SOFT252 Object-oriented Software Engineering with Design Patterns

20 CREDIT MODULE / 70% COURSEWORK, 30% IN-CLASS TEST

MODULE LEADER: JAMES HAYTER

MODULE AIMS

- To introduce industry standard production tools for content creation
- To investigate professional practice in relation to the sector
- To design a software solution that meets user requirements

ASSESSED LEARNING OUTCOME:

1. Use requirements analysis artefacts to progress into software design
2. Identify and use suitable design patterns
3. Develop and evaluate an object-oriented program to solve a given problem

OVERVIEW

You are required to produce a Java standalone software application that meets the requirement specification in Appendix 1 of this assignment. This application should follow the MVC pattern (as defined in lecture). In addition, your data model design must use at least two other software design patterns covered in the course.

Your implementation is in the Java programming language using the NetBeans IDE, consisting of:

- A UML diagram of your system with all classes and relationships present
- A Java NetBeans project, which provides a graphical user interface (GUI).
    - The GUI should provide the functionality listed in Appendix 1: Scenario.

You MUST use Git version control with Bitbucket or GitHub in your development process. You MUST add the module leader (user DarthWasabi in GitHub or Bitbucket) to your repository. You will be assessed on appropriate use of version control.

Assessment

Individual Coursework comprising one assignment with a GIT repository of incremental development leading to a final build or release. Repository includes itemized deliverables for final module submission. Typically, this will include your NetBeans project and UML documentation, with a current build or release executable. All code to be version controlled and commented all 3rd party assets and resources to be formally credited in the README file on the repository.

MODULE DELIVERY

Delivery format: Lecture/Practical format with inputs from the SOFT252 team. Module begins 26th September, comprising weekly 2-hour lecture and a 2-hour practical to support independent study. Module Delivery is led by James Hayter with additional inputs and tutorial support from Swen Gaudl, Mark Dixon and Christopher Haynes.

Delivery location: Babbage room 210

Delivery staff: James Hayter – james.hayter@plymouth.ac.uk
Swen Gaudl – swen.gaudl@plymouth.ac.uk
Mark Dixon - mark.dixon@plymouth.ac.uk
Christopher Haynes - christopher.haynes@plymouth.ac.uk

Duration: 13 weeks

Important dates: In-class test (Group 3) – 25th November 2019
In-class test (Group 1) – 26th November 2019
In-class test (Group 2) – 27th November 2019
Assignment hand-in (all groups) – 9th January 2020

Element	Description	Deadline	Percentage
	In-class test – Group 3	25/11/2019 (Week 18)	30%
	In-class test – Group 1	26/11/2019 (Week 18)	30%
	In-class test – Group 2	27/11/2019 (Week 18)	30%
C1	Assignment submission (all groups)	09/01/2020 (Week 25)	70%

NB: For a 20 credit module there will be on average 50hrs of taught content and staff led activities with 150hrs of independent study - students are expected to work through exercises, read and assimilate provided resource materials and prepare for the next session, final assessment will include elements from further study materials, extending taught materials. It is essential you apply your study skills and provide evidence of this additional work.

ASSIGNMENT DELIVERABLES:

You are required to submit a .zip folder including:

- A working executable of your project
- A text file with a link to the GitHub repository for your project
    - Make sure the module leader is added as a collaborator to the repository so it can be accessed by them

The repository requires the following:

- The NetBeans project of your assignment
    - All classes are commented appropriately
    - Java annotations are included for class relationships in code (e.g. overriding rom interfaces)
- Project includes appropriate Junit tests for all classes
    - Include tests for object relationships and methods
    - Use appropriate Java test annotations for each test
    - Focus tests on areas where data is changed or stored
- The UML diagram for the project

DELIVERABLES IN DETAIL

GIT Repository

An online repository of your project using the GIT service. You are graded on the following criteria:

- Repository readme file includes all additional resources (art, sound FX etc.) fully credited
- No previous versions of the project are present in the repository in a .zip or other compressed format
- Commits to the repository are appropriately commented
- Commits are in a consistent timely manner, at least once every week

Build of project

- Project runs with no crashes
    - If you are not able to fix a crash, provide a text file in the build folder explaining how to avoid it
- Project has appropriate GUI for each part of the system
- Data is saved to a text file included in the project
    - You may use JSON, XML or your own custom layout for data storage

UML Diagram

- Must either be in a PDF file or as a part of the NetBeans project using the EasyUML plugin
- Conforms to standard UML 2.0 formatting
- Contains all classes, interfaces and data structures used in the project
- Contains all relationships used in the project
- All text is readable and in a reasonable sized font
- If the diagram PDF is too large for one page, please split the documentation up in a logical manner and provide a contents list at the beginning of the document

COURSEWORK SUBMISSION

We require a digital submission uploaded to the module DLE site, allow sufficient time to upload project work and documentation. If your submission exceeds file size supported by the DLE please set up a folder on your Onedrive account with module code : 'SOFT252' set permissions to allow staff access – do not modify or remove this folder until results are confirmed. If your DLE submission documentation includes additional URL links, for example to project management, repository, YouTube, please ensure you have set the permissions to public and test all from an anonymous browser window.

.

NB: if we cannot access your material from the date of submission we may not be able to mark it!

Please refer to all the lecture content & further study resources on the DLE.

APPENDIX 1: SCENARIO

A clinic has approached you to develop a Patient Management System. They have provided the following scope of the project. The system should have at least four types of users: administrator, doctor, secretary, and patient. The user-specific functionalities are given below.

Functionality	User
Log in to the system using unique ID and password, and perform user-specific functionalities.	All users
Request to create account – this requires approval from a secretary.	Patient
Rate doctors and provide feedback messages.	Patient
View doctors' ratings.	Patient
Request appointment – there should be an avenue to ask for a specific doctor and a range of potential dates.	Patient
View his or her own history.	Patient
View appointment.	Patient
View prescription.	Patient
Request account termination.	Patient
Create own account.	Administrator
Add or remove doctor and secretary accounts.	Administrator
View the ratings of the doctors.	Administrator
Provide feedback to each doctor based on ratings and comments from patients.	Administrator
Approve patient accounts.	Secretary
Receive requests for appointments.	Secretary
Create an appointment between a patient and a free doctor.	Secretary
Give medicines to patients if available.	Secretary
Order and stock medicines if necessary.	Secretary
Remove patients.	Secretary
Approve account removal request from patients.	Secretary
View appointments.	Doctor
Make notes during a consultation.	Doctor
Inspect patient history.	Doctor

Propose and create future appointments for a specific patient.	Doctor
Prescribe medicines and dosages	Doctor
Create new medicines and request secretaries to order these.	Doctor

All users of the system must be identifiable with a given name, surname, an address, and a unique identification number (which will be assigned when an account is created: see **Appendix 2: User identification system** for valid identification number scheme). Each patient must also have an age and gender (assigned at birth) associated with them[2].

All requests or notifications for a specific user must appear as a message to the user when they log in. For instance, when an appointment is created, all involved users should receive a notification message. This message may be viewed once a relevant user logs in.

[2] In this document, *they* is sometimes used as a gender-neutral pronoun.

In addition, the system should record all appointments, in particular patient specific notes, and prescriptions. A doctor may review an individual patient record: neither an administrator nor a secretary should be able to access this record.

During an appointment, the doctor will log in to the system and make notes. They may also prescribe medicines (with required quantity) and recommend dosages: an example prescription is given in the Appendix. If medicines were prescribed, a patient may see a secretary and collect appropriate medicines. The secretary will then update the system to update the stock of medicines. At this point, the secretary may replenish the stock. If a prescribed medicine is not available, then the patient must be notified to come back when the stock has been replenished by a secretary.

For assessment purposes, please populate the system with an administrator, three doctors, three patients, and a secretary. Also, populate the clinic stock of medicine with 10 different products.

APPENDIX 2: USER IDENTIFICATION SYSTEM

The identification number for a user has two parts: the first letter is designated according to his or her role, and then there are four digits for numbers. The role specific letters are given below:

Role	**Letter**
Administrator	A
Doctor	D
Patient	P
Secretary	S

APPENDIX 3: SAMPLE PRESCRIPTION

Patient. Name: John Doe Address: Portland Square Building, Drake Circus, Plymouth, PL4 8AA. Sex: Male Age: 45	Doctor. Name: Serina James Address: Fictitious Clinic, Diagon Alley, Drake Circus, Plymouth, PL4 8AA

Notes.
Mr. John Doe has been suffering from fever for the last two weeks. He is coughing and the phlegm is green in colour. Suspected bacterial infection – we have collected his phlegm for testing. We are immediately starting him on an antibiotic course.

Medicine	**Quantity**	**Dosage**
Amoxicillin	24	4 per day – at least 6 hours between each dose.