|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | |  | |  | |  | |  | |
| PRCS251 Coursework Specification  PRCS251 Computer Science Integrating Project |
| Team: Dr Shirley Atkinson, Dr Nigel Barlow, Dr Torbjorn Dahl, Dr Mark Dixon, Mr John Forde, Dr Stavros Shiaeles, Dr Liz Stuart, Dr Ismini Vasilieou |

Table of Contents

1. PRCS251 Coursework Specification 1

1.1. Introduction 1

1.2. Learning Outcomes 1

1.3. Staff Team 1

1.4. Group Working 3

2. Deliverables 4

2.1. Key Dates 4

2.2. Deliverable Descriptions 5

2.3. Marking Criteria 9

3. Scenario 12

3.1. Pizza (or food) delivery service software 12

3.2. System Architecture 12

3.3. System Requirements 13

3.4. Desktop Application 14

3.5. Web Application 15

3.6. Mobile client 15

3.7. Asynchronous 17

3.8. Security 17

4. Project Management 18

4.1. Introduction 18

4.2. Progress meetings 18

4.3. Final Report 18

5. Requirements Analysis and Design 20

5.1. Introduction 20

5.2. Final Report 20

6. Database Design 21

6.1. Final Implementation 21

6.2. Final report 21

7. Human-Computer Interaction 22

7.1. Introduction 22

7.2. Cognitive Walkthrough 22

7.3. Usability evaluation 23

7.4. Final report 23

7.5. Usability of the final product 24

8. Security 25

8.1. Introduction 25

8.2. Final Implementation 25

8.3. Final Report 25

9. Software Engineering 26

9.1. Introduction 26

9.2. Final Report 26

9.3. Implementation 26

10. Peer Assessment 27

10.1. Introduction 27

10.2. Criteria 28

# PRCS251 Coursework Specification

## Introduction

This document outlines the coursework for the PRCS251 Computer Science Integrating Project. The coursework is worth 100% of the module. You are required to work in a team to carry out a substantial computing project as outlined in the given scenario. Your team will create a collection of materials to show you have collectively achieved the learning outcomes for the module.

You are to work in teams of 5. You must arrange yourselves into teams using the Coursework Groups option on the Module DLE page by the start of the module.

There is a prize awarded for an excellent second year computing project. These are normally noted at graduation.

## Learning Outcomes

By the end of the module provided you have engaged with the subject material and the teaching you should be able to:

1. Work as part of a team in the capture of requirements for, design of, and implementation of a substantial group project.
2. Apply knowledge gained from the modules studied so far on the BSc (Hons) Computer Science programme to solve a substantial problem.
3. Contribute to the presentation of the results of the project.

## Staff Team

This module provides you with an opportunity to refine and develop your existing skills. You are expected to develop your skills further depending on the nature of your team’s proposed solution. This will require you to undertake further, independent study. However, there is a team of staff available to help and support you during this module, each with their own specialism. Details are given below:

|  |  |  |
| --- | --- | --- |
| Staff member | Subject | Email |
| Shirley Atkinson  (Module Leader) | Software Engineering (.NET)  Project Management | [Shirley.atkinson@plymouth.ac.uk](mailto:Shirley.atkinson@plymouth.ac.uk) |
| Nigel Barlow | Software Engineering (Java/Android) | [Nigel.barlow@plymouth.ac.uk](mailto:Nigel.barlow@plymouth.ac.uk) |
| Torbjorn Dahl | Software Engineering (Java/JSP) | [Torbjorn.dahl@plymouth.ac.uk](mailto:Torbjorn.dahl@plymouth.ac.uk) |
| John Forde | HCI | [John.forde@plymouth.ac.uk](mailto:John.forde@plymouth.ac.uk) |
| Nick Outram | Software Engineering (iOS) | [Nicholas.outram@plymouth.ac.uk](mailto:Nicholas.outram@plymouth.ac.uk) |
| Stavros Shiaeles | Security | [Stavros.shiaeles@plymouth.ac.uk](mailto:Stavros.shiaeles@plymouth.ac.uk) |
| Liz Stuart | HCI | [Liz.stuart@plymouth.ac.uk](mailto:Liz.stuart@plymouth.ac.uk) |
| Ismini Vasileiou | Databases and Requirements Engineering | [Ismini.vasileiou@plymouth.ac.uk](mailto:Ismini.vasileiou@plymouth.ac.uk) |

Please ensure that you request any face-to-face meetings via email first.

Please note that we cannot be on call should you decide not to attend a lecture or workshop. It is not acceptable to email the academic afterwards to request a summary of key points. Those with valid extenuating circumstances (EC) should use the EC’s process as appropriate.

## Group Working

Successful group working does not just “happen”. It can be stressful and requires effort and compromise. Not everyone works in the same way or at the same pace and you will undoubtedly need to find a way to negotiate these issues. It is recommend that you look at the resources available on the Learn Higher website on group working - <http://archive.learnhigher.ac.uk/groupwork/help_for_students.php>

# Deliverables

## Key Dates

|  |  |  |
| --- | --- | --- |
| Deliverable Number | Deliverable | Date |
| 1 | Initial Planning Document Discussion meetings | Week commencing Monday 13th February 2017. Appointments to be scheduled via the DLE. |
| 2 | Initial Planning Documentation submission (online). | Thursday 23rd February 2017 by 4pm. |
| 3 | Architecture Demo 1 | Monday 13th February 2017 or Thursday 16th February 2017 (Times to be arranged via the DLE scheduler) |
| 4 | Architecture Demo 2 | Monday 6th March 2017 or Thursday 9th March 2017 (Times to be arranged via the DLE scheduler) |
| 5 | Mock Usability | Week commencing 13th March 2017. Liz Stuart will communicate exact details via the DLE. |
| 6 | Live Usability | Week commencing 27th March 2017. Exact details will be communicated via the DLE by Liz Stuart |
| 7 | Architecture Demo 3 | Monday 27th March or Thursday 30th March 2017. (Times to be arranged via the DLE scheduler) |
| 8 | Final Team Presentations | Monday 15th May 2017. Times to be arranged via the DLE scheduler. |
| 9 | Final Documentation Submission | Thursday 11th May 2017 by 10am |

## Deliverable Descriptions

All deliverables are mandatory for passing the module. Where attendance is mandatory, if you have valid Extenuating Circumstances at the end of the module, these will be taken into account for the marking criteria.

1. **Initial Planning Document Discussions.**

These meetings are timetabled with the team via the DLE scheduler. At these meetings you will present your draft initial planning document materials for discussion and feedback. Whilst attendance is not mandatory for all the team, notes will be made of these meetings detailing who attended and the topics discussed.

1. **Initial Planning Documentation**

This is an online submission of one PDF document that describes how your group has carried out the initial project setup. The contents of your initial project plan should be as follows:

* **Team Description**: Introduce your team giving details of your role (or schedule of roles) allocated if appropriate. Include any agile roles you deem fit. In addition, include contact details and preferences, eg: email only, phone, or skype.
* **Tools** : include details about the repository you are using for your code, remembering to allow permissions for the staff team to be able to access it. Provide a link to the project management tool you are using.
* **Initial requirements analysis and design**: Provide a product backlog with your estimated story points. Provide your use case diagram(s) for the whole system. Select TWO use cases which will be demonstrated and produce design level sequence diagram, state machines and class diagram.
* **Sprint plan**: An outline of which tasks will be addressed under which sprints. An outline of when the sprints will happen.
* **Communications plan**: An outline of how you will be managing your communications within your team and with your “client”. Provide a schedule of regular meetings, when and where they occur. Ensure this maps to what is illustrated in your project management tool.
* **Risk Analysis**: Consider what risks might occur, determine a course of action for each. Classify the risks as high/medium/low probability and high/medium/low impact. Choose to:
  + **Avoid the risk by taking suitable action**. Eg: avoid loss of code by devising a suitable backup).
  + **Accept the risk**. Specify a contingency plan to bring in if the risk does occur.
  + **Accept the risk and deal with the problem** at the time it occurs. Suitable for low impact and low probability.

No marks are allocated specifically to this document and this allows formative feedback on the overall project plan.

1. **Architecture Demo 1**

The three architecture demonstrations allow the team to give you feedback on your fundamental technical solutions. The connections between the database, desktop application and mobile application used in the demonstrations should be minimal as the demos’ only seek to establish that you have managed to get the fundamental connections working. **Attendance at these is mandatory**.

Times to be scheduled via the scheduler on the module DLE.

The first demo should show that you have been able to connect your desktop application to the University’s Oracle database server and that you can retrieve data from the database and show it on your desktop.

1. **Architecture Demo 2**

This second demo should show a web-client running on your laptop that extracts data from your Oracle database through your web-server.

1. **Mock Usability**

The usability team will issue details of the mock usability sessions separately. Details of the usability evaluation can be found in the supporting materials labeled “**Human-Computer Interaction**”.

1. **Live Usability**

As above

1. **Architecture Demo 3**

The mobile-database prototype should show a mobile client that extracts data from your Oracle database through your web-service.

1. **Final Team Presentations**

The purpose of the presentation is for the module team to see the software you have developed and to assist the team in assessing the project as a whole. You must be present to obtain the module mark, the exception being if you have extenuating circumstances for the module.

Your group will have 15 minutes for a presentation and demonstration of the software created. The presentations must be booked via the scheduler on the DLE. Two rooms have been booked throughout the presentation period. This allows the staff team to be in one room with one group presenting while the other group prepares in the other room. At the end of each presentation the staff team will move between the rooms.

Your presentation will be evaluated for the following:

* Introduction, structure and summing up.
* Content – depth and clarity.
* Style – presenters, handover, slides.
* Quality of demonstration and software.

Questions will be asked during the presentation. This is not implied criticism but a way of allowing the team to fully explore the implementation.

The presentation must be professional and business-like. The focus should be on the functionality and how you have achieved the users goals. Ensure you use at least two prepared user stories and use pre-prepared demonstration data. Copy and paste where necessary. Explore how to demonstrate a mobile application that does not require passing around a small phone. Ensure there is enough data within the database to sufficiently illustrate your application.

1. **Final Documentation Submission**

One zip file for the team should be submitted online via the DLE by the deadline indicated. Your documentation should contain as a minimum:

* A link provided to your software code repository showing the code that has been implemented. DO NOT SUBMIT CODE and do not change code after the deadline.
* Presentation materials to be presented, and
* A final report

The final report must contain an introduction and a conclusion. It must also have the following sections within the content body:

* Project Management
* Requirements Analysis and Design
* Database
* Security
* Software Engineering
* HCI

Further details on expectations for each of these sections can be found in the supporting materials.

## Marking Criteria

Your individual marks will be calculated using your group mark multiplied by your peer mark.

Group mark x peer mark % = individual grade

Eg: 55 x 80% = 44

Your peer mark will be calculated using the the OPAS system. Further details of how the peer assessment system works is given in the section “**Peer Assessment**”.

The marks for this module are allocated by looking at all the deliverables as a whole. Therefore your final report, implementation and presentation will all be examined together. The group marks will be allocated as follows:

|  |  |
| --- | --- |
| Category | % |
| Evidence of success in applying agile project management principles. This will relate to clear application of agile project management to your project. The marks will relate to the extent to which you have identified and carried out the appropriate activities. | /15 |
| Evidence of success in applying appropriate requirements engineering activities. This will relate to how the analysis and design aspects of the project were created and then applied to the work carried out. | /15 |
| Evidence of success in applying HCI principles to the project. This will relate to how the usability studies were carried out and the application of the findings from them to the project. | /15 |
| Evidence of success in applying an understanding of database analysis, design and implementation to the project. This will include how you designed, implemented and evaluated your databse. | /15 |
| Evidence of success in applying an understanding of Software Engineering. This will include how you designed, implemented and evaluated your software code. | /20 |
| Evidence of success in applying an understanding of Security. This will include how you designed and implemented security features within your project. | /15 |
| Evidence of success for generic skills. This will relate to your communications activities, how you have conducted your selves as team members and how you presented your team and software. | /5 |
| Total | /100 |

|  |  |
| --- | --- |
| Mark | Grade Criteria |
| Poor  (0 – 39%) | The quality of work is simply not good enough. The evidence provided was inadequate and indicated a lack of planning and/or robust activity. The evidence suggested there was a failure to achieve the expected standard for the topic. Work of this quality would not be acceptable in a professional environment. |
| Satisfactory  (40 – 49%) | The quality of work is good enough but still requires further rework to get it to a better standard. The evidence provided suggests an adequate level of success. There has been a reasonable level of achievement demonstrated. The evidence may have been poorly delivered and showed scope for further improvement. |
| Fair  (50-59%) | The quality of work submitted suggests some aptitude for delivering software projects of this nature. The evidence provided was clear and sufficient. Some of the content suggests there are still areas for further improvement or in some cases the evidence may have been incomplete. You have demonstrated a fair understanding of the issues and shown a fair amount of thoroughness, effort and independent learning. |
| Good  (60-69%) | The quality of work submitted is good. There is clear evidence provided of well-planned and implemented deliverables. The report content is professional, the software delivered is of a good standard and the evidence of interaction between the team is of appropriate, professional standard. There is a clear demonstration of a good understanding of the domain issues, personal thoroughness amongst the team, a good effort and independent learning. |
| Excellent  (70-100%) | The quality of work is outstanding with no significant flaws. The evidence provided is excellent and robust. The deliverables are all very high quality demonstrating a high level of understanding of the domain, personal thoroughness, effort and independent learning. |

# Scenario

This section outlines the scenario that is to be used to inform your analysis, design and development.

The description here is not exhaustive in terms of the detail. You are expected to flesh out the scenario yourselves and add further details as you feel is necessary. This must be done at the start of the project and be included in your **initial planning documentation**.

You must maintain a list of assumptions that will justify your decisions. These must be discussed during the fortnightly progress meetings.

## Pizza (or food) delivery service software

Your team has been contracted to produce an integrated system for a company that sells takeaway pizzas. Probably start with a small-scale local set up with just a single take away.

There will be multiple components to the final application. So far as users are concerned they will see at least one of three applications.

* A free-standing Java desktop application for the staff in the takeaway.
* A web browser delivered application that will be the interface to the system that the general public will see.
* A mobile application for the people on motorbikes (or in some cases pedal bikes) who deliver the food.

Please see a little later for details of what each component of the system will do.

## System Architecture

We are asking you to create this using a very prescribed architecture.

Your system must integrate multiple elements including the University’s Oracle database server, some “middleware” written on C# / .NET hosted on our Xserve and finally the clients. The basic three tier architecture is illustrated in Figure 1.



Mobile client





Desktop Application

DBMS (Oracle)

Middleware (C# and .net on Xserve)



Web Application

Figure 1 : A three-tier architecture

You will note that only the middleware connects directly to the DBMS. The middleware then exposes an API that the other clients can connect to. You will begin to see why in a while.

The core connectivity of your architecture will be evaluated during the scheduled architecture tech demos.

## System Requirements

Think carefully what the system as a whole must do. Here are just a few suggestions.

* Maintain records of customers.
* Maintain records of menus, prices etc.
* Maintain records of delivery riders, who has logged in as being available etc.
* Handle payment (OK we don’t really expect you to take money, but you could set up a Paypal test account to prove the concept).

There will be three types of client and their view of the system will be very different: There will also be different types of user, who will interact with the different applications. These are the types of user:

* **Customers**: Customers will see the web interface as the public interface (although you may also choose to add an additional mobile interface, or what about a half way house with a responsive, mobile friend web interface?). You will have to store customer details, and (just a suggestion) a history of past orders, so that customers can re-order the stuff they likes (or recall past orders of stuff they didn’t like)
* **Chefs**: These are the chefs in the takeway who are notified of orders as they come in (hint: think asynchronous) and who then prepare the food.
* **Administrators/Managers**: can access all the parts of the system necessary to allow maintenance of the data, such as adding new menus, changing prices, doing accounting. They should have access through the administrative desktop application (maybe also an additional feature of the web application).
* **Delivery riders**: will log on when they are available for work, get notified when a delivery job is available, decide to accept the job (or not), deliver the food and record that the delivery has been done.

Please see below for the applications the users will interact with.

## Desktop Application

The desktop application must be implemented using Java and Netbeans. It is the application the chefs in the takeaway who prepare the food will use to see what has been ordered. It may also be used by managers in the takeway to add or change menus, process and such. Also it should be used for accounting and other admin purposes.

## Web Application

Your web application is a public interface to the system providing access for anybody. It should connect web clients to your web server and database. Your web application could be created using either of the following:

* .NET framework using C#, hosted on Xserve, and an extension of your middleware. You may apply to Shirley Atkinson for hosting space on Xserve within the first three weeks of the module.
* Java/JSP using Apache Tomcat or Glassfish server. Either is freely available through NetBeans. You can host that on “Eyeore”. Please apply to Nigel Barlow.

Your client-side can use any appropriate technology eg; HTML/CSS/Javascript.

You will have to put some thought into the client and server at the design stage, in particular you might want to think about what functionality to prioritise and what additional functionality to leave out. If different individuals within the team work on the client and the server, it is important to coordinate these priorities.

Could the staff in the takeaway use the web interface to add or change menus and prices for accounting and admin purposes? You decide. If that is the case, managers and the general public will have very different views of the system.

## Mobile client

You must also create a native mobile client. That means that higher level tools such as Unity3D and PhoneGap are not permitted (though maybe Microsoft Xamarin is; please ask Nigel).

The mobile app is the interface the riders who deliver the food will use. We suggest you use Android for your client, but you may use iOS if you prefer. The mobile app could also provide a public interface to the system, but in this case what the public will want to see will be very different to the deliver riders view of the system.

The idea is that a delivery rider has this basic workflow:

* They log-on, indicating that they are available for work. In doing so their location is sent to “the system”.
* When an order arrives, “the system” identifies the nearest available rider and notifies them that a delivery is needed.
* The rider accepts the job, goes to the takeaway, collects the food, delivers it (and decide what happens if the rider does not want the job).
* On delivery, the recipient does <something> on the mobile device to confirm that the order has been received.
* While this is going on, the mobile device will know exactly where the rider is. Think about the ethics of exactly what you should monitor about them. Certainly I would suggest you turn all location reports and such off when the rider is not logged on as available for delivering.

The mobile app should not attempt to connect directly to the DBMS. It should connect to a web-service hosted by your middleware, and thus the database. That way the risk of getting out of date data from your database is minimized and the really ambitious of you can find ways to have observable entity classes so that the client(s) “just update” as if by magic.

For the purposes of this project, there is no requirement that your client is deployed on a real mobile device; running on an emulated device is all that is required.

And finally, it is common for groups not to be able to fully complete all of the functionality required. You should aim to look initially at all possible aspects that could be implemented and list them, or draw a rich picture to capture the essence of the possible functionality.

Having identified all the possibilities you can use this to create your initial Product Backlog. The current Product Backlog will be used at the start of each sprint to determine what will be included in the next increment (this becomes the Sprint Backlog). Progress should be shown in a Burn-down chart.

Note that it is better to complete a smaller set of functionality well than to attempt too much functionality at a superficial level.

## Asynchronous

Another thing I want you to include is thinking asynchronous. The staff in the restaurant aren’t going to want to constantly push a “refresh” button on their desktop application to see if an order has arrived. Similarly, the deliver riders aren’t going to want to constantly touch a “refresh” button on the mobile app. Both parties will want to be notified. A (not nice) solution is to have a polling thread that, say, once a minute polls the middleware to ask if anything new is going on. You will be shown stuff like this (and you have already seen threads in C# in year 1) if you haven’t already seen it in the mobile device lectures.

Incorporating asynchronous code into the public web interface is probably out of scope here as that involves delving into technologies such as Web Sockets and Ajax.

## Security

Finally, in order to attain the highest grade for this project, you will have to implement some security features. As this is not an area that is necessarily familiar to you, you will have to research different functionality, solutions and technologies. It is up to you to identify appropriate contributions to your project in this area. You will get a little added security content about common “gotchas” such as why using a hashed password to authenticate over a network is a very bad idea.

# Project Management

## Introduction

For this module you are expected to apply agile project management techniques to how you run the project. This will require that you develop a product backlog, estimate how long things will take and work in sprints.

During the first week of the module you will be introduced to the agile practices that we need you to develop and practice.

You must use Pivotal Tracker or some other agreed project management tool. You must invite Shirley Atkinson and allow her access to the tool. It is expected that you will use this weekly (see 4.2 below).

## Progress meetings

You should get into the habit of holding a progress meeting every week to evaluate the performance of your group. This is where you can evaluate your progress to date, discuss the features you are each working on, discuss any barriers in the way and then update your project management tool.

Shirley Atkinson will be reviewing your project management tool on a weekly basis and will expect to see some activity. If there is no activity, you are likely to get an email asking for information. Do not ignore this email.

## Final Report

The final report must contain a section on project management. Here you draw together the details of the work carried out and you analyse your team project management. Simply producing a glowing report of how well the group worked together is not sufficient; it is much better to recognize and evaluate any difficulties that have arisen and to demonstrate that you can reflect critically on your own work. It is an unusual project that does not encounter any problems.

Prior to writing the final report you should have a project closure meeting in which you review the project and the lessons learnt. Use the details from this meeting in the project management section of your final report.

The final report should also compare the plans you made at the start of the project and what had to change as you went along. In particular you should review your risk analysis and in the light of your experiences, comment on the appropriateness of your own evaluation. For example, did you identify appropriate risks, were your contingency plans appropriate?

# Requirements Analysis and Design

## Introduction

There will be a co-taught lecture introducing the main concepts around requirements analysis and design. For further information and support you should contact Ismini Vasileiou.

## Final Report

The final report section on “Requirements Analysis” should contain the following elements:

* A list of functional and non-functional requirements.
* Analysis and design
  + A use case model (use case diagram(s) with some basic descriptions of use cases) for the final version of the software you have produced.
  + User stories with acceptance tests consistent with the use cases identified in the use case model.
  + User requirements list (product backlog)
  + Select FOUR use cases which will be demonstrated and produce design level sequence diagram, state machines and class diagrams.
  + Evaluation of how your design changed during implementation.

The analysis and design will be worth 10% of the final grade and the evaluation 5%.

# Database Design

## Final Implementation

The database is to be implemented in Oracle and is to be left on the Oracle server. You will also need to populate the database with sufficient and representative data to allow for testing and demonstration of the finished system. Naturally, you will be expected to implement an appropriate level of data integrity.

## Final report

Your final report submissions must include a database diagram generated by Oracle SQL Developer, appropriate descriptions of your database objects (tables, views, sequences, database triggers, stored programs (procedures, functions)) and your SQL CREATE statements which should adhere to the naming and presentation conventions introduced in ISAD251.

Submit your normalized tables and compare your Schema with your Class Diagram. How does the final schema look like?

Do not forget what was covered in ISAD251!

# Human-Computer Interaction

## Introduction

Throughout the project there are a number of activities where you will receive feedback on your HCI approach. These should be incorporated into your final report and reflected upon.

## Cognitive Walkthrough

Details of the cognitive walkthrough are covered in a one-off lecture session early in the term. You will be provided with details of how to prepare for your cognitive walkthrough in that session. However, for completeness in this document, these are the questions which your expert reviewer will be asking in your cognitive walkthrough:

* Preparation: Has a substantial amount of work already been completed so as to enable a useful cognitive walkthrough?
* Preparation: Were three significant tasks prepared for the cognitive walkthrough as defined in the accompanying lecture session?
* Data: Was data captured appropriately during the cognitive walkthrough (audio/hard copy data recording by multiple tem members)
* Professional engagement: Did the group engage professionally and constructively with the feedback they received (not arguing all the time but accepting constructive feedback?)
* Overall: How well did the group manage their cognitive walkthrough?

As you can see, preparation is very important aspect of a successful cognitive walkthrough.

## Usability evaluation

Your team should carry out usability evaluations during each of the iterations of your product development. The number of “subjects” (people using your product) evaluated during each iteration should be determined by you/your team. **In addition to the usability evaluation you do yourselves**, a live usability evaluation session has been scheduled for you. We shall be enlisting first years to come to that session to review the products for you.

As this may be the first time you have run usability tests, we also provide you with a mock usability evaluation a few weeks before the live usability evaluation. During the mock session two or three members (you will be advised when you register on the day for the session) of your team will be in charge of the usability evaluation of your product. The remainder of your team will be required to take on the role of subjects for another team. Therefore, the attendance of all team members is **mandatory**.

Each group should be prepared (with sufficient paperwork and team members prepared) to run between 9 and 15 (or more if you like) individual usability evaluations during **both the mock and the live** usability evaluation session.

## Final report

Your performance during the usability study itself will not be explicitly marked, but you are required to write a report on the user feedback obtained during the usability study and include this in your final report. The requirements for this section are as follows:

* A progressive list (deriving from all user evaluations) of prioritized essential and desirable changes required. This list should be the result of synthesizing the data gathered during the usability evaluations of your product. In addition, you will also know of weaknesses in your product.
* Each list must be prioritized in order of importance. The priority of items will also be assessed.
* The HCI element will be marked on clarity. Note that diagrams may be added at the end of the document – these must be numbered Figure 1, 2 etc and referred to as such as in the main text.

The usability report must be a thorough critical evaluation of the usability of your product within the scope of this project. Please do not comment on features outside of this scope. Your report should also specify the strengths and weaknesses of your product and it should also report on user quotations, user preferences and the ease with which users were able to achieve their goals in task scenarios.

## Usability of the final product

The usability of your product will be assessed by members of the staff team during the final group presentation. You should aim to write software with an interface that supports user interaction in a functional but also easy to use, intuitive and also aesthetically pleasing manner.

We strongly advise that the most successful way to score highly in this area is to gather and act upon, as much user feedback on your product as often as you can particularly at the key stages of the software development process.

# Security

## Introduction

A range of technologies exist for making networked applications and communication less exposed to hackers and more reliable. You, as developers should utilize the “OWASP Top Ten” list to guide your secure coding efforts. The OWASP Top Ten details the most common web application security vulnerabilities, including basic methods to project against these vulnerabilities.

## Final Implementation

The degree to which your final system includes solutions to ensure reliability and security will be judged by your submitted code and functionality demonstrated. You should take into consideration and implement appropriate measures for as many of the OWASP Top Ten as you consider appropriate. For example these aspects ought to be considered:

* Protecting against SQL injection attacks.
* Protect against broken Authentication:
  + URL contains session id which can be used by someone else
  + Passwords not encrypted in either storage or transit
* Sensitive data protection: Sensitive data should be encrypted at all times, including in transit and rest. No exceptions. Credit card information and user passwords should never travel or be stored unencrypted. Passwords should always be hashed or salted.

The OWASP Top 10 checklist is a good one to take into consideration and can be found here <http://software-security.sans.org/resources/swat>. There are elements in this checklist that are for theoretical purposes only.

## Final Report

Your final report should thoroughly explain what appropriate measurements you were able to implement and why. You should discuss how they improve the quality of the system. You should also discuss those aspects you feel are appropriate to the system but you were unable to implement because of time. Extra marks will be given if you pentest your application with some automated tools and are able to illustrate that it is robust.

# Software Engineering

## Introduction

The software engineering element of your project will be marked based on your architecture demos (mentioned in section 2.2 above), your code and your report.

## Final Report

In the group report you are required to submit the following:

* A UML class diagram for the application as outlined in section 5.2 on requirements. Your software engineering section must therefore provide an evaluation detailing any shortcomings, good points, and assumptions made.

The quality of your design will be evaluated against common OO quality guidelines for object-oriented systems such as Craig Larman’s GRASP (General Responsibility Assignment Software Patterns) guidelines and the use of established software patterns.

## Implementation

Your final software suite will be marked according to how well it supports the requirements given in the Scenario in section 3 and the HCI requirements identified during the project. The code should adhere to the standard coding conventions for the language.

The marking will also include your Bitbucket commit log which should demonstrate regular commits from multiple project members. It is crucial that the Bitbucket log is available until after the examination panels have sat so that any discussion about achievements can be based on evidence.

# Peer Assessment

## Introduction

The peer assessment aspect provides the wherewithal to adjust marks according to personal input and effort. It is a mechanism to address some of the failures in contribution that may happen within a group work environment.

You will receive an email towards the end of the module informing you that the OPAS system is open for the input of your peer marks. Failure to input your marks will result in you receiving a penalty of -10% on the group mark.

The marks that you give will be taken into account when calculating your individual marks. This is outlined in section 2.3 marking criteria. However, the staff team on the module will have the final say in any arbitration arising out of any dispute.

Group members may agree that they wish all members of the group to be treated equally. In this case, all should agree to enter the same marks so that the group marks are shared equally.

When submitting peer assessment, use the free-text area where you can place written feedback that you would like the module team to see. You MUST given some constructive written feedback for each team member. Do use details as vague statements are not helpful – eg: member x failed to engage. How did they fail to engage? Eg: Member x failed to engage. They did not attend n meetings, they did not email, they did not respond to questions and did not deliver code.

Your assessment must be based on contribution and performance, NOT on personality. Eg: member y is an excellent worker, really nice and friendly.

## Criteria

The criteria for the peer assessment are as follows.

You grade your fellow team members on each of the assessment criteria in the range 0 to 5 as follows:

* 0: did not contribute
* 1: token contribution
* 2: willing but not successful
* 3: average
* 4: above average
* 5: outstanding

The assessment criteria are as follows:

* Overall level of participation.
* Suggestion of ideas/innovation.
* Performance in routine tasks.
* Ability and willingness in sorting out problems.
* General attitude to the tasks.
* Participation in assigned responsibilities.