

Informatics 121

Design Studio 3

# History

## Group Members:

Siyu Hu

Rasika Bhalchandra Athavale

Ace Jack Lowder

Chris Rodriguez

Payam Dowlat Yari

Shaolong Lin

## Table of Contents

<b>Introduction</b>	<b>3</b>
<b>Application Design</b>	<b>4</b>
Goals	4
Constraints	4
Assumptions	5
Audience	5
Other Stakeholders	6
Design Method 1 - Website Concept Map	7
Design Method 2 - Persona Analysis	10
Feature List	13
<b>Interaction Design</b>	<b>15</b>
Goals	15
Constraints	15
Assumptions	15
Audience	15
Other Stakeholders	16
Design Method 1 - Mockups	17
Search by Name	17
Search by Date	18

## **Design Studio 3: History**

**2**

Explore 19

Comparison 21

Description of Mockups 22

Design Method 2 - Feature Comparison 23

## **Architecture Design**

**25**

Goals 25

Constraints 25

Assumptions 25

Audience 26

Other Stakeholders 26

Design Method 1 - Reference Architecture Diagram 28

Reference Architecture Diagram 28

Description 29

Design Method 2 - ER Diagram for Database 31

Entity Relationship Diagram 31

Description 31

## **Conclusion**

**35**

## Introduction

For this design studio, we were tasked with creating a tool that allows users to archive and save a website's history. We decided to produce a single design iteration, with the majority of decisions and developments originating from supporting documents and artifacts generated along the way. We also researched various existing applications and APIs that could potentially influence or support our design. When conceiving the final design, we focused on the program's application, interaction, and architecture and applied various design methods to help develop these segments. It's in this variety that we were able to develop an application that we feel meets the essence of the project.

# Application Design

## Goals

- Track future changes to websites.
- Design a system to allow a side-by-side or overlayed comparison.
- Give the ability to select a date in the past and jump to that date's snapshot.
- Track changes to make spotting updates easier.
  - Generate a comparison report between two dates.
- Keep tracking data secure and anonymous.
- Comply with any laws regarding site monitoring and scraping.
- Be able to filter out days with little to no changes.
- Track personal site usage.
- Efficiently store data and have the ability to quickly present a visual representation.

## Constraints

- Non-public websites will be inaccessible.
  - Pages may change accessibility periodically.
- Sites or pages may go down permanently.
- We cannot go back in history and can only record future changes.
- We do not have access to previous states of servers
  - Any content that uses servers will not work or render.
- Some languages or frameworks may not be able to be parsed or saved.
- Technologies, such as flash, are being discontinued, so the content may not render correctly in the future.
- We won't be able to archive copyrighted content.

## Assumptions

- The user will have access to a smart device.
- Our program will have a subset of sites we'll be able to monitor
  - Websites will not ban us for repeated queries
- We'll be able to manage the load presented by users.
  - There's enough space to save content.
  - There's enough processing power to scrape data and crawl sites.
- Users will not expect any past data to be tracked, only the future.
- The backend will be responsible for monitoring, so the user does not need to keep the application or site open to keep monitor active.
- Website information will not be encrypted, so it can be read and displayed.
- The sites that we will monitor will not run any malicious scripts.
- All of the content we monitor or save will be under fair use.
- Sites will have some way to let us know which pages we can or cannot monitor (i.e. a robots.txt file).

## Audience

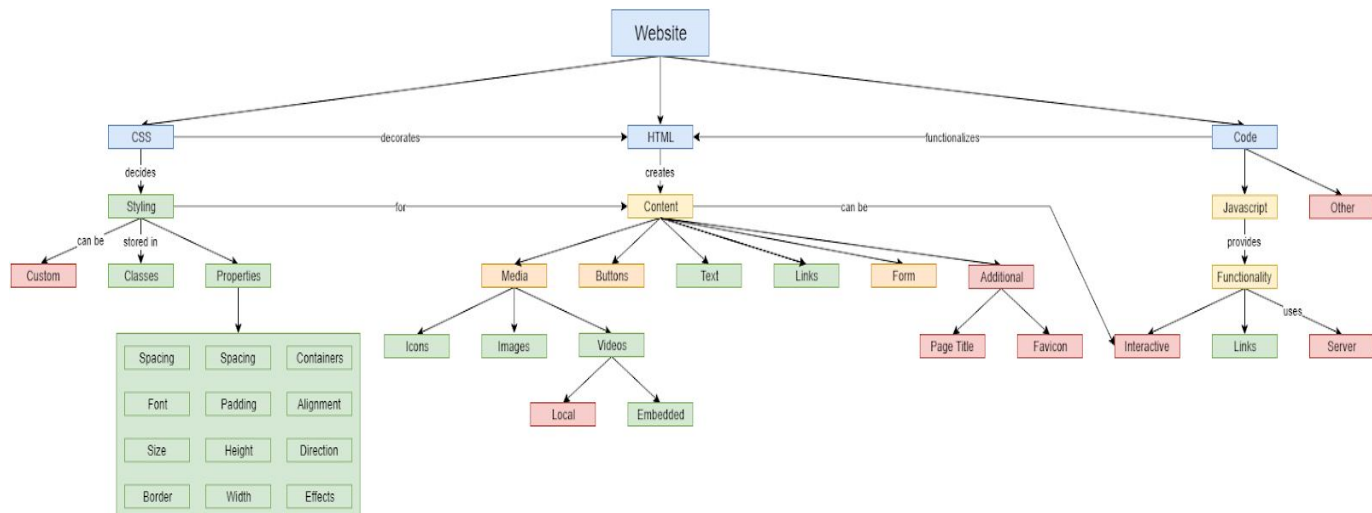
- **Families:** Will mainly use the software to monitor each other's online personas through social media. These users will be more interested in the visual aspects of the software and less interested in real time comparisons.
- **Businesses:** Would most likely use the software to monitor competitors and see how their sites have changed. Companies may want to view how they present themselves and manage their site over time.
- **Businesspeople**
  - **Stockbrokers:** Will be interested in viewing raw data, trends, and comparisons.
  - **Lawyers:** Would use the software to quickly view changes to any text, image, etc. on a site. Interested in the specific time of changes.

- **Students:** Quickly uses the software to complete assignments. Convenience is key for this user, and they may not use the program for prolonged periods of time.
- **Researchers:** Would monitor long spans of time and usage statistics as pages change. Flexibility of timeframes and views of these time are key, as researchers wants can vary drastically between fields and projects.
- **Personal site admin:** Wants to know how code, load times, and other statistics change as the site they oversee changes.

## Other Stakeholders

- **Owners of the sites being monitored:** These stakeholders may not want content to be saved or monitored due to personal reasons.
- **Server companies or system admins:** Administrators would be concerned with the number of requests being made and possibly any malpractice on our software's end.
- **Governing agencies:** These agencies may be interested in our monitoring policies and ensuring that our software complies with any federal rules and regulations.

## Design Method 1 - Website Concept Map

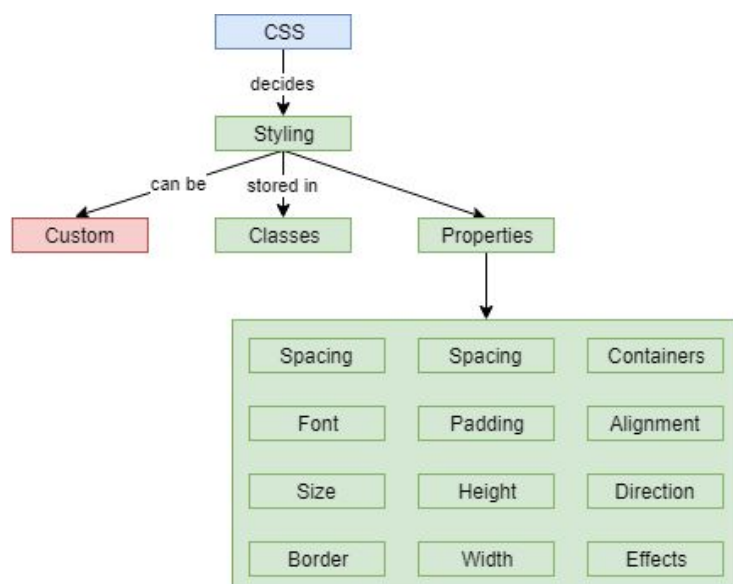


To determine which features are feasible for our software, we created a concept map that notates how sites function and which elements of a site we believe are necessary to store. Understanding the inner workings of websites allows us to learn the limitations that come with storing certain aspects of a website.

During this process, we learned that almost all sites are divided into three distinct sections that work together to display content and enable interactions. The CSS portion of a site denotes the styling for all of the content on a page. The HTML provides the actual content that is displayed on a web page. Javascript is often used to enable interactions on the site that users can utilize. While laying out each key segment of a site, we chose to save different portions of the data to maximize our efficiency while still achieving the goals that we aimed to implement.



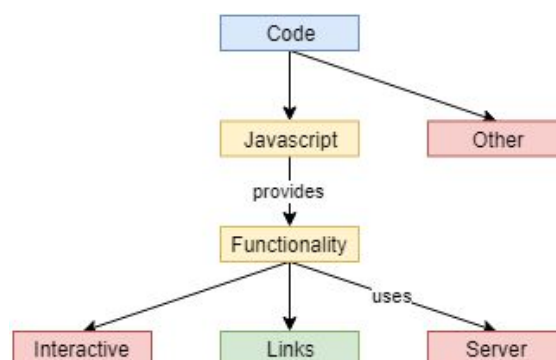
In our CSS section of the sitemap, we decided to keep most of the elements because we found that the visual aspects of websites were important to our design. This information is necessary to ensure that the websites we monitor display their content properly. Since many of our planned features focus on the visual changes of the websites, we wanted to make sure that we saved as much of this data as possible. The only CSS that we chose not to save were custom CSS elements. These custom elements would be difficult to store and manage but will only affect a small minority of websites.



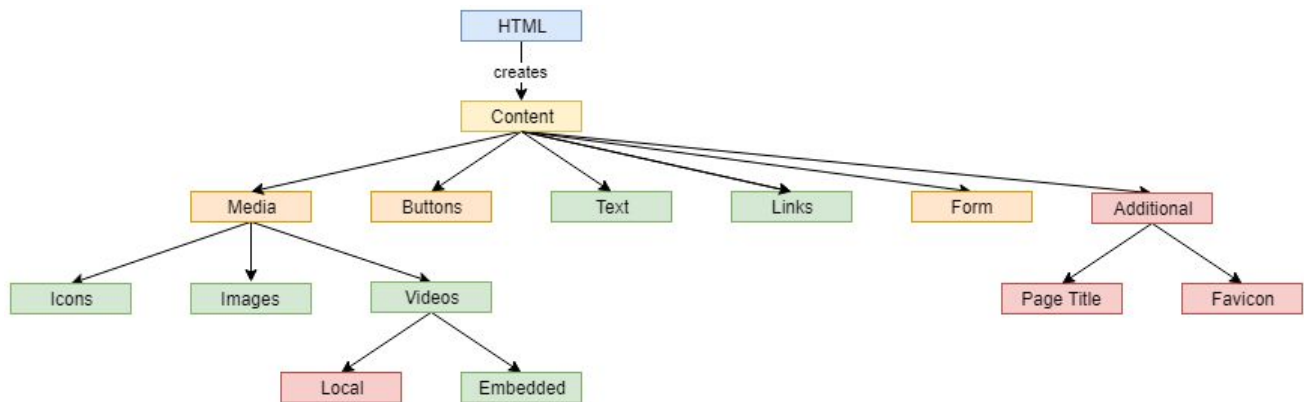
The code portion of a site provides functionality for the content on the page. This allows the content to become intractable by the user. Almost all of this functionality was deemed unnecessary for our current design iteration. The most important aspect that the code provides are links that the user can click to be directed to a separate page on the website. These links would need to be captured if we want to allow users to navigate an archived version of a site. One of the other

realizations that we had while creating the sitemap was that all of the code that interacts with servers would become obsolete in previous versions of a website. Our older snapshots would not have the ability to contact older server states so we plan to stay efficient by storing only the necessary code.

The final portion of the site is the HTML that stores the actual content for any given page. Web content comes in several forms so it was important for us to determine which elements of HTML we needed to save in order to preserve our software's functionality. The main issue with the content of web pages are the large file sizes that



accompany the media. These files would be saved under the condition that they do not take up a large amount of space on our servers. We also optimize our websites by refusing to save any video content that may appear on a website. We wanted to stick to saving relevant information that the user can easily compare with separate snapshots if needed. Similarly to the code, HTML stores the functionality for links in a website. In order to preserve the natural navigation of a site through menus and buttons, we plan to save all of the data in websites that allows for navigation.




Creating the sitemap to help understand how websites function and are put together on the backend allowed us to easily determine which information we need to store when choosing to monitor a site. This design method also brought forth several design decisions that allowed us to remain true to our original goals for the application design.

## Design Method 2 - Persona Analysis

Amy Chandler

**The Busy Businesswoman**



Age	Occupation	Location	Education
35	Lawyer	Irvine	College Graduate

### Goals

- Be able to see all changes
- Efficiently see changes over a large time period
- Curious to see if a change happens, rather than expecting one to

### Story


Amy's working on a case where an online company has been conducting some shady business. She's heard rumors of the site's sensitive content and terms changing without notifying customers, so she wants to use a tool to check this for her without having to scrutinize every page.

Amy is a very busy individual, so she's not interested in checking up on the changes day-to-day; rather, she's just going to want to see what changes happened since the last time she checked. Any small changes could be very important to her, so all reasonable site data must be tracked.

### Behaviors

- Busy
- Detail focused
- Efficient
- Impatient

**Buddy Moses**  
**The Analytical Student**



Age	Occupation	Location	Education
21	Student	Los Angeles	Some College

## Goals

- Be able to quickly view large changes to site content
- Save media content, if possible
- Not be weighed down by small changes

## Story

Buddy is working on a school project where he's monitoring trends of online content. Specifically, he wants to see how the front page of Reddit changes from day-to-day and wants to analyze the data based off that. He's going to be interested in seeing daily changes, and is confident that there will be a lot of differences as content changes quite often.


Buddy will be interested in a more general view of the changes, and will not need to scrutinize things like a change of a small label as most all of the content will be completely different every day.

## Behaviors

- Tech savvy
- Patient
- Analytical
- Data driven

Ismael Harmon

The Internet Historian



Age	Occupation	Location	Education
30	Retail Worker	San Diego	HS Diploma

## Goals

- Track sites for a long period of time
- Be able to track multiple sites
- Ignore days with no changes

## Story

Ismael works an average 9-5, but in his free time he wants to casually monitor how his favorite sites change over time. For example, he wants to track Apple's site changes throughout the year as products update and release cycles change.

It's just a side project, so complete thoroughness isn't that important. But what is important for Ismael is that he can monitor changes for a while. Also, he won't want to be overwhelmed by all the days collected, as changes for the sites he wants to monitor will be periodic and not too often.

## Behaviors

- Easygoing
- Inquisitive
- Passionate

Our personas help us capture various scenarios our program will have to encounter. Though not exhaustive, they help us derive features that should be included in the final product as well as features that may not be worth pursuing.

For Amy, her primary concern is centered around being able to track all data and quickly see any differences. We're able to meet her needs, as we achieve a deep level of tracking for a given site, and our comparison report means she doesn't have to scrutinize two pages side by side.

Buddy's needs are less granular than Amy's but are still fully realized. He'll be monitoring sites with a large number of changes daily, so a comparison report isn't very meaningful. However, our side-by-side comparison view offers Buddy a clean interface to quickly see changes at a high level. And given that he's interested in monitoring social media sites, we should be able to store the majority of the content as outlined in the concept map, as most media is hosted by the site and won't require us to locally save the media.

Lastly, Ismael is our most casual persona in terms of needs. He plans on monitoring sites for a long period of time, expecting changes but only every month or so. Given that he wants a very general view, our search by date screen would most likely be the best option for him. Thumbnails show a given site's snapshot, and narrowing these thumbnails offers Ismael a fast way to see when changes happen.

## Feature List

- Allow users to create accounts
  - Require login information and account creation
  - History is personalized and associated with the account
- Users may select a website to track through a url
  - Let users monitor multiple sites easily
  - Tracks changes indefinitely, or until stopped
- Be able to save thumbnails for quick view of changes
- View frequency of changes to a site
- Navigate archived versions of sites

- View website snapshots on specific dates
- Offer easy methods for viewing page differences
  - Compare website snapshots through a side-by-side view
- Keep track of site metrics
  - Get daily traffic to site
  - General trends
    - Average time users spend on the site

# Interaction Design

## Goals

- The software must capture the online history of user interfaces over time.
- The system should implement the search option into the interface.
- The system must allow users to compare two search results visually.
- Users should be able to go back to the previous version of a website easily.

## Constraints

- Users cannot interact with the previous (server-side) version of a website.
- Users can only make HTTP requests to the latest version of a web server.
- The system does not have access to the archive, and hence, does not provide information regarding the time before its development.

## Assumptions

- The history of websites starts from now (1990 for the purpose of mockups) and the information can be used in the future.
- Users know how to search for a specific webpage based on date, keywords, and address.
- Users are assumed to use the exact address of the website when searching based on the address.

## Audience

- **Law firms:** Law firm representatives might be involved in some initial testing of the look and feel of the UI such as paper-testing etc, so their opinion and experience will matter as they're one of the main audience members for the parent company.



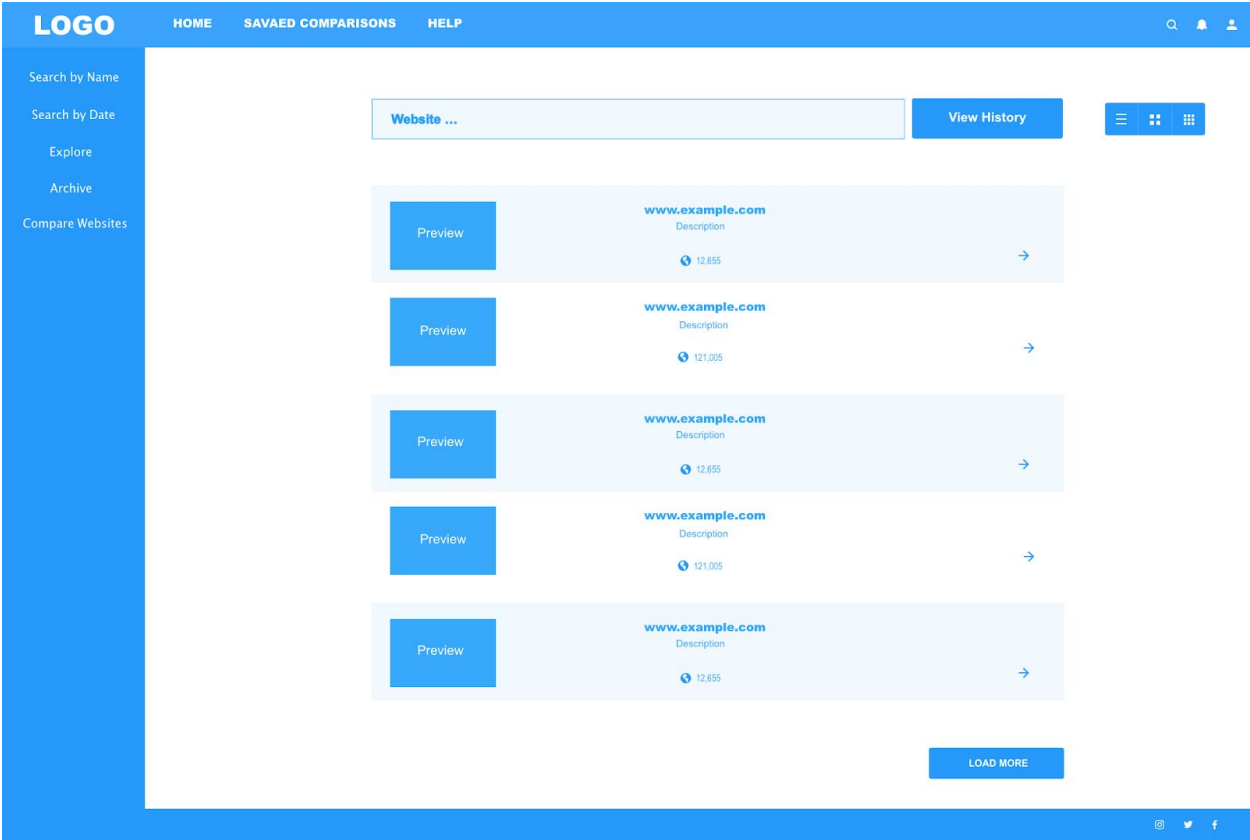
- **Universities:** University officials might be involved in some initial testing of the look and feel of the UI such as paper-testing etc, so their opinion and experience will matter as they're one of the main audience members for the parent company.
- **Individuals:** Individuals might be involved in some initial testing of the look and feel of the UI such as paper-testing etc, so their opinion and experience will matter as they're one of the main audience members for the parent company.
- **Families:** Families might be involved in some initial testing of the look and feel of the UI such as paper-testing etc, so their opinion and experience will matter as they're one of the main audience members for the parent company.
- **Designers and developers:** The designers and developers can use interaction design as a common language to describe the functionality of the system.
- **Owners of the site:** Owners of the site would impact the interaction as well as their vision and mission statement such as good usability etc impact this domain the most.

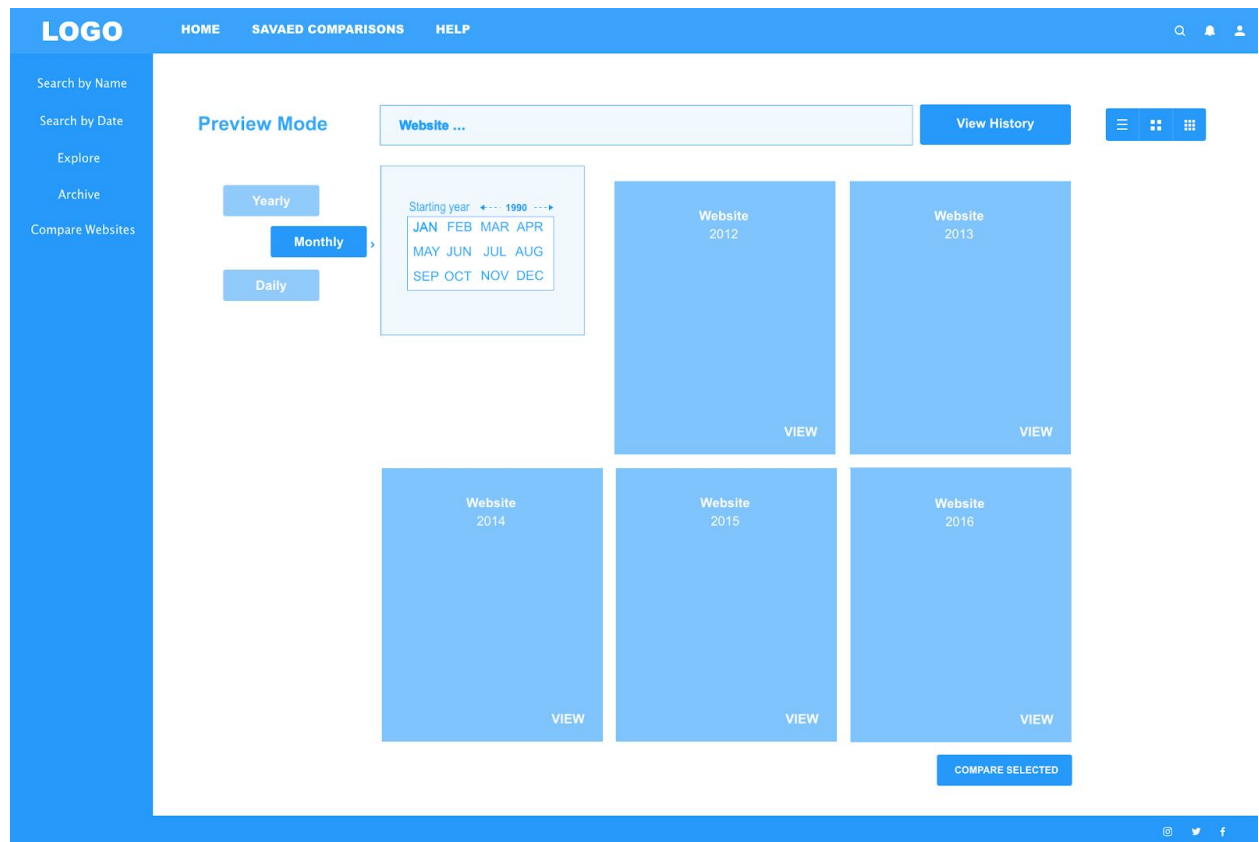
## Other Stakeholders

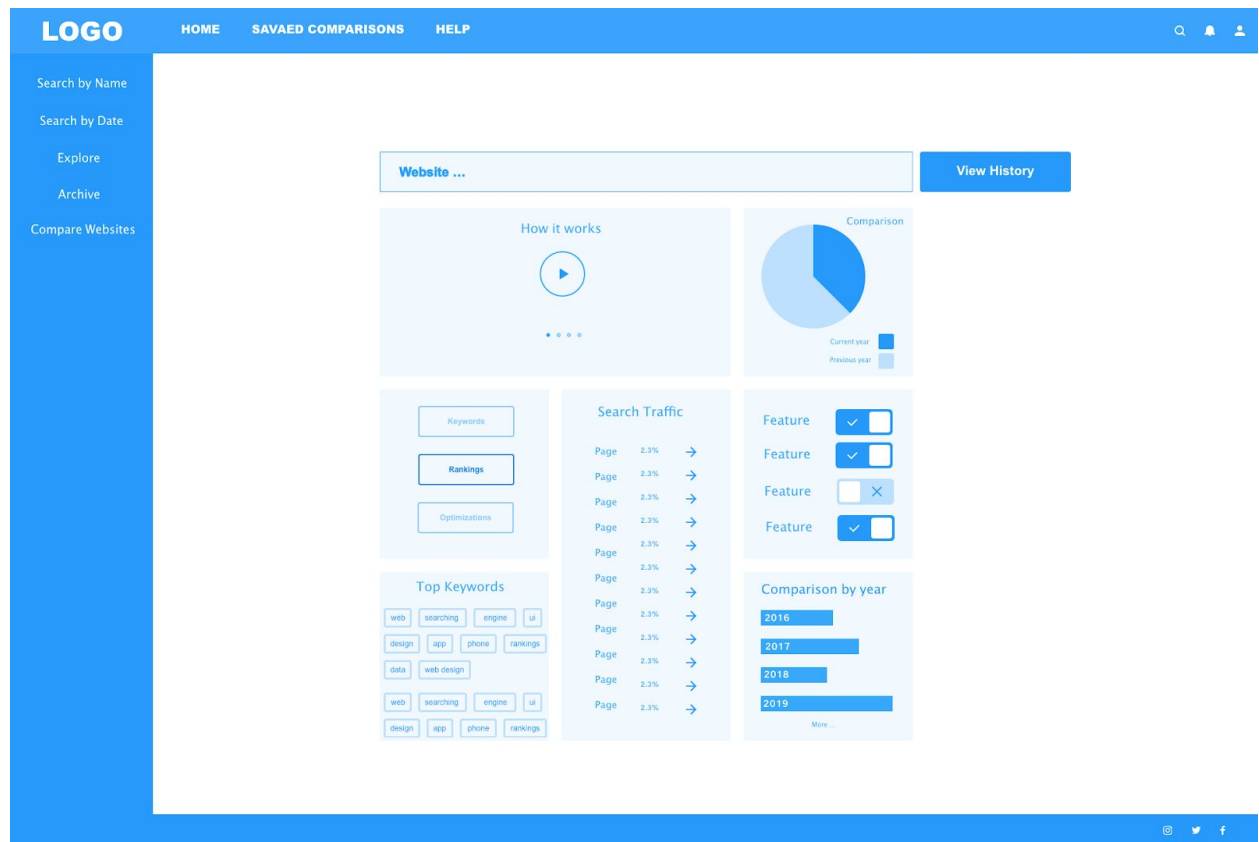
- **Customer Success Team:** The Customer Team will help sort out the features support for customers, their voice matters in the design.
- **UI/UX designer:** The experience and knowledge about UI heuristics that the UI/UX designers have will impact the UI.
- **Front-end UI and graphing libraries:** Depending on what kind of libraries/framework is chosen for the front-end, the look and feel of the UI may differ accordingly.
- **Archive.is :** This was used in the competitor analysis so their UI gave us some ideas about what's out there.
- **Wayback Machine UI:** This was used in the competitor analysis so their UI gave us some ideas about what's out there.

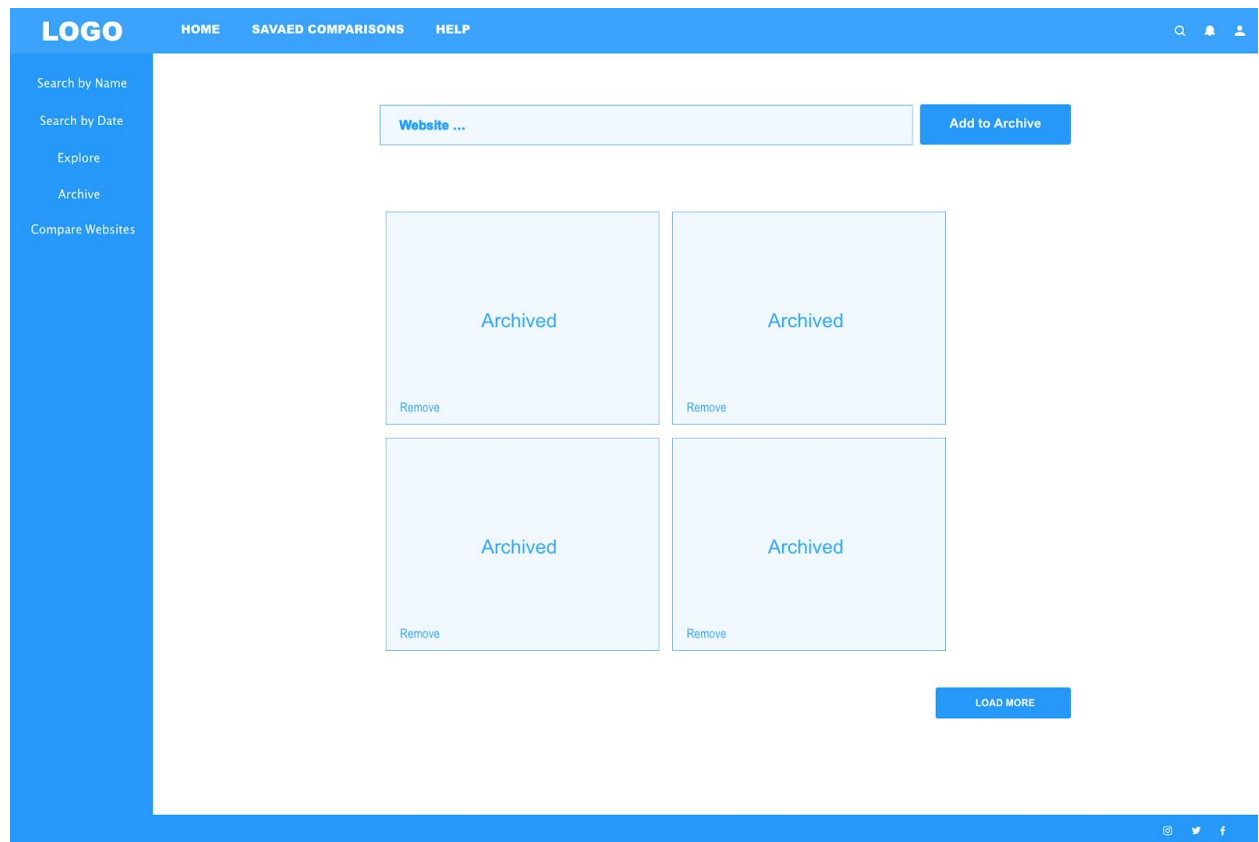
Design Method 1 - Mockups

Search by Name

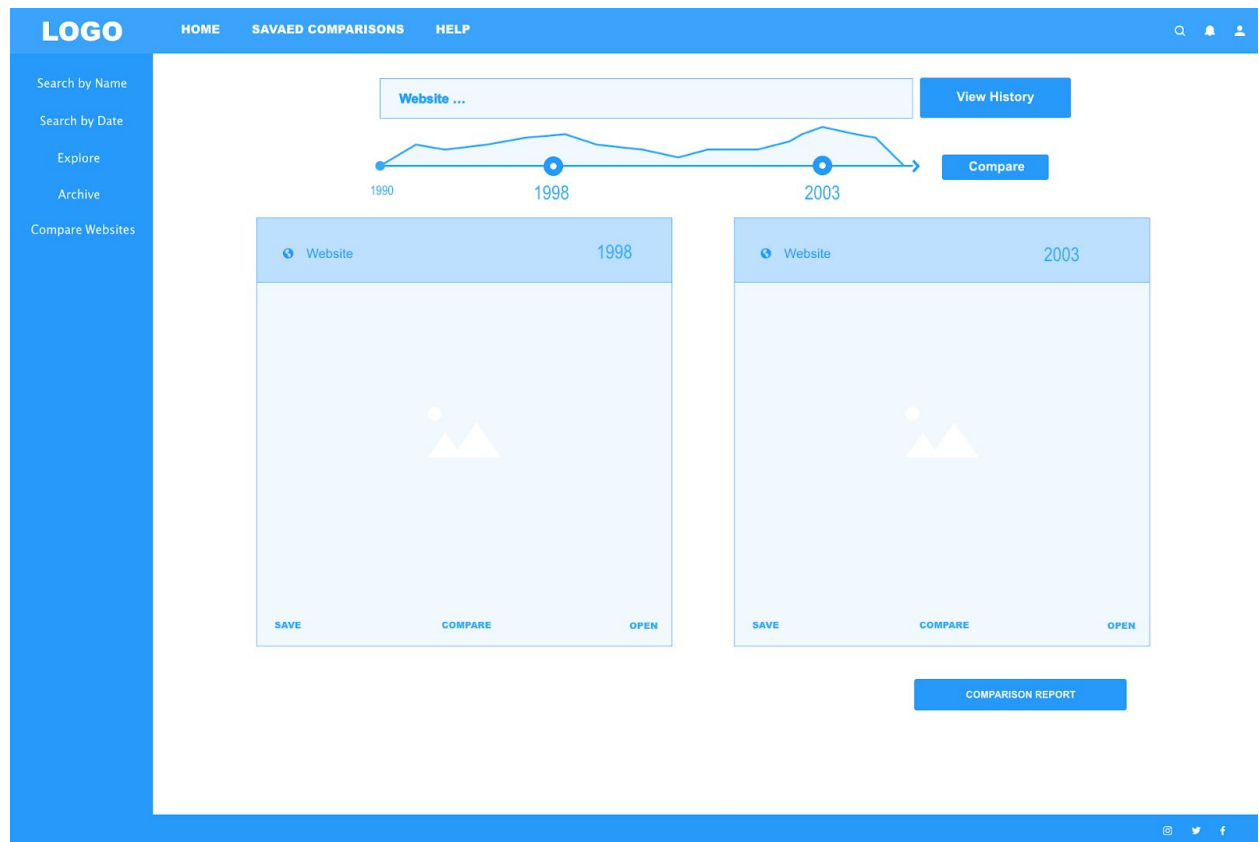


*Search by Date*

*Explore*

*Archive*

## Comparison



## Description of Mockups

- Search by Name

This is the main page or the first-page user interacts with. The most important item is the search engine bar. After inserting the name of the website by the user, the system displays the results of the search with a little preview of the page.

Users can choose how the results are displayed (menu on the top-right). The results include a description of the page and the number of times the page was searched. By clicking the small arrow that website will be open.

- Search by Date

Another feature that is provided for users is in addition to searching the name, they can specify the exact date or period they preview the history of a website.

The date can be chosen as yearly, monthly and so on. Users can also compare two or more different dates as a history of the same webpage.

- Explore

This page provides analytical data for a particular website. The data is retrieved from Alexa API but the interface is specially designed for the specific purpose of the users. Information includes traffic, keywords, number of active users and so on. There is a short video tutorial on the top of the page to provide more help.

- Comparison

On this page, users can select from the whole time frame and compare more detailed information about a website in two different times. After choosing the specific date from the timeline on the top for the first and second items, the system provides a comparison report. Users can save the information for a website or the comparison results.

- Archive

Using this option, the users can search and add the results to the archive for later. They could also remove from the archived search results.

## Design Method 2 - Feature Comparison

- **Identify competitors and their products.**
  - Wayback Machine (web.archive.org)
  - Archive today (archive.is)
- **Establish dimensions for comparisons.**
  - Homepage
  - Logo
  - Navigation bar
  - Search engine bar
  - Recommendations
  - Number of links
- **Conducting research based on website:**
  - **Wayback**
    - The most essential part of the Wayback Machine home page is the search engine bar that is centered at the middle top of the page below the navigation bar.
    - The logo of the website is located next to the main search engine.
    - There are two navigation bars on the top of the page each one includes around 7-10 items. There sign-in button at the top right.
    - There are buttons for donating options available both on the navbar and next to the main logo.
    - A line of recommended options is shown below the search bar.
  - **Archive**
    - Archive's website is very simple compared to Wayback
    - It also focuses on the search engine bar
    - Navbar is very minimal on the top right of the page including 3-5 items.
    - It has two search engine (snapshot and content)



- A limited number of links are available on the page
  - There is a simple logo on the top left of the page
  - The donation option is also available
- **Analyzing the results:**

	Wayback	Archive
# Search Engines	1	2
Logo	Large	Small
# Navigation bars	2	1
# Items on the navbar	7-10	3-5
Minimalist design	N	Y
# Donation option	Y	Y

# Architecture Design

## Goals

- The website will secure users' information through a login system.
- The website will preserve the intended date that the user wanted to start tracking a URL despite any network-related latencies etc.
- The website will show metrics such as daily traffic generated on the website and daily usage time on the website as a basis of comparison along with the layout changes.
- Simplify the process of adding more metrics.
- Not reinventing the wheel by using APIs

## Constraints

- The users cannot use Google account directly log in to the system, the users will have to sign up to the system using email.
- The system won't be able to stop the history recording until it receives input to do so from the user.
- The user cannot choose a date that is former than the using date, for example, if the user starts to use the app on 12/5/2019, he/she can only start to track on this date or after.
- The system won't be able to categorize websites since it's using the WayBackMachine API, thus, there is no Search By Category feature.

## Assumptions

- The system will use the Alexa API to get data about websites such as daily traffic and daily usage time.
- The users can view the comparison rendered by the front end well irrespective of their screen size etc.
- The system generates a new URL containing snapshots of websites daily.

- The default timestamp is to do so at 8:00 a.m. every day. At that time, the system will save the current URL and update the system to the next timestamp.
- The system is using WayBackMachine to generate the new URL for every timestamp.
- The website's front-end can send some data and handle receiving data to and from the backend.
- Users will not expect any past data to be tracked, only the future.

## Audience

- **Designers:** Some limitations of the architecture can decrease the functionality of the system which then changes the interaction and application design. Similarly, something that can be done in the architecture might add functionality as well leading to changes in both the other design domains.
- **Developers:** The developers of the system need to understand the architecture to implement it
- **Architects:** The architects need to have a deep domain of knowledge to design the architecture and they need to work with other designers to understand all the functionality.

## Other Stakeholders

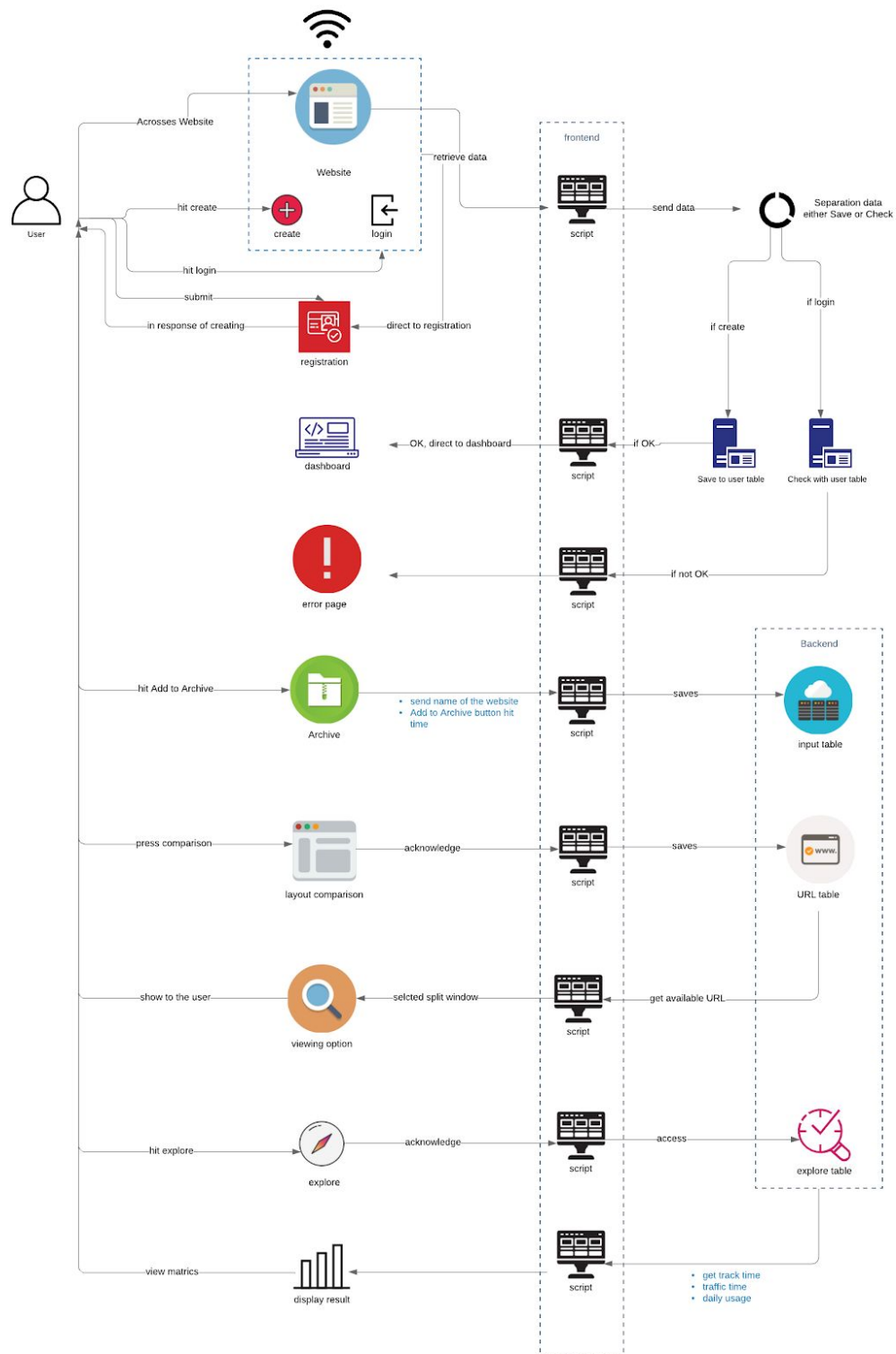
- **Privacy Laws:** Depending on the laws surrounding privacy, the authentication or password criteria could change, etc.
- **Other Websites:** Other websites that value monitoring the changes they make over time may want certain other functionality added which ultimately results in possibly making some underlying architecture changes if necessary.
- **WaybackMachine API:** If the WaybackMachine API is impacted in some way, then that impacts our design and functionality as well.
- **Alexa API:** Since our system uses Alexa API for generating the frequency metrics, then the changes made on Alexa API will also make differences to our metrics.

- **Owners of the site:** Owners of the site would impact the architecture as well as their vision and mission statement such as good usability etc impact all of the design domains.

## Design Method 1 - Reference Architecture Diagram

### Reference Architecture Diagram

<https://www.lucidchart.com/invitations/accept/4587ab3c-7a33-4bbc-a335-ccca587b43ca>



### *Description*

The reference helps to explain how the system will take in data, save data and access data.

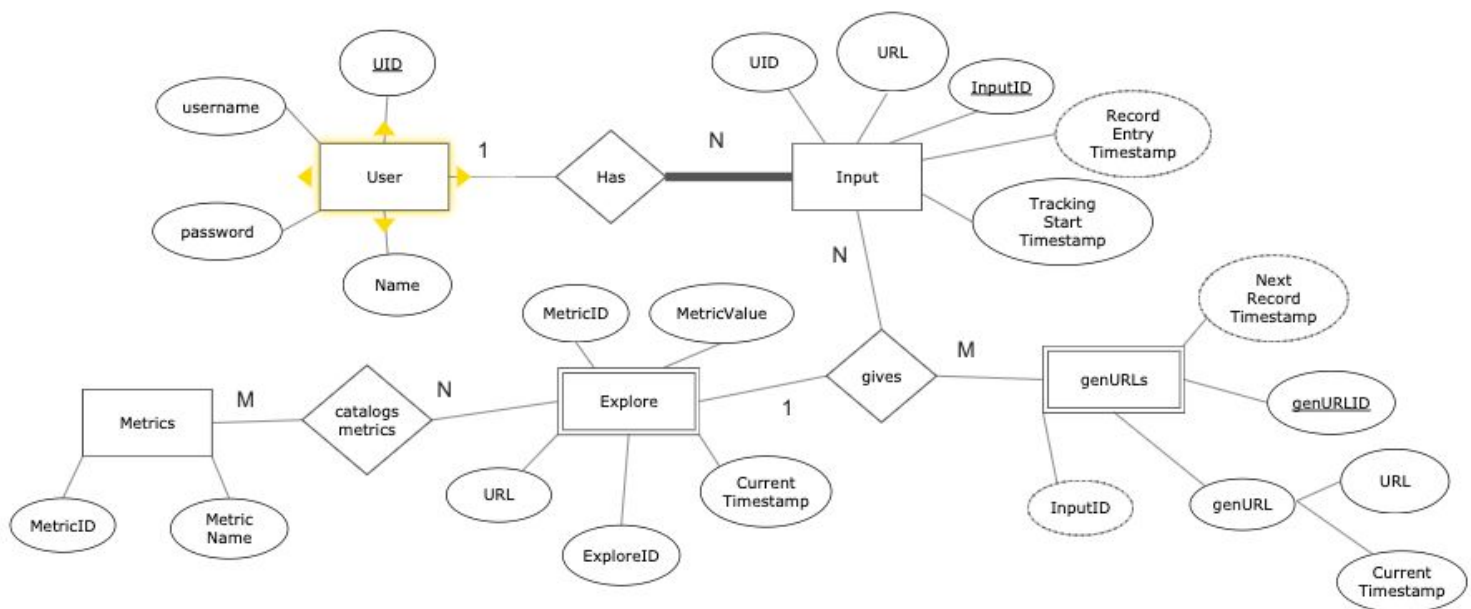
1. User will need to get access with the internet before they want to use the website
  - a. Website
    - i. Create
      1. If the user wants to create an account, the website will retrieve data and send to the frontend scripts
        - a. User will submit the registration in the response of creating account
    - ii. Login
      1. If the user wants to log in, the website will retrieve data and send to the frontend scripts as well
2. Frontend scripts collect data and separate the data to Save or Check
  - a. Create
    - i. Save the data
  - b. Login
    - i. Check the data with the saved data and see if they match
3. If the data is saved
  - a. Send back to the database
    - i. Successfully direct to the dashboard
4. If the data is not matched with the saved data
  - a. Send back to the database
    - i. Direct to the error page
5. The user hit “Add to Archive” button in the Archive section
  - a. Send the name of the website and the time when the user hits the “Add to Archive” button
    - i. Send data to the database
      1. Database saves the information to the backend, in the input table

6. The user hits views the comparison of websites
  - a. Frontend scripts send an acknowledge to the database
    - i. Sends event trigger information to the database
      1. Database saves information in the URL table
        - a. URL database send available URL back to the frontend scripts
          - i. Frontend scripts allows the user to select viewing options
            1. Finally, render the result to the user
7. The user hit explore
  - a. Explore acknowledge frontend scripts
    - i. Frontend scripts access the explore table
      1. Explore table get track time, traffic time, and daily usage and send them back to the front end
        - a. Frontend scripts render results
          - i. Finally, metrics can be viewed by the user

## Design Method 2 - ER Diagram for Database

### Entity Relationship Diagram

<https://cloud.smartdraw.com/share.aspx/?pubDocShare=08257855096C398D1C4A3ABA8110CEE6EEF>



### Description

With respect to the reference architecture above, the database consists of several entities and relationships which can be implemented as different tables of a database possibly. The description and details about the entries and relationships modelled above, are also described below:

- **User**
  - A user is an entity in this model that has 4 attributes: UID, name, username and password.
  - The username will be an email address which uniquely identifies the user.
  - Password is an 8 character string with certain restrictions about the presence of at



least 1 special character, 1 upper case letter and 1 lower case letter. UID is a uniqueidentifier to identify every user in other tables efficiently.

- A new record is saved in this table when a user registers on the website. After that, everytime the user tries to login using their username and password, this table is accessed and the username and password are authenticated. This is to accommodate one of our goals about user security.
- **Provides Relationship:**
  - After logging into the website, the user sees their dashboard and can input information to the website. A user can have many inputs, but a given input is associated only with 1 user as indicated by the cardinality
  - Input as an entity wouldn't exist if there was no user as indicated by the bold line.

- **Input**

- An input is meant to store the user's inputs and has 5 attributes: UID, URL, InputID, RecordEntryTimestamp, TrackingStartTimestamp.
- UID is a foreign key associating which input corresponds to which user. URL and RecordEntryTimestamp is the URL that is received from the website as input, and the timestamp of when the user wants to start recording the history (generated by the website's front-end). InputID is a uniqueidentifier for every input and RecordEntryTimestamp is a timestamp to note when this record was actually created.
  - The additional field of RecordEntryTimestamp is meant to catch some edge cases and ensure that user intent is preserved. For example, if a user wants to start recording some website at 11:59pm tonight, but due to some WiFi issues etc, the entry isn't made into the table till 12:01am tomorrow. We still want to maintain the fact that the user wanted to start archiving from today and not tomorrow.
- Data is saved in this table every time the user hits "Add to Archive" button.
- **Gives Relationship**

- Gives is a ternary relationship between Input, genURL and Explore. For every genURL and Explore record there has to be a corresponding Input record, and hence genURLs and Explore are weak entities. As shown by the double-lined rectangles
  - Any given input and any given URL have a single corresponding record in Explore as shown by the cardinalities.
- **genURLs**
    - genURLs is meant to store the historical URLs generated for the duration of the time period that the user archives them. These historical URLs are used by the front end for the comparisons. It has 4 attributes: NextRecordTimestamp, genURLID, genURL, InputID.
    - NextRecordTimestamp is a calculated timestamp that basically resets after a URL is generated, say today to tomorrow. It can be used to automatically capture the next URL. genURLID is a uniqueidentifier that can be used along with the InputID to uniquely identify every generated URL. genURL is a composite attribute that consists of the URL that is generated. This URL is generated using the WayBackMachine API. The API needs a timestamp and URL as inputs. In our design, we'll use the InputID to get the URL and NextRecordTimestamp. If the NextRecordTimestamp on any given day corresponds to the same day, that implies the URL for today's snapshot has not been generated. In which case a call to the API is made and saved in the genURL field. The timestamp at which this happens is saved in the CurrentTimestamp part of genURL and then NextRecordTimestamp is updated to tomorrow (say).
  - **Explore**
    - When websites change, they change not only in layout but also in other metrics such as traffic and daily usage time. The Explore feature is meant to show this comparison as well. In this data design, it has 5 attributes: MetricID, MetricValue, CurrentTimestamp, URL, ExploreID.

- The ExploreID is a uniqueidentifier for every record in this table.  
CurrentTimestamp records the time at which every entry was made. The metrics we currently want to track are daily traffic and daily usage history. The values for these metrics are available through the AlexaAPI. These values will be stored in the MetricValue field.
  - It is possible in the future, that more metrics need to be tracked. To make the process of adding more metrics simpler every metric has an id. So when you get a value for a particular metric, that value and its corresponding MetricID are saved in the entry.
  - In the event you decide to add a new metric, this table can be modified by just adding the appropriate values and timestamps for the new metricID.
- As alluded to above, this table uses MetricIDs to facilitate adding in extra metrics in the future. And hence this entity has a relationship with the Metric table. Many entries of this table may correspond to many entries of the Metric table as well and hence the many to many cardinality.

- **Metric**

- This entity is meant to create a unique mapping for every metric you might want to keep track of. It has two attributes, MetricID and MetricName
- MetricName is the name of the metric, such as “Daily Usage Traffic” etc.  
MetricID is a unique identifier for every name which is used by Explore.
- The highlight of this mapping is that every time a user wants to add a new metric just by adding a new row to this table and by adding the corresponding values in the Explore table, you can add as many new metrics as needed.

## Conclusion

Creating an archive of the internet is no small task. By breaking the problem down into smaller components, though, we were able to create the beginnings of a plan. Design methods helped us in making our design more robust, and also helped us realize which features were maybe unneeded or too difficult to reasonably accommodate. Also, by investigating existing software and APIs, we were able to use ideas and solutions that work, as well as take inspiration and improve or add new functionalities.

Our web archiver can successfully monitor multiple websites, which are easily searchable and cleanly organized. Comparisons and statistics are readily offered, and monitoring can continue indefinitely. All of this is computed and stored in an efficient manner that allows quick lookups and even potential additions or modifications to the underlying model. The program is fully featured, but still captures the essence of our requirements.