PiClock

Functional Test Plan

Jacob Alspaw
Arbazuddin Ahmed
Kareem Taleb
Benjamin Young

An open source, smart alarm clock platform that aims to improve upon the functionality of traditional alarm clocks by integrating new and useful technologies to motivate users. PiClock increases productivity by helping users overcome their early morning exhaustion and providing general information about their upcoming day.

Version 1.3

November 9, 2018

History of Changes

Version	Description of change(s)	Initials	Date
1.0	Initial draft	BY, JA, KT, AA	11/04/2018
1.1	Removed Simon game due to time constraints	BY, JA, KT, AA	11/05/2018
1.2	Modified document to more closely resemble examples	BY, JA, KT, AA	11/08/2018
1.3	Added testing responsibilites	BY, JA, KT, AA	11/09/2018

Table of Contents

1. Introduction

1.1 Importance of the Document

The Functional Test Plan document specifies the approach this project will take to test all aspects of its software. It outlines the testing that is needed to give the user the best possible experience with this software. It provides an overview of several kinds of testing and provides a detailed description of the functional testing plan, which ensures that the PiClock's actual behavior matches its requirements. Formally recording a testing plan in a document decreases the likelihood that the overall test suite will overlook a critical component of the system because such a document can be checked for completeness. This document is specifically intended to provide guidance for the testers of the PiClock and a reference for the PiClock's developers if the results of any tests necessitate modifications to the application.

1.2 The General Need for Testing

Testing is an essential aspect of the development of every software system. Casual tests run by developers as they write code are insufficient to determine whether the system performs reliably. Developing a formal test suite to test all of a system's components and their interactions is not guaranteed to find all of the system's faults, but it goes a long way towards making sure that the system functions properly for most users. Testing is the best method of alerting developers to faults in the software and of determining whether the product is ready for release to the public. In regard to the PiClock and in other systems whose behaviour depends heavily on user interaction, testing can determine certain user interaction patterns which cause faults, and the system can be modified to address this behavior accordingly.

1.3 How Testing Applies to the PiClock

The PiClock is a complex system consisting of components which have complex interactions with each other. As such, thorough testing is critical to ensuring that the PiClock functions properly in every scenario. Furthermore, as a result of the games, widgets, and customizable settings presented to the user, the PiClock relies heavily on user interaction, so many variations of user interaction must be simulated to guarantee correct behavior. The PiClock performs the important function of waking the user up in the morning, and a failure to perform this function correctly can result in disastrous consequences if the user misses an important early-morning event. Thorough testing decreases the odds that this failure will occur.

1.4 References

PiClock Software Requirements Specification, 23 September 2018.
*PiClock Software Design Document, 1*7 October 2018.

2. Test Design Specification

2.1 Functional Testing

The functional testing approach is described in detail in section 3. The PiClock will be tested to ensure that it fulfills the requirements in the Software Requirements Specification.

2.2 Compatibility Testing

Compatibility testing ensures that the PiClock app functions successfully on different platforms. The PiClock should function regardless of which operating system or Raspberry Pi model the user chooses to run it on. In particular, the clock and system usage statistics widgets use bash system calls whose functionalities can vary depending on the operating system on which they are run. The PiClock app will be run on several common Raspberry Pi operating systems to ensure it functions properly.

2.3 Stress and Performance Testing

Tests will be run to determine whether the PiClock meets the performance requirements specified in the Software Requirements Specification document. Testers will play games, view the slideshow, add new widgets to the slideshow, open settings, and register whether these operations took an acceptable amount of time. Additionally, testers will attempt to create unusual scenarios and register how the PiClock performs under these circumstances. For example, the tester could set a large number of alarms and check whether this affects the latency of any individual alarm being triggered.

2.4 Regression Testing

Regression testing is done after new changes are implemented into a system. If all the components that was previously implemented and tested, as well as the new components, are tested and all the tests pass, the system is working as it should, and the new functionality has not affected the system. However, if this is not the case, and a test or tests fail, then the system has ‘regressed’ and changes have to be made to the components to ensure that the new functionality works.

2.5 Unit Testing

Unit testing will be performed on new methods immediately after they are written using the Catch2 unit testing framework for C++. Using this framework, we will test if all methods accurately and consistently output the correct data given any input.

2.6 User Interface Testing

UI testing will be performed using the OpenHMITester open-source GUI testing tool. This tool will be used to capture a user’s interaction with the PiClock via taps on the touchscreen. The PiClock’s response to user actions, such as selecting an answer in a game or tapping a button to change a setting, will be noted and compared to the expected response.

2.7 User Acceptance Testing

The PiClock will be installed on a Raspberry Pi and presented to individuals who have agreed to test it. Users doing testing will be instructed to use the PiClock as they would regularly, so that the testing occurs under realistic conditions. They will also be instructed to attempt to use all the different components. They will play all the games, change settings, watch the widget slideshow, and set alarms. Ideally, they will use the PiClock with its intended functionality as an alarm to wake them up in the morning. They will then provide feedback on any observed unusual behavior and on the overall design.

3. Functional Test Specifications

Functional tests ensure that the PiClock's behaviour satisfied the functional requirements listed in the PiClock Software Requirements Specification document. For convenience, these requirements are shown below:

- F01: Basic clock with time and date, present on every screen
- F02: Alarm rings at time specified by user and continues ringing until user completes a task or failsafe is triggered
- F03: User plays a simple game to shut off the alarm
    - F03.1: Tic-Tac-Toe
    - F03.2: Concentration memory game
    - F03.3: Trivia game
    - F03.4: Math game
- F04: User can view the current weather conditions on home screen
- F05: User can view the week's weather forecast on home screen
- F06: User can view a random motivational quote on home screen
- F07: User can view times in different cities around the world
- F08: User can observe how many days until next major holiday
- F09: User can view Raspberry Pi's system statistics
- F10: Home screen rotates continually rotates between F04 - F09
- F11: User can adjust application settings to change which game is to be completed when alarm rings, game difficulty, which applications appear on home screen, and API configuration

Each row in the following tables lists which functional requirement the test aims to verify, a brief description of the test, and the expected outcome of the test.

3.1 Games and Alarm

SRS Requirement	Test Description	Expected Outcome
F02	Set an alarm for a time *t*, wait until time *t*.	Alarm rings at time *t*.
F02	Set an alarm for time *t*, wait until alarm rings at time *t*	A game is launched immediately after alarm begins ringing
F02	Play game launched after an alarm rings	Alarm does not stop ringing until game is completed or 5-minute time limit is reached
F03	Accumulate points greater than the number of points required by the game difficulty to complete the game	Game is completed, alarm shuts off, and app displays home screen
F03.1	Place an X on tic-tac-toe board which results in 3 Xs in a row	User wins the game and user's point total increases by 2
F03.1	Computer places an O on tic-tac-toe board which results in 3 Os in a row	Computer wins the game and user gains no points
F03.1	A player places a mark on the last empty square on the board and no player has won the game	Game is a tie and user's point total increases by 1
F03.1	Place an X in a square which does not give a decisive result and there are still open squares on the board	Computer places an O on an empty square on the board
F03.2	Begin concentration game	Board consists of a 4x2, 4x3, or 4x4 array of face-down cards on beginner, intermediate, and advanced difficulties, respectively
F03.2	Flip one card	Game waits for you to flip second card
F03.2	Flip a second card which matches first card	Cards remain face-up
F03.2	Flip a second card which does not match first card	Cards flip back over
F03.2	All cards are flipped face-up	Game is completed, alarm stops

F03.3	Launch trivia game / proceed to new question	User is presented with a trivia question with 4 potential answers. Question has not already appeared during this game
F03.3	Click incorrect answer	User informed that answer was incorrect, no points added to point total, game presents next question
F03.3	Click correct answer	User informed that answer was correct, 1 point added to point total, game presents next question unless point total exceeds threshold
F03.4	Launch math game / proceed to new question	User is presented with a math question of the form “<integer> <operator> <integer>?” where integers are within the range [-25, 25] and answers are within the range [-500, 500] and operator is one of {+,-,*,/}. Question has 4 potential answers and has not already appeared during this game.
F03.4	Click incorrect answer	User informed that answer was incorrect, no points added to point total, game presents next question
F03.4	Click correct answer	User informed that answer was correct, 1 point added to point total, game presents next question unless point total exceeds threshold

3.2 Widgets and Clock

SRS Requirement	Test Description	Expected Outcome
F01	Compare time and date displayed by PiClock's clock to actual time and date	PiClock's clock time and date matches actual time and date
F01	View every screen which can be displayed by the PiClock	Every screen displays the clock
F04	View the widget for the weather	The weather updates every rotation it

		is shown in the slideshow The weather is accurate to the city selected in the settings The widget displays the weather and relevant information Once the weather is shown for the set amount of time, the next widget in the slideshow is displayed
F05	View the widget for the forecast of the week	The forecast updates every rotation it is shown in the slideshow The forecast is accurate to the city selected in the settings The widget displays the forecast for the whole week of the set city Once the forecast widget is shown for the set amount of time, the next widget is displayed
F06	View quotes widget	The quote is changed every rotation it is shown in the slideshow Once the widget is shown for a set amount of time, the next widget is displayed
F07	View the widget for the world clock	The time of each city updates every rotation it is shown in the slideshow Accurate time is shown for 5 cities Once the widget is shown for a set amount of time, the next widget is displayed
F08	View the widget for the holiday countdown	The correct number of days until the next major holiday should display for each rotation it is shown in the slideshow If a day passed, the widget updates

		and displays the number as one less than the day before Once the widget is shown for a set amount of time, the next widget is displayed
F09	View system usage statistics widget	All 3 statistics automatically update every second
F09	Compare values displayed for CPU usage, memory usage, and disk usage to actual values obtained by directly accessing Raspberry Pi's system statistics	Values displayed by PiClock match true values
F10	View slideshow on home screen	Slideshow rotates between widgets selected in settings at a constant interval. Each widget is shown exactly once per cycle.

3.3 Settings

SRS Requirement	Test Description	Expected Outcome
F11	Change a setting, save and exit, then reopen settings application	Settings reflect change that was made
F11	Add a widget to slideshow, save settings	Widget now appears in slideshow
F11	Remove widget from slideshow, save settings	Widget no longer appears in slideshow
F11	Change game difficulty, save settings, wait for alarm to be triggered	Game launched when alarm rings has set difficulty
F11	Change which game is to be played when alarm rings, save settings, wait for alarm to ring	Chosen game launches when alarm rings
F11	Change clock between 12 and 24-hour time, save settings	Clock is displayed in specified format

4. Testing Responsibilities

Role	**Responsibilities**	**Name(s)**
Widgets	Run tests on each widget Refactor and fix all code and bugs for each widget	Ben Arbazzudin
Games	Run tests on each game Refactor and fix all code and bugs for each game	Ben Arbazzudin Kareem
Settings	Run tests for system settings Refactor and fix all code and bugs for system settings	Ben
UI	Run tests for the PiClock UI Refactor and fix all code and bugs for system settings	Jacob

5. Inspection Report

The inspection report describes the issues we as a team discovered throughout the construction of our functional test plan and the resolution status of those issues.

Issue	Issue raised on date	Resolution status as of 11/9/2018
Implementing Simon game did not seem feasible in time allotted to build the PiClock	11/4/2018	Resolved - Simon Game removed from design document
Original draft of document did not resemble new example documents posted online	11/7/2018	Resolved - Document modified to more closely resemble examples
Document lacked assignment of responsibilities to project members	11/9/2018	Resolved - Added section 4
Finalized report by checking for grammatical and formatting mistakes.	11/9/2018	Resolved - Grammatical mistakes were removed from document.