Coursework Specification

M2M Connect; SMS => PHP Processing

This coursework is worth 50% of the module's final mark. It is to be completed in teams of three people and it is assumed that all group members will get equal marks (unless it becomes obvious that one member of the group has contributed significantly more or less effort).

This coursework specification is designed to enable you to achieve the following module learning objectives:

1. Can design and implement a web application that is impervious to the most common web-based attacks
2. Can design and implement a web application that accesses and displays data from remote web services

Team Work

The application is to be implemented in teams of three. Developing an application as a member of a team is important to experience because the majority of web developers would normally work in groups, whether they are employed by an organisation that utilises Agile Methodologies or not. Freelance contractors in PHP (and other languages) will often collaborate as a typical web application will utilise multiple technologies; too many for one person to manage sensibly.

Framework

The SLIM framework is probably the most widely used micro-framework currently used to develop web applications in PHP. As a web developer, you would normally be expected to know at least one such framework, and SLIM is a reasonable starting point.

Objective

The EE SMS server accepts SMS/GPRS messages and stores them in XML format. The EE M2M Connect web-service makes these stored messages available to be downloaded via the EE SOAP server. Once a message has been downloaded by a SOAP client it can be parsed and the embedded data extracted, sanitised, validated and stored in a local database. After the data has been downloaded and stored, then web-page reports can be prepared on demand by a user.

Once an initial web-report has been created and displayed on a user's browser, AJAX or JSON could be used to dynamically update the reporting web-page when subsequent messages are downloaded and processed. Emails or SMS messages could notify users of the arrival of a message; numerical data such as temperatures could be displayed as a chart, SMS messages could be used to update the status of the circuit board (this will be a simulation).

The server that downloads and processes the SMS messages (and subsequently makes the data available to clients) will be implemented in Object Oriented PHP with MySQL as the database server. You also have to publish your finished application on the public facing PHP and MySQL servers. Be careful to remove any code you have written for debugging purposes before uploading your application.

NB a Rich Internet Application (RIA) is not required – the majority of all processing will be achieved via PHP on the server.

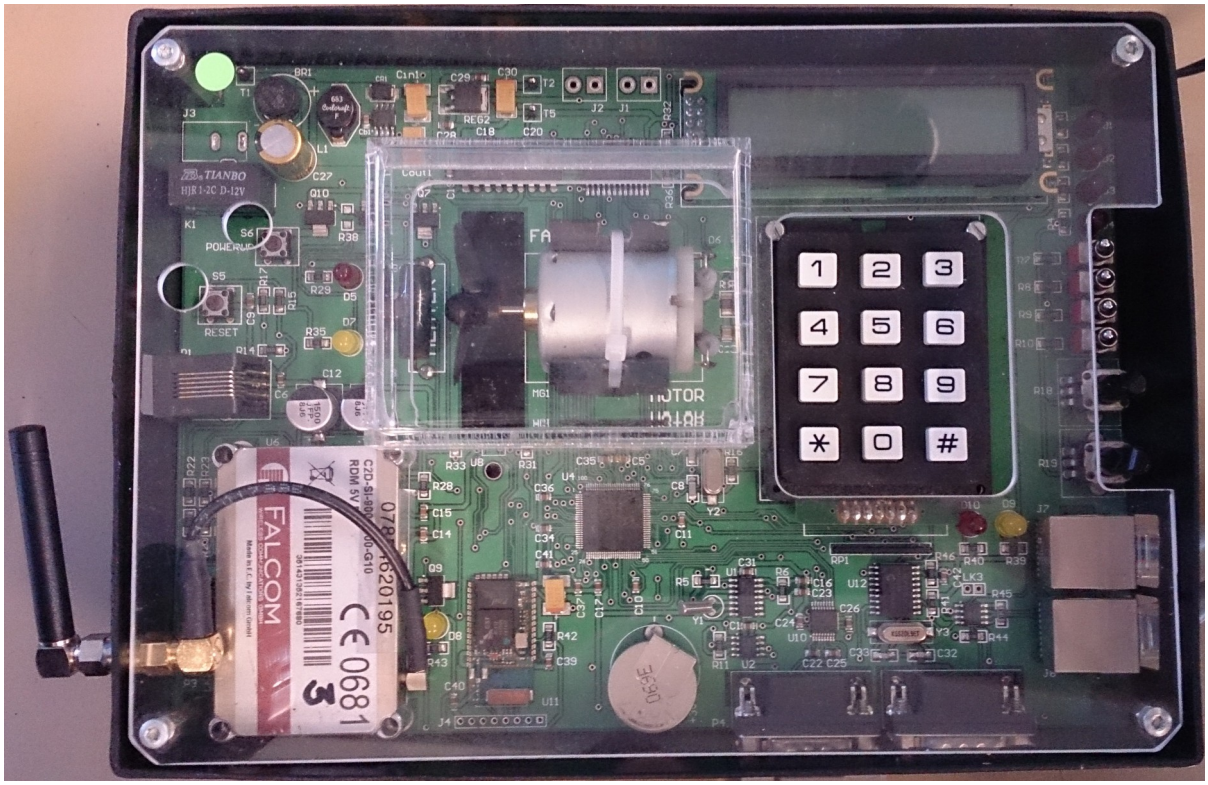
Coursework Specification

Generating the SMS Messages

The SMS messages you will process will be derived as if from a circuit board such as those used in the Telematics laboratory (see the photograph below). If you are taking the Telematics module, then this can be real data, otherwise an SMS client can be used to create SMS messages containing simulated data.

The circuit boards have four switches (see right-hand side of the photograph), a heater, a fan and a keypad. The current state of each of these components (switches - on/off, fan - forward/reverse, heater - temperature, keypad - last number entered) should be embedded within a message string and the message transmitted via SMS from the GPRS modem.

The format of the message string is for you to design. It could be XML, plain text or a compacted bit string. You will also need a way of identifying messages as your own as the EE SMS account is shared.



Coursework Specification

Implementation

A complete implementation will include the following methodologies and technologies that have been discussed this academic year:

- Model View Controller (MVC) Architecture & Single Point of Access
- Object Oriented PHP
  - Showing the usage of inheritance and dependency injection
  - You should use the SLIM micro-framework to implement the application
- Application of security techniques and avoidance of common web application vulnerabilities
  - Security will be achieved by correct file structure, proper sanitisation and validation techniques, etc
- Unit testing and security testing
- Validated HTML (using simple CSS for web-page layout/presentation)
- MySQL
- SOAP & WSDL file
- Docblock comments
- Consistent coding style (see the style guidelines made available earlier)
- Use of the Subversion Version Control Server

The minimum application implementation to achieve a passing mark of 40% is to download, validate, parse, store and display relevant SMS messages.

Possible extensions to the implementation could include

- implementation of registration and login/logout features, incorporating session management.
- displaying numerical data in chart form.
- an interface to send SMS messages back to the “circuit board” containing updated settings for the board.
- administration interface to maintain users/connection data in the database.
- logging all web-application activity in a database table.
- using AJAX and JSON (or an RSS/ATOM feed) to update the display in the browser as new SMS reports are downloaded.
- etc…

Suggested Team Roles

I suggest you allocate roles to your team members along these lines:

Team Member	Roles
A	Team leader, Software architect; Tester
B	Web Application Developer
C	Documenter/author; Business analyst/designer

It is strongly suggested that you agree a time plan (such as a Gantt chart).

You must use the Subversion server to coordinate your design and development efforts.

Coursework Specification

Submission

This is a piece of group coursework, to be completed in the teams to which you have been assigned.

This assessment includes a **must attend and pass** viva. The viva will give you the chance to demonstrate that the submitted work is your groups, and to give the you the opportunity to defend the direction, structure and conclusions of the work. Failure to meet these aims in the viva may result in your group members failing the viva and the assessment overall.

Non-attendance at the viva will mean you receive a zero mark.

MANDATORY

You have to:

- Use the SVN server to monitor the development of your application. Make sure you do regular commits as you develop your solution.
- Publish your finished application on the coursework servers: *php.tech.dmu.ac.uk* and *mysql.tech.dmu.ac.uk*
- Submit a CD containing
  - all PHP files
  - an SQL file to re-create your MySQL database, including the user accounts.
  - a readme file giving a brief explanation as to how to access and use your web-site.
  - Xdebug Trace and Profile files
- Produce and submit written documentation containing
  - Analysis & design diagrams (including as a minimum ERD, Use Case and Interactive Sequence) that show how your implementation complies with the MVC architecture
  - Specification of your solution
  - Implementation, including a description of how your code fulfills the specification
  - Test plans and logs
  - Summary & conclusions
  - References & bibliography
  - Document specifying which member of the group did what work. This must be signed by all members of the group.
- Attend a viva as a group at your allocated time. Timetables will be published in November 2016.

You should submit your coursework:

- at the Faculty of Technology Advice Centre (FoTAC)
- by 4.00 pm
- on or before the 12th of January 2017.

The attached marking scheme is to give you an idea as to where marks will be allocated. Truly brilliant submissions may be awarded marks above the stated maxima.

Coursework Specification

Vivas will take place during the two weeks commencing 16th January 2016. Your unmoderated mark is given to you at the viva.

Required Steps

Your web application should, as a minimum requirement to pass:

- Use a PHP SOAP client to download SMS messages from the M2M Connect server.
- Parse all downloaded messages.
- Store the downloaded message (content and metadata) in the database.
- Display the message content and metadata on the web-browser
  - Message metadata:
    - eg source SIM number, name, email address
  - Message content:
    - ie state of switches on the board, temperature, key-pad value, etc

Extension functionality.

- Check message metadata (eg name/phone number) against pre-stored values (preferably stored in a database).
- Send emails reporting the outcome to the user's stored email address (you could also send an SMS).
- Allow users to register with the web site.
- Require users to login and logout of the web site.
- Display numerical data in chart form.
- Implement an interface that will allow a web-application administrator to send updated circuit board settings via SMS to the source mobile phone modem.
- Implement an interface that will allow a web-application administrator to maintain users & phone numbers in the database.
- Log all web-site activity to a file or the database.
- Use AJAX, JSON or an RSS feed to update the display as new SMS reports are downloaded.

Notes

- Implement your solution with the SLIM micro-framework.
- PHP code must be security aware.
- All client HTML output will conform to the ***HTML*** DTD.
- Validate all web-pages at http://validator.w3.org/ - include the W3C validated logo on your web-pages when validated.
- Control the layout of the web-pages with a stylesheet.
- Include diagrams that clearly show how the files comprising your implementation conform to the MVC architecture.
- PHP scripts will be sampled for assessment of code styling.