Queen Mary
University of London

School of Electronic Engineering and Computer Science

EBU5304 – Software Engineering Group Project

30% coursework. [*Student groups are allocated by the module organiser*.]

A campus scooter sharing system

-developing the software using Agile Methods

1. General information

In the next few weeks, your team will be required to develop the software of a campus scooter sharing system using Agile methods. Iterations should be planned and Agile methods should be used in all activities, from requirements, through to analysis/design, implementation and testing.

It should be noted that determining the software requirements is one of the most important and complex phases in any development project. The given specification contains a lot of noises—requirements are described in an abstract and ambiguous way. You should apply requirement finding techniques and Agile methods to extract the relevant information at appropriate level. Most importantly, you need to prioritise the features that are implemented in accordance with both ease of implementation and meeting customer requirements. As Agile is designed to adapt to change, do expect that new requirements or change of requirements will be available during the development stage. As in real software though, there may be more details you want to know that are missing from the given specification. You can make your own assumptions but **do NOT over design**. Keep your design **SIMPLE**. Bear in mind that there is no absolute right answer – your solution may be perfectly appropriate.

Handout release date: **Tuesday, 5th March 2019**
First QM+ submission (Product backlog): **Friday, 22nd March 2019**
Final QM+ submission (Report and Software): **Friday, 31st May 2019**
Demonstration/feedback 1: workshop session in teaching week 2 (25-29 March)
Demonstration/feedback 2: workshop session in teaching week 3 (6-10 May)
Demonstration/feedback 3: workshop session in teaching week 4 (3-6 June)

Marks returned: Approximately 2-3 weeks after the final demonstration.

2. Specification of project

Queen Mary University of London plans to build a scooter sharing system in the Mile End campus to provide faster on-campus travel. In total there are 15 scooters on campus and the scooters must be picked up and returned to docking stations. There are 3 docking stations A, B and C around campus as shown in table 1 and marked in the campus map (see appendix). Each docking station has 8 scooter slots to lock the scooters in.

A	Library
B	Informatics Teaching Laboratories
C	Village Shop

Table 1 docking stations

All registered students and staff members can pick up and return a scooter by scanning their campus id cards at the any docking station. It is free to use however the scooter must be returned within 30 minutes and the total usage must not exceed 2 hours a day, otherwise a fine should be issued.

You are an Agile software development team that is responsible for developing this campus scooter sharing system. The high-level specifications and requirements are described as follows:

2.1 Docking station

The docking station has 8 scooter slots and each slot is equipped with a light. The docking station has a card reader for scanning campus id card and a simple screen to display messages.

Pick up a scooter

A registered user can pick up a scooter at any docking station. After scanning the campus id card, the station should release one scooter from its slot and the light at the slot should be flashing to inform the user. Once the user takes the scooter out of the slot, the light should stop flashing. If the user does not take the scooter after 1 minute, the scooter should be locked to the slot automatically. The screen should display relevant friendly messages during the operation.

Return a scooter

A registered user can return a scooter at any docking station that has empty slots. After scanning the campus id card, the station should make one empty slot ready and the light at the slot should be flashing to inform the user. Once the user puts the scooter into the slot, the scooter should be locked and the light should stop flashing. If the user does not put in the scooter after 1 minute, the light should stop flashing and the user should be unable to put the scooter in. The screen should display relevant friendly messages during the operation.

Note: For demonstration, you do not need to actually implement the scanning card function. You may simply use a text field to enter the QM number to simulate the scan. For the taking/putting scooter from/to the slot, you may use a button to simulate.

2.2 Management system

Register new users

QMUL Students and staff members are eligible to use the scooters however they must be registered to the system. To register, they must bring their campus id cards to the administration office. Once verified, their QM number, full name and email address should be added to the system.

Note: For demonstration, you must use real user information of your group members: QM number, full name and QM email address.

Fine

It is free to use the scooter however 1) each time a user must return a scooter within 30 minutes. 2) the total usage must not exceed 2 hours a day. A fine of £100 should be issued if a user fails to timely return the scooter or exceeds the maximum daily usage. If a fine has not been paid, the user should not be able to use the scooters.

Note: For demonstration, you do NOT need to implement the actual payment function. You may use a button to simulate the payment.

Usage

The system should be able to check the usage of each user at any time and should send each user a weekly usage report via email. The system should be able to monitor the status of each docking station, e.g. how many scooters are currently available at each station and how many scooters are currently in use.

Note: For demonstration, you do not need to actually implement the sending email function. You may simply use a message (or generate a text file) show the email contents.

2.3 Other requirements

- Basic restrictions and error checking must be considered: for example, the QM number should be 9 digits; email address must in the form of xxx@xxx; etc.
- It should be easy to use: that is, the user should be able to operate the system with common sense or with simple instructions.
- It should be user friendly: it should display messages promptly to user during the operation.

- The software must be developed using Java as a stand-alone application. Java SE 8 or above should be used.
- All input and output files should be in simple text file format. You may use plain Text (txt), CSV or XML. Do NOT use database.
- Your design must be flexible and extensible, so that it can adapt to continuously changing requirements and can be used in a general market in the future. E.g. adding more slots to the docking station; adding more docking stations; charge the use of the scooter; etc.
- Your design of the software must be capable of adapting to such future changes. That is, when developing a new system, you should be able to reuse the existing components. When adding new features to the existing system, you should make the least impact on the existing code.
- You only need to consider the logical information flow for implementing some functions. For example, you do NOT have to consider the actual communication between docking stations and the management system, you may assume they are all running in the same computer.
- You should create a few simple GUIs to represent the docking stations and the main management system. A possible example of this is illustrated in **Figure 1**:

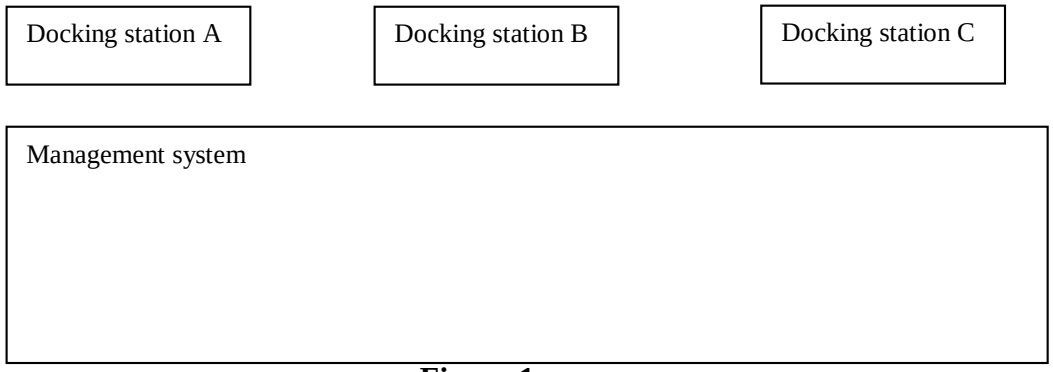

Figure 1

Your tasks are to define detailed requirements, design, develop and test the above described software using Agile methods. For any details of the software or operation which is not clearly stated, ***you may make your own assumptions***. In your report, make it clear where you have made assumptions. Feel free to design the software as long as it satisfies the basic requirements, but **do NOT over design it**.

3. Agile project management

Each coursework group has 6 students[1]. You are the Agile team working together to complete the coursework. All students in a group must work on all aspects of the project, in order to obtain full software engineering skills.

You should use the techniques you have learnt in the lectures to manage the project, e.g. Scrum, daily stand up meetings, working around a table, scrum master and one hand decision making etc.

4. First QM+ submission: 22nd March

The first QM+ submission is **Product Backlog**.

- Use the template provided on QM+, feel free to modify it to meet your needs.
- The submission must be an EXCEL file.
- Only the group leader should submit the excel file.
- The file must be named **ProductBacklog_groupXXX. xlsx**, where **XXX** is your group number.

5. Final QM+ submission: 31st May

The final submission includes a short report and software.
The short report should contain the following parts:

i. Project management
- The project management in your team working. E.g. using project management techniques, planning, estimating, decision making and adapting to changes.

ii. Requirements
- Apply the requirements finding techniques.
- Describe any changes of the product backlog since the first submission.
- Iteration and estimation of the stories.

iii. Analysis and Design
- A design class diagram describing the design of the software classes in the software, show the class relationships. Note that your design should *address the issue of reusability of software components.* You should provide clear justification for your proposed approach and show that your design is adaptable to change where necessary.
- Discuss the design of the software.
- Discuss the extent to which your design and the code that implements it meets the main design principles of programming.

iv. Implementation and Testing

[1] Due to the size of the cohort, some groups may occasionally have 7 students instead.

- Discuss the implementation strategy and iteration/built plan.
- Discuss the test strategy and test techniques you have used in your testing.
- Discuss the using of TDD. Note: TDD is not required for developing the whole software, however, you should try to use TDD to develop a few programs.

v. All reports should include a list of references in the appendix.

vi. Main screenshots of the system should be included in the appendix (**Note**: Convert them to JPEGs).

Final report must be in PDF format (maximum 15 pages, excluding the Appendix). This must be named **FinalReport_groupXXX.pdf**, where **XXX** is your group number. Only the group leader should submit the file.

The software should contain the following parts:

i. A working software application written in Java. All main functionality should be implemented. Code should be well documented.

ii. A set of test programs using Junit as an example of using TDD.

iii. JavaDocs.

iv. User manual.

You must submit a ZIP format file containing all the .java files of product programs and test programs, Javadocs, user manual and a Readme file to instruct how to set up or configure and run your software. Do not include .class files, as your programs will be re-compiled. This must be named S**oftware_groupXXX.zip**, where **XXX** is your group number. Only the group leader should submit the file.

6. Demonstration

Demonstration 1: during the workshop session in teaching week 2 (25-29 March)

- Product backlog
- Paper prototype
- Up-to-date iteration of WORKING software (can be only a few programs)

Demonstration 2: during the workshop session in teaching week 3 (6-10 May)

- Up-to-date iteration of WORKING software
- Unit Testing

Demonstration 3: during the workshop session in teaching week 4 (3-7 June)

- Final version of WORKING software
- Integration Testing

ALL group members MUST attend all the demonstration sessions. For each demonstration, it is OK if only some features are implemented, your code is incomplete or has bugs – just show your latest built of the working system, you still could receive full marks of the demonstration. Remember we look at setting a priority for the features that are implemented in accordance with both ease of implementation and meeting customer requirements. Detailed instructions of each demonstration will be sent out in due course.

7. Important notes

Only coursework materials submitted via QM+ will be accepted.

Although real software would require more advanced features (e.g. consideration of security, database, data synchronisation etc) this would distract from the core software engineering skills. The following guidelines MUST be followed.

- **Standard-alone application:** You must develop a stand-alone application and ignore any networking concerns. Students must NOT write HTML, JavaScript, servlets, JSPs, sockets, ASPs, PHPs, etc … This is NOT a network programming module.
- **NO database implementation**. Students should develop this application without using a database, i.e. do no not use MySQL database etc. Students should concentrate on their software engineering skills without being concerned by more precise deployment issues. **Hint**: Students should think about the use of Java interfaces for the database (or proxy database) access.
- **Code development.** Code should be written for maintainability and extensibility – it should both contain Javadocs and be clearly commented.
- **Code delivery.** A Readme file should explain clearly how to install, compile (i.e. what to type at the command line) and run the code. All code should run from the command line and MUST NOT require users to install any extra software (e.g. database, Eclipse or any other IDE) or extra Java libraries.
- **Key Points of report.** Focus on quality, not quantity – please pay attention to the page limit. Examiners will be impressed by groups who can criticise their solution and indicate how this can be improved in future iterations. Students should take care over the presentation of the report (and check for spelling and grammar mistakes) – they should imagine that this report will be presented to the client. Students should not spend hours and hours concentrating on making the most beautiful GUI, no extra marks will be gained. The focus of this work is software engineering – correct functionality and elegance of code (classes that do only one thing, methods that do only one thing, code that is not duplicated, delegation, i.e. following the principles outlined in the course) are much more important.
- **Key points of Participation and Achievement.** If students are not turning up to meetings or doing any work, then the module organiser need to be informed **immediately**. The coursework project is marked out of 100. Individual participation/achievement will be evaluated through the demonstration sessions. If a student has not participated at all in the group, they will get 0% of group marks.

8. Marks breakdown (approximate)

Group mark (maximum 100 marks)

Demonstration 1: 30%

- Ability to extract and define the software requirements using Agile techniques.
    - Use of appropriate fact finding techniques.
    - Correctness of writing user stories.
    - Correctness and completeness of product backlog.
    - Quality of prototype.
- Progress to date and project management

Demonstration 2: 20%

- Ability to refine the requirements through analysis and Ability to design high quality software
    - Quality of design
- Unit testing
- Progress to date and project management

Demonstration 3: 20%

- Quality of the final version software
- Integration testing
- Project management

Final submission: 30%

- Correctness of Java code – the code must match the design. *If the code does not match the design* (or if there is no correspondence between the design/code), then ALL these marks will be lost.
- Quality of Java code
- Testing: - appropriate test strategy
- Quality of report

Individual mark

Individual marks will be given according to the participation in the group: Quality of work performed and Understanding of the performed work. At each demonstration, each student will be evaluated through explaining the work and answering the questions. Grade will be awarded as below:

A	Satisfactory	Receive 100% group marks
B	Unsatisfactory	Receive 50% of group marks
C	No contribution or absent	Receive 0% of group marks

You, AS A GROUP, are responsible for managing any issues and for completing all of the tasks.

Please use the messageboard on QMPlus for enquires and discussions; do not email the lecturers unless it is a personal related issue.

Mile End Campus

Educational/Research

ArtsOne	37
ArtsTwo	35
Arts Research Centre	39
Bancroft Building	31
Bancroft Road Teaching Rooms	10
Peter Landin Building (Computer Science)	6
Engineering Building	15
G.E. Fogg Building	13
G.O. Jones Building	25
Geography	26
Graduate Centre	18
Informatics Teaching Laboratories	5
Joseph Priestley Building	41
Library	32
Law	36
Lock-keeper's Cottage	42
Occupational Health and Safety Directorate	12
People's Palace/Great Hall	16
Queens' Building	19
Scape Building	64
Temporary Building	61

Building construction site	**14**
Building closed for major refurbishment	**4**

Residential

Albert Stern Cottages	3
Albert Stern House	1
Beaumont Court	53
Chapman House	43
Chesney House	45
Creed Court	57
France House	55
Feilden House	46
Hatton House	40
Ifor Evans Place	2
Lindop House	21
Lodge House	50
Lynden House	59
Maurice Court	58
Maynard House	44
Pooley House	60
Selincourt House	51
Varey House	49

Facilities

Advice and Counselling Service	27
Bookshop	22
Canalside	63
Careers Centre	19
Clock Tower	20
CopyShop	56
The Curve (R)	47
Disability and Dyslexia Service	31
Drapers' Bar and Kitchen (R)	8
Ground Café (R)	33
The Nest	24
Housing Hub	48
IT Services	19
Mucci's (R)	29
Occupational Health Service/ Student Health Service	28
Octagon	19a
Portering and Postal Services	17
Qmotion Health and Fitness Centre Sports Hall	7
Santander Bank (£)	62
Security	38/54
St Benet's Chaplaincy	23
Student Enquiry Centre	19
Students' Union Hub	34
Union Shop (R)	9
Village Shop	52
Westfield Nursery	11

Information

Visitors who require further information or assistance should please go to the main reception in the Queens' Building.

The smoking of cigarettes or tobacco products are **only** permitted at designated smoking areas / shelters indicated on this map.

Electronic cigarettes permitted on outside spaces **only**.

These premises are alarmed and monitored by CCTV; please call Security on +44 (0)20 7882 5000 for more information.

Key

- Library/bookshop
- Fitness centre
- (R) Refreshment: Bar/Eatery/Coffee place
- (P) Staff car park
- Bicycle parking
- (BL) Bicycle lockers
- (£) Cash machine
- Smoking area / shelter