Crime Buster Application

Code Inspection Report

_____

Presented to

Mr. Jonathan Pautsch of

Next Century Corporation

_____

Software Engineering I

CMSC 447

_____

By

Angel Cheng

Sam Mendimasa

Katelyn Seitz

Zach Vance

25 April 2018

Crime Buster Application
Code Inspection Report

Table of Contents

1. Introduction

1.1 Purpose of This Document

The purpose of this document is to provide an overview of the coding practices that were adhered to in the creation of the Crime Buster web application. The description of practices include the coding and commenting conventions, as well as description of any software defects. A review of all of the systems current functionality and the team meetings are outlined. The intended audience for this product are the Baltimore Police Department, Next Century Corporation, and Dr. Wilson of the University of Maryland, Baltimore County.

1.2 References

1. Crime Buster System Requirements Specification
2. Crime Buster System Design Document
3. Crime Buster Testing Report

1.3 Coding and Commenting Conventions

We, as a team, decided to use the camel case naming conventions for variables, methods, and classes. This decision was based on the added readability and writability from this convention. Camel case is easier to type than several other naming conventions such as using underscores and is also easy to read as many programmers are use to it. Our coding and commenting conventions are adapted from the IEEE coding standards.

1.4 Defect Checklist

We decided to categorize our defects in the following categories: logic errors, security oversights, commenting errors, and coding convention errors. Please see table 1 below for the details of each of those categories. We list the defects that we found during the code review in table 2 below. We realize how important this piece of the code review is as defects in code leads to bugs and also makes it more difficult to maintain.

Table 1: Defect Categories

Category	Comments
Logic Errors	Logic errors occur when the code does not execute the way the programmer thought that

	it would. The logic was mistaken and the code performed different than was intended.
Security Oversights	A security oversight occurs when the code is vulnerable and has entry points for hackers to take advantage of the web application to do actions the application was not intended for. It may also provide the hacker access to the underlying computer system.
Commenting Errors	Commenting errors occur when the comments are not consistent over a large portion of code or are missing all together.
Coding Convention Errors	A coding convention error is when the code deviates from the normal coding convention followed by the rest of the code. These errors make the code difficult to maintain and hinders the ability of the code to evolve.

Table 2: Defect Checklist

Category	Defect
Logic Error	SQL filter query and / or logic incorrect.
Logic Error	Incorrect method called
Logic Error	Parameters do no match
Logic Error	SQL counts returning as objects instead of the expected integers.
Commenting Error	Missing comments
Commenting Error	Too many comments
Commenting Error	Too much broken code left commented out, hurts readability
Coding Convention Error	Variable naming not the best or most logical
Coding Convention Error	Some variables use underscores rather than the camel case naming convention.
Coding Convention Error	Method naming not the best or most

	logical
Security Oversights	Measures to avoid Cross-Site Scripting (XSS) have not been implemented
Security Oversights	The user data is not currently encrypted.

2. Code Inspection Process

2.1 Description

Coding inspection was completed by the programmers who wrote the code while simultaneously navigating through that part of the application. Based on different use cases, it was also inspected by teammates who did not write code for that particular part. In future spirals, code will be reviewed by a coder who's not on our team.

Our code was mainly reviewed and completed remotely using Github because of conflicting schedule, and different teammates takes different parts in the code. Our team meet twice in person and at least once online. Coding review is a long process, so each of us reviews our own part first and then meet together and review each assigned part together. In this way, it is efficient and easy to communicate.

2.2 Impressions of the Process

Our code inspection process has been mainly effective because we usually meet in person. However, we do not have a special document for people to write down and record the features and defects throughout the inspection, and each person has their own notes, it can be hard to find the notes or recall the memory for this document because we do not have a combined document, and this can be improved in the future.

Currently, the areas of our code that are least likely to have code defects are segments that have been reviewed by more than one teammate. Thus, all the file in db/ folder and mapVis.js are least likely to have remaining flaws, while index.php and table.js are more likely contains coding flaws because some of it is half completed and some are not fully inspected.

2.3 Inspection Meetings

During inspection meetings, different parts of code was reviewed by different teammates at the same time, and update the code simultaneously. Please refer to the Meeting Table below, Table 3.

Table 3. Meeting TImes

Date	Time	Attendance
4/7/2018	2:00pm-5:00pm	All Team Members
4/12/2018	8:30pm-10:00pm	All Team Members
4/14/2018	3:00pm-6:00pm	All Team Members
4/18/2018	5:30pm-8:30pm	Zach, Sam
4/28/2018	10:00am-4:00pm	All Team Members
4/29/2018	4:00pm-7:00pm	All Team Members

3. Modules

CrimeBuster is a web based application written in html, javascript, php, and it uses vue.js and c3.js framework. The javascript components of the application are used to create the visualizations, while the php files, serves as the endpoints that return datasets base on given parameters. The structuring of the files in the application are below. Please refer to the coding description table (table 4) below to view files with associated brief description of their functionality. Table 5 contains further description for files that are currently been implemented, but were not completed at the submission of this document.

3.1 Modules Inspected

Table 4. Coding Description Table - Files Completed

#	File	Brief Description
1	index.php	Main page that structure the application
2	styles/style.css	CSS styling for various div's in the main file
3	styles/style2.css	Additional CSS styling for containers used in main file
4	styles/w3.css	CSS styling that manages filters

5	images/avatar2.png	Default image for users
6	db/mydb.db	Sqlite database file
7	db/filterQuery.sql	Sql queries for filtering the database
8	db/dbEndPoint.txt	API address and key for updating datasets
9	db/sql_code.sql	Sql code for creating and adding data to the database
10	db/sql_queries.sql	Sql code that creates the users and comments table
11	db/getMapLocations.php	Php code to access and pull location data for the map
12	db/getActualDescriptionData.php	Php code to access and pull summary of description data for the charts visualizations
13	db/getActualDistrictData.php	Php code to access and pull summary of district data for the charts visualizations
14	db/getActualWeaponChartData.php	Php code to access and pull summary of weapon data for the charts visualizations
15	db/getLineChartData.php	Php code to access and pull data for the line graph visualization
16	c3-0.5.3/	C3.js library
17	scripts/vue.js	Vue.js file
18	scripts/chartVis.js	Makes charts visualizations using c3 library
19	scripts/mapVis.js	Makes map visualization using google map api
20	scripts/lineGraph.js	Makes line graph using c3 library
21	scripts/scripts.js	Has starter javascript functions for the web application
22	scripts/timeLine.js	Makes the timeline visualization
23	scripts/table.js	Makes the table visualization
24	db/getTableData.php	Access the database and pull the require

		data for the table visualization
25	scripts/heatMap.js	Makes the heatMap visualization
26	db/getHeatMapData.php	Access the database and pull the require data for the heatMap visualization

3.2 Modules that have not been completed

Table 5: Coding Description Table - Files that will be written soon

#	File	Brief Description	Projected Completion Date
1	index_compareCharts.php	Code to be able to view two charts side by side.	Version 2 of Crimebuster

All other files were completed for the final presentation.

4. Defects

The following table reviews the defects found in our system.

Module	Correctness	Coding Violation	Commenting Violation	User Friendliness	Fixed
Index.php (display tooltip)	"For full entry, click here" leads nowhere useful	None detected	No comments	Website linked has nothing to do with description	Yes
Index.php (Location/ Premise filter)	Does not filter data at all	None detected	No comments	Header name is a bit ambiguous	Yes
Index.php (various menu items)	Serve no function. Do absolutely	Buttons implemented without functions or	No comments	Buttons imply function. If they do nothing, the	Yes

	nothing when clicked	listeners		application is better off without them	
Index.php (general)	None detected	None detected	Commented code remains when it should be deleted	None detected	Yes
index.php	“View comments” does not do anything	Buttons implemented without functions or listeners	No violation	No violation	No, not implemented yet
index.php	The mail, user, and setting buttons on the top right do nothing useful.	Buttons implemented without functions or listeners	No violation	No violation	No, not implemented yet

9. Appendix A – Agreement Between Customer and Contractor

The customer agrees to a crime data visualization application that uses data from the open baltimore site to create various maps, charts, and graphs which will visualize crimes in baltimore. Use cases are included in the functional requirements section above of the behavior between the system and user. Additional features will be provided in further development spirals. When and if future changes to this document occur a drafted new document will be created. An electronic copy of both versions will be presented to the client for review. Upon approval, the draft will be finalized and signed off by both parties.

Client

Name: Date:

Signature

Comments

Team

Name: Date:

Signature

Name: Date:

Signature

Name: Date:

Signature

Name: Date:

Signature

10. Appendix B – Team Review Sign-off

This document has been collaboratively written by all members the team. Additionally, all team members have reviewed this document and agree on both the content and the format. Any disagreements or concerns are addressed in team comments below.

Team:

Name: Date:

Signature

Comments

Name: Date:

Signature

Comments

Name: Date:

Signature

Comments

Name: Date:

Signature

Comments

Appendix C – Document Contributions

Throughout the development of this document, each team member contributed in some way, hence the overall work distribution split evenly across all members (Angel 25%, Sam 25%, Katelyn 25%, and Zach 25%). The specific breakdown of work contribution is also divided across all sections, as we all worked on this document together.