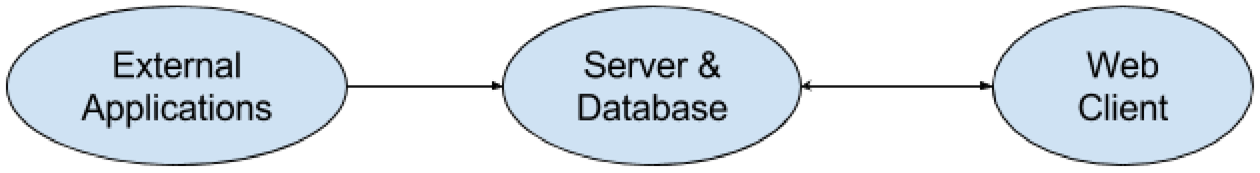
Group Name: Bug Dump
Group Members: Palak Parwal, Michelle Wang, Natalie Nguyen, Stephanie Tong, Ryota Saito, and Kevin Hsieh

README: Stationerry

Problem

The problem that this project helps solve is the lack of feedback that developers get about their projects or applications. This is a serious problem because without some kind of information about the performance of their application, a developer cannot work to improve the project and fix the errors. This kind of information could potentially improve the functionality of an application for developer.

Architecture



Our project is made up of three parts. These parts are external applications, a server with a database on it, and a web client.

External Applications
The external applications are applications made by users that they will connect to our servers to send the database error logs for their application(s).

For the purpose of this project our external applications portion is an a simple application that sends error logs to the server to be entered into the database. It allows us to show the functionality of the web client and how it connects to the server and database.

The error reports from the project are sent to a website that we have set up. This website has Django which calls a function every time it receives something. This function will parse the error report and will sent the information to the database.

Server and Database
The server is setup to connect to both the external applications created by users as well as the web client that displays statistics and information about the application a user has connected to it.

Our database contains three tables as detailed below:

1. auth_user
    1.1. ID (integer) (primary key)
    1.2. Password (varchar)
    1.3. Last_Login (datetime)
    1.4. Is_SuperUser (boolean)
    1.5. First_Name (varchar)
    1.6. Last_Name (varchar)
    1.7. Email (varchar)
    1.8. Is_Staff (boolean)
    1.9. Is_Active (boolean)
    1.10. Date_Joined (datetime)
    1.11. Username (varchar)
2. App
    2.1. App_Name (text)
    2.2. App_Version (text)
    2.3. Platform (text)
    2.4. username (foreign key)
3. BugReport
    3.1. Model_Name (text)
        3.1.1. This denotes the brand of the device
    3.2. Model_OS (text)
    3.3. Error_Type (text)
    3.4. Error_Message (text)
    3.5. Time (datetime)
    3.6. Status (text)
    3.7. Device_Model (text)
        3.7.1. This denotes the actual model of the device
    3.8. App_Name (foreign key)

The auth_user table is created by Django so we do not control the information it saves. It does cover all of the information we wanted to save so it works correctly for our purposes.

Django also creates primaries keys itself for the App table and the BugReport table.

The auth_user table contains the information for users that are signed up with the web client to use the features available on it.

The App table contains information about the applications that are connected to the web client. The foreign key, username, connects the application to the user that added it the web client. This makes sure a user only access information about a application they have connected.

The BugReport table is where the error logs from the external applications are stored. This table is the one that the search feature will look through to output statistics and information about the application.

The server populates the database by parsing the errors that come from the external applications. The language we use for parsing the incoming information is Python.

Web Client
The server also connects to the last portion of the project, the web client. This is a website where a user interacts with our system to obtain information about the website. The website will allow users to create accounts and connect projects to them. It will then use the error reports from the application to displays statistics about the errors. The site will also allow users to search through the error reports for specific information that they may want.

Features
The features for this project include the ability to create an account and connect an application to it, a search bar that will allow users to go through their error reports for more specific information, and statistics that give a general overview of the performance of the application.

Accounts
To create accounts, the information the user gives to the signup form will be sent to the Django Administration system so that the account can be created based on Django's model for a user. This information is sent using Ajax. The system will add the user to the auth_user table from where the user will be verified on subsequent logins. The system will make sure a user can only access their own application information.

Statistics
The statistics feature of the web client is most prevalent on the Dashboard when a user logs in. It gives users information about the number of errors ever reported, new errors, unique errors, and warnings. The Dashboard also includes a graph of the number of errors over the last week as well as details about the most common errors. It is a useful feature for quick and general information about an application.

Search Bar

Another feature that the web client will provide is a search bar. This will be available under the Errors tab on the website. Users will be able to search the errors in the BugReport table to find specific information about the error reports from their application. This can help if they are tackling a specific problem with their application.

User Input

There are very few places where the web client takes in information from the user. These are the signup page, the login page, and the search bar.

Signup Page

The signup page takes in the new user's name, username of choice, password, and an email to connect the account to. This information is parsed by Django to create the user and add their information to the database.

Login Page

The login page takes in the username and password of a user. The information will be compared to the information stored in the database. If it matches, the user is let onto the website. If the information provided does not match any entry in the database, the user will not be allowed onto the website and will be prompted for the correct information.

Search Bar

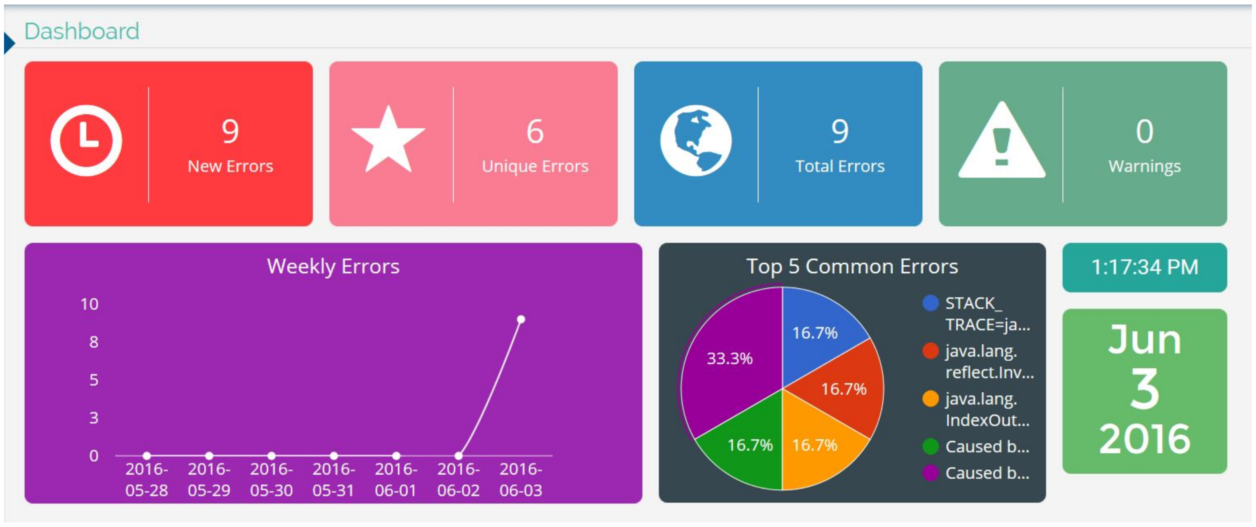
The search bar takes in a string from the user to search through the database for. The results of the search will be displayed on the Errors tab. It will include details about the time the error occurred, the type of error, the message of the error, the name of the application the error originated from, the version of the application, the operating system of the device that sent the error, and the model of the device and sent the error.

This kind of information lets a developer look at which specific platforms are having difficulty running their application. They can then cater to those systems to help ensure the best experience for users.

One of the features of our search bar is that we have filters to help make the search easier to use. You can filter by what you are looking for, the type of error, the name of a specific application, the application version, the application's operating system, the model of the device that sent the error, and the status of the error. These filters help refine our search for the best results.

Analysis

The server takes in error reports from the user's application. This information is parsed using the Google Charts API before it is displayed on the Dashboard of the web client.



The information displayed will follow this kind of format.

Application is available at:
http://stationerry.pythonanywhere.com/

Git repository:
https://github.com/waoimapanda/Stationerry