SWEN6837 - Capstone – NASA S.U.I.T.S.

Professor: Dr. Findler
findler@uhcl.edu

Group 1 – TASK UI
Srikanth Reddy Kalvakuntla
Netaji Yadla
Nagaleela Veerapalli

Group 2 – HEALTH UI
Nikunj Patel
Akansha Kadam

Group 3 – DATABASE
Saideep Katta
Bhaavan Bruhant Koney
Yash Chauhan

Group 4 - BUISNESS LOGIC
Jack Tran
Abeje Temesgen
Clement George

SWEN6837 - Capstone – NASA S.U.I.T.S.

Professor: Dr. Findler
findler@uhcl.edu

Group 1 – TASK UI
Srikanth Reddy Kalvakuntla
Netaji Yadla
Nagaleela Veerapalli

Group 2 – HEALTH UI
Nikunj Patel
Akansha Kadam

Group 3 – DATABASE
Saideep Katta
Bhaavan Bruhant Koney
Yash Chauhan

Group 4 - BUISNESS LOGIC
Jack Tran
Abeje Temesgen
Clement George

1 OVERVIEW
2 COMPONENT DIAGRAM
3 USE CASES
4 CLASS DIAGRAMS
5 ACTIVITY DIAGRAMS
6 SEQUENCE DIAGRAMS
7 KAN BAN
8 COMMUNICATION DATA MAPS
9 SOFTWARE REQUIREMENTS
10 QUIS
11 CODE
12 APPENDIX A: NASA S.U.I.T.S. CHALLENGE DOCUMENT

Contents

OVERVIEW 7
Database Functional Requirements 9
Business Logic Functional Requirements 11
Health UI Functional Requirements 12
Task UI Functional Requirements 13
COMPONENT DIAGRAM 15
NASA Suit Overview Component Diagram 15
USE CASES 17
DATBASE USE CASES 17
BUSINESS LOGIC USE CASES 23
Business Logic Overview 23
Database Use Case 25
Task UI Use Case 25
Health UI Use Case 25
HEALTH UI USE CASES 26
Actors of the Health UI System 26
Use Case Diagram – Astronaut View 27
Ground Control View 29
Timer View 30
TASK UI USE CASES 32
Task UI Overview 32
Astronaut (View and Perform tasks) 33
Ground Controller (Detects and Monitor tasks) 35
CLASS DIAGRAMS 37
BUSINESS LOGIC CLASS DIAGRAM 37
TASK UI CLASS DIAGRAM 38
ACTIVITY DIAGRAMS 39
BUSINESS LOGIC ACTIVITY DIAGRAMS 39
SEQUENCE DIAGRAM 43
BUSINESS LOGIC SEQUENCE DIAGRAMS 43
Health UI Sequence Diagram 45
KAN BAN 47

Business Logic KAN BAN 47
Task UI KAN BAN 48
COMMUNICATION DATA MAP 49
SOFTWARE REQUIREMENTS 51
DATABASE 51
DBA Requirements 51
Designer Requirements 51
End User Requirements for the Business layer 52
Task Analysis 53
BUSINESS LOGIC 54
Introduction 54
Task Execution 54
Analog Scaling and Limit Checking 65
Alarming 65
Analog Alarm Detection 66
Alarm Acknowledgement and Resetting 70
Alarm List 72
UML Diagram for Business Logic 72
Health UI Reqiuirements 73
QUIS 85
Questionnaire 85
Responses 93
Response1 93
Response2 102
Response3 111
Response4 120
Response5 129
Response6 138
Response7 147
Response8 156
Response9 165
Response10 174
Response11 183

Response12 192
Response13 201
Response14 210
Data Analysis 219
Metrics Analysis 255
PROJECT ANALYSIS 256
BUSINESS LOGIC 256
TASK UI 257
CODE 259
AlternativeHealthUIUpdates.cs 259
Amazon.cs 267
BL_Alarming.cs 269
BL_Main.cs 282
BL_ResetingAcknow.cs 283
BL_Scalling.cs 285
BL_Tasks.cs 292
BTemperature.cs 311
ColorCode.cs 313
CommonData.cs 314
DB_CSVFileParser.cs 318
DB_DataGenerator.cs 320
DB_GenerateData.cs 325
DB_TaskTesting.cs 328
DisplayDetails.cs 334
EditableObject.cs 337
EmergencyWindow.cs 347
EnlargePieText.cs 348
HUI_WarningWindow.cs 349
HealthUIUpdates.cs 351
HeartRate.cs 355
Keywords.cs 357
Scripts/Keywords.cs 367
Pressure.cs 377

PrimaryO2.cs 379
SecondaryO2.cs 381
SuitBattery.cs 383
TUI_TaskUIUpdate.cs 385
Water.cs 389
REFERENCES 391
Appendix A: NASA S.U.I.T.S. Challenge Document 392

OVERVIEW

The 2018 Spring Capstone project for the Master of Software Engineering at University of Houston Clearlake performed the design and development of the NASA S.U.I.T.S. challenge as defined per Appendix A: NASA S.U.I.T.S. of this document. This document presents and describes the work that the Capstone class executed for this challenge. The structure of the NASA S.U.I.T.S. as designed by our team is divided into 3 layers, namely, User Interface Layer (Health UI, Task UI), Business Logic Layer and the Database Layer. The Health UI layers deals with the display of the Suit statistics information on the HoloLens which is requested to Database Layer through Business Logic Layer. In the same way, the Task UI displays the task information on the HoloLens with which the Astronaut is expected to perform task assigned to him. The Business Logic layer sets alarms for various scenarios and alerts the astronaut, these alarms are displayed on HoloLens by the Task UI layer and Health UI layer. The Business Logic Layer is also responsible to request the data from the Database Layer and send it to Health UI layer and Task UI Layer. The Database Layer stores the information required by the Health UI and Task UI layers and sends it to those layers through requests from the Business Logic Layer as in Figure 1.

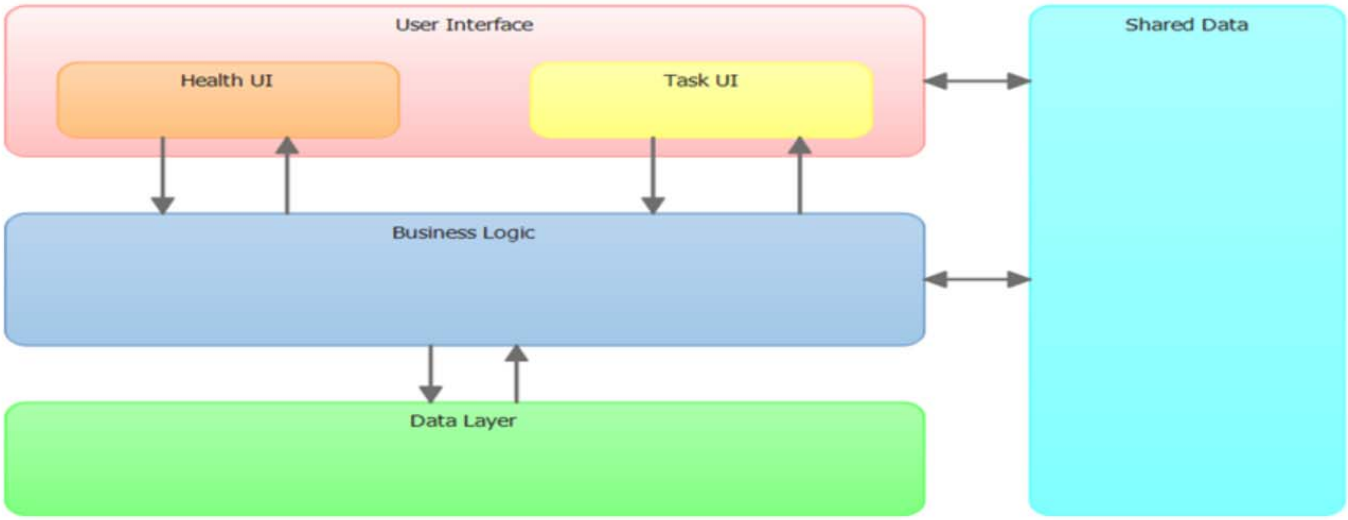


Figure 1: Overview of the Project Group Organizational Layers

The system consists of 3 layers as follows:

- The User Interface Layer consisting of two groups, Task and Health
- The Business Layer
- The Database Layer

The Database Layer is responsible to store and provide data required by the other layers. We have certain values that are to be stored into our database so that the other layers can access those for the user Interface layer we have health and task teams that require data that must be displayed on to the HoloLens. So, we used some specific technique to store that the data so that it can be accessed easily.

Database, Business Logic, Task UI, Health UI

We used cloud technology to access the data. We used Amazon web services (AWS) to store and access the data from and we used a component called as the AWS S3 buckets. Here we create some buckets that are used to store the data, the data is been put into the bucket in the form as JSON files for the task UI team and for the health UI we have created a Data generator that will create the data according the values required, and we are given some ranges that must be varied. So, we have the Business logic layer as the end user. We have a shared region where we put all the data and it is been accessed by the business logic layer and sent to the user interface layer. The business logic layer has created some tables to which we have to map our database tables for easier and faster data retrieval.

The Business Logic for this project provides the middleware for the HoloLens unit. It is an interface between the Database and the User Interface sections of the project. The two User Interface sections of the project are UI Health and UI Tasks. As this is a work in progress, some of the major functions that are contained in the Business Logic are as follows:

- Task Execution
- Analog Scaling and Limit Checking
- Alarming
- Alarm Acknowledgement and Resetting

The Health UI of the UI layer provides some visual information and some of the interaction with the Astronaut. Major functions are as follows:

- Analog Data in numeric format and chart form
- Alarming and Acknowledgement

The Task UI of the UI layer provides some visual information and some of the interaction with the Astronaut. Major functions are as follows:

- Task steps to instruct the astronaut
- Visual guides to aid the astronaut through the tasks

Database Functional Requirements

The Actors of Database

Database Administrator

- Designing the database
- Creating the database
- Maintaining the database
- Troubleshooting for errors
- Database security
- Backup and recovery of data

Database Designer

- Designing database schema
- Designing the classes
- Updating the database design which includes relating the classes
- Handle Data Modelling
- Design dataflow

End User

- Receive UI Data request
- Send request to Database
- Receive response from the Database
- Limit Data
- Send data to the UI

Assumptions

The assumption that we made are quite difficult one, but we came up with the right choice. Firstly, the creation of database in the form of .CSV (coma separated values) format. Secondly, creation of data generator that creates data for the health UI according to the high and low values that are been specified by the NASA.

Constraints

That database design has some constrains that made us choose the way of putting up the data on to the cloud. As we use the HoloLens which cannot access the data directly from the database. To access the data by the HoloLens we have to create a separate database and install it on to the HoloLens which will consume lot of space on the HoloLens. This made us use the Amazon web services by which we can create link that can be accessed by an API that is been installed on to the HoloLens. The one other constrain that we got is like the NASA asked us to develop everything in the JSON format.

Database

Risks

There were few cases reported about data leakage from AWS S3 buckets. The data is prone to leakage when the bucket is given public access and unencrypted. But the AWS S3 provides enough privacy services to keep the data safe.

Key Factors Influencing Design

Initially we had an idea to store the required information for the project in .csv files and access through it. But the inability of HoloLens to use its internal memory for the database of the project made us to create the database in cloud and access it with the web connectivity of HoloLens.

The Amazon Simple Storage Service (S3) provides the access of buckets where the data files can be stored and accessed. AWS S3 provides privacy for the data files and the files can be accessed with the help of Access keys and Secret keys.

Functional Design Decisions

Data of any valid format can be stored in S3 buckets. It can be accessed by providing Access keys and Secret keys and can also be accessed by creating a pre-signed URL for each file in the bucket. The information required by the Task UI layer is saved in .json format with fields of task number, step, caution, warning, holograms and step images. The AWS S3 buckets are accessible in Microsoft visual studio by installing AWS SDK. User can be able to list the objects in the bucket, read the files and can also write or upload new files into the S3 bucket.

Database Management System Decisions

The Amazon AWS provides management capabilities which can be administered using AWS Management Console. The various data management features provided by AWS S3 are:

- S3 Object tagging
- S3 inventory
- Storage Class Analysis
- Amazon CloudWatch metrics for amazon S3
- AWS CloudTrail Management
- Amazon S3 Data Lifecycle Management
- Cross-Region replication
- Cost monitoring and controls and event notifications.

Security and Privacy Design Decisions

We can upload or download data from Amazon S3 via SSL-encrypted endpoints using the HTTPS protocol. Amazon S3 can automatically encrypt the data and gives several choices for the key management. It uses Advanced Encryption Standard (AES) 256-bit symmetric keys. In S3, if we choose server-side encryption, there are three ways to manage encryption:

- SSE with Amazon S3 key management
- SSE with Customer-provided keys
- SSE with AWS KM

Database

Business Logic Functional Requirements

The Business Logic for this project provides the middleware for the HoloLens unit. It is an interface between the Database and the User Interface sections of the project. The two User Interface sections of the project are UI Health and UI Tasks. Some of the major functions that are contained in the Business Logic are as follows:

- Task Execution
- Analog Scaling and Limit Checking
- Alarming
- Alarm Acknowledge and Resetting

TASK EXECUTION

The Task Execution software is a used to display instructions for the astronaut. A sequence of instructions is displayed on the HoloLens that guides the Astronaut in performing a procedure or task. The steps for the tasks are stored in the Database provided to the system by NASA. The Business Logic reads the steps of the task and place task data into a buffer for the CURRENT TASK. The tasks are stepped forward and backward based on a NEXT and PRREVIOUS command that is passed from the User Interface Task software.

ANALOG SCALING and LIMIT CHECKING

This section of software for Analog Scaling and Limit Checking will reside in the Business Logic. The Business Logic will read the analog value from the Database software. There is a maximum and minimum value for every analog value. The maximum and minimum value for each analog value will be part of the Database. The business Logic will use the Maximum and Minimum values will place limits on the analog value that is read from the Database.

ALARMING

The Business Logic accesses Analog values as read from the database. Each analog value is tested for going above at two levels Hi and HiHi and below two levels of detection Lo, LoLo. There is a deadband for all four levels of alarm detection. Any of the four analog alarm levels may be enabled or disabled. The Alarm Acknowledge and Resetting section of software is the area that the active alarms are stored are read from for display on the UI, User Interface.

ANALOG ALARM DETECTION

The Alarm software evaluates an analog value for limits. There is a check for at four levels, HiHi, Hi, Lo, LoLo. The alarms are activated as the analog value increase above the Hi and HiHi limits and activates when the analog value decreases below the Lo and LoLo limits. There is a deadband for all four levels of alarm detection. The purpose of the deadband is to minimize oscillation around an alarm setpoint thus creating nuisance alarms.

Health UI Functional Requirements

Astronaut View:

- The Astronaut shall give commands by making hand gestures.
- The Astronaut shall perform tasks by giving voice commands.
- The Astronaut shall view the sprocket details.
- The Astronaut shall check for warnings.
- The Astronaut shall acknowledge the warnings.
- The Astronaut shall check for suit damage.
- The Astronaut shall report the details to Ground Control.
- The Astronaut shall operate the display window which includes drag display window, tag display window and close display window.

Ground Control View:

- The Ground Control shall check for warnings.
- The Ground Control shall import the sprocket details and the suit parameter details.
- The Ground Control shall provide guidance to the Astronaut if an error occurs.
- The Ground Control shall alert the Astronaut if a terminal alert occurs.

Timer View:

- The Timer shall display low-level warnings and mid-level warnings on the HoloLens screen.
- The Timer shall display caution notice on the HoloLens screen.
- The Timer shall display alert which includes the terminal alerts to notify the Astronaut.
- The Timer shall display error messages if an error occurs in a suit parameter.

Project Requirements

- Windows 10
- HoloLens
- Unity 3D 2.1
- Visual Studio Enterprise Edition 2015/2017
- C# programming Language

Task UI Functional Requirements

Functional Requirements for Astronaut:

- Astronaut should have access to various tasks available.
- Astronaut should be able to check the warnings/cautions.
- Astronaut should be able to locate the box
- Astronaut should be able to view the tasks on task screen.
- Astronaut should be able to view the picture.
- Astronaut should be able to request for previous steps of the task.
- Astronaut should be able to request for next steps of the task.
- Astronaut should be able to repeat the task.
- Astronaut should be able to able to confirm the steps performed.
- Astronaut should be able to switch to a different task in case of an interruption.
- Astronaut should be able to view 3D visualization to task steps that he is performing.
- Astronaut should be able to use gestures as the secondary guidance.
- Astronaut should be able to seek directional guidance.

Functional Requirements for Ground Controller

- The Ground Controller should be able to monitor the task performed by the Astronaut.
- Astronaut should be able to detect the errors performed by the Astronaut.
- The Ground Controller should be able to interrupt the tasks performed by Astronaut if there are any errors.
- The Ground Controller should be able to direct the Astronaut in step corrections if he detects any errors during task.

Functional Requirements for Timer

- The Timer should display the tasks and steps to be performed.
- The Timer should prompt for step confirmations.
- Timer should be able to show the warnings
- Timer should guide the astronaut to detect the box
- Timer should provide directional guidance for the astronaut.
- Timer should be able to display warnings.
- Timer should be able to display picture.
- Repeat Task steps when requested from the Astronaut.
- Listen to Astronaut’s voice commands in case of help.

COMPONENT DIAGRAM

The component diagram helps to model implementation details and of the system's required functionality. Additionally, it provides a logical unit block of the system, at a slightly higher abstraction than classes.

NASA Suit Overview Component Diagram

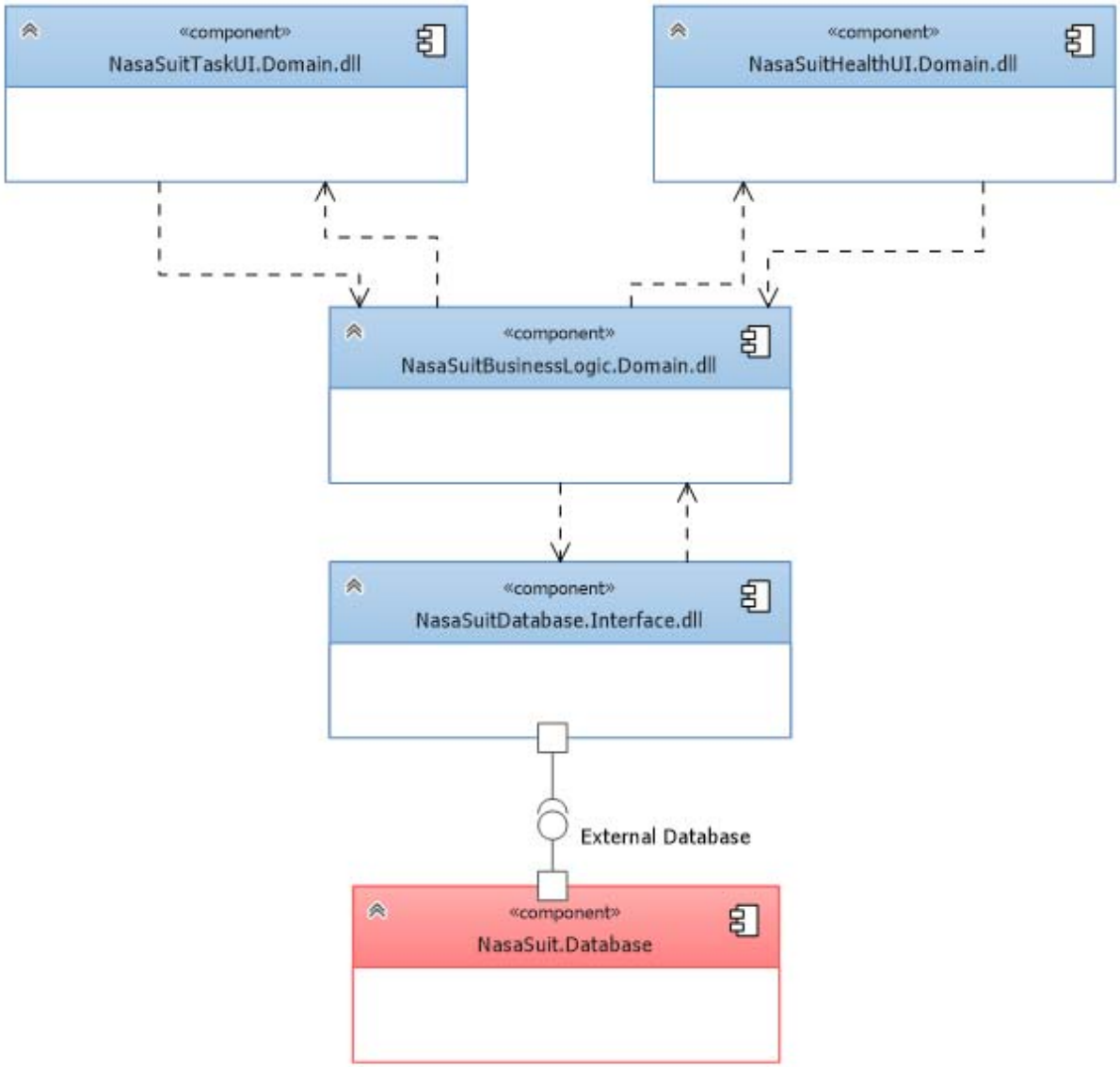


Figure 2: NASA Suit Overview Component Diagram

Database, Business Logic, Health UI, Task UI

USE CASES

The purpose of the Use Case is to capture how the actors or users of the NASA S.U.I.T.S. system communicate with its database layer and the functionalities of the different users can be found. The actor can be human or an external system. The Use Case diagrams gives sequence of actions that yield an observable result of value to a actor. Individual use case of each actor in the system describes how a actor interacts with the system to achieve a result. All use cases together describe the complete behavior of the system

DATBASE USE CASES

Database Administrator

- Designing the database
- Creating the database
- Maintaining the database
- Troubleshooting for errors
- Database security
- Backup and recovery of data

Database Designer

- Designing database schema
- Designing the classes
- Updating the database design which includes relating the classes
- Handle Data Modelling
- Design dataflow

End User

- Receive UI Data request
- Send request to Database
- Receive response from the Database
- Limit Data
- Send data to the UI

Use Case diagram of Database Administrator

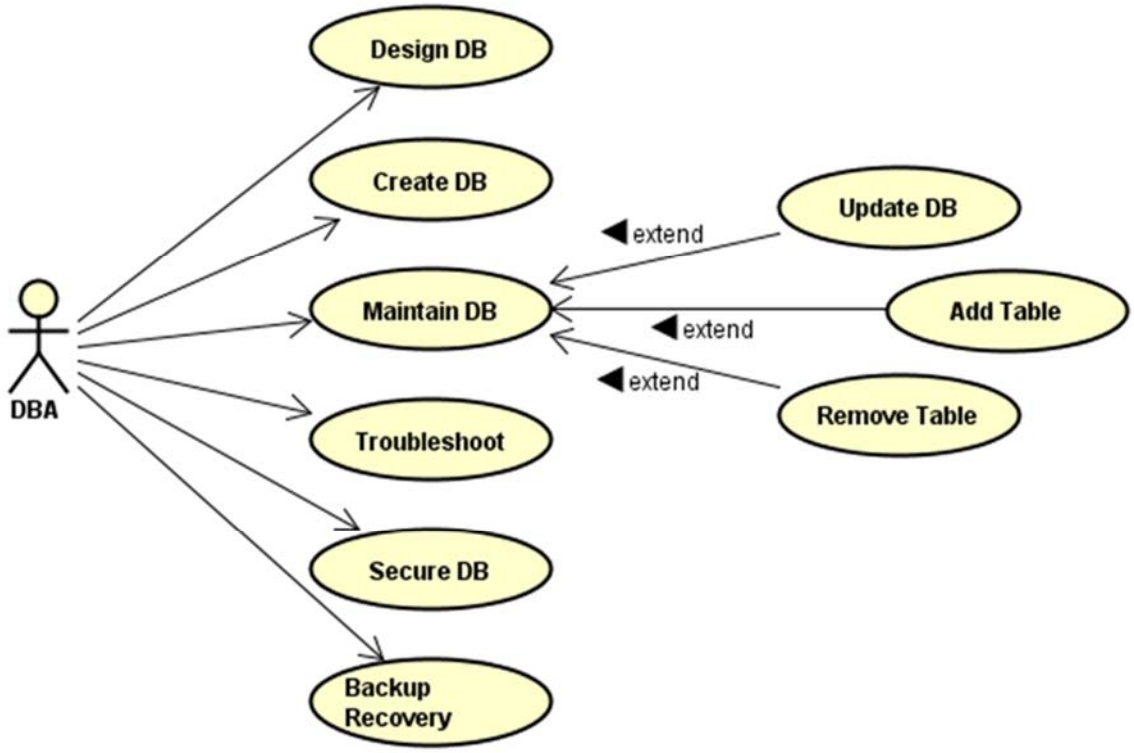


Figure 3: Use Case diagram of Database Administrator

Brief description	The above use case describes how the Database Administrator interacts with the database system.
Functionalities of the Database Administrator	The Database Administrator is responsible for various actions on the database which includes, designing the database, maintaining the database, troubleshooting for the problems, securing the database and backup and recovery of the database. For this project, the data is stored in Amazon web services' S3 storage in the form of data buckets.
Pre-Conditions	**1**. The system should be able to access data from the database.

	2. Data required by the Task UI layer should be accessible through business logic. Data required by the Health UI layer should be accessible through business logic.
Basic flow	**1**. The Database Administrator should create the database in AWS S3 storage. **2**. The Database should be accessible, provided with access credentials and permissions. **3.**The Database should be able to respond to the requests from the Business Logic Layer by providing the required information **4.** The Database is secured by login credentials and the access permissions.
Alternative flow	**1.** Keeping Data loss as a threat, the database should be backed up for recovery.
Post-conditions	**1.** The Health Layer and the Task UI Layer are able to access data from the Database Layer through the requests from Business Logic Layer. **2**.The Business Logic Layer is able to receive response to its data requests made for Health UI and Task UI Layers
Extensions	**1.**Update Database **2.**Add Tables **3.**Remove Tables

Use Case diagram of Database designer

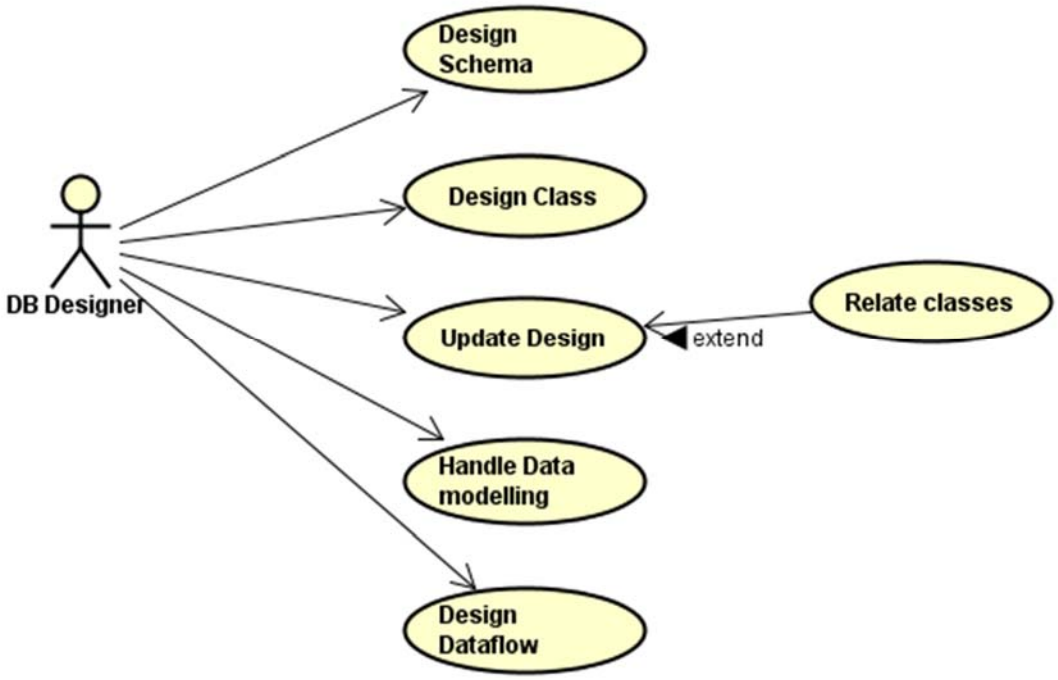


Figure 4: Use Case diagram of Database designer

Brief Description	The above Use Case diagram describes various actions of the Database Designer.
Functionalities of the Database Designer	A Database designer is responsible for designing database schema, designing system elements, establishing relations among elements, handle data modelling and design the dataflow inside the system.
Pre-Conditions	1. The Database designer should create a particular dataflow in the system. 2. Database schema and the relations between the classes must be described.
Basic flow	1. The Database designer should design a syllogistic figure for the database.

Database

	2. The Database designer should design classes for the data. 3. The Database designer should Update the design if necessary which includes relating the classes. 4. The Database designer should design a dataflow for the database system.
Post-Conditions	1. A particular schema of the database is obtained. 2. The relations between the classes of the database is obtained.

Use Case diagram of the End User

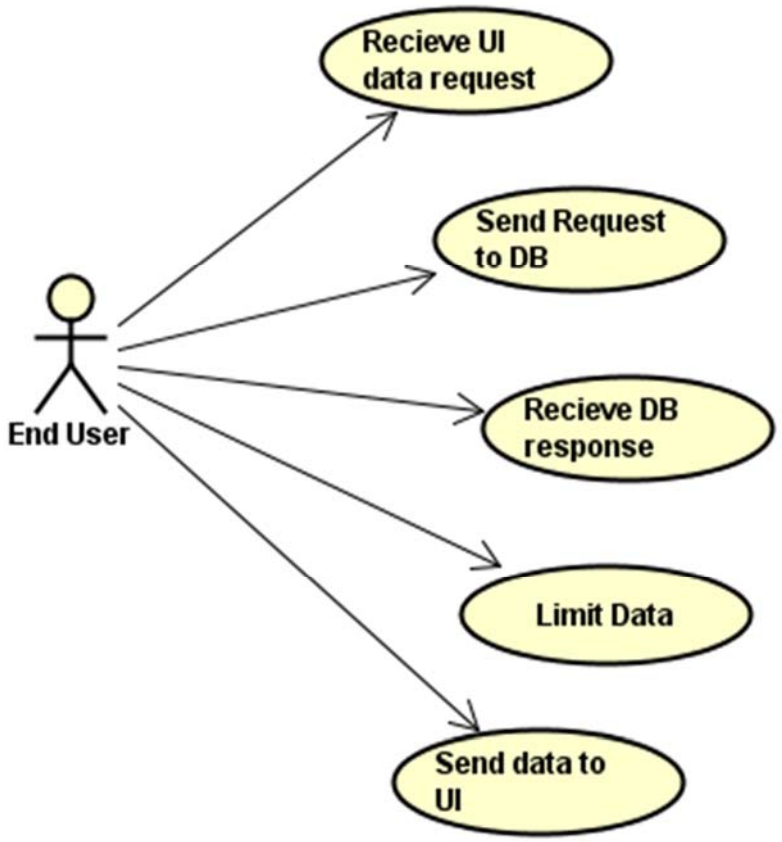


Figure 5: Use Case diagram of End User

Database

Brief Description	The above Use Case diagram describes the various actions performed by the End User on the system.
Functionalities of the Database Designer	**1.** The End User limits the data, by giving range for the suit statistics information that are displayed on the HoloLens by the Health UI. **2.** The End User receives the data from the database layer through requests from the business logic layer. **3.** The End User is able request data through voice commands and hand gestures.
Pre-Conditions	**1.** The End User should be able to access data from the Database layer. **2.**The End User should be able to request data .
Basic flow	**1 .**Receive Data request from the UI layer **2.** Send data request to the Database layer. **3.** Receive response from the database layer. **4.** Limit the data by giving ranges to the suit information statistics. **5.** Send data to the UI layers.
Post-Conditions	1. End User can receive data from the database layer. **2.** End user can request data with the help of hand gestures and voice commands.
Alternative flow	1. Abort the mission in case of emergency.

BUSINESS LOGIC USE CASES

Business Logic Overview

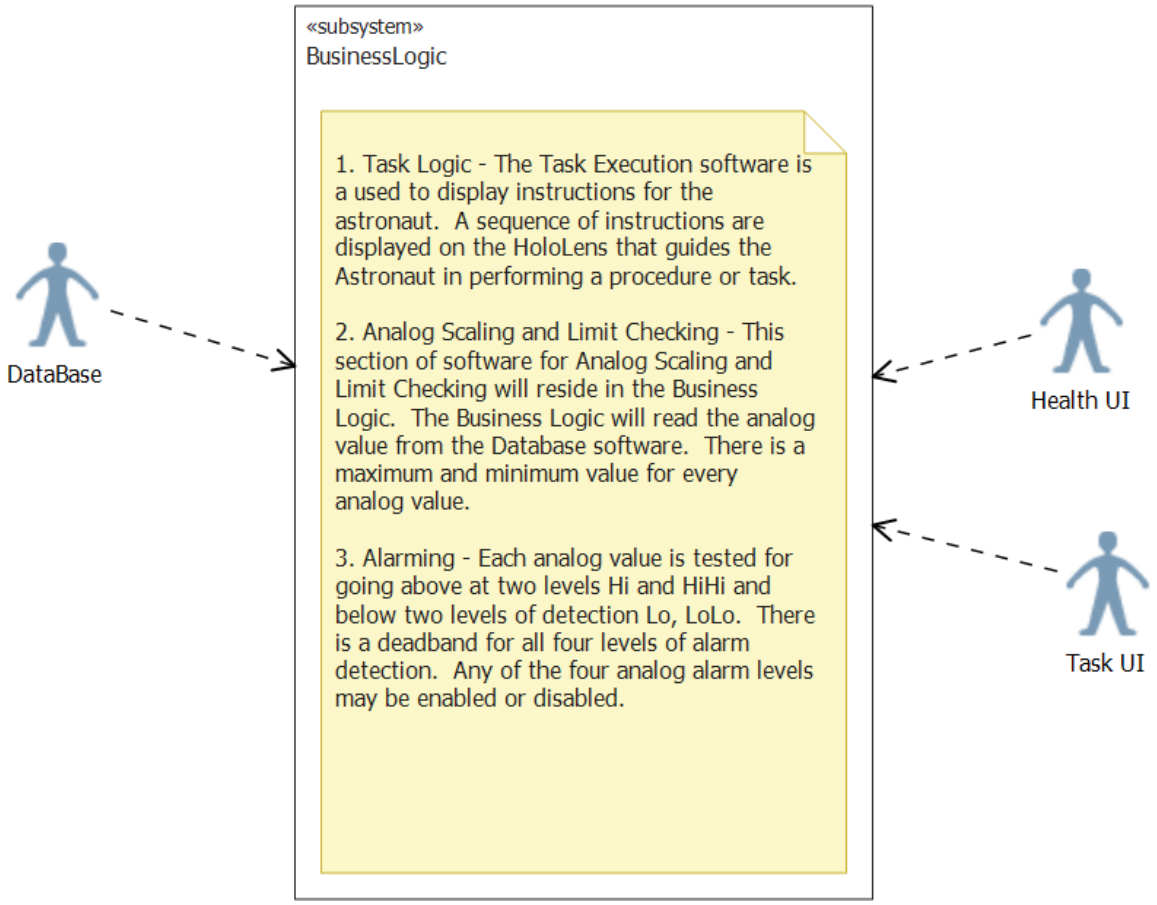


Figure 6: Business Logic Overview Use Case

Use case: Complete Task

Use case description: The astronaut should be able to complete a specific task by selecting a task from a list. The tasks can be removing and installing fuses with tools, operating keys and unlocking locked cases, installing cables into outlets with correct orientation and cable routing.

Primary actors: The astronaut and the system are the primary actors for this use case

Pre-Conditions: The pre-conditions for this use case include:

- The software system in up and running properly
- Extra vehicular activity (EVA) task board has the necessary parts

Post-Conditions: The post-conditions include:

Success condition:

- The specified task is accomplished successfully
- The astronaut has the option to select or search another task or navigate through suit status

Failure condition:

- Astronaut is not able to complete the necessary steps to complete a task
- Astronaut contacts the ground control or looks through a manual task list to complete a task

Use case: Display suit information and alarm when necessary

Use case description: It displays information about the astronaut's suit; it includes pressure, temperature, oxygen. When the levels are below or above the threshold limits, it should give an alarm.

Primary actors: The astronaut, the software system and data collection instruments are the primary actors.

Pre-Conditions: The following are the pre-conditions for display suit information

- The system in running properly
- The sensors are recording the required information correctly
- Suit information is logged to the database

Post-Conditions:

Success condition:

- The system displays suit information (Temperature in Fahrenheit, Pressure in pound-force per square inch (psi) and Oxygen in percent remaining).
- Astronaut should be able to read the information even when he is completing a task
- The system gives alarm when suit information status is above or below the threshold limits

Failure condition:

- The system not displaying suit information
- Ground control should be able to notify astronaut about suit information from the database
- Performing any task is impossible and the astronaut will go back to the spacecraft

Business Logic

Database Use Case

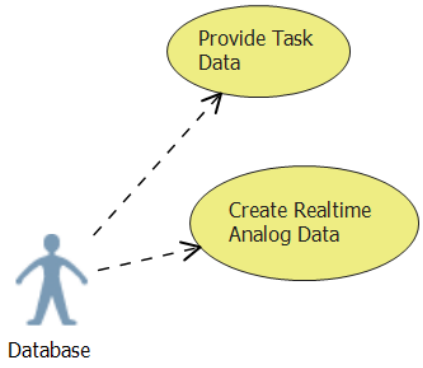


Figure 7: Business Logic Database Use Case

Task UI Use Case

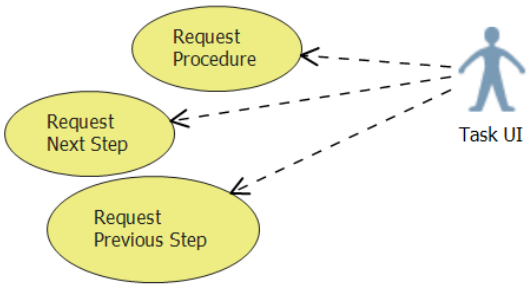


Figure 8: Business Logic Task UI Use Case

Health UI Use Case

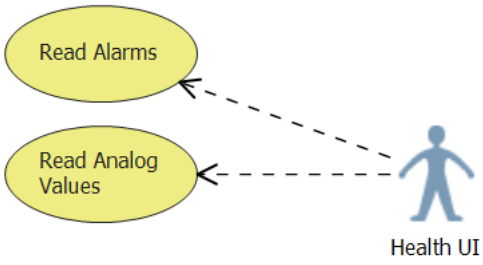


Figure 9: Business Logic Health UI Use Case

HEALTH UI USE CASES

Actors of the Health UI System

Astronaut
The Astronaut is a user who can operate the HoloLens and perform various tasks either by using hand gestures or voice commands.

Ground Control
The Ground Control is a base station that provides guidance to the Astronaut.

Timer
The Timer is responsible for providing warnings, terminal alerts and error messages to the Astronaut.

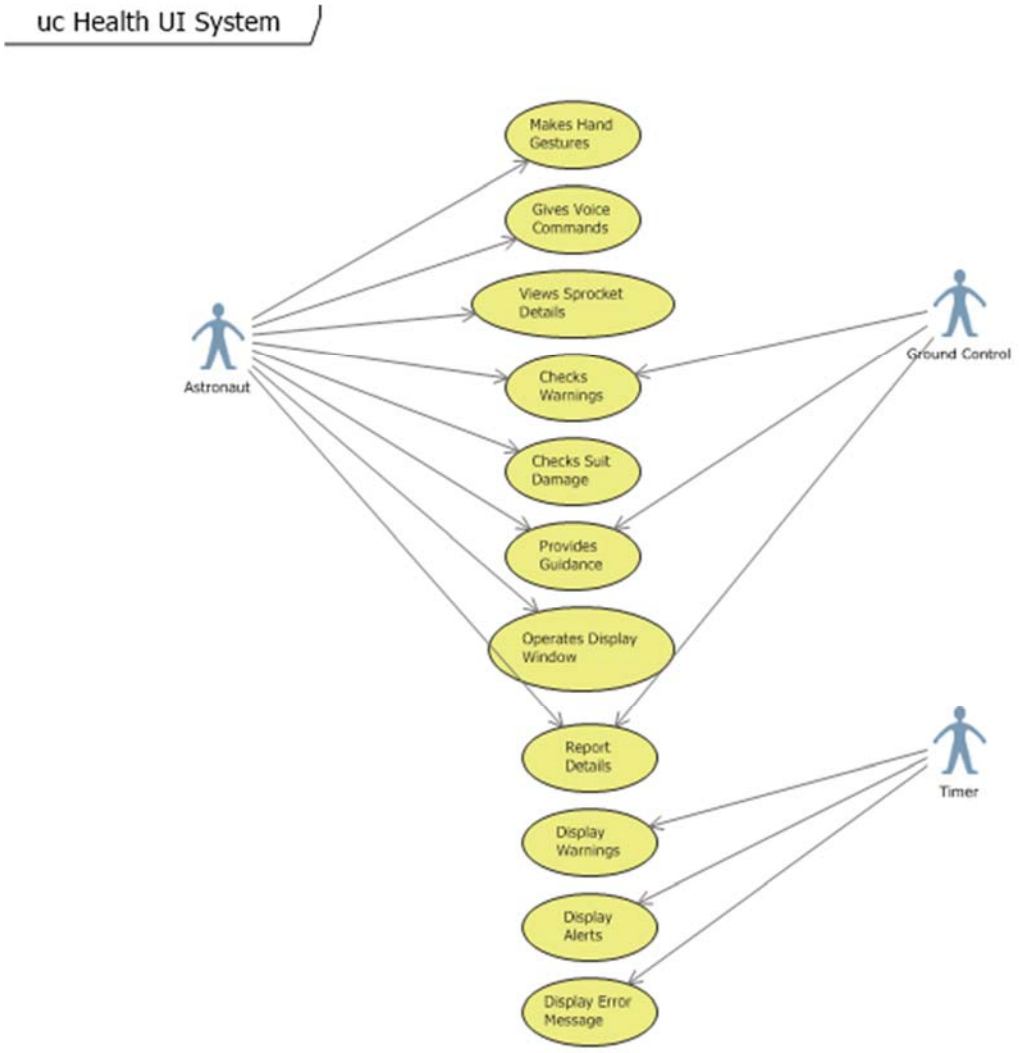


Figure 10: Cloud Level Use Case for Health UI

Brief Description:

The above use case diagram depicts the functionalities of all the actors that interact with the Health UI System. This system consists of three actors and each actor performs certain functionalities.

Use Case Diagram – Astronaut View

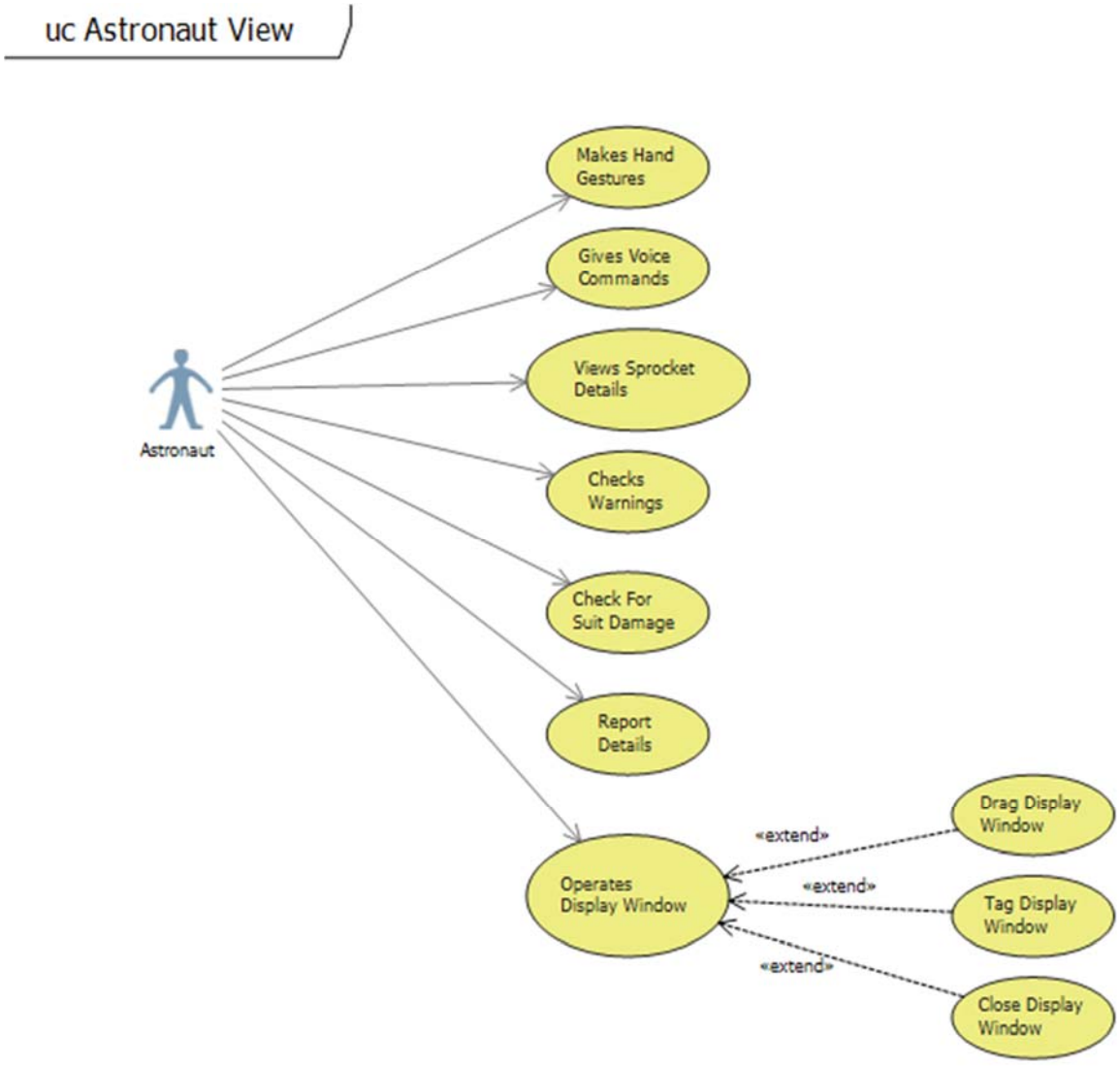


Figure 11

Brief Description:

This use case provides the functionalities of an Astronaut giving Voice Commands, Making Hand Gestures, Viewing Sprocket Details, Check Suit Damage, Report Details, Check Warnings and Operate Display Window.

Pre-conditions:

- The Astronaut must have started the HoloLens environment
- The Astronaut can use view the initial sprocket details.

Flow of Events:
The flow of events consists of two parts: Basic Flow and Alternative Flow. The steps that occur in normal case i.e. the steps with no errors are described in the basic flow. The steps that occur in exceptional cases are described in the alternative flow.

Basic Flow:

- The Astronaut uses voice commands or hand gestures to perform various tasks.
- The Astronaut can view the sprocket details like primary oxygen, secondary oxygen, water level, battery, pressure, temperature and heart rate.
- The Astronaut can check for warnings like low-level warnings and mid-level warnings.
- The Astronaut can check the suit damage and report the details to Ground Control.
- After the details are displayed on the display window, the Astronaut can operate the Display Window by using commands like Drag Display Window, Tag Display Window and Close Display Window.

Alternative Flow:

- Invalid Voice Command
- If the voice command provided by the Astronaut do not match, the Astronaut should use the appropriate voice command and must contact the Ground Control to retrieve the voice command for the specific task.
- Unable to report the details
- If the Astronaut cannot report the details, the Astronaut needs to check the connection with the Ground Control.

Post Conditions:

The Astronaut makes hand gestures or uses voice commands and can view the sprocket details, warnings and suit parameters.

Health UI

Ground Control View

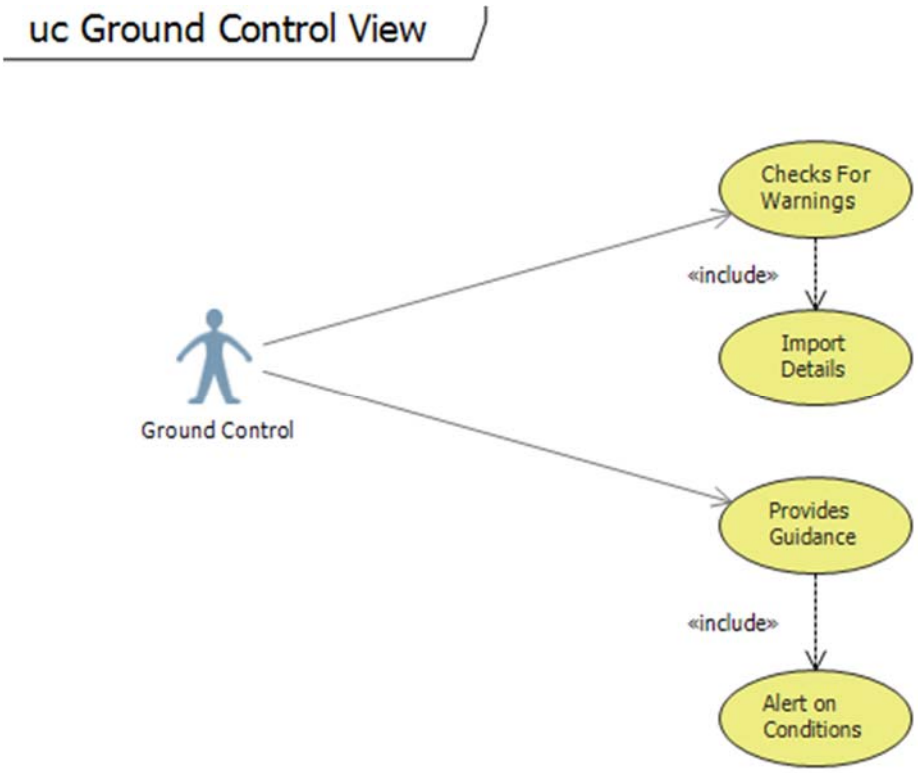


Figure 12: The Use Case for Ground Control

Brief Description:

This use case provides the functionalities of Ground Control in Checking for Warnings and Providing Guidance.

Pre-conditions:
The Ground Control must be connected to the HoloLens.

Flow of Events:
The flow of events consists of two parts: Basic Flow and Alternative Flow. The steps that occur in normal case i.e. the steps with no errors are described in the basic flow. The steps that occur in exceptional cases are described in the alternative flow.

Basic Flow:

- The Ground Control can check for warnings.
- The Ground Control can guide the Astronaut and provide alert on certain conditions.

Alternative Flow:

- Unable to import the suit details

Health UI

If the Ground Control cannot import the suit details, then the Ground Control needs to check the connection with the Astronaut.

Post Conditions:
The Ground Control provides guidance to the Astronaut.

Timer View

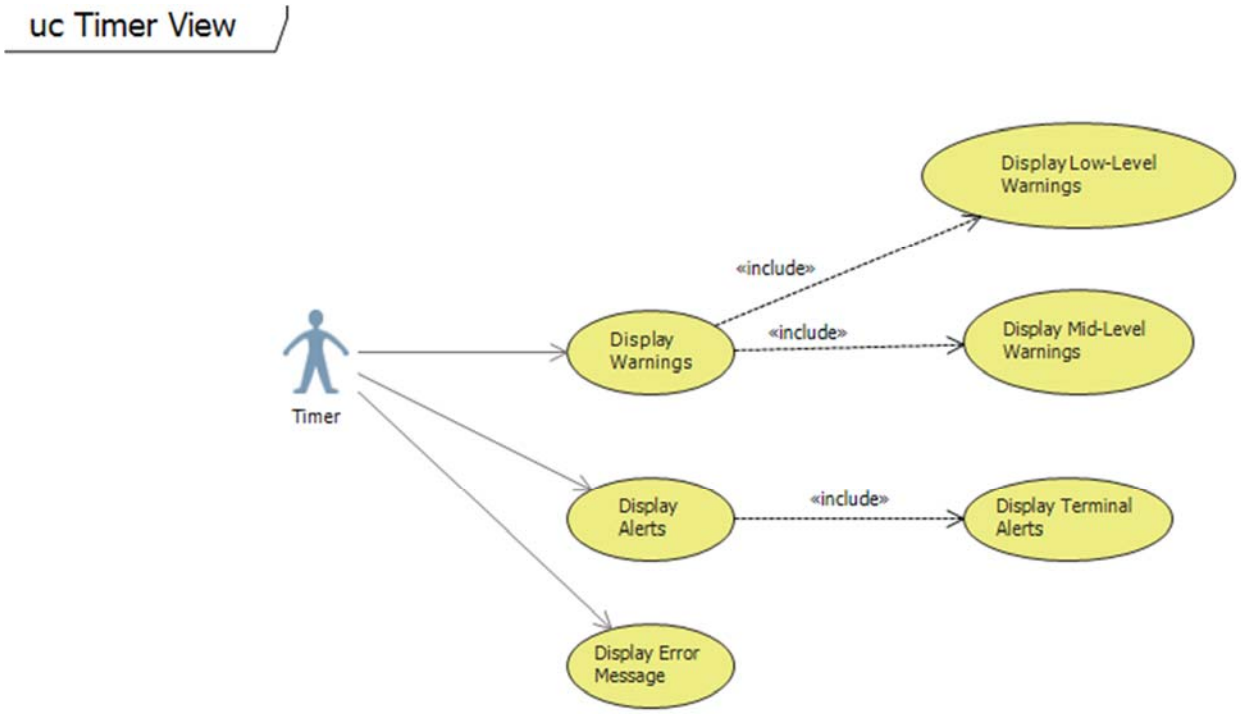


Figure13: Use Case Diagram for Timer

Brief Description:
This use case provides the functionalities of Timer displaying warnings, displaying alerts and displaying the error messages.

Pre-conditions:
The Timer displays the warnings on the HoloLens.

Health UI

Flow of Events:
The flow of events consists of two parts: Basic Flow and Alternative Flow. The steps that occur in normal case i.e. the steps with no errors are described in the basic flow. The steps that occur in exceptional cases are described in the alternative flow.

Basic Flow:

- The Timer can display warnings. The warnings can be low-level warning and mid-level warning.
- The Timer can display terminal alerts.
- When an error occurs in a suit parameter, the Timer can also display the error messages.

Alternative Flow:

- Error message does not get displayed
- If the Timer is not able to display the error message on the screen, then the Timer should retrieve the updated data about the suit parameters.

Post Conditions:
The Timer displays the warning and terminal alerts. The Timer also displays the error message about a particular suit parameter.

TASK UI USE CASES

Task UI Overview

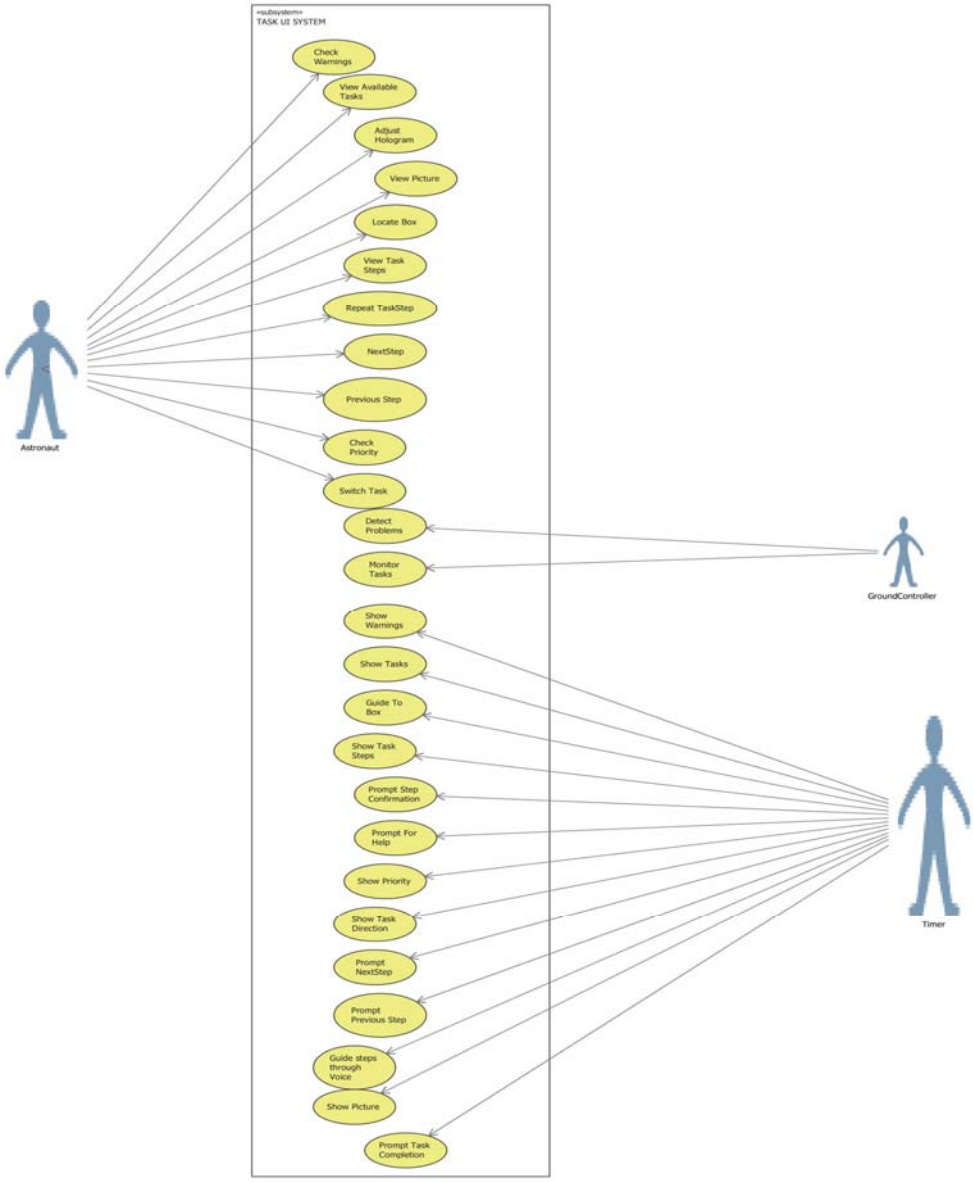


Figure 14

The above Use case describes the Cloud level functionalities of all the actors that interact with the Task UI System. There are 3 actors that interact with the system and each actor performs certain functionalities.

Functionalities of the Astronaut**:** View tasks, select tasks, Switch tasks, Checking Warning, View Available Task, Adjust Hologram, Move Hologram, View picture. This includes Performing tasks like load task, read task step, go to next step, go to previous step, repeat step.

Functionalities of the Ground Controller: Detect Problems, Monitor tasks.

Functionalities of the Timer: Project Tasks to be performed, Project Task steps, Guide to locate the box, Project Warnings, Prompt Step Confirmations, Prompt Next steps, Project Priority, Project Picture, Project Task Direction, Prompt Task Completion, Prompt for Help

Constraints

Ground Controller monitors the situation.
Task step changes based on the priority.

Astronaut (View and Perform tasks)

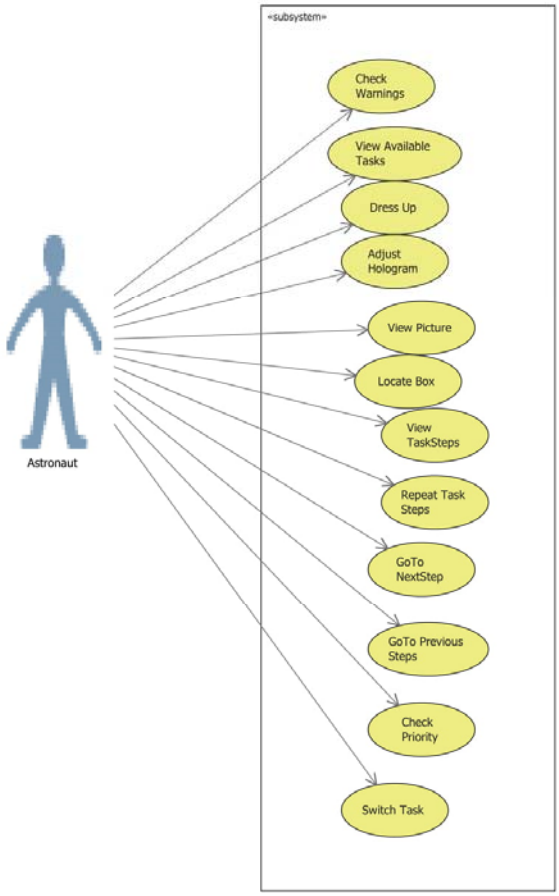


Figure 15

Task UI

Brief Description

This Use Case provides the functionalities of the Astronaut like Dress up, Locate the Box, View tasks, select tasks, Switch tasks, Checking Warning, View Available Task, Adjust Hologram, Move Hologram, view picture. This includes Performing tasks like load task, read task step, go to next step, go to previous step, repeat step

Preconditions

Astronaut:
- should check warnings.
- should be able to detect object.
- should be able to view Task UI.

Basic Flow:
- Astronaut can check cautions/ warnings.
- Astronaut should dress up
- Astronaut should be able to locate the box.
- Astronaut can adjust hologram.
- Astronaut should be able to view all the tasks.
- Astronaut should be able to view the picture.
- Astronaut should select task from the menu.
- Astronaut should be able to read task step.
- Astronaut has access to go to previous step.
- Astronaut has access to go to next step.
- Astronaut can be able to repeat the current step.
- Astronaut should be able to switch the task in case of priority

Alternate Flow:
- Request to locate task board
- Request to repeat whole task
- Seek directional guidance
- Switch to a different task in case of interruption from timer
- Check for warnings or cautions
- Use Gestures as secondary guidance.

Postconditions:
Astronaut can view tasks, read tasks, and complete tasks.

Task UI

Ground Controller (Detects and Monitor tasks)

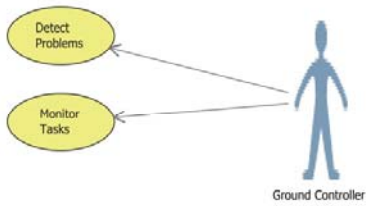


Fig16: Use Case for Ground Controller

Brief Description

This Use Case provides the functionalities of the Ground Controller like detecting the problems, communicating details about all the tasks and monitoring the tasks.

Preconditions:

Ground Controller need to monitor the situation.

Basic Flow:

- Ground Controller detects problems.
- Ground Controller send warnings.
- Ground Controller monitor tasks.

Alternate Flow:

- If the Ground Controller does not detect problems, astronaut should notify him.
- When the warning was not sent from the Ground controller, the Astronaut should notify him.

Post Conditions:

Ground Controller is responsible for Communicating details about all the tasks

Task UI

CLASS DIAGRAMS

BUSINESS LOGIC CLASS DIAGRAM

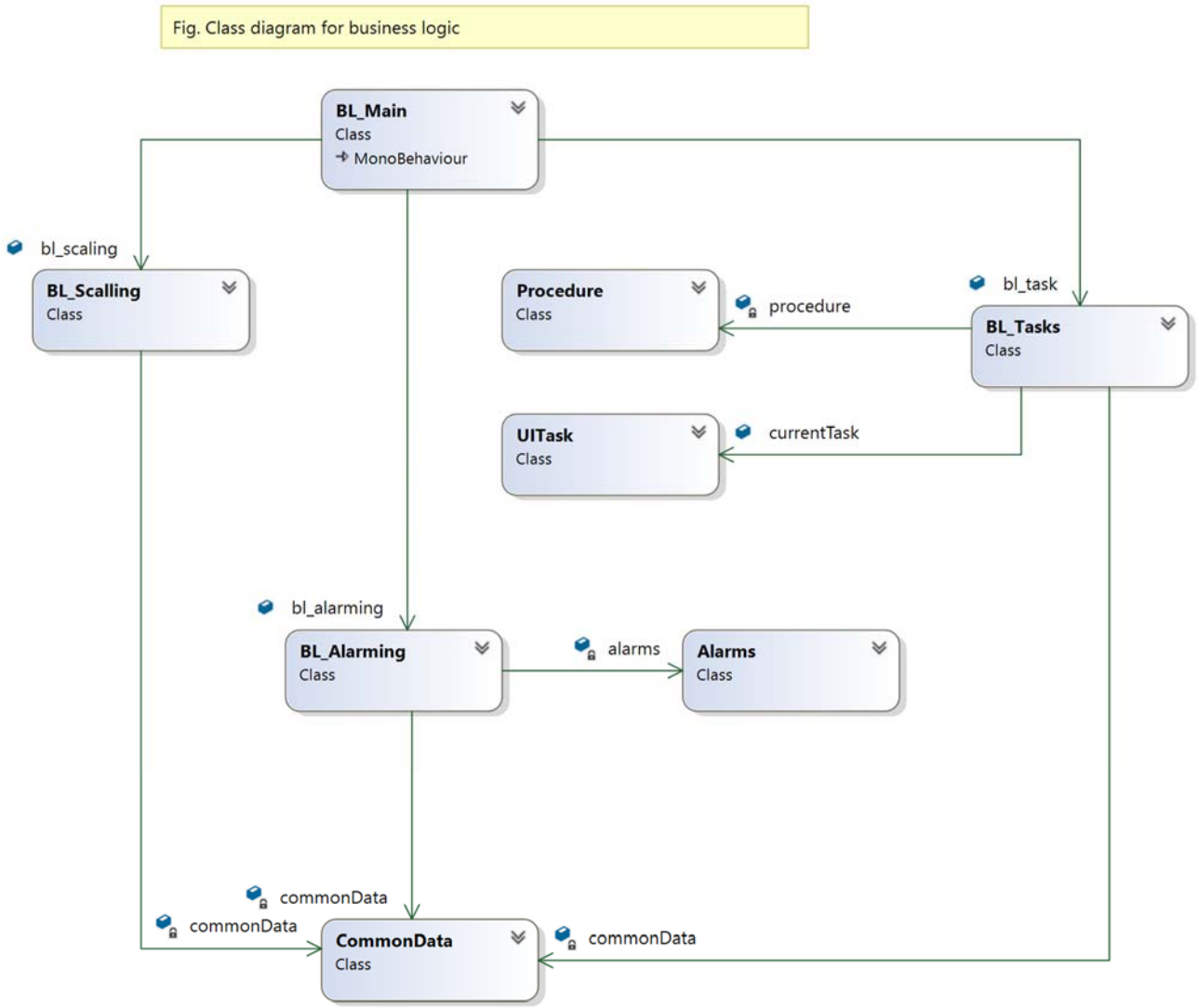


Figure 17: BUSINESS LOGIC CLASS DIAGRAM

TASK UI CLASS DIAGRAM

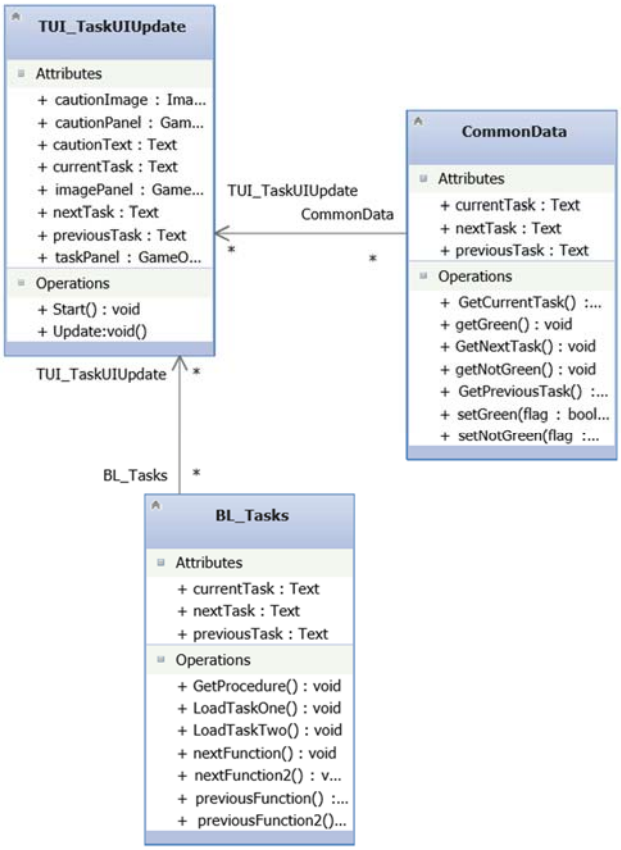


Figure 18: TASK UI CLASS DIAGRAM

ACTIVITY DIAGRAMS

BUSINESS LOGIC ACTIVITY DIAGRAMS

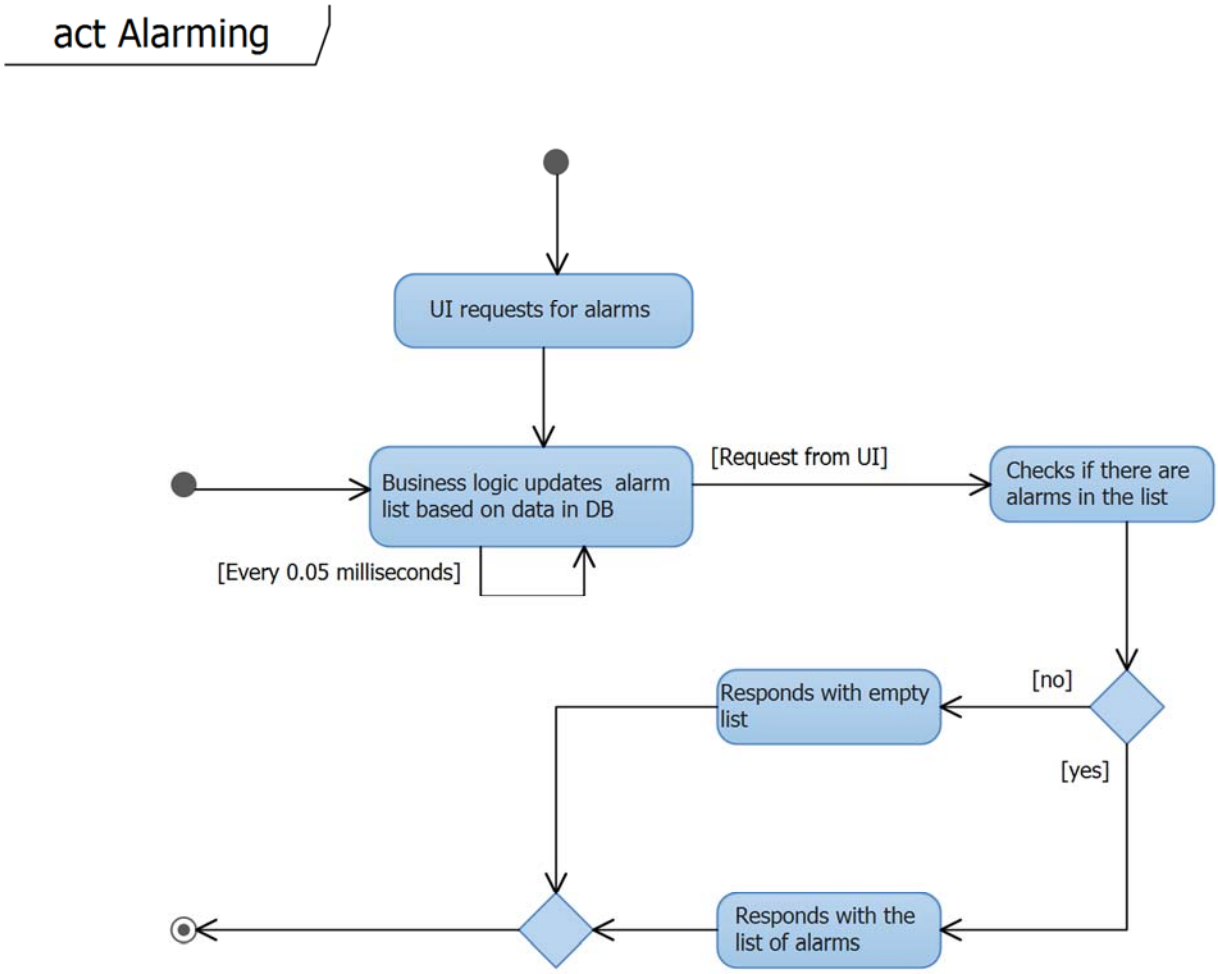


Figure 18: BUSINESS LOGIC ACTIVITY DIAGRAM for Active Alarms

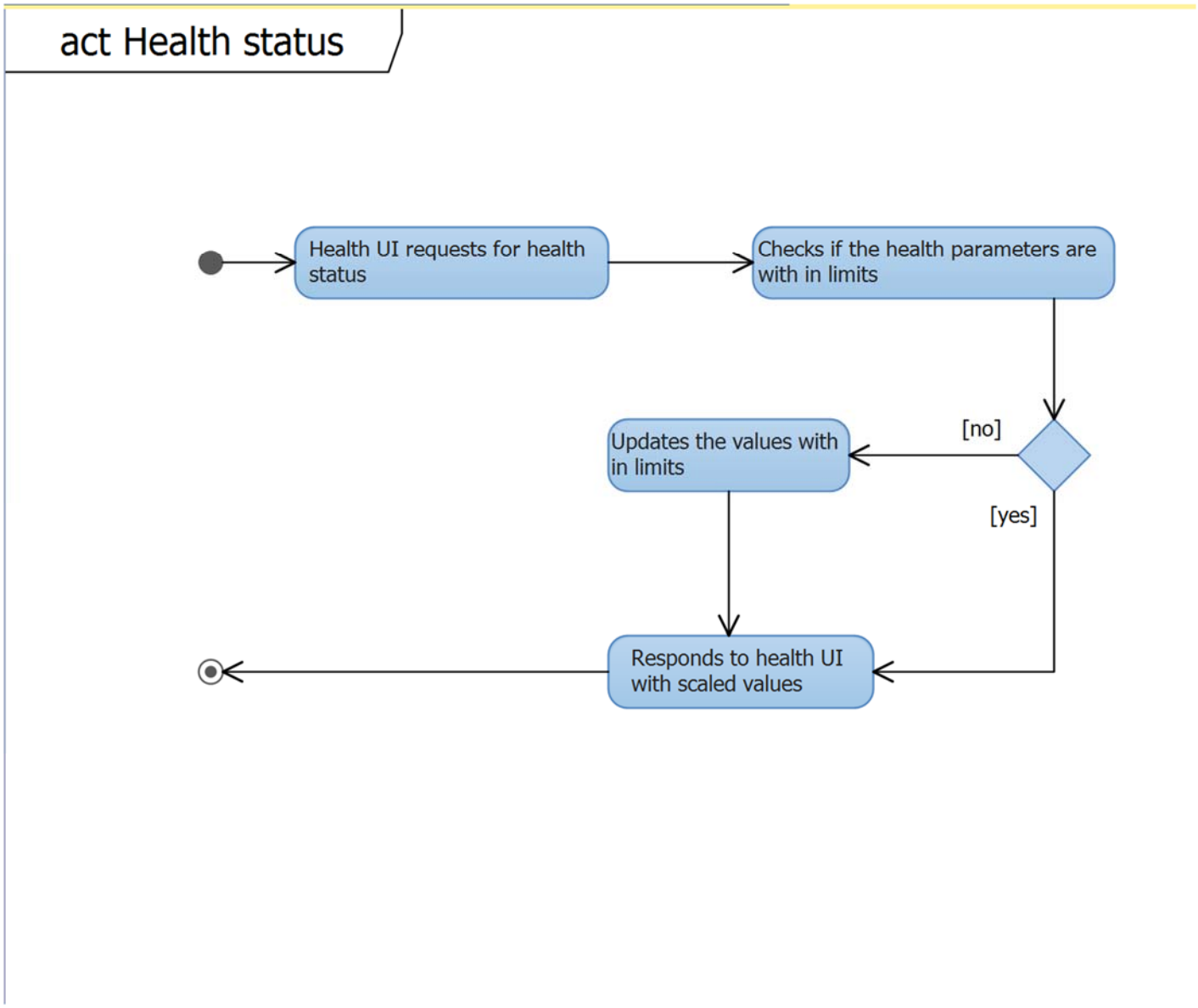


Figure 19: BUSINESS LOGIC ACTIVITY DIAGRAM for Health Status

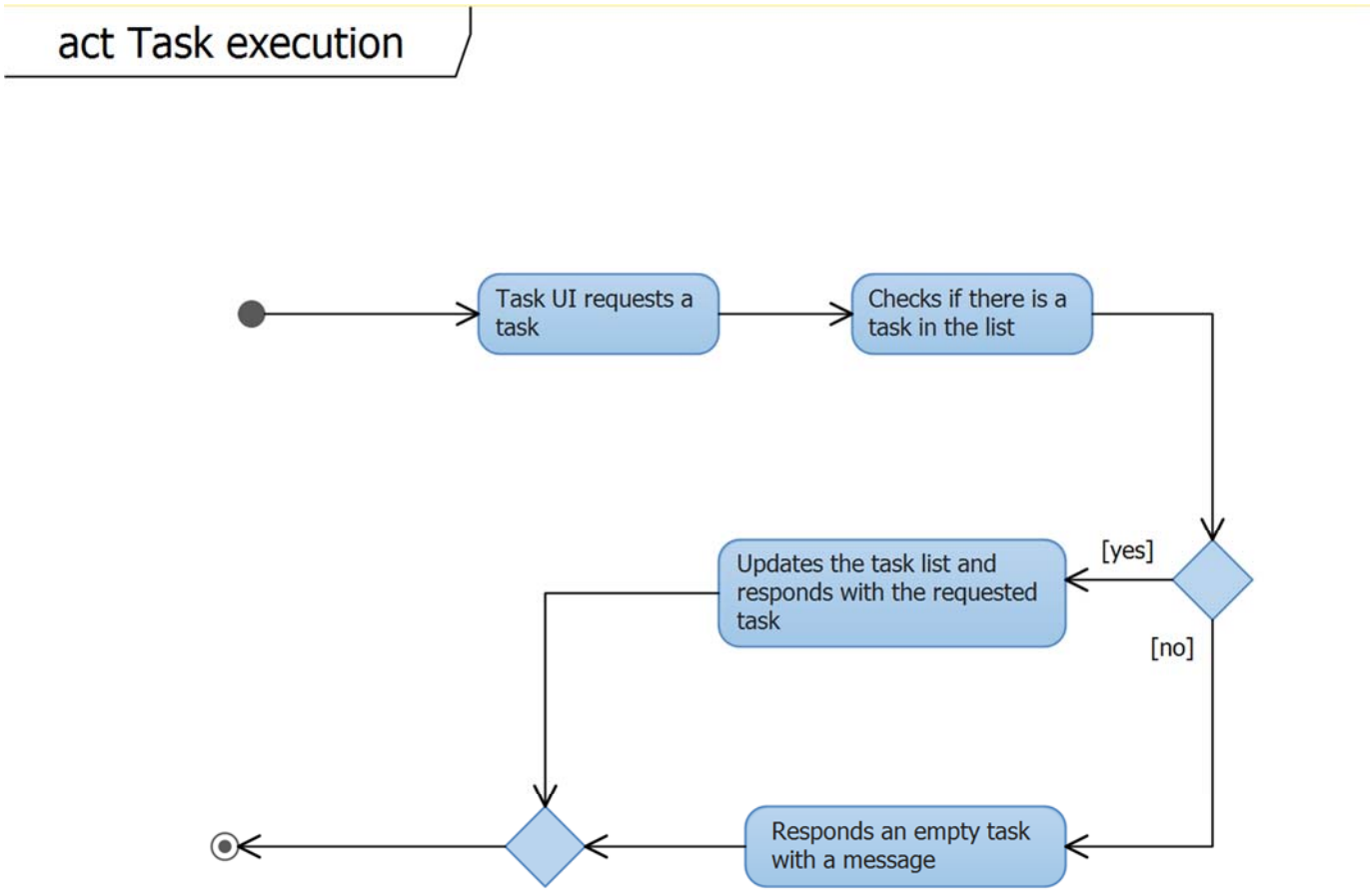


Figure 20: BUSINESS LOGIC ACTIVITY DIAGRAM for Task Update

SEQUENCE DIAGRAM

BUSINESS LOGIC SEQUENCE DIAGRAMS

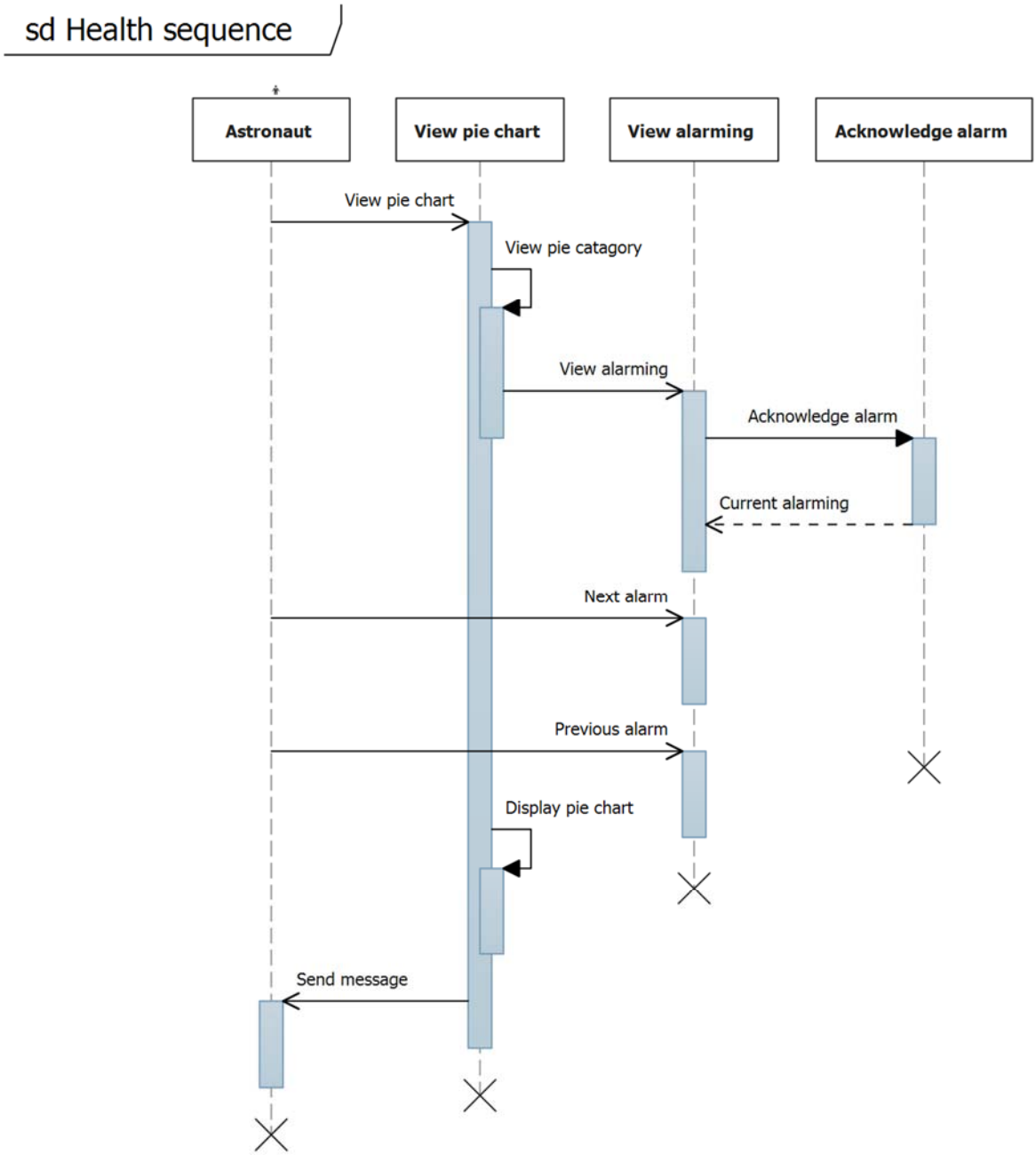


Figure 21: BUSINESS LOGIC SEQUENCE DIAGRAM for Health Status

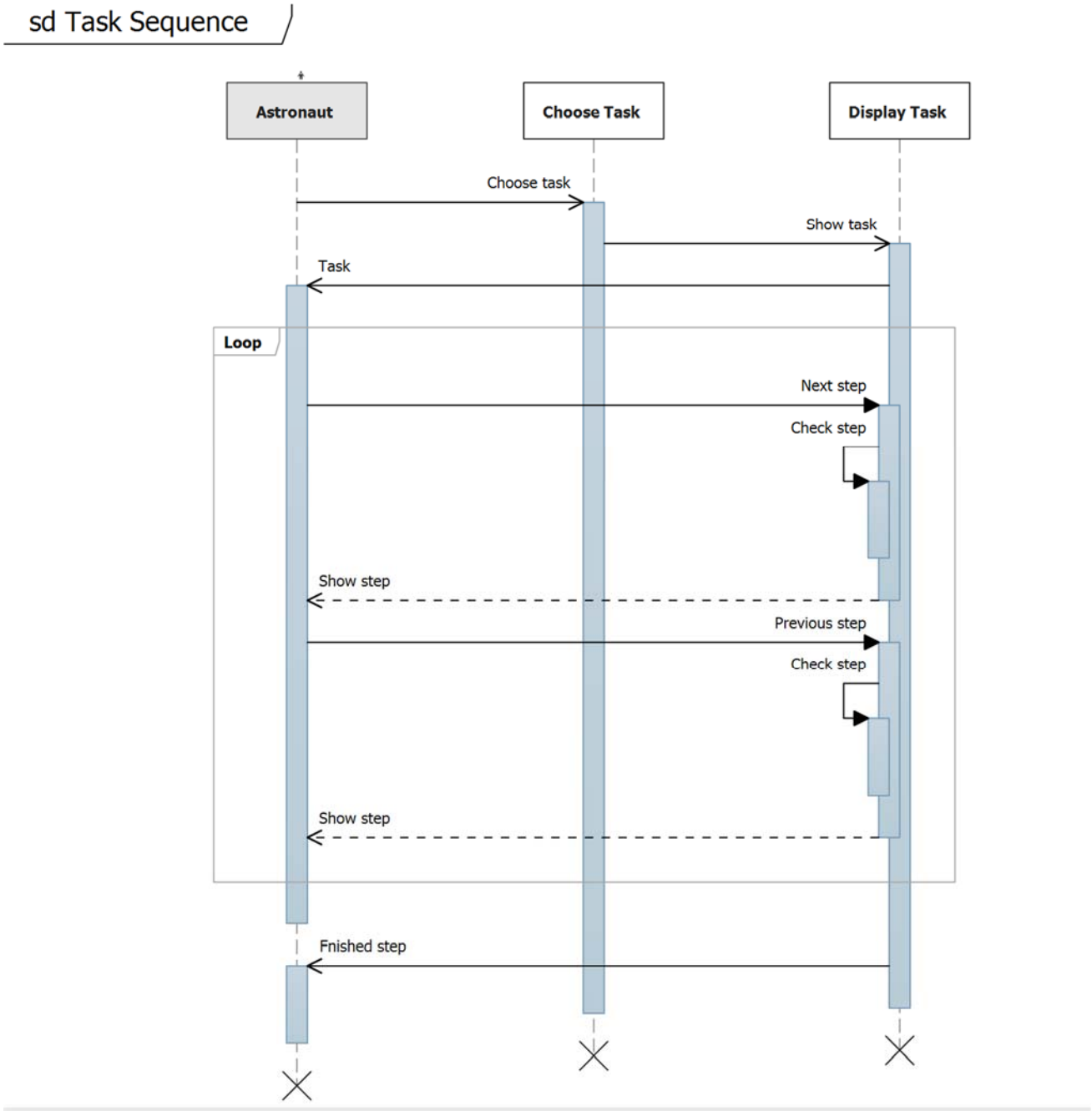


Figure 22: BUSINESS LOGIC SEQUENCE DIAGRAM for Task Sequence

Health UI Sequence Diagram

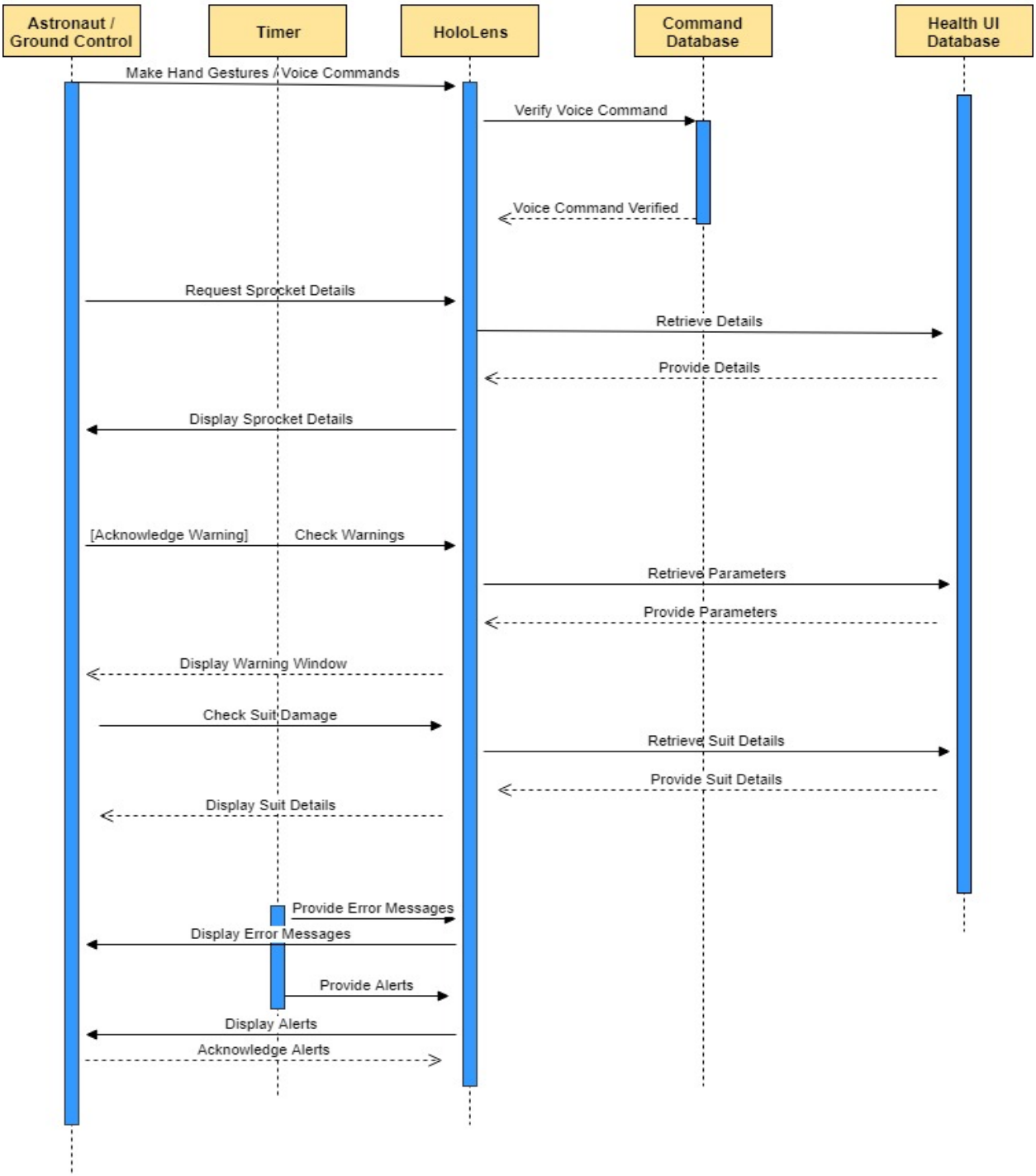


Figure Health UI Sequence Diagram

KAN BAN

Business Logic KAN BAN

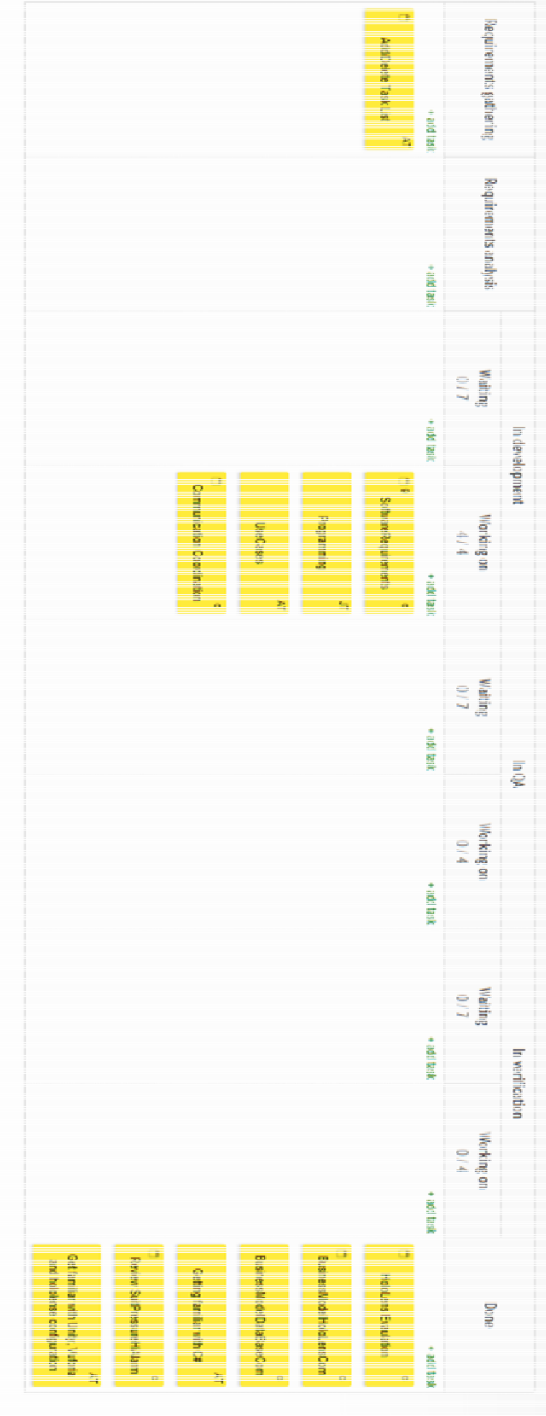


Figure 23: BUSINESS LOGIC KAN BAN

Task UI KAN BAN

COMMUNICATION DATA MAP

Analog Data from Database to Business Logic

	Format	Setpoint	Units	Active
Suit Pressure	XX.X		psi	Yes
Suit Pressure Minimum	XX.X		psi	Yes
Suit Pressure Maximum	XX.X		psi	Yes
Battery Life	XXX		psi	Yes
Battery Life Minimum	XXX	0	%	Yes
Battery Life Maximum	XXX	100	%	Yes
Current Task	XXX		Unitless	Yes
Current Task Minimum	XXX		Unitless	Yes
Current Task Maximum	XXX		Unitless	Yes
O2 Tank1	XX.X		%	Yes
O2 Tank1 Minimum	XX.X		%	Yes
O2 Tank1 Maximum	XX.X		%	Yes
O2 Tank2	XX.X		%	Yes
O2 Tank2 Minimum	XX.X		%	Yes
O2 Tank2 Maximum	XX.X		%	Yes
Suit Temperature				
Suit Temperature Minimum				
Suit Temperature Maximum				
Water				
Water Minimum				
Water Maximum				

Digital Data from System to Astronaut

Suit Pressure Hi Hi	XX.X		psi	Yes
Suit Pressure Hi	XX.X		psi	Yes
Suit Pressure Lo	XX.X		psi	Yes
Suit Pressure Lo Lo	XX.X		psi	Yes
Battery Life Hi Hi	XXX		%	No
Battery Life Hi	XXX		%	No
Battery Life Lo	XXX	30	%	Yes
Battery Life Lo Lo	XXX	15	%	Yes
Current Task Request Hi	XXX		Unitless	Yes
Current Task Request Lo	XXX		Unitless	Yes

Analog Data from Astronaut to System

Digital Data from Astronaut to System

Next Task Command

Database

SOFTWARE REQUIREMENTS

DATABASE

DBA Requirements

DBA shall create the system.

The system that is created will consist of a DB designer, DB administrator and an End-User (business layer).

DBA shall insert or update the tables into the system.

The tables will be created in such a way that the business layer can easily access it at any point of time they can update with any type of values.

DBA shall maintain DB.

The administrator will have all the access to the system that is been created it shall be checked all the time without fail and all the problems shall be updated with the right solution.

DBA shall design the DB.

According to the business layer requirement the DB administrator will design the database in such a way that it will easy to retrieve the data and send the data back.

DBA shall troubleshoot system problems.

When there is problem in retrieving and sending the data to the request made we should have a system in which we can get to know where the exact problem is going on this should be done by the DB administrator so that problem can be solved using certain troubleshooting steps.

DBA shall recover the data, if lost.

The DBA should have a recovery file in which we can have all the data that is been done till the last second in case of any emergency or lost data we can easily retrieve it by using certain troubleshooting steps and data backup should be created when important data is been updated.

Designer Requirements

DB designer shall perform data designing.

The data is designed in form of JSON scripts for task UI and in the form spread sheets for the health UI. The data for the health UI is generated using a C# code that will generated the values automatically from high value to the low value for all the 7 components.

Database

DB designer is responsible for data modelling.

The model that we used is amazon web services. In this we took a component called as the AWS S3 buckets in which we put the task and health UI data in the separate buckets and access is done by the business layer by sending the GET requests and we send reply by the POST request.

DB designer is responsible for creating class.

All the classes that are required by the business layer are given and the classes are created according to the business layer will be created by the DBA.

The designer shall establish relationships among the classes.

The designer will have the tables created and the business layer will have their table. To access these we map our tables to the business layer tables.

The designer shall direct the path of data flow.

The path is created by the flow of information from data layer to the business layer that is mapping the data tables to the business tables the data flow will be consistent.

The designer shall choose the file format in which the data should be stored or fetched.

According to the requirements that we got for health UI we created a spread sheet that consists of all the values of the 7 components from high to low and for the task UI we created a JSON script and we validated the file into .JSON format.

End User Requirements for the Business layer

The user shall request the system for data access.

The user will be business layer for us. The business layer will ask a particular data that is needed by them which they will access the cloud to get the data.

The end user shall respond to the commands issued by Task UI and Health UI.

The Health and task UI are the one who asks the business layer to retrieve the data. These ask them in the form of commands. The commands may be in the form of gestures or in the form of voice.

The user shall retrieve data from system.

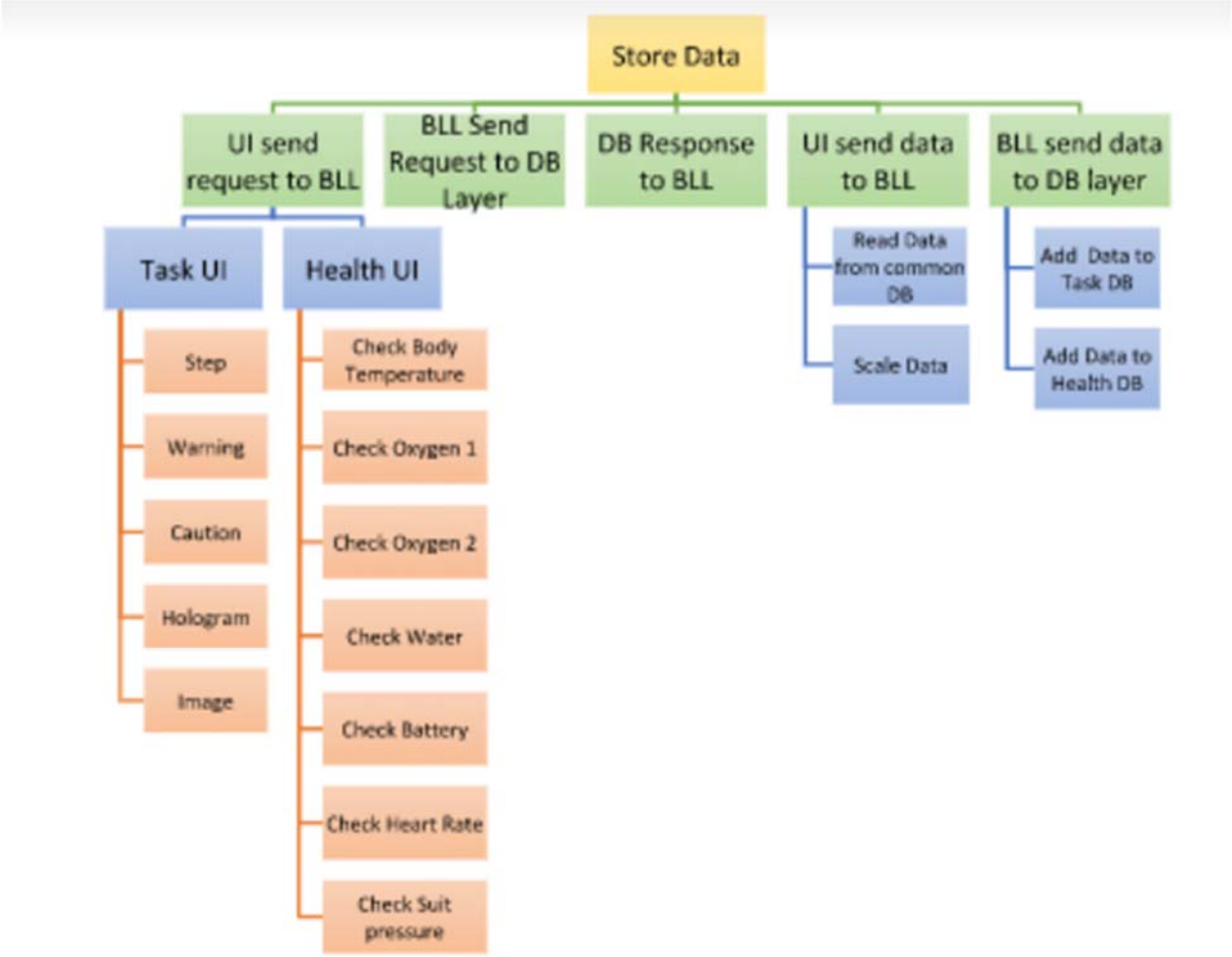
The system is the cloud that we have created to store the data. The business layer have the direct access to the cloud that we have created that make the request and retrieval easy for them.

The user shall limit the highest and lowest value of data as required by the system.

There are certain values that are given to us like “high high, high, low, low low”. We are given 7 components that are to be developed with these values and they are oxygen1, oxygen2, suit pressure, heart rate, battery, body temperature, and water.

Database

Task Analysis



Database

BUSINESS LOGIC

Introduction

The Business Logic for this project provides the middleware for the HoloLens unit. It is an interface between the Database and the User Interface sections of the project. The two User Interface sections of the project are UI Health and UI Tasks. Some of the major functions that are contained in the Business Logic are as follows:

1) Task Execution
2) Analog Scaling and Limit Checking
3) Alarming
4) Alarm Acknowledge and Resetting

Task Execution

The Task Execution software is a used to display instructions for the astronaut. A sequence of instructions are displayed on the HoloLens that guides the Astronaut in performing a procedure or task. The steps for the tasks are stored in the Database provided to the system by NASA. The Business Logic reads the steps of the task and place task data into a buffer for the CURRENT TASK. The tasks are stepped forward and backward based on a NEXT and PRREVIOUS command that is passed from the User Interface Task software. **There may also be a PREVIOUS TASK and NEXT TASK buffer, TBD.** The steps for a task will reside in the database as follows:

IN DATABASE:
Task 1: Disabling Alarm Procedure //(Please follow these steps)

Step	Text	Caution	Warning	Holograms	Images (as per original document step pictures)
1	On the RIGHT side of the EVA Kit, locate the PANEL ACCESS KEY.	The keys are on a tension-spring cable.		panelKeyPart highlight, arrow pointing key	
2	Use the PANEL ACCESS KEY to unlock the PANEL ACCESS DOOR LOCKS.			panelAccessDoor key part highlight, arrow showing direction of key rotation	step1Image.png

3	Carefully return keys to the side of the EVA kit.			panelKeyPart highlight, arrow pointing key part	
4	Insert your fingers in the CENTER OPENING and secure the PANEL ACCESS DOOR in an OPEN position.		Door can accidentally close	centerOpening highlight, arrow showing opening direction	Step3Image.png
5	On your belt, use the BLUE CARABINEER to securely tether to the TETHER CABLE inside the STORAGE.	Notice the TETHER CABLE is adjustable		Arrow pointing towards tether cable	Step4Image.png
6	Locate the E-STOP button and gently press down to temporarily disable the alarm			E-Stop button highlight, arrow pointing downwards	Step5Image.png
7	Locate the FUSIBLE DISCONNECT box.			Fusible disconnect box highlight	
8	Tether the BLUE CARABINEER to the TETHER CABLE.			Arrow pointing towards tether cable	
9	Remove the BLUE CARABINEER from the FUSIBLE DISCONNECT box			Highlight blue carabineer	Step7Image.png
10	Transfer the BLUE CARBINEER to STORAGE			Arrow pointing towards storage	
11	Open the FUSIBLE DISCONNECT box and secure the lid in the open position	Pull the locking tab toward STORAGE with the thumb		Arrow showing opening direction	Step8Image.png
12	Locate the BLACK DISCONNECT			Highlight black disconnect	
13	Tether it to the TETHER CABLE			Arrow pointing towards tether cable	

14	Remove the DISCONNECT	Pull up with the index and middle fingers while pushing down on the FUSE ACCESS PANEL with the thumb		Highlight disconnect , arrow pointing upwards	Step10Image.png
15	Place the DISCONNECT in the STORAGE			Arrow pointing towards storage	
16	Tether the FUSE ACCESS PANEL to the TETHER CABLE			Fuse access panel highlight, arrow pointing tether cable	
17	Remove the FUSE ACCESS PANEL by pulling straight up			Fuse access panel highlight, arrow pointing upwards	Step12Image.png
18	Place the FUSE ACCESS PANEL into STORAGE			Arrow pointing towards storage	
19	Tether the ALARM FUSE to the TETHER CABLE			Alarm fuse highlight, arrow pointing tether cable	
20	In Storage, locate the BLUE FUSE PULLER			Highlight blue fuse puller	
21	Use the BLUE FUSE PULLER to remove ONLY the ALARM FUSE	Rock the ALARM FUSE with the FUSE PULLER when pulling up		Alarm fuse highlight	Step16Image.png
22	Return the ALARM FUSE and the FUSE PULLER to STORAGE			Arrow pointing storage	

23	In STORAGE, locate the FUSE ACCESS PANEL			Fuse access panel highlight	
24	Reinstall it into the FUSIBLE DISCONNECT box			Highlight disconnect box, arrow pointing downwards	Step18Image.png
25	Remove the FUSE ACCESS PANEL tether from the TETHER CABLE and stow inside		All tethers are under spring tension and can retract quickly	Highlight fuse access panel	
26	In STORAGE, locate the DISCONNECT			Highlight disconnect	
27	Reinstall it into the FUSIBLE DISCONNECT box	The DISCONNECT must read “ON” in the upper right corner to restore conductivity		Highlight disconnect box, arrow pointing downwards	Step20Image.png
28	Remove the DISCONNECT tether from the TETHER CABLE		All tethers are under spring tension and can retract quickly	Highlight disconnect tether or arrow pointing	
29	Close the FUSIBLE DISCONNECT box cover			Highlight fusible disconnect box	
30	In STORAGE, use the BLUE CARABINEER to clip and lock the FUSIBLE DISCONNECT box cover.			Highlight blue carabineer	Step23Image.png
31	Remove the BLUE CARABINEER’s		All tethers are under spring tension and	Arrow pointing tether cable or highlight tether	

	tether from the TETHER CABLE		can retract quickly. End of Disabling Alarm Procedure		

Task 2: Rerouting Power Procedure: //(Procedure after final steps is to be discussed)

Step	Text	Caution	Warning	Holograms	Images (as per original document step pictures)
1	Locate the Aux. Power Input			Highlight aux, power input	
2	Locate BATTERY PACK and tether to TETHER CABLE			Highlight battery pack and arrow pointing tether cable	
3	Undo the BATTERY PACK LEADS from the AUX. POWER INPUT	Depress the red and black plastic hammers on the side of the AUX. POWER INPUT and pull the leads straight up		Highlight battery pack leads	Step3Image.png
4	Remove BATTERY PACK from AUX. POWER INPUT			Arrow pointing upwards and highlight	
5	Locate the ON/OFF switch on the back of the BATTERY PACK and switch it to the OFF position.			Highlight on/off and arrow showing off direction	Step5Image.png

6	Place the BATTERY PACK into STORAGE			Arrow pointing storage	
7	In STORAGE, find the replacement BATTERY PACK			Highlight battery pack	
8	Locate the ON/OFF switch on the back of the BATTERY PACK and switch it to the ON position			Arrow showing on direction and highlight	
9	Attach the replacement BATTERY PACK onto the AUX. POWER INPUT by the Velcro			Highlight aux.power input and arrow pointing downwards	
10	Insert the BATTERY PACK leads back into same colored ports	Depress the red and black plastic hammers on the side of the AUX. POWER INPUT and push leads straight into their ports		Highlight	
11	Conduct a GENTLE PULL TEST on the wires			Arrow pointing outwards	
12	Remove the BATTERY PACK tether from the TETHER CABLE		All tethers are under spring tension and can retract quickly	Highlight tether	
13	In STORAGE, locate the GRAY 220 Volt PLUG			Highlight the grey 220v plug	

14	Install it into the POWER OUT	Outlet and plug mate are stiff, ensure the full engagement of the plug into the outlet		Highlight power out	Step12Image.png
15	Locate the metal BUSS BAR and verify there are BLACK, GREEN & WHITE BUSSES, each with 2 openings			Highlight metal bus bar	
16	Insert the WHITE 220 VOLT LEAD into the LEFT WHITE BUSS opening and GENTLY TIGHTEN the thumbscrew	DO NOT over tighten the thumbscrew		Highlight white 200v lead and arrow showing tightening direction	Step16Image.png
17	Insert the GREEN 220 VOLT LEAD into the LEFT GREEN BUS opening	DO NOT over tighten the thumbscrew		Highlight green 200v lead	Step17Image.png
18	Insert the BLACK 220 VOLT LEAD into the LEFT BLACK BUS opening	DO NOT over tighten the thumbscrew		Highlight black 200v lead	Step18Image.png
19	Make sure the METAL LEADS are not sticking out the BACK of the BUSS BAR			Highlight metal leads	
20	Conduct a Gentle PULL TEST on each cable			Arrow pointing outwards	
21	In STORAGE, locate the 110 VOLT PLUG			Highlight 110 v plug	
22	Install it into POWER IN	Lift cover with one hand while installing PLUG into the outlet with the other. The lid		Highlight power in	Step21Image.png

		is spring-loaded			
23	Insert the WHITE 110 VOLT PLUG LEAD into the RIGHT WHITE BUS opening	DO NOT over tighten the thumbscrew		Highlight white bus opening	Step22Image.png
24	Insert the GREEN 110 VOLT PLUG LEAD into the RIGHT GREEN BUS opening	DO NOT over tighten the thumbscrew		Highlight green bus opening	Step23Image.png
25	Insert the BLACK 110 VOLT PLUG LEAD into the RIGHT BLACK BUS opening	DO NOT over tighten the thumbscrew		Highlight black bus opening	Step24Image.png
26	Conduct a Gentle PULL TEST on each cable			Arrow pointing outwards	
27	In STORAGE, locate the E-STOP KEY			Highlight estop key	Step25Image.png
28	Insert the KEY into the E-STOP and TURN to the RIGHT and the button will pop up			Highlight e-stop and arrow showing right direction	Step26Image.png
29	Remove the KEY and place it in STORAGE			Arrow pointing storage	
30	Locate the AUX. POWER SWITCH on the POWER IN box and switch it to the “ON” position			Highlight aux power switch and arrow pointing on direction	Step28Image.png
31	Can you please confirm YES or NO that the SYSTEM GO indicator light is GREEN?			System go indicator part hilight	lastImage.png
32	IF GREEN, read this script: EVA 1, this is Houston Mission Control.				LastImage.png

	Congratulations. PHALCON is reporting that they are reading a successful power restoration on their console. You are a go to untether from the TETHER CABLE and return to space station. Mission Control out				
33	IF NOT GREEN, read this script: EVA 1, this is Houston Mission Control. PHALCON confirms and is not able to report a successful power restoration on their console. EVA-1, please be advised that we have prepared some trouble-shooting steps for you to conduct on a future spacewalk			//Should go to troubleshooting procedure 2.	LastImage.png

TABLE: BusinessLogic1 – This is the first five tasks as shown in the database for the Tasks as displayed in TABLE:BusinessLogic3.

IN THE BUSINESS LOGIC:
From Table:BusinesLogic1, if Step = 3 then below is the PREVIOUS, CURRENT and NEXT steps that are read by the USER INTERFACE TASKS for displaying on the HoloLens.

Step	Text	Caution	Warning	Holograms	Images
PREVIOUS	Carefully return keys to the side of the EVA kit.				
CURRENT	Insert your fingers in the CENTER OPENING and secure the PANEL ACCESS DOOR in an OPEN position.		Door can accidentally close.	PanelAccessDoorAnimaiton, OpenArrow	Step3Image.png
NEXT	On your belt, use the BLUE CARABINEER to securely tether to the TETHER CABLE inside the STORAGE.	Notice the TETHER CABLE is adjustable		blueCarabineerAnimation	Step4Image.png

TABLE: BusinessLogic2 - The USER INTERFACE TASK will show PREVIOUS, CURRENT and NEXT tasks

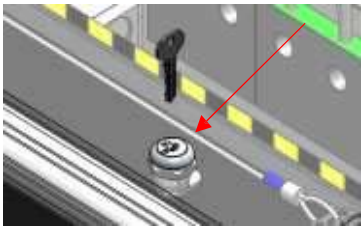
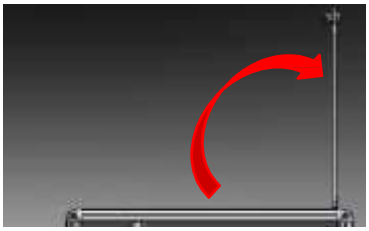
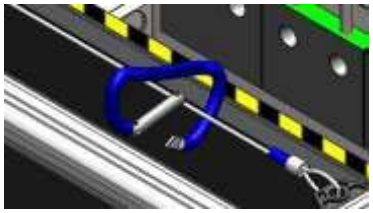
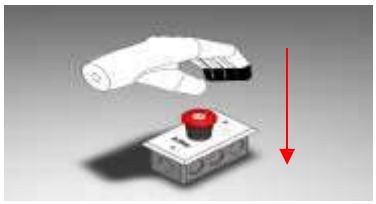
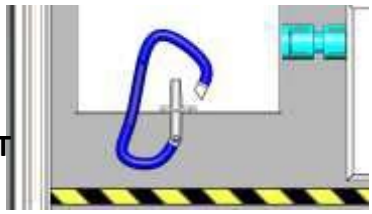
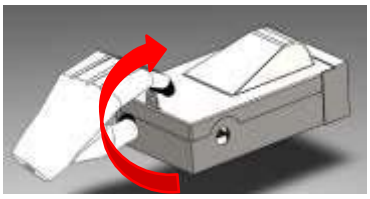
1. On the **RIGHT** side of the **EVA Kit**, locate and use the **PANEL ACCESS KEY** to unlock the **PANEL ACCESS DOOR LOCKS.** **CAUTION:** The keys are on a tension-spring cable.	1. 
2. Carefully return keys to the side of the EVA kit.	
3. Insert your fingers in the **CENTER OPENING** and secure the **PANEL ACCESS DOOR** in an **OPEN** position. **WARNING:** Door can accidentally close.	3. 
4. On your belt, use the **BLUE CARABINEER** to securely tether to the **TETHER CABLE** inside the **STORAGE**. **CAUTION:** Notice the TETHER CABLE is adjustable	4. 
5. Locate the **E-STOP** button and gently press down to temporarily disable the alarm.	5. 
6. Locate the **FUSIBLE DISCONNECT** box and tether the **BLUE CARABINEER** to the **TETHER CABLE**.	
7. Remove the **BLUE CARABINEER** from the **FUSIBLE DISCONNECT** box and transfer it to **STORAGE**.	7. 
8. Open the **FUSIBLE DISCONNECT** box and secure the lid in the open position. **CAUTION:** Pull the locking tab toward **STORAGE** with the index finger while lifting the cover with the thumb.	8. 

TABLE: BusinessLogic3 – From "FY18 NASA SUITS Design Challenge Description Revised_11.28.17"

Analog Scaling and Limit Checking

This section of software for Analog Scaling and Limit Checking will reside in the Business Logic. The Business Logic will read the analog value from the Database software. There is a maximum and minimum value for every analog value. The maximum and minimum value for each analog value will be part of the Database. The business Logic will use the Maximum and Minimum values will place limits on the analog value that is read from the Database. See the example below for **BL_SuitPressure** with the Minimum value **BL_SuitPressure** for at 0psi and Maximum value for **BL_SuitPressure** at 20psi.

BL_SuitPressureDB	BL_SuitPressureMin	BL_SuitPressureMax	BL_SuitPressure
-1.0	0.0	20.0	0.0
0.0	0.0	20.0	0.0
1.0	0.0	20.0	1.0
2.0	0.0	20.0	2.0
5.0	0.0	20.0	5.0
10.0	0.0	20.0	10.0
15.0	0.0	20.0	15.0
19.0	0.0	20.0	19.0
20.0	0.0	20.0	20.0
21.0	0.0	20.0	20.0

BL_SuitPressure is used in the UI, User Interface for displaying on the HoloLens. **TBD, we may add an alarm to let us know when the variable information from the Database is outside of the Min and Max ranges.**

Alarming

The Business Logic accesses Analog values as read from the database. Each analog value is tested for going above at two levels Hi and HiHi and below two levels of detection Lo, LoLo. There is a deadband for all four levels of alarm detection. Any of the four analog alarm levels may be enabled or disabled. The Alarm Acknowledge and Resetting section of software is the area that the active alarms are stored are read from for display on the UI, User Interface.

Analog Alarm Detection

The Alarm software evaluates an analog value for limits. There is a check for at four levels, HiHi, Hi, Lo, LoLo. The alarms are activated as the analog value increase above the Hi and HiHi limits and activates when the analog value decreases below the Lo and LoLo limits. There is a deadband for all four levels of alarm detection. The purpose of the deadband is to minimize oscillation around an alarm setpoint thus creating nuisance alarms. **Note: This information is NOT read from the UI alarming to display alarms. The UI Alarming reads the actual active alarms in Alarm Acknowledgement and Resetting software**

INPUT VARIABLES for the Analog Alarm Detection are as follows:

AnalogInput	The analog value that is being monitored
AnalogInputHiHi	The analog setpoint for HiHi which creates the alarm condition
AnalogInputHiHiDB	The differential amount that the analog value must go below to be NOT in HiHi alarm
AnalogInputHiHiAEn	The flag that indicates the HiHi alarm is being used. 1 = Enabled, 0=Disabled
AnalogInputHi	The analog setpoint for Hi
AnalogInputHiDB	The differential amount that the analog value must go below to be NOT in Hi alarm
AnalogInputHiEn	The flag that indicates the Hi alarm is being used. 1 = Enabled, 0= Disabled
AnalogInputLo	The analog setpoint for Lo
AnalogInputLoDB	The differential amount that the analog value must go above to be NOT in Lo alarm
AnalogInputLoEn	The flag that indicates the Lo alarm is being used. 1 = Enabled, 0= Disabled
AnalogInputLoLo	The analog setpoint for LoLo
AnalogInputLoLoDB	The differential amount that the analog value must go above to be NOT in LoLo alarm
AnalogInputLoLoEn	The flag that indicates the LoLo alarm is being used. 1 = Enabled, 0= Disabled

OUTPUT VARIABLES for the Analog Alarm Detection are as follows:

AnalogInputHiHiStatus	This is the HiHi flag that is set when an alarm is active. 1 = Active, 0 = Not Active
AnalogInputHiStatus	This is the Hi flag that is set when an alarm is active. 1 = Active, 0 = Not Active
AnalogInputLoStatus	This is the Lo flag that is set when an alarm is active. 1 = Active, 0 = Not Active
AnalogInputLoLoStatus	This is the LoLo flag that is set when an alarm is active. 1 = Active, 0 = Not Active

Example: SuitPressure

INPUT VARIABLES for the Analog Alarm Detection for SuitPressure are as follows:

Generic Name	Actual Variable Name	Example value
AnalogInput	BL_SuitPressure	0.0 to 20.0 psi
AnalogInputHiHiSP	BL_SuitPressureHiHiSP	18.0 psi
AnalogInputHiHiDB	BL_SuitPressureHiHiDB	17.8 psi
AnalogInputHiHiEn	BL_SuitPressureHiHiEn	1 = Enabled
AnalogInputHiSP	BL_SuitPressureHiSP	16.0 psi
AnalogInputHiDB	BL_SuitPressureHiDB	15.8 psi
AnalogInputHiEn	BL_SuitPressureHiEn	1 = Enabled
AnalogInputLoSP	BL_SuitPressureLoSP	4.0 psi
AnalogInputLoDB	BL_SuitPressureLoDB	4.2 psi
AnalogInputLoEn	BL_SuitPressureLoEn	1 = Enabled
AnalogInputLoLoSP	BL_SuitPressureLoLoSP	3.0 psi
AnalogInputLoLoDB	BL_SuitPressureLoLoDB	3.2 psi
AnalogInputLoLoEn	BL_SuitPressureLoLoEn	1 = Enabled

OUTPUT VARIABLES for the Analog Alarm Detection are as follows:

Generic Name	Actual VariableName	Example value
AnalogInputHiHiStatus	BL_SuitPressureHiHiStatus	1 = Active, 0 = Not Active
AnalogInputHiStatus	BL_SuitPressureHiStatus	1 = Active, 0 = Not Active
AnalogInputLoStatus	BL_SuitPressureLoStatus	1 = Active, 0 = Not Active
AnalogInputLoLoStaus	BL_SuitPressureLoLoStaus	1 = Active, 0 = Not Active

Sample of varying pressure for the BL_SuitPressure with resultant alarm activation for the 4 alarm levels:

BL_SuitPressure (psi)	BL_SuitPressure HiHiStatus	BL_SuitPressure HiStatus	BL_SuitPressure LoStatus	BL_SuitPressure LoLoStaus
0.0	0	0	1	1
1.0	0	0	1	1
2.0	0	0	1	1
3.0	0	0	1	1
3.1	0	0	1	1
3.2	0	0	1	1
3.3	0	0	1	0
4.0	0	0	1	0
4.1	0	0	1	0
4.2	0	0	1	0
4.3	0	0	0	0
10.0	0	0	0	0
15.0	0	0	0	0
15.8	0	0	0	0
16.0	0	1	0	0
17.0	0	1	0	0
17.8	0	1	0	0
18.0	1	1	0	0
19.0	1	1	0	0
18.0	1	1	0	0
17.8	1	1	0	0
16.0	0	1	0	0
15.8	0	1	0	0
15.0	0	0	0	0
10.0	0	0	0	0

Alarm Acknowledgement and Resetting

This software determines the flags for ACTIVE alarms and is the data used by the UI Alarming to determine the status to display an alarm on the UI, User Interface. All alarms are inputs to this section of code. Any alarm that is active will have a 1 as status to indicate it as an active alarm. This flag is held to a 1 state until acknowledged by the Astronaut via the UI.

The BL_SuitPressure (continued) is an example of the display of an alarm on the UI Alarm Display. The Alarm UI will display the Hi alarm based on the BL_SuitPressureHiAlarm flag which is generated by the Alarm Acknowledgement and Resetting software in the Business Logic software.

Example: BL_SuitPressure (continued)

BL_SuitPressure (psi)	BL_SuitPressure HiStatus	EUI_AlarmAck	BL_SuitPressure HiAlarm	ActiveDisplay
15.0	0	0	0	Not Displayed
15.8	0	0	0	Not Displayed
16.0	1	0	1	Displayed
17.0	1	0	1	Displayed
16.0	1	0	1	Displayed
15.8	1	0	1	Displayed
15.0	0	0	1	Displayed
15.8	0	0	1	Displayed
16.0	1	0	1	Displayed
17.0	1	0	1	Displayed
16.0	1	1	1	Displayed
15.8	1	0	1	Displayed
15.0	0	0	0	Not Displayed

BL_SuitPressureHiStatus: generated from the Analog Alarm Detection software.
AlarmAck: generated from the Alarm UI software.
BL_SuitPressureHiAlarm: generated Alarm Acknowledgement and Resetting software. Used by Alarm UI for alarm display.

PSUEDO-CODE for Alarm Acknowledge and Resetting example

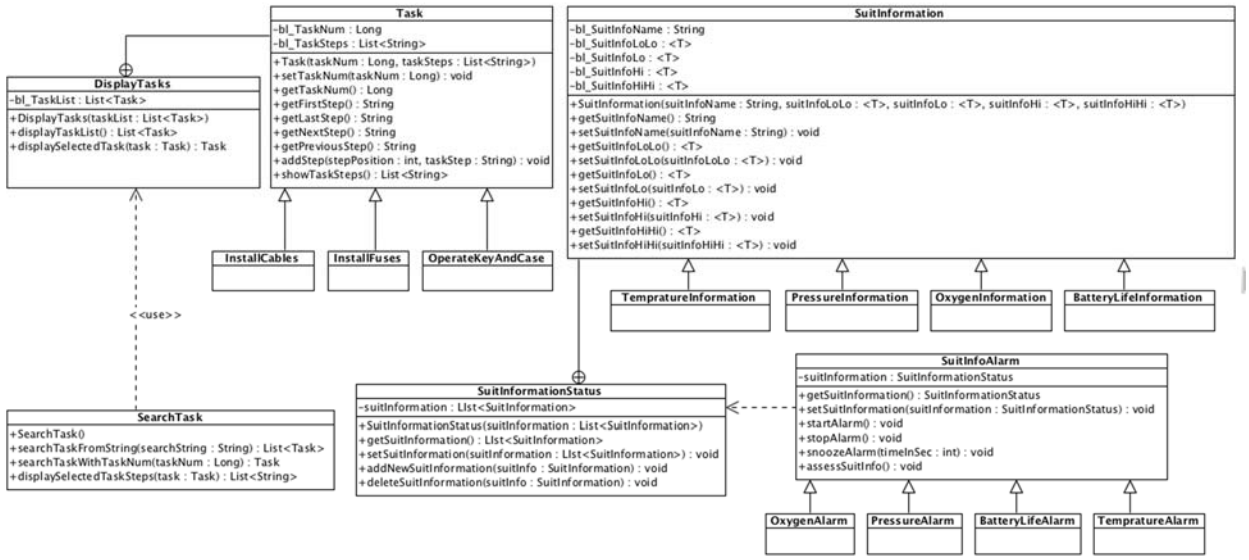
```
If (BL_SuitPressureHiStatus .GT. BL_SuitPressureHiStatusOLD) then BL_SuitPressureHiStatusONS = TRUE
If (BL_SuitPressureHiStatus .EQ. BL_SuitPressureHiStatusOLD) then BL_SuitPressureHiStatusONS = FALSE
BL_SuitPressureHiStatusOLD = BL_SuitPressureHiStatus

BL_SuitPressureHiTEMP = ( BL_SuitPressureHiTEMP .AND. NOT( EUI_AlarmAck ) .OR.
BL_SuitPressureHiONS

BL_SuitPressureHiAlarm = BL_SuitPressureHiTEMP .OR. BL_SuitPressureHiStatus
```

Alarm List

Below is a list of alarms and setpoints. **At this time setpoints are, TBD, to be determined.**

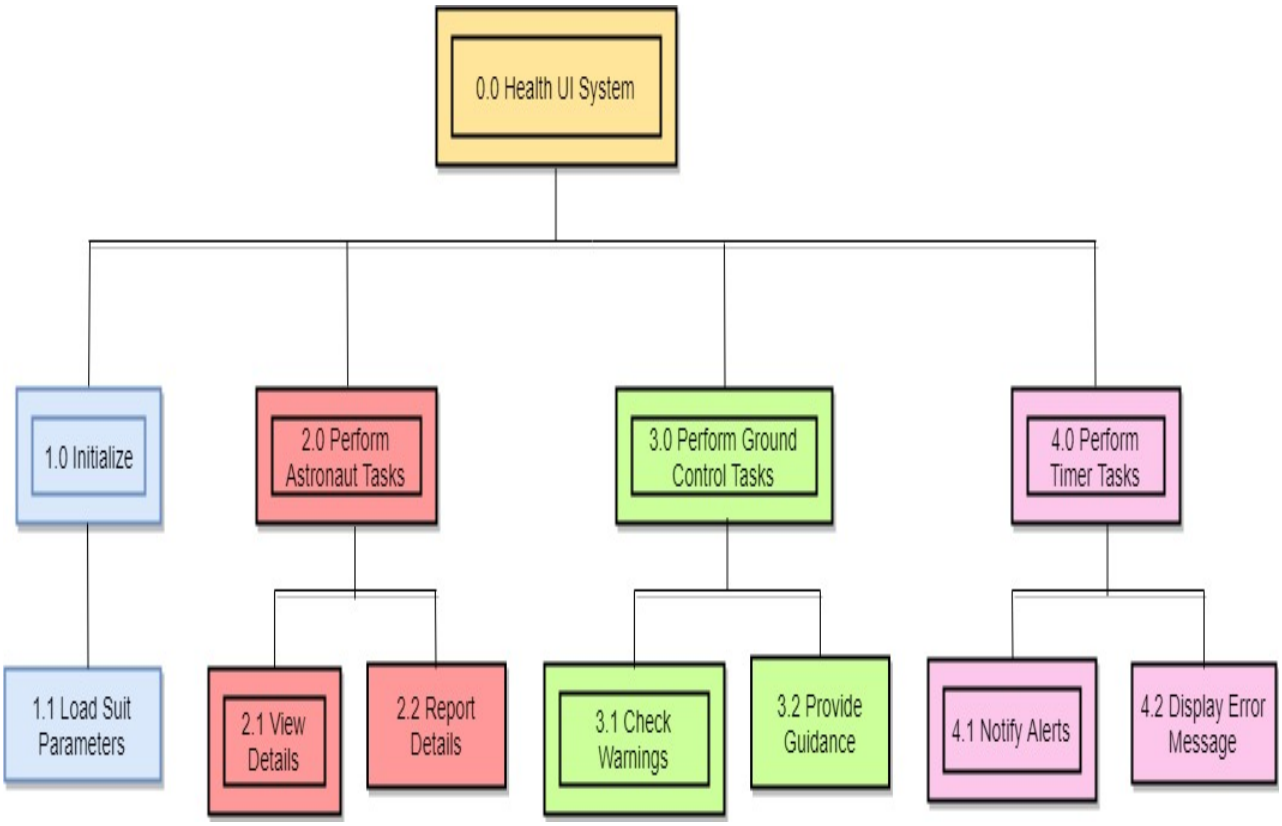
Alarm Variable Name	Setpoint	Description for Alarm Banner
BL_SuitPressureLoLo	3.9	Suit Pressure Lo Lo
BL_SuitPressureLo	4.1	Suit Pressure Lo
BL_SuitPressureHi	4.99	Suit Pressure Hi
BL_SuitPressureHiHi	5.2	Suit Pressure Hi Hi
BL_BatteryLifeLoLo	15	Battery Life Lo Lo
BL_BatteryLifeLo	30	Battery Life Lo
BL_O2Tank1Lo	255	O2 Tank1 Lo
BL_O2Tank2LoLo	900	O2 Tank2 Lo Lo
BL_O2Tank2Lo	1440	O2 Tank2 Lo
BL_O2Tank2Hi	3575	O2 Tank2 Hi
BL_O2Tank2HiHi	3600	O2 Tank2 Hi Hi
BL_SuitTempLoLo	90	Suit Temperature Lo Lo
BL_SuitTempLo	93	Suit Temperature Lo
BL_SuitTempHi	106.99	Suit Temperature Hi
BL_SuitTempHiHi	111	Suit Temperature Hi Hi
BL_WaterLevelLoLo	1.4	Water Level Lo Lo
BL_WaterLevelLo	2.1	Water Level Lo

UML Diagram for Business Logic



Health UI Reqiuirements

1. **Task Hierarchical Diagram**



Figure; Hierarchical diagram of tasks showing all the Health UI tasks

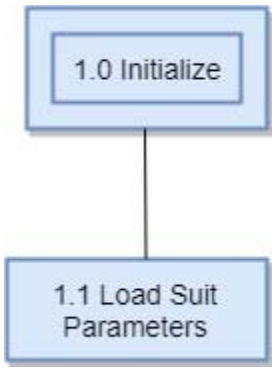


Figure: Hierarchical diagram of sub-system Initialize

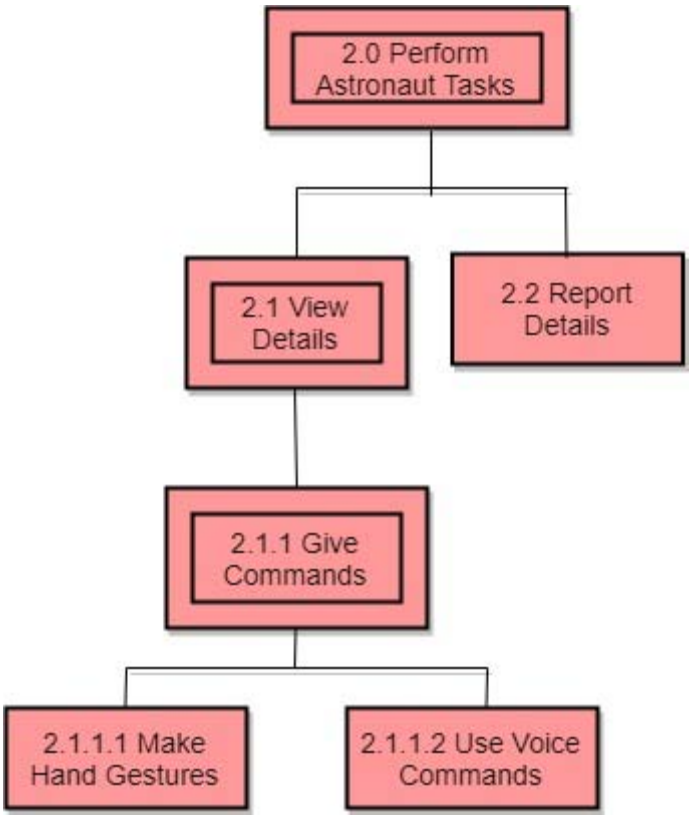


Figure: Hierarchical diagram of sub-system Perform Astronaut Tasks

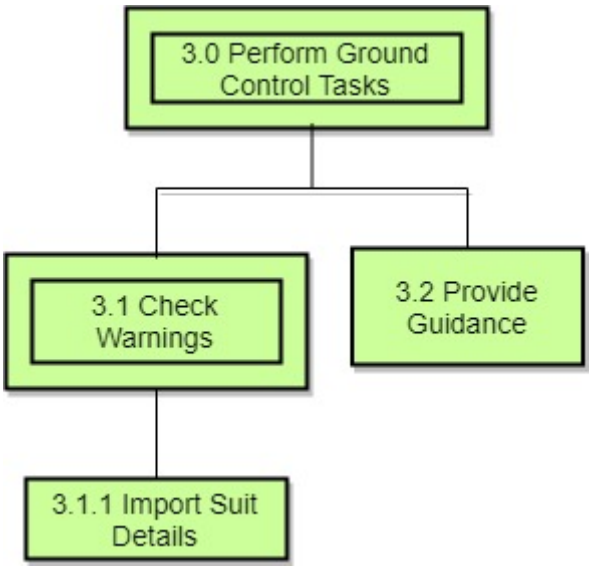


Figure: Hierarchical diagram of sub-system Perform Ground Control Tasks

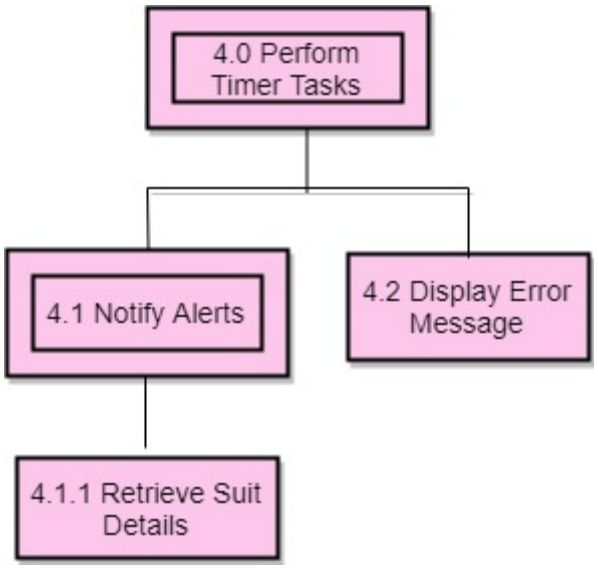


Figure: Hierarchical diagram of sub-system Perform Timer Tasks

2. **Task Analysis Specification**

Task – 0.0 Health UI System

1. **What is the goal of this task.**

The goal of this task is to provide health details of the Astronaut from various suit parameters.

2. **What sub-tasks define this task.**

The sub-tasks that define this task are as follows:
1.0 Initialize
2.0 Perform Astronaut Tasks
3.0 Perform Ground Control Tasks.
4.0 Perform Timer Tasks.

3. **Is this task a subunit of larger task.**

No.

4. **What non-interface functions does this task require.**

None.

5. **What kinds of inputs or actions does this task require from the user.**

The user is required to open the Health UI Application on the HoloLens.

Health UI

6. **What kinds of outputs / results are expected by performing this task.**

By performing this task, the Health details of the Astronaut gets displayed on the HoloLens Screen.

7. **What automatic actions does this task expect from the system.**

None.

8. **What special characteristics of this task should we record.**

None.

9. **In this subtree, is there a task that must come before this one.**

No.

10. **In this subtree, is there a task for which this one is required to be immediately preceding**. **Is there any specific sequence in which the tasks must be performed.**

No.

11. **Which, if any, primary classes / entities are involved in this subtask.**

The Database.

12. **How can this task fail (or end in non-completion).**

This task can fail if there is no Internet connection. This task may also end in non-completion if the application is not deployed appropriately.

13. **How frequently is this task performed.**

This task is performed whenever the Astronaut makes use of the Health parameters.

14. **How open is this task, especially in terms of its sequence or inputs.**

This task is open every time when the Astronaut is using the HoloLens.

15. **What if any are the specific usability expectations (e.g. ease of use) for this task and how** do we anticipate determining if we have satisfied the user expectations. (Hint: Like ease of use, ease of learning. As you specify this, be specific as to what "ease of use" and other usability expectations mean and how you will evaluate these).

Ease of use (How well the Astronaut can understand the user interface and perform various tasks).
Ease of use (How better the user can understand the Health UI and perform various tasks).

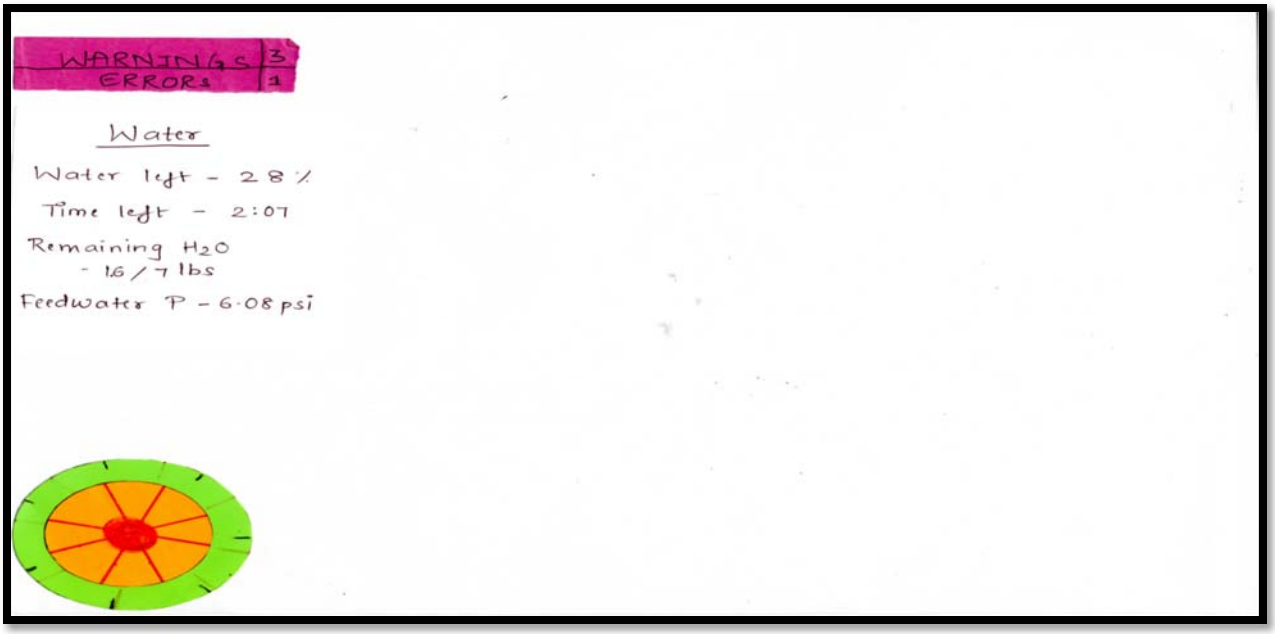
1. **Initial Paper Prototype**

Initial Screen

View Pie Details Screen

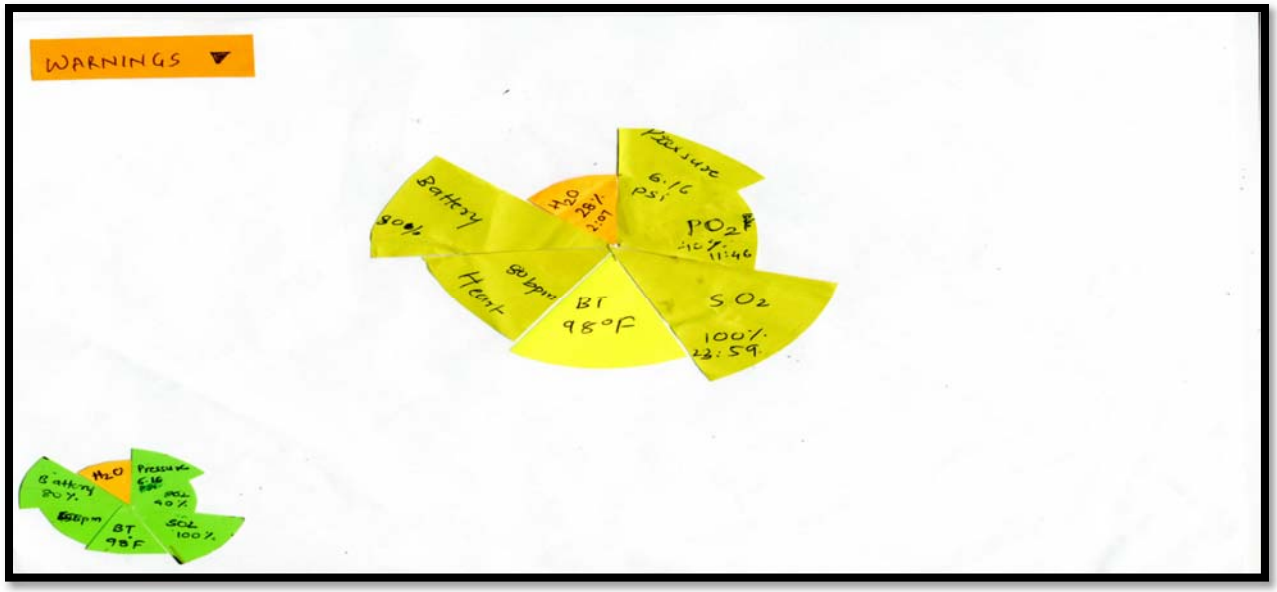
View Consumable Details (Water)

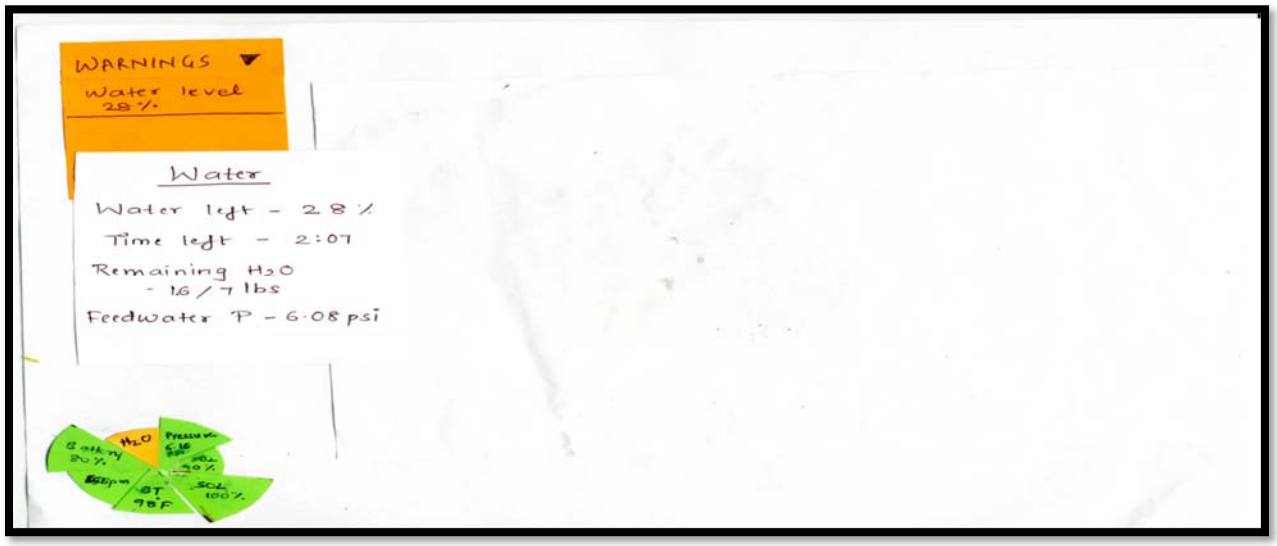
Health UI

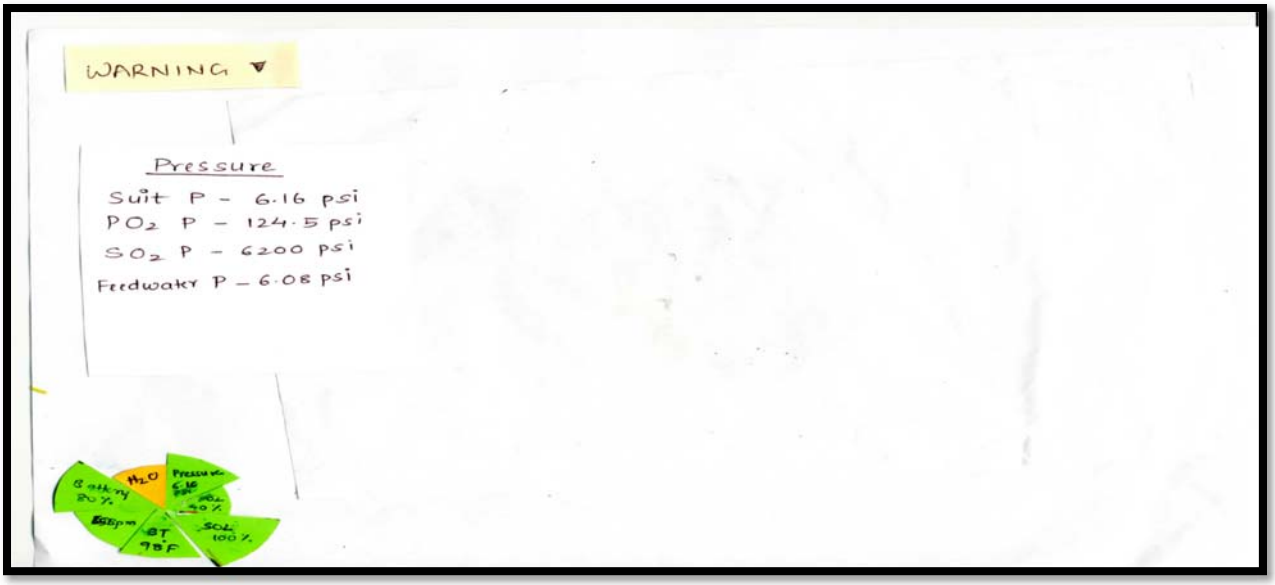


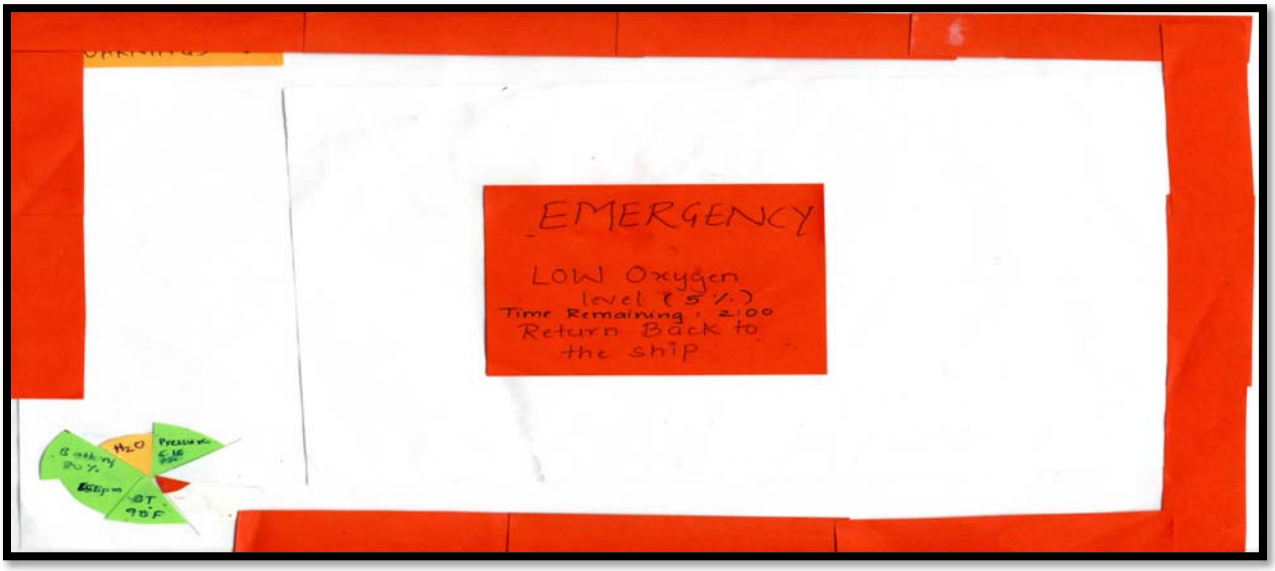
Emergency Alert: Oxygen Low

2. Paper Prototype After Usability Study

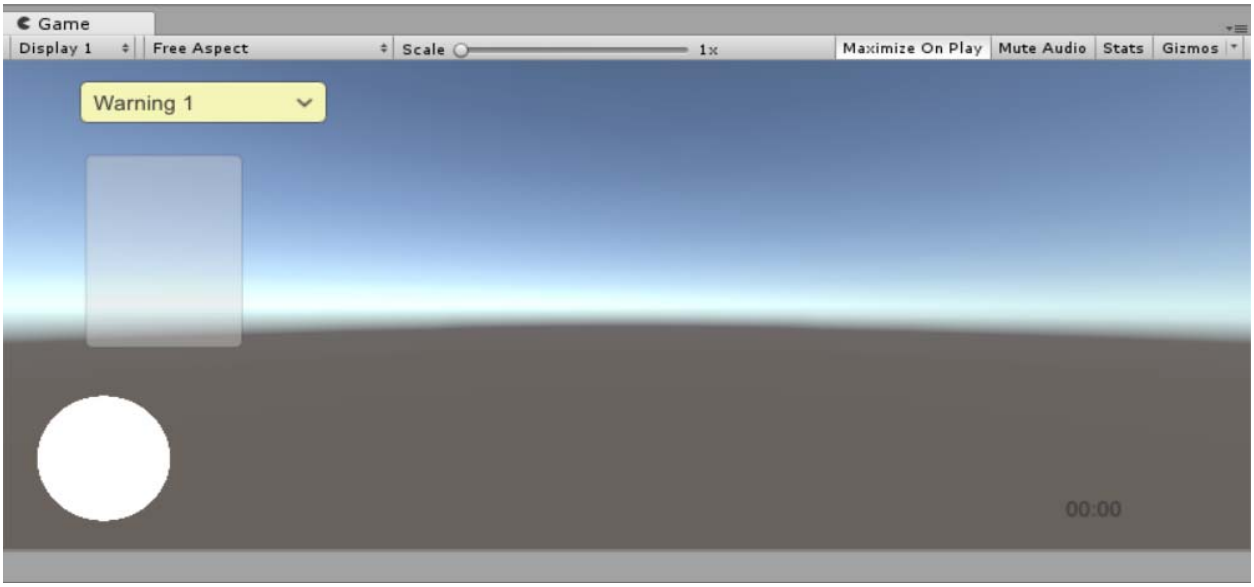

WARNINGS
Battery
80%
H2O
28%
2:07
Pressure
6.16
psi
PO2
11:46
Heart
BT
98°F
SO2
100%
12:59

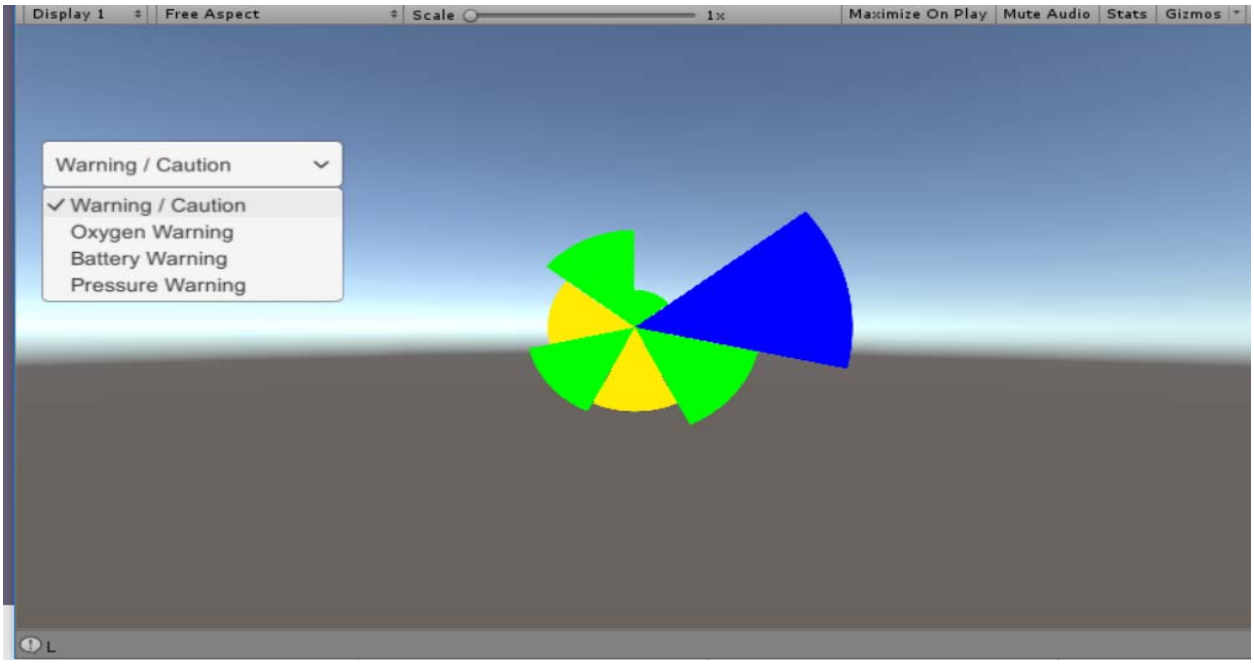

WARNINGS
Water level
28%
Water
Water left - 28%
Time left - 2:07
Remaining H2O
- 16/7 lbs
Feedwater P - 6.08 psi

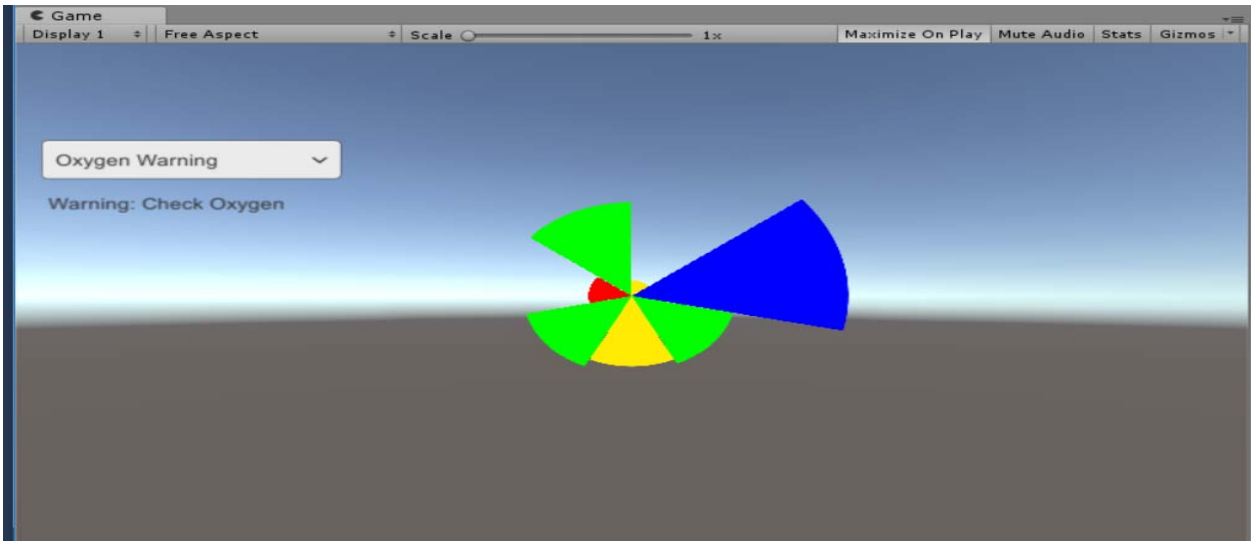

WARNING
Pressure
Suit P - 6.16 psi
PO2 P - 124.5 psi
SO2 P - 6200 psi
Feedwater P - 6.08 psi
H2O
SO2
100 %
BT
98F


EMERGENCY
LOW Oxygen
level (5 %)
Time Remaining : 2:00
Return Back to
the ship
H2O
BT
98F

3. User Interface Design of Health UI (Screenshots)






Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Oxygen Warning
Warning: Check Oxygen

Health UI

QUIS
Questionnaire

NASA S.U.I.T.S.

1. **Email address** *

2. **What is your gender?**
Mark only one oval.

- Male
- Female

3. **What is your age group?**
Mark only one oval.

- 18-23 years old
- 24-29 years old
- 30-34 years old
- 35-39 years old
- 40-44 years old
- 45-49 years old
- 50-54 years old
- 55-59 years old
- 60-64 years old
- 65 years and over

4. **What is the highest level of education you have completed?**
Mark only one oval.

- Some High School
- High School Graduate
- Some College
- Trade/Technical/Vocational Training
- College Graduate
- Some Post Graduate Work
- Post Graduate Degree

5.
What is your marital status?
Mark only one oval.

- Single/Never been married
- Married
- Separated
- Divorced
- Widowed

6.
Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?
Mark only one oval.

- Yes
- No

7.
Which of the following describes your race?
Mark only one oval.

- White
- Black or African-American
- Asian or Asian-American
- Native American/American Indian/Alaska Native
- Native Hawaiian/Other Pacific Islanders
- Some other race

8.
Are you now employed full-time, part-time, not employed, or retired?
Mark only one oval.

- Full-time
- Part-time
- Not employed
- Retired

9. **The information provided on the HoloLens is sufficient for completing the assigned tasks.**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	◯	◯	◯	◯	◯	◯	◯	◯	◯	Strongly Agree

10. **I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	◯	◯	◯	◯	◯	◯	◯	◯	◯	Strongly Satisfied

11. **The font of the Warning Message is easy to read and understand.**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Difficult	◯	◯	◯	◯	◯	◯	◯	◯	◯	Easy

12. **The color code for the pie (sprocket) is easy to understand.**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Difficult	◯	◯	◯	◯	◯	◯	◯	◯	◯	Easy

13. **The system response time for performing the tasks is appropriate.**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	◯	◯	◯	◯	◯	◯	◯	◯	◯	Strongly Appropriate

14.
The system provided appropriate warning messages related to the health of the Astronaut Suit.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	◯	◯	◯	◯	◯	◯	◯	◯	◯	Strongly Agree

15.
The voice commands for the Health UI clearly aligned with what tasks will get executed.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	◯	◯	◯	◯	◯	◯	◯	◯	◯	Strongly Agree

16.
The terminology used for Health UI is easy to understand.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Difficult	◯	◯	◯	◯	◯	◯	◯	◯	◯	Easy

17.
The Health UI voice commands were easily recognized by the system.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	◯	◯	◯	◯	◯	◯	◯	◯	◯	Strongly Agree

18.
Overall, the Health User Interface is easy to understand.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	◯	◯	◯	◯	◯	◯	◯	◯	◯	Strongly Agree

19.
Voice is audible.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Confusing	◯	◯	◯	◯	◯	◯	◯	◯	◯	Clear

20.
Instructions for voice commands.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Confusing	◯	◯	◯	◯	◯	◯	◯	◯	◯	Clear

21.
Designed for all level of users.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Never	◯	◯	◯	◯	◯	◯	◯	◯	◯	Always

22.
Color, Font of the text on theTask UI screen (Visual appearance).
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Poor	◯	◯	◯	◯	◯	◯	◯	◯	◯	Good

23.
Using the hand gestures.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Confusing	◯	◯	◯	◯	◯	◯	◯	◯	◯	Clear

24.
Position of messages on the screen.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Inconsistent	◯	◯	◯	◯	◯	◯	◯	◯	◯	Consistent

25.
Task U.I has clear sequence of screens.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Confusing	◯	◯	◯	◯	◯	◯	◯	◯	◯	Clear

26.
User can view next step/previous stepwhenever he/she liked to.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Inconsistent	◯	◯	◯	◯	◯	◯	◯	◯	◯	Consistent

27.
User can view the Cautions/warningsrelated to current step.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Never	◯	◯	◯	◯	◯	◯	◯	◯	◯	Always

28.
User can view the images related to current step.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Never	◯	◯	◯	◯	◯	◯	◯	◯	◯	Always

29.
Terminology used in the Task UI is related to the tasks.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Never	◯	◯	◯	◯	◯	◯	◯	◯	◯	Always

30.
Highlighting the current step helps user to clearly understand the task.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Not at all	◯	◯	◯	◯	◯	◯	◯	◯	◯	Very much

31.
Learning to operate the system.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Difficult	◯	◯	◯	◯	◯	◯	◯	◯	◯	Easy

32.
Navigation from one step to other is taking minimum time.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Inconsistent	◯	◯	◯	◯	◯	◯	◯	◯	◯	Consistent

33.
User can perform the task correctly.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Never	◯	◯	◯	◯	◯	◯	◯	◯	◯	Always

34.
Correcting mistakes.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Difficult	◯	◯	◯	◯	◯	◯	◯	◯	◯	Easy

35.
System speed.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Too Slow	◯	◯	◯	◯	◯	◯	◯	◯	◯	Fast Enough

36.
User can find all the information related to the step.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Never	◯	◯	◯	◯	◯	◯	◯	◯	◯	Always

37.
User is notified when the task is completed.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Unhelpful	◯	◯	◯	◯	◯	◯	◯	◯	◯	Helpful

38.
Overall task User Interface.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Frustrating	◯	◯	◯	◯	◯	◯	◯	◯	◯	Satisfying

39.
Holograms make the task process easier.
Mark only one oval.

	1	2	3	4	5	6	7	8	9	
Never	◯	◯	◯	◯	◯	◯	◯	◯	◯	Always

40.
Suggestions

_____
_____
_____
_____
_____

Powered by
Google Forms

Responses

Response1

NASA S.U.I.T.S.

Email address *

What is your gender?

- Male
- Female

What is your age group?

- 18-23 years old
- 24-29 years old
- 30-34 years old
- 35-39 years old
- 40-44 years old
- 45-49 years old
- 50-54 years old
- 55-59 years old
- 60-64 years old
- 65 years and over

What is the highest level of education you have completed?

- ◯ Some High School
- ◯ High School Graduate
- ◯ Some College
- ◯ Trade/Technical/Vocational Training
- ◯ College Graduate
- ◯ Some Post Graduate Work
- ◯ Post Graduate Degree

What is your marital status?

- ◯ Single/Never been married
- ◯ Married
- ◯ Separated
- ◯ Divorced
- ◯ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ◯ Yes
- ◯ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ○ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ○ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	◉	○	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	○	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	◉	○	○	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	◉	○	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	◉	○	○	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	◉	○	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

Response2

NASA S.U.I.T.S.

Email address *

What is your gender?

- Male
- Female

What is your age group?

- 18-23 years old
- 24-29 years old
- 30-34 years old
- 35-39 years old
- 40-44 years old
- 45-49 years old
- 50-54 years old
- 55-59 years old
- 60-64 years old
- 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ○ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ○ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ○ No

Which of the following describes your race?

- White
- Black or African-American
- Asian or Asian-American
- Native American/American Indian/Alaska Native
- Native Hawaiian/Other Pacific Islanders
- Some other race

Are you now employed full-time, part-time, not employed, or retired?

- Full-time
- Part-time
- Not employed
- Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	◉	○	○	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	◉	○	○	○	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	◉	○	○	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	◉	○	○	○	○	○	○	○	○	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	◉	○	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	◉	○	○	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	◉	○	○	○	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	○	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	◉	○	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	◉	○	○	○	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	◉	○	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	◉	○	○	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Suggestions

Response3

NASA S.U.I.T.S.

Email address *

What is your gender?

- Male
- Female

What is your age group?

- 18-23 years old
- 24-29 years old
- 30-34 years old
- 35-39 years old
- 40-44 years old
- 45-49 years old
- 50-54 years old
- 55-59 years old
- 60-64 years old
- 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ○ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ○ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ○ No

Which of the following describes your race?

- White
- Black or African-American
- Asian or Asian-American
- Native American/American Indian/Alaska Native
- Native Hawaiian/Other Pacific Islanders
- Some other race

Are you now employed full-time, part-time, not employed, or retired?

- Full-time
- Part-time
- Not employed
- Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	○	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

Response4

NASA S.U.I.T.S.

Email address *

sudheera.dama@gmail.com

What is your gender?

- ○ Male
- ◉ Female

What is your age group?

- ◉ 18-23 years old
- ○ 24-29 years old
- ○ 30-34 years old
- ○ 35-39 years old
- ○ 40-44 years old
- ○ 45-49 years old
- ○ 50-54 years old
- ○ 55-59 years old
- ○ 60-64 years old
- ○ 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	◉	○	○	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Suggestions

Response5

NASA S.U.I.T.S.

Email address *

akhilachowdary007@gmail.com

What is your gender?

- ○ Male
- ◉ Female

What is your age group?

- ◉ 18-23 years old
- ○ 24-29 years old
- ○ 30-34 years old
- ○ 35-39 years old
- ○ 40-44 years old
- ○ 45-49 years old
- ○ 50-54 years old
- ○ 55-59 years old
- ○ 60-64 years old
- ○ 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

Response6

NASA S.U.I.T.S.

Email address *

vmounikareddy1996@gmail.com

What is your gender?

- ○ Male
- ◉ Female

What is your age group?

- ◉ 18-23 years old
- ○ 24-29 years old
- ○ 30-34 years old
- ○ 35-39 years old
- ○ 40-44 years old
- ○ 45-49 years old
- ○ 50-54 years old
- ○ 55-59 years old
- ○ 60-64 years old
- ○ 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

Response7

NASA S.U.I.T.S.

Email address *

kavyakiran7142427@gmail.com

What is your gender?

- ○ Male
- ◉ Female

What is your age group?

- ◉ 18-23 years old
- ○ 24-29 years old
- ○ 30-34 years old
- ○ 35-39 years old
- ○ 40-44 years old
- ○ 45-49 years old
- ○ 50-54 years old
- ○ 55-59 years old
- ○ 60-64 years old
- ○ 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

Response8

NASA S.U.I.T.S.

Email address *

MirzaM4179@UHCL.edu

What is your gender?

- ○ Male
- ◉ Female

What is your age group?

- ◉ 18-23 years old
- ○ 24-29 years old
- ○ 30-34 years old
- ○ 35-39 years old
- ○ 40-44 years old
- ○ 45-49 years old
- ○ 50-54 years old
- ○ 55-59 years old
- ○ 60-64 years old
- ○ 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ○ Single/Never been married
- ◉ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	◉	○	○	○	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	◉	○	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	◉	○	○	○	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	◉	○	○	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	◉	○	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	◉	○	○	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	◉	○	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

Response9

NASA S.U.I.T.S.

Email address *

bhavirisettyu4193@uhcl.edu

What is your gender?

- (●) Male
- ( ) Female

What is your age group?

- (●) 18-23 years old
- ( ) 24-29 years old
- ( ) 30-34 years old
- ( ) 35-39 years old
- ( ) 40-44 years old
- ( ) 45-49 years old
- ( ) 50-54 years old
- ( ) 55-59 years old
- ( ) 60-64 years old
- ( ) 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	◉	○	○	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	◉	○	○	○	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	◉	○	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	○	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	◉	○	○	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	◉	○	○	○	○	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	◉	○	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	◉	○	○	○	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	◉	○	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	◉	○	○	○	○	○	○	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	◉	○	○	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	◉	○	○	○	○	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	◉	○	○	○	○	Always

Suggestions

Response10

NASA S.U.I.T.S.

Email address *

pratiknpal@gmail.com

What is your gender?

- (●) Male
- ( ) Female

What is your age group?

- ( ) 18-23 years old
- (●) 24-29 years old
- ( ) 30-34 years old
- ( ) 35-39 years old
- ( ) 40-44 years old
- ( ) 45-49 years old
- ( ) 50-54 years old
- ( ) 55-59 years old
- ( ) 60-64 years old
- ( ) 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	◉	○	○	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	◉	○	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	◉	○	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	◉	○	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	◉	○	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	◉	○	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	◉	○	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	◉	○	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	◉	○	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	◉	○	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	○	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	◉	○	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	◉	○	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	◉	○	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	◉	○	○	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	◉	○	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	◉	○	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Suggestions

Slow rendering

Response11

NASA S.U.I.T.S.

Email address *

vikasraghav1993@gmail.com

What is your gender?

- (●) Male
- ( ) Female

What is your age group?

- ( ) 18-23 years old
- (●) 24-29 years old
- ( ) 30-34 years old
- ( ) 35-39 years old
- ( ) 40-44 years old
- ( ) 45-49 years old
- ( ) 50-54 years old
- ( ) 55-59 years old
- ( ) 60-64 years old
- ( ) 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ○ College Graduate
- ○ Some Post Graduate Work
- ◉ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ◉ Full-time
- ○ Part-time
- ○ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	◉	○	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	◉	○	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	◉	○	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	◉	○	○	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	◉	○	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	◉	○	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	◉	○	○	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	◉	○	○	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	◉	○	○	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	◉	○	○	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	◉	○	○	○	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	◉	○	○	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	◉	○	○	○	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	◉	○	○	○	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	◉	○	○	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	◉	○	○	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	◉	○	○	○	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	◉	○	○	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	◉	○	○	○	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	◉	○	○	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	◉	○	○	○	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	◉	○	○	○	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	◉	○	○	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	◉	○	○	○	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	◉	○	○	○	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	◉	○	○	○	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	◉	○	○	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	◉	○	○	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	◉	○	Always

Suggestions

Response12

NASA S.U.I.T.S.

Email address *

alharbiw1384@uhcl.edu

What is your gender?

- ○ Male
- ◉ Female

What is your age group?

- ○ 18-23 years old
- ◉ 24-29 years old
- ○ 30-34 years old
- ○ 35-39 years old
- ○ 40-44 years old
- ○ 45-49 years old
- ○ 50-54 years old
- ○ 55-59 years old
- ○ 60-64 years old
- ○ 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ○ Single/Never been married
- ◉ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- (●) White
- ( ) Black or African-American
- ( ) Asian or Asian-American
- ( ) Native American/American Indian/Alaska Native
- ( ) Native Hawaiian/Other Pacific Islanders
- ( ) Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ( ) Full-time
- ( ) Part-time
- (●) Not employed
- ( ) Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	( )	( )	( )	( )	( )	( )	( )	( )	(●)	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	○	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

a great work.

Response13

NASA S.U.I.T.S.

Email address *

alurumamatha99@gmail.com

What is your gender?

- ○ Male
- ◉ Female

What is your age group?

- ◉ 18-23 years old
- ○ 24-29 years old
- ○ 30-34 years old
- ○ 35-39 years old
- ○ 40-44 years old
- ○ 45-49 years old
- ○ 50-54 years old
- ○ 55-59 years old
- ○ 60-64 years old
- ○ 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ○ Some College
- ○ Trade/Technical/Vocational Training
- ◉ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ○ Black or African-American
- ◉ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	◉	○	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	○	○	◉	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	◉	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Suggestions

Response14

NASA S.U.I.T.S.

Email address *

oezenwabachili1994@gmail.com

What is your gender?

- (●) Male
- ( ) Female

What is your age group?

- (●) 18-23 years old
- ( ) 24-29 years old
- ( ) 30-34 years old
- ( ) 35-39 years old
- ( ) 40-44 years old
- ( ) 45-49 years old
- ( ) 50-54 years old
- ( ) 55-59 years old
- ( ) 60-64 years old
- ( ) 65 years and over

What is the highest level of education you have completed?

- ○ Some High School
- ○ High School Graduate
- ◉ Some College
- ○ Trade/Technical/Vocational Training
- ○ College Graduate
- ○ Some Post Graduate Work
- ○ Post Graduate Degree

What is your marital status?

- ◉ Single/Never been married
- ○ Married
- ○ Separated
- ○ Divorced
- ○ Widowed

Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

- ○ Yes
- ◉ No

Which of the following describes your race?

- ○ White
- ◉ Black or African-American
- ○ Asian or Asian-American
- ○ Native American/American Indian/Alaska Native
- ○ Native Hawaiian/Other Pacific Islanders
- ○ Some other race

Are you now employed full-time, part-time, not employed, or retired?

- ○ Full-time
- ○ Part-time
- ◉ Not employed
- ○ Retired

The information provided on the HoloLens is sufficient for completing the assigned tasks.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	◉	○	○	○	Strongly Agree

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

	1	2	3	4	5	6	7	8	9	
Strongly Unsatisfied	○	○	○	○	○	○	○	○	◉	Strongly Satisfied

The font of the Warning Message is easy to read and understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The color code for the pie (sprocket) is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The system response time for performing the tasks is appropriate.

	1	2	3	4	5	6	7	8	9	
Strongly Inappropriate	○	○	○	○	○	○	○	○	◉	Strongly Appropriate

The system provided appropriate warning messages related to the health of the Astronaut Suit.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The voice commands for the Health UI clearly aligned with what tasks will get executed.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

The terminology used for Health UI is easy to understand.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	○	○	◉	Easy

The Health UI voice commands were easily recognized by the system.

	1	2	3	4	5	6	7	8	9	
Strongly Disagree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Overall, the Health User Interface is easy to understand.

	1	2	3	4	5	6	7	8	9	
Strongly Diasgree	○	○	○	○	○	○	○	○	◉	Strongly Agree

Voice is audible.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Instructions for voice commands.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

Designed for all level of users.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	◉	○	○	○	Always

Color, Font of the text on theTask UI screen (Visual appearance).

	1	2	3	4	5	6	7	8	9	
Poor	○	○	○	○	○	○	◉	○	○	Good

Using the hand gestures.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	◉	○	○	○	○	Clear

Position of messages on the screen.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	◉	○	Consistent

Task U.I has clear sequence of screens.

	1	2	3	4	5	6	7	8	9	
Confusing	○	○	○	○	○	○	○	○	◉	Clear

User can view next step/previous stepwhenever he/she liked to.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can view the Cautions/warningsrelated to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

User can view the images related to current step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Terminology used in the Task UI is related to the tasks.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Highlighting the current step helps user to clearly understand the task.

	1	2	3	4	5	6	7	8	9	
Not at all	○	○	○	○	○	○	○	○	◉	Very much

Learning to operate the system.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	○	○	◉	○	○	Easy

Navigation from one step to other is taking minimum time.

	1	2	3	4	5	6	7	8	9	
Inconsistent	○	○	○	○	○	○	○	○	◉	Consistent

User can perform the task correctly.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	○	○	○	◉	Always

Correcting mistakes.

	1	2	3	4	5	6	7	8	9	
Difficult	○	○	○	○	◉	○	○	○	○	Easy

System speed.

	1	2	3	4	5	6	7	8	9	
Too Slow	○	○	○	○	○	○	○	○	◉	Fast Enough

User can find all the information related to the step.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	○	◉	○	○	○	Always

User is notified when the task is completed.

	1	2	3	4	5	6	7	8	9	
Unhelpful	○	○	○	○	○	○	○	○	○	Helpful

0verall task User Interface.

	1	2	3	4	5	6	7	8	9	
Frustrating	○	○	○	○	○	○	○	○	◉	Satisfying

Holograms make the task process easier.

	1	2	3	4	5	6	7	8	9	
Never	○	○	○	○	◉	○	○	○	○	Always

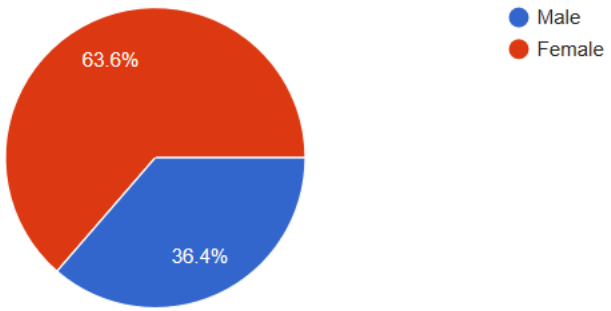
Suggestions

implementing more Holograms identifying certain objects could help

Data Analysis

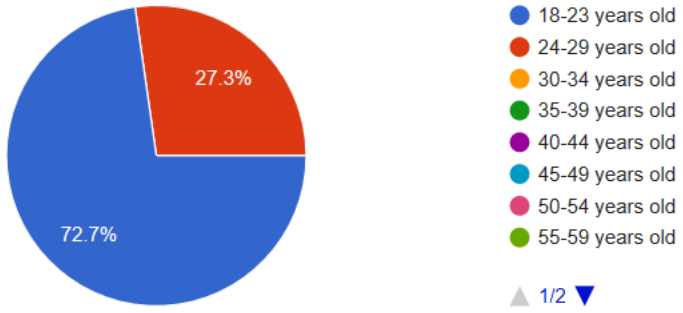
What is your gender?

11 responses



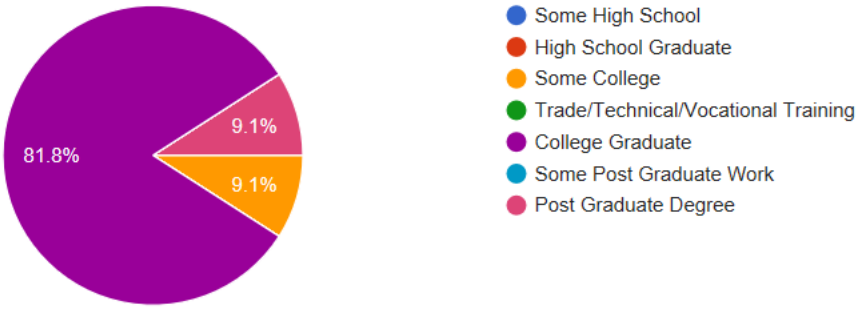
What is your age group?

11 responses



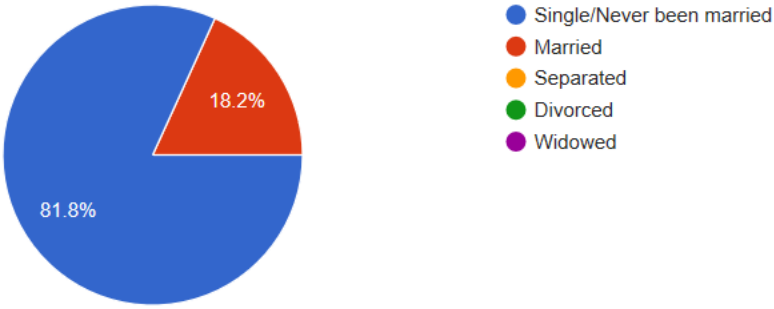
What is the highest level of education you have completed?

11 responses



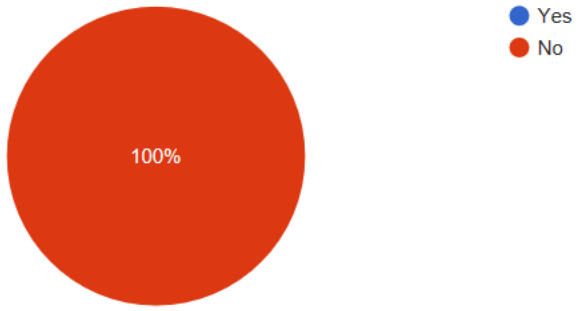
What is your marital status?

11 responses



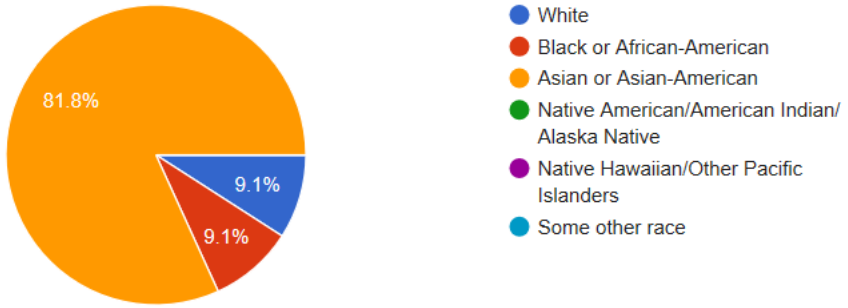
Are you of Hispanic, Latino, or Spanish origin, such as Mexican, Puerto Rican or Cuban?

11 responses



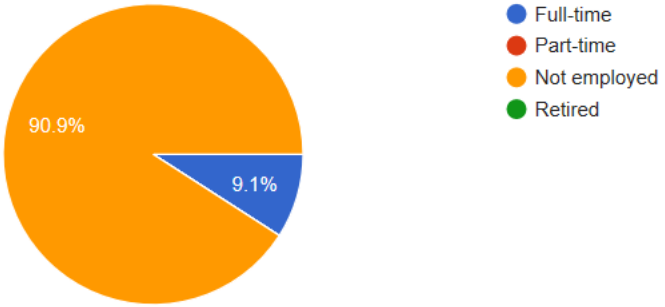
Which of the following describes your race?

11 responses



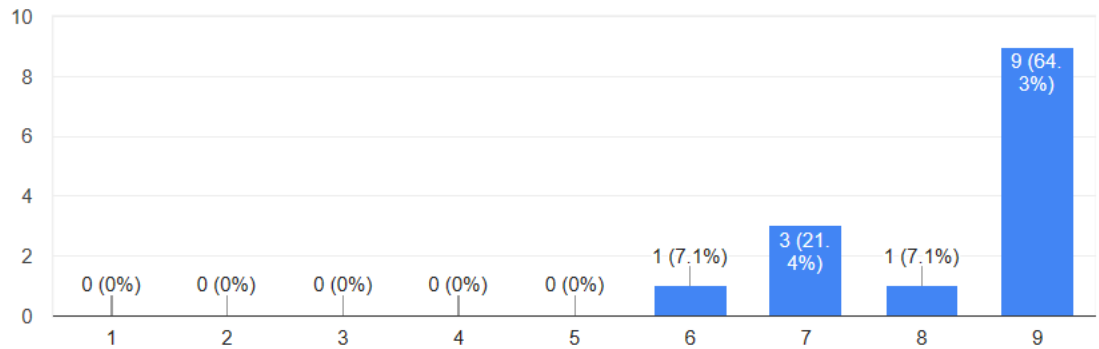
Are you now employed full-time, part-time, not employed, or retired?

11 responses



The information provided on the HoloLens is sufficient for completing the assigned tasks.

14 responses



Standard Deviation Calculator

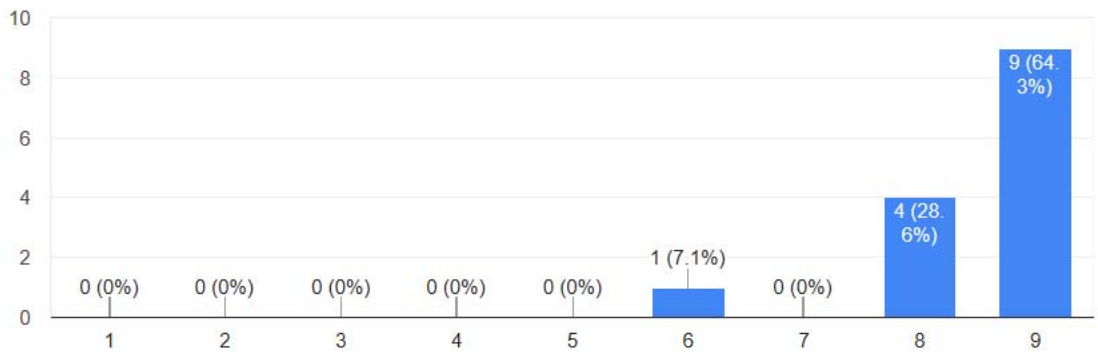
Result	
Sample Standard Deviation, s	1.0690449676497
Variance (Sample Standard), s^2	1.1428571428571
Population Standard Deviation, σ	1.0301575072754
Variance (Population Standard), σ^2	1.0612244897959
Total Numbers, N	14
Sum:	116
Mean (Average):	8.2857142857143
Standard Error of the Mean ($SE_{\bar{x}}$):	0.28571428571429

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8 - 8.5714285714286
90%, $1.645SE_{\bar{x}}$	7.8157142857143 - 8.7557142857143
95%, $1.960SE_{\bar{x}}$	7.7257142857143 - 8.8457142857143
99%, $2.576SE_{\bar{x}}$	7.5497142857143 - 9.0217142857143
99.9%, $3.291SE_{\bar{x}}$	7.3454285714286 - 9.226
99.99%, $3.891SE_{\bar{x}}$	7.174 - 9.3974285714286
99.999%, $4.417SE_{\bar{x}}$	7.0237142857143 - 9.5477142857143
99.9999%, $4.892SE_{\bar{x}}$	6.888 - 9.6834285714286

I am satisfied by the organization of details provided on the HoloLens screen (i.e. Details about Health of Astronaut Suit, Warning messages).

14 responses



Standard Deviation Calculator

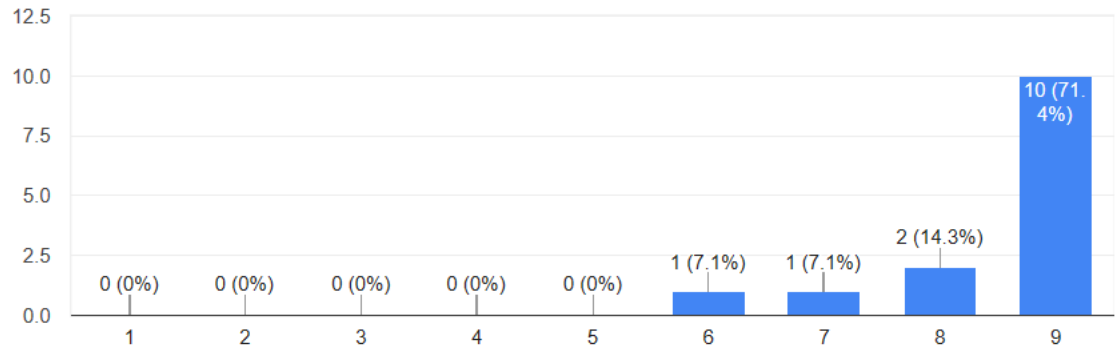
Result	
Sample Standard Deviation, s	0.85485041426511
Variance (Sample Standard), s^2	0.73076923076923
Population Standard Deviation, σ	0.82375447104791
Variance (Population Standard), σ^2	0.67857142857143
Total Numbers, N	14
Sum:	119
Mean (Average):	8.5
Standard Error of the Mean ($SE_{\bar{x}}$):	0.22846838336584

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.2715316166342 - 8.7284683833658
90%, $1.645SE_{\bar{x}}$	8.1241695093632 - 8.8758304906368
95%, $1.960SE_{\bar{x}}$	8.0522019686029 - 8.9477980313971
99%, $2.576SE_{\bar{x}}$	7.9114654444496 - 9.0885345555504
99.9%, $3.291SE_{\bar{x}}$	7.748110550343 - 9.251889449657
99.99%, $3.891SE_{\bar{x}}$	7.6110295203235 - 9.3889704796765
99.999%, $4.417SE_{\bar{x}}$	7.4908551506731 - 9.5091448493269
99.9999%, $4.892SE_{\bar{x}}$	7.3823326685743 - 9.6176673314257

The font of the Warning Message is easy to read and understand.

14 responses



Standard Deviation Calculator

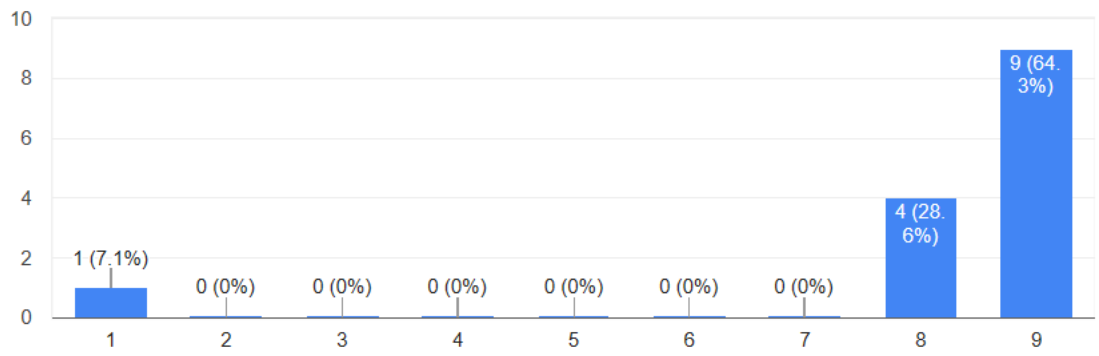
Result	
Sample Standard Deviation, s	0.94053994312596
Variance (Sample Standard), s^2	0.88461538461538
Population Standard Deviation, σ	0.90632696717497
Variance (Population Standard), σ^2	0.82142857142857
Total Numbers, N	14
Sum:	119
Mean (Average):	8.5
Standard Error of the Mean ($SE_{\bar{x}}$):	0.25136987326809

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.2486301267319 - 8.7513698732681
90%, $1.645SE_{\bar{x}}$	8.086496558474 - 8.913503441526
95%, $1.960SE_{\bar{x}}$	8.0073150483946 - 8.9926849516054
99%, $2.576SE_{\bar{x}}$	7.8524712064614 - 9.1475287935386
99.9%, $3.291SE_{\bar{x}}$	7.6727417470747 - 9.3272582529253
99.99%, $3.891SE_{\bar{x}}$	7.5219198231139 - 9.4780801768861
99.999%, $4.417SE_{\bar{x}}$	7.3896992697749 - 9.6103007302251
99.9999%, $4.892SE_{\bar{x}}$	7.2702985799725 - 9.7297014200275

The color code for the pie (sprocket) is easy to understand.

14 responses



Standard Deviation Calculator

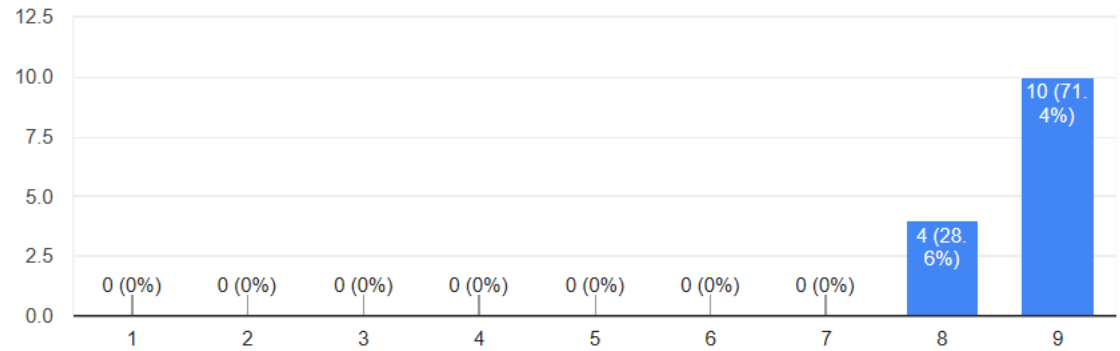
Result	
Sample Standard Deviation, s	2.1070264449125
Variance (Sample Standard), s^2	4.4395604395604
Population Standard Deviation, σ	2.0303814862217
Variance (Population Standard), σ^2	4.1224489795918
Total Numbers, N	14
Sum:	114
Mean (Average):	8.1428571428571
Standard Error of the Mean ($SE_{\bar{x}}$):	0.56312650440962

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	7.5797306384475 - 8.7059836472668
90%, $1.645SE_{\bar{x}}$	7.2165140431033 - 9.069200242611
95%, $1.960SE_{\bar{x}}$	7.0391291942143 - 9.2465850915
99%, $2.576SE_{\bar{x}}$	6.6922432674979 - 9.5934710182163
99.9%, $3.291SE_{\bar{x}}$	6.2896078168451 - 9.9961064688692
99.99%, $3.891SE_{\bar{x}}$	5.9517319141993 - 10.333982371515
99.999%, $4.417SE_{\bar{x}}$	5.6555273728798 - 10.630186912834
99.9999%, $4.892SE_{\bar{x}}$	5.3880422832853 - 10.897672002429

The system response time for performing the tasks is appropriate.

14 responses



Standard Deviation Calculator

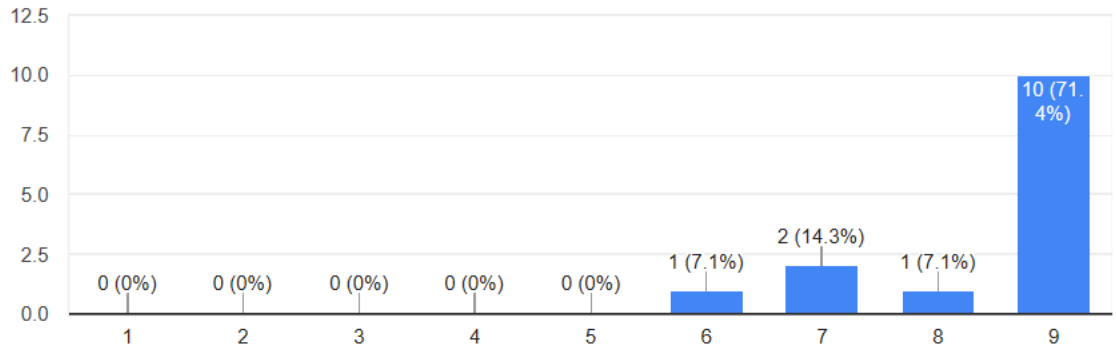
Result	
Sample Standard Deviation, s	0.4688072309385
Variance (Sample Standard), s^2	0.21978021978022
Population Standard Deviation, σ	0.45175395145263
Variance (Population Standard), σ^2	0.20408163265306
Total Numbers, N	14
Sum:	122
Mean (Average):	8.7142857142857
Standard Error of the Mean ($SE_{\bar{x}}$):	0.12529400275815

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.5889917115276 - 8.8395797170439
90%, $1.645SE_{\bar{x}}$	8.5081770797486 - 8.9203943488229
95%, $1.960SE_{\bar{x}}$	8.4687094688797 - 8.9598619596917
99%, $2.576SE_{\bar{x}}$	8.3915283631807 - 9.0370430653907
99.9%, $3.291SE_{\bar{x}}$	8.3019431512087 - 9.1266282773628
99.99%, $3.891SE_{\bar{x}}$	8.2267667495538 - 9.2018046790177
99.999%, $4.417SE_{\bar{x}}$	8.160862104103 - 9.2677093244684
99.9999%, $4.892SE_{\bar{x}}$	8.1013474527929 - 9.3272239757786

The system provided appropriate warning messages related to the health of the Astronaut Suit.

14 responses



Standard Deviation Calculator

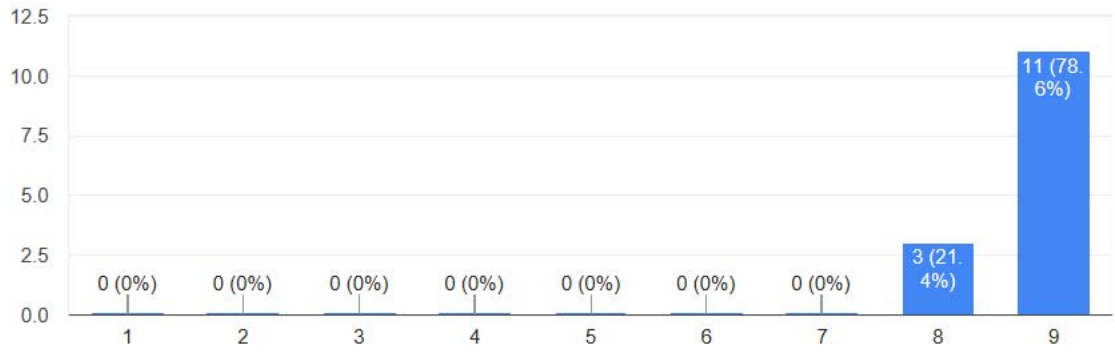
Result	
Sample Standard Deviation, s	1.0163498575624
Variance (Sample Standard), s^2	1.032967032967
Population Standard Deviation, σ	0.97937922862872
Variance (Population Standard), σ^2	0.95918367346939
Total Numbers, N	14
Sum:	118
Mean (Average):	8.4285714285714
Standard Error of the Mean ($SE_{\bar{x}}$):	0.27163092514963

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.1569405034218 - 8.7002023537211
90%, $1.645SE_{\bar{x}}$	7.9817385567003 - 8.8754043004426
95%, $1.960SE_{\bar{x}}$	7.8961748152781 - 8.9609680418647
99%, $2.576SE_{\bar{x}}$	7.728850165386 - 9.1282926917569
99.9%, $3.291SE_{\bar{x}}$	7.534634053904 - 9.3225088032389
99.99%, $3.891SE_{\bar{x}}$	7.3716554988142 - 9.4854873583286
99.999%, $4.417SE_{\bar{x}}$	7.2287776321855 - 9.6283652249574
99.9999%, $4.892SE_{\bar{x}}$	7.0997529427394 - 9.7573899144034

The voice commands for the Health UI clearly aligned with what tasks will get executed.

14 responses



Standard Deviation Calculator

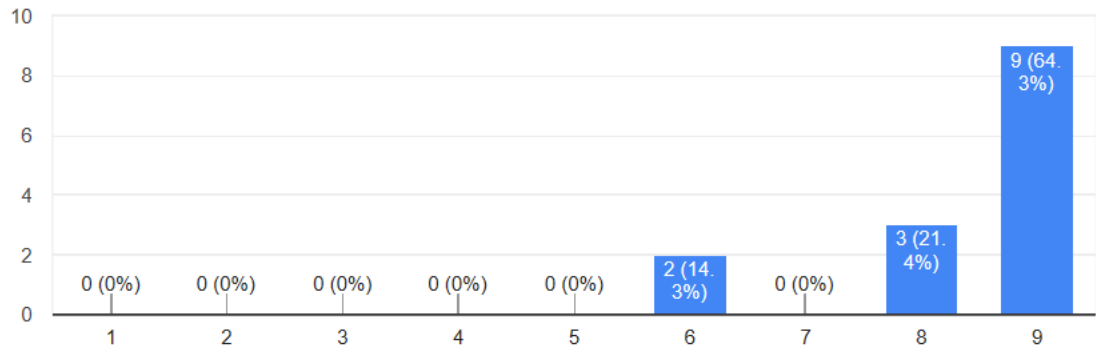
Result	
Sample Standard Deviation, s	0.42581531362632
Variance (Sample Standard), s^2	0.18131868131868
Population Standard Deviation, σ	0.41032590332414
Variance (Population Standard), σ^2	0.16836734693878
Total Numbers, N	14
Sum:	123
Mean (Average):	8.7857142857143
Standard Error of the Mean ($SE_{\bar{x}}$):	0.1138039295451

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.6719103561692 - 8.8995182152594
90%, $1.645SE_{\bar{x}}$	8.5985068216126 - 8.972921749816
95%, $1.960SE_{\bar{x}}$	8.5626585838059 - 9.0087699876227
99%, $2.576SE_{\bar{x}}$	8.4925553632061 - 9.0788732082225
99.9%, $3.291SE_{\bar{x}}$	8.4111855535814 - 9.1602430178472
99.99%, $3.891SE_{\bar{x}}$	8.3429031958543 - 9.2285253755743
99.999%, $4.417SE_{\bar{x}}$	8.2830423289136 - 9.288386242515
99.9999%, $4.892SE_{\bar{x}}$	8.2289854623797 - 9.3424431090489

The terminology used for Health UI is easy to understand.

14 responses



Standard Deviation Calculator

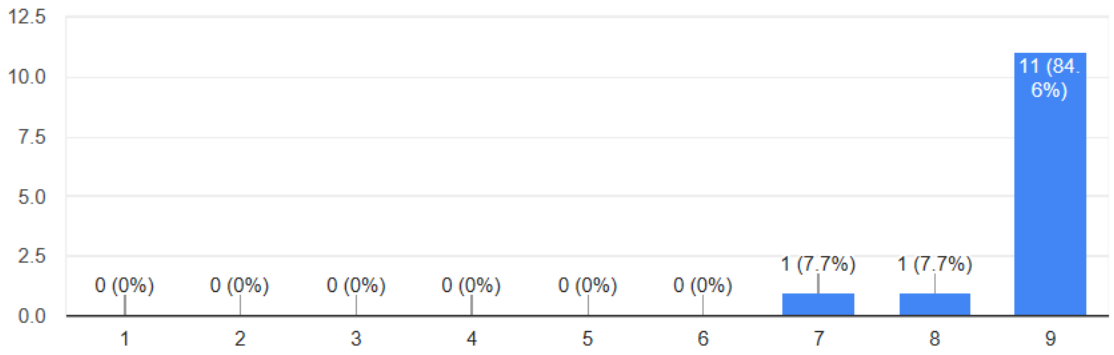
Result	
Sample Standard Deviation, s	1.0818177620698
Variance (Sample Standard), s^2	1.1703296703297
Population Standard Deviation, σ	1.0424656799519
Variance (Population Standard), σ^2	1.0867346938776
Total Numbers, N	14
Sum:	117
Mean (Average):	8.3571428571429
Standard Error of the Mean ($SE_{\bar{x}}$):	0.28912795861369

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.0680148985292 - 8.6462708157565
90%, $1.645SE_{\bar{x}}$	7.8815273652233 - 8.8327583490624
95%, $1.960SE_{\bar{x}}$	7.79045205826 - 8.9238336560257
99%, $2.576SE_{\bar{x}}$	7.612349235754 - 9.1019364785317
99.9%, $3.291SE_{\bar{x}}$	7.4056227453452 - 9.3086629689405
99.99%, $3.891SE_{\bar{x}}$	7.232145970177 - 9.4821397441087
99.999%, $4.417SE_{\bar{x}}$	7.0800646639462 - 9.6342210503395
99.9999%, $4.892SE_{\bar{x}}$	6.9427288836047 - 9.771556830681

The Health UI voice commands were easily recognized by the system.

13 responses



Standard Deviation Calculator

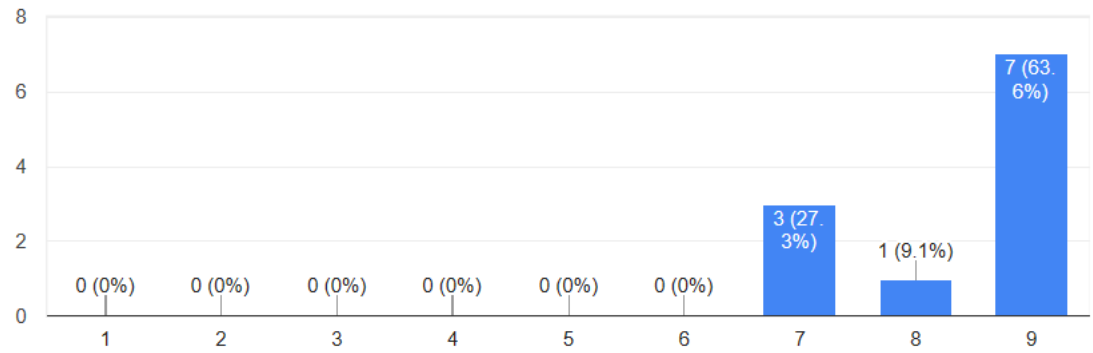
Result	
Sample Standard Deviation, s	0.59914468951528
Variance (Sample Standard), s^2	0.35897435897436
Population Standard Deviation, σ	0.57563959796522
Variance (Population Standard), σ^2	0.33136094674556
Total Numbers, N	13
Sum:	114
Mean (Average):	8.7692307692308
Standard Error of the Mean ($SE_{\bar{x}}$):	0.16617283842071

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.6030579308101 - 8.9354036076515
90%, $1.645SE_{\bar{x}}$	8.4958764500287 - 9.0425850884328
95%, $1.960SE_{\bar{x}}$	8.4435320059262 - 9.0949295325354
99%, $2.576SE_{\bar{x}}$	8.341169537459 - 9.1972920010025
99.9%, $3.291SE_{\bar{x}}$	8.2223559579882 - 9.3161055804733
99.99%, $3.891SE_{\bar{x}}$	8.1226522549358 - 9.4158092835258
99.999%, $4.417SE_{\bar{x}}$	8.0352453419265 - 9.5032161965351
99.9999%, $4.892SE_{\bar{x}}$	7.9563132436766 - 9.5821482947849

Overall, the Health User Interface is easy to understand.

11 responses



Standard Deviation Calculator

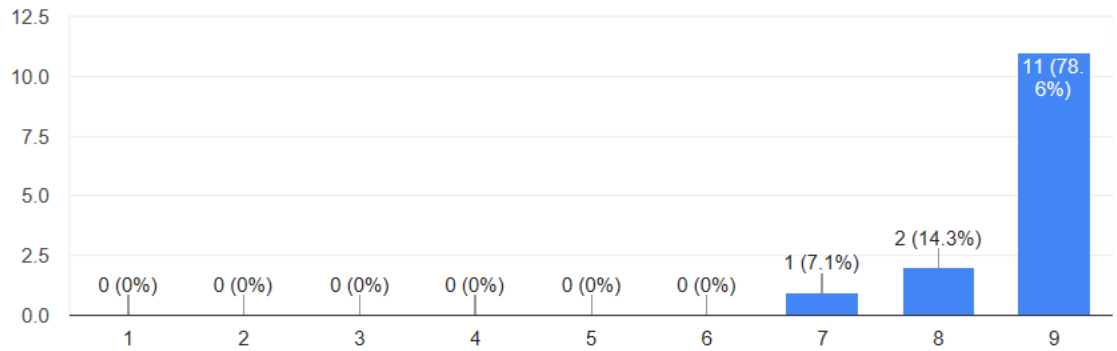
Result	
Sample Standard Deviation, s	0.92441627773718
Variance (Sample Standard), s^2	0.85454545454545
Population Standard Deviation, σ	0.88139633771206
Variance (Population Standard), σ^2	0.77685950413223
Total Numbers, N	11
Sum:	92
Mean (Average):	8.3636363636364
Standard Error of the Mean ($SE_{\bar{x}}$):	0.27872199485011

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.0849143687863 - 8.6423583584865
90%, $1.645SE_{\bar{x}}$	7.9051386821079 - 8.8221340451648
95%, $1.960SE_{\bar{x}}$	7.8173412537302 - 8.9099314735426
99%, $2.576SE_{\bar{x}}$	7.6456485049025 - 9.0816242223702
99.9%, $3.291SE_{\bar{x}}$	7.4463622785847 - 9.2809104486881
99.99%, $3.891SE_{\bar{x}}$	7.2791290816746 - 9.4481436455981
99.999%, $4.417SE_{\bar{x}}$	7.1325213123834 - 9.5947514148893
99.9999%, $4.892SE_{\bar{x}}$	7.0001283648296 - 9.7271443624431

Voice is audible.

14 responses



Standard Deviation Calculator

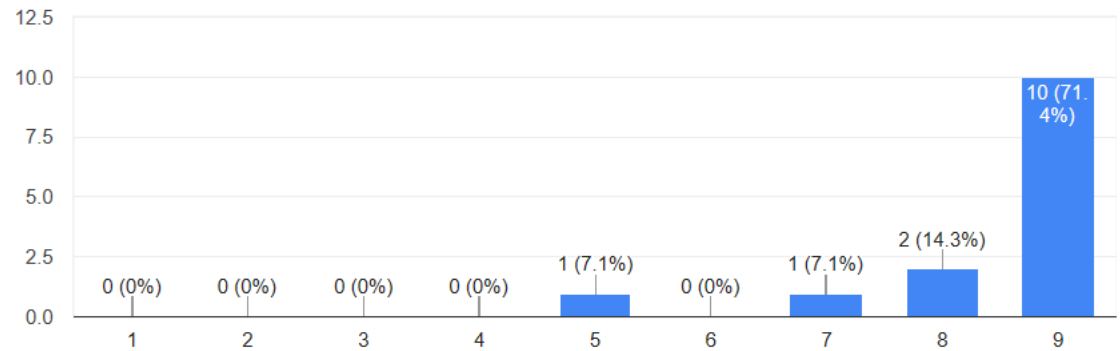
Result	
Sample Standard Deviation, s	0.61124984550213
Variance (Sample Standard), s^2	0.37362637362637
Population Standard Deviation, σ	0.58901508937395
Variance (Population Standard), σ^2	0.3469387755102
Total Numbers, N	14
Sum:	122
Mean (Average):	8.7142857142857
Standard Error of the Mean ($SE_{\bar{x}}$):	0.16336339282768

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.550922321458 - 8.8776491071134
90%, $1.645SE_{\bar{x}}$	8.4455529330842 - 8.9830184954872
95%, $1.960SE_{\bar{x}}$	8.3940934643435 - 9.034477964228
99%, $2.576SE_{\bar{x}}$	8.2934616143616 - 9.1351098142098
99.9%, $3.291SE_{\bar{x}}$	8.1766567884898 - 9.2519146400816
99.99%, $3.891SE_{\bar{x}}$	8.0786387527932 - 9.3499326757782
99.999%, $4.417SE_{\bar{x}}$	7.9927096081659 - 9.4358618204056
99.9999%, $4.892SE_{\bar{x}}$	7.9151119965727 - 9.5134594319987

Instructions for voice commands.

14 responses



Standard Deviation Calculator

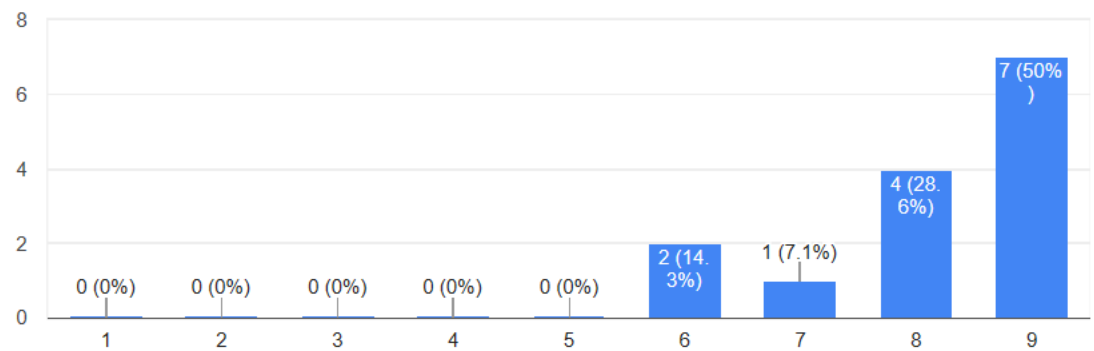
Result	
Sample Standard Deviation, s	1.1578684470437
Variance (Sample Standard), s^2	1.3406593406593
Population Standard Deviation, σ	1.115749953701
Variance (Population Standard), σ^2	1.2448979591837
Total Numbers, N	14
Sum:	118
Mean (Average):	8.4285714285714
Standard Error of the Mean ($SE_{\bar{x}}$):	0.3094533591281

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.1191180694433 - 8.7380247876995
90%, $1.645SE_{\bar{x}}$	7.9195206528057 - 8.9376222043372
95%, $1.960SE_{\bar{x}}$	7.8220428446803 - 9.0351000124625
99%, $2.576SE_{\bar{x}}$	7.6314195754574 - 9.2257232816854
99.9%, $3.291SE_{\bar{x}}$	7.4101604236808 - 9.446982433462
99.99%, $3.891SE_{\bar{x}}$	7.224488408204 - 9.6326544489389
99.999%, $4.417SE_{\bar{x}}$	7.0617159413026 - 9.7954269158403
99.9999%, $4.892SE_{\bar{x}}$	6.9147255957167 - 9.9424172614261

Designed for all level of users.

14 responses



Standard Deviation Calculator

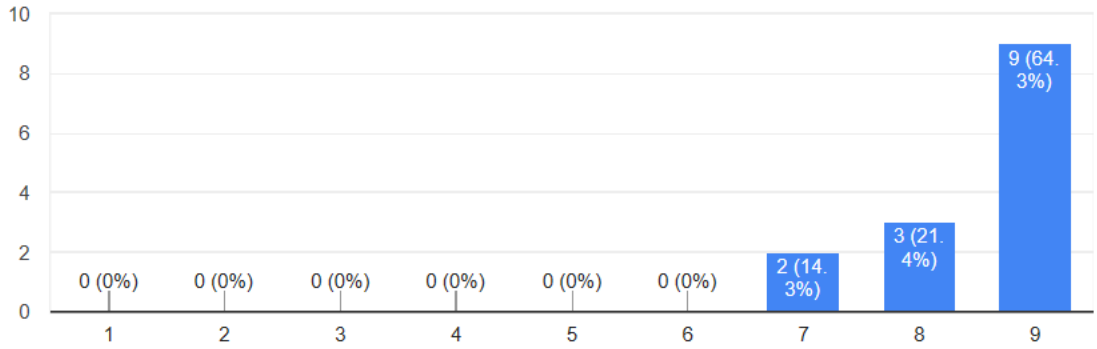
Result	
Sample Standard Deviation, s	1.0994504121566
Variance (Sample Standard), s^2	1.2087912087912
Population Standard Deviation, σ	1.059456926728
Variance (Population Standard), σ^2	1.1224489795918
Total Numbers, N	14
Sum:	114
Mean (Average):	8.1428571428571
Standard Error of the Mean ($SE_{\bar{x}}$):	0.29384048257409

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	7.8490166602831 - 8.4366976254312
90%, $1.645SE_{\bar{x}}$	7.6594895490228 - 8.6262247366915
95%, $1.960SE_{\bar{x}}$	7.5669297970119 - 8.7187844887024
99%, $2.576SE_{\bar{x}}$	7.3859240597463 - 8.899790225968
99.9%, $3.291SE_{\bar{x}}$	7.1758281147058 - 9.1098861710085
99.99%, $3.891SE_{\bar{x}}$	6.9995238251614 - 9.2861904605529
99.999%, $4.417SE_{\bar{x}}$	6.8449637313274 - 9.4407505543869
99.9999%, $4.892SE_{\bar{x}}$	6.7053895021047 - 9.5803247836096

Color, Font of the text on theTask UI screen (Visual appearance).

14 responses



Standard Deviation Calculator

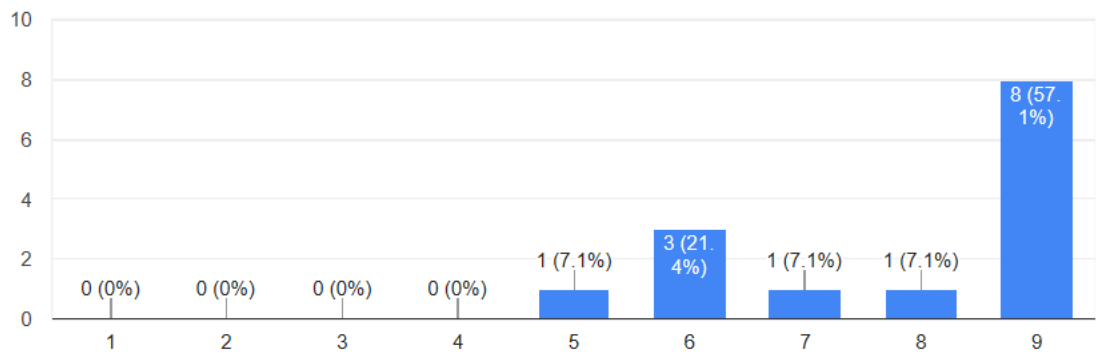
Result	
Sample Standard Deviation, s	0.75955452531275
Variance (Sample Standard), s^2	0.57692307692308
Population Standard Deviation, σ	0.7319250547114
Variance (Population Standard), σ^2	0.53571428571429
Total Numbers, N	14
Sum:	119
Mean (Average):	8.5
Standard Error of the Mean ($SE_{\bar{x}}$):	0.20299948573529

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.2970005142647 - 8.7029994857353
90%, $1.645SE_{\bar{x}}$	8.1660658459655 - 8.8339341540345
95%, $1.960SE_{\bar{x}}$	8.1021210079588 - 8.8978789920412
99%, $2.576SE_{\bar{x}}$	7.9770733247459 - 9.0229266752541
99.9%, $3.291SE_{\bar{x}}$	7.8319286924452 - 9.1680713075548
99.99%, $3.891SE_{\bar{x}}$	7.710129001004 - 9.289870998996
99.999%, $4.417SE_{\bar{x}}$	7.6033512715072 - 9.3966487284928
99.9999%, $4.892SE_{\bar{x}}$	7.506926515783 - 9.493073484217

Using the hand gestures.

14 responses



Standard Deviation Calculator

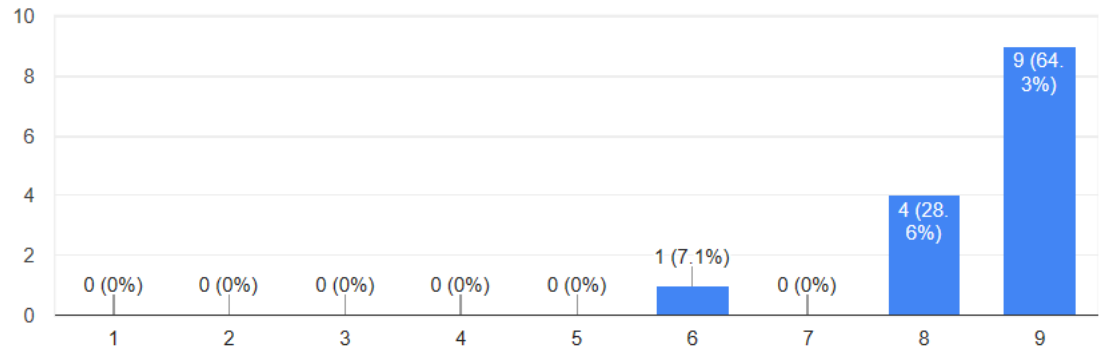
Result	
Sample Standard Deviation, s	1.5118578920369
Variance (Sample Standard), s^2	2.2857142857143
Population Standard Deviation, σ	1.4568627181694
Variance (Population Standard), σ^2	2.1224489795918
Total Numbers, N	14
Sum:	110
Mean (Average):	7.8571428571429
Standard Error of the Mean ($SE_{\bar{x}}$):	0.40406101782088

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	7.453081839322 - 8.2612038749637
90%, $1.645SE_{\bar{x}}$	7.1924624828275 - 8.5218232314582
95%, $1.960SE_{\bar{x}}$	7.0651832622139 - 8.6491024520718
99%, $2.576SE_{\bar{x}}$	6.8162816752363 - 8.8980040390495
99.9%, $3.291SE_{\bar{x}}$	6.5273780474943 - 9.1869076667914
99.99%, $3.891SE_{\bar{x}}$	6.2849414368018 - 9.4293442774839
99.999%, $4.417SE_{\bar{x}}$	6.072405341428 - 9.6418803728577
99.9999%, $4.892SE_{\bar{x}}$	5.8804763579631 - 9.8338093563226

Position of messages on the screen.

14 responses



Standard Deviation Calculator

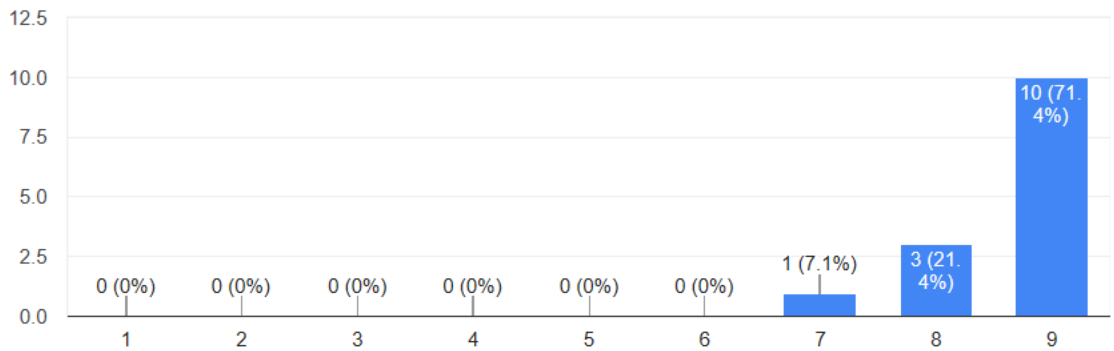
Result	
Sample Standard Deviation, s	0.85485041426511
Variance (Sample Standard), s^2	0.73076923076923
Population Standard Deviation, σ	0.82375447104791
Variance (Population Standard), σ^2	0.67857142857143
Total Numbers, N	14
Sum:	119
Mean (Average):	8.5
Standard Error of the Mean ($SE_{\bar{x}}$):	0.22846838336584

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.2715316166342 - 8.7284683833658
90%, $1.645SE_{\bar{x}}$	8.1241695093632 - 8.8758304906368
95%, $1.960SE_{\bar{x}}$	8.0522019686029 - 8.9477980313971
99%, $2.576SE_{\bar{x}}$	7.9114654444496 - 9.0885345555504
99.9%, $3.291SE_{\bar{x}}$	7.748110550343 - 9.251889449657
99.99%, $3.891SE_{\bar{x}}$	7.6110295203235 - 9.3889704796765
99.999%, $4.417SE_{\bar{x}}$	7.4908551506731 - 9.5091448493269
99.9999%, $4.892SE_{\bar{x}}$	7.3823326685743 - 9.6176673314257

Task U.I has clear sequence of screens.

14 responses



Standard Deviation Calculator

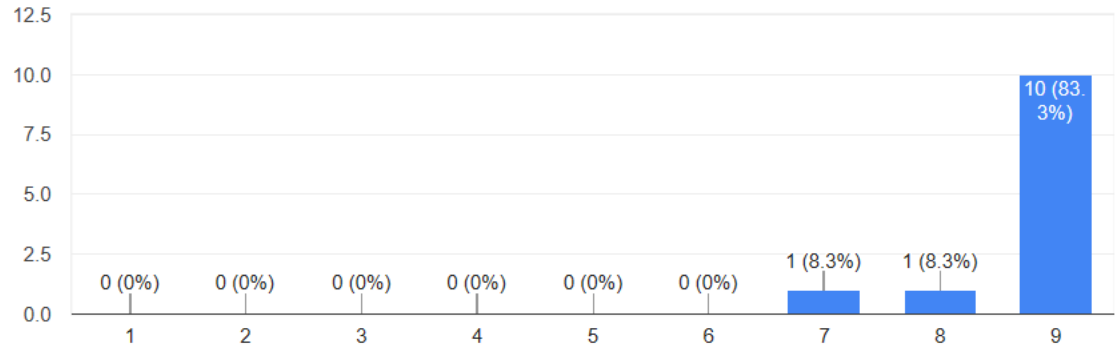
Result	
Sample Standard Deviation, s	0.63332369377665
Variance (Sample Standard), s^2	0.4010989010989
Population Standard Deviation, σ	0.6102859818084
Variance (Population Standard), σ^2	0.37244897959184
Total Numbers, N	14
Sum:	121
Mean (Average):	8.6428571428571
Standard Error of the Mean ($SE_{\bar{x}}$):	0.16926287693131

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.4735942659258 - 8.8121200197885
90%, $1.645SE_{\bar{x}}$	8.3644197103051 - 8.9212945754091
95%, $1.960SE_{\bar{x}}$	8.3111019040718 - 8.9746123816425
99%, $2.576SE_{\bar{x}}$	8.2068359718821 - 9.0788783138322
99.9%, $3.291SE_{\bar{x}}$	8.0858130148762 - 9.1999012708381
99.99%, $3.891SE_{\bar{x}}$	7.9842552887174 - 9.3014589969969
99.999%, $4.417SE_{\bar{x}}$	7.8952230154515 - 9.3904912702627
99.9999%, $4.892SE_{\bar{x}}$	7.8148231489092 - 9.4708911368051

User can view next step/previous stepwhenever he/she liked to.

12 responses



Standard Deviation Calculator

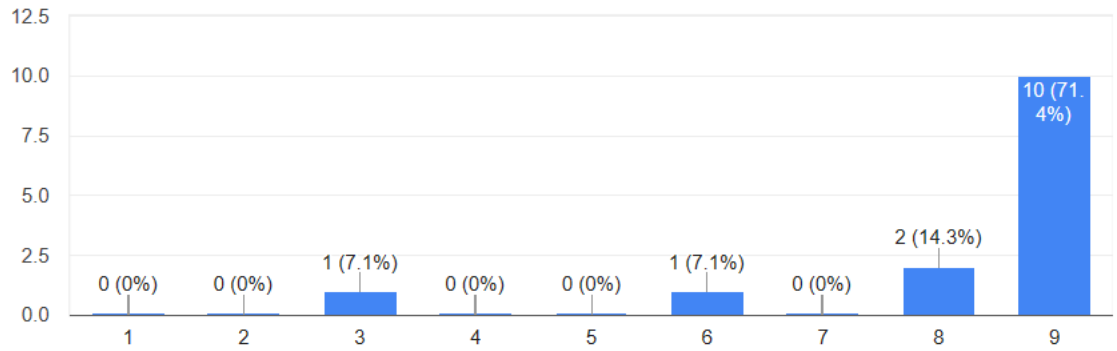
Result	
Sample Standard Deviation, s	0.62158156050806
Variance (Sample Standard), s^2	0.38636363636364
Population Standard Deviation, σ	0.5951190357119
Variance (Population Standard), σ^2	0.35416666666667
Total Numbers, N	12
Sum:	105
Mean (Average):	8.75
Standard Error of the Mean ($SE_{\bar{x}}$):	0.17943514064132

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.5705648593587 - 8.9294351406413
90%, $1.645SE_{\bar{x}}$	8.454829193645 - 9.045170806355
95%, $1.960SE_{\bar{x}}$	8.398307124343 - 9.101692875657
99%, $2.576SE_{\bar{x}}$	8.287775077708 - 9.212224922292
99.9%, $3.291SE_{\bar{x}}$	8.1594789521494 - 9.3405210478506
99.99%, $3.891SE_{\bar{x}}$	8.0518178677646 - 9.4481821322354
99.999%, $4.417SE_{\bar{x}}$	7.9574349837873 - 9.5425650162127
99.9999%, $4.892SE_{\bar{x}}$	7.8722032919827 - 9.6277967080173

User can view the Cautions/warningsrelated to current step.

14 responses



Standard Deviation Calculator

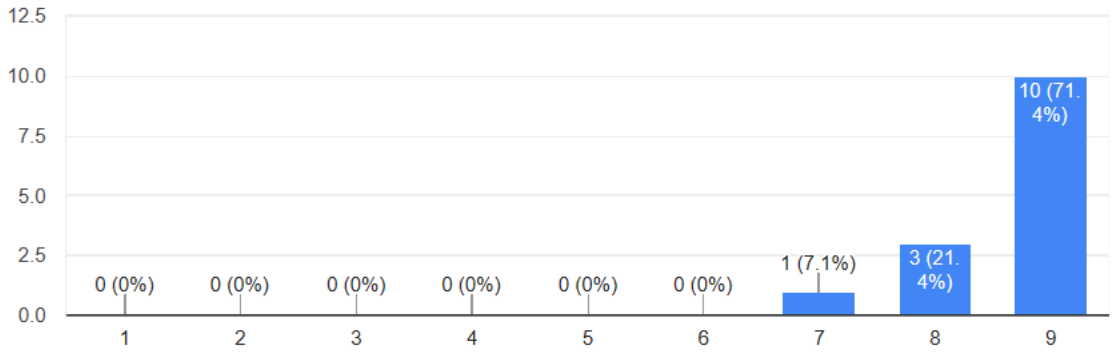
Result	
Sample Standard Deviation, s	1.7177163475235
Variance (Sample Standard), s^2	2.9505494505495
Population Standard Deviation, σ	1.6552328894652
Variance (Population Standard), σ^2	2.7397959183673
Total Numbers, N	14
Sum:	115
Mean (Average):	8.2142857142857
Standard Error of the Mean ($SE_{\bar{x}}$):	0.4590790042924

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	7.7552067099933 - 8.6733647185781
90%, $1.645SE_{\bar{x}}$	7.4591007522247 - 8.9694706763467
95%, $1.960SE_{\bar{x}}$	7.3144908658726 - 9.1140805626988
99%, $2.576SE_{\bar{x}}$	7.0316981992285 - 9.3968732293429
99.9%, $3.291SE_{\bar{x}}$	6.7034567111594 - 9.725114717412
99.99%, $3.891SE_{\bar{x}}$	6.428009308584 - 10.000562119987
99.999%, $4.417SE_{\bar{x}}$	6.1865337523262 - 10.242037676245
99.9999%, $4.892SE_{\bar{x}}$	5.9684712252873 - 10.460100203284

User can view the images related to current step.

14 responses



Standard Deviation Calculator

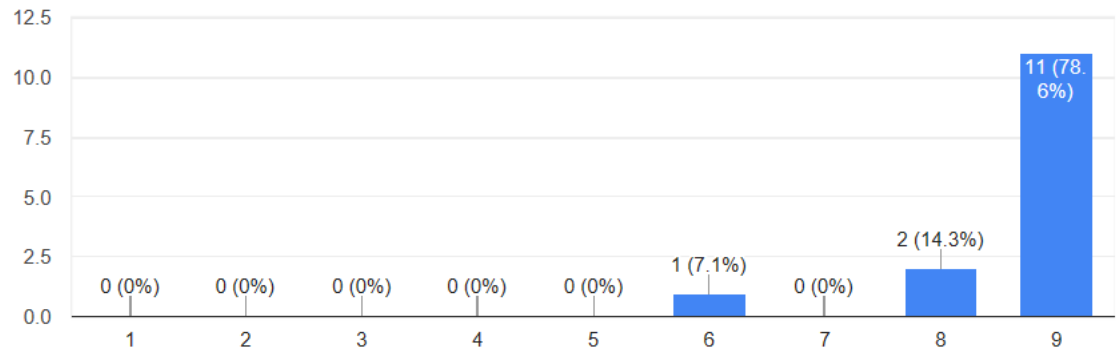
Result	
Sample Standard Deviation, s	0.63332369377665
Variance (Sample Standard), s^2	0.4010989010989
Population Standard Deviation, σ	0.6102859818084
Variance (Population Standard), σ^2	0.37244897959184
Total Numbers, N	14
Sum:	121
Mean (Average):	8.6428571428571
Standard Error of the Mean ($SE_{\bar{x}}$):	0.16926287693131

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.4735942659258 - 8.8121200197885
90%, $1.645SE_{\bar{x}}$	8.3644197103051 - 8.9212945754091
95%, $1.960SE_{\bar{x}}$	8.3111019040718 - 8.9746123816425
99%, $2.576SE_{\bar{x}}$	8.2068359718821 - 9.0788783138322
99.9%, $3.291SE_{\bar{x}}$	8.0858130148762 - 9.1999012708381
99.99%, $3.891SE_{\bar{x}}$	7.9842552887174 - 9.3014589969969
99.999%, $4.417SE_{\bar{x}}$	7.8952230154515 - 9.3904912702627
99.9999%, $4.892SE_{\bar{x}}$	7.8148231489092 - 9.4708911368051

Terminology used in the Task UI is related to the tasks.

14 responses



Standard Deviation Calculator

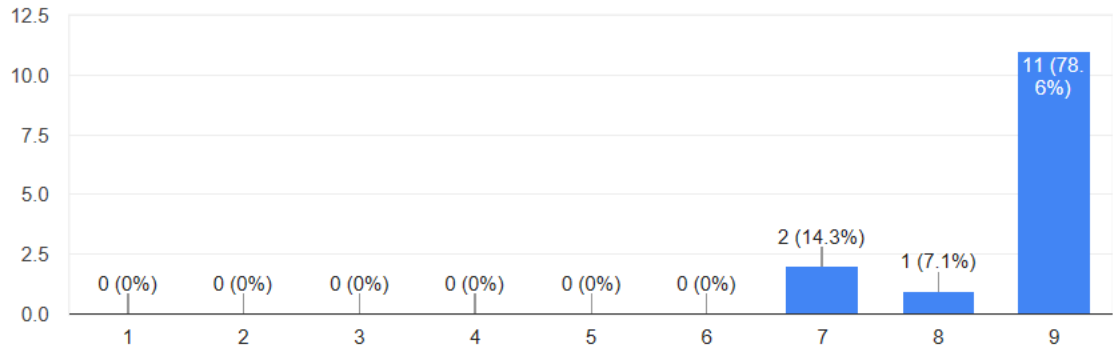
Result	
Sample Standard Deviation, s	0.8418973861411
Variance (Sample Standard), s^2	0.70879120879121
Population Standard Deviation, σ	0.81127262082861
Variance (Population Standard), σ^2	0.65816326530612
Total Numbers, N	14
Sum:	121
Mean (Average):	8.6428571428571
Standard Error of the Mean ($SE_{\bar{x}}$):	0.22500654098289

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.4178506018742 - 8.86786368384
90%, $1.645SE_{\bar{x}}$	8.2727213829403 - 9.012992902774
95%, $1.960SE_{\bar{x}}$	8.2018443225307 - 9.0838699631836
99%, $2.576SE_{\bar{x}}$	8.0632402932852 - 9.2224739924291
99.9%, $3.291SE_{\bar{x}}$	7.9023606164824 - 9.3833536692318
99.99%, $3.891SE_{\bar{x}}$	7.7673566918927 - 9.5183575938216
99.999%, $4.417SE_{\bar{x}}$	7.6490032513357 - 9.6367110343786
99.9999%, $4.892SE_{\bar{x}}$	7.5421251443688 - 9.7435891413455

Highlighting the current step helps user to clearly understand the task.

14 responses



Standard Deviation Calculator

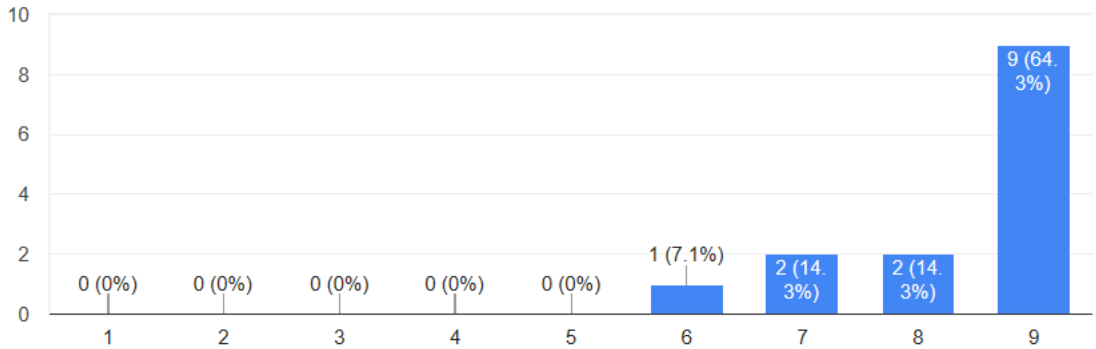
Result	
Sample Standard Deviation, s	0.74494634366849
Variance (Sample Standard), s^2	0.55494505494505
Population Standard Deviation, σ	0.71784825865149
Variance (Population Standard), σ^2	0.51530612244898
Total Numbers, N	14
Sum:	121
Mean (Average):	8.6428571428571
Standard Error of the Mean ($SE_{\bar{x}}$):	0.19909528496696

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.4437618578902 - 8.8419524278241
90%, $1.645SE_{\bar{x}}$	8.3153453990865 - 8.9703688866278
95%, $1.960SE_{\bar{x}}$	8.2526303843219 - 9.0330839013924
99%, $2.576SE_{\bar{x}}$	8.1299876887823 - 9.155726596932
99.9%, $3.291SE_{\bar{x}}$	7.9876345600309 - 9.2980797256834
99.99%, $3.891SE_{\bar{x}}$	7.8681773890507 - 9.4175368966636
99.999%, $4.417SE_{\bar{x}}$	7.7634532691581 - 9.5222610165562
99.9999%, $4.892SE_{\bar{x}}$	7.6688830087988 - 9.6168312769155

Learning to operate the system.

14 responses



Standard Deviation Calculator

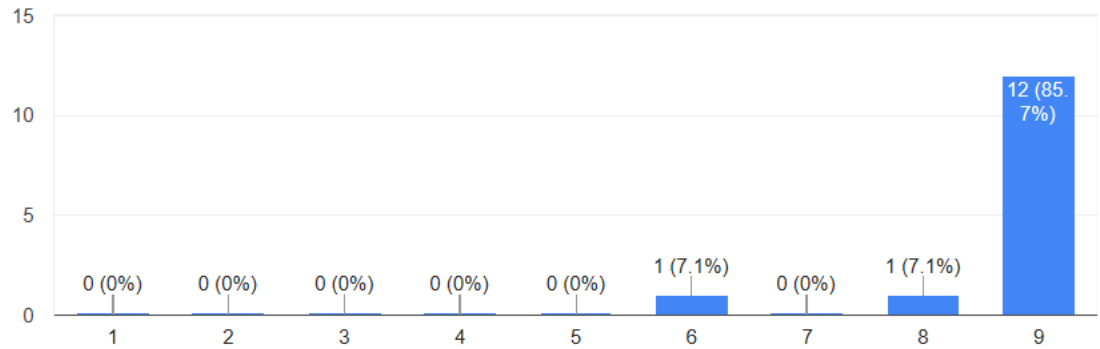
Result	
Sample Standard Deviation, s	1.0082080720186
Variance (Sample Standard), s^2	1.0164835164835
Population Standard Deviation, σ	0.97153360776682
Variance (Population Standard), σ^2	0.94387755102041
Total Numbers, N	14
Sum:	117
Mean (Average):	8.3571428571429
Standard Error of the Mean ($SE_{\bar{x}}$):	0.26945494143383

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.087687915709 - 8.6265977985767
90%, $1.645SE_{\bar{x}}$	7.9138894784842 - 8.8003962358015
95%, $1.960SE_{\bar{x}}$	7.8290111719326 - 8.8852745423532
99%, $2.576SE_{\bar{x}}$	7.6630269280093 - 9.0512587862764
99.9%, $3.291SE_{\bar{x}}$	7.4703666448841 - 9.2439190694016
99.99%, $3.891SE_{\bar{x}}$	7.3086936800238 - 9.4055920342619
99.999%, $4.417SE_{\bar{x}}$	7.1669603808296 - 9.5473253334561
99.9999%, $4.892SE_{\bar{x}}$	7.0389692836486 - 9.6753164306371

Navigation from one step to other is taking minimum time.

14 responses



Standard Deviation Calculator

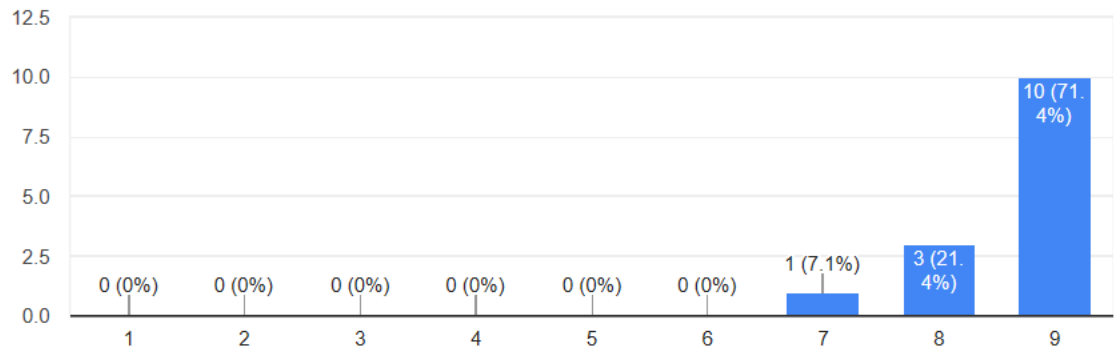
Result	
Sample Standard Deviation, s	0.82542030585556
Variance (Sample Standard), s^2	0.68131868131868
Population Standard Deviation, σ	0.79539490897572
Variance (Population Standard), σ^2	0.63265306122449
Total Numbers, N	14
Sum:	122
Mean (Average):	8.7142857142857
Standard Error of the Mean ($SE_{\bar{x}}$):	0.22060285604269

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.493682858243 - 8.9348885703284
90%, $1.645SE_{\bar{x}}$	8.3513940160955 - 9.0771774124759
95%, $1.960SE_{\bar{x}}$	8.281904116442 - 9.1466673121294
99%, $2.576SE_{\bar{x}}$	8.1460127571197 - 9.2825586714517
99.9%, $3.291SE_{\bar{x}}$	7.9882817150492 - 9.4402897135222
99.99%, $3.891SE_{\bar{x}}$	7.8559200014236 - 9.5726514271478
99.999%, $4.417SE_{\bar{x}}$	7.7398828991452 - 9.6886885294263
99.9999%, $4.892SE_{\bar{x}}$	7.6350965425249 - 9.7934748860465

User can perform the task correctly.

14 responses



Standard Deviation Calculator

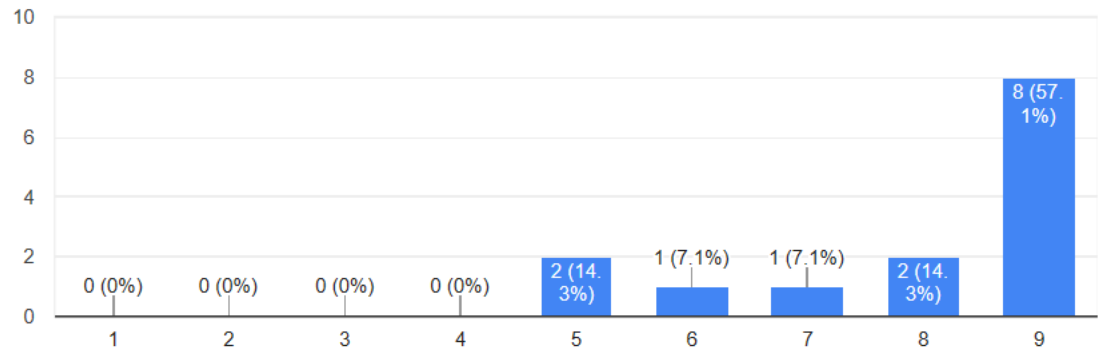
Result	
Sample Standard Deviation, s	0.63332369377665
Variance (Sample Standard), s^2	0.4010989010989
Population Standard Deviation, σ	0.6102859818084
Variance (Population Standard), σ^2	0.37244897959184
Total Numbers, N	14
Sum:	121
Mean (Average):	8.6428571428571
Standard Error of the Mean ($SE_{\bar{x}}$):	0.16926287693131

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.4735942659258 - 8.8121200197885
90%, $1.645SE_{\bar{x}}$	8.3644197103051 - 8.9212945754091
95%, $1.960SE_{\bar{x}}$	8.3111019040718 - 8.9746123816425
99%, $2.576SE_{\bar{x}}$	8.2068359718821 - 9.0788783138322
99.9%, $3.291SE_{\bar{x}}$	8.0858130148762 - 9.1999012708381
99.99%, $3.891SE_{\bar{x}}$	7.9842552887174 - 9.3014589969969
99.999%, $4.417SE_{\bar{x}}$	7.8952230154515 - 9.3904912702627
99.9999%, $4.892SE_{\bar{x}}$	7.8148231489092 - 9.4708911368051

Correcting mistakes.

14 responses



Standard Deviation Calculator

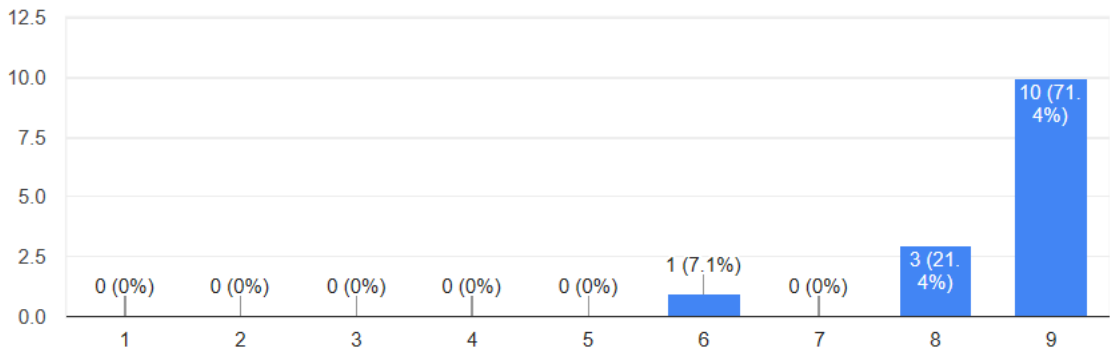
Result	
Sample Standard Deviation, s	1.5424399110244
Variance (Sample Standard), s^2	2.3791208791209
Population Standard Deviation, σ	1.4863322890489
Variance (Population Standard), σ^2	2.2091836734694
Total Numbers, N	14
Sum:	111
Mean (Average):	7.9285714285714
Standard Error of the Mean ($SE_{\bar{x}}$):	0.41223440619566

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	7.5163370223758 - 8.3408058347671
90%, $1.645SE_{\bar{x}}$	7.2504458303796 - 8.6066970267633
95%, $1.960SE_{\bar{x}}$	7.1205919924279 - 8.7365508647149
99%, $2.576SE_{\bar{x}}$	6.8666555982114 - 8.9904872589315
99.9%, $3.291SE_{\bar{x}}$	6.5719079977815 - 9.2852348593614
99.99%, $3.891SE_{\bar{x}}$	6.3245673540641 - 9.5325755030788
99.999%, $4.417SE_{\bar{x}}$	6.1077320564052 - 9.7494108007377
99.9999%, $4.892SE_{\bar{x}}$	5.9119207134622 - 9.9452221436806

System speed.

14 responses



Standard Deviation Calculator

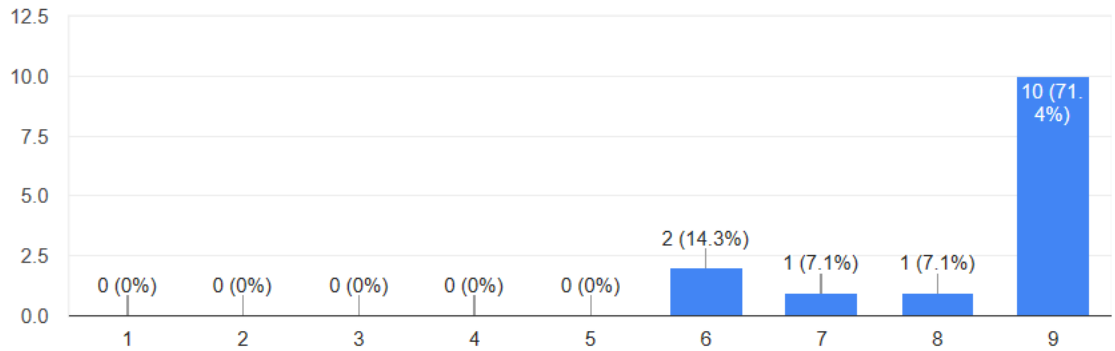
Result	
Sample Standard Deviation, s	0.85163062725264
Variance (Sample Standard), s^2	0.72527472527473
Population Standard Deviation, σ	0.82065180664829
Variance (Population Standard), σ^2	0.6734693877551
Total Numbers, N	14
Sum:	120
Mean (Average):	8.5714285714286
Standard Error of the Mean ($SE_{\bar{x}}$):	0.2276078590902

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.3438207123384 - 8.7990364305188
90%, $1.645SE_{\bar{x}}$	8.1970136432252 - 8.9458434996319
95%, $1.960SE_{\bar{x}}$	8.1253171676118 - 9.0175399752454
99%, $2.576SE_{\bar{x}}$	7.9851107264122 - 9.1577464164449
99.9%, $3.291SE_{\bar{x}}$	7.8223711071627 - 9.3204860356944
99.99%, $3.891SE_{\bar{x}}$	7.6858063917086 - 9.4570507511485
99.999%, $4.417SE_{\bar{x}}$	7.5660846578272 - 9.57677248503
99.9999%, $4.892SE_{\bar{x}}$	7.4579709247593 - 9.6848862180978

User can find all the information related to the step.

14 responses



Standard Deviation Calculator

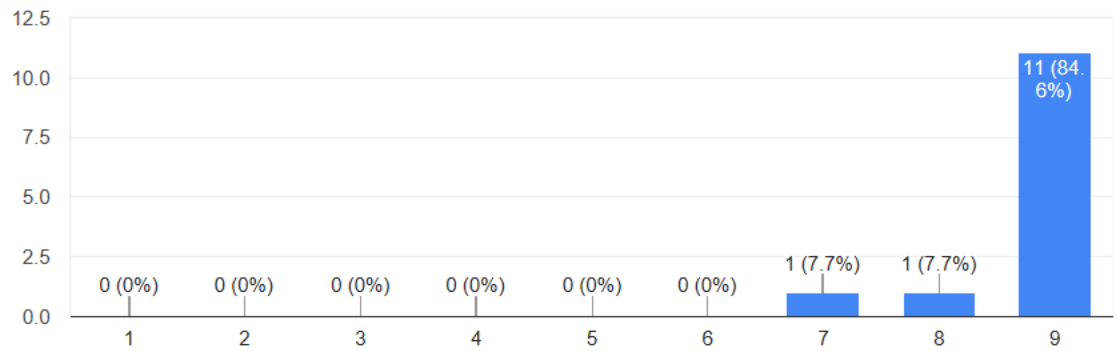
Result	
Sample Standard Deviation, s	0.93761446187699
Variance (Sample Standard), s^2	0.87912087912088
Population Standard Deviation, σ	0.90350790290525
Variance (Population Standard), σ^2	0.81632653061224
Total Numbers, N	14
Sum:	120
Mean (Average):	8.5714285714286
Standard Error of the Mean ($SE_{\bar{x}}$):	0.25058800551629

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.3208405659123 - 8.8220165769449
90%, $1.645SE_{\bar{x}}$	8.1592113023543 - 8.9836458405029
95%, $1.960SE_{\bar{x}}$	8.0802760806166 - 9.0625810622405
99%, $2.576SE_{\bar{x}}$	7.9259138692186 - 9.2169432736385
99.9%, $3.291SE_{\bar{x}}$	7.7467434452744 - 9.3961136975827
99.99%, $3.891SE_{\bar{x}}$	7.5963906419647 - 9.5464665008925
99.999%, $4.417SE_{\bar{x}}$	7.4645813510631 - 9.678275791794
99.9999%, $4.892SE_{\bar{x}}$	7.3455520484429 - 9.7973050944143

User is notified when the task is completed.

13 responses



Standard Deviation Calculator

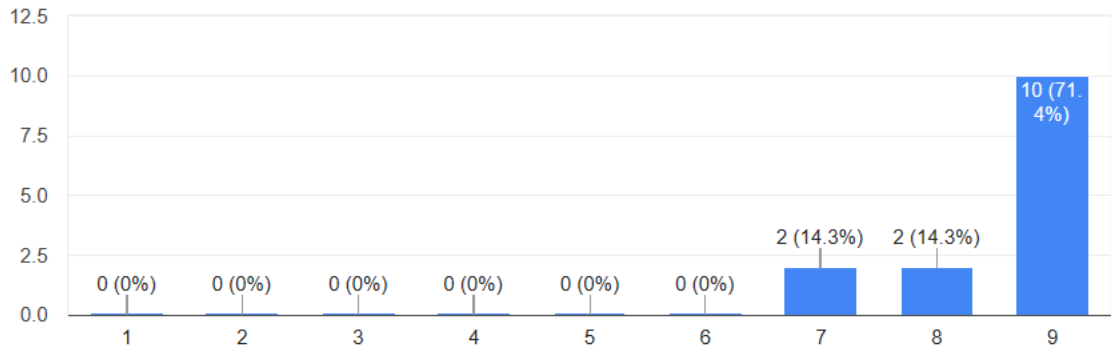
Result	
Sample Standard Deviation, s	0.57893422352184
Variance (Sample Standard), s^2	0.33516483516484
Population Standard Deviation, σ	0.55787497685048
Variance (Population Standard), σ^2	0.31122448979592
Total Numbers, N	14
Sum:	123
Mean (Average):	8.7857142857143
Standard Error of the Mean ($SE_{\bar{x}}$):	0.15472667956405

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.6309876061502 - 8.9404409652783
90%, $1.645SE_{\bar{x}}$	8.5311888978314 - 9.0402396735972
95%, $1.960SE_{\bar{x}}$	8.4824499937687 - 9.0889785776598
99%, $2.576SE_{\bar{x}}$	8.3871383591573 - 9.1842902122713
99.9%, $3.291SE_{\bar{x}}$	8.276508783269 - 9.2949197881596
99.99%, $3.891SE_{\bar{x}}$	8.1836727755306 - 9.387755795898
99.999%, $4.417SE_{\bar{x}}$	8.1022865420799 - 9.4691420293487
99.9999%, $4.892SE_{\bar{x}}$	8.0287913692869 - 9.5426372021416

0verall task User Interface.

14 responses



Standard Deviation Calculator

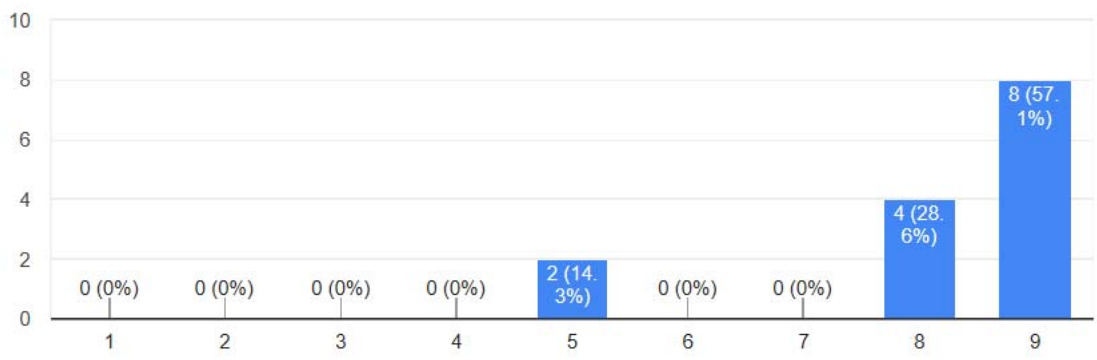
Result	
Sample Standard Deviation, s	0.75592894601845
Variance (Sample Standard), s^2	0.57142857142857
Population Standard Deviation, σ	0.72843135908468
Variance (Population Standard), σ^2	0.53061224489796
Total Numbers, N	14
Sum:	120
Mean (Average):	8.5714285714286
Standard Error of the Mean ($SE_{\bar{x}}$):	0.20203050891044

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8.3693980625181 - 8.773459080339
90%, $1.645SE_{\bar{x}}$	8.2390883842709 - 8.9037687585862
95%, $1.960SE_{\bar{x}}$	8.1754487739641 - 8.967408368893
99%, $2.576SE_{\bar{x}}$	8.0509979804753 - 9.0918591623819
99.9%, $3.291SE_{\bar{x}}$	7.9065461666043 - 9.2363109762528
99.99%, $3.891SE_{\bar{x}}$	7.785327861258 - 9.3575292815991
99.999%, $4.417SE_{\bar{x}}$	7.6790598135711 - 9.463797329286
99.9999%, $4.892SE_{\bar{x}}$	7.5830953218387 - 9.5597618210185

Holograms make the task process easier.

14 responses



Standard Deviation Calculator

Result	
Sample Standard Deviation, s	1.0690449676497
Variance (Sample Standard), s^2	1.1428571428571
Population Standard Deviation, σ	1.0301575072754
Variance (Population Standard), σ^2	1.0612244897959
Total Numbers, N	14
Sum:	116
Mean (Average):	8.2857142857143
Standard Error of the Mean ($SE_{\bar{x}}$):	0.28571428571429

Confidence Interval Approximations, If sampling distribution of the mean follows normal distribution

Confidence Level	Range
68.3%, $SE_{\bar{x}}$	8 - 8.5714285714286
90%, $1.645SE_{\bar{x}}$	7.8157142857143 - 8.7557142857143
95%, $1.960SE_{\bar{x}}$	7.7257142857143 - 8.8457142857143
99%, $2.576SE_{\bar{x}}$	7.5497142857143 - 9.0217142857143
99.9%, $3.291SE_{\bar{x}}$	7.3454285714286 - 9.226
99.99%, $3.891SE_{\bar{x}}$	7.174 - 9.3974285714286
99.999%, $4.417SE_{\bar{x}}$	7.0237142857143 - 9.5477142857143
99.9999%, $4.892SE_{\bar{x}}$	6.888 - 9.6834285714286

Suggestions

3 responses

Slow rendering
a great work.
implementing more Holograms identifying certain objects could help

Metrics Analysis

In looking at the demographics of the test subjects, the first point to make is that sample base is only fourteen and the demographics are not very diverse. Typically, you would like the test subjects to be of varying demographics. For example, the following statistics are noted as follows:

- 7 of 11 test subjects are female but not documented are 3 others that were also female
- 14 of 14 test subjects are under the age of 29 years old
- 9 of 11 are college graduates
- 9 of 11 are single
- 9 of 11 are asian
- 10 of 11 are unemployed

If there were more test subjects and from a more diverse demographics, then the metrics provided would have more validity. However, given the circumstances of an academic environment time limited by the semester, the fourteen test subjects are the best that can be provided. The four descriptive statistics that are recongnized in common statistical analysis are as follows:

- Measures of Frequency is the number of times a data point occurs.
- Measures of Central Tendency is the Median score for a category.
- Measures of Dispersion or Variance is the variance or stard deviation of the test samples.
- Measures of Position describes the scores in relation to eachother.

While some of the statistics may have had out-lying data points in general most of the data was fairly consistant. We saw that some test-takers did not like the hand gestures. That could have been corrected with beteer preparation of the test-taker prior to executing the study. And likewise, some users thought that it was difficult to find data on the HoloLens but over time and with some refinement that can be resolved.

PROJECT ANALYSIS

BUSINESS LOGIC

Summary of What Occurred:

- The Capstone team engineered an Augmented Reality for the NASA S.U.I.T.S. Challenge using the Microsoft HoloLens platform.
- Our group 4 developed the BUSINESS LOGIC which is the middleware that interfaces with the database and the user interface of the HoloLens.
- The Business Logic interfaces with the database to obtain information concerning the tasks that the astronaut is to perform and health data for the astronaut to view.
- The Business Logic interfaces with the Task UI to display a list of tasks for the astronaut to view on the HoloLens.
- The Business Logic interfaces with the Health UI to display information

From the perspective of the work distribution within our group. The three of us divided the work on an as need basis which led to the following work distribution between documentation and code development:

Team member 1: 90% documentation, 10% code development
Team member 2: 50% documentation, 50% code development
Team member 3: 10% documentation, 90% code development

Problems:

- Specifically defining the software requirements.
- Minimizing misunderstandings when developing the code
- Getting the various groups together at the same time.
- Accurately being able to track the software progress.

What did we learn?

- Learned about AR, Augmented Reality.
- Learned to apply the knowledge and techniques from the SWEN ciriculum
- Good Understanding about deploying to HoloLens from Unity
- Valuable information about analog scaling and displaying on a User Interface

What would we do differently?

- Create a Gantt Chart to track progress of the project.

Future Improvements:

- Alarm setpoints would be dynamic and modifiable by the astronaut on the HoloLens.
- Give the user the ability to modify the task procedure from the HoloLens versus hard coding.

TASK UI

Summary of What Occurred:

- We designed and created spacesuit informatics using an augmented reality (AR) Microsoft HoloLens platform.
- These visual display and audio environments will present information to aid astronaut subjects in performing simulated Extravehicular Activity (EVA) tasks
- To effectively guide a subject through a procedure, the design will use holograms that are spatially mapped to the EVA task board
- Ground Controller detects the problem and send the warnings to the Astronaut.
- Astronaut checks the warning and locate the box by getting the directional guidance from the database.
- Astronaut views the available tasks in the database projected by the timer on the HoloLens.
- The voice commands will be used by the astronaut as primary source of interaction with the system.
- The hand gestures will be secondary interaction to keep the hands of astronaut free while performing a task.
-

As part of our project, we designed Task U.I system which contains 3 panels
1.Task Panel:
Task Panel is located on the righthand side of the screen and it has 3 rows for displaying the task steps namely Previous Step, Current Step, Next Step and the current step will be highlighted.
2.Caution/Warning Panel:
Caution/Warning Panel is located on the top middle of the screen and it displays Cautions/Warnings related to the current step.
If there is a Caution related to the Current Step, the panel will be turned yellow
If there is a Warning related to the Current Step, the panel will be turned Red
3.Image Panel:
Image Panel is located on the top right of the screen and it displays the image related to the current step.

- User will be able to view the Pictures, Cautions/Warnings and Hologram related to the Current step
- User will be notified when the task1 is completed and use the voice command to perform the task2.

Problems:

- User is facing difficulty in identifying Parts related to task step
- Font Size
- Current step to get repeated

What did we learn:

- creating and importing assets in unity
- Knowledge about build and run project in Unity
- Good Understanding about deploying to HoloLens from Unity
- Created World Space Canvas and create U.I elements as Children to the Canvas in Unity3D

- Created Scripts for displaying text and for voice commands using C# in Unity
- Add text, images to Canvas via Scripts

What would we do differently?

- Pin Task panel to the Wall
- Zoom image panel
- Pointing highlighted holograms using3D arrows
- Parts related to the task step should get highlighted

Future Improvements:

- Object detection and Animations for each task step
- Holograms should display entire process of what need to be performed

CODE

AlternativeHealthUIUpdates.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class AlternativeHealthUIUpdates : MonoBehaviour {

    //public GameObject detailedPie;

    public GameObject displayDetails;
    public GameObject DetailPanel;
    public DisplayDetails dd;
    public BL_Main bl_main;

    public Image pressureFillImage;
    public Image heartRateImage;
    public Image bodyTemperatureImage;

    public Slider PrimaryOxygenSlider;
    public Image PrimaryO2Image;

    public Slider SecondaryOxygenSlider;
    public Image secondaryO2Image;

    public Slider WaterSilder;
    public Image WaterImage;

    public Slider BatterySilder;
    public Image BatteryImage;

    public ColorCode flagColors;



    CommonData commonData = CommonData.GetInstance();

    // Use this for initialization
    void Start()
    {
        //detailedPie.SetActive(false);
```

```
        displayDetails.SetActive(false);
        DetailPanel.SetActive(false);
    }

    // Update is called once per frame
    void Update()
    {
        UpdateDetailPiePanel();
        UpdateViewDetails();
        UpdateWarningWindow();
        UpdateSliders();
    }

    void UpdateSliders()
    {
        UpdatePressureSlider();
        UpdateHeartRateSlider();
        UpdateBodyTemperature();
        UpdatePrimaryO2();
        UpdateSecondaryO2();
        UpdateWater();
        UpdateBattery();
    }

    private void UpdateBattery()
    {
        BatterySilder.value = bl_main.bl_scaling.scallingBattery() / 100;

        if(commonData.BatteryValue > commonData.BatteryLoDB)
        {
            BatteryImage.color = flagColors.HColor;
        }
        else if(commonData.BatteryLoDB > commonData.BatteryValue && commonData.BatteryValue >
commonData.BatteryLoLoDB)
        {
            BatteryImage.color = flagColors.LColor;
        }
        else if(commonData.BatteryLoLoDB > commonData.BatteryValue)
        {
            BatteryImage.color = flagColors.LLColor;
        }
    }

    private void UpdateWater()
    {
        WaterSilder.value = bl_main.bl_scaling.scallingWater() / 100;
```

```
        if(commonData.WaterLoDB < commonData.WaterValue)
        {
            WaterImage.color = flagColors.HColor;
        }
        else if(commonData.WaterLoDB > commonData.WaterValue && commonData.WaterValue >
commonData.WaterLoLoDB)
        {
            WaterImage.color = flagColors.LColor;
        }
        else if(commonData.WaterLoLoDB > commonData.WaterValue)
        {
            WaterImage.color = flagColors.LLColor;
        }
    }

    void UpdatePressureSlider()
    {
        float newPressurePosition = bl_main.bl_scaling.ScalingFunction().Map(0, 100, -66, 68);
        pressureFillImage.rectTransform.localPosition = new Vector3(newPressurePosition, 0f, 0f);

        float SuitPressure = commonData.SuitPressureValue;
        if (SuitPressure >= commonData.SuitPressHiHiDB && SuitPressure <= commonData.SuitPressHiHiSP)
        {
            pressureFillImage.color = flagColors.LLColor;
        }
        else if (SuitPressure >= commonData.SuitPressHiDB && SuitPressure <=
commonData.SuitPressHiSP)
        {
            pressureFillImage.color = flagColors.LColor;
        }
        else if(SuitPressure <= commonData.SuitPressHiDB && SuitPressure >=
commonData.SuitPressLoDB)
        {
            pressureFillImage.color = flagColors.HColor;
        }
        else if (SuitPressure >= commonData.SuitPressLoSP && SuitPressure <=
commonData.SuitPressLoDB)
        {
            pressureFillImage.color = flagColors.LColor;
        }
        else if (SuitPressure >= commonData.SuitPressLoLoSP && SuitPressure <=
commonData.SuitPressLoLoDB)
        {
            pressureFillImage.color = flagColors.LLColor;
        }
    }

    void UpdateHeartRateSlider()
```

Health UI

```
    {
        float newHeartRatePosition = bl_main.bl_scaling.scallingHeartRate().Map(0f, 100f, 12.5f, 148f);
        heartRateImage.rectTransform.localPosition = new Vector3(newHeartRatePosition -80f, 0f, 0f);

        float heartRate = commonData.HeartRateValue;

        if (heartRate >= commonData.HeartRateHiHiDB && heartRate <= commonData.HeartRateHiHiSP)
        {
            heartRateImage.color = flagColors.LLColor;
        }
        else if (heartRate >= commonData.HeartRateHiDB && heartRate <= commonData.HeartRateHiSP)
        {
            heartRateImage.color = flagColors.LColor;
        }
        else if (heartRate <= commonData.HeartRateHiDB && heartRate >= commonData.HeartRateLoDB)
        {
            heartRateImage.color = flagColors.HColor;
        }
        else if (heartRate >= commonData.HeartRateLoSP && heartRate <= commonData.HeartRateLoDB)
        {
            heartRateImage.color = flagColors.LColor;
        }
        else if (heartRate >= commonData.HeartRateLoLoSP && heartRate <=
commonData.HeartRateLoLoDB)
        {
            heartRateImage.color = flagColors.LLColor;
        }
    }

    void UpdateBodyTemperature()
    {
        float newBodyTemperaturePosition = bl_main.bl_scaling.scallingBodyTemperature().Map(0f, 100f,
8f, 142f);
        bodyTemperatureImage.rectTransform.localPosition = new Vector3(newBodyTemperaturePosition -
70f, 0f, 0f);

        float bodyTemperature = commonData.BodyTemperatureValue;

        if (bodyTemperature >= commonData.BodyTemperatureHiHiDB && bodyTemperature <=
commonData.BodyTemperatureHiHiSP)
        {
            bodyTemperatureImage.color = flagColors.LLColor;
        }
        else if (bodyTemperature >= commonData.BodyTemperatureHiDB && bodyTemperature <=
commonData.BodyTemperatureHiSP)
        {
            bodyTemperatureImage.color = flagColors.LColor;
        }
```

```
    else if (bodyTemperature <= commonData.BodyTemperatureHiDB && bodyTemperature >=
commonData.BodyTemperatureLoDB)
    {
      bodyTemperatureImage.color = flagColors.HColor;
    }
    else if (bodyTemperature >= commonData.BodyTemperatureLoSP && bodyTemperature <=
commonData.BodyTemperatureLoDB)
    {
      bodyTemperatureImage.color = flagColors.LColor;
    }
    else if (bodyTemperature >= commonData.BodyTemperatureLoLoSP && bodyTemperature <=
commonData.BodyTemperatureLoLoDB)
    {
      bodyTemperatureImage.color = flagColors.LLColor;
    }
  }

  void UpdatePrimaryO2()
  {
    PrimaryOxygenSlider.value = bl_main.bl_scaling.scallingPrimaryOxygen() / 100;

    if(commonData.OxygenOneValue >commonData.PrimaryOxygenLoDB)
    {
      PrimaryO2Image.color = flagColors.HColor;
    }
    else if(commonData.OxygenOneValue <= commonData.PrimaryOxygenLoDB)
    {
      PrimaryO2Image.color = flagColors.LColor;
    }
  }

  void UpdateSecondaryO2()
  {

    Debug.Log(bl_main.bl_scaling.scallingSeondaryOxygen() / 100);
    SecondaryOxygenSlider.value = bl_main.bl_scaling.scallingSeondaryOxygen() / 100;

    if(commonData.OxygenTwoValue > commonData.SecondaryOxygenHiDB)
    {
      secondaryO2Image.color = flagColors.HHColor;
    }
    else if (commonData.SecondaryOxygenHiDB > commonData.OxygenTwoValue &&
commonData.OxygenTwoValue > commonData.SecondaryOxygenLoDB)
    {
      secondaryO2Image.color = flagColors.HColor;
    }
    else if(commonData.SecondaryOxygenLoDB > commonData.OxygenTwoValue &&
commonData.OxygenTwoValue > commonData.SecondaryOxygenLoLoDB)
```

```
    {
        secondaryO2Image.color = flagColors.LColor;
    }
    else if(commonData.OxygenTwoValue < commonData.SecondaryOxygenLoLoDB)
    {
        secondaryO2Image.color = flagColors.LLColor;
    }
}


void UpdateDetailPiePanel()
{
    //if (commonData.viewPieDetails)
    //{
    //    //detailedPie.SetActive(true);
    //    pie.SetActive(false);
    //}
    //else if (commonData.closePieDetails)
    //{
    //    //detailedPie.SetActive(false);
    //    pie.SetActive(true);
    //}
}

void UpdateViewDetails()
{
    if (commonData.viewPressure)
    {
        displayDetails.SetActive(true);
        dd.displayingType = displayType.SuitPressure;
        commonData.viewPressure = false;
    }
    else if (commonData.viewPrimaryOTwo)
    {
        displayDetails.SetActive(true);
        dd.displayingType = displayType.Oxygen;
        commonData.viewPrimaryOTwo = false;
    }
    else if (commonData.viewSecondaryOTwo)
    {
        displayDetails.SetActive(true);
        dd.displayingType = displayType.OxygenTwo;
        commonData.viewSecondaryOTwo = false;
    }

    else if (commonData.viewBattery)
    {
        displayDetails.SetActive(true);
```

```
        dd.displayingType = displayType.Battery;
        commonData.viewBattery = false;
    }
    else if (commonData.viewBodyTemperature)
    {
        displayDetails.SetActive(true);
        dd.displayingType = displayType.BodyTemperature;
        commonData.viewBodyTemperature = false;
    }
    else if (commonData.viewHeartRate)
    {
        displayDetails.SetActive(true);
        dd.displayingType = displayType.HeartRate;
        commonData.viewHeartRate = false;
    }
    else if (commonData.viewHTwoO)
    {
        displayDetails.SetActive(true);
        dd.displayingType = displayType.Water;
        commonData.viewHTwoO = false;
    }
    else if (commonData.closeDetailWindow)
    {
        displayDetails.SetActive(false);
        //  Debug.Log("Close DEtail");
        commonData.closeDetailWindow = false;
    }
}


void UpdateWarningWindow()
{
    if (commonData.viewWarnings)
    {
        DetailPanel.SetActive(true);
        commonData.viewWarnings = false;
    }
    else if (commonData.closeWarnings)
    {
        DetailPanel.SetActive(false);
        commonData.closeWarnings = false;
    }

    if(DetailPanel.active)
    {
        if(bl_main.bl_alarming.GetCurrentAlarm == null)
        {
            DetailPanel.SetActive(false);
```

```
            }
        }
    }
}

public static class ExtensionMethods
{
    public static float Map(this float value, float low1,float high1,float low2, float high2)
    {
        return low2 + (value - low1) * (high2 - low2) / (high1 - low1);
    }
}
```

Amazon.cs

```
using System;
using UnityEngine;

public class amazon : MonoBehaviour
{
   private object client;

   // Use this for initialization
   void Start()
   {
      string accessKey = "AKIAID34YIUFUW4PD5XQ";
      string secretKey = "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";

      //AmazonS3Config config = new AmazonS3Config();
      //config.ServiceURL = "https://nasasuits.s3.us-east-
2.amazonaws.com/disablingalarmprocedure.json?X-Amz-Expires=3597&X-Amz-Algorithm=AWS4-
HMAC-SHA256&X-Amz-Credential=AKIAJDD2SLSSRJWQWBRA/20180417/us-east-
2/s3/aws4_request&X-Amz-Date=20180417T193400Z&X-Amz-SignedHeaders=host&X-Amz-
Signature=8b1732edc44eba22b33958753a23bead002a37bdfe1c7d51358ae7031c5e3b51";

      //AmazonS3Client s3Client = new AmazonS3Client(
      //       accessKey,
      //       secretKey,
      //       config
      //       );

      //GetObjectRequest request = new GetObjectRequest();
      //request.BucketName = "nasasuits";
      //request.Key = "disablingalarmprocedure.json";
      //GetObjectResponse response = client.GetObject(request);
      //response.WriteResponseStreamToFile("C:\Users\Bhaavan Bruhant\Desktop\TASKS");

      //ListBucketsResponse response = client.ListBuckets();
      //foreach (S3Bucket b in response.Buckets)
      //{
      //    System.Console.WriteLine("{0}\t{1}", b.BucketName, b.CreationDate);
      //}
      //ListObjectsRequest request = new ListObjectsRequest();
      //request.BucketName = "my-new-bucket";
      //ListObjectsResponse response = client.ListObjects(request);
      //foreach (S3Object o in response.S3Objects)
      //{
      //    Console.WriteLine("{0}\t{1}\t{2}", o.Key, o.Size, o.LastModified);
      //}
   }
```

Database

```
    // Update is called once per frame
    void Update()
    {
    }
}
```

Database

BL_Alarming.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using UnityEngine;

public class BL_Alarming {

    private CommonData commonData = CommonData.GetInstance();
    private Alarms alarms = new Alarms();
    private Alarms alarmsHistory = new Alarms();

    public double SuitPressure;
    public double HeartRate;
    public double BodyTemperature;
    public double PrimaryOxygen;
    public double SecondaryOxygen;
    public double Water;
    public double Battery;

    int index = 0;

    public Alarm GetCurrentAlarm
    {
        get {
            if (alarms.alarms.Count == 0)
            {
                return null;
            }
            else
            {
                return alarms.alarms[0];
            }
        }
    }

    public Alarm GetPerviousAlarm
    {
        get{
            if(index <= 0)
            {
                return null;
            }
            else
            {
```

```
            return alarms.alarms[index - 1];
        }
    }
}

public Alarm GetNextAlarm
{
    get
    {
        if(index >= alarms.alarms.Count -1)
        {
            return null;
        }
        else
        {
            return alarms.alarms[index + 1];
        }
    }
}

//Turn flag checking on and off

//Suit Pressure
private bool SuitPressureHiHiEn = true;
private bool SuitPressureHiEn = true;
private bool SuitPressureLoLoEn = true;
private bool SuitPressureLoEn = true;

//HearRate
private bool HeartRateHiHiEn = true;
private bool HeartRateHiEn = true;
private bool HeartRateLoLoEn = true;
private bool HeartRateLoEn = true;

//BodyTemperature
private bool BodyTemperatureHiHiEn = true;
private bool BodyTemperatureHiEn = true;
private bool BodyTemperatureLoLoEn = true;
private bool BodyTemperatureLoEn = true;

//PrimaryOxygen
private bool PrimaryOxygenLoEn = true;

//SecondaryOxygen
private bool SecondaryOxygenLoEn = true;
private bool SecondaryOxygenLoLoEn = true;

//Water
```

```
private bool WaterLoEn = true;
private bool WaterLoLoEn = true;

//Battery
private bool BatteryLoEn = true;
private bool BatteryLoLoEn = true;

//Constructor
public BL_Alarming()
{
   //SuitPressure = commonData.SuitPressureValue;
   //HeartRate = commonData.HeartRateValue;
   //BodyTemperature = commonData.BodyTemperatureValue;
   //PrimaryOxygen = commonData.OxygenOneValue;
   //SecondaryOxygen = commonData.OxygenTwoValue;
   //Water = commonData.WaterValue;
   //Battery = commonData.BatteryValue;
}

public void AcknowldegeAlarmFunction()
{
   //Put alarm in temp variable and remove from alarms
   Alarm ackAlarm = alarms.alarms[index];
   alarms.alarms.RemoveAt(index);

   //Set Acknowleged time
   ackAlarm.timeAcknowledged = DateTime.Now;

   //Insert into History
   alarmsHistory.alarms.Insert(0, ackAlarm);

   //if we clear the last alarm we have to shrink the index
   if(index > alarms.alarms.Count-1 && index != 0)
   {
      index--;
   }

}

public void NextAlarmFunction()
{
   if(alarms.alarms.Count()-1 > index)
   {
      index++;
   }
}

public void PreviousAlarmFunction()
```

```
    {
        if(index > 0)
        {
            index--;
        }
    }

    // Alarming function
    public void BLAlarmingFunction()
    {
        ProcessVoiceCommands();
        ProcessPressure();
        ProcessHeartRate();
        ProcessPrimaryOxygen();
        ProcessSecondaryOxygen();
        ProcessWater();
        ProcessBattery();
    }

    //SuitPressure
    private void ProcessPressure()
    {
        SuitPressure = commonData.SuitPressureValue;
        if (SuitPressure >= commonData.SuitPressureMin && SuitPressure <=
commonData.SuitPressureMax)
        {

            // If SuitPressureHiHi enables, it will check how much suit pressure is and turn the alarm on or off,
depends on suit pressure value
            if (SuitPressureHiHiEn == true)
            {
                if (SuitPressure >= commonData.SuitPressHiHiDB && SuitPressure <=
commonData.SuitPressHiHiSP)
                {
                    if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.SuitPressureHiHiStatus) == null)
                    {
                        Alarm alarm = new Alarm(AlarmType.SuitPressureHiHiStatus, "HiHi Suit Pressure");
                        alarms.alarms.Insert(0, alarm);

                    }
                }
            }

            if (SuitPressureHiEn == true)
            {
                if (SuitPressure >= commonData.SuitPressHiDB && SuitPressure <=
commonData.SuitPressHiSP)
                {
```

```
            if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.SuitPressureHiStatus) == null)
            {
                Alarm alarm = new Alarm(AlarmType.SuitPressureHiStatus, "High Suit Pressure");
                alarms.alarms.Insert(0, alarm);
            }
        }
    }

    if (SuitPressureLoEn == true)
    {
        if (SuitPressure >= commonData.SuitPressLoSP && SuitPressure <=
commonData.SuitPressLoDB)
        {
            if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.SuitPressureLoStatus) == null)
            {
                Alarm alarm = new Alarm(AlarmType.SuitPressureLoStatus, "Low Suit Pressure");
                alarms.alarms.Insert(0, alarm);
            }
        }
    }

    if (SuitPressureLoLoEn == true)
    {
        if (SuitPressure >= commonData.SuitPressLoLoSP && SuitPressure <=
commonData.SuitPressLoLoDB)
        {
            if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.SuitPressureLoLoStatus) == null)
            {
                Alarm alarm = new Alarm(AlarmType.SuitPressureLoLoStatus, "LowLow Suit Pressure");
                alarms.alarms.Insert(0, alarm);
            }
        }
    }
  }
  else
  {
    Debug.Log("Something wrong here!");
  }
}

//HearRate
private void ProcessHeartRate ()
{
  HeartRate = commonData.HeartRateValue;
  if (HeartRate >= commonData.HeartRateMin && HeartRate <= commonData.HeartRateMax)
  {
```

```
        // If HearRateHiHi enables, it will check how much hear rate is and turn the alarm on or off,
depends on heart rate value
        if (HeartRateHiHiEn == true)
        {
           if (HeartRate >= commonData.HeartRateHiHiDB && HeartRate <=
commonData.HeartRateHiHiSP)
           {
              if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.HeartRateHiHiStatus) == null)
              {
                 Alarm alarm = new Alarm(AlarmType.HeartRateHiHiStatus, "HiHi Heart rate");
                 alarms.alarms.Insert(0, alarm);

              }
           }
        }

        if (HeartRateHiEn == true)
        {
           if (HeartRate >= commonData.HeartRateHiDB && HeartRate <= commonData.HeartRateHiSP)
           {

              if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.HeartRateHiStatus) == null)
              {
                 Alarm alarm = new Alarm(AlarmType.HeartRateHiStatus, "High Heart Rate");
                 alarms.alarms.Insert(0, alarm);
              }
           }
        }

        if (HeartRateLoEn == true)
        {
           if (HeartRate >= commonData.HeartRateLoSP && HeartRate <= commonData.HeartRateLoDB)
           {
              if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.HeartRateLoStatus) == null)
              {
                 Alarm alarm = new Alarm(AlarmType.HeartRateLoStatus, "Low Heart Rate");
                 alarms.alarms.Insert(0, alarm);
              }
           }
        }

        if (HeartRateLoLoEn == true)
        {
           if (HeartRate >= commonData.HeartRateLoLoSP && HeartRate <=
commonData.HeartRateLoLoDB)
           {
              if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.HeartRateLoLoStatus) == null)
              {
```

```
                Alarm alarm = new Alarm(AlarmType.HeartRateLoLoStatus, "LowLow Heart Rate");
                alarms.alarms.Insert(0, alarm);
            }
        }
    }
  }
  else
  {
  }
}

//BodyTemperature
private void ProcessBodyTemperature()
{
  BodyTemperature = commonData.BodyTemperatureValue;
  if (BodyTemperature >= commonData.BodyTemperatureMin && BodyTemperature <=
commonData.BodyTemperatureMax)
  {

    // If BodyTemperatureHiHi enables, it will check how much body temperature is and turn the
alarm on or off, depends on body temperature value
    if (BodyTemperatureHiHiEn == true)
    {
      if (BodyTemperature >= commonData.BodyTemperatureHiHiDB && BodyTemperature <=
commonData.BodyTemperatureHiHiSP)
      {
        if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.BodyTemperatureHiHiStatus) ==
null)
        {
          Alarm alarm = new Alarm(AlarmType.BodyTemperatureHiHiStatus, "HiHi Body
Temperature");
          alarms.alarms.Insert(0, alarm);

        }
      }
    }

    if (BodyTemperatureHiEn == true)
    {
      if (BodyTemperature >= commonData.BodyTemperatureHiDB && BodyTemperature <=
commonData.BodyTemperatureHiSP)
      {

        if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.BodyTemperatureHiStatus) == null)
        {
          Alarm alarm = new Alarm(AlarmType.BodyTemperatureHiStatus, "High Body
Temperature");
          alarms.alarms.Insert(0, alarm);
```

```
                }
            }
        }

        if (BodyTemperatureLoEn == true)
        {
            if (BodyTemperature >= commonData.BodyTemperatureLoSP && BodyTemperature <=
commonData.BodyTemperatureLoDB)
            {
                if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.BodyTemperatureLoStatus) == null)
                {
                    Alarm alarm = new Alarm(AlarmType.BodyTemperatureLoStatus, "Low Body
Temperature");
                    alarms.alarms.Insert(0, alarm);
                }
            }
        }

        if (BodyTemperatureLoLoEn == true)
        {
            if (BodyTemperature >= commonData.BodyTemperatureLoLoSP && BodyTemperature <=
commonData.BodyTemperatureLoLoDB)
            {
                if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.BodyTemperatureLoLoStatus) ==
null)
                {
                    Alarm alarm = new Alarm(AlarmType.BodyTemperatureLoLoStatus, "Low Low Body
Temperature");
                    alarms.alarms.Insert(0, alarm);
                }
            }
        }
    }
    else
    {
        Debug.Log("Body Temperature Something wrong here!");
    }
}

//PrimaryOxygen
private void ProcessPrimaryOxygen()
{
    PrimaryOxygen = commonData.OxygenOneValue;
    if (PrimaryOxygen >= commonData.PrimaryOxygenMin && PrimaryOxygen <=
commonData.PrimaryOxygenMax)
    {

        if (PrimaryOxygenLoEn == true)
```

```
        {
            if (PrimaryOxygen <= commonData.PrimaryOxygenLoDB)
            {
                if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.PrimaryOxygenLoStatus) == null)
                {
                    Alarm alarm = new Alarm(AlarmType.PrimaryOxygenLoStatus, "Low Primary Oxygen");
                    alarms.alarms.Insert(0, alarm);
                }
            }
        }
    }
    else
    {

    }
}

//SecondaryOxygen
private void ProcessSecondaryOxygen()
{
    SecondaryOxygen = commonData.OxygenTwoValue;
    if (SecondaryOxygen >= commonData.SecondaryOxygenMin && SecondaryOxygen <=
commonData.SecondaryOxygenMax)
    {
        if (SecondaryOxygenLoEn == true)
        {
            if (SecondaryOxygen <= commonData.SecondaryOxygenLoDB)
            {
                if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.SecondaryOxygenLoStatus) == null)
                {
                    Alarm alarm = new Alarm(AlarmType.SecondaryOxygenLoStatus, "Low Secondary
Oxygen");
                    alarms.alarms.Insert(0, alarm);
                }
            }
        }

        if (SecondaryOxygenLoLoEn == true)
        {
            if (SecondaryOxygen <= commonData.SecondaryOxygenLoLoDB)
            {
                if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.SecondaryOxygenLoLoStatus) ==
null)
                {
                    Alarm alarm = new Alarm(AlarmType.SecondaryOxygenLoLoStatus, "Low Low Secondary
Oxygen");
                    alarms.alarms.Insert(0, alarm);
                }
```

```
            }
        }
    }
    else
    {

    }
}

//Water
private void ProcessWater()
{
    Water = commonData.WaterValue;
    if (Water >= commonData.WaterMin && Water <= commonData.WaterMax)
    {
        if (WaterLoEn == true)
        {
            if (Water <= commonData.WaterLoDB)
            {
                if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.WaterLoStatus) == null)
                {
                    Alarm alarm = new Alarm(AlarmType.SecondaryOxygenLoStatus, "Low Water");
                    alarms.alarms.Insert(0, alarm);
                }
            }
        }

        if (WaterLoLoEn == true)
        {
            if (Water <= commonData.WaterLoLoDB)
            {
                if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.WaterLoLoStatus) == null)
                {
                    Alarm alarm = new Alarm(AlarmType.WaterLoLoStatus, "Low Low Water");
                    alarms.alarms.Insert(0, alarm);
                }
            }
        }
    }
    else
    {

    }
}

//Battery
private void ProcessBattery()
{
```

```
    Battery = commonData.BatteryValue;
    if (Battery >= commonData.BatteryMin && Battery <= commonData.BatteryMax)
    {
       if (BatteryLoEn == true)
       {
          if (Battery <= commonData.BatteryLoDB)
          {
             if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.BatteryLoStatus) == null)
             {
                Alarm alarm = new Alarm(AlarmType.BatteryLoStatus, "Low Battery");
                alarms.alarms.Insert(0, alarm);
             }
          }
       }

       if (BatteryLoLoEn == true)
       {
          if (Battery <= commonData.BatteryLoLoDB)
          {
             if (alarms.alarms.FirstOrDefault(x => x.type == AlarmType.BatteryLoLoStatus) == null)
             {
                Alarm alarm = new Alarm(AlarmType.BatteryLoLoStatus, "Low Low Battery");
                alarms.alarms.Insert(0, alarm);
             }
          }
       }
    }
    else
    {

    }
 }

 //Voice Commands
 private void ProcessVoiceCommands()
 {
   if(commonData.nextWarning)
    {
       NextAlarmFunction();
       commonData.nextWarning = false;
    }

   if(commonData.previousWarning)
    {
       PreviousAlarmFunction();
       commonData.previousWarning = false;
    }
```

```
      if(commonData.warningAcknowledged)
       {
          AcknowldegeAlarmFunction();
          commonData.warningAcknowledged = false;
       }
    }
}

//Alarms class
public class Alarms {

    public Alarms()
    {
       alarms = new List<Alarm>();
    }

    public List<Alarm> alarms;
    }

public class Alarm
{
    public Alarm(AlarmType type, string message = "Message not set.")
    {
       this.type = type;
       this.message = message;
       timeCreated = DateTime.Now;
       active = true;
    }

    public AlarmType type;
    public string message;
    public bool active;
    public DateTime timeCreated;
    public DateTime timeAcknowledged;

}

public enum AlarmType
{
 //SuitPressure
 SuitPressureHiStatus,
 SuitPressureHiHiStatus,
 SuitPressureLoStatus,
 SuitPressureLoLoStatus,

 //HeartRate
HeartRateHiStatus,
HeartRateHiHiStatus,
```

Business Logic

```
HeartRateLoStatus,
HeartRateLoLoStatus,

//BodyTemperature
BodyTemperatureHiStatus,
BodyTemperatureHiHiStatus,
BodyTemperatureLoStatus,
BodyTemperatureLoLoStatus,

//Primary Oxygen
PrimaryOxygenLoStatus,

//Secondary Oxygen
SecondaryOxygenLoStatus,
SecondaryOxygenLoLoStatus,

//Water
WaterLoStatus,
WaterLoLoStatus,

//Battery
BatteryLoStatus,
BatteryLoLoStatus,
}
```

BL_Main.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BL_Main : MonoBehaviour
{
   [Tooltip("Tick speed in milliseconds")]
   public float tickSpeed = 0.05f;
   private float timeSinceLastTick = 0.0f;


   public BL_Alarming bl_alarming;
   public BL_Scalling bl_scaling;
   public BL_Tasks bl_task;


   // Use this for initialization
   void Start()
   {
      bl_scaling = new BL_Scalling();
      bl_alarming = new BL_Alarming();
      bl_task = new BL_Tasks();
   }

   // Update is called once per frame
   void Update()
   {
      timeSinceLastTick += Time.deltaTime;

      if(timeSinceLastTick > tickSpeed)
      {

         bl_alarming.BLAlarmingFunction();
         timeSinceLastTick -= tickSpeed;
      }
   }
}
```

BL_ResetingAcknow.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BL_ResetingAcknow : MonoBehaviour {

    private CommonData commonData = CommonData.GetInstance();

    // Reseting function for High Suit Pressure only
    public bool BLResetingAcknow(
        bool SuitPressureHiStatus,
        bool SuitPressureHiHiStatus,
        bool SuitPressureLoStatus,
        bool SuitPressureLoLoStatus,
        double SuitPressure,
        bool AlarmAck,
        bool SuitPressureHiAlarm
    )
    {
        // If SuitPressureHiStatus and AlarmAck are both equal TRUE, SuitPressureHiAlarm returns true
        if (SuitPressureHiStatus == true && AlarmAck == true)
        {
            SuitPressureHiAlarm = true;
        }
        else
            SuitPressureHiAlarm = false;

        return SuitPressureHiAlarm;
    }

    // Use this for initialization
    void Start()
    {
        //Setting up initial value

        commonData.SuitPressHiHiSP = 18;
        commonData.SuitPressHiHiDB = 17.8f;

        commonData.SuitPressHiSP = 16;
        commonData.SuitPressHiDB = 15.8f;

        commonData.SuitPressLoSP = 4;
        commonData.SuitPressLoDB = 4.2f;

        commonData.SuitPressLoLoSP = 3;
```

```
        commonData.SuitPressLoLoDB = 3.2f;
    }

    // Update is called once per frame
    void Update()
    {

    }
}
```

BL_Scalling.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BL_Scalling {

    public BL_Scalling()
    {

        //Suit Pressure
        commonData.SuitPressureMax = 5.2f;
        commonData.SuitPressureMin = 3.9f;

        commonData.SuitPressHiHiSP = 5.2f;
        commonData.SuitPressHiHiDB = 5.0f;

        commonData.SuitPressHiSP = 4.99f;
        commonData.SuitPressHiDB = 4.8f;

        commonData.SuitPressLoSP = 4.1f;
        commonData.SuitPressLoDB = 4.3f;

        commonData.SuitPressLoLoSP = 3.9f;
        commonData.SuitPressLoLoDB = 4.09f;

        //Heart Rate
        commonData.HeartRateMax = 80.0f;
        commonData.HeartRateMin = 55.0f;

        commonData.HeartRateHiHiSP = 80.0f;
        commonData.HeartRateHiHiDB = 75.0f;

        commonData.HeartRateHiSP = 74.99f;
        commonData.HeartRateHiDB = 70.0f;

        commonData.HeartRateLoSP = 60.0f;
        commonData.HeartRateLoDB = 64.99f;

        commonData.HeartRateLoLoSP = 55.0f;
        commonData.HeartRateLoLoDB =  59.99f;

        //Body temprature
        commonData.BodyTemperatureMax = 110.0f;
        commonData.BodyTemperatureMin = 90.0f;
```

```
        commonData.BodyTemperatureHiHiSP = 110.0f;
        commonData.BodyTemperatureHiHiDB = 107.0f;

        commonData.BodyTemperatureHiSP = 106.99f;
        commonData.BodyTemperatureHiDB = 104.0f;

        commonData.BodyTemperatureLoSP = 93.0f;
        commonData.BodyTemperatureLoDB = 96.0f;

        commonData.BodyTemperatureLoLoSP = 90.0f;
        commonData.BodyTemperatureLoLoDB = 92.99f;


        //Primary Oxygen
        commonData.PrimaryOxygenMax = 850.0f;
        commonData.PrimaryOxygenMin = 0.85f;

        commonData.PrimaryOxygenLoDB = 255.0f;

        //Secondary Oxygen
        commonData.SecondaryOxygenMax = 3600.0f;
        commonData.SecondaryOxygenMin = 0.036f;

        commonData.SecondaryOxygenHiDB = 3575.0f;

        commonData.SecondaryOxygenLoDB = 1440.0f;

        commonData.SecondaryOxygenLoLoDB = 900.0f;

        //Water
        commonData.WaterMax = 7.0f;
        commonData.WaterMin = 0.07f;

        commonData.WaterLoDB = 2.1f;

        commonData.WaterLoLoDB = 1.4f;

        //Battery
        commonData.BatteryMax = 100.0f;
        commonData.BatteryMin = 1.0f;

        commonData.BatteryLoDB = 30.0f;

        commonData.BatteryLoLoDB = 15.0f;

    }

    private CommonData commonData = CommonData.GetInstance();
```

```
    //Sending out actual oxygen2
    public float actualOxygen2 ()
    {
        return commonData.OxygenTwoValue;
    }


    //------------------------------Suit Pressure-------------------------

    //Sending out actual suitpressure value
    public float actualSuitPressure ()
    {
        return commonData.SuitPressureValue;
    }

    //Scaling from number to percentage
    public float ScalingFunction(
        )
    {
        commonData.SuitPressureValue = BLScalingLimiting(commonData.SuitPressureValue);
        float SuitPressure_normalize = ((commonData.SuitPressureValue-
commonData.SuitPressureMin)*100) / (commonData.SuitPressureMax -
commonData.SuitPressureMin);
        return SuitPressure_normalize;
    }

    //Scaling and limiting function
    public float BLScalingLimiting(
        float DB_SuitPressure
        )
    {
        // This function is about to make sure that SuitPressure is between SuitPressureMax and
SuitPressureMin
        if (DB_SuitPressure < commonData.SuitPressureMin)
            DB_SuitPressure = commonData.SuitPressureMin;
        if (DB_SuitPressure > commonData.SuitPressureMax)
            DB_SuitPressure = commonData.SuitPressureMax;
        return DB_SuitPressure;
    }

    //------------------------------Heart Rate-------------------------

    //Sending out actual Heart Rate
    public float actualHearRate()
    {
        return commonData.HeartRateValue;
```

```
    }

    //Sending out scalling value
    public float scallingHeartRate()
    {
        commonData.HeartRateValue = BLScalingLimitingHeartRate(commonData.HeartRateValue);
        float heartrate_normalize = ((commonData.HeartRateValue -commonData.HeartRateMin)* 100) /
(commonData.HeartRateMax - commonData.HeartRateMin);
        return heartrate_normalize;
    }

    //Scaling and limiting function
    public float BLScalingLimitingHeartRate(
        float DB_HeartRate
        )
    {
        // This function is about to make sure that SuitPressure is between SuitPressureMax and
SuitPressureMin
        if (DB_HeartRate < commonData.HeartRateMin)
            DB_HeartRate = commonData.HeartRateMin;
        if (DB_HeartRate > commonData.HeartRateMax)
            DB_HeartRate = commonData.HeartRateMax;
        return DB_HeartRate;
    }

    //-------------------------------Body Temperature-------------------------

    //Sending out actual Body Temperature
    public float actualBodyTemperature()
    {
        return commonData.BodyTemperatureValue;
    }

    //Sending out scalling value
    public float scallingBodyTemperature()
    {
        commonData.BodyTemperatureValue =
BLScalingLimitingBodyTemperature(commonData.BodyTemperatureValue);
        float bodytemperature_normalize = ((commonData.BodyTemperatureValue -
commonData.BodyTemperatureMin) * 100) / (commonData.BodyTemperatureMax -
commonData.BodyTemperatureMin);
        return bodytemperature_normalize;
    }

    //Scaling and limiting function
    public float BLScalingLimitingBodyTemperature(
        float DB_BodyTemperature
        )
```

```
    {
        // This function is about to make sure that body temperature is between max and min
        if (DB_BodyTemperature < commonData.BodyTemperatureMin)
            DB_BodyTemperature = commonData.BodyTemperatureMin;
        if (DB_BodyTemperature > commonData.BodyTemperatureMax)
            DB_BodyTemperature = commonData.BodyTemperatureMax;
        return DB_BodyTemperature;
    }


    //-------------------------------Primary Oxygen-------------------------

    //Sending out actual Primary Oxygen
    public float actualPrimaryOxygen()
    {
        return commonData.OxygenOneValue;
    }

    //Sending out scalling value
    public float scallingPrimaryOxygen()
    {
        commonData.OxygenOneValue = BLScalingLimitingPrimaryOxygen(commonData.OxygenOneValue);
        float primaryoxygen_normalize = ((commonData.OxygenOneValue-
commonData.PrimaryOxygenMin) * 100) / (commonData.PrimaryOxygenMax -
commonData.PrimaryOxygenMin);
        return primaryoxygen_normalize;
    }

    //Scaling and limiting function
    public float BLScalingLimitingPrimaryOxygen(
        float DB_PrimaryOxygen
        )
    {
        // This function is about to make sure that primary oxygen is between max and min
        if (DB_PrimaryOxygen < commonData.PrimaryOxygenMin)
            DB_PrimaryOxygen = commonData.PrimaryOxygenMin;
        if (DB_PrimaryOxygen > commonData.PrimaryOxygenMax)
            DB_PrimaryOxygen = commonData.PrimaryOxygenMax;
        return DB_PrimaryOxygen;
    }

    //-------------------------------Secondary Oxygen-------------------------

    //Sending out actual Secondary Oxygen
    public float actualSeondaryOxygen()
    {
        return commonData.OxygenTwoValue;
    }
```

```
   //Sending out scalling value
   public float scallingSeondaryOxygen()
   {
      commonData.OxygenTwoValue=
BLScalingLimitingSecondaryOxygen(commonData.OxygenTwoValue);
      float secondaryoxygen_normalize = ((commonData.OxygenTwoValue-
commonData.SecondaryOxygenMin) * 100) / (commonData.SecondaryOxygenMax -
commonData.SecondaryOxygenMin);
      return secondaryoxygen_normalize;
   }

   //Scaling and limiting function
   public float BLScalingLimitingSecondaryOxygen(
      float DB_SecondaryOxygen
      )
   {
      // This function is about to make sure that secondary oxygen is between max and min
      if (DB_SecondaryOxygen < commonData.SecondaryOxygenMin)
         DB_SecondaryOxygen = commonData.SecondaryOxygenMin;
      if (DB_SecondaryOxygen > commonData.SecondaryOxygenMax)
         DB_SecondaryOxygen = commonData.SecondaryOxygenMax;
      return DB_SecondaryOxygen;
   }

   //------------------------------Water-------------------------

   //Sending out actual water
   public float actualWater()
   {
      return commonData.WaterValue;
   }

   //Sending out scalling value
   public float scallingWater()
   {
      commonData.WaterValue = BLScalingLimitingWater(commonData.WaterValue);
      float water_normalize = ((commonData.WaterValue-commonData.WaterMin) * 100) /
(commonData.WaterMax - commonData.WaterMin);
      return water_normalize;
   }

   //Scaling and limiting function
   public float BLScalingLimitingWater(
      float DB_Water
      )
   {
      // This function is about to make sure that water is between max and min
```

```
        if (DB_Water < commonData.WaterMin)
            DB_Water = commonData.WaterMin;
        if (DB_Water > commonData.WaterMax)
            DB_Water = commonData.WaterMax;
        return DB_Water;
    }

    //------------------------------Battery-------------------------

    //Sending out actual Battery
    public float actualBattery()
    {
        return commonData.BatteryValue;
    }

    //Sending out scalling value
    public float scallingBattery()
    {
        commonData.BatteryValue = BLScalingLimitingBattery(commonData.BatteryValue);
        float battery_normalize = ((commonData.BatteryValue-commonData.BatteryMin) * 100) /
(commonData.BatteryMax - commonData.BatteryMin);
        return battery_normalize;
    }

    //Scaling and limiting function
    public float BLScalingLimitingBattery(
        float DB_Battery
        )
    {
        // This function is about to make sure that Battery is between max and min
        if (DB_Battery <= commonData.BatteryMin)
            DB_Battery = commonData.BatteryMin;
        if (DB_Battery >= commonData.BatteryMax)
            DB_Battery = commonData.BatteryMax;
        return DB_Battery;
    }
}
```

BL_Tasks.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.Net;

public class BL_Tasks {

   private CommonData commonData = CommonData.GetInstance();

   private Procedure procedure;

   public UITask previousTask;
   public UITask currentTask;
   public UITask nextTask;

   public void AwakeFunction()
   {
      previousTask = new UITask();
      currentTask = new UITask();
      nextTask = new UITask();

      procedure = new Procedure();

      LoadTaskOne();

       //LoadProcedure2();
   }

   public Procedure GetProcedure()
   {
      return procedure;
   }


   public BL_Tasks ()
   {
      AwakeFunction();
      //Example for databse - map json data to C# object
   }

   public void nextFunction(
      bool nextStep
      )
   {
```

```
    bool greenFlag = commonData.getGreen();
    bool notGreenFlag = commonData.getNotGreen();

    if (nextStep == true)
    {

        int index = commonData.currentTask.stepNumber - 1;


        if (index == procedure.tasks.Count - 1)
        {
            //do nothing we are at the end
            commonData.nextTask = new UITask();
            commonData.currentTask = new UITask();
            commonData.currentTask.text = "Task Complete!";
            commonData.currentTask.stepNumber = procedure.tasks.Count;
            commonData.previousTask = new UITask();
        }
        else if (index == procedure.tasks.Count - 2)
        {
            //Go back one step only
            commonData.nextTask = new UITask();
            commonData.currentTask = procedure.tasks[index + 1];
            commonData.previousTask = procedure.tasks[index];
        }
        else
        {

            commonData.nextTask = procedure.tasks[index + 2];
            commonData.currentTask = procedure.tasks[index + 1];
            commonData.previousTask = procedure.tasks[index];

        }



        //whatever
        previousTask = commonData.previousTask;
        currentTask = commonData.currentTask;
        nextTask = commonData.nextTask;
    }
}

public void nextFunction2(
    bool nextStep
    )
{
```

```
bool greenFlag = commonData.getGreen();
bool notGreenFlag = commonData.getNotGreen();

if(nextStep == true)
{

   int index = commonData.currentTask.stepNumber - 1;


   if (index == procedure.tasks.Count-1)
   {
      //do nothing we are at the end
      commonData.nextTask = new UITask();
      commonData.currentTask = new UITask();
      commonData.currentTask.text = "Task Complete!";
      commonData.currentTask.stepNumber = procedure.tasks.Count;
      commonData.previousTask = new UITask();
      commonData.setGreen(false);
      commonData.setNotGreen(false);
   }
   else if (index == procedure.tasks.Count -2)
   {
      //Go back one step only
      commonData.nextTask = new UITask();
      commonData.currentTask = procedure.tasks[index +1];
      commonData.previousTask = procedure.tasks[index];
   }
   else
   {
      if (index == 29)
      {
         commonData.nextTask = new UITask();
         commonData.currentTask = procedure.tasks[30];
         commonData.previousTask = procedure.tasks[29];
      } else
      {
         if (greenFlag == true)
         {
            if ((index == 30) || (index == 29))
            {
               commonData.nextTask = new UITask();
               commonData.currentTask = procedure.tasks[31];
               commonData.previousTask = procedure.tasks[30];
               commonData.currentTask.stepNumber = 32;
            }

            if (index == 31)
            {
```

```
            commonData.nextTask = new UITask();
            commonData.currentTask = new UITask();
            commonData.currentTask.text = "Task Complete!";
            commonData.currentTask.stepNumber = 41;
            commonData.previousTask = new UITask();
        }


    }
    else
    {

        if (notGreenFlag == true)
        {

            if (index == 30)
            {
                commonData.nextTask = procedure.tasks[34];
                commonData.currentTask = procedure.tasks[32];
                commonData.previousTask = new UITask();
                commonData.currentTask.stepNumber = 33;
            }

            if (index >= 32)
            {
                commonData.nextTask = procedure.tasks[index + 2];
                commonData.currentTask = procedure.tasks[index + 1];
                commonData.previousTask = procedure.tasks[index];
            }
        }
        else
        {
            if (index < 30)
            {
                commonData.nextTask = procedure.tasks[index + 2];
                commonData.currentTask = procedure.tasks[index + 1];
                commonData.previousTask = procedure.tasks[index];
            }
            else
            {
                commonData.nextTask = new UITask();
                commonData.currentTask = procedure.tasks[30];
                commonData.previousTask = procedure.tasks[29];
            }

        }

    }
```

```
            }
        }


        //whatever
        previousTask = commonData.previousTask;
        currentTask = commonData.currentTask;
        nextTask = commonData.nextTask;
    }
}


public void previousFunction(
    bool previousStep
    )
{
    if (previousStep == true)
    {
        int index = commonData.currentTask.stepNumber-1;


        if(index == 0)
        {
            //do nothing we are at the beginning
        }
        else if (index == 1)
        {
            //Go back one step only
            commonData.nextTask = procedure.tasks[index];
            commonData.currentTask = procedure.tasks[index - 1];
            commonData.previousTask = new UITask();
        }
        else
        {
            commonData.nextTask = procedure.tasks[index];
            commonData.currentTask = procedure.tasks[index - 1];
            commonData.previousTask = procedure.tasks[index - 2];
        }

        previousTask = commonData.previousTask;
        currentTask = commonData.currentTask;
        nextTask = commonData.nextTask;

    }
}

public void previousFunction2(
    bool previousStep
```

```
    )
  {
    if (previousStep == true)
    {
      int index = commonData.currentTask.stepNumber - 1;


      if (index == 0)
      {
        //do nothing we are at the beginning
      }
      else if (index == 1)
      {
        //Go back one step only
        commonData.nextTask = procedure.tasks[index];
        commonData.currentTask = procedure.tasks[index - 1];
        commonData.previousTask = new UITask();
      }
      else
      {
        commonData.nextTask = procedure.tasks[index];
        commonData.currentTask = procedure.tasks[index - 1];
        commonData.previousTask = procedure.tasks[index - 2];
      }

      previousTask = commonData.previousTask;
      currentTask = commonData.currentTask;
      nextTask = commonData.nextTask;

    }
  }

  public void LoadTaskOne()
  {
    procedure = new Procedure();
    UITask task1 = new UITask();
    task1.stepNumber = 1;
    task1.text = "On the RIGHT side of the EVA Kit, locate the PANEL ACCESS KEY.";
    task1.caution = "The keys are on a tension-spring cable";
    task1.warning = "";
    task1.holograms = "";
    task1.images = "";
    UITask task2 = new UITask();
    task2.stepNumber = 2;
    task2.text = "Use the PANEL ACCESS KEY to unlock the PANEL ACCESS DOOR LOCKS.";
    task2.caution = "";
    task2.warning = "";
    task2.holograms = "";
```

```
task2.images = "2";
UITask task3 = new UITask();
task3.stepNumber = 3;
task3.text = "Carefully return keys to the side of the EVA kit.";
task3.caution = "";
task3.warning = "";
task3.holograms = "";
task3.images = "";
UITask task4 = new UITask();
task4.stepNumber = 4;
task4.text = "Insert your fingers in the CENTER OPENING and secure the PANEL ACCESS DOOR in an
OPEN position.";
task4.caution = "";
task4.warning = "Door can accidentally close ";
task4.holograms = "";
task4.images = "4";
UITask task5 = new UITask();
task5.stepNumber = 5;
task5.text = "On your belt, use the BLUE CARABINEER to securely tether to the TETHER CABLE inside
the STORAGE.";
task5.caution = "Notice the TETHER CABLE is adjustable";
task5.warning = "";
task5.holograms = "";
task5.images = "";
UITask task6 = new UITask();
task6.stepNumber = 6;
task6.text = "Locate the E-stop button, insert keys into it and turn it to off ";
task6.warning = "";
task6.holograms = "E-Stop";
task6.images = "6";
UITask task7 = new UITask();
task7.stepNumber = 7;
task7.text = "Locate the FUSIBLE DISCONNECT box";
task7.warning = "";
task7.holograms = "FusibleDisconnectBox";
task7.images = "";
UITask task8 = new UITask();
task8.stepNumber = 8;
task8.text = "Tether the BLUE CARABINEER to the TETHER CABLE";
task8.warning = "";
task8.holograms = "";
task8.images = "";
UITask task9 = new UITask();
task9.stepNumber = 9;
task9.text = "Remove the BLUE CARABINEER from the FUSIBLE DISCONNECT BOX";
task9.warning = "";
task9.holograms = "FusibleDisconnectBox";
task9.images = "9";
```

```
UITask task10 = new UITask();
task10.stepNumber = 10;
task10.text = " Transfer the BLUE CARABINEER to STORAGE";
task10.warning = "";
task10.holograms = "";
task10.images = "";
UITask task11 = new UITask();
task11.stepNumber = 11;
task11.text = " Open the FUSIBLE DISCONNECT box and secure the lid in the open position";
task11.caution = " Pull the locking tab towards STORAGE with the thumb ";
task11.warning = "";
task11.holograms = "FusibleDisconnectBox";
task11.images = "11";
UITask task12 = new UITask();
task12.stepNumber = 12;
task12.text = "Locate the BLACK DISCONNECT";
task12.warning = "";
task12.holograms = "FusibleDisconnectBox";
task12.images = "";
UITask task13 = new UITask();
task13.stepNumber = 13;
task13.text = " Tether it to the TETHER CABLE";
task13.warning = "";
task13.holograms = "";
task13.images = "";
UITask task14 = new UITask();
task14.stepNumber = 14;
task14.text = "Remove the DISCONNECT";
task14.caution = " Pull up with the index and middle fingers while pushing down on the FUSE
ACCESS PANEL with the thumb ";
task14.warning = "";
task14.holograms = "FusibleDisconnectBox";
task14.images = "14";
UITask task15 = new UITask();
task15.stepNumber = 15;
task15.text = "Place the DISCONNECT in the STORAGE";
task15.warning = "";
task15.holograms = "";
task15.images = "";
UITask task16 = new UITask();
task16.stepNumber = 16;
task16.text = "Tether the FUSE ACCESS PANEL to the TETHER CABLE";
task16.warning = "";
task16.holograms = "";
task16.images = "";
UITask task17 = new UITask();
task17.stepNumber = 17;
task17.text = "Remove the FUSE ACCESS PANEL by pulling straght up";
```

```
task17.warning = "";
task17.holograms = "";
task17.images = "17";
UITask task18 = new UITask();
task18.stepNumber = 18;
task18.text = "Place the FUSE ACCESS PANEL into STORAGE";
task18.warning = "";
task18.holograms = "";
task18.images = "";
UITask task19 = new UITask();
task19.stepNumber = 19;
task19.text = " Tether the ALARM FUSE to the TETHER CABLE";
task19.warning = "";
task19.holograms = "";
task19.images = "";
UITask task20 = new UITask();
task20.stepNumber = 20;
task20.text = "In Storage, locate the BLUE FUSE PULLER";
task20.warning = "";
task20.holograms = "";
task20.images = "";
UITask task21 = new UITask();
task21.stepNumber = 21;
task21.text = "Use the BLUE FUSE PULLER to remove ONLY the ALARM FUSE";
task21.caution = " Rock the ALARM FUSE with the FUSE PULLER when pulling up ";
task21.holograms = "";
task21.images = "21";
UITask task22 = new UITask();
task22.stepNumber = 22;
task22.text = "Return the ALARM FUSE and the FUSE PULLER to STORAGE";
task22.warning = "";
task22.holograms = "";
task22.images = "";
UITask task23 = new UITask();
task23.stepNumber = 23;
task23.text = "In STORAGE, locate the FUSE ACCESS PANEL and reinstall it into the FUSIBLE
DISCONNECT box.";
task23.warning = "";
task23.holograms = "FusibleDisconnectBox";
task23.images = "23";
UITask task24 = new UITask();
task24.stepNumber = 24;
task24.text = "Remove the FUSE ACCESS PANEL tether from the TETHER CABLE and stow inside";
task24.warning = " All tethers are under  spring tension and can retract quickly";
task24.caution = "";
task24.holograms = "";
task24.images = "";
UITask task25 = new UITask();
```

```
task25.stepNumber = 25;
task25.text = "In STORAGE, locate the DISCONNECT and reinstall it into the FUSIBLE DISCONNECT box";
task25.caution = "The DISCONNECT must read ON in the upper right corner to restore Conductivity";
task25.warning = "";
task25.holograms = "FusibleDisconnectBox";
task25.images = "25";
UITask task26 = new UITask();
task26.stepNumber = 26;
task26.text = " Remove the DISCONNECT tether from the TETHER CABLE";
task26.caution = "";
task26.warning = "All tethers are under spring tension and can retract quickly";
task26.holograms = "";
task26.images = "";
UITask task27 = new UITask();
task27.stepNumber = 27;
task27.text = " Close the FUSIBLE DISCONNECT box cover";
task27.caution = "";
task27.warning = "";
task27.holograms = "FusibleDisconnectBox";
task27.images = "";
UITask task28 = new UITask();
task28.stepNumber = 28;
task28.text = "In STORAGE, use the BLUE CARABINEER to clip and lock the FUSIBLE DISCONNECT box cover";
task28.caution = "";
task28.warning = "";
task28.holograms = "FusibleDisconnectBox";
task28.images = "28";
UITask task29 = new UITask();
task29.stepNumber = 29;
task29.text = "Remove the BLUE CARABINEER’s tether from the TETHER CABLE";
task29.caution = "";
task29.warning = "All tethers are under spring tension and can retract quickly";
task29.holograms = "";
task29.images = "";



procedure.tasks.Add(task1);
procedure.tasks.Add(task2);
procedure.tasks.Add(task3);
procedure.tasks.Add(task4);
procedure.tasks.Add(task5);
procedure.tasks.Add(task6);
procedure.tasks.Add(task7);
procedure.tasks.Add(task8);
```

```
        procedure.tasks.Add(task9);
        procedure.tasks.Add(task10);
        procedure.tasks.Add(task11);
        procedure.tasks.Add(task12);
        procedure.tasks.Add(task13);
        procedure.tasks.Add(task14);
        procedure.tasks.Add(task15);
        procedure.tasks.Add(task16);
        procedure.tasks.Add(task17);
        procedure.tasks.Add(task18);
        procedure.tasks.Add(task19);
        procedure.tasks.Add(task20);
        procedure.tasks.Add(task21);
        procedure.tasks.Add(task22);
        procedure.tasks.Add(task23);
        procedure.tasks.Add(task24);
        procedure.tasks.Add(task25);
        procedure.tasks.Add(task26);
        procedure.tasks.Add(task27);
        procedure.tasks.Add(task28);

        commonData.previousTask = new UITask();
        commonData.currentTask = task1;
        commonData.nextTask = task2;

        previousTask = new UITask();
        currentTask = task1;
        nextTask = task2;

    }

    public void LoadTaskTwo()
    {
        procedure = new Procedure();
        UITask task1 = new UITask();
        task1.stepNumber = 1;
        task1.text = "Locate the Aux. Power Input";
        task1.caution = "";
        task1.warning = "";
        task1.holograms = "Aux_PowerInput";
        task1.images = "";
        UITask task2 = new UITask();
        task2.stepNumber = 2;
        task2.text = "Locate BATTERY PACK and tether to TETHER CABLE";
        task2.caution = "Depress the red and black plastic hammers on the side of the AUX. POWER INPUT
and pull the leads straight up";
        task2.warning = "";
        task2.holograms = "";
```

```
task2.images = "";
UITask task3 = new UITask();
task3.stepNumber = 3;
task3.text = "Undo the BATTERY PACK LEADS from the AUX. POWER INPUT";
task3.caution = "";
task3.warning = "";
task3.holograms = "Aux_PowerInput";
task3.images = "2.3";
UITask task4 = new UITask();
task4.stepNumber = 4;
task4.text = "Remove BATTERY PACK from AUX. POWER INPUT";
task4.caution = "";
task4.warning = "";
task4.holograms = "Aux_PowerInput";
task4.images = "";
UITask task5 = new UITask();
task5.stepNumber = 5;
task5.text = "Locate the ON/OFF switch on the back of the BATTERY PACK and switch it to the OFF
position.";
task5.caution = "";
task5.warning = "";
task5.holograms = "";
task5.images = "2.5";
UITask task6 = new UITask();
task6.stepNumber = 6;
task6.text = "Place the BATTERY PACK into STORAGE";
task6.caution = "";
task6.warning = "";
task6.holograms = "";
task6.images = "";
UITask task7 = new UITask();
task7.stepNumber = 7;
task7.text = "In STORAGE, find the replacement BATTERY PACK";
task7.caution = "";
task7.warning = "";
task7.holograms = "";
task7.images = "";
UITask task8 = new UITask();
task8.stepNumber = 8;
task8.text = "Locate the ON/OFF switch on the back of the BATTERY PACK and switch it to the ON
position";
task8.caution = "";
task8.warning = "";
task8.holograms = "";
task8.images = "";
UITask task9 = new UITask();
task9.stepNumber = 9;
task9.text = "Attach the replacement BATTERY PACK onto the AUX. POWER INPUT by the Velcro";
```

```
task9.caution = "";
task9.warning = "";
task9.holograms = "Aux_PowerInput";
task9.images = "";
UITask task10 = new UITask();
task10.stepNumber = 10;
task10.text = "Insert the BATTERY PACK leads back into same colored ports";
task10.caution = "Depress the red and black plastic hammers on the side of the AUX. POWER INPUT and push leads straight into their ports";
task10.warning = "";
task10.holograms = "";
task10.images = "";
UITask task11 = new UITask();
task11.stepNumber = 11;
task11.text = "Conduct a GENTLE PULL TEST on the wires";
task11.caution = "";
task11.warning = "";
task11.holograms = "";
task11.images = "";
UITask task12 = new UITask();
task12.stepNumber = 12;
task12.text = "Remove the BATTERY PACK tether from the TETHER CABLE";
task12.caution = "";
task12.warning = "All tethers are under spring tension and can retract quickly";
task12.holograms = "";
task12.images = "";
UITask task13 = new UITask();
task13.stepNumber = 13;
task13.text = "In STORAGE, locate the GRAY 220 Volt PLUG";
task13.caution = "";
task13.warning = "";
task13.holograms = "";
task13.images = "";
UITask task14 = new UITask();
task14.stepNumber = 14;
task14.text = "Install it into the POWER OUT";
task14.caution = "Outlet and plug mate are stiff, ensure the full engagement of the plug into the outlet";
task14.warning = "";
task14.holograms = "PowerOutputBox";
task14.images = "2.12";
UITask task15 = new UITask();
task15.stepNumber = 15;
task15.text = "Locate the metal BUSS BAR and verify there are BLACK, GREEN & WHITE BUSSES, each with 2 openings";
task15.caution = "";
task15.warning = "";
task15.holograms = "BusBars";
```

```
task15.images = "";
UITask task16 = new UITask();
task16.stepNumber = 16;
task16.text = "Insert the WHITE 220 VOLT LEAD into the LEFT WHITE BUSS opening and GENTLY
TIGHTEN the thumbscrew";
task16.caution = "DO NOT over tighten the thumbscrew";
task16.warning = "";
task16.holograms = "";
task16.images = "2.16";
UITask task17 = new UITask();
task17.stepNumber = 17;
task17.text = "Insert the GREEN 220 VOLT LEAD into the LEFT GREEN BUS opening";
task17.caution = "DO NOT over tighten the thumbscrew";
task17.warning = "";
task17.holograms = "";
task17.images = "2.17";
UITask task18 = new UITask();
task18.stepNumber = 18;
task18.text = "Insert the BLACK 220 VOLT LEAD into the LEFT BLACK BUS opening";
task18.caution = "DO NOT over tighten the thumbscrew";
task18.warning = "";
task18.holograms = "";
task18.images = "2.18";
UITask task19 = new UITask();
task19.stepNumber = 19;
task19.text = "Make sure the METAL LEADS are not sticking out the BACK of the BUSS BAR";
task19.caution = "";
task19.warning = "";
task19.holograms = "BusBars";
task19.images = "";
UITask task20 = new UITask();
task20.stepNumber = 20;
task20.text = "Conduct a Gentle PULL TEST on each cable";
task20.caution = "";
task20.warning = "";
task20.holograms = "";
task20.images = "";
UITask task21 = new UITask();
task21.stepNumber = 21;
task21.text = "In STORAGE, locate the 110 VOLT PLUG";
task21.caution = "";
task21.warning = "";
task21.holograms = "";
task21.images = "2.21";
UITask task22 = new UITask();
task22.stepNumber = 22;
task22.text = "Install it into POWER IN";
```

```
task22.caution = "Lift cover with one hand while installing PLUG into the outlet with the other. The
lid is spring-loaded";
task22.warning = "";
task22.holograms = "Aux_PowerInput";
task22.images = "2.21";
UITask task23 = new UITask();
task23.stepNumber = 23;
task23.text = "Insert the WHITE 110 VOLT PLUG LEAD into the RIGHT WHITE BUS opening";
task23.caution = "DO NOT over tighten the thumbscrew";
task23.warning = "";
task23.holograms = "";
task23.images = "2.22";
UITask task24 = new UITask();
task24.stepNumber = 24;
task24.text = "Insert the GREEN 110 VOLT PLUG LEAD into the RIGHT GREEN BUS opening";
task24.caution = "DO NOT over tighten the thumbscrew";
task24.warning = "";
task24.holograms = "";
task24.images = "2.23";
UITask task25 = new UITask();
task25.stepNumber = 25;
task25.text = "Insert the BLACK 110 VOLT PLUG LEAD into the RIGHT BLACK BUS opening";
task25.caution = "DO NOT over tighten the thumbscrew";
task25.warning = "";
task25.holograms = "";
task25.images = "2.24";
UITask task26 = new UITask();
task26.stepNumber = 26;
task26.text = "Conduct a Gentle PULL TEST on each cable";
task26.caution = "";
task26.warning = "";
task26.holograms = "";
task26.images = "";
UITask task27 = new UITask();
task27.stepNumber = 27;
task27.text = "In STORAGE, locate the E-STOP KEY";
task27.caution = "";
task27.warning = "";
task27.holograms = "";
task27.images = "2.25";
UITask task28 = new UITask();
task28.stepNumber = 28;
task28.text = "Insert the KEY into the E-STOP and TURN it to ON position";
task28.caution = "";
task28.warning = "";
task28.holograms = "E-Stop";
task28.images = "2.26";
UITask task29 = new UITask();
```

```
task29.stepNumber = 29;
task29.text = "Remove the KEY and place it in STORAGE";
task29.caution = "";
task29.warning = "";
task29.holograms = "";
task29.images = "";
UITask task30 = new UITask();
task30.stepNumber = 30;
task30.text = "Locate the AUX. POWER SWITCH on the POWER IN box and switch it to the “ON” position";
task30.caution = "";
task30.warning = "";
task30.holograms = "Aux_PowerInput";
task30.images = "2.28";
UITask task31 = new UITask();
task31.stepNumber = 31;
task31.text = "Can you please say whether the SYSTEM GO indicator light is GREEN or NOT GREEN?";
task31.caution = "";
task31.warning = "";
task31.holograms = "";
task31.images = "2.last";
UITask task32 = new UITask();
task32.stepNumber = 32;
task32.text = "EVA 1, this is Houston Mission Control. Congratulations. PHALCON is reporting that they are reading a successful power restoration on their console. You are a go to untether from the TETHER CABLE and return to space station. Mission Control out";
task32.caution = "";
task32.warning = "";
task32.holograms = "";
task32.images = "";
UITask task33 = new UITask();
task33.stepNumber = 33;
task33.text = "EVA 1, this is Houston Mission Control. PHALCON confirms and is not able to report a successful power restoration on their console. EVA-1, please be advised that we have prepared some trouble-shooting steps for you to conduct on a future spacewalk";
task33.caution = "";
task33.warning = "";
task33.holograms = "";
task33.images = "";
UITask task34 = new UITask();
task34.stepNumber = 34;
task34.text = "There might be problem with power going into the circuit, Please go through the following check points";
task34.caution = "";
task34.warning = "";
task34.holograms = "";
task34.images = "";
```

```
UITask task35 = new UITask();
task35.stepNumber = 35;
task35.text = "BATTERY PACK is switched OFF";
task35.caution = "";
task35.warning = "";
task35.holograms = "";
task35.images = "";
UITask task36 = new UITask();
task36.stepNumber = 36;
task36.text = "BATTERY PACK leads are not inserted into their ports";
task36.caution = "";
task36.warning = "";
task36.holograms = "";
task36.images = "";
UITask task37 = new UITask();
task37.stepNumber = 37;
task37.text = "220V PLUG or leads where not installed correctly";
task37.caution = "";
task37.warning = "";
task37.holograms = "";
task37.images = "";
UITask task38 = new UITask();
task38.stepNumber = 38;
task38.text = "110V PLUG or leads where not installed correctly";
task38.caution = "";
task38.warning = "";
task38.holograms = "";
task38.images = "";
UITask task39 = new UITask();
task39.stepNumber = 39;
task39.text = "AUX. POWER SWITCH is in the OFF position";
task39.caution = "";
task39.warning = "";
task39.holograms = "";
task39.images = "";
UITask task40 = new UITask();
task40.stepNumber = 40;
task40.text = "E-STOP is still engaged";
task40.caution = "";
task40.warning = "";
task40.holograms = "";
task40.images = "";
UITask task41 = new UITask();
task41.stepNumber = 41;
task41.text = "Batteries in the BATTERY PACK are depleted";
task41.caution = "";
task41.warning = "";
task41.holograms = "";
```

```
task41.images = "";

procedure.tasks.Add(task1);
procedure.tasks.Add(task2);
procedure.tasks.Add(task3);
procedure.tasks.Add(task4);
procedure.tasks.Add(task5);
procedure.tasks.Add(task6);
procedure.tasks.Add(task7);
procedure.tasks.Add(task8);
procedure.tasks.Add(task9);
procedure.tasks.Add(task10);
procedure.tasks.Add(task11);
procedure.tasks.Add(task12);
procedure.tasks.Add(task13);
procedure.tasks.Add(task14);
procedure.tasks.Add(task15);
procedure.tasks.Add(task16);
procedure.tasks.Add(task17);
procedure.tasks.Add(task18);
procedure.tasks.Add(task19);
procedure.tasks.Add(task20);
procedure.tasks.Add(task21);
procedure.tasks.Add(task22);
procedure.tasks.Add(task23);
procedure.tasks.Add(task24);
procedure.tasks.Add(task25);
procedure.tasks.Add(task26);
procedure.tasks.Add(task27);
procedure.tasks.Add(task28);
procedure.tasks.Add(task29);
procedure.tasks.Add(task30);
procedure.tasks.Add(task31);
procedure.tasks.Add(task32);
procedure.tasks.Add(task33);
procedure.tasks.Add(task34);
procedure.tasks.Add(task35);
procedure.tasks.Add(task36);
procedure.tasks.Add(task37);
procedure.tasks.Add(task38);
procedure.tasks.Add(task39);
procedure.tasks.Add(task40);
procedure.tasks.Add(task41);

commonData.previousTask = new UITask();
commonData.currentTask = task1;
```

```
        commonData.nextTask = task2;

        previousTask = new UITask();
        currentTask = task1;
        nextTask = task2;
    }
}

public class UITask
{
    public int stepNumber;
    public string text = "";
    public string caution = "";
    public string warning = "";
    public string holograms = "";
    public string images = "";
}


public class Procedure
{
    public List<UITask> tasks;

    public Procedure()
    {
        tasks = new List<UITask>();
    }
}
```

BTemperature.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;

public class BTemperature : MonoBehaviour {

   // Use this for initialization

   public float radius = 1.0f;
   public Image objBT;
   public float bodyTemperature;
   public ColorCode cc;
   public GameObject bl_main;
   public CommonData commonData;
   void Start () {
      commonData = CommonData.GetInstance();
   }

            // Update is called once per frame
            void Update () {

      bodyTemperature = commonData.BodyTemperatureValue;
     radius = bl_main.GetComponent<BL_Main>().bl_scaling.scallingBodyTemperature();



      if (bodyTemperature >= commonData.BodyTemperatureHiHiDB && bodyTemperature <=
commonData.BodyTemperatureHiHiSP)
      {
         // Debug.Log("HH");
         cc.LLCol();
         objBT.color = cc.LLColor;
      }
      else if (bodyTemperature >= commonData.BodyTemperatureHiDB && bodyTemperature <=
commonData.BodyTemperatureHiSP)
      {
         // Debug.Log("H");
         cc.LCol();
         objBT.color = cc.LColor;
      }
      else if (bodyTemperature <= commonData.BodyTemperatureHiDB && bodyTemperature >=
commonData.BodyTemperatureLoDB)
      {
```

```
            //  Debug.Log("Ideal");
            cc.HCol();
            objBT.color = cc.HColor;
        }
        else if (bodyTemperature >= commonData.BodyTemperatureLoSP && bodyTemperature <=
commonData.BodyTemperatureLoDB)
        {
            //   Debug.Log("L");
            cc.LCol();
            objBT.color = cc.LColor;
        }
        else if (bodyTemperature >= commonData.BodyTemperatureLoLoSP && bodyTemperature <=
commonData.BodyTemperatureLoLoDB)
        {
            // Debug.Log("LL");
            cc.LLCol();
            objBT.color = cc.LLColor;
        }


            objBT.transform.localScale = new Vector3(radius/100, radius/100, 1.0f);

    }
}
```

ColorCode.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ColorCode : MonoBehaviour {

    // Use this for initialization
    public Color HHColor;
    public Color HColor;
    public Color LColor;
    public Color LLColor;

    private void Start()
    {
        HHColor = new Color(0.39f, 0.58f, 0.93f, 1.0f);
        HColor = new Color(0.49f, 0.98f, 0.6f, 1.0f);
        LColor = new Color(1.0f, 1.0f, 0.0f, 1.0f);
        LLColor = new Color(1.0f, 0.39f, 0.28f, 1.0f);
    }


    public void HHCol ()
    {
        HHColor = new Color(0.39f, 0.58f, 0.93f, 1.0f);
    }

    public void HCol()
    {
        HColor = new Color(0.49f, 0.98f, 0.6f, 1.0f);
    }


    public void LCol()
    {
        LColor = new Color(1.0f, 1.0f, 0.0f, 1.0f);
    }

    public void LLCol()
    {
        LLColor = new Color(1.0f, 0.39f, 0.28f, 1.0f);
    }
}
```

Health UI

CommonData.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CommonData {

   private static CommonData _commonData;

   private CommonData() { }

   public static CommonData GetInstance()
   {
      if(_commonData == null)
      {
         _commonData = new CommonData();
      }

      return _commonData;
   }

   //The common data  (flags and alarm definitions will go here)
   private bool editMode = false;
         public bool EditMode { get { return editMode; } set { editMode = value; } }

   public UITask previousTask;
   public UITask currentTask;
   public UITask nextTask;

   public UITask GetPreviousTask { get { return previousTask; } }
   public UITask GetCurrentTask { get { return currentTask; } }
   public UITask GetNextTask { get { return nextTask; } }


   public bool viewPieDetails = false;
   public bool closePieDetails = false;
   public bool viewBattery = false;
   public bool viewPrimaryOTwo = false;
   public bool viewSecondaryOTwo = false;
   public bool viewBodyTemperature = false;
   public bool viewPressure = false;
   public bool viewHTwoO = false;
   public bool viewHeartRate = false;
   public bool closeDetailWindow = false;
   public bool pinDetailWindow = false;
   public bool closePinnedWindows = false;
```

```
public bool viewErrors = false;
public bool closeErrors = false;
public bool nextError = false;
public bool previousError = false;
public bool viewWarnings = false;
public bool closeWarnings = false;
public bool nextWarning = false;
public bool previousWarning = false;

public bool startProcedure = false;
public bool nextStep = false;
public bool previousStep = false;
public bool showStepImage = false;
public bool hideStepImage = false;
public bool pinTaskList = false;
public bool closeTaskList = false;
public bool loadTaskOne = false;
public bool loadTaskTwo = false;
internal bool warningAcknowledged = false;
public static bool notgreen = false;
public static bool green = false;


//Health UI Values
public float OxygenOneValue { get; set; }
public float OxygenTwoValue { get; set; }
public float HeartRateValue { get; set; }
public float BodyTemperatureValue { get; set; }
public float SuitPressureValue { get; set; }
public float BatteryValue { get; set; }
public float WaterValue { get; set; }

//SuitPressure
public float SuitPressureMax { get; set; }
public float SuitPressureMin { get; set; }

public float SuitPressHiHiSP { get; set; }
public float SuitPressHiHiDB { get; set; }

public float SuitPressHiSP { get; set; }
public float SuitPressHiDB { get; set; }

public float SuitPressLoSP { get; set; }
public float SuitPressLoDB { get; set; }

public float SuitPressLoLoSP { get; set; }
public float SuitPressLoLoDB { get; set; }
```

```
// Heartrate
public float HeartRateMax { get; set; }
public float HeartRateMin { get; set; }

public float HeartRateHiHiSP { get; set; }
public float HeartRateHiHiDB { get; set; }

public float HeartRateHiSP { get; set; }
public float HeartRateHiDB { get; set; }

public float HeartRateLoSP { get; set; }
public float HeartRateLoDB { get; set; }

public float HeartRateLoLoSP { get; set; }
public float HeartRateLoLoDB { get; set; }

//Bodytemperature
public float BodyTemperatureMax { get; set; }
public float BodyTemperatureMin { get; set; }

public float BodyTemperatureHiHiSP { get; set; }
public float BodyTemperatureHiHiDB { get; set; }

public float BodyTemperatureHiSP { get; set; }
public float BodyTemperatureHiDB { get; set; }

public float BodyTemperatureLoSP { get; set; }
public float BodyTemperatureLoDB { get; set; }

public float BodyTemperatureLoLoSP { get; set; }
public float BodyTemperatureLoLoDB { get; set; }


//PrimaryOxygen
public float PrimaryOxygenMax { get; set; }
public float PrimaryOxygenMin { get; set; }
public float PrimaryOxygenLoDB { get; set; }


//SecondaryOxygen
public float SecondaryOxygenMax { get; set; }
public float SecondaryOxygenMin { get; set; }
public float SecondaryOxygenLoDB { get; set; }
public float SecondaryOxygenLoLoDB { get; set; }


//Water
public float WaterMax { get; set; }
```

```
    public float WaterMin { get; set; }

    public float WaterLoDB { get; set; }

    public float WaterLoLoDB { get; set; }

    //Battery
    public float BatteryMax { get; set; }
    public float BatteryMin { get; set; }

    public float BatteryLoDB { get; set; }

    public float BatteryLoLoDB { get; set; }

    //Task UI Values
    public string[][] PREVIOUS { get; set; }
    public string[][] CURRENT { get; set; }
    public string[][] NEXT { get; set; }

    public string[][] TASK_TABLE { get; set; }
    public float SecondaryOxygenHiDB { get; internal set; }

    public bool getGreen()
    {
        return CommonData.green;
    }
    public bool getNotGreen()
    {
        return CommonData.notgreen;
    }

    public void setGreen(bool flag)
    {
        CommonData.green = flag;
    }

    public void setNotGreen(bool flag)
    {
        CommonData.notgreen = flag;
    }
}
```

DB_CSVFileParser.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.IO;

public static  class DB_CSVFileParser {

    //public static List<string[]> readfile(string path, bool removeHeader)
    //{

    //    StreamReader reader = new StreamReader(path);
    //    List<string[]> rows = new List<string[]>();

    //    //If file has header and needs to be removed remove it.
    //    if (removeHeader && !reader.EndOfStream)
    //    {
    //        reader.ReadLine();
    //    }

    //    while (!reader.EndOfStream)
    //    {
    //        string[] row = reader.ReadLine().Split(',');
    //        rows.Add(row);
    //    }

    //    reader.Close();

    //    return rows;
    //}

    //public static void WriteFile(string path, List<string[]> rows)
    //{
    //    StreamWriter writer = new StreamWriter(path);

    //    foreach (string[] r in rows)
    //    {
    //        string row = "";
    //        for (int i = 0; i < r.Length; i++)
    //        {
    //            //Last value dont add comma at the end
    //            if (i == r.Length - 1)
    //            {
    //                row += (r[i]);
    //            }
    //            else
```

```
    //          {
    //              row += (r[i] + ",");
    //          }

    //      }

    //      writer.WriteLine(row);
    //  }

    //  writer.Close();
    //}
}
```

DB_DataGenerator.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class DB_DataGenerator : MonoBehaviour {

    ////Seed
    //System.Random random = new System.Random();
    //List<Pipe> pipes = new List<Pipe>();

    //public enum dataType { Canned, Dynamic, GenerateCanned };

    //public dataType operation = dataType.Dynamic;

    ////Used for both canned and dynamic
    //public float dataUpdateSpeed = .25f;
    //private float timesinceLastUpdate = 0.0f;

    ////Generate canned data variables
    //public bool includeHeader = false;
    //public int cannedSize = 1;
    //public string generateFileName = "test";

    ////Canned data variables
    //public string filename = "test";
    //private int currentCannedIndex = 0;
    //private List<string[]> cannedData = new List<string[]>();

    //private CommonData _commonData;

    ////Initialize list of pipes
    //private void Start()
    //{
    //    _commonData = CommonData.GetInstance();

    //    //Consumables
    //    pipes.Add(new Pipe("Oxygen", 0.0f, 100.0f, 100.0f, 25.0f, 0.0f, 0.0f, 100.0f, .1f, true));
    //    pipes.Add(new Pipe("Oxygen2", 0.0f, 100.0f, 100.0f, 25.0f, 0.0f, 0.0f, 100.0f, .1f, false));
    //    pipes.Add(new Pipe("Water", 0.0f, 32.0f, 32.0f, 25.0f, 0.0f, 0.0f, 32.0f, .01f, true));
    //    pipes.Add(new Pipe("Battery", 0.0f, 100.0f, 100.0f, 25.0f, 0.0f, 0.0f, 100.0f, .01f, true));

    //    //Biometric Data
    //    pipes.Add(new Pipe("Body Temperature", 91.0f, 110.0f, 110.0f, 10.0f, 91.0f, 10.0f, 98.7f, 0.01f,
true));
    //    pipes.Add(new Pipe("Heart Rate", 62.0f, 70.0f, 70.0f, 10.0f, 62.0f, 10.0f, 65.0f, .01f, true));
```

```
//    //Suit Data
//    pipes.Add(new Pipe("Suit Pressure", 3.0f, 4.9f, 4.9f, 10.0f, 0.0f, 0.0f, 4.9f, .01f, true));

//    //Example of accesing a pipe in the list by its name and toggeling inactive
//    //pipes.Where(pipe => pipe.PipeName == "Oxygen2").First().ToggleActive();

//    //Generate a canned file if selected
//    if(operation == dataType.GenerateCanned)
//    {
//        GenerateCannedDataSet(cannedSize, generateFileName, includeHeader);
//    }
//    else if( operation == dataType.Canned)
//    {
//        cannedData = RetrieveCannedDataSet(filename, true);
//    }

//}

////Populate values
//public void Update()
//{
//    timesinceLastUpdate += Time.deltaTime;

//    if (timesinceLastUpdate > dataUpdateSpeed)
//    {
//        timesinceLastUpdate -= dataUpdateSpeed;
//        switch (operation)
//        {
//            case dataType.Canned:
//                {
//                    //Advance index
//                    currentCannedIndex++;

//                    //If index out of range start over
//                    if(currentCannedIndex >= cannedData.Count)
//                    {
//                        currentCannedIndex = 0;
//                    }

//                    //Get row from canned data
//                    string[] row = cannedData[currentCannedIndex];

//                    //Update common data with row values
//                    _commonData.OxygenOneValue = float.Parse(row[0]);
//                    _commonData.OxygenTwoValue = float.Parse(row[1]);
//                    _commonData.WaterValue = float.Parse(row[2]);
//                    _commonData.BatteryValue = float.Parse(row[3]);
```

```
//                _commonData.BodyTemperatureValue = float.Parse(row[4]);
//                _commonData.HeartRateValue = float.Parse(row[5]);
//                _commonData.SuitPressureValue = float.Parse(row[6]);

//                break;
//            }
//        case dataType.Dynamic:
//            {
//                break;
//            }
//      }
//  }
//}

////Next Step increment for all pipes returns a string of the values after
//public string[] GetNextStep()
//{
//  List<string> result = new List<string>();

//  foreach (Pipe p in pipes)
//  {
//      p.AdvancePipe(1, random);
//      result.Add(p.Current.ToString());
//  }

//  return result.ToArray();
//}

////Generat a file with canned values
//public void GenerateCannedDataSet(int size, string FileName, bool includeHeader)
//{
//  string path = Application.streamingAssetsPath + "/" + FileName + ".csv";
//  List<string[]> fileContents = new List<string[]>();

//  //Write header if it needs to be included
//  if(includeHeader)
//  {
//      fileContents.Add(GenerateHeader());
//  }

//  //Generate Body of file
//  for(int i = 0; i < size; i++)
//  {
//      fileContents.Add(GetNextStep());
//  }

//  //Write file
//  DB_CSVFileParser.WriteFile(path, fileContents);
```

```
    //}

    //public List<string[]> RetrieveCannedDataSet(string FileName,bool removeHeader)
    //{
    //    string path = Application.streamingAssetsPath + "/" + FileName + ".csv";
    //    return DB_CSVFileParser.readfile(path, removeHeader);
    //}

    //private string[] GenerateHeader()
    //{
    //    List<string> result = new List<string>();

    //    foreach (Pipe p in pipes)
    //    {
    //        result.Add(p.PipeName);
    //    }

    //    return result.ToArray();
    //}
}

class Pipe
{
    string pipeName;
    float min;
    float max;
    float threshold1;
    float threshold1Percentage;
    float threshold2;
    float threshold2Percentage;
    float initial;
    float current;
    float increment;
    bool isActive;

    public float Current { get { return current; } }
    public string PipeName { get { return pipeName; } }

    public Pipe(string pipeName, float min, float max, float threshold1, float threshold1Percentage, float
threshold2, float threshold2Percentage, float initial, float increment, bool isActive)
    {
        this.pipeName = pipeName;
        this.min = min;
        this.max = max;
        this.threshold1 = threshold1;
        this.threshold2 = threshold2;
        this.threshold1Percentage = threshold1Percentage;
        this.threshold2Percentage = threshold2Percentage;
```

```
        this.initial = initial;
        this.increment = increment;
        this.current = initial;
        this.isActive = isActive;
    }

    public void AdvancePipe(int steps, System.Random random)
    {
        //If the pipe has been deactivated return with out advancing
        if (!isActive)
        {
            return;
        }

        //Move steps by amount passed in
        for (int lcv = steps; lcv > 0; lcv--)
        {
            //Get random number between 0 and 100
            int nextRandom = random.Next(0, 101);

            //If less than first threshold subtract by increment amount
            if (nextRandom <= threshold1Percentage)
            {
                if (current - increment > min)
                {
                    current -= increment;
                }
            }
            //Stay the same if between both thresholds
            else if (nextRandom <= 100.0f - threshold2Percentage)
            {

            }
            //If greater than second threshold add by increment amount
            else
            {
                if (current + increment < max)
                {
                    current += increment;
                }
            }
        }
    }
    public void ToggleActive()
    {
        isActive = !isActive;
    }
}
```

Database

DB_GenerateData.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public enum consumableType
{
   Oxygen,
   Oxygen2,
   SuitPressure,
   Battery,
   BodyTemperature,
   Water,
   HeartRate
};

public class DB_GenerateData : MonoBehaviour {



   public consumableType type;
   public float min;
   public float max;
   public float threshold1;
   public float threshold1Percentage;
   public float threshold2;
   public float threshold2Percentage;
   public float initial;
   public float current;
   public float increment;
   public bool isActive;
   System.Random random = new System.Random();
   public CommonData commonData = CommonData.GetInstance();


   void Awake()
   {
      switch (type)
      {
         case consumableType.Water:
            commonData.WaterValue = current;
            break;
         case consumableType.Oxygen:
            commonData.OxygenOneValue = current;
            break;
         case consumableType.Oxygen2:
```

```
            commonData.OxygenTwoValue = current;
            break;
        case consumableType.Battery:
            commonData.BatteryValue = current;
            break;
        case consumableType.SuitPressure:
            commonData.SuitPressureValue = current;
            break;
        case consumableType.BodyTemperature:
            commonData.BodyTemperatureValue = current;
            break;
        case consumableType.HeartRate:
            commonData.HeartRateValue = current;
            break;

    }
}

// Use this for initialization
void Start () {

        }

        // Update is called once per frame
        void Update () {
    switch(type)
    {
        case consumableType.Water:
            commonData.WaterValue = current;
            break;
        case consumableType.Oxygen:
            commonData.OxygenOneValue = current;
            break;
        case consumableType.Oxygen2:
            commonData.OxygenTwoValue = current;
            break;
        case consumableType.Battery:
            commonData.BatteryValue = current;
            break;
        case consumableType.SuitPressure:
            commonData.SuitPressureValue = current;
            break;
        case consumableType.BodyTemperature:
            commonData.BodyTemperatureValue = current;
            break;
        case consumableType.HeartRate:
            commonData.HeartRateValue = current;
            break;
```

```
        }

        //If the pipe has been deactivated return with out advancing
        if (!isActive)
        {
            return;
        }

        //Get random number between 0 and 100
        int nextRandom = random.Next(0, 101);

        //If less than first threshold subtract by increment amount
        if (nextRandom <= threshold1Percentage)
        {
            if (current - increment > min)
            {
                current -= increment;
            }
        }
        //Stay the same if between both thresholds
        else if (nextRandom <= 100.0f - threshold2Percentage)
        {

        }
        //If greater than second threshold add by increment amount
        else
        {
            if (current + increment < max)
            {
                current += increment;
            }
        }
    }
}
```

DB_TaskTesting.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;

public class DB_TaskTesting : MonoBehaviour {

 //   CommonData commonData = CommonData.GetInstance();

 //   private Procedure procedure;

            //void Awake () {
 //      procedure = new Procedure();
 //      UITask task1 = new UITask();
 //      task1.stepNumber = 1;
 //      task1.text = "On the RIGHT side of the EVA Kit, locate the PANEL ACCESS KEY.";
 //      task1.caution = "";
 //      task1.warning = " tension-spring cable.";
 //      task1.holograms = " ";
 //      task1.images = " ";
 //      UITask task2 = new UITask();
 //      task2.stepNumber = 2; ;
 //      task2.text = "Use the PANEL ACCESS KEY to unlock the PANEL ACCESS DOOR LOCKS.";
 //      task2.caution = "The keys are on a tension-spring cable.";
 //      task2.warning = " ";
 //      task2.holograms = " ";
 //      task2.images = " ";
 //      UITask task3 = new UITask();
 //      task3.stepNumber = 3 ;
 //      task3.text = "Carefully return keys to the side of the EVA kit.";
 //      task3.caution = "Door can accidentally close";
 //      task3.warning = " ";
 //      task3.holograms = " ";
 //      task3.images = " ";
 //      UITask task4 = new UITask();
 //      task4.stepNumber = 4;
 //      task4.text = "Insert your fingers in the CENTER OPENING and secure the PANEL ACCESS DOOR in
an OPEN position.";
 //      task4.caution = "";
 //      task4.warning = "Door can accidentally close ";
 //      task4.holograms = " ";
 //      task4.images = " ";
 //      UITask task5 = new UITask();
 //      task5.stepNumber = 5;
```

```
//      task5.text = "On your belt, use the BLUE CARABINEER to securely tether to the TETHER CABLE
inside the STORAGE.";
//      task5.caution = "Notice the TETHER CABLE is adjustable";
//      task5.warning = " ";
//      task5.holograms = " ";
//      task5.images = " ";
//      UITask task6 = new UITask();
//      task6.stepNumber = 6;
//      task6.text = "Locate the E-stop button and gently press down to temporarily disable the alarm";
//      task6.warning = " ";
//      task6.holograms = "E-Stop ";
//      task6.images = " ";
//      UITask task7 = new UITask();
//      task7.stepNumber = 7;
//      task7.text = "Locate the FUSIBLE DISCONNECT box";
//      task7.warning = " ";
//      task7.holograms = "FusibleDisconnectBox ";
//      task7.images = " ";
//      UITask task8 = new UITask();
//      task8.stepNumber = 8;
//      task8.text = "Tether the BLUE CARABINEER to the TETHER CABLE";
//      task8.warning = " ";
//      task8.holograms = " ";
//      task8.images = " ";
//      UITask task9 = new UITask();
//      task9.stepNumber = 9;
//      task9.text = "Remove the BLUE CARABINEER from the FUSIBLE DISCONNECT BOX";
//      task9.warning = " ";
//      task9.holograms = " ";
//      task9.images = " ";
//      UITask task10 = new UITask();
//      task10.stepNumber = 10;
//      task10.text = " Transfer the BLUE CARABINEER to STORAGE";
//      task10.warning = " ";
//      task10.holograms = " ";
//      task10.images = " ";
//      UITask task11 = new UITask();
//      task11.stepNumber = 11;
//      task11.text = " Open the FUSIBLE DISCONNECT box and secure the lid in the open position";
//      task11.caution = " Pull the locking tab towards STORAGE with the thumb ";
//      task11.warning = " ";
//      task11.holograms = " ";
//      task11.images = " ";
//      UITask task12 = new UITask();
//      task12.stepNumber = 12;
//      task12.text = "Locate the BLACK DISCONNECT";
//      task12.warning = " ";
//      task12.holograms = " ";
```

```
//      task12.images = " ";
//      UITask task13 = new UITask();
//      task13.stepNumber = 13;
//      task13.text = " Tether it to the TETHER CABLE";
//      task13.warning = " ";
//      task13.holograms = " ";
//      task13.images = " ";
//      UITask task14 = new UITask();
//      task14.stepNumber = 14;
//      task14.text = "Remove the DISCONNECT";
//      task14.caution = " Pull up with the index and middle fingers while pushing down on the FUSE
ACCESS PANEL with the thumb ";
//      task14.warning = " ";
//      task14.holograms = " ";
//      task14.images = " ";
//      UITask task15 = new UITask();
//      task15.stepNumber = 15;
//      task15.text = "Place the DISCONNECT in the STORAGE";
//      task15.warning = " ";
//      task15.holograms = " ";
//      task15.images = " ";
//      UITask task16 = new UITask();
//      task16.stepNumber = 16;
//      task16.text = "Tether the FUSE ACCESS PANEL to the TETHER CABLE";
//      task16.warning = " ";
//      task16.holograms = " ";
//      task16.images = " ";
//      UITask task17 = new UITask();
//      task17.stepNumber = 17;
//      task17.text = "Remove the FUSE ACCESS PANEL by pulling straght up";
//      task17.warning = " ";
//      task17.holograms = " ";
//      task17.images = " ";
//      UITask task18 = new UITask();
//      task18.stepNumber = 18;
//      task18.text = "Place the FUSE ACCESS PANEL into STORAGE";
//      task18.warning = " ";
//      task18.holograms = " ";
//      task18.images = " ";
//      UITask task19 = new UITask();
//      task19.stepNumber = 19;
//      task19.text = " Tether the ALARM FUSE to the TETHER CABLE";
//      task19.warning = " ";
//      task19.holograms = " ";
//      task19.images = " ";
//      UITask task20 = new UITask();
//      task20.stepNumber = 20;
//      task20.text = "In Storage, locate the BLUE FUSE PULLER";
```

```
//      task20.warning = " ";
//      task20.holograms = " ";
//      task20.images = " ";
//      UITask task21 = new UITask();
//      task21.stepNumber = 21;
//      task21.text = "Use the BLUE FUSE PULLER to remove ONLY the ALARM FUSE";
//      task21.caution = " Rock the ALARM FUSE with the FUSE PULLER when pulling up ";
//      task21.holograms = " ";
//      task21.images = " ";
//      UITask task22 = new UITask();
//      task22.stepNumber = 22;
//      task22.text = "Return the ALARM FUSE and the FUSE PULLER to STORAGE";
//      task22.warning = " ";
//      task22.holograms = " ";
//      task22.images = " ";
//      UITask task23 = new UITask();
//      task23.stepNumber = 23;
//      task23.text = "Locate the E-stop button and gently press down to temporarily disable the alarm";
//      task23.warning = " ";
//      task23.holograms = " ";
//      task23.images = " ";
//      UITask task24 = new UITask();
//      task24.stepNumber = 24;
//      task24.text = "Reinstall it into the FUSIBLE DISCONNECT BOX";
//      task24.warning = " ";
//      task24.holograms = " FusibleDisconnectBox ";
//      task24.images = " ";
//      UITask task25 = new UITask();
//      task25.stepNumber = 25;
//      task25.text = "Remove the FUSE ACCESS PANEL tether from the TETHER CABLE and store inside";
//      task25.warning = " All tethers are under  spring tension and can retract quickly";
//      task25.caution = " ";
//      task25.holograms = " ";
//      task25.images = " ";
//      UITask task26 = new UITask();
//      task26.stepNumber = 26;
//      task26.text = "In STORAGE, locate the DISCONNECT";
//      task26.caution = "";
//      task26.warning = " ";
//      task26.holograms = "FusibleDisconnectBox ";
//      task26.images = " ";
//      UITask task27 = new UITask();
//      task27.stepNumber = 27;
//      task27.text = "Reinstall it into the DISCONNECT"; ;
//      task27.caution = "The DISCONNECT must read ON in the upper right corner to restore
Conductivity";
//      task27.warning = " ";
//      task27.holograms = "FusibleDisconnectBox ";
```

```
//      task27.images = " ";
//      UITask task28 = new UITask();
//      task28.stepNumber = 28;
//      task28.text = " Remove the DISCONNECT tether from the TETHER CABLE";
//      task28.caution = "";
//      task28.warning = " tension-spring cable.";
//      task28.holograms = " ";
//      task28.images = " ";



//      procedure.tasks.Add(task1);
//      procedure.tasks.Add(task2);
//      procedure.tasks.Add(task3);
//      procedure.tasks.Add(task4);
//      procedure.tasks.Add(task5);
//      procedure.tasks.Add(task6);
//      procedure.tasks.Add(task7);
//      procedure.tasks.Add(task8);
//      procedure.tasks.Add(task9);
//      procedure.tasks.Add(task10);
//      procedure.tasks.Add(task11);
//      procedure.tasks.Add(task12);
//      procedure.tasks.Add(task13);
//      procedure.tasks.Add(task14);
//      procedure.tasks.Add(task15);
//      procedure.tasks.Add(task16);
//      procedure.tasks.Add(task17);
//      procedure.tasks.Add(task18);
//      procedure.tasks.Add(task19);
//      procedure.tasks.Add(task20);
//      procedure.tasks.Add(task21);
//      procedure.tasks.Add(task22);
//      procedure.tasks.Add(task23);
//      procedure.tasks.Add(task24);
//      procedure.tasks.Add(task25);
//      procedure.tasks.Add(task26);

//      CommonData.GetInstance().previousTask =task6 ;
//      CommonData.GetInstance().currentTask = task7;
//      CommonData.GetInstance().nextTask = task8;





  //}
```

```
    //public Procedure GetProcedure()
    //{
    //    return procedure;
    //}

}
```

DisplayDetails.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.EventSystems;


public enum displayType
{
   Oxygen,
   OxygenTwo,
   SuitPressure,
   Battery,
   BodyTemperature,
   Water,
   HeartRate
}

public class DisplayDetails : MonoBehaviour
{
   public GameObject displaywindow;
   public Text title;
   public Text content;
   public CommonData commonData = CommonData.GetInstance();
   public BL_Main bl_main;

   public displayType displayingType;




   public void Display(string header, string values)
   {
      title.text = header;
      content.text = values;
      displaywindow.SetActive(true);
   }

   void Start()
   {

   }

   private void Update()
   {
```

```
        UpdateViewDetails();
    }

    void UpdateViewDetails()
    {
        switch(displayingType)
        {
            case displayType.SuitPressure:
                {
                    Display("Suit Pressure Details", "Pressure value " +
bl_main.GetComponent<BL_Main>().bl_scaling.actualSuitPressure() + "PSI" + '\n' + '\n' + "Primary O2: "
+ (commonData.OxygenOneValue) + "psi" + '\n' + '\n' + "Secondary O2: " +
(commonData.OxygenTwoValue) + "psi");
                    break;
                }
            case displayType.Oxygen:
                {
                    float Po2 = bl_main.bl_scaling.scallingPrimaryOxygen();
                    Po2 = Mathf.Round(Po2 * 100f) / 100; ;
                    Display("Primary Oxygen Details", "Primary O2 left: " + Po2 + "%" + '\n' + '\n' + "Primary O2: "
+ (commonData.OxygenOneValue) + "psi");
                    break;
                }
            case displayType.OxygenTwo:
                {
                    float So2 = bl_main.bl_scaling.scallingSeondaryOxygen();
                    So2 = Mathf.Round(So2 * 100f) / 100;
                    Display("Secondary Oxygen Details", "Secondary O2 left: " + So2 + "%" + '\n' + '\n' +
"Secondary O2: " + (commonData.OxygenTwoValue) + "psi");
                    break;
                }
            case displayType.Battery:
                {
                    float bat = bl_main.bl_scaling.scallingBattery();
                    bat = Mathf.Round(bat * 100f) / 100;
                    Display("Battery Details", "Battery left: " + bat + "%");
                    break;
                }
            case displayType.BodyTemperature:
                {
                    Display("Temperature Details", "Body Temperature: " +
(bl_main.bl_scaling.scallingBodyTemperature()) + " F");
                    break;
                }
            case displayType.HeartRate:
                {
                    Display("Heart Details", "Heart Rate: " + (bl_main.bl_scaling.scallingHeartRate()) + " bpm");
                    break;
```

Health UI

```
            }
        case displayType.Water:
            {
                float water = bl_main.bl_scaling.scallingWater();
                water = Mathf.Round(water * 100f) / 100;
                Display("H2O Details", "H2O left: " + water + "%" + '\n' + '\n' + "H2O :" +
commonData.WaterValue + " lbs");
                break;
            }
      }
   }
}
```

EditableObject.cs

```
using UnityEngine;
using UnityEngine.XR.WSA.Persistence;
using UnityEngine.XR.WSA;
using System;

namespace HoloToolkit.Unity.InputModule
{

    /// This code was copied from the draggable script in the Holo Tool kit
    /// It will be attached on fixed holograms that highlight objects of interest
    /// When a user gives the voice command for edit mode these objects become configurable




    /// <summary>
    /// Component that allows dragging an object with your hand on HoloLens.
    /// Dragging is done by calculating the angular delta and z-delta between the current and previous
hand positions,
    /// and then repositioning the object based on that.
    /// </summary>
    public class EditableObject : MonoBehaviour, IFocusable, IInputHandler, ISourceStateHandler
    {
        /// <summary>
        /// Event triggered when dragging starts.
        /// </summary>
        public event Action StartedDragging;

        /// <summary>
        /// Event triggered when dragging stops.
        /// </summary>
        public event Action StoppedDragging;

        [Tooltip("Transform that will be dragged. Defaults to the object of the component.")]
        public Transform HostTransform;

        [Tooltip("Scale by which hand movement in z is multiplied to move the dragged object.")]
        public float DistanceScale = 2f;

        public enum RotationModeEnum
        {
            Default,
            LockObjectRotation,
            OrientTowardUser,
            OrientTowardUserAndKeepUpright
```

```
    }

    public RotationModeEnum RotationMode = RotationModeEnum.Default;

    [Tooltip("Controls the speed at which the object will interpolate toward the desired position")]
    [Range(0.01f, 1.0f)]
    public float PositionLerpSpeed = 0.2f;

    [Tooltip("Controls the speed at which the object will interpolate toward the desired rotation")]
    [Range(0.01f, 1.0f)]
    public float RotationLerpSpeed = 0.2f;

    public bool IsDraggingEnabled = true;

    private bool isDragging;
    private bool isGazed;
    private Vector3 objRefForward;
    private Vector3 objRefUp;
    private float objRefDistance;
    private Quaternion gazeAngularOffset;
    private float handRefDistance;
    private Vector3 objRefGrabPoint;

    private Vector3 draggingPosition;
    private Quaternion draggingRotation;

    private IInputSource currentInputSource;
    private uint currentInputSourceId;
    private Rigidbody hostRigidbody;

    //##########################################################
    private CommonData data;

    private WorldAnchorStore store;
    private WorldAnchor worldAnchor;

    void Awake()
    {
      data = CommonData.GetInstance();
    }

    void RemoveAnchor()
    {
      //This statement keeps the program from throwing an error when testing from the editor
      if (store == null)
        return;

      if (this.store.Load(this.gameObject.name,this.gameObject) != null)
```

```
        {
            DestroyImmediate(gameObject.GetComponent<WorldAnchor>());
            store.Delete(this.gameObject.name);
        }
    }

    void AddAnchor()
    {
        //This statement keeps the program from throwing an error when testing from the editor
        if (store == null)
            return;

        if (this.store.Load(this.gameObject.name, this.gameObject) == null)
        {
            worldAnchor = gameObject.AddComponent<WorldAnchor>();
            this.store.Save(this.gameObject.name, worldAnchor);
        }
    }

    void StoreLoaded(WorldAnchorStore store)
    {
        this.store = store;
    }

    //##########################################################
    private void Start()
    {
        WorldAnchorStore.GetAsync(StoreLoaded);

        if (HostTransform == null)
        {
            HostTransform = transform;
        }

        hostRigidbody = HostTransform.GetComponent<Rigidbody>();
    }

    private void OnDestroy()
    {
        if (isDragging)
        {
            StopDragging();
        }

        if (isGazed)
        {
            OnFocusExit();
        }
```

Health UI

```
    }

    private void Update()
    {
        if (IsDraggingEnabled && isDragging)
        {
            UpdateDragging();
        }
    }

    /// <summary>
    /// Starts dragging the object.
    /// </summary>
    public void StartDragging(Vector3 initialDraggingPosition)
    {
        if (!IsDraggingEnabled)
        {
            return;
        }

        if (isDragging)
        {
            return;
        }

        // TODO: robertes: Fix push/pop and single-handler model so that multiple HandDraggable
components
        //      can be active at once.

        // Add self as a modal input handler, to get all inputs during the manipulation
        InputManager.Instance.PushModalInputHandler(gameObject);

        isDragging = true;

        Transform cameraTransform = CameraCache.Main.transform;

        Vector3 inputPosition = Vector3.zero;
#if UNITY_2017_2_OR_NEWER
        InteractionSourceInfo sourceKind;
        currentInputSource.TryGetSourceKind(currentInputSourceId, out sourceKind);
        switch (sourceKind)
        {
            case InteractionSourceInfo.Hand:
                currentInputSource.TryGetGripPosition(currentInputSourceId, out inputPosition);
                break;
            case InteractionSourceInfo.Controller:
                currentInputSource.TryGetPointerPosition(currentInputSourceId, out inputPosition);
                break;
```

Health UI

```
        }
#else
        currentInputSource.TryGetPointerPosition(currentInputSourceId, out inputPosition);
#endif

        Vector3 pivotPosition = GetHandPivotPosition(cameraTransform);
        handRefDistance = Vector3.Magnitude(inputPosition - pivotPosition);
        objRefDistance = Vector3.Magnitude(initialDraggingPosition - pivotPosition);

        Vector3 objForward = HostTransform.forward;
        Vector3 objUp = HostTransform.up;
        // Store where the object was grabbed from
        objRefGrabPoint =
cameraTransform.transform.InverseTransformDirection(HostTransform.position -
initialDraggingPosition);

        Vector3 objDirection = Vector3.Normalize(initialDraggingPosition - pivotPosition);
        Vector3 handDirection = Vector3.Normalize(inputPosition - pivotPosition);

        objForward = cameraTransform.InverseTransformDirection(objForward);      // in camera space
        objUp = cameraTransform.InverseTransformDirection(objUp);               // in camera space
        objDirection = cameraTransform.InverseTransformDirection(objDirection);   // in camera space
        handDirection = cameraTransform.InverseTransformDirection(handDirection); // in camera space

        objRefForward = objForward;
        objRefUp = objUp;

        // Store the initial offset between the hand and the object, so that we can consider it when
dragging
        gazeAngularOffset = Quaternion.FromToRotation(handDirection, objDirection);
        draggingPosition = initialDraggingPosition;

        StartedDragging.RaiseEvent();
      }

      /// <summary>
      /// Gets the pivot position for the hand, which is approximated to the base of the neck.
      /// </summary>
      /// <returns>Pivot position for the hand.</returns>
      private Vector3 GetHandPivotPosition(Transform cameraTransform)
      {
        return cameraTransform.position + new Vector3(0, -0.2f, 0) - cameraTransform.forward * 0.2f; //
a bit lower and behind
      }

      /// <summary>
      /// Enables or disables dragging.
      /// </summary>
```

Health UI

```
    /// <param name="isEnabled">Indicates whether dragging should be enabled or disabled.</param>
    public void SetDragging(bool isEnabled)
    {
        if (IsDraggingEnabled == isEnabled)
        {
            return;
        }

        IsDraggingEnabled = isEnabled;

        if (isDragging)
        {
            StopDragging();
        }
    }

    /// <summary>
    /// Update the position of the object being dragged.
    /// </summary>
    private void UpdateDragging()
    {
        Transform cameraTransform = CameraCache.Main.transform;

        Vector3 inputPosition = Vector3.zero;
#if UNITY_2017_2_OR_NEWER
        InteractionSourceInfo sourceKind;
        currentInputSource.TryGetSourceKind(currentInputSourceId, out sourceKind);
        switch (sourceKind)
        {
            case InteractionSourceInfo.Hand:
                currentInputSource.TryGetGripPosition(currentInputSourceId, out inputPosition);
                break;
            case InteractionSourceInfo.Controller:
                currentInputSource.TryGetPointerPosition(currentInputSourceId, out inputPosition);
                break;
        }
#else
        currentInputSource.TryGetPointerPosition(currentInputSourceId, out inputPosition);
#endif

        Vector3 pivotPosition = GetHandPivotPosition(cameraTransform);

        Vector3 newHandDirection = Vector3.Normalize(inputPosition - pivotPosition);

        newHandDirection = cameraTransform.InverseTransformDirection(newHandDirection); // in
camera space
        Vector3 targetDirection = Vector3.Normalize(gazeAngularOffset * newHandDirection);
        targetDirection = cameraTransform.TransformDirection(targetDirection); // back to world space
```

Health UI

```
        float currentHandDistance = Vector3.Magnitude(inputPosition - pivotPosition);

        float distanceRatio = currentHandDistance / handRefDistance;
        float distanceOffset = distanceRatio > 0 ? (distanceRatio - 1f) * DistanceScale : 0;
        float targetDistance = objRefDistance + distanceOffset;

        draggingPosition = pivotPosition + (targetDirection * targetDistance);

        if (RotationMode == RotationModeEnum.OrientTowardUser || RotationMode ==
RotationModeEnum.OrientTowardUserAndKeepUpright)
        {
           draggingRotation = Quaternion.LookRotation(HostTransform.position - pivotPosition);
        }
        else if (RotationMode == RotationModeEnum.LockObjectRotation)
        {
           draggingRotation = HostTransform.rotation;
        }
        else // RotationModeEnum.Default
        {
           Vector3 objForward = cameraTransform.TransformDirection(objRefForward); // in world space
           Vector3 objUp = cameraTransform.TransformDirection(objRefUp);          // in world space
           draggingRotation = Quaternion.LookRotation(objForward, objUp);
        }

        Vector3 newPosition = Vector3.Lerp(HostTransform.position, draggingPosition +
cameraTransform.TransformDirection(objRefGrabPoint), PositionLerpSpeed);
        // Apply Final Position
        if (hostRigidbody == null)
        {
           HostTransform.position = newPosition;
        }
        else
        {
           hostRigidbody.MovePosition(newPosition);
        }

        // Apply Final Rotation
        Quaternion newRotation = Quaternion.Lerp(HostTransform.rotation, draggingRotation,
RotationLerpSpeed);
        if (hostRigidbody == null)
        {
           HostTransform.rotation = newRotation;
        }
        else
        {
           hostRigidbody.MoveRotation(newRotation);
        }
```

Health UI

```
        if (RotationMode == RotationModeEnum.OrientTowardUserAndKeepUpright)
        {
            Quaternion upRotation = Quaternion.FromToRotation(HostTransform.up, Vector3.up);
            HostTransform.rotation = upRotation * HostTransform.rotation;
        }
    }

    /// <summary>
    /// Stops dragging the object.
    /// </summary>
    public void StopDragging()
    {
        if (!isDragging)
        {
            return;
        }

        // Remove self as a modal input handler
        InputManager.Instance.PopModalInputHandler();

        isDragging = false;
        currentInputSource = null;
        currentInputSourceId = 0;
        StoppedDragging.RaiseEvent();
    }

    public void OnFocusEnter()
    {
        if (!IsDraggingEnabled)
        {
            return;
        }

        if (isGazed)
        {
            return;
        }

        isGazed = true;
    }

    public void OnFocusExit()
    {
        if (!IsDraggingEnabled)
        {
            return;
        }
```

```
        if (!isGazed)
        {
            return;
        }

        isGazed = false;
    }

    public void OnInputUp(InputEventData eventData)
    {

        AddAnchor();

        if (currentInputSource != null &&
            eventData.SourceId == currentInputSourceId)
        {
            eventData.Use(); // Mark the event as used, so it doesn't fall through to other handlers.

            StopDragging();
        }
    }

    public void OnInputDown(InputEventData eventData)
    {

        if (data.EditMode == true)
        {
            RemoveAnchor();
        }
        else
        {
            return;
        }


        if (isDragging)
        {
            // We're already handling drag input, so we can't start a new drag operation.
            return;
        }

#if UNITY_2017_2_OR_NEWER
        InteractionSourceInfo sourceKind;
        eventData.InputSource.TryGetSourceKind(eventData.SourceId, out sourceKind);
        if (sourceKind != InteractionSourceInfo.Hand)
        {
```

```
            if (!eventData.InputSource.SupportsInputInfo(eventData.SourceId,
SupportedInputInfo.Position))
            {
                // The input source must provide positional data for this script to be usable
                return;
            }
        }
#else
        if (!eventData.InputSource.SupportsInputInfo(eventData.SourceId, SupportedInputInfo.Position))
        {
            // The input source must provide positional data for this script to be usable
            return;
        }
#endif

        eventData.Use(); // Mark the event as used, so it doesn't fall through to other handlers.

        currentInputSource = eventData.InputSource;
        currentInputSourceId = eventData.SourceId;

        FocusDetails? details = FocusManager.Instance.TryGetFocusDetails(eventData);

        Vector3 initialDraggingPosition = (details == null)
            ? HostTransform.position
            : details.Value.Point;

        StartDragging(initialDraggingPosition);
    }

    public void OnSourceDetected(SourceStateEventData eventData)
    {
        // Nothing to do
    }

    public void OnSourceLost(SourceStateEventData eventData)
    {
        if (currentInputSource != null && eventData.SourceId == currentInputSourceId)
        {
            StopDragging();
        }
    }
  }
}
```

EmergencyWindow.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;


public class EmergencyWindow : MonoBehaviour {

   public GameObject window;
   public Text messageField;

   //Show(string)
   // Displays the indicated emergency nessage in the pop-up window
   public void Show (string message)
   {
      messageField.text = message;
     window.SetActive(true);
   }

   // Hide Window
   //Closes the pop-up window
   /* public void Hide()
    * {
    * window.setActive(false);
    * } */
}
```

EnlargePieText.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class EnlargePieText : MonoBehaviour
{

    // Use this for initialization

    public Text PO2Text;
    public Text SO2Text;
    public Text HeartText;
    public Text BatteryText;
    public Text PressureText;
    public Text WaterText;
    public Text TemperatureText;
    public BL_Main bl_main;

    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        PO2Text.text = ("P-O2: " + (Mathf.Round(bl_main.bl_scaling.scallingPrimaryOxygen() * 100f) / 100)
+ "%");
        SO2Text.text = ("S-O2: " + (Mathf.Round(bl_main.bl_scaling.scallingSeondaryOxygen() * 100f) / 100)
+ "%");
        HeartText.text = ("Heart: " + (int)(Mathf.Round(bl_main.bl_scaling.actualHearRate() * 100f) / 100) +
"bpm");
        BatteryText.text = ("Battery: " + (Mathf.Round(bl_main.bl_scaling.scallingBattery() * 100f) / 100) +
"%");
        PressureText.text = ("Pressure: " + (Mathf.Round(bl_main.bl_scaling.actualSuitPressure() * 100f) /
100) + "psi");
        WaterText.text = ("Water: " + (Mathf.Round(bl_main.bl_scaling.scallingWater() * 100f) / 100) +
"%");
        TemperatureText.text = ("Temperature: " +
(int)(Mathf.Round(bl_main.bl_scaling.actualBodyTemperature() * 100f) / 100) + "F");
    }
}
```

Health UI

HUI_WarningWindow.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class HUI_WarningWindow : MonoBehaviour {

   public GameObject bl_main;
   public GameObject warningDetailPanel;
   public Text prevwarning;
   public Text currentwarning;
   public Text nextwarning;
   public Text WarningText;
      public GameObject singleWarningDisplay;


            // Use this for initialization
            void Start () {

      // warningDetailPanel.SetActive(true);
      singleWarningDisplay.SetActive(false);
            }



   // Update is called once per frame
   void Update()
   {
      if (bl_main.GetComponent<BL_Main>().bl_alarming.GetCurrentAlarm != null)
      {

         singleWarningDisplay.SetActive(true);
         currentwarning.text =
bl_main.GetComponent<BL_Main>().bl_alarming.GetCurrentAlarm.message;
         WarningText.text = bl_main.GetComponent<BL_Main>().bl_alarming.GetCurrentAlarm.message;
      }
      else
      {
         singleWarningDisplay.SetActive(false);
         currentwarning.text = "";
         WarningText.text = "";
      }

      if (bl_main.GetComponent<BL_Main>().bl_alarming.GetPerviousAlarm != null)
      {
```

```
        prevwarning.text = bl_main.GetComponent<BL_Main>().bl_alarming.GetPerviousAlarm.message;
        }
        else
        {
            prevwarning.text = "";
        }
       if (bl_main.GetComponent<BL_Main>().bl_alarming.GetNextAlarm != null)
        {

            nextwarning.text = bl_main.GetComponent<BL_Main>().bl_alarming.GetNextAlarm.message;
        }
       else
        {
            nextwarning.text = "";
        }
    }
}
```

HealthUIUpdates.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class HealthUIUpdates : MonoBehaviour {

   public GameObject detailedPie;
   public GameObject pie;
   public GameObject displayDetails;
   public GameObject DetailPanel;
   public DisplayDetails dd;
   public BL_Main bl_main;
   public GameObject ewindow;

   CommonData commonData = CommonData.GetInstance();

   // Use this for initialization
   void Start () {
      detailedPie.SetActive(false);
      displayDetails.SetActive(false);
      DetailPanel.SetActive(false);

            }

            // Update is called once per frame
            void Update () {
      UpdateDetailPiePanel();
      UpdateViewDetails();
      UpdateWarningWindow();
      updateEmergencyMessage();
   }

   void UpdateDetailPiePanel()
   {
      if (commonData.viewPieDetails)
      {
         detailedPie.SetActive(true);
         pie.SetActive(false);
      }
     else if (commonData.closePieDetails)
      {

         detailedPie.SetActive(false);
         pie.SetActive(true);
```

```
    }

  }

  void UpdateViewDetails()
  {
    if (commonData.viewPressure)
    {
      displayDetails.SetActive(true);
      dd.displayingType = displayType.SuitPressure;
      commonData.viewPressure = false;
    }
    else if (commonData.viewPrimaryOTwo)
    {
      displayDetails.SetActive(true);
      dd.displayingType = displayType.Oxygen;
      commonData.viewPrimaryOTwo = false;
    }
    else if (commonData.viewSecondaryOTwo)
    {
      displayDetails.SetActive(true);
      dd.displayingType = displayType.OxygenTwo;
      commonData.viewSecondaryOTwo = false;
    }

    else if (commonData.viewBattery)
    {
      displayDetails.SetActive(true);
      dd.displayingType = displayType.Battery;
      commonData.viewBattery = false;
    }
    else if (commonData.viewBodyTemperature)
    {
      displayDetails.SetActive(true);
      dd.displayingType = displayType.BodyTemperature;
      commonData.viewBodyTemperature = false;
    }
    else if (commonData.viewHeartRate)
    {
      displayDetails.SetActive(true);
      dd.displayingType = displayType.HeartRate;
      commonData.viewHeartRate = false;
    }
    else if (commonData.viewHTwoO)
    {
      displayDetails.SetActive(true);
      dd.displayingType = displayType.Water;
      commonData.viewHTwoO = false;
```

```
        }
        else if (commonData.closeDetailWindow)
        {
            displayDetails.SetActive(false);
            //  Debug.Log("Close DEtail");
            commonData.closeDetailWindow = false;
        }
    }

    void UpdateWarningWindow()
    {
        if(commonData.viewWarnings)
        {
            DetailPanel.SetActive(true);


           commonData.viewWarnings = false;
        }
        else if(commonData.closeWarnings)
        {
            DetailPanel.SetActive(false);
             commonData.closeWarnings = false;
        }

        if (DetailPanel.active)
        {
            if (bl_main.bl_alarming.GetCurrentAlarm == null)
            {
                DetailPanel.SetActive(false);
            }
        }
    }

    void updateEmergencyMessage()
    {
        if ((commonData.SuitPressureValue >= commonData.SuitPressHiHiDB &&
commonData.SuitPressureValue <= commonData.SuitPressHiHiSP) || (commonData.SuitPressureValue
>= commonData.SuitPressLoLoSP && commonData.SuitPressureValue <=
commonData.SuitPressLoLoDB))
        {

            ewindow.SetActive(true);
            ewindow.GetComponent<EmergencyWindow>().Show("Suit Pressure Low. Return Back to the
Ship." + '\n' +
           "Pressure:" + commonData.SuitPressureValue + "psi");
        }
```

```
        if ((commonData.HeartRateValue >= commonData.HeartRateHiHiDB &&
commonData.HeartRateValue <= commonData.HeartRateHiHiSP) || (commonData.HeartRateValue >=
commonData.HeartRateLoLoSP && commonData.HeartRateValue <= commonData.HeartRateLoLoDB))
        {
            ewindow.SetActive(true);
            ewindow.GetComponent<EmergencyWindow>().Show("Heart Rate Abnormal. Return Back to the
Ship" + '\n' +
           "Pressure:" + commonData.HeartRateValue + "bpm");
        }

        if ((commonData.BodyTemperatureValue >= commonData.BodyTemperatureHiHiDB &&
commonData.BodyTemperatureValue <= commonData.BodyTemperatureHiHiSP) ||
(commonData.BodyTemperatureValue >= commonData.BodyTemperatureLoLoSP &&
commonData.BodyTemperatureValue <= commonData.BodyTemperatureLoLoDB))
        {
            ewindow.SetActive(true);
            ewindow.GetComponent<EmergencyWindow>().Show("Body Temperature Abnormal. Return
Back to the Ship" + '\n' +
           "Body Temperature:" + commonData.BodyTemperatureValue + "F");
        }



            if (commonData.OxygenTwoValue <= commonData.SecondaryOxygenLoLoDB)
            {
                ewindow.SetActive(true);
                ewindow.GetComponent<EmergencyWindow>().Show("Low Oxygen Levels. Return Back to the
Ship" + '\n' +
               "Oxygen:" + bl_main.bl_scaling.scallingSeondaryOxygen() + "%");
            }


        if(commonData.WaterLoLoDB > commonData.WaterValue)
        {
            ewindow.SetActive(true);
            ewindow.GetComponent<EmergencyWindow>().Show("Water Level Low" + '\n' +
           "Water Left:" + commonData.WaterValue + "%");
        }


        if(commonData.BatteryLoLoDB > commonData.BatteryValue)
        {
            ewindow.SetActive(true);
            ewindow.GetComponent<EmergencyWindow>().Show("Suit Battery Low" + '\n' +
           "Battery Left:" + commonData.BatteryValue + "%");
        }
    }
}
```

Business Logic, Health UI

HeartRate.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;

public class HeartRate : MonoBehaviour {

   // Use this for initialization
   public float radius = 1.0f;
   public Image objHR;
   public float heartRate;
   public ColorCode cc;
   public GameObject bl_main;
   public CommonData commonData;
   void Start () {
      commonData = CommonData.GetInstance();
   }

         // Update is called once per frame
         void Update () {

      heartRate = commonData.HeartRateValue;
      radius = (bl_main.GetComponent<BL_Main>().bl_scaling.scallingHeartRate())/100;

      if (heartRate >= commonData.HeartRateHiHiDB && heartRate <= commonData.HeartRateHiHiSP)
      {
         // Debug.Log("HH");
         cc.LLCol();
         objHR.color = cc.LLColor;
      }
      else if (heartRate >= commonData.HeartRateHiDB && heartRate <= commonData.HeartRateHiSP)
      {
         // Debug.Log("H");
         cc.LCol();
         objHR.color = cc.LColor;
      }
      else if (heartRate <= commonData.HeartRateHiDB && heartRate >= commonData.HeartRateLoDB)
      {
         // Debug.Log("Ideal");
         cc.HCol();
         objHR.color = cc.HColor;
      }
      else if (heartRate >= commonData.HeartRateLoSP && heartRate <= commonData.HeartRateLoDB)
      {
```

```
            //Debug.Log("L");
            cc.LCol();
            objHR.color = cc.LColor;
        }
        else if (heartRate >= commonData.HeartRateLoLoSP && heartRate <=
commonData.HeartRateLoLoDB)
        {
            // Debug.Log("LL");
            cc.LLCol();
            objHR.color = cc.LLColor;
        }

            objHR.transform.localScale = new Vector3(radius, radius, 1.0f);
    }
}
```

Keywords.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using HoloToolkit.Unity.InputModule;
using System;

public class Keywords : MonoBehaviour,ISpeechHandler {

    ///This script is where voice command functionality is defined
    ///The buisness layer will recieve voice commands and resolve them

    //Need Instance of buisnees layer
    public bool IsGlobalListener = true;
    public TextMesh testText;


    private CommonData data;

    private void OnEnable()
    {
        InputManager.Instance.AddGlobalListener(gameObject);
    }

    void Awake()
    {
        data = CommonData.GetInstance();
    }

    public void OnSpeechKeywordRecognized(SpeechEventData eventData)
    {
        switch (eventData.RecognizedText.ToLower())
        {
            case "start edit mode":
                {
                    StartEditMode();
                    break;
                }
            case "stop edit mode":
                {
                    StopEditMode();
                    break;
                }
            case "view pie details":
                {
                    ViewPieDetails();
```

```
            break;
        }
    case "close pie details":
        {
            ClosePieDetails();
            break;
        }
    case "view battery":
        {
            ViewBattery();
            break;
        }
    case "view primary o two":
        {
            ViewPrimaryOTwo();
            break;
        }
    case "view secondary o two":
        {
            ViewSecondaryOTwo();
            break;
        }
    case "view body temperature":
        {
            ViewBodyTemperature();
            break;
        }
    case "view pressure":
        {
            ViewPressure();
            break;
        }
    case "view h two o":
        {
            ViewHTwoO();
            break;
        }
    case "view heart rate":
        {
            ViewHeartRate();
            break;
        }
    case "close detail window":
        {
            CloseDetailWindow();
            break;
        }
    case "pin detail window":
```

```
        {
            PinDetailWindow();
            break;
        }
    case "close pinned windows":
        {
            ClosePinnedWindows();
            break;
        }
    case "view errors":
        {
            ViewErrors();
            break;
        }
    case "close errors":
        {
            CloseErrors();
            break;
        }
    case "next error":
        {
            NextError();
            break;
        }
    case "previous error":
        {
            PreviousError();
            break;
        }
    case "view warnings":
        {
            ViewWarnings();
            break;
        }
    case "close warnings":
        {
            CloseWarnings();
            break;
        }
    case "next warning":
        {
            NextWarning();
            break;
        }
    case "previous warning":
        {
            PreviousWarning();
            break;
```

```
    }
  case "start procedure":
    {
      StartProcedure();
      break;
    }
  case "next step":
    {
      NextStep();
      break;
    }
  case "previous step":
    {
      PreviousStep();
      break;
    }
  case "show step image":
    {
      ShowStepImage();
      break;
    }
  case "hide step image":
    {
      HideStepImage();
      break;
    }
  case "pin task list":
    {
      PinTaskList();
      break;
    }
  case "close task list":
    {
      CloseTaskList();
      break;
    }
  case "display pie":
    {
      DisplayPie();
      break;
    }
  case "display alternate":
    {
      DisplayAlternate();
      break;
    }
  case "load task one":
    {
```

```
            LoadTaskOne();
            break;
          }
        case "load task two":
          {
            LoadTaskTwo();
            break;
          }
        case "warning acknowledged":
          {
            WarningAcknowledged();
            break;
          }
        case "green":
        {
            Green();
            break;
         }
        case "not green":
          {
            NotGreen();
            break;
          }

        default:
          break;
      }
    }

    private void NotGreen()
    {
      testText.text = "Not Green light";
      CommonData.notgreen = true;
    }

    private void Green()
    {
      testText.text = "Green light";
      CommonData.green = true;
    }

    private void WarningAcknowledged()
    {
      testText.text = "Warning acknowledged";
      data.warningAcknowledged = true;
    }

    private void LoadTaskTwo()
```

```
{
    testText.text = "Load Task Two";
    data.loadTaskTwo = true;
}

private void LoadTaskOne()
{
    testText.text = "Load Task One";
    data.loadTaskOne = true;
}

private void DisplayAlternate()
{
    GameObject.FindGameObjectWithTag("PieUI").GetComponent<Canvas>().enabled = false;
    GameObject.FindGameObjectWithTag("AlternatePieUI").GetComponent<Canvas>().enabled = true;
}

private void DisplayPie()
{
    GameObject.FindGameObjectWithTag("PieUI").GetComponent<Canvas>().enabled = true;
    GameObject.FindGameObjectWithTag("AlternatePieUI").GetComponent<Canvas>().enabled = false;

}

private void CloseTaskList()
{
    testText.text = "Close Task List";
    data.closeTaskList = true;
}

private void PinTaskList()
{
    testText.text = "Pin Task List";
    data.pinTaskList = true;
}

private void HideStepImage()
{
    testText.text = "Hide Step Image";
    data.hideStepImage = true;
}

private void ShowStepImage()
{
    testText.text = "Show Step Image";
    data.showStepImage = true;
}
```

```
private void PreviousStep()
{
   testText.text = "Previous Step";
   data.previousStep = true;
}

private void NextStep()
{
   testText.text = "Next Step";
   data.nextStep = true;
}

private void StartProcedure()
{
   testText.text = "Start Procedure";
   data.startProcedure = true;
}

private void PreviousWarning()
{
   testText.text = "Previous Warning";
   data.previousWarning = true;
}

private void NextWarning()
{
   testText.text = "Next Warning";
   data.nextWarning = true;
}

private void CloseWarnings()
{
   testText.text = "Close Warnings";
   data.closeWarnings = true;
}

private void ViewWarnings()
{
   testText.text = "View Warnings";
   data.viewWarnings = true;
}

private void PreviousError()
{
   testText.text = "Previous Error";
   data.previousError = true;
}
```

```
private void NextError()
{
   testText.text = "Next Error";
   data.nextError = true;
}

private void CloseErrors()
{
   testText.text = "Close Errors";
   data.closeErrors = true;
}

private void ViewErrors()
{
   testText.text = "View Errors";
   data.viewErrors = true;
}

private void ClosePinnedWindows()
{
   testText.text = "Close Pinned Windows";
   data.closePinnedWindows = true;
}

private void PinDetailWindow()
{
   testText.text = "Pin Detail Window";
   data.pinDetailWindow = true;
}

private void ViewPieDetails()
{
   testText.text = "View Pie Details";
   data.viewPieDetails = true;
}

private void ClosePieDetails()
{
   testText.text = "Close Pie Details";
   data.viewPieDetails = false;
   data.closePieDetails = true;
}

private void ViewBattery()
{
   testText.text = "View Battery";
   data.viewBattery = true;
}
```

```
private void ViewPrimaryOTwo()
{
   testText.text = "View Primary O Two";
   data.viewPrimaryOTwo = true;
}

private void ViewSecondaryOTwo()
{
   testText.text = "View Secondary O Two";
   data.viewSecondaryOTwo = true;
}

private void ViewBodyTemperature()
{
   testText.text = "View Body Temperature";
   data.viewBodyTemperature = true;
}

private void ViewPressure()
{
   testText.text = "View Pressure";
   data.viewPressure = true;
}

private void ViewHTwoO()
{
   testText.text = "View H Two O";
   data.viewHTwoO = true;
}

private void ViewHeartRate()
{
   testText.text = "View Heart Rate";
   data.viewHeartRate = true;
}

private void CloseDetailWindow()
{
   testText.text = "Close Detail Window";
   data.viewPressure = false;
   data.viewPrimaryOTwo = false;
   data.viewSecondaryOTwo = false;
   data.viewHeartRate = false;
   data.viewBattery = false;
   data.viewBodyTemperature = false;
   data.viewHTwoO = false;
   data.closeDetailWindow = true;
```

```
    }

    private void StopEditMode()
    {
        testText.text = "Stop edit mode.";
        data.EditMode = false;
    }

    private void StartEditMode()
    {
        testText.text = "Start edit mode.";
        data.EditMode = true;
    }
}
```

Scripts/Keywords.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using HoloToolkit.Unity.InputModule;
using System;

public class Keywords : MonoBehaviour,ISpeechHandler {

    ///This script is where voice command functionality is defined
    ///The buisness layer will recieve voice commands and resolve them

    //Need Instance of buisnees layer
    public bool IsGlobalListener = true;
    public TextMesh testText;

    private CommonData data;

    private void OnEnable()
    {
        InputManager.Instance.AddGlobalListener(gameObject);
    }

    void Awake()
    {
        data = CommonData.GetInstance();
    }

    public void OnSpeechKeywordRecognized(SpeechEventData eventData)
    {
        switch (eventData.RecognizedText.ToLower())
        {
            case "start edit mode":
                {
                    StartEditMode();
                    break;
                }
            case "stop edit mode":
                {
                    StopEditMode();
                    break;
                }
            case "view pie details":
                {
                    ViewPieDetails();
                    break;
```

```
    }
case "close pie details":
    {
        ClosePieDetails();
        break;
    }
case "view battery":
    {
        ViewBattery();
        break;
    }
case "view primary o two":
    {
        ViewPrimaryOTwo();
        break;
    }
case "view secondary o two":
    {
        ViewSecondaryOTwo();
        break;
    }
case "view body temperature":
    {
        ViewBodyTemperature();
        break;
    }
case "view pressure":
    {
        ViewPressure();
        break;
    }
case "view h two o":
    {
        ViewHTwoO();
        break;
    }
case "view heart rate":
    {
        ViewHeartRate();
        break;
    }
case "close detail window":
    {
        CloseDetailWindow();
        break;
    }
case "pin detail window":
    {
```

```
            PinDetailWindow();
            break;
        }
    case "close pinned windows":
        {
            ClosePinnedWindows();
            break;
        }
    case "view errors":
        {
            ViewErrors();
            break;
        }
    case "close errors":
        {
            CloseErrors();
            break;
        }
    case "next error":
        {
            NextError();
            break;
        }
    case "previous error":
        {
            PreviousError();
            break;
        }
    case "view warnings":
        {
            ViewWarnings();
            break;
        }
    case "close warnings":
        {
            CloseWarnings();
            break;
        }
    case "next warning":
        {
            NextWarning();
            break;
        }
    case "previous warning":
        {
            PreviousWarning();
            break;
        }
```

```
case "start procedure":
  {
    StartProcedure();
    break;
  }
case "next step":
  {
    NextStep();
    break;
  }
case "previous step":
  {
    PreviousStep();
    break;
  }
case "show step image":
  {
    ShowStepImage();
    break;
  }
case "hide step image":
  {
    HideStepImage();
    break;
  }
case "pin task list":
  {
    PinTaskList();
    break;
  }
case "close task list":
  {
    CloseTaskList();
    break;
  }
case "display pie":
  {
    DisplayPie();
    break;
  }
case "display alternate":
  {
    DisplayAlternate();
    break;
  }
case "load task one":
  {
    LoadTaskOne();
```

```
            break;
        }
    case "load task two":
        {
            LoadTaskTwo();
            break;
        }
    case "warning acknowledged":
        {
            WarningAcknowledged();
            break;
        }
    case "green":
    {
            Green();
            break;
     }
    case "not green":
        {
            NotGreen();
            break;
        }

    default:
        break;
  }
}

private void NotGreen()
{
  testText.text = "Not Green light";
  CommonData.notgreen = true;
}

private void Green()
{
  testText.text = "Green light";
  CommonData.green = true;
}

private void WarningAcknowledged()
{
  testText.text = "Warning acknowledged";
  data.warningAcknowledged = true;
}

private void LoadTaskTwo()
{
```

```
        testText.text = "Load Task Two";
        data.loadTaskTwo = true;
    }

    private void LoadTaskOne()
    {
        testText.text = "Load Task One";
        data.loadTaskOne = true;
    }

    private void DisplayAlternate()
    {
        GameObject.FindGameObjectWithTag("PieUI").GetComponent<Canvas>().enabled = false;
        GameObject.FindGameObjectWithTag("AlternatePieUI").GetComponent<Canvas>().enabled = true;
    }

    private void DisplayPie()
    {
        GameObject.FindGameObjectWithTag("PieUI").GetComponent<Canvas>().enabled = true;
        GameObject.FindGameObjectWithTag("AlternatePieUI").GetComponent<Canvas>().enabled = false;

    }

    private void CloseTaskList()
    {
        testText.text = "Close Task List";
        data.closeTaskList = true;
    }

    private void PinTaskList()
    {
        testText.text = "Pin Task List";
        data.pinTaskList = true;
    }

    private void HideStepImage()
    {
        testText.text = "Hide Step Image";
        data.hideStepImage = true;
    }

    private void ShowStepImage()
    {
        testText.text = "Show Step Image";
        data.showStepImage = true;
    }

    private void PreviousStep()
```

```
    {
        testText.text = "Previous Step";
        data.previousStep = true;
    }

    private void NextStep()
    {
        testText.text = "Next Step";
        data.nextStep = true;
    }

    private void StartProcedure()
    {
        testText.text = "Start Procedure";
        data.startProcedure = true;
    }

    private void PreviousWarning()
    {
        testText.text = "Previous Warning";
        data.previousWarning = true;
    }

    private void NextWarning()
    {
        testText.text = "Next Warning";
        data.nextWarning = true;
    }

    private void CloseWarnings()
    {
        testText.text = "Close Warnings";
        data.closeWarnings = true;
    }

    private void ViewWarnings()
    {
        testText.text = "View Warnings";
        data.viewWarnings = true;
    }

    private void PreviousError()
    {
        testText.text = "Previous Error";
        data.previousError = true;
    }

    private void NextError()
```

```
{
   testText.text = "Next Error";
   data.nextError = true;
}

private void CloseErrors()
{
   testText.text = "Close Errors";
   data.closeErrors = true;
}

private void ViewErrors()
{
   testText.text = "View Errors";
   data.viewErrors = true;
}

private void ClosePinnedWindows()
{
   testText.text = "Close Pinned Windows";
   data.closePinnedWindows = true;
}

private void PinDetailWindow()
{
   testText.text = "Pin Detail Window";
   data.pinDetailWindow = true;
}

private void ViewPieDetails()
{
   testText.text = "View Pie Details";
   data.viewPieDetails = true;
}

private void ClosePieDetails()
{
   testText.text = "Close Pie Details";
   data.viewPieDetails = false;
   data.closePieDetails = true;
}

private void ViewBattery()
{
   testText.text = "View Battery";
   data.viewBattery = true;
}
```

```
private void ViewPrimaryOTwo()
{
   testText.text = "View Primary O Two";
   data.viewPrimaryOTwo = true;
}

private void ViewSecondaryOTwo()
{
   testText.text = "View Secondary O Two";
   data.viewSecondaryOTwo = true;
}

private void ViewBodyTemperature()
{
   testText.text = "View Body Temperature";
   data.viewBodyTemperature = true;
}

private void ViewPressure()
{
   testText.text = "View Pressure";
   data.viewPressure = true;
}

private void ViewHTwoO()
{
   testText.text = "View H Two O";
   data.viewHTwoO = true;
}

private void ViewHeartRate()
{
   testText.text = "View Heart Rate";
   data.viewHeartRate = true;
}

private void CloseDetailWindow()
{
   testText.text = "Close Detail Window";
   data.viewPressure = false;
   data.viewPrimaryOTwo = false;
   data.viewSecondaryOTwo = false;
   data.viewHeartRate = false;
   data.viewBattery = false;
   data.viewBodyTemperature = false;
   data.viewHTwoO = false;
   data.closeDetailWindow = true;
}
```

```
    private void StopEditMode()
    {
        testText.text = "Stop edit mode.";
        data.EditMode = false;
    }

    private void StartEditMode()
    {
        testText.text = "Start edit mode.";
        data.EditMode = true;
    }
}
```

Pressure.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;
public class Pressure : MonoBehaviour {

   // Use this for initialization

   public float radius = 100.0f;
   public Image objP;
   public float suitPressure;
   public CommonData commonData;


   public ColorCode cc;

   public GameObject bl_main;

   // public Color HHColor = new Color(0.39f, 0.58f, 0.93f, 1.0f);
   void Start () {
      commonData = CommonData.GetInstance();
   }

   // Update is called once per frame
   void Update()
   {

      radius = bl_main.GetComponent<BL_Main>().bl_scaling.ScalingFunction();
      suitPressure = commonData.SuitPressureValue;

      if (suitPressure >= commonData.SuitPressHiHiDB && suitPressure <= commonData.SuitPressHiHiSP)
      {
         cc.LLCol();
         objP.color = cc.LLColor;
      }
      else if (suitPressure >= commonData.SuitPressHiDB && suitPressure <= commonData.SuitPressHiSP)
      {
         cc.LCol();
         objP.color = cc.LColor;
      }
      else if (suitPressure <= commonData.SuitPressHiDB && suitPressure >=
commonData.SuitPressLoDB)
      {
         cc.HCol();
```

```
            objP.color = cc.HColor;
        }
        else if (suitPressure >= commonData.SuitPressLoSP && suitPressure <=
commonData.SuitPressLoDB)
        {
            cc.LCol();
            objP.color = cc.LColor;
        }
        else if (suitPressure >= commonData.SuitPressLoLoSP && suitPressure <=
commonData.SuitPressLoLoDB)
        {
            cc.LLCol();
            objP.color = cc.LLColor;

        }

            objP.transform.localScale = new Vector3(radius/100, radius/100, 1.0f);


    }
}
```

PrimaryO2.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;

public class PrimaryO2 : MonoBehaviour {

   // Use this for initialization
   public float radius = 100.0f;
   public Image obj;
   public float currentHealth;
   // public GameObject busLayer;
   public BL_Main bl_main;
   public ColorCode ccPO;

   public CommonData commonData = CommonData.GetInstance();

   void Start () {
      //Assign max value to cuurent health status for initialization
     // currentHealth = (Console.Read());

      //Attach the pressurelog from the buslayer if the buslayer was assigned in the editor
    /*  if (busLayer != null)
      {
         pressurelog = busLayer.GetComponent<BL_Main>();
      }
      //If not attached in the editor find one in the current scene
      else
      {
         busLayer = GameObject.FindGameObjectWithTag("Bus Layer");
         pressurelog = busLayer.GetComponent<BL_Main>();
      }*/
   }

         // Update is called once per frame
         void Update () {
      float scaledValue = bl_main.GetComponent<BL_Main>().bl_scaling.scallingPrimaryOxygen();

      radius = scaledValue;


      if (commonData.OxygenOneValue > commonData.PrimaryOxygenLoDB)
      { // Debug.Log("H");
         ccPO.HCol();
```

```
            obj.color = ccPO.HColor;
        }
        else if (commonData.OxygenOneValue <= commonData.PrimaryOxygenLoDB)
        {  //  Debug.Log("L");
            ccPO.LCol();
            obj.color = ccPO.LColor;
        }


            obj.transform.localScale = new Vector3(radius/100, radius/100, 1.0f);
    }
}
```

SecondaryO2.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;

public class SecondaryO2 : MonoBehaviour
{

   // Use this for initialization
   public float radius = 1.0f;
   public Image objSO2;
   public ColorCode ccSO;
   public CommonData commonData;
   public BL_Main bl_main;

   void Start()
   {
      commonData = CommonData.GetInstance();
      ccSO.HHCol();
      objSO2.color = ccSO.HHColor;
   }

   // Update is called once per frame
   void Update()
   {
      radius = (bl_main.GetComponent<BL_Main>().bl_scaling.scallingSeondaryOxygen());


     if (commonData.OxygenOneValue < 1.00f)
     {
         if (commonData.OxygenTwoValue > commonData.SecondaryOxygenHiDB)
         {
            // Debug.Log("HH");
            ccSO.HHCol();
            objSO2.color = ccSO.HHColor;
         }
         else if (commonData.SecondaryOxygenHiDB > commonData.OxygenTwoValue &&
commonData.OxygenTwoValue > commonData.SecondaryOxygenLoDB)
         {
            // Debug.Log("H");
            ccSO.HCol();
            objSO2.color = ccSO.HColor;
         }
```

Health UI

```
        else if (commonData.SecondaryOxygenLoDB > commonData.OxygenTwoValue &&
commonData.OxygenTwoValue > commonData.SecondaryOxygenLoLoDB)
        {
            //Debug.Log("L");
            ccSO.LCol();
            objSO2.color = ccSO.LColor;
        }

        else if (commonData.OxygenTwoValue < commonData.SecondaryOxygenLoLoDB)
        {
            //Debug.Log("LL");
            ccSO.LLCol();
            objSO2.color = ccSO.LLColor;
        }

        objSO2.transform.localScale = new Vector3(radius/100, radius/100, 1.0f);
    }
  }
}
```

SuitBattery.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;

public class SuitBattery : MonoBehaviour
{

   // Use this for initialization
   public float radius;
   public Image objB;

   public CommonData commonData;
   public ColorCode cc;
   public BL_Main bl_main;
   void Start()
   {
      commonData = CommonData.GetInstance();
      //Assign max value to current health status for initialization
   }

   // Update is called once per frame
   void Update()
   {
      radius = bl_main.bl_scaling.scallingBattery();


      if (commonData.BatteryValue > commonData.BatteryLoDB)
      {
         //  Debug.Log("HH");
         cc.HCol();
         objB.color = cc.HColor;
      }

      else if (commonData.BatteryLoDB > commonData.BatteryValue && commonData.BatteryValue >
commonData.BatteryLoLoDB)
      {
         // Debug.Log("L");
         cc.LCol();
         objB.color = cc.LColor;
      }

      else if (commonData.BatteryLoLoDB > commonData.BatteryValue)
      {
```

```
            //  Debug.Log("LL");
            cc.LLCol();
            objB.color = cc.LLColor;
        }


            objB.transform.localScale = new Vector3(radius/100, radius/100, 1.0f);
    }
}
```

Health UI

TUI_TaskUIUpdate.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;
using HoloToolkit.Unity;

public class TUI_TaskUIUpdate : MonoBehaviour {
   CommonData commonData = CommonData.GetInstance();
   public TextToSpeech textToSpeech;




   public Text previousTask;
   public Text currentTask;
   public Text nextTask;
   public GameObject imagePanel;
   public Text  cautionText;
   public GameObject taskPanel;
   public GameObject cautionPanel;
   public Image cautionImage;

   public BL_Tasks bl_tasks;

   private int currentstep = 0;
   private bool spoke = false;

   private int num = 1;

   void Start () {
      bl_tasks = new BL_Tasks();
      taskPanel.gameObject.SetActive(false);
      cautionPanel.gameObject.SetActive(false);
      imagePanel.gameObject.SetActive(false);


   }

         // Update is called once per frame
         void Update ()
   {
      if(commonData.loadTaskOne)
      {
         bl_tasks.LoadTaskOne();
```

```
        commonData.loadTaskOne = false;
        num = 1;
    }

    if(commonData.loadTaskTwo)
    {
        bl_tasks.LoadTaskTwo();
        commonData.loadTaskTwo = false;
        num = 2;
    }

    taskPanel.gameObject.SetActive(true);

    currentTask.text = "TASK ARRIVED";

    if (commonData.startProcedure)
    {
        GameObject[] SceneHolograms = GameObject.FindGameObjectsWithTag("Hologram");
        foreach (GameObject h in SceneHolograms)
        {
            h.GetComponent<Renderer>().enabled = false;
            h.GetComponent<BoxCollider>().enabled = false;
        }


        if (bl_tasks.currentTask.holograms != "")
        {
            String[] holograms = bl_tasks.currentTask.holograms.Split(',');
            foreach (string h in holograms)
            {
                GameObject hologram = GameObject.Find(bl_tasks.currentTask.holograms);


                hologram.GetComponent<Renderer>().enabled = true;
            }

        }


        previousTask.text = bl_tasks.previousTask.text;
        currentTask.text = bl_tasks.currentTask.text;
        //String voiceData = bl_tasks.currentTask.text;
        nextTask.text = bl_tasks.nextTask.text;


        cautionPanel.gameObject.SetActive(false);
        imagePanel.gameObject.SetActive(false);
```

```
        string warning = bl_tasks.currentTask.warning.ToString();
        string caution = bl_tasks.currentTask.caution.ToString();
        string images = bl_tasks.currentTask.images.ToString();


        if (images != "")
        {
           imagePanel.gameObject.SetActive(true);

           imagePanel.GetComponent<Image>().sprite =
Resources.Load<Sprite>(Convert.ToString(bl_tasks.currentTask.images));

        }
        if (warning != "")
        {
           cautionPanel.gameObject.SetActive(true);
           cautionPanel.GetComponent<Image>().color = new Color32(208, 46, 40, 214);
           cautionText.text = bl_tasks.currentTask.warning;
        }
        if (caution != "")
        {
           cautionPanel.gameObject.SetActive(true);
           cautionPanel.GetComponent<Image>().color = new Color32(255, 255, 114, 249);
           cautionText.text = bl_tasks.currentTask.caution;
        }

        if (currentstep != commonData.currentTask.stepNumber)
        {
           spoke = false;
           currentstep = commonData.currentTask.stepNumber;
        }


        if (spoke == false)
        {
           textToSpeech.StartSpeaking(bl_tasks.currentTask.text);
           spoke = true;
        }

        if (commonData.nextStep)
        {
           if( num == 1 ) bl_tasks.nextFunction(true);
           if (num == 2) bl_tasks.nextFunction2(true);
           commonData.nextStep = false;
        }
```

```
        if (commonData.previousStep)
        {
            if(num ==1 ) bl_tasks.previousFunction(true);
            if (num == 2) bl_tasks.previousFunction2(true);
            commonData.previousStep = false;
        }
    }
  }
}
```

Water.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;

public class Water : MonoBehaviour {

   // Use this for initialization
   public float radius = 1.0f;
   public Image objW;

   public CommonData commonData;
   public ColorCode cc;
   public BL_Main bl_main;



   void Start () {
      commonData = CommonData.GetInstance();
   }

         // Update is called once per frame
         void Update () {


      radius = (bl_main.bl_scaling.scallingWater()) / 100;




      if(commonData.WaterLoDB < commonData.WaterValue)
      {
         //   Debug.Log("H");
         cc.HCol();
         objW.color = cc.HColor;
      }
      else if (commonData.WaterLoDB > commonData.WaterValue && commonData.WaterValue >
commonData.WaterLoLoDB)
      {
         // Debug.Log("L");
         cc.LCol();
         objW.color = cc.LColor;
      }
```

Health UI

```
        else if (commonData.WaterLoLoDB > commonData.WaterValue)
        {
           //Debug.Log("LL");
            cc.LLCol();
           objW.color = cc.LLColor;
        }


           objW.transform.localScale = new Vector3(radius, radius, 1.0f);
    }
}
```

Health UI

REFERENCES

(n.d.). Retrieved May 2, 2018, from http://www.calculator.net/standard-deviation-calculator.html?numberinputs=9,9,9,9,9,9,9,9,9,8,8,8,8,6&x=60&y=18

Appendix A: NASA S.U.I.T.S. Challenge Document

NASA SUITS 2018 Design Challenge

Revised 10.13.2017

Introduction to the SUITS Program

The NASA Spacesuit User Interface Technologies for Students (NASA S.U.I.T.S.) Design Challenge is a mission-driven project in which university student teams design and create spacesuit informatics using an augmented reality (AR) Microsoft HoloLens platform. *Informatics* are the avionics of a spacesuit which help an astronaut become more efficient and effective during a spacewalk, often in the form of visual displays. The student-designed visual display and audio environments will present information to aid astronaut subjects in performing simulated Extravehicular Activity (EVA) tasks. After developing their environment, selected student teams will have the opportunity to travel to NASA Johnson Space Center to test their prototypes in an on-site facility.

Introduction to NASA's Advanced Spacesuit Informatics System

During a spacewalk, an astronaut's job involves focus, direction, and communication. Currently, an EVA crewmember communicates details about all tasks by means of a voice connection with mission control, an EVA partner, and an intravehicular (IVA) crewmember (an astronaut inside the pressurized spacecraft). For years, voice has been the only means of communicating during a spacewalk. However, NASA is developing innovative helmet-based displays which can perform this function much more efficiently, leading to less voice conversation, and a more proficient system. These displays align with the Informatics Subsystem of NASA's Advanced Spacesuit.

At JSC, the Advanced EVA Development team is working to develop a spacesuit for astronauts to use for future missions which travel further from Earth than ever before. The Exploration Extravehicular Mobility Unit (xEMU) will be designed for spacewalks in microgravity or on the lunar surface and will kick-start the design of future Mars spacesuits. Because of its bold mission requirements, the xEMU will have several technological upgrades that have never been used in space. One of these upgrades is the introduction of an Informatics subsystem.

The xEMU's Informatics subsystem will include many new capabilities that will make astronauts more resourceful and autonomous during EVAs, which is essential as missions move further and further away from ground support, both literally and figuratively. The crew will need to be *more autonomous*, and need more assistance from modern technology similar to the devices that make humans more autonomous on Earth. The Informatics subsystem will include new cameras, lights, a navigation system, and crew interfaces, including an electronic display which will display task procedures, schedules, pictures, schematics, videos, suit status, suit performance info, and other essential information.

The biggest design challenge for the Informatics subsystem is working out how best to interface with the human. What is the best design of robust, space-rated hardware that can stand up to years of use in a demanding deep-space environment, while keeping an astronaut safe? How do we best use emerging AR technologies to process and present data to a spacewalker in effective and intuitive ways? Which spacesuit data are the right data to sense, collect, and transmit? What will be the user experience of our future astronauts as they walk across the surface of Mars? These are the questions that NASA is only beginning to answer, and WE WANT YOUR HELP.

Test Objectives

The objective of the SUITS activity is to complete an EVA-simulated task while utilizing the HoloLens as an aid to instruct the user. The task may require dexterity, physical activity, and moving between various points within a room. A set of instructions are to be presented to the user via the display environment (audibly, visually, etc.) in order to complete the task most efficiently.

Initial Test Guidance

- The test subject conducting the EVA will be standing in a room with access to an EVA task board.
- The test subject will be wearing gloves (to limit dexterity) and a tether belt.
- One team member will be available as a support role during the EVA to read instructions and act as Mission Control.
- The test subject will be using a HoloLens during the EVA.
- Any external tools or additions to the HoloLens test setup must be *approved by NASA points of contact.*
- Teams will use Unity and C# to program interfaces. As this is necessary for compatibility with NASA provided data.
- NASA will provide subsets of data for the interface. This data will include spacesuit and biomedical data points, a set of instructions to complete the EVA tasks, and possibly other items available for processing by the user interface.
- The interface must be adaptable to multiple situations and sets of data.
- The team will have a set amount of time prior to the competition to incorporate the NASA data into the interface design.
- Teams will run through a procedure with communication to "ground control" with a specified latency with mission critical updates from ground control.

Test Scenario

Each student team will load their environment onto a HoloLens that will be worn by a test subject. After starting the environment on the HoloLens, the test subject will run through a procedure that involves completing a number of tasks. The procedure will include all the necessary information to complete the tasks; however, the Informatics environments developed by the student teams should aim to aid the test subject. The goal is to provide the

test subject information that makes them more efficient at completing a given task, without becoming a distraction.
The following tasks may be completed as part of the procedure:

- Providing suit information to ground control
  - This will involve a person reporting to the subject to provide information about a specific suit parameter. For instance, the question might be "What is your suit pressure?" or "How much battery life do you have left?"
  - The subject must be able to find this information in their display system, and deliver the information to "ground control," in this case, their partner.

- Various tasks with the EVA task board
  - Removing and reinstalling fuses with provided tools;
  - Operating keys and unlocking locked cases;
  - Installing cables into outlets with correct orientation
  - Cable routing

Example Procedure Format:

1. *Locate battery pack. Ensure that it is switched **ON**. Connect battery to load marked **BATTERY POWERED**.*
2. *Notate and verbally record that the **POWERED** light is enabled and lit.*
3. *Locate the key in **STORAGE** labeled "BREAKER BOX." Use BREAKER BOX key to open compartment labeled "BREAKER BOX."*

These tasks are made to simulate an EVA as accurately as possible on Earth. As a result, participants will be suited as such, wearing items such as bulky gloves or a tethered belt, for example. Outside distractions, stressors, and lighting changes may also be introduced in order to better simulate an EVA environment.

Figure 1. Shows the inside of an EVA taskboard. A similar grouping of tasks may need to be completed when students showcase at JSC.

Figure 2. Example of tools used to complete the tasks required in the EVAs.

Figure 3. JSC intern Sarah Kuchel uses HoloLens as a guide to complete the EVA task.

NASA Testing Logistics

Once student teams arrive at JSC, they will participate in two test days during the five-day, onsite experience.

- *Day One (No Testing): Arrivals, Welcomes, & Center and Facility Orientation*
- **Day Two Testing:** A test subject will wear a HoloLens loaded with the student team's environment. The subject will then complete a variety of tasks while following a procedure. The goal for the student teams is to develop a display and auditory environment that aids the test subject in performing these tasks as efficiently and correctly as possible. The test subject will give student teams feedback on their displays and the teams will be given time to implement changes to their design if needed.
- *Day Three (No Testing): Optimizing design solutions*
- **Day Four Testing:** Another test subject will perform the same series of tasks with the team's updated environments.
- *Day Five (No Testing): Wrap up activities and release*

Program Deliverables

Specific program deliverable requirements will be available via a separate document titled: NASA SUITS 2018 Proposal Guidelines. General team deliverables will include:

- Proposal Reports
- Project Schedule
- Code and Custom Asset
- Outreach Plan
- Final Report

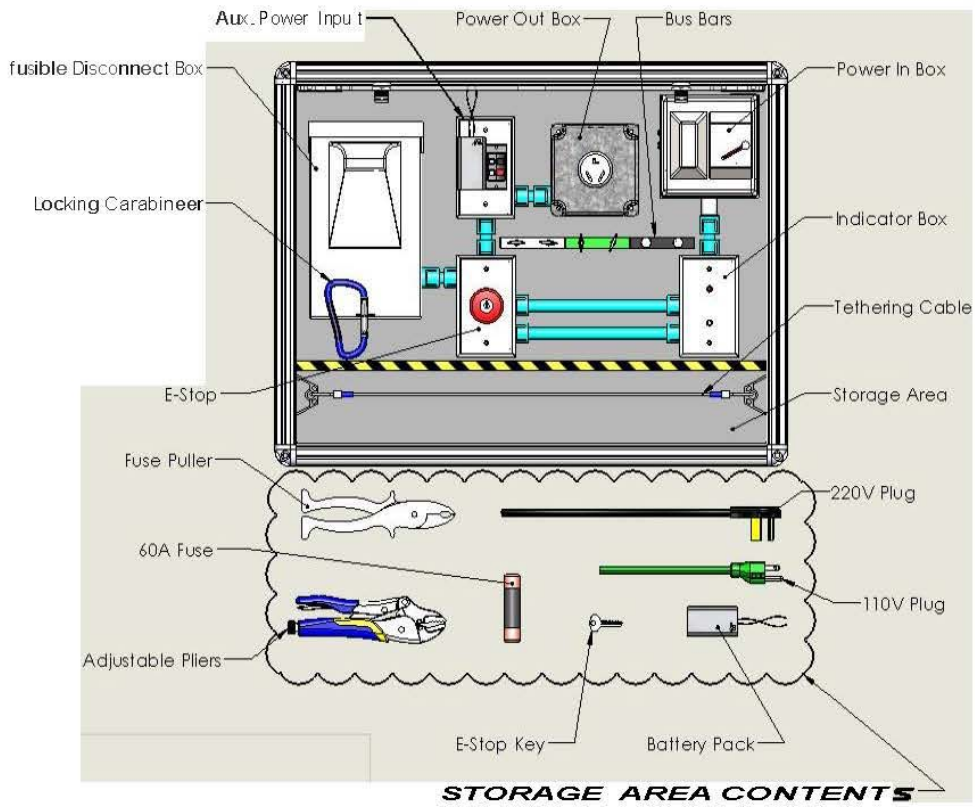
NASA EDUCATION EVA KIT-1

INTRODUCTION: Astronauts perform EVAs or Extravehicular Activities on the International Space Station. To improve efficiency and safety, astronauts train for months. Each task is scripted through what is called an EVA Checklist. These checklists are usually abbreviated and printed onto a small Cuff Checklist that the astronaut wears during space walks as shown in Figure 1. Below is the checklist for a simulated problem that an astronaut could encounter during a spacewalk. Familiarize yourself with the problem, and when ready begin the exercise.

Figure 1 Left: Checklist wrist cuff used by astronauts. Right: Mirrored wrist cuff used by astronauts. Courtesy of www.nasa.gov

Figure 2 Air Scrubber panel simulator.

EVA Schematic and Storage Area Contents:



5 4 3 2

(All items must be tethered or securely fastened during EVA operations.)

Astronaut / EVA 1 Prep: Put on the EVA Belt and Gloves

CapCom Script: "EVA 1, this is Houston Mission Control. Please be advised, an electrical short has damaged the power supply line to the Carbon Dioxide sensor on the main air-scrubbing unit for the International Space Station. There is now a fault alarm signaling and the air scrubbing unit is locked in the "OFF" status. This allows Carbon Dioxide to overtake the smaller auxiliary air scrubbing units. You must reroute the temporary battery supply power around the damaged circuit and then disable the alarm. We are a go to proceed with the rerouting procedure. Please confirm you are a "GO" to perform this procedure.

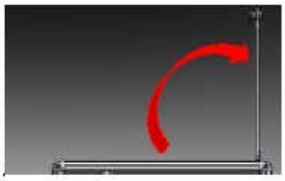
Disabling Alarm Procedure

1. On the **RIGHT** side of the **EVA Kit**, locate and use the **PANEL ACCESS KEY** to unlock the **PANEL ACCESS DOOR LOCKS.**
   CAUTION: The keys are on a tension-spring cable.

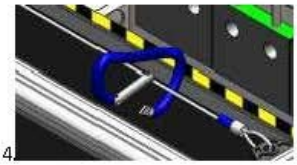
1.

2. Carefully return keys to the side of the EVA kit.

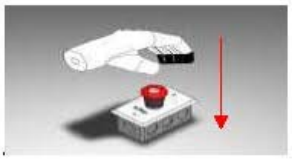
3. Insert your fingers in the **CENTER OPENING** and secure the **PANEL ACCESS DOOR** in an **OPEN** position.
   WARNING: Door can accidentally close.


3.

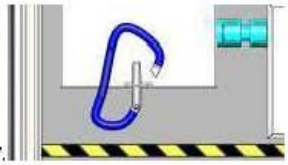
4. On your belt, use the **BLUE CARABINEER** to securely tether to the **TETHER CABLE** inside the **STORAGE**.
   CAUTION: Notice the TETHER CABLE is adjustable


4.

5. Locate the **E-STOP** button and gently press down to temporarily disable the alarm.

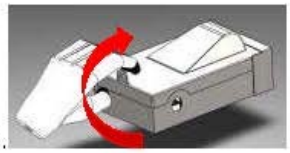

5.

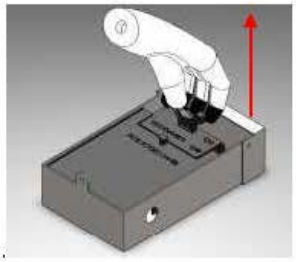
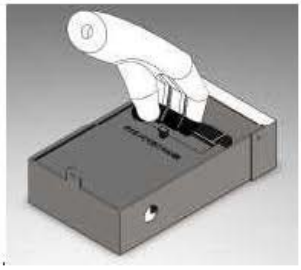
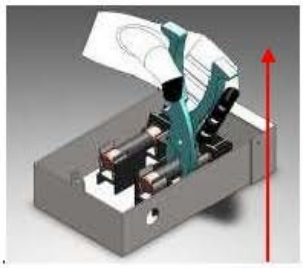
6. Locate the **FUSIBLE DISCONNECT** box and tether the **BLUE CARABINEER** to the **TETHER CABLE**.


7.

7. Remove the **BLUE CARABINEER** from the **FUSIBLE DISCONNECT** box and transfer it to **STORAGE**.

8. Open the **FUSIBLE DISCONNECT** box and secure the lid in the open position.
   CAUTION: Pull the locking tab toward **STORAGE** with the index finger while lifting the cover with the thumb.


8.

9. Locate the **BLACK DISCONNECT** and tether it to the **TETHER CABLE**. 10. Remove the **DISCONNECT** and place it in **STORAGE**. **CAUTION:** Pull up with the index and middle fingers while pushing down on the FUSE ACCESS PANEL with the thumb. 11. Tether the **FUSE ACCESS PANEL** to the **TETHER CABLE**. 12. Remove the **FUSE ACCESS PANEL** by pulling straight up. 13. Place the **FUSE ACCESS PANEL** into STORAGE. 14. Tether the **ALARM FUSE** to the **TETHER CABLE**. 15. In **Storage**, locate the **BLUE FUSE PULLER**. 16. Use the **BLUE FUSE PULLER** to remove **ONLY** the **ALARM FUSE**. **CAUTION:** Rock the **ALARM FUSE** with the **FUSE PULLER** when pulling up. 17. Return the **ALARM FUSE** and the **FUSE PULLER** to **STORAGE**.	 10.  12.  16.

18. In **STORAGE**, locate **the FUSE ACCESS PANEL** and reinstall it into the **FUSIBLE DISCONNECT** box.

19. Remove the **FUSE ACCESS PANEL** tether from the **TETHER CABLE** and stow inside.
WARNING: All tethers are under spring tension and can retract quickly.

20. In **STORAGE**, locate the **DISCONNECT** and reinstall it into the **FUSIBLE DISCONNECT** box.
CAUTION: The **DISCONNECT** must read "ON" in the upper right corner to restore conductivity.

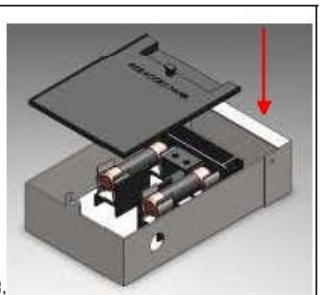
21. Remove the **DISCONNECT** tether from the **TETHER CABLE**.
WARNING: All tethers are under spring tension and can retract quickly.

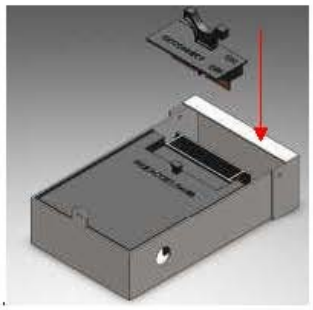
22. Close the **FUSIBLE DISCONNECT** box cover.

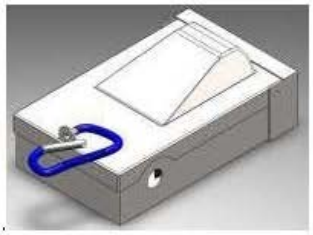
23. In **STORAGE**, use the **BLUE CARABINEER** to clip and lock the **FUSIBLE DISCONNECT** box cover.

24. Remove the **BLUE CARABINEER'**s tether from the **TETHER CABLE**.
WARNING: All tethers are under spring tension and can retract quickly.

End Of Disabling Alarm Procedure


18.


20.


23.

Rerouting Power Procedure

1. Locate the **Aux. Power Input**

2. Locate **BATTERY PACK** and tether to **TETHER CABLE**.

3. Undo the **BATTERY PACK LEADS** from the **AUX. POWER INPUT**.
   CAUTION: Depress the red and black plastic hammers on the side of the **AUX. POWER INPUT** and pull the leads straight up.

4. Remove **BATTERY PACK** from **AUX. POWER INPUT**.

5. Locate the **ON/OFF** switch on the back of the **BATTERY PACK** and switch it to the **OFF** position.

6. Place the **BATTERY PACK** into **STORAGE**.

7. In **STORAGE**, find the replacement **BATTERY PACK**.

8. Locate the **ON/OFF** switch on the back of the **BATTERY PACK** and switch it to the **ON** position.

9. Attach the replacement **BATTERY PACK** onto the **AUX. POWER INPUT** by the Velcro.

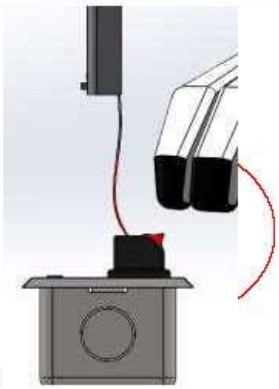
10. Insert the **BATTERY PACK** leads back into same colored ports.
    CAUTION: Depress the red and black plastic hammers on the side of the **AUX. POWER INPUT** and push leads straight into their ports.

11. Conduct a **GENTLE PULL TEST** on the wires.

12. Remove the **BATTERY PACK** tether from the **TETHER CABLE.**
    WARNING: All tethers are under spring tension and can retract quickly.

13. In **STORAGE,** locate the **GRAY 220 Volt PLUG.**

14. Install it into the **POWER OUT.**



3.

5.

12.

CAUTION: Outlet and plug mate are stiff, ensure the full engagement of the plug into the outlet.	

15. Locate the metal **BUSS BAR** and verify there are **BLACK, GREEN & WHITE BUSSES**, each with 2 openings.

16. Insert the **WHITE 220 VOLT LEAD** into the **LEFT WHITE BUSS** opening and **GENTLY TIGHTEN** the thumbscrew.
CAUTION: **DO NOT** over tighten the thumbscrew.

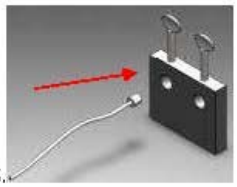
17. Insert the **GREEN 220 VOLT LEAD** into the **LEFT GREEN BUS** opening.
CAUTION: **DO NOT** over tighten the thumbscrew.

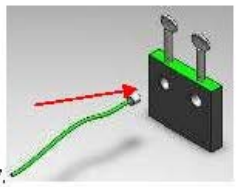
18. Insert the **BLACK 220 VOLT LEAD** into the **LEFT BLACK BUS** opening.
CAUTION: **DO NOT** over tighten the thumbscrew.

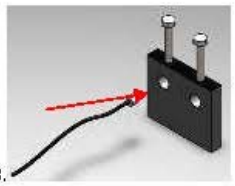
19. Make sure the **METAL LEADS** are not sticking out the **BACK** of the **BUSS BAR.**

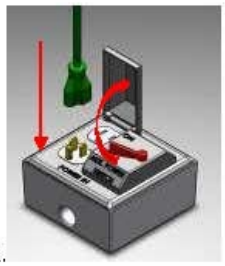
20. Conduct a Gentle **PULL TEST** on each cable.

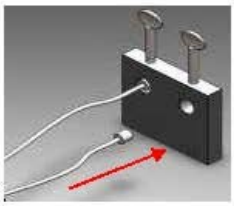
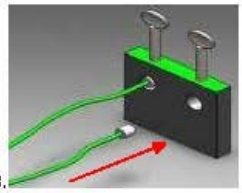
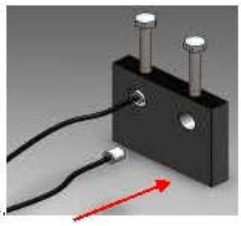
21. In **STORAGE**, locate the **110 VOLT PLUG** and install it into **POWER IN.**
CAUTION: Lift cover with one hand while installing PLUG into the outlet with the other. The lid is spring-loaded.


16.


17.


18.


21.

22. Insert the **WHITE 110 VOLT PLUG LEAD** into the RIGHT WHITE BUS opening. **CAUTION: DO NOT** over tighten the thumbscrew.	 22.
23. Insert the **GREEN 110 VOLT PLUG LEAD** into the RIGHT GREEN BUS opening. **CAUTION: DO NOT** over tighten the thumbscrew.	 23.
24. Insert the **BLACK 110 VOLT PLUG LEAD** into the RIGHT BLACK BUS opening. **CAUTION: DO NOT** over tighten the thumbscrew.	 24.
25. Conduct a Gentle **PULL TEST** on each cable	 25.
26. In **STORAGE**, locate the **E-STOP KEY**.	
27. Insert the **KEY** into the **E-STOP** and **TURN** to the **RIGHT** and the button will pop up.	 26.
28. Remove the **KEY** and place it in **STORAGE**.	

29. Locate the **AUX. POWER SWITCH** on the **POWER IN** box and switch it to the **"ON"** position. 30. Can you please confirm YES or NO that the **SYSTEM GO** indicator light is **GREEN?** 31. IF GREEN, read this script: EVA 1, this is Houston Mission Control. Congratulations. PHALCON is reporting that they are reading a successful power restoration on their console. You are a go to untether from the **TETHER CABLE** and return to space station. Mission Control out. 32. IF NOT GREEN, read this script: EVA 1, this is Houston Mission Control. PHALCON confirms and is not able to report a successful power restoration on their console. EVA-1, please be advised that we have prepared some trouble-shooting steps for you to conduct on a future spacewalk.	 28. Done.

Clean up

1. Engage the E-Stop button
2. Disconnect all leads for the 110 Volt & 220 Volt Plug LEADS
3. Unplug the 110V & 220V PLUGS and stow the plugs in the STORAGE AREA
4. Switch the BATTERY PACKS switch to the "OFF" position
5. Ensure all remaining items are either installed securely or tethered in the STORAGE AREA
6. Close the PANEL ACCESS DOOR and lock the door with the keys from the tethering belt to secure the area
7. Return tethering belt and gloves to the instructor

Troubleshooting

PROBLEM	REASON	CHECK POINTS
Alarm sounds when AUX. POWER SWITCH is switched on.	The wrong fuse was removed.	Replace the fuse that was taken out and remove the ALARM FUSE.
Nothing happens when AUX. POWER SWITCH is switched on.	There is no power going to the circuit.	-BATTERY PACK is switched "OFF" -BATTERY PACK leads are not inserted into their ports -220V PLUG or leads where not installed correctly -110V PLUG or leads where not installed correctly -AUX. POWER SWITCH is in the "OFF" position -E-STOP is still engaged -Batteries in the BATTERY PACK are depleted