# Verification and Validation Plan for Solar Water Heating Systems Incorporating Phase Change Material

Maya Grab

October 29, 2015

## Contents

# 1 General Information

The following section provides an overview of the test plan for Smart Waiter Solutions. This section explains the purpose of this document, the scope of the system, and an overview of the following sections

## 1.1 Purpose

The purpose of this document is to describe various test cases and procedures to evaluate functionality of Smart Waiter. This document is indented to be used as a reference for all future testing.

## 1.2 Scope

This test plan is used to evaluate Smart Waiter functionality. Various plans are described in detail to test expected functional and non function requirements are met.

## 1.3 Overview of Document

The following sections provide more detail about types of test that will be used. Information about the testing process is provided, and the software specifications that were discussed in the SRS document are stated. Testing processes is split up between system test for POC and the final demonstration.

# 2 Plan

This section provides a description of the software that is being tested, the team that will perform the testing, the milestones for the testing phase, and the budget allocated to the testing.

## 2.1 Software Description

The software being tested is a simulator for a SWHS incorporating PCM. Given the physical parameters of the system, including dimensions, properties of the water and PCM,and relevant physical constants, the simulator calculates the changes in temperature and energy of the water and PCM over time.

## 2.2 Test Team

The team that will execute the test cases, write and review the V&VR consists of:

- Maya Grab
- Dr. Spencer Smith
- Thulasi Jegatheesan

## 2.3 Milestones

### 2.3.1 Location

The location where the testing will be performed is Hamilton Ontario. The institution that will be performing the testing is McMaster University.

### 2.3.2 Dates and Deadlines

## 2.4 Budget

The budget for the testing of this system is being funded by McMaster University and NSERC.

# 3 Software Specification

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

## 3.1 Functional Requirements

- Input the physical constants, properties and initial temperatures of water and PCM, and dimensions of the tank
- Verify that the inputs satisfy the required physical constraints
- Compute the calculated values required to solve the governing differential equations
- Calculate the temperatures and energy of water and PCM over time.

## 3.2 Nonfunctional Requirements

Priority nonfunctional requirements are correctness, understandability, reliability, and maintainability.

# 4 Evaluation

This section first presents the methods and constraints that are to be used during the evaluation process. This is followed by how the data obtained by the testing will be evaluated, which includes: how the data will be recorded, how to move from one test to the next, and how to determine if the test was successful.

## 4.1 Methods and Constraints

### 4.1.1 Methodology

### 4.1.2 Extent of Testing

### 4.1.3 Test Tools

### 4.1.4 Testing Constraints

## 4.2 Types of Tests

### 4.2.1 Functional Testing

Functional testing is in place to uncover errors that occur in implementing requirements or design specifications. Concentration is on results rather then the internals functions of the program. This type of testing is very effective to evaluate functional requirements.

### 4.2.2 Structural Testing

Structural testing is in place to uncover errors during implementation of the application. Concentration is on evaluating structure of the application. This type of testing focuses on non functional requirements.

### 4.2.3 Unit Testing

Unit testing refers to providing specific input to a system and verifying it evaluates to expected output.This process can be done manually or automated.

### 4.2.4 Manual and Automatic Testing

Manual testing is done by people while automatic is processed through scripts.

### 4.2.5 Static and Dynamic Testing

Static testing refers to testing techniques that simulate a dynamic environment. This does not involve program execution. Instead, a walk-though through will be performed checking pre and post conditions evaluate to

requirements specified. As well to make sure proper syntax is used thoroughly.This type of testing is crucial in design stage. On the contrary, dynamic testing refers to executing the program while running test cases to view expected behaviour. This is done to find and fix defects in the program. This will be performed after implementation phase.

# 5 POC System Test Description

## 5.1 Barcode Scanning

Smart-Waiter needs to insure users are able to scan a barcode with minimal attempts. The number of expected attempts will be presented in the final SRS document.

### 5.1.1 Test Type

- Manual
- Functional test
- Unit test

### 5.1.2 Test Factors

- Performance
- Correctness
- Ease of use

### 5.1.3 Inputs

- Barcode to be scanned

### 5.1.4 Outputs

- Restaurant menu queried from database

### 5.1.5 Initial State

- Barcode scanning page in empty state

### 5.1.6 Methods of testing

- Dynamic test
- Static test

### 5.1.7 Test Cases

***Functional Unit Test***

*Summary*
Manual black box tests will be performed to assess barcode scanning. This test will be in the form of unit testing. Various test cases described below will be conducted. This is effective as it replicates real world usage and will provide a census of expected number of successful attempts for our knowledge. Performing the

| Test Case | Initial State | Input | Output |
|---|---|---|---|
| 5.1.1 | Barcode Scanning Page – Empty | Eligible Barcode | Restaurant Menu |
| 5.1.2 | Barcode Scanning Page – Empty | Corrupted Barcode | Message prompt reading, "Please try again". After third attempt, prompt will read, "Please contact waiter" and will indicate an option to report bug |
| 5.1.3 | Barcode Scanning Page – Empty | Picture without barcode | Message prompt reading, "Please try again". After third attempt, prompt will read, "Please contact waiter" and will indicate an option to report bug |

***Structural Test***

## 5.2 Database Querying

### 5.2.1 Test Factors

### 5.2.2 Inputs

### 5.2.3 Outputs

### 5.2.4 Initial State

### 5.2.5 Methods of Testing

### 5.2.6 Test Cases

# 6 Final Demonstration System Test Description

## 6.1 Account Login

Smart-Waiter must use accounts to keep track of a user's personal information. The account module has to provide a secure login service.

### 6.1.1 Test Type

Manual, functional dynamic test

### 6.1.2 Test Factors

Correctness, data integrity

### 6.1.3 Inputs

A new user's credentials, with valid information: username, first name, last name, date of birth, email, address, credit card
A new user's credentials, with valid information but a fake credit card
A Google account
A Facebook account

### 6.1.4 Outputs

A message M, containing either a success of failure message, depending on the account information given.
My Account menu
Add Credit Card menu

### 6.1.5 Initial State

Create account menu, empty

### 6.1.6 Methods of testing

Dynamic testing is used to ensure correctness and data integrity, and to observe the application behaviour when given incorrect information.

### 6.1.7 Test Cases

| Test Case | Initial State | Input | Output |
|---|---|---|---|
| 6.1.1 | Create account menu, empty | Username, first name, last name, email, date of birth, address, credit card information | Message prompt reading: “The username you entered is already in use, please choose a different username.” |
| 6.1.2 | Create account menu, empty | Username, first name, last name, email, date of birth, address, credit card information | My account menu |
| 6.1.3 | Create account menu, empty | Username, first_name, last name, email, date of birth, address, fake credit card | Message prompt reading “The credit card information you have provided is invalid, please enter a different credit card.” |
| 6.1.4 | Create account menu, empty | Google account information | Add credit card menu |
| 6.1.5 | Create account menu, empty | Facebook account information | Add credit card menu |

## 6.2 Order Transaction

Smart-Waiter needs to ensure that a user can send in their order, and pay for their order easily and securely. Order transaction will be vigorously tested to ensure complete customer satisfaction.

### 6.2.1 Test Type

Manual, functional dynamic test

### 6.2.2 Test Factors

Correctness, reliability, data integrity, data security, ease of use

### 6.2.3 Initial State

Order Test Cases: Restaurant menu module Credit Card Test Cases: Payment confirmation menu

### 6.2.4 Methods of testing

Dynamic testing is used to ensure validity, record the number of successful tests given a sample.

### 6.2.5 Test Cases

| Test Case | Initial State | Input | Output |
|---|---|---|---|
| 6.2.1 | Restaurant menu module | $n_3$ from set of orders $n_i$ | Message prompt reading: “This order does not follow the restaurant's menu policy, a waiter has been called to the table to assist with the ordering process” |
| 6.2.2 | Restaurant menu module | $n_7$ from set of orders $n_i$ | Order summary menu |
| 6.2.3 | Payment confirmation menu | Valid credit card | Payment receipt menu |
| 6.2.4 | Payment confirmation menu | Expired credit card | Message prompt, reading: “This credit card is expired, please enter valid credit card information” |
| 6.2.5 | Payment confirmation menu | Fake credit card | Message prompt reading: “The information provided is invalid, please provide another credit card” |
| 6.2.6 | Payment confirmation menu | VISA debit card | Payment receipt menu |
| 6.2.7 | Payment confirmation menu | VISA gift card | Message prompt reading “VISA gift cards are not accepted as a valid form of payment with Smart-Waiter, sorry for any inconvenience” |

## 6.3 Usability Testing

The goal of usability testing is to verify if a user can successfully use all functionalities of the application from start to finish in a timely manner. This type of testing aids in evaluating if functional requirements are met. As well it provides scope for evaluating non functional requirements.

### 6.3.1 Test Factors

- Reliability

- Performance
- Correctness
- Ease of use

#### 6.3.2 Inputs

Input will consist of user input for entire app:

- Account login
- Barcode
- Menu items selected
- Credit card details

#### 6.3.3 Outputs

Output will consist of a response for each user input:

- Successful login in response to account login
- Restaurant menu in response to barcode scan
- Order confirmation in response to menu items selected
- Transaction confirmation in response to credit card transaction

#### 6.3.4 Initial State

- User starts at the beginning of the process; account login page

#### 6.3.5 Methods of testing

- Functional unit test

### 6.3.6 Test Cases

***Functional Unit Test***

*Summary*
Manual black box tests will be performed to assess usability of the application. This test will be in the form of unit testing. Ten participants will be selected to evaluate this test. At least five will have some experience with using android applications. The remaining will have little to no experience. Each participant will be asked to conduct a walk through of the application. Number of errors made will be counted and recorded to further evaluate functional requirements and improve test cases. As well, qualitative feedback from participants will be taken in the end. For example, participants will be asked to comment on things they liked and things they didn't like. Also, they will be asked to provide a rating out of 10 in terms of experience; 1 being a poor experience and 10 being a wonderful experience. This feedback will help evaluate non function requirements; user interface appearance, learning requirements and accessibility.

# 7 Automated Testing Plan