|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **System Name: HestiService system** | | | | | |
| **Author:** Anke Brits | **Date:** 21 July 2024 | | | **Version:** 1.0.0 | |
|  | | | | | |
| **Use Case Name:** | Create work order | | **Use Case Type** | | |
| **Use Case ID:** | 4.1 | | Business Requirements: ◻ | | |
| **Priority:** | High | | System Analysis: ◻ | | |
| **Source:** | Client study (Hestico) | | System Design: ☒ | | |
| **Primary Business Actor (PBA):** | Technician | | | | |
| **Primary System Actor (PSA):** | None | | | | |
| **Other Participating Actors:** | None | | | | |
| **Other Interested Stakeholders:** | Admin | | | | |
| **Description:** | This use case describes the process of creating a work order.  The technician will click the "Create Work Order Update" button on the **scheduled service page**. The technician will then be directed to the “Create Work Order” screen where the newly created work order will be displayed.  The use case concludes when the work order has been created. | | | | |
| **Pre-condition:** | * The technician needs to be logged in * There needs to be an existing service request to create a work order from | | | | |
| **Trigger** | * The technician wants to create a new work order and clicks the "Create Work Order Update" button on the **scheduled service page**. | | | | |
| **Typical Course**  **of Events:** | **Actor Action** | **System Response** | | | |
| **Manual Action** | | | **Automated Action** |
| Step 1: The technician wants to create a new work order and clicks the "Create Work Order Update" button on the **scheduled service page**. |  | | | Step 2: The system redirects the technician to the “Create Work Order” screen which contains the following elements:  A heading with the text “Work Order Details” at the top of the screen.  A card displaying the work order details with the following information:   * Work Order ID * Service Request ID * Machine Type * Status * Technician (Employee name and Employee surname) * Reason * Date Started * Date Completed (if available) * "Create Work Order Update" button * “Cancel” button   The system uses Entity Framework Core to check if there is a work order already associated with the service request.  The system will send a request form the Angular frontend to the Work Order service where the service will make a http get request to the .NET Core backend which makes use of a Lambda LINQ Query which creates a SQL Create query to create the work order in the Work Order Entity and the corresponding information from tables that are referenced by the foreign keys. The tables referenced by the foreign keys are described below.  The system displays the work order details by using Entity Framework Core to retrieve only the related data in the Work\_Order table with the following attributes:   * Work\_Order\_Id (PK) * Service\_Request\_Id (FK) * Machine\_Type\_Id (FK) * Work\_Order\_Status\_Id (FK) * Employee\_Id (FK) * Reason * Date\_Started * Date\_Completed * Invoice\_Id (FK)   The system links the Work\_Order table to the Service\_Request table using the foreign key Service\_Request\_Id. The Service\_Request table has the following attributes:   * Service\_Request\_Id (PK) * Service\_Request\_Status\_Id (FK) * Service\_Type\_Id (FK) * Technician\_Id (FK) * Machine\_Type\_Id (FK) * Representative\_Id (FK) * Preferred\_Date * Alternative\_Date\_1 * Alternative\_Date\_2 * Alternative\_Date\_3 * Problem\_Description * Finalised\_Date * Work\_Order\_Id (FK)   The system links the Work\_Order table to the Machine\_Type table using the foreign key Machine\_Type\_Id. The Machine\_Type table has the following attributes:   * Machine\_Type\_Id (PK) * Name * Description * Serial\_Number   The system links the Work\_Order table to the Work\_Order\_Status table using the foreign key Work\_Order\_Status\_Id. The Work\_Order\_Status table has the following attributes:   * Work\_Order\_Status\_Id (PK) * Name * Description   The system links the Work\_Order table to the Employee table using the foreign key Employee\_Id. The Employee table has the following attributes:   * Employee\_Id (PK) * Employee\_Type\_Id (FK) * UserId * Name * Surname * Address * Phone\_No * Gender * Race * Preferred\_Name   The system links the Work\_Order table to the Invoice table using the foreign key Invoice\_Id. The Invoice table has the following attributes:   * Invoice\_Id (PK) * Customer\_Reference\_Number * Discount\_Id (FK) * Work\_Order\_Id (FK) * Quote\_Id (FK) * Invoice\_Status\_Id (FK) * Date\_Issued * Date\_Paid * Total\_Amount * Description   The system sets the foreign key attribute Invoice\_Id in the Work\_Order table to null since no Invoice has been generated.  The system sets the attribute Date\_Completed to null as well since the work order has not been completed.  The system logs the following when work order is created:   * user performing the operation * Transaction Type: * The description which contains the work order id.   In the following Audit\_Trail entity has the following attributes:   * Audit\_Trail\_Id (PK) * Date\_Time * User\_Name * Transaction\_Type * Description   The Audit\_trail\_Id is automatically incremented.  [ALT] |
| Step 3: The technician then clicks the “Create work order update” button.  [ALT] |  | | | Step 4: The system invokes use case 4.2 Update Work Order. |
| **Alternate Courses:** | [ALT] Step 2: There is already a work order connected to this service request. The system will send a request form the Angular frontend to the Work Order service where the service will make a http get request to the .NET Core backend which makes use of a Lambda LINQ Query which creates a SQL Select query to retrieve the work order from the Work Order Entity and the corresponding information from tables that are referenced by the foreign keys. The tables referenced by the foreign keys are described above in step 2. The system invokes use case 4.2 Update Work Order. | | | | |
| [ALT] Step 3: The technician clicks the cancel button. The technician is redirected to the **scheduled service page.** The created work order is still saved. | | | | |
| **Conclusion:** | The work order was successfully created and added to the database.  The work order update was successfully created and added to the database. | | | | |
| **Post-condition:** | The list of work orders on the “View Work Orders” screen is updated. | | | | |
| **Business Rues:** | * Only the technician can create work orders * Only the technician can create work order updates | | | | |
| **Implementation Constraints and Specifications:** | * None | | | | |
| **Assumptions:** | * None | | | | |
| **Open Issues:** | * None | | | | |