Jarvis Emulator
Software Requirements Specification
COP 4331, Fall 2015

Modification History

Version	Date	Who	Comment
v0.0	10/3/2015	Robin Schiro	Created document
v1.0	10/4/2015	Jimmy Lam	Added more to introduction and product overview, my requirements, and my events to the event table
v1.1	10/4/2015	Robin Schiro	Modified functional requirements so that all tables use a numbered list
v1.3	10/5/2015	Manuel Gonzalez	Added functional requirements specific to speech construction, and sections 3.7, 3.8 and 3.9 of requirements and events specific to speech construction
v1.4	10/5/2015	Julian Rojas	Added the last functional requirements & section 3.3, 3.4, 3.5, 3.6. Also added use case descriptions and my events
v1.5	10/6/2015	Jimmy Lam	Added user to stakeholders
v1.6	10/6/2015	Robin Schiro	Inserted new Use Case Diagram, updated ref doc links
v1.7	10/8/2015	Manuel Gonzalez	Verified that requirement Test Cases reference numbers match with the intended case
V1.8	10/8/2015	Julian Rojas	Grammar corrections. Test Cases verified.

Team Members:

- Jimmy Lam
- Julian Rojas
- Manuel Gonzalez
- Robin Schiro

Table of Contents

1) Introduction

a) **Software to be Produced**

i) The goal of this project is to produce a Windows desktop application called the Jarvis Emulator to enhance the desktop experience. This application can detect and recognize users as they enter or exit the room, and can respond to the users' voice commands though speech construction. It can also perform tasks for users, such as opening other applications, taking a picture of the users, displaying relevant information to the users through website RSS feeds, and logging users out of their computer. The front end GUI will allow the users to set up their profiles, train Jarvis to remember their faces, and configure other settings of the application. It will also display visuals to let the users know it is speaking to the users.

b) **Reference Documents**

i) Project Management Plan

ii) Concept of Operations

iii) Test Plan

c) **Applicable Standards**

i) **Testing Standard**

(1) We will create our test cases to be reasonable enough so that we can fairly evaluate the performance of our application.

(2) We will log our tests to keep track of our progress and make sure that our project is fully functional.

ii) **Coding Standard**

(1) We will write our code in classes so that we can each test our sections of the project and easily integrate our work into one cohesive project later on.

d) **Definitions, Acronyms, and Abbreviations**

i) **Definitions**

(1) Trained user: For all test cases, a "trained user" is one who has provided sufficient training data for his/her profile. This means that at least 50 pictures of his/her face at various angles have been captured by the application.

(2) Active user: The user who currently has control over the application.

ii) **Acronyms**

(1) API: Application Programming Interface

2) Product Overview

a) **Assumptions**

i) We assume that a Jarvis Emulator user will be using a Windows computer. We assume that the user will have an Internet connection for the RSS feeds used by Jarvis. We also assume that the user will have a webcam with a high enough resolution for Jarvis to recognize his face and a microphone with high enough quality to detect his voice correctly.

b) **Stakeholders**

i) **Clients**

(1) Dr. Turgut, our professor: Expects that we finish a presentable project and to be impressed by our work. She also has high hopes that all of her students do well in the course.

(2) Amirreza Samiei, our grader: Overlooks our project and will make sure we meet our requirements for the project. He also hopes that we are able to do well in the class.

(3) Normal Users: The users of the application expect our application to be easy to use and expect that it will improve their desktop experience with easy voice commands and relevant information feeds.

ii) **Developers**

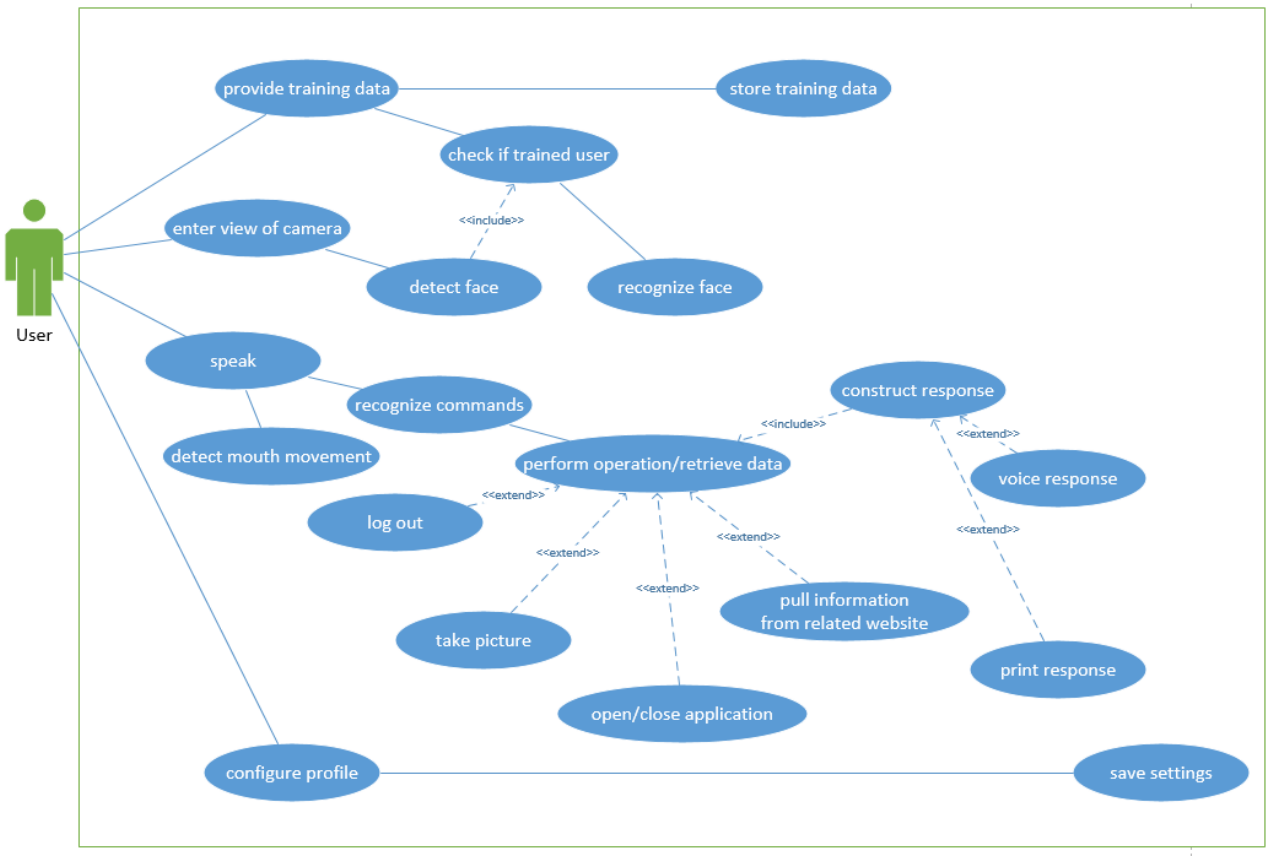
(1) We, the developers are also stakeholders of this project as our grade in the class depends on it. Since we also came up with the project idea, we want to make sure our project works well and impresses the users of this application.

c) **Event Table**

Event Name	*External Stimuli*	*External Responses*	*Internal data and state*
User Recognition	Trained user enters the view of the webcam	The application will greet the user and inquire about what the user would like it to do.	The application will store the name of the active user and wait for commands from this user.
User Inputting Configuration Settings	The user opens the 'Configuration' tab and inputs information into the settings fields.	The application saves inputted settings once the user hits 'Save'.	The application will store the settings in the user's profile file. While the user is inputting information, the application is still listening for commands.
Logout	User voice command	Logs out of computer into login screen	Logs user out and wait for user to return to log in
Open other applications	User voice command	Open desired applications	Searches for the application and open for user
Taking picture	User voice command	Takes picture of user	Uses the webcam to take picture of user and stores it
Data description through Natural Language	User voice commands to output specific data	Speaks to the user with the information requested	Specifically formatted data is fed into the speech construction module for Natural Language Generation. Upon completion, the generated text is output as audio using Text-To-Speech algorithms.
Response to user question	User voice command parsed as a question	Speaks to the user with the answer to the question or that it doesn't know the answer	The speech recognition module will parse the command, detect that is a question, check if it is supported and post to all modules the desired response, which ultimately will lead to feeding information to the speech construction module.
Notification of System Event	An important event happens within the system	Speaks to the user about the event that just happened	Once an event is detected, the speech construction module will generate and output a notification accordingly.

Weather forecast	User voice commands	Application displays/speaks to user	Jarvis will access weather RSS Feed/ API and it will pull information regarding the forecast
News Headlines	User Voice Commands	Application displays/speaks to user	Jarvis will access news website's RSS feed and pull articles headlines. It will give a link to full article
Mouth movement	User's mouth movement within box outline	Detects active user	Application detects who is talking if multiple users are present

d) **Use Case Diagram**



e) **Use Case Descriptions**

- **Enter View Of Camera**
  - User is within 5 feet of webcam.
  - **Exception:** User is not facing camera
- **Recognize Face**
  - Application tries to recognize user with 70% accuracy
  - **Exception:** User has not been trained or lighting conditions are not adequate.
- **Recognize Commands**
  - Jarvis is able to recognize keywords
  - **Exception:** Commands is not already specified.
- **Detect Mouth Movement**
  - Jarvis determine the active user by analyzing mouth movement when detecting faces.
  - **Exception**: No user speaks.
- **Configure Profile**
  - User is able set up profile, add preferences, and applications
  - **Exception:** User has not been recognized
- **User Speaks**
  - The user will communicate with Jarvis using voice commands
  - **Exception:** Jarvis is not able to recognize user's speech
- **Provide Training Data:**
  - User takes different pictures and provide them to the application through user interface
  - **Exception:** User has not been recognized
- **Store Training Data**
  - Jarvis will save user pictures
  - **Exception**: The user is not in view of the webcam; there is no space available.
- **Check If Trained User**
  - Jarvis will check if a detect face is a trained user or a new user
  - **Exception:** Jarvis is not able to detect face
- **Detect Face**
  - Jarvis will analyze frame if face has entered the scene.
  - **Exception:** Environmental conditions prevent detection from occurring (i.e. poor lighting)
- **Perform Operation**
  - Jarvis will take commands as input and complete the related task
  - **Exception:** The command is not supported by the application.
- **Log out**
  - Jarvis will log the user out of his or her computer
  - **Exception**: The user is not logged in
- **Take picture**
  - Jarvis will take a photo of the user and store the picture in a file
  - **Exception**: The user is not in view of the webcam; there is no space available.

- **Open/close application**
  - Jarvis will open and close applications based on user command
  - **Exception**: The desired application is not supported or is not defined in a path for Jarvis
- **Pull information from related websites**
  - Jarvis will use RSS to display to user the information from websites
  - **Exception**: The website is not available or does not support RSS
- **Voice response**
  - Jarvis will perform text to speech of its response construction
  - **Exception**: Jarvis does not construct a response
- **Print Response**
  - Jarvis will display its response to the GUI
  - **Exception**: Jarvis did not construct a response
- **Save profile**
  - Jarvis will save the user's profile to a text file
  - **Exception**: The user did not select to save his or her profile
- **Construct Response**
  - Jarvis will construct a response from a grammar to display to user
  - **Exception**: Jarvis does not recognize user voice

3) Specific Requirements

3.1 Functional Requirements

No: 1	
Statement:	The frames of the feed are processed under eigenanalysis using the OpenCV library. Recognition occurs with a minimum of 70% accuracy.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	Source of algorithm
Evaluation Method:	Set up the application with at most five trained users, with you being one of them. Then, exit and enter the view of the webcam 10 times. The application should recognize you at least 7 of the 10 times.
Revision History:	Robin Schiro | 10/3/15 | Created the requirement

No: 2	
Statement:	The application can track the position of the user's face.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	Source of algorithm
Evaluation Method:	Test Cases 2
Revision History:	Robin Schiro | 10/3/15 | Created the requirement

No: 3	
Statement:	The user interface allows the user to "train" the application for facial recognition.
Source:	Developers
Dependency:	Requirement 2
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Cases 3 and 4
Revision History:	Robin Schiro | 10/3/15 | Created the requirement

No: 4	
Statement:	The user interface allows the user to save and update set of configuration settings based on selections made in the 'Configuration' tab.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Cases 5
Revision History:	Robin Schiro | 10/3/15 | Created the requirement

No: 5	
Statement:	Jarvis shall recognize voice commands of the user with the window's speech library and should have 70% accuracy.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	This YouTube video: https://www.youtube.com/watch?v=KR0-UYUGYgA
Evaluation Method:	Test Case 6
Revision History:	Jimmy Lam | 10/4/15 | Created the requirement

No: 6	
Statement:	Jarvis shall open other applications based on user command.
Source:	Developers
Dependency:	No 1
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 7
Revision History:	Jimmy Lam | 10/4/15 | Created the requirement

No: 7	
Statement:	Jarvis shall log the user out of their computer based on user command.
Source:	Developers
Dependency:	No 1
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 8
Revision History:	Jimmy Lam | 10/4/15 | Created the requirement

No: 8	
Statement:	Jarvis shall take a picture of the user for the user and store the photo.
Source:	Developers
Dependency:	No 1
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 9
Revision History:	Jimmy Lam | 10/4/15 | Created the requirement

No: 9	
Statement:	Given specifically formatted data, the application should generate human language speech that summarizes and describes the data.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 14 and 15
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

No: 10	
Statement:	The application should answer basic user questions.
Source:	Developers
Dependency:	No. 5 and 9
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 11
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

No: 11	
Statement:	The application should greet the user through the speakers upon user recognition.
Source:	Developers
Dependency:	No 1
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 1
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

No: 12	
Statement:	Jarvis should be able to subscribe to Website RSS Feeds.
Source:	Developer
Dependency:	No. 5
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Cases 14
Revision History:	Julian Rojas | 10/5/15 | Created the requirement

No: 13	
Statement:	Jarvis should be able to detect mouth movement.
Source:	Developer
Dependency:	No. 1 & No. 2
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 15
Revision History:	Julian Rojas | 10/5/15 | Created the requirement

3.2 Interface Requirements

No: 14	
Module:	Speech Construction
Input:	Events, Formatted data from websites (RSS feeds),
Output:	Output: Text and audio of natural-language sentences
Data Type:	Input: String Output: MP3
Accuracy:	N/A
Receive/Send Frequency:	Depends on how often the user speaks
Timing Issues:	Asynchronous; Post status on regular basis (i.e. each second)

No: 15	
Module:	Speech Recognition
Input:	User voice commands
Output:	The operations that need to be performed
Data Type:	Output: Result of action (e.g. opening application, logging out, etc.)
Accuracy:	Output: 75% accuracy on parsing commands
Receive/Send Frequency:	Depends on how often the user speaks
Timing Issues:	Asynchronous; Post status on regular basis (i.e. each second)

No: 16	
Module:	Web Access
Input:	The URL of the website RSS feed
Output:	Formatted data from websites (RSS feeds) that the speech construction module can process
Data Type:	Input: URL (string) Output: XML document
Accuracy:	Output: Will depend on the accuracy of the RSS feed
Receive/Send Frequency:	One web access request and one response each time the user requests an update from a website
Timing Issues:	Asynchronous; Post status on regular basis (i.e. each second)

No: 17	
Module:	Facial Recognition
Input:	Video feed from webcam
Output:	A list of recognized users and the positions of their faces in the frame
Data Type:	Output: List of strings and corresponding Vec2 (minimum <0,0>, maximum is frame resolution) positions in the frame
Accuracy:	Output: More accurate than the sample program. Should perform correct identification at least 70% of the time
Receive/Send Frequency:	The output containing information of recognized faces will be transmitted at 30 Hz
Timing Issues:	Asynchronous; The application will need to use the facial recognition module to determine who is talking at any given time so that the speech recognition/construction module responds correctly.

No: 18	
Module:	User Interface
Input:	Text and button selections
Output:	Config file
Data Type:	Output: Organized, parsable configuration data that specify how the user's profile should be customized
Accuracy:	The configuration should 100% reflect the options and settings desired by the user.
Receive/Send Frequency:	The data will be used each time the user interacts with the application
Timing Issues:	During normal usage of the application, configuration data should not change. Therefore, there should not be a problem with timing the access of data by other modules.

3.3 Physical Environment Requirements

No: 19	
Statement:	Jarvis will work on any Desktop/Laptop as long as it is equipped with camera and microphone. Good lighting is necessary.
Source:	Developer
Dependency:	None
Conflicts:	Low lighting will decrease the effective range/ accuracy of the system
Supporting Materials:	None
Evaluation Method:	None
Revision History:	Julian Rojas | 10/5/15 | Created the requirement

3.4 User and Human Factor Requirements

No: 20	
Statement:	Any person who can speak should be able to use the system. Knowledge of supported commands is recommended in order to maximize the efficacy of program.
Source:	Developer
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	None
Revision History:	Julian Rojas | 10/5/15 | Created the requirement

3.5 Documentation Requirements

No: 21	
Statement:	A set of instructions will be included as a PDF. It will also have a list of commands. It will be available as a hard copy or on-line.
Source:	Developer
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	None
Revision History:	Julian Rojas | 10/5/15 | Created the requirement

3.6 Data Requirements

No: 22	
Statement:	Eigenanalysis algorithm is required for face recognition. Recognition requires 70% accuracy.
Source:	Developer
Dependency:	None
Conflicts:	Accuracy depends a lot on cam quality and lighting
Supporting Materials:	Source of algorithm
Evaluation Method:	None
Revision History:	Julian Rojas | 10/5/15 | Created the requirement

No: 23	
Statement:	Jarvis is required to keep profiles of saved users. These include application/website preferences.
Source:	Developer
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 5
Revision History:	Julian Rojas | 10/5/15 | Created the requirement

3.7 Resource Requirements

No: 24	
Statement:	The application needs a camera and microphone to function.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	None
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

3.8 Security Requirements

No: 25	
Statement:	The application should be able to correctly identify users with a 70% accuracy and separate user specific information.
Source:	Developers
Dependency:	No. 1
Conflicts:	In case is not possible to identify users with 70% accuracy, the user will be prompted to provide his/her name
Supporting Materials:	None
Evaluation Method:	None
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

No: 26	
Statement:	All user information will be backed up in the Windows OS's Appdata to prevent loss of information when updating or reinstalling the application
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	http://windows.microsoft.com/en-us/windows-8/what-appdata-folder
Evaluation Method:	None
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

3.9 Quality Assurance Requirements

No: 27	
Statement:	In case of a system fault, the system should notify the user and attempt to reset after 1 min of non-response
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 10
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

No: 28	
Statement:	The system should have a mean time between faults of 7 days, roughly failing once every week of continuous use.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Run the application over the course of several weeks and see how often it fails.
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

No: 29	
Statement:	The system shouldn't allocate more than 1 GB of memory usage
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 13
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

No: 30	
Statement:	The system should have an average response time no longer than 5 seconds. If longer than that the user should be notified that the current activity might take longer than usual.
Source:	Developers
Dependency:	None
Conflicts:	None
Supporting Materials:	None
Evaluation Method:	Test Case 12
Revision History:	Manuel Gonzalez | 10/5/15 | Created the requirement

Jarvis Emulator
Test Plan
COP 4331, Fall 2015

Modification History

Version	Date	Who	Comment
v0.0	10/3/2015	Robin Schiro	Created document
v1.0	10/4/2015	Jimmy Lam	Added my test cases
v1.1	10/5/2015	Manuel Gonzalez	Added Speech Construction and Quality Assurance Test Cases
v1.2	10/5/2015	Julian Rojas	Added Missing Test Cases
v1.3	10/6/2015	Robin Schiro	Modified ref doc links
v1.4	10/8/2015	Manuel Gonzalez	Modified test cases 10, 11, 12
v1.5	10/8/2015	Manuel Gonzalez	Removed test case 11
V1.6	10/8/2015	Julian Rojas	Modified Test Case 14

Team Members:

- Jimmy Lam
- Julian Rojas
- Manuel Gonzalez
- Robin Schiro

1) Introduction

a) **Overall Objective for Software Test Activity**

i) *Minimize Errors:* Tests will allow us to find bugs in the code, eliminate them, and verify that problems that had been created by those bugs no longer exist.

ii) *Improve Efficiency:* Because our application is heavily reliant on advanced algorithms, tests will permit us to hone in on algorithmic weaknesses to increase the speed and accuracy of the various modules in the application. These include facial recognition, face tracking, speech recognition, speech construction, and website data requests.

iii) *Maximize Usability:* User testing will contribute to the gradual improvement of modules that require user input. For example, testing will lead us through several iterations of our user interface until it is as intuitive as can be.

b) **Reference Documents**

i) Concept of Operations

ii) Project Management Plan

iii) Software Requirements Specification

2) Description of Test Environment

a) **Environment**

i) Hardware

(1) Virtual Machine with the following specifications:

(a) CPU: Intel i5-3570K @ 3.40 GHz, 2 cores

(b) RAM: 2 GB

(c) Video Card: VMWare Virtual SVGA 3D Adapter

(d) Storage: 100 GB HDD

(2) Webcam

(a) Video Quality: 720p

(b) Snapshot Quality: 3 MP

ii) Software

(1) Windows 7 Operating System

(2) .NET Framework 4.5

(3) OpenCV

iii) Room

(1) The room in which the webcam is operating must be sufficiently lit for the webcam to record frames with at least 720p quality.

iv) The test environment contains the minimum specifications required to run the Jarvis Emulator. The software will operate well with any machine and webcam that possess at least these specifications.

b) **Testers**

i) All developers will serve as testers for this project. Each developer with possess a copy of the same Virtual Machine to run tests on.

ii) Optional testers will be our professor, Dr. Damla Turget, and our assigned TA, Amirreza Samiei.

3) Stopping Criteria

a) **Discovery of Errors**

i) On a periodic basis, we will execute all test cases (or at least all test cases that the software supports at the time of execution). When we find bugs and areas that could use improvement in the software, we will document our discoveries in a spreadsheet. Once the tester has gone through all test cases, he will place the spreadsheet in the project repository for the remaining developers to view. If the discoveries require more than one person to resolve, a meeting will be held to determine how tasks should be assigned.

b) **No Errors Found**

i) If no problems are discovered when all test cases have been run, the project will be considered “good enough to deliver”.

c) **Definition of “Good Enough to Deliver”**

i) “Good Enough” does not require that no known errors exist. With such a limited development time frame, there will undoubtedly be bugs in the product by the time we must deliver it. However, issues that prevent any test cases from passing must be resolved when the project is completed. As such, test cases will be designed to define the application’s core features. Near the end of the development cycle, time will be spent to improve the look and feel of the application, but cosmetics will not be a priority. Workarounds will only be acceptable after at least three attempts have been made to create a proper solution for a problem.

4) Test Cases

It is assumed that the application is already running in every test case.

1. Facial Recognition	
Objective: Verify that the faces of different users are accurately recognized.	
Test Conditions: There are two or more trained users present ("trained user" is defined in SRS).	
Description:	**Expected Results:**
1. Have one user walk into view of the webcam (within five feet, facing the camera).	The application should greet that user by name.
2. Have the first user exit the room and a different user enter the room. This user should stand in a position similar to that described in Step 1.	The application should greet the new user by name.

2. Face Tracking	
Objective: Ensure that the application can track the position of a user's face.	
Test Conditions: See Test Environment	
Description:	**Expected Results:**
1. Click the 'Enable Tracking' button located in the 'Video Feed' tab of the application window.	A video feed coming from the webcam should appear inside the tab space.
2. Toggle the 'Display tracking borders' option on.	You should see a square surrounding the face of each user in view of the webcam (as long as those users are facing the camera).

3. User Interface – Training – New User	
Objective: Verify that a new user can provide training data to the application.	
Test Conditions: The user has not already provided training data to the application.	
Description:	**Expected Results:**
1. Go to the 'Configuration' tab and click the 'New User' button.	You should be redirected to the video feed tab. The feed should be visible, with 'Display tracking borders' enabled.
2. Enter the name of the new User at the top of the 'Video Feed' tab. 3. Press the 'Take Snapshot' button as many times as you'd like, preferably with your face at various angles and with various expressions.	The training pictures should be stored in the folder specified within the 'Configuration' tab (default to the folder called 'TrainingData' in the same directory as the executable). The pictures for the user should be in a folder named after the user in the 'TrainingData' folder.

4. User Interface – Training – Existing User	
Objective: Verify that an existing user can provide additional training data to the application.	
Test Conditions: The user's profile has already been created.	
Description:	**Expected Results:**
1. On the 'Configuration' tab, select your name from the 'Current User Profile' dropdown menu. 2. Click the 'Continue Training' button.	You should be redirected to the video feed tab. The feed should be visible, with 'Display tracking borders' enabled.
3. Press the 'Take Snapshot' button as many times as you'd like, preferably with your face at various angles and with various expressions.	The training pictures should be stored in the folder specified within the 'Configuration' tab (default to the folder called 'TrainingData' in the same directory as the executable). The pictures for the user should be in a folder named after the user in the 'TrainingData' folder.

5. User Interface – Configuration	
Objective: Verify that a user can customize his/her profile for the application.	
Test Conditions: Must be a trained user.	
Description:	**Expected Results:**
1. On the 'Configuration' tab, select your name from the 'Current User Profile' dropdown menu. 2. In the 'Applications' area of this tab, input a string of words in the 'Trigger Words' field. In the corresponding 'Application Path' field, browse and select an application that you will interact with using your inputted trigger words. 3. In the 'Websites' area of this tab, input a string of words in the 'Trigger Words' field. In the corresponding 'Website URL' field, input the URL of the website that you will interact with using your inputted trigger words. 4. Click the 'Save' button and close the application. 5. Reopen the application, go to the 'Configuration' tab, and select your name from the 'Current User Profile' dropdown menu.	You should see the configuration settings that you had set before you had closed the application.

6. Voice Command	
Objective: Verify that Jarvis can recognize user voice commands.	
Test Conditions: The user is logged in and said the command clearly.	
Description:	**Expected Results:**
4. A list of commands will be prepared and will be read clearly to Jarvis while using a microphone.	Jarvis should correctly output the words that the user said and respond accordingly.

7. Open other applications	
Objective: Verify that Jarvis can open other applications.	
Test Conditions: The user is logged in and clearly tells Jarvis to execute a valid command.	
Description:	**Expected Results:**
1. A list of windows applications will be made. 2. Jarvis will be instructed to open each application in a new window.	Jarvis should open up the application that the user called for in a new window.

8. Logging out	
Objective: Verify that Jarvis can log the user out of his or her computer.	
Test Conditions: The user is logged in and said the command clearly.	
Description:	**Expected Results:**
1. Jarvis will be told to log out of the computer.	Jarvis should be able to log the user out of his or her computer.

9. Taking pictures of the user	
Objective: Verify that Jarvis can take a picture of the user for the user.	
Test Conditions: The user is logged in, positioned in front of the camera, and said the command clearly.	
Description:	**Expected Results:**
1. A keyword for taking the picture will be said to Jarvis, and the user will pose for Jarvis. 2. The user will then check for the picture in the file that Jarvis saves it in.	Jarvis will take the picture and store it in a folder set by the user.

10. Notifying the user through speech	
Objective: Verify that Jarvis can correctly notify the user of system specific events.	
Test Conditions: The user is logged in.	
Description:	**Expected Results:**
1. A list of fake events will be created (such as "System failure", "System busy", "Answer found", etc.) 2. Each event will be triggered during runtime and with a user present.	Jarvis will verbally inform the user of the respective event once the event gets posted internally in the system.

11. Verbal response from user questions	
Objective: Verify that Jarvis can respond accordingly to preset user questions.	
Test Conditions: The user is logged in and said the question clearly	
Description:	**Expected Results:**
1. The user will ask "How are you, Jarvis?"	1. Jarvis should verbally respond with a general status of the system.
2. The user will ask "Who am I, Jarvis?"	2. Jarvis should state the name of the user, or say that it doesn't know the user.
3. The user will ask "How's the day today, Jarvis?"	3. Jarvis should verbally respond with the weather report from that day, or notify the user that he doesn't have that information if the data is not available within 1 minute of the request.
4. The user will ask "What model is my car?"	4. Jarvis should respond that it doesn't know this answer.

12. System response	
Objective: Verify that Jarvis' average response time is below 5 seconds.	
Test Conditions: The user is logged in.	
Description:	**Expected Results:**
1. Try a minimum of 10 randomly chosen events, commands or questions consecutively.	Jarvis should respond or notify the user accordingly within the 5 second threshold.

13. Memory usage	
Objective: Verify that Jarvis uses no more than 1GB of memory.	
Test Conditions: The user is logged in.	
Description:	**Expected Results:**
1. Have the system working for several days without shutting it down.	When monitoring memory usage and storage they shouldn't go over the threshold.

14. Jarvis acquires information from RSS Feed	
Objective: Verify that Jarvis can utilize weather forecast/news headlines data from a website.	
Test Conditions: Jarvis has already recognized the user. User says command clearly.	
Description:	**Expected Results:**
1. The user will speak to Jarvis through the microphone. 2. Jarvis will then recognize the keyword and match it with the command associated with it.	Jarvis will then pull information from RSS Feed or website's API, and either display it on the screen or speak it back to the user.

15. Detection of Active User – Multiple Users in View	
Objective: Verify that Jarvis can detect a user's mouth movement.	
Test Conditions: All involved users are trained users. There are at least two faces in view of the camera.	
Description:	**Expected Results:**
1. The user will speak to Jarvis. 2. Jarvis will detect multiple faces. 3. Jarvis will select an active user once it has noticed mouth movement.	If two or more users are present, Jarvis will try to recognize the active user depending on whether or not he/she is talking, hence mouth movement.