**K-Means Based Automated Student Group Generation**

**SMART PROJECT MANAGEMENT SYSTEM**

**2021-234**

Nipun Eranda Amarasekara

IT18009446

BSc (Hons) in Information Technology

Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

October 2021

**K-Means Based Automated Student Group Generation**

**SMART PROJECT MANAGEMENT SYSTEM**

**2021-234**

Nipun Eranda Amarasekara

IT18009446

Dissertation submitted in partial fulfillment of the requirements for the Bachelor of Science

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

October 2021

# DECLARATION

I declare that this is my own work and this dissertation1 does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. Also, I hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).



Signature: Date: 13/10/2021

The above candidate has carried out research for the bachelor’s degree Dissertation under my supervision.

Signature of the supervisor: Date:

# ABSTRACT

Worldwide universities conduct various types of Information Technology Projects. Most of the universities uses common project management solutions which includes industrial quality functionalities. However, there are requirements that doesn’t come with these common project management systems. Automatic Group Generation is one of the major requirement in universities. Project practical teaching has an important significance to enhance students’ practical ability and is also an important part in teaching reform in colleges and universities, however, there is lacking of scientific strategies to grouping students in project practical teaching [5]. Universities doesn’t use any logical way for grouping process. Usually groups will randomly generate or according to a registration list. However, the efficiency and effectiveness of a such group is unpredictable. Therefore, a Computer Based Group Generation system becomes a major requirement for universities.

The Smart Project Management System is proposed to overcome above mentioned requirements as well as most of the common features. System will generate project groups using student Skills, GPA and previous project solutions. With the GitHub integration to the system, grouping process can reach maximum accuracy. However, to calculate the accuracy system need to access various types of skills of the students. Therefore, System will provide facility to manage student profiles by their own, and students can define their very own skills on Programming knowledge. Using this strategy system can use the data that required for the grouping process. Once the process is finished assigned lecturers can check the groups and publish groups among students.

The end result of the solution is a project management system with highly customizable web application. Since the system will an open source solution, Universities can expand the capabilities of the system by their own. Smart project management can be used to improve efficiency and effectiveness in group projects.

*Key words: Project management system, Student grouping, grouping algorithm, data clustering*

# ACKNOWLEDGEMENT

I would like to present my deep gratitude for the endless support and guidance helped by our supervisor Ms. Uthpala Samarakoon (Senior Lecturer - Faculty of Computing SLIIT) for her valuable and constructive suggestions during the planning and development of this research work. Her guidance and support helped us the way to get into competitions and show to our talents and complete this SMART Project Management System. Also I would like to give second gratitude to our co-supervisor Ms. Kugathasan Archchana (Lecturer - Faculty of Computing SLIIT) for her encouragement and much needed motivation. Without our supervisors’ major contribution, I would have never obtained this accomplishment.

Not forgetting all the contributors who participated by answering our questionnaires, filling google forms and testing the solution. Furthermore, we would like to thank our parents for their patience, time and providing resources to acquire the needful and for other expenses. Special thanks to my team mates who worked as a team to make this research more effective and success. Thank you for your time, encouragement, and recommendations.

**TABLE OF CONTENTS**

[DECLARATION i](#_Toc85041083)

[ABSTRACT ii](#_Toc85041084)

[ACKNOWLEDGEMENT iii](#_Toc85041085)

[LIST OF FIGURES vi](#_Toc85041086)

[LIST OF TABLES vii](#_Toc85041087)

[LIST OF ABBREVIATION viii](#_Toc85041088)

[LIST OF APPENDICES ix](#_Toc85041089)

[1. INTRODUCTION 1](#_Toc85041090)

[1.1 Background & Literature survey 2](#_Toc85041091)

[1.2 Research Gap 6](#_Toc85041092)

[1.3 Research Problem 8](#_Toc85041093)

[1.4. OBJECTIVES 10](#_Toc85041094)

[1.4.1 Main Objectives 10](#_Toc85041095)

[1.4.2 Specific Objectives 11](#_Toc85041096)

[2.0 METHODOLOGY 12](#_Toc85041097)

[2.1 Feasibility Study 14](#_Toc85041098)

[2.2 Requirement Gathering and Analysis 14](#_Toc85041099)

[2.3 Design 15](#_Toc85041100)

[2.4 Implementation 16](#_Toc85041101)

[2.4.1 Information Extraction 16](#_Toc85041102)

[2.4.2 Clustering and Decision Tree 17](#_Toc85041103)

[2.4.3 Custom Grouping Algorithm 19](#_Toc85041104)

[2.5 Software Solution Development Components 25](#_Toc85041105)

[2.5.1 Web Application Development 25](#_Toc85041106)

[2.5.2 API development 25](#_Toc85041107)

[2.5.3 Database Handling 26](#_Toc85041108)

[2.5.4 Testing 27](#_Toc85041109)

[2.5.5 Deployment 32](#_Toc85041110)

[3. RESULTS AND DISCUSSION 33](#_Toc85041111)

[3.1 Results 33](#_Toc85041112)

[3.2 Research Findings 35](#_Toc85041113)

[3.3 User Interfaces 36](#_Toc85041114)

[3.4 Discussion 38](#_Toc85041115)

[4. SUMMARY OF THIS RESEACH CONTRIBUTION 41](#_Toc85041116)

[5. CONCLUSION 42](#_Toc85041117)

[6. REFERENCES 43](#_Toc85041118)

[7. APPENDICES 45](#_Toc85041119)

[Appendix – A: Sample questionnaire 45](#_Toc85041120)

[Appendix – B: Sample questionnaire response 46](#_Toc85041121)

[Appendix - C: Plagiarism Score 01 47](#_Toc85041122)

[Appendix - D: Plagiarism Score 02 48](#_Toc85041123)

[Appendix - E: Plagiarism Score 03 49](#_Toc85041124)

# LIST OF FIGURES

[Figure 1: factors that can be considered for group formations 4](#_Toc85041280)

[Figure 2: Feature Comparison with the existing solutions and proposed system 7](#_Toc85041281)

[Figure 3: Grouping Algorithm flow chart 10](#_Toc85041282)

[Figure 4: SDLC Life Cycle 12](#_Toc85041283)

[Figure 5: Breakdown Structure 13](#_Toc85041284)

[Figure 6 - System Architecture Diagram 15](#_Toc85041285)

[Figure 7: Data cleaning process code segment 17](#_Toc85041286)

[Figure 8: Clustering process code segment 18](#_Toc85041287)

[Figure 9: Sample clustering process results 18](#_Toc85041288)

[Figure 10: Decision tree algorithm output 19](#_Toc85041289)

[Figure 11: Decision Tree code segment 19](#_Toc85041290)

[Figure 12: Preparation for the algorithm 20](#_Toc85041291)

[Figure 13: Initial Group Formation Process 21](#_Toc85041292)

[Figure 14: Group balancing process code segment 23](#_Toc85041293)

[Figure 15: Group Balance Iterator code segment 24](#_Toc85041294)

[Figure 16: Sample output from the API 25](#_Toc85041295)

[Figure 17: Export the system 26](#_Toc85041296)

[Figure 18: Database scripts 27](#_Toc85041297)

[Figure 19: Sample postman output 27](#_Toc85041298)

[Figure 20: Execute the system 32](#_Toc85041299)

[Figure 21: First group generation Output 33](#_Toc85041300)

[Figure 22: Second group generation output 34](#_Toc85041301)

[Figure 23: Student Survey 36](#_Toc85041302)

[Figure 24: Student Group Generation UI 37](#_Toc85041303)

[Figure 25: Decision tree graphical output 39](#_Toc85041304)

[Figure 26: Sample system output 40](#_Toc85041305)

# LIST OF TABLES

[Table 1: Test case 01 28](#_Toc85029943)

[Table 2: Test case 02 28](#_Toc85029944)

[Table 3: Test case 03 29](#_Toc85029945)

[Table 4: Test case 04 29](#_Toc85029946)

[Table 5: Test case 05 30](#_Toc85029947)

[Table 6: Test case 06 31](#_Toc85029948)

[Table 7: Comparison between previous systems 35](#_Toc85029949)

[Table 8: Research Contributions 41](#_Toc85029950)

# LIST OF ABBREVIATION

Abbreviation Description

GPA Grade Point Average

SLIIT Sri Lanka Institute of Information Technology

API Application Programming Interface

SDLC Software Development Life Cycle

XML Extensible Markup Language

ITP Information Technology Project

JS JavaScript

UI User Interface

UX User Experience

JDBC Java Database Connectivity

AWS Amazon Web Services

CSV Comma Separated Values

# LIST OF APPENDICES

Appendix Description Page

[Appendix - A Sample questionnaire 35](#_Appendix_–_A:_1)

[Appendix - B Sample questionnaire response 36](#_Appendix_–_B:_1)

[Appendix - C Plagiarism Score 01 37](#_Appendix_-_C:)

[Appendix - D Plagiarism Score 02 38](#_Appendix_-_D:)

[Appendix - E Plagiarism Score 03 39](#_Appendix_-_E:)

# 1. INTRODUCTION

Project Management is quite simple with current technology and most of the time Project Manager has the technology to manage most of the management process automatically using existing solutions. For the industry, existing project management tools are powerful enough to reduce the workload for the manager. However, when it comes to the Undergraduate Project Management, most of the management tools are more advanced and also some of the required features are not available with common management solutions. Therefore, we’ve decided to develop a Project Management System where we define our own features and also included with common features.

Automate Group generation is one of the required feature for universities. However, there are existing systems used to automate group generation. But our goal is to develop a system with fair group generation. This system uses specific data from the students and using machine learning approach, we’re developing a system where group generation can be done fairly. Most of the time students suffering and complaining about conflicts among members. Our goal is to reduce those conflict by grouping student with similar skill structure. Existing project management systems doesn’t come with this feature.

To achieve this goal, we need to retrieve student data to process. But the problem is how we gather all these data. Students grow up with different skills and goals. Therefore, system need to gather data about student’s skills and goals to come up with a conclusion and continue the grouping process. Student grouping is depending on the skills and goals that student define in the system. Therefore, the overall success rate is depending on the student data which defined by the student.

## **Background & Literature survey**

Project Management Systems doesn’t come with facility to automatically generate project groups. Instead user must create a group and assign team members to the group. It is a time consume process when there are thousands of students. But from onward there are lot of facilities provided by the project management system and most of the time they’re not even use most of the facilities. Problem with all these system is the main focus is Industrial projects. They’re not exactly made for university use. By creating an open source project management system, users can create solutions for their requirements and also they can expand the system by themselves. Most of the project management systems are not open source solutions.

According to the existing grouping strategies in project practical teaching, teachers often grouping students according to their willingness, or by their enrollment numbers, or dividing them into groups at random [1][5]. By dividing students by these strategies not very accurate and effective. Students have different skills, different knowledge on programming and also their theoretical and practical skills are also different. By using non logical strategies for grouping effect on group management and also the success rate of the project. Therefore, we need a solution to integrate this grouping facility with the project management system. But the problem is student attributes that a university may consider is may be different than other universities when considering project groupings. By creating an open source project management system solves this issue. So the final solution will be a configurable and extendable project management system.

Since there are no automatic grouping facilities in any project management solutions, we need to integrate the grouping facility with the project management system. However, we need to implement the grouping facility first. In order to create better study groups, automatic computer-supported methods are proposed. This leads to several important advantages. Especially, it is possible to consider a large amount of information even from very different sources. Group creation can be performed very fast and anytime on demand by students or a learning system itself. Computer support also allows creation of anonymous groups in which members do not know their identity [16]. Therefore, we decided to explore some of the research projects and papers that researchers proposed. There are existing grouping solutions created by varies researchers. There is a grouping solution using fuzzy clustering. By grouping students considering scores of students’ previous professional curriculum as clustering characteristic quantity index, and bases on the index, divides students into different classes, then grouping students by heterogeneity grouping method reasonably and accurately [5]. In this research they divide students into 3 different classes. That is “very important”, “important” and “common”. After that they apply each class values to a weight comparing matrix table and based on weight scores they generate a project team structure based on class values. Issue in this grouping method is it only considers professional curriculum. In this research they used Object-Oriented Programming, Database Foundation and Application, Data Structures and Operating systems. There are multiple Object-Oriented Programming languages and likewise there are multiple sub sections to be consider when we need a fair grouping strategy. But this existing solution is also fair enough to increase the success rate and efficiency. But this solution can be extended and improved by using multiple algorithms to sort end results and increase efficiency.

GroupEng is another solution created by researchers to effectively assign student groups by applying multiple User-prioritized Academic and Demographic Factors [2]. This solution is far more complex than the other one. Using GroupEng users can group students using different types of rules. This fairly removes the above mention issue of lack of options. GroupEng solution uses heuristic guided stochastic greedy algorithm. Greedy algorithm simply iterates through the rules in priority order fixing “breaks” by swapping students between groups [2]. The following criteria are used to determine if groups “break” a rule:

Cluster: broken by groups which contain exactly 1 student with the given value.

Aggregate: broken by groups which mix values for the given attribute [2]. Usually1 group will break this rule for each value under consideration because the number of students is not evenly divisible by the group size. This is considered normal operation.

Distribute: broken by groups which have more or less than their “share” of members with any attribute value under consideration. This “share” is (#students with attribute value) / (# groups), or the two integers on either side of this value for fractions [2].

Balance: broken by groups for which the standard deviation of group member’s values for the attribute is larger than a tolerance times the class’s standard deviation for that attribute: StDev(group) > tol \* StDev(students) [2].

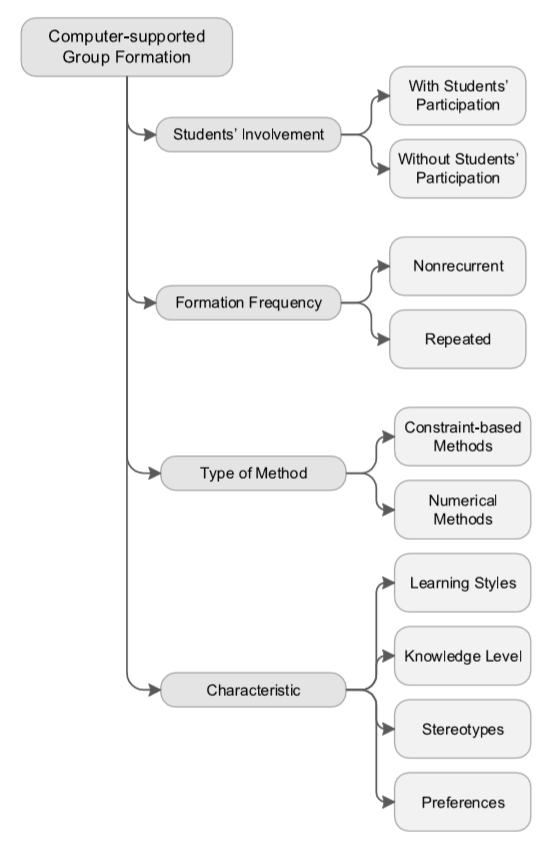
Once a high priority rule is finished, swaps which would break that rule are not allowed while processing lower priority rules [2]. Uneven group sizes are handled by filling out the smaller groups with placeholder students which are considered to have the lowest value in the class when computing deviations for balance rules and otherwise ignored [2]. GroupEng Solution achieved higher accurate results than other research solutions that we’ve studied.

Figure 1: factors that can be considered for group formations

When we consider grouping mechanisms there’re wide range of options can be selected. But the properties that include in those mechanisms are differ from another. Figure 1 shows the different factors that can be used for group formations. Since there are several choices to have, we need to select which factors can be used to achieve maximum efficiency and fairness.

Varies Researchers have worked on this topic and they’ve introduced solution as a part but not as a whole system. All these solutions are standalone solutions which not included in any project management systems that are currently in use. Our goal is to implement this Automatic Group Generation functionality to the Project management system and by doing that create a Smart project management system which satisfies major requirements that universities require.

## **Research Gap**

Above mentioned research solutions are targeted for a specific requirement that only satisfy specific dataset. The problem is, is it configurable for our own requirements and is it possible to use these solutions for our own system. Since all above mentioned systems are made for a single task it is bit complex to integrate it to the proposed project management system. The target is to use existing solutions and reconfigure them to satisfy our own requirements. Some of the solutions are complex to be configured and some of them are not enough sufficient to satisfy our requirement. So to implement an efficient and accurate system, we need to study these existing systems to configure and extend capabilities of the existing solutions.

According to the existing systems [5] [2] they specify what type of data need to be supply to the system and the end results that they expect from the implemented system. When we consider the Fuzzy Clustering algorithm [5], It considers student GPA and the field that student interested in. By using only these two components they managed to implement the grouping solution which is effective enough to make difference in the success rate. But the problem is variables that they used for the calculation is not sufficient to expect highest accurate end results. However, the GroupEng open source solution [2] is way ahead than the first research solution using Fuzzy Clustering Algorithm. But the problem GroupEng is a far more complex solution is for the expected solution. But the grouping process is mostly similar to the system that we’re expecting to build. So this solution is also a valuable resource for the development process.

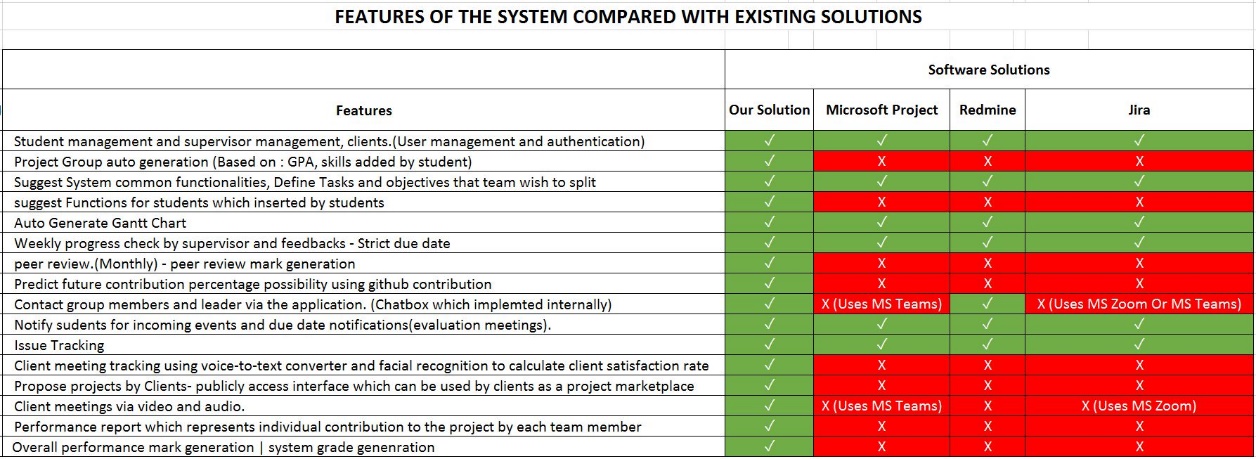


Figure 2: Feature Comparison with the existing solutions and proposed system

Figure 2 shows a table that we created by finding functionalities in the existing project management systems. Cells with ✕ symbol show the functionality is not implemented in the system and cells with ✓ symbol shows the functionality is implemented in the system. As you can see the grouping mechanism is not implemented in neither software solutions and it is a major requirement for any university to improve effectiveness and efficiency of university projects.

## **Research Problem**

Most Universities uses a standard grouping mechanism. One of the researches addressed this problem. According to the existing grouping strategies in project practical teaching, teachers often grouping students according to their willingness, or by their enrollment numbers, or dividing them into groups at random [1]. Using this strategy, the grouping problem is solved. But the problem is grouping that way creates conflicts within groups and students start to complain about the situations. But if the university create groups for them. They complain about different things. when the students, grouping for the project groups, there is a problem with that. Sometimes a group can have included all students as higher GPA students. Some project group can have all students as weak student. If university take the responsibility for grouping, usually the process is randomly grouping using a student list. There are some online tools which can be used for grouping. Those tools also using random group generation using a created student list [3]. But that is not effective and not fair. There are some qualified tools to do the process but it doesn’t come with project management tools. Those tools are available in standalone and also in web based applications [4]. Which leads to confusion. Because for the whole process university have to use different tools and it is bit confusing for management.

The best solution is to develop a system where project grouping can be done fairly and effectively. In a research paper which were published in 2010, It contains a solution for the grouping. It uses student’s previous compulsory professional curriculums scores to do the weighted statistical calculation according to their importance degree in the software project practical teaching, thus the professional mastery degree of student can be determined. Then, using the fuzzy clustering strategy to classified the students into different classes, based on this, doing the grouping action [5]. In this paper the scenario is different but the strategy we’re planning to do is the same. There is a research paper which they developed an open-source program, GroupEng, which assigns groups according to guidelines from education research [2]. Guidelines include avoiding isolating women or minorities and assigning multi-disciplinary groups of mixed abilities [2]. The program operates on a set of simple, flexible, faculty defined rules, keeps data local, and ensures “fairness” of group strengths [2]. Same procedure will to be used in this research where students will be grouped based on their skills and using their academic progress. And the major problem is how to determine the accuracy of the group formation. So we need to find an easy way to determine success rate of the implemented solution or otherwise we need to use the implemented solution for the test run for a whole project and see the end results by the end of the project. Easiest way to do is check the accuracy of expected result with the generated results of the balance between the team composition. Such as GPA and matching skills.

Since there is no grouping strategy implemented in current existing project management systems, Options are to use a standalone system to just for grouping requirement, or to implement a system where these required functions are integrated in to the same system so that system can automatically do the management easily.

# 1.4. OBJECTIVES

## **1.4.1 Main Objectives**

The main objective of the research is to implement a system that can automate the student project group generation. Easiest way to do the grouping is the current system where grouping is depending on the student registration or random group formation. However, that strategy is efficient by not effective. Conflicts inside the team increases while doing so. So the option is to generate student groups in a well manner and fairly. So the objective is to create an Algorithm using machine learning to increase the efficiency in group generation and the effectiveness in group working environment by generate groups considering student skills.

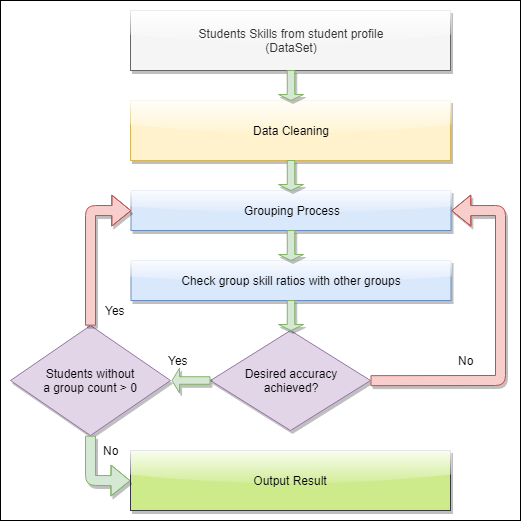


Figure 3: Grouping Algorithm flow chart

Figure 3 shows the grouping algorithm which to be used in our system. System will iterate the grouping process until the group composition rate of every group is at least fair. To create the system, we need to find a matching machine learning approach and to do so we need to complete some sub objective. Sub objectives are given below as specific objectives.

## **1.4.2 Specific Objectives**

In order to achieve the main objective there are specific objective to be completed. Specific objective that need to be attained is as followed.

1. Data collection.

Data collection is a major role in the process. Data is a key resource that we need to have for the data analysis. Without sample student data, we cannot proceed to later steps. Using sample data, we can create the data cleaning processes and start the model training. For data collection we’re using google forms. By sharing google forms to the students in SLIIT we’ll be able to get enough data for training and testing data.

1. Analyze the data and find relationships between data.

After data collection we need to find relationships with data and analyze how to use them for the grouping process. Some of the fields will be only used for separate students for different sections. For an instance there are 3 types of developers. Desktop developers, Web developers, Mobile developers. By separating data set can reduce processing time.

1. Using data analysis find an optimal algorithm or algorithms to achieve the expected results.

Currently we’re planning to use fuzzy clustering algorithm for grouping process. But later we might be needing additional algorithms for additional processes. Before identifying algorithms, we need to analyze data and then we can get an idea which algorithm suitable, efficient and effective.

# 2.0 METHODOLOGY

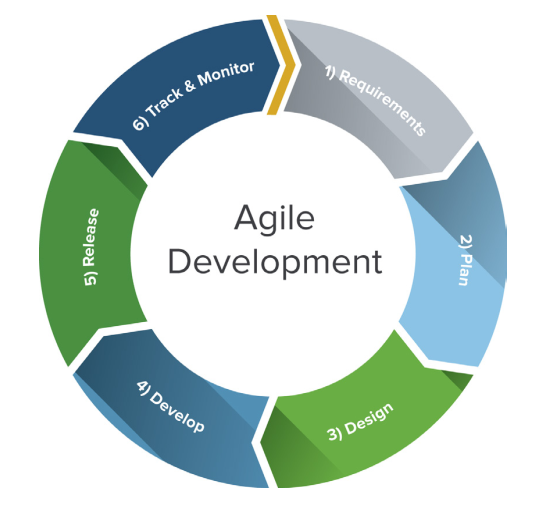
This section covers the procedures carried according to the software Development Life Cycle.

Figure 4: SDLC Life Cycle

For our solution, were using the agile methodology. The Internet economy has altered the rules of software engineering. Traditional development methodologies are too cumbersome to meet the rapidly changing requirements and short product cycles demanded by business. To meet these changing requirements, software developers have developed agile software development methodologies utilizing iterative development, prototyping, and templates [6]. In Each SDLC section will cover specific tasks used to complete the research components.

Since we’re using agile methodology, Development is an iterative process. Below list represents the flow of each iteration.

* Feasibility study
* Requirement gathering
* Designing
* Implementation
* Testing
* Maintenance

Given below is the work breakdown structure and we’ll explain how we completed each step with explanations.

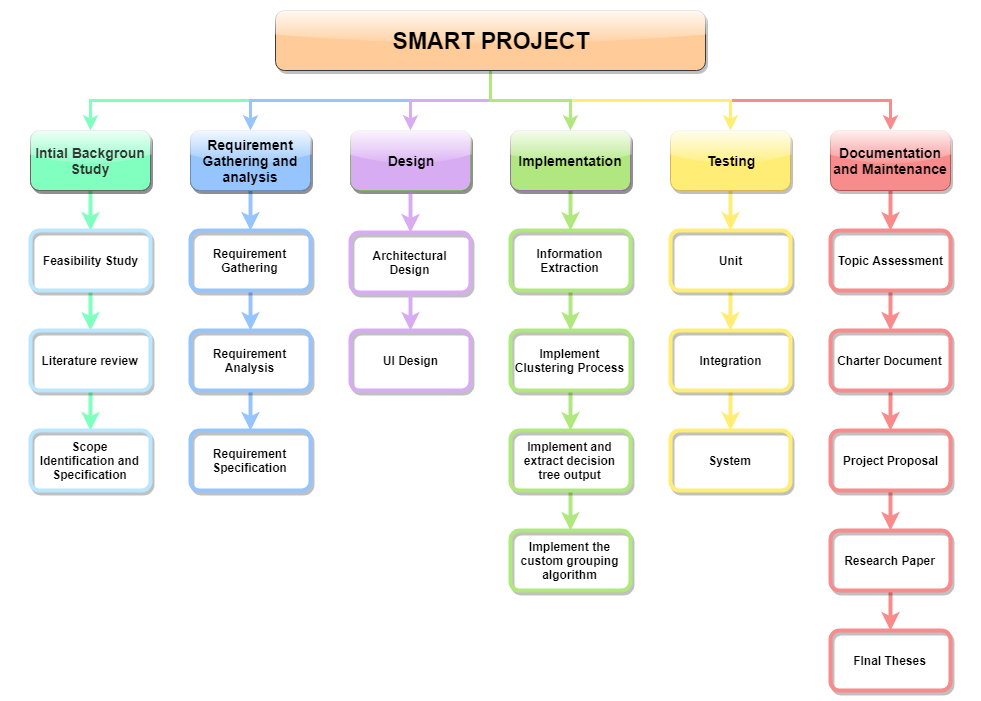


Figure 5: Breakdown Structure

Most of currently in use software solutions for project management are basically build for common use. But if we consider a university, there are more requirements to satisfy if we use a project management system. For an instance, Student group creation is one of the most required and costly task in a project management phase. Since the group formation is mostly managed in manual strategies, depending on the strategy time and the quality of the groups will change. If we consider managing the group creation quickly, the probability of low quality group formations is much higher than the quality focused group formation strategy.

Instead of managing student group creation manually, we are proposing a system that capable of create student groups systematically. Using skill based analysis and Average GPA analysis, system can generate student groups with higher quality and it increase the efficiency and effectiveness of the group formation process.

## **2.1 Feasibility Study**

* **Schedule feasibility**

Proposed system should finished in the given time period and according to the research scope, should be able to finish within the time period and from the time period we can arrange additional time period will be used for testing before demonstrations. According to the present timeline we’re few days ahead of the schedule.

* **Economy feasibility**

Since the whole solution in deployed in a virtual environment, only for the server cost will be need to be taken care of. The solution can be also deployed in the university servers but for the security reasons, better to be deployed in a different server.

* **Technical Feasibility**
  + Machine Learning knowledge

To implement grouping algorithms, we need to have a high level understanding on machine learning knowledge to identify algorithms and the logic resides within the source.

* Basic knowledge in APIs and MySQL Server

Need to have a basic knowledge in API implementations. Since the API will be developed using java, we need to have basic knowledge in java.

* UI and UX

Need need to have atleast minimum knowledge in Frontend Frameworks since we’re using React JS as the frontend development.

## **2.2 Requirement Gathering and Analysis**

The first phase in the SDLC is the requirement gathering process. In the process we’ve gathered current issues with the project management process in the SLIIT ITP module. We’ve contacted the lecture in charge of that module to gather information about the module process and the current issues having within the management.

* Survey Results

One of the major issue was the grouping mechanism. Usually there were conflicts between team members and the reason was the unbalanced group composition. Some groups having all the higher skill students and some of the groups having all the lower skilled students. So there is a need of a propper grouping mechanism for the project management.

## **2.3 Design**

The system architecture is shown in the figure 6.

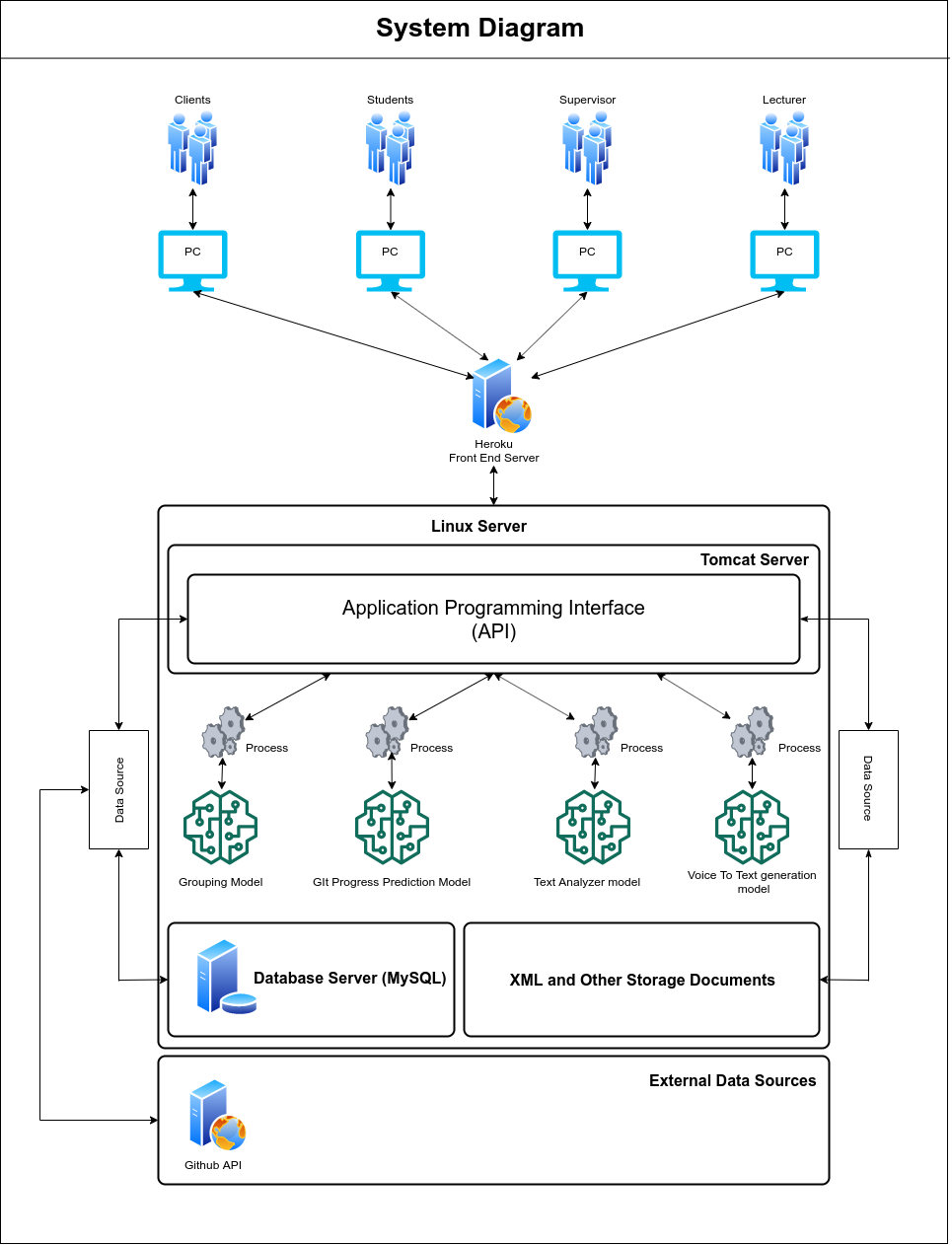


Figure 6 - System Architecture Diagram

According to the system diagram as shown in Figure 6, We’re hosting the backend server in a Linux environment and inside the Linux server there are 4 components. The system is mainly based on client-server architecture. System will extract data from the system database and from the GitHub API. In addition to that, system maintain local file storage to manage necessary outputs from the system. Below list represents the 4 components that exists inside the system.

* API running using the exported quarkus API executable or user can host the executable inside an apache server.
* MySQL Server
* Trained Models for decision Makings
* XML and Other Storage Documents

Heroku Server used to deploy the Frontend solution to interact with users. Frontend solution will be communicating with the users and the API.

## **2.4 Implementation**

### **2.4.1 Information Extraction**

This group formation methodology considers 7 factors to increase the fairness in the group composition.

* Develop Technology
* Develop For
* Programming Languages
* Frontend Frameworks
* Backend Frameworks
* Database Management Systems
* Version Controlling

Using a student survey form and the student profile, we can extract student skills and other necessary data to proceed with the group formation process. But in order to start the process, first we need to acquire each required components from each and every student that enrolled in a module. Since the all required information can be extracted from the system itself, it is easy to proceed to next phases.

### **2.4.2 Clustering and Decision Tree**

There are several strategies to group extracted student data. Widely used strategy is to create student groups randomly. But there are existing solutions that uses skill based analysis. For our system we’re using K-Means clustering algorithm to cluster student data into several groups. Basic concept is to create student lists considering different factors. Such as average GPA, development technology and programming languages.

Extracted student information firstly goes through the data clearing process and the generated student list will be go through the clustering process. Figure 7 represents the data cleaning code segment and Figure 8 represents the Clustering process code segment. Inside the clustering process students will be clustered into 3 clusters. Clustering process considers only the student GPA. Students will be clustered into High, Medium and Low GPA student groups. Then after the clustering process completed, system generates a csv file including the cluster number to identify the clusters in the next phase.



Figure 7: Data cleaning process code segment



Figure 8: Clustering process code segment

Figure 9 represents sample outputs of the clustering process. The issue of this process, system cannot identify a record if it is the High GPA record or a Medium GPA record. In order to identify the correct label for each record, we are using a Decision Tree algorithm.

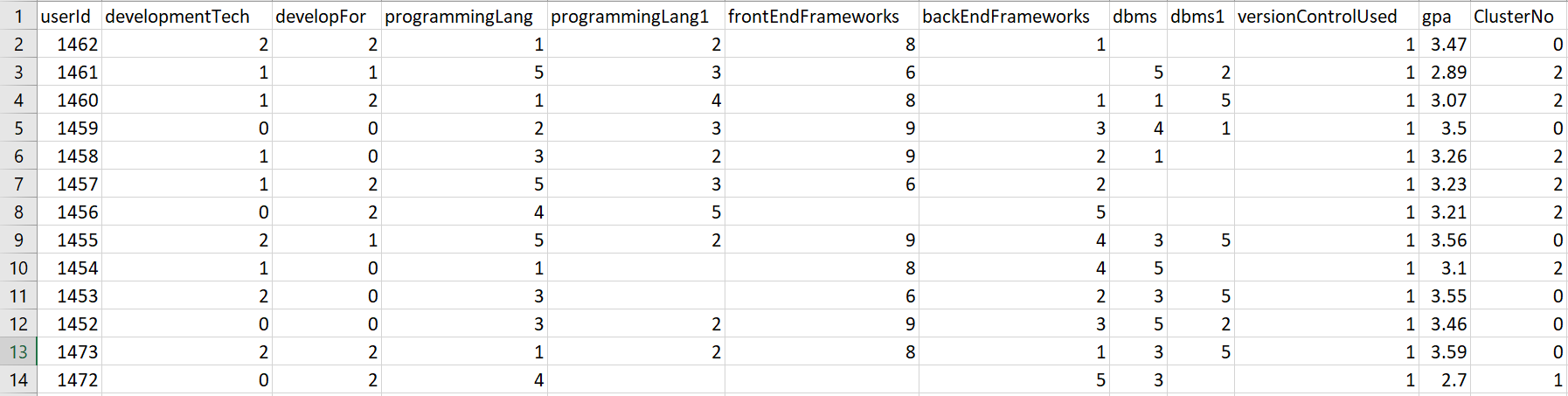


Figure 9: Sample clustering process results

After the clustering process, next phase is to label the clustered dataset. For this we’re using a decision tree algorithm. Figure 10 represents the output from the decision tree algorithm. Figure 11 represents the code segment used to construct the decision tree.

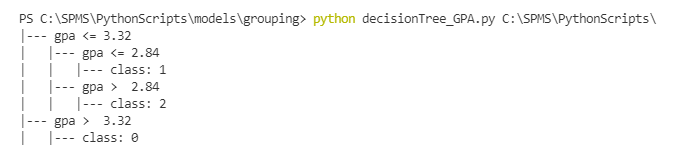


Figure 10: Decision tree algorithm output

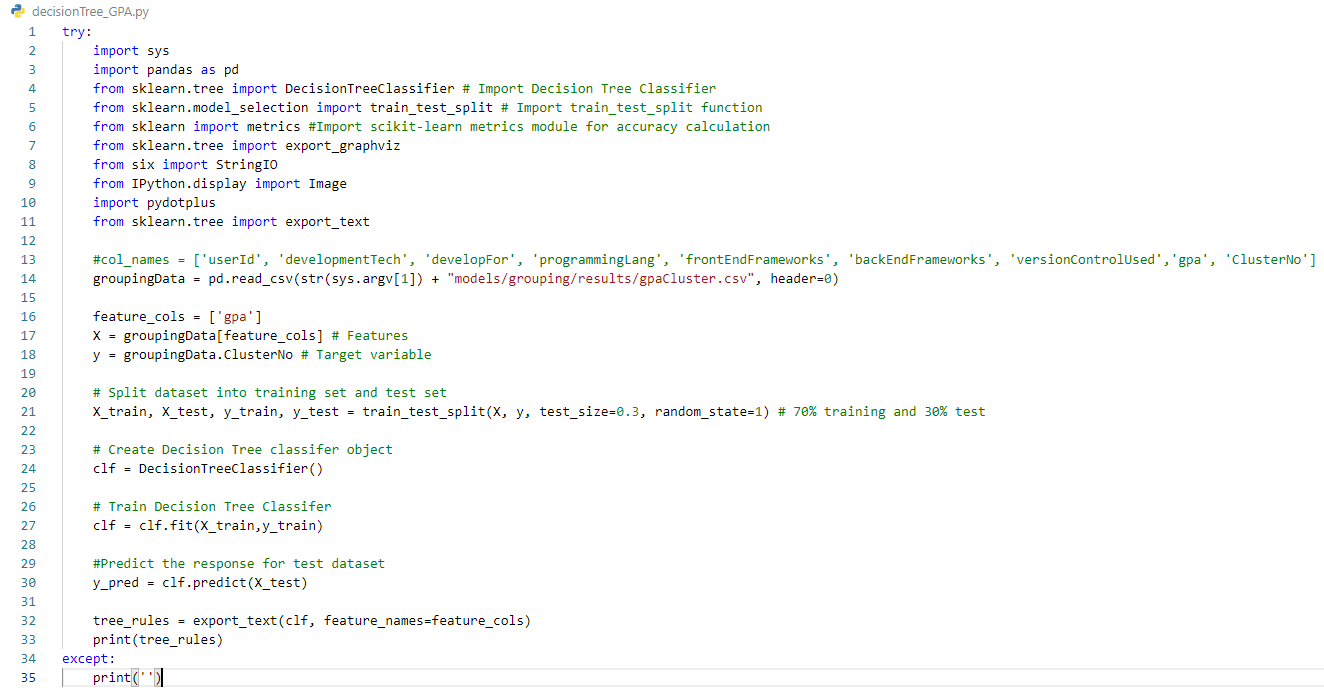


Figure 11: Decision Tree code segment

At this stage, all the necessary components of the grouping algorithm are acquired. Now we need to develop a custom algorithm that manage the group generation process. Since the group generation cannot be developed using previous group information, we need to develop an algorithm to decide which student is more suitable for a particular group. Next step is to build the custom algorithm to generate student groups.

### **2.4.3 Custom Grouping Algorithm**

Using above mentioned 7 factors, we can divide students into different sections. For an instance we can provide sections as Desktop Applications, Web Applications, Mobile Applications. But the theory is not to just group them separately, but to mix all sections and create a fair group composition where it is not just random but the group composition is same for all generated groups.

#### **2.4.3.1 Execute Preparation Programs**

Above mentioned clustering process and decision tree generation is executed inside this algorithm. In order to do that, we’re using a program called pyrunner. Using the pyrunner script we’re able to run python programs using a java program. After executing each python programs system will acquire the output from the decision tree. Figure 12 represents the code segment that relevant to above preparation process. After executing the python programs, system read the csv files that exported from the python programs and then it stores in the system as lists. Using identifyGPARules method, system stores the extracted GPA rules to the system.

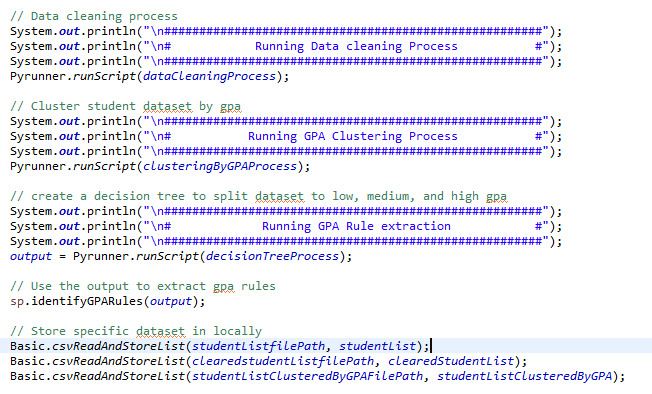


Figure 12: Preparation for the algorithm

Next we need to split extracted data into different groups. In order to do that, we’re creating a process flow that extract and store students into different lists for later usage. Next we need to store the average GPA of all students. The average GPA will be using to regularize the fluctuation of the average GPA among each student group.

#### **2.4.3.2 Group Formation Process**

After executing preprocessing components, Next process is to start group formation. In order to achieve fair and stable group composition, we create student groups with random students. This process is the majority uses to form student groups. Students will be randomly selected and system will generate student groups without considering any decision factor. This process helps the system to create a strong and fair group composition. Figure 13 represents the initial grouping process which completes the random group formation process.

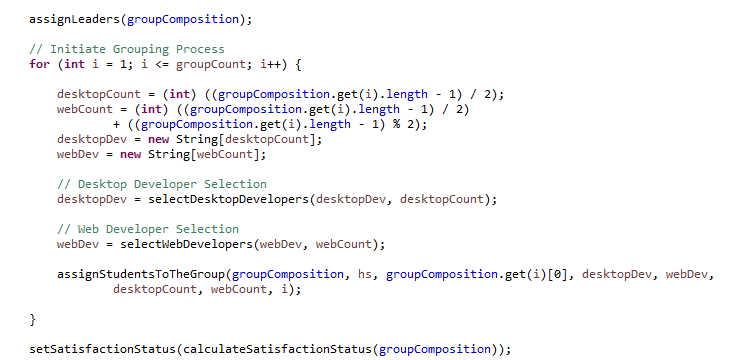


Figure 13: Initial Group Formation Process

If we inspect Figure 13, assignLeaders function will assign a leader for a particular group. Leader most of the times will be a desktop and web developer with higher average GPA. We’ve given priority to students with higher knowledge of programming languages for the leadership. As for the selectDesktopDevelopers function and selectWebDevelopers function iterates student lists that we’ve created earlier to select a student which suitable for the current group formation. Once the system select suitable students for the group assignStudentsToTheGroup function will return the generated student group and it will automatically add the student group into a list which contains all generated student groups.

Once the grouping process is finished, system will calculate satisfaction status of each group. Satisfaction status is calculated by the average GPA of a single group. If the average GPA is greater than the average GPA of all students, system record that group as a success. But the problem is that a single group average GPA cannot be higher than the average GPA of all students each time. In order to achieve 100% success rate, we must reduce the conditional value. Conditional value is the average GPA of all students.

At this stage the group formation algorithm is finished. But if we calculate the satisfaction rate of all generated groups, it is more likely to be lower than 100%. To increase the satisfaction rate of each group, we need to implement a group balancer.

#### **2.4.3.3 Group Balancing Algorithm**

Group balancer will iterate through satisfied groups and unsatisfied groups. System select an unsatisfied group and it will exchange lower GPA students with High or medium GPA students from the satisfied group. After the exchange system will calculate the satisfaction rate of each group. If the satisfied group becomes an unsatisfied group, the process will be roll backed and then system will try the exact process with another satisfied group. Figure 14 represents the group balancing process of the system. Until each group achieve 100% satisfaction rate, the iterative process will be continued. But after a limited time on iterations, the loop will be ended to avoid any infinite loop occurrence.



Figure 14: Group balancing process code segment

#### **2.4.3.4 Group Balance Iterator Process**

In order to achieve maximum effect on the generated groups, we need to iterate this process using different marginal GPA values. In order to that, we’ve created another process which is the Group Balance Iterator. In this process system will try to identify what is the most suitable Average GPA for the conditional value. As we mentioned above, System uses the average GPA of initial student list. In this process, that conditional value will be reduced by 0.01 at each iteration. Once it reaches the suitable value, system returns the student groups with the satisfaction rate of 100%. Figure 15 represents the group balance iteration code segment.

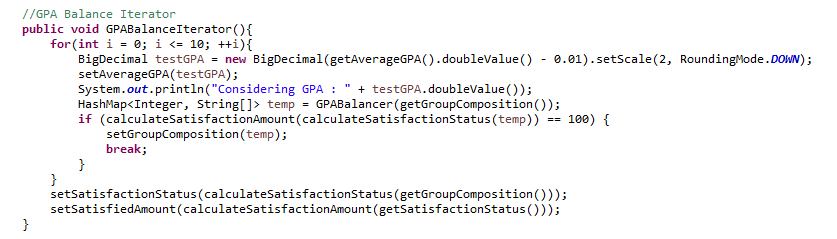


Figure 15: Group Balance Iterator code segment

As shown in the Figure 15, system will retrieve the current average GPA and decrease the value by 0.01. After the decrement, system will execute the GPABalancer algorithm to generate student groups using the new average GPA as the conditional value. Once the system completes the Group Balance Iterator process, system can output the generated student groups as a list. Figure 16 depicts a sample API output from the system. In the API output, we’re providing a group id for each group and a group name using the module code. In the student group list, each group has a computed average GPA, and also the satisfaction status. If something went wrong with the process, system indicates the process is malfunctioned and later it can be look for reasons. Providing these outputs will help to troubleshoot the system. If we consider the group composition of the below output, we’ve created a student group with 8 students. So the system will divide students fairly for each group in order to maintain the stability. In this instance, we can see a leader, three desktop developers and 4 web developers. We’ve given priority to web developers to have higher count of students in every group. We’ll discussing furthermore in the testing section about the final output.



Figure 16: Sample output from the API

## **2.5 Software Solution Development Components**

### **2.5.1 Web Application Development**

The end result of the project will be a web application to manage projects and the integrated capability to automate the grouping mechanism using student skills. We’ll be using React JS for the web application development. React is a JavaScript library for building user interfaces [7]. It is maintained by Facebook and a community of individual developers and companies.[[8]](https://en.wikipedia.org/wiki/React_(JavaScript_library)#cite_note-4)[[9]](https://en.wikipedia.org/wiki/React_(JavaScript_library)#cite_note-5)[[10]](https://en.wikipedia.org/wiki/React_(JavaScript_library)#cite_note-6). In development mode, we can simply run the web application using **npm run** command.

### **2.5.2 API development**

For the API development, we’re using java with maven quarkus[11]. Maven’s primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this goal, Maven deals with several areas of concern:

* Making the build process easy
* Providing a uniform build system
* Providing quality project information
* Encouraging better development practices

Maven aims to gather current principles for best practices development and make it easy to guide a project in that direction[12]. In development mode, we can execute the API by running **mvn compile quarkus:dev.** If users running the application inside a linux server, we’ve prepared a custom cammand script to execute application. In development mode, we can use **pmsDev b export** command to export the system to a jar file or else we can directly export and run the system by using **pmsDev b run** command. Figure 17 represents the application export process and Figure 18 represents the command to execute the exported jar file.

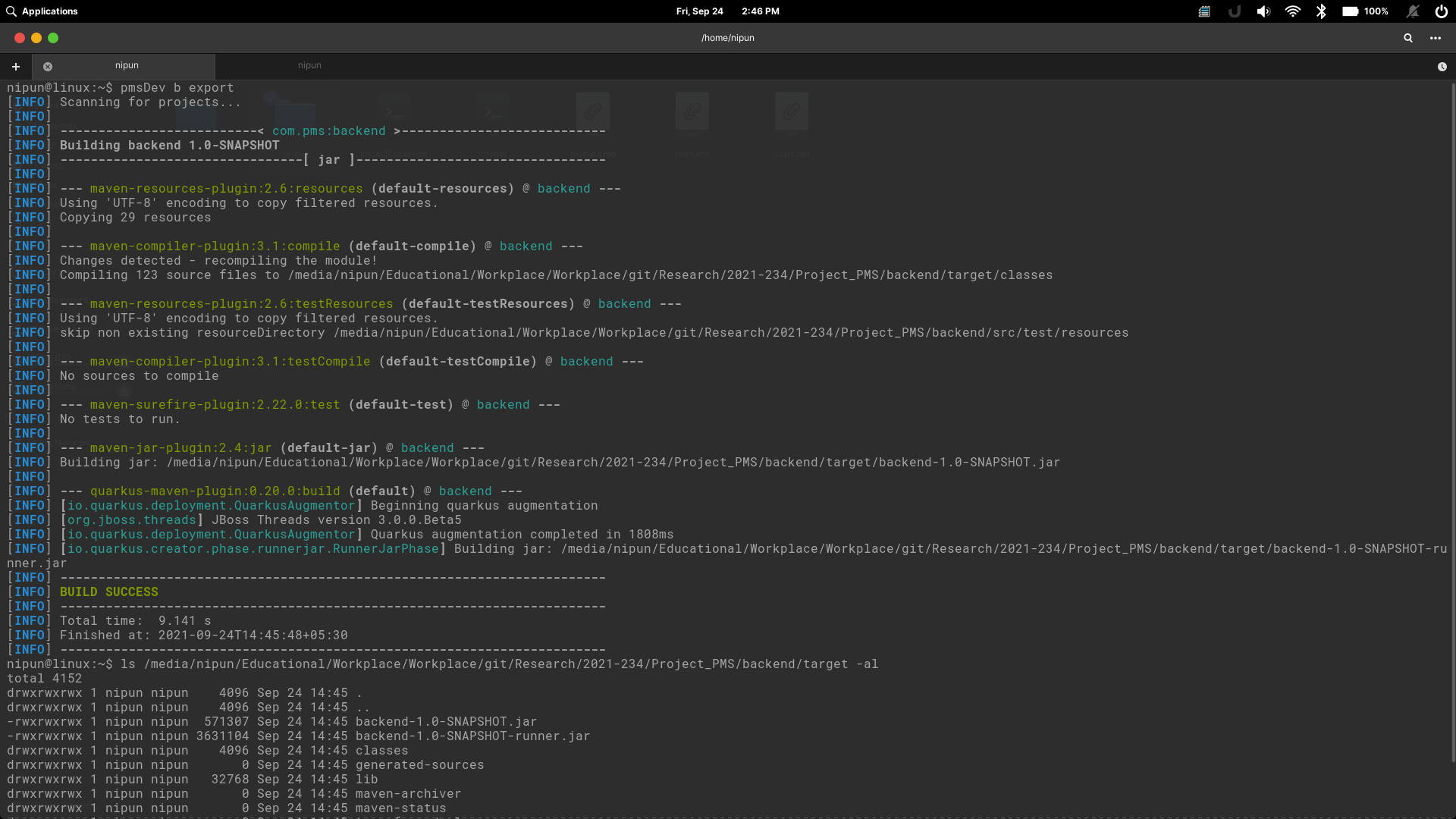


Figure 17: Export the system

### **2.5.3 Database Handling**

Using java JDBC driver makes it easy to manage MySQL database with the API. JDBC Driver is a software component that enables java application to interact with the database[13]. For the database querying and configuring database structures, we’ll be using MySQL workbench. This database structure will contain Tables, Procedures and functions to reduce the code repetition. In addition to that, we’re maintaining different files for different purposes. Such as, Database tables(models), Default values for tables, Procedures for the functionalities. We’ve created procedures only for write and update operations. Figure 19 represents the db script code segments for the database.

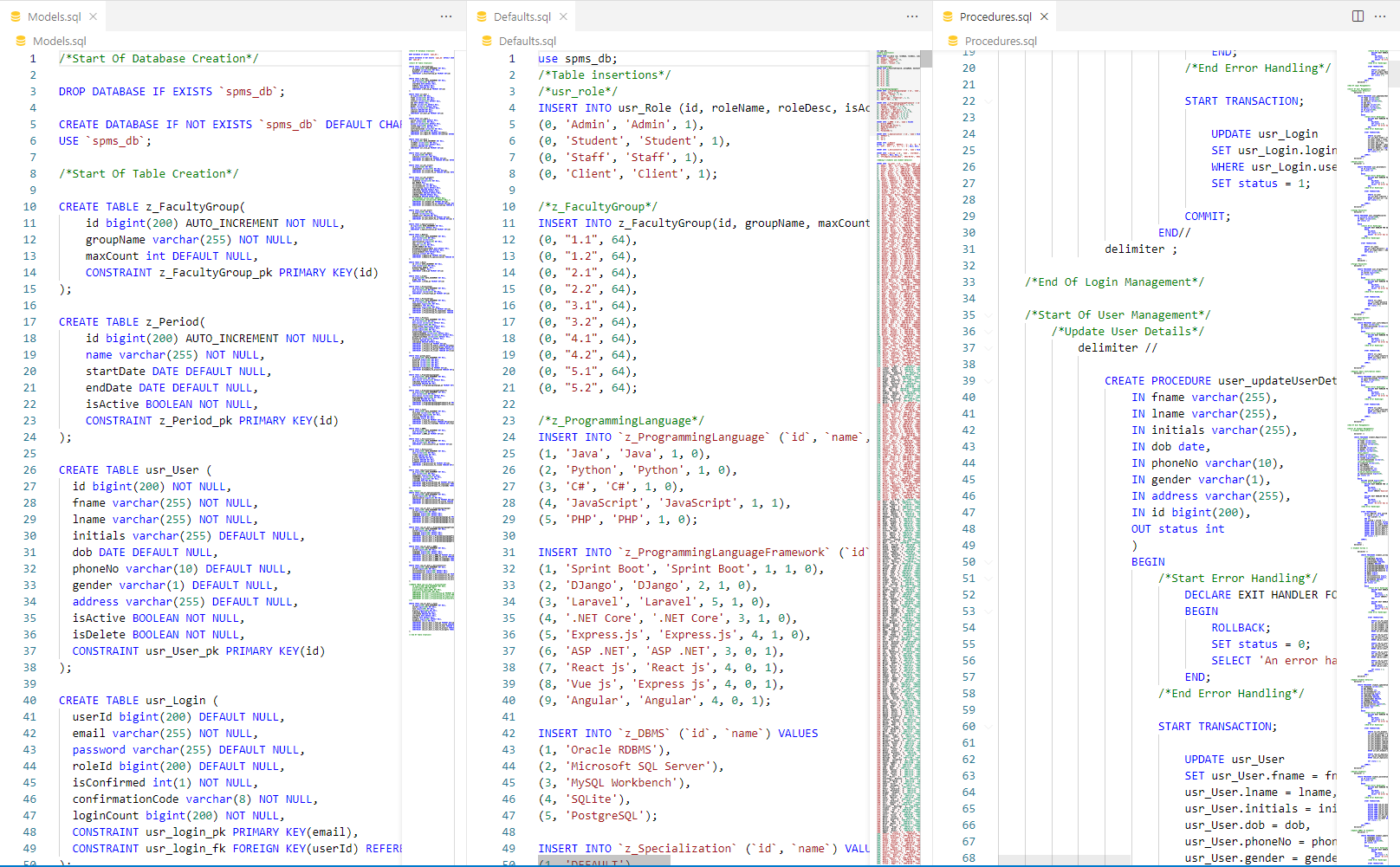


Figure 18: Database scripts

### **2.5.4 Testing**

**PostMan**

For the API testing we’re using Postman. Postman is a collaboration platform for API development. Postman's features simplify each step of building an API and streamline collaboration so the user can create better APIs faster[14].

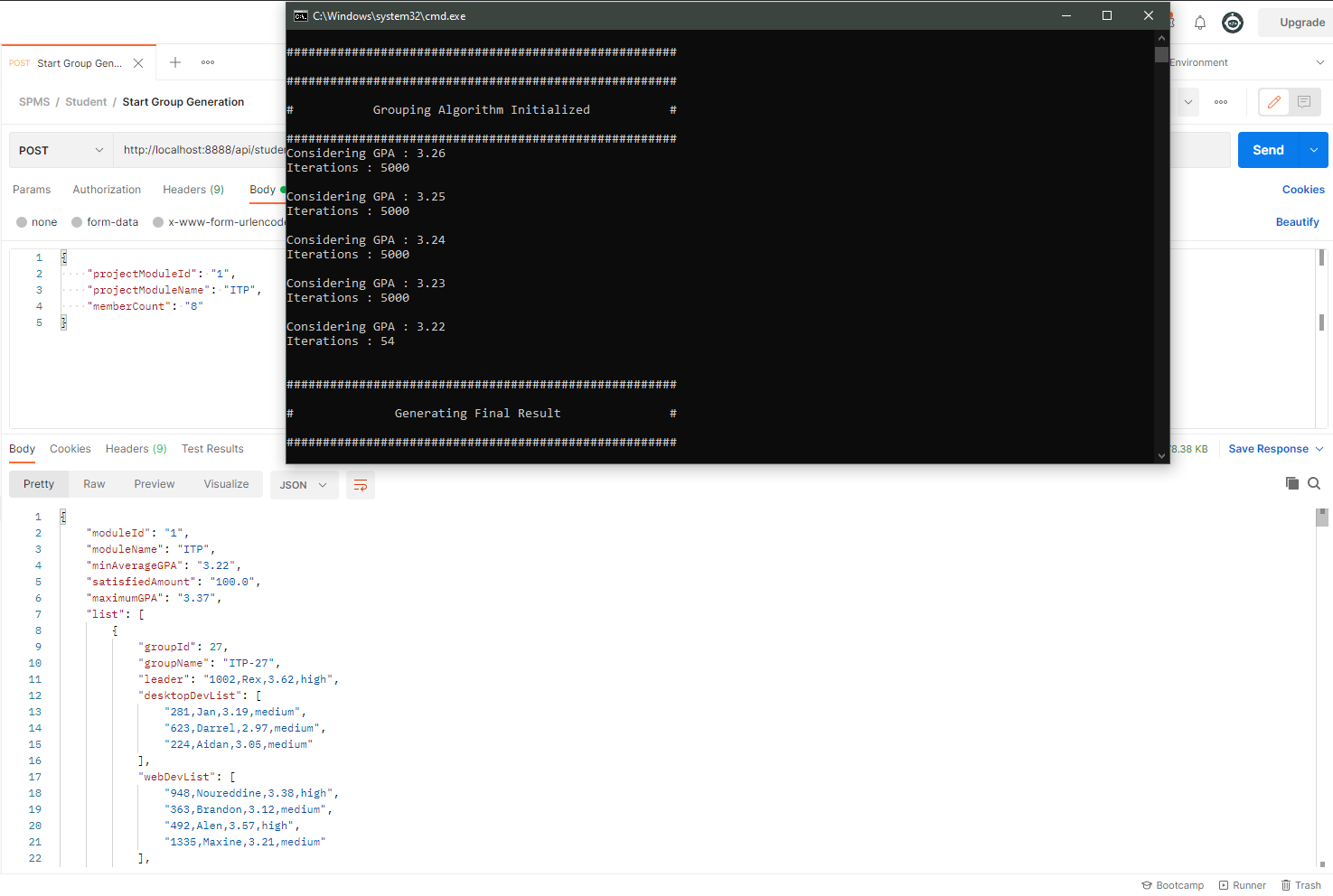


Figure 19: Sample postman output

In order to achieve maximum stabality in the system, we’ve continued the unit testing process until we integrate the system. Below tables are the test results of each unit testing phases.

Table 1: Test case 01

|  |  |
| --- | --- |
| Test case ID | 1 |
| Test Case description | Once user logged to the system increase the login count. |
| Pre- condition | User must login to the system. |
| Test procedure | Once user authenticated, increase the values of loginCount in the database. |
| Test Input Data | Email and Password |
| Expected Output | If the previous loginCount values is 5, next values should be 6. |
| Actual Output | System recorded loginCount as 6. |
| Result | Pass |

Table 2: Test case 02

|  |  |
| --- | --- |
| Test case ID | 2 |
| Test Case description | If someone registered to the system, a confirmation code should sent to them via the given email. |
| Pre- condition | User must give required details for the registration process. |
| Test procedure | Once the system verify the given data, system generates a random confirmation code and send it to the user via email. |
| Test Input Data | User details |
| Expected Output | System redirect user to the login page, but system shows “Verify the login first. Check your emails” when he tries to login. |
| Actual Output | System blocks the login process and shows Verify the login first. |
| Result | Pass |

Table 3: Test case 03

|  |  |
| --- | --- |
| Test case ID | 3 |
| Test Case description | Account verification via a verification link. |
| Pre- condition | User must locate the email and click the confirmation link. |
| Test procedure | Once the user click the link, the verification code will be sent to the server and the server will check the validity of the verification code. |
| Test Input Data | Verification Code. |
| Expected Output | Since the verification code is embedded with the link, Human errors should be zero and system should verify the login once the user click the link. System will show user “User account is activated” |
| Actual Output | Once the user click the link, system shows user account activated. |
| Result | Pass |

Table 4: Test case 04

|  |  |
| --- | --- |
| Test case ID | 4 |
| Test Case description | Student participate to the student survey, and system verify the user provided data. |
| Pre- condition | User must login to the system as a new user. If user is not a new user, the student can navigate to the student survey via the student profile in the system. |
| Test procedure | Once the student navigate into the student survey, student must fill every field before submitting the form. |
| Test Input Data | Student educational details and skill data. |
| Expected Output | If the student leave a field as empty when the student submitting the form, system shows a message “This field is empty” with the name of the field. |
| Actual Output | When student leave the programming languages field as empty, system showed as “This filed is empty: programming languages”. |
| Result | Pass |

Table 5: Test case 05

|  |  |
| --- | --- |
| Test case ID | 5 |
| Test Case description | When user try to create student groups, user must select a module. |
| Pre- condition | User must navigate to Group generation UI. |
| Test procedure | To initialize the group generation process, user should select a module from the dropdown. |
| Test Input Data | Module details. |
| Expected Output | If user select a module before running the group generation process, system will generate student groups without any issue. But if user doesn’t select a module before group generation, system shows “Please select a module.” |
| Actual Output | user selected the module before running the group generation process and system generated student groups without any issue. In second time, user didn’t select a module before group generation. system showed “Please select a module.” |
| Result | Pass |

Table 6: Test case 06

|  |  |
| --- | --- |
| Test case ID | 6 |
| Test Case description | When user try to create student groups, user must enter number of students per group. |
| Pre- condition | User must navigate to Group generation UI. |
| Test procedure | To initialize the group generation process, user should enter a number as the student count per group. |
| Test Input Data | Student count per group as a integer. |
| Expected Output | If user enter a number before running the group generation process, system will generate student groups without any issue. But if user doesn’t enter a number before group generation, system shows “Please enter a number for student count”. If user enter a negative value as a student count, then the system shows “Please enter a positive values as the student count”. |
| Actual Output | user entered 5 as student count before running the group generation process and system generated student groups without any issue. In second time, user didn’t enter a student count before group generation. system showed “Please enter a number for student count.”. When user entered a negative number as the student count, system showed “Please enter a positive values as the student count”. |
| Result | Pass |

### **2.5.5 Deployment**

**AWS**

AWS has significantly more services, and more features within those services, than any other cloud provider–from infrastructure technologies like compute, storage, and databases–to emerging technologies, such as machine learning and artificial intelligence, data lakes and analytics, and Internet of Things. This makes it faster, easier, and more cost effective [15].

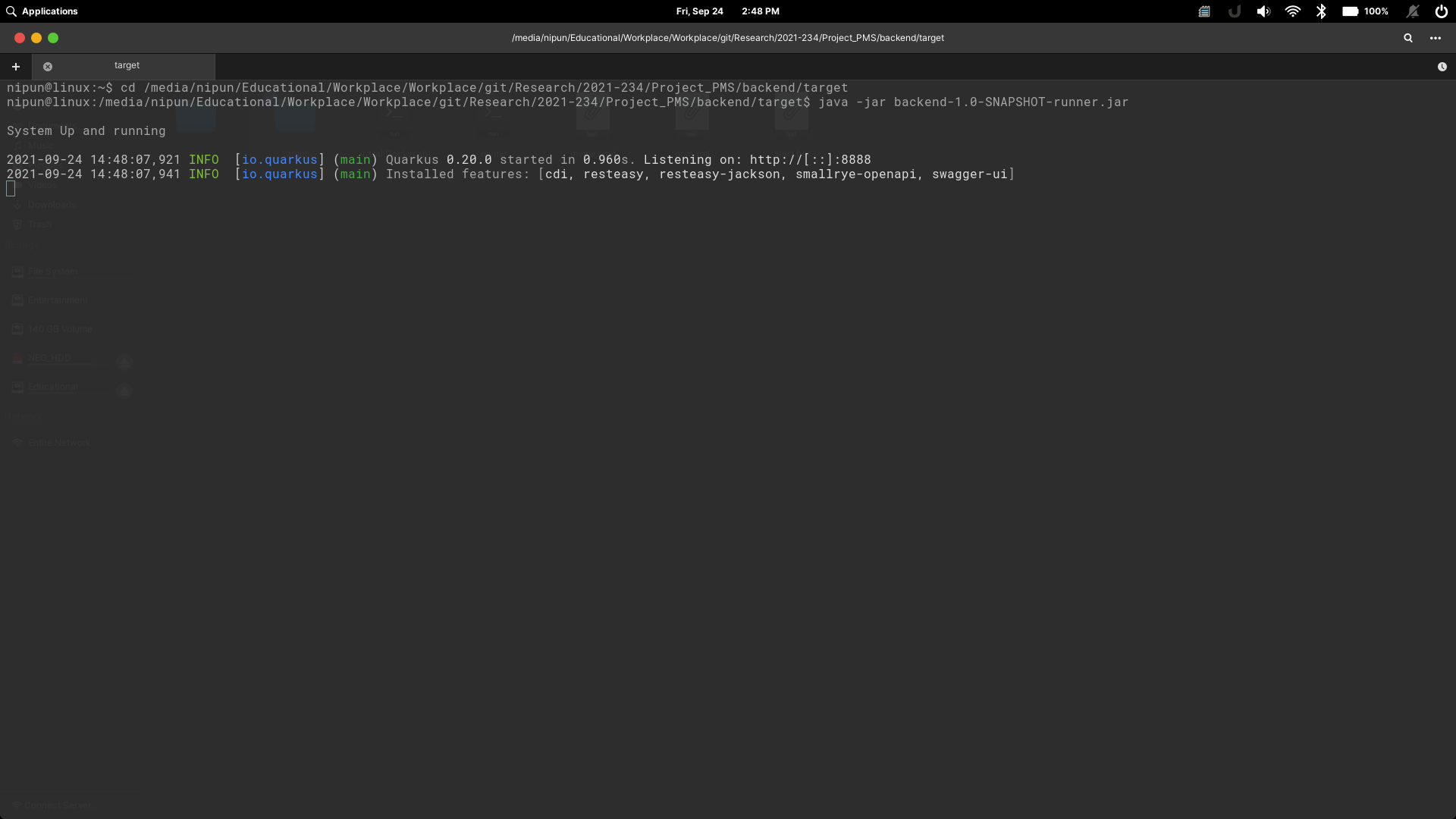


Figure 20: Execute the system

**Heroku**

Heroku is a cloud platform as a service supporting several programming languages. Heroku is used for container-based cloud Platform as a Service (PaaS). As we mentioned in the system overview, we deploy our frontend part in Heroku cloud platform. To do that we create a heroku account and initialie our project as a heroku GitHub Repository. Then we push our files to heroku repository. After that Heroku will automatically deploy our project to their servers. Finally, they are providing a URL to access our frontend.

# 3. RESULTS AND DISCUSSION

## **3.1 Results**

Once the system aquire student details and their skills, system was able to generate a student list with their skills. Complete system worked efficiently and interectively with student details. After aquiring student details, next step was to generate student groups with the student details. User need to navigate to the group generation UI in the system. Once the user enteres to the group generation UI, user need to select the module from the drop down and also the number of students per group. Then user can start the group generation process. It takes few minutes to generate groups since this is a iterative process. Once group generation is finished system shows the generated groups. User can decide from there, whether to continue with the generated groups or regenerate new groups.

Once user select Accept, system will store group data inside the database. After that students can login to the system and check for their student group and contact their team mates via client portal or using their emails. Figure 19 represents the final outputs from the group generation UI.

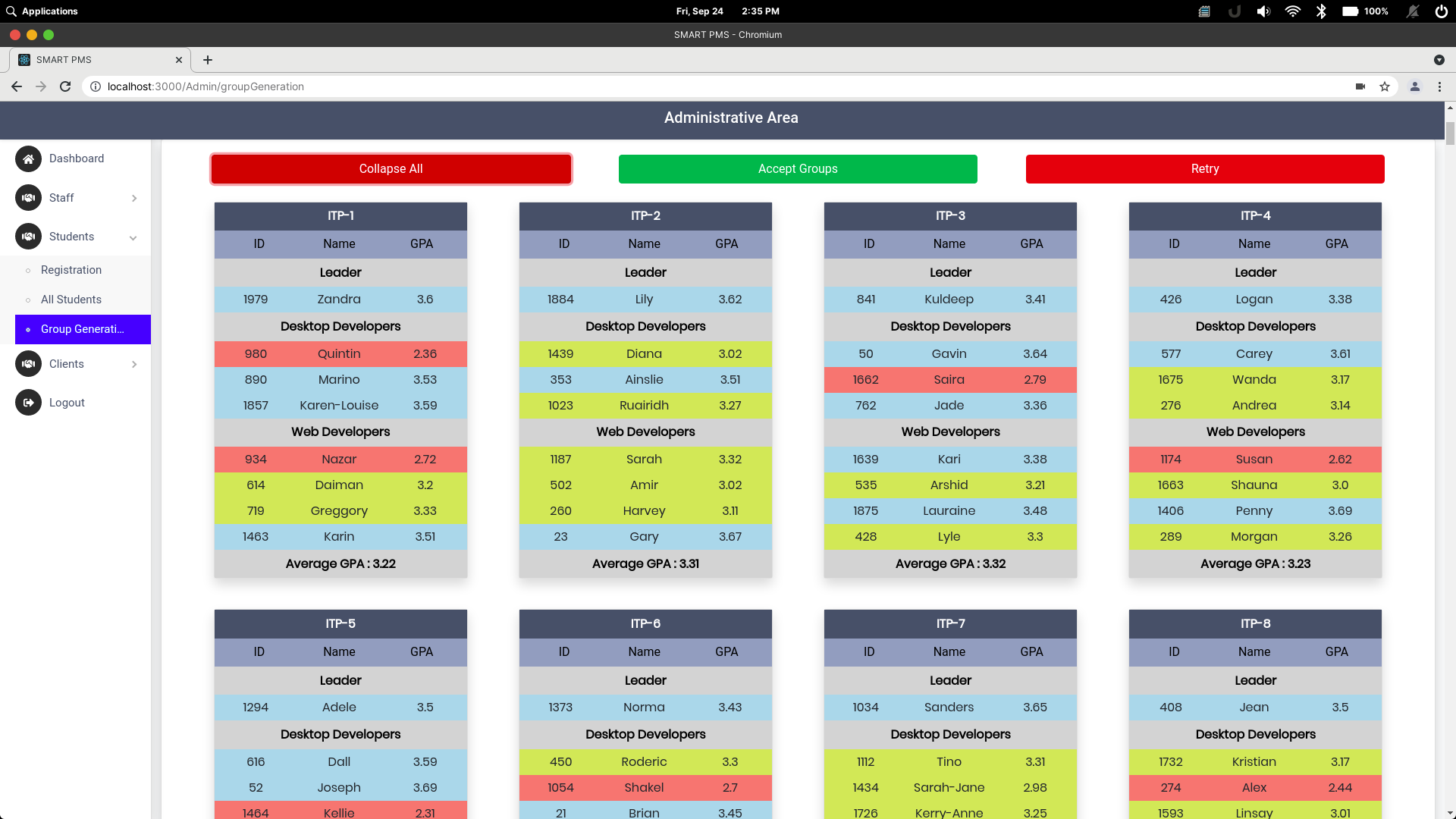


Figure 21: First group generation Output

In each iteration system will generate different outputs. Reason is to that we’re trying to create student groups more diverse than any manual strategy. So the process output can be much more different than the previous iteration. Since the iteration prcess checking average GPA of all generated groups in order to achieve the highest average GPA to increase the effectiveness of the system. Figure 20 represents the second iteration and group composition of both outputs are different from the other.

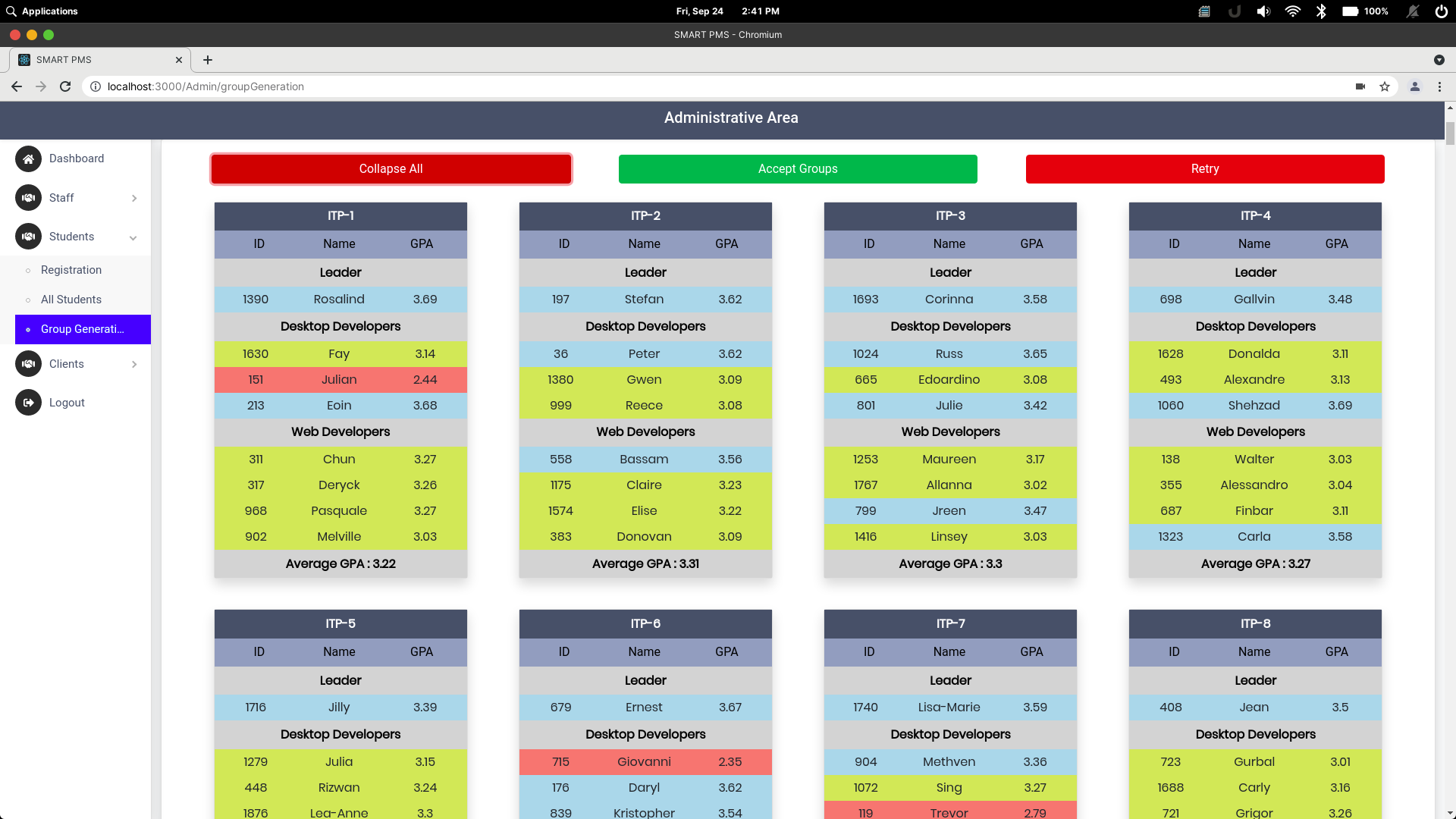


Figure 22: Second group generation output

## **3.2 Research Findings**

Primary goal of this research is to introduce a group generation environment to the project management systems. Most of project management systems are developed for common requirements, such as task management, maintain workflow and much more advanced procedures. Since most of existing project management systems doesn’t include this facility, it is valuable to have a group generation system in a project management system to a university environment.

During our initial research, we’ve found that most of universities currently using manual ways of group generation. Even from our experience, it is time consuming and exausting process for both management and students. So the goal was to develop a group generation functionality and merge the functionality with a newly created project management system where universities can manage the system by themselves.

After indentifying the problem, we’ve proposed a solution to develop a group generation algorithm. But in order to develop such algorithm, we had to decide which attributes and data we consider. So we did a requirement gathering and analysis. From what we’ve learned, we can easily develop and grouping algorithm using GPA of students. Since GPA is a main factor that universities consider to measure a student’s skill levels, we’ve decided to use the GPA as a contributing factor to the grouping algorithm. But using only GPA we can not decide student’s capability. So we’ve decided to extract student skill details from them.

In order to extract student’s skills, we’ve used a google survery. Using the survey, we were able to extact student skill data. This included various types of skills. Such as development technology, development media and programming languages. Then we figured, these attributes are more than enough to start development of the algorithm.

Compare to previous researches, main advantage of the system is, it is integrated with the project management system. By doing so, we can easily manage student progress and group progress by using the same system. Since the grouping algorithm combined with stabalizers, it is much more reliable, effective and efficient. Table 1 represents the result comparison between previous researches and other strategies.

Table 7: Comparison between previous systems

| Grouping Approach | Grouping Strength(Average GPA) | Group Count | Students per group |
| --- | --- | --- | --- |
| By Hand | 3.02 – 3.31 | 9 | 4 \* 5 + 3 \* 2 + 2 \* 2 |
| Proposed | 3.23 – 3.53 | 8 | 4 \* 6 + 3 \* 2 |
| Team-Maker | 2.9 – 3.46 | 8 | 4 \* 7 + 2 \* 1 |
| GroupEng | 3.11 – 3.27 | - | - |

The final solution is much more reliable to use and it is much more effective and efficient on group generation.

## **3.3 User Interfaces**

This chapter introduce the User Interfaces that relevant to Group Formation Process.

Figure 23 depicts the student survey user interface. It is used to collect student skill data, Such as development skills, development criteria, programming languages and programming frameworks. Using these skill data, we can generate student groups more effectively. This UI is shown, when user login to the system for the first time. If user is not a new user, they can locate the survey in their profile.

This process is mandatory for students before go through other sections. Since the grouping process is depending on the student skills, students must go through this survey in order to included in a student group. If a student did’t participate in the student survey, they will be out of the grouping process.

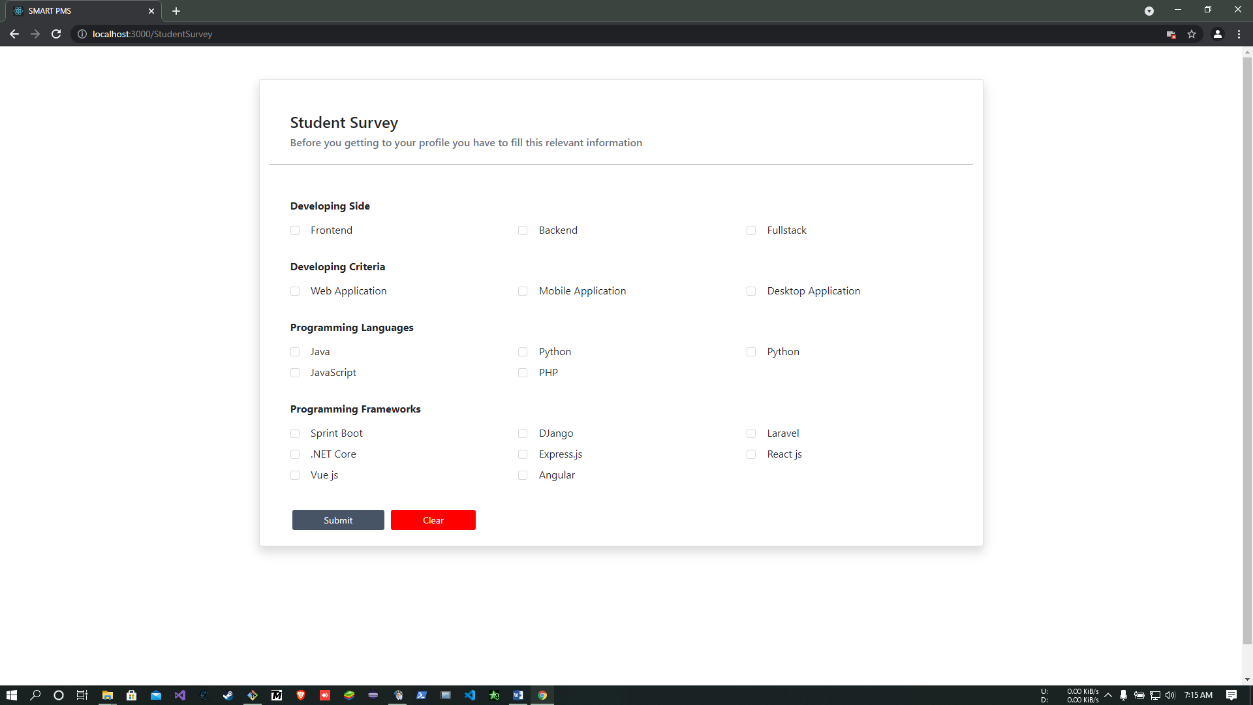


Figure 23: Student Survey

Figure 24 represents the student group generation UI. To navigate to this UI, user must be an administrator or a supervisor. Other users cannot access this UI without authorization. Users can generate student groups for any registered module. The Select project module dropdown contains all modules that registered in the system. If the relevant module is not included in the dropdown, user must contact administrator and add the module. Once user select a module, other fields will be automatically filled using database data.

Next we have the Member count. Member count is the student count per group. This count should be greater than three, Since the grouping process creates 3 major student group types. First one is the leader. Every group must have a leader. Second type is the desktop developers and the third type is the web developers. Once user enter a member count, system generation can be started.

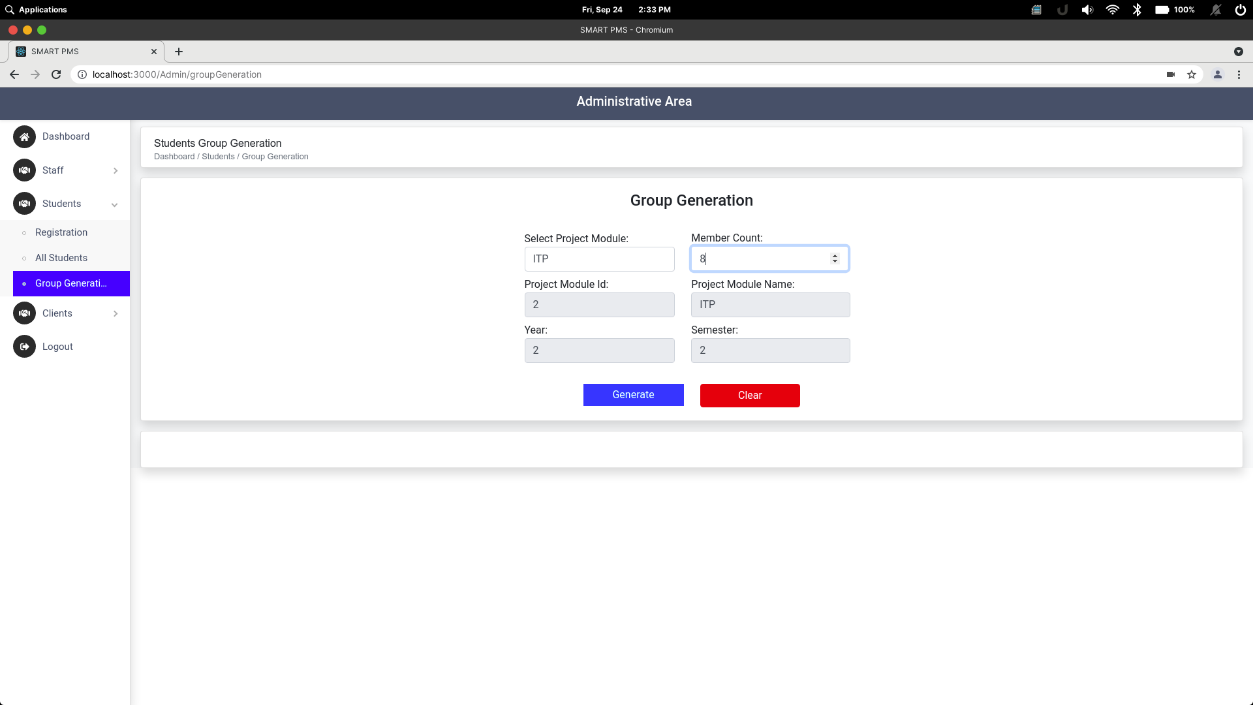


Figure 24: Student Group Generation UI

## **3.4 Discussion**

Most of project management system are for common uses. When we consider project management in universities, they have different requirements in addition to common requirements. Since most universities uses these common project management solutions, some of required features are not available to them. For this instance, we’re considering the student grouping feature. Most of universities are still using the traditional methods, such as random group generation or group generations following a registration student list. Using these methods will create unbalanced student groups.

When universities give the freedom to choose their team mates, most of the times groups that created by students are homogenous. Reason for that is students always try to be in a student group with higher skill levels. This process leads to create student groups with higher skills and the other groups becomes lower skilled groups. We can observe these unbalanced situations in most of the times.

The solution is to create a systematic solution which creates heterogeneous student groups. Heterogeneous groups contain students with different types of skills and skill levels. This leads to create more balanced and fair student groups. Studies shows that heterogeneous groups are much more effective rather than homogenous groups. These studies we’ve discussed in previous sections. So the solution is to create a program which manage the grouping process.

For this we’ve considered 7 factors. We’ve given priority to the GPA, since the GPA is a grading factor used in any university. We’ve used machine learning approach instead of a constant value to determine skill level. For an instance, we could use a constant value to check if the student is a highest skilled student or not. For an example, if we get the highest skilled level student must have 3.0 as the GPA, whole process considers this single value to determine student’s skill level. Instead of using that method, We’re using machine learning to determine the suitable value for each marginal value. Marginal values are the values that used to check which level is student in. Such as High level students, Medium level students and Low level students.

Using k-means and decision tree algorithms, we were able to calculate this GPA value dynamically. Which means this GPA values is depending on each student in the module. So the GPA value is not a constant value to decide the student’s skill level. Instead it’s decided by considering all the students in the module. Figure 25 represents the decision tree and the marginal values that returned by the algorithms.

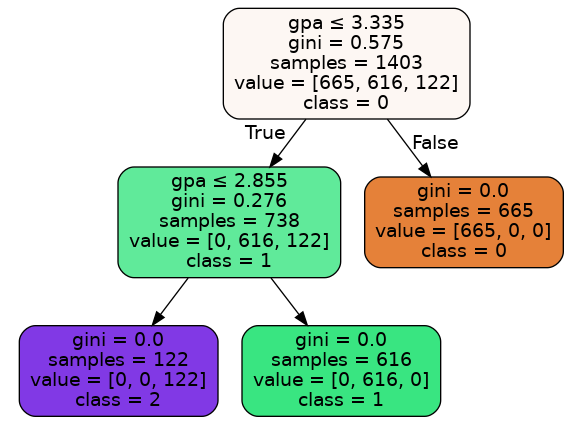


Figure 25: Decision tree graphical output

As you can see in the figure 25, the sample student list returned the marginal value for the highest level student as 3.335 and the student count is 665. Marginal value for the medium level student is 2.855 and the student count is 616, And students who lower than 2.855 are considered as low level students.

Using this method, we were able to automate the system to dynamically allocate the GPA marginal values. After this process is the grouping process. For that we’ve created a custom algorithm to identify each type of student and create heterogeneous groups instead of homogenous student groups.

Previous research solutions didn’t use this GPA dynamic allocation method to identify student groups. Instead of that, they used their field of studies, weighted calculations using given values for each field of study. This dynamic value allocation method is much more reliable and much more effective than the previous methods.

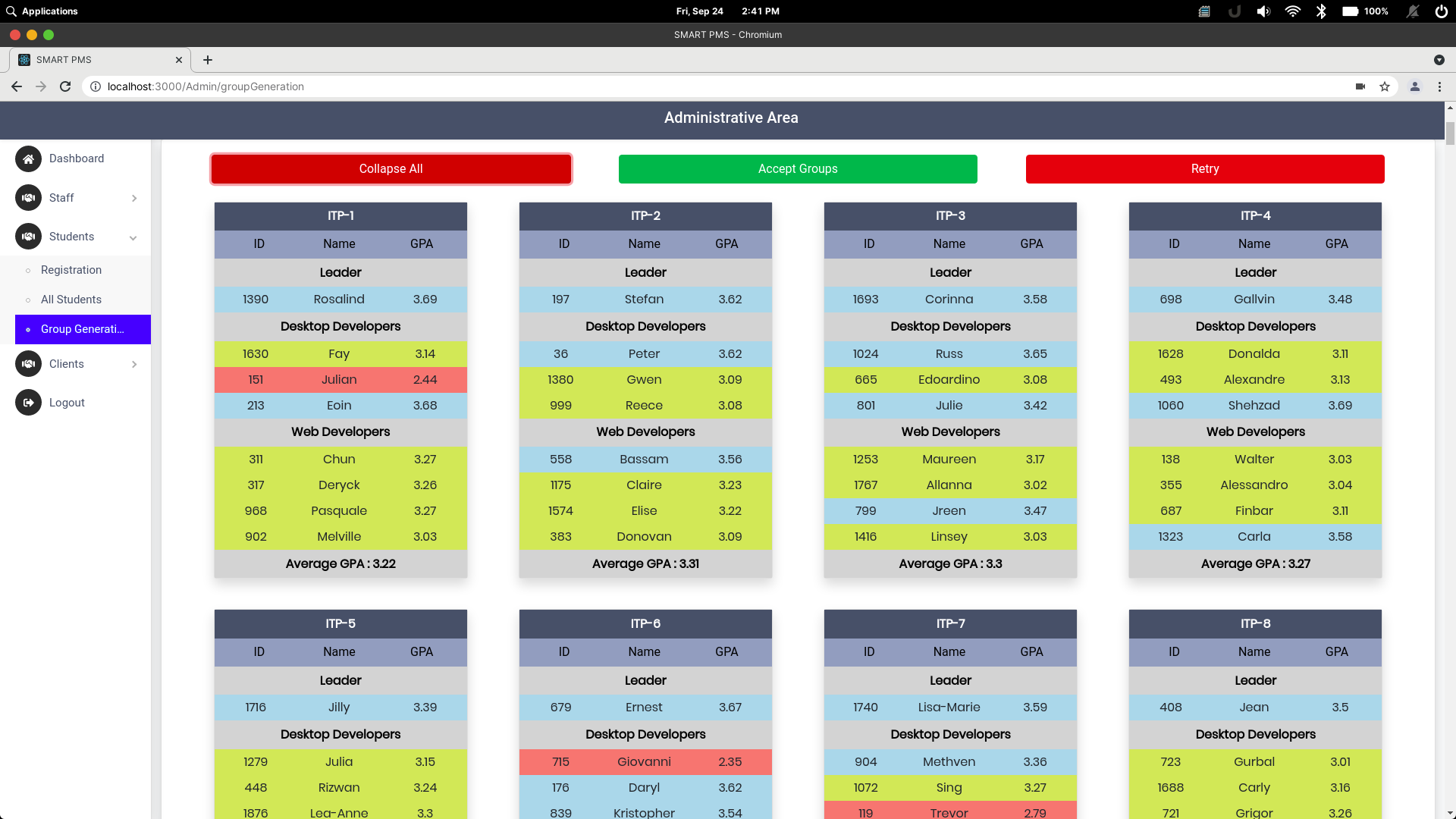
From the final results of the algorithm, we can clearly see that average GPA of each student groups are closer and not very diverse than other groups. This means the grouping process is successfully regulated those previously mentioned unbalanced situations. There weren’t any underpowered or overpowered groups. Instead of each and every group had a combination of each type of students. As we can see in the figure 26, sample group has student from each type. Blue colored student record is the highest level student, yellow colored student record is the medium level student, and the red colored student record is the low level student.

Figure 26: Sample system output

So instead of creating students with the same level, this solution creates a student groups with each type of students. Using this method, we can avoid such unbalanced situations and create much more effective groups.

# 4. SUMMARY OF THIS RESEACH CONTRIBUTION

Table 8: Research Contributions

|  |  |  |
| --- | --- | --- |
| Member | Component | Task |
| Amarasekara T.N.E. | Automate student group generation using student skills and machine learning approach. | * Collect data to initiate algorithm development. * Create a solution to collect student skills and data interactively. * Use machine learning approach to identify connection between data types. * Create an algorithm to generate student groups. * Create a solution to view outputs and give an overview of the overall groups. |

# 5. CONCLUSION

With the proposed project management system was able to manage the workflow of a common project in a university environment. Combined with major requirements for a university projects, system was able successfully execute a complete project management process workflow. Starting with group generation, GitHub contribution predictor predicts future contributions from the students. Furthermore, with the meeting documentation process, students can check previous meeting details. And finally the peer review was able to contribute for the evaluation phase of a project management process. Since the proposed project management system was able to cover major steps of project management. Even so the system is functional as proposed, system can be further improving with new technologies and features.

Most of project management systems that currently in use are mostly for common purposes. Developing a configurable project management system is much more valuable for institutes since they have different variations of requirements. By developing such system is much more effective for an educational environment.

Since the grouping process is required facility from universities, Integrating a grouping solution to a project management system is much more effective and efficient way. Instead of using multiple solutions to do multiple tasks, we’ve integrated these required features into one system. By integrating the grouping solution is very useful when it is communicating with the other components. For example, the student grouping process is the first and main task of any project management phase. Without a forming a group, students cannot initiate a project.

The advantage of the developed grouping solution is, it is dynamically changes depending on the students and their skills. Instead of creating groups without any considerations, This system is considering student’s skill levels. Results from our research is quite promissing when we consider the grouping solution. By comparing with existing solutions, Our solution is much more dynamic and much more calculated than the previous research works. We’ve come to the conclusion that this constructed grouping solution can be used in any university since the grouping process is only depending on student skills and their knowledge in the programming field.

# 6. REFERENCES

[1] Chen Di and Jiang Bei-zhan, “Some Problems in Teaching ‘Educational Technology in Project Teaching’”, Journal of e-Education Research,pp.63-65 2007.

[2] Dimiduk, T. G., & Dimiduk, K. C. (2011, December). Effectively assign student groups by applying multiple user-prioritized academic and demographic factors using a new open source program, GroupEng. In 2011 WEPAN Nat. Conf. Advancing Women: Transforming Eng. Educ (pp. 1-12).

[3] Chirag Mehta, & Tamara Swedberg, Team Maker, 2000. [online]. Available: <https://chir.ag/projects/team-maker/>. [Accessed: 19-Feb- 2021]

[4] Brant Gurganus, Team Maker, 2000. [online]. Available: <https://sourceforge.net/projects/team-maker/>. [Accessed: 19-Feb- 2021]

[5] Z. Wen-qiang and W. Ben-ting, "Project practical teaching grouping strategy based on fuzzy clustering algorithm," 2010 5th International Conference on Computer Science & Education, Hefei, China, 2010, pp. 678-681, doi: 10.1109/ICCSE.2010.5593662.

[6] Livermore, J. A. (2007, March). Factors that impact implementing an agile software development methodology. In Proceedings 2007 IEEE SoutheastCon (pp. 82-86). IEEE.

[7] ReactJS. (2021). A JavaScript library for building user interfaces. [online]. Available: <https://reactjs.org/> [Accessed 21-Feb-2021]

[8] Paul Krill, “React: Making faster, smoother UIs for data-driven Web apps”, May 15, 2014. [online]. Available: <https://www.infoworld.com/article/2608181/react--making-faster--smoother-uis-for-data-driven-web-apps.html>. [Accessed: 20-Feb-2021]

[9] Hemel, Zef, "Facebook's React JavaScript User Interfaces Library Receives Mixed Reviews", Jun 03, 2013. [online]. Available: <https://www.infoq.com/news/2013/06/facebook-react/> .[Accessed: 20-Feb-2021]

[10] Dawson, Chris, “JavaScript’s History and How it Led To ReactJS”, July 25, 2014. [online]. Available: <https://thenewstack.io/javascripts-history-and-how-it-led-to-reactjs/> .[Accessed: 20-Feb-2021]

[11] Quarkus. (2021). A Kubernetes Native Java stack tailored for OpenJDK HotSpot and GraalVM, crafted from the best of breed Java libraries and standards. [online]. Available: <https://quarkus.io/>. [Accessed: 21-Feb-2021]

[12] The Apache Software Foundation. (2021). What is maven. [Online]. Available: <https://maven.apache.org/what-is-maven.html>. [Accessed: 21-Feb-2021]

[13] Javatpoint. (2018). JDBC Driver. [online]. Available: <https://www.javatpoint.com/jdbc-driver>. [Accessed: 21-Feb-2021]

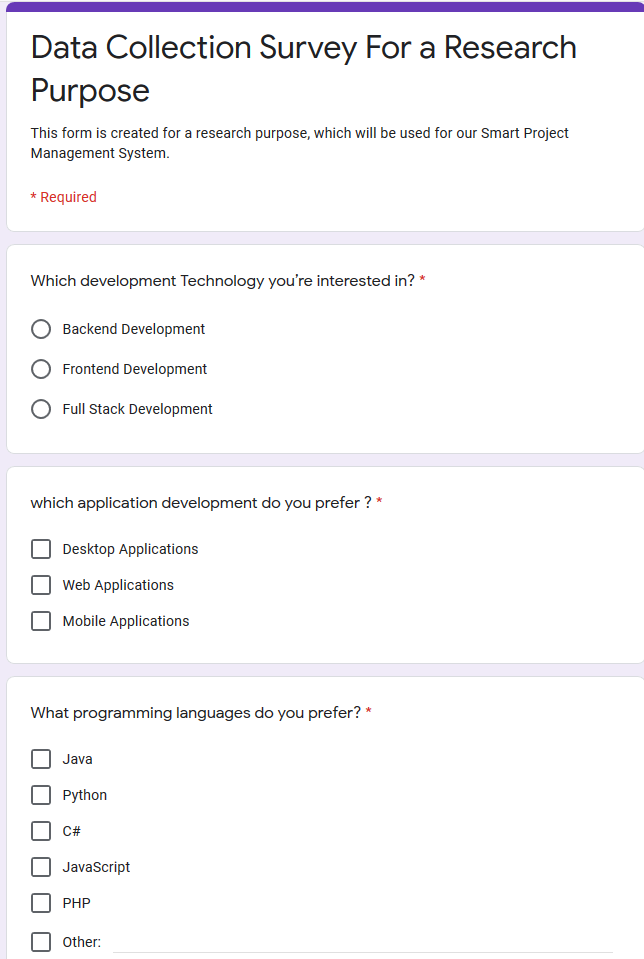
[14] Postman. (2021). The Collaboration Platform for API Development. [online]. Available: <https://www.postman.com/>. [Accessed: 21-Feb-2021]

[15] Amazon Web Services. (2021). Cloud computing with AWS. [online]. Available: <https://aws.amazon.com/what-is-aws/?nc1=f_cc>. [Accessed: 21-Feb-2021]

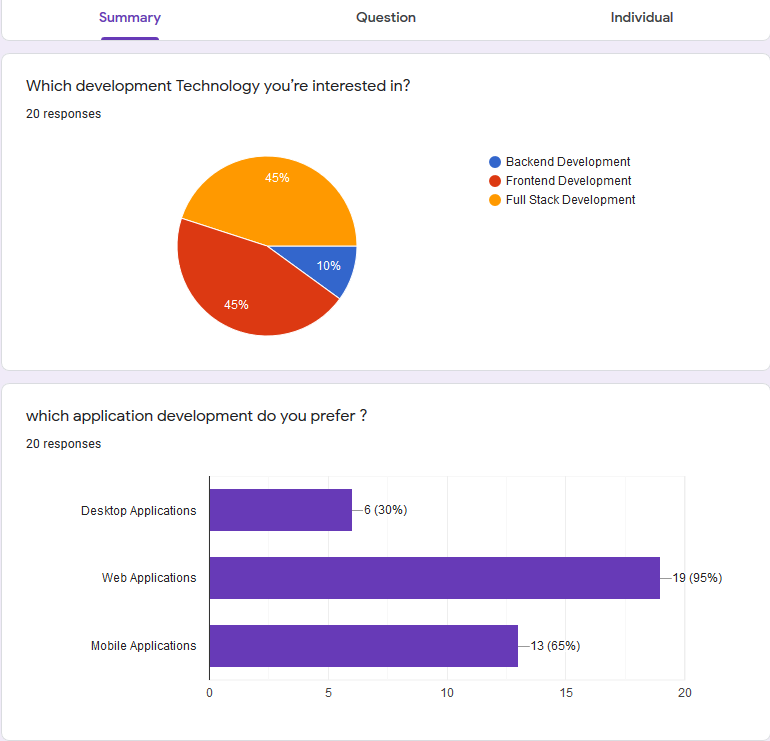
[16] Srba, I., & Bielikova, M. (2014). Dynamic group formation as an approach to collaborative learning support. IEEE transactions on learning technologies, 8(2), 173-186.

# 7. APPENDICES

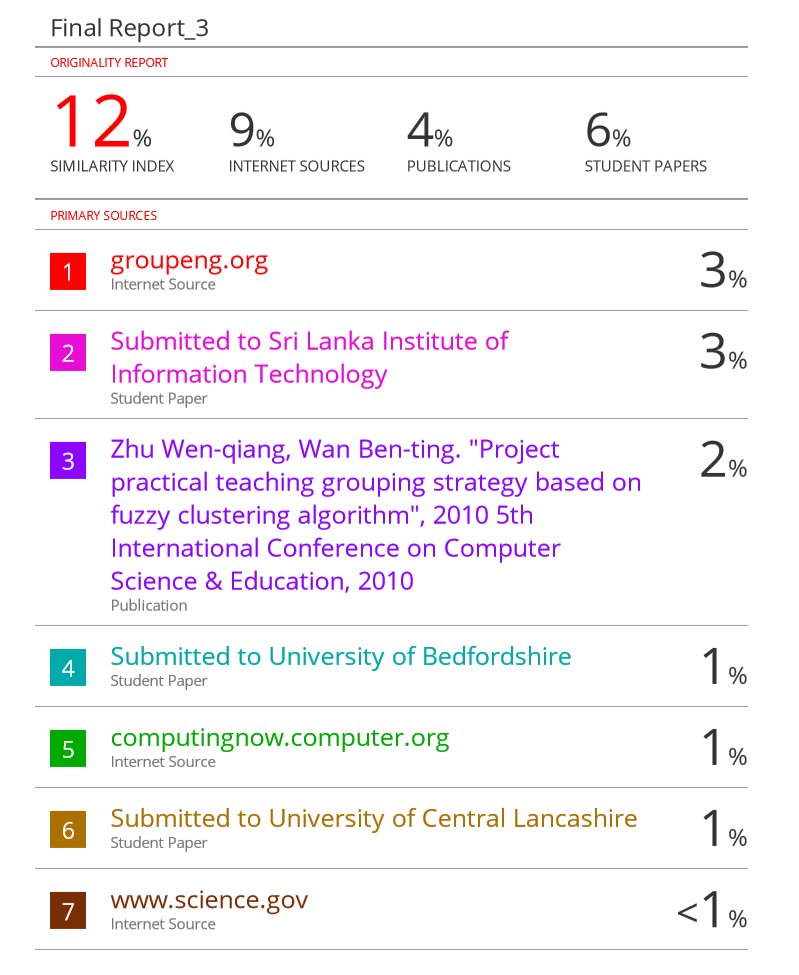
## Appendix – A: Sample questionnaire



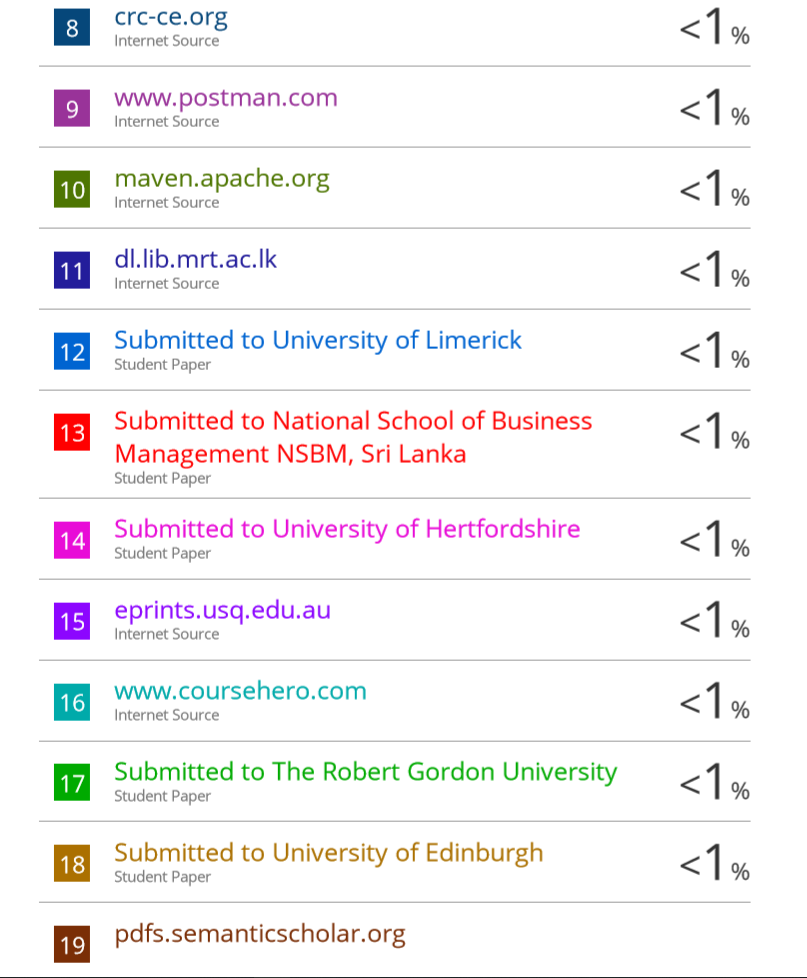
## Appendix – B: Sample questionnaire response



## Appendix - C: Plagiarism Score 01



## Appendix - D: Plagiarism Score 02



## Appendix - E: Plagiarism Score 03

