Test Plan and Results

Overall Test Plan Strategies

Our testing plan is divided into two phases, the first for component-level validation and the second to test full system integration. During the first phase, each component is individually tested under typical and edge-case conditions to ensure proper functionality. The second phase involves testing the integration of these components to ensure a seamless environment and to evaluate performance of the system.

Test Case Descriptions

User Profile Data Tests
UP1.1User Profile Data Test 1
UP1.2 This test will verify that the data a user enters in the profile section will be locally saved.
UP1.3 A user will enter their data into the profile section of the application. Once the page is saved, the data is locally stored and persists when the application/device shuts down.
UP1.4 Inputs: Sample user profile data including preferred name, height, and wingspan.
UP1.5 Outputs: Properly stored and retrievable user data
UP1.6 Normal
UP1.7 Blackbox
UP1.8 Functional
UP1.9 Unit Test
UP1.10 Results: User profile data was properly saved and persisted after application and device shut down.

UP2.2 User Profile Data Test 2
UP2.2 This test will validate that any data updates in the profile section are properly handled and overwrite previously saved data.
UP2.3 A user will enter new data into the profile section containing existing data. Once the page is saved, the new data is locally stored and overwrites previously stored data.
UP2.4 Inputs: Sample user profile data including preferred name, height, and wingspan.
UP2.5 Outputs: Properly stored and retrievable user data
UP2.6 Normal
UP2.7 Blackbox
UP2.8 Functional
UP2.9 Unit Test
UP2.10 Results: Previous user profile data was properly overwritten with the new data entered.

Camera Integration Tests
CAM1.1 Camera Integration Test 1
CAM1.2 This test will ensure the camera successfully integrates with the application to capture images for pathfinding.
CAM1.3 A user will open the in-app camera functionality to take an image of a bouldering wall. The captured image is properly submitted and processed for use.
CAM1.4 Inputs: Images of a bouldering wall captured using the device camera
CAM1.5 Outputs: Images are processed correctly and submitted to blob detection for

pathfinding
CAM1.6 Normal
CAM1.7 Blackbox
CAM1.8 Functional
CAM1.9 Integration Test
CAM1.10 Results: The camera page allowed for a user to take an image of a bouldering wall in the app. Image captured on the camera page was sent through blob detection and processed properly.

Image Gallery Integration Tests
GAL1.1 Image Gallery Integration Test 1
GAL1.2 This test will confirm the device's image gallery successfully integrates with the application to upload images for pathfinding.
GAL1.3 A user will open the in-app gallery functionality to upload a previously captured image of a bouldering wall. The chosen image is properly submitted and processed for use.
GAL1.4 Inputs: Images of a bouldering wall stored in device's image gallery
GAL1.5 Outputs: Images are processed correctly and submitted to blob detection for pathfinding
GAL1.6 Normal
GAL1.7 Blackbox
GAL1.8 Functional
GAL1.9 Integration Test
GAL1.10 Results: The gallery page opened the device's gallery in the app and sent the selected image through blob detection.

Blob Detection Tests
BD1.1 Blob Detection Test 1
BD1.2 This test will verify that the system accurately detects blobs in an image to identify holds on a bouldering wall.
BD1.3 After a user submits an image, it will be processed through blob detection to identify bouldering holds within the image. Detected blobs will accurately reflect the image and create a grid for the path-finding algorithm.
BD1.4 Inputs: Sample image of a bouldering wall
BD1.5 Outputs: Correctly identified blobs corresponding to holds
BD1.6 Normal
BD1.7 Functional
BD1.8 Blackbox
BD1.9 Unit Test
BD1.10 Results: Sample image was processed properly with blob detection finding all holds of the selected color.

BD2.1 Blob Detection Test 2
BD2.2 This test will evaluate the application's ability to detect blobs in noisy or low-light images to identify holds on a bouldering wall.
BD2.3 We will process images with poor lighting or noise using blob detection. We will then compare detected holds to the results from a high-quality image of the same bouldering wall to evaluate accuracy.
BD2.4 Inputs: Noisy or low-light images of a bouldering wall
BD2.5 Outputs: Majority of blobs are accurately detected

BD2.6 Abnormal
BD2.7 Functional
BD2.8 Blackbox
BD2.9 Unit Test
BD2.10 Results: Majority of the holds in a noisy image were detected, however, size detected was not entirely accurate. Background colors seem to sometimes integer with blob detection if it corresponds with the user selected color.

BD3.1 Blob Detection Test 3
BD3.2 This test will validate that the system accurately detects blobs in an image containing multiple paths.
BD3.3 After a user submits an image, they will have the option to select the color of the holds they would like to use for pathfinding. Detected blobs will accurately reflect the holds in the image corresponding to the selected color and create a grid for the path-finding algorithm.
BD3.4 Inputs: Sample image of a bouldering wall with multiple paths
BD3.5 Outputs: Correctly identified blobs corresponding to holds of a user-selected color
BD3.6 Normal
BD3.7 Functional
BD3.8 Blackbox
BD3.9 Unit Test
BD3.10 Results: Sample image containing multiple colored pathways was processed properly and the holds corresponding to the selected color were properly identified.

Hold Selection Tests
HS1.1 Hold Selection Test 1
HS1.2 This test will verify that a user can assign a hold difficulty attribute to the detected holds before sending the processed image for pathfinding.
HS1.3 After a user uploads an image of a bouldering wall for blob detection, they will be prompted with the option to assign hold difficulty to the detected holds.
HS1.4 Inputs: Sample image of a bouldering wall with multiple hold types
HS1.5 Outputs: Holds are correctly tagged with user-assigned types
HS1.6 Normal
HS1.7 Functional
HS1.8 Blackbox
HS1.9 Unit Test
HS1.10 Results: After blob detection, ‘list hold difficulty’ is selected and a list of holds opens. Selecting a hold allows to change the assigned difficulty, which is saved when modified.

A* Algorithm Tests
ALG1.1 A* Algorithm Test 1
ALG1.2 This test will ensure that the A* algorithm produces a valid path between the start and end holds.
ALG1.3 We will process a sample image of a bouldering wall through blob detection and submit it to the A* algorithm to verify its ability to produce a path between the start and end holds.
ALG1.4 Inputs: Sample image of a bouldering wall processed through blob detection
ALG1.5 Outputs: Valid path between start and end holds in the form of steps to be completed (ex: Left Foot, Hold 3, Hold 5)
ALG1.6 Normal

ALG1.7 Whitebox
ALG1.8 Functional
ALG1.9 Unit Test
ALG1.10 Results: Valid path between the assigned start and end holds was found for each limb.

ALG2.1 A* Algorithm Test 2
ALG2.2 This test will validate the algorithm's ability to handle cases where there is no valid path.
ALG2.3 We will process a sample image of a bouldering wall with no valid path through blob detection and submit it to the A* algorithm to check for error handling.
ALG2.4 Inputs: Sample image of bouldering wall with no valid path processed with blob detection
ALG2.5 Outputs: System receives an 'error' or 'no valid path found' message
ALG2.6 Abnormal
ALG2.7 Whitebox
ALG2.8 Functional
ALG2.9 Unit Test
ALG2.10 Results: System gives a message stating that there is no valid path found in the provided image.

ALG3.1 A* Algorithm Test 3
ALG3.2 This test will evaluate the algorithm's performance in producing a path or error message.
ALG3.3 We will process various sample images, including ones with paths of differing difficulty and ones with no valid path. Then, we will measure the time taken to generate a path or error message after submitting the processed image to the A* algorithm.
ALG3.4 Inputs: Various sample images: some containing paths of differing difficulty and some with no valid path processed with blob detection
ALG3.5 Outputs: System generates the valid path in correct format or error message in a timely manner
ALG3.6 Normal
ALG3.7 Whitebox
ALG3.8 Performance
ALG3.9 Unit Test
ALG3.10 Results: System generates path quickly, for most images in just a couple seconds. Hold difficulty is properly processed through the algorithm, resulting in unique paths from the same image when modified.

Pathfinding Tests
TH1.1 Pathfinding Test 1
PTH1.2 This test will test the integration between the three components of the pathfinding algorithm: blob detection, hold selection, A* algorithm.
PTH1.3 A user will submit a sample image, assign hold types, and execute the pathfinding algorithm. They can then view the generated path steps with the option to save the path or they will receive a 'no valid path found' message.
PTH1.4 Inputs: Sample image of a bouldering wall
PTH1.5 Outputs: Generated path or 'no valid path found' message is displayed to the user
PTH1.6 Normal
PTH1.7 Blackbox

PTH1.8 Functional
PTH1.9 Integration
PTH1.10 Results: System gives a message stating that there is no valid path found in the provided image.

PTH2.1 Pathfinding Test 2
PTH2.2 This test will verify that produced paths are displayed to the user in a clear manner.
PTH2.3 A user will submit a sample image to the system to generate a path. The results will be displayed to the user in the form of a list of steps and/or a visualization of the path on the user interface.
PTH2.4 Inputs: Sample image of a bouldering wall
PTH2.5 Outputs: Clear and seamless display of path steps and/or visualization on the UI
PTH2.6 Normal
PTH2.7 Blackbox
PTH2.8 Functional
PTH2.9 Integration
PTH2.10 Results: Path steps are displayed incrementally using ‘next’ and ‘previous’ buttons. Full path is displayed when ‘view final path’ is selected. Steps are color coded by limb.

PTH3.1 Pathfinding Test 3
PTH3.2 This test will evaluate the system’s performance in finding a path, if possible, and displaying the results to the user.
PTH3.3 We will process various sample images, including ones with paths of differing difficulty and ones with no valid path. Then, we will measure the time taken to generate the appropriate results and display them to the user.
PTH3.4 Inputs: Various sample images: some containing paths of differing difficulty and some with no valid path
PTH3.5 Outputs: Generated path or ‘no valid path found’ message is displayed in a timely manner
PTH3.6 Normal
PTH3.7 Blackbox
PTH3.8 Performance
PTH3.9 Integration
PTH3.10 Results: Path steps are displayed correctly within about a second using the display option buttons when a proper image is supplied. System gives a message stating that there is no valid path found in the provided image for images that do not contain a proper path.

Saved Paths Tests
SP1.1 Saved Paths Test 1
SP1.2 This test will verify that a user can save a produced path.
SP1.3 We will generate a path using a sample image and save it using the ‘Save Path’ functionality. We will then confirm the path results and original image are stored locally
SP1.4 Inputs: Generated path from pathfinding algorithm
SP1.5 Outputs: Path is successfully saved locally
SP1.6 Normal
SP1.7 Whitebox
SP1.8 Functional
SP1.9 Unit Test

SP1.10 Results: Path results and original image are saved in the local database when selected and a name can be given (defaults to the data/timestamp).

SP2.1 Saved Paths Test 2
SP2.2 This test will verify that previously saved paths are accessible to the user through the ‘Saved Paths’ page.
SP2.3 A user can navigate to the ‘Saved Paths’ page and view all previously saved paths. After selecting a previously saved path, the data will be displayed to the user.
SP2.4 Inputs: Previously saved path
SP2.5 Outputs: Path is successfully loaded and displayed to user
SP2.6 Normal
SP2.7 Blackbox
SP2.8 Functional
SP2.9 Unit Test
SP2.10 Results: The ‘Saved Paths’ page displays all locally saved paths. Selecting a saved path opens the display page with the display option buttons to walk through the steps or view the final path.

Full System Tests
FS1.1 Full System Test 1
FS1.2 This test will verify that all system components work together under normal operating conditions.
FS1.3 We will perform all steps from user profile entry to pathfinding and visualization. We can then verify end-to-end functionality of all components.
FS1.4 Inputs: Full system workflow inputs
FS1.5 Outputs: Successful pathfinding and visualization without errors
FS1.6 Normal
FS1.7 Blackbox
FS1.8 Functional
FS1.9 Integration
FS1.10 Results: Full system works with all inputs and does not provide any errors.

FS2.1 Full System Test 2
FS2.2 This test will evaluate the system’s performance under high data load.
FS2.3 We will simulate a user saving all profile data and multiple generated path data. We will observe how the system responds to this behavior.
FS2.4 Inputs: High volume of saved data
FS2.5 Outputs: System remains responsive and performs accurately
FS2.6 Abnormal
FS2.7 Blackbox
FS2.8 Performance
FS2.9 Integration
FS2.10 Results: Full system works with all inputs and does not provide any errors, system remains responsive and performs the correct behavior.

Overall Test Case Matrix

Test Case ID	Normal/Abnormal	Black/Whitebox	Functional/Performance	Unit/Integration
UP1	Normal	Blackbox	Functional	Unit
UP2	Normal	Blackbox	Functional	Unit
CAM1	Normal	Blackbox	Functional	Integration
GAL1	Normal	Blackbox	Functional	Integration
BD1	Normal	Blackbox	Functional	Unit
BD2	Abnormal	Blackbox	Functional	Unit
BD3	Normal	Blackbox	Functional	Unit
HS1	Normal	Blackbox	Functional	Unit
ALG1	Normal	Whitebox	Functional	Unit
ALG2	Abnormal	Whitebox	Functional	Unit
ALG3	Normal	Whitebox	Performance	Unit
PTH1	Normal	Blackbox	Functional	Integration
PTH2	Normal	Blackbox	Functional	Integration
PTH3	Normal	Blackbox	Performance	Integration
SP1	Normal	Whitebox	Functional	Unit
SP2	Normal	Blackbox	Functional	Unit
FS1	Normal	Blackbox	Functional	Integration
FS2	Abnormal	Blackbox	Performance	Integration