# INTRODUCTION

## OBJECTIVE

The Fortress of gamers is a web site developed for the sole purpose of providing gamers withs information that they will and might need to understand and play the game better. Gamers can find all the information about all the games in one single website. It allows users to insert information about new games and also allows them to update the records. When required the records can also be deleted. The user will be able to easily search the details provided the game name (primary key) is known. It allows for modification of any detail of an already present game and stores the newly entered content. In case if the details are not being found during the search, modify or delete, appropriate messages are displayed.

## SCOPE OF THE PROJECT

The project aims at developing a mini project that allows the end user to store information about different games, in a file with all the necessary details along with it. A file system should be developed to store the names and details of all the topics. The details must include the name of the game, genre of the game and writer of the game in one file and other information like publisher and release date of the game. The record of the required game must be displayed when required. A user must be able to add a new game and the necessary details. User must be able to view the record her or she desires by searching for it. If the record is present, it must be displayed. If it is not present, appropriate message should be displayed. Option to delete a record must also be provided. If the record is present, it must be deleted, and the space must be reclaimed. If it is not, an appropriate message should be displayed.

## MOTIVATION

I got the inspiration for developing the fortress of gamers website when I was searching for information about a game I was playing. There are many major game publisher websites that provide information about their own games and going through them is really a big burden. The need to be able to get information about all the games from a single source seemed extremely necessary.

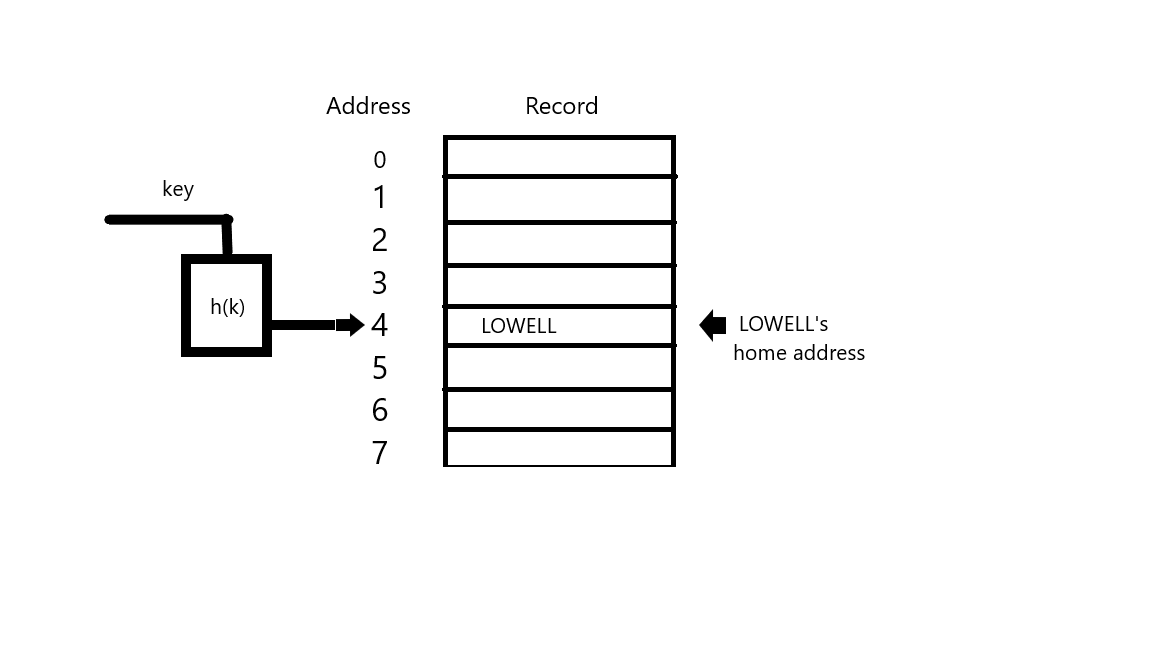
# METHODOLOGY

## HASHING

A hash function is like a black box that produces an address every time you drop in a key. More formally, it is a function h(K) that transforms a key K into an address. The resulting address is used as the basis for storing and retrieving records. In Fig. 2. 1, the key LOWELL is transformed by the hash function to the address 4. That is, h(LOWELL) = 4. Address 4 is said to be the home address of LOWELL. Hashing is like indexing in that it involves associating a key with a relative record address. Hashing differs from indexing in two important ways:

* With hashing, the addresses generated appear to be random-there is no immediately obvious connection between the key and the location of the corresponding record, even though the key is used to determine the location of the record. For this reason, hashing is sometimes referred to as randomizing.
* With hashing, two different keys may be transformed to the same address so two records may be sent to the same place in the file. When this occurs, it is called a collision and some means must be found to deal with it.

Consider the following simple example. Suppose you want to store 75 records in a file, where the key to each record is a person's name. Suppose also that you set aside space for 1,000 records. The key can be hashed by taking two numbers from the ASCII representations of the first two characters of the name, multiplying these together, then using the rightmost three digits of the result for the address. Table 10.1 shows how three names would produce three addresses. Note that even though the names are listed in alphabetical order, there is no apparent order to the addresses. They appear to be in random order.



**Fig 2.1** **Hashing the key LOWELL to address**

## TOOLS

### Python

**Python** is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

### Flask

**Flask** is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries.  It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program

### HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. HTML elements are the building blocks of HTML pages.

The definition of HTML is Hypertext Markup Language:

* Hypertext is the method by which one can move around on the web-by clicking on special text called hyperlink, which brings to the next page. The fact that it is hyper just means it is not linear-i.e., one can go to any place on the Internet whenever they want by clicking on links-there is no set order to do things in.
* Markup is what HTML tags do to the text inside them. They mark it as a certain type of

text (italicized text, for example).

* HTML is a Language, as it has code words and syntax like any language.

**CSS**

**Cascading Style Sheets** (**CSS**) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity.

**BOOTSTRAP**

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype the user’s idea or build an entire app with the help of Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

# SYSTEM REQUIREMENTS SPECIFICATION

## USER REQUIREMENTS

The Fortress of gamers is a single source website that provides users with all required information so that they will able to enjoy the game better. The website provides features like searching the game, updating the game record and if required deleting it. The website is designed with the sole purpose of making gamers feel hyped and comfortable about the user interaction.

## SOFTWARE REQUIREMENTS

* Operating System: Windows
* IDE: PyCharm Community Edition 2019.3.3 x64
* Technology: Python
* Web Technologies: Html, CSS

## HARDWARE REQUIREMENTS

* Hardware: Dual Core
* Hard Disk: 50 GB
* Speed: 1.4 GHz
* RAM :1GB
* Key Board: Standard Keyboard
* Touch Pad: Button Mouse
* Monitor: LED

## FUNCTIONAL REQUIREMENTS

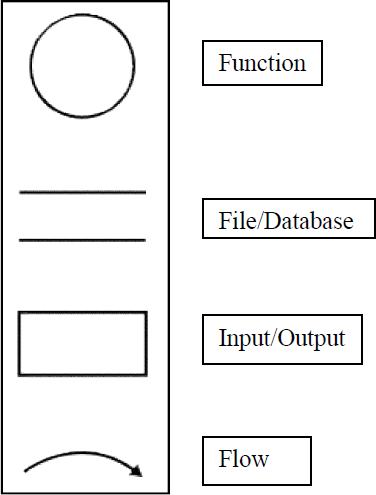
* + - Any new record inserted should be inserted into the original file.
    - The owner should be able to insert new records with all the required details without any consequences.
    - The application should not allow insertion of records with duplicate value.
    - User should be able to add, delete, view, search and modify the contents of the original file by using the key.
  1. **NON-FUNCTIONAL REQUIREMENTS**
     + The application should display appropriate messages to the user if something goes wrong like inserted duplicate records
     + The application should not crash.
     + The user should be able to understand the facilities the application is providing to the user.
     + The application should be user friendly.

# SYSTEM DESIGN AND DEVELOPMENT

Hashing is the process of converting a given key into another value. A hash value is used to generate new value according hash function. It is a computationally and storage space efficient form of data access which avoids the non-linear access time of ordered and unordered lists and structured trees and the often-exponential storage requirements of direct access of state spaces of large variable length keys.

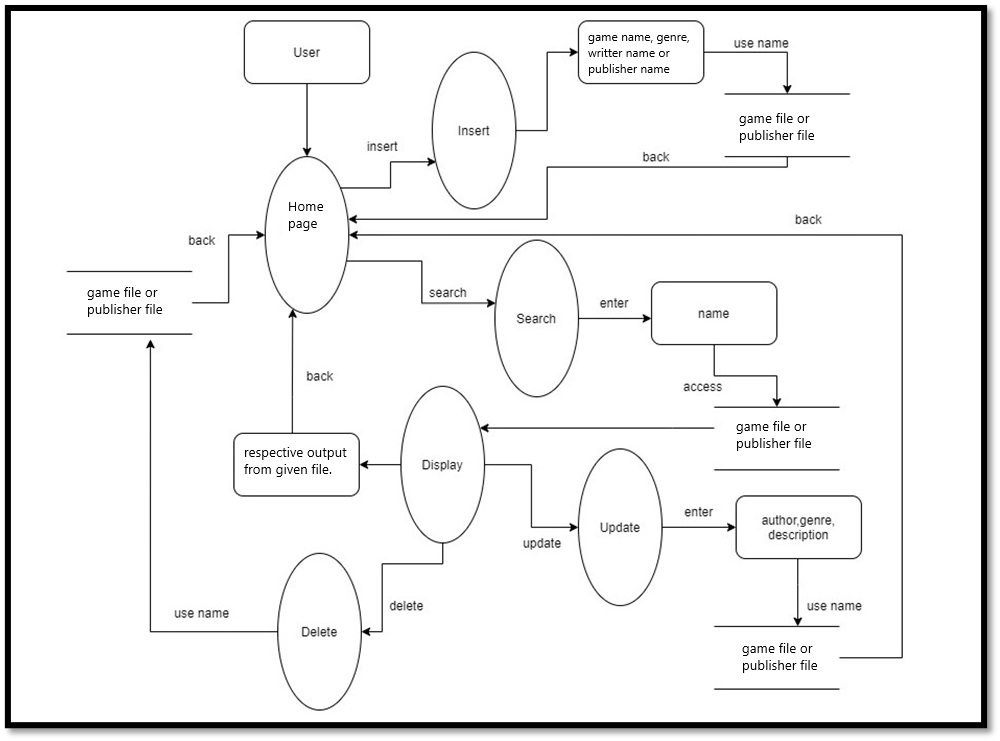
## ARCHITECTURAL DESIGN

There are different notations to draw data flow diagrams, defining different visual representations to processes, data stores, data flow, and external entities that is shown in Figure 4.1



**Fig 4.1 – Data Flow Diagram Notations**

The Figure 4.2 shows the flow of the mini project that the user will experience. Based on the Figure 2.3, System user has three options in the menu page-insert, search and display. When the user opts for insert option, the details are taken from the user and insert function is implemented on it. The primary key is sent to the hash function which will produce a hash value. All details along with the primary key is entered into the data file by seeking the hash value as position in the file. After which, the current record information is displayed on the gui. Similarly, when user opts for search option the name is taken from the user and search function **is** implemented on it. The name is sent to the hash function and respective hash value is produced using which the file is seeked for the position where the record is looked up on and its respective details are displayed. There the user has the option to either delete the current record or update the current record. If the user opted for update, the corresponding details has to be provided for the record, then the update function is implemented on it and the updated information of the current record is displayed on the gui. The same works for delete. The game details of the entered game name is deleted from the file and the user is redirected to the home page.



**Fig 4.2** **Architectural diagram of Current trends**

# IMPLEMENTATION

**5.1 LIST OF MODULES**

The project is implemented in Python programming language. The project is divided into 3 modules-

1. The functions module- This module implements the core and the additional functionality of the file.

2. The web application module implements the logic for handling user requests received over the web.

3. The UI module contains the front-end UI design and styling.

**5.2 MODULE DESCRIPTION**

**1. functions module:**

The functions model implements the main logic and handling operations needed to perform file structure operations. This model consists of functions.py file that contains all the logic functions like insert, update, delete and search functions including the hash function.

**2. Web application module:**

The web application module(app.py) implements the connection to functions.py, processes user requests received over the web and replies to them. This module is implemented in Flask, a micro-web framework for Python. The units of this module are represented as routes and their respective views, where the route describes the format of user request and the views implement the corresponding processing operations and response mechanism.

**3. The UI module:**

The UI module is divided into templated and styles. The templates contain HTML web page to be sent in response to the respective user requests. Styles contain the styling parameters for the web pages designed in templates.

dsafdsafdsafdsaffdafdsafffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

**5.3 ALGORITHM**

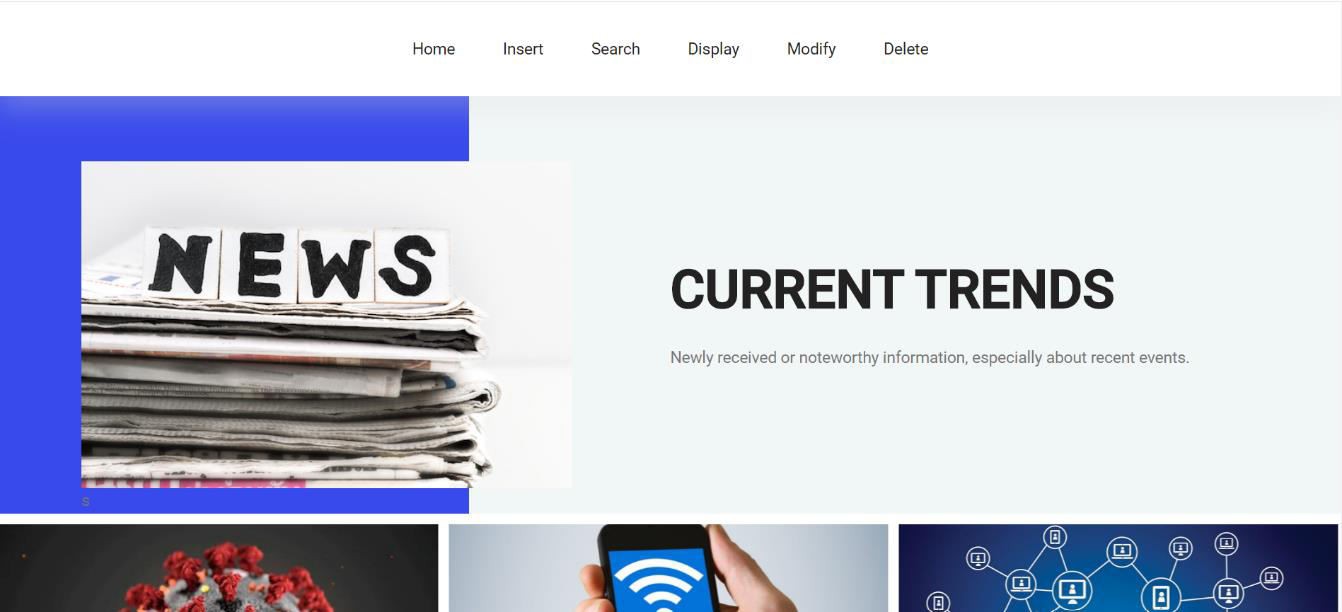
1. **Insert:** The insert algorithm accepts the three data that has to be inserted in the file along with the file name in which the file has to be inserted. The primary key (game name) is sent to the hash function and the position is calculated and seeked in the respective file. The other details, along with the primary key are added to the file at obtained position.
   * 1. Start.
   * 2. Input the details.
   * 3. Compute the position by sending game name to the hash function.
   * 4. Open the respective file and seek the obtained position.
   * 5. Check if the position in the file is free.
   * 6. If the position is free insert the new record in the file after padding.
   * 7. If the position is occupied find if the record is a duplicate.
   * 8. If the record is duplicate send appropriate message.
   * 9. Else if the record is not duplicated and there already exists a record.
   * 10.Then go to next hash location and repeat the above steps.
   * 11.End
2. **Search**: The search function accepts the game name as primary key and then searches the respective file for the required data.
   * 1.Start.
   * 2.Input the Game name (Primary Index)
   * 3.Send the primary key to the hash function.
   * 4.Open the respective file and seek the location.
   * 5.If a record exists in the obtained location check if the record has same primary key
   * 6.If the record is found display the same if not display appropriate message.
3. **Delete:** The delete function accepts the Game name user is looking for. It then searches the respected file for the Game name specified by the user. If present, then the record details are deleted from the file. Subsequently, the space is reclaimed in the memory. If it is not, then appropriate message is displayed for the absence of the record.
   * Start.
   * Input: Game name (Primary Index).
   * Open respective file in read mode.
   * Search for the record.
   * If the record exists the record is deleted and the space is reclaimed.
   * If the record is not found then appropriate message is displayed.
   * End.
4. **Display:** Display function fetches the data from the respective file and displays it on the user interface.
   * Start
   * Read the required file from the respective file.
   * Display the same.
5. **Modify:** The modify function accepts the Game name the user is looking for. It then searches the file for the game name specified by the user. If present, then the record details are deleted from the file. Subsequently, the space is filled with the new record the user enters. If it is not, then appropriate message is displayed for the absence of the record.
   * Start.
   * Input: Game name(Primary Index).
   * Search the file for the record.
   * If found delete the record.
   * Insert the new modified record in the same location
   * Display the new record.
   * End

**CHAPTER 6**

**RESULTS AND DISCUSSIONS**

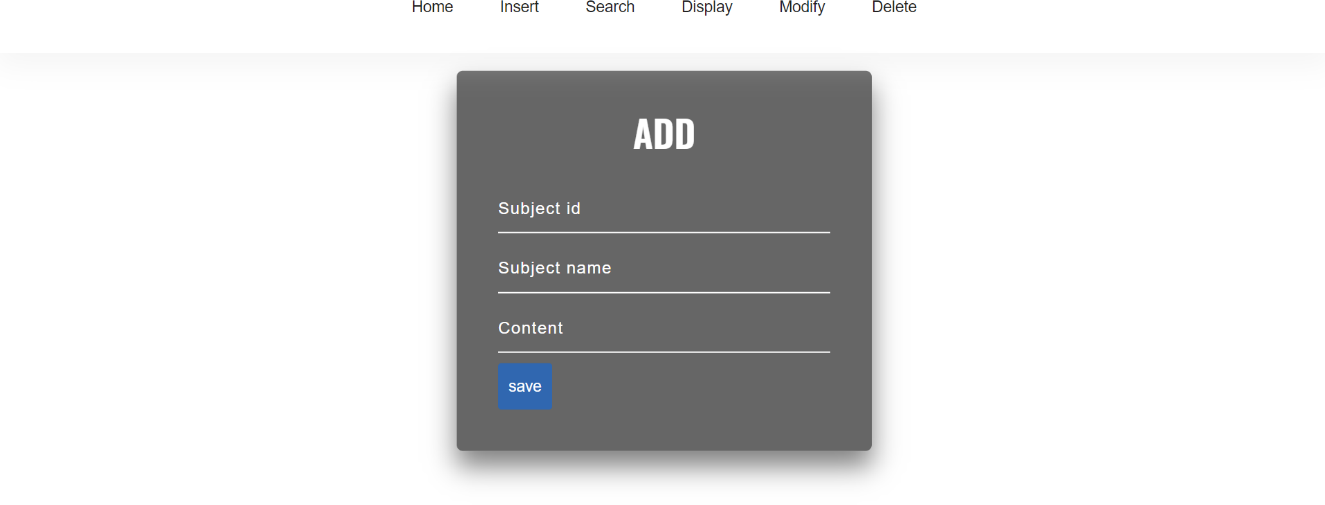
## SNAPSHOTS OF THE PROJECT AND DESCRIPTIONS

In computer systems, a snapshot is the state of a system at a particular point in time. The term was coined as an analogy to that in photography. It can refer to an actual copy of the state of a system or to a capability provided by certain systems.



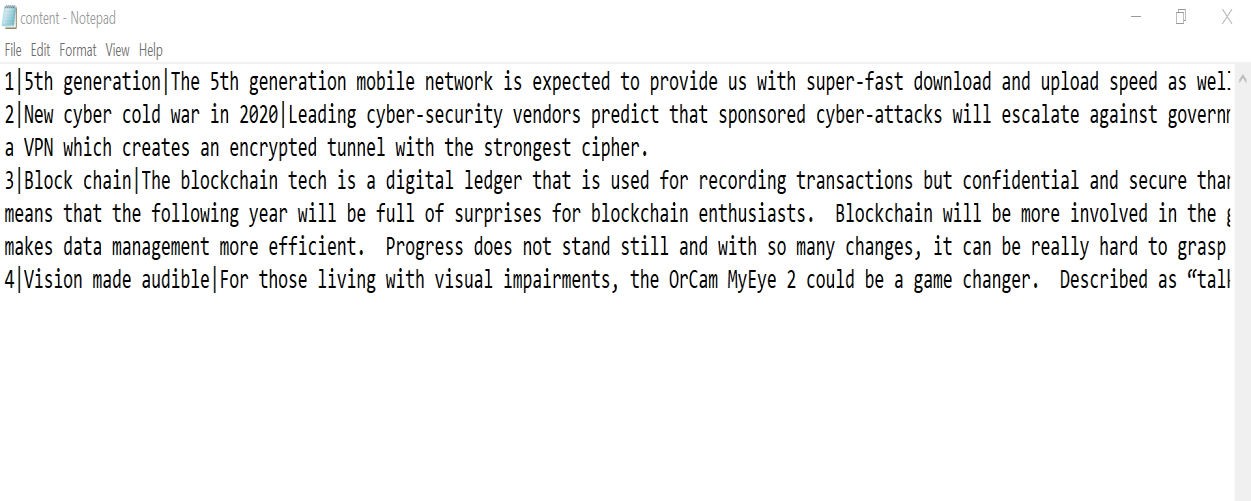
**Fig 6.1.1: Home page**

Home page describes the topic of the project that is “current trends” and the options that are available for the user to work on the topic along with contact details.



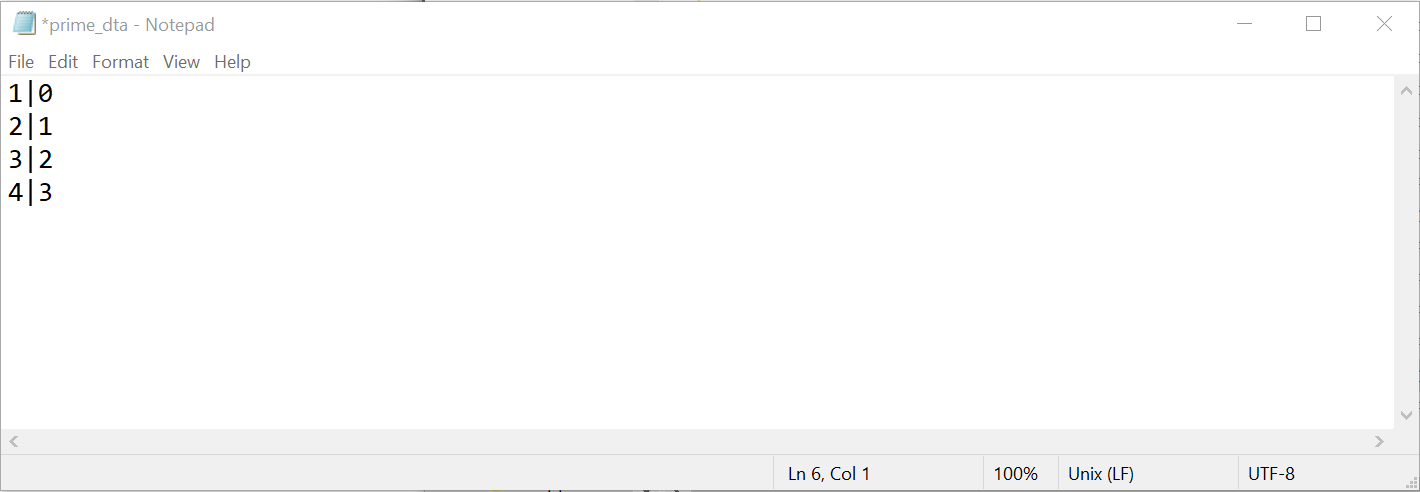
**Fig 6.1.2: Insert page**

This page allows the user to add the addition information regarding to current trends.



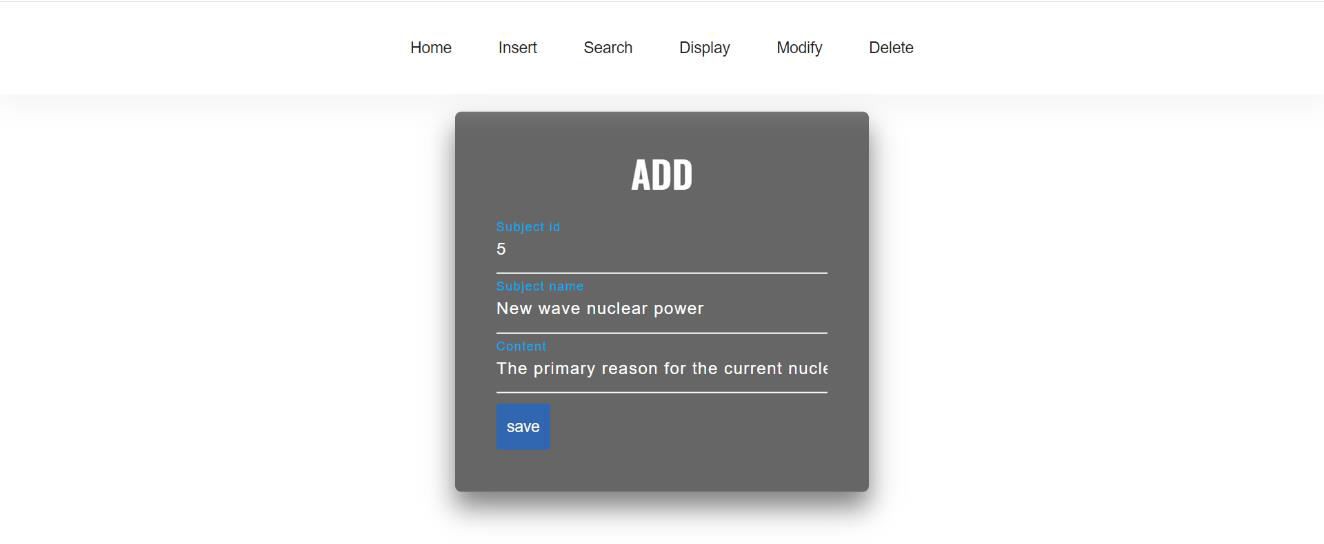
**Fig 6.1.3: Content file**

This page contains some of the topics of trending information, and this is before adding the new information to the file by the user.



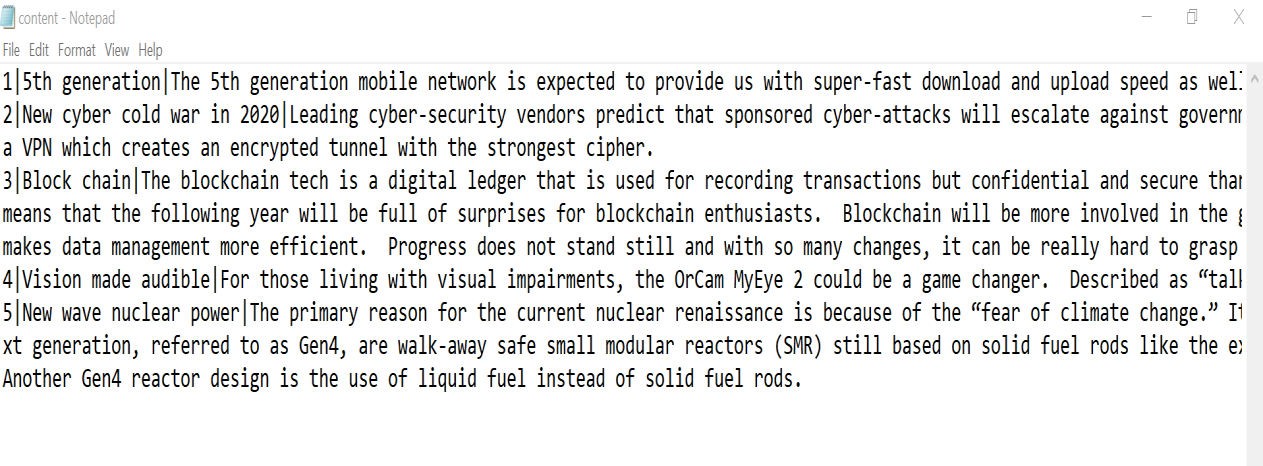
**Fig 6.1.4: Primary index file**

This page is the index file which has subject id and relative record number of the information/records that are present in the content file.



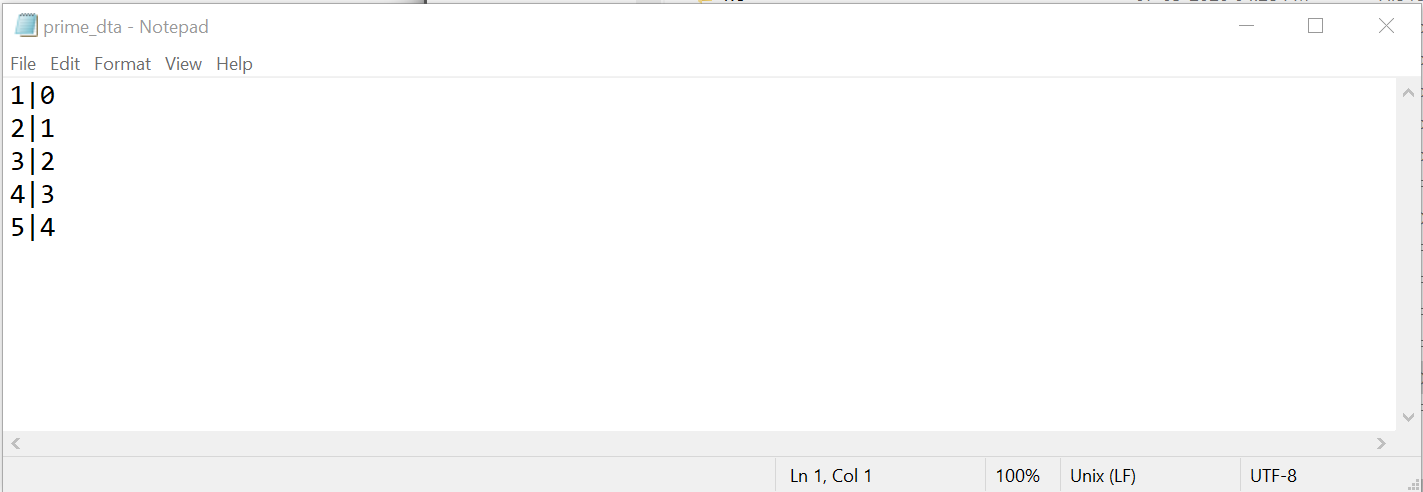
**Fig 6.1.5: Insert page**

In this page user is entering the additional information by giving subject id, subject name, and content after clicking save, the information will be saved to the content file.



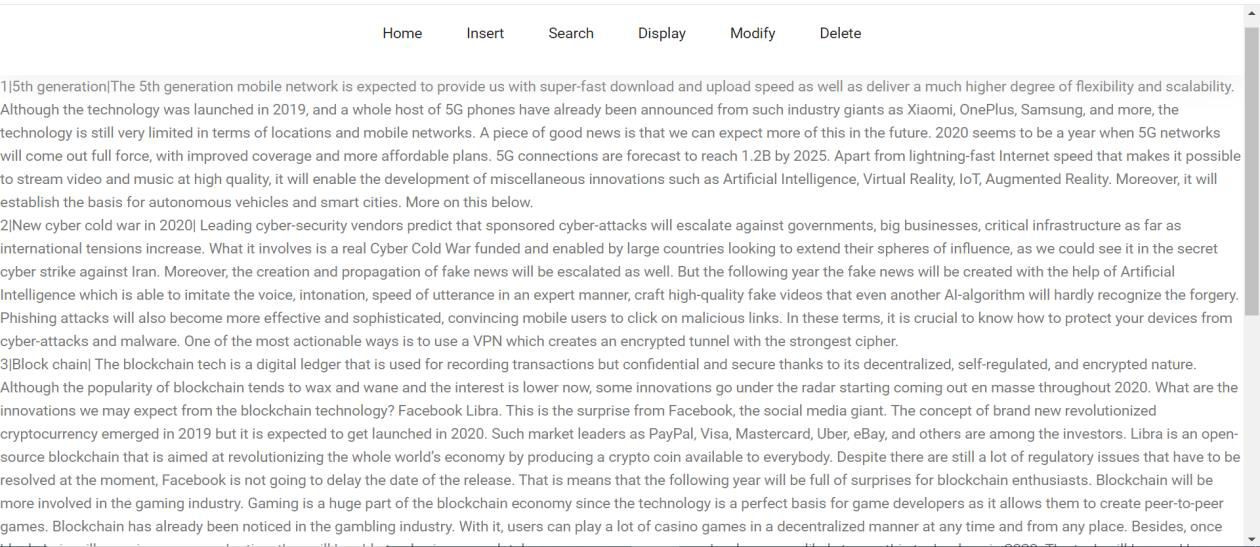
**Fig 6.1.6: Content file**

This page contains the fifth record information which was added by the user in the front end insert page, content file is modified.



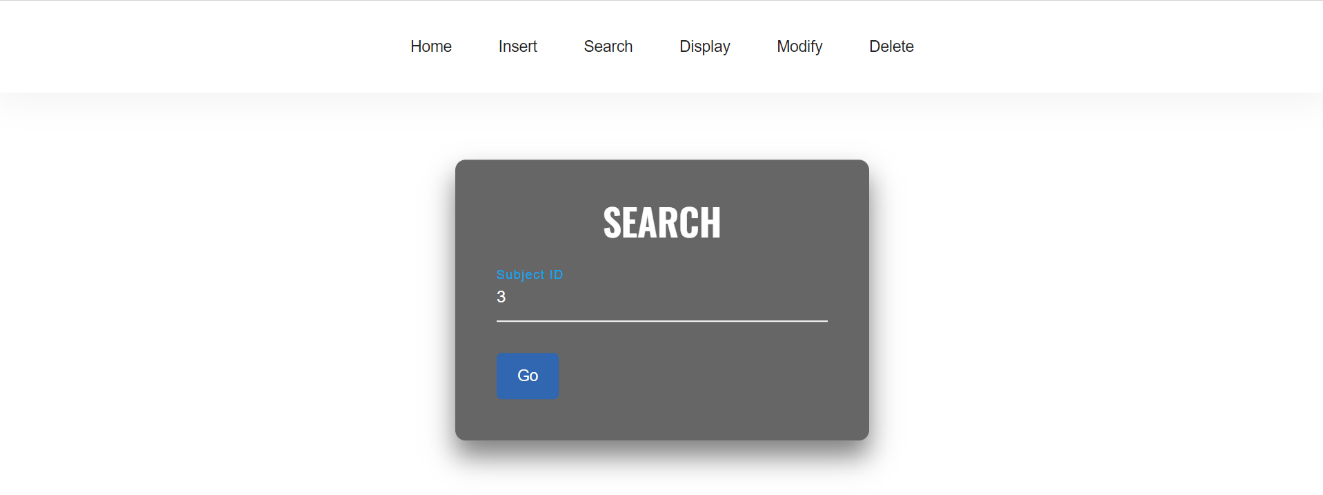
**Fig 6.1.7: Primary index file**

This page contains the new rrn of the primary index after inserting corresponding the new record by the user.



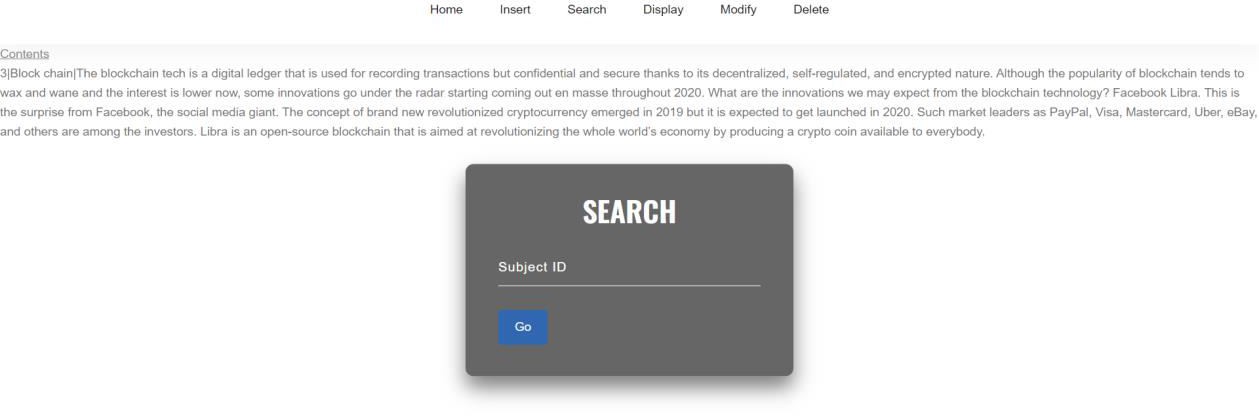
**Fig 6.1.8: Display page**

This page displays the contents that are present in the file, when user clicks on the display function.



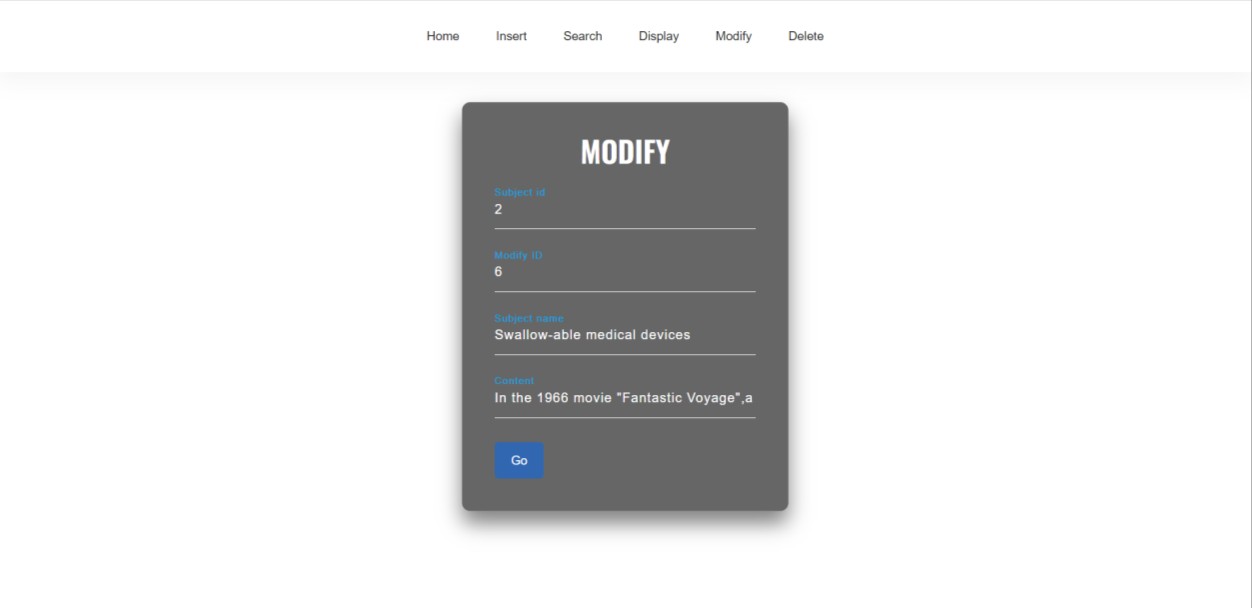
**Fig 6.1.9: Search page**

This page is used to search for the particular information required by the user. Search method takes subject id to fetch that record.



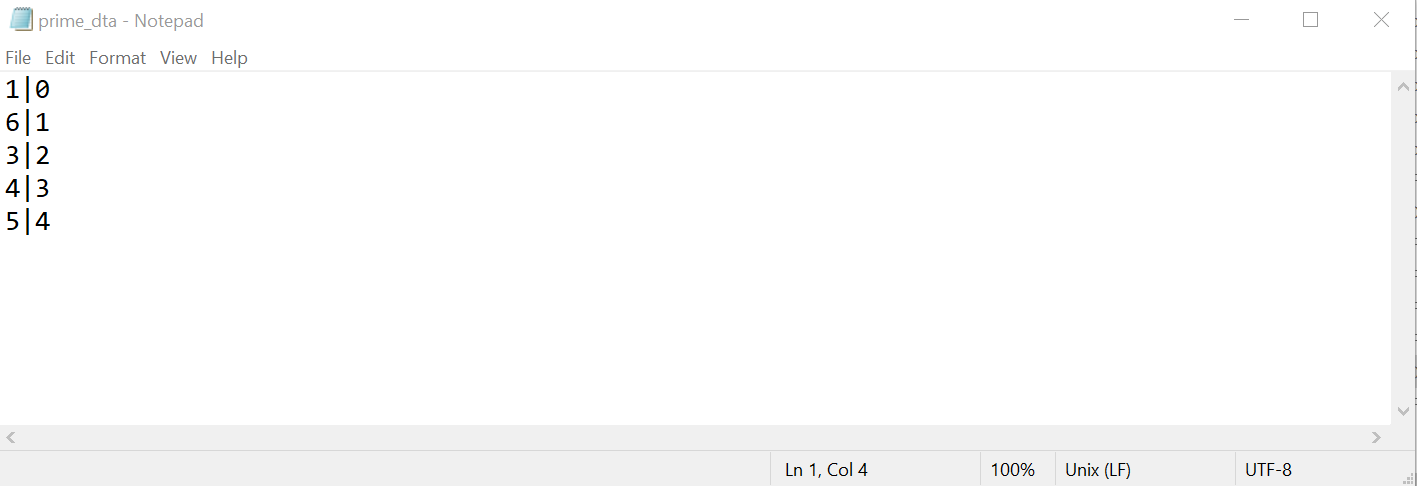
**Fig 6.1.10: Search page**

After entering the subject id by the user corresponding record details will be displayed on the screen.



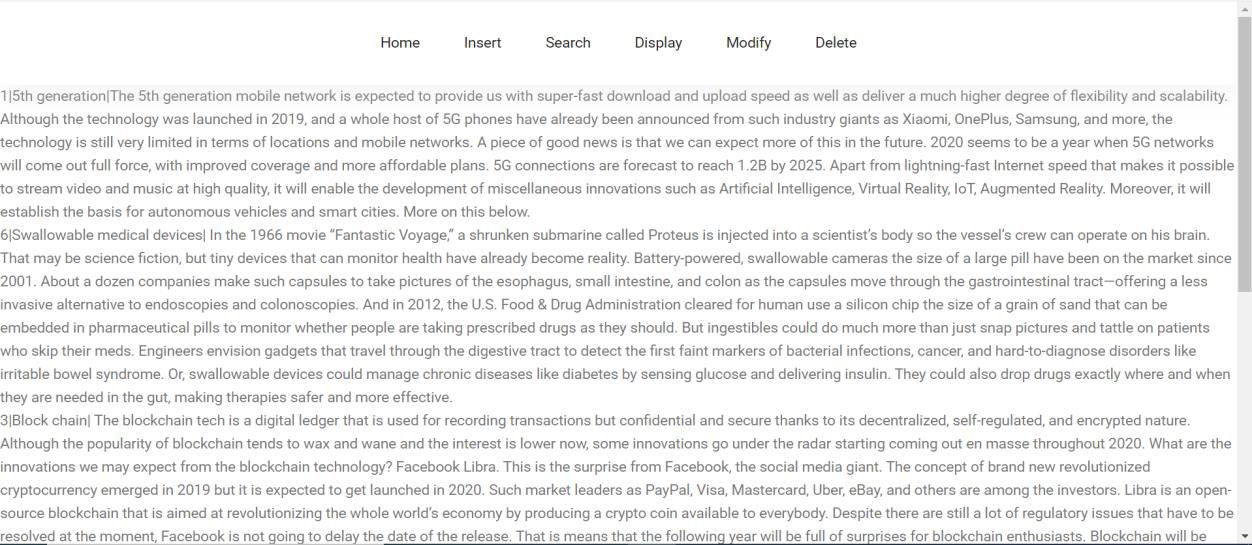
**Fig 6.1.11: Modify page**

This page is used to modify the particular record in the file in case if user has inserted any wrong or irrelevant information or to modify the same record with additional information.



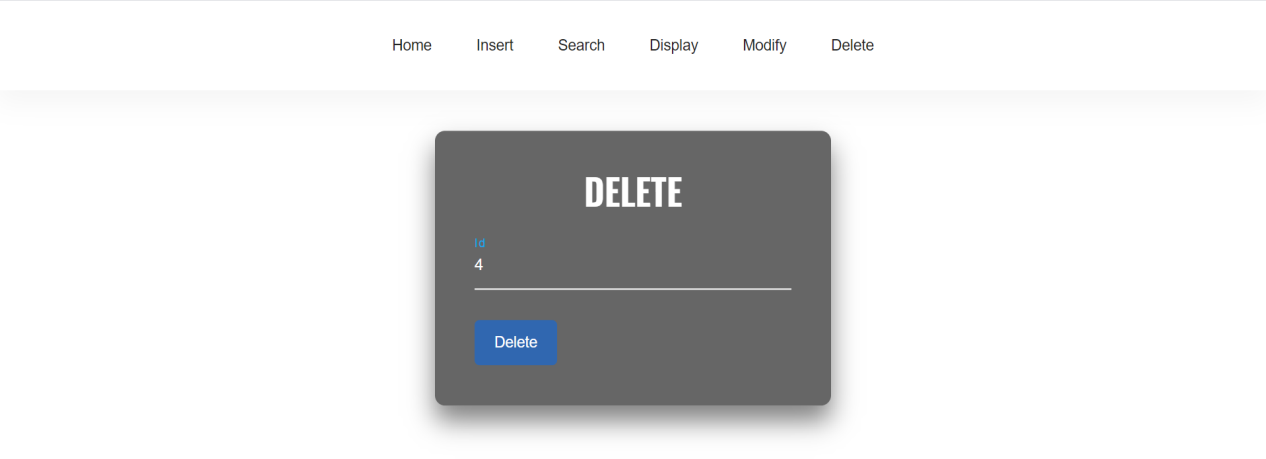
**Fig 6.1.12: Primary index file**

This file describes the subject id and rrn(relative record number) that has changed after the modification that has been made by the user.



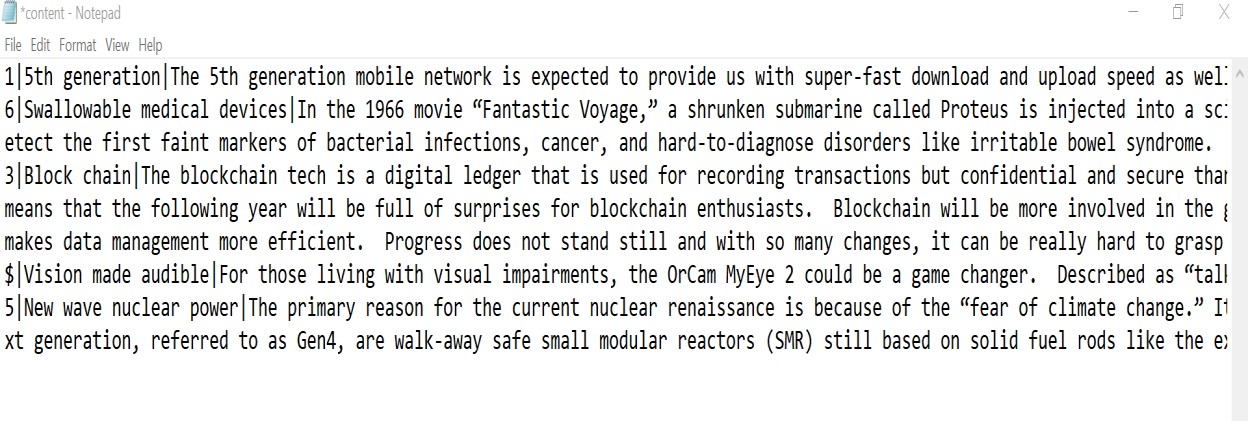
**Fig 6.1.13: Display page**

This page shows the content that is present in the file after the modification made by the user on the previous file.



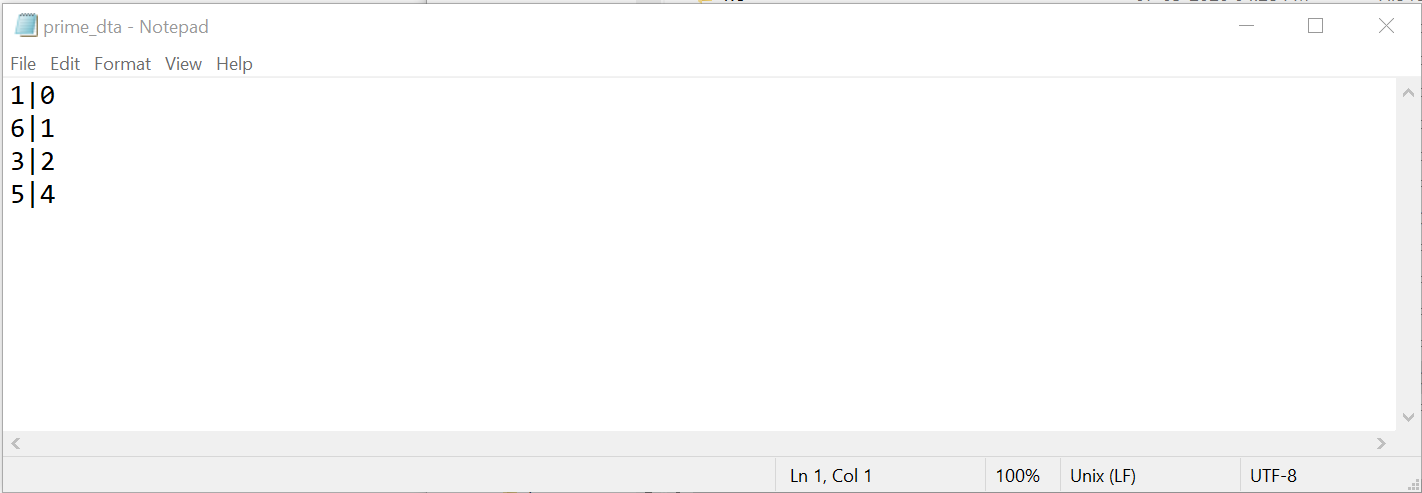
**Fig 6.1.14: Delete page**

This page is used to implement delete operation by entering the particular subject id that has to be deleted by the user.



**Fig 6.1.15: Content file**

This page tells the user that after deletion operation the subject id is represented by the special symbol $ which indicates that the record has got deleted.



**Fig 6.1.16: Primary index file**

This page tells the user that after deletion that particular subject id from the index file has been removed that is it no longer has the record to perform operations on it.

## OBSERVATION ABOUT PROJECT

The project is observed to correctly perform the operations as of user requirements like search, add, display, delete, modify. The details when entered by the user for addition is appended to the respective file. The user can search any game data by entering the game name in the search tab. If the search is successful then the details of the game is displayed. The user can view 2 options in the display page to update the same data or delete the same. When the user chooses to delete the record is deleted form the respective file and the user is redirected to the home page. If the user chooses to update, the user is redirected to the update page where the user is required to provide with new details. Once the details are provided the user is redirected to the display page.

**6.3 ADVANTAGES OF HASHING:**

• Hashing provides a more reliable and flexible method of data retrieval than any other data structure.

• It is faster than searching arrays and lists.

• Hash tables are particularly efficient when the maximum number of entries. • Storage is reduced.

**6.4 DISADVANTAGES OF HASHING:**

* Performance decreases when load factor is more.
* Collision may cluster, and this requires traversing the hash table one element at a time to find next available.

• Some keys such as floating points can lead to long chains and prefixes are not particularly meaningful.

## APPLICATION OF HASHING

Some applications are Message Digest, Password Verification, Compiler Operation, Rabin-Karp algorithm, Linking File name and path together.

# CONCLUSION

The project generalizes the file structure operations such as insert, update, search and delete, by abstracting away the core functionality of the same and to implement Hash function to Access. The project helps the user to keep track of all the information on various games provided by the user in a simple and structured file system, and can view the information stored in the files by using the SEARCH module which helps in retrieving the records of a desired game record. This project also enables the user to add a new entry to the file and also retrieve details from the File. ADD module has been implemented to insert records into the record file and to the index file. The DISPLAY module displays the file contents. The DELETE module helps in deleting an existing record entry, and the UPDATE module helps the user to update the current information of the existing anime record

# REFERENCES

### Text books:-

1. Michael J.Folk, Bill Zoellick, File Structure, 2nd edition
2. A.V. Oppenheim and R.W. Schafer, Digital Signal Processing, Englewood, N.J., Prentice Hall, 3 Edition, 1975.

### Websites -

1. https://www.w3schools.com
2. https://code.visualstudio.com
3. https://youtu.be/mqhxxeeTbu0