|  |  |
| --- | --- |
| Group 12 | *Exfiltrate & Disseminate Death Star Plans* |
| **Major:** | **Team members:** |
| CEG | Kalen Tullis |
| ITC | Simran Cheema |
| CS | Divyesh Ambaliya |
| CS  EE | Liliane Owens  Nathaniel Rieder |

**Test and Evaluation Master Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement | Test Method | Evaluation Method | Threshold | Objective |
| 1 | Raspberry Pi is in use | Observation | The Pi displays powered LED. |  |
| 2 | Server is in use | Observation | Server can boot to a desktop |  |
| 3 | Mobile app is in use | Observation | Mobile app is accessible | Display grid-images |
| 10.1 | Image recognition Libraries installed | Successful run of python script testing imports | Python script runs with no errors | Python can recognize all installed libraries. |
| 10.2 | Image recognition can run with any image of assumed format | Give software an image of assumed image. | Software identifies the image is of assumed format. | Software can identify all images of assumed format. |
| 10.3 | Image recognition can recognize a single death star image | Give software a Death Star image. | Software identifies the image as a Death Star image. | Software can always determine when an image contains the Death Star. |
| 10.4 | Image recognition can recognize a single non-death star image | Give software a non-Death Star image. | Software identifies the image as a non-Death Star image. | Software can always determine when an image does not contain a Death Star. |
| 10.5 | Image recognition can identify 10 death star images out of 100 images | Give software 100 photos that contains 10 Death star images | Model can recognize 80% of Death Star images | Model can recognize 100% of Death Star images. |
| 11 | Server can crop images | Software will crop any image. | The output is a smaller image | The server will crop images to our desired size. |
| 11.1 | Server can crop images based on a red circle | Software will crop image of assumed format and content | The output is content outlined in a red circle | The script will save multiple cropped images if there are multiple red circles. |
| 20.1 | Transceivers are recognized by Pi and Server | Run software to get transceiver status | Transceiver is recognized by kernel | Pi and Server can recognize the state of their own transceivers |
| 20.2 | Data can be encoded in carrier wave | Script can encode the letter “a” | Script on transmitter says the data was sent. | The software can successfully convert data into a transmittable radio wave. |
| 20.3 | Receivers can decode carrier wave | Receiver collects the sent “a” data | Receiver receives data from the transmitter | The Pi can receive data transmitted from the rebel server and vice versa. |
| 20.4 | Pi and Server can transfer any data | Pi and Server will transmit an image to each other. | A test image is transmitted. | The Raspberry Pi can transmit any image to the rebel server. |
| 20.5 | Pi and Server can transfer an image of assumed format | Transmission script works to transmit a single death star image | Received image is viewable on server | Transmission script can transmit a set of death star images |
| 20.6 | Communication can occur in both pi-to-server and server-to-pi | Observation of transmission output | The same image can be sent back and forth |  |
| 21 | Pi and Server can communicate in separate rooms | Observation | Pi and Server are placed in opposite rooms | Not to be detected by the guards while transmitting the data. |
| 22 | Pi and Server can communicate while separated by more than 5 meters | Observation | At least 5 meters | Transmit over 5-to-10-meter distance |
| 23 | Pi will transfer 1 full image in less than 1 minute | Timing image transmission using a stopwatch | Images take at most 1 minute per image transmission | Images take at most 30 seconds per image transmission |
| 23.1 | Pi will transfer 10 full images continuously. | Timing speed of complete transmission. | The 10 images are sent within 10 minutes. | The complete image transmission takes no more than 10 minutes. |
| 24 | Already tested in standards table |  | The standards table is tested successfully. | The final design is safe to use and operate. |
| 30.1 | Pi and Server download the same checksum package | Observation | Pi and Server use same checksum algorithm |  |
| 30.2 | Server can transmit checksum to Pi | Pi and Server can transmit “hello world” to each other | Hello World, is transmitted | Image checksum is transferred |
| 31.1 | Pi can compare checksums | Observation | Pi saves both Server and Pi checksums | A script determines the confirmation of lossless data transmission from the server |
| 31.2 | Pi can retransmit images | Tested in 20.3 |  | Pi is signaled to retransmit images based on checksums |
| 31.3 | Server can recognize failed images | Observation of resulting images | Server replaces latest image | Server can continue receiving images while computing checksum of previous image. Failed images are retransmitted at the end of transmission. |
| 40.1 | Pi can encrypt data | Encryption library can encrypt a text file | Text file is encrypted | Images are encrypted |
| 40.2 | Server can support the same encryption library | Same test as 40.1 |  |  |
| 50 | Mobile app will launch on Android phone OS | Observation | Mobile app will launch hello world on Android phone OS | Mobile apps are usable on Android systems. |
| 51 | Mobile app will display images | Observation | Mobile app displays hello world | Mobile app displays images |
| 60 | Mobile app will display images in a grid | Observation | Mobile app displays rectangular components in a grid | Mobile app displays images in a grid |
| 61 | Mobile app displays cropped images from server | Observation | Mobile app displays text from server input | Mobile app displays Death Star images from the server |
| 70 | Server hosts all mobile app content | Observation | Mobile app content is dependent on server | The server will display the content for the mobile app |
| 80 | Tested in Constraint 40 |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Constraint | Test Method | Evaluation Method | Threshold | Objective |
| 10 | Already tested in Requirement 23 |  | 10 minutes | Data transmission takes less than 10 minutes. |
| 20 | Already tested in Requirement 22 |  | 5 meters | Data can be transmitted across a short distance. |
| 30 | Already tested in standards table |  | The standards table is tested successfully. | The final design is safe to use and operate. |
| 40 | Record expenses in the budget. | Observation | $300 | The design costs less than $300 |
| 50 | Already tested in Requirement 50 |  | App on mobile phone | App displays 10 images |
| 60 | Already tested in Requirement 20 |  | A discrete signal is used | Data is transmitted using RF transceiver |
| 70 | See if the Pi has a physical connection to the server. | Observation | 0 physical connections | There are no physical connections between Pi and server. |
| 80 | Check the OS of the device | Observation | Linux OS in use | The Pi and server use a Linux OS |
| 90 | Check available USB ports on the Pi | Observation | 1 open USB port | The Pi will have at least 1 USB port open. |
| 100 | Already tested in Requirement 40 |  | All data between pi and server is encrypted. | All data between pi and server is encrypted. |
| 110 | Already tested in Requirement 70 |  | The app is hosted on the server | The app is hosted on the server. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Standard | Test Method | Evaluation Method | Threshold | Objective |
| 10 | Document review to align with IEEE1900.1-2019 terms | Glossary check for terminology compliance | Complete use of IEEE-compliant terms | Consistency in terminology across all project documentation |
| 20 | Validate data storage and file transfer between the Raspberry Pi and the Rebel Server. | Perform file transfer tests to verify interoperability and data storage. | Successful data transfer between systems | Smooth, error-free file transfers without interoperability issues. |
| 30 | Review and verify encryption protocols on the Raspberry Pi and Rebel Server. | Confirm that all data transmissions are securely encrypted per ISO/IEC 27002:2013. | Data encryption that prevents unauthorized access. | Complete encryption compliance, ensuring robust data security |
| 40 | Use transmission devices in standard range of operations. | Check that the device is operating in the standard frequency range shown on the data sheet. | All frequencies fall within the safe 3 kHz to 300 GHz range. | Full compliance with RF safety guidelines, ensuring no unsafe frequencies are used. |
| 50 | Test RF Transceiver configurations to validate WPAN data transmission at 2.4 GHz. | Ensure RF Transceiver transmits data using low power and supported modulation formats. | Functioning WPAN within the 2.4 GHz range. | High-quality WPAN transmission with low interference and stable data rates. |
| 60 | Conduct tests on the RF Transceiver system to ensure it meets LR-WPAN transmission standards. | Use low-power, short-range communication protocols to test data transfer stability. | Reliable, low-power transmission with no loss of data. | Consistent, high-quality LR-WPAN communication across the required distance. |
| 70 | Assess the RF Transceiver for compatibility with ISO/IEC/IEEE 8802-11:2018 to verify local and metro connectivity standards. | Check for transmission compatibility over multiple network bands. | RF Transceiver operates within local and metro connectivity requirements. | Reliable and standardized network communication across various distances. |
| 80 | Test JPEG, WebP, and PNG image formats on the Raspberry Pi to analyze compression and decompression time. | Measure transmission time and compression ratio for each image format. | Compression achieves fast, lossless image transmission. | Optimal balance of high compression ratio and low transmission time. |