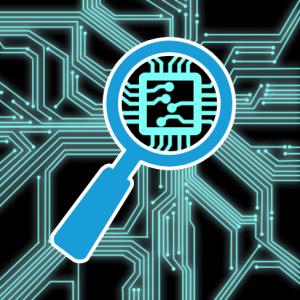
[25th April, 2018]



FINAL REPORT

[Easy PC - Customisable PC Build Mobile Application]

Group members:

Tung Pak Hei (20028212)

Ng Ho Cheung John Bosco (20026027)

Miao Wei Chu (20021695)

Mo Yu Hin (20026064)

II. Content Page

I. Cover Page…………………………………………………………………………1

II. Content Page………………………………………………………………...……2

III. Summary………………………………………………………………………….. 3

IV. Introduction.………………………………………………………………….…. 4

V. Objectives/Goals……………………………………………………………..... 5

VI. Problems………………………………………………………………………….. 6-7

VII. Solution………………………………………………………………………….... 8-15

VIII. Schedule………………………………………………………………………….... 16

IX. Resource Allocation .………………………………………………………….17-19

X. Appendix………………………………………………………………………….. 20

III. Summary

This mobile application that we have developed is primarily used for automatically generating a PC build with various levels of customisation, catering to the customers’ needs. This report covers the system requirements, specifications and testing methodologies of our application in detail, including the problems we encountered and how we tackled them.

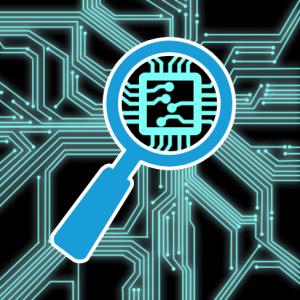
We considered the effectiveness and the efficiency of the database and algorithm to be our main concern and we came up with ideas to fulfill the customers’ needs precisely, for instance, an algorithm based on weighted score to split the budget into smaller portions for each component, we even plan to utilise a questionnaire in hopes to let our algorithm know clearly of customers’ desires, including the budget the customer is willing to spend, the brand or specifications on certain components, which is further discussed in latter parts of the proposal.

The application is fully built in 30 weeks, which was within schedule, details of time spent are as follows; 4 weeks for analysis, 12 weeks for development, 6 weeks for testing, and 4 weeks for an overall review. Every group member has contributed roughly 300 hours to a collectively 1200 hours to fulfill and tightly align with our requirements as listed in our proposal.

IV. Introduction

We are a group of four Information Technology enthusiasts that are currently acquiring a Higher Diploma in Information Technology in HKUSPACE. We have a strong passion in software development and dedication to strive for success.

“EasyPC” is an Android mobile application which helps users to generate their computer builds automatically by utilizing an algorithm built in the software. The name “EasyPC” is derived phonetically from the colloquial saying - “Easy peasy”, while PC is an acronym for Personal Computer. The aforementioned phrase means something that can be done effortlessly which coincides with our ultimate goal developing this software, which is to help end users to build their own PC with ease.



The logo is made of a magnifying glass looking into a processing chip, the magnifying glass resembles our search function and the processing chip represents our algorithm.

This idea was brought up because many of our friends who are not familiar with computers always complain about the device they bought from the chain brand companies was worse than expected and would like to build their own desktop personal computers (PCs) but unfortunately they do not have the knowledge to do so. Moreover, we see a lack of mobile applications which would cater to users’ desires therefore we thought this would be a great opportunity to provide a platform for these people to build their own PC according to their preferences. After searching the Android application marketplace, Google Play, applications of this sort are nowhere to be seen, this further shows that there is indeed a need for this application to be on the Android platform.

From our own observation, the main source of customers for local computer stores

is usually people who already know how to build a PC and looking for specific hardware or ones with the lack of knowledge on how to build a PC, while our main focus is on the latter so that is why we are developing this mobile application in hopes of companies acquiring our software.

Even though there are stores that provide lists of hardware for customers to choose from in the current market, this oftens gives an impression to customers they might not be updated, and more often than not, these lists have a set budget and little to no customisation options, we can see that they lack flexibility. Our system offers an instantaneous, most efficient, most up-to-date analysis for customers while offering flexibility while offering customisation options according to their own preferences. This would help companies to reach a wider audience, in turn more profit for the company which considers our software a worthy acquisition.

V. Requirements and Design Specifications

1. Functional Requirements
   1. PC Hardware Catalog  
      A full list of PC hardware will be displayed in this tab and sorted into their corresponding categories, including CPU, Graphics Card, Motherboard, RAM, Storage, Power Supply, Case, Disk Drive.
   2. Staff’s Pick/Recommended List  
      Several recommended PC builds will be shown. Details of each build will be shown upon request. This function is for those who are new to building a PC and provides them with choices on suggested builds with set components. The lists will be updated on a monthly basis.
   3. Basic automatic PC build generator with moderate customisation  
      The user will be asked on the budget they hope to spend on a PC build by using a slider and be given three choices regarding their planned usage, namely , example: gaming, and based on our algorithm, the application itself will generate a list for the user.
   4. Advanced automatic PC build generator using questionnaire

As opposed to the aforementioned build generator as listed in (1c), this allows further customisations so the PC build generated for our customers fulfills their expectations, a list of the questions and choices that will be included in the questionnaire are as follows:

1. Please input your budget for building the PC.  
   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Which brands of Graphics Processing Unit (GPU) do you prefer?

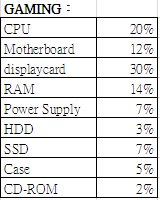
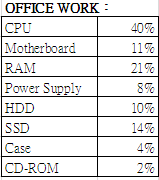
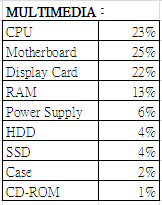
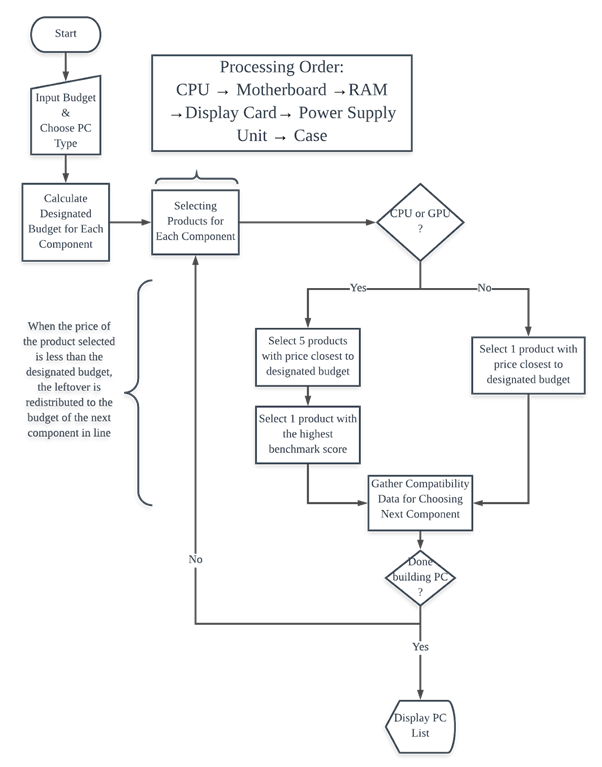
□Nvidia

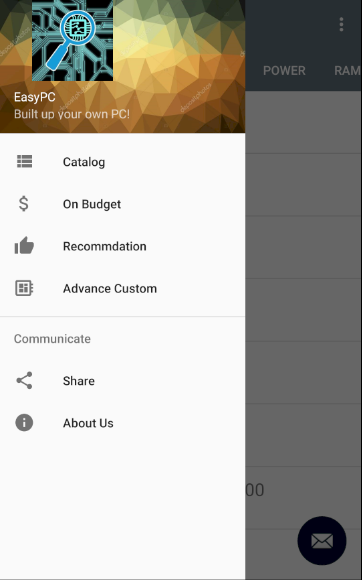
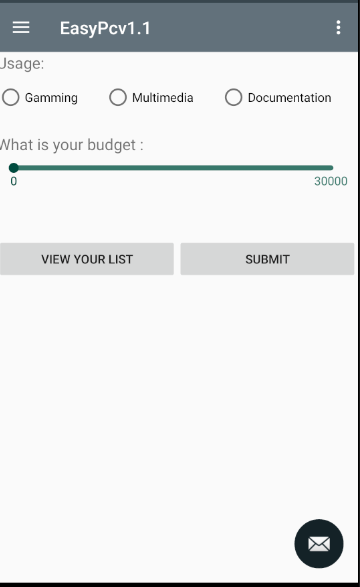
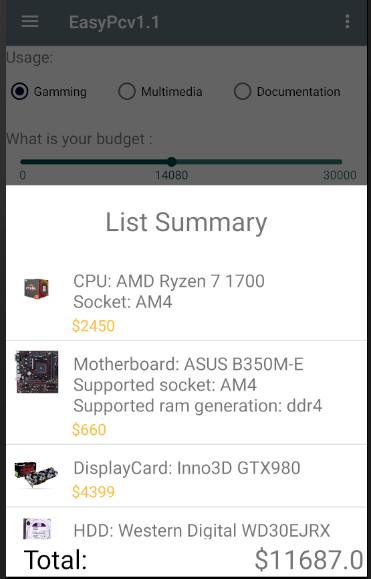
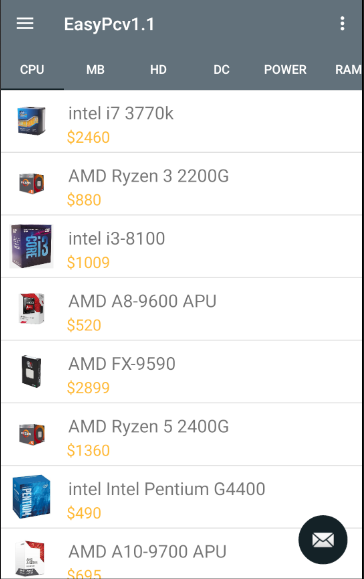
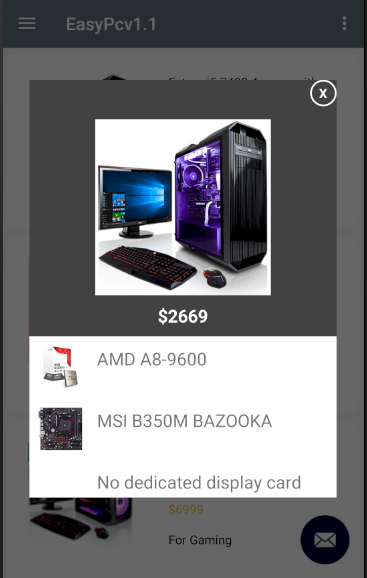
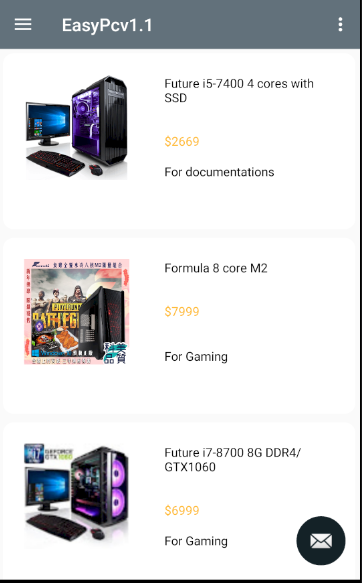
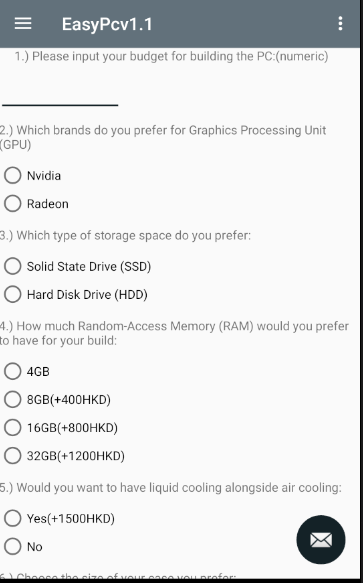
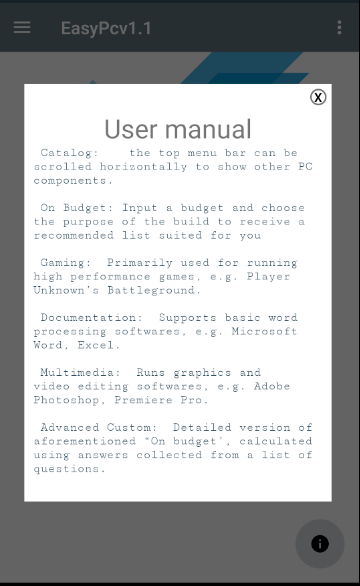
□Radeon

1. Which type of storage space do you prefer?

□Solid State Drive (SSD)   
□Hard Disk Drive (HDD)   
□Hybrid (SSD & HDD)

1. How much Random-Access Memory (RAM) would you prefer to have for your build?   
   □4GB  
   □8GB  
   □16GB  
   □32GB
2. Would you want to have liquid cooling alongside air cooling?  
   □Yes  
   □No
3. Choose the size of your case you prefer:  
   □Large (for ATX Motherboard, 12” x 9.6”)   
   □Medium (for MATX Motherboard, 9.6” x 9.6”)   
   □Small (for ITX Motherboard, 6.7” x 6.7”)   
   \*Please note that the height of the cases varies and the size of the case is based off of the Motherboard.
4. Please tick the corresponding boxes if you would want the following features/peripherals along with your build.  
   □ Microsoft Windows 10 (+300HKD)  
   □ ESET NOD32 Antivirus (+199HKD)  
   □ Adobe Photoshop (+229HKD)  
   □ Microsoft Office 2016 (+990HKD)  
   □ Logitech Wireless Mouse M320 (+199HKD)  
   □ Monitor (+1380HKD)  
   □ NZXT Hue+ Controller RGB Lighting (+470HKD)  
   □ Microsoft All-in-One Media Keyboard (+320HKD)  
   □ Kingston HyperX Cloud Stinger Headset (+348HKD)  
     
   1. Database
      1. A database is required to store all the information of the hardware components for our mobile application and for the algorithm to function as intended.
      2. The database itself is object-oriented, meaning that all the relevant information of each type of hardware component is stored within one table. Due to the fact that most of the components for building PCs have many-to-many relationships, which means that it will require a large amount of composite entities to be created if we were to implement a traditional database.
      3. Multiple Google Spreadsheets are used to replace traditional databases for storing data, each spreadsheet represent a table in a traditional database, and the relationships are handled by the algorithm and SQL-like query statements provided by Google. Further information regarding this language can be seen here: <https://developers.google.com/chart/interactive/docs/querylanguage>
      4. This method requires internet connection in order to access the databases.
   2. Algorithm
      1. The algorithm is developed based on the users’ input (budget and purpose of computer) and also considering all the compatibilities between each component to make sure that the parts picked can actually be built into a real computer.
      2. Firstly, how the purpose of the computer comes into play affects how the algorithm choose components. The budget is first divided by their weighted scores depending on the purpose of the computer being built.   
         To further elaborate on this, a computer that is strictly for processing documents and daily usage, will have little to none of its budget spent on a dedicated graphics card, while for a gaming computer, more of the budget must be spent for the same component.

* + 1. The weight scores are gathered by calculating the percentages of each component’s retail price as opposed to the price of the list. This calculation would give a rough idea on how the algorithm should spend its money on for the three main purposes, gaming, multimedia production, and office work. The percentages of each option are as follows:  
         
       For Gaming:   
         
       Total: 100%  
       For Office Work:  
         
       Total: 100%  
         
       For Multimedia:  
         
       Total: 100%
    2. Next, the compatibility is another concern when building a personal computer. Each component has different sockets and not all components would fit each other. The components chosen latter in the sequence would be chosen according to the compatibility info(socket) of the previous component chosen. The sequence for processing the lists is as follows: CPU -> Motherboard -> RAM -> Display Card (GPU) -> Power Supply Unit (PSU)-> Case.
    3. Another thing to note is that the Thermal Design Power (TDP) also needs to be taken into account as it has to be made sure that the power supply unit has enough power to provide to all the parts of the system. Each component has an TDP, by adding up all the TDP of all the components selected, it will show the TDP needed, then the algorithm can then choose the power supply unit that suits the build. Therefore, it would be selected in later parts of the algorithm.
    4. Components (CPU and GPU) that cost higher do not necessarily mean that the aforementioned components would perform better or be more cost-effective than components that cost cheaper. To address this, benchmark scores are used to determine whether they belong in the list. 5 components with the closest price to the designated budget for a specific part is furthered considered by comparing their benchmark scores, the higher the benchmark score, the better they are, and the one with the highest benchmark score and within the designated budget will be chosen to be on the list. The benchmark scores are acquired here: <https://www.cpubenchmark.net/> & <https://www.videocardbenchmark.net/>
    5. To ensure that the user will have their budget spent in the most efficient manner, the excess budget from choosing a component will be passed onto the next component in the sequence.   
         
       The following flowchart displays graphically on how the algorithm works:
    6. For example, if the input budget is $10000 and the type is Multimedia, the budget for CPU would be $2300, that of motherboard would be $2500, that of display card would be $2200, etc. Noted that the sequence mentioned above is as follows: CPU -> Motherboard -> RAM -> Display Card (GPU) -> Power Supply Unit (PSU)-> Case. If the budget allocated for CPU is $2300, and the actual cost for the selected CPU is $2250, the leftover $50 will be reallocated to the budget of Motherboard (which is next in the sequence).

1. Non-Functioning Requirements
   1. User Interface (UI)
      1. Navigation Drawer  
           
         A panel on the left would pop up if the user touches the left border of the screen and swipe left or simply tapping the button with three horizontal lines. The panel shows different options and functionalities our application has. Icons of commonly used for that specific usage are utilised to represent graphically what each function does. As seen from the figure above, the three horizontal lines with three dots on the left icon refers to catalog, the dollar sign refers to our “On budget” function, the thumbs up icon refers to our recommended lists, while the chip represents our “Advance Custom” function. Sharing and About Us pages are also available.
      2. On Budget Page  
           
         Radio buttons are used for selection of the usage of PC. The slider is used for users to select their budget, ranging from $0 - $30000. While the circle is held and moved along the bar, a number will be shown representing the budget selected. Buttons are also added in the middle of the screen for viewing the list generated and submission of inputs.
      3. Bottom Sheet Dialog  
           
         When input is submitted and user clicks “View Your List” as seen from (2aiii), a bottom sheet dialog would contain a transition effect that pops up from the bottom of the screen to the top. Details of each component and the total cost will be shown. User can scroll the middle of the dialog to navigate the different components selected by the algorithm. By holding the top border of the dialog and swiping downwards, the dialog will be hidden until the “View Your List” button is clicked again.
      4. Tab Layout:  
           
         This is used in the catalog. By clicking on the items on the menu bar, details all available components of that specific type will be shown on the screen. By holding and swiping the menu bar left or right, it will show more options of the types of components.
      5. Recommended List:  
           
         Card views and dialogs are used here. The lists are shown and if users want further information of each build, they can click on them and a pop up would be shown and the details can be scrolled up and down to view the details of other components of that build.
      6. Advance Customize:  
           
         A form is used to represent our questionnaire which consists of radio buttons, text views, checkboxes and user input. User can also choose to submit their answers when they have completed the form and click “View Your List” to review their list.
      7. User Manual:  
           
         On each page of the application, a floating action button appears on the bottom right of the screen, when clicked, a dialog would pop up and reveals the user manual for our application. Users can use this if they happen to not understand how our application works.
      8. Images embedding:  
         C:\Users\Administrator.WIN7-1607151737\Desktop\individual-20180409T043200Z-001\individual\9.PNG  
         Used to show the details of a specific component, including the brand, the series, the price, and the appearance.

VI. Evaluation

Testing methodologies

1. Testing stability and consistency On Budget

Input $8000 of the same usage three times and check if the results are the same.

(Refer to Table 1.2 in Appendix for further details)

Results: Lists acquired from the three attempts all resulted the same components.  
 This means that there is a high level of stability and consistency regarding the algorithms

1. Testing accuracy of On Budget
   1. Input a set amount $8000 for each usage type , and manually calculate the percentages and compare the ratios used with the one acquired.

TEST RESULT:

1st case 8000 documentation cost cost/budget

CPU intel i3 8100 lga 1151 1009 0.126

Gigabyte ega h110m h lga 1151 ddr4 430 0.05375

MSI GTX1050 1250 0.15625

Western digital wd30ejrx 629 0.078625

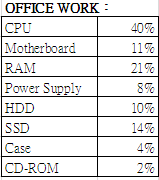
Kingston ddr4 2666 16gb 1600 0.2

Cooler master v850 1350 0.16875

Cooler master masterbox e500l 340 0.04

TOTAL COST: 6680

PLANNED BUDGET ALLOCATION:



(Refer to Table 1.2 for detailed results)

In terms of spending the budget, there are slight differentiations as for all the usage types. No signs of huge error. It can be seen that the later the component is in the sequence of processing, the more likely the cost of the component would be over the designated for the part because of the algorithm passing on leftover budget to components that are later in the sequence. The reason why there’s a rather significant disparity between the budget and the actual cost is because of the lack of variety for each type of products. If there were more data available, the results would be way more accurate.

* 1. Test the advance custom to see if customisation options are correctly applied.

PLANNED CHOICES:  
 Budget: 8000HKD

GPU Brand: Radeon

Storage space type: HDD

RAM size: 16GB

With Liquid cooling

Large motherboard

Liquid Cooling

Microsoft Windows

ESET NOD32 Antivirus

Adobe Photoshop

Microsoft Office 2016

Logitech Wireless Mouse M320

HP ProDisplay p19A Monitor

NZXT Hue+ Controller RGB

Microsoft All-in-One Media Keyboard

Kingston HyperX Cloud Stinger

RESULTS:

AMD Ryzen 5 2400G AM4 1360

GIGABYTE GA A320 DS3 AMR DDR4 600

SAPPhiRE Radeon RX550 1060

HGST HTS725050b7e630 305

Kingston ddr4 2666 16gb 1600

Cooler Master v850 1350

Cooler Master masterbox e500l 340

Liquid Cooling 1500

Microsoft Windows 300

ESET NOD32 Antivirus 199

Adobe Photoshop 229

Microsoft Office 2016 990

Logitech Wireless Mouse M320 199

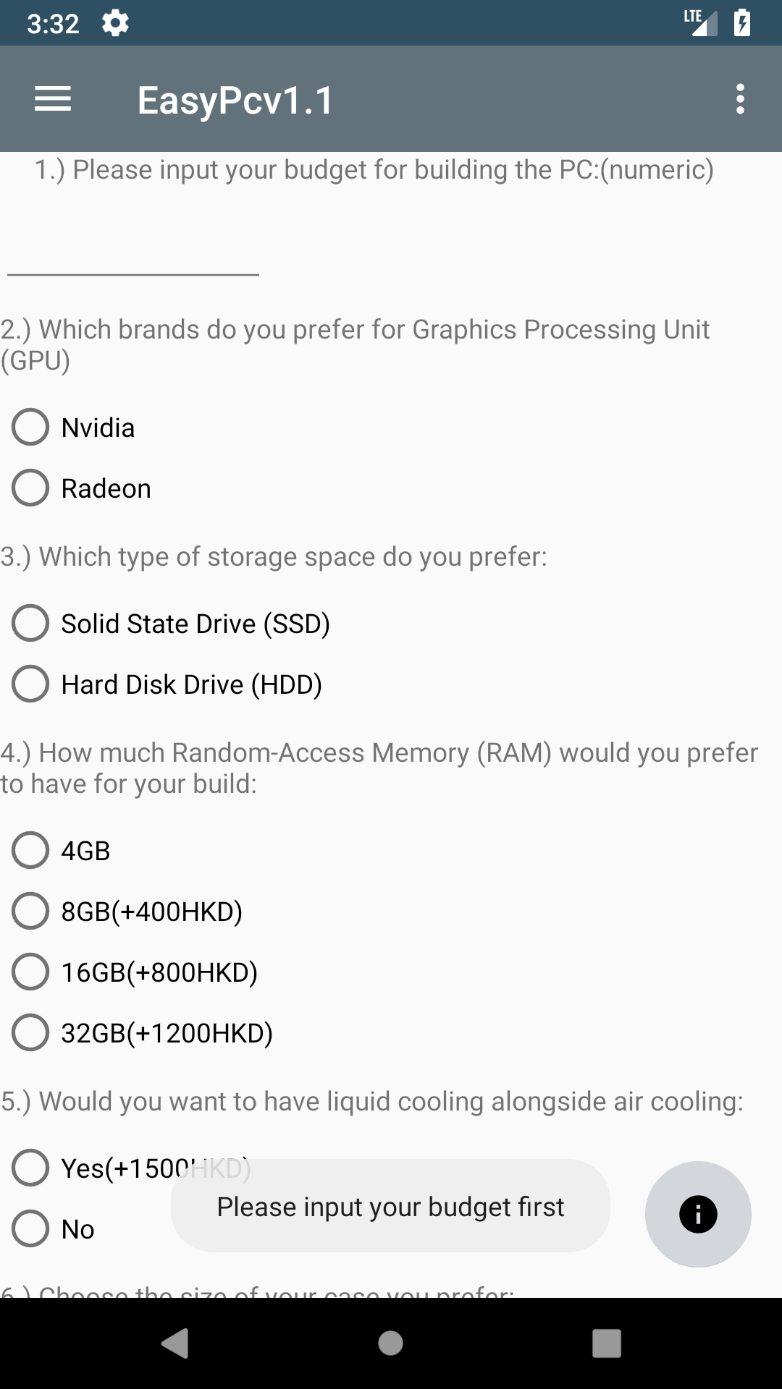
HP ProDisplay p19A Monitor 1380

NZXT Hue+ Controller RGB 470

Microsoft All-in-One Media Keyboard 320

Kingston HyperX Cloud Stinger 348

This shows that all the softwares or peripherals or add-ons with the right brands and specifications can be successfully added to the list and correctly added to the total. Noted that the additional fees are added on after the calculation for the list is done instead of consuming the original budget.

1. Testing for error checking  
   Inputting “0”s for budgets and failing to complete each questionnaire question.  
   

Error handling is successful. Errors are caught correctly and prompt the users to correct them accordingly. One of the examples is shown above.

1. Testing correct data display  
     
   Checking the data displayed on the catalog and recommended list and comparing that to the ones shown on the database(spreadsheets).  
     
   The results show that items are shown properly and in the right category.
2. User acceptance Test  
     
   Have end users a black box test to see if the application itself is user friendly and whether or not further changes have to be made in order to improve the user experience.   
     
   Results:   
   User suggests to show values of the softwares/peripherals/add-ons so as to ensure how much more they are using if they want that specific build in advance custom.  
   Suggestion has been adopted.  
     
   User suggests the catalog to be sorted in some way.   
   Suggestion has been adopted through sorting the catalog by value in ascending order.  
     
   User suggests that some wordings might be unclear for new users, adding a how this application works page would help.  
   Suggestion has been adopted, adding a user manual button using a floating action button that can be accessible from across the application.
3. Unit Test and Integration Test

Each part is section off from each other first and ensure that they work independently before moving on to integration. Once every standalone part is functional then we would start off to combine these separate parts into one application which would then be refined to turn into EasyPC today.

VII. Conclusion

Problems

1. The price of hardware may change frequently

There is a constant need to update the information on the database as prices of the components can change due to the introduction of new products in the market. The manufacturer will most likely reduce the price of the older products so that they can sell their old products faster. This shows that database has to be changed rapidly as time passes so as to ensure our database is the most up-to-date, including the addition of details of new products and adjustments to price of older products. We are unsure what to do in order for the software to have an updated database consistently.

1. Difficulties in considering the best parameter to use in algorithm

In this software, we aim to come up with an algorithm that is the most cost-effective, accurate, and suits customers preferences at the same time. We have to take the performance of individual components into consideration and determine which parameter the performance is based off of in an objective manner. It is hard to do so as there are many different variables in each component. Even with the algorithm we have, it is theoretically the best cost-effectiveness wise, but it might not be in practice. We have no way to assure that the PC list generated would best suit any scenarios. We only plan to be as accurate as we can using the knowledge we know.

1. Network Connection is a must

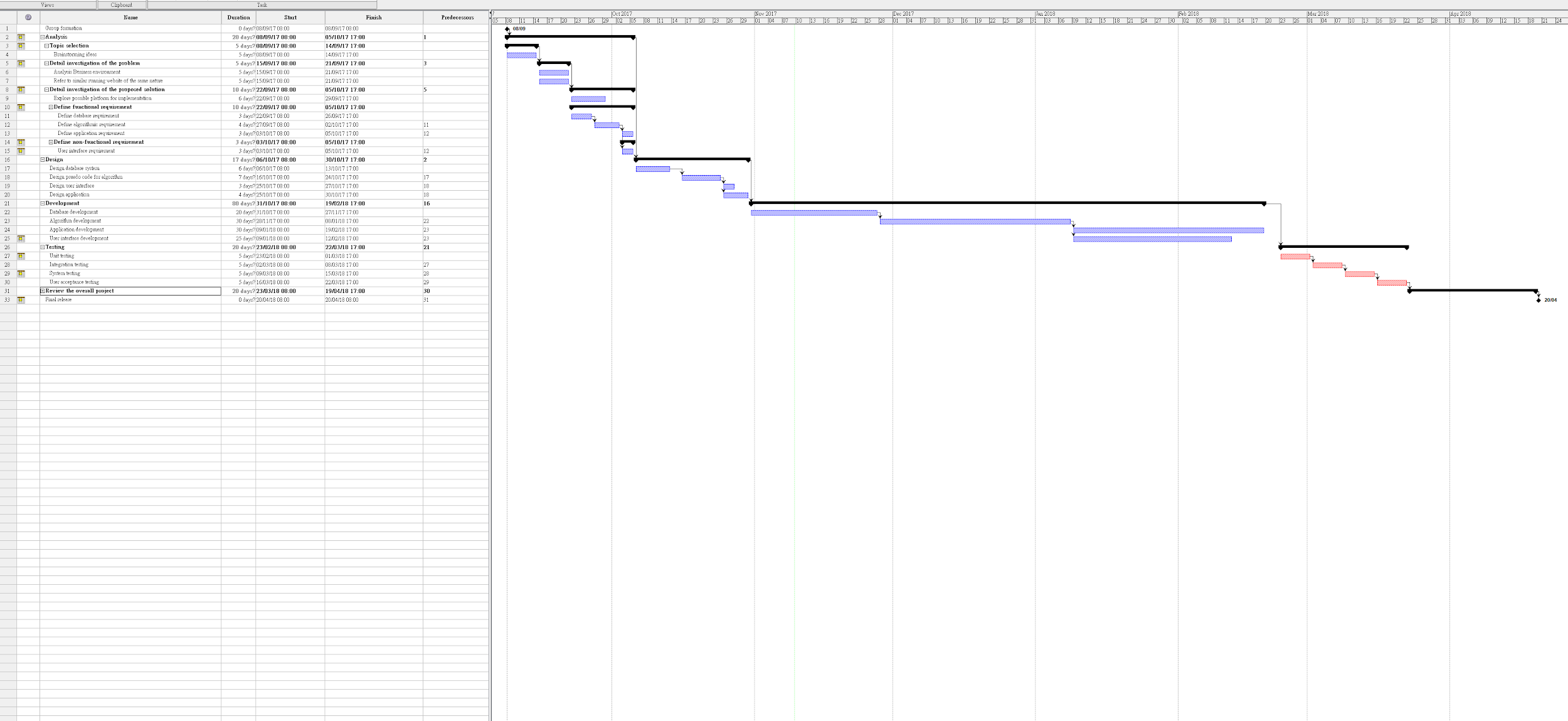
Since our application utilises Google Spreadsheets as our databases system, the device has to be connected to the internet in order for the application to function. But due to limited knowledge and resources, we are unable to use paid services or NoSQL databases.

1. Hard to catering to customers’ needs using limited amount of questions  
     
   It is hard to find a compromise between asking too much and too little information, it might come off as too intrusive or annoying when customers are asked too many questions but there might not be enough information for the algorithm to perform a precise judgement on what to put in the PC build to suit the customer’s preferences. We feel like the amount of questions we have on the questionnaire is just right but some users might say otherwise.
2. Rather slow processing time  
     
   The processing part of the application during On Budget or Advance Custom, consumers roughly 10-15 seconds to process, which to some might be way too long for an automated system. For mobile phones with poor specifications, might struggle when they use our application. Minimising our load times without 1-5 seconds would be a great improvement to our system.
3. No multi-language options  
     
   People who do not understand English will have a hard time using our application and even though we would hope to be able to distribute our application to a wider scale. As of right now, with the resources we have, it is not possible to implement multi-language support for our application.

Review of our project  
   
All in all, our application was completed on time and according to schedule and most importantly, of satisfactory results. Our project team had a lot of synergy and great minds to create such a detailed application for use. Each of our groupmate had their strengths and weaknesses and we utilised them well. Improvements can be made to increase the standard and perhaps that will bring us closer to be acquired by companies that would like to broaden their customer base and the application could also be modified to suit the companies’ needs into becoming a great asset that helps the business become more profitable.

Project Schedule

The following figure is the Gantt Chart representing the schedule for our project. For further inspection, please refer to our attached the file with the filename of “Final Year Project.pod”.



Resource Allocation

|  |  |  |  |
| --- | --- | --- | --- |
| Group member | Main roles | **Task in charge** | **Time allocation** |
| NG Ho Cheung John Bosco | Topic Selection | Brainstorming ideas | 15 hours |
| Project Manager | Project schedule | 16 hours |
| Resource allocation | 16 hours |
| Database Designer | Define database requirement | 9 hours |
| Design database system | 23 hours |
| Database development | 80 hours |
| Programmer | Assisting in programming | 25 hours |
| Testing | 4 different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **304 hours** |
| MO Yu Hin | Topic Selection | Brainstorming ideas | 15 hours |
| Business Analyst | Analyse business environment | 25 hours |
| Application Developer | Define application requirement | 19 hours |
| Design application | 22 hours |
| Application Development | 100 hours |
| Testing | 4 Different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **301 hours** |
| TUNG Pak Hei | Topic Selection | Brainstorming ideas | 15 hours |
| Technical Writer | Documentation | 20 hours |
| Algorithm Developer | Define algorithmic requirement | 17 hours |
| Design pseudocode for algorithm | 33 hours |
| Algorithm development | 100 hours |
| Testing | 4 different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **305 hours** |
| MIAO Weichu | Topic Selection | Brainstorming ideas | 15 hours |
| Researcher | Refer to similar running website of the same nature | 18 hours |
| Explore possible platform for implementation | 23 hours |
| UI Designer | Define user interface requirement | 12 hours |
| User interface development | 100 hours |
| Testing | 4 different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **301 hours** |

Topic Selection, testing and review the overall project are the common roles that every group mate involved.

NG Ho Cheung John Bosco is assigned as Project manager, Database designer because he is the group leader of the team, who is responsible for the management works, is interested in database design.

MO Yu Hin is assigned as Business Analyst, Application Developer, Application Programmer because has a good sense in business, has fast and accurate development skills.

TUNG Pak Hei is assigned as Technical Writer, Algorithm Developer, because he is fluent in English and has a great understanding in the aforementioned areas.

MIAO Weichu is assigned as Researcher and UI Designer is because he is knowledgeable in computer hardware and is experienced in using Adobe Photoshop, including designing User Interface.

APPENDIX

Fig 1.1:

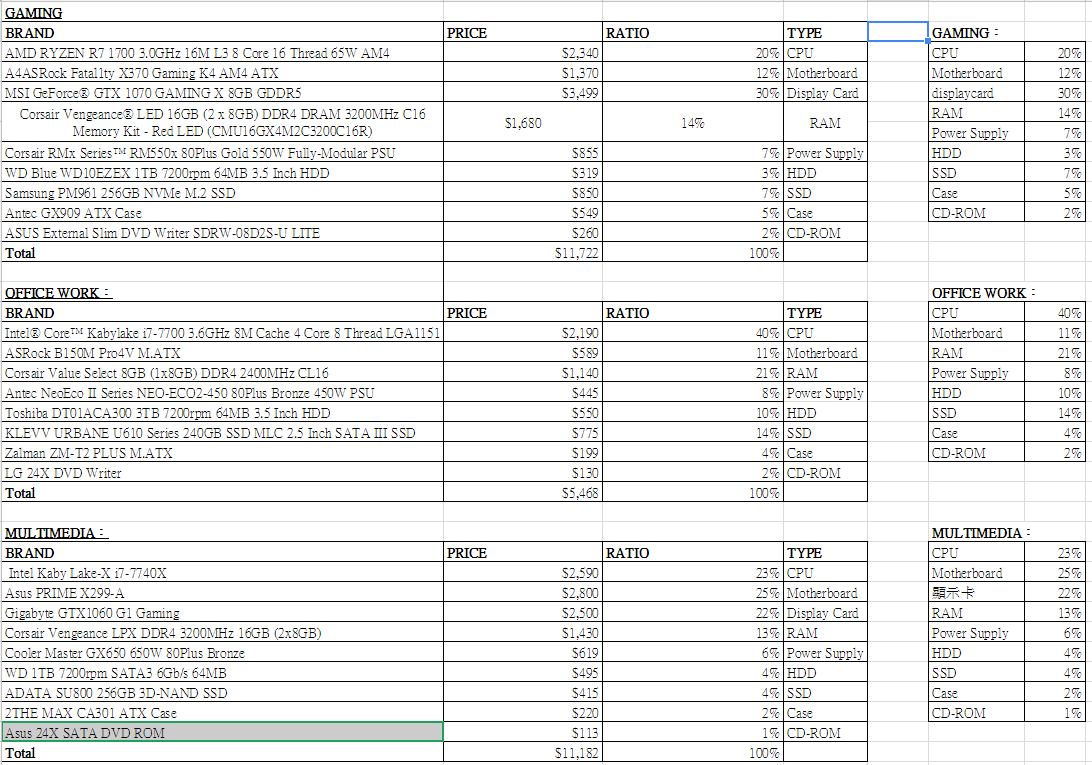


Table 1.2

Results:

1st attempt (8000) Gaming cost cost/budget  
AMD ryzen 5 1600x 2000 0.25

Asus b350me am4 ddr4 660 0.0825

Msi gtx1050 1250 0.156

Hgst HTS725050b7e630 305 0.038

Corsair ddr4 3200 8gb 920 0.115

Cooler master v850 1350 0.16875

Cooler master masterbox e500L 340 0.04

TOTAL COST: 6825

2nd attempt (8000) Gaming cost cost/budget  
AMD ryzen 5 1600x 2000 0.25

Asus b350me am4 ddr4 660 0.0825

Msi gtx1050 1250 0.156

Hgst HTS725050b7e630 305 0.038

Corsair ddr4 3200 8gb 920 0.115

Cooler master v850 1350 0.16875

Cooler master masterbox e500L 340 0.04

6825

3rd attempt (8000) Gaming cost cost/budget  
AMD ryzen 5 1600x 2000 0.25

Asus b350me am4 ddr4 660 0.0825

Msi gtx1050 1250 0.156

Hgst HTS725050b7e630 305 0.038

Corsair ddr4 3200 8gb 920 0.115

Cooler master v850 1350 0.16875

Cooler master masterbox e500L 340 0.04

6825

1st case 8000 multimedia cost cost/budget

AMD ryzen 5 1600x 2000 0.25

Asus b350m e am4 ddr4 660 0.0825

MSI GTX1050 1250 0.156

`Hgst GTS725050b7e630 305 0.038

Kingston ddr4 2666 16gb 1600 0.2

Cooler master v850 1350 0.16875

Cooler mster masterbox e500L 340 0.04

7505

1st case 8000 documentationcost cost/budget

CPU intel i3 8100 lga 1151 1009 0.126

Gigabyte ega h110m h lga 1151 ddr4 430 0.05375

MSI GTX1050 1250 0.15625

Western digital wd30ejrx 629 0.078625

Kingston ddr4 2666 16gb 1600 0.2

Cooler master v850 1350 0.16875

Cooler master masterbox e500l 340 0.04

6680

Chart 1.3

Project Schedule

The following figure is the Gantt Chart representing the schedule for our project. For further inspection, please refer to our attached the file with the filename of “Final Year Project.pod”.

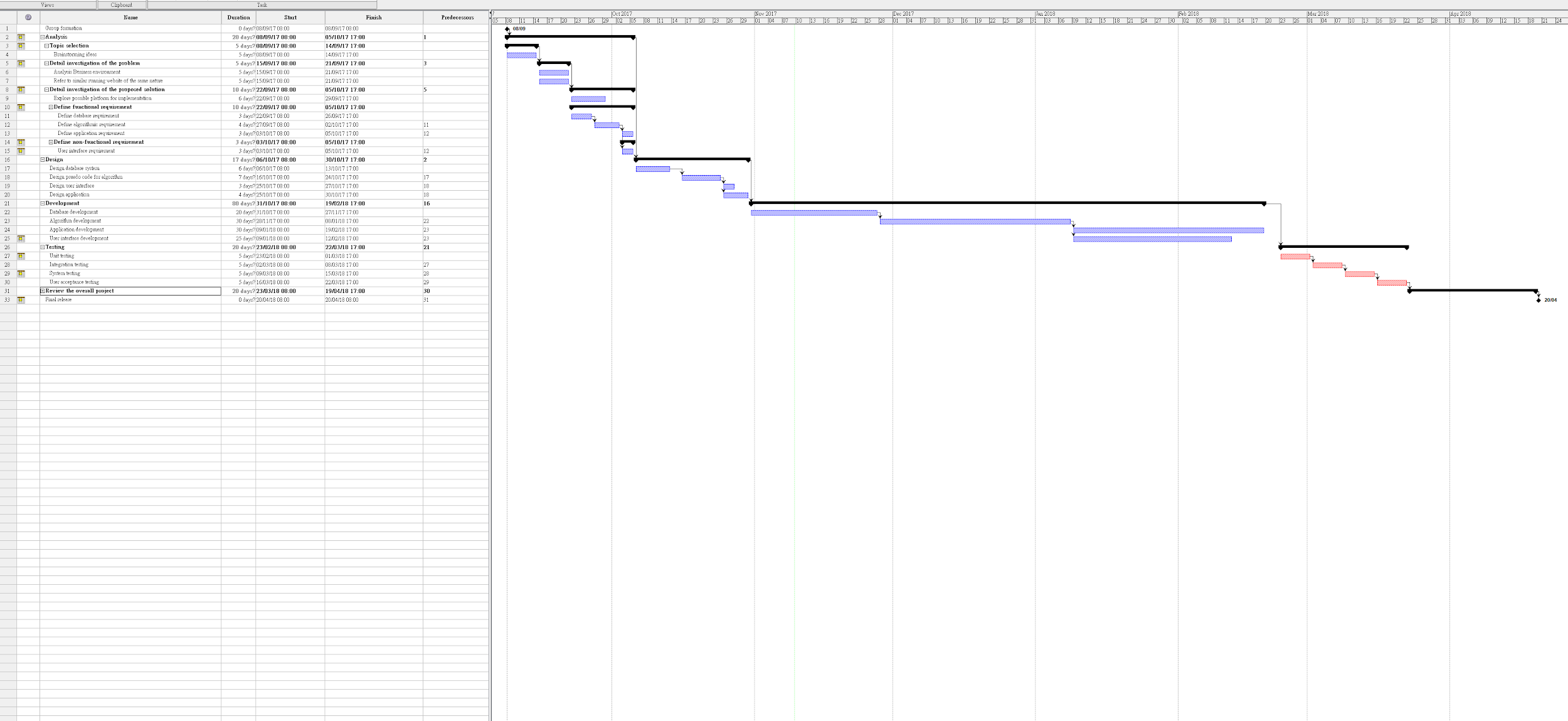


Chart 1.4  
Resource Allocation

|  |  |  |  |
| --- | --- | --- | --- |
| Group member | Main roles | **Task in charge** | **Time allocation** |
| NG Ho Cheung John Bosco | Topic Selection | Brainstorming ideas | 15 hours |
| Project Manager | Project schedule | 16 hours |
| Resource allocation | 16 hours |
| Database Designer | Define database requirement | 9 hours |
| Design database system | 23 hours |
| Database development | 80 hours |
| Programmer | Assisting in programming | 25 hours |
| Testing | 4 different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **304 hours** |
| MO Yu Hin | Topic Selection | Brainstorming ideas | 15 hours |
| Business Analyst | Analyse business environment | 25 hours |
| Application Developer | Define application requirement | 19 hours |
| Design application | 22 hours |
| Application Development | 100 hours |
| Testing | 4 Different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **301 hours** |
| TUNG Pak Hei | Topic Selection | Brainstorming ideas | 15 hours |
| Technical Writer | Documentation | 20 hours |
| Algorithm Developer | Define algorithmic requirement | 17 hours |
| Design pseudocode for algorithm | 33 hours |
| Algorithm development | 100 hours |
| Testing | 4 different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **305 hours** |
| MIAO Weichu | Topic Selection | Brainstorming ideas | 15 hours |
| Researcher | Refer to similar running website of the same nature | 18 hours |
| Explore possible platform for implementation | 23 hours |
| UI Designer | Define user interface requirement | 12 hours |
| User interface development | 100 hours |
| Testing | 4 different stages of testing | 60 hours |
| Overall Review | Review | 60 hours |
|  | | Total: **301 hours** |

Topic Selection, testing and review the overall project are the common roles that every group mate involved.

NG Ho Cheung John Bosco is assigned as Project manager, Database designer because he is the group leader of the team, who is responsible for the management works, is interested in database design.

MO Yu Hin is assigned as Business Analyst, Application Developer, Application Programmer because has a good sense in business, has fast and accurate development skills.

TUNG Pak Hei is assigned as Technical Writer, Algorithm Developer, because he is fluent in English and has a great understanding in the aforementioned areas.

MIAO Weichu is assigned as Researcher and UI Designer is because he is knowledgeable in computer hardware and is experienced in using Adobe Photoshop, including designing User Interface.