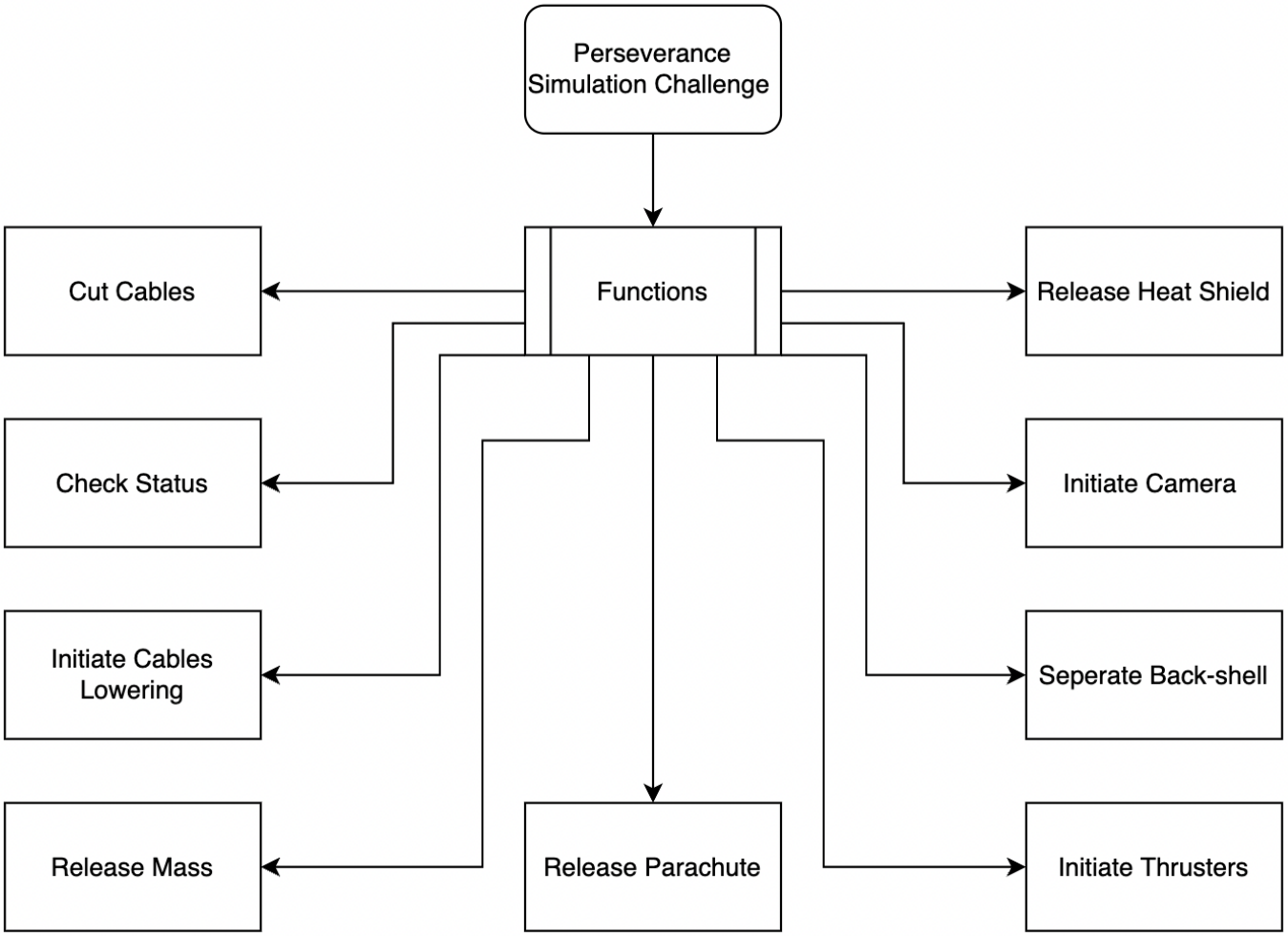
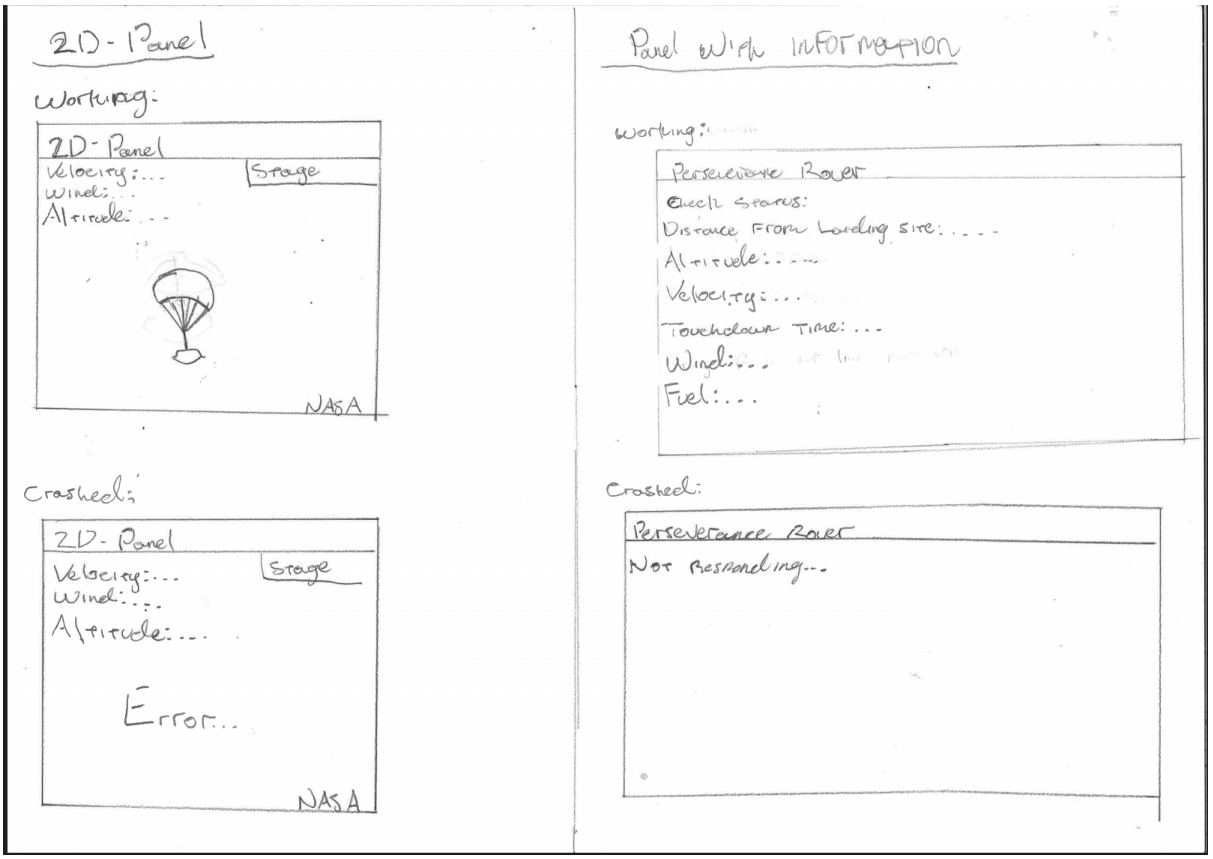
Criterion B: Design

Word Count: 0

DELIVERABLE 1: APPLICATION OVERVIEW FLOWCHART

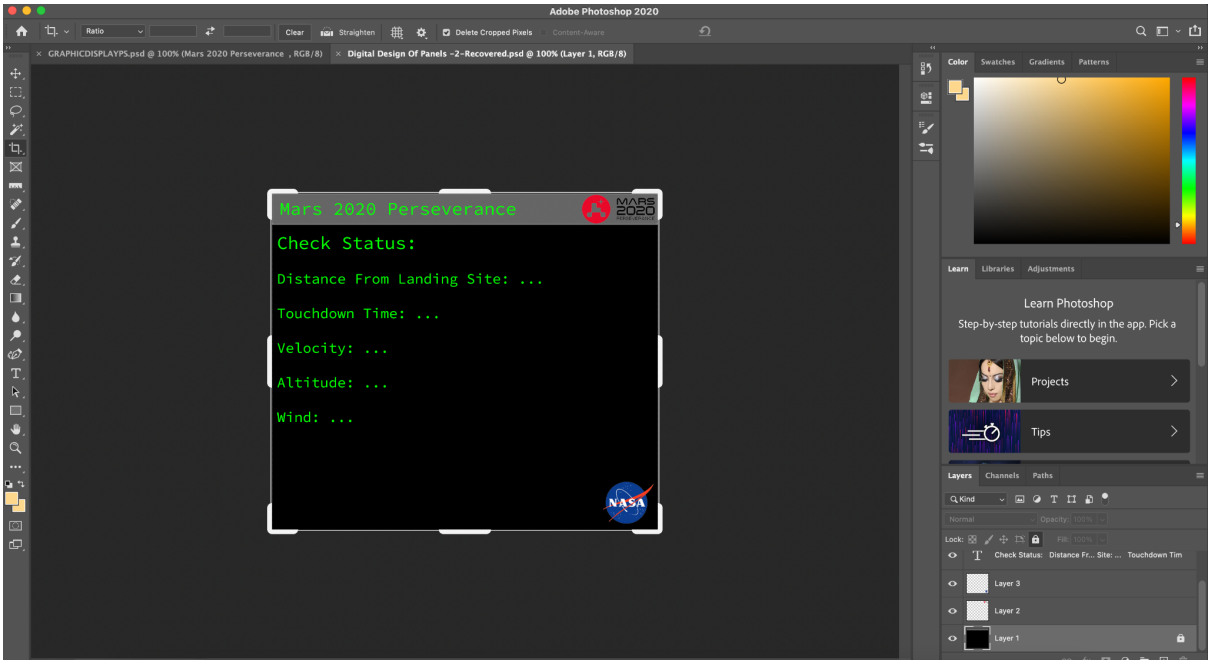


DELIVERABLE 2: THE SKETCHES

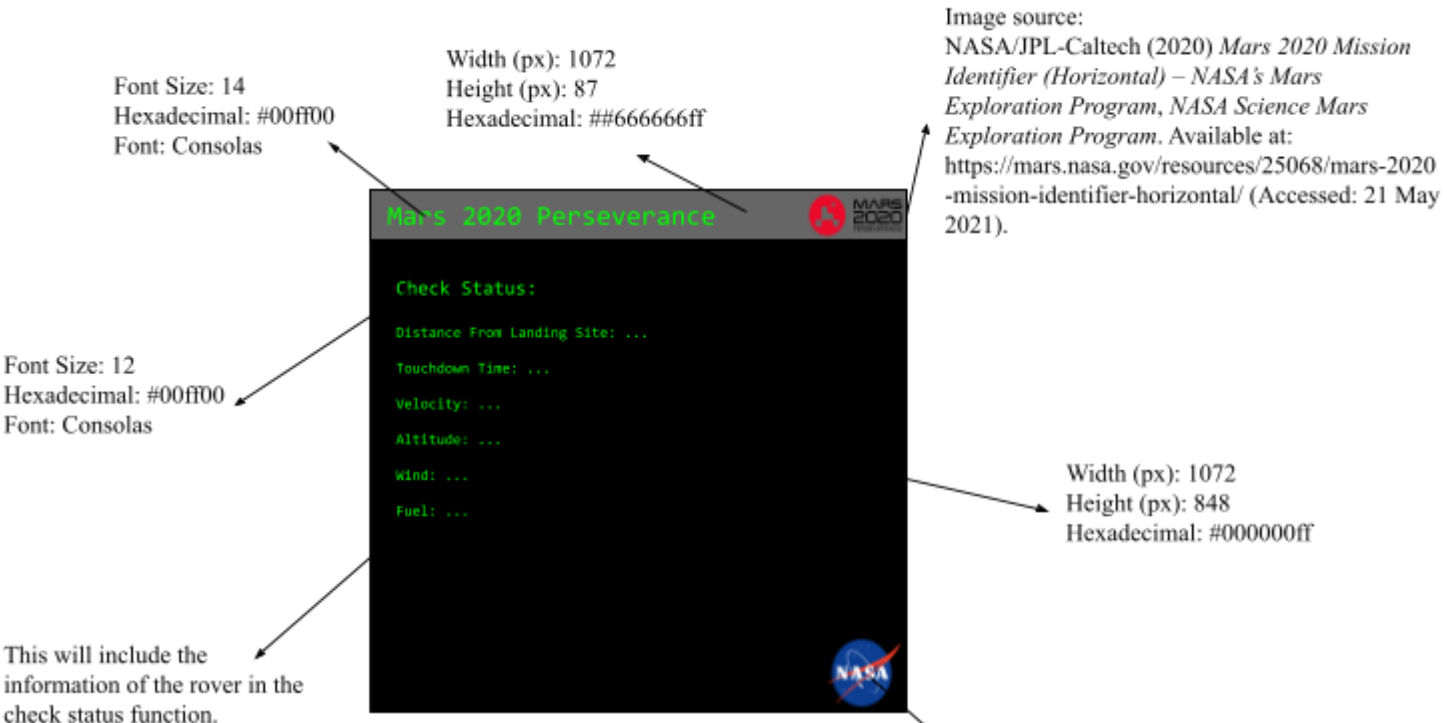


DELIVERABLE 3: DIGITAL SKETCHES WITH ANNOTATION (EXAMPLES)

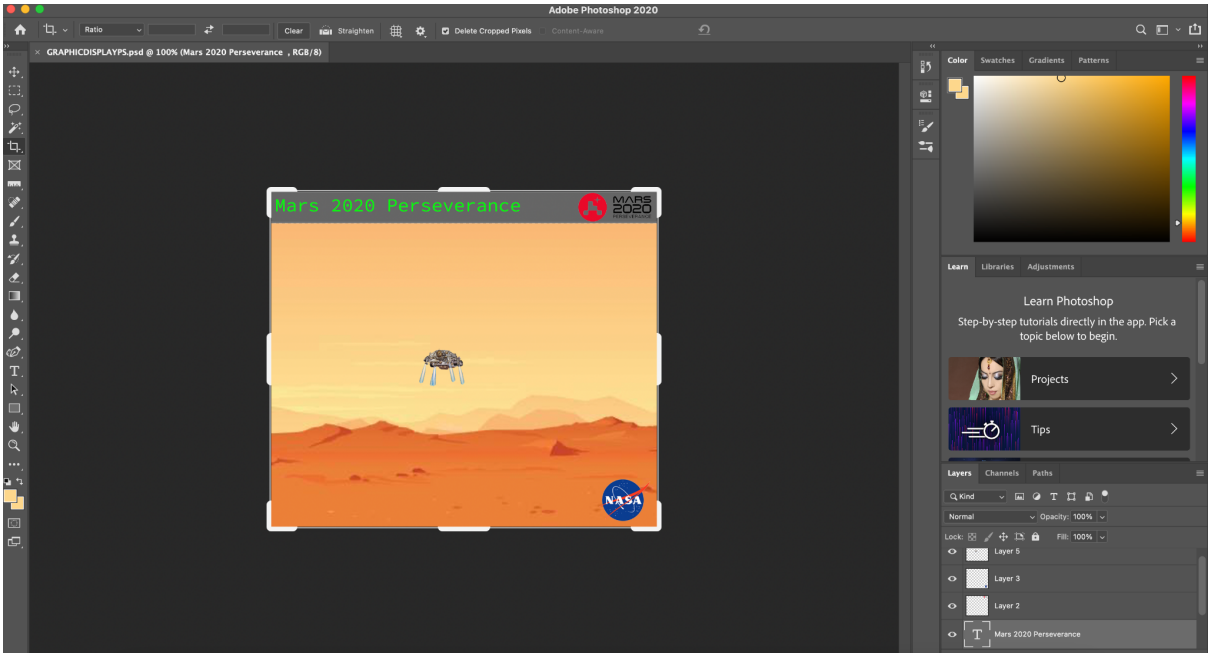
Design of Communication Panel (example):



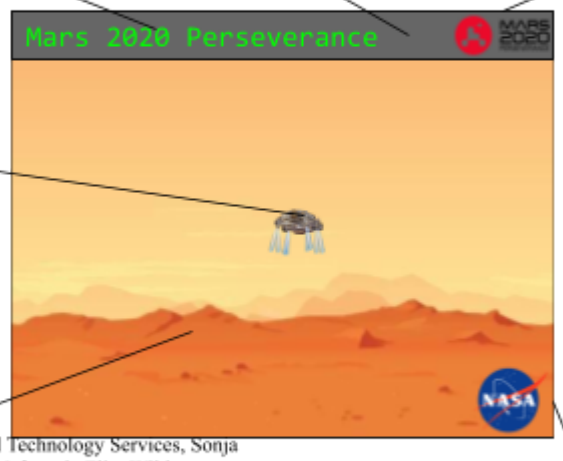
Annotation of Communication Panel:



Design of 2D Panel (example):



Annotation of 2D Panel:



Font Size: 14
Hexadecimal: #00ff00
Font: Consolas

Width (px): 1072
Height (px): 87
Hexadecimal: ##666666ff

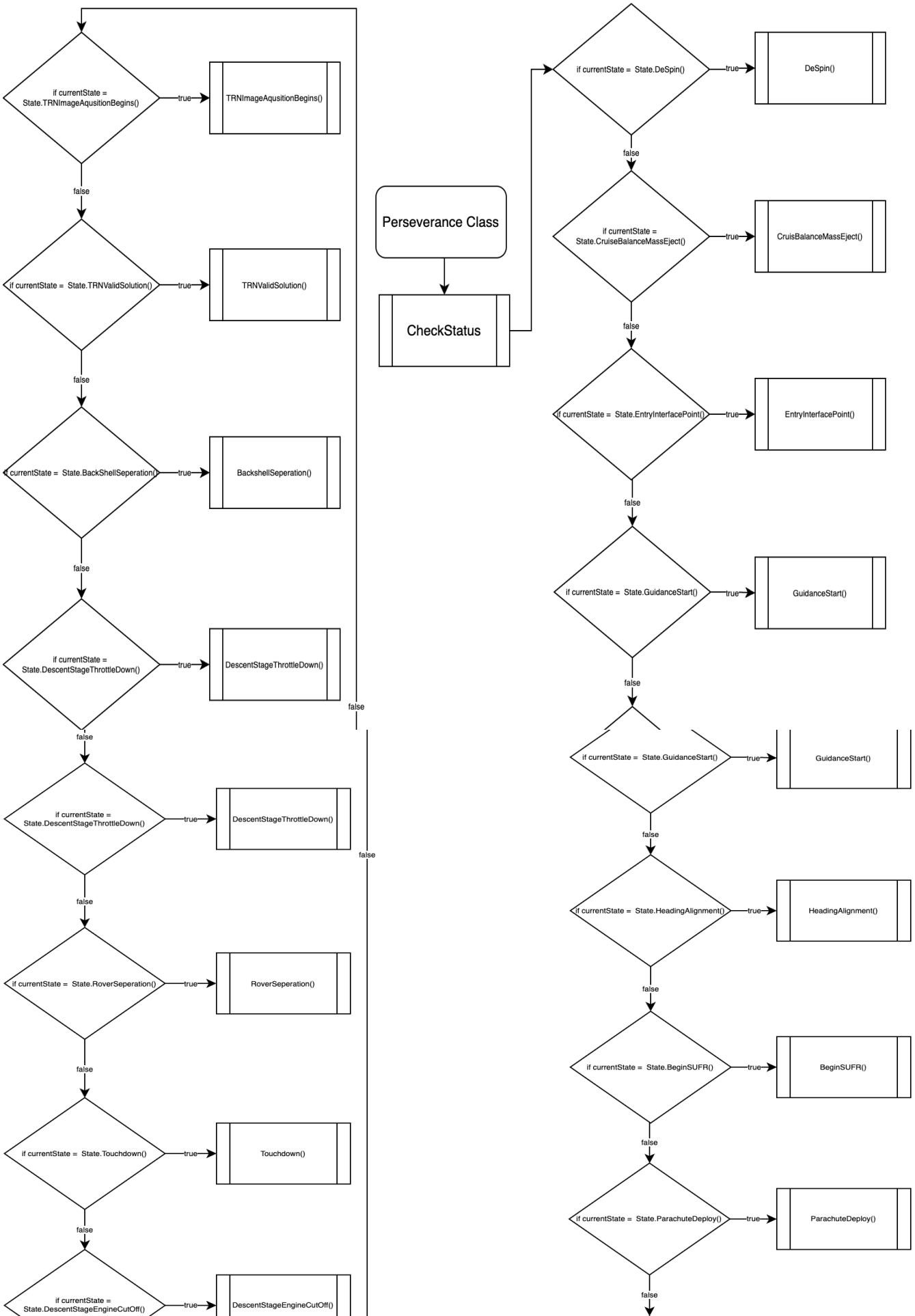
Image source:
NASA/JPL-Caltech (2020) *Mars 2020 Missi Identifier (Horizontal) – NASA's Mars Exploration Program, NASA Science Mars Exploration Program*. Available at: https://mars.nasa.gov/resources/25068/mars- -mission-identifier-horizontal/ (Accessed: 21 2021).

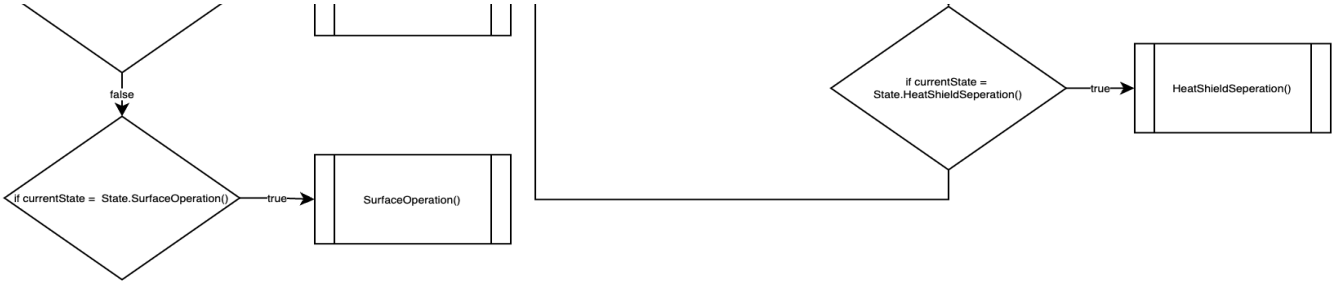
Image source:
NASA/JPL-Caltech (2020) *Mars Perseverance Rover, 3D model – NASA's Mars exploration program, NASA's Mars exploration program*. Available at: https://mars.nasa.gov/resources/25042/mars-perseverance-rover-3d-model/ (Accessed: 17 June 2021).
(Images put together by me)

Image source:
David Hitt : NASA Educational Technology Services, Sonja Alexander : HQ, Jeff Ehmen : Msfc and , Flint Wild : MSFC, Bert Ulrich : HQ (2018) 'Symbols of NASA'. Available at: https://www.nasa.gov/audience/forstudents/5-8/features/symbols-of-nasa.html (Accessed: 21 May 2021).

Image source:
David Hitt : NASA Educational Technology Services, Sonja Alexander : HQ, Jeff Ehmen : Msfc and , Flint Wild : MSFC, Bert Ulrich : HQ (2018) 'Symbols of NASA'. Available at: https://www.nasa.gov/audience/forstudents/5-8/features/symbols-of-nasa.html (Accessed: 21 May 2021).

DELIVERABLE 5(A): FLOWCHARTS PERSEVERANCE CLASS

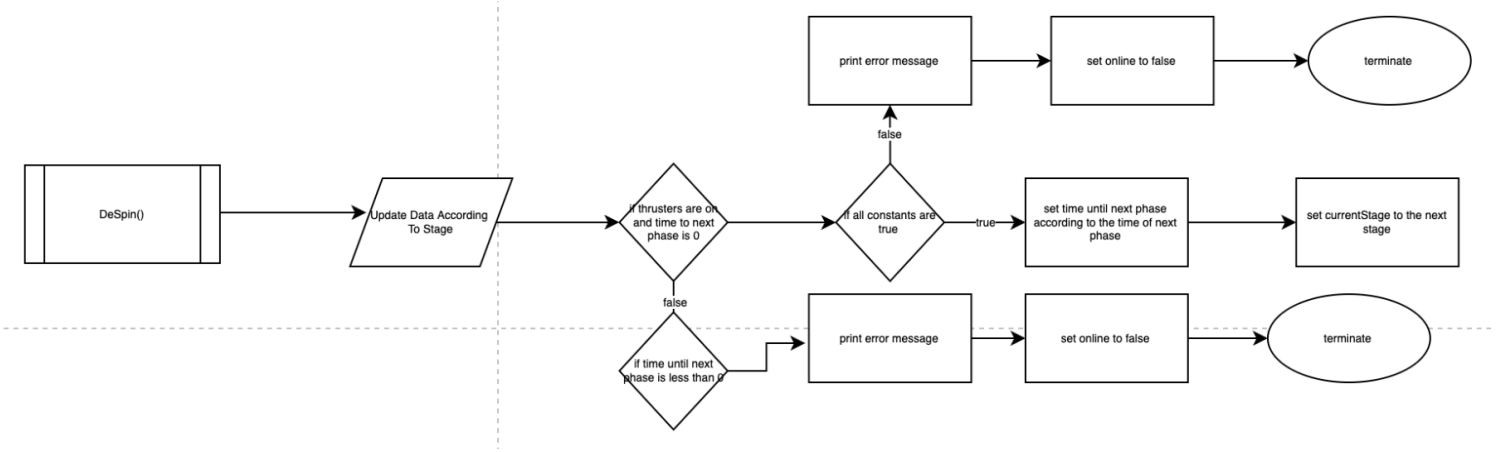



false
if currentState = State.SurfaceOperation()
true
SurfaceOperation()
if currentState = State.HeatShieldSeperation()
true
HeatShieldSeperation()

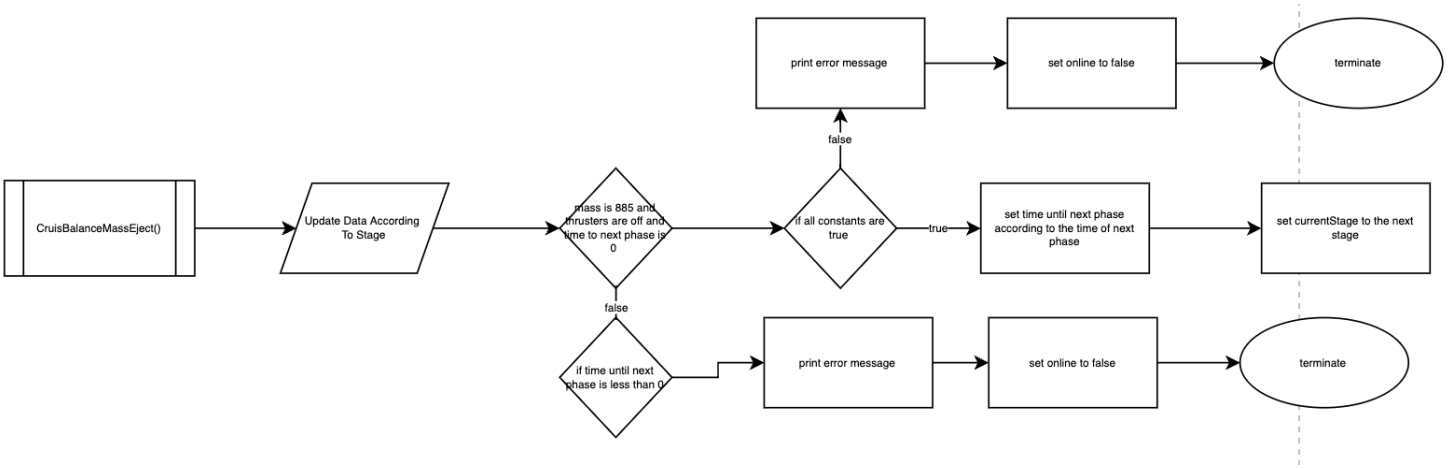
DELIVERABLE 5(B): FLOWCHARTS PERSEVERANCE CLASS

**** Since most of the landing stages are similar or equal the main 6 are demonstrated below.****

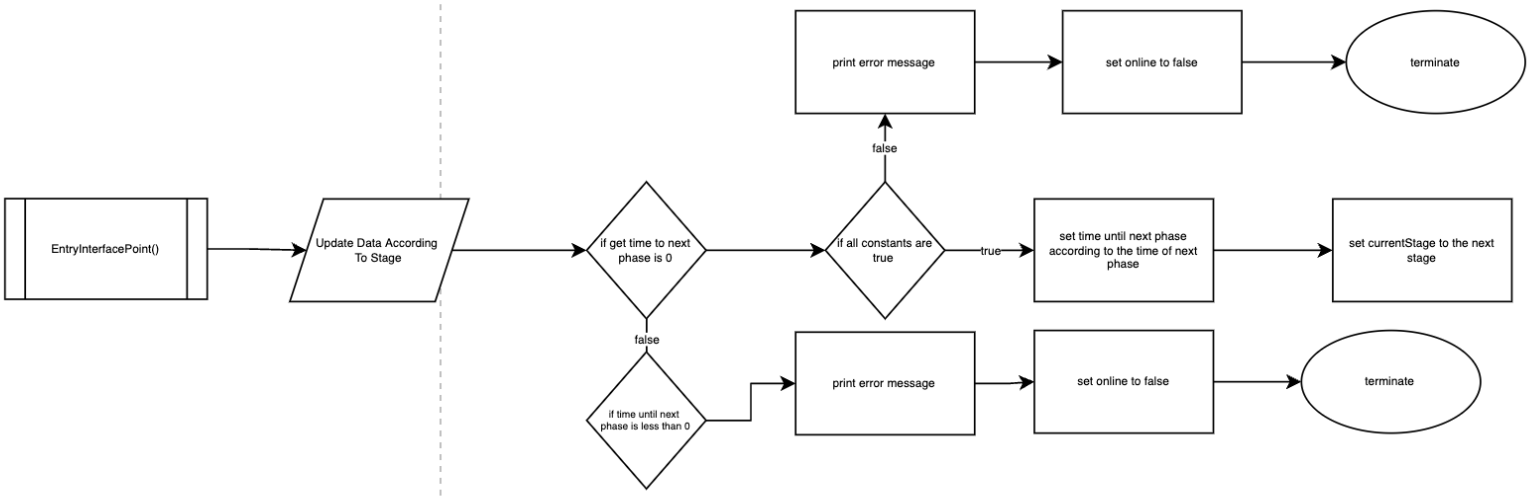
De-Spin:



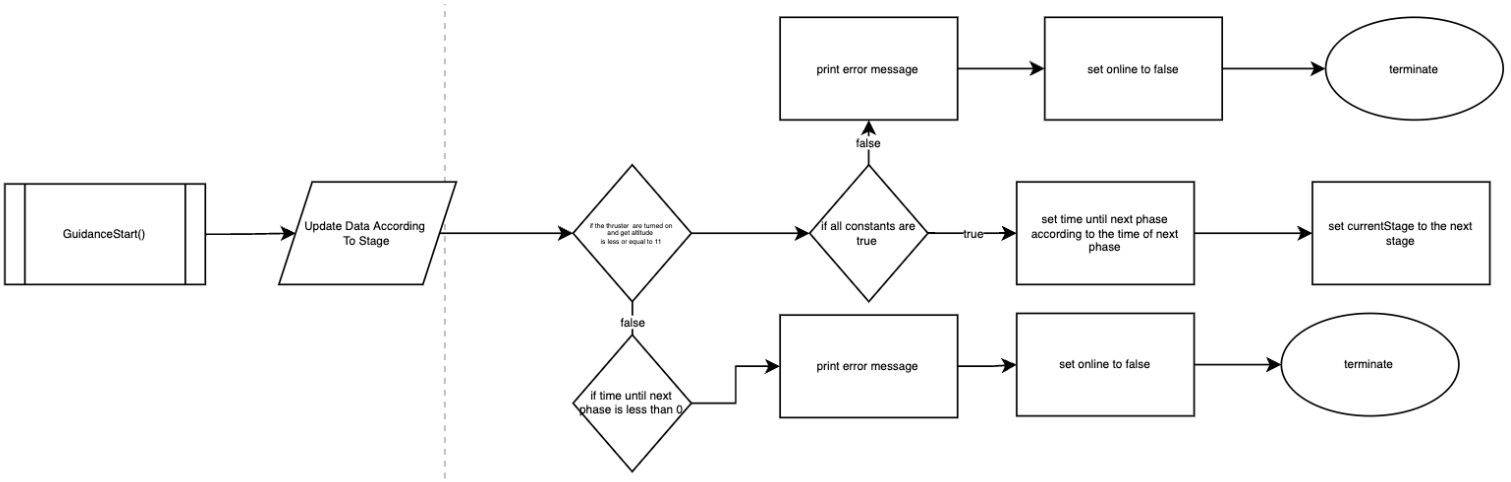
Cruise Balance Mass Eject:



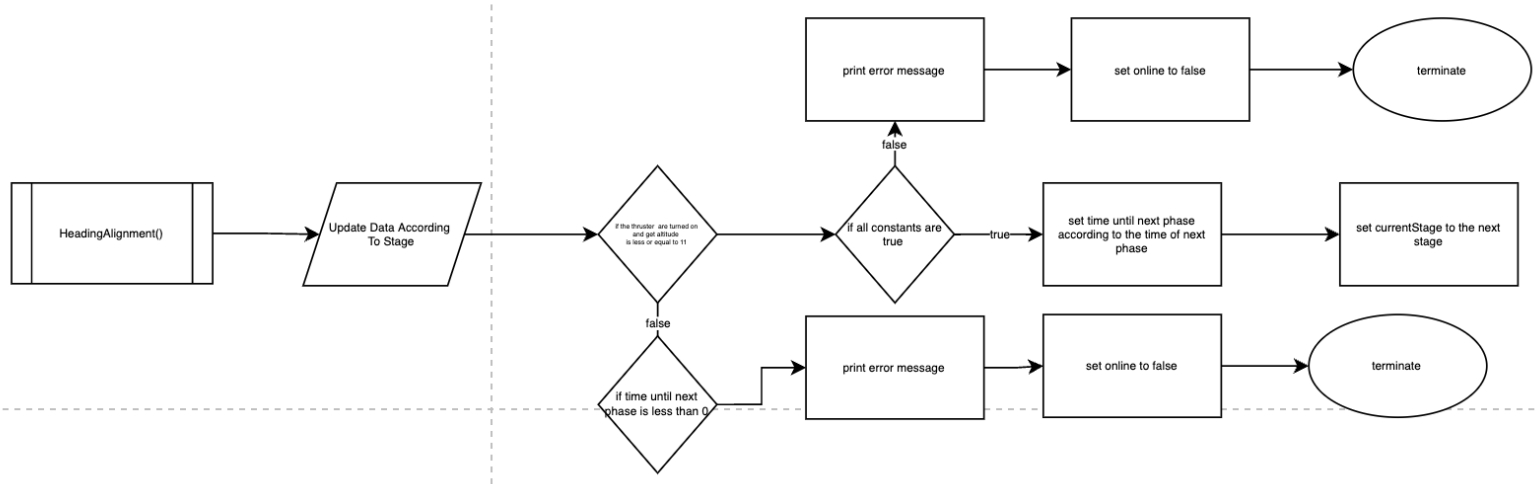
Entry Interface Point:



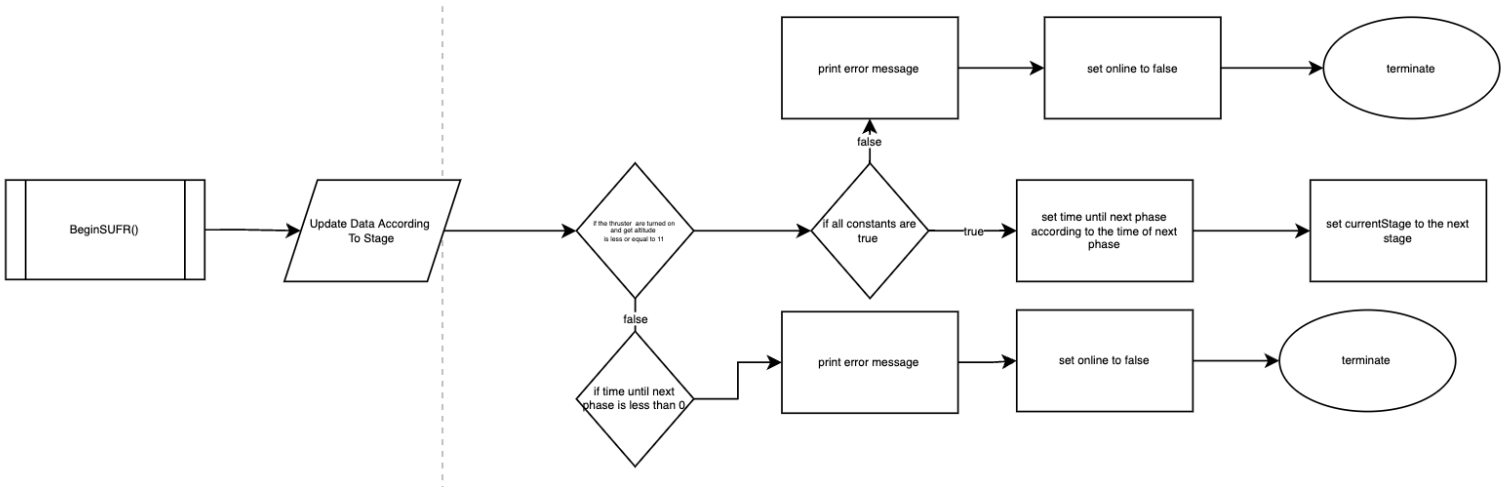
Guidance Start:



Heading Alignment:



Begin SUFR:


BeginSUFR()
Update Data According To Stage
if the thruster are turned on and get altitude is less or equal to 11
if all constants are true
false
true
print error message
set online to false
terminate
set time until next phase according to the time of next phase
set currentStage to the next stage
false
if time until next phase is less than 0
print error message
set online to false
terminate

DELIVERABLE 6(A): UML DIAGRAMS

Perseverance Rover
- currentState : State - velocity : double - altitude : double - distance : double - touchdownTime : double - timeToNextPhase: double - mass : double - heatshield : boolean - parachute : boolean - backShell : boolean - camera : boolean - cables : boolean - thruster : boolean - cableslength : double - eroor: String - online: boolean - completed: String - safelanding: boolean = false
- Perseverance(): void - CheckStatus() : void - De-spin() : void - CruiseBalanceMassEjected() : void - EntryInterfacePoint(): void - GuidanceStart() : void - HeadingAlignment() : void - BeginSUFR() : void - ParachuteDeploy() : void - HeatShieldSeparation() : void - TRNImageAcquisitionBegins() : void - TRNValidSolution() : void - BackshellSeparation() : void - DescentStageThrottleDown() : void - RoverSeparation() : void - Touchdown() : void - DescentStageEngineCutoff() : void - SurfaceOperation() : void + setLandingCoordinates(x : int , y : int) : void

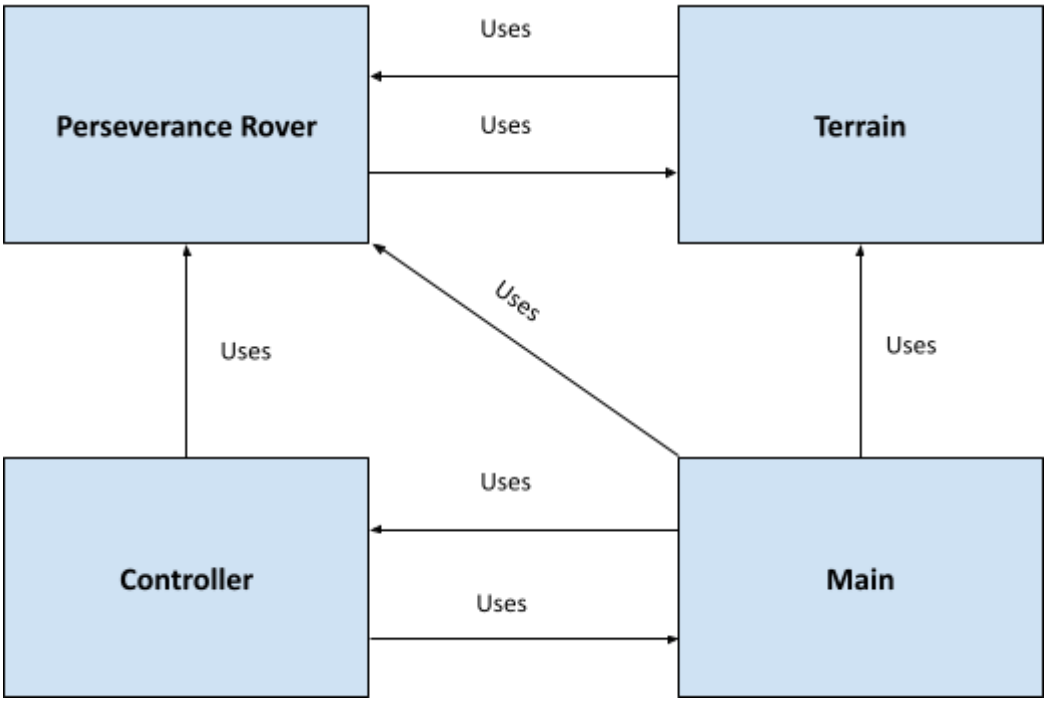
+ changeVelocity(num : double, rateOfChangeOfVelocity : double) : void
+ changeAltitude(num : double, rateOfChangeOfAltitude : double) : void
+ changeDistance(num : double, rateOfChangeOfDistance : double) : void
+ changeTouchDownTimes(num : double, RateOfChangeOFTouchDownTime : double) : void
+ changeTimeToNextphase(num : double, rateOfChangeOfTimeToNextPhase : double) : void
+ getCompleted() : String
+ getVelocity() : double
+ setVelocity(velocity1 : double) : void
+ getAltitude() : double
+ setAltitude(altitude1 : double) : void
+ getDistance() : double
+ setDistance(distance1 : double) : void
+ getTouchDownTime() : double
+ setTouchDownTime(touchDownTime1 : int): void
+ setMass(mass : int) : void
+ isHeatShield() : boolean
+ setHeatShield(boolean : heatShield1) : void
+ isParachute() : boolean
+ setParachute(parachute1 : boolean) : void
+ isBackShell() : boolean
+ setBackShel(backShell1 : boolean) : void
+ isCamera(): boolean
+ setCamera(backShell1 : boolean) : void
+ isCables() : boolean
+ setCables(cables1 : boolean) : void
+ setCablesLength(cablesLength1 : double) : void
+ getError() : String
+ getCurrentState() : State
+ setCurrentState(currentState1 : State) : void
+ getTimeToNextPhase() : double
+ setTimeToNextPhase(timeToNextPhase1 : int) : void
+ getMass() : int
+ getThrusters() : boolean
+ setThruster(thrusters1 : boolean) : void
+ isThrusters() : boolean
+ isOnline() : boolean
+ setOnline(online1 : boolean) : void
+ setSafeLanding(safeLanding1 : boolean) : void

Terrain
- LandingSpot() - map : ArrayList<Object>
+ Terrain() + landingSpot() : ArrayList<Object> + landingTerrain() : ArrayList<Object> + location() : int[][]

Controller
- timer : int - flyImage : ImageView - currentStage1 : Text - mainRover : ImageView - thrusterAttachment : ImageView - backShell : ImageView - heatShiel : ImageView - velocityText : Text - altitudeText : Text - distanceText : Text - timeToNextPhaseText : Text - timeUntilLandingText : Text - parachute : ImageView - thrusters : Group - MissionCompleted : Label - mainPane : AnchorPane - layOut : ImageView - rand : Random - allWind : ArrayList<ImageView> - min : int - max : int - alterWind() - initializeWind() - removeWind()
+ Controller() + run() : void + run() : void + getTimer() : int + setTimer(timer1 : int) : void

Main
-
+ start(primaryStage : Stage) : void + main(args : String[]) : void + studentCode() : void

DELIVERABLE 6(B): UML DIAGRAMS RELATIONSHIP



DELIVERABLE 7: Test Table

Success Criteria	**Reason**	**Data/Action**	**Outcome**
1	Check that the application goes through all landing stages	Have the user run the application	Passed Partially Passed Failed
2a	Basic layout of the GUI in relation to the specifications of the success criteria	Have the user run the application	Passed Partially Passed Failed
2b	ImageView of mars rover parts specified on the success criteria	Have the user run the application	Passed Partially Passed Failed
2c	Have a text boxes including all the essential information of the landing that updates every given time that is decided by the user.	Have the user test different times and run the application	Passed Partially Passed Failed
2d	Have two main images of the terrain that will transition. One that happens when the rover is falling and another one when it is landing.	Have the user run the application	Passed Partially Passed Failed
3	Have a class on the Perseverance project where the user can code their solution the challenge and run it to check if it works.	Have the user code the simulation and run it	Passed Partially Passed Failed
4	Make sure that a randomized mars terrain of 2d arrays is created every time that the application is ran.	Print out the coordinates of the zero and landing point every time the application runs	Passed Partially Passed Failed
5a	Check that manual has all stages of landing with basic information so that the student can understand the assignment properly.	Present manual book to student to see if they understand	Passed Partially Passed Failed
5b	Check that the manual has all functions necessary so that the	Present manual book to student and let them	Passed Partially Passed

	student can code the actions of the perseverance rover.	complete the challenge to see if function are missing	Failed