UOW MALAYSIA KDU PENANG UNIVERSITY COLLEGE
PART OF THE UNIVERSITY OF WOLLONGONG AUSTRALIA GLOBAL NETWORK

Validated by UNIVERSITY OF LINCOLN UNITED KINGDOM

Assignment Cover Page

Programme	**Course Code and Title**
Bachelor of Computer Science (Hons),	CSE3033/N Software Engineering
Student's name / student's id	**Lecturer's name**
1. 0204677 Lim Zhe Yuan 2. 0203787 Lee Chee Yew 3. 0203640 Otheya Kumar A/L Chandramohan	Tan Phit Huan

Date issued	**Submission Deadline**	**Indicative Weighting**
Week 2 – 6 Feb 2023	Week 11 – 14 April 2023	30%
Assignment title	Assignment 2	

This assessment assesses the following course learning outcomes

# as in Course Guide	**UOWM KDU Penang University College Learning Outcome**
CLO1	Apply concepts derived from current theories of advanced software engineering.

# as in Course Guide	**University of Lincoln Learning Outcome**
LO1	Synthesise concepts derived from current theories of advanced software engineering
LO3	Utilise and evaluate advanced software engineering techniques and processes in the development of a software artefact

Student's declaration

We certify that the work submitted for this assignment is our own and research sources are fully acknowledged.

Student's signature: Date:

Zhe Yuan
CY
Otheya

14/4/2023

TurnItIn Similarity Report

CSEAsgn2

ORIGINALITY REPORT

7%	0%	0%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to UOW Malaysia KDU University College Sdn. Bhd Student Paper	4%
2	Submitted to KDU College Sdn Bhd Student Paper	2%
3	Submitted to University of South Australia Student Paper	1%
4	ia802309.us.archive.org Internet Source	<1%
5	Submitted to Online Education Services Student Paper	<1%
6	Submitted to Huddersfield New College Student Paper	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

Table Of Content

Main Report

Section 1: Introduction to the Apps, Product Backlog and User Stories

The online timetable scheduling system that had been proposed by the team includes features that facilitate the management of schedules for roles such as students, school administrators and co-curriculum leaders. The system also provides sub-features that streamlines academic processes and make them easier, such as archiving, documenting, or searching for information about school personnels, aside from main features for time tabling. These system features were proposed based on the requirements of an online timetable scheduling system gathered from online sources.

While there are recommendations for normal time tabling features online, information on sub-features that could be implemented in the system remains scarce. However, several expert advices have been obtained online to derive and develop these sub-features for the timetable scheduling system. According to an Edecofy admin (2021), timetable scheduling systems should strive for features that enable smart scheduling, better communication, easy installation, and seamless integrations. Another Sweedu.com post (2022) mostly agree with the statement, but also adds that these systems should improve task productivity, efficiency, seamlessness, and modification abilities as well.

As a result, the team came up with newer, creative ideas for the proposed timetable scheduling system based on these advices during the planning phase. The following list shows the software requirements that have been identified and can be implemented for the system.

Software requirements

- **Students**

1. Login function and student authentication.
2. View and manage personalized schedules for classes, assignments and exams.
3. Search and filter courses based on preferences such as course level, semesters and personal preferences.
4. Being notified of reminders of upcoming events, exams and events.
5. Access to class materials and resources like lecture notes, assignments and video recordings.
6. A user friendly interface that is easy to understand and navigate.
7. Integration with other educational or examination systems such as Open Learning (OL) or Exam Portals.
8. Option to make changes to existing schedules or add new courses or events.
9. Integration of support for multilingual systems to accommodate international students.
10. Option to provide feedback to lecturer or event organizers about clashing dates.

- **Administrators**

1. Login function and admin authentication.
2. Option to assign lecturers or event managers to courses and events.
3. User management system to create and manage user roles and permissions
4. Course management system to insert, update and delete course information
5. Notifications of upcoming events, exams and events are sent to lecturers and event managers.
6. Option to generate reports based on course schedules, attendance records and student feedback.
7. Option to insert or remove users and their profiles.
8. Access control and user security to ensure protection of user data and system information.
9. Scheduling system to create and manage class schedules, which includes date, location and lecturer information.
10. Integration with other educational or course systems such as Open Learning (OL) or Canvas.

- **Co-curriculum leaders**

1. Add co-curriculum activities to timetables of society members.
2. Edit co-curriculum activity details.
3. Remove co-curriculum activities.
4. Schedule pre-emptive activities instantly.
5. Identify society members registered with the society.
6. Suggest appropriate activity time range based on member's weekly timetables.
7. Automatically remind society members about upcoming activities.
8. Send notifications to society members.
9. Identify society members that have conflicting timetables with society activities.
10. Report misbehaving society members to school admins.

After careful consideration, the requirements are then inserted into a backlog which is split into several sprints due to the massive amount of requirements gathered to develop the system. This is done to reduce developer workloads at any given point in time and allow them to work on features incrementally. It is estimated that three sprints are sufficient to develop most of the requirements, which allocates 3 weeks and a total of 120 working hours per sprint. The following shows the backlog created for the gathered requirements and the sprints required to develop them:

Sprint 1 (Duration of 3 weeks [8 hours x 15 days = 120 hours])

Backlog Items	Story Points (Size)	Day 0 (Estimate
As an administrator, I want to be able to access the user management system so that I can create and manage user roles and permissions.	XL	
Task 1A Design UI		2
Task 1B Code function		3
Task 1C Implement into system		1
As an administrator, lecturer and event leader, I want to be able to access the course management system so that I can insert, update and delete course information.	XL	
Task 2A Design UI		2
Task 2B Code function		3
Task 2C Implement into system		1
As an administrator, I want to be able to assign lecturers and event leaders to courses and events so that lecturers and event leaders can be assigned to their respective courses and events.	L	
Task 3A Design UI		2
Task 3B Code function		3
Task 3C Implement into system		1
As an administrator, I want to be able to provide access levels to all users based on their role in the system so that I am able to provide them access to material and systems based on their role.	XL	
Task 4A Collect list of users		2
Task 4B Code function		3
Task 4C Implement into system		1
As an administrator, I want to be able to add and remove users and their profiles so that I can monitor the status of each user and their profiles.	M	
Task 5A Collect list of users		1
Task 5B Code function		1
Task 5C Implement into system		1
As an administrator, I want to implement a 2-step authentication system for login function and encryption of sensitive data so that the safety of users can be ensured.	M	
Task 6A Design UI		1

Task 6B Code function		2
Task 6C Implement into system		2

Table 1: *Sprint 1 backlog*

Sprint 2 (Duration of 3 weeks [8 hours x 15 days = 120 hours])

Backlog Items	Story Points (Size)	Day 0 (Estimate
As an administrator, I would like to implement a user feedback system so that users are able to provide feedback about their clashing schedules or problems present in the system that they have found.	M	
Task 1A Design UI		3
Task 1B Code function		3
Task 1C Implement into system		2
As an event leader, I want to be able to access the event management system so that I can add, update and remove event schedules and their details.	L	
Task 2A Design UI		2
Task 2B Code function		3
Task 2C Implement into system		1
As an event leader, I want to be able to identify social members registered with the society so that I can social members can be easily identified for attendance.	M	
Task 3A Collect list of members		1
Task 3B Code function		1
Task 3C Implement into system		1
As a lecturer, I want to be able to use a system that generate reports based on course schedules, attendance records and student feedback so that I can document those reports to the higher ups in the school.	L	
Task 4A Design UI		2
Task 4B Code function		3
Task 4C Implement into system		1
Improve scheduling system: Adaptive scheduling based on user availability or level.	S	
Task 5A Discuss new requirements		3
Task 5B Design UI		3
Task 5C Implement into system		2

Improvement to system. By implementing personalized functionalities based on preferences, users are able to use the system much more easily and with a higher efficiency with personalized functions.	S	
Task 6A Design UI		3
Task 6B Code function		3
Task 6C Implement into system		2

Table 2: *Sprint 2 backlog*

Sprint 3 (Duration of 3 weeks [8 hours x 15 days = 120 hours])

Backlog Items	Story Points (Size)	Day 0 (Estimate
As a user, I want to be able to have a discussion between users about the schedules of each member so that I can ensure proper scheduling between members without conflict.	M	
Task 1A Design UI		2
Task 1B Code function		2
Task 1C Implement into system		1
As a user, I want to be able to have data backup and disaster recovery programs so that in the case of an attack by malicious parties or system corruption, systems are in place in order to protect user data and system information.	XL	
Task 2A Design UI		5
Task 2B Code function		5
Task 2C Implement into system		3
As a user, I would like to be notified by a notification system so that I can be informed of upcoming important events, exams and cancellations.		
Task 3A Design UI		2
Task 3B Code function		2
Task 3C Implement into system		1
Improve user interface: User friendliness and personalization	S	
Task 4A Discuss new requirements		4
Task 4B Design UI		2
Task 4C Implement into system		1

As a user, I would like to have more interactions that offer functionalities with other system so that I can execute tasks that are outside of the main purpose of the system.	M	
Task 5A Design UI		5
Task 5B Code function		3
Task 5C Implement into system		1
As a user, I would like to have the option to choose different languages in the system so that I use the system more comfortably with the preferred language of my choice outside of English.	S	
Task 6A Design UI		3
Task 6B Code function		2
Task 6C Implement into system		1

Table 3: *Sprint 3 backlog*

Section 2: Class Diagram Design

The following class diagram shows a conceptual structure of class relationships that can be adopted by the timetable scheduling system. It efficiently uses inheritance, and the overall cardinality of the diagram shows that many-to-many relationships have been drastically eliminated. Each class has high cohesion and unrelated classes are loosely coupled.

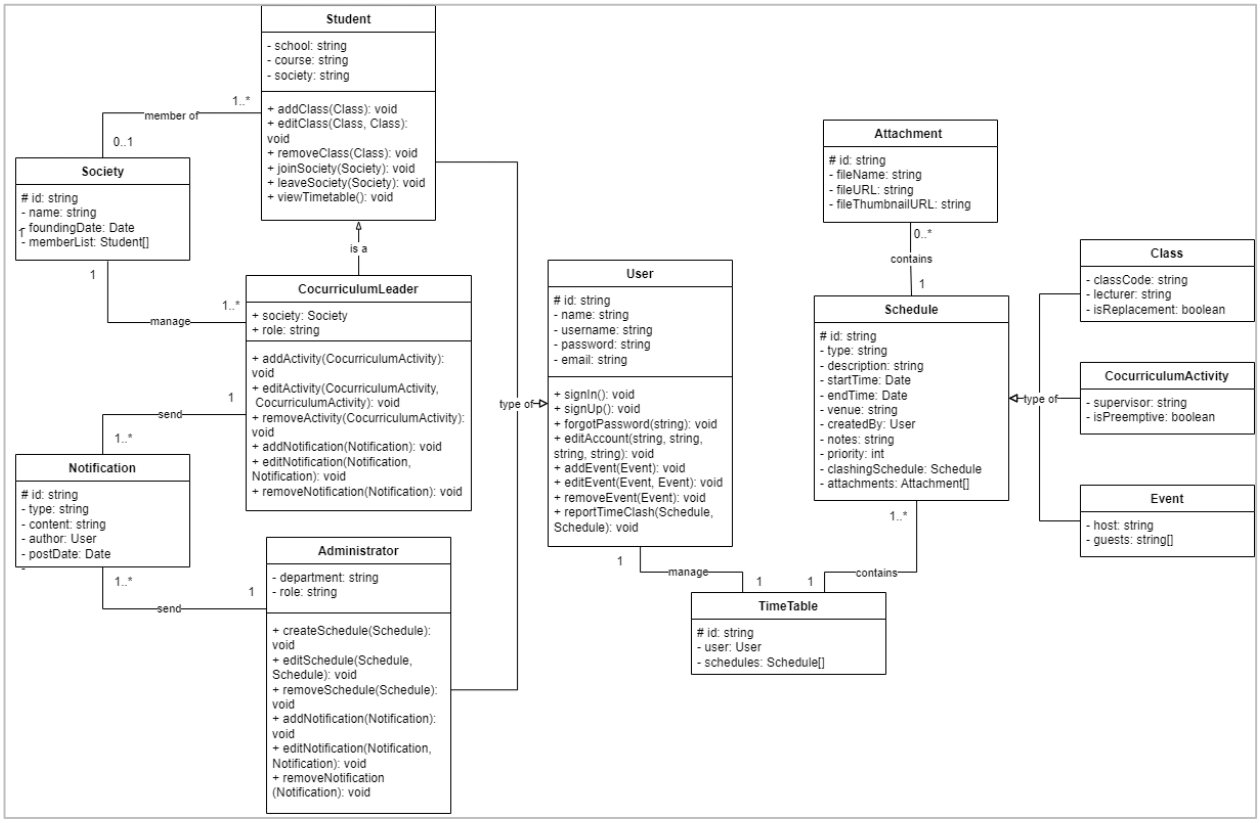

Figure 1: *Class diagram for timetable scheduling system*

Section 3: Software Evolution

Obtained feedback

The concept of the online timetable scheduling system was presented to the targeted users and their feedback have been obtained as well as the user stories have been developed for them to have a better understanding of the features available in the online timetable scheduling system. The feedback received are very important as they will be used to improve the user experience as well as to create an easily usable software experience for the users so that it will not be a chore while using the software. The user's feedback will also be able to help validate whether the features implemented meet the users need which will ensure the value of this software to the users.

For students, the first feedback that has been received is that students should be able to check for time conflict so a time conflict checker feature will be helpful for students to avoid schedule conflicts. This feature allows students to check for any conflicts that might turn up in their schedule due to overlapping course schedules or other activities and the time conflict checker will also be able to suggest alternative class timing that may be a better choice for the student's schedule. This feature is essential in improving the overall learning experience of students by reducing stress and enabling them to plan their schedules better.

Another feedback that was received from the students is they need a feature where the software will be able to recommend courses based on a student's academic history, interests and learning preferences. This feature will be able to help students discover new courses that align with their academic goals and improve their learning outcomes while also enhancing student's engagement, reduce dropout rates and improve overall academic performance.

For administrators, the first received feedback is that it would be useful to have a room booking management feature that allows the administrators to manage and book rooms for classes, meetings and events which streamlines the scheduling process while reducing errors and conflicts as well as increasing efficiency. The room booking management feature ensures that rooms are allocated appropriately while avoiding conflicts to reduce the workload of administrators by automating the booking process.

Another feedback that was received from the administrators is that they need a way to track and manage attendance records for courses and events. This feature will be able to monitor students' attendance and identify patterns and trends which then it can generate a report for academic purposes. This attendance management feature ensures that students attend classes regularly and provides insights into their academic progress which is essential in improving academic performance and ensuring that students meet the requirements for academic credits.

For co-curriculum leaders, the first feedback is that co-curriculum leaders are responsible for managing and tracking member engagement in various activities, so they need a performance tracking feature that allows co-curriculum leaders to track and manage the performance of society members in various activities. This feature also enables them to monitor member engagement, identify areas of improvement and provide constructive feedback to ensure that members are engaged and motivated as well as co-curricular activities are organized effectively.

The second feedback that co-curriculum leaders gave is the need for a volunteer feature that allows co-curriculum leaders to recruit and manage volunteers for various events and

activities. This feature will be able to streamline the recruitment process, keeps track of volunteers and ensures the smooth running of events as well as ensuring the volunteers are assigned to suitable activities while their contributions are recognized so that the volunteers will also get gratitude for helping out.

Perfective Changes

The first perfective change that will be made to the software is to not only generate reports based on course schedules and attendance records but can also manage the attendance and keep track of them as co-curriculum leaders and administrators will be able to take attendance and generate reports of attendance records easily. The user will be able to select the event or activity for which they want to take attendance from a drop-down menu and take attendance by marking the members who are present and absent will be automatically saved to the database system so that it can be accessed by the user at any time.

The next perfective change is to reduce the number of steps taken to access multiple major features in the software which will provide users with a quicker and easier way to access the major features. For instance, the software will be able to allow students to create and save personalized study plans based on their schedules which would be convenient for the students as there can be multiple study sessions and all they would need to do is just access the saved study plans in a click. By introducing these perfective changes, users will be able to access multiple major features in fewer steps which helps in reducing the time and effort required to perform tasks as this will improve the overall user experience and increase productivity thus making it easier for users to achieve their goals within the application.

The final perfective change is to overhaul the user interface design of the software which is aimed at improving the software's usability and enhancing the users overall experience and this perfective change involves a few redesigns of the visual layout and the user experience (UX) of the system. Some users have also given feedback and helped identify areas that need further improvement to ensure that the new design is optimized for the user’s needs such as simplifying the navigations, implementing new designs, fix certain UI bugs and more. Overhauling the UI design of the software can also significantly improve the software's usability, accessibility as well as user experience and by incorporating user feedbacks to optimize the design for the user’s needs, the software can become more user-friendly and more effective in meeting its intended purpose.

Section 4: Software Testing

The following section shows the validation processes to exhaustively test some of the prominent requirements needed to develop the timetable scheduling system using white box and black box testing:

1. Create and manage user roles and permissions.

Black Box Testing:

Input Condition	Valid Equivalence Class	Valid Boundary	Invalid Equivalence Class	Invalid Boundary
Name				
type	String	Auditor [1]	Others	1 [6]
Length	>1 [2] <20 [3]		<1 [7] >20 [8]	
Permissions				
type	String array [4] Each content must be one of: administrator, view_timetable, view_user_info, modify_timetable, modify_user, modify_schedule, manage_ notifications, manage_society,	[“administrator”, “view_timetable”, “view_user_info”, “modify_timetable”, “modify_user”, “modify_schedule”, “manage_notifications”, “manage_society”] [5]	Empty array Not a string array Other array content	[] [9] [1,2,3] [10] [“modify_system”] [11]

Valid Test Cases

Name	Permissions	Boundaries Covered
Auditor	[“administrator”, “view_timetable”, “view_user_info”, “modify_timetable”, “modify_user”, “modify_schedule”,	1, 2, 3, 4, 5

	“manage_notifications”, “manage_society”]	

Invalid Test Cases

Name	Permissions	Boundaries Covered
1	[]	6
	[]	7
Administratorismyresponsibility	[]	8
Administrator	[]	9
Administrator	[1,2,3]	10
Administrator	[“modify_system”]	11

White Box Testing:

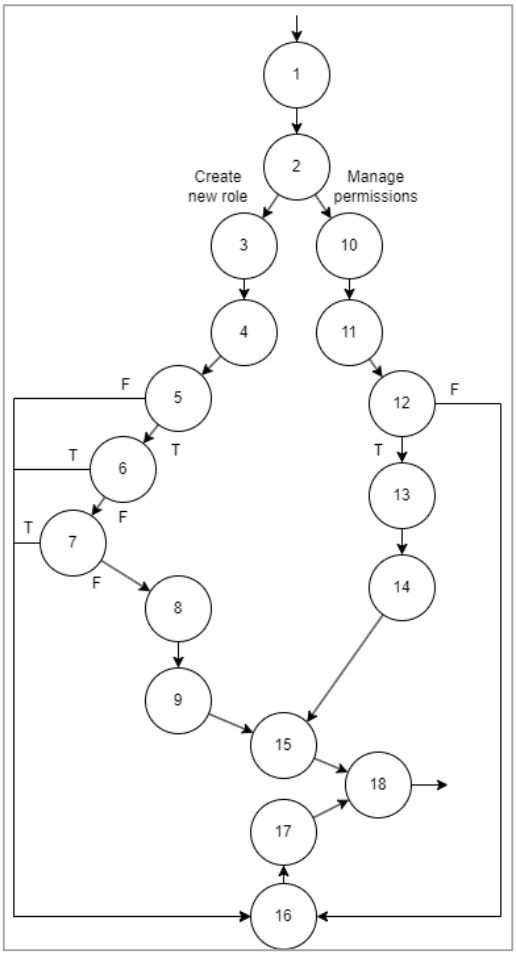

Figure 2: *Control flow graph for creating and managing user roles and permissions*

General code
1 - Get user choice
2 - Decide user choice to create or manage user roles and permissions.
15 - Inquire if user want to continue operations
16 - Capture and notify error to user
17 - Enable the input form for user modifications
18 - Ending connector

Scope: Create new roles
3 - Show role creation interface
4 - Initialize and store information inputs for new role
5 - Check whether inputs are valid
6 - Check whether role already exist
7 - Check whether database capacity is full
8 - Create new user role based on inputs
9 - Update system user interface to include new user role

Scope: Manage role permissions
10 - Show role permissions manager interface and show initial permission values
11 - Initialize and store edited permission values
12 - Check whether edited inputs are valid
13 - Edit user role permissions based on new inputs
14 - Update system user interface to include user role changes

Cyclomatic complexity = 6

Independent paths =
a - [1-2-3-4-5-6-7-8-9-15-18]
b - [1-2-3-4-5-16-17-18]
c - [1-2-3-4-5-6-16-17-18]
d - [1-2-3-4-5-6-7-16-17-18]
e - [1-2-10-11-12-13-14-15-18]
f - [1-2-10-11-12-16-17-18]

Valid test case

- User choice to create new role, all inputs are valid, role was inexistent, role database is not full.

Independent path taken: **a**
Expected output: New role is created and user is prompted whether they want to continue operations or not.

- User choice to manage existing role permissions, all edited inputs are valid.

Independent path taken: **e**
Expected output: Role permissions are updated to new values and user is prompted whether they want to continue operations or not.

Invalid test case

- User choice to create new role, some inputs are invalid, role was inexistent, role database is not full.

Independent path taken: **b**
Expected output: Error message is shown at the affected input components to notify the user of invalid inputs and the input form is re-enabled.

- User choice to create new role, all inputs are valid, role already exists, role database is not full.

Independent path taken: **c**
Expected output: Error message is shown to notify the user about the existence of the old role and the input form is re-enabled.

- User choice to create new role, all inputs are valid, role was inexistent, role database is full.

Independent path taken: **d**
Expected output: Error message is shown to notify the user about insufficient storage of the role database and the input form is re-enabled.

- User choice to manage existing role permissions, some edited inputs are invalid.

Independent path taken: **f**
Expected output: Error message is shown at the affected edit components to notify the user of invalid inputs and the edit form is re-enabled.

2. Create and manage events, event schedules and event details.

Black Box Testing:

Input Condition	Valid Equivalence Class	Valid Boundary	Invalid Equivalence Class	Invalid Boundary
Title				
Type	String [1]		Others [10]	
Length	>=1 [2] <=200 [3]		<1 >200	0 [11] 201 [12]
Date & Time				
Type	Date [4]		Others [13]	
Range	Any date later or equal to current date [5]		Other dates [14]	
Author				
ID	ID of existing user [6]		ID of non-existing user [15]	
Notes				
Type	String [7]		Others [16]	
Length	>=1 [8] <=200 [9]		<1 >200	0 [17] 201 [18]

Valid Test Cases

Title	Date & Time	Author	Notes	Boundaries Covered
Going to Zoo	12/12/2023	0204677	Bring jacket	1,2,3,4,5,6,7,8,9

Invalid Test Cases

Title	Date & Time	Author	Notes	Boundaries Covered
123	12/12/2023	0204677	Bring jacket	10
	12/12/2023	0204677	Bring jacket	11
(name exceeding 200 characters)	12/12/2023	0204677	Bring jacket	12
Going to Zoo	Abc	0204677	Bring jacket	13
Going to Zoo	12/12/2022	0204677	Bring jacket	14
Going to Zoo	12/12/2023	Abc1234	Bring jacket	15
Going to Zoo	12/12/2023	0204677	123	16
Going to Zoo	12/12/2023	0204677		17

Going to Zoo	12/12/2023	0204677	(notes exceeding 200 characters)	18

White Box Testing:

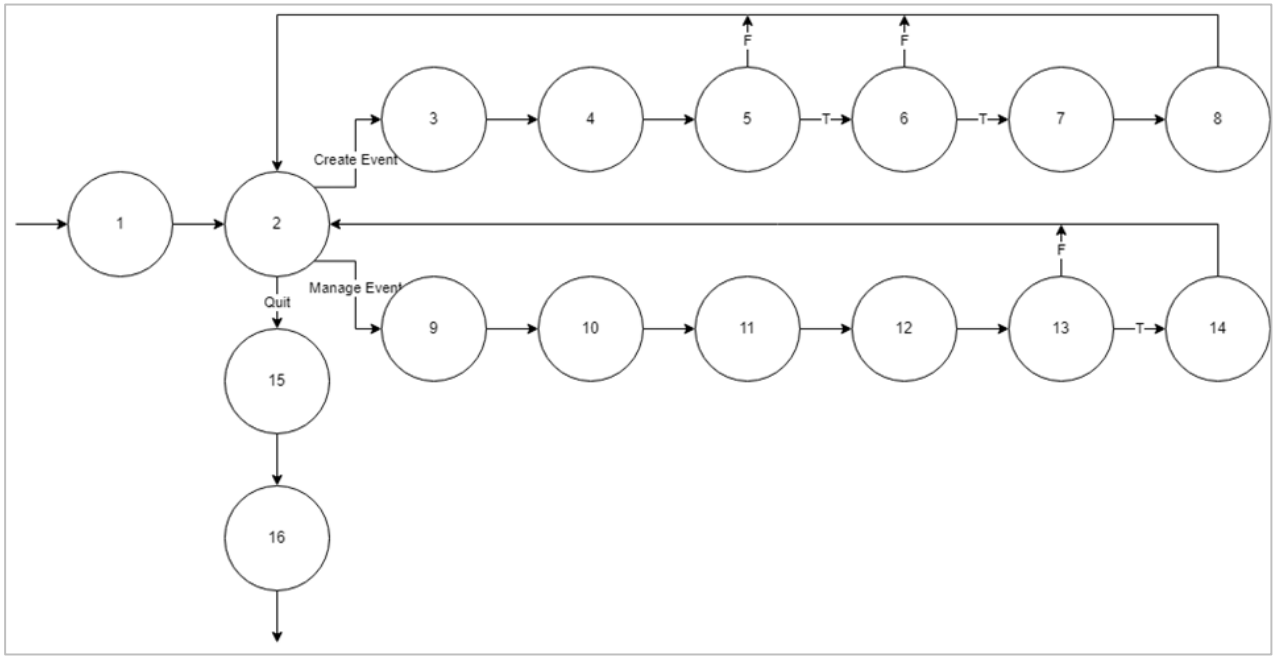

Figure 3: *Control flow graph for the creation and management of events*

General code:
1 - Select event
2 - Create or manage event
15 - Confirmation for end of connection
16 - Ends connection
Scope: Create new event
3 - Initialize inputs for new event
4 - Store input information for new event
5 - Check validity of inputs
6 - Check the existence of the new event in the system
7 - Create new event slot in database based on the inputs
8 - Update system to ensure changes are made
Scope: Manage existing event
9 - Display existing events manage by the event leader
10 - Select an existing event
11 - Display information of selected event
12 - Initialize inputs for selected event details
13 - Check validity of inputs
14 - Update system to ensure changes are made

Cyclomatic complexity = 6

Independent Paths:
1. 1-[2-3-4-5-6-7-8]-15-16
2. 1-[2-3-4-5]-15-16
3. 1-[2-3-4-5-6]-15-16
4. 1-[2-9-10-11-12-13-14]-15-16
5. 1-[2-9-10-11-12-13]-15-16
6. 1-2-15-16

Valid Test Case

- User chose to create a new event, all inputs are valid, name of event does not exist.

Independent path taken: 1
Expected output: New event is created and the user is sent back to the event management interface.

- User chose to create a new event, all or some inputs are valid, name of event does not exist.

Independent path taken: 2
Expected output: Error occurred and the user is sent back to the event management interface.

- User chose to create a new event, all inputs are valid, name of event exist.

Independent path taken: 3
Expected output: Error occurred and the user is sent back to the event management interface.

- User chose to manage event, all inputs are valid.

Independent path taken: 4
Expected output: Event schedules and details are updated and the user is sent back to the event management interface.

- User chose to manage event, all or some inputs are valid.

Independent path taken: 5
Expected output: Error occurred and the user is sent back to the event management interface.

- User chose to quit the system.

Independent path taken: 6
Expected output: Event management system interface is ejected.

3. Assign Lecturers and Event Leaders to courses and events.

Black Box Testing:

Input Condition	Valid Equivalence Class	Valid Boundary	Invalid Equivalence Class	Invalid Boundary
Schedule				
Type	Course Event	Code starts with 'C' [1] Code starts with 'EVE' [2]	Others [7]	
ID	ID of existing schedule [3]		ID of non-existing schedule [8]	
Role				
Name	Lecturer [4] Event Leader [5]		Others [9]	
Type	String [6]		Others [10]	

Valid Test Cases

Schedule	Role	Boundaries Covered
CSE3033	Lecturer	1, 3, 4, 6
EVE1234	Event Leader	2, 3, 5, 6

Invalid Test Cases

Schedule	Role	Boundaries Covered
12345678	Student	7
12345678	Student	8
CSE3033	Student	9
CSE3033	1	10

White Box Testing:

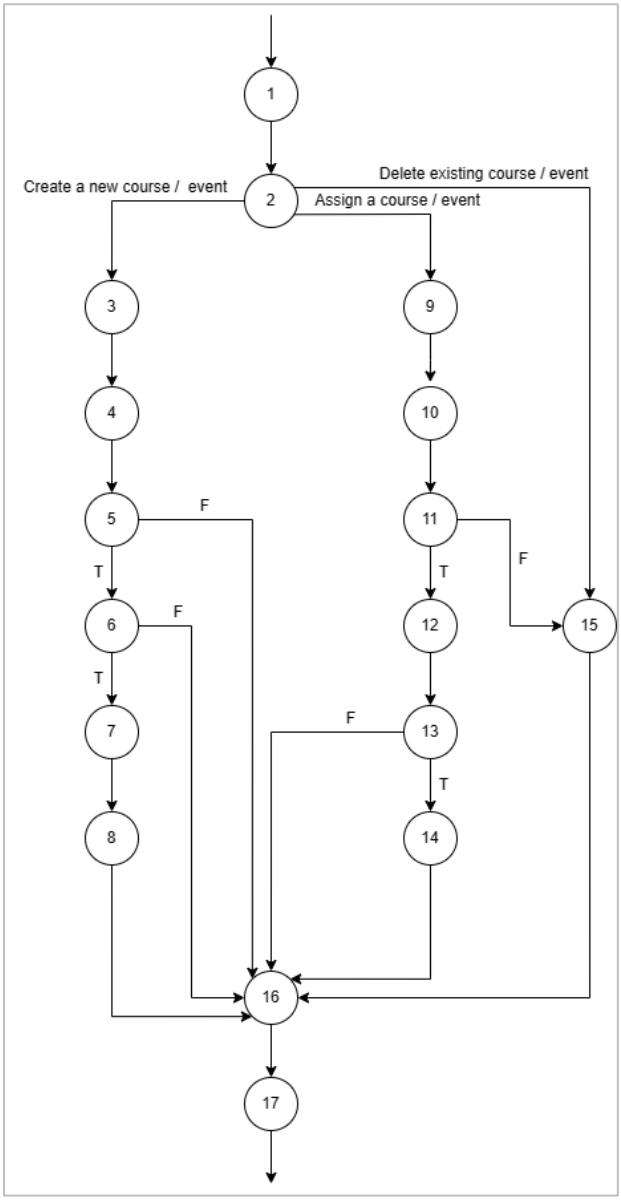

Figure 4: *Control flow graph for assign Lecturers and Event Leaders to courses and events*

General code:
1 - Select course / event
2 - Create or manage course / event
16 - Confirmation for end of connection
17 - Ends connection

Scope: Create new course / event
3 - Initialize inputs for new course / event
4 - Store input information for new course / event
5 - Check validity of inputs

6 - Check the existence of the new course / event in the system
7 - Create new course / event slot in database based on the inputs
8 - Update system to ensure changes are made

Scope: Manage existing course / event
9 - Display existing course / event managed by the user
10 - Select an existing course / event
11 - Display information of selected course / event
12 - Initialize inputs for selected course / event details
13 - Check validity of inputs
14 - Update system to ensure changes are made
15 – Delete an existing course / event

Cyclomatic complexity = 7

Independent Paths:

1. 1-[2-3-4-5-6-7-8]-16-17
2. 1-[2-3-4-5]-16-17
3. 1-[2-3-4-5-6]-16-17
4. 1-[2-9-10-11-12-13-14]-16-17
5. 1-[2-9-10-11-12-13]-16-17
6. 1-[2-9-10-11-15-16]-17
7. 1-2-16-17

Valid Test Case

- User is a lecturer, selects a course to manage, and has permission to edit. Independent path taken: 4
  Expected output: Lecturer is able to assign themselves to the course and the database is updated accordingly. User is sent back to the course management interface.

- User is an event leader, selects an event to manage, and has permission to edit. Independent path taken: 4
  Expected output: Event leader is able to assign themselves to the event and the database is updated accordingly. User is sent back to the event management interface.

- User attempts to assign themselves to a course / event they are already assigned to. Independent path taken: 5
  Expected output: Error message is displayed indicating that the user is already assigned to the selected course / event. User is sent back to the course / event management interface.

- User attempts to assign themselves to a course / event that does not exist.
  Independent path taken: 2
  Expected output: Error message is displayed indicating that the course/event does not exist. User is sent back to the course / event management interface.

- User attempts to assign themselves to a course / event without the necessary permissions.
  Independent path taken: 5

Expected output: Error message is displayed indicating that the user does not have the necessary permissions to make the requested changes. User is sent back to the course / event management interface.

- User selects an existing course / event to delete and confirms the deletion process.
  Independent path taken: 6
  Expected output: The selected course / event is deleted from the system and the connection ends with a confirmation message.

References

Edecofy (2021) *Handle Your School Curriculum Effectively, By Automating your School Timetable Management*. Available at: https://www.edecofy.com/blog/handle-your-school-curriculum-effectively-by-automating-your-school-timetable-management/ [accessed 3 April 2023].

Monsoonfish Inc. (2023) *UX design impact on mobile App Growth*. LinkedIn. Available at: https://www.linkedin.com/pulse/ux-design-impact-mobile-app-growth-monsoonfish?trk=pulse-article [accessed 14 April 2023].

Sweedu.com (2022) *Timetable Management System: What are you missing if your school doesn't use one?* Available at: https://sweedu.com/blog/school-timetable-management-system/ [accessed 3 April 2023].

thalesgroup (2023) *What is a software maintenance process? 4 types of software maintenance, 4 Types of Software Maintenance*. Available at: https://cpl.thalesgroup.com/software-monetization/four-types-of-software-maintenance [accessed 14 April 2023].

Marking Rubric

Section	Failed (0-49)	Third Class (50-59)	Second Class Lower (60-69)	Second Class Upper (70-79)	First Class (80-100)	Mark
Section 1: Introduction to the Apps, Product Backlog and User Stories (40%)	The software is totally out of scope or does not contain much meaningful features, missing product backlog and user stories.	The scopes of software are explained but largely irrelevant. Many areas of product backlog and user stories are incomplete or incorrect.	The scopes of software are explained and minor area is irrelevant. 3-4 areas of product backlog and user stories are incomplete or incorrect.	The scopes of software are explained and relevant. 1-2 areas of product backlog and user stories are incomplete or incorrect.	The scopes of software are explained and relevant. All areas of product backlog and user stories are correct and precise.	Raw mark /100 Section mark /40
Section 2: Class Diagram Design (20%)	The class diagram design is not provided or totally in wrong UML diagram format.	The class diagram is given but it is not designed according to Section 1 entirely. Many notations and relationships are missing.	The class diagram is given and designed according to Section 1. 3-4 notations and relationships are missing.	The class diagram is given and designed according to Section 1. 1-2 notations and relationships are missing.	The class diagram is given and designed according to Section All notations and relationships are correct with relevant assumptions made.	Raw mark /100 Section mark /20
Section 3: Software Evolution (20%)	No feedback process was attempted. No perfective change is proposed or totally irrelevant.	Minimum effort in getting feedback and largely irrelevant. The compilation of feedback to perfective changes proposal are largely incomplete and wrong.	Feedback inquiry is attempted. Feedbacks given are relevant but lacking of correct explanations to construct perfective changes plan.	Feedback inquiry is attempted. Feedbacks given are relevant and only minor errors in perfective changes plan construction.	Feedback inquiry is attempted. Feedbacks given are relevant and construction of perfective changes plan are detailed and concise.	Raw mark /100 Section mark /20

Section 4: Software Testing (20%)	No test plan is provided or all test areas are incorrect.	Test plan is provided but only white box or black box testing is given.	Test plan with both white box and black box testing are provided. 3-4 test cases are not appropriate or incomplete.	Test plan with both white box and black box testing are provided. 1-2 test cases are not appropriate or incomplete.	Test plan with both white box and black box testing are provided. All test cases are appropriate and described in detailed and concise.	Raw mark /100 Section mark /20
					Total Score:	/100

Written comments area: