Software Requirements Specification

for

Sight & Sound Audio Connections Point of Sale

Version 1.0

Prepared by Luke Harris

November 1, 2017

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 3

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 3

2.6 User Documentation 3

2.7 Assumptions and Dependencies 4

3. External Interface Requirements 4

3.1 User Interfaces 4

3.2 Hardware Interfaces 5

3.3 Software Interfaces 5

3.4 Communications Interfaces 5

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Luke Harris | 11/1 | Initial Draft | 1.0 |
| Luke Harris | 11/5 | Added User Stories |  |
| Luke Harris | 11/8 | Added Use Case Diagrams |  |
|  |  |  |  |

# Introduction

## Purpose

This is version 1.0 of Sight & Sound Audio Connections Point of Sale App. This app uses a Mongo database to store and track inventory, accounting information, and customer transactions. This app is also designed to integrate with Sight & Sounds web site to allow for online sales and inventory updates.

## Document Conventions

The documentation that follows is expected to follow these conventions:

·    The type font for the document will be Times.

·    Headers of main sections (I.E. 1, 2) will be in 18 point font and bold.

·    Headers for sub-sections (I.E 2.1, 2.2) will be in 14 point font and bold.

· Content will be in 12 point font.

## Intended Audience and Reading Suggestions

This document is intended for all readers.

Developers, and testers will get the best content from sections 2, 3, 4, 5, and 6

Project managers will get the best content from sections 2, 4, 5

Users will get the best content from sections 1, 2, 4, and 6

## Product Scope

This project will provide Sight & Sound with an easy to use, custom Point of Sale system, that can be integrated with their website. It will store inventory, customer transactions, and basic accounting reports. It will have the ability to create user and admin accounts, with different levels of access. Through the API, users will have the ability to view, and purchase products online via Sight & Sounds website.

References

Project Proposal, Luke Harris, Version 1.0, 11/1/2017:

<https://github.com/imp0st3r/ssacpos>

SRS, Luke Harris, Version 1.0, 11/1/2017:

<https://github.com/imp0st3r/ssacpos>

Use Case Diagrams, Luke Harris, Version 1.0, 11/1/2017:

<https://github.com/imp0st3r/ssacpos>

Desktop / Tablet / Mobile UIs, Luke Harris, Version 1.0, 11/1/2017:

<https://github.com/imp0st3r/ssacpos>

# Overall Description

## Product Perspective

The intention of this project is to make a functionally simple, Point of Sale system that can integrate with the customer’s website. The PoS will keep track of inventory, accounting, and transactions. It will also contain a RESTful API that can integrate the companies’ inventory with their website.

## Product Functions

* Store inventory in MongoDB
  + Subs
  + Speakers
  + Amps
  + Kits
  + Lights
  + Boxes
  + Players
  + Accessories
* Store customer transactions in MongoDB
  + Name
  + Address
  + Phone
  + Email
  + Vehicle Information
  + Products Purchased
  + Labor
  + Notes
* Provide Accounting Reports
  + Monthly Summary
  + Total Sales
  + Best Sellers
* Provide API for Website

## User Classes and Characteristics

This app is intended for use by Sight & Sound Employees, but could be adapted to any business wanting a simple Point of Sale with website capabilities.

## Operating Environment

SSAC Point of Sale is a MEAN stack web app with the addition of Neo4j. This means that it can be run from any Linux, Windows, or OSX server. This app can run on any JavaScript enabled device.

Currently, the official server specs are:

Ubuntu Linux 16.04 Server

MEAN Stack

* MongoDB
* Express
* Angular
* Node

## Design and Implementation Constraints

May require an internet connection. Server will be local per customer request. Currently planned to be released in English.

## User Documentation

Manual, Luke Harris, Version 1.0, 11/1/2017:

https://github.com/imp0st3r/ssacpos

## Assumptions and Dependencies

App requires the following to run:

NodeJS

MongoDB

# External Interface Requirements

## User Interfaces

This app will adjust to fit any device size. Sight & Sound Audio Connections logo appears at the top left of every screen and serves as a button to take the user to the home (dashboard) screen if they are logged in, or the splash screen for a guest if a user is not logged in. A menu to navigate between sections is always available at the top right of every screen.

A Registered user will have the following options on their dashboard:

Manage Products

Manage Speakers

Manage Subs

Manage Amplifiers

Manage Boxes

Manage Players

Manage Accessories

Manage Lighting

Manage Invoices

List Invoices

Create Invoices

Edit Invoices

Delete Invoices

An Admin will have the following options on their dashboard:

Manage Products

Manage Invoices

Manage Users

Manage Reports

**\*\*\*See Appendix D for UI layouts\*\*\***

## Hardware Interfaces

This app can be run on any device that is JavaScript enabled and has a steady internet connection. It has been developed for mobile, tablets, and desktops.

## Software Interfaces

This is a Javascript Web App developed on the MEAN stack

Express Jade Templates v4.15.0

Bootstrap CSS v3.3.7

JQuery v3.2.1

AngularJS v2.0

Google Map API v3.2.7

MongoDB v3.4.2

NodeJS v6.10.2

RESTful API calls are used to access the user, product, and transaction databases, and pass data between the server and the app.

## Communications Interfaces

This application requires a JavaScript compatible browser, and an internet connection. Data is transferred over the HTTP protocol, and through RESTful API calls to MongoDB.

# System Features

## Registration

4.1.1 Description and Priority

A default admin account should be supplied, and all admins should be able to create users. High Priority

4.1.2 Stimulus/Response Sequences

Admins will have the ability to “register” new users to the PoS

4.1.3 Functional Requirements

REQ 1: UI must be functional

REQ 2: Application must be connected to MongoDB

REQ 3: Email must not already exist in the system

REQ 4: All fields must validate

## Login

4.2.1 Description and Priority

All registered users of the app should be able to login. High Priority

4.2.2 Stimulus/Response Sequences

To login the user can select “Login” from the splash screen. They will then be prompted for their user email and password. If an invalid email or password are entered the user will be notified. On a successful login the user will be redirected to the dashboard screen.

4.2.3 Functional Requirements

REQ 1: UI must be functional

REQ 2: Application must be connected to MongoDB

REQ 3: User must exist in the database

REQ 4: Email and Password must be correct

## Products

4.3.1 Description and Priority

All users of the app should be able to CRUD products. High Priority

4.3.2 Stimulus/Response Sequences

A user must select “manage products” to view the options to CRUD products.

4.3.3 Functional Requirements

REQ 1: UI must be functional

REQ 2: Application must be connected to MongoDB

REQ 3: User must authenticate

REQ 4: Fields must validate

## Invoicing

4.6.1 Description and Priority

All registered users of the app should be able to CRUD invoices. High Priority

4.6.2 Stimulus/Response Sequences

A user must select “manage invoices” to view the options to CRUD invoices.

4.6.3 Functional Requirements

REQ 1: UI must be functional

REQ 2: Application must be connected to MongoDB

REQ 3: User must authenticate

REQ 4: Fields must validate

## Reporting

4.6.1 Description and Priority

All admins of the app should be able to request accounting reports. High Priority

4.6.2 Stimulus/Response Sequences

An admin must select “manage reports” to view the options for reports.

4.6.3 Functional Requirements

REQ 1: UI must be functional

REQ 2: Application must be connected to MongoDB

REQ 3: Admin must authenticate

REQ 4: Fields must validate

Appendix B: Analysis Models

User Stories

Registered User:

A registered user can log in to their account.

A registered user can logout of their account.

A registered user can CRUD product categories.

A registered user can CRUD products within categories.

A registered user can CRUD invoices.

Registered Admin:

A Registered admin can login to their account.

A Registered admin can logout of their account.

A registered admin can CRUD users.

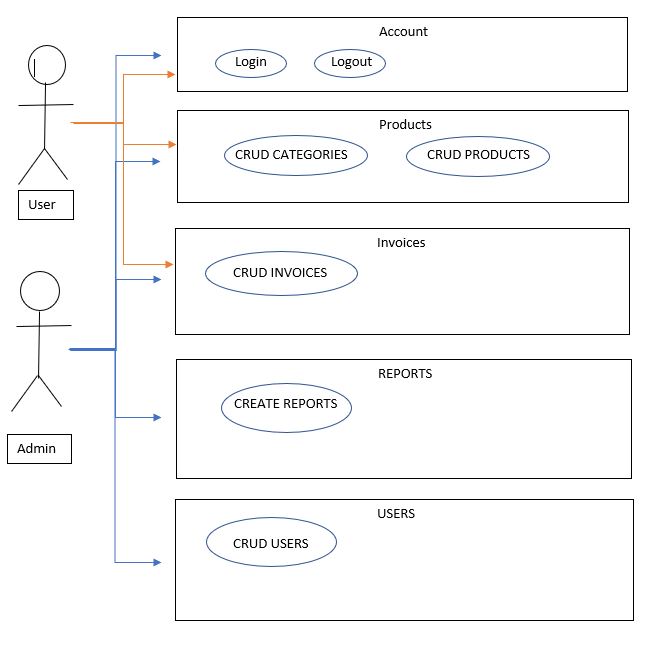
A registered admin can CRUD product categories.

A registered admin can CRUD products within categories.

A registered admin can CRUD invoices.

A registered admin can request reports.

Use Case Diagrams



Appendix C: Screen Collections

Splash / Login Screens

Dashboard Screens

Product Screens

Product Brand Screens

List Brands

Create Brands

Edit Brands

Delete Brands

Product Category Screens

List Brand Categories

Create Brand Category

Edit Brand Categories

Delete Brand Categories

Product Item Screens

List All Items

List Items by Brand and Category

Create Item by Brand and Category

Edit Item by Brand and Category

Delete Item by Brand and Category

Invoice Screens

List of Open Invoices Ordered by Date

List of Closed Invoices for the Month

Create Invoice

Print Invoice

Edit Invoice

Delete Invoice

User Screens

List Users

Create Users

Edit Users

Delete Users

Report Screens

Product Reports

Sales Reports

-weekly

-monthly

-quarterly

-yearly

Usage Reports