Requirements Engineering

FoneFix

Submitted By: Jack Teahan (t00192059)

Computing with Software Development

Date Submitted: dd/mm/yyyy

**Table of Contents**

[1. Introduction/overview 3](#_Toc474162299)

[2. Functional Components 4](#_Toc474162300)

[3. User Requirements 5](#_Toc474162301)

[4. System Requirements 6](#_Toc474162302)

[4.1. System Level Use Case Diagram 6](#_Toc474162303)

[4.2. Module 1 (Use active Verb + Noun) 6](#_Toc474162304)

[4.2.1. Functional Requirement 1 (Use active verb + noun) 6](#_Toc474162305)

[4.2.2. Functional Requirement 2 (Use active verb + noun) 6](#_Toc474162306)

[4.2.3. Functional Requirement 3 (Use active verb + noun) 6](#_Toc474162307)

[4.3. Module 2 (Use active Verb + Noun) 6](#_Toc474162308)

[4.3.1. Functional Requirement 1 (Use active verb + noun) 6](#_Toc474162309)

[4.3.2. Functional Requirement 2 (Use active verb + noun) 6](#_Toc474162310)

[4.4. Module 3 (Use active Verb + Noun) 6](#_Toc474162311)

[4.4.1. Functional Requirement 1 (Use active verb + noun) 6](#_Toc474162312)

[4.4.2. Functional Requirement 2 (Use active verb + noun) 6](#_Toc474162313)

[5. System Model 7](#_Toc474162314)

[5.1. Level-0 DFD 7](#_Toc474162315)

[5.2. Level-1 DFD 7](#_Toc474162316)

[5.3. Level-2 DFD (Process P1: Title) 7](#_Toc474162317)

[5.4. Level-2 DFD (Process P2: Title) 7](#_Toc474162318)

[5.5. Level-2 DFD (Process P3: Title) 7](#_Toc474162319)

[6. Data Model (Class Diagram) 8](#_Toc474162320)

[6.1. Class Diagram 8](#_Toc474162321)

[6.2. Relational Schema 8](#_Toc474162322)

[6.3. Database Schema 8](#_Toc474162323)

[7. Conclusion 9](#_Toc474162324)

[8. Appendices 10](#_Toc474162325)

[8.1. Appendix A – Title 10](#_Toc474162326)

[8.2. Appendix B – Title 10](#_Toc474162327)

# Introduction/overview

FoneFix is a system that revolves around an admistrator to carry out and manage repairs and dealing with customers. It allows the administrator to add customers, stock and repairs, update the status of each of these and other general information regarding these functions. The System takes in repairs, addisgs it an ID, and also assigns a customer to the Repair being worked on. This system is designed to deal with repairing phones for companys in bulk

# Functional Components

# User Requirement

## Manage Customers

* + 1. FoneFix will Register new a customer on the system
    2. FoneFix will update a customer’s details
    3. FoneFix will De-register a customer

## Manage Stock

* + 1. FoneFix will record details of a stock item on the system
    2. FoneFix will update details of a stock item
    3. FoneFix will allow a stock enquiry

## Process Repair

* + 1. FoneFix will log repairs into the system
    2. FoneFix will cancel repairs
    3. FoneFix will complete repairs and log them into the system
    4. FoneFix will replenish used stock

## Carry-Out Administration

* + 1. FoneFix will issue out invoice of repair
    2. FoneFix will receive payment from repair
    3. FoneFix will perform Stock analysis
    4. FoneFix will perform Repair Analysis

# System Requirements

|  |  |  |
| --- | --- | --- |
| **Functional Requirements** | **Non-Functional Requirements** | **Domain Requirements** |
| 1. To allow Repairs, customers and stock to be added to the system 2. To allow Customers and stock to be updated 3. To allow repairs to be cancelled / completed 4. The system will allow the admin to send invoices to customers and receive payments 5. The system will allow the generation of analysis for repair and stock | 1. The system will be run by the administrator primarily. 2. The System must be easy to navigate 3. The system must also be fully function | 1. The admin will have access from work and it can be accessed without the use of an internet connection |

## System Level Use Case Diagram

Fone Fix will allow the administrator to Manage different customers. This allows the admin to add a customer, update a customer, and delete a customer

Fone Fix will allow the administrator to Manage Stock, this allows the administrator to add new stock, update Stock, and delete Stock

Fone Fix will allow the administrator to Process Repairs. This allows the administrator to log repairs, Cancel Repairs, complete repairs, and replenish stock used in repairs

Fone Fix will allow the administrator to Carry out administration. This allows the administrator tosend out invoices, receive payment, and generate analysis’s on stock and repairs

FoneFix

Administrator

## Manage Customers

This module contains functions that Add Customer, Update Customer, and Deregister Customer.

### Add Customer

FoneFix will register new a customer on the system

Administrator

Customer

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Customer** | |
| **Use Case Id** | 4.2.1 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | FoneFix will Register new a customer on the system | |
| **Preconditions** | Customer must fill in their details on a registration form | |
| **Trigger** | none | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts register customer function  **Step 4:** Administrator enters customers details:   * CustomerName * PhoneNumber * Email * Street * Town * County * Eir\_Code | **Step 2:** Assign Customer ID  **Step 3:** Display UI  **Step 5:** System validates Customer details:   * All fields must be entered * Phone number must have valid format * Fields must not contain numeric values only * Email must have valid format * Eir\_Code must have valid format   **Step 6:** Set Customer\_Balance to 0  **Step 7:** Set Status to “A” (Active)  **Step 8:**  Set Register\_Date to System Date  **Step 9**: Save Customer details in Customer File:   * Customer\_ID * Customer\_Name * Phone\_Number * Email * Street * Town * Count * Eir\_Code * Customer\_Balance * Register\_Date * Status   **Step 9**: Display Confirmation message  **Step 10**: Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 6**: A blank Field Detected  **Step 7**: Display message “This field must be entered”  **Step 8**: Position cursor in offending field and return to step 3 |
| **Invalid Phone Number** |  | **Step 6**: An invalid Phone Number entered  **Step 7**: Display message “This phone number format is invalid”  **Step 8**: Position cursor in offending field and return to step 3 |
| **Numeric Values Detected** |  | **Step 6**: Numeric values detected in some fields  **Step 7**: Display message “These fields must not contain any numbers”  **Step 8**: Position cursor in offending field and return to step 3 |
| **Invalid Email** |  | **Step 6**: An invalid Email entered  **Step 7**: Display message “This Email format is invalid”  **Step 8**: Position cursor in offending field and return to step 3 |
| **Invalid Eircode** |  | **Step 6**: An invalid Eircode entered  **Step 7**: Display message “This Eircode format is invalid”  **Step 8**: Position cursor in offending field and return to step 3 |
| **Conclusions** | Customer is registered | |
| **Post conditions** | Customer can now request a repair | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Update Customer

FoneFix will update a customer’s details

Administrator

<<INCLUDE>>

<<EXTENDS>>

**Administrator System**

Start function

Display UI

Enter Customer name

Retrieve a summary of customers with name

Display UI

Selects customer to update

Retrieve all details for selected customer

Display UI for Updating

Administrator updates required fields

Validate

Update

Customer

Fail

Displays appropriate error message on UI

Pass

Move Cursor back to invalid field

System updates customer details

Confirm Dialog

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Customer** | |
| **Use Case Id** | 4.2.2 | |
| **Priority** | Medium | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | FoneFix will update a customer’s details | |
| **Preconditions** | Customer must be registered | |
| **Trigger** |  | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts update customer function  **Step 3:** Enter Customer name  **Step 6:** Select Customer to update  **Step 9:** Administrator updates required details | **Step 2:** Display UI  **Step 4:**Retrives summary of customers with name from customer file   * CustomerName * CustomerID   **Step 5:** Display UI  **Step 7:** Retrieves all data for selected customer   * CustomerName * PhoneNumber * Email * Street * Town * County * Eir\_Code   **Step 8:** Display UI  **Step 10:** System validates the new customer details:   * All fields must be entered * Phone number must have valid format * Fields must not contain numeric values only * Email must have valid format * Eir\_Code must have valid format   **Step 11**: Updated Customer details saved into Customer file  **Step 12**: Display Confirmation message  **Step 13**: Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 11**: A blank Field Detected  **Step 12**: Display message “This field must be entered”  **Step 13**: Position cursor in offending field and return to step 5 |
| **Invalid Phone Number** |  | **Step 11**: An invalid Phone Number entered  **Step 12**: Display message “This phone number format is invalid”  **Step 13**: Position cursor in offending field and return to step 5 |
| **Numeric Values Detected** |  | **Step 11:** Numeric values detected in some fields  **Step 12**: Display message “These fields must not contain any numbers”  **Step 13**: Position cursor in offending field and return to step 5 |
| **Invalid Email** |  | **Step 11**: An invalid Email entered  **Step 12**: Display message “This Email format is invalid”  **Step 13**: Position cursor in offending field and return to step 5 |
| **Invalid Eircode** |  | **Step 11**: An invalid Eircode entered  **Step 12**: Display message “This Eircode format is invalid”  **Step 13**: Position cursor in offending field and return to step 5 |
| **Conclusions** | Customer details updated | |
| **Post conditions** | Customer can now request a repair again | |
| **Business Rules** | Not allowed if customer is de-registered | |
| **Implementation Constraints** |  | |

### De-Register customer

FoneFix will De-register a customer

Customer

Administrator

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Deregister Customer** | |
| **Use Case Id** | 4.2.3 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | FoneFix will Deregister a customer | |
| **Preconditions** | Customer must give notice | |
| **Trigger** |  | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts deregister customer function  **Step 3:** Enter Customer name  **Step 5:** Select Customer to deregister | **Step 2:** Display UI  **Step 4:**Retrives summary of customers with name from customer file   * CustomerName * CustomerID   **Step 6:** Show confirm Dialog  **Step 7:** Save changes to customer file  **Step 8:** Display conformation message  **Step 9:** Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | Customer is deregistered from system | |
| **Post conditions** | Customer no longer can request repairs | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Manage Stock

This module will contain functions that will Record Stock, Update Stock, and perform Stock Enquiry

### New Stock

FoneFix will record details of a stock item on the system

Administrator

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **New Stock** | |
| **Use Case Id** | 4.3.1 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | FoneFix will record details of a stock item on the system | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts New Stock function  **Step 4:** Administrator enters stock details   * StockName * StockType * StockPrice * StockQuantity | **Step 2:** Assign Stock ID  **Step 3:** Display UI  **Step 5:**  System Validates input   * All fields must be entered * StockPrice must have valid format * Fields must not contain numeric values only   **Step 6:** Save Stock details in Inventory file:   * StockID * StockName * StockType * StockPrice * StockQuantity   **Step 7**: Display Confirmation message  **Step 8**: Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 6**: A blank Field Detected  **Step 7**: Display message “This field must be entered”  **Step 8**: Position cursor in offending field and return to step 3 |
| **Invalid Price Format** |  | **Step 6**: Non valid format detected for StockPrice  **Step 7**: Display message “This is not a valid price format  **Step 8**: Position cursor in offending field and return to step 3 |
| **Numeric Values Detected** |  | **Step 6**: Numeric values detected in some fields  **Step 7**: Display message “These fields must not contain any numbers”  **Step 8**: Position cursor in offending field and return to step 3 |
| **Conclusions** | New Stock added to the System | |
| **Post conditions** | New stock have values | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Update Stock

FoneFix will update details of a stock item

Administrator

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Stock** | |
| **Use Case Id** | 4.3.2 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | FoneFix will update details of a stock item | |
| **Preconditions** | System must have stock already be registered in the system | |
| **Trigger** |  | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts Update Stock function  **Step 3:** Enter stock name  **Step 6:** Select Stock to update  **Step 9:** Administrator updates required details | **Step 2:** Display UI  **Step 4:** Retrieves summary of Stock with name from inventory file   * StockName * StockID   **Step 5:** Display UI  **Step 7:**  Retrieves all data for selected Stock inventory file  **Step 8:** Display UI  **Step 10**: System validates the new customer details   * All fields must be entered * StockPrice must have valid format * Fields must not contain numeric values only   **Step 11**: Updated Stock details saved into inventory file  **Step 12**: Display confirmation message  **Step 13:** Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 11**: A blank Field Detected  **Step 12**: Display message “This field must be entered”  **Step 13**: Position cursor in offending field and return to step 5 |
| **Numeric Values Detected** |  | **Step 11**: Numeric values detected in some fields  **Step 12**: Display message “These fields must not contain any numbers”  **Step 13**: Position cursor in offending field and return to step 5 |
| **Invalid Price Format** |  | **Step 11**: Non valid format detected for StockPrice  **Step 12**: Display message “This is not a valid price format  **Step 13**: Position cursor in offending field and return to step 5 |
| **Conclusions** | Stock details has been updated | |
| **Post conditions** | Stock now have proper values | |
| **Business Rules** | Stock must be in System | |
| **Implementation Constraints** |  | |

### Stock Enquiry

FoneFix will allow a stock enquiry

Administrator

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Stock Enquiry** | |
| **Use Case Id** | 4.3.3 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | FoneFix will allow stock enquiry | |
| **Preconditions** | System must have stock already in system | |
| **Trigger** | Repairs, parts sold, etc… | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts Stock Enquiry function  **Step 3:** Administrator enters stock name  **Step 6:** Administrator can select stock and view values of stock   * StockID * StockName * StockType * StockPrice * StockQuantity   **Step 7:** Administrator makes enquiries on required fields | **Step 2:** Display UI  **Step 4:** Retrieves summary of Stock with name from inventory file  **Step 5:** Display UI  **Step 8**: Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 5**: Display message “These fields must contain only numbers” |
| **Conclusions** | Stock enquiry has been processed | |
| **Post conditions** | Administrator now has enquired on stock | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

Sample Stock Report

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***StockID*** | ***StockName*** | ***StockType*** | ***StockPrice*** | ***StockQuantity*** |
| 111 | IPhone6Screen | Screen | €50 | 24 |
| 1089 | iPhone6sScreen | Screen | €60 | 22 |
| 95 | SamsungS6Battery | Battery | €35 | 35 |

## Process Repair

This process will contain functions that will allow the administrator to Log repairs, cancel repairs, complete repairs, replenish Stock

### Log Repair

FoneFix will log repairs into the system

Customer

Administrator

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Log Repair** | |
| **Use Case Id** | 4.4.1 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | FoneFix will log repairs into the system | |
| **Preconditions** | Customer must be added to system | |
| **Trigger** | Repair stopped | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts Log Repair customer function  **Step 4:** Administrator fills out details of repair   * PhoneSerialNum * Password * CustomerName * CustomerID * Description | **Step 2:** Assign Repair ID to Customer  **Step 3:** Display UI  **Step 6:**  System Validates input   * PhoneSerialNum must contain all numeric characters * CustomerID must be a registered ID   **Step 7:**  Repair status is set to “in progress”  **Step 8:** Repair details are saved to the Repair file   * RepairID * PhoneSerialNumber * Password * CustomerID * RepairStatus * Description   **Step 9 :** Show confirmation dialog  **Step 10**: Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Non-Numeric Values** |  | **Step 7**: Non-Numeric values detected in PhoneSerialNum field  **Step 8**: Display message “These fields must contain only numbers”  **Step 9**: Position cursor in offending field and return to step 3 |
| **Invalid CustomerID** |  | **Step 7:** No Customer ID matching the requested ID  **Step 8:**  Display message “There are no customers registered with this ID”  **Step 9:** Position cursor in offending field and return to step 3 |
| **Conclusions** | Repair is logged into the system | |
| **Post conditions** | Repair ready for repair | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Cancel Repair

FoneFix will cancel repairs

Administrator

Customer

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Cancel Repair** | |
| **Use Case Id** | 4.4.2 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | FoneFix will cancel repairs | |
| **Preconditions** | Customer must give notice for cancelation | |
| **Trigger** |  | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts Cancel Repair customer function  **Step 3:** Administrator enters  Customer name or repairID  **Step 6:** Administrator selects which repair from customer they are to cancel | **Step 2:** Display UI  **Step 4:**  System validates repair details   * RepairID must be numeric values only   **Step 5:**Retrives summary of repairs with name from Repair file   * CustomerName * RepairID   **Step 7:** Show confirm dialog  **Step 8:**  System calculates the price of already completed work  **Step 9:** System sets repair status to “Not Fixed” and stores to Repair file  **Step 10**: Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Non-Numeric Values** |  | **Step 5**: Non-Numeric values detected in repairID  **Step 6**: Display message “These fields must contain only numbers”  **Step 7**: Position cursor in offending field and return to step 2 |
| **Conclusions** | Repair is cancelled | |
| **Post conditions** | Repair is available for collection | |
| **Business Rules** | Repair must be started | |
| **Implementation Constraints** |  | |

### Complete Repair

FoneFix will complete repairs and log them into the system

Administrator

Customer

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Complete Repair** | |
| **Use Case Id** | 4.4.3 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | FoneFix will complete repairs | |
| **Preconditions** | Repair must be in system | |
| **Trigger** |  | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts Complete Repair customer function  **Step 3:** Administration locates Repair ID and customer name  **Step 6:**Administrator selects required Repair to complete | **Step 2:** Display UI  **Step 4:**  System Validates input   * Repair ID must contain numeric characters only   **Step 5:**Retrives summary of repairs with name from Repair file   * CustomerName * RepairID   **Step 7:** Show confirm dialog  **Step 8:**  System calculates the price of work  **Step 9:** System sets repair status to “Fixed” and stores value to Repair file  **Step 10**: Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Non-Numeric Values** |  | **Step 5**: Non-Numeric values detected in ID field  **Step 6**: Display message “These fields must contain only numbers”  **Step 7**: Position cursor in offending field and return to step 2 |
| **Conclusions** | Repair is completed | |
| **Post conditions** | Repair is ready for collection | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Replenish Stock

FoneFix will replenish used stock

Administrator

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Replenish Stock** | |
| **Use Case Id** | 4.4.4 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | FoneFix will replenish used Stock | |
| **Preconditions** | Repair is completed | |
| **Trigger** |  | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:** Administrator starts Replenish Stock function  **Step 3:** Administration locates Repair ID and customer name  **Step 6:**Administrator selects required Repair to enquire from repair file | **Step 2:** Display UI  **Step 4:**  System Validates input   * Repair ID must contain numeric characters only   **Step 5:**Retrives summary of repairs with name and ID from Repair file   * Customer Name * RepairID   **Step 7:** Parts used in selected repair are added to Required Parts file  **Step 8:** Stock quantity of selected parts are updated to the Inventory file   * RequiredPartID * RequiredPartName * RequiredPartQuantity |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Non-Numeric Values** |  | **Step 5**: Non-Numeric values detected in ID field  **Step 6**: Display message “These fields must contain only numbers”  **Step 7**: Position cursor in offending field and return to step 2 |
| **Conclusions** | Stock has been replenished | |
| **Post conditions** | Repair is no longer on the System | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Carry-Out Administration

This process will contain functions that will allow the user to issue Invoices,receive payments and perform stock and sales analysis’s

### Issue Invoice

FoneFix will issue out Invoice of repair

Administrator

Customer

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Issue Invoice** | |
| **Use Case Id** | 4.5.1 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | FoneFix will issue Invoices to customers | |
| **Preconditions** | Customer must be registered | |
| **Trigger** | Repair must be completed | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:**  Administrator starts up Issue Price function  **Step 3:** Administration locates customer by name or customerID  **Step 7:** Administrator selects required customer and enters total cost of repair | **Step 2:** Display UI  **Step 4:** System validate input   * customer ID must contain numeric characters only   **Step 5:** Retrieves summary of customer with name and ID from customer file   * CustomerName * CustomerID   **Step 6:** Display UI  **Step 8:** Show confirmation Dialog  **Step 9:** InvoiceID and InvoiceAmount is saved to an invoice file  **Step 10:** Invoice is sent to the customer  **Step 11:** System adds repair price to Customer balance in Customer file  **Step 12:** Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Non-Numeric Values** |  | **Step 5**: Non-Numeric values detected in customerID  **Step 6**: Display message “These fields must contain only numbers”  **Step 7**: Position cursor in offending field and return to step 2 |
| **Conclusions** | Customer has been issued price | |
| **Post conditions** | Customer balance has been changed | |
| **Business Rules** | Reapir must be complete | |
| **Implementation Constraints** |  | |

### Receive Payment

FoneFix will receive payment for Repairs done

Administrator

Customer

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Receive Payment** | |
| **Use Case Id** | 4.5.2 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | FoneFix will receive payment from repair | |
| **Preconditions** | Repair must be finished | |
| **Trigger** | Customer needing repair | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:**  Administrator starts up receive payment function  **Step 3:** Administration locates customer by name or customerID  **Step 7:** Administrator selects required customer | **Step 2:** Display UI  **Step 4:** System validate input   * customer ID must contain numeric characters only   **Step 5:** Retrieves summary of customer with name and ID from customer file   * CustomerName * CustomerID   **Step 6:** Display UI  **Step 8:** Show confirmation Dialog  **Step 9:** System subtracts repair price from Customer balance in customer file  **Step 10:** Payments are recorded into payments file   * Amount paid * Date paid * CustomerID * Customer balance   **Step 11:** Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Non-Numeric Values** |  | **Step 5**: Non-Numeric values detected in customerID  **Step 6**: Display message “These fields must contain only numbers”  **Step 7**: Position cursor in offending field and return to step 2 |
| **Conclusions** | Payment has been recieved | |
| **Post conditions** | Customer balance has been changed | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Stock Analysis

FoneFix will perform stock analysis on system sock

Administrator

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Stock Analysis** | |
| **Use Case Id** | 4.5.3 | |
| **Priority** | medium | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | FoneFix will perform stock analysis on system stock | |
| **Preconditions** | Stock must be registered on system | |
| **Trigger** | Analysis required | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:**  Administrator starts up Stock analysis function  **Step 3:** Administration selects 2 dates to generate analysis report range | **Step 2:** Display UI  **Step 4:** System validate input   * Date format must be valid * All fields must be entered   **Step 5:** System generates a report of all stock called AllStock which is saved to the StockAnalysis file with a StockAnalysisID  **Step 6:** System displays the report with the following values of the dates chosen from Inventory file   * StockID * StockName * StockType * StockPrice * StockQuantity   **Step 7:** Show confirmation Dialog  **Step 8:** Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 5**: A blank Field Detected at step 3 with date  **Step 6**: Display message “This field must be entered”  **Step 7**: Position cursor in offending field and return to step 2 |
| **Invalid format** |  | **Step 5:** an invalid format for date entered  **Step 6:** Display message “This field has an valid date format  **Step 7:** Position cursor in offending field and return to step 2 |
| **Conclusions** | Analysis has been generated | |
| **Post conditions** | Stock Analysis is shown for selected dates | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Repair Analysis

FoneFix will perform Repair analysis on System

Administrator

<<INCLUDE>>

<<EXTENDS>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Repair Analysis** | |
| **Use Case Id** | 4.5.3 | |
| **Priority** | medium | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | FoneFix will perform Repair analysis on System | |
| **Preconditions** | Repairs must be logged on system | |
| **Trigger** | Analysis required | |
| **Expected Scenario** | **Administrator** | **System** |
|  | **Step 1:**  Administrator starts up Repair analysis function  **Step 3:** Administration selects 2 dates to generate analysis report range | **Step 2:** Display UI  **Step 4:** System validate input   * Date format must be valid * All fields must be entered   **Step 5:** System generates a report of all repairs called All\_Repair which is saved to the RepairAnalysis file with a RepairAnalysisID  **Step 6:** System displays the report with the following values of the dates chosen from repair file   * RepairID * PhoneSerialNum * Password * CustomerName * CustomerID * Description   **Step 7:** Show confirmation Dialog  **Step 8:** Clear UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 5**: A blank Field Detected at step 3 with date  **Step 6**: Display message “This field must be entered”  **Step 7**: Position cursor in offending field and return to step 2 |
| **Invalid format** |  | **Step 5:** an invalid format for date entered  **Step 6:** Display message “This field has an valid date format  **Step 7:** Position cursor in offending field and return to step 2 |
| **Conclusions** | Analysis has been generated | |
| **Post conditions** | Repair Analysis is shown for selected dates | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD

Request Repair

Customer

FoneFixSYS

Repair Confirmation

## Level-1 DFD

P3

Process Repair

Customer

Saves Repair details to file

Takes in customer details

Gets customer details for repair

P1

Manage

Customers

Repair details for each customer

D1

Customer file

D2

Repair file

Checks all used parts

Saves customer details to file

D4

Inventory file

D1

Customer file

Receive customer data

Saves stock details to file

Gather Stock data

P2

Manage

Stock

P4

Carry-Out-Administration

Gather repair data

Receive payments

Issue invoice

D2

Repair file

Customer

## Level-2 DFD (Process P1: Manage Customers)

P1.2

Update Customer

P1.1

Add Customer

Current prope

Gather current Customer details

Collect Customer data

Store Customer details in file

D1

Customer file

Stores updated data in file

Customer status set to deactivated

Gather current Customer details

Customer

P1.3

De-Register Customer

## Level-2 DFD (Manage Stock)

P2.2

Update Stock

P2.1

New Stock

Current prope

Gather Stock details

Store Stock details in file

Stores updated Stock data in file

D4

Inventory file

Admin makes enquirys on selected stock

Gather Stock details

P2.3

Stock Enquiry

## Level-2 DFD (Process Repair)

P3.2

Cancel Repair

P3.1

Log Repair

Current prope

Gather current Repair details

D4

Repair file

Sets repair status to deactivated

Store Repair details in file

Current prope

D4

Inventory file

Sets repair status to complete

Gather current Repair details

P3.3

Complete Repair

Stock quantitys are updated

Gather Repair details

P3.4

Replenish Stock

Parts used in repair are added to file

D4

Required Parts file

## Level-2 DFD (Carry-Out-Administration)

Current prope

Current prope

Invoice

Customer

Payments

Current prope

P4.2

Receive Payment

Customer balance updated

P4.1

Issue Invoice

Customer balance updated

D4

Customer file

Current prope

Customer details

Payments recorded

D4

Payments File

D4

Invoice file

Invoice saved to file

Stock details

D4

Inventory file

D4

Stock Analysis file

Analysis data

P4.3

Stock Analysis

P4.4

Repair Analysis

Repair details

Analysis data

D4

Repair Analysis file

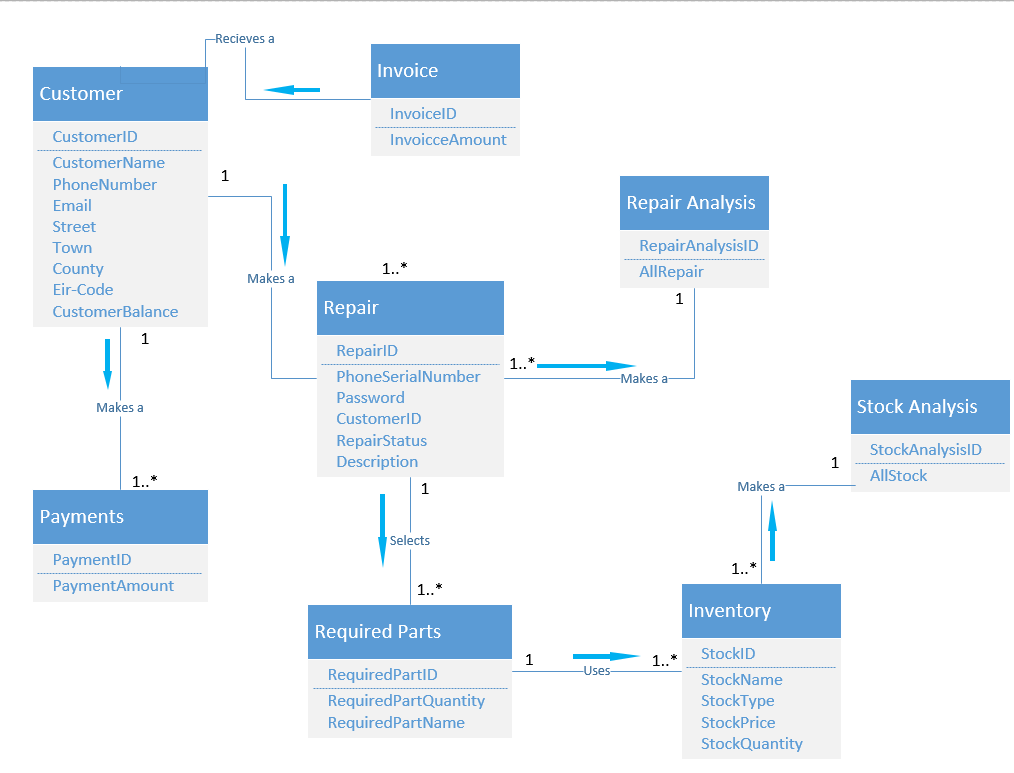
D4

Repair file

# Data Model (Class Diagram)

Brief introduction……

## Class Diagram



## Relational Schema

Customer (**CustomerID,** CustomerName,PhoneNumber,Email,Street,Town,County,Eir-Code, CustomerBalance, RepairID, PaymentID, InvoiceID)

Payments (**PaymentID**, PaymentAmount, CustomerID)

Repair (**RepairID**, PhoneSerialNumber,Password,CustomerID,RepairStatus,Description, CustomerID)

RequiredParts (**RequiredPartID,** RequiredPartQuantity, RequiredPartName, RepairID, StockID)

Inventory (**StockID**, StockName, StockType, StockPrice, StockQuantity)

RepairAnalysis (**RepairAnalysisID,** AllRepair, RepairID)

StockAnalysis (**StockanalysisID,** AllStock, StockID)

## Database Schema

**SCHEMA: FoneFixSYS**

**Relation: Customers:**

Attributes:

CustomerID numeric (5) NOT NULL auto\_increment,

CustomerName char (30) NOT NULL,

PhoneNumber varchar (15) NOT NULL,

Email varchar (20) NOT NULL,

Street char (15) NOT NULL,

Town char (15) NOT NULL,

County char (15) NOT NULL,

Eir-Code char (15) NOT NULL,

CustomerBalance numeric (10) NOT NULL,

RepairID numeric (5) NOT NULL,

PaymentID numeric (5) NOT NULL

InvoiceID numeric (5) NOT NULL

**PRIMARY KEY:** CustomerID

**FOREIGN KEY:** RepairID **REFERENCES** Repair table

**FOREIGN KEY:** PaymentID **REFERENCES** Payments table

**FOREIGN KEY:** InvoiceID **REFERENCES** Invoice table

**Relation: Payments:**

Attributes:

PaymentID numeric (5) NOT NULL auto\_increment,

PaymentAmount numeric (10) NOT NULL,

CustomerID numeric (5) NOT NULL,

**PRIMARY KEY:** PaymentID

**FOREIGN KEY:** CustomerID **REFERENCES** Customer table

**Relation: Repair:**

Attributes:

RepairID numeric (5) NOT NULL auto\_increment,

PhoneSerialNumber numeric (20) NOT NULL,

Password char (30) NOT NULL,

RepairStatus boolean (true/false) NOT NULL,

Description char (30) NOT NULL,

CustomerID numeric (5) NOT NULL,

**PRIMARY KEY:** RepairID

**FOREIGN KEY:** CustomerID **REFERENCES** Customer table

**Relation: Required Parts:**

Attributes:

RequiredPartID numeric (5) NOT NULL auto\_increment,

RequiredPartQuantity numeric (50) NOT NULL,

RequiredPartName char (30) NOT NULL,

RepairID numeric (5) NOT NULL,

StockID numeric (5) NOT NULL,

**PRIMARY KEY:** RequiredPartID

**FOREIGN KEY:** RepairID **REFERENCES** Repair table

**FOREIGN KEY:** StockID **REFERENCES** Inventory table

**Relation: Inventory:**

Attributes:

StockID numeric (5) NOT NULL auto\_increment,

StockName char (30) NOT NULL,

StockType char (30) NOT NULL,

StockPrice numeric (10) NOT NULL,

StockQuantity numeric (50) NOT NULL,

**PRIMARY KEY:** StockID

**Relation: RepairAnalysis:**

Attributes:

RepairAnalysisID numeric (5) NOT NULL auto\_increment,

AllRepair char (50) NOT NULL,

RepairID numeric (5) NOT NULL,

**PRIMARY KEY:** RepairAnalysisID

**FOREIGN KEY:** RepairID **REFERENCES** Repair table

**Relation: StockAnalysis:**

Attributes:

StockanalysisID numeric (5) NOT NULL auto\_increment,

AllStock char (50) NOT NULL,

StockID numeric (5) NOT NULL,

**PRIMARY KEY:** StockanalysisID

**FOREIGN KEY:** StockID **REFERENCES** Inventory table

# Conclusion

The goal of my project was to create a system in which an administrator would be allowed to add repairs and track them on a system. The use case and diagrams were drawn to best describe the system at work and the interactions between the functions and entities

A prototype was created using visual studio to show a working example of the system and the actions required in it. The protoype had to be easy to navigate and hv suitable validation which clearly marks in places that were incorrectly entered.

Once the ability to save data is added to sstem should be able to perform as planned with little to know trouble as the majority of the code for validation and linking is already there, and shouldn’t prove too difficult in the future.

# Appendices

## Appendix A – Title

## Appendix B – Title

Might include:

* **Lookup / Reference tables**
* **Sample reports / Listings**