4TB6: Verification and Validation Report

Stonecap Solutions - Smart Serve

Max Turek *turekm*
Ryan Were *werer*
Sam Nusselder *nusselds*
Peter Minbashian *minbashp*
David Bednar *bednad*1

April 6, 2023

1 Revision History

Date	Developer(s)	Change
03/08/23	Max Turek Sam Nusselder Ryan Were Peter Minbasian David Bednar	Initial Draft
04/05/23	Sam Nusselder	Updated VnV Report to address issues and feedback from instructors, Tas, and test users. Also ensured this document aligned with preceding documents such as the SRS document and VnV Plan by updating the corresponding requirement numbers in the SRS document and test cases in the VnV Plan.

Contents

List of Tables

2 Symbols, Abbreviations and Acronyms

Refer to Definitions, Terms, Acronyms and Abbreviations table in SRS.

3 General Information

This document serves to detail the results of our Verification and Validation Plan of our system Smart Serve.

3.1 Relevant Documentation

Relevant documentation includes the VnV Plan and SRS document. This provides a detailed explanation of all of our tests that we reference in this report.

4 Functional Requirements Test Results

Most test results are qualitative and not measured, so we are simply saying "Pass" or "Fail" Depending on the observed behaviour of the system during testing. If the test results are quantitative, then the measured result is used to determine if the test was a "Pass" or "Fail."

Test Number	Functional Requirements	Inputs	Expected Outputs	Actual Outputs	Results
ST-FR-ICT-01	ODR2 ODR4 ODR7	Ingredients are all in inventory.	The operator has no notifications on the Web App relating to ingredients being out of inventory. The menu on a user account has all drinks being available.	Web App displays all drinks are available for order.	Pass
ST-FR-ICT-02	ODR7 ODR8	Operator inputs all ingredients available and dispenser location of each ingredient into the Web App.	The ingredients and dispenser location map is updated to match the inputted ingredients and dispenser locations added by the operator. The menu on a user account has all drink combinations possible available.	Web App displays all given drink combinations for multiple combinations of inputed drink ingredients (rum and coke ingredients resulted in rum and coke and rum selection).	Pass
ST-FR-WAUT-01	ODR1	Operator scans QR code with their phone.	Operator successfully reaches the Web App. Web App is functional.	Upon scanning the Web App, the login screen is shown.	Pass

Table 1: Test Results for Functional Requirements (1)

Test Number	Functional Requirements	Inputs	Expected Outputs	Actual Outputs	Results
ST-FR-DOT-01	ODR3	Operator orders a drink on a user account.	Operator can view the order number in the queue of orders.	User is able to successfully order a drink. Queue updates automatically as soon as a drink is ordered.	Pass
ST-FR-CPT-01	MDR3 MDR6	Operator orders a drink on a user account with no cup present.	Smart Serve searches for a cup by doing a full rotation, waiting 8 seconds and repeating this to give the user 3 chances to place their cup.	When no cup is present hardware successfully searches for a cup and doesn't pour without a cup present.	Pass
ST-FR-CPT-02	MDR3 MDR6	Operator supplies a cup into the pouring area while the initial order is still active.	Smart Serve begins making the drink ordered as soon as the cup is sensed. The drink is poured until completion.	Drink is successfully made and no error message is relayed to the web app.	Pass

Table 2: Test Results for Functional Requirements (2)

Test Number	Functional Requirements	Inputs	Expected Outputs	Actual Outputs	Results
ST-FR-DMT-01	MDR2 MDR4 ODR10 ODR11	System is dispensing a drink.	Completed drink made with the correct proportions. The correct user is removed from the queue.	Pump successfully dispenses drink into cup with correct proportions. User is removed from the queue once their drink is complete.	Pass
ST-FR-DMT-02	MDR6 MDR7	Operator orders a drink on a user account while a full cup is present.	Smart Serve checks to see if the cup is full and rejects it as Smart Serve will not pour a drink in a full cup. Smart Serve will continue checking every 8 seconds to see if an empty cup is present to pour into.	The full cup is skipped and the system scans for an empty one.	Pass
ST-FR-DMT-03	MDR2 MDR4 ODR10	Operator adds a cup to the pouring area while scanning from previous test is still occurring.	The system successfully scans the cups for the empty cup. It then pours the drink into the empty cup.	The full cup is skipped still and the drink is successfully poured into the empty cup.	Pass
ST-FR-DMT-04	MDR6 MDR7	Operator orders a drink on a user account while a half full cup is present.	Smart Serve checks to see if the cup is full and rejects it as Smart Serve will not pour a drink in a full cup. Smart Serve will continue checking every 8 seconds to see if an empty cup is present to pour into.	The half full cup is skipped still and the drink is successfully poured into the empty cup.	Pass

Table 3: Test Results for Functional Requirements (3)

5 Nonfunctional Requirements Test Results

Most test results are qualitative and not measured, so we are simply saying "Pass" or "Fail" Depending on the observed behaviour of the system during testing. If the test results are quantitative, then the measured result is used to determine if the test was a "Pass" or "Fail."

Test Number	Functional Requirements	Inputs	Expected Outputs	Actual Outputs	Results
ST-NFR-UR-01	UHR3	Testers are asked to navigate the web page and place an order. Testers will rate their navigating and ordering experience on a scale from 1 to 5: 1 - unusable, 2 - poor, 3 - satisfactory, 4- good, 5 - excellent.	The average score from the testers in the survey is greater than 3	Testers rate that their navigation and ordering experience are all 3 or higher	Pass
ST-NFR-UR-02	UHR1	Testers are asked to grab a cup. Testers will report if the machine was too high, too low, or fine.	More than half of the testers say the height is fine.	Testers unanimously agree that the machine height is desirable	Pass
ST-NFR-PR-01	PR1	Operator orders a drink on a user account	The drink is made and ready within 60 seconds of input.	Drink is ordered and available to customer within 50 seconds.	Pass
ST-NFR-PR-02	PR2	Testers are asked to navigate the Web App and perform many different operations. Testers will rate the responsiveness of the webpage on a questionnaire with a scale from 1 to 5: 1 - unusable, 2 - slow, 3 - satisfactory, 4- fast, 5 - instant.	The average score from the testers in the survey is larger than 3	9 testers consisting of friends and family of the smart-serve team members performed the outline task. The test results consisted of two 5's, one 4, four 3's, and two 2's. The average result was 3 and 1/3. The result was larger than 3 and passing.	Pass

Table 4: Test Results for Non Functional Requirements (1)

Test Number	Functional Requirements	Inputs	Expected Outputs	Actual Outputs	Results
ST-NFR-PR-03	PR3	User initiates test case	The order is added to Smart Serves internal database within 20 seconds of input	All orders were received in the database within the required time frame.	Pass
ST-NFR-PR-04	PR4 PR5	Operator orders a drink consisting of a single type of alcohol mixed without anything on a user account	The drink must contain less than 1.1x the amount of expected alcohol	A total of 12 test runs were conducted where all drinks were under the limit.	Pass

Table 5: Test Results for Non Functional Requirements (2)

6 Comparison to Existing Implementation

This section is not appropriate for this project.

7 Unit Testing

This section is not applicable for this project. The important boundary test cases were included in the above tests.

8 Changes Due to Testing

Following our Revision 0 demo, the feedback we received was that the machine needed to have a more streamlined process from start to finish for the end-user. The user should be able to order a drink on the web app with messages/notifications throughout the process until their drink is finished and ready to be retrieved. We have decided to remove the idea of having an operator replenish the empty cup in the turntable between orders and instead give the opportunity to users to order ahead and place them in a virtual queue with waiting times. Once their order is ready to be processed they are alerted to place a cup in the machine. This was done to reduce the extra use cases that would arise if users weren't responsible for placing their own cup along with improving the autonomy of our system. Our testing also, demonstrated that our project was missing communication with the user therefore we have decided to include message boxes/notifications at checkpoints throughout the process to help streamline the order process. This would include a message upon drink selection, during drink creation and finally once drink is complete. Respectively, these would trigger the user to place a cup on the turntable to prepare the drink, prevent any user from ordering a drink during drink creation and finally alert the drink needs to be retrieved upon completion and the next user is available to order.

9 Automated Testing

The code that is used to control the hardware components of the system, will be using PyTest, which helps check for syntax and semantic errors along with verifying certain functionalities of our system. This would include testing the output of the pump sequence, the sensor sequences and the turntable sequence. The Testing tool React Testing Library will be used to automate testing of the Web Application.

10 Trace to Requirements

Test Case	Requirement(s)
ST-FR-ICT-01	ODR2, ODR4, & ODR7
ST-FR-ICT-02	ODR7 & ODR8
ST-FR-WAUT-01	ODR1
ST-FR-DOT-01	ODR3
ST-FR-CPT-01	MDR3 & MDR6
ST-FR-CPT-02	MDR3 & MDR6
ST-FR-DMT-01	MDR2, MDR4, ODR10, & ODR11
ST-FR-DMT-02	MDR6 & MDR7
ST-FR-DMT-03	MDR2, MDR4, & ODR10
ST-FR-DMT-04	MDR6 & MDR7
ST-NFR-UR-01	UHR3
ST-NFR-UR-02	UHR1
ST-NFR-PR-01	PR1
ST-NFR-PR-02	PR2
ST-NFR-PR-03	PR3
ST-NFR-PR-04	PR4, & PR5

11 Trace to Modules

Test Case	Requirement(s)
ST-FR-ICT-01	M2, M3, M5, M6, & ODR7
ST-FR-ICT-02	M2, M3, M5, M6, & ODR7
ST-FR-ICT-03	M6 & M7
ST-FR-DMT-01	M7
ST-NFR-UR-01	M5
ST-NFR-PR-02	M6, M7, M8, & M9

12 Code Coverage Metrics

Code coverage was found for the React web application only. There are some lines in the React code that were modified for the Rev 1 Demo.

File	Branch (percent)	Funcs (percent)	Uncovered Line s
All files	0	0	
src	0	0	
App.tsx	100	0	10-17
index.tsx	100	100	7-19
reportWebVitals.ts	0	0	3-10
src/components	0	0	
Buttons.tsx	100	0	
CategoryTitle.tsx	100	0	9
Drinks.tsx	0	0	
Header.tsx	0	0	14-62
Ingredients.tsx	100	0	6-38
TextInput.tsx	0	0	
src/database	100	100	
data.ts	100	100	10
disps.ts	100	100	1
ings.ts	100	100	1
src/pages	0	0	
Admin.tsx	100	0	9-11
Login.tsx	0	0	10-56
Main.tsx	100	0	6

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection. Please answer the following question:

1. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)

   Max: It seemed initially a plan was good for created structure for when we begin to test, however in theory is not always what ends up happening in reality. We noticed that there were some tests that were being conducted that didn't necessarily support our requirements or they were just no longer applicable to the requirements we were trying to create. We needed to modify some of the test cases and add a few in order to verify all our requirements along with testing for all use cases of our system. These changes mostly occurred due to the fact that we didn't really account for all use cases of our system until we could actually physical test it and allow other stakeholder access our system. In order to anticipate these changes in future projects it would make sense to have other stakeholders that are not a part of building and designing the system to generate and test use cases that potentially our system could do.

   Ryan: Our initial VnV Plan differed quite a bit from the testing that we actually conducted in out VnV Report. When we were conducting the tests, we realized that some of the behaviour that we initially expected was not the most optimal, and we needed to further clarify exactly what behaviour we expected for the different scenarios. This helped to refine some of our functionality. This ultimately changed some test cases. One example of this is as a team we tested the scenarios of our device pouring an order for 0, 1, and two cups with scenarios where they are empty and full. When we tried the scenario with 0 cups, our initial functionality was that we continuously scan for a cup while giving time for a customer to put a cup in. We decided that after a certain point, we should actually stop this and send a message to the user that they need to put in a cup. These changes occurred because we needed to spend more time thinking about our device in its various use cases than we had. In future projects, I think I would be able to better anticipate these as I have a greater appreciation for the planning out of a project after completing most of our Capstone.

   David: Our VnV Plan had a comprehensive list of test cases and scenarios that we wanted to cover to ensure that all requirements were met. However, during the

actual testing phase, we encountered unexpected errors and issues. As a result, we had to modify our test cases and add new ones to cover these scenarios. For example, we discovered that our system did not handle unexpected inputs well, which was not covered in our initial plan. We had to create new test cases to cover these scenarios and modify our existing ones accordingly. These changes occurred because we did not have a full understanding of the potential issues that could arise during testing. To anticipate these changes in future projects, we would need to conduct more thorough risk assessments and have contingency plans in place for unexpected issues that may arise during testing.

Peter: Our initial VnV plan consisted of testing around our project with features that we no longer found optimal as we began developing the project. We found that some of the initial ideas we had could be replaced with simpler and more intuitive solutions. For example, we were planning on having and testing a light indicator to signal to the user when the drink was complete. However, as we began to develop the project more, we realized that the idea was simply not good. Some of our initial tests revolved around this light feature and thus the testing format had to slightly be altered. We chose to alter the tests by having them be more general. For example, the test would simply be that the user is notified that the drink is complete rather than signaling the method by which they are notified. In the future, we can adjust this by not having such specific test cases which are solution-oriented, as opposed to requirement oriented.

Sam: Our VnV plan had to be updated when completing our VnV report. This was because some of the functionalities were adjusted or removed as they weren't as necessary. One example of this was the light feedback system that would show a green light, or red light depending on the status of the system. This was found to be redundant and unnecessary as the user and/or operator can simply be sent a message indicating the status of the system. When performing the test cases we set out in the VnV plan we realized that there were other functionalities we weren't testing for that required more test cases. We didn't anticipate these test cases until we built the physical Smart Serve system. One potential way to anticipate these changes in future projects would be to have other stakeholders or potential users that aren't directly involved in creating or developing the project and test the system to ensure all the different use cases are covered.