# Wiki Formatting Design Document

Authors
Mario Macias - memacias@google.com, Dustyn Loyda- dloyda@google.com, Alexandro Fernandez - alexandrof@google.com

## Background

PRD Document:
https://docs.google.com/document/d/1NrieKZQc6yoFq4EUnBqdNYIu3KFcCMwpnW48SevVphY/edit?usp=sharing

The problem is how to implement a receive function that allows for headers, paragraphs and links to other pages.

## Proposal

Through the use of HTML tags we can achieve paragraph separation and header formatting.

## Implementation Details

Default pages will load using the prescribed rendering format from our wiki assignment, from there, user input from thread creations underneath the city dark fact will utilize a variable as a sandbox. This will allow the parser to not have to execute unwanted commands whilst reading from the user input. Then the variable will be given as plain text to the parser so that the parser can then render it onto the template and the wiki page will load safely. This same concept applies to file reading in order to keep user given content from running unwanted commands.

The use of HashMaps to store States/Provinces as keys with a HashMap of each city as a value which will contain the city's thread content. In essence the structure would be a HashMap storing a set of HashMaps which then contain plain text data. This helps maintain small time complexities but saves the complexity of having to delete from two separate structures when cleaning up the facts about said cities.

The use of <p> to separate paragraphs by reading through the given input and looking for line breaks to insert said <p> tags prior to exporting onto the wiki. The <p> tag in HTML, takes some text in between tags and adds a blank line before and after the tags.

The thread posting function will open up a mini text editor which will allow the user to input a title, content and make links within that content. This allows us to separate the content into plain text content and the link they are trying to post. When saving the data back onto the file 2 sections would be saved: Title and content which will allow us to render the user content with the appropriate HTML tags.

The use of <h1>-<h6> to format section titles with altered text to distinguish the title of the section from the section context. The <h1> - <h6> tags in HTML, take the text within the tags and bolden it as well as increment the size of said text, the higher the number following the h the smaller the font will be on the text to be formatted.

Users will be able to post comments for cities, which will be stored and reflected every time after the posting. This content will be sandboxed by the following:

> User discussion thread contents are injected into a variable for which the parser would not try to parse. The parser would simply add the contents of the variable as plain text at HTML render, similar to sandboxing the data.

# Security Considerations

Treat input from users as a string into a variable

- This will prevent users from being able to give the parser commands that could compromise the wiki
    - e.g. user_in = ‘El Paso, Texas <script>window.alert(‘Hacked!’)</script>’
        - Will be treated as variable content instead of a string then command

Files will be read from a secure file on the server, to open the file a key must be passed along with the request to open the file, otherwise file will not open and contents will not be editable

- Not securing the files will allow non-authorized users to make changes into whatever they please which would go against our policies.

Name hashing to allow for only the original poster of a comment/thread to edit said comment/thread

- ‘MarioMaciasMaciasWorks@example.com’ == 12341sdfa12e ?
    - If the user trying to edit gives the credentials matching to said hash value then allow for the posting to be edited, otherwise notify the user of invalid credentials.

# Alternatives Considered

- Alternative markup language
    - Using HTML is better in terms of documentation, especially as we are being introduced to it and having that documentation to do basic functions allows us to spend more time on the core functionality of the wiki.
- Alternative data structure
    - The use of HashMaps would allow us to retrieve data in O(1) time vs an array traversal for the same chunk of data could take O(n) time. Searching for multiple chunks of data would exponentially increase loading times when using linear data structures.

# Resources

- https://www.w3schools.com/TAGS/default.ASP
- https://en.wikipedia.org/wiki/Richard_Ramirez (example of dark history of a city)
- https://developer.mozilla.org
- https://www.geeksforgeeks.org/hash-map-in-python/
- https://www.geeksforgeeks.org/python-hash-method