# *Software Engineering*
# *Software Requirements Specification (SRS) Document*

*Edward di Girolamo*
*BestTech Company*
*Udemy Course Example*
*03/23/2023*

# Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| Final Draft | Team | All sections being Filled | 3/23/2023 |

# Review & Approval

### Requirements Document Approval History

| Approving Party | Version Approved | Signature | Date |
|---|---|---|---|
| **Edward G. Di Girolamo** | | | |
| **Bruce Wayne** | | | |

### Requirements Document Review History

| Reviewer | Version Reviewed | Signature | Date |
|---|---|---|---|
| **Bill Fences** | | | |
| **Steve Works** | | | |
| **Larry Book** | | | |

# Contents

# 1. Introduction

**1. Introduction**

This document has been written to outline the background, needs, and vision our client has expressed to us in meetings and other forms of communication. Defined here are the purpose, scope, and various details to outline in the best of our abilities our clients expectations and project parameters. Our customer the Electronic Entertainment Cafe aka. EE_Cafe, are an open plan coffee and tea house located in New York, New York. They provide various computer networks for their guests and employees.

**2. Scope of this Document**

Our customer the Electronic Entertainment Cafe aka. EE_Cafe and the software engineering team at BestTech Software Company will be working on the project. The EE_Cafe has a Cisco Meraki based wireless hardware installation on site. This document provides the project details and timeline for each component and final completion.

**3. Overview**

The product will be an web dashboard that integrates with the Meraki API to let the admin add, subtract, and deploy networks based on events happening in the store. The customer has asked for a automatic deployable Ansible script to automate the website deployment as they may be switching cloud web providers and don't want to have to rebuild the server.

**4. Business Context**

During normal operating hours the EE_cafe allows guests to work on a normal guest wifi network and the employees get a private network. However the EE_cafe allows companies to host private meetings and local gaming clubs to host wireless lan parties on site.

# 2. General Description

**2.1 Product Functions**

The EE_Cafe wants to utilize the API of their Cisco Meraki systems to create new networks with bandwidth limitations for their specific private guests. This will keep the gaming groups, the corporate groups, and general guests on separate networks. Because several general store operators are not very tech savvy these functions must be on a web based interface used in the store or accessible to offsite employees.

**2.2 Similar System Information**

This product can be designed for use with web browsers and the lowest cost web server available. It has bee requested that the website be deployable with Ansible in a simple playbook. The code for the website will be hosted on Github. The current networking hardware and API that the website needs to access is Cisco Meraki. Only general security precautions need to be taken.

**2.3 User Characteristics**

The users for this website will be employees of the EE_Cafe. Each having varied experience with computers. The will know which guests and events are going on in the

cafe and will need a simple button and form system to add new networks and remove networks on the website. Some minimal monitoring of bandwidth uses may be used as a log or as a dashboard.

### 2.4 User Problem Statement

Currently the owner has access to the Meraki Web app but wants a custom web interface for her employees. Most employees are familiar with using websites and know that there will be new networks to add and old ones to remove. However, the process must be as simple and non-technical as possible.

### 2.5 User Objectives

The customer wants the dashboard deployed as an website so that the owner can login and check the networks from anywhere. The customer wants a dashboard with the features to add, remove, and list networks, as well as, users and devices on those networks. The customer wants the code online and deployable through an Ansible script.

### 2.6 General Constraints

Keep the system simple this website is primarily for the customers internal uses. It must be easily to redeploy and connect to their Meraki hardware.

# 3. Functional Requirements

1. **A Form and Button based website shall operate the Cafe's internal/external Meraki Wireless networks**
    1. All data shall be stored in the websites database.
    2. Very high criticality
    3. Limited user experience requires the simplest dashboard.
    4. The above issue will be mitigated by using the web app dashboard instead of CLI or other means.
    5. This requirement is the basis of the project; all other aspects depend on it.

2. **The Website Shall be deployed on a Webserver.**
    1. An MVC web app python flask will be deployed on the server. The front end having html, css, and javascript, the backend using an linux based python web server and SQL database managed by python flask.

2. Very high criticality
3. Technical issues may arise in versioning python flask and Meraki API's.
4. Given the store maintains their subscription with third party API's we will still version lock the system. All future upgrades of hardware are not in the scope of this project.
5. This requirement depends on requirement number one.

3. **Products and Code provided to EE_Cafe shall be uploaded to their Github account.**
    1. All code needs to be readily redeployed therefore online in an accessible git account with a network automation script written in Ansible.
    2. Very high criticality
    3. We do not foresee any technical risks involved in this requirement.
    4. The only issue here is that an experience network automation specialist will need to run the Ansible script, however, this is easily overcome by training.
    5. This requirement is dependent on requirement one.

# 4. Interface Requirements

## 4.1 User Interfaces

- **4.1.1 GUI**
  The user interface for this program is the interface provided by Microsoft Access 2007. Access includes forms and reports for the users to query and organize data to suit their needs. Forms and reports both have builders that let the user specify which fields they want to use and which constraints they want to define.

- **4.1.2 CLI**
  Only the ansible script will require a knowledge of linux cli.
- **4.1.3 API**
  Our product will have the ability to add an API. We will be making calls to the Meraki API.
- **4.1.4 Diagnostics or ROM**
  We will provide links to third party documentation in the appendices.

### 4.2 Hardware Interfaces

All software will be hosted in the cloud. The website will be accessible with any authorized user with a device that has a web browser and internet connection.

### 4.3 Communications Interfaces

The website will be hosted on an open cloud server and communicate with Third Party Cisco Meraki API

### 4.4 Software Interfaces

Our web server will be hosted on an ubuntu platform. Running python flask. API calls will be made from the web service to Cisco Meraki.

## 5. Performance Requirements

The website needs to be accessible 24/7 with automatic redeployment available on an as needed basis. The website is only for employees so only the most basic web server is required.

Ubuntu OS:
Single core
1G of Ram
20GB Available Hard Drive Space
Minimum bandwidth requirements

## 6. Other non-functional attributes

.

### 6.1 Security

The system shall be designed with only the customer requirements of a user login and our recommendations of preventing the top 10 OWASP web security flaws in python flask provided apps.

### 6.2 Binary Compatibility

This is a webbased system and no binaries will be compiled.

### 6.3 Reliability

Only redeploy automation will be in the scope of this document. All backups and monitoring will be provided by the cloud service provider.

### 6.4 Maintainability

The system shall be maintained by EE_Cafe and their staff.

### 6.5 Portability

The system will be version locked and designed to deploy only on an Ubuntu server.

### 6.6 Extensibility

The system will use the MVC (Model View Controller) design based python flask library(s). Any one versed in using python flask will be able to extend the capabilities of the web app.

### 6.7 Reusability

The system should be designed to be reused for any network changes the EE_Cafe deemed needed.

### 6.8 Application Affinity/Compatibility

This system is a one time production and will be version locked to increase the longevity as long as current libraries and OS releases are available. If version locking were not to be used the system may require break fix coding to update library changes.

### 6.9 Resource Utilization

The resources need to run this system will be a cloud based web server, any user computer with a web browser, an antive Cisco Meraki API key and subscription.

### 6.10 Serviceability

The maintenance of the system should be able to be sufficiently performed by any person with a basic understanding of Python Flask, Ubuntu, and Ansible.

## 7. Operational Scenarios

## Scenario A: Initial Network Monitoring

The user shall enter in credentials and login to the configured url for the server and login page. After appropriate authorization the user may click on the dashboard page and see the current networks and devices that are deployed at the site.

## Scenario B: Guest or Temporary Network Additions

The authorized user shall be able to add a new network to be broadcast by the Meraki Wireless devices on location. Each network shall have options to control the bandwidth consumed from the overall network's capacity and the devices that can log into the network.

## Scenario C: Returning Networks to Normal Operations

The authorized user can remove temporary networks or reset the devices to the initial configuration laid out by the locations normal operating procedures.

# 8. Preliminary Use Case Models and Sequence Diagrams

This section presents a list of the fundamental sequence diagrams and use cases that satisfy the system's requirements. The purpose is to provide an alternative, "structural" view of the requirements stated above and how they might be satisfied in the system.

## 8.1 Use Case Model

This section presents a list of the fundamental sequence diagrams and use cases that satisfy the system's requirements. The purpose is to provide an alternative, "structural" view of the requirements stated above and how they might be satisfied in the system.

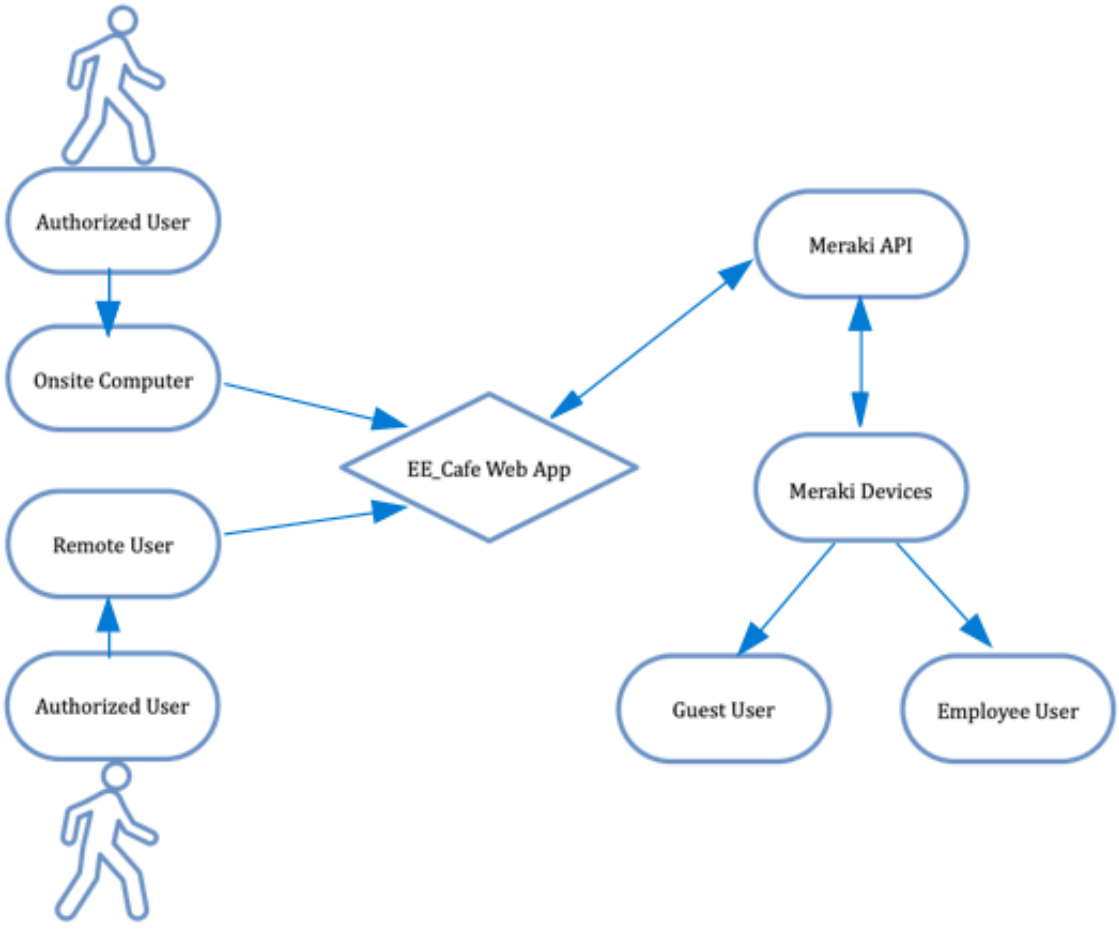
A manager at the EE_cafe opens the shop in the morning. Regular guests enter and leave during the normal operating of the cafe. On the schedule in the managers office the manager is notified about a company meeting with 8 guests that need a private network for a presentation and a local gamer convention that needs a high bandwidth private network for 20 guests. The peak capacity of the cafe is around 50 people for the internet bandwidth.

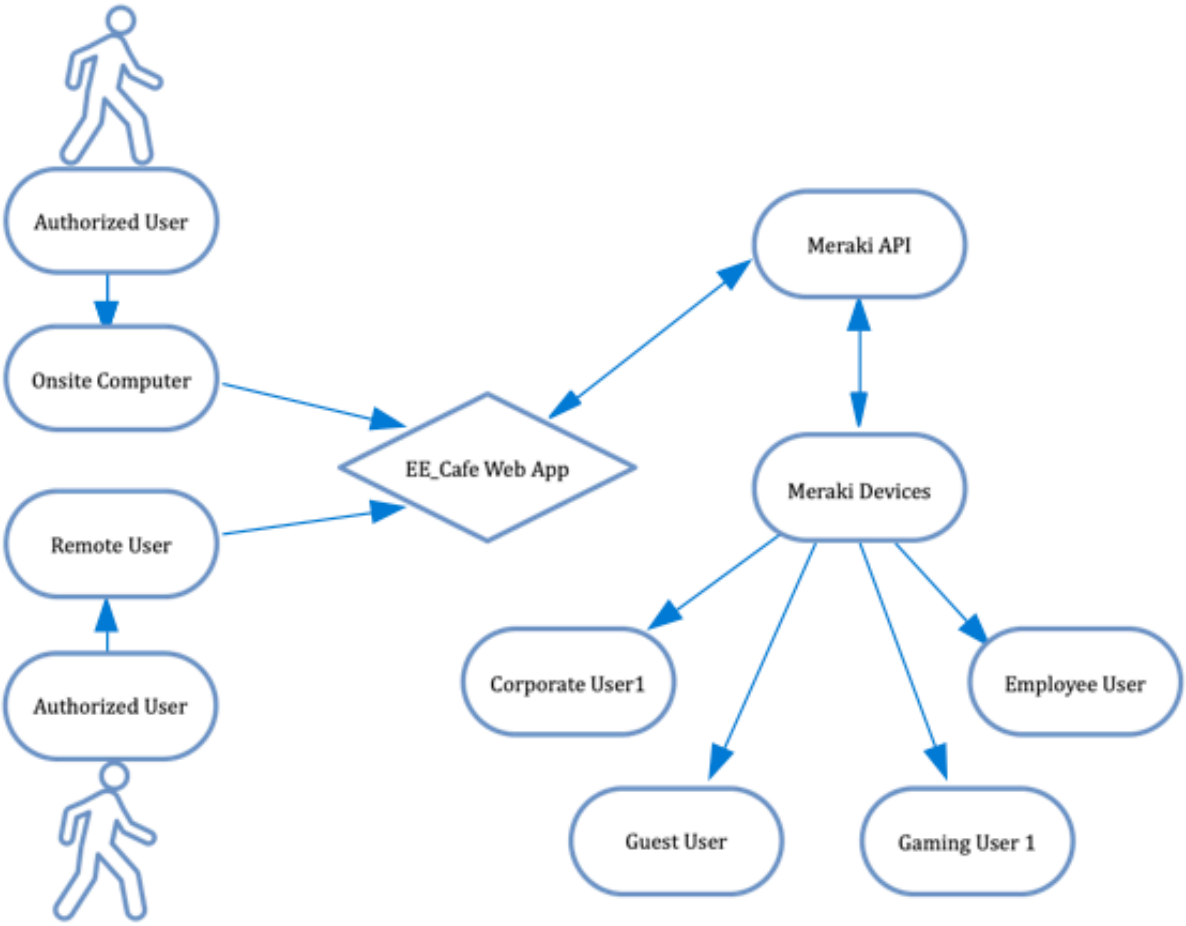
Before the guests arrive the manager logs into the EE_Cafe web app and checks the networks dashboard. The manager sees only the employee and guest wifi open with whatever devices are currently connected. The manager then clicks "Add Network" to add a new network; specifying the name, password, bandwidth allocation percentage and the max number of devices. During the events the manager can check that no additional devices are on those networks, as well as, the total on the original guest and employee's networks.

Afterwards the manager remove individual networks or if all events complete simultaneously the manager can reset the system back to the normal configuration.

## 8.2 Sequence Diagrams

Normal Operating Network Flow Configuration (First and Last Config)




Authorized User
Onsite Computer
Remote User
Authorized User
EE_Cafe Web App
Meraki API
Meraki Devices
Corporate User1
Guest User
Gaming User 1
Employee User

# 9. Updated Schedule

The Scrum project management board will be available on the GitHub url using Github projects.

# 10. Updated Budget

An updated budget is attached at the end of this document

# 11. Appendices

## 11.1 Definitions, Acronyms, Abbreviations

EE_Cafe- Electronic Entertainment Cafe (the fictional Customer)

## 11.2 External URLS and documentation

Python Flask Documentation - https://flask.palletsprojects.com/en/2.2.x/

Cisco Meraki Documentation - https://developer.cisco.com/meraki/api-v1/#!introduction/meraki-dashboard-api

Ansible Documentation - https://docs.ansible.com/

OWASP info - https://owasp.org/

Python Flask-Security Documentation - https://flask-security.readthedocs.io/en/3.0.0/

REST API documentation and useful links