as long as these are classified as public. If not, only the users included in the object’s access list (and administrators) are allowed to perform such action. To demonstrate the uniqueness of the concept behind the developed system, 17 timelines and 114 events were created, mainly regarding the history of Instituto Superior Técnico and three famous personalities that were actively involved in the creation and evolution of the institution. However, we must first discuss about the traditional methods for organising and displaying historical data on the Web.

A. Traditional Methods

Alfredo Bensaude was nominated the first principal of Instituto Superior Técnico, for which he fought hard to be able to provide better conditions for its students. Despite being extremely interactive, the existing website which concerns the history of Alfredo Bensaude [3] is troublesome when one desires to comprehend how the different aspects of Alfredo Bensaude’s life (for example regarding his scientific activity, his personal life, or his education) are overlapping and interconnecting with each other. Since the multiple events are organised by different themes, it is not clear how these are placed in time and, at some point, it seems that the multiple parts of Alfredo Bensaude’s history are entirely independent of each other and occurred at different moments. On the other hand, considering that various types of textual and media documents are presented in a format similar to a slideshow, one interesting feature that we withdraw from this website is related to how the data is displayed to the user, through the usage of several simple dynamic elements.

Known for having a great futuristic vision, Duarte Pacheco is the one responsible for the translation of the Instituto Superior Técnico’s facilities to its present location, hence giving shape to the ideas previously conceived by Alfredo Bensaude. Nonetheless, the only Web page that contains information about Duarte Pacheco [5] presents the same problems that were previously discussed: the displayed information lacks the design and functionality required to provide the readers with a more engaging and fluid experience. In this case, data is presented as a simple big block of text, having no component of interactive and dynamic nature. Additionally, the written contents are too focused on Duarte Pacheco’s life as a politician, rather than on the aspects of his life as the principal of Instituto Superior Técnico.

As a way to honour the type of person he was and his achievements, the people from Instituto Superior Técnico, together with the department responsible for the gathering and preservation of historical contents regarding the referred Institute (Núcleo de Arquivo), developed a website containing some information about Abreu Faro [6]. Nevertheless, despite the fact that the presented Web pages contain different types of textual and visual elements, these are poorly organised and do not follow a logical scheme. There are lots of pictures and documents, but little to no information to give context to this elements. Overall, the Web site itself does not include a conducting wire that allows the reader to understand the information to be transmitted.

It is important to remember that all these three personalities have a common background related to Instituto Superior Técnico but in no way these associations are displayed anywhere.
clear and complete information about each individual, it is essential that we describe and define the relationships between the multiple existing entities. The digital technologies already have a series of features that had not been used efficiently in current representations of history, and further thought had to be given about it. Ultimately, the idea that was always present in our mind was that the reader must be able to travel through history as if he was reliving his own personal experience, even in moments that he did not experience in the first place.

B. WikiTime’s Method

When we first thought about developing such application, our main goal was to preserve all the beneficial aspects that the traditional methods have while working on the improvement of the defective ones. TimelineJS itself already allowed, for a single entity, the presentation of historical data in a proper format, enabling the use of several different types of textual and media documents and the organisation of the multiple existing events in chronological order. The biggest problem, however, lied within the impossibility to associate any related timelines, or events, with one another.

The history of an organisation is mainly composed of the people that belonged to it, and by knowing about the life of each individual, we are also allowing ourselves to comprehend the impacts better one had to a particular institution. To support and demonstrate the validity of this concept, we have begun by designing the timeline concerning the history of Instituto Superior Técnico, in which were included three other timelines that also take part in the ones related to Alfredo Bensaude, Duarte Pacheco, and Abreu Faro. As such, once a user starts exploring the timeline "Instituto Superior Técnico", he will surely be able to move to the main timelines of the referred characters, as these also contain some sub-timelines that are included in the timeline of the institute’s history. It becomes clear that an underground network of inter-related timelines and sub-timelines exists, and the methods for access and travel along its channels are the following two:

1) Fig. 5 is a screenshot of the timeline "Instituto Superior Técnico". The first aspect that we would like to highlight is related to the existence of two types of elements shown in the slider: the events, marked with an "E", and the timelines, referred to with a "T". Regarding the latest, if an element contains such mark, this means that the slide’s headline is actually a link that, upon being clicked, will trigger the application to redirect the user to that particular timeline, as shown in Fig. 6;

2) The other technique is represented in Fig. 7, which consists of displaying a list of all the timelines that are associated with the one the user is currently visualising. This list can be obtained by clicking on the "Related" button that is placed on the right bottom corner of the interface. One must keep in mind that every slide included in the slideshow will display a different list: while for events, this list is limited to all the timelines that possess a copy of the event, for sub-timelines, the list includes all parent and children timelines as well as other that have equal tag values;

By implementing the described techniques, we allow users to move from timeline to timeline almost seamlessly and enable them to obtain further information about a certain topic. Even so, it is true that further thinking regarding this topic can be made, with the intention of improving the developed methods for presenting related information. Ideally, it would be nice to have multiple sliders, one on top of the other, containing the distinct events of the multiple associated timelines. Deciding on which slides, from which timelines, would be displayed in the interface, would be the responsibility of the user. Upon moving the time frame in one timeline, the other could also go forward, or backwards, accordingly. However, in our understanding, a task like this would require the knowledge of someone more experienced in the graphic design for the Web, as sometimes it can be hard for a developer like us to have this kind of sensibility.
VIII. CONCLUSIONS

History has never been so actively disseminated as it is today. Throughout the years of its short existence, the WWW has allowed historians to present their knowledge and experiences by using distinct types of strategies, like for example blogs, forums, or even personal Web pages. However, the prevailing presentations of historical data, limited by the early versions of HTML, are still structured in a tabular layout that does not allow readers to efficiently comprehend the topics under examination. Recently, new digital technologies have been introduced, which allow the creation of visually engaging timelines that display the contents in a more interactive and dynamic format. But in our perspective, these modern software applications still face some problems that we consider to be of great importance, namely the inability to represent the multiple linked data accurately or to reuse and embed any previously created information. Furthermore, the majority of the existing tools either do not have support for collaborative sharing and production of contents, or it is rather limited in the number of people allowed to work on a joint project.

The goal of this research was to address these referred limitations by defining and implementing a Web-based software system, entitled WikiTime, which allows the simple creation and editing of timelines for proper visualisation and organisation of time-related data. Moreover, the envisaged solution was developed with the words flexibility and integration in mind. Building a Web Service API that would allow the complete integration of WikiTime’s functionalities with other external applications, was also a key aspect that was taken into consideration. When compared with the traditional methods, WikiTime’s application offers a more effective interface, enabling users to move quickly through time, to efficiently analyse and compare different historical periods, and to quickly move between closer or wider perspectives. Built upon a collaborative working environment, it is capable of providing its users with the possibility to properly organise events in time, to display the complex relationships that compose history, and to reuse or encapsulate previously created timelines and events.

In conclusion, WikiTime’s software application has proven to be a tool worthy of chronological display, providing meaning and form to history’s connotations of narrative, and allowing for greater comprehension of the world’s past, present, and future.

REFERENCES


