# ISSUE TRACKING

Group 3E – Doctors Program

Harry Hacker, Jamie White, Tiberius Paharnicu & William Grice

# CONTENTS

# ISSUE TRACKING

## HOW WE TRACKED ISSUES

We decided the best way to track issues (that may or may not be ongoing) was to create a google spreadsheet and create tables to better organise ourselves. The tables we used were, the people (the members of the group), the bugs table (which had each bug we encountered as well as an ID for it) and finally a sprint table that holds the ID of each bug we had for that sprint, who found and figured out the bug as well as if it had been fixed yet.

### WHY WE CHOSE TO DO IT THIS WAY:

We chose to do it this way because it seemed to make the most sense at the time. It allowed us to see any issues other group members were having and work as a team to try and find a solution to that problem. Many of these bugs did not take too long to figure out but were still useful for each of us to go to in case there was a similar problem someone else had already faced.

## THE TABLES

### PEOPLE

| People | |
|---|---|
| Person ID | Name |
| HH | Harry Hacker |
| JW | Jamie White |
| TP | Tiberius Paharnicu |
| WG | William Grice |

Each person has an ID (just made up of the first letter of their forename and surname). In larger groups this would be an ideal way to keep track of whose having difficulty with what bug & who to go to for help if you are encountering something similar.

### BUGS TABLE

| Bugs Table | | |
|---|---|---|
| Bug ID | Feature | Description |
| 1 | GitLab | Driver needed to connect to the database can't be found when downloaded to someones computer. |
| 2 | Login Screen | The program can't find the correct username and password for the user trying to login. |
| 3 | Login Screen | Error message wouldn't show for an incorrect username, but would if nothing was entered. |
| 4 | View Month & Year | Can't access database in any way. |
| 5 | Login Screen | Panel wont load after the new messages screen. |
| 6 | View Month & Year | Problem with grabbing month and year based off of the date format. |
| 7 | Creating Database | Script to create database doesn't register the doctor's table. |
| 8 | Main Page | More issues with the grid layout not forming correctly. I'll need to come back to this later. |
| 9 | Edit Visit Details | Unable to change cells in the table. |
| 10 | View Visit Details | The program currently shows everyones appointment information instead of only showing the current doctor. |
| 11 | Edit Visit Details | Unable to check what the appointment ID is (Less of a bug, there's no feature yet implemented that shows this.) |
| 12 | Program | Program won't run despite there being no reason not to. (Figured it out for anyone having the same issue. I was missing the classpath file, may save you some time) |
| 13 | Edit Visit Details | Program doesn't select the appointment by the appointment ID. |
| 14 | Summary Information | Same issue as bug 10 but for the summary information. (shows everyones information to the same doctor) |
| 15 | Summary Information | For loop wont go through and create a new row for each patient's summary information. After one time it stops running, needs a fix quickly. |
| 16 | View Visit Details | ArrayIndexOutOfBoundsException needs to be solved. Shouldn't take long. |
| 17 | Edit Visit Details | Unable to get information on an appointment and display it to the user. |
| 18 | Assigning a new doctor | Error as the program only selects the current user logged in as the new doctor for the patient. |
| 19 | Logs | Logs are all showing the same message despite being in different classes. |
| 20 | New Appointment | Issues creating a new row for the table. |
| 21 | Assigning a new doctor | For some reason the cell isnt replaced with the new doctor, it gets added onto the existing doctor. So there are two ids in the doctors field instead of just the new one. |
| 22 | New Appointment | Not sure how to add the ID to the table if It's going up by one. Solution may be to just get a "count" of the rows and then make that the new appointment id as it should start with an id of 0. |

Each bug that was encountered meant a new row was added with an id, the feature that the bug was found in as well as a brief description of the issue. This was done throughout each sprint, but team members had repeated issues and as the bug was already written somewhere it usually was not written again. Many bugs were also figured out as quickly as they were found and because of this were not written down inside the table.

## SPRINTS

| Sprint 1 | | | Sprint 2 | | | Sprint 3 | | |
|---|---|---|---|---|---|---|---|---|
| Bug ID | Fixer | Fixed? | Bug ID | Fixer | Fixed? | Bug ID | Fixer | Fixed? |
| 1 | HH | Yes | 16 | HH | Yes | 18 | HH | Yes |
| 2 | HH, WG | Yes | 10 | HH, WG | Yes | 19 | JW | Yes |
| 3 | HH | Yes | 9 | WG | Yes | 20 | WG | Yes |
| 4 | HH | Yes | 11 | WG, JW | Yes | 21 | HH | Yes |
| 6 | HH | Yes | 12 | WG | Yes | 22 | WG | Yes |
| 7 | JW | Yes | 13 | WG | Yes | | | |
| 5 | HH | Yes | 15 | HH | Yes | | | |
| | | | 14 | HH | Yes | | | |
| | | | 17 | WG, JW | Yes | | | |
| | | | 8 | HH | Yes | | | |

The sprints table has all the information held in what place. The bug that was encountered in what sprint, the “fixer” who found and most likely fixed the bug as well as clarification that the bug was in fact fixed at some point (mainly so someone does not believe the error still exists and tries to fix it).