**CHAPTER 1**

**INTRODUCTION**

**1.1 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.

**1.2 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.

**1.3 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.

**CHAPTER 2**

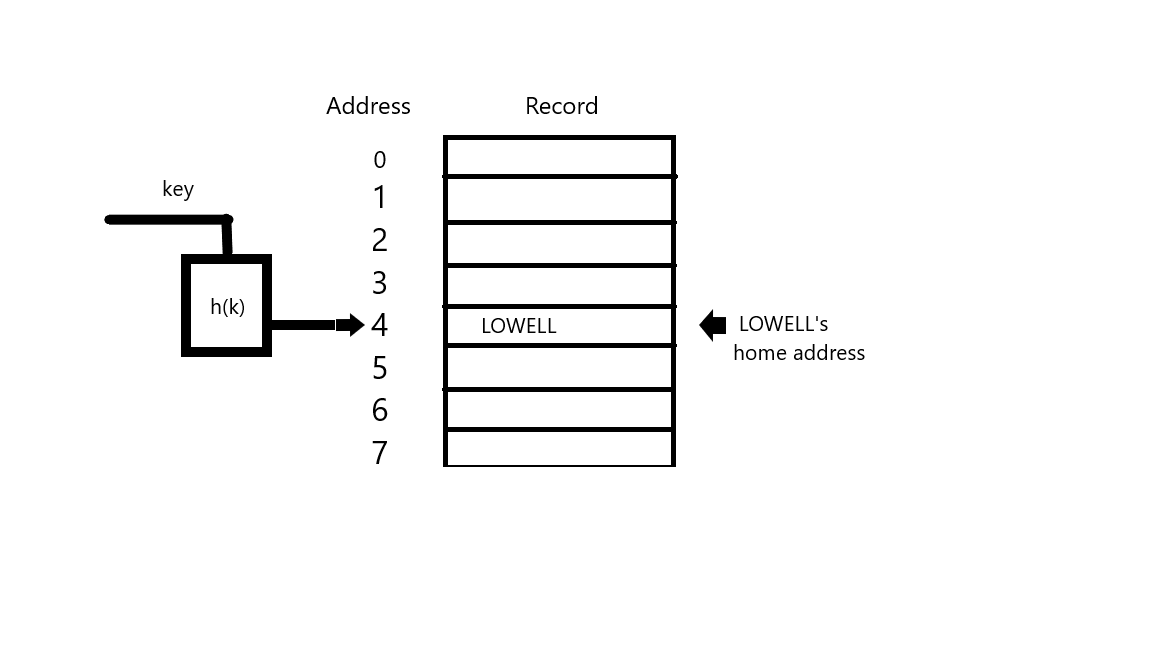
**METHODOLOGY**

**2.1 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**

**2.2 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 lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks.

* Flask uses thread-local objects internally so there’s no need to pass objects around from function to function within a request in order to stay threadsafe. This approach is convenient, but requires a valid request context for dependency injection or when attempting to reuse code which uses a value pegged to the request.
* Werkzeug implements WSGI, the standard Python interface between applications and servers.
* Jinja is a template language that renders the pages the application serves.
* Its Dangerous securely signs data to ensure its integrity. This is used to protect Flask’s session cookie.
* The debug support the server will reload itself on code changes, and it will also provide the user with a helpful debugger if things go wrong.
* Flask provides a really simple way to give feedback to a user with the flashing system. The flashing system basically makes it possible to record a message at the end of a request and access it on the next (and only the next) request. This is usually combined with a layout template to expose the message

### 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.

* 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

**VISUAL STUDIO CODE**

Visual Studio Code is a source code editor that can be used with a variety of programming languages,including Java, JavaScript, Go, Node.js and C++. It is based on the Electron framework which is used to develop Node.js web apps that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language, contrary to Microsoft Visual Studio which uses the proprietary. solution file and project-specific project files. It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many of Visual Studio Code features are not exposed through menus or the user interface, but can be accessed via the command palette

Visual Studio Code can be extended via extensions available through a central repository. This includes additions to the editorand language support. A notable feature is the ability to create extensions that add support for new languages, themes, and debuggers, perform static code analysis, and add code linters using the Language Server Protocol.

**GIT**

Git is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows

**GITHUB**

GitHub brings together the world's largest community of developers to discover, share, and build better software. Projects on GitHub can be accessed and manipulated using the standard Git command-line interface and all of the standard Git commands work with it. GitHub also allows registered and unregistered users to browse public repositories on the site. Multiple desktop clients and Git plugins have also been created by GitHub and other third parties that integrate with the platform

**CHAPTER 3**

**SYSTEM REQUIREMENTS SPECIFICATION**

**3.1 USER REQUIREMENTS**

The users of this application can be categorized as- application developer and end user. As an application developer, the user requirements are as follows-

1. Python version >= 3.7

2. Pip version > 20.2

3. Python packages – json, math

For an end user the user requirements are as follows-

1. RAM > 4GB

2. Processor : Intel® CoreTM i5-6500 CPU @ 3.20GHz pr greater

3. System type: 64-bit operating system, x64-based processor

4. Modern web browser

**3.2 SOFTWARE REQUIREMENTS**

The software for the development and deployment of the project includes-

1. Python

2. Pip

3. VS Code

4. Flask

5. Git

**3.3 HARDWARE REQUIREMENTS**

The hardware for the development of the projects include-

1. RAM > 4GB

2. Processor : Intel® CoreTM i5-6500 CPU @ 3.20GHz pr greater

3. System type: 64-bit operating system, x64-based processor

**3.4 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.

**3.5 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.

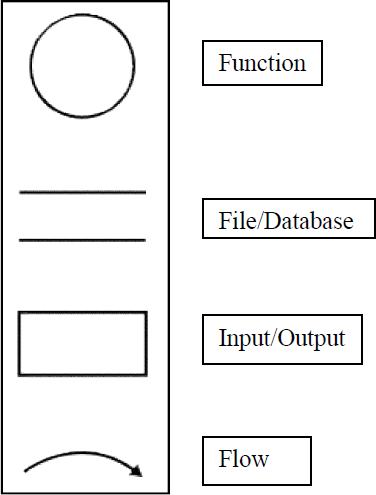
**CHAPTER 4**

**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.

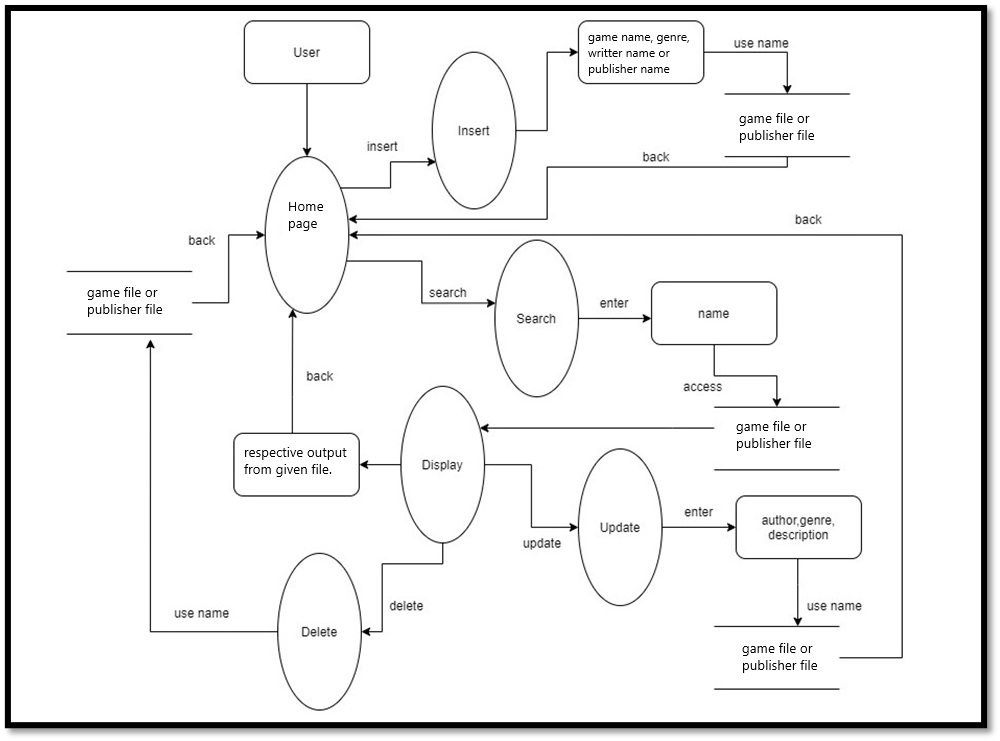
**4.1 ARCHITECTURAL DESIGN**

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 Fortress of gamers**

**CHAPTER 5**

**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

**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. 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.

:

1. Search the index for the key provided
2. If key exists:  
    retrieve RRN  
   Else:  
    Display record not found
3. Read record from file at RRN
4. Unpack record
5. Display record

3. Record Updation:

1. Perform Record Search
2. Create new record with updated values
3. Pack the record
4. Write the updated record at RRN

4. Record Deletion:

1. Perform Record Search
2. Place delimiter at beginning of record
3. Add RRN to avail list
4. Remove key, RRN from index

5. Index Insertion:

1. Descend to the leaf where the key fits.
2. If the node has an empty space, insert the key/reference pair into the node.
3. If the node is already full, split it into two nodes, distributing the keys evenly between the two nodes.
4. If the node is a leaf, take a copy of the minimum value in the second of these two nodes and repeat this insertion algorithm to insert it into the parent node.
5. If the node is a non-leaf, exclude the middle value during the split and repeat this insertion algorithm to insert this excluded value into the parent node.

6.Index Deletion:

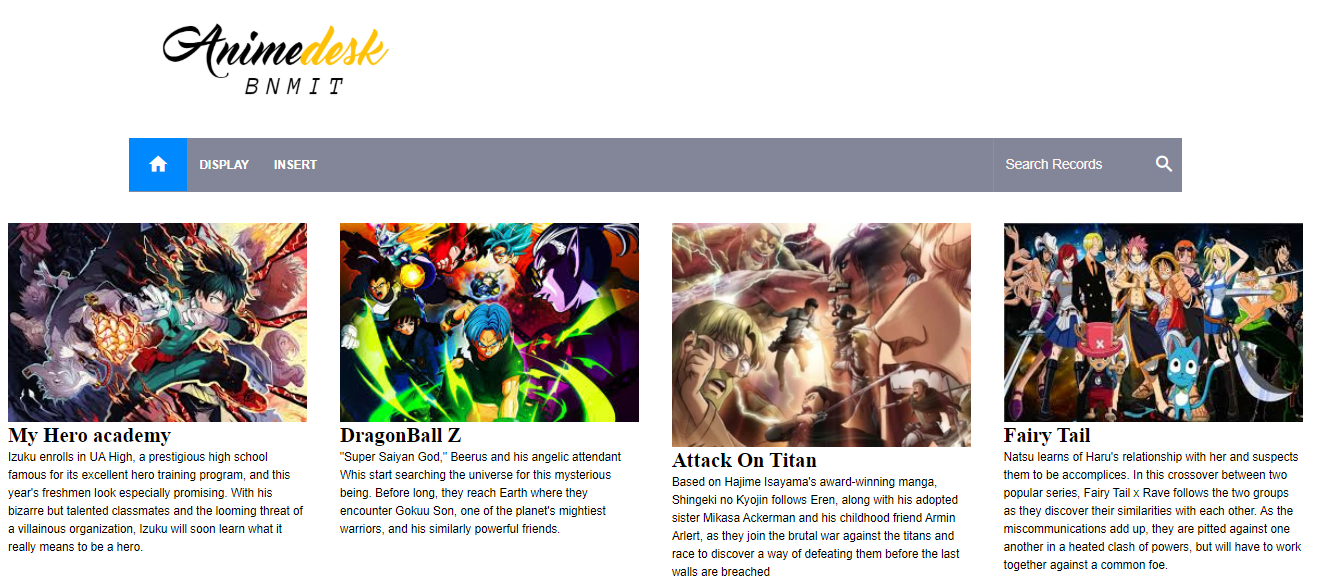
1. Descend to the leaf where the key exists.
2. Remove the required key and associated reference from the node.
3. If the node still has enough keys and references to satisfy the invariants, stop.
4. If the node has too few keys to satisfy the invariants, but its next oldest or next youngest sibling at the same level has more than necessary, distribute the keys between this node and the neighbor.
5. Repair the keys in the level above to represent that these nodes now have a different “split point” between them; this involves simply changing a key in the levels above, without deletion or insertion.
6. If the node has too few keys to satisfy the invariant, and the next oldest or next youngest sibling is at the minimum for the invariant, then merge the node with its sibling; if the node is a non-leaf, we will need to incorporate the “split key” from the parent into our merging.
7. Repeat the removal algorithm on the parent node to remove the “split key” that previously separated these merged nodes — unless the parent is the root and we are removing the final key from the root, in which case the merged node becomes the new root (and the tree has become one level shorter than before).

**CHAPTER 6**

**RESULTS AND DISCUSSIONS**

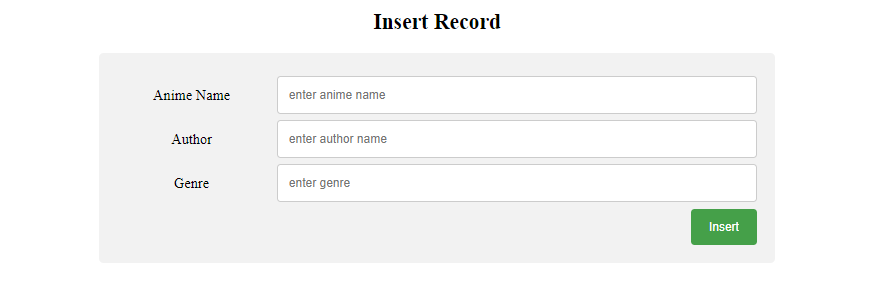
**6.1 SNAPSHOTS OF THE PROJECT AND DESCRIPTION**

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.



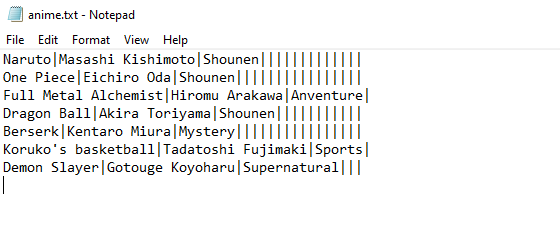
**Fig 6.1.1: Home page**

Home page describes the topic of the project that is “Animedesk” and the options that are available for the user to work on the topic.

****

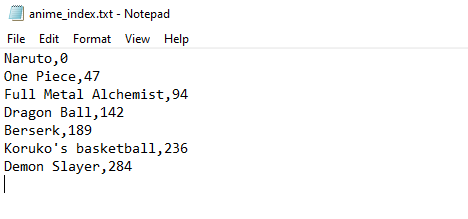
**Fig 6.1.2: Insert page**

This page allows the user to add information regarding various animes.

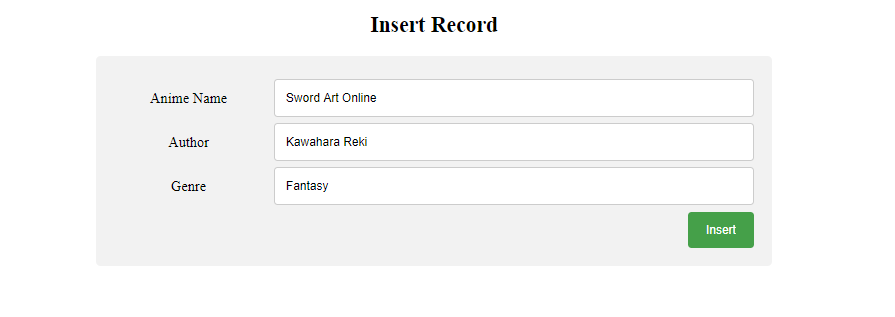


**Fig 6.1.3: Data file**

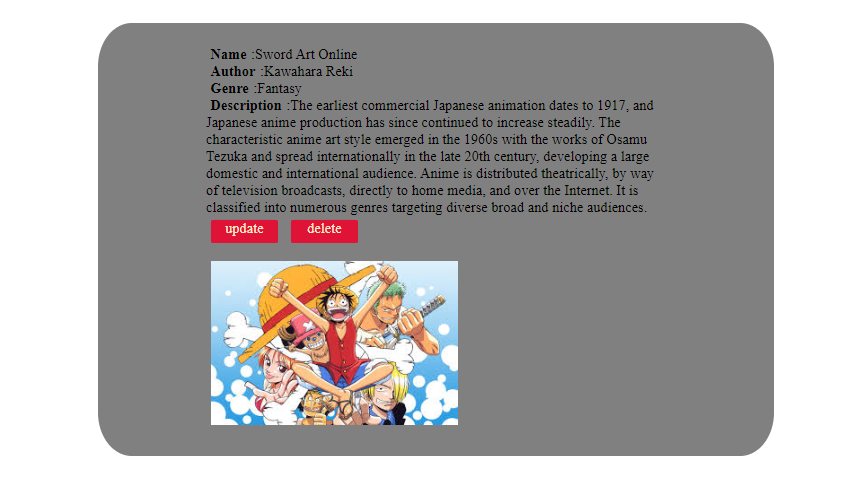
This page contains information on some of the animes which were previously added by the user, and this is before adding the new information to the file by the user.

**Fig 6.1.4: Index file**

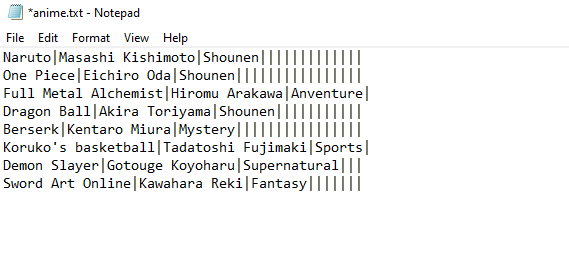
This page is the index file which has anime name and relative record number of the information/records that are present in the data file.

 **Fig 6.1.5: Insert page**

In this page user is entering information on a new anime by giving the anime name, author and genre, after clicking insert the information will be saved to the data file

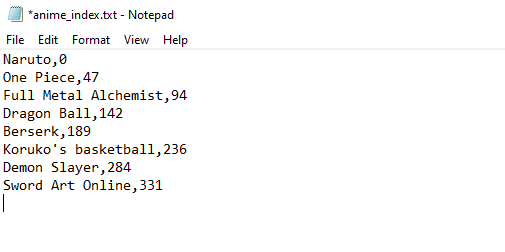
 **Fig 6.1.6: Display page for the current record**

In this page, the user is viewing the information of the inserted record

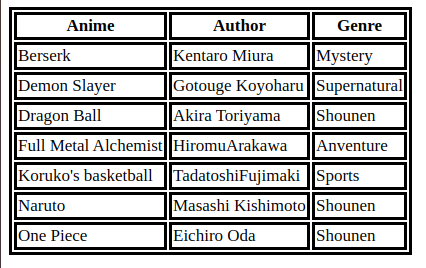


**Fig 6.1.7: Data file after insertion**

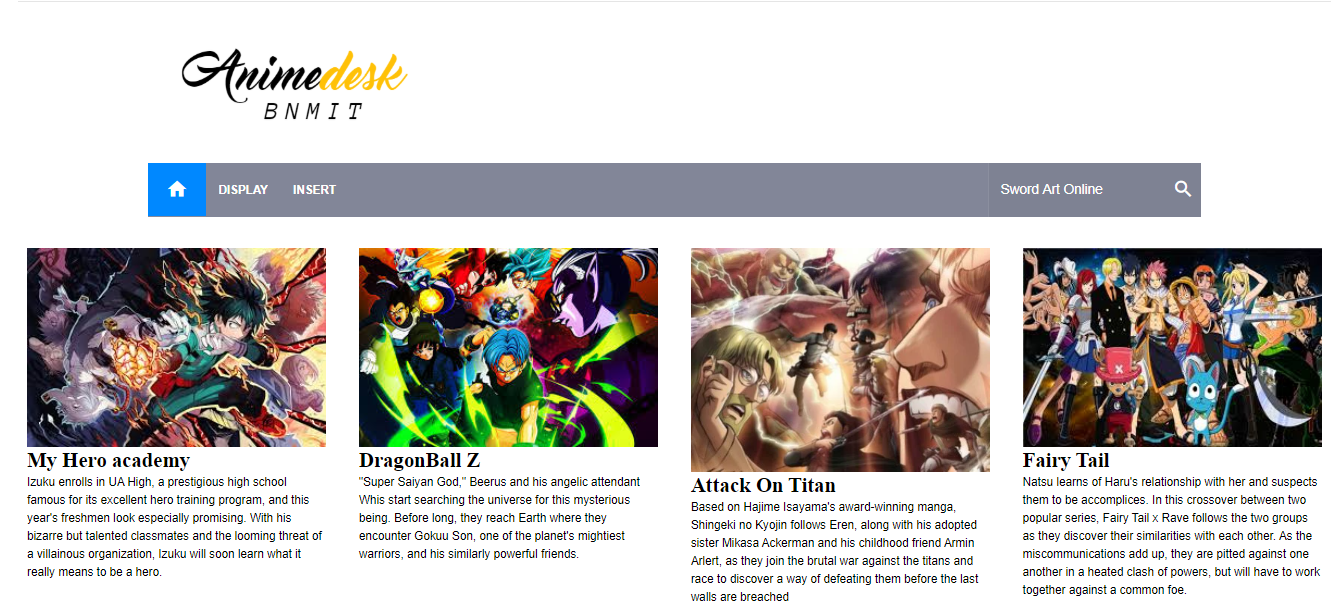
This page contains the eighth record information which was added by the user in the front end insert page, data file is modified.

 **Fig 6.1.8: Index file after insertion**

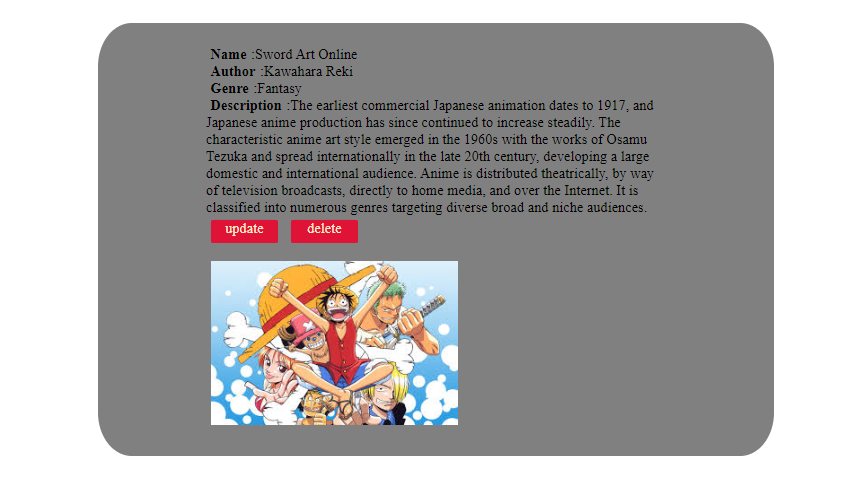
This page contains the primary key and rrn of the new record, after the insertion of the new record by the user.

**Fig 6.1.9: Display page**

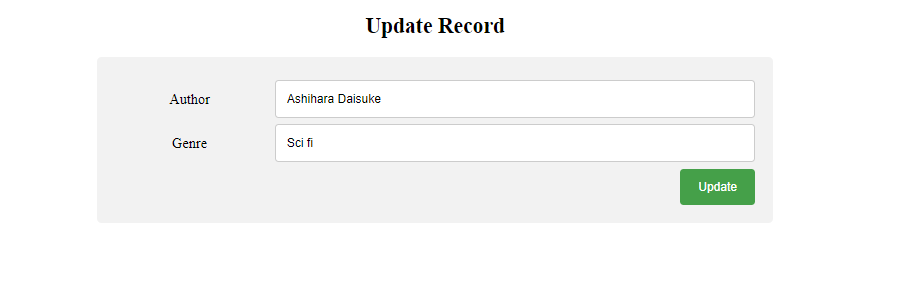
In the home page, if the user chooses the display option, the user gets redirected to this page which displays the contents that are present in the file in a sorted order in the form of a table.

 **Fig 6.1.10: Searching record in home page**

In the home page, user enters the anime name on the search bar to search for the particular information required by the user. Search method takes anime name as the primary key to fetch the corresponding record.

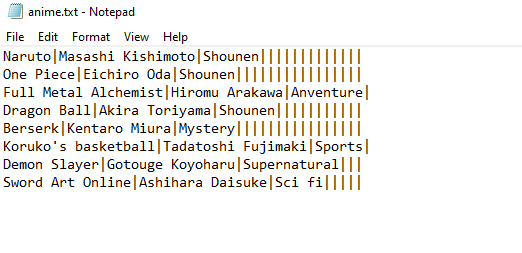
 **Fig 6.1.10: Search result page**

After user enters the anime name, the corresponding record details will be displayed on the screen. Additionally, this page also allows the user to either update or delete this record.



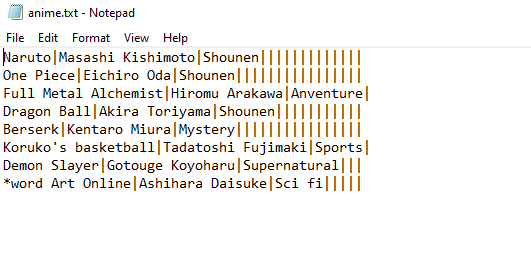
**Fig 6.1.11: Update page**

If the user opts the update option, the user is redirected to this page 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.



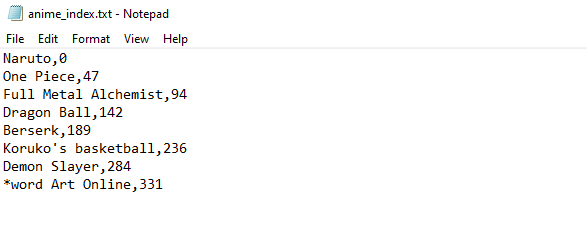
**Fig 6.1.12: Data file after Deletion**

This file describes the updated anime information that has changed after the modification that has been made by the user



**Fig 6.1.13: Data file after deletion**

If the user opted for the delete option, this page tells the user that after delete operation the anime name is represented by the special symbol \* which indicates that the record has got deleted



**Fig 6.1.14: Index file after Deletion**

This page tells the user that after deletion, that particular anime name from the index file is represented by the special symbol \* which indicates that the record has got deleted.

**6.2 OBSERVATION ABOUT THE PROJECT**

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 the optimization method of Indexed Sequential Access on top of the same. APIs were developed in order to enable to use “naivedb” for any file structures application. The application is observed to perform insertion, search, modify, delete and display operations. The user can enter the information of any number of anime record, which will be stored in the data file. After entering all the required details, user press insert button the insert() function runs in which respective record details are stored in data file and primary keys are stored in index file along with address/rrn.

The user can enter the anime name to get respective details. After entering the anime name on the search bar, search() function is called which checks for the corresponding anime name in index file and then in data file. If anime name is not present in the index file then a dialogue box appears displaying the record with anime name is not found. If the search is successful, the user also has the option to either modify or delete the corresponding record.

The user can view whole file details by clicking Display button. The whole file is display in form of a table with field headers.

ISAM is a method for creating, maintaining, and manipulating computer files of data so that records can be retrieved sequentially or randomly by one or more keys. In this method, records are stored in the file using the primary key. An index value is generated for each primary key and mapped with the record

Advantages of ISAM

* In indexed sequential access file, sequential file and random file access is possible.
* It accesses the records very fast if the index table is properly organized.
* The records can be inserted in the middle of the file.
* It provides quick access for sequential and direct processing.
* It reduces the degree of the sequential search.

Disadvantages of ISAM

* Indexed sequential access file requires unique keys and periodic reorganization.
* Indexed sequential access file takes longer time to search the index for the data access or retrieval.
* It requires more storage space.

**CONCLUSION**

The project generalizes the file sructure operations such as insert, update, search and delete, by abstracting away the core functionality of the same and to implement the optimization method of Indexed Sequential Access on top of the same The project helps the user to keep track of all the information on various animes 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 anime 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 recowrd 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

### 

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

### Websites

* https://www.w3schools.com
* https://code.visualstudio.com
* https://www.programiz.com
* https://app.diagrams.net