Logo

Description automatically generated

**WEB APPLICATION DEVELOPMENT**

Group 5 – Topic: Bugtracker

*Prepared by:*

VÕ CÔNG KHA, ITITIU18206  
PHẠM ĐĂNG KHOA, ITITIU18276  
HÀ MINH CHIẾN, ITITIU18302  
PHẠM HÀNG ANH TUẤN, ITITIU18179

**PROJECT PROPOSAL REPORT**

*Dr: N. V. Sinh*

**CATALOGUE**

**INTRODUCTION**

* Motivation and Problem Statement…………………………………..1
* Learning Outcome……………………………………………………….1
* Contribution rates.………………………………………………………1

**LITERATURE REVIEW**

* Platforms and Tools used
  + React .js…………………………………………………………..…2
  + Express.js………………………………………………………..…2
  + MySQL……………………………………………………………….2
* Similar Systems…………………………………………………………..3

**SYSTEM DESIGN**

* Planning……………………………………………………………………4
* Specific details
  + Entity Relationship Diagram…………………………………….5
  + Table Relationship Diagram…………………………………….6
  + Use Case Diagram………………………………………………...7
  + Source code details...…………………………………………….7
  + MySQL schema generation code..……………………………12

**IN-DEV IMAGES**

* Images of the in-development program…………….……………..14

**CONCLUSION**

* List of completed works………………………………………………19
* Pros and Cons…………..………………………………………………19
* Future Works ……………………………………………………………19

**INTRODUCTION**

1. **Motivation and Problem Statement:**

When running applications on computers, users may encounter various unwanted features, errors that may or may not cause the application to execute thoroughly. Such features or errors like those are known as bugs or glitches.

Although those bugs or glitches might possibly give users some unexpected advantages, they make user’s experience go bad overall and generally. In a life cycle of an application, especially in the video games industry, bugs and glitches appearance is inevitable. No matter how careful and accurate a developer or software designer is, they will always encounter them numerous times. Another problem is that, developers and software designers often do not have time to manually search for hidden bugs in their creations. Therefore, they ask users of the application for help. Because user base is vast, every bug which user discovers will save developers lots of time. All they need to do is to collect the complete set of bugs and glitches and resolve them for the next application’s update.

That’s why the Bugtracker system is crucial to every computer applications and group 5 of the Web Application Development class had started to develop a system used to create reports of bug for the staff team to handle.

1. **Learning Outcome:**

* Understand the Model – View – Control structure of a Web Application.
* Advance skills in designing databases in application.
* Advance skills in developing Web’s frontend and backend.
* Develop skills in programming with various Web Frameworks.

1. **Contribution Rates**

* Võ Công Kha - ITITIU18206 (25%)
* Phạm Đăng Khoa - ITITIU18276 (25%)
* Hà Minh Chiến - ITITIU18302 (25%)
* Phạm Hàng Anh Tuấn - ITITIU18179 (25%)
* More information in the Planning section page 4

**LITERATURE REVIEW**

**Platforms and Tools used:**

1. React.js

* React.js is an open – source JavaScript library used for front-end user interface or components development. It was created originally by Jordan Walke and is currently maintained by Facebook and other community developers.
* Based on various online surveys, it indicates that React has been one of the most popular assets to create user interfaces. Mainly because of its high performance, flexibility and the powerful support community.

1. Express.js:
   * Express.js is an open – source web application framework for Node.js used for designing and building backend of web applications. Thanks to Express written in JavaScript, it gives programmers and developers easier to build web applications.
   * It is lightweight and can help organize one’s web application on the backend side into a secure Model – View – Control structure.
2. MySQL:
   * MySQL is an open – source relational database management system (RDBMS).
   * It works with the host’s operating systems to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.
   * MySQL is responsible to power storage solution of many popular websites, such as Flickr, MediaWiki, Twitter, YouTube and Facebook.

**Similar Systems**

* Bugify: bug tracking system, free for users to report bugs.

**Graphical user interface, application, website

Description automatically generated**

*Bugify UI (https://bugify.com)*

* Mojang’s Bugtracker: Bug Tracking System by Mojang, used for reporting bugs and glitches of their products (Minecraft, Cobalt, …)

Graphical user interface, application

Description automatically generated

*Mojang’s bugtracking system (https://bugs.mojang.com/projects/MC/issues)*

**SYSTEM DESIGN**

1. **Planning:**

|  |  |
| --- | --- |
| **Timestamp and Duration** | **Activities** |
| Week 1 | Research phase for Topic (ALL) |
| Week 2 | Fundamental knowledge preparation (ALL) |
| Week 3 🡪 4 | Resource Planning (ALL) |
| Week 5 🡪 8 | Implementation:   * Backend: Kha, Khoa   + Create and design databases.   + Connect backend web server to frontend application     - Declare Routes and Controllers.     - Declare connection between Express app and DBMS.     - Secure connection between React and Express apps.     - Considers possible vulnerability. * Frontend: Tuan, Chien   + General Design + Divide UI Components: Tuan, Chien   + Implements React components:     - Login, CustomerRegister, BugView, Message.     - AssignBug, StaffView, ProjectView, CreateBug. |
| Week 9 | Testing and Review:   * UI: Tuan, Chien * DB and backend: Kha, Khoa |
| Week 10 | General Test (ALL)  Issue report (Kha, Others Supporting) |

1. **Specific details**
2. Entity Relationship Diagram

**Diagram

Description automatically generated**

When the database is formatted, it forms this table:

|  |
| --- |
| people: (id, name, username, password, role) |
| bug: (id, title, description, *staffID, userID*) |
| update: (*authorID, bugID,* time, content) |

1. Table Relationship Diagram

Diagram

Description automatically generated

1. Use Case Diagram

Diagram

Description automatically generated

1. Source code details:

* Backend:

Text

Description automatically generatedText

Description automatically generated

*App.js and config.js at src/*

* + The main App.js file for starting the backend server.
  + On the right are the information that the backend server uses. This includes host, password, database name, IP address of the database. In addition, tokenExpires regulates when will a user’s token timeout so that they will have it renewed in order to continue operation on the web application.

Text

Description automatically generatedText

Description automatically generated

*auth.js file in src/controllers/*

* + This is the main core authorization of login, package jsonwebtoken is used for replacing sessions. Furthurmore, bcrypt package is also used to hash classified credentials of user, giving them an extra layer of security.

Text

Description automatically generated Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

*bug.js file in src/controllers/*

* + bug.js provides all of the operations needed to interact with bugs, such as: createBug, getBug, updateBug, …
  + bug.js contains various queries to the SQL database.

Text

Description automatically generated

*database.js at src/controllers*

* + database.js regulates the constant info of the database used for the application. This file is created for multiple use at different locations in the application directory tree.

Text

Description automatically generatedText

Description automatically generated

*people.js at src/controllers/*

* + people.js provides all information about the user and interactions.
  + people.js contains various queries to the SQL database.

Text

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

*Routing .js files at src/routes/ (bug.js, people.js, login.js) respectively*

* + These files regulates the routing between React app and Express server, allowing crossing over data transfer.
* Frontend: (click [here](https://github.com/Khoa100500/WebApplicationProject-BugTracker/tree/main/frontend/src) to view the source codes)

1. MySQL schema generation code:

**CREATE TABLE `bug\_report`**

**(**

**`bug\_id` binary(16) NOT NULL,**

**`title` varchar(200) DEFAULT NULL,**

**`bug\_description` text,**

**`bug\_status` bit(1) DEFAULT NULL,**

**`userID` binary(16) DEFAULT NULL,**

**PRIMARY KEY (`bug\_id`)**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

* **Initialize ‘bug\_report’ table**

**CREATE TABLE `bug\_update`**

**(**

**`bug\_id` binary(16) NOT NULL,**

**`content` text,**

**`update\_time` timestamp NULL DEFAULT NULL,**

**`authorID` binary(16) DEFAULT NULL,**

**KEY `update\_report\_idx` (`bug\_id`),**

**CONSTRAINT `update\_report` FOREIGN KEY (`bug\_id`) REFERENCES `bug\_report` (`bug\_id`) ON DELETE CASCADE ON UPDATE CASCADE**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

* **Initialize ‘bug\_update’ table**

**CREATE TABLE `people`**

**(**

**`id` binary(16) NOT NULL,**

**`p\_name` varchar(200) DEFAULT NULL,**

**`role\_id` int NOT NULL,**

**`pass\_word` varchar(255) DEFAULT NULL,**

**`user\_name` varchar(200) NOT NULL,**

**PRIMARY KEY (`id`),**

**UNIQUE KEY `user\_name` (`user\_name`),**

**KEY `people\_role\_idx` (`role\_id`),**

**CONSTRAINT `ppl\_role` FOREIGN KEY (`role\_id`) REFERENCES `people\_role` (`role\_id`) ON DELETE CASCADE ON UPDATE CASCADE**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

* **Initialize the ‘people’ table**

**CREATE TABLE `people\_role`**

**(**

**`role\_id` int NOT NULL,**

**`role\_name` varchar(5) NOT NULL,**

**PRIMARY KEY (`role\_id`)**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

* **Initialize the ‘people\_role’ table**

**CREATE TABLE `work\_on` (**

**`staffID` binary(16) DEFAULT NULL,**

**`bug\_id` binary(16) DEFAULT NULL,**

**KEY `staffID` (`staffID`),**

**KEY `bug\_id` (`bug\_id`),**

**CONSTRAINT `work\_on\_ibfk\_1` FOREIGN KEY (`staffID`) REFERENCES `people` (`id`),**

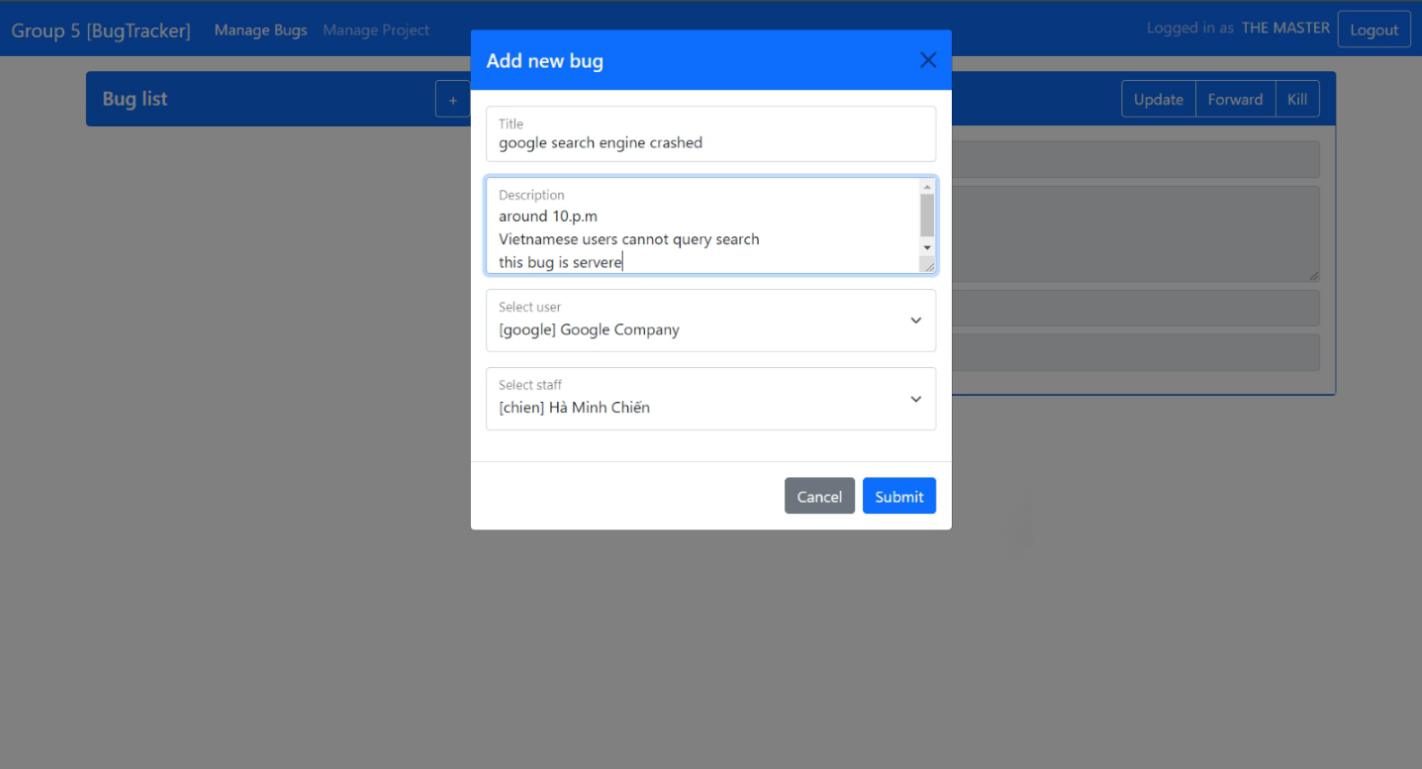
**CONSTRAINT `work\_on\_ibfk\_2` FOREIGN KEY (`bug\_id`) REFERENCES `bug\_report` (`bug\_id`)**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

* **Initialize the ‘work\_on’ table**

**IN-DEV IMAGES**

**Images of the In – Development application:**



*Adding new bugs*

*Graphical user interface, text, application, email

Description automatically generated*

*Bug view*

*Graphical user interface, application

Description automatically generated*

*Editing Staff information*

*Graphical user interface, text, application, email

Description automatically generated*

*Forwarding bugs*

*Graphical user interface, text, application, email

Description automatically generated*

*Killing bug*

*Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated*

*Project Management UI*

*Graphical user interface, application

Description automatically generated*

*Adding new Staff to team*

*Graphical user interface, application

Description automatically generated*

*Adding new User*

*Graphical user interface, application

Description automatically generated*

*Removing / Demoting Staff*

*Graphical user interface, text, application

Description automatically generated*

*Update Bug Status*

**CONCLUSION**

1. **List of completed works**

* Designed the UI.
* Succesful implementation of backend Express server.
* Testing phase shows pleasant and expected results. All errors had been handled.

1. **Pros and Cons**

* Pros:
  + Information strictly secured by using bcrypt library.
  + Lightweight, High Performance Application.
  + Async, Await implemented to avoid the [Pyramid of Doom](http://callbackhell.com/).
  + Try/Catch the Tower of Terror.
* Cons:
  + Media such as images or videos are not supported.
  + UI design is still decent, not very eye – catching.

1. **Future Works**

* Extend the bug creation system to include media.
* More development on the web user interface.
* Implementation of search bug function.
* Integrate into mobile app with React Native.