Miguel Padilla, Diego Acosta

CS 3331

May 6, 2025

Use Case Scenarios

**Scenario 1: Scientist Uses Assess Orbit Status**

**Actor:** Scientist  
**Goal:** Determine whether debris is still in orbit and calculate risk level.

**Preconditions:**

* The program is running.
* The user has selected the “Scientist” role from the UI.

**Trigger:**

* The user selects “Assess Orbit Status” from the Scientist menu.
* Scientist selects "Assess Orbit Status" from the main menu.
* Scientist is authenticated and logged in.
* The dataset (CSV) is loaded and accessible.

**Scenario:**

1. Scientist logs in.
2. Selects "Assess Orbit Status" from the menu.
3. System presents option to track objects in LEO.
4. Scientist selects "Assess if debris is still in orbit".
5. System evaluates using orbit type, longitude, days old, and conjunction count.
6. System calculates orbital drift and determines risk level.
7. CSV and TXT reports are generated.
8. Interaction is logged.

**Extensions:**

* If input data is missing, system logs and handles with error.

**Postconditions:**

* The debris objects are updated with new fields: still\_in\_orbit, risk\_level.
* CSV and TXT files are saved.
* Action is recorded in the system log.

**Scenario 2: Administrator Creates User**

**Actor:** Administrator  
**Goal:** Add a new user account to the system.

**Preconditions:**

* The program is running.
* The user has selected the “Administrator” role from the UI.

**Trigger:**

* The user selects “Create User” from the Administrator menu.
* Administrator selects "Create User" from the main menu.
* Administrator is authenticated and logged in.
* User management functionality is available.

**Scenario:**

1. Administrator logs in.
2. Selects "Create User".
3. Enters new username, password, and role.
4. System validates input and confirms user creation.
5. Logs the creation event with timestamp.

**Extensions:**

* If username already exists, error message is shown.
* If role is invalid, prompt is shown to retry.

**Postconditions:**

* New user is added to the system.
* Action is recorded in the system log.

**Scenario 3: Scientist Tracks Objects in Space**

**Actor:** Scientist  
**Goal:** Retrieve and view orbital object data categorized by type.

**Preconditions:**

* The program is running.
* The user has selected the “Scientist” role from the UI.

**Trigger:**

* The user selects “Track Objects in Space” from the Scientist menu.
* Scientist selects "Track Objects in Space" from the main menu.
* Scientist is authenticated and logged in.
* Space object data is loaded into the system.

**Scenario:**

1. Scientist logs in.
2. Selects "Track Objects in Space" from the menu.
3. System presents object categories: Rocket Body, Payload, Debris, Unknown.
4. Scientist selects a category (e.g., Payload).
5. System displays objects with full details: Record ID, Satellite Name, Country, Orbit Type, Launch Year, Launch Site, Longitude, Avg. Longitude, Geohash, Days Old.
6. The action is logged.

**Extensions:**

* If selected category has no data, system notifies user and returns to menu.

**Postconditions:**

* Requested object list is displayed.
* The action is recorded in the system log.

**Scenario 4: Space Agency Rep Generates Density Report**

**Actor:** Space Agency Representative  
**Goal:** Generate a report showing space object density across orbital altitudes.

**Preconditions:**

* The program is running.
* The user has selected the “Space Agency Representative” role from the UI.

**Trigger:**

* The user selects “Generate Density Report” from the Space Agency menu.
* Space Agency Rep selects "Generate Density Report" from the menu.
* Space Agency Rep is authenticated and logged in.
* Space object data includes valid altitude or longitude values.

**Scenario:**

1. Space Agency Rep logs in.
2. Selects "Generate Density Report" from the menu.
3. System calculates object density across defined altitude bands.
4. System categorizes density levels (e.g., Low, Moderate, High).
5. Report is generated and saved.
6. The action is logged.

**Extensions:**

* If data is insufficient for report generation, system displays error and logs it.

**Postconditions:**

* Density report file is saved.
* Log file is updated with the action.

**Scenario 5: Administrator Deletes a User**

**Actor:** Administrator  
**Goal:** Remove an existing user account from the system.

**Preconditions:**

* The program is running.
* The user has selected the “Administrator” role from the UI.

**Trigger:**

* The user selects “Delete User” from the Administrator menu.
* Administrator selects "Delete User" from the menu.
* Administrator is authenticated and logged in.
* There are existing users in the system to delete.

**Scenario:**

1. Administrator logs in.
2. Selects “Delete User” from the main menu.
3. System displays a list of existing users.
4. Administrator selects a user to delete.
5. System confirms and deletes the user.
6. The deletion is logged with a timestamp.

**Extensions:**

* If administrator attempts to delete a protected/root account, the system shows an error message.
* If user does not exist anymore (e.g., deleted in parallel session), system handles gracefully.

**Postconditions:**

* User is removed from the system.
* Deletion event is logged.