**CHAPTER 1**

**INTRODUCTION**

1. **OVERVIEW OF THE PROJECT**

The Internet of Things can be described as connecting everyday objects like smart-phones, Internet TVs, sensors and actuators to the Internet where the devices are intelligently linked together enabling new forms of communication between things and the people. Nowadays, automation plays a crucial role and are implemented either using the microcontroller or computer. Microcontroller cannot run multiple programs at a time. With the use of Microcontroller it is difficult to control both the appliances and surveillance at a time i.e. it is very complex to perform the multiple functions simultaneously.

We can achieve this with the help of computer, but it is very expensive for this purpose and it will consume more power. The Raspberry Pi is a single board computer and it can be used to overcome these problems. Simply, the Raspberry Pi system functions like a computer with the small setup. It contains GPIO pins and the USB ports. Using these ports we can control these appliances with the help of sensors and as well as it interfaces the camera for surveillance. Raspberry Pi can be used for multiple purposes based on our requirements.

1. **SYSTEM SPECIFICATION**
   * 1. **HARDWARE REQUIREMENTS**

* Raspberry Pi Kit
* SD Card
* TV / Monitor and Keyboard
* Power Supply and Relay
* HDMI cable
* Ethernet and Relay

1. **SOFTWARE REQUIREMENT**

* Raspbian OS
* VNC Server
* VNC viewer
* Etcher
* MobaXTerm

1. **DEVELOPMENT TOOLS**

* Apache web server
* PHP
* Library wiringpi
* PYTHON
* Android studio
  1. **PROBLEM STATEMENT**

The focus of my project is on helping the users to operate home appliances with their own smartphones and to help elderly or handicapped people live a more independent life as long as possible. The objective of our system is to take care of several domestic systems that may be normally difficult for those who are handicap or elderly to take care.

The proposed idea will allow a user with any kind of android enabled device to run a piece of downloadable software such as a smartphones. This application will allow the user to control a device that is connected to any home appliance i.e. Pi enabled. The focus of this application will be to direct the security system with webcam surveillance, door sensor notification and a light control system. Sensors will be connected to the home appliances with Pi so that they can be monitored and controlled easily.

**CHAPTER 2**

**SYSTEM STUDY**

* + 1. **EXISTING SYSTEM**

Several groups have done extensive research into the use of smart home devices to help the handicap and the elder people. The University of Germany has described the challenges regarding smart homes, especially for supporting the elderly and handicapped. The purpose is to compensate for handicaps and support the individual in order to give them a more independent life. Bluetooth is a global standard for connecting a wide range of devices. It provides security by encrypting data using a 128 bit long shared key but the disadvantage of system is providing very short range of communication. Radio Frequency (RF) systems have become increasingly popular recently with the advancements in RF technology such as Bluetooth and ZigBee.

These products offer a much more reliable short range network then previous Infrared devices which had interference and security issues. But RFID tags are more expensive, less reliable and application specific i.e. no one tag fits all. This project will focus on Smartphones enabled systems for the smart home with focus on the Raspberry Pi applications. Although many systems have been researched and proposed, very few have been implemented. This project aims to build on the previous research described to implement a wireless sensor network to monitor the home appliances and can be controlled by smart phones.

**2.1.1 DRAWBACKS IN EXISTING SYSTEM**

* Implementation Cost is too high
* Slight learning curve
* Reliability of the software quality is low
* Customization based on the user needs is difficult.
* Less security and the authentication is available.

**2.2 PROPOSED SYSTEM**

Now a days there exists a digital signage, it requires huge amount of power consumption. But this raspberry pi controller needs only a few volts power which performs the whole job of computer and many people can afford it due to their low cost ($35). Our aim is to develop a user friendly, web based, cheap, effective and compact sized digital signage system which can be controlled and modified by the users. The content updated are controlled by means of Apache web server and the data from database are scheduled by Content Management System. The user denotes the client and the company denotes the organization through which we can control the contents that need to be displayed in the screen.

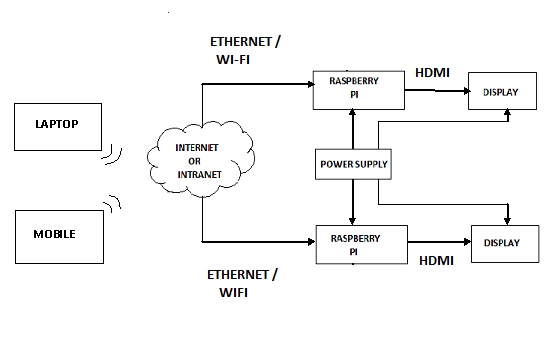
**2.2.1 ADVANTAGES OF PROPOSED SYSTEM**

* Less cost for the Implementation.
* It permits additional flexibility through humanoid devices.
* It permits a decent vary of quantifying the ability.
* Provides security and the better authentication method.
* Additional vendors may be simply superimposed.

**CHAPTER 3**

**SERVER SIDE DEVELOPMENT**

1. **BLOCK DIAGRAM**

****FIGURE 3.1.1 BLOCK DIAGRAM

**3.2 BLOCK DIAGRAM DESCRIPTION**

**3.2.1 POWER SUPPLY**

The Raspberry Pi needs an external power supply for start up. Hence the power adapter is used to turn on the kit every time with the below mentioned power ratings and the levels.

* 5v input power supply
* Model-B-5VDC,700-1500mA.

**3.2.2 DISPLAY DEVICE**

* Graphic displays are dot matrix and energy efficient.
* Electronic visual display or video display that uses the light modulating properties of liquid crystals.
* Thin film transistor (TFT) devices are screen for flat panel displays.

**3.2.3 HDMI CABLE**

* Proprietary audio/video Interface for transferring uncompressed video data and compressed or uncompressed digital audio data to a monitor.
* It implements EIA/CEA-861standard which is basically define video format and image formats with five HDMI connector like type-A, type-B, type-C, type-D, and type-E.
* We use HDMI cable for connecting our kit with the display devices for the purpose of transmitting the contents.

**3.3 KIT SPECIFICATIONS**

FEATURES MODEL B

CHIP - Broadcom BCM2835 SoC

CPU - 700MHZ Low Power ARM1176JZ-F Processor

GPU - Dual Core Video Core Multimedia Co-Processor

MEMORY - 512 MB SD RAM

ETHERNET - On Board 10/100 Ethernet RJ45 Jack

USB 2.0 - Dual USB Connector

AUDIO OUTPUT - 3.5mm jack, HDMI

**3.4 HARDWARE LAYOUT**

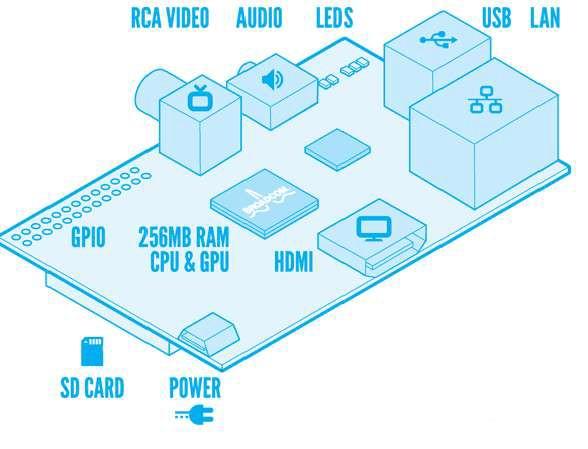


FIGURE 3.4.1 HARDWARE LAYOUT RASPBERRY PI

**3.5 HARDWARE COMPONENTS**

**3.5.1 PI BOARD**

Raspberry Pi is a credit-card sized computer manufactured and designed in the United Kingdom by the Raspberry Pi foundation. It has a Broadcom BCM2835 system on chip (soc) which includes an ARM1176JZF-S 700 MHZ processor, video core IV GPU and was originally shipped with 512 megabytes of RAM. It use SD card for booting. Python is used as programming language and it supports HDMI Video with resolution upto 1920\*1200. The Raspberry Pi can be 24 connected to a HDTV or a computer display via HDMI or to most TV and video capture devices via an RCA composite video jack. It can be powered by a micro USB adapter.

The operating system is stored on a SD card and the device plays high definition video, has audio and access to some of the micro processors input/output pins (known as the GPIO). The HDMI plugs directly into newer TV‘s and Monitors and displays Hi-Defquite well. There is also a composite video for older monitors and TV‘s. The Pi has two USB 2.0 ports and it can be used for keyboard and the mouse. Once you start using the Pi, you may want to get an USB powered hub to provide access to USB flash drives, USB external hard drives, web cams or many other accessories.

**3.5.2 GPIO (GENERAL-PURPOSE INPUT/OUTPUT)**

It is a generic pin on an integrated circuit whose behaviour including whether it is an input or output pin.The production Raspberry Pi board has a 26-pin 2.54 mm (100 mil) expansion header, marked as P1 arranged in a 2x13 strip. They provide 8 GPIO pins plus access to I²C, SPI, UART) as well as +3.3 V, +5 V and GND supply lines. GPIO pins to connect it to the PIR sensor for means of motion detection. These pins can also be used for other interfaces.

GPIO pins can be configured to be input or output

* GPIO pins can be enabled/disabled.
* Input values are readable (typically high=1, low=0).
* Output values are writable/readable.

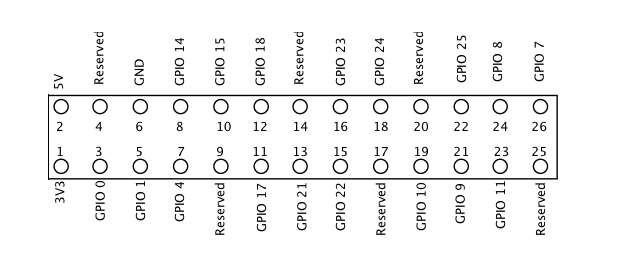


FIGURE 3.5.2.1 GPIO PINS

**3.5.3 AUDIO JACK**

A standard 3.5 mm TRS connector is available on the Raspberry Pi for stereo audio output. Any headphone or 3.5mm audio cable can be connected directly. Although this jack cannot be used for taking audio input, USB mic or USB sound cards can be used.

**3.5.4 RCA VIDEO**

RCA Video outputs (PAL and NTSC) are available on all models of Raspberry Pi. Any television or screen with a RCA jack can be connected with the RPi.

**3.5.5 STATUS LEDs**

Status LEDs represents the availability of the internet connection. There are 5 status LEDs on the RPi that show the status of the activities as follows:

* OK‖–SD Card Access (via GPIO16) - labeled as "OK" on Model B Rev1.0 boards and "ACT" on Model B Rev2.0 and Model A boards
* POWER‖ - 3.3 V Power - labeled as "PWR" on all boards
* FDX‖ - Full Duplex (LAN) (Model B) - labelled as "FDX" on all boards
* LNK‖ - Link/Activity (LAN) (Model B) - labelled as "LNK" on all boards
* 10M/100‖ - 10/100Mbit (LAN) (Model B) - labelled (incorrectly)

**3.5.6 USB 2.0 PORT**

USB 2.0 ports are the means to connect accessories such as mouse or keyboard to the Raspberry Pi. There is 1 port on Model A, 2 on Model B and 4on Model B+. The number of ports can be increased by using an external powered USB hub which is available as a standard Pi accessory.

**3.5.7 DISPLAY DEVICES**

A liquid-crystal display (LCD) is a [flat panel display](http://en.wikipedia.org/wiki/Flat_panel_display), [electronic visual display](http://en.wikipedia.org/wiki/Electronic_visual_display), or [video display](http://en.wikipedia.org/wiki/Video_display) that uses the light modulating properties of [liquid crystals](http://en.wikipedia.org/wiki/Liquid_crystal). Liquid crystals do not emit light directly. LCD typically consists of a layer of [molecules](http://en.wikipedia.org/wiki/Molecule) aligned between two [transparent](http://en.wikipedia.org/wiki/Transparency_%28optics%29) [electrodes](http://en.wikipedia.org/wiki/Electrode), and [two polarizing](http://en.wikipedia.org/wiki/Polarizer) [filters](http://en.wikipedia.org/wiki/Filter_%28optics%29) (parallel and perpendicular), the axes of transmission of which are (in most of the cases) perpendicular to each other. Without the [liquid crystal](http://en.wikipedia.org/wiki/Liquid_crystal) between the polarizing filters, light passing through the first filter would be blocked by the second (crossed) polarizer .LCDs are available to display arbitrary images (as in a general-purpose computer display) or fixed images which can be displayed or hidden.

**3.5.8 SYSTEM ON CHIP (SOC)**

A system on a chip is an integrated circuit (IC) that integrates all components of a computer or other electronic system into a single chip. It may contain digital, analog, mixed- signal, and often radio-frequency functions.

* Low power consumption.
* Memory blocks includes ROM, RAM, EEPROM and flash memory.
* Timing sources including oscillators and phase-locked loops.
* Peripherals including counter-timers, real-time timers and power-on reset generators.
* External interfaces including industry standards such as USB, FireWire, Ethernet, USART, SPI.

**3.5.9 RELAY CIRCUIT**

A Relay is electrically operated switches, which allow low power circuits to switch a relatively high voltage or current on/off. For a relay to operate a suitable pull in and holding current should be passed through its coil. Relay coils are designed to operate from a particular voltage often its 5V or 12V. The function of relay driver circuit is to provide the necessary current energize the relay coil, when a LOGIC 1 is written on the PORT PIN thus turning on the relay. The relay is turn off by writing LOGIC 0 on the port pin. In our system four relays are used for device control.

**3.5.10 OPERATING SYSTEM**

The Raspberry Pi primarily uses Linux kernel-based operating systems. The ARM11 is based on version 6 of the ARM which is no longer supported by several version so by several popular versions of Linux. The OS included with NOOBS,

* Archlinux ARM
* Pidora (Fedora Remix)
* Raspbmc and the XBMC open source digital media center
* RISC OS – The operating system of the first ARM-based computer
* Raspbian (recommended)

**3.5.11 BOOT PROCESS**

The Raspberry Pi does not boot as a traditional computer. The Video Core i.e. the Graphics processor actually boots before the ARM CPU. The boot process of the Raspberry Pi can be explained as follows:

* When the power is turned on, the first bits of code to run is stored in a ROM chip in the SoCs and is built into the Pi during manufacture. This is the called the first-stage boot loader.
* The SoCs is hardwired to run this code on startup on a small RISC Core (Reduced Instruction Set Computer). It is used to mount the FAT32 boot partition in the SD Card so that the second-stage boot loader can be accessed. This file can be seen while mount process of an OS.
* The first-stage boot loader has not yet initialized the ARM CPU (meaning CPU is in reset) or the RAM. So, the second-stage boot loader also has to run on the GPU.
* The bootloader.bin file is loaded into the 128K 4 way set associative L2 cache of the GPU and then executed. This enables the RAM and loads start.elf which is also in the SD Card. This is the third-stage boot loader and is also the most important. It is the firmware for the GPU, meaning it contains the settings or in our case, has instructions to load the settings.

**3.5.12 ADVANTAGES**

* Small in Size
* Low Cost
* Low Power Consumption

**3.5.13 APPLICATIONS**

* Teaching programming concepts.
* Teaching hardware interfacing.
* Robotics for controlling motors, sensors.
* Media centre
* Digital photo frames, tablets.
* Arcade Machines
* Tablet Computer
* Home Automation
* Decoration lights controller
  1. **HARDWARE SETUP**

**3.6.1 ASSEMBLING KIT**

* Plug the preloaded SD Card into the Pi.
* Plug the USB keyboard and mouse into the Pi , perhaps via a USB Hub. Connect the Hub to power, if necessary.
* Plug the video cable into the screen(TV) and into the Pi.
* Plug your extras into the Pi(USB WiFi ,Ethernet cable, hard drive etc.). This is where you may really need a USB Hub.
* Ensure that your USB Hub(if any) and screen are working.
* Plug the power source into the main socket.
* With your screen on, plug the other end of the power source into the Pi.
* The Pi should boot up and display messages on the screen.



FIGURE 3.6.1.1 ASSEMBLED KIT

**3.6.2 INSTALLING AND BOOTING RASPBERRY PI**

* By means of Win 32 Disk Imager, the Raspbian OS is written into the SD Card. And connect the SD Card to the kit.
* Install and boot all necessary file in SD card and power up the board.
* Then connect HDMI cable with raspberry pi and with display device.
* Using Wi-Fi Dongle , the Internet connection been given to the kit.
* Using Connectify the IP address of raspberry pi is noted.
* In PUTTY Terminal, the IP address been entered to the Pi Desktop.
  1. **SOFTWARE SETUP**
     1. **WIN32 DISK IMAGER**

Inorder to write the Os disk image file to SD card, we are using Win32 Disk Imager. It is a free utility available for windows program for saving and restoring images from removable drives (USB drives SD memory cards).It can be used to write boot images to SD flash drives, making it bootable. This program is designed to write a raw disk image to a removable device or backup a removable device to a raw image file. It is very useful for embedded development namely ARM development projects. The figure below is the win 32 disk imager is used to write the image file in the Secure Digital Card.

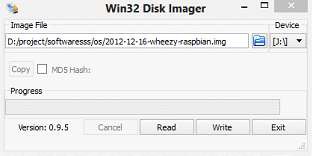


FIGURE 3.7.1.1 WIN32 DISK IMAGER

**3.7.2 PUTTY**

PUTTY is a terminal used to enter into the Pi desktop. It is a free and open source terminal emulator, serial console and network file transfer application. To enable windows PC server as a router over Ethernet or Wi-Fi, we are Using Connectify. It act as a wireless access point and enables users to share files and Internet connections between multiple computing devices without the need for a separate physical access point or router and provides security.

* + 1. **APACHE WEB SERVER**

Apache provides the mechanics for getting a Web page to a user. Apache is a stable, mission-critical-capable server, and it runs more than 65 percent of all Web sites on the Internet. The PHP component actually sits inside Apache, and we use Apache and PHP together to create your dynamic pages. Apache is the most popular web server available. A web server's job is basically to accept requests from clients and send responses to those requests. A web server gets a URL, translates it to a filename (for static requests), and sends that file back over the internet from the local disk, or it translates it to a program name (for dynamic requests), executes it, and then sends the output of that program back over the internet to the requesting party.

If for any reason, the web server was not able to process and complete the request, it instead returns an error message. The word, web server, can refer to the machine (computer/hardware) itself, or the software that receives requests and sends out responses. The raspbian os is installed by means of writing the disk image file into the sd card through win 32 disk imager. It is then booted up by means of connecting the kit with a network. The apache server is being installed as follows

* sudo apt-get install apache5 php
  + 1. **LIBRARY WIRING PI**

Wiring Pi is the  GPIO Interface library for the Raspberry Pi , it is a PIN based GPIO access library written in C for the BCM2835, BCM2836 and BCM2837 SoC devices used in all Raspberry Pi. versions. It’s released under the GNU LGPLv3 license and is usable from C, C++ and RTB (BASIC) as well as many other languages with suitable wrappers (See below) It’s designed to be familiar to people who have used the Arduino “wiring” system1 and is intended for use by experienced C/C++ programmers. It is not a newbie learning tool.

WiringPi is extendable and modules are provided to extend wiringPi to use analog interface devices on the Gertboard, and to use the popular MCP23x17/MCP23x08 (I2C 7 SPI) GPIO expansion chips, as well as  module that will allow blocks of up to 4 74×595shift registers to be daisy-chained together for an additional 32-bits worth of output as a single unit. (You can have several blocks of 4 74x595s if needed). WiringPi supports analog reading and writing, and while there is no native analog hardware on a Pi by default, modules are provided to support the Gertboards analog chips and other A/D and D/A devices can be implemented relatively easily.

**3.7.5 WIRINGPI DEVLIB**

The devLib is a set of library routines implemented using wiringPi to give you easy access to some popular peripherals. Devices supported include character LCD displays (based on the Hitachi HD44780U chips), and graphical ones – e.g. the common 128×64 pixel displays with the generic 12864H driver chip. The DS1302 RTC clock chip, sensors based on the Maxdetect chips (e.g. RHT003) the Gertboard and PiFace interface boards and so on.

**3.7.6 PINS**

 Pin numbering of the BCM2835 GPIO port(s) on the Raspberry Pi has been a source of great confusion since the designs for the Pi were first published. In the early days (even before hardware was available) the default usable GPIO pins were simply referred to by number as GPIO0 through GPIO7. Additionally there were pins for other purposes, SPI, I2C and serial. This was highlighted on the original image on the eLinux Raspberry Pi Wiki site too.

So when initially writing wiringPi, I chose to have the same default pin numbering scheme and numbered them from 0 upwards. This is no different to how the Arduino operates – “Pin 13” on the Arduino is Port B, bit 5 for example. The underlying hardware definitions are hidden by a simplified numbering scheme. On the Pi, using wiringPi, pin 0 is BCM\_GPIO pin 17 for example) wiringPi supports its own pin numbering scheme as well as the BCM\_GPIO pin numbering scheme, and as of Version 2, it also supports the physical hardware pin numbers (for the P1 connector only), but I would like to suggest you stick to the simplified wiringPi pin numbers. That way your programs will be portable over different hardware revisions without needing any changes.

The following tables give the mapping of the Raspberry Pi GPIO Pins to the (P1) GPIO connector in relation to the pin numbers and the physical location on the connector. This is a representation of the GPIO connector as viewed looking at the board from above. The GPIO connector is to the top-right of the board with the Ethernet and USB sockets to the bottom.

Since the 26-pin GPIO connectors, a new 40-pin connector has appeared on newer Pi’s. There is also the compute Module boards.

The best way to get a description of the GPIO connector on whatever Pi it is currently running on is to use the gpio command:

$ gpio readall

**3.7.4 PHPMYADMIN**

PhpMyAdmin is a [free and open source](http://en.wikipedia.org/wiki/Free_and_open_source) tool written in [PHP](http://en.wikipedia.org/wiki/PHP) intended to handle the administration of [MySQL](http://en.wikipedia.org/wiki/MySQL) with the use of a [web browser](http://en.wikipedia.org/wiki/Web_browser). It can perform various tasks such as creating, modifying or deleting [databases](http://en.wikipedia.org/wiki/Database), [tables](http://en.wikipedia.org/wiki/Table_%28database%29), [fields](http://en.wikipedia.org/wiki/Field_%28computer_science%29) o[r rows](http://en.wikipedia.org/wiki/Row_%28database%29); executin[g SQL](http://en.wikipedia.org/wiki/SQL) statements; or managing users and permissions. phpMyAdmin had already become one of the most popular PHP applications and MySQL administration tools, with a large community of users and contributors. phpMyAdmin supports a wide range of operations on MySQL, MariaDB and Drizzle. PhpMyAdmin is a very popular MySQL management software package.

Features provided by the program include:

1. Web interface

2. MySQL database management

3. Import data from [CSV](http://en.wikipedia.org/wiki/Comma-separated_values) and [SQL](http://en.wikipedia.org/wiki/SQL)

4. Administering multiple servers

5. Creating PDF graphics of the database layout

**CHAPTER 4**

**CLIENT SIDE DEVELOPMENT**

* 1. **ANDROID STUDIO**

Android Studio is Android's official IDE. It is purpose built for Android to accelerate your development and help you build the highest-quality apps for every Android device. It offer tools custom-tailored for Android developers, including rich code editing, debugging, testing, and profiling tools.

**4.2 REQUIREMENTS**

* Linux, GNOME or KDE desktop
* Tested on Ubuntu® 14.04 LTS, Trusty Tahr (64-bit distribution capable of running 32-bit applications)
* 64-bit distribution capable of running 32-bit applications
* GNU C Library (glibc) 2.19 or later
* 3 GB RAM minimum, 8 GB RAM recommended; plus 1 GB for the Android Emulator
* 1280 x 800 minimum screen resolution
* Android Studio provides GUI tools that simplify the less interesting parts of app development.
  1. **LAYOUT EDITOR**

When working with XML layout files, Android Studio provides a drag-and-drop visual editor that makes it easier than ever to create a new layout. The Layout Editor was built in unison with the Constraint Layout API, so you can quickly build a layout that adapts to different screen sizes by dragging views into place and then adding layout constraints with just a few clicks.

**4.4 VECTOR ASSEST STDUIO**

Android Studio makes it easy to create a new image asset for every density size. With Vector Asset Studio, you can select from Google-provided material design icons or import an SVG or PSD file. Vector Asset Studio can also generate bitmap files for each screen density to support older versions of Android that don't support the Android vector drawable format.

**4.4.1 APK ANALYZER**

You can use the APK Analyzer to easily inspect the contents of your APK. It reveals the size of each component so you can identify ways to reduce the overall APK size. It also allows you preview packaged assets, inspect the DEX files to troubleshoot multidex issues, and compare the differences between two APKs.

**4.4.2 TRANSLATION EDITOR**

The Translations Editor gives you a single view of all of your translated resources, making it easy to change or add translations, and to find missing translations without opening each version of the strings.xml file. It even provides a link to order translation services.

**4.4.3 INSTANT RUN**

Android Studio's Instant Run feature pushes code and resource changes to your running app. It intelligently understands the changes and often delivers them without restarting your app or rebuilding your APK, so you can see the effects immediately.

**4.4.4 FAST AND FEATURE-RICH EMULATOR**

The Android Emulator installs and starts your apps faster than a real device and allows you to prototype and test your app on various Android device configurations: phones, tablets, Android Wear, and Android TV devices.

**4.4.5 INTELLIGENT CODE EDITOR**

The code editor helps you write better code, work faster, and be more productive by offering advanced code completion, refactoring, and code analysis. As you type, Android Studio provides suggestions in a dropdown list. Simply press Tab to insert the code.

**4.4.6 CONFIGURE BUILDS WITHOUT LIMITS**

Android Studio's project structure and Gradle-based builds provide the flexibility you need to generate APKs for all device types.

**4.4.7 ROBUST AND FLEXIBLE BUILD SYSTEM**

Android Studio offers build automation, dependency management, and customizable build configurations. You can configure your project to include local and hosted libraries, and define build variants that include different code and resources, and apply different code shrinking and app signing configurations.

**4.4.8 OPTIMIZED FOR ALL ANDROID DEVICES**

Android Studio provides a unified environment where you can build apps for Android phones, tablets, Android Wear, Android TV, and Android Auto. Structured code modules allow you to divide your project into units of functionality that you can independently build, test, and debug.

**4.4.9 DESIGNED FOR TEAMS**

Android Studio integrates with version control tools, such as GitHub and Subversion, so you can keep your team in sync with project and build changes. The open source Gradle build system allows you to tailor the build to your environment and run on a continuous integration server such as Jenkins.

**CHAPTER 5**

**SYSTEM DESCRIPTION**

**5.1 MODULES**

Modules of home automation are

* Getting Started
* Number of Relay
* Setup Relay
* PI URL Address

**5.2 MODULE DESCRIPTION**

**5.2.1 GETTING STARTED**

Getting started is the documentary section it has the help menu and the setup option for the different raspberry pi based on the pins of the raspberry pi and the required software are provided in downloadable links and these figure shows the connectivity of raspberry pi with the library wiring pi.

**5.2.2 NUMBER OF RELAY**

The number of relay used in raspberry pi can be matched in the software section in order to work them correctly based on the pins and the name and the icon can be customized by the user in different ways in order to use them more conveniently.

**5.2.3 SETUP RELAY**

Setup relay is the place where the user can customize the name of the relay the number cannot be changed in the section but all the relays appear in the home page can be changed based on the dependencies. It has various option such changing the relay label assigned pi and the wiring pin and the relay icon.

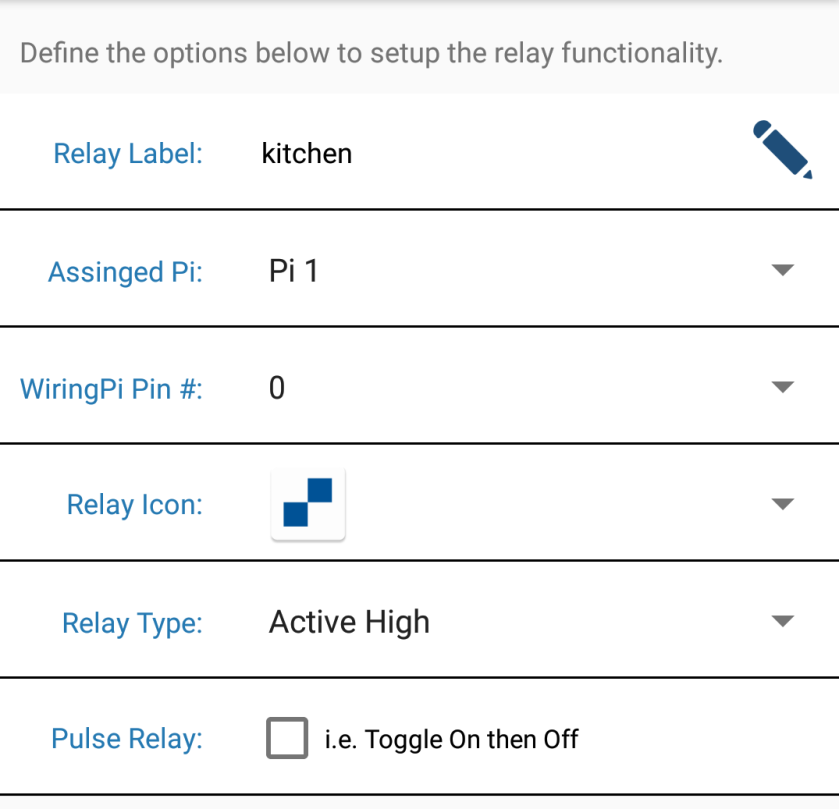


FIGURE 5.2.3.1 SETUP RELAY

**5.2.4 PI URL ADDRESSES**

Pi url address has to set the url of the raspberry pi each relay can be assigned to a pi in the setup relays settings menu.

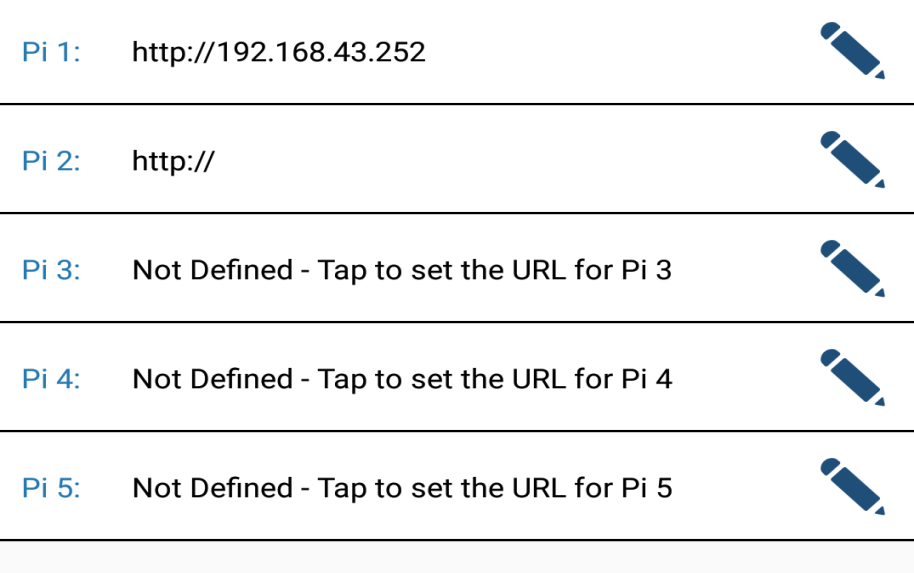


FIGURE 5.2.4.1 URL ADDRESS

**CHAPTER 6**

**SYSTEM TESTING**

System testing involves unit testing, integration testing, white-box testing, black-box testing. Strategies for integration software components into a functional product include the bottom-up strategy, the top-down strategy, and sandwich strategy. Careful planning and scheduling are required to ensure that modules that will be available for integration into evolving software product when needed a serious of testing are performed for the proposed system before the system is ready for user acceptance testing. Below are the types of testing:

* Unit Testing
* Validation Testing
* Performance Testing
* Usability Testing

**6.1 UNIT TESTNG**

Instead of testing the system as a whole, Unit testing focuses on the modules that make up the system. Each module is taken up individually and tested for correctness in coding and logic. The advantages of unit testing are:

* Size of the module is quite small and that errors can easily are located.
* Confusing interactions of multiple errors in wide different parts of the software is eliminated.
* Modules level testing can be exhaustive.

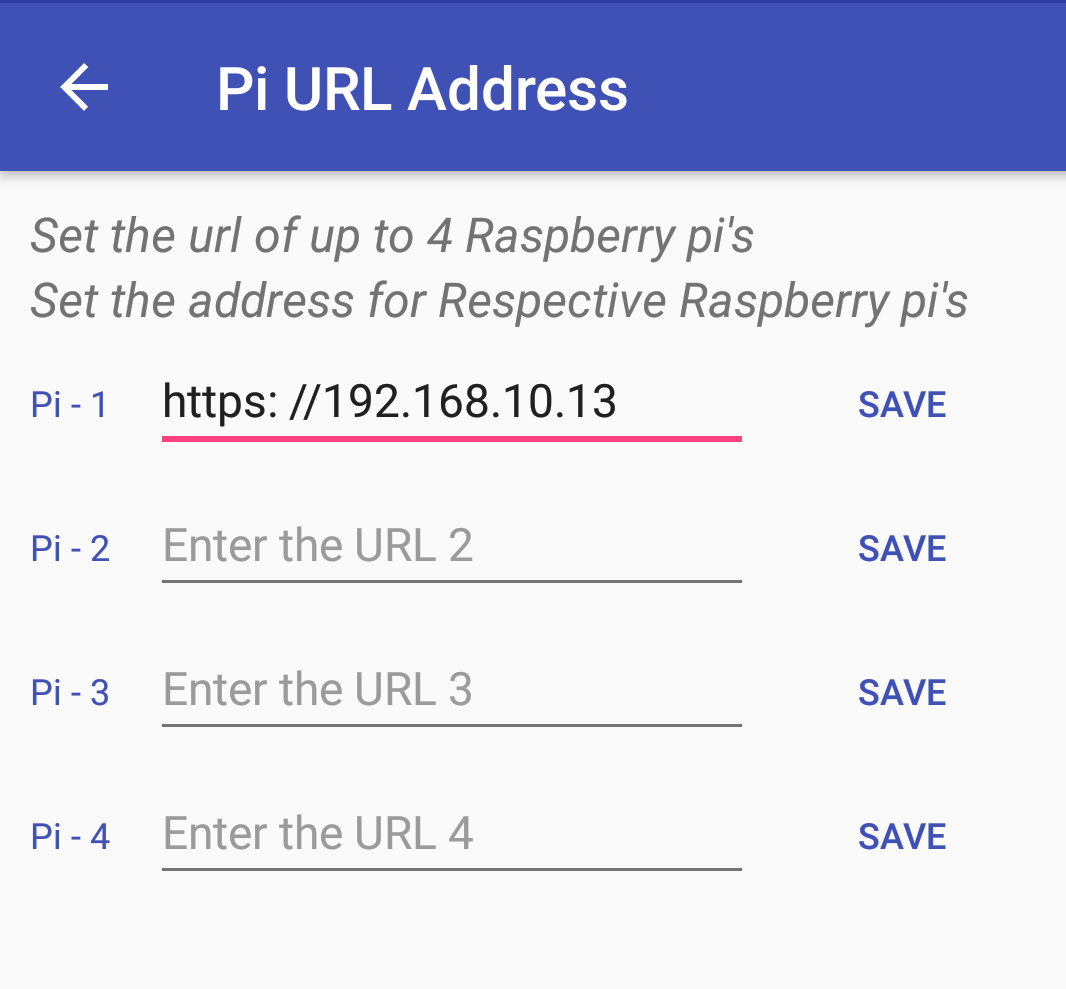


FIGURE 6.1.1 UNIT TESTING

**6.2 VALIDATION TESTING**

After the culmination of black box testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin. Validation testing can be defined as many, but a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer. Validation refers to the process of using the software in a live environment to find errors. During the course of validation system may occur and the software will be changed.

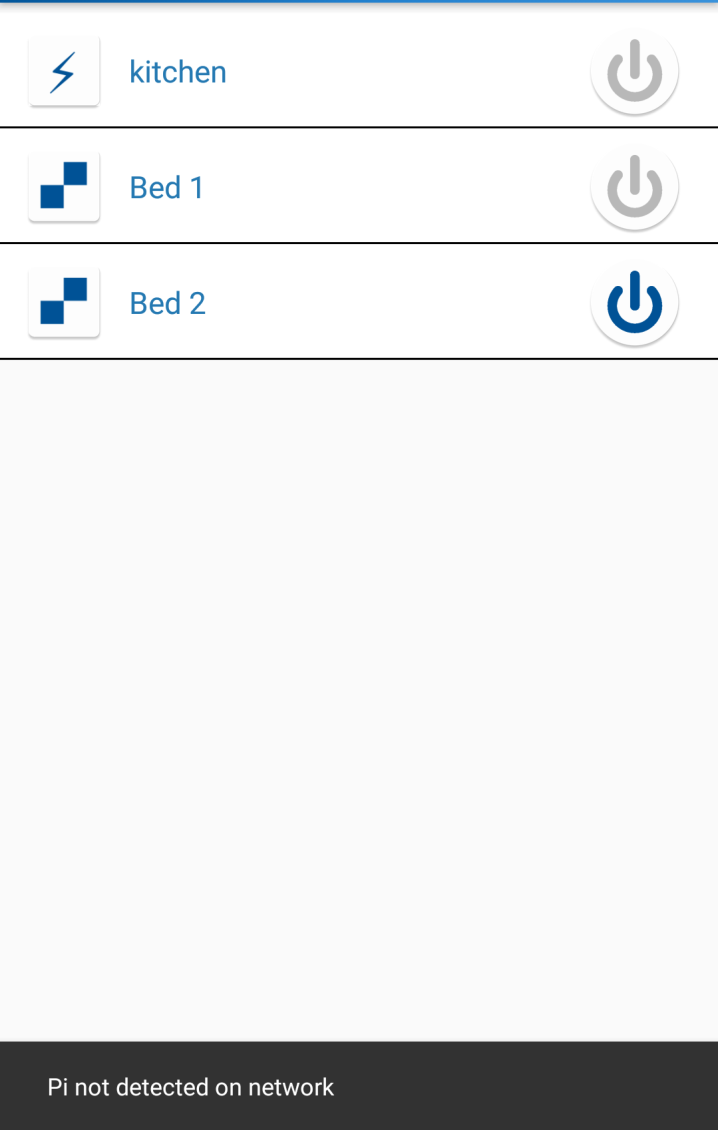


FIGURE 6.2.1 VALIDATION TESTING

**6.3 PERFORMANCE TESTING**

Performance Testing is a type of testing to ensure software applications will perform well under their expected workload.

Features and Functionality supported by a software system is not the only concern. A software application's performance like its response time, reliability, resource usage and scalability do matter. The goal of Performance Testing is not to find bugs but to eliminate performance bottlenecks The focus of Performance Testing is checking a software program's

* Speed - Determines whether the application responds quickly
* Scalability - Determines maximum user load the software application can handle.
* Stability - Determines if the application is stable under varying loads

Performance Testing is popularly called as “Perf Testing” and is a subset of performance engineering.

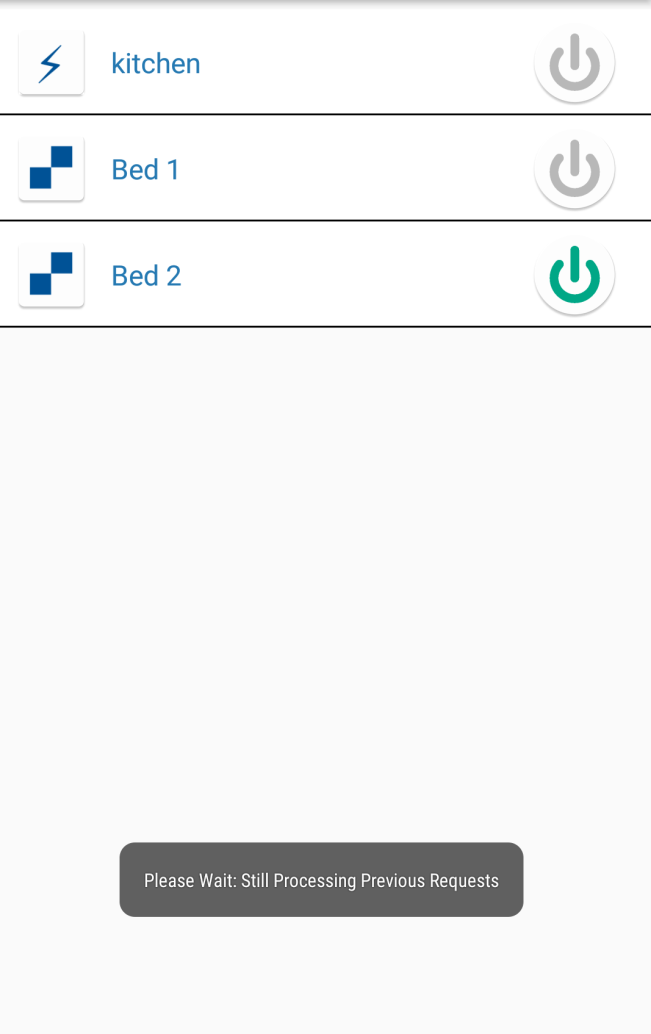


FIGURE 6.3.1 PERFORMANCE TESTING

**6.4 USABILITY TESTING**

Usability Testing is a type of software testing where, a small set of target end-users, of a software system, it to expose usability defects. This testing mainly focuses on the user's ease to use the application, flexibility in handling controls and ability of the system to meet its objectives. It is also called User Experience Testing. This testing is recommended during the initial design phase of SDLC, which gives more visibility on the expectations of the users.

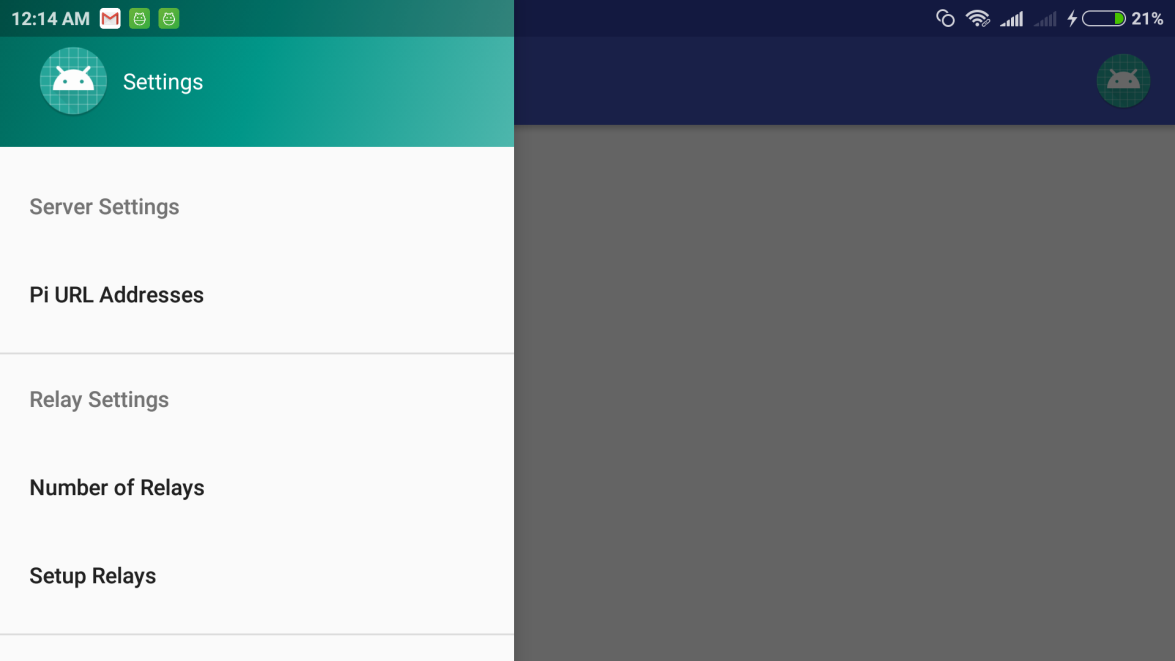


FIGURE 6.4.1 USABILITY TESTING

**CHAPTER 7**

**IMPLEMENTATION AND MAINTANANCE**

**7.1 SYSTEM IMPLEMENTATION**

The user then changes over to this new fully tested system and old system is discontinued. Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective.

Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user. The major steps involved in the implementation are:

* Careful planning
* Investigation of the current system and its constraints
* Design of methods to achieve the change over
* The implementation phase comprises of several Training Activities

**7.2 SYSTEM MAINTENANCE**

The maintenance phase of the software life cycle is the time period in which a software product performs useful work. For maintaining this system properly the following points are to be followed strictly. The executable file of forms and reports are given to the end users. Also the backup should be taken in order to safe guard the system. Software product enhancement may involve providing new functional capabilities, improving use displays and modes of interaction, upgrading external documents and internal documentation. Problem correction involves modification and revalidation of software to correct errors. Software maintenance is a microcosm of software development cycle. Enhancement and adaptation of software, reinitiates development in the analysis phase while correction of a software problem may reinitiates then development cycle in the analysis phase, design phase or implementation phase. Thus all the tools and techniques used to develop software are potentially useful for software maintenance.

**CHAPTER 8**

**CONCLUSION AND FUTURE ENHANCEMENT**

**8.1 CONCULSION**

The devices produced enable the user to control the appliances using pre-existing devices such as their Smartphone or home computer. The interfaces are intuitive and easy to use and provide the user with a more accessible interface then those found in the home. The devices are also very easy to integrate into existing applications and require only a small amount of expertise to install. Our research shows the many types of applications for implementing home automation and the applications are not limited to those discussed in this paper.

The technology used could be implemented in a wide variety of applications that require the use of sensors and appliances. This project successfully designed a system that communicates with a mobile device such as a Smartphone or laptop via Raspberry Pi to control a door sensors and a light switches and a camera to stream live video, but has many possible applications that could benefit from this work.

**8.2 SCOPE FOR FUTURE ENHANCEMENT**

Even though home automation is not a new industry, it still is not a standard for most homeowners. Because this is an industry that has experienced slow growth for a few decades now, it is hard to predict where exactly home automation is headed in the next several years. Still, considering that personal electronic devices like smartphones and tablets are in our hands or pockets all day long, it seems only natural that home automation ”which can easily interact with these personal devices is close to being just as important to homeowner, too.

One big change expected in coming years is the introduction of new technology for use in home security systems. Just this last year, ADT released a user friendly home security system that can connect with up to 16 different security cameras and can be easily managed from an Internet browser by anyone with intermediate computer skills. Even more advanced is the face-recognition camera by Netatmo, also introduced in 2015. This camera can be programmed to recognize certain faces, like friends and family members, and will send an alert to the administrator smart phone when they arrive at their front door. And if an unrecognized face arrives at your front door, you will also receive an alert and be able to watch a live feed from your smart phone.

**CHAPTER 9**

**BIBLIOGRAPHY**

**9.1 BOOK REFERENCES**

[1] Remote-Controlled Home Automation Systems with Different Network Technologies.

[2] Home Automation using ZigBee Protocol

[3] LaurI., “Microcontroller based home automation system with security”, International Journal of Advanced Computer Science and Applications, vol. 1, no. 6, pp. 60-65, 2010.

[4] N. Sriskanthan and Tan Karande, “Bluetooth Based Home Automation Systems,” Journal of Microprocessors and Microsystems, 2002, Vol. 26, pp. 281-289

**9.2 WEBSITE REFERENCES**

[1] http://www.instructables.com/id/RaspberryPi-Android-Switch-Home-automation/

[2] https://www.hackster.io/AnuragVasanwala/

[3] https://www.techopedia.com/definition/27747/raspberrypi

[4] http://www.extremetech.com/computing/124317-whatis-raspberry-pi-2.

[5] http://www.extremetech.com/newforum/137-raspberry-pi-2.

**CHAPTER 10**

**APPENDICES**

**10.1 SCREEN SHOTS**

**10.1.1 AUTOMATION MENU**

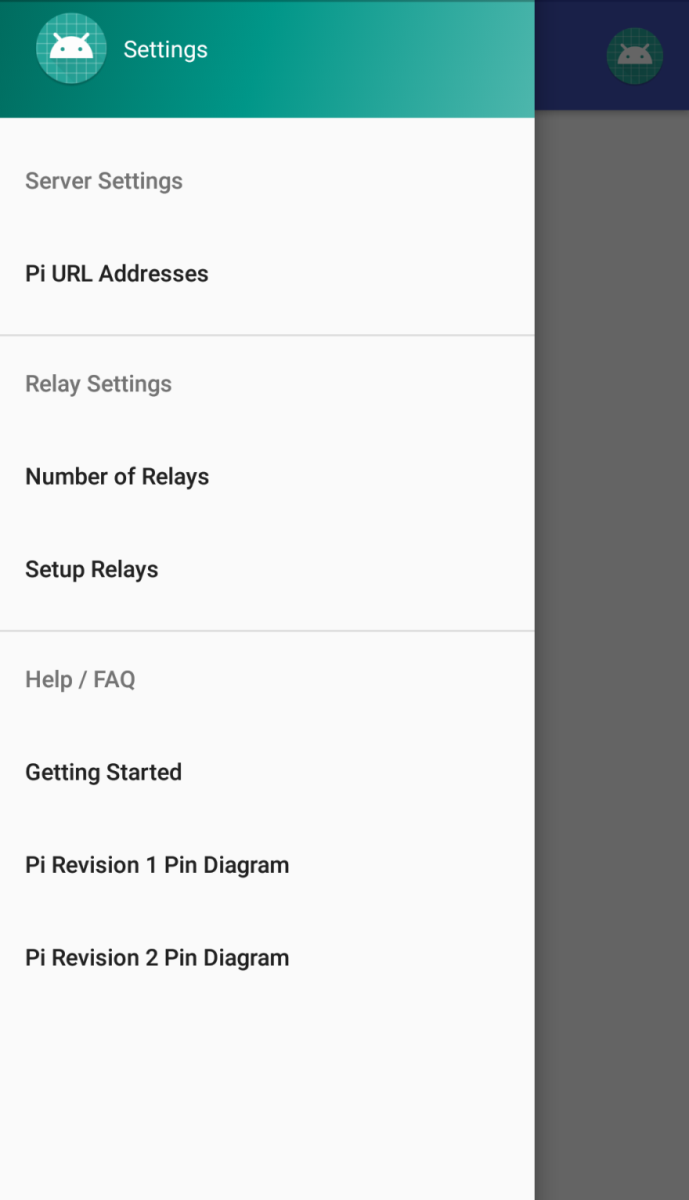
****

FIGURE 10.1.1 AUTOMATION MENU

This page contains various options for entering url addresses for the pi and the number of relays and setup the relay with the getting started help menu

**10.1.2 PI URL ADDRESSES**

****

FIGURE 10.1.2 PI URL ADDRESSES

This Page has option to enter the url for the address communication with the board and the android app

**10.1.3 EDITING URL**

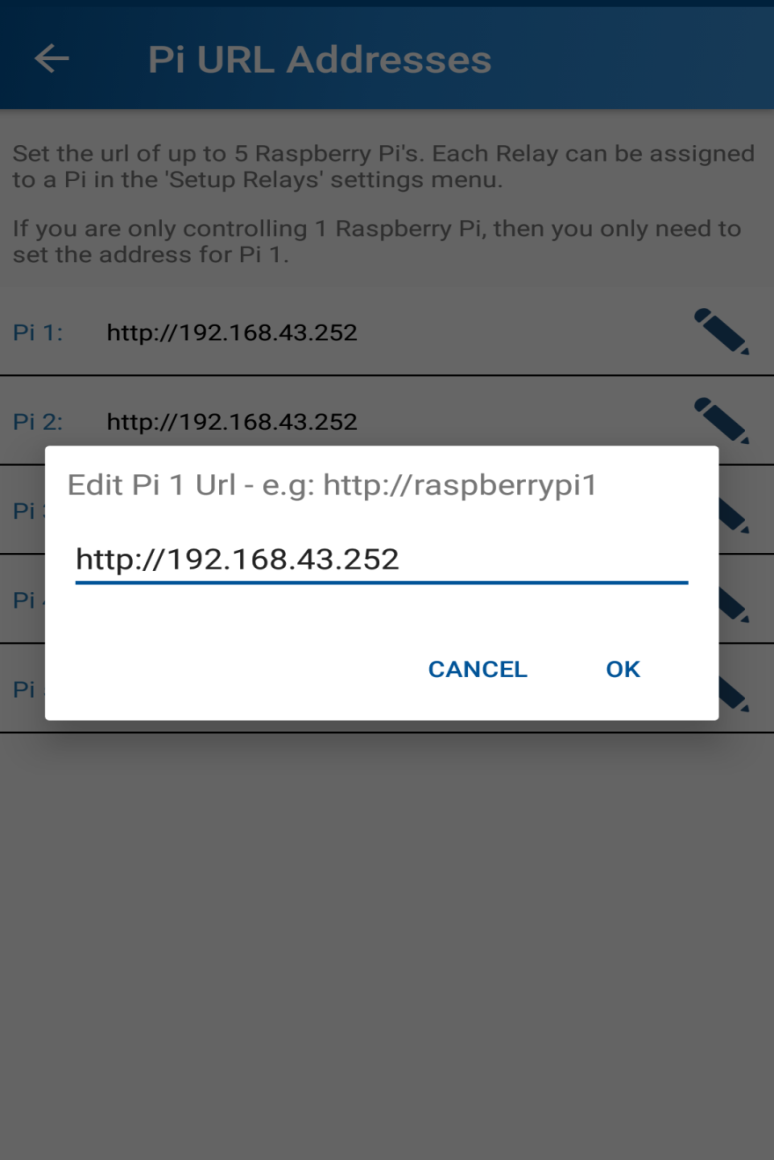
****

FIGURE 10.1.3 EDITING URL

This page is used to edit the url of the raspberry pi to use to connect the relays

**10.1.4 CHANGING RELAY**

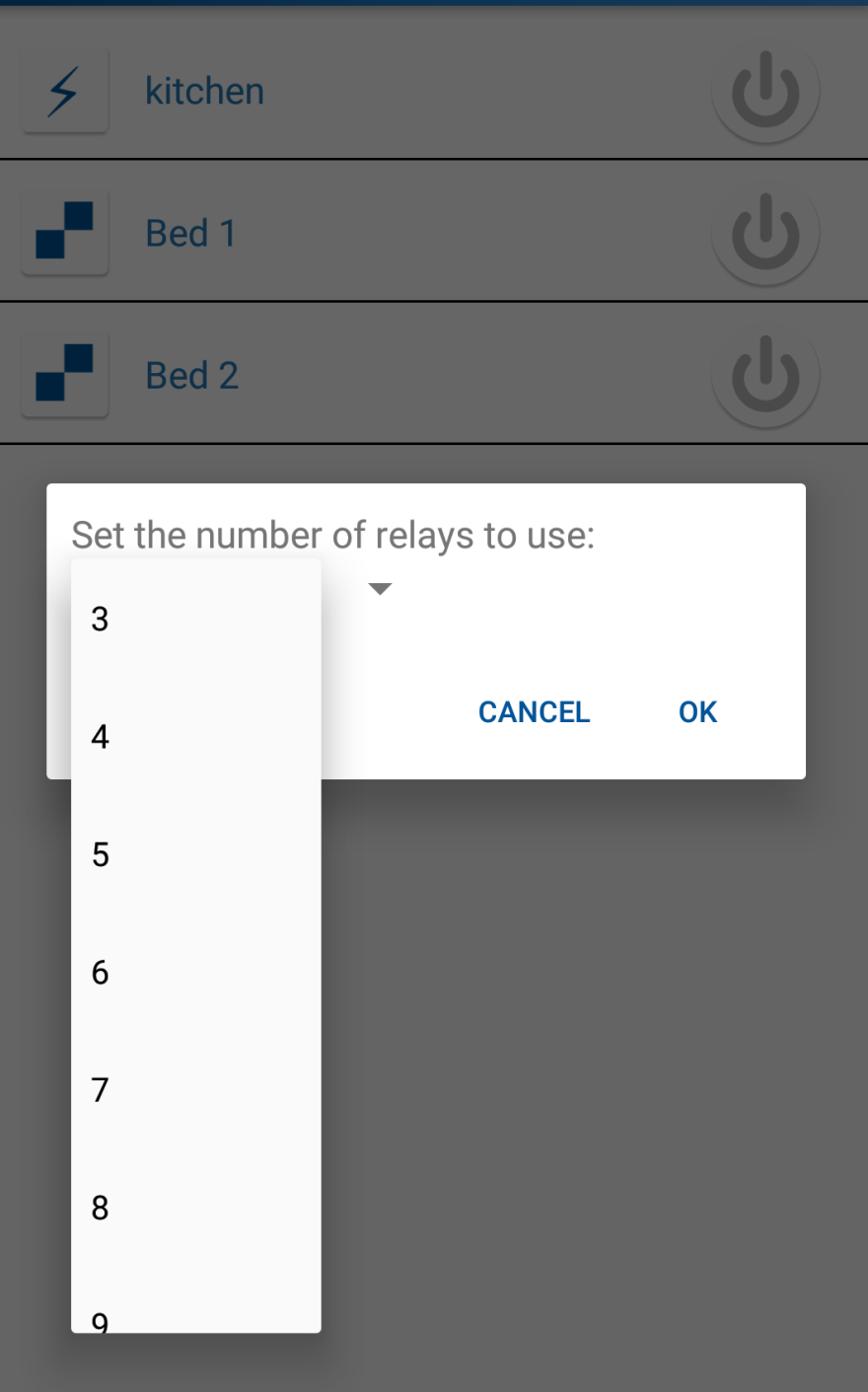
****

FIGURE 10.1.4 CHANGING RELAY

This page is used to control the number of relays for the application

**10.1.5 SETTING UP RELAY**

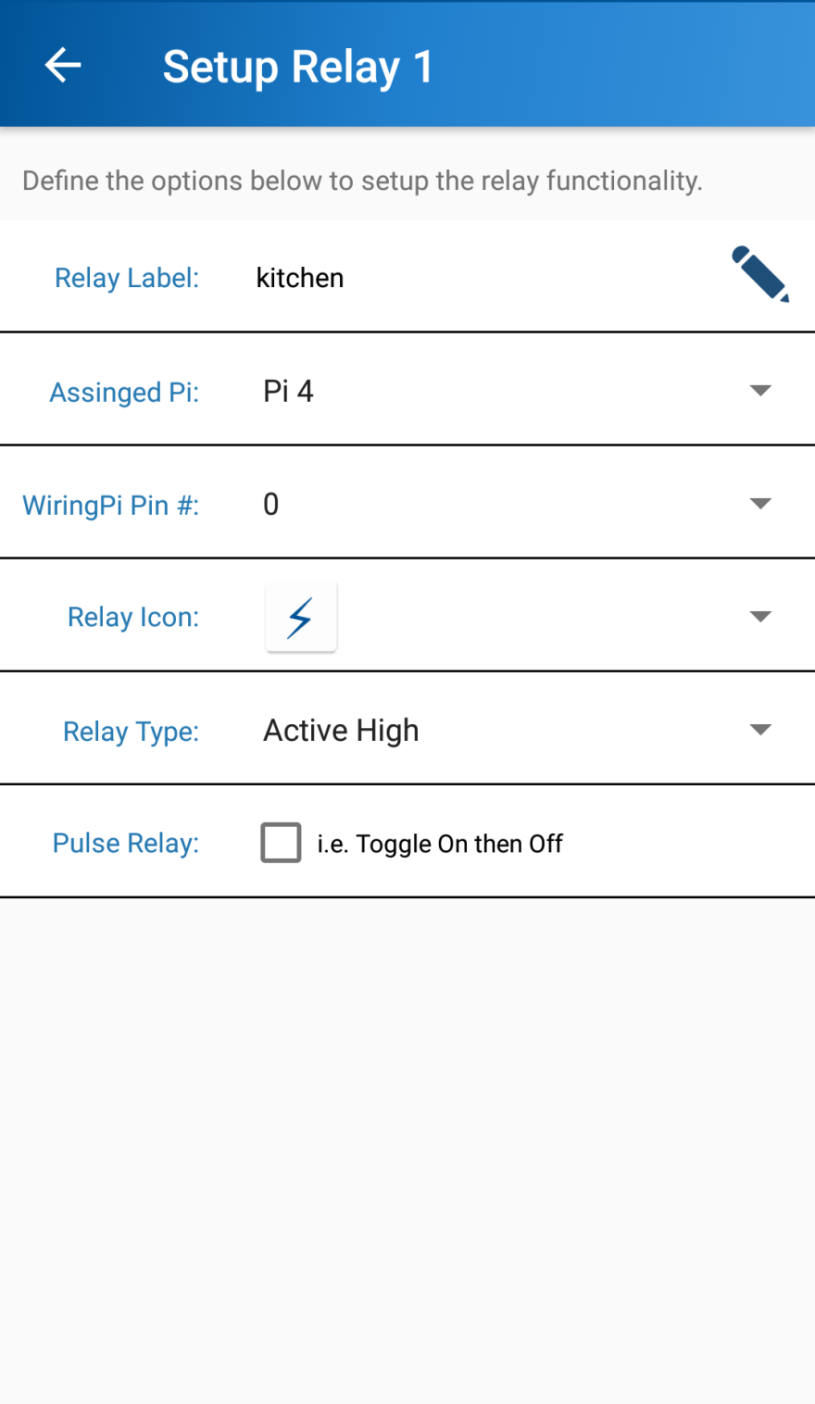
****

FIGURE 10.1.5 SETTING UP RELAY

This page is used to control the relay label and the pi assigned pi with wriring pi and the relay icon

**10.1.6 PIN MODEL**

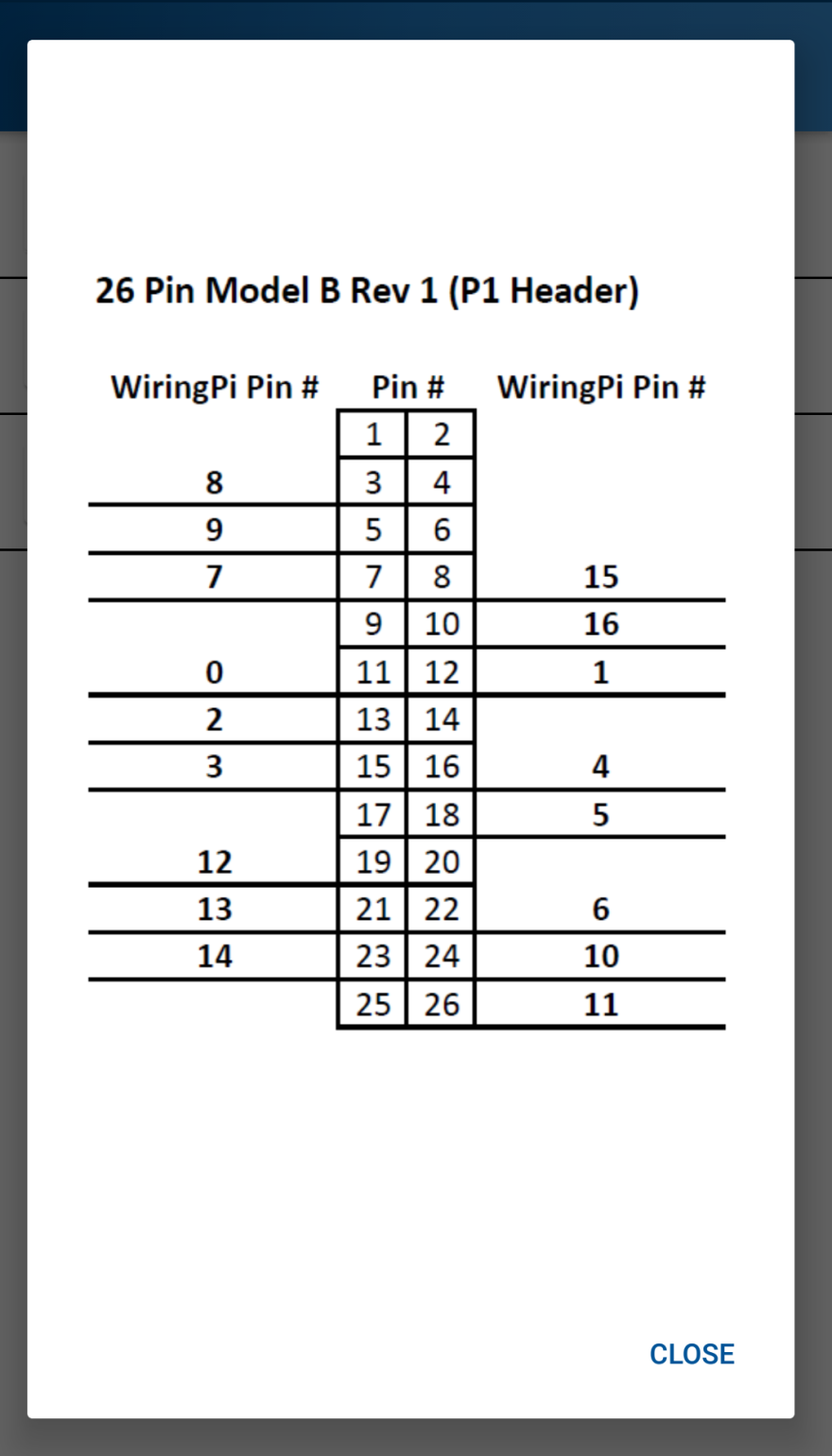
****

FIGURE 10.1.6 PIN MODEL

**10.1.7 HOME**

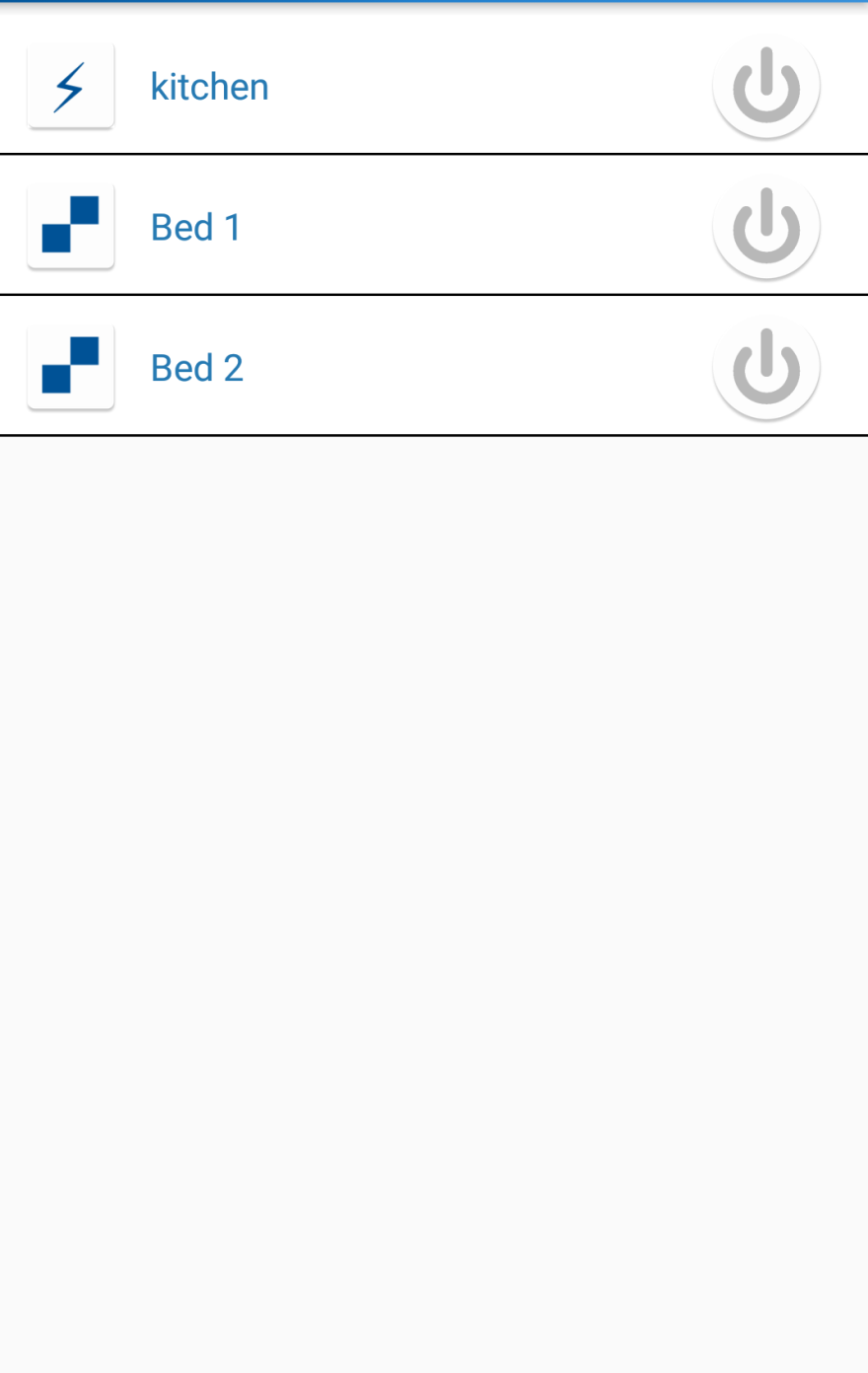
****

FIGURE 10.1.7 HOME

**10.2 SAMPLE CODING**

**//Getting started.java**

package com.tech.android.raspberryhomeautomation;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.webkit.WebResourceRequest;

import android.webkit.WebView;

import android.webkit.WebViewClient;

import android.widget.Button;

import android.widget.Toast;

public class GettingStarted extends AppCompatActivity {

private Button button , button1;

private WebView webView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_getting\_started);

webView=(WebView)findViewById(R.id.webview);

webView.setWebViewClient(new Customclient());

button=(Button)findViewById(R.id.buttonload);

button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

webView.loadUrl("http://192.168.43.252/");

}

});

button1=(Button)findViewById(R.id.buttonloadoff);

button1.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

webView.loadUrl("http://192.168.43.252/off.php");

}

});

}

public class Customclient extends WebViewClient

{

@Override

public boolean shouldOverrideUrlLoading(WebView view, String url) {

view.loadUrl(url);

return true;

}

}

}

**//Relaycontroller.java**

package com.tech.android.raspberryhomeautomation;

import android.content.Context;

import android.content.Intent;

import android.content.SharedPreferences;

import android.os.Bundle;

import android.support.design.widget.FloatingActionButton;

import android.support.design.widget.Snackbar;

import android.view.View;

import android.support.design.widget.NavigationView;

import android.support.v4.view.GravityCompat;

import android.support.v4.widget.DrawerLayout;

import android.support.v7.app.ActionBarDrawerToggle;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.Toolbar;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Button;

import android.widget.Toast;

public class RelayController extends AppCompatActivity

implements NavigationView.OnNavigationItemSelectedListener {

@Override

protected void onCreate(final Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_relay\_controller);

Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);

setSupportActionBar(toolbar);

DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer\_layout);

ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(

this, drawer, toolbar, R.string.navigation\_drawer\_open, R.string.navigation\_drawer\_close);

drawer.addDrawerListener(toggle);

toggle.syncState();

NavigationView navigationView = (NavigationView) findViewById(R.id.nav\_view);

navigationView.setNavigationItemSelectedListener(this);

}

@Override

public void onBackPressed() {

DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer\_layout);

if (drawer.isDrawerOpen(GravityCompat.START)) {

drawer.closeDrawer(GravityCompat.START);

} else {

super.onBackPressed();

}

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.relay\_controller, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

@SuppressWarnings("StatementWithEmptyBody")

@Override

public boolean onNavigationItemSelected(MenuItem item) {

// Handle navigation view item clicks here.

int id = item.getItemId();

if (id == R.id.nav\_address) {

// Handle the camera action

Intent intent=new Intent(RelayController.this,PiUrl.class);

startActivity(intent);

} else if (id == R.id.nav\_number) {

} else if (id == R.id.nav\_setup) {

} else if (id == R.id.nav\_help1) {

Intent intent=new Intent(RelayController.this,GettingStarted.class);

startActivity(intent);

} else if (id == R.id.nav\_diagram1) {

} else if (id == R.id.nav\_diagram2) {

}

DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer\_layout);

drawer.closeDrawer(GravityCompat.START);

return true;

}

}

**//activity\_getting\_started.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_margin="5dp"

android:orientation="vertical"

android:layout\_height="match\_parent"

tools:context="com.tech.android.raspberryhomeautomation.GettingStarted">

<WebView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/webview"/>

<Button

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/buttonload"

android:text="Turn Relay On"/>

<Button

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/buttonloadoff"

android:text="Turn Relay Off"/>

</LinearLayout>

**//activity\_pi\_url.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_margin="10dp"

android:orientation="vertical"

tools:context="com.tech.android.raspberryhomeautomation.PiUrl">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Set the url of up to 4 Raspberry pi's"

android:textSize="16dp"

android:textStyle="italic" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Set the address for Respective Raspberry pi's"

android:textSize="16dp"

android:textStyle="italic" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:weightSum="2">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="0.5"

android:text="Pi - 1 "

android:textColor="@color/colorPrimary" />

<EditText

android:id="@+id/eturl1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:ems="10"

android:hint="Enter the URL 1"

android:inputType="textPersonName" />

<Button

android:id="@+id/btnsave1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="0.5"

android:background="@android:color/transparent"

android:text="Save"

android:textColor="@color/colorPrimary" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:weightSum="2">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="0.5"

android:text="Pi - 2 "

android:textColor="@color/colorPrimary" />

<EditText

android:id="@+id/eturl2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:ems="10"

android:hint="Enter the URL 2"

android:inputType="textPersonName" />

<Button

android:id="@+id/btnsave2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="0.5"

android:background="@android:color/transparent"

android:text="Save"

android:textColor="@color/colorPrimary" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:weightSum="2">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="0.5"

android:text="Pi - 3 "

android:textColor="@color/colorPrimary" />

<EditText

android:id="@+id/eturl3"

android:layout\_width="wrap\_content"

android:inputType="textPersonName" />

<Button

android:id="@+id/btnsave3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="0.5"

android:background="@android:color/transparent"

android:text="Save"

android:textColor="@color/colorPrimary" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:weightSum="2">

<EditText

android:id="@+id/eturl4"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:ems="10"

android:hint="Enter the URL 4"

android:inputType="textPersonName" />

<Button

android:id="@+id/btnsave4"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="0.5"

android:background="@android:color/transparent"

android:text="Save"

android:textColor="@color/colorPrimary" />

</LinearLayout>

**PLAGIARISM**

