Project Specification

CITS 5206 Professional Computing

Unit Coordinator: Prof. Rachel Cardell-Oliver

Unit Budget Planner

Student Name	Student ID
Liangbo Jin	23078811
Ethan Chen	23067035
Lida Tong	22957193
Yu Zhu	23053734
Keli Yin	22450718
Nara Eam	22805226

Table of Contents

Glossary of Term

1. System: The budget planning application which facilitate stakeholders in planning and checking the budget.
2. User: All stakeholders that will be using the application.
3. Client: Prof. Rachel Cardell-Oliver, Head of Department, UWA Computer Science.

1. Background of Current System

Current System

Monitoring unit budget in the School of Physics, Mathematics and Computing for each semester is a very cumbersome and tedious process using spreadsheet system. Each unit coordinator needs to fill in the EMS Unit budget spreadsheet for individual unit every semester, which includes the hours allocated to preparation and delivery of teaching activities and marking. Also, it includes casual salary costs and any software or hardware costs. After gathering all the unit budget spreadsheets, the academic service assistant manually combined them for the Head of Department.

Current Limitations

One limitation of the current process is that the budget spreadsheet is at individual unit level for each semester, it is difficult to break down the hours and costs of different types of activities for delivering units.
Moreover, there is no automatic check of the business rules when submitting the unit budget spreadsheet and keep track of the budget during semester as some activities may take longer time than others such as marking activities.

Project Objectives

To improve the process, Head of Department is looking for a solution software that can automatically read the excel files and save the data in a database which contains current and historical unit budgets. She would also prefer to have a user interface for entering and/or updating unit budget information. The system can then process the data and produce the summary report that facilitate her in understanding the costs of workload hours and paid casual teaching, compare the budgets of different groups of units and track budget expenditure during the semester.

2. Client and Stakeholders

Client

Rachel Cardell-Oliver, Head of Department, UWA Computer Science
Rachel is the Head of Department who will need to use the system to oversee and manage the unit budget and spot any anomalies. She would like the system to be able to check business rules automatically after uploading all the unit budget spreadsheets to the system and be able to provide a summary report of each unit budgets. For easier analyse, the system should allow her to write queries to filter the items that she is interested in. In addition, the system should allow monitoring of the budget progress throughout the semester.

Other Stakeholders

Academic Services Staff
Hass Sadeghi-Barzani is the Academic Services staff who need to manually combine all the unit budget spreadsheets provided by the unit coordinators every semester. The system will improve his working efficiency in doing this tedious job.

Head of School
Head of School needs the system to better at doing budgeting every year. As currently it is hard to compare the hours and costs of different types of activities for delivering different units in different semester.

Head of Finance
Head of School will need the system at better doing analysis for bookkeeping. As currently it is hard to drill down on the unit budget data and categories them in the individual accounts.

3. Scope of Project

(Refer to GitHub Unit_Budget_Planner/Group_Meeting/Meeting_Record/ 20210805_Meeting_Minutes)

3.1 Project Start Date

29 July 2021

3.2 Project End Date

22 October 2021

3.3 Project Scope

Description

This unit budget planner is being undertaken for the School of PMC for the purpose of creating an application to simplify the process of planning unit budget and producing the summary report.

In Scope

- Develop one application for the client which meet the requirements
- Provide demonstration and documentation

Out of Scope

- Provide support after final delivery
- Adapt the application for other school

3.4 Assumptions

- Each user will use similar spreadsheet format.
- Only authorized person can access to the spreadsheets.
- The user has the knowledge of structured query language.
- Tools chosen can perform well and meet expectations.
- All stakeholders have similar requirements and priorities.
- The project scope is confirmed and will not be changed in the later stage.

3.5 Constraints

There are some constraints that the project must follow:

- The input of the application must be the spreadsheets which users are currently using.
- The application must be a desktop application.
- The ~~example~~ data must be kept confidential.
- Each deliverable must be provided in time.
- The project must be done before deadline.

3.6 Resources

In developing this system, two main resources will be used which will include software and data. The software that we use to develop this system will mainly be open-source software such as ElectronJS, SQLite and Python. The main data is the spreadsheets that provided by the users for budgeting analysis.

3.7 Team Members & Skills

(Refer to GitHub Unit_Budget_Planner/Team_Skills)

Team members and skills are important to the success of the project as it can help in dividing the responsibilities for each team members, assigning relevant tasks and choosing the development tools.

3.8 Safety, Security & Risks

Risk Number	Risk Description	Impact	Occurrence	Risk Management Method
1	The unit budgets are confidential data, there is a security risk that unauthorize users might access to the database.	High	Low	Store all data locally, and program run as desktop application.
2	Risk of unexpected delay as none of the group members have prior experience.	Medium	Medium	Communicate and report current progress daily, and comprehensive research before starting the project. Using the idea of minimum viable products, start from something small that is enough for client to start and test with.
3	Insufficient communication as most of the	Low	Medium	Make meeting agenda and minutes, make sure everyone is on the right track.

	communication take place online.			
4	Risk of lack of compatibility testing.	Low	Medium	We will try to run the application on varied computers includes computer in campus which should be similar to user’s computer environment
5	Some assumptions during the application development are made wrongly.	High	High	Try not to make any assumption before discussing with client.

4. System Requirements

(Refer to GitHub Unit_Budget_Planner/Group_Meeting/Meeting_Record/ 20210812_meeting.docx)

Requirement 1:
Requirement types: Functional
Event/Use cases:

- A user can run the application on their local machine according to their Operating System.

Description: The app should be in a desktop format
Rationale: The data that used are sensitive and confidential so only people with the system can used.
Fit criterion: The system should run locally without needing the internet connection
Conflicts: None
Source: This requirement was raised by the head of the department

Requirement 2:
Requirement types: Functional
Event/Use cases:

- A user uploads the spreadsheet

Description: The system shall allow the user to import the spreadsheet
Rationale: The main input from the user is spreadsheet.
Fit criterion: The system should be able to read the spreadsheet and store the information in the database
Conflicts: None
Source: This requirement was raised by the head of the department

Requirement 3:
Requirement types: Functional
Event/Use cases:

- A user uploads a bulk of spreadsheet at the same time

Description: The system shall allow the user to import a group of spreadsheets
Rationale: The main input from the user is a bulk of spreadsheet.
Fit criterion: The system should be able to read and store a group of spreadsheets data into the database
Conflicts: None
Source: This requirement was raised by the head of the department

Requirement 4:
Requirement types: Functional
Event/Use cases:

- The system should produce the summary report

Description: The system should be able to produce summary report
Rationale: The interactive dashboard that produce the summary report provide benefits to the users by enabling user to be able to spot any anomaly easily compared to just checking from the spread sheet.
Fit criterion: The summary report should be produced in a dashboard format
Conflicts: None
Source: This requirement was raised by the head of the department

Requirement 5:
Requirement types: Functional
Event/Use cases:

- A user can choose option on the menu to filter

Description: The system should be able to filter the summary report
Rationale: User might want to see specific information such as budget for semester 1 2019 budget for specific unit.
Fit criterion: The system should have menu for users to filter maybe filtering according to hour spend on delivery or assessment
Conflicts: None
Source: This requirement was raised by the head of the department

Requirement 6:
Requirement types: Functional
Event/Use cases:

- A user can write their own SQL query to get the information they want

Description: The system shall allow user to type in their own SQL query
Rationale: The filter function that is already set might not be enough for user to get specific information they want.
Fit criterion: The query function should work the same way as the SQL query language
Conflicts: None
Source: This requirement was raised by the head of the department

Requirement 7:
Requirement types: Functional
Event/Use cases:

- A user uploads the spread sheet with comment

Description: The system should be able to store the comment from the spreadsheet
Rationale: Some comments are important because they are the business rules, therefore, user want to store in a database, so it is easy to find when they need it later.
Fit criterion: The system should be able to read in the comment and store in the database
Conflicts: None
Source: This requirement was raised by the head of the department

Requirement 8:
Requirement types: Functional
Event/Use cases:

- User has already uploaded the spreadsheet and spot an anomaly

Description: The system should allow the user to pull up the comment and check if there is actual anomaly happen.
Rationale: User want to compare the anomalies with the business rules and check whether it is important to let tutors or unit coordinator know.
Fit criterion: The system should provide a way to retrieve the comment for the user to check when anomaly happen
Conflicts: None
Source: The requirement was raised by the head of the department

Requirement 9:
Requirement types: Functional
Event/Use cases:

- User can track current and historical budget

Description: The system shall be able to track the historical budget for user
Fit criterion: Users might want to compare the previous budget and current budget for specific unit
Fit criterion: The system should have a function for getting the total previous budget and total current budget.
Conflicts: None
Source: The requirement was raised by the head of the department

Requirement 10:
Requirement types: Non-functional (User Interface)
Event/Use cases:

- A user can click on roles from the menu according to their roles

Description: The system shall provide different interface for different users according to their roles such as Unit Coordinator and Head of department
Rationale: Different users might want to see different functions that they can perform
Fit criterion: The system should have the function for user to choose different roles according to their own roles.
Conflicts: None
Source: The requirement was raised by the head of the department

5. Future Requirements

(Refer to GitHub Unit_Budget_Planner/Resources/Future_requirements.docx)

There are some requirements that may be a good idea to implement but not at the current stage of the development. Future requirements that need to be consider may include

- Users able to login and can see different information for different users
- The system can automatically check the business rules
- Monitor the cost information
- Import enrolment number
- Make it to be a secure web page

All these requirements are important however they either have low impact or low urgency to the stakeholders at the moments.

6. Technology Choices

(Refer to GitHub Unit_Budget_Planner/Group_Meeting/Meeting_Record/20210805_Meeting_Minutes)

Based on our team's capabilities, licencing and ease of use, we decide to choose 'Electron' as the main framework for developing our project. The 'Electron' supports most of the technologies that our team already known, which includes SQLite, Python, JavaScript, and related Testing frameworks.

6.1 Database:

Options:

- **SQLite**
  SQLite is very lightweight and easy to deploy. There are plenty of simple GUI that can help to easily access data and quickly modify testing data during project development. Although the functionality of SQLite may be limited, it is powerful enough to suit this project. SQLite can save data in a single local file, which in our case is more secure.

- **MySQL**
  MySQL is an open-source relational database management system. It is similar to SQLite. MySQL is also a good fit for this project, but not everyone in our group has experienced in using it therefore we are less familiar with MySQL than SQLite.

- **Oracle**
  Oracle Database is a multi-model database management system. Typically, Oracle is used for databases that run online transaction processing, data warehousing. It is extremely powerful, but it is complicated in configuration, deployment, and management plus it required licensing. Therefore, it is not suitable to use in this project.

Final decision： SQLite

6.2 Desktop APP:

Framework:

Options:

- **ElectronJS**
  Electron JS is a free and open-source software framework. It allows for the development of desktop GUI applications using web technologies. Most of us have Web development experience and Electron gives us the ability to use Web development skills to implement a Desktop APP.

- **JavaFX**
  JavaFX is a Java library. The applications written using this library can run consistently across multiple platforms. However, there is no requirement that the final product needs to support multiple platforms. In addition, not everybody in our group can do programming proficiently. Also, Java requires the installation of JDK and the IDE such as IntelliJ, which is more complicated to use.

Final decision: ElectronJS

Front-end:

Options:

- HTML + CSS + JavaScript + JQuery
  This combination of HTML + CSS + JavaScript + JQuery has been popular for years and there are plenty of resources that we can reference. And this combination is what we did in last semester in CITS5505, so everybody in our group is happy and confident to use this again.

- React + Sass
  React framework is efficient and flexible. It works well with known libraries or frameworks and can make the code more easily reused. However, our project is too simple to use React. Also, it will take much longer to set up the environment for React.

Final decision: plain HTML + CSS + JavaScript + JQuery

Back-end:

Options:

- Python
  Python provides a lot of libraries that may be very useful for this project (e.g. talking to SQLite and creating REST endpoints). Most importantly, all members of our team have experience in Python programming, and we are more confident in Python programming than in other programming languages.

- Express.js
  Express.js is a JavaScript based framework which allows us to implement back-end features such as routing. However, to integrate Express into this project requires more dependencies and complex configurations, which makes implementation more difficult.

Final decision: Python

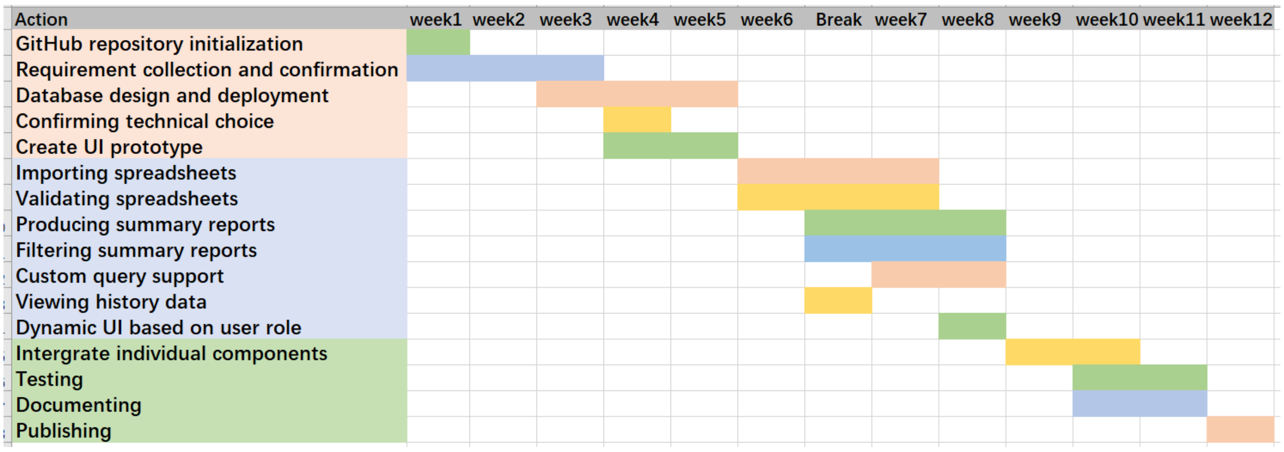
Testing:

- For front-end unit testing, we will use is Jest which is probably the most popular JavaScript testing tool.
- For back-end unit testing, we will use PyUnit which is part of the standard Python library, so that we will not have to install any additional modules to run it.
- For integration testing, we will use Selenium which will be configured to run with Chrome or Firefox to test the acceptance test.

Final decision: Jest, PyUnit, Selenium

7. Project Plan

(Refer to GitHub Unit_Budget_Planner/Group_Meeting/Meeting_Record/ 24082021_Meeting_Minutes)



Action	week1	week2	week3	week4	week5	week6	Break	week7	week8	week9	week10	week11	week12
GitHub repository initialization	🟩												
Requirement collection and confirmation	🟦	🟦	🟦										
Database design and deployment			🟧	🟧	🟧								
Confirming technical choice				🟨									
Create UI prototype				🟩	🟩								
Importing spreadsheets						🟧	🟧	🟧					
Validating spreadsheets						🟨	🟨	🟨					
Producing summary reports							🟩	🟩	🟩				
Filtering summary reports							🟦	🟦	🟦				
Custom query support								🟧	🟧				
Viewing history data							🟨						
Dynamic UI based on user role									🟩				
Intergrate individual components										🟨	🟨		
Testing											🟩	🟩	
Documenting											🟦	🟦	
Publishing													🟧

<u>**Responsibilities and Deadlines**</u>

Importing spreadsheets:
Ethan Chen, Yu Zhu
Start time: 30/08/2021 Deadline: 17/09/2021

Validating spreadsheets:
Ethan Chen, Yu Zhu
Start time: 30/08/2021 Deadline: 17/09/2021

Producing summary reports:
Nara Eam, Lida Tong
Start time: 04/09/2021 Deadline: 22/09/2021

Filtering summary reports:
Nara Eam, Lida Tong
Start time: 04/09/2021 Deadline: 22/09/2021

Custom query support:
Liangbo Jin , Keli Yin
Start time: 11/09/2021 Deadline: 22/09/2021

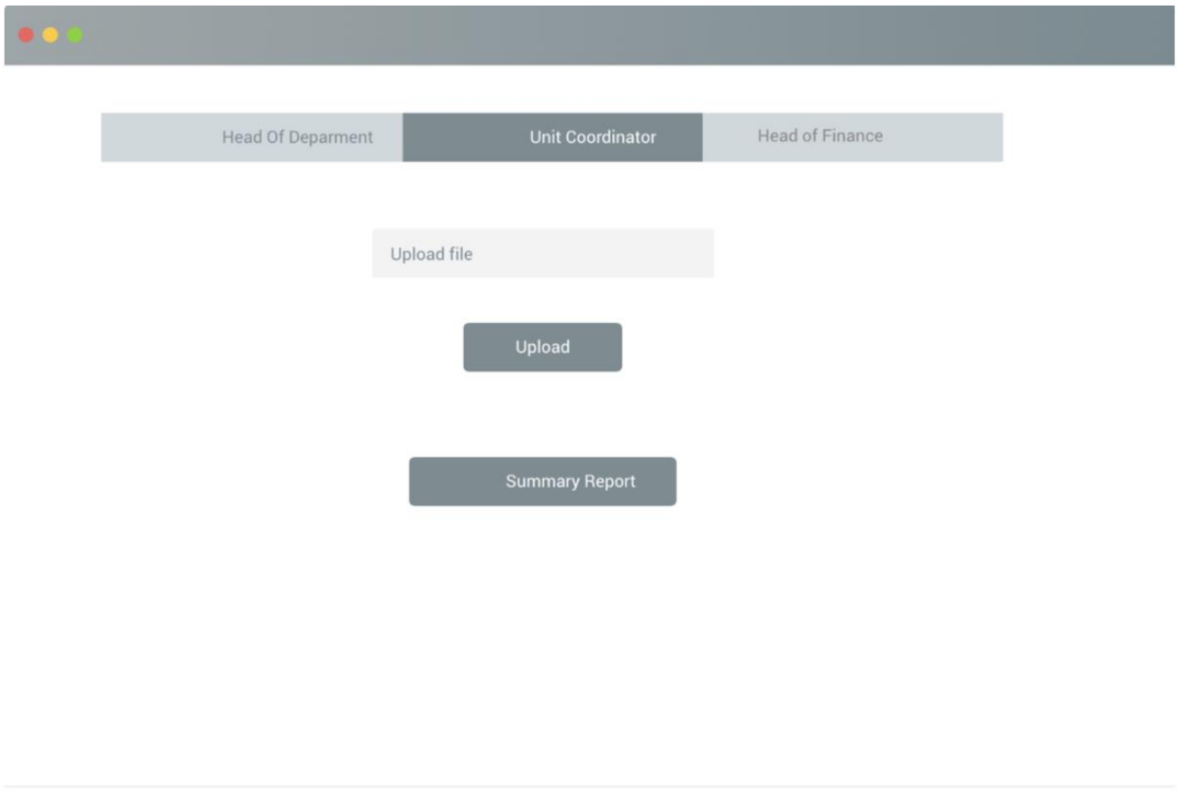
Viewing history data:
Liangbo Jin, Keli Yin
Start time: 04/09/2021 Deadline: 11/09/2021

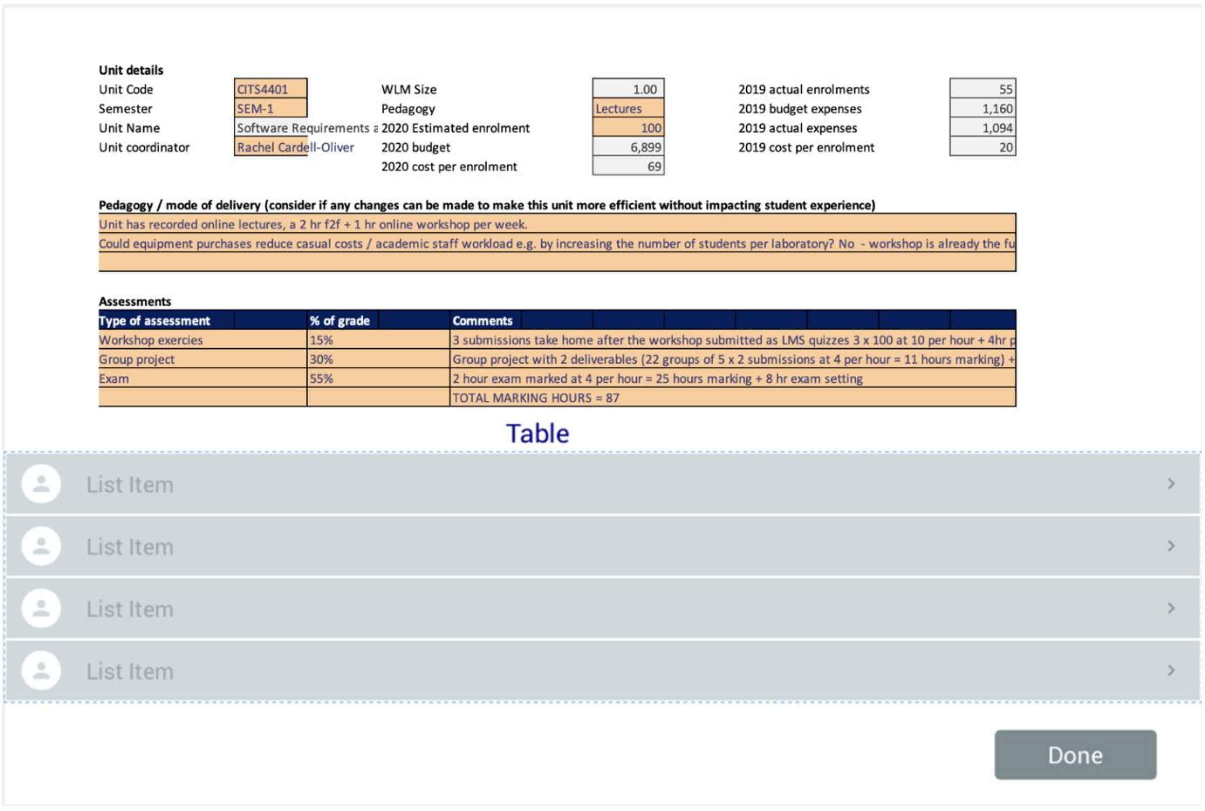
Dynamic UI based on user role
Everyone
Start time: 15/09/2021 Deadline: 22/09/2021

8. Prototypes

8.1 User Interface

(Refer to GitHub Unit_Budget_Planner/Resources/Prototypes.png)





Unit details

Unit Code	CITS4401	WLM Size	1.00	2019 actual enrolments	55
Semester	SEM-1	Pedagogy	Lectures	2019 budget expenses	1,160
Unit Name	Software Requirements a	2020 Estimated enrolment	100	2019 actual expenses	1,094
Unit coordinator	Rachel Cardell-Oliver	2020 budget	6,899	2019 cost per enrolment	20
		2020 cost per enrolment	69		

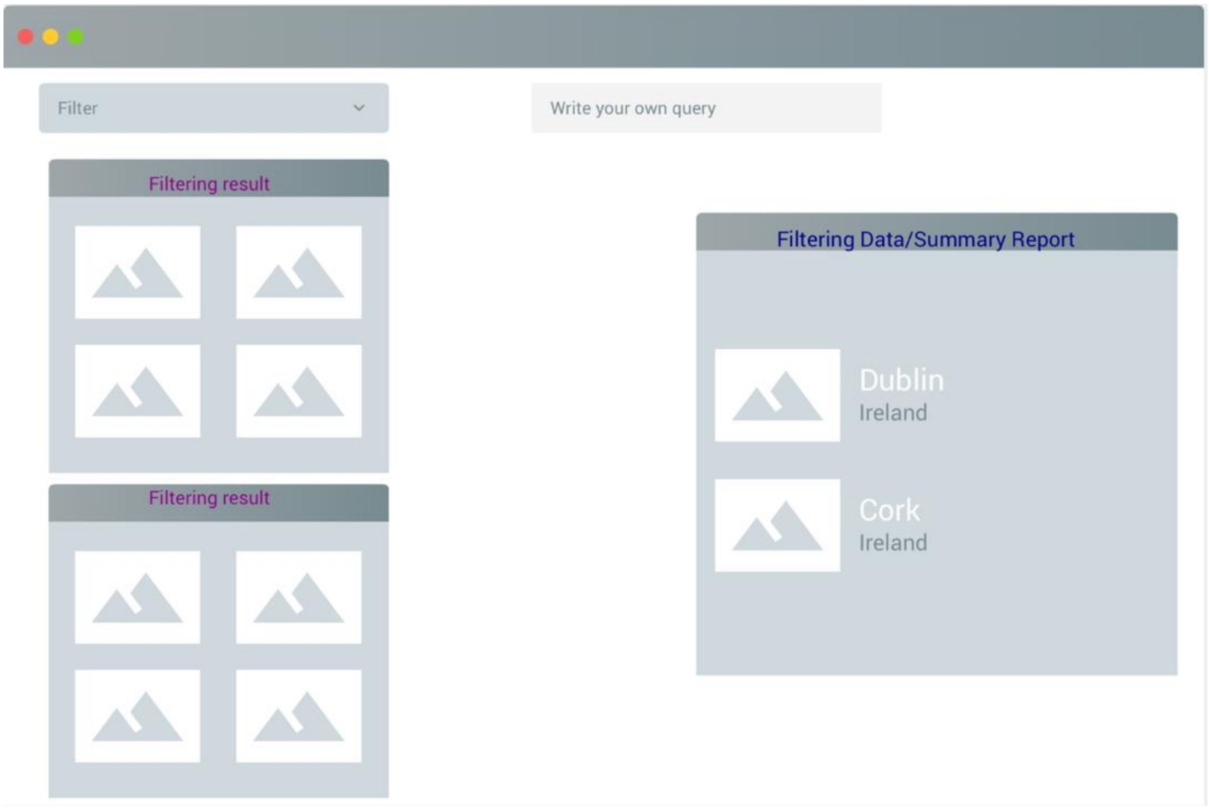
Pedagogy / mode of delivery (consider if any changes can be made to make this unit more efficient without impacting student experience)

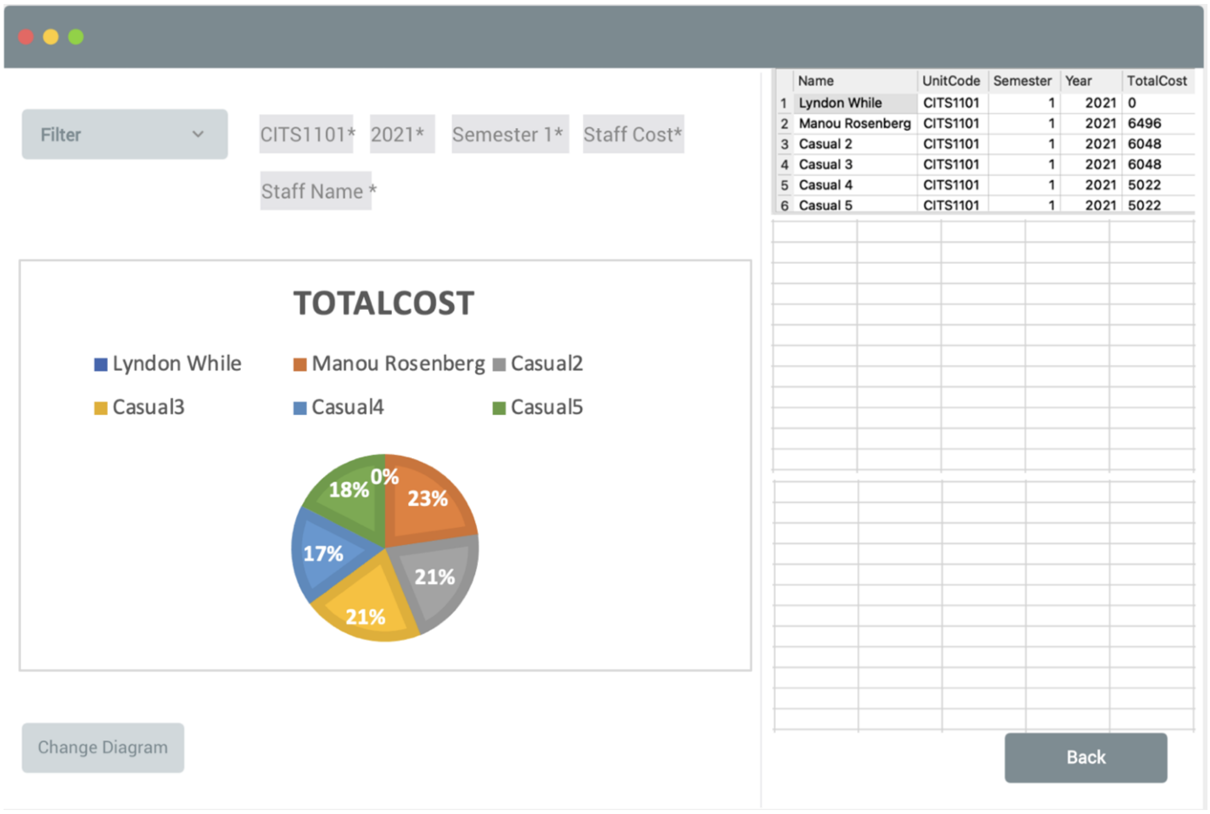
Unit has recorded online lectures, a 2 hr f2f + 1 hr online workshop per week.

Could equipment purchases reduce casual costs / academic staff workload e.g. by increasing the number of students per laboratory? No - workshop is already the fu

Assessments

Type of assessment	% of grade	Comments
Workshop exercies	15%	3 submissions take home after the workshop submitted as LMS quizzes 3 x 100 at 10 per hour + 4hr p
Group project	30%	Group project with 2 deliverables (22 groups of 5 x 2 submissions at 4 per hour = 11 hours marking) +
Exam	55%	2 hour exam marked at 4 per hour = 25 hours marking + 8 hr exam setting
		TOTAL MARKING HOURS = 87





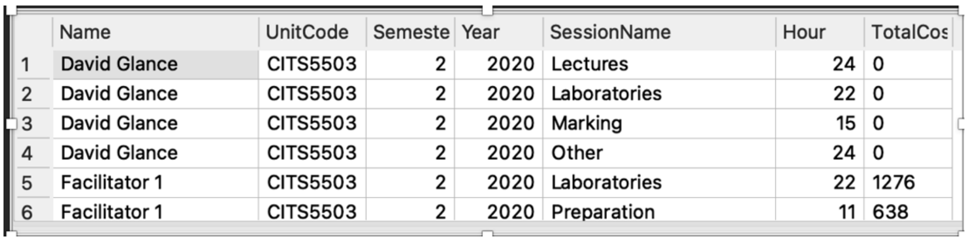
	Name	UnitCode	Semester	Year	TotalCost
1	Lyndon While	CITS1101	1	2021	0
2	Manou Rosenberg	CITS1101	1	2021	6496
3	Casual 2	CITS1101	1	2021	6048
4	Casual 3	CITS1101	1	2021	6048
5	Casual 4	CITS1101	1	2021	5022
6	Casual 5	CITS1101	1	2021	5022

8.2 Queries for Database

(Refer to GitHub Unit_Budget_Planner/Database_Structur/Sample\Queries.docx)

1. List all staffs in all units, as well as their Total cost and hours that spend on each session during the whole semester

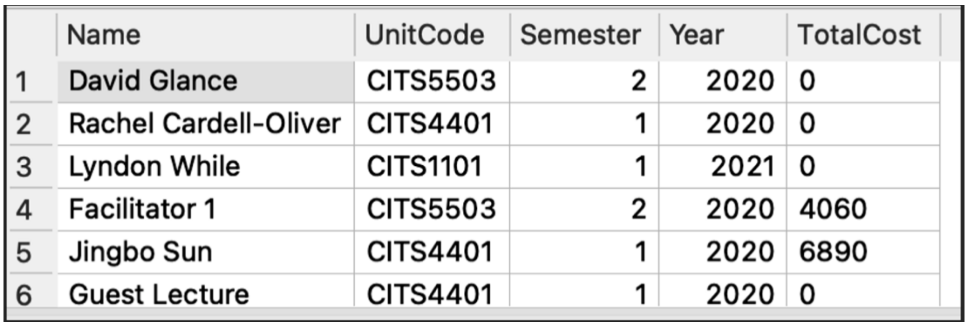
```
Select S.Name,U.UnitCode, U.Semester, U.Year, E.SessionName,
A.Hour,A.HourlyRate*A.Hour AS TotalCost
From Activities A JOIN Staff S USING (StaffID)
                  JOIN Session E USING (SessionID)
                  JOIN Unit U USING (UnitID)
```



	Name	UnitCode	Semeste	Year	SessionName	Hour	TotalCos
1	David Glance	CITS5503	2	2020	Lectures	24	0
2	David Glance	CITS5503	2	2020	Laboratories	22	0
3	David Glance	CITS5503	2	2020	Marking	15	0
4	David Glance	CITS5503	2	2020	Other	24	0
5	Facilitator 1	CITS5503	2	2020	Laboratories	22	1276
6	Facilitator 1	CITS5503	2	2020	Preparation	11	638

2. List the Total Cost for each Staff in one Unit

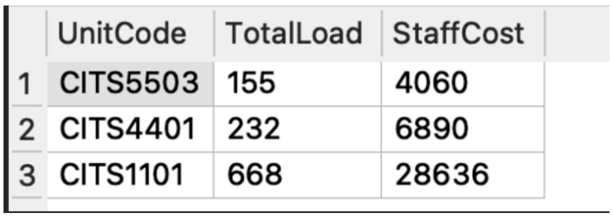
```
Select S.Name, U.UnitCode, U.Semester, U.Year, SUM(A.Hour*A.HourlyRate) AS TotalCost
From Activities A JOIN Staff S USING (StaffID)
                  JOIN Session E USING  (SessionID)
                  JOIN Unit U USING (UnitID)
Group by S.StaffID
```



	Name	UnitCode	Semester	Year	TotalCost
1	David Glance	CITS5503	2	2020	0
2	Rachel Cardell-Oliver	CITS4401	1	2020	0
3	Lyndon While	CITS1101	1	2021	0
4	Facilitator 1	CITS5503	2	2020	4060
5	Jingbo Sun	CITS4401	1	2020	6890
6	Guest Lecture	CITS4401	1	2020	0

3. List the total Staff cost and workload for each Unit

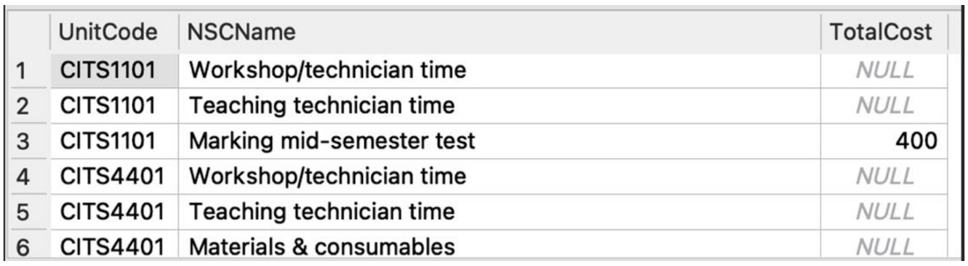
```
Select U.UnitCode, SUM(A.Hour) AS TotalLoad, SUM(A.Hour * A.HourlyRate) AS
StaffCost
From Activities A JOIN Staff S USING (StaffID)
                  JOIN Session E USING (SessionID)
                  JOIN Unit U USING (UnitID)
Group By U.UnitID
```



	UnitCode	TotalLoad	StaffCost
1	CITS5503	155	4060
2	CITS4401	232	6890
3	CITS1101	668	28636

4. List all the Non-Salary Costs for each Unit

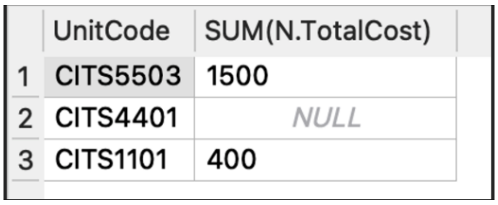
```
Select U.UnitCode,N.NSCName, N.TotalCost
From OtherCost O JOIN NonSalaryCosts N USING (NSCID)
            JOIN Unit U USING (UnitID)
 Order by U.UnitCode
```



	UnitCode	NSCName	TotalCost
1	CITS1101	Workshop/technician time	NULL
2	CITS1101	Teaching technician time	NULL
3	CITS1101	Marking mid-semester test	400
4	CITS4401	Workshop/technician time	NULL
5	CITS4401	Teaching technician time	NULL
6	CITS4401	Materials & consumables	NULL

5. List the Total Non-Salary Cost for each Unit

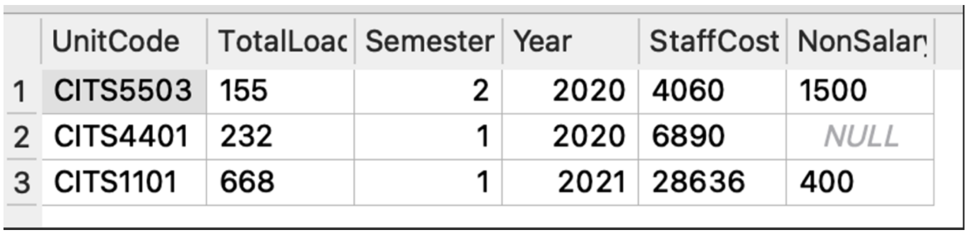
```
Select U.UnitCode, SUM(N.TotalCost)
From OtherCost O JOIN NonSalaryCosts N USING (NSCID)
        JOIN Unit U USING (UnitID)
Group by U.UnitID
```



	UnitCode	SUM(N.TotalCost)
1	CITS5503	1500
2	CITS4401	NULL
3	CITS1101	400

6. List the Total Non-Salary Cost and Staff Cost for each Unit

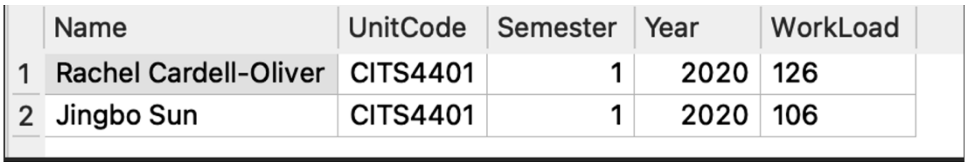
```
Select U.UnitCode, SUM(A.Hour) AS TotalLoad, U.Semester,U.Year,SUM(A.Hour *
A.HourlyRate) AS StaffCost,
(Select SUM(N.TotalCost)
From OtherCost O JOIN NonSalaryCosts N USING (NSCID)
        JOIN UNIT Z USING (UnitID)
Where Z.UnitID = U.UnitID
Group by Z.UnitID) AS NonSalaryCost
From Activities A JOIN Staff S USING (StaffID)
              JOIN Session E USING (SessionID)
              JOIN Unit U USING (UnitID)
Group By U.UnitID
```



	UnitCode	TotalLoac	Semester	Year	StaffCost	NonSalar
1	CITS5503	155	2	2020	4060	1500
2	CITS4401	232	1	2020	6890	NULL
3	CITS1101	668	1	2021	28636	400

7. List all Staffs, and their teaching unit in 2020, where their workload is greater than 100 hours

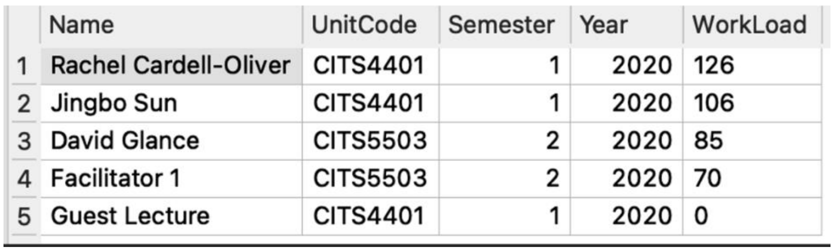
```
Select S.Name, U.UnitCode, U.Semester,U.Year,SUM(A.Hour) AS WorkLoad
From Activities A JOIN Staff S USING (StaffID)
              JOIN Unit U USING (UnitID)
              JOIN Session E USING (SessionID)
Where U.Year = 2020
Group by S.StaffID
Having WorkLoad >= 100
```



	Name	UnitCode	Semester	Year	WorkLoad
1	Rachel Cardell-Oliver	CITS4401	1	2020	126
2	Jingbo Sun	CITS4401	1	2020	106

8. List all staffs and their workload in the units in 2020, and rank the workload from high to low

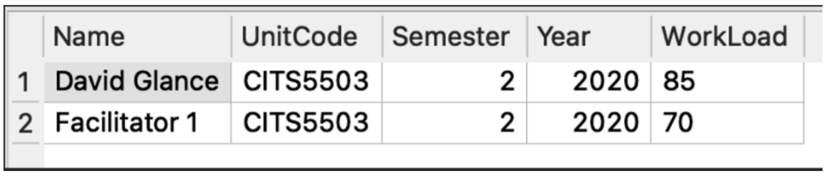
```
Select S.Name, U.UnitCode, U.Semester,U.Year,SUM(A.Hour) AS WorkLoad
From Activities A JOIN Staff S USING (StaffID)
                  JOIN Unit U USING (UnitID)
                  JOIN Session E USING (SessionID)
Where U.Year = 2020
Group by S.StaffID
Order by WorkLoad DESC
```



	Name	UnitCode	Semester	Year	WorkLoad
1	Rachel Cardell-Oliver	CITS4401	1	2020	126
2	Jingbo Sun	CITS4401	1	2020	106
3	David Glance	CITS5503	2	2020	85
4	Facilitator 1	CITS5503	2	2020	70
5	Guest Lecture	CITS4401	1	2020	0

9. List the staff that teaches CITS5503 in 2020 semester 2, and show their total workload

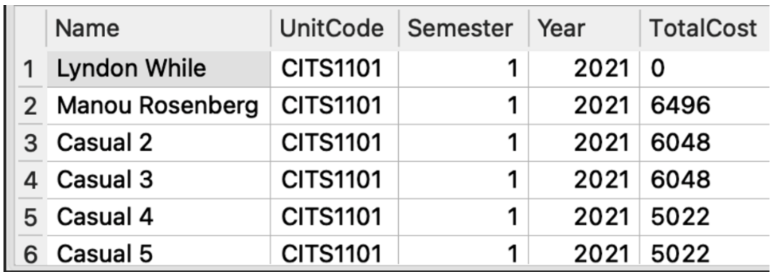
```
Select S.Name, U.UnitCode, U.Semester,U.Year,SUM(A.Hour) AS WorkLoad
From Activities A JOIN Staff S USING (StaffID)
                  JOIN Unit U USING (UnitID)
                  JOIN Session E USING (SessionID)
Where U.Year = 2020 and U.UnitCode = "CITS5503" and U.Semester = 2
Group by S.StaffID
```



	Name	UnitCode	Semester	Year	WorkLoad
1	David Glance	CITS5503	2	2020	85
2	Facilitator 1	CITS5503	2	2020	70

10. List the unit CITS1101 in 2021 semester 1, and show their staff, as well as their budget cost

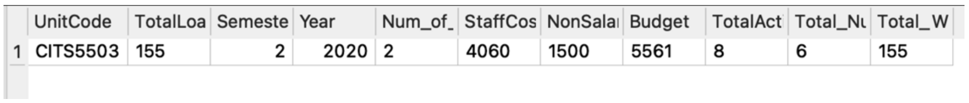
```
Select S.Name, U.UnitCode, U.Semester, U.Year, SUM(A.Hour*A.HourlyRate) AS TotalCost
From Activities A JOIN Staff S USING (StaffID)
                  JOIN Session E USING (SessionID)
                  JOIN Unit U USING (UnitID)
Where U.UnitCode = "CITS1101" and U.Year = 2021 and U.Semester = 1
Group by S.StaffID
```



	Name	UnitCode	Semester	Year	TotalCost
1	Lyndon While	CITS1101	1	2021	0
2	Manou Rosenberg	CITS1101	1	2021	6496
3	Casual 2	CITS1101	1	2021	6048
4	Casual 3	CITS1101	1	2021	6048
5	Casual 4	CITS1101	1	2021	5022
6	Casual 5	CITS1101	1	2021	5022

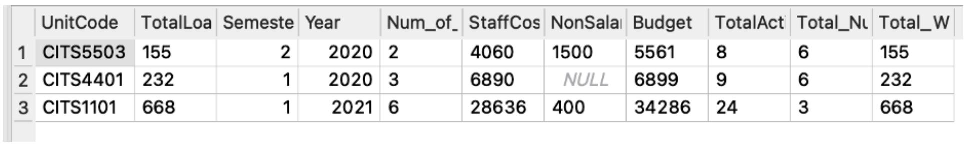
11. Print the sample summary for CITS 5503 in 2020 Semester 2 (We can add more data if required)

```
Select U.UnitCode, SUM(A.Hour) AS TotalLoad, U.Semester,U.Year,
(Select COUNT(DISTINCT P.Name)
From Activities A JOIN Staff P USING (StaffID)
JOIN Unit R USING (UnitID)
Where R.UnitCode = U.UnitCode
) AS Num_of_Staff,
SUM(A.Hour * A.HourlyRate) AS StaffCost,
(Select SUM(N.TotalCost)
From OtherCost O JOIN NonSalaryCosts N USING (NSCID)
JOIN UNIT Z USING (UnitID)
Where Z.UnitID = U.UnitID
Group by Z.UnitID) AS NonSalaryCost,
(Select B.Cost
From Unit G JOIN Budget B USING (UnitID)
Where IsEstimated = "YES" and B.IsLastSemester = "NO" and G.UnitID = U.UnitID ) AS
Budget,
(Select COUNT(*)
From Activities A JOIN Unit P USING (UnitID)
Where P.UnitID = U.UnitID
Group by P.UnitID) AS TotalActivities,
(Select COUNT(*)
From OtherCost O JOIN Unit L USING (UnitID)
JOIN NonSalaryCosts USING (NSCID)
Where L.UnitID = U.UnitID
Group by L.UnitID) AS Total_Number_of_NSC,
(Select SUM(A.Hour)
From Activities A JOIN Unit N USING (UnitID)
Where N.UnitID = U.UnitID
Group by N.UnitID) AS Total_WorkLoad
From Activities A JOIN Staff S USING (StaffID)
                    JOIN Session E USING (SessionID)
                    JOIN Unit U USING (UnitID)
Where U.UnitCode = "CITS5503" and U.Year = 2020 and U.Semester = 2
Group By U.UnitID
```



	UnitCode	TotalLoa	Semeste	Year	Num_of_	StaffCos	NonSala	Budget	TotalAct	Total_Nu	Total_W
1	CITS5503	155	2	2020	2	4060	1500	5561	8	6	155

Or all units in all year semester



	UnitCode	TotalLoa	Semeste	Year	Num_of_	StaffCos	NonSala	Budget	TotalAct	Total_Nu	Total_W
1	CITS5503	155	2	2020	2	4060	1500	5561	8	6	155
2	CITS4401	232	1	2020	3	6890	NULL	6899	9	6	232
3	CITS1101	668	1	2021	6	28636	400	34286	24	3	668