**Chapter 3**

**Methodology**

Since the main idea behind this project was to allow customers to top up their smart meters via electronic means, the project was built as an electronic recharge platform which integrated various application programming interfaces (API) to effect payments. The APIs used for the integration were iPay and Stripe which provide support for mobile commerce and e-commerce transactions. The supported payment platforms include:

1. Mobile commerce systems
   1. MTN Mobile Money
   2. Vodafone Cash
   3. TiGO Cash
   4. Airtel Money
2. Debit card platforms
   1. VISA
   2. MasterCard

The Enersmart Recharge System was designed with both mobile and non-mobile users in mind. Therefore, the system features both a web platform and a native android interface. The following software were used in programming the android interface:

1. Android Studio (with Java implementation)
2. SQL
3. Android Software Development Kit

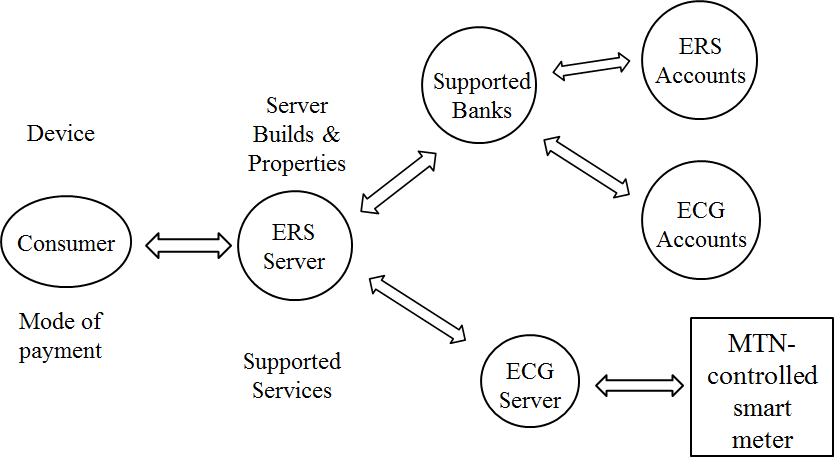
The software for programming the web application were:

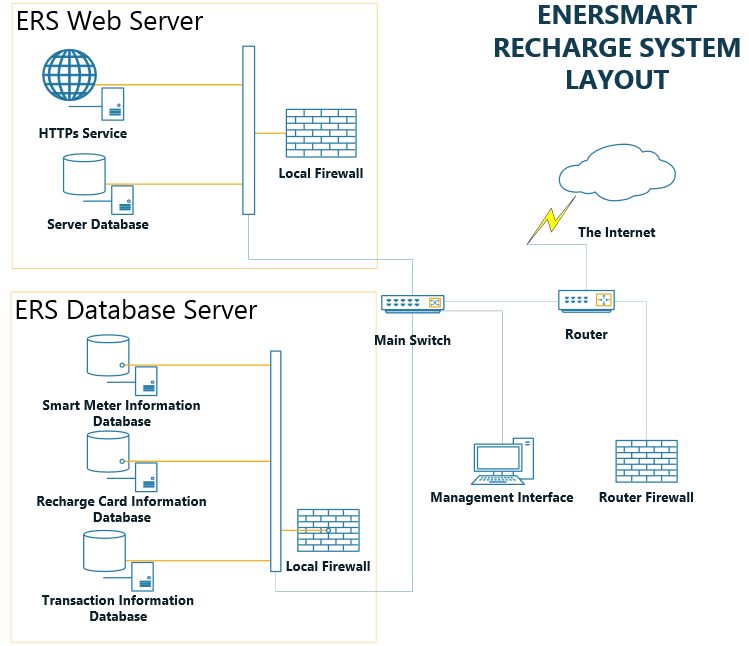
1. PHP5
2. SQL
3. HTML5 & CSS3
4. JQuery & JavaScript

These two interfaces were built to communicate with the Enersmart Recharge System Web Server and Databases. These were implemented using the following tools:

1. Apache2 (to run the web service)
2. SQL (to program the required databases)
3. IPtables (for firewall support)

Both the web server and database were hosted on an Ubuntu 14.04 LTS operating system for speed and stability and increased security.

The diagram below shows how the Enersmart Recharge System integrates with external systems

**System Design**

The figure above gives an illustration of what the components of the Enersmart Recharge System look like.

Both the web and android platforms possessed similar menus. This was for the ease of simplicity and user familiarity. Both interfaces display the following menus:

1. Welcome Screen
2. Registration
3. Login Screen
   1. Recharge Menu
      1. Recharge Method
   2. Monitoring Console
   3. Transaction History

The system was initially designed to run on a high performance computer with the following minimum specifications:

1. 6GHz CPU processing power
2. 4GB RAM
3. 100Mbps RJ-45 Ethernet LAN port
4. 64GB Solid State Drive

The system may be run on a computer having lower specifications; however, its performance will be affected considerably.

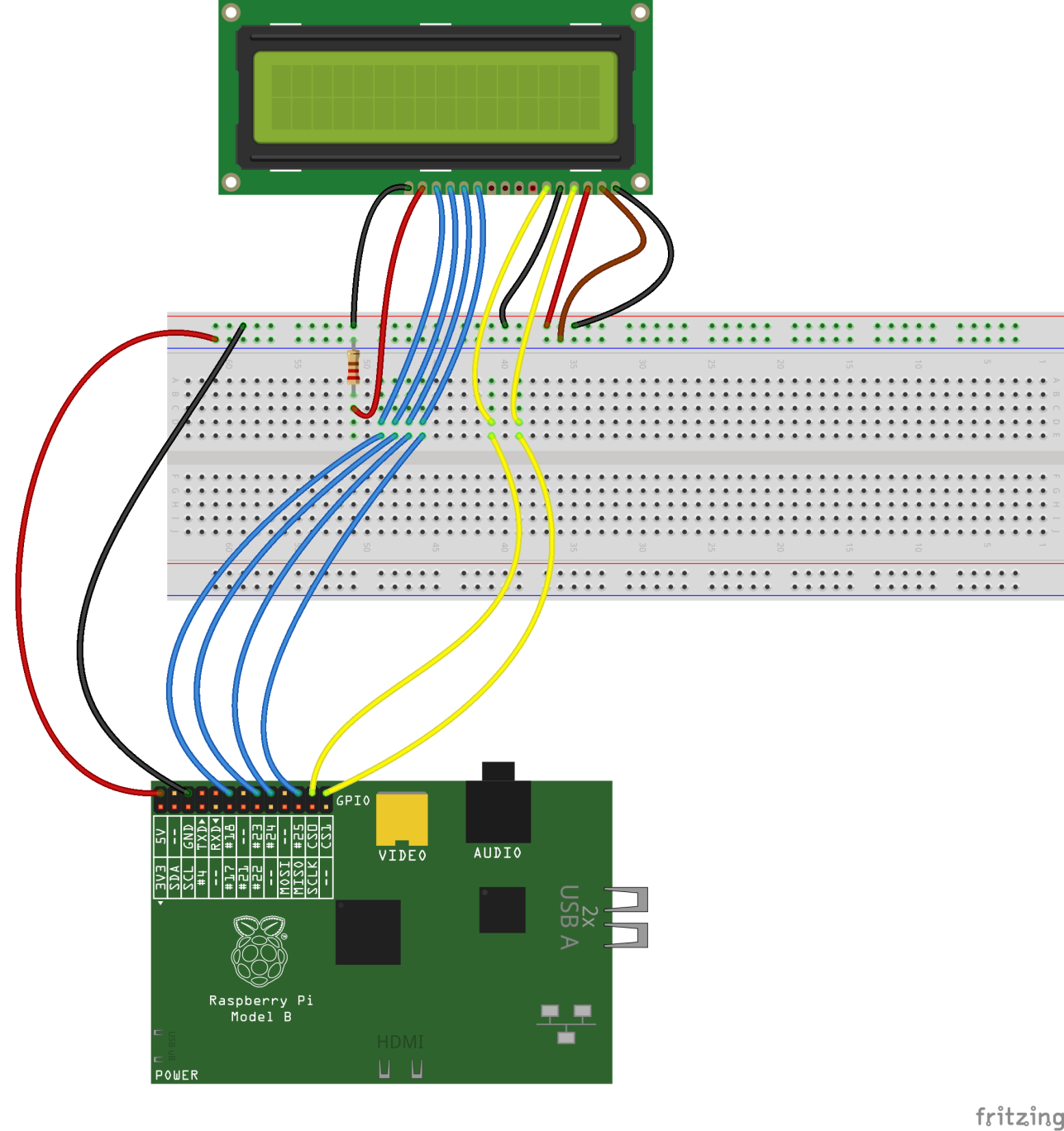
**Experimental Work**

Although the criteria for running the system effectively have been clearly stated, due to financial constraints, a prototype of the server was built on a low-performance laptop with the following specifications:

1. 1.2GHz dual core CPU
2. 3GB of RAM
3. 120GB of hard drive storage
4. 100Mbps RJ-45 Ethernet LAN

Also due to the unavailability of an actual smart meter, a microcontroller was programmed to emulate the functions of one.

The diagram below shows the breadboard view of the pseudo smart meter:

****

The associated source code for programming the RaspberryPi is given below:

The first scenario was to test to verify whether electronic payments via the Enersmart Recharge System were possible using debit card and/or mobile money.

The second scenario was testing to verify that the monitoring and control functions worked as stated in the objectives.

Finally, penetration tests and vulnerability assessments were performed by various third-party experts to assess the security of various aspects of the system in conformity to approved worldwide standards to ensure that the entire security of the Enersmart Recharge System met these standards.

Seven different standards were used in assessing the security of the system. They are listed as shown below:

1. Penetration Testing Execution Standard (PTES)
2. Open Web Application Security Project (OWASP)
3. Open Source Security Testing Methodology Manual (OSSTMM)
4. Information Systems Security Assessment Framework (ISSAF)
5. National Institute of Standards and Technology: Information Security Testing and Assessment (NIST SP800-115)

The security audit was performed according to the following summarized procedure:

The components of the system whose security was assessed are the encryption of the keys used in the SQL database and the encryption of SSL and HTTPS connections from the web server. Again, data (usernames, passwords, and credit & debit card information) present on, and transmitted from, the web application, local MySQL database and Android application were also tested for reliable encryption.

Software used for the auditing include:

1. Zenmap
2. Subterfuge
3. dSploit
4. SQLmap
5. Mozilla Firefox Web Developer Tools

**Results and Discussion**

The table below shows the results of the security auditing that was performed on the Enersmart Recharge System:

|  |  |  |  |
| --- | --- | --- | --- |
| **VULNERABILITY NAME** | **NUMBER OF TESTS RUN** | **TEST RUNTIME** | **VULNERABLE(N)** |
| SSL-HEARTBLEED | 3 | 45 MINUTES X 10 DAYS | X |
| SSL-KNOWN-KEY | 3 | 45 MINUTES X 10 DAYS | X |
| SSL-POODLE | 3 | 45 MINUTES X 10 DAYS | X |
| SSLv2 | 3 | 45 MINUTES X 10 DAYS | X |
| HTTP-APACHE-NEGOTIATION | 3 | 45 MINUTES X 10 DAYS | X |
| HTTP-BACKUP-FINDER | 3 | 45 MINUTES X 10 DAYS | X |
| SSL-CCS-INJECTION | 3 | 45 MINUTES X 10 DAYS | X |
| SSL-CERT | 3 | 45 MINUTES X 10 DAYS | X |
| SSL-DATE  (TLS RANDOMNESS) | 3 | 45 MINUTES X 10 DAYS | X |
| SSL-DH-PARAMS | 3 | 45 MINUTES X 10 DAYS | X |
| SSL-ENUM-CIPHERS  (TLS v1.2) | 3 | 45 MINUTES X 10 DAYS | X |

From the results indicated above, the Enersmart Recharge System was not susceptible to SSL or HTTPs attacks. Also,

**Chapter 4**

**Conclusion**