Table of Contents

Functional Specifications for Android Task Automation App

1. Introduction

1.1 Overview

The application is a task automation application for Android devices. It will allow users to write tasks using Javascript, and these tasks are automatically performed on the user's device when specific facts are met. The application's purpose is to simplify everyday actions a user would normally perform manually on their device, i.e turning their phone to silent mode while at work or in a meeting. What makes this application different than other similar applications is that it allows the user to write their own task scripts from scratch and download them to their device, rather than being limited to default tasks already created for them. This would be attractive to those who already have coding knowledge such as software developers. We aim to make the application universal to all of the user's Android devices. The user will be able to download individual or all tasks to any of their devices, and then control which tasks operate on each device using an on/off toggle. The user will upload their scripts to their own webspace, and then enter the URL into the application on their device to download the task.

1.2 Product Context

The reason for the development of this application is for both personal benefit, and with the hopes of creating a beneficial service to others. The sole purpose of this application is to simplify the user's everyday tasks. We want the user to trust the application to perform the basic everyday tasks automatically. We also want to give them the ability to use their own coding skills to create their own personalised task scripts. We want to expand our software developing skills, as well as gain new skills such as Javascript interpreting, and application development. Android applications are commonly created to be uploaded to the Google Play Store, and downloaded by other users for a small price. Therefore, financial gain is also a possible context for this application should we consider to continue to develop it further.

1.3 Glossary

Android - Mobile operating system developed by Google.

Android SDK - Android Software Development Kit, developer tool used to create applications for Android devices.

Javascript - A high-level, dynamic programming language of Web content production.

URL - Uniform Resource Locator, a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it.

JavaScript Interpreter - Executes lines of JavaScript code in a way for it to be used with Java and Android SDK.

Web Server - A computer system that processes requests via the basic network protocols to distribute information on the internet.

2. General Description

2.1 Product / System Functions

The general functionality that our application will provide is as follows:

- Create profiles
- Download scripts
- Display system statistics
- Interpret and run JavaScript scripts in the background
- Enable/disable profiles
- Change key variables for any script
- View scripts

2.2 User Characteristics and Objectives

Since our application requires the use of scripts written in JavaScript which can be written by the user themselves, we expect the majority of our users to be familiar with writing code in this language. Therefore, it can be said that the majority of our users are likely to be developers or computer programming students/hobbyists and are likely to be aged 13 and older.

However, the scripts do not have to be written by the user and our application will provide some starter scripts by default and the ability to change the variables of any script downloaded without editing the code itself directly. We believe these features will expand our user base to also include those who may not be as familiar with JavaScript programming or programming in general.

The target audience is currently used to manually adjusting their device using the Android system's settings. Our aim is to simplify this using triggers to adjust the settings automatically.

From a user's perspective, the following would be desirable features and help to form a "wish list" of sorts for our system:

- Aesthetically pleasing view of which profiles are currently on/off
- Straightforward process of downloading task scripts
- Dynamic and reactive interface when performing actions on the application
- Editing JavaScript files within the application
- Ability to communicate with all of the device's hardware when writing scripts
- An in-application tutorial showing the user how to navigate the app

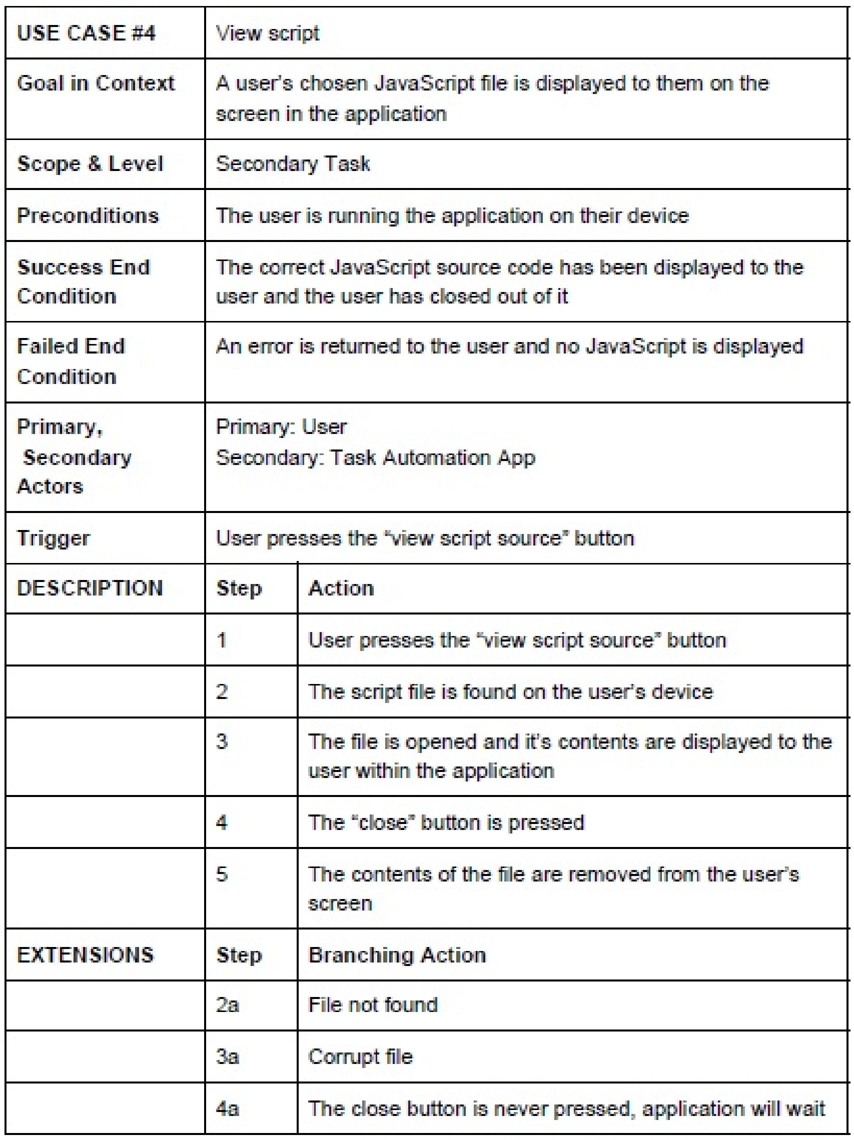
2.3 Operational Scenarios

The following are use cases for our application:

USE CASE #1	Download script	
Goal in Context	A user downloads a script to their device from a URL which they provide to the application.	
Scope & Level	Primary Task	
Preconditions	The application is running on the user's device which is connected to the internet. There is a JavaScript file to be found at the URL which is provided by the user.	
Success End Condition	The script has been downloaded to the user's device and is available to be used in the application.	
Failed End Condition	An error is returned to the user and nothing is downloaded onto their device.	
Primary, Secondary Actors	Primary: User Secondary: Internet, Task Automation App	
Trigger	User entering a URL into search bar and hitting "Download" button.	
DESCRIPTION	Step	Action
	1	A URL is entered by the user.
	2	"Download" button is pressed.
	3	The script is fetched from the given URL.
	4	The script is stored on the user's device.
	5	The script is made available to the user in application.
EXTENSIONS	Step	Branching Action
	3a	Invalid URL is entered
	3b	URL is unreachable (does not exist or no internet connection)
	3c	File found at URL is not written in Javascript
	4a	Insufficient space on device to store script

USE CASE #2	Create profile	
Goal in Context	A user creates a profile within the application	
Scope & Level	Primary Task	
Preconditions	The user is running the application on their device and is in the "Profiles" section	
Success End Condition	A profile has been created	
Failed End Condition	An error is returned to the user and no profile is created	
Primary, Secondary Actors	Primary: User Secondary: Task Automation App	
Trigger	User clicks the "new profile" button	
DESCRIPTION	**Step**	**Action**
	1	"New profile" button is pressed
	2	The "create new profile" interface is displayed on the screen
	3	Details are entered by the user
	4	The "create profile" button is pressed
	5	The profile is created
	6	The profile appears in the list of profiles available
EXTENSIONS	**Step**	**Branching Action**
	5a	Invalid details are entered

USE CASE #3	Enable profile	
Goal in Context	A user enables an existing profile within the application	
Scope & Level	Primary Task	
Preconditions	The user is running the application on their device and is in the "Profiles" section	
Success End Condition	A profile has been enabled	
Failed End Condition	An error is returned to the user, the profile is not enabled and no scripts are run	
Primary, Secondary Actors	Primary: User Secondary: JavaScript Interpreter, Task Automation App	
Trigger	User toggles a profile "on/off" button to the "on" position	
DESCRIPTION	**Step**	**Action**
	1	User toggles a profile "on/off" button to the "on" position
	2	The scripts contained within the selected profile are found on the device
	3	These scripts are loaded from the device and passed to the JavaScript Interpreter
	4	The red "profile off" indicator is set to green for the selected profile
EXTENSIONS	**Step**	**Branching Action**
	2a	Script cannot be found
	3a	Invalid script
	4a	Error occurred, indicator remains red and "on/off" switch toggles back to "off". Error message displayed to user.



USE CASE #4	View script	
Goal in Context	A user's chosen JavaScript file is displayed to them on the screen in the application	
Scope & Level	Secondary Task	
Preconditions	The user is running the application on their device	
Success End Condition	The correct JavaScript source code has been displayed to the user and the user has closed out of it	
Failed End Condition	An error is returned to the user and no JavaScript is displayed	
Primary, Secondary Actors	Primary: User Secondary: Task Automation App	
Trigger	User presses the "view script source" button	
DESCRIPTION	Step	Action
	1	User presses the "view script source" button
	2	The script file is found on the user's device
	3	The file is opened and it's contents are displayed to the user within the application
	4	The "close" button is pressed
	5	The contents of the file are removed from the user's screen
EXTENSIONS	Step	Branching Action
	2a	File not found
	3a	Corrupt file
	4a	The close button is never pressed, application will wait

2.4 Constraints

- **Javascript Interpretation:** The biggest constraint for this project is the implementation of the JavaScript interpreter which our application will use. It is essential to implement this correctly at an early stage in the development because of how critical it is to the rest of our application.
- **Device hardware access:** Scripts written for our application will not be able to access certain hardware of the device they operate on. This is due to the our application not having root privileges on the device and also that we will not have the time to include functionality for all of the available hardware that modern devices have.
- **Time:** The Functional Specification deadline is the 2nd December 2016. The project deadline is March 2017. These deadlines of course limit our time to work on the project and any documentation to go with it.
- **Device storage space:** The amount of scripts which can be used by our users will depend on their available storage space. This is due to the requirement that each script be downloaded onto their devices local storage in order to be interpreted and run in the background. In general, we expect most scripts to be relatively small compared to the storage space available on modern mobile devices so this shouldn't be a massive issue.
- **Internet connection:** Slow internet or no internet connection at all will affect the applications ability to download task scripts from the user's webspace. We will need decent connection to create and test the application also.

3. Functional Requirements

3.1 Downloading Task Scripts

Description: This function is the process of downloading a task script from a webspace to the users device. A URL is entered into the application and the script is fetched and stored to the device. It is made available to the user to be used in a profile.

Criticality: This function is essential for the running of the application, as the task scripts contain the code that allows the device to perform the desired tasks.

Technical Issues: The main issue with this function is to ensure the URL is valid, and that there is data available where the address is pointing to.

Dependencies with other requirements

None.

3.2 Interpreting & Running Scripts

Description: The application parses JavaScript scripts and interprets them using its JavaScript interpreter. It then generates android-compatible Java code based on these instructions which can interact with the Android device's hardware and peripherals directly.

Criticality: This is the most critical function to the application because it encompasses the core functionality the application provides. Without this function, none of the other functions would serve a purpose.

Technical Issues: JavaScript cannot communicate directly with Android, it must first be translated into Java, which can. A JavaScript interpreter will be used to overcome this issue. Also, checks must be in place to ensure that the script does not pose a security risk to the user's device.

Dependencies with other requirements

- *Downloading Task Scripts* : A script will need to be downloaded by the application before this function will take place as it operates on scripts.
- *Profile Creation* : A profile will need to be created by the user which contains a downloaded script in order for that script to be run.
- *Profile Enabling* : A profile created by a user which contains at least one script will need to be enabled in order for this functionality to take place.

3.3 Profile Creation

Description: This is the creation of individual profiles that can be made up of one or more task script. While active, the profile allows the chosen tasks to run in the background on the device. Profiles allow the user to control when particular tasks are active.

Criticality: Some users may prefer to have all their task scripts running altogether at all times. However the profiles feature gives them the control to turn some tasks on or off without having to constantly delete or redownload the task script on their device.

Technical Issues: The profile is mainly just an indication which tasks are currently running in the background. The only issue with creating a profile would be if there were no scripts downloaded to be used by the profile.

Dependencies with other requirements

- *Downloading Task Scripts* : At least one script needs to be available for a profile to be created.
- *Interpreting & Running Task Scripts* : The task scripts being used in the profile need to be interpreted for them, and the profile, to work.

3.4 Profile Enabling

Description: A profile is set to run by the user using an on/off switch by setting it to the “On” position. This triggers the interpreting and running of the scripts contained in that profile in the background on the user’s device.

Criticality: This function is essential to the application because without the enabling of profiles, no scripts would ever actually be run.

Technical Issues: Scripts contained in enabled profiles need to be capable of running in the background on the user’s device.

Dependencies with other requirements

- *Profile Creation* : A profile will need to be created by the user in order for it to be enabled

3.5 Changing Script Variables

Description: This function gives the user the ability to edit variables of some tasks in order for them to suit the user’s needs better. This could be entering a particular location for the device’s GPS to track, for example.

Criticality: As not all tasks will have changeable variables, and there is the option to write multiple scripts per variable, this function is not highly critical. However it would be more efficient and user friendly to have this option available.

Technical Issues: A possible issue could be invalid user input, but this could be handled with error messages or suggestions.

Dependencies with other requirements

- *Interpreting & Running Task Scripts* : The task scripts being edited need to be available on the device and interpreted correctly for the variables to be changed and ran.

3.6 Profile Deletion

Description: A profile which is no longer needed/wanted by the user can be deleted from the profiles screen. The scripts used in that profile remain available for use in new/other profiles but the profile itself is removed.

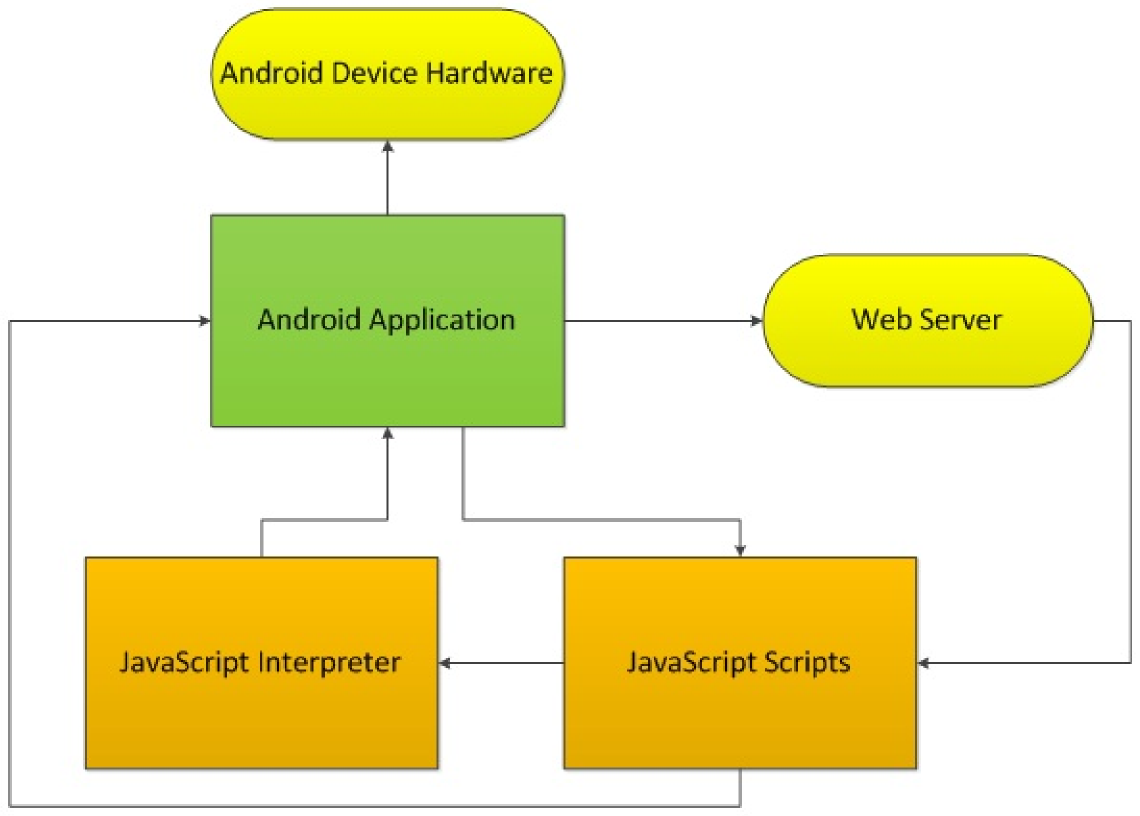
Criticality: This function is not as essential to the core functionality of the application as Profile Creation or Profile Enabling because profiles do not need to be deleted in order for scripts to be run. However, the user may want to delete profiles as they see fit.

Technical Issues: None.

Dependencies with other requirements

- *Profile Creation* : A profile will need to be created by the user in order for it to be deleted.

4. System Architecture



The diagram above illustrates the architecture of the system and the relationships between the system modules. The Android Application is the main module. JavaScript Interpreter and Javascript Scripts are secondary modules. The Web Server and Android Device Hardware are both Third Party Modules.

4.1 Android Application

The Android Application is the main interface the user comes into contact with. It is the way the user gains control of their task scripts. It should therefore be user-friendly, efficient, fast and aesthetically pleasing.

4.2 JavaScript Scripts

These are scripts written in JavaScript which are stored in the file system of the user's device for use with the application.

4.3 JavaScript Interpreter

The JavaScript Interpreter acts as a translator which is able to interpret JavaScript at runtime and convert it into relevant Java code which will be able to interact with the Android device hardware.

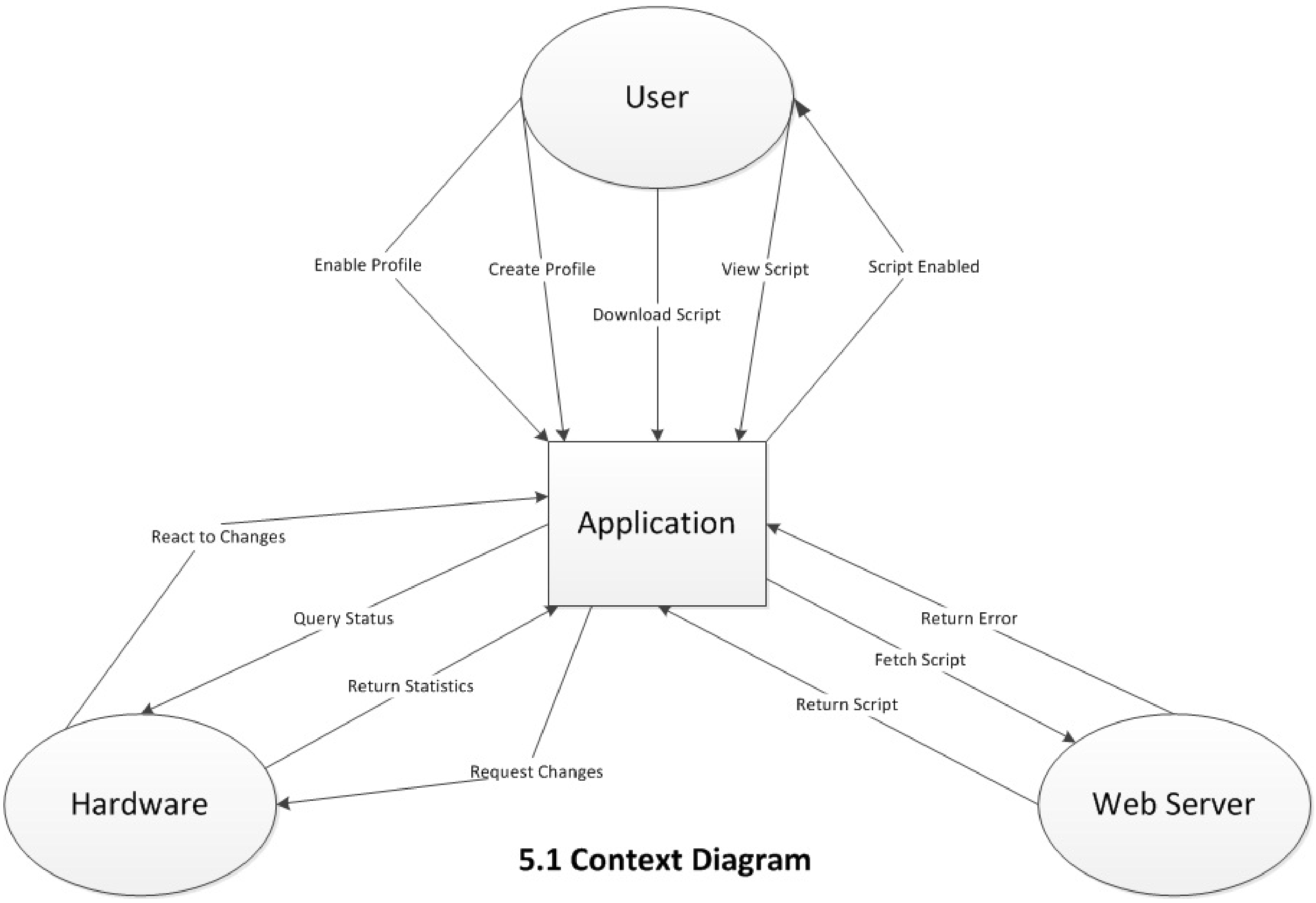
4.4 Web Server

This is a third-party device which our application will communicate with when fetching scripts from the Internet.

4.5 Android Device Hardware

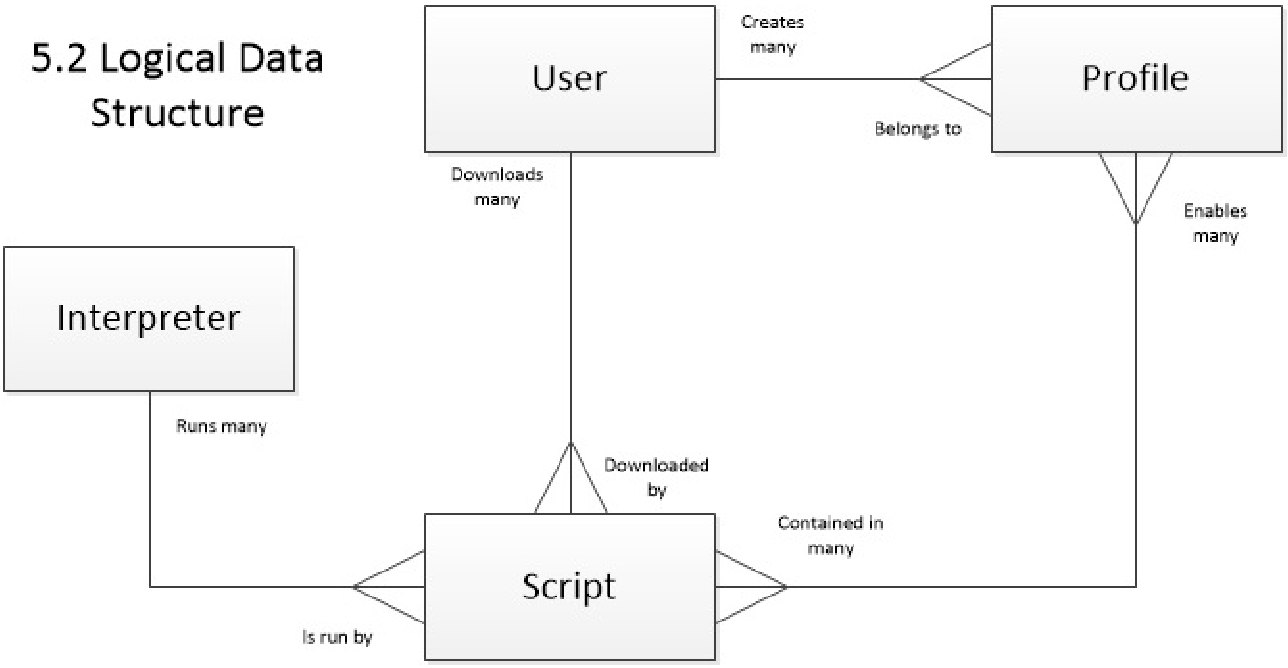
This represents the third-party hardware which the user's device contains. Our application will be able to interact with this hardware.

5. High-Level Design



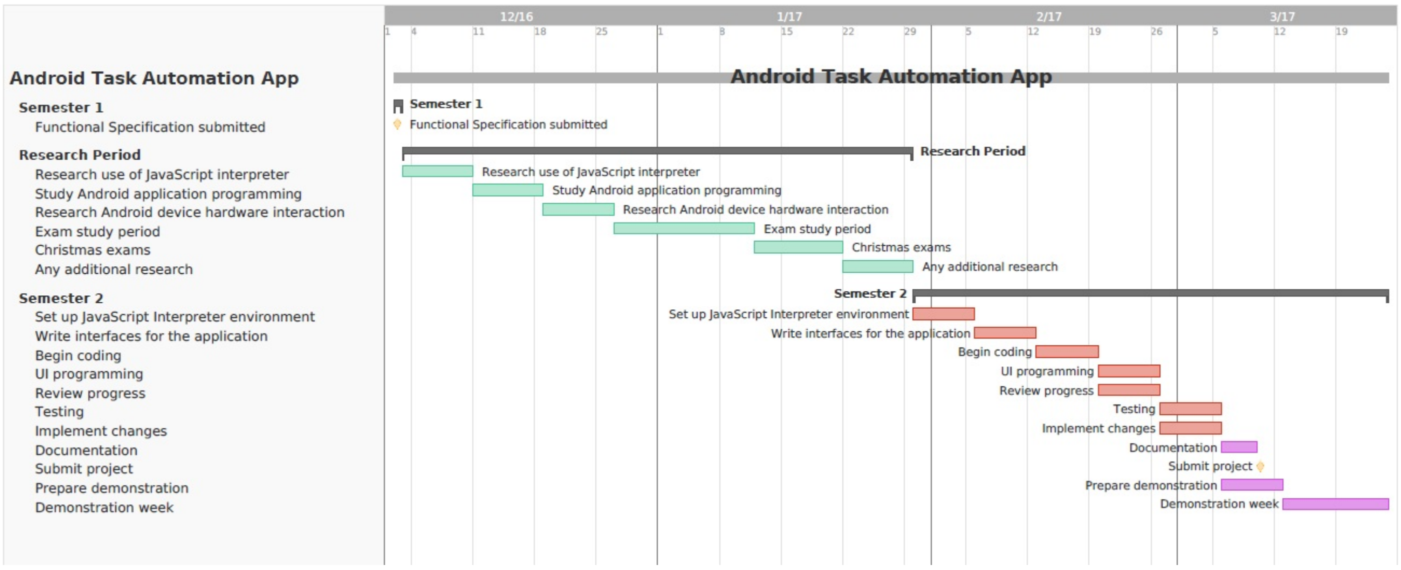
5.1 Context Diagram

5.2 Logical Data Structure



6. Preliminary Schedule

6.1 Gannt Chart



7. Appendices

https://www.android.com/

https://www.javascript.com/