SOFTWARE ENGINEERING PROJECT

INDIVIDUAL REPORT

HUSSEIN AHMED TEJAN

STUDENT NO: 130108432

GROUP: 3C

GROUP MEMBERS:

Jaseera Parvin Mohamed Abubacker

Nathan Winslow

Sanjay Manikandhan

Rebecca Jane-Bell

Contents Page

Week1

Ice-Breaker Meetup

- Corresponded with my group members to appoint a suitable time that would be possible for us to get together and plan before hand.
- I met with my group members before my lecture and discussed with them how we planned to tackle the project.
- Decided on who would take project leader.

Week2

Plan

- In our second meeting, we set up SVN to prepare in advance for the major platform of the project; furthermore we discussed producing UML diagrams such as Entity Relationship Diagrams, Use Case Diagrams with the intent of understanding the design and requirements of the system we intended to build.
- We observed how SVN works through testing it with files and folders.
- We gave ourselves a deadline to ensure that the UML diagrams were done by week 3.

Week3

- Completed my entity relationship in addition to my class diagram and joined it with the other modules
- Single-handedly completed the sequence diagram by myself

Week4

- Started sketching my module in terms of how the GUI should look
- Commenced on the database filling in the tables with the primary and foreign keys

Week5

- I discussed GUI plans whether I intended to use GUI Builder or code up my module
- I did research into GUI layout managers, frames, buttons, panels via multiple methods

Week6

- I sought advice from some TAs with how to implement certain ideas I had thought of producing, but was unsure as where to start.

- I re-discussed with my group as how to make the most benefit of the GUI features within the IDE: Netbeans.

Week7

- Made changes to the Database

Week8

- Made designs of what to include in my system

Week9

- Altered certain designs found in my system that were originally there

Week10

- I started to structure my back end code more to ensure that everything was in place to enable my GUI to work effectively with objects of the back end code.
- I commenced in fusing the back end classes with my GUI classes, instantiating objects and calling methods from the back end code into the GUI.

Week11

- I made some hand screens to help me figure out how to make a realistic Scheduled Maintenance Booking system that would match the specification criteria from top to bottom.
- I finalised how I would set up all my classes and frames, as I had to change certain designs of which I had initially planned to implement.
- I started thinking slightly more architectural and implemented into my code the changes I designed in my screens.
- I altered certain features of my GUI classes to make the texture more appealing and attractive to the eye.

Week12

- I aligned my code to ensure that it was presented in a very neat and tidy depicter, ensuring that methods, variables were inline in addition to modifiers throughout all my classes.

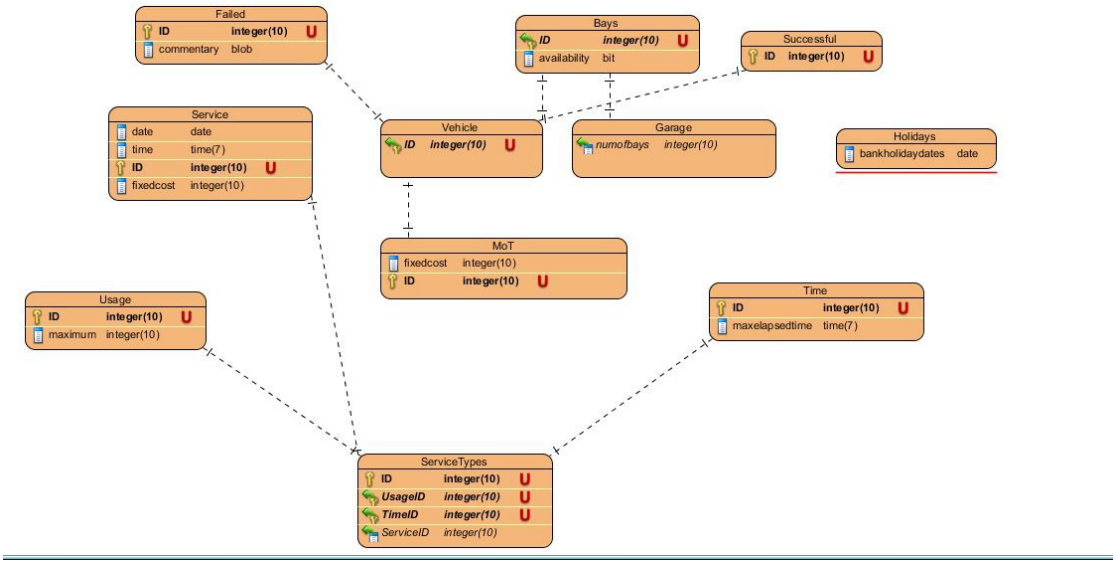
- I did testing on all my classes to check that everything worked fine together without any problems, as well as tested it with my counter modules of which myself and my group members would integrate everything all together.

Scheduled Maintenance Booking

I built a Scheduled Maintenance Booking system that would enable customers to service their vehicle and take an MoT test. The mechanic would be in charge of blocking dates customers could book an appointment with and permitting them to make bookings in a single bay.

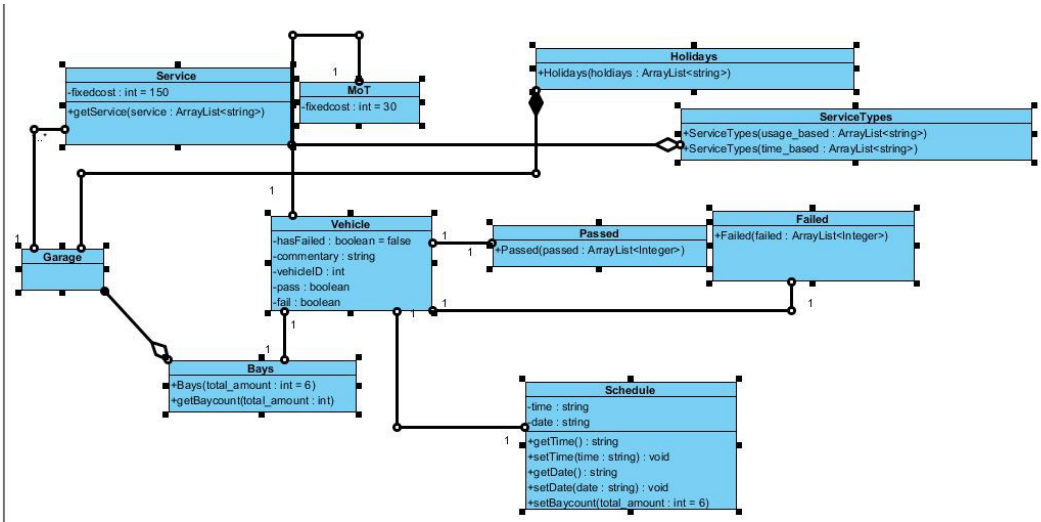
Diagrams

Entity Relationship Diagram (Scheduled Maintenance Bookings)



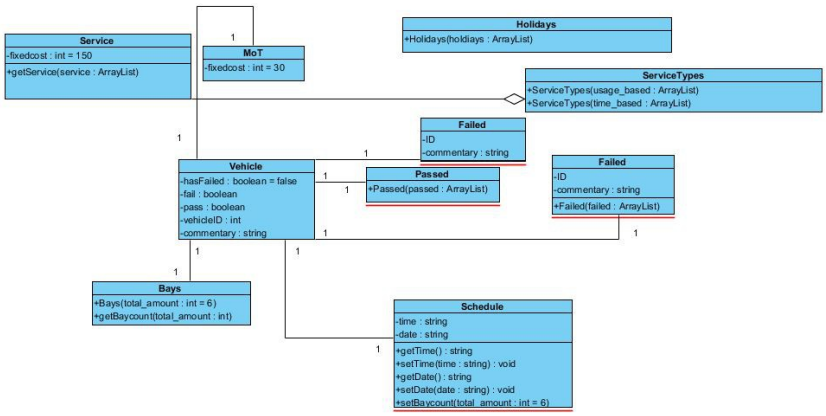
- I had designed this whole ERD based on my module all by myself.

Original Class Diagram (Scheduled Maintenance Bookings)

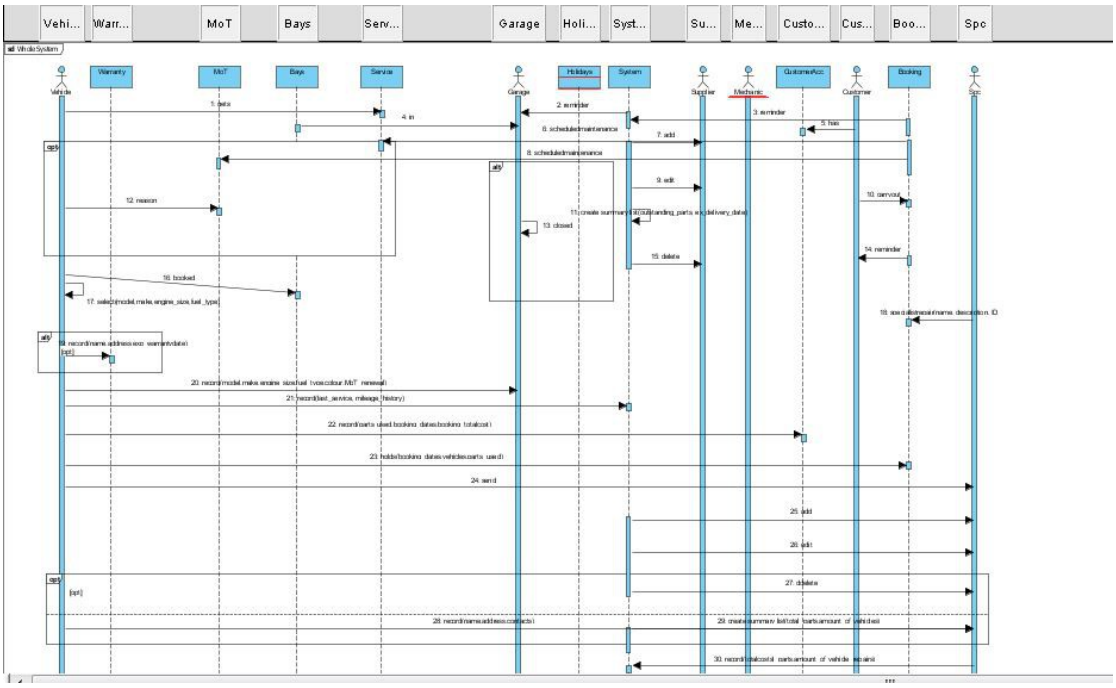


- I had designed this original Class Diagram based on my module all by myself in addition to the refined version featured below which was incorporated and integrated with the overall Class Diagram of the system.

Final Class Diagram (Scheduled Maintenance Bookings)



Sequence Diagram



Individual Ambiguities

1. “A vehicle can fail an MOT test and this must be recorded”, this statement is not precise in terms of where exactly the failed vehicle's reasons for failing would be stored as in which specific system.

2. “The garage is not open on public and bank holidays”, this statement is unclear as one does not take account of the months in the system in any section, thus it is not really clear how one would know anyway is any sort of holiday, I.e Christmas Day (25th December).

3. It was not very clear if the mechanic would have the option of adding holidays or whether the holidays were already written in stone suited with real life public and bank holidays.

4. It was unclear for me whether it was compulsory for me to enable the mechanic to permit the customer to request to book either a Time Based Service or Usage Based Service based upon “A service is carried out according to the vehicle manufacturer's schedule, which is either usage or time based. Hence, the 'or' makes it very meticulous for the mechanic to only enable all customers to do one type of service or to enable all customers the option of choosing which service they would like to book an appointment for.

5. It was very difficult for me to consider the scenario as follows “A usage based schedule stipulates the maximum number of miles the vehicle can be used between services, typically it is between 10000 and 12000 miles”, since in real life an individual would only need to renew the servicing of their vehicle if they had exceeded 10000 miles (even before the expiry date of their service) and or after the expiry date had passed since they last serviced their vehicle.

6. It was very difficult for me to consider the scenario as follows “A time based schedule stipulates the maximum elapsed calendar time between services, such as a year or two years.”, taking into account the above in real life an individual possessing a vehicle would have to renew their service a year from their last service date or else they would be jeopardising others lives on the road in addition to their own life.

7. It was quite ambiguous as to what the fixed time duration would be for an MoT, Usage Based Service and a Time Based Service as it was not stated in the specification, although it was noted that the Garage system was open from 9:00 am – 5:30 pm on weekdays and 9 am – 12 noon on Saturdays, furthermore one would have to do general research or alternatively have a connection with a real life Garage to know how exactly the system would work prior to commencing such a project.

8. It was ambiguous as to whether the bays had to be pre-allocated to a certain type of Booking or whether they needed to be generally available for any type of booking and unavailable if taken by another customer based upon “There are six bays in total, each of which can be occupied by only one vehicle….. Bookings taken must respect the availability of bays”.

9. Looking back, it was quite confusing for me, as to whether my module would require the current mileage of a customer's vehicle as the last point in the specification stated: “Current mileage of the vehicle must be recorded during the diagnosis and repair booking”.

10. It was very ambiguous as to how short or long the wait would be in order for customers to book an appointment in the future based upon “Appointments can only be made in the future”.

11. It was also ambiguous baring in mind that the opening hours of the Garage from Monday to Friday was from 9:00 am – 5:30 pm as I stated above, but in this case this is because it is not very clear what the extra half an hour would be necessary for; henceforth it breaks the booking time pattern which is every hour rather than every half an hour. It leaves one wondering what exactly would be the point of the Garage closing at 5:30 pm and if that is the case, what about Saturdays would the Garage run from 9:00 am to 11:00 am rather than 12:00 noon.

<u>Group Ambiguities</u>

- SVN would be very inactive and thus I was unable to have access to SVN most times and to make matter worst it would only work temporarily on my laptop.

- Lab Lockdown would interrupt my ability to work on the project on the ITL machines, this forced me to make use of my laptop.

- Due to my group not implementing their GUI as I did, we were unable to produce a single jar file that consisted of one central Garage Booking System as they had produced main methods in each of their classes, whereas I had a whole class consisting of a single main method and this had made it extremely difficult to merge and integrate each of our systems to form a whole system.

<u>Integration and Team Work and Project Experience</u>

In my opinion, I found the project a great means to experience how a working environment would realistically exist as I was put into a group of members whom I am unfamiliar with and we had a single mission to produce a single system with integration and this opened my eyes as to what to expect in such an environment.

However, with that being said I was not pleased with the scenario of the project as it was quite boring, while on the other hand I can understand the reason behind why such a scenario was presented, due to the fact that one would have to do certain things in a workplace which he/she is instructed to fulfil with deadlines and in such cases it is not necessarily what an individual would be interested in doing.

I have learned how to respect other peoples views and opinions expressed in a team and in doing so I have managed to put forward my views without force, but rather with reasoning and justifications as to why I believe such decisions would be more suitable.

I had been left alone to do the Bookings by myself without the person responsible for my counter module to assist me and I strongly believe this was extremely unfair on me as it should have been both of us working together to produce a Booking system that would be beneficial for Diagnosis and Repair Booking and Scheduled Maintenance Booking. I had voiced my concerns to my group leader and the individual, but to be fair I should have spoken up much louder to make sure things went the way I expected. Also, I had dependencies with the database that had been built by my group leader, henceforth I had to wait on my group leader in order for everything to be put into perspective in regards to my module, eventually I put my foot down and made changes to the database to ultimately ensure that my module was in place to build the GUI along with the database. Moreover, I had to agree to my group leader suggesting that we submit my system (Scheduled Maintenance Booking) as a separate jar file in a folder along with my group's jar file and this was difficult to swallow as I had attempted countless times to integrate our system as a whole for many hours.

If I were to repeat the project, I would still keep my group leader the same, but I would be more stricter with him in regards to the database and what I required, in addition to I would also be much stricter with my partner who was doing Diagnosis and Repair Booking and would therefore motivate them more to do the work.

Overall, I feel that we all coexisted as a team together without too many problems, but with despite that I believe things could have gone much differently had we all discussed our concerns and met up more regularly.

References

Please find below enclosed the 3rd Party libraries that I had used within my module.

- Jcalendar → JDateChooser, jgoodies-common, jgoodies-looks
- rs2xml (A library pre-requisite to enable the database to be filled in through the GUI)
- sqlite-jdbc (A JDBC driver to enable me to connect my classes to the Database through an established connection)
- junit-4.6 (A simple unit testing framework for Java and it was required to help write repeatable tests for my project class files)

Source:

toedter.com/jcalendar/

What is JCalendar?

“Introduction

JCalendar is a Java date chooser bean for graphically picking a date. JCalendar is composed of several other Java beans, a JDayChooser, a JMonthChooser and a JYearChooser. All these beans have a locale property, provide several icons (Color 16×16, Color 32×32, Mono 16×16 and Mono 32×32) and their own locale property editor. So they can easily be used in GUI builders. Also part of the package is a JDateChooser, a bean composed of an IDateEditor (for direct date editing) and a button for opening a JCalendar for selecting the date.”

“License

This program is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation. If you like and use it, just let me know. If you find any errors or things you don’t like, please contact Kai Toedter.**”**

Why I needed JCalendar?

I got kind of sick of the old Calendar object and wanted to be unique and decided to integrate a 3^{d} Party Library into my module, so that I would be able to physically display a Calendar to the user who would attempt to book an appointment and in doing so prompt the user if they could or could not book an appointment to come to the Garage, thus I had a vision to block previous passed dates, bank and public holidays, Sundays as well as alter the timings that the user would be able to book an appointment on Saturday. Furthermore, I would have used the same Library again if I was to do the module once again as I strongly believe that it helped direct me to introduce my vision into reality baring in mind the capabilities it brings with it such as the following methods: getDate(), getDay.

List of **Classes**, *Methods* and ***Attributes*** from Visual Paradigm (Class Diagram)

Service	**MoT**	**Holidays**	**Garage**	**Vehicle**	**Passed**	**Failed**	**Schedule**	**ServiceTypes**	**Bays**
This class was created to be a superclass of the types of services i.e Usage Based and Time Based. This would require the customer to pay the cost of services.	**This class was created to enable the customer to pay the cost of an MoT test as well as help the mechanic grade the customer.**	**This class was created to be a superclass of the types of holidays such as Bank holidays and Public holidays.**	**This class was created initially to disable customers from booking any appointment on blocked dates.**	**This class was created**	**This class was created to keep track of the vehicles that had passed the MoT test.**	**This class was created to keep track of the vehicles that had failed the MoT test and the reasons behind why the vehicles had failed.**	**This class was created to handle the time and date of bookings that were scheduled and interacted with the calendar I produced only**	**This class was created initially to store the types of Services available.**	**This class was created to keep count of how many bays are available and were originally available, thus only enabling one**

							allowing the mechanic to book customers days the Garage was open.		**vehicle at a time to be booked in a bay.**
getService()	*getBaycount( )*	*getTime( )*	*setTime()*	*getDate()*	*setDate()*	*setBayCount ()*			
Fixedcost ***- Service price/ MoT price***	***HasFailed*** ***- returns a boolean list of the vehicle that had failed an MoT test and the reasons behind failing***	***Comme ntary*** ***- stores the reason behind a vehicle failing an MoT test***	***VehicleI D*** ***- stores each custome r's vehicleI D***	***Pass*** ***- returns a boolean as to whether a vehicle has passed***	***Fail*** ***- returns a boolean as to whether a vehicle has failed***	***Time*** ***- stores the time of the booking made***	***Date*** ***- stores the date of the booking made***		

Group Grading Table

Jaseera Parvin Mohamed Abubacker **M** **Justification:** **Met up regularly in group meetings and was vocal in participating in group discussions.**
Nathan Winslow **M** **Justification:** **Was likely the best candidate to take charge of such a stressful project in our group, however it would have been much easier on me if we were able to meet up more frequently than scheduled to sort out tables relating to my module in the Database.**
Sanjay Manikandhan **M** **Justification:** **Was very willing to meet up with myself regularly even outside of our group meetings.**
Rebecca Jane-Bell **F** **Justification:** **Missed quite a lot of meetings and ultimately made things more stressful and burdening for myself by not making enough effort to get together to discuss how we should or could integrate features in connection with both of our modules how other groups naturally were, thus it would have made life much easier for me had she met up with me regularly to allow both of us to assist each other in finishing our related modules.**