COMP319: Software Engineering II

Assignment 1 (2025/2026)

(100% mark for Assignment 1 is 20% of COMP319 grade)

Deadline for Assignment 1: 3rd of December 2025, 17:00

OBJECTIVE

This assignment Object Oriented Pattern design and involves you producing a class design and implementation for a messaging system.

Assignment number	1 of 1
Weighting	20%
Assignment Circulated date provided to class	26/9/2025
Deadline Day & Date & Time	3rd of December 2025 at 17:00 (5 PM)
Submission Mode	**Electronic submission on Canvas (zip)**
Learning outcome assessed	1. **Understand the key problems driving research and development in contemporary software engineering (eg the need to develop software for embedded systems).** 2. **Be conversant with approaches to these problems, as well as their advantages, disadvantages, and future research directions.**
Submission necessary in order to satisfy Module requirements	No

Purpose of assessment **Marking criteria**	To assess the students' ability to understand the use of object orientated patterns. See end of document
Late Submission Penalty	Standard UoL Policy

Instructions

This design task requires you to produce a class design and implementation for a messaging system. The messaging system should be able to send messages to users who have contact details, for example telephone numbers, email addresses and postal addresses. The system needs to use a chain of responsibility depending on the urgency of the message, or the type of message to send.

Note the code does not need to connect to real email providers or SMS services, but there should be stub code in place that indicates what will happen, for example something that prints on the console using System.println a message, for example "sending hello Seb to +44 12345678 via CSoft SMS".

The code should be able to support multiple SMS providers and multiple email providers, but as said before it doesn't need to actually send the SMS.

(If you wish you can add this code in as project of your own).

The interface to your messaging package should use the façade structure, with a public class. The message senders need to have the ability to indicate a cost, for example 10p per SMS message. The message senders should be able to fail to send a message (you can simulate this by using the java Random class and fail some of the time) and then it will be sent to the next message sender in the chain.

When sending a message you need to have a response from the system, the response needs to have a text message as well as a enum response code, for example OK.

You are expected to demonstrate the use of the following OO techniques and patterns.

Factory

Facade

Chain of responsibility

Open closed principle

Single responsibility

The code can be written in Java™ or C#.

Responsibilities

You should make a list of the relevant responsibilities for each class and determine what data should be modelled.

Marking

Relevant use of factory class for the problem given	20%
Relevant use of the chain of responsibility for the problem given	20%
Good open/closed application	20%
Single responsibility	20%
Proper façade structure	10%
Overall code readability and quality	10%

The code should be in a package called messaging.

There should be package called main, with a class Main, with an entry point to the code, static void main(String args[]). This method should test all the features of your code. There should be a class called User which contains all the user's contact details. The code should properly validate all the contact information such as email addresses, telephone numbers and postcodes.

Note you must use the patterns given to solve the problem given, so not just have for example a piece of factory code which is to solve a different problem. Please implement the pattern explicitly in your code and not rely of the Java JDK libraries.

Submission format

All the source code should be zipped up in it correct path structure and uploaded to Canvas.

Manifest file

There should be a manifest file, included in the zip, called manifest.txt which describes each of the Java source files. The format is as follows.

Name	Purpose	Java pattern of principle included
IMessage	The interface for a message	Single responsibility, open closed

Make sure you have highlighted in which source files the aspects of the marking scheme have been covered. Files without a manifest, will lose marks.
The code must compile.