# Table of Contents

[Table of Contents 1](#_Toc422747221)

[1. Intro 3](#_Toc422747222)

[1.1. Concept 3](#_Toc422747223)

[1.2. Objective 3](#_Toc422747224)

[1.3. Plan 3](#_Toc422747225)

[1.4. Background 4](#_Toc422747226)

[1.5. Problem Definition 4](#_Toc422747227)

[1.6. Suggested Solution 4](#_Toc422747228)

[2. Feasibility Study 5](#_Toc422747229)

[2.1. Quora 5](#_Toc422747230)

[2.2. Study 8](#_Toc422747231)

[3. Analysis 10](#_Toc422747232)

[3.1. Hardware: Raspberry Pi 10](#_Toc422747233)

[3.1.1 Raspberry Pi Specifications 10](#_Toc422747234)

[3.2. Existing Products 11](#_Toc422747235)

[3.2.1. Xively 11](#_Toc422747236)

[3.2.2. Particle 11](#_Toc422747237)

[4. Requirements 12](#_Toc422747238)

[4.1. Introduction 12](#_Toc422747239)

[4.1.1. Product Overview 12](#_Toc422747240)

[4.2. Specific Requirements 12](#_Toc422747241)

[4.2.1. External Interface Requirements 12](#_Toc422747242)

[4.2.2. Software product features 13](#_Toc422747243)

[4.2.3. Software System Attributes 15](#_Toc422747244)

[4.2.4. Database Requirements 16](#_Toc422747245)

[5. Design 17](#_Toc422747246)

[5.1. Architectural Design 17](#_Toc422747247)

[5.1.1. Introduction 17](#_Toc422747248)

[5.1.2. Architectural Representation 17](#_Toc422747249)

[5.1.3. Architectural Goals and Constraints 17](#_Toc422747250)

[5.1.4. Use-Case View 18](#_Toc422747251)

[5.1.5. Logical View 19](#_Toc422747252)

[5.1.6. Process View 20](#_Toc422747253)

[5.1.7. Module Decomposition View 21](#_Toc422747254)

[5.1.8. Data View 22](#_Toc422747255)

[5.2. Conceptual Design 22](#_Toc422747256)

[5.2.1. Experiential Flowchart 22](#_Toc422747257)

[5.2.2. Interface Mockups 22](#_Toc422747258)

[5.2.3. Implementation Plan 23](#_Toc422747259)

[6. Implementation 24](#_Toc422747260)

[7. Testing 25](#_Toc422747261)

# Intro

## Concept

Internet of things is a new concept that the industry is moving towards. It’s a result of the maturity of embedded, mobile and handheld devices. Embedded devices now have developed a lot, its development became much easier and the range of applications became very wide. Its capabilities have grown, now you can connect an embedded board to internet easily so embedded devices became “ONLINE”. On the other hand, cloud devices have developed a lot there is now a lot of tools that can handle storing and communicating data also data science developed so now data became the oil every one searching for because processing data using artificial intelligence, NLP, Data science, … etc. have reached a place where you can extract useful information that can be a milestone to building more useful products, so now comes the missing link connecting embedded devices to the internet so we can have full use out of it INTERNET OF THINGS.

## Objective

Our goal is to build a platform for developers seeking to build apps using the internet of things concept, so that it will become easier. Imagine a developer who wants to build some sensors that will acquire some data then show it to the user he needs to work with some low level and embedded code on the board he is using then he needs to start building some form of backend that will receive the data and store it and he needs to do it in an efficient and secure way and then he needs to start building a web or mobile interface the user can interact with and all of that requires different knowledge and expertise to do so our goal is to build a platform that will abstract a lot for him and let him do the minimum amount of work .

## Plan

First, there's going to be a web IDE that developers can write server side code that they want to run when data is sent and we are searching into being able also to flash code remotely although this would need special hardware to handle and writing libraries and language wrappers to be able to communicate with the device.

Second, we are going to build a REST API to handle receiving and providing data so that the developers can use it to access information related to users from our servers.

Third, a library to be used on the embedded device itself to ensure a connection to the server with a secure and efficient way with as little footprint as possible with minimum resources consumption.

## Background

Internet of things is a new concept that the industry moving towards it’s a result of the maturity of embedded, mobile and handheld devices. embedded devices now have developed a lot its development became much easier and the range of applications became very wide its capabilities have grown now you can connect an embedded board to internet easily so embedded devices became “Online” on the other hand cloud devices have developed a lot there is now a lot of tools that can handle storing and communicating data also data science developed so now data became the oil every one searching for because processing data using artificial intelligence, NLP, Data science, … etc. have reached a place where you can extract useful information that can be a milestone to building more useful products, so now comes the missing link connecting embedded devices to the internet so we can have full use out of it.

## Problem Definition

The problem mainly is there is not enough Internet of Things application in the market right now , this is mainly due to three reasons first the low resources of embedded devices that doesn't allow sophisticated apps running on embedded devices , Second the lack of Internet of things developers this is because IOT Requires knowledge and experience for different domains from embedded and low level knowledge to different routing protocols to server side programming , the Third reason is the difficulty of handling big number of connections and securing this connection.

## Suggested Solution

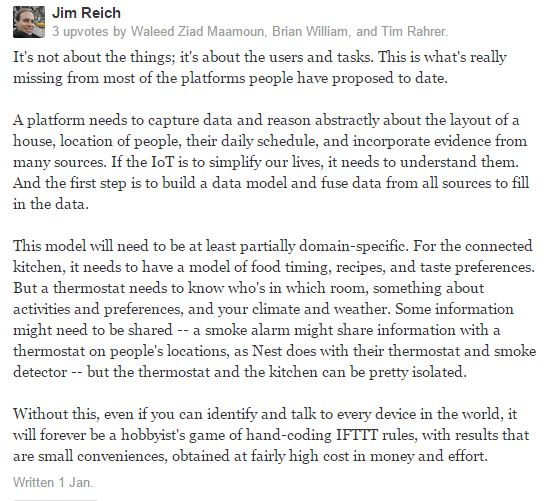
Our goal is to build a platform for developers seeking to build apps using the internet of things concept, so that it will become easier. Imagine a developer who wants to build some sensors that will acquire some data then show it to the user he needs to work with some low level and embedded code on the board he is using then he needs to start building some form of backend that will receive the data and store it and he needs to do it in an efficient and secure way and then he needs to start building a web or mobile interface the user can interact with and all of that requires different knowledge and expertise to do so our goal is to build a platform that will abstract a lot for him and let him focus on the functionality of his project without worrying about scalability , Security and reliability.

# Feasibility Study

## Quora

Quora is a question-and-answer website where questions are asked, answered, edited and organized by its community of users. We went on the website and asked a panel of experts about our project and what they would like to see in it that doesn’t exist in any other known platforms as shown in the figure below.

Below are some of the answers by people who are experienced in the field.



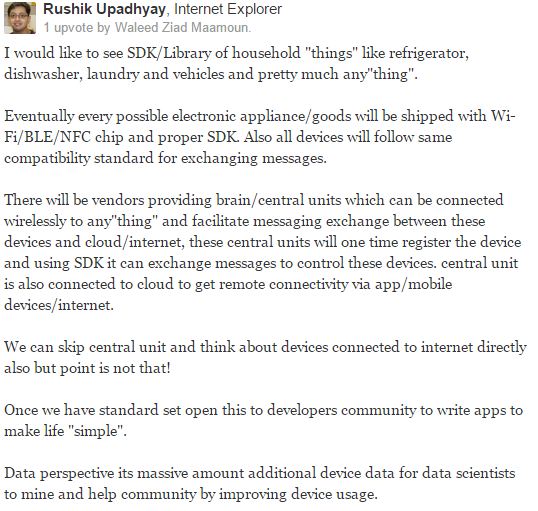
- Jim Reich

Researcher at Carnegie Mellon University



- Zhou Mingchun

IOT Expert at Samsung Electronics

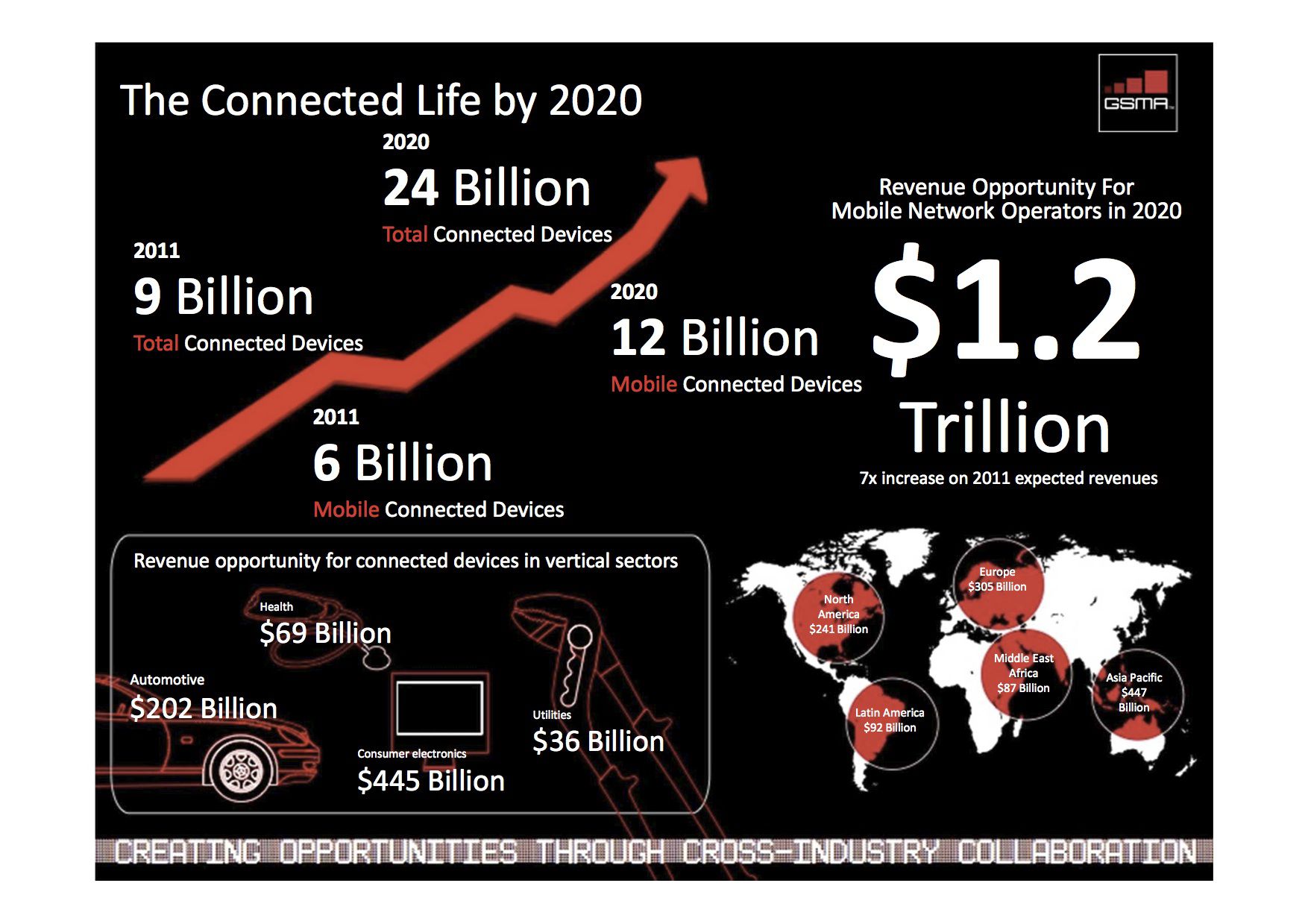


- Rushik Upadhyay

PayPal

## Study

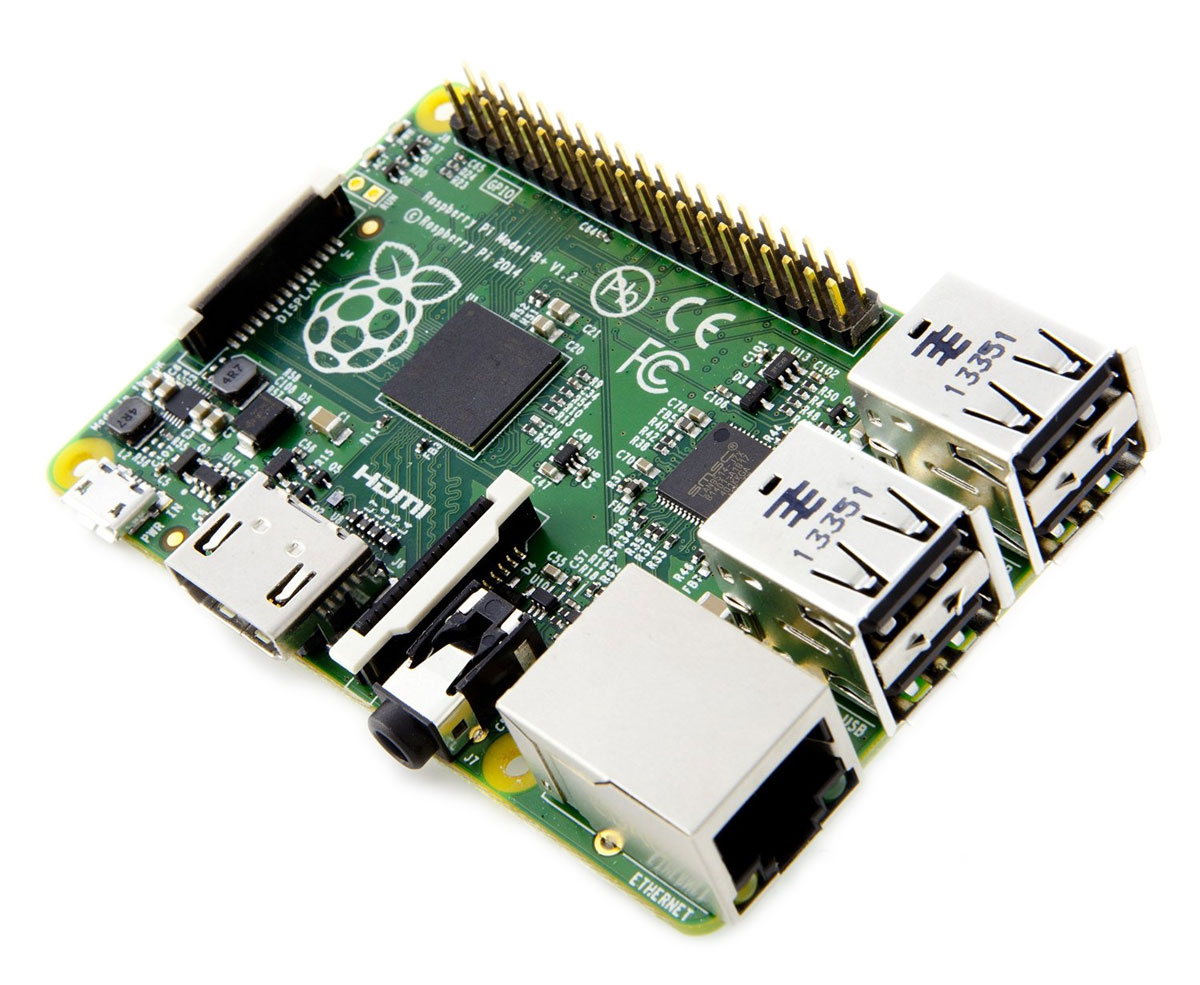
In 2011 there was estimated a number of 9 billion total connected devices to the internet, 6 billion of those were mobile connected devices. In 2020 those numbers are estimated to be 24 billion total connected devices, 12 billion of which are mobile connected devices which makes the revenue opportunity for mobile network operators in 2020 $1.2 trillion which is a 7 times increase on 2011 expected revenue. Also the revenue opportunity for connected devices in vertical sectors such as automotive, health, computer electronics and utilities are $202 billion, $69 billion, $445 billion and $36 billion respectively.



# Analysis

## Hardware: Raspberry Pi

The Raspberry Pi is a low cost, credit card sized computer that plugs into any normal TV or monitor. It enables people to explore computing, whether they are old, middle aged or young. It can also be used to teach computer programming which uses special computer languages such as Scratch and Python. Raspberry Pi came from the idea that, without an affordable solution, children of the future would grow up without being computer literate in the new age of technology.



What you can do with the Raspberry Pi vastly outweighs what you can’t do. The possibilities are practically endless. From something as simple as using it as normal computer with an office package and an internet browser to making your own retro games console. The GPIO (General-purpose input/output) terminals can be used in the same vein as a switch to turn objects on and off or even using it as a quick reaction game.

### 3.1.1 Raspberry Pi Specifications

|  |  |
| --- | --- |
| Specification |  |
| CPU | Broadcom BCM2835 SOC 700MHz |
| GPU | Broadcom VideoCore IV |
| Memory | 512 MB (shared with GPU) |
| USB | 2 x USB 2.0 |
| Video | Phono (RCA), HDMI (up to 1.4 compatible) and DSI socket |
| Audio | 3.5 mm jack |
| Storage | SD card slot |
| Network | 10/100 Mb/s |
| Power | Micro USB DC 5 V @ 800 mA for the Pi board |
| Dimensions | 85.6 mm x 53.98 mm |
| Operating Software | Supplied with Raspbian |
| I/O | 26 pin GPIO customisable pin array |
| Package Contents |  |
|  | Raspberry Pi Model B board |
|  | Wi-Fi Dongle |
|  | USB Keyboard |
|  | USB Mouse |
|  | 4 Port USB Hub |
|  | 2 Port USB PSU @ 2.1A |
|  | Micro USB Cable |
|  | SD Card with Raspbian OS pre-installed |

## Existing Products

### Xively

Xively is a division of LogMeIn Inc., a global, public company that provides remote access and collaboration products including Rescue, Boldchat, join.me, and Cubby. Xively by LogMeIn offers an Internet of Things (IoT) platform as a service, business services, and partners that enable businesses to quickly connect products and operations to the Internet.

### https://tctechcrunch2011.files.wordpress.com/2015/05/particle.pngParticle

Particle (formerly known as Spark) offers a suite of hardware and software tools to help the user prototype, scale, and manage your Internet of Things products.

With Particle, the user's prototype is backed up by a real-time, secure and scalable cloud platform. The cloud platform will support your first product to your one millionth product, with our low-cost hardware modules providing a simple connection to the cloud at scale.

# Requirements

## Introduction

### Product Overview

Our product is an Internet of things Development platform that ease the process of development application using the internet of things concept and abstracts a lot of work needed so that the developer would concentrate on the functionality of his app only.

## Specific Requirements

### External Interface Requirements

#### User Interfaces

1) A welcome screen with explanation of the project and some images to illustrate and links to other screens.

2) A login screen where the user is going to provide his user name and password to proceed to his account with a link to a forget password page.

3) Forget password page where he enters his mail and a send button

4) A sign up page where the user is going to provide his information to create a new account for him

5) After logging in the user is going to be shown his dashboard, a page where he can see how many device is connected last time each device sent anything and other details about his usage

6) When clicking on every device a page is shown for every device showing the data the device have sent statistics about the data the average and so in and some graphing about the data sent and links to the firmware code running on the device and the cloud code, when clicking on any of them it expands to show the code when the user press edit an instance of the web IDE is shown with code inside of it

7) The IDE page a text editor is shown in the left, a small window with folder view of files and folders below it a set of buttons which is {Run , Stop , Deploy } and then in the bottom center a window of output where the output of the code is viewed.

#### Hardware interfaces

##### Device side

Our target hardware device side is TI CC 3200 which is a microprocessor (ARM® Cortex®-M4 Core at 80 MHz) with a built in WiFi Receiver which is ideal for Internet of things applications.

##### Server side

Cloud code and data will run on server side on an average computer so it has no specific hardware interface and all hardware communications are managed by the underlying operating system on the web server.

#### Software Interfaces

The device code communicates with web server to send him data acquired and status and the server communicates with device code to send it commands or updated codes the server side code deals with REST requests to provide data and deals with DBMS to insert or update data in the database

#### Communication Protocols

The device is going to communicate with the web server using encrypted MQTT protocol on top of TCP/IP, the server is going to deal with client side using standard HTTP and deal with REST calls using HTTPs.

### Software product features

#### Functional Requirements

##### User login

The user should be able to login with his user name and password to the system

##### User sign up

The user should be able to create a new account while providing information like {name, Email, password}

##### User forget password

The user should enter his mail and the password be reset and sent to his mail.

##### Connect devices

The user should be able to connect to his devices and the system should provide him with a unique API key that the device can send to so the system would know to which user belong this device.

##### Accept data

The system should be able to accept data from the devices and store it for each user.

##### Showing Data

The system should provide the user with the data in tables.

##### Graphing data

The system should provide different graphing according to the data types

##### Device status

The system should provide the user with information about the status of the device including last time it sent , how many times it sent data , the time of every time data was sent , graph with these attributes

##### More than 1 device

The system should allow the users to have more than 1 connected device

##### Rest API

The system should have a rest API so that the user can hit the server and obtain his data

##### JS Library

A library would be made in Java script that handles REST and API calls that can be embedded in the developer code to ease the process of communicating with the System.

##### Online IDE

There should be an online IDE that the user can write different kinds of code in it.

##### Online IDE (2)

The IDE should have the basic features of a basic IDE like having a text editor, running code, stopping, deploying.

##### Server side code

The system should allow the user to write code that runs in the cloud based that can be triggered when his data arrives and the Rest API should handles how he gets his output the target language would be Java with a room for extension afterwards.

##### Flashing code remotely

The system should allow the user to write device code and flash it remotely to his device.

##### Device library

In order for the device to be able el be flashed remotely a library should be developed to allow this to happen.

##### Real time data

All the data sent must be handled and processed in real time as the type of applications running would be time critical.

##### Real time data (2)

In order for data to be real time the application protocol to be used is MQTT.

##### MQTT (Device)

To be able to provide MQTT a library should be developed on the device so that data can be sent using this protocol.

##### Device library (2)

In order for the server to deal with MQTT the server should have the right message broker to handle messages being sent.

### Software System Attributes

#### Reliability

The system should be reliable as in server must not crash and must be available all the time as developer applications rely on it , also data should have a backup in case any kind of damage happens

#### Security

The system of course must be secure as the data it contains is very critical and security is going to be achieved through:

1) Encrypting communication between the device and the server RSA and AES

2) Encrypting communication between server and client using SSL/TLS.

#### Maintainability

The System should be maintainable as in the future it can support languages other than java easily, also the library can have other targeted hardware so the cost of change should be minimal.

#### Performance

The library should be highly optimized and have the least footprint as possible so that there would be room for developers application specially with the limited resources of embedded devices so the library shouldn’t take more than 20% of processor time.

### Database Requirements

The database targeted would be MySQL but should have plans in the future for no SQL database like MongoDB so code dealing with database should be highly extendable.

# Design

## Architectural Design

### Introduction

Our product is an Internet of things development platform that ease the process of development application using the internet of things concept and abstracts a lot of work needed so that the developer would concentrate on the functionality of his app only.

### Architectural Representation

This document details the architecture using the views defined in the “4+1” model, but the views used to document the Internet of things platform application are:

##### Use Case View

Audience: All the stakeholders of the system, including the end-users.

##### Logical View

Audience: Designers.

##### Process View

Audience: Integrators.

##### Module Decomposition View

Audience: Programmers.

##### Data View

Audience: Data specialists, Database administrators

### Architectural Goals and Constraints

##### Server Side

IOTA will be hosted on one of Apache web servers running on a Java platform, and connecting to one of the MySQL Database servers. All communication with client has to comply with public HTTPS, MQTT , TCP/IP communication protocol standards.

##### Client Side

Users will be able to access IOTA only online. Clients/users are requiring using a modern web browser such as Mozilla or Chrome

##### Device Side

IOTA Libraries will be running on RTOS named “RTEMS” with portability options in any ARM Architecture board but target hardware which these libraries will be tested is “TI CC 3200”

#### Security

User’s devices can only be accessed , updated or removed by the users who created it , all communications with devices and server will be secured and cannot be sniffed or altered

#### Persistence

Data persistence will be addressed using a relational database.

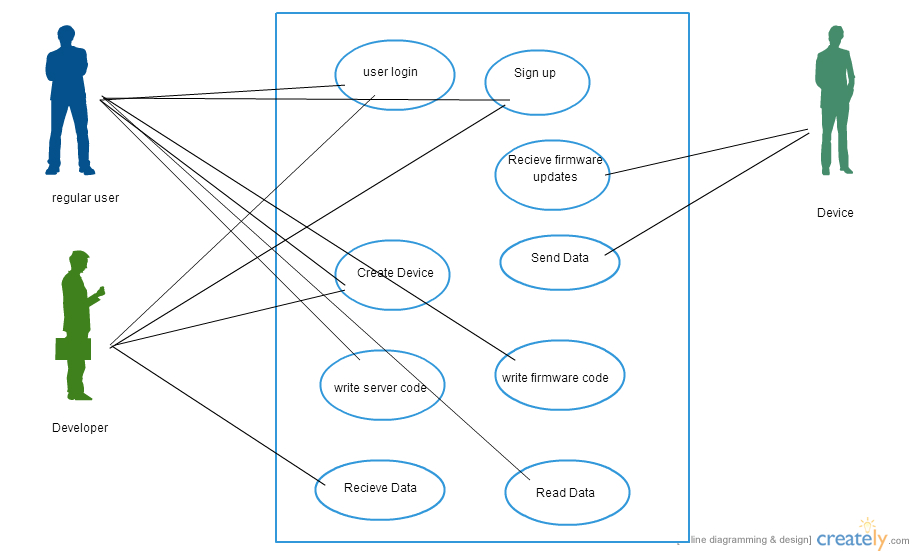
#### Performance

The system should respond to the user in a real time way meaning that the level of server responsiveness that a user senses as sufficiently immediate or that enables the server to keep up with some external process such as change in sensor reading value

### Use-Case View

This is a list of use-cases that represent major functionality of the final system

* Sign up for the user
* Login for the user
* Create a new device
* Read data sent from device
* Write server code
* Deploy firmware code in device
* Deal with the system in “programmatic” way
* Send data from device to server
* Receive firmware updates from server to device



#### Actors

As described in the actors’ correspondence diagram below, web user could be one of three types.

##### Regular User

The user who can create a device, receives data from them, write server code or deploy firmware code.

##### Developer

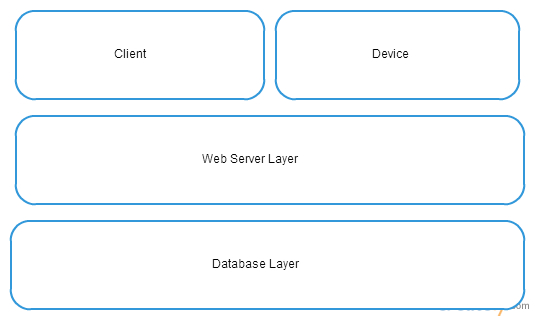
A special type of user who can do all the user roles but in programmatic way so that he can embed it in his application

##### Device

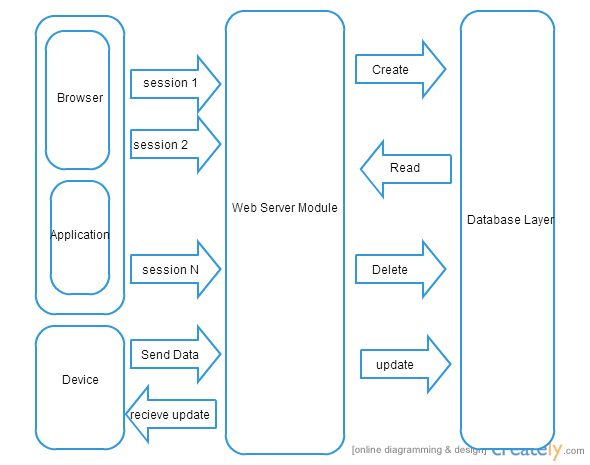
The device that sends data to the server and receives firmware updates from server

### Logical View

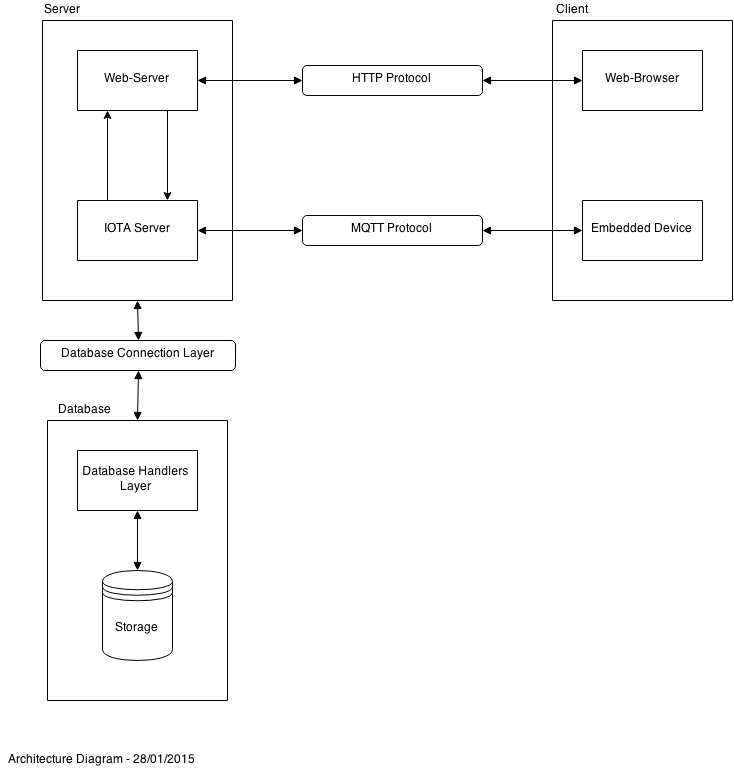
IOTA is divided into layers based on the N-tier architecture



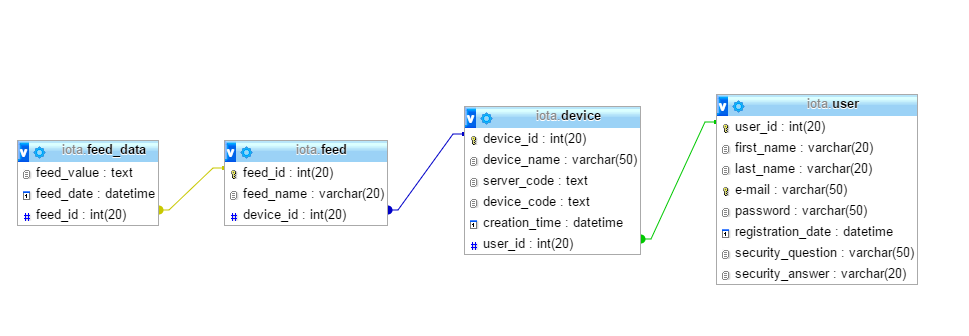
### Process View



### Module Decomposition View



### Data View



## Conceptual Design

### Experiential Flowchart

The Architectural Designs consists of three main parts

* The Server, consisting of
  + Web-Server
  + IOTA Server
* The Client, consisting of
  + Web-Browser
  + Embedded Device
* The Database, consisting of
  + Database Handlers Layer
  + Storage

The Web-Server and the Web-Browser are connected through the HTTP Protocol and the IOTA Server and Embedded Device are connected through the MQTT Protocol. While the Server and the Database are connected through a Database Connection Layer. See fig. hary.

### Interface Mockups

UI Screenshots

### Implementation Plan

##### Timeline

# Implementation

# Testing