Targets:

- Enable the robot to follow the user
    - Need: Robot will follow the person by tracking their phone
    - Need: Robot will follow user unless physically incapable
    - Need: Decision left to be made by SD team to follow user outside of predefined area
    - Function: followUser - robot maintains a set distance, z, between itself and the user while following
        - Metrics: distance between robot and user
        - Value: z (cm), z_min = 100cm
        - Justification: The robot shouldn't be so far away from the user that it loses track of them or becomes hard to follow or too close to the user where it becomes uncomfortable or an inconvenience
        - Test: make the robot follow a user taking turns and going around corners without the robot losing track of the user and navigating safely
        - Importance: extreme
    - Function: centerUser - robot adjusts motor output to keep user in the center of the frame
        - Metrics: offset of the user being tracked from the center of the frame
        - Value: x (pixels) = 0, y (pixels) = 0
        - Justification: The robot needs to keep track of the user and should follow directly behind them not going all over the place
        - Test: image analysis shows the user in the center frame of robot camera throughout the path followed
        - Importance: extreme
- Enable Robot to scan environment for obstacles or hazards
    - Need: Robot detect obstacles
    - Need: Control module will interact with different types of cameras for object detection
    - Need: Robot will follow user unless physically incapable
    - Need: Notify user if unsafe to follow

    - Function: avoidObstacles - image detection and path finding algorithm to detect and path around obstacles that obstruct the robot's path to the user
        - Metric: Highlight potential obstructions detected in the camera feed with pathfinding analysis
        - Value: num_obstacles (integer) = 0, robot_path_clear (boolean) = true
        - Justification: The robot needs to be able to navigate around obstacles in its path to follow the user, shouldn't be getting stuck all the time
        - Test: image analysis shows obstructions in-between the service robot and user; service robot moves around obstruction
        - Importance: extreme
    - Function: emergencyStop - brings the robot to a complete stop when about to collide with an unexpected hazard
        - Metric: Robot comes to a full stop
        - Value: time_to_stop (seconds) < 1s, distance_to_stop (cm) < 50cm
        - Justification: To keep everyone around the robot safe, prevent damage to the robot itself and to prevent liability
        - Test: service robot stops without colliding with identified hazard
        - Importance: extreme
- User is able to connect to the robot using Bluetooth through an app
    - Need: The robot will follow the user by tracking their phone
    - Need: Decision left to be made by SD team to determine what hardware to use
    - Need: Notify user if unsafe to follow
    - Function: connectToRobot
        - Metric: Service robot connects to app via Bluetooth
        - Value: robot_id (name/string) = cart1/robot1 (generated by owner)
        - Justification: To allow the user to control the robot with a connected device
        - Test: robot connects to user device; user can see robot is connected through mobile app
        - Importance: extreme

- Function: enableAutonomousMode – allows user to turn autonomous mode on and off using the app
    - Metric: Service robot enters autonomous navigation mode to follow user
    - Value: autonomousModeOn (boolean) = true/false
    - Justification: This is needed to control whether or not the robot navigates the environment autonomously
    - Test: the user presses autonomous mode on their mobile app and the robot begins to follow the user
    - Importance: extreme
- Function: enableManualControl - switches the robot from autonomous mode to manual mode
    - Metric: User can move the robot through physical interaction or virtual joystick
    - Value: autonomousModeOn (boolean) = true/false
    - Justification: User must be able to move the robot themselves
    - Test: when the user presses manual mode the robot stops following them and responds to manual input; user can make the robot stop, move forward, backwards, left and right
    - Importance: moderate
- Function: virtualJoystick - allows the user to control the motors of the robot manually
    - Metric: User has manual control over the motors using the virtual joystick
    - Value: speed (meters per second) < 10 mps, direction (string)forward/backward/right_turn/left_turn, stop (bool) = true/false
    - Justification: User needs to be able to manually input instructions to the motor system.
    - Test: the user presses manual mode, and the virtual joystick appears on the mobile app; the user can move the robot forward, backwards, left, and right by moving the virtual joystick forward, backwards, left, and right
    - Importance: moderate
- Function: crashDetection - prevents the user from crashing into people or objects while using the bot in manual mode

- Metric: Robot evaluates the safety of instructions provided by the user in manual mode and chooses to ignore or follow them
- Value: will_collide (boolean) = true/false
- Justification: Prevents the user from harming others or damaging the robot using the motor system by evaluating if the provided instructions are safe to follow
- Test: through software analysis, the robot detects collisions while the user manually moves it; the robot indicates through an app notification a crash is imminent and either stops or ignores the command
- Importance: moderate

Summary

There are three targets our project must accomplish, in no particular order they are: Enable the robot to follow the user, enable robot to scan environment, user is able to connect to robot. Enabling the robot to follow the user will require the followUser function, and centerUser function. This target will satisfy the customer needs of: robot will follow user by tracking their phone, Robot will follow user unless physically unable to, and to follow user outside of predefined area once (if necessary). Enabling the robot to scan the environment will require the avoidObstacles function, and emergencyStop function. This target will satisfy the customer needs of: Robot detecting obstacles, Control module will interact with different types of cameras, Robot will follow user unless physically incapable, and notifying user if unable to follow. User being able to connect to robot will require the connectToRobot function, enableAutonomousMode function, enableManualControl function, virtualJoystick function, crashDetection function. This target will satisfy the customer needs of: the robot following the user by tracking their phone, The robot being able to notify the user that the robot cannot follow because the path is unsafe.