Student Name: Calum Risk

Student ID: 1814778

Date: 29/05/2020

Abstract

Creating a Python application named ‘Shared Power’ to help tradesmen share expensive tools and specialist equipment.

**APP011-1: INTRODUCTION TO SOFTWARE ENGINEERING (DTS)**

**ASSIGNMENT 2: SHARED POWER**

**Contents**

[1. Introduction 2](#_Toc41626565)

[2. Task Description and Assumptions 2](#_Toc41626566)

[3. Implementation 2](#_Toc41626567)

[4. Challenges 4](#_Toc41626568)

[5. Testing and Scenarios 5](#_Toc41626569)

[6. Conclusion 5](#_Toc41626570)

[7. References 6](#_Toc41626571)

[8. Appendix 7](#_Toc41626572)

[Testing 7](#_Toc41626573)

[Run Application 7](#_Toc41626574)

[Login 7](#_Toc41626575)

[Create Account 8](#_Toc41626576)

[Tool Owner Functionality 9](#_Toc41626577)

[Tool User Functionality 14](#_Toc41626578)

[Dispatch Rider Functionality 19](#_Toc41626579)

[Insurance Company Functionality 22](#_Toc41626580)

[Database Tables 25](#_Toc41626581)

[Log File 25](#_Toc41626582)

[Use Case Diagram 26](#_Toc41626583)

# Introduction

This is an individual assignment to create a Python application named ‘Shared Power’ that helps tradesmen share expensive tools and specialist equipment.

# Task Description and Assumptions

The purpose of the program is to allow tradesmen to offer their tools for hire and hire other tradesmen’s tools when required. There are two main user types defined; registered owners that publish tools for hire and other registered users that hire tools. Other parties involved include dispatch riders for delivery of tools and an insurance company for damage investigations. Invoices are produced monthly for all. These include hire charges, delivery costs, late fees and a flat charge of £5 per bill.

Based on the task description, I have made the following assumptions:

* There are four types of users:
  + Tool Owner
  + Tool User
  + Dispatch Rider
  + Insurance Company
* Tool Owner, Tool User and Dispatch Rider accounts can be created by anyone. Insurance Company accounts can only be created by system administrators. System administrators do not need an account and can make changes by editing the Python code and/or by updating the data contained within the database.
* Tool User users can see all the dates a tool has been booked for before making a booking. It is not limited by only 6 weeks in advance.

# Implementation

I used a number of sources to help assist me with the creation of my code. These include:

* Studyfied for Tkinter option menu creation. [8]
* Codemy.com Tkinter YouTube series. [2]
* TheNewBoston Python GUI with Tkinter YouTube series. [10]
* Soumilshah1995 - How to read and write Image to a database using Python SQLITE YouTube video. [6]
* Sunbearc22’s GitHub code - A ttk frame allowing vertical scrolling only. [9]
* Giacomo Debidda - MVC pattern in Python: SQLite. [4]

Before enrolling on this module, I had no previous experience with the Python programming language and limited experience with other programming languages. In order to start coding using Python, I was advised to install Conda. Conda is an open source package management system and environment management system for Python and other programming languages. [3] A Python script uses a Conda environment to run with environment variable and packages installed for the environment. Along with Conda, I installed the Anaconda Navigator, which shows a number of applications that can be used with Conda. Having tried some of these applications, the most suitable application for me to code this program was the IDE (Integrated Development Environment) PyCharm. As an IDE, PyCharm can be used to create Python scripts. I was made aware of PyCharm by a number of sources online and chose it as my IDE because of its available features that aid with coding. For instance, the software continuously analyses the code being created to suggest best practices and possible mistakes.

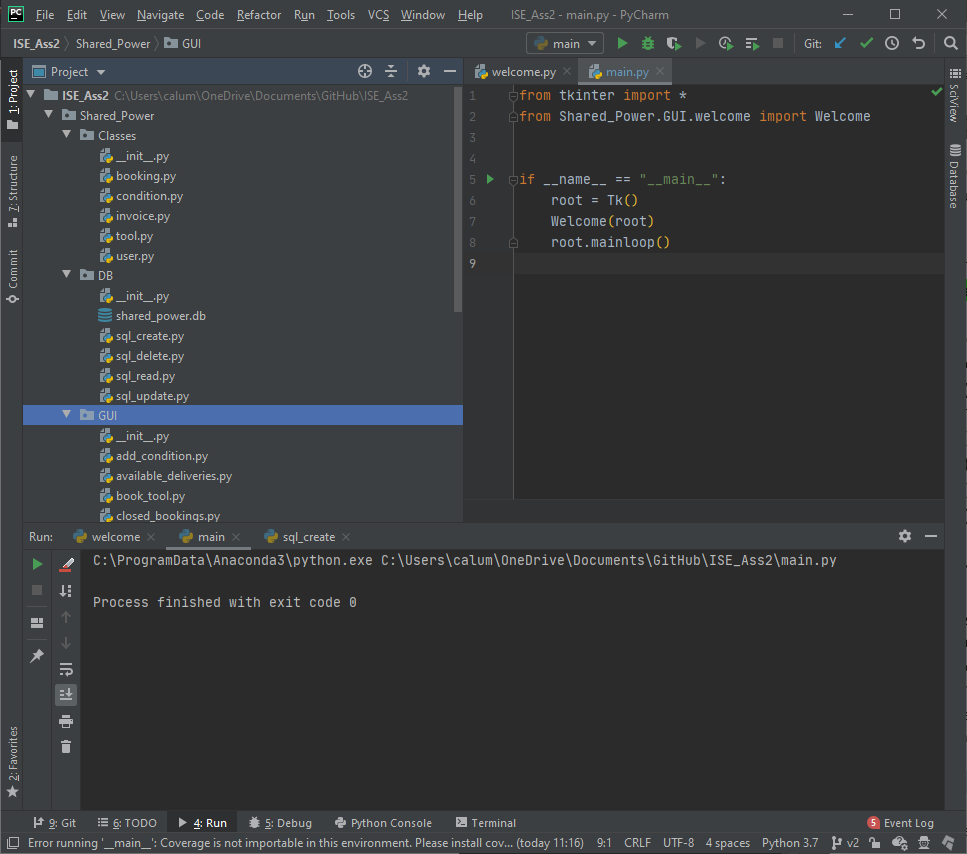


Figure : PyCharm

Due to the requirements of the program, I felt a GUI (Graphical User Interface) would significantly improve the user experience. The Tkinter package allows for the creation of a cross-platform GUI, meaning it works on Windows, macOS and Linux without any code modifications. [1] Based on my skillset, the support available online and the cross-platform functionality, I chose to use Tkinter to create a GUI.

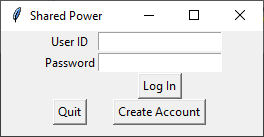


Figure : Tkinter GUI Window

I felt the only suitable solution to store the data for the ‘Shared Power’ program was a database. This is because there is a vast amount of data collected, which needs to be accessible in all areas of the program. SQLite3, or SQLite for short, is a fast, lightweight database engine for Python. [5] With the use of SQLite, I organised the SQL modules according to CRUD principles (Create, Read, Update, Delete). This makes the respective classes simple to call throughout the code. Figure 3 shows how the SQL modules have been organised.

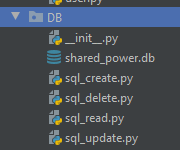


Figure : SQL Modules

For my code, I used an object-oriented programming (OOP) approach by organised all code into classes. This allowed the relevant parts of code to be called when required. It also enable code to be reused by passing different values into classes depending on the scenario. I feel my code reflects some design patterns associated with OOP. For instance, the ‘Pool’ package contains classes that are called upon to temporarily store objects that then get passed to the relevant SQL class to create, read, update or delete data. This methodology is referred to as an object pool and is a type of creational pattern. The object is no longer required once the SQL functionality is performed. As well as this, my code reflects the mediator behavioural pattern where each user type view is the mediator. For example, a Tool Owner is able to call other classes once they have logged in. Their home page stays the same, acting as a central base. Other GUI classes open in a new window without affecting the home page. [7]

# Challenges

During the time of creating my program, my skills were significantly improving. As a result, this led to regular improvements, which involved a lot of re-coding.

A more specific issue I had was writing image files to the database and retrieving them to display in the GUI. To store the file in the database, it must be read as a binary file. Likewise, it is written as a binary file when retrieved, which enables the file to be formatted correctly for opening again. Evidence of uploading image files can be found in the “create\_tool.py” and “add\_condition.py” modules. Evidence of retrieving the image and displaying this on the GUI can be found in “manage\_booking.py” and “manage\_case.py” modules.

# Testing and Scenarios

Testing and scenarios can be found in the appendix section. Overall, the program runs smoothly and as intended. I have included some error handling, however due to time constraints have not managed to implement completely throughout the application. The errors that are handled are written to a log file with a timestamp. This can be found Logs package.

# Conclusion

Overall, the program was a success. There are a few issues whereby if the correct data is not supplied errors will occur. However, this does this does not make the program unusable. The GUI provides a significant enhanced experience over a console-based application. There is logic to the program structure with the use of classes.

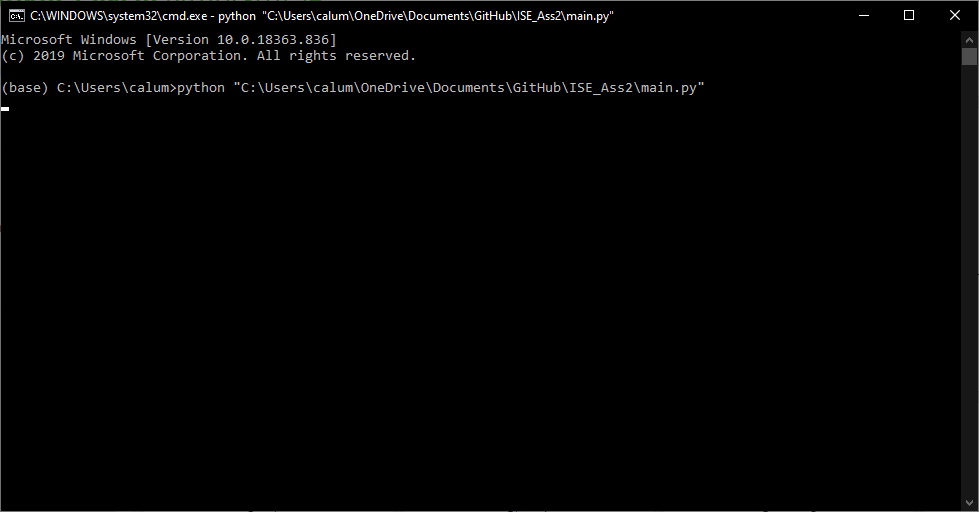
# References

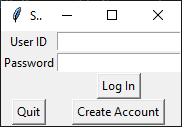
1. Amos, D. 2020. Python GUI Programming With Tkinter. [Online]. [29 May 2020]. Available from: <https://realpython.com/python-gui-tkinter/>
2. Codemy.com. 2019. Python GUI's With TKinter. [Online]. [29 May 2020]. Available from: <https://www.youtube.com/playlist?list=PLCC34OHNcOtoC6GglhF3ncJ5rLwQrLGnV>
3. Conda. c2017. Conda. [Online]. [29 May 2020]. Available from: <https://conda.io/en/latest/>
4. Debidda, G. 2017. MVC pattern in Python: SQLite. [Online]. [29 May 2020]. Available from: <https://www.giacomodebidda.com/mvc-pattern-in-python-sqlite/>
5. Python Central. 2013. Introduction to SQLite in Python. [Online]. [29 May 2020]. Available from: <https://www.pythoncentral.io/introduction-to-sqlite-in-python/>
6. Soumilshah1995. 2019. How to read and write Image to a database using Python SQLITE. [Online]. [29 May 2020]. Available from: <https://www.youtube.com/watch?v=VvM-uAp9zW8&list=LLhZb76mKnBlZ9uZzfoRyi5A&index=14&t=638s>
7. Source Making. c2020. Design Patterns. [Online]. [29 May 2020]. Available from: <https://sourcemaking.com/design_patterns>
8. Studyfied. 2019. Tkinter optionmenu widget. [Online]. [29 May 2020]. Available from: <https://studyfied.com/tutorial/tkinter/optionmenu-widget/>
9. Sunbearc22. 2018. tkinterWidgets. [Online]. [29 May 2020]. Available from: <https://github.com/sunbearc22/tkinterWidgets/blob/master/scrframe.py>
10. TheNewBoston. 2019. Python GUI with Tkinter Playlist. [Online]. [29 May 2020]. Available from: <https://www.youtube.com/playlist?list=PL6gx4Cwl9DGBwibXFtPtflztSNPGuIB_d>

# Appendix

## Testing

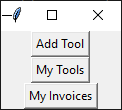
### Run Application



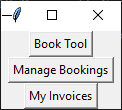


### Login

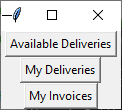
* Tool Owner



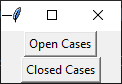
* Tool User



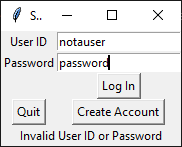
* Dispatch Rider



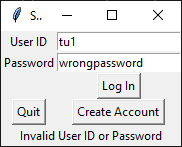
* Insurance Company



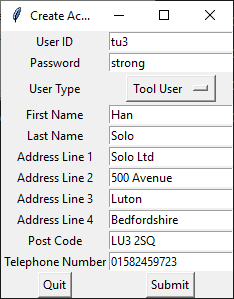
* Incorrect Username



* Incorrect Password



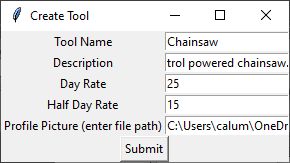
### Create Account



*Window closes after selecting submit*

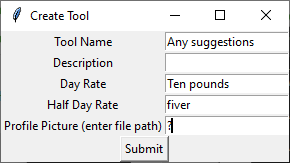
### Tool Owner Functionality

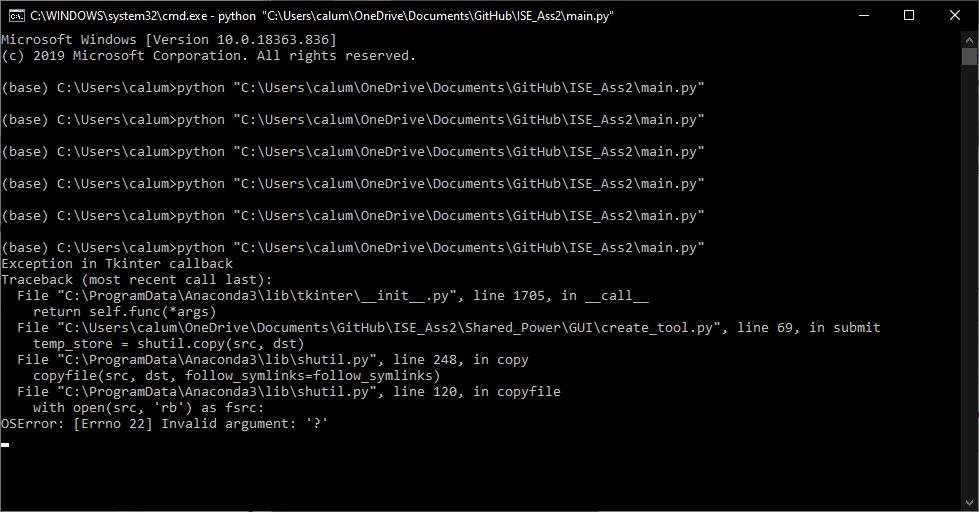
* Add Tool (valid)



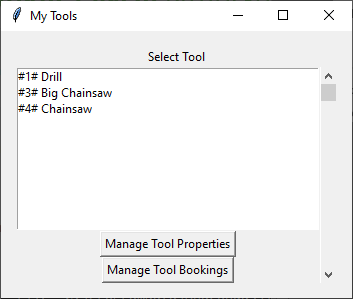
*Window closes once submit is selected.*

* Add Tool (invalid)

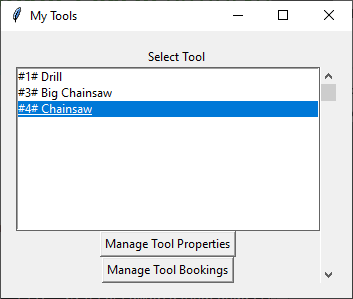




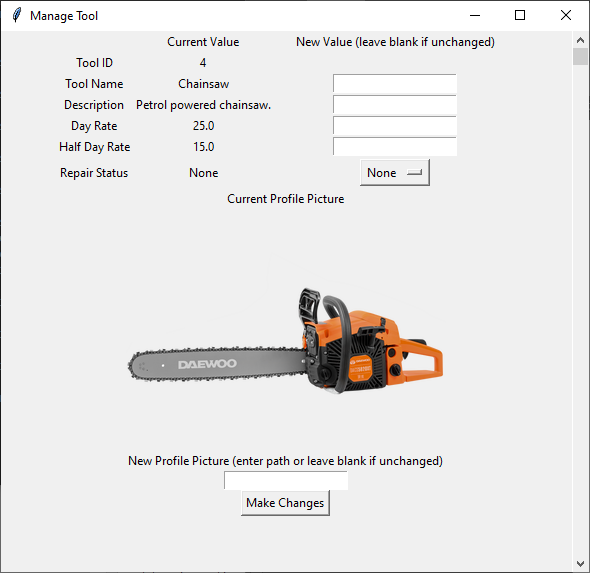
* My Tools



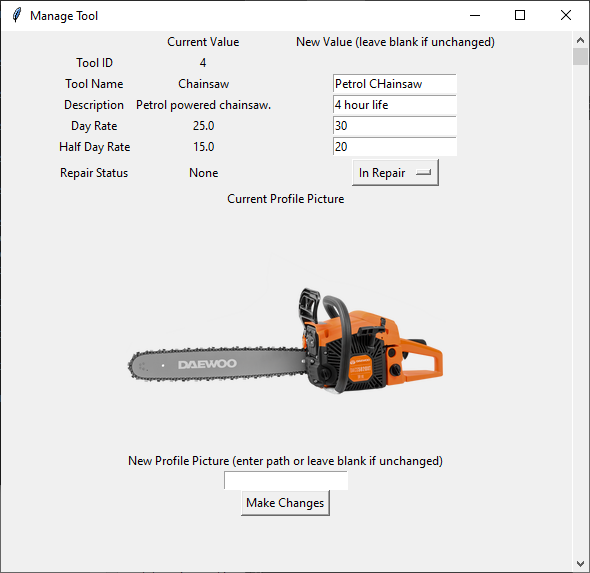
* Manage Tool Properties



*Select Manage Tool Properties*

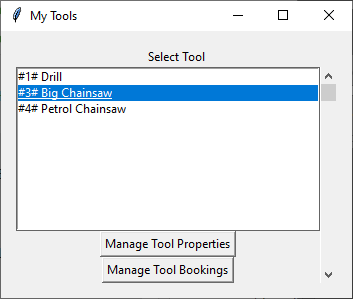


* Update Tool Details

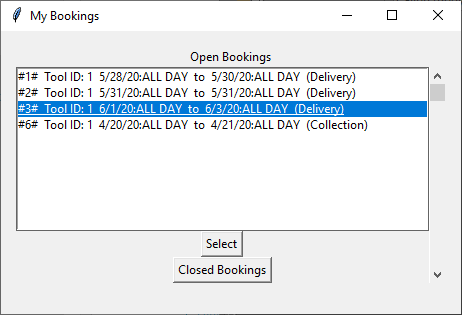


*Window closes after Make Changes is selected.*

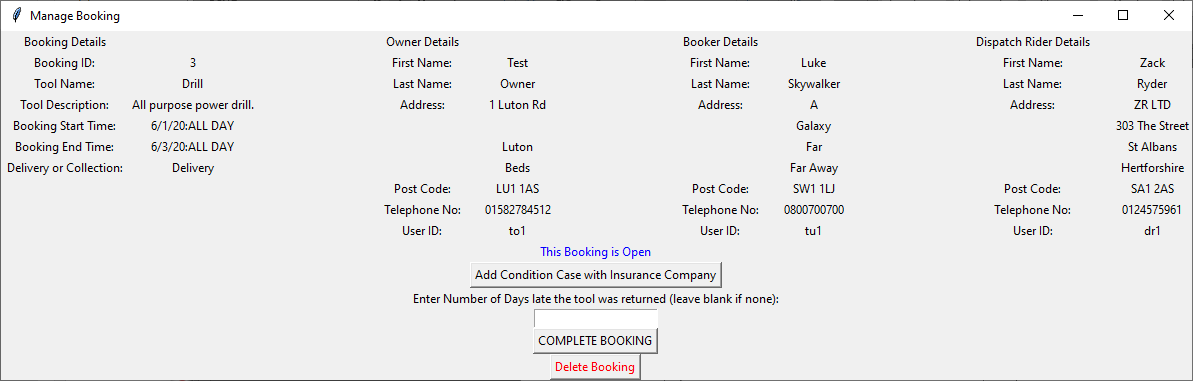
* Manage Tool Bookings



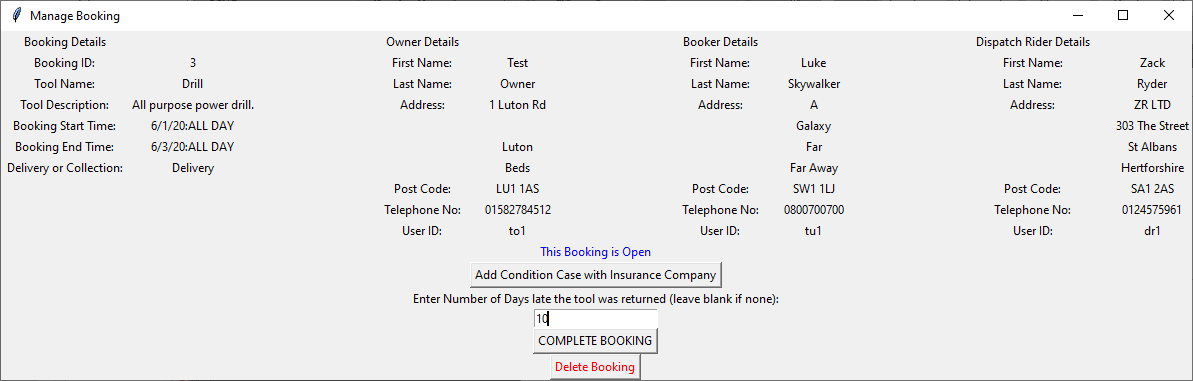
*Select Manage Tool Bookings.*



*Select booking.*

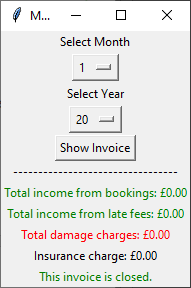


* Complete Booking



*Entered days late and Close Booking button. Window closes.*

* My Invoices
  + No bookings / no open bookings

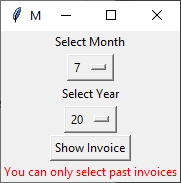


*No charge insurance charge if there are no bookings.*

* + Bookings / open bookings

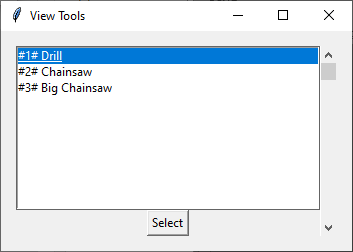


* + Future date invoice

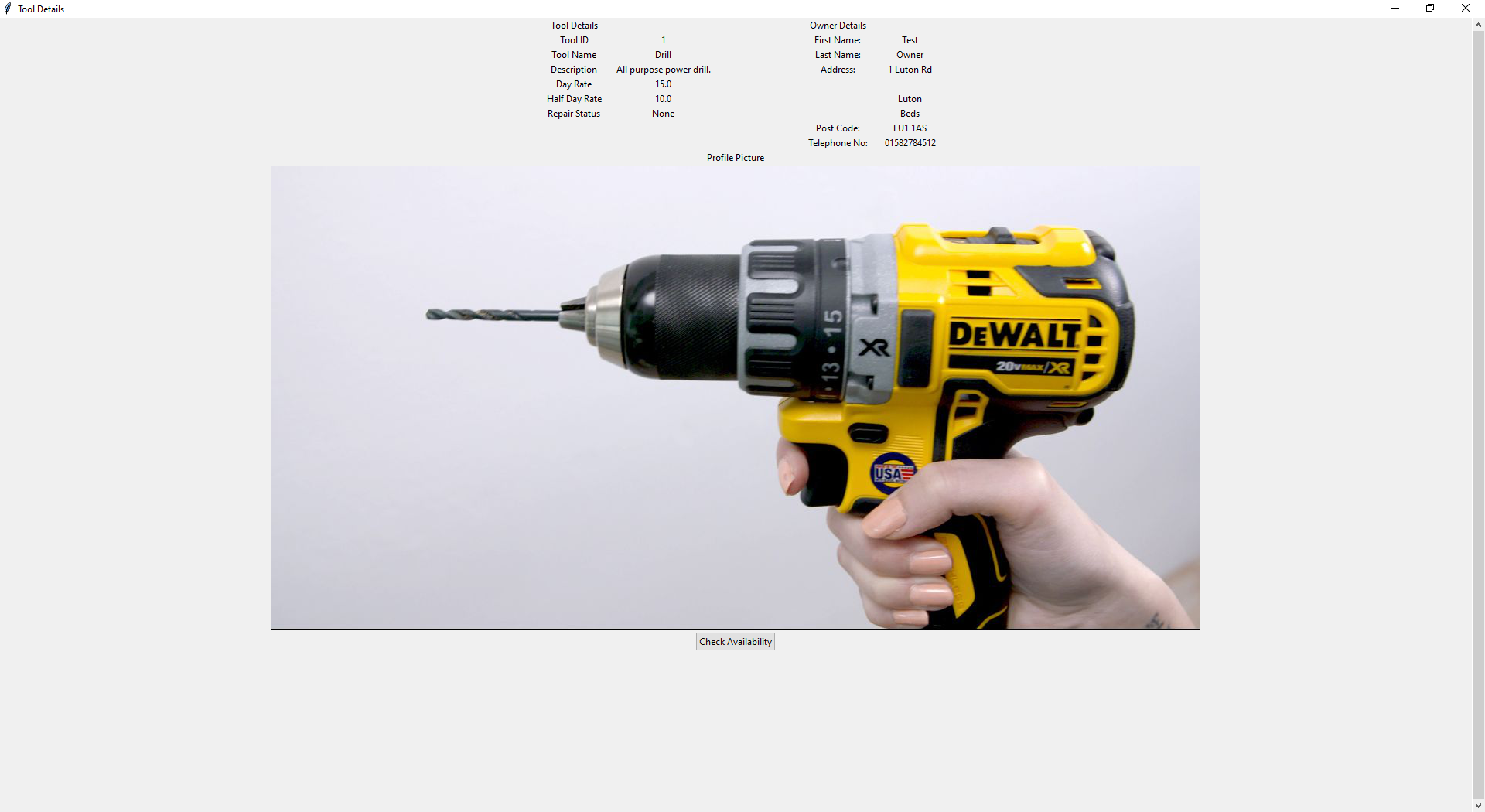


### Tool User Functionality

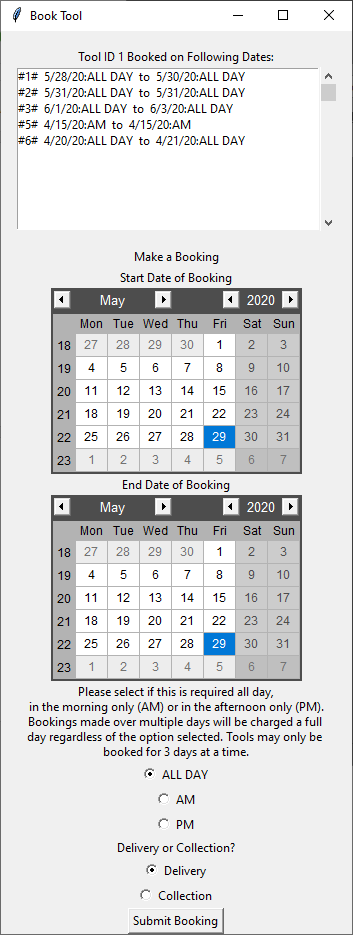
* Book Tool



*Petrol chainsaw not in list because it is in repair.*



*Select Check Availability button.*

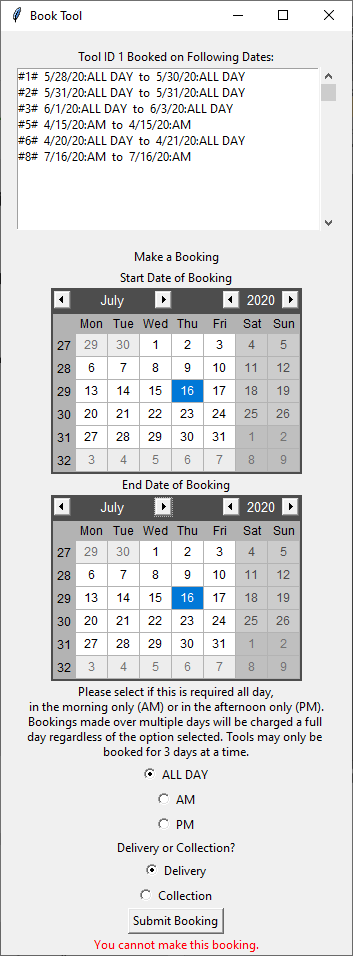


*Shows upcoming bookings and provides options to book tool.*

* Submit Booking – valid

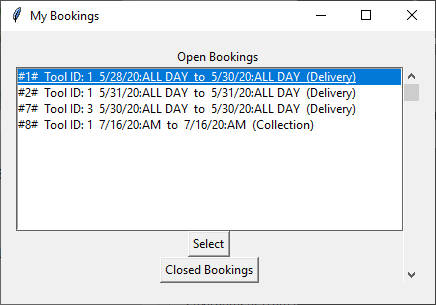
*Submits without error.*

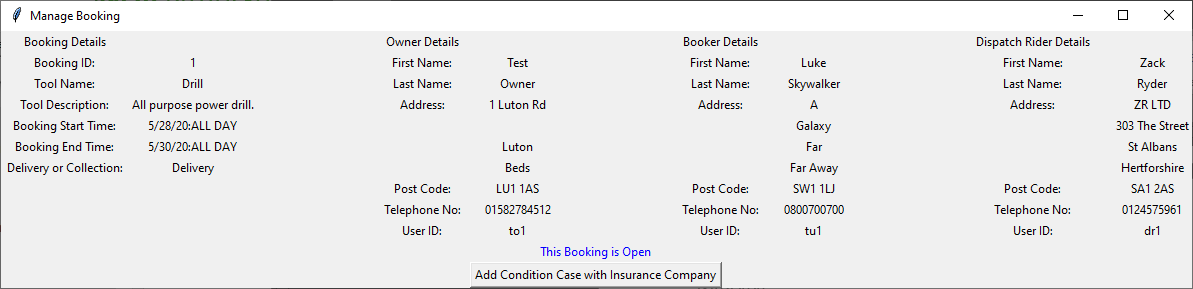
* Submit Booking – invalid



*This message occurs for invalid bookings.*

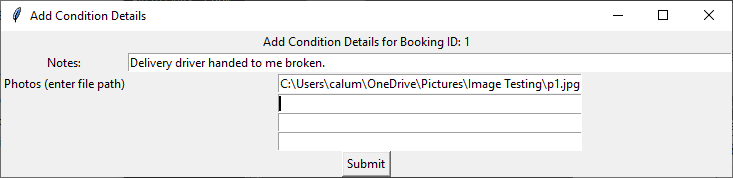
* Manage Bookings





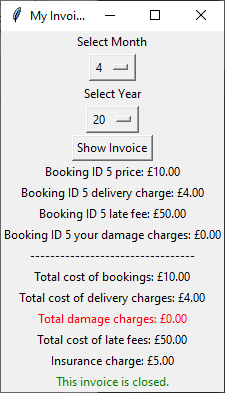
*Unable to delete or complete the booking as Tool User.*

* Add Condition Case with Insurance Company



*Only one of the four file paths used and was able to submit.*

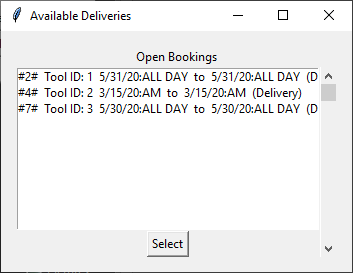
* My Invoices
  + Closed



*Includes delivery charges.*

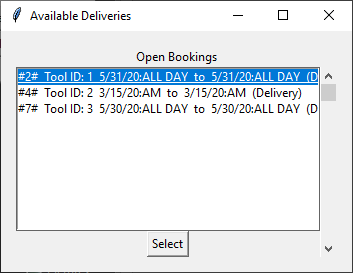
### Dispatch Rider Functionality

* Available Deliveries



*Lists all unassigned deliveries.*

* Assign Delivery



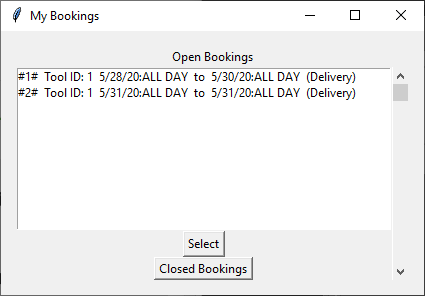
*Shows unassigned deliveries.*



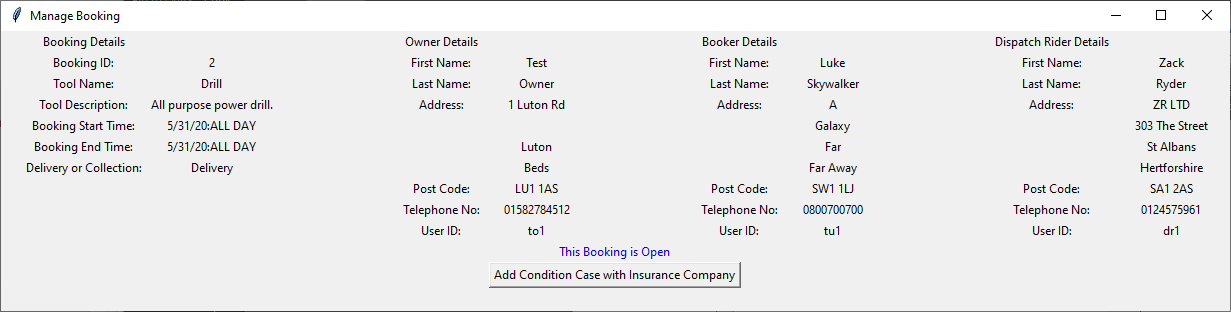
*Take Delivery button is available for Dispatch Rider users.*

*Window closes after taking delivery.*

* My Deliveries

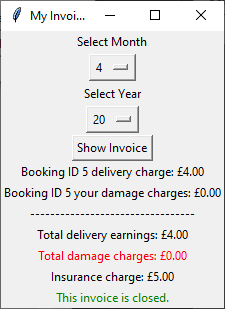


*Deliveries assigned can be viewed.*



*Take Delivery option no longer available once it has been assigned.*

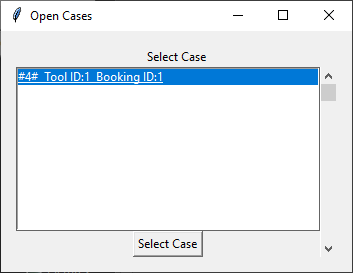
* My Invoices



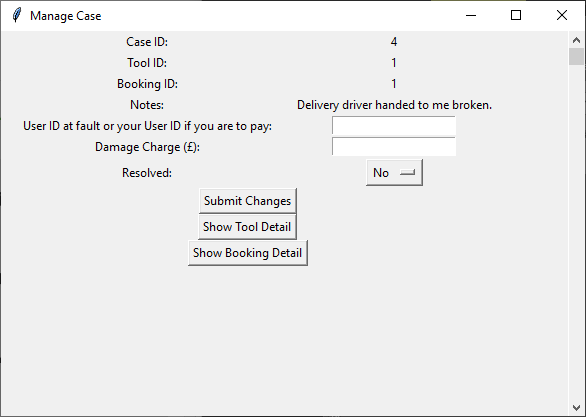
*Shows relavant details for user.*

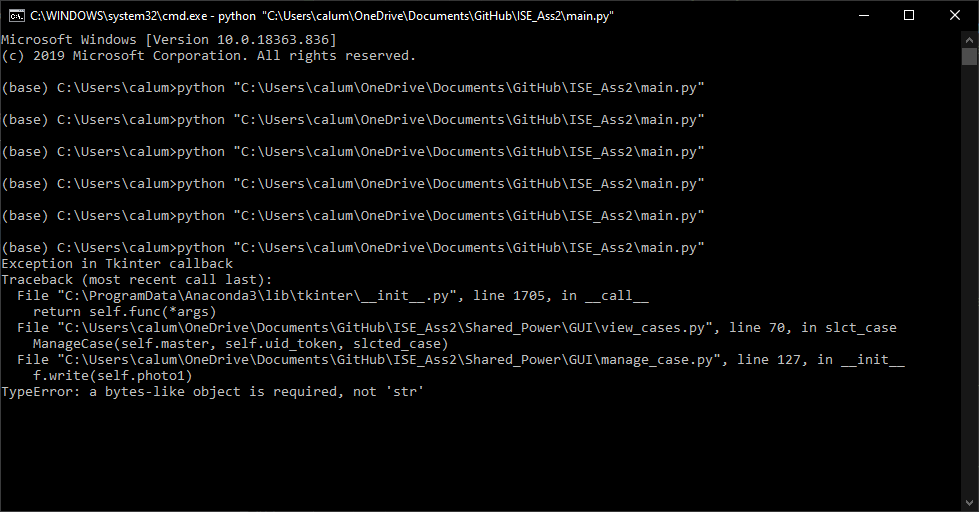
### Insurance Company Functionality

* Open Cases



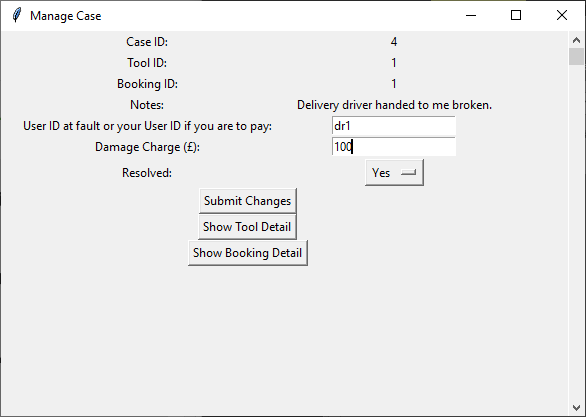
*Shows condition cases logged that are open.*





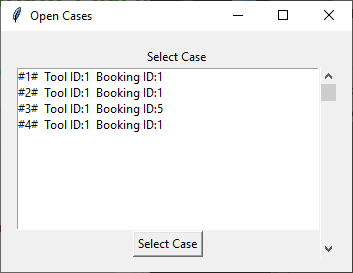
*Seemingly error loading images. Most likely because 4x images were not uploaded when case was logged.*

* Submit Changes / Resolve Case



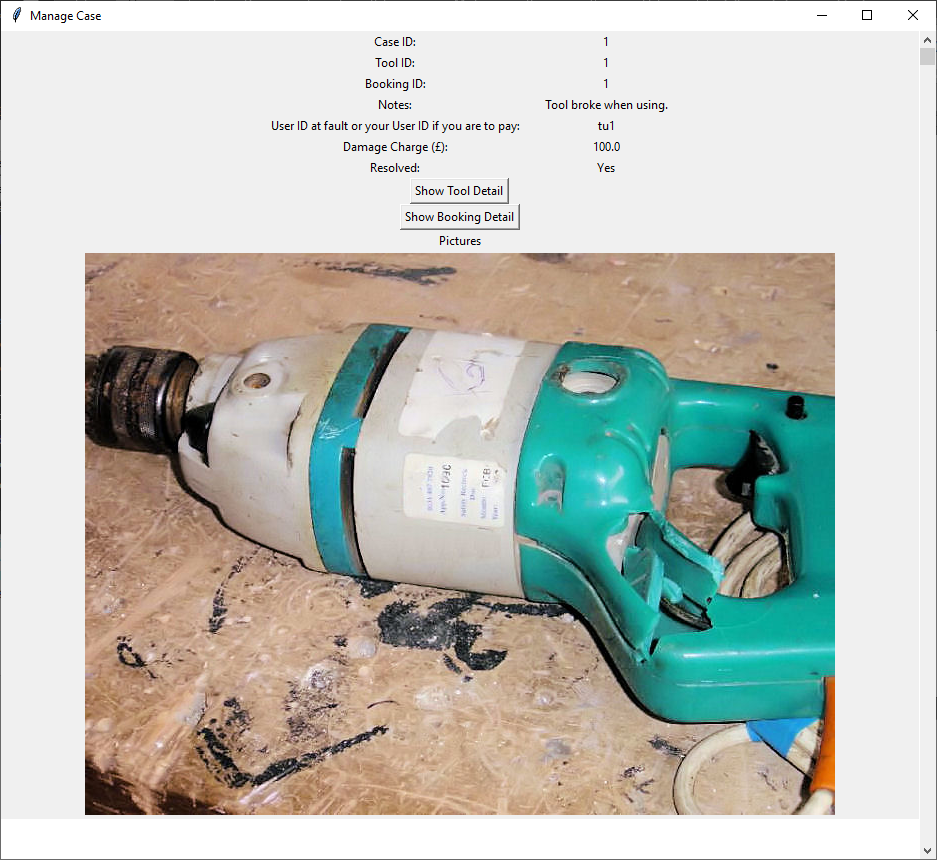
*Window closes after submitted.*

* Closed Cases



*Shows a list of closed cases.*

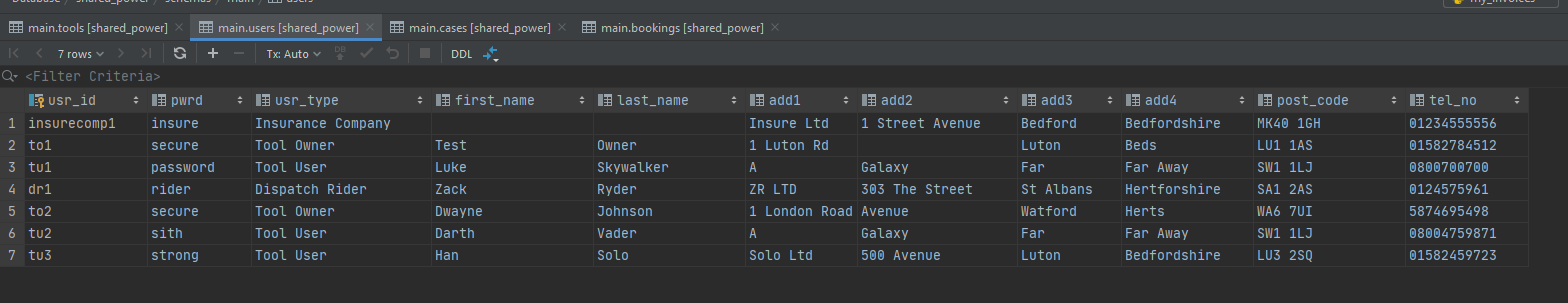
* Open Closed Case

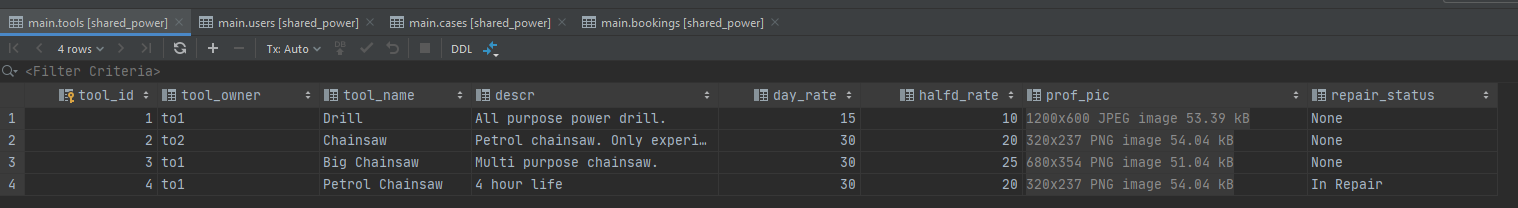


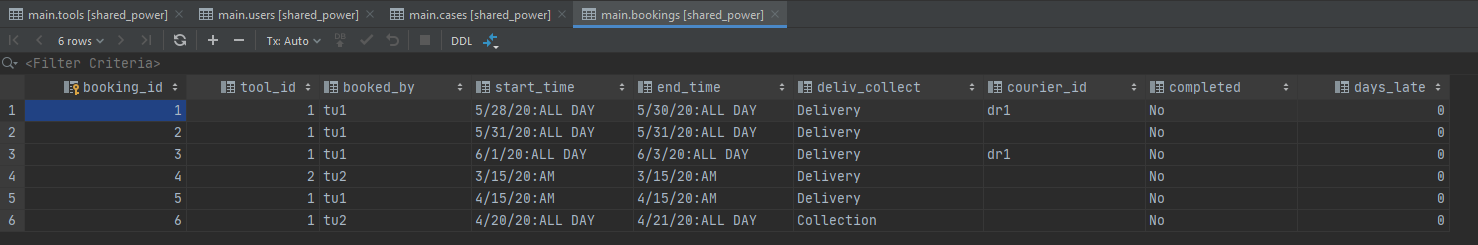
*The pictures load correctly. Scrollbar works. No option to make changes.*

### Database Tables

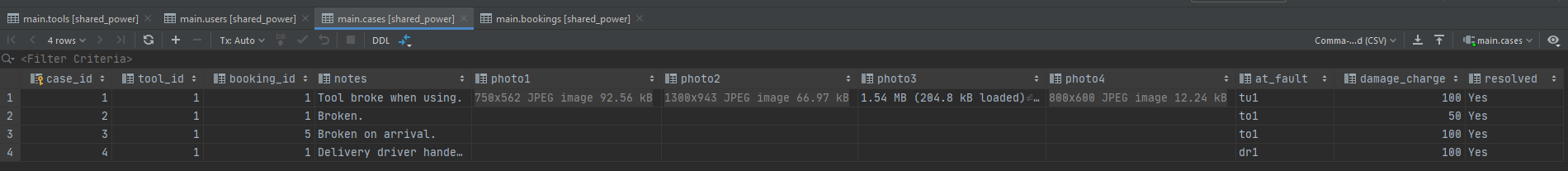
* Users



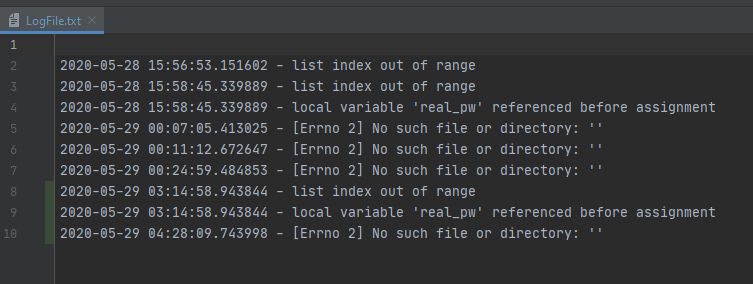
* Tools
* Bookings



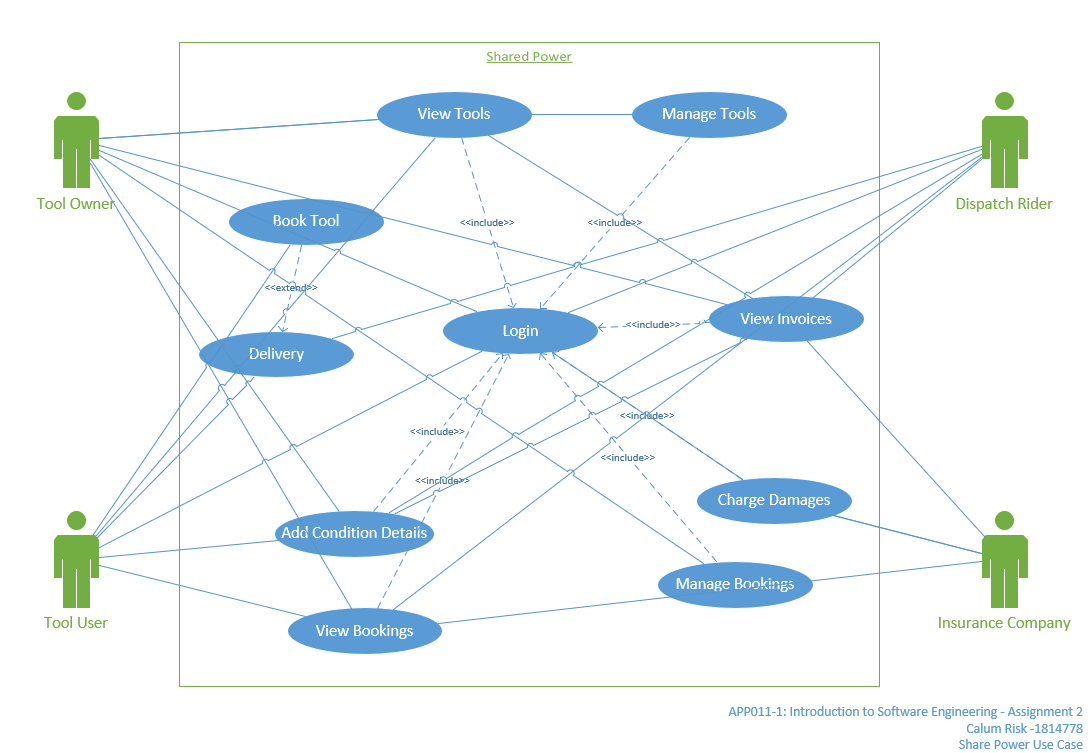
* Cases



### Log File



## Use Case Diagram



*Also in project directory as PDF and Visio file.*