CD5004 Advanced Programming – Emergency Call Centre

User Guide:

1. Execute the program by running the Main.java file
2. You will then be provided with an options menu of what can be done with the app

Text

Description automatically generated

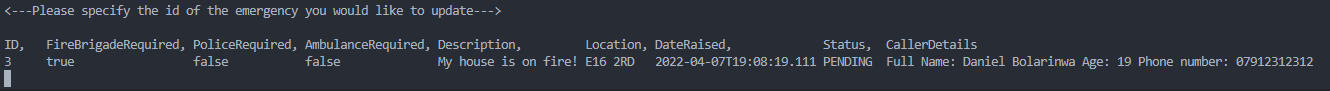
1. **Main menu option 1:** Record Emergency -> which will record the general information about the emergency. First will be emergency service that is required.

Text

Description automatically generated

Then you’ll be prompted to enter the details of the individual raising the emergency which will include full name, age, and phone number.  
Text

Description automatically generated  
Next you’ll be asked for details about the emergency you are raising including the description for the emergency and the location of the emergency by postcode only. Once this is done you are notified that your emergency has been recorded and you get redirected back to the main menu where you can make another choice.  
Text

Description automatically generated   
  
**Main menu option 2:** Update Emergency details -> which allows you to update the details of any existing emergency in the system. You are asked to input the ‘id’ of the emergency which you would like to update details for.  
  
Once you input the emergency’s id you are asked what attribute of the emergency you would like to update.   
Text

Description automatically generated  
**Update submenu option 1:** you are prompted to specify what extra services; you would like to add to the emergency because some emergencies might require more than one emergency service for example an incident of knife crime would need police for investigation and ambulance for the victim of the crime. In this case a user might have originally created an emergency for police then can update that the emergency will also require an ambulance. Once you’ve entered the service to add you’re redirected to the main menu again.  
Text

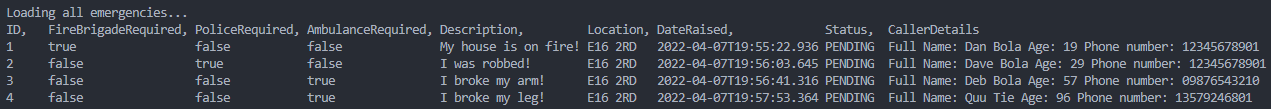
Description automatically generated  
**Update submenu option 2:** you are asked to enter a new description for the emergency. Once you entered the new description, you’re redirected to the main menu again.  
  
**Update submenu option 3:** you are asked to update the postcode location of the emergency. Once you entered the new postcode location, you’re redirected to the main menu again.  
  
**Update submenu option 4:** you are asked to update the information of the individual who raised the emergency. Once you have finished updating the individual’s details, you’re redirected to the main menu again.  
Text

Description automatically generated  
**Update submenu option 5:** you are prompted to input the status you would like to set the emergency to. Once you have stated the status to set the emergency to, you’re redirected to the main menu again.  
A screenshot of a computer

Description automatically generated with medium confidence  
**Update submenu option 6:** you are simply redirected back to the main menu  
Text

Description automatically generated  
**Main menu option 3:** Archive Resolved Emergencies -> which will remove any emergencies that have been marked as resolved from system. Once archiving is done, you’re presented with the main menu prompt again. You can verify that archiving completed successfully by generating a report (instructions to generate a report below) for all emergencies in the system and you should notice that none if any of the emergencies that are displayed have a status of ‘RESOLVED’.  
Text

Description automatically generated  
**Main menu option 4:** Generate reports -> which allows the user to generate reports for the emergencies in the system, you can filter by emergency service or status as well simply generating a report for all services.  
Text

Description automatically generated  
**Report submenu option 1:** this simply displays a list of all the current emergencies in the system  
  
**Report submenu option 2:** this allows you to filter the displayed emergencies by service  
Text

Description automatically generated  
The example below was filtered for ambulance emergencies only.  
  
**Report submenu option 3:** this allows you to filter the displayed emergencies by status  
Text

Description automatically generated  
The example below was filtered for ‘RESOLVED’ emergencies only.  
  
**Report submenu option 4:** you are simply redirected back to the main menu  
Text

Description automatically generated  
**Main menu option 5:** Save and Quit -> which allows the system to store state information about the data in the system so the user and system can retrieve them when they use the program again ensuring that emergencies don’t get lost in transit which could result in lack of trust in the emergency

Use Cases:

* You can record consumer emergencies for the police, ambulance, and fire brigade public emergency services.
* You can also update emergency details of emergencies that have already been recorded this allows you to update an emergency’s status one the necessary emergency service has resolved the issue as well as retrospectively change any other emergency data to avoid misinformation to the relevant authorities.
* You can also archive the emergencies which you have updated as resolved so you don’t keep around unnecessary data which in the future may slow down the system and in a fast-paced environment like an emergency call center you don’t want any delay in logging consumers emergencies.
* You can also generate reports on the emergencies that recorded in the system in real time. These reports can also be filtered by a specific emergency service for example you can generate a report to view all the emergency’s that require the police. This can also be done for the emergency’s status.

Problem Solving:

A problem I faced along the way was based around the requirement that an emergency should be able to require multiple emergency services. For example, a stabbing incident will need both an ambulance to attend to the victim as well as police to investigate the situation in the area. This was a challenge because my initial approach was to have a single required service field which could be set to any of the three as a string separated by spaces. This was very inefficient and complicated to manipulate that required services attribute and extract the individual services from the string.

I then tried to enumerate the required service attribute but also hit a dead end there because Enum variables in java can only take on one of the possible Enum values at a single time, so it wasn’t possible for me to have required service set to more than one Enum at a time.

The end solution to this problem was implementing 3 different attributes as part of the emergency corresponding to the 3 services. These 3 attributes would be stored as a Boolean data type where if a service is needed, we will simply set its Boolean attribute to true and if it is not needed it will be set to false. This way we can have and idempotent way of setting multiple required services for any given emergency.