**DESIGN OF MESSAGE BUFFER PROJECT**

Aimed to build a user-friendly and efficient message buffer, we here by submit some specifications and an overview of our project.

Conditions Specified:

* 1 GB file-system given to handle the implementation.

INTRODUCTION:

This project is intended to develop a discussion forum with all the server side and client side coding done by ourselves in python. A file system is designed as per the requirements of the discussion forum. This document describes the specifications and design features of the project.

scope:

* The project runs on limited resources. A 1GB file system is provided for server side storage of the data.
* The UX is implemented in HTML5. The server and client communication uses REST-API and all the client side and server-side coding is done in python.
* Pycharm tool is used to run the python code.

Functional requirements:

* Every user is going to have a unique user-id and password.
* Unless an account is created visitors are not allowed to post anything in the forum.
* Unregistered users can only search or view the question –answer sessions.
* We are providing users fixed five categories which are: Education, Entertainment, Politics, Sports and Others.
* We are going to provide three level hierarchy with Questions(level 0), Answers (level1) and Reply to those answers(level2).
* There is no reply to replies but supporting multiple replies to an answer.

NON-FUNCTIONAL REQUIREMENTS:

* Meetings:For users who need constant interaction to resolve their problems meetings are provided where a user can post the question and time at which he can will wait for other users to respond to his question and interact with them.
* Hanging on the Wire:Users are provided with attractive UX in which all the top questions are displayed on the main page of the forum.
* Add to Bag:Users can add questions and answers he feel interesting to his bag and read them when ever needed.

Design Features:

**Data Design:**

* The file system is of size 1 GB. This file system is used to store entire User data, metadata (categories, forums, questions, answers and replies) and actual forum data (Questions, answers and replies).
* The internal data structure used to store the data regarding the forums, questions and answers is the inbuilt dictionary of python library. Linked lists are used to connect different data blocks of file system. Each block is of size 1 KB.

FILE SYSTEM IMPLEMENTATION:

**File system allocation**

1 KB 1 MB – 32\*5 B

pointers user metadata cat\_metadata

forum metadata

question metadata

answers metadata

reply metadata

questions data

answers data

reply data

0 MB

1 MB

3 MB

33 MB

103 MB

123 MB

323 MB

883 MB

1 GB

**Block1: (pointers)**

1. It is of 1024 bytes,it starts from 0 to 400bytes(Hex)
2. It consists of various pointers like

* users\_offset = it points to the next user metadata
* no\_of\_users = number of users
* forum\_offset = it points to the next forum metadata
* ques\_offset = it points to the next question metadata
* act\_ques\_pointer = it points to the next actual question data address
* no\_of\_ques = number of questions
* ans\_offset = it points to the next answer metadata
* act\_ans\_pointer= it points to the next actual answer data address
* no\_of\_ans=number of answers

**Block2: (user metadata)**

1.it starts from 400 bytes(1kb ) and ends at FFF60 bytes((1024\*1024)– 32\*5)

2. each user metadata is of 32 bytes

|  |
| --- |
| User name 12bytes |
| Password 12bytes |

**Block3: (category metadata)**

1.it starts from FFF60 and ends at 100000 bytes

2.each category metadata consist of 32 bytes

|  |
| --- |
| Category id 1 byte |
| Category name 15 bytes |
| Next\_category addr 4 bytes |
| Forum address 4 bytes |
| + 8 bytes |
| Total 32 bytes |

**Block4: (forum metadata)**

1.it starts from 100000 and ends at 200000 bytes

2.each category metadata consist of 64 bytes

|  |
| --- |
| Next\_forum\_ptr 4 bytes |
| Category id 1byte |
| Forum\_name 27 bytes |
| User\_name 12bytes |
| Question\_start \_addr 4 bytes |
| Question\_end\_addr 4 bytes |
| +12 bytes |
| Total 64 bytes |

**Block5: (questions metadata)**

1.it starts from 300000 and ends at 2100000 bytes

2.each questions metadata consist of 128 bytes

|  |
| --- |
| Question\_id 4 bytes |
| +points(likes) 4bytes |
| -points(dislikes) 4bytes |
| Tag[0] 8bytes |
| Tag[1] 8bytes |
| Username 12bytes |
| Answer\_start 4 bytes |
| Answer\_end 4 bytes |
| Actual\_ques\_pointer 4 bytes |
| Next\_question 4 bytes |
| Forum\_name 28bytes |
| Length\_of\_question 4 bytes |
| Time\_stamp 24bytes |
| +16bytes |
| Total 128 |

**Block6: (answers metadata)**

1.it starts from 2100000 and ends at 6700000 bytes

2.each answers metadata consist of 128 bytes

|  |
| --- |
| Answer\_id 4 bytes |
| +points(likes) 4bytes |
| -points(dislikes) 4bytes |
| Actual\_ans\_addr 4bytes |
| Next\_ans\_id 4bytes |
| Reply\_start\_addr 4bytes |
| Reply\_end\_addr 4 bytes |
| User\_name 12bytes |
| Length\_ans 4 bytes |
| Time\_stamp 24bytes |
| +60bytes |
| Total 128 |

**Block7: (reply metadata)**

1.it starts from 6700000 and ends at 7B00000 bytes

2.each reply metadata consist of 64 bytes

|  |
| --- |
| reply\_id 4 bytes |
| Next\_reply\_addr 4bytes |
| Actual\_reply\_addr 4bytes |
| User\_name 12bytes |
| Time\_stamp 24bytes |
| Reply\_length 4bytes |
| +12bytes |
| Total 64bytes |

**Block8: (actual question data)**

1. it starts from 7B00000 and ends at 14300000 bytes

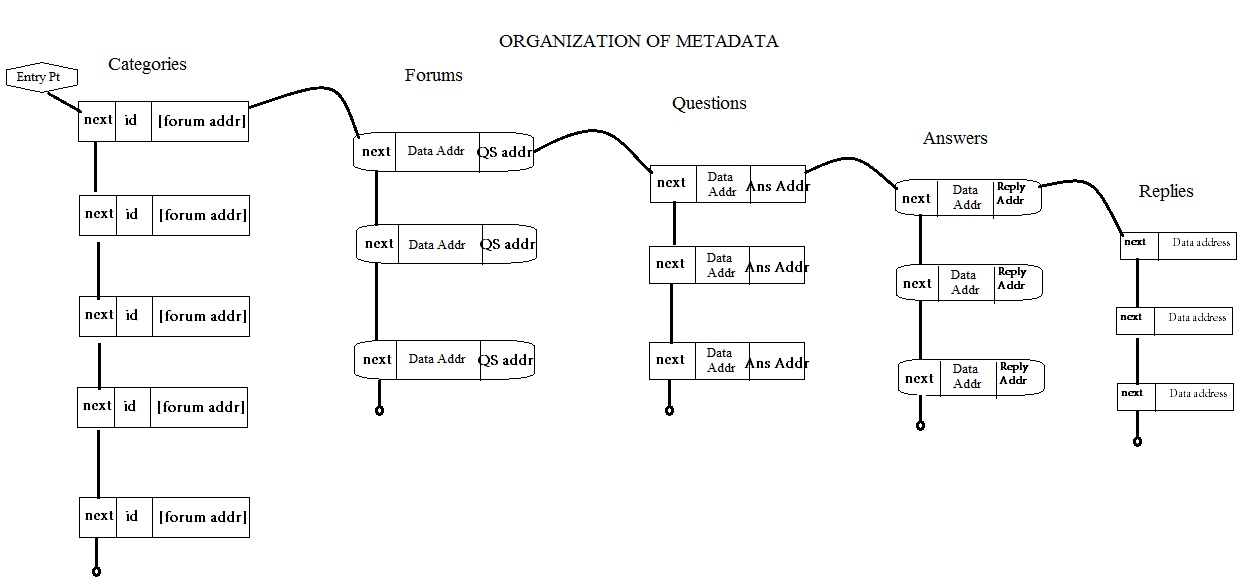
**Block9: (actual answer data)**

1. it starts from 14300000 and ends at 37300000 bytes

**Block9: (actual reply data)**

1. it starts from 37300000 and ends at bytes 40000000 bytes

Organization of Metadata



**Data structures used in caching**

1.Forum data is a list of list of size 5

* each element consists of list of forum names of each category.

2.Forum cache is a list of 10 elements

* each element is a tuple, consisting of forum name and time stamp

3.Questions is a list of lists of 10 elements

* each element is a tuple of metadata of question and actual question.

4.Answer cache is a list of 4 elements

* category\_id
* forum\_name
* question\_id
* answers list