Test Plan and Results

Safe Assistant - Alex Vennemeyer, Ethan Hocker, and Brendan Link

Overall Test Plan

The Safe Assistant has numerous subsystems such as Voice Input, Intent Processing, Voice Output, and Networking. Each of these systems will be tested individually to ensure that they are functioning correctly. For the systems that our users will directly interact with, such as Voice Input and Output, we will be primarily focusing on the requirements specification through manual blackbox testing. This will simulate the experience that our users have while using the Safe Assistant. The remaining subsystems such as Intent Processing and Networking are not something the user interacts with or sees directly, so we will test these systems based on their implementation using whitebox testing.

Once the testing of each of these subsystems is complete, we will then test the system as a whole. The Safe Assistant is designed around being a secure, privacy-oriented device, so our team has numerous security tests to ensure that the system's authentication functions work properly. Additionally, one of our goals with the Safe Assistant was to create a platform that performs comparably to other voice assistants on the market. To make sure that our system's performance is adequate, our final set of tests will ensure the speed and uptime of our system meets our standards. Specifically, we want our Safe Assistant to be able to run on a RaspberryPi 4 and be able to run every hour of the day.

Test Cases

VI1.1 Voice Input Test 1: Hotword Detection

VI1.2. This test will ensure the voice assistant can recognize a spoken hotword.
VI1.3. We will speak the hotword and ensure the assistant recognizes that it was spoken.
VI1.4. Input: We will speak the hotword to the voice assistant.
VI1.5. Outputs: The voice assistant will make a notification sound to signal that it detected the hotword
VI1.6. Normal
VI1.7. Blackbox
VI1.8. Functional
VI1.9 Unit

VI2.1 Voice Input Test 2: Speech Input after Hotword

VI2.2 This test will ensure that the voice assistant records speech for an appropriate amount of time after the hotword.

VI2.3 After speaking the hotword, we will check to see that the voice assistant recorded a user's subsequent spoken lines.
VI2.4 Input: Voice lines spoken after the hotword
VI2.5 Output: A file containing the lines spoken after the hotword
VI2.6. Normal
VI2.7 Whitebox
VI2.8 Functional
VI2.9 Unit

VI3.1 Voice Input Test 3: Speech to Text
VI3.2 This test will ensure that voice data captured from the user is transcribed properly
VI3.3 We will speak different voice lines to our assistant and expect a proper transcription with an acceptable average degree of accuracy.
VI3.4 Inputs: Varying spoken lines
VI3.5 Outputs: Transcribed voice lines
VI3.6 Normal
VI3.7 Whitebox
VI3.8 Performance
VI3.9 Unit

VO1.1 Voice Output Test 1: Voice indication is played once hotword is detected
VO1.2 The Safe Assistant must play a voice indication each time that the hotword is detected. This is critical to inform the user that (1) the Safe Assistant heard them and (2) their voice is about to be recorded.
VO1.3 User will speak the hotword and listen for the voice indication
VO1.4 Input: Speaking the hotword
VO1.5 Output: Voice indication
VO1.6 Normal
VO1.7 Blackbox
VO1.8 Functional
VO1.9 Unit

VO2.1 Voice Output Test 2: Voice outputs a task in progress message
VO2.2 To inform the user that the Safe Assistant is still processing an intent once it has received the intent
VO2.3 While a large intent is being processed, the Safe Assistant will inform the user that it is still processing
VO2.4 Input: An large intent
VO2.5 Output: A speech output informing the user that an intent is processing
VO2.6 Abnormal
VO2.7 Blackbox
VO2.8 Functional
VO2.9 Unit

VO3.1 Voice Output Test 3: Voice outputs the results from the executed skill
VO3.2 To provide the user with the information desired as a result of the intent
VO3.3 After an intent has been processed, the Safe Assistant will output the results of the intent
VO3.4 Input: A processed intent
VO3.5 Output: A voice output of the results of the intent
VO3.6 Normal
VO3.7 Blackbox
VO3.8 Functional
VO3.9 Unit

VO4.1 Voice Output Test 4: Voice accent can be altered by the user
VO4.2 To allow the user to change the accent of the Safe Assistant's voice based on personal preference
VO4.3 Once a user changes the Safe Assistant accent, all voice outputs will utilize the new accent
VO4.4 Input: A new voice accent
VO4.5 Results: All voice outputs use new accent
VO4.6 Abnormal
VO4.7 Blackbox
VO4.8 Functional
VO4.9 Unit

VO5.1 Voice Output Test 5: Voice volume can be altered by user
VO5.2 To allow the user to change the volume of the Safe Assistant's voice output
VO5.3 Once a user changes the voice output volume, all voice outputs will be at the new volume setting
VO5.4 Input: A new volume setting
VO5.5 Result: All voice outputs will be at new volume
VO5.6 Normal
VO5.7 Blackbox
VO5.8 Functional
VO5.9 Unit

VO6.1 Voice Output Test 6: Voice speed can be altered by user
VO6.2 To allow the user to alter the speed at which Safe Assistant speaks
VO6.3 Once a user changes the voice speed, Safe Assistant will always speak at the new speed setting
VO6.4 Input: A new voice speed setting
VO6.5 Result: All voice outputs will be at the new speed
VO6.6 Abnormal
VO6.7 Blackbox
VO6.8 Functional
VO6.9 Unit

IP1.1 Intent Processing Test 1: System can parse text input and determine a logical intent

IP1.2 To take the text that was input by the user via speech and logically determine what the user's intent is

IP1.3 Once a user's voice input is converted to text, a logical intent is determined based on the text

IP1.4 Input: Voice input that has been converted to text

IP1.5 Result: Safe Assistant determines what the user's intent is

IP1.6 Normal

IP1.7 Whitebox

IP1.8 Functional

IP1.9 Unit

IP2.1 Intent Processing Test 2: System can handle varying wording for the same intent

IP2.2 To allow the user to make an intent request using varying wording

IP2.3 Varying wording for the same intent will be identified and processed correctly by Safe Assistant

IP2.4 Input: Varying wording for the same intent

IP2.5 Result: Safe Assistant identifies both requests correctly as the same intent

IP2.6 Normal

IP2.7 Whitebox

IP2.8 Functional

IP2.9 Unit

SDK1.1 SDK Test 1: Applications using sdk cannot overwrite system activation phrases

SDK1.2 To ensure new functions don't overwrite built-in system functions

SDK1.3 Safe Assistant will not accept hotwords for new functions that overlap with existing hotwords

SDK1.4 Input: A new function and hotword

SDK1.5 Result: Safe Assistant ensures that the hotword does not overlap

SDK1.6 Normal

SDK1.7 Blackbox

SDK1.8 Functional

SDK1.9 Unit

NT1.1 Network Test 1: On startup, device determines its role in the network

NT1.2 Verify that a voice assistant device successfully determines its role as a client or server from the configuration file on the device.

NT1.3 A mock configuration file will be specified and the startup function will be called. The test will verify that on startup the settings determined by the startup function match the settings given in the config file.

NT1.4 Input: Mocked Configuration File

NT1.5 Output: Data structure tracking settings

NT1.6 Normal

NT1.7 Blackbox
NT1.8 Functional
NT1.9 Unit Test

NT2.1 Network Test 2: On startup, client sends a request to authenticate with server
NT2.2 Verify that a client device will request to connect with the server on startup
NT2.3 The client startup function will be called, and the calls to network infrastructure will be mocked. The Mock objects will track the network calls and verify that the request to connect with the server was made.
NT2.4 Input: mocked client configuration
NT2.5 Output: a call to the mocked network infrastructure
NT2.6 Normal
NT2.7 Whitebox
NT2.8 Functional
NT2.9 Unit Test

SC1.1 Security Test 1: Server prompts client for authentication
SC1.2 Verify that the server requests authentication from clients that connect
SC1.3 A mock client will send a message to the server and verify that an authentication request is received in response
SC1.4 Input: Mocked request to connect message
SC1.5 Output: request to authenticate message
SC1.6 Normal
SC1.7 Blackbox
SC1.8 Functional
SC1.9 Unit Test

SC2.1 Security Test 2: Server issues a token to client on successful authentication
SC2.2 Verify that server marks client as trusted upon receipt of valid credentials
SC2.3 A mock client will send a valid authentication message to the server and mark it as trusted
SC2.4 Input: Client connection request, followed by a valid authentication response
SC2.5 Output: Server authentication Challenge, followed by an authentication success message
SC2.6 Normal
SC2.7 Blackbox
CS2.8 Functional
CS2.9 Unit Test

SC3.1 Security Test 3: Server does not issue a token to client on failed authentication
SC3.2 Verify that server does not mark client as trusted upon receipt of invalid credentials
SC3.3 A mock client will send an invalid authentication message to the server and will not mark it as trusted
SC3.4 Input: Client connection request, followed by a invalid authentication response

SC3.5 Output: Server authentication Challenge, followed by an authentication failure message
SC3.6 Abnormal
SC3.7 Blackbox
CS3.8 Functional
CS3.9 Unit Test

SC4.1 Security Test 4: Server refuses to accept data from clients without a valid token
SC4.2 Verify that server will not accept communication from clients that are not authenticated
SC4.3 A mock client will send voice data to the server without authenticating. The server will not accept the incoming voice data
SC4.4 Input: a message containing voice data
SC4.5 Output: no response from server
SC4.6 Abnormal
SC4.7 Blackbox
SC4.8 Functional
SC4.9 Unit test

FS1.1 Full System Test 1: User gives a request and Safe Assistant provides a response
FS1.2 This test will ensure that all of the software and hardware works together to properly detect voice lines and provide meaningful responses
FS1.3 For this test we will wake the voice assistant with a hotword, ask varying questions or give commands, and ensure that response are meaningful
FS1.4 Input: Hotword, varying spoken questions and commands
FS1.5 Output: Meaningful responses
FS1.6 Normal
FS1.7 Blackbox
FS1.8 Functional
FS1.9 Integration

PT1.1 Performance Test 1: System can answer most requests in a reasonable time frame
PT1.2 This test will ensure that the processing time for voice assistant responses is reasonable
PT1.3 This test will require us to to prompt the voice assistant with different questions and commands and ensure the acreage response time is deemed reasonable
PT1.4 Inputs: Varying spoken questions and commands
PT1.5 Outputs: Voice assistant responses and the time taken to process the outputs
PT1.6 Normal
PT1.7 Whitebox
PT1.8 Performance
PT1.9 Unit

PT2.1 Performance Test 2: System runs 24/7
PT2.2 This test will ensure that our voice assistant runs sustainably for extended periods of time
PT2.3 This test requires the assistant to perform its tasks when asked at random intervals over an extended period of time without a shutdown or performance issues

PT2.4 Inputs: Voice lines given at random intervals
PT2.5 Outputs: Responses given in standard timeframe
PT2.6 Normal
PT2.7 Blackbox
PT2.8 Performance
PT2.9 Unit

PT3.1 Performance Test 3: Safe Assistant can run fully from a Raspberry Pi 4
PT3.2 This test ensures that the Raspberry Pi 4 has enough resources to run Safe Assistant
PT3.3 This test requires us to perform standard Safe Assistant tasks (see FS1.1) and receive results on the Raspberry Pi similar to the results expected on a standard sized computer
PT3.4 Inputs: Varying spoken questions and commands
PT3.5 Outputs: Reasonable responses
PT3.6 Normal
PT3.7 Blackbox
PT3.8 Performance
PT3.9 Integration

Test Case Matrix

	Normal/ Abnormal	**Blackbox/ Whitebox**	**Functional/ Performance**	**Unit/ Integration**
VI1	Normal	Blackbox	Functional	Unit
VI2	Normal	Whitebox	Functional	Unit
VI3	Normal	Whitebox	Performance	Unit
VO1	Normal	Blackbox	Functional	Unit
VO2	Abnormal	Blacbox	Functional	Unit
VO3	Normal	Blackbox	Functional	Unit
VO4	Abnormal	Blackbox	Functional	Unit
VO5	Normal	Blackbox	Functional	Unit
VO6	Abnormal	Blackbox	Functional	Unit
IP1	Normal	Whitebox	Functional	Unit
IP2	Normal	Whitebox	Functional	Unit
SDK1	Normal	Blackbox	Functional	Unit
NT1	Normal	Blackbox	Functional	Unit
NT2	Normal	Whitebox	Functional	Unit
SC1	Normal	Blackbox	Functional	Unit
SC2	Normal	Blackbox	Functional	Unit
SC3	Abnormal	Blackbox	Functional	Unit
SC4	Abnormal	Blackbox	Functional	Unit
FS1	Normal	Blackbox	Functional	Integration
PT1	Normal	Whitebox	Performance	Unit
PT2	Normal	Blackbox	Performance	Unit
PT3	Normal	Blackbox	Performance	Integration