Automatic Backup Wi-Fi Camera

Final Report

CSCI 6838 Research Project and Seminar
Instructor: Dr. Alfredo Pérez-Dávila

Shivang Dave
Sumanjali Tirunagaru
Vignesh Kirubakaran
Nischal Reddy Tamma
Gowri Priya Paladugu

Automatic Backup Wi-Fi Camera

Final Report

CSCI 6838 Research Project and Seminar

Instructor: Dr. Alfredo Pérez-Dávila

Team

Shivang Dave

Sumanjali Tirunagaru

Vignesh Kirubakaran

Nischal Reddy Tamma

Gowri Priya Paladugu

<u>Table of Contents</u>

Abstract

The primary objective and vision behind this project is to enable users to make use of new technologies in pocket friendly manner by offering cheap and easy to setup alternatives. Our goal for this project is to come up with an easy to implement open source system that allows end-user to connect to camera wirelessly and stream its video feed on their iOS devices. An obvious application for this system is an Automatic Backup Wi-Fi Camera. The targeted audience for this system in consumer market is people whose car's do not have inbuilt rear camera for visual assistance. Along with backend, we are planning to develop and release an iOS application with a very elegant and user-friendly interface to compliment the system.

With this system, the user will be able to take advantage of an external camera for visual assistance while reversing their car. We are using combination of an inexpensive system board and sensors which are easy to assemble and allows easy replacement if case of damage. Major focus of this project is on making the user experience seamless. The iOS application's interface allows users to setup the system without much hassle. Once the initial setup is done, system becomes autonomous. We are integrating manual overriding options inside the application so that user always has control. After becoming autonomous, if the system detects a change of transmission it will send a remote notification to the linked iOS device. Once user clicks on the notification, they will be able to stream the video feed directly on their iOS device. Therefore, user doesn't need to buy a screen for the system separately which saves them a bunch of money and keeps overall cost of the system down.

Since the project is based on IoT, it makes it and efficient. The metrics such as performance, complexity, security, stability and latency will be used in this project. We will measure and test the whole system for reliability and robustness making the product deployable. With minor modifications, system can be used for various security solutions as a surveillance system. Thus, this system is extremely scalable.

I. Problem Statement

The primary objective of this project is to develop a Mobile Application which enables end-user to establish a connection between RPi camera and an iOS device. The basic idea behind this implementation is to detect change of transmission (i.e. reverse gear etc.) using an IR beam break sensor which triggers camera linked with a raspberry pi, once the gear is put in reverse mode. This implementation is built on top of concepts of Internet of Things. There are many high-end cars that give the functionality of a screen with rear camera. It may not be always possible that all the customers will have this functionality which may not be affordable. We are trying to formulate a solution in which Mobile screen will act as the screen that displays the video from the rear camera. Another cost-efficient solution for the rear camera is Raspberry pi camera. All this works out to be cheaper and more efficient than the existing solutions.

A. Project Background

Internet-Of-Things is the Connection of various hardware devices where each device is embedded with electronics, software and connectivity which enables the hardware devices to connect and communicate in the Internet. Internet-Of-Things is one of the most trending technology at present. This technology allows to improve the performance, accuracy, Economic benefit and reduce human intervention. It is estimated that 30 billion objects will be using this technology by 2020.

B. Stakeholders

Stakeholders are those who can positively or negatively impact the output of the projects. It is very important to identify the names of stakeholders during the initiation stage of the projects. Few examples of stakeholders can be the customers, the clients, the project team members, sellers, buyers, sponsors etc. It is mandatory to identify the stakeholders and manage their expectations throughout the life-cycle of the project.

Stakeholders for this project, would be all the team members who are responsible to design the application, implement and produce a prototype. And other teams, professor and prospective end users can perform roles of External Stakeholders. They can give suggestions and provide their expectation for this project.

C. Users

As described earlier, the end user for this project are those who wants to setup this system in a car without with inbuilt rear camera. This system has potential to be used for various security solutions with minor modifications. Hence, users for this system depends on how the system is used.

D. Risks

Risks involved are as following:

- Server on Raspberry Pi might crash or stop working or may become faulty.
- System is prone to catch dust particles and perform poorly under unclean environment.
- Latency may not stay consistent depending on Wireless signal strength.

E. Assumptions

Assumptions made for developing a functional prototype are as following:

- Notification always arrives.
- Internet access is always available.
- Raspberry pi is functioning over an independent power supply.

Automatic Backup Wi-Fi Camera

Project Vision, Scope & Plan

I. Project Background

Internet-Of-Things is the Connection of various hardware devices where each device is embedded with electronics, software and connectivity which enables the hardware devices to connect and communicate in the Internet. Internet-Of-Things is one of the most trending technology at present. This technology allows to improve the performance, accuracy, Economic benefit and reduce human intervention. It is estimated that 30 billion objects will be using this technology by 2020.

A. Stakeholders

Stakeholders are those who can positively or negatively impact the output of the projects. It is very important to identify the names of stakeholders during the initiation stage of the projects. Few examples of stakeholders can be the customers, the clients, the project team members, sellers, buyers, sponsors etc. It is mandatory to identify the stakeholders and manage their expectations throughout the life-cycle of the project.

Stakeholders for this project, would be all the team members who are responsible to design the application, implement and produce a prototype. And other teams, professor and prospective end users can perform roles of External Stakeholders. They can give suggestions and provide their expectation for this project.

B. Users

As described earlier, the end user for this project are those who wants to setup this system in a car without with inbuilt rear camera. This system has potential to be used for various security solutions with minor modifications. Hence, users for this system depends on how the system is used.

C. Risks

Risks involved are as following:

- Server on Raspberry Pi might crash or stop working or may become faulty.
- System is prone to catch dust particles and perform poorly under unclean environment.
- Latency may not stay consistent depending on Wireless signal strength.

D. Assumptions

Assumptions made for developing a functional prototype are as following:

- Notification always arrives.
- Internet access is always available.
- Raspberry pi is functioning over an independent power supply.

II. Vision

The primary objective and vision behind this project is to enable users to make use of new technologies in pocket friendly manner by offering cheap and easy to setup alternatives. Our goal for this project is to come up with an easy to implement open source system that allows end-user to connect to camera wirelessly and stream its video feed on their iOS devices. An obvious application for this system is an Automatic Backup Wi-Fi Camera. The targeted audience for this system in consumer market is people whose car's do not have inbuilt rear camera for visual assistance. Along with backend, we are planning to develop and release an iOS application with a very elegant and user-friendly interface to compliment the system.

With this system, the user will be able to take advantage of an external camera for visual assistance while reversing their car. We are using combination of an inexpensive system board and sensors which are easy to assemble and allows easy replacement if case of damage. Major focus of this project is on making the user experience seamless. The iOS application's interface allows users to setup the system without much hassle. Once the initial setup is done, system becomes autonomous. We are integrating manual overriding options inside the application so that user always has control. After becoming autonomous, if the system detects a change of transmission it will send a remote notification to the linked iOS device. Once user clicks on the notification, they will be able to stream the video feed directly on their iOS device. Therefore, user doesn't need to buy a screen for the system separately which saves them a bunch of money and keeps overall cost of the system down.

Since the project is based on IoT, it makes it and efficient. The metrics such as performance, complexity, security, stability and latency will be used in this project. We will measure and test the whole system for reliability and robustness making the product deployable. With minor modifications, system can be used for various security solutions as a surveillance system. Thus, this system is extremely scalable.

A. List of Features

- Easy to assemble and setup
- Backend is based on Node JS. Hence, stream can be accessed via browser
- Elegant User Interface using Material Design Guidelines

- Wireless connection to the camera feed
- Remote notification
- Control over acceptance or rejection of live stream request
- Manually triggering video stream incase notification fails to arrive
- Compatibility: iOS 11+
- Compatibility with wide range of iOS devices: iPhone, iPod & iPad

B. Add-on Features (Future Updates)

- Collision detection alerts using ultrasonic sensor
- Smart directions using machine learning libraries

III. Project Plan

A. Milestones

This section highlights all of the milestones for successful implementation.

Milestone (s)	Software Process Activity
User Requirements / SRS Document	Requirement Collection
Project Schedule	Planning
System Architecture	Data Flow Analysis
Prototype	Frontend & Backend Designing
Integration	Implementation
Validation & Error Reporting	Testing

B. Statement of Work

Work Product (s)	Team Member who work on it
Project Documents	Shivang Dave Sumanjali Tirunagaru Vignesh Kirubakaran Nischal Reddy Tamma Gowri Priya Paladugu
System Architecture and Flow Designing	Shivang Dave Sumanjali Tirunagaru
Hardware setup and configurations	Shivang Dave Sumanjali Tirunagaru Nischal Reddy Tamma
iOS Application Design & Development (Front end)	Shivang Dave
Server setup and APIs (Back end)	Shivang Dave Sumanjali Tirunagaru Vignesh Kirubakaran Gowri Priya Paladugu
Integration	Gowri Priya Paladugu Nischal Reddy Tamma
Validation & Error Reporting	Sumanjali Tirunagaru Vignesh Kirubakaran

C. Risk Plan

Risk(s)	Probable Solution(s)
Server on Raspberry Pi might crash or stop working or may become faulty.	Programming a fail-safe inside the server.
System is prone to catch dust particles and perform poorly under unclean environment.	Covering essential server parts before deployment.
Latency may not stay consistent depending on Wireless signal strength.	Reducing network load by compressing video feed packets.
Sensor may stop working.	Replacing the sensor should solve the issue.

D. Resource List

Resource (s)	Availability
Raspberry Pi 3B module	Always
RPi Camera Module	Always
IR Break Beam Sensors	Always
IOS mobile	Always
Windows/MacOS to develop scripts	Always

Automatic Backup Wi-Fi Camera

Software Requirements Specification

I. Introduction

The primary objective of this project is to develop a Mobile Application which enables end-user to establish a connection between RPi camera and an iOS device. The basic idea behind this implementation is to detect change of transmission (i.e. reverse gear etc.) using an IR beam break sensor which triggers camera linked with a raspberry board, once the gear is put in reverse mode. It then sends a notification to user's iOS and lets them livestream the camera feed. Once the gear is moved from reverse mode to any other mode, IR sensor detects this change and stops the stream automatically. This implementation is built on top of concepts of Internet of Things.

II. Background

Internet-Of-Things is the Connection of various hardware devices where each device is embedded with electronics, software and connectivity which enables the hardware devices to connect and communicate in the Internet. Internet-Of-Things is one of the most trending technology at present. This technology allows to improve the performance, accuracy, Economic benefit and reduce human intervention. It is estimated that 30 billion objects will be using this technology by 2020.

A. Problem Statement

There are many high-end cars that give the functionality of a screen with rear camera. It may not be always possible that all the customers will have this functionality which may not be affordable. We are trying to formulate a solution in which Mobile screen will act as the screen that displays the video from the rear camera. Another cost-efficient solution for the rear camera is Raspberry pi camera. All this works out to be cheaper and more efficient than the existing solutions.

III. Proposed System

The proposed system will make use of minicomputer Raspberry Pi which establishes its own network to communicate with the user's mobile device. This system will help to stream the video from the RPi camera to an iOS application. The application accesses the video stream via network established by Raspberry Pi. Also, user can manually toggle the camera feed. But the actual video stream can only be triggered with the help of IR Circuit breaker. Whenever the IR beam is broken, it will notify the server to start camera feed and will send a remote notification to the user's iOS device. Once the user confirms the stream, raspberry pi will start encoding the camera feed. And the encoded camera feed will be hosted on local network hence the iOS device will be able to stream it in real time.

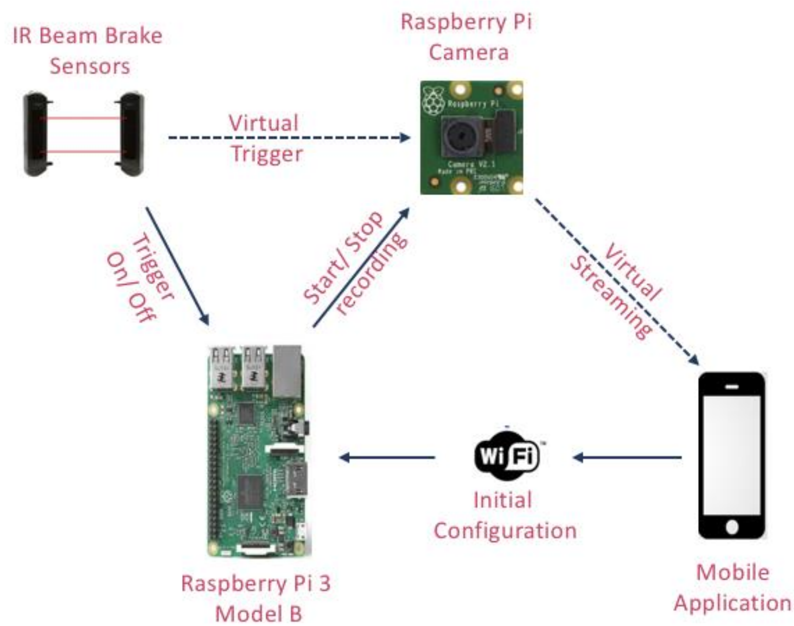


Fig. Proposed System Architecture

A. Technical Constraints

Application is required to be connected to the same network as the server in order to stream the video feed. If the connection is lost during an active session, the stream will still stay active until sensor stops it but user won't be able to join the session from midway. This issue is currently being analyzed and will be solved on before final release. In order to restore the functionality, user is required to relaunch the application and re-trigger the sensor which will restart the stream.

B. Design Constraints

Over the years, software development has gone through many changes. There are various development patterns that will enable companies to come up with quality designs and software's at a rapid pace. MVC architecture pattern allows designers and developers to work on different modules of a system at the same time. It also contributes indirectly towards fulfilling non-functional requirements such as security, scalability etc. by keeping design, logic and controllers separate from each other.

Also, build the application interface using Material Design: Google's mature, constantly developed visual language with comprehensive documentation and pattern library. The principles of Material Design include issues from typography to iconography, layouts to onboarding, and the presentation of the new features in an application. Consistent use of Material Design significantly reduces the risk of a bad user interface. As *'Consistency is difficult but essential'*.

C. Use Case(s)

Figure shown below highlights primary use cases of Proposed System.

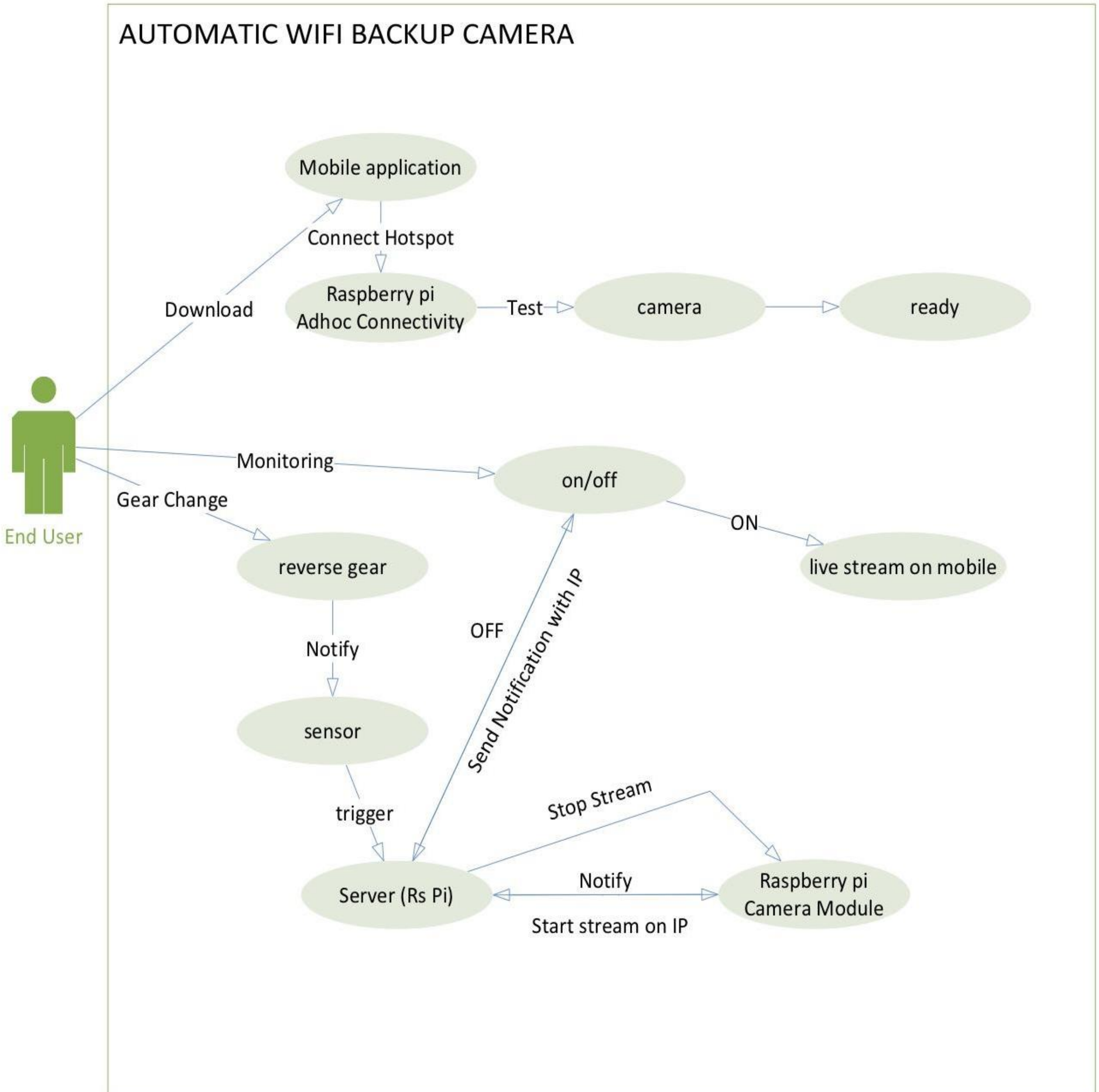


Fig. Use Case(s) for Proposed System

IV. Requirements

- ❑ Raspberry Pi 3 - Model B should be used to host networking, main server and sensors.

- ❑ To design and develop an iOS Application with a mature and simplistic UI.

- ❑ To use Wi-Fi for communication between Hardware components and iOS application.

- ❑ To develop a communication protocol between the User's Mobile and Rpi camera via the mobile application.

- ❑ The application should keep track of the changes in the sensor and notify the user for live stream with minimal delay.

A. System Requirements

- ❑ *Platform:*
  - ❑ Client – iOS
  - ❑ Server – Raspbian OS

- ❑ *Modules:*
  - ❑ RPi Camera Module
  - ❑ IR Break Beam Sensor

- ❑ *Programming Languages:*
  - ❑ Client - Swift 4.0
  - ❑ Server - Node JS
  - ❑ Scripts - Python, Bash

- ❑ *Source Control:*
  - ❑ Github

- ❑ *Streaming Protocol / Framework and their Latency over Wi-Fi:*
  - ❑ RTSP ~ 10s
  - ❑ RTMP ~ 7s
  - ❑ WebRTC ~ 3s
  - ❑ OpenCV + Flask (Frameworks) ~ 1s

B. Functional Requirements

This section highlights functional requirements for Proposed System.

Req. #	Description	Priority
FR1	User should be able to download and install the application.	High
FR2	User can have an option to reject the streaming when notified.	High
FR3	User can have an option to stop the streaming.	High
FR4	User should be able to connect to the network securely through the Application without leaving it.	High
FR5	When the reverse gear is set (IR sensor breaks), then Camera should start encoding and server should start streaming.	High
FR6	Users should get a real time notification when the live stream is launched.	High
FR7	Support for cross-platform functionality.	Low

C. Non-Functional Requirements

This section highlights non-functional requirements for Proposed System.

#	Requirement	Description	Priority
NF1	Security	- System should be immune from any kind of memory leak and unauthorized access. - Access to the internet has been limited for notification service making it immune for any malware infection and DDoS attacks.	High
NF2	Reliability & Usability	- System is required to perform under wide range of undesirable operational conditions such as extreme climate conditions, sensor malfunction, internet access taken down etc. - System is required to livestream the camera feed with least possible latency.	High

NF3	Scalability	- System architecture should be scalable enough to add additional features by integrating more sensors and application should be able to adapt to those changes.	Medium

V. Milestones

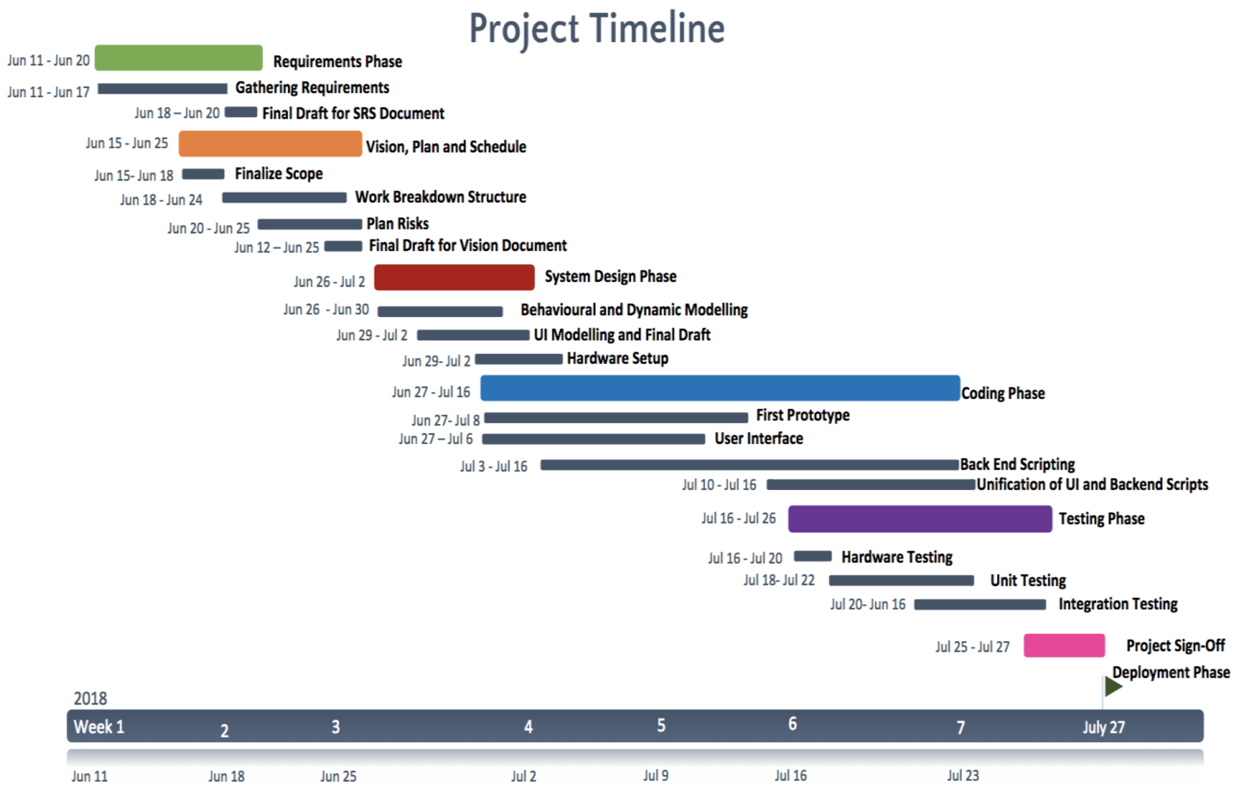
This section highlights all of the milestones for successful implementation.

Milestone (s)	Software Process Activity
User Requirements / SRS Document	Requirement Collection
Project Schedule	Planning
System Architecture	Data Flow Analysis
Prototype	Frontend & Backend Designing
Integration	Implementation
Validation & Error Reporting	Testing

Automatic Backup Wi-Fi Camera

Project Timeline/Schedule

I. Project Schedule *(estimated)*



Task	Approx. Start Date	Approx. End Date
Requirements Phase	**06/11/2018**	**06/20/2018**
Preliminary Information gathering & analyzing basic requirements	06/11/2018	06/17/2018
Final Draft for Software Requirement Specification document.	06/18/2018	06/20/2018
Vision, Plan, Schedule	**06/15/2018**	**06/25/2018**
Finalize Scope	06/15/2018	06/18/2018
Work breakdown structure	06/18/2018	06/24/2018

Plan Risks	06/20/2018	06/25/2018
Final Draft for Vision Document	06/12/2018	06/25/2018
Design Phase	**06/26/2018**	**07/2/2018**
Generating UML diagram(s) for various use cases & feasibility study for proposed architecture.	06/26/2018	06/30/2018
System Architecture & UI Designing	06/26/2018	07/2/2018
Hardware Setup	06/29/2018	07/2/2018
Final Draft for Design Document	07/1/2018	07/2/2018
Coding Phase	**06/27/2018**	**07/16/2018**
First Prototype	06/27/2018	07/08/2018
User Interface Scripts	06/27/2018	07/06/2018
Backend Scripting	07/03/2018	07/16/2018
Unification of Frontend & Backend	07/10/2018	07/16/2018
Testing & Maintenance	**07/16/2018**	**07/26/2018**
Hardware Testing	07/16/2018	07/20/2018
Unit Testing	07/18/2018	07/22/2018
Integration Testing	07/20/2018	07/16/2018
Deployment	**07/25/2018**	**07/27/2018**

Automatic Backup Wi-Fi Camera

System Design

I. Design Analysis

Design Analysis for the project includes three modeling schematics to describe the functionality and layout of the entire product. This project includes multiple components such as iOS Mobile Application and the Wi-Fi Module connected Camera and sensors. These different components work individually but function in complete synergy to let the user control and use the product with ease.

A. Behavioral Modeling

User interaction with the system and how the system responds to the same is recorded in this. Use cases defining each action and what it includes or extends to is seen here. Each use case describes interaction, functionality and response.

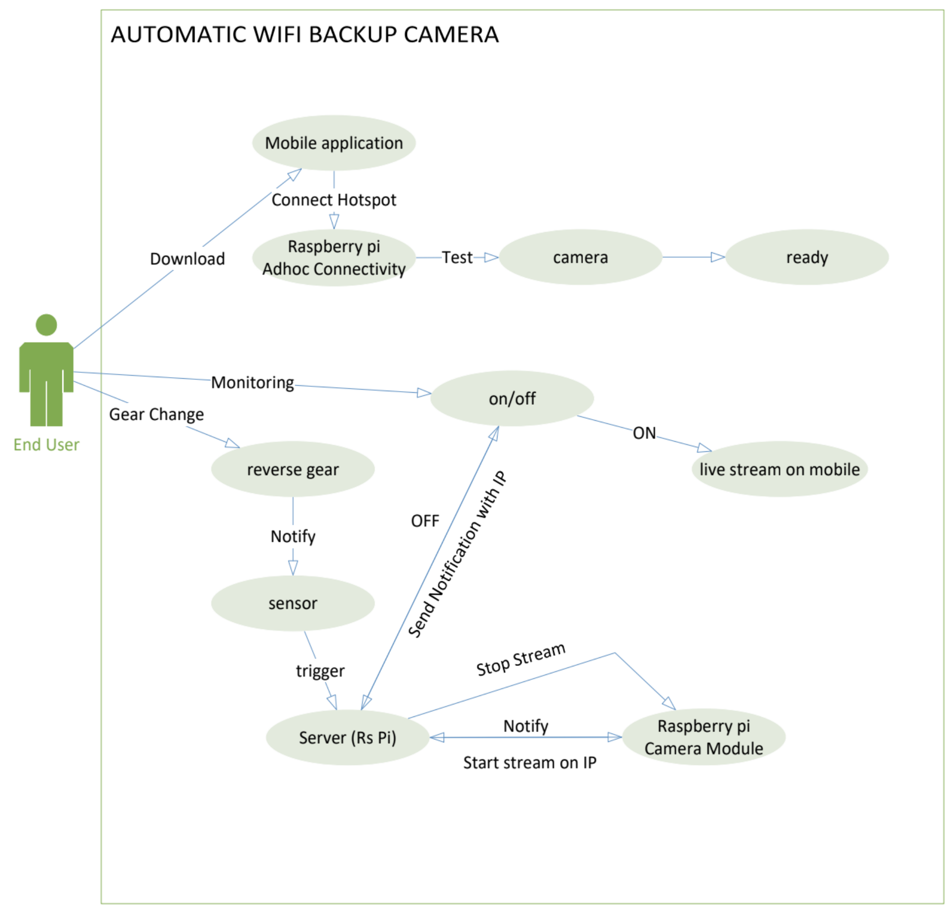


Fig: Use case Diagram for Proposed System

B. Structural Modeling

It’s a design framework where the components, attributes and the relationships between them are identified and expressed. It is the main building block of object oriented modelling. It is used for general conceptual modelling of the systematic of the application and for detailed modelling translating the models into programming code.

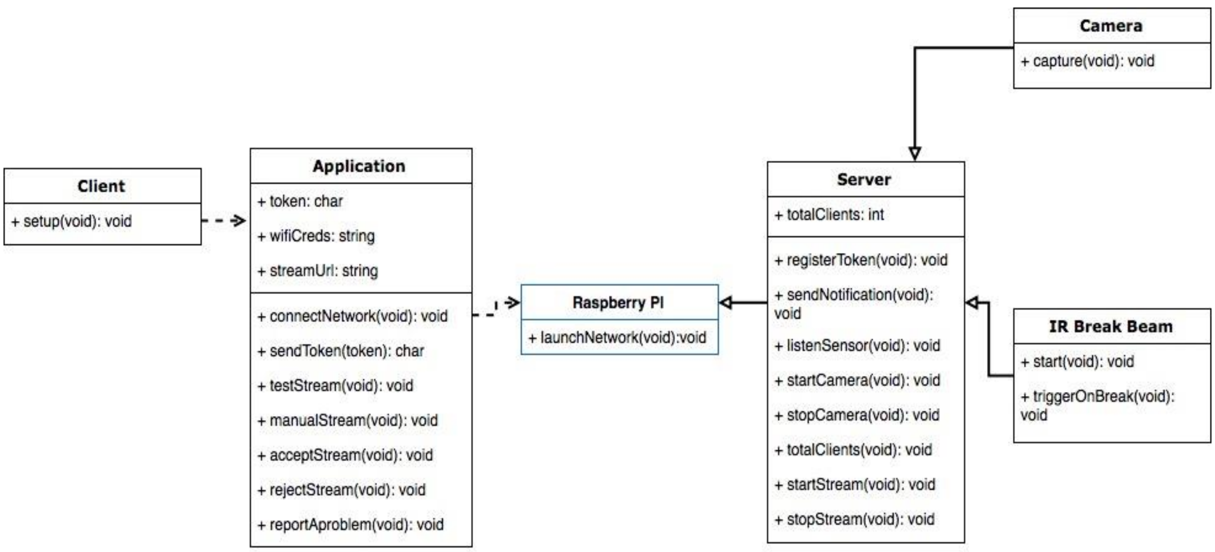


Fig: Class Diagram for Proposed System

The classes above represent both the main elements, interactions in the application and the classes to be programmed for the system to work correctly. We have identified total of 5 classes that we think are necessary in order for the system to function. As the above figure shows, we are taking centralized approach when it comes to communication between different components. Our server will take requests and serve to each component over local network. What makes it more interesting is that it creates its own wireless network and sets up itself as an access point. This not only makes communication easier but it also reduces latency by a big margin.

Another huge advantage of centralized approach is that server acts as a master in master-slave configuration. All the other components are always in control of the server so there is only a small chance of malfunction because of a component. Application class represents basic structure of the iOS application and how it will be programmed to handle necessary functions. Server class inherits important methods from classes of connected components. This allows server to observe and keep the system running. Once it observes some activity, it will send a remote notification to client’s iOS device.

C. Dynamic Modeling

Behavior of the system which recurs over a period of time is defined in this modeling. Here the Sequence Diagram describes the action the system takes over time. When the server is up and starts running, user connects to its Wi-Fi (ad hoc network). Then the system carries out a series of action in an order and is described in the figure below. The initial configuration of the Wi-Fi module with mobile applications is crux of the process. User accepts or rejects the video, which will have different actions based on the decision.

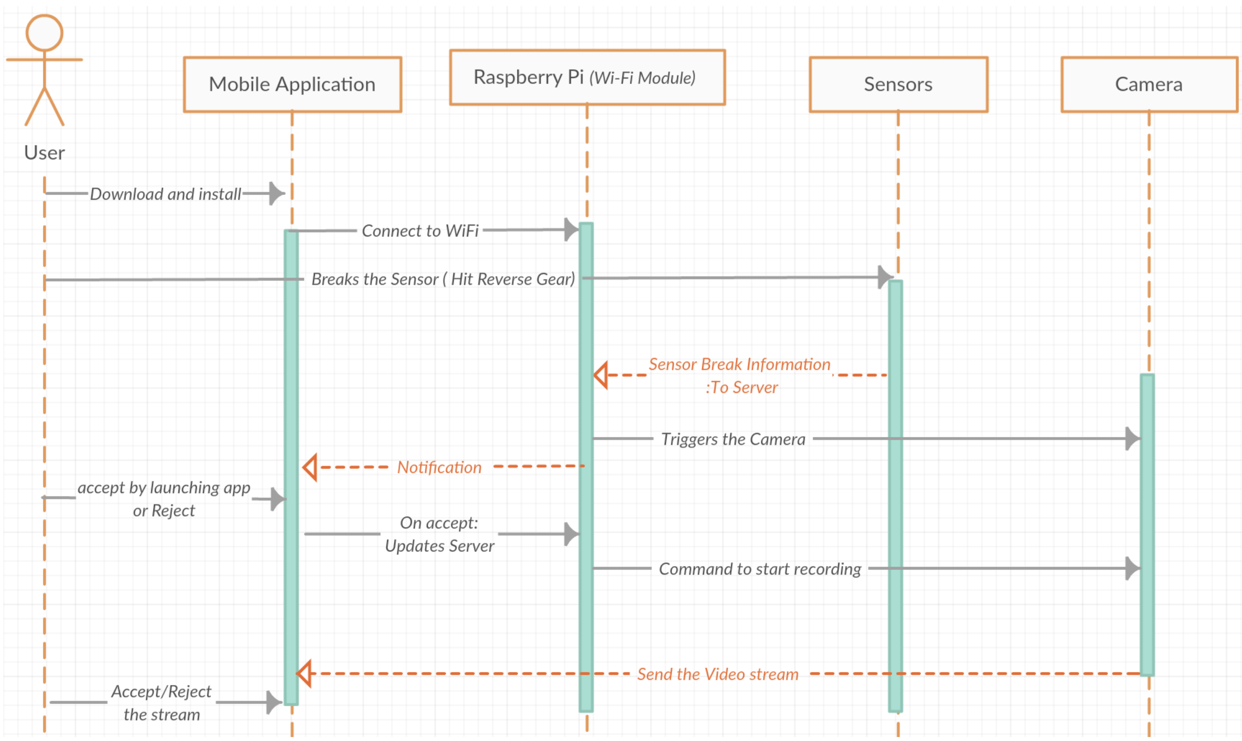


Fig: Sequence Diagram for Proposed System

II. Architecture

A. System Architecture

The architecture shows the building blocks of the whole systems of which Raspberry Pi 3B plays a key role, this is the processing side of the camera that need to be controlled. Controlling the Raspberry Pi ensures the functionality of controlling the Camera. Hence a User interface is required to interact with the device. All the other components are expected to communicate with this interface. Mobile application which is installed on an IOS mobile device communicates with the server on this module. Raspberry Pi can be mounted to any device to enable Wi-Fi activity. Here we program it to control the sensors and camera.

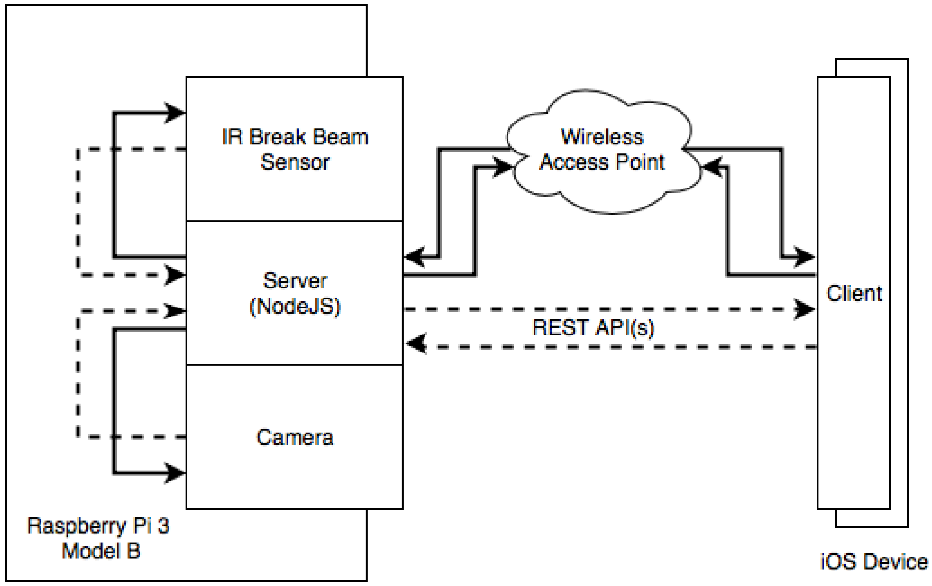
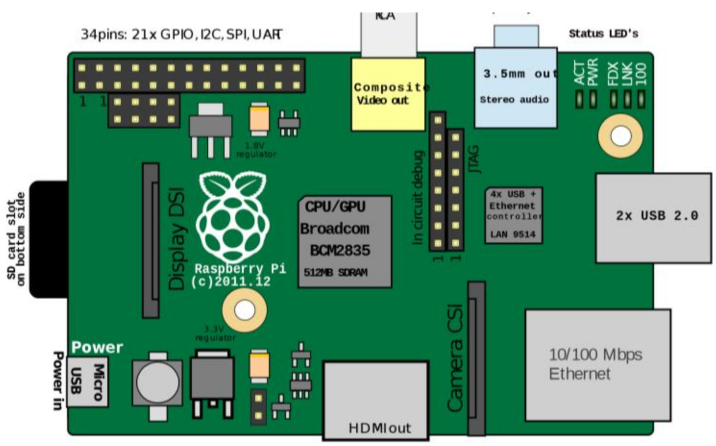


Fig: System Architecture

B. Raspberry Pi 3B Architecture

The Raspberry Pi 3 Model B is the third generation Raspberry Pi. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi Model B+ and Raspberry Pi 2 Model B.



Raspberry Pi 3 GPIO Header

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I²C)	DC Power 5v	04
05	GPIO03 (SCL1 , I²C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)	(I²C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Rev. 2 29/02/2016 www.element14.com/RaspberryPi

Fig: Raspberry Pi 3 Model B

Whilst maintaining the popular board format the Raspberry Pi 3 Model B brings you a more powerful processor, 10x faster than the first generation Raspberry Pi. Additionally, it adds wireless LAN & Bluetooth connectivity making it the ideal solution for powerful connected designs. It has improved power management, with an upgraded switched power source up to 2.5 Amps, to support more powerful external USB devices.

For our implementation, we will be using one pin from each pin type from Raspberry Pi board. Along with that, we will be using Camera CSI port to connect camera module to the system. To connect IR break beam sensor we will use one GPIO and one ground pin. However, power provided by GPIO pin may not be enough. Thus, we are adding an extra 5V DC power in addition to GPIO connection.

C. User Interface

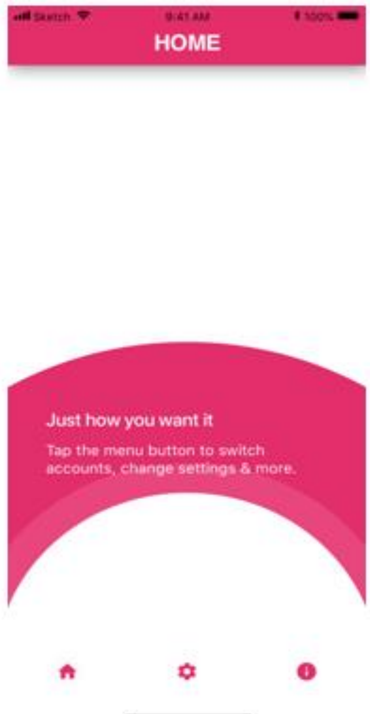
Initial UI Designs for iOS Mobile Application:

Welcome

Connect to Wi-Fi alert

Confirm Wi-Fi

Camera



Tips on Home Screen

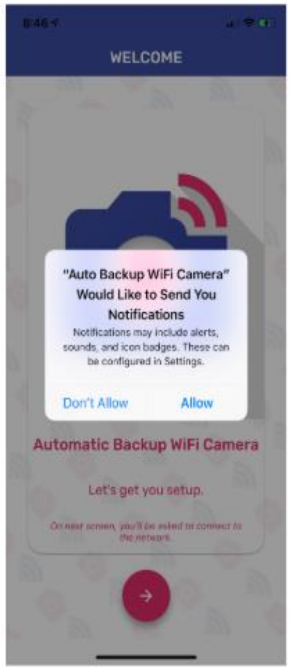
Home

Settings

About

Final UI Screenshots for iOS Mobile Application:

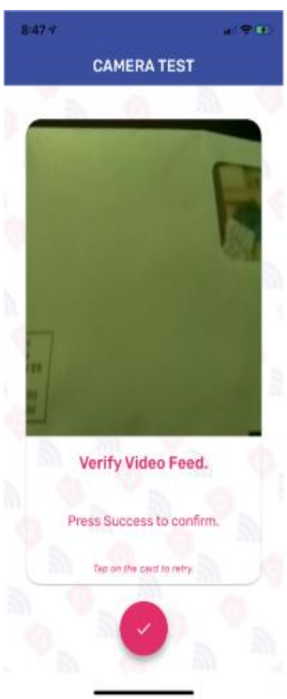
Welcome



Notification Permission

Join WiFi

Successful WiFi connection



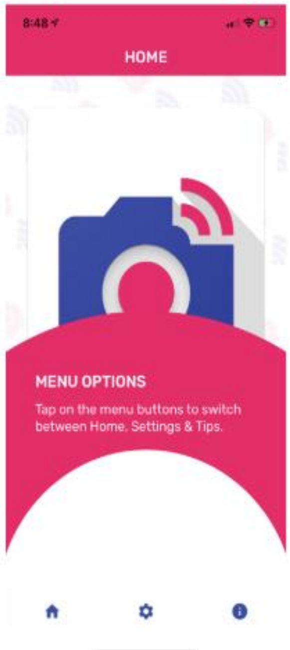
Camera Test

Home (First launch)

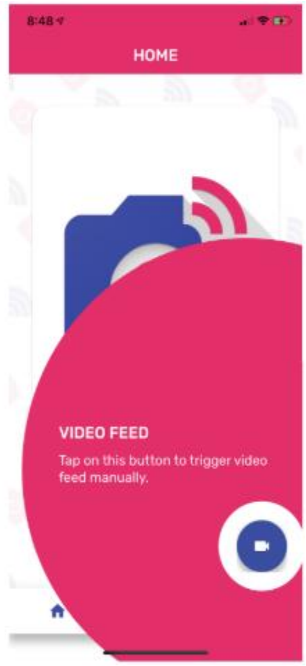
Manual Video Feed

After stream finishes

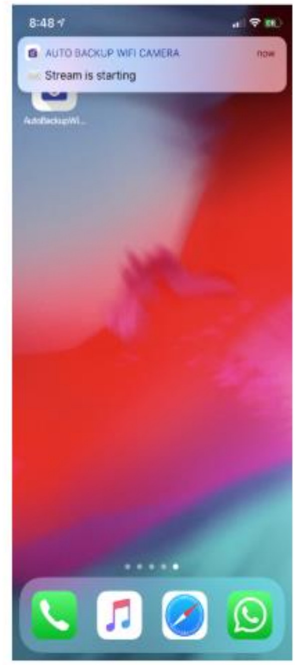
Settings



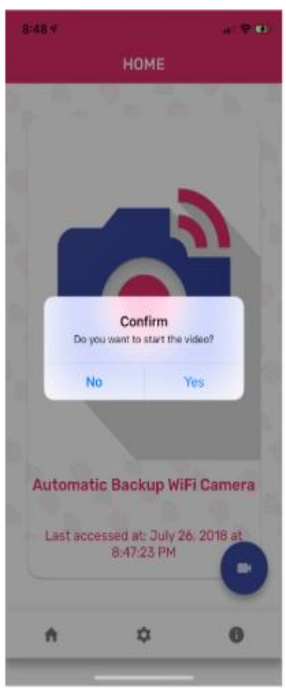
Tip for tabbar



Tip for camera button



Incoming remote notification



Tapping on notification

Tap YES

D. System Algorithm & Flow Chart

Step 1: Hardware setup & Server initialization

Step 2: User downloads and installs the application

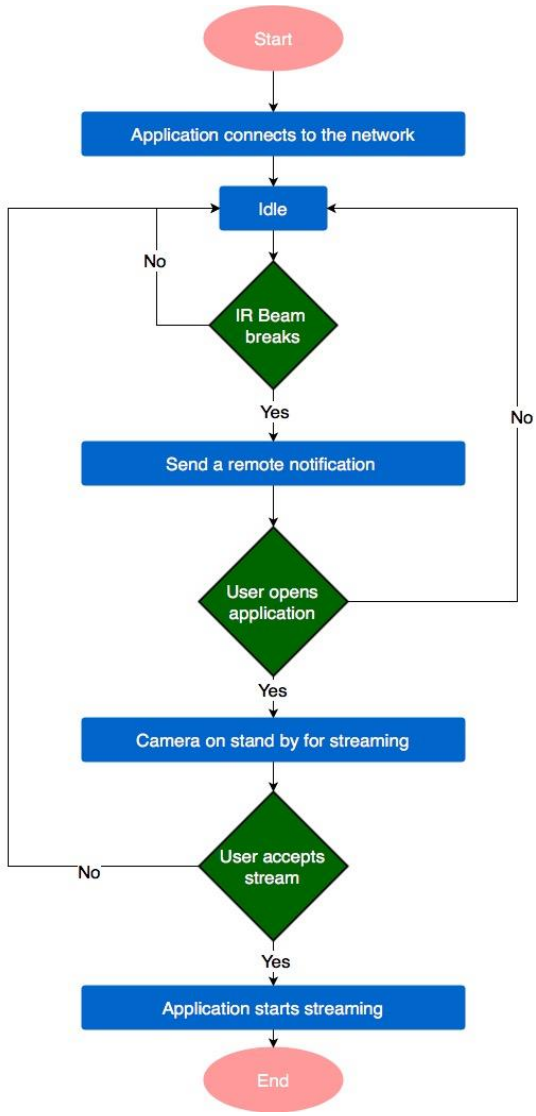


Fig: System flow chart

Step 3: Application launch and initial setup

Step 3-A: Application connects to system's network via Wi-Fi

Step 3-B: Server registers device token required for remote notification

Step 3-C: IR Break Beam sensor starts observing

Step 3-D: Camera test and confirmation

Step 3-E: At setup completion, each component except server goes in idle state

Step 4: When user hits the reverse gear & IR beam breaks, a remote notification is sent with an IP Address (*location of video stream*)

Step 5: If user launches the app from notification, then an alert is shown whether to start the live feed or not. If yes then it starts streaming live video feed

Step 6: If user rejects, changes will be discarded as false positive

Step 7: Server goes to standby and components go to idle state

Optional: *User manually triggers the live stream*

III. Design Scheme and Specifications

Material Design is a visual language by Google that synthesizes the classic principles of good design with the innovation of technology and science. It allows designers to create a single underlying system that unifies the user experience across platforms, devices and input methods. It is inspired by the physical world and its textures, including how they reflect to light and cast shadows.

We want our application's interface to be as user-friendly and as elegant as possible. Thus we have decided to use Google's material components kit to design our application. We believe that their comprehensive guides and instructions will allow us to design a faster, unique and highly responsive interface for our iOS application.

Specifications:

- **Modelling tool**: Draw.io
- **UI Design tool**: Photoshop CC 2018, Sketch
- **UI Colors**: Color Palettes Tool *by Material.io*
- **Font family**: Rubik-medium, Rubik-regular
- **Icons**: Material icons by *Material.io*
- **Animations:** Fluid & Motion UI, CoreAnimation
- **Debugger**: Fabric, Crashlytics
- **Remote Notification**: Apple Push Notification Service

IV. References

A. Raspberry Pi Documentation

B. Material.io

C. Wikipedia

D. Draw.io

E. Google Images

Automatic Backup Wi-Fi Camera

Test Plan

I. Introduction

This Test Plan document will provide information about the testing aspects of the Automatic Backup Wi-Fi Camera project. Here we will breakdown our project into its smallest components and describe all of their corresponding testing details, namely the features to be/not to be tested, test deliverables, possible risks, test schedule, item pass/fail criteria and testing approaches.

A. Testing Scope

The system testing for this project is a critical phase for ensuring our prototype has met all the critical standards. For deployment, our team has decided to focus testing in a controlled and limited environment. By testing in a mockup environment it will allow to measure efficiency in a large portion of hardware tests. To test the product indoor before setup in real world, it will be flexible and throughput when testing the Raspberry Pi module, camera and sensor devices.

II. Test Items

This section describes the steps to be taken to test this project as a whole. First, testing all the hardware components individually to ensure that each module works as intended. Then, work by testing the script modules then moving on to integration testing. Aside from the hardware section, each testing phase will be tested with the integrated components and will work on top of a reliable, connected system.

A. Hardware Tests

Test ID	Hardware	Input	Output	Test	Priority
1	Raspberry Pi 3B+	HTTP request, data from IR Beam Sensor, Data frames from RPi Camera	JSON data for the notification, html response with data frames from the camera.	Testing the functionality of the operating system to ensure that the program on the Raspberry Pi can process all communication between the Mobile Interface, sensor and camera module.	Critical

2	IR Beam Break	Commands from Raspberry Pi.	Sensor status data to Raspberry Pi.	Test that sending the sensor data to Raspberry Pi.	Critical
3	RPi Camera	Commands from Raspberry Pi.	Data frames to Raspberry Pi	Test that sending the captured data frames of the to Raspberry Pi.	Critical

B. Unit Tests

I. Server

Test ID	Modules	Input	Output	Test	Priority
1	Connection	Token from HTTP user request.	JSON response to the user with the success message	We will test that the data can be received is a token in required and valid format and sending data in JSON format.	Critical
2	Notification	Request with a Boolean flag value.	JSON response to the user with notification data.	We will test the functionality of sending notification appropriately based on the incoming flag value.	Moderate
3	Video Streaming	Boolean value from the sensor data.	Html response with contains the data frames.	We will test the functionality of feeding data frames in the form of html response based on the appropriate input request.	Critical
4	Total Number of Clients Connected	GET request from user.	JSON data with total number of clients.	We test the functionality by dummy clients connecting to server and get the value as expected	Medium

II. User Interface

Test ID	Modules	Input	Output	Test	Priority
1	Correct constraints depending on API results	Navigate through setup process	Successful Initial Setup	User will be registering device and setup process every time the application is installed.	Critical
2	One time setup	Successful Initial Setup	Home Screen on every launch	Once user finishes device registration successfully, they will be redirected to the home screen on every launch.	High
3	Stream VC	HTML response with contains the url for video stream.	Play the video from the URL	User should be able to see the video streamed by the server once user opens the notification.	Critical
4	Move to Home screen once stream finishes	Stream ending notification	Automatic navigation to Home Screen	As soon as the stream finishes, user will be popped out of player and will land on home screen.	High

C. Integration Tests

Test ID	Layer	Input	Output	Test	Priority
1	Sensor Layer	Changes in IR beam levels (simulation for gear levels)	Electric signal on the pins of Raspberry Pi read as a digital input.	We will test by breaking the IR beam with different things like coin, scale, pen etc. Then, we will verify the change of the values by sensor are similar in all conditions.	High
2	Hardware I/O Layer	Control commands From Raspberry Pi.	Operation of the camera and changing values of the sensor.	We will use Raspberry Pi to send a command to turn on or off the sensor, then read switch on and off the camera based on the sensor values.	High

3	User Interface Layer	User Interaction We will test by	Information display to user	We will test the UI by clicking on some objects on UI display and check the response is reflected on information display According to the desired action.	High

III. Features To Be Tested

This sections details which requirements will be tested during the testing phase. The requirements listed below correspond to the various requirements specified in our SRS.

A. Customer Requirements

This section describes all the customer requirements and gives a brief description of how each feature will be tested to meet the acceptance criteria.

- **Central Control Unit**

  Description: The central control unit is responsible for all communication between the mobile application, IR beam sensors and RPi camera. The control unit will transmit the status of the IR beam sensors to the camera and sends the data from the camera to mobile application.

  Test Approach: We will test the Raspberry Pi for all physical and functional tests. We will connect all the ports like USB, Ethernet, power, audio, and HDMI and test it to make sure it works correctly. We will also test it for its communication with the Hardware I/O Layer to ensure its functioning as well.

- **IR Beam Break Sensors**

  Description: Sensors that monitor the status of the gear and report when kept in reverse mode.

  Test Approach: We will provide different break down scenarios to the sensor and measure its accuracy such that it won't break or malfunction.

- **RPi Camera**

  Description: Starts and stops recording upon the command.

Test Approach: We will test the response time and efficiency of the video feed to the user in different connections.

- **Mobile Application**

Description: The mobile application will be used to interface with the server on Raspberry Pi. The application is responsible for registering with the server and accessing the video. The application will be built with a user-friendly interface to enable user to access easily.

Test Approach: We must be able to get the notification when the video streaming starts and able to accept or reject the stream. We will test the quality and efficiency of the video stream under different connection speeds.

B. Performance Requirements

This section describes all the performance requirements listed on our System Requirement Specification document and gives a brief description of how each feature will be tested to meet the acceptance criteria of the requirement.

- **Sensor Accuracy**

Description: Proper sensor readings of soil moisture levels must be captured to ensure accurate and efficient watering schedules. This requires solid construction of the sensor modules and proper software analysis of the data provided by the sensors.

Test Approach: All sensors will be tested for their accuracy. To get the accurate reading from the soil moisture sensor, the sensors will be placed on the right depth of soil. Rain sensor and temperature sensor will be on the rooftop. While coding, we will make sure the calibration will be exact.

- **Device Power Malfunction**

Description: If the mobile application fails to receive communication from the server for a specified period of time, it will notify the user that system is offline.

Test Approach: We will test every device and make sure that there will not be any power malfunction. We will also test the device to make sure it will go to fail-safe mode in case of any damage to device.

- **Latency for Streaming**

  Description: Real time Video Streaming is very important for this project. But delay in video frames for few seconds does harm to user. After getting sensor breaks, it should start streaming video. So, the response time is important.

  Test Approach: We will initiate a transfer for the video streaming instantly. We will note the time difference between the sensor break off and response transfer to user. We will test this multiple times and under different environments like varying locations, different networks and internet speed to make sure our latency is consistent. It should not take more than 3 seconds for our system to respond with video frames.

C. Packaging Requirements

This section describes all the packaging requirements and gives a brief description of how each feature will be tested to meet the acceptance criteria.

- **Control Unit Box**
  Description: The components of the control unit, including its chip module, will be placed inside a mountable hard plastic container.

  Test Approach: All the components will be tested if it fits inside the box. Box must be perfect enough such that module would not move.

- **IR Beam Break Sensors Packaging**
  Description: IR sensors will be located along with the Raspberry pi module box and attached firmly to the box such that it would not move.

  Test Approach: IR sensors will be tested with the package that has been installed on and work exactly as it intended to. We will make sure that the proper documentation gives good knowledge about installing these sensors.

- **RPi Camera Packaging**
  Description: RPi Camera will be located within the main module box facing the camera to the outside.

  Test Approach: We will test that the RPi Camera fits firmly with the main module package and the box secures it.

- **Connecting Cables**
  Description: Power cable and Ethernet cable will be included in the final package. Their purpose is to be used with the central control unit.

  Test Approach: The cables will be tested before placing inside the final box. We will check the quantity and quality of cables before deploying.

D. Safety Requirements

This section describes all the safety requirements and gives a brief description of how each feature will be tested to meet the acceptance criteria of the requirement.

- **IR Beam Break Sensor Insulation**
  Description: The electronic components of these sensors must be properly insulated against external environmental conditions to ensure that they do not malfunction when in use.

  Test Approach: The wires connecting the sensor must not touch any conductor or other modules.

- **Proper Wiring of Raspberry Pi 3B module**
  Description: The central control unit will be powered from an external plug. It is very important that this unit is wired properly to accept the external input and cause no harm to the device or sensors

  Test Approach: We will test that the wires coming and going to the unit are not harm, insulated and connected properly.

IV. Features Not To Be Tested

This section will describe the requirements that will not be tested during the testing phase. Most of the requirements cannot be tested, their functionality has no impact on the system operations.

A. Customer Requirements

All customer requirements will be tested.

B. Performance Requirements

All performance requirements will be tested.

C. Packing Requirements

User Manual: A user manual will be given to the user that describes the usage and operations of this product. It will be provided in the final packaging as a CD-ROM.

Reasoning: User manual needs no testing. It is just written instructions for the user.

D. Safety Requirements

All safety requirements will be tested.

V. Testing Approaches

This section will describe the approach followed for testing this project. These tests ensures that every critical requirement specified in SRS is met and tested for completion. This section will also describe the methods and tools which will be used throughout the testing phase project.

A. Strategy

The strategy approach for project implementation is to start testing from the beginning. Each component will be tested after it finishes. This approach is to prevent integration of incomplete and untested components. Once component tests are successful we will then begin integration testing with each completed component.

So, once a component is integrated with another, tests the functionality of the Integration before adding in another component. Each integration will be tested to ensure each sub-component functions as it is described. To provide a thorough understanding before system testing, we test as overall functionality that is possible at the point of integration.

Once integration is successful, steps for the overall system testing begins. The strategy to test all functionality as a whole to ensure that each and every requirement has been met and works properly.

B. Tools and Environment

The following is a list of tools to be used for development testing:

- Xcode
- Visual Studio
- Mocha.js
- TestNG

Environment needs*:*

This sections describes the hardware, software, operating systems, and any other attribute of the environments that could affect the test.

- iOS device, macOS or Windows.
- Sensor, RPi Camera, Raspberry Pi 3B+, IOS Mobile Phone.
- Node, NPM, xCode

C. Test Metrics

Priority	Description	Pass Criteria	Fail Criteria
Critical	Features that system relies on for critical functionality. And these features must be completed for the system to function accurately and effectively.	100%	Less than 100%
High	Features that are highly important by the system but are not completely required for the system to function properly.	90%+	Less than 90%
Moderate	Features that improve the quality of the system but are not required for the system to function properly.	75%+	Less than 75%
Low	Features reserved for future requirements or extras that do not impact the system functionality.	0%+	N/A

VI. Item Pass/Fail Criteria

A. Hardware Tests

Test ID	Hardware	Pass Criteria	Fail Criteria
1	Raspberry Pi 3B+	• Processes HTTP requests. • Relay commands to the sensor and camera.	• Does not process HTTP requests. • Does not relay commands to the sensor and camera.
2	IR Beam Break Sensor	• Reflects the beam connection values. 1 - connect 0 - disconnect	• Does not reflects the beam connection values.
3	RPi Camera	• Start and Record the video when instructed.	• Does not start or Record the video when instructed.

B. Unit Tests

Test ID	Module	Pass Criteria	Fail Criteria
1	Connection	• Should be able to accept the user connection through Wi-Fi Module. • Store the permission key file.	• Does not able to accept the user connection through Wi-Fi Module. • Does not store the permission key file
2	Notification	• Should be able to send the notification.	• Does not able to send the notification.
3	Video Streaming	• Should be able to create a HTML response with live video stream and send to the user.	• Does not able to create or send the HTML response with live video to the user.
4	Total no of Clients	• Should be able to send a JSON response with the total number of clients connected to server at that time.	• Does not able create a JSON response • Does not able send the correct number of connected clients to server.

5	Correct constraints depending on API results	• Navigate through setup process. • Successful Initial Setup. • Help user registering device and setup process	• Unable to navigate through setup process. • Fail during Initial Setup. • Unable to setup process and register the user with the server.
6	One time setup	• Redirect to Home screen on every launch.	• Unable not Redirect to Home screen on every launch.
7	Start Streaming	• Stream the data frames from the HTML response from the server.	• Unable not stream the data frames from the HTML response from the server.
8	Move to Home screen after Streaming.	• Stream ending notification • Navigation to the Home Screen after video stream	• Unable to show notification after ending video stream. • Unable to navigate to the Home Screen after video stream.

C. Integration Test

Test ID	Layer	Pass Criteria	Fail Criteria
1	Sensor Layer	• Reflects the beam connection values. 1 - connect 0 - disconnect • Should be able produce electric signal on the pins of Raspberry Pi read as a digital input.	• Does not reflect the correct beam connection values. • Unable to convert electric signals to digital signal.
2	Hardware I/O Layer	• Convert electric signal to correct digital signal. • Execute control commands to camera and sensor. • Interface with UI and process the user request appropriately.	• Could not convert electric signal to correct digital signal. • Unable to execute control commands to camera and sensor. • Fail to respond or process the user request appropriately.

3	User Interface Layer	• Interface with Hardware I/O Layer. • Parse incoming and outgoing data into proper format. .	• Fail to send and receive data with Hardware Interface Layer. • Fail to parse incoming or outgoing data into proper format.

VII. Test Deliverables

This section explains what would be the output of the product after completion. It explains the expected deliverables and the methods we will use to have the operated tests and documents all the tests performed.

A. System Test Plan

System test plan briefly explains what components of the project are to be tested and how to carry out the test.

B. Test Case Specification

Each test case will consist of following terms:

• **Test Case ID**: A unique ID of the test case.

• **Description:** Details of the test performed and description of the test.

• **Pre-Condition:** The condition the system for performing the test.

• **Post Condition:** The expected condition by the system after the test completion.

• **Input:** The input for to the test case.

• **Expected Output:** The expected output from the test case for the given input.

• **Process:** Step by step walkthrough of the test performed.

C. Test Case Results

Each test case results will be documented with following properties:

• **Test Case ID**: A unique ID of the test case.

• **Tester Name**: Name of the person who perform the test.

• **Test Date:** Date of the test was performed.

• **Inputs**: All inputs given to perform the test case.

• **Expected Output**: The output that was expected for the given input.

• **Actual Output**: The actual output of the function

- **Result of Test:** Test Pass/ Test Fail.
- **Fault ID:** A unique ID of the specific type of fault.
- **Description:** Explanation of the reason behind pass/fail of the test case.

D. Faults

Each fault will be documented with following properties:

- **Test Case ID**: A unique ID of the test case.
- **Fault ID:** A unique ID of the specific type of fault.
- **Result ID:** A unique ID of the specific type of fault.
- **Error Log**: A warning information or message associated with specific fault.
- **Description:** Detailed explanation of the reason of the fault.
- **Fault Fix Status:** The condition of the fault whether it's been fixed or not.

VIII. Test Schedule

No.	Task Name	Planned Start Date	Planned Finish Date	Estimated Time
1.	**Systems Testing Phase**	**7/16/2018**	**7/26/2018**	**43**
1.1.	**System Testing Phase (Hardware Testing)**	**7/16/2018**	**7/20/2018**	**6**
1.1.1	Raspberry Pi 3B+	7/16/2018	7/20/2018	2
1.1.2	IR Beam Break Sensor	7/16/2018	7/20/2018	2
1.1.3	RPi Camera	7/16/2018	7/20/2018	2
1.2	**System Testing Phase (Unit Testing)**	**7/18/2018**	**7/22/2018**	**19**
1.2.1	Connection	7/18/2018	7/22/2018	1
1.2.2	Notification	7/18/2018	7/22/2018	1
1.2.3	Video Streaming	7/18/2018	7/22/2018	5
1.2.4	Total No of Clients	7/18/2018	7/22/2018	0.5

1.2.5	Correct constraints depending on API results	7/18/2018	7/22/2018	2
1.2.6	One time setup	7/18/2018	7/22/2018	2
1.2.7	Stream VC	7/18/2018	7/22/2018	6
1.2.8	Move to Home screen after Streaming.	7/18/2018	7/22/2018	1.5
1.3	**System Testing Phase (Integration Testing)**	**7/20/2018**	**7/26/2018**	**18**
1.3.1	Sensor Layer	7/20/2018	7/26/2018	6
1.3.2	Hardware I/O layer	7/20/2018	7/26/2018	6
1.3.3	User Interface Layer	7/20/2018	7/26/2018	6

IX. Roles and Responsibilities

All the team members are responsible for testing the software and other deliverables. Unit test cases are done by the team member who developed the particular functionality. All other testing’s like Hardware and Integration test cases will be contributes by every team Member.

Appendix:

Source Code: **https://github.com/ShivangDave/Server**
https://github.com/ShivangDave/Automatic-Backup-WiFi-Camera

Server Side Scripts - Node JS

Filename: index.js

/* Npm Packages require to execute the module*/

```
var express = require('express');
var bodyParser = require('body-parser');
var apn = require('apn');
var app = express();
var fs = require('fs');
var request = require('request');
var gpio = require('rpi-gpio');
const cv = require('opencv4nodejs');

var token;
var clients = 0;
var interval;
var sensor_int;
var reverse = false;
var camera = new cv.VideoCapture(0);
camera.set(3,320);
camera.set(4,240);
```

/* Input from GPIO pins on Raspberry Pi */

```
gpio.setup(2,gpio.DIR_IN,looping);
gpio.setMode(gpio.MODE_BCM);
```

/* Function to initialize the camera (Shivang Dave)*/

```
function initCam()
{
  if(camera!=null)
  {
    reInitCam();
  }
  else
  {
    camera = new cv.VideoCapture(0);
```

```
    camera.set(3,600);
    camera.set(4,400);
  }
}
```

/* Function to which loopes each microsecond to record based on the GPIO pin value (<u>Shivang Dave</u>) */

```
function looping()
{
  sensor_int = setInterval(()=>{
    gpio.read(2, function(err, value) {
        if (err) throw err;
        if(value==true)
        {
          if(reverse==true)
          {
            reverse = false;
            releaseCam();
            sendNotification(1);
            console.log('Driving');
          }
        }
        else
        {
          if(reverse!=true)
          {
            reverse = true;
            initCam();
            sendNotification(0);
            console.log('Reverse');
          }
        }
    });
  },1);  }
```

/* Function to generate the send internal POST request to notify user (<u>Sumanjali Tirunagaru</u>) */

```
function sendNotification(arg)
{
  base_url = "http://localhost:3000/notify";
  request.post(base_url,{json:{'flag':arg}},function(err,res,body)
  {
    if(!err)
```

```
    {
      console.log('Notification Sent!');
    }
  });
}
```

/* Function to get stop recording by making camera handler to null (<u>Vignesh Kirubakaran and Sumanjali Tirunagaru</u>)*/

```
function releaseCam()
{
  clearInterval(interval);
  if(camera!=null)
  {
    camera.release();
    camera = null;
  }
}
```

/* Function to reinitialize the camera (<u>Shivang Dave and Vignesh</u>)*/

```
function reInitCam()
{
  releaseCam();
  camera = new cv.VideoCapture(0);
  camera.set(3,320);
  camera.set(4,240);
}
```

/* Function to get generate each frame while camera starts recording (<u>Vignesh Kirubakaran</u>)*/

```
function get_frame()
{
        var jpeg;
        if(camera==null)
        {
          reInitCam();
        }
        var some = camera.read();
        var bytes = cv.imencode('.jpg',some).toString('base64');
        return bytes;    }
```

/* Function to get validate the token from the users mobile phone is Null or Not ((<u>Gowri Priya and Nischal Reddy</u>))*/

```
function checkToken()
```

```
{
  if(token!=null)
  {
    return true;
  }
  else
  {
    return false;
  }
}
```

/* Function to register the users mobile application token at server ((<u>Gowri Priya</u>))*/

```
function setToken(arg)
{
        token = arg;
        clients += 1;
        console.log(token);
}
```

/* To parse Incoming body request to JSON format */

```
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: false}));
app.use(function(req,res,next){
        res.locals.errors = null;
        next();
});
```

/* Server handles a GET request with url '/start' - sends the video frame as Html response (<u>Shivang Dave</u>) */

```
app.get('/start', function(req,res)
{
        req.on("close", function() {
            res.end("Stopped");
            releaseCam();
        });
        if(camera==null)
        {
          initCam();
        }
        res.setHeader('Content-Type','multipart/x-mixed-replace; boundary=frame');
```

```
	res.setHeader('Connection', 'close');

	interval = setInterval(function(){
	   res.write("\r\n--frame\r\n");
	   res.write("Content-Type: image/jpeg\r\n\r\n");
	   res.write(Buffer(get_frame(),'base64'));
	},33.33); //60 FPS ~ 16.66 //30fps ~ 33.33 //shutter speed
})
```

/* Server handles a GET request with url '/stop
- stops the video stream as Text response
(<u>Gowri Priya and Nischal Reddy</u>)*/

```
app.get('/stop', function(req,res){
  res.setHeader('Content-type','text/plain');
  res.send('Stopped');
  releaseCam();
})
```

/* Server handles a GET request with url '/checkToken -
sends a boolean response True if token present else response is False.
(<u>Sumanjali Tirunagaru and Gowri Priya</u>)*/

```
app.get('/checkToken', function(req,res){
  res.setHeader('Content-type','application/json');
  res.send(JSON.stringify({
	token: checkToken()
  }))
})
```

/* Server handles a POST request with url '/connect from User
and receives user's Token - sends a success message as response.
(<u>Vignesh Kirubakaran</u>)*/

```
app.post('/connect', function(req,res){
	res.setHeader('Content-Type', 'application/json');
		res.send(JSON.stringify({
			response: "Token Registered!"
		}));
		setToken(req.body.token);
})
```

/* Server handles a POST request with url '/notify -
sends the Notification Object along with message in the notification in the response.
(<u>Shivang Dave and Sumanjali Tirunagaru)</u> */

```
app.post('/notify', function(req,res){
	/*0 for start
	1 for stop
	{flag:0 or 1}
	*/
		var flag = req.body.flag;
		let service = new apn.Provider({
	  cert: "/home/pi/Desktop/Server/Certificate/cert.pem",
	  key: "/home/pi/Desktop/Server/Certificate/key.pem",
	 });
		var note = new apn.Notification();

	 //note.expiry = Math.floor(Date.now() / 1000) + 3600; // Expires 1 hour from
now.
		note.expiry = 0
		note.priority = 10
	 note.badge = 1;
	 note.sound = "ping.aiff";
	 if(flag != 1)
		{
			note.alert = "\u2709 Stream is starting";
		}
		else
		{
			note.alert = "\u2709 Stream ended!";
		}
		note.aps.category = flag
	 note.payload = {'messageFrom': 'Shivang Dave'};
	 note.topic = "sd.Automatic-Backup-WiFi-Camera";

	 service.send(note, token).then( (result) => {
			res.setHeader('Content-type','application/json');
			res.send(JSON.stringify({
				response: "Notification sent!",
				flag: flag
			}))	});	})
```

/* Server is up and running on port 3000 on Raspberry Pi */

```
app.listen(3000, function(){
	console.log('Server started on port 3000');
})
```

/* server object ‘app’ is returned when this file is used as module in other Js files.*/

```
Module.exports =>{app};
```

Server Side Unit Test Scripts - Node JS, MochaJS

Filename: index.test.js

```
/* Npm Packages  require to execute the module*/
const expect = require('expect');
const request = require('supertest');

//const {app} = require('./../dummyserver.js');
const {app} = require('./../index.js');
```

/*Test script for /connect module in server -
(<u>Sumanjali Tirunagaru</u>)*/

```
describe('POST: /connect ',()=>{
  it('Should Test the POST request of \'/connect\' response details', (done)=>
  {
    var expected = JSON.stringify({response: 'Token Registered!'});

    request(app)
    .post('/connect')
     .set("Content-Type", "application/json")
    .send({
    token : 'Automatic Backup WiFi Camera'
    })
    .expect(200)
    .expect("Content-Type", "application/json; charset=utf-8")
    .expect((res)=>{
      expect(res.body.response).toBe('Token Registered!');
    })
    .end(done);
  }); //it
}); //describe
```

/*Test script for /checkToken module in server for Token null condition
(<u>Sumanjali Tirunagaru</u>)*/

```
describe('GET: /checkToken : token - null',()=>{
  beforeEach((done)=>{
    request(app)
    .post('/connect')
     .set("Content-Type", "application/json")
    .send({
    token : null
  }).end(done);
  });
```

```
 it('Should Test response is FALSE when Token : NULL ', (done)=>
 {
   request(app)
   .get('/checkToken')
   .expect(200)
   .expect("Content-Type", "application/json; charset=utf-8")
   .expect((res)=>{
     expect(res.body.token).toBe(false);
   })
   .end(done);;
 }); //it
}); //describe
```

/*Test script for /checkToken module in server for Token - not null condition
(<u>Sumanjali Tirunagaru) </u>*/

```
describe('GET: /checkToken : token - Not null',()=>{
 beforeEach((done)=>{
   request(app)
   .post('/connect')
    .set("Content-Type", "application/json")
   .send({
   token : 'Automatic Backup WiFi Camera'
 }).end(done);
 });

 it('Should Test response is TRUE when Token : NOT NULL', (done)=>
 {
   request(app)
   .get('/checkToken')
   .expect(200)
   .expect("Content-Type", "application/json; charset=utf-8")
   .expect((res)=>{
     expect(res.body.token).toBe(true);
   })
   .end(done);;
 }); //it
}); //describe
```

/*Test script for /notify module in server for flag value 0 and 1 and tested for required response
(<u>Sumanjali Tirunagaru) </u>*/

```
describe('POST: /notify',()=>{
  it('should test: flag value = 1 => Stream Ended',(done)=>{

    request(app)
    .post('/notify')
    .set("Content-Type", "application/json")
    .send({
    flag : 1
    })
    .expect(200)
    .expect("Content-Type", "application/json; charset=utf-8")
    .expect((res)=>{
        expect(res.body.response).toBe('Notification sent!');
        expect(res.body.flag).toBe(1);
        expect(res.body.note.aps.alert).toBe('✉Stream ended!');
        }).end(done);
  }); //it

  it('should test: flag value != 1 => Stream starts',(done)=>{

    request(app)
    .post('/notify')
    .set("Content-Type", "application/json")
    .send({
    flag : 0
    })
    .expect(200)
    .expect("Content-Type", "application/json; charset=utf-8")
    .expect((res)=>{
        expect(res.body.response).toBe('Notification sent!');
        expect(res.body.flag).toBe(0);
        expect(res.body.note.aps.alert).toBe('✉Stream is starting');
        }).end(done);
  }); //it
});//describe
```

Server Side Package - Node JS

Filename: package.json

/*JSON file generated by npm installing the required Dependency and Devdependency files and to start the server */

```
{
  "name": "server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "node index.js",
    "test": "mocha **/*.test.js",
    "test-watch": "nodemon --exec 'npm test'"
  },
  "author": "Shivang Dave",
  "license": "ISC",
  "dependencies": {
    "apn": "^2.2.0",
    "body-parser": "^1.18.3",
    "express": "^4.16.3",
    "nodemon": "^1.17.5",
    "npm": "^6.2.0",
    "npmlog": "^4.1.2",
    "request": "^2.87.0",
    "rpi-gpio": "^1.0.0"
  },
  "devDependencies": {
    "expect": "^1.20.2",
    "mocha": "^3.0.2",
    "supertest": "^2.0.0"
  }
}
```

Weekly Reports:

CSCI 6838 - Research Project And Seminar
Summer 2018

Weekly Report : 11th June 2018 - 18th July 2018

Automatic WiFi Backup Camera (*Team 3)*

Meeting Agenda	
Time	1:00 PM 13th June 2018 and 4:30 PM 15th June 2018
Location	Delta PC Lab
Attendees:	Shivang, Sumanjali, Vignesh, Nischal, Gowri Priya
Topic to Discuss	- To analyze the basic requirements of the project. - To design the workflow of the project. - To decide on the languages to be used. - To discuss about the Protocols to be followed. - To discuss the roles of each member.

Milestones Achieved:

- Initial draft for the Requirement Specification Document.
- Workflow for the Project.

What was done this week:

Created initial draft for SRS document, agreed for the Workflow, Technology and Hardware to be used in the project.

Risks, Impediments, Blockers:

Deciding on the streaming protocol for the video with least latency.

Next week objective(s):

- To generate a Design Document and UI design of iOS app.
- To Generate the UML diagrams.

CSCI 6838 - Research Project And Seminar
Summer 2018

Weekly Report : 18th June 2018 - 25th July 2018

Automatic WiFi Backup Camera (*Team 3)*

Meeting Agenda	
Time	4:00 PM 21st June 2018 and 4:00PM 25th June 2018
Location	Delta PC Lab
Attendees:	Shivang, Sumanjali, Vignesh, Nischal, Gowri Priya
Topic to Discuss	- To discuss and prepare the vision and scope document. - Working on the protocol for streaming video. - Topics to include in a design document

Milestones Achieved:

- Initial draft for the Vision document.
- Workflow for the Project.
- Structure for a design document

What was done this week:

Created initial draft for Vision document, working on different protocols for the streaming with low latency.

Risks, Impediments, Blockers:

Deciding on the streaming protocol for the video with least latency.

Next week objective(s):

- To generate a Design Document and UI design of iOS app.
- To Generate the UML diagrams.

CSCI 6838 - Research Project And Seminar
Summer 2018

<u>Weekly Report : 25th June 2018 - 2nd July 2018</u>

Automatic WiFi Backup Camera

Team 3

Meeting Agenda	
Time	4:00 PM 29th June 2018 and 4:00PM 2nd July 2018
Location	Delta PC Lab
Attendees:	Shivang, Sumanjali, Vignesh, Nischal, Gowri Priya
Topic to Discuss	- To discuss and work on final draft for the Design document. - Working on the prototype. - working on Nodes JS scripts

Milestones Achieved:

- Final draft for the Design document.
- System Architecture & UI Designing
- Hardware Setup

What was done this week:

Created final draft for Design Document and Systems architecture and UI designs, working on first prototype and hardware setup and initial backend scripts

Risks, Impediments, Blockers:

Deciding on the simple yet user friendly UI interface for the Mobile application.

Next week objective(s):

- To prepare final first prototype.
- To work on the coding part of the front-end and back-end for the project.

CSCI 6838 - Research Project And Seminar
Summer 2018

Weekly Report : 2nd July 2018 - 9th July 2018

Automatic WiFi Backup Camera

Team 3

Meeting Agenda	
Time	4:00 PM 6th July 2018 and 4:00PM 9th July 2018
Location	Delta PC Lab
Attendees:	Shivang, Sumanjali, Vignesh, Nischal, Gowri Priya
Topic to Discuss	- Front End Design - Working on the prototype. - Working on Nodes JS scripts

Milestones Achieved:

- Front End Development.
- Initial Server setup

What was done this week:

- Working on first prototype and hardware setup and initial backend scripts

Next week objective(s):

- To work on the coding part of the back-end for the project.

CSCI 6838 - Research Project And Seminar
Summer 2018

Weekly Report : 9th July 2018 - 16th July 2018

Automatic WiFi Backup Camera

Team 3

Meeting Agenda	
Time	4:00 PM 13th July 2018 and 4:00PM 16th July 2018
Location	Delta PC Lab
Attendees:	Shivang, Sumanjali, Vignesh, Nischal, Gowri Priya
Topic to Discuss	- Working on Test Plan Document. - Working on the Video Streaming protocol. - Integrating Hardware Components - Working on Nodes JS scripts

Milestones Achieved:

- Front End Development.
- Initial draft for Test Plan Document.

What was done this week:

- Working on hardware setup and backend scripts including protocol to be implemented.
- Prepared a draft for Test plan document.

Next week objective(s):

- To work on the integrating hardware and testing the project modules.

CSCI 6838 - Research Project And Seminar
Summer 2018

Weekly Report : 16th July 2018 - 23rd July 2018

Automatic WiFi Backup Camera

Team 3

Meeting Agenda	
Time	4:00 PM 18th July 2018 and 4:00PM 20th July 2018
Location	Delta PC Lab
Attendees:	Shivang, Sumanjali, Vignesh, Nischal, Gowri Priya
Topic to Discuss	- Final draft and presentation Test Plan Document. - Integrating all Software and hardware Modules. - Working on Unit and Integration tests manually. - Working on Final prototype.

Milestones Achieved:

- Test Plan Document.
- Unit Tests.
- Integration of all Modules

What was done this week:

- Working on Unit and Integration tests.
- Prepared and presented final draft for Test plan document.
- work on final prototype.

Next week objective(s):

- To finish final report and final prototype of the project.

CSCI 6838 - Research Project And Seminar
Summer 2018

Weekly Report : 23rd July 2018 - 27th July 2018

Automatic WiFi Backup Camera

Team 3

Meeting Agenda	
Time	4:00 PM 24th July 2018 1:00 PM 26th July 2018 4:00 PM 27th July 2018
Location	Delta PC Lab
Attendees:	Shivang, Sumanjali, Vignesh, Nischal, Gowri Priya
Topic to Discuss	- Final Report Documentation - Final Presentation. - Integrating all Software and hardware Modules. - Working on Final prototype.

Milestones Achieved:

- Final Prototype
- Deployment.

What was done this week:

- Working on Integration tests.
- Presented final Report for Project
- Work on final prototype.