Marking scheme:

Mandatory Deliverables (published, trace & profile files, SQL script):

1. Application successfully published and executes on the PHP & MySQL servers
2. SQL script to create database and users; readme.txt file
3. Xdebug trace and profile files produced
4. All requirements fully complied with; complete documentation as specified

Architecture (SLIM/Twig, MVC, Single point of entry):

1. Application successfully uses the SLIM/Twig framework
2. Single point of entry architecture has been implemented successfully in SLIM/Twig
3. Secure file structure has been implemented successfully with the current SLIM/Twig framework
4. Completely conforms to a secure architecture, based upon SLIM/Twig

Code (completeness of application, coding style, documentation):

1. SMS messages downloaded, parsed, stored and viewed; OO PHP
2. Email/SMS sent when new SMS received; Consistent coding style ; most classes have page level docblocks
3. All classes & methods are commented with docblocks
4. Professional standard code, fully styled, fully documented scripts

Teamwork (SVN):

1. SVN used to manage collaboration
2. Clearly defined roles within the team; SVN used consistently
3. Mature team attitude, underpinned by comprehensive usage of SVN
4. Professional attitude taken to working as a team

Testing & Security (unit testing, vulnerability considerations validation and sanitisation):

1. Attempt to filter user-entered data
2. An attempt has been made to create unit tests for some classes; entered data is filtered
3. Unit testing used in development; all data is filtered, relevant error messages generated for the user
4. Fully tested for both code integrity & vulnerabilities; all user entered data fully validated and sanitised

Page layout & compliance (UX, validated HTML & CSS):

1. Web pages comply with the HTML 5 DTD; styles are applied by a CSS
2. User interface is clear and easy to use
3. User experience is positive
4. Professional UX; completely validated and optimised code

Possible Extensions (user registration, authentication, JSON, charts, activity & error logging, etc):

1. -----
2. One or more extensions implemented
3. Multiple extensions implemented
4. Multiple extensions implemented, to a professional standard

Objective:

1. Download, validate, store and display telemetry data
2. Data will be formatted and sent to EE M2M server
3. Use:
   1. PHP language
   2. Slim for framework
   3. Twig for output
   4. Doctrine for database access
   5. Monolog for logging

About EE M2M server:

1. Accepts SMS/GPRS (General packet radio service) messages -> stores them in XML format
   1. GPRS is a packet oriented mobile data standard on 2G and 3G networks
2. Stored messages can be downloaded by the SOAP server
3. Downloaded->Parsed->Data extracted->Sanitised->Validated->Stored in database->Web page report prepared on demand for user
4. Ajax request used to dynamically update the report
5. Email or SMS message notify user of new message
6. Numerical data such as temperature displayed as charts
7. SMS messages used to update status of circuit board
8. We are to create this server processing and outputting to the user in Object Oriented PHP with MariaDB

Components of the circuit board:

1. Four switches (on/off)
2. Fan (forward/reverse)
3. Heater (temperature)
4. Keypad (last number entered)

The current state of each component to be embedded withing a message string and transmitted via SMS to your account

- We are to decide the format of these messages (plaintext, XML, compacted bit string)

- These messages need to be uniquely identified by our team, as EE SMS account is shared

Must be implemented:

1. PHP SOAP client to download SMS messages from the M2M Connect server
2. Parse all downloaded messages
3. Validate all content
4. Store the downloaded messages
   1. Content: State of switches, temperature, key-pad value, etc
   2. Metadata: SIM number, name, email address

Should be implemented:

1. Separation of Concerns Architecture & Single Point of Access
2. Object Oriented PHP (SOLID guidelines)
3. SLIM framework
4. TWIG template
5. Monolog logging
6. Security techniques and avoidance of common web app vulnerabilities
7. Unit testing and secure testing
8. Validated HTML 5
9. MySQL/MariaDB
10. SAOP & WSDL file
11. Docblock comments
12. Consistent coding style following PHP-FIG PSR-1 & PSR-12 standards
13. Use of Subversion Version Control Server (SVC server)
14. Validation of all web-pages at http://validator.w3.org/ - include the W3C validated logo on your web-pages when validated.

Possible extensions:

1. Registration and login, with session management
2. Numerical data in chart form
3. Interface to send SMS back to circuit board, to update settings for board
4. Admin interface to maintain users/connection data in database
5. Logging all web-app activity
6. Using AJAX and JSON (or an RSS/ATOM feed) to update the display in the browser as new SMS reports are downloaded
7. Checking message metadata against pre-stored values, avoiding duplicate messages
8. Sending an SMS or email reporting the receipt of new message to the user’s connection detail (alert on new message)

Team roles:

* Team leader
* Software architect
* Web app dev
* Database admin
* Tester (eg usability testing with Selenium, unit testing with PHP Unit)
* Business analyst
* Interface designer
* Documenter (DocBlocks)
* Coding standard compliance (PSR-1 & 12)

**COMMIT ONCE A WEEK MINIMUM** with relevant comments

Summary of personal contribution to your team’s application need to be uploaded after submission date, before Viva.

Upload and configure your application to the coursework servers: php.tech.dmu.ac.uk and mysql.tech.dmu.ac.uk

NB you will need to use a campus machine to upload to these servers.

Either by being physically in a laboratory o using the VPN

using the VMware Horizon Remote Desktop application

Submit your coursework via your teamwork SVN account. Include the following:

♣ all PHP scripts

♣ an SQL file to re-create your MySQL database, including the user accounts and dummy data.

♣ a readme.txt/readme.md file giving a brief explanation as to how to access and use your web-application.

♣ Examples of Xdebug Trace and Profile files NB not all xdebug files – just a few of each.

♣ Examples of PhpUnit scripts (if implemented)

♣ A Personal Statement detailing your personal contribution to the application must be uploaded to your personal SVN account.