—---

Crowdsourcing Dataset Platform

Team Name: Crowdsourcing Dataset Platform
Team Members: Jacob Samar, Alan Hencey, David Gilson

—---

Introduction

The goal of this project is to create a web application that allows users to both access and create public datasets for use in their research or software development. The motivation behind the project is to create a more central hub for users to find datasets and to also add to/create public datasets for overall public benefit. Lots of datasets are spread around different sites and applications, and there might be different datasets that are either underdeveloped or non-existent that users could add to if given the ability. In a way, the motivation is similar to how Wikipedia is thought of, in that the public is able to create/modify datasets to benefit the overall public dataset domain for open source software or research use.

This project idea is semi-novel. Google has created a similar application at https://crowdsource.google.com/about/ but this website only allows users to contribute to Google's proprietary datasets. Our project will allow any developer to request custom data or data-labeling services from the users who can all contribute to building open source datasets for use by anyone.

Customer Value

The primary customer would be someone who requires specific datasets for their research or software project and needs an application that helps to centralize publicly available datasets. As mentioned before, publicly available datasets can be spread across many sites and applications that a user doesn't want to have to sift through to acquire specific datasets. The user might also have datasets they either have made from their own research or experiences that they'd like to make publicly available. Our project will provide a place for developers to come and find pre-existing data sets that meet their needs as well as build completely new, customized datasets from scratch.

Currently, developers who need to create a new dataset must collect thousands of samples of data and then meticulously label and/or modify the data before it can be used to develop models. This can be done by a team of researchers or a group of paid volunteers, both of which require excessive time and money. The new capability provided by this project is a centralized platform where, given enough users, developers will be able to quickly and cost-effectively build datasets through collective contribution. We will be able to measure the success of the platform through how quickly developers are able to crowdsource the data they need for their datasets. If enough users are contributing to building datasets then developers should be able to reach their dataset goals in a relatively short period of time. If not, then some datasets may never be finished.

Proposed Solution & Technology

This software will be web based, so users will be able to access it through a web browser. The website will be broken down into two main components. The first is a search engine that scours popular dataset websites as well as our own website for existing datasets, allowing users to see if a dataset already exists that matches their needs. The second is a crowdsourcing tool that allows developers to post requirements for a desired dataset as well as contribute to other datasets that have been posted.
The architecture of our system has three main parts: a front-end, a scraping script, and a backend with a database. The front-end will be designed using the web framework bootstrap. It will enable the user to search all the publicly available datasets within our application, create a public request for a dataset, and contribute to public dataset requests submitted by users. A python scraping script will be created to scour the web for different public datasets that will be added to our database. Our backend database will be implemented with SQL and will store public dataset info and aggregate the datasets that can be created with our application. Public datasets will have descriptive info regarding the data and a link to the dataset on the web.
The minimum viable product of our application would be the functionality of requesting specific datasets that can be crowdsourced by other users. This is the more novel and useful functionality of our proposed system. The aggregation of all publicly available datasets and the ability to effectively search them based on metadata, features, size, etc. would be very beneficial for our target audience. This search engine and dataset collection would allow machine learning engineers to quickly and conveniently find/create datasets for their projects.
Our system will be tested throughout the development process with automated unit and integration tests to ensure functionality. For more comprehensive testing, hopefully we can create a minimal viable product in time to perform some informal user testing. From that user testing, we would hopefully get some insight into what features users like, any potential UI issues, and the overall usability of our system.

Team

Jacob Samar has built several similar components that will be needed to make this application work. This includes different web scraping applications for collecting data and several web based GUIs. Our web framework of choice is bootstrap, which is a framework Samar has never used before. However, web frameworks have similar programming concepts, so the previous projects written in react-native and ember.js give a base of experience.

Alan Hencey has experience creating SQL databases and with Python web scraping.

David Gilson has created web-based GUIs with the tools (Bootstrap) that will be used in the project interface. This tool is a library/template set for creating easy to use, adaptable GUIs with HTML/CSS/Javascript, and he has used it to create both desktop and mobile supported GUIs.

The roles of product manager and product owner will be randomly assigned each week. This will ensure that each member of the team will get some experience in each role throughout the project.

Project Management

Completion of the system is feasible.
We will meet twice weekly remotely or face-to-face if necessary.

Schedule:

- Sprint 1 - Ends February 18th
    - Complete project proposal
- Sprint 2 - Ends March 4th
    - Plan major future sprint milestones
    - Create first draft comprehensive design of platform functionality and integration.
    - Have barebones website page structure and navigation, SQL database table design, and beginnings of web crawler/search engine.
    - barebones website structure, initial database setup, web crawler started
- Sprint 3 - Ends March 18th
    - TBD
    - Setup login/register pages, kaggle web crawler finished
- Sprint 4 - Ends April 1st
    - TBD
    - Convert website structuring to php
- Sprint 5 - Ends April 15th
    - TBD
    - Implement login sessions, finish UCI web crawler, create dummy page for viewing datasets, implement internal datasets viewing + uploading
- Sprint 6 - Ends April 29th
    - TBD
    - Create db entry for external datasets, code to load scrapped datasets, search page for internal and external datasets, user page created, functionality for viewing/deleting internal datasets
- Sprint 7 - Ends May 13th
    - TBD
    - Connecting + bug fixing the different pages of the website, update navbar functionality, implement dataset downloading, improved data upload (multiple files), add saving datasets to user profile, download internal dataset as zip file
    - Overall code cleanup and bug fixing.

There are no regulatory or legal constraints, but one social concern is if malicious users submit bad data to others' datasets. We will allow the creators/owners of a dataset to screen contributed data in order to filter out data that does not meet the criteria they required.

The data needed will all be open source data found through web scraping or data that has been contributed by users.
The minimum functionality that will need to be implemented for the project to be useful is the ability for developers to post requirements for datasets, and contribute to each others' datasets. Without the search engine the platform can still operate and be useful, but possibly not as convenient.

Development Process

We have chosen the Scrum framework as our development process. This is largely due to its flexibility of development, its structured tasks and roles, and that it is a process that each member of the team has some familiarity with. In addition, the three questions addressed in each daily scrum ('What did you do yesterday?', 'What will you do today?', 'Are there any issues blocking you?') are simple and efficient in enabling teamwork and communication.
XP explicitly engages with users as a main part of the process, which, without a dedicated user outside our group, makes it less worthwhile for this project. Waterfall seemed like a very rigid process, and didn't offer the flexibility of development that scrum offers.

Use Case

There are a few use cases that this software could handle, but one could possibly be this:

A researcher who is programming an AI photograph development tool would like some base picture data sets of common objects or perhaps even data already made by AI. The researcher could go to the site, search for datasets with tags related to their needs, and train their AI photo generation tool. After the researcher feels the tool does an adequate job and has produced photographic datasets based on their own tool, they could create or contribute back to any AI tagged photographic dataset on the site. This method progressively builds onto and varies the datasets available on the site.

Test Cases

Our test cases were performed by running the website locally and then manually entering information, navigating, and interacting with the website exactly how a user would. Specific use cases that were tested were logging in/registering a user, including entering passwords that did not match and trying to register a username that was already taken, navigating between pages and ensuring the the session variables tracked relevant session information, and all of the functionality associated with creating, updating, modifying, following, searching, and deleting datasets. Initially, these features were tested individually as they were added to the project. However, towards the end of the semester, longer test cases were run that encompassed testing most or all of the features of the website within a single session.

Code Review

Our code review process for our project was to commit, then review. Each of us would generally be working on a separate feature at any given time and would perform our own tests before committing those changes to the project. Once those changes were pulled by everyone, each of us would then test and review the new features to ensure they had seamlessly integrated with the project. These review processes were typically performed individually, but closer to the end of the semester when the commits became larger we began reviewing as a group in order to more quickly identify and eliminate bugs in the code.

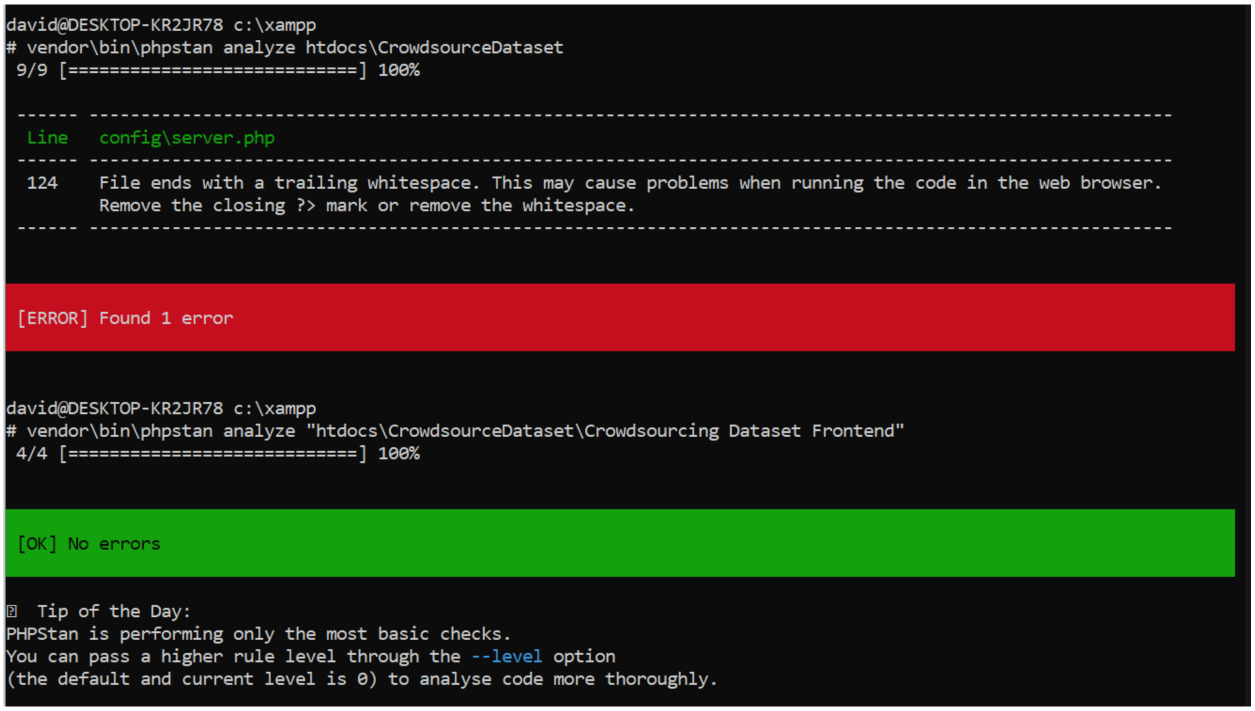
Static Analysis

We made use of the tool PHPStan, which is a free static analysis tool that checks php files for common errors.The errors it checks for includes the following:

- The existence of classes used in instanceof, catch, typehints and other language constructs.
- The existence and accessibility of called methods and functions.
- Checks the number of passed arguments, whether a method returns the same type it declares to return.
- Existence and visibility of accessed properties. It will also point out if a different type from the declared one is assigned to the property.
- Correct number of parameters passed to sprintf/printf calls based on format strings.
- Existence of variables while respecting scopes of branches and loops.
- Useless casting like (string) ‘foo’ and strict comparisons (=== and !==) with different types as operands which always result in false.

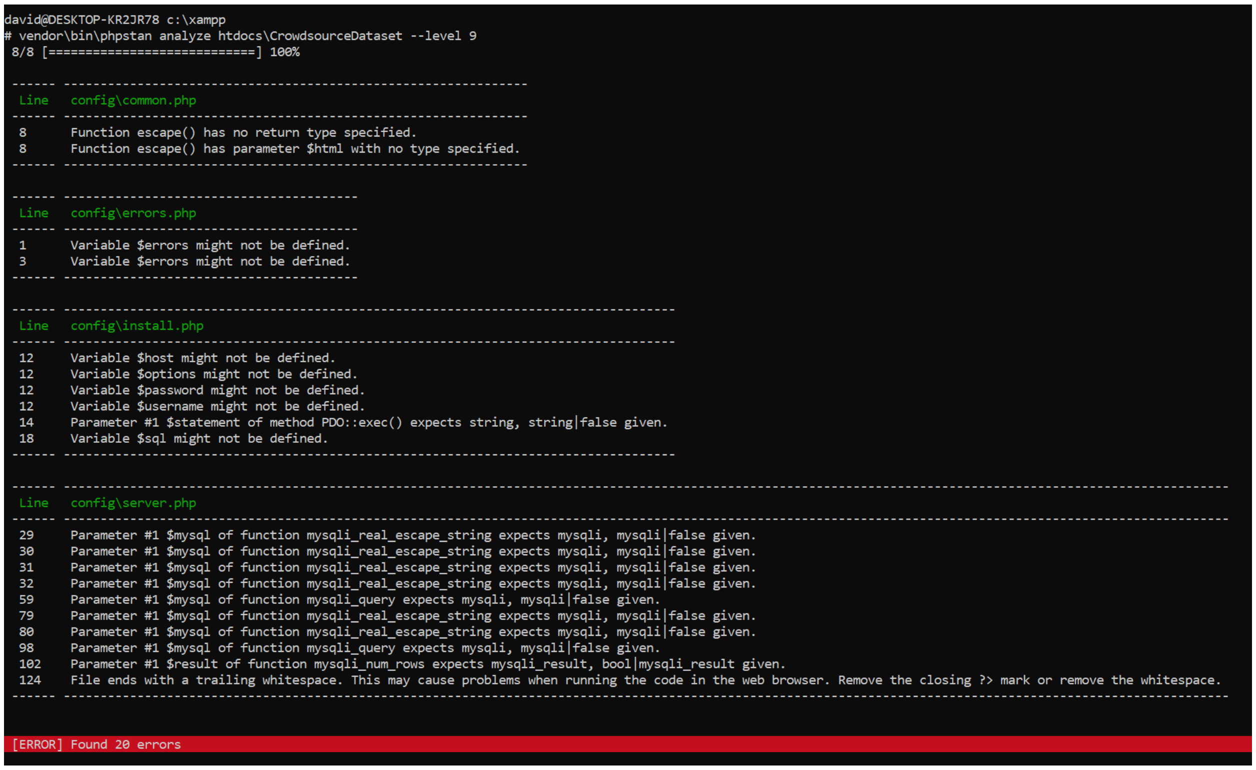
Github link: https://github.com/phpstan/phpstan

Static Analysis at Level 0 Testing (Lowest Level)



```
david@DESKTOP-KR2JR78 c:\xampp
# vendor\bin\phpstan analyze htdocs\CrowdsourceDataset
 9/9 [============================] 100%

 ------ ----------------------------------------------------------------------------------------------------------
  Line   config\server.php
 ------ ----------------------------------------------------------------------------------------------------------
  124    File ends with a trailing whitespace. This may cause problems when running the code in the web browser.
         Remove the closing ?> mark or remove the whitespace.
 ------ ----------------------------------------------------------------------------------------------------------


 [ERROR] Found 1 error


david@DESKTOP-KR2JR78 c:\xampp
# vendor\bin\phpstan analyze "htdocs\CrowdsourceDataset\Crowdsourcing Dataset Frontend"
 4/4 [============================] 100%


 [OK] No errors

?  Tip of the Day:
PHPStan is performing only the most basic checks.
You can pass a higher rule level through the --level option
(the default and current level is 0) to analyse code more thoroughly.
```

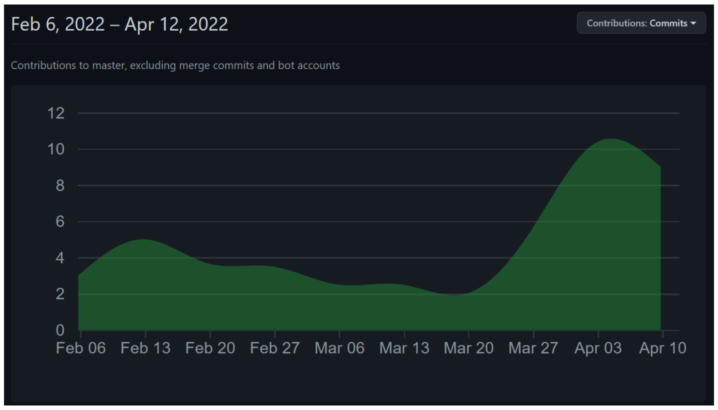
Static Analysis at Level 9 Testing (Highest Level)

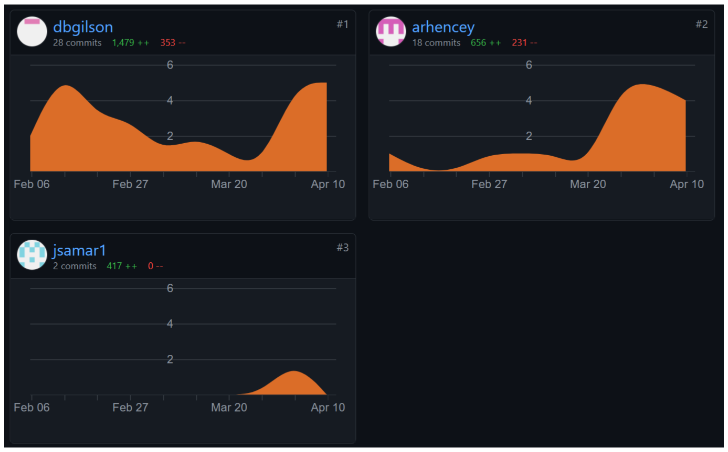


```
david@DESKTOP-KR2JR78 c:\xampp
# vendor\bin\phpstan analyze htdocs\CrowdsourceDataset --level 9
 8/8 [============================] 100%

 ------ ---------------------------------------------------------------
  Line   config\common.php
 ------ ---------------------------------------------------------------
  8      Function escape() has no return type specified.
  8      Function escape() has parameter $html with no type specified.
 ------ ---------------------------------------------------------------

 ------ ----------------------------------------
  Line   config\errors.php
 ------ ----------------------------------------
  1      Variable $errors might not be defined.
  3      Variable $errors might not be defined.
 ------ ----------------------------------------

 ------ ---------------------------------------------------------------------------------
  Line   config\install.php
 ------ ---------------------------------------------------------------------------------
  12     Variable $host might not be defined.
  12     Variable $options might not be defined.
  12     Variable $password might not be defined.
  12     Variable $username might not be defined.
  14     Parameter #1 $statement of method PDO::exec() expects string, string|false given.
  18     Variable $sql might not be defined.
 ------ ---------------------------------------------------------------------------------

 ------ ---------------------------------------------------------------------------------------------------------------------------------------------------------------------
  Line   config\server.php
 ------ ---------------------------------------------------------------------------------------------------------------------------------------------------------------------
  29     Parameter #1 $mysql of function mysqli_real_escape_string expects mysqli, mysqli|false given.
  30     Parameter #1 $mysql of function mysqli_real_escape_string expects mysqli, mysqli|false given.
  31     Parameter #1 $mysql of function mysqli_real_escape_string expects mysqli, mysqli|false given.
  32     Parameter #1 $mysql of function mysqli_real_escape_string expects mysqli, mysqli|false given.
  59     Parameter #1 $mysql of function mysqli_query expects mysqli, mysqli|false given.
  79     Parameter #1 $mysql of function mysqli_real_escape_string expects mysqli, mysqli|false given.
  80     Parameter #1 $mysql of function mysqli_real_escape_string expects mysqli, mysqli|false given.
  98     Parameter #1 $mysql of function mysqli_query expects mysqli, mysqli|false given.
  102    Parameter #1 $result of function mysqli_num_rows expects mysqli_result, bool|mysqli_result given.
  124    File ends with a trailing whitespace. This may cause problems when running the code in the web browser. Remove the closing ?> mark or remove the whitespace.
 ------ ---------------------------------------------------------------------------------------------------------------------------------------------------------------------


 [ERROR] Found 20 errors
```

Project Measurement

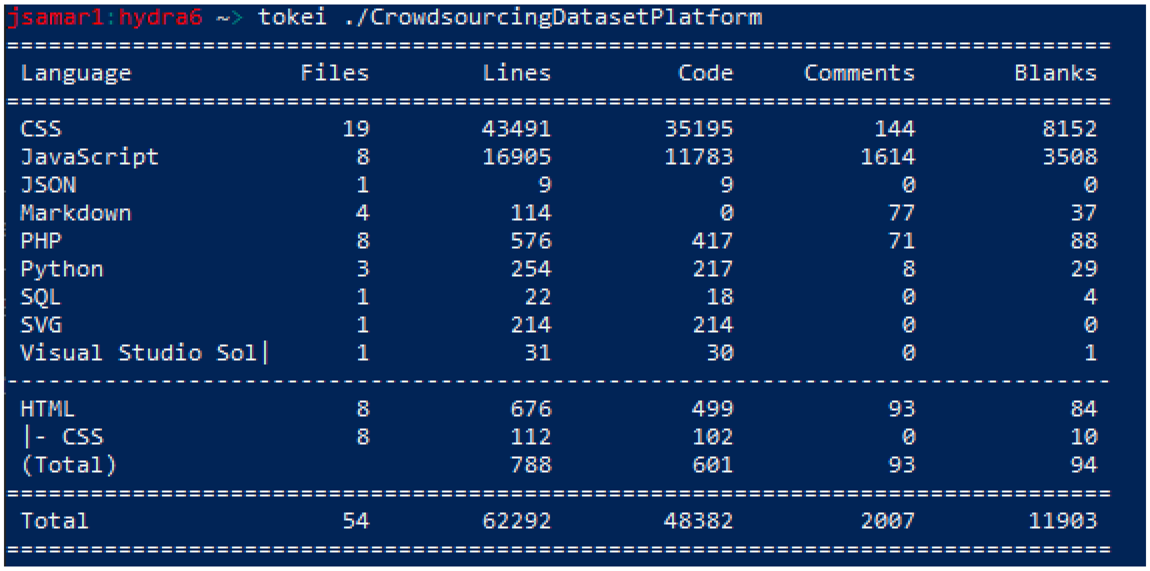
Our initial resource for project measurement was Github Insights. This tool is an easy to implement, built-in tool to Github used for evaluating several metrics on a repo. We were able to create a contribution over time graph and an individual commits over time.





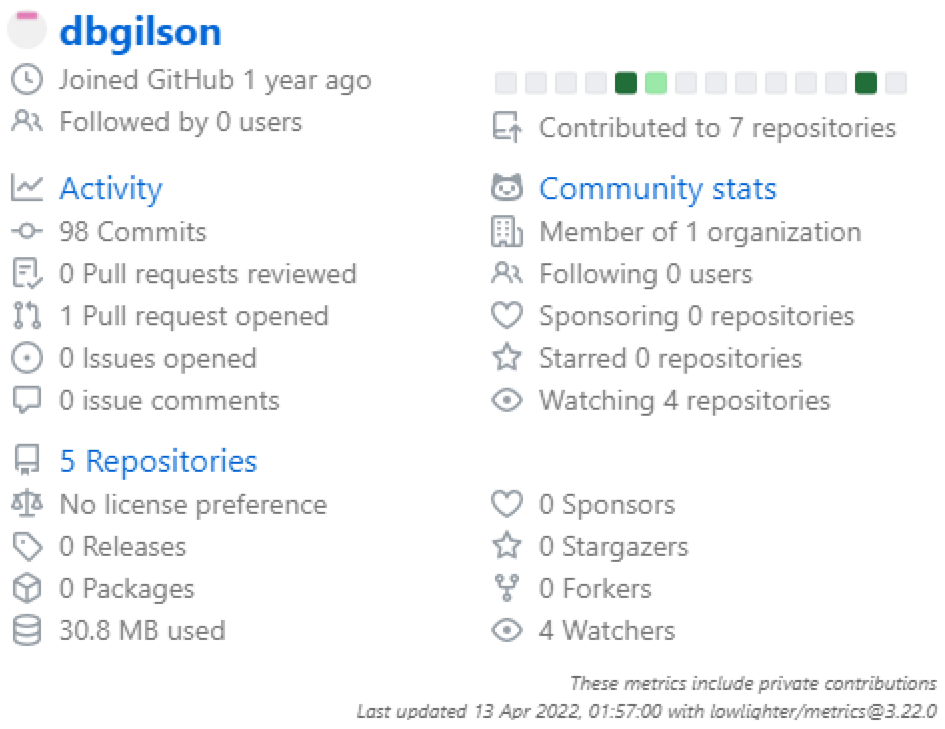
Viewing lines added/removed doesn't really account for copying/template usage however, so that can skew views on unique line creation. A less interesting metric created was a graphic of activity for the repo. It charted the discussions, issues, and pull requests issued. As our team typically communicated through discord, this offered very little insight into our project.

We also made use of a lines of code tool. This tool can be used to determine language usage, as well as determining certain practices used in the code like how many comments or blank lines are used with that particular language, as well as the project in general.



```
jsamar1:hydra6 ~> tokei ./CrowdsourcingDatasetPlatform
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 CSS                    19        43491        35195          144         8152
 JavaScript              8        16905        11783         1614         3508
 JSON                    1            9            9            0            0
 Markdown                4          114            0           77           37
 PHP                     8          576          417           71           88
 Python                  3          254          217            8           29
 SQL                     1           22           18            0            4
 SVG                     1          214          214            0            0
 Visual Studio Sol|      1           31           30            0            1
-------------------------------------------------------------------------------
 HTML                    8          676          499           93           84
 |- CSS                  8          112          102            0           10
 (Total)                            788          601           93           94
===============================================================================
 Total                  54        62292        48382         2007        11903
===============================================================================
```

Finally, we experimented using a continuous Github metrics tool that can be used whenever a commit is made (or perhaps enacted on a daily schedule) that updates metrics about a user or the repository. The tool we used was https://github.com/lowlighter/metrics, which has a couple of interesting measurements. A couple of metrics that we could get were per user, so a measurement about general github activity/stats, as well as recent code activity and coding habits, was made, which can be beneficial in determining work activity and practices followed: image



dbgilson

Joined GitHub 1 year ago
Followed by 0 users

Contributed to 7 repositories

Activity
98 Commits
0 Pull requests reviewed
1 Pull request opened
0 Issues opened
0 issue comments

Community stats
Member of 1 organization
Following 0 users
Sponsoring 0 repositories
Starred 0 repositories
Watching 4 repositories

5 Repositories
No license preference
0 Releases
0 Packages
30.8 MB used

0 Sponsors
0 Stargazers
0 Forkers
4 Watchers

These metrics include private contributions
Last updated 13 Apr 2022, 01:57:00 with lowlighter/metrics@3.22.0

Deployment

Currently, our project is being tested by starting up the SQL server and then running the pages in a browser with PHP. In order to deploy this project, we would need to first register a domain name so that users could find our website. Then, we would need to pay a web hosting service to host our website on the internet under our domain name. Finally, we would need to use FTP to transfer our codebase to the website hosting service and set up the SQL database by creating the necessary table, as well as run the scripts to populate our search engine with results from Kaggle and UCI. At this point, our website should be accessible to anyone through a browser.

Lessons Learned

The process of using PHP was fine for our database implementation as the database and website was hosted on the same machine, having little-to-no latency and not requiring too many computational resources. However, PHP has computation done at the server level, and in a more real world application of this project, it's possible that the project implementation could bog down a server having lots of requests as a majority of the project was coded in PHP. It would be beneficial to incorporate Node.js in terms of displaying and formatting the data instead of the server handling it as to better distribute the computational strain.

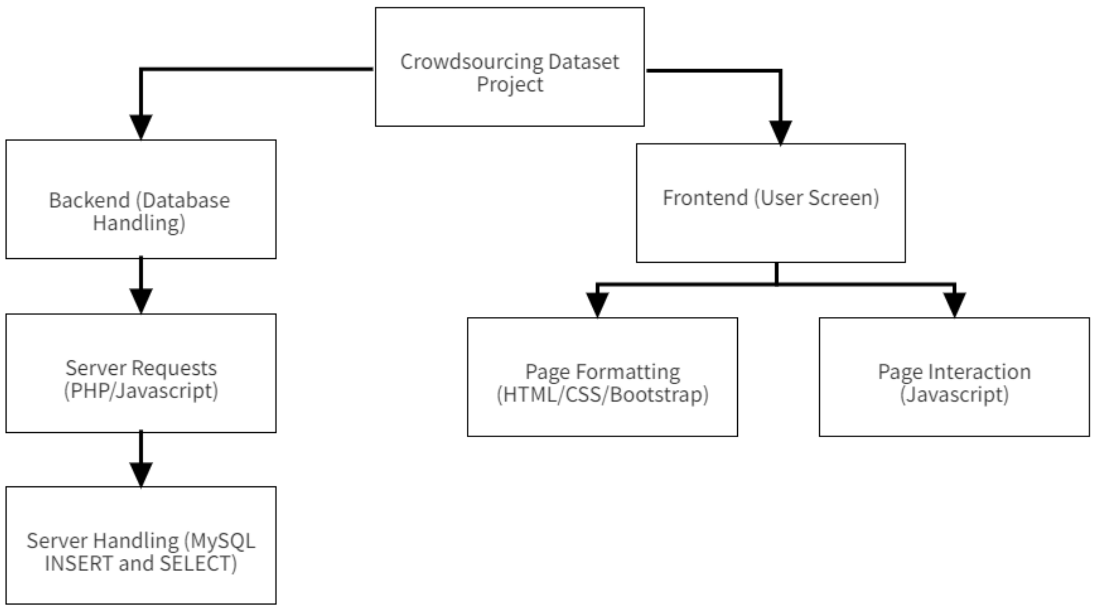
Future Work

There are several features that could be added in the future. First, expanding our dataset functionality by allowing other users to submit modification requests to internal datasets. This would enable our site to be a collaboration space for machine learning developers. After, the biggest improvement would be to implement support for any type of datasets (csv's, audio, pickle files, text files, etc). As of now, our website is limited to only images.

Another future addition would be to create a ranking system when searching for datasets, possibly through metrics like number of downloads or number of contributions (could filter by this as well). This should make seeing more active and reliable datasets much easier instead of simply searching by keywords.

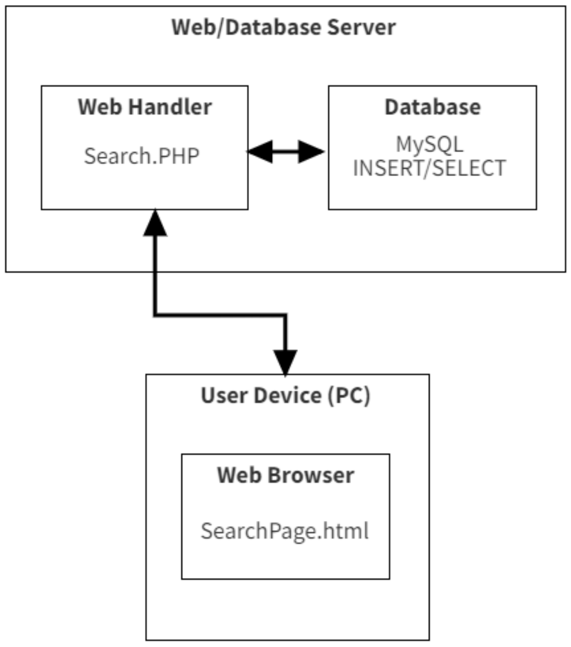
Architectural Views

Logical View



Crowdsourcing Dataset Project
Backend (Database Handling)
Server Requests (PHP/Javascript)
Server Handling (MySQL INSERT and SELECT)
Frontend (User Screen)
Page Formatting (HTML/CSS/Bootstrap)
Page Interaction (Javascript)

Deployment View



Web/Database Server
Web Handler
Search.PHP
Database
MySQL INSERT/SELECT
User Device (PC)
Web Browser
SearchPage.html