|  |  |
| --- | --- |
| Use Case Name: | **Create New Part** |
| ID: | **1.0** |
| Summary: | This use case allows the authorized inventory staff to create a new inventory item. |
| Primary Actors: | Inventory Staff |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The inventory staff must have successfully logged into the system. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “create new part.” 2. The primary actor enters part information. 3. If the part already exists then maintain attributes. **(1.2 include maintain part attributes)** 4. If all information fields are filled out then create the new part. 5. Else, tell primary actor to refill information in missing fields. 6. The system validates the inputted data. **(1.1 include validate part)** 7. The system adds the part data to the database and generates a unique bar code. |
| Post Conditions: | 1. The system adds the part data to the database. |
| Alternative Flows/Exceptions | Invalid Part Information, not all the fields were entered correctly |

**Part Replacement System Use Case Specification**

|  |  |
| --- | --- |
| Use Case Name: | **Maintain Part Attributes** |
| ID: | **1.2** |
| Summary: | This use case allows the inventory staff to maintain the created part attributes. |
| Primary Actors: | Inventory Staff |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The inventory staff must have successfully logged into the system. 2. The part must have been previously created. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “maintain part attributes.” 2. If the part has been previously created then update attributes. 3. The system validates the inputted data. **(include 1.1 validate data)** 4. The system saves the data to the database. |
| Post Conditions: | 1. The system saves the data to the database. |
| Alternative Flows/Exceptions | Invalid Information |

|  |  |
| --- | --- |
| Use Case Name: | **Issue On Unit Part** |
| ID: | **2.0** |
| Summary: | This use case allows the inventory staff to issue on unit parts. |
| Primary Actors: | Inventory Staff |
| Secondary Actors/Stakeholders: | Mechanic |
| Preconditions: | 1. The inventory staff must have successfully logged into the system. 2. Must have a valid RO unit to issue replacement. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “issue on unit part.” 2. If the part is a valid RO unit then enter data for replacement. 3. If the on unit has been previously issued then update date on unit. (**include 2.1 update on unit)** 4. The system validates the inputted data. (**include 2.2 validate on unit part**) 5. The function decrements the inventory count for the issued item. 6. The system saves the data to the database. 7. The system sends a notice to the mechanic to replace the off unit part. |
| Post Conditions: | 1. The system saves the data to the database. 2. The system sends a notice to the mechanic to replace the off unit part. |
| Alternative Flows/Exceptions | Invalid On Unit, |

|  |  |
| --- | --- |
| Use Case Name: | **Create New Customer** |
| ID: | **3.0** |
| Summary: | This use case allows the warehouse staff to create a definition for a new customer. |
| Primary Actors: | Warehouse staff |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The warehouse staff must have successfully logged into the system. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “create new customer.” 2. The primary actor enters the customer data. 3. If the customer has been previously defined then update the customer information. **(include 3.2 update customer information)** 4. If all customer fields are filled then create new customer. 5. Else, tell the primary actor to refill the fields. 6. The system validates the inputted data. **(include 3.1 validate data)** 7. The system adds the data to the database and assigns the customer a unique id number. |
| Post Conditions: | 1. The system adds the data to the database and assigns the customer a unique id number. |
| Alternative Flows/Exceptions | Invalid Information, not all the fields were entered correctly |

|  |  |
| --- | --- |
| Use Case Name: | **Update Customer Info** |
| ID: | **3.2** |
| Summary: | This use case allows the warehouse staff to update the customer information. |
| Primary Actors: | Warehouse staff |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The warehouse staff must have successfully logged into the system. 2. The customer has been previously defined. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “update customer information.” 2. If the customer was previously defined then update info. 3. The system validates the inputted data. **(include 3.1 validate data)** 4. The system saves the data to the database. |
| Post Conditions: | 1. The system saves the data to the database. |
| Alternative Flows/Exceptions | Invalid Information, The customer wasn’t previously defined. |

|  |  |
| --- | --- |
| Use Case Name: | **Create Vendor** |
| ID: | **4.0** |
| Summary: | This use case allows the warehouse staff to create a vendor. |
| Primary Actors: | Warehouse staff |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The warehouse staff must have successfully logged into the system. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “create vendor.” 2. The primary actor enters the vendor information. 3. If the vendor was previously defined then update info. **(include 4.2 update vendor)** 4. If all the vendor fields are filled then create vendor. 5. Else, tell the primary actor to refill the fields. 6. The system validates the inputted data. **(include 4.1 validate data)** 7. The system adds the data to the database. |
| Post Conditions: | 1. The system adds the data to the database. |
| Alternative Flows/Exceptions | Invalid Information, not all the fields were entered correctly |

|  |  |
| --- | --- |
| Use Case Name: | **Update Vendor Info** |
| ID: | **4.2** |
| Summary: | This use case allows the warehouse staff to update the vendor information. |
| Primary Actors: | Warehouse staff |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The warehouse staff must have successfully logged into the system. 2. The vendor has been previously defined. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “update vendor information.” 2. If the vendor was previously defined then update info. 3. The system validates the inputted data. **(include 4.1 validate data)** 4. The system saves the data to the database. |
| Post Conditions: | 1. The system saves the data to the database. |
| Alternative Flows/Exceptions | Invalid Information, The vendor wasn’t previously defined. |

|  |  |
| --- | --- |
| Use Case Name: | **Receive Aircraft** |
| ID: | **5.0** |
| Summary: | This use case allows the warehouse staff to receive aircraft from customers. |
| Primary Actors: | Warehouse staff |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The warehouse staff must have successfully logged into the system. 2. The aircraft must be from a valid APP customer. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “receive aircraft.” 2. The primary actor enters the aircraft data. 3. If the aircraft was previously defined then update info. **(include 5.3 update aircraft attributes)** 4. If all the aircraft fields are filled then receive aircraft. 5. Else, tell primary actor to enter missing fields. 6. The system validates the inputted data. **(include 5.1 validate data)** 7. The system adds the data to the database. |
| Post Conditions: | 1. The system adds the data to the database. |
| Alternative Flows/Exceptions | Invalid Information, not all the fields were entered correctly |

|  |  |
| --- | --- |
| Use Case Name: | **Update Aircraft Attributes** |
| ID: | **5.3** |
| Summary: | This use case allows the warehouse staff, warehouse manager, inspector, and mechanic to update aircraft attributes. |
| Primary Actors: | Warehouse staff, Warehouse Manager, Inspector, and Mechanic |
| Secondary Actors/Stakeholders: | None. |
| Preconditions: | 1. The primary actor must have successfully logged into the system. 2. The aircraft must have been previously defined. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “receive aircraft.”   1. If the aircraft was previously defined then update info. 2. The system validates the inputted data. **(include 5.1 validate data)** 3. The system saves the data to the database. |
| Post Conditions: | 1. The system saves the data to the database. |
| Alternative Flows/Exceptions | Invalid Information, the aircraft wasn’t previously defined |

|  |  |
| --- | --- |
| Use Case Name: | **Create Repair Order** |
| ID: | **6.0** |
| Summary: | This use case allows the inspector to issues a repair order to a received off unit. |
| Primary Actors: | Inspector |
| Secondary Actors/Stakeholders: | Vendor |
| Preconditions: | 1. The inspector must have successfully logged into the system. 2. The off unit must be a valid part. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “create repair order.” 2. The primary actor enters the repair order. 3. If the repair order was previously defined then update order. **(include 6.2 update repair order)** 4. If all the aircraft fields are filled then receive aircraft. 5. Else, tell primary actor to enter missing fields. 6. The system validates the inputted data. **(include 6.1 validate data)** 7. The system adds the data to the database and generates a unique repair order number. |
| Post Conditions: | 1. The system adds the data to the database and generates a unique repair order number. |
| Alternative Flows/Exceptions | Invalid Information, not all the fields were entered correctly |

|  |  |
| --- | --- |
| Use Case Name: | **Update Repair Order** |
| ID: | **6.2** |
| Summary: | This use case allows the inspector to update a repair order to a received off unit. |
| Primary Actors: | Inspector |
| Secondary Actors/Stakeholders: | Vendor |
| Preconditions: | 1. The inspector must have successfully logged into the system. 2. The off unit must be a previously defined |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “update repair order.” 2. If the repair order was previously defined then update order. 3. The system validates the inputted dat**a. (include 6.1 validate data)** 4. The system saves the data to the database and generates a unique repair order number. |
| Post Conditions: | 1. The system saves the data to the database and generates a unique repair order number. |
| Alternative Flows/Exceptions | Invalid Information |

|  |  |
| --- | --- |
| Use Case Name: | **Inspect Repaired Off Unit** |
| ID: | **10.0** |
| Summary: | This use case allows the inspector to inspect repaired parts. |
| Primary Actors: | Inspector |
| Secondary Actors/Stakeholders: | Vendor |
| Preconditions: | 1. The inspector must have successfully logged into the system. 2. The off unit must have a repair order associated with it. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “inspect repaired off unit.” 2. The primary actor enters the repair order number into the system. 3. If the inspect repaired off unit was previously defined then update. **(include 10.2 update inspection)** 4. The inspector then manually inspects the off unit according to FAA guidelines. 5. If the off unit has been successfully repaired then check “pass inspection.”  * **(include 1.0 create new part)** * **(include 1.1 validate part)** * The system adds the part to the database and generates a unique barcode.  1. If the off unit is checked with “reject” then enter inspection date and reason for rejection in comment area.  * Item is shipped back to vendor  1. The system validates the inputted data. **(include 10.1 validate data)** 2. The system adds the data to the database. |
| Post Conditions: | 1. The system adds the data to the database. |
| Alternative Flows/Exceptions | Invalid Information |

|  |  |
| --- | --- |
| Use Case Name: | **Update Inspection** |
| ID: | **10.2** |
| Summary: | This use case allows the inspector to update an inspection to a received off unit. |
| Primary Actors: | Inspector |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The inspector must have successfully logged into the system. 2. The inspection must have been previously recorded. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “update inspection.” 2. If the inspection was previously defined then update inspection. 3. The system validates the inputted data. **(include 10.1 validate data)** 4. The system saves the data to the database |
| Post Conditions: | 1. The system saves the data to the database. |
| Alternative Flows/Exceptions | Invalid Information |

|  |  |
| --- | --- |
| Use Case Name: | **Match Off Unit and On Unit** |
| ID: | **7.0** |
| Summary: | This use case allows the inspector to match received off units and issued on units. |
| Primary Actors: | Inspector |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The inspector must have successfully logged into the system. 2. The off unit and on unit must have been previously defined |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “match off unit and on unit.” 2. Inspect matches off unit and on unit. 3. The system validates the inputted data. **(include 7.1 validate data)** 4. The replacement fee is calculated and recorded. 5. The system adds the data to the database. |
| Post Conditions: | 1. The system adds the data to the database |
| Alternative Flows/Exceptions | Invalid Information |

|  |  |
| --- | --- |
| Use Case Name: | **Receive Off Unit** |
| ID: | **8.0** |
| Summary: | This use case allows the inspector to receive off units from a customer’s aircraft. |
| Primary Actors: | Inspector |
| Secondary Actors/Stakeholders: | None |
| Preconditions: | 1. The inspector must have successfully logged into the system. 2. The aircraft must be from a valid APP customer. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “receive off unit.” 2. The inspector enters the data. 3. If the off unit was previously defined then update. **(include 8.1 update off unit)** 4. The inventory count is increased for off unit. 5. The system validates the inputted data. **(include 8.1 validate data)** 6. The system adds the data to the database. |
| Post Conditions: | 1. The system adds the data to the database. |
| Alternative Flows/Exceptions | Invalid Information |

|  |  |
| --- | --- |
| Use Case Name: | **Open Replacement Report** |
| ID: | **9.0** |
| Summary: | This use case allows the warehouse manager to produce a report listing of open replacements. |
| Primary Actors: | Warehouse Manger |
| Secondary Actors/Stakeholders: | Customer |
| Preconditions: | 1. The inspector must have successfully logged into the system. 2. The on unit or off unit must be a valid part. |
| Main Flow/Normal:  (including sub-flows) | 1. This use case starts when the primary actor selects “open replacement report.” 2. The warehouse manger enters the replacement of either an off unit or on unit. 3. If the replacement is completed and passed inspection the aircraft maintenance is “complete” and is allowed to be release to the customer. 4. The system validates the inputted data. **(include 8.1 validate data)** 5. The system adds the data to the database. |
| Post Conditions: | 1. The system adds the data to the database. |
| Alternative Flows/Exceptions | Invalid Information |