**Framework for apps and hardware in Raspberry Pi’s**

**Written by: Dylan Westra**

**Date: 11-07-2019**

**Version 1**

# Introduction

# Index

# Summary

Operating systems like Android and IOS have a framework that makes deploying apps simple. This makes those operating systems capable of easily connecting the hardware to the software without too much trouble. However until this day there isn’t such a system that comes close for the Raspberry Pi. In theory Android could work on a raspberry pi, but it’s far too heavy to run properly and to start developing apps for Android is a big learning curve for others. For this research a Raspberry Pi with Raspbian (Debian Linux based) operating system will be used. Raspbian is being developed by the Raspberry Pi creators themselves and has many utilities available for developing which, will be useful for the library’s created.

## Goal

The goal is to make a container in which to run apps on a server and client side. The server is always a Raspberry Pi and for now for the client we’ll use an apache2 server on the raspberry pi. This will result in our client library’s and apps written in web technology languages (JS, PHP, HTML, etc.) which most software engineers can work with and also reaches less experienced developers. Because the client container is running on an apache2 server, the client can be reached from any device with a web browser. The apps in the client container can be a standalone apps without interference from the server. The server sided app container will be written in Python3, which is the main language used for hardware control in Raspbian. Only in the server apps should one manage the hardware for the Raspberry Pi. The connection between the server and client containers will be made by a web socket connection for communication. There will be a few stock apps already implemented: Audio control, Screen brightness control (for the official Raspberry Pi Screen) and a simple music player. On the client side there has to be a container to automatically add any apps installed and has to be able to close or open apps.

## What not to expect

To reduce developing time some features which could be useful will not be added

- Notification bar

- multiple clients (e.g. android app)

## Alternatives

As alternative an X / wayland window manager could be used instead of a web client. In this way every app would run native inside the operating system. The reason this isn’t picked is because of the development time required for X.

# Server side

For the server side we will make use of Python3. Python is an easy understandable language gaining more popularity throughout the years. For this reason the apps will be written in python.