Project Development for Blood Bank Application and Convertor for Software Testing

Rosziati Ibrahim
Universiti Tun Hussein Onn Malaysia, Malaysia, https://orcid.org/0000-0002-9176-0309

Mizani Mohamad Madon
Universiti Tun Hussein Onn Malaysia, Malaysia, https://orcid.org/0000-0001-9892-1112

Zhiang Yue Lee
Universiti Tun Hussein Onn Malaysia, Malaysia, https://orcid.org/0000-0003-0412-7204

Piraviendran A/L Rajendran
Universiti Tun Hussein Onn Malaysia, Malaysia, https://orcid.org/0000-0002-5678-2668

Jahari Abdul Wahab
Sena Traffic System Sdn. Bhd., Malaysia, https://orcid.org/0000-0003-4859-7267

Faaizah Shahbodin
Universiti Teknikal Malaysia Melaka, Malaysia, https://orcid.org/0000-0002-8015-4506

Abstract: This paper discusses the steps involve in project development for developing the mobile application, namely Blood Bank Application and developing the convertor for software testing. The project development is important for Computer Science students for them to learn the important steps in developing the application and testing the reliability of the application. The first step involved is the development of the mobile application. Then the convertor is developed to convert the mobile application in ".apk" format into Java program in ".java" format. The java program is then tested under Eclipse environment using JUnit. Finally, the Java program is tested for its capability of generating test cases using SenaTLSParser. The comparison of the results of the time taken to produce test cases is presented using Junit and SenaTLSParser. Based on the results, SenaTLSParser is more reliable compared with JUnit since its response time is less than JUnit. The whole steps involved in this project development are discussed in this paper.

Keywords: Project Development, Education Technology, Software Testing, Test Cases, Java Programming

Citation: Ibrahim, R., Mohamad Madon, M., Lee, Z. Y., Rajendran, P. A., Abdul Wahab, A., & Shahbodin, F. (2023). Project Development for Blood Bank Application and Convertor for Software Testing. In M. Shelley, O.

T. Ozturk, & M. L. Ciddi, *Proceedings of ICEMST 2023-- International Conference on Education in Mathematics, Science and Technology* (pp. 41-52), Cappadocia, Nevsehir. Turkiye. ISTES Organization.

Introduction

Project development is important for students to learn in developing and completing a project within the given time. In this case study, the students are given a project to complete within eight weeks. The project consists of two stages. The first stage is developing the mobile application, namely Blood Bank Application and the second stage is developing the convertor to convert the mobile application in ".apk" format to ".java" format. The purpose of the development of the convertor is to test the practicallity of converting the mobile appplication into Java program. The Java program is then tested under Eclipse environment using Junit. Finally the Java program is tested for its capability of generating test cases using SENATLSParser.

The first step in project development is to gather all the necessary requirements for the project. The second step is to transform the gathered requirements into UML specification. The third step is the development stage. Based on the UML specification, the application is developed. The final step is to test the developed application in order to see the relaibility of the application. Based on the project given, students have to complete these four steps within eight weeks.

Related Work

Nowadays, a software tester has many choices of techniques in order to conduct a software testing. A tester can generate the test cases and testing each module manually but it is time consuming (Afrin and Mohsin, 2017). The implementation of automation tool is necessary in order to reduce cost for testing manually. Example of some of the automation tool for automating the generation of test cases can be found in (Shin and Im, 2017; Du et al., 2019; Elqortobi et al., 2020; Meiliana et al., 2017; Mishra et. Al., 2017; and Ibrahim et al., 2020).

JUnit can be used for repeatable automated software testing within Eclipse (Junit, 2022). It provides a lot of assert functions for testing the share common test data features, expected results, test suites for test organizing and running, and test runner for graphical and textual. An entire or part of object, or interaction between several objects can be implemented also. SENATLSParser can generate test case automatically by examining the source codes line by line. It enables the tester to find the high quality solutions easily (Ibrahim et al., 2022).

In this study, JUnit and SENATLSParser will be used to test a convertor application which can convert the blood bank application with ".apk" format into ".java" source file. The results of the time taken between Junit and SENATLSParser will be compared in order to indicate the performance of Junit and SENATLSParser.

The research of Li et al. (2017) presents DroidBot to support the model-based test input generation with less extra requirements. The main criteria of Droidbot is lightweight UI-guided test input generator for Android apps because it does not need an advanced knowledge on the unexplored code. It gives UI-guided input generation and will generate on-the-fly at runtime. Then, based on the transition model, it will generate the UI-guided test inputs. In order to reach the effectiveness in most cases, we need to generate the input using depth-first strategy.

Next, based on Wang and Liu (2018), software testing is one of the most important criteria in domestic and abroad software researchers to achieve software quality assurance. However, it spends so much time during the software development process. The Particle Swarm Optimization (PSO) can help in optimizing a problem by trying the process iteratively. The approach used based on specific measure of quality. Therefore, the PSO algorithm was introduced to generate the software case and to design novel software test case automatic generation algorithm.

The study by Mao et al. (2016) introduced an Android testing approach called Sapienz. Sapienz performed better compare to the Android Monkey. It is also a practical testing tool. Other than that, even only app's APK file can be accessed, Sapienz still supports in multi-level instrumentation. Besides, Ibrahim et al. (2022) have discussed on generating the test cases with SENATLSParser using Eclipse environment. Based on the research, they found out that SENATLSParser has high reliability and responsed faster than using manual testing. Based on the respective result, it shows that test case optimization is available. The software tester can also discard the source code that redundant. Code smell functionality give results that it is efficient to generate the test cases automatically based on source code (Ibrahim et al., March 2020). Table 1 summarizes the related work for this study.

Table 1. Summary of Related Work

Authors	Technique	Summary
Li et al. (2017)	DroitBot	Support model-based test input generation with less extra requirements
Wang and Liu (2018)	PSO	Results show that it is better than the conventional PSO
Mao et al. (2016)	Android Testing	Perform better compared with Android Monkey
Ibrahim et al. (2022)	SENATLSParser	SENATLSParser has response time faster than manual testing based on the source codes

Project Development

In project development, four main steps are necessary to complete a project within the given time. Figure 1 shows these 4 steps. The project is given to students to be completed within eight weeks. Students must follow the steps given in Figure 1.

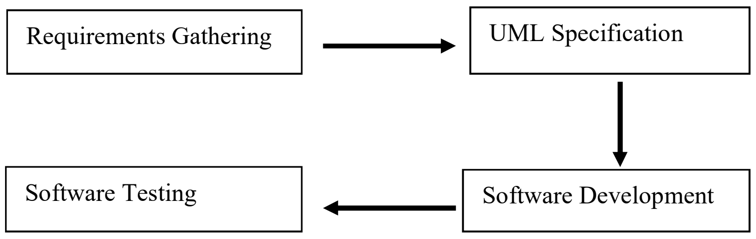


Figure 1. Project Development

Requirements Gathering

Based on Figure 1, the first step is the requirements gathering. Requirements regarding the blood bank mobile application and the convertor must be collected by students. These requirements are then being converted to UML specification. Requirements for blood bank mobile application include the application should be able to have login, register, profile view and make appointment. Meanwhile, requirements for convertor include the capability for the application to read, convert and save.

UML Specification

UML specification consists of few diagrams. Two important diagrams that are needed to develop the application are use case diagram and class diagram. Figure 2 shows an example of the use case diagram for JUnit and Convertor and Figure 3 shows an example of the class diagram for the project.

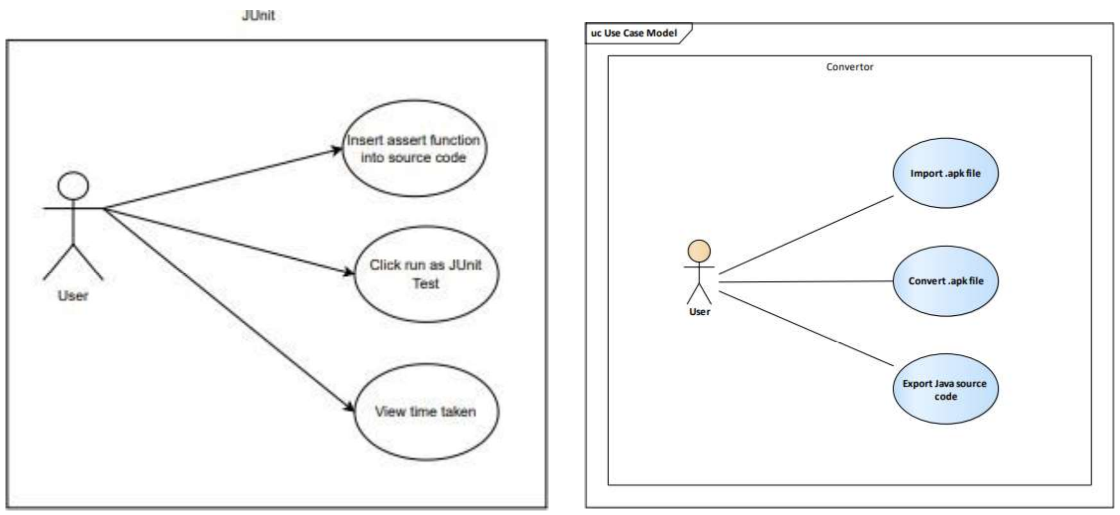


Figure 2. Use Case Diagram for JUnit and Convertor

Based on Figure 2, use case diagram for Convertor, for example, consists of 3 use-cases. They are import, convert and export. Figure 4 shows the example of class diagram for blood bank mobile application.

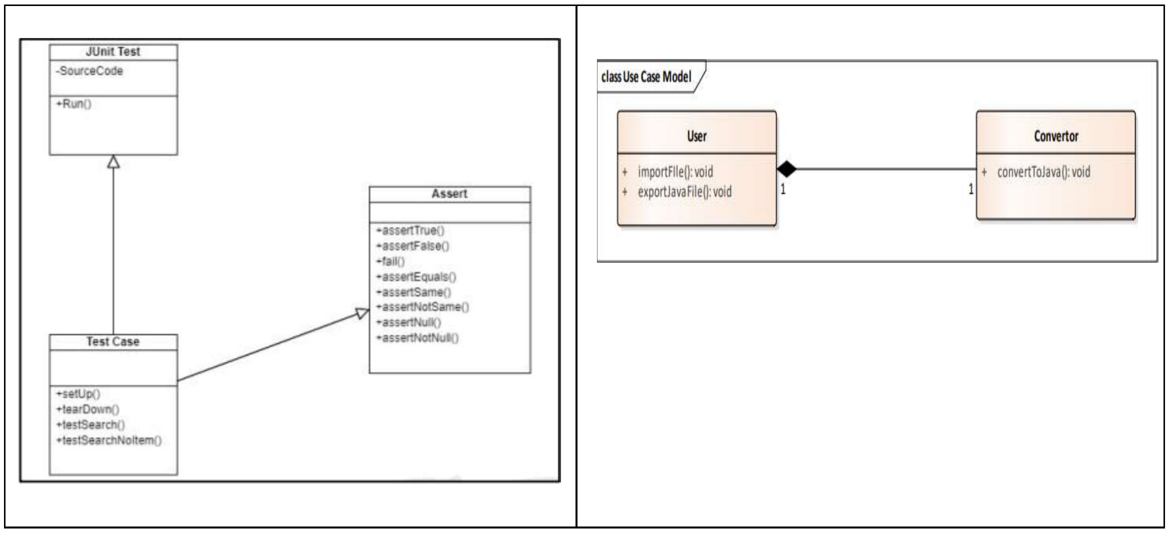


Figure 3. Class Diagram for JUnit and Convertor

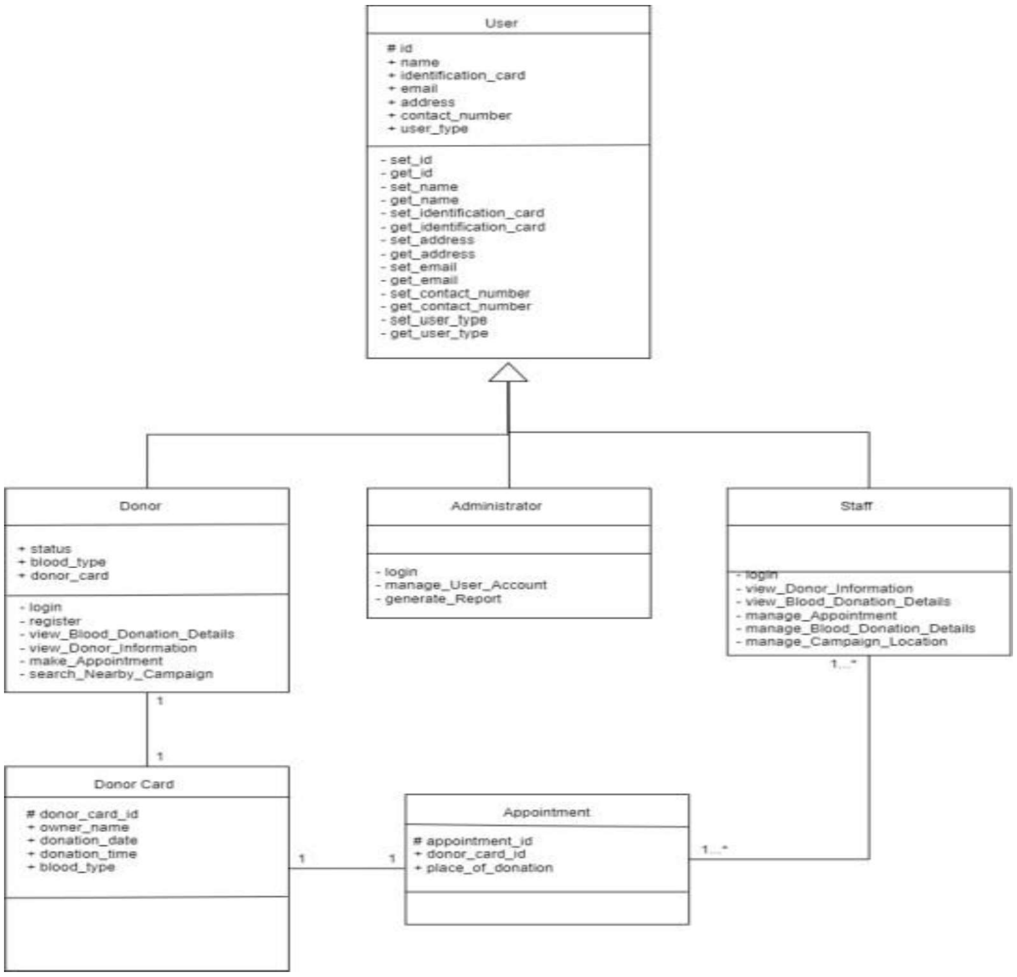
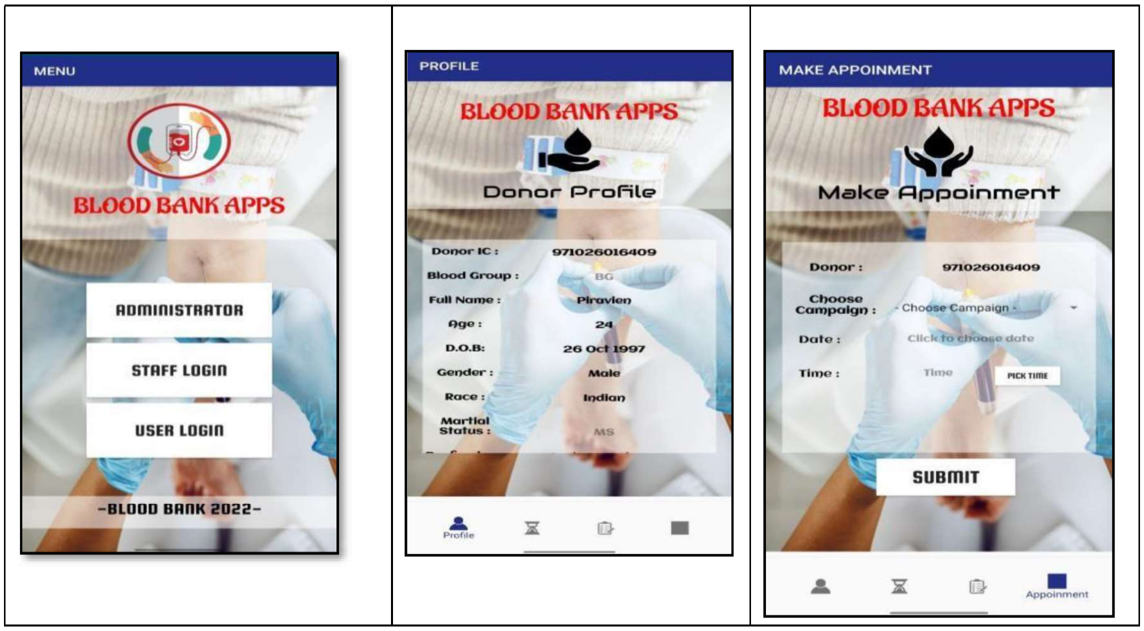


Figure 4. Class Diagram for Blood Bank Mobile Application

Software Development

UML specification have been widely used for software development. Examples of study that used the diagrams in UML specification include (Khuran et al., 2016; Ribeiro et al., 2018; Aman et al., 2014; and Ibrahim et al., 2011). The software can be developed based on the use case diagram in Figure 2 and class diagram in Figure 3. The blood bank mobile application can be developed based on class diagram in Figure 4.


MENU

BLOOD BANK APPS

ADMINISTRATOR

STAFF LOGIN

USER LOGIN

-BLOOD BANK 2022-

PROFILE

BLOOD BANK APPS

Donor Profile

Donor IC : 971026016409
Blood Group : BG
Full Name : Piravien
Age : 24
D.O.B: 26 Oct 1997
Gender : Male
Race : Indian
Martial Status : MS

MAKE APPOINMENT

BLOOD BANK APPS

Make Appoinment

Donor : 971026016409
Choose Campaign : - Choose Campaign -
Date : Click to choose date
Time : Time PICK TIME

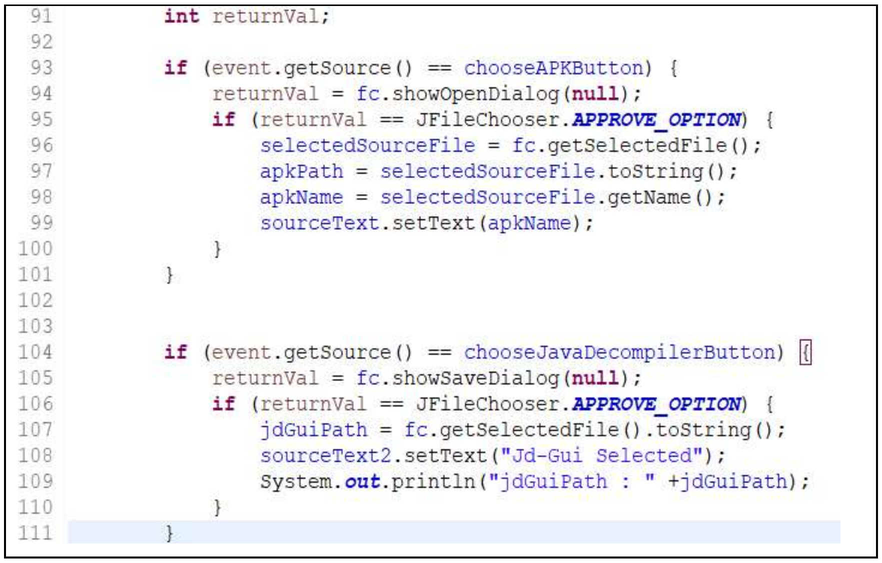
SUBMIT

Figure 5. User Interface for Software Development

The blood bank mobile application, for example, have been developed based on the requirements gathering in step 1 of project development. The user interfaces for login, profile view and make appointment are shown in Figure 5. For the convertor, Figure 6 shows the segmentation codes for the development of the convertor. The convertor is developed using Java programming. For the conversion of blood bank mobile application from ".apk" format to ".java" format. Figure 7 shows the user interface for the conversion process.

Software Testing

Software testing is an important stage to test the reliability of the software being developed. Software testing can be done for any software developed (Jamil et al., 2016; Lawana, 2014). For the project, students have been asked to test the software that they developed using the embedded JUnit function inside the Eclipse environment and the SENATLSParser tool that the students need to import and install in Eclipse environment. Figure 8 shows the process for importing the project in Eclipse environment.

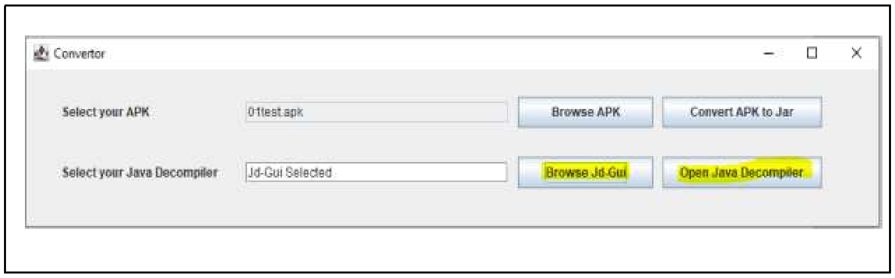


```
int returnVal;

if (event.getSource() == chooseAPKButton) {
    returnVal = fc.showOpenDialog(null);
    if (returnVal == JFileChooser.APPROVE_OPTION) {
        selectedSourceFile = fc.getSelectedFile();
        apkPath = selectedSourceFile.toString();
        apkName = selectedSourceFile.getName();
        sourceText.setText(apkName);
    }
}


if (event.getSource() == chooseJavaDecompilerButton) {
    returnVal = fc.showSaveDialog(null);
    if (returnVal == JFileChooser.APPROVE_OPTION) {
        jdGuiPath = fc.getSelectedFile().toString();
        sourceText2.setText("Jd-Gui Selected");
        System.out.println("jdGuiPath : " +jdGuiPath);
    }
}
```

Figure 6. Segmentation of Source Code for Convertor

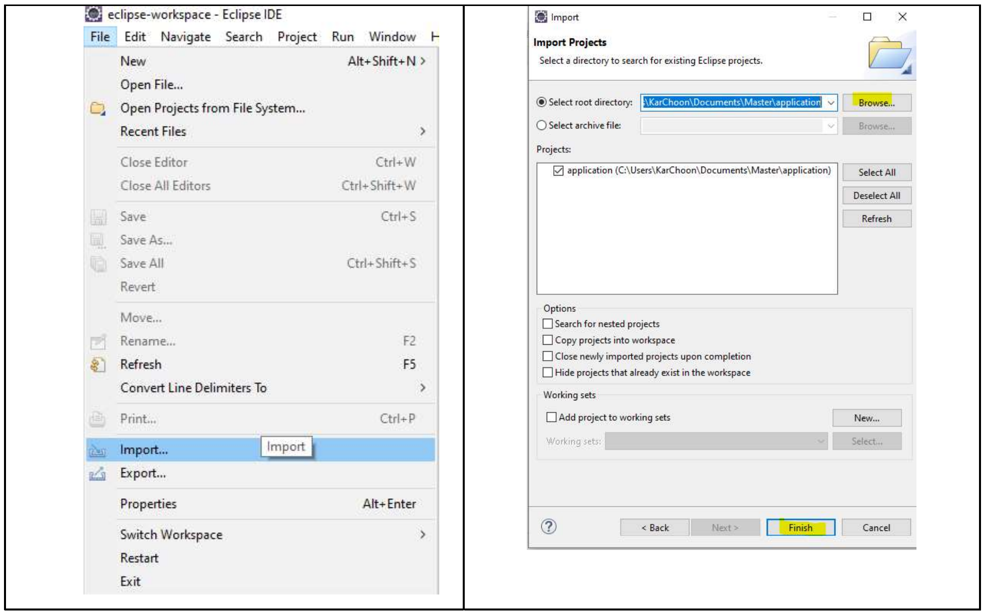


Convertor

Select your APK: 01test.apk

Select your Java Decompiler: Jd-Gui Selected

Figure 7. Convertor Interface



Import

Import Projects

Select a directory to search for existing Eclipse projects.

(●) Select root directory: \KarChoon\Documents\Master\application

( ) Select archive file:

Projects:

[✓] application (C:\Users\KarChoon\Documents\Master\application)

Options

[ ] Search for nested projects

[ ] Copy projects into workspace

[ ] Close newly imported projects upon completion

[ ] Hide projects that already exist in the workspace

Working sets

[ ] Add project to working sets

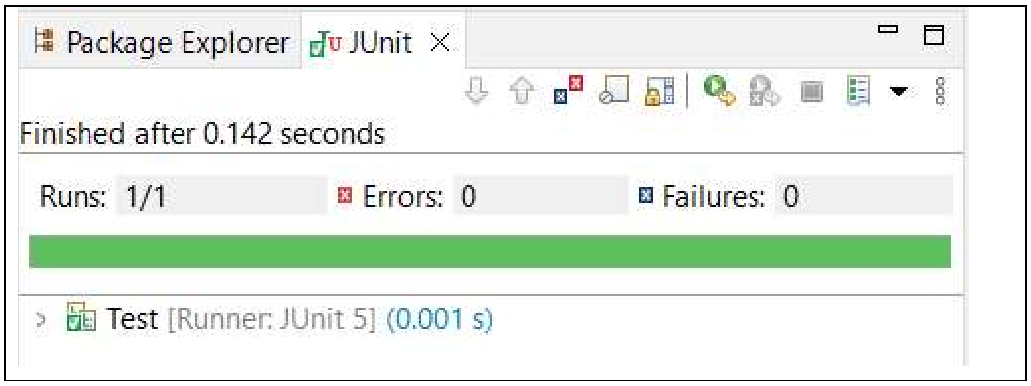
Working sets:

Figure 8. Import Project and Application

Based on Figure 8, by using the convertor and import file, the blood bank mobile application in ".apk" format is able to be converted into Java programming ".java" source codes. Then, SENATLSParser is used for testing the Java source codes. Students are able to learn the whole steps in project development to complete the project within the given period.

Results and Discussion

JUnit has been used to test the blood bank mobile application. The assert function is used for the software testing. Figure 9 shows the assert function is used for the blood bank mobile application. The result for the assert function using JUnit is also shown in Figure 9. The time taken to execute the assert function using JUnit is 0.142s. Meanwhile, SENATLSParser tool is also used for the software testing. The time taken to execute and generate the test cases is 76ms as shown in Figure 10. Based on the execution time using assert function from JUnit and SENATLSParser tool, it has been shown that SENATLSParser is able to execute faster than JUnit. Figure 11 shows the test cases that have been generated aoutomatically from SENATLSParser tool.

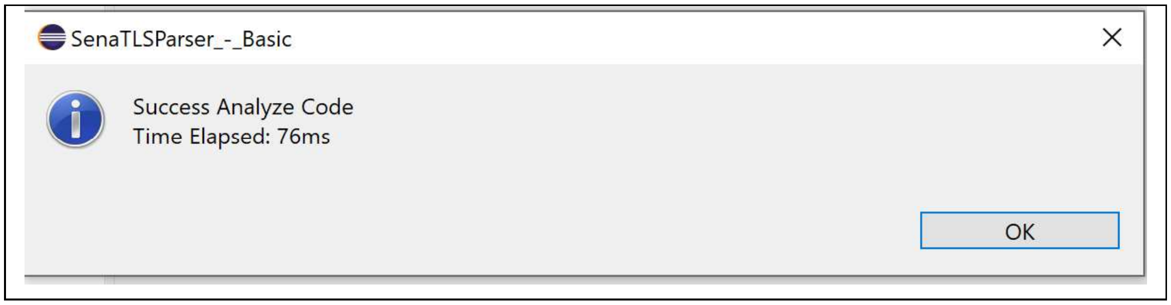


Finished after 0.142 seconds

Runs: 1/1 Errors: 0 Failures: 0

Test [Runner: JUnit 5] (0.001 s)

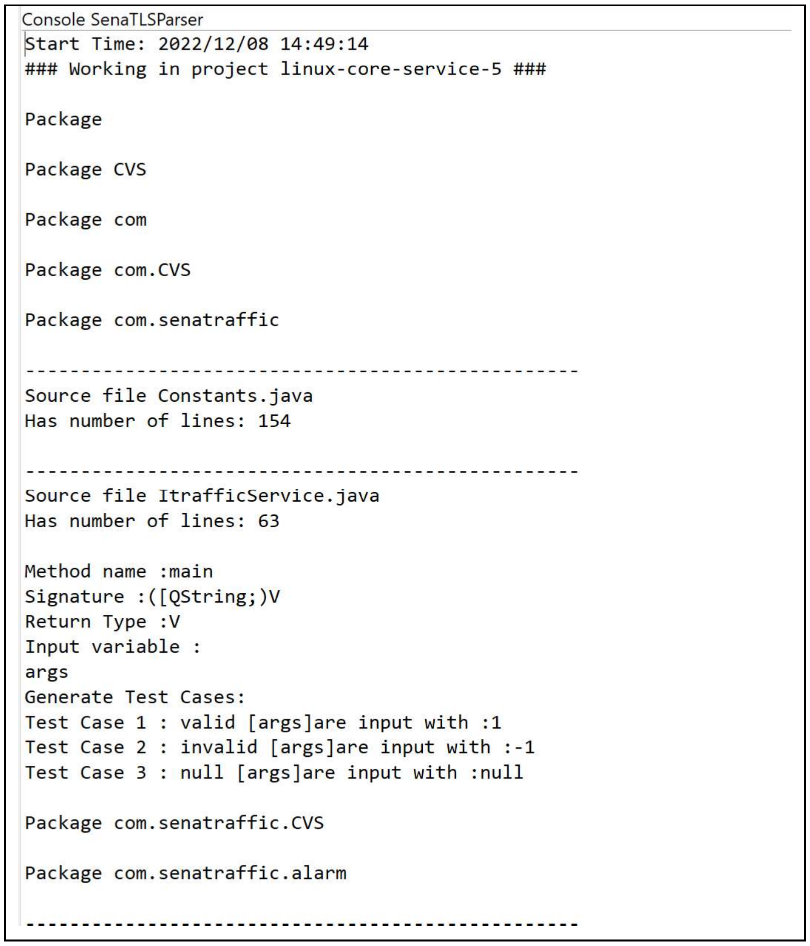
Figure 9. JUnit Testing



Success Analyze Code
Time Elapsed: 76ms

Figure 10. SenaTLSParser

The convertor is also tested for software testing using assert function from JUnit and SENATLSParser tool. Three different tests are performed for the convertor. Table 2 shows the results for the test cases generated and the time taken to generate the test cases. Figure 12 shows the graph for the three software testing cases.


Console SenaTLSParser

```
Start Time: 2022/12/08 14:49:14
### Working in project linux-core-service-5 ###

Package

Package CVS

Package com

Package com.CVS

Package com.senatraffic

--------------------------------------------------
Source file Constants.java
Has number of lines: 154

--------------------------------------------------
Source file ItrafficService.java
Has number of lines: 63

Method name :main
Signature :([QString;)V
Return Type :V
Input variable :
args
Generate Test Cases:
Test Case 1 : valid [args]are input with :1
Test Case 2 : invalid [args]are input with :-1
Test Case 3 : null [args]are input with :null

Package com.senatraffic.CVS

Package com.senatraffic.alarm

--------------------------------------------------
```

Figure 11. Console for SenaTLSParser

Table 2. Results of Test Cases

Function Name	***No. of Tests***	***Time Taken (ms)***	
		JUnit (ms)	***SENATLSParser (ms)***
Convertor.java	1	31	1
	2	25	1
	3	27	2
Average Time Taken (ms)		27.67	1.33

Based on Table 2, SENATLSParser tool gives good results as compared with JUnit for the three different tests. The average of 1.33ms has been used to generate the test cases using SENATLSParser tool as compared with 27.67ms using assert function from JUnit. This shows that SENATLSParser tool is capable to generate faster test cases automatically compared with JUnit.

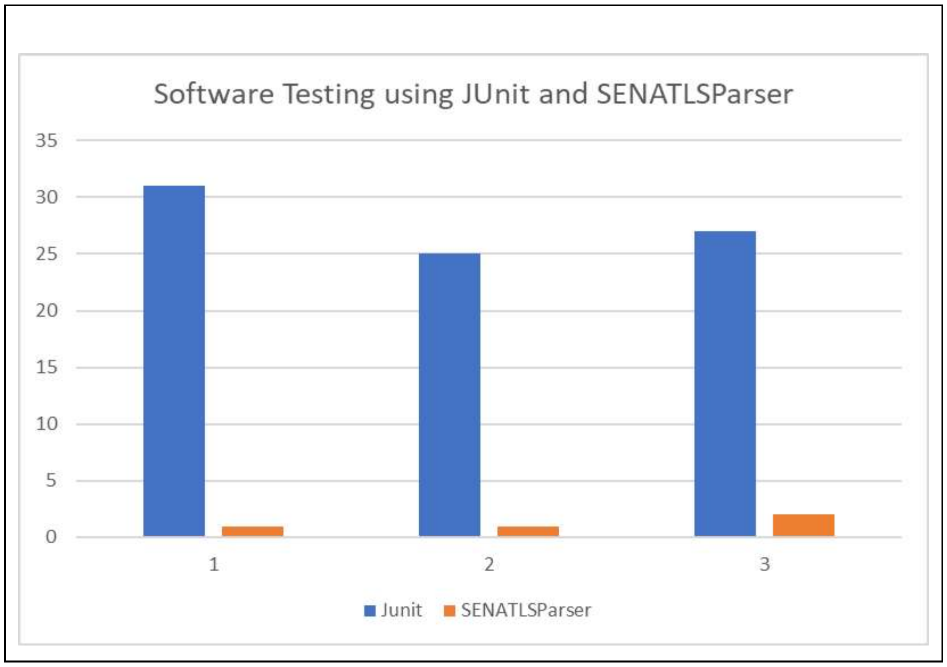


Figure 12. Results for Generating Test Cases using JUnit and SenaTLSParser

Based on Figure 12, SENATLSParser tool is more reliable to be used as a tool for software testing in term of generating the test cases automatically. The three cases have shown that SENATLSParser outperformed the JUnit.

Conclusion

In conclusion, the students have been successfully implemented the project within eight weeks that has been given to them. Project development is important to enable the successful of implementation of a project. The convertor is used to convert the blood bank mobile application from ".apk" format into Java programming ".java" source codes by using the method of executing window batch command (.bat) file and Java Decompiler. After converting, the java source code is testing using JUnit and SENATLSParser. The comparison results of time taken between JUnit and SENATLSParser shows that SENATLSParser uses the least time for software testing. Based on the result, SENATLSParser is more reliable as compared with JUnit since its execution time is less than JUnit.

Acknowledgements

The authors would like to thank Universiti Tun Hussein Onn Malaysia (UTHM) for supporting this research. The authors received funding for this study from Industry Grant from SENA Traffic System Sdn. Bhd. under Grant Vote No M081.

References

Afrin, A., and Mohsin, K. (2017). “Testing approach: Manual testing vs automation testing.” *Global Sci-Tech*, 9(1), 55-60, 2017.

Aman, H., and Ibrahim, R. (2014). “Formalization of Transformation Rules from XML Schema to UML Class Diagram.” *International Journal of Software Engineering and its Applications,* 8 (12), pp. 75-90, 2014.

Du, Y., Pan, Y., Ao, H., Alexander, N.O., and Fan, Y., (2019). “Automatic Test Case Generation and Optimization Based on Mutation Testing,” *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C).* doi:10.1109/qrs-c.2019.00105, 2019.

Elqortobi, M., Rahj, A., Bentahar, J. and Dssouli, R., (2020). “Test Generation Tool for Modified Condition/Decision Coverage: Model Based Testing,” *In Proceedings of the 13th International Conference on Intelligent Systems: Theories and Applications (SITA’20).* Association for Computing Machinery, New York, NY, USA, Article 38, 1–6, 2020.

Ibrahim, N., Ibrahim, R., Saringat, M.Z., Mansor, D and Herawan, T. (2011). “Consistency rules between UML use case and activity diagrams using logical approach.” *International Journal of Software Engineering and its Applications,* 5 (3), pp. 119-134, 2011.

Ibrahim, R., Ahmed, M., Nayak, R. and Jamel, S., (March 2020). “Reducing Redundancy of Test Cases Generation using Code Smell Detection and Refactoring”. *Journal of King Saud University - Computer and Information Science,* Volume 32, Issue 3, March 2020.

Ibrahim, R., Amin, A. A. B, Jamel, S. and Abdul Wahab, J., (2020). “EPiT: A Software Testing Tool for Generation of Test Cases Automatically,” *International Journal of Engineering Trends and Technology,* 68(7),8-12, 2020.

Ibrahim, R., AbuSalim, S. W. G., Jamel, S. and Abdul Wahab, J., (2022). “Sena TLS-Parser: A Software Testing Tool for Generating Test Cases” *International Journal of Advanced Computer Science and Applications* (IJACSA), 13(6), 2022.

Jamil, M. A., Arif, M., Abubakar, N. S. A., & Ahmad, A. (2016). “Software testing techniques: A literature review.” In *2016 6th International Conference on Information and Communication Technology for The Muslim World (ICT4M)* (pp. 177-182). IEEE.

JUnit 2022. *JUnit4.* [Online]. Available: https://junit.org/junit4/

Khurana, N., Chhillar, R.S., and Chhillar, U.A., (2016). “A novel technique for generation and optimization of test cases using use case, sequence, activity diagram and genetic algorithm,” *Journal of Software*, vol. 11, no. 3, pp. 242-250, 2016.

Lawanna, A. (2014). “The theory of software testing.” *AU Journal of Technology, 16*(1), 35-40. 2014.

Li. Y, Yang. Z, Guo. Y and Chen. X, (2017). "DroidBot: a lightweight UI-Guided test input generator for android," *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pp. 23- 26.

Mao, K, Harman, M, and Jia, Y. (2016). “Sapienz: multi- objective automated testing for Android applications.” *In Proceedings of the 25th International Symposium on Software Testing and Analysis (ISSTA 2016).*

Association for Computing Machinery, New York, NY, USA, 94–105. https://doi.org/10.1145/2931037.2931054

Meiliana, I. Septian, R. Daniel A., and Gaol, F., (2017). “Automated Test Case Generation from UML Activity Diagram and Sequence Diagram using Depth First Search Algorithm,” *ICCSCI,* 2017.

Mishra, D., Mishra, R. Das, K., and Acharya, A., (2017). “Test Case Generation and Optimization for Critical Path Testing Using Genetic Algorithm,” *SocProS,* 2017.

Ribeiro, F.G.C., Pereira, C.E. Rettberg, A., and Soares, M.S., (2018). “Model-based requirements specification of real-time systems with UML, SysML, and MARTE,” *Software & Systems Modeling*, vol. 17, no. 1, pp. 343-361, 2018.

Shin K.W., and Lim, D.J., (2017). “Model-based automatic test case generation for automotive embedded software testing,” *International Journal of Automotive Technology*, 19(1), 107–119. doi:10.1007/s12239-018-0011-6, 2017.

Wang, Z. and Liu, Q., (2018). "A Software Test Case Automatic Generation Technology Based on the Modified Particle Swarm Optimization Algorithm," *2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS),* pp. 156-159, 2018.