# CM10025 Programming 2: Coursework 2 Specification 2023/24

Professor Alan Hayes

| | |
|---|---|
| Date Set | Monday 18th March 2024 |
| Submission Date | 20:00 BST Friday 3rd May 2024 (Moodle) |
| Proportion of unit's assessment | 50% |

**Abstract**
This document describes the coursework for students who are taking CM10025 Programming 2. Students will work in **groups** to produce a piece of software. However, the focus of the coursework is not upon the software's functionality but upon the **process of its production** including the dynamics of working in a team. You are asked to follow an Agile software process to research design, develop and test a Personal Informatics (PI) software system. Weekly labs have been set up for you to attend (in your teams) and report/receive support upon your milestones and adherence to the Agile methodology (and in particular the Scrum component).

## 1 Assignment Information

The purpose of the assignment is to exercise and demonstrate an Agile approach to the software development life cycle (SDLC) of a software engineering project, reviewing the validity of your requirements and designs as your project evolves through its lifecycle. The assignment is a **group coursework** to specify, design and develop a working software system following an Agile software process. Your group will be required to produce a **written report** that explains your analysis of a PI problem, your SDLC approach and products (spec, designs, test plans and results) and a critical analysis of your work.

Your software system will belong to a domain known as Personal Informatics (PI): a field of research and development that focuses on helping people to collect personally relevant information about themselves. Personal Informatics is also referred to as 'living by numbers', 'personal analytics', 'quantified self', and 'self-tracking' [4]. PI data may be gathered for many kinds of user behaviours and feelings, and can be linked to things like sleep, diet, movement, socialising and productivity-related habits. Such 'tracking data' can be collected from many different sources, from manual inputs (e.g. activity logging apps, mood ratings) through wearable fitness trackers (e.g. Fitbit[1]), sensing devices (e.g. smartphones, Arduinos), productivity monitoring tools (e.g.RescueTime[2]) to biometric devices (e.g. glucose monitors[3]).The goal of any PI system is to help a person to monitoring and reflect on their own personal behaviour to find ways of living a betterlife [5]. Designing PI systems to be motivating and relevant requires integration of PI data for users and ways to make patterns memorable so they makes sense over time [6].

The premise of a PI system is that it can demonstrate how a person is living their life so they

[1] www.fitbit.com
[2] www.rescuetime.com
[3] www.gluco-wise.com

can understand themselves better and then decide whether and how they might change things. PI systems have been designed to help motivate physical fitness [1], improving self-management of health conditions [7], and foster environmentally-friendly habits [2]. However, some PI systems can have undesirable effects on their users. Hollis et al. found that emotion or mood trackers can amplify feelings rather than help people to manage them, perhaps because people trust too much in algorithmic treatments of their data without understanding any 'hidden insights' about their inner selves owe no more or less than to the inventiveness of their designers [3].

### 1.1 What to Do - Overview

Your group must work together to research, specify, design, develop and test software for a Personal Informatics software system with your target users as current university students. Your system could be designed to help students with anything that might improve understanding and potentially improving on an aspect of their habitual behaviour. Examples include daily walking, coffee consumption, mood, screen time, or cooking with fresh vegetables. However, your PI system should collect more than one kind of data (e.g. vegetables and mood) so possible relations between kinds of PI data can be explored.

This task will require the consideration of material covered in lectures and additional research in the domain of Personal Informatics. This additional research must involve (1) articles on Personal Informatics to help choose the types of habits and PI data you will gather, (2) interviewing potential users (other students) about your PI concept, and (3) comparing existing Personal Informatics systems that are related to your challenge. You are expected to compare and contrast ideas from these sources. You will find an initial set of requirements in section 2 of this document to bootstrap your process, and relevant peer-reviewed sources are provided in the reference section.

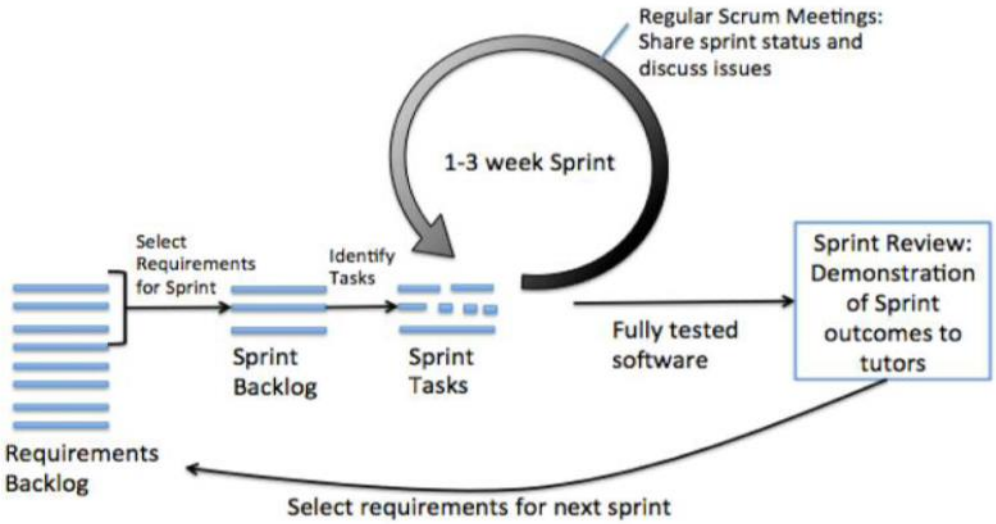
We expect you to make use of the Computer Science subject resources linked from the University of Bath Library at http://www.bath.ac.uk/library/subjects/comp-sci/. You are expected to adjust your understanding of your PI concept as your project develops. Remember that accessing resources such as the ACM Digital Library via this page will ensure you are able to download full-text articles without charge. If you are working off-campus, make sure you set up a VPN connection so that your machine is recognised by the ACM servers as a legitimate UoB user.

### 1.2 Following an Agile Lifecycle: Scrum

Your group must follow an Agile software development lifecycle involving a minimum of one and ideally two sprints. Figure 1 illustrates the Scrum development lifecycle. You will learn about the Scrum SDLC.

You will need to use your requirements specification to prioritise and select requirements for each sprint. Selected requirements will form your sprint backlog, which can then be broken down into tasks for your group members to complete during the sprint. Each sprint should last between 1-3 weeks, and include design, implementation and testing. You should hold regular 'Scrum' meetings during the sprint to share status and discuss issues. At the end of each sprint you should give a short, informal demonstration of your software to the course

tutors and discuss the outcomes of the sprint with respect to its planned backlog and tasks. Your tutor will give you feedback every week on your progress and adherence to Agile principles.




You can find out more about the Agile methodology at:
https://www.agilealliance.org/agile101/agile-basics/

### 1.3 Programming

You will be implementing your designs, carrying out software testing on your evolving system. Programming in this unit is treated only as a means to an end — enabling you to experience a test-driven software development lifecycle. Consequently, your programming ability and code structure will not be assessed as part of this assignment, but you are required to produce a testable software system. Tutors will assist you in translating your software designs into your implementations and in putting your test plans into action. You are expected to take advantage of group meetings with your tutor for sprint reviews, as well as the implementation and testing activities you will carry out.

You are free to choose any platform (e.g PC, Smartphone, Tablet, Arduino) and any object oriented programming language to develop your Personal Informatics software, providing that you feel competent regarding your selection. Course tutors will only be able to provide support for languages and tools that you have been exposed to on your course so far. You MUST NOT choose tools that cannot implement your designs. Doing so would in effect be choosing to fail because you would no longer be engineering a software system. You would be hacking software without integrity.

### 1.4 'Tracking Data' for your Personal Informatics Software

Your system can make use of any source of Personal Informatics data at all. For example, you may simply ask users to manually enter data by responding to questions in a form presented by the user interface. You may also obtain data from sensors (e.g. using Arduino components) or devices like Fitbit (using their APIs).

You are **not** required to develop working interfaces with existing tracking technologies in order to acquire 'real' data (e.g. downloading data from Fitbit API), however you are encouraged to do so if it is within your abilities. You are welcome to 'mock-up' some tracking data to be used by your Personal Informatics software. For example, you might write some software to automatically generate artificial tracking data, simulating inputs from certain devices or services.

## 2 Initial Requirements

We have specified an initial set of requirements for your Personal Informatics system. These initial requirements are to be improved and expanded by your group in order to produce an unambiguous software requirements specification as part of your coursework. If you find any ambiguity in these initial requirements this is intentional. You should add further detail based on your PI research, as described above. All requirements that you include in your specification should be verifiable by test. When incorporating requirements that we have provided into your Software Requirements Specification you should cite 'Coursework Specification' as their source.

We require you to follow an Agile Scrum approach (this is a non-functional requirement on your project). You will be expected to explain in your final report how you dealt with any new or amended requirements sprint by sprint, and how such requirements are reflected in your design and in your testing strategy. As you will learn, a client's needs or the expected environment of operation often change during an SDLC. It is therefore important that software engineers follow an SDLC, which allows flexibility in accommodating new requirements or changes thereto.

Sections 2.1 and 2.2 specify initial Non-Functional and Functional requirements for your system, respectively.

### 2.1 Non-Functional Requirements

| 1. Software Development Process | | |
|---|---|---|
| 1.1 | Your software process must be consistent with Scrum methodology. | Priority: High<br>Author: AMH |
| 1.2 | Your software development should include at least one and ideally two sprints | Priority: Medium<br>Author: AMH<br>Dependencies: 1.1, 1.3 |
| 1.3 | Each sprint should last between 1 and 3 weeks. | Priority: Medium<br>Author: AMH<br>Dependencies: 1.1, 1.2 |
| 1.4 | Must regularly review the functional requirements associated with envisioned system features. | Priority: High<br>Author: AMH |

| 2. Expanding Initial Requirements | | |
|---|---|---|
| 2.1 | Must expand upon all initial requirements, based on your PI research. | Priority: High<br>Author: AMH<br>Dependencies: 3.2 |
| 2.2 | Must expand upon the initial Functional requirements and add additional functionality to the system to deliver features you have chosen to offer. | Priority: High<br>Author: AMH |
| 2.3 | Additional requirements should be established using appropriate requirements gathering techniques. | Priority: High<br>Author: AMH<br>Dependencies: 2.1, 2.2 |

| 3. Background Research | | |
|---|---|---|
| 3.1 | Must read and cite at least three articles in the area of Personal Informatics, at least one of which must be drawn from the reference section of this coursework document. | Priority: High<br>Author: AMH<br>Dependencies: 3.2 |
| 3.2 | Should read and cite at least six articles of any kind. | Priority: Medium<br>Author: AMH |

| 4. Testing | | |
|---|---|---|
| 4.1 | Must adopt a test-driven development approach, including production of test plans. | Priority: High<br>Author: AMH |
| 4.2 | Must provide evidence of testing (e.g. JUnit output) | Priority: High<br>Author: AMH |

**2.2 Functional Requirements**

Note that the priority levels and relationships for the following requirements are not specified (i.e. there are also no MUST or SHOULD statements, dependencies, etc.). These will therefore need to be decided by your group.

| 6. Identifying Trends and Relationships in PI Data | | |
|---|---|---|
| 6.1 | Permit a user to compare their data over time (e.g. to see if they have walked 5% more each day for the last n days, or if their diet is regularly less vegetable-intensive on a Monday). | Author: AMH |
| 6.2 | Permit a user to compare different kinds of PI data within a fixed period of time (e.g. to find out if changes in their mood have followed changes in their diet over the last 14 days). | Author: AMH |

| 5. Viewing and Collecting Tracking Data | | |
|---|---|---|
| 5.1 | Store data on user activity relevant to the chosen PI concept (see Section 3 'Tracking Data' for more detail). | Author: AMH<br>Dependencies: 3 |
| 5.2 | Allow the user to access tracking data that is stored. | Author: AMH |
| 5.3 | Permit the user to manually enter any tracking data which can not be obtained automatically from a tracking device or service (e.g. allowing the user to enter the number of cups of coffee consumed each day, or to enter a rating that represents their mood for the day). | Author: AMH |

| 7. Goals and Achievements | | |
|---|---|---|
| 7.1 | Permit a user to manage targets (goals) for a tracked activity. | Author: AMH |
| 7.2 | Permit a user to set a daily or weekly goal value for a particular data variable (e.g. Steps Goal >= 10000, Productivity Goal >= 60%, Cups of Coffee Goal <= 2). | Author: AMH |
| 7.3 | Permit a user to update or change a goal. | Author: AMH |
| 7.4 | Incorporate a feature to motivate the user to achieve their goals (e.g. scoring points, receiving trophies/ badges, competing with other users) | Author: AMH |

**3 Coordinating Group Work**

You may wish to use technologies that support group work (e.g. Google Docs, Trello, SVN, git/github, pastebin, Facebook groups, Slack) but may only do so if you have positive confirmation from all group members that they can access such technologies from wherever in the world they are working (e.g. Google services are not available in China). Also note that course tutors may not be able to provide assistance for all of the technologies that you choose. It is essential for groups to maintain concise records of their meetings so that they can keep track of their software process and report on its management. You must include copies of your meeting records as an appendix to your deliverable.

You may decide to allocate group tasks to specific individuals or to deal with them as subgroups. Whatever you decide, make sure you decide as soon as you can and make sure that you achieve consensus within your group about the way the group is going to work.
You are advised to communicate and coordinate regularly online as well as holding 'Scrum' meetings with your tutor. All members of the group should participate, and the group should keep 'minutes' of its meetings. 'Minutes' include the date, time, who attended, issues discussed, who agreed to do what by the next meeting ("actions"), and the planned date, time and location/platform for the next meeting.

**This is a group-based project the output of which is one report that represents the work of the entire group**. Some sections of the report will be contributed to by single individuals, some by multiple group members. All members of the group must participate in the

assignment. No group has the right to exclude the work of one or more members from the submitted work of the group.

**3 Final Report Specification**

There is one written deliverable for this group coursework, which requires submission of a single report. You are strongly advised to have completed your work well in advance of the deliverable deadline, so that the group as a whole is in a good position to finalise their work in time to make the deadline. Your attention is drawn to the penalty for late submission of coursework, as set out in the Student Handbook and the associated rules and regulations.

Your final group report will document the complete analysis, design, implementation and testing of your Personal Informatics system, as well as your project management activities as they have implemented your software process.

An electronic version (pdf format, do not zip) must be uploaded on Moodle by the deadline.

Please ensure that all fonts used are legible and appropriate. Attention should be paid to report presentation: all pages must be numbered, all diagrams must be clearly labelled, and suitable headings and sub-headings used. Fonts in figures should be no smaller than the font in their captions. Number your sections but do not go deeper than four levels of hierarchy (e.g. Section 1.2.3.4).

The expected length of the main body of the report is 20 pages; do not exceed this limit as additional pages will not count toward the assessment. Additional material (such as questionnaires, interview transcripts, details of meetings, etc.) may be included in clearly titled appendices. The cover page, abstract, table of contents, GCF, appendix and references do not count toward the page limit. All sources of information should be correctly referenced and a full list of these references provided at the end of the report.

A guide to the format and salient elements of the report is given below. We have included a guide as to how you might allocate your total "budget" of up to 20 pages between sections of the report. The actual distribution of pages within each section is up to you, as long as you do not exceed 20 pages for the main body of your report.

**Cover Page**

This should include the unit code (CM10025), title of your report, group name and number, the date and a table of authors (the members of your group) on this page. For example,

| Group Member | Username | Degree Course |
|---|---|---|
| M. Ahmed | msa123 | BSc Computer Science (yr. 1) |
| T. Black | tdb567 | BSc Computer Science and Artificial Intelligence (yr. 1) |
| R. Blanc | rsb324 | MComp Computer Science (yr. 1) |

**Title and Abstract**
A single page. The first line should be the report title. On the second line should be a comma separated list of authors. An “abstract" of the document - about 200 words summarising the problem, key features of your designed solution, the features so far implemented by your three completed sprints, and directions for further evolving your software system.

**Table of Contents**
A table containing the title of each section and subsection of the report, with page numbers. Numbering should start from the title page.
Importantly, you should also indicate who were the authors of each section or subsection. In most cases, the authors will be a subgroup who worked together to prepare it.

**Introduction (2 pages)**
An account of the problem your software system is intended to address, including a description of the problem area (Personal Informatics) supported by relevant literature (See NFR3.1 and NFR3.2), what can make a PI software system effective for its users, and a clear statement of the main idea behind your particular PI software system.

**Agile Software Process Planning and Management (2 pages)**
This section must describe how you carried out your project as an Agile process, including how you planned and tracked your sprints, and how Scrum meetings were used to manage the evolution of your idea and the features you decided to include in your software system.

**Software Requirements Specification (5 pages)**
This section should include:

- An account of how you established your detailed system requirements (building on the initial requirements).
- A description of the specific domain of your personal informatics system, linked to the viewpoint of students as your target users.
- Use cases to illustrate the services your PI system is intended to offer to relevant actors, including target users.
- A set of functional and non-functional requirements for your system, organised into sensible groups and following the guidance given you in lectures (e.g. relationships and priorities).

**Design (5 pages)**
This section should include:

- A set of UML models describing the low-level design of your system (class models, sequence diagrams, etc.)
- A justification of your chosen design. This should explicitly show how your design meets both functional and non-functional requirements, making direct reference them and the rationale behind your design as it embodies your PI idea.

Make sure you include explanatory text with your design models: the meaning of a diagram

is very rarely self-evident because it depends on an explanatory frame for interpretation. So you must make sure to express in words what you wish the reader to understand by them.

**Software Testing (Verification) (2 pages)**
This section should include:

- Testing plans, indicating how you planned to perform verification of your PI system. These plans should reflect your requirements and design work, and be completed before any implementation (coding) work begins.
- Evidence of testing - test case results.

**Reflection and Conclusion (4 pages)**
This section should include two main sections:

- A critique of your software system's requirements specification, design and testing, including what might be improved and why you feel it would be better with these changes.
- A critical reflection on the group's software process, including evidence of Agility in your software process, evidence of having evolved your requirements to reflect changes in understanding of problem and viability of designs.

**References**
A list of references to articles from which you have gathered information as part of your research and have cited in the body of your report. We recommend that you consult the University guide to citing references, which is available from the University library.

**Appendices**
You must include your one-page final GCF as the first appendix to your report. You should include records of your group meetings (minutes) as the second appendix. You may include transcripts of interviews and/or other evidence of primary research you have conducted as the third appendix.
Also, you may include earlier design models to evidence how your ideas have evolved as an appendix. if you wish to include, as well as any unit tests or additional test runs you might wish to include with your report.

## Marking Scheme on the next page.

## 4 Marking Criteria and Feedback

Note that you will be receiving verbal feedback every week from your tutor in your scheduled tutorial and/or lab session. Marking of your final deliverable will be in accordance with the following criteria:

### 4.1 Agile Process (30 marks)

Evidence of your Scrum activity, including how you planned and tracked your sprints, evolved your requirements, your designs and refined your test cases.
Critical reflection on strengths, weaknesses and proposed future actions to improve your Agile process, team spirit and coordination.

### 4.2 Problem Analysis and Requirements Specification (20 marks)

A clearly written account of your analysis of the problem to be solved, including the articles and interview evidence to support your claims about what can make PI systems effective for their users.
Critical reflection on understanding of problem and future directions.
A well-organised set of requirements for your system that conform to the principles you have been taught, that are coherent with your idea for an effective PI system.
Use Cases with explanatory text, illustrating services to be offered by your PI software system.

### 4.3 Design (20 marks)

UML models describing the structural and dynamic OO design of your system (class models and sequence diagrams), each with clear explanatory text.
Rationale for your design, with explicit connections to functional and non-functional requirements.

### 4.4 Software Testing (10 marks)

Description of per-sprint test-driven strategy, including how you managed to coordinate, run, analyse and respond to your test results as members of remote teams.
Evidence of testing e.g. unit test results.

### 4.5 Reflection and Conclusion (20 marks)

A critical reflection upon the software that you have developed including what works well, what works not so well and what improvements you would make if you had more time.
Also, include some further reflection upon the Agile methodology. How well did the Agile process work for you as a group? Would you adopt the method for future projects? What worked particularly well? What didn't work for your group? Are there any suggestions, based upon your experience, on how things could be improved?

## 5 Plagiarism and University Regulations

- Use pdf format for your report component of your submission. Do not zip your submission. You will submit your final report by uploading it to the CM10313 moodle page.
- Plagiarism is a serious offence. You must therefore make sure that you understand what plagiarism is and how you can avoid it. Moodle integrates a plagiarism detection

service(https://www.bath.ac.uk/guides/tools-help-and-advice-to-support-technology-enhanced-learning/#urkundoriginality-detection).

Please check the following websites for more information:
http://www.bath.ac.uk/library/help/infoguides/plagiarism.html
and
https://www.bath.ac.uk/corporate-information/academic-integrity-statement/ .
• Late submissions will be subject to the University regulations on penalties to be applied.

**References**


1. Fritz, T., Huang, E. M., Murphy, G. C., and Zimmermann, T. Persuasive technology in the real world: a study of long-term use of activity sensing devices for fitness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2014), ACM, pp. 487–496.
2. He, H. A., Greenberg, S., and Huang, E. M. One size does not fit all: Applying the transtheoretical model to energy feedback technology design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2010), CHI '10, ACM, pp. 927–936.
3. Hollis, V., Pekurovsky, A., Wu, E., and Whittaker, S. On being told how we feel: How algorithmic sensor feedback influences emotion perception. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 2,* 3 (Sept. 2018).
4. Li, I., Dey, A., and Forlizzi, J. A stage-based model of personal informatics systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2010)*, ACM, pp. 557–566.
5. Li, I., Medynskiy, Y., Froehlich, J., and Larsen, J. Personal informatics in practice: improving quality of life through data. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems (2012),* ACM, pp. 2799–2802.
6. Rapp, A., Marcengo, A., Buriano, L., Ruffo, G., Lai, M., and Cena, F. Designing a personal informatics system for users without experience in self-tracking: a case study. *Behaviour & Information Technology 37*, 4 (2018), 335–366.
7. Tollmar, K., Bentley, F., Viedma, C., and Libertyville, I. Mobile health mashups. *Proc. Pervasive Health 12* (2012).