### lmu_blakCourse Submission Cover Sheet Module:

### CS5002 Software Engineering

### Assignment no: 002

### Weighting: 35% of the module mark

### Deadline: 3 PM Monday 22nd of April 2024

**Module Leader: Ramzi Djemai Student ID:**

PLAGIARISM

You are reminded that there exist regulations concerning plagiarism. Extracts from these regulations are printed below. Please sign below to say that you have read and understand these extracts:

(signature:) Date:

This header sheet should be attached to the work you submit. **No work will be accepted without it.**

Extracts from University *Regulations on*

Cheating, Plagiarism and Collusion

Section 2.3: "The following broad types of offence can be identified and are provided as indicative examples …..

1. **Cheating: including taking unauthorised material into an examination; consulting unauthorised material outside the examination hall during the examination; obtaining an unseen examination paper in advance of the examination; copying from another examinee; using an unauthorised calculator during the examination or storing unauthorised material in the memory of a programmable calculator which is taken into the examination; copying coursework.**
2. **Falsifying data in experimental results.**
3. Personation, where a substitute takes an examination or test on behalf of the candidate. Both candidate and substitute may be guilty of an offence under these Regulations.
4. **Bribery or attempted bribery of a person thought to have some influence on the candidate's assessment.**
5. Collusion to present joint work as the work solely of one individual.
6. Plagiarism, where the work or ideas of another are presented as the candidate's own.
7. Other conduct calculated to secure an advantage on assessment.
8. Assisting in any of the above.

Some notes on what this means for students:

1. Copying another student's work is an offence, whether from a copy on paper or from a computer file, and in whatever form the intellectual property being copied takes, including text, mathematical notation and computer programs.

**CS5002 – Hasan Khan – 21002595**

**Coursework Report**

Contents

[Course Submission Cover Sheet Module: 1](#_Toc164971469)

[CS5002 Software Engineering 1](#_Toc164971470)

[Assignment no: 002 1](#_Toc164971471)

[Weighting: 35% of the module mark 1](#_Toc164971472)

[Deadline: 3 PM Monday 22nd of April 2024 1](#_Toc164971473)

[Section 1: Planning, requirements modelling and analysis 2](#_Toc164971474)

[Gantt Chart: 3](#_Toc164971475)

[Use Case Model: 5](#_Toc164971476)

[High-Level Use Case Descriptions: 6](#_Toc164971477)

[Becoming a new member Collaboration and Sequence Diagram: 8](#_Toc164971478)

[How to Design a Class Diagram: 9](#_Toc164971479)

[Analysis Class Diagram: 12](#_Toc164971480)

[Section 2: Design 13](#_Toc164971481)

[Moving Forward: 13](#_Toc164971482)

[GUI Interface Design and explanation: 14](#_Toc164971483)

[Appendix: 16](#_Toc164971484)

[References: 20](#_Toc164971485)

# Section 1: Planning, requirements modelling and analysis

## Gantt Chart:

Important: I did not have access to Visual Paradigm. Here is the Gant Chart in its table form.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | Start Date | End Date | Duration (hrs) | Phase |
| Initiation Phase | 17/04/2024 | 17/04/2024 | 1 | Initiation Phase |
| Define project scope and objectives | 17/04/2024 | 17/04/2024 | 11 | Initiation Phase |
| Identify Stakeholders | 17/04/2024 | 17/04/2024 | 1 | Initiation Phase |
| Conduct Initial Risk Assessment | 17/04/2024 | 17/04/2024 | 1 | Initiation Phase |
| Obtain project approval | 17/04/2024 | 17/04/2024 | 1 | Initiation Phase |
| Planning Phase | 17/04/2024 | 17/04/2024 | 1 | Planning Phase |
| Gather System requirements | 17/04/2024 | 17/04/2024 | 1 | Planning Phase |
| Define project plan | 17/04/2024 | 17/04/2024 | 11 | Planning Phase |
| Create risk management plan | 17/04/2024 | 17/04/2024 | 1 | Planning Phase |
| Analysis and Design Phase | 17/04/2024 | 17/04/2024 | 2 | Analysis and Design Phase |
| Analyse current systems | 18/04/2024 | 18/04/2024 | 1 | Analysis and Design Phase |
| Gather user feedback | 18/04/2024 | 18/04/2024 | 1 | Analysis and Design Phase |
| Design System Architecture | 18/04/2024 | 18/04/2024 | 1 | Analysis and Design Phase |
| Develop database schema | 18/04/2024 | 18/04/2024 | 2 | Analysis and Design Phase |
| Implementation phase | 18/04/2024 | 18/04/2024 | 1 | Implementation phase |
| Develop software modules | 18/04/2024 | 18/04/2024 | 6 | Implementation phase |
| Integrate components | 18/04/2024 | 18/04/2024 | 2 | Implementation phase |
| Conduct coding | 18/04/2024 | 18/04/2024 | 2 | Implementation phase |
| Testing phase | 18/04/2024 | 18/04/2024 | 2 | Testing phase |
| Write test cases | 18/04/2024 | 18/04/2024 | 2 | Testing phase |
| Conduct unit testing | 18/04/2024 | 18/04/2024 | 2 | Testing phase |
| System testing | 19/04/2024 | 19/04/2024 | 2 | Testing phase |
| User acceptance testing | 19/04/2024 | 19/04/2024 | 2 | Testing phase |
| Deployment phase | 19/04/2024 | 19/04/2024 | 2 | Deployment phase |
| Deploy the system | 20/04/2024 | 20/04/2024 | 2 | Deployment phase |
| Train users | 20/04/2024 | 20/04/2024 | 2 | Deployment phase |
| Provide ongoing support | 20/04/2024 | 20/04/2024 | 2 | Deployment phase |

## Use Case Model:

A diagram of a company

Description automatically generatedA diagram of a diagram

Description automatically generated

## High-Level Use Case Descriptions:

1. Becoming a member:

Description: This use case allows user to sign up for a membership through the Front Desk Staff. The Front Desk Staff must interact with them before they sign up.

Expanded Use Case Description:

Title: Becoming a Member

Primary Actor(s): Front Desk Staff, User

Description: This use case allows front desk staff to add a new member to the leisure centre's membership system. The process includes collecting necessary information from the user, verifying the information, assigning a unique membership ID, issuing a membership card, and updating the membership records.

Preconditions:

* Front desk staff are logged into the system.
* The new member provides accurate personal information and selects a membership plan.

Postconditions:

* The new member is successfully added to the system.
* A unique membership ID is assigned, and a membership card is issued.
* Membership records are updated with the new member's information.

Main Success Scenario:

* Front desk staff initiate the process to add a new member.
* The system prompts front desk staff to enter the new member's details, including name, contact information, and membership preferences.
* Front desk staff verify the provided information and process the membership payment.
* The system assigns a unique membership ID to the new member and issues a membership card.
* Front desk staff update the membership records with the new member's information.
* The system confirms the successful addition of the new member.

Alternates:

If the provided information is incomplete or inaccurate, front desk staff may request the new member to provide the necessary details or correct the information before proceeding with the membership registration process.

1. Manage bookings, utilising facilities/services:

Description: This use case enables members to manage bookings for various facilities and activities offered by the leisure centre, such as sports facilities, classes, events, and venue rentals. It includes functionalities for checking availability, making, modifying, and cancelling bookings.

Expanded Use Case Description:

Title: Manage Facility and Service Booking

Primary Actor: Member, Front Desk Staff

Description: This use case allows members to make bookings for various facilities and activities offered by the leisure centre through the front desk staff. The process includes checking the availability of facilities, selecting a suitable time slot, confirming the booking, and receiving a booking confirmation.

Preconditions:

* The member is logged into the system.
* The desired facility is available for booking during the selected time slot.

Postconditions:

* The booking is successfully confirmed.
* The member receives a booking confirmation via email or message.
* The booked facility is reserved for the member during the specified time slot.

Main Success Scenario:

* The member initiates the process to make a facility booking.
* The system presents available facilities and time slots to the member.
* The member selects the desired facility and time slot for booking.
* The system confirms the availability of the selected facility and time slot.
* The member confirms the booking.
* The system generates a booking confirmation and notifies the member.

Alternates:

If the selected facility or time slot is not available, the system prompts the member to choose an alternative option.

1. Producing Reports:

Description: This use case allows Administrators, Sales Supervisors, Marketing Coordinators, Human Resources, and the Finance Officers to generate various reports for management and administrative purposes such as membership statistics, booking summaries, financial reports, staff performance reports, marketing campaign effectiveness, and inventory management reports.

1. Marketing Campaigns:

Description: This use case involves managing marketing activities, including creating and updating promotional campaigns, managing customer communication channels (e.g., email, social media), analysing marketing data, and generating reports on performance. Primarily this focus is on the Marketing Coordinators and Sales Supervisors as they can brand themselves, promote offers and make the business profit.

1. Human Resources:

Description: This use case includes functionalities for managing human resources, such as employee recruitment, onboarding, training, performance evaluation, and payroll processing. It also involves maintaining employee records and generating HR-related reports.

1. Sales Supervisors:

Description: This use case involves the Sales Supervisors to manage sales activities, including inventory management, processing sales transactions, handling customer inquiries, managing discounts and promotions, and generating sales reports. They also interact with members for upgrading membership, handling inquiries, discounts, and promotions etc.

1. Finance Officers:

Description: This use case includes functionalities for managing financial operations, such as budgeting, invoicing, expense tracking, revenue management, and financial reporting. It also involves managing supplier relationships and processing payments.

## Becoming a new member Collaboration and Sequence Diagram:

How to Design a Collaboration Diagram (for the entire Use Case Diagram):

* Identify Actors and System Components: Start by identifying the actors (such as User, Member, Front Desk Staff, Administrator etc) and the system components (Administrative System, Communication System, Management Information System)
* Map Interactions: For each use case I would map out the interactions between the actors and system components. Determining which systems are responsible for fulfilling specific actions or behaviours requested by the actors.
* Visualize Interactions: I will Use arrows to indicate the flow of communication between actors and components and label the interactions with actions.

Collaboration Diagram (Use Case: Becoming a member):

A diagram of a company

Description automatically generated

## How to Design a Class Diagram:

Actors:

* Front Desk Staff
* Member
* Sales Supervisors
* Marketing Coordinators
* Human Resources
* Finance Officers
* Managers

Classes:

Membership System:

Attributes: MembershipID=Int, MembersInformation=String, MembershipPlan=String,

Methods: AddMember(), VerifyInformation(), IssueMembershipCard(), UpdateMembershipRecords(), MembershipRenewal()

Booking System:

Attributes: BookingID=Int, FacilityServices=String, TimeSlot=Float, Available=Boolean,

Methods: CheckAvailability(), MakeBooking(), ModifyBooking(), CancelBooking(), ConfirmBooking()

Report Generation System:

Attributes: ReportNumberID=Int, ReportType=String, ReportInfo=String

Methods: GenerateReport()

Marketing Campaign Management:

Attributes: CampaignID=Int, CampaignDetails=String, MarketingData=String

Methods: CreateCampaign(), UpdateCampaign(), AnalyzeData(), GeneratePerformanceReport()

Human Resources Management:

Attributes: EmployeeID=Int, EmployeeInfo=String, RecruitmentData=String, TrainingData=String, PayrollInfo=String

Methods: RecruitEmployee(), OnboardEmployee(), EvaluatePerformance(), ProcessPayroll(), GenerateHumanResourcesReport()

Sales Management:

Attributes: SalesID=Int, SalesInfo=String, SalesTransactions=Float, InventoryInformation=String, CustomerInquiries=String, SalesData=String

Methods: ProcessTransaction(), ManageInventory(), HandleInquiries(), ManageDiscountsPromotions(), GenerateSalesReport()

Financial Management:

Attributes: BudgetData=String, InvoiceInformation=String, ExpenseData=String, RevenueData=String, SupplierInformation=String, PaymentInformation=String

Methods: ManageBudget(), TrackExpenses(), ManageRevenue(), GenerateFinancialReport(), ProcessPayments()

Management System:

Attributes: ManagerID=int, ManagerData=String, ManageRoles(), CommunicationDetails()

Methods: Communicate(), ViewReports(), GenerateManagementReports(), ManageOperations(), ManageMarketingCampaigns(), ManageFinances(), ManageHumanResources()

Relationships:

Front Desk Staff and Member are associated with the Membership System. The Front Desk has one or more actions with the Membership System but the Members themselves have no access to this system.

Members are associated with the Booking System. They have one or more actions associated such as viewing time slots and booking themselves in for facilities/services.

Various Classes have an aggregation relationship with the Report Generation System for generating different reports. It is not a composition relationship as reports are not always being generated so there is not a strict dependency between the Report Generation System and the classes (one can exist without the other). The classes can generate one or more reports, but the report can only belong to one class.

Marketing Coordinators and Sales Supervisors interact with the Marketing Campaign Management system. Both actors can have 1 or more interactions with the system.

Human Resources interacts with the Human Resources Management system. They can do 1 more things related to the system.

Sales Supervisors interact with the Sales Management system and Finance Officers interact with the Financial Management system. Both actors have 1 or more actions related to the system.

The Management System interacts with the Managers. The Managers have 1 or more uses for the Management System and the Management System is used by ‘n’ (where ‘n’ is equal to the number of managers). There is a composition relationship between this main Management System and the other management systems as the managers must exist to oversee all reports and know what is happening in the Leisure Centre. It is a composition relationship as there will be no structure between all management systems if the Management System didn’t exist and without structure, the other management systems would fail as there is nothing to oversee and manage (the departments would fail to do its job successively). The Management System can overlook one or many operations within the other management systems, but the other management systems can only be overlooked by ‘m’ (where m is equal to the number of financial/HR/Marketing/Sales Managers).

A black and white sign with white text

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedAnalysis Class Diagram:

# Section 2: Design

## Moving Forward:

Database Design:

Technique/Product: Entity-Relationship Diagrams and Database Schemas

Purpose: I would design an ERD for each system to visualize the relationships between different data entities in the system. This ensures efficient data storage and retrieval.

User Interface Design:

Technique/Product: Wireframes

Purpose: Wireframes visually represent the layout, structure, and flow of the user interface. This will help in testing user interactions, gathering feedback from all the stakeholders as well as ensuring usability and user satisfaction.

Security Design:

Technique/Product: Threat Modelling and Security Architecture

Purpose: Threat modelling identifies any potential security threats (and vulnerabilities) in the system which helps to devise mitigation security controls. Security architecture is about the security mechanisms, access controls, encryption, authentication, and authorization mechanisms which protect the system from unauthorized access and attacks.

Design, Testing and Deployment:

Technique/Product: Necessary tools to design and test.

Purpose: The end phase of the journey - creating the systems and adding or removing content based on the feedback with the interface design. After deployment the system may experience errors which can be modified through updates to the system. Any last-minute modifications can be assessed and made too.

## GUI Interface Design and explanation:

Important: I have used Eclipse IDE and Java (jdk21). I have used Swing and awt imports.

A screenshot of a computer

Description automatically generated

This is the GUI for the Membership system. As you can see Membership ID is auto-generated (I will get back to this later) and there are inputs fields for the necessary requirements when adding a new user. The Membership Plan is a select option (you don’t need to type anything there are only 3 options). There is a need for proof of ID so I have inputted a photo ID import which will be saved onto the membership file for that member. On the bottom of the screen there are 3 button options which allow the Front Desk Staff to add the new member, verify the member (by sending a text or an email with a verification code) as well as an Edit Members which allows the staff to edit/delete any member’s records. The Verify Member and Edit Members will take you to another Interface with the relevant input fields and such.

(Since this is currently the testing phase) For feedback purposes from the Leisure Centre the GUI can be modified to include the Logo in the page and colours to enhance design. I can add some more options as well such as an Employee Login which is a type of verification for the employees so that the membership system only allows employees to access and use the system.

A screenshot of a computer

Description automatically generated

As you can see there are 3 select options for the Membership Plan.

A screenshot of a computer

Description automatically generated

I am currently showcasing and testing what adding a member would look like.

A screenshot of a computer

Description automatically generated

As you can see, after you press ‘Add Member’ you will get a prompt that will issue an ID. This will be linked to the database records (assuming the member has passed verification).

# Appendix:

Code:

import javax.swing.\*;

import javax.swing.border.EmptyBorder;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

public class Main extends JFrame {

// Attributes for the input fields

private JTextField memberIdField;

private JTextField phoneNumberField;

private JTextField streetAddressField;

private JTextField postcodeField;

private JComboBox<String> membershipPlanComboBox;

private JLabel photoIdLabel;

// Main Constructor

public Main() {

// The frame of the GUI

setTitle("Membership System");

setSize(400, 350);

setDefaultCloseOperation(***EXIT\_ON\_CLOSE***); //exit when you close

setLayout(new BorderLayout());

// Create input panel - grid layout

JPanel inputPanel = new JPanel(new GridLayout(6, 2, 5, 5));

inputPanel.setBorder(new EmptyBorder(10, 10, 10, 10));

// Stating the GUI components

JLabel memberIdLabel = new JLabel("Membership ID:");

JLabel phoneNumberLabel = new JLabel("Phone Number:");

JLabel streetAddressLabel = new JLabel("Street Address:");

JLabel postcodeLabel = new JLabel("Postcode:");

JLabel membershipPlanLabel = new JLabel("Membership Plan:");

JLabel photoIdLabel = new JLabel("Photo ID:");

memberIdField = new JTextField();

memberIdField.setEditable(false); // Membership ID is automatically generated so cant add anything here

phoneNumberField = new JTextField();

streetAddressField = new JTextField();

postcodeField = new JTextField();

// Membership plan box to click and decide what membership

String[] membershipPlans = {"YearlyMembership", "6MonthsMembership", "3MonthMembership"};

membershipPlanComboBox = new JComboBox<>(membershipPlans);

JButton addMemberButton = new JButton("Add Member");

addMemberButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

addMember();

}

});

JButton importPhotoButton = new JButton("Import Photo");

importPhotoButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

importPhoto();

}

});

JButton verifyMemberButton = new JButton("Verify Member");

verifyMemberButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

verifyMember();

}

});

JButton editMemberButton = new JButton("Edit Members");

editMemberButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

editMember();

}

});

// components for the input panel - declaring

inputPanel.add(memberIdLabel);

inputPanel.add(memberIdField);

inputPanel.add(phoneNumberLabel);

inputPanel.add(phoneNumberField);

inputPanel.add(streetAddressLabel);

inputPanel.add(streetAddressField);

inputPanel.add(postcodeLabel);

inputPanel.add(postcodeField);

inputPanel.add(membershipPlanLabel);

inputPanel.add(membershipPlanComboBox);

inputPanel.add(photoIdLabel);

inputPanel.add(importPhotoButton);

// Creating the button in the panel

JPanel buttonPanel = new JPanel(new GridLayout(1, 3, 5, 5));

buttonPanel.setBorder(new EmptyBorder(5, 10, 5, 10));

buttonPanel.add(addMemberButton);

buttonPanel.add(verifyMemberButton);

buttonPanel.add(editMemberButton);

// Adding the input panel and button panel to the frame - layering it out nicely

add(inputPanel, BorderLayout.***CENTER***);

add(buttonPanel, BorderLayout.***SOUTH***);

// have to make the frame visible

setVisible(true);

}

// Method to add a new member

private void addMember() {

// Generate membership ID (for demonstration, generating a random number)

int memberId = (int) (Math.*random*() \* 1000000); //up to 1,000,000 different ID numbers

// Retrieving data from text fields and combo box

String phoneNumber = phoneNumberField.getText();

String streetAddress = streetAddressField.getText();

String postcode = postcodeField.getText();

String membershipPlan = (String) membershipPlanComboBox.getSelectedItem();

// Call methods - related to adding a member

boolean success = AddMember(memberId, phoneNumber, streetAddress, postcode, membershipPlan);

// Displaying the result

if (success) {

JOptionPane.*showMessageDialog*(this, "Member added successfully! Membership ID: " + memberId);

} else {

JOptionPane.*showMessageDialog*(this, "Failed to add member. Please check input data.");

}

}

// Method to import photo for PhotoID - actually works to choose a file - but problems with uploading

private void importPhoto() {

JFileChooser fileChooser = new JFileChooser();

int result = fileChooser.showOpenDialog(this);

if (result == JFileChooser.***APPROVE\_OPTION***) {

File selectedFile = fileChooser.getSelectedFile();

// Display the selected file name or process the selected image file

String fileName = selectedFile.getName();

JOptionPane.*showMessageDialog*(this, "Photo imported: " + fileName);

}

}

// The method to verify a new member

private void verifyMember() {

// For demonstration purposes, this is the message - prints this out in console

System.***out***.println("Verifying new member...");

}

// The method to edit an existing member

private void editMember() {

// DEMONSTRATION PURPOSES – look above

System.***out***.println("Editing member...");

}

// Method to add a member to the system

private boolean AddMember(int memberId, String phoneNumber, String streetAddress, String postcode, String membershipPlan) {

// Logic for adding a member

System.***out***.println("New Member Added:");

System.***out***.println("Membership ID: " + memberId);

System.***out***.println("Phone Number: " + phoneNumber);

System.***out***.println("Street Address: " + streetAddress);

System.***out***.println("Postcode: " + postcode);

System.***out***.println("Membership Plan: " + membershipPlan);

// Returns true if member is successfully added

return true;

}

// THE MAIN METHOD

public static void main(String[] args) {

SwingUtilities.*invokeLater*(new Runnable() {

*@Override*

public void run() {

new Main();

}

});

}

}

# References:

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Draw.io ¬ Available at = https://app.diagrams.net/ ¬ Accessed: 20/04/2024

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Use\_Case .ppt ¬ Available at = <https://bblearn.londonmet.ac.uk/ultra/courses/_51474_1/cl/outline> ¬ Accessed: 20/04/2024

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Use\_Cases\_continued.ppt ¬ Available at = <https://bblearn.londonmet.ac.uk/ultra/courses/_51474_1/cl/outline> ¬ Accessed: 20/04/2024

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Lecture 06 – Collaboration Diagrams.ppt ¬ Available at = <https://bblearn.londonmet.ac.uk/ultra/courses/_51474_1/cl/outline> ¬ Accessed: 20/04/2024

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Lecture 09 - Sequence Diagrams.pptx ¬ Available at = <https://bblearn.londonmet.ac.uk/ultra/courses/_51474_1/cl/outline> ¬ Accessed: 20/04/2024

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Lecture 07 – Class\_Diagram (1:2).pptx ¬ Available at = <https://bblearn.londonmet.ac.uk/ultra/courses/_51474_1/cl/outline> ¬ Accessed: 21/04/2024

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Lecture 8 – Class Diagram (2:2).ppt ¬ Available at = <https://bblearn.londonmet.ac.uk/ultra/courses/_51474_1/cl/outline> ¬ Accessed: 21/04/2024

Author(s) = N/A ¬ Publication Year = N/A ¬ Publisher = N/A ¬ Title = Lecture 8 – Use\_Cases\_continued.ppt ¬ Available at = <https://bblearn.londonmet.ac.uk/ultra/courses/_51474_1/cl/outline> ¬ Accessed: 21/04/2024

Author(s) = Bijoyan Das ¬ Publication Year = 08/12/2016 ¬ Publisher = N/A ¬ Title = Java Swing Tutorial 1 : Introduction [SERIES] ¬ Available at = https://www.youtube.com/watch?v=H63PEmEqhkM&list=PLMQ4k-hUWGNmQwP3u5HP894NnQQ9lGY\_d&index=1 ¬ Accessed: 21/04/2024