Members: Vardan Saini, Arthur Chan

Mini Project 2 Report

Introduction

The purpose of this project is to build a command-line interface application which provides persistent question/answer services. The application consists of two python programs, phase1.py and phase2.py. Given that the file "Posts.json", "Tags.json", and "Votes.json" exists in the current directory, phase1.py constructs a MongoDB database called "291db" containing three collections. Afterward, phase2.py allows the user to operate on the '291db' by posting questions, answers, and votes using a command line interface.

Application Overview

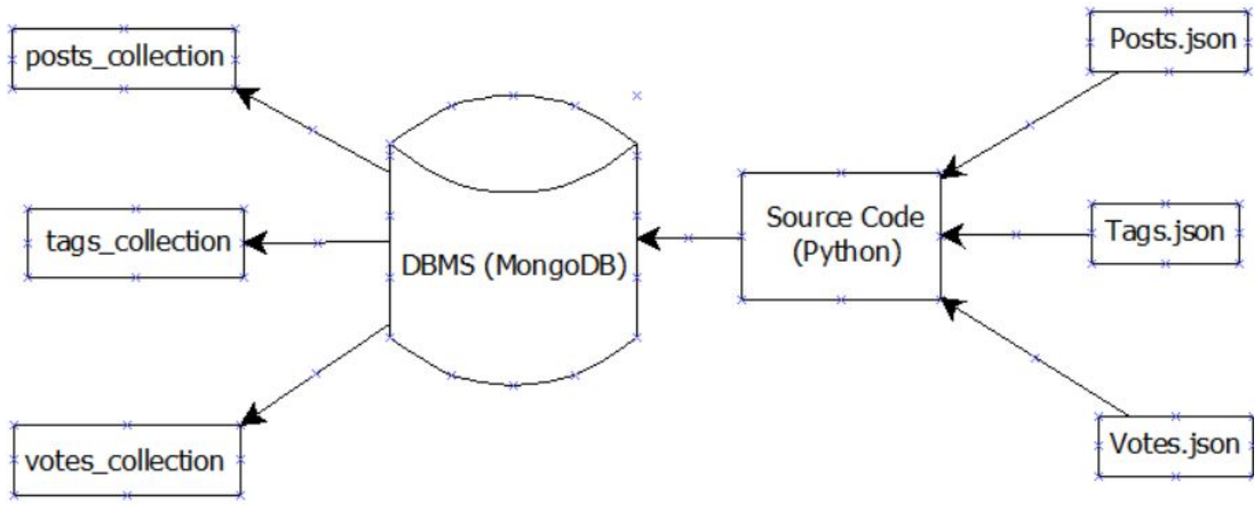

Figure 1: Phase1 Application Overview

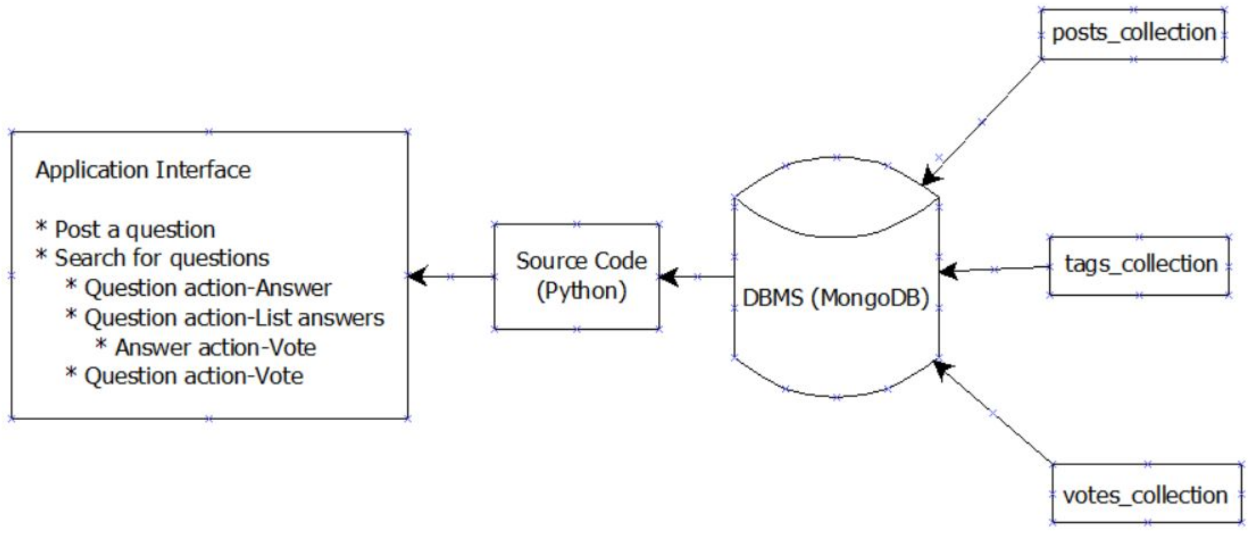

Figure 2: Phase2 Application Overview

User Guide

1. From the shell, navigate to the directory containing the file phase1.py and phase2.py

2. Copy Posts.json, Tags.json, Votes.json to this directory
3. Create an empty folder
4. Start a mongoDB server by executing the command “mongod --port (port #) --dbpath (empty folder name) &”
5. Build the database from the json data by executing the command “python3 phase1.py”
6. Start the command-line application by executing the command “python3 phase2.py”

Application Path

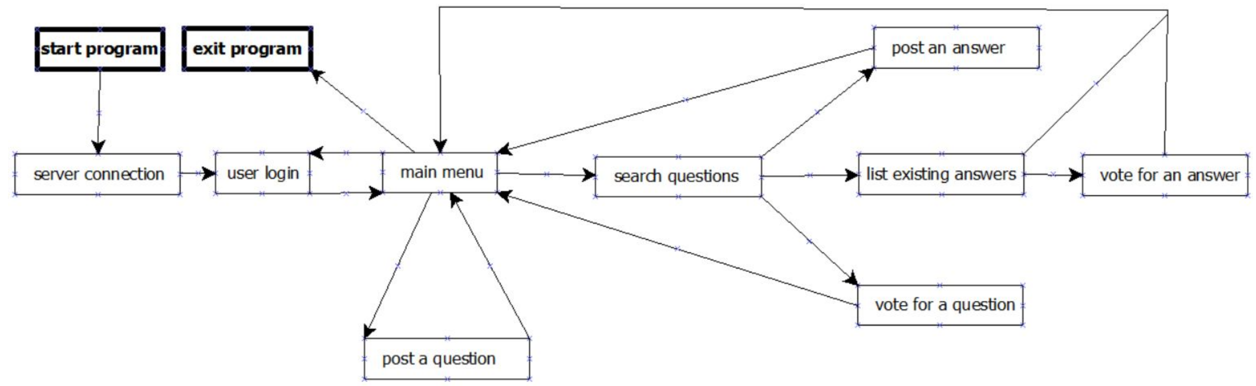

Figure 3: Possible Use Case Sequence

Software Design

Phase1.py is a script which loads each json file and builds the corresponding collection. Phase2.py contains a number of functions which provides the necessary user interface, and acts as drivers for the following major database functionalities:

- displayUserReport(userID) - Given a user ID, prints the number of questions the user has created, the average score of those questions, the number of answers the user has created, the average score of those answers, and the number of votes the user has casted.
- generateUniquePostID(type) - Helper function for insert tasks. Given an integer 1, 2, or 3, it returns a unique Id field in posts_collection, tags_collection, and votes_collection respectively.
- postQuestion(userID) - Given a user ID, it prompts the user for a title, body, and tags. After the user has entered the prompts, it inserts the new question post into posts_collection. If the user has entered one or more tags, it inserts the tag into tags_collection or updates the corresponding tags.
- searchQuestion() - It prompts the user for keywords separated by spaces. It returns a list of questions with at least one matching keyword in title, body, or tag field.
- postAnswer(userID, selectedQuestion) - Allows users to give answers to questions. It asks the user to enter an answer for the question and insert it into posts_collection.
- listAnswers(postID) - This function lists all available answers for the selected question.
- castVote(userID, postID) - Increases 1 score for a user with a user ID on one particular post if requested by user and user without user ID can vote as many times as they want on any post. Also updates votes_collection with provided information, to keep track of who voted when and on what.

Testing Strategy

The general strategy for testing phase1 is to execute it on the lab machine, then verify that the database “291db” containing the three collections is built in under 5 minutes. The general strategy for testing phase2 is to execute all database operations from the user interface, then verify that each database operation is executed successfully using a mongo client shell.

- User report - all user informations reported are correct
- Post question - posts_collection and tags_collection are updated correctly. A unique post ID is assigned.
- Search question - corrected search result is returned, and user is able to select a question for further actions
- Post answer - posts_collection is updated correctly, A unique post ID is assigned.
- List answers - all corresponding answers are returned. The accepted answer is marked. The user is able to select an answer for further action
- Vote - posts_collection and votes_collection are updated correctly. A unique vote ID is assigned. An identified user is blocked from voting on the same post more than once.

Work Breakdown

The project works are organized in a Github repository.

Component of Project	Completed by	Estimate of time
Phase 1: Build document stores from the three json files	Arthur Chan Vardan Saini	5 hours
Phase 2: program start, User report upon login, main menu loop, user logout, program exit	Arthur Chan	5 hours
Phase 2: ‘Post a question’	Arthur Chan	5 hours
Phase 2: ‘Search for questions’	Arthur Chan	6 hours
Phase 2: ‘Question action-Answer’	Vardan Saini	4 hours
Phase 2: ‘Question action-List Answers’	Vardan Saini	4 hours
Phase 2: ‘Question/Answer action-Votes’	Vardan Saini	3 hours
Report.pdf	Arthur Chan Vardan Saini	1 week