

A Project Report for Assignment -1 of COP 701

On

Local Markdown wiki and editor

Submitted to

Rahul Narain

(Instructor COP 701)

by

Akshay Pratap Singh (2022MCS2058)

Anurag Krishna Sharma (2022MCS2071)

Nitesh Dohre (2022MCS2070)

Department of Computer Science and Engineering

Indian Institute of Technology, Delhi

Session: 2022-23

CONTENTS

1. INTRODUCTION	
1.1 OBJECTIVE	2
1.2 PROJECT DESCRIPTION	3
2. SYSTEM STUDY	
2.1 TOOLS AND TECHNOLOGY USED	4
2.2 SOFTWARE AND HARDWARE REQUIREMENTS	5
2.3 FUNCTIONAL REQUIREMENTS	5
3. SOFTWARE DESIGN	
3.1 DATA FLOW DIAGRAM	6
3.2 USE CASE DIAGRAM	7
3.3 CLASS DIAGRAM	8
4. PARSER WITH REGEX	
4.1 AUTOMATA AND REGULAR EXPRESSION	9
4.2 PYTHON REGEX LIBRARY	10
4.3 IMPLEMENTATION OF PARSE	10-11
5. SOFTWARE ARCHITECTURE	
5.1 MODEL VIEW CONTROLLER	12-13
5.2 MODULARITY	13
6. ADDITIONAL REQUIREMENT	
6.1 UNIT TESTING	14
6.2 VERSION CONTROL	15
7. RESULTS	16-17
8. CONCLUSION & CONTRIBUTION	18
9. BIBLIOGRAPHY	19

1. INTRODUCTION

1.1 OBJECTIVE

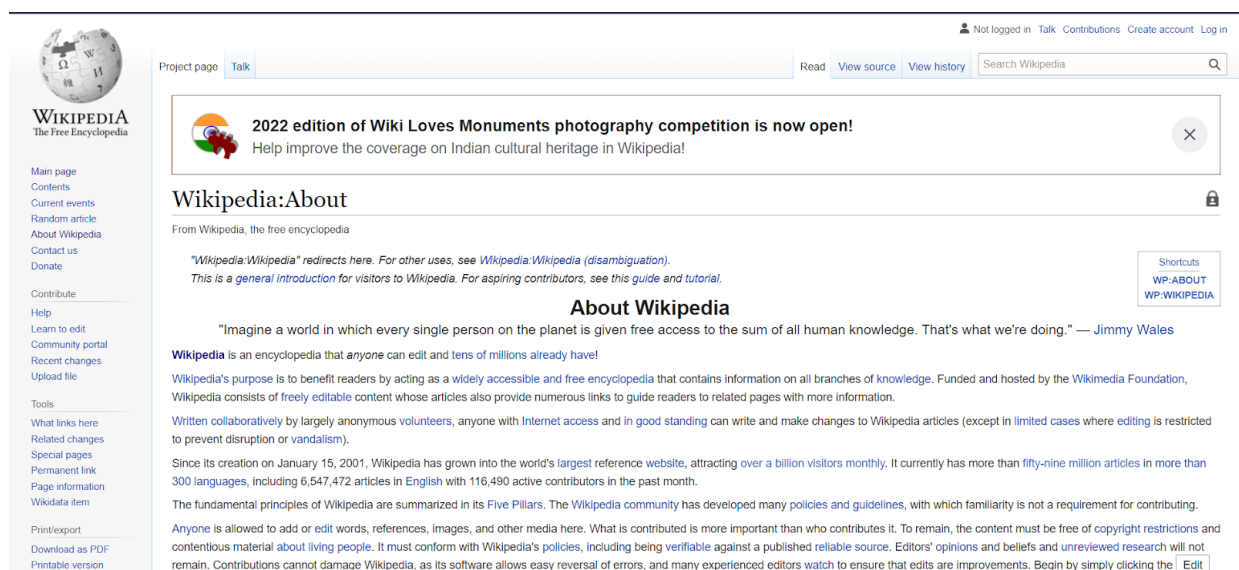
"Imagine a world in which every single person on the planet is given free access to the sum of all human knowledge. That's what we're doing." -

Jimmy Wales

(Founder Wikipedia)

Wikipedia is a multilingual, openly collaborative online information platform. Like the "wikis" that came before it, the online encyclopedia's content is editable by volunteers from across the globe. Wikipedia has thousands of editors, from issue experts to casual fans, who can expand, delete, or change information. This allows for a wide array of data to be supplied and verified about a particular person, place, or thing.

Wikipedia can function as a significant starting point for research, providing users with general information that can be followed up with more legitimate sources outside of the site. For instance, a Wikipedia article may introduce a reader to a particular concept or idea, leading to further exploration of the finer details and the veracity of the claims made.

The image is a screenshot of the Wikipedia homepage. At the top, there's a navigation bar with links like 'Project page', 'Talk', 'Read', 'View source', 'View history', and a search bar. Below this, a banner for the '2022 edition of Wiki Loves Monuments photography competition' is visible. The main heading is 'Wikipedia:About'. Below the heading, it says 'From Wikipedia, the free encyclopedia'. A quote from Jimmy Wales is featured: 'Imagine a world in which every single person on the planet is given free access to the sum of all human knowledge. That's what we're doing.' — Jimmy Wales. The text explains that Wikipedia is an encyclopedia that anyone can edit, funded by the Wikimedia Foundation, and consists of freely editable content. It mentions that Wikipedia has over a billion visitors monthly and more than 50 million articles in over 300 languages. The page also lists the fundamental principles of Wikipedia, such as being free, open, and collaborative, and mentions the 'Five Pillars' of Wikipedia. On the left side, there's a sidebar with links to 'Main page', 'Contents', 'Current events', 'Random article', 'About Wikipedia', 'Contact us', 'Donate', 'Contribute', 'Help', 'Learn to edit', 'Community portal', 'Recent changes', 'Upload file', 'Tools', 'What links here', 'Related changes', 'Special pages', 'Permanent link', 'Page information', 'Wikidata item', 'Print/export', 'Download as PDF', and 'Printable version'. On the right side, there's a 'Shortcuts' box with links to 'WP:ABOUT' and 'WP:WIKIPEDIA'.

(Source - www.wikipedia.com)

1.2 PROJECT DESCRIPTION

In this assignment, we were asked to make an application which is similar to this existing Wikipedia. This application should allow users to create, update an article and Remove and view the article present in the repository in markdown format.

The project is designed as per the task that has features like creating the page, modifying a page, viewing a page and deleting a page. We have implemented two of the additional features Version control and Unit testing in this software assignment to make a modular, robust, and non decoupled manner to ensure the OOPS concepts like Encapsulation, Abstraction.

2. SYSTEM STUDY

2.1 TOOLS AND TECHNOLOGY USED

Python - Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant white space. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming.

Tkinter - Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, Tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with Tkinter is the fastest and easiest way to create a GUI application. Creating a GUI using Tkinter is an easy task.

Regex - A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern. RegEx can be used to check if a string contains the specified search pattern. Python has a built-in package called re, which can be used to work with Regular Expressions.

Python Imaging Library - Python Imaging Library is a free and open-source additional library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.

Visual Studio - Microsoft Visual Studio is an integrated development environment from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps.

2.2 SOFTWARE AND HARDWARE REQUIREMENTS

- Python 2.6, 2.7, or 3.3+ (required)
- Operating system: Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 11,
- Laptop / Desktop – Any Standard Laptop / Desktop

System Requirements The application should be installed into a device, system, or any machine in such a way that it should have basic requirements like supporting software and hardware of the device, accessing inbuilt software, say camera for mobile device, internet permissions, and potential security issues such as a virus or any malware detection. Laptops and Desktops are preferred devices.

2.3 FUNCTIONAL REQUIREMENTS

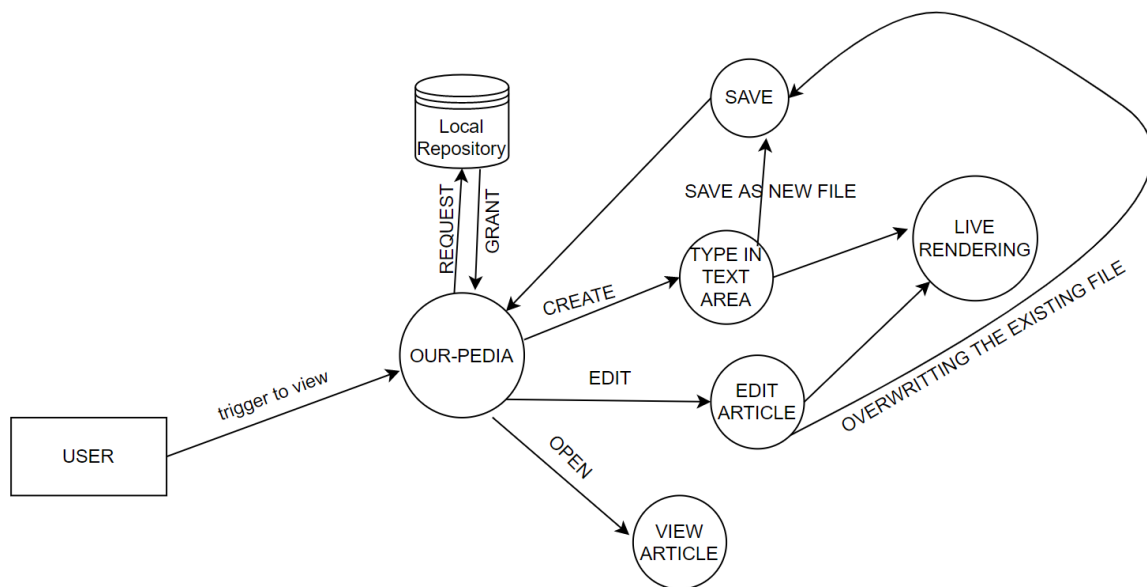
Given below is a list of functional requirements that the application should meet:

- User: only one user at a time can operate the program execution
- user can either create a new file or edit a existing file, both can not be done simultaneously.

3. SOFTWARE DESIGN

3.1 CONTEXT DIAGRAM (DFD) -

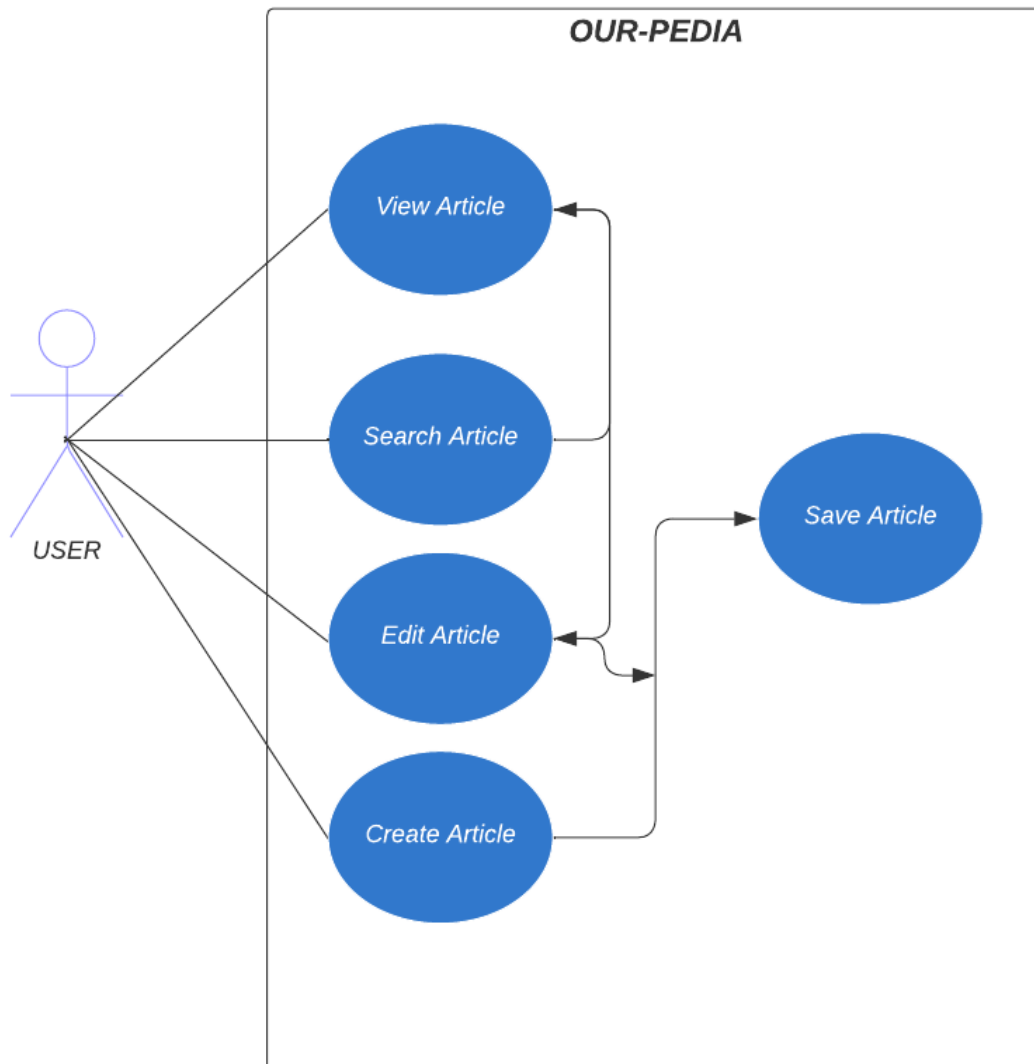
A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.



In the very first step user start the application, OUR-PEDIA GUI window will be displayed, user can open existing files, can edit the files which are already present in the local directory, or user can create a new file by writing in the text window, and save the file into directory into .md file.

3.2 USE CASE DIAGRAM

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

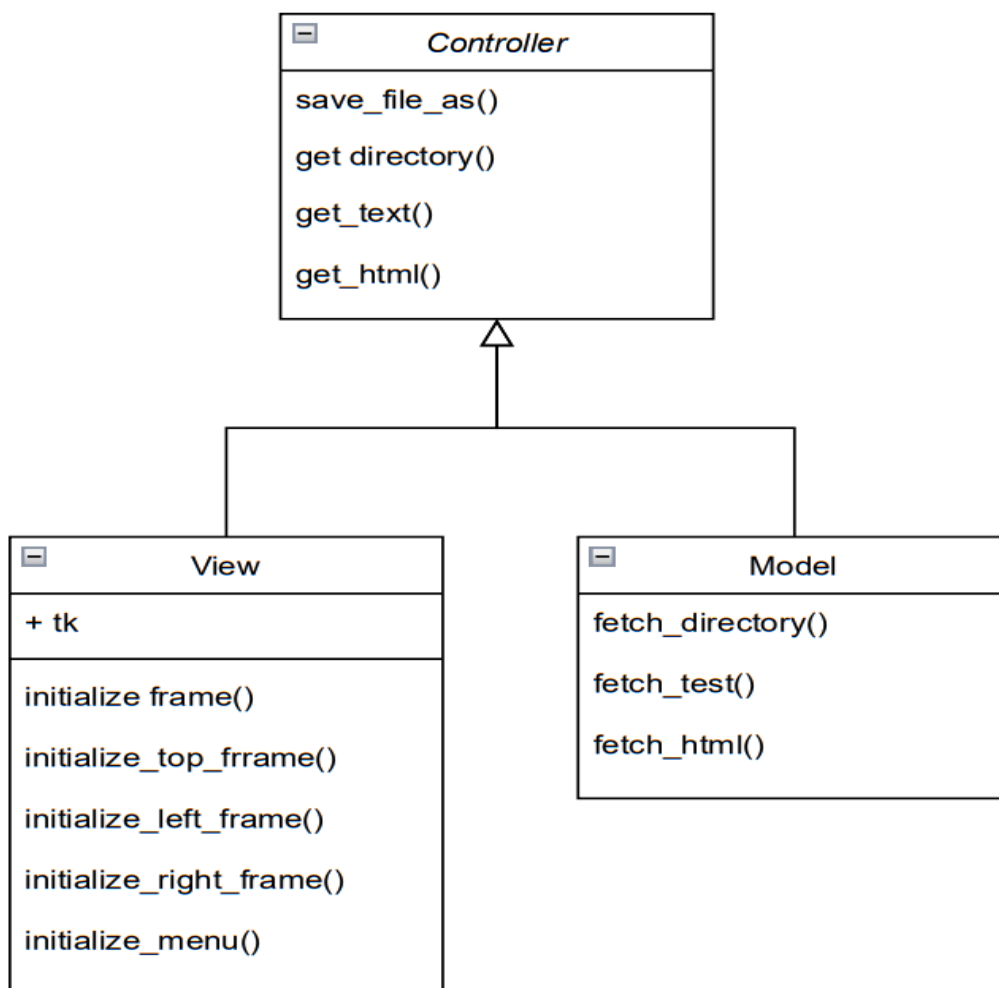


In the OUR-PEDIA application users have use cases like create, update, view, search and delete article in repositories

3.3 CLASS DIAGRAM

Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modeling its classes, attributes, operations, and relationships between objects.

CLASS DIAGRAM OF OUR-PEDIA



In this software assignment we have divided the entire code into three different modules and implemented as a three class with having unique functionalities.

4. PARSER WITH REGEX

4.1 AUTOMATA AND REGULAR EXPRESSIONS

A regular expression is a concept used in programming for a sophisticated pattern matching method, it can recognise and generate a pattern of strings. The parsers, which are a core to communicate with computers, play an important role in converting high-level programming level to a machine understandable command and it does so by certain steps.

The parser takes a formal input and with that into a data structure which is some kind of parse tree or abstract syntax tree which gives them a structural representation, Parser can range from very simple `print()` function for python to complex HTML parsing syntax to an eye-pleasing format with all the front end featuring.

One of the classes of the parser is regular expressions, In other contexts, regular expressions are instead used prior to parsing, as the lexing step whose output is then used by the parser.

In our project, we have used a python library called “regEx” by which we can write our own regular expressions and pattern matching formulae which we have converted into a different data structure “HTML” which then processes the HTML tags and views the final output.

4.2 PYTHON REGEX LIBRARY

RegEx relies on the use of literal characters and meta characters. A metacharacter is any American Standard Code for Information Interchange (ASCII) character that has a special meaning. By using meta characters and possibly literal characters, you can construct a regex for finding strings or files that match a pattern rather than a specific string. For example, in this project, we want to change the heading style from very large to large. We have a markdown syntax `#` for heading which is the biggest and `##` which says not as big as `#`, it's a heading convention. Rather than searching for each specific `#` (that could take forever and be prone to error), we search for the pattern of string that starts on a new line with `#` followed by a space.

Examples of Regex

Square brackets can be used to define a list or range of characters to be found. So:

- ❖ **[abc]** matches a or b or c
- ❖ **[A-Za-z]** matches any uppercase or lowercase letter.
- ❖ **\d** matches any digit.
- ❖ **\s** matches any space.
- ❖ **^** asserts the position at the start of the line.
- ❖ **\$** asserts the position at the end of the line.

4.3 IMPLEMENTATION OF PARSER

This project has implemented the parser in three steps

- a. Pattern matching basic markdown syntax
- b. Creating a ADS (HTML tags equivalence of markdown)
- c. Viewing of HTML tags through GUI

(Source - <https://librarycarpentry.org/lc-data-intro/01-regular-expressions/>)

44 lines (31 sloc) | 1.09 KB

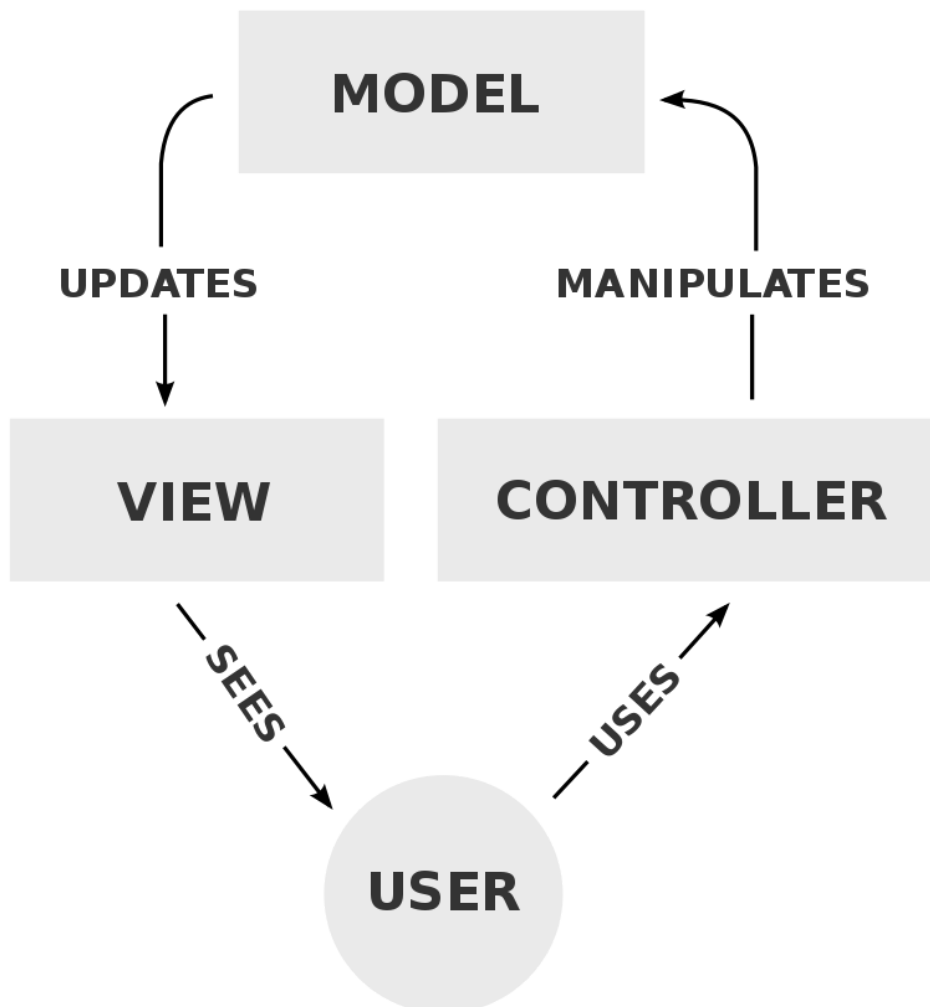
```
1  import regex as re
2
3  def reg(result):
4      #heading h1
5      regex_h1 = r"^(#\s)(.*)"
6      subst_h1 = "<h1>\2</h1>"
7
8      #heading h2
9      regex_h2 = r"^(##\s)(.*)"
10     subst_h2 = "<h2>\2</h2>"
11
12     #heading h3
13     regex_h3 = r"^(###\s)(.*)"
14     subst_h3 = "<h3>\2</h3>"
15
16     #bold and italics
17     regex_ib = r"(\*\s*)(\b)([^\s]*)(\b)(\*\s*)"
18     subst_ib = "<em><b>\3</b></em>"
19
20     #bold
21     regex_b = r"(\*\s*)(\b)([^\s]*)(\b)(\*\s*)"
22     subst_b = "<b>\3</b>"
23
24     #italics
25     regex_i = r"(\s*)(\b)([^\s]*)(\b)(\s*)"
26     subst_i = "<em>\3</em>"
27
28     #ordered list
29     regex_ol = r"^( \d+ \. \s+ )(.)"
30     subst_ol = "<li>\1</li>"
31
```

5. SOFTWARE ARCHITECTURE

5.1 MODEL VIEW CONTROLLER

Model–view–controller (MVC) is a software architectural pattern commonly used for developing user interfaces that divide the related program logic into three interconnected elements. This is done to separate internal representations of information from the ways information is presented to and accepted by the user.

Traditionally used for desktop graphical user interfaces (GUIs), this pattern became popular for designing web applications. Popular programming languages have MVC frameworks that facilitate the implementation of the pattern.



Model : The central component of the pattern. It is the application's dynamic data structure, independent of the user interface. It directly manages the data, logic and rules of the application.

View : Any representation of information such as a chart, diagram or table. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.

Controller : Accepts input and converts it to commands for the model or view.

(Source: <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>)

5.2 MODULARITY

The module simply means the software components that have been created by dividing the software. The software is divided into various components that work together to form a single functioning item but sometimes they can perform as a complete function if not connected with each other. This process of creating software modules is known as **Modularity** in software engineering.

In the development of OUR-PEDIA we have achieved modularity by implementing the concept of object oriented programming of python and by implementing architecture Model View Controller. Using MVC we have separated the entire software into 3 parts where the module View.py is responsible for all the GUI functions like initializing the Frames, initializing the menu and displaying the articles etc.

Second module Model.py is used to fetch the data from the local repository and fetching the text and converting it into the html.

And the third module controller.py bridge between user model and view. Controller is the module which is responsible for getting the data from database using module Model.py and displaying it into the window using the module View.py

6. ADDITIONAL FEATURES

6.1 UNIT TESTING

Unit Testing is a software testing technique by means of which individual units of software, i.e. group of computer program modules, usage procedures, and operating procedures are tested to determine whether they are suitable for use or not.

It is a testing method using which every independent module is tested to determine if there is an issue by the developer himself. It is correlated with the functional correctness of the independent modules.

The objective of Unit Testing is:

1. To isolate a section of code.
2. To verify the correctness of the code.
3. To test every function and procedure.
4. To fix bugs early in the development cycle and to save costs.
5. To help the developers to understand the code base and enable them to make changes quickly. To help with code reuse.

(source- <https://www.geeksforgeeks.org/unit-testing-software-testing/>)

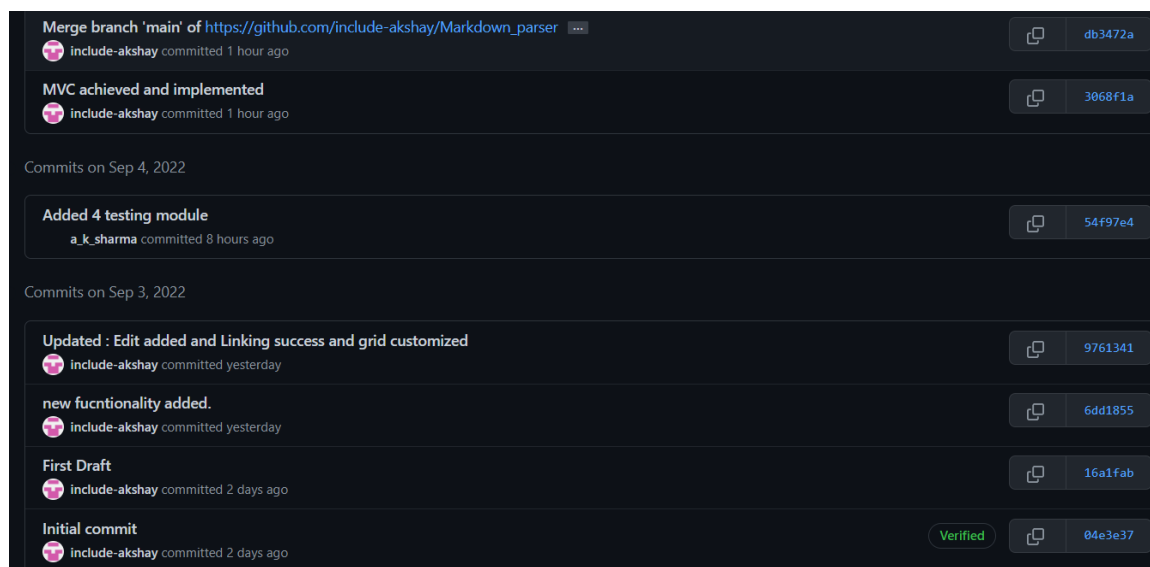
```
1 import unittest
2 import md_to_html
3
4 class TestCalc(unittest.TestCase):
5
6     def test_h1(self):
7         self.assertEqual(md_to_html.reg("# Head1"), "<h1>Head1</h1>")
8         self.assertEqual(md_to_html.reg("# SomefirstHeading"), "<h1>SomefirstHeading</h1>")
9
10    def test_h2(self):
11        self.assertEqual(md_to_html.reg("## Heading is 2"), "<h2>Heading is 2</h2>")
12        self.assertEqual(md_to_html.reg("## SomeText"), "<h2>SomeText</h2>")
13
14    def test_bold(self):
15        self.assertEqual(md_to_html.reg("***Heading is 2***"), "<b>Heading is 2</b>")
16        self.assertEqual(md_to_html.reg("***SomeText***"), "<b>SomeText</b>")
17
18    def test_italic(self):
19        self.assertEqual(md_to_html.reg("*Heading is 2*"), "<em>Heading is 2</em>")
20        self.assertEqual(md_to_html.reg("*SomeText*"), "<em>SomeText</em>")
21
22
23
24
25 if __name__ == '__main__':
26     unittest.main()
```

6.2 VERSION CONTROL

Version control, also known as source control, is the practice of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time.

Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.

(source-<https://www.atlassian.com/git/tutorials/what-is-version-control>)



The following result obtained from the model of Local Markdown wiki and editor aka OUR-PEDIA

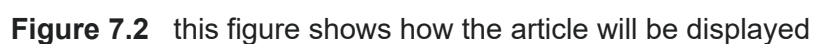
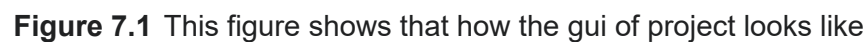




Figure 7.3- In this figure it is shown that, how will it get edited

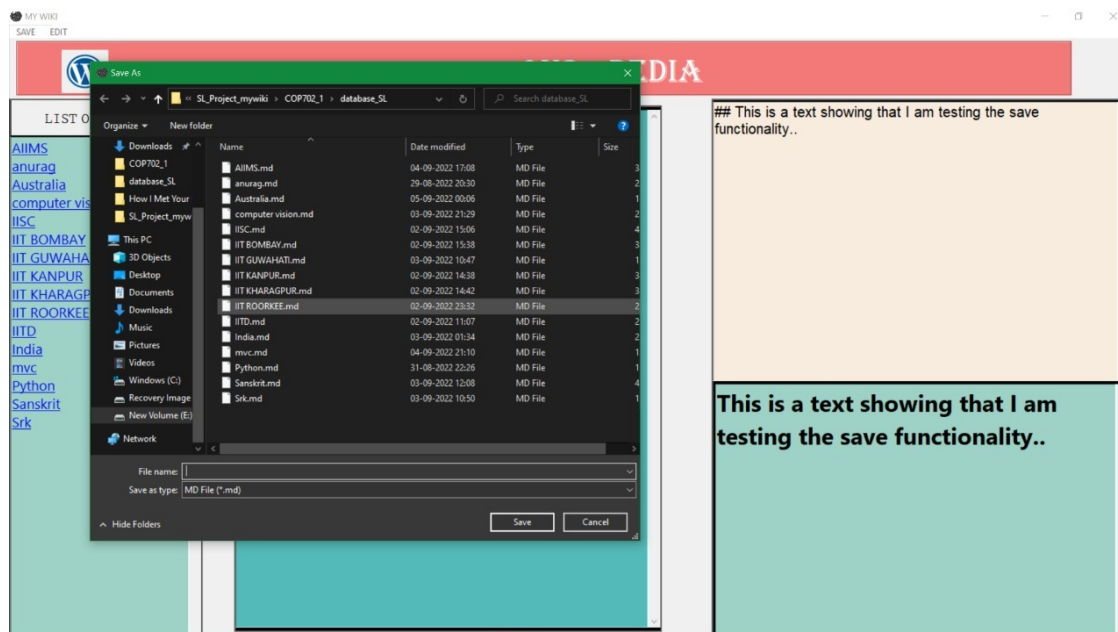


Figure 7.4- Saving the edited or newly created file into the directory using save button

8. CONCLUSION AND CONTRIBUTION

This project is a prototype of a very useful application which helps millions of readers everyday with information and knowledge in every field possible. We developed this application with a similar motive and while developing this we learned a lot about software architectural design, testing, implementation of grammars, version control and so on.

We were a team of 3 and we worked closely throughout the project sharing ideas and shooting them down and brainstorming again.

Akshay did the software design component, he designed and developed the MVC architecture for our application, he extensively researched the tkinter library and modified it to our needs.

Nitesh contributed with the version control, testing and the formation of this documentation, along with Akshay he helped design the UI and layout.

Anurag played a role in regex formation, parsing and bits of testing and git version control.

9. BIBLIOGRAPHY

[1] Links referred

- www.wikipedia.com
- <https://www.geeksforgeeks.org/unit-testing-software-testing/>
- <https://docs.python.org/3/library/tkinter.html>
- <https://www.w3schools.com/python/default.asp>
- <https://librarycarpentry.org/lc-data-intro/01-regular-expressions/>

[2] Books referred

- Python Programming - Kiran Gurbani
- Learning Python - Mark Lutz
- Spinellis & Gousios (eds.), *Beautiful Architecture: Leading Thinkers
Reveal the Hidden Beauty in Software Design*

[3] YouTube Channels referred

- CS Dojo
- Edureka
- codewithharry
- codemy.com