**Infrastructure**

**Database Schema**

We use MySql to implement our Bugoverflow database. There are totally seven entities which are Bugs, BugLocations, Users, Questions, Answers, VoteTypes and PostFeedback. All bugs’ information all store in Bugs’ tables and the relative location data will keep in the Buglocations’ tables. Users can create questions to report the new bugs which they found or answer other users’ questions. Everyone can vote these questions and keep these records in the table of PostFeedback. We can use PostFeedback to rank these questions and find which one is more popular.

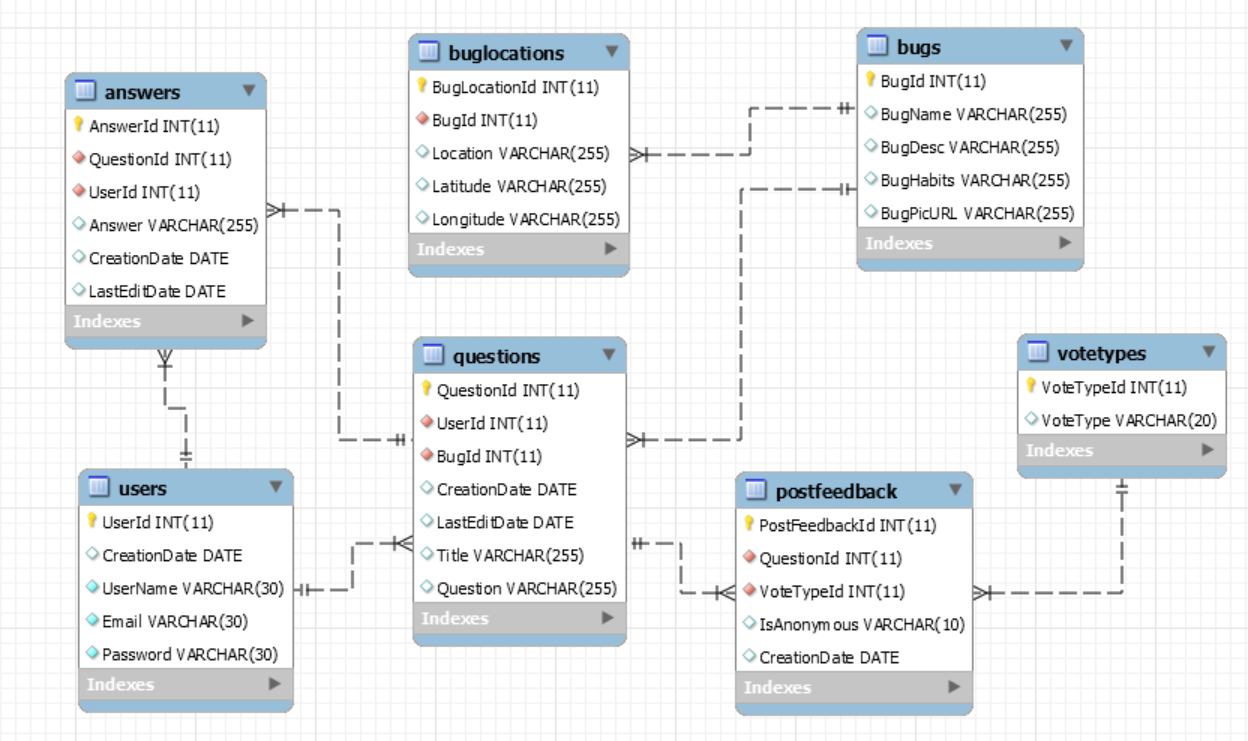


Figure: Current Database Schema of Bugoverflow

The table, Bugs will store the bugs’ data which include bugs’ names, descriptions, habits and pictures. Specifically with pictures, we will store them in the ASW Cloud and keep their URL in the table.

The table, BugLocations will store the bugs locations’ data which include the locations the bugs appeared, the latitude and longitude related to the location. Basiclly, we will use the latitude and longitude as the coordinate to plot the location in the Googel Map. Users can offer all these information or some of them. If users only tell us the locations, we will convert them into appropriate latitude and longitude.

The table, Users will keep users’ information which include the dates when users created, usernames, their emails and passwords. The created data will be generated by the system. User name should be unique. If duplicated, the system should ask users to create another username. The passwords should be encryped and we will store the hashed password in the database.

The table, Questions will have the data of quesitons which include the questions’ titles, their contents, created dates and revised dates. Created dates and revised dates are all generated by the system. Revised dates can be updated many times and the database only keep the last update.

The table, Answers will have the data of answers which include the answers, referred questions’s ides, created dates and revised dates. Like the table of Questions, created dates and revised dates in the table of Answers are all generated by the system. Revised dates can be updated many times and the database only keep the last update.

The table, VoteTypes will have three types of votes. They are good, ok and bad. Users can choose one of these valuse to vote for questions.

The table, PostFeedback will store the data of feedbacks which is other users vote for the questions. The attributes in the table inculde related question ids, vote options from the table of VoteTpes, anonymous vote or not and created date. We will assume these votes all anonymous and will change it if users log in and vote. Like the table of Questions, created dates in the table is generated by the system.

**Database function**

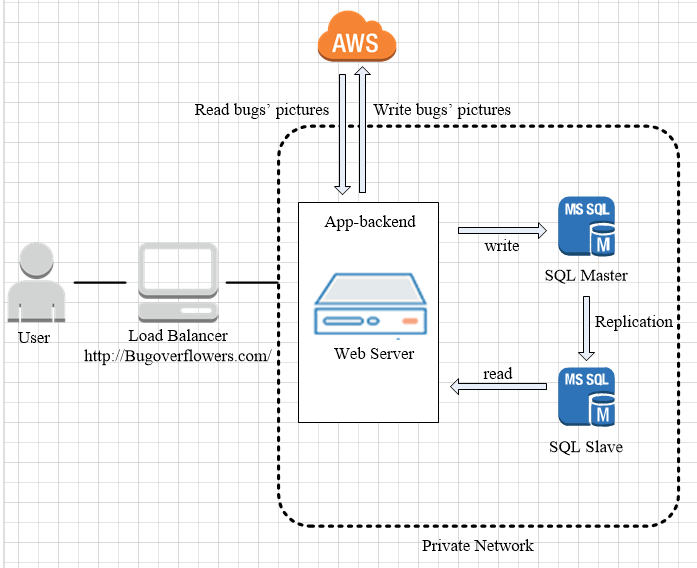


Figure: Current Infrastructure in DEV environment

In our DEV environment, we will create three vagrant boxes with three different static addresses. One is web server which will hold the application backend. The other two boxes are designed for the Master/Slave replication in the database. They present two databases in two servers, SQL Master and SQL Slave. Our application will send the data to SQL Master and read the data from SQL Slave.

**Improvements**

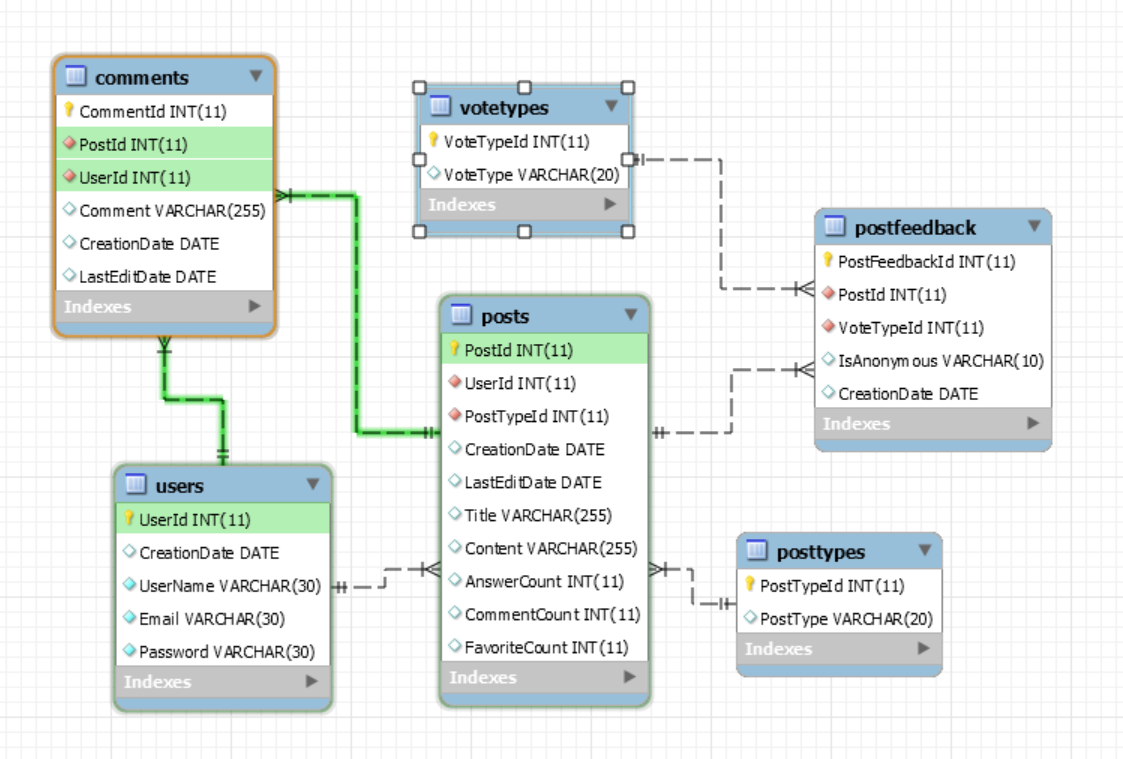


Figure: Database Schema of Bugoverflow in old version

When we created the database schema, we choose to use the second way of applying Lean principles to the technology value stream which Gene Kim lists in the book of DevOps Handbook1. This way is from Ops to Dev, the flow feedback. We need to understand and respond to the needs of all customers, whatever internal and external. According to lean thinking, we realize that the most important customer is the internal one – the next downstream work center, Dev. Hence, we got the feedback from Dev and revised the database schema. For example, in the old version of database schema, users will create posts which are questions or answers. Users can add comments after each question or answer. Dev went through this schema has too much layers and it was too complex to implement. Then, we changed and use this current one.

In the future, we will create A/B test in our daily work and everyone can test the database function. If anyone finds some place need to improve, he or she can revise the database. It is the third way of applying Lean principles to the technology value stream which Gene Kim lists in the book of DevOps Handbook1. In this way, we can create the continual experimentation and learning. We will break things early and often. Finally, we will makes deploys go more smoothly.

Work Cited:

1. Gene Kim, Patrick Debois, John Willis, Jez Humble, “The Devops Handbook: How to Create World-class Agility, Reliability, and Security in Technology Organizations.” IT Revolution Press, LLC. 2016

**Appendix:**

Database tables:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Bugs** | | | | |
| **Column** | **Attributes** | **Data Type** | **Null or Not Null** | **Explanation** |
| 1 | BugId | INT | Not Null | Primary Key |
| 2 | BugName | VARCHAR (255) | Null | The bug's name could be null first. After identified and offered the answer by other users, we can add its name. |
| 3 | BugDesc | VARCHAR (255) | Null | Description or definition for the bug |
| 4 | BugHabits | VARCHAR (255) | Null | Habits column should be bug's favorite food or living environment. |
| 5 | BugPicURL | VARCHAR (255) | Null | We store the bugs' pictures in the cloud and offer the web links. |
| **Table: BugLocations** | | | | |
| **Column** | **Attributes** | **Data Type** | **Null or Not Null** | **Explanation** |
| 1 | BugLocationId | INT | Not Null | Primary Key |
| 2 | BugId | INT | Not Null | Foreign Key for the table of Bugs |
| 3 | Location | VARCHAR (255) | Null | Users can offer the data of location or latitude and longitude. If users only offer the location, we need using PHP to transfer it into attitude and longitude. |
| 4 | Latitude | VARCHAR (255) | Null |  |
| 5 | Longitude | VARCHAR (255) | Null |  |
| **Table: Users** | | | | |
| **Column** | **Attributes** | **Data Type** | **Null or Not Null** | **Explanation** |
| 1 | UserId | INT | Not Null | Primary Key |
| 2 | CreationDate | Date | Not Null | This data is auto generated by the system. |
| 3 | UserName | VARCHAR(30) | Not Null | User name should be unique. If duplicated, the system should ask users to create another username. |
| 4 | Email | VARCHAR(30) | Not Null |  |
| 5 | Password | VARCHAR(30) | Not Null | We will store the hashed password in the database |
| **Table: Questions** | | | | |
| **Column** | **Attributes** | **Data Type** | **Null or Not Null** | **Explanation** |
| 1 | QuestionId | INT | Not Null | Primary Key |
| 2 | UserId | INT | Not Null | Foreign Key for the table of Users |
| 3 | BugId | INT | Not Null | Foreign Key for the table of Bugs |
| 4 | CreationDate | Date | Null | This data is auto generated by the system. |
| 5 | LastEditDate | Date | Null | Question can be edited by users and this data is auto generated by the system. |
| 6 | Title | VARCHAR (255) | Null |  |
| 7 | Question | VARCHAR (255) | Null | This is the content of the question. |
| **Table: Answers** | | | | |
| **Column** | **Attributes** | **Data Type** | **Null or Not Null** | **Explanation** |
| 1 | AnswerId | INT | Not Null | Primary Key |
| 2 | QuestionId | INT | Not Null | Foreign Key for the table of Questions |
| 3 | UserId | INT | Not Null | Foreign Key for the table of Users |
| 4 | Answer | VARCHAR (255) | Null | This is the content of the answer. |
| 5 | CreationDate | Date | Null | This data is auto generated by the system. |
| 6 | LastEditDate | Date | Null | Question can be edited by users and this data is auto generated by the system. |
| **Table: VoteTypes** | | | | |
| **Column** | **Attributes** | **Data Type** | **Null or Not Null** | **Explanation** |
| 1 | VoteTypeId | INT | Not Null | Primary Key |
| 2 | VoteType | VARCHAR (20) | Null | There are three types of values, good, ok and bad. |
| **Table: PostFeedback** | | | | |
| **Column** | **Attributes** | **Data Type** | **Null or Not Null** | **Explanation** |
| 1 | PostFeedbackId | INT | Not Null | Primary Key |
| 2 | QuestionId | INT | Not Null | Foreign Key for the table of Questions |
| 3 | VoteTypeId | INT | Not Null | Foreign Key for the table of VoteTypes |
| 4 | IsAnonymous | VARCHAR (10) | Not Null | Choose the default value, Yes. |
| 5 | CreationDate | Date | Null | This data is auto generated by the system. |